Disney Meets Darwin

An Evolution-based Interface for Exploration and Design of Expressive Animated Behavior

by

Jeffrey John Ventrella

MFA Computer Graphics/Video
Syracuse University, 1987

BFA Art Education/Art History
Virginia Commonwealth University, 1984

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning,
in Partial Fulfillment of the requirements of the degree of

MASTER OF SCIENCE in MEDIA ARTS AND SCIENCES
Massachusetts Institute of Technology
June, 1994

Signature of Author_____

Program in Media Arts and Sciences
May 6, 1994

Certified by_____

Ronald MacNeil
Principal Research Associate, Program in Media Arts and Sciences
Thesis Supervisor

Accepted by_____

Stephen Benton
Chairperson, Departmental Committee on Graduate Students
Program in Media Arts and Sciences

Disney Meets Darwin

An Evolution-based Interface for Exploration and Design of Expressive Animated Behavior

by

Jeffrey John Ventrella

Submitted to the Program in Media Arts and Sciences,
School of Architecture and Planning, on May 6, 1994
in Partial Fulfillment of the requirements of the degree of

MASTER OF SCIENCE in MEDIA ARTS AND SCIENCES

ABSTRACT

In this thesis I describe a computer animation system called the Character Evolution Tool. It was developed both as a prototype system for animators and graphic designers, and as a testbed for the applicability of genetic algorithms in the design process. Although the focus is primarily on physically based articulated characters, these are considered as only a sub-class in the class of all graphical objects which can exhibit expressive motion behavior, termed, "behavior objects." The Character Evolution Tool employs a genetic algorithm for the automatic evolution of goal-oriented behavior in animated graphics, with an overlay of interactive evolution. This overlay affords the user of the system the ability to encourage expressivity and communicative behaviors in the animated characters, as they evolve towards some otherwise pre-defined objective goal. Central to this system is an experimental technique for guiding the direction of evolution by way of a gesture drawn into the scene by the user. This gesture constitutes a novel approach to defining the genetic algorithm's objective fitness function, in that the characters are encouraged to emulate properties of the gestured motion, thereby assuming some of the expressive qualities that the user has specified.

Thesis Supervisor: Ronald MacNeil

Title: Principal Research Associate, Program in Media Arts and Sciences

This work was supported in part by Paws, Inc., USDOT, and NIF.

Disney Meets Darwin

An Evolution-based Interface for Exploration and Design of Expressive Animated Behavior

by

Jeffrey John Ventrella

The following people served as readers for this thesis:

Joe Marks, Ph.D.

Research Scientist, Mitsubishi Electric Research Laboratories, Inc.

Lecturer on Computer Science, Harvard University

Mitchel Resnick, Ph.D.

Assistant Professor, Program in Media Arts and Sciences

ACKNOWLEDGMENTS

Thanks to Suguru Ishizaki for his continual inspiration and advice

I would also like to thank:

Ron MacNeil for the many interesting ideas and great conversations, which helped to form this thesis

Muriel Cooper for creating the Visible Language Workshop and the original vision, which has made this work possible

Louie Weitzman for his friendly advice and survival tips, and for the fuzzy kitties—Mies and Bou

All the other folks of the Visible Language Workshop who have contributed ideas and debugging tips: Dave, Karen, Didier, BC, Craig, Allen, Maia, Xiaoyang, Lisa, Yinyin, Ishantha, Earl, Robin, Christa, John, and of course, Sukey

Mitch Resnick for his ideas about ideas about evolution

Joe Marks for his sound advice on the use of genetic algorithms in character animation

Tim Bird of Paws, Inc., for an inspired mixture of algorithms and laughs

Linda Peterson for helping me with organization, and much appreciated consultation

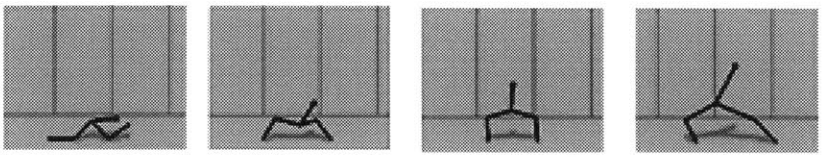
Brother Philip for the good laughs over the phone, which helped to reduce the pains in the frontal lobe

Nuala for a loving background to all the madness

Finally, and most of all, to my Mom

TABLE OF CONTENTS

1. INTRODUCTION



In this thesis I describe a computer animation tool I have developed and the reasons behind its development. It is called the Character Evolution Tool. It serves both as a prototype tool for animators and designers, and as a testbed for the applicability of evolutionary algorithms in computer animation. The work is built on an assumption that not just animated characters but all kinds of graphical entities can have a "body language." It also assumes that this body language is something that can *evolve* over time to become progressively more effective in *expressing* something, and having a *style* of moving about. These qualities are still not easily achieved in most computer-based character animation systems—yet they are abundant in classic cel animated cartoons.

To imbue characters with personality, humor, and style is a goal for many animators. This research can be described as an ongoing quest which began while I was exploring simple physically based human-like figures for animation. I wanted my figures to be able to move on their own, using internal forces, as a starting point for higher levels of control. This basic motivation has been the impetus for a number of other systems developed (Badler et al., 91). For enhancing humorous and stylistic qualities in my figures, I found that I required an added level to the techniques developed thus far: a highly interactive interface allowing enhanced communication between the animator and the animated.

Genetic algorithms (Holland, 75) have been used recently for evolving goal-directed behavior in physically based, articulated figures (Ngo and Marks, 93). (Refer ahead to "Background Research" for a quick definition of genetic algorithms). This work has successfully shown that genetic algorithms are effective in optimizing their motions for locomotion and other objectively defined goals. But it has not, however, focused on a means to imbue the figures with expressive characteristics and stylistic variations by way of a designer's interactive contribution to the evolutionary process. I have proposed to address the problem of designing humor and body language in animated characters with an hypothesis: expressivity, not just objectively-defined behaviors, can be the product of progressive refinement. I have attempted to test this by combining the automatic optimizing capabilities of a genetic algorithms with direct interactive tools, and allowing a

blending of automated and user-guided evolution. This system can be seen then as extending the standard genetic algorithm in that it adds multiple levels of interactivity to the automatic process, to achieve higher levels of control. This interactivity is manifest on two basic levels:

1) interactively guiding evolution
2) gesturing to demonstrate motion

Expressivity Through Guided Evolution

In this system, the means for deriving expressive behavior in these characters is biologically inspired. It uses an evolutionary algorithm which allows identifiable behavioral qualities in animated graphics to emerge over time. The term, "expressive" as used here is defined as the ability of a graphical object, through MOTION, to *evoke* information, ideas, or feelings, for the sake of communication, humor, or aesthetics.

As a first glimpse of the varieties of "characters" one can evolve with this system, Figure 1 illustrates three varieties (swarms, blinkers, and articulated figures).

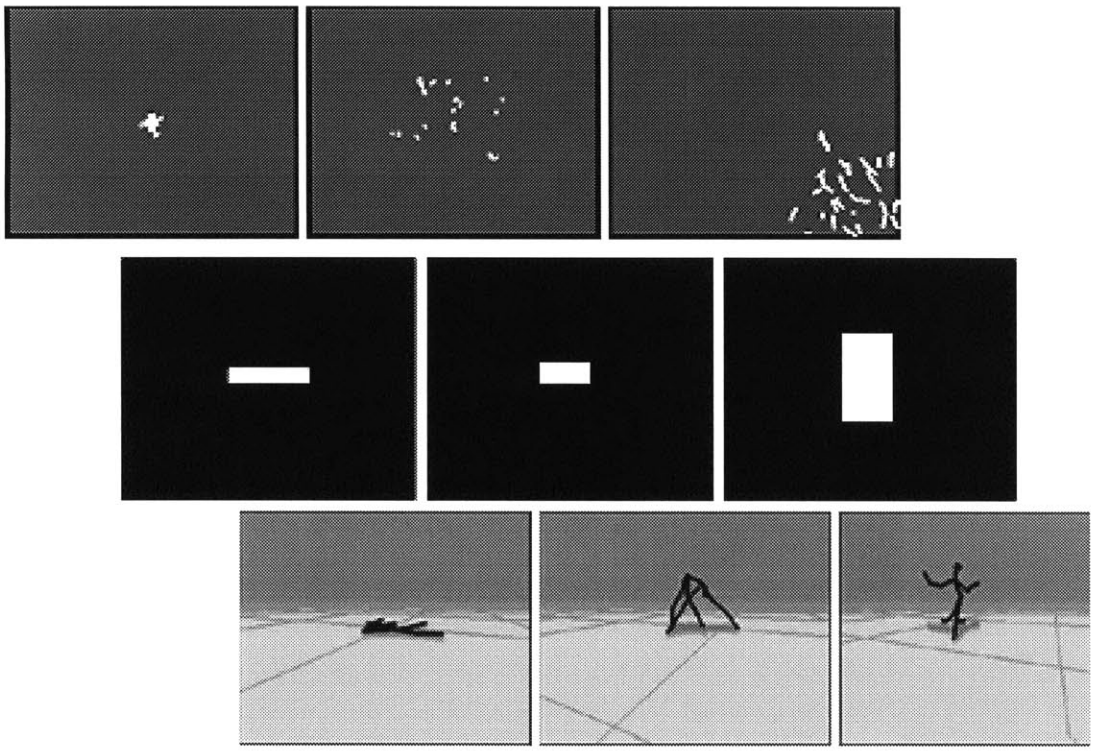


Figure 1 Three stages of evolution for three of the classes of characters available in this system: swarms, blinkers, and articulated figures.

This illustration shows three stages of each character's evolution towards more expressive, and otherwise useful, forms and motions. The swarms become more life-like, the blinkers become more effective attention-grabbers and acquire optimal blinking rates (in this illustration, only shape and size can be shown), and the articulated figures exhibit locomotion or humorous dance-like movements.

The graphical objects I have focused on in this research are animal-like articulated stick figures with autonomous motion, but the concepts and techniques from this class of graphical objects have been carried over to a larger class of graphical objects which can exhibit behavior (they can change states over time). This includes graphical interface widgets, and other object-oriented graphical agents which can have behavior. They are called, in this thesis, "behavior objects." In Figure 2, the generic concept of a "character" is illustrated, as being any behavior object which can exhibit dynamic behavior for purposes of expressivity.

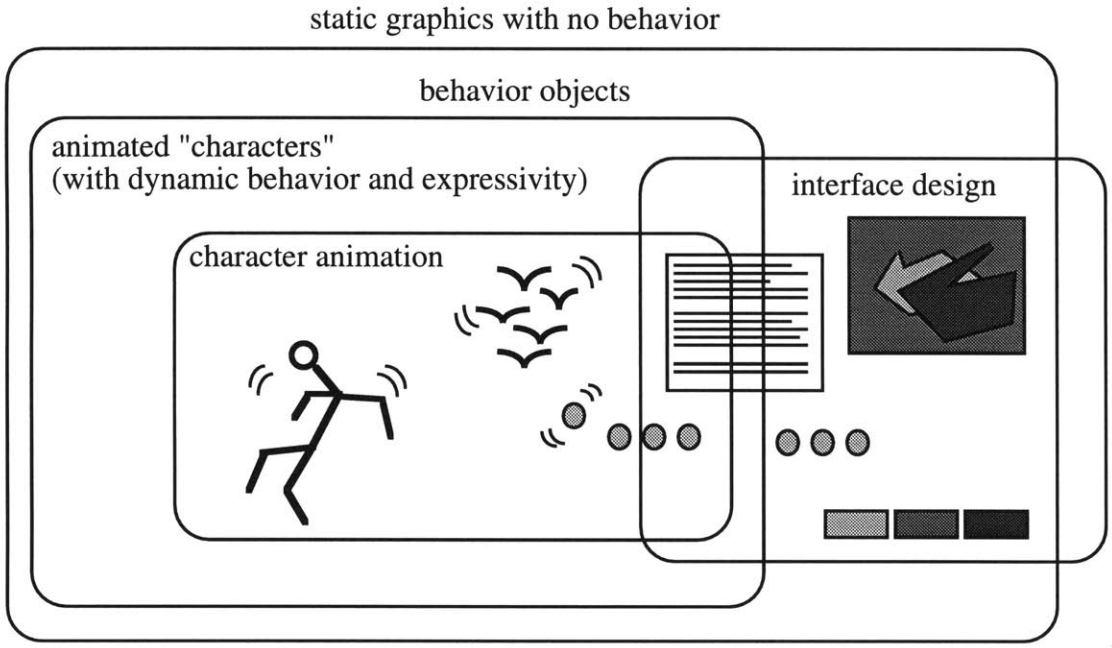


Figure 2 The general concept of a "character" as defined in the Character Evolution Tool—a graphical object with expressive dynamic behavior.

Thus the class of objects known as *animated characters* can be expanded to include all graphical objects which can exhibit motion behavior. In this thesis, I sometimes reference this general notion of *character*, in order to place my work in the larger context of

communication graphics. Primarily I will focus on the articulated figures, and a set of tools for directing this class of behavior objects, as the focus of this research.

Computer Animation Not Alive Enough

Physically based animated characters whose behaviors adapt through the use of a genetic algorithm have been shown to demonstrate realistic behaviors such as locomotion. But they are often devoid of the expressive content which makes traditional hand-drawn characters so effective in comparison. In these systems, the *science* of the *design* of animated characters is not yet useful for the *art*.

Animation, seen as an art, should be concerned more with motion-based expression and communication of certain ideas and feelings than on mere simulation of physics or animal behavior, which has been a major focus of most research in this area. In general, animators have a story to tell, and it is usually not totally reliant on physical realism or efficiency of motion within spacetime constraints. The Disney tradition of character animation reminds us that physics and biological realism are routinely violated for the purposes of character personality and narration. Systems which incorporate genetic algorithm's for evolution of behavior in articulated figures solve the *physical* problems of optimizing behavior, but do not afford the *optimization of expressivity*, above and beyond the constraints from evolutionary fitness pressures. Optimization of expressivity cannot easily be accomplished with straightforward fitness functions. This thesis assumes that it requires some feedback with a human in the optimizing loop. Animation systems would benefit from tools which support expressive communication between the animator and the animated, as things evolve, while simultaneously keeping behaviors within objectively defined constraints.

"Move Like This"—Survival of the Fittest

An experimental method for interactively demonstrating motion by gesturing into the scene has been implemented in the Character Evolution Tool. It will be discussed as a potential new contribution to character animation research. Since this system is based on an evolutionary paradigm, the gesture tool essentially allows a free form line which is drawn into the scene to become a form of evolutionary pressure which drives a character's motions to assume attributes of the gesture. Two gesture matching algorithms have been implemented which compare the input gesture with the character's motions. Figure 3 offers a cartoon explanation of the concept behind the gesture tool.

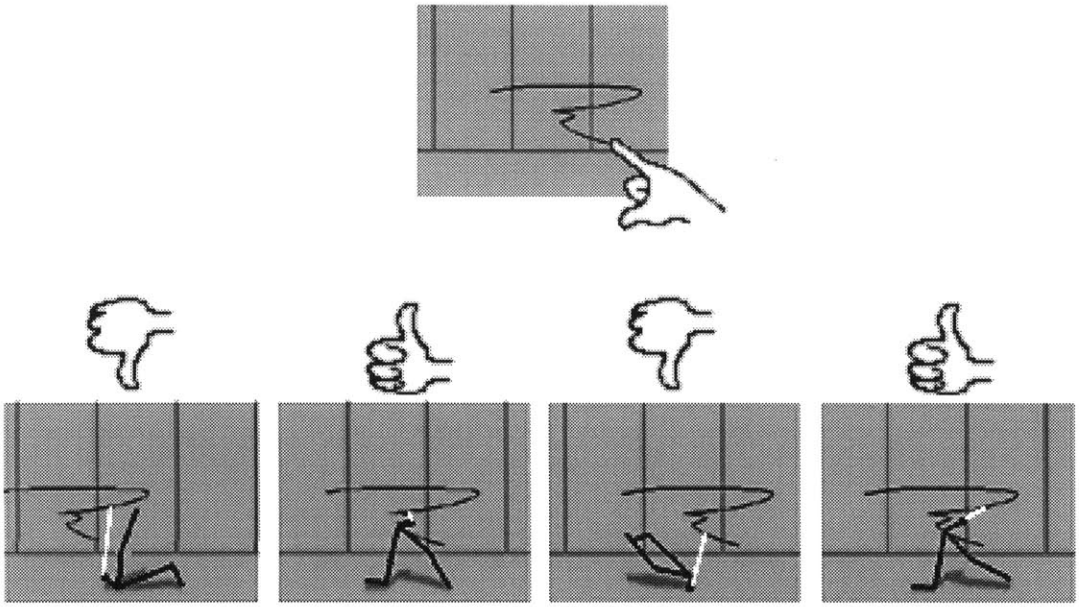


Figure 3 A gesture is drawn by hand, then the individual characters of a population are evaluated according to how closely a body part (the head) follows the gesture, through time. This evaluation is used as a fitness criterion for evolution of behaviors in the population. Over generations, the motions begin to mimic the motion of the gesture.

Evolution and Design

Underlying this research is the idea that Darwinian Evolution and Design, often considered antithetical processes, can be brought together into one coherent system. We often refer to Design as a top-down organizational process, involving decision and planning, and we consider Evolution to be a bottom-up, distributed process, involving chance. But creative design (as viewed in this thesis) generally involves a certain amount of bottom-up organization and chance. The idea in this thesis is that computational evolution can augment the design process for this reason. Genetic algorithms are good at searching for optimal solutions in arbitrarily large multiparameter spaces. Design can also be said to be an activity of searching a large problem space for solutions. Both make use of experimentation, both make use of iterative evaluation. These kinds of notions are much more thoroughly explored by Hybs and Gero (92).

In creative design, especially in the earlier stages of the design process, designers are not always completely aware of what they are making or how to go about doing it. Serendipity often plays a role in the process, and one generally improves the working design iteratively. In short, the act of designing itself is an evolutionary process. Considering genetic algorithms as *serendipity enhancers* (chance mutations can beget successful strategies), we

can see how a process based on evolution can be merged with a design tool—if the interface is carefully constructed to support this. I have attempted to develop an interface which brings these processes together, within the context of animated graphics.

2. BACKGROUND AND RELATED WORK

Autonomous Character Animation

The fine art of character animation has evolved from what it was in the classic days of Disney, Warner Brothers, and Fleischer. The introduction of computer software in the animation studio has influenced the industry and the way we think about creating and even watching an animated character. Animation was originally a craft whose structure was based primarily on the animation cel—the 1/30th-of-a-second picture frame—as the main building block. This is a direct outcome of the physical properties of movie film.

The introduction of computational techniques which simulate the physics of interacting bodies and the motor control systems of animals has introduced new systems for generating motion, and has brought into the art of animation the concept of autonomy. In these *task level* animation systems, the animator only needs to specify general goals and the character (as an autonomous agent) takes care of the many details necessary to accomplish this task (Zeltzer, 91). This is still more the realm of research than actual applied use, but some predict that the art of animation may, in the future, appear more like the direction of actors in a movie than the drawing of hundreds of picture frames.

Spacetime Constraints

In the field of character animation research, there have been a number of developments in methods for generating goal-directed behavior in characters represented as articulated skeletal figures (Sims, 87), (Cohen, 92), (Ngo and Marks, 93), (van de Panne and Fiume, 93). These figures are modeled in virtual physical worlds for realism, and their internal motions can adapt, within the constraints of that physical world, to obey user-specified constraints, for the purposes of locomotion and other specified behaviors. This approach to generating motion is often referred to as the *spacetime constraints* paradigm (Witkin and Kass, 88). The spacetime constraints paradigm is a technique which aims to automate some of the tasks of the animator, whereby dynamics associated with the physics of motion are left up to a physically based model, and the autonomous motions of articulated bodies are automatically optimized within the system. In these systems, the animator tells the character where to go and when to get there, for instance, and the spacetime constraint system automatically computes the optimal motions of the character for getting there at the right time. Traditional animation concepts such as squash and stretch, anticipation, and follow-through, have been shown to emerge through the spacetime constraints system.

Motion Control

The spacetime constraints approach assumes that the articulated figure has some ability to change its internal structure (like the angles in its joints) in such a way as to affect goal-directed behavior. Thus, not only must some physics model be used, but there must also be some modeling of a motor control system in the character. There have been many kinds of motor control used in this regard. Many, as one might expect, are biologically inspired, and use theories from neurology and physiology. The virtual roach developed by McKenna (90), for instance, uses a gait controller which employs a series of oscillators which generate stepping patterns, analogous to a system observed in actual roaches. The articulated figures developed by Ngo and Marks (93), van de Panne and Fiume (93), and others, use stimulus/response models which allow the characters to sense their relation to the environment, and to move their parts accordingly, each individual action being the direct result of stimulus. Typical senses used in these models include proprioceptive senses of joint angles, and contact with the ground surface, for a number of body parts. Responses typically involve changing various joint angles. Once a stimulus/response model is created, exactly *what* kinds of responses should be activated by *what* sensors for goal-directed behavior is a difficult problem, and can be a difficult design task. This is where the evolutionary algorithms have proven useful.

The Genetic Algorithm

It may not be too surprising that the genetic algorithm should have entered into the domain of autonomous character animation research—it is biologically inspired, and it is good at dealing with large search spaces (such as finding the best parameters for locomotion). Below I will explain the primary concepts of the genetic algorithm.

The genetic algorithm (GA) (Holland, 75) (Goldberg, 89) is a searching and optimization technique derived from the mechanics of Darwinian evolution. It operates on a population of individuals (potential solutions to a problem), updating the population in parallel, over many generations. In the case of this research, a good individual, or *solution*, is a setting of motion attributes which cause a character to perform some desired behavior. As the population is updated in the GA, the better (or more "fit") individuals pass on more of their characteristics to the individuals in the next generation, while the less fit individuals pass on less. Since individuals reproduce through sexual reproduction, offspring have the potential to inherit good traits from two fit parents, taking the best of both worlds and thereby acquiring even better traits. The result is that the population as a whole progressively improves over a series of generations.

Individuals within a population are represented in the GA as strings (in the classic GA they are bit strings of 1's and 0's but strings have also been used which consist of other element such as real numbers or integers). A string is sometimes referred to as a *chromosome*, and the elements in the string are sometimes referred to as *genes*. The chromosome, taken as an encoded representation of an individual solution, is called the individual's *genotype*. Each gene in the genotype influences some attribute or attributes in the individual (with the whole set of attributes comprising the *phenotype*). Thus, the phenotype is the expression of the genotype. The fitness of an individual is determined by an objective fitness function—an evaluation criterion which measures some feature, not of the genotype, but of the phenotype. Figure 4 illustrates this concept—that the phenotype is the representation which is evaluated, and that the genotype is the representation which is affected by the evaluation.

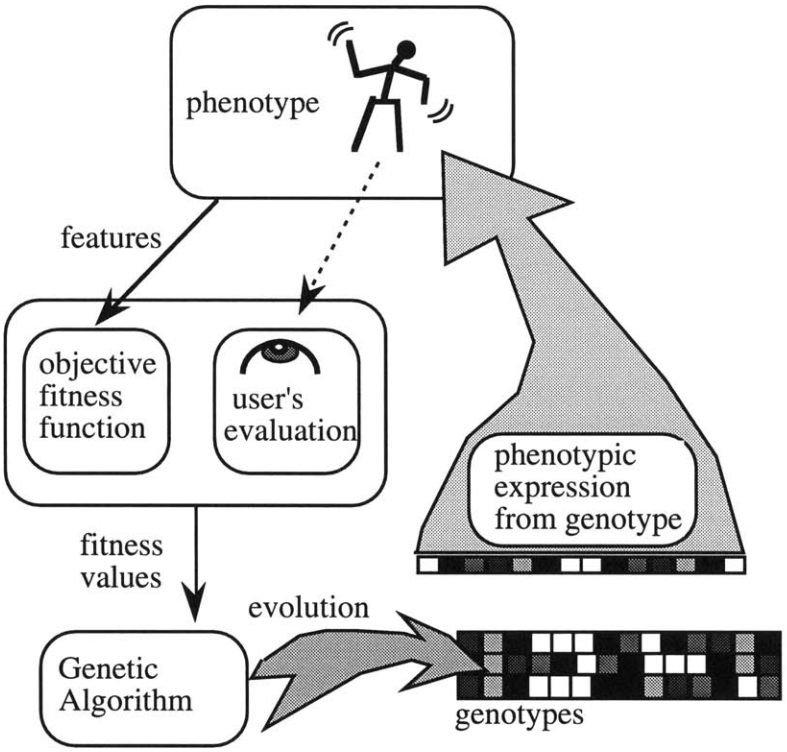


Figure 4 Schematic illustrating a basic concept in genetic algorithms—that there are two components to a representation: the phenotype and the genotype. The genotype is an encoded set of parameters that determine attributes in the phenotype. The phenotype is evaluated by an objective fitness function or a human evaluator, and the genotype is affected by this evaluation, through the operators of the genetic algorithm.

Many varieties of the GA have been implemented, but they all have some basic things in common. A typical GA works like this:

Initialize An initial population of genotypes is created with random settings of gene values.

Evaluate Each genotype's resulting phenotype is evaluated for fitness.

Select Pairs of genotypes are chosen randomly from the population for mating—with the chances of being chosen proportional to fitness.

Crossover These genotypes mate via crossover: they produce one or two offspring genotypes, which inherit randomly sized, alternating "chunks" of gene sequences from each parent.

Mutate Each offspring's genes have a chance of being mutated (isolated genes can be changed randomly—this encourages experimentation in the population, across generations).

Update The offspring genotypes comprise the next generation, and replace the genotypes in the previous one.

Repeat The process is repeated, beginning with the Evaluate step, for a set number of generations, or until the average fitness of the population reaches a desired level.

Most GA's incorporate variations on this form, and experimenters use different settings for things like population size, mutation rate, fitness scaling, and crossover rate (affecting the sizes of the parental chunks which the offspring genotypes inherit).

Artificial Life

Artificial Life is the study of human-made systems which exhibit behaviors that are characteristic of natural organic processes. It complements the traditional biological sciences, which are largely concerned with *analysis*, by attempting to *synthesize* life-like systems (Langton, 91). Life, and the myriad systems which are characteristic of life, are seen as emergent behavior. Many artificial life researchers use genetic algorithms in modeling the evolution of self-organizing phenomena, employing a bottom-up methodology. Reproduction, predator/prey dynamics, primitive communication, locomotion, and functional morphology are among the phenomena that have been modeled. The influence of artificial life concepts can be witnessed in a growing number of software products for exploring evolution and emergent phenomena. These include SimLife and other "Sim" products by *Maxis*, and many others.

Interactive Evolution

Genetic algorithm-based systems which replace the objective function with a human are typically called Interactive Evolution systems. These have been developed by (Dawkins, 86), (Sims, 91), Latham (Todd, 92), (Tolson, 93), (Baker, 93), and others. What distinguishes these systems from other uses of the GA is that they incorporate a *human* fitness function. In these systems, abstract forms (typically) are evolved through the selections of favorable images by a user, through a number of generations of evaluations. These systems support the notion of an "aesthetic search" in which the fitness criteria are based primarily on the visual response of the user. Interactive evolution is useful when fitness is not measurable by way of any known computational evaluation techniques. As an example of an interactive evolution system, Richard Dawkins' *Blind Watchmaker* is illustrated in Figure 5. The software was developed as an accompaniment to the book by the same title, as an interactive illustration of evolutionary principles. In the illustration, the top panel shows one generation of a population of biomorphs from which a user has chosen an individual to spawn a new generation, with genetic mutations for variation. By repeating this action a number of times, one can breed a desired biomorph. This system employs asexual reproduction (one parent per generation). Many other interactive evolution systems, including the Character Evolution Tool, allow for sexual reproduction as well, in which two or more individuals can be chosen from the first generation for mating, to create the next generation.

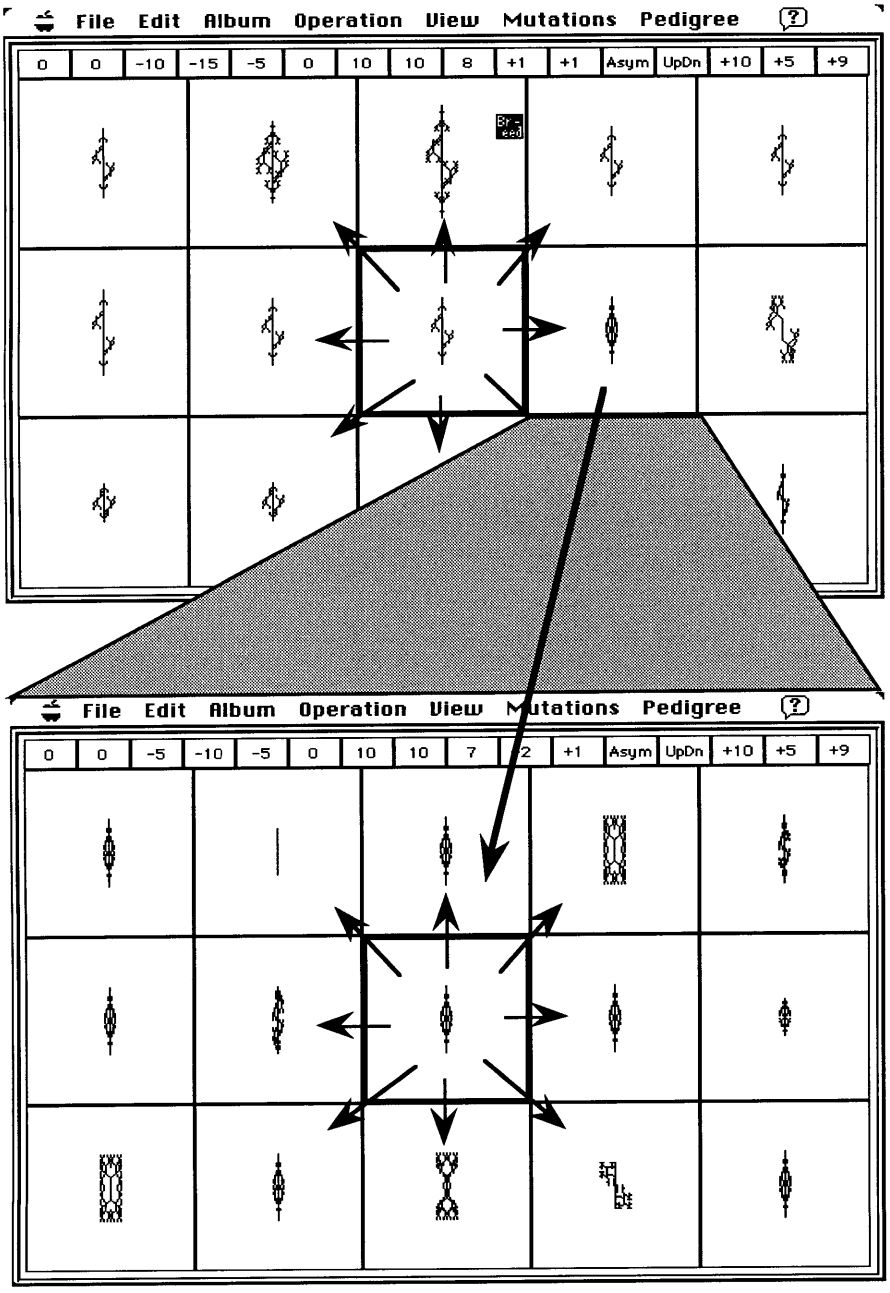


Figure 5 Two views of Dawkins' *Blind Watchmaker* (with a schematic overlay) as an example of an interactive evolution system. The top panel shows a population of biomorphs from which a user has picked an individual to spawn a new population with genetic mutation, shown in the bottom panel.

Gesturing for Animation

Scripting by Enactment techniques have been developed for mapping motions from a human body to an animated 3D computer model (Ginsberg, 83). In these techniques, multiple cameras placed in a room sense diodes attached to a performer's body as he/she moves about—the system merges the camera views to synthesize the 3D model. One can use this technique for quick scripting of expressive movements in an animated character, and it requires no *drawing*.

Still, most animators (at least in the present) are also graphic artists, and are very skilled in the art of expression through drawings. Tools which complement this skill have been explored. Algorithms which extract features from human-made gestures drawn in a computer display have been developed for enhancing the communication between user and computer in Computer-Aided Design research. Donoghue [93] has developed a system in which attributes from gestures such as rate of drawing, direction, "tilt" of the stylus, pressure on the tablet, and other features are parsed and then used for a variety of applications. These applications include rough prototyping for page layout design, and quick scripting of animated character movements. This facet of her system is derived from earlier work by Baecker (69) which demonstrates the use of an input stylus to create a motion path specifying frame-by-frame positions of animated characters. For specifying character animation movements in Donoghue's system, a sketching grammar interprets geometric aspects of the sketch input and temporal aspects of how the character should move. The Character Evolution Tool may be seen as extending Donoghue's work in using gestures to specify motions in characters. The added dimension is that this system allows the characters the ability (as autonomously moving articulated figures) to optimize internal motions to best approach the motion specified by features of the gesture. It includes *adaptation*.

3. APPROACH

In the Disney tradition, animation is *the illusion of life* (Thomas, 81). Character animation research has added to this the *simulation* of life. Animals are complex things which, according to Darwinism, have become the complex things that they are because of evolution. Artificial life researchers take this approach when modeling artificial organisms, in studying emergent self-organizing phenomena. As indicated by the background research I have outlined, the evolutionary techniques which are central to artificial life research have also begun to influence character animation research. I have taken this approach, with an added emphasis on the creation of funny characters with personality, by way of interactive techniques. One might infer from the title of this thesis that the Character Evolution Tool is based on "survival of the cutest." But it is aimed at being more than just this—the qualities that one can extract from a population can have a great range of body language.

Two Levels of Evolution

In this thesis, I consider expressivity to be the product of progressive refinement, which requires that a human be in the GA loop. Figure 6. illustrates a behavior in which the user has affected the course of automatic evolution via an overlay of interactive evolution.

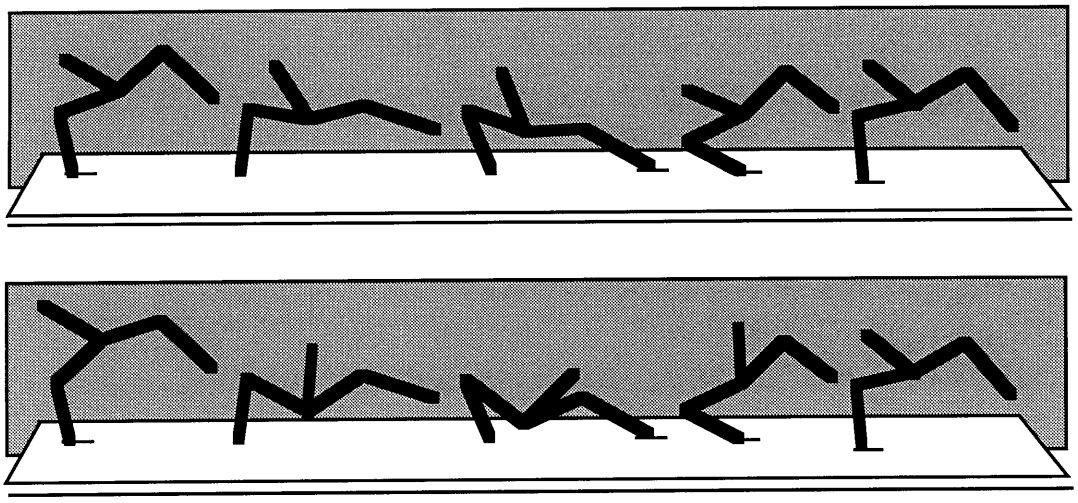
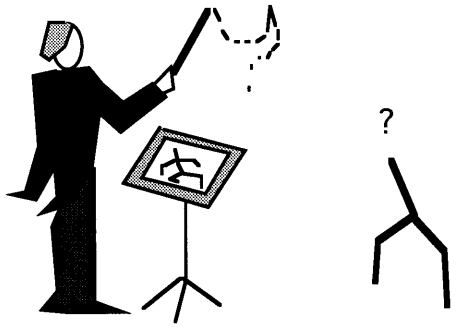


Figure 6 The top panel illustrates a walking pattern which emerged in a population under fitness pressures for locomotion and holding the head high. This character's behaviors have been automatically optimized through these fitness pressures. The bottom panel shows a character from the same population in which a user affected the direction of evolution by favoring ancestors who walked with a desirable style.

In the top panel of the illustration, a walking pattern is shown which emerged in a population evolved under fitness pressures for locomotion and holding the head high. This character is optimized according to these fitness functions. The bottom panel shows a character from the same population in which the user has affected the direction of evolution by favoring ancestors who walked with a particular style. This is accomplished by combining the automatic optimizing capabilities of a GA with interactive tools, and allowing a blending of automated and user-guided evolution. And perhaps most importantly, the *proportion* of user-guided vs. automated evolution can vary.

This can be seen as the overall approach: a system which blends automatically-driven evolution with the critical vision of an interacting human. Essentially, the *source* of evolution at any given time may not be entirely distinguishable to the user—for instance, if an active objective fitness function is encouraging locomotion, through the GA, the user may also be encouraging behaviors that make the locomotion look like swaggering, or skipping, or shuffling. "Shuffling", then, can be the label the user attaches to this behavior. In the final analysis, the user may not care how much a final behavior was influenced by objective functions vs. his/her control. What counts is that a desirable behavior was achieved.



Gesturing

Lying at the conceptual center of this thesis is the gesture tool, which was conceived for the purpose of enhancing the interactive level of genetic algorithms for character animation. The idea was to design a tool which allows an interactive motion from the user to be brought under the grasp of the genetic algorithm, which uses that motion in a specialized fitness function. Although it is an experimental component of this thesis, tests have shown success.

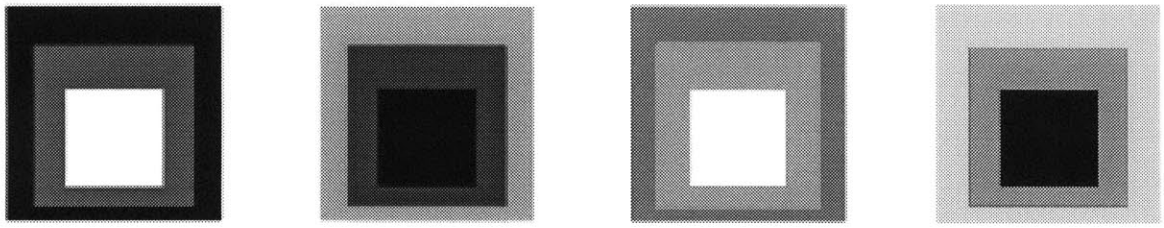
Here the notions of design and evolution are brought together in an experimental marriage. The design part can be described as the line-drawing that is *gestured* into the scene by the user. The evolution part comes into play as that gesture becomes a component in a

specialized fitness function, which affects the evolution of the population. In developing this technique, two kinds of algorithms were implemented which interpret features of the gesture and relate them to some features of the characters as they move. The first algorithm compares the absolute position of a 2D character's moving head to the absolute position of a traveling point on the gesture. I found that this algorithm imposed too harsh a constraint on the evaluation. The second algorithm compares the direction and speed of the character's head motion to the direction and speed of a moving point on the gesture. This algorithm was found to be more flexible in that it allowed comparisons *at a distance*. These two algorithms are described in more detail at the end of the following section.

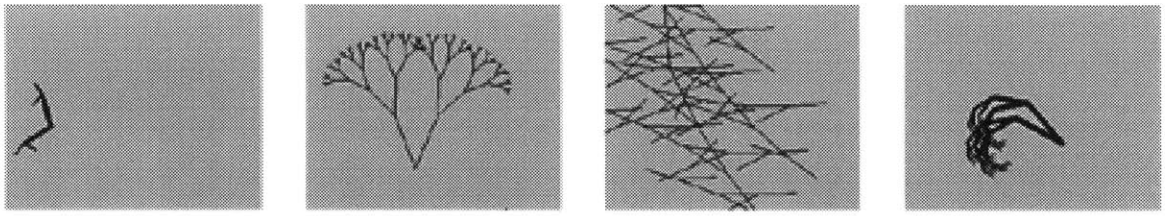
4. NUTS AND BOLTS

Behavior Objects

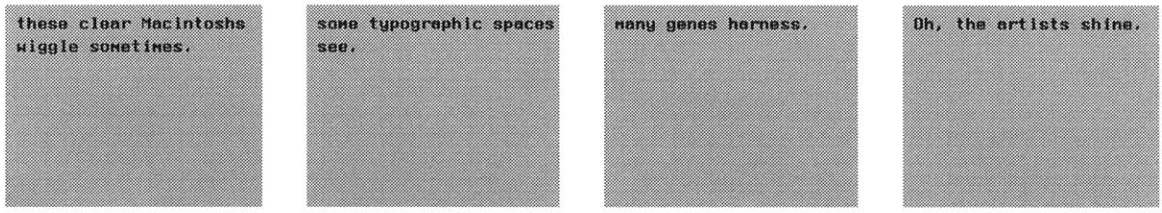
Twelve species of behavior objects have been implemented. I have experimented with many kinds so as to explore the uses of evolution in a number of graphical domains and to generalize my system for purposes of demonstration and research. Most of the behavior objects are graphical, and most are dynamic in the sense that they exhibit animated motion. Below they are each listed and briefly described, accompanied by four sample snapshots to show variety.



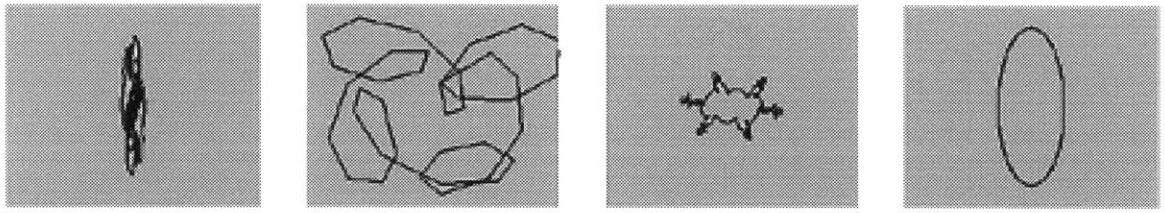
Color images representing Josef Albers' "Homage to the Square" series, with variable color components in the regions in the image



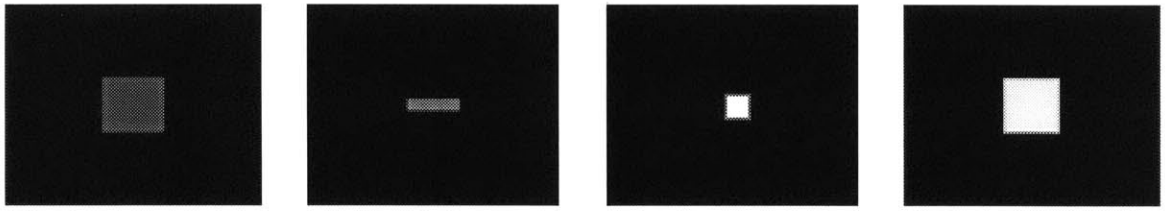
Fractal tree biomorphs similar to Dawkins' Blind Watchmaker biomorphs (Dawkins, 86), but incorporating more degrees of asymmetry



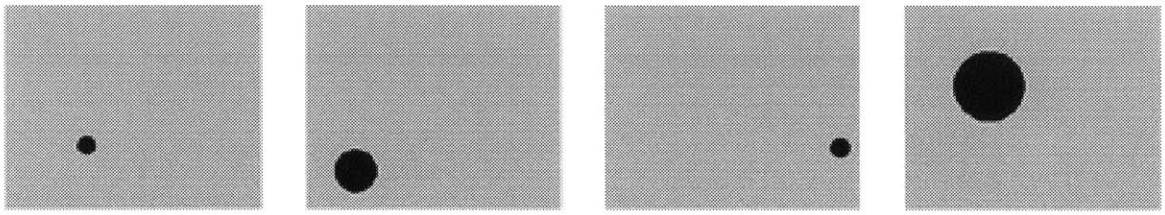
Random grammatically correct sentences in which grammatical variations and vocabulary can be changed (these are the only non-graphical behavior objects)



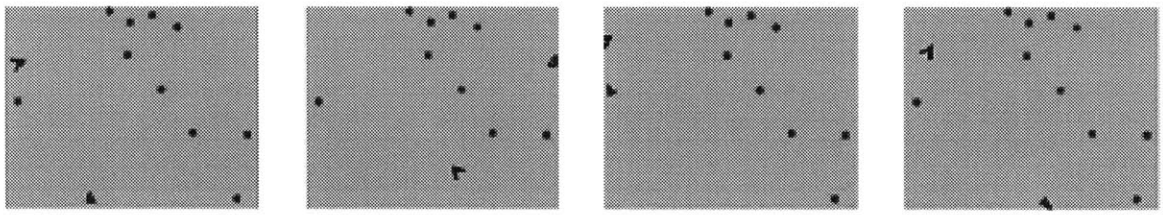
Biomorphs consisting of curves defined by multiple sine functions.



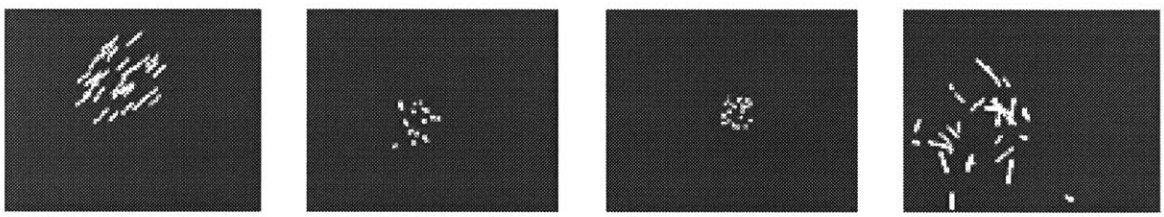
Blinkers - flashing, shifting rectangles (these are meant to be candidates for effective attention-grabbers in a graphical interface)



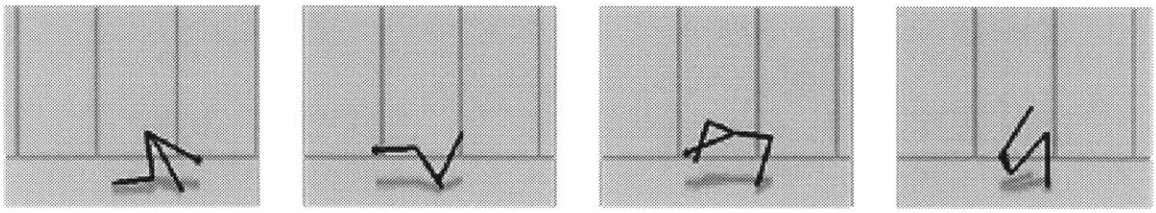
Bouncing balls with simple physics constraints determining their motions, and some autonomy (they can jump in a variety of ways)



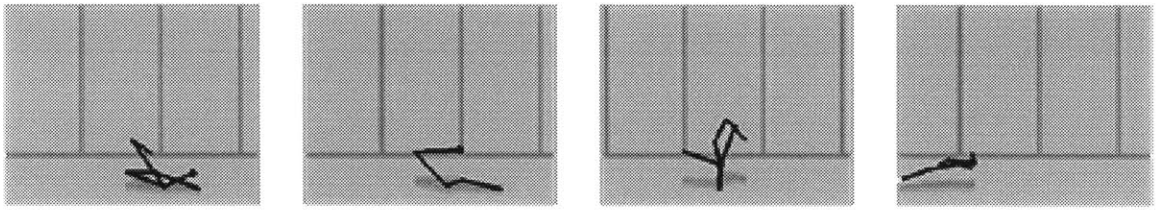
Animal Microworlds consisting of Logo-like turtles, bits of food, and predators—behaviors in the population of turtles can evolve for better eating and to avoid being eaten.



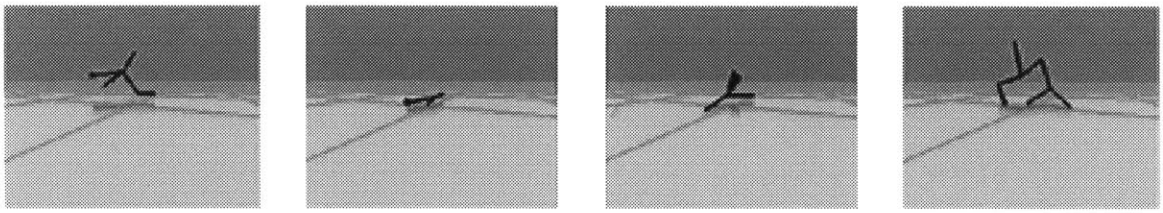
Swarming/schooling/flocking bits which exhibit collective behavior—variations in number and interactive forces create many varieties in the overall dynamic form



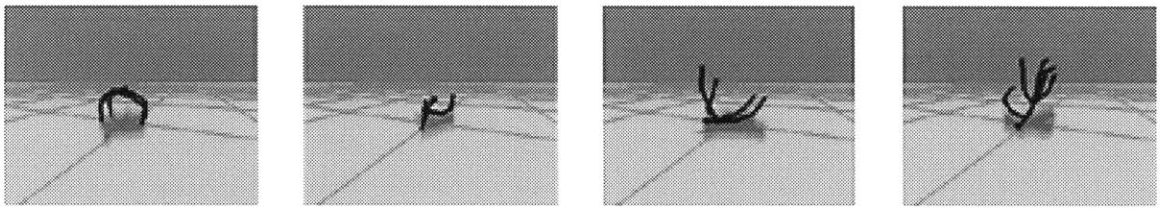
2D articulated stick figures with a fixed topology and variations in rhythms of multiple joint angle motions



2D articulated stick figures with variable morphology as well as motions



3D articulated stick figures with variable morphology and motions



3D articulated stick figures like above except the variations in morphology are constrained to a more realistic segmented animal-like scheme

As indicated by this list, a variety of domains have been explored within this system. What do they all have in common?

- They have behavior (their states can change over time). There are two distinct ways in which their states can change:

1) most of them are animated: they are continually changing their states in the sense that they exhibit autonomous motion. For instance, the bouncing balls are always bouncing.

2) The *nature* of this motion can also change (the attributes which define the kinds of motion). These kinds of changes are the results of genetic operations. For instance, the degree of upward jump in a bouncing ball can change the nature of its motion. Non-motion attributes can change as well, such as the colors in the "Homage to the Square" species.

- Each behavior object has an associated genotype (the behavior object itself is the phenotype).

- They occur in multiples (as a species consisting of a population of individual behavior objects). Another way of thinking about them is that they are *variations on a theme.*

- Individual genes of the genotype can be manipulated in real time resulting in visual transformations of individual attributes, as indicated in figure 7.

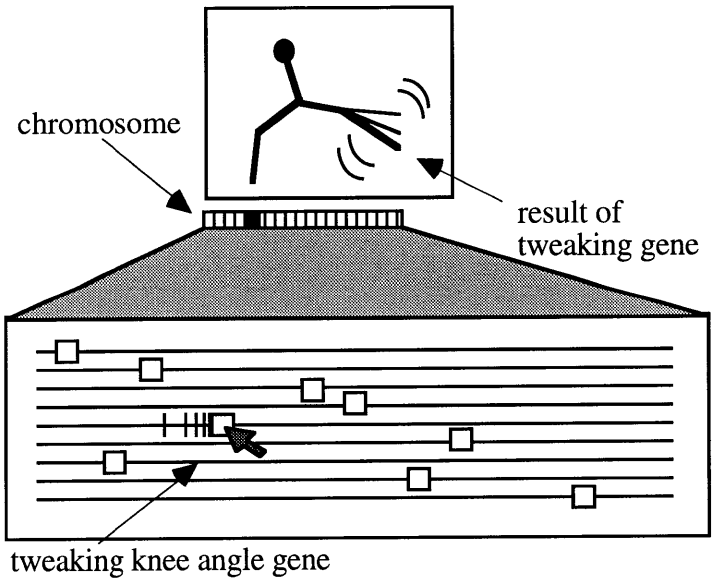


Figure 7 The user can "open up the hood" to see the contents of a chromosome. As the user tweaks an individual gene, the affects of changing this gene can be seen in real time in the phenotype.

The Articulated Figures

The species of behavior objects which I have concentrated on in this thesis are the articulated stick figures. An earlier species which is not represented in this system is seen as a predecessor to the current phylum. It is called "Walker", and the anatomy consists simply of two 3D sticks (legs) joined together to form a hairpin shape (Figure 8). A Walker can autonomously change the angles at which the legs are joined together in a variety of ways. The changes of the angles in each of two angular coordinates in both of the legs are determined by sine functions whose amplitudes, phases, and frequencies can vary from one individual Walker to another (these parameters remain constant during the lifetime of the individual). The parameters are represented as genes and a GA is used to optimize these genes for walking behaviors (Ventrella, 92).

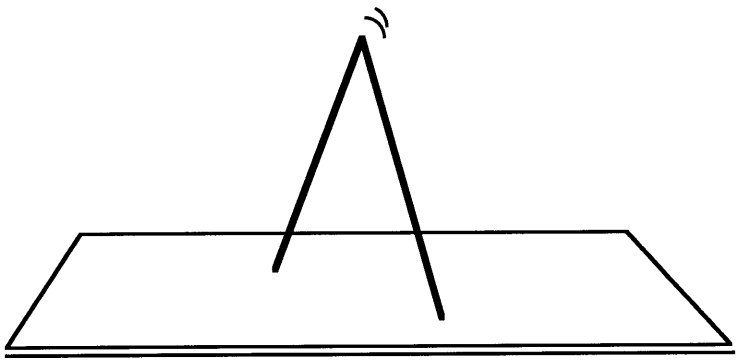


Figure 8 Walker

For this thesis, four new species of articulated figures have been designed. Each species that I have developed is more complex then the previous one. In a very real sense to me, they *evolved* from one to the other as I progressively extended my design of the basic phenotype. The latest species, which I call the "Vertebrates", constitutes the most complex phenotype design in the Character Evolution Tool. Figure 9 shows four variations each of the four example individuals of each species of articulated figures.

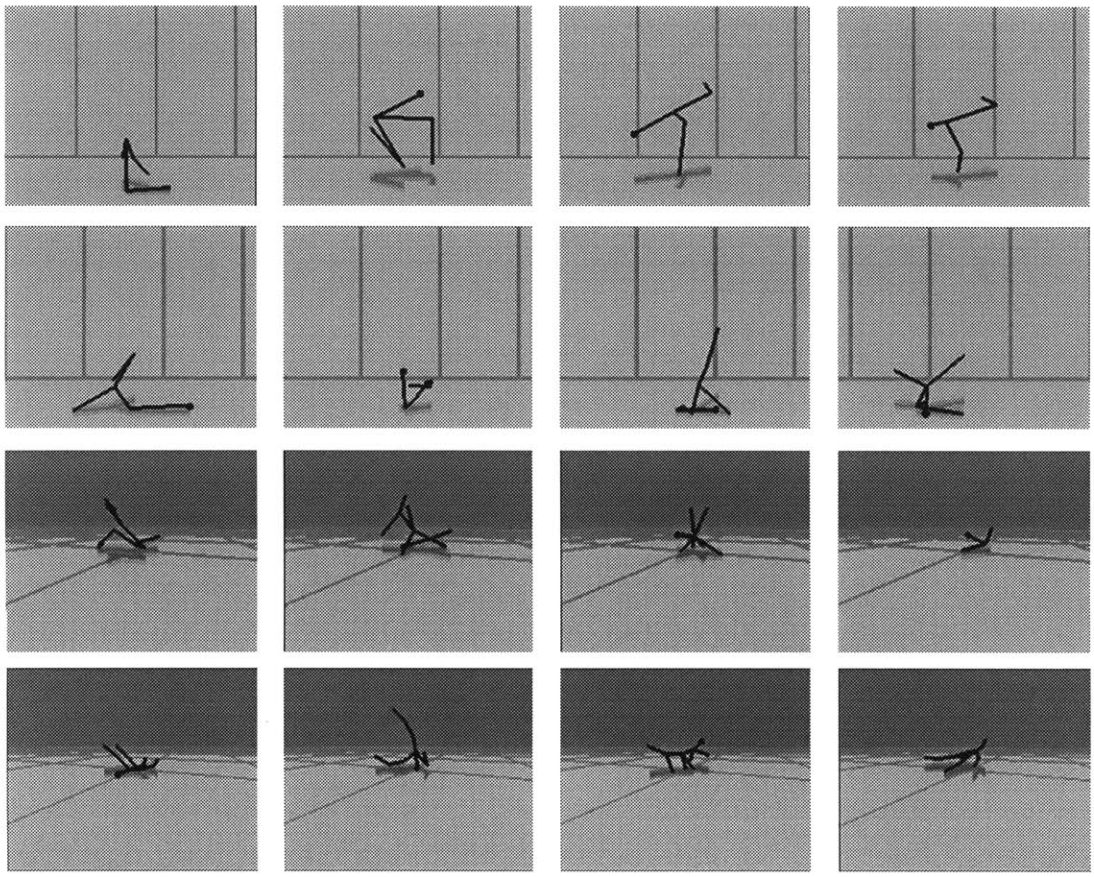


Figure 9 Four random variations from each of the four species of articulated figures are shown. The top row shows the 2D species with fixed topologies, the second row shows the 2D species with variable morphology, the third row shows the 3D species with variable morphology, and the last row shows the 3D species (called the "Vertebrates") with variable morphology, incorporating a segmented scheme.

Morphology

The figures are represented as rigid bodies made of interconnected limbs of varying lengths, joined at various angles. Each figure conforms to a tree-like topology—no closed loops can occur among the parts. The orientations of the limbs are hierarchical, as in most skeletal systems (for instance, when your elbow bends, everything from the elbow down rotates about the elbow joint). A body is deformable internally (autonomous changing of joint angles) but is rigid in the sense that it does not deform passively in response to environmental stimuli, such as collisions.

The human form emerged from the animal kingdom through natural selection according to Darwinism. Likewise, in this little world, human-like forms can emerge. In the spirit of artificial life studies, I have made most of the species of articulated figures have variable

morphologies. The number of limbs, the angles at which limbs are connected, and the lengths of the limbs can vary according to genetic variation. This opens up the arena for many possible forms as well as motions. Appendix A offers a detailed explanation of the scheme for morphology developed for the Vertebrates. Figure 10 shows four examples from the Vertebrates, and demonstrates the variety of morphology in an initial un-evolved population.

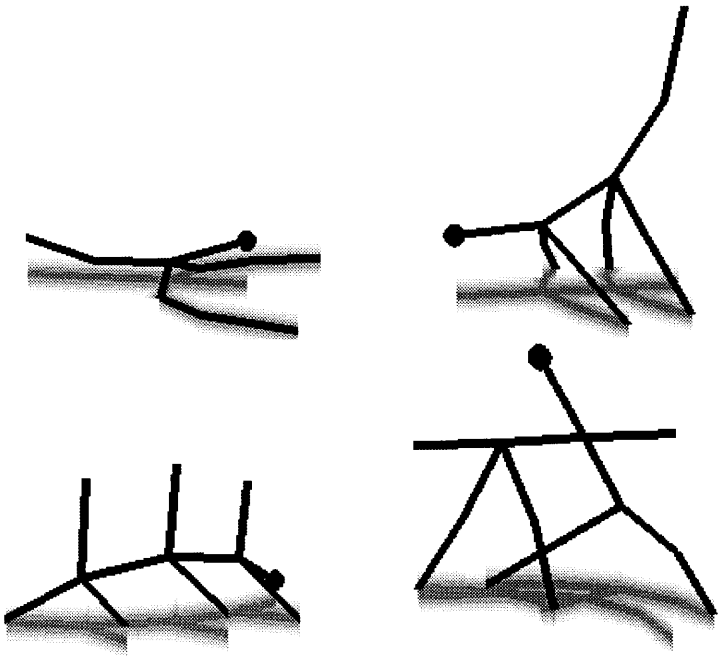


Figure 10 Four examples of the Vertebrates, the most complex of the articulated figure species, demonstrating morphological variation.

I have experimented with evolving the Vertebrates without any interactive intervention, with fitness pressures for locomotion to see what would spontaneously emerge. The most interesting finding is the fact that morphology and motion behavior usually evolve in tandem. A form can constrain a motion style, but also a highly fit motion style can encourage emergence of forms which support that motion style. These notions were the impetus for an artificial life experiment I had developed parallel to this thesis for exploring emergent morphology along with locomotion (Ventrella, 94). It was demonstrated that anatomies and locomotion behaviors reminiscent of some familiar species of animals tended to emerge together. The environment for this artificial life experiment has been folded into the Character Evolution Tool so that one can explore emergent morphologies and motions for their own sake.

Motor Control

The motor control of the collection of figures in the Character Evolution Tool uses the same scheme as the Walker. Motion is the result of multiple simultaneous sine functions. In a similar fashion to McKenna's (90) motor control for the virtual roach (yet much simpler), this system utilizes a rhythmic motor control program which consists of a combination of sinusoidal angular motions in multiple limbs. The differences from one figure's motor program to another's can vary greatly according to genetic variation. A figure's motor program is continually active at all times and cannot change during the figure's lifetime. This is a time-based motion control system, as opposed to a physical-based control system. Physical-based control systems are used in [Ngo and Marks, 93], [van de Panne and Fiume, 93], and others, in which the physical aspects of the environment determine the motions of the limbs, such as contact with the ground surface, global tilting angle of the figure, etc. My figures have no stimulus/response modeling, nor any proprioceptive senses, and so their motor programs cannot change in response to the environment.

In the simplest of the articulated figure species, consisting of five limbs (figure 11), joint angle changes are determined by collections of sine functions (one sine function for each of four joints).

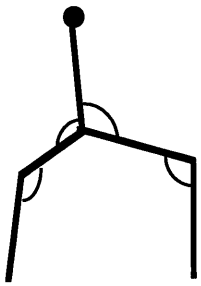


Figure 11 The simplest of the articulated figures. To achieve motion, the four joint angles (shown as arcs) are changed according to sine functions of varying amplitudes, frequencies, and phases.

In the more complex species which have variable morphologies, I have devised a different scheme, due to the fact that the number of limbs (and therefore joints) can vary. Also, the joint angles (being 3D) must be specified with more than one angular coordinate (such as *yaw*, *pitch*, and *roll*—I deal with yaw and pitch only). The scheme collapses the number of genes necessary to specify the joint angle parameters in potentially complex phenotypes,

such that the species can have a fixed genome length. This scheme is described in more detail in Appendix B.

Periodicity

In any one figure, the frequencies of sine functions controlling angles in the joints are set to be either equal or having ratios of 2.0 or 0.5. For instance, one joint angle sine function may have a frequency of F, and another might have a frequency of F/2 or F*2. Having whole-number frequency ratios encourages periodic motions in the figures—every gesture the figure makes is certain to be repeated over and over again as the animation runs. This is helpful for the acquisition of locomotion skills, in which periodicity can help (and it makes them potentially good dancers as well). But, as we shall see in the discussion on the gesture tool below, this can impose constraints on the matching of figure motions with the user's gesture.

Physics

In the models that researchers have developed for simulation of articulated stick figures using such techniques as forward dynamics (Badler et al., 91), many realistic phenomena can be produced such as falling under gravity, bouncing collisions, and momentum and inertia effects. Many of these models are very robust, and sacrifice computational speed for simulation accuracy. The amount of detail in an implementation of a physics model should reflect the purpose of the research, or of the *art*, in some cases. I have constructed a *qualitative physics* model which works sufficiently for the general purposes of this research (dynamic design for visual communication and entertainment). It is simple yet it produces many of the salient features of interacting rigid bodies in the real world, such as gravitational effects, inertia, angular and translational momentum, friction, and dampening. Appendix B offers a more detailed description of the physics model developed for these figures.

This abbreviated physics model also does not place a burden on computation and thus is very fast for the purposes of real-time animation. As many as twelve of the Vertebrates can be animated at the same time on the screen at animation rates of more than ten frames per second. The ability to animate in real time or near real time is crucial for the interactive evolution aspect of this system to work—the user must experience true motion in each of the figures in order to do any comparative evaluation of subtle motion styles.

Architecture

There are five major components of this system:

1) the genotype library
2) the phenotype library
3) the genetic algorithm
4) the automatic evolution engine
5) the interface—which ties all of the other parts together.

Figure 12 illustrates the overall scheme of the Character Evolution Tool, showing each of these components.

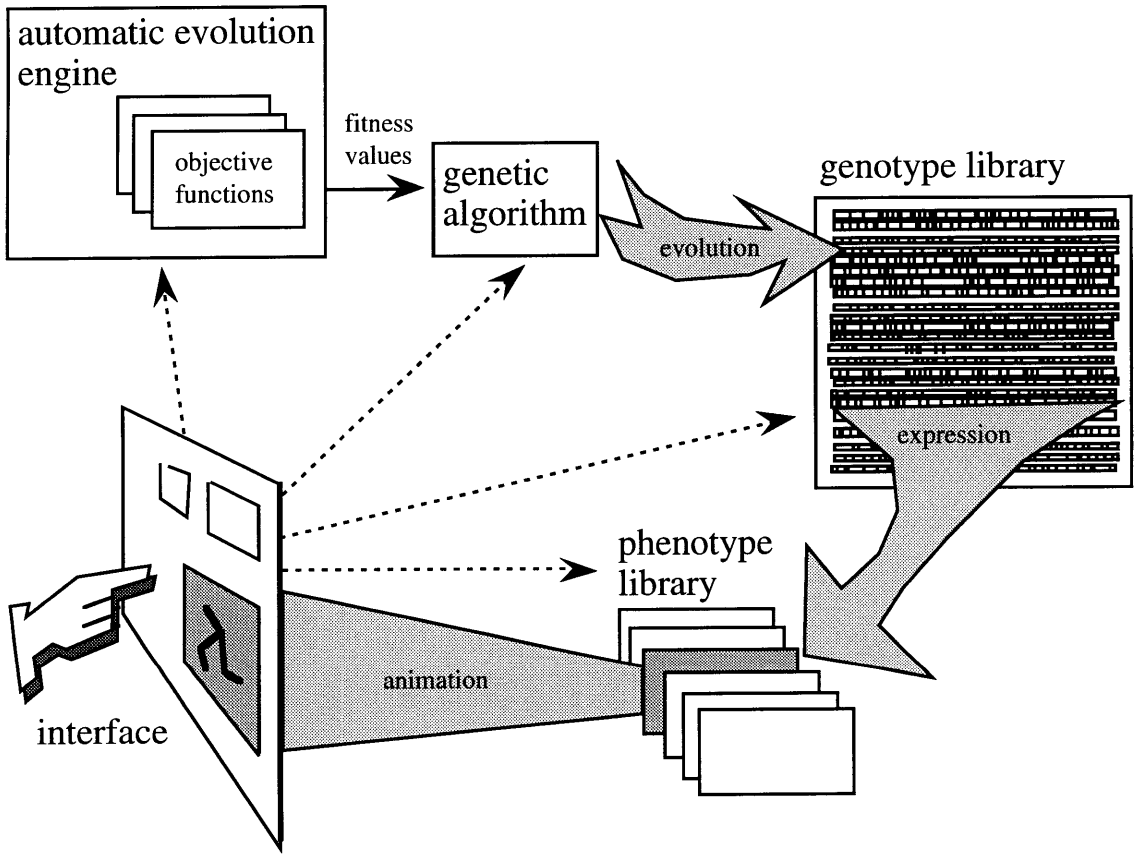


Figure 12 The overall scheme of the Character Evolution Tool. Through the graphical interface (lower left of the illustration), the user has the ability to control many levels of the system.

1) The Genotype Library

A large library of genotypes are held in memory during interaction with this system. This library contains the genotypes for each species, with each species consisting of a maximum

population of 100 behavior objects (some smaller subset of the total population is typically active during interaction). Each species' genome can have anywhere from 1 to 50 genes. Genes are stored as floating point numbers ranging from 0.0 to 1.0. These values are interpreted in a variety of ways by the phenotypes as parameters for visual attributes. For instance, a gene specifying the color of an object might be scaled to an integer value ranging from 0 to 255; a gene specifying an amplitude of joint motion might be scaled to a real number value ranging from 0.0 to 90.0, etc. The genotypes in the whole library are initialized as random at the start of the program run. As the user manipulates genes, interactively evolves populations, or initiates automatic evolution, the genotypes are changed accordingly. A population of genotypes for one species can be saved in a file at any time, and previously saved files can be loaded into the library at any time.

2) The Phenotype Library

Associated with each genotype is, of course, a phenotype. Twelve species of phenotypes (behavior objects) have been created and each are loaded into memory at the start of the program run (these were outlined at the beginning of section 4). At any one time, one of these species is displayed on the screen as a population (showing variation). While a species is active, the user can manipulate the genotypes associated with the behavior objects in that species either by tweaking genes directly, interactively evolving, or initiating automatic evolution. In each case, as genotypes are altered, the phenotypes reflect the changes immediately as visual transformations. At any time, the user can switch to another species and work with it for a while and then return to the previous species—the state of the genotypes will remain unchanged, upon switching species.

3) The Genetic Algorithm

The GA is in charge of the basic genetic operators: selection, mutation, and crossover. It is activated when the user selects the "new generation" button after assigning fitness values to favored individuals, or when the automatic evolution engine has completed an evaluation pass—or some mixture of both. To do its thing, the GA requires fitness values supplied by interactive evolution or the automatic evolution engine. This is the only input the GA uses in determining the probabilities for selection of each individual for reproduction.

Selected individuals reproduce via crossover: two selected individuals contribute alternating chunks of their chromosomes to each offspring in the new generation. These chunks can have varying lengths. This is determined by the variable *crossover rate*: As parental genes are being copied into an offspring chromosome, the source can switch from one parent to

the other a number of times. The probability of switching is determined by crossover rate. Crossover rate is computed per species, and is inversely proportional to the species' genotype size.

The setting of mutation rate (interactively controllable) determines the chances of a single gene in a genotype being randomly mutated immediately after reproduction. In mutation, a gene's value can increase or decrease by as much as 1.0 or -1.0, but with higher chances in the 0.0 range. The mutation operator insures against a resulting mutated gene value exceeding the bounds of (0.0, 1.0) by a wrap-around procedure. The mutation rate can affect the amount of variation in the offspring. It can be seen—in the context of creative design—as a determinant for the amount of automatic experimentation in a population, as it evolves.

4) The Automatic Evolution Engine

Some species of behavior objects come with the ability to evolve according to one or more pre-defined fitness functions. This component essentially allows the fitness functions to kick in and to affect evaluation of the phenotypes automatically. Examples of fitness functions are offered in the section which details the Interface. When the automatic evolution engine is turned on, a biological clock is initialized which has a set duration—at the end of this duration, all the phenotypes are evaluated, ranked by fitness, and mated proportional to ranking. The new generation of phenotypes is automatically created and appears on the screen at the end of the biological clock's period, and the cycle begins again. At any time the user can contribute to the evaluation of the phenotypes via interactive evolution. The user can also set the proportions of multiple fitness functions to guide the direction of evolution.

5) The interface

The interface brings together all the parts of the system and displays a species of behavior objects in multiple windows. Through the interface, the user can interact with the other parts of the system, using a mouse as input device. These interactions include:

- choosing among different species of behavior objects
- changing the size of the population
- initiating automatic evolution
- adjusting the weights of the pre-defined fitness functions
- adjusting the duration of the biological clock for automatic evolution

- adjusting mutation rate
- tweaking genes directly
- resetting the entire population's genotypes to random values
- interactive evolution
- saving and loading genotype files.

The components of interface are described in more detail in the following section.

The Interface

A large amount of this research has been dedicated to visualizing the processes of the GA and making the many components of the GA accessible to the user. To begin, a schematic of the interface is given in Figure 13, indicating each of the functions.

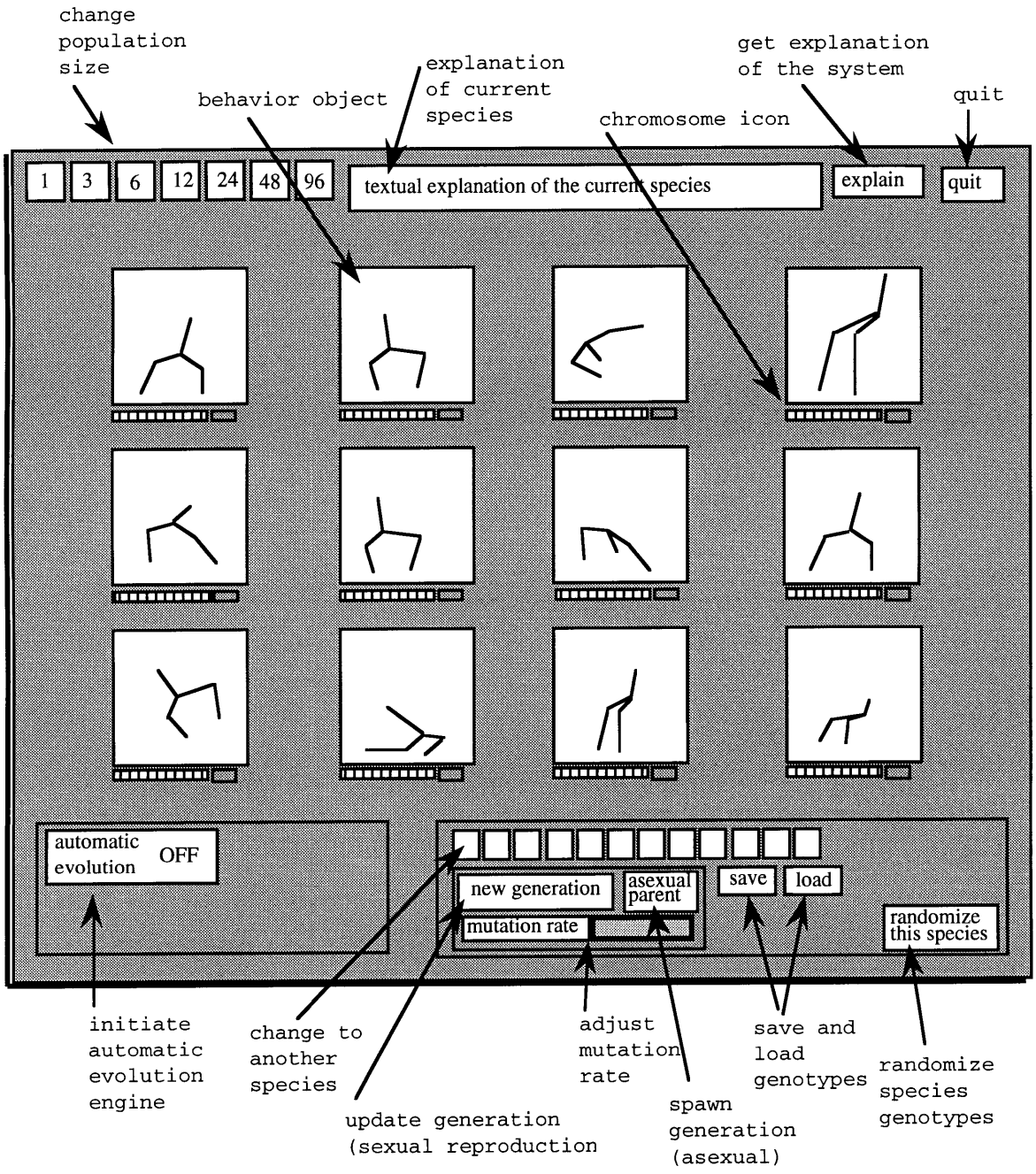


Figure 13 A schematic of the interface. In this depiction, twelve individuals from the simplest articulated figure species are displayed, showing variations among them. Each of the interactive functions are labeled.

Channel Surfing

One can use the interface in a passive mode—just watching a behavior objects do its thing, switching to another species of behavior objects, turning on automatic evolution and watching the results, and reading the explanations which accompany each species. The passive mode is suggested as a way to get to know the system.

Evolving/Gene tweaking

One can also actively change the behavior objects by affecting their genotypes. This can be done on three basic levels:

1) **The Watson and Crick Level** (Resnick, 94), where the user can open up a genotype and tweak individual genes and directly see their results in the phenotypes
2) **The Darwin Level**, where the user controls evolutionary processes pertaining to mutation rate, reproduction styles, and fitness criteria
3) **The Creator Level**, where the user is thinking primarily on the *phenotype* level, and designing through interactive evolution, gesturing, etc.

These three levels are described below:

1) The Watson and Crick Level

This is the lowest level of control. The user can click on a button at the right of each chromosome icon to "open the hood" and see the gene values represented as rows of slider knobs, which can be interactively adjusted. Changing the value of an individual gene results in real time visual results in the behavior object. For instance, if a gene specifies the length of a limb in an animated character, changing the gene value would result in the limb visually increasing or decreasing in length. Figure 14 shows an example of a behavior object (the fractal tree biomorph) whose genotype has been opened for tweaking.

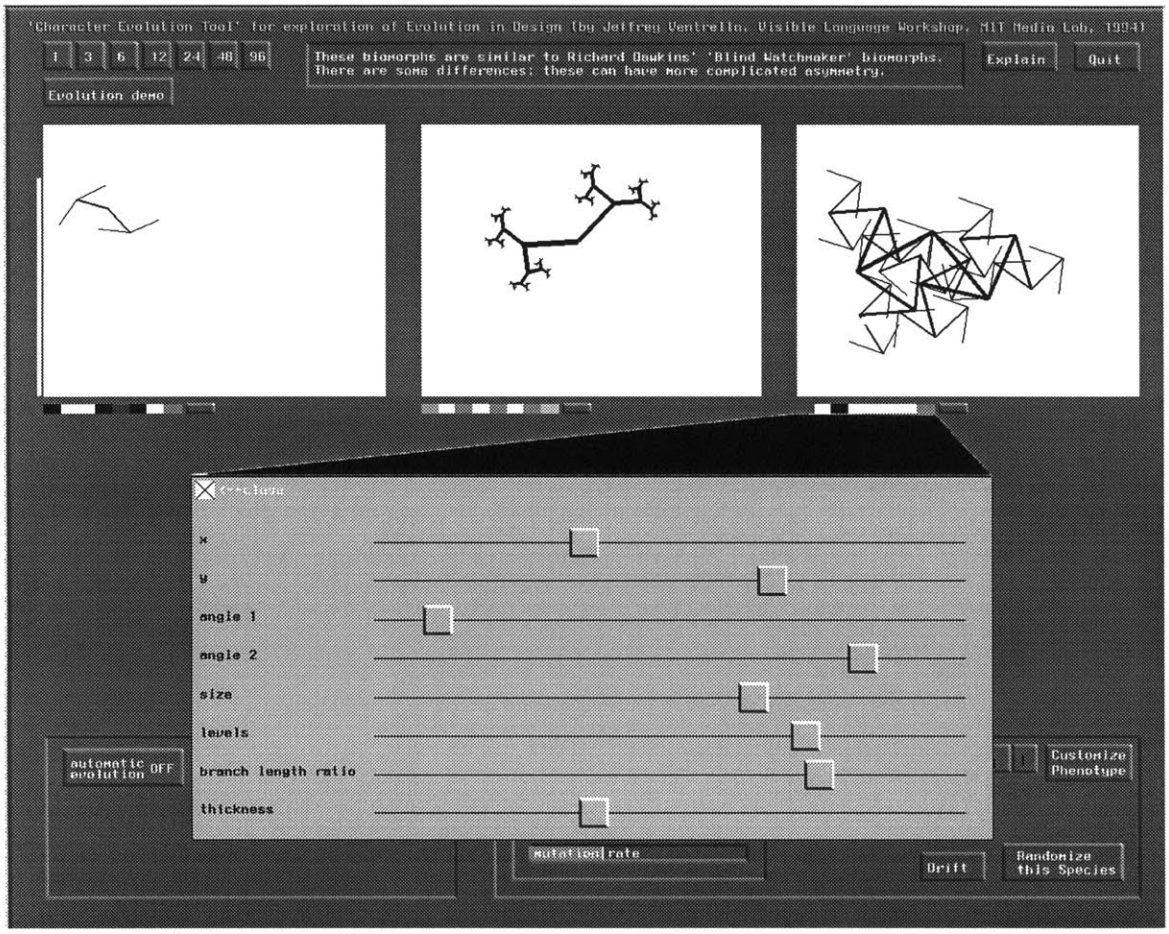


Figure 14 A behavior object's associated genotype can be opened and individual genes can be tweaked. The genes for the species shown (the fractal tree biomorph) are labeled at the left of the genotype panel. They are: X (position) Y (position), (branching) angle 1, (branching) angle 2, size, (fractal) levels, branch length ratio, and thickness (of lines).

At this level one can get to know the particular effects of each gene on the phenotype by interactively adjusting the values and seeing the results in real time. A technique similar to this is described by Oppenheimer (91), who suggests that this kind of isolation of input parameters to a phenotype and the instantaneous feedback from adjusting the parameters affords one the ability to *sense* the geometry, enhancing intuition about the visual form.

Behavior objects whose genomes are relatively short, such as the tree biomorph (having eight genes) are easy to explore in this mode. Other behavior objects, such as the Vertebrates, have on the order of fifty genes, many of which have indirect effects on the phenotype and cannot be noticed until other genes are changed as well. The Watson and Crick Level becomes problematic in this case—which actually serves a didactic purpose,

demonstrating how one cannot easily design form and behavior in high-dimensional search spaces by manipulating isolated parameters. One must move up to a higher level of abstraction, such as the Darwin level.

2) The Darwin Level

On this level, one can utilize the dynamics of Darwinian evolution by turning on the automatic evolution engine and setting up fitness functions. One can also set mutation rate, population size, and the duration of the biological clock's period.

Some of the species come with their own sets of fitness functions. These species can be automatically evolved according to fitness functions and the relative weights associated with multiple fitness functions, if there are more than one. At the end of each evaluation period, the values created by the fitness functions for each individual are added up to determine the final fitness values. A fitness function can contribute a positive (reward) or a negative (penalty) amount to the final fitness.

One example of a species which has a pre-defined fitness function is the fractal tree biomorph species. Its associated fitness function rewards each biomorph according to the degree in which the vertices of the biomorph are distributed in the picture space. This is done by taking the sum of the distances between every point with every other point in the biomorph (if each is within the picture space). This encourages the evolution of shapes in which the branches tend to be space-filling. Figure 15 illustrates four un-evolved biomorphs sampled from a population of twenty-four (shown at top) and one biomorph (shown at bottom) representing the population after this fitness function has affected evolution for twenty generations.

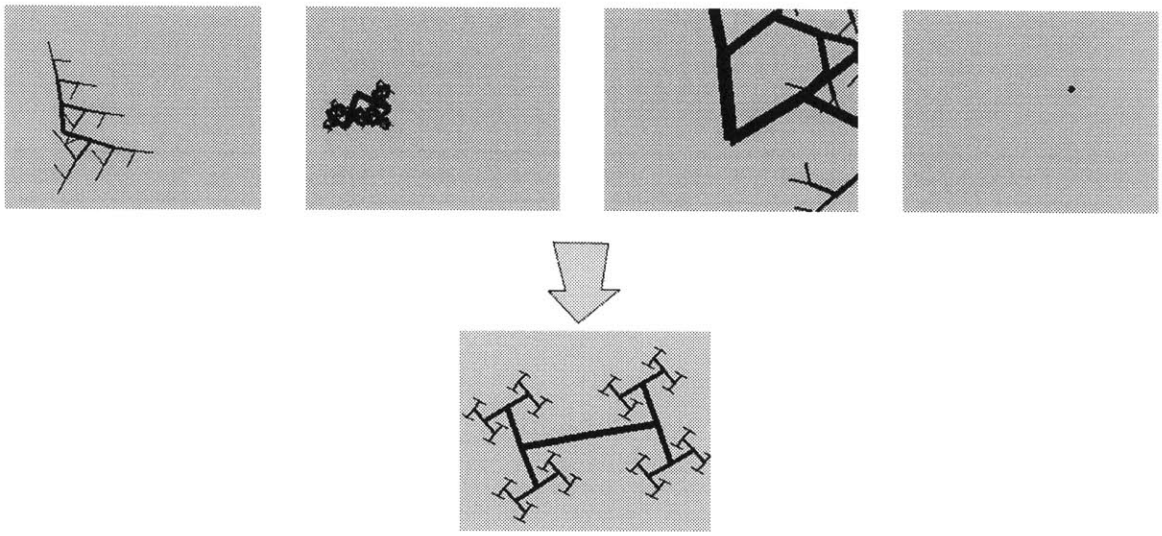


Figure 15 Automatic evolution of space-filling behavior in the fractal tree biomorph

The articulated figure species each come with a set of five fitness functions, the proportional weights of which can be adjusted by the user. Figure 16 shows a close-up of the automatic evolution engine with different settings for weights in each of these fitness functions. Each of these can be adjusted by the user at any time, as well as the biological clock (shown as "evaluate duration." (the other buttons in this panel will be explained below as I discuss their associated functions).

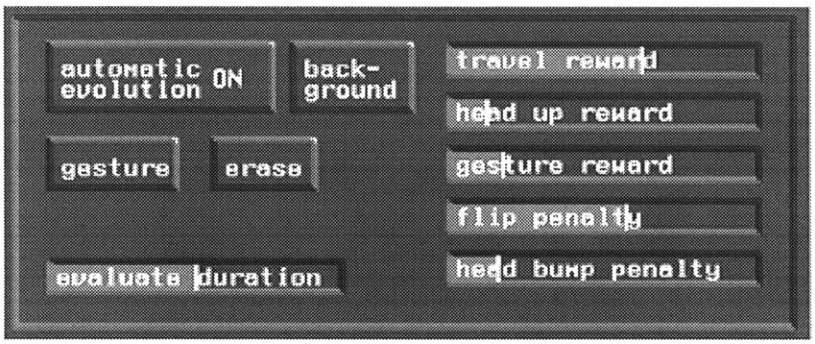


Figure 16 The automatic evolution engine control panel showing fitness functions for the articulated figure species

These fitness functions are:

Travel Reward
The distance between the starting point and the ending point after the evaluation period is used as a positive contribution to overall fitness—this promotes locomotion.

Head up Reward
To encourage more realistic mammal-like (and human-like) morphology and motion

Gesture Reward
When a gesture is drawn into any window, features of this gesture (seen as a *space-time* object) are compared to features in each of the characters' head motions (another *space-time* object) better matching of features determines a positive contribution to fitness.

Flip Penalty
The angular velocity of each figure's body is accumulated in a variable during evaluation which is used as a penalty for excessive rotational motion.

Head Bump Penalty
Each time a character's head encounters the ground surface during evaluation, an amount is subtracted from overall fitness

Setting the weights of these functions at differing proportions can affect the direction of evolution in many ways. As stated before, while the automatic evolution engine is running, the user can contribute to evolution at any time by interactively assigning fitness values to individuals, adjusting mutation rate (for more or less experimentation in the population), and changing population size.

Background Evolution
There is a feature that allows the graphical animations to be run in the background (eliminating graphics computations), which speeds up the process considerably. This mode is initialized by selecting the Background button (shown in the illustration above). It is useful before initializing background mode to increase the population size so that the GA

will have a larger selection of individuals to work with. While a population is evolving in the background, a time series graph displays the average fitness of the population over the generations so that progress can be monitored. While background mode is on, pressing any mouse button discontinues it and brings the display back to normal. Figure 17 illustrates this graph. It illustrates a frequent effect: average fitness during evolution exhibits sharp increases at irregular intervals. This is caused by chance mutations or crossovers in the population suddenly creating individuals of higher fitness. Artificial life experimenters have similarly observed, in their models, evolutionary stasis interrupted by periods of rapid change. This is compared to the pattern of *punctuated equilibria* observed in the fossil record (Eldredge and Gould, 72).

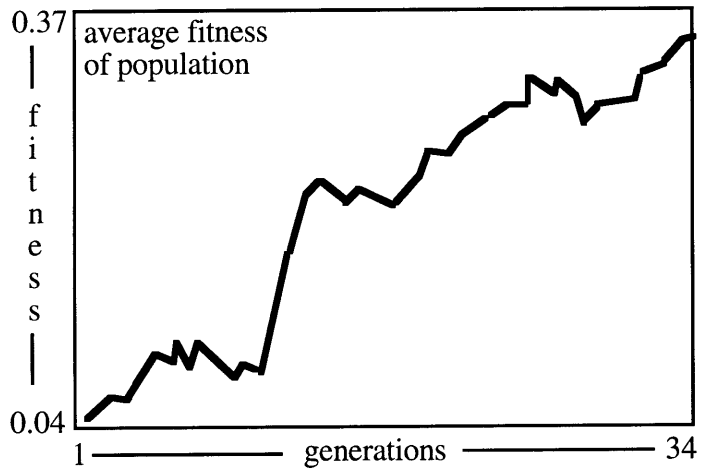


Figure 17 A time series graph is displayed when the user chooses to put the graphics in the background and run automatic evolution without visual display. This graph allows one to monitor the progress of a population.

3) The Creator Level

Very important in this design research is developing user-interface techniques which offer the user (as animator-designer) the ability to think on a high level—an expressive level, and not to be too distracted by the mechanics of Darwinian evolution. The two main features of this system which allow this are the interactive evolution feature and gesturing.

The interactive evolution feature is reminiscent of Richard Dawkins' Blind Watchmaker (86), Karl Sims' interactive evolution system (91), and many other systems recently developed. In a typical interactive evolution system, a collection of images are displayed for the user to peruse, and select favorable examples. A variety of techniques for selection and mating have been explored. The Character Evolution Tool employs an intuitive,

flexible reproduction scheme. In Figure 18, a close up of the panel with general controls is shown. On the lower left is the Reproduction area. Here, the user can choose asexual reproduction, create a new generation (for sexual reproduction), or alter the mutation rate.

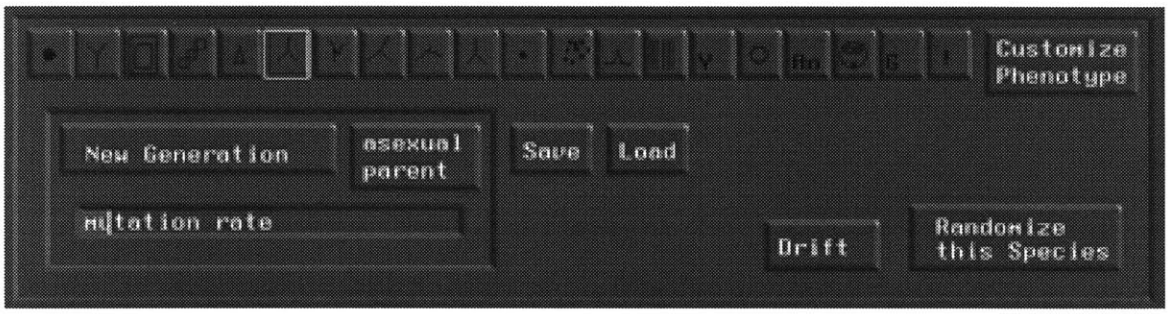


Figure 18 The "genetics" panel of the interface. At the lower left is the Reproduction area, where the user can choose asexual reproduction, create a new generation (for sexual reproduction), and alter the mutation rate.

In my system, two forms of reproduction are possible: asexual and sexual. In asexual reproduction, choosing the asexual tool ("asexual parent" button in the illustration) and then clicking on a specific behavior object's animation window signifies that the behavior object's genotype have offspring with variations, which constitute the next generation. Sexual reproduction is initiated by clicking the "new generation" button. This procedure produces a new generation in which the most fit contribute statistically more genetic material. Assigning fitness is done simply by selecting inside the image—holding the mouse button longer increases the fitness. A "fitness thermometer" at the left of the image visualizes fitness as it is being adjusted. Figure 19 illustrates this feature.

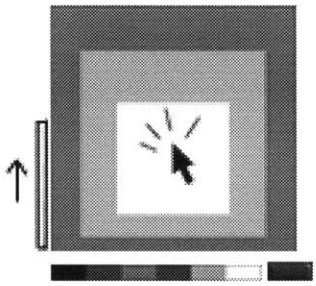


Figure 19 By clicking the mouse cursor into a behavior object's window, one can assign a positive fitness value to that behavior object. Holding the mouse cursor longer causes the fitness to increase. It is visualized at the left of the window in the fitness "thermometer." Fitness can not exceed 1.0.

The user can also re-adjust fitness at any time by grabbing the fitness thermometer and lowering the level. The sexual reproduction feature is very flexible. One can assign NO fitness values before selecting the new generation button, in which case all individuals have equal chance of mating for the next generation. Or one can assign a fitness value to only one individual, in which case reproduction is asexual. Or one can assign fitness values to two, three, or any number of individuals, in which case the proportions of fitnesses determine selection for mating. Figure 19 illustrates this mechanism.

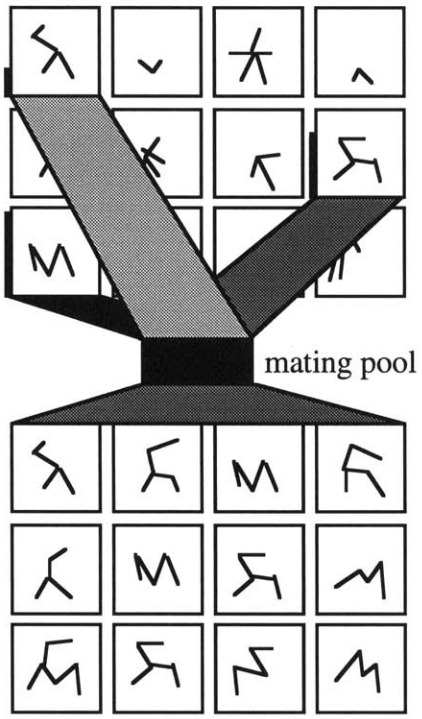


Figure 19 In sexual reproduction, any number of individuals from one generation can contribute genetic material to the next generation, through mating. Chances of being selected for mating are proportional to fitness. In this illustration, the more fit individuals are shown with darker paths connecting their associated windows with the mating pool, to indicate the degree in which they contribute genes.

Gesturing

The gesture tool offers a novel approach to the definition of a fitness function by bringing some element of motion "style" into the grasp of a genetic algorithm. The gesture tool is derived from the research of Karen Donoghue (92). In her work, gestures drawn on a computer display through a pressure/tilt-sensitive stylus are parsed for specific attributes, including direction, pressure, tilt, and rate of the drawing action—and these attributes, once parsed, directly determine the actions of animated characters. Unlike Donoghue's scheme, the characters in my system are encouraged to emulate features in the gesture, *statistically*,

through evolutionary pressure. This variation of the spacetime constraints paradigm of saying *what* but not *how* gives the population some freedom to converge on the gesture in a variety of possible ways. Essentially the trajectory traced out by each character's head motion is compared to the gesture to determine how well a character's head motion matches a feature of the gesture. Less of a match between gesture and character head motion results in lower fitness values. Note that this technique is not bound to head tracking only—it can as easily be applied to any other body part (consider evolving an Elvis Presley Hip).

The gesture tool was implemented to be used with the 2D articulated figures. Problems of matching the 2D gesture with 3D motions were not tackled. The question of exactly which features of a gesture to compare to the motions of a character is not a trivial question. Although I have not investigated the capabilities of this technique thoroughly, experiments for this thesis have been successful and have set some groundwork for further research in this regard. Two versions of the gesture tool have been implemented: 1) absolute distance differential, and 2) direction/speed differential. They are described below.

1) Absolute distance differential

The first algorithm developed for matching the gesture with the motion of a character's head was based on absolute coordinate comparisons. The absolute positioning of the head during the gesture-matching period is compared to a moving point on the gesture as the character moves about. Figure 20 illustrates this technique.

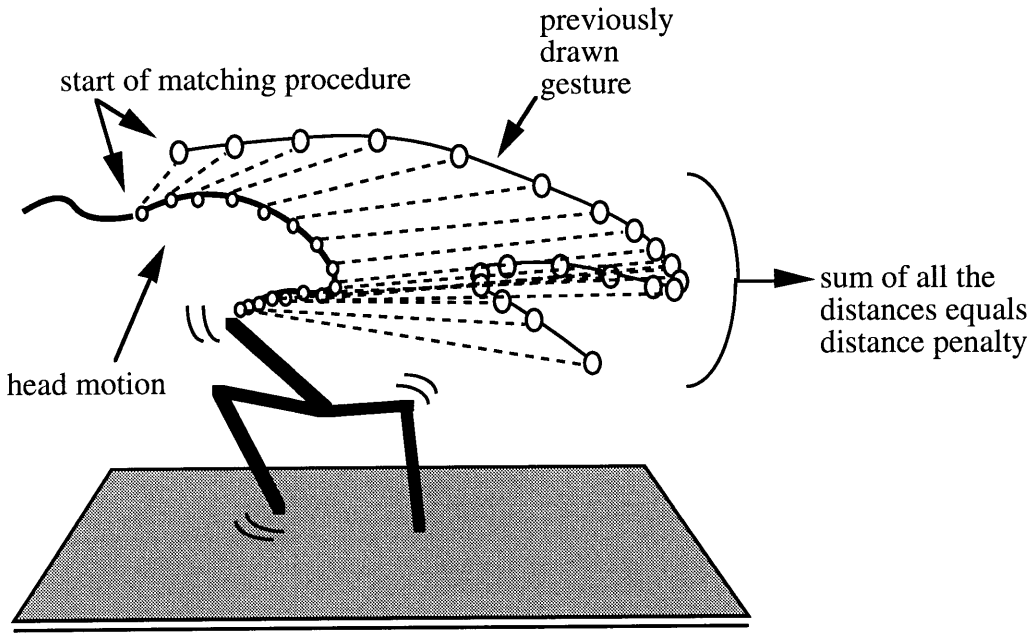


Figure 20 The absolute distance differential technique used for matching the gesture with the head motion of an articulated figure. For a specified duration, the position of the head is compared to the position of a moving point along the gesture. During that time, the distances between the two paths are accumulated to determine an overall penalty for distance.

2) Direction/speed differential

This technique is an improvement over the first technique. Instead of comparing absolute Cartesian distances between points in the gesture and points in the head trajectory, it compares their directions and speeds at corresponding locations, as indicated in figure 21. These components represent higher level features, in that they each require two consecutive points along the path to be calculated. This technique has shown to be more flexible because it imposes less constraint on the fitness evaluation by permitting the motion to be compared *at a distance*—instead of using absolute proximity as a criterion. It measures similarity between motion and gesture, regardless of the gesture's actual location in space.

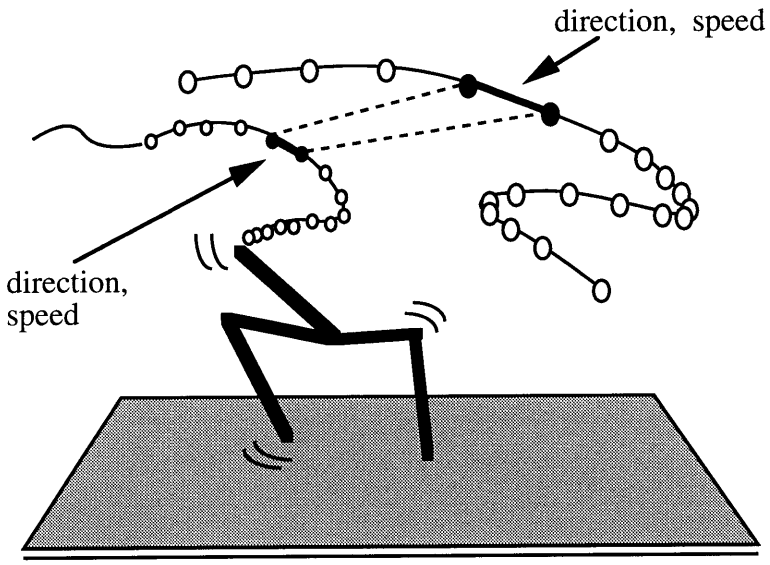


Figure 21 The direction/speed differential technique used for matching the gesture with the head motion of an articulated figure. For a specified duration, the direction and the speed of the head motion is compared to the implied speed and direction determined by two adjacent points along the gesture. Differences in direction and in speed are accumulated to determine penalty.

While this way of measuring a character's similarity to the gesture is more flexible, it was still found in preliminary tests that characters whose motions could approximate the gesture in any way were few and far between—and motions in the population could not easily converge on the gesture. I identified two reasons for this:

1) Even if a character's head motion traced a trajectory that was similar to the gesture, it could not receive any credit if its trajectory was traced at a different *rate* than that of the gesture

2) The character's trajectory may have matched the gesture but its matching may have been *offset* by a certain time amount.

These are comparison problems which have to do with the nature in which the gesture is *read* by the fitness function as it matches features to the character's motion.

Read my Motion...Now!

For these reasons, I chose to add to the population's genome two unique genes, which I call "motion scanning genes." These genes affect the way in which the fitness function compares the character's motion to the gesture, by causing it to read the gesture in a variety

of ways, based on genetic variation. One gene controls the point in time that the fitness function begins to compare the gesture to the motion. The other gene controls the rate in which the gesture is scanned, as it compares it to the motion. The motion scanning genes can be described as messages from the characters to the fitness function, sounding something like: "hey fitness function, read my head motion *now*", or "read my head motion *this* fast", where *now*, and *this fast*, can vary among the population. The motion scanning genes give the fitness function a better chance at identifying similarities between character motions and the gesture, because all individuals' motions are not read exactly *at the same time* or *at the same rate* by the fitness function as it matches them to the gesture.

Saving and Loading Genetic Data

The state of a population can be saved into a file of genotypes at any time. A saved file can also be read in at any time for further work. By saving and loading files, one can work on a population over more than one session, and develop a library of families to work on over time.

5. RESULTS

The Character Evolution Tool allows one to generate forms and behaviors in a number of different graphical domains, and to save them for later use. It offers a variety of levels at which to explore and control these forms and behaviors. By allowing behavior objects to be explored at varying levels, it educates at the same time as it functions as a designing tool—this is a necessary part of the way in which it works. To test its viability as a *tool to think with*, a number of people were asked to interact with the system, so that I could get reactions and comments.

Subject's Responses

In some cases subjects were not told anything about what the system was about nor what they were supposed to do with it. The reason for this was to see if the interface communicated enough to give the subject a sense of what was going on, and to measure to what degree the information unfolded as the user explored. In most cases, as soon as the subject was told (or discovered) that there was a genetic evolution basis to the system, he/she began to understand things much more, due to the availability of a familiar metaphor.

Reactions to the animated figures are almost always positive—people find them amusing right from the start, even before they have begun to manipulate them. This has served the system well: it establishes a backdrop of motivation for further exploration into the nature and control of their behaviors.

Although there is an obvious element of chance in the creation of behaviors, there is also a significant degree of predictability over the course of the interactions, and people tend to have some sense of ownership to the behaviors that have evolved.

Lag Time Learning

There was a realization as a result of implementing the gesture tool. It has to do with the rate of genetic learning in the population, and how incompatible this is with the notion of telling someone to "move like this." Clearly, a choreographer, when demonstrating a movement to a dancer, expects immediate results (whether or not the results are good the first time). This kind of immediacy is expected when one is communicating non-verbally to another person, or even to an object in a graphical interface. We are conditioned, for instance, to expect an object in a typical graphical user interface to *drag* across the screen

when we place the mouse cursor into it, click the mouse button, and move the mouse. Some actions, by their nature, demand immediate results. The idea of gesturing to a population—and expecting the population to genetically evolve over many generations in order to fit that gesture—is uncommon. Not only that but it is counter-intuitive. For this reason, the development of an interface for making this activity meaningful has been a challenge. But, absurd biological analogies aside, it may be useful when seen as a spacetime constraint solution, in that it presents a new way of thinking about the specification of motions in objects which have their own innate ways of achieving motion.

Aperiodic Expressions

One drawback to the gesture tool is that the set of possible head motions that an articulated figure can create is a very small subset of the number of possible gestures one can motion into the scene. The articulated characters are not sophisticated enough to generate *aperiodic* motions (which is a quality of many forms of linear expression). Their joint angle changes are created by series of sine functions each of whose frequencies are identical (or at least related by whole number ratios). Thus, any gestural action a character makes will be repeated over and over again. This is of course useful for locomotion, which is primarily a periodic activity in most animals. But it becomes a disadvantage in that it is difficult for a character to approximate most gestures a user may draw. For this reason, I have constrained the gestures I have drawn to have some repetitive qualities to enable some matching to be found in the characters' motions.

This problem would be alleviated if my characters had the ability to change their joint angles according to more complex motor control algorithms, allowing for aperiodic motions. Stimulus/response models such as those cited in section 2 would offer this kind of flexibility.

Prototyping and Phenotyping

The Character Evolution Tool is successful when seen as a research environment for a genotype/phenotype methodology of design. This is apparent in the accumulation of behavior objects which I have been able to quickly prototype for my own research and for demonstrating the concepts. A phenotype template was developed early on in the research to give me (and potentially other graphics programmers) a quick way to prototype a new behavior object. Figure 22 illustrates this template. The collection of different behavior objects I have created has not only increased my design repertoire, it has also offered many people a chance to see graphic behaviors demonstrated through a *genetic lens*.

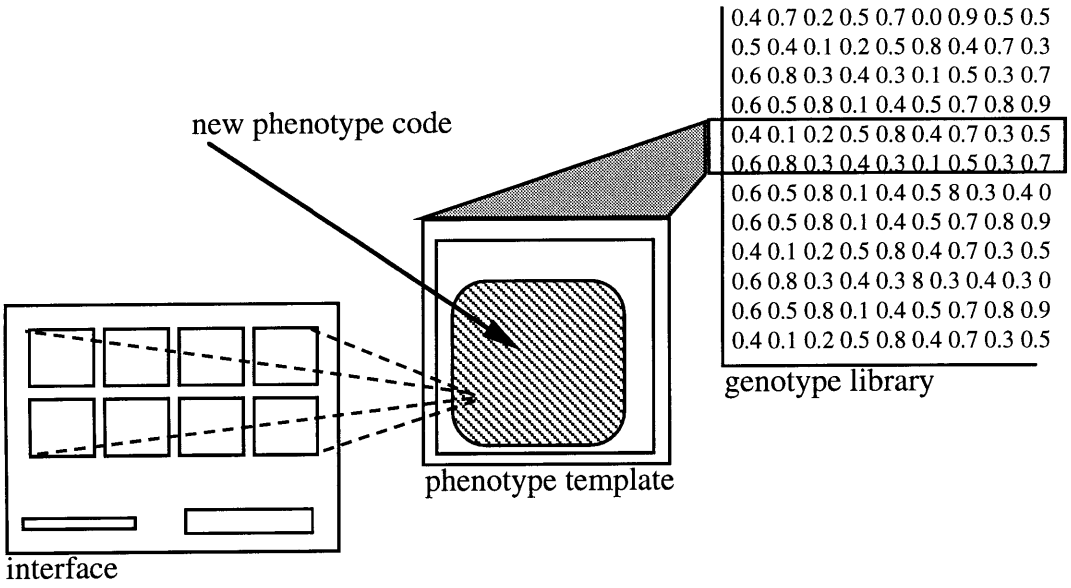


Figure 22 A new species of behavior objects can be created using the phenotype template.

Artificial Life

As a research tool for artificial life experiments, the Character Evolution Tool has proven useful. It has supplied an environment for a project in studying two interdependent emergent phenomena: morphology and locomotion behavior (Ventrella, 94).

6. CONCLUSIONS

The Character Evolution Tool is partly a research environment and partly a prototype for future design applications. At least three potential applications for the Character Evolution Tool are suggested. They are outlined below:

An Animation Tool

The Character Evolution Tool is primarily aimed as being an eventual component in the ensemble of tools for the designer of dynamic media. The future of the art of animation may look more like the direction of live actors in the production of a movie than the drawing of thousands of individual picture frames. The Character Evolution Tool contributes towards this eventual ideal.

Imagining this tool in the context of animation production, here is a make-believe dialogue (interpreted *verbally* here) which a future animator may have, while attempting to develop a character.

> "Bring up the 'Human Species.' Now, load the genotype file, 'Paul'. He still is not *funny* enough for the animated commercial I am scripting."
>
> "Paul is already in a sitting position, but I need to make him more upright—a tweak here, a tweak there...*that's* better."
>
> "OK, now, I want Paul to have the ability to cock his head, or maybe perk his shoulders, in a way to indicate *confused intrigue*, perhaps.
> I'll breed this population (multiple versions of Paul) for a while, and try to tease this behavior out ..."
>
> "Well, Paul, I'm not getting exactly the kinds of motions I need, but, you know, this funny motion would be great as an action to use at the end of the scene when you take a bite of the cereal. Let's save the genotype file for this action to be used later, and call it 'Gracious_Yum_Yum!'."
>
> "Now, a second action is needed after the yum yum shot, but I have a specific motion you need to follow, so here, learn to follow this gesture" (draws a cute gesture across the screen indicating the direction and style of Paul's head motion).....etc.. etc...

A Research Environment for Exploring Evolution in Design

A tool which allows a designer to easily represent a design problem as a genotype/phenotype pair could be very useful, if not only to augment the designing, then to at least give illumination to the nature of the design problem and the important variables involved. One facet of the Character Evolution Tool I have developed is that it may be seen as a testbed for a genotype/phenotype methodology of designing. During development, as I continued to explore new graphical domains (adding new species of behavior objects to the system) I began to modularize the process of prototyping new behavior object species—and to give myself the ability to prototype a new behavior object species in a short amount of time. To be able to do this, I found, can be a powerful way to explore theme-and-variation within a particular graphical domain. With more work on the interface, the system could be developed further toward this goal, making it easier to prototype new genotype/phenotype representations. Currently, this is done by making alterations to the C language code. Simplifying this process, or even eliminating the need to write code, would be desirable research development to further this tool.

PPP: Professional Phenotype Programmer

But since eliminating programming altogether is a large task, one might more easily imagine enhancing the modular aspect of this system in that phenotype "plug-ins" could be generated easily in C code or C++. The Character Evolution Tool already comes with a modular skeletal framework for creating new phenotypes. A two-tiered design strategy would be useful in this regard: imagine a design firm or an animation production house in which an official meta-designer, or *professional phenotype programmer* is employed. The chief artist finds a need for a new class of design objects. He or she then asks the phenotype programmer to design a representation for that class, as a phenotype (genotypes would already be available in a huge library). The main part of the system would already be in place, and the phenotype programmer would write the code which takes a set of genes and turns them into parameters for the new class of design objects. Having created this new phenotype, the chief artist and design team could then plug the phenotype into the main system and perform the many functions which are presented in this thesis. The Phenotype Programmer may also be in charge of designing fitness functions to accompany the phenotype, in accordance with how the design team intends the phenotypes to be critiqued in automatic evolution runs.

A Tool for Exploration of Evolutionary Principles

Another domain in which my work may indirectly offer contribution is the exploration of Artificial Life concepts, Darwinian evolution, and principles of emergence (Resnick, 92). I have observed that people using this system tend to become engaged in the populations of animated characters (more than the other species), to make comments on the way in which they are evolving, and to try their hand at breeding characteristics. This observation has lead me to see this as a *tool to think with.* The interface to the Character Evolution Tool has an instructional side in that it must bring some of the deeper ideas in GA mechanisms and evolution in general to the understanding of the user in order to be fully exploited as a design tool. For the purposes of exploring emergence and artificial life methods, the behavior objects can be compared to individuals within a single species, each species having its own unique genome and expression into a visual form. This facet of the Character Evolution Tool allows an extended way to explore artificial evolution, via many examples of species.

Design and Evolution

In reference to the uses of artificial life techniques in computer graphics, Steven Levy asks, "How can a bottom-up approach satisfy two seemingly exclusive goals: creating organisms which display novel behaviors, and creating organisms which do what we want them to do?" (Levy, 92). This gets to the heart of the problem in using evolutionary techniques in the design of graphical objects.

A constant philosophical underpinning to this project is a question sounding like the debate on Creationism vs. Evolutionary theory: can a system modeled after Darwinian evolution contribute to the design process? Dawkins (86) reminds us that Darwinism signifies a universe "without design." In this spirit, artificial life research is often about setting up minimal initial states and analyzing self-organizing dynamical systems as they emerge—a form of theoretical biology, a kind of *what-if* modeling where organization emerges bottom-up style (Langton, 91). It is very distinct from deliberate top-down design methodologies. Yet as we have seen, GA's have been explored as aids to solving complex design problems (Goldberg, 83). Perhaps new paradigms of Design are emerging themselves, which include adaptive systems separate from human problem solving, acting as a collaborator to the human creator. The Character Evolution Tool provides an environment with which to study this notion by bringing these two evolutionary processes together (the designer's designing, and the GA's searching).

User Interfaces to the Genetic Algorithm

A designer may ask, "How can I remain *in charge* of what's going on? If a GA is controlling evolution, I want to have the freedom to tell it *when*; *how much*; for *how long*; and for *what purpose*. And most importantly, I want it to communicate to me what it is doing at all times". How can this system give a designer as much control as possible in using the GA, while on the other hand, letting it do what it is good at? Should the GA be a black box? According to Darwinism, earthly DNA has been doing its phenomenal work throughout the history of earth life without being explicitly "known" by any organisms for the purposes of generally getting around and reproducing. The exception is the recent disruption of biological evolution by humankind, in which the black box is recently being opened, and genetic information is being used directly by the species for purposes of its advancement. At times the activities of the GA should be transparent to the designer—it must communicate. At other times, it should do the dirty work as the designer thinks on a high level. In summary, one aspect of this thesis may be categorized as design research in GA interfaces. The approach taken in this thesis is to offer the user varying levels of access to the nitty gritty mechanics of the GA dynamics, depending on what the user is interested in knowing at a particular time.

Conclusion

The animation system I have described in this thesis represents an intersection of many inquiries. Among them are: the nature of expressivity in visual objects, autonomy vs. control in animated characters, and the use of evolution as a metaphor and as an actual computational technique in graphic design. On the one hand this thesis research has produced a highly interactive artifact which serves as a prototype for future animation and design systems, but it has also presented an environment for which these inquiries can be explored. And at the same time, it can be fun to use.

Research in simulating natural processes continues to enhance the arts of imaging and animation at a rapid pace. The merging of artificial life with computer graphics has begun to produce images and animations of artificial creatures which are very realistic and which elicit sympathy and emotion in viewers, perhaps owing to a certain amount of autonomy in their behaviors. As techniques for autonomous character animation improve, we will see more examples of real time characters who can respond in increasing ways to the animators who make them and to viewers alike. This being a catalyst for my thesis research, the Character Evolution Tool presents some new ways to interact with autonomous characters, for the development of expressive body language.

7. FUTURE WORK

Clip Behaviors

Due to the fact that the genotype data structures for all the species are identical, a file saved from one species' genotypes can actually be loaded into another species' genotypes in memory. This is an intriguing feature which was not originally intended, but discovered once when I accidentally loaded a file into the wrong species. The results of this are almost always uninteresting, because there is very little mapping from one species genome to another. But a logical mapping is conceivable. This leads one to imagine the potential for a system which allows this to be done in an organized fashion. It would allow a form of inter-species breeding—impossible in the biosphere but very common in the ideosphere.

An idea suggested by Resnick (94) comes to mind, which he calls "clip behaviors." Clip behaviors are analogous to clip art (commercially available images, icons, and symbols used in many desktop publishing systems for inclusion in designs). If the genotype/phenotype representations for behavior objects in the Character Evolution Tool were designed with mappings between them, such that species could be treated as classes and sub-classes, it would be feasible to load the genes for behaviors evolved from one species into another species' phenotypes, with predictable results. A feature of this sort would be a very powerful catalyst for the transferal of ideas from one design domain to another.

Family Trees

One can save and load genotype files for populations of evolved behavior objects. But as yet a scheme has not been developed in this system for relating the files according to familial context, for handling *genealogy*. There is also no scheme for visualizing genealogical development. Further development in this regard would be a great contribution to the system, and it would empower the user with a rich environment with which to develop and manage developing, customized taxonomies of behavior objects.

Flesh and Bones

This thesis deals primarily with the creation of motion behavior. Line drawings serve well for this and so, in designing my articulated figures, I have not put flesh on the skeletons. But a complete animation studio would need a means for doing this as a next step in production. Once the motion behaviors had been developed for a character, an animator would have to then dress the character in flesh, clothes, what have you. The Character

Evolution Tool does not address this need. But it can be seen as one component in an ensemble of animator's tools in which another component would be available for "dressing the motion".

Interacting Characters

A useful extension to this tool would be an environment in which *multiple characters* of different species could interact with each other. Most animated narratives do not feature one lone character in a simple environment, but have two or more characters which interact with each other. As yet there is no mechanism for evolving behaviors for interacting with other characters (of the same species or of different species). Adding this level to the Character Evolution Tool would bring an element of *ecosystem* into play—what would become evolvable would not only be individual behaviors, but interactions among individual behaviors.

Imagine a future extension of the interface to the Character Evolution Tool in which each animation window included more than one species of characters. One can see that evolution would take on a new level of complexity—it would become co-evolution. One reason it would be more complex is that the concept of *fitness* may no longer apply to one character alone, but could apply to *another* character in the same scene, or even to the mutual interactions between two or more characters. To handle the multiplicity of fitness targets, the user would need to have the freedom to select any one of the characters in the scene, or to select the whole interactive *ecosystem*, to reward fitness. In working on this level, it is conceivable that one can be evolving, not just characters, but interactions between characters.

A co-evolution system would probably require an entirely different interface than the one represented in the Character Evolution Tool. No more being trapped in their own little test tubes—each character would have to be free to move about and interact with other characters within a large environment, in order to be evaluated. This may pose problems, since interactive evolution typically involves evaluation of a collection of individuals in a population (presented in a grid) to guide evolution. In conclusion, evolving an ecosystem would require a new interface design strategy, perhaps displaying something more akin to what a scene in the final animation would look like—the addition being the inclusion of the viewer (as user) interacting with an evolving cast of characters.

Glossary

Since this thesis brings together some terms from biology, graphic design, and computer science, there may be some difficulty in relating them. I have tried to keep confusion to a minimum. To help further, the following glossary is offered which gives definitions for some of the key terms used in this thesis.

articulated figure - in the context of computer animated characters, a geometrical object having parts that can move in relation to each other, such as sticks connected at angular joints.

behavior object - any computational object (usually expressed graphically) having states which can change over time, according to rules or environmental conditions

character - in this thesis, the class of all graphical objects which can exhibit expressive behavior. This includes cartoon characters as well as user interface widgets which have communicative motion behavior.

character animation - the art of conceiving and animating a persona (human, animal, or otherwise), traditionally in the medium of film, but more recently with computational tools.

chromosome - another name for the *genotype*.

evolution - the changes within a population over time, due to mating and mutation. Evolution does not always imply *better*, but always implies *changing*. In this thesis, *better* is generally the goal.

expressivity - the ability of a graphical object to communicate information or evoke feelings and aesthetic qualities through motion behavior

gene - one of the elements (bit, integer, real number, if-then statement, etc.) in the genotype. In this thesis, a gene consists of a real number within the range of (0,1).

genetic algorithm - a searching and optimizing technique based on the mechanics of Darwinian Evolution.

genome - the template for all the genotypes of one species. All the genotypes of one species have the same number of genes, and there is a one-to-one correspondence between genes, in terms of how each gene effects the phenotype.

genotype - in a genetic algorithm, this is the representation which consists of encoded parameters for the phenotype.

spacetime constraints - a computer-graphical technique in which an animator tells an autonomous agent (usually an articulated figure) *what* to do, but not *how*. For instance, *where* to go and *when* to get there may be specified, and the details of motion for achieving this are computed automatically.

species - in the context of this thesis, any coherent family of visual objects or visual relationships. Each species has its own particular visual characteristics, generative rules, and evaluation criteria.

phenotype - in genetic algorithms, the phenotype is the representation (as opposed to the *genotype*) which exhibits features that can be evaluated. The phenotype is the visible, behavioral expression of the genotype.

Appendix A
The Morphology Scheme for the Articulated Figures

This section explains the morphological schemes for the most complex of the species of articulated figures in the Character Evolution Tool, the Vertebrates. The expression of morphology from the genotype representation was designed to offer a means to generate great variety in the phenotypes. Variations in topological connectivity of limbs and angles in the joints give rise to structures resembling spiders, birds, four-legged mammals, human-like shapes, and a great variety of monsters. In the predecessor to the Vertebrates I had designed a biomorph with an open-ended structure based on a rather idiosyncratic scheme for expression of morphology from the genotype. This was done as an experiment to see if the pressure to achieve locomotion would cause symmetrical structures to emerge from a search space of mostly asymmetrical forms. Many trial runs suggested that the search space was too large for the GA to easily converge on symmetry, and that the representation may not have been designed sufficiently for *evolvability* —it did not incorporate a "constrained embryology", in which there are few genes yet each gene controls a more powerful feature of the phenotype [Dawkins, 89]. In discussing his own design for a biomorph, Dawkins refers to the emergence of segmentation in animal morphology in the natural world as a watershed event—as a likely increase in evolvability for those animals which chanced upon it. No such structural scheme exists in my previous phenotype design.

After exploring variations on this open-ended biomorph design, I settled on a generalized bilateral-symmetric structure—the Vertebrates—which still allowed for variation, and even occasional asymmetric features, but was characterized by a more realistic embryological scheme—with segmentation. In this biomorph design, a few basic elements are always present: a central body functioning as a backbone, having anywhere from one to four segments, and opposing pairs of limbs (as many as three), each limb having anywhere from one to four segments. Thus, the simplest topology is one body segment with a pair of opposing one-segment limbs. All segment lengths are set to be equal. Figure 1. shows four typical un-evolved anatomies of figures in their default (resting) stances. In some cases, asymmetry can arise. For instance, if the number of body segments is smaller than the number of limb pairs, extra limb pairs grow out from other limbs. This embryological quirk has been kept in the biomorph design, keeping with the idea that features like this might by chance produce useful strategies for locomotion.

A list of effects of the genes for morphology is given below:

- number of body segments
- pitch angle of joints between body segments
- number of opposing limb pairs
- pitch angle at the branch point of each limb pair
- yaw (or veer) angle at the branch point of each limb pair
- changes in pitch angle at the branching angle among consecutive limb pairs
- changes in veer angle at the branching angle among consecutive limb pairs
- number of segments in a limb
- pitch angle of joints between limb segments
- veer angle of joints between limb segments

Appendix B
The Motor Control Scheme for the 3D Articulated Figures

Attributes of motions within the whole collection of joints are created by one number series generating function, as a way to coordinate them. This function takes five control parameters (determined by five genes) as input, and generates a periodic series as output (with each number of the series corresponding to one joint angle in the figure). The pseudo code example below illustrates the function.

```
INPUT:      parameters from genes: (size, start, step  low, high)
            number_of_joints

integers:          size, start, step, number_of_joints
real numbers:      low, high

BEGIN
number = start

LOOP from index=1 to index=number_of_joints:
 (
  number = number + step
  output_parameter[index] = low + (number MOD size)/size * (high -
low)
 )
END

OUTPUT:     a periodic series of real-number parameter values
```

The purpose of this scheme is essentially to allow a small number of parameters to determine an indefinitely long series of values to control actions in multiple limbs, where the number of limbs is arbitrarily large. The series is periodic to encourage regularity, yet with a large number of possible polyrhythms in the total figure's motions. Thus, a figure can move all its limbs either in total synchrony, with various wave motions, in alternating fashions, or in irregular ways. The periodic series generator is used six times (with six associated five genes as inputs) to determine settings for the following six motion attributes, along the whole series of joints:

- amplitude of sine wave displacement in the pitch angle of the joint
- amplitude of sine wave displacement in the veer angle of the joint
- phase offset of the pitch angle sine wave
- phase offset of the veer angle sine wave
- on-off switch to enable or disable the joint's pitch angle motion
- on-off switch to enable or disable the joint's veer angle motion

Appendix C
The Physics Model for the Articulated Figures

Here I will describe the physics model for the 2D and 3D articulated figures. Motion is determined by the use of a qualitative physics model, tailored specifically for this system. It incorporates forward dynamics—acceleration is caused by forces exerted internally within the figure by way of deformations in its internal structure (autonomously changing angles of the joints connecting limbs).

The model is simple yet produces many of the salient features of interacting objects in the real world, such as gravitational effects, inertia, angular and translational momentum, friction, and dampening. An articulated figure is treated essentially as a rigid body which can deform itself internally but cannot be deformed passively by way of outside forces such as collision with the ground surface. So, for instance, when the figure collides with the ground surface, the angles of its joints are not affected. Only the overall angular velocity and translational velocity are affected.

Collision with the Ground Surface

The only environmental agent that affects the figure is the ground, with which the figure has frequent contact due to gravity. Most of the computation happens here, each time a part of the body encounters the ground. Translational and angular velocities of the figure change—with the nature of the change depending on where the collision is in relation to the figure's center of mass, and the velocity of the point of contact upon collision. The four basic functions of this model are explained in *non-mathematical* terms, with low-tech illustrations. I will describe these collision functions for the 3D figures, since the 2D figure world can be seen as basically a subset of the 3D world.

1) vertical translational velocity

Upward motion of the figure is caused when the collision point lies *under* the center of mass. This is basically the equivalent of a *bounce*. The degree (an angle quantity) of which the point lies under the center of mass determines the degree in which downward velocity of the contact point is transferred to upward velocity in the whole figure. A dampening constant affects the amount of energy absorbed by the ground and figure, thus lessening the bounce effect. Figure C.1 illustrates this function.

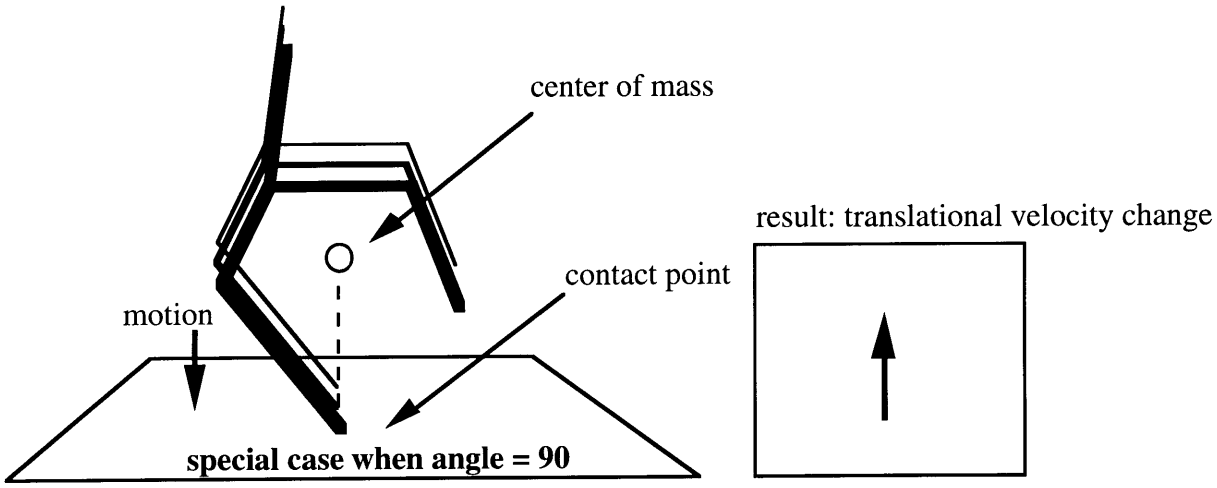


Figure C.1 Upon collision with the ground, downward motion in the collision point is converted into upward motion for the whole figure, when the contact point is below the figure's center of mass.

2) angular velocity on the XZ and YZ (vertical) planes

Downward collision with the ground causes angular velocity changes when the contact point is NOT directly below the center of mass. The degree in which the contact point is below the center of mass (an angle value) determines the degree in which the downward motion at the point of contact is converted into upward velocity in the whole figure (as illustrated in collision function 1). When this angle value is exactly 0 degrees or 180 degrees (limit cases), the vertical movement of the contact point is converted entirely into angular velocity in the vertical planes. Usually, the angular relation of the contact point to the center of mass is neither 0, 90, or 180; thus some combination of translational and angular velocity results. Figure C.2 illustrates this function.

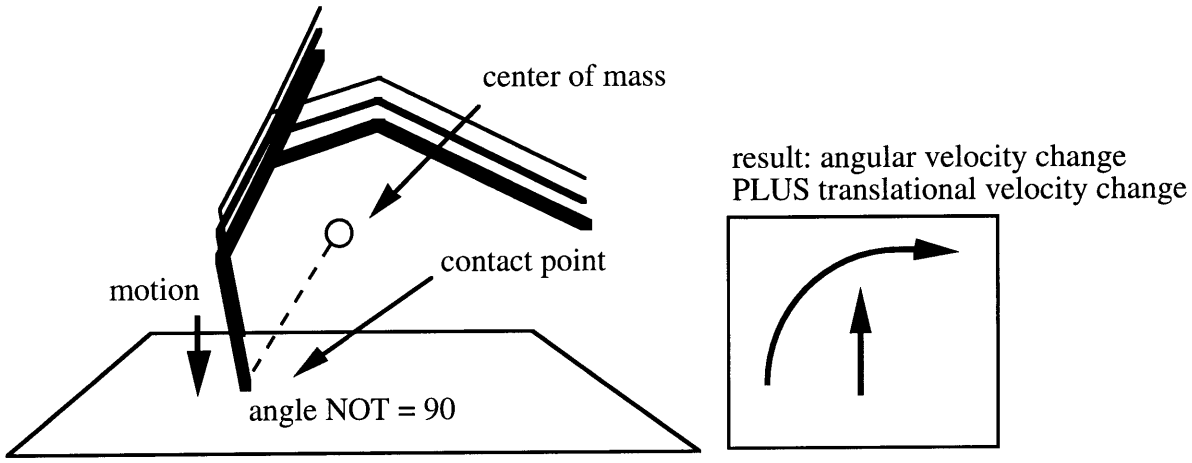


Figure C.2 Upon collision with the ground, downward motion in the collision point is converted into a combination of upward velocity and angular velocity for the whole figure, when the contact point is NOT directly below the figure's center of mass.

3) horizontal translational velocity

Due to friction, any horizontal motion of the collision point causes a degree of opposite motion in the whole figure. Here the horizontal movement in the collision point affects the amount of reaction force—when the XY components of the movement is *parallel* to an XY line connecting the contact point with the center of mass. Figure C.3 illustrates this function.

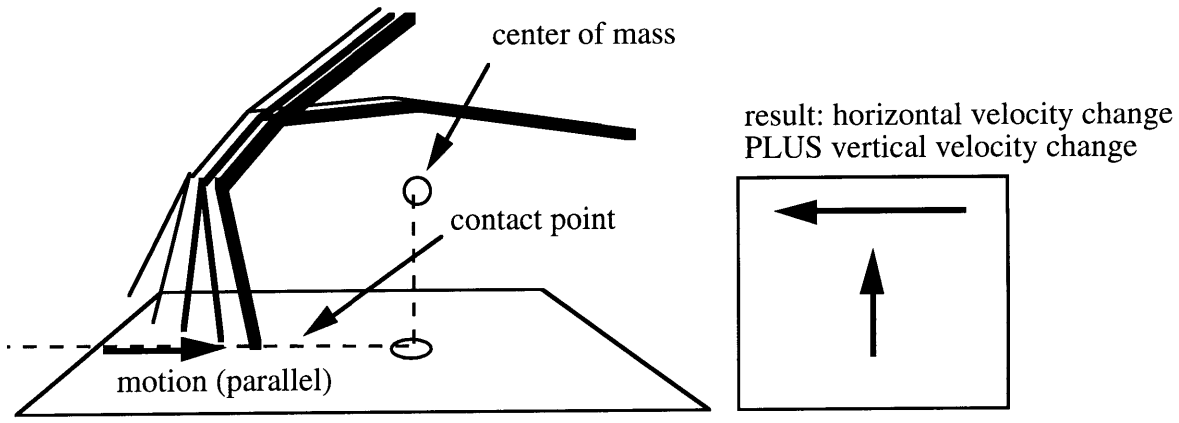


Figure C.3 Upon collision with the ground, any degree of horizontal motion in the collision point is converted into an opposite horizontal push for the whole figure, when movement point is parallel to an XY line connecting the collision point to the XY center of mass.

4) **angular velocity on the XY (ground) plane**

Angular momentum in the XY plane (around the Z, or vertical axis) is changed when horizontal movement in the contact point is perpendicular to a line connecting the contact point with the center of mass, thus describing an angular change about the center of mass. Figure C.4 illustrates this function.

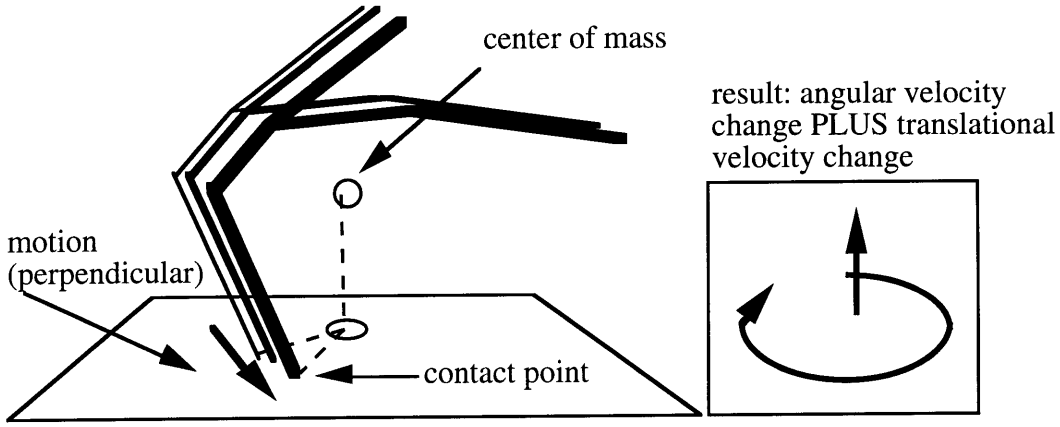


Figure C.4 Upon collision with the ground, horizontal motion in the collision point is converted into angular velocity about the vertical axis, when the motion is perpendicular to an XY line connecting the collision point to the XY center of mass.

In all of these cases, the amount of downward movement in the collision point (how much force there is in the collision) determines the amount of kinetic energy which is converted to angular or translational velocity.

REFERENCES

Badler, N., Barsky, B., Zeltzer, D. *Making Them Move*. Morgan Kaufman, 1991.

Baecker, R. *Picture-Driven Animation*. Proceedings of the Spring Joint Computer Conference 1969 pages 273-288.

Baker, E., and Margo Seltzer. *Evolving Line Drawings*. Harvard University Center for Research in Computing Technology. Technical Report TR-21-93. 1993.

Cohen, Michael. F. *Interactive Spacetime Control for Animation*. Computer Graphics 26, 2, July 1992.

Dawkins, R. *The Evolution of Evolvability*. Artificial Life (proceedings, ed. Langton) Addison-Wesley 1989. pages 201-220.

Dawkins, R. *The Blind Watchmaker*. Longman's Scientific and Technical. 1986.

Donahue, Karen. *An Intelligent Sketchpad: A Gestural Language for Denoting Temporal Behaviors in Dynamic Design*. Masters Thesis, MIT 1992.

Ginsberg, C. M. *Human Body Motion as Input to an Animated Graphical Display*. Master's Thesis, MIT, 1983

Goldberg, David. *Computer-aided Gas Pipeline Operation using Genetic Algorithms and Rule Learning*. (Doctoral dissertation, University of Michigan) Dissertation Abstracts International, 44(10), 3174B, (University Microfilms No. 8402282). 1983.

Goldberg, David E. *Genetic Algorithms in Search, Optimization, & Machine Learning*. Addison-Wesley. 1989.

Holland, J. H. *Adaptation in Natural and Artificial Systems*. The University of Michigan Press, Ann Arbor. 1975.

Hybs, Ivan, and Gero, John. S. *An Evolutionary Process Model of Design*. Design Studies Vol 13, number 3 July 1992.

Langton, Christopher, editor. *Artificial Life* (proceedings). Addison-Wesley 1991.

Levy, Steven. *Artificial Life* (from panel discussion at SIGGRAPH conference). Computer Graphics, 26, 2 July 1992. page 407.

McKenna, Michael. *A Dynamic Model of Locomotion for Computer Animation.* Master's Thesis, MIT, 1990.

Ngo, Thomas J. and Marks, Joe. *Spacetime Constraints Revisited.* Computer Graphics (SIGGRAPH Proceedings) 1993.

Oppenheimer, Peter. *The Artificial Menagerie.* Artificial Life (Proceedings - edited by Chris Langton). Addison-Wesley 1991. pages 251-274.

Resnick, Mitchel. *Beyond the Centralized Mindset* . Ph.D. dissertation, MIT, 1992.

Resnick, Mitchel. (personal communication) 1994.

Sims, Karl. *Interactive Evolution of Dynamical Systems.* Proceedings of the First European Conference on Artificial Life Cambridge. MIT Press, 1992.

Sims, Karl. *Locomotion of Jointed Figures over Complex Terrain.* Master's Thesis, MIT. OCLC NO: 17246083 1987.

Sims, Karl. *Artificial Evolution for Computer Graphics.* Computer Graphics, volume 25, number 4 (SIGGRAPH Proceedings). July 1991.

Thomas, F. and O. Johnson. *Disney Animation: The Illusion of Life.* New York, Abbeville Press. 1981.

Todd. S. and Latham, W. *Evolutionary Art and Computers.* Academic Press: Harcourt, Brace, Jovanovich. 1992.

van de Panne, M., and Fiume, E. *Sensor-Actuator Networks.* Computer Graphics SIGGRAPH Proceedings 1993. pages 335-342.

Ventrella, Jeffrey. *Explorations in the Emergence of Morphology and Locomotion Behavior in Animated Characters.* Proceedings of the Fourth Workshop on Artificial Life. MIT Press. (to be published August 1994).

Ventrella, Jeffrey. *Walker: An Experiment in Evolving Walking Behaviors in Computergraphic Figures.* (unpublished paper) Syracuse University Research Computing Services. 1992. (current email at MIT: `ventrell@media.mit.edu`).

Witkin, A. and Kass, M. *Spacetime Constraints.* Computer Graphics, volume 22 number 4 August 1988. pages 159-168.

Zeltzer, David. *Task Level Graphical Simulation: Abstraction, Representation, and Control.* from the book: Making Them Move, edited by Badler, Barsky, and Zeltzer. 1991. page 3.