Physics-Informed Neural Controlled Differential Equations for Long Horizon Multi-Agent Motion Forecasting

Shounak Sural*
Carnegie Mellon University
Pittsburgh, PA, USA
ssural@andrew.cmu.edu

Charles Kekeh
Amazon Robotics
North Reading, MA, USA
ckkekeh@amazon.com

Wenliang Liu
Amazon Robotics
North Reading, MA, USA
liuwll@amazon.com

Federico Pecora
Amazon Robotics
North Reading, MA, USA
fpecora@amazon.com

Mouhacine Benosman
Amazon Robotics
North Reading, MA, USA
mbenos@amazon.com

Abstract

Long-horizon motion forecasting for multiple autonomous robots is challenging due to non-linear agent interactions, compounding prediction errors, and continuous-time evolution of dynamics. Learnt dynamics of such a system can be useful in various applications such as travel time prediction, prediction-guided planning and surrogate simulation. In this work, we aim to develop an efficient trajectory forecasting model conditioned on multi-agent goals. Motivated by the recent success of physics-guided deep learning for partially known dynamical systems, we develop a model based on neural Controlled Differential Equations (CDEs) for long-horizon motion forecasting. Unlike discrete-time methods such as RNNs and transformers, neural CDEs operate in continuous time, allowing us to combine physics-informed constraints and biases to jointly model multi-robot dynamics. The proposed approach learns differential equation parameters that can be used to predict the trajectories of a multi-agent system starting from an initial condition, while enforcing physics constraints to predict robot motion over extended periods of time. Our experiments show that our method is capable of learning non-trivial dynamics for such systems.

1 Introduction

Spatio-temporal dynamics modeling for multiple interacting autonomous mobile robots (AMRs) can be challenging [1–4]. Non-linear coupled dynamics of such robots, hidden agents and irregular sampling times across multiple robots often add to the complexity. Such dynamics modeling can benefit from methods that learn the continuous-time evolution of the joint state of the multi-agent system. Modern robot motion forecasting methods primarily rely on transformers or GNNs to model robot interactions and then predict robot poses in discrete-time ([3],[5],[2],[4]) with auto-regressive models. However, many such auto-regressive methods tend to predict divergent trajectories with errors compounding over time steps, especially in scenarios such as sharp turns, sudden events and cases with missing or irregularly-timestamped data. Even for pose data recorded at 1 Hz, there can still be significant movement and robot interactions within a one second time window. Furthermore, pose data is likely to be estimated at different times across robots within the time window, fundamentally

*Work done during internship at Amazon Robotics

Machine Learning and the Physical Sciences Workshop, NeurIPS 2025.

causing a mismatch when rounded off to the nearest second. With many interacting robots moving in close quarters, these truncation errors tend to add up over time. Neural Differential Equations (NDEs) [6, 7] are a type of neural network that can model the temporal dynamics of the multi-robot system more explicitly and smoothly in the form of continuous-time differential equations. Additionally, NDEs can naturally incorporate physics constraints directly into the motion forecasts.

Physics-informed neural networks [8–11] and neural operator learning [12–14], along with their combinations [15, 16], have shown promise in various applications in science such as chemical kinetics and fluid dynamics [11, 17]. They have been used to solve inverse problems in science [18–20] and to build neural surrogate simulators [20–22]. These models work well for modeling systems where the dynamics evolve based on physical laws, which are difficult to precisely model based on known equations. Multi-agent motion forecasting for autonomous robots is another such application where NDEs can potentially benefit the modeling of robot dynamics. In this work, we explore a variant of NDEs called Neural Controlled Differential Equations (NCDEs) [23] that can be used to explicitly model how control dynamics laws guide the evolution of robot states.

2 Related Work

Motion Forecasting: Well-known existing motion forecasting methods such as Trajectron++ [3] and PreCOG [24] typically deal with prediction of a future time window of a few seconds, and their rollouts for longer time horizons diverge from true states. Neural CDEs [23] have been explored as a continuous time equivalent of discrete-time RNN models [25]. However, the basic neural CDE architecture expects the entire time-series to be available at the time of inference. This does not work for robot forecasting tasks where predictions have to be made online. An extension to the neural CDE work [26] introduces an online variant. However, even in this case, the control path used by the model is the history of the variable being predicted. In this work, we differ from existing approaches by incorporating more explicit controls in terms of reference linear and angular velocities that the robots aim to reach over a long time horizon. Furthermore, by modeling higher order derivatives such as acceleration and enforcing soft unicycle dynamics constraints, our neural CDE variant can learn smooth and more realistic dynamics for the evolution of robot states over time given only an initial state. We posit that a model that is aware of the underlying physical laws, and learns deviations from such laws using data with a continuous-time differential equation, can stabilize long-horizon forecasts.

Robot Dynamics Simulation: In a different context, multi-agent reinforcement learning (RL)-based planners and other neural planners rely on data and scenario generation from simulators. These simulators can be slow due to explicit modeling of physics with graphics engines and are expensive to run. The slow speed of such simulators makes it difficult to mine a variety of challenging scenarios. An alternative approach to this problem is the use of a deep learning-based surrogate simulator that learns from real-world data, and can evolve in continuous time. We show that a neural CDE-based trajectory forecasting model that is capable of learning the underlying differential equations can effectively simulate dynamics over long horizons to replicate the capabilities of a much slower simulator. This can be used as a good resource for training RL policies, potentially helping with the sample efficiency problem for rare scenarios. In robotics and RL, the Sim2Real gap [27] describes how models and algorithms fail in the real world for robotic tasks after performing well in simulation. With a physics-informed neural CDE-based surrogate simulator that is trained purely on real-world data, it might also be possible to bridge this gap.

3 Physics-informed Neural CDEs for Motion Forecasting

We aim to model the dynamics of N robots that operate in a shared environment. The joint state of the multi-agent system composed of all these robots at time t is denoted by $S_t = (x_i^t, y_i^t, \theta_i^t), \forall i \in \{1, 2, ..., N\}$. Additionally, to model the dynamics of this system conditioned on goals, we aim to use Model Predictive Control (MPC)-generated control references over the time horizon as a time-series of goals. These control guidances, which are the reference linear and angular velocities, are represented as $C_t = (v_i^t, \omega_i^t), \forall i \in \{1, 2, ..., N\}$. Our goal is to learn a differential equation that describes how the robots move.

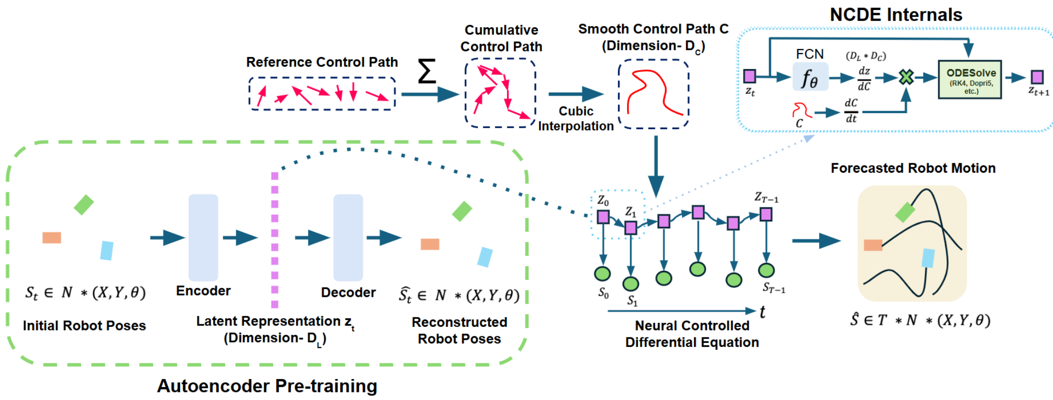

Figure 1: Architecture of our model that uses an autoencoder followed by a latent neural CDE which is additionally guided by reference controls for motion forecasting. S_t and Z_t represent the multi-agent state and latent state at time t, D_L refers to the dimension of the learned latent representation and D_C refers to the dimension of the reference controls.

Autoencoder: We start by training an autoencoder (AE) that denoises the raw input poses and learns a richer latent representation. The poses of all drives are then reconstructed from the latent state. The latent state provides an alternate basis space, which allows more efficient and expressive learning of dynamics within the neural CDE. This is the first step in our training, as shown in Figure 1. The encoder and decoder consist of simple 5-layer fully connected networks (FCNs) with hidden layers of size 512 that learn a 30-dimensional latent space for the neural CDE.

Latent CDE: Once the autoencoder is pretrained, we use the obtained latent representation for the neural CDE. For additional context, the robots operating in robotic warehouses use MPC to generate reference linear and angular velocities for the robots to track. This information is available to the robots beforehand at the start of each time horizon, either directly in the form of MPC references or can be derived approximately based on waypoint and goal information during inference. However, since CDEs work on continuous control paths, we take a cumulative sum of these reference controls over time, fit a piecewise cubic spline to generate a smooth curve, and then use this differentiable reference to learn the movement of robots over time.

The neural network f_θ, which is an FCN with 4 hidden layers of width 512, is learned from data, thereby modeling the parameters of the underlying differential equation. This derivative is combined with the derivative of the smooth reference control path with chain rule of differentiation. The neural differential equation is then used as part of a standard ODE solver such as the 4-th order Range-Kutta to integrate over time to predict future latent states over a 60 second time horizon. These states are then passed through the pre-trained decoder to reconstruct individual robot poses from the joint state.

Physics-Informed Loss: We start with a Mean Squared Error (MSE) reconstruction loss for the AE, followed by a combination of an MSE future pose prediction loss, a physics-informed unicycle loss for dynamic feasibility and an acceleration regularization loss to ensure smooth transitions in velocity across time steps. The pose prediction loss is defined as $L_{pred} = \|S_{1:T-1} - \hat{S}_{1:T-1}\|^2$, where S and $\hat{S}$ represent the ground truth and predicted states for the multi-agent system across the time horizon T. The unicycle loss is captured by $L_{uni} = \sum_{t=0}^{T-2}[(\dot{x}_t - v_t \cos\theta_t)^2 + (\dot{y}_t - v_t \sin\theta_t)^2 + (\dot{\theta}_t - \omega_t)^2]$ enforcing adherence to the physical constraints between the independently predicted velocity and angle predictions. Acceleration regularization is captured by $L_{acc} = \frac{1}{T-1}\sum_{t=0}^{T-2}\left(\frac{\|\hat{v}_{t+1}\| - \|\hat{v}_t\|}{\Delta t}\right)^2$ to ensure there are no sudden jumps in predictions. In summary, for the physics-informed training of the PINCoDE model, we use the combined Physics-Informed Forecasting Loss (PIFL) defined as

$$L_{PIFL} = W_{pred}L_{pred} + W_{uni}L_{uni} + W_{acc}L_{acc}$$
$$W_{pred}, W_{uni}, W_{acc} > 0$$

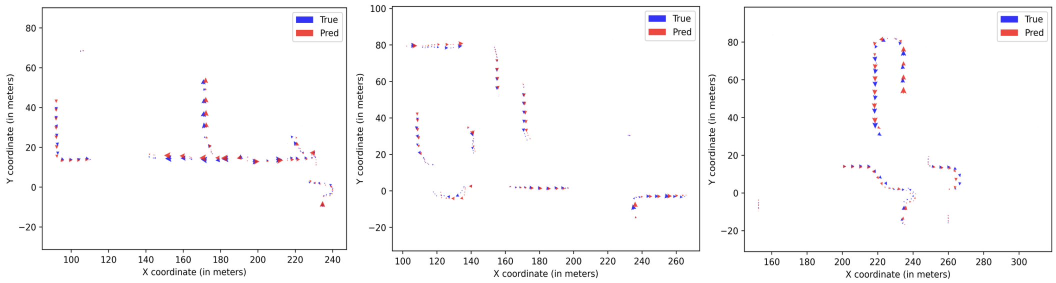

Figure 2: Motion forecasting performance over a 60 second time horizon. Blue arrows show ground truth and red arrows show prediction, with strong correspondences over long time horizons.

4 Experiments and Evaluation

Dataset: We use one month of data from a warehouse floor application that consists of multiple robots moving in an unstructured environment. This dataset records poses and other relevant information for several robots every second. For this study, we divide the data into four minute chunks with a slide length of 10 seconds and consider the top 10 robots with the most movement within this time window. This strategy is adopted to ensure that we have significant robot movement in our dataset, along with overlaps that effectively work as data augmentation.

We also model robot goals as part of our goal-driven motion forecasting approach. To achieve this, a sequence of target future velocities is available to the robots at the beginning of the prediction horizon. This is typically obtained through a combination of direct MPC-generated references for smaller windows (e.g. 5 seconds) and can be derived approximately for longer horizons (1-4 minutes) based on motion goal locations and intermediate waypoints. In practice, for training our model on warehouse robot data recorded over months, we use sparsely recorded actual future linear and angular velocities over the forecasting horizon, which serve as an approximate proxy for reference goal velocities. During inference and deployment, MPC generated goals can be used for shorter horizons and reference velocities based on waypoints for the longer ones, potentially with more finetuning.

Furthermore, pose data from various robots get estimated and recorded at different time instants within each second. This non-uniformity can hinder model learning capability. Hence, we fit cubic splines for each dimension of the robot data- $(x, y, \theta, v, \omega)$ to represent continuous variations. The cubic splines for every dimension for each robot are then sampled at the top of every second to create a more uniform dataset at 1Hz. We have a total of over 250,000 sequences with each sequence consisting of 60 seconds of poses, linear and angular velocities for 10 robots.

Experiments: We perform an 80:20 train-to-val split of the data for our experiments. Effectively, 24 days of robot data are used for training and the remaining 6 are used for validation. This separation ensures validation is performed on unseen data with no mixing. We use the commonly used Average Displacement Error (ADE) metric for our evaluation with the time horizon T set to 60 seconds. The metric can be defined as $\text{ADE} = \frac{1}{NT}\sum_{i=1}^{N}\sum_{t=0}^{T-1}\sqrt{(\hat{x_i^t} - x_i^t)^2 + (\hat{x_i^t} - y_i^t)^2}$.

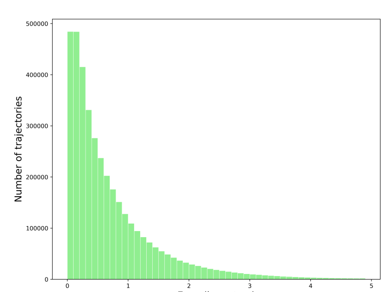

Figure 3: Distribution of validation errors from our neural CDE model

Results and Analysis: Some visual results from our neural CDE-based forecasting model are shown in Figure 2. The red arrows depict reconstructed validation poses, while the blue ones represent the ground truth. The arrows indicate direction of travel. It can be seen that the predicted poses match quite well with the ground truth, with accurate directions of the arrows. Furthermore, the size of each arrow is proportional to the velocity magnitude, showing strong correspondence between the dynamics of the predictions and ground truth. Although poses are predicted at 1Hz and can be sampled at any intermediate time, the predictions are visualized once every 5 seconds for clarity. Table 1 compares the average ADE for the neural

Method	ADE (5s)	ADE (60s)
NCDE (Only Pred Loss)	0.24 m	0.95 m
NCDE (Pred + Phy Loss)	**0.18 m**	**0.77 m**

Table 1: ADE scores across varied time horizons for 10 robots trained with and without a physics-informed loss

CDE model across short and longer time horizons for the model trained over 60 seconds, and the effect of an additional physics-informed loss function. While our neural CDE model works quite well producing low average errors below 1 m (in a 23 m average motion span), physics-informed losses can significantly help in producing trajectories that do not deviate as much from the ground truth, and their effect stays consistent for longer times. In Figure 3, we plot the distribution of error predictions for each robot trajectory. While the best ADE from Table 1 shows a value of 0.77 m, the actual distribution of forecasting errors show that for a large number of cases, the prediction errors are much lower, less than 0.4 m. Moreover, there are very few instances with large errors. We also quantify the runtime of our 3.29 million parameter AE-neural CDE model on a single Nvidia A10G GPU with 24GB memory. A batch of 2048 sequences, each with 10 robots and 60 seconds, takes less than 1 second for inference, paving the way for a parallelizable, fast and differentiable surrogate simulator.

5 Conclusion

In summary, we have developed a method that can model the joint latent dynamics of multiple robots in continuous time over long time horizons of 60 seconds. Our method uses neural controlled differential equations which incorporate goal velocities over time as part of the control path for the neural CDE. We explore the capabilities of these models for motion forecasting as opposed to more commonly used transformer/RNN-based forecasting models. Our results show that continuous-time methods with goal conditioning in terms of reference controls are highly promising for motion forecasting over long horizons. Furthermore, adding a physics-informed loss significantly improves the quality of rollouts. In future, we aim to scale this to a larger number of robots and explore GNN-based methods within the CDE that can explicitly capture interactions between agents.

References

[1] Mustafa Baniodeh, Kratarth Goel, Scott Ettinger, Carlos Fuertes, Ari Seff, Tim Shen, Cole Gulino, Chenjie Yang, Ghassen Jerfel, Dokook Choe, Rui Wang, Vinutha Kallem, Sergio Casas, Rami Al-Rfou, Benjamin Sapp, and Dragomir Anguelov. Scaling laws of motion forecasting and planning – a technical report, 2025. URL `https://arxiv.org/abs/2506.08228`.

[2] Nigamaa Nayakanti, Rami Al-Rfou, Aurick Zhou, Kratarth Goel, Khaled S. Refaat, and Benjamin Sapp. Wayformer: Motion forecasting via simple and efficient attention networks. In *2023 IEEE International Conference on Robotics and Automation (ICRA)*, pages 2980–2987, 2023. doi: 10.1109/ICRA48891.2023.10160609.

[3] Tim Salzmann, Boris Ivanovic, Punarjay Chakravarty, and Marco Pavone. Trajectron++: Dynamically-feasible trajectory forecasting with heterogeneous data. In *Computer Vision – ECCV 2020: 16th European Conference, Glasgow, UK, August 23–28, 2020, Proceedings, Part XVIII*, page 683–700, Berlin, Heidelberg, 2020. Springer-Verlag. ISBN 978-3-030-58522-8. doi: 10.1007/978-3-030-58523-5_40. URL `https://doi.org/10.1007/978-3-030-58523-5_40`.

[4] Ari Seff, Brian Cera, Dian Chen, Mason Ng, Aurick Zhou, Nigamaa Nayakanti, Khaled S. Refaat, Rami Al-Rfou, and Benjamin Sapp. Motionlm: Multi-agent motion forecasting as language modeling, 2023. URL `https://arxiv.org/abs/2309.16534`.

[5] Shaoshuai Shi, Li Jiang, Dengxin Dai, and Bernt Schiele. Mtr++: Multi-agent motion prediction with symmetric scene modeling and guided intention querying. *arXiv preprint arXiv:2306.17770*, 2023.

[6] Ricky T. Q. Chen, Yulia Rubanova, Jesse Bettencourt, and David Duvenaud. Neural ordinary differential equations. In *Proceedings of the 32nd International Conference on Neural Information Processing Systems*, NIPS'18, page 6572–6583, Red Hook, NY, USA, 2018. Curran Associates Inc.

[7] Yulia Rubanova, Ricky T. Q. Chen, and David Duvenaud. Latent odes for irregularly-sampled time series, 2019. URL `https://arxiv.org/abs/1907.03907`.

[8] George Em Karniadakis, Ioannis G. Kevrekidis, Lu Lu, Paris Perdikaris, Sifan Wang, and Liu Yang. Physics-informed machine learning. *Nature Reviews Physics*, 3(6):422–440, 2021. ISSN 25225820. doi: 10.1038/s42254-021-00314-5. URL `https://doi.org/10.1038/s42254-021-00314-5`.

[9] M. Raissi, P. Perdikaris, and G.E. Karniadakis. Physics-informed neural networks: A deep learning framework for solving forward and inverse problems involving nonlinear partial differential equations. *Journal of Computational Physics*, 378:686–707, 2019. ISSN 0021-9991. doi: https://doi.org/10.1016/j.jcp.2018.10.045. URL `https://www.sciencedirect.com/science/article/pii/S0021999118307125`.

[10] Somdatta Goswami, Ameya D. Jagtap, Hessam Babaee, Bryan T. Susi, and George Em Karniadakis. Learning stiff chemical kinetics using extended deep neural operators. *Computer Methods in Applied Mechanics and Engineering*, 419:116674, 2024. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2023.116674. URL `https://www.sciencedirect.com/science/article/pii/S0045782523007971`.

[11] Wang Xiao, Ting Gao, Kai Liu, Jinqiao Duan, and Meng Zhao. Fourier neural operator based fluid–structure interaction for predicting the vesicle dynamics. *Physica D: Nonlinear Phenomena*, 463:134145, 2024. ISSN 0167-2789. doi: https://doi.org/10.1016/j.physd.2024.134145. URL `https://www.sciencedirect.com/science/article/pii/S0167278924000964`.

[12] Nikola Kovachki, Zongyi Li, Burigede Liu, Kamyar Azizzadenesheli, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Neural operator: learning maps between function spaces with applications to pdes. *J. Mach. Learn. Res.*, 24(1), January 2023. ISSN 1532-4435.

[13] Zongyi Li, Nikola Kovachki, Kamyar Azizzadenesheli, Burigede Liu, Kaushik Bhattacharya, Andrew Stuart, and Anima Anandkumar. Fourier neural operator for parametric partial differential equations, 2021. URL `https://arxiv.org/abs/2010.08895`.

[14] Lu Lu, Pengzhan Jin, Guofei Pang, Zhongqiang Zhang, and George Em Karniadakis. Learning nonlinear operators via deeponet based on the universal approximation theorem of operators. *Nature Machine Intelligence*, 3(3), 03 2021. ISSN ISSN 2522-5839. doi: 10.1038/s42256-021-00302-5. URL `https://www.osti.gov/biblio/2281727`.

[15] Zongyi Li, Hongkai Zheng, Nikola Kovachki, David Jin, Haoxuan Chen, Burigede Liu, Kamyar Azizzadenesheli, and Anima Anandkumar. Physics-informed neural operator for learning partial differential equations, 2023. URL `https://arxiv.org/abs/2111.03794`.

[16] Mohammad Sadegh Eshaghi, Cosmin Anitescu, Manish Thombre, Yizheng Wang, Xiaoying Zhuang, and Timon Rabczuk. Variational physics-informed neural operator (vino) for solving partial differential equations. *Computer Methods in Applied Mechanics and Engineering*, 437:117785, 2025. ISSN 0045-7825. doi: https://doi.org/10.1016/j.cma.2025.117785. URL `https://www.sciencedirect.com/science/article/pii/S004578252500057X`.

[17] Tadbhagya Kumar, Anuj Kumar, and Pinaki Pal. A physics-informed autoencoder-neuralode framework (phy-chemnode) for learning complex fuel combustion kinetics. In *NeurIPS Machine Learning and the Physical Sciences Workshop*, 2024.

[18] Sulagna Sarkar, Anqi Ji, Zachary Jermain, Robert Lipton, Mark Brongersma, Kaushik Dayal, and Hae Young Noh. Physics-informed machine learning for inverse design of optical metamaterials. *Advanced Photonics Research*, 4(12), 10 2023. ISSN ISSN 2699-9293. doi: 10.1002/adpr.202300158. URL `https://www.osti.gov/biblio/2067627`.

[19] Lu Lu, Raphaël Pestourie, Wenjie Yao, Zhicheng Wang, Francesc Verdugo, and Steven G. Johnson. Physics-informed neural networks with hard constraints for inverse design. *SIAM Journal on Scientific Computing*, 43(6):B1105–B1132, 2021. doi: 10.1137/21M1397908. URL `https://doi.org/10.1137/21M1397908`.

[20] Zhongkai Hao, Songming Liu, Yichi Zhang, Chengyang Ying, Yao Feng, Hang Su, and Jun Zhu. Physics-informed machine learning: A survey on problems, methods and applications, 2023. URL `https://arxiv.org/abs/2211.08064`.

[21] James Donnelly, Alireza Daneshkhah, and Soroush Abolfathi. Physics-informed neural networks as surrogate models of hydrodynamic simulators. *Science of The Total Environment*, 912: 168814, 2024. ISSN 0048-9697. doi: https://doi.org/10.1016/j.scitotenv.2023.168814. URL `https://www.sciencedirect.com/science/article/pii/S0048969723074430`.

[22] X. I. A. Yang, S. Zafar, J.-X. Wang, and H. Xiao. Predictive large-eddy-simulation wall modeling via physics-informed neural networks. *Phys. Rev. Fluids*, 4:034602, Mar 2019. doi: 10.1103/PhysRevFluids.4.034602. URL `https://link.aps.org/doi/10.1103/PhysRevFluids.4.034602`.

[23] Patrick Kidger, James Morrill, James Foster, and Terry Lyons. Neural Controlled Differential Equations for Irregular Time Series. *Advances in Neural Information Processing Systems*, 2020.

[24] Nicholas Rhinehart, Rowan McAllister, Kris Kitani, and Sergey Levine. Precog: Prediction conditioned on goals in visual multi-agent settings, 2019. URL `https://arxiv.org/abs/1905.01296`.

[25] Alex Sherstinsky. Fundamentals of recurrent neural network (rnn) and long short-term memory (lstm) network. *Physica D: Nonlinear Phenomena*, 404:132306, March 2020. ISSN 0167-2789. doi: 10.1016/j.physd.2019.132306. URL `http://dx.doi.org/10.1016/j.physd.2019.132306`.

[26] James Morrill, Patrick Kidger, Lingyi Yang, and Terry Lyons. Neural controlled differential equations for online prediction tasks, 2021. URL `https://arxiv.org/abs/2106.11028`.

[27] Wenshuai Zhao, Jorge Peña Queralta, and Tomi Westerlund. Sim-to-real transfer in deep reinforcement learning for robotics: a survey. In *2020 IEEE Symposium Series on Computational Intelligence (SSCI)*, pages 737–744, 2020. doi: 10.1109/SSCI47803.2020.9308468.