

Toward Trusted Autonomous Space Guidance: Meta-Reinforcement Learning, Certification, and Adaptive Lunar Landing

Prof. Roberto Furfaro

**Professor and Deputy Director of Space4 Center
Department Of Systems and Industrial Engineering
Department of Aerospace and Mechanical Engineering
University of Arizona**



Motivations: AI/ML in Space Flight

- Over the past few years, Machine Learning has **made enormous advancements**
 - E.g. Convolutional Neural Networks, Recurrent Neural Networks, LLM (ChatGPT)
- Enabled by **computing speed** (e.g. Massive Parallel and GPUs) and **data availability**
 - We are in a **Big Data Age!**
- **Role of AI in Space Flight:**
 - Fly *autonomous trajectories* robustly adapting to
 - Uncertain and unmodeled dynamics and environment
 - Real-time **decision making** (e.g. space exploration)
- **In this talk: Meta-Learning & Meta- Reinforcement Learning Methods + Certification for Planetary (Lunar) Landing**





Vanilla Policy Gradient: REINFORCE Algorithm

$$\text{recall: } J(\theta) = E_{\tau \sim p_{\theta}(\tau)} \left[\sum_t r(\mathbf{s}_t, \mathbf{a}_t) \right] \approx \frac{1}{N} \sum_i \sum_t r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t})$$

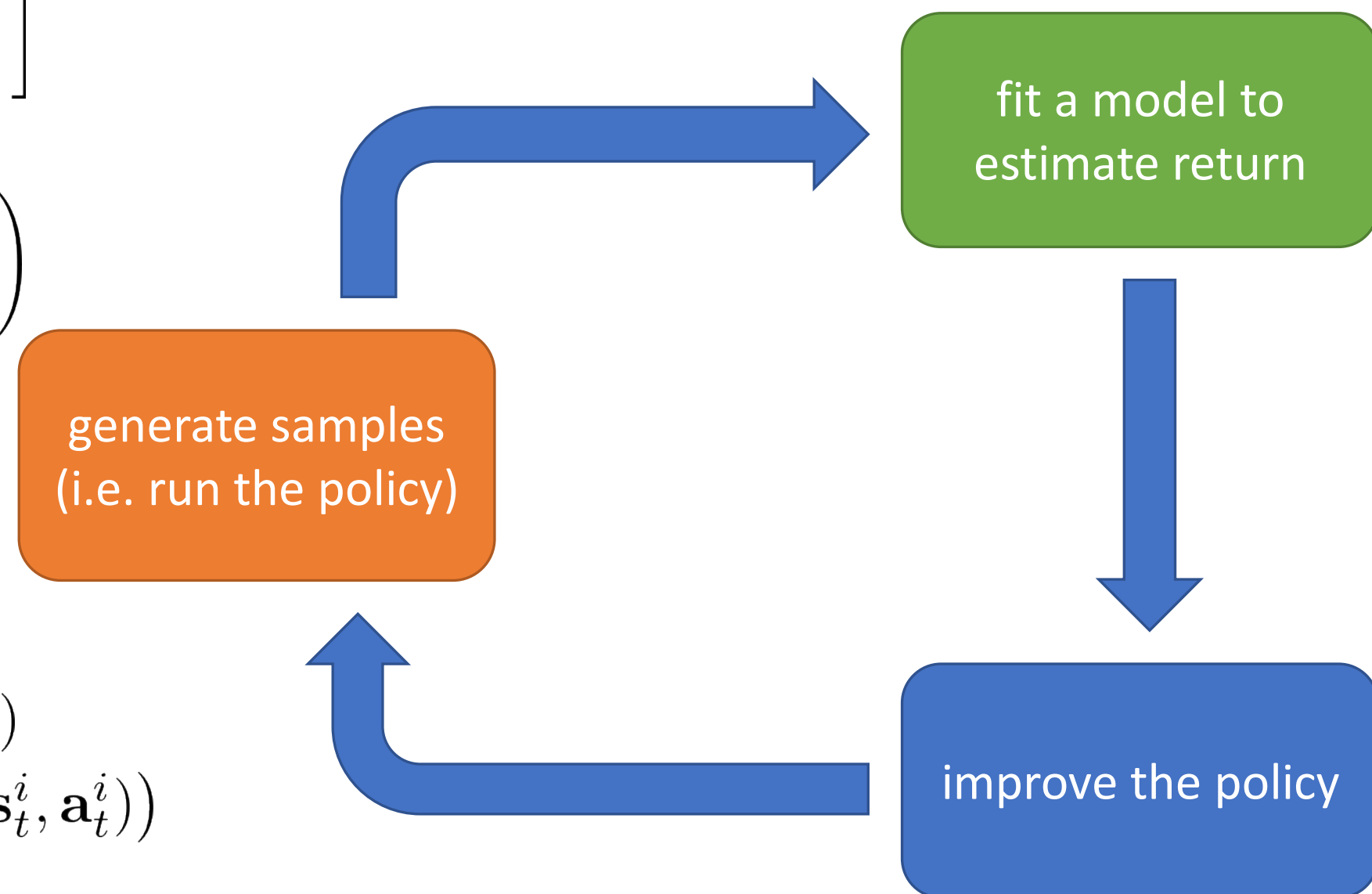
$$\nabla_{\theta} J(\theta) = E_{\tau \sim \pi_{\theta}(\tau)} \left[\left(\sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t | \mathbf{s}_t) \right) \left(\sum_{t=1}^T r(\mathbf{s}_t, \mathbf{a}_t) \right) \right]$$

$$\nabla_{\theta} J(\theta) \approx \frac{1}{N} \sum_{i=1}^N \left(\sum_{t=1}^T \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_{i,t} | \mathbf{s}_{i,t}) \right) \left(\sum_{t=1}^T r(\mathbf{s}_{i,t}, \mathbf{a}_{i,t}) \right)$$

$$\theta \leftarrow \theta + \alpha \nabla_{\theta} J(\theta)$$

REINFORCE algorithm:

1. sample $\{\tau^i\}$ from $\pi_{\theta}(\mathbf{a}_t | \mathbf{s}_t)$ (run the policy)
2. $\nabla_{\theta} J(\theta) \approx \sum_i \left(\sum_t \nabla_{\theta} \log \pi_{\theta}(\mathbf{a}_t^i | \mathbf{s}_t^i) \right) \left(\sum_t r(\mathbf{s}_t^i, \mathbf{a}_t^i) \right)$
3. $\theta \leftarrow \theta + \alpha \nabla_{\theta} J(\theta)$



Actor-Critic Approach: Fit the Value Function

$$V^\pi(\mathbf{s}_t) = \sum_{t'=t}^T E_{\pi_\theta} [r(\mathbf{s}_{t'}, \mathbf{a}_{t'}) | \mathbf{s}_t]$$

$$J(\theta) = E_{\mathbf{s}_1 \sim p(\mathbf{s}_1)} [V^\pi(\mathbf{s}_1)]$$

Training Data

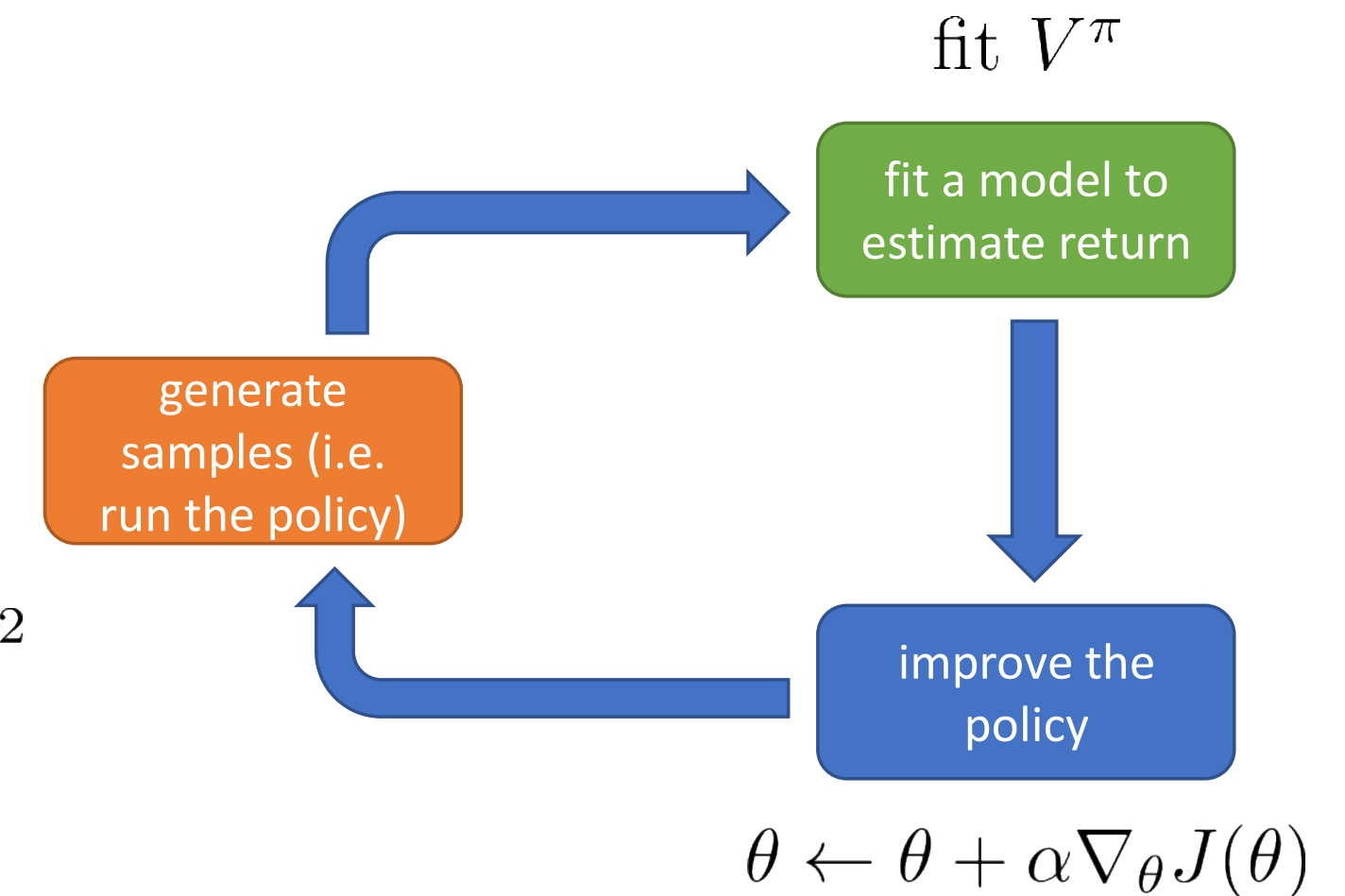
$$\left\{ \left(\mathbf{s}_{i,t}, \underbrace{\sum_{t'=t}^T r(\mathbf{s}_{i,t'}, \mathbf{a}_{i,t'})}_{y_{i,t}} \right) \right\}$$

Monte Carlo Approx

$$V^\pi(\mathbf{s}_t) \approx \sum_{t'=t}^T r(\mathbf{s}_{t'}, \mathbf{a}_{t'})$$

Supervised Regression

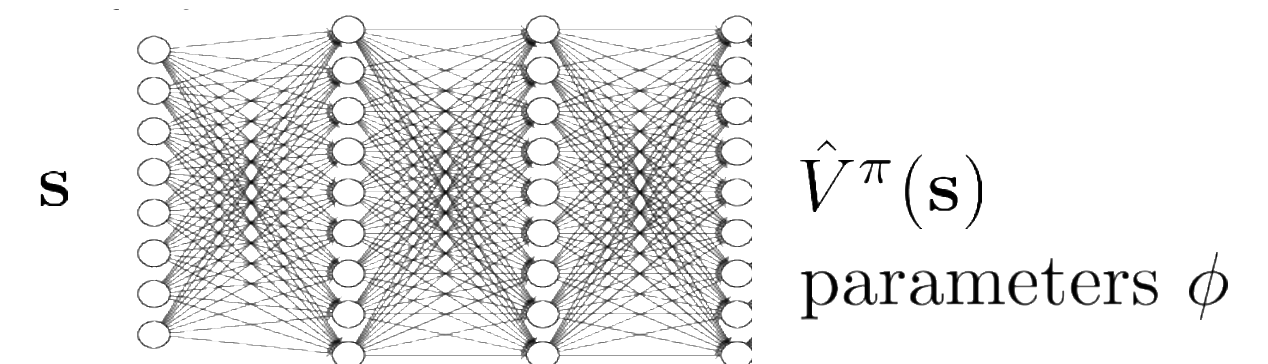
$$\mathcal{L}(\phi) = \frac{1}{2} \sum_i \left\| \hat{V}_\phi^\pi(\mathbf{s}_i) - y_i \right\|^2$$



Batch Actor-Critic Algorithm

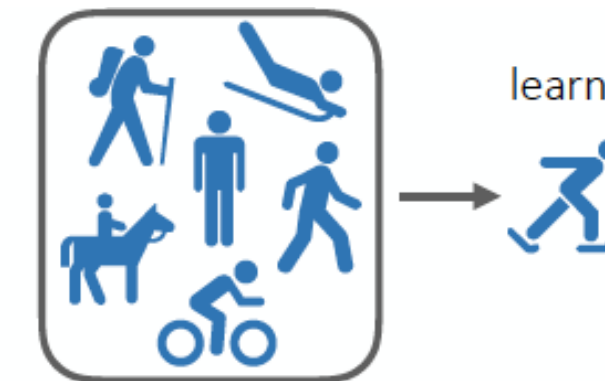
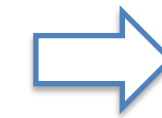
1. sample $\{\mathbf{s}_i, \mathbf{a}_i\}$ from $\pi_\theta(\mathbf{a}|\mathbf{s})$ (run it on the robot)
2. fit $\hat{V}_\phi^\pi(\mathbf{s})$ to sampled reward sums
3. evaluate $\hat{A}^\pi(\mathbf{s}_i, \mathbf{a}_i) = r(\mathbf{s}_i, \mathbf{a}_i) + \hat{V}_\phi^\pi(\mathbf{s}'_i) - \hat{V}_\phi^\pi(\mathbf{s}_i)$
4. $\nabla_\theta J(\theta) \approx \sum_i \nabla_\theta \log \pi_\theta(\mathbf{a}_i | \mathbf{s}_i) \hat{A}^\pi(\mathbf{s}_i, \mathbf{a}_i)$
5. $\theta \leftarrow \theta + \alpha \nabla_\theta J(\theta)$

Deep network for $V(\mathbf{s})$



How Do We Enable Adaptation and Robustness in RL?

- **Meta-Learning: “Learning to Learn”**
 - Training on a distribution of tasks (few shots per task)
- **Continual (“Lifelong”) Learning**
 - Efficiently learn a sequence of tasks from non-stationary distributions





Meta-Learning: Problem Definition (I)

- **Supervised/Imitation Learning problem** solves the following:

$$\begin{aligned} & \arg \max_{\phi} \log p(\phi | \mathcal{D}) \\ & \quad \text{Model Parameters} \quad \quad \quad \text{Training Data} \\ & \quad \quad \quad \swarrow \quad \quad \quad \searrow \\ & = \arg \max_{\phi} \log p(\mathcal{D} | \phi) + \log p(\phi) \\ & \quad \quad \quad \nearrow \quad \quad \quad \nwarrow \\ & \quad \text{data likelihood} \quad \quad \quad \text{regularizer (e.g., weight decay)} \\ & = \arg \max_{\phi} \sum_i \log p(y_i | x_i, \phi) + \log p(\phi) \end{aligned}$$

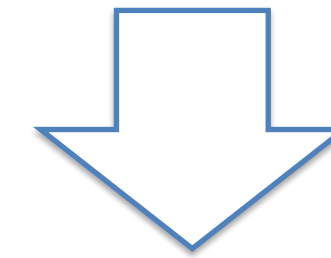
- **Issue:** need a large number of data (Task limited)



Meta-Learning: Problem Definition (II)

- How can we incorporate additional data:

$$\arg \max_{\phi} \log p(\phi | \mathcal{D}, \mathcal{D}_{\text{meta-train}}) \quad \leftarrow \quad \begin{cases} \mathcal{D}_{\text{meta-train}} = \{\mathcal{D}_1, \dots, \mathcal{D}_n\} \\ \mathcal{D}_i = \{(x_1^i, y_1^i), \dots, (x_k^i, y_k^i)\} \end{cases}$$





Meta-Learning: Problem Definition (III)

- Do not keep the additional data (meta training set) around:
Learn the *meta-parameters* θ

$$\arg \max_{\phi} \log p(\phi | \mathcal{D}, \mathcal{D}_{\text{meta-train}}) \approx \arg \max_{\phi} \log p(\phi | \mathcal{D}, \theta^*)$$

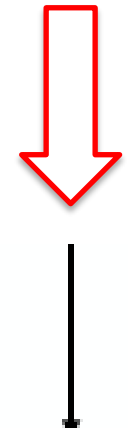
$$\theta^* = \arg \max_{\theta} \log p(\theta | \mathcal{D}_{\text{meta-train}}) \quad \leftarrow \text{This is the Meta-Training}$$

- Some Math details:

$$\begin{aligned} \log p(\phi | \mathcal{D}, \mathcal{D}_{\text{meta-train}}) &= \log \int_{\Theta} p(\phi | \mathcal{D}, \theta) p(\theta | \mathcal{D}_{\text{meta-train}}) d\theta \\ &\approx \log p(\phi | \mathcal{D}, \theta^*) + \log p(\theta^* | \mathcal{D}_{\text{meta-train}}) \end{aligned}$$

← assume $\phi \perp\!\!\!\perp \mathcal{D}_{\text{meta-train}} | \theta$

$$\arg \max_{\phi} \log p(\phi | \mathcal{D}, \mathcal{D}_{\text{meta-train}}) \approx \arg \max_{\phi} \log p(\phi | \mathcal{D}, \theta^*)$$


$$\theta^* = \arg \max_{\theta} \log p(\theta | \mathcal{D}_{\text{meta-train}})$$



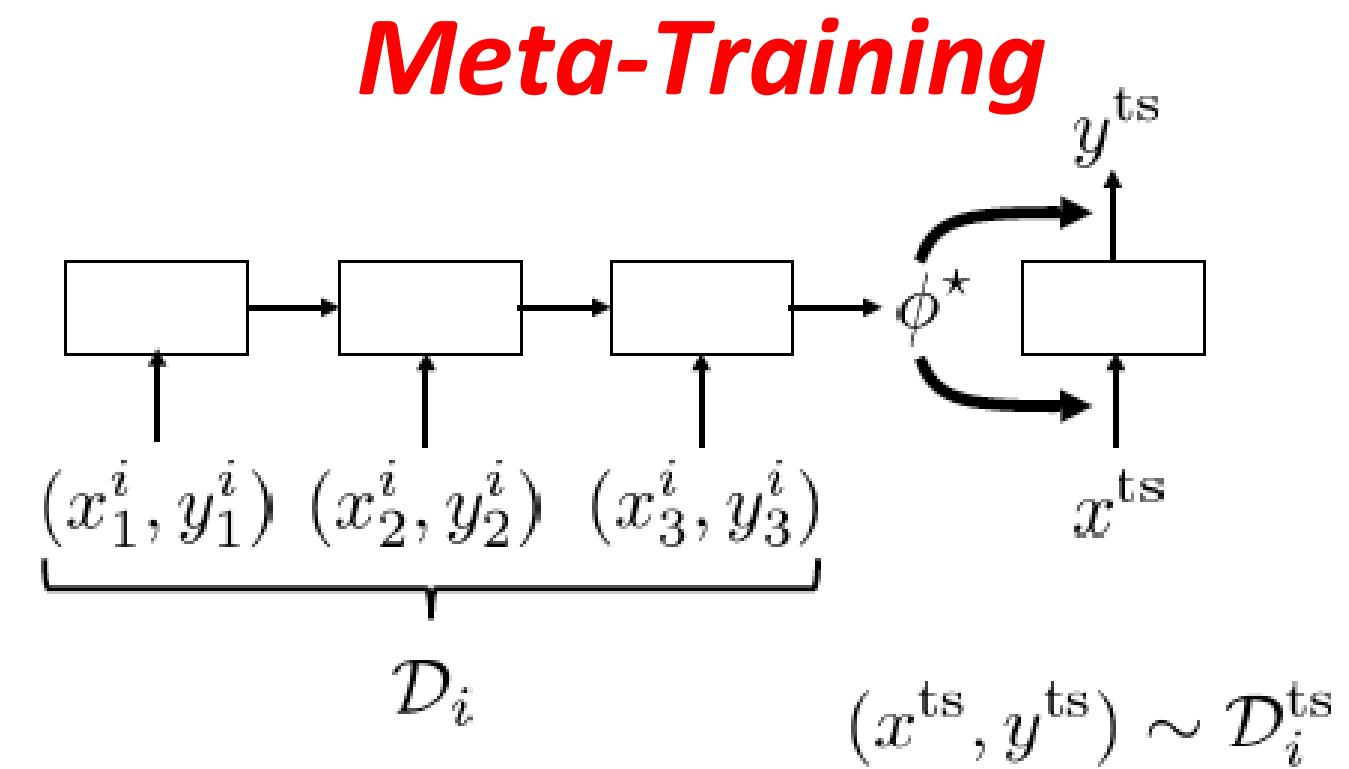
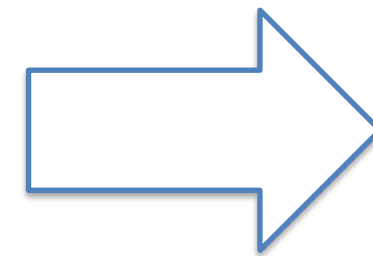
Meta-Learning: Problem Definition (IV)

- Meta-Learning: $\theta^* = \arg \max_{\theta} \log p(\theta | \mathcal{D}_{\text{meta-train}})$
- Adaptation: $\phi^* = \arg \max_{\phi} \log p(\phi | \mathcal{D}^{\text{tr}}, \theta^*) \quad \leftarrow \quad \phi^* = f_{\theta^*}(\mathcal{D}^{\text{tr}})$
- Complete **Meta-Learning Formulation**: *Learn parameters θ such that $\phi = f_{\theta}(\mathcal{D}_i^{\text{tr}})$ is good for $\mathcal{D}_i^{\text{ts}}$*
- Example: Recurrent Networks

$$\mathcal{D}_{\text{meta-train}} = \{(\mathcal{D}_1^{\text{tr}}, \mathcal{D}_1^{\text{ts}}), \dots, (\mathcal{D}_n^{\text{tr}}, \mathcal{D}_n^{\text{ts}})\}$$

$$\mathcal{D}_i^{\text{tr}} = \{(x_1^i, y_1^i), \dots, (x_k^i, y_k^i)\}$$

$$\mathcal{D}_i^{\text{ts}} = \{(x_1^i, y_1^i), \dots, (x_l^i, y_l^i)\}$$



The Reinforcement Meta-Learning Problem (Meta-RL)

- Reinforcement Learning (Deterministic View):

$$\theta^* = \arg \max_{\theta} E_{\pi_{\theta}(\tau)}[R(\tau)] \quad \mathcal{M} = \{\mathcal{S}, \mathcal{A}, \mathcal{P}, r\} \quad \text{Markov Decision Process}$$

- Reinforcement Meta-Learning (Deterministic View):

$$\theta^* = \arg \max_{\theta} \sum_{i=1}^n E_{\pi_{\phi_i}(\tau)}[R(\tau)] \quad \text{where } \phi_i = f_{\theta}(\mathcal{M}_i)$$

- Meta-Training: Distribution of Markov Decision Processes (MDPS)

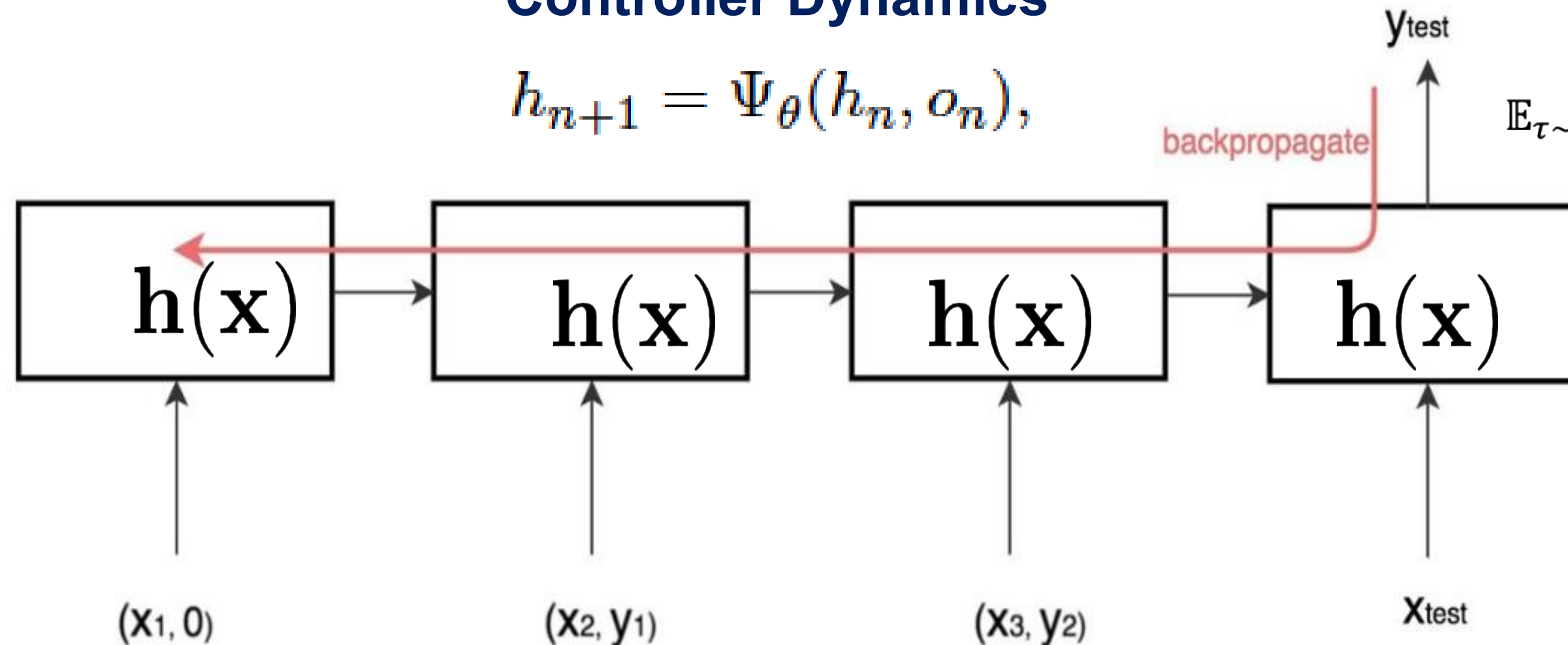
$$\{\mathcal{M}_1, \dots, \mathcal{M}_n\} \quad \mathcal{M}_i \sim p(\mathcal{M})$$

Meta-RL: Recurrent NN optimized via Proximal Policy Optimization ▶▶▶

Recurrent Neural Network

Controller Dynamics

$$h_{n+1} = \Psi_{\theta}(h_n, o_n),$$



The recurrent model store features of which **environment** we are currently in

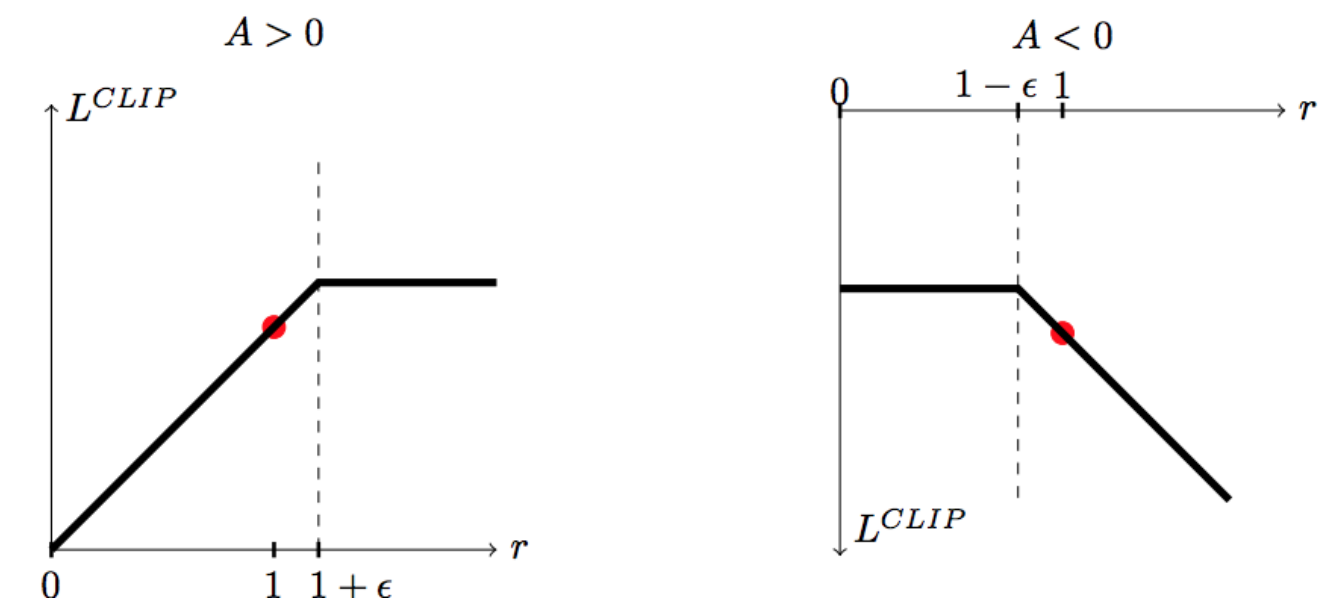
$$u_n = \Pi_{\theta}(h_n, o_n)$$

PPO Optimization

$$L_{\theta_k}^{CLIP}(\theta) = \mathbb{E}_{\tau \sim \pi_{\theta_k}} \left[\sum_{t=0}^T \left[\min(r_t(\theta) A_t^{\pi_{\theta_k}}, \text{clip}(r_t(\theta), 1 - \epsilon, 1 + \epsilon) A_t^{\pi_{\theta_k}}) \right] \right]$$

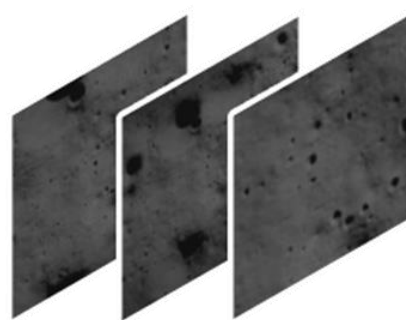
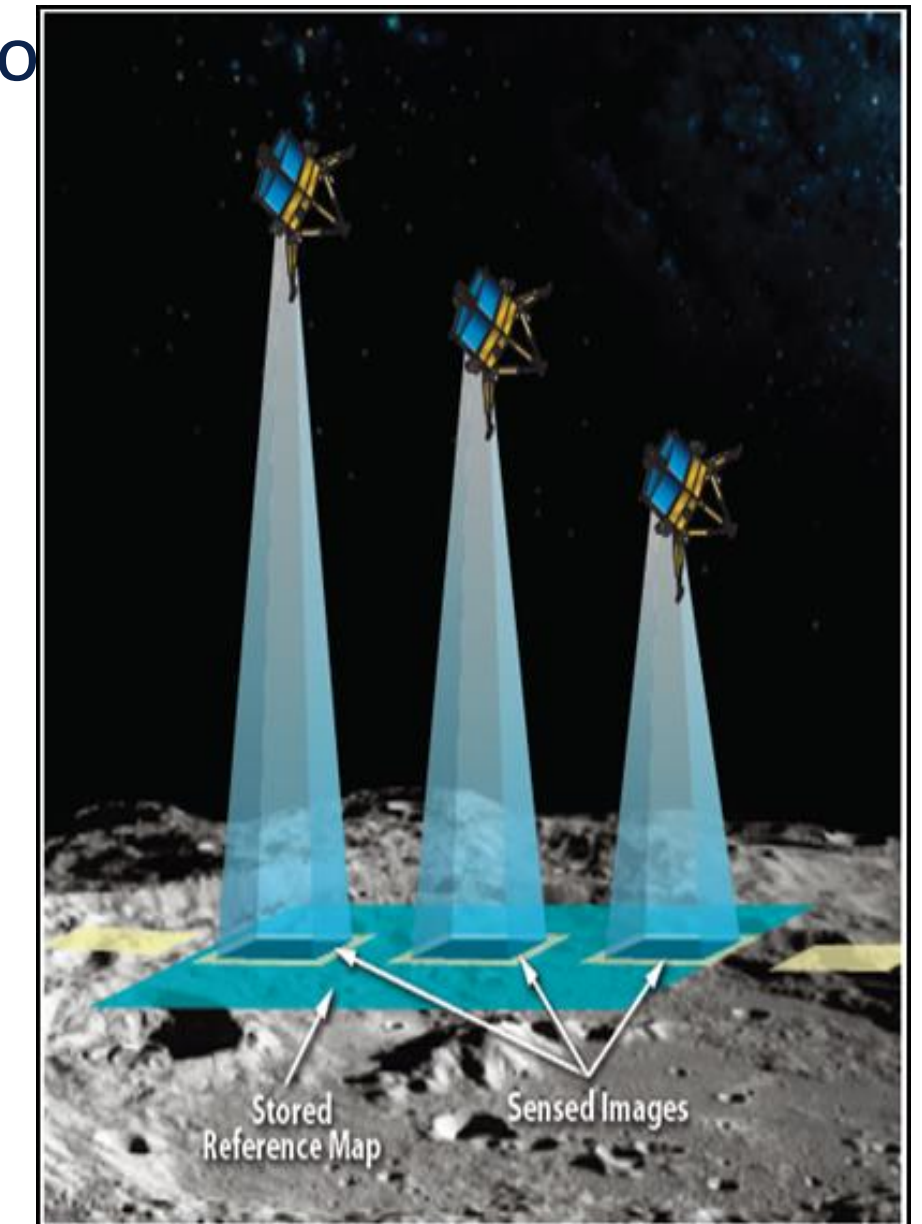
Probability Ratio

$$r_t(\theta) = \frac{\pi_{\theta}(a_t | s_t)}{\pi_{\theta_k}(a_t | s_t)}$$

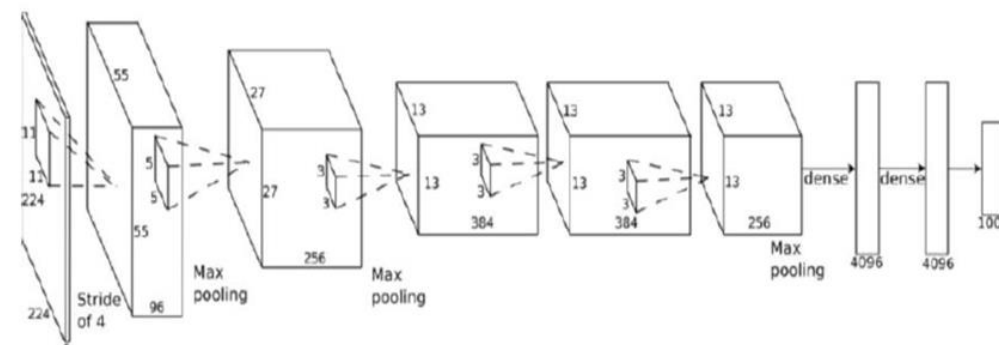


From Sensor Data to Action: Policy Gradient Methods For Planetary Landing

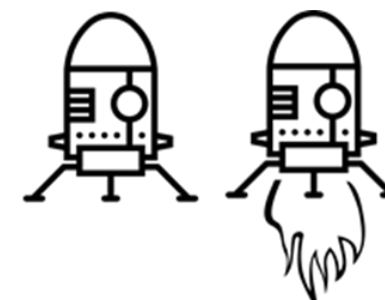
- **Goal:** Develop a Deep RL framework to find a policy that can drive a spacecraft to the planetary surface with pinpoint accuracy and in a fuel efficient fashion
 - Integrate both **attitude and trajectory** guidance and control
 - Integrate **Hazard Avoidance**
- Use a **Policy Gradient approach** to learn the NN policy that maps sensor data to action through simulation
 - Solve Partially Observable MDP (**POMDP**)
 - Use a new **Proximal Policy Optimization (PPO)** method
 - Ensure state-of-the-art performance in many Deep RL problems
 - Develop a Deep Reinforcement **Meta-Learning** for adaptation



Input Images



$$\pi_{\theta}(\mathbf{u}_t | \mathbf{o}_t)$$



Thrust on/off

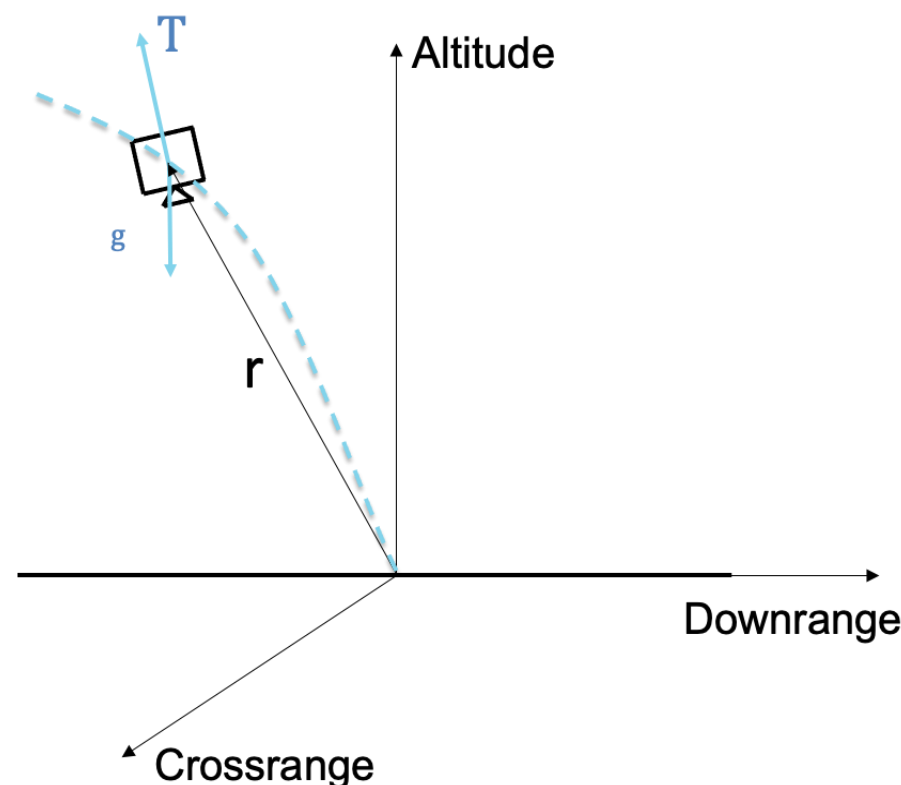
Image-Based Pinpoint 3-DOF Lunar Landing: Problem Statement

- Objectives

- Provide **guidance command** in a **pinpoint lunar landing** scenario
- Embed navigation, guidance and control in a **single self-contained closed-loop system**
- Use **optical and radar altimeter** sensors
- **Robust to uncertain dynamics** (gravity and spacecraft mass)
- **Robust to actuator failure**

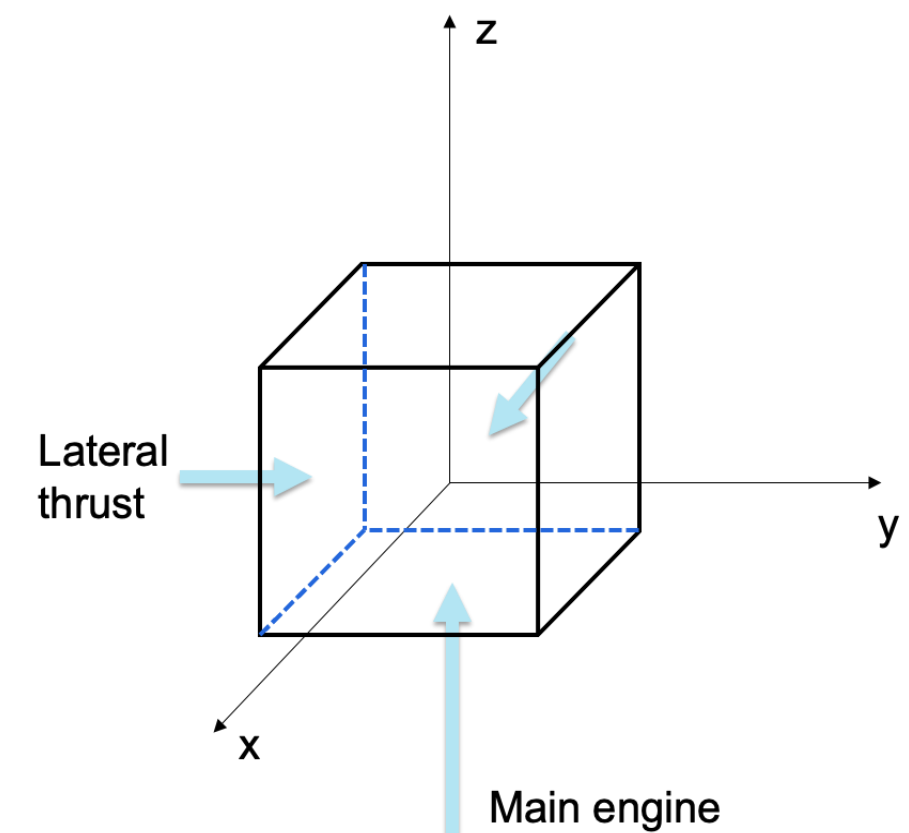
Dynamical model

$$\begin{aligned}\dot{\mathbf{r}} &= \mathbf{v} \\ \dot{\mathbf{v}} &= \mathbf{g} + \frac{\mathbf{T}}{m} \quad 0 < T_{min} < T < T_{max} \\ \dot{m} &= -\frac{\|\mathbf{T}\|}{I_{sp}g_0}\end{aligned}$$



Spacecraft model

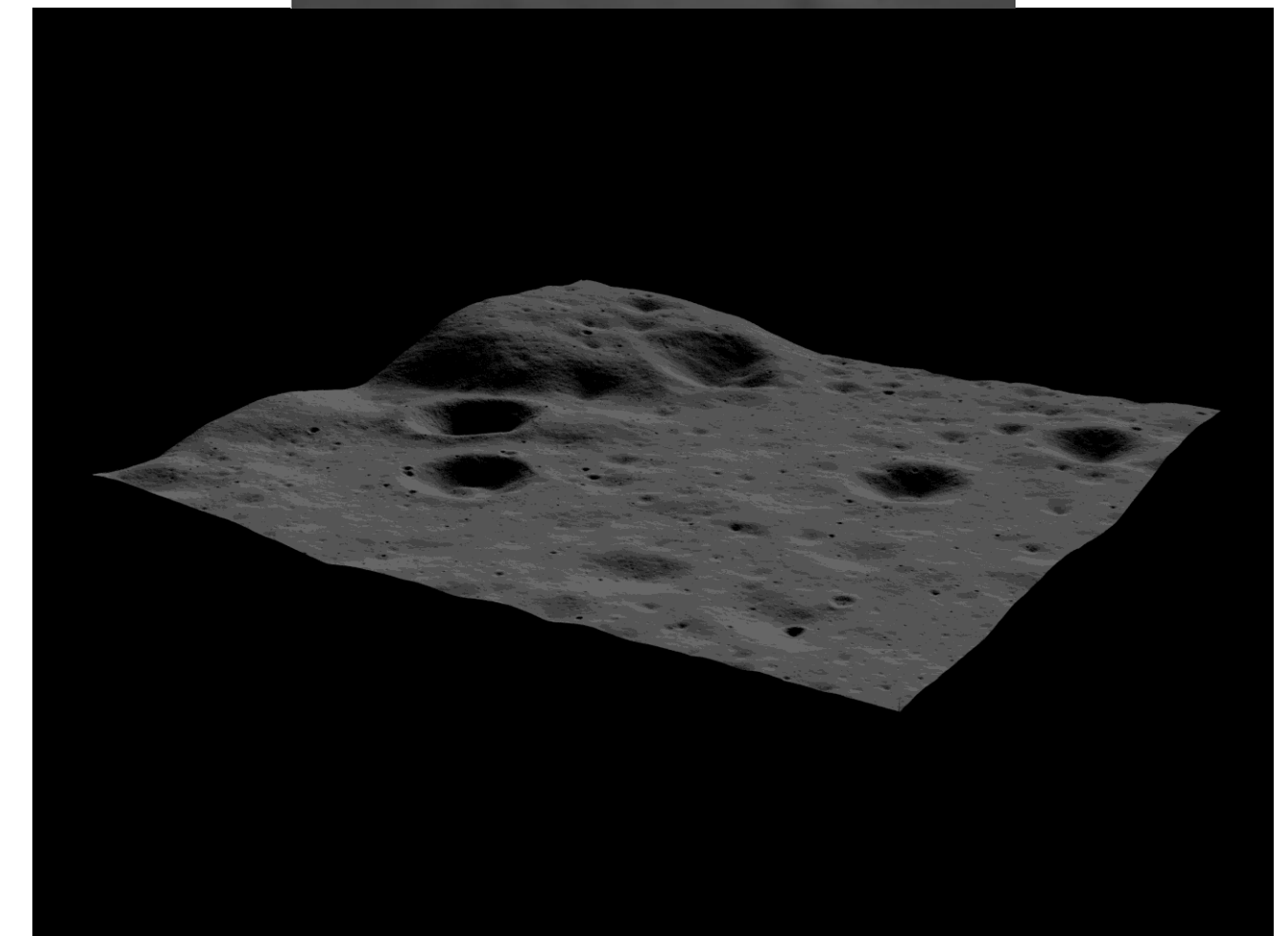
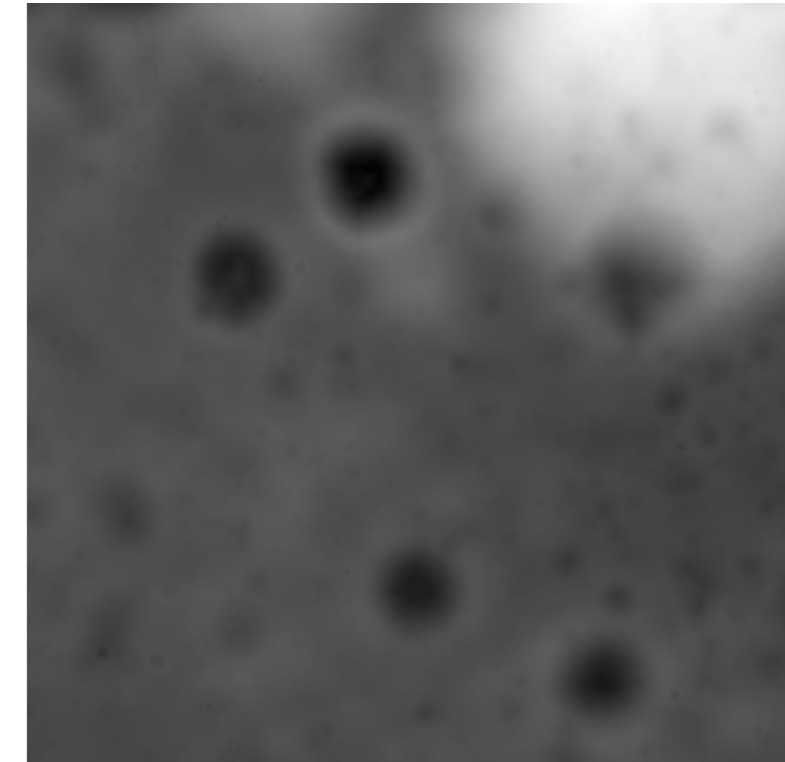
- Thrust
 - Throtttable
 - 3-axis control
- Sensors
 - Camera: boresight always assumed perpendicular to the ground
 - Altimeter: providing altitude and vertical rate



Moon Landing Problem: Lunar Surface Simulation via VisualEnv



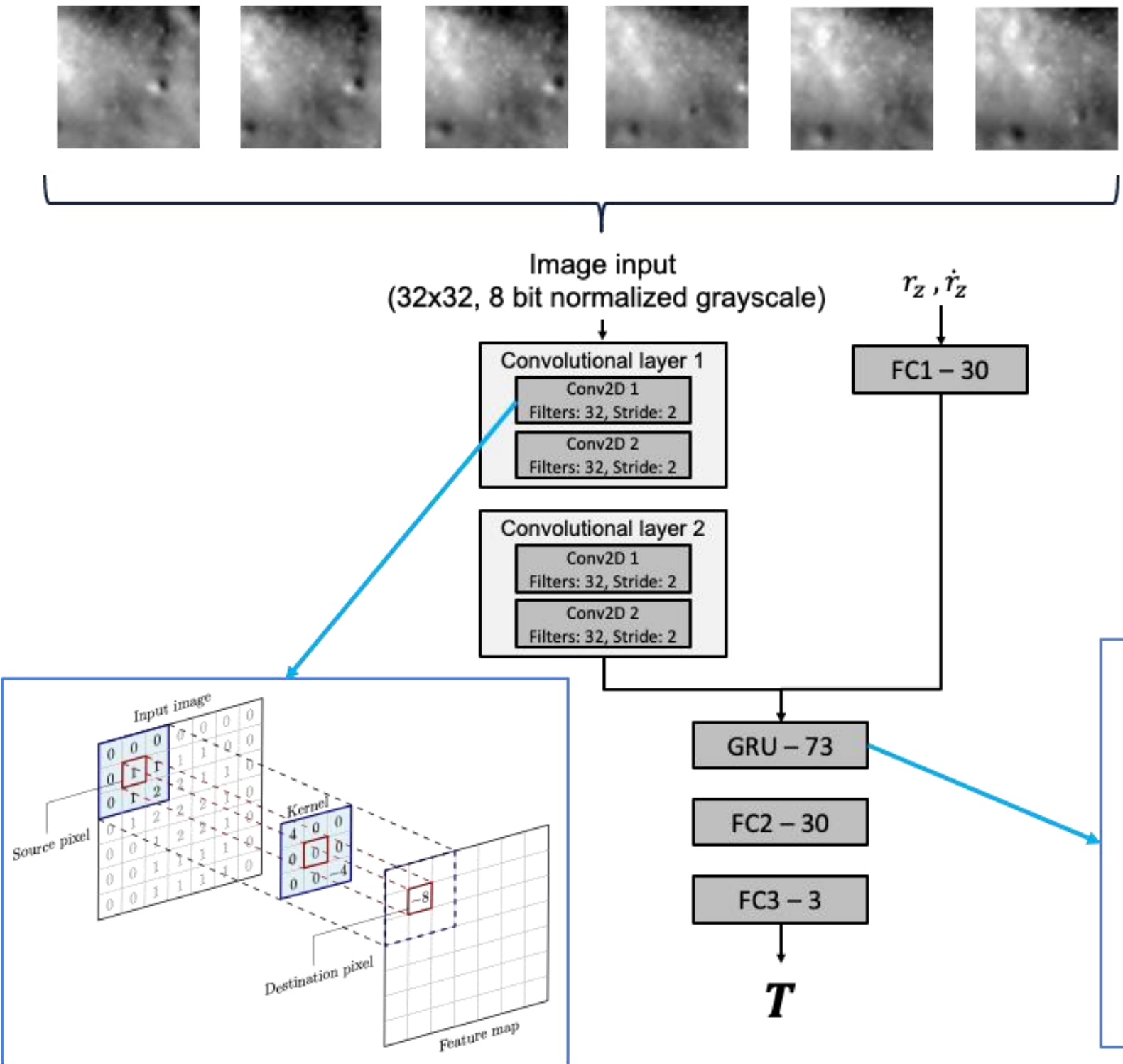
- **Digital Terrain Model (DTM)** from LROC database* (Apollo 16 landing site)
- DTM used to create displacement of the ground
- **Cycles renderer** working with physically based materials and ray tracing technology
 - Use a rough material to simulate lunar regolith
 - Create realistic lighting using a sun-like source at infinitum
- **Camera**
 - Focal length: 50 mm
 - Resolution: 32x32 pixels





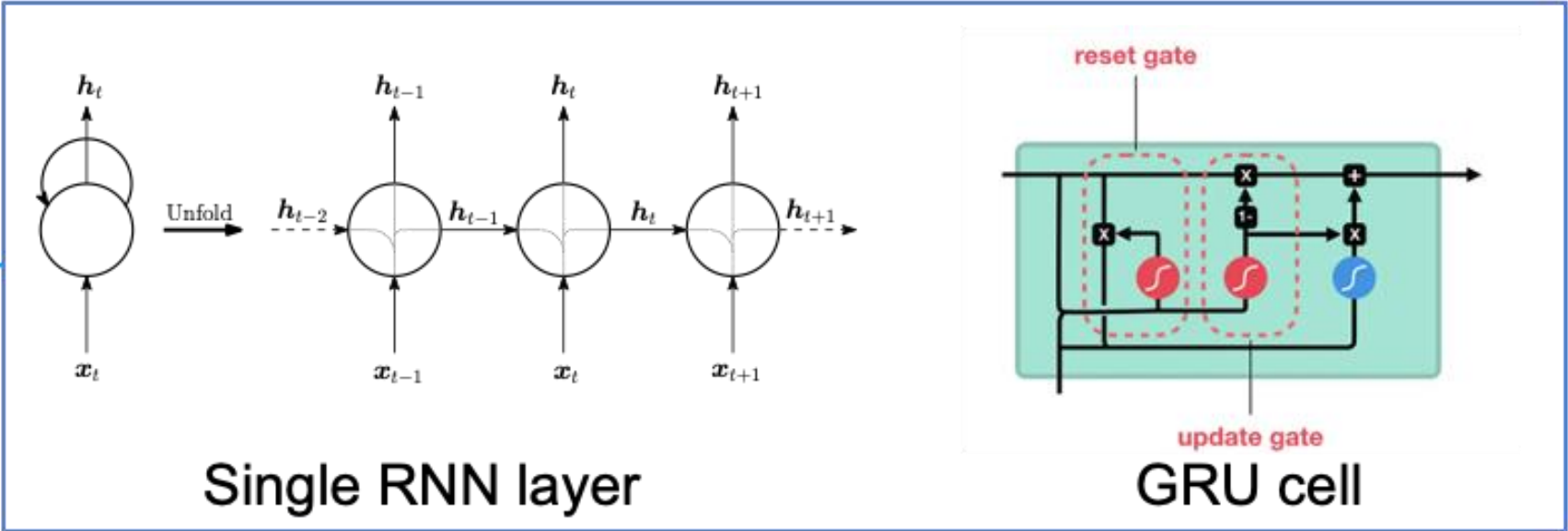
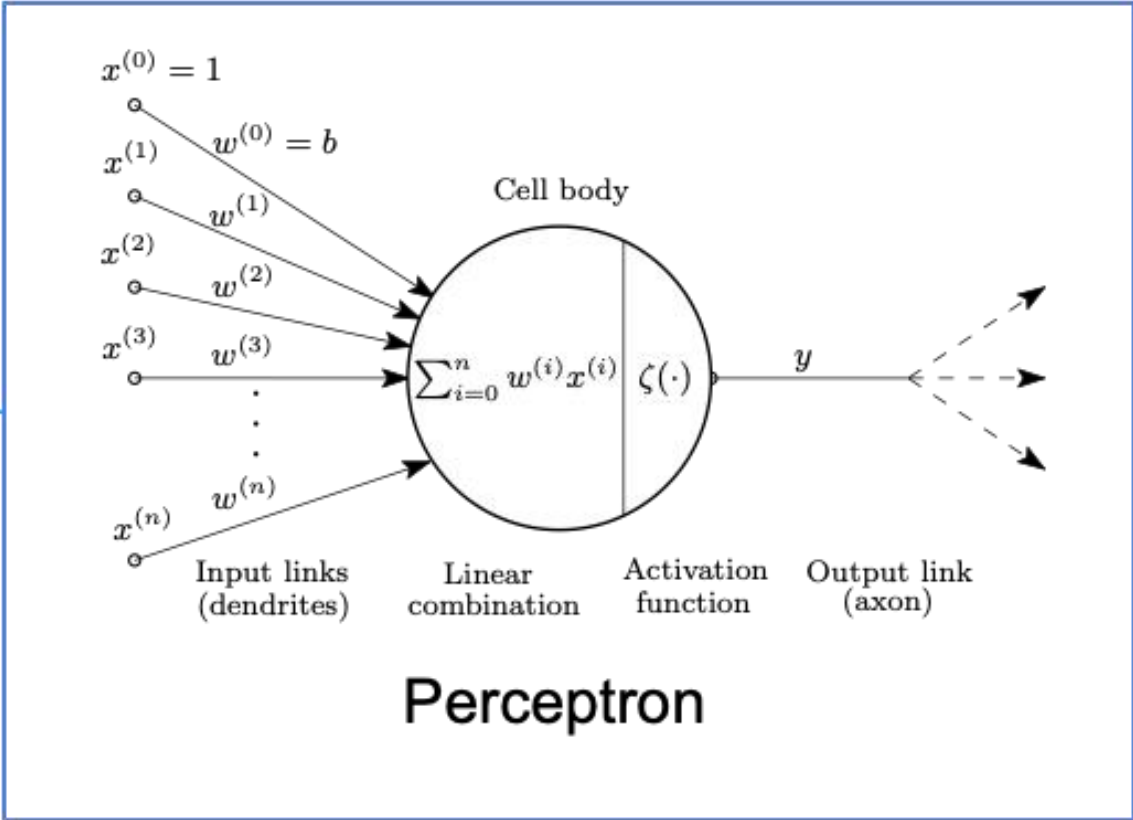
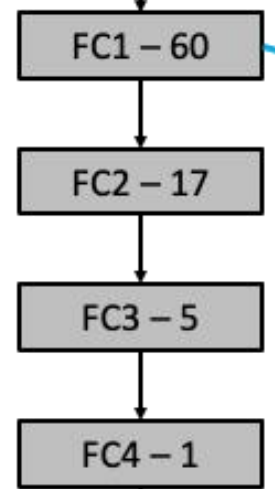
Deep Neural Network Architectures

Policy Network



Critic Network

v_{err}, t_{go}, r_z





Reward Function

$$r = \underbrace{\alpha \|\mathbf{v} - \mathbf{v}_{targ}\|}_{\text{Velocity vector field}} + \underbrace{\beta \|\mathbf{F}_B\|}_{\text{Steps incentive}} + \eta + \underbrace{\kappa(r_z < 0 \text{ and } \|\mathbf{r}\| < r_{lim} \text{ and } \|\mathbf{v}\| < v_{lim})}_{\text{Successful landing}}$$

Control effort

Velocity vector field

Steps incentive

Successful landing

$$\mathbf{v}_{targ} = -v_0 \left(\frac{\hat{\mathbf{r}}}{\|\hat{\mathbf{r}}\|} \right) \left[1 - \exp\left(-\frac{t_{go}}{\tau}\right) \right]$$

$$t_{go} = \frac{\|\hat{\mathbf{r}}\|}{\|\hat{\mathbf{v}}\|}$$

$$\|\hat{\mathbf{r}}\| = \begin{cases} \mathbf{r} - [0 & 0 & 5] & \text{if } r_z > 5 \\ [0 & 0 & r_z] & \text{otherwise} \end{cases}$$

$$\|\hat{\mathbf{v}}\| = \begin{cases} \mathbf{v} - [0 & 0 & -2] & \text{if } r_z > 5 \\ \mathbf{v} - [0 & 0 & -1] & \text{otherwise} \end{cases}$$

$$\tau = \begin{cases} \tau_1 & \text{if } r_z > 5 \\ \tau_2 & \text{otherwise} \end{cases}$$

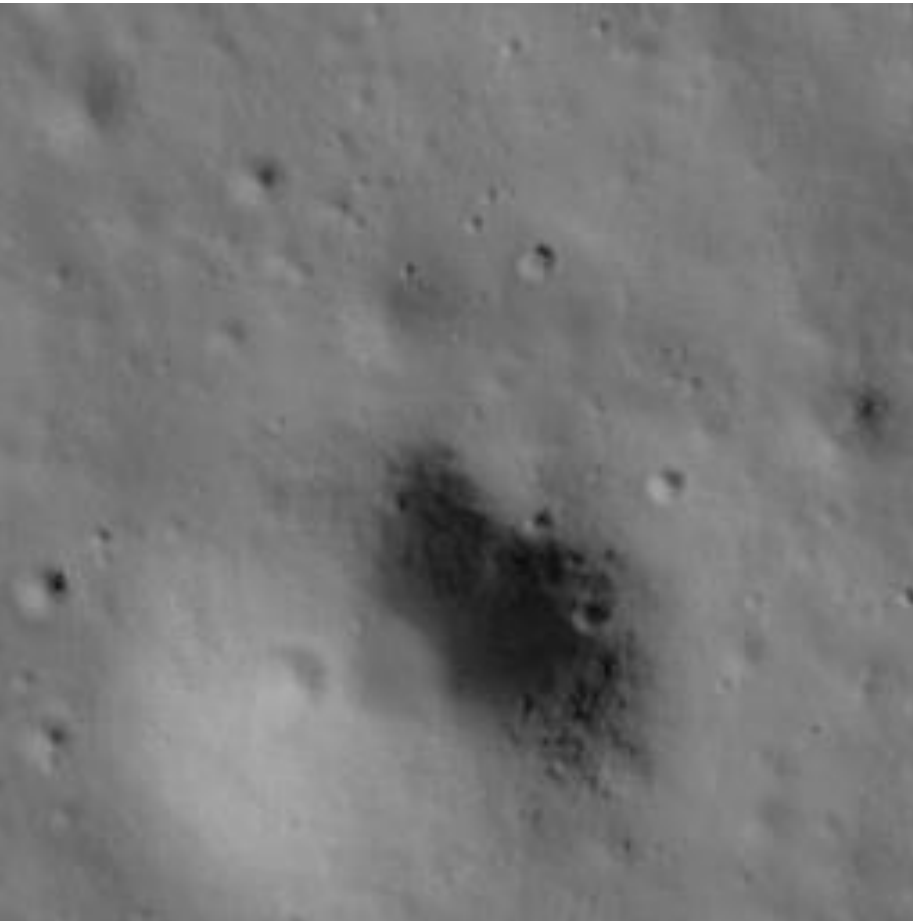
Hyperparameters

v_0 (m/s)	τ_1 (s)	τ_2 (s)	α	β	η	κ
100	20	100	-0.01	-0.05	0.01	10

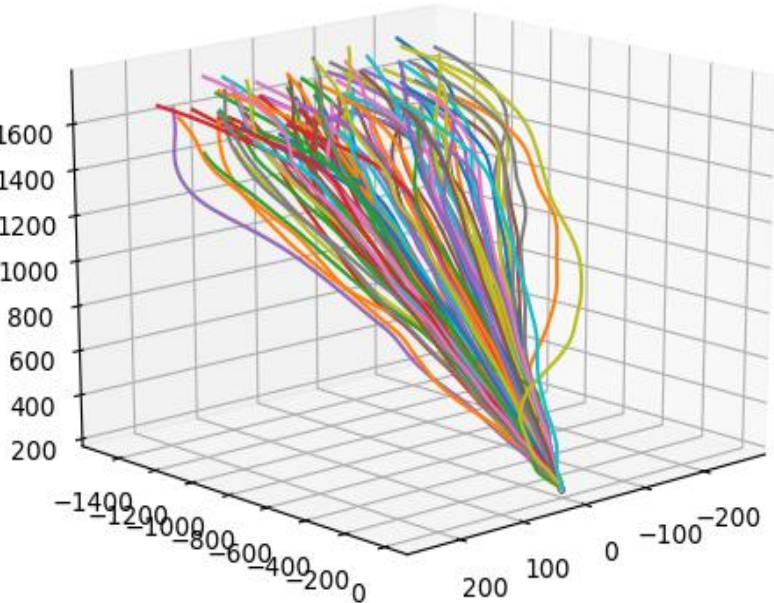
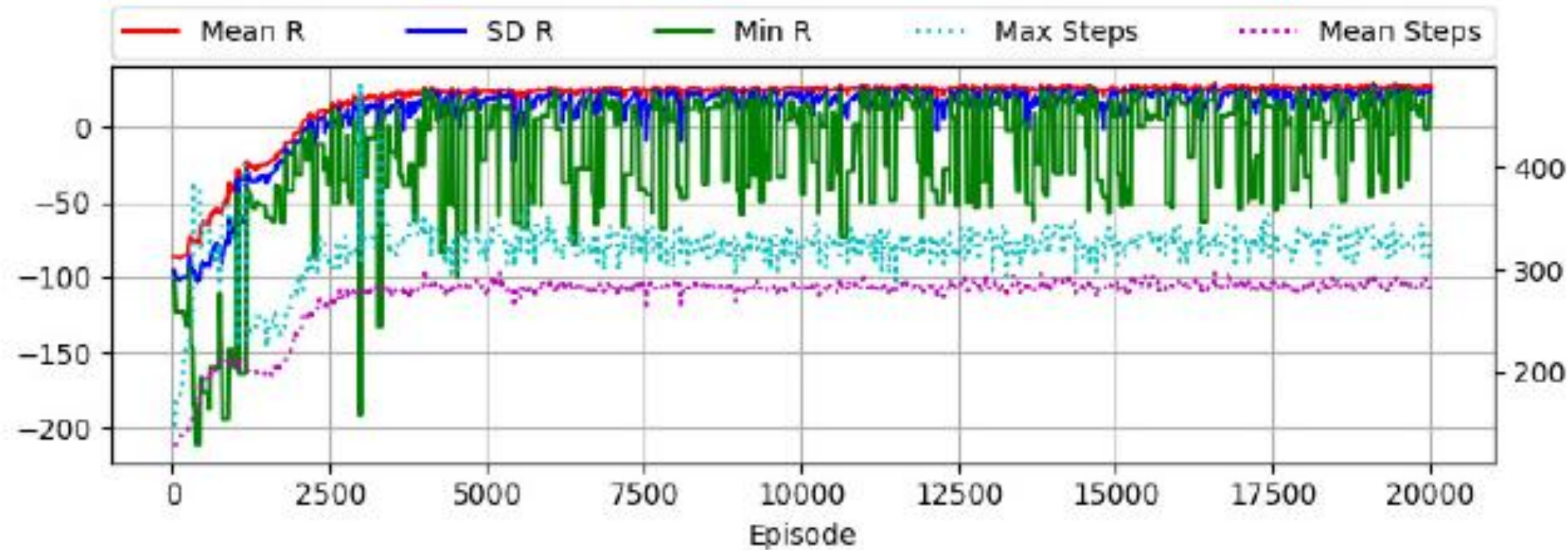
- Penalizes error with target velocity field $\mathbf{v}_{targ}$ (reward shaping)
 - Velocity field based on gaze heuristic potential function → keep velocity aligned with line of sight
- Penalizes control effort $\mathbf{F}_B$
- Gives positive incentive to perform more timesteps η (penalizes constraint violation)
- Gives a bonus for successful landing κ



Closed-Loop meta-RL Guidance Results



Meta- Training

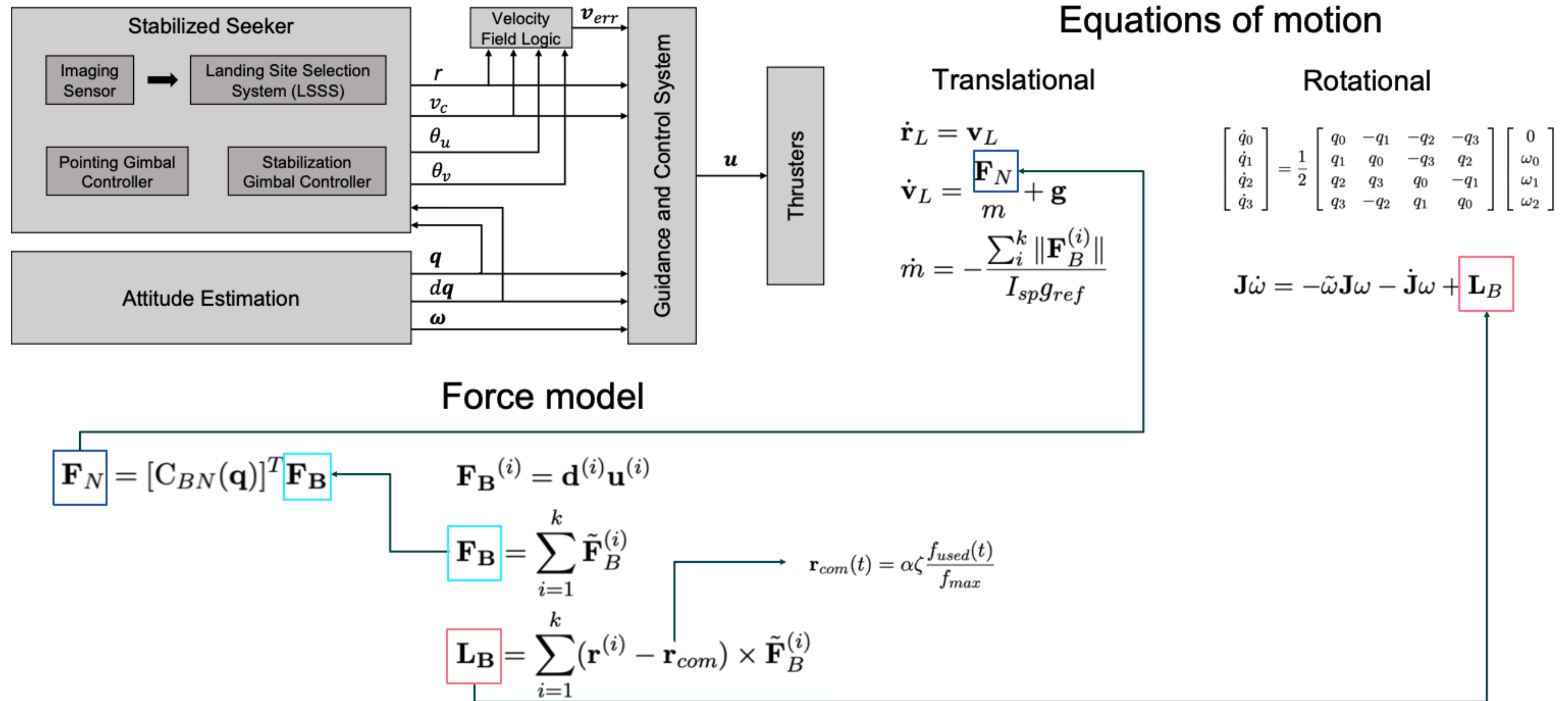


Monte Carlo Performances (100 trajectories)

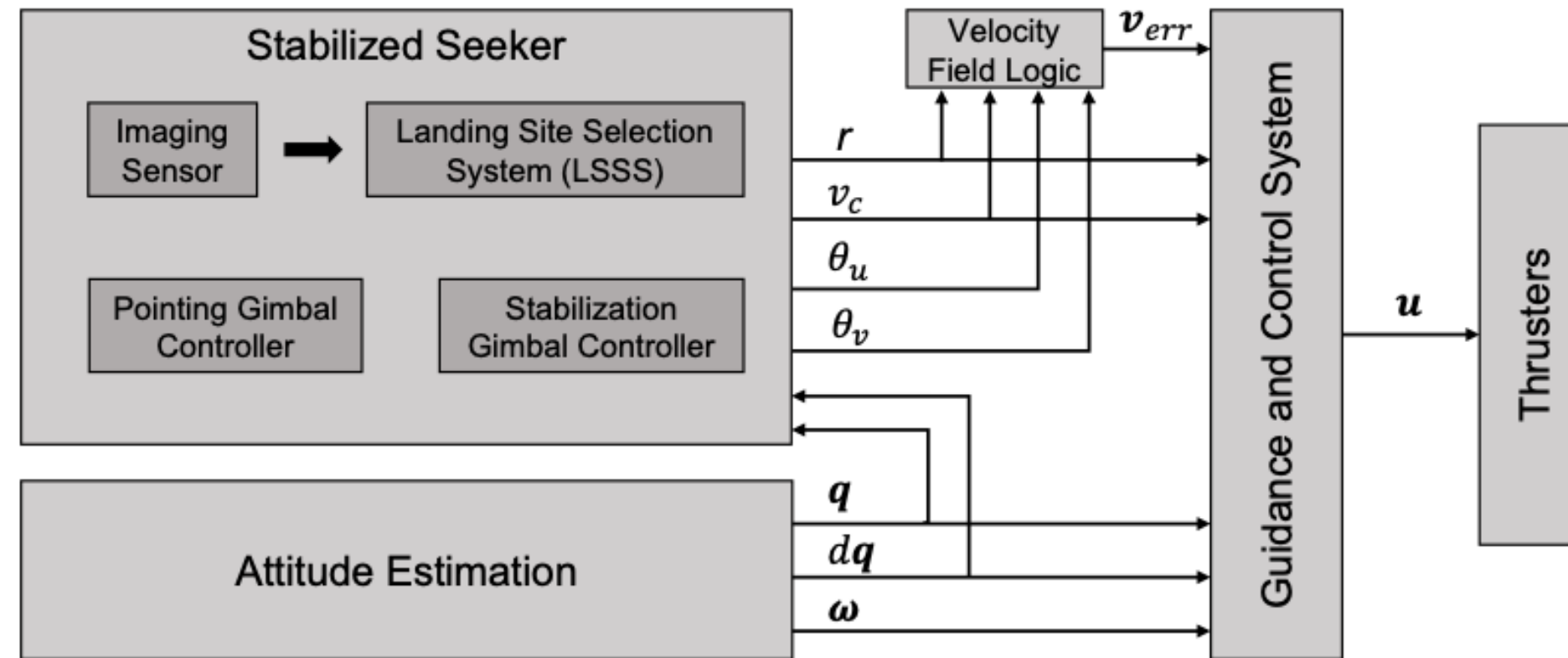
N° episodes	Final position error norm [m]		Final velocity error norm [m/s]		Fuel consumption [kg]
	μ	σ	μ	σ	
20000	1.25	0.10	1.18	0.87	123.46

Scorsoglio, A., D’Ambrosio, A., Ghilardi, L., Gaudet, B., Curti, F. and **Furfaro, R.**, 2022. Image-Based Deep Reinforcement Meta-Learning for Autonomous Lunar Landing. *Journal of Spacecraft and Rockets*, 59(1), pp.153-165.

Extension: 6-DOF Meta-RL GNC with Autonomous Hazard Avoidance/Landing Site Selection (I)



Extension: 6-DOF Meta-RL GNC with Autonomous Hazard Avoidance/Landing Site Selection (II)



Thrusters model

$$\mathbf{u}_{cmd}^{(i)} = \text{clip}(u_{max} \mathbf{u}_{\pi}, u_{min}, u_{max})$$

$$\mathbf{u}_{AF}^{(i_{AF})} = \begin{cases} \mathbf{u}_{cmd}^{(i)} s_{AF} & \text{if fail and } i == i_{AF} \\ \mathbf{u}_{cmd}^{(i)} & \text{otherwise} \end{cases}$$

Actuators and seeker lag

$$\dot{\mathbf{u}} = \frac{\mathbf{u}_{AF} - \mathbf{u}}{\tau_{ctrl}}$$

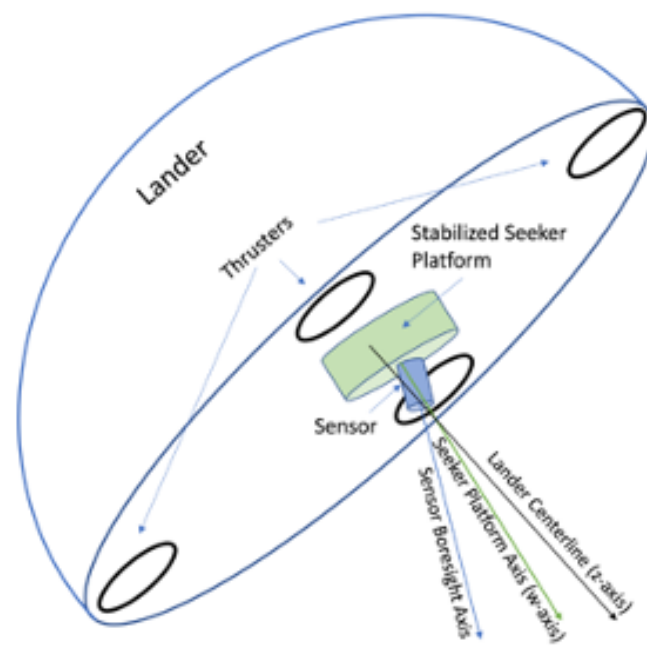
$$\dot{\theta}_u = \frac{\tilde{\theta}_u - \theta_u}{\tau_{seeker}}$$

$$\dot{\theta}_v = \frac{\tilde{\theta}_v - \theta_v}{\tau_{seeker}}$$

$$\dot{r} = \frac{\tilde{r} - r}{\tau_{seeker}}$$

$$\dot{v}_c = \frac{\tilde{v}_c - v_c}{\tau_{seeker}}$$

Seeker model



$$\mathbf{r}_{TL}^S = [\mathbf{C}_{SN}(\mathbf{q}_0)] \mathbf{r}_{TL}^N$$

$$\tilde{\theta}_u = \arcsin(\hat{\boldsymbol{\lambda}}^S \cdot \mathbf{u})$$

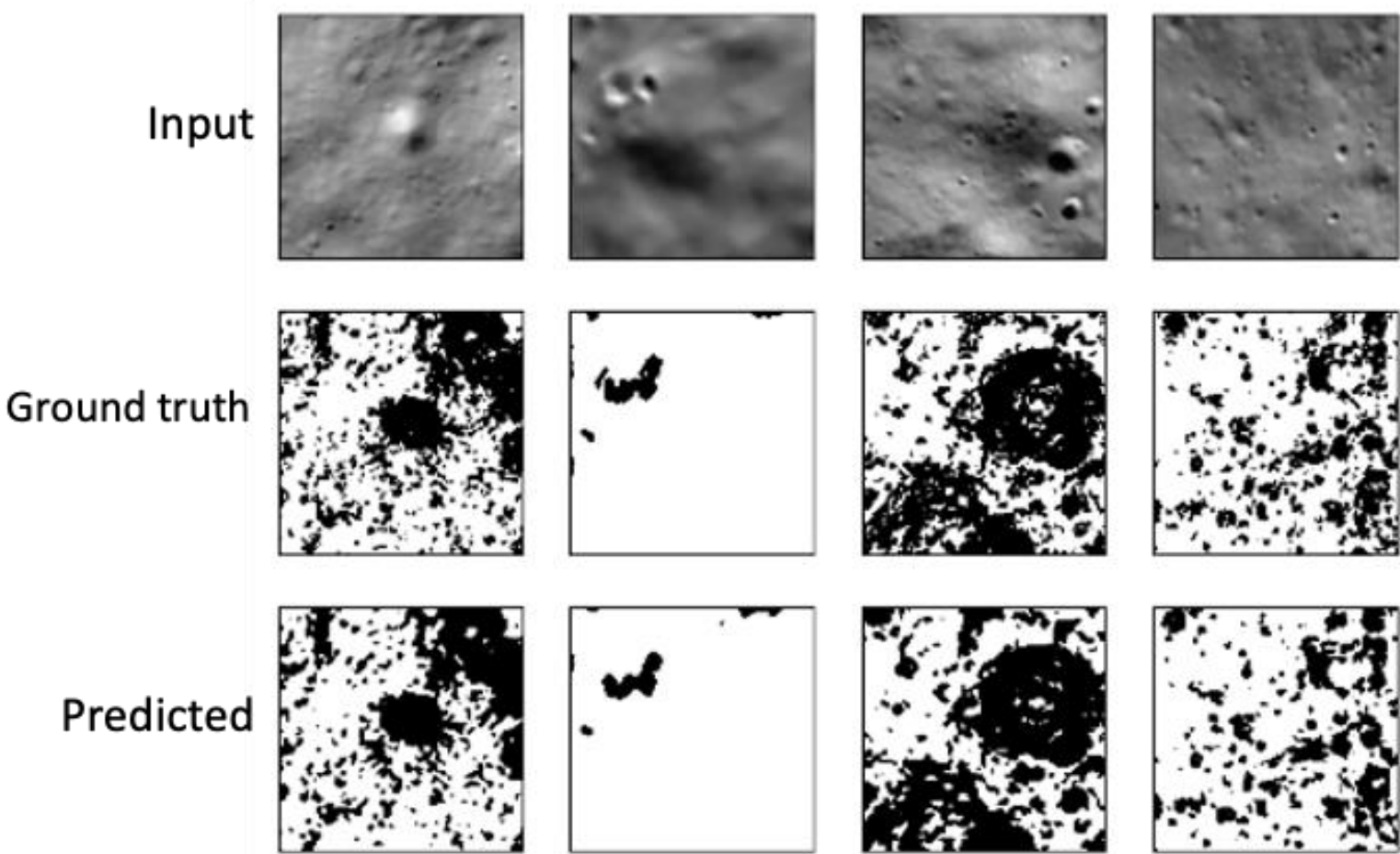
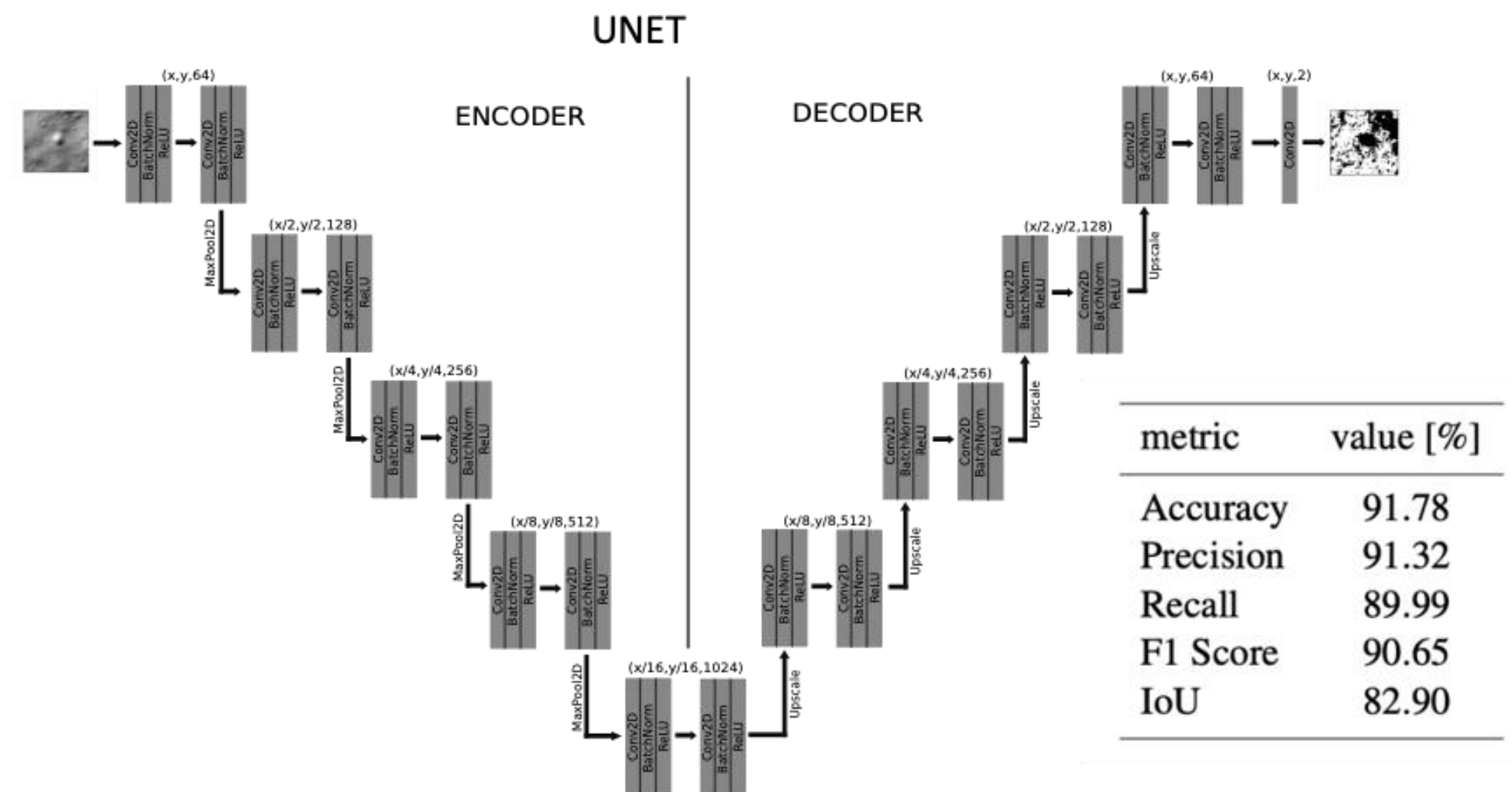
$$\tilde{\theta}_v = \arcsin(\hat{\boldsymbol{\lambda}}^S \cdot \mathbf{v})$$

$$\tilde{r} = \|\mathbf{r}_{TL}\|$$

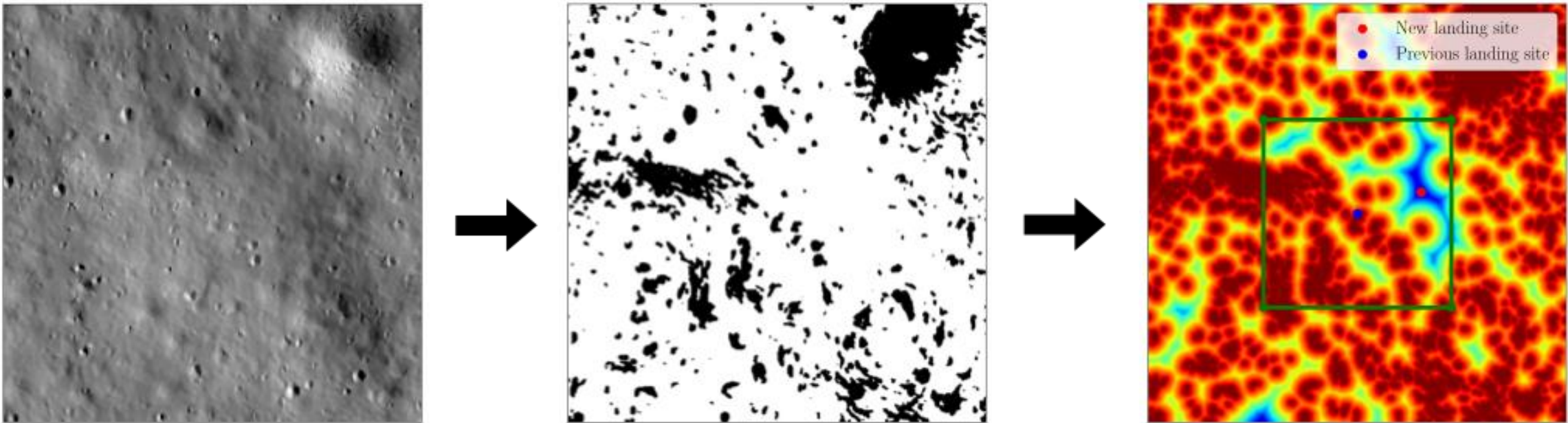
$$\tilde{v}_c = -\frac{\mathbf{r}_{TL} \cdot \mathbf{v}_{TL}}{\|\mathbf{r}_{TL}\|}$$



Landing Site Selection System (LSSS): U-net Design and Training Results



LSSS pipeline



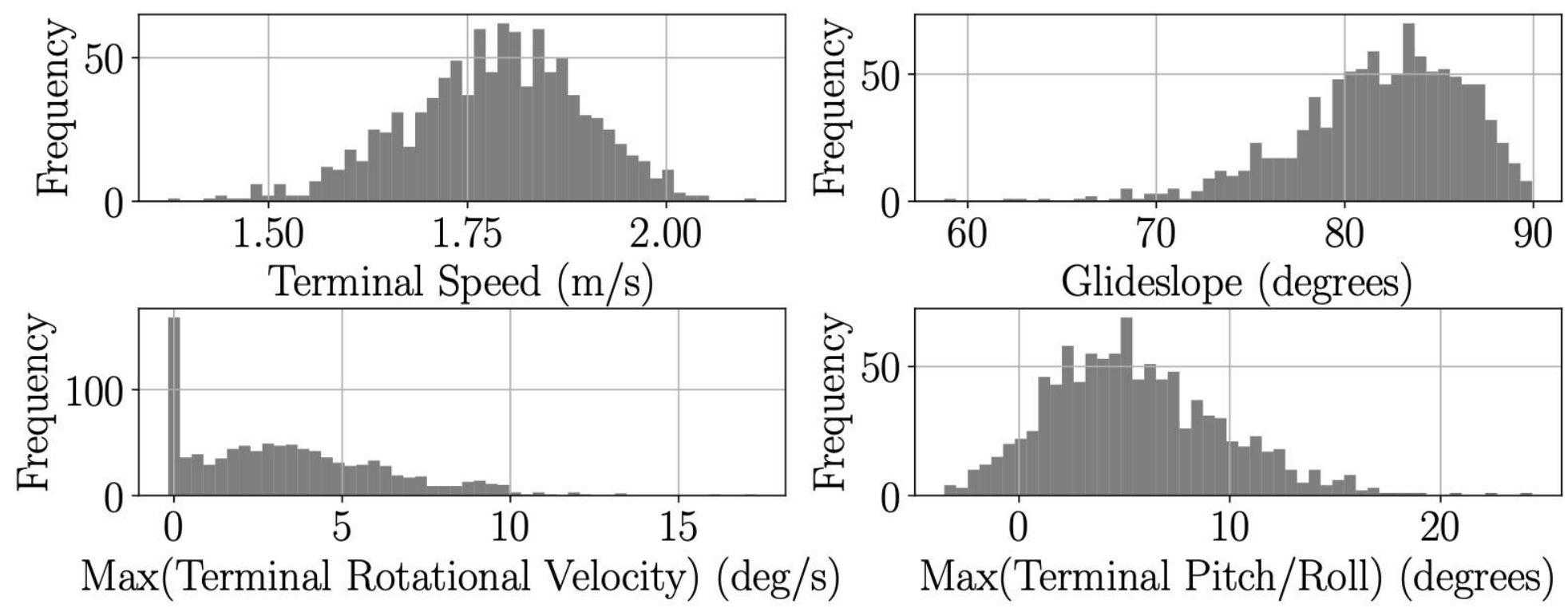
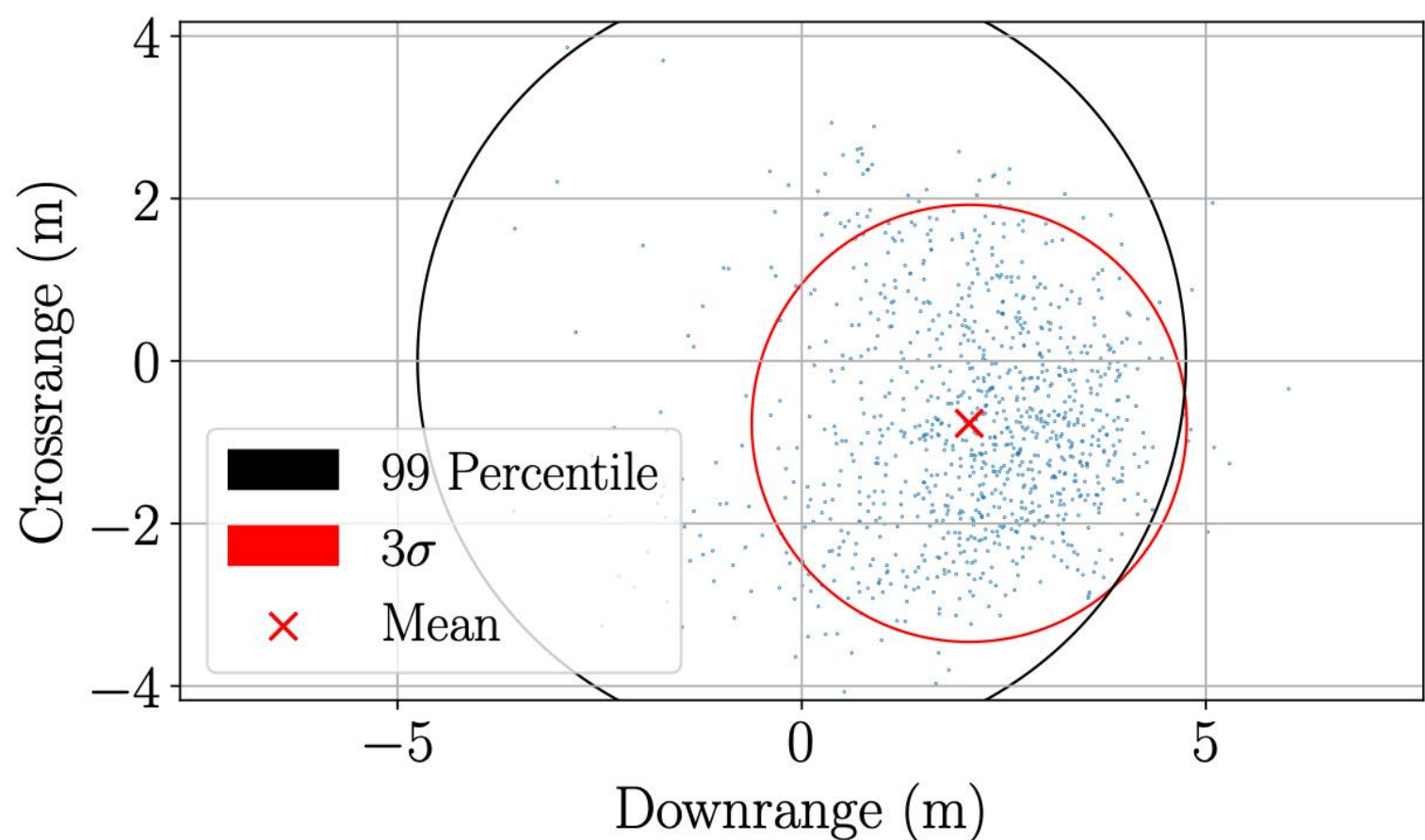
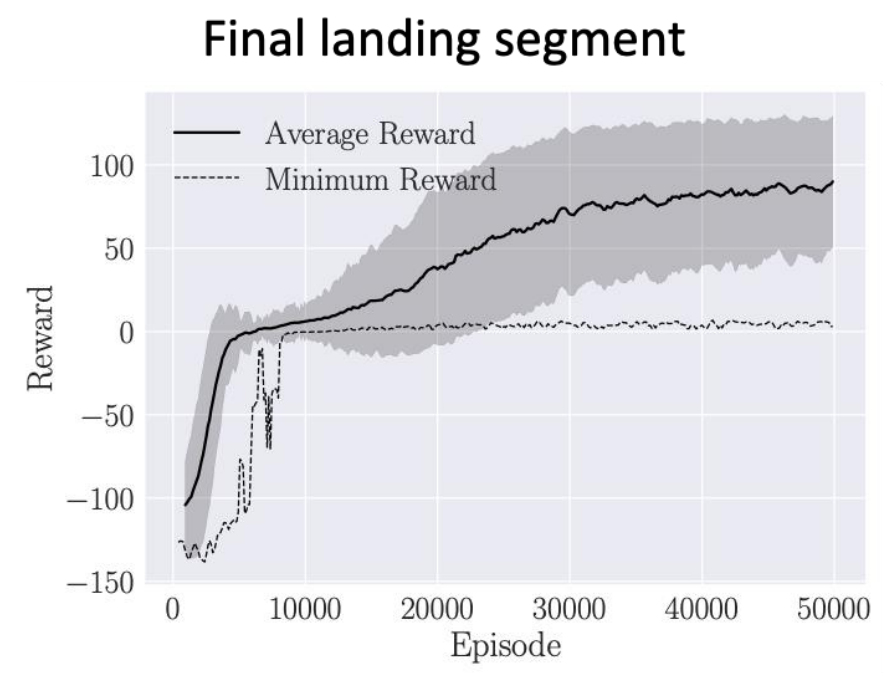
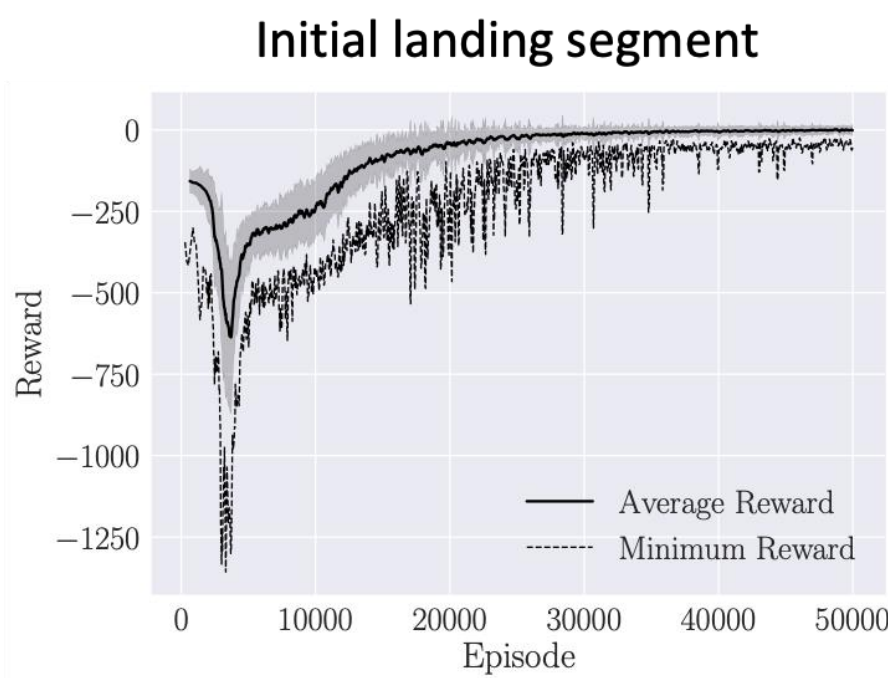
6-DOF Meta-RL Closed Loop GNC: Training, Validation and Testing



Training campaign

Max divert range tested (δ): $[0.05, 0.1, 0.15, 0.2]\rho$

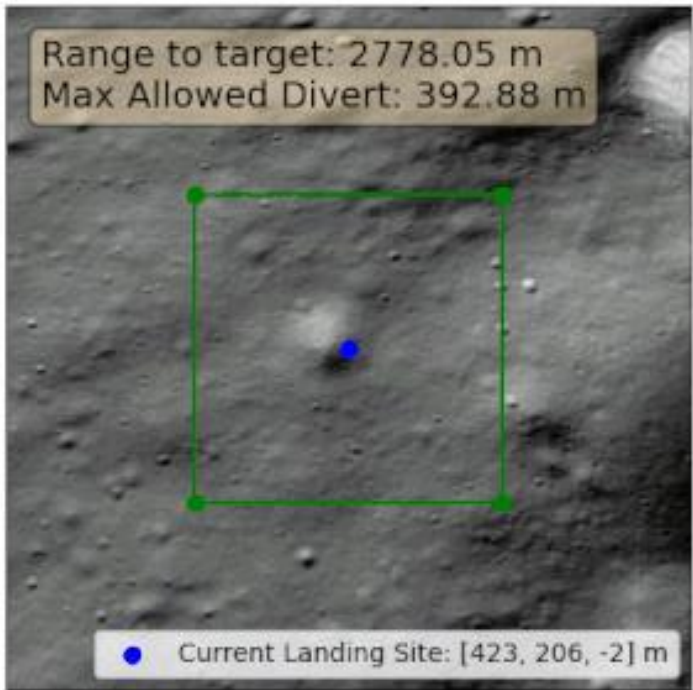
δ	Miss [m]		$\ \mathbf{v}_L\ $ [m/s]		$\max(\mathbf{q})$ [deg]		$\max(\omega)$ [deg/sec]		$\max(\theta_{gs})$ [deg]	
	μ	σ	μ	σ	μ	σ	μ	σ	μ	σ
0.05	4.06	0.32	1.69	0.12	0.84	3.01	2.65	2.64	85.00	2.65
0.10	3.90	0.62	1.78	0.11	3.38	2.93	2.99	2.80	82.03	5.12
0.15	3.38	0.74	1.79	0.10	6.94	2.67	3.32	2.96	82.38	4.10
0.20	2.78	0.89	1.79	0.10	6.81	4.13	3.48	2.64	82.16	4.41



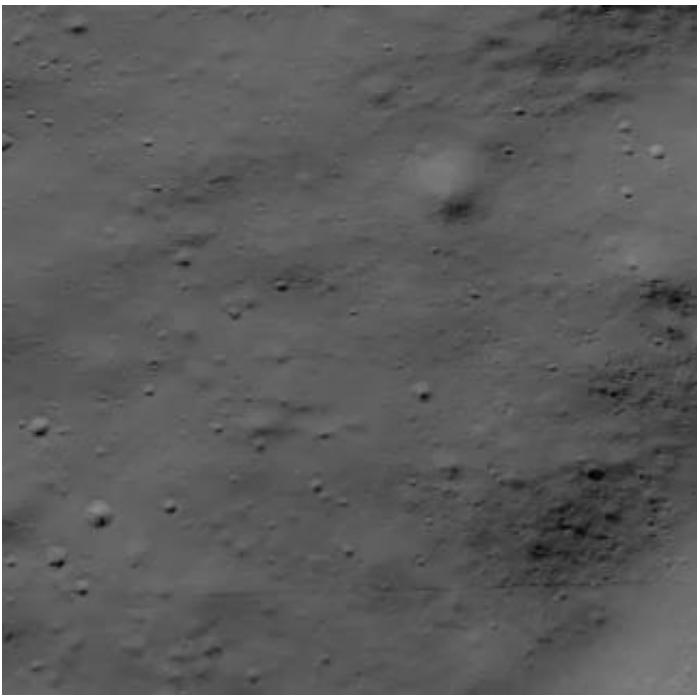
LSSS Performance (I)



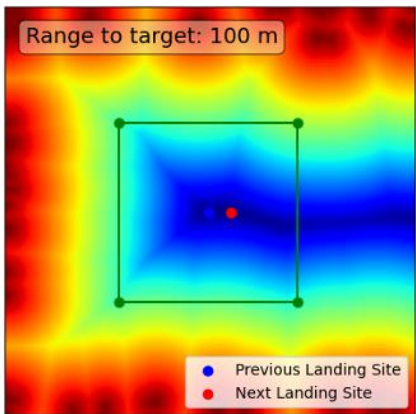
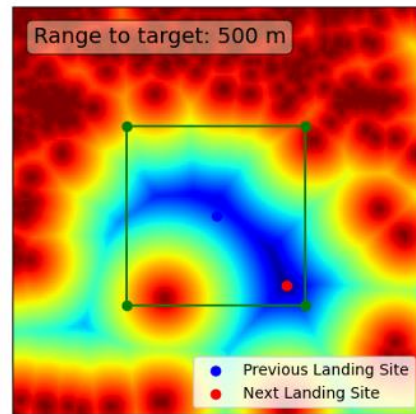
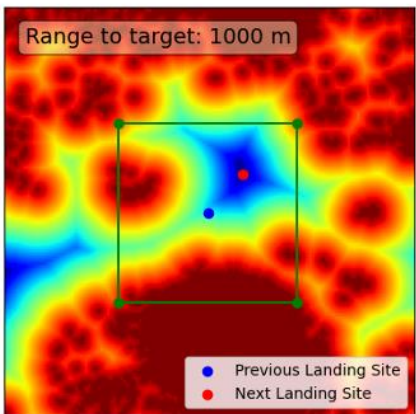
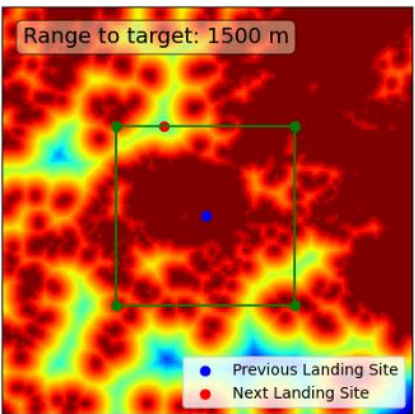
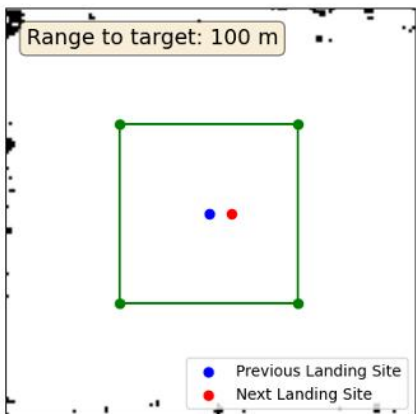
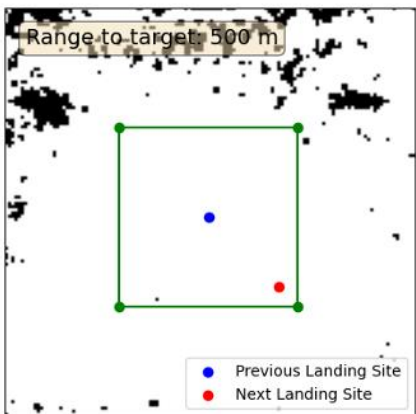
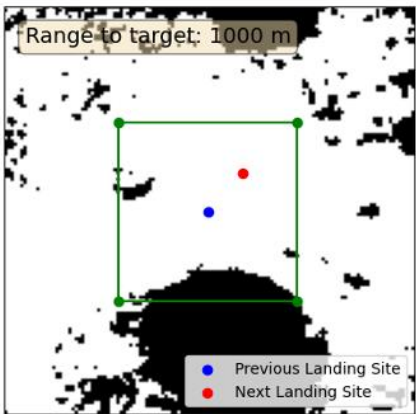
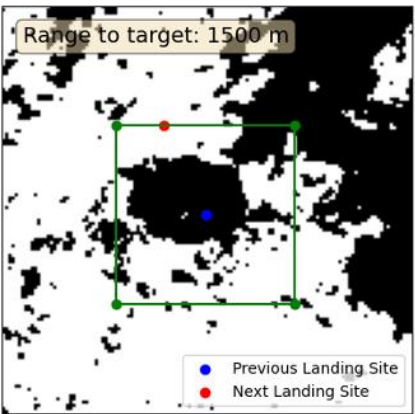
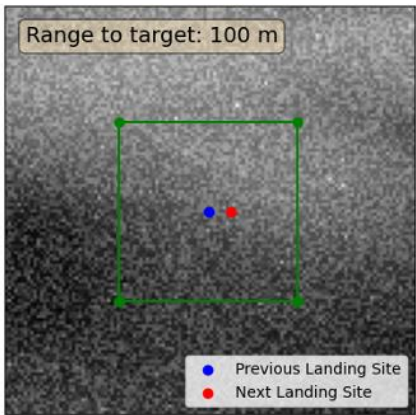
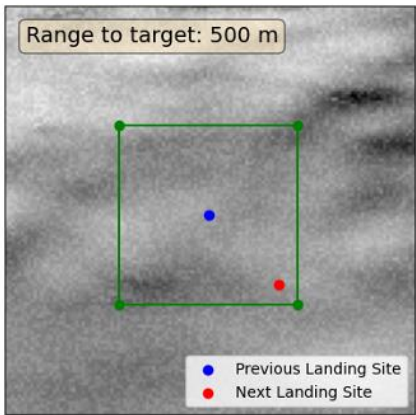
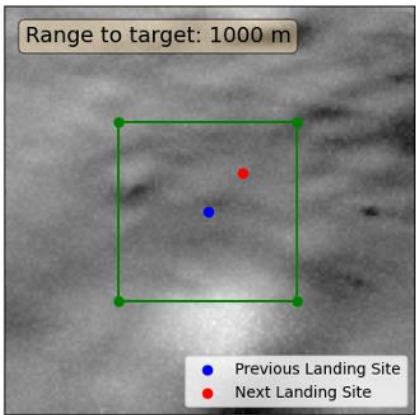
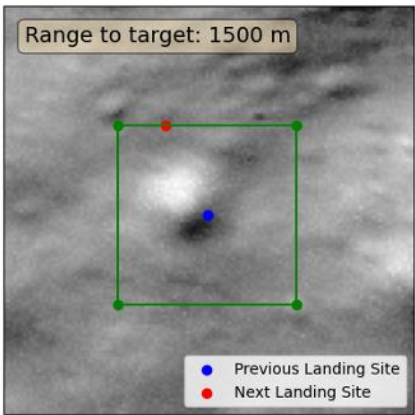
Seeker camera



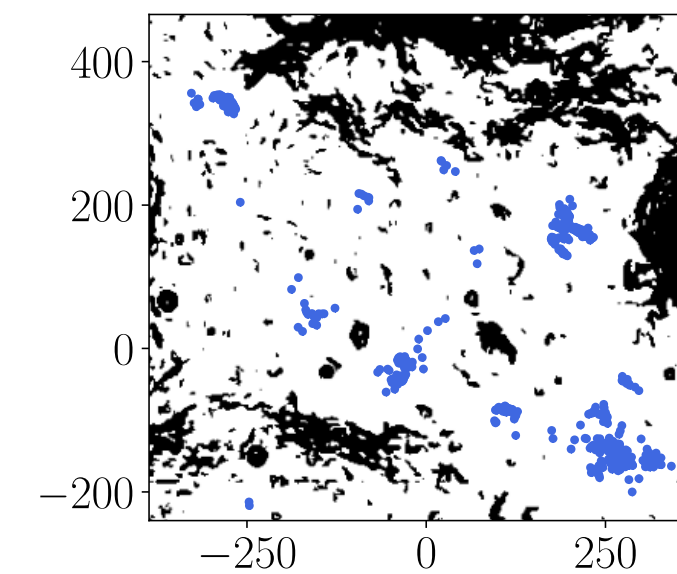
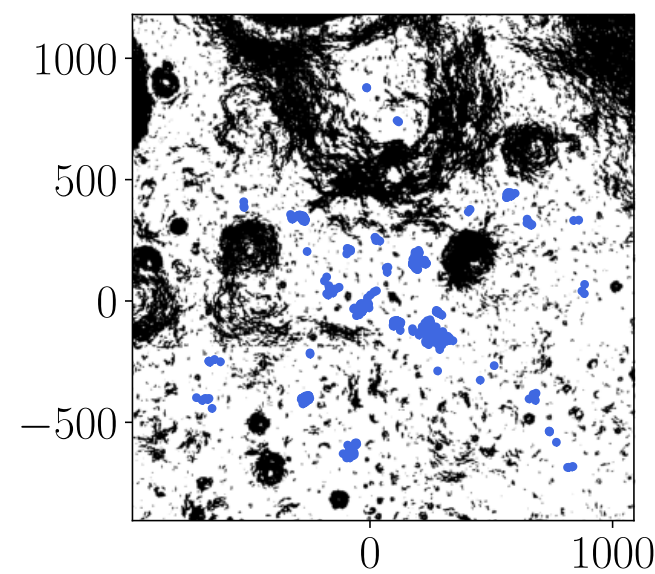
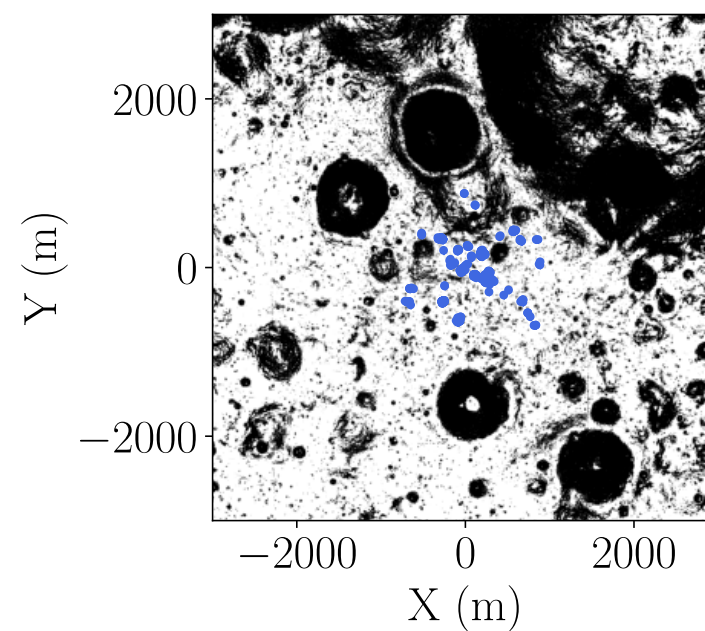
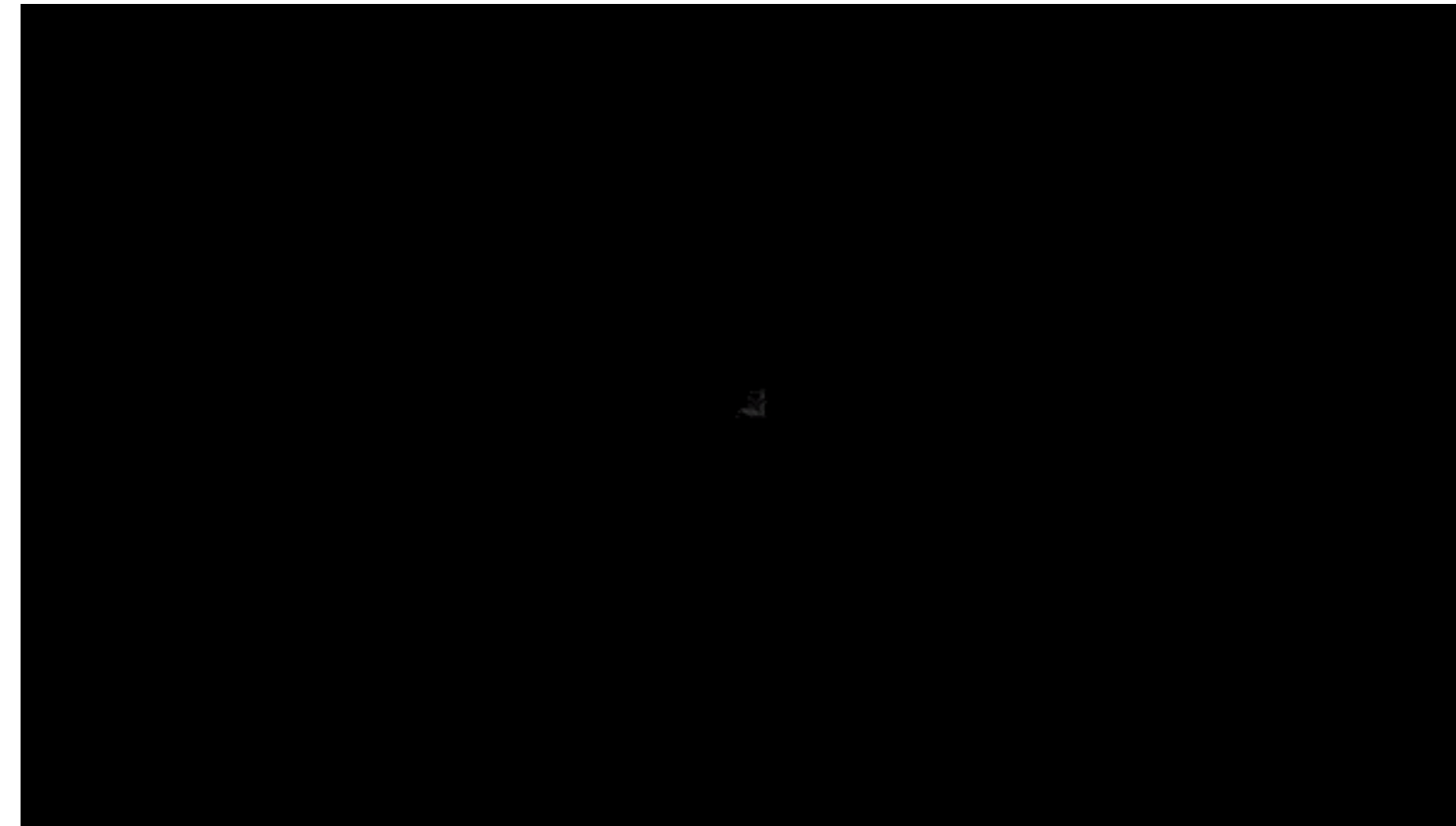
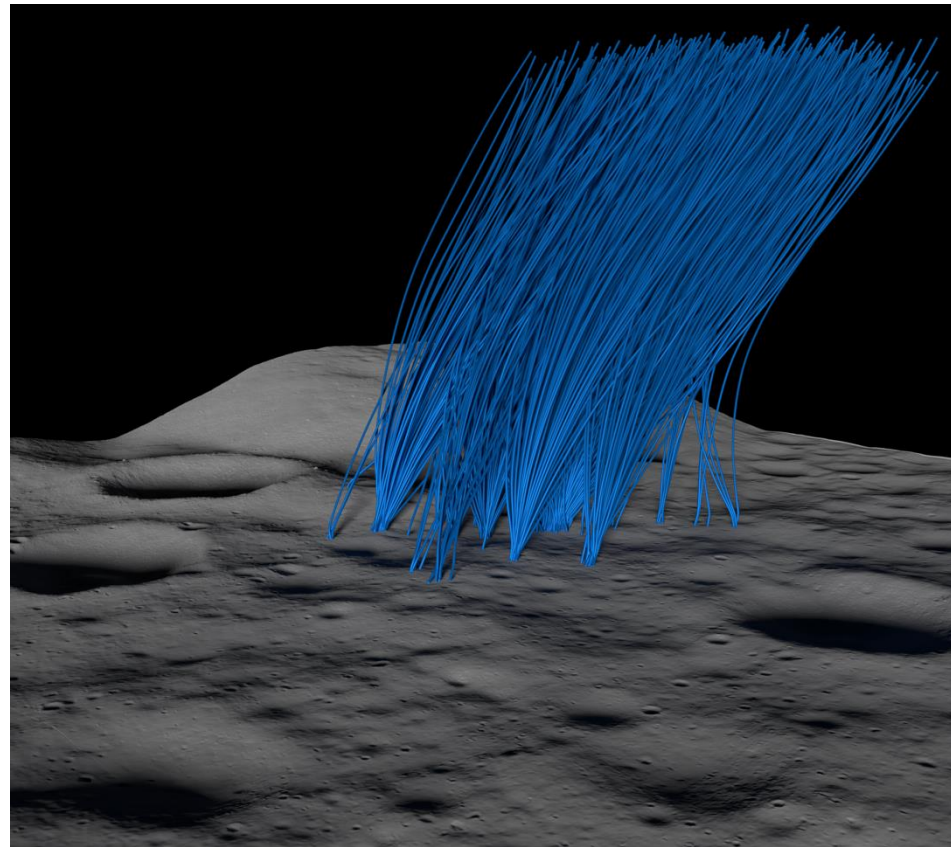
Body-fixed camera



Divert maneuvers



LSSS Performance (II)



Scorsoglio A., Gaudet, B., Ghilardi, L. and **Furfaro, R.**, 2025. Meta-reinforcement learning guidance, navigation, and control for autonomous lunar landing with safe site selection. *Neural Computing with Application*- pp.1-30



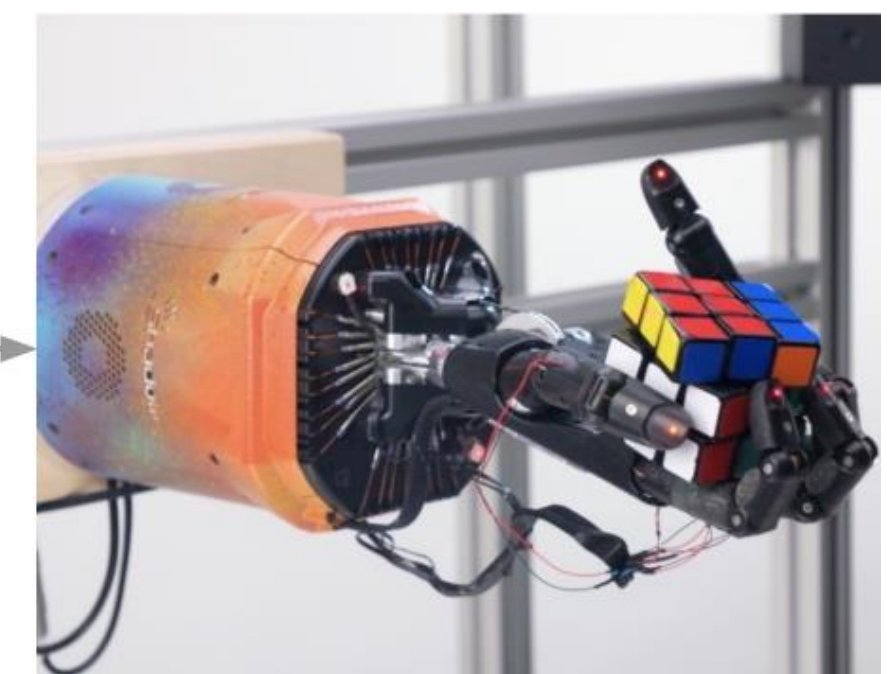
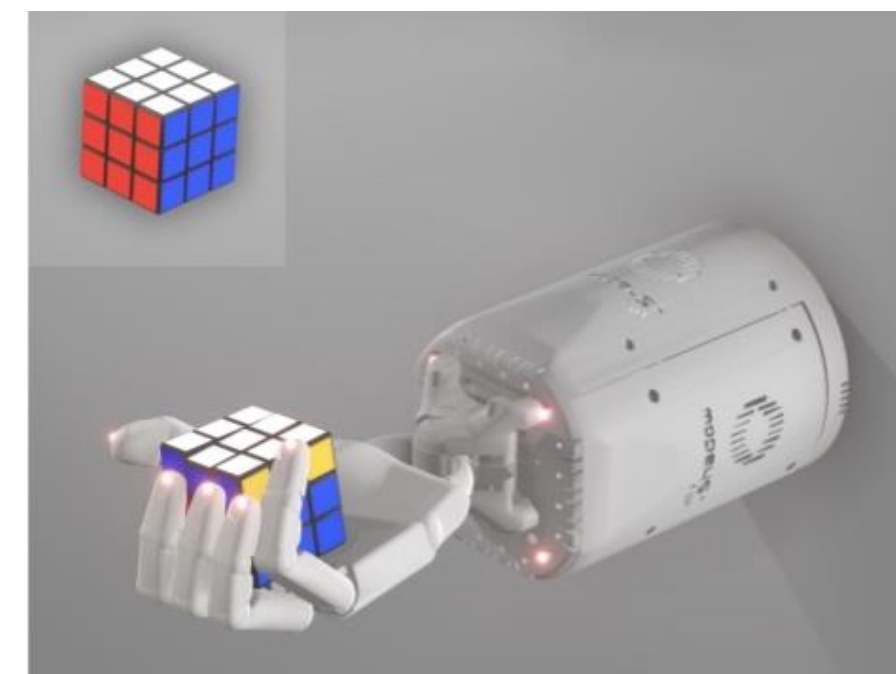
Meta-RL: Challenges and Future Directions

- **Challenges**

- RL-based GNC for aerospace systems are currently trained in **simulated environments**
 - **Expensive to train** in operational environments
- RL-based systems are **not certifiable** with standard practice
 - No **Gain/Phase Margin** available

- **Present/Future needs**

- **Safe RL & Meta-RL:** Methods to constrain behavior during training scenarios
- **Trustworthy/Certifiable RL:** methods to ensure convergence & performance guarantees
- We discuss **two approaches**
 - **Guidance Gain Adaptation** via Meta-RL
 - **Stability Certification** via **Sparse Identification of Non-Linear Dynamics (SINDy)**



Fractional-Polynomial Powered Descent Guidance (FP^2DG): The Analytical Backbone for Gain Adaptation



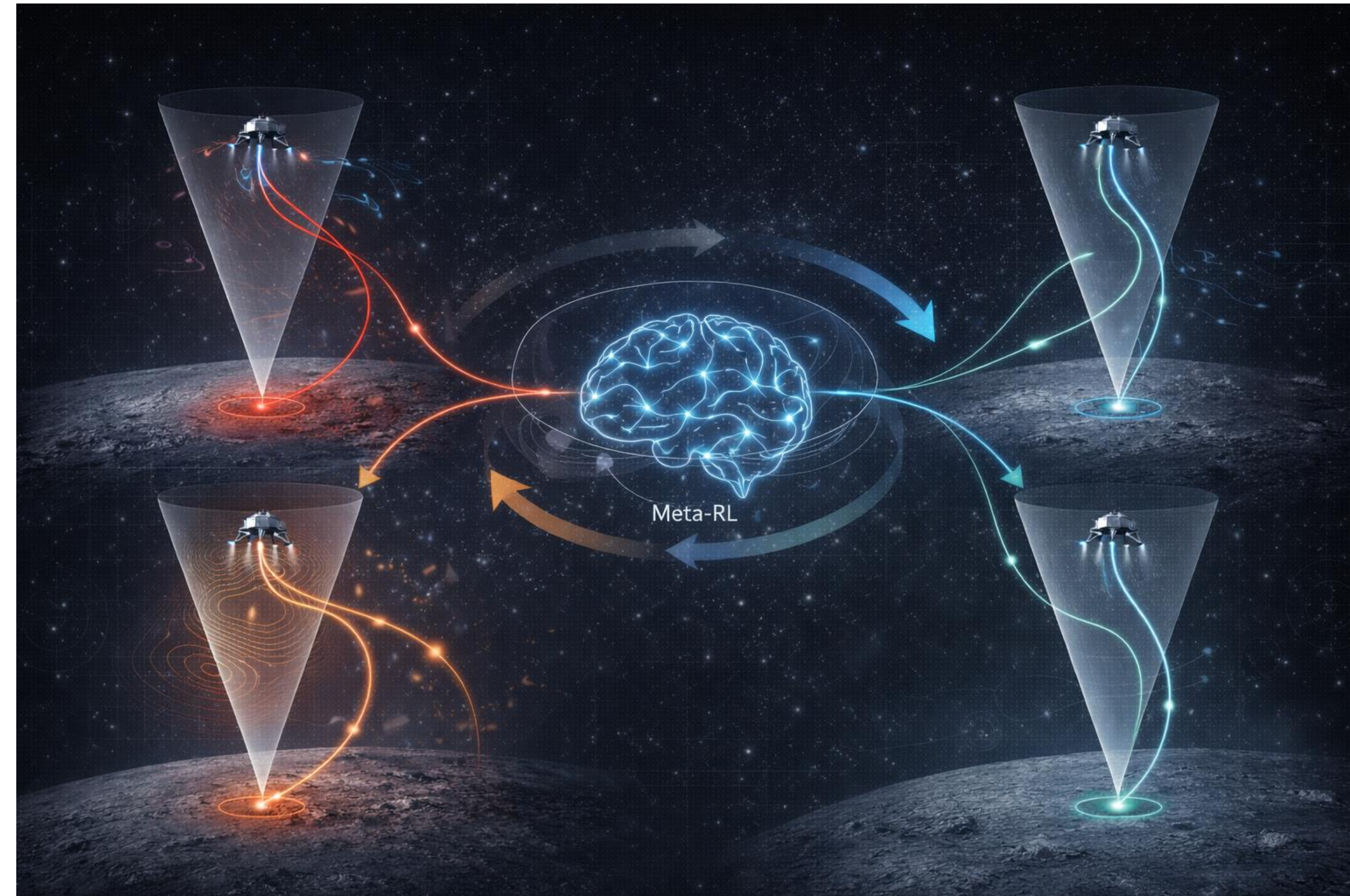
- Introduced by Ping Lu (2019, JGCD) the general FP^2PDG generalizes the heritage into a two-parameters family
- **General approach:** Given the current state $\mathbf{r}(t), \mathbf{V}(t)$ solve for $\mathbf{c}_1, \mathbf{c}_2$ using the thrust acceleration profile $\mathbf{a}_T = \mathbf{c}_1 \phi_1(t_{go}) + \mathbf{c}_2 \phi_2(t_{go})$. Set the reference acceleration profile to be $\mathbf{a}_d = \mathbf{a}_{T_f}^* + \mathbf{c}_2 t_{go}^\gamma$ into the tracking law on arrives to the closed-loop acceleration command

$$\mathbf{a}_T = \gamma \left[\frac{k_r}{2(\gamma + 2)} - 1 \right] \mathbf{a}_{T_f}^* + \left[\frac{\gamma k_r}{2(\gamma + 2)} - \gamma - 1 \right] \mathbf{g} + \frac{(\gamma + 1)}{t_{go}} \left(1 - \frac{k_r}{\gamma + 2} \right) (\mathbf{V}_f^* - \mathbf{V}) + \frac{k_r}{t_{go}^2} (\mathbf{r}_f^* - \mathbf{r} - \mathbf{V} t_{go})$$

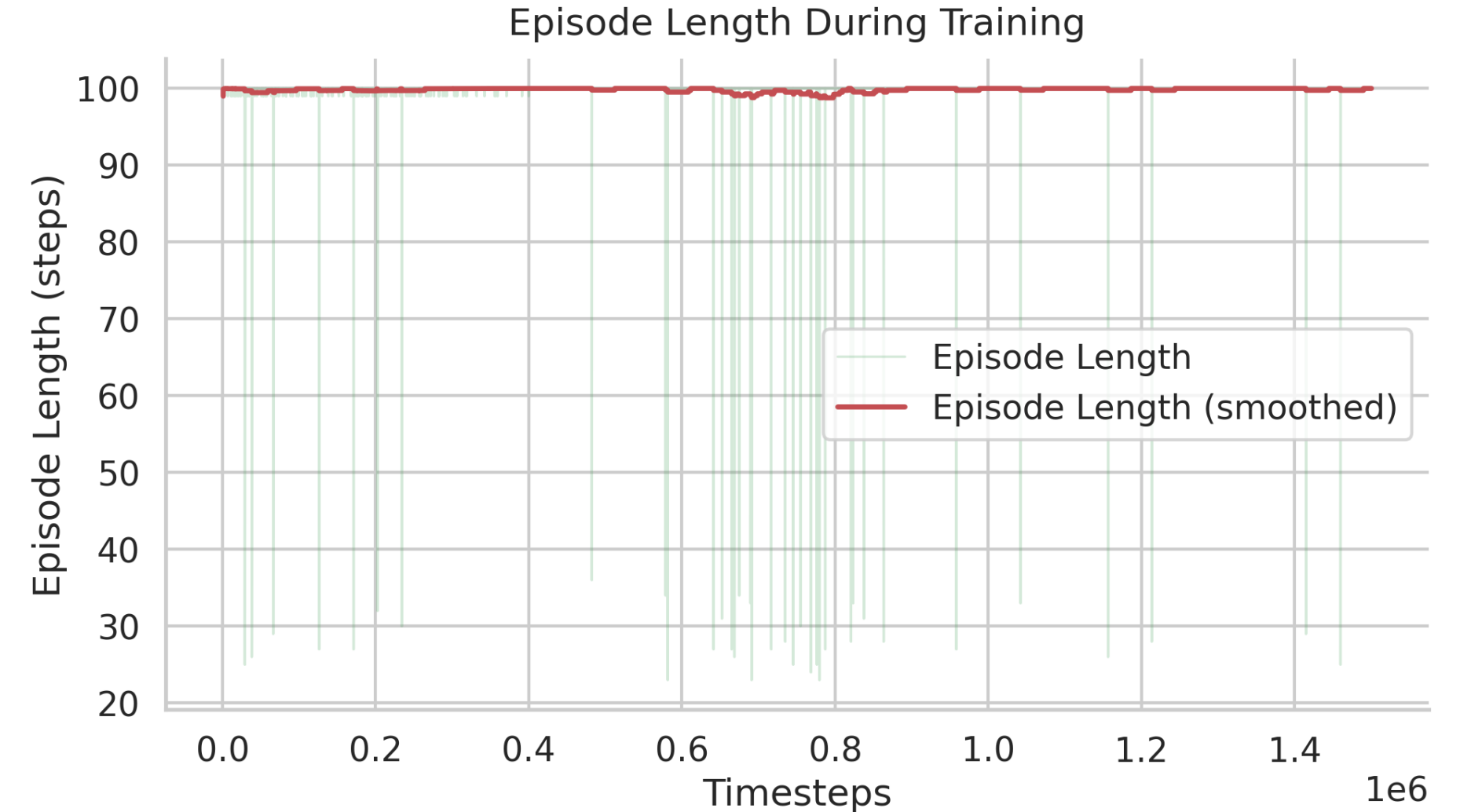
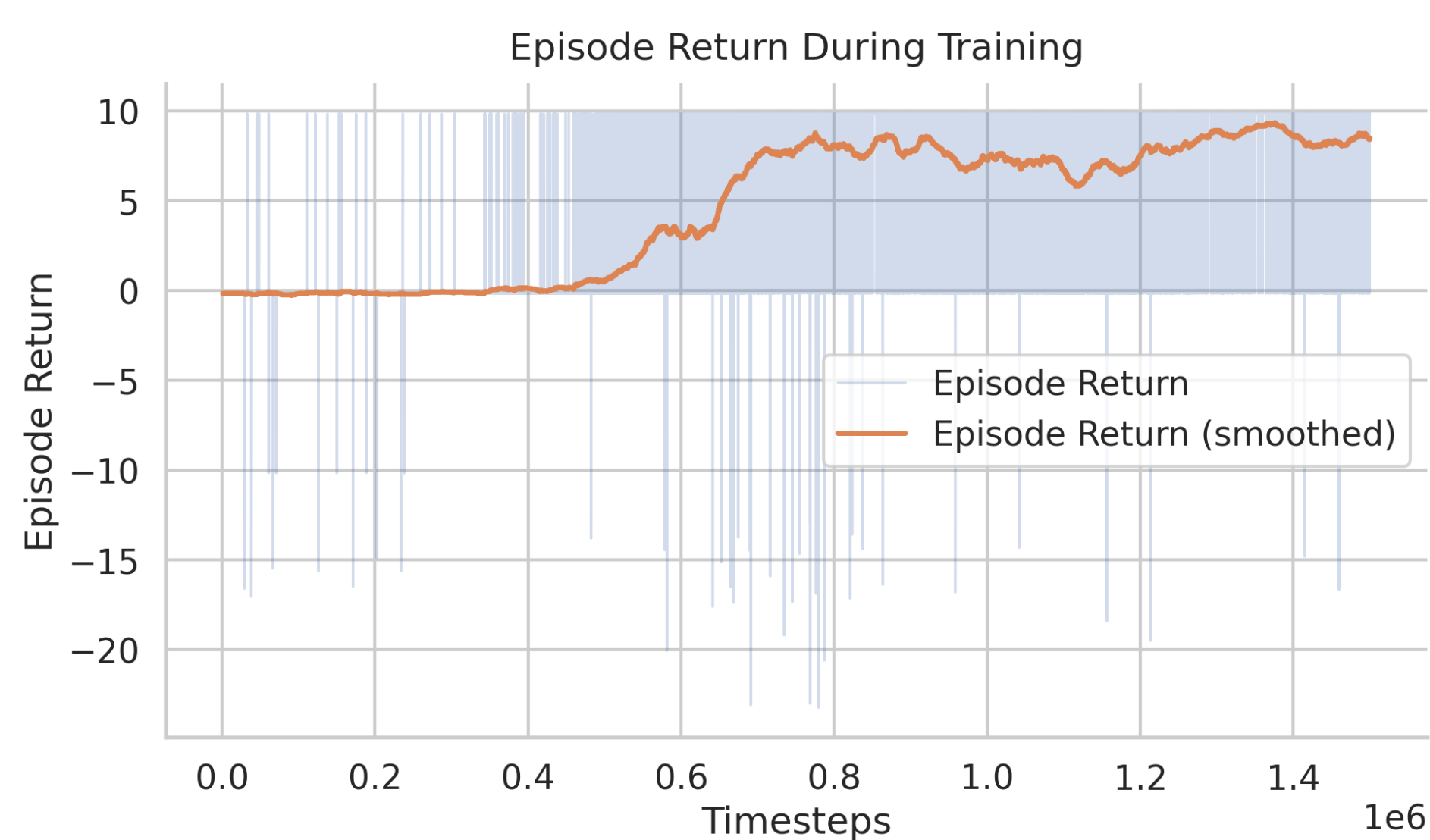
- This equation represents a family of two parameters guidance laws for $\gamma > 0, k_r \geq 2(\gamma + 2)$

Approach: Meta-RL tuning of the Guidance Gains ▶▶▶

- FP²DG defines a **family of guidance behaviors** parameterized by k_r and γ
- Static tuning selects a **single fixed behavior**, which can degrade under constraints and uncertainty
- Real powered descent involves **changing conditions**: constraints, disturbances, and model mismatch
- *Meta-RL learns a state-dependent scheduling policy across a distribution of scenarios*
- The controller adapts $k_r(t)$ and $\gamma(t)$ online using state history (memory)
- This enables robust, **constraint-aware guidance while preserving the analytical structure**



Learning Behavior: Return Improves and Episode Stabilizes ▶▶▶



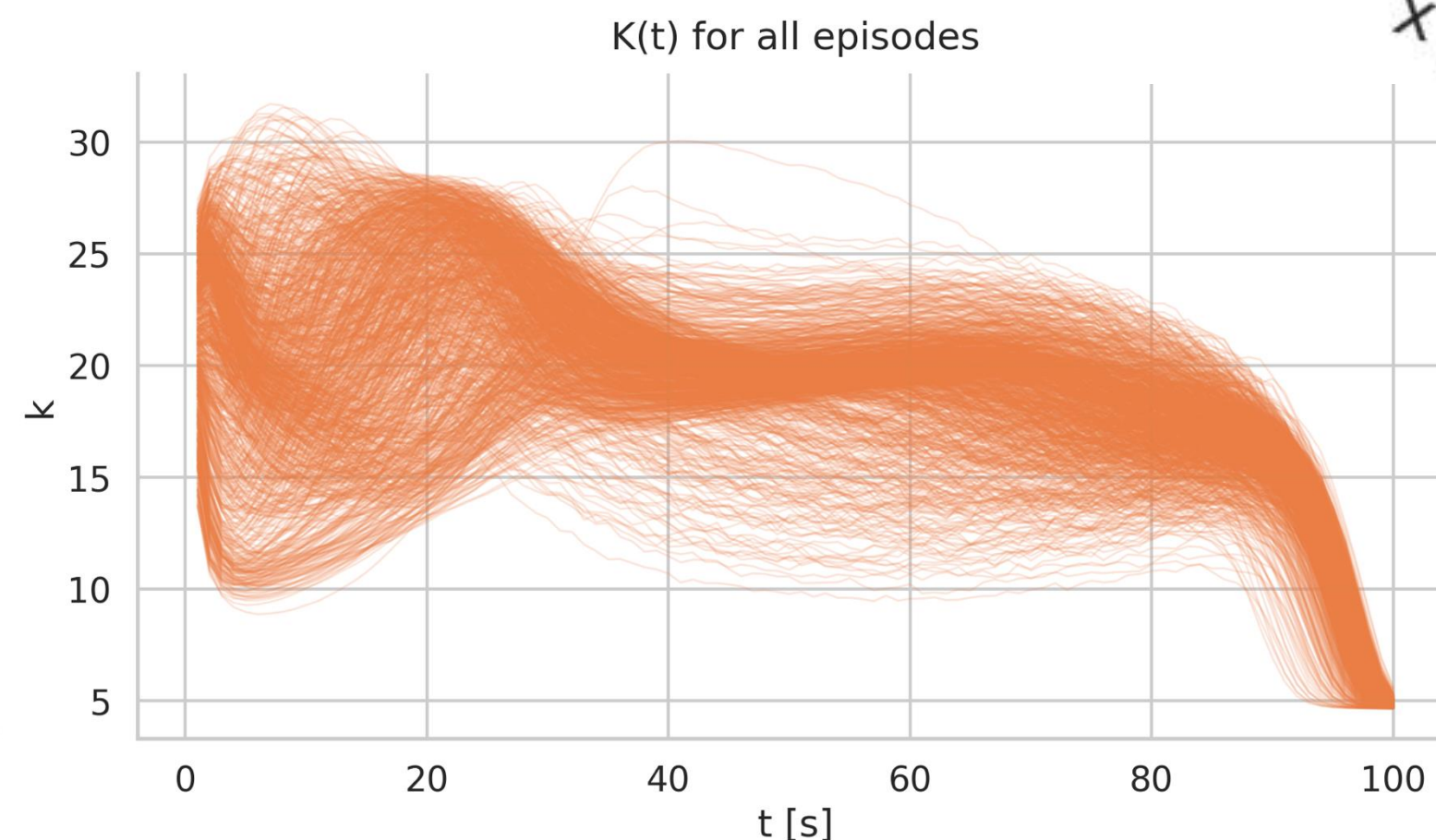
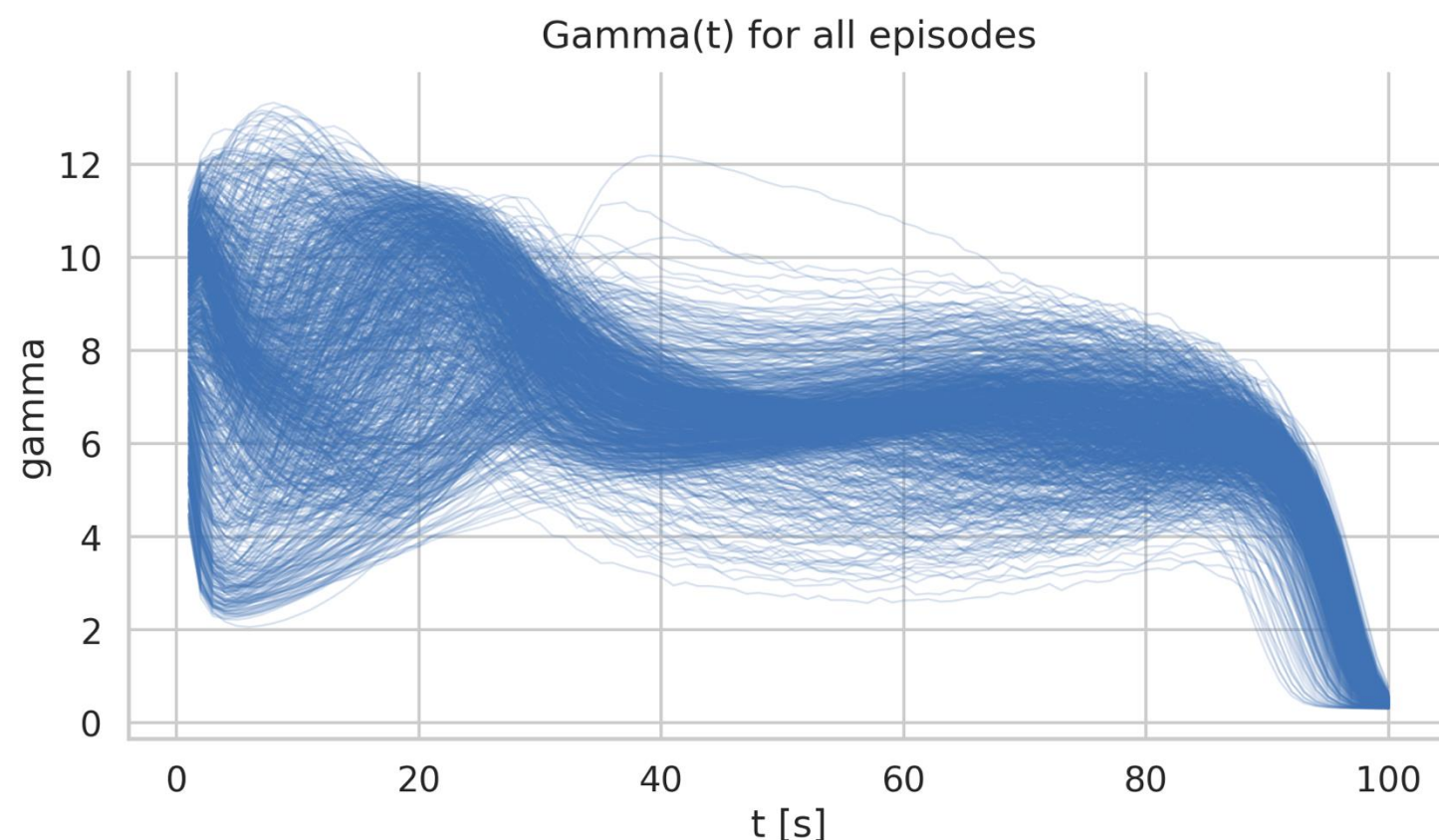
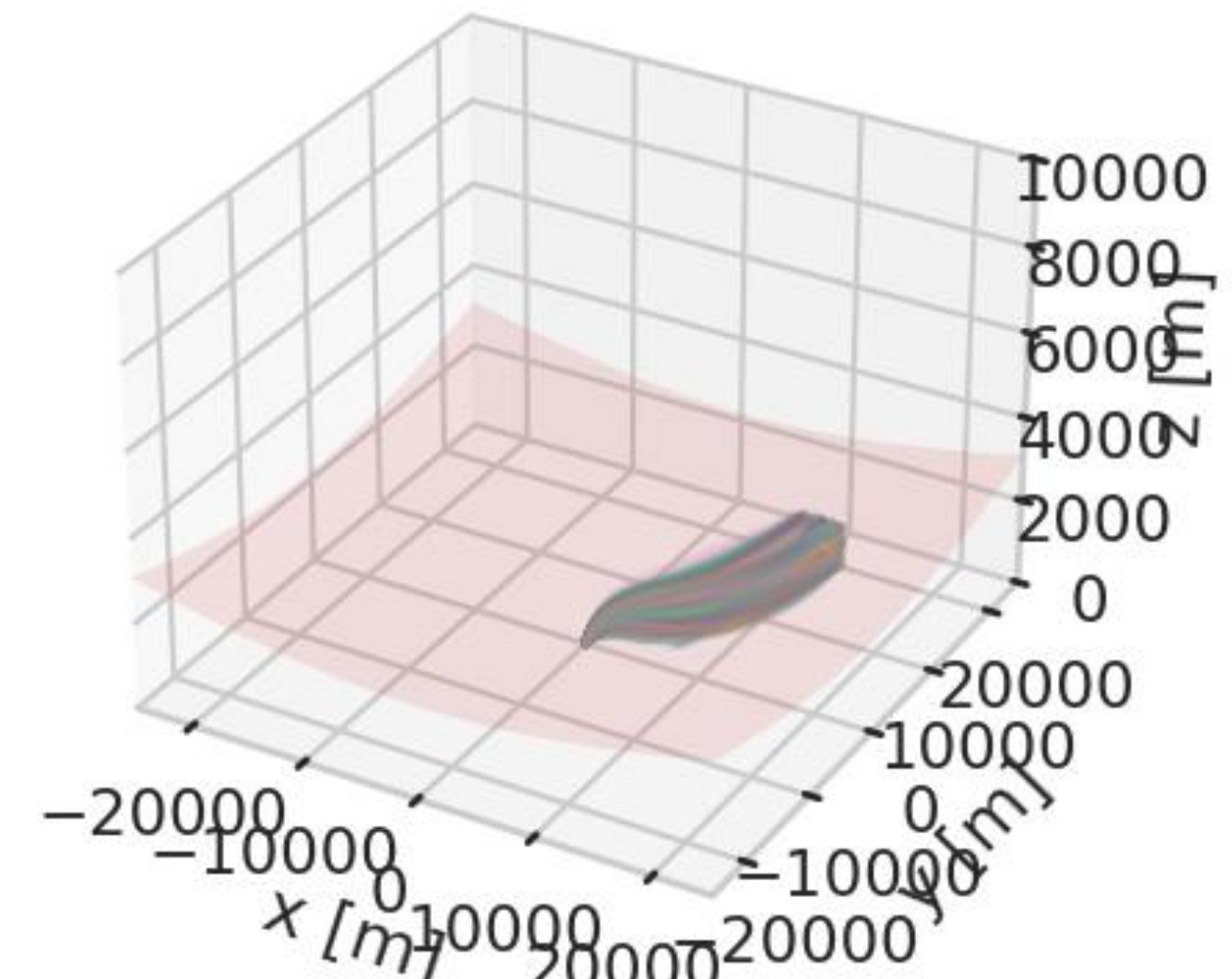
- **Training return improves** after the early exploratory phase.
- **Episode lengths converge** toward the full-horizon regime as successful landings dominate.
- The base constrained case is **the hardest of the final runs**.

What Does the Recurrent Scheduler Learns



- **Trajectory ensemble** indicates that the learned scheduler generates a **coherent family of descent paths**
- **Recurrent policy does not converge to a constant pair k_r, γ** . It learns smooth state-dependent schedules.

3D Trajectories



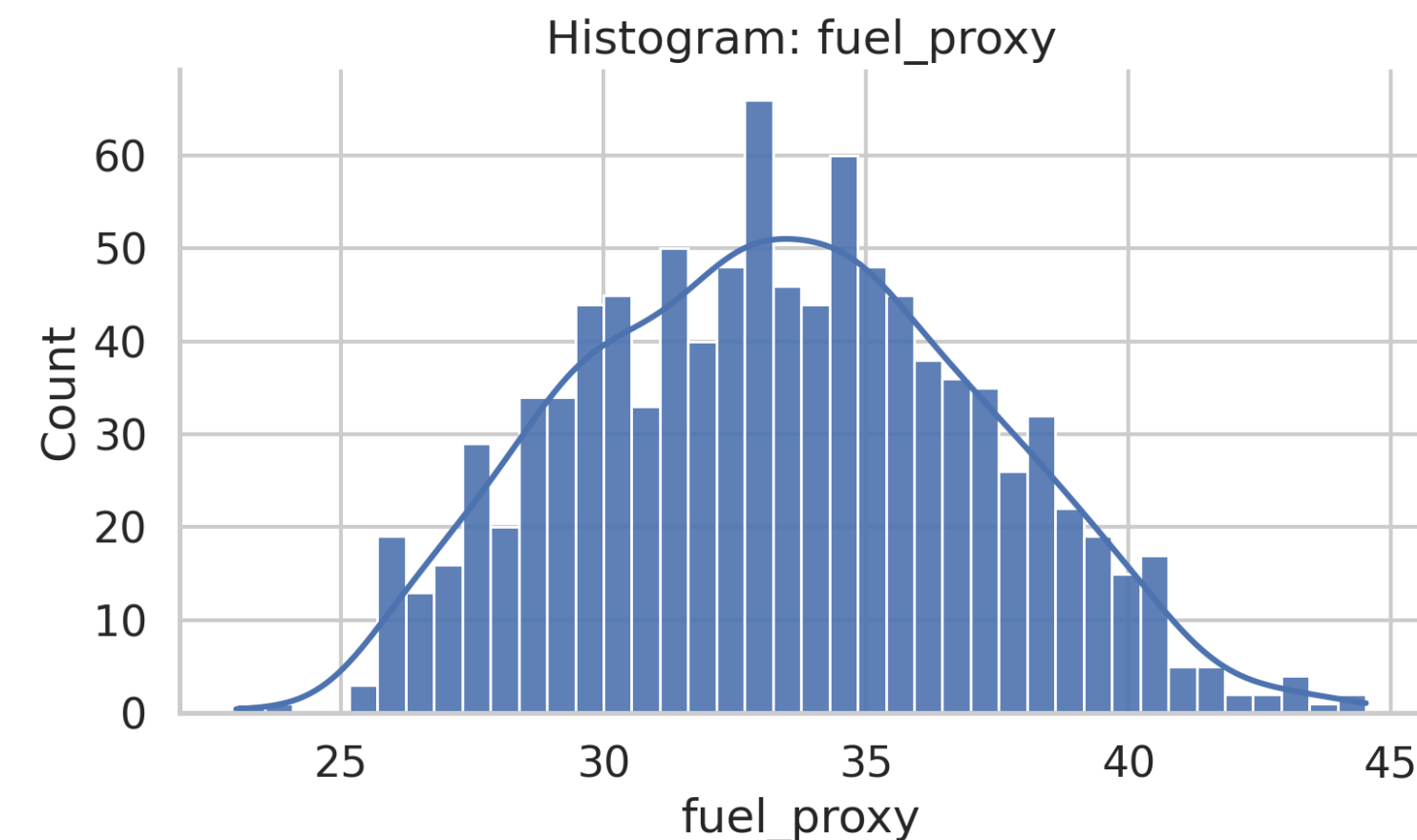
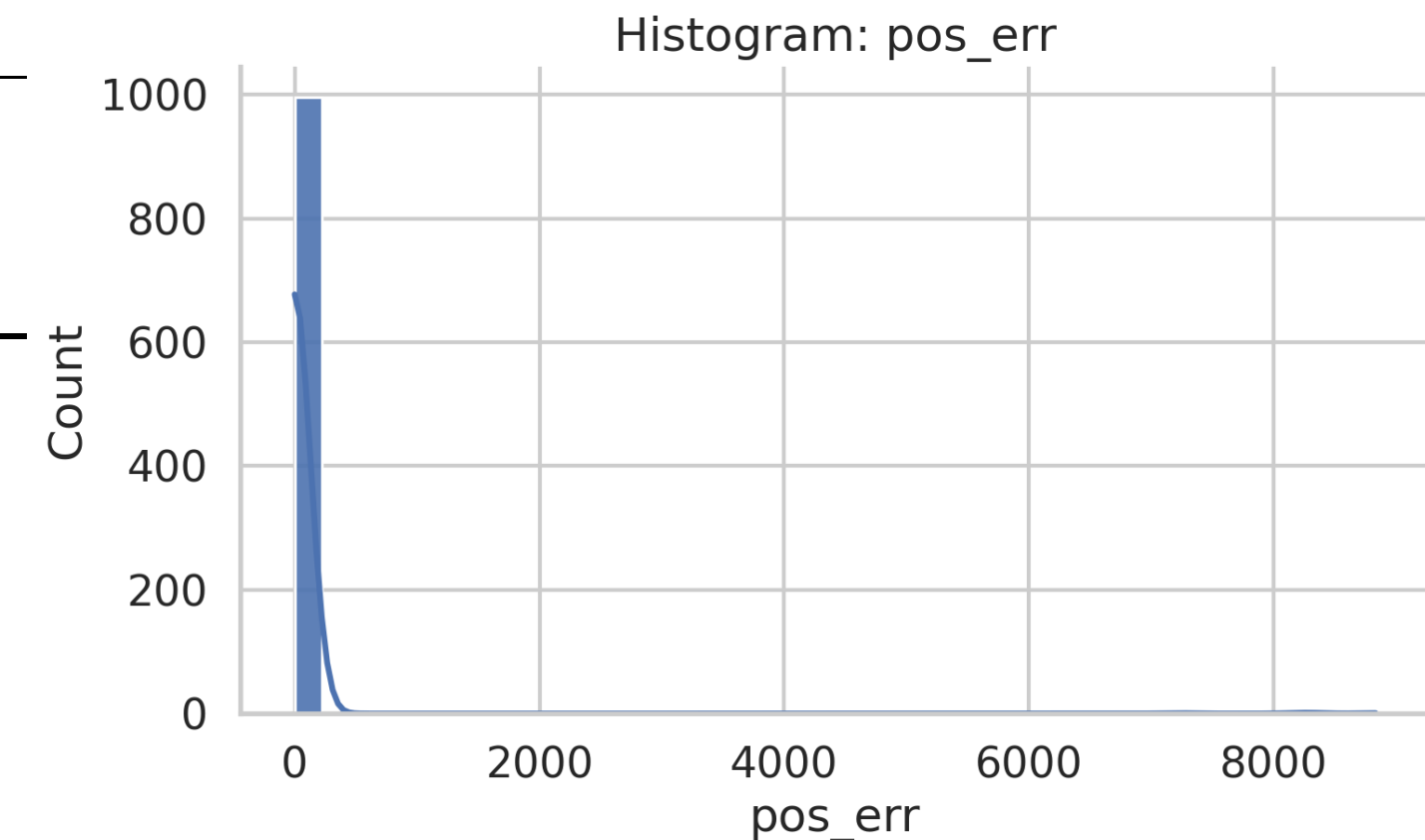
Quantitative Summary from the Saved Final Runs

Run	success rate	$\ r - r_f\ $ [m]	$\ v - v_f\ $ [m/s]	fuel proxy [kg]
FINAL_constraint	0.856	0.044 ± 0.027	0.185 ± 0.102	15328 ± 594
FINAL_higher_thrust	0.992	0.027 ± 0.016	0.119 ± 0.066	15668 ± 545
FINAL_initial_condition	0.990	0.040 ± 0.023	0.178 ± 0.096	14767 ± 629

- The **strongest runs** achieve about **99% late training success**.
- Successful late-training episodes **land within centimeters** and comfortably below the **coded terminal-velocity threshold**.
- Easier initial conditions **reduce fuel proxy**; larger thrust authority **tightens terminal control**.

Table 1: Key setup parameters extracted from the saved final run configurations.

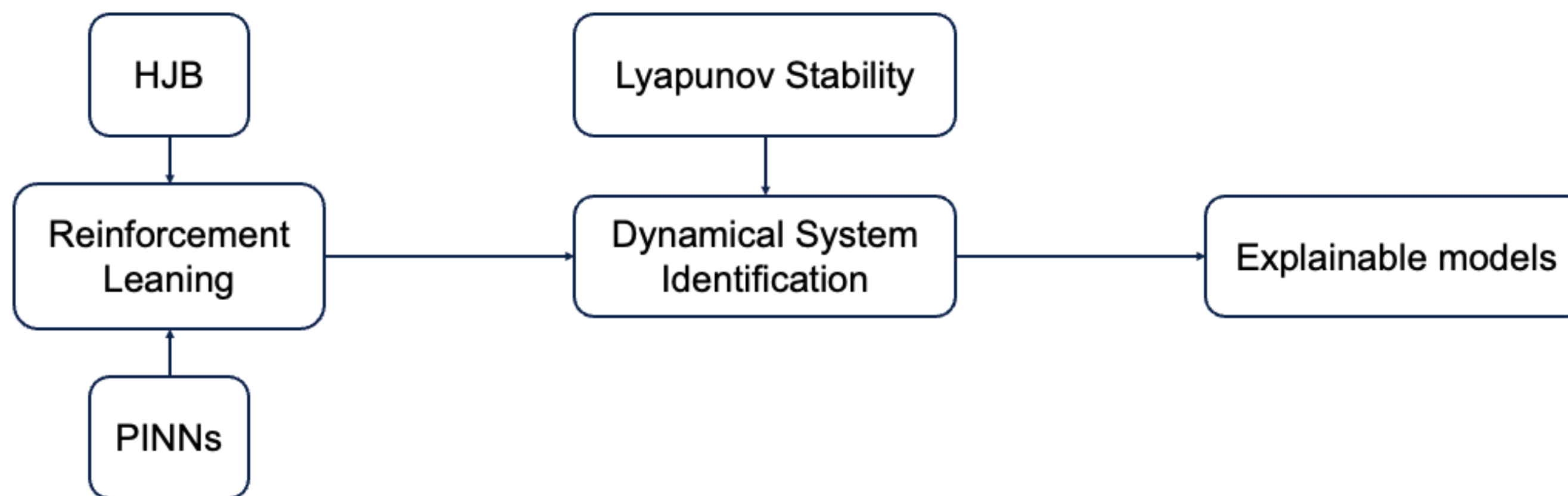
Run	$T_{\max}$ [kN]	θ_{gs} [deg]	nominal v_{0z} [m/s]	best eval mean return
FINAL_constraint	800	85	-140	9.855
FINAL_higher_thrust	1000	80	-140	9.817
FINAL_initial_condition	800	80	-100	9.860

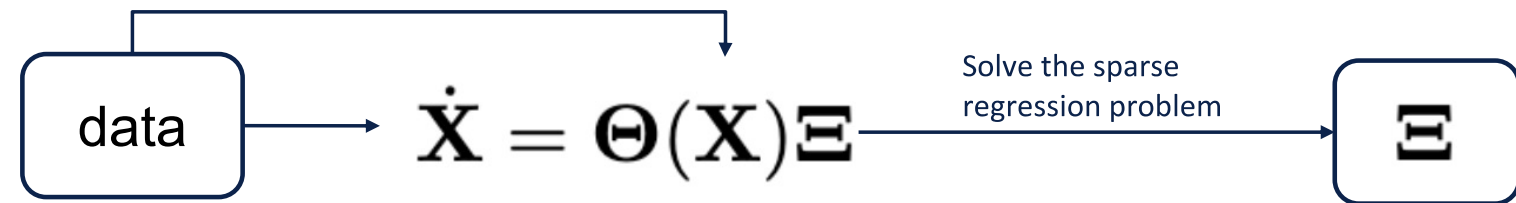


System Certification: Physics-Informed Reinforcement Learning & Lyapunov Function



- **Stability of RL-based agents** is rarely studied and certified → needed for wider adoption
- Systematic procedure for ensuring theoretical guarantees during and after training.
- **Idea:** Use **PINNs**, **Lyapunov stability** and **dynamical system identification** to create a procedure for training and certifying neural networks to increase their **safety** and **explainability**.





8

- $$\dot{x}_i = \begin{cases} \dot{x}_1 = f(x_1, x_2, x_3, x_4) \\ \dot{x}_2 = g(x_1, x_2, x_3, x_4) \\ \dot{x}_3 = h(x_1, x_2, x_3, x_4) \\ \dot{x}_4 = k(x_1, x_2, x_3, x_4) \\ \dot{V} = q(x_1, x_2, x_3, x_4) \\ \frac{d}{dt} \left(\int V dt \right) = z(x_1, x_2, x_3, x_4) \end{cases}$$

Check Lyapunov stability

$$z(x_1, x_2, x_3, x_4) > 0$$
$$q(x_1, x_2, x_3, x_4) < 0$$

- $$z(x_1, x_2, x_3, x_4) > 0$$
- $$\dot{V} = \left(\frac{\partial z}{\partial \mathbf{x}} \right)^T \cdot (\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u}_{RL}(\mathbf{x})) < 0$$
- 

Identified system



RL agent output

The Axisymmetric Satellite Attitude Control Problem

Euler equations for axisymmetric spacecraft

$$\dot{x}_1 = amx_2 + u_1$$

$$\dot{x}_2 = -amx_1 + u_2$$

$$\dot{x}_3 = mx_4 + x_2x_3x_4 + (x_1/2)(1 + x_3^2 - x_4^2)$$

$$\dot{x}_4 = -mx_3 + x_1x_3x_4 + (x_2/2)(1 + x_4^2 - x_3^2)$$

With:

$$\mathbf{x} = [x_1, x_2, x_3, x_4] = [\omega_1, \omega_2, w_1, w_2]$$

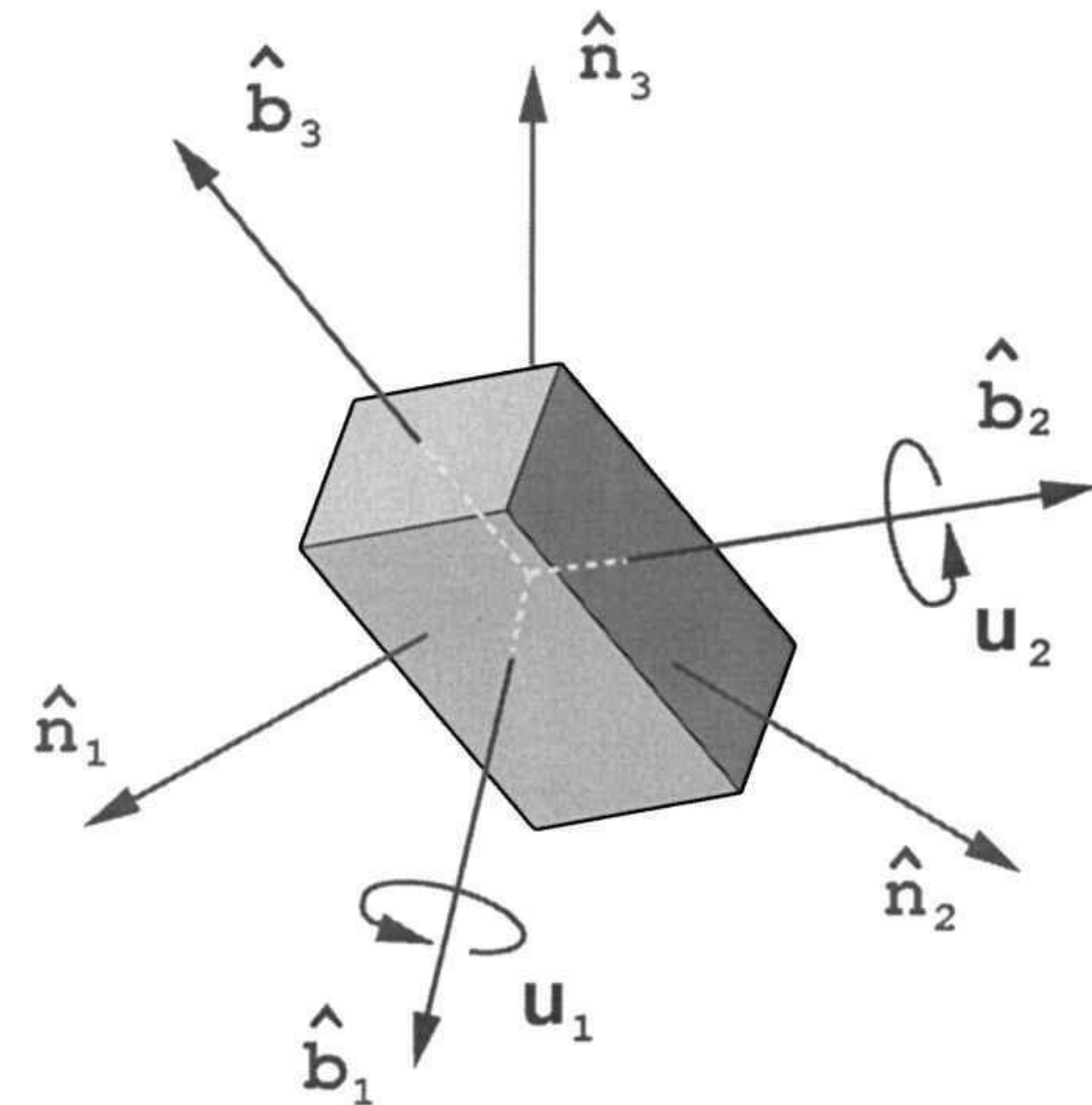
$$\dot{\omega}_1 = a\omega_2\omega_3 + u_1$$

$$\dot{\omega}_2 = -a\omega_1\omega_3 + u_2$$

$$w_1 = \frac{\beta}{1+\gamma}$$

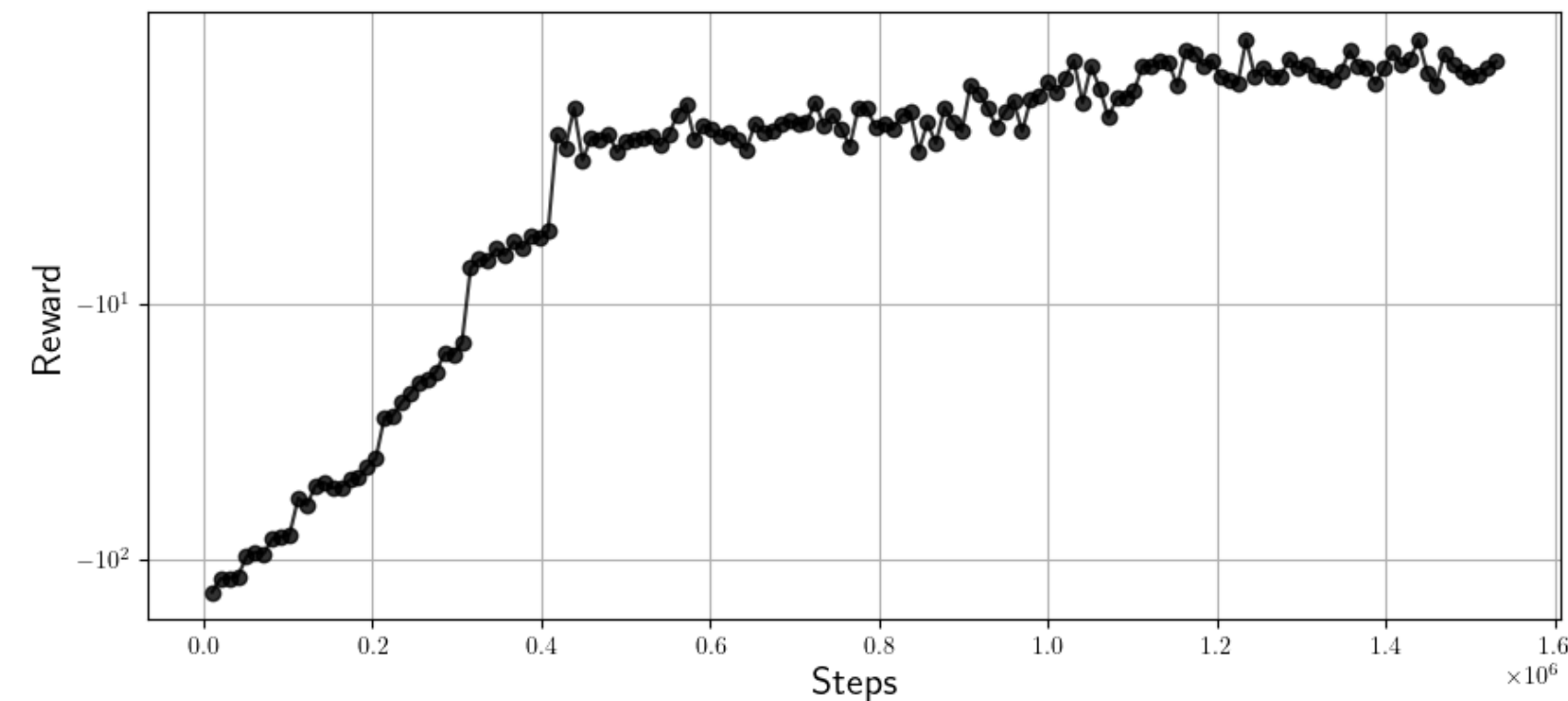
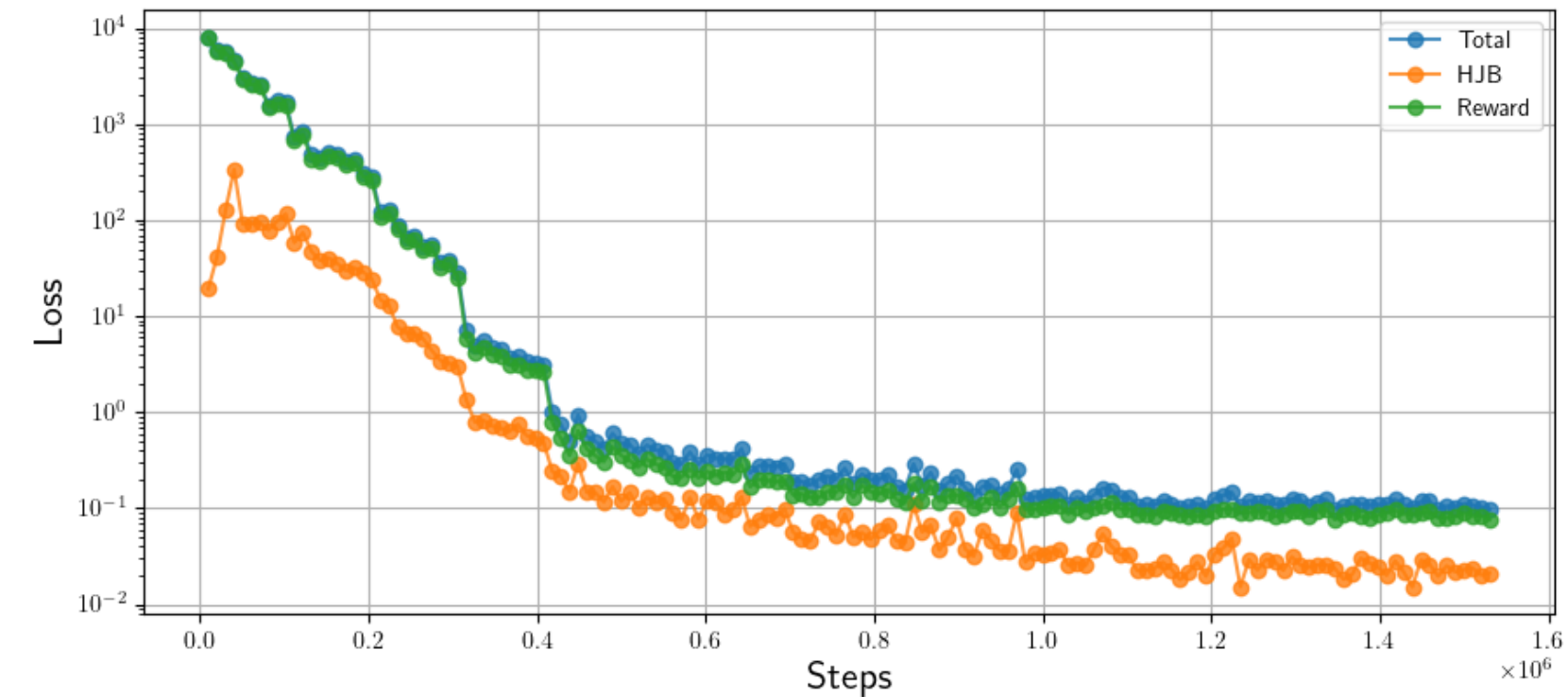
$$w_2 = -\frac{\alpha}{1+\gamma}$$

$$\hat{n}_3 = \alpha \hat{b}_1 + \beta \hat{b}_2 + \gamma \hat{b}_3$$



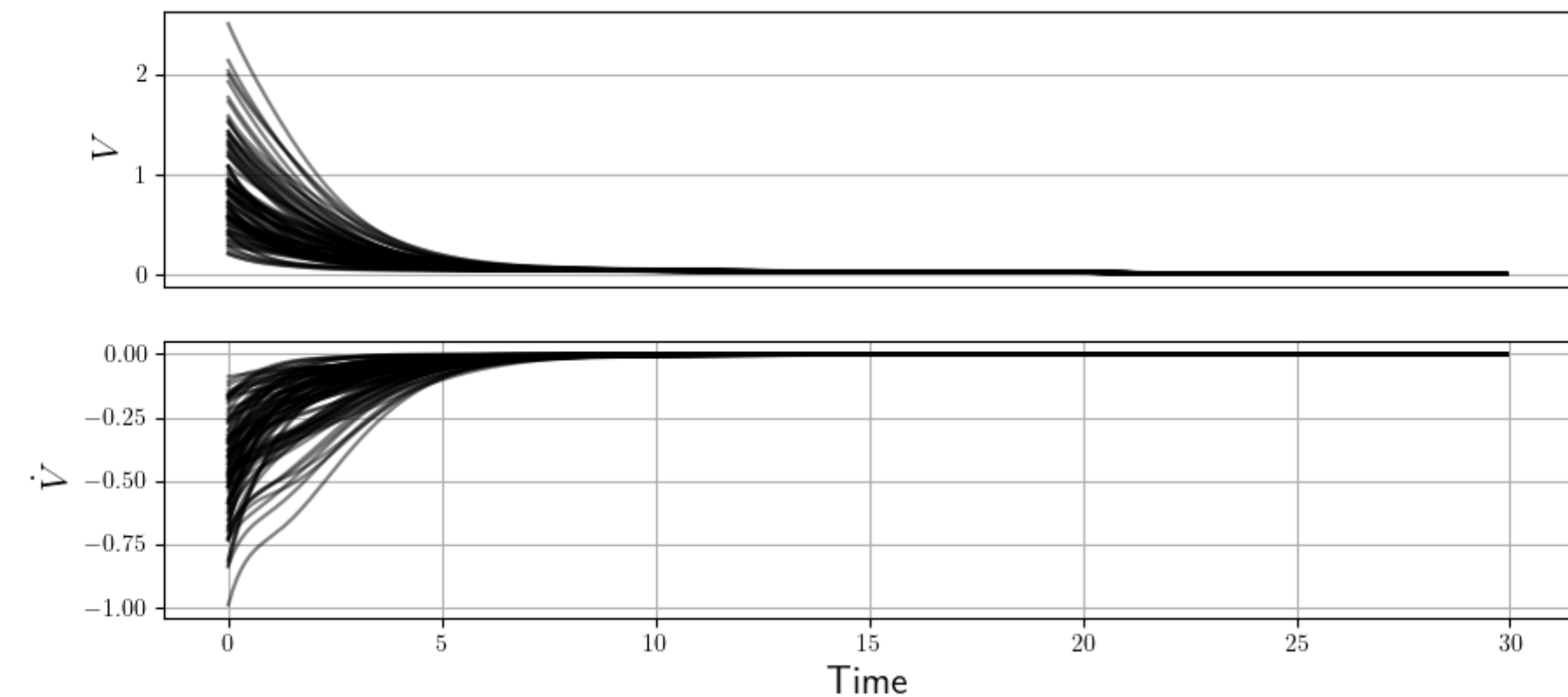
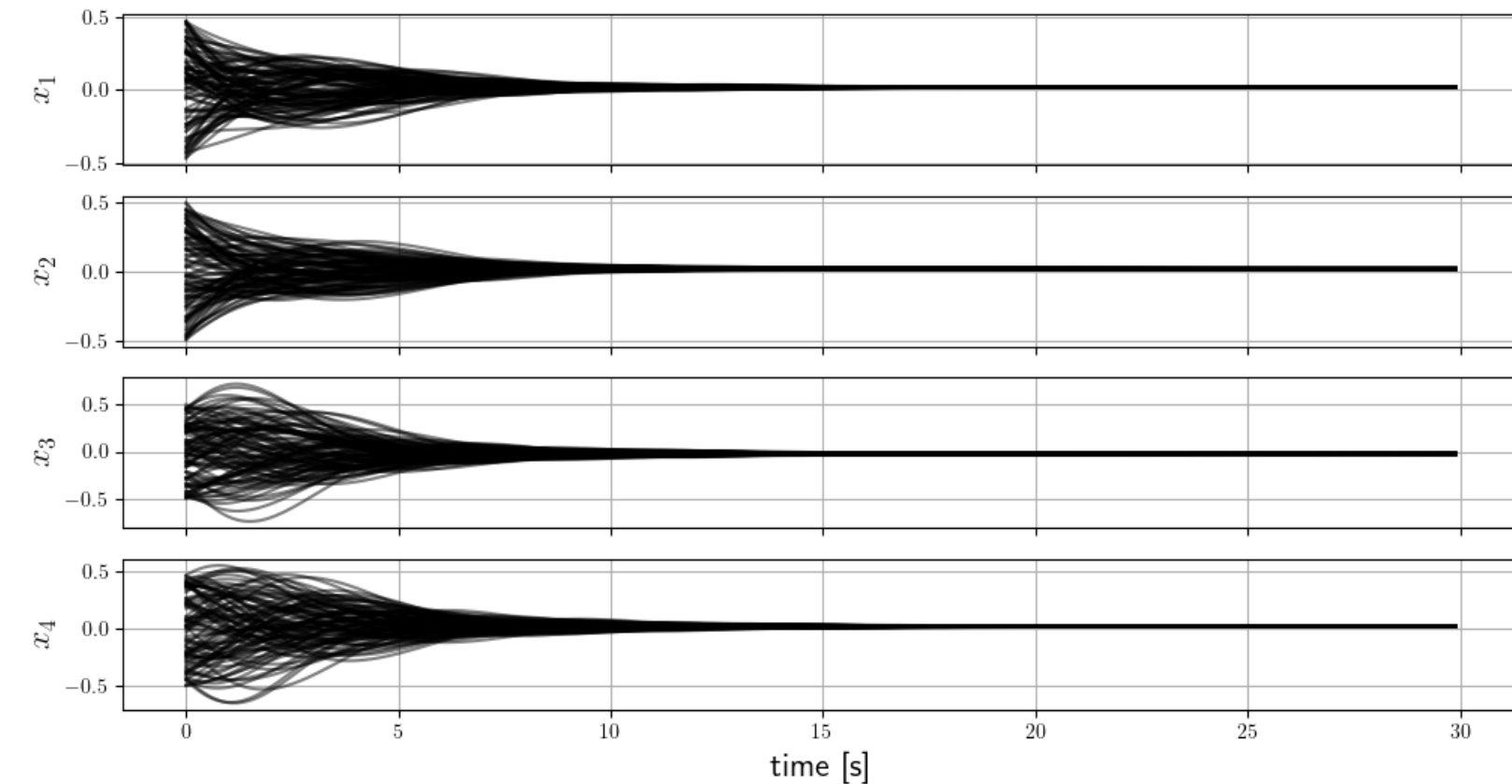
Numerical Results: RL Agent Training (I)

- Trained for **1.5M steps**
- Critic losses show that it **effectively learns** the value function
- The reward history reaches a **plateau**, showing convergence
- **Monte Carlo simulation** run after training to generate a dataset for certification process
 - 100 trajectories
 - Randomized initial conditions



Numerical Results: RL Agent Training (II)

- Trained for **1.5M steps**
- Critic losses show that it **effectively learns** the value function
- The reward history reaches a **plateau**, showing convergence
- **Monte Carlo simulation** run after training to generate a dataset for certification process
 - 100 trajectories
 - Randomized initial conditions

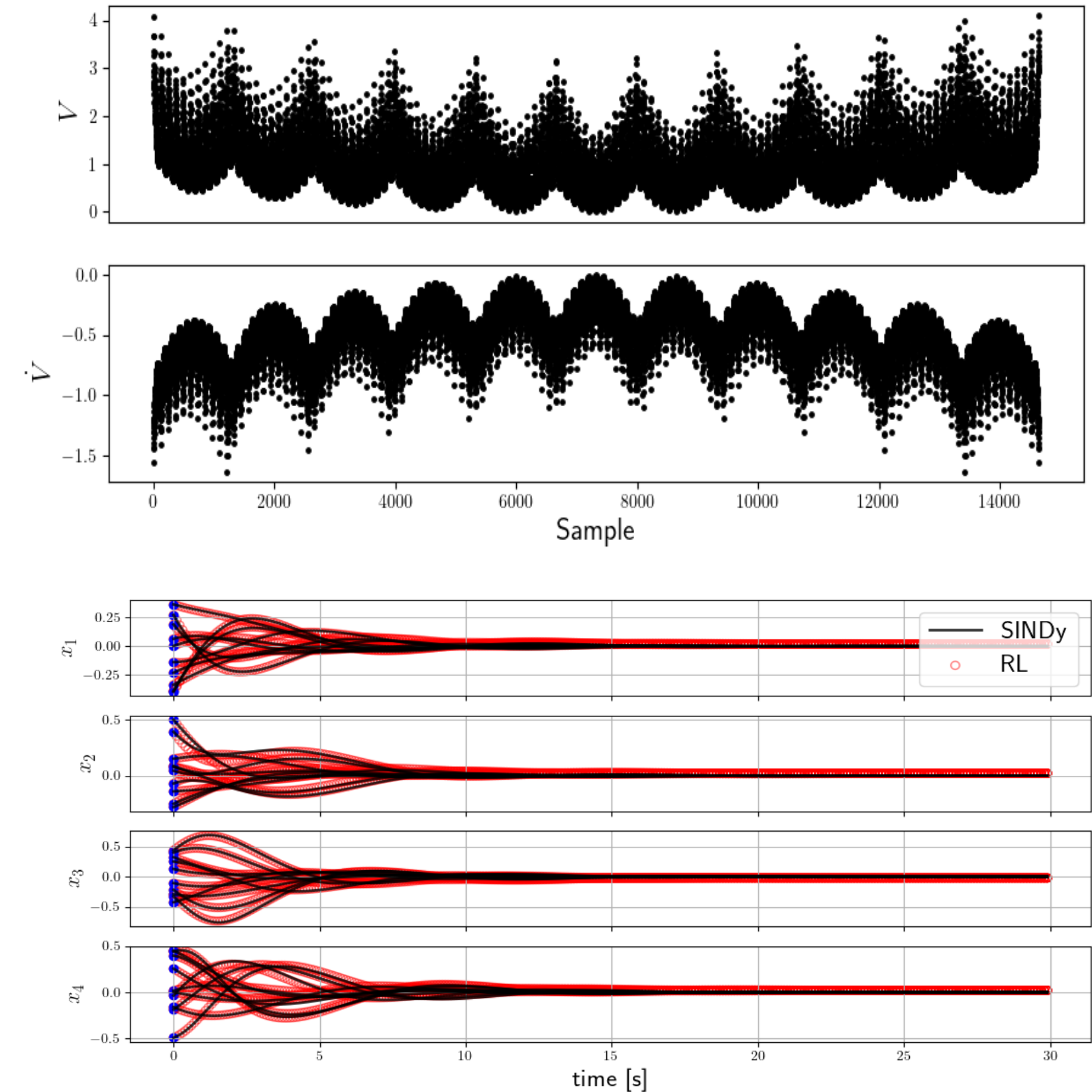


Numerical Results: SINDy identification of the Closed-Loop Dynamics



- Model constrained to 3rd order. Coefficient threshold set to 0.1.

$$\begin{aligned}
 \dot{x}_1 &= f(x_1, x_2, x_3, x_4) = -0.735x_1 - 0.360x_3 + 0.213x_4 + 0.364x_2x_3x_4 \\
 \dot{x}_2 &= g(x_1, x_2, x_3, x_4) = -0.568x_2 - 0.187x_3 - 0.263x_4 \\
 \dot{x}_3 &= h(x_1, x_2, x_3, x_4) = 0.5x_1 - 0.5x_4 + 0.5x_1x_3^2 - 0.5x_1x_4^2 + 1.0x_2x_3x_4 \\
 \dot{x}_4 &= k(x_1, x_2, x_3, x_4) = 0.5x_2 + 0.5x_3 + 1.0x_1x_3x_4 - 0.5x_2x_3^2 + 0.5x_2x_4^2 \\
 \left(\int \dot{V} dt \right) &= V = z(x_1, x_2, x_3, x_4) = 2.250x_1^2 + 2.331x_1x_3 - 0.854x_1x_4 + 2.592x_2^2 \\
 &+ 1.140x_2x_3 + 2.642x_2x_4 + 3.006x_3^2 + 0.138x_3x_4 + 3.150x_4^2 - 0.435x_1^2x_2 \\
 &- 0.316x_1^2x_4 + 0.283x_1x_3^2 + 0.204x_1x_3x_4 + 0.822x_2^2x_3 - 0.405x_2^2x_4 \\
 &+ 0.410x_2x_3^2 - 0.492x_2x_3x_4 - 0.154x_3x_4^2 + 0.170x_4^3 \\
 \dot{V} &= q(x_1, x_2, x_3, x_4) = -1.584x_1^2 - 0.606x_1x_3 + 0.145x_1x_4 - 1.431x_2^2 - 0.479x_2x_4 \\
 &- 1.154x_3^2 - 1.136x_4^2
 \end{aligned}$$



Scorsoglio, A., D'Ambrosio, and **Furfaro, R.**, 2026. Stability Explainable Stability Certification of Reinforcement Learning Control Policies Using Sparse Dynamics Identification, *Journal of Guidance, Control and Dynamics* (In-Press)

Numerical Results: Analytical/Explainable Asymptotically Stable Controller

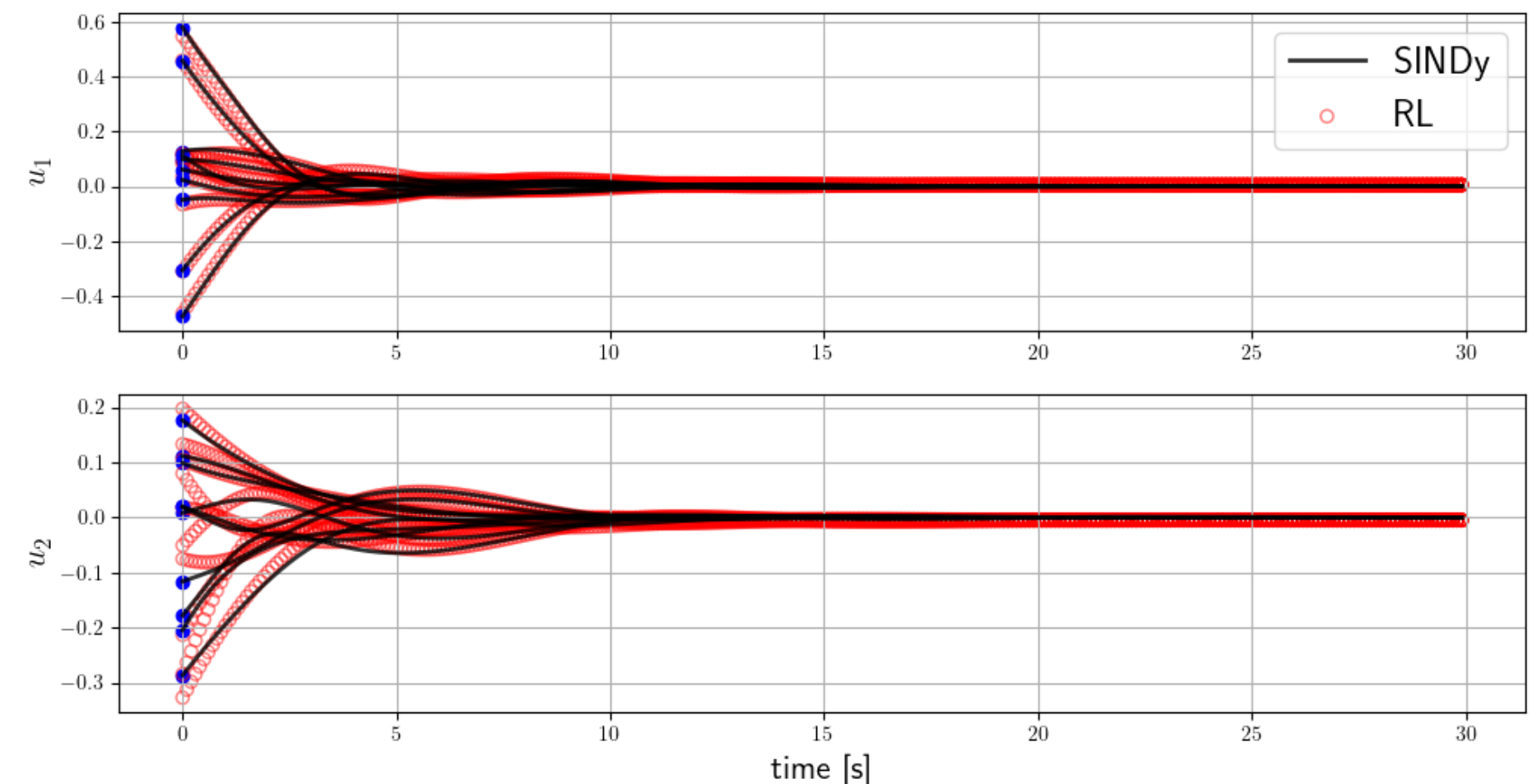


- Model V and $\dot{V}$ are positive and negative respectively on the entire domain, proving the recovered closed loop model is **Lyapunov Asymptotically Stable**.
- The recovered **control** is therefore guaranteed to be **Asymptotically Stable**.
- The control law is recovered by subtracting the known equations of motion from the recovered model:.

$$u_1(x_1, x_2, x_3, x_4) = -0.735x_1 - 0.360x_3 + 0.213x_4 + 0.364x_2x_3x_4 + 0.250x_2$$

$$u_2(x_1, x_2, x_3, x_4) = -0.568x_2 - 0.187x_3 - 0.263x_4 - 0.250x_1$$

- RL in this case is used to **discover the control law**, which is then recovered using SINDy. The resulting control has a **simple and explainable analytical form**



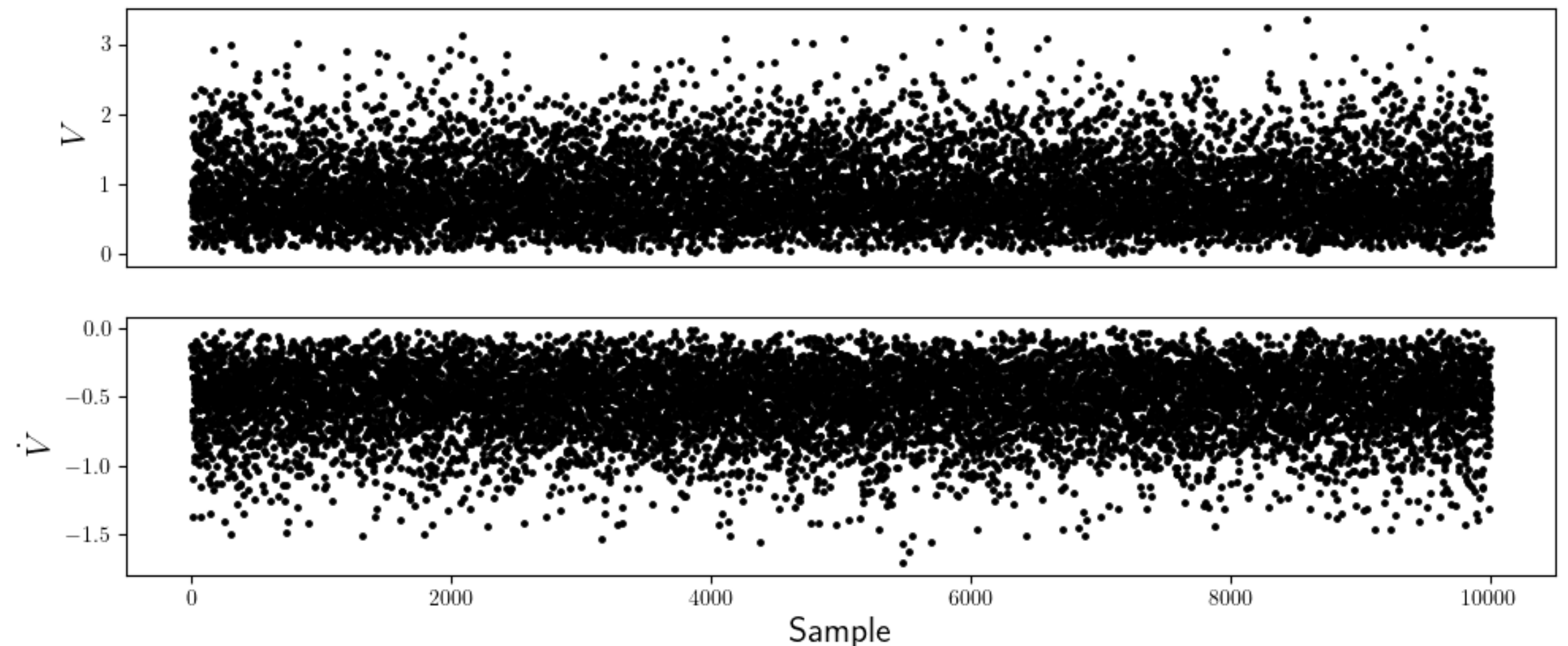


Numerical Results: RL-based Stability Certification

- The model for V obtained via PySINDy is also used to **certify** the neural network control directly, by finding a **Lyapunov function** of the closed loop system.
- The V and $\dot{V}$ are computed by exploiting the chain rule of differentiation (using u given by the RL controller directly):

$$\dot{V}_{lyap}(\mathbf{x}) = \frac{dV_{lyap}}{dt} = \left(\frac{\partial V_{lyap}}{\partial \mathbf{x}} \right)^T \frac{d\mathbf{x}}{dt} = \left(\frac{\partial V_{lyap}}{\partial \mathbf{x}} \right)^T (\mathbf{f}(\mathbf{x}) + \mathbf{g}(\mathbf{x})\mathbf{u})$$

- V and $\dot{V}$ are strictly positive and negative respectively over sampled points in the domain.
- **The conditions for Lyapunov stability are satisfied**, which proves that V is a Lyapunov function for the closed loop system, **so the controller is asymptotically stable.**



Recent Papers UA-SSEL on GNC/ML



- Scorsoglio, A., D'Ambrosio, and **Furfaro, R.**, 2026. Stability Explainable Stability Certification of Reinforcement Learning Control Policies Using Sparse Dynamics Identification, *Journal of Guidance, Control and Dynamics* (In-Press)
- Federici, L., Scorsoglio, A. and **Furfaro, R.**, 2026. Optical Bench Validation of Meta-Reinforcement Learning for Image-Based Spacecraft Guidance and Navigation. *Expert Systems*, 43(7), p.e70313.
- Federici, L., Benedikter, B. and **Furfaro, R.**, 2026. Reinforcement-learning-enhanced model predictive control with application to autonomous planetary landing. *Journal of Guidance, Control, and Dynamics*, 49(3), pp.788-805.
- Scorsoglio, A., Gaudet, B., Ghilardi, L. and **Furfaro, R.**, 2025. Meta-reinforcement learning guidance, navigation, and control for autonomous lunar landing with safe site selection. *Neural Computing and Applications*, pp.1-30.
- Federici, L. and **Furfaro, R.**, 2025. Improving reinforcement learning performance in spacecraft guidance and control through meta-learning: a comparison on planetary landing. *Neural Computing and Applications*, 37(22), pp.17249-17271.
- Scorsoglio, A., D'Ambrosio, A., Le Corre, L., Gray, B., Reddy, V. and **Furfaro, R.**, 2025. Physics-informed orbit determination for X-GEO space situational awareness. *Acta Astronautica*.
- D'Ambrosio, A. and **Furfaro, R.**, 2024. Learning fuel-optimal trajectories for space applications via Pontryagin neural networks. *Aerospace*, 11(3), p.228.
- Federici, L., Scorsoglio, A., Zavoli, A. and **Furfaro, R.**, 2023. Autonomous Guidance Between Quasiperiodic Orbits in Cislunar Space via Deep Reinforcement Learning. *Journal of Spacecraft and Rockets*, 60(6), pp.1954-1965.
- Drozd, K., **Furfaro, R.** and Mortari, D., 2024. Rapidly Exploring Random Trees with Physics-Informed Neural Networks for Constrained Energy-Optimal Rendezvous Problems. *The Journal of the Astronautical Sciences*, 71(1), p.9.





Recent Papers UA-SSEL on GNC/ML

- De Florio, M., Schiassi, E., Calabrò, F. and **Furfaro, R.**, 2024. Physics-Informed Neural Networks for 2nd order ODEs with sharp gradients. *Journal of Computational and Applied Mathematics*, 436, p.115396.
- Federici, L. and **Furfaro, R.**, 2024. Meta-Reinforcement Learning with Transformer Networks for Space Guidance Applications. In *AIAA SCITECH 2024 Forum* (p. 2061).
- Federici, L., Scorsoglio, A., Ghilardi, L., D'Ambrosio, A., Benedikter, B., Zavoli, A. and **Furfaro, R.**, 2022. Image-Based Meta-Reinforcement Learning for Autonomous Guidance of an Asteroid Impactor. *Journal of Guidance, Control, and Dynamics*, 45(11), pp.2013-2028.
- Scorsoglio, A., D'Ambrosio, A., Ghilardi, L., Gaudet, B., Curti, F. and **Furfaro, R.**, 2022. Image-Based Deep Reinforcement Meta-Learning for Autonomous Lunar Landing. *Journal of Spacecraft and Rockets*, 59(1), pp.153-165.
- Gaudet, B. and **Furfaro, R.**, 2023. Terminal Adaptive Guidance for Autonomous Hypersonic Strike Weapons via Reinforcement Metalearning. *Journal of Spacecraft and Rockets*, pp.1-13.
- Scorsoglio, A., **Furfaro, R.**, Linares, R. and Massari, M., 2023. Relative motion guidance for near-rectilinear lunar orbits with path constraints via actor-critic reinforcement learning. *Advances in Space Research*, Volume 71, Issue 1, Pages 316-335
- Federici, L., Scorsoglio, A., Zavoli, A. and **Furfaro, R.**, 2022. Meta-reinforcement learning for adaptive spacecraft guidance during finite-thrust rendezvous missions. *Acta Astronautica*, 201, pp.129-141.
- D'Ambrosio, A., Schiassi, E., Johnston, H., Curti, F., Mortari, D. and **Furfaro, R.**, 2022. Time-Energy Optimal Landing on Planetary Bodies via Theory of Functional Connections. *Advances in Space Research* , Volume 69, Issue 12, Pages 4198-4220
- Schiassi, E., D'Ambrosio, A. and **Furfaro, R.**, 2023. Bellman Neural Networks for the Class of Optimal Control Problems With Integral Quadratic Cost. *IEEE Transactions on Artificial Intelligence*.
- Schiassi, E., D'Ambrosio, A., Drozd, K., Curti, F. and **Furfaro, R.**, 2022. Physics-Informed Neural Networks for Optimal Planar Orbit Transfers. *Journal of Spacecraft and Rockets*, pp.1-16.



Recent Papers UA-SSEL on GNC/ML

- Gaudet, B., **Furfaro, R.**, Linares, R. and Scorsoglio, A., 2021. Reinforcement Metalearning for Interception of Maneuvering Exoatmospheric Targets with Parasitic Attitude Loop. *Journal of Spacecraft and Rockets*, 58(2), pp.386-399.
- Gaudet, B., Drozd, K. and **Furfaro, R.**, 2022. Adaptive Approach Phase Guidance for a Hypersonic Glider via Reinforcement Meta Learning. In *AIAA SCITECH 2022 Forum*(p. 2214).
- **Furfaro, R.**, D'Ambrosio, A., Schiassi, E. and Scorsoglio, A., 2022. Physics-Informed Neural Networks for Closed-Loop Guidance and Control in Aerospace Systems. In *AIAA SCITECH 2022 Forum* (p. 0361).
- Scorsoglio, A., D'Ambrosio, A., Ghilardi, L., Gaudet, B., Curti, F. and **Furfaro, R.**, 2022. Image-Based Deep Reinforcement Meta-Learning for Autonomous Lunar Landing. *Journal of Spacecraft and Rockets*, 59(1), pp.153-165.
- D'ambrosio, A., Schiassi, E., Curti, F. and **Furfaro, R.**, 2021. Pontryagin neural networks with functional interpolation for optimal intercept problems. *Mathematics*, 9(9), p.996.
- **Furfaro, R.** and Mortari, D., (2020). Least-squares solution of a class of optimal space guidance problems via theory of connections. *Acta Astronautica*. Volume 168, Pages 92-103.
- Gaudet, B., Linares, R. and **Furfaro, R.**, (2020a). Deep Reinforcement Learning for Six Degree-of-Freedom Planetary Landing. *Advances in Space Research*, 65(7), 1723-1741.
- Gaudet, B., Linares, R. and **Furfaro, R.**, (2020b). Adaptive Guidance and Integrated Navigation with Reinforcement Meta-Learning. *Acta Astronautica*. Volume 169, Pages 180-190.
- Gaudet, B., Linares, R. and **Furfaro, R.**, (2020c). Terminal Adaptive Guidance via Reinforcement Meta-Learning: Applications to Autonomous Asteroid Close-Proximity Operations. *Acta Astronautica*. Volume 171, June 2020, Pages 1-13.



Recent Papers UA-SSEL on GNC/ML

- **Furfaro R.**, Scorsoglio, A., Linares, R., Massari, M., (2020). Adaptive Generalized ZEM-ZEV Feedback Guidance for Planetary Landing via a Deep Reinforcement Learning Approach. *Acta Astronautica*, Volume 171, June 2020, Pages 156-171.
- Gaudet, B., **Furfaro, R.** and Linares, R., (2020). Reinforcement learning for angle-only intercept guidance of maneuvering targets. *Aerospace Science and Technology*, Volume 99, p.105746.
- Gaudet, B., Linares, R., & **Furfaro, R.** (2020). Six Degree-of-Freedom Hovering over an Asteroid with Unknown Environmental Dynamics via Reinforcement Learning. In *AIAA Scitech 2020 Forum* (p. 0953).
- Holt, H., Armellin, R., Scorsoglio, A., & **Furfaro, R.** (2020). Low-Thrust Trajectory Design Using Closed-Loop Feedback-Driven Control Laws and State-Dependent Parameters. In *AIAA Scitech 2020 Forum* (p. 1694).
- **Furfaro, R.**, Bloise, I., Orlandelli, M., Di Lizia, P., Topputo, F., & Linares, R. (2018). Deep learning for autonomous lunar landing. In *2018 AAS/AIAA Astrodynamics Specialist Conference* (pp. 1-22).
- **Furfaro, R.**, Bloise, I., Orlandelli, M., Di Lizia, P., Topputo, F., & Linares, R. (2018). A recurrent deep architecture for quasi-optimal feedback guidance in planetary landing. In *IAA SciTech Forum on Space Flight Mechanics and Space Structures and Materials* (pp. 1-24).
- **Furfaro, R.**, Wibben, D.R., Gaudet, B. and Simo, J., 2015. Terminal multiple surface sliding guidance for planetary landing: development, tuning and optimization via reinforcement learning. *The Journal of the Astronautical Sciences*, 62(1), pp.73-99.
- Gaudet, B. and **Furfaro, R.**, 2014. Adaptive pinpoint and fuel efficient mars landing using reinforcement learning. *IEEE/CAA Journal of Automatica Sinica*, 1(4), pp.397-411.

Questions?



<https://ssel.arizona.edu/>



Space Systems
Engineering Laboratory