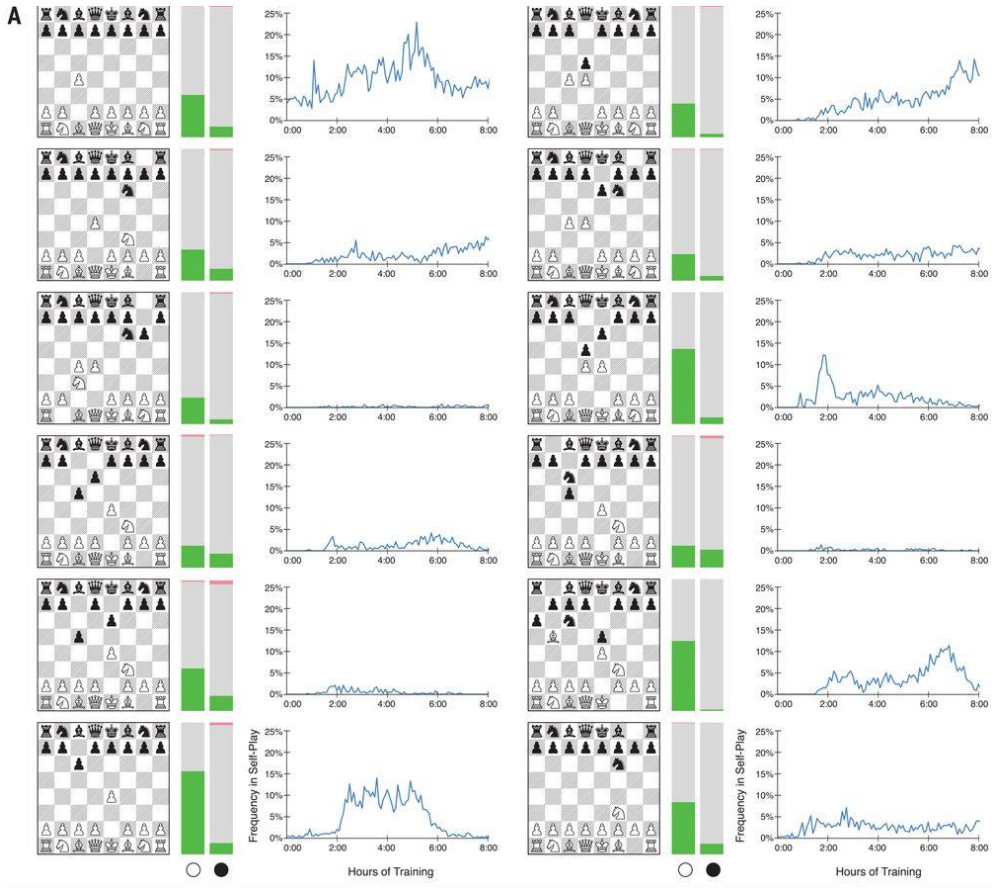
Special Topics: Reinforcement Learning

Virtual summer school on Machine Learning in Electron Microscopy

Rama K. Vasudevan
CNMS/ORNL
18th August 2023

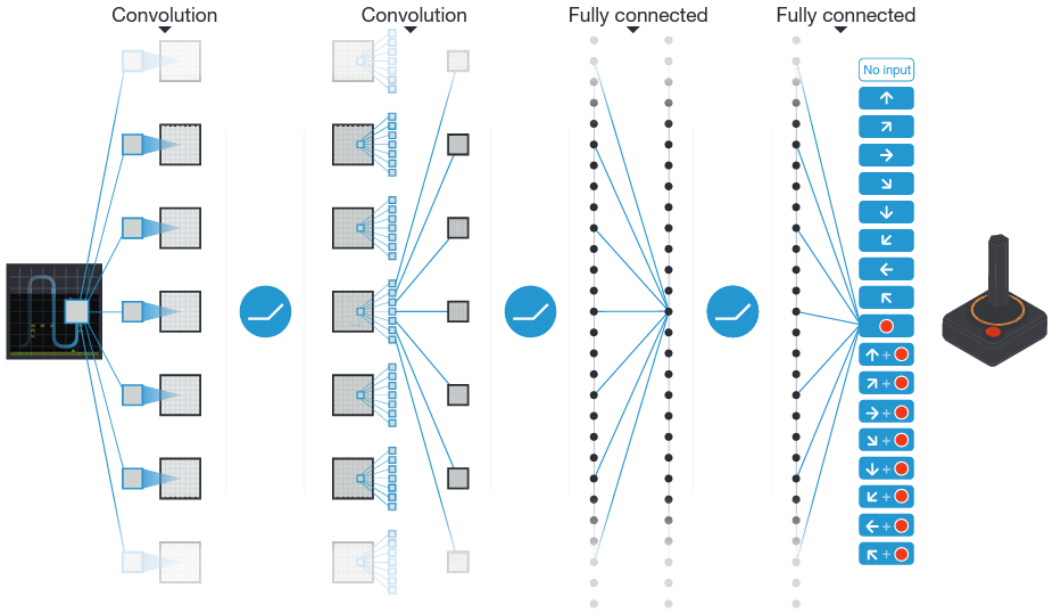
Reinforcement Learning (RL)

AlphaZero playing Go, Chess, Shogi



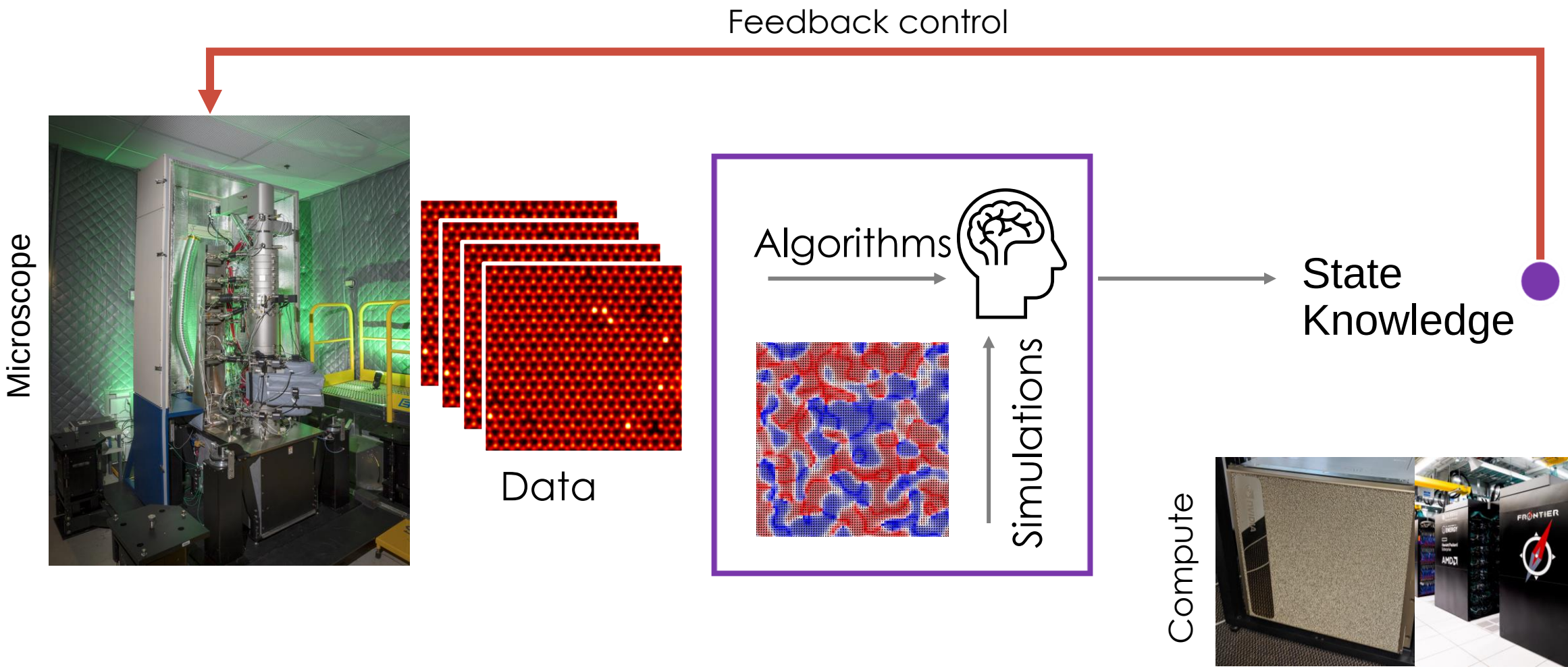
Silver et al. Science 362, 1140 (2018)

Deep Q Learning for Atari Games

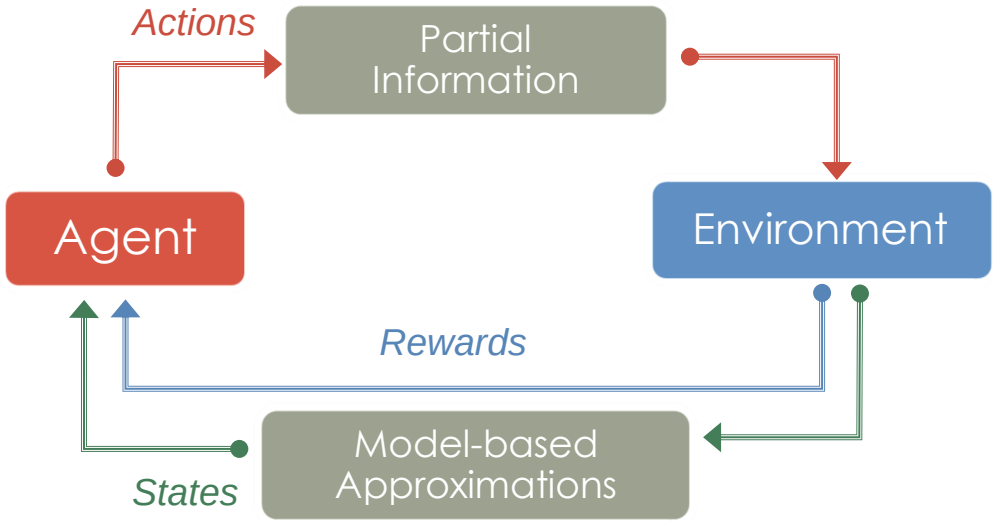


Mnih et al. Nature 518, 529 (2015)

But RL can be useful in many situations….



Reinforcement Learning Basics



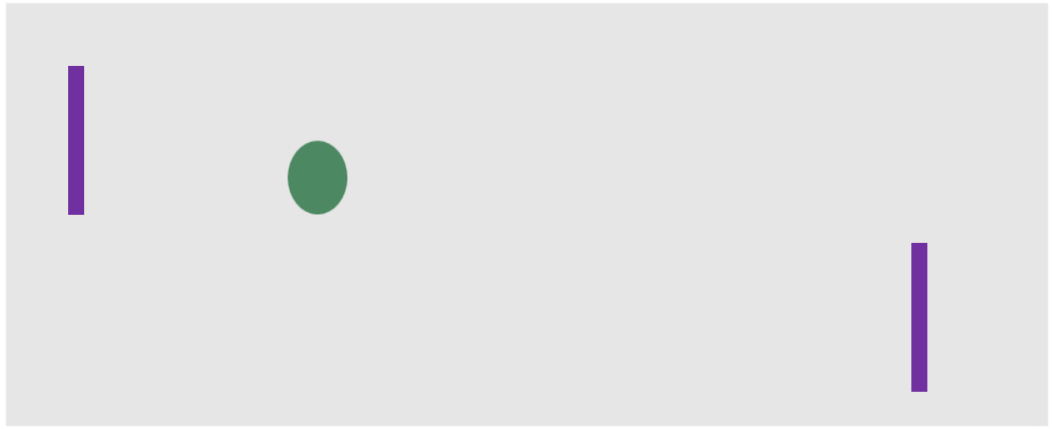
- RL is neither supervised nor unsupervised – it deals with optimal decision making in uncertain environments
- Variants of RL include on-policy learning and off-policy learning, and fully offline learning.

- We wish to learn stochastic policies that map states to actions to maximize some reward
- Two main types of RL: model-based, and model-free
- We can deal with continuous and discrete action spaces
- Policies are generated that aim to maximize expected future rewards emitted from the environment

Reinforcement Learning Intuition

- In RL, we are not using supervised or unsupervised machine learning. **We don't know the 'correct' answer through supervision.** So where to start? Answer: Trial random actions

Pong



Up-Down-Up-Down-Up-Up	**Bad**
Down—Down-Up-Up-Down	**Good**
Down—Down-Up-Up-Down	**Good**
Up—Down-Up-Down-Down	**Bad**

Supervised ML

maximize

$$\sum_i \log p(y_i|x_i)$$

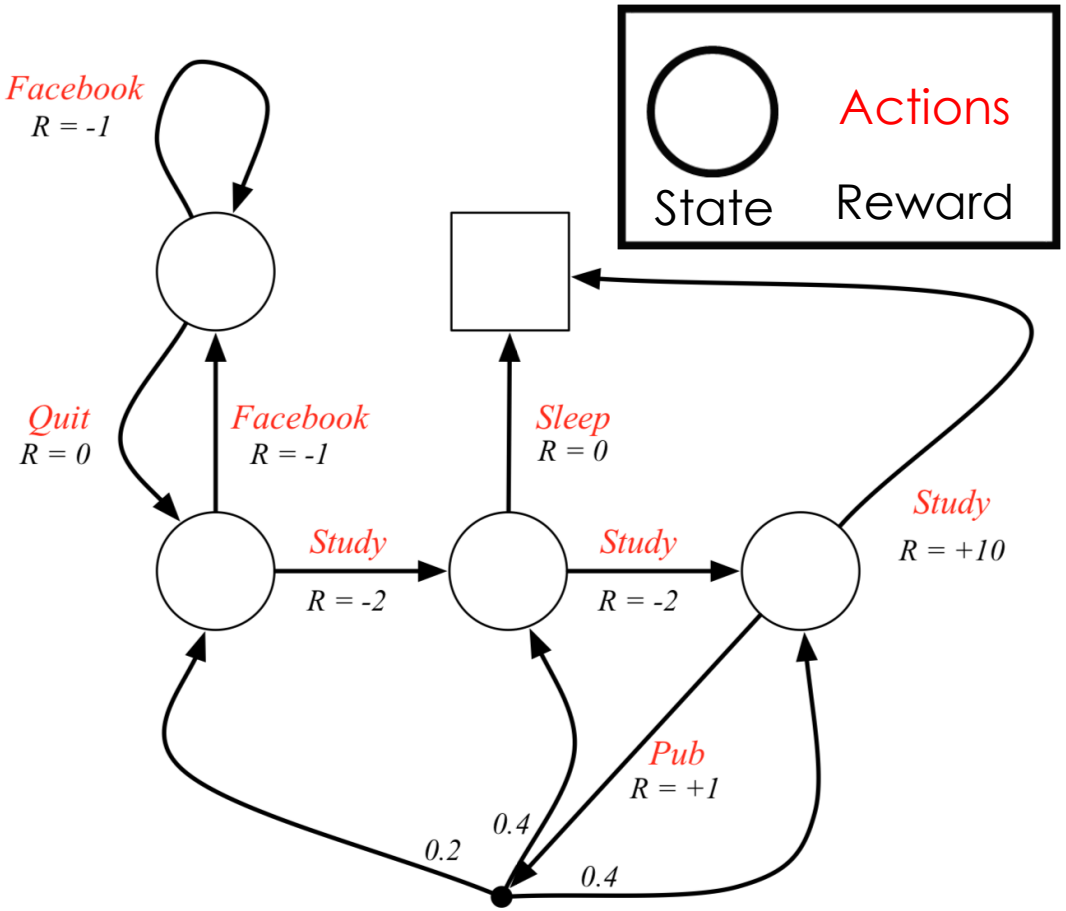
Reinforcement Learning

maximize $\log p(y_i|x_i)$

maximize $-1*\log p(y_i|x_i)$

Markov Decision Process (MDP)

- We consider (generally) Markov Process – the future depends only on the current state



From David Silver's RL lecture notes

- We can sample the Markov Process:
  - Study-Study-Study-Pass
  - Study-Study-Pub-Start-Study-Sleep
  - Facebook-Start-Facebook-Study-Sleep

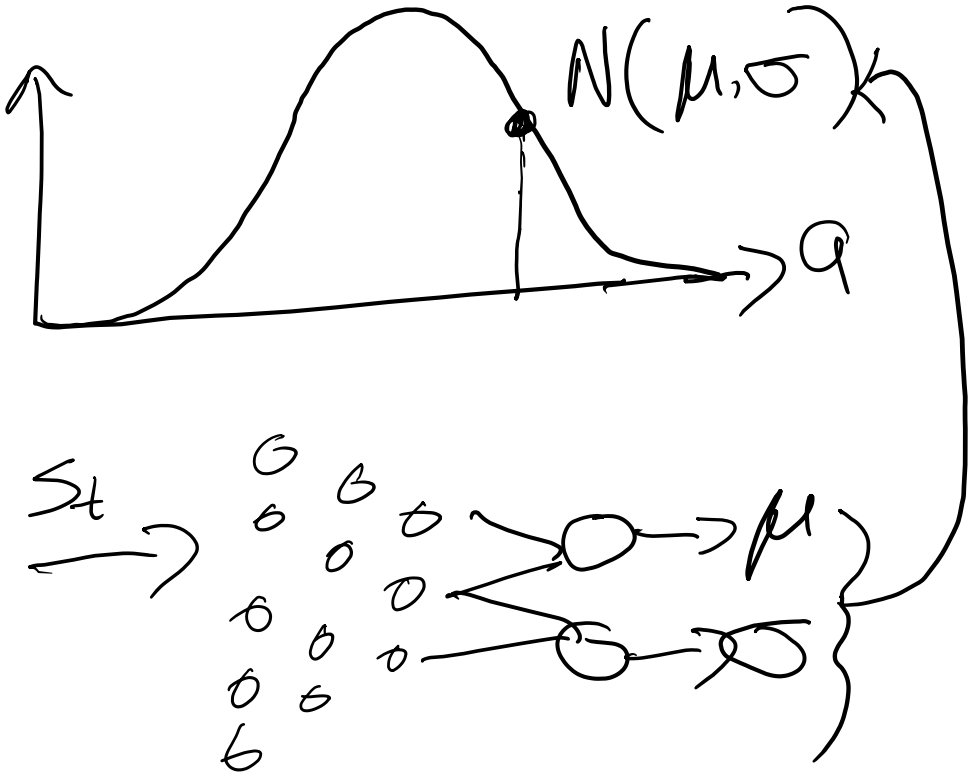
An extension to this is a **Markov Decision Process**, where we also obtain rewards from one state transition to another, and we perform actions that get us into different states

$(S_1, A_1, R_1, \ldots, S_n, A_n, R_n)$

Definitions – Policies

Stochastic Policy π_θ, $a_t \sim \pi(\cdot|s_t)$

e.g.



: continuous actions sampled this way.

Definitions

τ = Trajectory = $(s_0, a_0, s_1, a_1, \dots)$

sequences of states and actions

In stochastic environment, it's governed by env. dynamics, conditioned on (s_t, a_t)

$$s_{t+1} \sim \underbrace{P}_{\text{env. dynamics}}(\cdot \mid s_t, a_t)$$

Definitions - Rewards + Returns

Reward functions R

$$r_t = R(s_t, a_t, s_{t+1}) \quad (\text{or } r_t = R(s_t, a_t))$$

Discounted Returns:

$$R(\tau) = \sum_{t=0}^{\infty} \gamma^t r_t \qquad \gamma \in (0, 1)$$

→ 0: no horizon, 1: infinite horizon.

The RL Problem

How to optimize policy π_θ to maximize the expected returns, i.e.

$$J(\pi_\theta) = \mathop{\mathbb{E}}_{\tau \sim \pi_\theta}[R(\tau)]$$

Discrete: $\sum_{i=0}^{\infty} x_i p_i$

Note: $\mathbb{E}(x) = \int_{-\infty}^{\infty} x f(x) dx$

We want to use grad. asc. on θ, i.e.

$$\theta_{k+1} = \theta_k + \nabla_\theta J(\pi_\theta)|_{\theta_k}$$

RL Problem

$$\nabla_\theta J(\pi_\theta) = \nabla_\theta \mathbb{E}[R(\tau)]$$

$$= \nabla_\theta \int P(\tau|\theta) R(\tau)$$

Expand Expectation.

$$= \int_\tau \nabla_\theta P(\tau|\theta) R(\tau)$$

log derivative trick

$$= \int_\tau P(\tau|\theta) \nabla_\theta \log P(\tau|\theta) R(\tau)$$

$$= \mathop{\mathbb{E}}_{\tau \sim \pi_\theta} [\nabla_\theta \log P(\tau|\theta) R(\tau)]$$

Expectation

log Trick:

$$\nabla_x \log f(x) = \frac{f'(x)}{f(x)} \Rightarrow f'(x) = f(x) \nabla_x \log f(x)$$

$$\nabla_\theta \log p(\tau|\theta) = \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) \longrightarrow \text{no env. dep. on } \theta$$

(this comes from $p(\tau|\theta) = p_0(s_0) \prod_{t=0}^{T} p(s_{t+1}|s_t, a_t) \pi_\theta(a_t$

and some manipulations)

RL Policy Gradient

$$\nabla_\theta J(\pi_\theta) = \mathop{\mathbb{E}}_{\tau \sim \pi_\theta}\left(\nabla_\theta \log P(\tau|\theta) R(\tau)\right)$$

$$= \mathop{\mathbb{E}}_{\tau \sim \pi_\theta}\left(\sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) R(\tau)\right)$$

In reality we estimate this expectation numerically e.g. with sample mean:

$$\hat{g} = \frac{1}{|D|} \sum_{\tau \in D} \sum_{t=0}^{T} \nabla_\theta \log \pi_\theta(a_t|s_t) R(\tau)$$

$D = \{\tau_i\}_{i=1,\ldots,N}$

Standard Policy Gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) G_t]$$

What does it mean though?

It means that we want to move the policy parameters in the direction of increased returns G, and not move it in the directions where G would actively reduce

Actor-Critic (A2C) Method

Standard Policy Gradient:

$$\nabla_\theta J(\theta) = \mathbb{E}_\tau[\sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) G_t]$$

But G_t has VERY high variance, leading to very slow convergence. Subtract a baseline for faster convergence

Define Advantage Function:

$$A(s_t, a_t) = r_{t+1} + \gamma V_v(s_{t+1}) - V_v(s_t)$$

This is equal to the Q function minus the Value Function

Actor-Critic Policy Gradient:

$$\nabla_\theta J(\theta) \sim \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t)(r_{t+1} + \gamma V_v(s_{t+1}) - V_v(s_t))$$

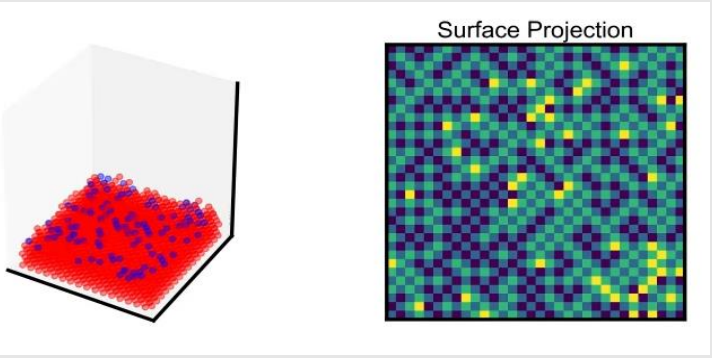
Separate Neural Net needed for Value Estimation ('Critic')

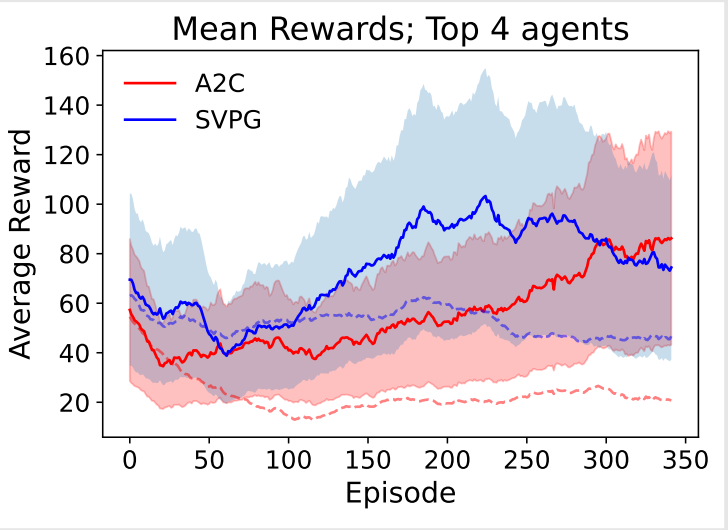
$$= \sum_{t=0}^{T-1} \nabla_\theta \log \pi_\theta(a_t|s_t) A(s_t, a_t)$$

Encourages actions that increase 'advantage'

Reinforcement Learning Examples

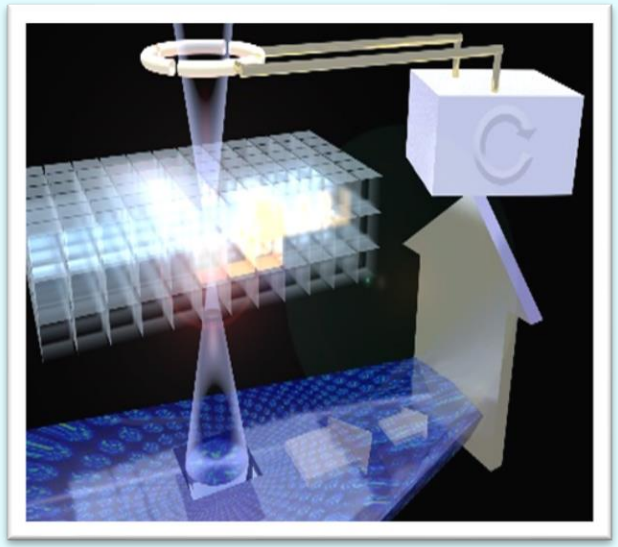
Synthesis Trajectory

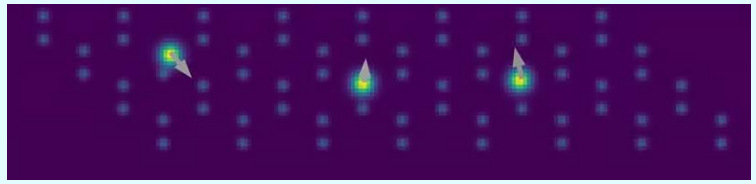


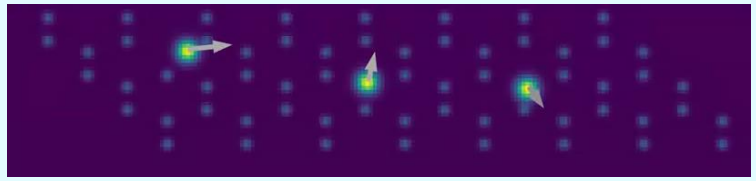


arXiv:*2006.15644 (2020)*

Atomic Manipulation

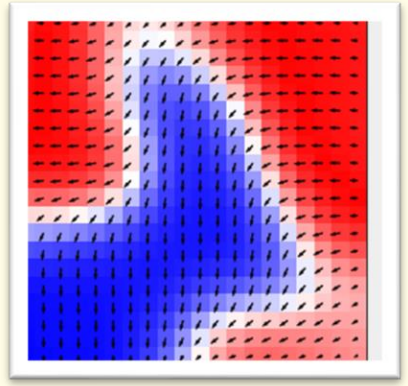


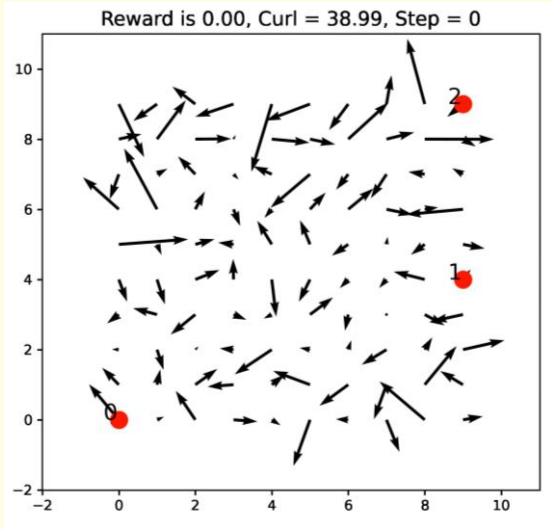




Nanotech. *33*, 115301 (2021)

Topological Defects

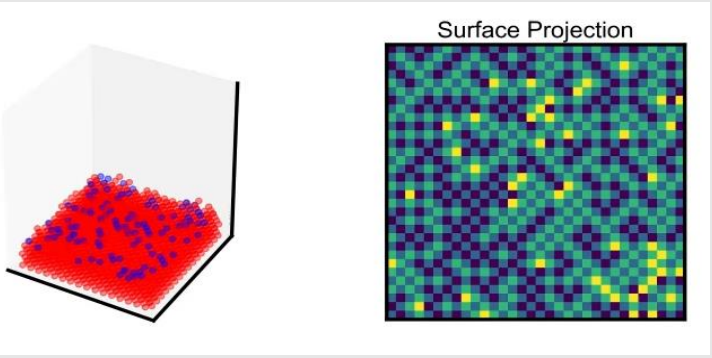


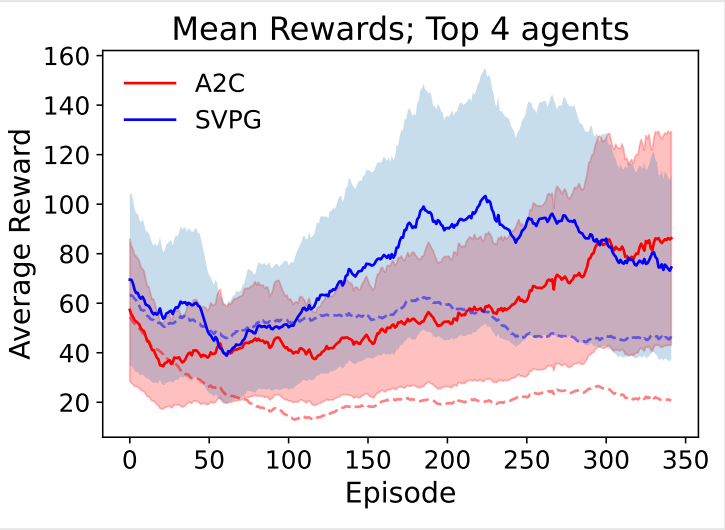


arXiv:2202.10988 (2022)

Reinforcement Learning Examples

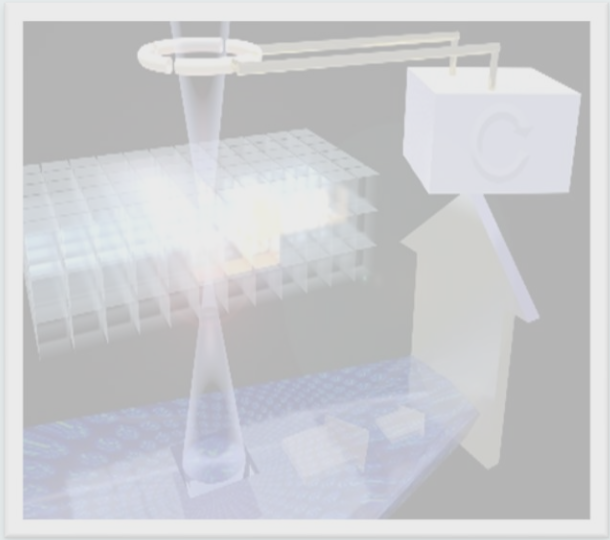
Synthesis Trajectory





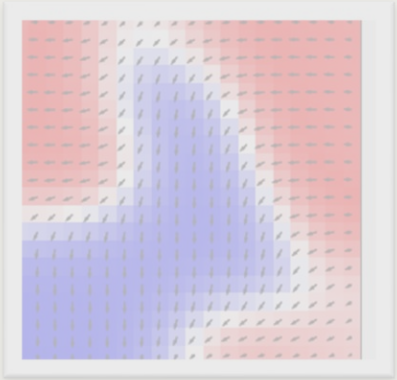
arXiv:*2006.15644 (2020)*

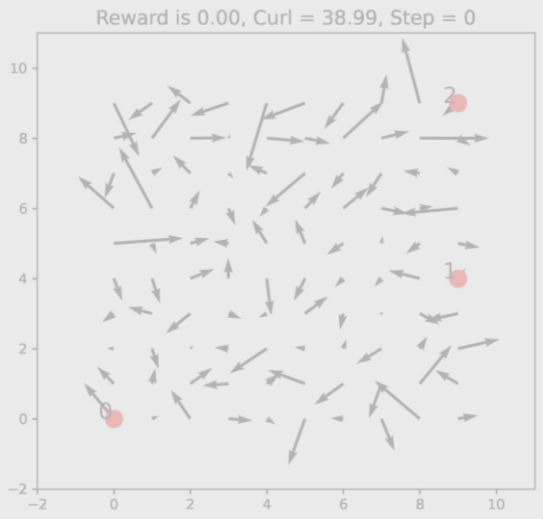
Atomic Manipulation



Nanotech. *33*, 115301 (2021)

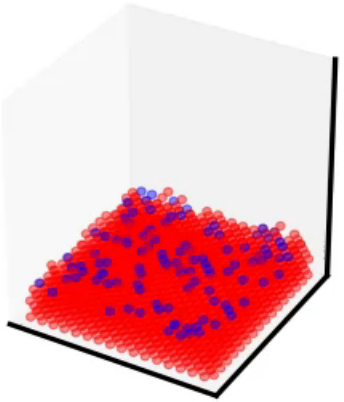
Topological Defects

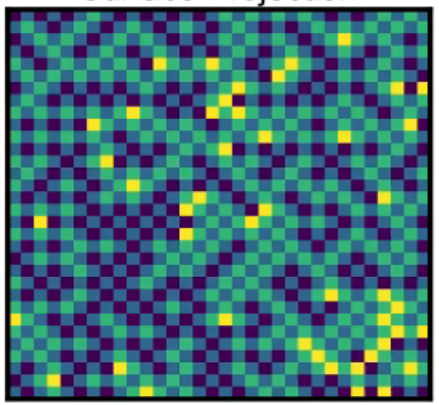


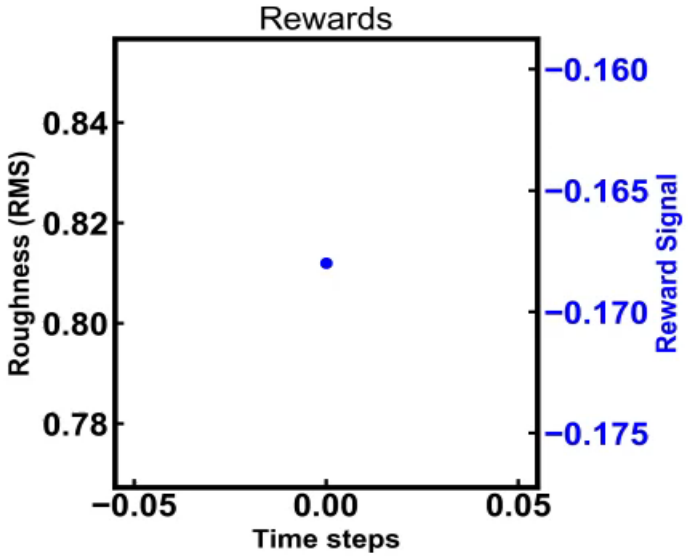


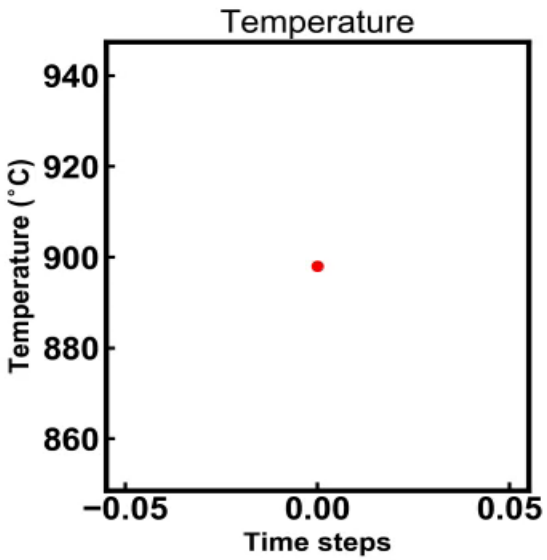
arXiv:2202.10988 (2022)

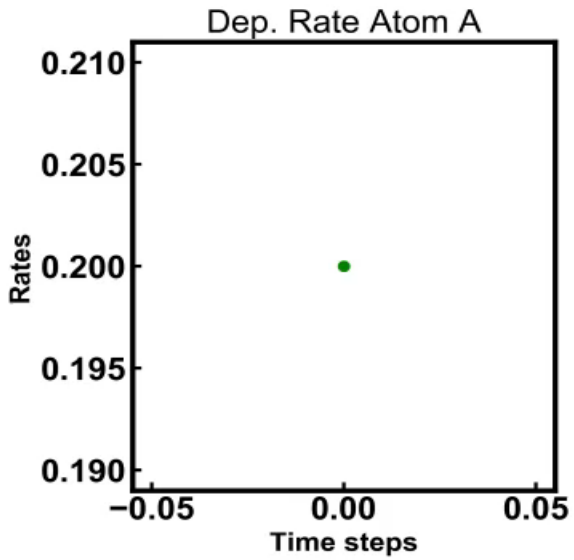
KMCEnv: A materials synthesis Environment in the OpenAI Gym Platform

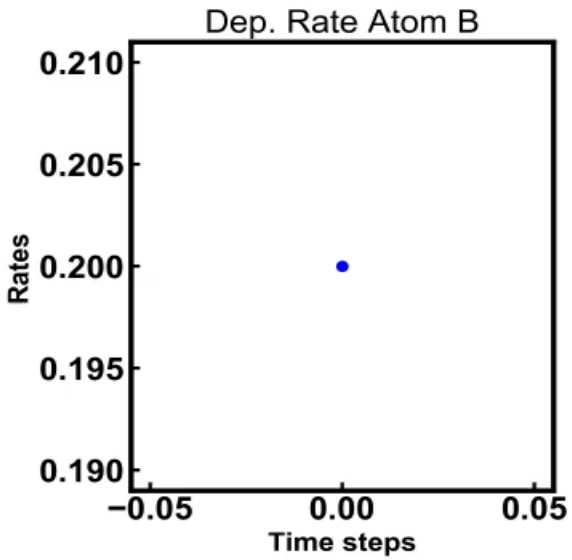






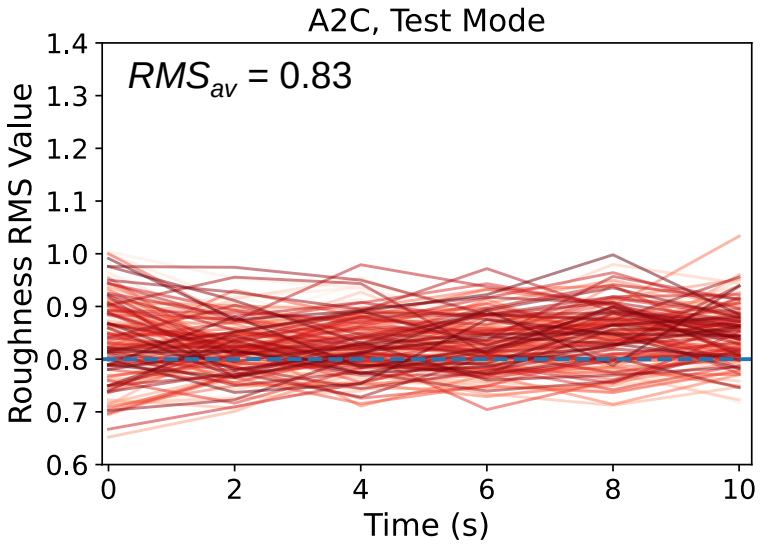




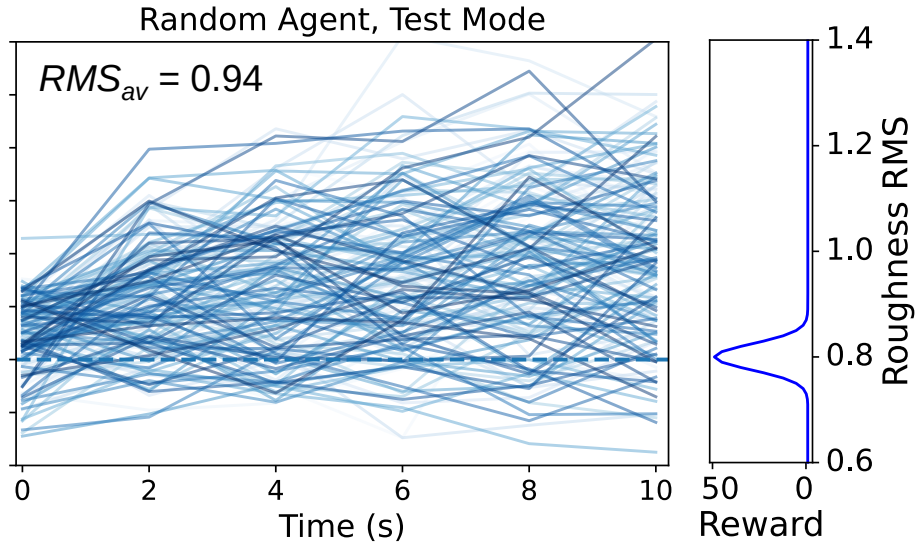


Results

Trained Agent



Random Agent

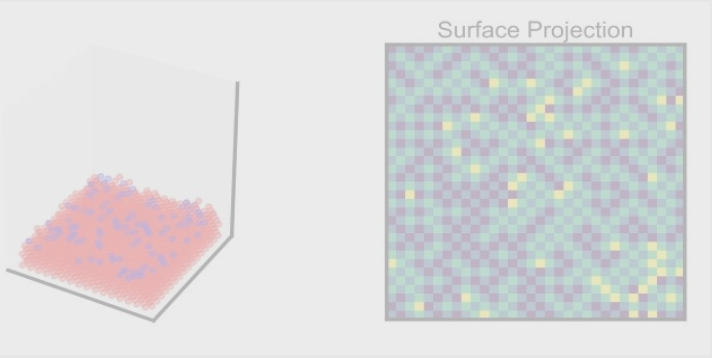


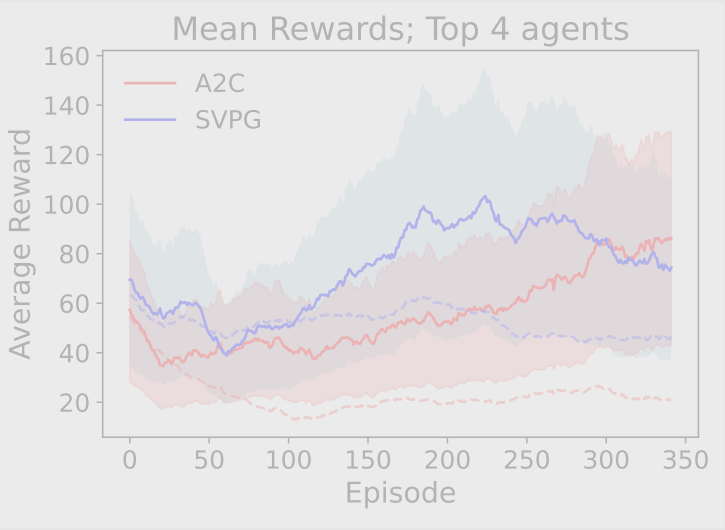
Compared to random actions, the trained agent can keep the material to within a specific roughness range throughout the deposition process

Liu et al. arXiv 2006.15644 (2020)

Reinforcement Learning Examples

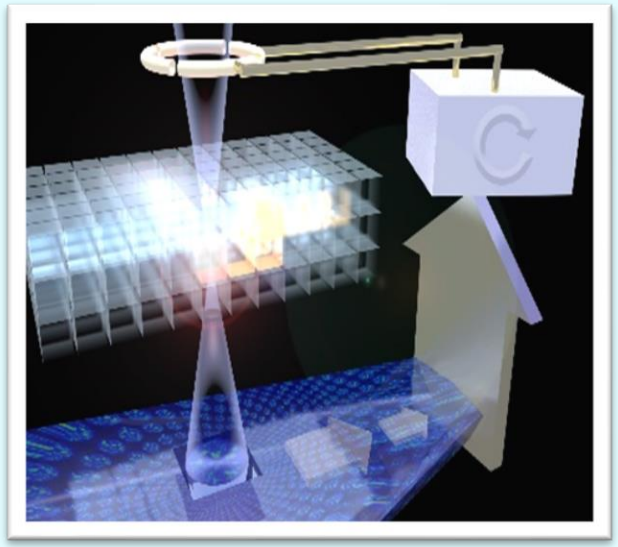
Synthesis Trajectory

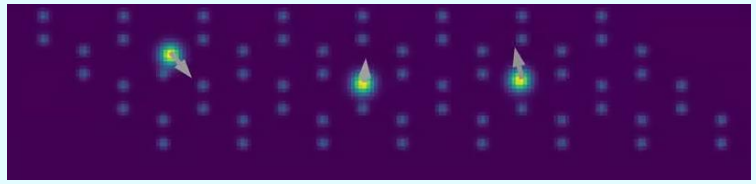


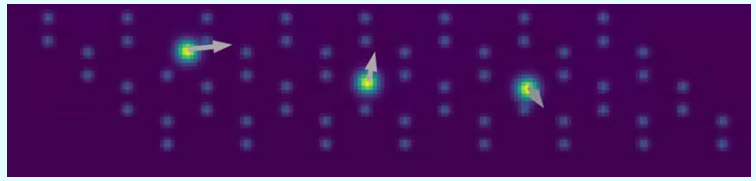


arXiv:*2006.15644 (2020)*

Atomic Manipulation

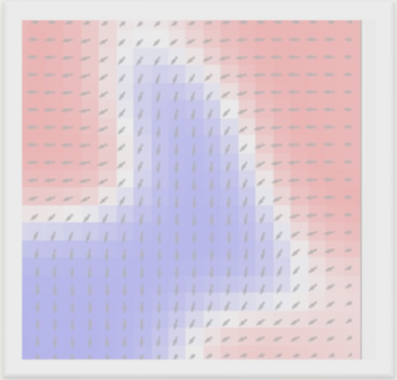


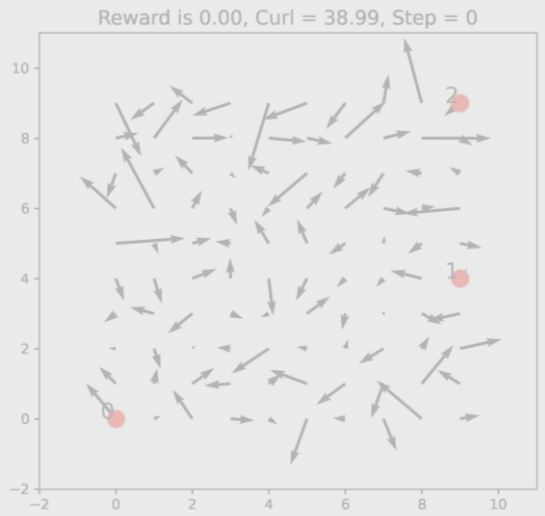




Nanotech. *33*, 115301 (2021)

Topological Defects

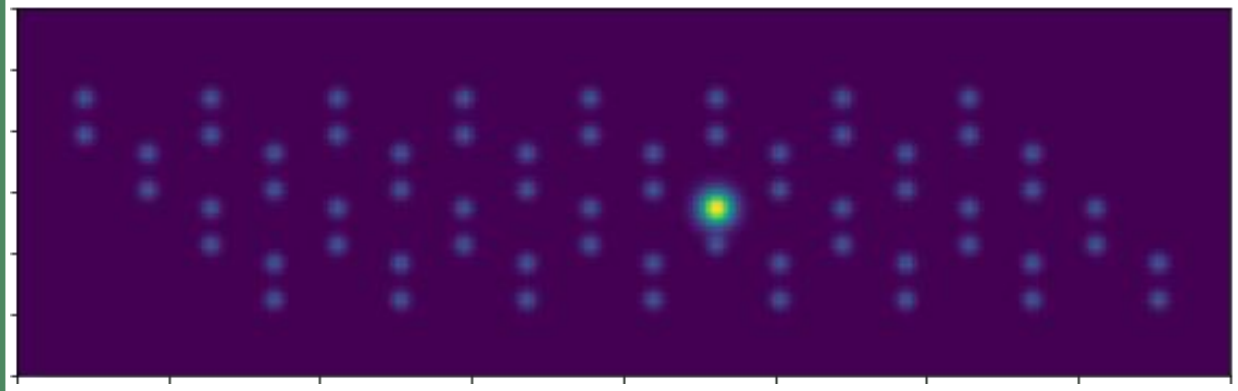


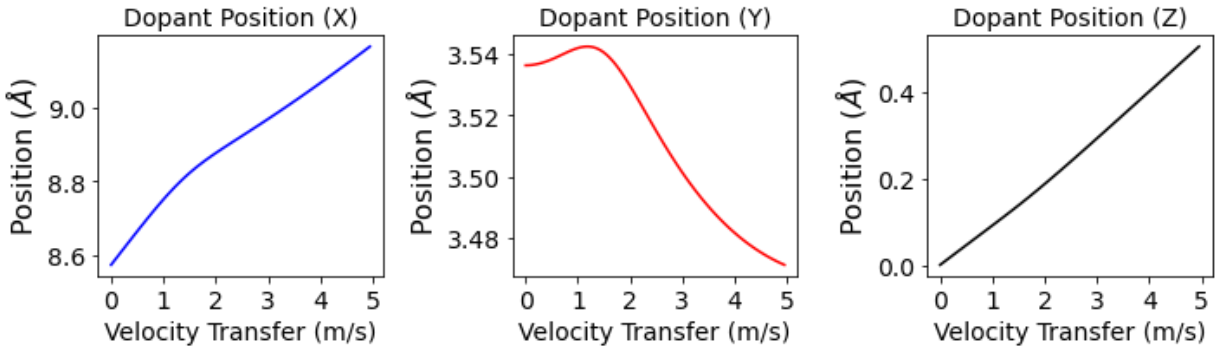


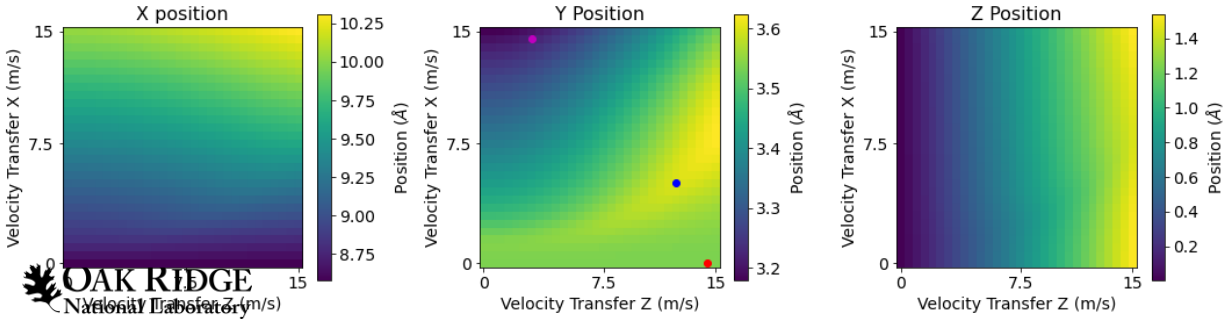
arXiv:2202.10988 (2022)

OAK RIDGE National Laboratory

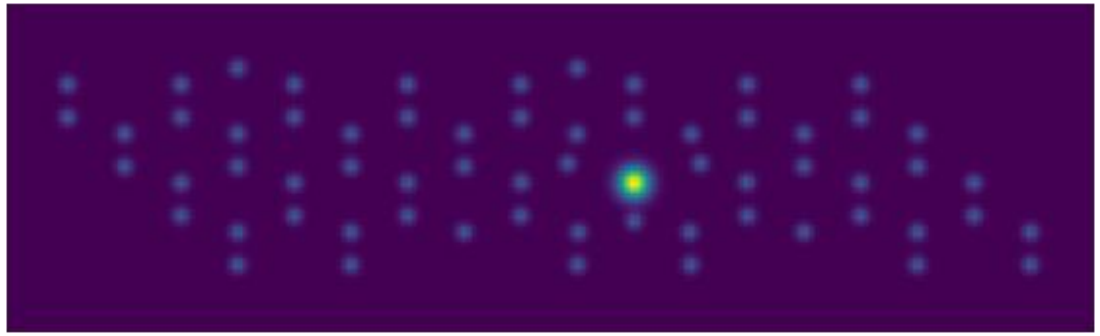
Molecular Dynamics Environment





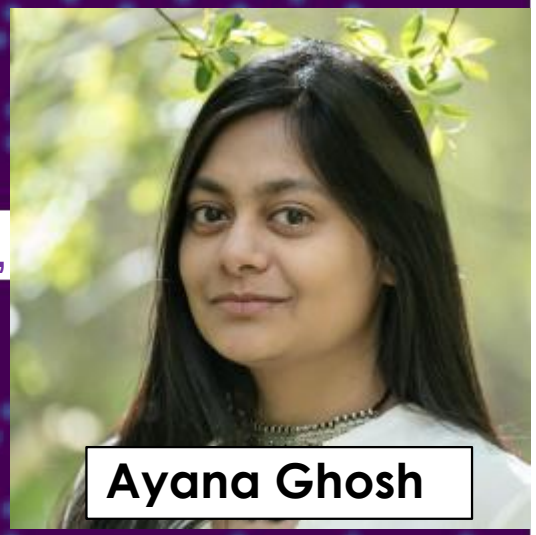


X_v = 1E-4 , Z_v = 14.5



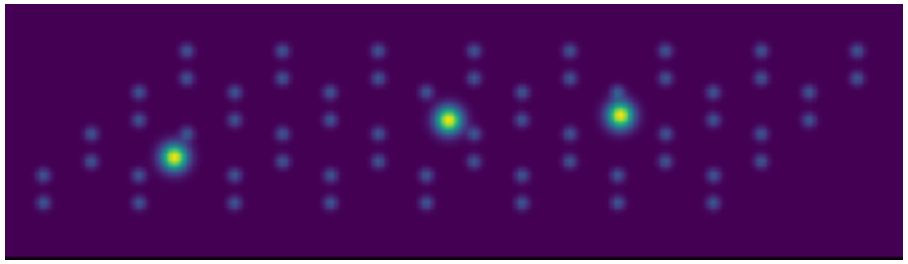
X_v = 5.2, Z_v = 12.4

X_v = 14.5 ,



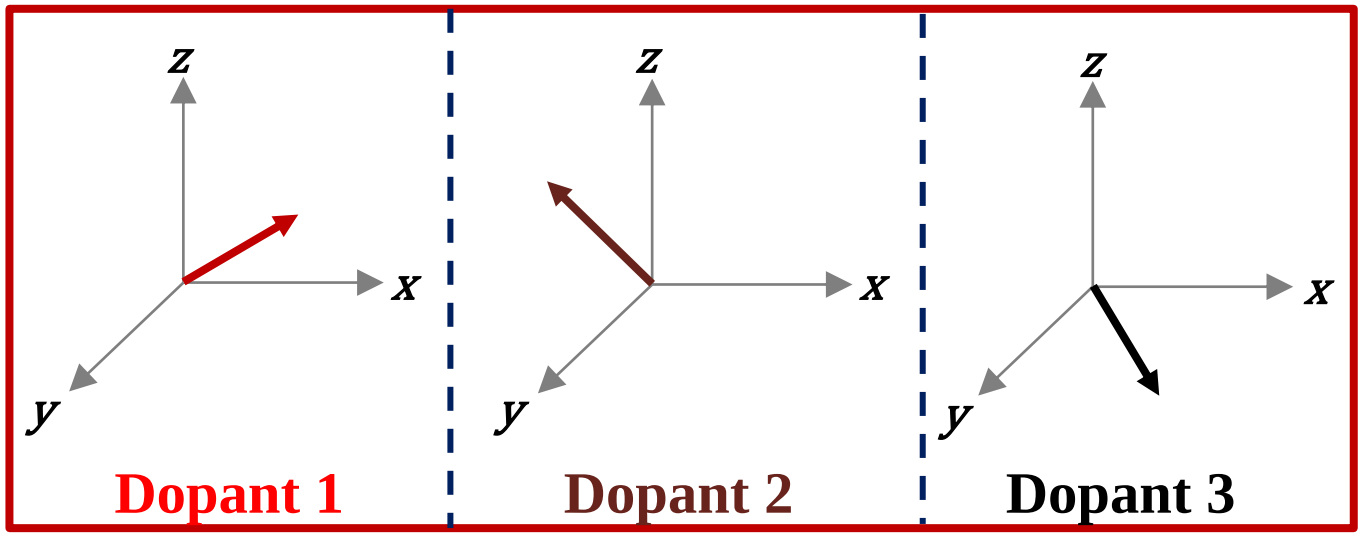
RL Environment for Atomic Fabrication: MD exploration

State: Si dopants in graphene



Goal: Move dopants together by changing momentum in MD

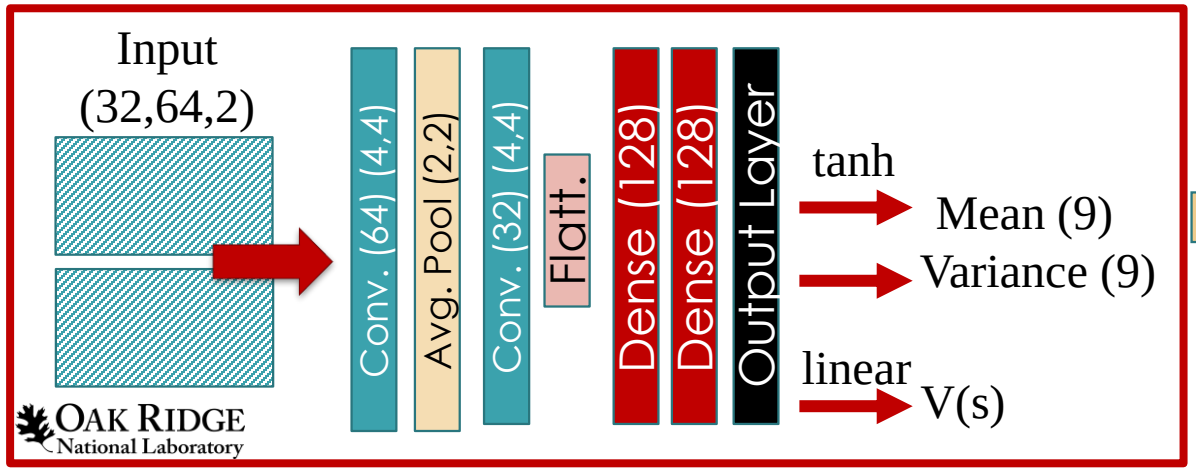
Action

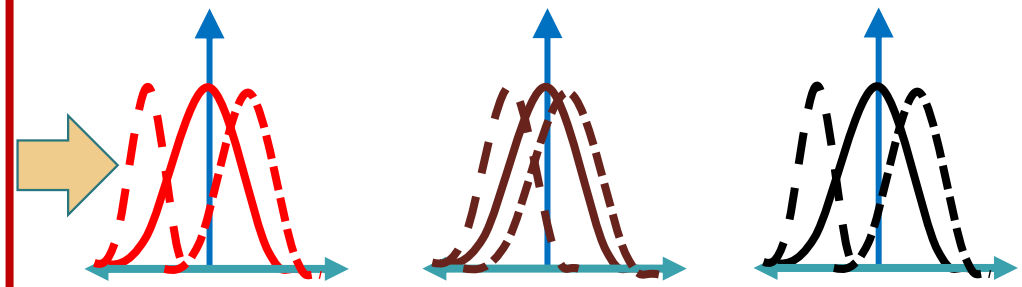


(v_{x1}, v_{y1}, v_{z1}) (v_{x2}, v_{y2}, v_{z2}) (v_{x3}, v_{y3}, v_{z3})

Sample

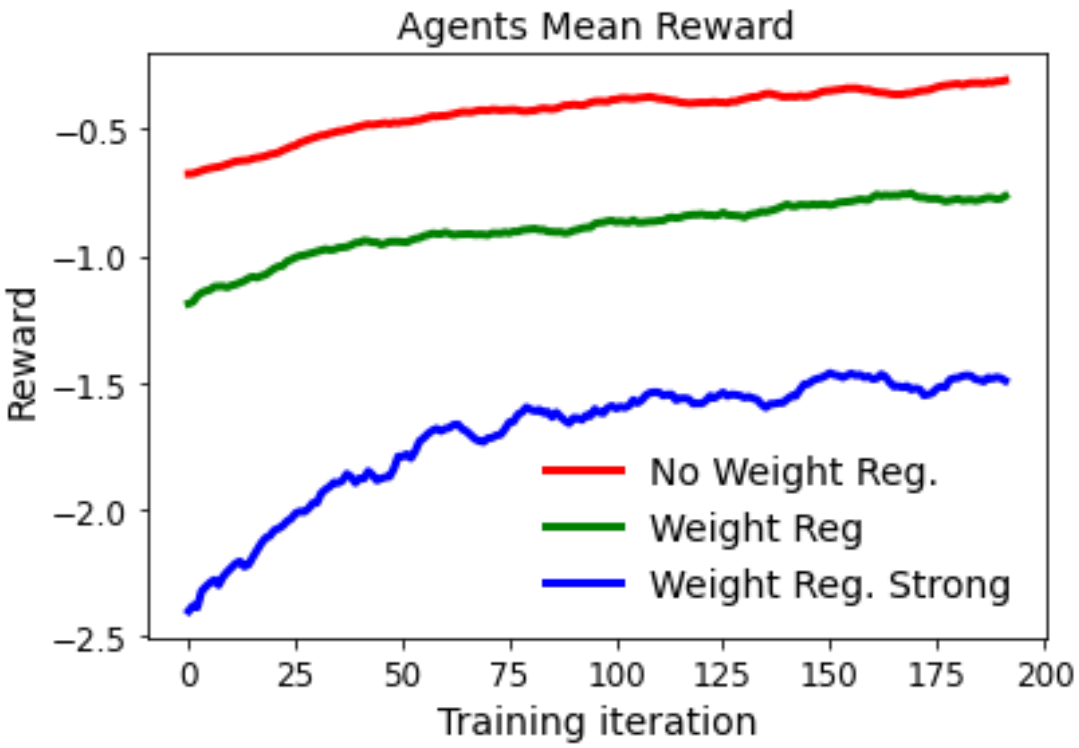
Actor/Critic Networks





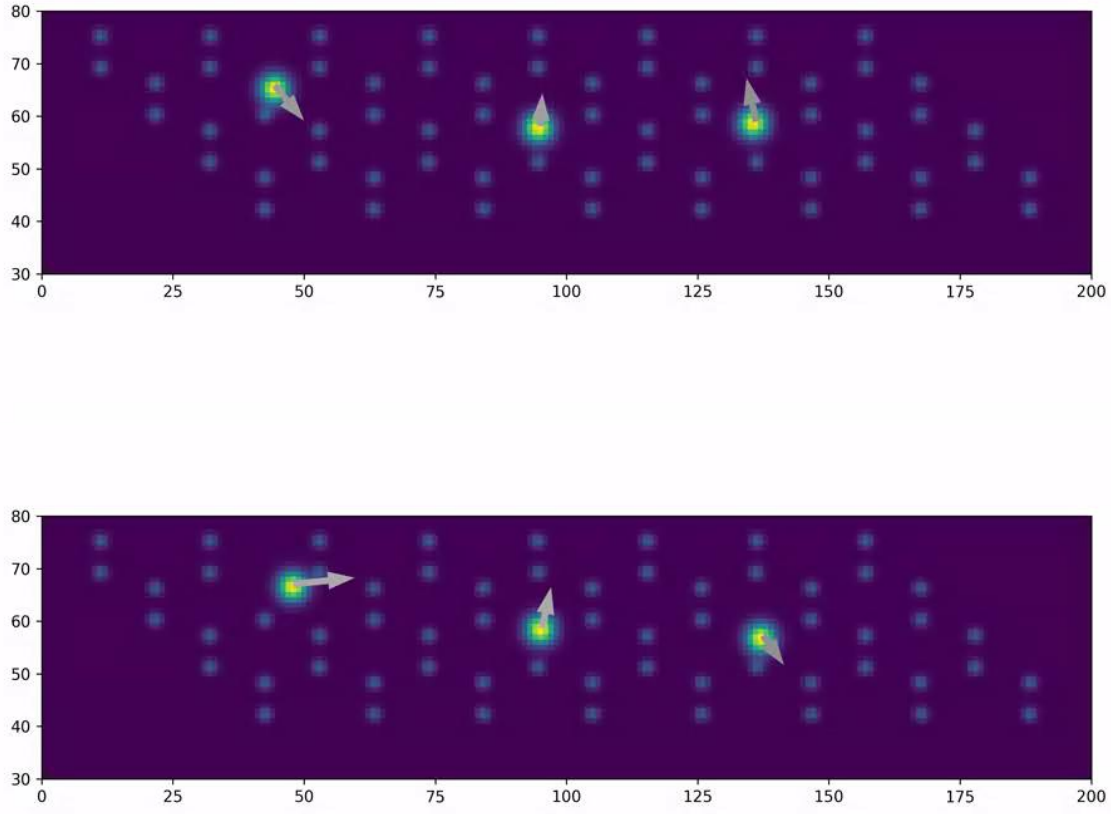
Multivariate Gaussian Policy

Results: SVPG

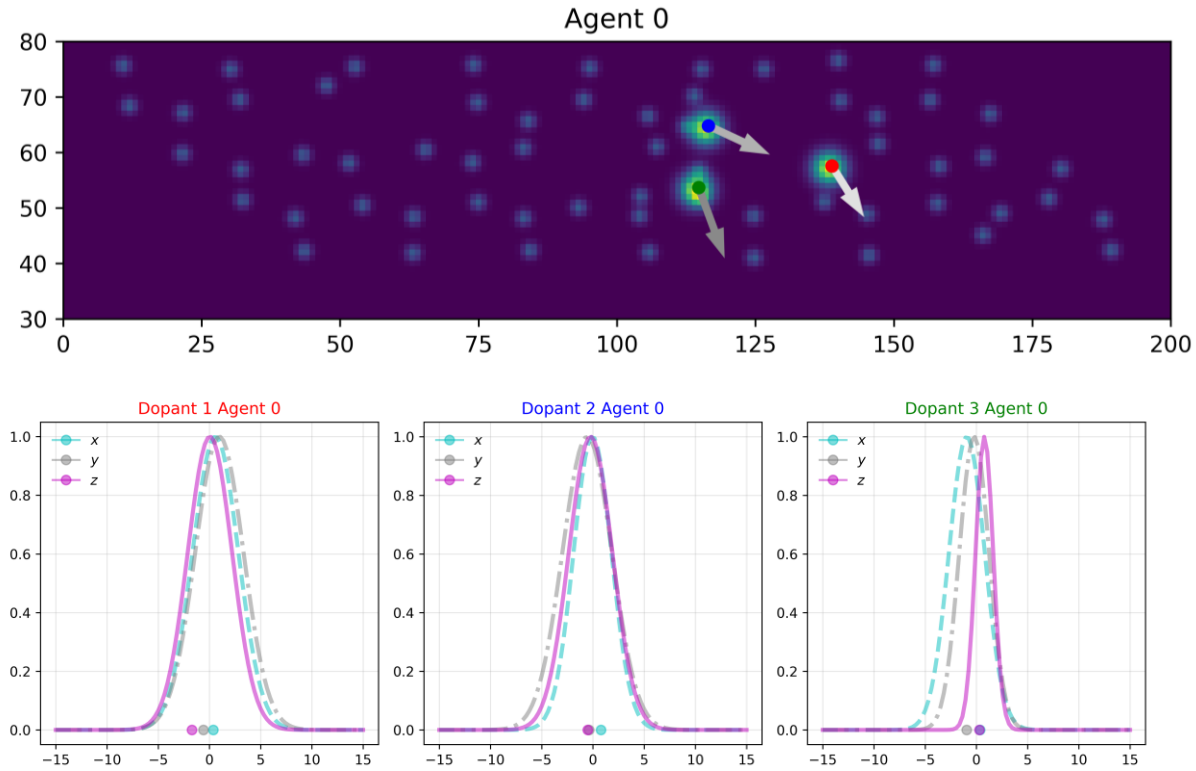


Nanotech. *33*, 115301 (2021)

Runs of trained agents



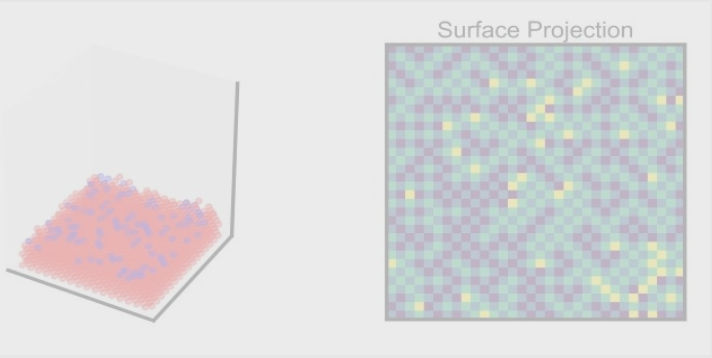
Results: Policy Inspection

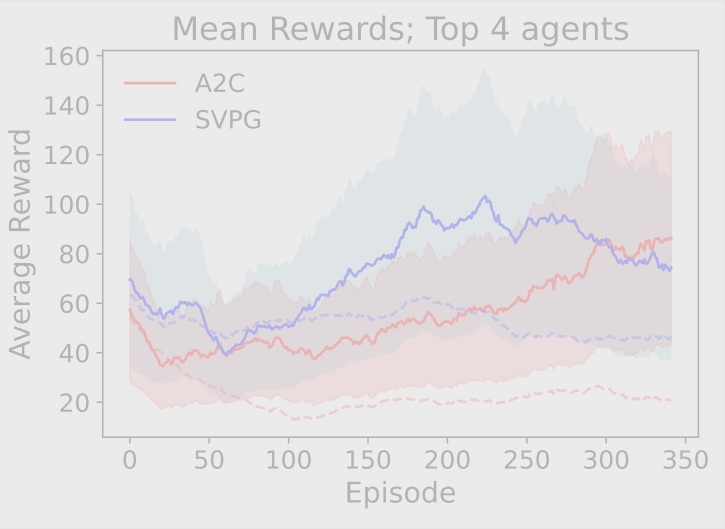


- Highly stochastic environment leads to conservative policies
- Z-component is not a delta function around zero- implies small z component is necessary to move dopant (also backed up by theoretical work)
- Policy inspection may become a useful tool to understanding the dynamics of the system -> relevant dynamics are learned, somewhat simplifying the problem.

Reinforcement Learning Examples

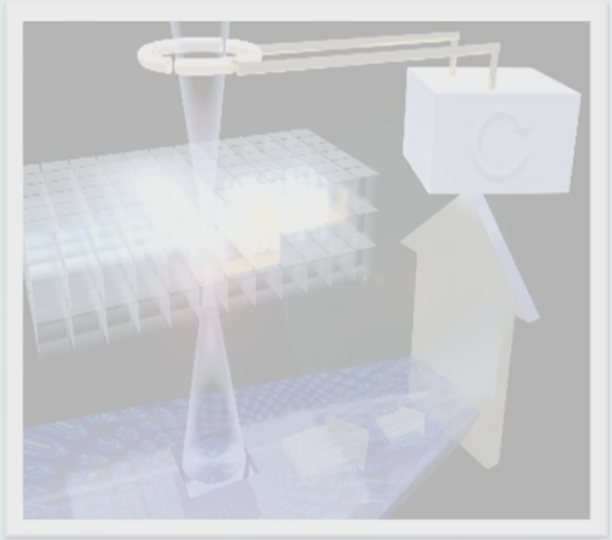
Synthesis Trajectory

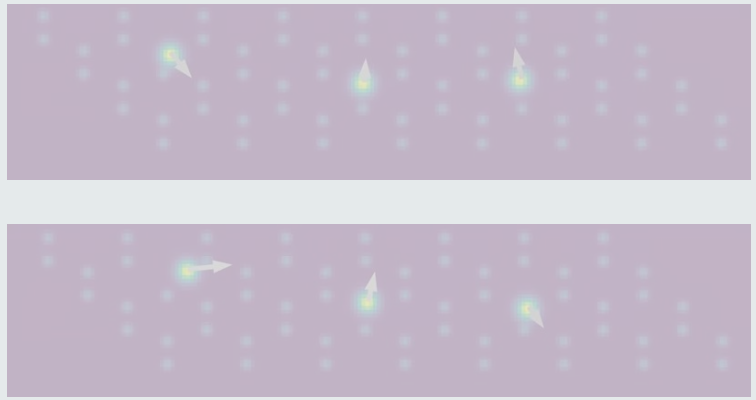




arXiv:*2006.15644 (2020)*

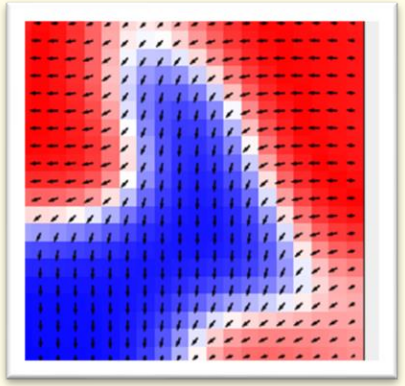
Atomic Manipulation

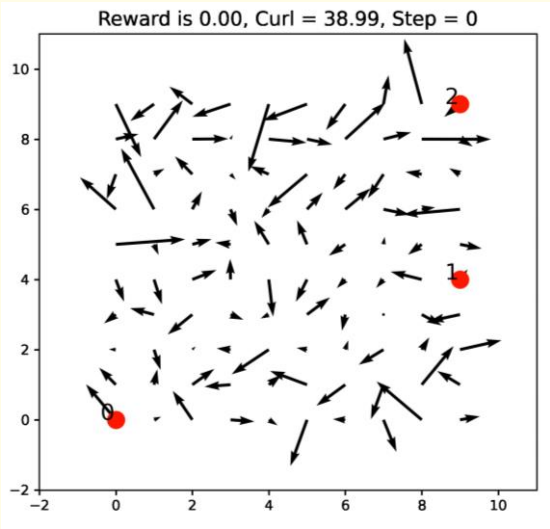




Nanotech. *33*, 115301 (2021)

Topological Defects

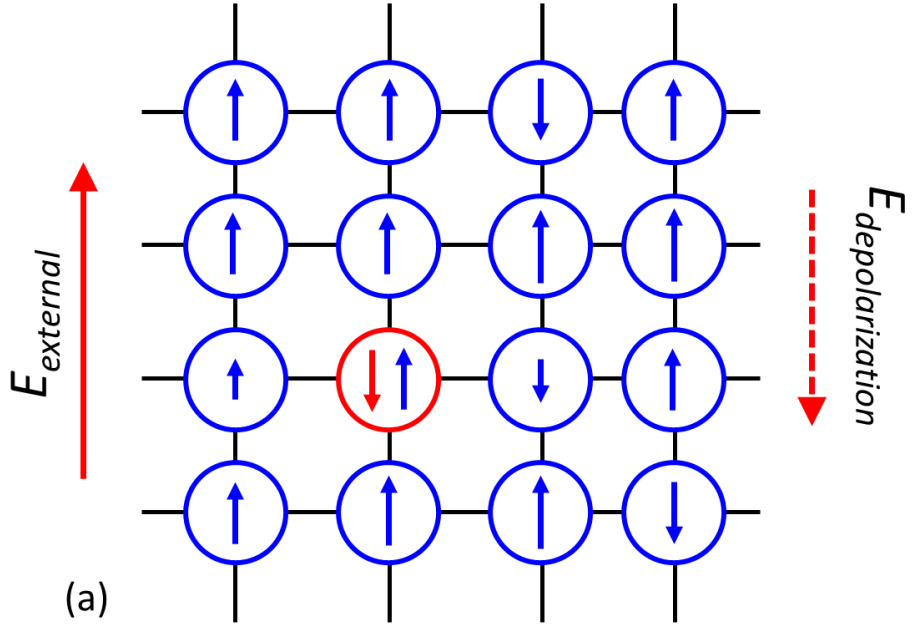


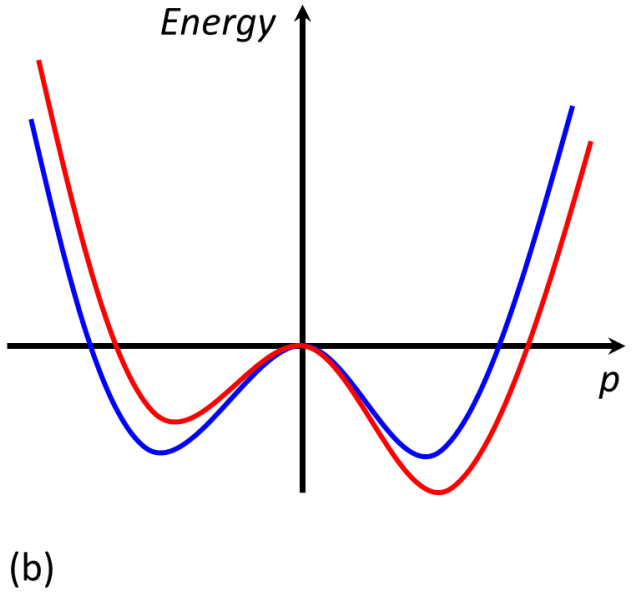


arXiv:2202.10988 (2022)

Discrete Landau Model

2D Discrete Landau Model

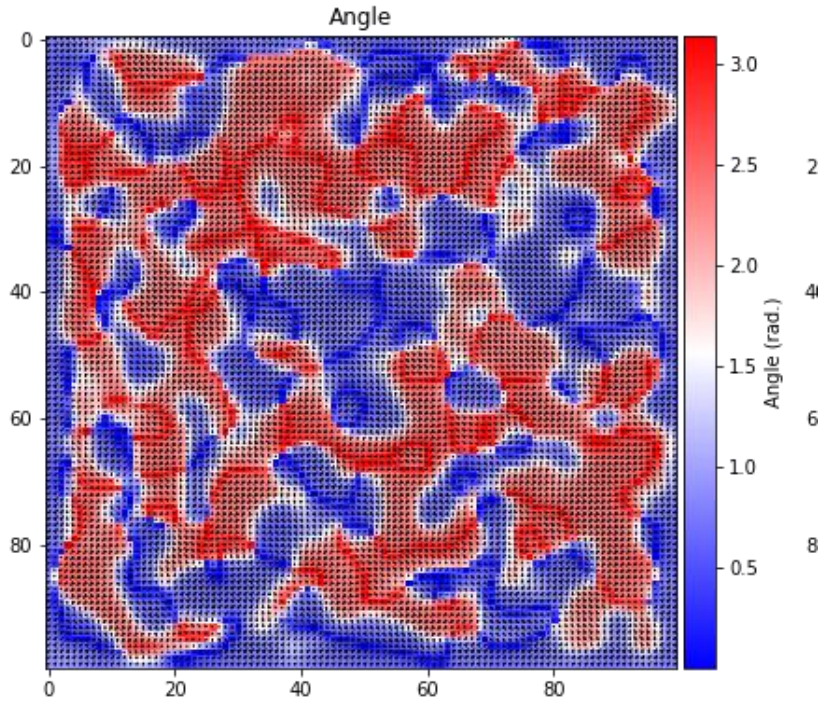




- Simple discrete time-dependent Landau formulation for ferroelectrics
- Code is available at github.com/ramav87/FerroSim

$$\mathrm{F} = \sum_{i,j}^{N} \left((\alpha/2)p^2 + (\beta/4)p^4 - E_{loc}p + K \sum_{k,l} \left(p_{i,j} - p_{i+k,j+l}\right)^2 \right)$$

$$\frac{dp_{ij}}{dt} = -\gamma^{-1} \left(\beta p_{ij}^3 + \alpha p_{ij} + K \sum_{k,l} \left(p_{ij} - p_{kl}\right) - E_{loc} \right)$$

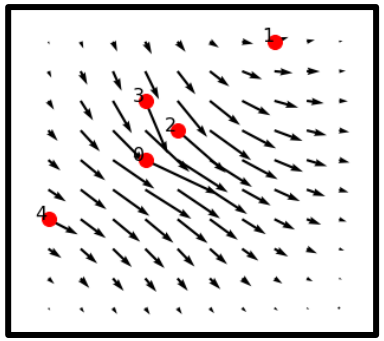


Kalinin, Ziatdinov, Vasudevan (JAP) (2020)

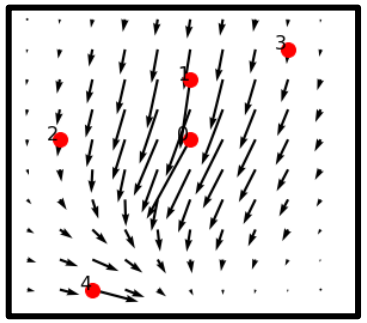
Action table

Action Index	Defect Selection	Move
0	--Pass--	--Pass--
1	0	Left
2	0	Down
3	0	Right
4	0	Up
5	1	Left
6	1	Down

Take action

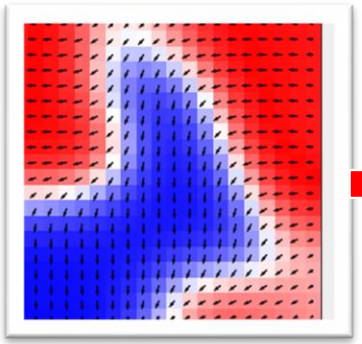


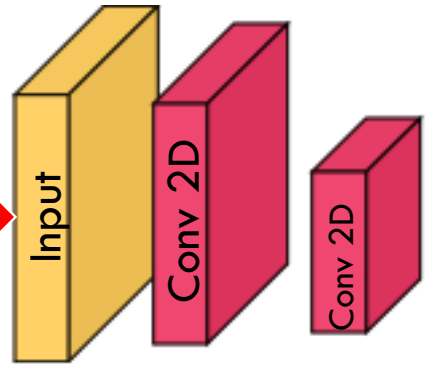
Update+Run Simulation



Next State

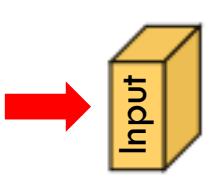
Polarization map





Flatten

Defects Table



Dense

Dense

Step No.

3

Dense

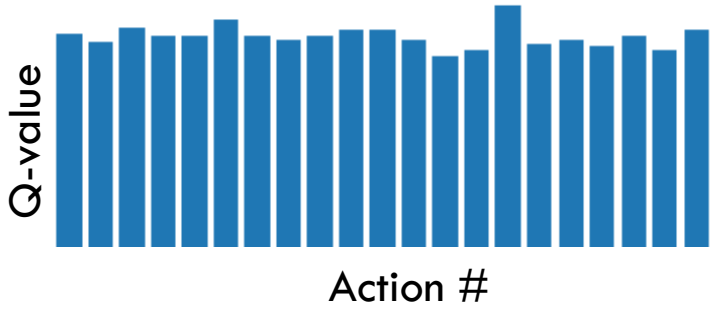
Concatenate

Dense

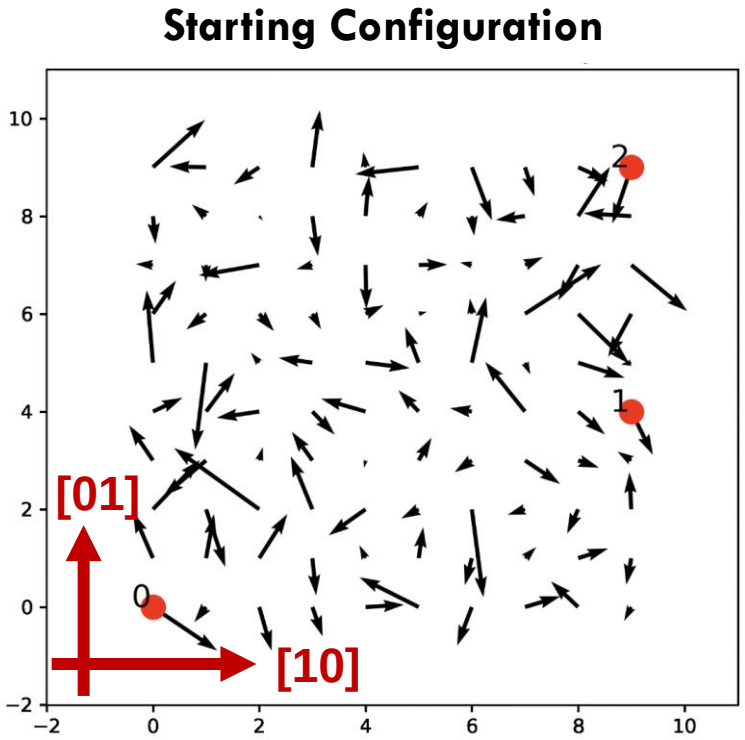
Dense

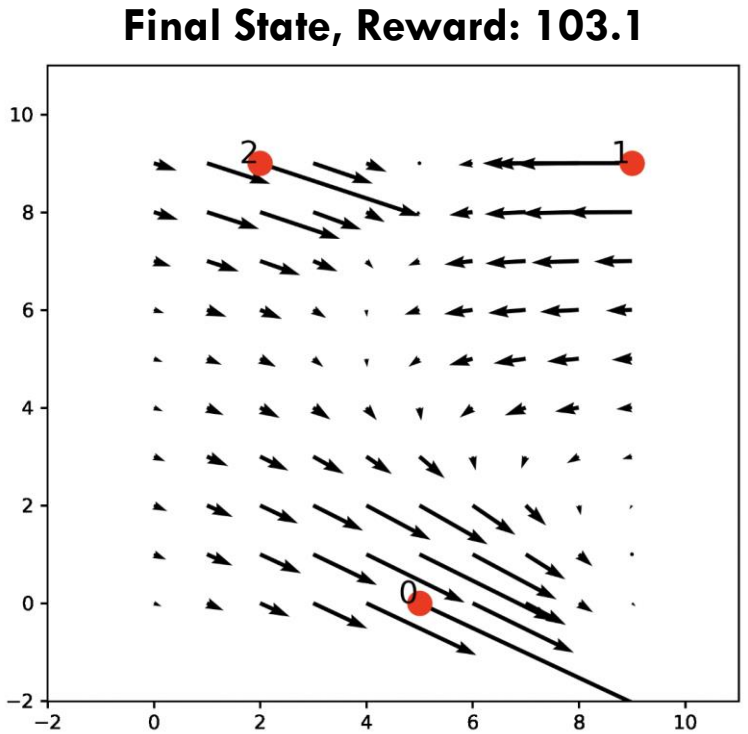
Q Values

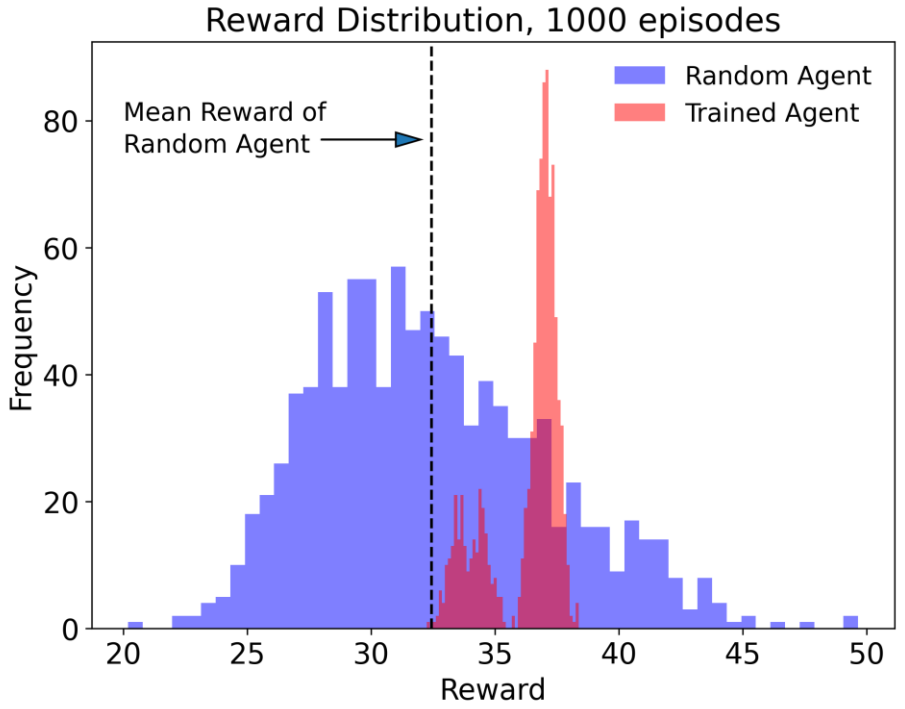
argmax (Q)



Vasudevan et al. MLST 3, 04LT03 (2022)

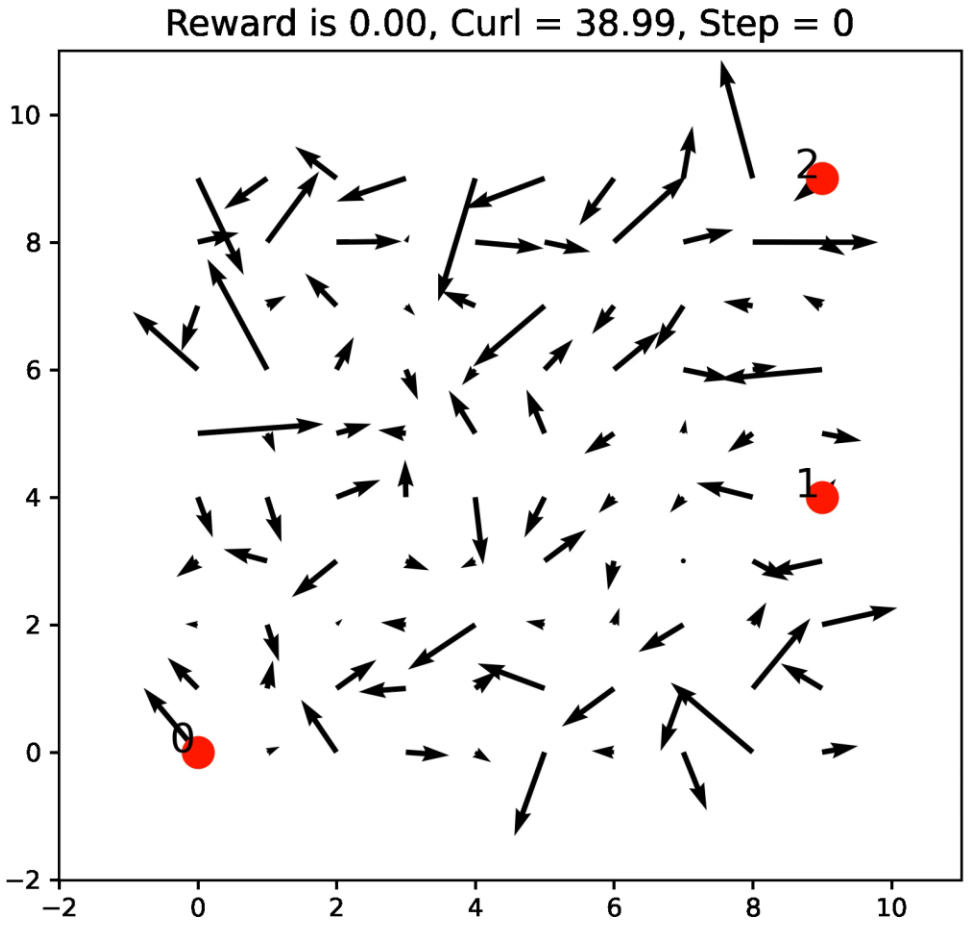


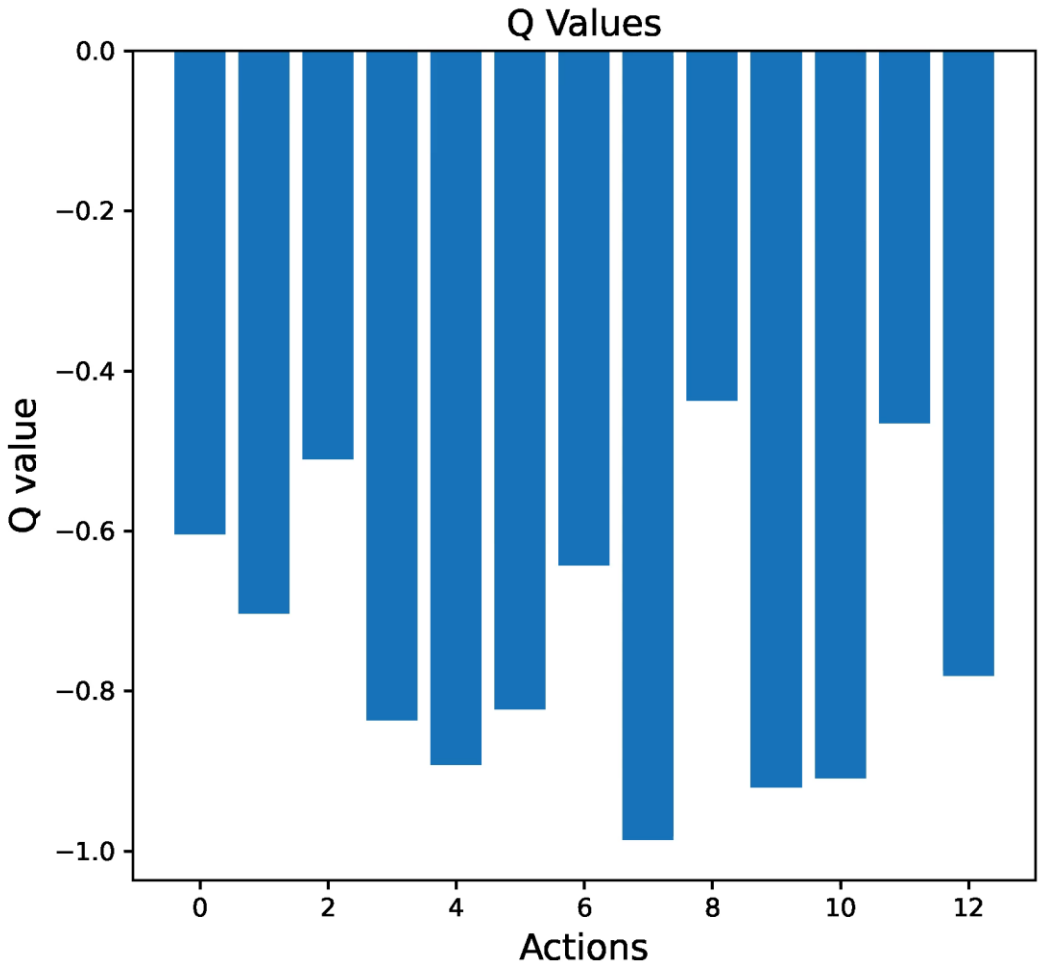


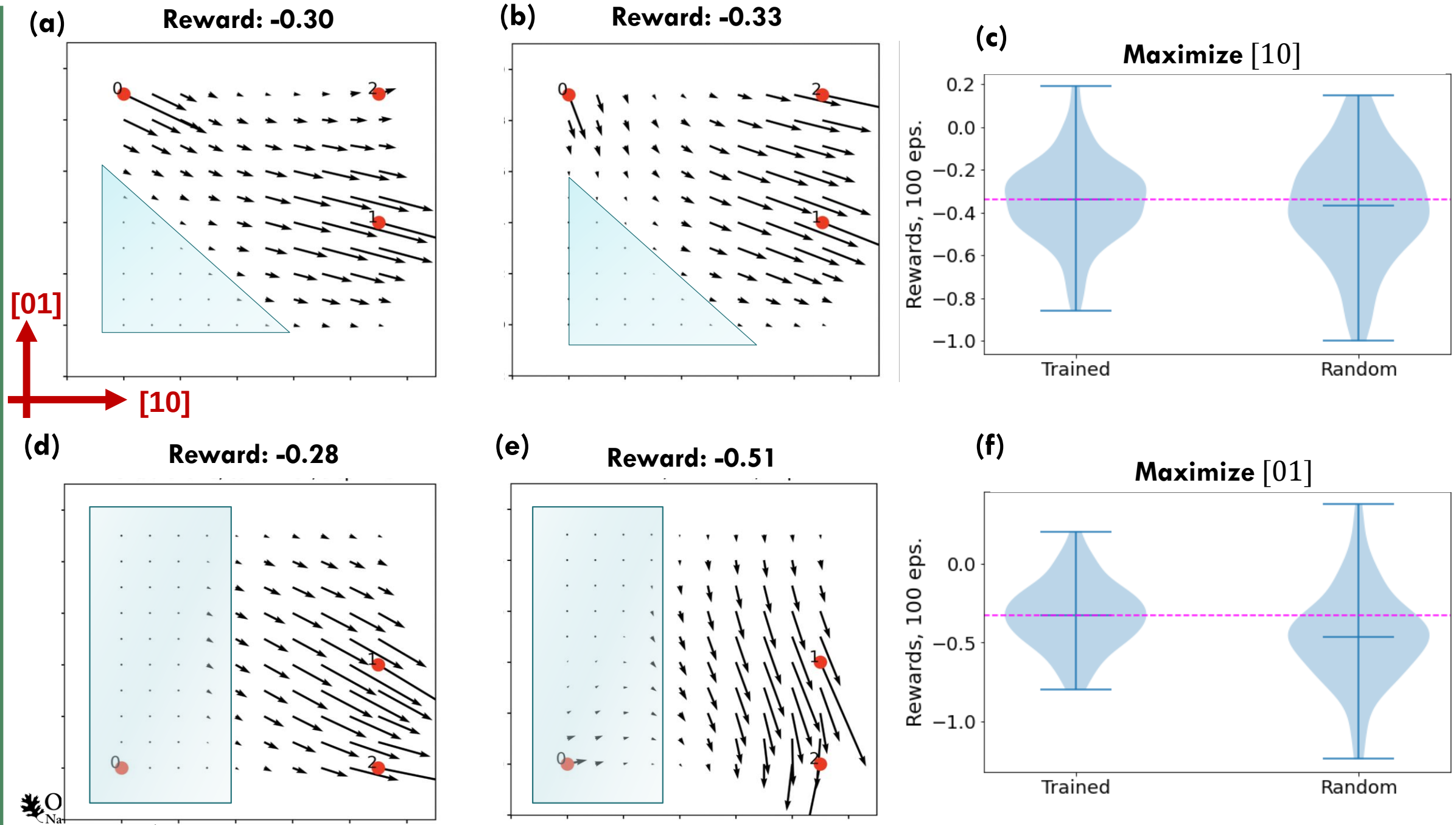


Vasudevan et al. MLST 3, 04LT03 (2022)

Agent moves the defects to triangular pattern






(a)
Reward: -0.30
(b)
Reward: -0.33
(c)
Maximize [10]
[01]
[10]
Rewards, 100 eps.
0.2
0.0
−0.2
−0.4
−0.6
−0.8
−1.0
Trained
Random
(d)
Reward: -0.28
(e)
Reward: -0.51
(f)
Maximize [01]
Rewards, 100 eps.
0.0
−0.5
−1.0
Trained
Random

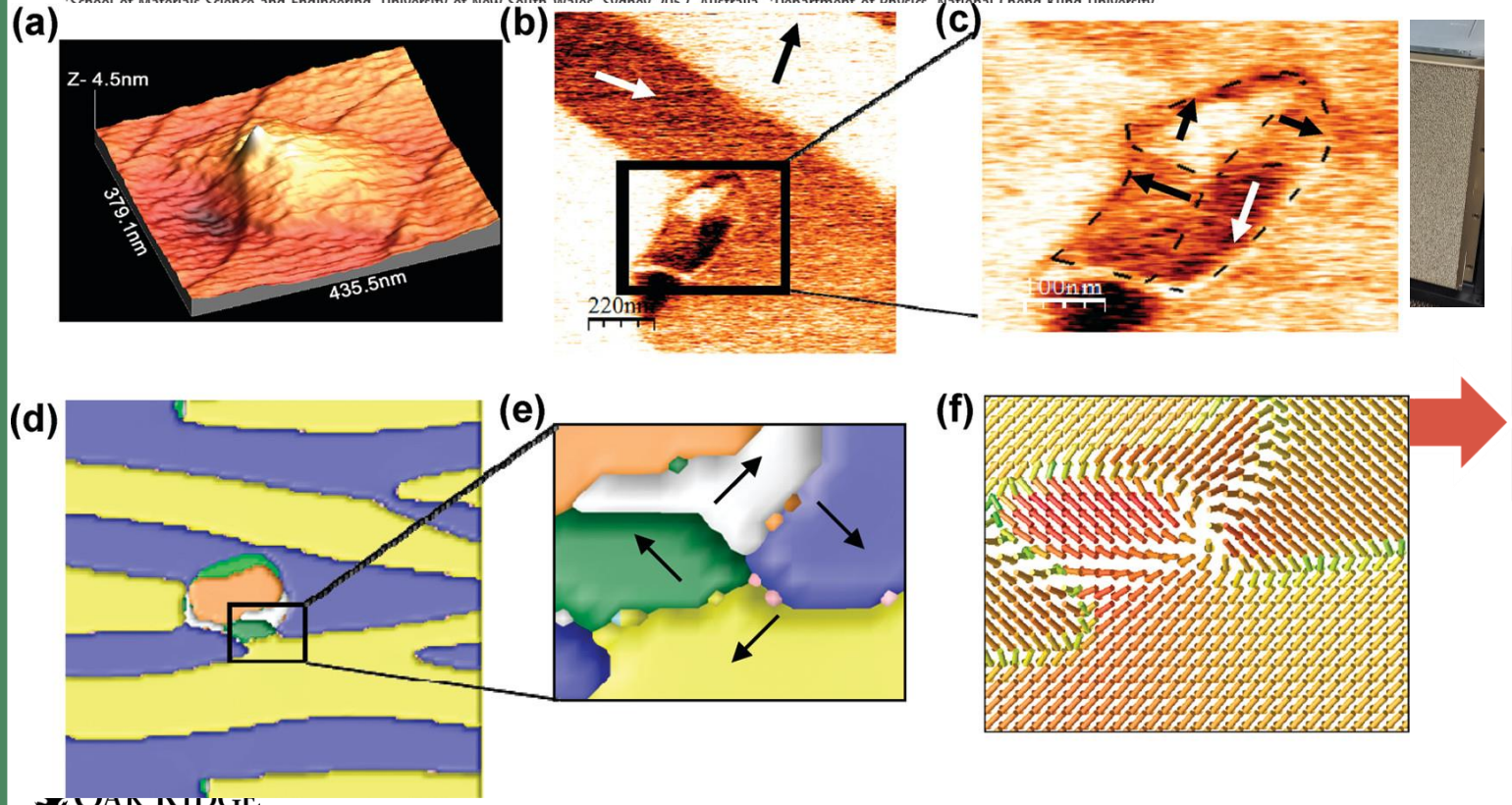
What about experiment?

Experimental workflow

Exploring Topological Defects in Epitaxial $BiFeO_3$ Thin Films

Rama K. Vasudevan,† Yi-Chun Chen,‡ Hsiang-Hua Tai,‡ Nina Balke,§ Pingping Wu,⊥ Saswata Bhattacharya,⊥ L. Q. Chen,⊥ Ying-Hao Chu,† I-Nan Lin,# Sergei V. Kalinin,*,§ and Valanoor Nagarajan†,*

†School of Materials Science and Engineering, University of New South Wales, Sydney 2052, Australia, ‡Department of Physics, National Cheng Kung University



Workflow

Write automation script

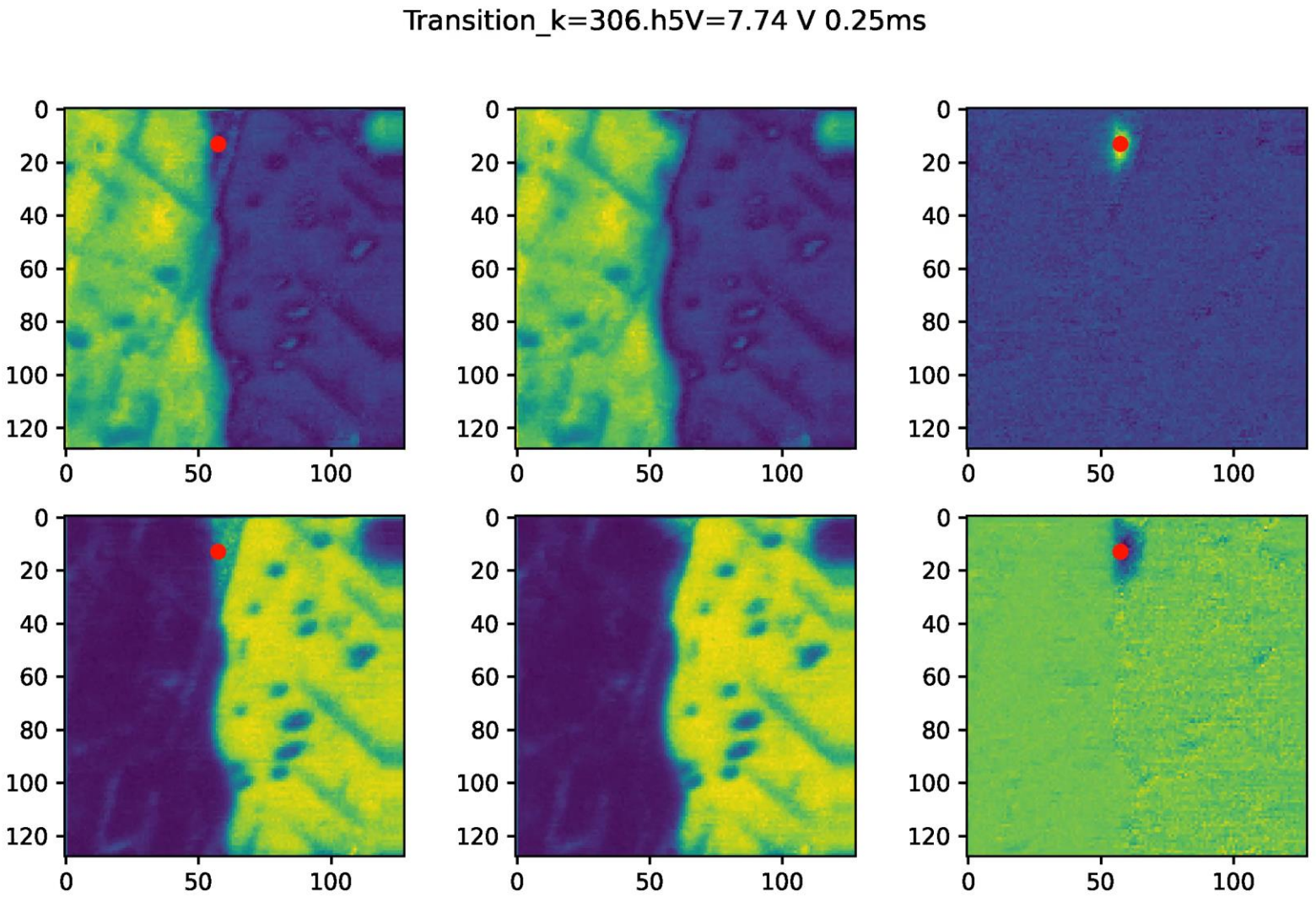
Collect data on state transitions with policy

From data, train a surrogate model

Train RL agent, obtain policy

Update Policy

Wall Manipulation in (110) $PbTiO_3$ thin flims

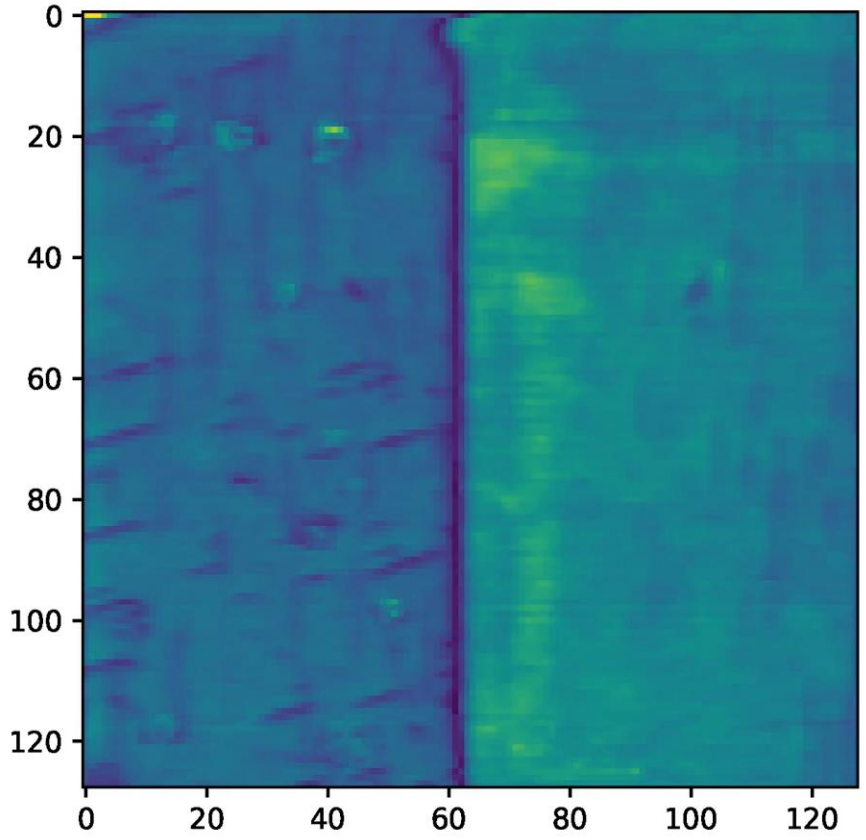


B. Smith et al.
(under review)

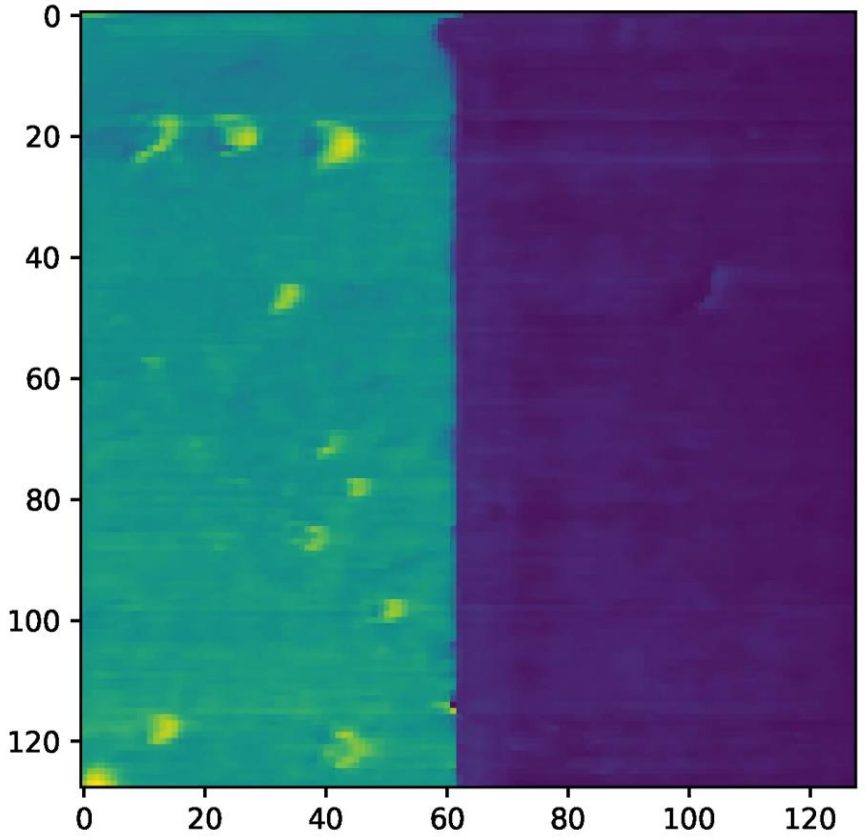
Sample from J. C. Yang (NCKU/Taiwan)

Same experiment, different day

Vertical PFM Amplitude



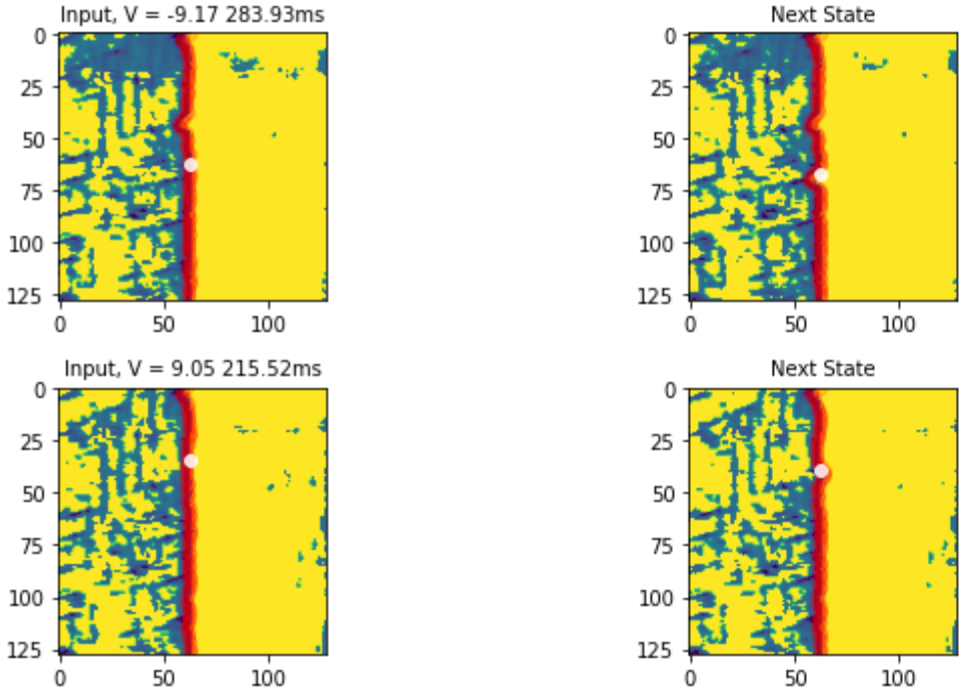
Vertical PFM Phase

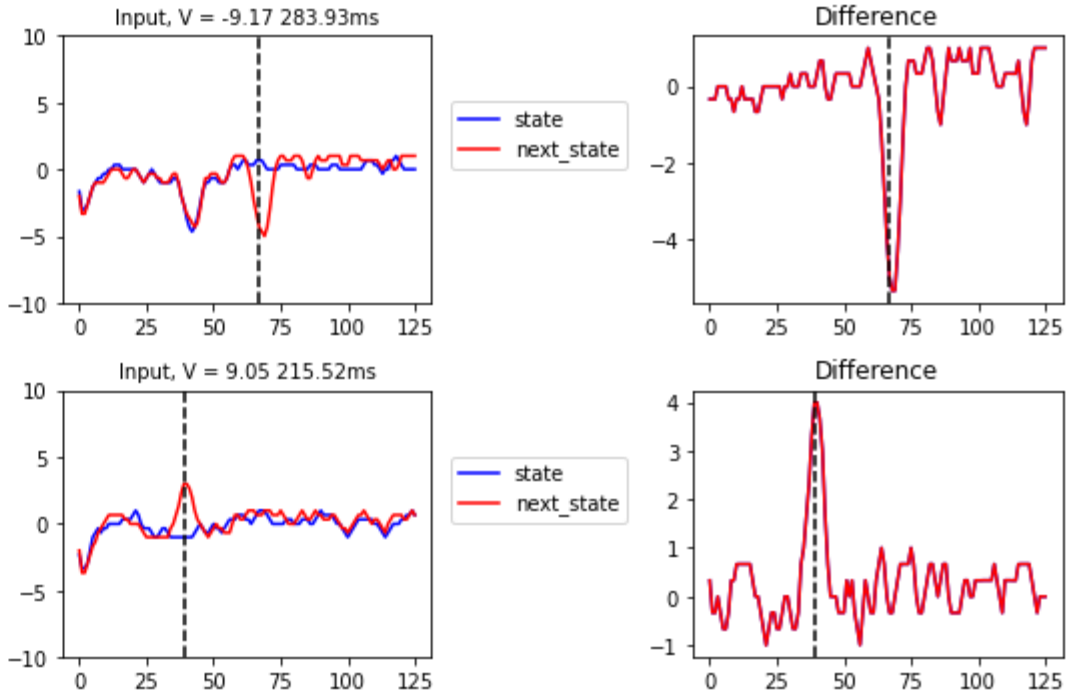


B. Smith et al.
(under review)

Surrogate Model Training

Example of training data





B. Smith et al.
(under review)

Surrogate model: need for physics regularization

Modified Surrogate Model Loss Function

$$\frac{1}{n}\sum(y-\hat{y})^2 + C_{local}(\frac{1}{n}\sum(y-\hat{y})^2) + L_{global}$$

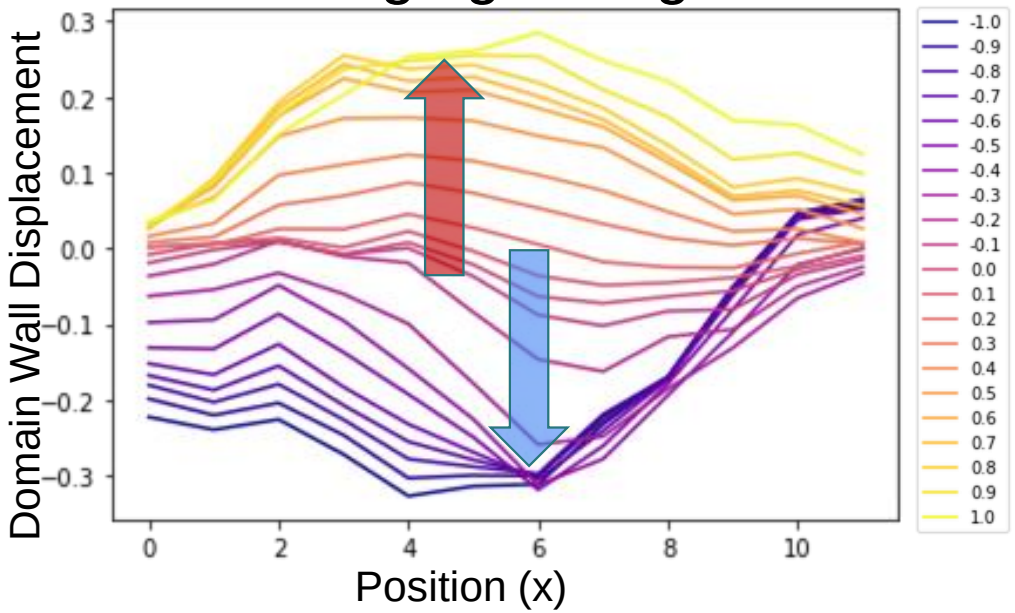
$$C_{local} = \begin{cases} 1, & sign(V) \neq sign(\hat{y}) \\ 0, & otherwise \end{cases}$$

$$L_{global} = \begin{cases} \frac{1}{s}, & \int_0^n f_k(y) < \int_0^n f_{k-1}(y) \\ 0, & otherwise \end{cases}$$

- Wall Displacement should be positive for positive potential (in this geometry)
- Additionally, penalty term for deviating from monotonicity (higher V should not result in a lower absolute displacement than the same position at lower V)

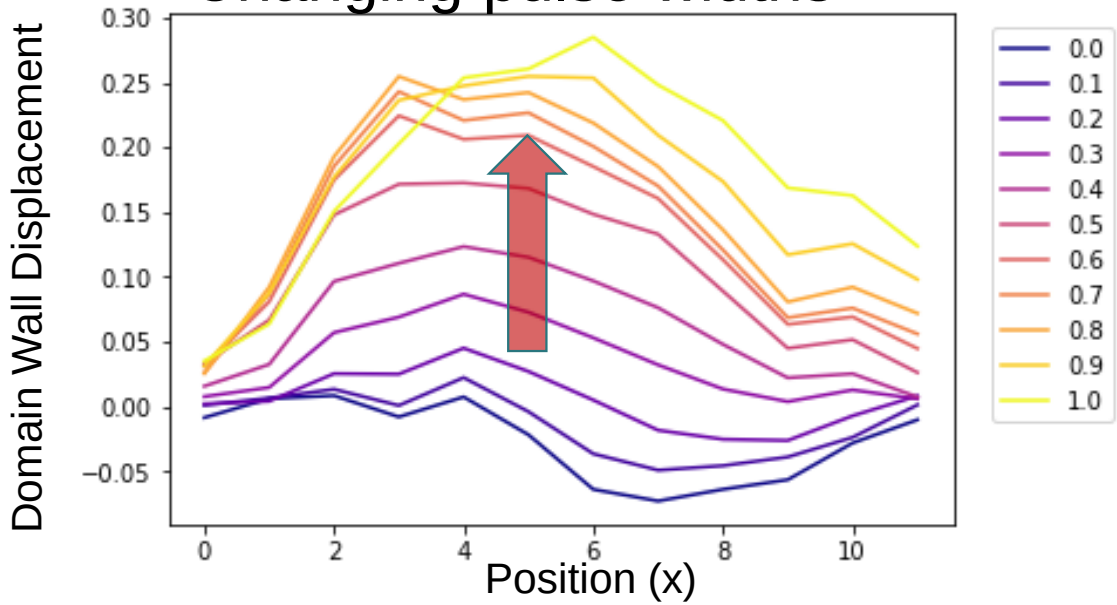
Wall displacement – Surrogate model predictions

Changing voltages



Negative Voltages

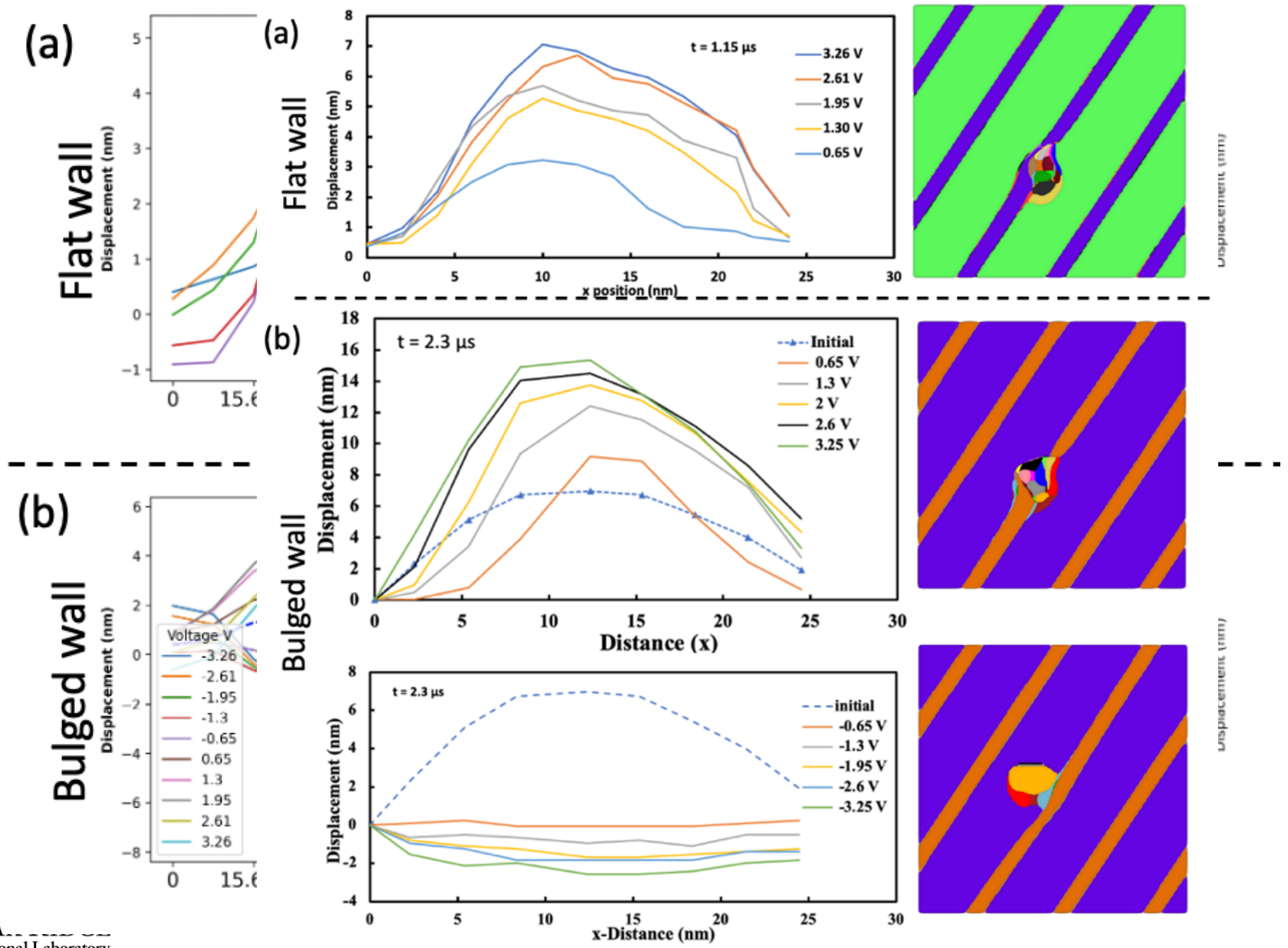
Changing pulse widths



Positive Voltages

B. Smith et al.
(under review)

Wall displacement – Surrogate model predictions



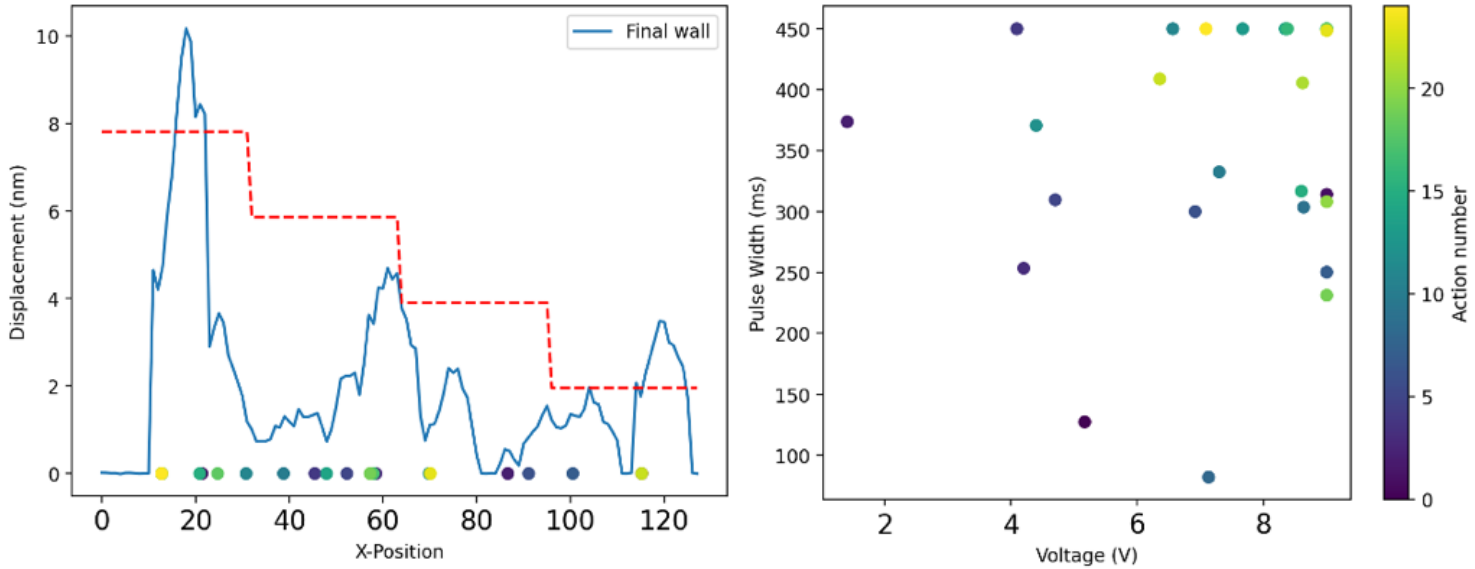
B. Smith et al. (under review)

Major change in the ‘phase diagram’ of the domain wall based on initial structure

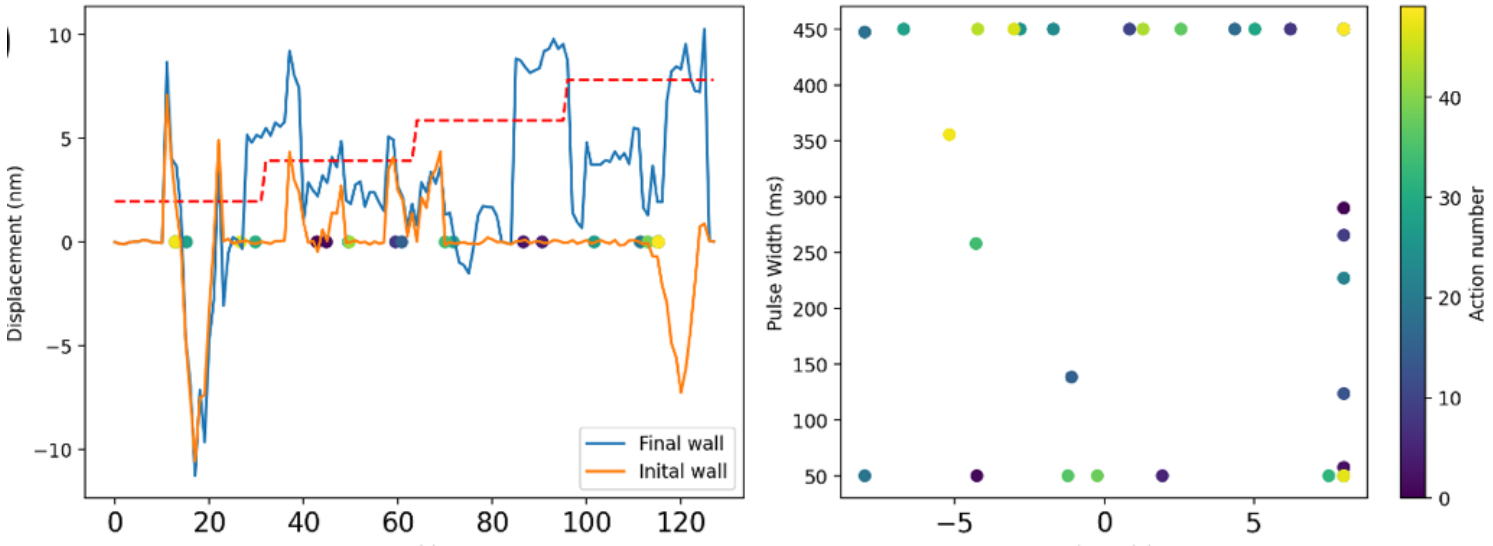
Backed up by phase field simulations (B. Pant, Y. Cao, UT Arlington)

Reinforcement Learning: Autonomous Wall manipulation

Target structure 1



Target structure 2



B. Smith et al. (under review)