Analyzing and Improving Fault Tolerance of Learning-Based Navigation Systems

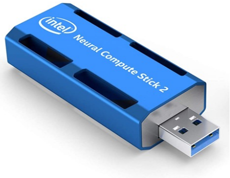
Zishen Wan[1], Aqeel Anwar[1], Yu-Shun Hsiao[2], Tianyu Jia[2]
Vijay Janapa Reddi[2], Arijit Raychowdhury[1]

[1]Georgia Institute of Technology, Atlanta, GA, [2]Harvard University, Cambridge, MA

✉ zishenwan@gatech.edu

Dec. 5-9, 2021
San Francisco, CA, USA

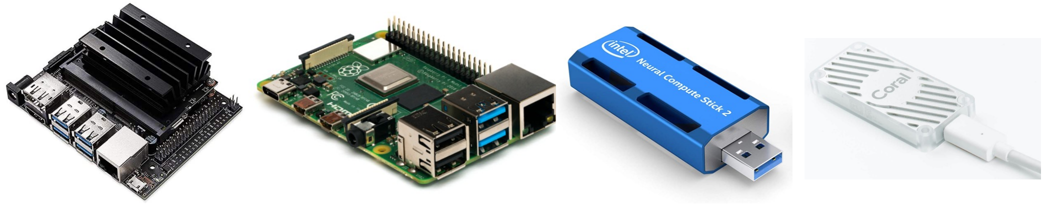
Safety of Autonomous Navigation



- End-to-end learning-based autonomous navigation system
- Specialized hardware accelerator
- Hardware Fault
  - Transient fault
  - Permanent fault
- Traditional protection method
  - Hardware module redundancy

Safety of Autonomous Navigation

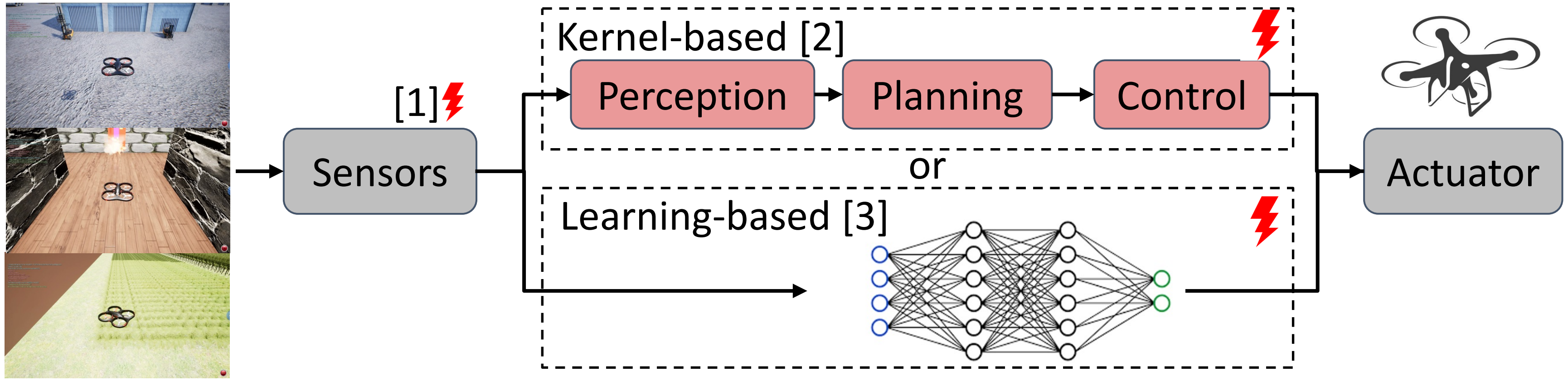
How is resilience of learning-based navigation system to hardware faults?
How do we detect and mitigate hardware faults?



- Transient fault
- Permanent fault

- Traditional protection method
  - Hardware module redundancy

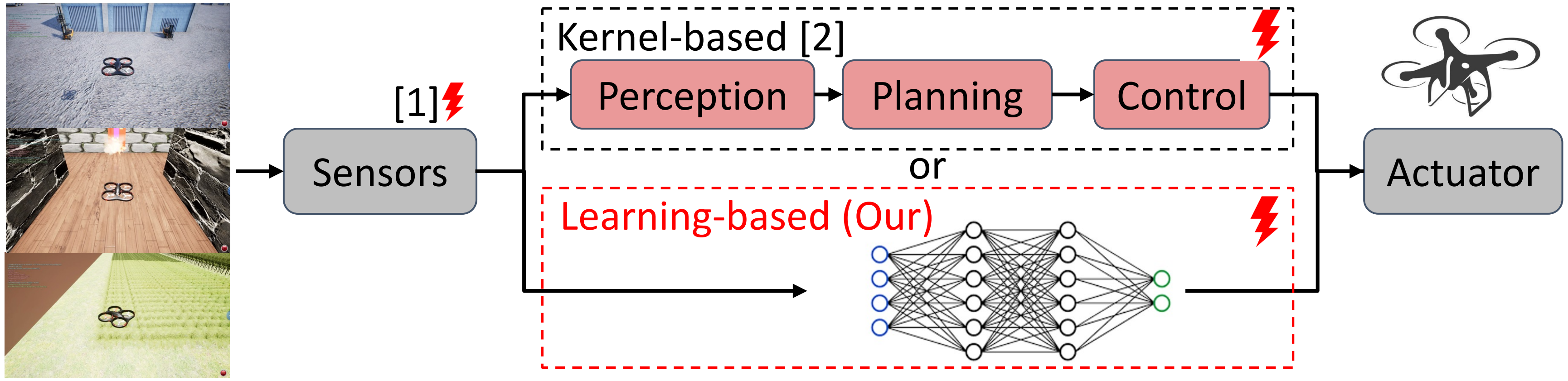
Related Work



[1] A. Toschi et al., NPC'19
[2] Y. Hsiao*, Z. Wan* et al., arXiv'21

Related Work

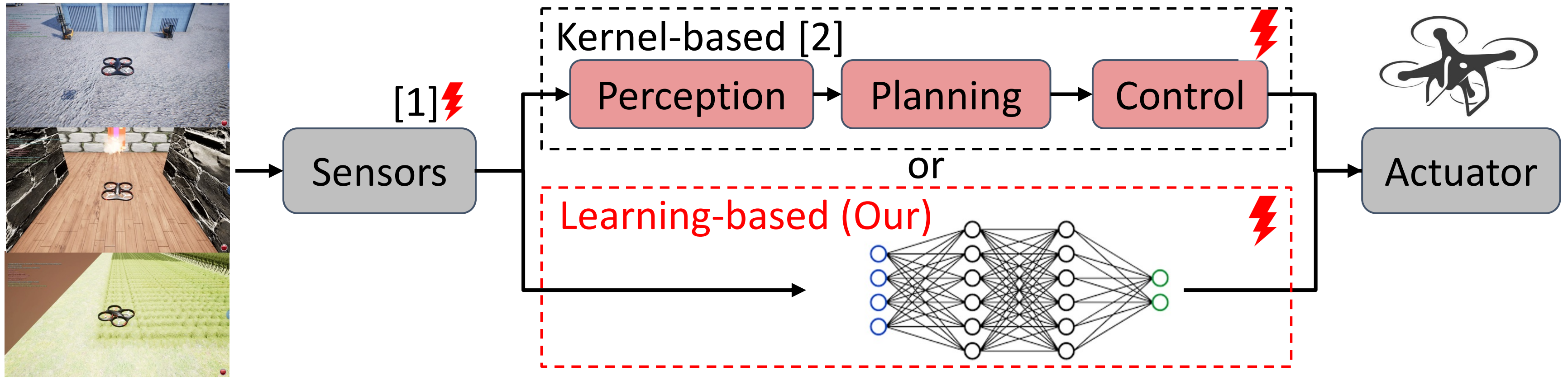
- Reliability of autonomous systems



[1] A. Toschi et al., NPC'19
[2] Y. Hsiao*, Z. Wan* et al., arXiv'21

Related Work

- Reliability of autonomous systems

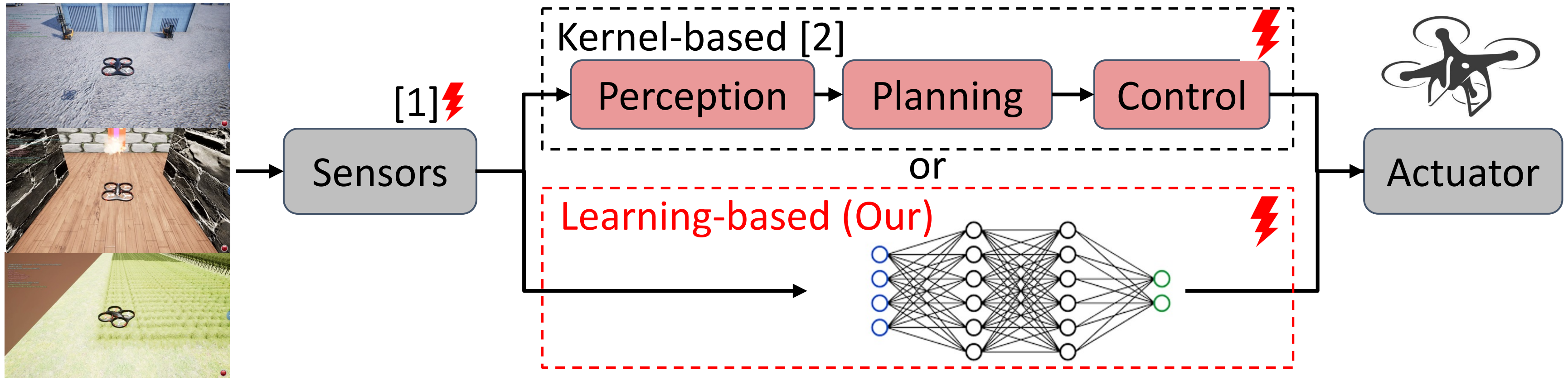


- Fault characterization
  - Neural network in supervised learning: PytorchFI[3], Ares[4], SC'17[5]
  - End-to-end reinforcement learning-based (Our)

[1] A. Toschi et al., NPC'19
[2] Y. Hsiao*, Z. Wan* et al., arXiv'21
[3] A. Mahmoudetal et al., DSN'20
[4] B. Reagen et al., DAC'18
[5] G. Li et al., SC'17

Related Work

- Reliability of autonomous systems



- Fault characterization
  - Neural network in supervised learning: PytorchFI[3], Ares[4], SC'17[5]
  - End-to-end reinforcement learning-based (Our)
- Fault mitigation
  - Hardware redundancy-based method: DMR, TMR
  - Application-aware method (Our)

[1] A. Toschi et al., NPC'19
[2] Y. Hsiao*, Z. Wan* et al., arXiv'21
[3] A. Mahmoudetal et al., DSN'20
[4] B. Reagen et al., DAC'18
[5] G. Li et al., SC'17

This work

Analyzing and Improving fault tolerance of learning-based navigation systems, that is:

- A fault injection tool-chain for learning-based systems
- Hardware fault study in learning-based systems
- Fault mitigation techniques for learning-based systems

This work

Analyzing and Improving fault tolerance of learning-based navigation systems, that is:

- A fault injection tool-chain for learning-based systems
- Hardware fault study in learning-based systems
- Fault mitigation techniques for learning-based systems

Fault Model and Fault Injection

- Fault Type
  - Transient fault
    - Random bit-flip
  - Permanent fault
    - Stuck-at-0
    - Stuck-at-1

Fault Model and Fault Injection

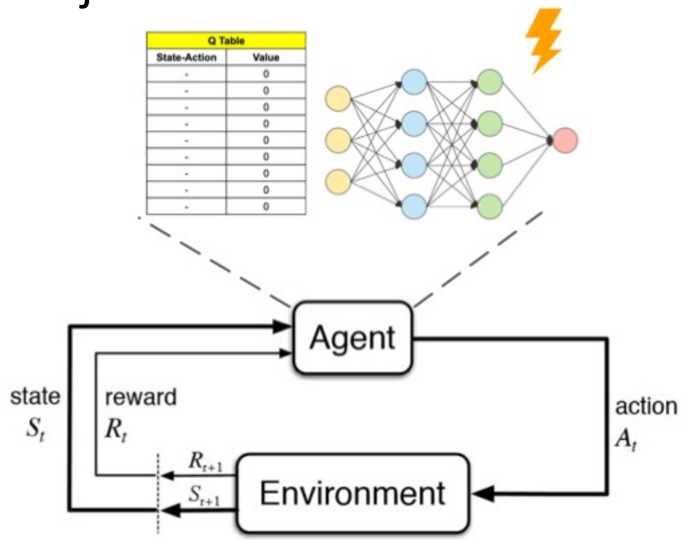
- Fault Type
  - Transient fault
    - Random bit-flip
  - Permanent fault
    - Stuck-at-0
    - Stuck-at-1
- Fault Location
  - Memory [1,2,3]

[1] B. Reagen et al., DAC'18
[2] G. Li et al., SC'17
[3] P. N. Whatmough et al., ISSCC'17

Fault Model and Fault Injection

- Fault Type
  - Transient fault
    - Random bit-flip
  - Permanent fault
    - Stuck-at-0
    - Stuck-at-1
- Fault Location
  - Memory [1,2,3]

- Fault Injection
  - Methodology
    - Static injection
    - Dynamic injection



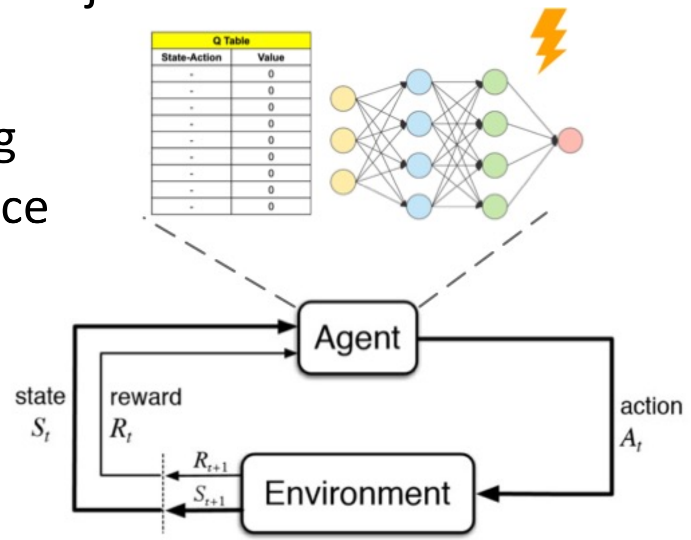
[1] B. Reagen et al., DAC'18
[2] G. Li et al., SC'17
[3] P. N. Whatmough et al., ISSCC'17

Fault Model and Fault Injection

- Fault Type
  - Transient fault
    - Random bit-flip
  - Permanent fault
    - Stuck-at-0
    - Stuck-at-1
- Fault Location
  - Memory [1,2,3]

[1] B. Reagen et al., DAC'18
[2] G. Li et al., SC'17
[3] P. N. Whatmough et al., ISSCC'17

- Fault Injection
  - Methodology
    - Static injection
    - Dynamic injection
  - Phases
    - Training
    - Inference

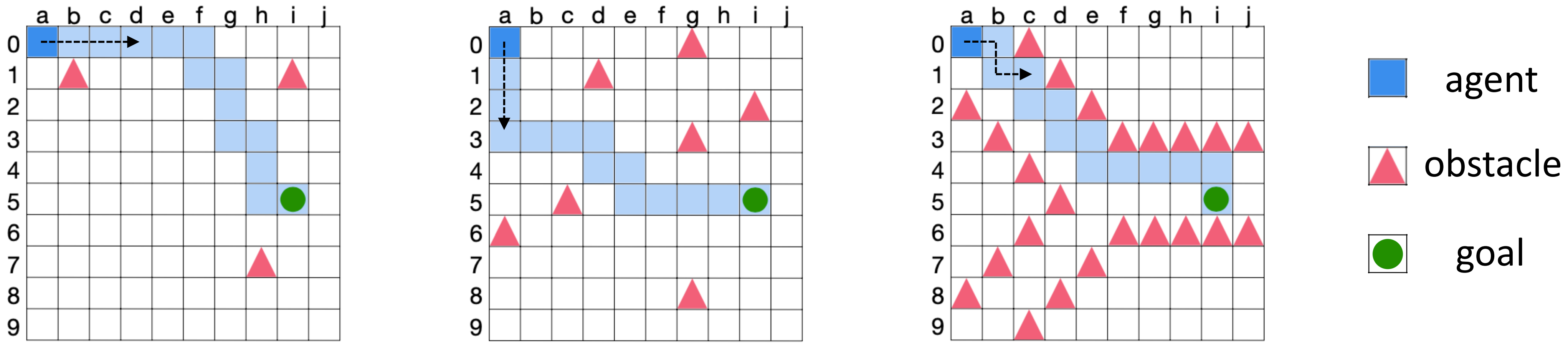


This work

Analyzing and Improving fault tolerance of learning-based navigation systems, that is:

- A fault injection tool-chain for learning-based systems
- Hardware fault study in learning-based systems
- Fault mitigation techniques for learning-based systems

Grid-Based Navigation Problem

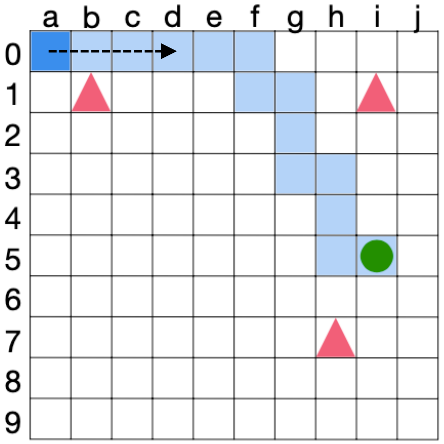


Low obstacle density

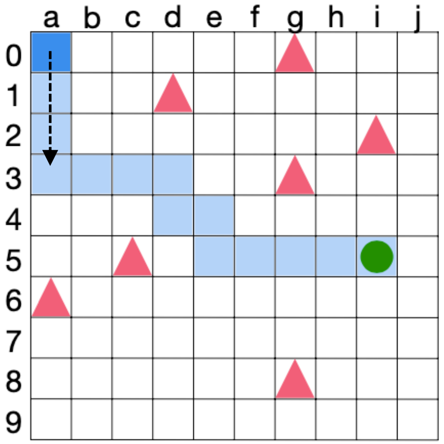
Middle obstacle density

High obstacle density

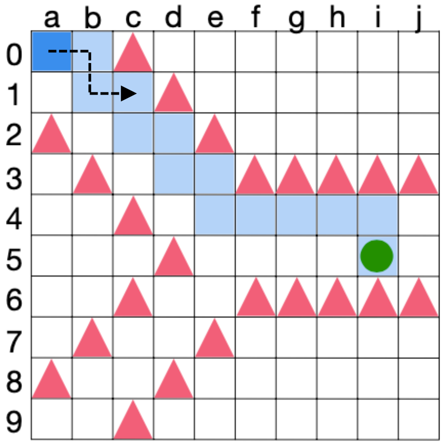
Grid-Based Navigation Problem



Low obstacle density



Middle obstacle density

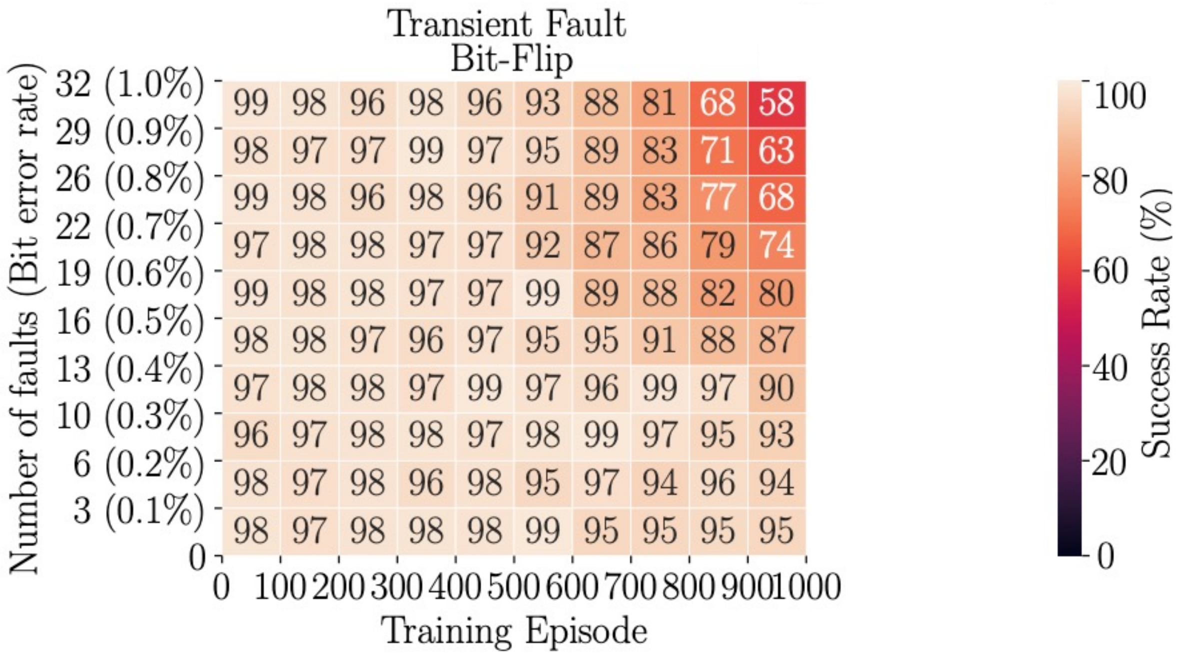


High obstacle density

- Algorithm paradigm: NN-based method, Tabular-based method
- Evaluation metric: agent's success rate

Faults in Grid World (Training)

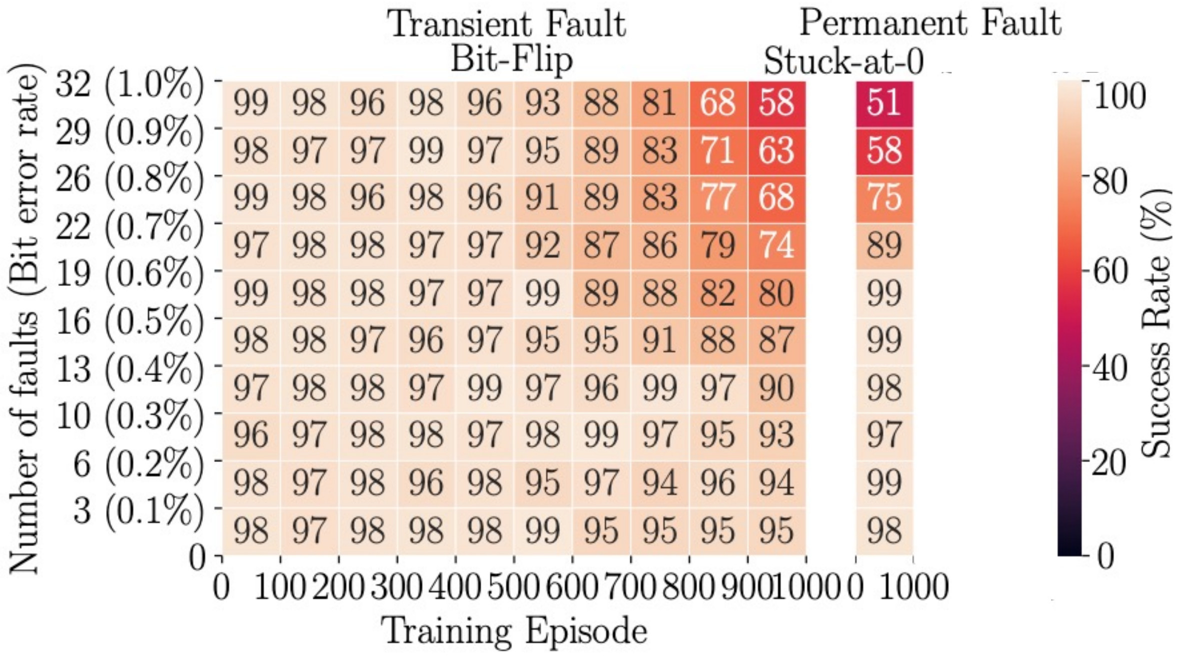
NN-based method: (The darker, the worse)



➢ Transient fault occurred in later episodes with high BER has higher impact.

Faults in Grid World (Training)

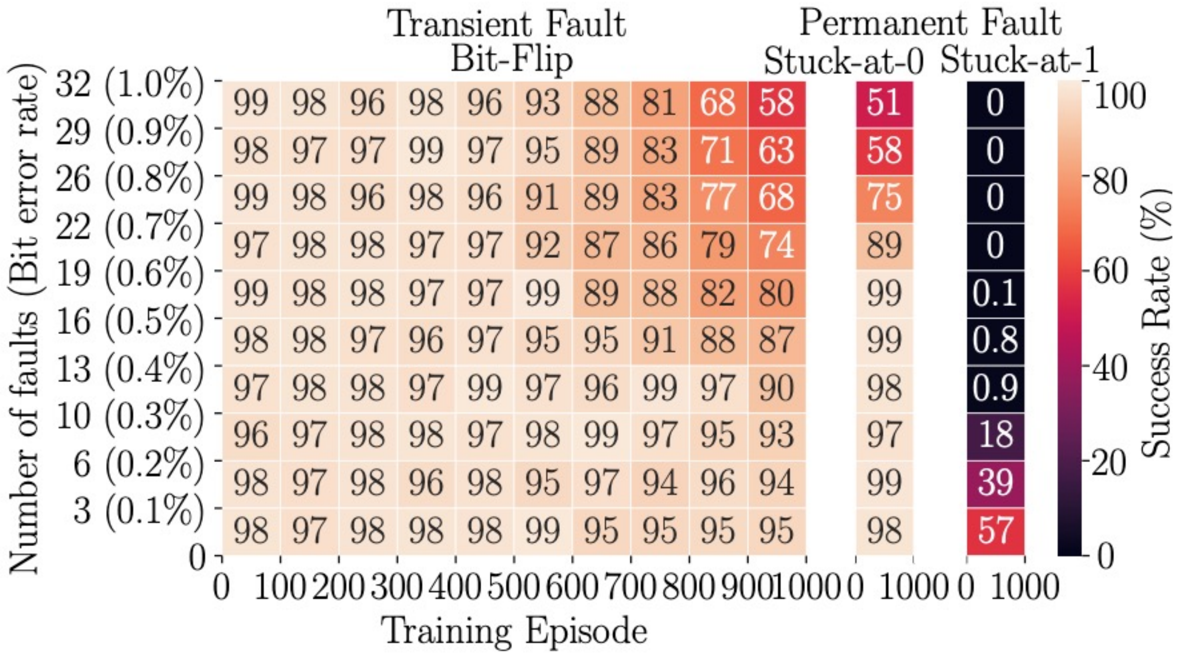
NN-based method: (The darker, the worse)



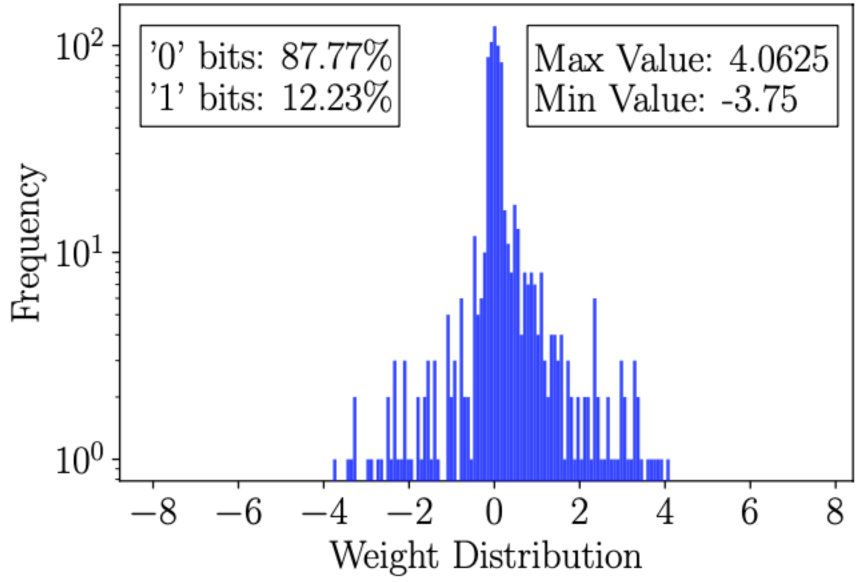
- ➢ Permanent fault stuck-at-0 has comparable impact as transient fault.

Faults in Grid World (Training)

NN-based method: (The darker, the worse)



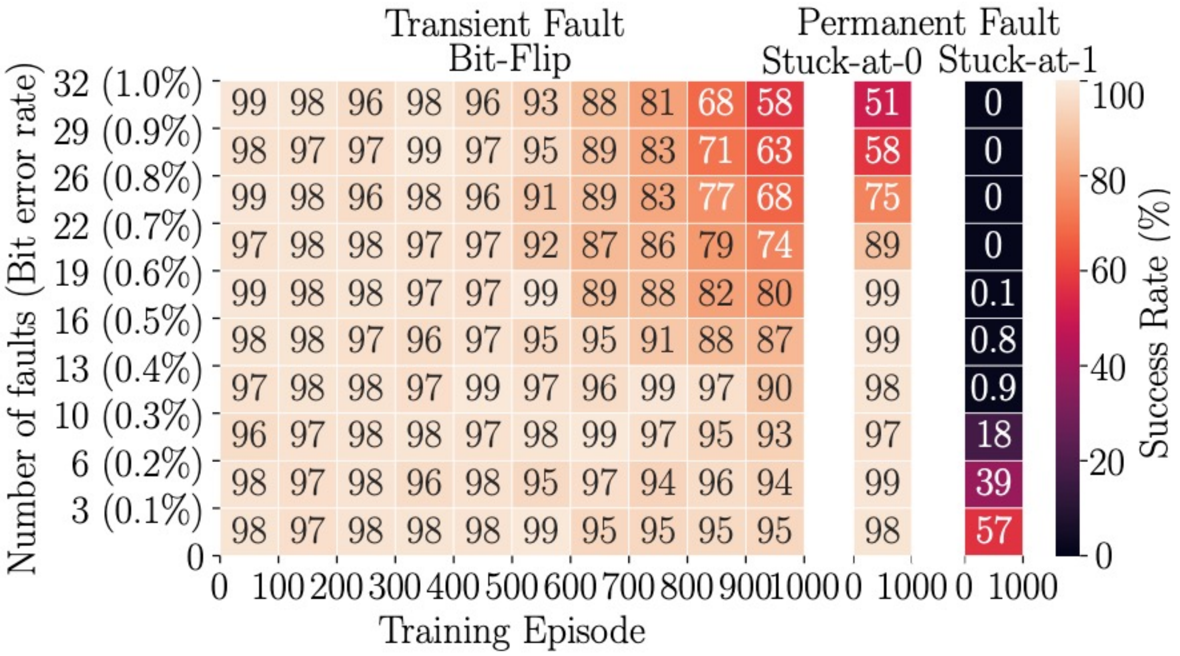
NN-based policy weight distribution:



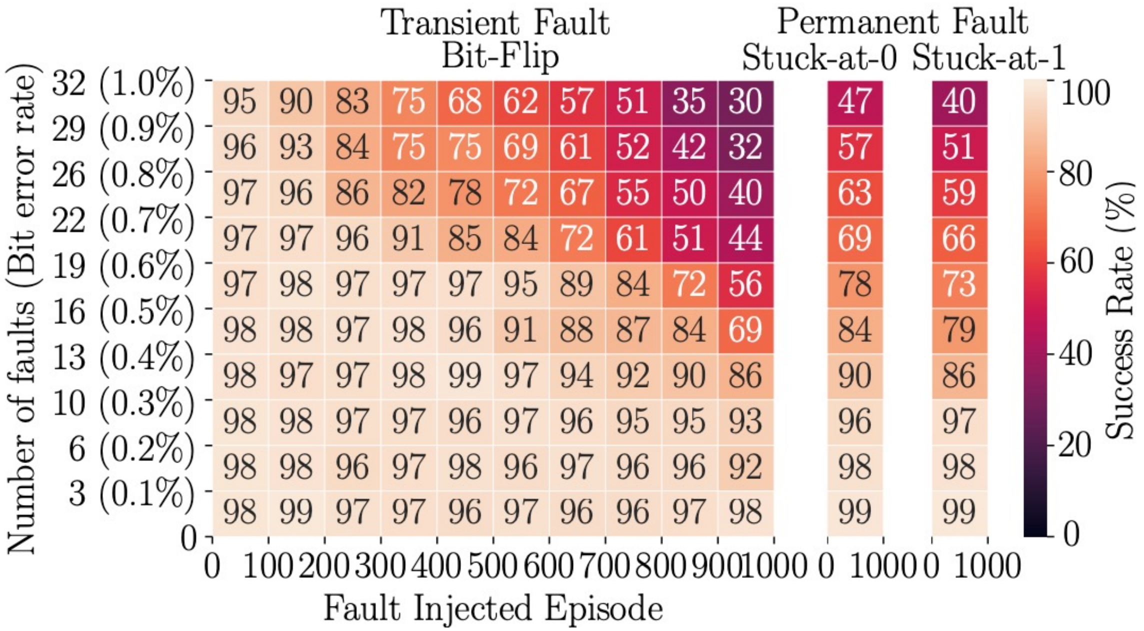
- ➢ Permanent fault stuck-at-1 has much severer impact than stuck-at-0.

Faults in Grid World (Training)

NN-based method: (The darker, the worse)



Tabular-based method:

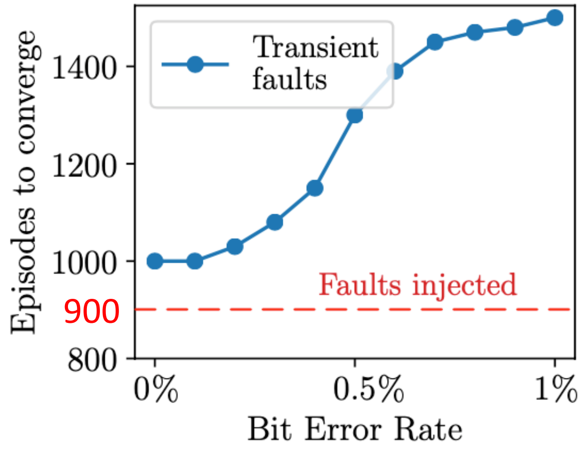


➢ NN-based policy exhibit higher resilience than Tabular-based policy (except stuck-at-1).

Faults in Grid World (Convergence)

NN-based method

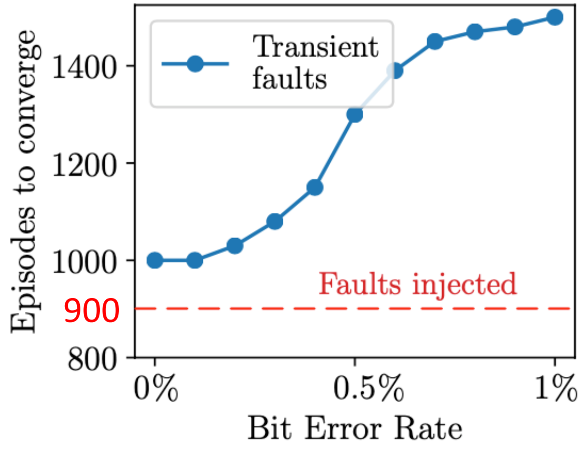
Transient fault

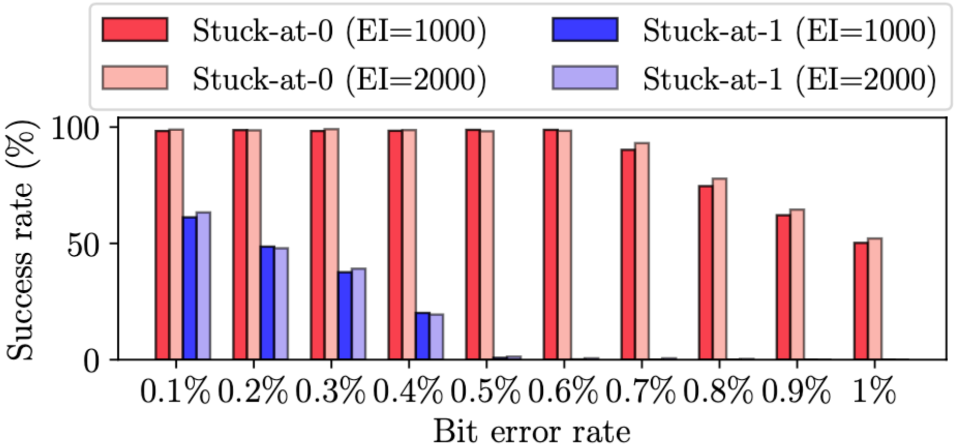


- System can finally achieve convergence (>95% success rate) after transient faults injected.

Faults in Grid World (Convergence)

	NN-based method
Transient fault	
Permanent fault	





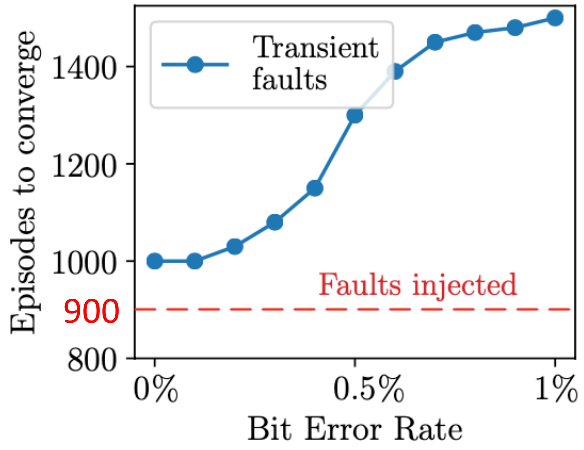
- System can finally achieve convergence (>95% success rate) after transient faults injected.
- Extra training time doesn't bring obvious improvements under permanent faults.

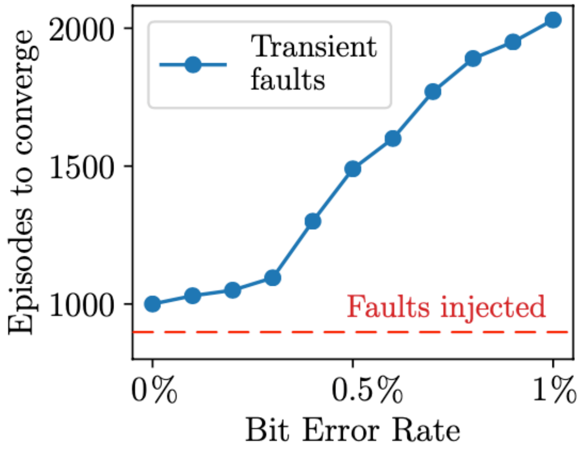
Faults in Grid World (Convergence)

NN-based method

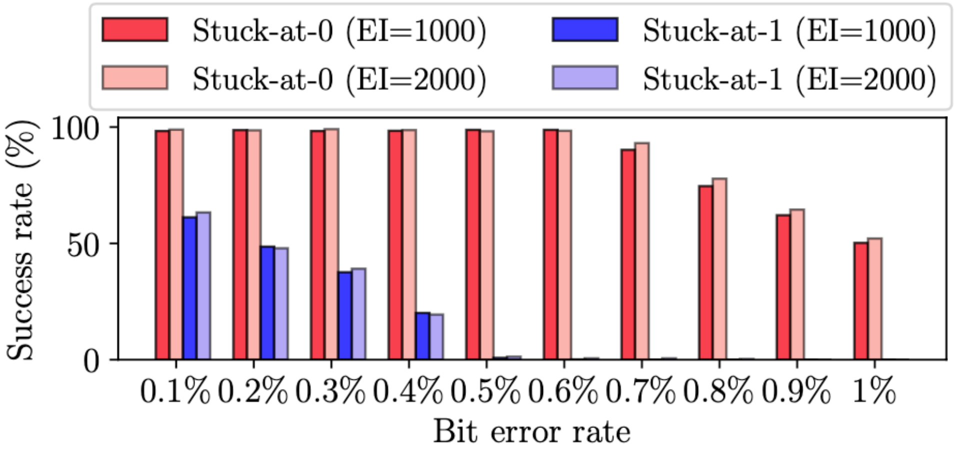
Tabular-based method

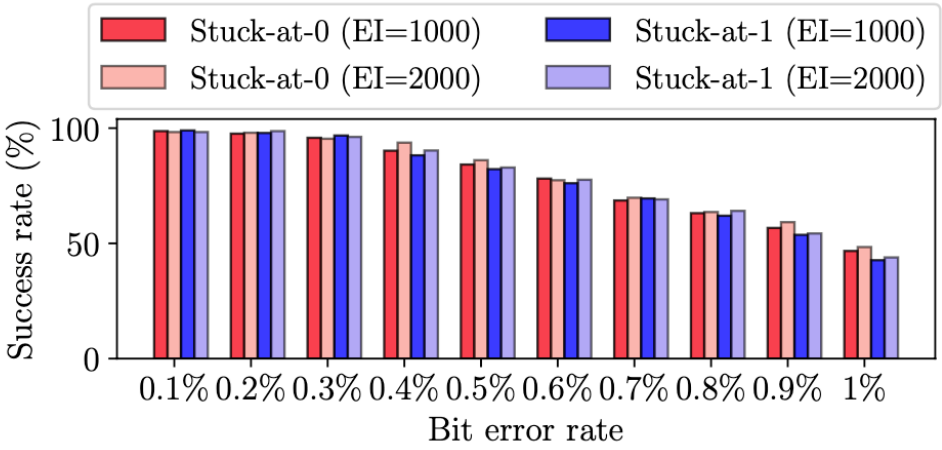
Transient fault





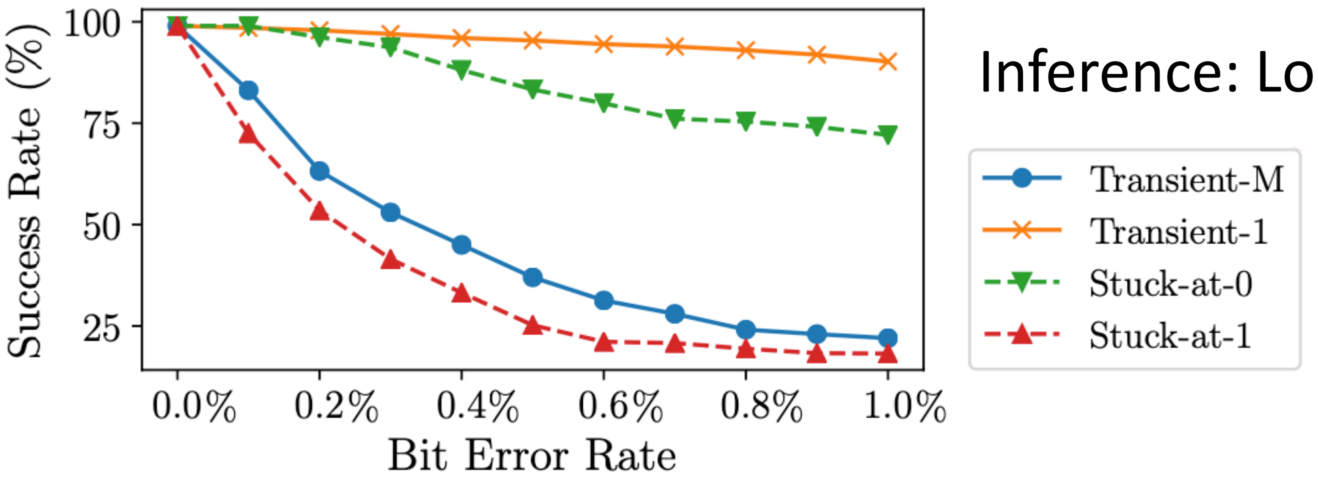
Permanent fault





Faults in Grid World (Inference)

NN-based method:



Inference: Long-term decision-making process

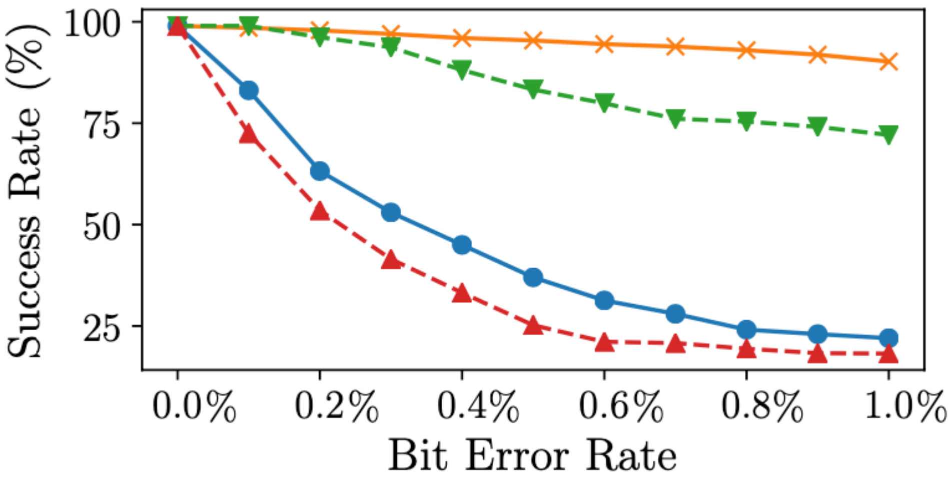
Transient-M: impact all steps

Transient-1: impact single step

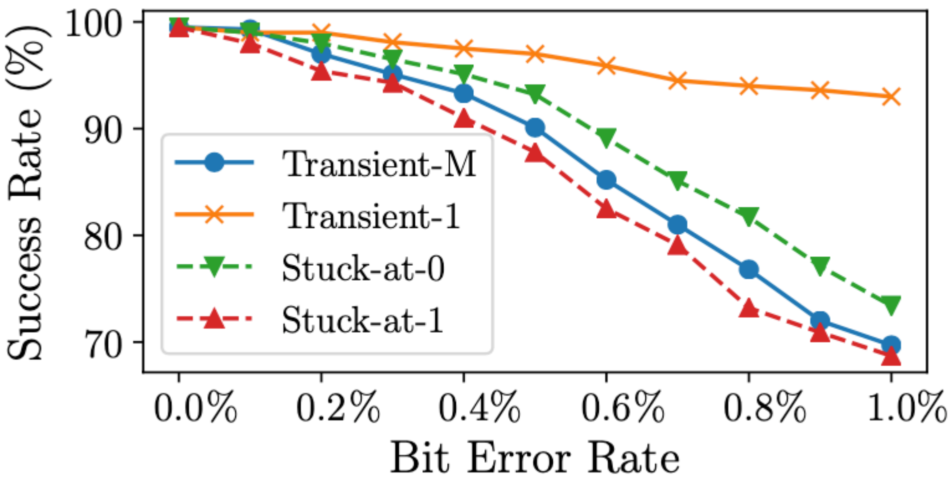
- ➢ Transient fault: Transient-1 has a negligible effect compared to Transient-M.
- ➢ Permanent fault: Stuck-at-1 has a much severe impact on policy than Stuck- at-0

Faults in Grid World (Inference)

NN-based method:



Tabular-based method:



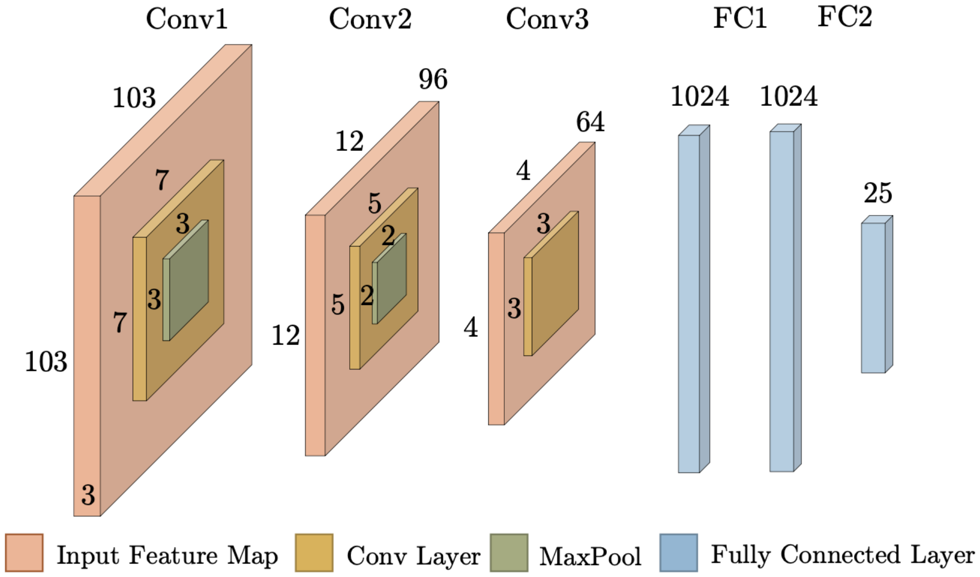
- Transient fault: Transient-1 has a negligible effect compared to Transient-M.
- Permanent fault: Stuck-at-1 has a much severe impact on policy than Stuck- at-0

Drone Autonomous Navigation Problem

Environments and demos:

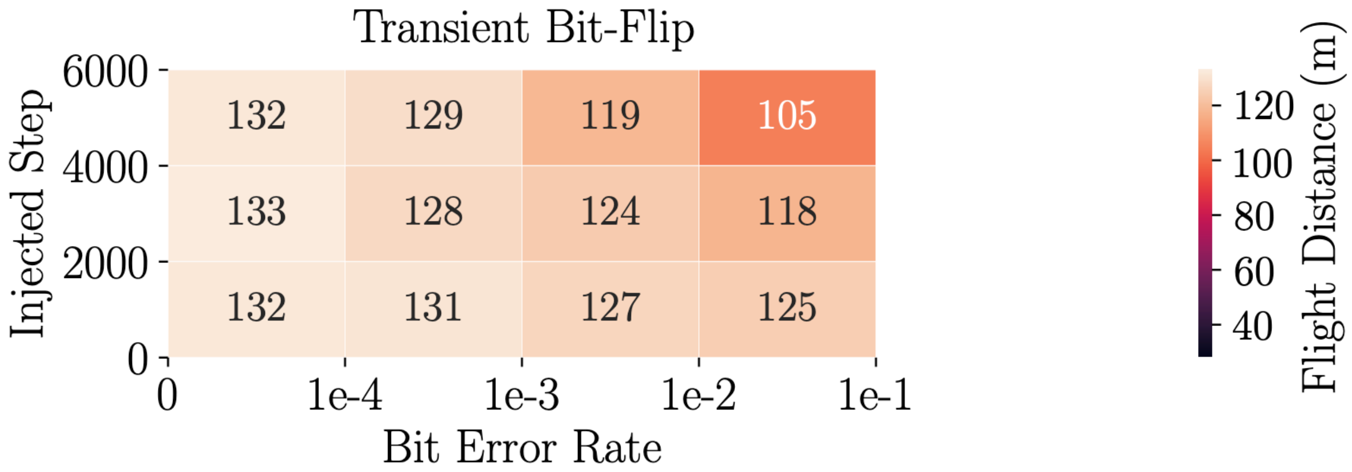
(PEDRA: Powered by Unreal Engine and AirSim)

Policy architecture:



➢ Evaluation metric: drone safe flight distance (the longer, the better).

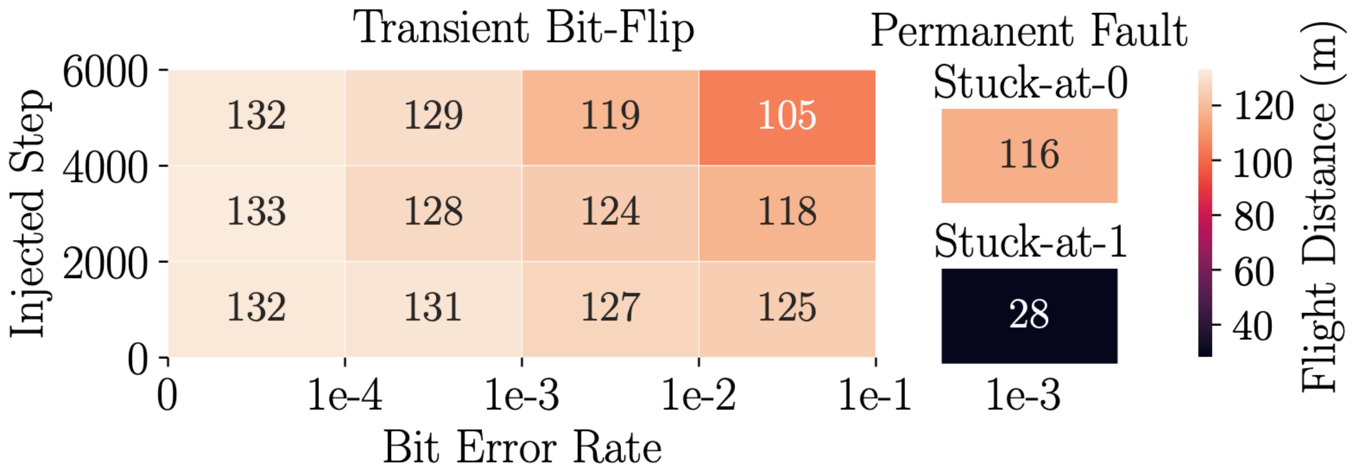
Faults in Drone Navigation (Training)



Higher (lighter) is better

- Training method: offline training -> online fine-tunning using transfer learning
- Transient fault: occurred at latter episodes with higher BER impact flight quality more.

Faults in Drone Navigation (Training)



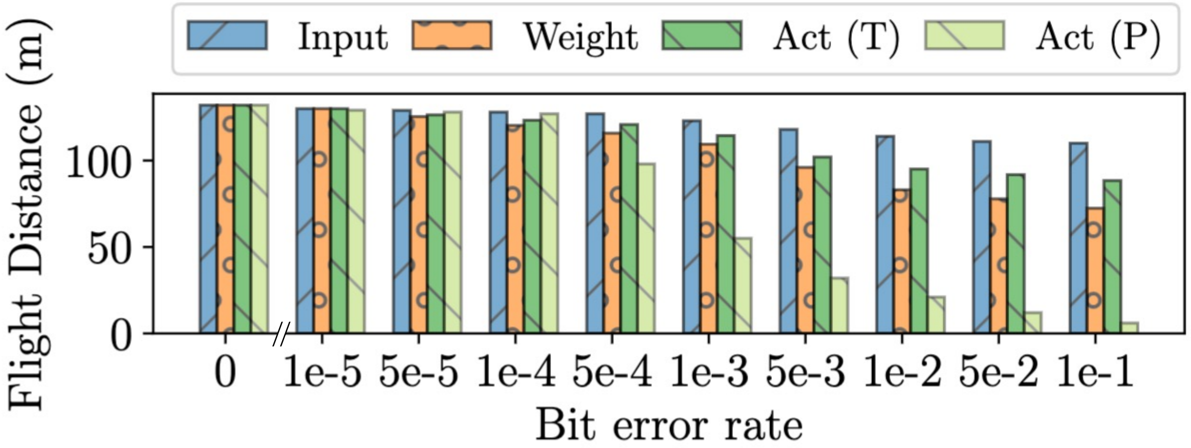
Higher (lighter) is better

- Training method: offline training -> online fine-tunning using transfer learning
- Transient fault: occurred at latter episodes with higher BER impact flight quality more.
- Permanent fault: stuck-at-1 has much severe impact than stuck-at-0

Faults in Drone Navigation (Inference)

Different data locations:

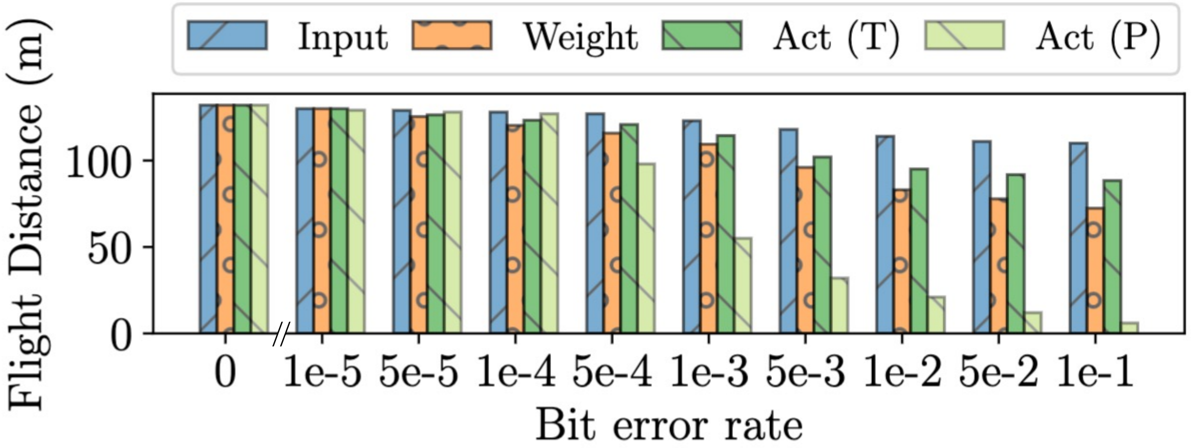
(the higher, the better)



- Weights are sensitive to transient faults while input buffer is resilient.

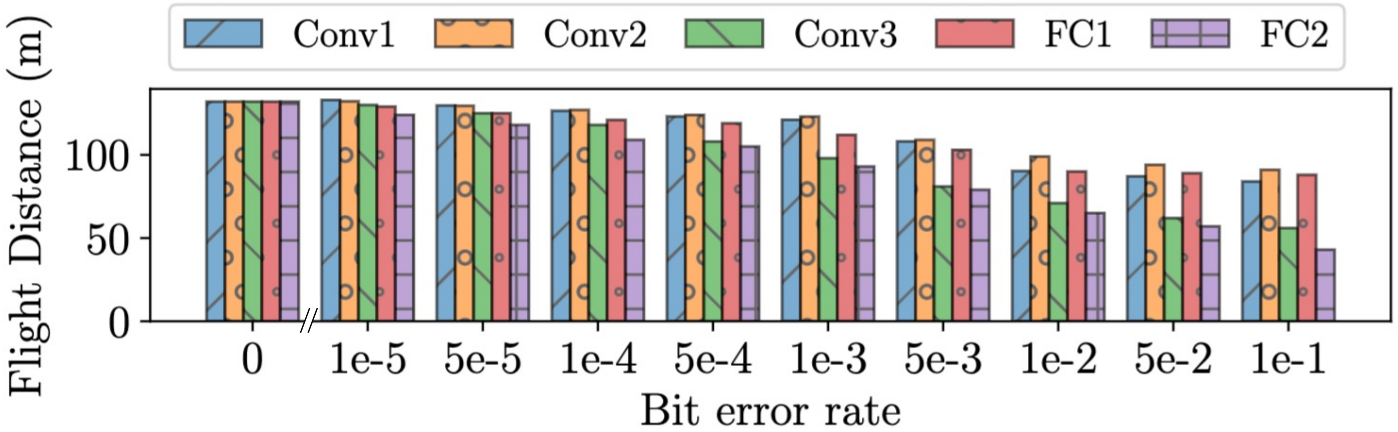
Faults in Drone Navigation (Inference)

Different
data locations:
(the higher, the better)



- Weights are sensitive to transient faults while input buffer is resilient.

Different
NN layers:
(the higher, the better)

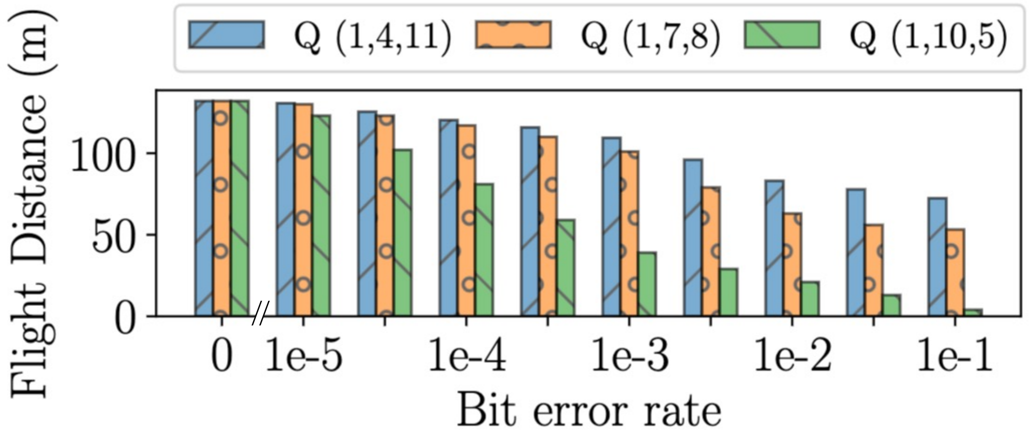


- Conv3: no followed pooling layer
- FC2: directly dictates the drone actions

Faults in Drone Navigation (Inference)

Different
data types:
(the higher, the better)

Fixed-point datatype: Q (sign, integer, fraction)

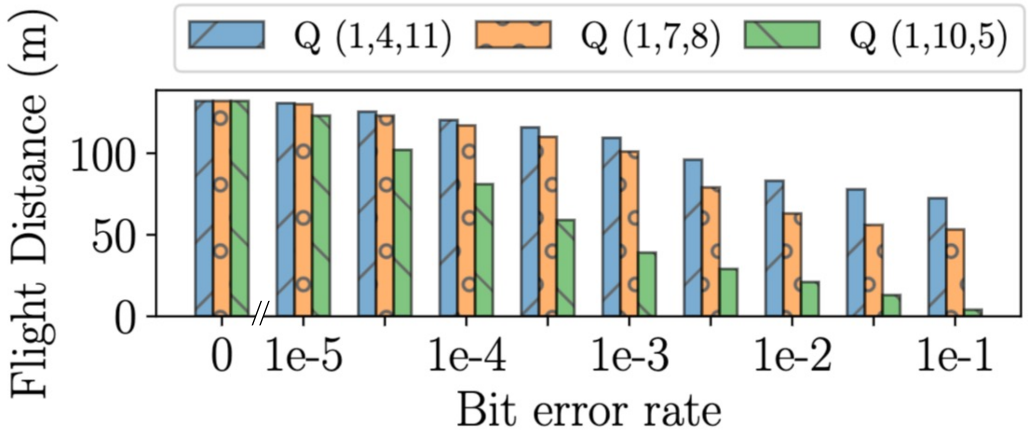


- Data types should optimally capture the value range rather than pursuing an unnecessarily large range

Faults in Drone Navigation (Inference)

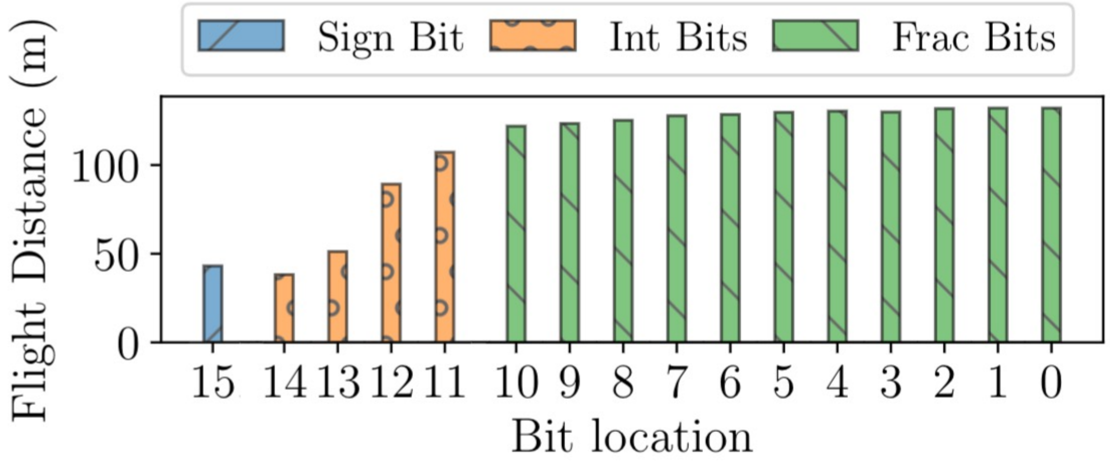
Different data types:
(the higher, the better)

Fixed-point datatype: Q (sign, integer, fraction)



- Data types should optimally capture the value range rather than pursuing an unnecessarily large range

Different bit locations in Q (1,4,11):
(the higher, the better)



- Only sign and high-order integer bits are vulnerable

This work

Analyzing and Improving fault tolerance of learning-based navigation systems, that is:

- A fault injection tool-chain for learning-based systems
- Hardware fault study in learning-based systems
- Fault mitigation techniques for learning-based systems

Training: Adaptive Exploration Rate Adjustment

- Detection: change in cumulative reward
- Recovery: dynamically adjust exploration-to-exploitation ratio and speed

Training: Adaptive Exploration Rate Adjustment

- Detection: change in cumulative reward
- Recovery: dynamically adjust exploration-to-exploitation ratio and speed

	Detection
Transient fault	Reward drop exceeds *x*% within *y* continuous episodes
Permanent fault	Reward is still low after going to steady-exploitation states

Training: Adaptive Exploration Rate Adjustment

- Detection: change in cumulative reward
- Recovery: dynamically adjust exploration-to-exploitation ratio and speed

	Detection		Recovery
Transient fault	Reward drop exceeds *x*% within *y* continuous episodes	→	Increase exploration rate (ER)
Permanent fault	Reward is still low after going to steady-exploitation states	→	Revert the exploration rate to initial and slow down its decreasing speed by $2^n\times$

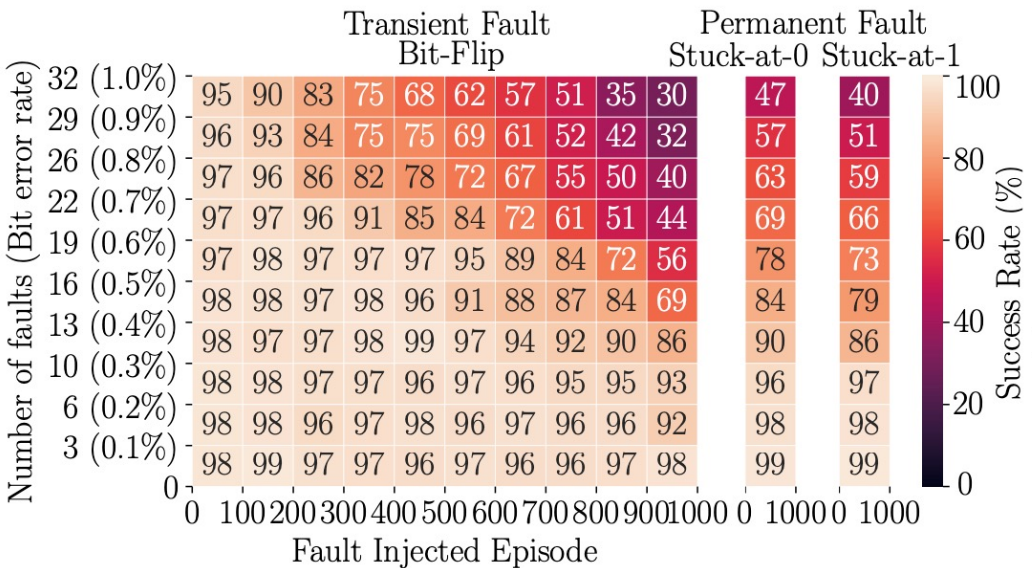
f(r): reward drop
f(t): fault occurrence time

$$ER_{new} = ER_{old} + \delta(ER) = ER_{old} + \alpha \times min(f(r), f(r)f(t))$$

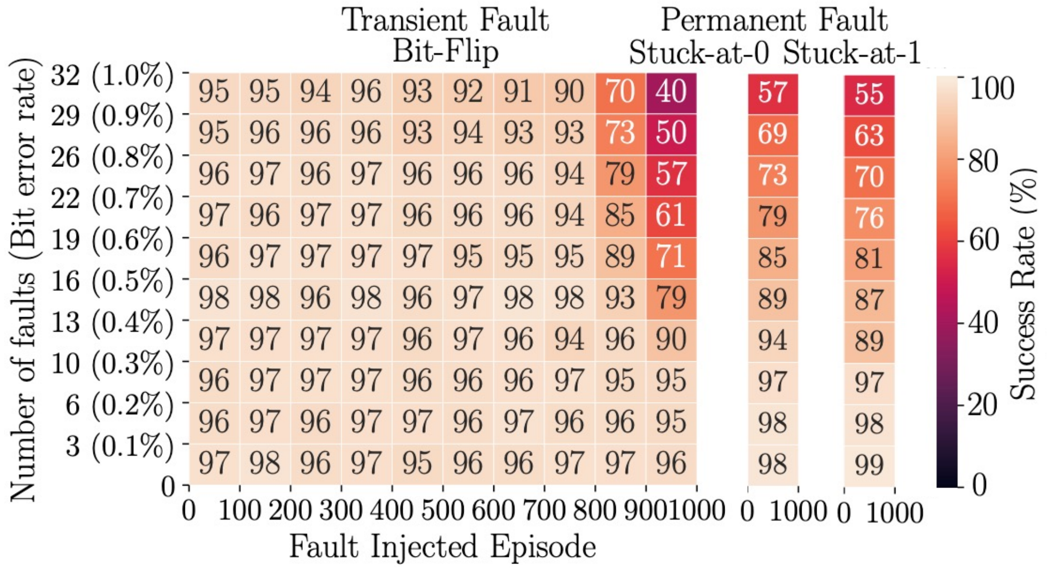
Training: Adaptive Exploration Rate Adjustment

- Evaluation:

Before fault mitigation:



After fault mitigation:



➢ The impact of both transient fault and permanent fault during training can be relieved.

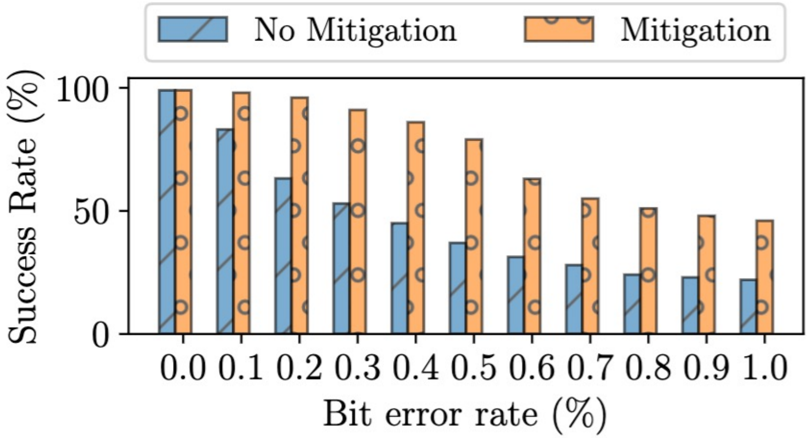
Inference: Value Range-Based Anomaly Detection

- Detection: statistically anomaly detection, (a_i, b_i) -> $(1.1a_i, 1.1b_i)$
- Recovery: skip faulty operations

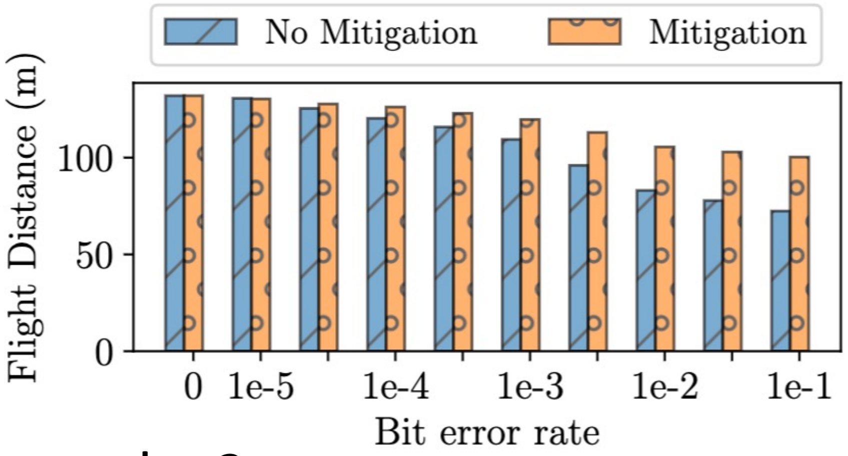
Inference: Value Range-Based Anomaly Detection

- Detection: statistically anomaly detection, (a_i, b_i) -> $(1.1a_i, 1.1b_i)$
- Recovery: skip faulty operations
- Evaluation:

Grid World navigation



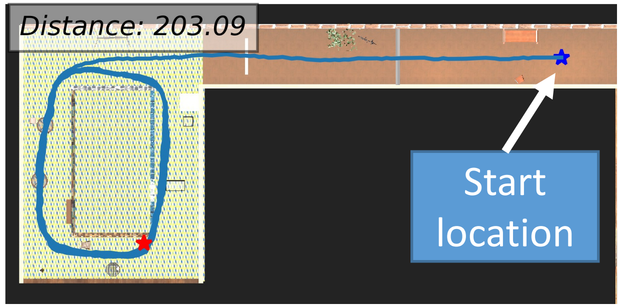
Drone autonomous navigation



- ➢ Grid World: agent's success rate increase by 2x
- ➢ Drone autonomous navigation: safe flight distance increases by 39%

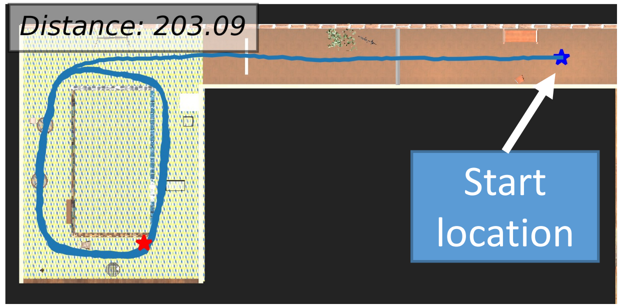
Drone Flight Trajectory Demo

No fault:

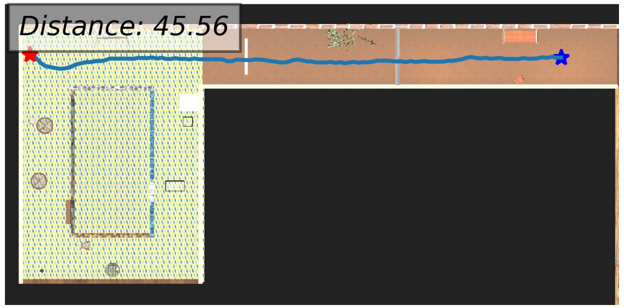


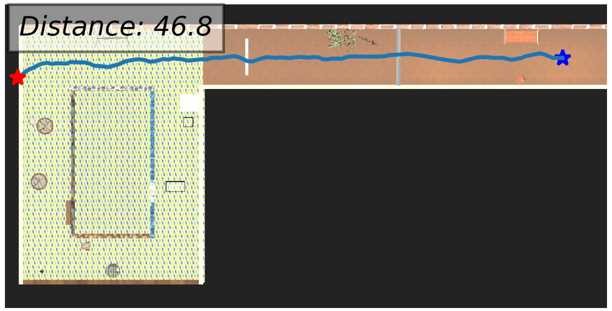
Drone Flight Trajectory Demo

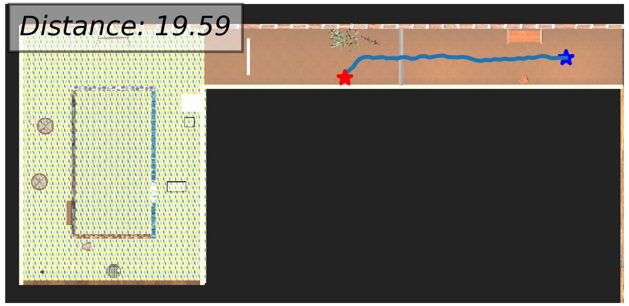
No fault:



Fault injected:

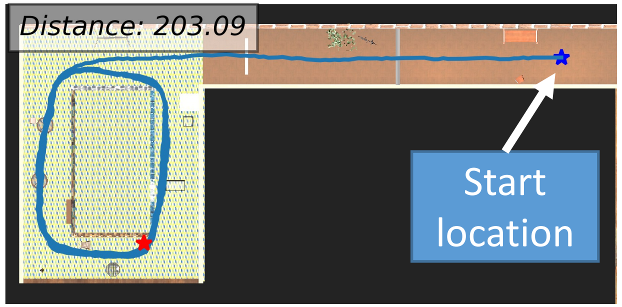




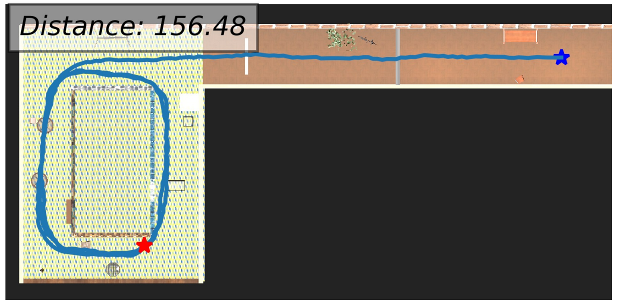


Drone Flight Trajectory Demo

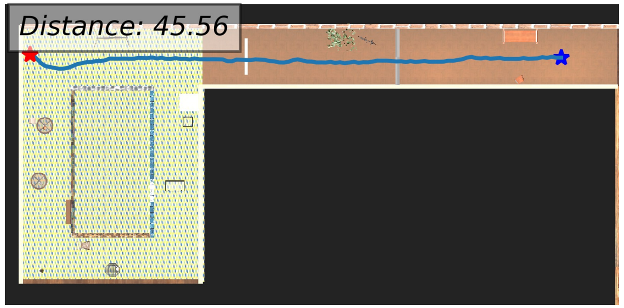
No fault:

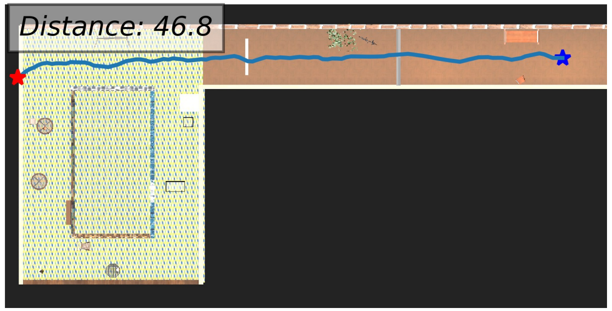


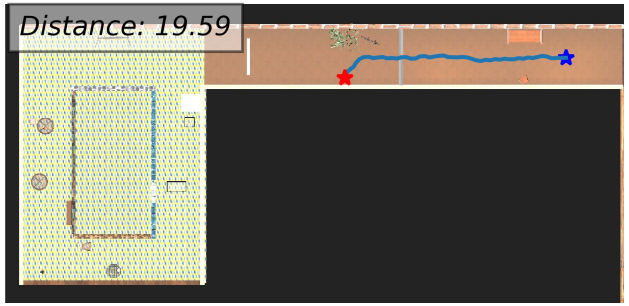
Fault mitigated:



Fault injected:





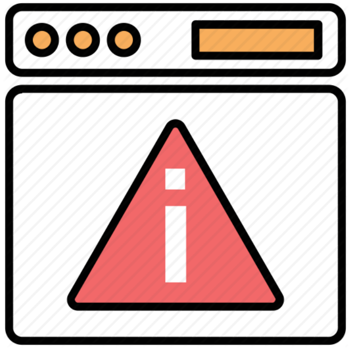


In this talk, “Analyzing and Improving Fault Tolerance of Learning-Based Navigation System”

The **safety and reliability** of end-to-end **learning-based navigation systems** is important, but not well understood

A **fault injection tool-chain** that emulates hardware faults and enables rapid fault analysis of learning-based navigation systems

Large-scale **fault injection study** in both training and inference stages of learning-based systems against permanent and transient faults



Low-overhead **fault detection and recovery techniques** for both training and inference

Thank you
Any Question?

Email: zishenwan@gatech.edu