2008

Morphology Independent Dynamic Locomotion Control for Virtual Characters

Adrian Boeing
Edith Cowan University

Morphology Independent Dynamic Locomotion Control for Virtual Characters

Adrian Boeing, *Member, IEEE*

***Abstract*— Physically based animation of virtual characters is an attractive technology for computer games. It enables characters to dynamically react to interactions with the environment. Existing dynamic simulation controllers are often complex to understand and manipulate, and so are of limited use for animators. This paper presents an extended spline-based control strategy similar to splines used in standard keyframe animation techniques. Unlike existing dynamic control strategies, this allows animators to modify the control system parameters in a manner similar to traditional kinematic animation techniques. A genetic algorithm is employed to produce the initial control parameters for the desired gait, and extend the parameters to enable sensory feedback. The controllers are simulated in a 3D environment and demonstrated for bipedal, tripedal and snake-like characters.**

I. Introduction

Developing animations for the locomotion of a virtual character is a time consuming task. Typically, the animated motion is created by hand using keyframes, or collected from a motion capture system. Data collected from a motion capture system is then interpolated with splines to produce smooth transitions through the keyframes. Similarly, splines are often employed to interpolate hand-crafted animation data.

The advantage of these approaches is that it provides animators direct control over the characters' motions, enabling them to specify exactly the motion they desire. However, these kinematic approaches are simply snapshots of the dynamics of the character at a given point in time. As a result, the animations are well suited to non-interactive applications where the expected dynamic reactions from any interaction between the characters and the environment (or other characters) can be manually added to the animation.

Interactive applications differ in that the interactions between objects cannot be reliably generated prior to execution and must be calculated as they occur. There are a number of approaches that can be taken to alleviate this problem. One approach is to emulate the physical plausibility of the motion by combining a suitable subset of transitions between frames from an existing key framed motion database [1]. Although this approach provides the ability to emulate a pre-recorded range of motions, it does not provide the full motion responses that a dynamically simulated character can provide [2].

A. Boeing is with the School of Computer Science, Edith Cowan University, Perth, Australia. (e-mail: a.boeing@ecu.edu.au).

An alternative approach is to employ the pre-generated kinematic animations until a situation occurs where a dynamic response is required. At this point, the state of the kinematic system is transferred to a full dynamic simulation. A simple form of this is commonly seen in computer games where rag-doll simulations (ie: uncontrolled dynamic characters) are employed when a character dies. This allows a physically realistic animation of a character falling down. Coupling a control system to the virtual character allows for a greater range of motions to be described. Zordan [3] used a feed back control system to blend motion capture animations with dynamic motions.

The switch from kinematic motions to dynamic motions requires instantaneous impulses to be applied at the beginning of the dynamic simulation. This creates difficulties in developing an appropriate control structure that can set the appropriate controller responses for matching and tracking the kinematic animation [2][3][4].

Finally, kinematic based animation can be completely replaced with a full dynamic simulation [4,5]. This allows a fully interactive animation and eliminates the problems encountered when trying to merge kinematic and dynamic animations. Furthermore, this enables the automated generation of motions (eg: locomotion) and allows the animation and interaction of fantasy characters (eg: multi-legged characters) [6]. Full dynamic simulation typically requires more complex control structures and makes the integration of existing kinematic animation techniques troublesome [4].

II. Locomotion Control Systems

Human and animal gaits have been extensively studied, starting in the late 18th century with notable researchers such as Eadweard Muybridge [7]. The demand for control systems for legged locomotion was initially driven by robotics research [4]. As early walking robots were constructed, a system for balancing the robots was required. The most difficult legged system to balance are bipeds as they are generally unstable, especially in humanoid configurations.

Statically balanced robots maintain balance by ensuring that the center of mass is within the supporting leg base area. As a result, statically balanced robots feature a small footstep, and slow speed. With further research into walking robots, dynamic walking was realized [8]. During dynamic walking, the center of gravity may lie outside the supporting leg base area during some periods of the walk cycle. The

978-1-4244-2974-5/08/$25.00 ©2008 IEEE

removal of the static balance constraint allows for a greater range, faster and more realistic gaits to be achieved [8].

There are a number of control systems that are applicable to legged locomotion. Possible control systems range from simple oscillators [9] to control algorithms such as state machines [4], to neural networks [10][11]. The simplest oscillators consist of a set of sinusoidal function generators whose outputs are combined to generate the control signal for an actuator. These systems can represent a range of outputs by altering the phase and amplitude of the sinusoids [9]. Such systems are generally incapable of expressing the complexity required for sustained locomotion [12]. Thus, more sophisticated forms of control are desirable.

A common technique for maintaining bipedal stability is the use of the Zero Moment Point (ZMP) constraint. The ZMP is the point where the sum of all moments is equal to zero. If the ZMP is within the support polygon formed by the feet, the character will be stable. If the ZMP leaves that region, the character will begin to fall. Using this constraint with the combination of pressure, sensors have been the key to the walking control of robots such as Honda's ASIMO [13]. The disadvantage of this technique is that the ZMP constraint is too tight, resulting in limited gaits, and it will only apply to legged characters and cannot be applied to general morphologies (eg: snakes).

Another common technique for locomotion control is the use of state machines [4]. This control strategy has been successfully demonstrated for a large range of morphologies and applications [4][14]. Yin et al. [4] presented a state machine based approach for the control of virtual bipedal characters. The disadvantage to the state based approach is that an appropriate number of states must be constructed for differing gaits and morphologies. Yin et al. inserted extra "dummy" states to the control system to enable the transition between different gaits for the same morphology. This approach would not scale to a general morphological configuration.

Neural networks have demonstrated stable control for a variety of morphological configurations of both robots and virtual characters [10][11][15]. Typically neural control of legged characters mimics the central pattern generator seen in real animals. The automatic generation and optimization of locomotion gaits have been demonstrated for a variety of configurations [11][15]. This makes neural networks an attractive choice for controlling generic character animations. Tuning and tweaking the control system of a neural network is a very complex task, and it is not feasible to require animators to manually adjust the neural control parameters to achieve the motion they desire.

III. SPLINE BASED CONTROLLER

Each of the outlined control strategies has their advantages and disadvantages. Simple oscillators are not capable of expressing the range of motions required by most applications, and common algorithmic approaches are tied to the morphological or gait structure. Neural networks provide flexible control, but are difficult to manually manipulate.

A spline based control system provides a greater range of motions that can be expressed than simple oscillators, whilst also providing a control structure that is understandable and familiar to animators [16]. Like neural controllers, spline based controllers have been successfully demonstrated for the automated generation of gaits for simulated and real robots [16][17][18].

Splines are piecewise polynomial functions expressed by a set of control points [19]. There are many different forms of splines, each with their own attributes. There are two desirable properties for the control splines to posses:

- Continuity, so that the generated control signal translates to smooth velocity and acceleration changes.
- Locality of the control points, to reduce the influence of alterations of one control point to the overall shape of the spline.

The spline controller comprises of a set of connected Hermite splines. Each spline can be defined by a variable number of control points allowing variable degrees of freedom. The function used to interpolate the control points, given starting point p1, ending point p2, tangent values t1 and t2, and interpolation point s is shown below:

$$f(s) = h1.p1+h2.p2+h3.t1+h4.t2 \quad (1)$$

Where:

h1=2s3-3s2+1
h2=-2s3+3s2
h3=s3-2s2+s
h4=s3-s2

Three connected splines are combined to form the overall control structure for one joint. The three splines are responsible for three different phases of the character's gait. The initial phase is responsible for blending in the gait (eg: moving the character from a stationary position into a walking motion). The second set of splines contains the repeated cyclic information for the characters gait. And finally, the end spline is used to blend the gait out (eg: moving the character safely back to a stationary position).

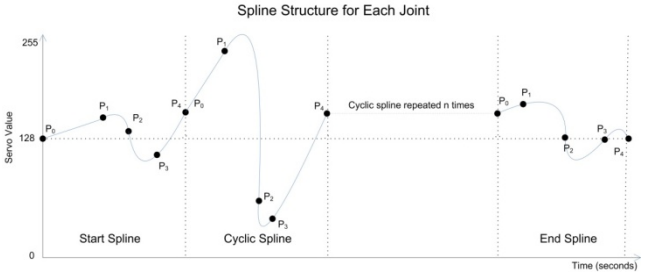

Fig. 1. Example spline controller structure.

An example of a simple spline controller is illustrated in Fig. 1. The depicted spline indicates the controller's output value for one degree of freedom (e.g.: joint servo).

The number of control points required for the simple spline controller is given by:

$$a \cdot (i + c) \tag{2}$$

Where,

a is the number of actuators

i is the number of control points in the initialization spline

c is the number of control points in the cyclic spline

Cubic Hermite splines were implemented in the controller as they offer a number of desirable characteristics over other splines. The major advantage of the Hermite spline is that the curve passes through all the control points. As the position and tangent are explicitly specified at the end of each segment, it is easy to create a continuous cyclic curve. This allows for a control representation that is far simpler for humans to manipulate using a keyframing-styled approach, whilst still possessing the desired properties to allow fast and flexible gait generation.

The output of the spline controller is coupled to a PID controller. A PID controller contains a proportional, derivative, and integral reaction to an error input, and is described in (3).

$$r(t) = K_p e(t) + K_i \int_0^t e(\tau)\, d\tau + K_d \frac{de}{dt} \tag{3}$$

Where,

$r(t)$ is the controller output,

$e(t)$ is the error value given from the difference between the desired set point and the current value,

K_p is the proportional term,

K_i is the integral term,

and K_d is the derivative term

If the spline control system is directly connected to the joint angles, then over time the accumulated errors from the open loop control causes the gait to deviate from the desired gait. This is illustrated in Fig. 2, where the characters torso is slowly dropping towards the ground.

A. Sensory Feedback

Without feedback the dynamic controller will animate in a manner very similar to that achievable by standard kinematic animation techniques. To produce an adaptive animation the control system must include feedback from the environment. This allows a character to react differently depending on whether it is walking along flat terrain, or walking up a steep incline.

In order to incorporate feedback information into the spline controller, another dimension must be added to the controller. The extended control points specify their locations within the gaits cycle time, and the feedback value. This results in a set of control surfaces for each actuator. The number of control points required for the extended controller is given by (4). Extending the controller in this form significantly increases the number of control points required. Fig. 3 illustrates a resulting control surface for one actuator.

$$a \cdot (i + c \cdot f) \tag{4}$$

Where f is the number of control points used to sample the feedback.

The actuator evaluates the desired output value from the enhanced controller as a function of the cycle time, and the input reading from the sensor. The most appropriate sensory feedback was found to be the torsos inclination towards the ground plane. Thus, the resultant controller was expressed in terms of the percentage cycle time, the torsos inclination angle, and the output control signal.

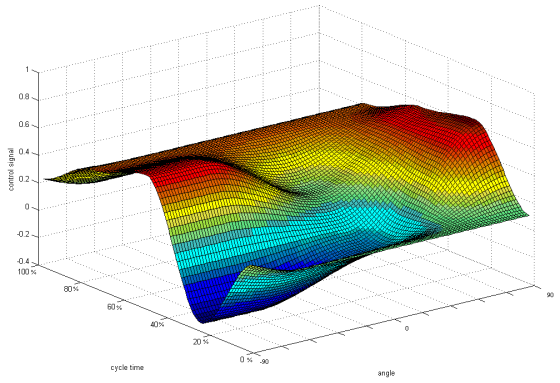

Fig. 3. A spline controller extended to include feedback.

IV. Controller Evaluation

A genetic algorithm was used to automatically generate and fit the desired characters gaits. To optimize the performance of the GA the gait was evolved in multiple phases [20].

The basic methodology for the genetic algorithm is presented below [21]:

1. Randomly initialize a population of chromosomes

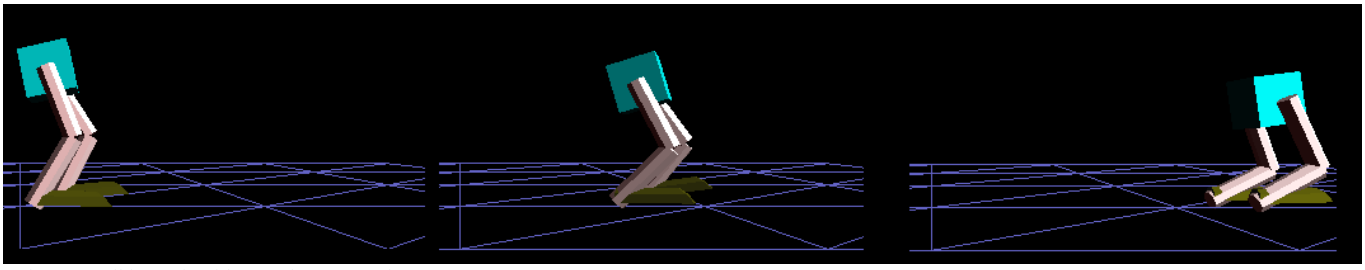

Fig. 2. Walking gait with open loop control

2. While the terminating criteria has not been satisfied
    a. Evaluate the fitness of each chromosome
    b. Remove the lower fitness individuals
    c. Generate new individuals, determined by a certain selection scheme, utilizing selected operations.

Each of the splines control points values was encoded to an 8 bit fixed point number. The PID controller parameters were evolved in a separate phase, where each PID parameter was encoded to 16 bits. The spline control values were concatenated to create the chromosome. The genetic algorithm used roulette wheel selection, and a one point crossover and single bit mutate operator. Each GA operated with a population of 50 individuals over 500 generations for automated gait generation, and 30 generations for gait fitting and PID generation.

One of the most complex tasks in evolving a valid gait is the selection of an appropriate fitness function. The fitness function must return a single numerical value indicating the appropriateness of a solution with respect to the overall goal. Since there is no straightforward performance measurement for a good gait, the function must be expressed as a combination of the desired factors.

Reeve [15] experimented with various legged robot configurations and investigated a number of fitness functions for evolving neural controllers. Although Reeve reports that the Speed5 fitness function (average speed of the walker over five seconds) performs adequately, improvement on the performance of the algorithm can be achieved through the use of more complex functions. Reeve proposed five different extended fitness functions:

- FND – (forward not down): The average speed the walker achieves minus the average distance of the center of gravity below the starting height.
- DFND – (decay FND): Similar to the FND function, except it uses an exponential decay of the fitness over the simulation period.
- DFNDF – (DFND or fall): As above, except a penalty is added for any walker whose body touches the ground.
- DFNDFA – (DFNDF active): This function incorporates features of the actual control system into its evaluation of the gait. The function evaluates the individual neurons and ensures they are active, and are not stuck at an on or off value.
- DFNDFO – (DFNDFA oscillator): As above, with the added constraints that both the neurons and legs oscillate.

Ziegler and Banzhaf [17] utilized fitness functions which compared the generated trajectory of a gait to the desired path. The trajectory was specified to include an initial acceleration, then a straight walk along the desired path, and a deceleration. The square difference of the actual walk from the desired was then summed over the duration of the gait and returned as the fitness value. In order to optimize the performance of the evolution algorithm, Ziegler and Banzhaf [17] introduced premature termination conditions to the fitness function. The premature termination condition ensured that the initial trajectory was within a valid range of the desired trajectory. Thus, if the desired trajectory were forwards movements, then any gait that produced backwards movement would be terminated.

TABLE I
GENETIC ALGORITHM PARAMETERS

Parameter	Value
Population Size	50
Generations (automated, fitting)	500,30
Crossover rate	70%
Mutate rate	30%
Selection scheme	Fitness-proportional
Encoding	Binary, fixed point

The basic fitness function implemented followed both the principles implemented by Reeve [15] and Ziegler and Banzhaf [17]. During the initial phases of evolution, the fitness is evaluated purely from the distance the character traveled forward minus the distance the character center of gravity lowered. This is a combination of aspects Reeve's FND and Ziegler's premature termination conditions [15][17]. During later phases of evolution, the average speed at which the robot moved and the distance the robot wavered from the desired path are incorporated. Finally, the distance the robot is at termination from its desired terminating point is taken into consideration to emulate the effect of Reeve's DFND.

The spline based control system was evolved for a number of robot morphologies. This included a basic bipedal character, a humanoid biped, a tripod and a snake character. The dynamics were simulated through the Physics Abstraction Layer [22][23] with Dynamechs [24], ODE [5], and Bullet physics library [26] as the back end simulators. The simple biped has large feet and lacks a complete torso. Each leg has three degrees of freedom: the foot, the knee, and at the hip. The android biped is a more realistic representation of human character, and has dimensions comparable to a human being. It has an additional degree of freedom at the hip allowing rotation around the sagittal axis. The tripod is a modification of the simple biped, with an additional leg and no feet. Finally, the snake is composed of a chain of rigid bodies, with 6 degrees of controllable freedom.

The spline controllers were configured with four control points per cyclic phase and two for the blending phases. The number of control points was not altered for generating the locomotion in all the experiments.

Fig. 4. illustrates an evolved walk cycle of a humanoid character. The character achieves locomotion by lifting and bending one of its legs, and stretching its other leg to propel it forwards. The bent leg is then placed on the ground, then the stretched leg is bent and the cycle repeats itself.

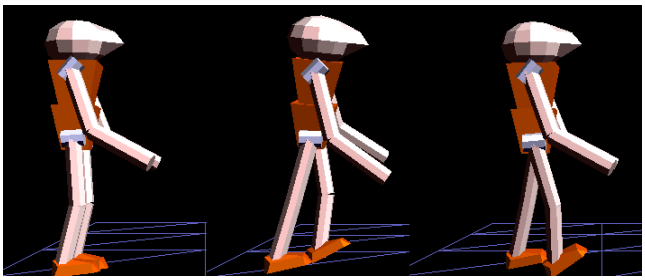

Fig. 4. Humanoid walking gait.

A static walking gait for the simple biped is illustrated in Fig. 5. Fig. 6 demonstrates a dynamic jumping gait generated for the humanoid character. These results demonstrate that the spline based approach is capable of expressing simple walking animations, as well as more complex gaits such as jumping. Unlike other approaches, both spline controllers had the exact same configuration and did not require any changes to the controller structure (such as additional "dummy" states). They only differed in the control point parameters representing the spline control signals [4].

This also demonstrates the ability for the spline control system to produce gaits that are not statically balanced, unlike control systems that enforce the ZMP constraint.

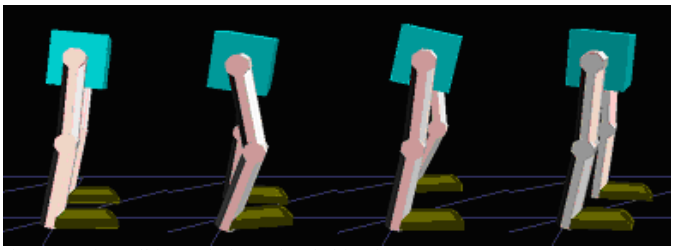

Fig. 5. Static walking gait for a simple biped.

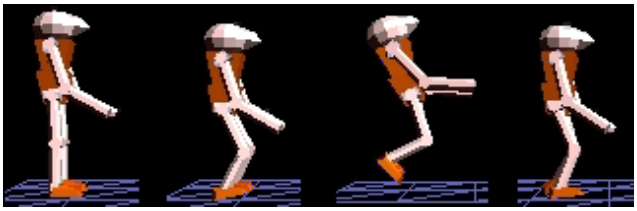

Fig. 6. Jumping gait for a humanoid biped.

Gaits for non-bipedal characters were also successfully evolved utilizing the spline control system. The most successful gait evolved for the tripod is illustrated in Fig. 7. The tripod achieves forward motion by thrusting its rear leg towards the ground, and lifting its forelimbs. The tripod then gallops with its fore limbs to produce a dynamic gait. Again, the controller structure was not altered to produce this motion, demonstrating that the spline based control system is independent of the characters morphology.

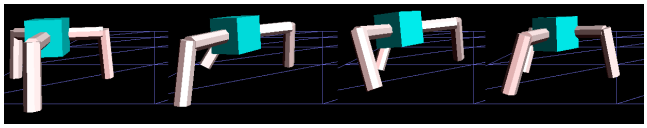

Fig. 7. Galloping gait for a tripedal character.

Since the spline control system is not dependent on particular rules (such as the ZMP, or foot contact conditions) it can be applied to non-legged characters as well. A snake-like character's movement was evolved. It produced the basic sinusoidal movement that is evident in real snakes.

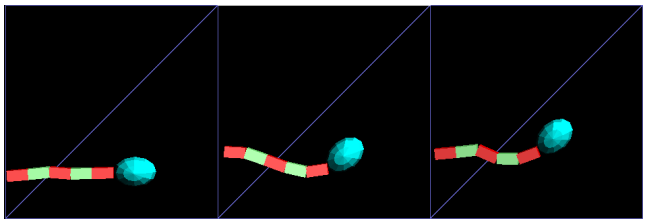

Fig. 8. Snake gait.

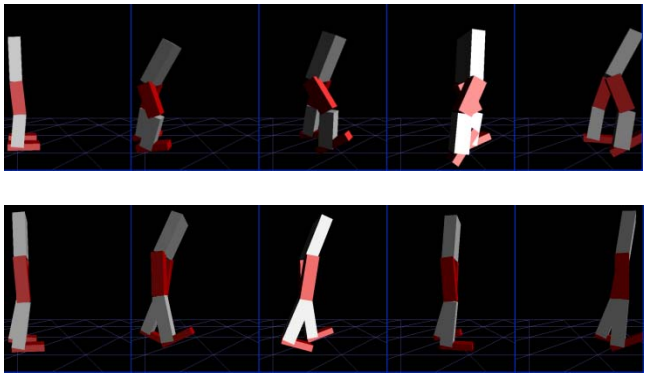

Fig. 9. Walking gaits for a biped with altered morphology.

Fig 9. depicts the same bipedal character based on the simple biped configuration. The dimensions of the biped have been altered to elongate the torso and legs to differing lengths. The second set of illustrations show a walking gait for the same biped with longer legs. This demonstrates the ability of the spline controller to adapt to slightly different morphological configurations of the same character.

Fig. 10. illustrates the basic biped with an extended controller that incorporates the angular feedback from the torso. This enables the character to successfully navigate rugged terrain.

The resultant spline control system is visualized in Fig. 11. The control parameters in the spline directly represent the control points for the spline. This enables an animator to modify the resulting control signals directly in a manner already familiar to animators. Finally, Fig. 12 illustrates the increase in fitness for the evolution of a gait.

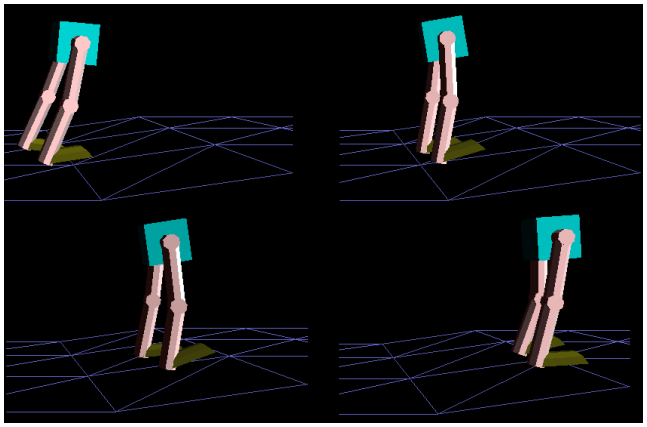

Fig. 10. Walking gait for a simple biped responding to alterations in the terrain surface.

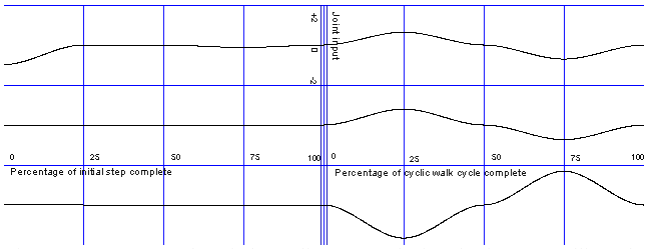

Fig. 11. An example of the spline control signals for controlling the simple biped.

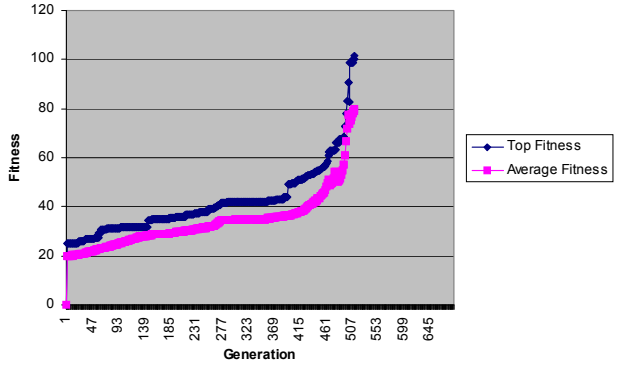

Fig. 12. Increase in average and top fitness during the evolution of a walking gait.

V. Conclusion

A spline based control system is a flexible controller representation capable of expressing a variety of gaits for a number of different character morphologies. It is not limited to human-like bipeds, and can be applied to non-legged characters. A genetic algorithm can be employed to optimize manually created gaits, or to autonomously generate complete gaits. The controller can be extended to incorporate sensory feedback, and provides an intuitive representation that enables easier manual tweaking of the control parameters.

There are many avenues to investigate further. Whilst the control system provided acceptable gaits for simple morphologies with simple sensors, more investigation is required for generating locomotion for complex morphologies and complex sensor configuration. A number of enhancements and investigations could be made to the genetic algorithm and fitness functions to determine the optimal configurations. A fitness function system with a multiobjective evolutionary algorithm could provide users an intuitive method for generating gaits. Methods for quickly adapting the control splines to slight changes in morphology could be explored, and the ability to blend a wider range of behaviours together needs to be developed.

Acknowledgments

The author would like to thank T. Bräunl for his advice.

References

[1] L. Kovar, M. Gleicher, and F. Pighin. “Motion Graphs.” in *Proc. ACM SIGGRAPH*, 2002. pp 473-482.

[2] P. Wrotek, O.C. Jenkins, and M. McGuire. “Dynamo: Dynamic, data-driven character control with adjustable balance.” in *ACM Sandbox Symposium on Video Games*. 2006. pp 61-70

[3] V. B. Zordan. “Motion capture-driven simulations that hit and react.” in *ACM SIGGRAPH Symposium on Computer Animation*. 2002 pp 89-96

[4] K. Yin, K. Loken, and M. Panne, “SIMBICON: Simple Biped Locomotion Control.” in *Proc. ACM SIGGRAPH* . 2007

[5] J. K. Hodgins, W. Wooten, D. Brogan, and J. O'Brien. “Animating Human Athletics.” in *Proc. ACM SIGGRAPH*, 1995. pp 71-78

[6] K. Sims, “Evolving Virtual Creatures.” in *Proc. ACM SIGGRAPH*, 1994. pp 15-22.

[7] E. Muybridge, 1887. *Animals in Motion.*

[8] A. Takanishi, M. Ishida, Y. Yamazaki and I. Kato. “The realization of dynamic walking by the biped walking robot WL-10RD.” in *Proceedings of the ICAR,* 1985 pp 459-466.

[9] J. Ventrella, “Explorations in the Emergence of Morphology and Locomotion Behavior in Animated Characters.” in *Proceedings of the Fourth Workshop on Artificial Life* 1994 pp 436-441

[10] M. Lewis, “Genetic Programming approach to the Construction of a Neural Network for Control of a Walking Robot.” in *IEEE International Conference on Robotics and Automation,* 1992 pp 2618-2623.

[11] G.S. Hornby, S. Takamura, S. Yokono, J. Hanagata, O. T. Yamamoto, M. Fujita. “Evolving Robust Gaits with AIBO.” in *IEEE International Conference on Robotics and Automation*, 2000 pp 3040-3045.

[12] D. Arnold. “Evolution of Legged Locomotion.” MSc. Thesis, Simon Fraser University, School of Computing Science. 1997

[13] K. H. Hirai. “The development of Honda humanoid robot.” in *IEEE Conference on Robotics and Automation*, 1998 pp 1321-1326.

[14] M. Lewis. “Genetic Algorithms for Gait Synthesis in a Hexapod Robot.” in *Recent Trends in Mobile Robots*, 1994 pp 317-331.

[15] R. Reeve, “Generating walking behaviours.” PhD Thesis, University of Edinburgh, Institute of Perception, Action and Behaviour. 1999

[16] A. Boeing, T. Bräunl. “Evolving Splines: An alternative locomotion controller for a bipedal robot.” in *Proceedings of the Seventh International Conference on Control, Automation, Robotics and Vision*, 2002 pp.5.

[17] J. Ziegler, and W. Banzhaf, “Evolution of Robot Leg Movements in a Physical Simulation.” in *Proceedings of the Fourth International Conference on Climbing and Walking Robots, CLAWAR*. 2001 pp. 395-402

[18] A. Boeing, S. Hanham, T. Bräunl. “Evolving Autonomous Biped Control from Simulation to Reality.” in *Proceedings of the Second International Conference on Autonomous Robots and Agents*, 2004 pp 440-445.

[19] R. H. Bartels, *An Introduction to Splines for use in Computer Graphics and Geometric Models*. Morgan Kaufmann. 1987

[20] A. Boeing, T. Bräunl. “Evolving a Controller for Bipedal Locomotion.” in *Proceedings of the Second International Symposium on Autonomous Minirobots for Research and Edutainment,* 2003. pp 43-52.

[21] D. Goldberg, *Genetic Algorithms in Search Optimization and Machine Learning*. Addison-Wesley. 1989

[22] A. Boeing, T. Bräunl. “Evaluation of real-time physics simulation systems.” in *Proceedings of the 5th International Conference on Computer Graphics and Interactive Techniques in Australia and Southeast Asia,* 2007 pp. 281 - 288.
[23] Physics Abstraction Layer.[Online] http://pal.sourceforge.net/ (accessed August 14, 2007).
[24] Dynamechs (Dynamics Of Mechanisms): A Multibody Dynamic Simulation Library. [Online] http://dynamechs.sourceforge.net/ (accessed August 14, 2007).
[25] Open Dynamics Engine. [Online] http://www.ode.org/ (accessed August 14, 2007).
[26] Bullet Physics Library. [Online] http://bullet.sourceforge.net/ (accessed August 14, 2007).