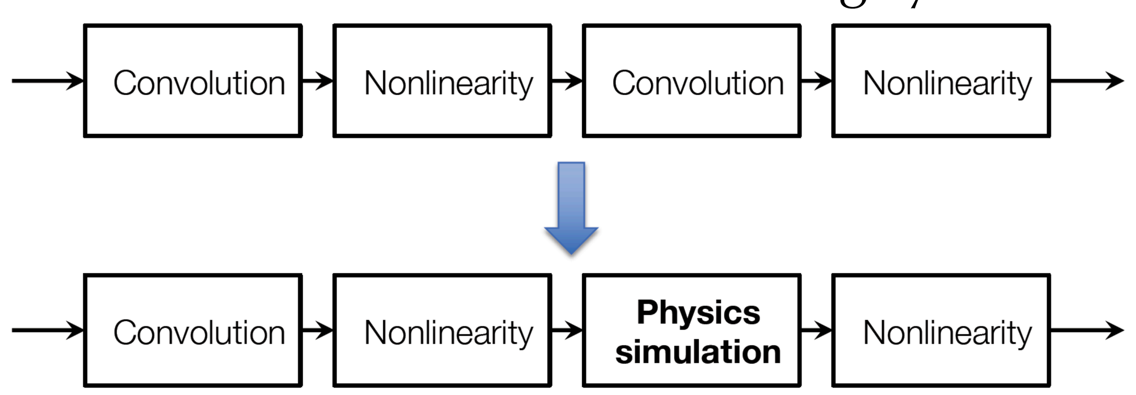
End-to-End Differentiable Physics for Learning and Control

Filipe de Avila Belbute-Peres[1], Kevin Smith[2], Kelsey Allen[2], Josh Tenenbaum[2], J. Zico Kolter[1,3]

1. Carnegie Mellon University 2. Massachusetts Institute of Technology 3. Bosch Center for AI

Introduction & Motivation

- Simulation-based models have strong physical knowledge and learn efficiently, but are inflexible; learned models are flexible, but require extensive (re)training and predictive precision decays fast
- We develop a differentiable physics engine, which has both precise knowledge of physics and can be embedded in an end-to-end learning system



- Previous similar work have either used numerical differentiation methods or relied solely on auto-differentiation. We propose finding the analytic gradients by differentiating the dynamics equations
- This system contributes to a recent trend of incorporating components with structured modules into end-to-end learning systems such as deep networks.

Dynamics LCP

- Rigid body dynamics can be framed as a linear complementarity problem (LCP)
- Newtonian dynamics represented in terms of velocity, using a discrete time step

$$\mathcal{M}\dot{v} = f^{(c)} + f \quad \Rightarrow \quad \mathcal{M}(v_{t+dt} - v_t) = dt f_t^{(c)} + dt f_t$$

- Constraints are added to enforce rigid body dynamics

$\mathcal{J}_e \lambda_e = 0$ Equality constraints

$(\lambda_c, \mathcal{J}_c v + c) \in \mathcal{C}$ Contact constraints

$(\lambda_f, \mathcal{J}_f v + E\gamma) \in \mathcal{C}$, $(\mu\lambda_c - E^T\lambda_f, \gamma) \in \mathcal{C}$ Friction constraints

where $\mathcal{C}(a,b) = \{a \geq 0, b \geq 0, a^T b = 0\}$

- Equality constraints define joints, contact constraints prevent penetrations and friction constraints define frictional forces
- Contact and friction constraints are inequalities and also have a complementarity term ($a^T b = 0$), which characterizes the LCP formulation
- Solvable via primal-dual interior point method

Differentiable physics

- Optimality conditions for LCP can be written compactly

$$\mathcal{M}x + A^T y + G^T z + q = 0$$
$$Ax = 0$$
$$Gx + Fz + s = m$$
$$s \geq 0,\ z \geq 0,\ s^T z \geq 0.$$

where: $x := -v_{t+dt}$, $y := \lambda_e$, $z := \begin{bmatrix}\lambda_c \\ \lambda_f \\ \gamma\end{bmatrix}$, $q := -\mathcal{M}v_t - dt f_t$, $A := \mathcal{J}_e$, $G := \begin{bmatrix}\mathcal{J}_c & 0 \\ \mathcal{J}_f & 0 \\ 0 & 0\end{bmatrix}$, $s := \begin{bmatrix}a \\ \sigma \\ \zeta\end{bmatrix}$, $m := \begin{bmatrix}c \\ 0 \\ 0\end{bmatrix}$, $F := \begin{bmatrix}0 & 0 & 0 \\ 0 & 0 & E \\ \mu & -E^T & 0\end{bmatrix}$

- Using matrix differential calculus, we now take the differentials of the system above

$$\mathrm{d}\mathcal{M}x + \mathcal{M}\mathrm{d}x + \mathrm{d}A^T y + A^T\mathrm{d}y + \mathrm{d}G^T z + G^T\mathrm{d}z + \mathrm{d}q = 0$$
$$\mathrm{d}Ax + A\mathrm{d}x = 0$$
$$\mathrm{d}z \circ (Gx + Fz - m) + z \circ (\mathrm{d}Gx + G\mathrm{d}x + \mathrm{d}Fz + F\mathrm{d}z - \mathrm{d}m) = 0$$

or in matrix form:

$$\begin{bmatrix}\mathcal{M} & G^T & A^T \\ D(z^\star)G & D(Gx^\star + Fz^\star - m) + F & 0 \\ A & 0 & 0\end{bmatrix}\begin{bmatrix}\mathrm{d}x \\ \mathrm{d}z \\ \mathrm{d}y\end{bmatrix} = \begin{bmatrix}-\mathrm{d}\mathcal{M}x^\star - \mathrm{d}A^T y^\star - \mathrm{d}G^T z^\star - \mathrm{d}q \\ -D(z^\star)\mathrm{d}Gx^\star - D(z^\star)\mathrm{d}Fz^\star + D(z^\star)\mathrm{d}m \\ -\mathrm{d}Ax^\star\end{bmatrix}$$

- This system is linear in the unknowns (dx, dy, dz), simple to solve for desired differentials
- By defining

$$\begin{bmatrix}d_x \\ d_z \\ d_y\end{bmatrix} := \begin{bmatrix}\mathcal{M} & G^T & A^T \\ D(z^\star)G & D(Gx^\star + Fz^\star - m) + F & 0 \\ A & 0 & 0\end{bmatrix}^{-T}\begin{bmatrix}\left(\frac{\partial \ell}{\partial x^\star}\right)^T \\ 0 \\ 0\end{bmatrix}$$

we can derive the gradients

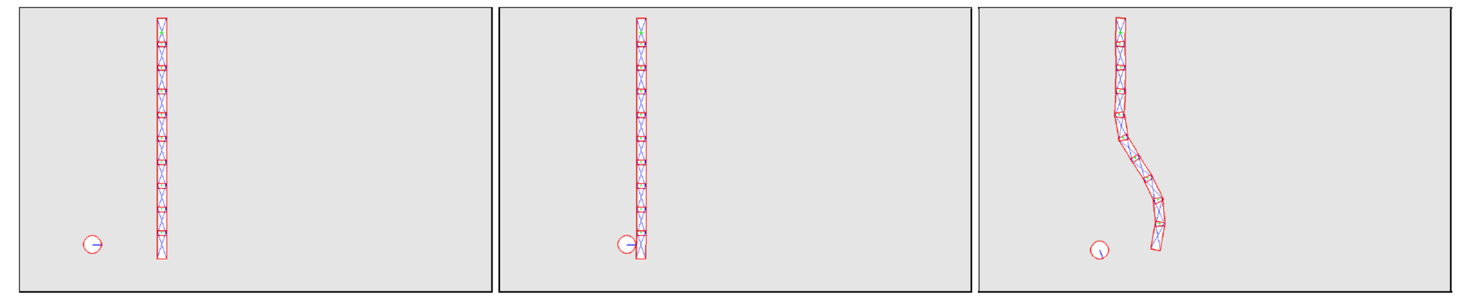
$$\frac{\partial \ell}{\partial q} = -d_x \qquad \frac{\partial \ell}{\partial \mathcal{M}} = -\frac{1}{2}(d_x x^T + x d_x^T)$$
$$\frac{\partial \ell}{\partial m} = D(z^\star)d_z \qquad \frac{\partial \ell}{\partial G} = -D(z^\star)(d_z x^T + z d_x^T)$$
$$\frac{\partial \ell}{\partial A} = -d_y x^T - y d_x^T \qquad \frac{\partial \ell}{\partial F} = -D(z^\star)d_z z^T$$

- Importantly, since we have already solved the LCP in the forward pass, we can compute the backward pass with just one additional solve based upon the LU-factorization of the LCP matrix

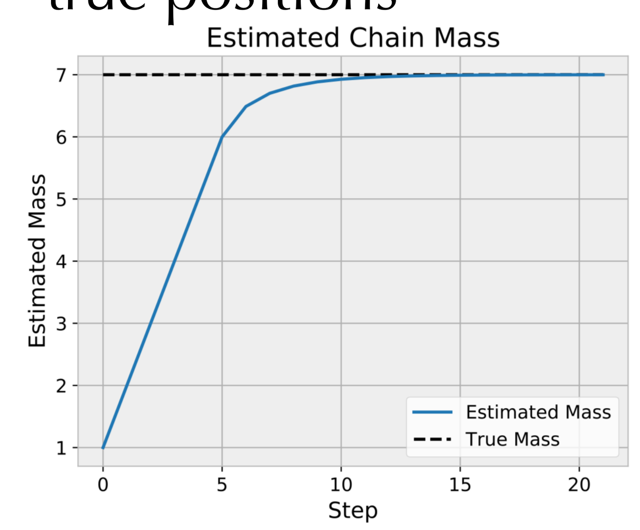
- **We can effectively differentiate through the simulation at no additional cost to just running the simulation itself**

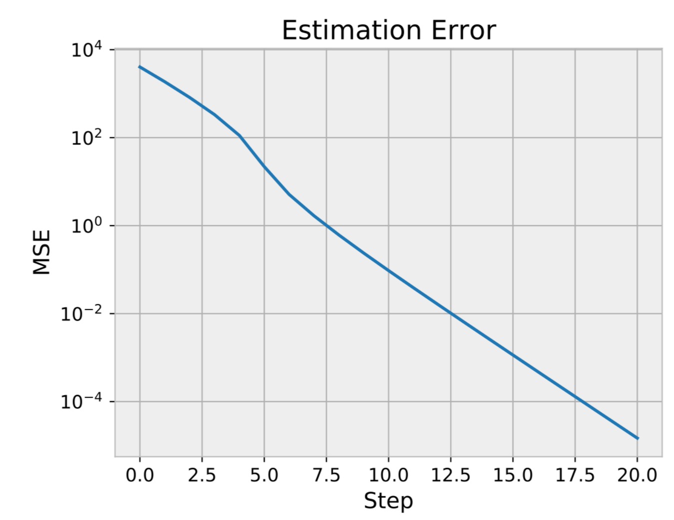
Parameter learning

- A ball of known mass hits chain. Object positions are observed. Task is inferring the chain mass

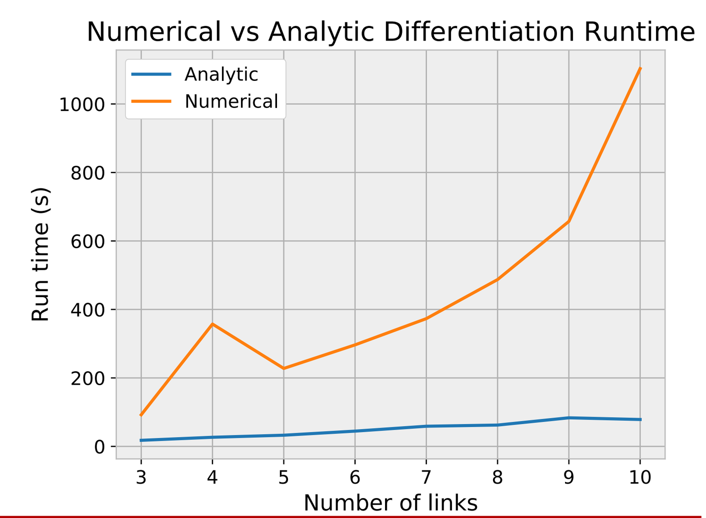


- Simulations are iteratively unrolled starting with an arbitrary mass. Estimated chain mass is minimized using gradient of MSE between the simulated and true positions



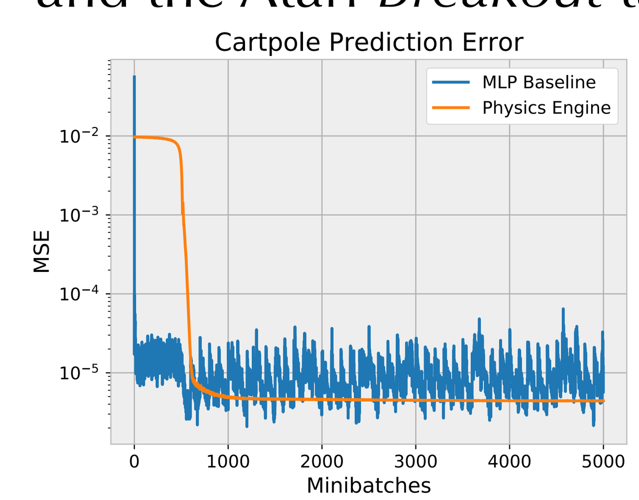


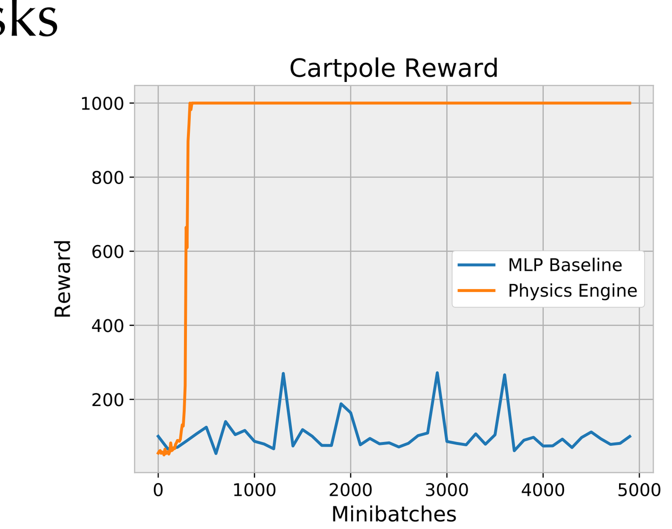
- The engine's analytical gradients are more efficient than numerical gradients (finite differences) as number of parameters increase



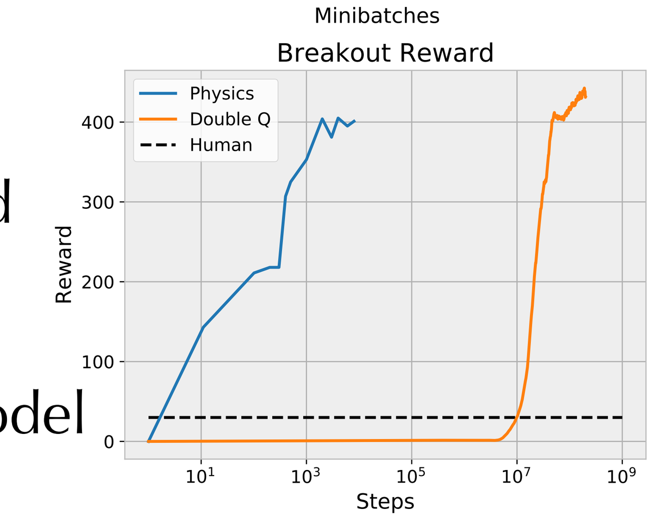
Control

- Since the physics engine is differentiable, we use it in conjunction with iLQR for control in the *Cartpole* and the Atari *Breakout* tasks



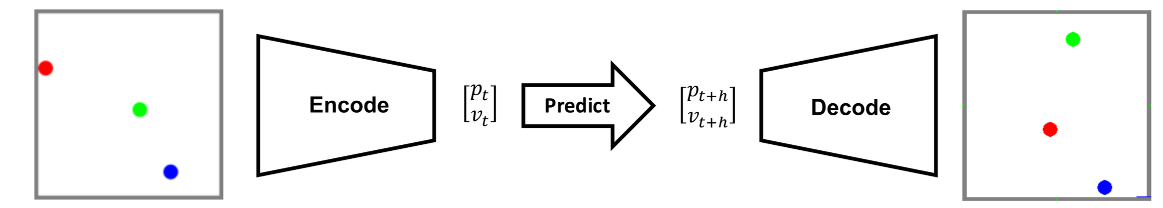


- Parameters are learned from simulations with random policy. Performance is tested as parameters are learned. High reward is achieved before learning a perfect model



Visual prediction

- After observing 3 frames of a billiard ball-like scene, predict positions 10 frames into the future



- Autoencoder architecture. *Encoder* maps frames into physical predictions. *Engine* steps physics into the future. *Decoder* draws image from physics state
- Training is performed with only partially labeled data. For unlabeled examples, prediction uses the estimated parameters (notice the hats):

$$\hat{\phi}_t = encoder(x), \quad \hat{\phi}_{t+dt} = physics(\hat{\phi}_t), \quad \hat{y} = decoder(\hat{\phi}_{t+dt})$$

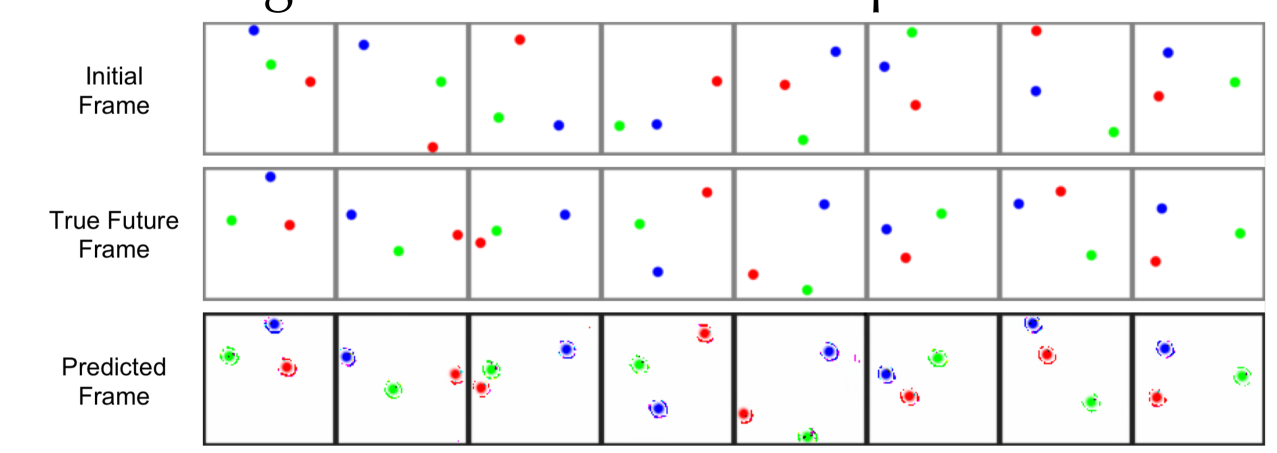
- When labels are available, prediction uses the true labels (notice the lack of hats):

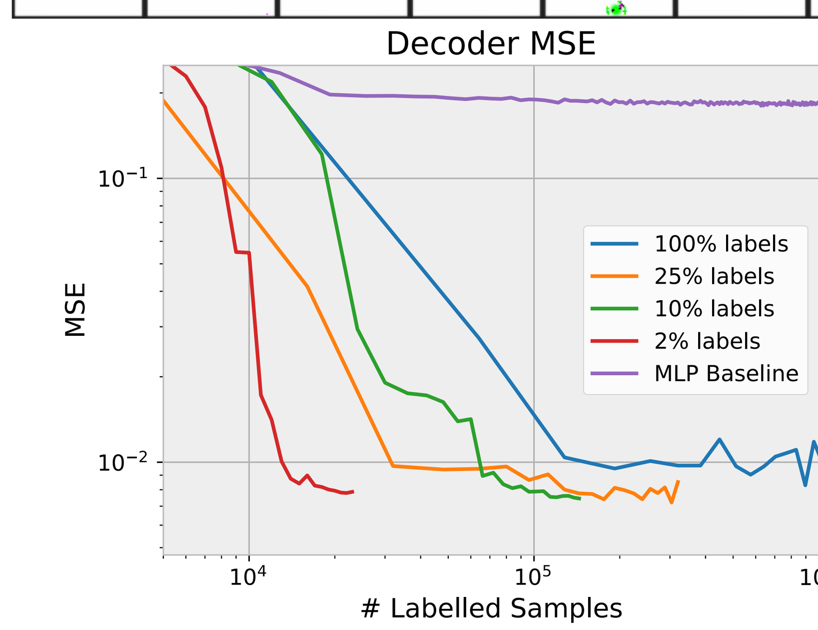
$$\hat{\phi}_t = encoder(x), \quad \hat{\phi}_{t+dt} = physics(\phi_t), \quad \hat{y} = decoder(\phi_{t+dt})$$

- Loss is composed of three terms when labels are available, or only the decoder loss when unlabeled

$$\mathcal{L} = \mathcal{L}_{enc} + \mathcal{L}_{phys} + \mathcal{L}_{dec},$$
$$\mathcal{L}_{enc} = \ell(\hat{\phi}_t, \phi_t), \ \mathcal{L}_{phys} = \ell(\hat{\phi}_{t+dt}, \phi_{t+dt}), \ \mathcal{L}_{dec} = \ell(\hat{y}, y)$$

- Physics structure embedded into the model allows for learning with few labeled samples





External resources

- Code available at:

https://github.com/locuslab/lcp-physics