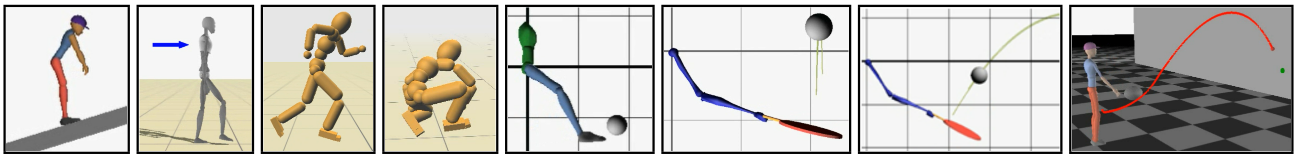
Learning Reduced-Order Feedback Policies for Motion Skills

Kai Ding[1] Libin Liu[2] Michiel van de Panne[1] KangKang Yin[3]

[1]University of British Columbia [2]Microsoft Research Asia [3]National University of Singapore



(a) Balancing (b) Walking (c) Running (d) Rolling (e) Kicking (f) Juggling (g) Volleying (h) Ball Hitting

Figure 1: *Given a set of sensory observations and a set of control actions for a given motion skill, our method learns low-dimensional linear feedback strategies that enable robust motions.*

Abstract

We introduce a method for learning low-dimensional linear feedback strategies for the control of physics-based animated characters around a given reference trajectory. This allows for learned low-dimensional state abstractions and action abstractions, thereby reducing the need to rely on manually designed abstractions such as the center-of-mass state or foot-placement actions. Once learned, the compact feedback structure allow simulated characters to respond to changes in the environment and changes in goals. The approach is based on policy search in the space of reduced-order linear output feedback matrices. We show that these can be used to replace or further reduce manually-designed state and action abstractions. The approach is sufficiently general to allow for the development of unconventional feedback loops, such as feedback based on ground reaction forces. Results are demonstrated for a mix of 2D and 3D systems, including tilting-platform balancing, walking, running, rolling, targeted kicks, and several types of ball-hitting tasks.

CR Categories: I.3.7 [Computer Graphics]: Three-Dimensional Graphics and Realism—Animation;

Keywords: human simulation, control, character animation

1 Introduction

Human motions have a passive component that is dictated by physics alone, and an active component that is dictated by the control of the muscles. Although the passive component is well understood, the modeling of the control remains an obstacle for physics-based models of human motion. Considerable effort has been devoted to developing control strategies for specific motions such as diving, vaulting, running, walking, and in-place balancing. Not surprisingly, the carefully-designed use of feedback is a crucial element in the development of a good controller.

A common feature of many control systems developed for character animation is the introduction of *abstracted* or *simplified* models of the simulated character. These serve to summarize the most relevant state information, such as the center of mass (COM) position and velocity, total angular momentum, and the present-and-future locations of the feet. They can also serve to define simplified or abstract actions, such as the use of inverse kinematics to guide foot placement or the selection of the swing hip angle as the primary target for balance feedback. These abstractions are the product of human insight, and they significantly simplify the design of good control strategies.

In this paper we investigate the possibility of learning reduced-order feedback abstractions and their related feedback loops instead of having to manually design them. A solution to this problem would enable the rapid development of a much larger array of skills for simulated characters. Our proposed solution takes the form of linear feedback policies, $\mathcal{F} : \mathbb{R}^n \rightarrow \mathbb{R}^m$ that define a mapping from observed changes in sensory state, δs, to changes in control actions, δa. This creates a feedback structure *around nominal reference trajectories for the sensory state and actions (given apriori)*. The feedback is applied at every control time step, in the case of continuous motions, or at the start of a motion in the case of discrete motions. A reduced-order linear feedback policy is then defined via an intermediate projection into a space with reduced dimensionality: $\mathcal{F}_r : \mathbb{R}^n \rightarrow \mathbb{R}^r \rightarrow \mathbb{R}^m$. Achieving a reduction in dimensionality requires r to be smaller than both the input and output dimensions, i.e., $r < m, n$. The parameters of the control policies are then given by the matrices that perform the mappings outlined above. A feedback policy of full or reduced dimension is learned by optimizing these matrices according to a cost function that evaluates any given policy using a policy rollout, i.e., a finite-duration simulation of the policy in action.

Having established a setting for the design of reduced-order policies, there remain many questions to be answered. What is the minimal value of r that is needed to support the robust control of a given motion task? What types of sensory variables can be usefully exploited in the feedback loops? How should the cost functions be designed for a given motion task? Can the reduced-order framework 'discover' simple feedback laws such as those demonstrated in the past for the control of locomotion gaits [Raibert and Hodgins 1991; Yin et al. 2007]?

We evaluate the given control framework using a set of four planar motion skills (in-place balancing, targeted kicking, juggling, volleying) and four 3D motion skills (walking, running, rolling, and ball hitting). A diverse set of sensory inputs is used across these scenarios: full state vectors, ground reaction forces, target locations, delayed inputs, and noisy inputs. The results show that low-dimensional linear feedback policies are effective for many motion skills and that non-traditional sensory observations can be surprisingly useful as inputs in constructing feedback loops. The method helps enable the black-box design of feedback loops, where an end user can simply provide a nominal motion requiring feedback, enumerate sets of possibly-relevant inputs and outputs for the design of the feedback loop, and a task-appropriate cost function.

2 Related Work

The utility of reduced-order models is well known for the efficient simulation of passive dynamical systems. In the context of computer animation, reduced order models have been developed for the simulation of deformable bodies, e.g., [James and Pai 2002; Barbič et al. 2009] and fluids [Treuille et al. 2006]. The model reduction is computed using modal analysis or analysis of the data arising from the full-order model. The use of modal analysis has been proposed as a tool for character animation [Kry et al. 2009], although it leaves balance feedback as an important open problem. A more recent approach [Jain and Liu 2011] builds on these ideas and addresses issues of balance and motion planning. Walking, squatting, and chin-up motions are synthesized using 10-20 modes. Our work shares the goal of finding reduced representations, but without relying on reduced-order models for the dynamics, and with feedback paths having as few as 1-3 dimensions.

Our work is motivated in part by the hypothesis that humans exploit muscle synergies [Berniker et al. 2009] to reduce effective dimensionality. In this vein, manually-designed state abstractions are commonly used in the design of walking, running, and balancing controllers for physics-based characters. Many methods use the center of mass position and velocity as key variables for adapting foot placements [Raibert and Hodgins 1991; Yin et al. 2007; Wang et al. 2009; Wang et al. 2010; Lee et al. 2010; Kwon and Hodgins 2010]. Inverted pendulum models are based on a similar choice of state abstraction [Tsai et al. 2010; Coros et al. 2010; Mordatch et al. 2010]. The trunk orientation can also be used as a key state variable [Laszlo et al. 1996]. A center-of-mass model can be augmented with separate components that model angular momentum and global angular position [Ye and Liu 2010]. Modeling the net angular momentum is also demonstrated to be a key component for skilled in-place balancing behaviors [Macchietto et al. 2009]. A three-link articulated model can be used as a simplified model of the torso and legs for the control of locomotion [da Silva et al. 2008]. Other methods also rely on a small set of motion features as a key aspect of computing the applied controls [de Lasa et al. 2010; Muico et al. 2009] or as part of a distance metric [Sok et al. 2007]. Existing motion data for a movement can be used to constructe a PCA-based subspace within which motion trajectories can be optimized [Safonova et al. 2004]. Forward dynamics simulations that track an example motion can be constructed using a stochastic search process and significant offline compute time, as demonstrated in [Liu et al. 2010]. However, this produces only an open-loop solution. Most recently, compact non-linear control solutions are learned for challenging motions such as bicycle stunts [Tan et al. 2014].

Our work is related to the design of linear output feedback in classical control theory [Lewis and Syrmos 1995]. Linearized trajectory tracking has been demonstrated for locomotion, e.g., [da Silva et al. 2008]. Output feedback is usually designed using model-based approaches and a linearized system so that the static output feedback matrix can be designed by convex optimization or solving a system of linear matrix inequalities [Scherer et al. 1997]. Because the Ricatti equations are not well suited to the development of reduced-order controllers [de Oliveira and Geromel 1997], a number of alternative approaches have been developed for reduced-order design. These include solving minimum rank problems [David and De Moor 1994] and reduced basis approaches [Burns and King 1998]. The reduced-order design problem is known to be non-convex and non-smooth even for linear systems subject to practical design criteria, therefore leading to the use of numerical optimization methods of the output feedback matrix [Burke et al. 2003]. Model reduction for nonlinear control systems has been proposed, e.g., [Lall et al. 2002], by utilizing PCA on the state space in conjunction with balanced truncation. However, the approach requires reference data for the closed-loop system, requires advance knowledge of a meaningful 2-norm on the state, and is demonstrated on a limited and smoothly-nonlinear example: a 5-link planar chain with angular springs, one active joint, and with no collisions.

The work of [Liu et al. 2012; Liu et al. 2013] makes use of the basic reduced order control (ROC) idea that is elaborated in this paper. However, the contributions of [Liu et al. 2012] lie elsewhere, namely with the control parameterization and transitions. With regard to ROC, it summarizes the idea and then refers the reader to an unrefereed technical report for all further details. The current paper provides the following contributions beyond what is described in [Liu et al. 2012]: (a) the ROC ideas are positioned in relation to the relevant prior art; (b) the impact of the reduction order is explored ($r = 3$ is used for all results in [Liu et al. 2012]); (c) the robustness to force and terrain perturbations are evaluated and compared to manually designed ROC alternatives for walking, such as SIMBICON; (d) experiments are carried out on a richer variety of problems, including in-place balance, repeated rolls, juggling, volleying, hitting, and kicking; (e) we more thoroughly evaluate, interpret, and discuss the ROC results; (f) we investigate adding regularization to encourage further feedback matrix sparsity; (g) we compare the effect of local vs global optimization methods.

3 Feedback Structure

The feedback policies we explore apply changes in control actions as a linear function of changes in sensory observations:

$$\delta a = M_F \cdot \delta s \tag{1}$$

where M_F is an $m \times n$ feedback matrix (subscripted F for full-order), $\delta a = a - \hat{a}$, and $\delta s = s - \hat{s}$. A nominal open-loop reference policy is assumed to be available, consisting of reference control actions, $\hat{a}$, and reference sensory observations, $\hat{s}$. The feedback can be applied to continuous motions, such as in-place balancing, walking, running, and rolling, or to discrete motion tasks, such as kicking, juggling, and volleying. In the case of continuous motions, the reference controls and sensed observations take the form of time-indexed trajectories, i.e., $\hat{a}(t)$ and $\hat{s}(t)$. For discrete motion tasks, they take the form of fixed nominal values. The matrix M_F provides *static output feedback*, where *static* means that the feedback gains do not change over time, and *output feedback* indicates the direct use of measurements in computing feedback, in contrast to conventional *state feedback* methods.

We explore the use of linear feedback strategies in a broad setting. Motion controls are represented using target angle trajectories that are tracked via joint-based proportional derivative (PD) controllers. Control actions can then be diverse in nature, including changes to the target angles, changes to the PD-gains, and changes to the timing of the control points that define spline-based trajectories for our discrete motion skills. The sensed observations that drive the feedback can be equally diverse, and can include state variables, ground reaction forces, goal locations, center of mass locations, or other types of measurements.

Reduced order linear feedback: The full-order linear feedback policy is parameterized by the $m \cdot n$ elements of M_F. In order to define more compact policies, we can factor M_F into two components: (i) a $r \times n$ sensory projection matrix M_{sp} that projects high-dimensional sensory observations to a reduced-order state space;

and (ii) a $m \times r$ action projection matrix M_{ap} that maps the reduced-order state back to the full action space to produce the feedback compensation. The feedback policy then becomes:

$$\delta a = M_{ap} \cdot M_{sp} \cdot \delta s \tag{2}$$

The reduced order feedback policy has $r(m + n)$ parameters. Choosing $r < mn/(m+n)$ guarantees a policy with fewer parameters than the full policy. This further implies $r < \min(m, n)$. The intermediate reduced-order space of dimension r can be thought of as a latent space defined by a small number of abstract *composite variables* [Slotine et al. 1991] that are particularly useful for providing feedback. We use **(n:r:m)** to describe a linear feedback policy with n-dimensional sensory state, r-dimensional reduced-order space, and m-dimensional actions. Full-order feedback policies will be denoted by **(n:F:m)**.

Additional sparsity can be enforced by encouraging rows and columns to be zero. This can implicitly perform feature selection among actions (rows of M_{ap} set to zero) and sensory observations (columns of M_{sp} set to zero). This is implemented as part of the policy optimization process, as we discuss next.

4 Policy Optimization

We apply policy search using repeated rollouts in order to optimize the linear feedback sructure, M, which consists of either the full matrix M_F or its reduced-order factored form, i.e., $M_{ap} \cdot M_{sp}$. Given a desired motion task, a cost function is defined. These share a common structure:

$$cost(M) = w \cdot [S(M), E(M), U(M), R(M)] \tag{3}$$

The function score is a weighted sum of four terms: $S(M)$ rewards structures that make the motion as robust as possible; $E(M)$ measures how well the resulting motion meets the environment constraints; $U(M)$ measures how well the motion satisfies user specifications; and $R(M)$ is an optional regularization term used to encourage the sparsity of M and therefore implicitly feature selection on the sensing and control variables. We use L_1 regularization terms for the norms of column vectors in the sensory projection matrix M_{sp} as well as L_1 norms of row vectors in the action projection matrix M_{ap}. This yields:

$$R(M) = w_0 \sum_i \sum_j \left\| M_{sp_{ij}} \right\|_1 + w_1 \sum_i \sum_j \left\| M_{ap_{ij}} \right\|_1 \tag{4}$$

We use a stochastic global optimization technique, Covariance Matrix Adaption (CMA) [Hansen 2006] to optimize the feedback structure. The optimization begins from an initial guess consisting of zero entries. For some control tasks, the optimization is challenging due to the complexity of the dynamical system and the fact that good solutions may only be found in a highly restricted region of the parameter space. For these tasks we therefore break the optimization into multiple stages, each with increasing difficulty, and each using the solution of the previous stage as a starting point.

5 Motion Skills

We apply our method of learning feedback policies to the set of eight motion skills shown in Figure 1. The motions and an understanding of the feedback capabilities are best seen in the accompanying video and the supplementary material. For physics simulation, the 3D examples use Open Dynamics Engine [ODE] while the planar simulations use Box2D [Box2D]. The results are computed using a single-threaded implementation on a modern PC, except for the running and rolling optimization, which use an 18-core cluster. We test the full-order linear feedback policies as well as several reduced orders. Table 1 provides a summary of the policies tested, their structure, the offline computation times, and the final optimized value of the cost function.

In the remainder of this section, we detail the sensory variables, control actions, cost functions, and staged-learning (if any) for each skill. For readability, we immediately follow this with the related results. We first describe the four continuous skills (balancing, walking, running, rolling), and then move on to the discrete skills (kicking, juggling, volleying, hitting). The feedback for discrete actions is computed once at the start of the motion. This is different from the continuous action case, where feedback is applied at each simulation time step with respect to given reference trajectories for states and actions.

5.1 Balancing

Given a free-standing character, the goal is to provide the character with the ability to maintain balance on a tilting platform, as shown in Figure 2. The reference control, θ_0, consists of the four fixed target joint angles of a balanced pose on a level platform. The feedback policy computes changes to these target angles over time so that the character is able to adapt to the tilting platform. The sensory observations are defined as the net change in the slope of the platform $\Delta\alpha = \alpha - \alpha_0$ and the changes in the ground reaction forces $\Delta F_c = F_c - F_{c_0}$. Here, $\alpha_0 = 0^\circ$ and F_{c_0} is given by the pair of ground reaction forces seen at the heel and toe when the character stands on a level platform. The feedback can then be formulated as:

$$[\delta\theta]_{3\times1} = M \cdot \begin{bmatrix} \Delta\alpha_{1\times1} \\ \Delta F_{c_{4\times1}} \end{bmatrix}_{5\times1} \tag{5}$$

In the optimization, we only consider the robustness term $S(M)$. We reward policies that enable the longest duration of sustained balance, $t_{balance}$. Additionally, we consider that the character is most stable when the feet are in full contact with the platform. This is measured by t_{stable}, the overall duration of continuous, stable contact. Equation 6 shows the cost function. Each policy rollout is 30 s long and is driven by a predefined spline trajectory that controls the platform angle over time, $\alpha(t)$.

$$cost(M) = -\log(t_{balance} + 0.3 \cdot t_{stable}) \tag{6}$$

Results: The planar balancing character is placed near the pivot point of a tilting platform, as seen in Figure 2. A Catmull-Rom spline curve is used to model the tilt angle as a function of time. These curves are illustrated above the balancing characters. The vertical red bar marks the current point in time for the given image. One of these curves is used as a training scenario, i.e., to optimize the feedback policy, while four are used for testing. Full-order, first-order, and second-order solutions are learned. A number of remarks can be made about the results. The entire body is used to maintain balance, although a naive solution might only use the ankle joint. The arms and trunk naturally counter-rotate with respect to each other in order to help maintain near-zero net angular momentum. In some test scenarios the character can be propelled into the air and still successfully lands and recovers. The first order solution is slightly less capable than the second-order and full-order solutions according to the final values of the cost function. We compare the learned solutions to a hand-tuned solution that only uses the ankle, and the hand-tuned solution is found to be markedly inferior to the learned policies. In order to test the ability of the system to cope with delayed sensory information, we experiment with adding 150 ms of delay to the sensed state during both training and

Feedback	Params	Evals×10^3	Time (hrs)	Cost
Balance				
5:F:4	20	8.7	0.11	-3.65
5:2:4	18	5	0.43	-3.64
5:1:4	9	0.5	0.02	-3.61
5:F:4∗	20	9.7	0.09	-3.64
Walking: full state (FSA)				
108:3:19	381	56	90	733
108:2:19	254	95	397	741
108:1:19	127	96	399	711
Walking: center of pressure (COP)				
6:F:19	114	57	118	792
6:3:19	75	97	431	714
6:2:19	50	95	429	705
6:1:19	25	78	272	757
Walking: ground reaction forces (GRF)				
6:F:19	114	60	145	732
6:3:19	75	88	209	912
6:2:19	50	91	380	1010
6:1:19	25	94	333	1675
Walking: SIMBICON				
2:1:1	4	1	11	715
Running: reduced sensory-state and actions (RSA)				
12:F:9	108	18	0.37†	3.45
12:3:9	63	18	0.30†	3.39
12:2:9	42	18	0.37†	3.51
12:1:9	21	-	-	fail
Running: full state and action (FSA)				
88:1:39	127	-	-	fail
Rolling				
88:1:39	127	11.5	2.04†	4.45
Kicking				
2:F:26	52	20	69.8	32.7
2:1:26	28	26	164.2	110.7
Juggling				
21:F:24	504	23	3.4	0.0173
21:3:24	135	85	25.3	0.0021
21:2:24	90	55	14.9	0.0022
21:1:24	45	66	18.0	0.0196
Volleying				
21:F:24	504	13	9.14	0.089
21:3:24	135	-	-	fail
21:2:24	90	43	47.7	0.125
21:1:24	45	-	-	fail
Ball Hitting				
2:F:38	76	14.9	39.9	0.005
2:1:28	30	6.4	37.5	2.55

† Computed using 18 cores.
∗ Sensory state delayed by $150ms$.

Table 1: *Results.*

Order	FS	COP	GRF	HM
full	n/a	203	75	failed
1	94	85	107	137
2	167	184	80	90
3	73	193	63	120

Table 2: *Maximum recoverable forces for walking, in N.*

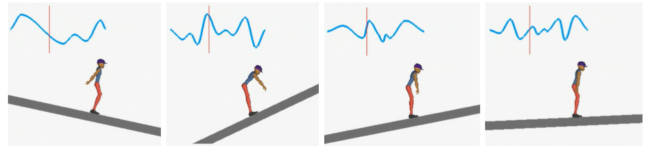


Figure 2: *Balancing*

testing. With full-order feedback, this produces only a slight decrease in performance. The balancing example also demonstrates the success of a non-traditional form of feedback; the sensed state consists of the ground tilt angle and the ground reaction forces, but it otherwise has no information about the state of the character, as would be the case for a more classical feedback structure. However, the joints remain aware of their own local positions and velocities through their individual PD controllers.

5.2 Walking

For this motion skill, the goal is to learn a feedback policy that provides robust walking. We use an implementation of the SIMBICON balance strategy [Yin et al. 2007] to provide the reference trajectories. This employs a hand-tuned balance feedback law on the swing hip and stance ankle, depending on the horizontal distance from the stance ankle to character's center of mass (COM) and the velocity of COM. Our goal is to replace the SIMBICON feedback law with a learned policy given a collection of possibly relevant sensory data and control actions. To simplify the problem, we use a fixed SIMBICON control law to maintain balance in the coronal plane, i.e., lateral balance, and seek to learn a reduced-order sagittal plane feedback policy. We control the target pose θ for all joints of the character at each simulation time step.

We experiment with four sets of sensory inputs. Equation 7 defines the FSA feedback policy which uses the full state (FS) description for the 3D character. Equation 8 shows the GRF policy, which uses the ground reaction forces (GRF) on both feet of the character as the sensory inputs. Equation 9 defines the COP policy, which uses the distance ΔD between center of pressure (COP) and COM of the feet as the sensory data where $\Delta D = p_{cop} - p_{com}$. Lastly, equation 10 describes a high-dimensional mixed (HM) policy that uses the COP, GRF, 15 randomly selected features from the state vector, and 3 Gaussian noise terms to test for invariance with respect to additional superfluous inputs.

$$[\delta\theta]_{19\times1} = M^{\text{FS}} \cdot [\Delta s_{full}]_{108\times1} \quad (7)$$

$$[\delta\theta]_{19\times1} = M^{\text{GRF}} \cdot \begin{bmatrix} \Delta F_{C_{\text{swing}_{3\times1}}} \\ \Delta F_{C_{\text{stance}_{3\times1}}} \end{bmatrix}_{6\times1} \quad (8)$$

$$[\delta\theta]_{19\times1} = M^{\text{COP}} \cdot \begin{bmatrix} \Delta D_{\text{swing}_{3\times1}} \\ \Delta D_{\text{stance}_{3\times1}} \end{bmatrix}_{6\times1} \quad (9)$$

$$[\delta\theta]_{19\times1} = M^{\text{HM}} \cdot \begin{bmatrix} \Delta D_{\text{COP}_{6\times1}} \\ \Delta F_{\text{GRF}_{6\times1}} \\ \Delta s_{15\times1} \\ \Delta N_{\sigma_{3\times1}} \end{bmatrix}_{30\times1} \quad (10)$$

In the optimization, $S(M)$ rewards feedback policies that enables a sustained walk of duration $t_{balance}$. $U(M)$ is used to constrain the resulting motion to have a mean step length, l, a mean velocity per step, v, and a mean reference state, s, when the swing foot strikes the ground. We also encourage the character to walk with minimal energy, as measured by $U_2(M)$. $R(M)$ is used to perform feature

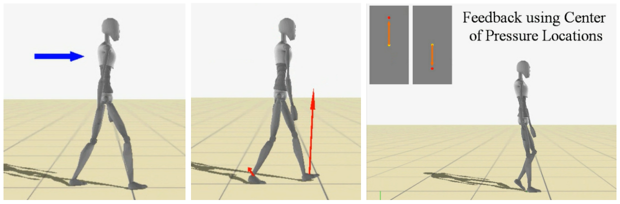


Figure 3: *Walking*

selection when using the reduced-order form of feedback structure. Equation 11 shows the cost function for the optimization and the weights are set to be: $w = [2000, 1000, 1000, 0.5, 0.0001, 200]$. T represents the total simulation time of a rollout. We adopt a three-stage optimization strategy for this example. In stages one and two, we optimize using 5 s and 50 s rollouts, respectively. In stage three, we apply gradually increasing forces during the scenario so that the feedback is refined so as to enhance the robustness of the walking motion.

$$\begin{aligned}
&cost(M) = S(M) + U_1(M) + U_2(M) + R(M)\\
&S(M) = w_1 \cdot (T - t_{balance})\\
&U_1(M) = \tfrac{\sum_{i=1}^{N}(w_2(l_i-l_0)^2 + w_3(v_i-v_0)^2 + w_4(s_i-s_0)^2)}{N}\\
&U_2(M) = w_5 \cdot \bar{\tau}^2\\
&R(M) = w_6 \cdot (\textstyle\sum_i \sum_j \left\|M_{sp_{ij}}\right\|_1 + \sum_i \sum_j \left\|M_{ap_{ij}}\right\|_1)
\end{aligned} \tag{11}$$

Results: The walking results are summarized in Table 1. The results show that all types of sensory-state feedback can be successful. Example results are shown in Figure 3. In all cases, one-dimensional reduced order feedback proves to be sufficient to yield robust feedback for the sagittal component of 3D walking. We do not develop a full-order policy for FSA walking because of the large number of parameters it would have ($108 \times 19 = 2052$). The outright failure of the HM policy to achieve stable locomotion using the full matrix clearly illustrates the benefit of working directly with the reduced-order feedback space.

For the FSA and COP cases, all orders yield roughly equivalent performance in terms of the final optimized cost function. While a higher-order solution should at least match the performance of a lower-order solution, this is not strictly the case for FS, COP, and GRF. We attribute this to the optimizations not finding the global minimum in the high-D parameter spaces.

We further evaluate the feedback policies according to the maximum recoverable force (MRF) for an applied external force of 0.4 s duration applied at the beginning of the motion. The results are shown in Table 2. Many of the feedback policies improve upon an optimized version of SIMBICON, which has a MRF of 80 N in our experiments. The second, third, and full-order COP feedback policies are found to be the most robust.

We next examine whether the reduced-order dimensions have an intuitive interepretation. In looking at the L_1 norms of the columns of M_{sp}, the two most important features are found to be: (1) the angular velocity of the swing knee; and (2) the angular velocity of the swing hip. These features are highly correlated with the step length in the sagittal plane and thus are informative for sagittal balance. Thus while the resulting solutions are compact and robust, they do not directly reproduce the choice of state variables seen in controllers such as SIMBICON. In future work, we wish to verify the effectiveness of reduced-order control for both sagittal and coronal plane, i.e., not relying on SIMBICON for the coronal plane balance feedback as we currently do.

5.3 Running

A feedback policy for 3D running is developed around a motion capture clip, for which we first need to compute a reference set of controls. Figure 4 shows our simulated running character. We begin with one full gait cycle of motion capture of a fast run, which is made symmetric and then concatenated six times to produce a reference motion for sampling-based reconstruction of the controls [Liu et al. 2010]. The resulting open-loop control strategy is driven by PD servos tracking a sequence of target poses $\{\boldsymbol{p}_i\}, i \in \{1, \ldots, n\}$. A pose $\boldsymbol{p} = \{\boldsymbol{q_j}\}, j \in \{1, ..., m\}$, where $\boldsymbol{q}_j$ is the quaternion representing the orientation of the jth joint in its parent frame, and m is the number of joints. Each $\boldsymbol{p}_i$ has a corresponding tracking duration Δt_i. The PD servo tracks a linear interpolation of $\boldsymbol{p}_i$ and $\boldsymbol{p}_{i+1}$ for Δt_i seconds and then goes on to interpolate $\boldsymbol{p_{i+1}}$ and $\boldsymbol{p}_{i+2}$. Phase resetting is applied upon foot contact, as in SIMBICON [Yin et al. 2007]. In our experiments, $\Delta t_i = 0.0625s$, and one running step has exactly four target poses. The simulated motion of the above controller is not necessarily cyclic or symmetric even though the reference trajectory is. We create a symmetric set of controls for one run cycle by concatenating the controls for one step (half cycle) with its symmetrically mirrored version. This is used as an initial reference control. We denote the simulated running motion as $\hat{s}$.

Optimization is carried out using a series of sequential stages. Rollouts of 2, 4, 16, and 100 locomotion cycles are used during stages 0,1,2, and 3, respectively. We move CMA to the next stage when the iteration count exceeds 1000 or the value of objective function is smaller than a chosen threshold. The components of the cost function are given by:

$$\begin{aligned}
&cost(M) = S(M) + U(M)\\
&S(M) = w_t(N_d T_c - t_{balance})\\
&U(M) = w_s E_s + w_p E_p + w_\tau E_\tau
\end{aligned} \tag{12}$$

where N_d is the desired number of running cycles, T_c is the length of one reference running cycle. The simulation is terminated after a fall or when the desired number of cycles is reached. N_s is the actual number of cycles of the simulation, whose termination time is denoted as $t_{balance}$. We use $(w_t, w_s, w_p, w_\tau) = (200, 50, 10, 0.001)$ for stage zero and $w_s = 10$ for the remaining stages.

The definitions of E_s, E_p, and E_τ are given in Equations 12-14. E_s measures the symmetry of the simulated motion where $\boldsymbol{p}_i$ denotes the end pose of the ith step of the simulated run, and $\boldsymbol{r} = \{\boldsymbol{q}_0, \dot{\boldsymbol{q}}_0, h_0, \boldsymbol{v}_0\}$ denotes the root state. Here, $\boldsymbol{q}_0$, $\dot{\boldsymbol{q}}_0$, h_0 and $\boldsymbol{v}_0$ are the orientation, the angular velocity, the height, and the linear velocity of the root, all as measured in the frame defined by character's root link facing direction. The functions $d_p(\boldsymbol{p_i}, \boldsymbol{p_j})$ and $d_r(\boldsymbol{r_i}, \boldsymbol{r_j})$ measure the pose and root difference between adjacent steps respectively. The overbar notation denotes a symmetric mirroring operation. E_p defines a pose energy, where $\boldsymbol{s}$ and $\tilde{\boldsymbol{s}}$ are the simulated and the reference motions. E_τ defines the control energy, where τ_j is the joint torque of joint j.

$$E_s = \frac{1}{N_s}\sum_{i=1}^{N_s}[d_p(\overline{\boldsymbol{p}_{i-1}}, \boldsymbol{p}_i) + d_r(\overline{\boldsymbol{r}_{i-1}}, \boldsymbol{r}_i)] \tag{13}$$

$$E_p = \frac{1}{T}\int d_p(\boldsymbol{s}, \tilde{\boldsymbol{s}})dt \tag{14}$$

$$E_\tau = \frac{1}{T}\int \sum_{j=1}^{m} ||\tau_j|| dt \tag{15}$$

After stage zero of the optimization, we update the reference motion and reference actions to be those of the current simulation. This additional bootstrapping step helps ensure that the reference motion is highly consistent with a dynamically feasible running cycle.

During stage 4, the character runs for 100 locomotion cycles while applying external forces on the torso for 0.1 s every 5 running cycles. These forces are either $125N$ or $250N$ in magnitude and along one of the axial or diagonal directions in the plane, applied at the phase where the left foot contacts the ground. The simulation is terminated whenever the character falls. This stage is important for improving the robustness of the feedback strategy.

We test two sets of sensory input and control parameters. The first uses full-body state and action (FSA) control. A straightforward choice for the sensory input s is the full-body state $s_f = \{h_0, \boldsymbol{v_0}, \boldsymbol{q_0}, \dot{\boldsymbol{q}}_0, \boldsymbol{q_j}, \dot{\boldsymbol{q}}_j\}, j \in \{1, \ldots, m\}$ of 88 dimensions. h_0 is the height of the root; $\boldsymbol{v_0}$, $\boldsymbol{q_0}$ and $\dot{\boldsymbol{q}}_0$ are the linear velocity, the orientation and the angular velocity of the root. These quantities are in the facing coordinate frame of the root. $\boldsymbol{q_j}$ and $\dot{\boldsymbol{q}}_j$ are the rotation and angular velocity of joint j in the parent body's coordinate frame. A straightforward choice of the control parameters a is the PD target pose $a = \{\boldsymbol{q_j}\}, j \in \{1, \ldots, m\}$ of 39 dimensions.

Second, we experiment with a manually-chosen reduced set of state and action (RSA) variables. We manually select several key sensory properties and action parameters. A 12-dimensional $s_r = \{\boldsymbol{q_0}, \boldsymbol{c}, \dot{\boldsymbol{c}}, \boldsymbol{d}\}$, where $\boldsymbol{q_0}$ is the root orientation; $\boldsymbol{c}$ and $\dot{\boldsymbol{c}}$ are the COM position and linear velocity; and $\boldsymbol{d}$ is the vector pointing from the COM to the stance foot. These properties are in the facing frame of the root. We choose the hips and the waist as our key joints for a 9-dimensional action vector, $a = \{\boldsymbol{q_{swhip}}, \boldsymbol{q_{sthip}}, \boldsymbol{q_{waist}}\}$. All these rotations are defined relative to their parent frame. To achieve more coordinated spinal postures, $\delta \boldsymbol{q_{waist}}$ is also applied to the chest joint. The given δs and δa is used during right stance and mirrored during left stance.

Equations 16 and 17 summarize the FSA and RSA feedback policies, respectively.

$$[\delta \boldsymbol{a}]_{39\times 1} = M \cdot [\Delta s_f]_{88\times 1} \quad (16)$$
$$[\delta \boldsymbol{a}]_{9\times 1} = M \cdot [\Delta s_r]_{12\times 1} \quad (17)$$

Results: The control policy for running aims to achieve full 3D control with no prior information regarding possible strategies for separation of the task into sagittal and lateral control. Attempts to design a first-order policy using the 88-dimesional full state vector and the 39-dimensional action space failed to produce a stable control policy. Because of the dimensionality of the problem, we did not test higher order policies that used the full state-and-action (FSA) space because a second order FSA solution would already have 254 parameters. A reduced set of sensory state and action (RSA) variables, i.e., 12:r:9, yields successful policies for full, second, and third-order policies for robust 3D running. Robustness to terrain variations is also achieved by including example variations in the final optimization stage. Figure 4 shows several results. While we found it possible to produce some successful first-order (12:1:9) policies for flat terrain, the resulting systems were fragile with respect to perturbations and thus we treat this as a failure case. The fragile nature of this solution is expected, given the need for both sagittal and coronal plane balance feedback.

The robustness of the RSA policies with respect to external perturbations are tested by applying external forces for 0.2 s along a set of 8 different directions. The simulated character runs for 30 steps and the forces are applied at the beginning of the 10th step, which is a right stance step. The maximum perturbation force that can be sustained in any given direction typically ranges from 140-180 N,

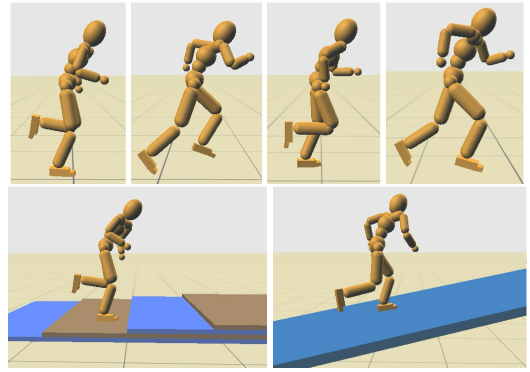


Figure 4: *Running level terrain, steps, and slopes.*

and can go as high as 500 N for a backwards push that slows the character. No significant differences were noted between the second, third, and full-order policies for the RSA feedback structure, although they still provide the benefit of being significantly more compact.

5.4 Rolling

We learn a reduced-order feedback loop for a parkour-style front roll obtained from motion capture. Figure 5 shows example motions. We repeat a motion-captured parkour roll, $\hat{s}$, four times to obtain a cyclic kinematic reference motion. We again use sampling-based control [Liu et al. 2010] to develop a set of open-loop controls that imitate the reference motion in a forward dynamics simulation. We use Δt_i=0.1 s The phase resets upon right foot contact and right elbow contact. We select a consecutive set of target poses from the segment of the reconstructed motion that most closely tracks its reference cycle. This gives us an open-loop rolling control $\hat{a}$ that is able to roll the character for one cycle.

The feedback policy uses the full state and actions as for the FSA running, i.e., Equation 16. It also shares the objective function of the running controller. The policy optimization follows a sequences of stages that are analogous to those used for the optimization of the running controller. Stage three uses 50 cycles of locomotion. Because rolling is more stable than running, we are able to get results that can roll the character forever without requiring an optimization stage that adds perturbations. We nevertheless proceed with a final optimization stage with perturbations added in eight directions. We use cost function weights $(w_t, w_s, w_p, w_\tau) = (200, 10, 10, 0.005)$. The character will roll 2, 4, 16, 50 cycles, respectively, during each of the four phases of optimization.

Results: We learn a successful first-order linear feedback policy for this skill using the full-body state and action descriptions, i.e., 88:1:39. With an extra optimization stage with terrain variations, the rolls are robust to these as well. Figure 5 illustrates example rolls on flat terrain and onto a step. Developing model-based controllers for this type of task would be difficult because of the rapidly changing ground contacts. In contrast, policy search methods do not need an explicit model of the dynamics; the impact of control is simply observed via policy rollouts. The robustness to pushes varies with respect to the push direction. For example, the character can recover from a large push to the forward right direction (460N), but only a small push to the forward left (220N). This is explained by the asymmetric parkour roll. The results are robust to 10 cm steps in the terrain.

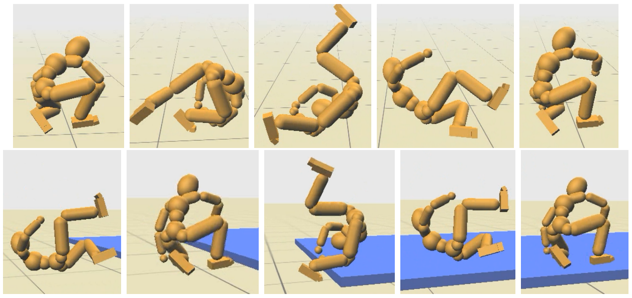


Figure 5: *Forward Rolling with reduced-order linear feedback.*

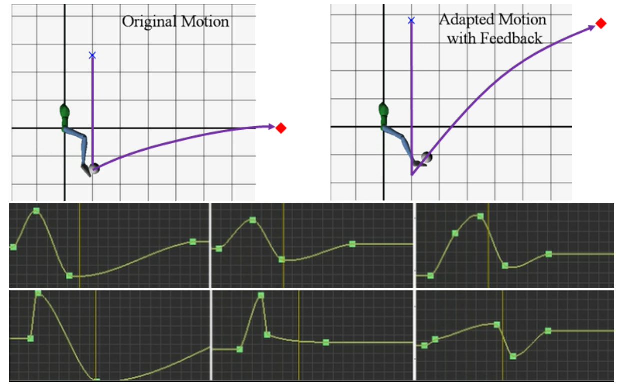


Figure 6: *Kicking*

5.5 Kicking

The kicking motion skill is the first of several discrete motions we investigage and is illustrated in Figure 6. It uses a 3-joint planar leg to kick a ball towards a target. A kicking feedback policy enables the reference kicking motion, which consists of a kick to a ball that is dropped from height h_{d_0} and hits a target at height p_{d_0}, to be adapted to work for different values of $[h_d, p_d]$. The action variables are given by the control points of the spline curve that defines the target joint angles over time for the kick. The control actions can adjust both the times u_t and corresponding values u_θ of the control points with respect to their default values, $[u_{t_0}, u_{\theta_0}]$. The sensory state consists of $\Delta h_d = h_d - h_{d_0}$ and $\Delta p_d = p_d - p_{d_0}$. Equation 18 gives the form of the feedback policy.

$$\begin{bmatrix} \delta u_t \\ \delta u_\theta \end{bmatrix}_{12\times1} = M \cdot \begin{bmatrix} \Delta h_{d_{1\times1}} \\ \Delta p_{d_{1\times1}} \end{bmatrix}_{2\times1} \tag{18}$$

In the optimization, we expect the leg to make contact with the ball and kick it towards the target point. This constraint is specified in the $U(M)$ term. A large penalty is applied in the $S(M)$ term when the character entirely misses the ball. For each simulation rollout, we use 112 pairs $[h_{d_i}, p_{d_i}]$ that span the desired range of target locations. Equation 19 shows the cost function.

$$\begin{aligned} cost(M) &= \sum_i (S_i(M) + U_i(M)) \\ S_i(M) &= \begin{cases} 0, & \text{the character kicks the ball;} \\ 100, & \text{the character fails to kick the ball.} \end{cases} \\ U_i(M) &= \|p_i - p_{d_i}\|_1 \end{aligned} \tag{19}$$

Results: Only full-order and first-order policies are considered for this motion task because the sensory state is two dimensional, The full-order policy succeeds in accurately hitting the target for significant variations in initial height and target height. The accompanying video shows the resulting accuracy as well as providing a visualization of how the PD-target angle splines change as a function of the sensed state variables. The linear policy is accurate at hitting the target despite the fact that the final ball trajectory is sensitive to the precise details of how the foot strikes the ball. Figure 6 depicts the reference trajectory and an example adapted trajectory. The first-order policy performs significantly worse, as can be expected given the two independent sources of required adaptations. The policy optimization is slow for this planar example because of the 112 example tasks that are run for each policy evaluation, with each of these taking 1–5 s of simulation time.

5.6 Juggling

In this motion skill a planar arm performs in-place juggling of a ball with a paddle, in the face of moderate perturbations. The planar arm has three degrees of freedom (shoulder, elbow, and wrist), as illustrated in Figure 7. The control actions are analogous to that used for kicking. The arm trajectory for the next hit is computed when the ball reaches its peak height. The sensory state consists of the state of ball, given by the position Δp, the horizontal velocity Δv_x, and the state of the arm, Δs:

$$\begin{bmatrix} \delta u_t \\ \delta u_\theta \end{bmatrix}_{24\times1} = M \cdot \begin{bmatrix} \Delta p_{2\times1} \\ \Delta v_{x_{1\times1}} \\ \Delta s_{18\times1} \end{bmatrix}_{21\times1} \tag{20}$$

One of the motion objectives is to have the ball repeatedly achieve the same state, as measured at the instant of its peak height. This state, $[p_i, v_x]$, should be similar to the initial state $[p_0, v_{x_0}]$. The cost function also rewards feedback policies that enable the system to succeed for a long time duration t_v before failing. Equation 21 gives the cost function for the optimization. Here, N refers to the number of juggling hits. A two stage optimization is applied. Optimization is first applied using 5 s rollouts, which is then extended to 15 s rollouts for the second stage.

$$\begin{aligned} cost(M) &= S(M) + U(M) \\ S(M) &= 100 \cdot t_v^{-0.95} \\ U(M) &= \sum_{i=1}^{N} \frac{20 \cdot \|p_i - p_0\|_2^2 + 10 \cdot \|v_{x_i} - v_{x_0}\|_2^2}{N} \end{aligned} \tag{21}$$

Results: This discrete task aims to achieve stable bouncing of the ball on the paddle. It achieves similar success for all synthesized linear policies, regardless of their order, as documented in Table 1. The spline control trajectories for three different hits are shown along the bottom of Figure 7. The juggling can recover from small external perturbations. It cannot cope with large perturbations, in part because the base of the arm is fixed in space.

5.7 Volleying

This motion skill uses two planar three-link arms to repeatedly volley a ball back-and-forth to each other. Figure 8 shows a typical volley trajectory. The initial controller only enables this system to achieve two or three volleys. The goal is thus to develop a feedback policy that supports sustained volleying in the face of moderate perturbations. The character configuration and the control inputs are the same as for juggling. The two arms use mirrored versions of the same control policy. The feedback policy is invoked to compute an adapted arm trajectory every time the ball crosses midline between the two arms. The sensory state is defined by observed deviations

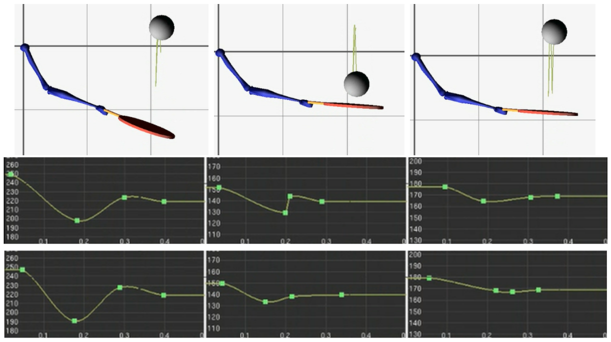


Figure 7: *Juggling*

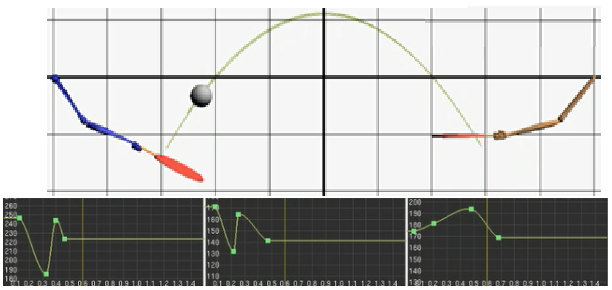


Figure 8: *Volleying*

in the state of the ball from that observed in the reference volley, as given by Δp and Δv, and the deviation of the state of the arm from its reference state, Δs. Equation 22 gives the applied feedback policy.

$$\begin{bmatrix} \delta u_t \\ \delta u_\theta \end{bmatrix}_{24\times 1} = M \cdot \begin{bmatrix} \Delta p_{y_{1\times 1}} \\ \Delta v_{2\times 1} \\ \Delta s_{18\times 1} \end{bmatrix}_{21\times 1} \tag{22}$$

The cost function is analogous to that used for juggling. At its peak height, the cost function penalizes deviations in the position, p, and horizontal velocity, v_x, with respect to their reference values, $p_0 = [2.5, 0.0]$ and $v_{x_0} = 2.5$. The cost function further rewards the time duration of successful volleying, t_v. Equation 23 details the cost function. We adopt the same two-stage incremental scheme as for juggling, using 5 and 30 s rollouts for the two stages.

$$\begin{aligned} &cost(M) = S(M) + U(M) \\ &S(M) = T - t_v \\ &U(M) = \frac{\sum_{i=1}^{N} \|p_i - p_0\|_2^2}{N} + \frac{\sum_{i=1}^{N} \|v_{x_i} - v_{x_0}\|_2^2}{N} \end{aligned} \tag{23}$$

Results: This motion skill is particularly challenging for feedback control, given the small tolerances that are required to achieve sustained volleys. Successful full-order and second-order solutions are learned, but the optimization was unable to achieve acceptable results using first-order and third-order policies. It should nevertheless be possible to develop a successful third-order policy, given that it subsumes the space of all second-order policies.

5.8 Ball Hitting

In this motion skill, a 3D physics-based character learns to adapt a ball-hitting stroke to hit different spatial target positions on a wall, p_{t_d}, as seen in Figure 9. The character has 17 links and 39 degrees of freedom. The tennis swing controller drives the 5 DOF right arm using splines that have a total of 38 DOF, as illustrated in the video. A standing controller is applied to other parts of the character. The reference swing motion enables the standing character to hit an upcoming ball towards a default target position, p_{t_0}, on the wall. The feedback policy is applied here to adapt the controller to different desired target positions, p_{t_d}. We use the same form of control parameters as previous examples and we use the the distance, Δp_t, between current desired wall position, p_{t_d}, and the default one, p_{t_0} as the sensory inputs of the feedback policy:

$$\begin{bmatrix} \delta u_t \\ \delta u_\theta \end{bmatrix}_{38\times 1} = M \cdot \begin{bmatrix} \Delta p_t \end{bmatrix}_{2\times 1} \tag{24}$$

In the optimization, $S(M)$ is used to penalize policies that fail to strike the ball. In the $U(M)$ term, if the ball falls to the ground before hitting the wall, $S(M)$ will set a penalty according to the remaining horizontal distance to the target position on the wall, $d_{xz} = \|p_{ball_{xz}} - p_{wall_{xz}}\|_2$. Otherwise, it rewards policies where the ball hits the wall at a position, p_c, nearby the target, p_{t_d}. For each evaluation in the optimization process, we evaluate feedback policies for eight different desired target positions. Equation 25 gives the cost function for this example.

$$\begin{aligned} &cost(M) = \sum_i (S_i(M) + U_i(M)) \\ &S_i(M) = \begin{cases} 0, & \text{the character hits the ball;} \\ 100, & \text{the character fails to hit the ball.} \end{cases} \\ &U_i(M) = \begin{cases} 10 \cdot d_{xz}^2, & \text{the ball fails to hit the wall.} \\ 10 \cdot \|p_c - p_{t_d}\|_2^2, & \text{the ball hits the wall;} \end{cases} \end{aligned} \tag{25}$$

Results: The full order solution works well, while the first order result is much less accurate which is unsurprising given the 2D parameterization of the target position. The linear feedback structure can be thought of as defining a set of action synergies that span the 38 actuated degrees of freedom of the standing character.

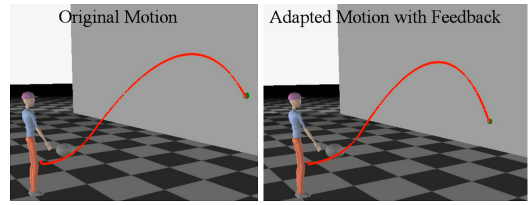
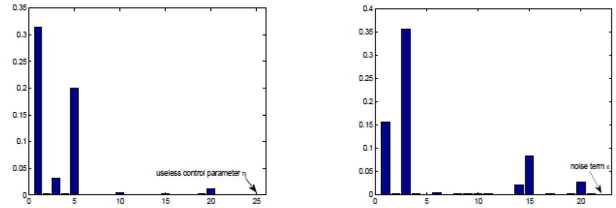


Figure 9: *Ball hitting*

6 Discussion

Model Reduction: The reduced dimensional space can be seen as modeling a compact "summary state" from the supplied sensory state, in which case M_{sp} can be seen as a type of state estimation matrix. Alternatively, it can also be seen as modeling a reduced-dimension actuation basis. For this case, M_{ap} can be seen as a matrix that defines the actuation synergies.

There is a compromise to be made in setting the desired dimensionality, r, of the reduced-order feedback. Having more dimensions allows for additional flexibility in the feedback. However, it also introduces a larger number of parameters, thereby increasing the optimization time and the likelihood of ending in a local minima. Our choice of feedback matrix factorization is such that it will still contain redundancies. For example, the pair of matrices $M_{ap} \cdot M_{sp}$ and $M'_{ap} \cdot M'_{sp}$ represent identical feedback structures when $M'_{ap} = a \cdot M_{ap}$ and $M'_{sp} = (1/a) \cdot M_{sp}$.



(a) L1 norms of M_{ap} row vectors (b) L1 norms of M_{sp} column vectors

Figure 10: *Results of Regularization for Juggling*

As described earlier, the $R(M)$ regularization term in the cost function can be used to achieve further sparsity in the feedback policy. We test this for juggling using a variant of the (21:1:24) policy. We augment the control actions with an additional variable that has no impact on the dynamics, and augment the sensory inputs with a variable that samples from a Gaussian distribution, $\mathcal{N}(0, 1)$ each time the feedback policy is invoked. This yields a (22:1:25) policy. The weight of the regularization term is set to 3.0. Figure 10 shows that the row and column vectors that correspond to these features are effectively eliminated, and shows a sparse first-order feedback policy that is dominated by 5 sensory features and 3 actuators.

Optimization method: We have tested the impact of using a simple local optimization method instead of the CMA stochastic optimization. We implement a greedy stochastic local search algorithm and compare its performance on the balancing and volleying tasks. For balancing, the two methods converge to a solution with very similar performance. For the case of volleying, the local optimization method fails to find a solution of comparable quantity. The convergence to local minima using local optimization methods is not surprising, as the reduced-order design problem is non-convex and non-smooth even for linear systems [Burke et al. 2003].

Contrast with model-based feedback methods: Sophisticated model-based control methods offer an alternative approach, as we have full access to the underlying model. However, model-based solutions currently require significant expertise to deploy in comparison to the black-box forward dynamics simulator needed for our approach. The methods proposed in this paper can also be easily applied to motions such as rolling, where the impact of effects such as rapidly changing contacts, friction, and ground compliance are all implicitly modeled through the episodic rollouts.

Limitations: Our work has a number of limitations. We do not currently allow for phase-dependent feedback because the larger number of parameters is likely be problematic for our current optimization methodology. However, it might be possible to use the static output feedback as a point of departure for subsequent refinement of the feedback in a phase-dependent fashion. The design of the cost function still requires care and there may be the need to shape the optimization using multiple stages. The current feedback policy currently allows no direct user control over the final style of feedback. The scalability of the method could still benefit from further investigation. Currently, policy optimization requires significant offline computation, although it is highly amenable to parallelization.

7 Conclusions

We have presented a method that can be used to learn reduced-order linear feedback policies for a variety of motion skills. The user provides a reference motion, a set of sensory state variables, a set of action variables, and a skill-specific cost function. Given this input, the method is then capable of synthesizing task-specific reduced-order linear control policies. Our results demonstrate that compact linear feedback policies can be effective for controlling a diverse range of motion skills. They further show that motion skills can be robustly controlled using very low-order feedback paths (1–3 dimensions), while optimizations can fail for full-feedback cases.

There are numerous directions for future work. We wish to learn feedback policies for articulated figure models that are rigged with musculotendon models. The generic, embodied nature of our approach means that muscle activations can simply be treated as another type of control action. The impact of muscle dynamics and muscles that span multiple joints would be implicitly modeled during policy rollouts. It would be interesting to compare and contrast the learned feedback policies to human feedback strategies.

The optimization process might be improved upon in several ways. Recent work in robotics and machine learning points to several strategies that reduce the number of required rollouts. It may be possible to bootstrap from one learned reduced-order feedback policy to another for a related motion skill, or to bootstrap from observations of human feedback strategies. We wish to develop a principled method for creating the staged learning that is needed for some of our tasks. The feedback structure itself can also be adapted in several ways. Optimizing for affine feedback would allow the reference sensory state and actuation to be automatically tuned to achieve the best result.

Acknowledgements

We gratefully acknowledge funding from NSERC, GRAND NCE, and the Singapore Ministry of Education Academic Research Fund Tier 2 (MOE2011-T2-2-152).

References

BARBIČ, J., DA SILVA, M., AND POPOVIĆ, J. 2009. Deformable object animation using reduced optimal control. In *ACM SIGGRAPH 2009 papers*, ACM, SIGGRAPH '09, 53:1–53:9.

BERNIKER, M., JARC, A., BIZZI, E., AND TRESCH, M. C. 2009. Simplified and effective motor control based on muscle synergies to exploit musculoskeletal dynamics. *Proceedings of the National Academy of Sciences 106*, 18, 7601–7606.

BOX2D. , http://box2d.org/.

BURKE, J., LEWIS, A., AND OVERTON, M. 2003. A nonsmooth, nonconvex optimization approach to robust stabilization by static output feedback and low-order controllers. In *Proc. ROCOND 2003*.

BURNS, J. A., AND KING, B. B. 1998. A reduced basis approach to the design of low-order feedback controllers for nonlinear continuous systems. *Journal of Vibration and Control 4*, 3, 297–323.

COROS, S., BEAUDOIN, P., AND VAN DE PANNE, M. 2010. Generalized biped walking control. *ACM Transctions on Graphics 29*, 4, Article 130.

DA SILVA, M., ABE, Y., AND POPOVIĆ, J. 2008. Interactive simulation of stylized human locomotion. *ACM Trans. Graph. 27* (August), 82:1–82:10.

DAVID, J., AND DE MOOR, B. 1994. Designing reduced order output feedback controllers using a potential reduction method. *American Control Conference, 1994 1*, 845–849.

DE LASA, M., MORDATCH, I., AND HERTZMANN, A. 2010. Feature-Based Locomotion Controllers. *ACM Transactions on Graphics 29*, 3.

DE OLIVEIRA, M., AND GEROMEL, J. 1997. Numerical comparison of output feedback design methods. In *American Control Conference, 1997. Proceedings of the 1997*, vol. 1, 72 –76 vol.1.

HANSEN, N. 2006. The CMA evolution strategy: a comparing review. In *Towards a new evolutionary computation. Advances on estimation of distribution algorithms*, J. Lozano, P. Larranaga, I. Inza, and E. Bengoetxea, Eds. Springer, 75–102.

JAIN, S., AND LIU, C. K. 2011. Modal-space control for articulated characters. *ACM Trans. Graph. 30* (October), 118:1–118:12.

JAMES, D. L., AND PAI, D. K. 2002. Dyrt: dynamic response textures for real time deformation simulation with graphics hardware. *ACM Trans. Graph. 21* (July), 582–585.

KRY, P. G., REVERET, L., FAURE, F., AND CANI, M.-P. 2009. Modal locomotion: Animating virtual characters with natural vibrations. *Computer Grahics Forum 28*, 2.

KWON, T., AND HODGINS, J. K. 2010. Control systems for human running using an inverted pendulum model and a reference motion capture sequence. *The ACM SIGGRAPH / Eurographics Symposium on Computer Animation (SCA 2010)*.

LALL, S., MARSDEN, J. E., AND GLAVAŠKI, S. 2002. A subspace approach to balanced truncation for model reduction of nonlinear control systems. *International journal of robust and nonlinear control 12*, 6, 519–535.

LASZLO, J., VAN DE PANNE, M., AND EUGENE, F. 1996. Limit cycle control and its application to the animation of balancing and walking. In *SIGGRAPH '96*, ACM, 155–162.

LEE, Y., KIM, S., AND LEE, J. 2010. Data-driven biped control. *ACM Trans. Graph. 29* (July), 129:1–129:8.

LEWIS, F., AND SYRMOS, V. 1995. *Optimal control*. A Wiley-Interscience publication. J. Wiley.

LIU, L., YIN, K., VAN DE PANNE, M., SHAO, T., AND XU, W. 2010. Sampling-based contact-rich motion control. *ACM Transctions on Graphics 29*, 4, Article 128.

LIU, L., YIN, K., VAN DE PANNE, M., AND GUO, B. 2012. Terrain runner: control, parameterization, composition, and planning for highly dynamic motions. *ACM Trans. Graph. 31*, 6, 154.

LIU, L., YIN, K., WANG, B., AND GUO, B. 2013. Simulation and control of skeleton-driven soft body characters. *ACM Transactions on Graphics (TOG) 32*, 6, 215.

MACCHIETTO, A., ZORDAN, V., AND SHELTON, C. R. 2009. Momentum control for balance. *ACM Trans. Graph. 28* (July), 80:1–80:8.

MORDATCH, I., DE LASA, M., AND HERTZMANN, A. 2010. Robust Physics-Based Locomotion Using Low-Dimensional Planning. *ACM Transactions on Graphics 29*, 3.

MUICO, U., LEE, Y., POPOVIĆ, J., AND POPOVIĆ, Z. 2009. Contact-aware nonlinear control of dynamic characters. *ACM Transactions on Graphics 28*, 3.

ODE. Open dynamics engine, `http://www.ode.org/`.

RAIBERT, M. H., AND HODGINS, J. K. 1991. Animation of dynamic legged locomotion. *SIGGRAPH Comput. Graph. 25* (July), 349–358.

SAFONOVA, A., HODGINS, J. K., AND POLLARD, N. S. 2004. Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. *ACM Trans. Graph. 23* (Aug.), 514–521.

SCHERER, C., GAHINET, P., AND CHILALI, M. 1997. Multiobjective output-feedback control via lmi optimization. *Automatic Control, IEEE Transactions on 42*, 7, 896–911.

SLOTINE, J.-J. E., LI, W., ET AL. 1991. *Applied nonlinear control*, vol. 60. Prentice-Hall Englewood Cliffs, NJ.

SOK, K. W., KIM, M., AND LEE, J. 2007. Simulating biped behaviors from human motion data. *ACM Trans. Graph. 26* (July).

TAN, J., GU, Y., LIU, C. K., AND TURK, G. 2014. Learning bicycle stunts. *ACM Transactions on Graphics (TOG) 33*, 4, 50.

TREUILLE, A., LEWIS, A., AND POPOVIĆ, Z. 2006. Model reduction for real-time fluids. *ACM Transactions on Graphics 25*, 3 (July), 826–834.

TSAI, Y.-Y., LIN, W.-C., CHENG, K. B., LEE, J., AND LEE, T.-Y. 2010. Real-time physics-based 3d biped character animation using an inverted pendulum model. *IEEE Transactions on Visualization and Computer Graphics 16* (March), 325–337.

WANG, J. M., FLEET, D. J., AND HERTZMANN, A. 2009. Optimizing walking controllers. *ACM Trans. Graph. 28* (December), 168:1–168:8.

WANG, J. M., FLEET, D. J., AND HERTZMANN, A. 2010. Optimizing walking controllers for uncertain inputs and environments. *ACM Trans. Graph. 29* (July), 73:1–73:8.

YE, Y., AND LIU, C. K. 2010. Optimal feedback control for character animation using an abstract model. *ACM Trans. Graph. 29* (July), 74:1–74:9.

YIN, K., LOKEN, K., AND VAN DE PANNE, M. 2007. Simbicon: Simple biped locomotion control. *ACM Trans. Graph. 26*, 3, Article 105.