Variational Autoencoder with Differentiable Physics Engine for Human Gait Analysis and Synthesis

Naoya Takeishi, Alexandros Kalousis
University of Applied Sciences and Arts Western Switzerland (HES-SO)
Geneva, Switzerland
{naoya.takeishi,alexandros.kalousis}@hesge.ch

Abstract

We address the task of learning generative models of human gait. As gait motion always follows the physical laws, a generative model should also produce outputs that comply with the physical laws, particularly rigid body dynamics with contact and friction. We propose a deep generative model combined with a differentiable physics engine, which outputs physically plausible signals by construction. The proposed model is also equipped with a policy network conditioned on each sample. We show an example of the application of such a model to style transfer of gait.

1 Introduction

Analysis and synthesis of human gait are prevalent issues in biomechanics [see, e.g., 40]. In this work, we aim to address them via learning generative models of human gait motion. Generative models of gait can help us to analyze gait patterns by examining inferred quantities such as latent representations, as well as to synthesize gait patterns with desired properties. They are useful, for example, in clinical decision making for treatment of pathological gait.

One of the challenges in learning generative model of gait is to ensure the physical validity of model's outputs. As motion of gait must always follow the physical laws, a generative model should also yield motions that comply with the physical laws. However, purely data-driven generative models (e.g., ones only with deep neural networks) can often produce physically impossible or implausible motion patterns. A quick remedy is to impose regularization that penalizes violation of the physical laws, but it does not guarantee the compliance of the physical laws outside the training data regime.

We suggest learning deep generative models built from a physics simulator as well as neural networks (see figure 1), so that the outputs comply *by construction* with the physical laws encoded in the simulator. More specifically, we incorporate a differentiable simulator of articulated rigid body dynamics [see, e.g., 7, 5, 9, 10, 12, 14, 17, 27, 11, 28, 39] into the framework of variational autoencoders (VAEs) [22, 29]. We also consider a policy network for controlling the agent in the physics simulator and condition it with the latent representation to provide a sample-dependent control law. We present an application of such hybrid generative models to style transfer of human gait.

Related work Combination of differentiable physics engines and machine learning models such as neural networks have been actively studied recently. Many studies have been done in the context of prediction or system identification [e.g., 7, 31, 19, 25, 8, 20, 32, 33, 37, 38, 6, 21, 41] and control or reinforcement learning [e.g., 7, 31, 16, 15, 34, 6, 18, 42]. Some researchers have investigated generative modeling or related methodologies combined with differentiable physics engines [7, 14, 17, 2, 35]. For example, de A. Belbute-Peres et al. (2018) [7] suggested an autoencoding architecture with a differentiable physics simulator inside. Takeishi and Kalousis (2021) [35] proposed a method to strike a balance between physics models and data-driven models in learning VAEs combined with physics models. Our work is on this track of research but is more focused on gait modeling.

NeurIPS 2021 Workshop on Deep Generative Models and Downstream Applications

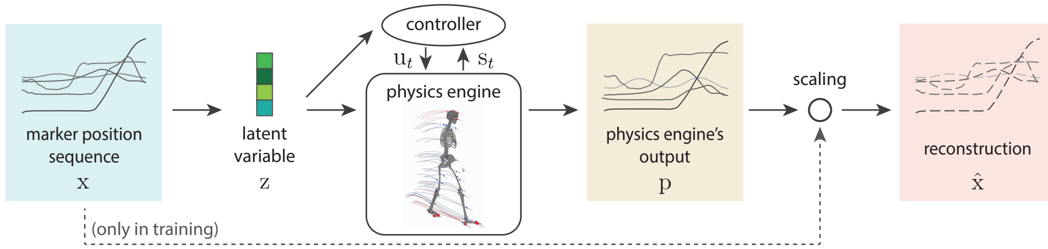


Figure 1: Diagram of the proposed generative model with differentiable physics engine.

Our task combines features of related tasks (see table 1). In human pose tracking (i.e., pose estimation from measurements such as videos and motion capture data) [e.g., 3], consistency with the physical laws is often considered via inverse kinematics (IK), inverse dynamics (ID), and/or physics-informed regularization. Reinforcement learning (RL) is at the intersection of machine learning, control, and physics simulation and has been applied to human motion [e.g., 23]. One of the strengths of VAE and its variants, which have been applied to gait as well [e.g., 4], is the capability of amortized inference, with which latent variables for new observations can be inferred quickly. Our method is also notable in the sense that training of the model including a policy network is done with a single gradient descent loop only with the evidence lower bound (and some regularizers) as objective.

	Physics	Controller	Amortized inference
IK/ID/tracking	✓		
Control/RL	✓	✓	
(V)AE			✓
Ours	✓	✓	✓

Table 1: Features of related tasks.

2 VAE with physics engine for gait

2.1 Target measurements

In this work, we deal with time-series of the three-dimensional position of markers attached to a subject, which can be measured by motion capture systems. If there are m markers, each sequence is $3m$-dimensional multivariate time-series. We suppose we have a collection of such sequences as data and would like to learn a generative model from them. Simultaneous treatment of other signals that are often available in gait analysis, such as ground reaction force and electromyography measurements, is an extension to be addressed in the future.

2.2 Model architecture

The architecture of the proposed method, depicted in figure 1, follows the autoencoding structure. Given a marker position sequence as input data x, encoder networks compute latent variable z. The decoder comprises a differentiable physics engine for rigid body dynamics simulation. An agent in the physics engine is controlled with a neural network (i.e., policy network) conditioned on the latent variable z. The physics engine should return a simulated marker position sequence, denoted by p in figure 1. Finally, we match the scale of p with that of x to give the final reconstruction $\hat{x}$.

Let us formalize the idea. Let $\mathrm{x} \in \mathbb{R}^{\ell \times 3m}$ be an input marker position sequence of length ℓ. The encoder networks, $\boldsymbol{\mu}_{\text{encoder}} : \mathbb{R}^{\ell \times 3m} \to \mathbb{R}^{d_z}$ and $\sigma^2_{\text{encoder}} : \mathbb{R}^{\ell \times 3m} \to \mathbb{R}_{>0}$, give the sufficient statistics of the approximated posterior of the latent variable $\mathrm{z} \in \mathbb{R}^{d_z}$, that is,

$$\mathrm{z} \sim \mathcal{N}(\mathrm{z}; \boldsymbol{\mu}_{\text{encoder}}(\mathrm{x}), \sigma^2_{\text{encoder}}(\mathrm{x})\boldsymbol{I}). \tag{1}$$

Latent variable z is subsequently utilized in two ways. Firstly, we compute from z the initial condition $\mathrm{s}_0 \in \mathbb{R}^{d_s}$ fed into the physics simulator (because x is marker position while s include joint angles):

$$\mathrm{s}_0 = \boldsymbol{f}_{\text{initializer}}(\mathrm{z}). \tag{2}$$

Secondly, we use z as an additional argument of the controller of the simulator's agent, that is,

$$\mathrm{u}_t = \boldsymbol{f}_{\text{controller}}(\mathrm{s}_t, \mathrm{PositionalEncoding}(\mathrm{z}; t)), \tag{3}$$

where $\mathrm{s}_t \in \mathbb{R}^{d_s}$ and $\mathrm{u}_t \in \mathbb{R}^{d_u}$ are the simulator's state and input signal (i.e., action) at time t, respectively. We use the technique called positional encoding to vary z slightly at each time t. Given such a conditioned controller, the physics engine runs the simulation of rigid body dynamics.

The core computation of the physics engine is the temporal transition of the state variable s_t given action u_t following the rigid body dynamics. Namely, for each of $t = 0, \dots, \ell - 1$, it computes

$$\mathrm{s}_{t+1} = \mathrm{TemporalTransition}(\mathrm{s}_t, \mathrm{u}_t). \tag{4}$$

The state variable s_t comprises the generalized position and velocity of the agent, an articulated rigid body. More specifically, in this work, it comprises the position, velocity, orientation, and angular velocity of the floating base of the agent, as well as the angle and angular velocity of the joints of the agent's articulated body. Consequently, the action u_t is generalized force applied to each joint (i.e., torque). We used a 21-degree-of-freedom model of the human musculoskeletal system as the agent, so $d_s = 21 \times 2$ and $d_u = 21 - 6$ as we do not directly control the floating base, which 6 dofs. The physics engine should finally output a simulated position sequence of the markers attached to the agent. We denote such a simulated sequence by $\mathrm{p} \in \mathbb{R}^{\ell \times 3m}$.

At the final stage of the decoding process, we rescale p because the scale of the simulator's agent is fixed[1], while the data may comprise subjects with different scales. To this end, we compute the optimal scaling factors between p and x using differentiable convex optimization [1]. Let $\mathrm{p}_{t,i,j}$ be the element of p corresponding to the position of the j-th marker along the i-axis at timestep t, for $t = 1, \dots, \ell$, $i = X, Y, Z$, and $j = 1, \dots, m$. Let $\hat{\mathrm{x}} \in \mathbb{R}^{\ell \times 3m}$ be the final reconstruction after scaling, and let $\hat{\mathrm{x}}_{t,i,j}$ be the element of $\hat{\mathrm{x}}$ analogously to the case of $\mathrm{p}_{t,i,j}$. Furthermore, let $\mathrm{p}_{t,i,\mathrm{fb}}$ denote the element of p corresponding to the floating base position along the i-axis at timestep t. With these notions, the final reconstruction $\hat{\mathrm{x}}$ is given by the following linear scaling of p:

$$\mathbb{E}[\hat{\mathrm{x}}_{t,i,j}] = \alpha_i(\mathrm{p}_{t,i,j} - \mathrm{p}_{t,i,\mathrm{fb}}) + \beta_i \mathrm{p}_{t,i,\mathrm{fb}} + \gamma_i, \tag{5}$$

where $\phi := \{\alpha_i, \beta_i, \gamma_i\}$ is the set of scaling factors. They are computed via the following problem:

$$\min_{\alpha_i,\beta_i,\gamma_i} \quad \sum_{t=1}^{\ell} \sum_{j=1}^{m} |\mathbb{E}[\hat{\mathrm{x}}_{t,i,j}] - \mathrm{x}_{t,i,j}|^2 \quad \text{s.t.} \quad \alpha_i \in [\alpha_{\mathrm{lb}}, \alpha_{\mathrm{ub}}], \beta_i \in [\beta_{\mathrm{lb}}, \beta_{\mathrm{ub}}], \gamma_i \in [\gamma_{\mathrm{lb}}, \gamma_{\mathrm{ub}}], \tag{6}$$

for $i = X, Y, Z$. Note that this convex optimization layer takes the original input x unlike ordinary autoencoder structures. It is not problematic because it needs x only in training, and in a test phase, we can use arbitrary scaling factors for generating $\hat{\mathrm{x}}$ (e.g., ones computed with some reference datapoint).

2.3 Learning

Learning is done by maximizing the evidence lower bound (ELBO) of the marginal log likelihood [see, e.g., 22], that is, $\mathbb{E}_{\mathrm{z} \sim p_{\mathrm{encoder}}}[\log p_{\mathrm{decoder}}(\mathrm{x} \mid \mathrm{z})] - D_{\mathrm{KL}}(p_{\mathrm{encoder}}(\mathrm{z}) \parallel p_{\mathrm{prior}}(\mathrm{z}))$, where p_{encoder} is the distribution in equation 1, p_{decoder} is a distribution such that the first moment is given by equation 5, and p_{prior} is some prior distribution of z. To reduce the variance, we use the path derivative ELBO [30]. Note that the policy network is also trained within this scheme altogether.

In addition to the evidence lower bound, we take several regularization terms into account. First, we penalize the magnitude of the power by the agent's action, i.e., the product of torque and angular velocity. We also penalize the first-order and second-order differences (along time) of u_t to prevent implausible torque sequences. In the case of conditional modeling, which will be introduced later in section 2.4, we consider independence-enforcing regularization for disentanglement of the latent variable. We empirically found that when training the proposed model, it was essential to start training from short sequences and then feed longer sequences gradually.

2.4 Incorporating conditional variable for style transfer

We present an extension of the model in section 2.2. We focus on the application of style transfer; given some x and some features c describing a "style" of the gait, we would like to generate new gait motion x' having altered gait style $c' \neq c$. To this end, we incorporate a conditional variable c into the model. It particularly appears in the encoder part of the model; instead of computing z directly from x, we first compute some intermediate quantity $\mathrm{y} \in R^{d_y}$ from x and c and then compute z from y and c′, where c′ has the same value with c in training but has an arbitrary value in test.

[1] It is also possible to optimize agent's scale during training or infer it in an amortized manner, but sample-dependent physical property of an agent might cause issues in the numerical stability of simulation (e.g., tuning simulator's setting may become difficult). Such an approach should be explored in future studies.

During training, equation 1 is to be replaced by

$$\mathrm{y} = \boldsymbol{g}(\mathrm{x}, \mathrm{c}) \quad \text{and} \tag{7}$$

$$\mathrm{z} \sim \mathcal{N}(\mathrm{z}; \tilde{\boldsymbol{\mu}}_{\mathrm{encoder}}(\mathrm{y}, \mathrm{c}), \tilde{\sigma}^2_{\mathrm{encoder}}(\mathrm{y}, \mathrm{c})\boldsymbol{I}), \tag{8}$$

where $\boldsymbol{g}$, $\tilde{\boldsymbol{\mu}}_{\mathrm{encoder}}$, and $\tilde{\sigma}^2_{\mathrm{encoder}}$ are neural networks. In this two-stage computation, the intermediate variable y should capture *the part of the information of* x *that is not described by* c. Then, z is again encoded with c so that z retains all the information from x and c. In other words, y should be a representation of x disentangled from c. Such semantics of y and z are not automatically obtained by simply maximizing the evidence lower bound. We ensure the independence between c and y and the dependence between z and c by imposing regularization on the Hilbert–Schmidt independence criterion (HSIC) [13]. It is known that HSIC between two random variables becomes zero if and only if the variables are statistically independent [13], and when the variables are dependent, HSIC takes a positive value. We minimize the following quantity as regularizer: $\lambda_1 \widehat{\mathrm{HSIC}}(\mathrm{c}, \mathrm{y}) - \lambda_2 \widehat{\mathrm{HSIC}}(\mathrm{c}, \mathrm{z})$, where λ_1 and λ_2 are hyperparameters, and $\widehat{\mathrm{HSIC}}$ means an empirical estimation of HSIC. We used the Gaussian kernel with width determined by the median trick. We note that regularization of VAEs with HSIC was also studied in [26, 36].

In a test phase, we perform another branch of computation from

$$\mathrm{z}' \sim \mathcal{N}(\mathrm{z}'; \tilde{\boldsymbol{\mu}}_{\mathrm{encoder}}(\mathrm{y}, \mathrm{c}'), \tilde{\sigma}^2_{\mathrm{encoder}}(\mathrm{y}, \mathrm{c}')\boldsymbol{I}), \tag{8$'$}$$

where c′ may have a value different from that of c. If y is successfully disentangled from c, this new z′ informed by c′ should attain information of c′ (and not of c), which would enable style transfer of x into some $\hat{\mathrm{x}}'$ having the property of c′. Here, we assume that the condition variable c (or c′) does not contain the scale of the subject. Such an assumption enables us to use the scaling factor ϕ computed with the original c even for $\hat{\mathrm{x}}'$. In figure 2, we show the computation flow of the model with the original or altered conditional variable. During training, only the upper part of figure 2 with the original c is run. The lower part of figure 2 with the altered c′ works in applying style transfer.

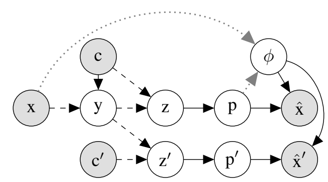


Figure 2: Diagram of the proposed model with conditional variable for style transfer.

3 Preliminary experiment

3.1 Configuration

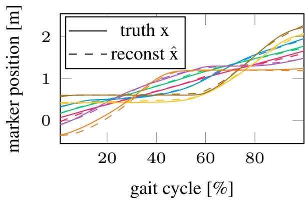
Dataset We used a public dataset of human locomotion [24]. We divided the 50 subjects of the dataset into training, validation, and test sets. From the original dataset, we extracted the data of marker position measurements during walking and used them as x. We had $n = 328$, 34, and 97 sequences for the training, validation, and test sets, respectively. Each sequence contains one gait cycle (i.e., from a heel strike to the next heel strike). We aligned the length of all the sequences to be $\ell = 500$ with cubic interpolation. We also used the information of the gait cadence as the conditional variable c, which varied from 40 to 170 [step/min] within the dataset.

Model As a differentiable physics engine in the proposed model, we used `nimblephysics` [39] library. Other parts of the model were neural networks. $\boldsymbol{g}$ comprised a multilayer perceptron (MLP) for computing features from x and c, a self-attention layer, and an average-pooling (along time) layer. $\tilde{\boldsymbol{\mu}}_{\mathrm{encoder}}$, $\boldsymbol{f}_{\mathrm{initializer}}$, and $\boldsymbol{f}_{\mathrm{controller}}$ were MLPs.

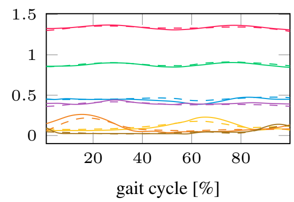
3.2 Result

Figures 3a–3c show an example of the reconstruction by the proposed method, and figure 3d shows the corresponding motion of the agent in the physics simulator. The result on a test sample without style transfer is displayed. It successfully mimics gait motion. For the whole test set, the average reconstruction root-mean-square error was 1.94 ± 0.36 [cm]. The error is relatively large for the later part of the sequence. We again emphasize that the inferred motion, such as one in figure 3, inherently complies with the physical laws up to the fidelity of the physics simulator.

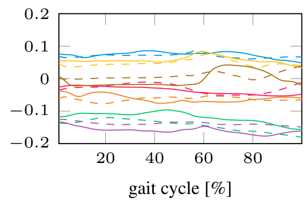
We show an example of the style transfer, where we tried to change the cadence of gait. We randomly picked 20 test samples and performed style transfer with the method presented in section 2.4. The



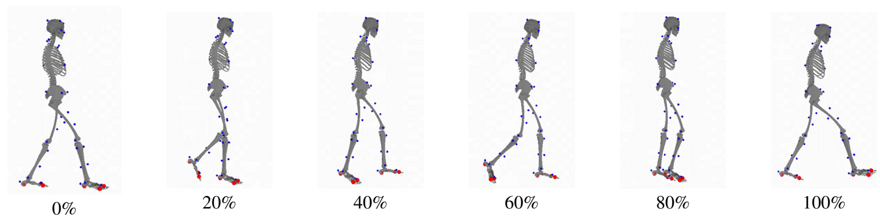
(a) X-axis (+:forward)



(b) Y-axis (+:above)



(c) Z-axis (+:right)



(d) visualization of simulator output

Figure 3: *(a–c)* Example of reconstruction. Only some selected markers are shown; different colors correspond to different markers. *(d)* Corresponding simulator's output at gait cycle from 0% to 100%.

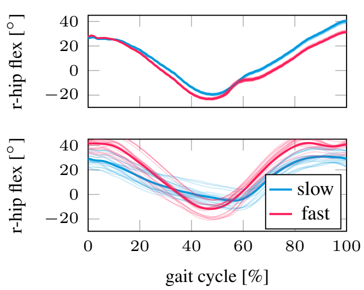


Figure 4: Example of style transfer of gait cadence. In both plots, right hip flexion angle within a gait cycle is displayed. The thick lines are the average of each case. Best viewed in color. *(Top)* Angle inferred in the simulator of the proposed model, with c′ set to be values in slow or fast ranges. *(Bottom)* Angle directly computed from some training samples with inverse kinematics, with slow cadence or fast cadence. In both plots, the minimum angle comes earlier in fast gait than in slow gait. Meanwhile, the generated signals lack the variance.

value of c′ is randomly drawn from a slow cadence range ($40 \leq c' \leq 66$ [step/min]) or a fast cadence range ($144 \leq c' \leq 170$ [step/min]). Figure 4a shows the right hip flexion angle computed by the simulator with the altered c′ given to the model. In figure 4b, for comparison, we show the angle of the same joint inferred with inverse kinematics from some training samples with similar ranges of c. Note that such angles computed with inverse kinematics are never used as input to the proposed model. Comparing the two plots of figure 4, we can find that the generated signals successfully mimic the tendency that the minimum value of the angle comes at earlier gait cycle in fast gait. Meanwhile, the generated signal show relatively small variability (especially at initial condition) partly because the test data contain a much smaller number of subjects than that of training data.

4 Conclusion

We proposed a deep generative model with a differentiable physics engine for modeling human gait, which can produce physically-consistent signals by construction. We presented an example of the application to style transfer of gait cadence. The presented work is preliminary, and we are working on a number of extensions, model variants, and applications. They include the use of some muscle models, style transfer based on physical / biological models instead of unstable disentanglement, and learning on large scale data of both healthy and pathological gait.

Acknowledgments and Disclosure of Funding

This work was supported by the Swiss National Science Foundation Sinergia project *Modeling pathological gait resulting from motor impairments* (CRSII5_177179).

References

[1] A. Agrawal, B. Amos, S. Barratt, S. Boyd, S. Diamond, and J. Z. Kolter. Differentiable convex optimization layers. In *Advances in Neural Information Processing Systems 32*, pages 9562–9574, 2019.

[2] M. A. Aragon-Calvo and J. C. Carvajal. Self-supervised learning with physics-aware neural networks – I. Galaxy model fitting. *Monthly Notices of the Royal Astronomical Society*, 498(3):3713–3719, 2020.

[3] M. A. Brubaker, D. J. Fleet, and A. Hertzmann. Physics-based person tracking using the anthropomorphic walker. *International Journal of Computer Vision*, 87(1-2):140–155, 2010.

[4] J. A. Candido Ramos, L. Blondé, S. Armand, and A. Kalousis. Conditional neural relational inference for interacting systems. In *Machine Learning and Knowledge Discovery in Databases*, number 12979 in Lecture Notes in Computer Science, pages 182–197. 2021.

[5] J. Carpentier and N. Mansard. Analytical derivatives of rigid body dynamics algorithms. In *Robotics: Science and Systems XIV*, 2018.

[6] J. Collins, R. Brown, J. Leitner, and D. Howard. Follow the gradient: Crossing the reality gap using differentiable physics (RealityGrad). arXiv:2109.04674, 2021.

[7] F. de A. Belbute-Peres, K. A. Smith, K. R. Allen, J. B. Tenenbaum, and J. Z. Kolter. End-to-end differentiable physics for learning and control. In *Advances in Neural Information Processing Systems 31*, pages 7178–7189, 2018.

[8] F. de Avila Belbute-Peres, T. D. Economon, and J. Z. Kolter. Combining differentiable PDE solvers and graph neural networks for fluid flow prediction. In *Proceedings of the 37th International Conference on Machine Learning*, pages 2402–2411, 2020.

[9] J. Degrave, M. Hermans, J. Dambre, and F. wyffels. A differentiable physics engine for deep learning in robotics. *Frontiers in Neurorobotics*, 13:6, 2019.

[10] A. Falisse, G. Serrancolí, C. L. Dembia, J. Gillis, and F. De Groote. Algorithmic differentiation improves the computational efficiency of OpenSim-based trajectory optimization of human movement. *PLOS ONE*, 14(10):e0217730, 2019.

[11] C. D. Freeman, E. Frey, A. Raichuk, S. Girgin, I. Mordatch, and O. Bachem. Brax – A differentiable physics engine for large scale rigid body simulation. arXiv:2106.13281, 2021.

[12] M. Geilinger, D. Hahn, J. Zehnder, M. Bächer, B. Thomaszewski, and S. Coros. ADD: Analytically differentiable dynamics for multi-body systems with frictional contact. *ACM Transactions on Graphics*, 39 (6):190, 2020.

[13] A. Gretton, O. Bousquet, A. Smola, and B. Schölkopf. Measuring statistical dependence with Hilbert-Schmidt norms. In *Algorithmic Learning Theory*, number 3734 in Lecture Notes in Artificial Intelligence, pages 63–77. 2005.

[14] E. Heiden, D. Millard, E. Coumans, Y. Sheng, and G. S. Sukhatme. NeuralSim: Augmenting differentiable simulators with neural networks. arXiv:2011.04217, 2020.

[15] P. Holl, V. Koltun, and N. Thuerey. Learning to control PDEs with differentiable physics. In *Proceedings of the 8th International Conference on Learning Representations*, page 2020, 2020.

[16] Y. Hu, J. Liu, A. Spielberg, J. B. Tenenbaum, W. T. Freeman, J. Wu, D. Rus, and W. Matusik. ChainQueen: A real-time differentiable physical simulator for soft robotics. In *Proceedings of the 2019 IEEE International Conference on Robotics and Automation*, pages 6265–6271, 2019.

[17] Y. Hu, L. Anderson, T.-M. Li, Q. Sun, N. Carr, J. Ragan-Kelley, and F. Durand. DiffTaichi: Differentiable programming for physical simulation. In *Proceedings of the 8th International Conference on Learning Representations*, 2020.

[18] Z. Huang, Y. Hu, T. Du, S. Zhou, H. Su, J. B. Tenenbaum, and C. Gan. PlasticineLab: A soft-body manipulation benchmark with differentiable physics. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[19] J. Ingraham, A. Riesselman, C. Sander, and D. Marks. Learning protein structure with a differentiable simulator. In *Proceedings of the 7th International Conference on Learning Representations*, 2019.

[20] M. Jaques, M. Burke, and T. Hospedales. Physics-as-inverse-graphics: Unsupervised physical parameter estimation from video. In *Proceedings of the 8th International Conference on Learning Representations*, 2020.

[21] K. M. Jatavallabhula, M. Macklin, F. Golemo, V. Voleti, M. Weiss, B. Considine, J. Parent-Lévesque, K. Xie, L. Paull, F. Shkurti, D. Nowrouzezahrai, and S. Fidler. ∇Sim: Differentiable simulation for system identification and visuomotor control. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[22] D. P. Kingma and M. Welling. Auto-encoding variational Bayes. In *Proceedings of the 2nd International Conference on Learning Representations*, 2014.

[23] S. Lee, M. Park, K. Lee, and J. Lee. Scalable muscle-actuated human simulation and control. *ACM Transactions on Graphics*, 38(4):73, 2019.

[24] T. Lencioni, I. Carpinella, M. Rabuffetti, A. Marzegan, and M. Ferrarin. Human kinematic, kinetic and EMG data during different walking and stair ascending and descending tasks. *Scientific Data*, 6(1):309, 2019.

[25] J. Liang, M. Lin, and V. Koltun. Differentiable cloth simulation for inverse problems. In *Advances in Neural Information Processing Systems 32*, pages 771–780, 2019.

[26] R. Lopez, J. Regier, M. I. Jordan, and N. Yosef. Information constraints on auto-encoding variational Bayes. In *Advances in Neural Information Processing Systems 31*, pages 6117–6128, 2018.

[27] Y.-L. Qiao, J. Liang, V. Koltun, and M. C. Lin. Scalable differentiable physics for learning and control. In *Proceedings of the 37th International Conference on Machine Learning*, pages 7847–7856, 2020.

[28] Y.-L. Qiao, J. Liang, V. Koltun, and M. C. Lin. Efficient differentiable simulation of articulated bodies. In *Proceedings of the 38th International Conference on Machine Learning*, pages 8661–8671, 2021.

[29] D. J. Rezende, S. Mohamed, and D. Wierstra. Stochastic backpropagation and approximate inference in deep generative models. In *Proceedings of the 31st International Conference on Machine Learning*, pages 1278–1286, 2014.

[30] G. Roeder, Y. Wu, and D. Duvenaud. Sticking the landing: simple, lower-variance gradient estimators for variational inference. In *Advances in Neural Information Processing Systems 30*, pages 6928–6937, 2017.

[31] C. Schenck and D. Fox. SPNets: Differentiable fluid dynamics for deep neural networks. In *Proceedings of the 2nd Conference on Robot Learning*, pages 317–335, 2018.

[32] S. S. Schoenholz and E. D. Cubuk. JAX MD: A framework for differentiable physics. In *Advances in Neural Information Processing Systems 33*, pages 11428–11441, 2020.

[33] C. Song and A. Boularias. Identifying mechanical models through differentiable simulations. In *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, pages 749–760, 2020.

[34] C. Song and A. Boularias. Learning to slide unknown objects with differentiable physics simulations. In *Robotics: Science and Systems XVI*, 2020.

[35] N. Takeishi and A. Kalousis. Physics-integrated variational autoencoders for robust and interpretable generative modeling. arXiv:2102.13156, 2021.

[36] N. Takeishi and Y. Kawahara. Knowledge-based regularization in generative modeling. In *Proceedings of the 29th International Joint Conference on Artificial Intelligence*, pages 2390–2396, 2020.

[37] K. Um, R. Brand, Y. R. Fei, P. Holl, and N. Therey. Solver-in-the-Loop: Learning from differentiable physics to interact with iterative PDE-Solvers. In *Advances in Neural Information Processing Systems 33*, pages 6111–6122, 2020.

[38] K. Wang, M. Aanjaneya, and K. Bekris. A first principles approach for data-efficient system identification of spring-rod systems via differentiable physics engines. In *Proceedings of the 2nd Conference on Learning for Dynamics and Control*, pages 651–665, 2020.

[39] K. Werling, D. Omens, J. Lee, I. Exarchos, and C. K. Liu. Fast and feature-complete differentiable physics for articulated rigid bodies with contact. arXiv:2103.16021, 2021.

[40] D. A. Winter. *Biomechanics and Motor Control of Human Movement*. John Wiley & Sons, Ltd., 4th edition, 2009.

[41] Y. Yin, V. Le Guen, J. Dona, E. de Bézenac, I. Ayed, N. Thome, and P. Gallinari. Augmenting physical models with deep networks for complex dynamics forecasting. In *Proceedings of the 9th International Conference on Learning Representations*, 2021.

[42] M. Zamora, M. Peychev, S. Ha, M. Vechev, and S. Coros. PODS: Policy optimization via differentiable simulation. In *Proceedings of the 38th International Conference on Machine Learning*, pages 7805–7817, 2021.