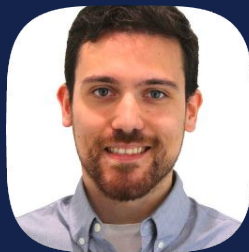


DeepMind

Constraint-Based Graph Network Simulator



Yulia
Rubanova*



Alvaro
Sanchez-Gonzalez*



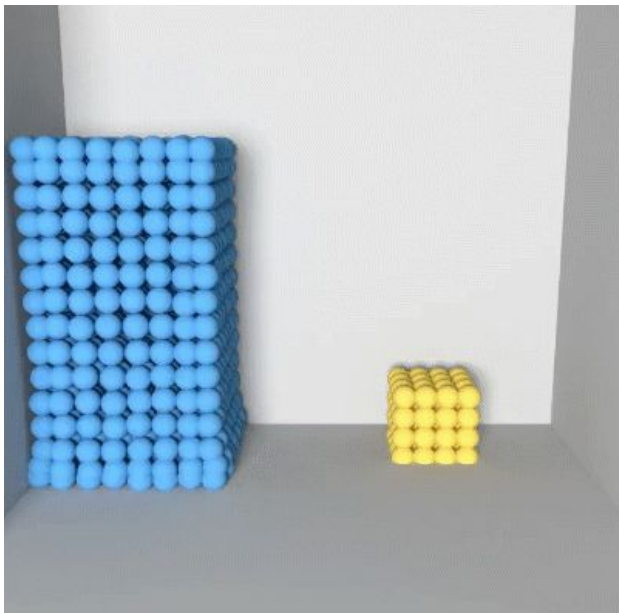
Tobias
Pfaff



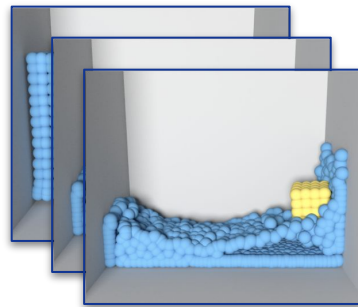
Peter
Battaglia



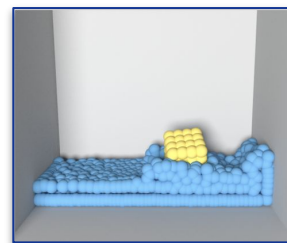
Physical simulations



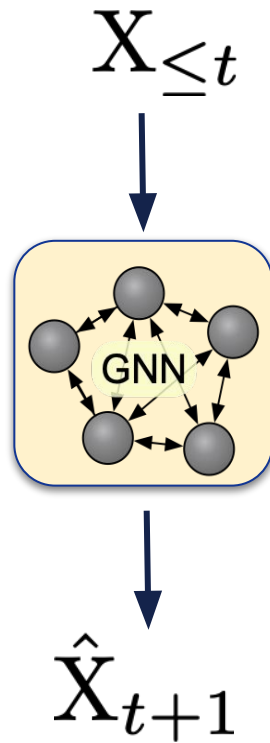
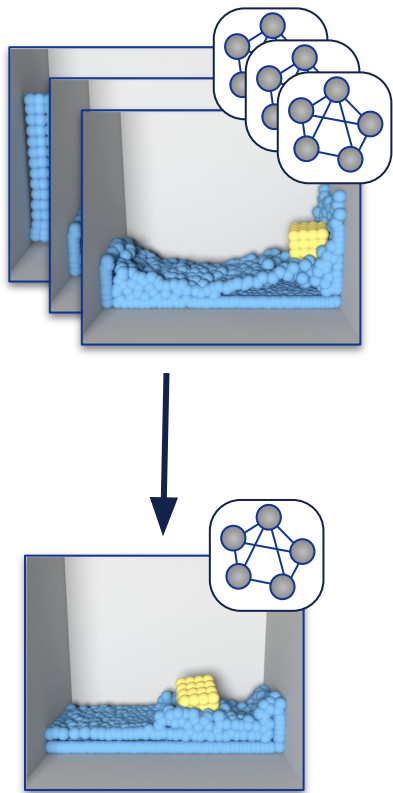
$$\mathbf{X}_{\leq t}$$



$$\hat{\mathbf{X}}_{t+1}$$



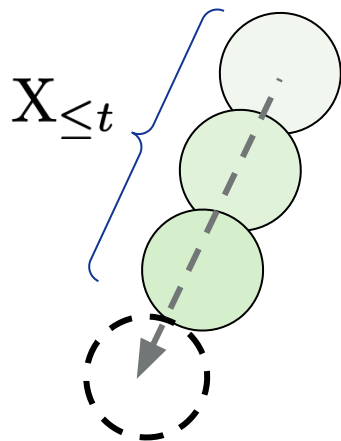
Graph Network Simulator*



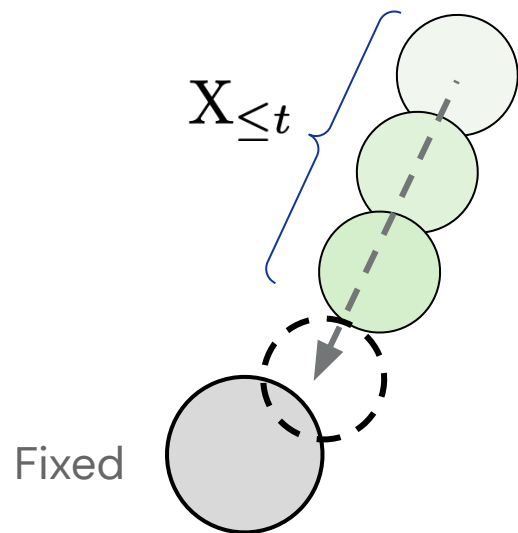
The next state is predicted directly



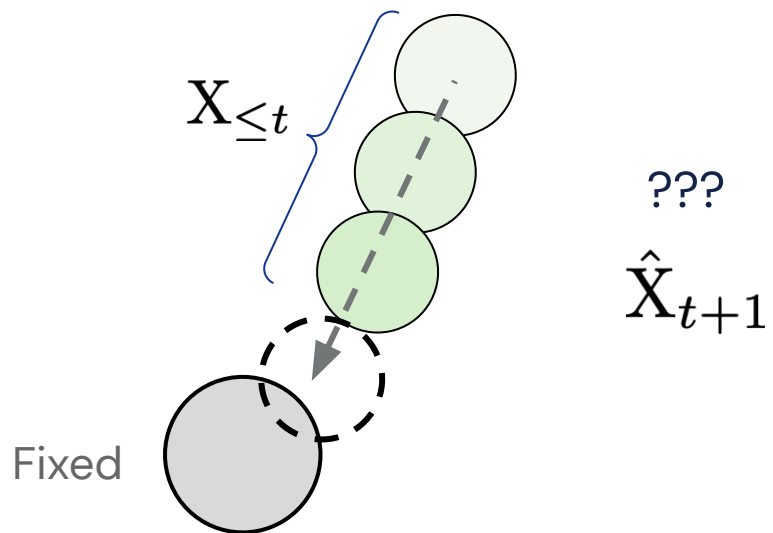
Graph Network Simulator



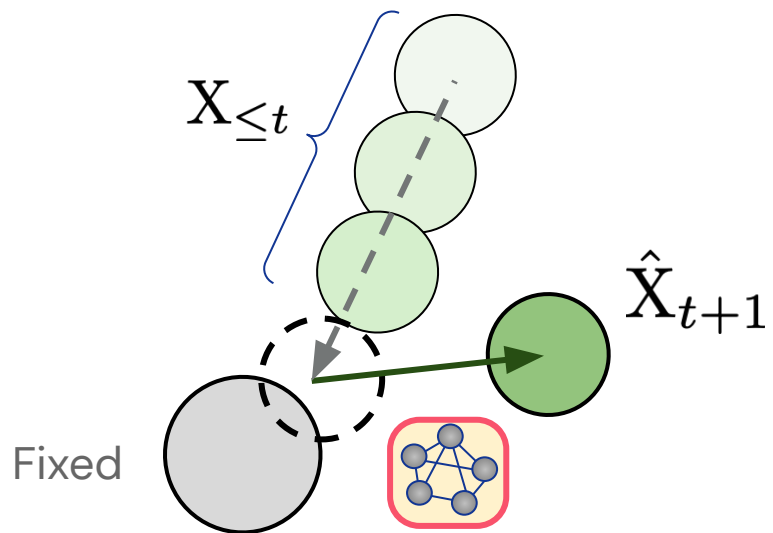
Graph Network Simulator



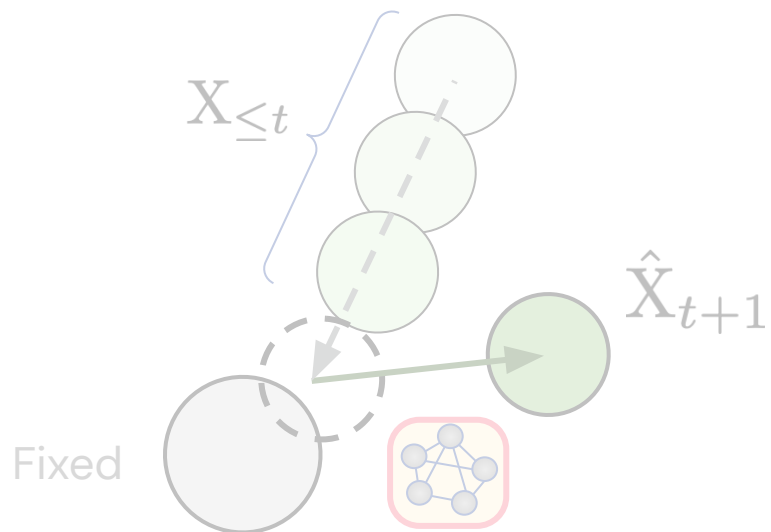
Graph Network Simulator



Graph Network Simulator



Graph Network Simulator



Traditional Physical Simulators

Physical constraints

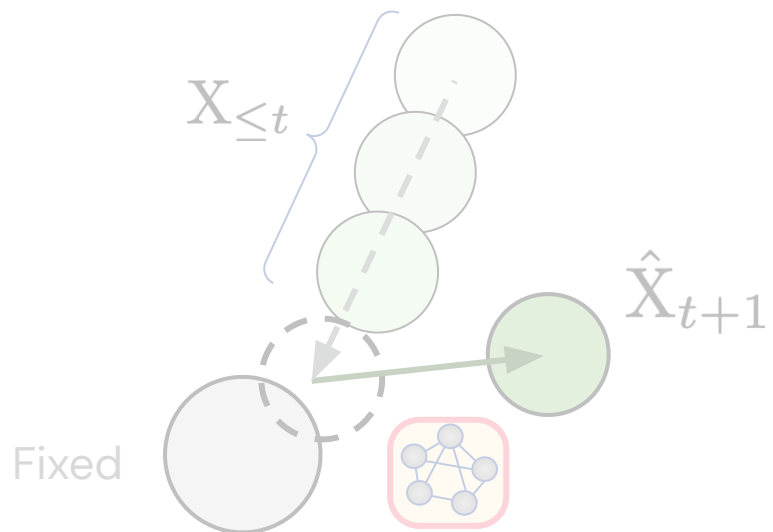
$$m_1 v_{1i} + m_2 v_{2i} = m_1 v_{1f} + m_2 v_{2f}$$

$$\frac{1}{2} m_1 v_{1i}^2 + \frac{1}{2} m_2 v_{2i}^2 = \frac{1}{2} m_1 v_{1f}^2 + \frac{1}{2} m_2 v_{2f}^2$$

Solve to find \hat{X}_{t+1}



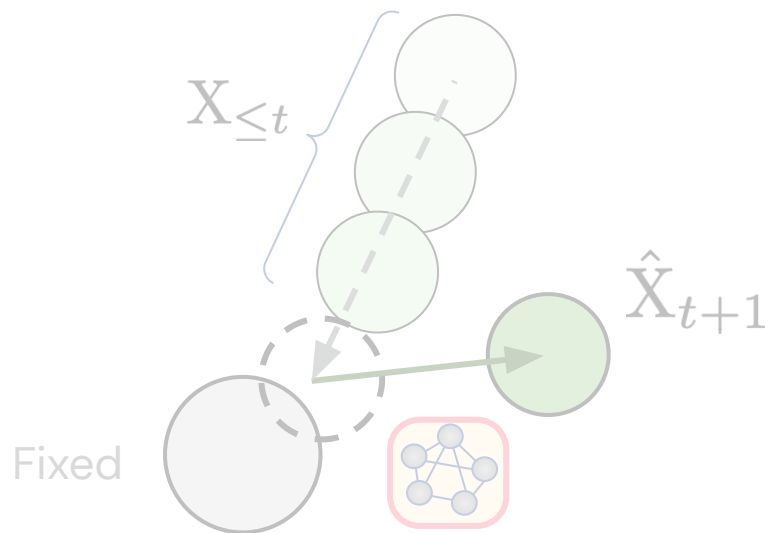
Graph Network Simulator



Our approach

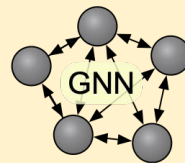


Graph Network Simulator

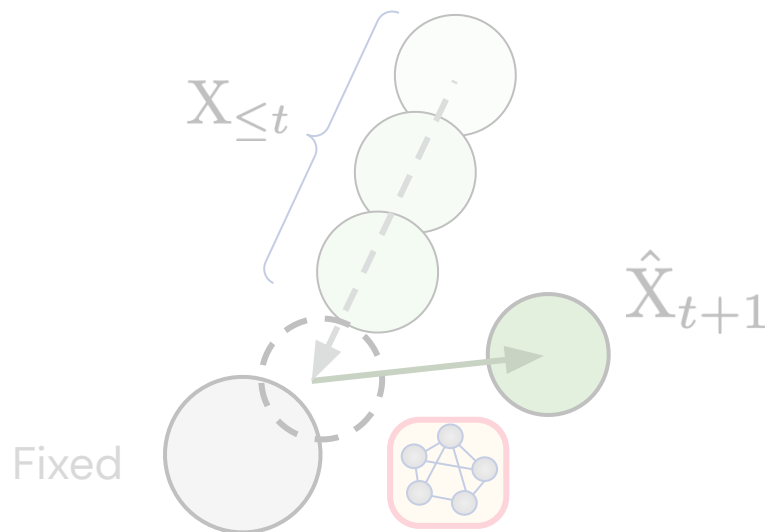


Our approach

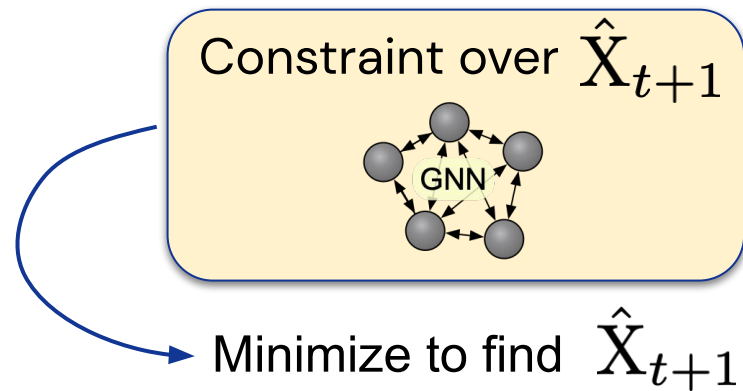
Constraint over \hat{X}_{t+1}



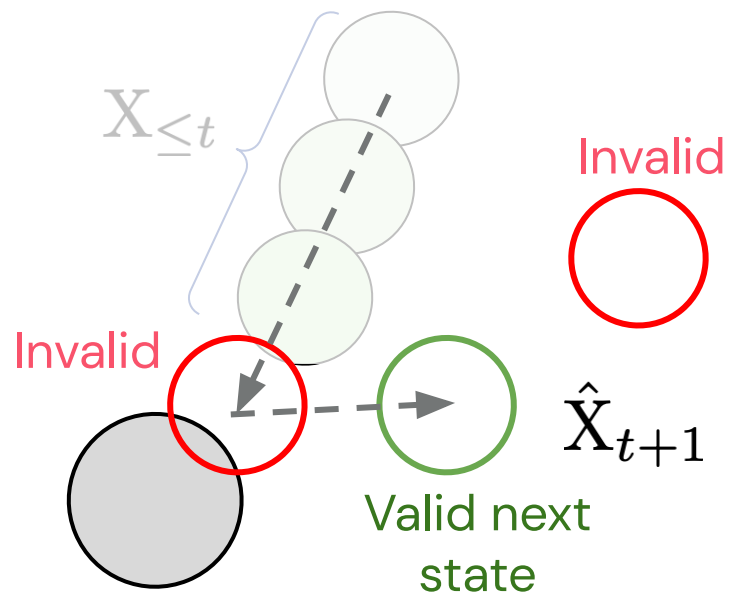
Graph Network Simulator



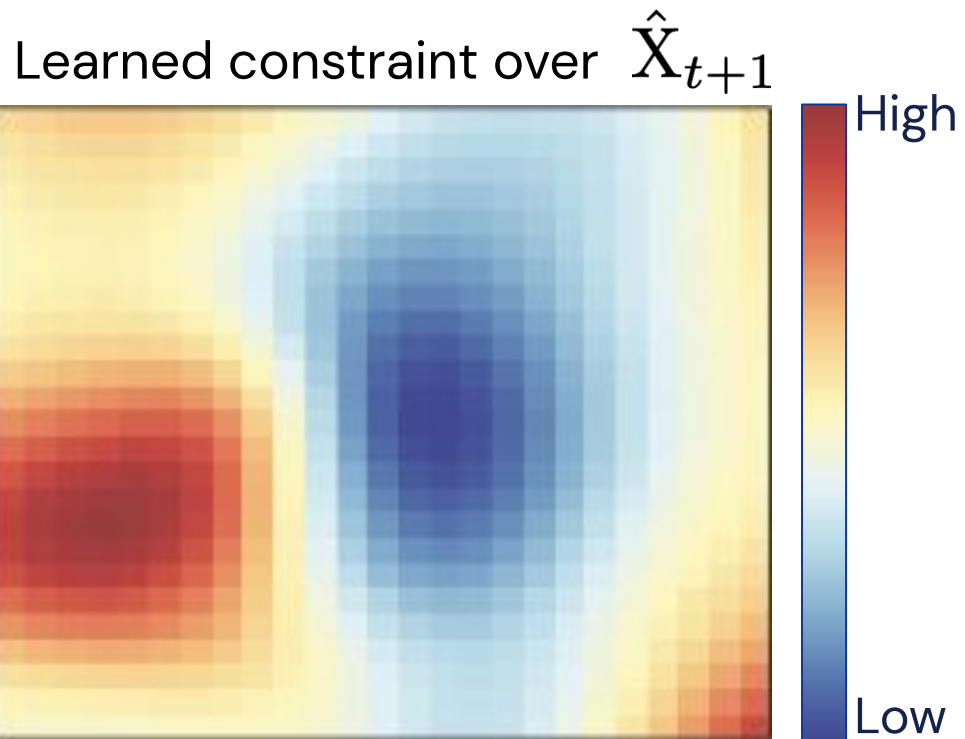
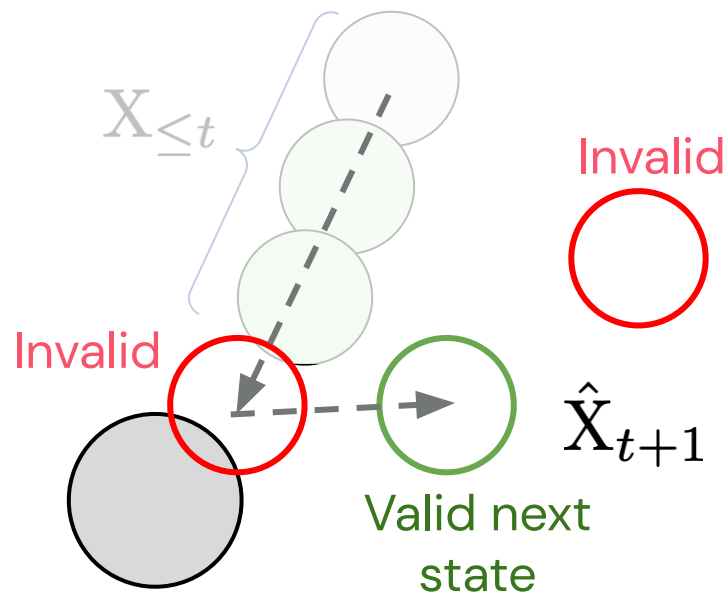
Our approach



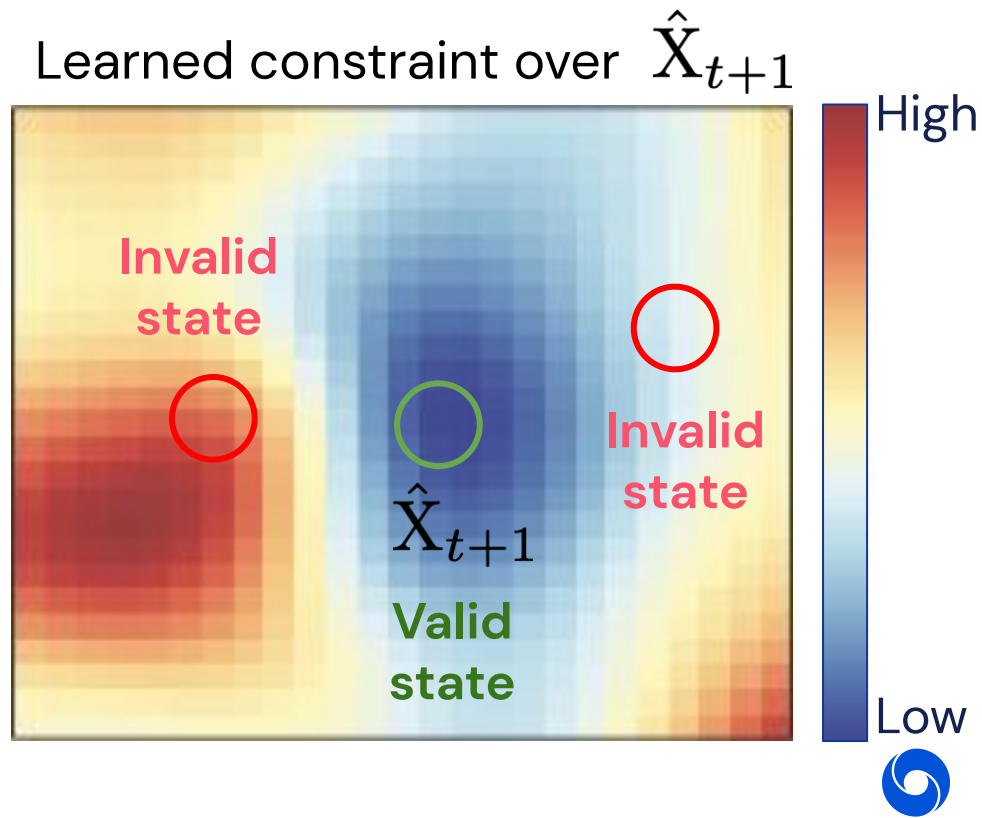
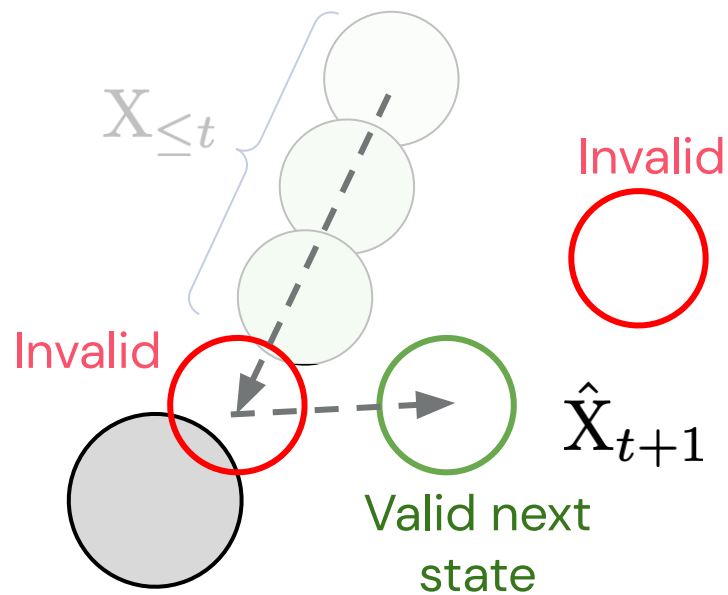
What does the constraint function learn?



What does the constraint function learn?



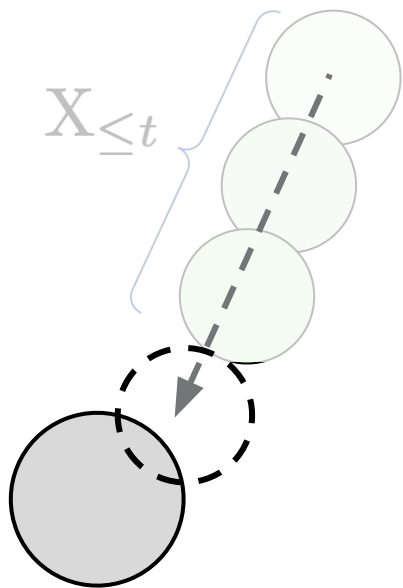
What does the constraint function learn?



Forward pass and training

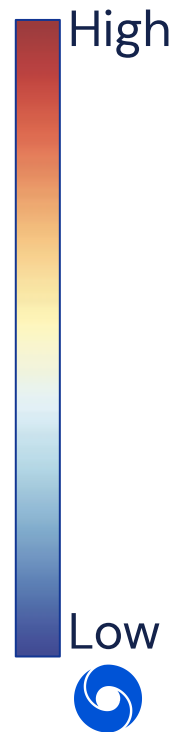
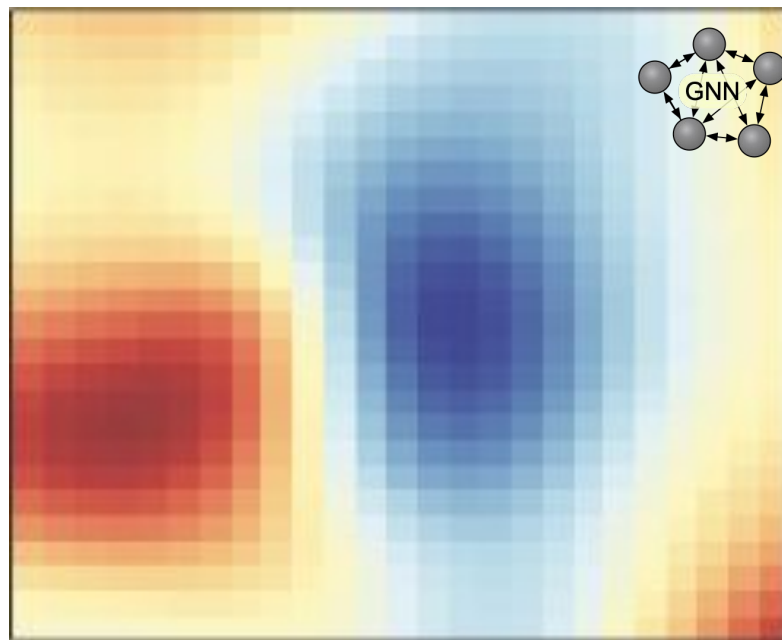


Forward pass: find the minimum of the constraint function



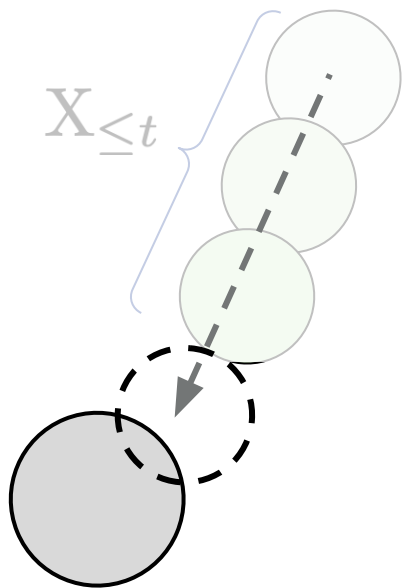
???

\hat{X}_{t+1}



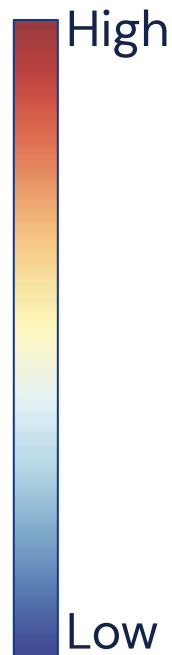
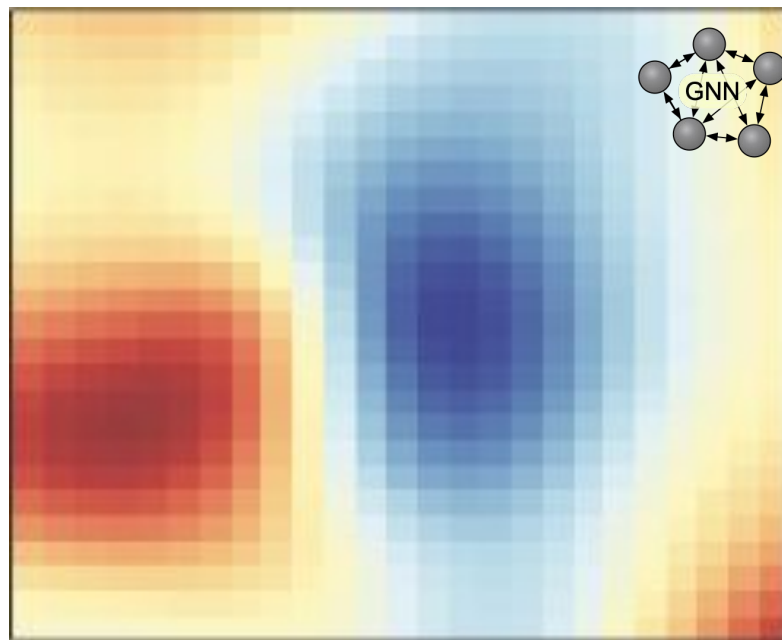
Forward pass: find the minimum of the constraint function

Run gradient descent to find \hat{X}_{t+1}



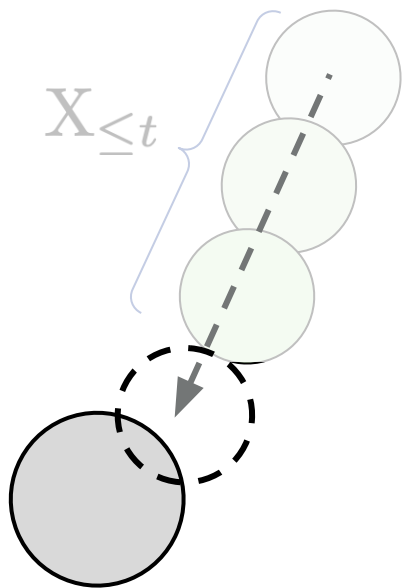
???

\hat{X}_{t+1}



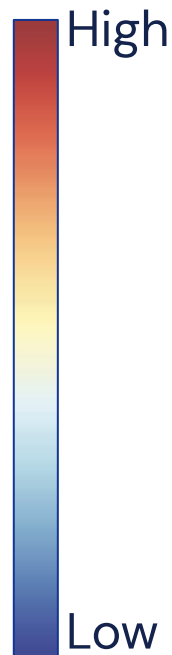
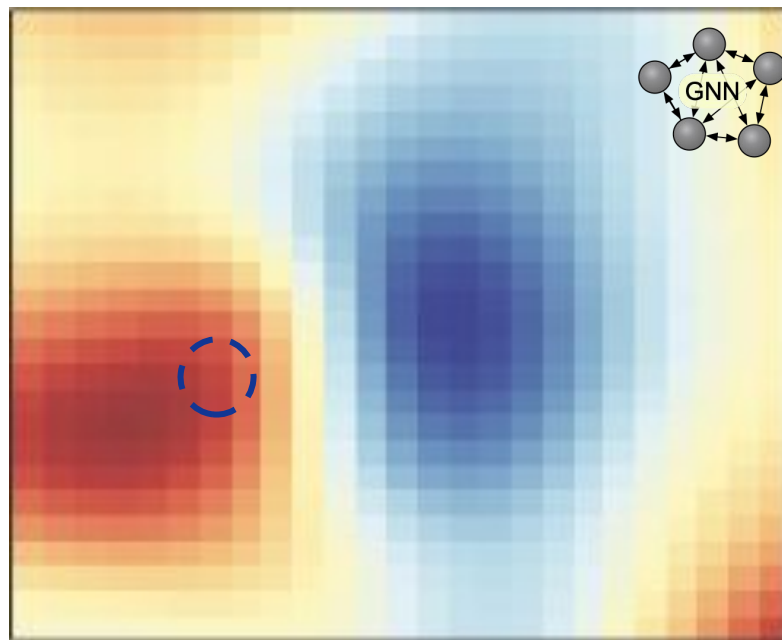
Forward pass: find the minimum of the constraint function

Run gradient descent to find \hat{X}_{t+1}



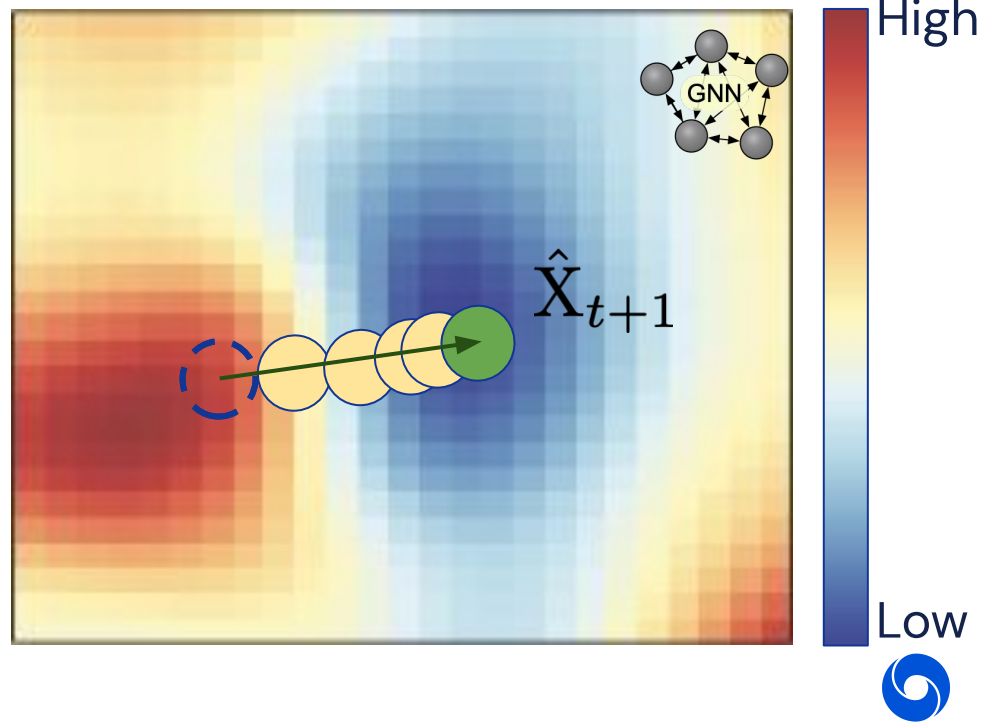
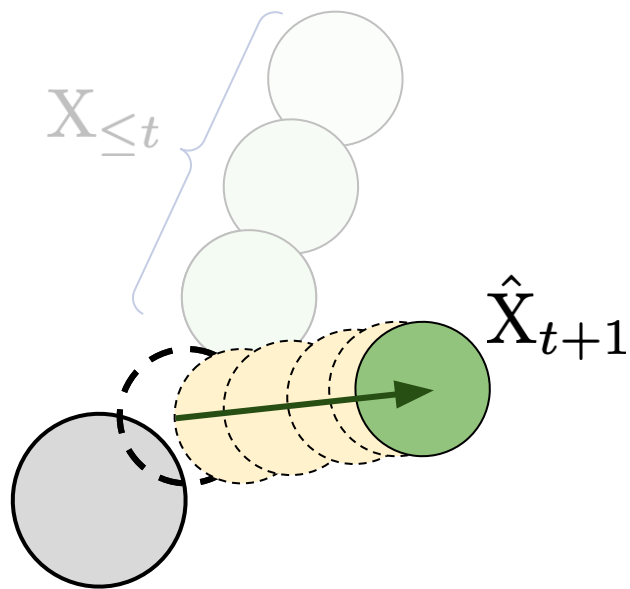
???

\hat{X}_{t+1}

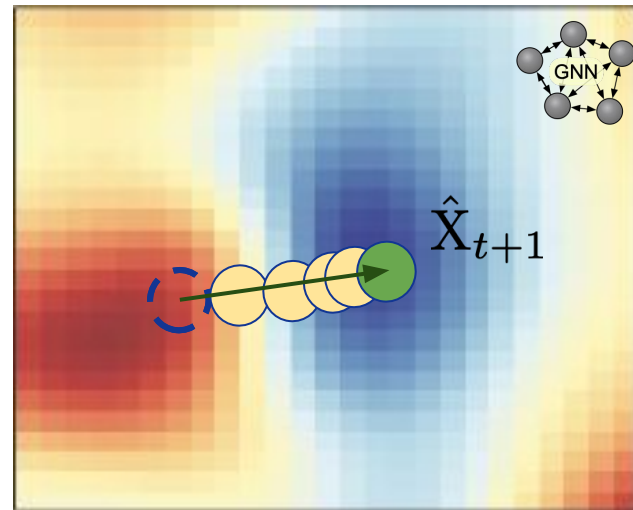


Forward pass: find the minimum of the constraint function

Run gradient descent to find \hat{X}_{t+1}

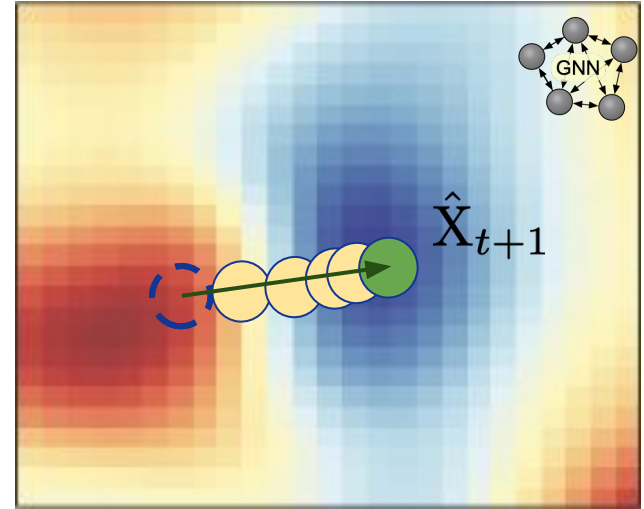


Training



Training

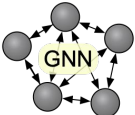
GD iterations are differentiable!

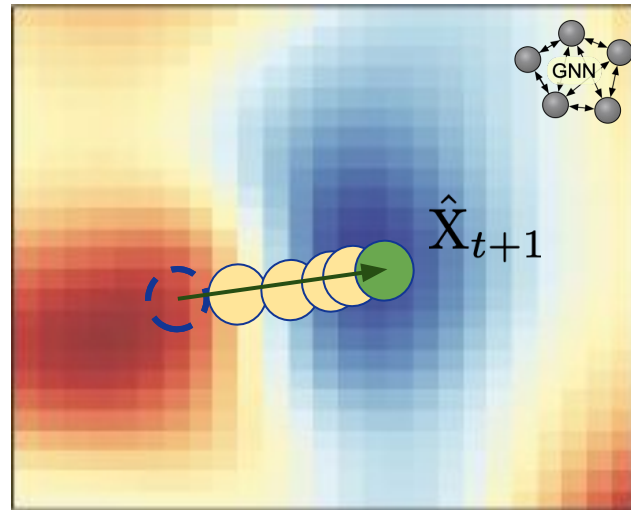


Training

GD iterations are differentiable!

Compute $\mathcal{L} = \text{MSE}(\hat{X}_{t+1}, X_{t+1})$

Backprop through GD solver to
update the  parameters

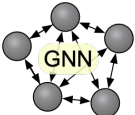


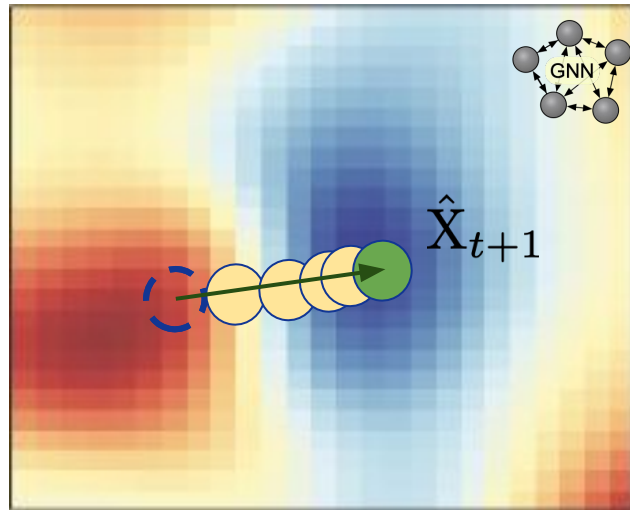
Training

GD iterations are differentiable!

Compute $\mathcal{L} = \text{MSE}(\hat{X}_{t+1}, X_{t+1})$

Backprop through GD solver to

update the  parameters



We don't need supervision on constraints

Model is trained from position trajectories

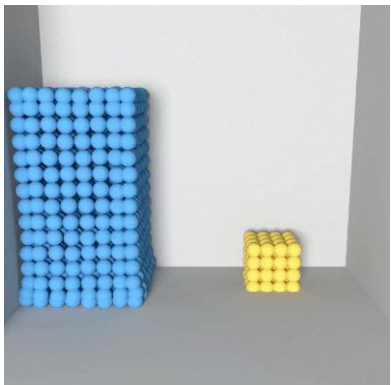
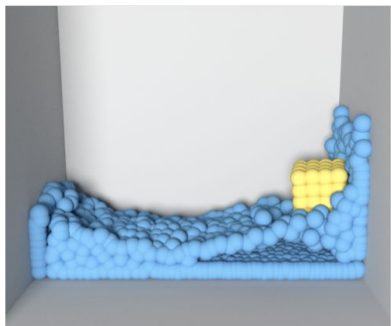


Experiments

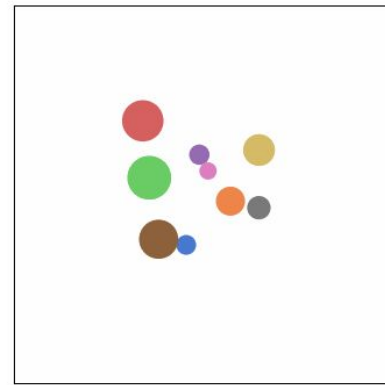


Domains

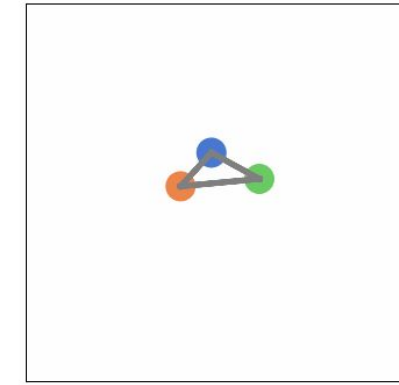
BoxBath



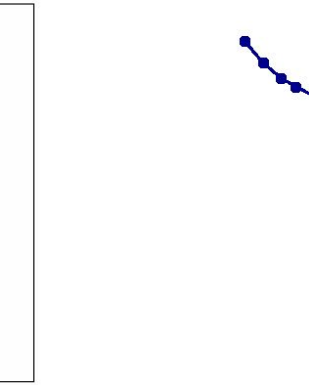
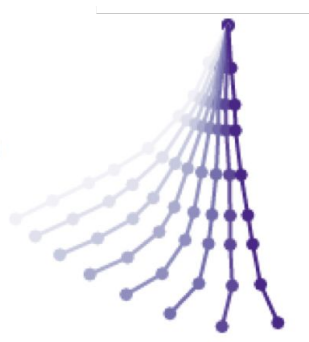
Bouncing Balls



Bouncing Rigid



Rope



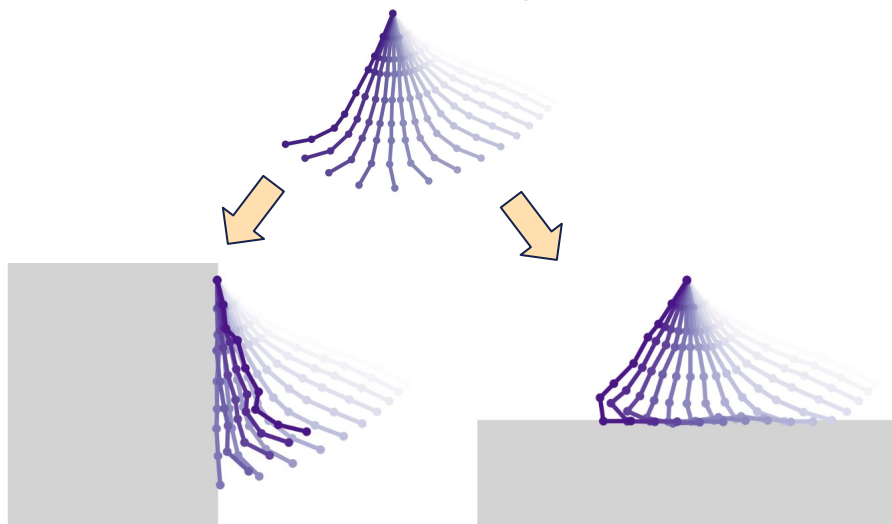
Unique advantages of the constraint model

We can add new constraints at test time

At test time optimize:

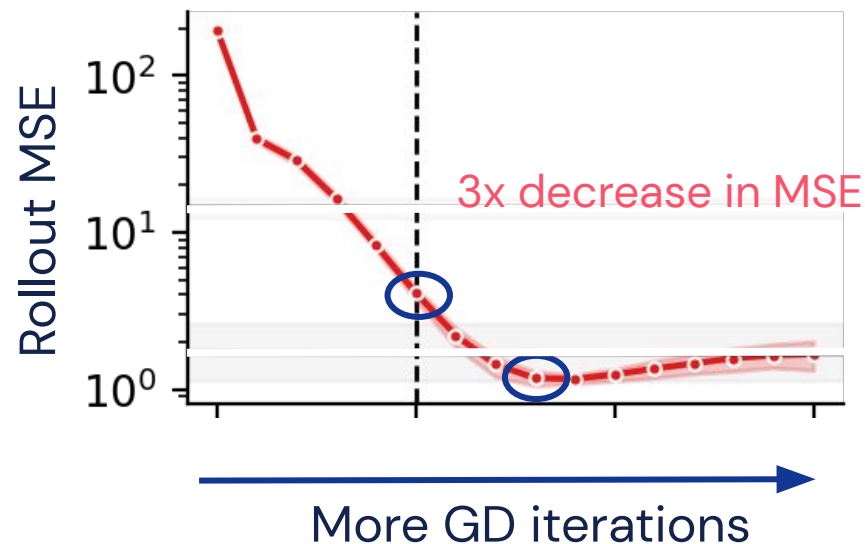
$$f_C(\mathbf{X}_{\leq t}, \mathbf{X}_{t+1}) + f_{\text{obstacle}}(\mathbf{X}_{t+1})$$

a learned constraint a new constraint
(e.g. new obstacle)

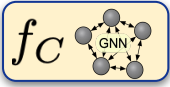


No collisions ever observed at training time!

We can tune the simulation accuracy at test time

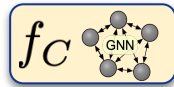


Constraint-Based Graph Network Simulator

Model idea: Learn a constraint f_C  over the future states

Find the future state as the minimum of the constraint

f_C  is trained entirely from the position trajectories



Unique properties:

- Iteratively refine a solution and improve accuracy at test time
 - Adding new constraints at test time
 - Learning an interpretable space of possible solutions
 - Model complex systems using a shallow GNN
- } Not possible with previous models

