Supplementary: Learning Articulated Rigid Body Dynamics with Lagrangian Graph Neural Network

Ravinder Bhattoo
Department of Civil Engineering
Indian Institute of Technology Delhi
`cez177518@iitd.ac.in`

Sayan Ranu
Department of Computer Science and Engineering
Yardi School of Artificial Intelligence
Indian Institute of Technology Delhi
`sayanranu@iitd.ac.in`

N. M. Anoop Krishnan
Department of Civil Engineering
Yardi School of Artificial Intelligence
Indian Institute of Technology Delhi
`krishnan@iitd.ac.in`

A Appendix

A.1 Experimental systems

Here, we describe the details of the experimental systems considered in the work, namely, n-link systems that represent a chain when the bars are relatively long and represent a rope when the bars are very small in comparison to the total length of the chain/rope. In addition, we also describe the complex systems, which can be considered as the tensegrity structures. We describe the constraints and the physics-based equations for predicting the Lagrangian of these systems.

A.1.1 n-link

In an n-link system, links are connected by hinges where they are free to rotate. These links are rigid, thus, impose a distance constraint between both the end points as

$$||q_i - q_{i-1}|| = l_i \tag{13}$$

where, l_i represents the length of the link connecting the $(i-1)^{th}$ and i^{th} node. This constraint can be differentiated to write in the form of a Pfaffian constraint as

$$q_i\dot{q}_i - q_{i-1}\dot{q}_{i-1} = 0 \tag{14}$$

Note that such constraint can be obtained for each of the n links considered to obtain the $A(q)$.

The Lagrangian of this system has contributions from potential energy due to gravity and kinetic energy. Thus, the Lagrangian can be written as

$$\mathcal{L} = \sum_{i=1}^{n} \left(1/2 m_i \dot{q}_{cm,i}^{\mathrm{T}} \dot{q}_{cm,i} + 1/2 I \omega_{cm}^2 - m_i g y_{cm,i}\right) \tag{15}$$

where g represents the acceleration due to gravity in the y direction, I represents the moment of inertia of link, ω represents the angular velocity and m represents the mass of the link.

A.1.2 Complex system

The complex system (with tensegrity structures) is a combination of the rope and rod system. Here, components with segment size significantly lower than the size of the system are termed as rope and

36th Conference on Neural Information Processing Systems (NeurIPS 2022).

parts with single segment, where the length of the segment is comparable to that of the system, is termed as a rod. These rods and ropes are connected such that ropes are in tension and rods are in compression holding the whole structure (see Fig. 3). The constraints similar to the n-link system are present in this system as well. The Lagrangian of this system is given as

$$\mathcal{L} = \sum_{i=1}^{n} \left(1/2 m_i \dot{q}_{i,cm}^{\mathsf{T}} \dot{q}_{i,cm} + 1/2 I \omega_{cm}^2 - m_i g y_{i,cm}\right) \tag{16}$$

where g represents the acceleration due to gravity in the y direction, I represents the moment of inertia of link, ω represents the angular velocity and m represents the mass of the link.

A.1.3 Details of the system with varying link properties

Length: 0.81331408, 1.15980562, 0.8647536 and 1.17632355

Mass: 1.83244264, 1.18182497, 1.30424224, 1.43194502 and 1.61185289

Moment of inertial: 1.21233911, 1.18340451, 1.52475643 and 1.29122914

A.1.4 Systems with drag force

The drag force on a link is applied as $-0.01 \times v$, where v is the velocity of the link (mean of velocities of both the nodes). The drag force is equally distributed to the nodes of the link.

A.2 Derivation of λ

From the EL equation, the acceleration is given by 2

$$\ddot{q} = M^{-1}\left(-C\dot{q} + \Pi + \Upsilon - A^T(q)\lambda + F\right) \tag{17}$$

and the constraints equation is given by $A(q)\ddot{q} + \dot{A}(q)\dot{q} = 0$. Substituting for $\ddot{q}$ from Eq 17, the constraint equation is modified as

$$AM^{-1}\left(-C\dot{q} + \Pi + \Upsilon - A^T\lambda + F)\right) + \dot{A}\dot{q} = 0 \tag{18}$$

where $A(q)$ and $\dot{A}(q)$ is written as A and $\dot{A}$, respectively for simplicity. The equation can be simplified as

$$\begin{aligned} AM^{-1}A^T\lambda &= \dot{A}\dot{q} + AM^{-1}\left(-C\dot{q} + \Pi + \Upsilon + F)\right) \\ \lambda &= (AM^{-1}A^T)^{-1}\left(\dot{A}\dot{q} + AM^{-1}\left(-C\dot{q} + \Pi + \Upsilon + F)\right)\right) \end{aligned} \tag{19}$$

A.3 Trajectory visualization

For visualization of trajectories of actual and trained models, videos are provided as supplementary material. The supplementary materials contains:
(a) **8-link system:** Here, an 8-link chain is connected to supports at both ends is stretched downward in initial condition.
(b) **10-link system:** Here, a 10-link chain is connected to support as one end and free from other end.
(c) **Tensegrity structure (T2):** A tensegrity structure is created using rope and rodes as described in section A.1.2.

A.4 Implementation details

A.4.1 Dataset generation

Software packages: numpy-1.20.3, jax-0.2.24, jax-md-0.1.20, jaxlib-0.1.73, jraph-0.0.1.dev0

Hardware: Memory: 16GiB System memory, Processor: Intel(R) Core(TM) i7-10750H CPU @ 2.60GHz

All the datasets are generated using the known Lagrangian for the system along with the constraints, as described in Section A.1. For 4-link system, we create the training data by performing forward simulations. A timestep of $10^{-5}s$ is used to integrate the equations of motion. The velocity-verlet algorithm is used to integrate the equations of motion due to its ability to conserve the energy in long trajectory integration.

The datapoints were collected every 10^{-3} time-step and for 10000 datapoints. It should be noted that in contrast to the earlier approach, here, we do not train from the trajectory. Rather, we randomly sample different states from the training set to predict the acceleration.

A.5 Architecture

For LGNN, we use two different size of fully connected feed forward neural network for initial embeddings and message passing. For initial embeddings, we use a linear layer to project the features to latent space with size 5. For message passing, we use a MLPs with a single hidden layer with 10 neurons. We use squareplus as activation functions for all non-linear MLPs.

For GNS and LGN, we follow a similar architecture suggested in the literature [19, 22]. Specifically, For initial embedding of node and edge features we use linear transformation with latent dimension with size 8. We use message passing MLPs with a single layer with 128 neuron. We use squareplus as activation functions for all non-linear MLPs. For GNS the output is acceleration values at node and for LGN the output is lagrangian of the whole system.

For CLNN, we follow a similar architecture suggested in the literature [6, 13]. Specifically, we use a fully connected feed forward neural network with 2 hidden layers each having 128 hidden units with a square-plus activation function. For LGNN, all the MLPs consist of two layers with 5 hidden units, respectively. Thus, LGNN has significantly lesser parameters than LNN.

A.6 Training details

The training dataset is divided in 75:25 ratio randomly, where the 75% is used for training and 25% is used as the validation set. Further, the trained models are tested on its ability to predict the correct trajectory, a task it was not trained on. All models are trained for 10000 epochs with early stopping. A learning rate of 10^{-3} was used with the Adam optimizer for the training.

•Lagrangian Graph Neural Network (LGNN)

Parameter	Value
Node embedding dimension	5
Edge embedding dimension	5
Hidden layer neurons (MLP)	10
Number of hidden layers (MLP)	1
Activation function	squareplus
Optimizer	ADAM
Learning rate	$1.0e^{-3}$
Batch size	10

•Graph Neural Simulator (GNS) and Lagrangian Graph Network (LGN)

Parameter	Value
Node embedding dimension	8
Edge embedding dimension	8
Hidden layer neurons (MLP)	128
Number of hidden layers (MLP)	1
Activation function	squareplus
Optimizer	ADAM
Learning rate	$1.0e^{-3}$
Batch size	10

•Constraint Lagrangian Neural Network (CLNN)

Parameter	Value
Hidden layer neurons (MLP)	128
Number of hidden layers (MLP)	2
Activation function	squareplus
Optimizer	ADAM
Learning rate	$1.0e^{-3}$
Batch size	10

A.7 Rollout and energy error

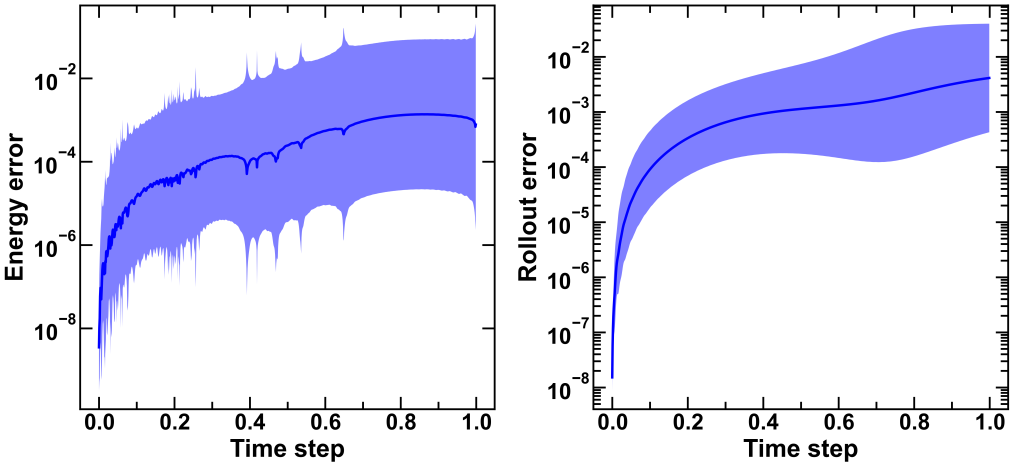

Figure 8: (a) Energy and (b) rollout error of 4–link chains (with different length, mass and moment of inertial) predicted using LGNN in comparison with ground truth. Shaded region shows the 95% confidence interval based on 20 forward simulations with random initial conditions.

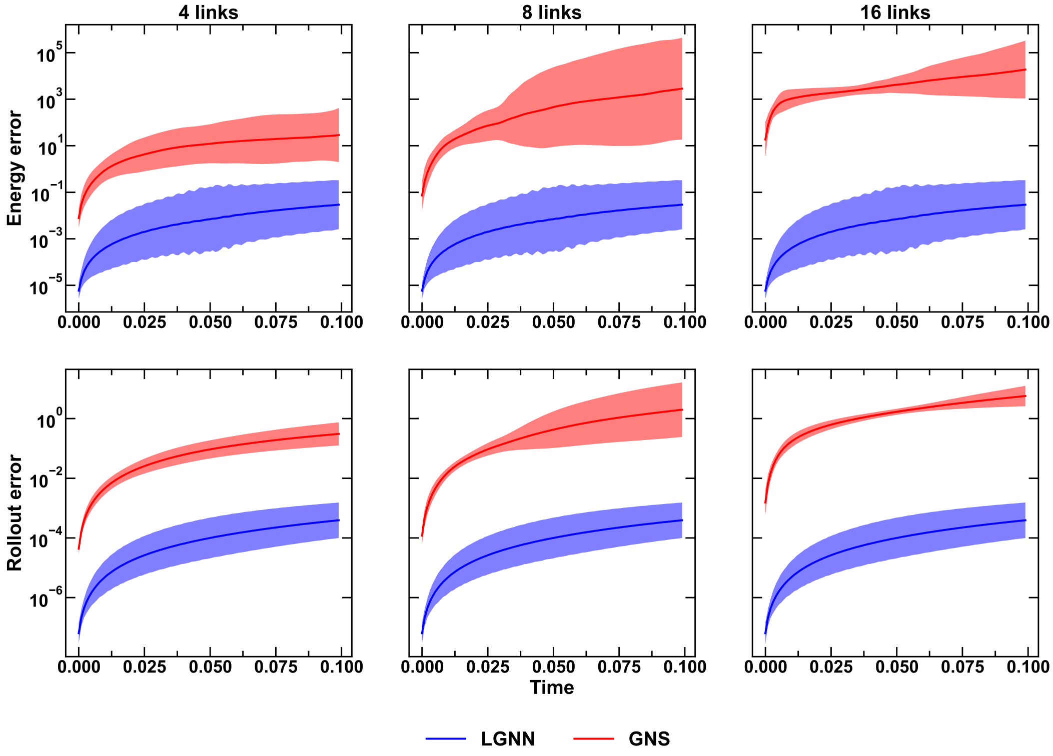

Figure 9: Absolute energy and rollout error of $n-$link chains predicted using LGNN in comparison with ground truth for $n = 4, 8, 16$. Note that LGNN is trained only on the 4-link chain and predicted on all the systems. Shaded region shows the 95% confidence interval based on 100 forward simulations with random initial conditions.

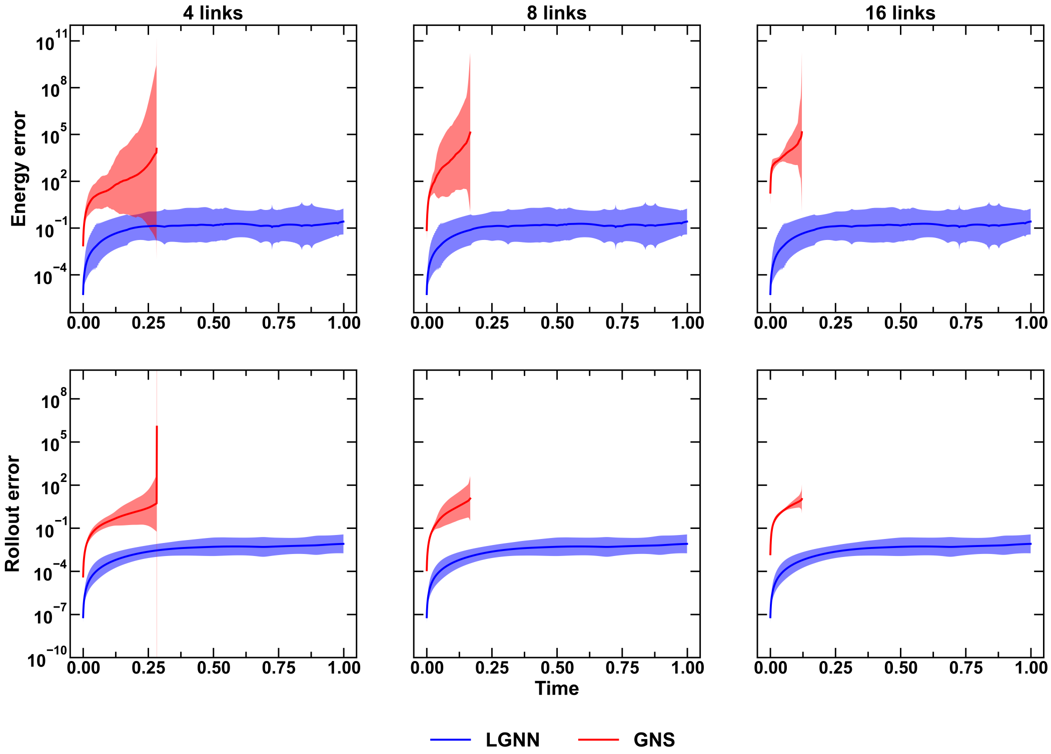

Figure 10: Absolute energy and rollout error of $n-$link chains predicted for a longer trajectory using LGNN in comparison with ground truth for $n = 4, 8, 16$. The simulation using GNS exploded after some time leading to truncation of plot. Note that LGNN is trained only on the 4-link chain and predicted on all the systems. Shaded region shows the 95% confidence interval based on 100 forward simulations with random initial conditions.

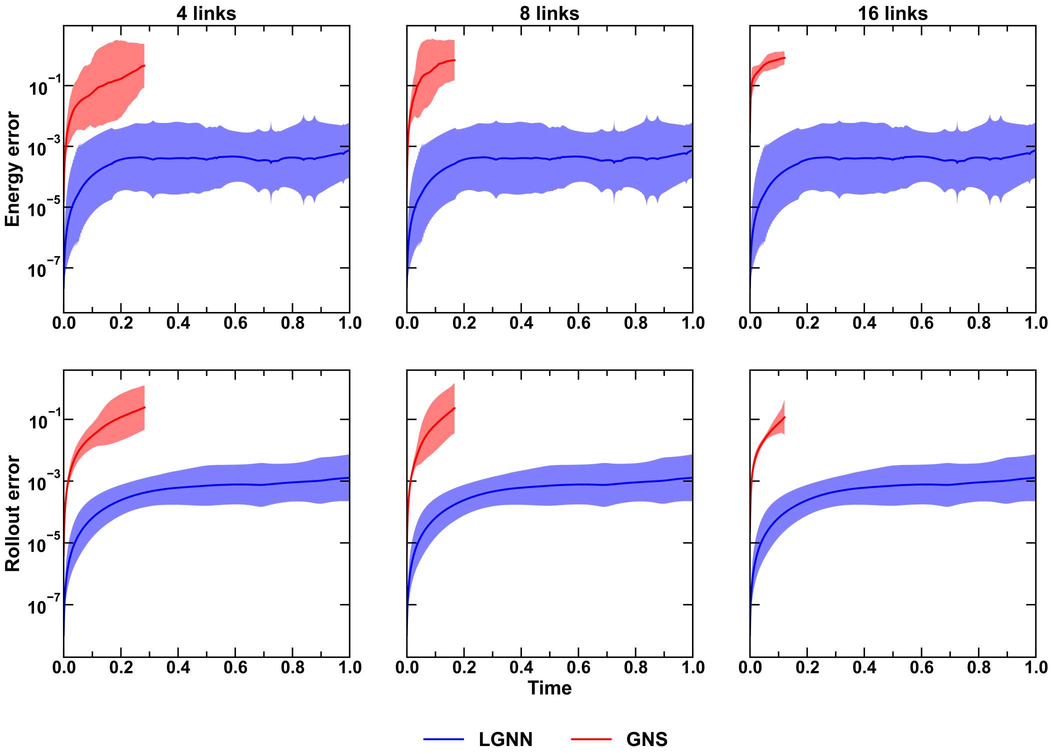

Figure 11: Energy and rollout error of $n-$link chains predicted for a longer trajectory using LGNN in comparison with ground truth for $n = 4, 8, 16$. The simulation using GNS exploded after some time leading to truncation of plot. Note that LGNN is trained only on the 4-link chain and predicted on all the systems. Shaded region shows the 95% confidence interval based on 100 forward simulations with random initial conditions.

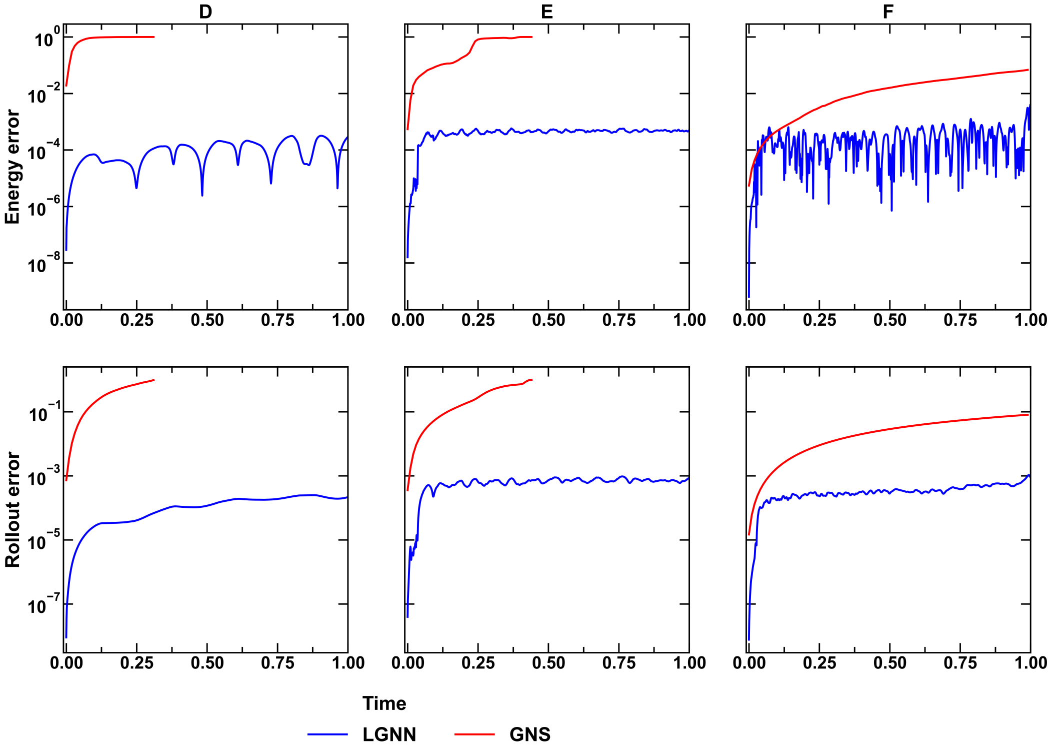

Figure 12: Absolute energy and rollout error of systems predicted using LGNN in comparison with ground truth for T1, T2, and T3. Note that LGNN is trained only on the 4-segment chain and predicted on all the systems. Since the starting configuration of the simulation is fixed (perfect structure as shown in Figure 3), there are no error bars generated for this system.

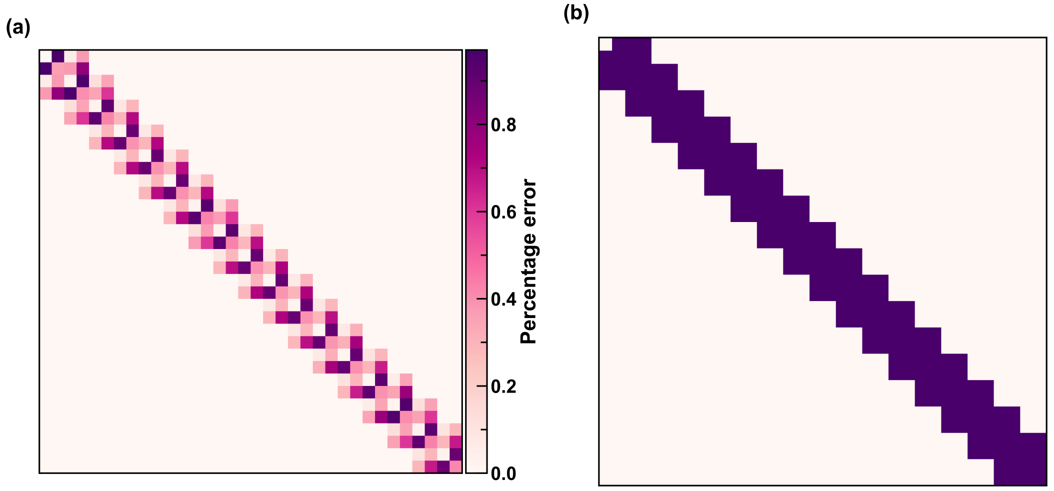

Figure 13: (a) Percentage error and (b) non-zero terms in the percentage error of the mass matrix learned by the LGNN in comparison to the ground truth for 16 links. It may be noticed that the percentage error in the learned mass matrix is extremely low with a maximum of 1%.

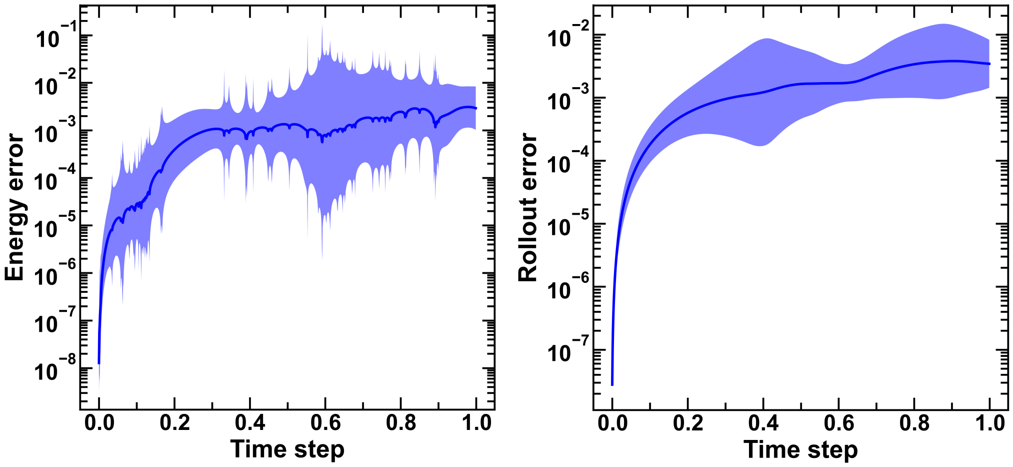

Figure 14: (a) Energy and (b) rollout error of 4–link chains with drag predicted using LGNN in comparison with ground truth. Shaded region shows the 95% confidence interval based on 20 forward simulations with random initial conditions.

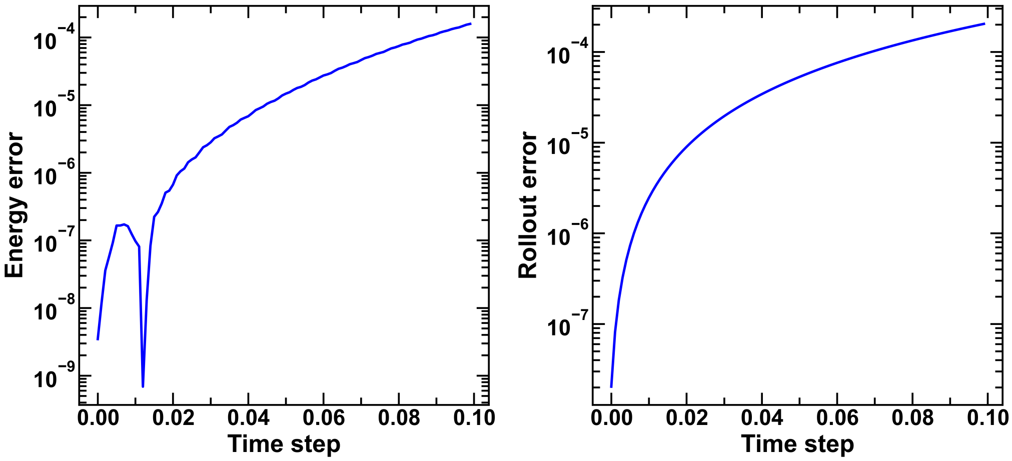

Figure 15: (a) Energy and (b) rollout error of 100–link chains predicted using LGNN in comparison with ground truth. Note that LGNN is trained only on the 4-segment chain and predicted on all the systems.

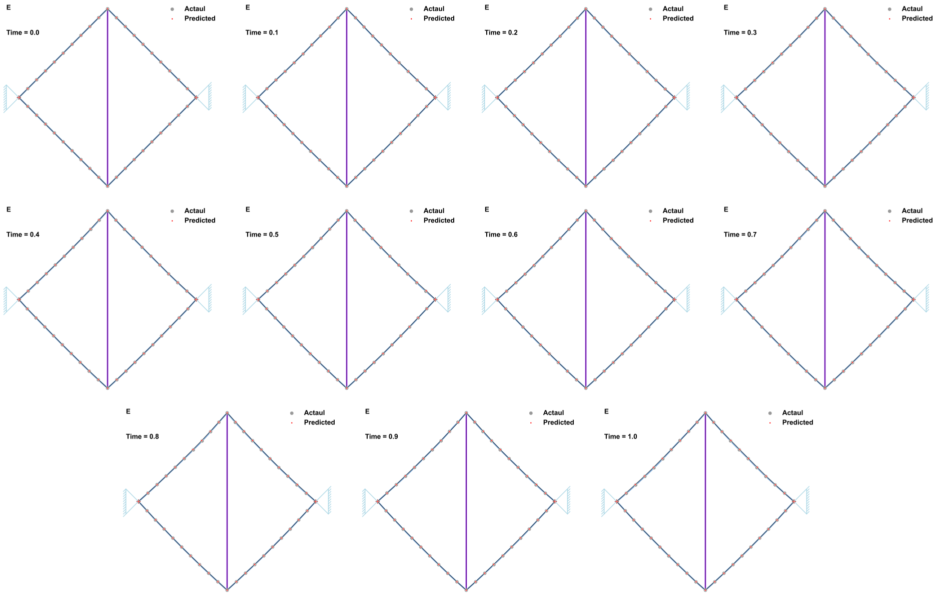

Figure 16: Snapshots of system E during simulation.

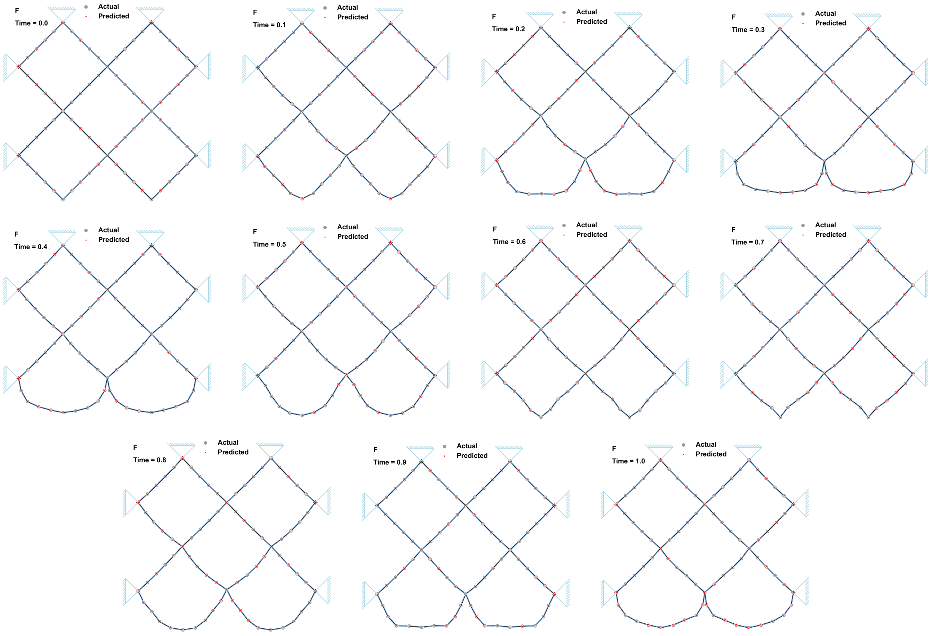

Figure 17: Snapshots of system F during simulation.