SAND2020-7953PE

Hypersonic Guidance and Control via Deep Reinforcement Learning Methods

Roberto Furfaro

Professor and Director of SSA-Arizona Initiative
Department Of Systems and Industrial Engineering
Department of Aerospace and Mechanical Engineering
Defense and Security Research Institute
University of Arizona

Autonomy for Hypersonic, Virtual Field Day, Aug 4-6, 2020

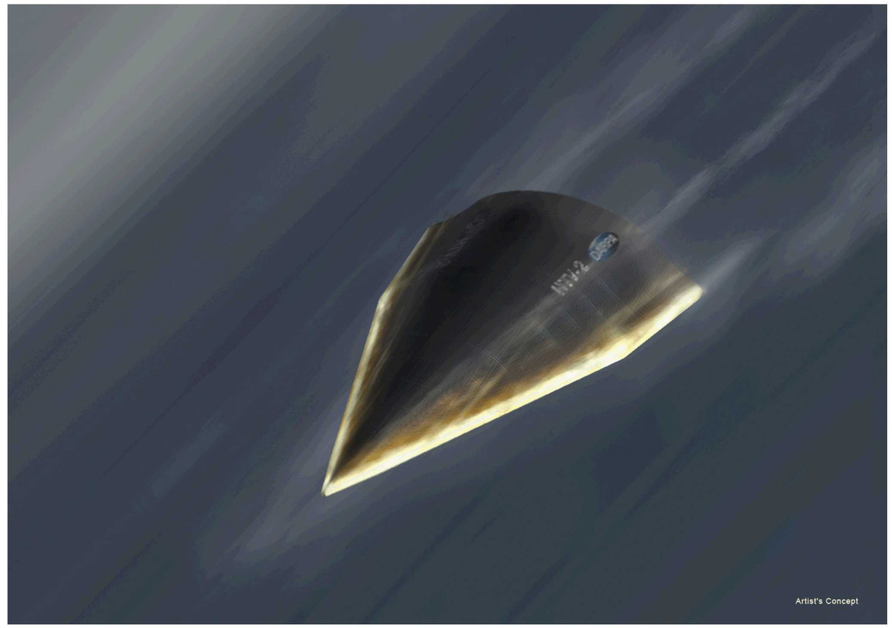
Sandia National Laboratories is a multimission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525.

Outline of the Talk

- Hypersonic Re-Entry Problem via fast ELM-based Advantage Actor-Critic (A2C) Reinforcement Learning
  - Problem formulation
  - Algorithm description
  - Training and Validation
  - Results
- Path forward
  - Deep Meta-Reinforcement Learning: Current Effort

Reinforcement Learning for Hypersonic Flight: ELM-based A2C

- The Advantage Actor Critic Algorithm (A2C) is utilized for generating a policy, π_{θ_μ}, that drive an unpowered re-entry vehicle towards a target while avoiding a maximum on heat rate and constrained control.
- Our A2C algorithm makes use of Extreme Learning Machines (ELMs) to train a critic neural network that approximates the value function, V.
- Use of this ELM rather than deeper neural networks is beneficial because it greatly speeds up the run-time of the algorithm while approximating V well.



http://www.shackletonenergy.com/reentry

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.

Problem Setup: Equations of Motion

- State:
  - Radius
  - Longitude
  - Latitude
  - Velocity magnitude
  - Flight path angle
  - Heading angle
- Control:
  - Bank angle

$$\frac{\mathrm{d}r}{\mathrm{d}t} = V\sin(\gamma)$$

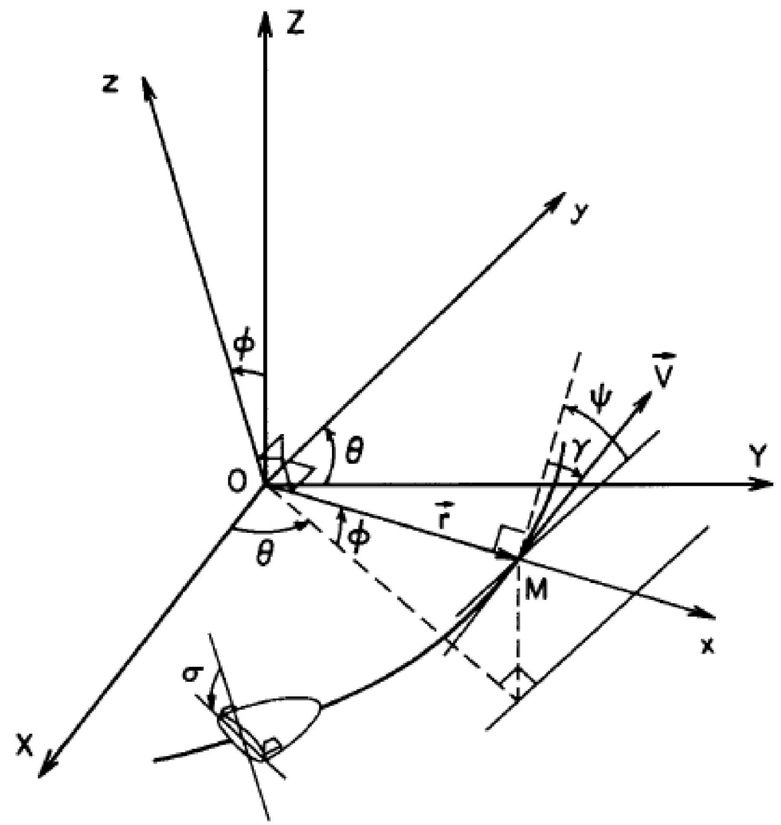
$$\frac{\mathrm{d}\theta}{\mathrm{d}t} = \frac{V\cos(\gamma)\cos(\psi)}{r\cos(\phi)}$$

$$\frac{\mathrm{d}\phi}{\mathrm{d}t} = \frac{V\cos(\gamma)\sin(\psi)}{r}$$

$$\frac{\mathrm{d}V}{\mathrm{d}t} = -\frac{D}{m} - g\sin(\gamma)$$

$$V\frac{\mathrm{d}\gamma}{\mathrm{d}t} = \frac{L}{m}\cos(\sigma) - g\cos(\gamma) + \frac{V^2}{r}\cos(\gamma)$$

$$V\frac{\mathrm{d}\psi}{\mathrm{d}t} = \frac{L\sin(\sigma)}{m\cos(\gamma)} - \frac{V^2}{r}\cos(\gamma)\cos(\psi)\tan(\phi)$$



Problem Setup: Aerodynamics

- Angle of attack is specified as a function of velocity.
- C_L and C_D are approximated by fitting the data in the hypersonic regime.

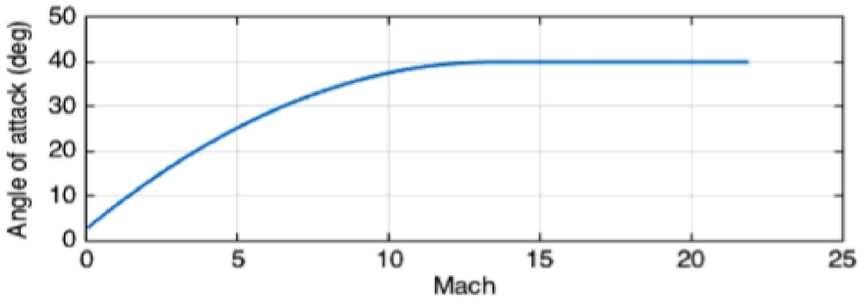
$$D = \frac{\rho V^2 A_{\text{ref}} C_D}{2}$$

$$L = \frac{\rho V^2 A_{\text{ref}} C_L}{2}$$

$$C_L = -0.041065 + 0.016292\alpha + 0.0002602\alpha^2$$

$$C_D = 0.080505 - 0.03026 C_L + 0.86495 C_L^2$$

$$\alpha = \begin{cases} 40, & \text{if } V > 4570 \text{ m/s} \\ 40 - 0.20705\left(V - 4570\right)^2 / 340^2, & \text{otherwise} \end{cases}$$



Problem Setup: Boundary and Path Constraints

- An initial and final state to be reached are specified.
- Bank angle has a minimum and maximum
- The heat rate of the rlv vehicle must stay below a certain value.

$$\dot{Q} = k_Q \sqrt{\rho} V^{3.15} \leq \dot{Q}_{\max}$$

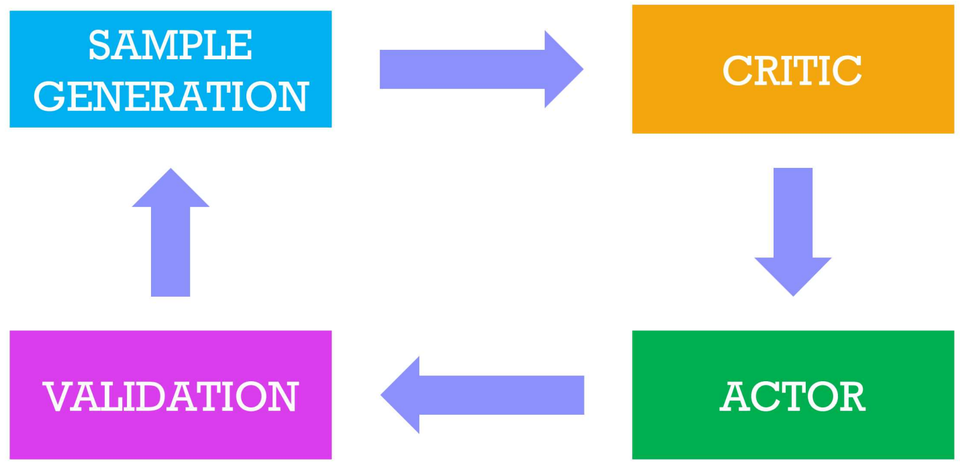
$$\sigma \in [\sigma_{\min}, \sigma_{\max}]$$

$$\boldsymbol{x}(t_0) = \boldsymbol{x}_0$$

$$\boldsymbol{x}(t_f) = \boldsymbol{x}_f$$

ELM-based A2C Algorithm: Basics

- The algorithm is composed of 4 major blocks:
  1. Sample trajectory generation with current policy
  2. Critic training for estimating V
  3. Updating the policy with V estimate
  4. Monte Carlo (MC) evaluation



1. Trajectories are generated based on a stochastic π_{θ_u} that maps states to controls.
2. Makes use of an ELM to approximate the discounted reward-to-go of states along a trajectory.
3. Uses the approximated discounted reward-to-go of the states for all sampled trajectories to perform a policy update.
4. Performs Monte Carlo analysis on the deterministic π_{θ_u} after the update to determine whether to continue iterating.

A2C Algorithm Specifics: Sample Generation

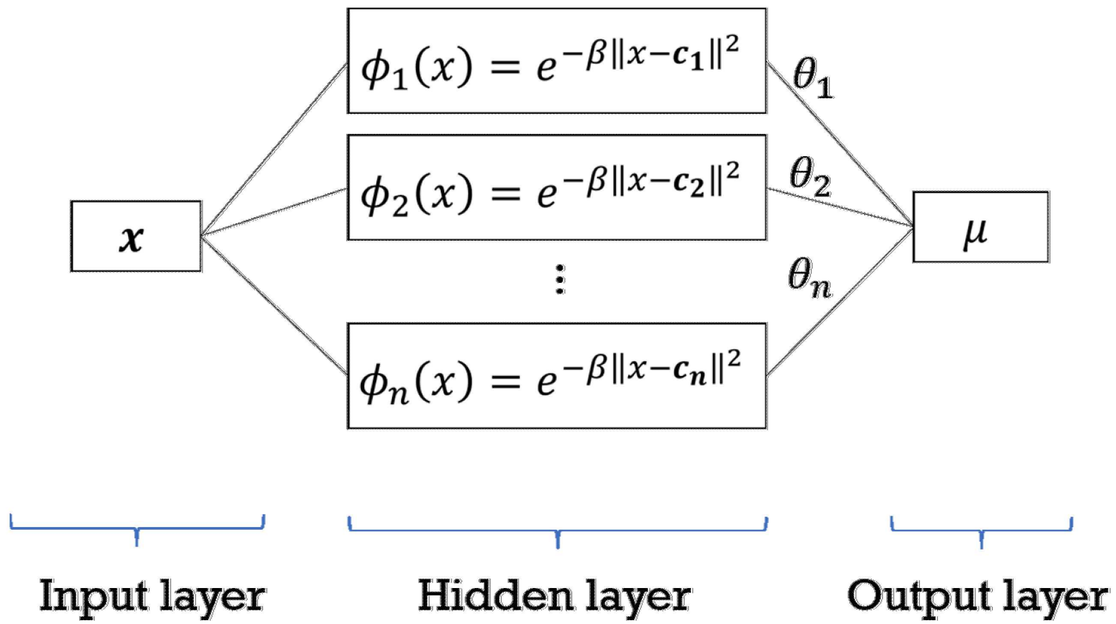
SAMPLE GENERATION

- Trajectories are generated by having the agent (vehicle) interact with its environment (EoMs) by instantiating some and the perturbing initial state.
- Sample trajectories consist of states, bank angles, and rewards at each timestep $(\boldsymbol{x}_{i,t}, u_{i,t}, r_{i,t})$.
- The policy is described as a gaussian distribution with fixed std:

$$u = \pi_{\theta_u} = \mathcal{N}(\mu_u, \sigma^2)$$

- The mean of the gaussian represents the output of a neutral net:

$$\mu_u - \phi(\boldsymbol{x})^{\mathrm{T}}\theta_u$$

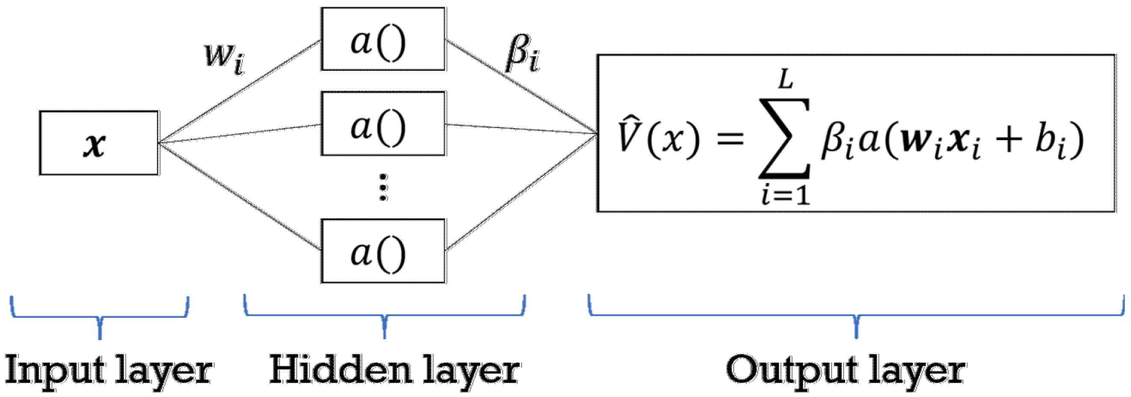


A2C Algorithm Specifics: Critic

CRITIC

- $\nabla_\theta J(\pi_\theta) = \mathbb{E}[\nabla \log \pi_\theta(u|\boldsymbol{x}) Q^\pi(\boldsymbol{x}, u)]$
- $\mathbb{E}[Q^\pi(\boldsymbol{x}, u)]$ is very hard to compute, so we approximate $Q^\pi(\boldsymbol{x}, u)$ which means we also approximate $\nabla_\theta J(\pi_\theta)$
- The variance of our approximate estimate is greatly reduced by subtracting the value function from $\hat{A}^\pi(u, \boldsymbol{x}) = \hat{Q}^\pi(\boldsymbol{x}, u) - \hat{V}^\pi(\boldsymbol{x})$
- Thus, we approximate the advantage function:
- $Q^\pi(\boldsymbol{x}, u)$ is a function of the r and V.
- Thus, we just need $\hat{V}^\pi(\boldsymbol{x})$ which found via an ELM using the following training set:

$$\left\{ \boldsymbol{x}_{i,t}, \sum_{t'-t}^{T} \left(\gamma_1^{t'-t} r_1(\boldsymbol{x}_{i,t'}, u_{i,t'}) + \gamma_2^{t'-t} r_2(\boldsymbol{x}_{i,t'}, u_{i,t'}) \right) \right\}$$



A2C Algorithm Specifics: Actor

ACTOR

- Compute the gradient of the performance objective and update the policy parameters via stochastic gradient ascent:
  - Use the ELM-based approximate Value Function to estimate the Advantage

$$\nabla J(\pi_\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(u_{i,t}|\boldsymbol{x}_{i,t}) \hat{A}(u_{i,t}, \boldsymbol{x}_{i,t})$$

$$\nabla \log \pi_\theta = \frac{\pi_\theta - \mu}{\sigma^2} \phi(\boldsymbol{x})$$

A2C Algorithm Specifics: Validation

VALIDATION

- After the policy update we repeat the sample generation step, but set the variance of the gaussian to zero because we are no exploring the action space, hence a MC test is performed.
- The cumulative reward of each MC trajectory is found and the mean is calculated.
- If the difference in the mean of the MC trajectory's cumulative reward at the current and last epoch is below a tolerance for 5 epochs, learning is stopped.

Reward Function

- Description of terms:
  1. Penalty if maximum heat rate is exceeded
  2. Hints when maximum heat rate is being approached
  3. Shaping reward that penalizes when the heading angle is not pointed at the target
  4. Terminal penalty applied at the final time based on whether the position is close to the target

$$r(t) = w_{Q_{\max}} \max(0, Q_t - Q_{\max})$$

$$+ w_{Q_{\text{mgn}}} \max(0, Q_t - Q_{\text{mgn}})$$

$$+ w_\gamma |\gamma_t - \gamma^*|$$

$$+ w_r \delta(t - t_f) ||\boldsymbol{r}_t - \boldsymbol{r}_f||$$

ELM-based A2C Algorithm: Specifics

Algorithm 1: A2C for Hypersonic Entry

for $k = 1, \ldots,$ *max epochs* **do**
  for $i = 1, \ldots,$ *episodes per batch* **do**
    for $t = 1, \ldots,$ *time steps per episode* -1 **do**
      Sample policy $\pi \rightarrow u$;
      generate samples $(\boldsymbol{x}_{i,t}, u_{i,t}, r_{i,t})$;
  fit $\hat{V}(\boldsymbol{x})$ to sampled reward-to-go
  $\left\{\boldsymbol{x}_{i,t}, \sum_{t'-t}^{T} \left(\gamma_1^{t'-t} r_1(\boldsymbol{x}_{i,t'}, u_{i,t'}) + \gamma_2^{t'-t} r_2(\boldsymbol{x}_{i,t'}, u_{i,t'}\right)\right\}$;
  for $i = 1, \ldots,$ *episodes per batch* **do**
    for $t = 1, \ldots,$ *time steps per episode* -1 **do**
      evaluate $\hat{A}_n^{\pi}(u_{i,t}, \boldsymbol{x}_{i,t}) =$
      $\sum_{t'=t}^{T} \left(\gamma_1^{t'-t} r_1(\boldsymbol{x}_{i,t'}, u_{i,t'}) + \gamma_2^{t'-t} r_2(\boldsymbol{x}_{i,t'}, u_{i,t'})\right) - \hat{V}^{\pi}(\boldsymbol{x}_{i,t})$;
  evaluate $\nabla J(\pi_\theta) \approx \frac{1}{N} \sum_{i=1}^{N} \sum_{t=1}^{T} \nabla_\theta \log \pi_\theta(u_{i,t}|\boldsymbol{x}_{i,t}) \hat{A}(u_{i,t}, \boldsymbol{x}_{i,t})$;
  update policy $\boldsymbol{\theta}_{k+1} = \boldsymbol{\theta}_k + \alpha \nabla_\theta J(\pi_{\theta_k})$;
  test new policy $\pi_{\theta_k} \rightarrow$ perform M.C. to obtain mean cumulative reward mean$(R_k) = \frac{1}{N} \sum_{i=0}^{N} \sum_{t=0}^{T} r(\boldsymbol{x}_{i,t}, u_{i,t})$;
  calculate average change, E, in $mean(R)$ among last 5 epochs;
  if $E < \epsilon$ **then**
    break

Numerical Results: Simulation Setup

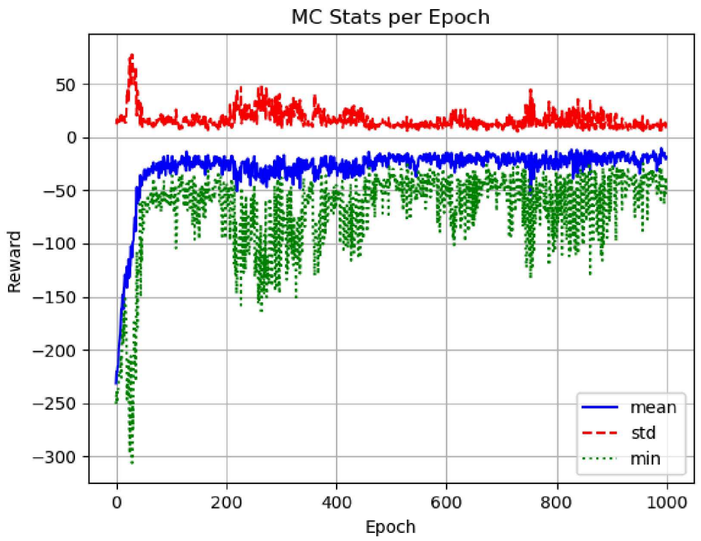
- BCs and PCs:

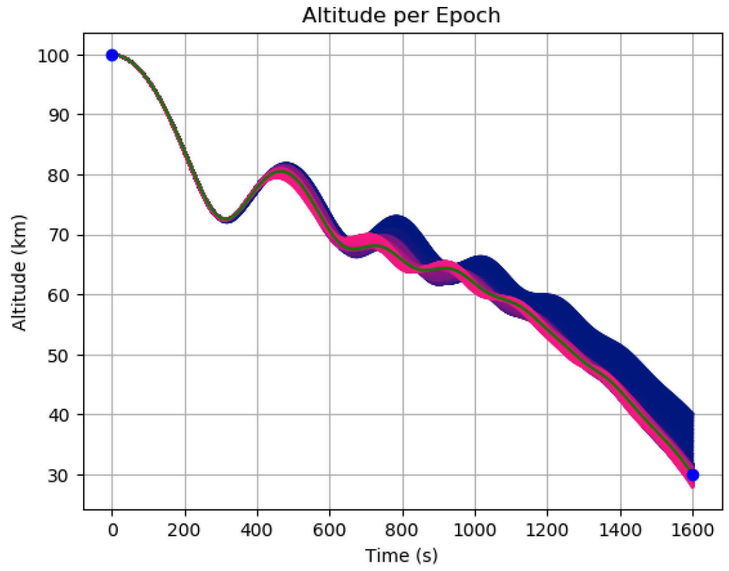
Initial state element	Value	final state element	Value	Heat rate parameter	Value
r_0 (km)	6478	r_f (km)	6408	$w_{Q_{\max}}$ (Kw/m^2)	1500
θ_0 (deg)	0	θ_f (deg)	70	$w_{Q_{\text{mgn}}}$ (Kw/m^2)	1425
ϕ_0 (deg)	0	ϕ_f (deg)	8	$\sigma_{\min}$ (deg)	-80
V_0 (km/s)	7.45	V_f (km/s	free	$\sigma_{\max}$ (deg)	80
γ_0 (deg)	0	γ_f (deg)	free		
ψ_0 (deg)	0	ψ_f (deg)	free		

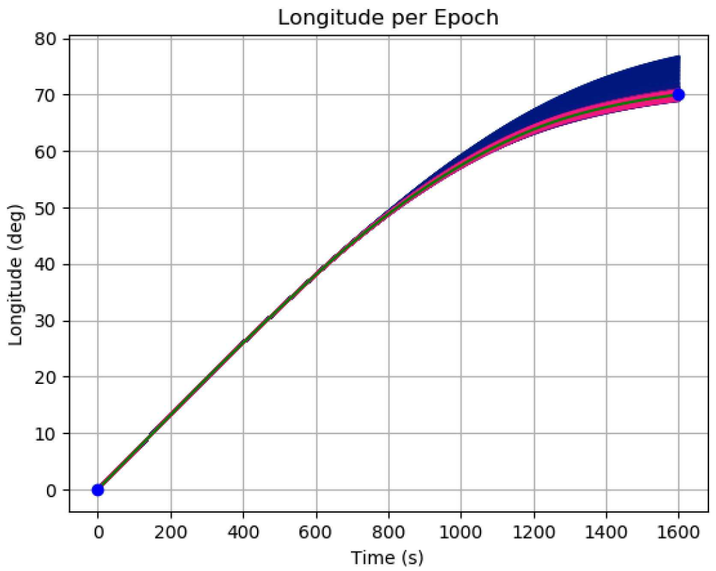
- Perturbation bounds:

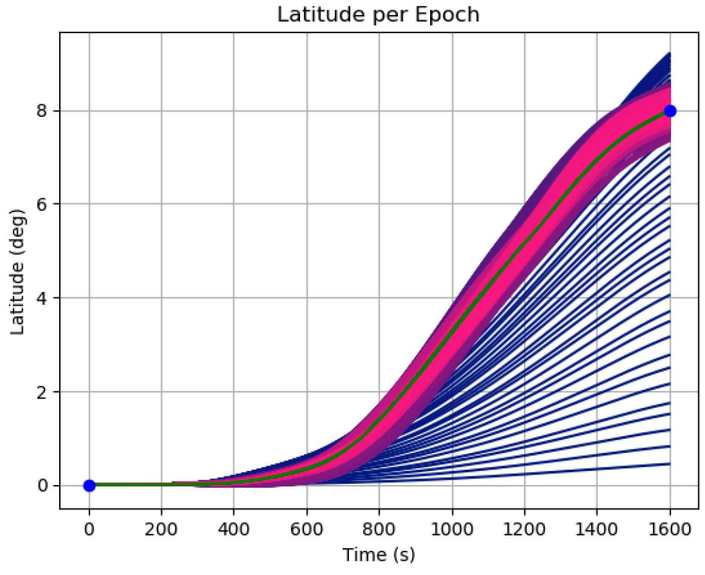
Perturbed Parameter	Value
r_0 (km)	$\pm$ 1
θ_0 (deg)	$\pm$ 0.1
ϕ_0 (deg)	$\pm$ 0.1
V_0 (km/s)	$\pm$ 0.05
γ_0 (deg)	$\pm$ 0.1
ψ_0 (deg)	$\pm$ 0.1

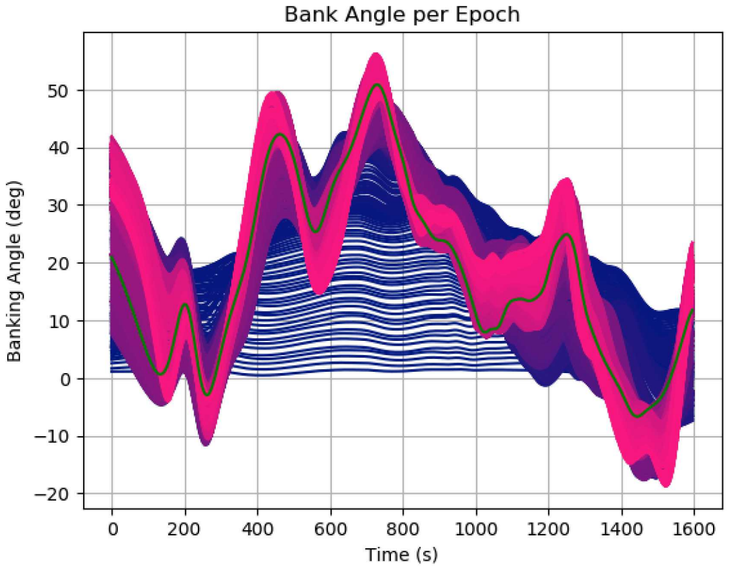
Numerical Results: Training

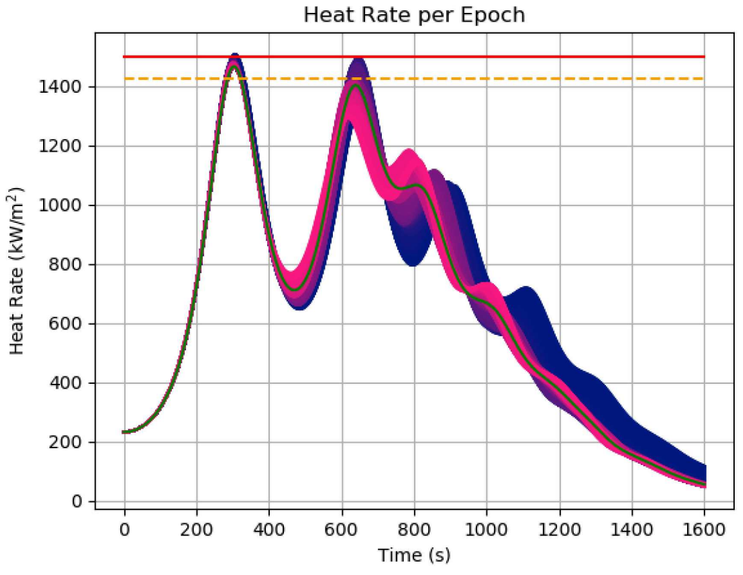










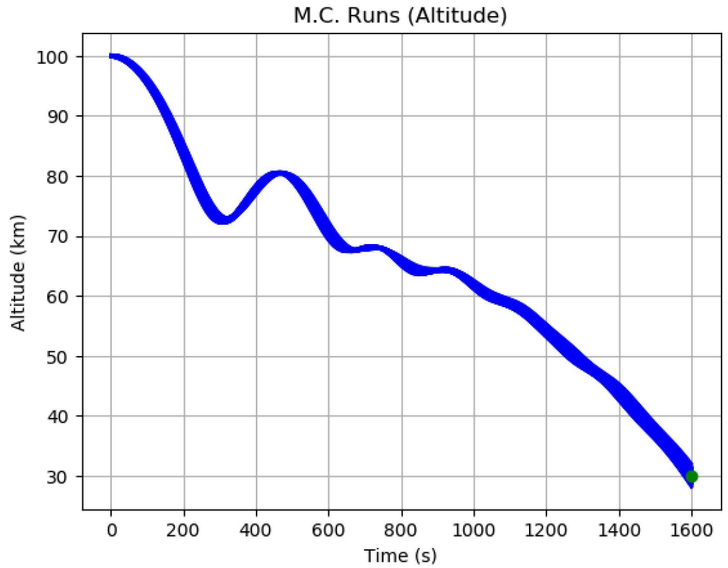


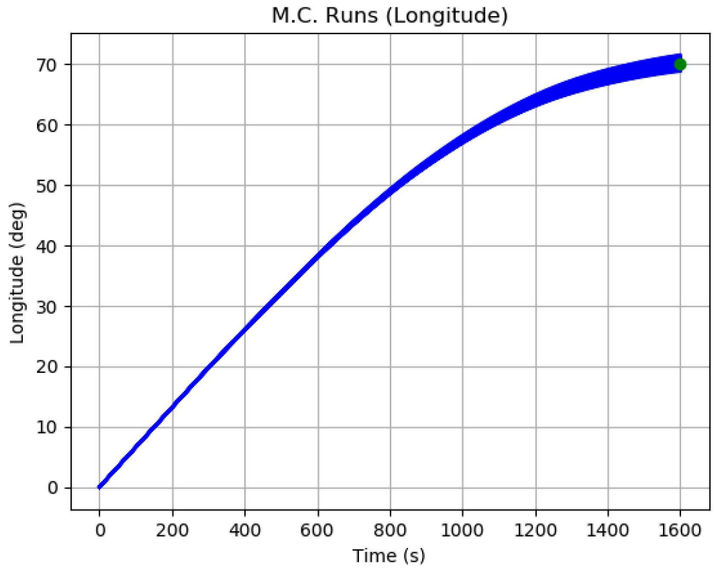
Numerical Results: ELM Performance

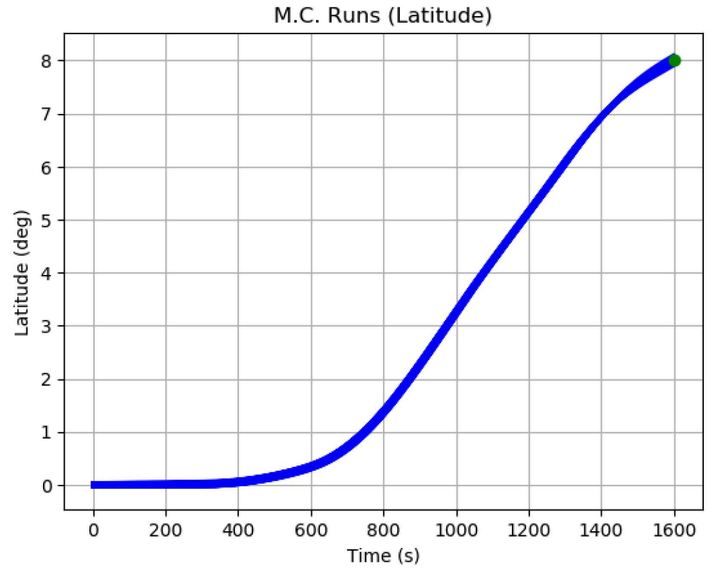
- Average critic training time is significantly less than iteration time.
- Average Critic normalized root mean square error as on an order of 10^{-2}.

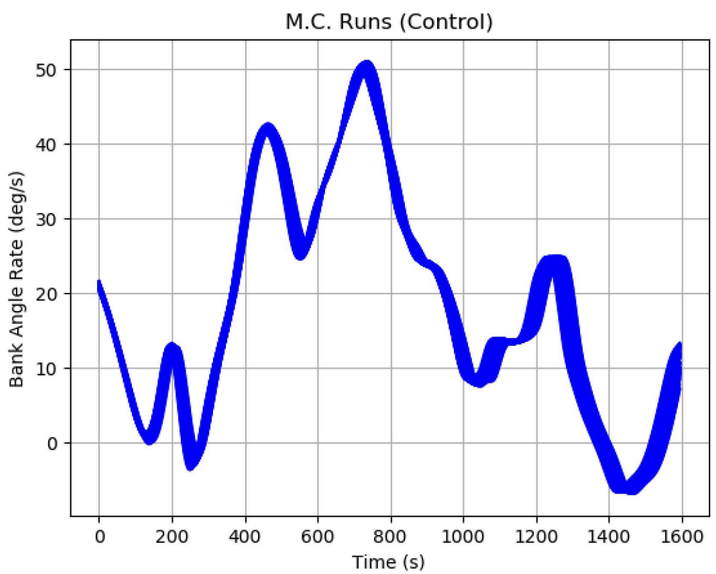
# of iterations	Total training time (hours)	Average iteration time (s)	Average critic training time (s)	Average critic NRMSE
1000	4.77	17.18	1.26	3.11×10^{-2}

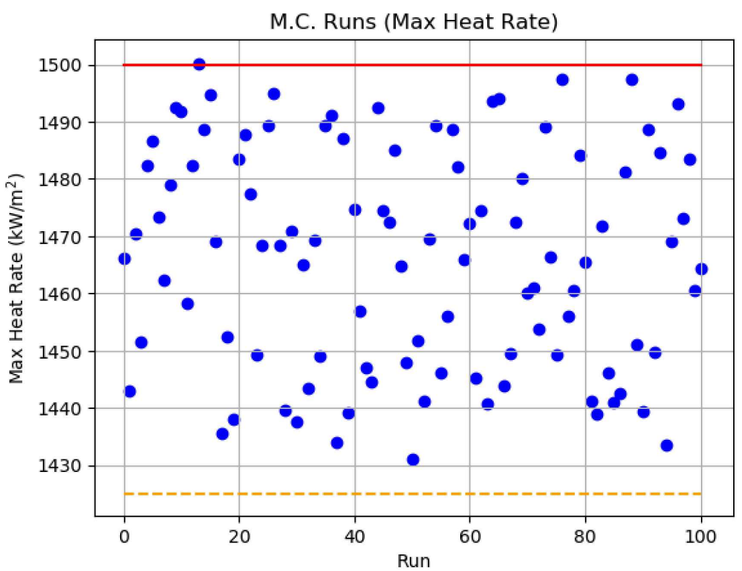
Numerical Results: Testing

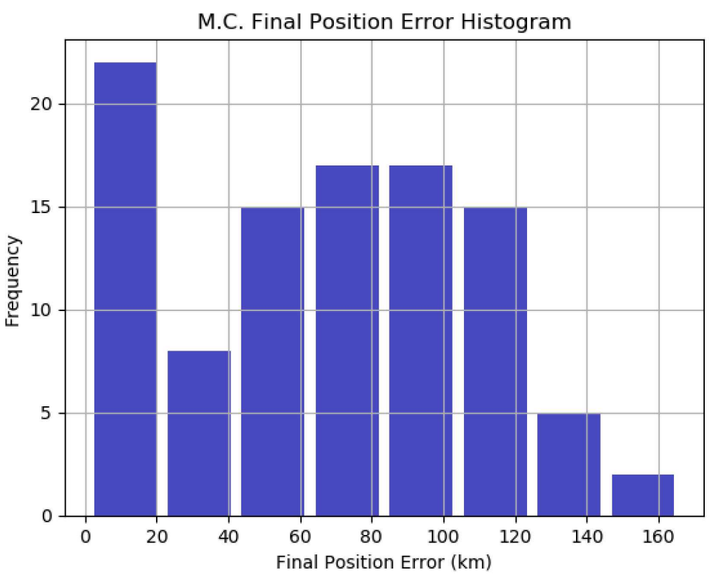










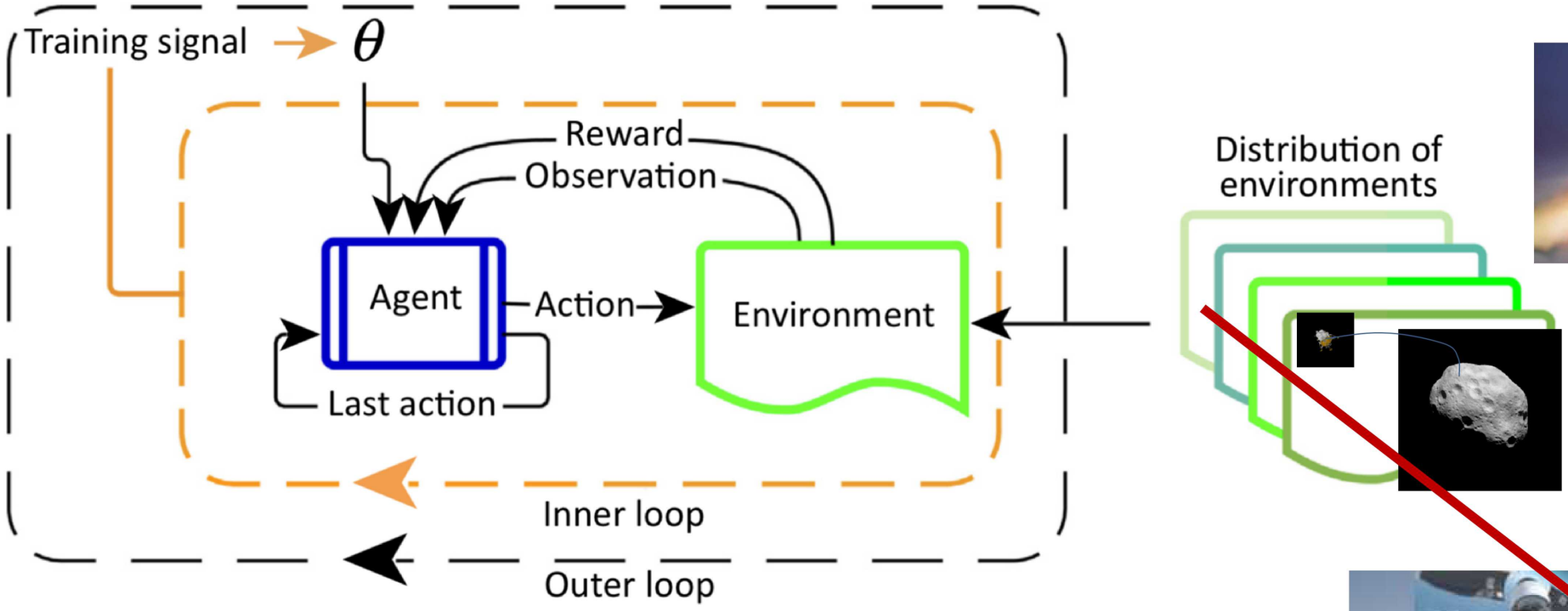


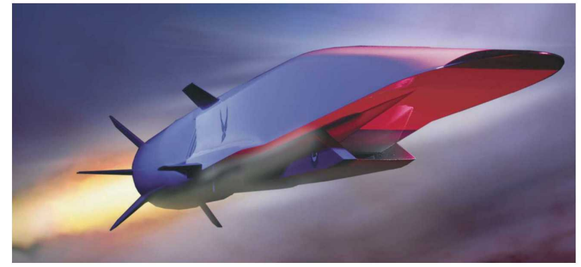
Meta Reinforcement Learning (M-RL)

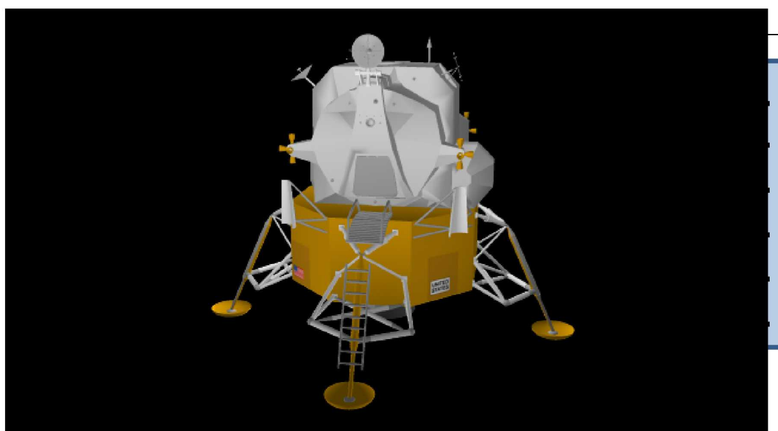
- **Meta-Learning: Learning to Learn**
  - Leverage data from previous tasks to acquire a learning procedure that can quickly adapt to new tasks
  - Assumption: draw meta-training and meta-tests from some task distribution $\rho(Task)$
- Goal: Find a learning procedure $\vartheta' = u_\psi(D_T^{tr}, \vartheta)$ that can learn a range of tasks T from a small dataset D_T^{tr}
  - Recurrent-based or gradient-based meta-learning approach

$$\min_{\boldsymbol{\vartheta},\boldsymbol{\psi}} E_{T\sim\rho(T)}[\mathcal{L}(D_T^{test}, \boldsymbol{\vartheta'})] \qquad s.t. \qquad \boldsymbol{\vartheta'} = u_{\boldsymbol{\psi}}(D_T^{tr}, \boldsymbol{\vartheta})$$

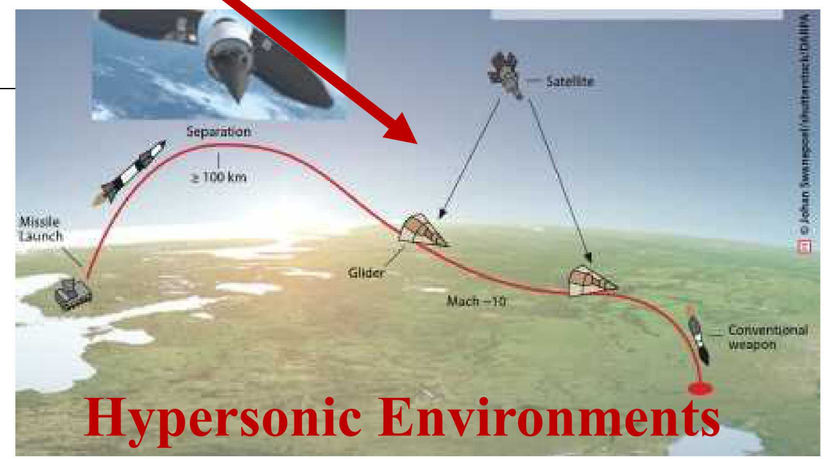
Meta Reinforcement Learning for Space Guidance & Control: Adaptation to Hypersonic G&C





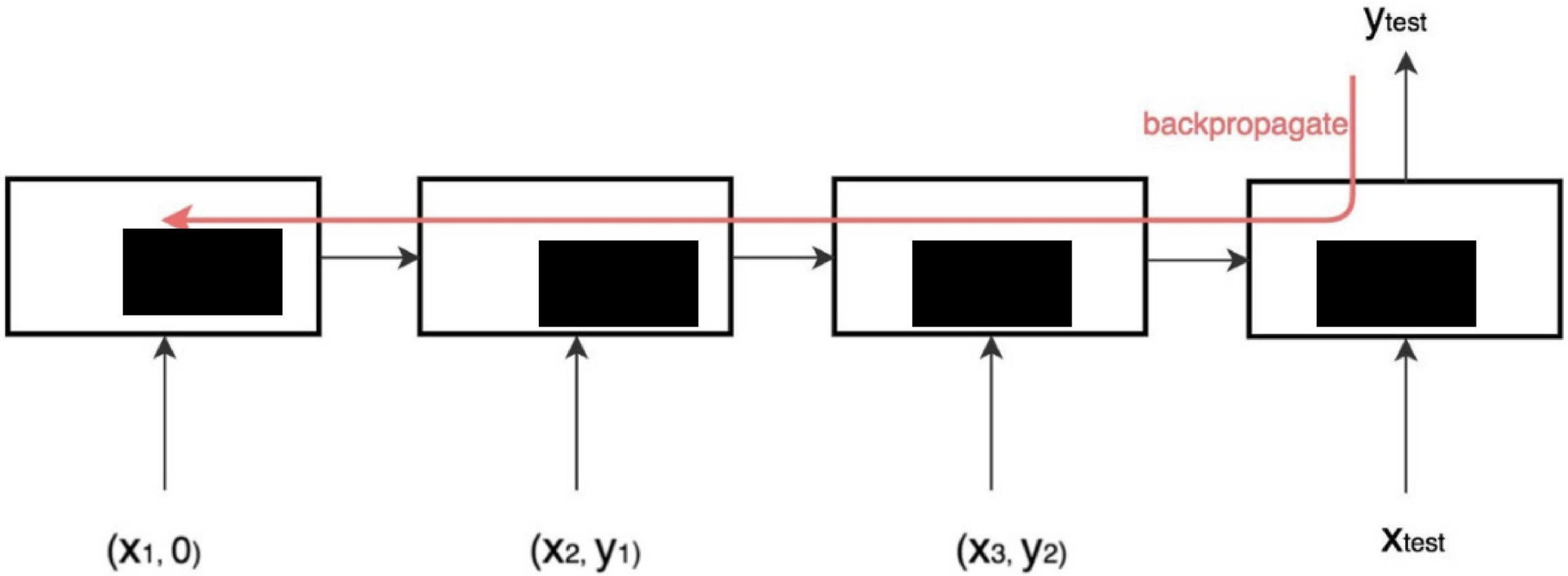


- Unknown Dynamics
- Actuator Failure
- Sensor Distortion
- Inertia Tensor Variation
- Initial Center of Mass Variation
- Initial Mass Variation



Meta Reinforcement Learning for Space Guidance: Using Recurrent Policies

Recurrent Neural Network



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

Deep Meta-Learning with Proximal Policy Optimization (PPO)

- **PPO** is a variation of the **Trust Region Policy Optimization (TRPO)**
  - TRPO use a constraint to reduce the size of the gradient step taken at each iteration
    - It uses the **Kullback-Leiber (KL) divergence** between old and new policy as constraint
  - PPO approximate TRPO process by using a **clipped objective function**

$$\begin{aligned} \underset{\boldsymbol{\theta}}{\text{minimize}} \quad & \mathbb{E}_{p(\boldsymbol{\tau})}\left[\frac{\pi_{\boldsymbol{\theta}}(\mathbf{u}_k|\mathbf{x}_k)}{\pi_{\boldsymbol{\theta}_{\text{old}}}(\mathbf{u}_k|\mathbf{x}_k)} A^{\pi}_{\mathbf{w}}(\mathbf{x}_k, \mathbf{u}_k)\right] \\ \text{subject to} \quad & \mathbb{E}_{p(\boldsymbol{\tau})}\left[\text{KL}\left(\pi_{\boldsymbol{\theta}}(\mathbf{u}_k|\mathbf{x}_k), \pi_{\boldsymbol{\theta}_{\text{old}}}(\mathbf{u}_k|\mathbf{x}_k)\right)\right] \leqslant \delta \end{aligned}$$

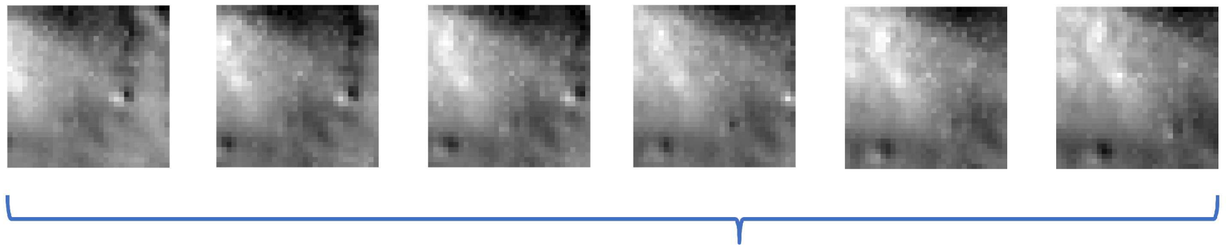
PROBABILITY RATIO

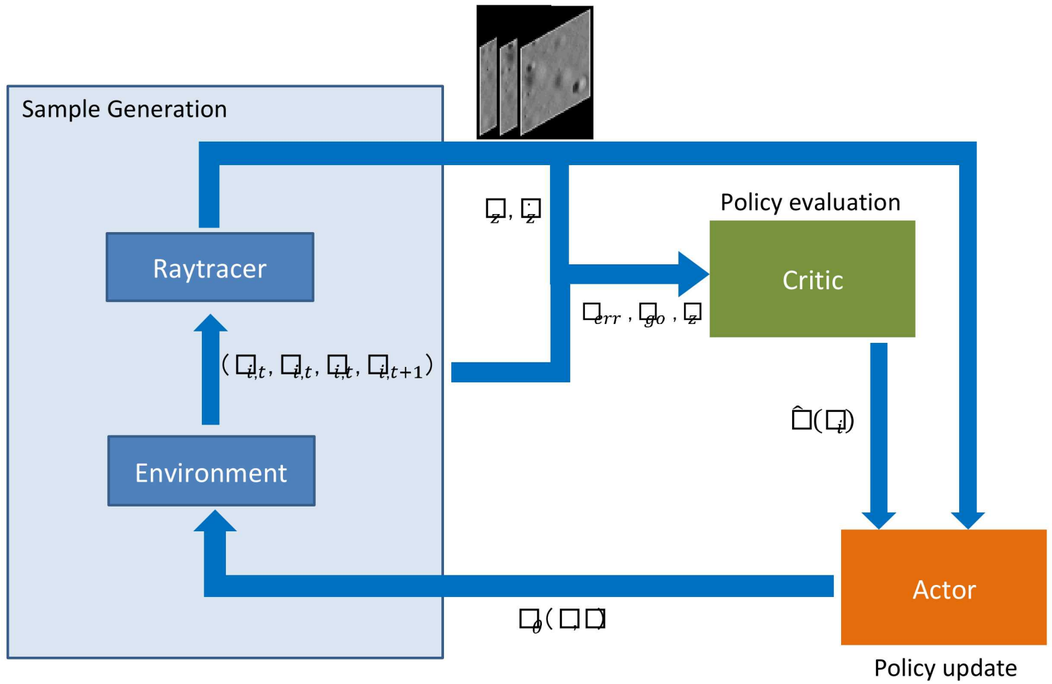
$$p_k(\boldsymbol{\theta}) = \frac{\pi_{\boldsymbol{\theta}}(\mathbf{u}_k|\mathbf{x}_k)}{\pi_{\boldsymbol{\theta}_{\text{old}}}(\mathbf{u}_k|\mathbf{x}_k)}$$

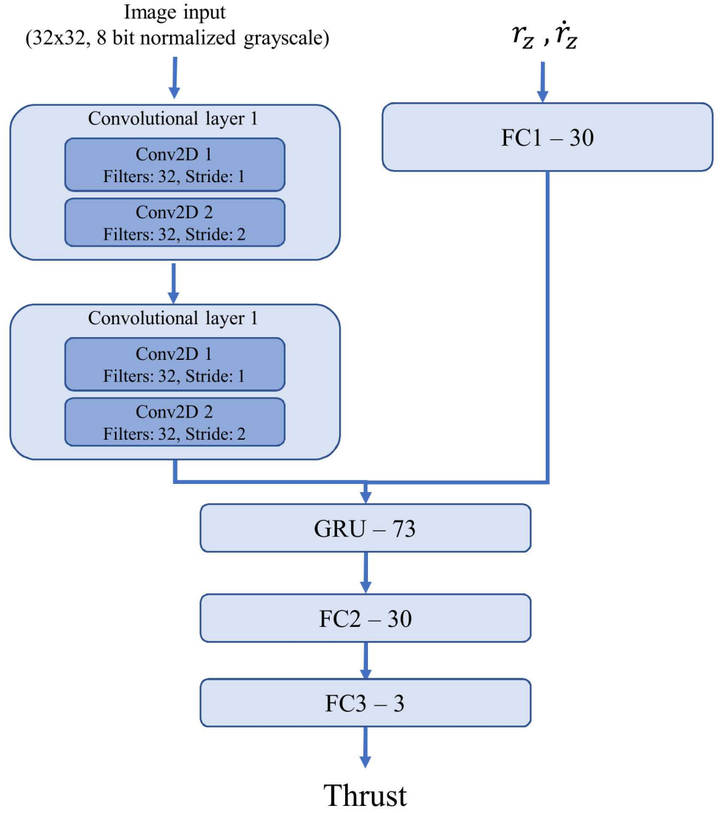
CLIPPED COST FUNCTION

$$J(\boldsymbol{\theta}) = \mathbb{E}_{p(\boldsymbol{\tau})}\left[\min\left[p_k(\boldsymbol{\theta}), \text{clip}(p_k(\boldsymbol{\theta}), 1-\epsilon, 1+\epsilon)\right] A^{\pi}_{\mathbf{w}}(\mathbf{x}_k, \mathbf{u}_k)\right]$$

Image-based Meta RL for Lunar Landing

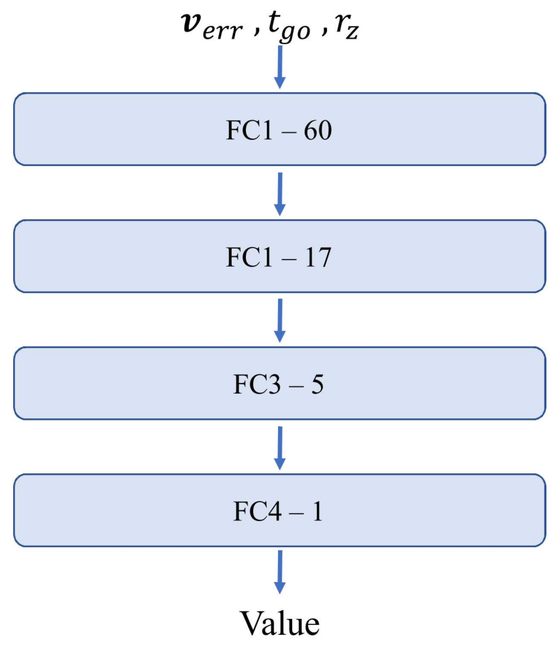






Value function observation:

$$obs_{VF} = [\boldsymbol{v_{err}} \quad t_{go} \quad r_z]$$



Results: landing with actuator failure and uncertain mass and gravity

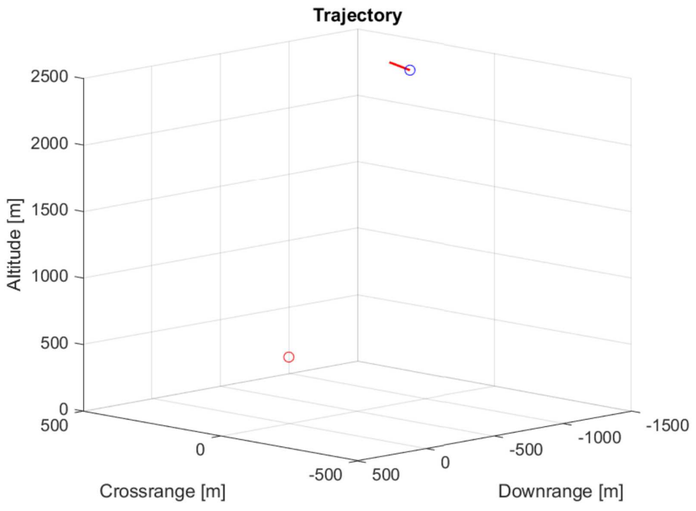
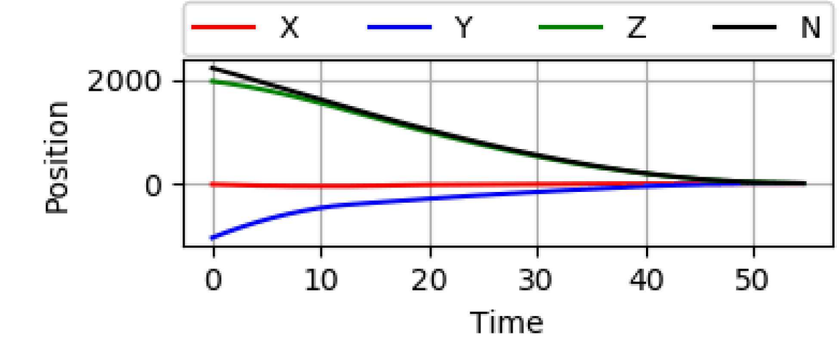


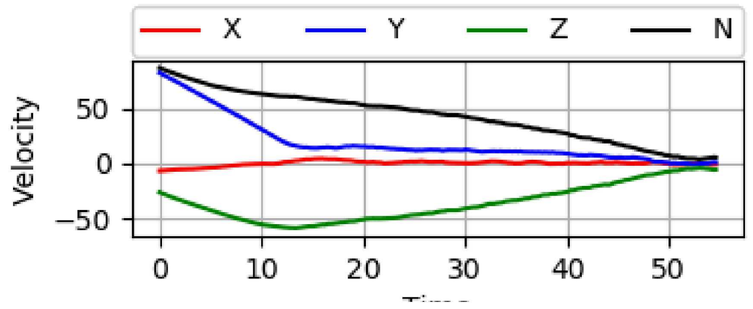
Table 2: Initial states

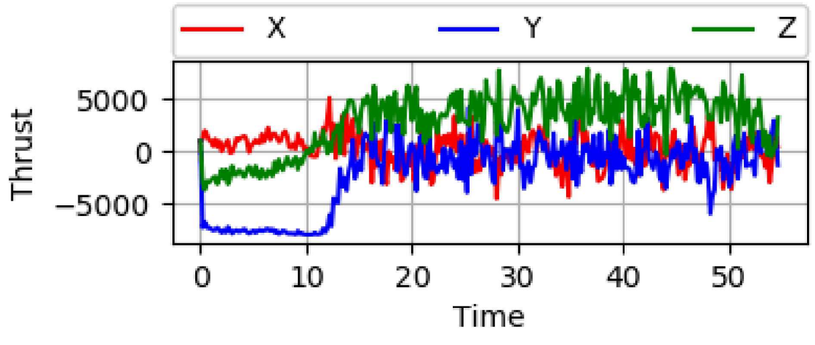
	Position		Velocity	
	Min (m)	Max (m)	Min (m/s)	Max (m/s)
Downrange	-1500	-1000	80	100
Crossrange	-200	200	-20	20
Elevation	2300	2500	-30	-20

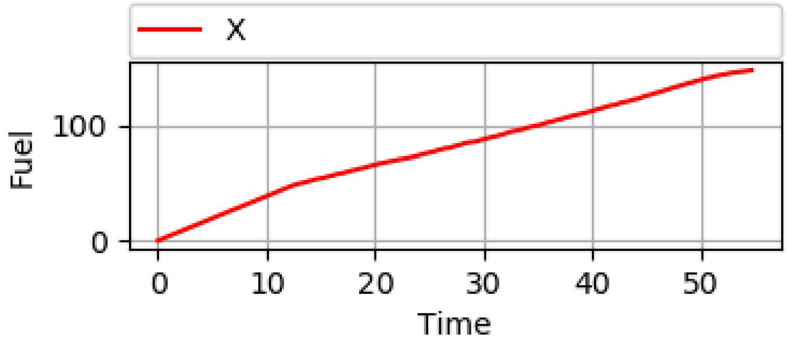
Table 3: Target states

	Position (m)	Velocity (m/s)
Downrange	0	0
Crossrange	0	0
Elevation	500	-1



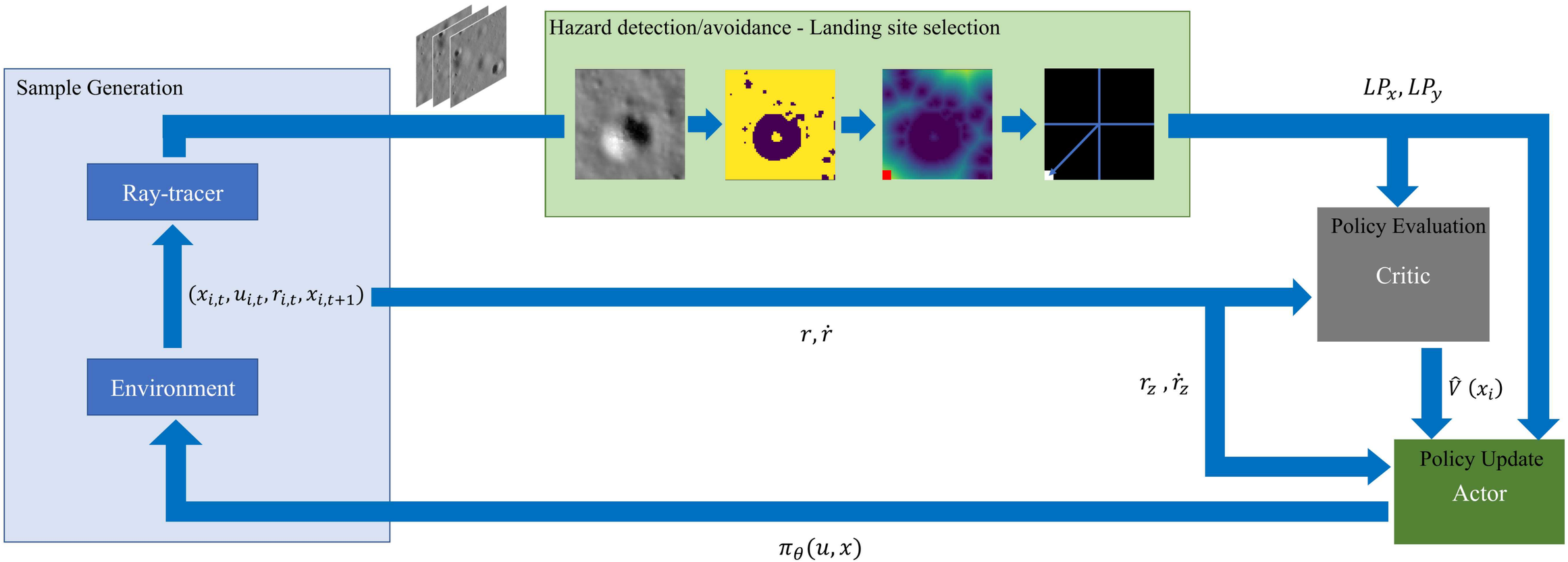






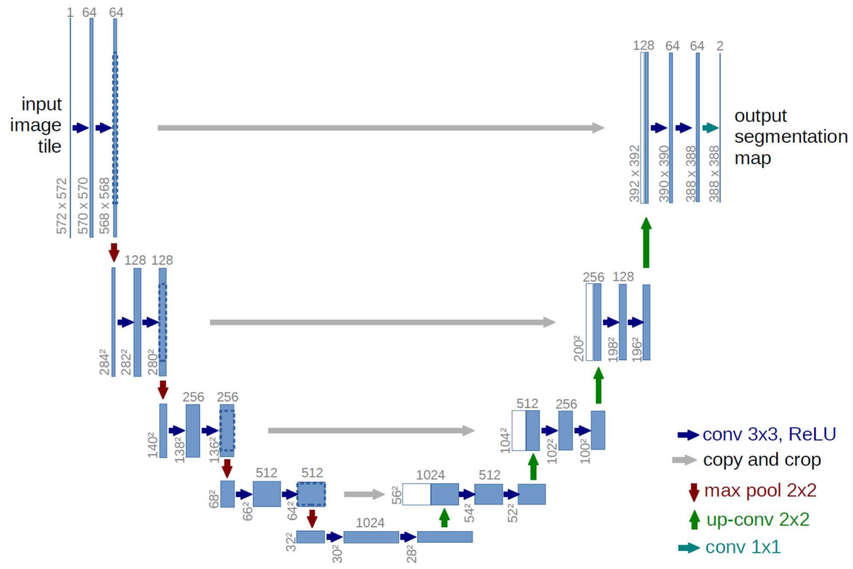
Real-time landing site selection and hazard avoidance: Architecture and Training

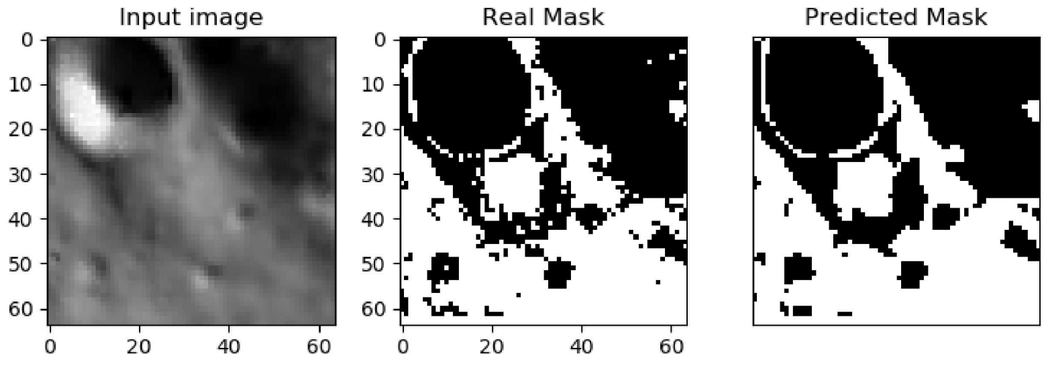
Introduce hazard detection and avoidance into the RL framework – automatic landing site selection



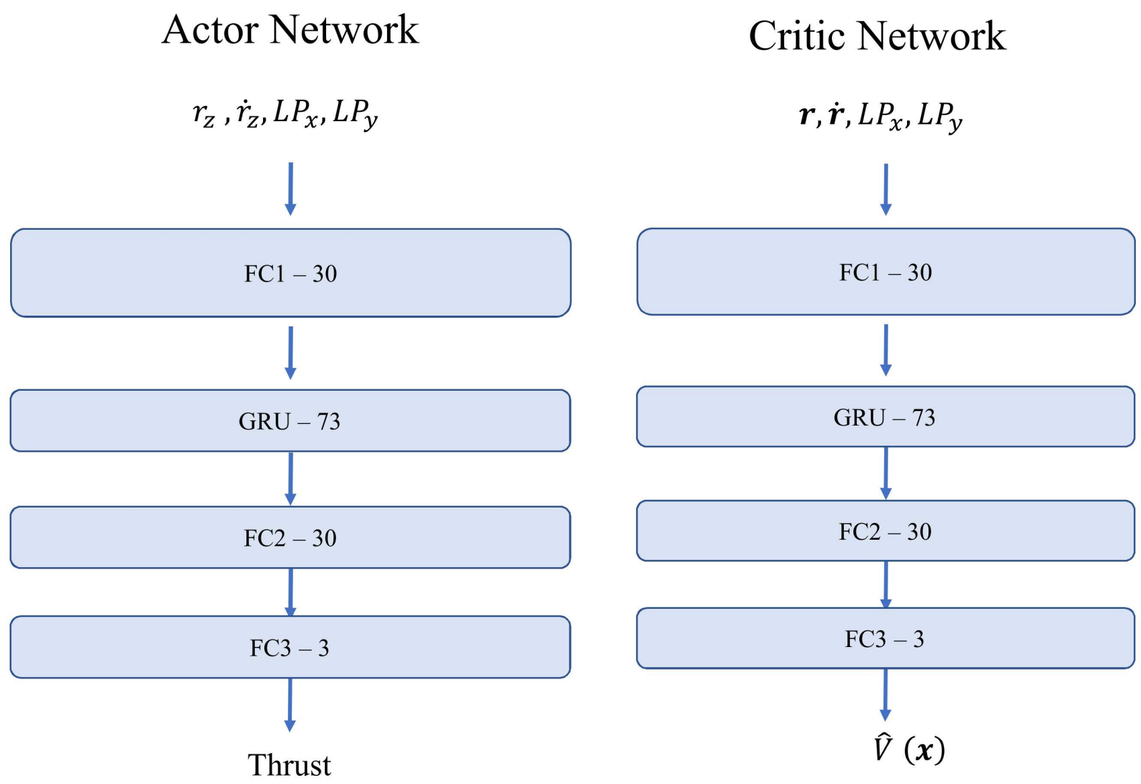
Policy optimization: networks

Hazard Detection Network

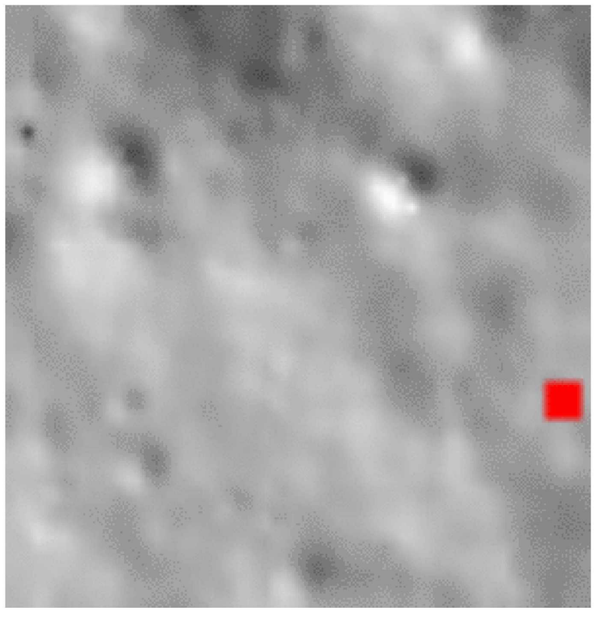




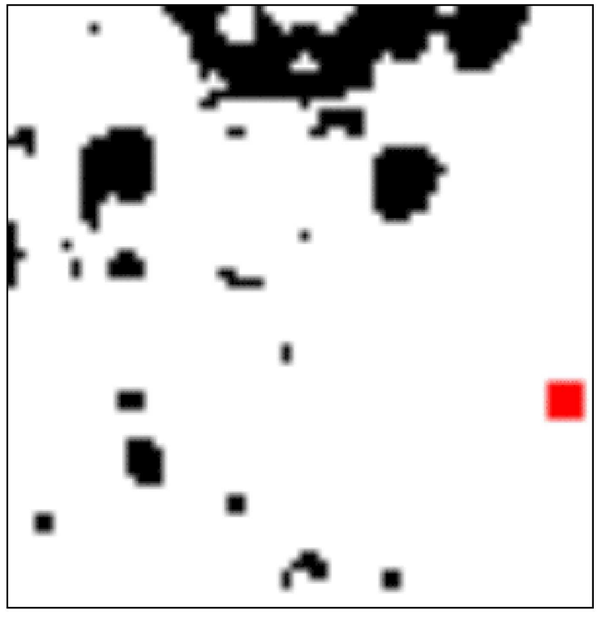
Reinforcement learning networks



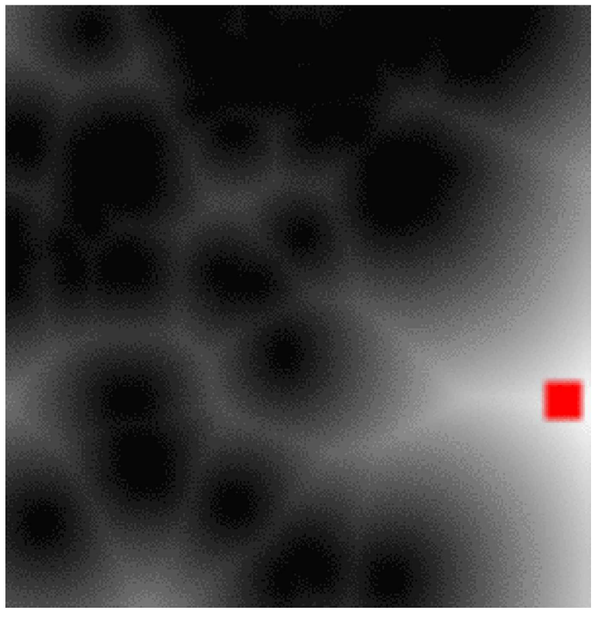
Real-time landing site selection and hazard avoidance: Testing and Validation



True image



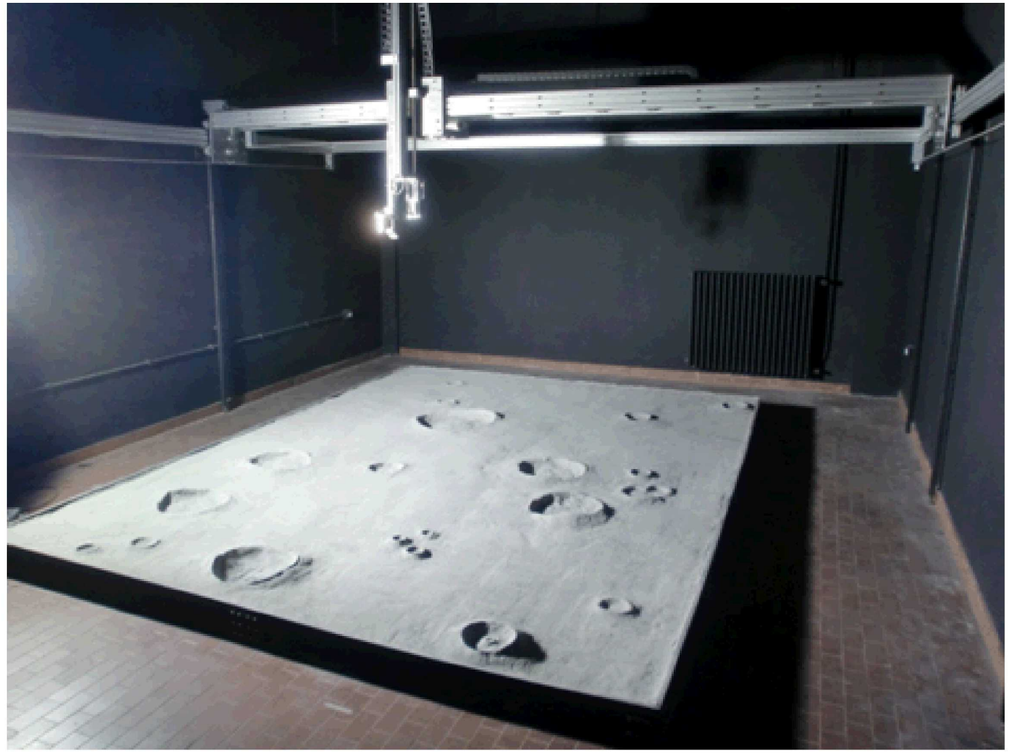
Safe/hazard mask



Hazard map

From Sim to Real: Transfer Learning

- **Lunar Landing Simulator: School of Aerospace Engineering, University of Rome**
  - **Cartesian robot** installed on a faithful reproduction of the lunar region located at the Mare Serenitatis (23°Nord, 14°East) with a scale of 1cm:2000m.
  - The lunar soil simulant is made by sifted basalt powder, while the craters are made by calk using moulds.



Dimentions	Length: 4 m y-axis Width: 3 m x-axis Heigth: 2 m z-axis
Step motor	Torque: 820 N cm Accuracy: 64 steps/mm Min. speed: 2 mm/min Max. speed: 12 m/min Power: 400 W

Recent Papers UA-SSEL on GNC/ML

- **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.
- 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.
- **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.
- Scorsoglio, A., **Furfaro, R.,** Linares, R. and Gaudet, B., (2020). Image-based Deep Reinforcement Learning for Autonomous Lunar Landing. In *AIAA Scitech 2020 Forum* (p. 1910).

Recent Papers UA-SSEL on GNC/ML

- 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?

This research was funded by the Sandia National Laboratories Laboratory-Directed Research and Development Program. Sandia National Laboratories is a multi-mission laboratory managed and operated by National Technology & Engineering Solutions of Sandia, LLC, a wholly owned subsidiary of Honeywell International Inc., for the U.S. Department of Energy's National Nuclear Security Administration under contract DE-NA0003525