EUROGRAPHICS 2010 / T. Akenine-Möller and M. Zwicker
(Guest Editors)

Volume 29 (2010), Number 2

Synthesis of Responsive Motion Using a Dynamic Model

Yuting Ye and C. Karen Liu[†]

Georgia Institute of Technology

Abstract

Synthesizing the movements of a responsive virtual character in the event of unexpected perturbations has proven a difficult challenge. To solve this problem, we devise a fully automatic method that learns a nonlinear probabilistic model of dynamic responses from very few perturbed walking sequences. This model is able to synthesize responses and recovery motions under new perturbations different from those in the training examples. When perturbations occur, we propose a physics-based method that initiates motion transitions to the most probable response example based on the dynamic states of the character. Our algorithm can be applied to any motion sequences without the need for preprocessing such as segmentation or alignment. The results show that three perturbed motion clips can sufficiently generate a variety of realistic responses, and 14 clips can create a responsive virtual character that reacts realistically to external forces in different directions applied on different body parts at different moments in time.

Categories and Subject Descriptors (according to ACM CCS): Computer Graphics [I.3.3]: Animation—

1. Introduction

A challenging task in computer animation is the synthesis of responsive virtual characters that match the agility of humans when reacting to unexpected dynamic disturbances. In such uncertain environments, the strategies humans employ to respond and to recover are difficult to categorize or to describe procedurally. Researchers have tackled this problem with physics simulation and data-driven techniques. Although physics simulation can successfully synthesize realistic passive reactions to arbitrary forces, the design of a robust and purposeful controller operating against disturbances remains a difficult task. Alternatively, data-driven techniques use real-world examples to directly capture the agility and the style of human motions, but they rely on a large dataset of motions that cover response and recovery behaviors induced by all possible perturbations. In addition to the cost of acquisition and storage, the key issue with a large dataset is that it requires tedious preprocessing, such as segmentation, classification, or warping. Even if one manages to generate a large dataset with perfectly aligned motions, selecting the most appropriate motion clips based on physical perturbations is still challenging due to the difficulty of measuring and quantifying perturbations during motion acquisition.

To address these problems, we propose a fully automatic, parameter-free approach to synthesizing the motion of a responsive virtual character reacting to arbitrary, unexpected external forces during walking. Our algorithm discovers the dynamic structure of input motions using a Gaussian process-based nonlinear dynamic model in a low-dimensional space. Our method overcomes the main issue associated with large datasets because the model allows imperfect input data without any preprocessing as a result of its predictive power. This model predicts the most probable pose based on the current dynamic state of the character at every frame, which results in a motion similar to the input both kinematically and dynamically. During perturbations, we use physics simulation to deform the current pose, which subsequently triggers a transition toward an appropriate example in subsequent inferences. Interleaving model inference and physics simulation in this fashion can generate transitions among multiple examples under persistent forces and thus produce novel variations in the synthetic motions. We highlight the predictive power of our algorithm using a relatively small set of examples. We also demonstrate that a wider variety of motions can be synthesized by increasing the number of examples and dividing them into local mod-

[†] {yuting,karenliu}@cc.gatech.edu

els and then choosing among them at runtime. These design choices lead to a novel method that automatically selects the appropriate motions based on the dynamic states of the character.

Our results demonstrate the generative power of this method by synthesizing a continuous walking sequence with responses to various perturbations arbitrarily provided by the user. The underlying model for the synthetic sequence contains only three perturbed examples and a normal sequence, yet it produces realistic human reactions to different perturbations, indicating that our dynamic model and the simulation are consistent with the laws of physics. We also show a richer variety of responses generated from 14 perturbed examples grouped into four models. However, our algorithm is still limited to the range of motions that can be interpolated by the original data. For example, the character cannot respond to a perturbation that requires a drastically different recovery strategy unseen in the examples.

This paper has two major contributions: 1) We use a nonlinear dynamic model to organize a small amount of motion data without preprocessing and parameter tuning; and 2) we propose a method that applies dynamic responses to existing data based on physics simulations.

2. Related work

Physics simulation with dynamic controllers can generate realistic responses to physical perturbations. These controllers are either manually designed [YLvdP07, SCCH09] or optimized [AdSP07, YCBvdP08, CBYvdP08, WFH09] for robust behaviors. They can also be concatenated or combined to exhibit a wide range of dynamic responses [FvdPT01, dSDP09, CBvdP09, JYL09]. The success of these controllers usually depends on the careful tuning of parameters or a complex optimization procedure and robust balance strategies that sacrifices motion quality. To improve motion quality and facilitate the design of such dynamic controllers, researchers have proposed learning them automatically from captured motions. Controllers are developed to create transitions between poses [PZ05] or two sequences [SPF03, ZMCF05, Nat] or to reproduce an entire sequence under perturbations [ZH02, YCP03, SKL07, dSAP08, YL08, MZS09]. In particular, Muico *et al.* [MLPP09] derive controllers from multiple motion sequences that automatically adjust themselves to environmental contacts. These controllers perform robustly and conform well to the example motions because the physical parameters are tailored to the designated motions. However, they are also restricted to synthesizing motions very similar to the input, which greatly limits the range of allowable perturbations.

Natural responsive motions can also be generated by blending or concatenating motion capture sequences directly. To produce high quality animation, most data-driven algorithms require motion clips to be kinematically and structurally similar [KGP02, LCR*02]. This requirement becomes increasingly challenging in more dynamic and unpredictable scenarios such as a character physically interacting with the environment. To synthesize a standing person responding to pushes on the waist, Yin *et al.* [YPvdP05] used linear and angular momentum profiles to parameterize the motion space, largely reducing the required examples (66 motion sequences). Our method handles more challenging situations in which the characters must maintain a natural walking motion against arbitrary perturbations. Arikan and Forsyth [AFO05] achieved a similar goal with over a thousand examples in an effort to train an oracle that could predict perceptually pleasing deformations under perturbations. While their results are very compelling, considerable manual processing and user input are required to learn a powerful oracle. Because they built the oracle and applied nearest neighbor method directly in the high-dimensional parameterization space, a large number of examples is required to ensure smooth interpolation. Our method instead works in a low-dimensional space that reflects the most essential differences in perturbed motions so that it requires far fewer data to generalize a large range of new situations. This design choice also greatly eases the burden of processing a large volume of raw input data.

A special challenge in designing efficient algorithms is the high dimensionality in human motion. Li *et al.* [LMFP08] showed that a linear dynamic probabilistic model takes hours to infer transitions between two sequences in the space of joint positions. Many researchers seek to simplify computation using dimension reduction techniques [SHP04, CH07, TLP06, BP08]. While linear models are simple and effective, we are interested in nonlinear models such as the Gaussian process latent variable model (GPLVM) [Law04] because they provide a significantly smaller space that captures the nonlinear nature of human motions. GPLVM-based methods have been successfully applied to character animation [GMHP04] and computer vision [UFHF05]. In particular, we use SGPLVM with back constraints [LQC06] because it provides smooth two-way mapping between high- and low-dimensional spaces, allowing us to learn a meaningful dynamic model in the low-dimensional space.

Similar to modeling human poses, many researchers approximate human motion dynamics with statistical models [BH00, LWS02]. Lau *et al.* [LBJK09] used a dynamic Bayesian network to synthesize natural spatial and temporal variations from a small set of data. Gaussian process is used by Ikemoto *et al.* [IAF09] to transfer animators' edits of a character's motion to another motion, and by Ma *et al.* [MLD09] to transfer facial animations. In contrast, we are interested in modeling the transition dynamics from input motions. In this regard, we use a variation of the Gaussian process dynamic model (GPDM) [WFH06, UFF06] because it can accurately capture walking dynamics from a single example. Our method differs from GPDM in two aspects. First, GPDM learns the motion representation and dynamics simultaneously because it must balance consistency with the data and smoothness in dynamics. Consequently, it usually yields a difficult optimization problem sensitive to the scaling of the variables and the initial state. In contrast, our method separates the learning of motion representations and dynamics into two simple optimization problems thanks to the smooth latent space resulting from the use of back constraints. Second, GPDM synthesizes the entire sequence at once while we synthesize new motions on a per frame basis to incorporate interactive perturbations. Because our predictions under perturbations always lie on high probability regions, the results we produce are similar to what GPDM would produce as an entire sequence.

3. Algorithm

Our method consists of a model learning phase and a motion synthesis phase. The learning phase directly takes a few captured sequences as input without preprocessing, producing a nonlinear dynamic model that represents the data in a low-dimensional space. This dynamic model is then used in the synthesis phase to produce new sequences responsive to interactive user perturbations. At each time instance, we solve for a new pose by maximizing its likelihood in the learned model given the current dynamics. Interactive user input can then deform the new pose according to the laws of physics.

3.1. Model learning

Given a small number of input examples, which include a normal walk sequence and several perturbed ones that exhibit dynamic responses and recovery, our task is to model the variations between the normal and perturbed motions. We first learn a low-dimensional space, called *latent* space, which captures the most essential differences among the input motions, then we learn a dynamical model in the latent space to capture the motion transition. The latent space is solved using a nonlinear dimensional reduction method, scaled GPLVM with back constraints. This method provides a nonparametric probabilistic model that uses a low dimensional space to explain the variations in the input motions. Scaled GPLVM is an effective algorithm that reduces the dimension of data from very few examples and takes into account the intrinsic scales in each input dimension. Augmenting with back constraints, this algorithm produces a smooth model that enables us to learn a sensible dynamic model in the latent space as a separate process. The dynamic model, formulated as a Gaussian process with second-order dynamics, provides a generative mechanism for synthesizing new sequences kinematically and dynamically similar to the input motions.

Directly from the captured motions without any preprocessing such as time alignment or annotation, we compute a set of features for each pose. A feature vector contains two consecutive poses that correspond to $\frac{1}{30}$ of a second for temporal coherence. Ideally, we would like to choose a minimal set of features from which important dynamic structures can be revealed and joint configurations can be efficiently reconstructed with little ambiguity. Our first attempt was to use joint degrees of freedom (DOFs), but they failed to capture the dynamic structure of the motions. Joint position has also been widely used as a feature. However, we could not reconstruct the full body configuration from joint positions alone due to ambiguities in the end effectors. Instead, we chose the center-of-mass (COM) positions of each body node expressed in the character's local coordinates to align the first frame in each input sequence. In addition, global velocity is included to compute global positions from integration. Finally, because heel strike and toe off events are important for walking motions, we replaced the COM of the feet with heel and toe positions. In total, a feature vector has 120 dimensions for a character represented by 18 body nodes and 42 DOFs. Before training, we subtract from each vector the mean value computed from the normal walk with complete cycles because we want an unbiased mean that represents the expected behavior.

Given N D-dimensional feature vectors $\mathbf{Y}_{1:N}$, we want to solve for the corresponding d-dimensional vectors $\mathbf{X}_{1:N}$ in a low dimensional space with $d < D$. GPLVM provides such a mapping as the conditional likelihood of $\mathbf{Y}$ given $\mathbf{X}$, modeled as a Gaussian distribution with a covariance matrix $\mathbf{K}(\mathbf{X}, \Theta)$, Θ being the hyperparameter of the model (Equation (1)).

$$p(\mathbf{Y}|\mathbf{X},\Theta) = \frac{1}{(2\pi)^{\frac{DN}{2}}|\mathbf{K}|^{\frac{D}{2}}}\exp(-\frac{1}{2}\mathrm{tr}(\mathbf{K}^{-1}\mathbf{Y}\mathbf{Y}^T)). \quad (1)$$

We use maximum a posteriori (MAP) estimation to compute $\mathbf{X}$ and Θ as the minimum of Equation (2), assuming a Gaussian prior on $\mathbf{X}$ and a uniform prior on Θ (see Appendix A for details).

$$L_P = -\ln p(\mathbf{Y}|\mathbf{X},\Theta) - \ln p(\mathbf{X}) - \ln p(\Theta). \quad (2)$$

Once $\mathbf{X}$ is solved, we want to model its time evolution in the latent space as an approximation of the dynamics in the feature space. Similar to GPDM [WFH06], we model the dynamics in $\mathbf{X}$ as a second-order autoregressive series, which results in a joint Gaussian distribution $p(\mathbf{X}|\Phi)$, where Φ captures the hyperparameters in this model (Appendix A). Since $\mathbf{X}$ is known here, we need to solve for only Φ by minimizing Equation (3) :

$$L_D = -\ln p(\mathbf{X}|\Phi) - \ln p(\Phi). \quad (3)$$

We use scaled conjugate gradient [Møl93] to solve both problems. With $d = 3$ and N around 300 in all our tests, $\mathbf{X}$ and Θ are solved in 10 minutes, and Φ is solved within a minute. We experienced with different latent dimensions and found that while a 2D space creates many intersections in latent trajectories, a 3D latent space is sufficient to represent the data accurately. We also considered a 4D space when we found a first-order dynamic model unable to reproduce the training motion dynamics. Our experiments show that upgrading to second-order dynamic model significantly improves accuracy while increasing the dimension of the latent space to 4D has little improvement.

3.2. Motion synthesis

We can generate new motions similar to the examples by inferring the latent space model with a dynamic prior. When no perturbations occur, this dynamic model can synthesize an infinite normal walking sequence from an initial state. When a user applies perturbations such as a push, we deform the subsequent pose using a physics-based method and update its latent position accordingly. As a result, the new latent position deviates from the normal motion and moves toward an example with similar perturbation dynamics. The new position also affects future poses because it perturbs the dynamics of the latent trajectory. Depending on the push, the perturbed trajectory may then follow one single example or navigate among a few of them. At the end of the perturbation, the latent trajectory exhibits responses by following the latent dynamics and eventually transitions back to normal walking. As long as the latent trajectory stays in the high probability region in the model, the synthesized motion appears to use similar strategies as the examples in reaction to perturbations.

Figure 1 summarizes the motion synthesis procedure for frame $n+1$. To handle continuous user input, this procedure is repeated at every frame to formulate different optimization problems that are solved with sequential quadratic programming [GSM96]. Given the current motion and its latent trajectory, we infer the next pose $\tilde{\mathbf{q}}_{n+1}$ and its latent position $\tilde{\mathbf{x}}_{n+1}$ from the latent model. Without any perturbations, $\tilde{\mathbf{q}}_{n+1}$ is directly output as the desired pose ($\mathbf{q}_{n+1} = \tilde{\mathbf{q}}_{n+1}$). If the character is perturbed, we modify $\tilde{\mathbf{q}}_{n+1}$ based on the laws of physics to get the final pose $\mathbf{q}_{n+1}$ and solve for its latent position $\mathbf{x}_{n+1}$. The results are then fed back to the pipeline for subsequence synthesis. We describe each component of the algorithm in detail as follows:

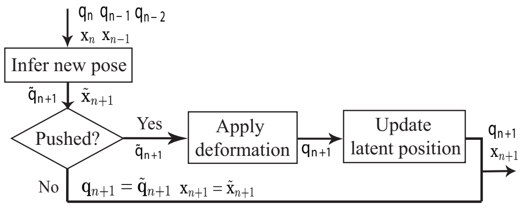


Figure 1: *Motion synthesis flow chart for frame $n+1$.*

Infer new pose. Denoting a new sequence as $\hat{Y}$ and its latent trajectory as $\hat{X}$; we want to maximize the likelihood of this new pair given the training set. The conditional likelihood can be written in terms of the learned models using Bayes' rule as in Equation (4):

$$p(\hat{Y}, \hat{X} | \mathbf{Y}, \mathbf{X}, \Theta, \Phi) \propto p(\hat{Y}, \mathbf{Y} | \hat{X}, \mathbf{X}, \Theta) p(\hat{X}, \mathbf{X} | \Phi). \quad (4)$$

Because the second-order dynamic model takes current and previous latent positions to infer the next one, the new sequence takes three feature vectors in which the last one is a function of the new pose $\tilde{\mathbf{q}}_{n+1}$. We can then solve for $\tilde{\mathbf{q}}_{n+1}$ and $\tilde{\mathbf{x}}_{n+1}$ by minimizing Equation (5): $\tilde{\mathbf{q}}_{n+1}, \tilde{\mathbf{x}}_{n+1} = \arg\min L_S(\mathbf{q}_{n+1}, \mathbf{x}_{n+1})$, where

$$\begin{aligned} L_S(\mathbf{q}_{n+1}, \mathbf{x}_{n+1}) = & -\ln p(\hat{Y}(\mathbf{q}_{n+1}), \mathbf{Y} | \hat{X}(\mathbf{x}_{n+1}), \mathbf{X}, \Theta) \\ & -\ln p(\hat{X}(\mathbf{x}_{n+1}), \mathbf{X} | \Phi). \end{aligned} \quad (5)$$

Apply deformation. With interactive user perturbations, we apply perturbation forces to deform $\tilde{\mathbf{q}}_{n+1}$ based on Lagrange's equation of motion. The deformed pose $\mathbf{q}_{n+1}$ will satisfy Equation (6):

$$\left(\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_j} - \frac{\partial L}{\partial q_j} \right) \bigg|_{\mathbf{q}_{n+1}} = u_j + \mathbf{J}_j^{c\,T} \mathbf{f}_c + \mathbf{J}_j^{ext\,T} \mathbf{f}_{ext}. \quad (6)$$

Here, L is the Lagrangian of the dynamic system, u_j the internal torque in q_j, $\mathbf{J}_j^c$ the j^{th} column of the Jacobian matrix that projects the ground contact force $\mathbf{f}_c$ onto q_j, and $\mathbf{J}_j^{ext}$ projects the external forces $\mathbf{f}_{ext}$ onto q_j. We use finite differences to compute the joint velocity and the acceleration as $\dot{\mathbf{q}}_n \equiv \frac{1}{\Delta t}(\mathbf{q}_n - \mathbf{q}_{n-1})$ and $\ddot{\mathbf{q}}_n \equiv \frac{1}{\Delta t^2}(\mathbf{q}_{n+1} - 2\mathbf{q}_n + \mathbf{q}_{n-1})$.

Due to the lack of force information in the kinematics data, we have to approximate the generalized forces with certain assumptions. We assume that the internal torques are roughly the same as they are in the predicted motion and that the ground reaction forces can only change smoothly. Therefore, we approximate the generalized forces from the predicted pose $\tilde{\mathbf{q}}_{n+1}$ as in Equation (7). Compared to other deformation methods such as the one in [AFO05], we found our method generates poses that are more consistent with the dynamic model.

$$u_j + \mathbf{J}_j^{c\,T} \mathbf{f}_c \approx \left(\frac{d}{dt} \frac{\partial L}{\partial \dot{q}_j} - \frac{\partial L}{\partial q_j} \right) \bigg|_{\tilde{\mathbf{q}}_{n+1}}. \quad (7)$$

We substitute the right-hand side of Equation (7) into Equation (6). Because $\frac{\partial L}{\partial q_j}$ does not depend on $\tilde{\mathbf{q}}_{n+1}$ or $\mathbf{q}_{n+1}$, we cancel it on both sides and arrive at the following objective:

$$G_j(\mathbf{q}_{n+1}) = \frac{d}{dt} \frac{\partial L}{\partial \dot{q}_j} \bigg|_{\mathbf{q}_{n+1}} - \frac{d}{dt} \frac{\partial L}{\partial \dot{q}_j} \bigg|_{\tilde{\mathbf{q}}_{n+1}} - \mathbf{J}_j^{ext\,T} \mathbf{f}_{ext}. \quad (8)$$

Without perturbations, the foot contacts are maintained very well thanks to a good dynamic model. However, Equation (8) may break the contacts with external forces. To prevent foot sliding during perturbations, an additional soft positional constraint $\mathbf{C}(\mathbf{q}_{n+1})$ is imposed on the support foot, which is determined by simple velocity thresholding. Together we have $\mathbf{q}_{n+1} = \arg\min \frac{1}{2}(\|\mathbf{G}\|^2 + \|\mathbf{C}\|^2)$, where $\mathbf{G}$ consists of G_j for all the DOFs.

Update latent position. Finally, we solve for the latent position $\mathbf{x}_{n+1}$, which corresponds to $\mathbf{q}_{n+1}$, by minimizing Equation (5) again.

4. Results

To evaluate our algorithm, we synthesize a walking character responding to arbitrary pushes from input data captured at 30 frames per second. We first test our algorithm on a small data set with one normal walk and three different perturbed walks in which the same subject takes one or two steps to recover from an unexpected push. We then work with a larger data set of 14 perturbed walks grouped into four different models (Figure 4). Each input motion contains one or two walking cycles ranging from 50 to 70 frames, totaling about 300 poses in each model.

Latent model learning. Given a normal walk and three perturbed walks, we learn a three-dimensional latent space that represents the most significant variations in the 120-dimensional feature space. Figure 2 shows the resulting latent trajectories. This model correctly organizes the normal walk into two limit cycles, but the three perturbed motions form locally-deformed cycles around the normal walk and gradually transition to it toward the end (in counter-clockwise order). The beginning and the end of the deformed cycles coincide with the onset and recovery moments of the perturbed motions, respectively. The result demonstrates that our model captures the fundamental dynamic structures of the input motions without any prior knowledge of the motion contents such as walking phases or different recovery strategies used.

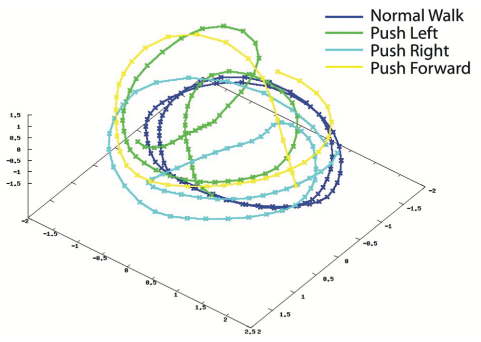
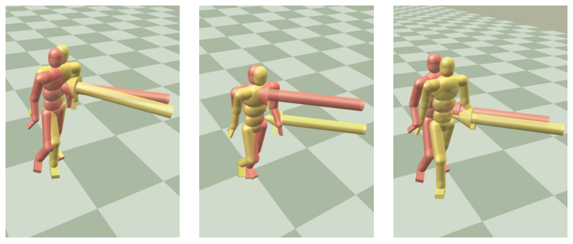


Figure 2: *Latent trajectories of four walking motions.*

Motion synthesis. We conduct several experiments to test the generalization ability of our algorithm. We first test whether the dynamic model can generate smooth transitions between motions by reproducing the forward push example in the above model (Figure 2). Because these motions are not aligned, our algorithm needs to produce transitions between the normal walk and the forward push. When we apply an estimated forward push to a synthesized normal walk, the virtual character correctly responds by gradually transitioning to the forward push example. Accordingly, the corresponding latent trajectory smoothly connects the two examples with intermediate points rather than snapping immediately to the closest point in the push motion or changing abruptly to random positions. After reaching the end of the forward push example, the synthesized trajectory transitions back to the normal walk following the dynamic flows learned from the other examples.

In the next test, we evaluate the generalizability of a single example motion using a model with one normal walk and one backward push sequence in which the push is applied on the left side of the subject's torso during right-leg support. We apply pushes to the character from different directions on different body parts and at different timing (Figure 3). Results show that one example can generalize to a reasonable range of new situations. For example, different directions of backward pushes induce realistic responsive motions. However, the character responds only marginally to a forward push and quickly resumes the normal walk. We observe that different magnitudes of force and points of application induce smaller variations than different timing and force directions. Therefore, they are more generalizable.

We further test the capability of the model in Figure 2 by applying random pushes to generate a continuous responsive sequence. The resulting motion reacts immediately to unexpected perturbations with various recovery strategies seen in the examples, such as changing step size and timing. Pushes applied during recovery lead to smooth transitions among different examples without any visual discrepancy. From all the results we produce, the synthesized trajectories are smooth, and they remain in the high probability region in the latent space.



(a) Directions (b) Body parts (c) Timing

Figure 3: *Perturbations (indicated by arrows) in different situations. In Figure 3(c), the yellow character is pushed at a later time.*

Model switching. Given more examples, a model is generalizable to a wider range of situations. However, the performance of our algorithm decreases because a higher-dimensional latent space is needed to capture more variations in a large input dataset. To this end, we design a simple model-switching scheme based on two observations: 1) Trajectories that are significantly different and thus far away in the latent space have little chance to be interpolated; and

2) perturbation directions and timing play a more important role in dynamic responses. As proof of concept, we simply use the force direction as a criterion to divide examples into different models that we choose at runtime.

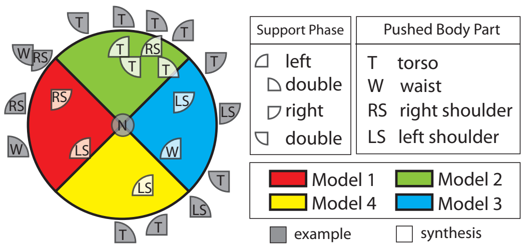


Figure 4: *Example perturbations vs. synthetic perturbations. The angle of the circle indicates perturbation directions (character facing north), and the distance to the center indicates perturbation magnitudes (only for synthetic ones).*

We arrange 14 perturbed motions into four models (Figure 4) that share the same normal walk so that each model alone is able to generate a normal sequence. When a user applies a push to the character interactively, a model is chosen according to the force direction. If the chosen model is different from the one being used, the last three poses are used as initial states for the new model to synthesize subsequent responses. The new model then takes over until another push triggers a different model. With 14 examples, we can generate a rich variety of reactions to various pushes without any visual discrepancy at the model-switching moment (Figure 5).

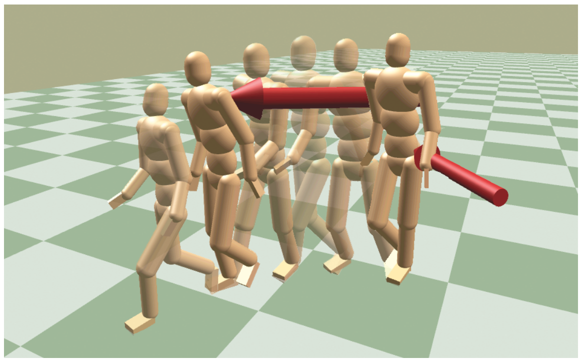


Figure 5: *Continuous responses to two consecutive pushes.*

Time performance. Although offline learning is relatively efficient, the online synthesis runs at only one frame per second on a 2.8GHz Intel processor due to our naive computation of the large kernel matrix. To speed up the computation, many existing acceleration techniques such as active set methods or sparse approximation methods [LSH03, SG06] can be applied. With a more efficient implementation based on these acceleration algorithms, we expect our method to run at interactive rates.

5. Discussion and conclusion

In this section, we discuss some design choices and limitations of our method.

Data variability. In all our tests, we used four motions to learn a 3*D* latent space with second-order dynamics. We chose motions that could be intuitively interpolated into a model. For example, including a forward push and a side push is more likely to produce a meaningful interpolation than including a push at the left-leg support and one at the right-leg support. Our method can include a large number and variety of examples. However, the computation time will increase significantly as the model requires higher dimensional latent space and more costly computation of the covariance matrix in the Gaussian process. We found four motions with around 300 frames a good compromise between model capability and computation efficiency. Instead of increasing the number of input motions and the latent space dimensions, we propose a simple model-switching scheme to address this issue. We demonstrate that with 14 examples grouped into four models, we can synthesize a virtual character reacting to arbitrary forces without extra runtime cost compared to using a single model. While our simple experimental scheme works very well, we plan to explore more sophisticated clustering and indexing schemes in the future.

Model generalizability. Our model produces new poses by interpolation and extrapolation of input data based on the learned probability function. High-likelihood regions in the model correspond to poses that are similar to the input and thus permit both high-quality interpolation and extrapolation, and the quality deteriorates as the likelihood decreases in regions with sparse data. When a deformation that does not lie on the latent space occurs due to perturbations, the subsequent prediction often uses large torques to eliminate these deformations, resulting in visually stiff motions. In spite of this inherent limitation of data-driven approaches, our algorithm provides greater generalizability than previous methods by fully exploring the capacity of the available data, as our results show that three examples can already generate a visually rich set of responsive motions. Encouraged by the success in walking motions, we plan to apply our method on different walking styles and even different types of motions such as running or climbing because our method makes no assumptions about the input. As a base motion and specific variations are supplied for training, our method can be applied.

Model verification. Although our method can deal with noisy input and missing information, we require that at least one of the perturbed examples provides transition dynamics to normal walking. In addition, we require that all models share a normal walk motion so that switching between them is seamless during normal motions. To allow perturbations in the middle of recovery would require models also sharing perturbed examples. Otherwise, the transition point will score low likelihood in the newly chosen model. We currently rely on manual testing to evaluate whether a model works as expected. We would like to use our method for guiding data gathering in areas where examples are sparse and to come up with systematic methods for evaluating the quality of a database.

In conclusion, we present a fully automatic method that generalizes a small number of examples by combining a latent variable model and a physics-based method. Given only three similar perturbed motions, we can generate a variety of realistic responses to perturbations that are not presented in the training data. We also experimented with a simple model-switching scheme that works with a larger dataset. With a database of 15 motions, we can generate walking motions responsive to arbitrary perturbations, which employ realistic human balancing strategies such as changing step size or timing. We are interested in applying our method to create transitions among a motion database. Using the latent space model and the dynamic model as both a quality metric and a motion synthesis mechanism, we can make use of small databases without any manual processing effort. However, to work with a large data set efficiently, we need to explore automatic methods for clustering data into local models.

Appendix A: Implementation details

We describe our exact formulations in this section for completeness. We refer the readers to Lawrence [Law06] and Wang *et al.* [WFH08] for detailed explanations of GPLVM and GPDM, respectively.

SGPLVM with back constraints. We denote the feature vectors as $\mathbf{Y}_{N\times D}$ and the latent vectors as $\mathbf{X}_{N\times d}$. The hyperparameters of the model include a diagonal scaling matrix $\mathbf{W}_{D\times D} = diag(w_1, \cdots, w_D)$, white noise variance β, and forward kernel parameters α and γ. With back constraining, latent positions become linear combinations of backward kernels computed from data points. The weight matrix $\mathbf{A}_{N\times d}$ and backward kernel parameters α' and γ' are solved to get $\mathbf{X}$. We place a Gaussian prior on the latent positions and a uniform prior over the hyperparameters. L_P is then formulated as

$$\begin{aligned} L_P &= \frac{D}{2}\ln|\mathbf{K}| - N\ln|\mathbf{W}| + \frac{1}{2}\mathrm{tr}(\mathbf{K}^{-1}\mathbf{Y}\mathbf{W}^2\mathbf{Y}^T) \\ &\quad + \frac{1}{2}\mathrm{tr}(\mathbf{X}\mathbf{X}^T) + \ln\beta\alpha\gamma\alpha'\gamma', \\ \mathbf{X} &= \mathbf{K}'\mathbf{A}. \end{aligned}$$

Radial basis functions (RBF) are used in both forward kernel $\mathbf{K}_{N\times N}$ and backward kernel $\mathbf{K}'_{N\times N}$. Their entries are

$$k(\mathbf{x}_i, \mathbf{x}_j) = \alpha\exp(-\frac{\gamma}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2) + \delta_{\mathbf{x}_i,\mathbf{x}_j}\beta^{-1},$$

$$k'(\mathbf{y}_i, \mathbf{y}_j) = \alpha'\exp(-\frac{\gamma'}{2}\|\mathbf{y}_i - \mathbf{y}_j\|^2),$$

where δ is the Dirac delta function. Latent trajectories are initialized as the first three principle components of the features. Hyperparameters are initialized as $\beta = 0.1, \alpha = \gamma = 1.0$, and $\alpha' = \gamma' = 2.0$.

Dynamic model. Given M latent trajectories $\mathbf{X}_{1:M}$, we solve for a second-order dynamic model in the latent space as follows:

$$\begin{aligned} L_D &= \frac{d}{2}\ln|\mathbf{K}_D| + \frac{1}{2}\mathrm{tr}(\mathbf{K}_D^{-1}\mathbf{X}_{out}\mathbf{X}_{out}^T) + \ln\beta_D\alpha_D\gamma_1\gamma_2, \\ \mathbf{X}_{out} &= [X_{3:n_1}^{(1)\,T}, \cdots, X_{3:n_M}^{(M)\,T}]^T, \end{aligned}$$

where β_D is the white noise variance, and α_D, γ_1, and γ_2 are parameters of kernel $\mathbf{K}_D$. An entry in $\mathbf{K}_D$ is computed from a pair of latent positions in $\mathbf{X}_{in} = [X_{1:n_1-2}^{(1)\,T}, \cdots, X_{1:n_M-2}^{(M)\,T}]^T$:

$$\begin{aligned} &k_D((\mathbf{x}_{i-1}, \mathbf{x}_i), (\mathbf{x}_{j-1}, \mathbf{x}_j)) \\ &= \alpha_D\exp(-\frac{\gamma_1}{2}\|\mathbf{x}_{i-1} - \mathbf{x}_{j-1}\|^2 - \frac{\gamma_2}{2}\|\mathbf{x}_i - \mathbf{x}_j\|^2) + \delta\beta_D^{-1}. \end{aligned}$$

Hyperparameters are initialized as $\alpha_D = \beta_D = 1.0$.

Inference. We use three previous frames to infer a new pose. The likelihood of a sequence with three feature vectors $\hat{Y}(\mathbf{q}_{n+1})$ and the corresponding latent positions $\hat{X}(\mathbf{x}_{n+1})$ is then computed as

$$\begin{aligned} L_S(\hat{Y}, \hat{X}) &= \frac{1}{2}\ln|\mathbf{K}(\hat{X})| + \frac{1}{2}\mathrm{tr}(\mathbf{K}^{-1}(\hat{X})\mathbf{Z}_Y\mathbf{W}^2\mathbf{Z}_Y^T) \\ &\quad + \frac{1}{2}\ln|\mathbf{K}_D(\hat{X}_{in})| + \frac{1}{2}\mathrm{tr}(\mathbf{K}_D^{-1}(\hat{X}_{in})\mathbf{Z}_X\mathbf{Z}_X^T), \end{aligned}$$

where

$$\begin{aligned} \hat{Y} &= [\mathbf{y}_{n-1}, \mathbf{y}_n, \mathbf{y}_{n+1}(\mathbf{q}_{n+1})]^T, \hat{X} = [\mathbf{x}_{n-1}, \mathbf{x}_n, \mathbf{x}_{n+1}]^T, \\ \mathbf{Z}_Y &= \hat{Y} - \mathbf{K}_{\hat{X},\mathbf{X}}\mathbf{K}^{-1}\mathbf{Y}, \\ \mathbf{K}(\hat{X}) &= \mathbf{K}_{\hat{X},\hat{X}} - \mathbf{K}_{\hat{X},\mathbf{X}}\mathbf{K}^{-1}\mathbf{K}_{\hat{X},\mathbf{X}}^T, \\ \hat{X}_{in} &= [\mathbf{x}_{n-1}, \mathbf{x}_n]^T, \hat{X}_{out} = \mathbf{x}_{n+1}^T, \\ \mathbf{Z}_X &= \hat{X}_{out} - \mathbf{K}_{\hat{X}_{in},\mathbf{X}_{in}}\mathbf{K}_D^{-1}\mathbf{X}_{out}, \\ \mathbf{K}_D(\hat{X}_{in}) &= \mathbf{K}_{\hat{X}_{in},\hat{X}_{in}} - \mathbf{K}_{\hat{X}_{in},\mathbf{X}_{in}}\mathbf{K}_D^{-1}\mathbf{K}_{\hat{X}_{in},\mathbf{X}_{in}}^T. \end{aligned}$$

References

[AdSP07] Abe Y., da Silva M., Popović J.: Multi-objective control with frictional contacts. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2007), pp. 249–258.

[AFO05] Arikan O., Forsyth D. A., O'Brien J. F.: Pushing people around. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2005), pp. 59–66.

[BH00] Brand M., Hertzmann A.: Style machines. In *SIGGRAPH* (2000), pp. 183–192.

[BP08] Barbič J., Popović J.: Real-time control of physically based simulations using gentle forces. In *SIGGRAPH Asia* (2008), pp. 1–10.

[CBvdP09] Coros S., Beaudoin P., van de Panne M.: Robust task-based control policies for physics-based characters. In *SIGGRAPH Asia* (2009), pp. 1–9.

[CBYvdP08] COROS S., BEAUDOIN P., YIN K. K., VAN DE PANN M.: Synthesis of constrained walking skills. In *SIGGRAPH Asia* (2008), pp. 1–9.

[CH07] CHAI J., HODGINS J. K.: Constraint-based motion optimization using a statistical dynamic model. In *SIGGRAPH* (2007), p. 8.

[dSAP08] DA SILVA M., ABE Y., POPOVIĆ J.: Interactive simulation of stylized human locomotion. In *SIGGRAPH* (2008), pp. 1–10.

[dSDP09] DA SILVA M., DURAND F., POPOVIĆ J.: Linear bellman combination for control of character animation. *ACM Trans. Graph. 28*, 3 (2009), 1–10.

[FvdPT01] FALOUTSOS P., VAN DE PANNE M., TERZOPOULOS D.: Composable controllers for physics-based character animation. In *SIGGRAPH* (2001), pp. 251–260.

[GMHP04] GROCHOW K., MARTIN S. L., HERTZMANN A., POPOVIĆ Z.: Style-based inverse kinematics. In *SIGGRAPH* (2004), pp. 522–531.

[GSM96] GILL P., SAUNDERS M., MURRAY W.: *SNOPT: An SQP Algorithm for Large-scale Constrained Optimization*. Tech. Rep. NA 96-2, University of California, San Diego, 1996.

[IAF09] IKEMOTO L., ARIKAN O., FORSYTH D.: Generalizing motion edits with gaussian processes. *ACM Trans. Graph. 28*, 1 (2009), 1–12.

[JYL09] JAIN S., YE Y., LIU C. K.: Optimization-based interactive motion synthesis. *ACM Trans. Graph. 28*, 1 (2009), 1–12.

[KGP02] KOVAR L., GLEICHER M., PIGHIN F.: Motion graphs. *ACM Trans. Graph. 21*, 3 (2002), 473–482.

[Law04] LAWRENCE N. D.: Gaussian process latent variable models for visualisation of high dimensional data. In *In NIPS* (2004).

[Law06] LAWRENCE N. D.: *The Gaussian process latent variable model*. Tech. Rep. CS-06-03, The University of Sheffield, Department of Computer Science, 2006.

[LBJK09] LAU M., BAR-JOSEPH Z., KUFFNER J.: Modeling spatial and temporal variation in motion data. In *SIGGRAPH Asia* (2009), pp. 1–10.

[LCR*02] LEE J., CHAI J., REITSMA P. S. A., HODGINS J. K., POLLARD N. S.: Interactive control of avatars animated with human motion data. *ACM Trans. Graph. 21*, 3 (2002), 491–500.

[LMFP08] LI L., MCCANN J., FALOUTSOS C., POLLARD N.: Laziness is a virtue: Motion stitching using effort minimization. In *Short Papers Proceedings of EUROGRAPHICS* (2008).

[LQC06] LAWRENCE N. D., QUIÑONERO-CANDELA J.: Local distance preservation in the gp-lvm through back constraints. In *ICML* (2006), pp. 513–520.

[LSH03] LAWRENCE N., SEEGER M., HERBRICH R.: Fast sparse gaussian process methods: The informative vector machine. In *Advances in Neural Information Processing Systems 15* (2003), pp. 609–616.

[LWS02] LI Y., WANG T., SHUM H.-Y.: Motion texture: a two-level statistical model for character motion synthesis. In *SIGGRAPH* (2002), pp. 465–472.

[MLD09] MA X., LE B. H., DENG Z.: Style learning and transferring for facial animation editing. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2009), pp. 123–132.

[MLPP09] MUICO U., LEE Y., POPOVIĆ J., POPOVIĆ Z.: Contact-aware nonlinear control of dynamic characters. *ACM Trans. Graph. 28*, 3 (2009), 1–9.

[Møl93] MØLLER M. F.: A scaled conjugate gradient algorithm for fast supervised learning. *Neural Netw. 6*, 4 (1993), 525–533.

[MZS09] MACCHIETTO A., ZORDAN V., SHELTON C. R.: Momentum control for balance. *ACM Trans. Graph. 28*, 3 (2009), 1–8.

[Nat] NATURALMOTION: euphoria.

[PZ05] POLLARD N. S., ZORDAN V. B.: Physically based grasping control from example. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2005), pp. 311–318.

[SCCH09] SHIRATORI T., COLEY B., CHAM R., HODGINS J. K.: Simulating balance recovery responses to trips based on biomechanical principles. In *ACM SIGGRAPH/Eurographics Symposium on Computer Animation* (2009), pp. 37–46.

[SG06] SNELSON E., GHAHRAMANI Z.: Sparse gaussian processes using pseudo-inputs. In *Advances in Neural Information Processing Systems 18* (2006), MIT press, pp. 1257–1264.

[SHP04] SAFONOVA A., HODGINS J. K., POLLARD N. S.: Synthesizing physically realistic human motion in low-dimensional, behavior-specific spaces. In *SIGGRAPH* (2004), pp. 514–521.

[SKL07] SOK K. W., KIM M., LEE J.: Simulating biped behaviors from human motion data. In *SIGGRAPH* (2007), p. 107.

[SPF03] SHAPIRO A., PIGHIN F., FALOUTSOS P.: Hybrid control for interactive character animation. In *Pacific Graphics* (2003), p. 455.

[TLP06] TREUILLE A., LEWIS A., POPOVIĆ Z.: Model reduction for real-time fluids. *ACM Trans. Graph. 25*, 3 (July 2006), 826–834.

[UFF06] URTASUN R., FLEET D. J., FUA P.: 3d people tracking with gaussian process dynamical models. In *CVPR* (2006), pp. 238–245.

[UFHF05] URTASUN R., FLEET D. J., HERTZMANN A., FUA P.: Priors for people tracking from small training sets. In *ICCV* (2005), pp. 403–410.

[WFH06] WANG J. M., FLEET D. J., HERTZMANN A.: Gaussian process dynamical models. In *Advances in Neural Information Processing Systems 18*. 2006, pp. 1441–1448. Proc. NIPS'05.

[WFH08] WANG J. M., FLEET D. J., HERTZMANN A.: Gaussian process dynamical models for human motion. *IEEE Trans. Pattern Anal. Mach. Intell. 30*, 2 (2008), 283–298.

[WFH09] WANG J. M., FLEET D. J., HERTZMANN A.: Optimizing walking controllers. In *SIGGRAPH Asia* (2009), pp. 1–8.

[YCBvdP08] YIN K., COROS S., BEAUDOIN P., VAN DE PANNE M.: Continuation methods for adapting simulated skills. In *SIGGRAPH* (2008), pp. 1–7.

[YCP03] YIN K., CLINE M. B., PAI D. K.: Motion perturbation based on simple neuromotor control models. In *Pacific Graphics* (2003), p. 445.

[YL08] YE Y., LIU C. K.: Animating responsive characters with dynamic constraints in near-unactuated coordinates. In *SIGGRAPH Asia* (2008), pp. 1–5.

[YLvdP07] YIN K., LOKEN K., VAN DE PANNE M.: Simbicon: simple biped locomotion control. In *SIGGRAPH* (2007), p. 105.

[YPvdP05] YIN K., PAI D. K., VAN DE PANNE M.: Data-driven interactive balancing behaviors. In *Pacific Graphics* (Oct. 2005).

[ZH02] ZORDAN V. B., HODGINS J. K.: Motion capture-driven simulations that hit and react. In *ACM SIGGRAPH/Eurographics symposium on Computer animation* (2002), pp. 89–96.

[ZMCF05] ZORDAN V. B., MAJKOWSKA A., CHIU B., FAST M.: Dynamic response for motion capture animation. In *SIGGRAPH* (2005), pp. 697–701.