

# Can Transformers Perform Low-Level Control In-Context?

**Ebonye Smith**

*EECS, UC Berkeley*

EBONYESMITH@BERKELEY.EDU

**Aidan Andrews**

*UIUC*

AIDANSA2@ILLINOIS.EDU

**Alex Frias**

*USC*

ALEXISFR@USC.EDU

**Ayush Pandey**

*EECS, UC Merced*

AYUSHPANDEY@UCMERCED.EDU

**Gireeja Ranade**

*EECS, UC Berkeley*

RANADE@EECS.BERKELEY.EDU

## Abstract

One of the remarkable abilities of Transformer-based models such as GPT-2 and GPT-3 is in-context learning. In this work, we investigate the ability of Transformer-models to perform “in-context” control of unstable nonlinear dynamical systems and act as a closed-loop controller. Specifically, we investigate a GPT-2 style controller for two classical control systems — the cartpole system and the acrobot. We find that Transformer models can simultaneously perform estimation and control and accurately predict modes for a switching controller. Furthermore, the Transformer-based controller can also stabilize some out-of-distribution test systems with fairly limited context information as compared to baselines data-driven controllers. We thus illustrate how transformer-based controllers can be used for data-driven control for non-linear control systems.

**Keywords:** deep learning, in-context learning, low-level control, stabilization

## 1. Introduction

Transformers have emerged as a powerful architecture for modeling sequential data, demonstrating success in auto-regressive prediction across various domains, e.g. language (Vaswani et al., 2023), vision (Dosovitskiy et al., 2021), and decision-making (Chen et al., 2021). Nonlinear low-level control, specifically for unstable dynamical systems, represents a particularly challenging case for an auto-regressive task. Unlike text or high-level planning, low-level control for unstable systems is sensitive to small perturbations and prediction errors, thus the transformer is required to make more precise predictions.

Additionally, transformers have illustrated abilities to perform in-context learning (ICL), which refers to the ability of the model during inference to generate an output by leveraging context examples (input-output pairs) and a new corresponding input query as a prompt. This capability is interesting because it allows the model to leverage demonstrations to adapt to new tasks without fine-tuning (Brown et al., 2020; Lieber et al., 2021; Rae et al., 2021; Black et al., 2022). Although recent work has demonstrated impressive ICL capabilities for high-level planning or mid-level control for robotic tasks (Fu et al., 2024), there remains a limited understanding of how transformers will perform for fine-grained, precise tasks like low-level stabilizing control for unstable, nonlinear

systems. In this work, we investigate the capability of transformers to perform as data-driven control models for unstable, nonlinear systems for two classical systems — the cartpole system (Åström and Murray, 2021, Ch.3) and the acrobot (Murray and Hauser, 1991).

In this paper, we show that a transformer-based model can successfully perform low-level control for the cartpole and the acrobot systems. With many training examples, the model can incorporate key information about the control strategy for the system into its weights and learn to stabilize the system even with very few context examples. At earlier training checkpoints, we find that the model can leverage the in-context examples to perform control for a substantial fraction of systems as well. A key step in getting these systems to successfully train included adding Gaussian noise to the training data to avoid the compounding error when using transformer models in closed loop.

## 2. Related Work

### 2.1. In-Context Learning for Control

ICL is an effective model capability to evaluate downstream adaptation to new tasks without requiring gradient updates. Early studies investigated the fundamental capabilities of transformers for ICL in controlled settings, such as linear regression (Garg et al., 2022; Raventós et al., 2023). There has been tremendous work in understanding in-context learning for language models (Olsson et al., 2022; Akyürek et al., 2024; Xie et al., 2021; Lin and Lee, 2024; Wei et al., 2023; Yin and Steinhardt, 2025; Wies et al., 2023; Pan et al., 2023; Min et al., 2022). In this paper, we focus on sequential decision making in the context of closed-loop control. Raparthy et al. (2023) further specifies that ICL in sequential decision-making requires careful training on datasets with diverse trajectories and stochasticity to influence the ability of the model to generalize to novel tasks.

Du et al. (2023) consider the question: can learn to predict Gauss-Markov processes in-context? They show that for certain cases, this is indeed possible. Daniels et al. (2025) similarly investigates a task that involves dynamical systems prediction along with associative recall. However, each of these tasks focuses on a underlying system that is linear, and where the prediction happens in open-loop. In contrast, our current papers considers a non-linear control task, where the transformer must iteratively perform actions, similar to the Decision Transformer Framework Chen et al. (2021). Specifically, Chen et al. (2021) and Janner et al. (2021) develop frameworks that abstract RL as a sequence modeling problem, with the goal of producing a sequence of actions that leads to a sequence of high rewards. Building on transformer-based RL representation, several works have used this framework to address different aspects of control, such as reward-agnostic control representations (Sun et al., 2023), model predictive control warm-starting (Celestini et al., 2024), and system identification (Zhang et al., 2025b). Xu et al. (2022) introduced the concept of prompting these RL-based transformers to investigate ICL capabilities for control.

We build on this as well as work such as Zhang et al. (2025a) that explores fine-tuning a language foundation model to perform closed loop control, as well as Wang et al. (2024a) that explores natural language-based prompting. Other works in this vein include Yu et al. (2023) that demonstrated how natural language descriptions can be translated into reward functions, and Mandi et al. (2023) that explored multi-robot collaboration through text-based task coordination between agents and motion planning. Other works investigate action generation directly based on dynamically updated trajectory context (Zhu et al., 2024; Wang et al., 2024b). Other approaches rely on prompt-engineered LLMs for robotics reasoning (Yin et al., 2025; Di Palo and Johns, 2024) and domain-adapted vision-language models for manipulation tasks (Li et al., 2023).

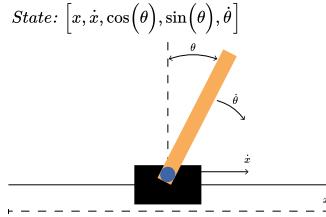


Figure 1: Cartpole system

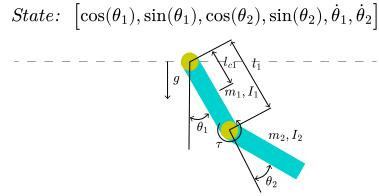


Figure 2: Acrobot system

## 2.2. Data-driven Control

Before the emergence of deep learning techniques, classical control theory consisted of more interpretable data-driven methods for both system identification and controller synthesis. System identification methods include Sparse Identification of Nonlinear Dynamics (SINDy) (Brunton et al., 2016), data-driven Koopman analysis (Kutz et al., 2016; Rowley et al., 2009; Schmid, 2010), multi-layer perceptron networks (MLPs) (Su and McAvoy, 1993), and Recurrent Neural Networks (RNNs) (Al Seyab and Cao, 2008). SINDy and data-driven Koopman Analysis, also referred to as Dynamic Mode Decomposition (DMD), provide linear representations of nonlinear dynamics that can be integrated with classical control frameworks, while MLPs and RNNs offer expressive nonlinear representations that can directly approximate system dynamics from data. Data-driven approaches that directly synthesize a control policy include imitation learning and RL. In imitation learning, control policies are trained to reproduce expert demonstrations through supervised learning (Pomerleau, 1988; Schaal, 1999; Argall et al., 2009). RL extends this paradigm by optimizing policies through trial-and-error interactions with a dynamic environment motivated by achieving a large reward (Sutton et al., 1998; Asada et al., 1999; Peters and Schaal, 2008).

## 3. Methods

### 3.1. General Setup

We consider the problem of data-driven low-level control for unstable, nonlinear dynamical systems, focusing on two canonical benchmark systems: a cart-pole and a 2-link underactuated robotic system called acrobot. The objective is to train a model that can “in-context” control the system, given the start of a trajectory. Particularly, the transformer model must be able to stabilize systems whose parameters were not seen during training. We train two separate transformer models, one for the cartpole and one for the acrobot. Each model is a decoder only transformer model similar to the GPT-2 architecture, consisting of 12 layers, 8 attention heads, a 256-dimensional embedding space, and absolute position encoding. We use AdamW optimizer with a learning rate of  $10^{-4}$ , weight decay of  $10^{-2}$ , batch size of 64, and no dropout.

### 3.2. Trajectory Generation

#### 3.2.1. CARTPOLE SYSTEM

Our goal is to study if the transformer can learn to control nonlinear systems. For this we consider a dataset  $\mathcal{D}_{\text{cart}}$  consisting of approximately 3 million trajectories. Each trajectory ( $\mathcal{T}$ ) differs by variation of physical parameters (mass and length of the pole) and initial conditions. The cart mass is

fixed at  $m_c = 2$  kg. The pole mass and pole length are sampled as  $m_p \sim \text{Uniform}(0.3, 1.0)$ ,  $l_p \sim \text{Uniform}(1.0, 1.5)$ , where the mass is in kilograms and length in meters. Similarly, the initial condition is defined as  $x_{\text{init}} = 0$ ,  $\dot{x}_{\text{init}} = 0$ ,  $\theta_{\text{init}} \sim \text{Uniform}(\frac{\pi}{2}, \frac{3\pi}{2})$  rad,  $\dot{\theta}_{\text{init}} \sim \text{Uniform}(-1, 1)$  rad/s. We augment the state as  $s_i = [x, \dot{x}, \cos(\theta), \sin(\theta), \dot{\theta}]$ , where  $x$  is the cart position and  $\theta$  is the angle of the pole with respect to the upward position. Each  $\mathcal{T}$  is generated using an energy-based swing-up and LQR stabilization switching controller (Tedrake, 2023) and the Runge-Kutta method (RK4) (Butcher, 1996) to solve the ordinary differential equation (ODE) to compute the next state based on the control action. Specifically, the controller produces a control input ( $a_i$ ) corresponding to the force applied to the cart and a controller mode label ( $m_i$ ), and RK4 uses the dynamics ODE and the control action to produce the updated state ( $s_{i+1}$ ) and the updated distance from the goal position ( $d_{i+1}$ ) which is  $s_{\text{goal}} = [0, 0, 1, 0, 0]$ , the upright position. We restricted the controller mode ( $m_i$ ) to switch back to swing-up mode once it entered the LQR mode to make the controller more successful at stabilizing the system as system parameters varied.

Although the switching controller is suboptimal compared to other sophisticated controllers, it is the least computationally expensive to use in generating system trajectories. For example, methods like differential dynamic programming (Mayne, 1966) require solving an optimization problem to obtain each trace, which is not feasible when millions of traces are required for training of the transformer model. The switching controller has fixed parameters tuned to handle the specified ranges of masses and lengths, thus, trajectory generation was vectorized for a more reasonable data generation time.

### 3.2.2. ACROBOT SYSTEM

We generate a dataset  $\mathcal{D}_{\text{acro}}$  consisting of 22 million trajectories. The parameters, link lengths ( $l_{l_1}, l_{l_2}$ ) in meters and link masses ( $m_{l_1}, m_{l_2}$ ) in kg are varied across trajectories as follows:  $l_{l_1} \sim \text{Uniform}(0.5, 1.0)$ ,  $l_{l_2} \sim \text{Uniform}(1.0, 1.5)$ ,  $m_{l_1} \sim \text{Uniform}(1.5, 2.0)$ ,  $m_{l_2} \sim \text{Uniform}(1.0, 1.5)$ . We augment the state  $s_i$  as  $s_i = [\cos(\theta_1), \sin(\theta_1), \cos(\theta_2), \sin(\theta_2), \dot{\theta}_1, \dot{\theta}_2]$ , where  $\theta_1$  denotes the angle of the first link with respect to the downward vertical position, and  $\theta_2$  is the angle of the second link relative to the first link. Again, each  $\mathcal{T}$  is generated using an energy-based swing-up and LQR stabilization switching controller (Spong, 2002; Tedrake, 2023) and RK4 to produce the next state ( $s_{i+1}$ ) and the updated distance from the goal position ( $d_{i+1}$ ) which is  $s_{\text{goal}} = [-1, 0, 1, 0, 0, 0]$ . Because the acrobot is a much more complex system than the cartpole, designing a general switching controller that could handle varying system parameters proved to be difficult. We modified the controller in Spong (2002) to add motion smoothing, tapering off near the upright goal position, and hysteresis to prevent abrupt switching between controller modes.

Given that the acrobot ground-truth controller is suboptimal, stabilization took upwards of 3500 time steps, which is inappropriate for training the transformer in a timely manner. Thus, we extracted shorter traces that feature a substantial swing-up and LQR stabilization. These truncations naturally introduced diversity of the initial conditions of traces.

### 3.3. Trajectory Generation with Gaussian Noise

For both the cartpole and the acrobot, the ideal closed-loop swing-up and LQR switching-controller will produce  $\mathcal{T}$  where  $s_i$  converges to  $s_{\text{goal}}$  and  $a_i$  converges to 0. However, methods based on imitation learning (i.e. training a transformer) are exposed to severe error compounding from which it cannot recover at inference when applied in closed-loop (Rajaraman et al., 2020). This is because

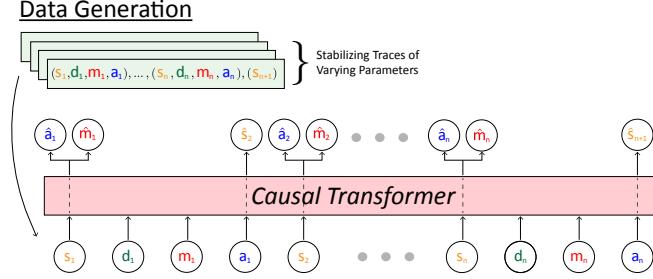


Figure 3: Training paradigm for the transformer. The top shows the stabilizing training dataset. As input the model takes the sequence as input. Each  $s_i$  is used to predict  $\hat{a}_i$  and  $\hat{m}_i$ , and each  $\hat{a}_i$  is used to predict  $\hat{s}_{i+1}$

during training the model is subject to the distribution of expert demonstrations, but at inference, it is subject to its own distribution which is technically out of distribution. Iterative fine-tuning algorithms, e.g. DAGGER (Ross et al., 2011; Celestini et al., 2024), seek to address this issue by fine-tuning the model on corrected exploratory traces to encourage the model to learn adjustments to its errors.

We leverage a much cheaper alternative to address this issue by simply injecting Gaussian noise into the state once the controller switches to LQR. This instead exposes the model to erroneous data featuring a noisy version of the state rather than the true state, and LQR is forced to generate corrective actions that the transformer can learn for. This also produced an additional benefit of handling the order of magnitude issue because transformers struggle when tasked with predicting values that range from a large magnitude to a small magnitude, like control traces for stabilization range from a large magnitude and converge to values on the order of  $10^{-3}$ .

### 3.4. Training Procedure

The transformer is trained using teacher-forcing on all three prediction heads  $(a_i, m_i, s_i)$  as shown in Figure 3. The combined loss function is  $\mathcal{L} = \|\hat{a} - a\|_2^2 + \|\hat{s} - s\|_2^2 + \mathcal{L}_{\text{BCE}}(\hat{m}, m)$  where  $\hat{a}$ ,  $\hat{s}$ , and  $\hat{m}$  denote predicted control, state, and controller mode label, respectively, and  $\mathcal{L}_{\text{BCE}}$  denotes binary cross-entropy with logits loss.

## 4. Results

### 4.1. Inference

We conduct in-distribution and OOD experiments. In-distribution experimental systems are drawn according to distributions described in sections 3.2.1 and 3.2.2. OOD experimental systems are drawn from the following distributions:

$$\begin{aligned}
 \textbf{Cartpole: } & m_c = 2, \quad m_p \sim \text{Uniform}(1.1, 2.0), \quad l_p \sim \text{Uniform}(1.6, 2.1), \\
 \textbf{Acrobot: } & l_{l_1} \sim \text{Uniform}(1.0, 1.5), \quad l_{l_2} \sim \text{Uniform}(1.5, 2.0), \\
 & m_{l_1} \sim \text{Uniform}(1.5, 2.0), \quad m_{l_2} \sim \text{Uniform}(1.0, 1.5).
 \end{aligned}$$

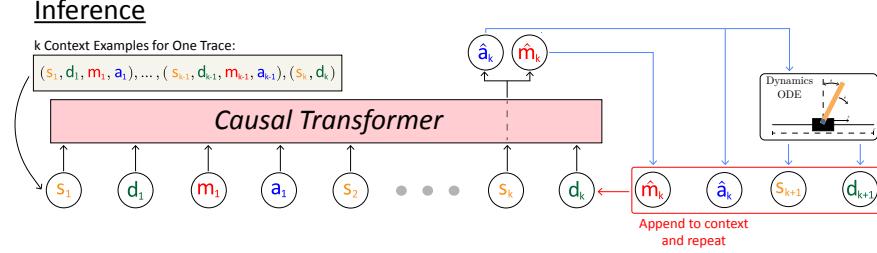


Figure 4: Closed-loop inference procedure. The input sequence shows  $k$  context examples  $\mathcal{T}_k$  from a system with parameters not seen during training. The model predicts  $\hat{a}_k$  and  $\hat{m}_k$ .  $\hat{a}_k$  is used to generate the next state  $s_{k+1}$  and  $d_{k+1}$ .

We generate reference trajectories for comparison using the same swing-up and LQR switching controller as for generating training data. During inference, the transformer model is used as a closed-loop controller. Specifically, the model receives  $k$  context examples  $\mathcal{T}_k = ((s_1, d_1, m_1, a_1), \dots, (s_{k-1}, d_{k-1}, m_{k-1}, a_{k-1}), (s_k, d_k, ?, ?))$  from the reference trajectory. Using  $\mathcal{T}_k$ , the model predicts action  $\hat{a}_k$  and controller mode label  $\hat{m}_k$ . Then,  $\hat{a}_k$  is input into the dynamics ODE, and  $s_{k+1}$  and  $(d_{k+1})$  are computed, and the sequence  $(\hat{m}_k, \hat{a}_k, s_{k+1}, d_{k+1})$  is appended to  $\mathcal{T}_k$ , and this process is repeated until the maximum sequence is achieved. The full process is illustrated in Figure 4. For simulation, we use the OpenAI gym environment (Brockman et al., 2016).

## 4.2. Cartpole System Analysis

### 4.2.1. PERFORMANCE OF MODEL VS REFERENCE TRAJECTORIES

We demonstrate a successfully stabilizing trajectory for the cartpole system in Figure 5. Each subplot features states,  $s_i = [x, \dot{x}, \theta, \dot{\theta}]$ , control actions  $a_i$ , and controller mode labels  $m_i$  plotted versus time for both the reference trace (in cyan) and the transformer trace (in red). In this example, the transformer is given 20 examples in-context. Within each subplot, we observe that although the transformer does not perfectly track the reference trajectory at all times, the control label switches appropriately, and stabilization is achieved as indicated by  $\dot{x}$ ,  $\theta$ , and  $\dot{\theta}$  converging to zero. Note that context examples  $\mathcal{T}_{20}$  do not feature LQR data, and this is consistent for all test evaluations.

### 4.2.2. ANALYSIS OF TRAINING CHECKPOINTS

Emergence of ICL capabilities requires careful selection of the appropriate training checkpoint that illustrates successful stabilization as the number of context examples  $k$  in  $\mathcal{T}_k$  increases. We randomly sample 100 in-distribution test cartpole systems, and we evaluate the empirical stabilization success rate across various training checkpoints. A trajectory is characterized as successfully stabilized if  $|\sin(\theta)| < 0.2$  and  $|\dot{\theta}| < 0.3$  for the last 20 states in the trajectory. Figure 6 displays the fraction of systems that were stabilized versus training step for context lengths 1 (green), 6 (orange), 12 (purple), and 20 (brown). We observe that as training progresses, the fraction of systems that are stabilized increases. We also observe that for a given training step, the stability fraction increases as the number of context examples increase.

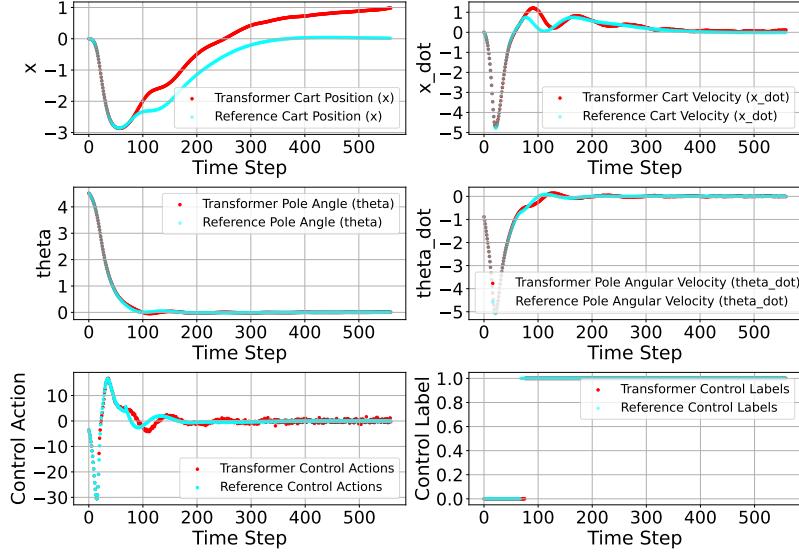


Figure 5: Cartpole comparison of Reference Trajectory (cyan) and Transformer Trajectory given 20 context examples (red). Subplots features states  $s_i = [x, \dot{x}, \theta, \dot{\theta}]$ , control actions  $a_i$ , and controller mode labels  $m_i$  plotted against time. In each subplot, we see that transformer trajectory differs from the reference trajectory, but both trajectories are stabilizing.

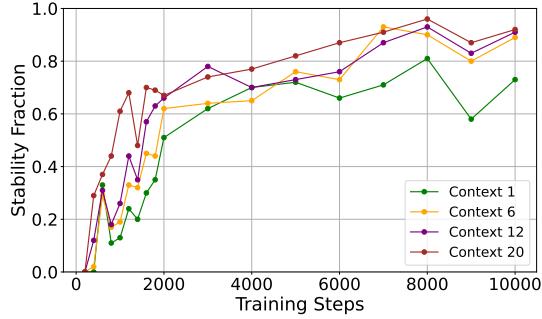


Figure 6: Cartpole stability fraction vs. training step for context lengths 1, 6, 12, 20 for 100 in-distribution systems.

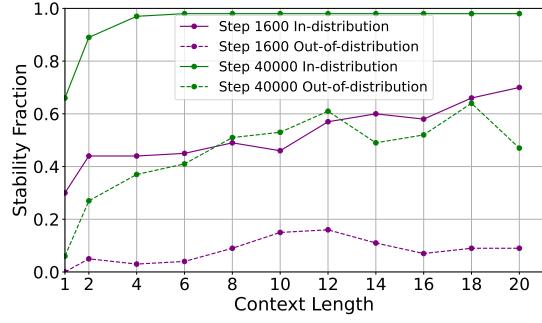


Figure 7: Cartpole stability fraction vs. context length for training steps 1600 and 40000 using 100 in-distribution and OOD test systems.

#### 4.2.3. PERFORMANCE OF IN-CONTEXT LEARNING

**In-distribution Analysis** As stated in Section 4.2.2, the cartpole transformer exhibits ICL capabilities for in-distribution test data. Based on Figure 6, we select training step 1600 as a training step of interest, since it shows the largest impact of context, i.e. the largest jump in ability to stabilize between context 1 (30% systems stabilized) and context 20 (70% systems stabilized). We investigate training checkpoint 1600 more closely by additionally sampling 100 OOD test systems, and evaluating the transformer at this checkpoint across various contexts. Figure 7 shows the stability versus context length for training step 1600 (purple) for in-distribution test systems (solid)

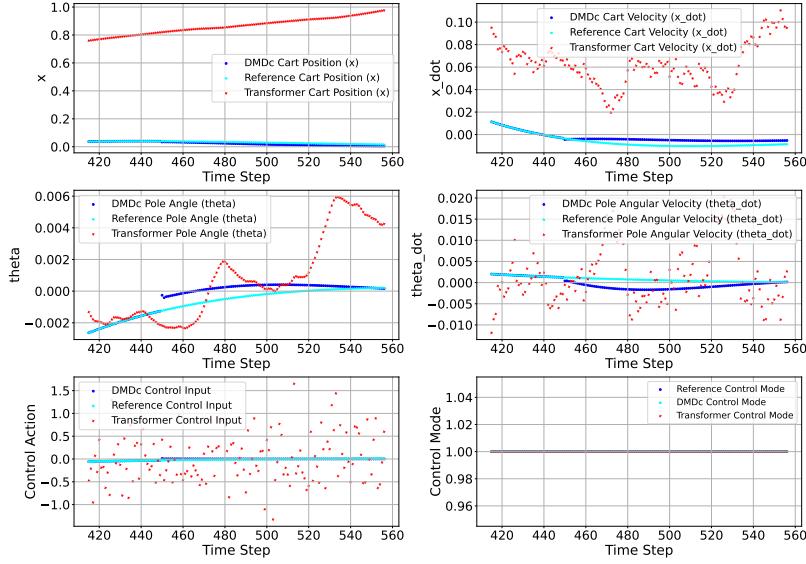


Figure 8: DMDc of LQR phase using  $k = 35$  context examples to approximate the system dynamics. Subplots feature states,  $s_i = [x, \dot{x}, \theta, \dot{\theta}]$ , control actions  $a_i$ , and controller mode labels  $m_i$ , respectively, plotted versus time. DMDc tracks the reference trajectory more closely than the transformer does, but the transformer achieves stabilization regardless as indicated by  $\theta$  and  $\dot{\theta}$  near 0.

and OOD test systems (dashed). We observe that although the model displays ICL capabilities for in-distribution test systems, it struggles to successfully stabilize OOD test systems.

**Out-of-distribution Analysis** To further investigate the model struggling with OOD test systems, we evaluate model performance at a much later checkpoint to see if ICL capabilities for OOD systems emerge as training continues. In Figure 7, we show the stability fraction versus context length for training step 40000 (green) for in-distribution test systems (solid) and OOD test systems (dashed). The first observation to make is that the transformer needs fewer in-context examples to stabilize in-distribution test systems as indicated by its achievement of stability fraction 0.9 with only 2 context examples (solid green line). The second observation is that ICL capabilities seem to emerge for OOD test systems as the dashed green line displays  $< 0.1$  stability fraction with 1 context example and  $> 0.6$  stability fraction for 18 context examples. This shows that the transformer model is able to incorporate high-level principles of control for the system into its weights and apply these for OOD examples.

**Baseline Comparison** Finding a data-driven system identification baseline that required as few context examples that our transformers required proved to be very difficult. Our models are able to learn the correct switching conditions between swing-up and LQR during training, thus, seeing LQR data in-context is not necessary for successful stabilization. However, this expectation is unrealistic for a classical data-driven system ID method. Therefore, we focus our baseline only on the LQR phase. We compare the cartpole model during the LQR phase to Dynamic Mode Decomposition with Control (DMDc) (Proctor et al., 2016), an extension of DMD that utilizes both measurements of the system and applied external control ( $k = 35$  examples) to fit an approximate linear model for the dynamics. We use this approximate model in conjunction with our ground-

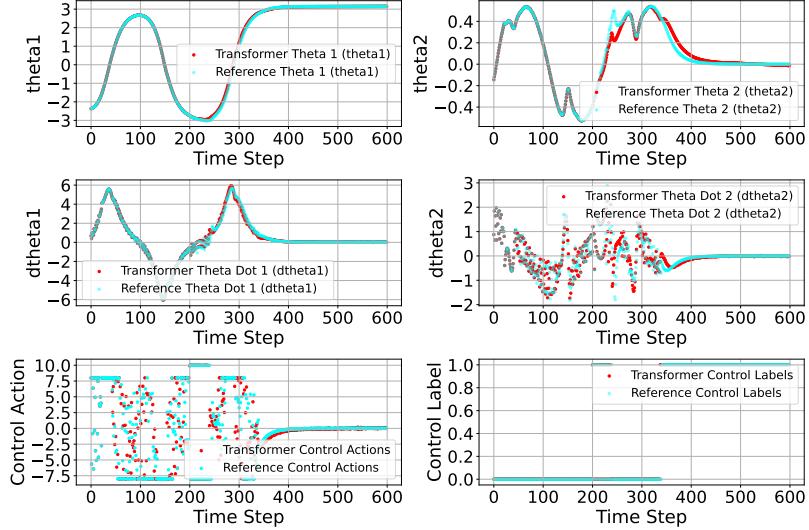


Figure 9: Acrobot comparison of Reference Trajectory and Transformer Trajectory given 40 context examples. Subplots features states  $s_i = [\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2]$ , control actions  $a_i$ , and controller mode labels  $m_i$  plotted against time. In each subplot, we see that transformer tracks the reference trajectory very closely, and both trajectories are stabilizing.

truth switching controller to roll out the predicted trajectory. Figure 8 shows a trajectory example comparing the cartpole transformer model (red), DMDc (dark blue), and the reference trajectory (cyan) for states,  $s_i = [x, \dot{x}, \theta, \dot{\theta}]$ , control actions  $a_i$ , and controller mode labels  $m_i$ , respectively, plotted versus time. Note that they y-axis scale has been changed (as compared to Figure 5 to highlight the small differences among the trajectories which would otherwise be invisible. We observe that although DMDc tracks the reference trajectory better than the transformer model in all subplots, the transformer still achieves stabilization.

### 4.3. Acrobot System

#### 4.3.1. PERFORMANCE OF MODEL VS REFERENCE TRAJECTORIES

Figure 9 shows an example of a successfully stabilized trajectory for the acrobot. Each subplot displays states  $s_i = [\theta_1, \theta_2, \dot{\theta}_1, \dot{\theta}_2]$ , control actions  $a_i$ , and controller mode labels  $m_i$ , respectively, plotted versus time for the reference trace (cyan) and the transformer model (red). We observe that on each subplot, the transformer trajectory tracks the reference trajectory closely, and both traces stabilize.

#### 4.3.2. ANALYSIS OF TRAINING CHECKPOINTS

We randomly sample 100 in-distribution acrobot test systems, and we evaluate the stability fraction across various training checkpoints for contexts 1 (green), 10 (orange), 30 (purple), and 40 (brown). A trajectory is characterized as successfully stabilized if  $|\sin(\theta_1)| < 0.2$  and  $|\sin(\theta_2)| < 0.3$  for the last 20 states in the trajectory. In Figure 10, we observe that as training progresses, the stability fraction increases for each context, and checkpoint 230,000 is the best checkpoint where the model leverages context to aid with stabilization as indicated.

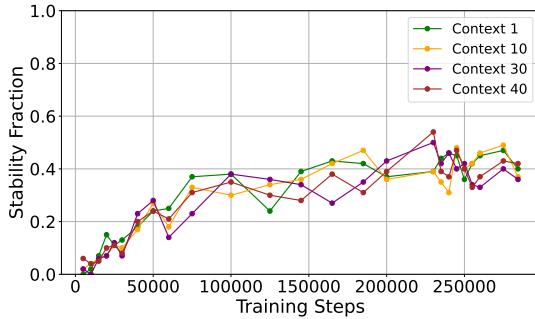


Figure 10: Acrobot stability fraction vs. training step for context lengths 1, 10, 30, 40 for 100 in-distribution systems.

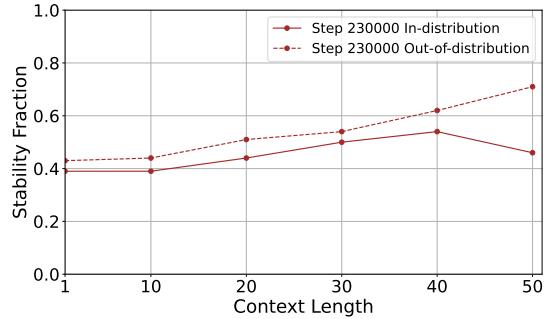


Figure 11: Acrobot stability fraction vs. context length for training step 230,000 using 100 in-distribution and OOD test systems.

#### 4.3.3. PERFORMANCE OF IN-CONTEXT LEARNING

We evaluate checkpoint 230,000 of the acrobot using 100 in-distribution and OOD test systems. As shown by the solid line in Figure 11, the acrobot model leverages context to aid with stabilization up until context 40 for in-distribution test systems, and performance degrades slightly with 50 context examples. For OOD test systems, as indicated by the dashed line, the acrobot model performs slightly better than on in-distribution test systems, and context is leveraged to aid with stabilization as the model achieves a 0.7 stability fraction using 50 context examples.

## 5. Conclusion

We investigate in-context learning capabilities of a transformer model for low-level stabilizing control for two classical unstable nonlinear dynamical systems, the cartpole and the acrobot. While the Transformer requires a large amount of data for training, after seeing these examples it is able to incorporate high-level principles of control into its weight and stabilize both unseen in-distribution and out-of-distribution systems. By simultaneously performing estimation and control, this paradigm eliminates the need for a separate system identification step. Thus, we show that Transformer-based models are a viable option for performing low-level data-driven control. Much further work is required to fully investigate this possibility. Specifically, can a model be trained to be able to control multiple systems, e.g. can a single transformer model do in-context control for both the cartpole and the acrobot? Such investigations would lay the path towards a foundation model for control. Safety remains an important consideration, and being able to provide performance certificates would be required for deployment in safety-critical scenarios.

## Acknowledgments

This material is based upon work supported by the National Science Foundation Graduate Research Fellowship under Grant No. DGE-2146752. This work was also supported by NSF CAREER ECCS2240031 and 2023 CITRIS Institute seed grant. We also thank Dr. Claire Tomlin from UC Berkeley for her valuable input on this paper.

## References

- Ekin Akyürek, Bailin Wang, Yoon Kim, and Jacob Andreas. In-context language learning: Architectures and algorithms. *arXiv preprint arXiv:2401.12973*, 2024.
- RK Al Seyab and Yi Cao. Nonlinear system identification for predictive control using continuous time recurrent neural networks and automatic differentiation. *Journal of Process Control*, 18(6): 568–581, 2008.
- Brenna D Argall, Sonia Chernova, Manuela Veloso, and Brett Browning. A survey of robot learning from demonstration. *Robotics and Autonomous Systems*, 57(5):469–483, 2009.
- Minoru Asada, Eiji Uchibe, and Koh Hosoda. Cooperative behavior acquisition for mobile robots in dynamically changing real worlds via vision-based reinforcement learning and development. *Artificial Intelligence*, 110(2):275–292, 1999.
- Karl Johan Åström and Richard Murray. *Feedback Systems: An Introduction for Scientists and Engineers*. Princeton University Press, 2021.
- Sidney Black, Stella Biderman, Eric Hallahan, Quentin Anthony, Leo Gao, Laurence Golding, Horace He, Connor Leahy, Kyle McDonell, Jason Phang, et al. Gpt-neox-20b: An open-source autoregressive language model. In *Proceedings of BigScience Episode# 5—Workshop on Challenges & Perspectives in Creating Large Language Models*, pages 95–136, 2022.
- Greg Brockman, Vicki Cheung, Ludwig Pettersson, Jonas Schneider, John Schulman, Jie Tang, and Wojciech Zaremba. OpenAI Gym, 2016.
- Tom Brown, Benjamin Mann, Nick Ryder, Melanie Subbiah, Jared D Kaplan, Prafulla Dhariwal, Arvind Neelakantan, Pranav Shyam, Girish Sastry, Amanda Askell, et al. Language models are few-shot learners. *Advances in Neural Information Processing Systems*, 33:1877–1901, 2020.
- Steven L Brunton, Joshua L Proctor, and J Nathan Kutz. Discovering governing equations from data by sparse identification of nonlinear dynamical systems. *Proceedings of the National Academy of Sciences*, 113(15):3932–3937, 2016.
- John Charles Butcher. A history of runge-kutta methods. *Applied Numerical Mathematics*, 20(3): 247–260, 1996.
- Davide Celestini, Daniele Gammelli, Tommaso Guffanti, Simone D’Amico, Elisa Capello, and Marco Pavone. Transformer-Based Model Predictive Control: Trajectory Optimization via Sequence Modeling. *IEEE Robotics and Automation Letters*, 9(11):9820–9827, November 2024. ISSN 2377-3766. doi: 10.1109/LRA.2024.3466069.
- Lili Chen, Kevin Lu, Aravind Rajeswaran, Kimin Lee, Aditya Grover, Misha Laskin, Pieter Abbeel, Aravind Srinivas, and Igor Mordatch. Decision transformer: Reinforcement learning via sequence modeling. *Advances in Neural Information Processing Systems*, 34:15084–15097, 2021.
- Sultan Daniels, Dylan Davis, Dhruv Gautam, Wentinn Liao, Gireeja Ranade, and Anant Sahai. Different simultaneous mechanisms for in-context recall have distinct learning dynamics. In *Workshop on High-dimensional Learning Dynamics*, 2025.

Norman Di Palo and Edward Johns. Keypoint action tokens enable in-context imitation learning in robotics. *arXiv preprint arXiv:2403.19578*, 2024.

Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale, 2021. URL <https://arxiv.org/abs/2010.11929>.

Zhe Du, Haldun Balim, Samet Oymak, and Necmiye Ozay. Can transformers learn optimal filtering for unknown systems? *IEEE Control Systems Letters*, 7:3525–3530, 2023. doi: 10.1109/LCSYS.2023.3335318.

Letian Fu, Huang Huang, Gaurav Datta, Lawrence Yunliang Chen, William Chung-Ho Panitch, Fangchen Liu, Hui Li, and Ken Goldberg. In-context imitation learning via next-token prediction. *arXiv preprint arXiv:2408.15980*, 2024.

Shivam Garg, Dimitris Tsipras, Percy S Liang, and Gregory Valiant. What can transformers learn in-context? a case study of simple function classes. *Advances in neural information processing systems*, 35:30583–30598, 2022.

Michael Janner, Qiyang Li, and Sergey Levine. Offline reinforcement learning as one big sequence modeling problem. In M. Ranzato, A. Beygelzimer, Y. Dauphin, P.S. Liang, and J. Wortman Vaughan, editors, *Advances in Neural Information Processing Systems*, volume 34, pages 1273–1286. Curran Associates, Inc., 2021.

J Nathan Kutz, Steven L Brunton, Bingni W Brunton, and Joshua L Proctor. *Dynamic mode decomposition: data-driven modeling of complex systems*. SIAM, 2016.

Xinghang Li, Minghuan Liu, Hanbo Zhang, Cunjun Yu, Jie Xu, Hongtao Wu, Chilam Cheang, Ya Jing, Weinan Zhang, Huaping Liu, et al. Vision-language foundation models as effective robot imitators. *arXiv preprint arXiv:2311.01378*, 2023.

Opher Lieber, Or Sharir, Barak Lenz, and Yoav Shoham. Jurassic-1: Technical details and evaluation. *White Paper. AI21 Labs*, 1(9):1–17, 2021.

Ziqian Lin and Kangwook Lee. Dual operating modes of in-context learning, 2024. URL <https://arxiv.org/abs/2402.18819>.

Zhao Mandi, Shreeya Jain, and Shuran Song. Roco: Dialectic multi-robot collaboration with large language models, 2023. URL <https://arxiv.org/abs/2307.04738>.

David Mayne. A second-order gradient method for determining optimal trajectories of non-linear discrete-time systems. *International Journal of Control*, 3(1):85–95, 1966.

Sewon Min, Xinxi Lyu, Ari Holtzman, Mikel Artetxe, Mike Lewis, Hannaneh Hajishirzi, and Luke Zettlemoyer. Rethinking the role of demonstrations: What makes in-context learning work?, 2022. URL <https://arxiv.org/abs/2202.12837>.

Richard M Murray and John Edmond Hauser. *A case study in approximate linearization: The ac-robot example*. Electronics Research Laboratory, College of Engineering, University of California Berkeley, 1991.

Catherine Olsson, Nelson Elhage, Neel Nanda, Nicholas Joseph, Nova DasSarma, Tom Henighan, Ben Mann, Amanda Askell, Yuntao Bai, Anna Chen, et al. In-context learning and induction heads. *arXiv preprint arXiv:2209.11895*, 2022.

Jane Pan, Tianyu Gao, Howard Chen, and Danqi Chen. What in-context learning "learns" in-context: Disentangling task recognition and task learning, 2023. URL <https://arxiv.org/abs/2305.09731>.

Jan Peters and Stefan Schaal. Reinforcement learning of motor skills with policy gradients. *Neural Networks*, 21(4):682–697, 2008. ISSN 0893-6080. doi: 10.1016/j.neunet.2008.02.003.

Dean A Pomerleau. Alvinn: An autonomous land vehicle in a neural network. *Advances in Neural Information Processing Systems*, 1, 1988.

Joshua L Proctor, Steven L Brunton, and J Nathan Kutz. Dynamic mode decomposition with control. *SIAM Journal on Applied Dynamical Systems*, 15(1):142–161, 2016.

Jack W Rae, Sebastian Borgeaud, Trevor Cai, Katie Millican, Jordan Hoffmann, Francis Song, John Aslanides, Sarah Henderson, Roman Ring, Susannah Young, et al. Scaling language models: Methods, analysis & insights from training gopher. *arXiv preprint arXiv:2112.11446*, 2021.

Nived Rajaraman, Lin Yang, Jiantao Jiao, and Kannan Ramchandran. Toward the fundamental limits of imitation learning. *Advances in Neural Information Processing Systems*, 33:2914–2924, 2020.

Sharath Chandra Raparthi, Eric Hambro, Robert Kirk, Mikael Henaff, and Roberta Raileanu. Generalization to new sequential decision making tasks with in-context learning. *arXiv preprint arXiv:2312.03801*, 2023.

Allan Raventós, Mansheej Paul, Feng Chen, and Surya Ganguli. Pretraining task diversity and the emergence of non-bayesian in-context learning for regression. *Advances in Neural Information Processing Systems*, 36:14228–14246, 2023.

Stéphane Ross, Geoffrey Gordon, and Drew Bagnell. A reduction of imitation learning and structured prediction to no-regret online learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, pages 627–635. JMLR Workshop and Conference Proceedings, 2011.

Clarence W Rowley, Igor Mezić, Shervin Bagheri, Philipp Schlatter, and Dan S Henningson. Spectral analysis of nonlinear flows. *Journal of Fluid Mechanics*, 641:115–127, 2009.

Stefan Schaal. Is imitation learning the route to humanoid robots? *Trends in Cognitive Sciences*, 3(6):233–242, 1999.

Peter J Schmid. Dynamic mode decomposition of numerical and experimental data. *Journal of Fluid Mechanics*, 656:5–28, 2010.

Mark W Spong. The swing up control problem for the acrobot. *IEEE Control Systems Magazine*, 15(1):49–55, 2002.

- Hong Te Su and Thomas J McAvoy. Integration of multilayer perceptron networks and linear dynamic models: a hammerstein modeling approach. *Industrial & Engineering Chemistry Research*, 32(9):1927–1936, 1993.
- Yanchao Sun, Shuang Ma, Ratnesh Madaan, Rogerio Bonatti, Furong Huang, and Ashish Kapoor. Smart: Self-supervised multi-task pretraining with control transformers, 2023. URL <https://arxiv.org/abs/2301.09816>.
- Richard S Sutton, Andrew G Barto, et al. *Reinforcement Learning: An Introduction*, volume 1. MIT Press Cambridge, 1998.
- Russ Tedrake. *Underactuated Robotics*. 2023. URL <https://underactuated.csail.mit.edu>.
- Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Łukasz Kaiser, and Illia Polosukhin. Attention is all you need, 2023. URL <https://arxiv.org/abs/1706.03762>.
- Yen-Jen Wang, Bike Zhang, Jianyu Chen, and Koushil Sreenath. Prompt a robot to walk with large language models. In *2024 IEEE 63rd Conference on Decision and Control (CDC)*, pages 1531–1538. IEEE, 2024a.
- Yen-Jen Wang, Bike Zhang, Jianyu Chen, and Koushil Sreenath. Prompt a Robot to Walk with Large Language Models, October 2024b.
- Jerry Wei, Jason Wei, Yi Tay, Dustin Tran, Albert Webson, Yifeng Lu, Xinyun Chen, Hanxiao Liu, Da Huang, Denny Zhou, and Tengyu Ma. Larger language models do in-context learning differently, 2023. URL <https://arxiv.org/abs/2303.03846>.
- Noam Wies, Yoav Levine, and Amnon Shashua. The learnability of in-context learning, 2023. URL <https://arxiv.org/abs/2303.07895>.
- Sang Michael Xie, Aditi Raghunathan, Percy Liang, and Tengyu Ma. An explanation of in-context learning as implicit bayesian inference. *arXiv preprint arXiv:2111.02080*, 2021.
- Mengdi Xu, Yikang Shen, Shun Zhang, Yuchen Lu, Ding Zhao, Joshua Tenenbaum, and Chuang Gan. Prompting decision transformer for few-shot policy generalization. In *International Conference on Machine Learning*, pages 24631–24645. PMLR, 2022.
- Kayo Yin and Jacob Steinhardt. Which attention heads matter for in-context learning?, 2025. URL <https://arxiv.org/abs/2502.14010>.
- Yida Yin, Zekai Wang, Yuvan Sharma, Dantong Niu, Trevor Darrell, and Roei Herzig. In-Context Learning Enables Robot Action Prediction in LLMs, March 2025.
- Wenhao Yu, Nimrod Gileadi, Chuyuan Fu, Sean Kirmani, Kuang-Huei Lee, Montse Gonzalez Arenas, Hao-Tien Lewis Chiang, Tom Erez, Leonard Hasenclever, Jan Humplík, Brian Ichter, Ted Xiao, Peng Xu, Andy Zeng, Tingnan Zhang, Nicolas Heess, Dorsa Sadigh, Jie Tan, Yuval Tassa, and Fei Xia. Language to rewards for robotic skill synthesis, 2023. URL <https://arxiv.org/abs/2306.08647>.

Xiangyuan Zhang, Weichao Mao, Haoran Qiu, and Tamer Başar. Decision transformer as a foundation model for partially observable continuous control. In *2025 American Control Conference (ACC)*, pages 239–244. IEEE, 2025a.

Xilun Zhang, Shiqi Liu, Peide Huang, William Jongwon Han, Yiqi Lyu, Mengdi Xu, and Ding Zhao. Dynamics as prompts: In-context learning for sim-to-real system identifications, 2025b. URL <https://arxiv.org/abs/2410.20357>.

Jiaqiang Ye Zhu, Carla Gomez Cano, David Vazquez Bermudez, and Michal Drozdal. InCoRo: In-Context Learning for Robotics Control with Feedback Loops, February 2024.