Simple Steps for Simply Stepping

Chun-Chih Wu, Jose Medina, and Victor B. Zordan

University of California, Riverside
{ccwu,medinaj,vbz}@cs.ucr.edu

Abstract. We introduce a general method for animating controlled stepping motion for use in combining motion capture sequences. Our stepping algorithm is characterized by two simple models which idealize the movement of the stepping foot and the projected center of mass based on observations from a database of step motions. We draw a parallel between stepping and point-to-point reaching to motivate our foot model and employ an inverted pendulum model common in robotics for the center of mass. Our system computes path and speed profiles from each model and then adapts an interpolation to follow the synthesized trajectories in the final motion. We show that our animations can be enriched through the use of step examples, but also that we can synthesize stepping to create transitions between existing segments without the need for a motion example. We demonstrate that our system can generate precise, realistic stepping for a number of scenarios.

1 Introduction

Motion capture playback, blending, and modification have become the standard suite of motion synthesis tools used for animating video games and increasing number of feature animations. A standard practice associated with such applications is the generation of 'transitions' which combine two motion sequences together to create one longer sequence. Transitions also play important role in increasing responsiveness of character in interactive applications. However, the quality of the resulting motion depends on the choice of method(s) for creating the transition. Tell-tale artifacts of a poor transition in general include unnatural foot sliding and physically implausible motion. Groups of researchers have addressed these issues by explicitly removing so-called 'foot skate' [1, 2] and by enforcing various physical characteristics during motion transition [3, 4].

In video games especially, foot skate often appears when a character transitions from standing in one configuration to another. Unless the feet are perfectly aligned, naive transition techniques will induce unnatural foot sliding as the character goes from the initial to the ending stance. A difficult problem in this scenario is to create a transition which accounts for the differences in the placements of the feet while also accounting for the movement of the body in a realistic manner. We observe that a human actor would tackle similar scenarios by shifting the weight of the body and taking a step to re-position the feet. We

propose that a similar mechanism for stepping is necessary to generate a plausible transition. Based on this insight, a new problem arises during the special conditions of transitions where a character begins and ends standing.

This paper introduces a general method for synthesizing stepping actions in humanoid characters. While the technique is showcased in conjunction with a motion database of examples which enrich the final motion, the power of the approach comes from our models which drive the character using idealized, parametric trajectories for the stepping foot and projected center of mass. We show that these trajectories can be simple mathematical functions built empirically from observations of example stepping movements and parameterized to be controlled by key features of the desired action. The result is a stepping system that allows a character to 'transition' from one double-stance pose to another automatically by *stepping.* The simplicity of the approach lends itself to being adopted easily and immediately by game developers and technical animators alike. To show the generalness of the results, we demonstrate example animations for a variety of scenarios.

2 Related work

Editing motion capture data has been the focus of an immense number of publications in the past two decades, too many to mention here individually. However, we focus our discussion of the related work on the domain of motion transitions and previous research related to balanced stepping.

Since the introduction of automatic data re-use using motion graphs [5–7], the concept of generating transitions has become increasingly popular. Some contributions highlight ideal lengths for transitions [8], use of multiple frames to generate high quality transitions [9] and ways to generate transitions with constraints [10]. Transitions are typically performed by blending the root and joint angles over time. Cleaning up transitions is usually done with an algorithm for correcting foot motion [1, 2] and inverse kinematics (IK). Tools for correcting balance ensure that motion transitions remain physically plausible by controlling the center of mass (COM) or zero moment point [11, 12]. Closely related research includes balance maintenance for stepping activities [13, 14]. However, these projects are focused on choosing how to step in response to a disturbance as opposed to our goal which is to generate a specific step based on a desired foot placement. In this manner, we are more aligned with researchers interested in driving animation with footprints, as in [15].

Researchers in robotics have also proposed techniques for automatically generating stepping motion for use in control of humanoid robots. Similar challenges in this area include choosing step location and maintaining balance. Various numerical values have been introduced to define balance (for a summary see [16]) and many balance control papers have appeared. One big difference between our goals and those of roboticists is that we care more about the simplicity of the solution than on precise control as long as our technique does not sacrifice visual quality. One simplifying model is to treat the dynamics as an inverted

pendulum [17] and to control the robot to perform stepping based on a point mass and massless leg. We use such a conceptual model to dictate the motion of the COM for a stepping action. Another group of researchers introduce the concept of the "capture point" which is the step placement (point) which yields a single step to recover from a perturbation [18]. We feel a similar model could be used in conjunction with our technique to choose the position of the foot and the timing, which are the two required inputs to our system.

3 Stepping Algorithm

The algorithm we employ has two particular foci. First, we control the stepping foot, both its path and its speed along that path. And second, we control the COM, again both position and velocity. We choose to control the COM in order to make the visible weight shift that corresponds to stepping actions in humans. We propose that these two factors alone encapsulate many of the physical attributes of a single step. While we also include motion examples of stepping to enrich the upper-body movement of the final motion, our hypothesis is that by moving the stepping foot and COM realistically we can generate believable stepping simply.

Our technique incorporates these two components into an animation transition using optimization. In particular, we employ a frame-by-frame optimizer which takes a frame from starting blend as input and produces a modified posture that enforces the desired stepping foot and COM trajectories. We expect the choice of timing and position for the final stepping foot be provided as input to our system. Along with the character's current motion, we extract an example motion from our database and use the sequence to compute the initial paths for the foot and COM.

We break the description of our algorithm into two phases, preprocessing and step generation. In the preprocessing stage, we determine the necessary inputs to the stepping algorithm, specifically:

1. Input (from the user) the final stepping foot position and step duration
2. Identify the closest example from the step database
3. Adjust the ending pose from example using IK to place foot
4. Extract the COM ending place from the (adjusted) end pose

Steps 3 and 4 are used solely to determine the final position of the COM based on the motion sequence selected. Alternatively, we can force the system to transition to a specific motion sequence. In this case, Steps 2 and 3 can be skipped. Motion generation follows a straightforward sequence:

1. Compute the stepping foot path, P_f, and speed profile, V_f
2. Compute the COM path, P_c and speed profile, V_c
3. Blend to example with support foot as root
4. Modify blend with optimizer to meet COM/foot trajectories

The starting blend from Step 3 is treated as the input to the optimizer. To keep the support foot from moving during transition, the blend is performed by treating the chosen support foot as the fixed root of the branching chain for the body. All other parts of the body move by smoothly interpolating the included joint angles. More details about each part of our algorithm are described in the following sections.

4 Stepping foot control

To define the stepping foot motion appropriate for the desired step/transition, we determine the foot's path and its speed along that path. We assume that the path and speed profile are related by the distance covered from start to finish. That is, the total path displacement must equal the integral of the function chosen for the speed. We also assume that distance covered is monotonically increasing along the path. We follow a similar set of definitions and assumptions for COM control.

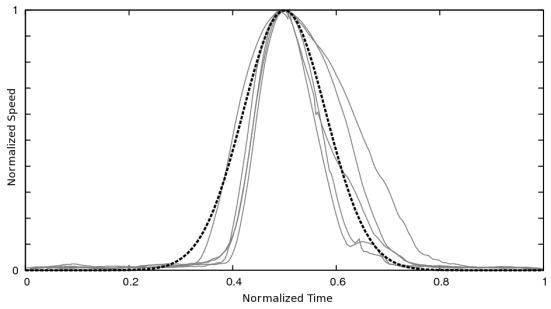

Fig. 1. Foot speed, V_f. The black, dashed curve is the idealized normal speed fit using a Gaussian centered at 0.5. The rest are normalized sample profiles taken from various examples in our step database.

We model stepping as if it is a point-to-point reach. Upon inspection of our database of examples, we found remarkable uniformity - nearly-linear, point-to-point paths for each stepping foot. There has been in-depth investigation performed on hand point-to-point movement for reaching tasks (see [19, 20]) and, in this body of work, it is commonly accepted that the hand traverses an approximately straight-line path with a bell-shaped speed profile. For our foot model, we adopt a similar estimate for the foot trajectory, P_f, by forcing the foot to traverse the line segment formed by its starting and ending position while using a normal Gaussian to serve as the bell-shaped speed curve. That is

$$V_f = ae^{\frac{-(x-0.5)^2}{2w^2}} \tag{1}$$

defines the speed along P_f. This idealized speed curve is plotted in comparison to several speed profiles taken from our database in Figure 1. When the recorded curves are normalized in both time and amplitude, they show remarkable similarity, *independent of the stepping direction, the length of the step, or the duration.* We adjusted the shape (width) of our normalized Gaussian shown by manually setting the constant, w, to be 0.08. This value is used for all our results using stepping examples. To align the speed profile with the path, we must control the area under the curve to be equal to the distance from the start to the end of the footstep. We automatically tune the Gaussian by scaling amplitude, a, after integrating the curve for the normalized amplitude shown in the figure. Note this integration need only be done once and can then be scaled by a to match the specific (known) distance covered in the to-be-synthesized motion.

5 Center of mass control

As with the foot, to control the COM we will define a simplified model which captures the features of the human examples. Again our path and speed are both idealized from observations about the stepping examples recorded in our database. The COM path follows a parabola-like trajectory starting and ending at known points and moving toward and away from the pivot foot. For P_c, we found empirically that a simple quadratic Bezier curve which uses the start, end, and pivot as control points reasonably maps out the path of the COM found in examples in our database. Comparisons appear in the results section.

For the speed, we observe in the recorded motions consistent trends in the trajectory of the COM velocity broken down into three phases. 1) Push off. In this phase, *before* the foot is lifted, the COM begins to accelerate at a fairly constant rate toward the support foot. 2) Free fall. The second stage has the foot off the ground and we see a trajectory that mimics an unactuated inverted pendulum with the center of mass accelerating uniformly away from the support foot (now out of static balance.) 3) Landing. The stepping foot reaches the ground and the motion induced in the second stage is dissipated with a slow changing acceleration back toward the (original) support foot. What we infer from these observations is that three stages with constant acceleration reasonably describe the observed velocity profiles. Note, these phenomena are described in the coordinate frame oriented toward the support foot.

An idealized inverted pendulum which pivots about the support foot models both our path and speed observations reasonably. Based on a pendulum model, our choice of path trajectory, P_c, is sensible since an idealized pendulum moves its body on a ballistic, quadratic path. To fit the velocity characteristics for the COM, we could approximate the effects of the stepping leg as applying a uniform 'push-off' and 'landing' force before and after the step. (The *minimum jerk* theory for reaching partially supports this proposition [20].) An idealized constant force would yield a constant acceleration for push-off and landing. Constant acceleration is also reasonable for the middle phase when the body feels only the effects of gravity. Thus, for the model of

our COM speed profile, V_c, we choose the piecewise linear function shown in Figure 2. We derive the terms of the velocity segments shown from known (or approximate) values for timing, $t_0 - t_3$, and the COM displacement, D, which is extracted from the Bezier curve, P_c.

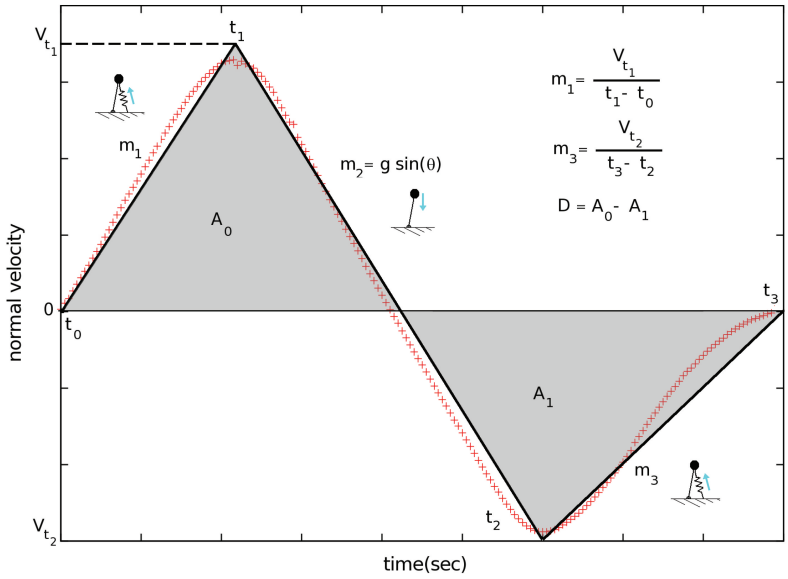

Fig. 2. COM speed, V_c. Ideal and actual speed profiles in the direction of the support foot. That is, only motion toward and away from the pivot foot contribute to the data plotted, plus signs are derived from a real example. The timing information, $t_0 - t_3$, which delimit the stages (push off, free fall, and landing, respectively) can be estimated from the motion example by detecting when the stepping foot leaves and reaches the ground. Based on the pendulum model, m_2 is set to $gsin(\theta)$ where θ is the lean angle between vertical plane and support leg and g is gravity. Areas, A_0 and A_1, link D, the displacement of the COM derived from P_c, to the slopes m_1 and m_3.

6 Synthesis for Stepping

To generate the starting blend given the stepping action parameters, we propose a simple, but effective interpolation synthesis technique. The problem here is to concatenate the motion the character is currently following with the stepping motion in the example. To be successful, the transition should not introduce any unwanted artifacts. The most straightforward solution is to align the transition-to motion globally to the character's current position and facing direction and to blend the root position and orientation as well as the joint angles. However, in general, this approach introduces undesirable foot sliding. Instead, we align the

support foot of the before and after motion and use this as the fixed root for the blend. The system then performs the blend by interpolating over the errors for the root orientation and the joint angles across the transition sequence. Note we do allow the support foot to rotate across the transition. This rotation is usually small if the facing direction of the two motions are closely aligned and acts to pivot the foot if there is a larger discrepancy. We show that such rotations appear natural looking in our results. In our implementation, our system interpolates by 'slerp'-ing quaternions, with a simple ease-in/ease-out (EIEO) time-based weighting across the transition.

Once we have the starting blend, we modify it to uphold the stepping foot and COM trajectories determined for the transition. We accomplish this goal on a frame-by-frame basis, first applying IK to place the stepping foot at the desired position and then using an optimization to reach the desired COM. The optimizer works by moving the pelvis position in the horizontal plane and using an IK sub-routine [21] to generate adjustments for each leg which enforce the proper foot placement. A similar approach is described here [22]. Further, our solver constrains the height of the pelvis to maintain a set maximum length for each leg, lowering the pelvis automatically to avoid stretching either leg beyond its limit. The beauty of this optimization routine is that it chooses only the placement of the pelvis in the horizontal plane, but shifts the entire body and subsequently controls the projected COM.

Our implementation uses Numerical Recipes' BFGS routine [23] which employs a quasi-Newton, gradient-based search to determine the pelvis location. The initial location is taken from the starting blend. The performance index is simply the error between the desired and current projected COM. The position of the desired COM is extracted from P_c by moving along the Bezier curve until the path displacement satisfies V_c. Likewise, the stepping foot location is set each frame to follow P_f while also satisfying the rate determined from V_f.

7 Implementation and Results

Our final implementation includes additional details that need to be described. Our stepping database was recorded by systematically creating a series of examples of normal steps taken in various directions. In total, we include twenty examples in our step database. We use our system in two modes: starting from a known stance and transitioning to a modification of one of the step examples; or by combining two existing clips, i.e. without using a step example. When generating a step animation that includes an example, we select the example in the database which is closest based on the (relative) desired step location. For results without using examples, we found that adjusting the width of Equation 1 is sometimes necessary to avoid abrupt movement of the stepping foot. The running time of our system is amply fast to be used at interactive rates.

Results. We show two types of results in the accompanying video to demonstrate the range of animations that are possible using the technique. First, we include examples which use our stepping database. We show an animation of a

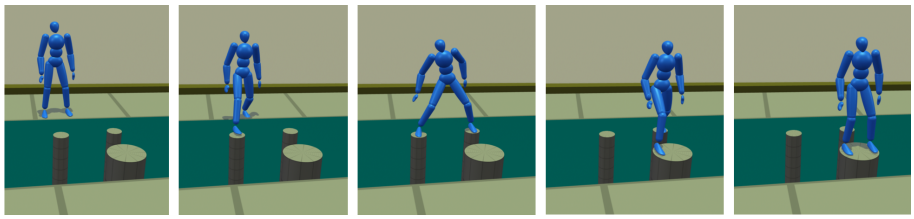

Fig. 3. Careful stepping. Navigating an extreme environment by precisely placing steps shows off a series of four steps completely synthesized by our system.

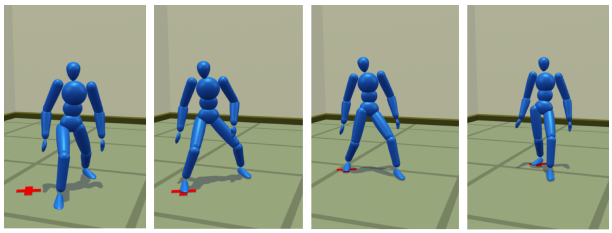

Fig. 4. Animation of a series of steps pivoting on the left foot. The red X marks the consecutive end locations of the steps synthesized, the X values were taken from the motion capture sequence shown in the lefthand plot of Figure 5.

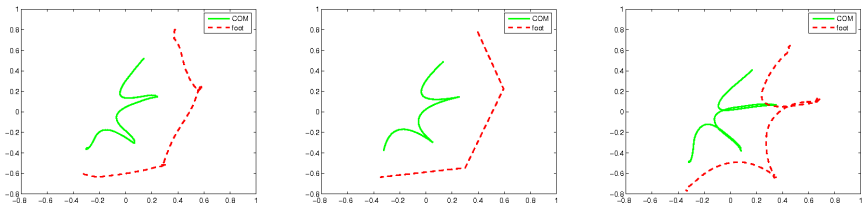

Fig. 5. Comparisons for stepping. Cartesian plots for the foot and COM paths shown in red and green lines respectively over three consecutive steps. On the left is a contiguous motion capture example held out of the database but used as target input for foot placement and timing. In the middle is motion resulting from our model (also shown in Figure 4). On the right is the starting blend, as described in the text.

series of steps using left and right feet alternately to create a careful navigation (see Figure 3.) We compare the quality of a second synthesized series with a continuous motion sequence of three steps held out of the database (see Figures 5 and 4). Next, we include two animation examples that are generated without the step database. The goal here is to breakdown the contributions of each component of our system and to show off the power of our technique for creating seamless transitions by stepping. In the video, we show a turning task which is derived from simply by rotating a contiguous motion of a "ready-stance" in martial arts. We contrast the optimized result with the starting blend. Next, we modify a series of fighting attacks to control the direction of one kick by changing the stepping motion preceeding the attack.

8 Discussion and conclusions

We have demonstrated the power of our simple method for generating controlled stepping movement. The underlying assumptions in our system are motivated by motor theorists and are supported by comparisons with motion capture examples of stepping. While the technique is very simple, used in combination with a stepping motion database we can generate rich motion that is comparable to unmodified motion captured stepping.

Our approach does include certain limitations. First, the system does not make any modifications to the upper body. While we know the upper body will respond to the movement of the lower body during stepping, we rely on the upper-body response embedded in the stepping example. When we remove this example, the motion of the upper body is computed solely from the interpolants and their blend. There is no guarantee that this will result in realistic motion. Likewise, the pivoting of the support foot is derived solely from the starting blend and we feel it is acceptable but not truly reflective of what we see in the motion database. The piecewise linear model for the velocity of the COM is likely too over-simplified to match human motion tightly, although we found it acceptable for our purposes. And finally, if user inputs a final stepping foot position where the distance is farther than the reaching limitation of single step for current subject, our system currently is unable to automatically generate multiple steps to achieve the final destination since this requires motion planning which is beyond the scope of this paper.

Motion generation from a system like the one proposed is useful for directly animating a character. However, we believe it is also potentially valuable for informing a control system when employed in the activation of a physical character or robot. We see this as a promising direction for future work. As is, the system we describe is easy to implement and fast to run, and so we hope it is adopted by game developers and animators who need to take steps toward stepping.

References

1. Kovar, L., Schreiner, J., Gleicher, M.: Footskate cleanup for motion capture editing. Symposium on Computer animation (2002) 97–104

2. Ikemoto, L., Arikan, O., Forsyth, D.: Knowing when to put your foot down. Interactive 3D graphics and games (2006) 49–53
3. Rose, C., Guenter, B., Bodenheimer, B., Cohen, M.F.: Efficient generation of motion transitions using spacetime constraints. In: ACM Siggraph. (1996) 147–154
4. Shin, H., Kovar, L., Gleicher, M.: Physical touch-up of human motions. Proceedings of the 11th Pacific Conference on Computer Graphics and Applications (2003) 194
5. Arikan, O., Forsyth, D.: Interactive motion generation from examples. ACM Siggraph (2002) 483–490
6. Kovar, L., Gleicher, M., Pighin, F.: Motion graphs. ACM Siggraph (2002) 473–482
7. Lee, J., Chai, J., Reitsma, P., Hodgins, J., Pollard, N.: Interactive control of avatars animated with human motion data. ACM Siggraph (2002) 491–500
8. Wang, J., Bodenheimer, B.: Computing the duration of motion transitions: an empirical approach. Symposium on Computer animation (2004) 335–344
9. Ikemoto, L., Arikan, O., Forsyth, D.: Quick transitions with cached multi-way blends. Interactive 3D graphics and games (2007) 145–151
10. Gleicher, M., Shin, H., Kovar, L., Jepsen, A.: Snap-together motion: assembling run-time animations. Interactive 3D graphics (2003) 181–188
11. Boulic, R., Mas, R., Thalmann, D.: Position control of the center of mass for articulated figures in multiple support. Eurographics Workshop on Animation and Simulation (1995) 130–143
12. Tak, S., Song, O., Ko, H.: Motion Balance Filtering. Computer Graphics Forum **19** (2000) 437–446
13. Yin, K., Pai, D., van de Panne, M.: Data-driven interactive balancing behaviors. Pacific Graphics (2005)
14. Kudoh, S., Komura, T., Ikeuchi, K.: Stepping motion for a humanlike character to maintain balance against large perturbations. Proc. of Intl Conf. on Robotics and Automation (2006)
15. van de Panne, M.: From footprints to animation. Computer Graphics Forum **16** (1997) 211–223
16. Popovic, M., Goswami, A., Herr, H.: Ground Reference Points in Legged Locomotion: Definitions, Biological Trajectories and Control Implications. The International Journal of Robotics Research **24** (2005) 1013
17. Kajita, S., Kanehiro, F., Kaneko, K., Yokoi, K., Hirukawa, H.: The 3D linear inverted pendulum mode: a simple modeling for a bipedwalking pattern generation. Intelligent Robots and Systems (2001)
18. Pratt, J., Carff, J., Drakunov, S., Goswami, A.: Capture Point: A Step toward Humanoid Push Recovery. Proceedings of the IEEE-RAS/RSJ International Conference on Humanoid Robots (2006)
19. Abend, W., Bizzi, E., Morasso, P.: Human arm trajectory formation. Brain **105** (1982) 331–348
20. Flash, T., Hogan, N.: The coordination of arm movements: an experimentally confirmed mathematical model. Journal of Neuroscience **5** (1985) 1688
21. Tolani, D., Goswami, A., Badler, N.I.: Real-time inverse kinematics techniques for anthropomorphic limbs. Graph. Models Image Process. **62** (2000) 353–388
22. Metoyer, R., Zordan, V.B., Hermens, B., Wu, C.C., Soriano, M.: Psychologically inspired anticipation and dynamic response for impacts to the head and upper body. IEEE Transactions on Visualization and Computer Graphics (TVCG) (2008)
23. Press, W., Teukolsky, S., Vetterling, W., Flannery, B.: Numerical Recipes in C. Cambridge University Press, New York (1994)