GAIT ANIMATION AND ANALYSIS FOR BIOMECHANICALLY-ARTICULATED SKELETONS

by

ERIC DAVID WILLS

A DISSERTATION

Presented to the Department of Computer and Information Science
and the Graduate School of the University of Oregon
in partial fulfillment of the requirements
for the degree of
Doctor of Philosophy

March 2008

University of Oregon Graduate School

Confirmation of Approval and Acceptance of Dissertation prepared by:

Eric Wills

Title:

"Gait Animation and Analysis for Biomechanically-Articulated Skeletons"

This dissertation has been accepted and approved in partial fulfillment of the requirements for the Doctor of Philosophy degree in the Department of Computer & Information Science by:

Kent Stevens, Chairperson, Computer & Information Science
Arthur Farley, Member, Computer & Information Science
Christopher Wilson, Member, Computer & Information Science
Gregory Retallack, Outside Member, Geological Sciences

and Richard Linton, Vice President for Research and Graduate Studies/Dean of the Graduate School for the University of Oregon.

March 22, 2008

Original approval signatures are on file with the Graduate School and the University of Oregon Libraries.

An Abstract of the Dissertation of

Eric David Wills for the degree of Doctor of Philosophy

in the Department of Computer and Information Science to be taken March 2008

Title: GAIT ANIMATION AND ANALYSIS FOR BIOMECHANICALLY-ARTICULATED SKELETONS

Approved: _____

Kent A. Stevens

Digital three-dimensional (3D) models are useful for biomechanical analysis because they can be interactively visualized and manipulated. Synthesizing and analyzing animal locomotion with these models, however, is difficult due to the large number of joints in a fully articulated skeleton, the complexity of the individual joints, and the huge space of possible configurations, or poses, of the skeleton taken as a whole. A joint may be capable of several biological movements, each represented by a degree of freedom (DOF). A quadrupedal model may require up to 100 DOFs to represent the limbs and trunk segments only, resulting in extremely large spaces of possible body configurations. New methods are presented here that allow limbs with any number of biomechanical DOFs to be kinematically exercised and mapped into a visualization space. The spaces corresponding to the ranges of motion of the left and right limbs are automatically intersected and pruned using biological and locomotion constraints. Hind

and fore spaces are similarly constrained so that Genetic Algorithms (GAs) can be used to quickly find smooth, and therefore plausible, kinematic quadrupedal locomotion paths through the spaces. Gaits generated for generic dog and reptile models are compared to published gait data to determine the viability of kinematics-only gait generation and analysis; gaits generated for *Apatosaurus*, *Triceratops*, and *Tyrannosaurus* dinosaur models are then compared to those generated for the extant animals. These methods are used for several case studies across the models including: isolating scapulothorax and shoulder joint functionality during locomotion, determining optimal ankle heights for locomotion, and evaluating the effect of limb phase parameters on quadrupedal locomotion.

CURRICULUM VITAE

NAME OF AUTHOR: Eric David Wills

PLACE OF BIRTH: Santa Monica, CA, USA

DATE OF BIRTH: November 19, 1977

GRADUATE AND UNDERGRADUATE SCHOOLS ATTENDED:

University of Oregon, Eugene, OR USA

DEGREES AWARDED:

Doctor of Philosophy in Computer and Information Science, 2008,
University of Oregon
Master of Science in Computer and Information Science, 2002,
University of Oregon
Bachelor of Science in Computer and Information Science/Mathematics, 2000,
University of Oregon

AREAS OF SPECIAL INTEREST:

Evolutionary Locomotion Synthesis
Computational Biomechanics
Computer Graphics and Animation
Parallel and Distributed Computing

PROFESSIONAL EXPERIENCE:

Principal Engineer, Kaibridge, Inc., 5 years
Software Engineer, Integrated Measurement Systems, 3 years

GRANTS, AWARDS AND HONORS:

Graduate Teaching Fellow of the Year, Department of Computer and Information Science, University of Oregon, 2005

Inducted into Upsilon Pi Epsilon, International Honor Society for the Computer and Information Disciplines, 2003

Departmental Honors for completion of an honors thesis, Department of Computer and Information Science, University of Oregon, 2000

PUBLICATIONS:

Stevens, K. A., Larson, P., Wills, E. D., & Andersen, A. (2008). Rex, sit: Digital modeling of Tyrannosaurus rex at rest. In P. Larson & K. Carpenter (Eds.), *Tyrannosaurus rex: The tyrant king*. Bloomington, IN: Indiana University Press.

Stevens, K. A., & Wills, E. D. (2007). Kinematic constraints on the reconstruction of Dinosaur gaits. *Proceedings of the Symposium on Vertebrate Paleontology and Comparative Anatomy, 55*, 54.

Wills, E. D., & Stevens, K. A. (2007). Computational explorations of quadrupedal locomotion in Dinosaurs based on modern analogs. *Proceedings of the American Society of Biomechanics Northwest Symposium, 3*, 18.

Wills, E. D., & Stevens, K. A. (2007). Isolating functional degrees of freedom in limbs during locomotion. *Proceedings of the Symposium on Vertebrate Paleontology and Comparative Anatomy, 55*, 46.

Stevens, K. A., Wills, E. D., & Ernst, S. W. (2006). 3D Visualization of allometric changes in whole skeletons: Posture, proportion, and range of motion. *Journal of Vertebrate Paleontology, 26*(Suppl. 3A), 128.

Stevens, K. A., Parrish, J. M., & Wills, E. D. (2005). Ontogenetic changes within the tyrannosaurid skeleton. In R. Scherer (Chair), *Tyrannosaur Symposium 2005.* Symposium conducted at the Burpee Museum of Natural History, Rockford, IL.

Stevens, K. A., & Wills, E. D. (2001). Gracile versus robust cervical vertebral designs in sauropods. *Journal of Vertebrate Paleontology, 21*(Suppl. 3A), 104.

ACKNOWLEDGEMENTS

I would like to express my sincere appreciation to my family and friends for their support and advice throughout this process. Specifically, I would like to thank my caring parents Dr. David and Laurie Wills, my dedicated and entertaining sister Dr. Megan Wills Kullnat, and especially my loving, understanding, and beautiful wife Shelby Stair Wills. I would like to thank my committee for their guidance and expertise, especially Professor Kent A. Stevens, whose focused attention and patience over the last decade have shaped me into the professional that I am today. This work was supported in part by National Science Foundation grant 0093929.

To my family and friends.

TABLE OF CONTENTS

Chapter Page

LIST OF FIGURES

Figure Page

Figure Page

Figure Page

Figure Page

Figure Page

LIST OF TABLES

Table Page

Table Page

CHAPTER I

INTRODUCTION

Animation is the process of creating an illusion of movement through the presentation of a discrete sequence of images. The perceived movement of an animation is outlined by key images, or frames, that represent important events along the animation timeline. In creating an animation of a moving animal, events such as feet touching or lifting off the ground are identified as keyframes. From the context of reconstructing locomotion for extinct animals, these hand and foot ground contact events can be correlated with fossilized trackway data to provide a basis for possible walk cycles. Hand/foot ground contact events, however, only constrain the animal's external interactions with the environment; it is the kinematics of the bones and joints that constrain, at least in part, the movements of the animal's body during locomotion. It is the goal of this dissertation to provide new methods for synthesizing and analyzing gait animations by exploring the kinematics of an animal's limbs while they are in contact with the ground.

Animations that accurately depict animal locomotion are useful for gait analysis. Such animations provide a description of the movement of an animal's limb, trunk, neck,

head, and tail with respect to the gait cycle timeline. Muybridge (1887) was the first to investigate animal locomotion using photographic sequences of animals engaged in various gaits. Image sequences were created using a linear array of cameras to record the movements of a passing animal. These image sequences provide a visual description of limb movements and timings from the fixed viewpoint of the cameras.

Photographic animations are ultimately limited for locomotion analysis purposes by the inability to directly manipulate the models, except by navigating along the gait cycle timeline. Two-dimensional (2D) animation sequences are further limited by their inherent fixed viewpoint; an investigator cannot view the model from an arbitrary perspective to best observe points of interest. One goal of the research presented in this paper is the exploration of dinosaur locomotion, for which photographic sequences are of course not available.

Digital three-dimensional (3D) skeletal models can be interactively manipulated and visualized, making them useful for locomotion analysis. Such models can be based on fossilized bones, allowing investigations regarding the movements of extinct animals. To provide confidence in the results of simulated movements, the models must be accurate and capture sufficient complexity of the joints and articular surfaces. The necessary complexity of these models dictates an enormous amount of effort in building and posing the models. Digital models can be animated by using motion capture data (Delaney, 1998) to pose the skeleton. While this is an important technique for creating animations of extant animals for research and other purposes, motion capture is not an option for extinct animals.

Many joints are capable of more than one kind of movement. These movements are often given physiological terms such as flexion/extension and adduction/abduction. Each of these physiological movements will be regarded as a degree of freedom (DOF). The shoulder joint, for instance, has three such DOF, namely flexion/extension, adduction/abduction, and internal/external rotation (i.e., rotation about the long axis of the upper arm). An entire limb, therefore, may have ten or more DOFs, so a quadruped model may easily contain 100 or more DOFs distributed throughout the limbs, the girdles that attach the limbs to the trunk, and the trunk.

While biological joints are capable of effectively continuous movements, it is a practical necessity to discretize the movements of a digital model for computational analysis (e.g., into 100 steps for each DOF, or some other manageable sampling method). The space of discrete skeletal poses grows exponentially with the number of DOFs and the number of samples per DOF; the sheer magnitude of these spaces makes posing a challenge and searching these spaces intractable.

The space of all kinematic poses for a biologically-accurate skeletal model is very large and may contain poses that are not possible with respect to an animal's myology. For this reason, investigators often build musculoskeletal models that are limited in terms of posing by the muscles, ligaments, and tendons of the model (Hutchinson, Anderson, Blemker, and Delp, 2005; Sellers, Dennis, Wang, and Crompton, 2004; Sellers and Manning, 2007). Such musculoskeletal models are driven by specifying muscle activation values. These activation values are interpreted by physics simulations, in which muscle, gravitational, and contact moments are used to update joint angles. Searching the space of

possible muscle activation values is computationally intensive due to the necessary use of physics simulations for each evaluation. Construction of these musculoskeletal models is laborious, even more so than construction of the original model because several muscles, ligaments, and tendons affect each joint. Furthermore, musculoskeletal models are difficult to accurately reconstruct for extinct animals because muscle dimensions and other characteristics are largely unknown and must be based on modern analogues, if available (Hutchinson et al., 2005).

Sequences of poses representing plausible gaits can be presumed to exist somewhere within the posing spaces of the 3D skeletal models, assuming that joint Ranges of Motion (ROMs) are accurately represented. In this way, limb and trunk osteologies provide a template for movement upon which the musculature acts. Locomotion can therefore be studied using only the kinematics of the bones and joints of a skeleton, but methods are needed for navigating the massive kinematic search spaces. It is the purpose of this thesis to explore this hypothesis by presenting new methods that search kinematic posing spaces for plausible locomotion and presenting the results of locomotion studies conducted using these methods.

New techniques will be presented here that allow automatic gait generation and analysis of biomechanically-articulated skeletons using Genetic Algorithm Gait Analysis (GAGA) methods. GAGA methods allow each limb to be exercised to determine its potential contribution to locomotion. There is no theoretical limit to the number of DOFs allowed per limb, or in the skeleton as a whole (although computer memory and processing power ultimately limit the fidelity of the limb explorations). All DOFs can be modeled to

represent any biologically possible movement at each joint (i.e., they are not limited to idealized primitives).

In animals, limbs do not act in isolation; they function as part of a whole organism. As such, there are times during locomotion when a limb's pose is constrained by what is happening elsewhere in the body. During walking gaits, there is a time when both limbs of a bipedal system are in contact with the ground. With both limbs planted on the ground, there is a limited number of poses for each limb that do not cause the limbs to separate at their shared root. GAGA methods utilize this constraint, along with a bilateral symmetry constraint, to dramatically prune the size of the limb's posing space, leaving a space relevant to locomotion. Quadrupedal gaits are handled similarly, with an additional trunk constraint that further prunes hindlimb and forelimb spaces so that they are relevant to quadrupedal locomotion.

Genetic Algorithms (GAs) are then used to find smooth and therefore plausible gaits through the constrained spaces. The GAs use a highly-optimized candidate representation and fitness function that guarantees every generated gait will at least move the animal forward by a specified stride length. The GAs then search for gaits that are smooth in terms of body pitching, yawing, rolling, and lateral/vertical displacement. The GA also attempts to minimize unnecessary angular excursions at the joints. These fitness terms allow gaits to be analyzed in terms of the locomotion goals of real animals.

Forward walking gaits were generated for two extant models (i.e., a generic dog and reptile) and for models of three dinosaurs (i.e., *Apatosaurus*, *Triceratops*, and *Tyrannosaurus*). The walking gaits generated for the extant models were compared

qualitatively with published data and serve as controls for verifying that the gaits generated using GAGA methods match real-world animal gaits in terms of joint and limb functionality. The comparison of gaits generated using GAGA methods to real-world analogues provides confidence in the gaits generated for the extinct dinosaur models and the gait analysis of these models.

The gait observables used by the fitness function to define an optimal gait can be used to quantitatively analyze gaits. The dog, *Apatosaurus*, and *Triceratops* models were analyzed to determine the contributions of the scapulothorax and shoulder joints to locomotion. Also, the dog, *Apatosaurus*, *Triceratops*, and *Tyrannosaurus* were analyzed to determine optimal ankle height based on pes flexibility. Finally, the dog, reptile, *Apatosaurus*, and *Triceratops*, were analyzed to determine the effect of gait phase parameters on quadrupedal locomotion.

Due to the use of kinematics only (i.e., no dynamic simulations are used) and the highly-optimized GAs, GAGA methods are able to automatically generate forward walking gaits for quadrupedal models in under five minutes on a consumer laptop (as of the time of this printing). Comparable methods use musculoskeletal models and dynamics simulation to automatically generate bipedal gaits, but the process can take weeks on supercomputer clusters (Sellers and Manning, 2007). Other methods are able to simulate quadrupedal locomotion, but only on simple spring-based models consisting of a rigid trunk, a hip/shoulder joint per limb, and the remainder of each limb modeled by a spring (Herr, Huang, and McMahon, 2002). In fairness, these methods are able to generate gaits with

aerial phases, while the GAGA methods are limited to walking gaits due to the lack of dynamics. The remainder of this dissertation is organized as follows:

Chapter II contains a review of methods related to the automated generation and analysis of locomotion. Automatic character animation techniques first provided simple methods to automatically generate animation keyframes. Next, a review of Evolutionary Algorithms (EAs) will be presented before a review of methods that utilize EAs to automatically generate locomotion. Next, a survey will enumerate current state-of-the-art methods for evaluating gaits, especially with respect to extinct animals.

Chapter III will present the GAGA methods and techniques. Methods for defining joints and DOFs will first be presented. Next, limbs will be exercised to determine configuration spaces and those spaces will be pruned based on biological and locomotion constraints so that they can be efficiently searched using GAs. Also, trunk-based constraints will be utilized to further prune the spaces for quadrupedal locomotion. The use of pipelines will be demonstrated to maximize the reuse of data between discrete operation for both bipedal and quadrupedal gait generation. Finally, specific data will be presented on the extant and extinct models used in later studies.

Chapter IV will present data from sensitivity analyses and studies conducted using GAGA methods. First, an exploration of the GA genetic parameters will demonstrate the process of tuning the GAs and evaluate their performance. Sensitivity analyses will then show the robustness of the GAGA methods under changes to the parameters used to build configuration spaces and generate gaits. Next, qualitative gait analysis will show that the gaits generated for extant animals closely match published gaits. Finally, quantitative gait

analysis will demonstrate that GAGA methods can be used for careful gait analysis and present the results of three specific case studies.

Chapter V contains a discussion of the benefits and limitations of the GAGA methods. The significance of the case study results from Chapter IV will also be discussed. The paper will conclude with an ending summary in Chapter VI. The Appendices contain geometric data for the five models described in this paper, pseudocode for some of the presented algorithms, and a glossary of terms and acronyms used in this paper. Finally, the Bibliography lists all references cited in the body of this paper.

CHAPTER II

BACKGROUND AND RELATED WORK

In this chapter, background and related work will be presented that provide a foundation for the new methods and analysis presented in later chapters. These related works come from a variety of areas, including early methods for automatically generating character animation, the use of computational search techniques to evolve locomotion, and computational methods for analyzing gaits using digital 3D models. Ideas from each of these areas provided insight and inspiration for the new methods presented later in this paper.

First, global optimization techniques allow automatic generation of animation by specifying high-level goals, but these techniques do not scale well to complex models. Techniques using localized controllers scale well to more complex models, but require a great amount of manual effort to coordinate joint activity. Finally, fragment composition techniques allow varied locomotion by blending simple locomotion sequences, but often compromise physical and biomechanical realism.

Next, a background of evolutionary search techniques (i.e., EA) and basic applications will be presented. EAs are particularly useful in determining good solutions for complex optimization and synchronization problems. EAs will not always find the optimal solution to a problem, but will almost always find a solution that is nearly optimal, as will be discussed. For locomotion synthesis, quickly finding near-optimal solutions allows a character to interactively respond to stimuli while maintaining biomechanical accuracy.

Next, evolutionary locomotion synthesis techniques will be presented. These techniques combine automatic character animation and EA methods to produce locomotion. Successful applications from several areas will be presented. Methods for computationally evolving controllers have been used to coordinate joint behavior for locomotion of several human and animal models. Artificial Life (ALife) methods have been used to simultaneously evolve creature minds and bodies, creating strange new creatures capable of unique methods of locomotion. Also, robotics techniques have been utilized for robot locomotive learning.

Next, modern digital 3D gait analysis methods and tools will be presented. Complex musculoskeletal models allow accurate biomechanical evaluation, but require laborious definition of the model and assumptions when modeling extinct animals. Static and dynamic constraints can be useful in pruning the space of limb configurations for locomotion. Finally, EA techniques can be used to find optimal gaits, providing insight into the limb movements and maximum speeds of extinct animals.

Automatic Character Animation

Modern locomotion synthesis techniques are based on earlier ideas for automatically generating physically-accurate animation sequences. Early methods relied upon global optimization of an animation sequence to produce physically-accurate animation. These methods were the first used to generate such animations, but did not scale well to complex models. Another approach involved defining localized physical controllers to enforce the physically-accurate behavior of individual joints. These methods localized the force and torque optimization problems, increasing both scalability and the complexity needed to coordinate controllers. Finally, previously-generated animation sequences were blended and concatenated at animation time. The resulting motion sequences were generated inexpensively with no user interaction, but sometimes at the expense of physical and/or biomechanical accuracy.

Global Optimization

An animation sequence can be specified by defining a set of objects and the forces and torques applied to those objects over time. By applying these forces and torques, a physical simulation can produce an animation sequence. Manually specifying forces and torques to satisfy an animation goal is intractable for all but the simplest goals. The necessary forces and torques can, however, be automatically generated by optimizing the forces and torques required to satisfy high-level animation goals. As the number of animation DOFs grows, optimization soon becomes computationally prohibitive.

Optimization tractability can be maintained for complex models by varying time step resolution or reducing the number of animation DOFs, but physical and biomechanical accuracy may suffer as a result.

Witkin and Kass (1988) introduced Spacetime Constraints as a new method for automatically generating high-level animation. Using this method, the animator specified high-level goals for what a character would do and how they would do it. Specifically, the animator would constrain what the character has to do, how the action should be performed, the physical structure of the character, and the resources available to the character. Optimization techniques and physical simulation were used to generate the low-level animation details.

The Spacetime Constraints for each physical object were specified by typing LISP expressions into a Graphical User Interface (GUI) function box. Expressions were required for the mass and inertial parameters, the optimization criteria, and kinetic energy with respect to linear and angular velocity. The kinetic energy expression was used during optimization to evaluate the effect of forces and torques on the object. Lines could be drawn between boxes to indicate hierarchical relationships between objects. The optimization process would then solve for the time-dependent force and torque functions that best satisfied the minimization and maximization criteria for each object.

Spacetime Constraints were shown to effectively produce jumping animations for Pixar's *Luxo, Jr.* lamp. Three 1-DOF joints were used to control the posture and motion of the lamp. The authors were able to produce a variety of animations by adjusting the physical parameters and optimization criteria. For example, increasing the weight of the

lamp's base resulted in more vertical squash prior to and following a jump. Figure 1 shows frames of a jumping animation for the lamp, including the realistic squash and stretch that is automatically generated using these methods.

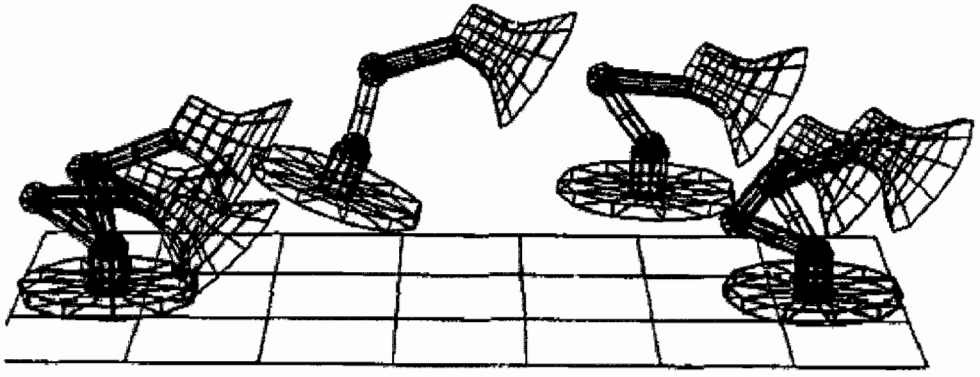


Figure 1. The *Luxo Jr.* lamp jumping.

Note. From "Spacetime constraints" by A. Witkin and M. Kass, 1988, *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, p. 167.

Specifying the necessary constraint and kinetic expressions became prohibitively expensive as model complexity grew. The expressions required to describe the lamp's link segments were quite complex even though the lamp utilized only three 1-DOF joints. The search space for optimal force and torque functions increases exponentially with the number of interacting objects, so optimization becomes prohibitively expensive as object hierarchies become more complex.

Another problem with the Spacetime Constraints method was that the animator had to specify the number of discrete time steps used to represent force and torque functions. This restriction could have led to functions that were undersampled and/or oversampled at places along the animation timeline. Undersampling of the function could have resulted in animations that did not satisfy constraints. Oversampling of the function would unnecessarily increase the complexity of the search space.

Liu, Gortler, and Cohen (1994) suggested the use of variable-rate sampling for force and torque functions. A hierarchical wavelet was used to represent the torque function for each joint DOF. Hierarchical wavelets are similar to hierarchical B-splines in that segments of the spline can be adaptively refined to add or remove resolution. Hierarchical layers of B-splines represent more detailed but redundant information. Hierarchical layers of wavelets represent differences between layers, adding more detail to each layer without reproducing data.

By using a wavelet to represent torque functions, few samples were needed during periods of joint inactivity. Conversely, more samples were allocated during high activity periods. The use of wavelets dramatically reduced optimization time when compared to analogue runs using fixed sampling. The speedup suggests that, at least in their animations, there are significant periods of time when joint inactivity can be exploited to reduce the optimization search space. Variable-rate sampling increased the applicability of Spacetime Constraints only slightly; optimization of models with a large number of joint DOFs remained intractable.

Ngo and Marks (1993) attempted to reduce user interaction with Spacetime Constraints by employing Genetic Programming (GP) to generate animation. GP will be discussed in more detail later in this paper. The primary benefit of the GP approach is that it eliminated the need for the animator to specify complex kinetic expressions for each body. Instead, trajectories were encoded as behaviors that were automatically evolved to satisfy animation constraints.

Behaviors were represented as Stimulus Response (SR) pairs. Stimuli were triggered using input from a set of sensors. Four sensor types were utilized: angle sensors, tactile sensors that measured ground contact forces, kinesthetic sensors that measured vertical velocity of the Center of Mass (COM), and position sensors which monitored the vertical position of the COM. Each sensor was defined by its type, a stimulus center, and a stimulus extent. Responses consisted of a set of target joint angles and a duration parameter. A critically damped equation of motion was used to ensure smooth motion during response and when switching between SR pairs.

Only one SR pair was active at a time. A new pair was selected during each simulation step. Pair selection was performed by first normalizing each stimulus input to within its stimulus extent range. The pair with the smallest sum difference between normalized sensor inputs and stimulus centers was selected for activation. The pair would remain active until another SR pair was identified with a smaller sum difference between normalized sensor inputs and stimulus centers.

Sets of 10 SR pairs were evolved to produce 2D articulated figure motion. Motions were evolved for basic stick figure walking, skipping, shuffling, and jumping. Figure 2

shows an evolved walking motion. Criteria such as maximum horizontal distance and maximum vertical height were used to drive the evolution process. The global SR pairs worked well on a low-DOF stick figure, but would not easily scale to more complicated articulated figures. Similar to the Spacetime Constraints method, it would become increasingly difficult for the optimization scheme to satisfy animation goals by globally controlling all animation DOFs.

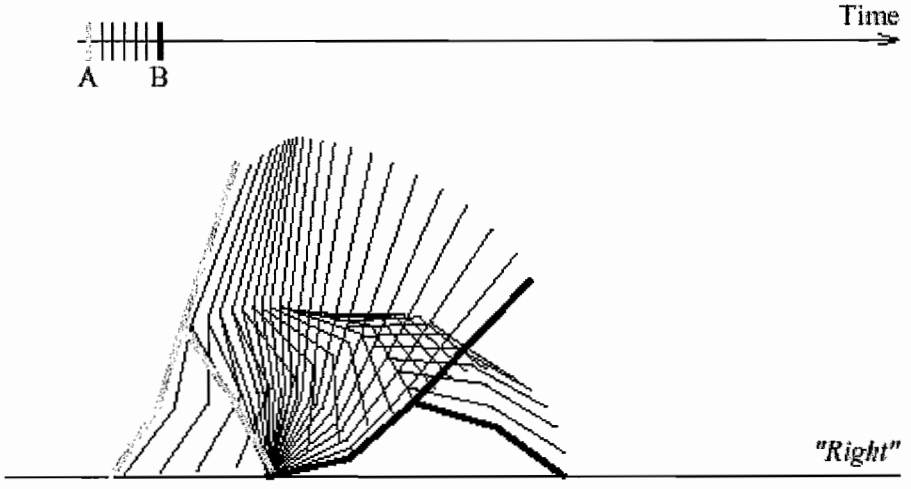


Figure 2. SR-based walking.

Note. From "Spacetime constraints revisited" by J. T. Ngo and J. Marks, 1993, *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, p. 348.

Torkos and van de Panne (1998) used global trajectory optimization and trackway data to synthesize quadruped locomotion. To combat previous optimization complexity

problems, trajectory optimization was restricted to just two key mass points, one at the pelvic girdle and one between the shoulder girdles. Figure 3 shows these two key mass points. The trackway print locations and timings, along with a nominal limb length for the fore and hind limbs, provided enough data to optimize the motion of the animal's body. Motion of the cervical, dorsal, caudal, and limb joints were calculated based on the animal's trajectory.

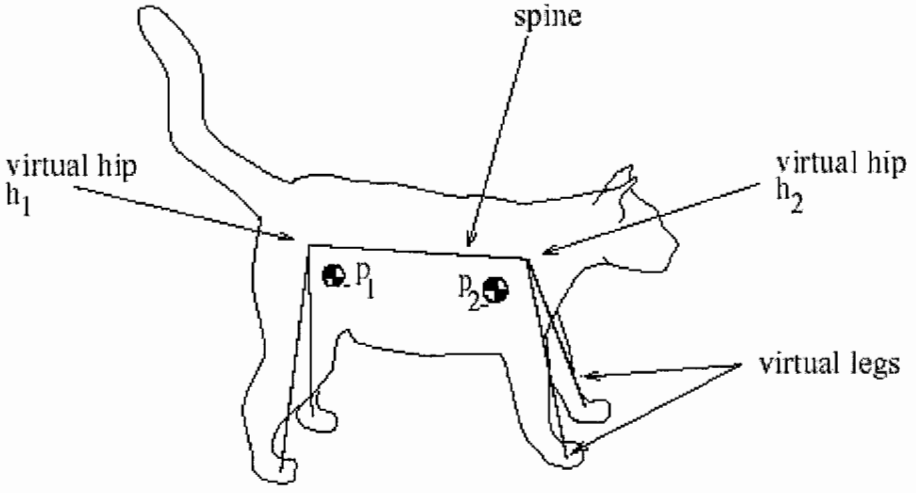


Figure 3. Key mass points.

Note. From "Footprint--based Quadruped Motion Synthesis" by N. Torkos and M. van de Panne, 1998, *Graphics Interface*, p. 156.

The animal's dorsal vertebral column was treated as a pair of springs, one spring between each key mass point and the animal's COM. The dorsal joints were configured using Inverse Kinematics (IK) such that they satisfied the spring constraints. The support phase of each limb was also configured using IK such that the manus (i.e., hand) and pes (i.e., foot) were located at the manus and pes print locations. For a review of IK, specifically for anthropomorphic limbs, see Tolani, Goswami, and Badler (2000). The suspended phase of each limb was determined by optimizing the path of the end effecter (i.e., manus or pes) to seek the next print while avoiding collisions with the ground and other limbs. The cervical and caudal joints were handled independently for non-locomotion purposes.

The use of simple IK to determine limb joint configurations requires several assumptions. The primary assumption is that the joints must be simple hinge or ball and socket joints. These simple joints lend themselves well to IK due to their fixed centers of rotation, but are not biologically accurate. IK for biologically-accurate joints is an intractable problem that can only be handled by case-specific approximation. Also, IK is not intrinsically physically accurate, although pinning the end effectors with physically-accurate constraints typically produces visually-acceptable results.

These methods were used to successfully generate quadruped locomotion that followed a trackway. The body trajectories were preplanned and IK was used to keep the manus and pes coincident with trackway prints, so the limbs were not directly used to produce locomotion. This may have resulted in a "marionette" effect, in which the body is moving and the limbs appear to be along for the ride instead. The optimization process

may have also caused to body and limbs to enter configurations in which the animal would appear visually uncomfortable. The idea of path planning for trackway following is good, but the limbs should be controlled such that they at least appear to drive the animal along the path.

Chung and Hahn (1999) used a similar IK approach to simulate human walking. Instead of utilizing trackway data, pes print locations were generated from a general path description. The print-planning algorithm compensated for turning, obstacles, and changes in grade by altering step length. Print locations were always planned two steps ahead so that the character would show anticipation. The print locations, along with a path for the pelvis to follow, pinned the origin and end effecter of the limb, leaving knee and hip configurations to be solved by IK.

The path of the pelvis was described using a parametric spline. Two control points were used to describe pelvis motion during each step. The first control point represented the pelvis position when the supporting pes was directly under the pelvis. The height of the pelvis at the time was determined by the amount of knee flexion in the supporting leg. The second control point represented the pelvis position when both limbs were supporting (called *dual support*). The pelvis position during dual support was determined by minimizing the angular accelerations caused by IK to achieve dual support. The clearance height of the swinging leg was also governed by a spline. Control points were initially placed at the pes print locations and at a height such that the pes would clear the ground. Additional control points were added so that the pes would clear obstacles or uneven terrain. Figure 4 shows a generated walk on uneven terrain.

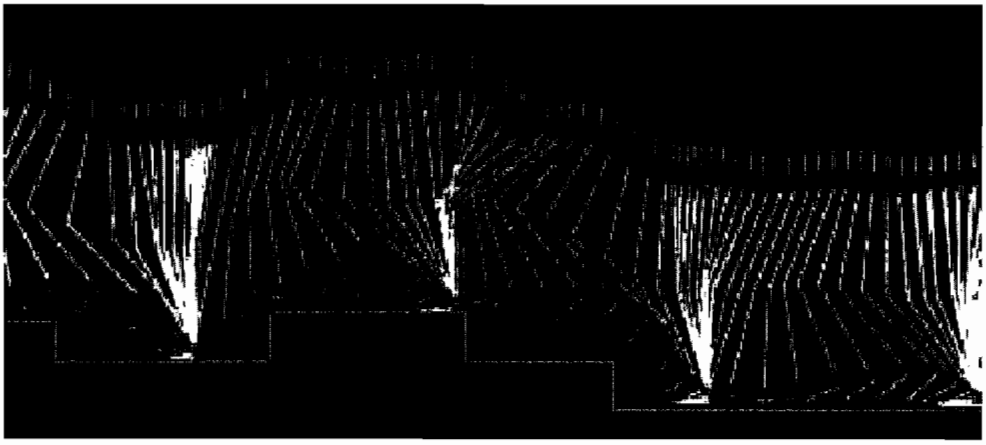


Figure 4. Walking on uneven terrain.

Note. From "Animation of Human Walking in Virtual Environments" by S. Chung and J. K. Hahn, 1999, *Proceedings of Computer Animation, 99,* p. 13.

These methods produced forward locomotion on uneven terrain, but the limbs may again have had a "marionette" look due to the use of IK. A hybrid of global and local optimization techniques were used for obstacle avoidance. The trackway print locations were precomputed to compensate for terrain and obstacles. The pelvis and pes optimizations, however, were run at animation time to determine walking behavior for the current step. The animation-time optimization of the pelvis location and pes clearance height is important for walking on uneven terrain.

Global optimization is effective when planning paths for a small number of objects. As the number of objects, physical relationships between objects, and animation time increases, the optimization search space becomes prohibitively large. The size of the

search space can be reduced by using IK to remove optimization DOFs, but the animated figure may take on a "marionette" look. The techniques presented in this section did not generate highly-realistic locomotion, but did provide valuable ideas and motivation for later locomotion systems.

Local Controllers

An alternative to global optimization involves the use of localized controllers to produce physically-accurate animation. These controllers drive joints towards goal configurations by applying forces and torques. The joint motion created by these controllers is not necessarily biomechanically accurate with respect to muscles, tendons, and ligaments, but it is physically accurate. The controllers require only a target configuration, so complex optimization schemes are not necessary to determine forces and torques. Generating locomotion using local controllers is then a problem of specifying target joint angles with respect to limb goals and synchronizing those limb goals.

The KLAW (Keyframe-less Animation of Walking) system (Bruderlin and Calvert, 1989) represents one of the first uses of local physical controllers for animation. To produce walking animations, local controllers were used to drive limb joints toward key configurations. Joint torques were approximated using a model for the difference between current and target angle and the time remaining to reach the target angle. This type of controller is known as a Proportional Derivative (PD) controller. The key configurations were specified by a finite state machine that coordinated higher-level limb goals. Figure 5

illustrates the relationship between limb goals and the gait timeline. At the top level, a set of walking parameters were used to control the finite state machine.

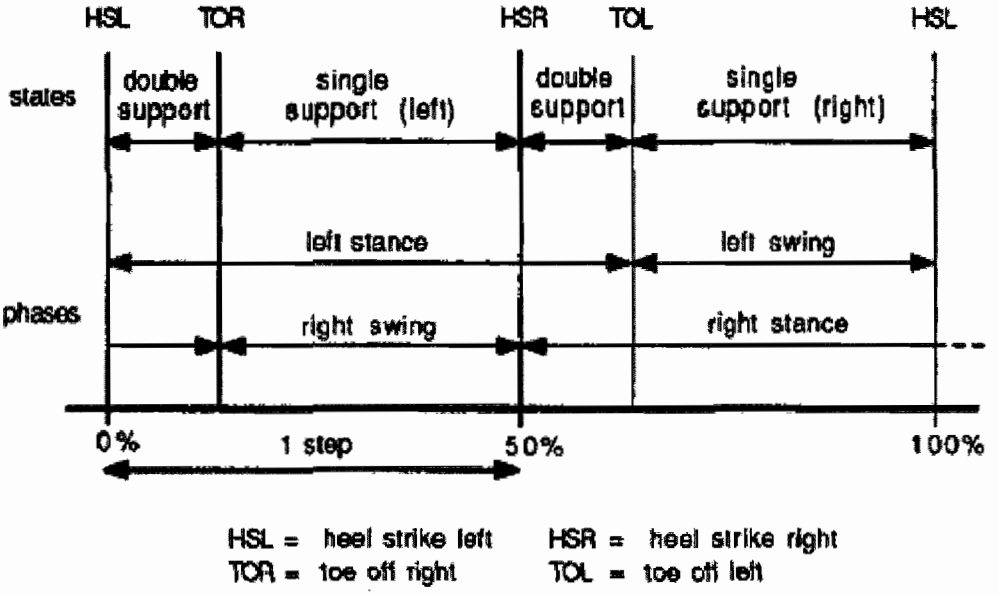


Figure 5. Gait timeline.

Note. From "Goal-directed, dynamic animation of human walking" by A. Bruderlin and T. W. Calvert, 1989, *Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, p. 235.

Three locomotion parameters were used to describe basic walking animations: forward velocity, step length, and step frequency. Forward velocity equals the product of step length and step frequency, so only two of the three parameters needed to be specified. In addition, up to 28 locomotion attributes could be varied to customize the walk. These parameters included: lateral distance between the feet, toe clearance during swing phase,

and the maximum rotation and list of the pelvis. These parameters were used to determine key joint configurations during limb stance and swing phases and the timing for transitions between states.

KLAW successfully generated lower-body walking animations using a simple (10 DOF) human model. The model could only walk forward, varying speed but not direction. Control of this simple walking scheme was intuitive. The animator could easily adjust walking speed and alter gait appearance by modifying the three primary locomotion parameters. For example, short stride length and high stride frequency resulted in short, quick steps. Long stride length and low stride frequency resulted in long, deliberate (but not leaping) strides.

Raibert and Hodgins (1991) presented a more general framework for animating locomotion. Here again, local controllers were used to drive joints and finite state machines were used to coordinate the controllers. Instead of driving joints toward key configurations, controllers acted as actuators to simulate the effect of muscles, ligaments, and tendons on joints. The finite state machine realized high-level limb goals by providing actuator subgoals to each joint controller.

Limb activity was divided into five states: The thrust and unloading states handled the application of forward and upward forces by the limb, followed by relaxation of the limb after lift off. The flight state was responsible for the mid-air behavior of the limb, preparing it for landing. The loading and compression states dealt with manus or pes contact with the ground and subsequent compression of the limb. Figure 6 illustrates the gait state machine. Limb activities were coordinated to produce specific gaits. For

example, quadruped trotting was generated by synchronizing diagonal limbs; quadruped bounding was generated by synchronizing fore and hind limbs.

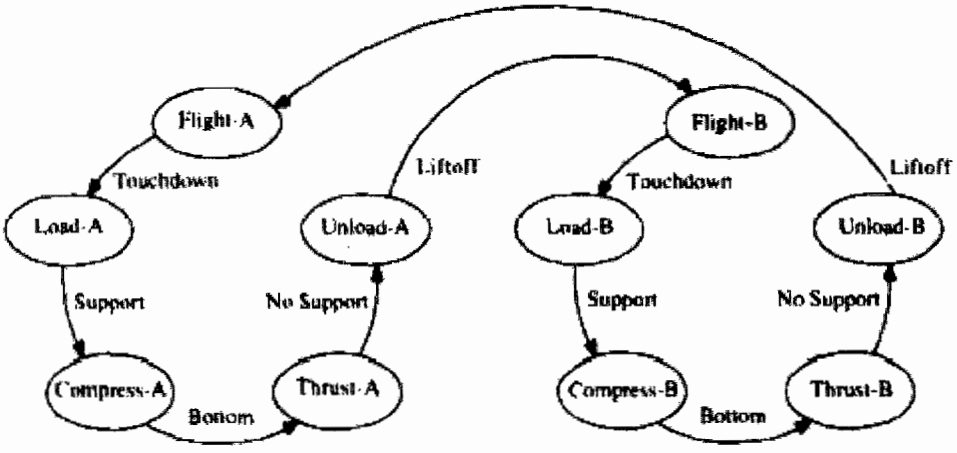


Figure 6. Gait state machine.

Note. From "Animation of dynamic legged locomotion" by M. H. Raibert and J. K. Hodgins, 1991, *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, p. 353.

Additional control methods were used to maintain character balance during locomotion. In general, a character is considered to be balanced if the character's COM is above the manus or pes of a supporting limb. If multiple limbs are supporting the character, the character is considered balanced if the COM is over the convex region defined by the manus/pes of all supporting limbs. This region is known as the support region or support polygon. Balance was maintained by applying forces and torques to the

character's COM to ensure the character remained upright and the COM stayed within the support region.

This framework was used to successfully generate several forward gaits including: biped running and galloping, quadruped trotting, bounding, and galloping, and kangaroo hopping. The framework extended previous work by actively maintaining character balance and simulating biomechanically-accurate motions. Adding gaits to the framework was particularly difficult because new actuator goals had to be specified along with limb-synchrony relationships.

Hodgins, Wooten, Brogan, and O'Brien (1995) extended the work presented by Raibert and Hodgins (1991) to generate realistic animations of human running, cycling, and vaulting. Like previous work, actuators were used for low-level joint control and finite state machines were used to coordinate actuators. Localized actuator goals were hand tuned, so defining new behaviors was a difficult task. After tuning, behaviors could be easily controlled by modifying high-level parameters.

In addition to forward running, the state-specific control functions also handled arbitrary turning. Previous methods used target velocity and step length parameters to predict the forward position of the next pes print. Turning was achieved by incorporating a facing direction to predict the lateral displacement of the next pes print location (shortening the forward displacement). The lateral displacement of the next pes print location was a function of the facing direction, so this technique could be used to facilitate an arbitrary amount of lateral reaching during turning. Arm animations were coordinated with the leg

animations to produce full body animations. Balance was preserved during turning, allowing realistic-looking running animations that followed user-defined paths.

Laszlo, van de Panne, and Fiume (1997) proposed a method for controlling human and robot walking by adding closed-loop control to periodic notions. Using a technique called Limit Cycle Control, periodic walking motions were automatically perturbed to maintain balance. Open-loop walking animation was generated using finite state machines and PD controllers. The open-loop walking of an upright character was considered the limit cycle. As a character began to lose balance, control methods would force the character back into the limit cycle. Figure 7 illustrates the Limit Cycle Control technique.

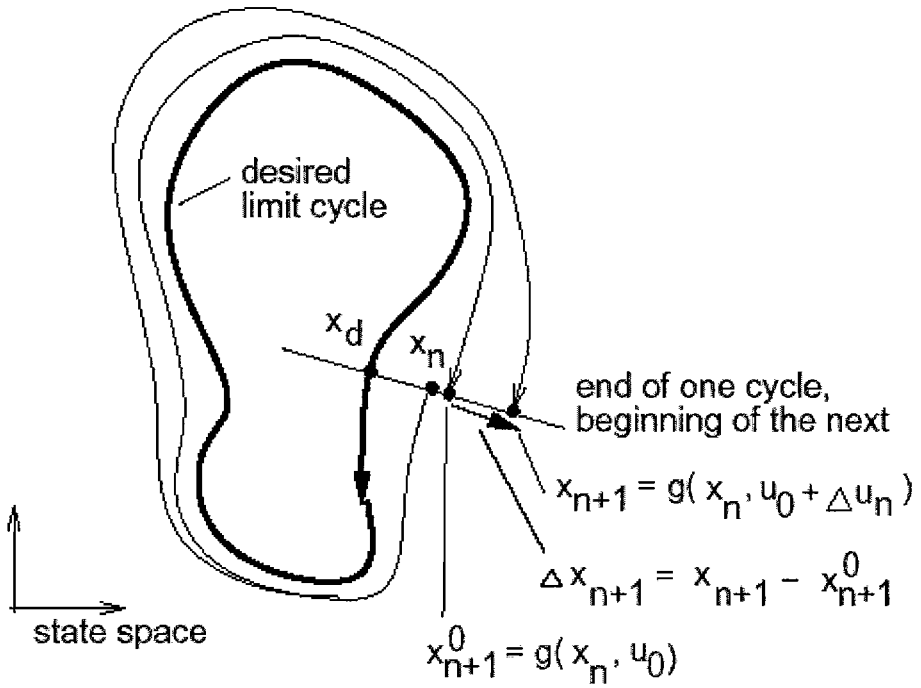


Figure 7. Limit Cycle Control.

Note. From "Control of Physically-based Simulated Walking" by J. F. Laszlo, M. van de Panne, and E. Fiume, 1997, *Proceedings of IMAGINA*, p. 234.

The Limit Cycle Control method allowed an interesting and intuitive control method for simulated humans and robots. The forward speed of the character could be controlled by leaning the character forward or backward. As the character leaned forward, walking speed would increase to keep the character from falling over forward. Likewise, turning could be controlled by leaning the character to the side or turning the hips. Unfortunately, only one gait was supported. Ideally, a similar technique could be used to allow backward steps, sideways steps, and gait changes.

To increase behavior repertoire of animated characters, Badler et al. (1995) used Sense Control Action (SCA) loops to control virtual character animation. During the sense phase of the loop, information was collected from the environment using sensors. The sensors provided information such as the distance and orientation of important objects. The control phase used input from the sensors to determine whether the character should be attracted or repelled from the objects. Finally, the action phase used control methods to accomplish the goals set forth by the control phase.

Locomotion was generated using a SCA loop and a Parallel Transition Network (PaT-Net). The SCA loop defined the "body" of the character: sensing possible locations for the next pes print, determining if those locations were safe, and then acting to take the step. The PaT-Net represented the "mind" of the character, determining high-level path planning goals. PaT-Nets consisted of hierarchical state machines that determined the behavior of the SCA loop. The PaT-Net architecture was designed to facilitate arbitrarily-complex behaviors. To this end, PaT-Net nodes could spawn new networks, allowing parallel execution and communication between networks. In theory, a complex PaT-Net could produce and transition between many different gaits.

The PaT-Net/SCA architecture was used to simulate a game of hide and seek between virtual characters. The PaT-Nets were used to determine high-level goals such as where to hide and where to look. The SCA loop was used to generate low-level locomotion. In related work, characters could follow a user-defined path. Addition of the PaT-Net allowed automatic path planning at animation time. This decomposition into

character minds and bodies is a powerful paradigm, allowing arbitrarily-complex behaviors to be built on existing mind and body functionality.

Faloutsos, van de Panne, and Terzopoulos (2001) proposed a standardized framework for physical controllers to drive character animation. By creating a controller module that met the specifications of a standard interface, animators could add their control modules to the behavior repertoire of an animated character. An initial network provided controllers that allowed a character to regain balance, take protective steps, and stand up in multiple ways after falling down.

Each controller contained a set of preconditions, a set of postconditions, and an expected performance. The preconditions were ranges of values for each joint DOF. A controller could become active only if all character DOFs were within the ranges specified by the preconditions. The postconditions specified the target values of each joint DOF. Due to ground contact and obstacles, a controller may have been unable to reach a postcondition state. The expected performance defined the error tolerance for the controller. The controller would report a failure and become inactive if any of the character DOFs exceeded the ranges specified by the expected performance. Only one controller was active at a time. Active controllers were picked based on which controller's preconditions best matched the current state of the character. Figure 8 shows a sample set of controllers and their relationships.

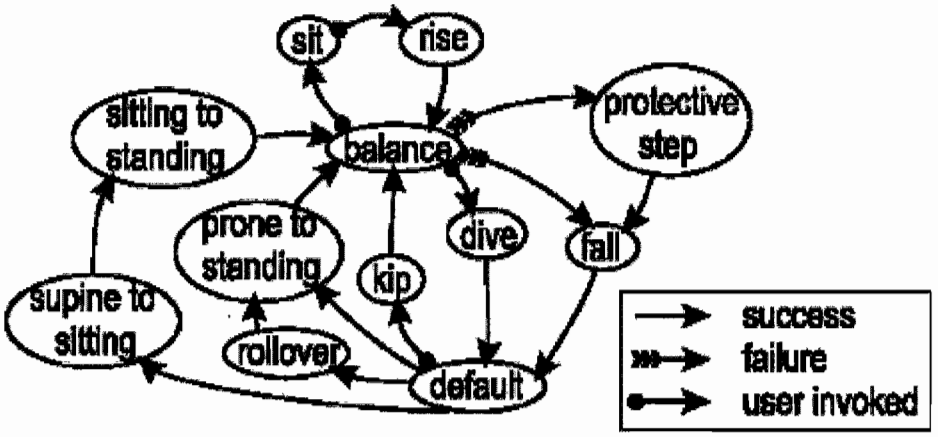


Figure 8. Controller graph.

Note. From "*Composable controllers for physics-based character animation*" by P. Faloutsos, M. van de Panne, and D. Terzopoulos, 2001, New York: ACM Press, p. 27.

The standardized framework allowed behaviors to be specified without the necessary creation of complex state machines and/or control functions. However, determining appropriate preconditions for each behavior would be difficult, especially as the behaviors count grew. A character may begin to fall backward, only to match the preconditions of an undesired behavior. Perhaps a distinction between standard and recovery behaviors would alleviate this problem. A recovery behavior could be activated upon failure of a standard or recovery behavior.

Using local controllers, the animation system can specify the target configuration for a joint and rely on a controller to produce physically-accurate motion. Controllers can be tailored for a specific joint to produce biomechanically-accurate motion. To produce locomotion, proper joint configurations must be determined based on limb goals and limb

goals must be coordinated. These goals and relationships are typically specified using a large number of parameters. Later, work will be presented that utilized EAs to evolve such parameter sets. For more information on physical controllers, see van de Panne's (2000) review of control methods for animating articulated characters.

Fragment Composition

Global optimization and Local Controller techniques can produce physically-accurate motion sequences but require a great amount of effort to specify complex character behavior. An alternative is to create complex animations by compositing existing simple motion sequences. For the purpose of generating locomotion, interactive acceleration and deceleration could be achieved by blending between a fast walking animation and a slow walking animation. The downside to this approach lies in the difficulty of creating motion transitions that are both physically and biomechanically accurate.

Rose, Guenter, Bodenheimer, and Cohen (1996) used Spacetime Constraints and IK to generate smooth motion transitions. Spacetime Constraints techniques were used to minimize the metabolic energy used by the character during the transition. In general, natural movements conserve energy as much as possible, so animations that conserve energy tend to look natural. Minimal metabolic expenditure was achieved by minimizing the joint torques necessary to transition between motion sequences. Figure 9 shows an example of simple linear interpolation versus animation using minimal metabolic expenditure.

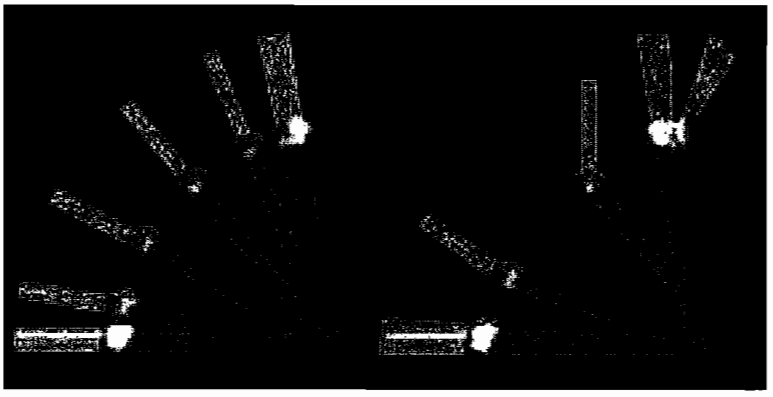


Figure 9. Minimum energy expenditure animation.

Note. From "Efficient generation of motion transitions using spacetime constraints" by C. Rose, B. Guenter, B. Bodenheime, and M. F. Cohen, 1996, *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, p. 152.

IK was used to keep supporting manus/pes stationary on the ground during transition. IK was propagated upward from the end effecters as necessary while forces and torques transformed joint configurations between motion sequences. Enforcing stationary end effecter constraints could have led to loss of character balance, which was not explicitly preserved. Natural-looking transition required the appropriate selection of motion sequences so that the transitions were short and did not require high-magnitude changes to joint configurations.

The Spacetime Constraints and IK techniques were used to generate motion transitions for a 44 DOF human model. Transitions were generated using a basis library of soccer motions. Physical and biomechanical accuracy of the transition motions was not explicitly enforced, yet the motions appeared plausible due to relatively-short transition

times and the minimization of metabolic energy during transition. The visual plausibility of transitions would likely falter in an interactive environment that required many unpredictable transitions.

Improv (Perlin and Goldberg, 1996) was designed to be a flexible animation system capable of automatic transitions. The animation system consisted of two subsystems: an animation engine and a behavioral engine. The animation engine allowed basic motions to be stored as actions. Actions could be layered and transitioned between. Noise functions were applied to actions to ensure that actions looked slightly different each time they were run. The noise functions provided visual nondeterminism in the produced animations. The behavioral engine allowed linking of actions to greetings, responses, and gestures.

Actions defined their constituent joint DOF values at the beginning and end of the action. Mutually-exclusive actions could be clustered into groups. If several such groups were defined, an action from each group could be performed simultaneously without oscillating a joint DOF. Two or more actions from a group could be animated simultaneously by applying a weighted average of the actions to the joint DOFs. Transitioning between mutually-exclusive actions was performed by varying the weighting values used in calculating the weighted sum of actions.

Improv has been used to create virtual actors that respond to spoken commands by performing actions. The use of a noise function during interpolation of DOF allowed natural-looking motions similar those created by physical controllers. The interpolated motions were not physically correct, so care was needed to ensure that the motions looked physically correct. Intermediate joint configurations would likely be necessary to specify

actions that look physically correct. Locomotion would be difficult to generate with such a system because gaits would have to be manually specified.

Badler et al. (1998) presented a Parameterized Action Representation (PAR) designed to link natural language with high-level animation goals. All actions available to a character were represented as PARs and stored in an *Actionary*. Agents were given natural language instructions, which were executed by initializing the appropriate PAR with context-specific information. For example, the command "Walk to your bicycle" might have initialized a walking PAR with information about the agent, the agent's current location, and the location of the bicycle. Low-level animation was driven by Motion Capture data or by SCA loops and PaT-Nets (Badler et al., 1995). Figure 10 shows the PAR template.

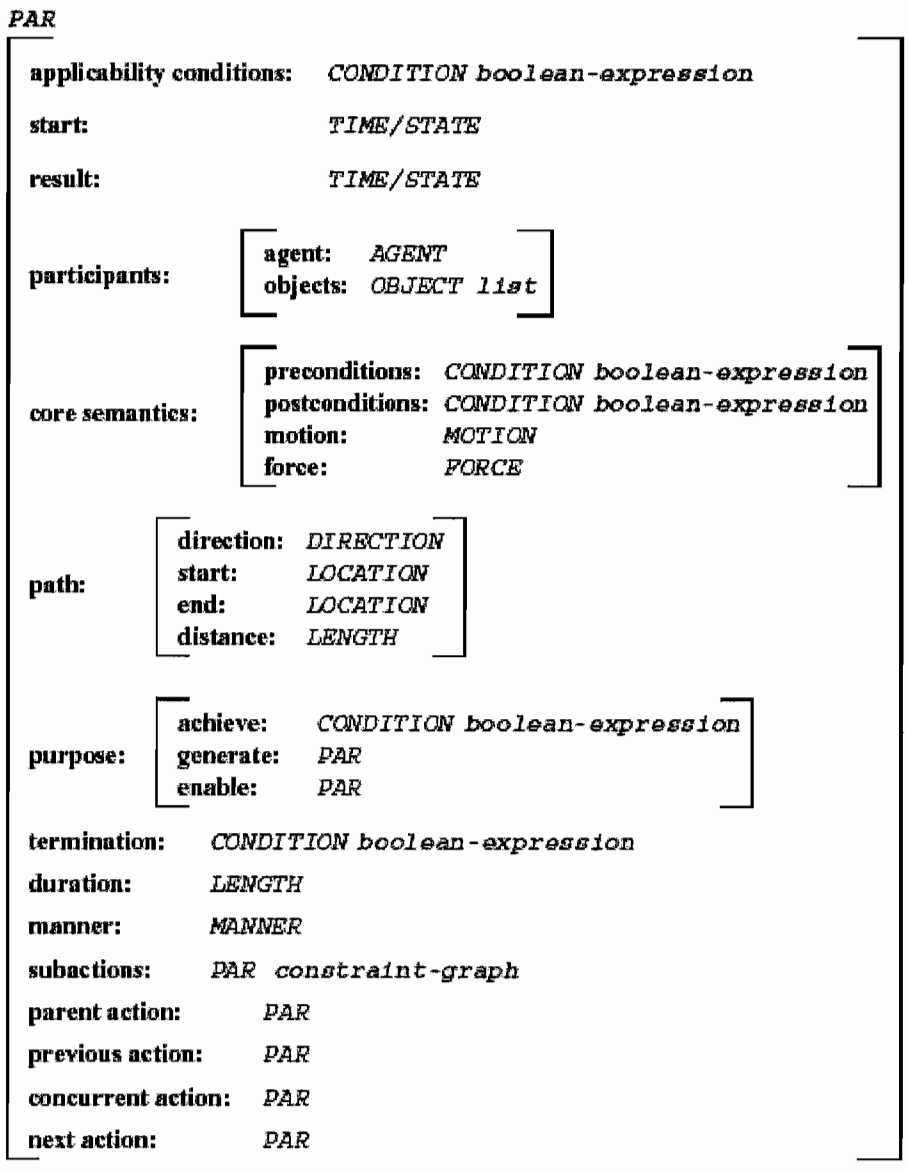


Figure 10. PAR template.

Note. From "A Parameterized Action Representation for Virtual Human Agents" by N. Badler, R. Bindiganavale, J. Bourne, M. Palmer, J. Shi, and W. Schuler, 1998, *Workshop on Embodied Conversational Characters*, p. 3.

Each active agent maintained a queue of initialized PARs. When no PAR was currently active, the applicability conditions of the PAR at the front of the queue were checked. The first applicable PAR then ran its preparatory actions. The preparatory actions transitioned the agent into a starting configuration for the execution steps. The execution steps were then run. Upon termination, the PAR ran its post actions, which depended on the success or failure of the execution steps. PARs could be nested in the preparatory, execution, and post action phases to specify complex actions.

The PAR representation was used to control five agents in a simulated lodge. Four of the agents could be controlled from different geographical locations using natural language commands. Possible actions included: walking, sitting in a chair or on a bed, talking to others, climbing a ladder, opening doors, shaking hands, bowing, and drinking. The fifth agent was an autonomous waiter that would fill empty glasses with water from a pitcher. When the pitcher became empty, the waiter would return to the kitchen to fill the pitcher.

The PAR representation was similar to the standardized controller framework presented by Faloutsos et al. (2001). Both methodologies utilized pre and post conditions and failure handling. The PAR framework was less reactive and more goal oriented, using existing motion sequences to accomplish high-level goals. Motion transitions were specified by the PARs, so physical accuracy was not guaranteed. PARs are well suited for controlling interactive characters in an environment where interaction has greater importance than the physical accuracy of motion.

Allbeck and Badler (2002) extended the PAR representation to incorporate personality and emotion into motions. Effort and shape parameters were used to automatically adjust motion captured or procedurally generated animation sequences. Effort parameters adjusted agent velocity, acceleration, and inertial parameters. Shape parameters changed the physical form of the agent's body as it moved through space. Using these methods, animation sequences were successfully modified to convey character personality and emotional traits.

Sun and Metaxas (2001) introduced a method for blending motion-captured locomotion sequences. Motion sequences were generated and stored using sagittal elevation angles. The sagittal plane is defined as the plane that bisects the left and right side of a character's body. To determine a sagittal elevation angle, a body segment was first projected onto the sagittal plane. The elevation angle for a segment was defined as the angle between the projected segment and the gravity vector. Figure 11 demonstrates the acquisition of sagittal elevation angles. A limb was represented using the elevation angles of four limb segments: the pes, the lower limb, the upper limb, and the pelvis. Sets of limb elevation angles were stored as 4-Dimensional points.

Figure 11. Sagittal elevation angles.

Note. From "Automating gait generation" by H. C. Sun and D. N. Metaxas, 2001, *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, p. 263.

Locomotion datasets were lists of 4-Dimensional limb configurations. For each dataset, a step length and height was computed and stored. While walking on even or uneven terrain, a pes print planner determined the reasonable pes print locations. For each step, the location of the next pes print determined an optimal step length and height for the current step. A new motion sequence was then selected such that the desired step length and height could be achieved by blending the current and next motion sequences. A user could also manually adjust the optimal step length and height. Additional parameters such as stance width and toe out angle were used to fine-tune gait appearance.

Using these methods, characters successfully walked up and down hills and followed curved paths. Curved paths were followed by simply changing the orientation of the character's sagittal plane, so balance was not preserved during turning. Nor was balance preserved during acceleration and deceleration. The selection and interpolation

algorithms were computationally efficient enough to be run at animation time. Similar techniques are common in more recent video games, in which characters can turn, walk on uneven terrain, and change gaits using only a few motion sequences.

Choi, Lee, and Shin (2003) introduced a method for automatically navigating a character through a virtual environment by blending motion captured sequences. Given a starting and ending location, a virtual trackway was generated and used to guide the character through the environment. Motion sequences were blended and concatenated to create an animation sequence that followed the trackway.

A set of possible pes prints was created by randomly sampling pes location and orientations in the environment. Each possible pes print was tested to ensure the location was safe (i.e., not water, fire), and that the orientation was reasonable considering to the local terrain. Safe pes prints were added to a directed graph, connected by edges that represented possible motion transitions. Edge weights were proportional to the step length between pes prints with a penalty based on the amount of motion clip degradation necessary to achieve the step. Path planning then consisted of finding the lowest-cost path from starting node to ending node. The resulting sequence of motion clips was concatenated and smoothed to produce the final animation. Figure 12 shows a character navigating a generated trackway.

Figure 12. Navigating a generated trackway.

Note. From "Planning biped locomotion using motion capture data and probabilistic roadmaps" by M. G. Choi, J. Lee, and S. Y. Shin, 2003, *ACM Transactions on Graphics (TOG), 22*(2), p. 196.

Trackways were successfully created to guide characters through an uneven environment with obstacles. Input motion sequences included: starting walk, stopping walk, walking straight, turning left, turning right, running, and broad jumping. Even in a small environment, several thousand pes print nodes and tens of thousands of transition edges were required to generate natural-looking animation. Not surprisingly, construction of the trackway was computationally intensive and had to be performed offline. Breaking the planning task into smaller subgoals might allow this technique to be used interactively.

Physically-accurate motion sequences are difficult to specify and generate. The flexibility and utility of these sequences can be increased by combining them with other physically-accurate sequences. It is difficult, however, to determine the appropriate

sequences to blend and the order in which to blend them. Fragment composition techniques are appropriate for applications that require blending between only a few motion sequences at the cost of physical and biomechanical accuracy, such as video games. Some of the ideas presented in the section, however, may be useful for generating locomotion, specifically in transitioning between gaits.

Automatic character animation techniques have been presented that successfully produced character animation with varying degrees of physical and biological realism. Many of these techniques were used directly to generate locomotion. Of the presented techniques, the methods involving local controllers produced the best locomotion results. The architectures based on local controllers were highly extensible, but required the manual tuning of large parameter sets. In the next section, EAs will be presented that can be used to automatically tune large parameter sets. For a general review of high-level animation methods, see the review presented by Giang, Mooney, Peters, and O'Sullivan (2000).

Evolutionary Algorithms

EAs encompass a class of search algorithms. These algorithms are typically used to find good solutions to problems that have large solution spaces (called fitness landscapes). An initial population of candidate solutions is created randomly or seeded using some heuristic. A fitness function is used to score the candidates based on how well they satisfy the optimization criteria. Each algorithm iteration involves mating and mutating the population, scoring the population using the fitness function, and selecting fit

candidates for the next generation. The highest-fitness candidate is returned after a set number of generations.

GAs were introduced by Holland (1992). GAs provide a simple mechanism for searching large fitness landscapes. Candidates, called genotypes, are usually stored as fixed-length binary strings. Candidates typically specify boolean or integer values. Wright (1991) discussed the encoding of real value parameters as binary strings. Mating can be achieved by swapping data between two candidate strings. One or more crossover points determine where data swapping starts or stops. Mutation can be performed by flipping random bits. The evaluated genotypes represent possible problem solution and are called phenotypes.

GP (Koza, 1990) allows the evolution of instruction sets to satisfy criteria. Programs can be evolved to efficiently solve problems, or to accomplish a task using sense/response pairs as demonstrated by Ngo and Marks (1993). Programs can be represented as fixed-length strings of operations or as variable-length parse trees. The mating of trees can be accomplished by swapping subtrees between two candidates. Mutation within a tree can be performed by randomly changing a node type or value to another valid type or value.

Sims (1991) described how GP can be used to evolve images and textures. Images were evolved from scratch to meet aesthetic criteria or modified to add a stochastic element to textures. Both fixed-length strings and variable-length trees were used to specify image operators. Fixed-length strings were used to represent pairs consisting of an image parameter and a modification to that parameter. Variable-length trees were used to specify

image operations and parameters. Tree nodes consisted of functions such as vector transformations, procedural noise generators, and image processing functions. Tree leaves were used to specify parameters for the operators. Similar techniques can be used to evolve 3D character morphologies.

Artificial Neural Networks (ANNs) were introduced by Rosenblatt (1958). ANNs provide a mechanism for simulating a learning mind. Neural networks are typically directed graphs consisting of activation nodes and weighted links. Node activation is based on an activation function which uses the weights and activation status of incoming links as input. A simple activation function uses the sum of the activated link weights to determine activation; the node is activated if the sum exceeds a threshold. Animated characters can utilize ANNs as virtual brains to establish links between stimuli to responses. Figure 13 illustrates a typical neural network.

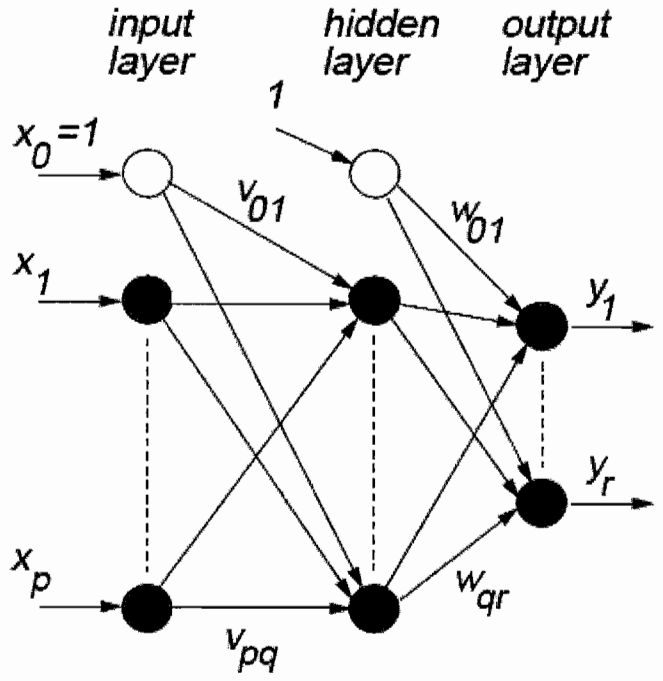


Figure 13. A typical neural network.

Note. From "Neuroanimator: Fast neural network emulation and control of physics-based models" by R. Grzeszczuk, D. Terzopoulos, and G. Hinton, 1998, SIGGRAPH 98 Conference Proceedings. *Annual Conference Series*, p. 10.

Neural networks are usually organized into three layers: an input layer, a hidden layer, and an output later. The input layer consists of nodes that are activated based on some stimulus. Upon activation of a node, all links outgoing from the node are activated. The hidden layer consists of a number of interconnected nodes. Inputs to hidden layer nodes are links from the input layer or other hidden nodes. Output links from hidden layer nodes lead to output nodes or other hidden layer nodes. The output layer contains nodes

that contribute to a decision made by the network. The number of input and output nodes depends on the desired network stimuli and responses. The hidden node count is often increased until satisfactory results are reached.

ANNs are trained by adjusting link weights. Networks are typically trained by propagating changes throughout the system. For example, if the network responds correctly, activated link weights are increased. Conversely, activated links weights are reduced when the network responds incorrectly. Alternatively, EAs can be used to train networks by evolving a set of link weights. The fitness of a set of links weights is determined by network performance when using those weights.

EAs provide an effective way to search large problem spaces. The primary difficulties in using these algorithms lie in determining the candidate representation and the definition of an appropriate fitness function. In the next section, techniques will be presented that evolve the minds and/or bodies of animated characters for locomotion.

Evolutionary Locomotion

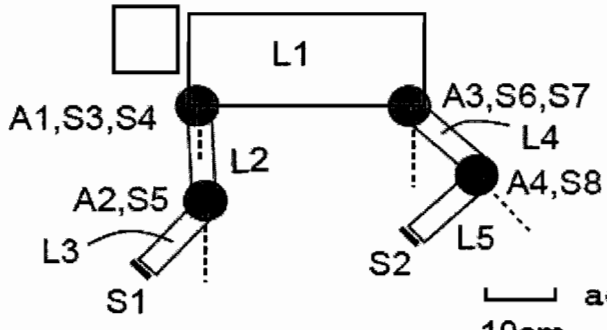
Automatically generating locomotory ability is difficult due to the size and interconnectedness of gait parameter sets. EAs have been employed to evolve the parameter sets that govern Local Controller architectures. The parameter sets were typically evaluated based on their ability to generate forward locomotion. ALife methods have been used to create interesting new creatures capable of locomotion. ALife methods typically use EAs to evolve both creature morphologies and control architectures. Finally, EA and robotics techniques have been used to train and improve robot locomotion. The

robots typically used simple yet effective evolution methods that could be useful for simplifying animation architectures.

Controller Evolution

Physical controllers can be optimized for locomotion by evolving gait parameters sets or by evolving ANN weights. Controllers were evolved for creatures with fixed biological structures. Modifying the morphology of a creature would necessitate re-evolution of the gait parameters or network weights. Controller fitness was typically determined by the creature's ability to achieve the desired gait. Such evolution schemes promoted the emergence of a single gait.

van de Panne (1993) introduced Sensor Actuator Networks (SANs). SANs provide a framework for correlating sensors with character behaviors, similar to the SR pairs presented by Ngo and Marks (1993). Characters were defined by connecting rigid links and 1-DOF joints. Sensors and PD actuators were utilized to control joint motion. Sensor types included: angle sensors, touch sensors for determining ground contact, eye sensors for visual tracking, and length sensors to monitor the linear distance between points of interest. Angular actuators were used to drive joints and linear actuators were used to exert forces between body links. Figure 14 shows an example SAN specification.



sensor	type	link	min	max
S1	touch			
S2	touch			
S3	angle	2	−180	−10
S4	angle	2	25	180
S5	angle	3	−180	−80
S6	angle	4	−180	42
S7	angle	4	55	180
S8	angle	5	−180	−105

link	mass (kg)
L1	1.0
L2	0.15
L3	0.10
L4	0.15
L5	0.10

actuators:

act.	min	max	ks	kd
A1	−20	30	0.4	0.01
A2	−85	−55	0.4	0.01
A3	40	55	0.4	0.01
A4	−110	−105	0.4	0.01

Figure 14. Example SAN specification.

Note. From "Sensor-actuator networks" by M. van de Panne and E. Fiume, 1993, *Proceedings of SIGGRAPH, 93*, p. 336.

Relationships between sensors and actuators were defined using an ANN. The input layer of the network consisted of one node per sensor, the hidden layer contained one hidden node per sensor node, and the output layer contained one node per actuator. The input layer nodes were fully connected to the hidden and output layers. The hidden layer nodes were fully connected to the output layer. Incoming weight values were summed by the actuator nodes to determine both whether the actuator would activate/deactivate and the amount of force or torque to be applied by the actuator. A hysteresis function was used to avoid chattering effects that can be caused by rapidly switching an actuator on and off.

The neural networks weights were evolved to optimize the forward locomotion speed of animated characters. Optimization was performed without the use of EA. Sets of neural network weights were first randomly generated until a configuration was found with satisfactory locomotion results. Stochastic Gradient Ascent and Simulated Annealing (two extensions of the Hill Climbing search technique) were then used to fine tune the network weights. Both of these Hill Climbing techniques performed roughly equally well, one sometimes outperforming the other and vice versa. A single GA could have been used to accomplish the goals of both optimization phases.

SANs were used to generate locomotion in 10 distinct creatures, including the *Luxo, Jr.* lamp. Each creature used only a few 1-DOF joints. Creature simplicity allowed the optimization algorithms to quickly converge on efficient gaits. The ANN paradigm allowed deterministic relationships to be established between sensors and actuators. Each evolved network was only capable of one type of locomotion, so other techniques are needed to be incorporated to allow multiple gaits and gait transitions.

Grzeszczuk and Terzopoulos (1995) presented a two-pass algorithm for evolving locomotion. Low-level controllers were first evolved to maximize objective functions. Each low-level controller consisted of a set of time-dependent control functions, one per actuator. The control functions were represented in a discrete form using B-splines. Objective functions were based on locomotive goals, such as moving forward at a desired speed or turning towards a specific direction. Simulated Annealing was used to maximize the objective functions by varying spline parameters.

Controllers were optimized to perform basic low-level action such as walking forward, walking backward, and turning with different radii. The low-level controllers were then combined to create high-level controllers capable of performing more complex tasks. A greedy algorithm was shown to quickly select an appropriate sequence of low-level controllers to accomplish an animation goal. Low-level controllers were also sequenced using Simulated Annealing, which better achieved animation goals at the cost of animation-time performance. Figure 15 shows a shark model comprised of low-level controllers.

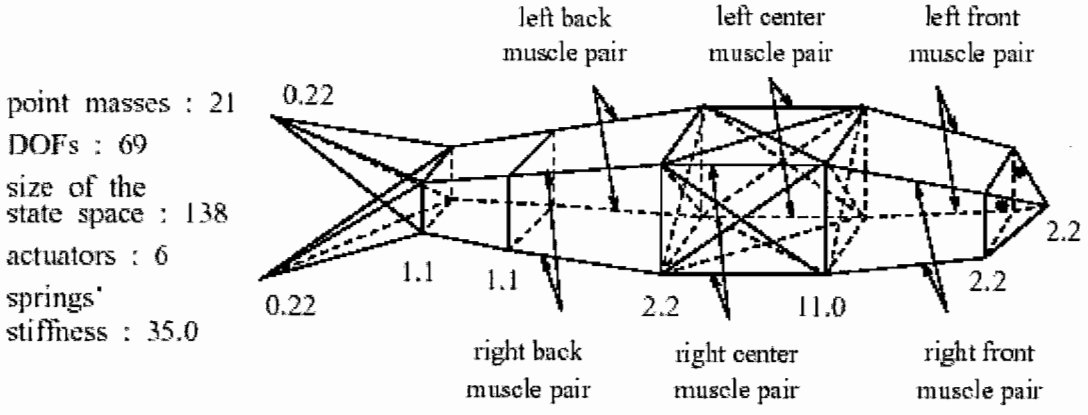


Figure 15. Shark model using low-level controllers.

Note. From "Automated learning of muscle-actuated locomotion through control abstraction" by R. Grzeszczuk and D. Terzopoulos, 1995, *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, p. 70.

The benefits and drawbacks of this method are similar to those related to the fragment composition techniques presented earlier. The sequencing of low-level controllers is particularly attractive because simple controllers can be easily evolved to meet animation criteria. Controller sequences must be generated at animation time for the creatures to behave in an interactive environment. Greedy algorithms present an interesting method to quickly sequence controllers. After sequence selection, motions must be carefully blended to prevent discontinuities.

An approach to evolving controllers using GP was presented by Gritz and Hahn (1997). As input, the user specified a detailed character model and a fitness metric. Character details included bone dimensions, masses, moments of inertia, and joint limits. The fitness metric incorporated animation goals for the character. Using the fitness metric, a controller program was evolved that specified target joint angles as a function of animation time. PD methods were used to drive joints toward target angles.

Controller programs were represented as LISP S-expressions. Candidate programs consisted of one expression per joint DOF. The expressions defined target joint angles as time-dependent functions. Programs were scored by the fitness metric based on how well a character completed the animation goals. Fitness metrics were usually in the form of a primary goal (e.g., "Move to location X.") and secondary goals (e.g., "Don't fall down."). Goal importance could be weighted to encourage the optimization of primary goals over others. Goals could also be phased in later during the evolution process to ensure that primary goals were satisfied first.

Evolved S-expression controllers were used to generate physically-accurate animations of the *Luxo Jr.* lamp. The lamp was "taught" to hop forward to a goal location and to limbo under a pole. To limbo, the lamp was given the primary goal of forward locomotion with a secondary goal of not hitting the pole. The use of multiple evolution objectives allows more complex animation goals. Complex goals would likely require more evolutionary iterations and/or a larger genetic population to satisfactorily evolve. Goals such as the limbo, however, would be difficult to achieve by blending and concatenating existing motion sequences. Figure 16 shows the *Luxo Jr.* lamp's limbo animation.

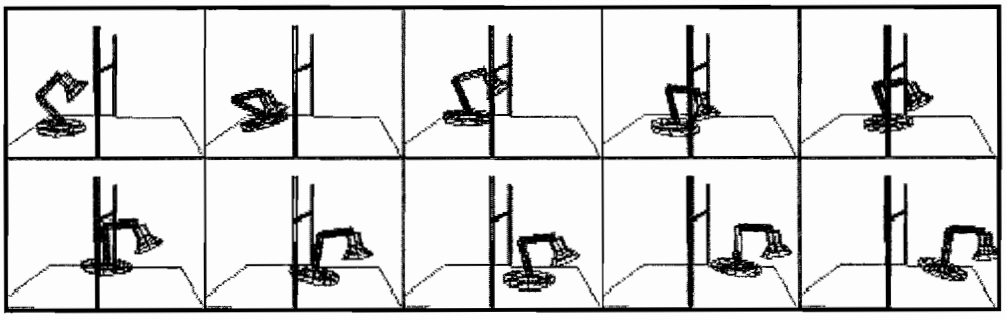


Figure 16. The *Luxo Jr.* lamp limboing.

Note. From "Genetic programming evolution of controllers for 3-D character animation" by L. Gritz and J. K. Hahn, 1997, *Genetic Programming, 97*, p. 143.

Grzeszczuk et al. (1998) introduced the NeuroAnimator framework. NeuroAnimators used ANNs to produce physically-accurate behavior instead of the standard numerical integration techniques. The neural networks were trained prior to animation using physical simulation time steps as inputs and outputs. The neural networks predicted the next state of a physical system given the current state and a time step, so real-time dynamics simulation could be performed with arbitrary time steps. Use of the neural networks greatly increased runtime efficiency because prediction was computationally inexpensive and there is no need for multiple integration steps between frame renderings.

NeuroAnimators could be used to train physical controllers for optimized behavior. Neural network weights were not evolved because modifying the weights could invalidate their physically-accurate predictions. Instead, a controller was evolved for each animated DOF. Controllers were represented by time-dependent functions that modified the neural network inputs. Partial derivatives of the NeuroAnimator output states could be computed with respect to control inputs, so a gradient-based optimization was used to efficiently evolve the controllers.

NeuroAnimators were used to evolve controllers for animation sequences such as: a pendulum trying to reach a goal state, a truck attempting to park at a specified location and orientation, a lunar lander attempting to land with a low descent velocity at a specified location and orientation, and a dolphin swimming with an optimized forward velocity. Using a neural network to replace numerical integration reduced the computational complexity of dynamic simulation but required network training. Current computer hardware and numerical integration techniques are fast enough that training drawbacks

probably outweigh performance gains. However, the NeuroAnimator approach does provides an efficient means for optimizing controllers.

Kang, Cho, and Lee (1999) extended the legged hopping model presented by Raibert and Hogins (1991) to evolve human running models. Two distinct hopping legs were effectively coordinated to produce running motions. A standing phase and a jumping phase were used to control the legs. During the standing phase, the leg first acted as a spring absorbing downward velocity. The leg then propelled the character forward and up. During the jumping phase, the limb was driven towards its landing configuration. The jumping phase was activated when the limb length exceeded a threshold. Upon activation of the jumping phase, a landing time was computed and used to activate the standing phase.

To maintain character balance, the COM was driven towards a point over the character's stance-leg pes, similar to the balancing technique used by Raibert and Hogins (1991) .When no feet were in contact with the ground, the COM was driven towards a point over the mean location of the feet. Unlike previous work, the upper body was actively controlled to change the character's COM. A combination of arm and lower back movements were used to displace the character's COM towards its target location.

The angular accelerations, maximum angles, spring constants, and nominal leg length were all editable parameters. GAs were used to evolve parameter sets. Running models were scored based on animation smoothness and energy consumption. Smoothness was evaluated by minimizing the difference in angular velocities between jumping and standing phases. This criterion promotes soft footfalls during running. Energy consumption was minimized by reducing the angular accelerations necessary to produce

the animation. By adjusting the target energy consumption, the run could be made to appear brisk or tired. In general, varying the amount of energy consumed by a character could result in animation that is not biologically accurate, especially if joint limits are not enforced.

These methods were used to successfully generate human running animations at various speeds. Figure 17 shows a running gait evolved using the GA. The evolved parameter sets generated only forward locomotion. GA techniques alone are likely insufficient to produce locomotion along an arbitrary path. To follow an arbitrary path, the character must be able to modify their gait for turning, which requires a mapping between path stimulus and gait response. A method for varying gait parameters based on environmental and user stimulus is necessary for interactive characters.

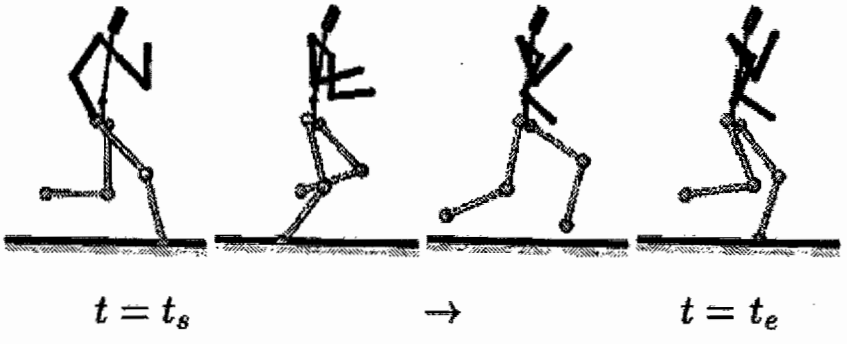


Figure 17. Running gait evolved using GAs.

Note. From "An efficient control over human running animation with extension of planar hopper model" by Y. M. Kang, H. G. Cho, and E. T. Lee, 1999, *Journal of Visualization and Computer Animation, 10*(4), p. 224.

Ijspeert and Arbib (2000) introduced a model to generate locomotion for simulated 3D salamanders. The model consisted of three control abstractions: a vision-based control circuit, an ANN, and a set of muscle actuators. The vision-based control circuit provided inputs to drive the speed and direction of the reptile. The vision circuit provided non-oscillating input to each side of the body, alternating and symmetrical for forward locomotion or asymmetrical for turning. The vision circuit provided input for the ANN which generated motoneural outputs to drive simulated muscles. The body trunk and limbs were actuated using spring-based muscles.

A leaky integrator neural network was used to simulate the salamander's central pattern generators. Leaky integrator neurons differed from traditional artificial neurons in that they used additional timing and bias parameters to determine activation. The timing and bias parameters helped reduced actuator oscillations by adding complexity to the activation model. A GA was used to train the neural network for optimal locomotive efficiency in three stages. First, the limb central pattern generator was trained to generate reaching and supporting movements which were shared by the limbs. The body central pattern generator was next trained to produce coupled contractions for body bending. Finally, the neural network was trained to coordinate limb and body movements.

These methods were used to generate both swimming and trotting gaits. Figure 18 shows the generated salamander trotting animation. The ANN allowed brainstem-like control over locomotion, taking non-oscillating input from only four input nodes. The virtual salamanders could turn by bending their bodies and using asymmetric step lengths. The three-step evolution process provided a mechanism for training subnetworks of the

neural network with different goals. Subdivision of the evolutionary process affords quicker and more reliable convergence by the GA.

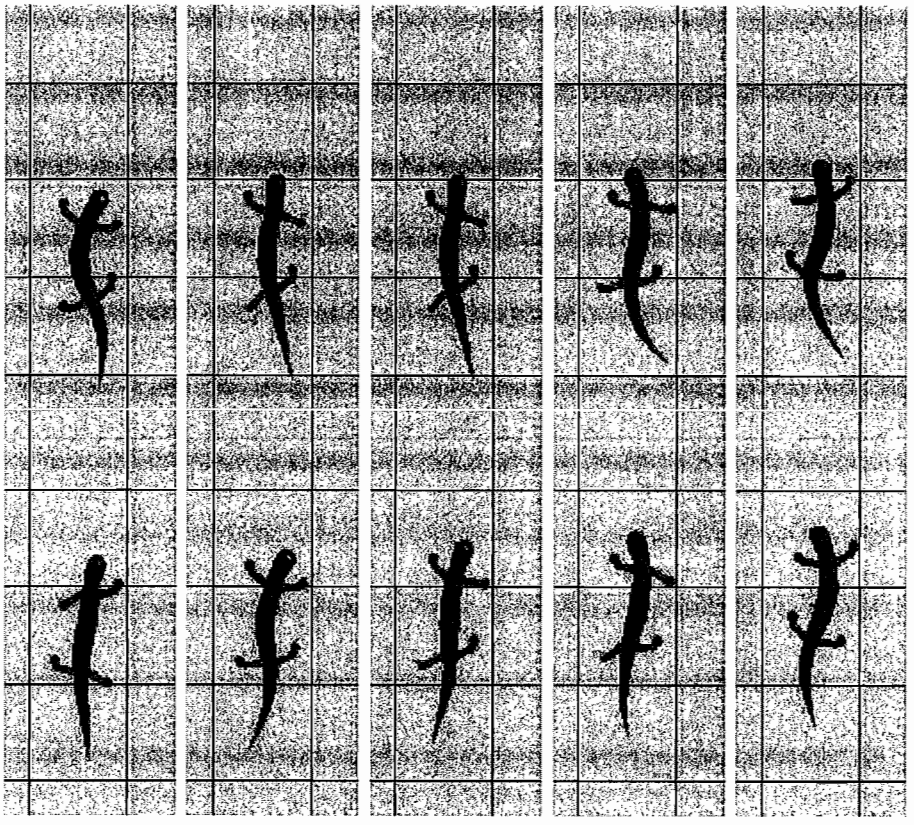


Figure 18. Salamander trotting animation.

Note. From "Visual tracking in simulated salamander locomotion" by A. J. Ijspeert, and M. Arbib, 2000, in J. A. Meyer and A. Berthoz (Eds.), Proceedings of SAB'00, From Animals to Animats 6, Paris, France, p. 92.

Sellers et al. (2004) used GAs to explore possible gait strategies for *Australopithecus afarensis*. A 2D model consisting of only the hind limbs and pelvic girdle was used to evaluate the metabolic costs incurred by different gaits. By varying the maximum knee extension angle, the simulated hominid could be made to adopt a human-like or chimpanzee-like walking gait. Considering the metabolic costs of these gait variations, inferences were made about the walking behavior of *Australopithecus afarensis*.

A finite state machine was used to produce locomotion. The state machine utilized three states, each representing a key pose for a leg. Each state was mirrored across the sagittal plane to animate the contralateral limb. Muscle activation levels were used to simulate a muscle pair for each joint by applying torques about the joint. Each state contained seven parameters: duration, right hip activation level, right knee activation level, right ankle activation level, left hip activation level, left knee activation level, and left ankle activation level. The set of 21 parameters was used as a linear genome for the GA. The GA optimized the parameters such that maximal forward distance was covered for a fixed metabolic cost.

Simulation results showed that bent-knee gaits consumed significantly more metabolic energy than a human-like gait. Figure 19 shows the effect of limiting knee extension on the cost of travel between the two models. A Basic Metabolic Rate (BMR) was used to estimate metabolic costs not related to locomotion. The metabolic costs could have been overestimated due to the lack of natural energy-saving mechanisms such as spring elements and complex joint morphologies.

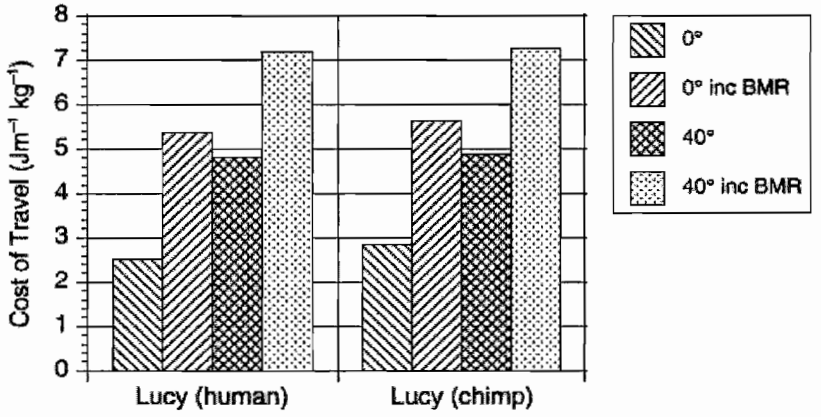


Figure 19. Effect of limiting knee extension on cost of travel.

Note. From "Evaluating alternative gait strategies using evolutionary robotics" by W. I. Sellers, L. A. Dennis, W.-J. Wang, and R. H. Crompton, 2004, *Journal of Anatomy, 204*(5), p. 349.

Controllers have been devised to produce efficient locomotion in humans, fish, snakes, salamanders, and extinct hominids. Locomotion was generated by evolving controller parameters sets or by evolving neural weights for ANNs. Parameter-based controllers are well suited for generating and studying a given type of gait, although parameter values could be varied to produce different gaits and gait transitions. Neural network controllers show promise in supporting gait variations by varying network inputs, but network training can be difficult and computationally expensive.

Artificial Life

ALife techniques have been used to simultaneously evolve the bodies and control architectures of virtual creatures. Unlike other work that has attempted to recreate realistic locomotion for known creatures, work based in ALife has attempted to create interesting new creatures and new forms of locomotion. ALife provides an application for locomotion synthesis, but also introduces fresh perspectives on the evolution of locomotion. Strange yet efficient gaits can be embraced when there are few preconceived notions about how a creature should move about.

Sims (1994a) introduced a method for simultaneously evolving the bodies and control architectures of virtual creatures. Creature morphology was determined by a directed graph in which nodes represented bones and links represented joints. Each node contained bone dimension data and its parent link specified a joint type. Joint types included: rigid, revolute, twist, universal, bend-twist, twist-bend, and spherical. Substructures could be linked recursively to reproduce limb-like structures. A recursive limit prevented infinite recursion.

Each joint owned an ANN which utilized sensors and effectors to determine the behavior of the joint. Sensors types included: joint angle, ground contact, and photosensors. Effectors operated as angular actuators to produce joint torques. Joint torque magnitude was based on neural input and a maximum strength value proportional to the cross-sectional areas of the adjacent bones. Neuron activation was based on evolvable expressions that derived output from input signals. Neural networks could be linked to adjacent joints (separated by one bone) to allow synchronization between joints.

GP was used to simultaneously evolve creature minds and bodies. An initial population of creatures was randomly generated. Each creature was then evaluated based on forward distance traveled in a set amount of time. Highly fit creatures were mated and mutated for the next generation. Mating and mutating modified both the morphologies of the creatures and their neural network structures and weights. Creatures were successfully evolved to walk, swim, jump, or follow a light source. For these experiments, energy consumption did not factor into creature fitness.

Interesting forms of locomotion emerged during the later stages of evolution. Swimming creatures used many synchronized paddles or winding snake-like motions to propel themselves. Swimming creatures that followed a light source developed fins for steering. Walking creatures used lizard-like walks, rocking motions, and inchworm-like pushing and pulling motions for forward locomotion. The jumping creatures all used similar strategies involving the compression and expansion of limb-like structures. Gait possibilities are interesting to explore, and perhaps give some insight into the origin of locomotion for certain animals. Figure 20 shows examples of creatures evolved for walking.

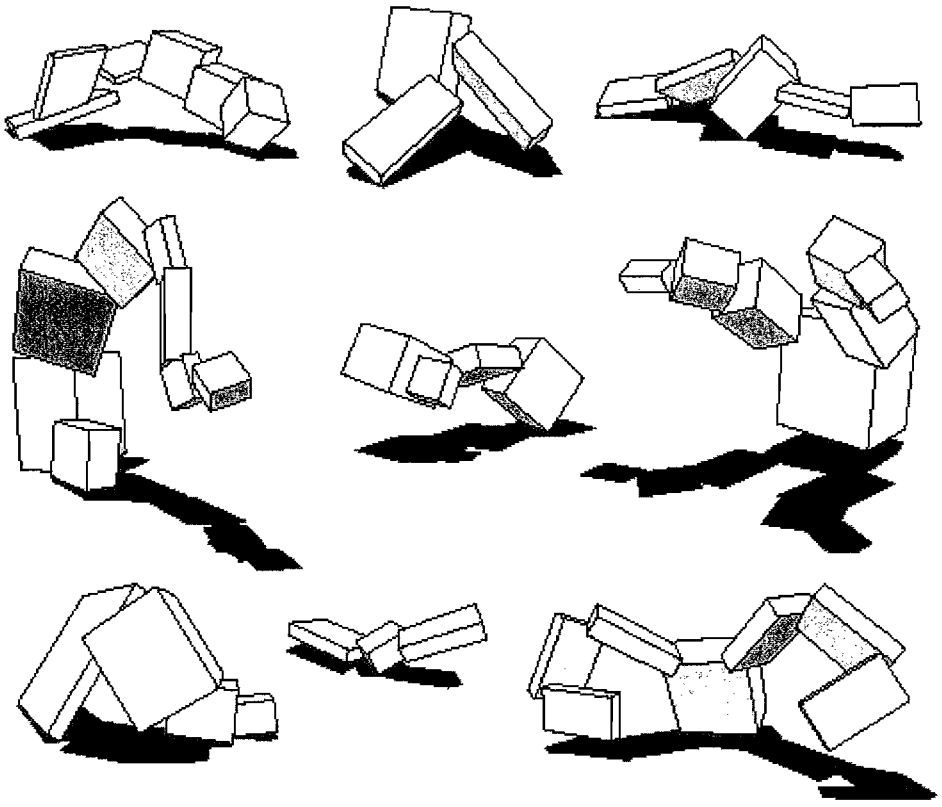


Figure 20. Creatures evolved for walking.

Note. From "Evolving virtual creatures" by K. Sims, 1994, *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, p. 21.

Sims (1994b) extended his earlier work by using competition to co-evolve populations of virtual creatures. Instead of determining the fitness of a creature by its ability to achieve forward locomotion, fitness was based on a creature's ability to outperform its competitors. The co-evolution techniques closely resembled natural selection, in which there is no specific optimization criterion. Using a survival of the fittest

mentality, competition losers had a chance of being eliminated from the population. Co-evolution produced competitive behaviors that would likely not have emerged using standard EA techniques.

A competition was devised to encourage protective behaviors for creatures. Competitions took place in a virtual arena containing two virtual creatures and a cube. Before competition, the cube was place in the center of the arena with the creatures on opposite side of it. The starting distance between a creature and the cube was proportional to the creature's height, discouraging the evolution of creatures that simply fall onto the cube. Competition ended after one creature had been in contact with the cube for a set amount of time, or a maximum amount of time has elapsed. After competition ended, the distance was calculated between each creature and the cube. Creature fitness was based on a ratio of these distances, so creatures were encouraged to both reach the cube and keep it away from their opponent. Figure 21 shows the virtual competition arena.

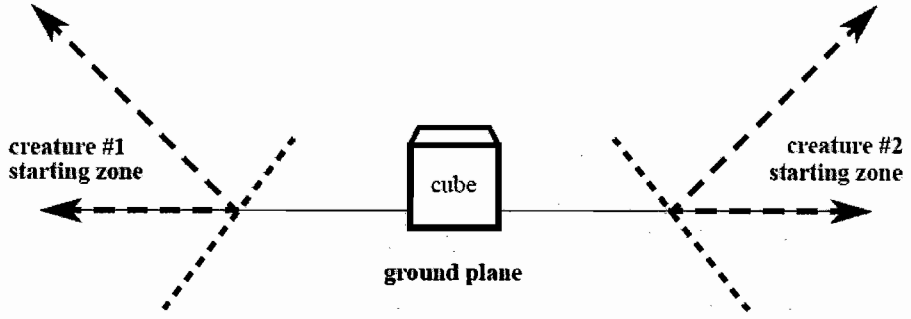


Figure 21. Virtual competition arena.

Note. From "Evolving 3D Morphology and Behavior by Competition" by K. Sims, 1994, *Artificial Life, 1*(4), p. 354.

Co-evolution produced some interesting strategies for winning the competition. Some creatures used arm-like structure to block access to the cube, some curled up around the cube, and some moved the cube away from the center of the arena. These behaviors would not have emerged through individual evolution. If the competition had been a race, then results would likely have been similar to those produced using individual evolution. Co-evolution methods are well suited for evolving creatures to outperform competitors, especially when offensive and defensive mechanisms are necessary components of a successful strategy.

Ray (2001) built upon Sims' framework by allowing the user to interact with the evolutionary process. An initial random population was first generated and displayed for the user. Still images and motion sequences could be viewed for each creature. Aesthetically pleasing creatures could then be selected for combination and/or further

evolution. Selected creatures could be evolved to perform behaviors such as: walking, jumping, swimming, and following. In this way, a user could evolve a virtual creature to his/her liking.

Directed graphs were used to represent virtual creature minds and bodies using an approach similar to that of Sims (1994). Evolvable color attributes were included with the morphology parameters for added customizability. New sensor types were added to the joint control networks including: position, velocity, angular velocity, force, and color perception. A new type of effecter modified color attributes. The additional parameters, sensors, and effectors supported a larger range of behaviors and appearances for the virtual creatures. Figure 22 shows an example evolved virtual pet.

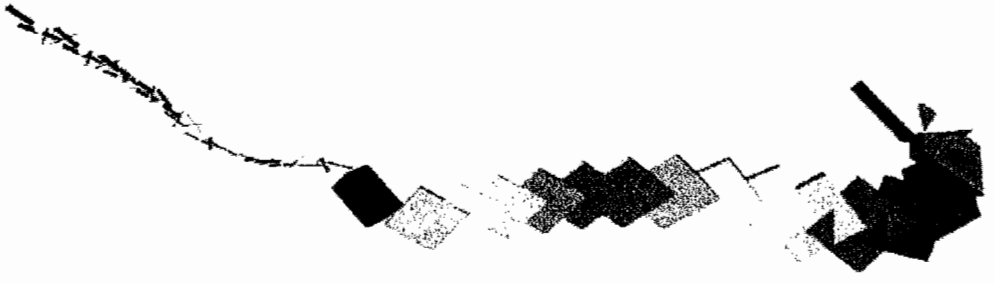


Figure 22. Example evolved virtual pet.

Note. From "Aesthetically Evolved Virtual Pets" by T. S. Ray, 2001, *Leonardo, 34*(4), p. 314.

This framework provided a means for users to interactively evolve virtual pets. A natural extension would be a system for nurturing the virtual pets. The implementation did not use collision avoidance to prevent interpenetration of body segments. As a corollary, multiple interpenetrating (and therefore redundant) body segments may have been considered in contact with the ground or water. The additional ground contact may have caused unpredictable and possibly unrealistic results during physical simulation of the creatures.

Hornby and Pollack (2001) used parametric Lindenmayer Systems (L-Systems) to generate creatures that exhibited morphological symmetries like those seen in natural creatures. L-systems consist of grammatical rewriting rules that are applied in parallel to an input string, resulting in an output string that is a variation of the input string. Lindenmayer (1968) first used such systems to model the development of cellular organisms. Later, Parametric L-system (Lindenmayer, 1974) extended "traditional" L-systems by using parameters and algebraic expressions when determining which production rule to use. The use of parametric L-systems allows the evolution of creatures with hierarchies of similarities, such as limbs and vertebral structures.

Creature behavior was controlled using an ANN. The networks were created by evaluating a string of commands outputted by the L-systems. The commands created, duplicated, and merged neurons and adjusted link weights. The neural network used a single input node, driven by either a sigmoid, linear, or oscillating function. Morphologies were constructed similarly by interpreting LOGO-style build commands. These commands allowed the creation and backtracking of rigid links and the creation of several joint types.

When a joint was created, an output neuron was also created in the neural network to drive the joint.

The L-systems responsible for generating creature morphologies and neural controllers were evolved using GP. The production rules of the L-systems were evolved to maximize the distance traveled by a creature during a set period of time. Creatures were penalized for dragging contact, which encouraged the creatures to take steps. Experimental data showed that the creatures evolved using L-systems exhibited more regularity and greater fitness than creatures evolved using earlier approaches. These results were likely due to the emergence of limb-like structures. Figure 23 shows two example L-system generated creatures.

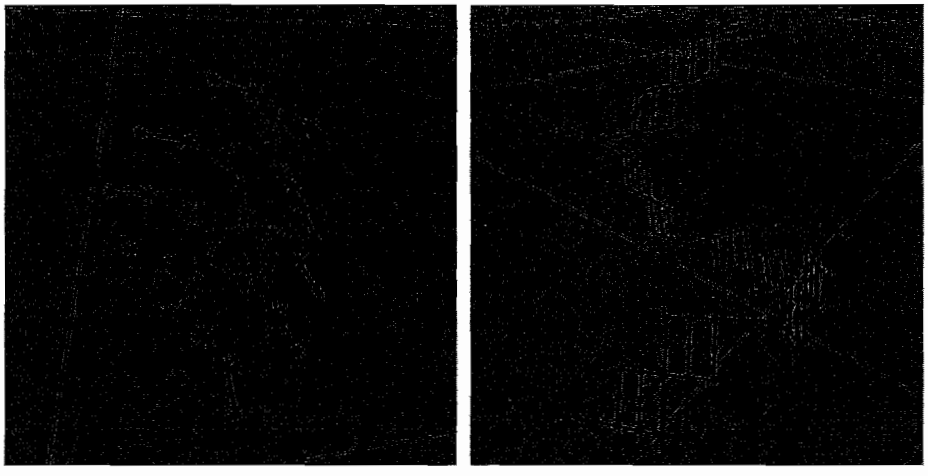


Figure 23. L-system generated creatures.

Note. From "Body-brain co-evolution using L-systems as a generative encoding" by G. S. Hornby and J. B. Pollack, 2001, *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, p. 874.

The ALife work presented thus far has explored the simultaneous evolution of character minds and bodies. Bongard and Pfeifer (2002) presented a method for comparing the locomotive efficiency of creatures with different bodies but similar minds. Ten different legged creatures were created, each with unique size, masses, and body plans. Each creature was equipped with eight actuated 1-DOF joints, four angle sensors at key joints, and four touch sensors at key body locations. The creatures were evaluated based on the forward distance they were able to travel during a fixed time interval. Figure 24 shows the ten creatures that were created and evaluated for locomotory efficiency.

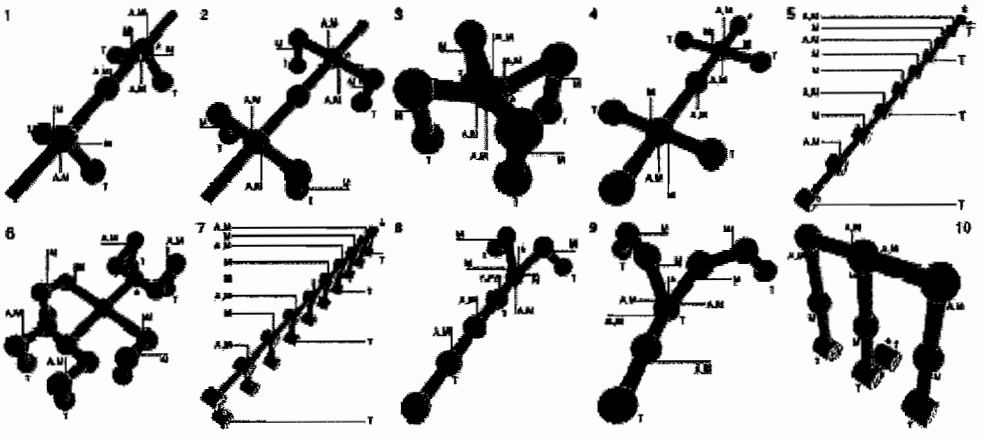


Figure 24. Ten creatures evaluated for locomotory efficiency.

Note. From "A Method for Isolating Morphological Effects on Evolved Behaviour" by J. C. Bongard and R. Pfeifer, 2002, *Proceedings of the Seventh International Conference on the Simulation of Adaptive Behavior*, p. 306.

Creature behavior was generated using an ANN. Each creature's neural network consisted of an input node for each of the eight sensors, three hidden nodes, and an output node for each of the eight actuators. A simple threshold model was used to determine neuron activation based on input link weights. A GA was used to evolve the neural network weights. In this way, the creatures were able to learn how to use their neural networks, but not evolve the neural networks themselves.

All 10 creatures were evolved to generate some form of forward locomotion. Quadrupedal and hexapedal creatures generated the fastest forward locomotion. Tripedal creatures also produced fast locomotion. Creatures with a greater number of limbs and long, snake-like creatures generated the slowest locomotion. This result was likely due to additional friction caused by having more body points in contact with the ground. Another possible explanation is that it was more difficult for the neural network to coordinate larger numbers of limbs. This theory was supported by showing that increasing the number of hidden layer neurons increased the forward locomotion speed achievable by the highly-limbed and snake-like creatures.

ALife techniques have been used to generate interesting new creatures with equally interesting new methods of producing locomotion. In all of the presented work, ANNs were utilized to control joints. Creatures have been evolved based on individual performance, by survival of the fittest, or by user interaction. Observing the successes and failures of a creature trying to walk or swim gives some insight into the advantages and disadvantages of certain gaits. For more information on ALife techniques for character animation, see Taylor's (2000) review of the area.

Robotics

Robotics is a large research area unto itself. Over the past decade, researchers in the field of Evolutionary Robotics have developed techniques for self-learning and self-designing robots. For the purpose of this review, current state-of-the-art work will be presented on methods for evolving locomotion in robots. Azevedo (2001) discussed correlations between the studies of human and robot locomotion. Azevedo argued that biomechanical observations can improve robot stability and reduce energy consumption. Similarly, robotics can be used to evaluate prosthetic limbs and test alternate joint configurations and gait strategies to aid in the study of human locomotion.

Golubovic and Hu (2002) used GAs to optimize locomotion for the Sony AIBO robot. The AIBO was a quadrupedal robotic pet intended to behave like a cat or dog. The robot had the ability to emulate instinct, emotion, and learning capabilities. Each AIBO learned from experience during a training phase, settling on a final set of behaviors when it reached virtual maturity. The locomotion engine for the AIBO was robust, allowing the robot to walk forward, turn, follow targets, and get back up after falling over.

GAs were used to tune the AIBO locomotion engine for maximum speed and stability. The gait generator utilized a set of six parameters for the front limbs and six parameters for the hind limbs. These parameters were used to generate cyclic motions for each end effecter that roughly followed an elliptical path. The governing parameters were the width and height of the path, the x, y, and z coordinates of the center of rotation for the path, and the angle of the paw relative to the ground. A thirteenth parameter was used to specify the forward velocity of the robot. The parameter set was evolved to maximize

forward velocity and stability. To measure stability, readings were taking from three gyro sensors during trial runs. Lower variance between readings indicated higher stability. Figure 25 shows the configuration of the AIBO forelimb.

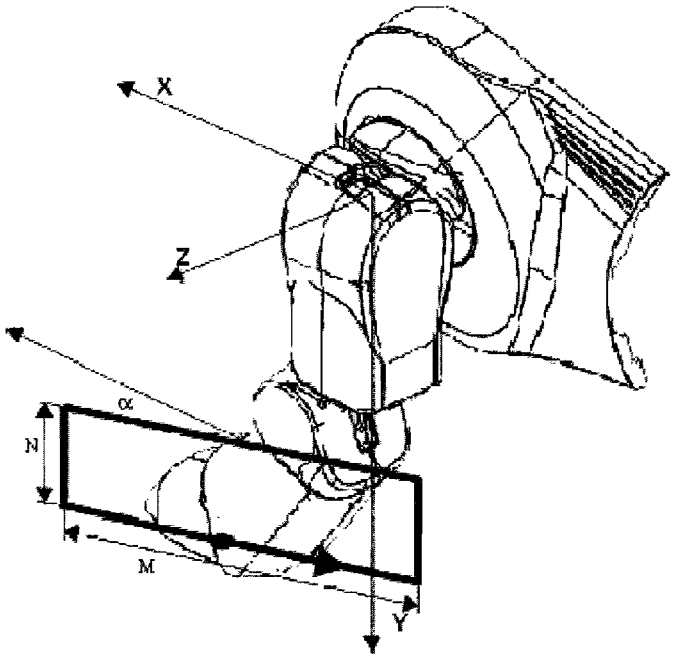


Figure 25. AIBO forelimb.

Note. From "A hybrid evolutionary algorithm for gait generation of Sony legged robots" by D. Golubovic and H. Hu, 2002, *Proceedings of the 28th Annual Conference of the IEEE Industrial Electronics Society, 4*, p. 2595.

Locomotion evolution experiments were conducted in hardware on the AIBO itself. This approach diverged from the typical simulation approach. Golubovic and Hu admit

that running simulations would have been easier, but contend that simulation may miss important interactions between the robot and its environment. A good physics simulation should produce negligible errors if mass, inertial, and environmental conditions are properly modeled. Simulation saves both time and hardware costs, but may neglect subtle physical effects.

Wolff and Nordin (2003) utilized GP to evolve locomotion on a humanoid robot named Elvina. Elvina had 12 joint DOFs, three per limb. Locomotion was generated by interpolating a set of whole-body pose configurations. Locomotion was evaluated in both simulations and hardware. Balance was maintained during locomotion by keeping the projection of the COM onto the ground within the robot's support polygon. The robot's upper limbs were utilized to offset its COM while the lower limbs were used for locomotion.

Genetic programs resembling assembly language were used to produce locomotion. The programs took the robot's current pose configuration as input and computed the robot's next pose configuration. Each genetic program consisted of a maximum of 256 instructions. Instructions contained an operator (add, subtract, multiply, divide, and sin), source registers, and a destination register. The evolved sequence of poses was cycled to produce continuous locomotion. Gaits were evaluated based on the robot's ability to remain upright and travel forward quickly. The fitness function was thus based on maximum distance covered in a set amount of time and minimal change in robot head height.

The approach successfully generated forward locomotion on both the physical and simulated robot. These experiments show the utility of the GP paradigm by evolving a complete locomotion program from scratch. The GP approach is well suited for evolving programs that can be run in hardware. Wolff and Nordin point out that while evolving on actual hardware is more accurate, it is time consuming and the hardware upkeep price can be high. Simulating early evolution followed by subsequent evolution of good programs on real hardware may be an economical compromise.

Zhang and Vadakkepat (2003) used GAs to evolve locomotion for RoboSapien, a 12-DOF humanoid robot. RoboSapien could walk continuously on flat ground and climb stairs while avoiding obstacles. The robot used the Zero Moment Point (ZMP) method (Vukobratovic and Juricic, 1969) for maintaining balance. The ZMP is the point on the ground where the sum of all active force moments is zero. To maintain balance, the robot's ZMP must stay within the support region. This method is effectively equivalent to keeping a character's COM above the support region.

Locomotion was generated by calculating hip and pes trajectories. The horizontal component of the hip was calculated using the support phase period, the swing phase period, and the total step period (implicitly giving the dual support period). The horizontal component of the swing leg trajectory was calculated by taking into account the swing phase period and the total step period. The height component of the hip and swing leg trajectories was determined by checking for obstacles in the leg's path. Trajectories were stored as cubic polynomials. The set of parameters governing the hip and pes trajectory

calculation were evolved to maximize robot balance by minimizing the distance between the actual and optimal ZMP.

The gait generation strategy was used to generate locomotion in hardware. The robot was able to maintain balance while walking forward and avoiding obstacles. The robot was also able to maintain balance while climbing stairs. The robot's motion was limited to forward movement, simplifying the motion equations but limiting the robot's usability. The approach to calculating hip trajectory was similar to that used by Chung and Hahn (1999). Figure 26 shows the hip and pes trajectories while avoiding an obstacle. Hip motion was decomposed into horizontal and vertical trajectories, so adding a lateral trajectory should be tractable.

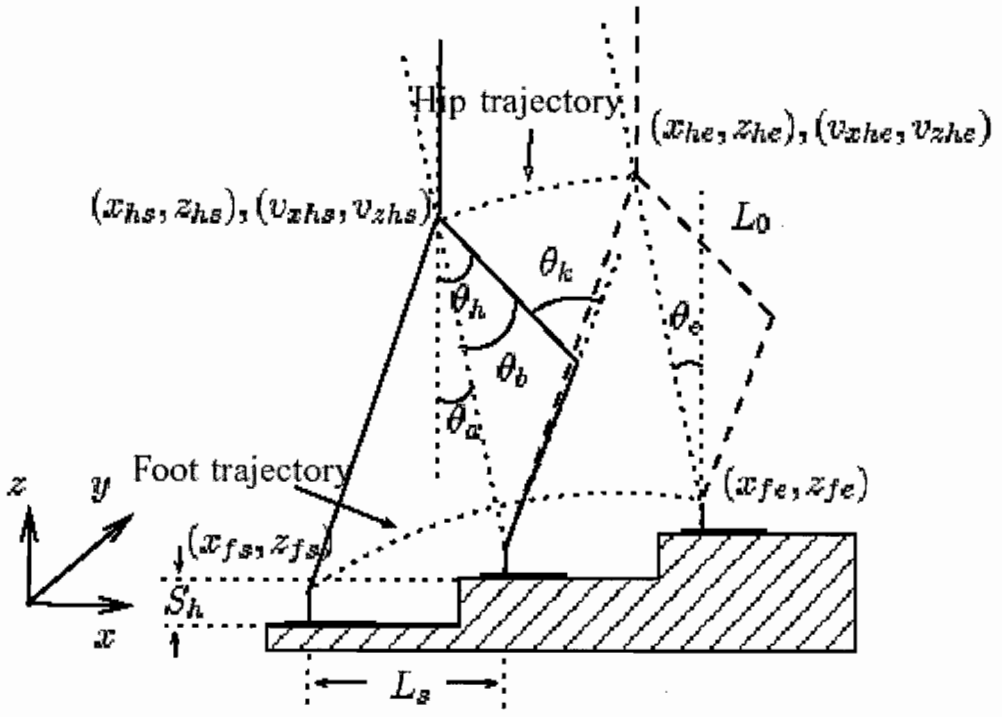


Figure 26. Hip and pes trajectories during obstacle avoidance.

Note. From "An evolutionary algorithm for trajectory based gait generation of biped robot" by R. Zhang and P. Vadakkepat, 2003, *Proceedings of the International Conference on Computational Intelligence, Robotics and Autonomous Systems*, p. 3.

Wyeth and Yik (2003) used the ZMP method, GAs, and multiple-DOF joints to produce locomotion for the GuRoo robot. GuRoo operated using 23 DOFs, 15 of which were used in producing locomotion. The hip joints used 3 DOFs, the knee joints used 1 DOF and the ankle joints used 2 DOFs. The remaining DOFs were used to drive the head, neck assembly, and the arms. The robot was capable of shifting its weight, walking, turning, shaking hands, and waving.

Locomotion was generated by evolving a set of key pes locations. Eight key locations were used to fully specify the walking motion of one leg (two swing phase locations, four support phase locations, and two transitional locations). The key locations were then mirrored and applied out of phase to drive the contralateral leg. Figure 27 illustrates the key pes locations. A GA was then used to evolve a locomotion sequence that minimized the difference between actual and optimal ZMP trajectory. The key locations were normalized so that they could be scaled to adjust stride length and pes clearance height during locomotion.

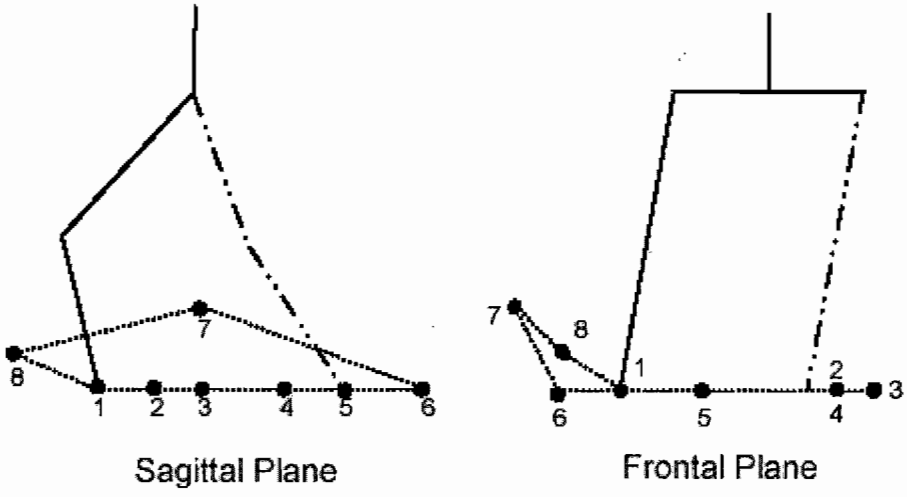


Figure 27. Key pes locations.

Note. From "Evolving a locus based gait for a humanoid robot" by G. K. Wyeth and D. T. F. Yik, 2003, *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003)*, p. 1640.

Experimental results showed that it is possible to evolve a stable forward walk by specifying only pes locations. By specifying pes locations and balancing the robot's body, the robot could have followed a trackway. Similar method could be used to evolve trackway-following motion sequences for virtual characters. IK was used to solve for knee and hip configurations, which was possible due to the use of simple hinge and ball and socket joints. The hip height was fixed; adding a hip height parameter would allow more variation in the evolved gaits.

Recent methods for evolving robot locomotion have focused on the use of GA and GP. The robotics techniques presented use sensor input in determining phenotype fitness, but do not use sensor input to vary responses at execution time. To provide more complex behaviors, such as following, a sense and response system could be evolved using GP or ANNs. In this way, character animation techniques can be valuable to robotics. Similarly, stable walking techniques based on new genotype representations could be useful for character animation.

Computational Gait Analysis

Computational methods have aided the study of biomechanics by providing virtual 3D environments for conducting biomechanics experiments and visualizing results. 3D visualizations allow an investigator to arbitrarily rotate a model and zoom in on points of interest without comprising image quality, a vast improvement over previous 2D methods. Models are typically built from geometric primitives or bone shape data obtained from 3D scanners. Locations of articulation (i.e., joints) within the model are animated using simple

forward kinematics, Motion Capture data, or complex dynamic and/or musculoskeletal models. Biomechanical models and methods are particularly valuable for locomotion models in that they add biological and mechanical credibility.

The Software for Interactive Musculoskeletal Modeling (SIMM) project was introduced by Delp and Loan (1995) and later revisited with an improved feature set (Delp and Loan, 2000). SIMM allowed user to visualize, edit, and experiment on 3D biomechanical skeletons. Models consisted of a bone file containing 3D bone shape information, a joint file specifying the kinematics of the model's joints, and a muscle file specifying muscle-tendon parameters. SIMM represented joint kinematics using three orthogonal translations and three rotations about arbitrary axes, providing six total DOFs. The order in which the rotations and translations were applied affected joint behavior and needed to be specified by the user. Specifying the order of 3D transformations can be unintuitive and should be consistent between models, and it is still an important issue in 3D modeling today.

A generalized coordinate is a variable with some range of values. SIMM joints were assigned a number of generalized coordinates based on their behavior. A hinge joint used one generalized coordinate while a ball-and-socket joint utilized three generalized coordinates. A joint's six DOFs were controlled by manipulation of its generalized coordinates through the use of kinematic functions, one function per DOF. These functions were defined by spline curves which could be edited using a graphical interface. An elbow, for example, had one generalized coordinate representing its flexion/extension movement. Kinematic functions mapped the elbow flexion/extension coordinate to rotation about an

oriented axis, translation in the *x* direction, and translation in the *y* direction (relative to the proximal bone).

The basic idea of a generalized coordinate is widely used in animation to define hierarchies of movement. At the highest level, a generalized coordinate representing the animation timeline is used to drive all child animations. A drawback to the approach used by SIMM is that the user had to decide which DOFs should be controlled by each generalized coordinate. The number of generalized coordinates that could be associated with a joint was limited by the joint's six DOF; each joint DOF was controlled by at most one generalized coordinate. Due to this restriction, multiple generalized coordinates could not cause rotation about the same axis, potentially forcing non-biological orthogonality into the joint model. To avoid this limitation, users could have built compound joints by utilizing multiple individual joints with additive effects between bones.

SIMM provided two important high-level functions. By specifying joint kinematics and a musculoskeletal model (including muscles, ligaments, and tendons), a user could build an animation timeline by providing muscle activation values as a function of animation time. SIMM would then simulate and animate the resulting joint kinematics. Inversely, given a skeletal animation (inputted directly using keyframes or from Motion Capture data), SIMM could compute the muscle forces and moments necessary to achieve the animation. Figure 28 illustrates a simulation of the human pectoralis major muscle, including the muscle lines of action. SIMM is now widely used in biomechanics laboratories around the world.

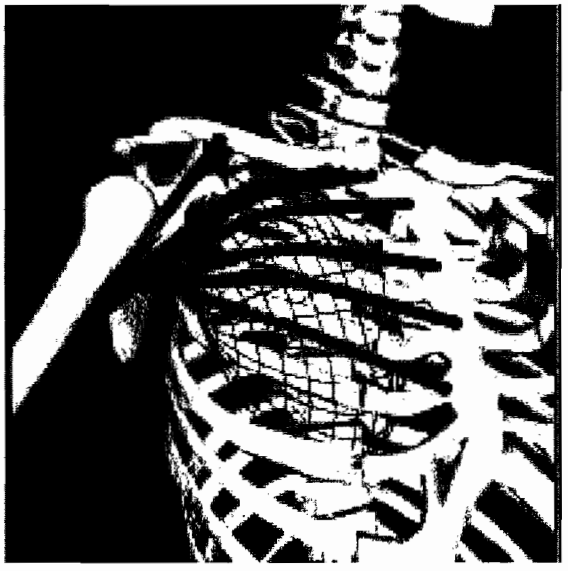


Figure 28. SIMM model of the pectoralis major muscle.

Note. From "A computational framework for simulating and analyzing human and animal movement" by S. L. Delp and J. P. Loan, 2000, *Computing in Science & Engineering [see also IEEE Computational Science and Engineering], 2*(5), p. 48.

SIMM has been applied to simulate the behavior of knee implants during a stepup exercise (Piazza and Delp, 2001). Simulated motions of the implants were observed while manipulating of the tibiofemoral and patellofemoral knee joints. The simulation used recorded muscle activations as input and predicted joint kinematics using the forces and moments generated by the muscles, ligaments, tendons, and contact between implant surfaces. Implant surfaces were represented by polyhedral meshes obtained by exporting triangulated geometries from CAD representations. During the simulation, the number of contact points between articulating surfaces was allowed to vary, providing an accurate

estimation of the contact forces between surfaces. Figure 29 illustrates the biomechanical model used for evaluating the implants during the stepup exercise.

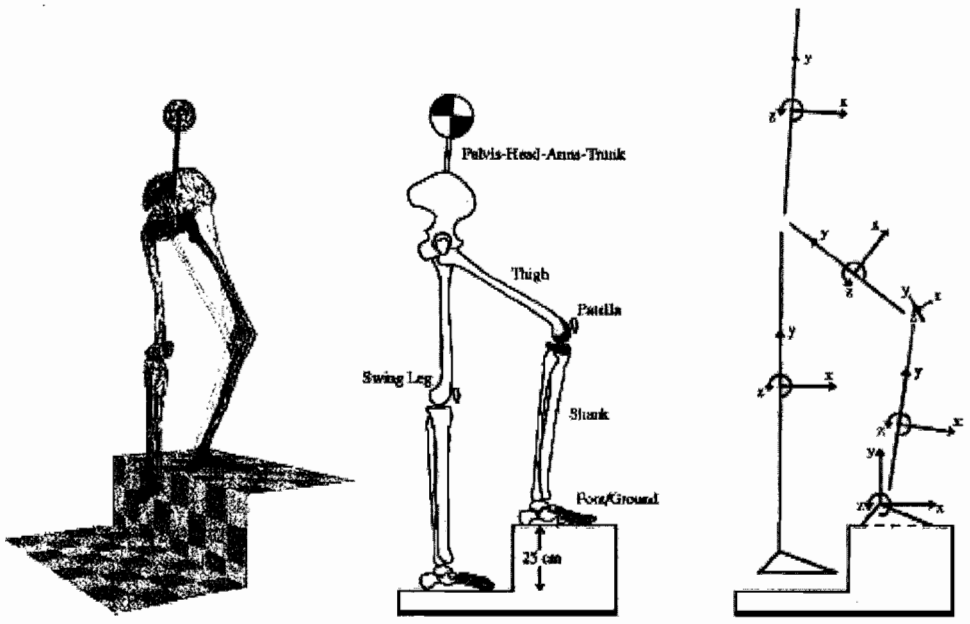


Figure 29. Biomechanical model used for the stepup exercise.

Note. From "Three-Dimensional Dynamic Simulation of Total Knee Replacement Motion During a Step-Up Task" by S. J. Piazza and S. L. Delp, 2001, *Journal of Biomechanical Engineering, 123*, p. 600.

Complex musculoskeletal models are necessary for precision studies such as evaluating the effects of prosthetic implants. Such models are useful for studying and generating locomotion, but construction of the model is often prohibitively laborious with respect to the application. Sufficient data for reconstruction of a complete musculoskeletal

model of sometimes not even available, as can be the case with extinct species. Simple kinematic joints can, in principal, emulate the functionality of a complex musculoskeletal joint provided that the joint's ROM was constructed using appropriate biological and physiological constraints.

Another system for modeling biological joints was introduced by Maciel, Nedel, and Dal Sasso Freitas (2002). In addition to standard planar, uniaxial, biaxial, and polyaxial, the system provided a sliding-axis joint. The sliding-axis joint allowed the instantaneous axis or a hinge joint to be moved along a parametric curve when rotating the joint. Joint motion was represented by specifying a minimum, neutral, and "comfortable" configuration for the joint. Final joint axes and angles were therefore interpolates of these input configurations. Figure 30 shows a knee modeled using a sliding-axis joint.

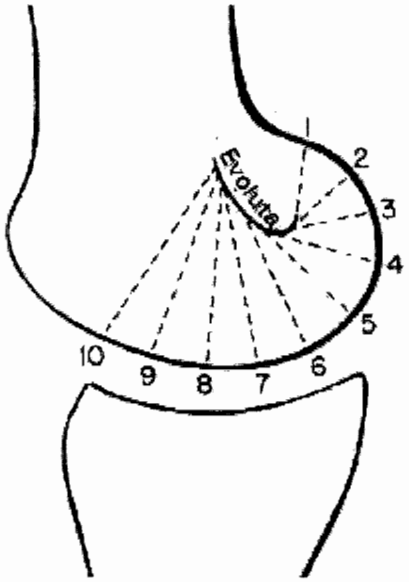


Figure 30. Knee represented using a sliding-axis joint.

Note. From "Anatomy-based joint models for virtual human skeletons" by A. Maciel, L. P. Nedel, and C. M. Dal Sasso Freitas, 2002, *Proceedings of Computer Animation, 2002,* p. 224.

Hutchinson et al. (2005) used SIMM to create a musculoskeletal model of the *Tyrannosaurus rex* hindlimbs. The model consisted of ten joint DOFs (i.e., flexion/extension, abduction/adduction, medial/lateral rotation) and 33 muscle groups crossing the hip, knee, ankle, and toe joints of each limb. Muscle groups were modeled using bone geometry data by specifying joint rotation axes, muscle attachment locations, and muscle-tendon geometry and paths. Figure 31 shows the *Tyrannosaurus rex* hindlimb musculoskeletal model.

Figure 31. Tyrannosaurus rex hindlimb musculoskeletal model.

Note. From "Analysis of hindlimb muscle moment arms in Tyrannosaurus rex using a three-dimensional musculoskeletal computer model: implications for stance, gait, and speed" by J. R. Hutchinson, F. C. Anderson, S. S. Blemker, and S. L. Delp, 2005, *Paleobiology, 31*(4), p. 682.

The hindlimb musculoskeletal model was used to determine and analyze the flexor and extensor muscle moments about the limb joints. Muscle moment arms were evaluated based on several static limb poses with varying sagittal elevation angles (i.e., flexion/extension of the hip). Results showed that more upright poses have significant

mechanical advantage of the joints when compared to less upright poses. This result seems intuitive because non-columnar limbs cause potentially unnecessary torques about the limb's joints. These torques are caused by a joint's position being noncollinear with the Ground Reaction Force (GRF), which is related to the muscle moment arms necessary to stabilize an animal.

Results also showed that the static moment arms are not of the magnitude expected from a proficient runner, providing further evidence that *Tyrannosaurus rex* was not a fast runner. These results provide insight into the necessary muscle mass to support *Tyrannosaurus rex* while standing, which is relatively small compared to the estimated muscle mass necessary for *Tyrannosaurus rex* to run quickly. The muscle-moment-arm and mass estimates are based on static poses, so are not based on a fully-dynamic musculoskeletal model. These methods do not recreate gaits, but allow an analysis of the muscle masses necessary for gaits.

Hutchinson and Gatesy (2006) explored possible hindlimb configurations of *Tyrannosaurus rex* during locomotion and while standing. They identified a mid-stance configuration for the each limb based on the limb's GRF. Specifically, a limb's GRF points up and back as the limb makes contact with the ground and points up and forward as the limb accelerates the body; mid-stance is a limb joint configuration when the GRF is vertical. At mid-stance, there is a family of solution configurations based on the hip height of the animal. They observed that optimal hip height cannot be determined from bone osteology alone, because the animal has limbs flexible enough to allow the animal to lie down.

Once hip height has been determined, there is still a family of limb configuration solutions possible based on a fixed hip and toe positions. Given hip, knee, ankle, and main toe joints with 90° of flexion/extension (i.e., four DOFs) sampled at a 1° resolution, there are more than 65 million possible joint configurations. Figure 32 illustrates the families of possible hindlimb configurations. Hutchinson and Gatesy suggested the use of GRF-based pruning criteria to remove unlikely joint configurations from the solution space. For example, configurations with the GRF in front of the knee were excluded. Configurations were also removed that contained any joint such that the GRF's moment arm about the joint exceeded a maximum value. The maximum moment arm value was based on the moment arm that could be generated if 5% of the animal's body mass was dedicated to muscles crossing the joint.

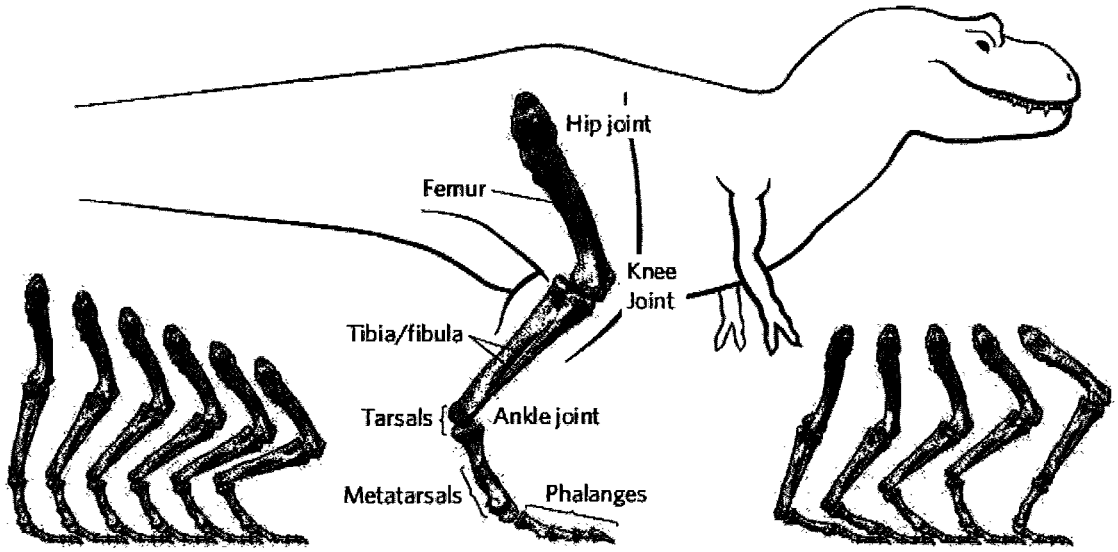


Figure 32. Possible mid-stance configurations.

Note. From "Beyond the bones" by J. R. Hutchinson and S. M. Gatesy, 2006, *Nature(London), 440*(7082), p. 292.

When exploring the space of all possible limb configurations, it is necessary to prune the space to make searching it tractable. Some animals have limbs that utilize up to eight discrete DOFs. In the above example, sampling four 90° DOFs (at a 1° resolution) creates more than 65 million limb configurations. Adding another 90° DOF increases the size of the space exponentially, expanding it to almost six billion configurations. Increasing the space to six 90° DOFs expands the space to over 530 billion configurations. Pruning the space using GRF-based constraints is intuitive, but requires dynamic simulation to accurately determine COM and GRF values. Dynamic

simulation is relatively-expensive computationally, so simulation of large numbers of configurations can be intractable.

Henderson (2006) used trackway data to recreate quadrupedal gaits. Gait key poses were created by manually positioning joints. A system of partial differential equations was then used to derive the joint angles necessary to achieve joint positions (i.e., IK). Joints were positioned such that the manus and pes positions and orientations approximated the trackway data. The limbs were additionally constrained such that there was no vertical displacement of the body during locomotion.

Due to the large masses of the dinosaur and elephant models, gaits were constrained based on the stability of the static poses. Specifically, only a single limb was allowed to be in the swing phase at a time. In addition, the COM was constrained to lie within a triangular region defined by the manus and pes positions of the three supporting limbs, called the Stability Triangle. Intuitively, wider-stance trackways provided a larger Stability Triangle than more narrow trackways. Results showed that the models with a centrally-located COM were most stable in wide trackways while models with a more posterior COM were most stable in narrow trackways. Figure 33 shows a *Brachiosaurus* model with its associated trackway, COM, and Stability Triangle.

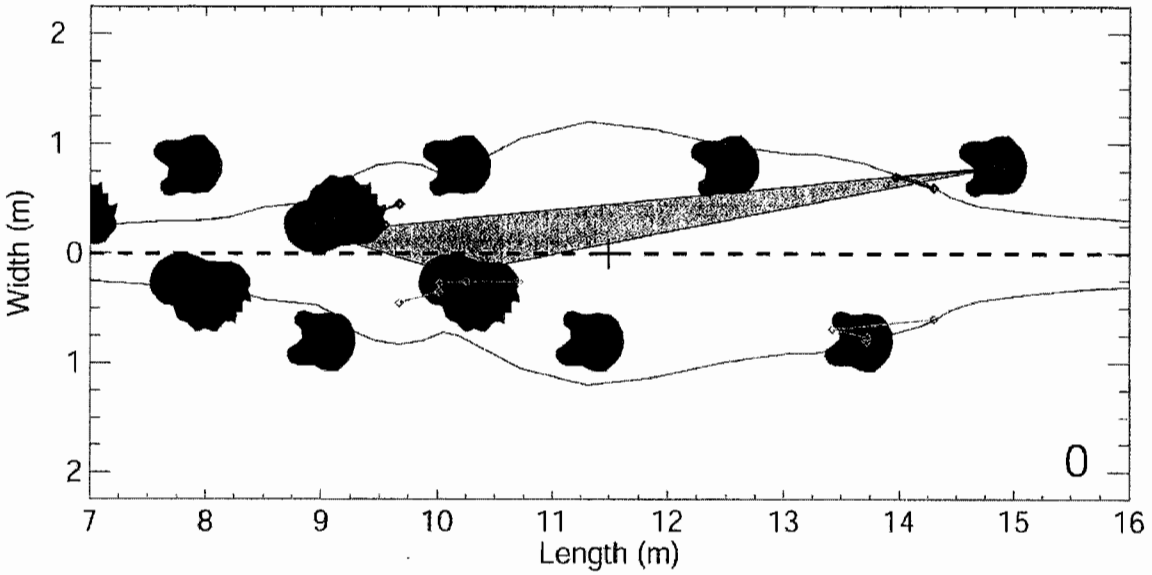


Figure 33. *Brachiosaurus* trackway, COM, and Stability Triangle.

Note. From "Burly gaits: centers of mass, stability, and the trackways of sauropod dinosaurs" by D. M. Henderson, 2006, *Journal of Vertebrate Paleontology, 26*(4), p. 917.

In a static pose, a quadruped will tip over if its COM is not within its Stability Triangle. As a corollary, maintaining balance with four supporting limbs is relatively easy. Maintaining balance with three supporting limbs is more difficult but still very plausible. Maintaining balance with two supporting limbs is only possible if the COM lies in the plane connecting the two supporting limbs (and perpendicular to the ground), implying that an animal is only statically stable on two supporting limbs if those limbs are diagonal (e.g., right hindlimb and left forelimb) and the animal exerts enormous effort to balance itself.

During locomotion, however, the animal is always moving and never statically posed. The position of the COM and supporting manus/pes is therefore not enough information to determine stability; the dynamics of the COM (i.e., velocity and acceleration) must also be considered to prevent overly conservative stability estimates. For example, an animal's COM may be outside the animal's Stability Triangle but the COM's velocity might be such that the COM will be back within the Stability Triangle very quickly. Such a situation could be considered statically instable but dynamically stable.

Sellers and Manning (2007) extended earlier work (Sellers et al., 2004) by adding elastic elements (Hill, 1938) to their muscle model. In addition, several changes were made to their GA implementation (i.e., using the Open Dynamics Engine in favor of Dynamechs) for improved computational performance and stability. These new methods were used to determine the maximum running speeds of five extinct bipeds (i.e., *Compsognathus*, *Velociraptor*, *Dilophosaurus*, *Allosaurus*, and *Tyrannosaurus*) and three extant bipeds (i.e., *Dromaius*, *Struthio*, and *Homo*). Figure 34 shows evolved gaits of the various models.

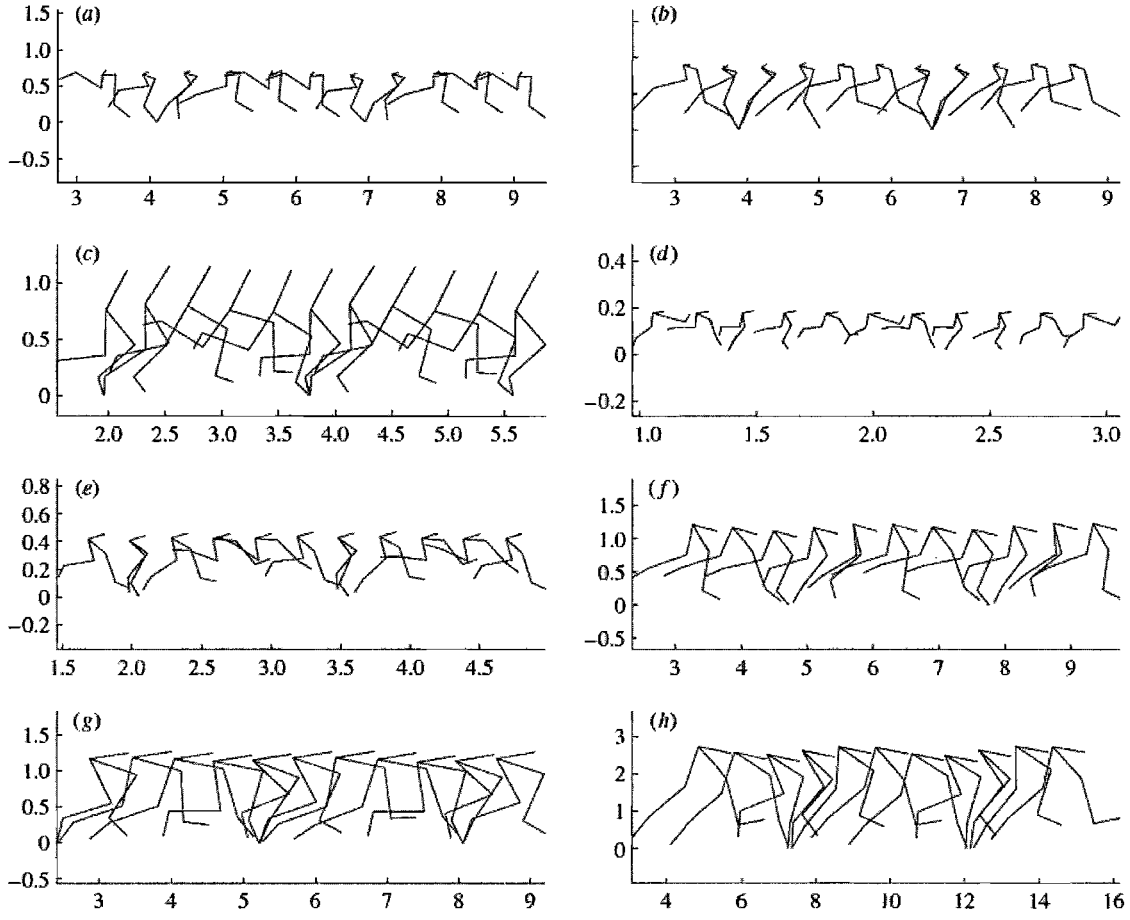


Figure 34. Evolved gaits of various models.

Note. From "Estimating dinosaur maximum running speeds using evolutionary robotics" by W. I. Sellers and P. L. Manning, *Proceedings of the Royal Society B: Biological Sciences, 274*(1626), p. 2713.

Fixed-length candidates were used to represent 61 parameters: five activation levels for 12 muscles (i.e., six per limb) and a parameter representing the cycle time. The candidates represented ten key-frames during the stride; the left-right muscle activation values swapped for the second half (i.e., keyframes 6-10) of the stride. The GA fitness

function was based on maximum running speed achieved during a fixed simulation time interval. Therefore, candidates that produced fast running speeds were rewarded while candidates that caused the model to fall over were heavily penalized.

GA populations consisting of 1000 candidates were evaluated for up to 1000 iterations. The GA loop was interrupted early if a steady maximum forward velocity was maintained between iterations. The GA process was repeated at least five times to ensure that a near-optimal candidate was found. Furthermore, the entire process was repeated at least 20 times with the fittest candidates from the previous run seeding the population for the next run. This process took between several days to several weeks to run on a modern parallel supercomputer (as of this printing).

Results showed that top speeds achieved by the simulated extant models closely corresponded to published top speeds. This result indicates that the top speeds achieved by the extinct models are likely also accurate, depending on the accuracy of the musculoskeletal model. The uncertainty of muscle parameters was explored using a sensitivity analysis; muscle masses were varied over the range [2.5%, 7.5%], showing that total muscle mass has a linear effect of maximum velocity. The sensitivity of muscle contraction velocity parameters was also tested.

This work demonstrates the possibility of evolving gaits for a variety of models using a fairly simple hindlimb muscle model; two summary muscle groups (i.e., flexor and extensor) were simulated for each limb's primary joints. The algorithm takes a considerable amount of CPU time to complete. This cost is due to the number of iterations and reseedings necessitated by the massive size of the search space (i.e., 488-bit candidates

assuming eight-bit floating point resolution for the 61 parameters). In addition, the algorithm is expensive because it requires a dynamic simulation for each candidate evaluation.

Gait analysis techniques are valuable to both biomechanics and animation. Musculoskeletal models are laborious to construct but allow accurate reproduction of muscle forces and torques, provided the muscle parameters (i.e., muscle mass, contraction velocity, and elasticity) are represented realistically. Also, GRF-based and stability-based constraints can be used to prune the space of possible limb configurations for locomotion. In principal, these constraints could be used in conjunction with GA methods to reduce the computational cost associated with evolving optimal gaits.

Summary

Current state-of-the-art techniques for evolving locomotion are based GAs, GP, or ANNs. GAs are useful for evolving a fixed set of gait cycle parameters. Gait cycle parameters are typically used as input to pattern generators that produce periodic locomotion. The pattern generators produce forward walking animations, although the Limit Cycle Control method (Laszlo et al., 1997) allowed turning by perturbing open-cycle motions. GAs are well suited for cases where little user and environmental interaction is required, such as exploring alternative gait strategies.

GP and ANNs are useful for systems that require a character to respond based on input stimuli. ANNs have been used predominately to link stimuli to responses. Genetic programs can also be used to directly modify character configurations based on input

criteria. It is important to note that neural networks and genetic programs can only be effectively evolved to perform one primary task. For example, a virtual creature can be trained to achieve a complex goal such following a light source with a specific gait, but it is unlikely that the creature would be able to change gaits or perform other tasks using the same neural network or genetic program.

Musculoskeletal models can provide biologically-accurate representations of joint motion, depending on the complexity and accuracy of the models. Musculoskeletal models can be utilized by GAs to automatically create gait animations, but the creation of models is difficult due to model complexity and availability of data. Evaluation of the models is computationally expensive due to the use of physics simulation to determine the fitness of each candidate gait and the extremely-large solution spaces needed to represent all possible combinations of muscle actions.

GRF and trackway constraints can be used to prune the space of possible solutions, but evaluation of a limb's GRF also requires physical simulation. Constraining the space with respect to mid-stance limb configurations may seem intuitive, but mid stance may be the time during the stance phase that the limb is most osteologically underconstrained. At mid stance, limbs are capable of a number of leaning, bending, and squatting motions. Conversely, limbs are much more osteologically constrained as they reach forward and backward at the beginning and end of stance.

In the next chapter, methods will be presented that utilize constraints derived from the biomechanics of tetrapod limb joints and gaits that involve more than one limb in contact with the ground to constrain the space of possible limb configurations at the

beginning and end of stance. A GA will then be utilized to quickly find smooth, and therefore plausible, paths through the constrained spaces to automatically generate gait animation sequences. The GA evaluates candidate fitness based on the overall smoothness of limb movements, so no physical simulation is necessary. The result is a set of algorithms that generate plausible walking gait animations with very little computational cost.

CHAPTER III

METHODS AND MATERIALS

The GAGA techniques and methodologies are presented is this chapter. GAGA automatically generates forward bipedal and quadrupedal walking gaits using biomechanically-accurate skeletons. A representation for discrete joint movement is first presented. Next, joint movements are combined to explore the kinematic capabilities of a limb. The kinematic capabilities are then analyzed and constraints are applied to allow the use of GAs to find paths through the spaces of limb configurations. GA-based methods are then presented that quickly create efficient bipedal and quadrupedal forward walking gaits. The chapter concludes with descriptions of the models used for the investigations presented in the next chapter.

Functional Degrees of Freedom

In the absence of dynamics, the kinematics-only models are completely rigid until flexibility is added to the joints. Flexibility is added at joints by specifying ROMs. A ROM is categorized by the number of functional movements that its corresponding joint

is capable of performing. Each of these functional movements is represented by a Functional Degree of Freedom (FDOF). A shoulder ROM, for example, has three FDOFs representing: flexion/extension, abduction/adduction, and medial/lateral rotation; a knee has one FDOF representing its flexion/extension movement. The FDOF framework is similar to the generalized coordinate framework used by SIMM (Delp and Loan, 1995).

Each ROM manipulates all six of the joint's geometric DOFs (i.e., three orthogonal translations and three Euler angle rotations about fixed orthogonal axes). The geometric DOFs manipulated by the ROM are applied relative to the joint's parent joint in the model skeleton. Three 6-DOF keyframes are used to specify each FDOF. The three keyframes represent the extremes of movement and a neutral position for the movement, similar to Maciel (2002). All neutral FDOF configurations within a ROM must be coincident. Based on input FDOF values, a ROM uses cubic spline interpolation to determine intermediate configurations for each movement. The configurations are then combined to determine the final output ROM configuration. Figure 35 shows the flexion/extension movement of an *Apatosaurus* elbow from right-lateral view.

Figure 35. Flexion/extension movement of an *Apatosaurus* elbow.

FDOFs are a powerful representation, allowing a continuum of movement granularity from simple, idealized hinge or universal joints to a movement path constrained by musculoskeletal lines of action. In the next section, techniques will be presented that combine all of a limb's FDOFs to determine a total ROM for that limb.

Limb Ranges of Motion

To determine the effect of FDOFs on locomotion, the manus and pes are first positioned and oriented on the ground based on trackways. All combinations of FDOFs are then evaluated at a resolution between 4° and 6°, depending on the number and complexity of the FDOFs. Translation-only FDOFs (e.g., scapulothorax elevation) are evaluated at an appropriate resolution based on each model's dimensions (e.g., 5cm for *Apatosaurus*). The position and orientation of the manus and pes are maintained fixed on the ground throughout the evaluation process (see Appendix F).

Thompson and Holmes (2007) used a similar approach to recreate a possible walking gait cycle for a *Chasmosaurus* forelimb. Polyester resin casts were made from fossilized bones, and those casts were articulated using thin, flexible wires. The manus was then held fixed on the ground while the limb was manually exercised to determine plausible limb configurations during the cycle. The body was not allowed to be displaced laterally or vertically and the scapula was not allowed to move on the rib cage during the cycle, which would have unnecessarily overconstrained the gait.

Repositioning and reorienting a manus or pes on the ground to match its original position and orientation, similar to Sticky IK methods (McKenna and Zeltzer, 1990), causes all proximal elements to be repositioned and reoriented. Figure 36 demonstrates this effect by showing flexion/extension of the *Apatosaurus* forelimb while maintaining a fixed manus position and orientation. For each combination of FDOF values, an element proximal to the forelimb or hindlimb system (e.g., the 3rd dorsal for the *Apatosaurus* forelimbs and the 1st sacral for the *Apatosaurus* hindlimbs) is monitored for position and orientation. This proximal element is referred to as the root element of the system. The set of all possible root element positions and orientations reachable by exercising the limb's FDOFs is called the Limb Range of Motion (LROM).

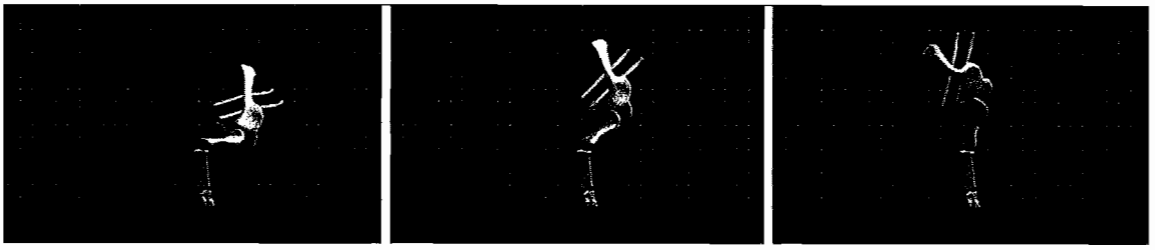


Figure 36. Apatosaurus elbow flexion/extension with fixed manus.

LROMs can be organized for convenient access by first determining the Axis-Aligned Bounding Box (AABB) that contains all of the LROM samples (by position). The 3D space within the AABB is then subdivided into a number of boxes with constant dimensions. Each box is populated with all LROM samples such that the sample's position is inside the box. The 3D grid of boxes, in summary containing all LROM samples, is called the *LROM space*. An LROM space can be visually represented by finding the LROM sample in each LROM space box that requires the least Root Mean Square (RMS) orientation change to lie within the box's bounds. Figure 37 shows a visualization of the *Apatosaurus* right forelimb LROM space, with minimum RMS orientation change represented by color (i.e., 0° < green < 15° < purple < 30° < yellow < 45° < red).

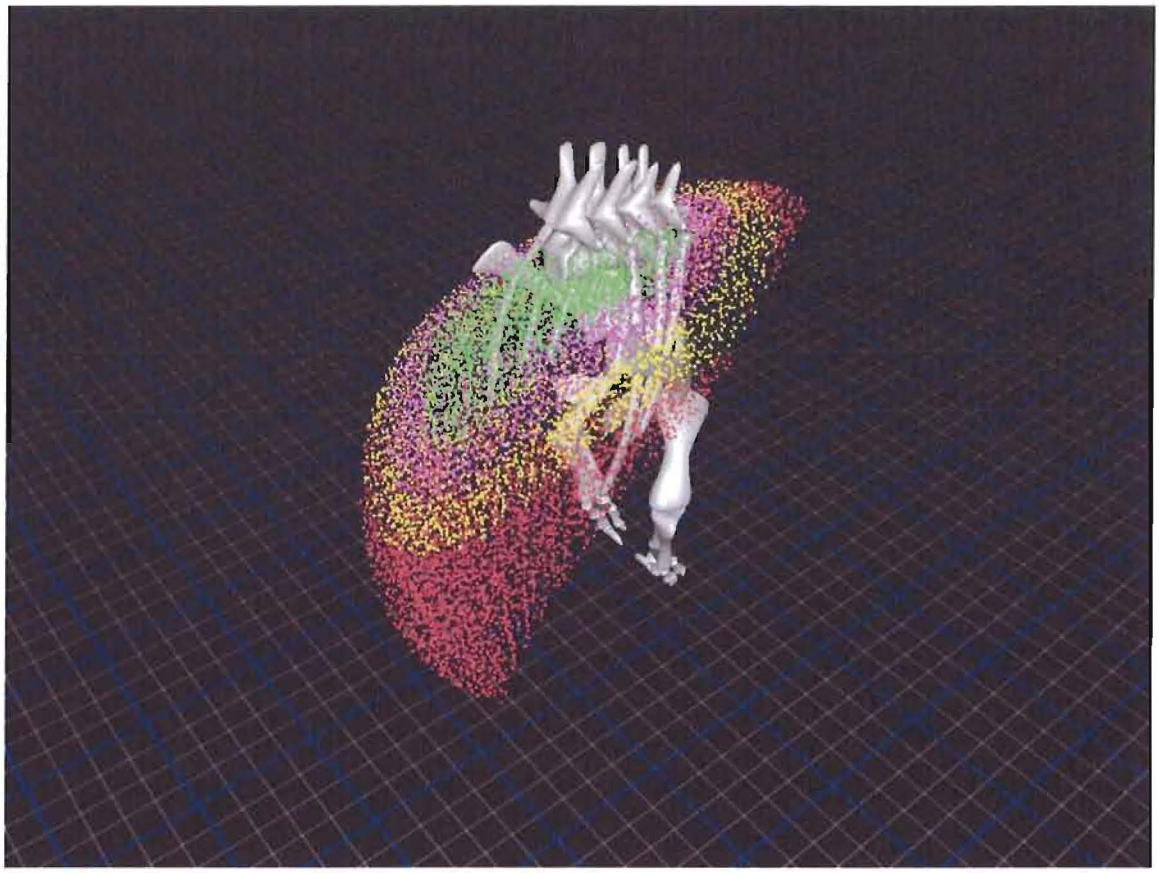


Figure 37. Visualization of the *Apatosaurus* right forelimb LROM space.

LROM spaces allow the representation and visualization of the total ROM that a limb is capable of operating within. Limbs do not act in isolation during locomotion; the LROM space that represents all possible root positions and orientations when both left and right limbs are in contact with the ground can be summarized as the intersection of the left and right LROM spaces. Methods for applying bipedal constraints with respect to locomotion will be presented in the next section.

Bipedal Gait Reconstruction

The following three sections describe the algorithms used to automatically generate bipedal walking gaits: Constraints must first be applied to the LROM spaces to ensure that locomotion does not effectively pull the body apart. A GA then finds smooth, plausible paths through the constrained space. Finally, a pipeline architecture will be presented that maximizes reuse of precomputed data, minimizing unnecessary repeat computational operations.

Constraints

The responsibilities of each limb during locomotion can roughly be divided into two phases: the limb supports the animal's mass and propels the animal forward during the stance phase (also referred to as the support phase); the limb is off the ground preparing for the next stance phase during the swing phase (also referred to as the suspended phase or step phase). The relative amount of time that a limb spends in the stance phase is the limb's *duty factor*. The duty factor is a normalized term (on the range [0.0, 1.0]); a duty factor of 0.0 would indicate that the limb is never in contact with the ground while a duty factor of 1.0 would indicate that the limb never leaves the ground.

Bipeds utilize two limbs for locomotion so therefore have two relevant duty factors. The two duty factors are related by a phase term, which represents, for each stride, the relative elapsed time between the right limb beginning its stance phase and the left limb beginning its stance phase. This phase term is called the *contralateral phase*

and can apply to either hindlimbs or forelimbs. Like the duty factor, the contralateral phase is a normalized term (on the range [0.0, 1.0]). Figure 38 illustrates a bipedal duty vector with a 0.6 duty factor for each limb and a 0.5 (i.e., 180°) contralateral phase.

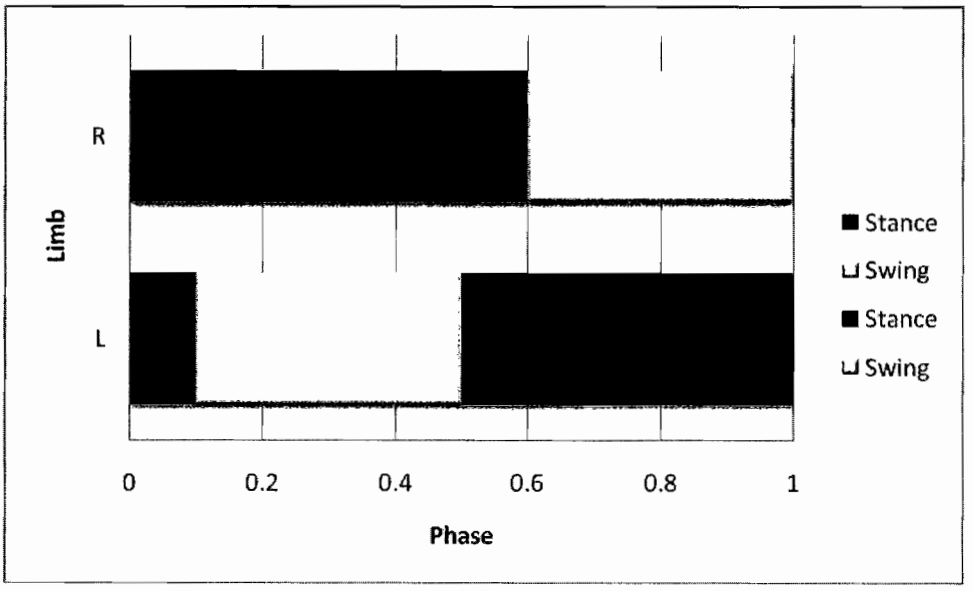


Figure 38. Example bipedal duty vector.

During walking gaits, limbs have a duty factor greater than 0.5 (i.e., limbs are in the stance phase for at least half of each stride). Each limb is in contact with the ground for more than half of the stride, so there must be periods of time during which both limbs are in contact with the ground. These periods of time are called dual support. In the absence of turning, the movements of each limb are assumed to be bilaterally symmetrical and out of phase. With a contralateral phase of 0.5, a walk cycle has two

dual support phases (i.e., one just after each limb begins its stance phase). These constraints provide a powerful mechanism for pruning the LROM spaces with respect to forward locomotion.

To take advantage of the above constraints, discrete key events must be identified during the walk cycle. These events correspond to limbs beginning their stance or swing phases: "Right Down" (RD) represents the beginning of the right limb's stance phase. "Right Up" (RU) is the beginning of the right limb's swing phase. "Right when Left Down" (RLD) is the right limb's configuration when the left limb begins its stance phase. Similarly, "Right when Left Up" (RLU) is the right limb's configuration when the left limb begins its swing phase. Equivalent events are also identified for the left limb. Figure 39 illustrates the relationship between these events, with noted similarity to Figure 5 (Bruderlin and Calvert, 1989).

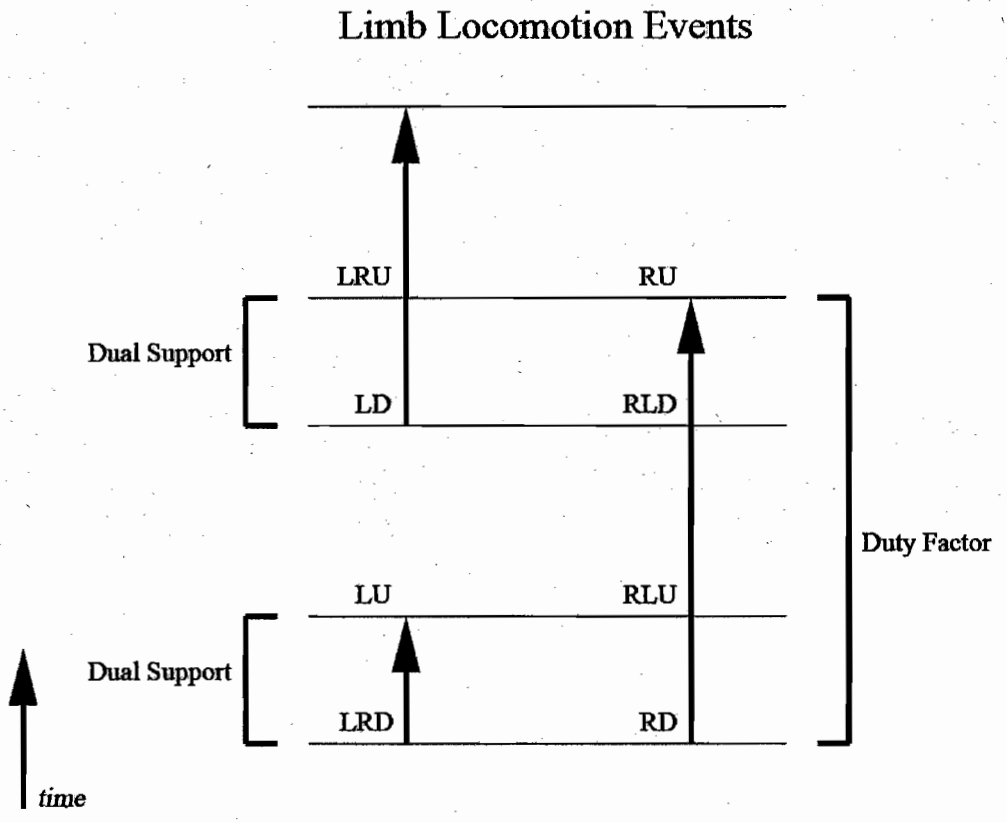


Figure 39. Events related to bipedal walking gaits.

During locomotion, but not dual support, the root element position and orientation is determined by the stance limb's LROM space. During dual support, both limbs are in the stance phase, so the root element positions and orientations predicted by the LROM spaces must be coincident. Otherwise, the body would be effectively pulled apart! The root element positions and orientations predicted by the LROM spaces must be coincident at RD and LRD. Likewise, the predictions for RLU and LU, RLD and LD, and RU and LRU must be coincident.

The left-and-right-side ROMs are bilaterally symmetrical, so the root element position and orientation caused by a right-limb FDOF configuration will be mirrored across the sagittal plane when the same FDOF configuration applied to the left limb. The limb movements are assumed to be bilaterally symmetrical during forward locomotion, so left-and-ride-side locomotion events that share an FDOF configuration (e.g., RD and LD) cause root element positions and orientations that are mirrored across the sagittal plane.

The dual support constraint forces left-and-right-side events that occur at the same time to have coincident root element positions and orientations. The bilateral symmetry constraint forces left-and-right-side events that share an FDOF configuration to have root element positions and orientation that are mirrored across the sagittal plane. Combining these constraints, LD must have a root element position and orientation that is mirrored across the sagittal plane from RD's root element position and orientation. Furthermore, RLD must have a root element position and orientation that is coincident with LD's. The same relationships exist between LRU, RLU, and RU. Figure 40 illustrates these relationships.

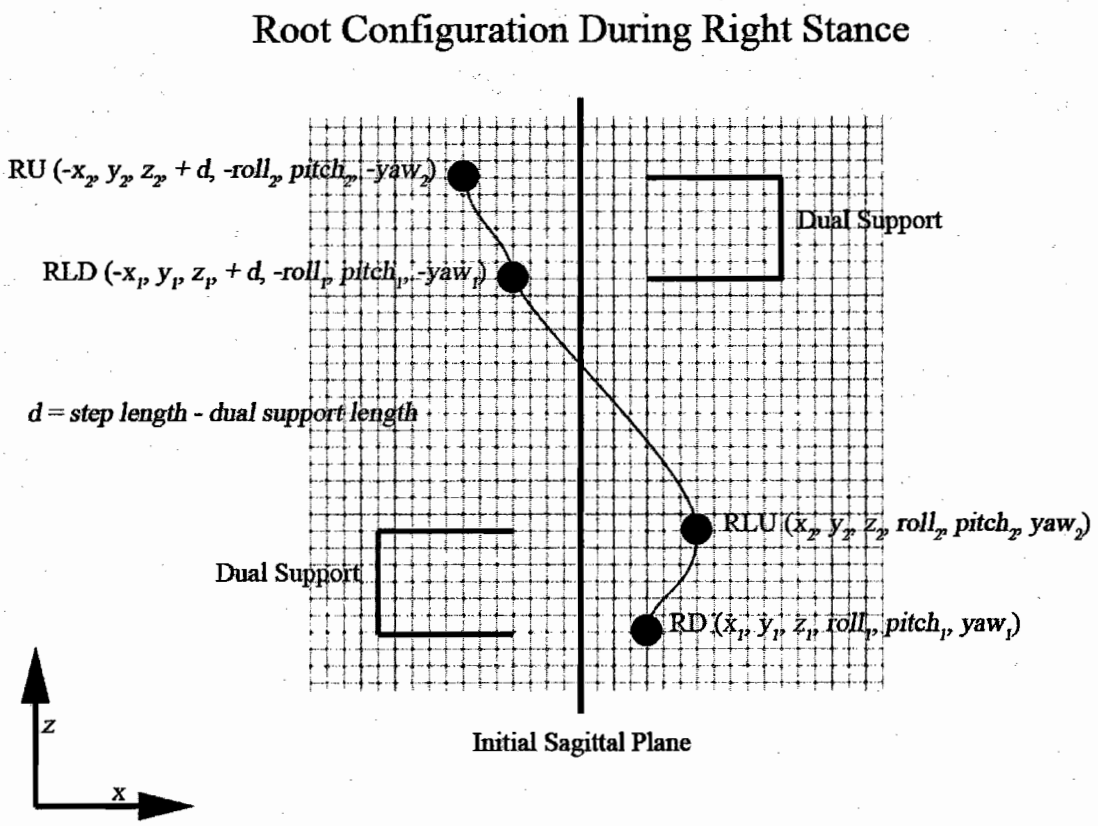


Figure 40. Relationships between discrete key events during forward locomotion.

Amazingly, the dual support and bilateral symmetry constraints force the eight original key locomotion events to be described by only two unique FDOF configurations (e.g., the configurations responsible for RD and RLU). The two discrete FDOF configurations apply to only one of the two limbs, so one LROM space is sufficient to represent a walk cycle. Specification of a walk cycle therefore consists of selecting two samples from the LROM space. The distance along the direction of travel between events is known (e.g., the dual support distance in the case of RD and RLU), further

constraining the selection process. The next section will cover the selection of LROM space samples in more detail.

Reconstruction

Bipedal gaits are reconstructed from pairs of LROM space samples, specifically samples representing RD and RLU. Due to the dual support and bilateral symmetry constraints, an LROM space sample is eligible to be selected for RD if and only if there exists an LROM space sample for RLD that is mirrored across the sagittal plane and further along the direction of travel (see Figure 40). The distance along the direction of travel between RD and RLD is the forward distance traveled while the limb is in its swing phase (equal to *step length – dual support length*). Similarly, an LROM space sample is eligible to be selected for RLU if and only if there exists a suitable LROM space sample for RU.

The set of all candidates for RD and RLU can be visualized by pruning the LROM space. All samples that do not have a suitable sibling sample (i.e., mirrored across the sagittal plane and properly forward along the direction of travel) are removed. The LROM space is populated with discrete samples, so the chance of exact matches (within floating point precision) is extremely low. For this reason, a minimum error term is used that aggregates position and orientation error between samples. Figure 41 shows a pruned *Apatosaurus* forelimb LROM space with blue representing samples that are siblings but not candidates.

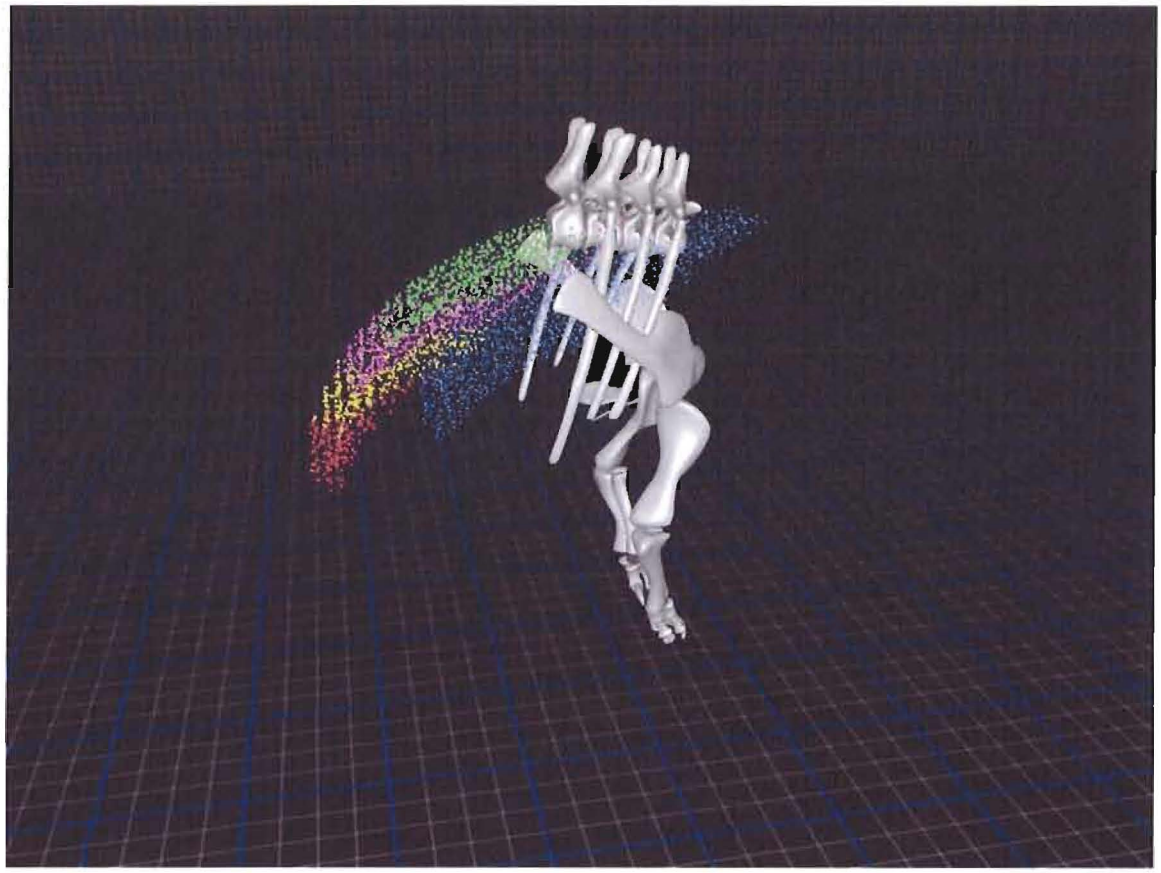


Figure 41. Pruned *Apatosaurus* forelimb LROM space.

For clarity, Figure 42 shows a comparison of the *Apatosaurus* forelimb LROM space before (left) and after (right) pruning. Recall that after pruning, the positions colored blue are not candidates eligible for selection as RD or RLU. In this example, the pruning process reduced the space from 5,221,125 samples to 406,377 candidate samples.

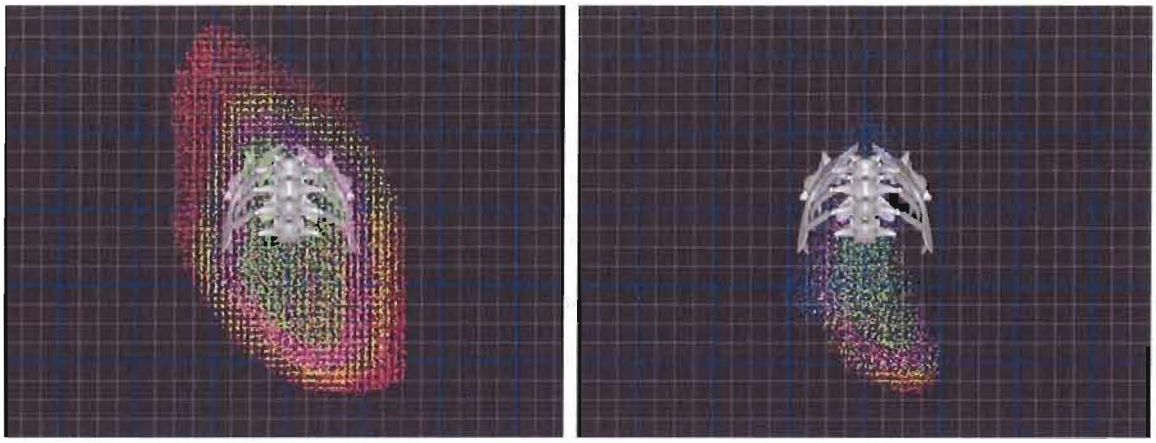


Figure 42. Apatosaurus forelimb LROM space before and after pruning.

A GA is used to evaluate candidate solutions that represent bipedal walking gaits (see Appendix G). Each candidate consists of an LROM space sample and associated FDOF configuration for RD and an LROM space sample and associated FDOF configuration for RLU. The GA selects optimal candidates using an aggregate fitness function that penalizes pitching, rolling, and yawing of the body and large changes in FDOF values between the RD and RLU configurations. The fitness function also rewards the lateral and vertical smoothness of the gait. The lateral smoothness is relative to the initial sagittal plane; the vertical smoothness is relative to an adjustable target height parameter.

The FDOF-value term discourages limbs from undergoing unnatural movements to achieve the kinematic-smoothness goals of the body. If a limb contains several redundant FDOFs (i.e., with near-coincident instantaneous axes of rotation), many solutions may result in a single root node position and orientation. In this case, RD and

RLU candidates may be selected that cause unwanted and unrealistic rotations between key configurations. For example, knee, ankle, and manus joints can counterrotate relative to each other through quite a large total angular excursion without causing much forward movement at the hip.

Changes in FDOF values represent angular excursions (unless the FDOF is translation only), so discouraging large changes in FDOF values in effect discourages large angular excursions summated across all FDOFs. FDOF values are normalized on the range [-1.0, 1.0], so this representation prevents FDOFs with larger total angular excursions from being penalized proportionally more than FDOFs with less angular excursion. Unnatural limb movements could likely also be avoided by penalizing similar criteria such as sum angular acceleration (Raibert and Hogins, 1991; Chung and Hahn, 1999).

The target height term encourages vertical smoothness and walking at an appropriate height. Imagine an animal walking with very bent limbs; without specific osteological support, excessive bending at a joint causes a torque about the joint (Hutchinson, 2005). This torque is caused by the position of the joint not being collinear with the GRF vector. Counteracting the torque is mechanically and energetically expensive and nominally avoided by animals. Rewarding candidate solutions proportionally based on the vertical height of the body encourages walking with straight limbs, which better approximates energetic constraints. Conversely, specifying lower target height values allows the evaluation of more-squat gaits. In this way, mammalian

models can walk with generally straight limbs without encouraging reptilian models to use an inappropriately-high walk.

To reduce the total number of candidate solutions, the candidate LROM space samples are organized into two data structures. RD represents the start of the right limb's stance phase, so it must be at least a step length back from the front of the space so that the body can be moved forward by the step length during stance. All candidate samples that are at least a step length back from the front of the space are stored in a linear array called the *back data.*

RLU is further along the direction of travel from RD by the dual support length, so there are a limited number of candidates that can be selected for RLU based on the selection of RD. The candidate LROM space samples are therefore also divided into coronal slices and stored in a two-dimensional array called the *slice data.* When the GA selects a sample for RD from the back data, RD's coronal slice is determined using a constant-time operation. RLU's slice is then computed using the dual support length and a sample is chosen for RLU within that slice. Figure 43 illustrates the back data and slice data and how they relate to the LROM space.

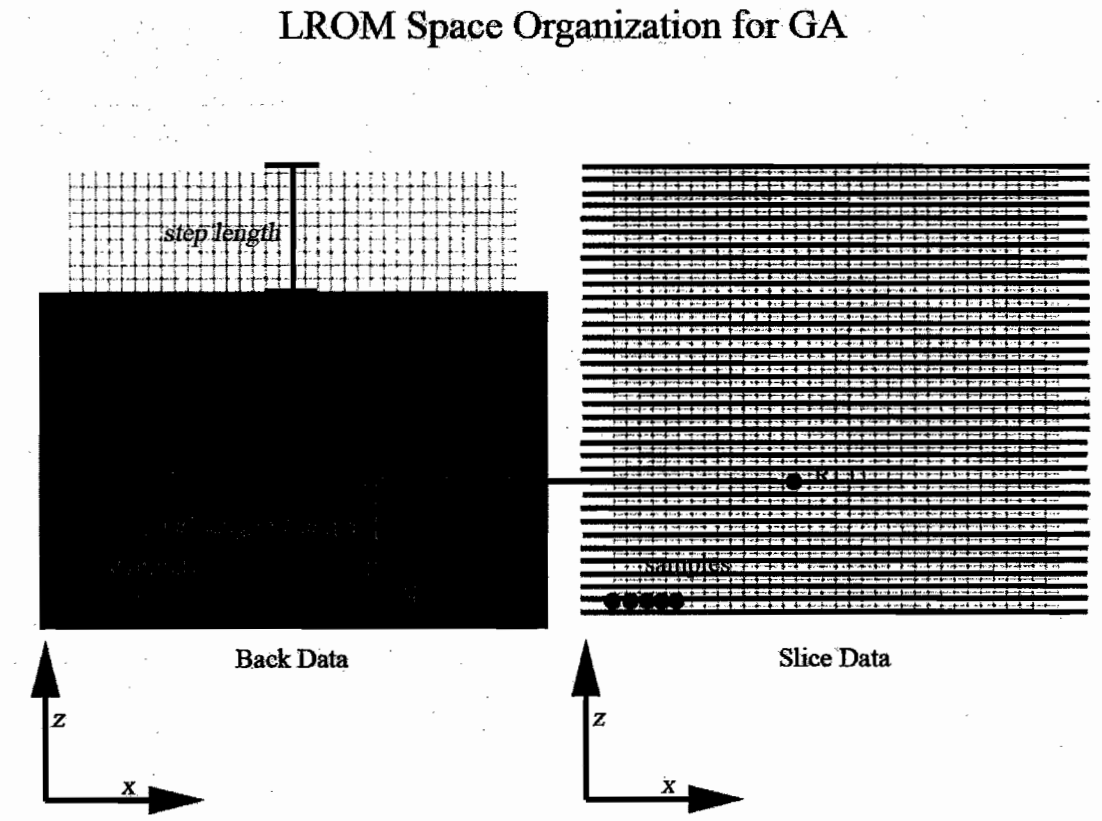


Figure 43. Back data and slice data organization.

The back data and slice data organization provides two important advantages to the GA. First, any candidate solution will at least accomplish a stance phase that moves the body forward by the step length. Candidates then need only be evaluated on how well they move the body forward (i.e., minimum body roll, pitch, and yaw, and changes in FDOF values; maximum lateral and vertical smoothness), allowing the GA to converge much more quickly than it would if the solution space included candidate solutions incapable of accomplishing a successful stance phase.

The second important advantage of the back data and slice data organization (which is really a corollary of the first advantage) is that a candidate solution can be represented and encoded as two integer values: one value for the index of RD in the back data and one value for the index within RLU's slice in the slice data (the slice is determined by RD). Candidate solutions can therefore be represented by fixed-length linear binary strings. The number of bits used to represent candidate solutions is determined dynamically by finding the next power of two bigger than the number of back data samples and adding it to the next power of two bigger than the largest (in terms of sample count) coronal slice in the slice data.

During GA iterations, each candidate in the population has a chance to be mutated and/or combined with another candidate. The probably of being mutated is based on a mutation coefficient; a coefficient of 1.0 indicates that on average every candidate will have one random bit flipped. The probability of a candidate being combined with another candidate is based on a crossover coefficient. A candidate is combined with another randomly-selected candidate using single-point crossover; an index is randomly selected before which the first candidate receives the second candidate's binary data and after which the second candidate receives the first candidate's binary data.

The solution selected by the GA is used to reconstruct all eight of the key locomotion events. An additional FDOF configuration is selected to represent the midpoint of the swing phase. The mid-swing configuration is selected using an IK method that iteratively adjusts each FDOF to find a configuration that positions the manus or pes above its initial position by a specified step height. The right-side FDOF

configurations are used to specify the associated left-side events (e.g., RD to LD). The left-side events are then offset in animation time based on the contralateral phase.

The dual support and bilateral symmetry constraints allow the LROM space to be significantly pruned, and ensure that the remaining samples do not pull the body and limbs apart during dual support. The back data and slice data organization allow the GA to search a space of candidate solutions which will at least move the body forward by the appropriate amount during the stance phase. The resulting implementation can automatically generate bipedal walk animations in less than one minute (7 FDOFs, 1000 candidate population, 10,000 GA iterations) on a consumer laptop (as of this printing). Figure 44 shows an example *Apatosaurus* forelimb walk animation created using these methods.

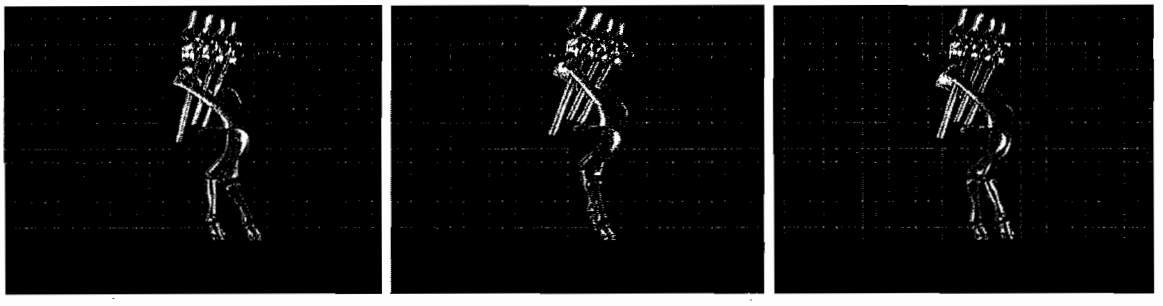


Figure 44. *Apatosaurus* forelimb walking animation.

The GA quickly finds smooth paths through the constrained LROM spaces. The LROM space exploration and organization operations, along with the GA operations, are

discrete operations that can be organized in terms of data flow such that input parameters can be modified at each of these stages without repeating earlier stages. The next section will cover the organization of this pipeline.

Pipeline

The process of automatically generating bipedal walking animations is divided into discrete operations. Between each operation, data is saved so that an operation need only be repeated if its input variables are modified. The Explore Space operation takes the model's FDOFs as input, along with angular and translational resolution parameters. The operation outputs the LROM space samples in unorganized form. The Organize Space operation takes the unorganized space data, organizes it into 3D boxes, and applies the bipedal constraints. The operation takes as input the unorganized space data, the duty factor, the constraint error tolerance, the step height, and the number of sample boxes along the primary locomotion axis.

After the LROM space has been explored, organized, and pruned, the Find Path operation plots a path through the constrained space using standard GA parameters as input. The GA parameters include: crossover coefficient, mutation coefficient, candidate population size, and number of GA iterations. The Find Path operation also generates and outputs a complete bipedal animation file, which is visualized and interpolated using the Animate Path operation. The Animate Path operation utilizes an adjustable animation time parameter to vary the animation playback speed. Figure 45 illustrates the operations of the bipedal gait pipeline.

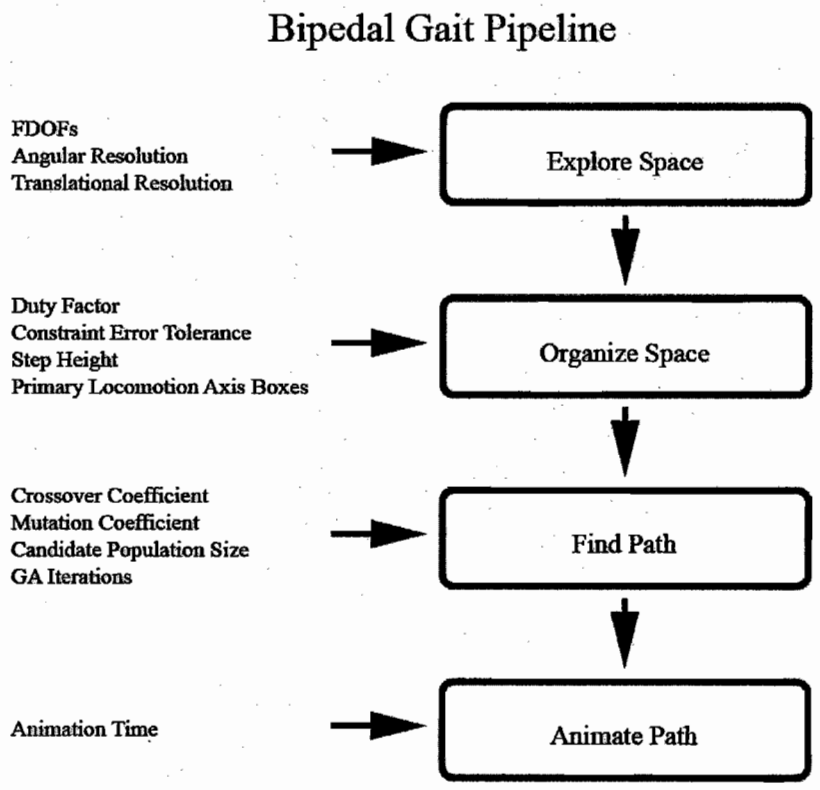


Figure 45. Bipedal gait pipeline.

The bipedal gait pipeline allows the computationally-efficient generation of bipedal walking gaits, even when changes to input parameters is necessary. Quadrupedal locomotion is significantly more complicated, requiring parallel bipedal pipelines and additional constraints for summarizing the ROM of the animal's trunk. The next section will present methods related to the generation of quadrupedal gaits.

Quadrupedal Gait Reconstruction

The following three sections describe the algorithms used to automatically generate quadrupedal walking gaits. Constrained hindlimb and forelimb LROM spaces must first be generated. Those spaces must then be further constrained to simulate the ROM of the animal's trunk. A GA then finds smooth, plausible paths through the constrained hindlimb and forelimb space. Finally, an extension of the bipedal gait pipeline architecture maximizes reuse of the additional quadrupedal data.

Constraints

A quadrupedal gait can be represented as two independent bipedal gaits (i.e., fore and hind, each with duty factors and a contralateral phase), provided that additional constraints are satisfied. An animal's fore and hind limbs are connected by a trunk consisting of some number of vertebrae. Each vertebral joint has a ROM, so the trunk itself can be represented by a higher-level ROM that summarizes the movements of the individual vertebrae. The fore and hind gaits must be coordinated in such a way that they act as if they are connected by the trunk. The convention of representing quadrupedal locomotion with two bipedal gait systems is supported by Griffin, Main, and Farley (2004), who state that dog fore and hind quarters generally act like two independent bipeds.

The trunk ROM can be exercised like any other ROM, allowing possible forelimb root element positions and orientations to be determined for each hindlimb LROM space

sample. For each hindlimb space sample, there are some number forelimb space samples that are reachable by the trunk. The forelimb and hindlimb LROM spaces can therefore be further pruned based on the trunk constraint; all hindlimb samples that cannot reach a single forelimb sample are removed, along with all forelimb samples that cannot be reached by any hindlimb samples. Figure 46 illustrates how the trunk constraint is applied to the fore and hind LROM spaces.

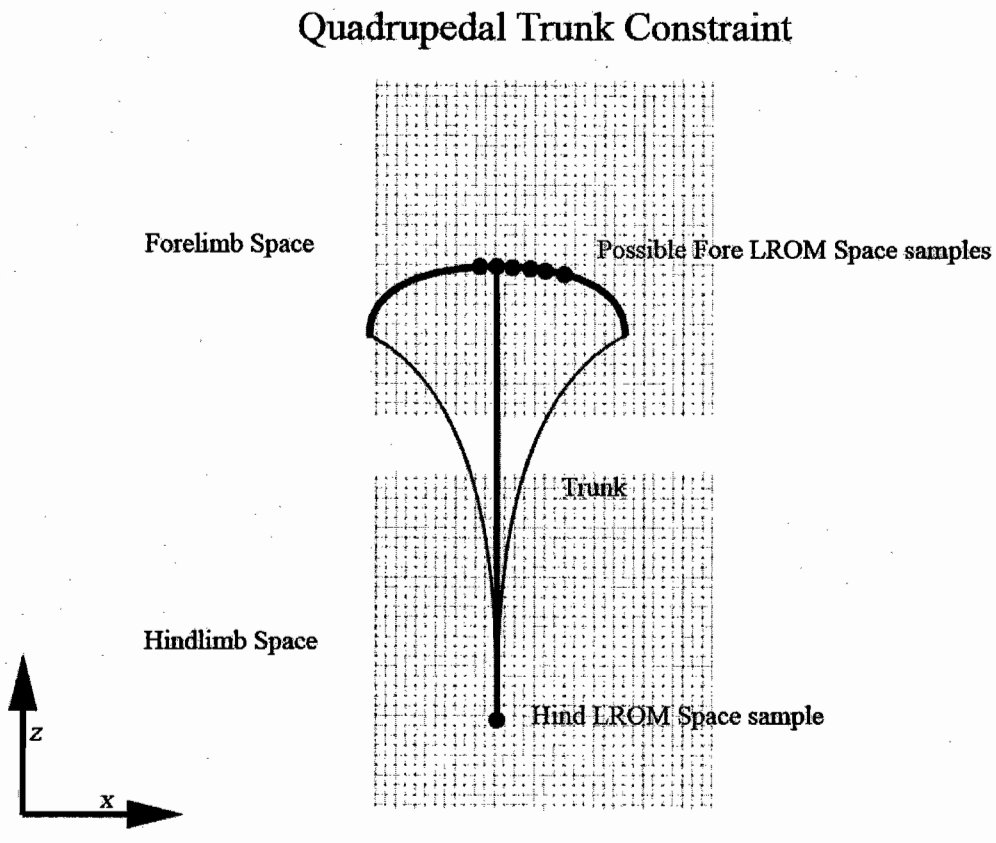


Figure 46. Quadrupedal Trunk Constraint from dorsal view.

The forelimb and hindlimb gait components are related by a phase term called the *ipsilateral phase*. Like contralateral phase, the ipsilateral phase is a normalized term (on the range [0.0, 1.0]). The ipsilateral phase describes the relative elapsed stride time between the right hindlimb beginning its stance phase and the right forelimb beginning its stance phase. The two contralateral phases (fore and hind), the ipsilateral phase, and each limb's duty factor fully describe the quadrupedal duty vector. Figure 47 shows an example duty vector based on a 0.5 contralateral phase for both fore and hind limbs, a 0.55 ipsilateral phase, and 0.6 duty factors for all limbs.

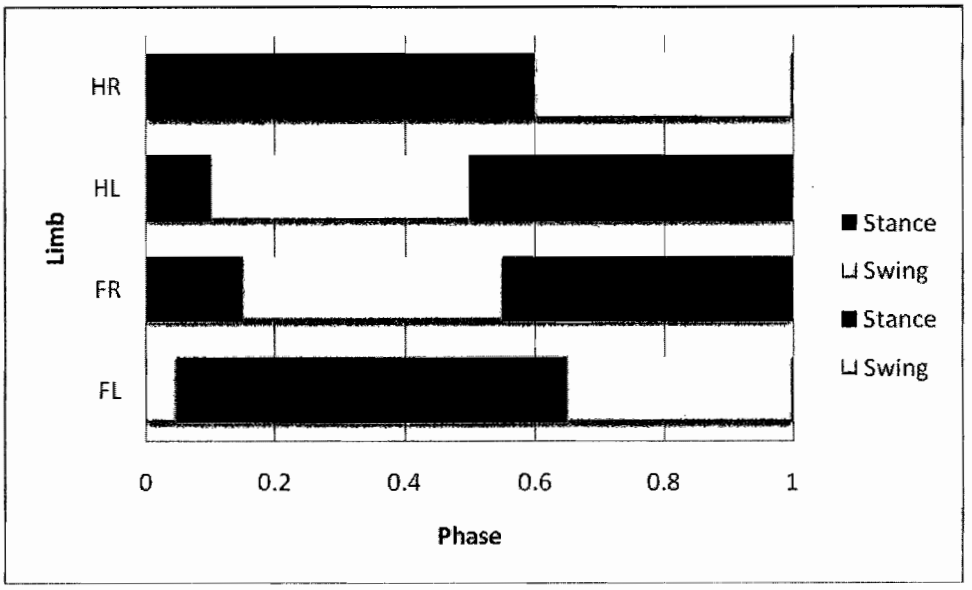


Figure 47. Example quadrupedal duty vector.

The ipsilateral phase does not affect the trunk-constraint-based pruning process because discrete samples in the hindlimb space are compared to discrete samples in the forelimb space. When constructing a walking gait based on key locomotion events, however, the ipsilateral phase determines the relative delay between associated events with respect to the hindlimbs and forelimbs (e.g., hindlimb RD and forelimb RD). The algorithm for constructing quadrupedal gaits based on key locomotion events will be discussed in the next section.

Reconstruction

Forelimb and hindlimb animations can be generated in such a way that they obey the trunk constraint while achieving the same GA goals used for bipedal gaits (i.e., minimum body roll, pitch, and yaw, and changes in FDOF values; maximum lateral and vertical smoothness). The algorithm is straightforward with a constant ipsilateral phase of 0.0; associated fore and hind events occur at the same time, so the list of possible samples for the forelimb RD can be determined based only on the hindlimb RD sample (and likewise for the forelimb RLU with respect to the hindlimb RLU). Supporting an arbitrary ipsilateral phase complicates the algorithm.

Given an ipsilateral phase of p, the forelimb RD occurs at time p (the hindlimb RD always occurs at time 0.0). The complete hindlimb path can be determined by an RD and an RLU sample, so the hindlimb root element position and orientation at time p can be computed from the path. Therefore, the possible forelimb RD samples can be determined for a given ipsilateral phase, RD sample, and RLU sample (and likewise for

the forelimb RLU sample). The possible forelimb RD and RLU positions must be computed for each RD-RLU pair so that the GA will only consider forelimb samples that satisfy the trunk constraint. The process of finding satisfactory forelimb samples (within an error tolerance) is performed prior to the GA iterations to ensure that each RD-RLU pair is evaluated only once.

There is one important corollary to the trunk-constraint satisfaction algorithm. The ipsilateral phase determines, in part, the z component (i.e., translation along the locomotion axis) of the hindlimb root element position at time p (along with the z component of the hindlimb RD sample position). The additional forward translation forces the selection of forelimb samples that are unnecessarily pushed towards the front of the fore LROM space. To compensate for this effect, the z component of the hindlimb root position at time p is set to the z component of the hindlimb RD sample position. Figure 48 illustrates the selection of possible forelimb RD samples based on the ipsilateral phase and an RD-RLU pair.

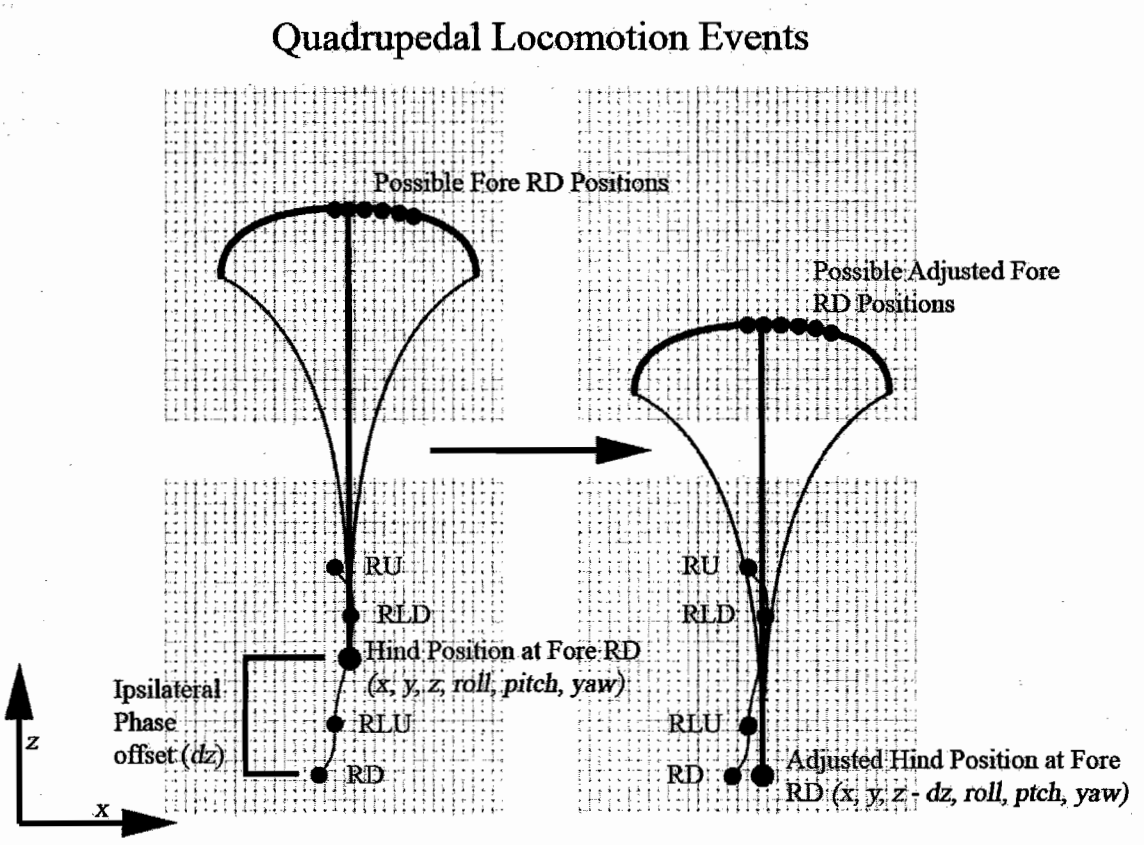


Figure 48. Forelimb RD selection based on a hindlimb RD-RLU pair.

The computational speed of the possible-forelimb-sample selection algorithm benefits from the back data and slice data organization; the algorithm needs only to evaluate hindlimb RD-RLU pairs that are eligible for selection as pairs for a hindlimb gait. The back data and slice data organization therefore dramatically decreases the computational complexity of the selection algorithm (i.e., from $O(n^2)$ to $O(n*m)$, where m is roughly the square root of n) and allows the GA candidate evaluation function to remain a constant-time operation. The resulting implementation is able to automatically

generate quadrupedal walking gaits in under three minutes (five hindlimb FDOFs, seven forelimb FDOFs, 1000 candidate population size, 10,000 GA iterations) on a consumer laptop (as of this printing). Figure 49 shows an *Apatosaurus* quadrupedal walking gait generated using these methods.

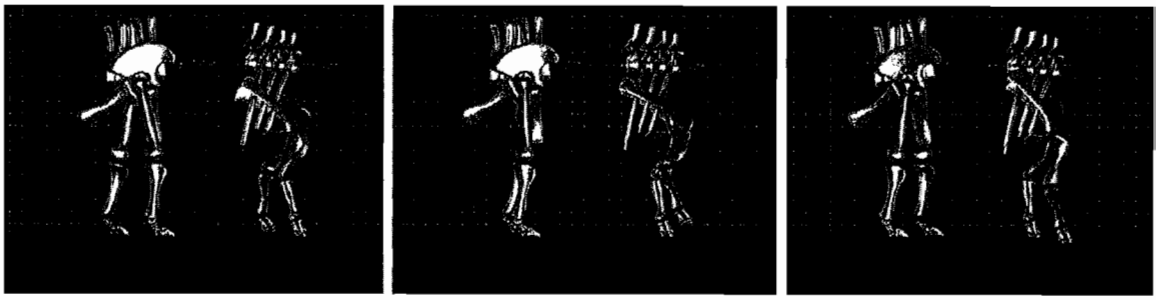


Figure 49. Apatosaurus quadrupedal walking animation.

The quadrupedal gait generation process involves a number of discrete operations, much like the bipedal gait generation process. Like the bipedal pipeline, the quadrupedal pipeline should also maximize reuse of computed data so that input variables can be changed for a pipeline stage without necessitating the reevaluation of earlier stages. The next section will cover the quadrupedal gait pipeline.

Pipeline

The quadrupedal gait pipeline is similar to the bipedal gait pipeline, and utilizes the first two bipedal gait pipeline operations. The Quadrupedal Pipeline begins with three parallel sets of operations. Bipedal Explore Space and Organize Space operations are used to prepare the forelimb and hindlimb LROM spaces. In addition, an Explore Trunk Space operation is used to exercise the trunk ROMs and prepare the higher-level trunk ROM for the possible-forelimb-sample selection process. The Explore Trunk Space operation takes the trunk FDOFs as its only input.

The Find Quad Path operation takes the forelimb and hindlimb LROM spaces and the trunk ROM as inputs. In addition, the Find Quad Path takes as input the standard GA parameters (i.e., crossover coefficient, mutation coefficient, candidate population size, number of GA iterations), the ipsilateral phase, and a constraint error tolerance for selecting possible forelimb samples. The Find Quad Path operation plots paths through the forelimb and hindlimb spaces while satisfying the trunk constraint. The Find Quad Path outputs two complete gait animations: one for the forelimbs and one for the hindlimbs.

Finally, the Animate Path operation interpolates and visualizes the two input animations. The arbitrary ipsilateral phase causes the hindlimb and forelimb animations to loop at different times (i.e., with an ipsilateral phase of p, the forelimb animation loops at time p while the hindlimb animation loops at time 1.0). The difference in looping times causes the forelimbs to return to their original position before the hindlimbs, so the

z component (i.e., translation along the locomotion axis) of the forelimb root element's position is adjusted to compensate.

The fore and hind LROM spaces are based on discrete samples, so it is not possible to simultaneously animate the fore and hind limbs while maintaining contact with the ground and keeping the trunk perfectly intact. For this reason, the animal's trunk is representing by a self-adjusting, semi-translucent ribbon (see Figure 49). Figure 50 illustrates the quadrupedal gait pipeline.

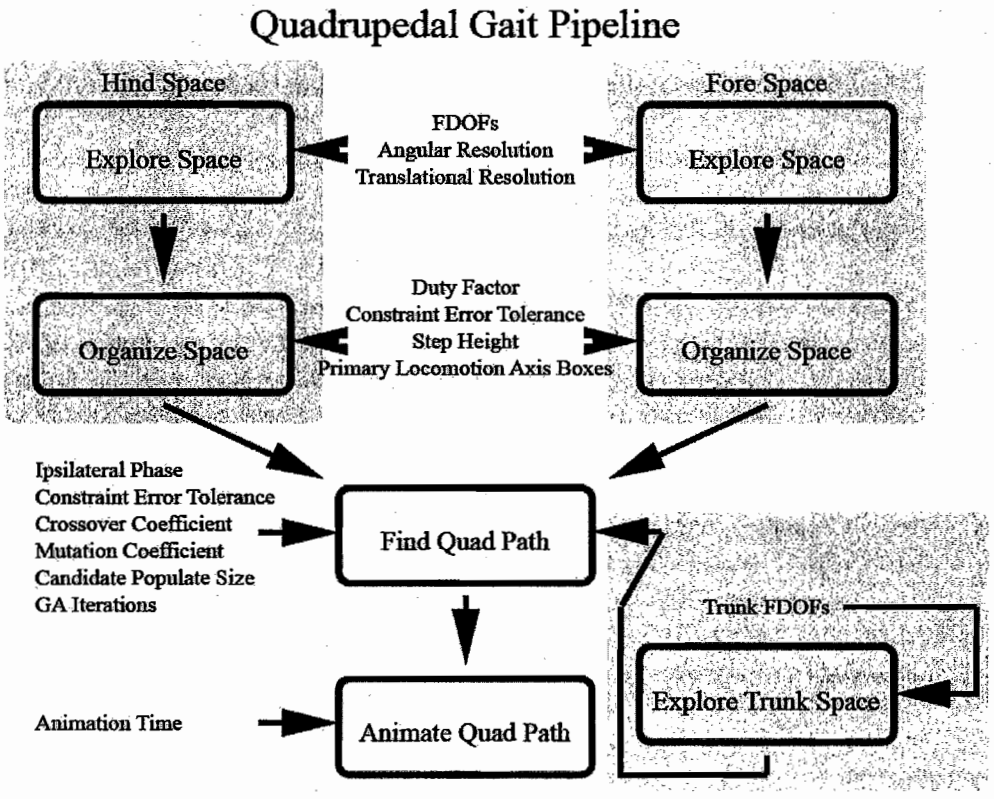


Figure 50. Quadrupedal gait pipeline.

The bipedal gait pipeline allows the computationally-efficient generation of quadrupedal walking gaits, even when changes to input parameters are necessary. Additional discrete operations refine walking gaits, visualize trackways, and allow the scaling of model elements. The next few sections will present these operations.

Gait Refinement

LROMs are explored at a resolution of between 4° and 6°. Gait animations resulting from the methods described in this chapter are not particularly sensitive to the LROM sampling resolution (see the sensitivity analysis in the next chapter), but the effective sampling resolution can be increased by refining gait animations. The sampling resolution is multiplicative across all FDOFs (i.e., doubling the sampling resolution of an LROM with 8 FDOFs causes the number of LROM samples to increase by a factor of 2^8 = 256). However, the sampling resolution is additive if each FDOF is iteratively resampled without modifying the other FDOFs.

Each FDOF is iteratively resampled at a higher resolution (i.e., typically 1°). Resampling the FDOF changes both the LROM and LROM space. A new gait animation is then created using the same GA fitness function that was used to create the original gait animation, but using the modified LROM space that represents each original animation keyframe with one highly-sampled FDOF. The refined gait animations vary little from their original counterparts in terms in terms of functional joint movements. They do, however, exhibit significantly-higher fitness values and are therefore generally smoother

in terms of body roll, pitch, yaw, and lateral/vertical error. Figure 51 shows a dog hindlimb gait before and after refinement.

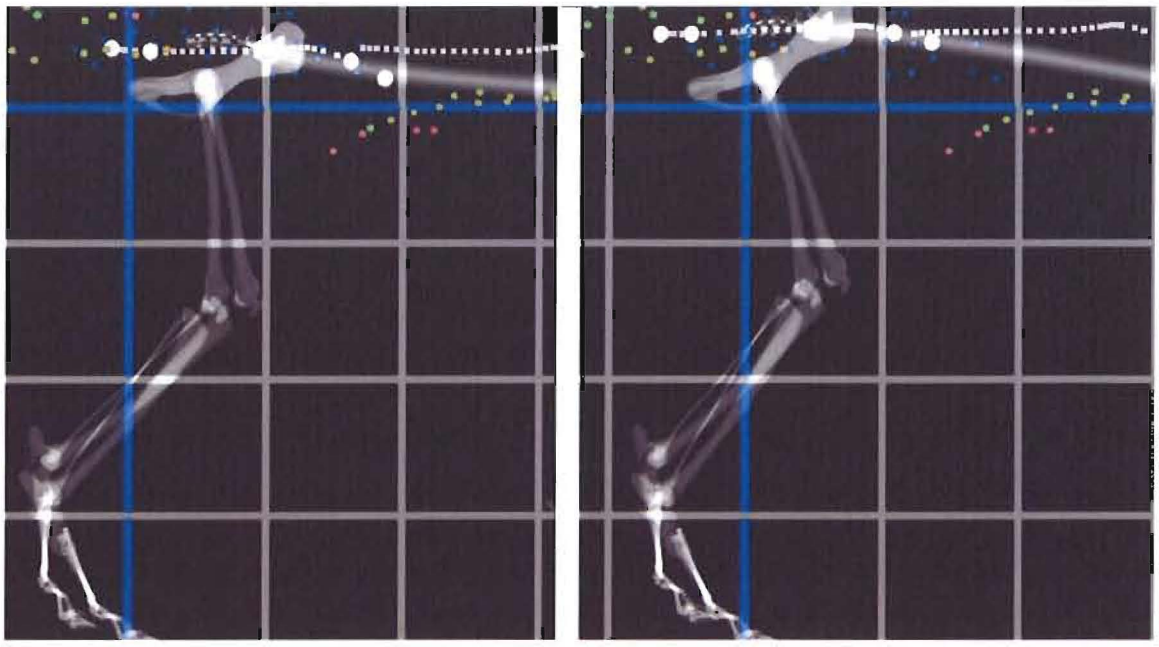


Figure 51. Dog hindlimb gait animation before and after refining.

Gaits can be refined by iteratively resampling FDOFs the increase the resolution of the LROM spaces in certain locations found relevant to locomotion. It is often useful to visualize the positional relationships between generated trackway print locations, which are based on locomotion parameters such as the step length and step width. The next section will cover the visualization of generated trackways.

Trackway Visualization

Trackways are generated by calculating the position of the manus/pes on the ground plane when the associated limb begins its stance phase. In this way, trackways are generated as a result of gaits; gaits are not generated based on trackway data as with Torkos and van de Panne (1998) and Henderson (2006). Manus prints are visualized as flattened spheres; pes prints are visualized as flattened cubes. Figure 52 shows an example generated *Apatosaurus* trackway.

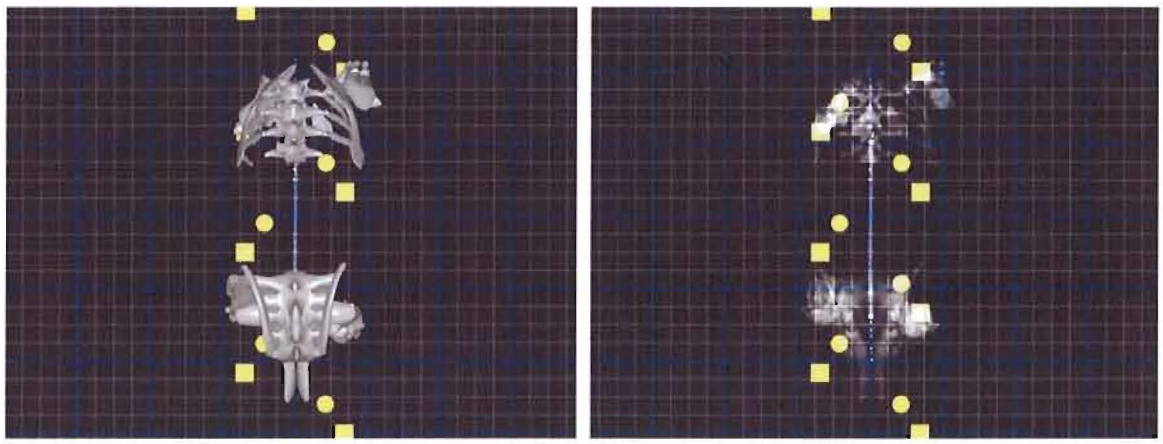


Figure 52. Example generated *Apatosaurus* trackway.

Parameters such as step length and step width modify the constrained LROM space and therefore the generated walking gaits and trackways. Scaling elements of a model (i.e., the femur or crus) will also modify the LROM space, changing generated

walking gaits and trackways. The next section will cover the scaling of specific model elements.

Scaling Model Elements

Model elements can be arbitrarily scaled (i.e., either isotropically or anisotropically) by scaling the element's distal joint ROM. A ROM is scaled by multiplying its base translational components by the scale factor(s). The translational components of the ROM's constituent FDOFs are also multiplied by the scale factor(s). The ROM's rotational components are not affected by the scaling process. To visualize the change in element scale, the element's shape scale is multiplied by the scale factor(s).

Models

Five skeletal models are currently available for analysis: a generic dog, a generic reptile, and three dinosaurs. The available dinosaurs are an *Apatosaurus*, a *Triceratops*, and a *Tyrannosaurus*. Each model consists of joint ROMs and bone shapes for the animal's hindlimbs, forelimbs, and trunk. In this section, the joints, LROM spaces, and trunk ROMs (for quadrupeds only) will be presented for each model.

Dog

The generic dog model proportions are based primarily on the standard German Shepherd (Shaw, 2007a, 2007b). Table 30 (see Appendix A) lists the six geometric DOFs for each of the dog model's FDOFs. The dog hindlimb utilizes six FDOFs at four joints: hip flexion/extension, abduction/adduction, and medial/lateral rotation; knee flexion/extension; ankle flexion/extension; pes flexion/extension. Figure 53 shows the dog hindlimb joints.

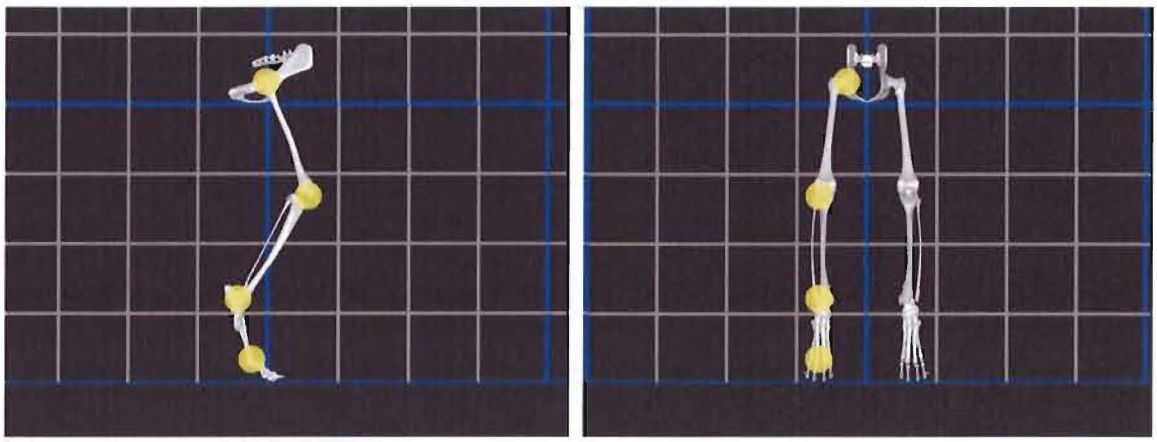


Figure 53. Dog hindlimb joints.

The dog forelimb has seven FDOFs at five joints: scapulothorax rotation; shoulder flexion/extension, abduction/adduction, and medial/lateral rotation; elbow

flexion/extension, wrist flexion/extension; manus flexion/extension. Figure 54 shows the dog forelimb joints.

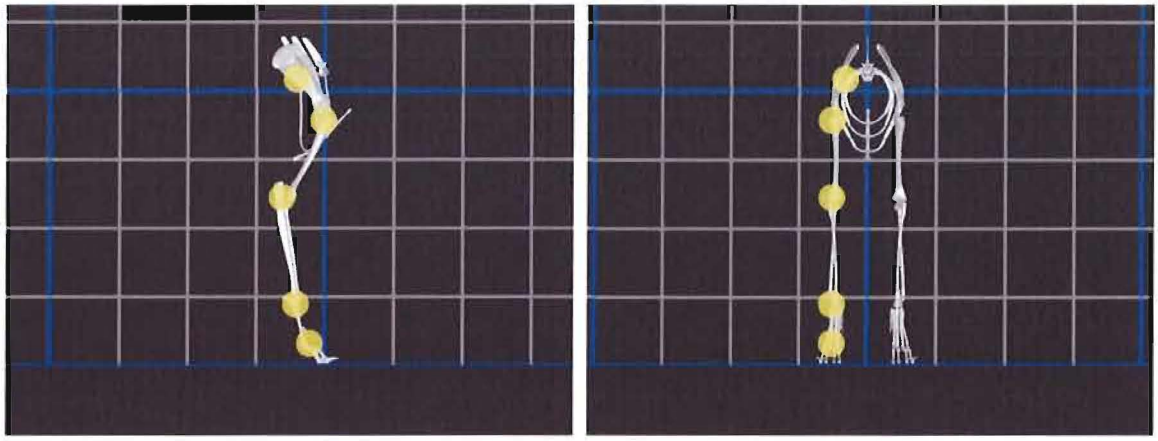


Figure 54. Dog forelimb joints.

The dog hindlimb LROM was sampled at 5° resolution creating 308,000 samples (4.41 MB, 24 bytes per sample). Figure 55 shows the dog hindlimb LROM space. The hindlimb LROM space is highly parasagittal (i.e., all LROM space root locations lie within, or close to, the animal's sagittal plane). The parasagittal LROM space is a result of the pes, ankle, knee, and hip flexion/extension FDOFs having instantaneous axes of rotation that are nearly coincident. Exercising any of these FDOFs causes the root to move and rotate along an arc (i.e., the pes FDOF having the largest radius because of the distance between the pes joint and the root, similarly the hip FDOF having the smallest radius). These flexion/extension FDOFs are therefore considered redundant.

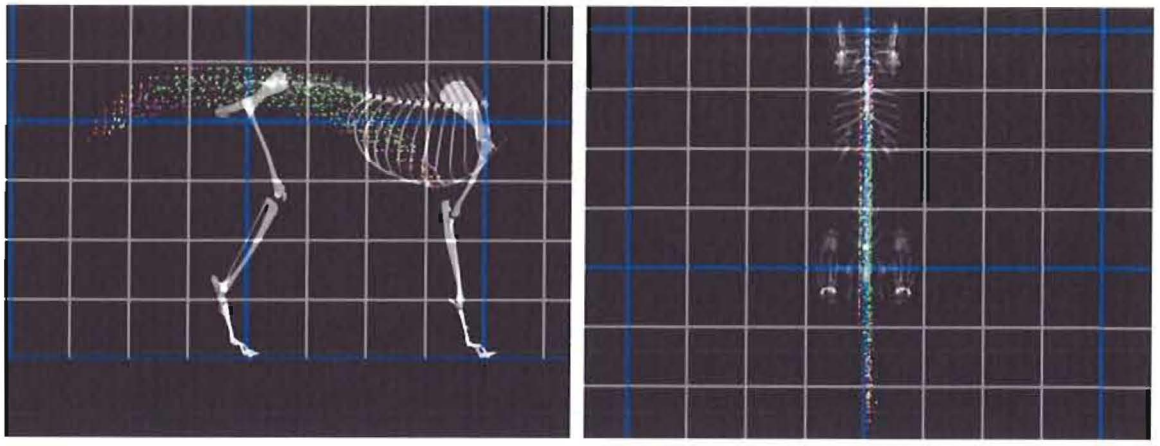


Figure 55. Dog hindlimb LROM space.

The constrained dog hindlimb LROM space is based on a step length of 0.48 times the hip height. The space contains 10,795 samples. Figure 56 shows the constrained dog hindlimb LROM space.

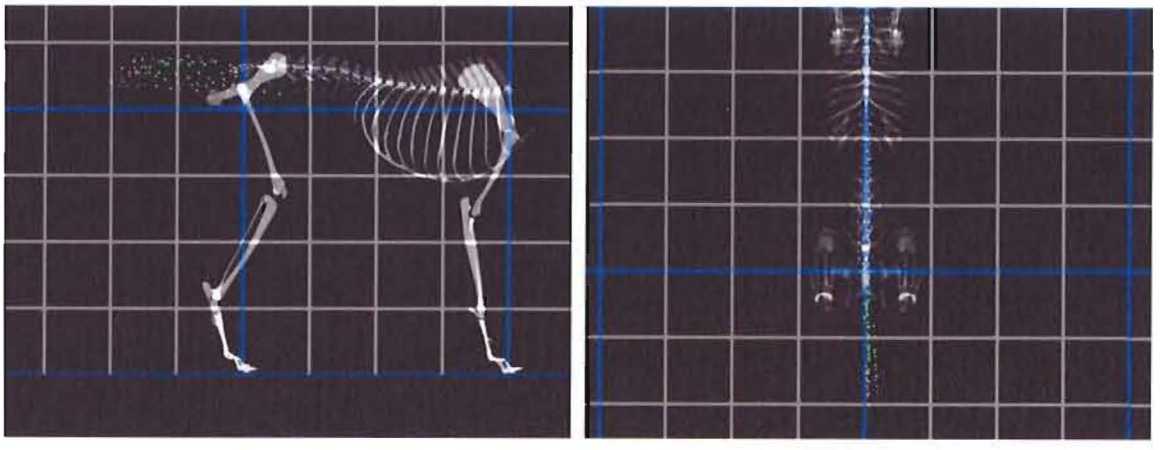


Figure 56. Constrained dog hindlimb LROM space.

The dog forelimb LROM was sampled at 5° resolution creating 3,326,400 samples (76.1 MB, 24 bytes per sample). Figure 57 shows the dog forelimb LROM space. Like the hindlimb LROM space, the forelimb LROM space is highly parasagittal.

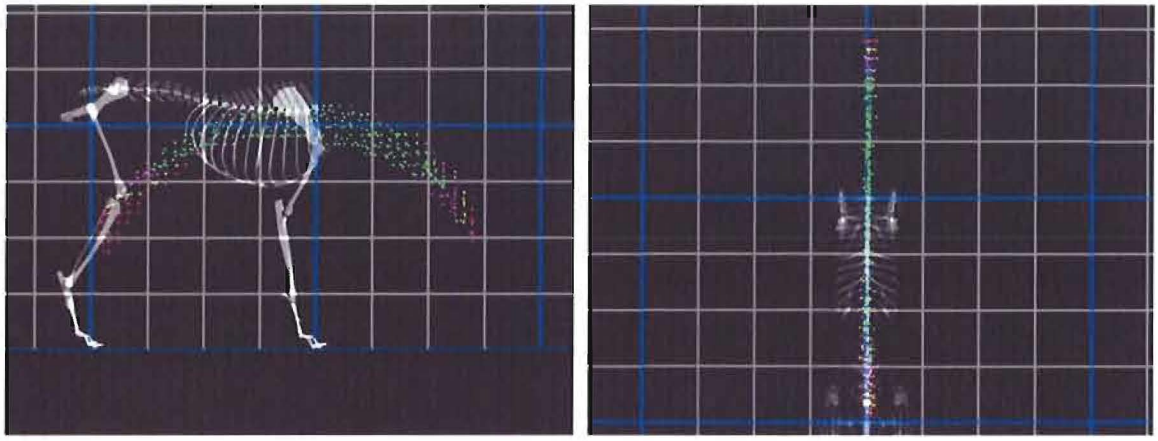


Figure 57. Dog forelimb LROM space.

The constrained dog forelimb LROM space is based on a step length 0.48 times the hip height. The space contains 4,122 samples. Figure 58 shows the constrained dog forelimb LROM space.

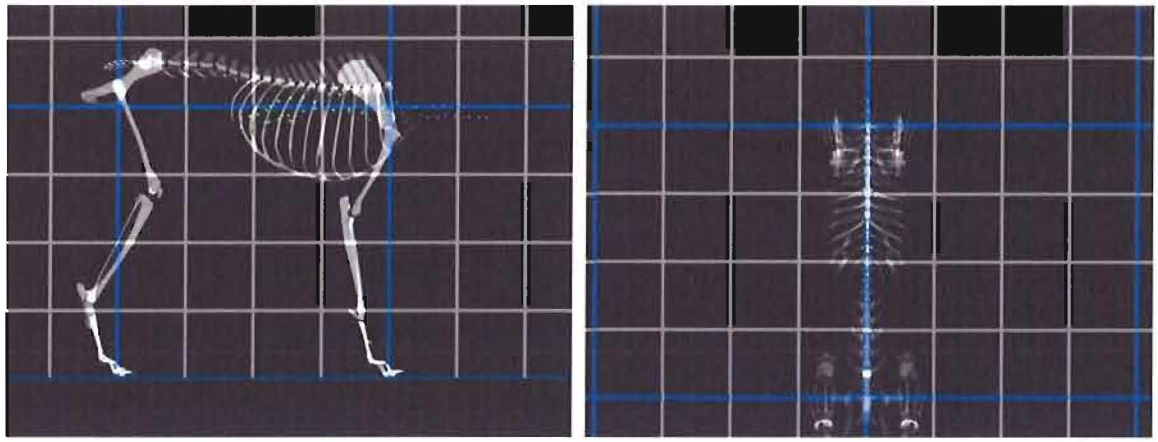


Figure 58. Constrained dog forelimb LROM space.

The dog trunk ROM consists of 19 vertebral joints. The trunk ROM represents 21 mediolateral samples and 21 dorsoventral for a total of 441 samples. Figure 59 shows the dog trunk ROM space.

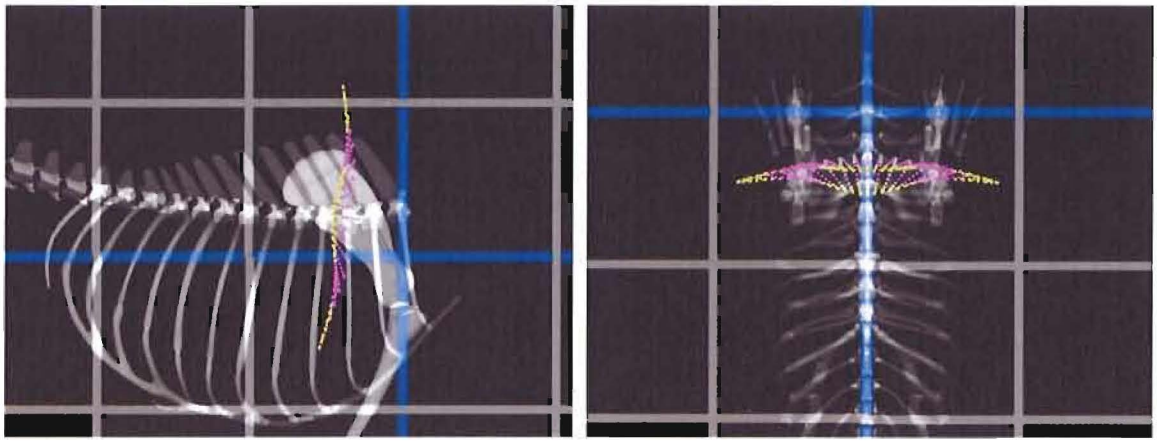


Figure 59. Dog trunk ROM space.

Reptile

The generic reptile model is based primarily on the Komodo dragon (Fotosearch, 2005). Table 31 (see Appendix B) lists the six geometric DOFs for each of the reptile model's FDOFs. The reptile hindlimb has eight FDOFs at five joints: hip flexion/extension, abduction/adduction, and inner/outer rotation; knee flexion/extension; crus pronation/supination; ankle flexion/extension; pes flexion/extension and medial/lateral rotation. Figure 60 shows the reptile hindlimb joints.

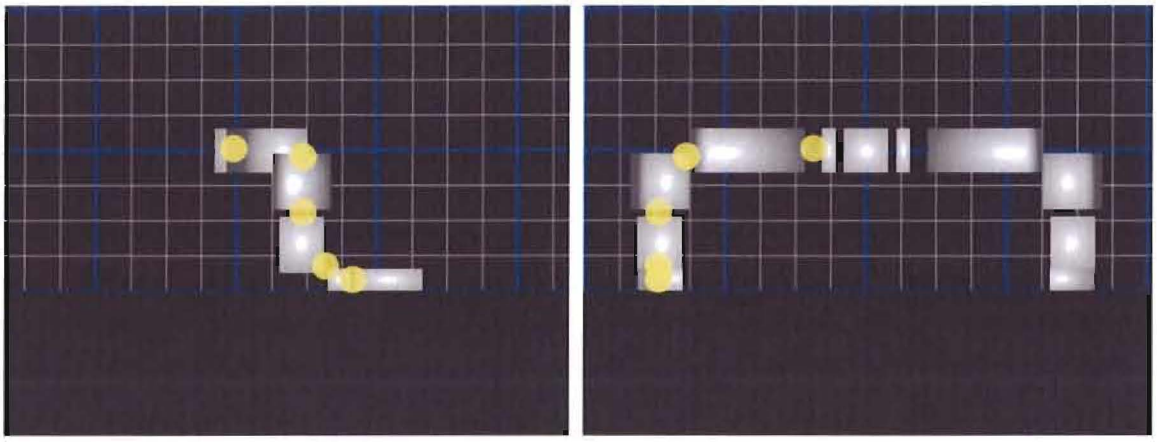


Figure 60. Reptile hindlimb joints.

The reptile forelimb uses eight FDOFs at five joints: shoulder flexion/extension, abduction/adduction, and inner/outer rotation; elbow flexion/extension, antibrachium

pronation/supination; wrist flexion/extension; manus flexion/extension and medial/lateral rotation. Figure 61 shows the reptile forelimb joints.

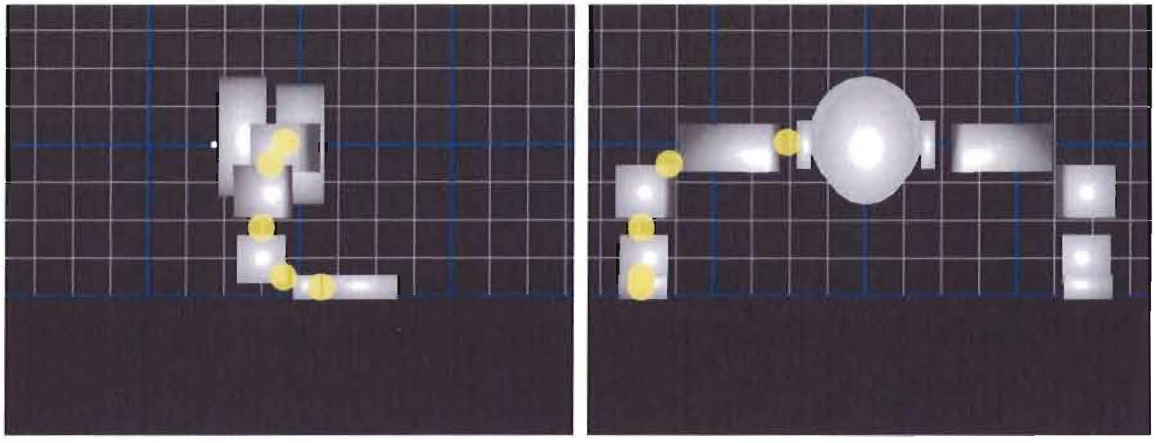


Figure 61. Reptile forelimb joints.

The reptile hindlimb LROM was sampled at 5.5° resolution creating 7,076,160 samples (161 MB, 24 bytes per sample). Figure 62 shows the dog hindlimb LROM space. The space is quite robust in part because of the knee axis, which is almost parallel with the sagittal plane due to the reptile's sprawling stance (i.e., contrary to mammals, which have knee flexion/extension axes that are nearly perpendicular to the sagittal plane). The reptile also utilizes medial/lateral rotation at the foot and pronation/supination at the crus to move the root along arcs that are not parasagittal.

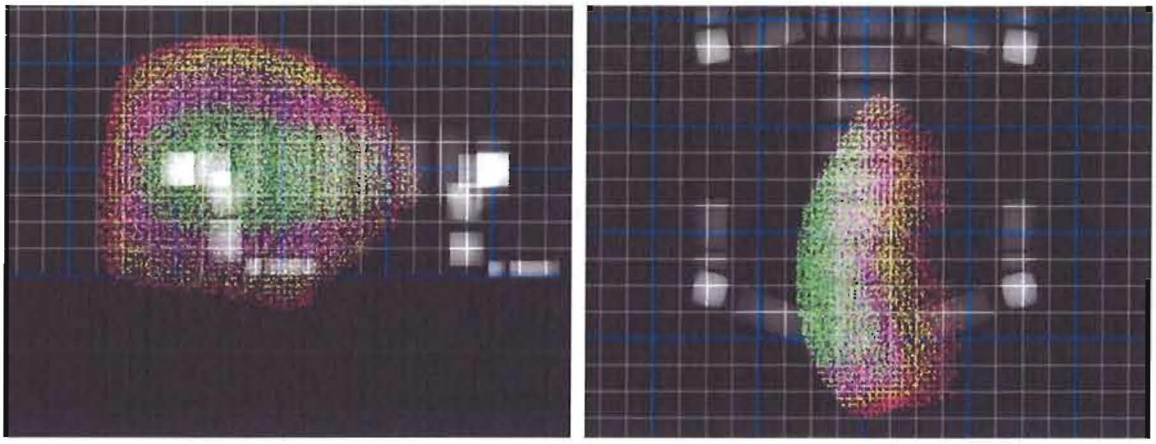


Figure 62. Reptile hindlimb LROM space.

The constrained reptile hindlimb LROM space is based on a step length 2.2 times the hip height. The space contains 136,108 samples. Figure 63 shows the constrained reptile hindlimb LROM space.

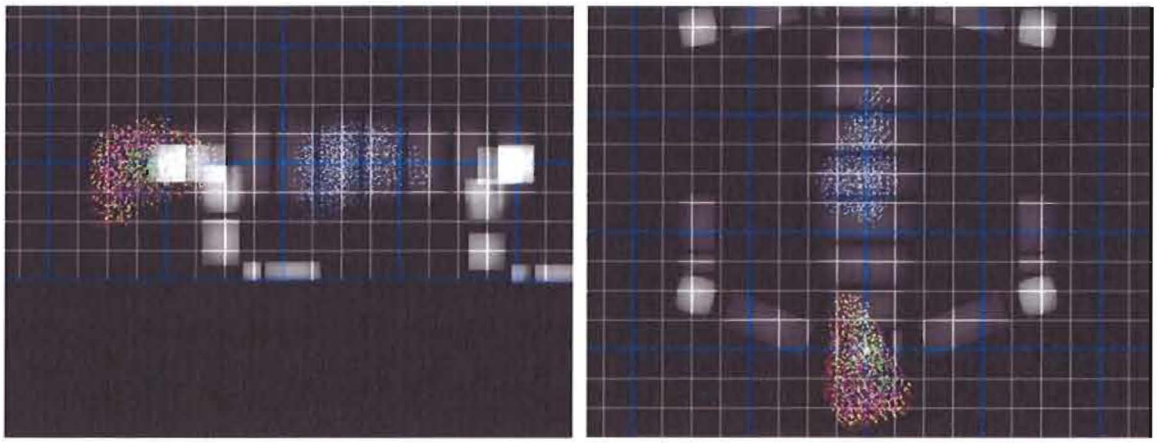


Figure 63. Constrained reptile hindlimb LROM space.

The reptile forelimb LROM was sampled at 5.5° resolution creating 2,857,680 samples (65.4 MB, 24 bytes per sample). Figure 64 shows the reptile forelimb LROM space. Like the hindlimb LROM space, the forelimb space is quite robust. Similar to the hindlimb, the robust nature of the space is largely due to the medial/lateral rotation at the manus, pronation/supination at the antibrachium, and flexion/extension at the knee that move the root along non-parasagittal arcs.

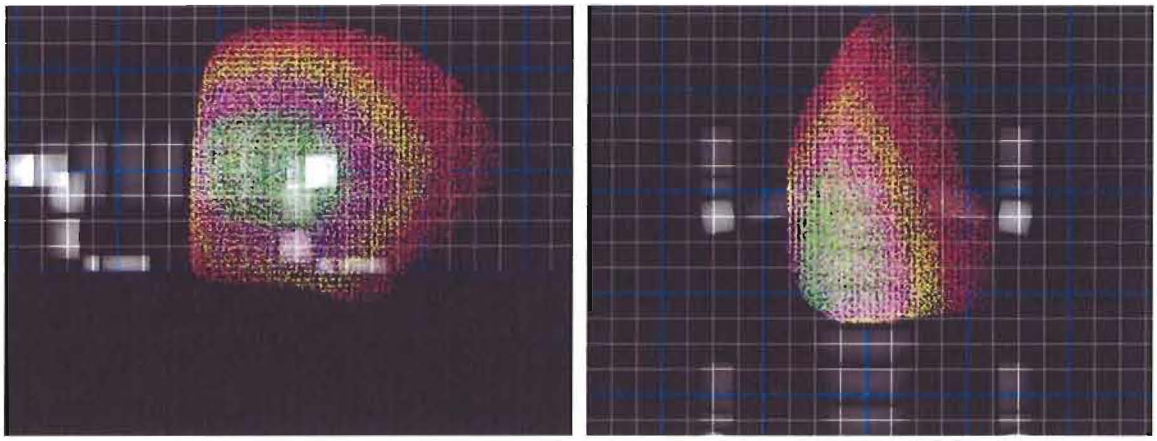


Figure 64. Reptile forelimb LROM space.

The constrained reptile forelimb LROM space is based on a step length 2.2 times the hip height. The space contains 19,431 samples. Figure 65 shows the constrained reptile forelimb LROM space.

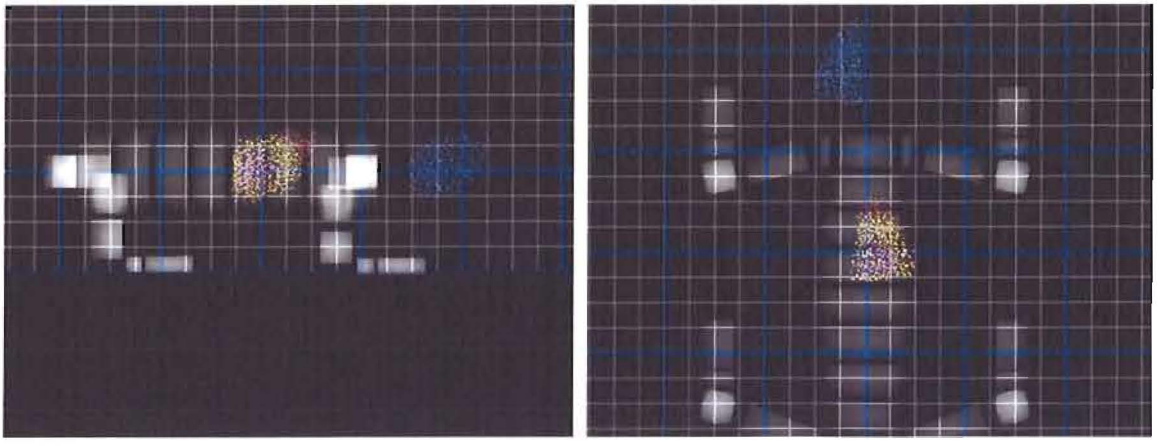


Figure 65. Constrained reptile forelimb LROM space.

The reptile trunk ROM consists of six vertebral joints. The trunk ROM represents 41 mediolateral samples and 41 dorsoventral for a total of 1681 samples. Figure 66 shows the reptile trunk ROM space.

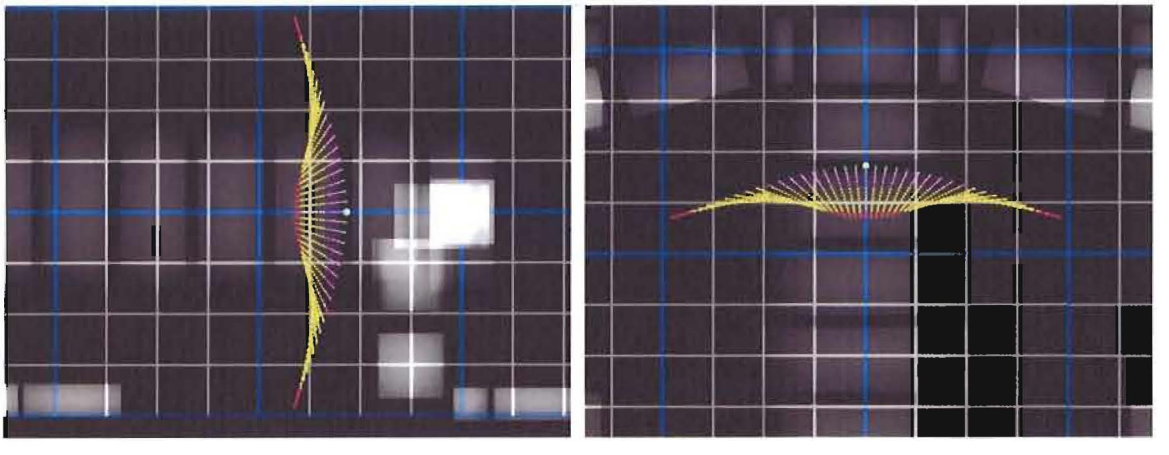


Figure 66. Reptile trunk ROM space.

Apatosaurus

Apatosaurus was a sauropod dinosaur that lived approximately 140 million years ago during the Jurassic period. *Apatosaurus* was up to 23 meters long. The specific bone dimensions and morphologies are based on the Carnegie Mellon specimen CM 3018 (Stevens, 2007a). Table 32 (see Appendix C) lists the six geometric DOFs for each of the *Apatosaurus* model's FDOFs.

The *Apatosaurus* hindlimb utilizes five FDOFs at three joints: hip flexion/extension, abduction/adduction, and medial/lateral rotation; knee flexion/extension; ankle flexion/extension. Figure 67 shows the *Apatosaurus* hindlimb joints.

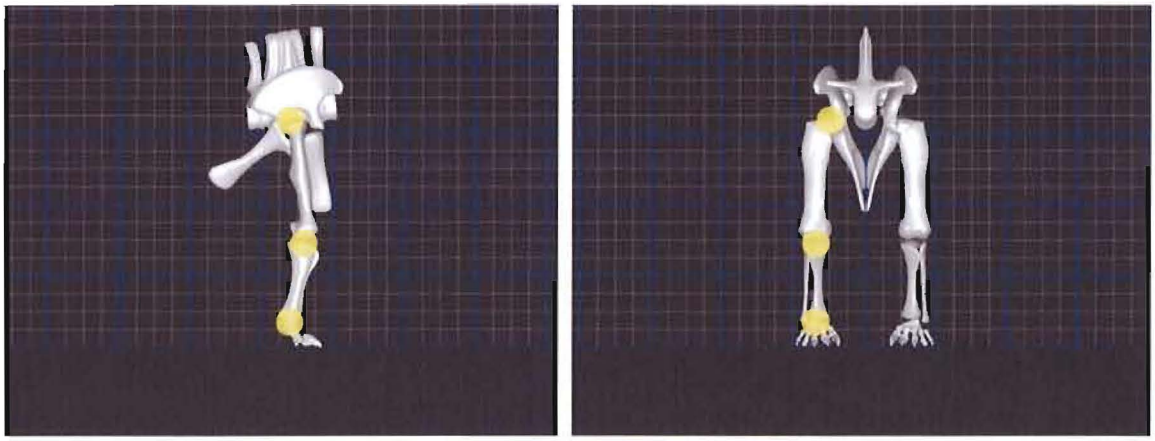


Figure 67. *Apatosaurus* hindlimb joints.

The *Apatosaurus* forelimb utilizes seven FDOFs at four joints: scapulothorax rotation and elevation; shoulder flexion/extension, abduction/adduction, and medial/lateral rotation; elbow flexion/extension, wrist flexion/extension. Figure 68 shows the *Apatosaurus* forelimb joints.

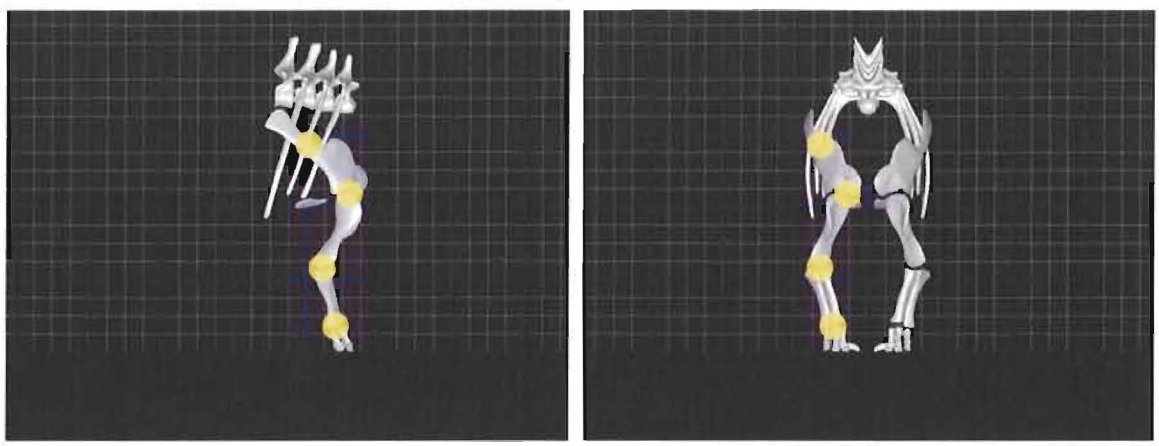


Figure 68. Apatosaurus forelimb joints.

The *Apatosaurus* hindlimb LROM was sampled at 4° resolution creating 87,975 samples (2.01 MB, 24 bytes per sample). Figure 69 shows the *Apatosaurus* hindlimb LROM space. The hindlimb LROM space is highly parasagittal due to the nearly-coincident ankle, knee, and hip flexion/extension FDOFs, similar to the dog hindlimb space.

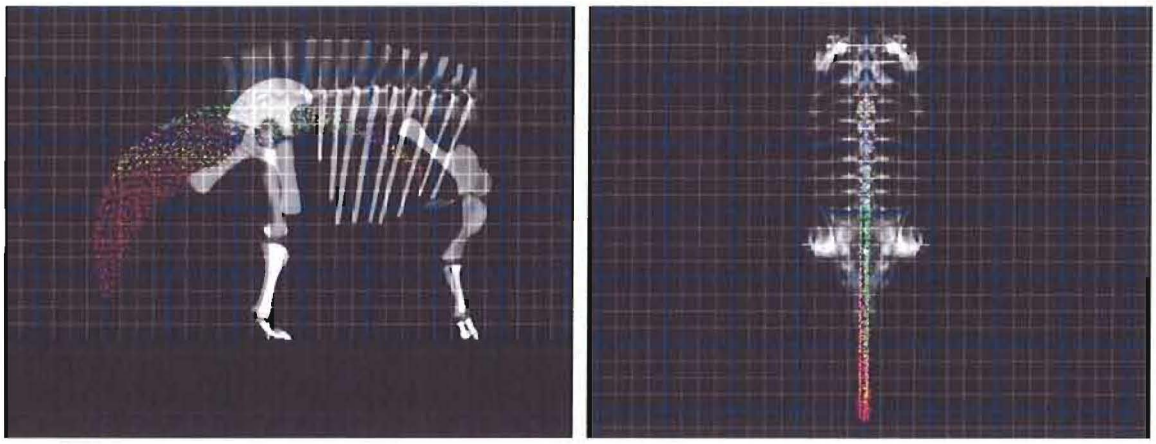


Figure 69. *Apatosaurus* hindlimb LROM space.

The constrained *Apatosaurus* hindlimb LROM space is based on a step length 0.34 times the hip height. The space contains 3,628 samples. Figure 70 shows the constrained *Apatosaurus* hindlimb LROM space.

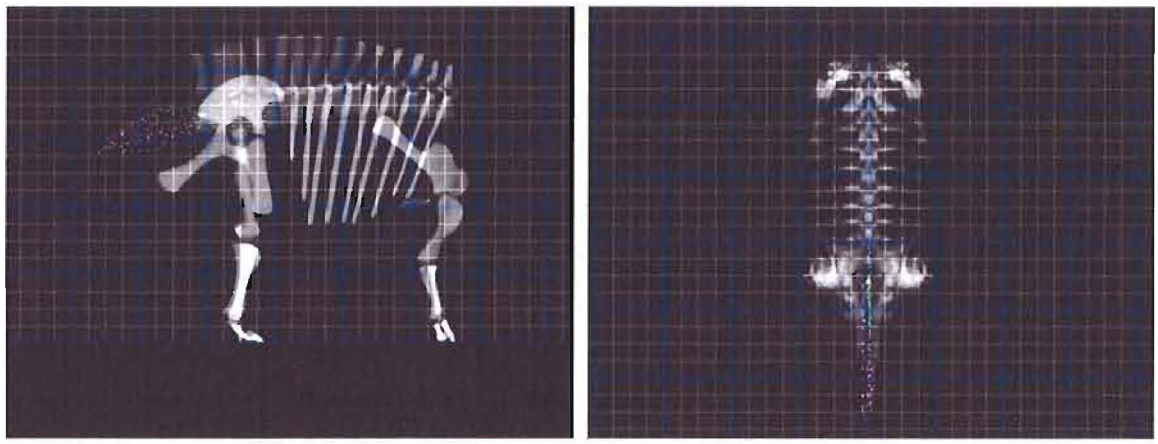


Figure 70. Constrained *Apatosaurus* hindlimb LROM space.

The *Apatosaurus* forelimb LROM was sampled at 4° resolution (5 cm resolution for scapulothorax elevation) creating 5,221,125 samples (119 MB, 24 bytes per sample). Figure 71 shows the *Apatosaurus* forelimb LROM space. The forelimb LROM space is considerably more robust than the hindlimb space, due to non-coincident wrist, elbow, and shoulder flexion/extension, and scapulothorax rotation FDOF axes. Combinations of these FDOF values allow the root to both move along non-parasagittal arcs and change elevation.

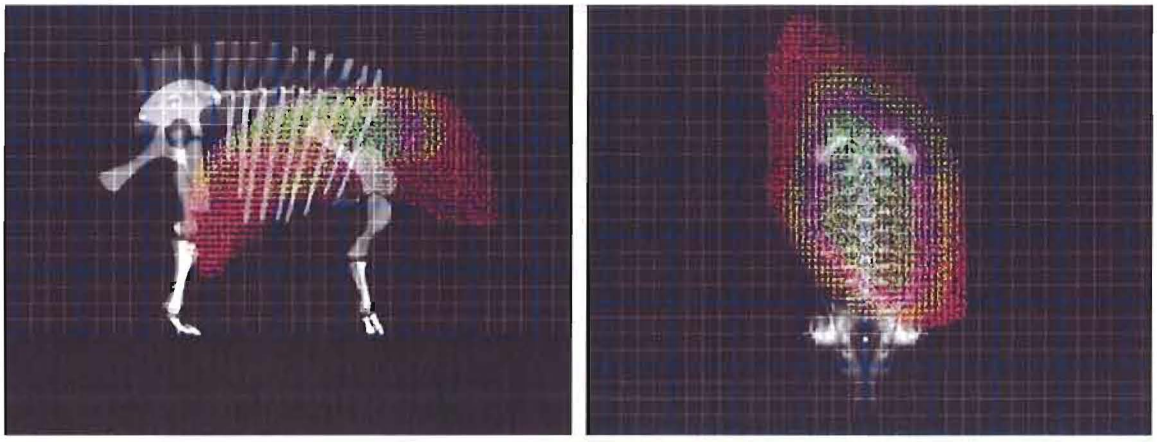


Figure 71. Apatosaurus forelimb LROM space.

The constrained *Apatosaurus* forelimb LROM space is based on a step length 0.34 times the hip height. The space contains 406,377 samples. Figure 72 shows the constrained *Apatosaurus* forelimb LROM space.

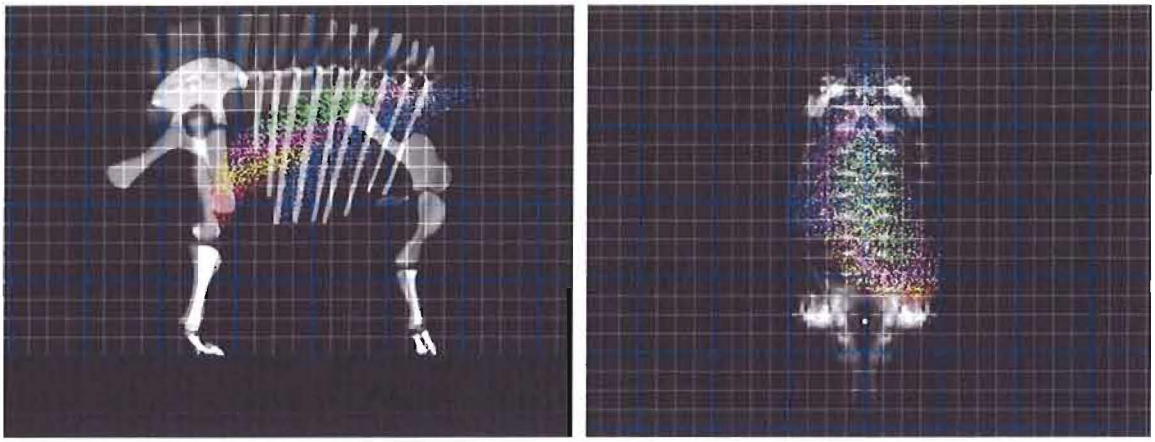


Figure 72. Constrained *Apatosaurus* forelimb LROM space.

The *Apatosaurus* trunk ROM consists of nine vertebral joints. The trunk ROM represents 21 mediolateral samples and 21 dorsoventral for a total of 441 samples. Figure 73 shows the *Apatosaurus* trunk ROM space.

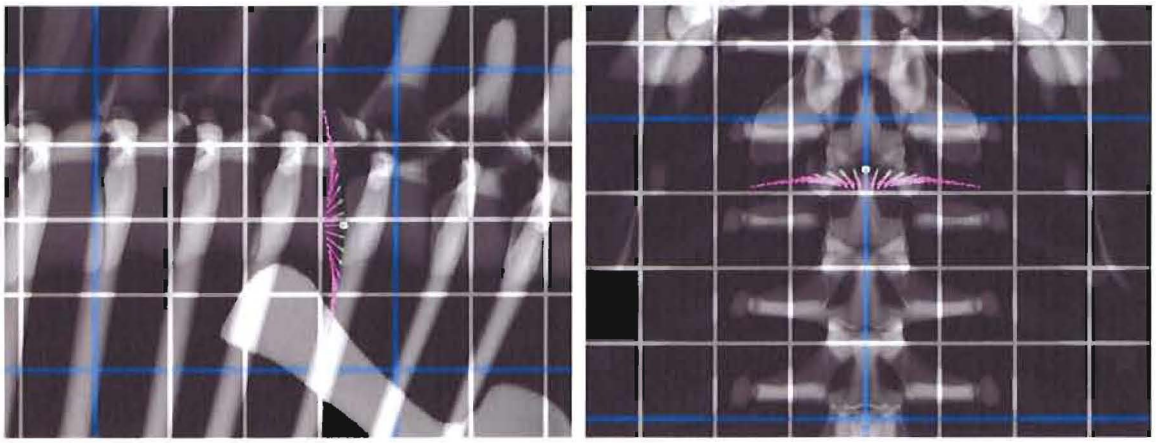


Figure 73. Apatosaurus trunk ROM space.

Triceratops

Triceratops was a ceratopsid dinosaur that lived during the Late Cretaceous Period, approximately 68 to 65 million years ago. *Triceratops* was up to 9 meters in length. Bone dimensions and morphologies are based on the Black Hills Institute of Geological Research *Triceratops horridus* specimen TCM 2001.93.1, "Kelsey" (Stevens, 2007b). Table 33 (see Appendix D) lists the six geometric DOFs for each of the *Triceratops* model's FDOFs.

The *Triceratops* hindlimb utilizes five FDOFs at three joints: hip flexion/extension, abduction/adduction, and medial/lateral rotation; knee flexion/extension; ankle flexion/extension. Figure 74 shows the *Triceratops* hindlimb joints.

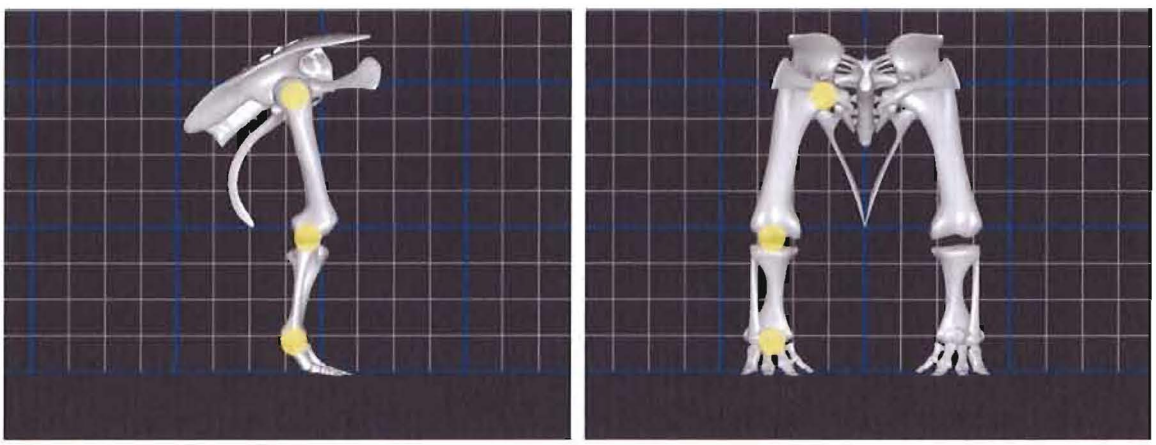


Figure 74. Triceratops hindlimb joints.

The *Triceratops* forelimb utilizes seven FDOFs at four joints: scapulothorax rotation and elevation; shoulder flexion/extension, abduction/adduction, and medial/lateral rotation; elbow flexion/extension, wrist flexion/extension. Figure 75 shows the *Triceratops* forelimb joints.

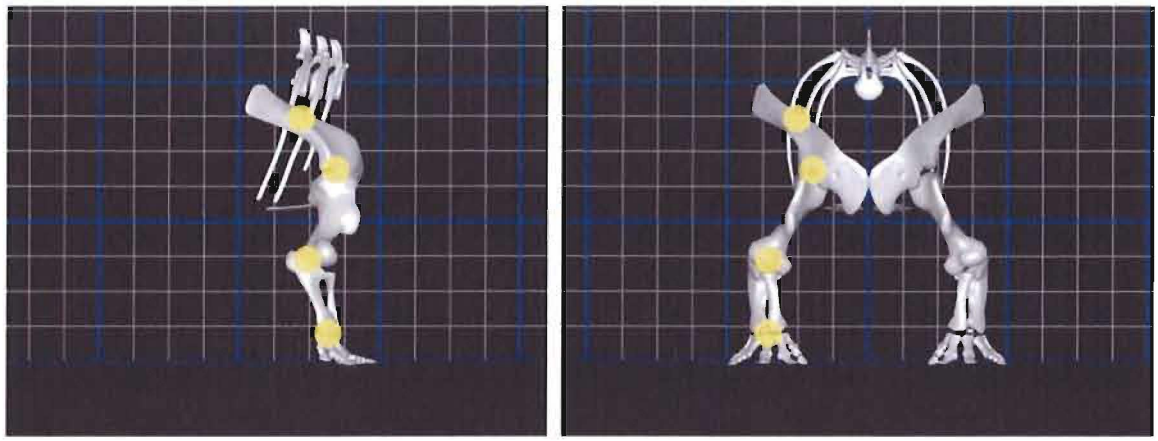


Figure 75. Triceratops forelimb joints.

The *Triceratops* hindlimb LROM was sampled at 4° resolution creating 350,784 samples (8.02 MB, 24 bytes per sample). Figure 76 shows the *Triceratops* hindlimb LROM space. The hindlimb LROM space is nearly parasagittal, which is surprising considering that the knee and hip flexion/extension FDOFs are not coincident. The ankle and knee FDOF axes are nearly coincident, however, allowing combinations of those FDOF values to move the root along a parasagittal arc. The distance between the hip

joint and root is small, so the hip FDOFs are capable of a large amount of root orientation change but little root position change.

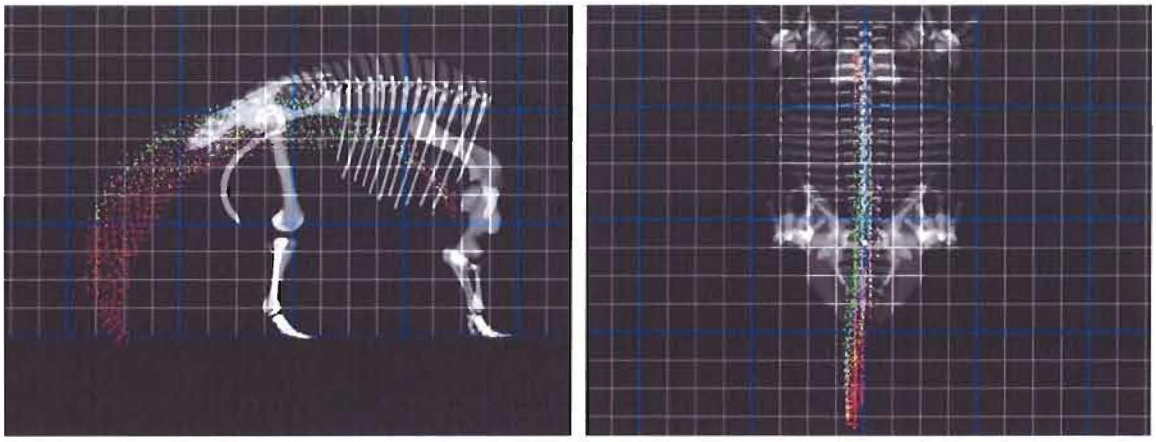


Figure 76. Triceratops hindlimb LROM space.

The constrained *Triceratops* hindlimb LROM space is based on a step length 0.34 times the hip height. The space contains 6,086 samples. Figure 77 shows the constrained *Triceratops* hindlimb LROM space.

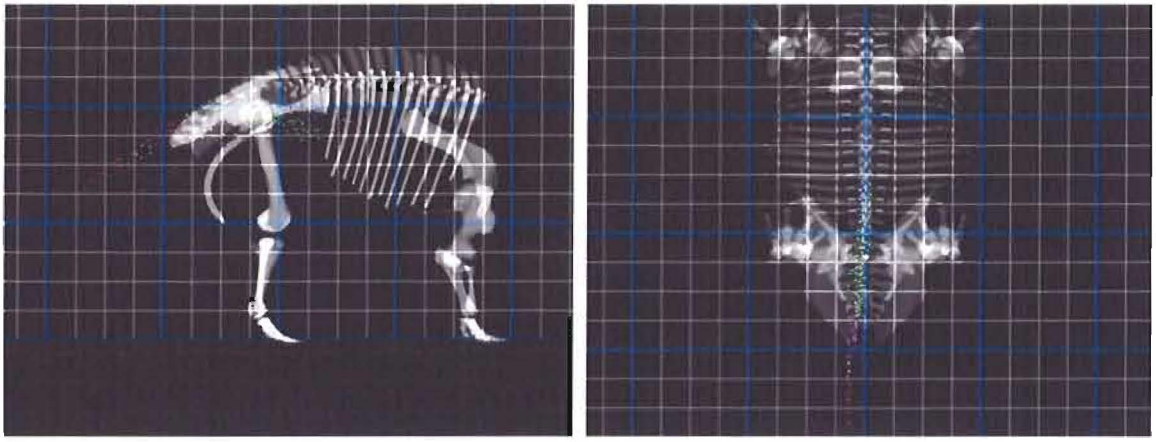


Figure 77. Constrained *Triceratops* hindlimb LROM space.

The *Triceratops* forelimb LROM was sampled at 4° resolution (5 cm resolution for scapulothorax elevation) creating 1,272,960 samples (29.1 MB, 24 bytes per sample). Figure 78 shows the *Triceratops* forelimb LROM space. The non-coincident wrist, elbow, and shoulder flexion/extension, and scapulothorax rotation FDOF axes produce a robust forelimb LROM space, similar to that of the *Apatosaurus* forelimb space.

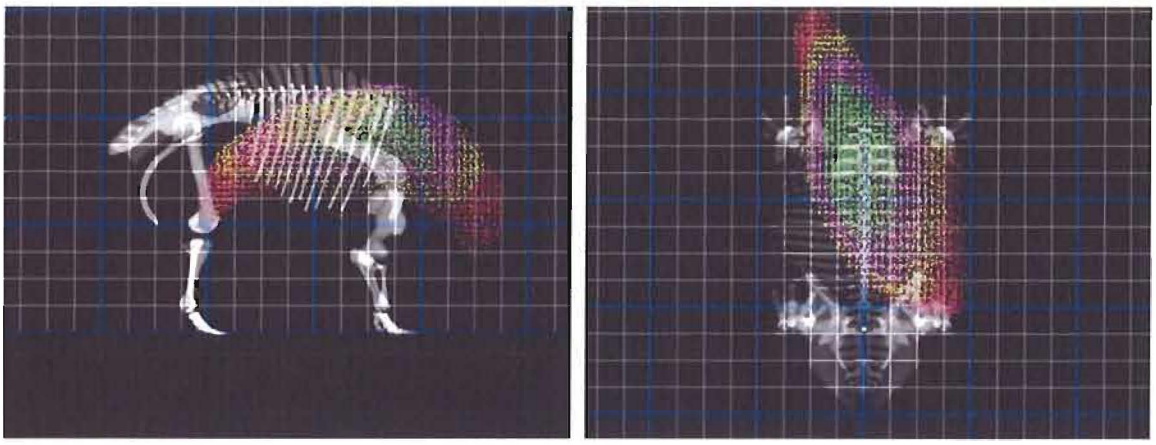


Figure 78. *Triceratops* forelimb LROM space.

The constrained *Triceratops* forelimb LROM space is based on a step length 0.34 times the hip height. The space contains 43,580 samples. Figure 79 shows the constrained *Triceratops* forelimb LROM space.

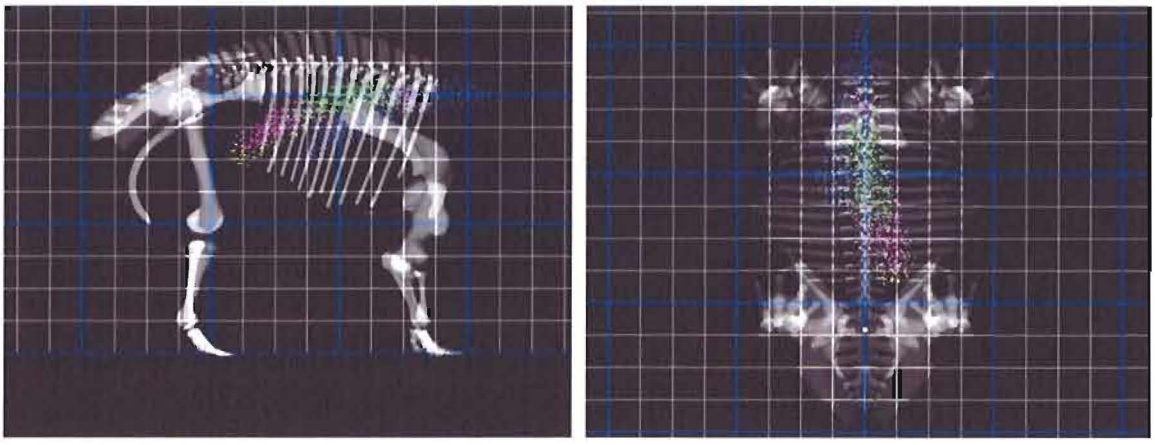


Figure 79. Constrained *Triceratops* forelimb LROM space.

The *Triceratops* trunk ROM consists of 15 vertebral joints. The trunk ROM represents 21 mediolateral samples and 21 dorsoventral for a total of 441 samples. Figure 80 shows the *Triceratops* trunk ROM space.

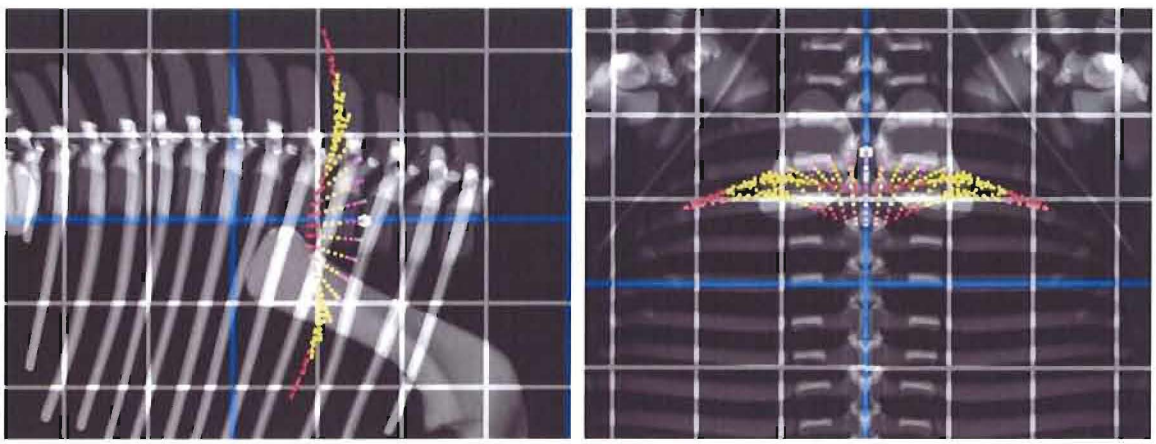


Figure 80. Triceratops trunk ROM space.

Tyrannosaurus

Tyrannosaurus was a theropod dinosaur that lived during the Late Cretaceous Period, approximately 68 to 65.5 million years ago. *Tyrannosaurus* was up to 13 meters in length. Bone dimensions and morphologies are based on the Black Hills Institute of Geological Research *Tyrannosaurus* specimen BHI-3033, "Stan" (Stevens, 2007c).

Table 34 (see Appendix E) lists the six geometric DOFs for each of the *Tyrannosaurus* model's FDOFs.

The *Tyrannosaurus* hindlimb utilizes six FDOFs at four joints: hip flexion/extension, abduction/adduction, and medial/lateral rotation; knee flexion/extension; ankle flexion/extension; phalangeal flexion/extension. Figure 81 shows the *Tyrannosaurus* hindlimb joints.

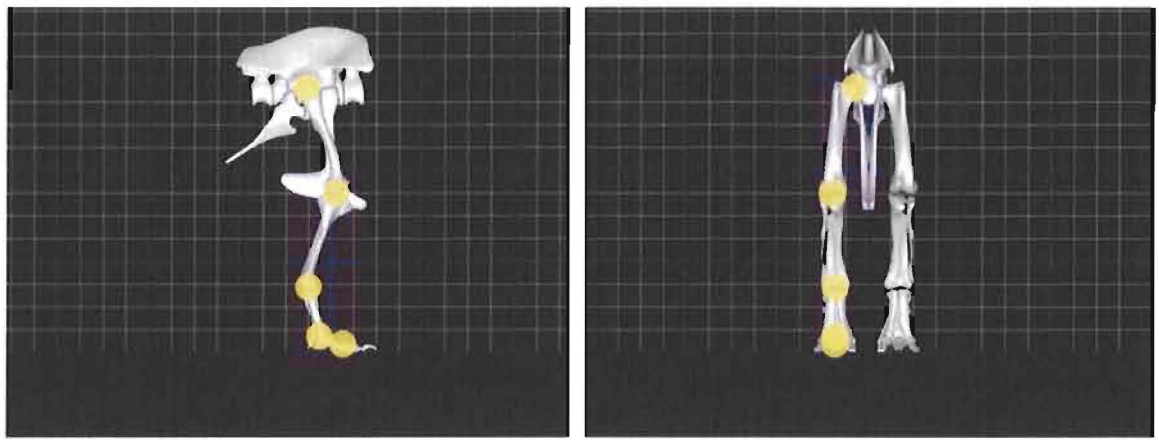


Figure 81. *Tyrannosaurus* hindlimb joints.

The *Tyrannosaurus* hindlimb LROM was sampled at 4° resolution creating 2,882,880 samples (65.9 MB, 24 bytes per sample). Figure 82 shows the *Tyrannosaurus* hindlimb LROM space. The space is highly parasagittal, which is not surprising considering the large number of redundant FDOFs; the phalangeal, ankle, knee, and hip flexion/extension FDOFs all have near-coincident axes. The large number of redundant

FDOFs creates an extremely large number of LROM space samples (i.e., 2,882,880 samples, compared to 80,640 samples in the comparable dog hindlimb LROM space). A GA may have trouble selecting appropriate limb configurations with so many combinations of phalangeal, ankle, and knee FDOF values allowing similar root orientations and positions.

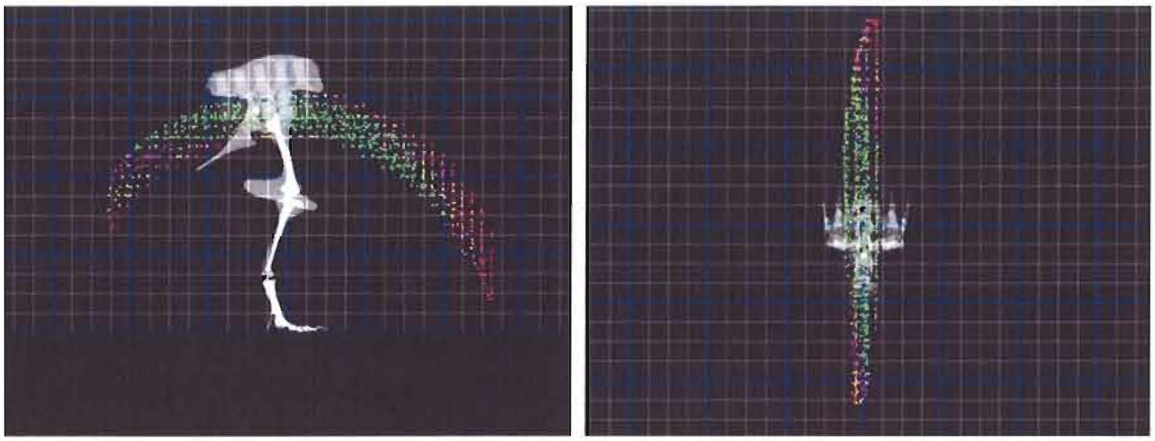


Figure 82. Tyrannosaurus hindlimb LROM space.

The constrained *Tyrannosaurus* hindlimb LROM space is based on a step length 0.98 times the hip height. The space contains 24,346 samples. Figure 83 shows the constrained *Tyrannosaurus* hindlimb LROM space.

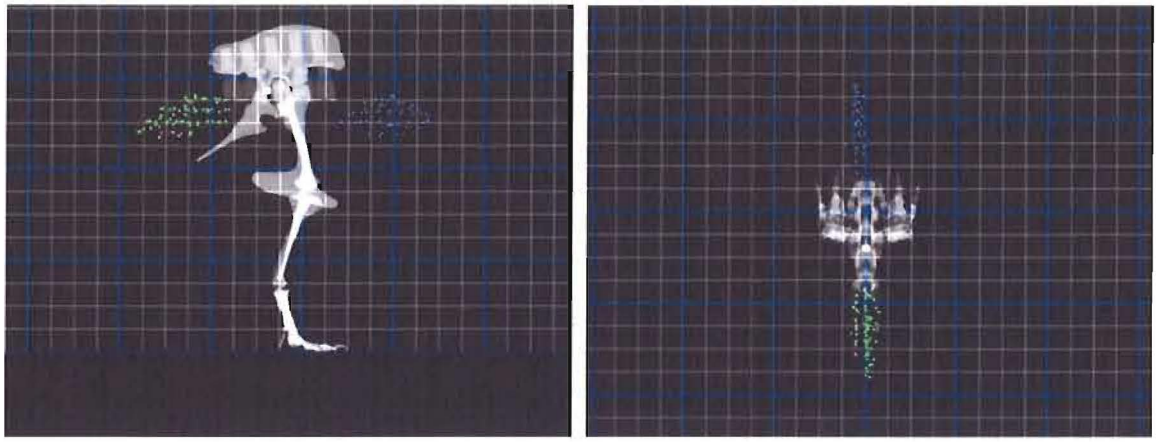


Figure 83. Constrained *Tyrannosaurus* hindlimb LROM space.

Summary

The methods presented in this chapter allow the exploration and evaluation of a limb's summary range of motion. LROMs are created by evaluating all combinations of a limb's FDOFs at some angular resolution. LROMs are subsequently organized into LROM spaces for visualization and experimentation. LROM spaces provide an intuitive map of the positions and orientations reachable by the root of the limb system which is particularly useful for observing the behaviors of limb joints during locomotion.

Bipedal constraints (i.e., dual support and bilateral) provide substantial pruning of LROM spaces, removing configurations with positions and orientations that are not possible during locomotion. The GA selects candidates from the LROM space corresponding to the beginning and ending of the stance phase. Compared to mid stance,

there are relatively-few ways that a limb can reach forward and back (i.e., given an adequately- long step length), so joint functionality is largely unambiguous. Limb movements are dependent on the limb's FDOFs, so limb movements and joint functionality can be analyzed in terms of the contributions made by each FDOF during locomotion.

The GA utilizes a candidate representation which guarantees that all possible solutions at least cause the limbs to propel the root position forward by a specified step length. Therefore, the GA need only be concerned with finding a solution that is optimally smooth in terms of the fitness function, allowing fast convergence. The fitness function discourages body pitching, yawing, and rolling, as well as lateral swaying, vertical bobbing, and unnecessary angular excursions at the joints. A pipeline architecture connects the discrete operations necessary to automatically generate walking gaits, and allows maximum reuse of data between operations. Appendix H demonstrates the process of selecting and evaluating a bipedal candidate gait.

Quadrupedal walking gaits can be automatically generated by utilizing two parallel bipedal gaits connected by a trunk region. Exercising the trunk's ROM allows further pruning of the fore and hind LROM spaces to remove configurations which cannot be connected by the trunk. Similar to the bipedal candidate representation, the quadrupedal candidate representation guarantees that all possible solutions at least propel the root position forward by a specified step length and ipsilateral phase. Again, a pipeline architecture connects the discrete quadrupedal operations, allowing reuse of data computed at earlier stages.

Finally, five models were presented: two extant, three extinct; four quadrupedal, one bipedal. Resulting LROM spaces, both unconstrained and constrained, were presented along with visualizations of the trunk ROMs (for the quadrupeds only). In the next chapter, sensitivity analyses will be presented that use these models to determine the genetic parameters used by the GAs and for determining the impact of the LROM-space-generation parameters on solution fitness. The models will also be evaluated qualitatively with respect to functional joint behavior during locomotion, and finally evaluated quantitatively to test specific hypotheses about FDOF behaviors.

CHAPTER IV

RESULTS AND ANALYSIS

In this chapter, data will be presented from sensitivity analyses of the parameters associated with the GAGA process and from studies carried out using GAGA methods. First, data will be presented related to the selection of the GA genetic parameters and parent selection method. Next, a sensitivity analysis will explore the effect of the parameters used to create and organize LROM spaces. Next, a qualitative analysis will compare walking gaits generated with GAGA to analogous real-world gaits. Finally, specific quantitative studies will test gait hypotheses using GAGA methods.

Genetic Parameters

The following sections provide the results of a sensitivity analysis for the genetic parameters used by the GA. The sensitivity analysis provides a means for tuning the GA and verifies that the chosen genetic parameters values are appropriate for the solution space. All test runs were based on the generation of *Apatosaurus* quadrupedal walking gaits using 39-bit binary candidates (i.e., 10 bits representing an index into the back data,

7 bits representing an index into the slice data, and two 11-bit integers representing indices into the forelimb space based on the previous two hindlimb indices). Unless otherwise noted, all runs consist of 10,000 GA iterations with a candidate population size of 1000. All GA runs were executed ten times so that an accurate mean and standard deviation could be established.

Parent Selection

After each GA iteration (i.e., following crossover and mutation), parents are selected to populate the candidate population for the next GA iteration. To select parents, the GA compares each candidate solution with another randomly-selected candidate solution in the population. The candidate solution with the higher fitness value then has a fixed change of being selected. This selection method is known as Tournament Selection. Tournament Selection has an advantage over other selection methods (e.g., Rank Selection or Fitness Proportional Selection) in that it does not require sorting of the population, therefore decreasing computation time. Table 1 shows the effect of varying the Tournament Selection higher-fitness-selection coefficient (0.5 crossover coefficient, 0.5 relative mutation coefficient).

Table 1.

Effect of varying Tournament Selection coefficient

	Tournament Selection coefficient										
Fitness (x10^3)	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
M	5.73	6.75	6.90	7.23	7.66	8.08	8.57	8.68	8.64	8.71	8.68
SD	0.15	0.08	0.20	0.28	0.32	0.26	0.16	0.19	0.24	0.37	0.32

Tournament Selection performed best with a selection coefficient between 0.6 and 1.0. A Tournament Selection coefficient of 0.9 was used for actual GA runs and for the remaining sensitivity analysis runs (see following sections). Always choosing the lower-fitness candidate (i.e., coefficient of 0.0) provides a fitness of 5.73 ± 0.15 (x 10^{-3}), providing a good estimate of the worst-case solutions. Figure 84 shows a visual representation of Table 1.

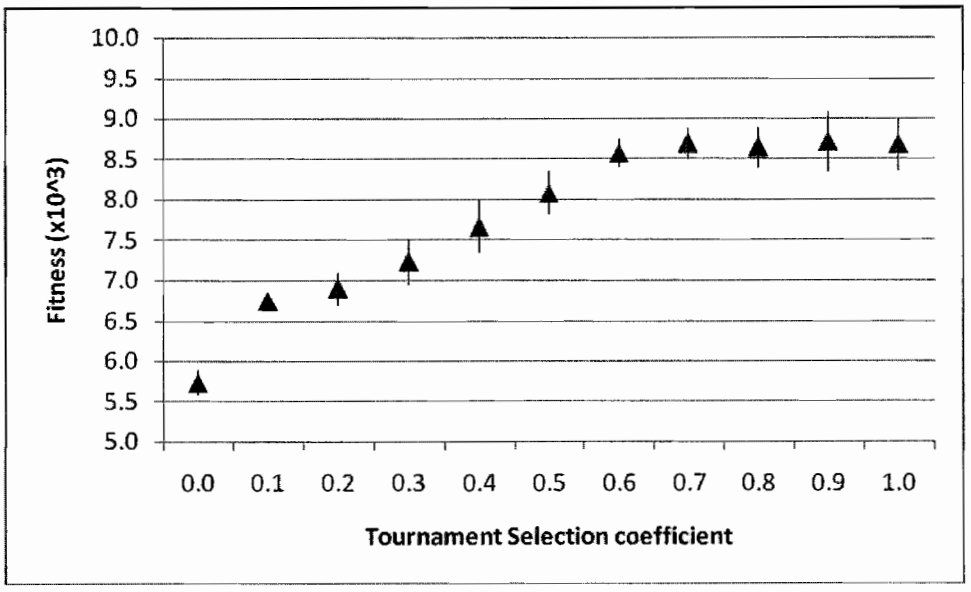


Figure 84. Effect of varying Tournament Selection coefficient.

Crossover Coefficient

During each GA iteration, each candidate solution has a chance of being mated with another randomly-selected candidate solution based on the crossover coefficient. Candidates are mated by selecting a single crossover index; the first candidate receives the second candidate's genetic information up to a randomly-selected crossover index while the second candidate receives the first candidate's genetic information after the crossover index. Table 2 shows the effect of varying crossover coefficient in the absence of mutation.

Table 2.

Effect of varying crossover coefficient with no mutation

	Crossover coefficient										
Fitness (x10^3)	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
M	6.13	7.79	8.26	8.59	8.52	8.77	8.69	8.71	8.90	8.83	8.71
SD	0.24	0.66	0.52	0.42	0.41	0.36	0.28	0.36	0.41	0.29	0.28

Without mutation, the GA depends on crossover to explore the space. Not surprisingly, the GA performed poorly with a crossover coefficient below 0.5. 0.8 is henceforth considered a high-performance crossover coefficient for the GA. Figure 85 shows the results of Table 2.

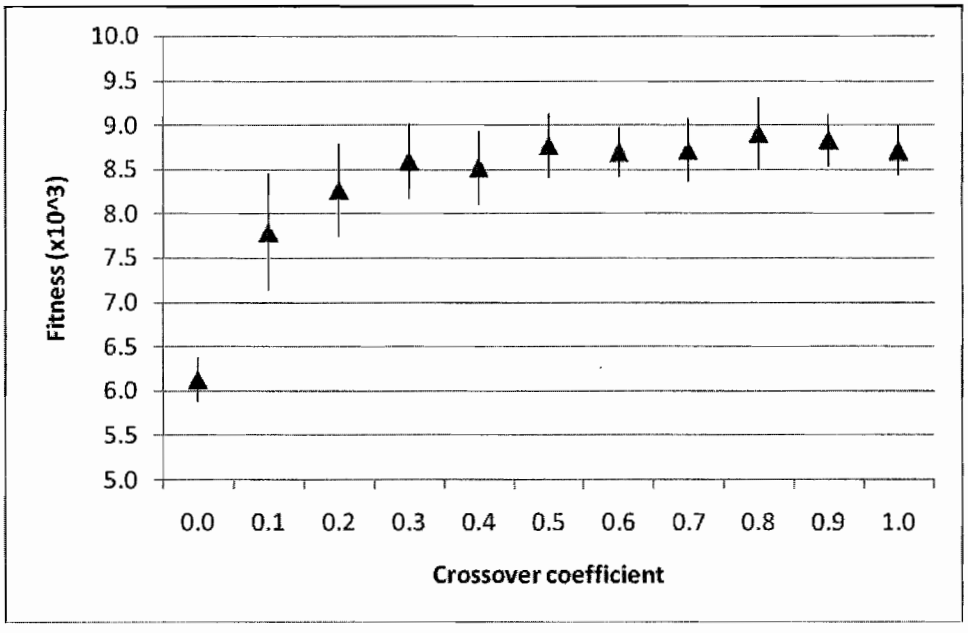


Figure 85. Effect of varying crossover coefficient with no mutation.

With mutation enabled, a tradeoff must be made between the crossover coefficient and the mutation coefficient; a great amount of combined mating and mutation will perturb too many high-fitness candidates and hinder the success of the GA. Table 3 shows the effect of varying crossover coefficient with a high-performance relative mutation coefficient of 0.5 (see next section).

Table 3.

Effect of varying crossover coefficient with fixed mutation

	Crossover coefficient										
Fitness (x10^3)	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
M	8.97	9.02	8.96	8.96	8.79	8.74	8.63	8.66	8.53	8.57	8.63
SD	0.35	0.41	0.33	0.35	0.37	0.32	0.27	0.31	0.06	0.11	0.21

As predicted, a high crossover coefficient hindered GA convergence performance when using a significant amount of mutation. Figure 86 shows a visual representation of Table 3.

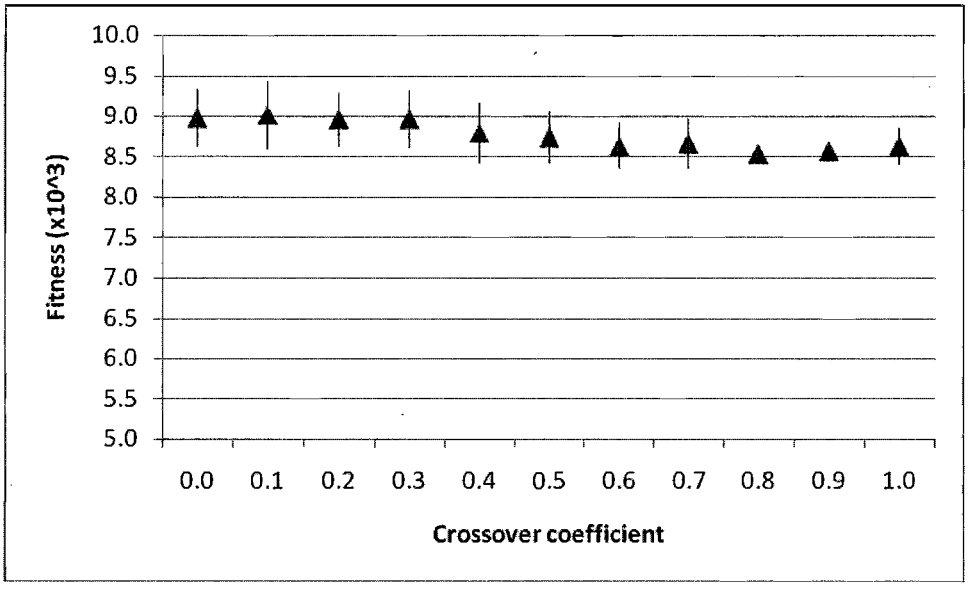


Figure 86. Effect of varying crossover coefficient with fixed mutation.

Mutation Coefficient

During each GA iteration, every bit in the solution space has a chance of being flipped, thus mutating some of the candidate solutions. The mutation coefficient is scaled by the inverse of the candidate size so that a relative mutation coefficient of 1.0 implies that one bit per candidate will be flipped on average. Table 4 shows the effect of varying the relative mutation coefficient in the absence of crossover.

Table 4.

Effect of varying relative mutation coefficient with no crossover

	Relative mutation coefficient										
Fitness (x10^3)	1/32	1/16	1/8	1/4	1/2	1	2	4	8	16	32
M	8.05	7.89	8.39	8.89	9.16	8.75	8.66	8.48	8.43	8.25	8.34
SD	0.63	0.58	0.58	0.52	0.35	0.20	0.14	0.25	0.35	0.27	0.38

The GA benefited most from a relative mutation coefficient of 0.5. Coefficients below 0.5 did not perturb the solution space enough to generate high-fitness candidates. Coefficients above 0.5 perturbed the space so much that too many high-fitness candidates were destroyed, impeding GA convergence performance. Figure 87 shows a visual representation of Table 4.

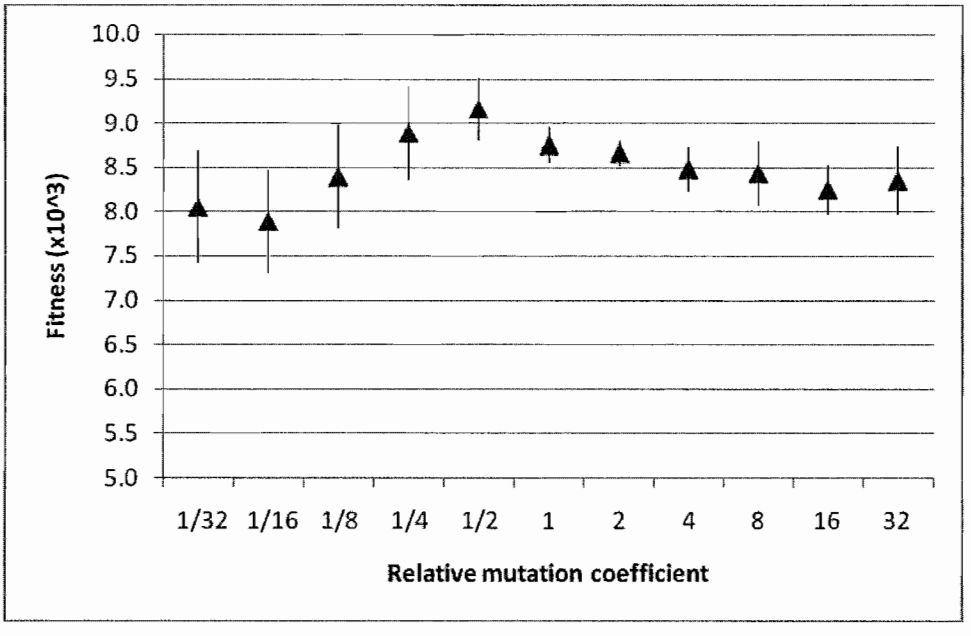


Figure 87. Effect of varying relative mutation coefficient with no crossover.

Table 5 shows the effect of varying relative mutation coefficient with a high-performance fixed crossover coefficient of 0.8 (see previous section).

Table 5.

Effect of varying relative mutation coefficient with fixed crossover

	Relative mutation coefficient										
Fitness (x10^3)	1/32	1/16	1/8	1/4	1/2	1	2	4	8	16	32
M	8.66	8.64	8.67	8.55	8.64	8.66	8.60	8.44	8.35	8.16	8.16
SD	0.32	0.31	0.34	0.16	0.20	0.09	0.14	0.23	0.21	0.25	0.31

Similar to varying crossover coefficient with a high-performance relative mutation coefficient, higher relative mutation coefficients hindered GA convergence performance when a high-performance crossover coefficient was used. GA convergence performance was best when using a high performance relative mutation coefficient with a modest crossover coefficient, so a relative crossover coefficient of 0.5 was used with a crossover coefficient of 0.1 for actual GA runs. Figure 88 shows Table 5 in graph form.

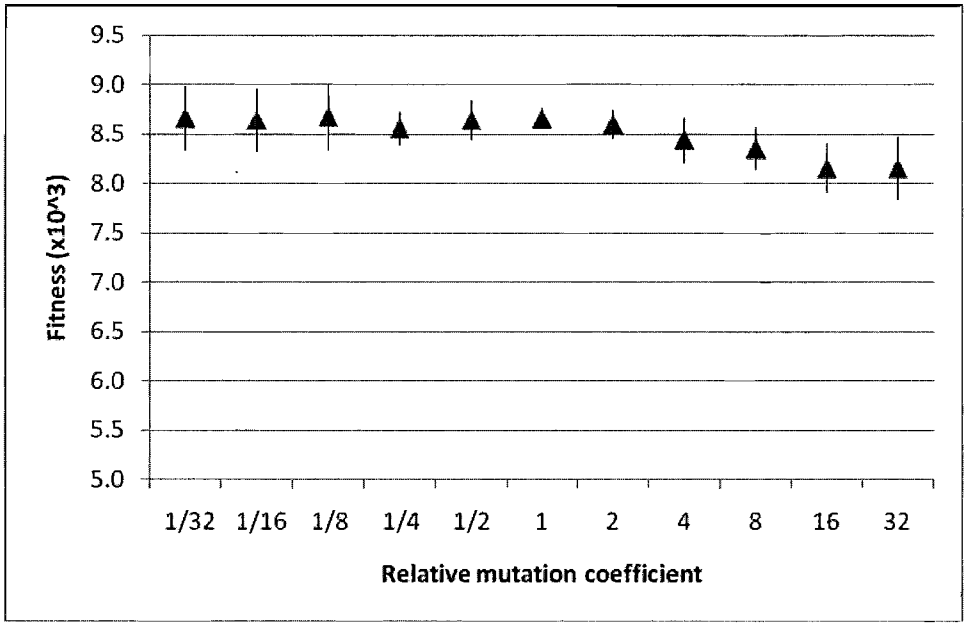


Figure 88. Effect of varying relative mutation coefficient with fixed crossover.

Candidate Population Size

The candidate population size determines the amount of initial randomness in the solution space; larger populations have a greater chance of generating initial candidates with high fitness. Larger populations also allow a greater amount of genetic material to be transferred between GA iterations, which provides more genetic diversity for mating and mutation. Table 6 shows the effect of varying candidate population size (0.1 crossover coefficient, 0.5 relative mutation coefficient). The GA was limited to 100 runs to exaggerate the effect of limiting population size.

Table 6.

Effect of varying candidate population size

	Candidate population size										
Fitness (x10^3)	10	100	200	300	400	500	600	700	800	900	1000
M	6.80	8.39	8.35	8.54	8.61	8.71	8.90	8.79	8.80	8.91	8.96
SD	0.62	0.63	0.61	0.50	0.37	0.39	0.32	0.27	0.35	0.36	0.35

Maximum fitness grew almost monotonically with candidate population size, indicating that larger populations do improve GA convergence performance. The standard deviations also decreased significantly as population size increased, suggesting that larger populations allow more consistency in the candidate solutions found. Figure 89 shows a visual representation of Table 6.

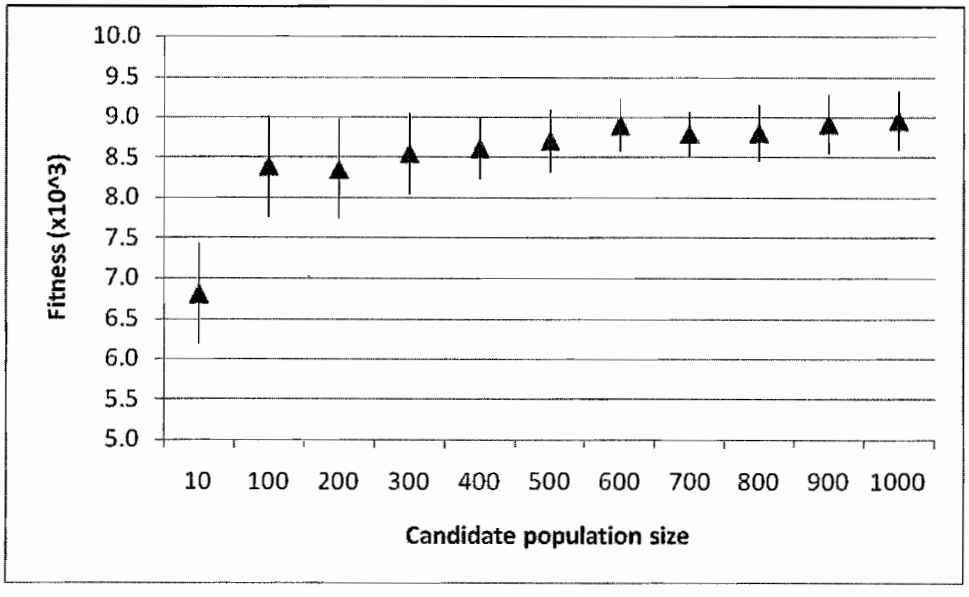


Figure 89. Effect of varying candidate population size.

Convergence Comparison

The GA was run against a simulated Hill Climbing algorithm to determine the relative convergence performance of the GA. Hill Climbing was simulated by running the GA with no crossover, a relative mutation coefficient of 1.0, and Best Selection to select population parents (i.e., the most-fit candidate in each population is used to populate the entire next population). In this way, the simulated Hill Climbing algorithm flipped one bit per candidate on average. The simulated Hill Climbing algorithm used a population size of 39 (i.e., the same size as the solution candidates) so that each genotype bit is flipped once per iteration on average.

The GA was run with a crossover coefficient of 0.1 and a relative mutation coefficient of 0.5 (see previous sections). The GA used a population size of 39 to be consistent with the simulated Hill Climbing algorithm (at the cost of some GA convergence performance). With equal candidate solution and population sizes, the two algorithms should be approximately equal in terms of computational cost. Table 7 shows the results of the GA/Hill Climbing comparison.

Table 7.

Comparison of GA/Hill Climbing convergence performance

	Number of iterations										
Fitness (x10^3)	10	100	200	300	400	500	600	700	800	900	1000
						GA					
M	6.17	7.53	7.81	7.94	7.99	8.10	8.11	8.11	8.14	8.23	8.23
SD	0.39	0.49	0.48	0.51	0.53	0.59	0.60	0.60	0.64	0.58	0.57
						Hill Climbing					
M	6.59	7.16	7.32	7.42	7.43	7.44	7.44	7.45	7.46	7.47	7.47
SD	0.53	0.58	0.54	0.65	0.65	0.67	0.67	0.68	0.68	0.68	0.68

Initially, the simulated Hill Climbing algorithm outperformed the GA. This is likely a result of the Hill Climbing algorithm's more aggressive rewarding of fit candidates; the GA sometimes selects less-fit candidates to maintain diversity in the candidate population. The GA outperformed the simulated Hill Climbing algorithm at 100 iterations and beyond. After 300 iterations, the simulated Hill Climbing algorithm improved very little in fitness because it likely converged on a local maximum. The GA continued to improve until at least 900 iterations. Figure 90 shows the results from Table 7 in graph form.

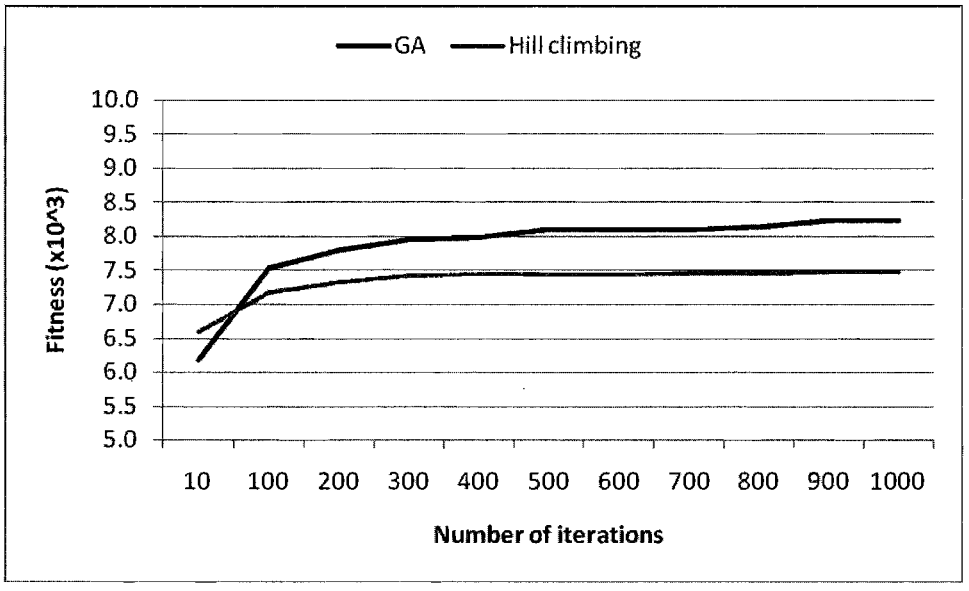


Figure 90. Comparison of GA/Hill Climbing convergence performance.

The data presented in this section demonstrates the process of tuning the GA for maximum convergence. With the GA tuned, the sensitivity of input parameters on GA performance must next be evaluated. The next section will present data related to the various input parameters for exploring and organizing LROM spaces and for finding smooth bipedal and quadrupedal walking gait paths.

Sensitivity Analysis

The LROM space representation allows the search for "optimal" walking gait paths through the space, but how sensitive are the generated gaits to variations in the parameters used to generate the space? In this section, the sensitivity of space

exploration and organization parameters will be analyzed. The effects of iteratively refining spaces will also be addressed. Finally, the aggregate fitness function coefficients were varied to determine their qualitative effects on the generated walking gaits. All data in this section is based on the *Apatosaurus* forelimb, which utilizes seven FDOFs with a wide variety of functionality. Data was collected across ten GA runs to determine a mean and standard deviation for each data point.

Space Exploration

Each LROM is sampled at a specified angular resolution. The exact resolution is typically selected such that a large number of samples are generated, but not so many that computer memory and execution time become limiting factors. The number of samples grows exponentially with the sampling resolution (i.e., double the sampling resolution increases the size of the space by a factor of 2^*FDOF count*). The exact number of samples depends on the angular sampling resolution and the sum angular excursion of a limb's FDOFs. Figure 91 shows how the number of samples and the number of constrained samples (i.e., using the dual support and bilateral symmetry constrained) relates to sampling resolution.

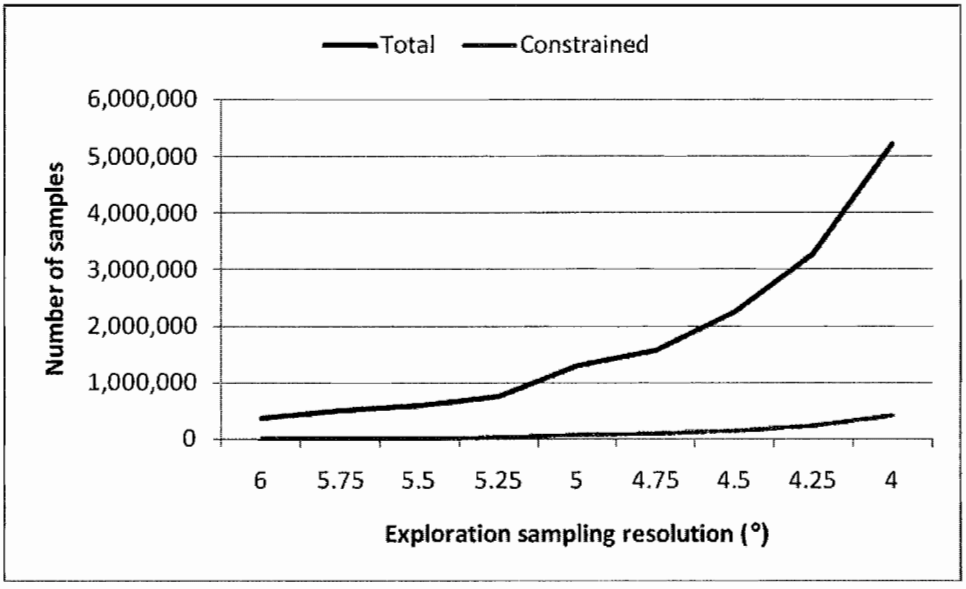


Figure 91. *Apatosaurus* forelimb sample counts.

The total number of LROM samples grows exponentially, as expected. The number of constrained samples grows less quickly, but still increases from 7,719 samples at a 6° angular sampling resolution to 406,377 samples at a 4° sampling resolution. Fitness values were collected for sampling resolutions between 4° and 6° determine the effect of the angular sampling resolution on generated walking gait fitness. Table 8 shows the effect of varying angular sampling resolution.

Table 8.

Effect of varying sampling resolution on walk fitness

	Exploration sampling resolution (°)								
Fitness (x10^3)	6	5.75	5.5	5.25	5	4.75	4.5	4.25	4
M	7.22	8.99	9.18	9.64	8.45	8.61	9.00	9.79	10.39
SD	0.84	0.00	0.34	1.08	1.03	1.13	0.89	1.64	1.04

The GA finds paths through the constrained LROM spaces, so it makes sense to evaluate fitness relative to the size of the constrained spaces. Table 8 shows that there is little variation in fitness based on sampling resolution, while Figure 91 shows that the size of the constrained spaces grows quickly as sampling resolution decreases (i.e., increasing the size of the space). This contrast indicates that the fitness of generated walking gaits is not dependent on the number of samples in the LROM space. Figure 92 shows a graphical representation of Table 8.

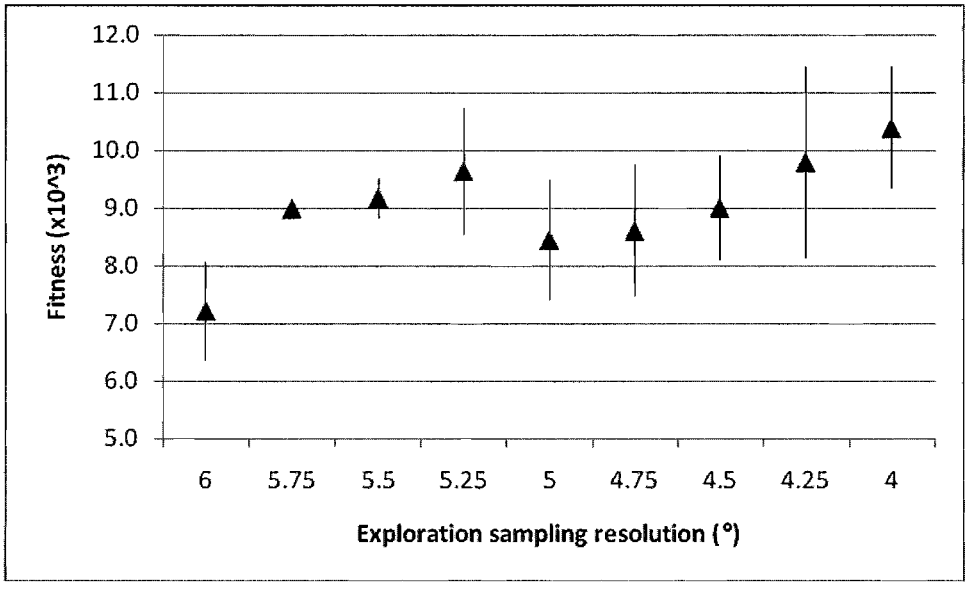


Figure 92. Effect of varying sampling resolution on walk fitness.

Qualitatively, each FDOF accomplishes some functional purpose during locomotion. The functional purposes of the FDOFs must not be dependent on the space sampling or organization parameters, otherwise FDOF functionality could not be determined using the LROM space representation. Figure 93 shows stance frames from *Apatosaurus* forelimb walking gaits generated from LROM spaces sampled at 4° (i.e., 7,719 samples) and 6° (i.e., 406,377 samples) resolutions. The figure shows minor differences in some joint angles, but that the seven FDOFs accomplish similar goals in terms of body pitch, yaw, roll, vertical displacement, and forward travel.

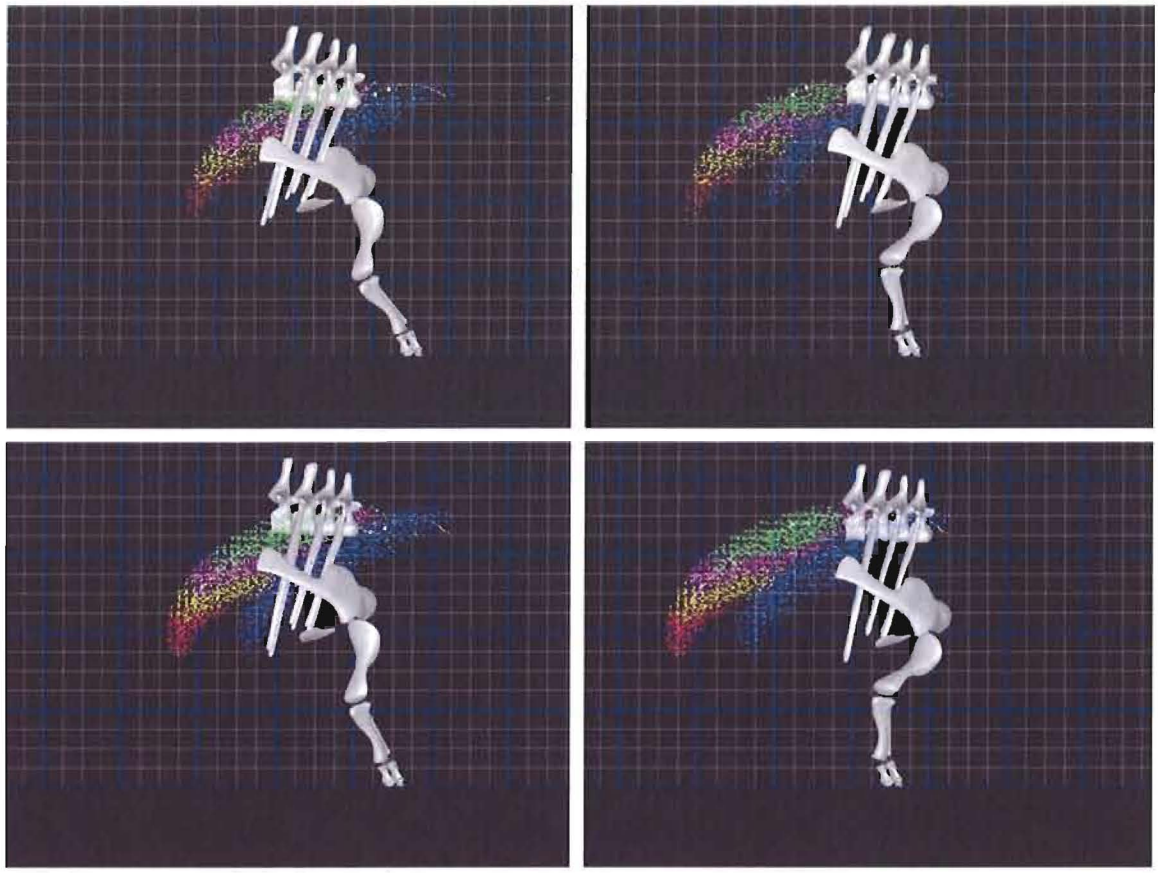


Figure 93. Comparison of low (top) and high (bottom) resolution sampling.

The angular sampling resolution of the sampling process does not have a significant effect on walking gait fitness. The sampling resolution also does not have a significant effect on the functional purpose of the animal's joints during locomotion. The next section will present data collected on the effect of varying space organization parameters on gait fitness.

Space Organization

The samples of a constrained LROM space are sorted into a number of 3D boxes. The number of boxes along the direction of travel is specified and the number of boxes along the other two axes (i.e., lateral and vertical) is computed such that the boxes are geometric cubes. Larger box counts improve the computational efficiency of the GA because each box contains relatively-few samples so fewer evaluations are necessary. However, very large box counts can create a space that is too sparsely populated for the GA to find high-fitness walking gaits. Fitness values were collected for spaces created with box counts between 30 and 60 to determine the effect of box count on gait fitness. Table 9 shows the effect of box count on walking gait fitness.

Table 9.

Effect of varying LROM space box count on walk fitness

	LROM space boxes along direction of travel						
Fitness (x10^3)	30	35	40	45	50	55	60
M	10.13	8.74	10.16	9.47	9.71	8.94	8.87
SD	1.34	1.02	1.61	0.69	1.45	1.23	0.85

Similar to the effect of exploration sampling resolution, the box count has little effect on the fitness of generated walks. The fitness data does show a trend towards lower fitness values with larger box counts, which is to be expected considering that

relatively-fewer samples are available to be evaluated by the GA. Figure 94 shows a graphical representation of the data from Table 9.

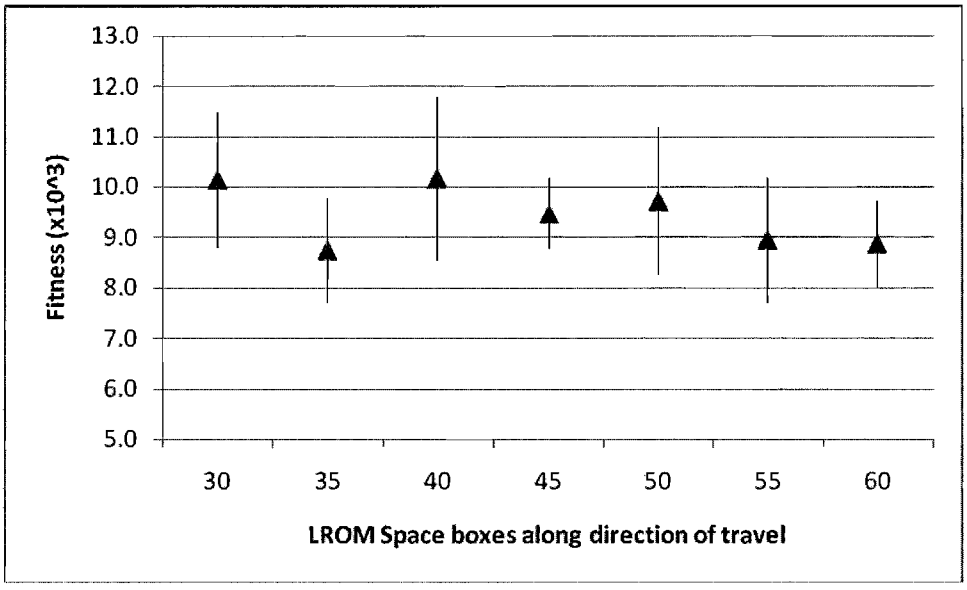


Figure 94. Effect of varying box count on walk fitness.

Like sampling resolution, organization box count must not have an effect on the functional purposes of the FDOFs. Figure 95 shows stance frames from *Apatosaurus* forelimb walking gaits generated from LROM spaces organized with box counts of 30 and 60. Again, the figure shows small differences in joint angles and that the FDOFs accomplish the same high-level locomotion goals between the two walking gaits; analogous FDOFs are utilized for the same locomotion purposes by both gaits.

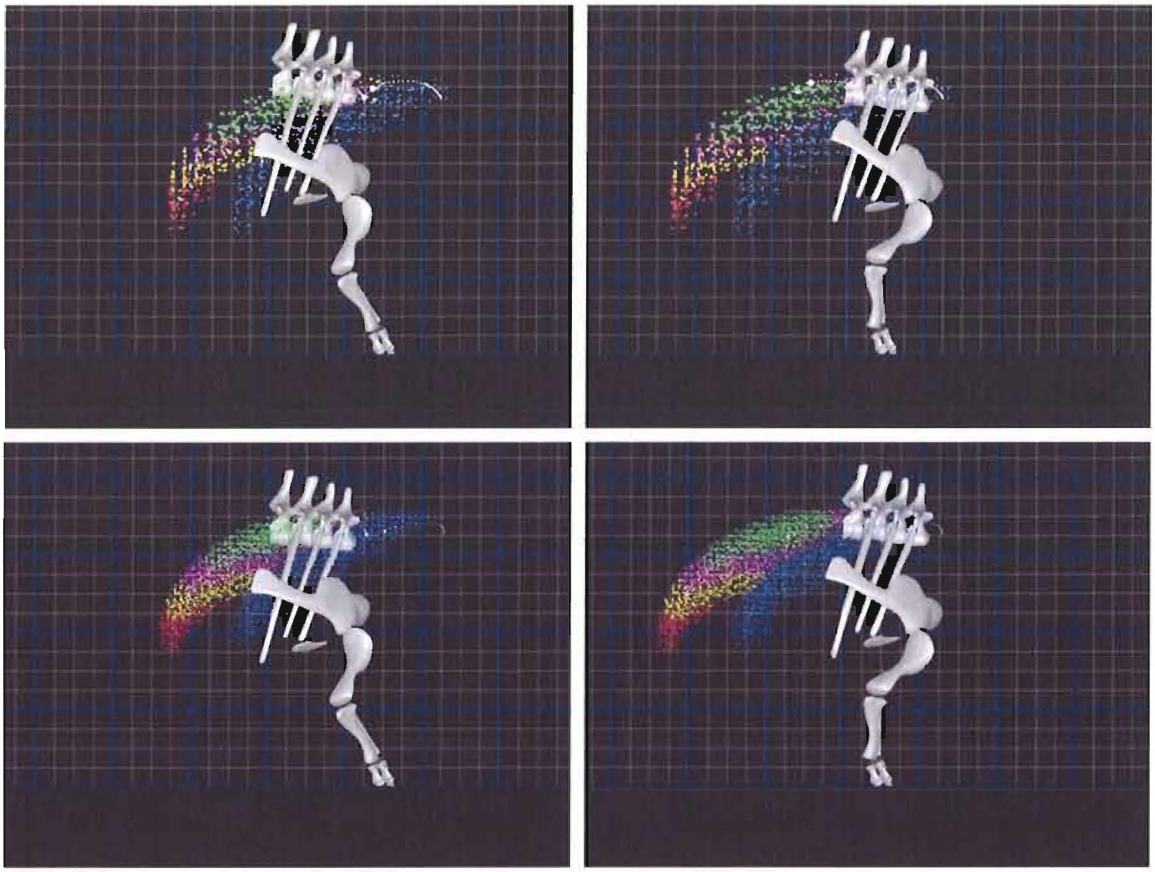


Figure 95. Comparison of low (top) and high (bottom) box counts.

The LROM space box count does not have a significant effect on walking gait fitness. The box count also does not have a significant effect on the functional purpose of the animal's joints during locomotion. In the next section, data will be presented related to the iterative refinement of FDOFs. The refinement process adds resolution to LROM spaces at locations found to be related to locomotion.

Space Refinement

After generating a walking gait from an LROM space created with some angular sampling resolution and box count, the space can be iteratively refined to populate the LROM space with more samples in locations found to be related to locomotion. A new LROM space is created for a target FDOF using the existing walking gait keyframes to specify all other FDOF values. The target FDOF is then sampled at a higher resolution, a new walking gait is generated, and the new gait animation is used to resample the next FDOF. Table 10 shows a comparison of candidate fitness values before and after iterative refinement at a 1° angular resolution.

Table 10.

Comparison of original and refined walk fitness

	Exploration sampling resolution (°)								
Fitness (x10^3)	6	5.75	5.5	5.25	5	4.75	4.5	4.25	4
					Original				
M	7.22	8.99	9.18	9.64	8.45	8.61	9.00	9.79	10.39
SD	0.84	0.00	0.34	1.08	1.03	1.13	0.89	1.64	1.04
					Refined				
M	130.61	23.98	41.54	29.65	101.00	83.30	96.54	123.17	61.45
SD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00	11.34

The refinement process substantially increases candidate fitness, in some cases by an order of magnitude or more. The iterative process allows each FDOF to be fine-tuned to better match the fitness criteria, explaining the large increase in candidate fitness.

Figure 96 shows a graphical representation of candidate fitness before and after the iterative refinement process.

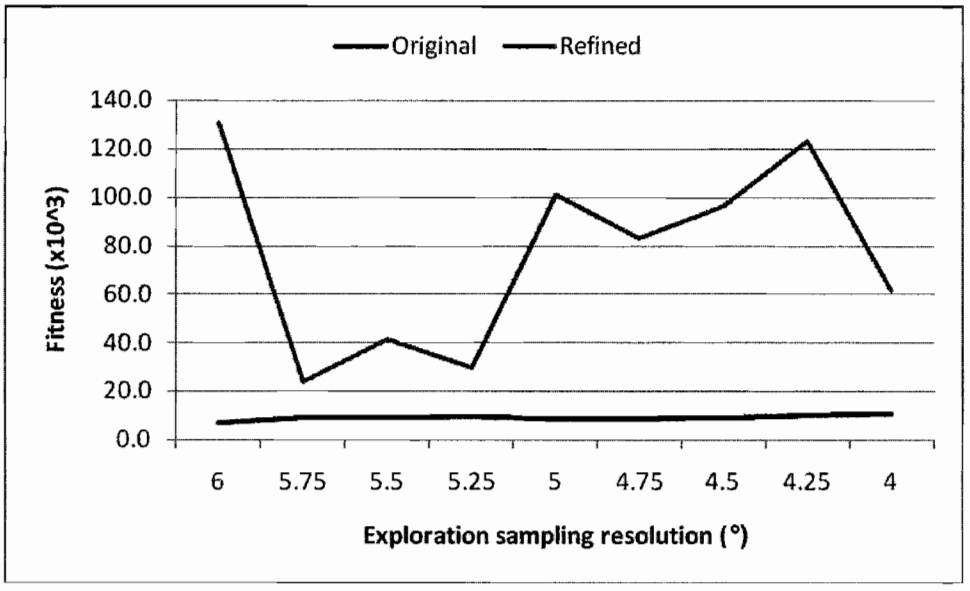


Figure 96. Effect of iterative refinement on candidate fitness.

The refinement process can substantially increase candidate fitness, but does it alter the functional purposes of the FDOFs? Figure 97 shows stance frames from *Apatosaurus* forelimb walking gaits generated using an LROM space with a 4° angular sampling resolution and 50 boxes along the direction of travel. The figure shows that while fitness increased from 12.27 (x10^-3) to 97.37 (x10^-3), the individual joint angles remained largely unchanged.

Figure 97. Walking gait before (top) and after (bottom) refinement.

Iterative refinement of FDOFs can dramatically increase the fitness values of walking gaits, but does not significantly modify the functional purposes of the animal's joints. In the next section, the coefficients associated with the terms of the fitness function will be varied, and data collected on the resulting gait fitness and functional joint purposes will be presented.

Fitness Function

The bipedal and quadrupedal fitness functions aggregate several error terms into a single error term. The error terms are: sum change in root pitch from neutral pose, sum change in root yaw from neutral pose, sum change in root roll from neutral pose, sum root lateral displacement from the initial sagittal plane, sum root vertical displacement from the target root height, and sum change in FDOF values. These values are computed at the RD and RLU keyframes only, allowing an extremely inexpensive fitness calculation (i.e., constant time with respect to the number of FDOFs). The final fitness is $1.0 / (1.0 + e)$, where e is the sum error term.

Each fitness error term has an associated coefficient that determines the term's contribution to the final fitness value. The coefficients were initially determined by observing walking gaits generated for the generic dog and reptile models. Initially, both models had a tendency to pitch and roll, so the pitch and roll error coefficients were increased. All other coefficients have unit value as of the time of this writing. The same error coefficients (and therefore the same fitness function) are used across all models for the generation of walking gaits.

Data was collected to determine the effect of varying the error coefficients on walking gait fitness. Table 11 shows the effect of modifying the pitch error coefficient on *Apatosaurus* forelimb gait fitness. The pitch fitness term generally discourages the body from pitching (i.e., rotation of the root about the lateral axis) during locomotion. Lowering the pitch error coefficient does indeed increase the resulting pitch error. The overall fitness remains generally unchanged, as several other error terms show reduced

error values (e.g., yaw , roll, and FDOF error). Conversely, raising the pitch error coefficient decreases the resulting pitch error.

Table 11.

Effect of varying pitch fitness coefficient on fitness error terms

	Fitness term						
Error	Pitch	Yaw	Roll	Lateral	Vertical	FDOF	Total
				Half pitch coefficient value			
M	6.84	4.75	7.16	12.79	13.88	26.98	72.39
SD	2.92	2.83	2.69	5.25	8.26	5.96	10.06
				Base pitch coefficient value			
M	3.89	5.94	10.54	6.49	11.93	32.69	71.47
SD	2.17	3.11	8.35	4.76	5.53	9.93	11.55
				Double pitch coefficient value			
M	3.36	5.48	12.49	15.80	9.88	31.24	78.26
SD	2.39	3.48	6.48	12.04	4.85	12.93	19.01

Qualitative observations were also made to determine whether or not the error coefficients have a significant effect on the functional purpose of joints with respect to locomotion. Figure 98 shows frames from a dog forelimb walk. The dog forelimb is particularly vulnerable to pitching due to the large number of redundant FDOFs (i.e., FDOFs with near-coincident instantaneous axes or rotation). The figure shows frames from a walk generated using one tenth the base pitch coefficient and from a walk generated using ten times the base pitch coefficient.

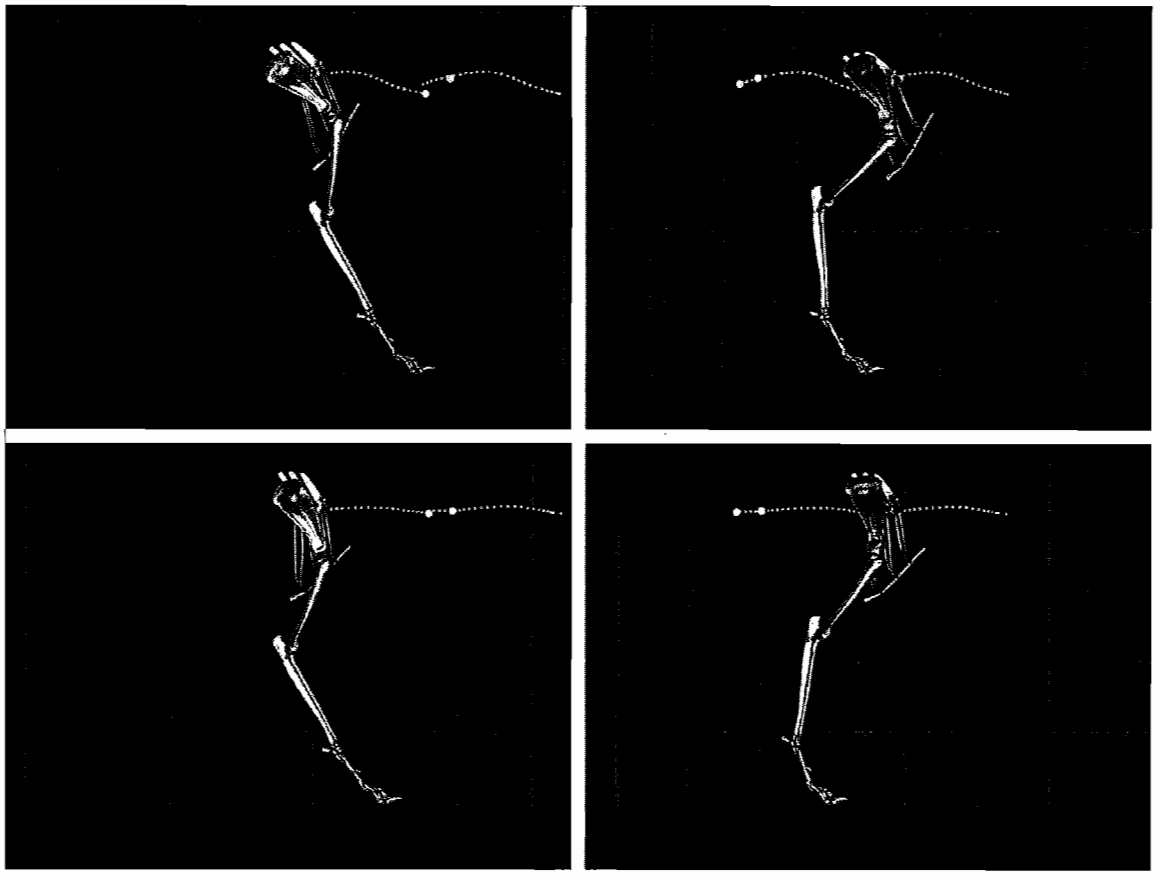


Figure 98. Comparison of low (top) and high (bottom) pitch error coefficient.

Figure 98 shows that the walk generated with a relatively-low pitch coefficient (top) is pitched back substantially more than the walk generated with a relatively-high pitch coefficient (bottom). Despite the two-orders-of-magnitude change in pitch error coefficient, the functional purposes of the limb's joints are the same in both gaits (e.g., humeral retraction; knee, wrist, and manus flexion).

Table 12 shows the effect of varying the yaw error on gait fitness. The yaw error term generally discourages body yawing (i.e., rotation of the root around the vertical axis)

during locomotion. Yaw error is increased with a reduced yaw error coefficient; yaw error is reduced with an increased yaw error coefficient. As with modifications to the pitch error coefficient, sum fitness remains largely unchanged as the difference in yaw error is distributed to the other error terms.

Table 12.

Effect of varying yaw fitness coefficient on fitness error terms

	Fitness term						
Error	Pitch	Yaw	Roll	Lateral	Vertical	FDOF	Total
				Half yaw coefficient value			
M	3.47	15.45	9.21	8.14	12.88	30.52	79.66
SD	2.23	14.85	3.64	3.68	11.43	10.12	26.21
				Base yaw coefficient value			
M	3.89	5.94	10.54	6.49	11.93	32.69	71.47
SD	2.17	3.11	8.35	4.76	5.53	9.93	11.55
				Double yaw coefficient value			
M	4.81	4.26	8.53	13.76	7.70	32.19	71.25
SD	4.77	2.75	4.50	9.33	4.13	12.71	10.23

The roll error term generally discourages rolling of the body (i.e., rotation of the root about the longitudinal axis) during locomotion. Table 13 shows the effect of varying the roll error coefficient on gait fitness. Lowering the yaw error coefficient increases yawing, raising the yaw error coefficient decreases yawing, as expected.

Table 13.

Effect of varying roll fitness coefficient on fitness error terms

	Fitness term						
Error	Pitch	Yaw	Roll	Lateral	Vertical	FDOF	Total
				Half roll coefficient value			
M	5.26	6.74	14.25	13.26	12.07	26.52	78.10
SD	2.60	4.87	10.75	6.18	8.92	7.10	19.15
				Base roll coefficient value			
M	3.89	5.94	10.54	6.49	11.93	32.69	71.47
SD	2.17	3.11	8.35	4.76	5.53	9.93	11.55
				Double roll coefficient value			
M	6.47	4.02	9.30	14.54	12.18	22.60	69.11
SD	2.62	2.28	2.93	5.38	8.36	8.11	10.49

The lateral error term generally discourages the body from lateral swaying (i.e., translation along the lateral axis) during locomotion. Table 14 shows the effect of varying the lateral error coefficient on gait fitness. Lowering the lateral error coefficient increases the lateral swaying of generated gaits. Raising the lateral coefficient, however, did not have a significant effect on the lateral error. The lack of change in lateral error is most likely because the lateral error is near its minimum possible value based on the sampling resolution of the LROM space (which is supported by the other tables in this section).

Table 14.

Effect of varying lateral fitness coefficient on fitness error terms

	Fitness term						
Error	Pitch	Yaw	Roll	Lateral	Vertical	FDOF	Total
				Half lateral coefficient value			
M	5.61	4.48	9.94	13.69	11.14	23.55	68.41
SD	2.98	3.59	6.24	5.90	8.59	7.45	15.22
				Base lateral coefficient value			
M	3.89	5.94	10.54	6.49	11.93	32.69	71.47
SD	2.17	3.11	8.35	4.76	5.53	9.93	11.55
				Double lateral coefficient value			
M	4.80	6.94	9.05	6.92	14.87	31.86	74.44
SD	3.20	4.38	5.81	2.29	10.60	8.61	12.33

Table 15 shows the effect of varying the vertical error coefficient on gait fitness. The vertical fitness term generally discourages vertical "bobbing" of the body (i.e., translation along the vertical axis) during locomotion. Lowering the coefficient increases vertical error; raising the coefficient reduces vertical error. The total error remains large unchanged.

Table 15.

Effect of varying vertical fitness coefficient on fitness error terms

	Fitness term						
Error	Pitch	Yaw	Roll	Lateral	Vertical	FDOF	Total
				Half vertical coefficient value			
M	5.36	5.45	8.28	9.07	15.90	28.76	72.82
SD	2.97	4.91	4.46	6.85	7.25	14.63	10.80
				Base vertical coefficient value			
M	3.89	5.94	10.54	6.49	11.93	32.69	71.47
SD	2.17	3.11	8.35	4.76	5.53	9.93	11.55
				Double vertical coefficient value			
M	5.76	4.91	8.57	11.90	4.44	28.10	63.68
SD	3.15	4.38	2.55	3.94	2.15	21.42	18.49

The FDOF error terms discourages the limb from undergoing large angular joint excursions during locomotion. The FDOF error term is particularly effective when dealing with limbs with many redundant (i.e., near-coincident instantaneous axes of rotation) FDOFs, such as the dog and *Tyrannosaurus* limbs. Without this term, the GA might select a gait with a flexed ankle joint and extended pes joint for one keyframe and an extended ankle joint and flexed pes joint for the next keyframe. Qualitatively, such a gait looks unrealistic (and painful) because of the large joint angular accelerations joints relative to the movements of the rest of the body.

Figure 99 shows a comparison between a *Tyrannosaurus* walking gait created using an FDOF error coefficient of 0.01 (top) and a gait created using a unit FDOF error coefficient (bottom). When using a very small FDOF error coefficient, the GA selects a gait that uses a large amount of ankle and knee flexion and pes joint extension to keep the body smoothly moving forward. Such large angular excursions during a small portion of

the total stance phase looks unnatural. When using a unit FDOF error coefficient, the gait is similarly smooth but uses much less angular excursion during early stance.

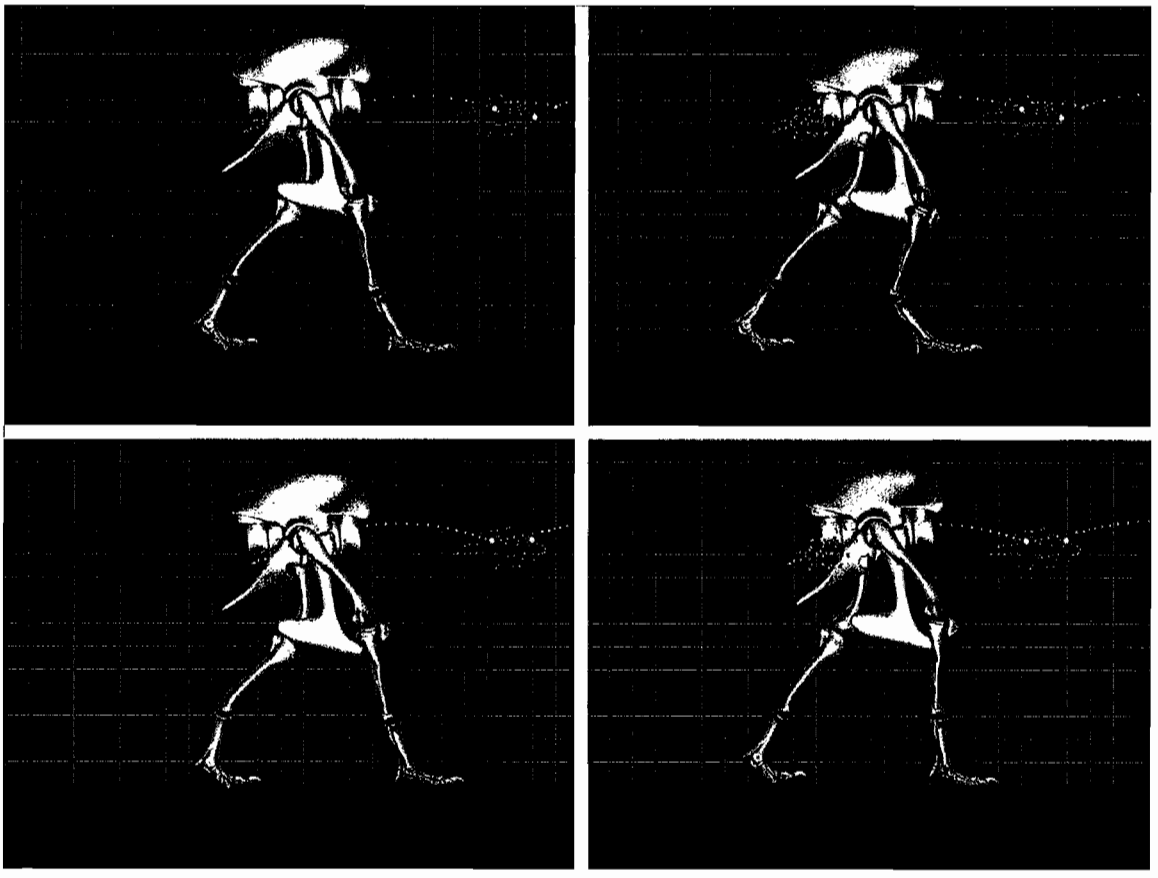


Figure 99. Comparison of low (top) and high (bottom) FDOF error coefficient.

In principal, there are many approaches that could approximate the FDOF error term. In a dynamics simulation that keeps track of physical properties such as mass, velocity, and acceleration, minimizing either angular velocity or accelerations would

likely achieve the same goals. Raibert and Hogins (1991) and Chung and Hahn (1999) utilized angular accelerations for similar purposes. In the absence of physics, minimizing angular excursion would achieve a similar goal. FDOF indices are normalized, however, so joints with small angular excursions are not favored over joints with larger angular excursions.

Table 16 shows the effect of varying the FDOF error coefficient on gait fitness. Not surprisingly, FDOF error increases when the FDOF error coefficient is reduced. Likewise, FDOF error decreases when the FDOF error coefficient is increased. Again, the sum error remained largely unchanged.

Table 16.

Effect of varying FDOF fitness coefficient on fitness error terms

	Fitness term						
Error	Pitch	Yaw	Roll	Lateral	Vertical	FDOF	Total
				Half FDOF coefficient value			
M	2.86	6.64	8.18	7.65	8.12	39.62	73.06
SD	1.15	2.60	4.73	4.51	3.81	19.54	17.30
				Base FDOF coefficient value			
M	3.89	5.94	10.54	6.49	11.93	32.69	71.47
SD	2.17	3.11	8.35	4.76	5.53	9.93	11.55
				Double FDOF coefficient value			
M	6.58	4.73	9.59	11.30	17.36	21.10	70.65
SD	2.72	2.79	7.09	5.28	11.36	7.97	13.17

The data presented in this section shows that the changes in the fitness function coefficients affect the generated gaits in intuitive ways (e.g., increasing the penalty

associated with body pitching results in walks with generally less pitching). However, the overall fitness values remain consistent; difference in error is distributed among the other error terms. In the next section, generated walking gaits will be qualitatively compared to extant animal data to determine how closely the generated gaits match real-world analogues.

Qualitative Gait Analysis

In this section, walking gaits automatically generated using GAGA methods will be compared to published examples of real-world animal gaits. First, the generic dog and reptile gaits will be compared against analogous examples of observed real-world locomotion. Next, walking gaits generated for the *Apatosaurus*, *Triceratops*, and *Tyrannosaurus* models will be qualitatively compared to locomotion characteristics of the dog and reptile gaits. Finally, observations on the functional locomotion purposes of specific joints will be discussed.

Dog

Goslow (1981) analyzed the forelimb and hindlimb behaviors of dogs during locomotion. High-frame-rate movies were first taken of walking dogs. The film was then projected onto a glass plane, onto which celluloid images of the major limb bones were placed to simulate the internal osteological structure of each limb at several keyframes. Figure 100 shows a comparison of a hindlimb walking gait generated using

Goslow (1981) methods (bottom) with a hindlimb walking gait generated using quadrupedal GAGA methods (top).

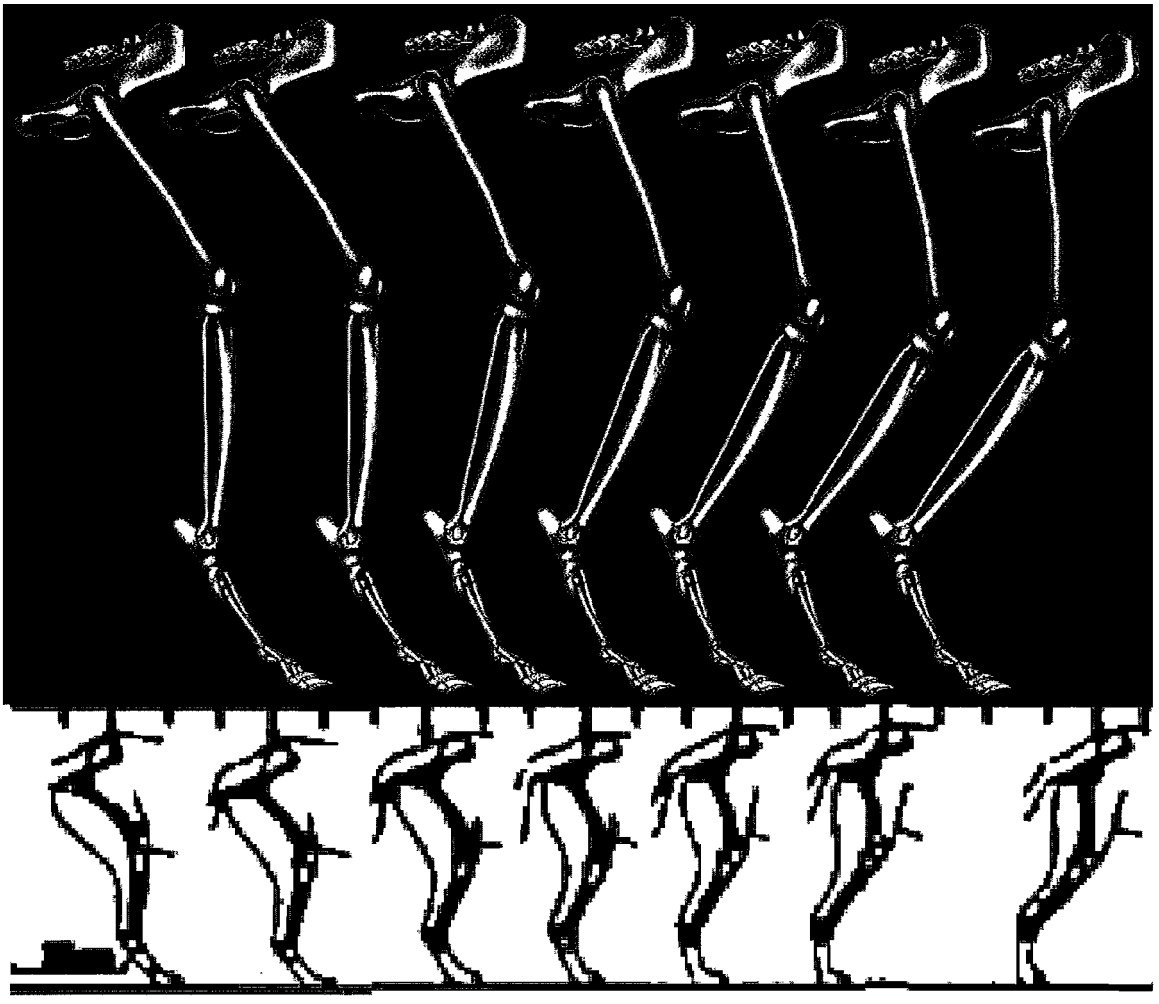


Figure 100. Comparison of GAGA (top) and Goslow (bottom) hind dog walk.

Note. Goslow images (bottom) from "Electrical activity and relative length changes of dog limb muscles as a function of speed and gait" by G. E. Goslow, 1981, *Journal of Experimental Biology, 94*(1), p. 32.

In both walking gaits, the femur begins stance (left) with the femur protracted and the crus almost vertical. Throughout stance, the femur retracts and the knee extends slightly. At the end of stance (right), the femur is nearly vertical in both gaits. During stance, the ankle and pes joints flex together, causing the ankle height to increase in the Goslow model. In the GAGA model, the ankle flexes more than the pes joint, causes less increase in ankle height. In summary, the functional movements are nearly identical between the two gaits. The difference in pes/ankle joint functionality appears to be an issue that cannot be solved with pure kinematics. Figure 101 shows a comparison of a forelimb gait generated using Goslow (1981) methods (bottom) with a forelimb gait generated using quadrupedal GAGA methods (top).

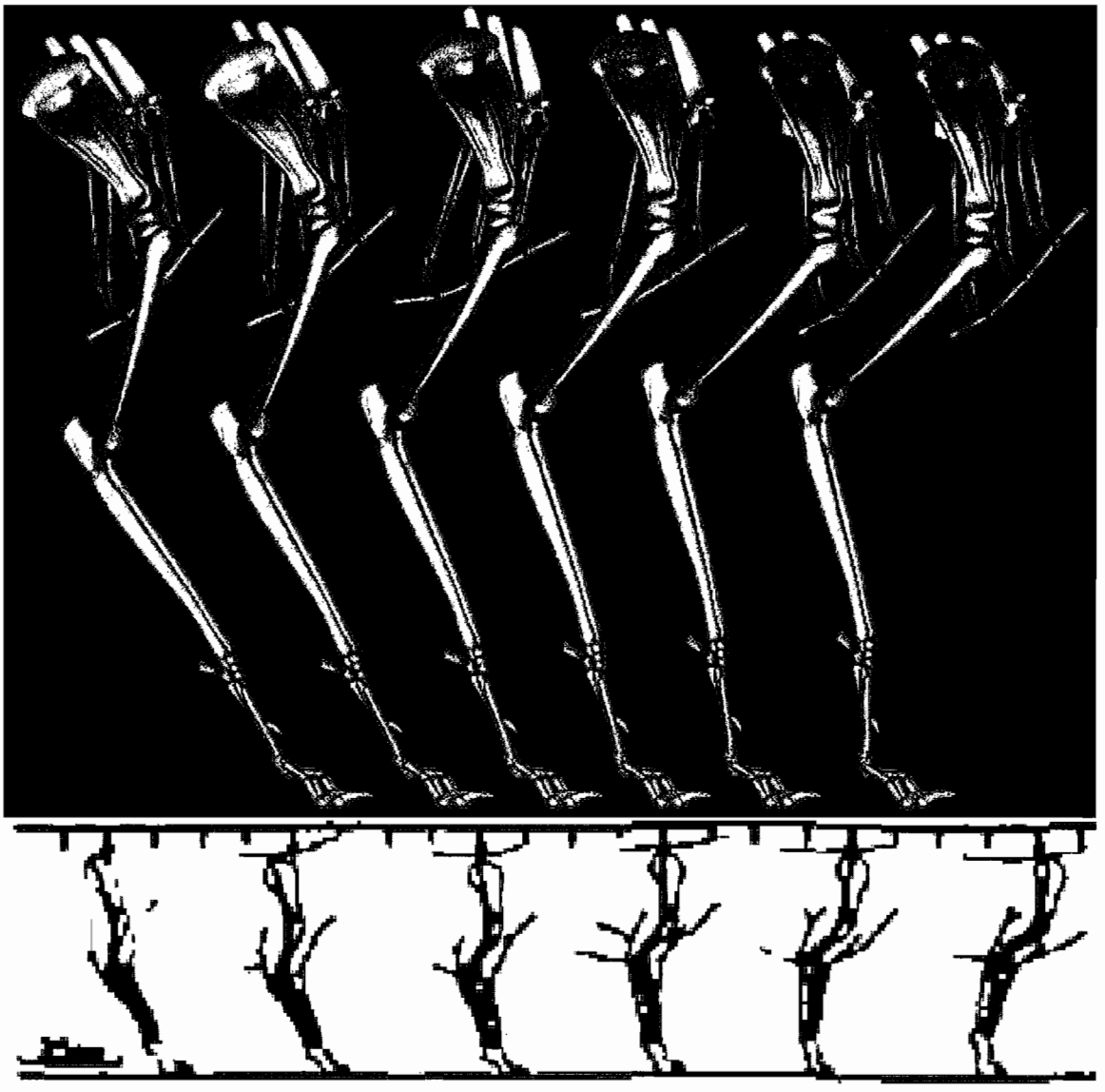


Figure 101. Comparison of GAGA (top) and Goslow (bottom) fore dog walk.

Note. Goslow images (bottom) from "Electrical activity and relative length changes of dog limb muscles as a function of speed and gait" by G. E. Goslow, 1981, *Journal of Experimental Biology, 94*(1), p. 19.

In both gaits, stance begins (left) with the scapula pointing forward, the humerus pointing slightly back, and the antibrachium outstretched and pointing forward. At the end of stance (right), the scapula is nearly vertical, the shoulder and elbow joints have counterrotated, and the manus joint has flexed. The joints appear to have the same functional behaviors between the two gaits. The only noticeable difference between the two gaits is that the GAGA stance phase starts with the limb farther forward and less outstretched than the Goslow model, causing the Goslow stance phase to end with the limb farther back than with the GAGA model. This longitudinal shift could be caused by any number of reasons, such as slightly-different limb proportions between the models or the combination of associated hindlimb and trunk behaviors (not shown).

Muybridge (1887) is responsible for the world's first careful analyses of animal locomotion. In the late 1800s, Muybridge used linear arrays of cameras to capture animal locomotion in a sequence of photographs. Today, Muybridge's animal studies, which are substantial in breadth, are still relevant for their historical and academic significance. Figure 102 shows a comparison of a Muybridge (1887) sequence of dog locomotion photographs with a quadrupedal walking gait generated using GAGA methods.

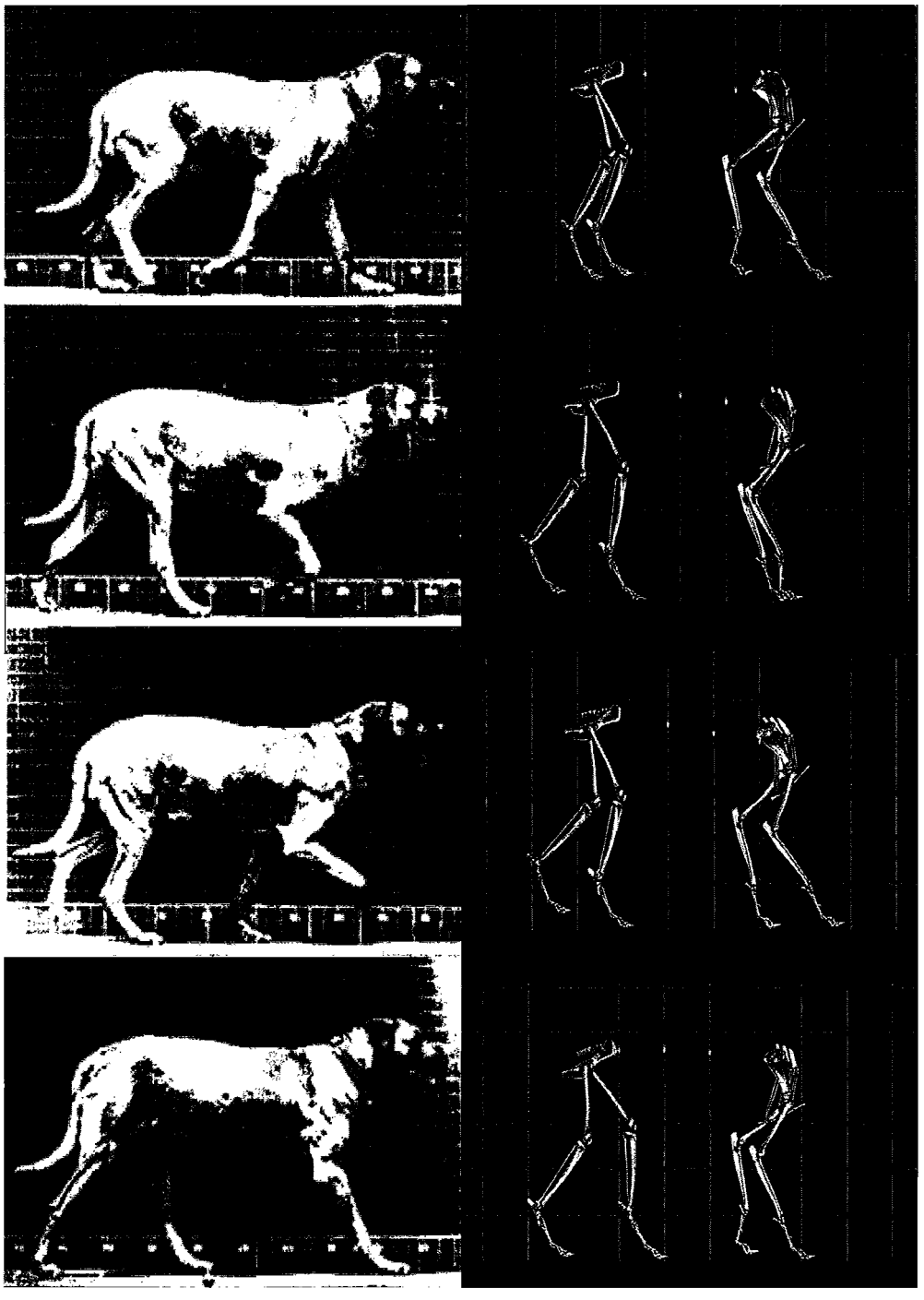


Figure 102. Comparison of Muybridge (left) and GAGA (right) dog walk.

Note. Muybridge images (left) from "*Animals in motion [1957 reprint]*" by E. Muybridge, 1887, New York: Dover, p. 115.

The comparison confirms that the ipsilateral phase and the general limb movements of the GAGA gait are correct with respect to the Muybridge gait. The most noticeable difference between the Muybridge and GAGA gaits is the lack of wrist flexion during the swing phase. Recall that the swing phase is created using a simple IK method because the stance phases are the focus of these studies. The wrist FDOF could be extended to allow sufficient flexion for the swing phase, but such an increase in angular excursion would dramatically increase the total size of the forelimb LROM space (see Figure 57).

The quadrupedal dog walking gait generated using GAGA methods has been shown to be qualitatively similar to published dog walking gait data from two studies. This analysis shows that, at least with a generic dog model, the GAGA methods are able to generate a walking gait that is similar to real-world gaits in terms of joint functionality. In the next section, walking gaits generated for the generic reptile model will be compared to observed real-world walking gaits.

Reptile

Unfortunately, careful gait studies are not available for the Komodo dragon, the specific reptile on which the generic reptile model is based. To determine how well the gross limb and body movements of a quadrupedal reptile gait generated using GAGA methods compares to the gait of a real-world Komodo dragon, the GAGA gait was compared to published video clips of Komodo dragon walking gaits (Fotosearch, 2007).

Figure 103 shows a comparison between frames of a walking gait generated using GAGA methods to frames of a real Komodo dragon walking.

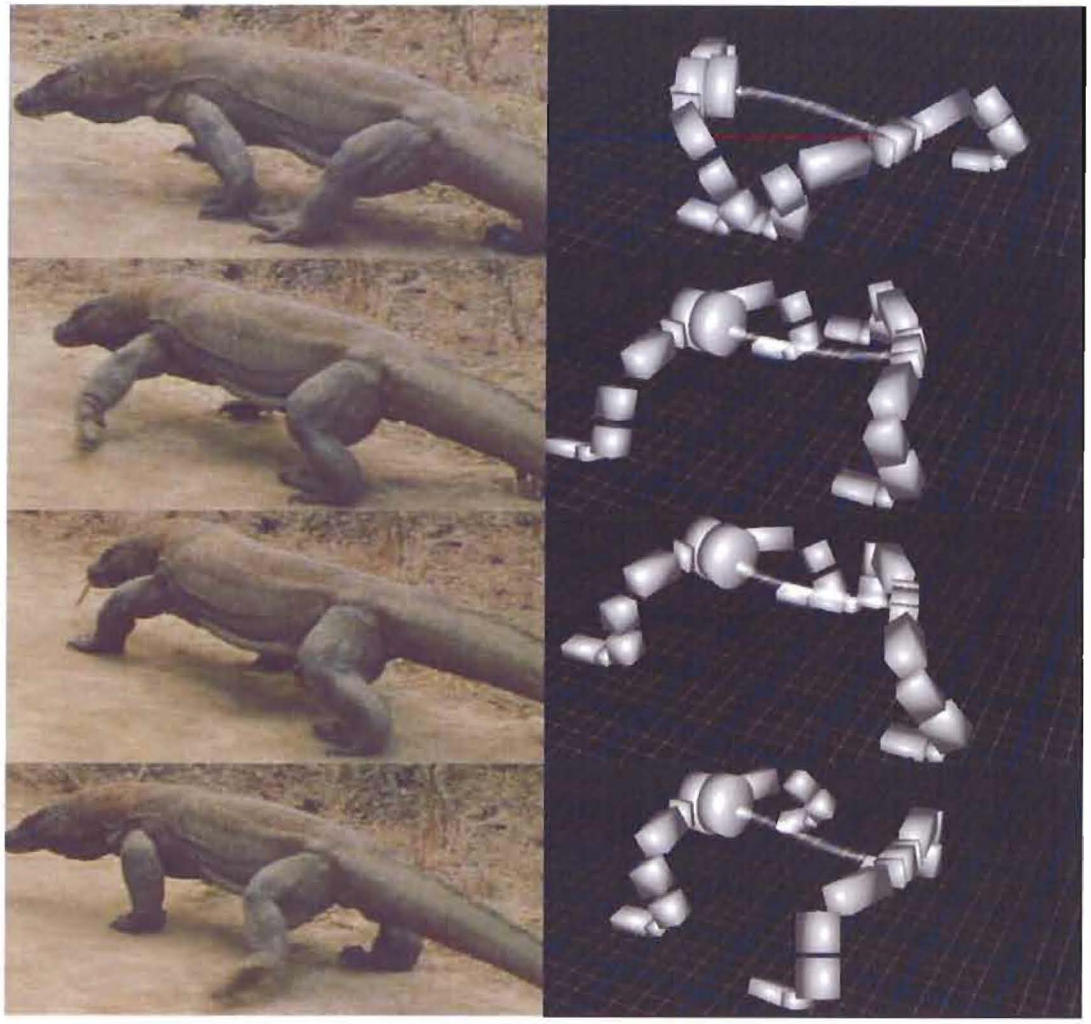


Figure 103. Comparison of Komodo dragon (left) and GAGA reptile (right) walk.

Note. Fotosearch images (left) from "Komodo dragon walking on ground" by Fotosearch, 2007, retrieved November 21, 2007, from http://www.fotosearch.com/DVA002/006-0044.

Both gaits operate with an ipsilateral phase of 0.5; diagonal limbs start and end stance at the same time. Both pelvic and pectoral assemblies rotate towards the stance limb during the limb's stance phase, which propels the body forward on an arc centered at the manus/pes position. This pelvic/pectoral rotation also helps the swing limbs to travel forward faster and to cover a greater distance before the limb begins stance, allowing longer stride lengths. The opposite rotations of the pelvic and pectoral assemblies cause the trunk to bend in a sinuous manner. Reptilian back bending is discussed by Reilly (1997, 1998). Figure 104 shows a comparison of reptile back bending between frames from the reptile walking gait generated using GAGA and published images (Reilly, 1998).

Figure 104. Comparison of GAGA (top) and Reilly (bottom) reptile gait images.

Note. Reilly images (bottom) from "Sprawling locomotion in the lizard *Sceloporus clarkii*: quantitative kinematics of a walking trot" by S. M. Reilly, 1997, *Journal of Experimental Biology, 200*(4), p. 756.

The reptile limb movements are somewhat unintuitive, mostly because the knee and elbow axes are nearly parallel to the sagittal plane at mid stance. Therefore, flexion/extension of the knee/elbow at mid stance causes the crus/antibrachium to rotate mediolaterally with essentially no protraction/retraction. In order for the knee and elbow to be used functionally during locomotion, inner and outer rotation at the hip and shoulder must be used to rotate the femur/humerus and reorient the knee/elbow axes so that the crus and antibrachium can protract and retract.

Reorientation of the knee and elbow axes causes mediolateral movement of the crus and antibrachium that must be absorbed by mediolateral rotation at the pes and manus to avoid twisting and or slipping of the pes/manus on the ground. In addition, protraction/retraction of the femur and humerus must be absorbed by pronation/supination in the crus and antibrachium to avoid twisting or slipping of the pes/manus on the ground. Figure 105 shows frames from a reptile hindlimb walking gait generated using GAGA methods.

Figure 105. Reptile hindlimb stance phase.

Rotation of the femur is particularly obvious at the end of stance, when the knee is extended to allow a long stride. Flexion at the ankle and pes joints also helps to increase the stride. The pronation and supination of the antibrachium can also be seen clearly. In the absence of crus pronation/supination, the twisting of the limb caused by hip flexion/extension would need to be absorbed somewhere else in the limb; otherwise the twist would be transmitted through the pes, causing the pes to rotate on the ground about

the vertical axes. Figure 106 shows an analogous reptile forelimb walking gait generated using GAGA methods.

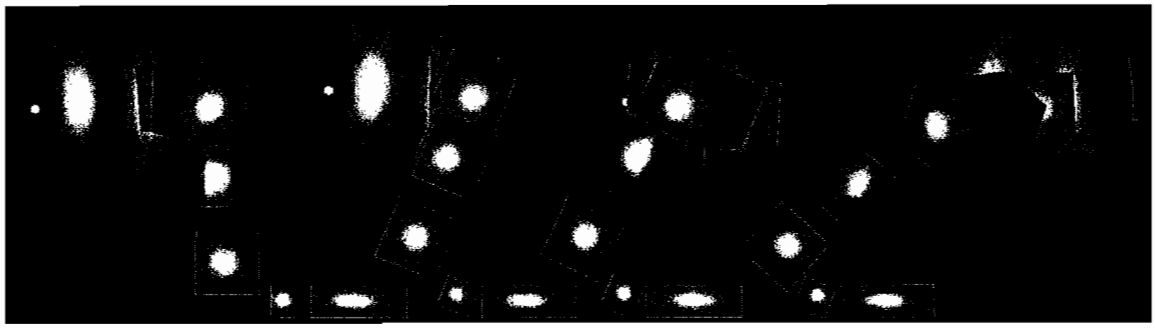


Figure 106. Reptile forelimb stance phase.

Like the hindlimb, the forelimb also uses the rotated elbow axis, along with wrist and manus joint flexion/extension to achieve long stride lengths. As with the hindlimb, limb twist caused by shoulder flexion/extension is absorbed by pronation/supination in the antibrachium.

The quadrupedal reptile walking gait generated using GAGA methods has been shown to be qualitatively similar to published examples of reptile locomotion. In particular, the generated gait exhibits sinuous back bending that is seen in real-world reptiles. In addition, functional joint behavior is consistent with that observed in reptiles. GAGA methods have now been shows to recreate qualitatively-accurate walking gaits using a single fitness function. The differences in the generated gaits between the dog

and reptile models can therefore be attributed to the differences in limb geometries between the models. This observation is supported by Pike and Alexander (2002), who showed that animals with similar limb proportions and stances have similar gaits. Conversely, animals with differing limb proportions and stances displayed differences in limb joint behaviors. In the next section, the GAGA methods will be used to generate quadrupedal walking gaits for the *Apatosaurus* sauropod dinosaur model.

Apatosaurus

A quadrupedal *Apatosaurus* walking gait was automatically generated using the same GAGA methods and fitness function used to generate the dog and reptile gaits. Figure 107 shows hindlimb stance frames from the *Apatosaurus* walking gait. The *Apatosaurus* hindlimb has one less FDOF than the dog hindlimb; there is no flexion/extension in the pes. The hip, knee, and ankle flexion/extension FDOFs have nearly-parallel axes of rotation, like the dog's FDOFs. These three FDOFs function in the same way that they do in the dog: retraction at the hip coupled with a slight extension of the knee, both of which are counterrotated by flexion of the ankle to keep the pes in place on the ground.

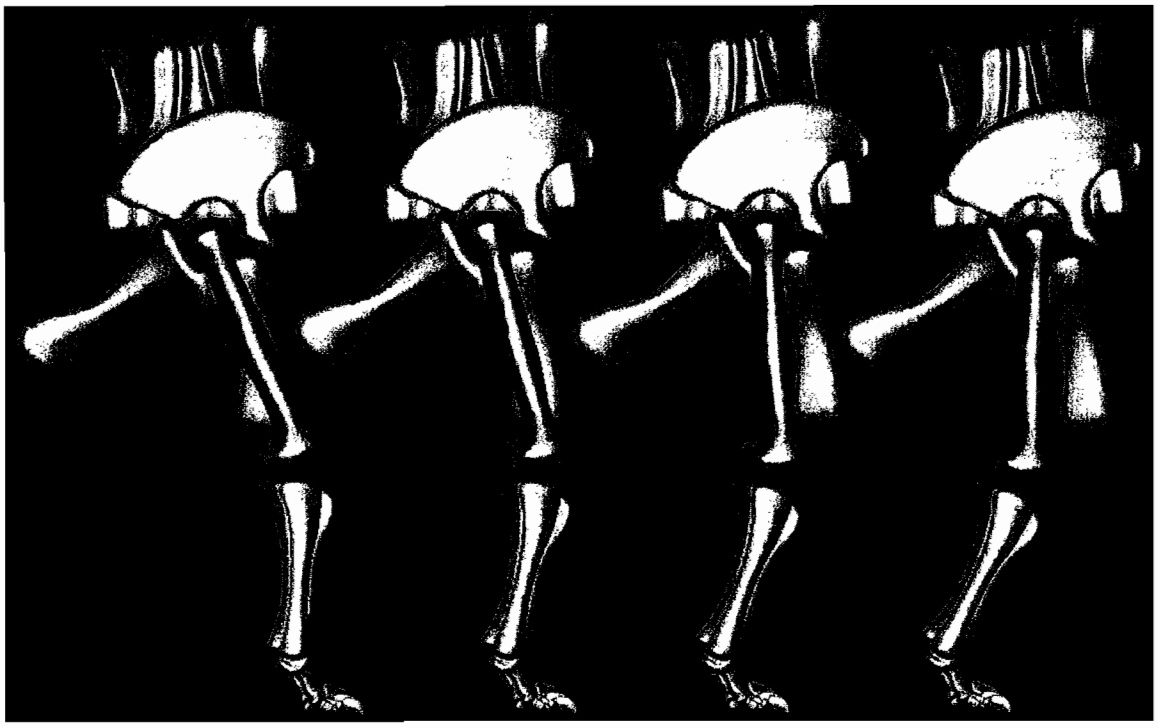


Figure 107. Apatosaurus hindlimb stance phase.

Like the *Apatosaurus* hindlimb, the forelimb has no manus flexion/extension FDOF. The forelimb has one additional FDOF at the scapulothorax joint. The dog model has one FDOF that describes scapulothorax movement because scapula movement is well understood for dogs. Movements of scapulae in dinosaurs, however, are not well understood due to the lack of osteological features indicating their possible positions and orientations. The *Apatosaurus* forelimb is therefore given two FDOFs (i.e., rotation and elevation) so that the GA can find suitable scapula positions and orientations.

Figure 108 shows forelimb stance frames from the *Apatosaurus* walking gait. Similar to the dog forelimb, the shoulder and elbow joints counterrotate, with the wrist

joint flexing to keep the manus planted on the ground. The scapulothorax joint is used functionally in the same way that it is used in the dog. In the dog, the scapula is used to retract the limb during stance; the *Apatosaurus* scapula movement is not as pronounced, but scapulothorax does help to retract the limb during stance.

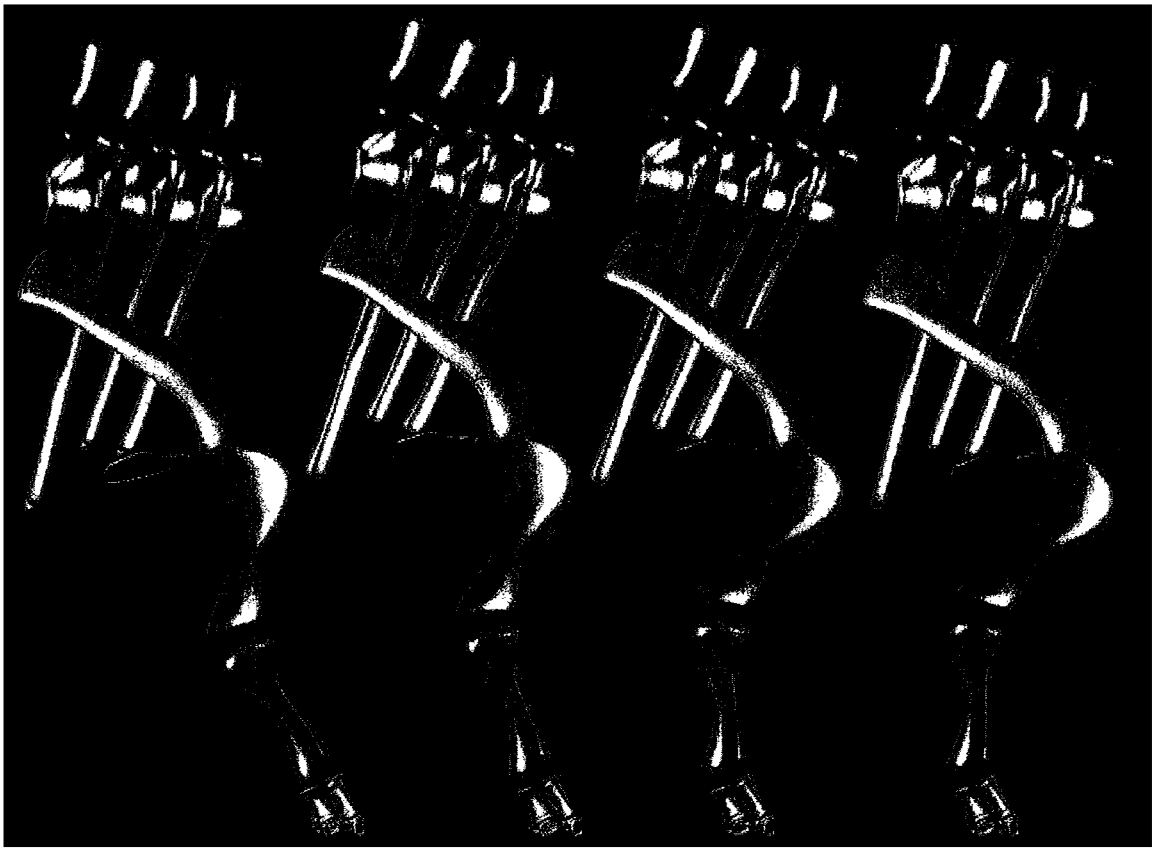


Figure 108. Apatosaurus forelimb stance phase.

The *Apatosaurus* forelimb has relatively-large diversity in the instantaneous axes of rotation of the scapulothorax rotation and shoulder, elbow, and ankle flexion/extension FDOFs. The differing axes of rotation for these FDOFs creates a robust LROM space (see Figure 71), as opposed to the dog's highly parasagittal forelimb LROM space (see Figure 57). The smoothest path through the *Apatosaurus* forelimb space does not utilize much scapula movement, indicating that the scapulothorax joint many not have been used much during locomotion. A quantitative analysis of the interactions between the *Apatosaurus* scapulothorax and shoulder joints will be presented in a later section. In the next section, the GAGA methods will be used to generate a walking gait for the *Triceratops* ceratopsid dinosaur model.

Triceratops

A quadrupedal *Triceratops* walking gait was automatically generated using the same fitness function used to generate the dog and reptile gaits. Figure 109 shows hindlimb stance frames from the *Triceratops* walking gait. Again, similar to the dog hindlimb, the hip retracts, the knee extends slightly, and the ankle flexes to keep the pes in contact with the ground. The knee and ankle FDOF axes are nearly-coincident, which allows a mostly-parasagittal LROM space (see Figure 74). The shoulder FDOF axis, however, differs significantly from the knee and ankle FDOF axes. It is therefore interesting that the *Triceratops* displays mammalian hindlimb joint behaviors. The difference in FDOF axes does cause some body rolling towards the stance limb. This body rolling could functionally aid in helping the swing leg clear the ground.

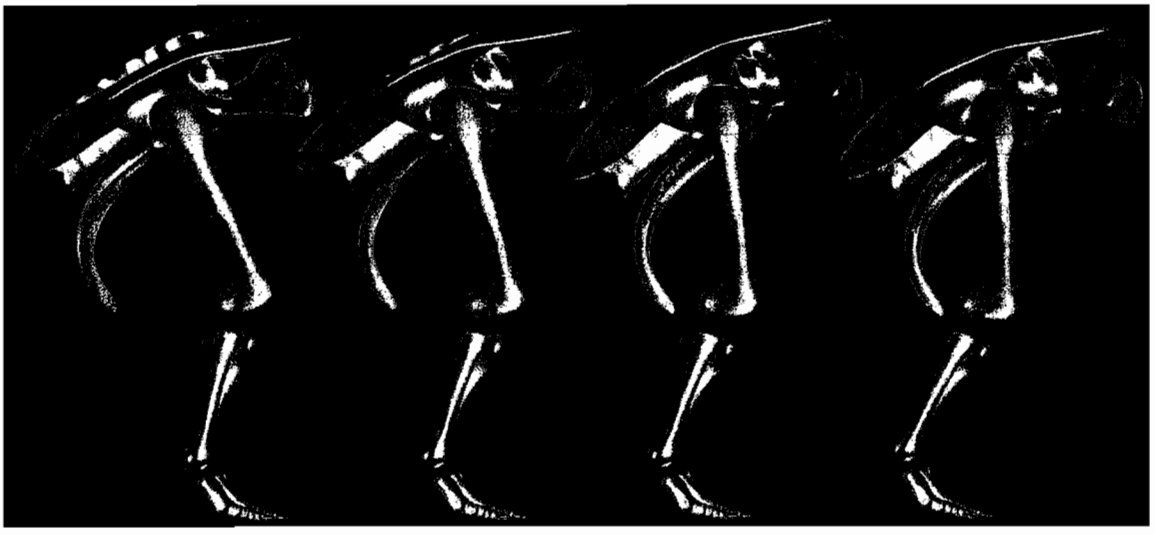


Figure 109. Triceratops hindlimb stance phase.

Figure 110 shows forelimb stance frames from the generated *Triceratops* walking gait. Like the dog forelimb, again, the shoulder retracts while the elbow counterrotates. The wrist flexes to keep the manus in contact with the ground. Like the dog but unlike the *Apatosaurus*, the scapulothorax is used to further retract the limb. Like the *Apatosaurus* forelimb LROM space, the *Triceratops* LROM space is quite robust (see Figure 78) due to the diverse instantaneous axes of rotation of the limb's FDOFs. In the case of the *Triceratops* forelimb, the smoothest path through the space utilizes the scapulothorax joint, indicating that is was probably used during locomotion.

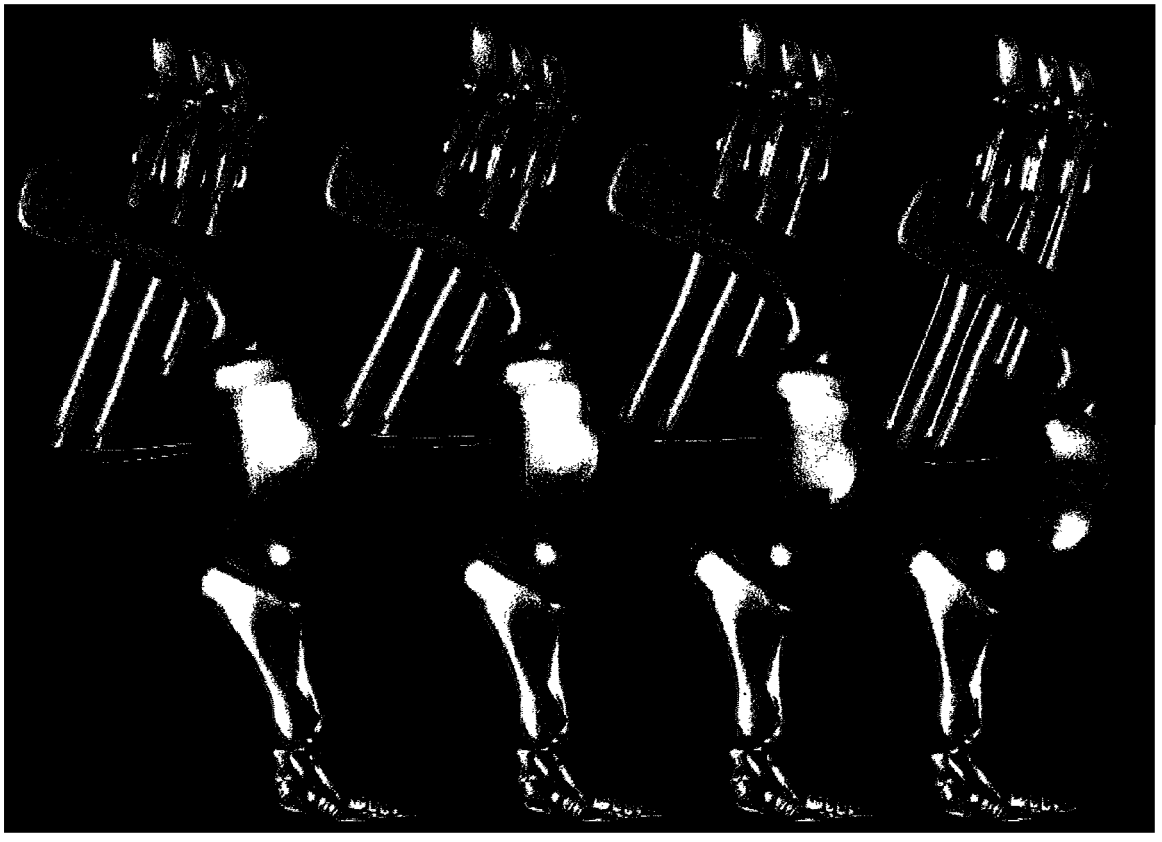


Figure 110. Triceratops forelimb stance phase.

Quadrupedal walking gaits have been generated for the *Apatosaurus* and *Triceratops* dinosaur models using the same GAGA methods and fitness function used to automatically generate walking gaits for the generic dog and reptile models. The GAGA models and fitness function have been shown to create walking gaits with functional joint behaviors that approximate the functional joint behavior of real animals, so it is reasonable to assume that the functional joint behaviors observed in the dinosaur walking

gaits are also realistic. In the next section, a walking gait will be generated for the *Tyrannosaurus* theropod dinosaur model.

Tyrannosaurus

A bipedal *Tyrannosaurus* walking gait was automatically generated using the same fitness function used to generate the dog and reptile gaits. The *Tyrannosaurus* limb has one extra FDOF; the *Tyrannosaurus* pes has a phalangeal flexion/extension FDOF that no other model has. The phalangeal joint FDOF is redundant (i.e., has a near-collinear instantaneous axis of rotation) with the hip, knee, ankle, and pes flexion/extension joints. Such a large number (i.e., five) of redundant FDOFs would be a nightmare for any IK system due to the massive number of near-identical configurations at mid stance.

At the beginning and end of stance, however, there are relatively-few ways that the limb can stretch out for long strides. Selection of keyframes at or near these events greatly reduces the number of possible limb configurations (i.e., the space is pruned from 2,882,880 samples to 24,346, a reduction by more than two orders of magnitude). The fitness function's FDOF error term helps to ensure that problematic sets of keyframes (i.e., similar root positions and orientations with dramatically-different internal joint angles) are not selected. Selection of such problematic sets of keyframes has been observed when using a very low FDOF error coefficient in the fitness function. Figure 111 shows hindlimb stance frames from the *Tyrannosaurus* walking gait.

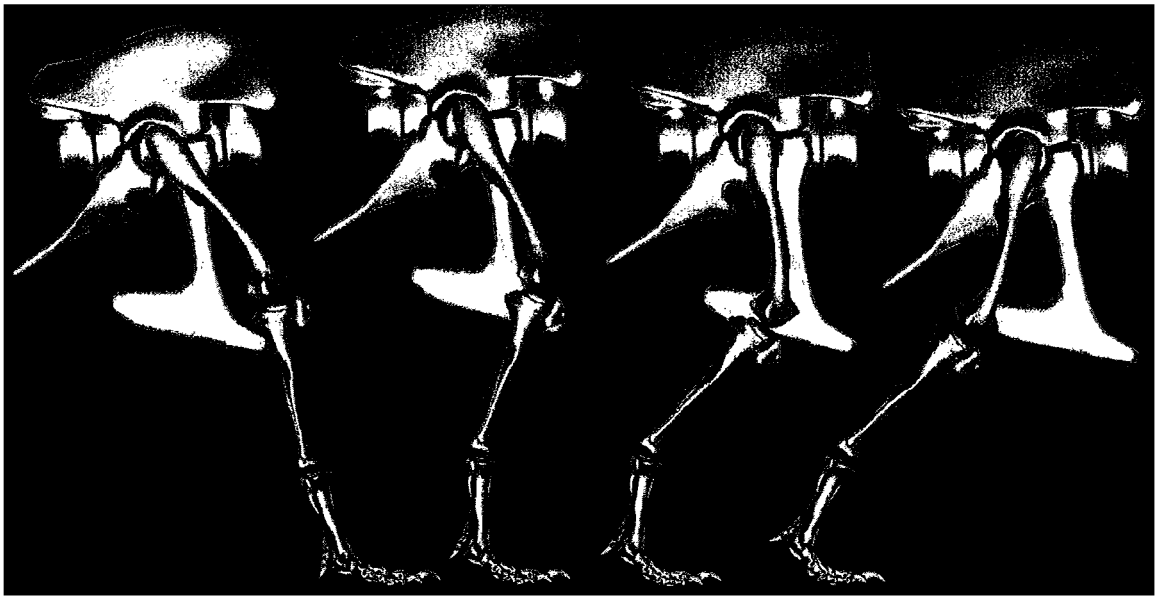


Figure 111. Tyrannosaurus hindlimb stance phase.

Again like the dog hindlimb, the *Tyrannosaurus* hip retracts the limb while the knee extends slightly. A combination of ankle, pes, and phalangeal joint rotation keep the distal digits of the pes planted on the ground. The GA is able to find a smooth and functional combination of movements for these joints by selecting keyframes only at and near the start and end of the stance phase and by utilizing the FDOF error term in the fitness function. In the next section, an overview of the functional purposes of joints with respect to locomotion will be presented.

Joint Functionality

Across the five models, hindlimb and forelimb joints seemed to be utilized for the same functional purposes. The five models provide diversity in body plans and limb proportions, from mammalian and lizard extremes to dinosaurs that fall somewhere in between. The differences in body plans are especially apparently when comparing the LROM spaces of the various limbs. Even with this diversity, very few differences in the gross functionality of the joints were observed.

In hindlimbs, hips generally protract the limb. Knees generally extend slightly to allow the animal to move forward with minimal vertical bobbing. It is interesting to note that all of the gaits cause the hindlimb and forelimb root to travel on a slight vertical arc even though the fitness function tries to minimize vertical displacement. Keyframes are only selected at and near the beginning and end of stance, so this arc is caused by interpolation between the keyframes; adding a keyframe at mid stance could flatten out the vertical arc. This vertical arc is generally accepted as a characteristic for the walking gaits of both mammals and reptiles (Farley, 1997), so it is interesting the GA implicitly selects gaits that exhibit this behavior without an explicit fitness function term to encourage it.

The reptile hindlimb differs in that the knee axis of rotation is rotated almost 90° relative to the dog's knee axis of rotation. Inner/outer rotation of the hip is used to reorient the knee axis so that it can contribute to locomotion. Flexion/extension of the hip causes rotation of the limb about the world vertical axis which is cancelled out by pronation/supination within the crus, preventing the pes from twisting on the ground.

The ankle and pes elements vary from having one flexion/extension FDOFs (i.e., *Apatosaurus, Triceratops)* to having three flexion/extension FDOFs (i.e., *Tyrannosaurus*). Regardless of the number of FDOFs, the function of these distal elements is to flex in summary, counter rotating with the retraction at the hip. When possible, the pes elements also raise the height of the ankle, which aids the vertical smoothness of the walk and allows longer stride lengths. The model reptile model additionally utilizes medial/lateral rotation in the pes to absorb mediolateral movement of the crus.

In the forelimbs, the scapulothorax and shoulder joints act together to retract the limb. The elbow counterrotates along with the wrist and manus joints to move the animal along a smooth path while keeping the distal elements of the manus in contact with the ground. Like the reptile hindlimb, the reptile forelimb also differs from that of the other models. Inner/outer rotation at the shoulder is required to reorient the elbow axis for locomotion, and flexion/extension at the shoulder causes rotation of the limb which is cancelled out by pronation/supination within the antibrachium.

In this section, quadrupedal walking gaits were generated for extant animals and subsequently compared to published gait images. Joint functionality was nearly identical between simulated and published gaits, indicating that the GAGA methods and fitness function generate walking gaits that are closely related to real-world analogues. Gaits generated for dinosaurs using these methods, therefore provide valuable insight into the probably functional joint behaviors of these extinct animals. In the next section, specific

hypotheses will be presented that are related to joint and body functionally with respect to locomotion.

Quantitative Gait Analysis

In this section, quantitative data will be presented to support specific locomotion hypothesis. The data presented will represent variations in gait observables (i.e., body pitch, roll, yaw, lateral and vertical displacement, and angular excursion at FDOFs) sampled at a standard interval over complete gait cycles. This gait observable data differs from that presented earlier, which focused on gait fitness error. The fitness function does not evaluate complete gait cycles for performance reasons, so the fitness error terms (i.e., based on the configurations at limb events RD and RLU) are the only data available to the fitness function. Gait observable data collected over a complete gait cycle provides a more realistic representation of the generated walking gait and is therefore better suited for sensitive studies.

First, walking gaits will be generated without the use of some limb FDOFs to determine the functional purposes of those FDOFs with respect to locomotion. Next, ankle height will be varied for animals with little pes flexibility and pes FDOFs will be disabled for animals with greater pes flexibility to determine optimal ankle heights with respect to pes flexibility. Lastly, ipsilateral phase will be varied to evaluate the effect of this phase term on quadrupedal gait observables.

Scapulothorax/Shoulder Contributions

The scapulothorax and shoulder joints of the *Apatosaurus* and *Triceratops* models have a combined five FDOFs. The scapulothorax and shoulder joints of the dog models have a combined four FDOFs. Combinations of those FDOFs, mostly scapulothorax rotation and shoulder flexion/extension, are used to propel the animal forward. The dog model's forelimb LROM space is highly parasagittal, so abduction/adduction and medial/lateral rotation are likely not utilized much. The *Apatosaurus* and *Triceratops* models, however, have robust forelimb LROM spaces due to the diversity in FDOF instantaneous axes of rotation and therefore likely utilize combinations of all available FDOFs to propel the animal forward smoothly.

To determine the functional purposes of the scapulothorax and shoulder FDOFs, observable data was collected for gaits that were not allowed to utilize certain combinations of FDOFs. The collected data shows that disallowing FDOFs forces the gait to either make greater use of other FDOFs or suffer greater pitching, yawing, rolling, or lateral/vertical displacement during locomotion. Table 17 shows the effect of removing FDOFs on dog forelimb gait observables.

Table 17.

Effect of removing forelimb FDOFs on dog gait observables

	Observable							
	Body angle (°)			Body displacement (cm)		FDOF angular excursion (°)		
Value	Pitch	Yaw	Roll	Lateral	Vertical	Scapulothorax [a]	Shoulder 2 [b]	Shoulder 3 [c]
	All limb FDOFs							
M	1.10	0.19	0.17	0.08	2.80	22.42	0.01	0.00
SD	0.60	0.07	0.08	0.02	1.59	4.02	0.01	0.00
	Missing shoulder abduction/adduction FDOF							
M	0.71	0.24	0.28	0.11	1.30	21.49	0.00	0.00
SD	0.61	0.13	0.08	0.02	0.52	3.29	0.00	0.00
	Missing shoulder abduction/adduction and scapulothorax FDOFs							
M	1.41	0.55	0.04	0.12	0.86	0.00	0.00	0.00
SD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Missing shoulder medial/lateral rotation FDOF							
M	0.80	0.19	0.20	0.09	1.82	23.21	0.00	0.00
SD	0.62	0.05	0.08	0.03	1.26	1.97	0.00	0.00
	Missing shoulder medial/lateral rotation and scapulothorax FDOFs							
M	1.41	0.55	0.04	0.12	0.86	0.00	0.00	0.00
SD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Note. [a] Scapulothorax rotation. [b] Shoulder abduction/adduction. [c] Shoulder medial/lateral rotation.

As expected, shoulder abduction/adduction and medial/lateral rotation are almost entirely unused. Scapulothorax joint rotation can be removed at the cost of additional body pitching and yawing. Again, the lack of shoulder abduction/adduction and medial/lateral rotation and the general lack of body yawing, rolling, and lateral displacement can be attributed to the highly-parasagittal nature of the LROM space. Table 18 shows the effect of removing FDOFs on *Apatosaurus* forelimb gait observables. The *Apatosaurus* was not capable of a step length 0.34 times the hip height without all forelimb FDOFs, so a step length 0.31 times the hip height was used for this study.

Table 18.

Effect of removing forelimb FDOFs on Apatosaurus gait observables

	Observable							
	Body angle (°)			Body displacement (cm)		FDOF angular excursion (°)		
Value	Pitch	Yaw	Roll	Lateral	Vertical	Scapulothorax [a]	Shoulder 2 [b]	Shoulder 3 [c]
	All limb FDOFs							
M	1.44	1.46	1.73	3.71	6.52	8.78	12.19	7.98
SD	0.54	0.57	0.43	2.25	4.58	5.08	6.48	4.66
	Missing shoulder abduction/adduction FDOF							
M	1.92	1.53	2.63	4.07	3.97	23.36	0.00	0.60
SD	0.41	0.33	0.60	0.60	1.11	3.30	0.00	1.88
	Missing shoulder abduction/adduction and scapulothorax FDOFs							
M	2.22	1.54	3.32	10.70	10.37	0.00	0.00	0.00
SD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Missing shoulder medial/lateral rotation FDOF							
M	1.07	0.99	2.55	4.74	5.26	23.36	3.31	0.00
SD	0.28	0.01	1.08	2.35	3.50	5.31	5.33	0.00
	Missing shoulder medial/lateral rotation and scapulothorax FDOFs							
M	1.41	4.37	3.62	5.48	6.66	0.00	13.17	0.00
SD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Note. [a] Scapulothorax rotation. [b] Shoulder abduction/adduction. [c] Shoulder medial/lateral rotation.

Unlike the dog model gaits, the *Apatosaurus* walking gaits make extensive use of shoulder abduction/adduction and medial/lateral rotation to produce smooth forward locomotion. Removing shoulder abduction/adduction causes more body roll and yaw, but increased scapulothorax rotation keeps the body moving forward in relatively-smooth manner. Additionally removing scapulothorax rotation causes more pitching and yawing, and a substantial increase in lateral and vertical displacement.

Removing shoulder medial/lateral rotation causes increased rolling and lateral displacement, but increased scapulothorax rotation help to keep the body moving forward relatively smoothly. Body rolling, yawing, and lateral displacement are all increased by

additionally removing the scapulothorax rotation FDOF. It is interesting to note that shoulder medial/lateral rotation is virtually unused in the absence of abduction/adduction, indicating the coupling between the FDOFs. Table 19 shows the effects of removing FDOFs on *Triceratops* gait observables. The *Triceratops* was not capable of a step length 0.34 time the hip height without all forelimb FDOFs, so a step length 0.24 times the hip height was used for this study.

Table 19.

Effect of removing forelimb FDOFs on Triceratops gait observables

	Observable							
	Body angle (°)			Body displacement (cm)		FDOF angular excursion (°)		
Value	Pitch	Yaw	Roll	Lateral	Vertical	Scapulothorax [a]	Shoulder 2 [b]	Shoulder 3 [c]
					All limb FDOFs			
M	2.27	1.03	1.66	5.31	1.67	12.86	17.92	5.32
SD	0.08	0.52	0.51	0.34	1.09	4.23	2.31	3.65
					Missing shoulder abduction/adduction FDOF			
M	2.28	0.48	1.31	2.37	1.84	25.95	0.00	5.83
SD	0.48	0.24	0.13	0.59	1.55	1.14	0.00	1.47
					Missing shoulder abduction/adduction and scapulothorax FDOFs			
M	1.74	4.58	3.50	5.81	0.77	0.00	0.00	16.00
SD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00
					Missing shoulder medial/lateral rotation FDOF			
M	2.42	0.99	2.19	4.45	2.93	20.21	6.39	0.00
SD	0.56	0.39	0.93	2.26	0.43	5.64	6.66	0.00
					Missing shoulder medial/lateral rotation and scapulothorax FDOFs			
M	2.51	5.36	7.32	4.64	2.79	0.00	8.23	0.00
SD	0.00	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Note. [a] Scapulothorax rotation. [b] Shoulder abduction/adduction. [c] Shoulder medial/lateral rotation.

Like *Apatosaurus*, the *Triceratops* utilizes combinations of shoulder abduction/adduction and medial/lateral rotation to move the body forward smoothly. Without shoulder flexion/extension, increased scapulothorax rotation helps to minimize the observables. Additionally removing scapulothorax rotation causes a substantial increase in body roll and yaw.

Removing shoulder medial/lateral rotation causes an increase in pitch, roll, and vertical displacement, but again an increased in scapulothorax rotation helps to keep the body moving forward relatively smoothly. By additionally removing the scapulothorax rotation FDOF, the body experiences a significant increase in yawing and rolling. It is also interesting to note that *Triceratops* does not utilize as much shoulder abduction/adduction in the absence of medial/lateral rotation, indicating the coupling between these FDOFs.

The data presented in this section showed that shoulder abduction/adduction and medial/lateral rotation are not utilized by the dog walking gaits. The *Apatosaurus* and *Triceratops* walking gaits use combinations of these FDOFs to create smooth forward walking gaits. With both models, increased scapulothorax rotation was utilized to partially compensate for the loss of either shoulder FDOF. Both models experienced significant increases in gait observable values in the absence of scapulothorax rotation and one of the shoulder FDOFs. In the next section, optimal ankle heights will be determined for animals with varying amounts of pes flexibility.

Optimal Ankle Height

Coombs (1978) describes a high ankle height as a cursorial (i.e., running) specialization, as opposed to graviportal (i.e., specialized for bearing weight) animals which generally have lower ankle heights. The generic dog has a relatively-high ankle height, as does the *Tyrannosaurus* model. Both of these models also have a highly flexible manus and pes (modeled by a manus/pes joint in both models and an additional phalangeal joint in the *Tyrannosaurus* model). The *Apatosaurus* and *Triceratops* models, however, have relatively-low ankle heights and a relatively-inflexible manus and pes. Due to these correlations, it seems reasonable that a lower ankle height may be necessary for animals with inflexible feet.

Two studies were conducted to determine the effect of ankle height on gait observables. First, the hindlimb proportions of the *Apatosaurus*, *Triceratops*, dog, and *Tyrannosaurus* models were varied to determine an optimal ankle height. Ankle heights were adjusted by scaling the limb elements distal to the ankle and then scaling the crus/antibrachium to maintain a constant hip height. Figure 112 shows the *Apatosaurus* model with differing ankle heights. The optimal limb proportions were not evolved as they were in the work presented by Sims (1994b, 1994b) and others; instead, GAs were used to evaluate three possible ankle heights for each animal. For the second study, gait observable data was collected for the dog and *Tyrannosaurus* models both with and without pes flexibility to determine the effect of pes flexibility on the smoothness of the generated walking gaits.

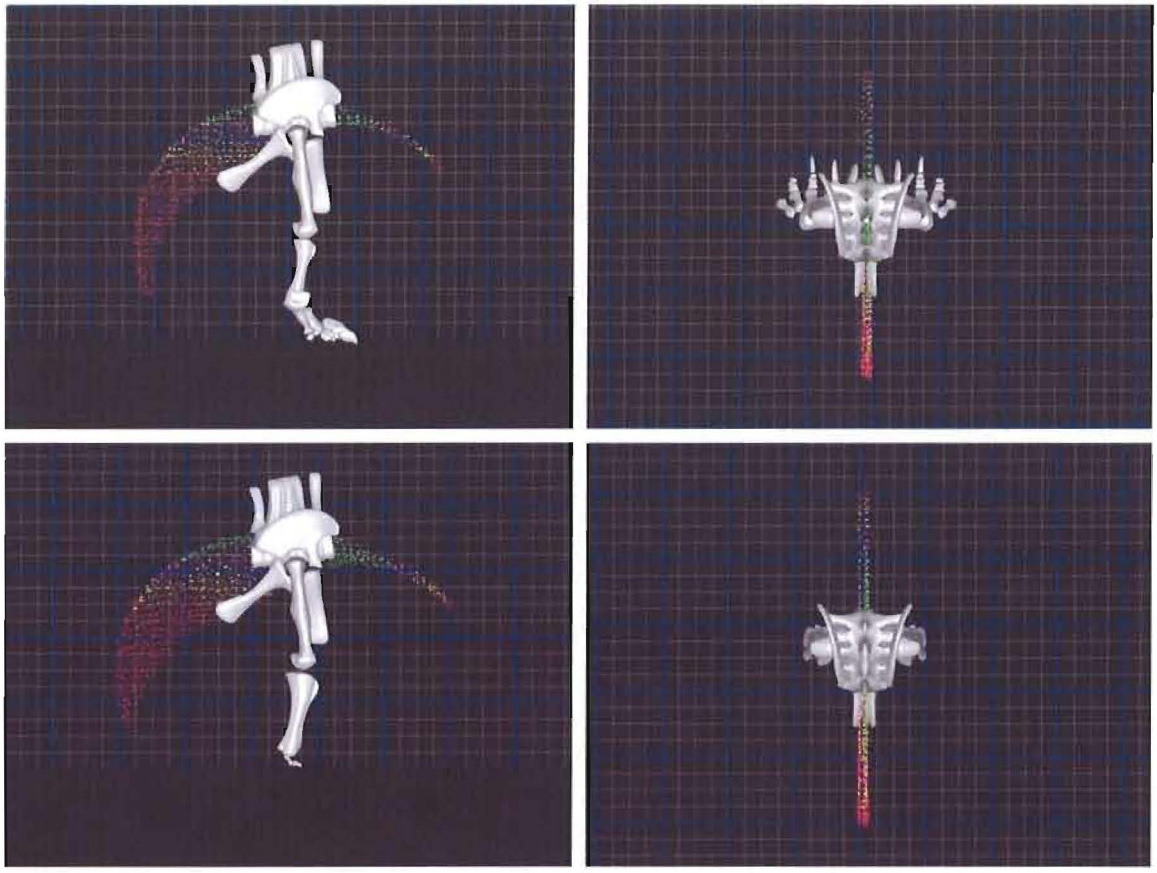


Figure 112. Comparison of high (top) and low (bottom) *Apatosaurus* ankle heights.

When walking, animals "pole vault" over their limbs along a vertical arc during the limb's stance phase (Farley, 1997). In the absence of pes flexibility, the radius of the "pole vault" arc is largely determined by the distance between the ankle joint and root positions. Angular excursion at the knee affects this arc to a lesser extent because the knee joint is closer to the root. Similarly, angular excursion at the shoulder has even less effect on the arc. Therefore, animals with limited to no pes flexibility should be capable

of taking longer steps with lower ankle heights. Table 20 shows the effect of varying ankle height on *Apatosaurus* gait observables.

Table 20.

Effect of varying ankle height on Apatosaurus gait observables

	Gait observable				
Value	Pitch (°)	Yaw (°)	Roll (°)	Lateral (cm)	Vertical (cm)
			Half ankle height		
M	0.97	0.13	0.15	0.81	5.79
SD	0.30	0.02	0.00	0.04	1.15
			Base ankle height		
M	1.64	0.15	0.10	0.62	4.17
SD	0.29	0.01	0.02	0.14	0.37
			Double ankle height		
M	11.21	6.50	1.02	4.31	29.53
SD	0.00	0.00	0.00	0.00	0.00

As expected, the *Apatosaurus* experiences a general increase in all gait observables as the ankle height increases. Body pitch and vertical displacement increase dramatically with ankle height, which is significant because these observables contribute directly to an animal's kinematic ability to take relatively-long steps. In terms of smoothness of forward locomotion, the animal would have benefitted most from a very low ankle. However, this observation is purely kinematic; a tradeoff must exist between ankle height and overall pes size such that the animal's pes could support a significant portion of its overall mass. Table 21 shows the effect of varying ankle height on *Triceratops* gait observables.

Table 21.

Effect of varying ankle height on Triceratops gait observables

	Gait observable				
Value	Pitch (°)	Yaw (°)	Roll (°)	Lateral (cm)	Vertical (cm)
			Half ankle height		
M	1.17	0.49	0.51	6.49	2.06
SD	0.35	0.05	0.05	0.41	0.87
			Base ankle height		
M	2.46	0.76	2.06	6.93	4.02
SD	0.31	0.03	0.04	0.01	0.10
			Double ankle height		
M	3.07	2.23	1.46	4.96	11.91
SD	0.46	0.07	0.06	0.25	0.45

Like *Apatosaurus*, the *Triceratops* also experiences a general increase in almost all gait observables as ankle height increases, although not as pronounced as with *Apatosaurus*. In fact, the lateral displacement decreases with respect to ankle height. This observed difference between the two models is likely related to the less-parasagittal LROM space of the *Triceratops*, indicating more complex interactions between FDOFs. Like *Apatosaurus*, a low ankle height is kinematically optimal for *Triceratops*. Table 22 shows the effect of varying ankle height on dog gait observables.

Table 22.

Effect of varying ankle height on dog gait observables

	Gait observable				
Value	Pitch (°)	Yaw (°)	Roll (°)	Lateral (cm)	Vertical (cm)
			Half ankle height		
M	2.65	1.36	0.43	0.97	2.92
SD	0.00	0.00	0.00	0.00	0.00
			Base ankle height		
M	3.31	1.70	0.64	0.98	0.68
SD	0.31	0.54	0.33	0.41	0.09
			Double ankle height		
M	6.12	1.67	1.01	0.65	2.23
SD	1.03	0.14	0.70	0.07	0.40

The dog model experiences increased pitching as ankle height increases, similar to the other models. However, analogous trends were not observed with respect to the other gait observables, as they were with the *Apatosaurus* and *Triceratops* models. The dog model has one pes FDOF while the *Apatosaurus* and *Triceratops* have no pes flexibility. This additional FDOF contributes to a more-robust LROM space, which would intuitively allow smoother gaits. Table 23 shows the effect of varying ankle height on *Tyrannosaurus* gait observables.

Table 23.

Effect of varying ankle height on Tyrannosaurus gait observables

	Gait observable				
Value	Pitch (°)	Yaw (°)	Roll (°)	Lateral (cm)	Vertical (cm)
			Half ankle height		
M	1.67	4.91	2.14	6.19	11.42
SD	0.21	0.62	0.61	1.81	2.18
			Base ankle height		
M	1.70	4.76	1.91	3.89	14.95
SD	0.27	0.16	0.34	0.74	2.40
			Double ankle height		
M	4.05	1.13	1.80	4.66	14.58
SD	1.12	0.08	0.87	0.90	2.02

Like the dog model, gaits generated for the *Tyrannosaurus* model show a general increase in body pitching, but no similar trends in terms of the other gait observables. In fact, body yawing is significantly reduced with a high ankle height. These differences can be attributes to the two pes FDOFs of the *Tyrannosaurus* model, contributing to a dense, largely-parasagittal LROM Space. Figure 113 shows a comparison of LROM spaces corresponding to high and low *Tyrannosaurus* ankle heights.

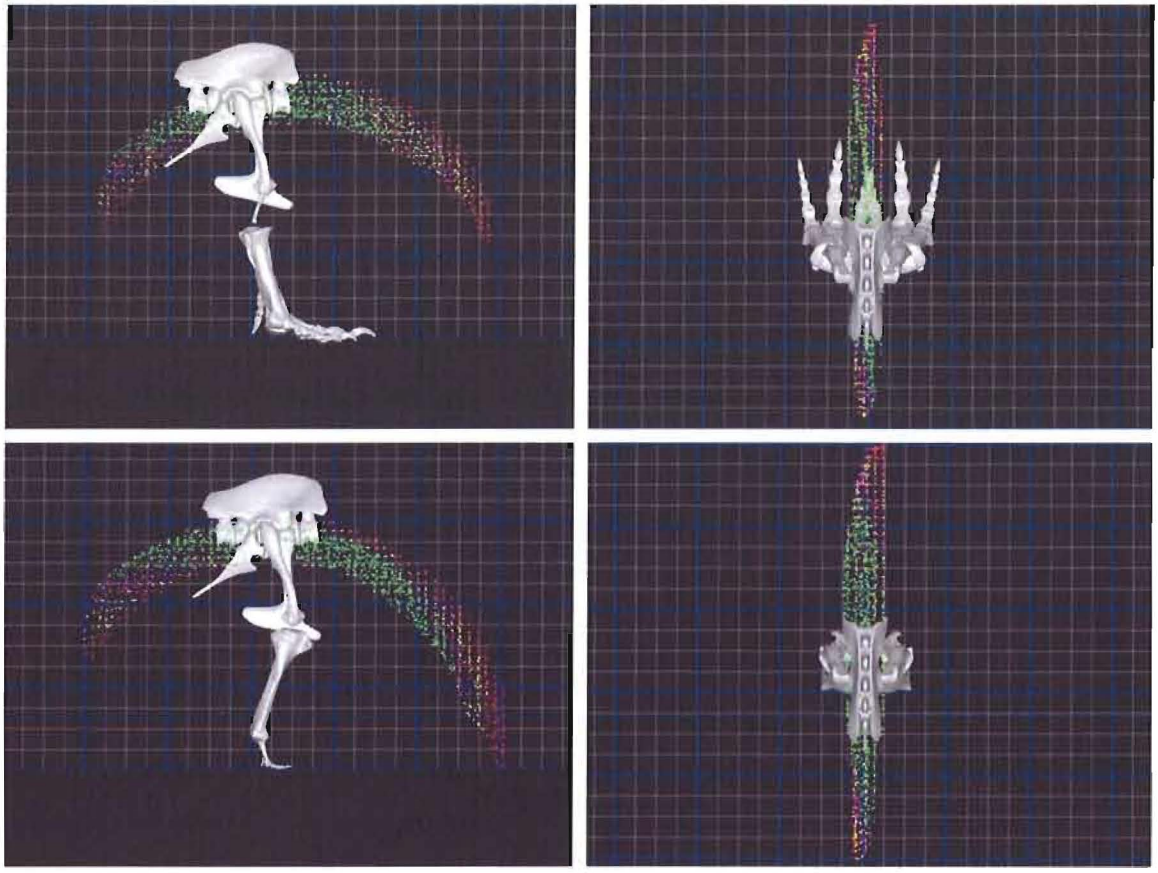


Figure 113. Comparison of high (top) and low (bottom) *Tyrannosaurus* ankle heights.

Across all four models, reduced body pitching was observed in gaits generated with low ankle heights. Body pitching is an important gait observable because it directly relates to an animal's reasonable step length. Additional reduction in other gait error terms was observed for *Apatosaurus* and *Triceratops* gaits, likely due the lack of flexibility in the *Apatosaurus* and *Triceratops* pes. Lower ankle heights have a kinematic advantage over higher ankle heights, so there must be other advantages for higher ankle heights in cursorial animals. One such advantage could be related to the dynamics of the

limb; a higher ankle height indicates a longer pes, which provides a greater moment arm for thrusting the animal forward.

The *Apatosaurus* and *Triceratops* models have no flexibility in the pes, resulting in low optimal ankle heights. The dog and *Tyrannosaurus* models do have pes flexibility, so the functional purpose of that flexibility can be evaluated by collecting gait observable data for walking gaits with and without pes flexibility. Like the models without pes flexibility, the dog and *Tyrannosaurus* models should be capable of longer step lengths when utilizing pes flexibility. Table 24 shows the effect of removing pes FDOFs on dog gait observables. The dog model was not capable of a step length 0.48 times the hip height without pes FDOFs, so a step length 0.26 times the hip height was used for this study.

Table 24.

Effect of removing pes FDOF on dog gait observables

	Gait observable					
Value	Pitch (°)	Yaw (°)	Roll (°)	Lateral (cm)	Vertical (cm)	Pes [a] (°)
				All FDOFs		
M	0.74	1.14	0.63	0.48	0.62	20.14
SD	0.13	0.35	0.18	0.11	0.25	14.34
				Missing pes joint FDOF		
M	3.26	4.59	2.89	1.02	6.03	0.00
SD	0.00	0.00	0.00	0.00	0.00	0.00

Note. [a] Pes flexion/extension.

As predicted, the dog makes extensive use of the pes flexion/extension FDOF. Without this FDOF, all gait observable are increased significantly. This data supports the observation that a relatively-high ankle is not very useful without a highly-flexible pes. Table 25 shows the result of remove pes and phalangeal joint FDOFs on the *Tyrannosaurus* gait observables. The *Tyrannosaurus* model is not capable of a step length 0.98 times the hip height without pes FDOFs, so a step length 0.64 times the hip height was used for this study.

Table 25.

Effect of removing pes FDOFs on Tyrannosaurus gait observables

	Gait observable						
Value	Pitch (°)	Yaw (°)	Roll (°)	Lateral (cm)	Vertical (cm)	Pes [a] (°)	Phalangeal [b] (°)
	All FDOFs						
M	1.46	0.61	1.09	3.75	10.53	37.44	0.00
SD	0.36	0.10	0.10	0.69	5.15	13.01	0.00
	Missing pes jand phalangeal joint FDOFs						
M	3.08	5.28	2.40	4.53	19.36	0.00	0.00
SD	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Note. [a] Pes flexion/extension. [b] Phalangeal flexion/extension.

Like the dog, the *Tyrannosaurus* model experienced increases in all gait observables in the absence of pes and phalangeal flexion/extension FDOFs. The pes joint FDOF was used extensively, but the phalangeal joint FDOF was surprisingly unutilized. The phalangeal joint FDOF was observed in use during the qualitative analysis of the *Tyrannosaurus* walking gait in a previous section, so the lack of phalangeal-joint

utilization is likely due to the reduced step length used for this study. Intuitively, this is a nice result as it indicates that the phalangeal joint FDOF is used only to gain extra limb retraction during relatively-long steps.

Animals with limited or no pes flexibility seem to be able to make use of a low ankle height to achieve long, smooth strides. Conversely, animals with high ankles seemingly need a large about of pes flexion/extension to achieve long, smooth strides. The data presented in this section supports these arguments by showing that animals with no pes flexibility take smoother strides with low ankles and that animals with high ankles make extensive use of pes flexion/extension. In the next section, the effect of ipsilateral phase will be evaluated on quadrupedal gaits.

Ipsilateral Phase

Ipsilateral phase describes the phase relationship between the hindlimbs and forelimbs. This phase is a normalized quantity (i.e., on the range [0.0, 1.0]) representing the full 360° of possible phases. An ipsilateral phase of 0.0 or 1.0 indicates that the right hindlimb and right forelimb begin their stance phases at the same time (i.e., a pacing gait); a phase of 0.5 indicates that the right hindlimb and left forelimb begin their stance phases at the same time (i.e., a diagonal trotting gait). In this section, data will be presented that shows the relationship between ipsilateral phase and gait observables.

Gait observable data was collected for ipsilateral phases between 0.0 and 0.9 at 0.1 phase intervals. This gait observable data shows quantitatively how gaits change with respect to ipsilateral phase. In general, animals with non-parasagittal gait movements

(i.e., with substantial yaw and/or lateral displacement) should be affected by ipsilateral phase because this phase changes the relative time at which hindlimbs and forelimbs are yawed and displaced relative to each other. Conversely, animals with highly-parasagittal limb movements should not be greatly affected by ipsilateral phase.

Some amount of diversity in yaw and lateral displacement observable values with respect to ipsilateral phase is expected due to the discrete sampling of the trunk ROM space. It is interesting to note that extreme differences in yaw and lateral displacement between forelimbs and hindlimbs may not even be achievable by the trunk of an animal. Table 26 shows the effect of varying ipsilateral phase on the dog yaw and lateral displacement observables.

Table 26.

Effect of varying ipsilateral phase on dog gait observables

	Ipsilateral phase									
Observable	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
	Yaw (°)									
M	5.64	6.97	7.11	6.42	5.12	5.18	5.34	6.03	7.21	6.89
SD	1.17	2.19	2.02	1.11	0.74	0.38	0.39	1.47	1.89	0.50
	Lateral displacement (cm)									
M	1.48	1.25	1.36	1.33	1.48	1.51	1.45	1.66	1.51	1.49
SD	0.23	0.22	0.23	0.28	0.22	0.36	0.28	0.30	0.17	0.35

As predicted, the dog model gait observables are not largely dependent on ipsilateral phase. The yaw values vary between ~5° and ~7°, which generally fall within

the standard deviations. The lateral displacements also show little variation. Figure 114 shows a graphical representation of the yaw observable with respect to ipsilateral phase.

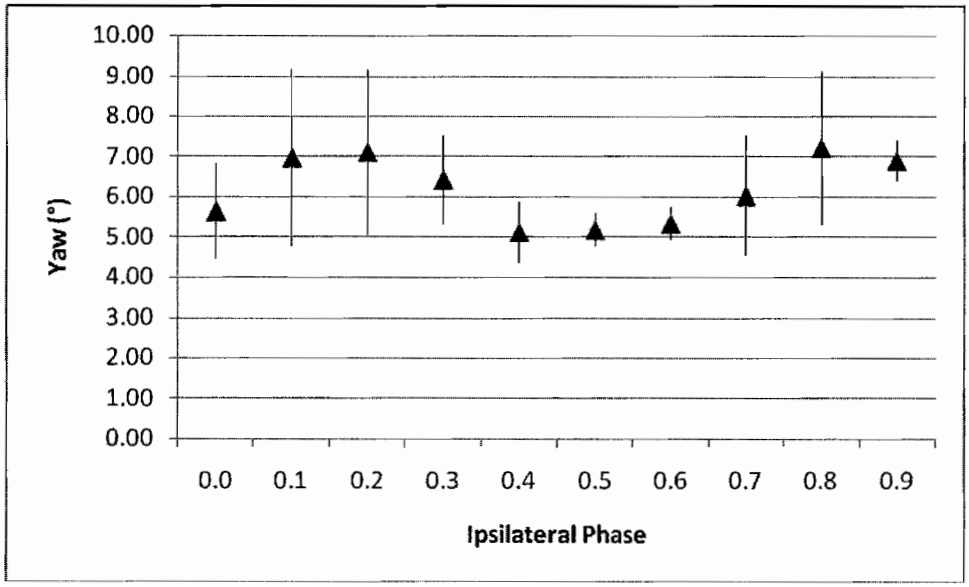


Figure 114. Effect of varying ipsilateral phase on dog body yawing.

Like the dog model, the *Apatosaurus* hindlimb LROM space is highly parasagittal. Unlike the dog model, the *Apatosaurus* forelimb LROM space is robust, but the model is capable of smooth paths through the forelimb space. Table 27 shows the effect of varying ipsilateral phase on *Apatosaurus* yaw and lateral offset observables.

Table 27.

Effect of varying ipsilateral phase on Apatosaurus gait observables

	Ipsilateral phase									
Observable	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
	Yaw (°)									
M	1.38	1.29	1.79	1.43	1.51	1.41	1.31	1.65	1.76	1.58
SD	0.16	0.39	1.23	0.29	0.26	0.49	0.45	0.54	1.49	0.26
	Lateral displacement (cm)									
M	3.65	3.96	3.90	3.99	3.45	5.14	4.33	2.84	4.33	3.83
SD	0.20	1.56	2.51	0.52	1.45	3.11	1.84	1.19	1.07	1.85

Like the dog model, the *Apatosaurus* yaw and lateral displacement gait observables are largely unaffected by ipsilateral phase. Figure 115 shows a graphical representation of the effect of ipsilateral phase on the *Apatosaurus* yaw observable.

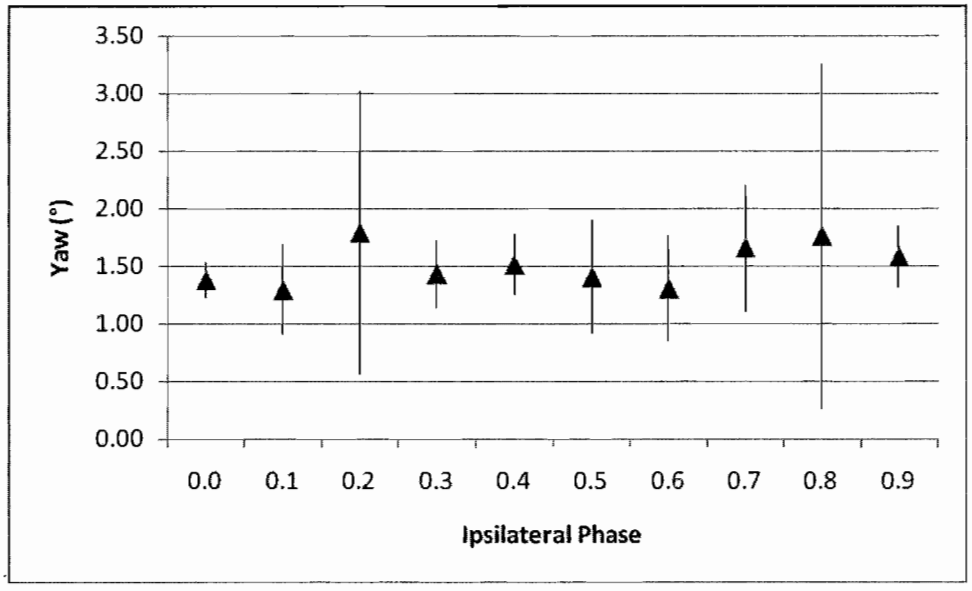


Figure 115. Effect of varying ipsilateral phase on *Apatosaurus* body yawing.

Like the *Apatosaurus* model, the *Triceratops* hindlimb LROM space is highly parasagittal and the forelimb LROM space is robust, but the model is capable of supporting smooth forward locomotion. Table 28 shows the effect of varying ipsilateral phase on *Triceratops* yaw and lateral displacement.

Table 28.

Effect of varying ipsilateral phase on Triceratops gait observables

	Ipsilateral phase									
Observable	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
	Yaw (°)									
M	5.12	5.32	4.66	4.20	4.46	4.53	4.66	4.35	2.94	3.79
SD	1.71	1.62	1.56	1.53	1.26	1.11	2.74	1.81	1.37	0.75
	Lateral displacement (cm)									
M	11.84	10.94	11.04	10.75	9.32	10.40	11.33	11.55	11.31	12.26
SD	3.06	1.48	2.69	2.76	1.69	1.71	2.61	2.03	2.22	1.45

Again, there is no obvious relationship between ipsilateral phase and the yaw and lateral displacement observables of the *Triceratops* model. Figure 116 shows a visual representation of the effect of ipsilateral phase on the *Triceratops* yaw and lateral displacement observables.

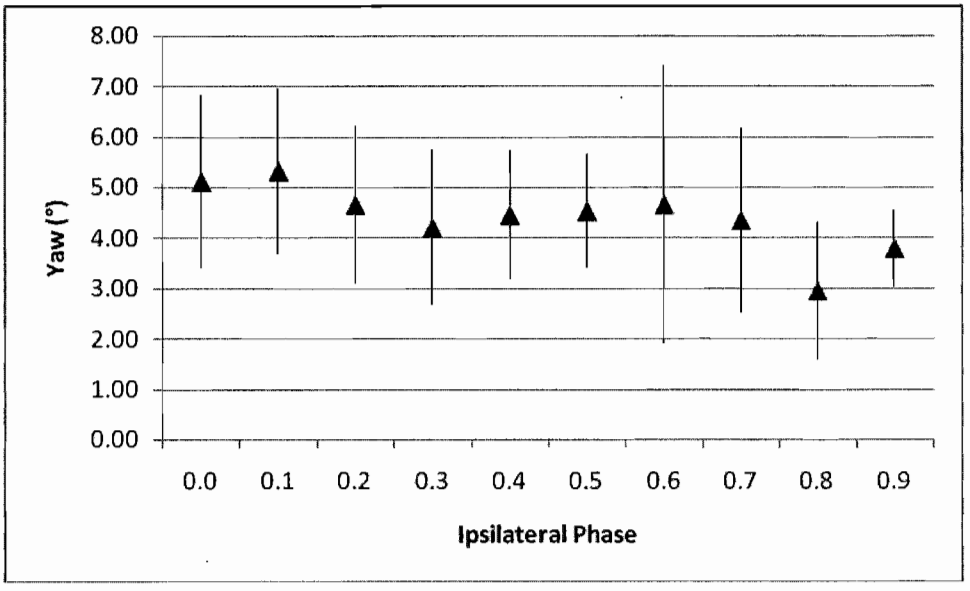


Figure 116. Effect of varying ipsilateral phase on *Triceratops* body yawing.

Unlike the dog, *Apatosaurus*, and *Triceratops* models, the generic reptile model is not capable of either parasagittal hindlimb or forelimb movements due to its sprawling gait. Therefore, it seems intuitive that the reptile would likely not be capable of accomplishing the requested hind and fore step lengths for certain ipsilateral phase values. For example, an ipsilateral phase of 0.0 should give similar orientation for hind and fore roots because a 0.5 phase gives nearly opposite orientation due to sinuous trunk movements. Such similar orientation would only be possible if the orientations pointed the both roots nearly forward, otherwise the trunk would need to bend in unnatural ways with at least two inflection points. Table 29 shows the result of varying ipsilateral phase on the reptile yaw and lateral displacement observables.

Table 29.

Effect of varying ipsilateral phase on reptile gait observables

	Ipsilateral phase									
Observable	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9
	Yaw (°)									
M	23.60	26.34	28.50	28.00	25.76	27.02	27.49	27.14	27.58	25.84
SD	2.36	2.97	3.41	4.40	5.17	2.37	3.27	3.32	2.35	1.19
	Lateral displacement (cm)									
M	24.27	25.08	25.81	26.49	25.51	24.79	26.57	30.04	27.43	27.58
SD	1.07	2.24	4.12	2.57	2.86	5.56	5.81	3.76	2.82	3.43

Surprisingly, there does not seem to be a relationship between ipsilateral phase and body yaw or lateral displacement. Figure 117 shows a graphical representation of the effect of ipsilateral phase on reptile gait yawing and lateral displacement.

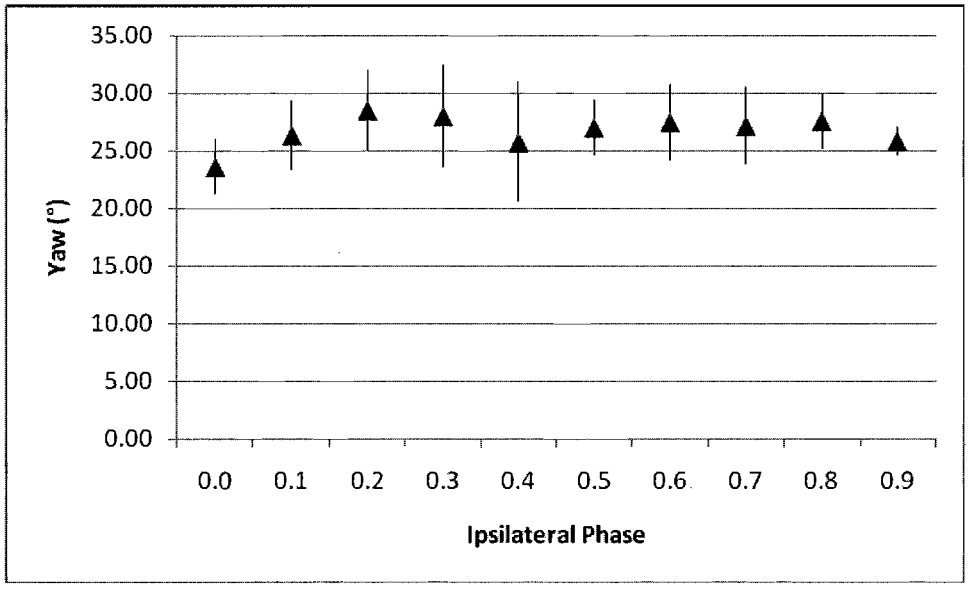


Figure 117. Effect of varying ipsilateral phase on reptile body yawing.

Upon further investigation, it became obvious that the reptile model's hindlimbs are kinematically capable of pointing the hindlimb root nearly forward, allowing quadrupedal gaits with any ipsilateral phase value. The ability to point the hindlimb root forward could be taken away by limiting the shoulder flexion/extension FDOF, but doing so would result in less-characteristic reptile gaits. Figure 118 compares reptile gaits with a 0.5 ipsilateral phase (top) and a 0.0 ipsilateral phase (bottom).

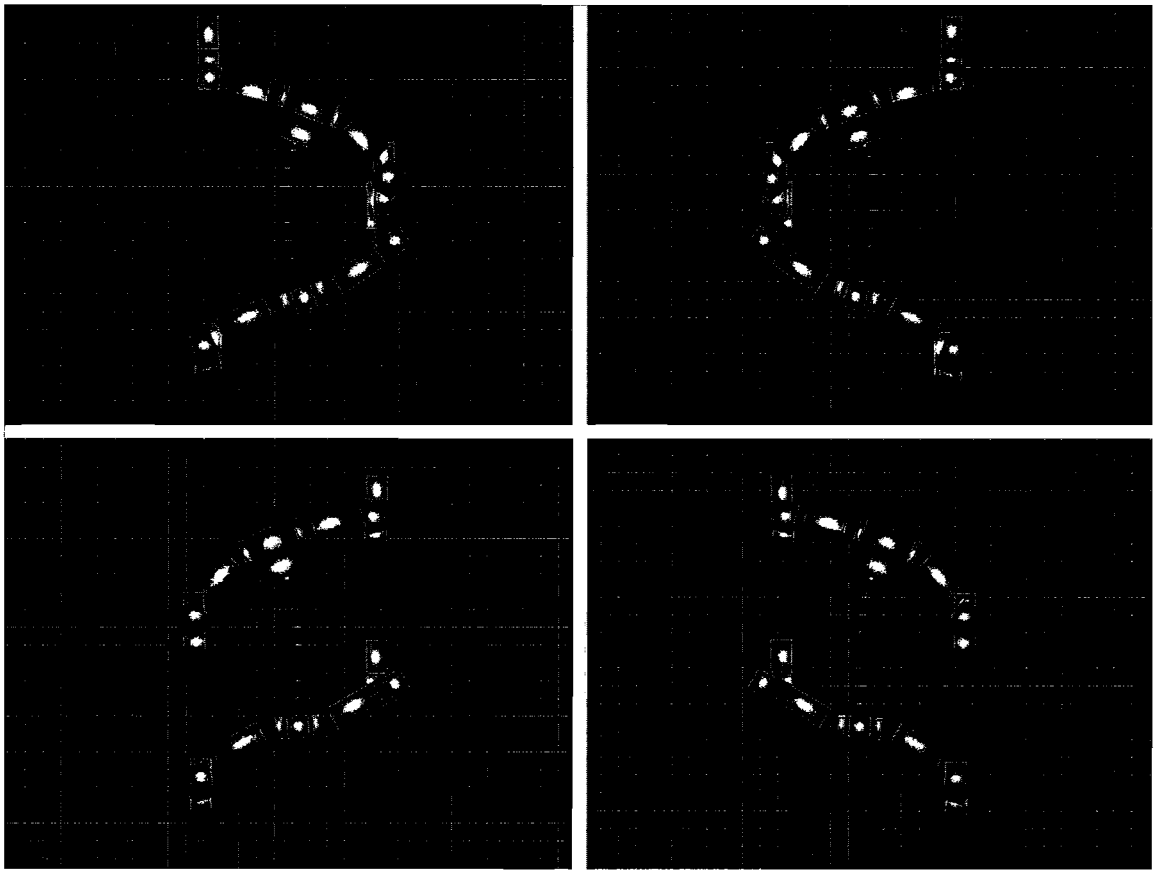


Figure 118. Comparison of 0.5 (top) and 0.0 (bottom) reptile ipsilateral phases.

Based on these results, ipsilateral phase seems to be a non-kinematic issue that cannot be resolved with this approach. Ipsilateral phase is more likely related to the issue of support; animals with a relatively-low COM will tend to use diagonal couplet (i.e., trotting) gaits because they can more easily keep their COM between supporting limbs. Hildebrand (1980) supports this argument by stating that tetrapods tend to avoid lateral couplet gaits at slow speeds in favor of lateral sequence and diagonal couplet gaits because of the increased support triangle area. Alexander (1983) reports that contralateral and ipsilateral gait phases observed in animals are related to limb lengths and velocity (i.e. Froude numbers).

The data presented in the section confirm that the gait observables of quadrupedal models capable of parasagittal gait movements (i.e., dog, *Apatosaurus*, *Triceratops*) are not significantly affected by ipsilateral phase. Surprisingly, the gait observables of the reptile model were also not significantly affected by ipsilateral phase. The latter result helps to confirm the observations of others that gait selection and ipsilateral phase are stability and dynamics issues, and not simple kinematic issues.

Summary

The data presented in this chapter confirms the utility of the LROM space representation and the use of efficient GAs to find gaits that smoothly move the animal through the space. Data related to the tuning of the GA was first presented. Parent selection parameters were varied, along with crossover and mutation coefficients to find optimal GA parameters. Candidate populate size was also varied to find a sufficient

population size parameter. A convergence comparison was then presented that showed the tuned GA's ability to outperform a simulated Hill Climbing algorithm.

A sensitivity analysis of the parameters used to build the LROM spaces demonstrated that the angular sampling resolution used to initially exercise the limb did not significantly alter the fitness of generated gaits. Varying the number of LROM space boxes that the LROM root positions are sorted into also did not have a significant effect of generated gait fitness. Space refinement was shown to significantly increase gait fitness without changing functional joint behavior, indicating that joint behavior is not sensitive to isolated changes in fitness error values. Lastly, an analysis of variations to the fitness function coefficients showed that changes to the coefficients had an intuitive result on gait fitness (e.g., increasing the pitch error coefficient decreased the amount of observed pitching) without affecting functional joint behaviors.

A qualitative gait analysis demonstrated that the hindlimb and forelimb movements of the generated quadrupedal dog gaits match closely to published data. Similarly, the hindlimb and forelimb movements of the generated quadrupedal reptile gaits matched closely to published data. Comparisons were then made between the hindlimb and forelimb movements of the dog and reptile to analogous movements of the *Apatosaurus*, *Triceratops,* and *Tyrannosaurus* dinosaurs. Observations were then presented on the similarities in joint functionality between animals.

Finally, a quantitative analyses presented data on specific gait hypotheses: The dog model was shown to not utilize its shoulder abduction/adduction and medial/lateral rotation FDOFs; the *Apatosaurus* and *Triceratops* models were shown to make use of

combinations of these FDOFs to achieve smooth forward locomotion. Next, animals without pes flexibility were shown to benefit from low ankle heights while animals with high ankle heights were shown to depend heavily on pes flexibility. Lastly, ipsilateral phase was shown to not have a significant effect on kinematically-generated gaits, indicating that ipsilateral phase selection is more an issue of stability and dynamics than kinematics.

The data presented in this chapter demonstrates some of the capabilities and limitations of the GAGA methods presented in the previous chapter. These methods are capable of exploring an LROM, and sorting the results into an LROM space, pruning the space, and finding combinations of configurations in the space that represent smooth gaits. The next chapter will present an in-depth discussion of the capabilities and limitations of the GAGA approach and the significance of the data and results presented in this chapter.

CHAPTER V

DISCUSSION

3D animations of locomotion are particularly useful for gait analysis. The GAGA methods presented in earlier chapters allow such animations to be automatically generated using kinematic explorations of limb capabilities and GAs to find plausible locomotion paths through LROM spaces. Using these methods, generated walking gaits were compared to real-world analogues as controls to confirm the viability of a particular GA fitness function for locomotion synthesis. The same fitness function was then used to generate forward walking gaits for dinosaurs, about which relatively little is known. The methods and investigations presented in this paper have several interesting implications and corollaries, which will be discussed in this chapter.

Limb biomechanics are usually described in proximal-to-distal order. For example, the hip and knee joints are generally considered to be responsible for moving the animal forward, while the ankle joint and pes flexibility are responsible for adjusting to uneven terrain (Daley, Felix, and Biewener, 2007). When describing joint behaviors with respect to LROM spaces, however, it is most intuitive to consider joints in distal-to-proximal order. This distal-to-proximal ordering is more natural when considering

locomotion as a process of first constraining the location of the foot that is in contact with the ground, then considering the forward movement of the body as constrained by the kinematics of the limb and body above that foot.

In general, the contribution that an FDOF makes to the LROM space increases with the 3D distance between the joint and the root position (i.e., exercising the FDOF in isolation causes the root to rotate on an arc about the joint). Of course the distance between a joint and the root will generally change during locomotion due to intermediate joints. Combinations of FDOFs will move the root along paths which are largely within the animal's sagittal plane if the limb's FDOFs have axes of rotation that are nearly perpendicular to the sagittal plane. Conversely, a limb with divergent FDOF axes will have a broader LROM space.

It is therefore the distal-most joints (e.g., the ankle joint and pes flexibility) that have the greatest effect in moving the root joint forward, allowing the root to move on an arc long enough to move the animal forward by a step length. The mid-limb joints (i.e., knee and elbow) can contribute to the arc of the root to a lesser extent. The mid-limb joints can also alter the limb's length through flexion/extension, allowing less vertical displacement during locomotion. The proximal joints (e.g., hip, shoulder, and scapulothorax) are sufficiently close to the root position that their major contribution is to modify the root orientation such that body pitching, rolling, and yawing are minimized.

The configurations of an LROM space are typically distributed along an animal's sagittal plane, with enough thickness in the space to allow many possible forward movements. When an LROM space is initially generated and visualized, it is sometimes

obvious that there are not enough parasagittal LROM space configurations to allow a reasonable step length. These situations typically occur when a generally-planar LROM space is not aligned with the animal's sagittal plane, either by rotation due to the manus/pes orientation parameter or lateral translation due to step width parameter. In these cases, the step width, manus/pes orientation and/or limb FDOFs must be modified to better align the LROM space with the animal's sagittal plane. Otherwise, not enough configurations will survive the contralateral constraint process and the GA will not be able to find forward paths through the constrained space. Figure 119 shows an example of a planar but non-parasagittal LROM space for the *Apatosaurus* hindlimb.

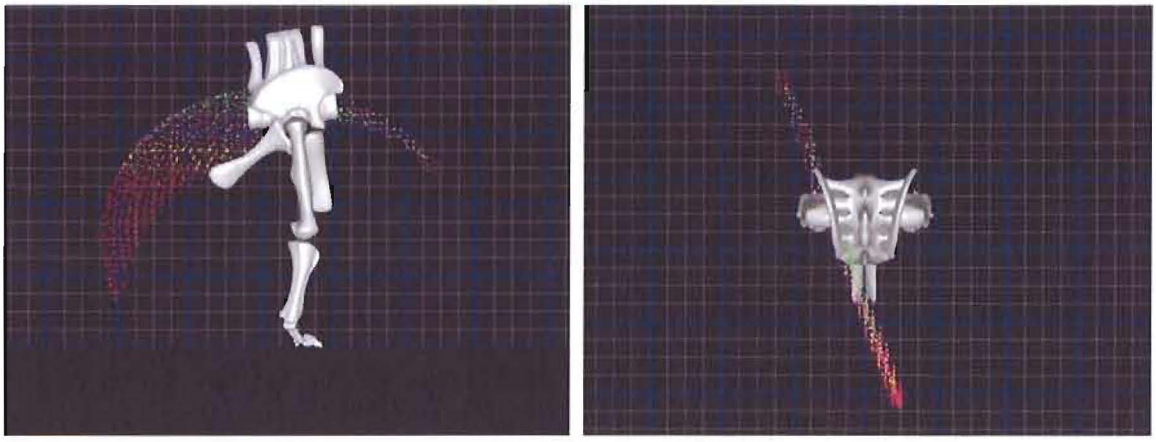


Figure 119. Planar but non-parasagittal *Apatosaurus* hindlimb LROM space.

Variations on FDOFs can quickly be evaluated due to the fast computational speed of the GAGA methods (i.e., automatically generating bipedal gaits in about one

minute and quadrupedal gaits in under five minutes on a laptop). For example, pronation/supination in the generic reptile model's crus and antibrachium can be turned off by changing a scripted parameter. The reptile would likely not be capable of its current step length 2.2 times the hip height, indicating that the animal requires crus/antibrachium pronation/supination to achieve that step length without twisting or squishing the manus/pes on the ground. If the reptile were able to achieve a step length 2.2 times the hip height without pronation/supination, the GA gait observables (e.g., body pitch, roll, and yaw, and lateral/vertical displacement) and visualization of the gait itself would give indications as to whether or not the gait would be realistic.

The case studies presented in the last chapter show the utility of the GAGA methods for quantitative analysis. The studies varied scripted FDOF parameters to evaluate the functionality of FDOFs, used scripted scaling values to vary ankle height, and varied phase parameters to determine the effect of ipsilateral phase on quadrupedal gaits across animals. These case studies yielded significant and publishable results. Due to the computational efficiency of the kinematics-only exploration, constraint, and GA algorithms, study data can be collected relatively quickly; roughly 100 quadrupedal data points can be collected (i.e., ten GA runs per data point) over a 24 hour period.

The contributions of the scapulothorax and shoulder joint FDOFs were isolated for the dog, *Apatosaurus*, and *Triceratops* models. The investigation revealed that the dog model uses little shoulder abduction/adduction and medial/lateral rotation, and utilizes scapulothorax rotation to minimize body pitching during locomotion. These results are intuitive given the highly-parasagittal nature of the dog forelimb LROM space.

The forelimbs of *Apatosaurus* and *Triceratops*, however, make extensive use of shoulder abduction/adduction and medial/lateral rotation to allow the limbs to navigate smoothly through their broader LROM spaces (i.e., which result from divergent limb FDOF axes).

Cursorial (i.e., specialized for running) animals typically exhibit high ankle heights while graviportal (i.e., specialized for bearing weight) animals typically have lower ankle heights (Coombs, 1978). The investigation into the significance of ankle height revealed that animals with lower ankle heights have a kinematic advantage over animals with higher ankle heights. Animals with higher ankle heights require additional flexibility in their manus and pes to be capable of long strides.

Gaits were generated with varying ipsilateral phase values for all four quadrupedal models. The results of this investigation showed no discernible kinematic advantages for some phase values over others. These results were not surprising for the dog, *Apatosaurus*, and *Triceratops* models, as those models are capable of generally-parasagittal limb movements that should allow locomotion without much lateral bending of the trunk. The reptile, however, utilizes extensive sinuous lateral trunk movements during locomotion (Reilly, 1997, 1998), so it was surprising that ipsilateral phase had little effect on the gait observables. The lack of change in trunk behavior was due to ROMs of the reptile hips; the ROMs needed to allow angular excursions sufficient for reptilian limb movements, but these ROMs then also allowed the hips to keep the body pointing relatively forward during locomotion (see Figure 118). These results indicate that gait phase is a complicated issue that requires more than kinematics to properly investigate.

A qualitative analysis was used to determine how well GAGA-generated gaits match those of real-world animals. Much thought was given to the possibility of a quantitative analysis to show these properties. Ultimately, there did not seem to be any reasonable quantitative means for comparison. Exact joint angles are difficult to determine reliably and are dependent on a large number of external factors. Froude numbers are often used to compare gaits, but usually in terms of relative velocities (which cannot be easily computed using kinematics-only models). For these reasons, a quantitative gait comparison study is left for future work.

GAGA methods use kinematics only to analyze gaits; dynamics were omitted by design to allow an exploration of the viability of kinematics-only gait analysis. Stability studies based on support triangles (Henderson, 2006) would be difficult because the system does not keep track of an animal's COM. COM tracking could be added, however, by scripting a mass parameter for each body segment and then computing the instantaneous overall COM during locomotion. Similarly, gaits with an aerial phase cannot be modeled with GAGA methods because the system has no way of predicting COM trajectories during aerial phases. These omissions have been largely strategic; adding terms to the fitness function that require dynamics simulation would increase the running time of the GAs by several orders of magnitude. Nonetheless, dynamics-based gait observables could be integrated into the GAGA methods at a future time.

The GAGA methods have applications in any area that requires a quantitative analysis of changes to joint functionality or limb proportions. FDOFs can be based on any movement, so LROM spaces could theoretically be generated for a typical knee joint

and a knee joint with a prosthetic articular surface, similar to the study by Piazza and Delp (2001). The two spaces could be compared to determine how well the prosthetic approximates a typical knee joint. Gaits could also be generated from the two spaces so that the difference in movements could be viewed in terms of gait observables, but the fitness function and gait observables might need to be modified so that the biological knee model behavior matches closely to that of a real knee. Such fitness function changes are straightforward with respect to GAGA, but determining an appropriate fitness function in terms of the high-level goals of a joint or limb can be difficult.

GAGA methods could also be used for limb-based studies other than locomotion. These methods could be used to investigate how an animal transitions from a sitting to standing posture and vice versa, similar to Stevens, Larson, Wills, and Anderson (2008). Such transitions are likely generally constrained by the osteology of an animal with the musculature acting on the bones and joints, much like with locomotion. Therefore, GAGA methods could be used for the kinematic synthesis and analysis of these movements.

CHAPTER VI

CONCLUSIONS

The LROM space representation provides intuitive visualizations of LROMs consisting of discrete FDOFs. Changes in FDOFs can be easily detected by comparing resulting LROM spaces. LROM spaces are pruned using dual support and bilateral symmetry constraints so that only limb configurations relevant to bipedal locomotion are considered by the GAs. LROM spaces are further pruned by trunk constraints for quadrupedal locomotion. Highly-optimized GAs find plausible paths through these large spaces with relatively-little computational effort, allowing the evaluation of many FDOF and gait parameter variations.

Walking gaits are generated from only two key configurations, representing the RD and RLU gait events. These two key configurations are near the beginning of the right limb's support phase, when the limb is reaching forward. The RD and RLU gait events, the dual support constraint, and the bilateral symmetry constraint determine two additional key configurations, namely for RLD and RU. The RLD and RU configurations are near the end of the right limb's support phase, when the limb is reaching back. The limb is sent on a "pole vaulting" trajectory between RLU and RLD

which is determined by all four right-side limb events. The four right-side events are bilaterally mirrored to the left side, specifying configurations for the four left-side events. In this way, configurations for a total of eight locomotion events are specified from the original two key configurations (i.e., RD and RLU).

Limbs have relatively few ways of reaching forward and backwards, compared to a limb's many possible mid-stance configurations. The GAs are therefore able to search relatively-small spaces of possible configurations (i.e., pruned by the dual support and bilateral symmetry constraints). GA performance is additionally improved by a candidate representation and fitness function that guarantee that each candidate gait at least accomplishes a specified stride length. The GAs therefore focus entirely on reducing gait error values (i.e., body pitch, yaw, roll, lateral/vertical displacement, and angular excursions at the joints).

The gaits generated using GAGA methods were shown to be stable with respect to changes in the LROM space parameters. Changes to the fitness function error terms were shown to have intuitive effects on the generated gaits. Gaits can be refined to further increase fitness, but this process does not have a significant effect on joint functionality, indicating that limb movements are highly constrained by their limited ROMs near the beginning and end of their stance phase. By pruning LROM spaces such that only configurations during dual support are considered, GAGA methods are able to quickly and accurately reproduce complete gait cycles.

Future work will likely involve further qualitative and quantitative studies on the animal models presented in this paper and on animals not yet modeled. There are

potentially a number of additional publishable results that could come out of future studies using these methods. A quantitative study may be devised to compare GAGA-generated gaits to those of real-world animals. Such a study would likely require extensive collaboration with a laboratory capable of collecting large amounts of animal gait data. This type of study would provide more insight into the capabilities and limitations of kinematic-only gait synthesis and analysis.

APPENDIX A

DOG MODEL DATA

Table 30.

Dog FDOFs

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Hip_Joint_R						
Base	0	0	0	-8	-9	-8
FDOF 1						
Minimum	-25	0	0	0	0	0
Neutral	-12	0	0	0	0	0
Maximum	20	0	0	0	0	0
FDOF 2						
Minimum	0	-8	0	0	0	0
Neutral	-12	0	0	0	0	0
Maximum	0	8	0	0	0	0
FDOF 3						
Minimum	0	0	-10	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	10	0	0	0
Knee_Joint_R						
Base	0	0	0	0	0	41
FDOF 1						
Minimum	-20	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	20	0	0	0	0	0
Ankle_Joint_R						
Base	0	0	0	0	0	47
FDOF 1						
Minimum	-20	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	15	0	0	0	0	0

Dog FDOFs (continued)

FDOF	Rotation (°)			Position (cm)		
	x	*y*	*z*	*x*	*y*	*z*
Pes_Joint_R						
Base	0	0	0	0	0	23
FDOF 1						
Minimum	-20	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	20	0	0	0	0	0
Hip_Joint_L						
Base	0	0	0	8	-9	-8
FDOF 1						
Minimum	-25	0	0	0	0	0
Neutral	-12	0	0	0	0	0
Maximum	20	0	0	0	0	0
FDOF 2						
Minimum	0	8	0	0	0	0
Neutral	-12	0	0	0	0	0
Maximum	0	-8	0	0	0	0
FDOF 3						
Minimum	0	0	10	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	-10	0	0	0
Knee_Joint_L						
Base	0	0	0	0	0	41
FDOF 1						
Minimum	-20	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	20	0	0	0	0	0
Ankle_Joint_L						
Base	0	0	0	0	0	47
FDOF 1						
Minimum	-20	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	20	0	0	0	0	0
Pes_Joint_L						
Base	0	0	0	0	0	23
FDOF 1						
Minimum	-20	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	20	0	0	0	0	0
Scapulothorax_Joint_R						
Base	0	0	0	-10	2	-5
FDOF 1						
Minimum	8	0	0	0	0	0
Neutral	8	0	0	0	0	0
Maximum	36	0	0	0	0	0

Dog FDOFs (continued)

FDOF	Rotation (°)			Position (cm)		
	x	*y*	*z*	*x*	*y*	*z*
Shoulder_Joint_R						
Base	0	0	0	0	0	18
FDOF 1						
Minimum	-26	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	20	0	0	0	0	0
FDOF 2						
Minimum	0	-8	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	8	0	0	0	0
FDOF 3						
Minimum	0	0	-10	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	10	0	0	0
Elbow_Joint_R						
Base	0	0	0	0	0	31
FDOF 1						
Minimum	-30	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	20	0	0	0	0	0
Wrist_Joint_R						
Base	0	0	0	0	0	39
FDOF 1						
Minimum	-25	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	35	0	0	0	0	0
Manus_Joint_R						
Base	0	0	0	0	0	17
FDOF 1						
Minimum	-25	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	25	0	0	0	0	0
Scapulothorax_Joint_L						
Base	0	0	0	10	2	-5
FDOF 1						
Minimum	8	0	0	0	0	0
Neutral	8	0	0	0	0	0
Maximum	36	0	0	0	0	0
Shoulder_Joint_L						
Base	0	0	0	0	0	18
FDOF 1						
Minimum	-26	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	20	0	0	0	0	0

Dog FDOFs (continued)

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
FDOF 2						
Minimum	0	8	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	-8	0	0	0	0
FDOF 3						
Minimum	0	0	10	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	-10	0	0	0
Elbow_Joint_L						
Base	0	0	0	0	0	31
FDOF 1						
Minimum	-30	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	20	0	0	0	0	0
Wrist_Joint_L						
Base	0	0	0	0	0	39
FDOF 1						
Minimum	-25	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	35	0	0	0	0	0
Manus_Joint_L						
Base	0	0	0	0	0	17
FDOF 1						
Minimum	-25	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	25	0	0	0	0	0

APPENDIX B

REPTILE MODEL DATA

Table 31.

Reptile FDOFs

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Hip_Joint_R						
Base	0	0	0	0	0	10
FDOF 1						
Minimum	0	-50	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	10	0	0	0	0
FDOF 2						
Minimum	-10	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	10	0	0	0	0	0
FDOF 3						
Minimum	0	0	-60	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	8	0	0	0
Knee_Joint_R						
Base	0	0	0	0	0	58
FDOF 1						
Minimum	-11	-26	2	0	0	0
Neutral	0	0	0	0	0	0
Maximum	9	21	2	0	0	0
Crus_PS_R						
Base	0	0	0	0	0	42
FDOF 1						
Minimum	0	0	-60	0	0	0
Neutral	0	0	0	0	0	0

Reptile FDOFs (continued)

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Maximum	0	0	12	0	0	0
Ankle_Joint_R						
Base	0	0	0	0	0	50
FDOF 1						
Minimum	-20	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	5	0	0	0	0	0
Pes_Joint_R						
Base	0	0	0	0	0	37
FDOF 1						
Minimum	-30	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	5	0	0	0	0	0
FDOF 2						
Minimum	0	0	-20	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	20	0	0	0
Hip_Joint_L						
Base	0	0	0	0	0	10
FDOF 1						
Minimum	0	50	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	-10	0	0	0	0
FDOF 2						
Minimum	-10	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	10	0	0	0	0	0
FDOF 3						
Minimum	0	0	60	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	-8	0	0	0
Knee_Joint_L						
Base	0	0	0	0	0	58
FDOF 1						
Minimum	-11	26	-2	0	0	0
Neutral	0	0	0	0	0	0
Maximum	9	-21	-2	0	0	0
Crus_PS_L						
Base	0	0	0	0	0	42
FDOF 1						
Minimum	0	0	60	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	-12	0	0	0
Ankle_Joint_L						

Reptile FDOFs (continued)

FDOF	Rotation (°)			Position (cm)		
	x	*y*	*z*	*x*	*y*	*z*
Base	0	0	0	0	0	50
FDOF 1						
Minimum	-20	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	5	0	0	0	0	0
Pes_Joint_L						
Base	0	0	0	0	0	37
FDOF 1						
Minimum	-30	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	5	0	0	0	0	0
FDOF 2						
Minimum	0	0	20	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	-20	0	0	0
Shoulder_Joint_R						
Base	0	0	0	0	0	20
FDOF 1						
Minimum	0	-40	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	10	0	0	0	0
FDOF 2						
Minimum	-10	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	10	0	0	0	0	0
FDOF 3						
Minimum	0	0	-30	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	10	0	0	0
Elbow_Joint_R						
Base	0	0	0	0	0	63
FDOF 1						
Minimum	12	-37	-3	0	0	0
Neutral	0	0	0	0	0	0
Maximum	-4	15	-1	0	0	0
Antibrachium_PS_R						
Base	0	0	0	0	1	47
FDOF 1						
Minimum	0	0	-25	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	30	0	0	0
Wrist_Joint_R						
Base	0	0	0	0	-1	45
FDOF 1						

Reptile FDOFs (continued)

FDOF	Rotation (°)			Position (cm)		
	x	*y*	*z*	*x*	*y*	*z*
Minimum	-20	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	5	0	0	0	0	0
Manus_Joint_R						
Base	0	0	0	0	0	37
FDOF 1						
Minimum	-30	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	5	0	0	0	0	0
FDOF 2						
Minimum	0	0	-20	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	20	0	0	0
Shoulder_Joint_L						
Base	0	0	0	0	0	20
FDOF 1						
Minimum	0	40	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	-10	0	0	0	0
FDOF 2						
Minimum	-10	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	10	0	0	0	0	0
FDOF 3						
Minimum	0	0	30	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	-10	0	0	0
Elbow_Joint_L						
Base	0	0	0	0	0	63
FDOF 1						
Minimum	12	37	3	0	0	0
Neutral	0	0	0	0	0	0
Maximum	-4	-15	1	0	0	0
Antibrachium_PS_L						
Base	0	0	0	0	1	47
FDOF 1						
Minimum	0	0	25	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	-30	0	0	0
Wrist_Joint_L						
Base	0	0	0	0	-1	45
FDOF 1						
Minimum	-20	0	0	0	0	0
Neutral	0	0	0	0	0	0

Reptile FDOFs (continued)

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Maximum	5	0	0	0	0	0
Manus_Joint_L						
Base	0	0	0	0	0	37
FDOF 1						
Minimum	-30	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	5	0	0	0	0	0
FDOF 2						
Minimum	0	0	20	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	-20	0	0	0

APPENDIX C

APATOSAURUS MODEL DATA

Table 32.

Apatosaurus FDOFs

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Hip_Joint_R						
Base	3	0	0	22	-13	11
FDOF 1						
Minimum	-21	0	1	0	0	0
Neutral	0	0	0	0	0	0
Maximum	18	0	-1	0	0	0
FDOF 2						
Minimum	0	-10	-1	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	13	1	0	0	0
FDOF 3						
Minimum	0	0	-10	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	10	0	0	0
Knee_Joint_R						
Base	-4	-2	1	-32	18	149
FDOF 1						
Minimum	-20	0	0	0	-3	0
Neutral	0	0	0	0	-5	0
Maximum	74	0	0	0	-14	0
Ankle_Joint_R						
Base	0	0	0	6	7	128
FDOF 1						
Minimum	-19	-5	-28	0	2	1
Neutral	0	0	0	0	0	0

Apatosaurus FDOFs (continued)

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Maximum	20	-5	29	0	0	0
Hip_Joint_L						
Base	3	0	0	-22	13	-11
FDOF 1						
Minimum	-21	0	1	0	0	0
Neutral	0	0	0	0	0	0
Maximum	18	0	-1	0	0	0
FDOF 2						
Minimum	0	-10	-1	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	13	1	0	0	0
FDOF 3						
Minimum	0	0	-10	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	10	0	0	0
Knee_Joint_L						
Base	-4	-2	1	32	-18	-149
FDOF 1						
Minimum	-20	0	0	0	3	0
Neutral	0	0	0	0	5	1
Maximum	74	0	0	0	14	0
Ankle_Joint_L						
Base	0	0	0	-6	-7	-128
FDOF 1						
Minimum	-19	-5	-28	0	2	1
Neutral	0	0	0	0	0	0
Maximum	20	-5	29	0	0	0
Scapulothorax_Joint_R						
Base	2	-9	-13	-83	-16	53
FDOF 1						
Minimum	-17	5	-5	0	0	0
Neutral	0	0	0	0	0	0
Maximum	10	0	4	0	0	0
FDOF 2						
Minimum	0	0	0	0	15	0
Neutral	0	0	0	0	0	0
Maximum	0	0	0	0	-10	0
Shoulder_Joint_R						
Base	0	8	-20	-4	-2	29
FDOF 1						
Minimum	-35	0	0	0	0	0
Neutral	-4	-3	1	0	0	0
Maximum	35	0	0	0	0	0
FDOF 2						

Apatosaurus FDOFs (continued)

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Minimum	0	-18	0	0	0	0
Neutral	-4	-3	1	0	0	0
Maximum	0	18	0	0	0	0
FDOF 3						
Minimum	0	0	-10	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	10	0	0	0
Elbow_Joint_R						
Base	3	6	-40	-15	6	100
FDOF 1						
Minimum	-33	-15	2	1	2	0
Neutral	0	0	0	0	0	0
Maximum	25	2	4	0	0	0
Ulnocarpal_Joint_R						
Base	-54	0	15	-6	2	88
FDOF 1						
Minimum	-10	-14	-3	0	0	0
Neutral	0	0	0	0	0	0
Maximum	24	27	13	0	0	0
Scapulothorax_Joint_L						
Base	2	-9	-13	-21	-16	97
FDOF 1						
Minimum	-17	5	-5	0	0	0
Neutral	0	0	0	0	0	0
Maximum	10	0	4	0	0	0
FDOF 2						
Minimum	0	0	0	0	15	0
Neutral	0	0	0	0	0	0
Maximum	0	0	0	0	-10	0
Shoulder_Joint_L						
Base	0	8	-20	4	2	-29
FDOF 1						
Minimum	-35	0	0	0	0	0
Neutral	-4	-3	1	0	0	0
Maximum	35	0	0	0	0	0
FDOF 2						
Minimum	0	-18	0	0	0	0
Neutral	-4	-3	1	0	0	0
Maximum	0	18	0	0	0	0
FDOF 3						
Minimum	0	0	-10	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	10	0	0	0
Elbow_Joint_L						

Apatosaurus FDOFs (continued)

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Base	3	6	-40	15	-6	-100
FDOF 1						
Minimum	-33	-15	2	1	2	0
Neutral	0	0	0	0	0	0
Maximum	25	2	4	0	0	0
Ulnocarpal_Joint_L						
Base	-54	0	15	6	-2	-88
FDOF 1						
Minimum	-10	-14	-3	0	0	0
Neutral	0	0	0	0	0	0
Maximum	24	27	13	0	0	0

APPENDIX D

TRICERATOPS MODEL DATA

Table 33.

Triceratops FDOFs

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*Z*
Hip_Joint_R						
Base	0	0	0	8	-11	-29
FDOF 1						
Minimum	8	20	16	0	0	0
Neutral	0	2	-3	0	0	0
Maximum	-1	-31	-22	0	0	0
FDOF 2						
Minimum	-3	-8	13	0	0	0
Neutral	0	2	-3	0	0	0
Maximum	1	6	-8	0	0	0
FDOF 3						
Minimum	-12	3	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	15	-3	0	0	0	0
Knee_Joint_R						
Base	0	0	0	91	21	12
FDOF 1						
Minimum	-20	4	-24	0	0	0
Neutral	-23	4	-1	0	0	0
Maximum	-19	0	63	0	0	0
Ankle_Joint_R						
Base	4	-20	-1	77	-1	0
FDOF 1						
Minimum	-58	-1	9	3	0	0
Neutral	-4	10	0	0	-1	-1

Triceratops FDOFs (continued)

FDOF	Rotation (°)			Position (cm)		
	x	*y*	*z*	*x*	*y*	*z*
Maximum	56	20	0	3	-1	-3
Hip_Joint_L						
Base	0	0	0	8	-11	29
FDOF 1						
Minimum	-8	-20	16	0	0	0
Neutral	-4	-2	-3	0	0	0
Maximum	-1	31	-22	0	0	0
FDOF 2						
Minimum	3	8	13	0	0	0
Neutral	-4	-2	-3	0	0	0
Maximum	-1	-6	-8	0	0	0
FDOF 3						
Minimum	12	-3	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	-15	3	0	0	0	0
Knee_Joint_L						
Base	0	0	0	91	21	-12
FDOF 1						
Minimum	20	-4	-24	0	0	0
Neutral	23	-4	-1	0	0	0
Maximum	19	0	63	0	0	0
Ankle_Joint_L						
Base	-4	20	-1	77	-1	0
FDOF 1						
Minimum	58	1	9	3	0	0
Neutral	4	-10	0	0	-1	1
Maximum	-56	-20	0	3	-1	3
Scapulothorax_Joint_R						
Base	0	0	0	72	27	-5
FDOF 1						
Minimum	9	4	-2	0	0	0
Neutral	0	0	0	0	0	0
Maximum	-17	-3	-2	0	0	0
FDOF 2						
Minimum	0	0	0	1	10	1
Neutral	0	0	0	0	0	0
Maximum	0	0	0	-2	-13	-2
Shoulder_Joint_R						
Base	0	0	0	13	-4	14
FDOF 1						
Minimum	3	2	-23	0	0	0
Neutral	0	0	0	0	0	0
Maximum	-5	0	39	-4	-2	-5
FDOF 2						

Triceratops FDOFs (continued)

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Minimum	0	7	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	-12	0	0	0	0
FDOF 3						
Minimum	-8	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	8	0	0	0	0	0
Elbow_Joint_R						
Base	0	0	0	66	-5	-8
FDOF 1						
Minimum	6	28	39	1	-4	-2
Neutral	0	0	0	0	0	0
Maximum	6	-16	-13	0	0	0
Wrist_Joint_R						
Base	0	0	0	48	0	0
FDOF 1						
Minimum	1	1	20	0	0	0
Neutral	0	0	0	0	0	0
Maximum	-1	-3	-32	3	1	0
Scapulothorax_Joint_L						
Base	0	0	0	40	27	-60
FDOF 1						
Minimum	-9	-3	-3	0	0	0
Neutral	0	0	0	0	0	0
Maximum	17	3	-2	0	0	0
FDOF 2						
Minimum	0	0	0	1	10	1
Neutral	0	0	0	0	0	0
Maximum	0	0	0	-2	-13	-2
Shoulder_Joint_L						
Base	0	0	0	13	-4	-14
FDOF 1						
Minimum	1	-3	-23	0	0	0
Neutral	0	0	0	0	0	0
Maximum	5	-3	41	-3	-4	2
FDOF 2						
Minimum	0	-7	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	12	0	0	0	0
FDOF 3						
Minimum	8	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	-8	0	0	0	0	0
Elbow_Joint_L						

Triceratops FDOFs (continued)

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Base	2	0	0	67	-5	8
FDOF 1						
Minimum	-4	-26	40	0	-4	1
Neutral	0	0	0	0	0	0
Maximum	-6	16	-13	0	1	0
Wrist_Joint_L						
Base	0	0	0	48	0	0
FDOF 1						
Minimum	-2	-1	20	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	1	-32	4	1	0

APPENDIX E

TYRANNOSAURUS MODEL DATA

Table 34.

Tyrannosaurus FDOFs

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Hip_Joint_R						
Base	0	0	0	4	5	-6
FDOF 1						
Minimum	0	-10	-35	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	20	0	0	0
FDOF 2						
Minimum	0	-10	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	5	0	0	0	0
FDOF 3						
Minimum	-10	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	10	0	0	0	0	0
Knee_Joint_R						
Base	0	0	0	114	7	23
FDOF 1						
Minimum	0	0	-20	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	25	0	0	0
Ankle_Joint_R						
Base	0	0	0	115	-4	0
FDOF 1						
Minimum	0	0	-25	0	0	0
Neutral	0	0	0	0	0	0

Tyrannosaurus FDOFs (continued)

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Maximum	0	0	30	0	0	0
Pes_Joint_R						
Base	0	0	0	60	-10	1
FDOF 1						
Minimum	0	0	-20	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	30	0	0	0
Phalangeal_Joint_R						
Base	0	0	0	21	1	0
FDOF 1						
Minimum	0	0	-5	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	30	0	0	0
Hip_Joint_L						
Base	0	0	0	-4	-5	6
FDOF 1						
Minimum	0	-10	-35	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	20	0	0	0
FDOF 2						
Minimum	0	-10	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	5	0	0	0	0
FDOF 3						
Minimum	-10	0	0	0	0	0
Neutral	0	0	0	0	0	0
Maximum	10	0	0	0	0	0
Knee_Joint_L						
Base	0	0	0	-113	-6	-20
FDOF 1						
Minimum	0	0	-20	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	20	0	0	0
Ankle_Joint_L						
Base	0	0	0	-115	4	0
FDOF 1						
Minimum	0	0	-25	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	30	0	0	0
Pes_Joint_L						
Base	0	0	0	-60	10	-1
FDOF 1						
Minimum	0	0	-20	0	0	0
Neutral	0	0	0	0	0	0

Tyrannosaurus FDOFs (continued)

	Rotation (°)			Position (cm)		
FDOF	*x*	*y*	*z*	*x*	*y*	*z*
Maximum	0	0	30	0	0	0
Phalangeal_Joint_L						
Base	0	2	0	-21	-1	0
FDOF 1						
Minimum	0	0	-5	0	0	0
Neutral	0	0	0	0	0	0
Maximum	0	0	30	0	0	0

APPENDIX F

FIXING ORIENTATION AND POSITION

1. Store the initial *y* axis of the end effector as *initialYAxis*
2. Store the initial *z* axis of the end effector as *initialZAxis*
3. Store the initial quaternion rotation state of the root as *initialRotation*
4. Store the initial position of the end effector as *initialEndPosition*
5. Store the initial position of the root as *initialRootPosition*
6. For each frame rendered following a limb configuration change:
    a. Store the current *y* axis of the end effector as *yAxis*
    b. Store the current *z* axis of the end effector as *zAxis*
    c. Calculate the quaternion that rotates *yAxis* into *initialYAxis* and store as *yRotation*
    d. Multiply *zAxis* by *yRotation*
    e. Calculate the quaternion that rotates *zAxis* into *initialZAxis* and store as *zRotation*
    f. Set the root orientation to *zRotation*yRotation*initialRotation*

g. Store the current end effector position as *position*

h. Calculate the vector that moves *position* to *initialEndPosition* and store as *translation*

i. Set the root position to *initialRootPosition* + *translation*

APPENDIX G

GENETIC ALGORITHM STRUCTURE

1. Create an initial population consisting of randomly-generated, fixed-length binary strings and store as *population*
2. Create a binary string to hold the overall best (i.e., highest fitness) candidate solution and store as *globalBest*
3. For each genetic algorithm iteration:
    a. Apply crossover to the population by potentially combining each candidate with another randomly-selected binary string
    b. Apply mutation to the population by potentially flipping bits in each binary string
    c. Evaluate the fitness of the population, storing the fitness of each candidate
    d. If any candidate has a fitness higher than that of *globalBest*, store that candidate as *globalBest*

e. Select new parents for the next iteration by comparing each candidate to another randomly-selected candidate and usually selecting (i.e., based on a probability coefficient) the higher-fitness candidate

f. Store the new parents as *population*

4. Report and handle the highest-fitness candidate, *globalBest*

APPENDIX H

EXAMPLE BIPEDAL CANDIDATE

1. An LROM space for the right *Tyrannosaurus* hindlimb is generated by exploring all possible combinations of the limb's six FDOFs (see Appendix E) at a 5° angular resolution while keeping the pes position/orientation fixed on the ground (see Appendix F); the space contains 2,882,880 total samples
2. Each sample in the LROM space contains 12 floating point values: the FDOF values that specify the joint configurations of the limb (i.e., one float per FDOF, six total floats), the 3D position of the sample (i.e., 3 floats), and the pitch, yaw, and roll of the root element necessary to reach the sample position using the associated FDOF values
3. The LROM space is organized into 3D boxes, with 50 boxes along the direction of travel (i.e., the number of boxes along the direction of travel is an input parameter):
    a. The LROM space measures 5.03 meters along the direction of travel, so each box is 10 cm x 10 cm x 10 cm

b. The space is 0.57 meters wide along the lateral axis, so there are 6 boxes along the lateral axis
c. The space is 2.76 meters tall along the vertical axis, so there are 28 boxes along the vertical axis
d. The space contains 6 x 50 x 28 boxes for a total of 8,400 boxes
e. Each LROM space sample is sorted into the 3D box that surrounds the sample's 3D position, resulting in a list of samples for each 3D box

4. The LROM space is pruned based on the dual support and bilateral symmetry constraints; a suitable sibling sample is attempted to be found for each sample in the LROM space:
    a. The three indices of the sample's 3D box are determined based on the sample's 3D position, the bounds of the LROM space, and the 3D box dimensions
    b. The forward index of the sibling 3D box is determined by subtracting the dual support length from the step length (i.e., the step length is an input parameter, the dual support length is a function of the step length and the duty factor), then dividing by the box dimensions to determine the number of boxes that the animal will move forward through between the RD and RLD events
    c. The sibling lateral index is determined by reflecting the original index across the sagittal plane (i.e., using the lateral index of the sagittal plane)
    d. The vertical sibling index is equal to the original vertical index

e. If the sibling 3D box contains a sample with an RMS angular offset (i.e., the absolute value of the sibling sample angles minus the original sample angles) less than the specified epsilon error, store that sample as a sibling match for the original sample; if multiple sibling samples are found with less than the epsilon angular offset, use the sibling sample with the smallest angular offset

f. Remove all samples from the space that do not have a sibling match, ensuring that all remaining samples satisfy the dual support and bilateral symmetry constraints

g. The constrained space contains 80,640 total samples (i.e., including sibling samples)

5. The size of the linear binary candidates is computed:

a. The size of the back data is computed as the number of non-sibling samples in the constrained space that are at least a step length back from the front of the space (i.e., so selections for the RD event will be able to complete a step of the specified length); the back data contains 11,145 samples, so 14 bits are required to index all back data samples

b. The size of the largest slice in the slice data is computed by looking for the largest number of samples in any plane of 3D boxes perpendicular to the direction of travel; the largest slice contains 3,665 samples, so 12 bits are needed to address all slice data samples

c. Each candidate is a 14-bit index into the back data representing a sample for the RD event and a 12-bit index into a slice in the slice data representing a sample for the RLU event (i.e. the specific slice for the RLU event sample is determined by the slice of the RD event sample and the forward distance that the animal must travel during dual support); each candidate is represented by a 26-bit binary string

6. The GA is run for 10,000 iterations with a population of 1000 candidates (see Appendix G), each iteration:
   a. Each candidate is potentially combined with another randomly-selected candidate using single-point crossover and a crossover probability of 0.1
   b. Each candidate is potentially mutated with a probability of 0.5 per candidate (i.e., on average one bit is flipped for half of the candidates in the population)
   c. The fitness of each candidate is calculated as $1.0 / (1.0 + e)$, where e is the weighted sum of terms stored in the RD and RLU event samples (i.e., from the 12 values stored in each sample): 5.0 times the sum body pitch caused by the RD and RLU events, 2.0 times the sum body yaw cause by these events, the sum body roll caused by these events, the sum vertical displacement between the event sample positions and the target vertical height, the sum lateral displacement between the event sample positions and the initial sagittal plane position, and the sum differences between the FDOF values of the RD and RLU events; the body roll/pitch/yaw and sum

FDOF difference terms are multiplied by the step length so that these angular terms can be related to the translational terms

d. If any candidate has higher fitness than the highest-fitness candidate found thus far, store that candidate as the highest-fitness candidate

e. Select new parents for the next GA iteration using Tournament Selection with a 0.9 probability of selecting the higher-fitness candidate (i.e., compare each candidate in the population with another randomly-selected candidate and select the higher-fitness candidate 90% of the time)

7. The highest-fitness candidate after the 10,000 GA iterations is 0 0 1 0 0 0 1 1 0 0 1 1 0 0 1 1 1 0 1 0 1 0 1 1 1 0, which corresponds to 1,531st entry in the back data (i.e., specified by the 14 highest-order bits), which is in the 6th slice (i.e., determined by the forward 3D position of the sample), and the 3,362nd entry (i.e. determined by the 12 lowest-order bits) in the 12th slice (i.e., determined by the slice of the RD sample and the number of slices the animal must travel through during dual support to achieve the specified step length); the fitness of this candidate is 1.67642 x 10^-3
8. The stored sibling samples for the RD and RLU event samples are retrieved and used as the RLD and RU event samples, respectively
9. Animate the highest-fitness candidate by creating an animation curve for each FDOF of the right limb; each curve has six key frames: the FDOF values at RD, RLU, RLD, RU, a FDOF value for mid swing phase to get the pes off the ground

(i.e., determined by simple IK and/or direct posing), and the FDOF value at RD to ensure that the animation is cyclical

10. Create animation curves for the left limb using the same key frames as the right limb but advanced along the animation timeline by 0.5 times the animation time to ensure a 0.5 ipsilateral phase
11. During animation, fix the position/orientation of the right pes on the ground between the RD and RLD events and fix the position/orientation of the left pes on the ground between the LD and LRD (i.e., between the RLD and RD) events (see Appendix F)

APPENDIX I

GLOSSARY

2D: Two Dimensional

3D: Three Dimensional

ALife: Artificial Life

ANN: Artificial Neural Network

BMR: Basic Metabolic Rate

COM: Center of Mass

DOF: Degree of Freedom

EA: Evolutionary Algorithm

FDOF: Functional Degree of Freedom

GA: Genetic Algorithm

GAGA: Genetic Algorithms Gait Analysis

GP: Genetic Programming

GRF: Ground Reaction Force

GUI: Graphical User Interface

IK: Inverse Kinematics

KLAW: Keyframe-less Animation of Walking.

L-Systems: Lindenmayer Systems

LD: Left Down

LRD: Left when Right Down

LROM: Limb Range of Motion

LRU: Left when Right UP

LU: Left Up

M: Mean

PAR: Parameterized Action Representation

PD: Proportional Derivative

RD: Right Down

RLD: Right when Left Down

RLU: Right when Left Up

RMS: Root Mean Square

ROM: Range of Motion

RU: Right Up

SAN: Sensor Actuator Network

SCA: Sense Control Action

SD: Standard Deviation

SIMM: Software for Interactive Musculoskeletal Modeling

SR: Stimulus Response

ZMP: Zero Moment Point

BIBLIOGRAPHY

Alexander, R. M., & Jayes, A. S. (1983). A dynamic similarity hypothesis for the gaits of quadrupedal mammals. *Journal of Zoology, London, 201*, 135-152.

Allbeck, J., & Badler, N. (2002). Toward Representing Agent Behaviors Modified by Personality and Emotion. *Proceedings of the First Annual Joint Conference on Autonomous Agents and Multiagent Systems*, 202-207.

Azevedo, C. (2001). On the Interaction between the Human and The Robot in Bipedal Walking. *Proceedings of International Symposium on Mobile, Climbing and Walking Robots-Karlsrhue-Germany*.

Badler, N., Bindiganavale, R., Bourne, J., Palmer, M., Shi, J., & Schuler, W. (1998). A Parameterized Action Representation for Virtual Human Agents. *Workshop on Embodied Conversational Characters*.

Badler, N., Webber, B., Becket, W., Geib, C., Moore, M., Pelachaud, C., et al. (1995). Planning and parallel transition networks: Animation's new frontiers. *Computer Graphics and Applications: Proc. Pacific Graphics, 95*, 101–117.

Bongard, J. C., & Pfeifer, R. (2002). A Method for Isolating Morphological Effects on Evolved Behaviour. *Proceedings of the Seventh International Conference on the Simulation of Adaptive Behavior*, 305-311.

Bruderlin, A., & Calvert, T. W. (1989). Goal-directed, dynamic animation of human walking. *Proceedings of the 16th annual conference on Computer graphics and interactive techniques*, 233-242.

Choi, M. G., Lee, J., & Shin, S. Y. (2003). Planning biped locomotion using motion capture data and probabilistic roadmaps. *ACM Transactions on Graphics (TOG), 22*(2), 182-203.

Chung, S., & Hahn, J. K. (1999). Animation of Human Walking in Virtual Environments. *Proceedings of Computer Animation, 99*, 4-15.

Coombs Jr, W. P. (1978). Theoretical Aspects of Cursorial Adaptations in Dinosaurs. *The Quarterly Review of Biology, 53*(4), 393-418.

Delaney, B. (1998). On the trail of the shadow woman: the mystery of motion capture. *Computer Graphics and Applications, IEEE, 18*(5), 14-19.

Daley, M. A., Felix, G., & Biewener, A. A. (2007). Running stability is enhanced by a proximo-distal gradient in joint neuromechanical control. *Journal of Experimental Biology, 210*(Pt 3), 383-394.

Delp, S. L., & Loan, J. P. (1995). A graphics-based software system to develop and analyze models of musculoskeletal structures. *Computers in Biology and Medicine, 25*(1), 21-34.

Delp, S. L., & Loan, J. P. (2000). A computational framework for simulating and analyzing human and animal movement. *Computing in Science & Engineering [see also IEEE Computational Science and Engineering], 2*(5), 46-55.

Faloutsos, P., van de Panne, M., & Terzopoulos, D. (2001). *Composable controllers for physics-based character animation.* New York: ACM Press.

Farley, C. T. (1997). Mechanics of locomotion in lizards. *Journal of Experimental Biology, 200*(16), 2177-2188.

Fotosearch. (2005, May 14). Komodo dragon stock photos and images. Retrieved November 13, 2007, from http://www.fotosearch.com/photos-images/komodo-dragon.html.

Fotosearch. (2007, March 12). Komodo dragon walking on ground. Retrieved November 21, 2007, from http://www.fotosearch.com/DVA002/006-0044.

Giang, T., Mooney, R., Peters, C., & O'Sullivan, C. (2000). Real-Time Character Animation Techniques. *Image Synthesis Group Trinity College Dublin, Technical Report TCD-CS-2000-06.*

Golubovic, D., & Hu, H. (2002). A hybrid evolutionary algorithm for gait generation of Sony legged robots. *Proceedings of the 28th Annual Conference of the IEEE Industrial Electronics Society, 4*, 2593- 2598.

Goslow, G. E. (1981). Electrical activity and relative length changes of dog limb muscles as a function of speed and gait. *Journal of Experimental Biology, 94*(1), 15-42.

Griffin, T. M., Main, R. P., & Farley, C. T. (2004). Biomechanics of quadrupedal walking: how do four-legged animals achieve inverted pendulum-like movements? *Journal of Experimental Biology, 207*(20), 3545-3558

Gritz, L., & Hahn, J. K. (1997). Genetic Programming Evolution of Controllers for 3-D Character Animation. *Genetic Programming 1997: Proceedings of the 2nd Annual Conference*, 139-146.

Grzeszczuk, R., & Terzopoulos, D. (1995). Automated learning of muscle-actuated locomotion through control abstraction. *Proceedings of the 22nd annual conference on Computer graphics and interactive techniques*, 63-70.

Grzeszczuk, R., Terzopoulos, D., & Hinton, G. (1998). Neuroanimator: Fast neural network emulation and control of physics-based models. SIGGRAPH 98 Conference Proceedings. *Annual Conference Series*, 9–20.

Henderson, D. M. (2006). Burly gaits: centers of mass, stability, and the trackways of sauropod dinosaurs. *Journal of Vertebrate Paleontology, 26*(4), 907-921.

Herr, H. M., Huang, G. T., & McMahon, T. A. (2002). A model of scale effects in mammalian quadrupedal running. *Journal of Experimental Biology, 205*(7), 959-967.

Hildebrand, M. (1980). The Adaptive Significance of Tetrapod Gait Selection. *American Zoologist, 20*(1), 255-267.

Hill, A. V. (1938). The Heat of Shortening and the Dynamic Constants of Muscle. *Proceedings of the Royal Society of London. Series B, Biological Sciences, 126*(843), 136-195.

Hodgins, J. K., Wooten, W. L., Brogan, D. C., & O'Brien, J. F. (1995). *Animating human athletics*. New York: ACM Press.

Holland, J. H. (1992). *Adaptation in natural and artificial systems*. Cambridge, MA: MIT Press.

Hornby, G. S., & Pollack, J. B. (2001). Body-brain co-evolution using L-systems as a generative encoding. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2001)*, 868–875.

Hutchinson, J. R., Anderson, F. C., Blemker, S. S., & Delp, S. L. (2005). Analysis of hindlimb muscle moment arms in Tyrannosaurus rex using a three-dimensional musculoskeletal computer model: implications for stance, gait, and speed. *Paleobiology, 31*(4), 676-701.

Hutchinson, J. R., & Gatesy, S. M. (2006). Beyond the bones. *Nature(London), 440*(7082), 292-294.

Ijspeert, A. J., & Arbib, M. (2000). Visual tracking in simulated salamander locomotion. *Proceedings of SAB'00, From Animals to Animats 6*, 88-97.

Kang, Y. M., Cho, H. G., & Lee, E. T. (1999). An efficient control over human running animation with extension of planar hopper model. *Journal of Visualization and Computer Animation, 10*(4), 215-224.

Koza, J. R. (1990). Genetically breeding populations of computer programs to solveproblems in artificial intelligence. *Proceedings of the Second International Conference on Tools for Artificial Intelligence*, 819-827.

Laszlo, J. F., van de Panne, M., & Fiume, E. (1997). Control of Physically-based Simulated Walking. *Proceedings of IMAGINA*, 231-241.

Lindenmayer, A. (1968). Mathematical models for cellular interaction in development, Parts I and II. *Journal of Theoretical Biology, 18*, 280-315.

Lindenmayer, A. (1974). Adding Continuous Components to L-Systems. *Lecture Notes In Computer Science, 14*, 53-68.

Liu, Z., Gortler, S. J., & Cohen, M. F. (1994). Hierarchical spacetime control. *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, 35-42.

Maciel, A., Nedel, L. P., & Dal Sasso Freitas, C. M. (2002). Anatomy-based joint models for virtual human skeletons. *Proceedings of Computer Animation, 2002*, 220-224.

McKenna, M., & Zeltzer, D. (1990). Dynamic simulation of autonomous legged locomotion. *Proceedings of the 17th annual conference on Computer graphics and interactive techniques*, 29-38.

Muybridge, E. (1887). *Animals in motion [1957 reprint]*. New York: Dover.

Ngo, J. T., & Marks, J. (1993). Spacetime constraints revisited. *Proceedings of the 20th annual conference on Computer graphics and interactive techniques*, 343-350.

Perlin, K., & Goldberg, A. (1996). Improv: a system for scripting interactive actors in virtual worlds. *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 205-216.

Piazza, S. J., & Delp, S. L. (2001). Three-Dimensional Dynamic Simulation of Total Knee Replacement Motion During a Step-Up Task. *Journal of Biomechanical Engineering, 123*, 599-606.

Pike, A. V. L., & Alexander, R. M. N. (2002). The relationship between limb-segment proportions and joint kinematics for the hind limbs of quadrupedal mammals. *Journal of Zoology, 258*(4), 427-433.

Raibert, M. H., & Hodgins, J. K. (1991). Animation of dynamic legged locomotion. *Proceedings of the 18th annual conference on Computer graphics and interactive techniques*, 349-358.

Ray, T. S. (2001). Aesthetically Evolved Virtual Pets. *Leonardo, 34*(4), 313-316.

Reilly, S. M. (1997). Sprawling locomotion in the lizard *Sceloporus clarkii*: quantitative kinematics of a walking trot. *Journal of Experimental Biology, 200*(4), 753-765.

Reilly, S. M. (1998). Locomotion in *Alligator mississippiensis*: kinematic effects of speed and posture and their relevance to the sprawling-to-erect paradigm. *Journal of Experimental Biology, 201*(18), 2559-2574.

Rose, C., Guenter, B., Bodenheimer, B., & Cohen, M. F. (1996). Efficient generation of motion transitions using spacetime constraints. *Proceedings of the 23rd annual conference on Computer graphics and interactive techniques*, 147-154.

Rosenblatt, F. (1958). The perceptron: a probabilistic model for information storage and organization in the brain. *Psychological Review, 65*(6), 386-408.

Sellers, W. I., Dennis, L. A., Wang, W.-J., & Crompton, R. H. (2004). Evaluating alternative gait strategies using evolutionary robotics. *Journal of Anatomy, 204*(5), 343-351.

Sellers, W. I., & Manning, P. L. (2007). Estimating dinosaur maximum running speeds using evolutionary robotics. *Proceedings of the Royal Society B: Biological Sciences, 274*(1626), 2711-2716.

Shaw, L. (2007, July 2). Illustrated Standard of the German Sheppard Dog, The Forehand. Retrieved October 17, 2007, from http://www.shawlein.com/The_Standard/02_The_Forehand/The_Forehand.html

Shaw, L. (2007, July 2). Illustrated Standard of the German Sheppard Dog, The Hindquarters. Retrieved October 17, 2007, from http://www.shawlein.com/The_Standard/05_The_Hindquarters/The_Hindquarters.html

Sims, K. (1991). Artificial Evolution for Computer Graphics. *Computer Graphics, 253*(4), 319-328.

Sims, K. (1994a). Evolving 3D Morphology and Behavior by Competition. *Artificial Life, 1*(4), 353-372.

Sims, K. (1994b). Evolving virtual creatures. *Proceedings of the 21st annual conference on Computer graphics and interactive techniques*, 15-22.

Stevens, K. (2007, November 25). *Apatosaurus*. Retrieved November 28, 2007, from http://www.cs.uoregon.edu/~kent/paleontology/Apatosaurus/2005/index.html

Stevens, K. (2007, November 25). *Triceratops*. Retrieved November 28, 2007, from http://www.cs.uoregon.edu/~kent/paleontology/Triceratops/index.html

Stevens, K. (2007, November 25). *Tyrannosaurus*. Retrieved November 28, 2007, from http://www.cs.uoregon.edu/~kent/paleontology/Tyrannosaurus/index.html

Stevens, K. A., Larson, P., Wills, E. D., & Andersen, A. (2008). Rex, sit: Digital modeling of Tyrannosaurus rex at rest. In P. Larson & K. Carpenter (Eds.), *Tyrannosaurus rex: The tyrant king*. Bloomington, IN: Indiana University Press.

Sun, H. C., & Metaxas, D. N. (2001). Automating gait generation. *Proceedings of the 28th annual conference on Computer graphics and interactive techniques*, 261-270.

Taylor, T. (2000). Artificial life techniques for generating controllers for physically modelled characters. *Proceedings of the First International Conference on Intelligent Games and Simulation (GAME-ON 2000).*

Thompson, S., & Holmes, R. (2007). Forelime stance and step cycle in *Chasmosaurus Irvinensis* (Dinosaurua: Neoceratopsia). *Palaeo-Electronica*(10.1.5A).

Tolani, D., Goswami, A., & Badler, N. I. (2000). Real-time inverse kinematics techniques for anthropomorphic limbs. *Graphical Models, 62*(5), 353-388.

Torkos, N., & van de Panne, M. (1998). Footprint--based Quadruped Motion Synthesis. *Graphics Interface*, 151-160.

van de Panne, M. (2000). Control for simulated human and animal motion. *IFAC Annual Reviews in Control*(24), 189-199.

van de Panne, M., & Fiume, E. (1993). Sensor-actuator networks. *Proceedings of SIGGRAPH, 93*, 335–342.

Vukobratovic, M., & Juricic, D. (1969). Contribution to the synthesis of biped gait. *IEEE Trans Biomed Eng, 16*(1), 1-6.

Witkin, A., & Kass, M. (1988). Spacetime constraints. *Proceedings of the 15th annual conference on Computer graphics and interactive techniques*, 159-168.

Wolff, K., & Nordin, P. (2003). Learning biped locomotion from first principles on a simulated humanoid robot using linear genetic programming. *Proceedings of the Genetic and Evolutionary Computation Conference (GECCO-2003)*, 12-16.

Wright, A. H. (1991). Genetic algorithms for real parameter optimization. *Foundations of Genetic Algorithms, 1*, 205–218.

Wyeth, G. K., & Yik, D. T. F. (2003). Evolving a locus based gait for a humanoid robot. *IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS 2003), 2*, 1638-1643.

Zhang, R., & Vadakkepat, P. (2003). An evolutionary algorithm for trajectory based gait generation of biped robot. *Proceedings of the International Conference on Computational Intelligence, Robotics and Autonomous Systems*.