Simple Machines for Scaling Human Motion

Nancy S. Pollard
Brown University[1]

Abstract. This paper describes a fast technique for modifying human motion sequences in a way that preserves physical properties of the motion. Reference motion may be obtained from any source: motion capture data, keyframed motion, or procedurally generated motion. We show that by deriving a simplified control system *from motion data only*, we are able to modify the motion in a physically realistic way at nearly real-time speeds, because we can scale and modify the simplified system directly. To demonstrate the effectiveness of this approach, we animate running motion for a variety of characters over a range of velocities. Results can be computed at several frames per second.

1 Introduction

If animated characters in virtual environments are to be believable, they must exhibit a high level of adaptability in the performance of motion tasks. Because high quality motion is expensive to generate, it is important to make the best use of motion we already have. Much effort has been spent investigating how existing motion can be modified to fit new situations. Current approaches include motion interpolation, constrained optimization, and use of a control system.

One difficulty with many current approaches to motion warping is the effort required on the part of the designer or animator to precisely define the task or behavior. For example, when a control system is created, task information is embedded within that control system, and the burden lies with the designer to make it very general. In a constrained optimization approach, the user must specify a set of constraints and an optimization function that adequately describe a task. Choosing a good set of constraints and a good optimization function is a difficult problem, and recent research has focused on techniques that allow this information to be interactively modified.

This paper explores an alternative representation of a task that provides fast performance and may be easier to specify. A task is defined based on the simplest machine capable of performing the activity. Simple machine analogies are used to scale a running motion to new characters, change a character's running velocity for steady state running, and allow a running character to accelerate and decelerate (see Appendix).

To achieve fast performance, we give up complete physical realism – the resulting motion is only guaranteed to reflect the physically correct behavior that is captured in the simple machine approximation. We will show, however, that the quality of the results compares favorably to results from previous work in scaling an entire control system to new characters [6]. The advantages demonstrated in the current paper are vastly improved computation speed and an ability to make a greater variety of changes to the original motion.

⋆ Box 1910, Computer Science Department, Brown University, Providence, RI 02912, nsp@cs.brown.edu

2 Background

A variety of techniques have been used to alter existing motion, including interpolation, constrained optimization, and dynamic scaling of control system parameters. Techniques such as inverse kinematics may also be used to modify motion while meeting constraints such as maintaining balance [13][8]. Interpolation or blending of joint angle curves can be used when many examples are available (e.g. [17][3][15]). The main advantage of working with joint angle curves is speed. The motion must be postprocessed, however, to avoid constraint violations, and the results may not be physically plausible.

If only a single reference motion (or no reference motion) is available, constrained optimization may be used. An optimization approach that involves forward simulation can create physically plausible results, as in [20][12]. The main disadvantage of this approach is the time required to perform optimization over a large number of degrees of freedom. Popović and Witkin's [12] work is the most similar to that presented here, because it uses a simplified version of a human character to make the computational problem tractable. Their approach still requires minutes to generate a new motion, however, because forward simulation is required to evaluate a point in the search space. Optimization need not involve forward simulation, as in [3][5][18]. Many of these approaches are fast enough to support interactive modification of constraints by an animator. They achieve good time performance by avoiding forward dynamic simulation within the optimization loop, but give up guaranteed physical realism as a result.

If a control system is available, some amount of flexibility will be designed into that control system. Full and partial dynamic systems have been used to animate complex figures since at least the mid-80's [4][2][14][16][7][9]. Creation of robust and flexible control systems has consistently proven difficult, however, and the animator or programmer must have extensive knowledge about the details of the behavior. Control system parameters can be dynamically scaled to obtain additional flexibility, as in [6]. This approach works over a limited range of characters, however, and it requires a great deal of time, because it includes a search that must be performed over high-level control parameters.

In this paper, we build on the dynamic scaling approach of [6], but work with approximations of the full human system to make scaling tractable. We also work only with the reference motion data, not the full control system, to make the results applicable to other motion sources such as motion capture and keyframed data. We show that it is possible to maintain physical realism for the most important characteristics of a task, while avoiding much of the search required in other approaches, so that a user can vary parameters such as the character geometry, character scale, and motion velocity and see the effects at interactive speeds.

3 Approach

To modify a motion sequence, we follow the process outlined in Figure 1. The only inputs required are a reference motion, a description of the character (with minimal information about physical properties), and the requested modifications. The *reference motion* consists of joint angles and body root position over time, and can come from

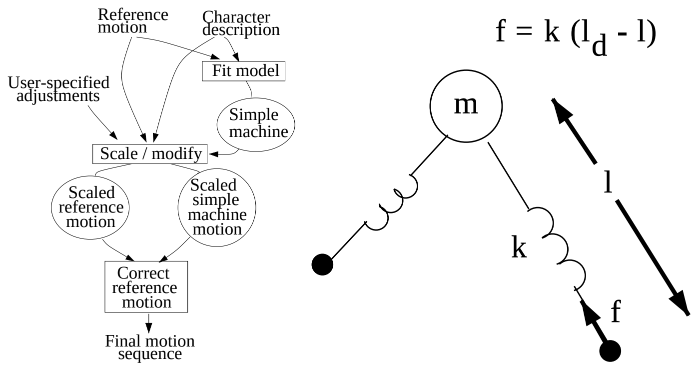

Fig. 1. (Left) Block diagram of the scaling process. (Right) The task model for running is a point mass. Force is applied along the line from the stance foot to the mass point.

any source, including motion capture or keyframed motion. The *character description* consists of joint locations, degrees of freedom, and physical parameters. The only required physical parameters are the total mass and center of mass location for each body part (e.g. the lower leg). The *user-specified adjustments* can be a new character description, new velocity, and/or a new acceleration. The output is the *final motion sequence*. Because the scaling operations can be performed rapidly, the user can immediately see the effects of their modifications.

This scaling algorithm has three key steps. The first step is to fit a simple task model to the reference motion data. This step requires some knowledge of the task. A simple model for jumping, for example, may differ from a simple model for running. The task model need only be defined once for each new behavior, however. The second step is to scale and modify both the reference motion and the simple machine version of this motion. The simple machine motion can be scaled in a physically correct manner using the direct techniques described in this paper. The reference motion cannot, however; direct scaling of the reference motion may cause the character to violate some obvious physical constraints such as maintaining balance or maintaining ground contact. The third step is to correct the scaled reference motion so that it matches the scaled motion of the simple machine. This ensures that the final motion sequence does obey the most important physical laws: those incorporated into the simple machine description of the task. The sections below describe this three-step process for a variety of examples derived from a single reference running motion.

4 Fit Abstract Model

The reference motion data contains no explicit model of the task or motion strategy. The role of the simple machine approximation is to develop such a model based on the input data. This physically-based representation of the motion makes it possible to modify the motion in a physically-plausible way.

Running motion from a physically-based simulation described in Hodgins et al. [7] is used for all examples shown in this paper. We used only the motion itself and the character description from this simulation. Similar information could easily be obtained from motion capture, keyframed data, or other sources.

The task model used for running is shown in Figure 1 (right). It consists of a point mass and two feet. The point mass represents the center of mass of the character. It is connected to each foot by a spring. In Figure 1, m is the total system mass, f is the ground contact force, k is the spring stiffness, l the actual leg length, and l_d the desired leg length. McMahon [10] uses a similar model in his analyses of human runners.

Ground forces for this task model are reverse-engineered from the acceleration of the center of mass. Motion curves for the character center of mass are extracted from the reference motion data and differentiated twice to obtain acceleration $\underline{a}_{com}(t)$. Force can then be expressed as:

$$\underline{f}_{leg}(t) = m(\underline{a}_{com}(t) - \underline{g}) \tag{1}$$

Two separate approximations were used for line of action of the force. Each approximation captures, on average, 80% of the total force represented in equation 1. In the model shown in Figure 1, forces are assumed to lie *only* along the leg. The second model assumes forces only in the vertical direction. When pure vertical forces are assumed, the ability to speed up or slow down is lost, as is any side-to-side motion of the body. The model is useful, however, for scaling steady state velocity (Section 5.2).

Control equations for the simple machine are fit to the measured ground force information to form a complete description of the simple running machine. The leg force of the simple machine is controlled as follows:

$$f_{leg}(t) = k(l_d(t) - l), \quad l_d(t) = l_0 + a + bt + ct^2 \tag{2}$$

where l_0 is the leg length at touchdown, t ranges from 0 to 1 over a single stance phase, and a, b, and c are coefficients modulating desired leg length. We will call Equation 2 the *control system* for this simple machine. A second order model for desired leg length was the simplest model that provided an adequate fit to the data. Adding damping parameters to the model did not result in a better fit or allow us to create a simpler model for this particular dataset.

Parameters $f_{leg}(t)$ and l_0 are measured properties of the motion. The remaining four parameters (a, b, c, and k) are fit to the input data over a single stance phase using a least squares solution. A search is then performed that allows initial state as well as the four model parameters to vary in order to obtain a repeatable running cycle. The resulting simple machine will run with steady-state velocity, center of mass trajectory, and feet trajectories very similar to those of the original motion.

Figure 2 (left) shows the desired leg length l_d and the actual leg length plotted over time for a single stance phase. Figure 2 (right) compares the vertical motion of the center of mass for the reference motion to that of the simple model over one running stride. The fit is qualitatively quite good. Most of the differences are due to the fact that the simple machine data was postprocessed to create a repeatable running cycle.

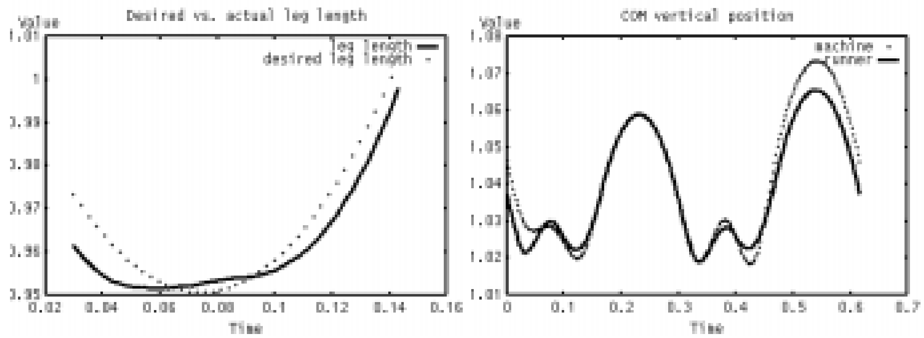

Fig. 2. (Left) Desired leg length (dotted line) plotted vs. actual leg length (solid line) over one stance phase. (Right) Simple machine center of mass height (dotted line) is compared to reference center of mass height (solid line). One stride of the running motion is shown.

5 Scale / Modify Motion

Once a simple machine representation of the reference motion is available, the motion parameters can be changed to meet new objectives. This section describes scaling the motion to new characters, changing steady-state velocity, and adding acceleration and deceleration to the running motion.

5.1 Scale to New Characters

The motion of the simple machine can be accurately scaled to a new character based on a relative length measurement and a relative mass measurement. The analysis is based on dimensionless groups. One set of dimensionless groups that describes the simple running machine is:*

$$\theta, \quad \frac{gT^2}{L}, \quad \frac{x}{L}, \quad \frac{fT^2}{ML}, \quad \frac{kT^2}{M} \tag{3}$$

where θ is any angle measurement, g is acceleration due to gravity, T is any time measurement, L is length, x is a position, M is mass, f is force, and k is stiffness. If these quantities are the same for two characters, then the two motions are *dynamically similar*.

We know that gravity remains the same for the two characters. This information, plus a relative length measurement, tells us how to scale time T and position x. (For example, if L scales by a factor of two, T^2 must scale by a factor of two to keep the second dimensionless group constant.) In addition, a relative mass measurement allows us to scale force f and stiffness k. Thus, if we know the relative length parameters and the relative masses of the two characters, we can scale the control parameters of the simple machine and re-simulate to obtain the new motion. Equivalently, we can

* Note that radians are dimensionless, and so θ on its own forms a dimensionless group. For a discussion of how to create and use dimensionless groups, see for example [19].

Parameter	Units	Scale Factor
Time	s	$L^{\frac{1}{2}}$
Positions	m	L
Velocities	m/s	$L^{\frac{1}{2}}$
Angular Positions	$radians$	1
Angular Velocities	$radians/s$	$L^{\frac{-1}{2}}$

Table 1. Scale factors for simple machine motion, based on a single relative length parameter.

scale the output motion and avoid the forward simulation step. Rules for scaling output motion to a new character are shown in Table 1.

5.2 Scaling Steady-State Velocity

Velocity cannot be scaled using principles of dynamic similarity. Steady-state velocities can be scaled directly, however, in the special case where ground forces are applied only in the vertical direction. Eliminating horizontal forces will prevent the character from speeding up or slowing down, so this approximation is limited to steady-state running.

We begin with the observation that flight time for running tends to remain fairly constant over a range of velocities [11]. In other words, the vertical trajectory of the ballistic component of the running motion will be roughly the same for different velocities. If forces are applied only in the z-direction, velocity can be scaled by stretching the paths of the feet and center of mass horizontally, while keeping the total length of time over the motion (and hence of the ballistic portions of the motion) the same.

The horizontally-scaled motion can be generated using the same laws of physics and the same control system as the original motion, but it has a different steady-state velocity as desired. We can see how this works by examining the expressions for acceleration $\underline{a}_{com}(t)$ and velocity $\underline{v}_{com}(t)$ when only vertical forces are allowed:

$$\underline{a}_{com}(t) = \begin{bmatrix} 0 \\ 0 \\ \frac{f_{leg}(t)}{m} + g \end{bmatrix}, \quad \underline{v}_{com}(t) = \begin{bmatrix} A \\ B \\ v_{com,z}(t) + \int_0^t a_{com,z}(t)dt \end{bmatrix} \tag{4}$$

where $f_{leg}(t)$ is the vertical ground contact force, m is the mass of the character, and g is acceleration due to gravity. The vertical control system has been completely separated from horizontal velocity. Velocity constants A and B can be changed without interfering with the control system or the vertical motion of the center of mass over time. In other words, horizontal velocity can be scaled by a factor s simply by multiplying terms A and B by factor s. The resulting motion matches patterns found in biomechanical measurements of running [11] during flight. It does not match during stance, however, because the stance time does vary somewhat with velocity. Adapting the control law to achieve this effect is a topic of future work.

5.3 Acceleration and Deceleration

Acceleration and deceleration cannot be achieved with a simple set of scaling laws. However, motion can be rapidly scaled to add acceleration and deceleration through the use of a transition table for the simple machine.

The simple machine's state can be described by two parameter values measured at the top of the flight phase of a stride: dimensionless horizontal velocity $U = \frac{u}{\sqrt{gl_0}}$ and dimensionless vertical height $H = \frac{h}{l_0}$, where u is the horizontal velocity, g is acceleration due to gravity, l_0 is leg length at touchdown, and h is vertical height.[⋆⋆] Given a state, a desired touchdown angle for the leg, θ_0, and a control algorithm c, we can simulate the next stride and compute the next state. In other words, a physically-based forward simulation S will take a state, a touchdown angle, and a control algorithm to a new state at the top of the next flight phase:

$$< U, H, \theta_0, c > \stackrel{S}{\rightarrow} < U', H' > \qquad (5)$$

Expression 5 can be used to compute a transition table from one state to the next based on touchdown angle θ_0. The transition table can be computed offline. The use of dimensionless parameters ensures that the transition table applies for any character. This transition table can be used to select a series of touchdown angles to control acceleration or deceleration as specified by the user.

6 Correct Reference Motion

Once a character's motion has been scaled, that motion is mapped back to the complete character using the simplest possible technique: inverse kinematics at each frame of the animation. The center of mass of the character and the positions of the feet are incrementally moved toward their desired positions until the error is visually imperceptible. A single marker is matched for the center of mass, and three markers are used for each foot to preserve foot orientation.

The inverse kinematics problem is set up as follows:

$$dX = Jd\theta, \quad dX = [\Delta \underline{M}_0,\ \Delta \underline{M}_1,\ ...,\ \Delta \underline{M}_n,\ \Delta \underline{R},\ \Delta \underline{\theta}]^T \qquad (6)$$

where dX represents desired motion, $d\theta$ is the unknown character motion, J the system Jacobian, $\Delta \underline{M}_i$ is the desired change in position of marker i (e.g. center of mass position), $\Delta \underline{R}$ is the desired change in root translation and orientation, and $\Delta \underline{\theta}$ is the desired change in joint angles. Desired motion dX is designed to move the center of mass and the feet toward their desired positions and to discourage large deviations from the reference motion at any particular joint. Note that this system is overconstrained, and so the final marker positions will not be exactly the same as the simulated positions.

Using this approach, we expect to see poor continuity, because the solution at one frame may differ from the solution found at the next frame. In practice, however, this naive approach works well, due to the relatively close fit of the original reference motion.

[⋆⋆] Note that U is the Froude number commonly used, for example, in hydraulic engineering. Alexander [1] used it to estimate speeds from stride lengths.

7 Results

Results for various types of scaling are shown in the Appendix. The *first row* shows the original man runner, a physically-based simulation developed by Hodgins [7], and the *second row* shows shows a fast version of the running motion (1.2 times the speed of the original). The *third row* shows results from the acceleration / deceleration technique discussed in Section 5.3. The runner is in the process of slowing to half speed over several strides. The segment shown occurs 1.7 seconds into the motion, when the character's velocity is $3.3m/s$ (compared to $4.6m/s$ for the original runner). The *fourth row* and the *fifth row* show the results of scaling the original motion to two new characters with very different physical characteristics. The child has been slowed to a velocity 0.7 times the scaled velocity. The *sixth row* shows the results of previous work in scaling the entire control system of the original runner to new characters [6]. Although the characters in the fifth and sixth rows are running at the same velocity, their running motions are different. The motion created using the simple machine approximation appears to be more balanced. On the other hand, the child with the full control system runs with his feet planted further apart, which may be required for lateral stability. These differences point to improvements that could be made in both systems. Animations of the results shown in this paper can be viewed at the following web page: *http://www.cs.brown.edu/people/nsp/simpleMachine.html*.

8 Discussion

This paper demonstrates that motion scaling can be achieved at interactive speeds while maintaining a good level of physical realism. With the very basic inverse kinematics approach used in these examples (implemented by the author), one to three frames could be computed and displayed per second on a 200 MHz Sun workstation.

Although the results shown here are very promising, much can still be done to improve the quality of the resulting motion. One area of future work is to explore the value of more complex machine representations such as including the angular momentum of the runner or the effect of swinging the arms. The effectiveness of this approach must also be tested for motions other than running. Finally, the reference motion that is responsible for the good quality of these results also represents a potential handicap. Quirks in the original motion will remain in scaled versions. Perhaps one or more example motions can be used as references in developing a robust strategy for performing a task. Developing robust motion strategies from examples is an interesting open area of research.

Acknowledgments

I would like to thank Jessica Hodgins for use of the runner simulation and motion data, for introducing me to the scaling problem, for technical discussions, and encouragement. Thanks also to Audrey Wong and the anonymous reviewers for many helpful comments on drafts of this paper.

References

1. ALEXANDER, R. M. Estimates of speeds and dinosaurs. *Nature 261* (1976), 129–130.
2. BRUDERLIN, A., AND CALVERT, T. W. Goal-directed, dynamic animation of human walking. In *Computer Graphics (SIGGRAPH 89 Proceedings)* (July 1989), vol. 23, pp. 233–242.
3. BRUDERLIN, A., AND WILLIAMS, L. Motion signal processing. In *SIGGRAPH 95 Proceedings* (Aug. 1995), Annual Conference Series, ACM SIGGRAPH, Addison Wesley, pp. 97–104.
4. GIRARD, M. Interactive design of 3d computer-animated legged animal motion. *IEEE Computer Graphics and Applications June* (1987), 39–51.
5. GLEICHER, M. Motion editing with spacetime constraints. In *Proceedings of the 1997 Symposium on Interactive 3D Graphics* (Providence, RI, Apr. 1997), pp. 139–148.
6. HODGINS, J. K., AND POLLARD, N. S. Adapting simulated behaviors for new characters. In *SIGGRAPH 97 Proceedings* (1997), ACM SIGGRAPH, Addison Wesley, pp. 153–162.
7. HODGINS, J. K., WOOTEN, W. L., BROGAN, D. C., AND O'BRIEN, J. F. Animating human athletics. In *SIGGRAPH 95 Proceedings* (Aug. 1995), Annual Conference Series, ACM SIGGRAPH, Addison Wesley, pp. 71–78.
8. KO, H., AND BADLER, N. I. Animating human locomotion with inverse dynamics. *IEEE Computer Graphics and Applications* (March 1996), 50–59.
9. LASZLO, J., VAN DE PANNE, M., AND FIUME, E. Limit cycle control and its application to the animation of balancing and walking. In *SIGGRAPH 96 Proceedings* (Aug. 1996), Annual Conference Series, ACM SIGGRAPH, ACM Press, pp. 155–162.
10. MCMAHON, T. A., AND CHENG, G. C. The mechanics of running: how does stiffness couple with speed? *Journal of Biomechanics 23* (1990), 65–78.
11. NELSON, R., BROOKS, C., AND PIKE, N. Biomechanical comparison of male and female distance runners. *Annals of the NY Academy of Sciences 301* (1977), 793–807.
12. POPOVIĆ, Z., AND WITKIN, A. Physically-based motion transformation. In *SIGGRAPH 99 Proceedings* (Aug. 1999), Annual Conference Series, ACM SIGGRAPH, ACM Press.
13. R. BOULIC, R. MAS, D. T. Position control of the center of mass for articulated figures in multiple support. In *6th Eurographics Workshop on Animation and Simulation* (1995), pp. 130–143.
14. RAIBERT, M. H., AND HODGINS, J. K. Animation of dynamic legged locomotion. In *Computer Graphics (SIGGRAPH 91 Proceedings)* (July 1991), T. W. Sederberg, Ed., vol. 25, pp. 349–358.
15. ROSE, C. F., COHEN, M. F., AND BODENHEIMER, B. Verbs and adverbs: Multidimensional motion interpolation. *IEEE Computer Graphics and Applications September/October* (1998), 32–40.
16. TU, X., AND TERZOPOULOS, D. Artificial fishes: Physics, locomotion, perception, behavior. In *SIGGRAPH 94 Proceedings* (July 1994), Annual Conference Series, ACM SIGGRAPH, ACM Press, pp. 43–50.
17. UNUMA, M., ANJYO, K., AND TAKEUCHI, R. Fourier principles for emotion-based human figure animation. In *SIGGRAPH 95 Proceedings* (Aug. 1995), Annual Conference Series, ACM SIGGRAPH, Addison Wesley, pp. 91–96.
18. VAN DE PANNE, M. From footprints to animation. *Computer Graphics Forum 16*, 4 (Oct. 1997), 211–223.
19. WELTY, J. R., WICKS, C. E., AND WILSON, R. E. *Fundamentals of momentum, heat, and mass transfer*. Wiley, New York, 1984.
20. WITKIN, A., AND KASS, M. Spacetime constraints. In *Computer Graphics (SIGGRAPH 88 Proceedings)* (Aug. 1988), J. Dill, Ed., vol. 22, pp. 159–168.

Fig. 3. Original and scaled runners. Snapshots are spaced at 0.066s intervals. (1) The original man runner. (2) Running at 1.2 times the original velocity. (3) After decelerating from $4.6m/s$ to $3.3m/s$. (4) Running motion scaled to troll. (5) Scaled to child and slowed to 0.7 times scaled velocity. (6) The child from [6].