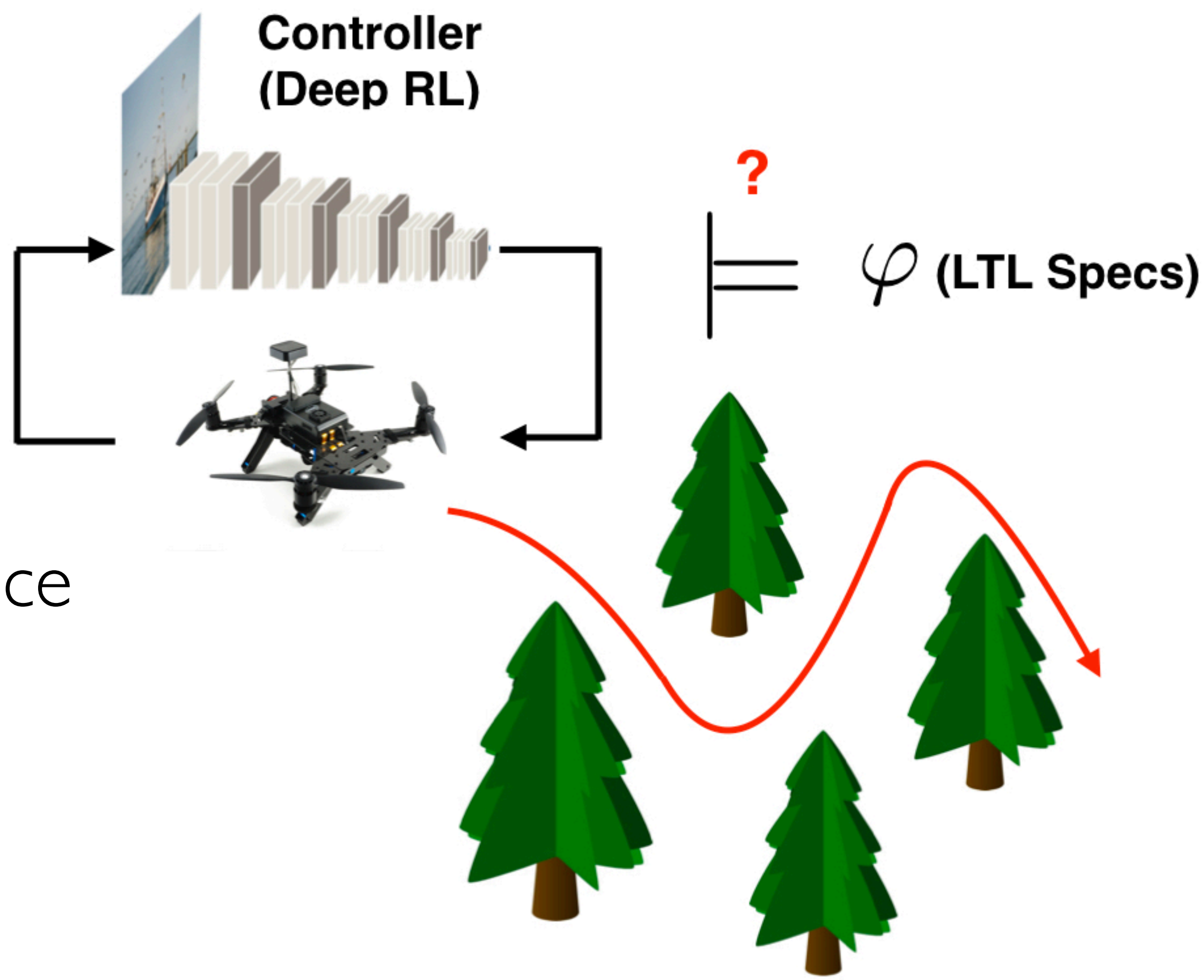
Assured Perception and Control of Autonomous Systems Using Formal Verification of Neural Networks

Yasser Shoukry

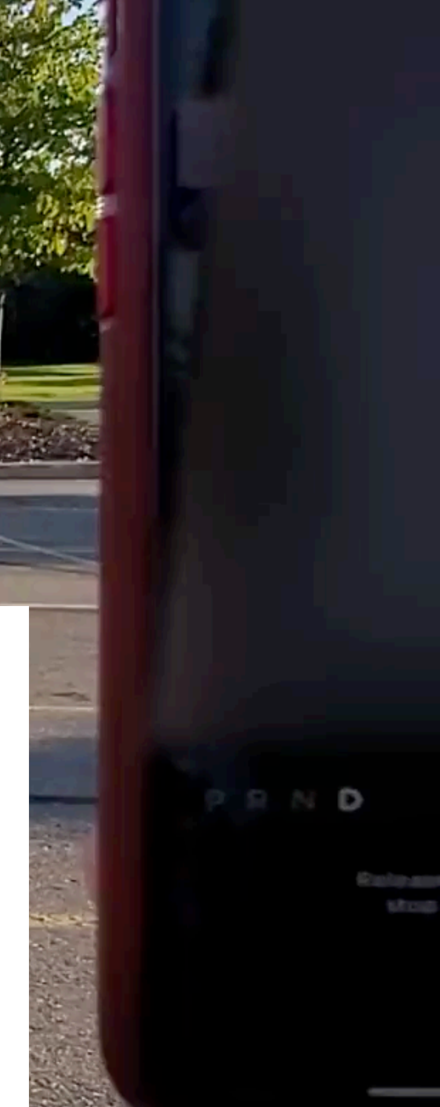
Associate Professor
Resilient Cyber-Physical Systems Lab
Electrical Engineering and Computer Science
University of California, Irvine



February 2, 2024

UCIrvine

Resilient CPS Lab

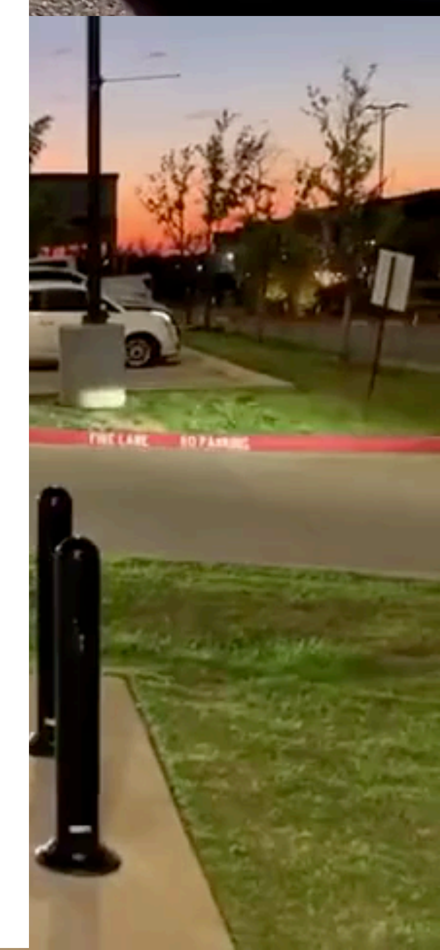


Video released of Uber self-driving crash that killed woman in Arizona

New footage of the crash that killed Elaine Herzberg raises fresh questions about why the self-driving car did not stop

▲ Uber dashcam footage shows lead up to fatal self-driving crash – video

The Guardian, Mar 22 2018



Assured NN-based Perception

Perception-based control is an enabling technology for the state of the art of Autonomous systems.

These systems rely on **machine vision** to detect ***objects of*** interest.

Assured NN-based Perception

Subtle changes lead to several misidentifications

Assured NN-based Perception

Subtle changes lead to several misidentifications

SOTA perception-based systems are not reliable due to the use of learning base neural networks.

Assured NN-based Perception

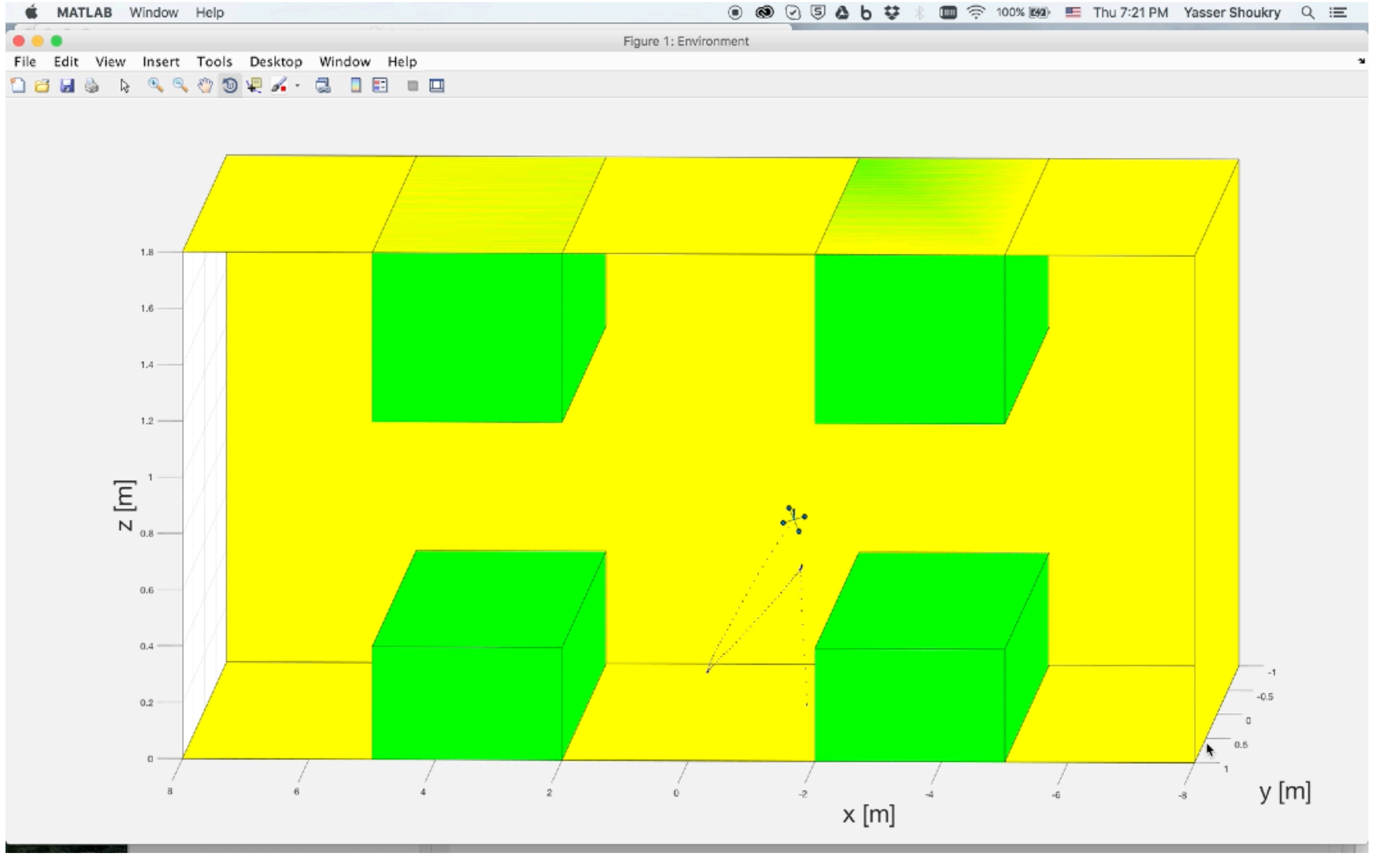
Subtle changes lead to several misidentifications

SOTA perception-based systems are not reliable due to the use of learning base neural networks.

Assured NN-based Perception
=
Design Neural Networks for machine Vision with provable guarantees.

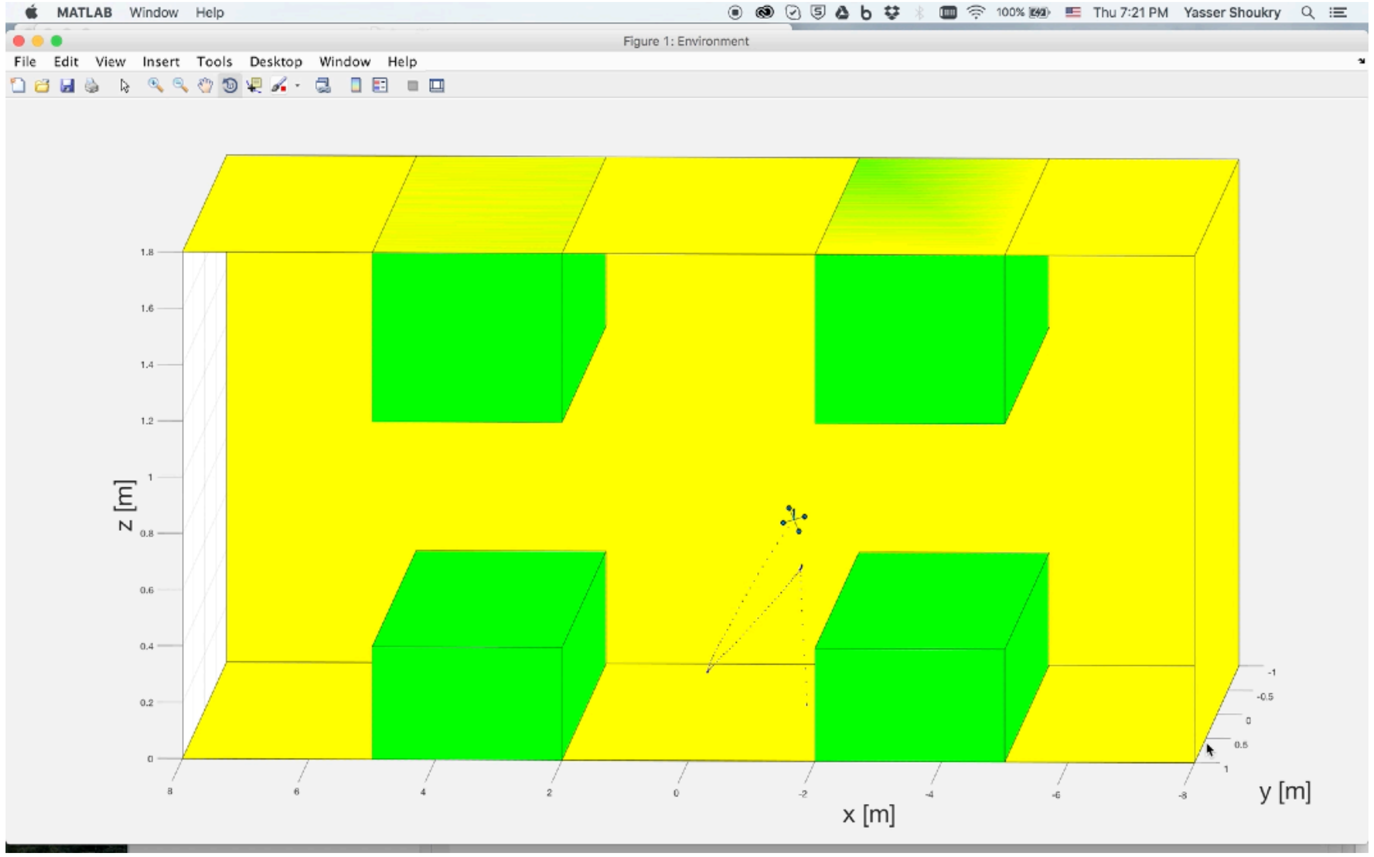
CPS Lab
Resilient

Assured NN-based Control

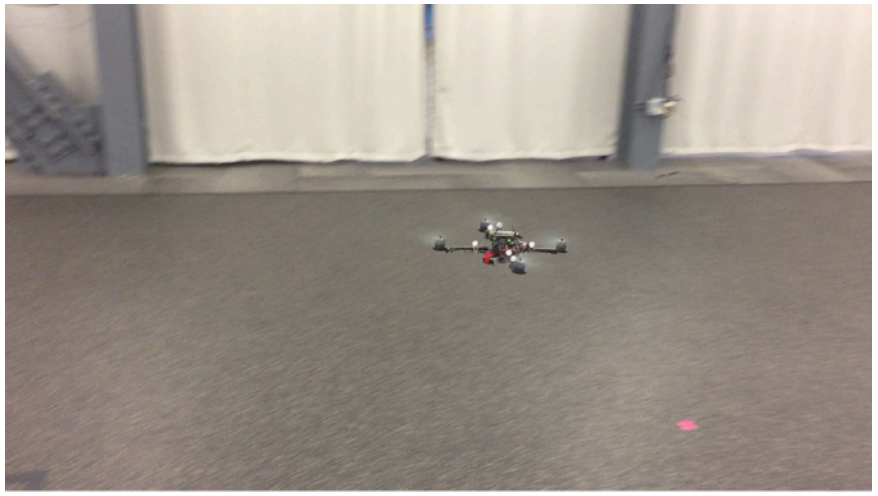


Training Phase

Assured NN-based Control

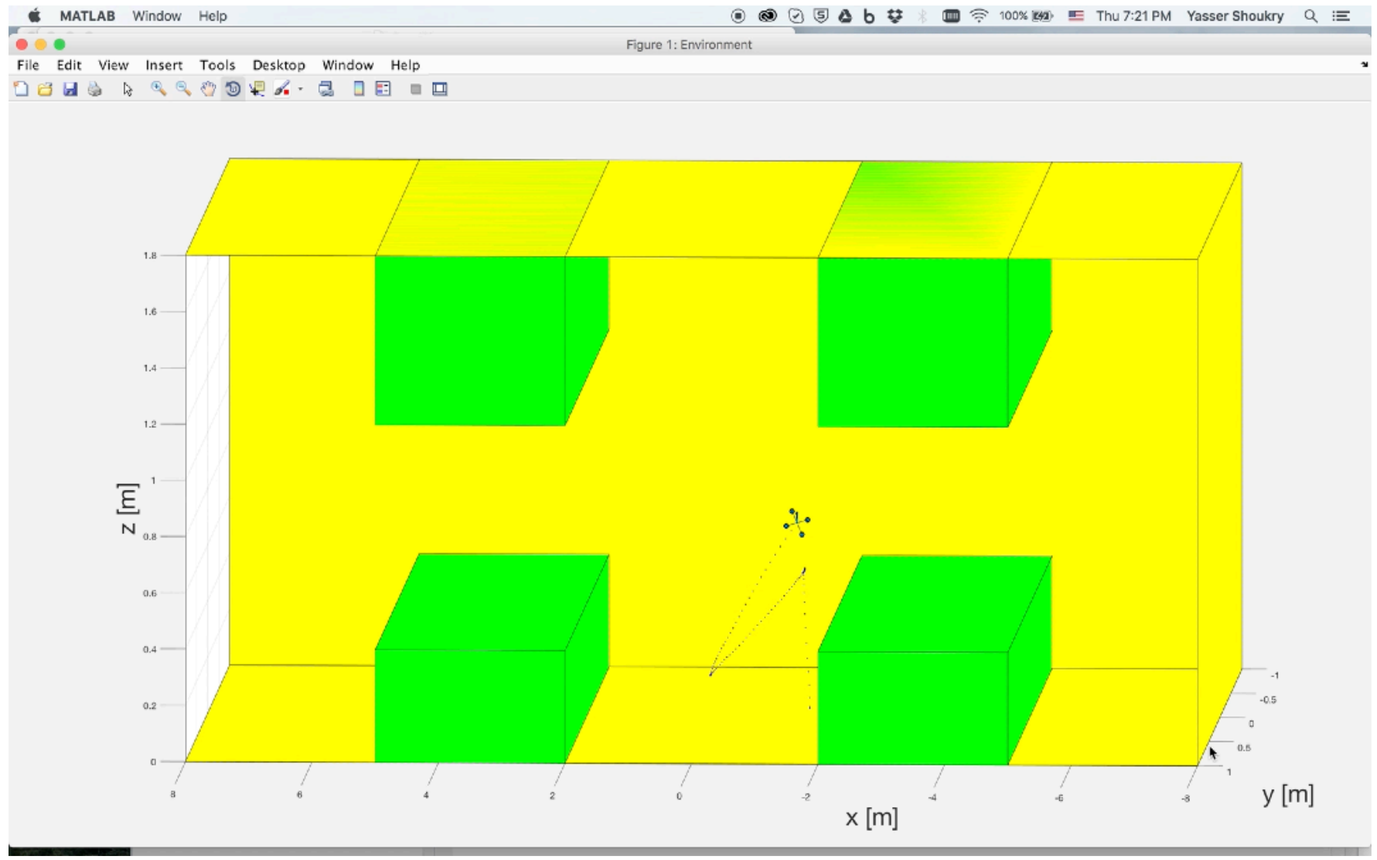


Training Phase

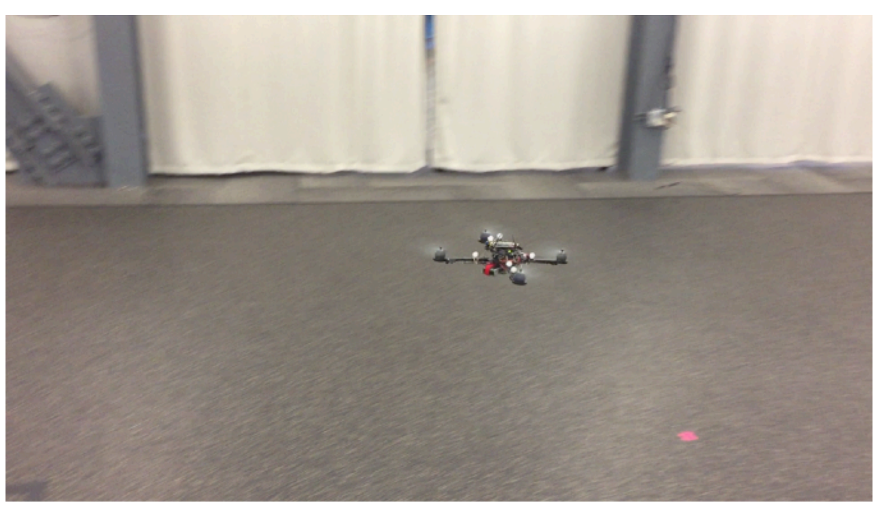


Different dynamics

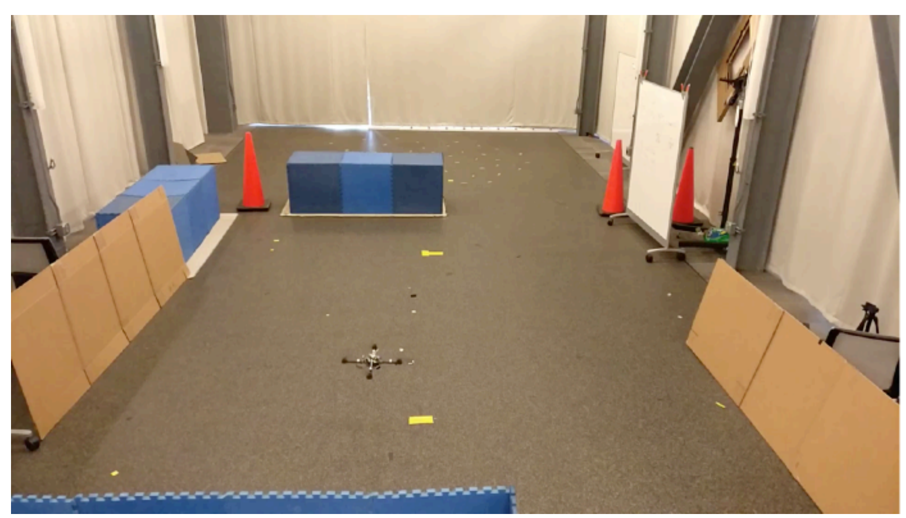
Assured NN-based Control



Training Phase

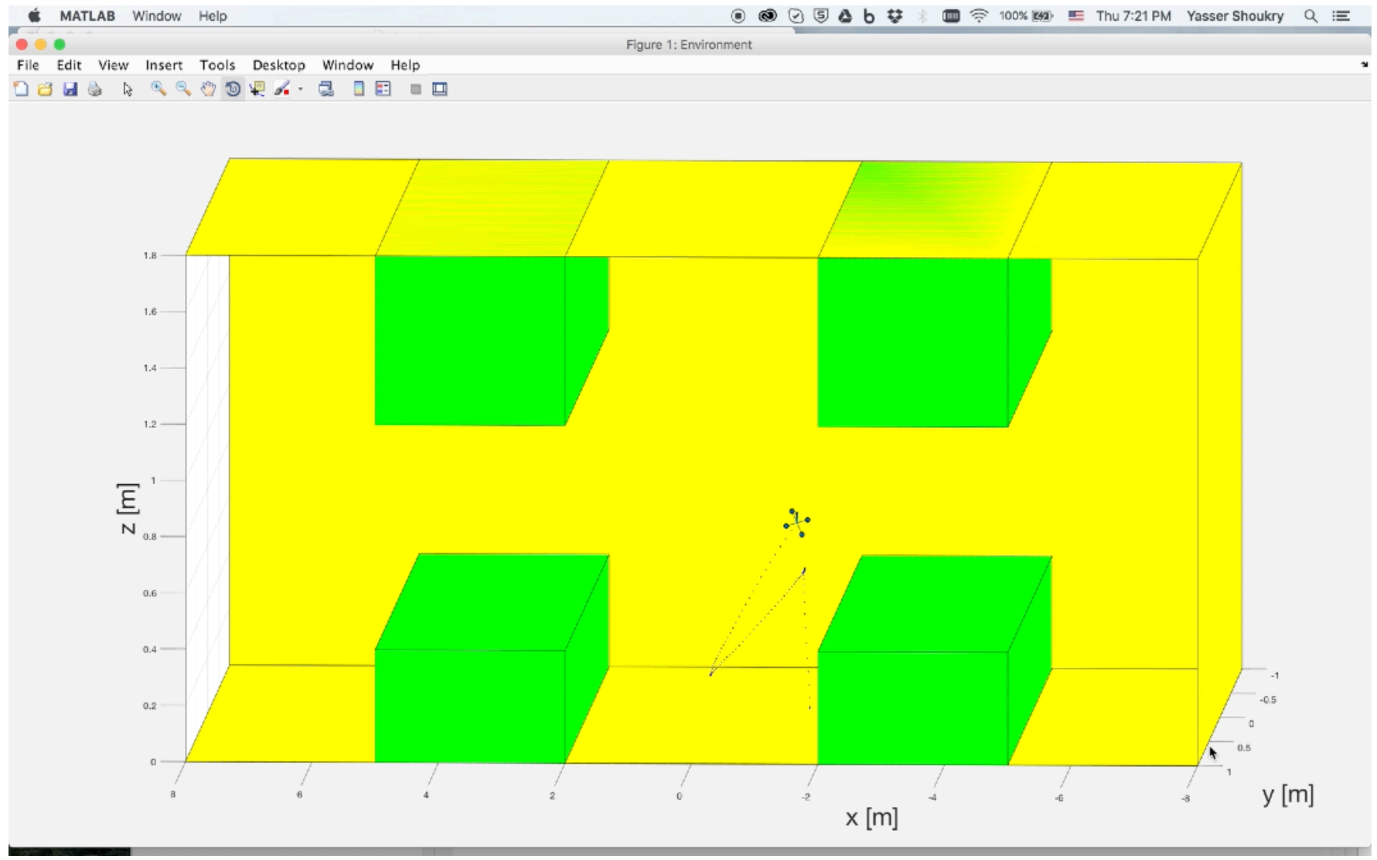


Different dynamics

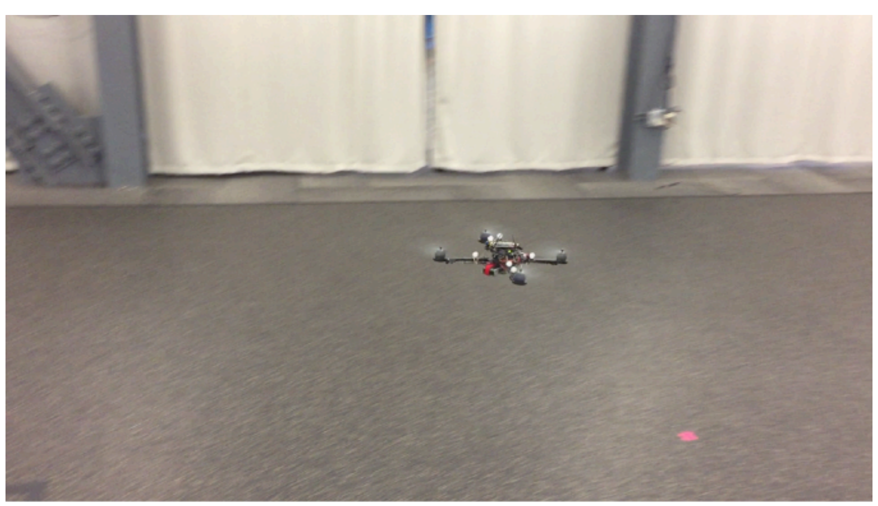


Different workspaces

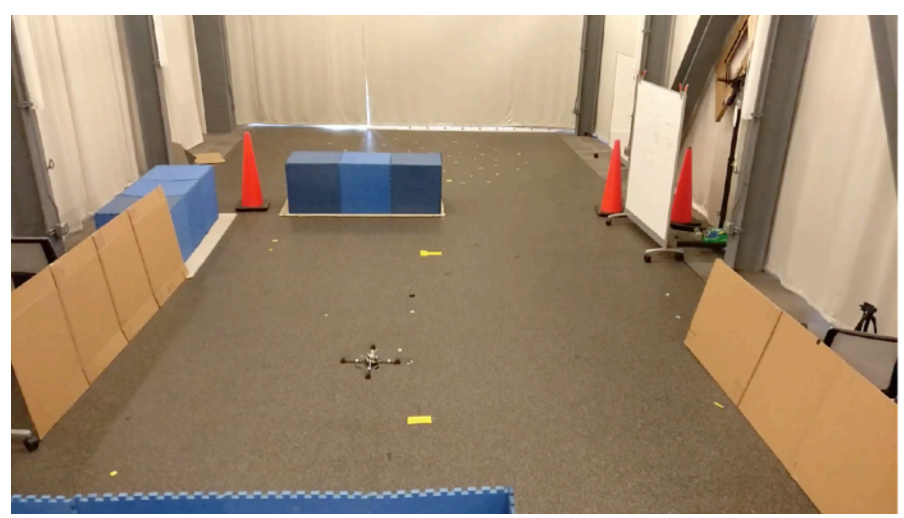
Assured NN-based Control



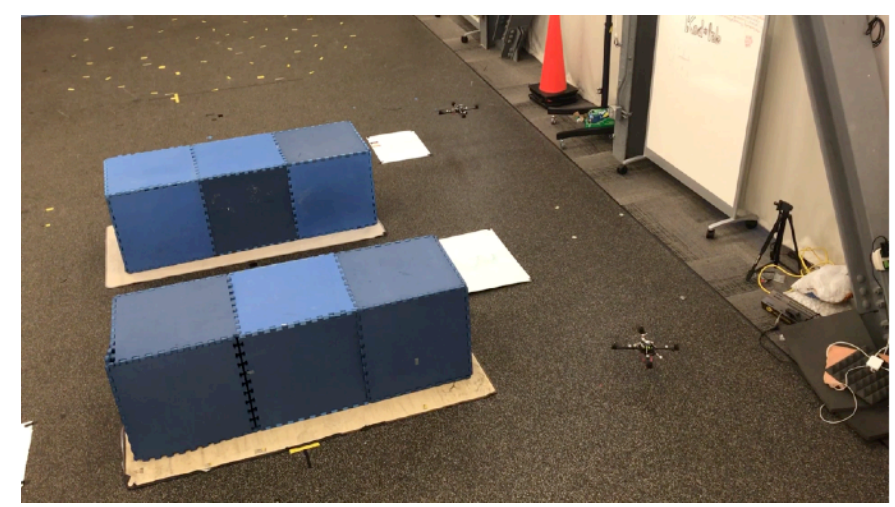
Training Phase



Different dynamics

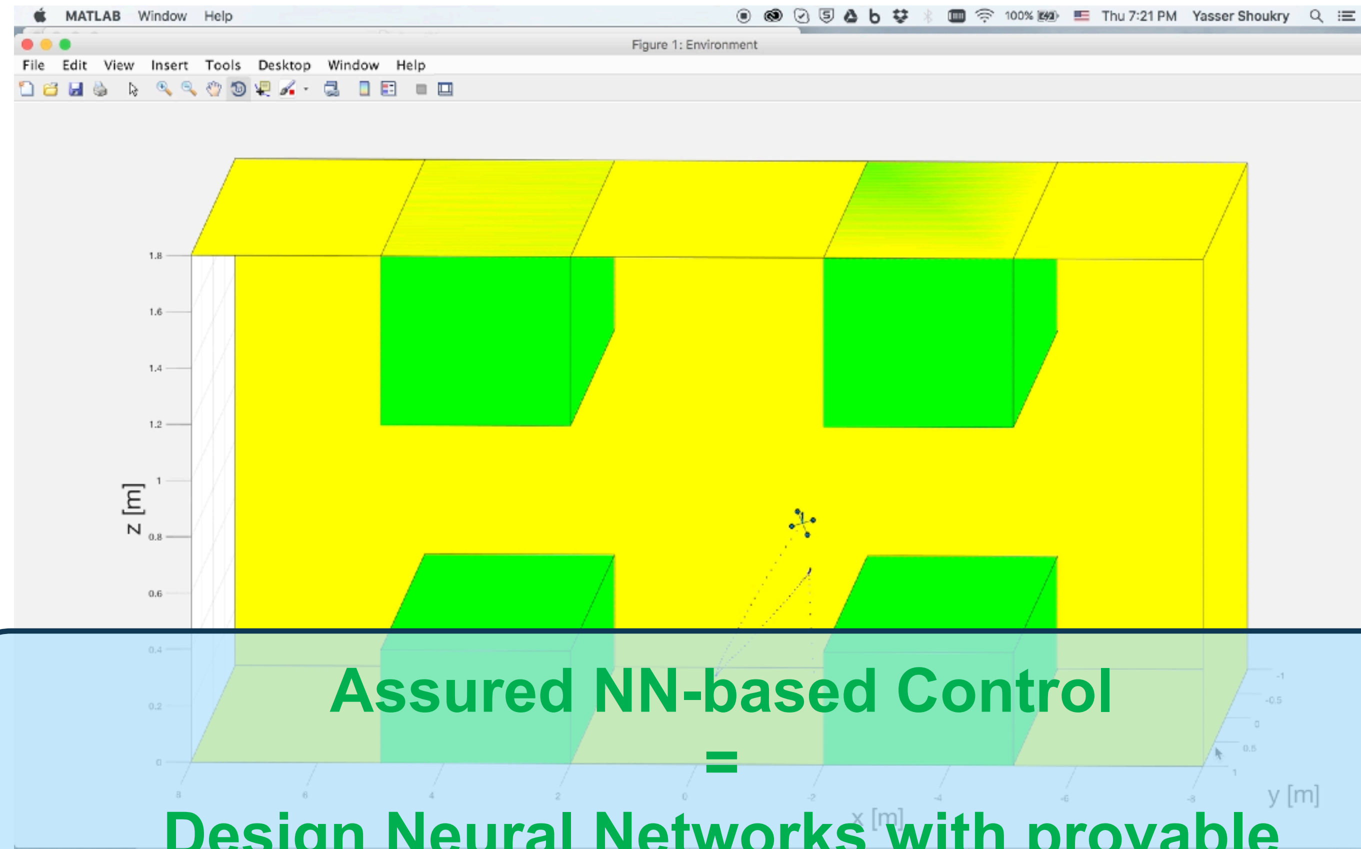


Different workspaces



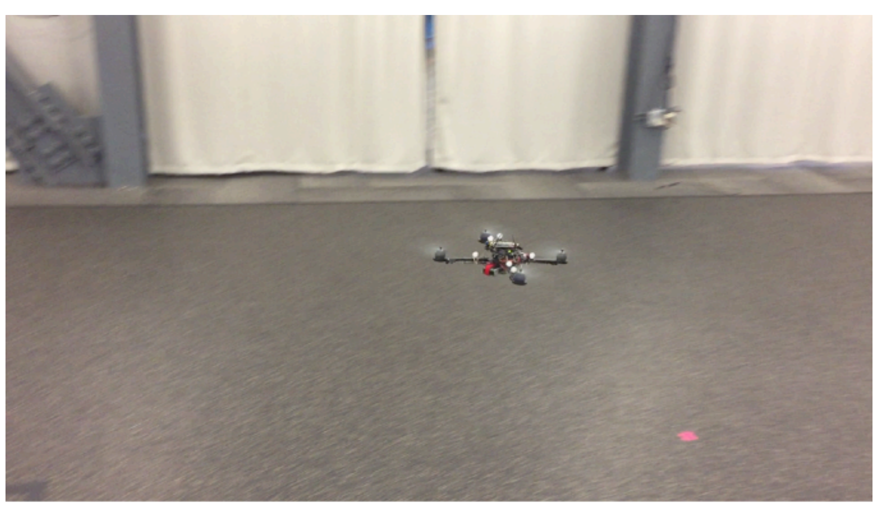
Different temporal mission

Assured NN-based Control

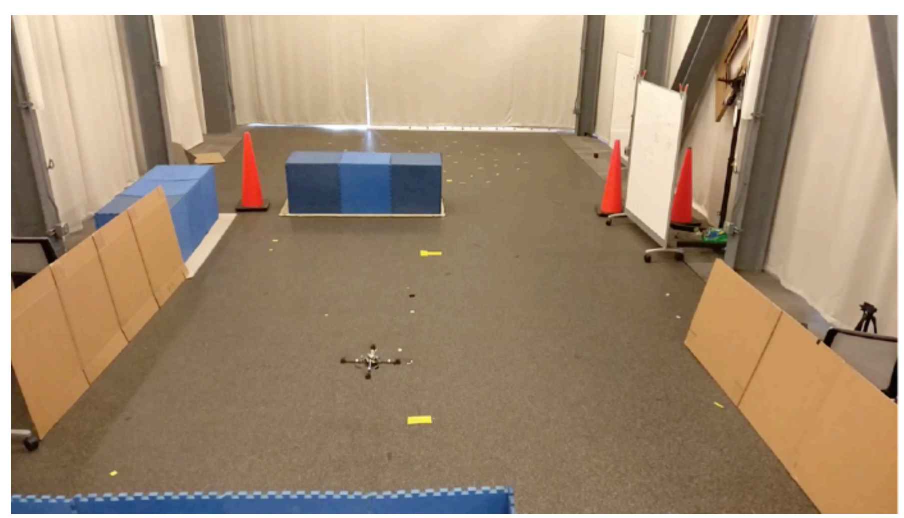


Training Phase

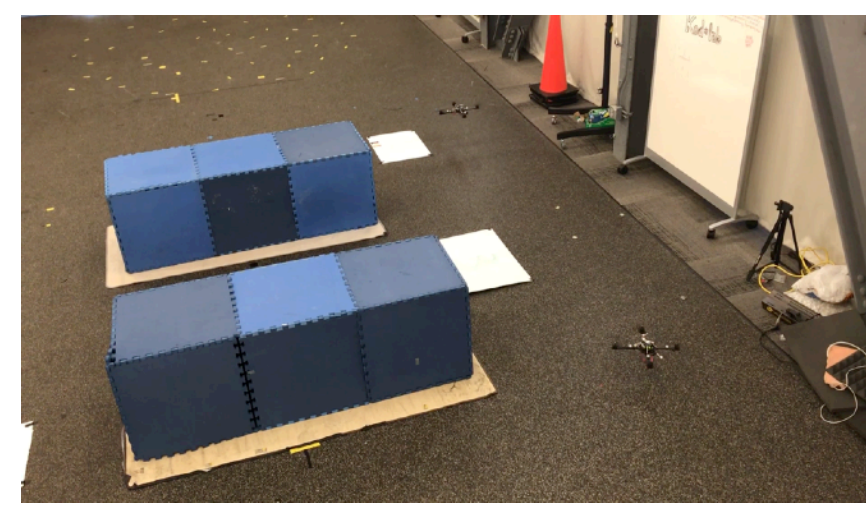
Assured NN-based Control
=
Design Neural Networks with provable generalization guarantees.



Different dynamics

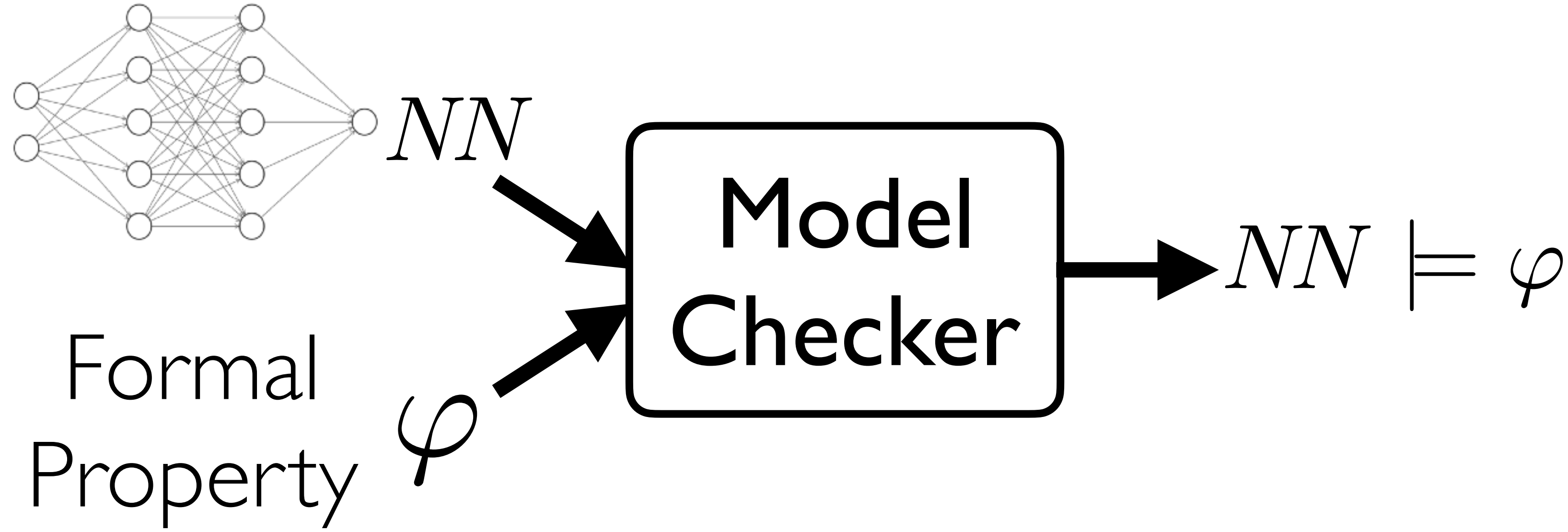


Different workspaces

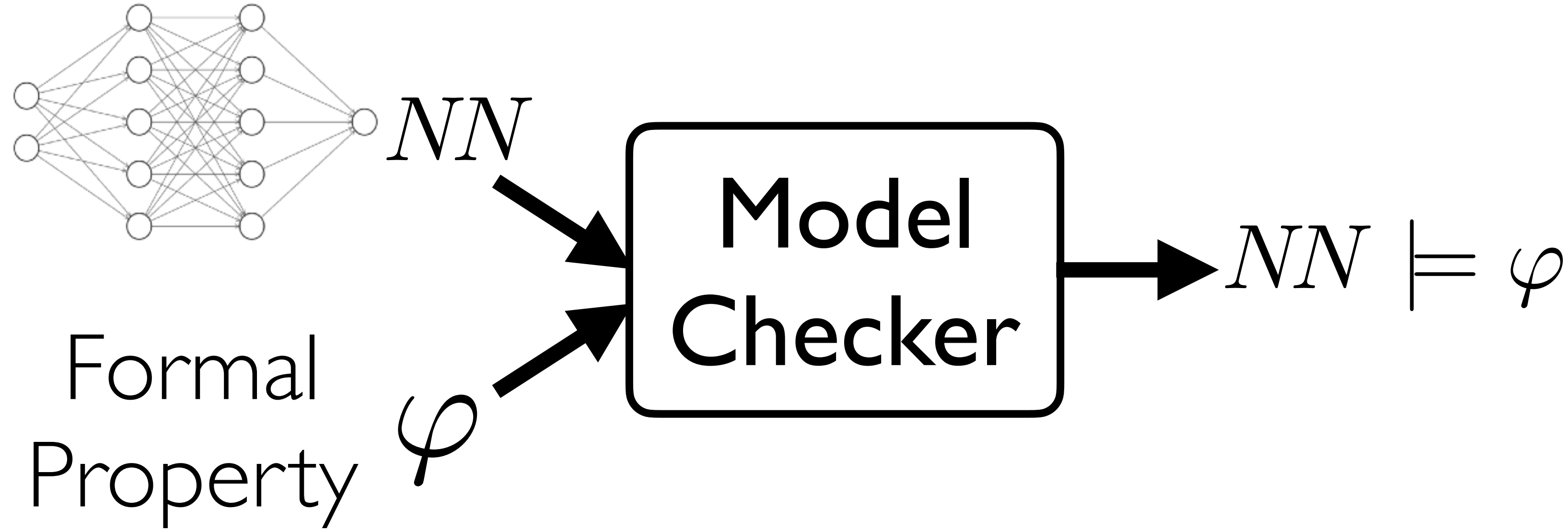


Different temporal mission

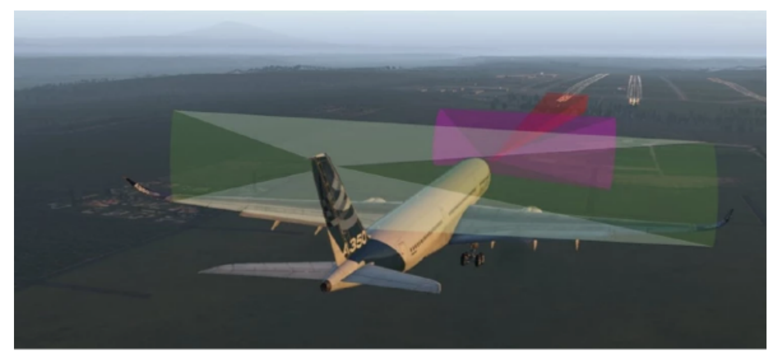
Formal Verification Tools for NN Analysis



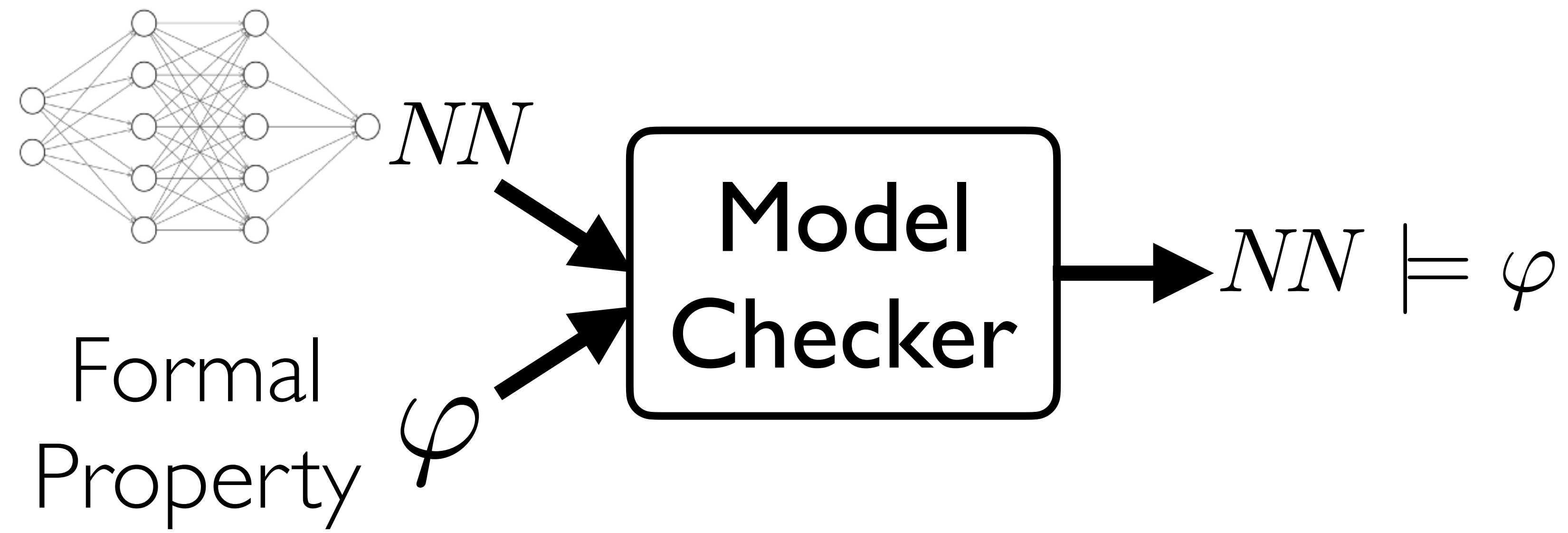
Formal Verification Tools for NN Analysis



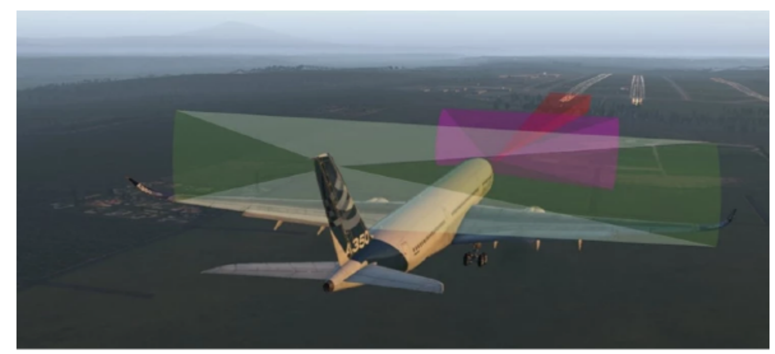
Assured NN-based Perception



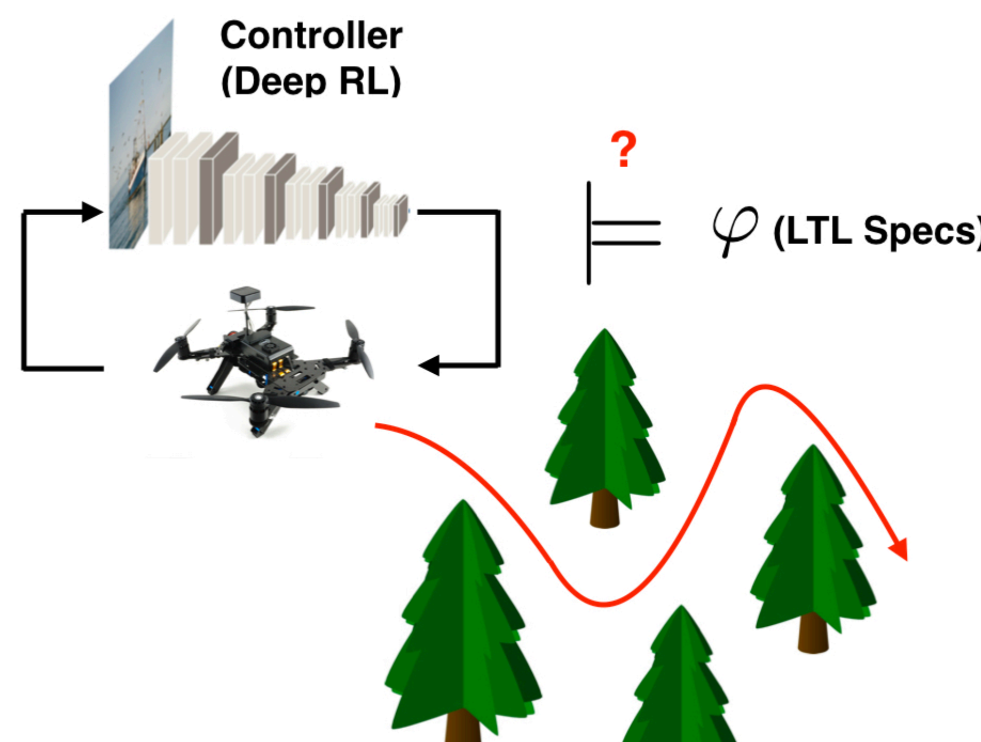
Formal Verification Tools for NN Analysis



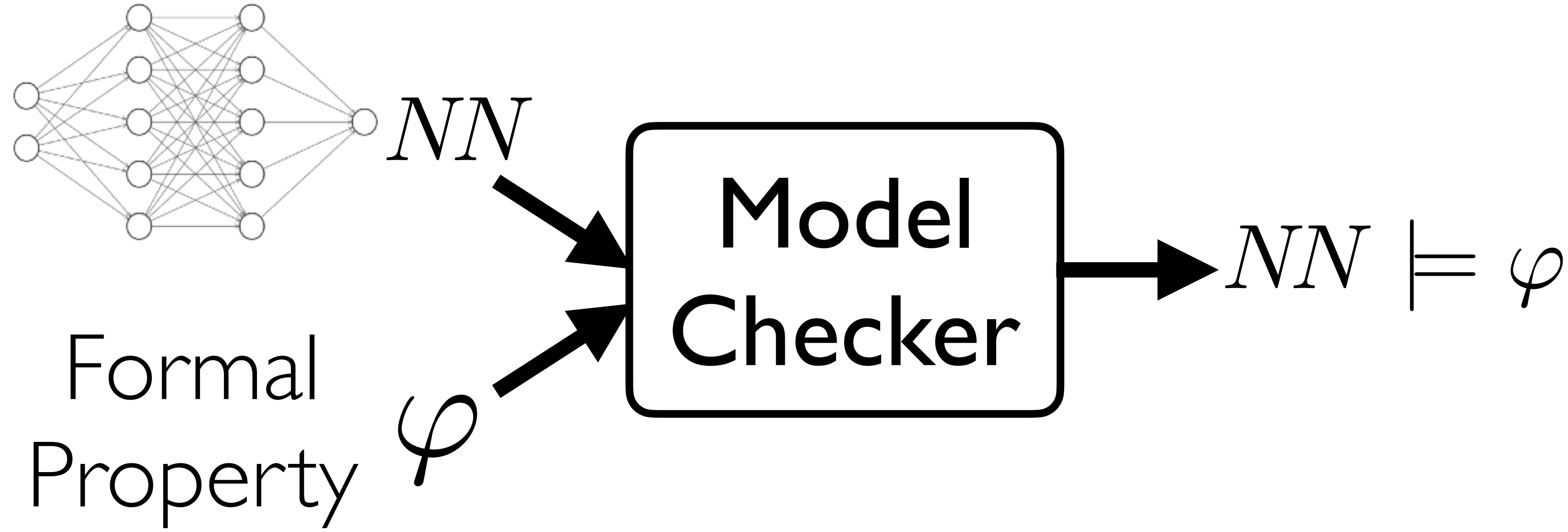
Assured NN-based Perception



Assured NN-based Control

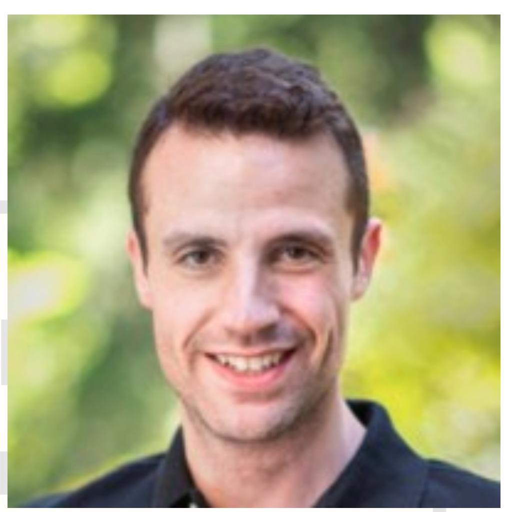


Formal Verification Tools for NN Analysis



Assured NN-based Perception

Haitham Khedr



Dr. James Ferlez

H. Khedr, J. Ferlez, and Y. Shoukry, "PEREGRiNN: Penalized-Relaxation Greedy Neural Network Verifier,"CAV, 2021.

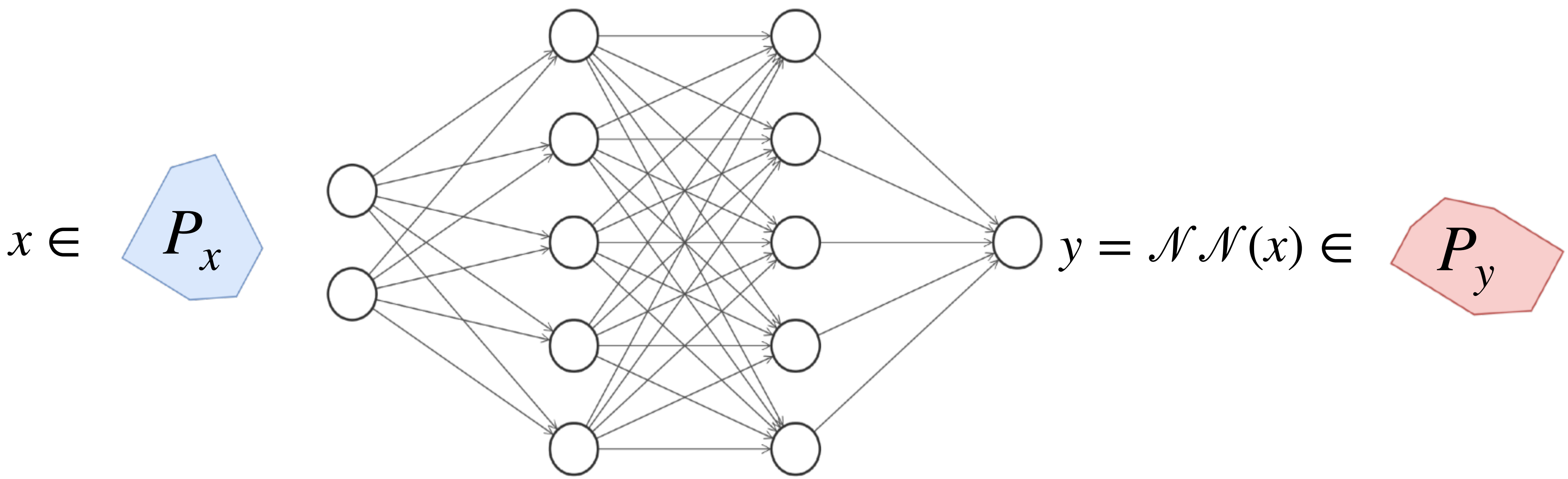
J. Ferlez and Y. Shoukry, "Bounding the Complexity of Formally Verifying Neural Networks: A Geometric Approach," CDC 2021.

J. Ferlez, H. Khedr, and Y. Shoukry, “FastBATLLNN: Fast Box Analysis of Two-Level Lattice Neural Networks,” HSCC 2022.

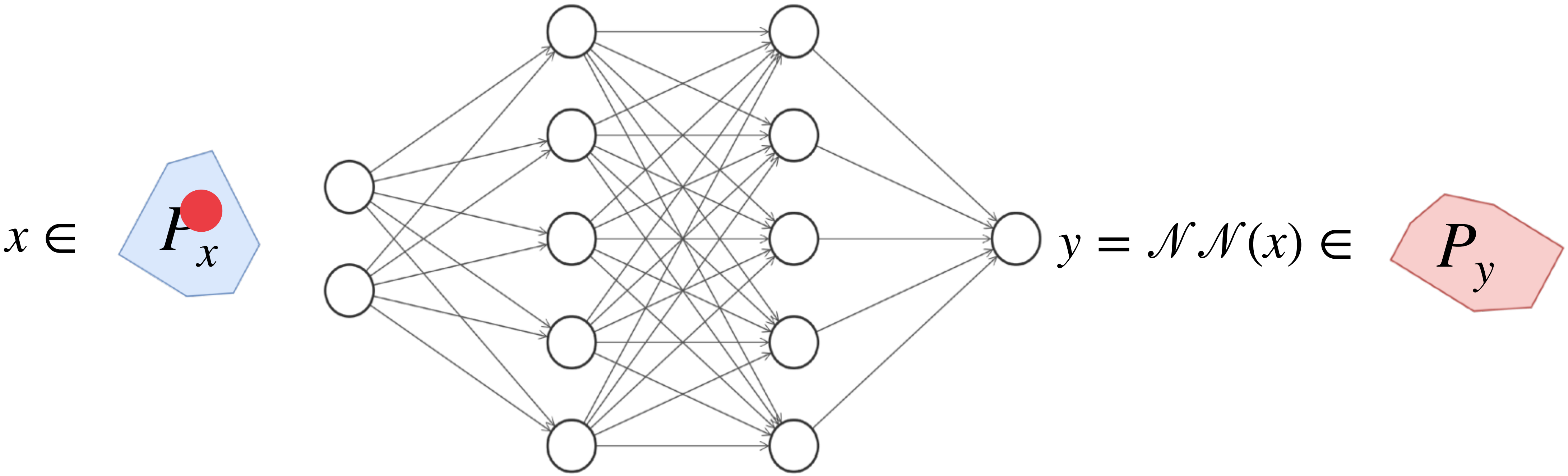
H. Khedr and Y. Shoukry, “DeepBern-Nets: Taming the Complexity of Certifying Neural Networks using Bernstein Polynomial Activations and Precise Bound Propagation,” arXiv 2023.

Haitham Khedr and Yasser Shoukry, “CertiFair: A Framework for Certified Global Fairness of Neural Networks,” 37th AAAI Conference on Artificial Intelligence (AAAI-23).

Input-Output Verification

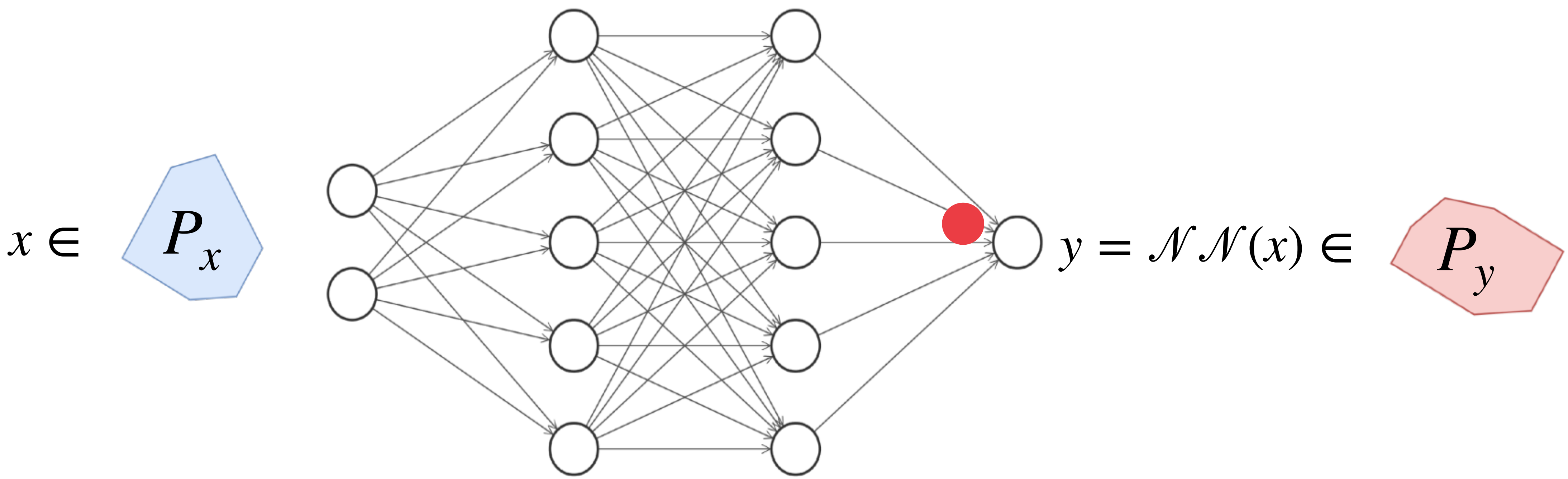


Input-Output Verification

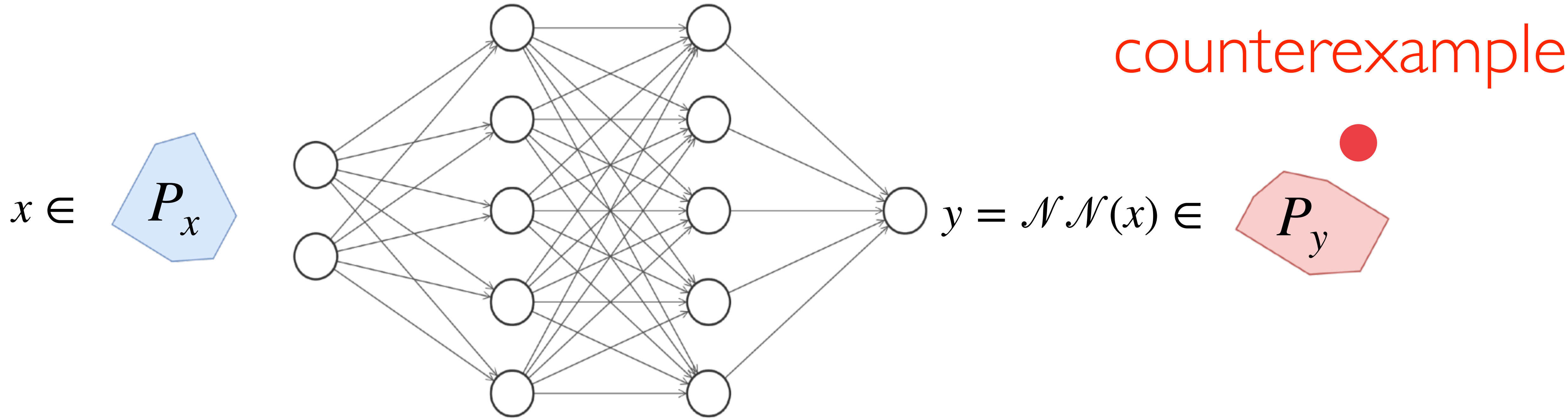


Resilient CPS Lab

Input-Output Verification

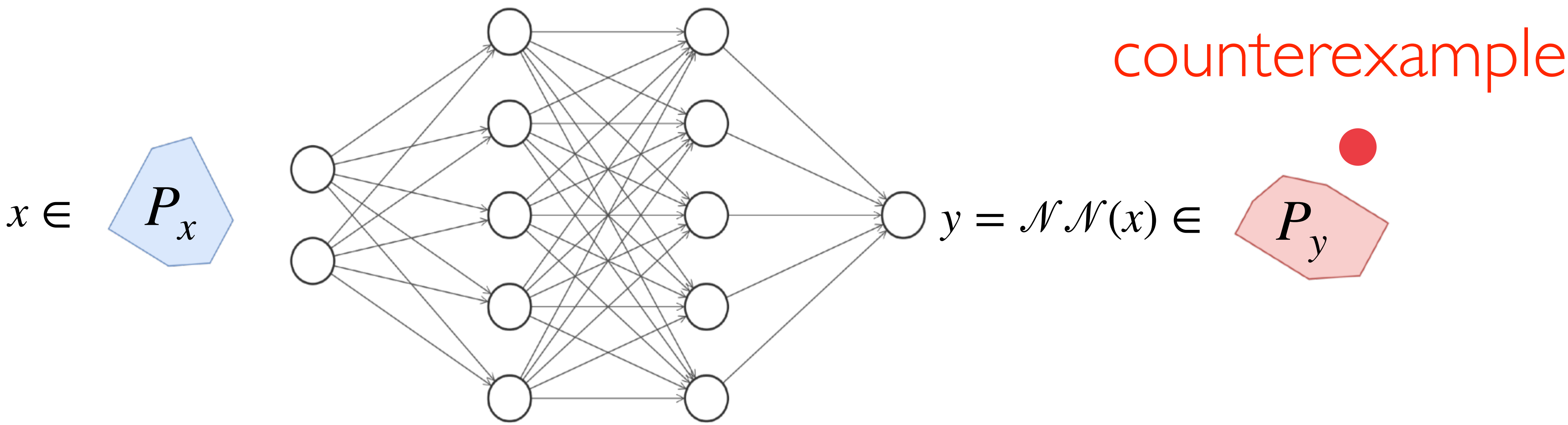


Input-Output Verification



Resilient CPS Lab

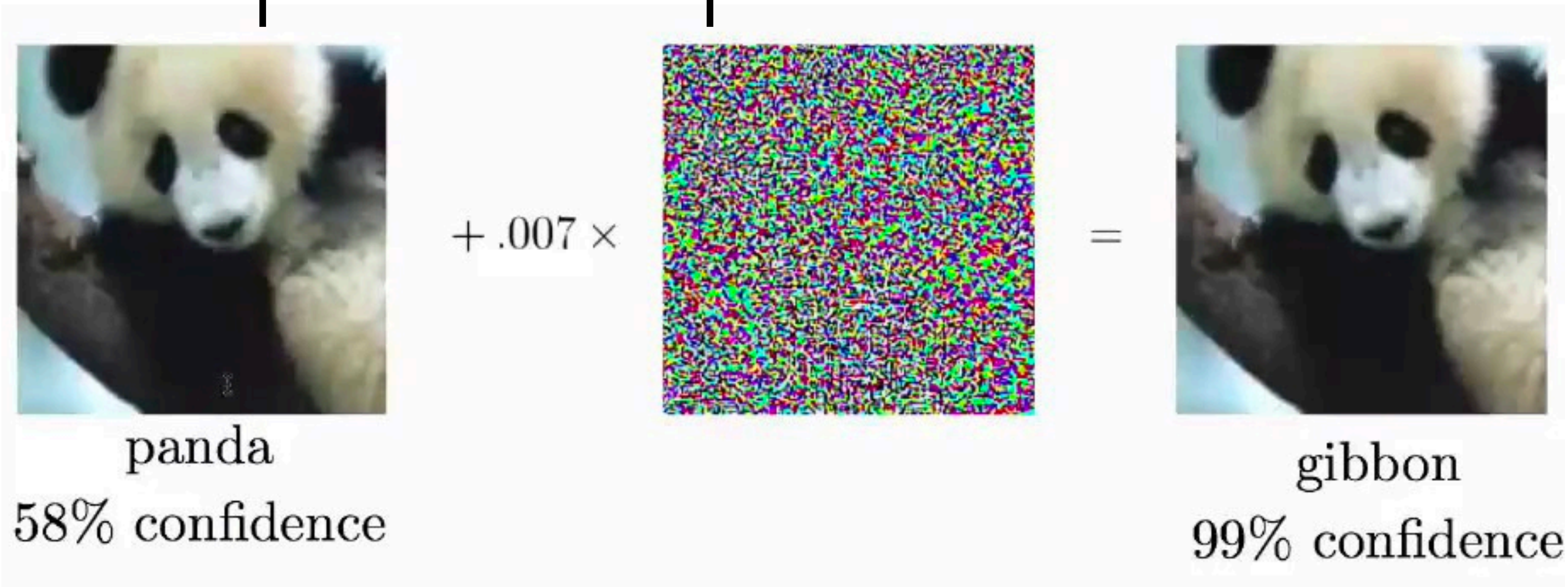
Input-Output Verification



$$\left\{ x \in \mathbb{R}^{k_0} \mid x \in P_x \wedge \mathcal{N}\mathcal{N}(x) \notin P_y \wedge \left(\bigwedge_{\ell=1}^{m} h_\ell(x, \mathcal{N}\mathcal{N}(x)) \leq 0 \right) \right\} = \varnothing$$

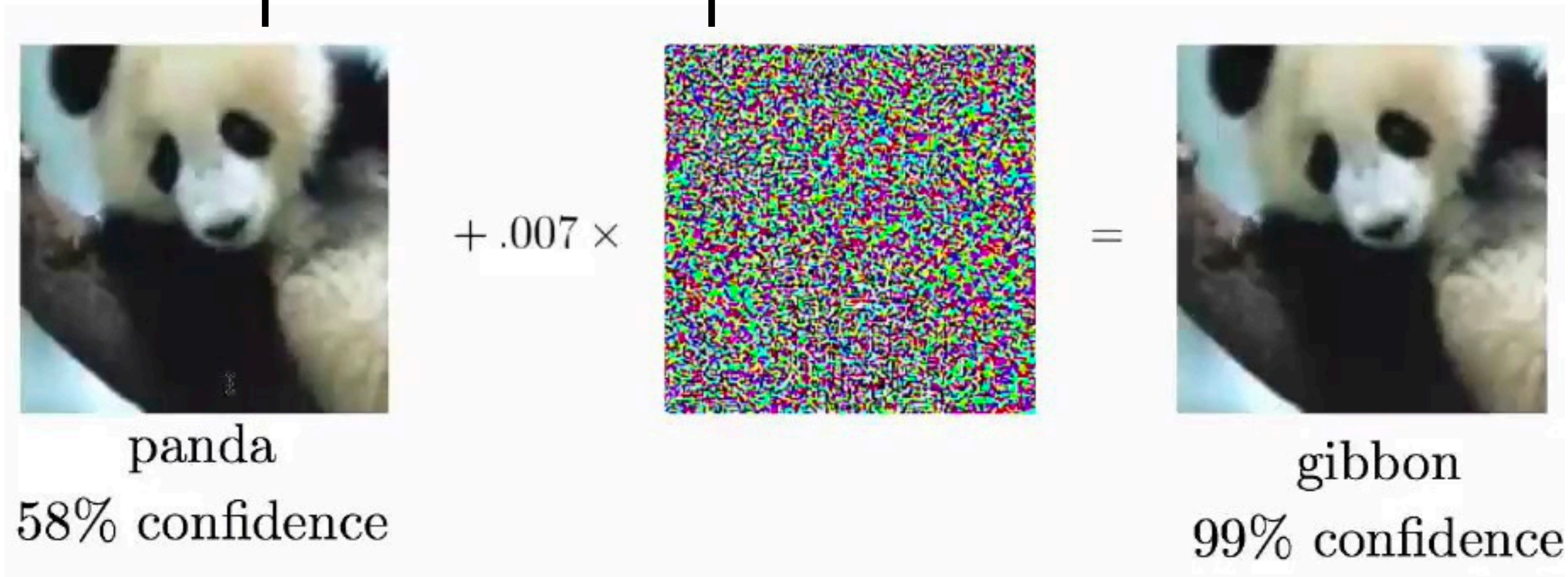
Input constraints — Output constraints — Linear input/output constraints

Input-Output Verification



Adversarial robustness:

Input-Output Verification



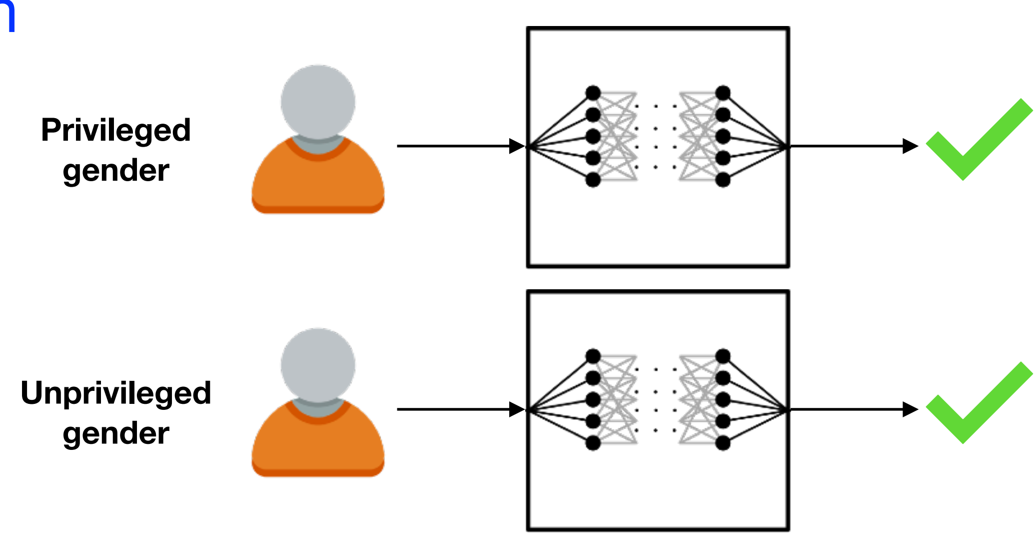
Adversarial robustness:

$$\{x \mid x \in \mathbb{R}^{k_0}, \underbrace{\|x - x'\|_\infty \le \epsilon}_{P_x}, \underbrace{\max_{i=1,\ldots,n} \mathcal{N}\mathcal{N}(x)_i = \mathcal{N}\mathcal{N}(x)_m}_{P_y}\} = \emptyset$$

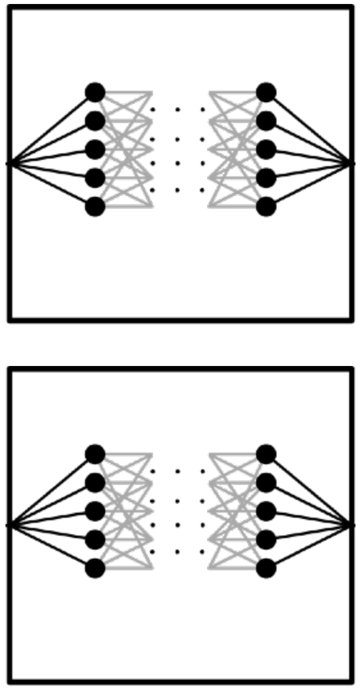
Input-Output Verification

Fairness of Decision Making?

- Similar individuals are to be treated similarly by the decision model (e.g. Hiring decision)
- Examples of similarity
    - Gender/race (sensitive attribute) invariance
    - Closeness in feature space

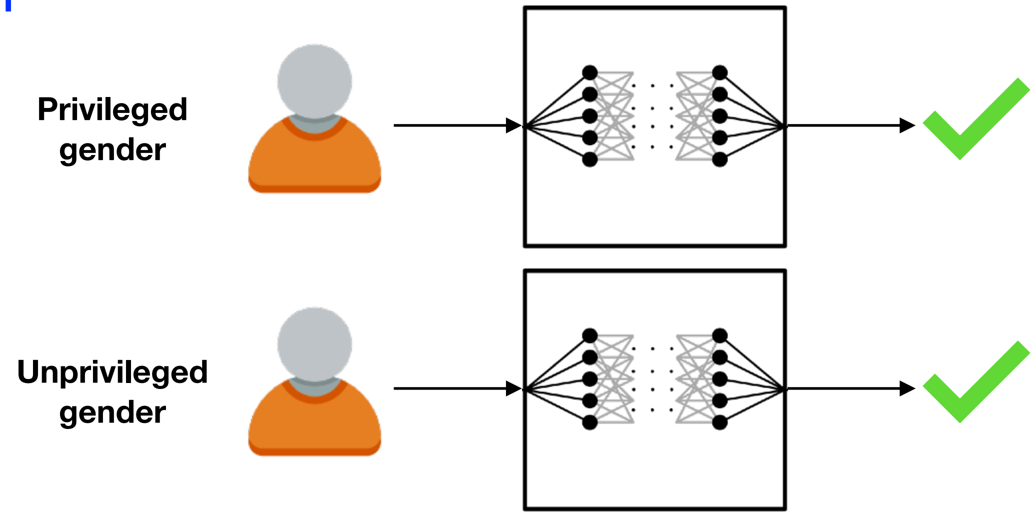


Input-Output Verification

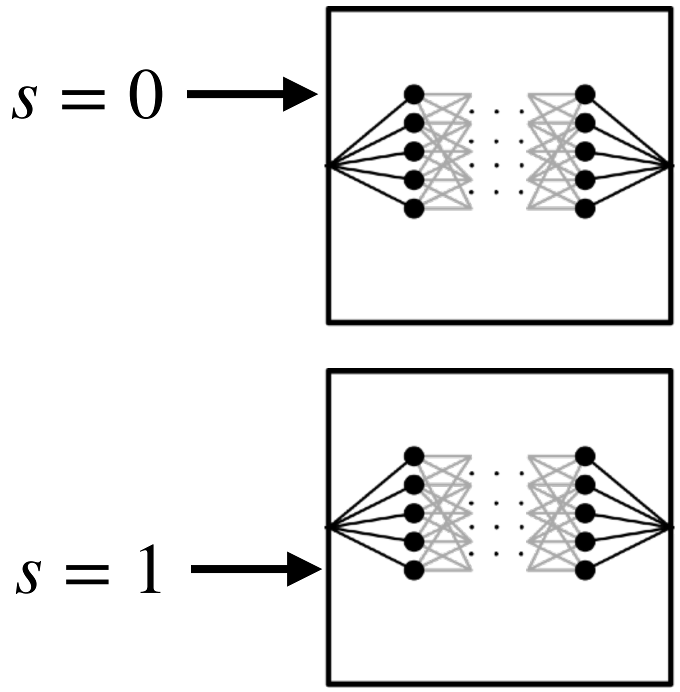


Fairness of Decision Making?

- Similar individuals are to be treated similarly by the decision model (e.g. Hiring decision)
- Examples of similarity
    - Gender/race (sensitive attribute) invariance
    - Closeness in feature space

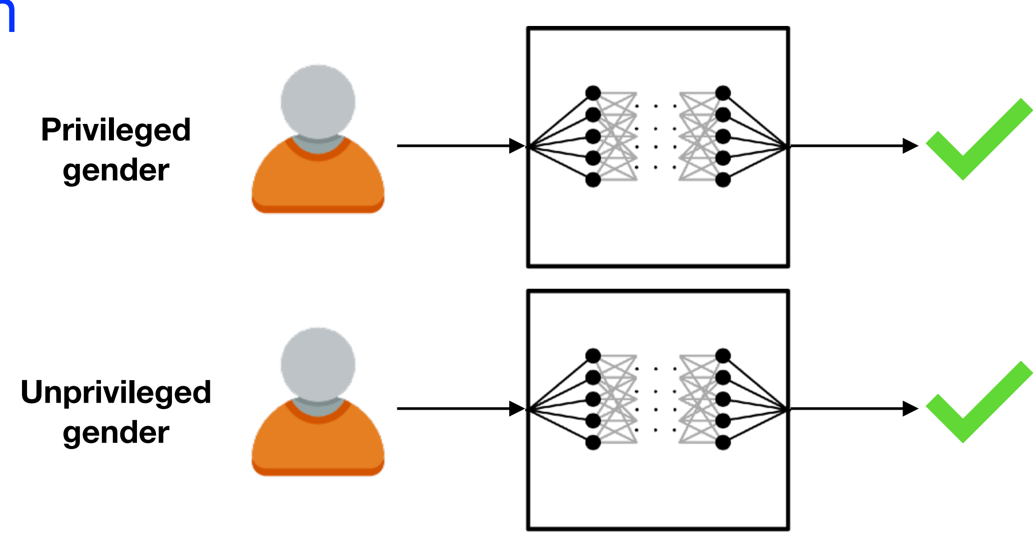


Input-Output Verification

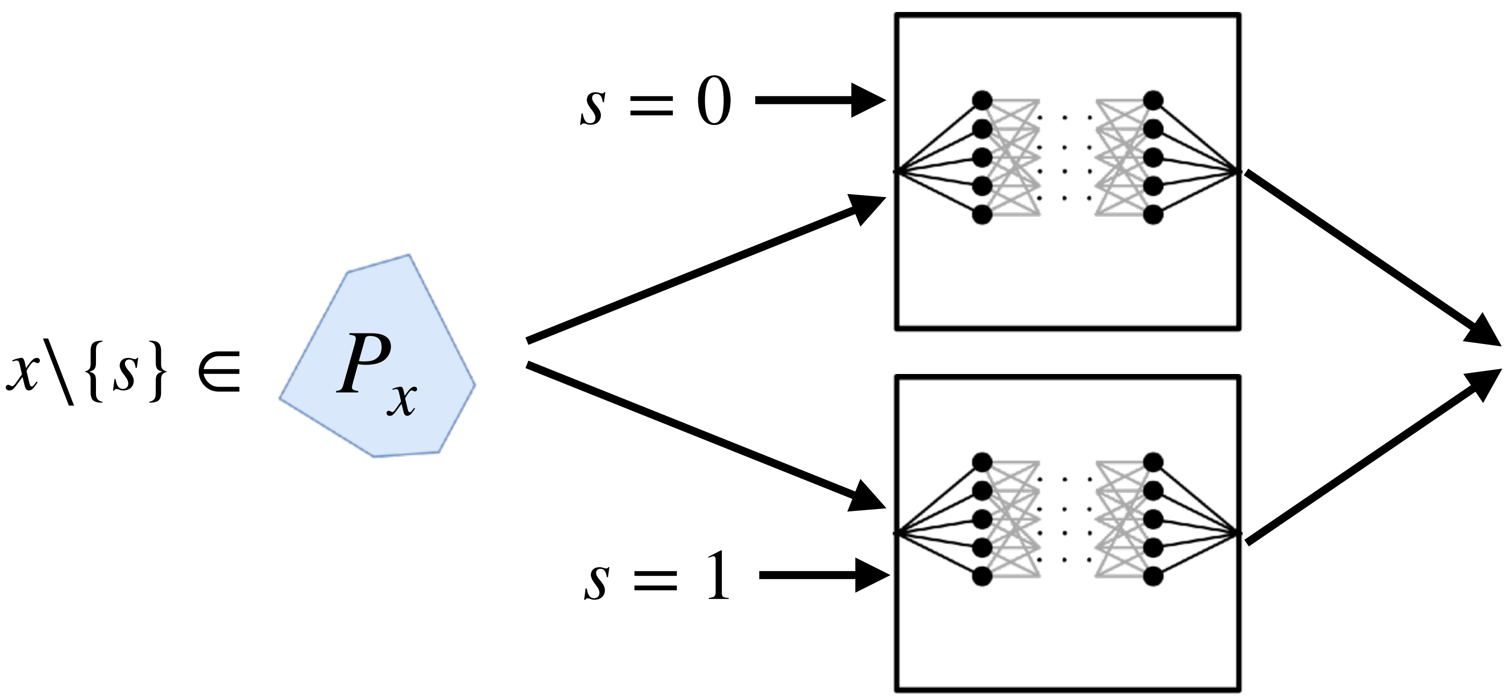


Fairness of Decision Making?

- Similar individuals are to be treated similarly by the decision model (e.g. Hiring decision)
- Examples of similarity
  - Gender/race (sensitive attribute) invariance
  - Closeness in feature space

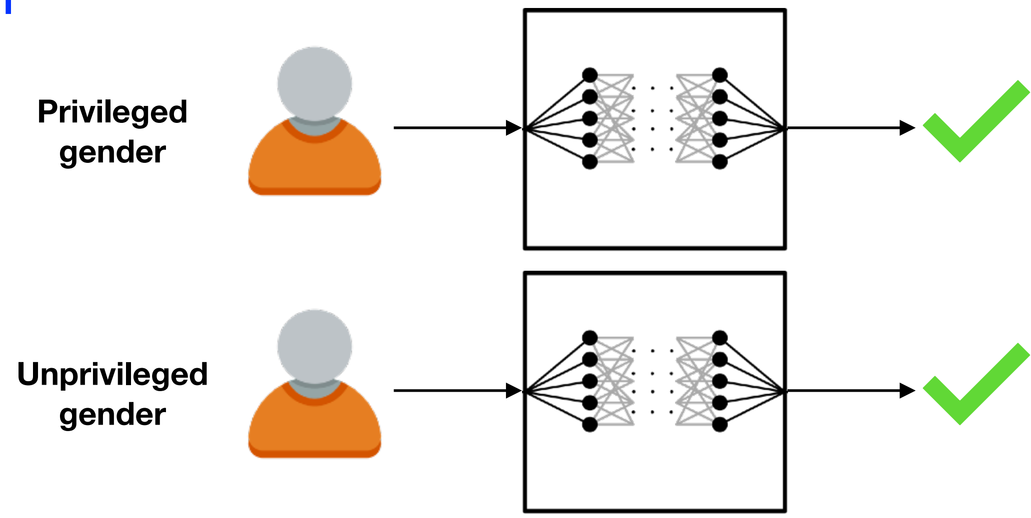


Input-Output Verification

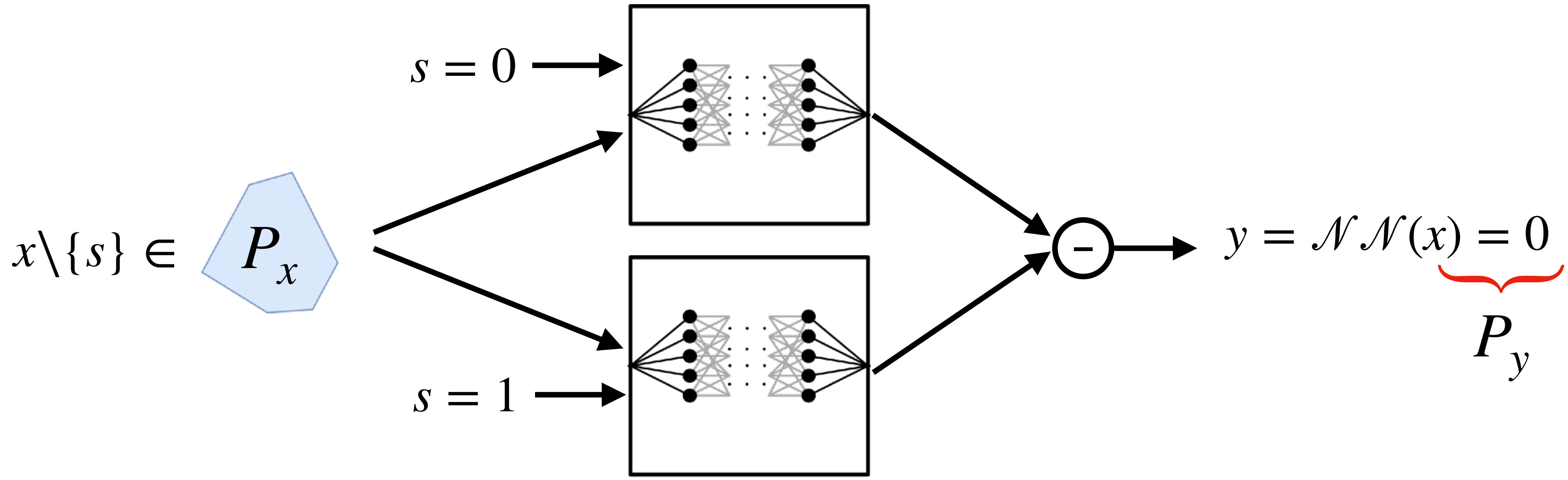


Fairness of Decision Making?

- Similar individuals are to be treated similarly by the decision model (e.g. Hiring decision)
- Examples of similarity
    - Gender/race (sensitive attribute) invariance
    - Closeness in feature space

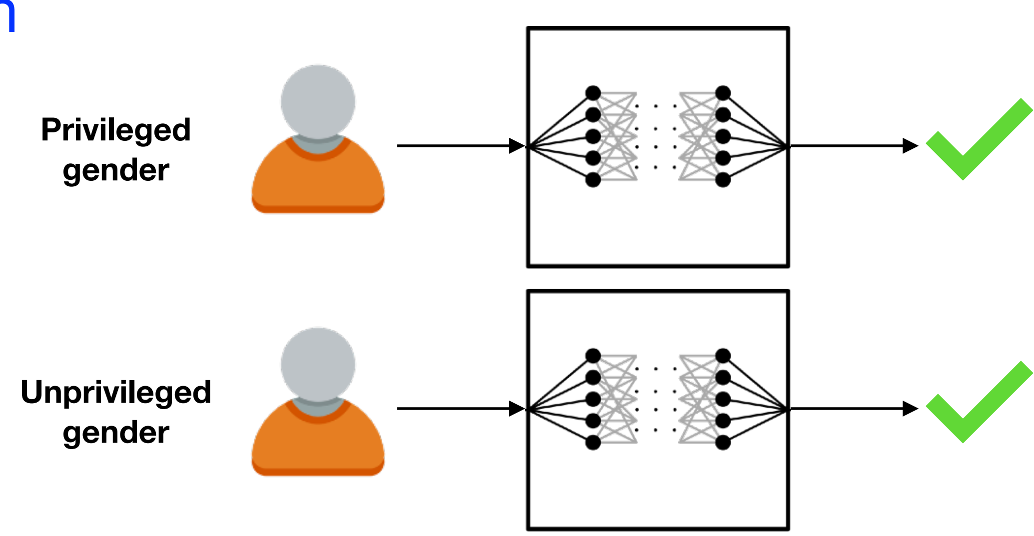


Input-Output Verification

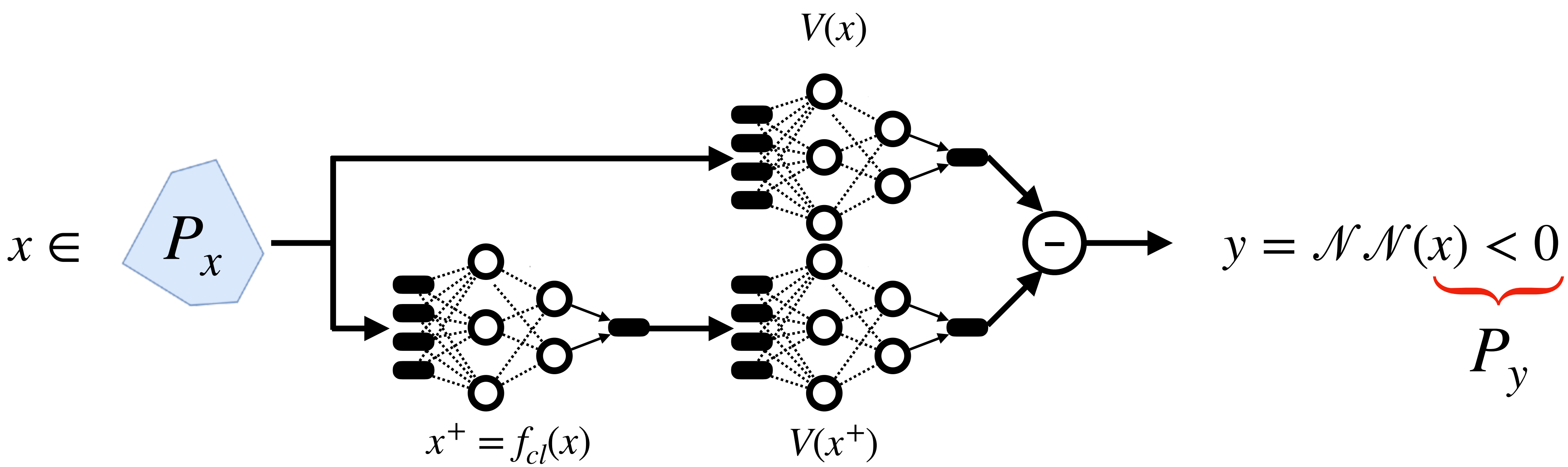


Fairness of Decision Making?

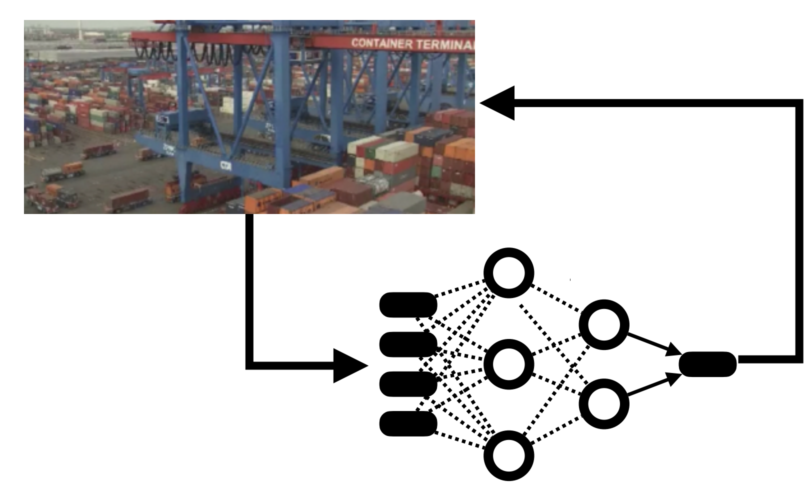
- Similar individuals are to be treated similarly by the decision model (e.g. Hiring decision)
- Examples of similarity
    - Gender/race (sensitive attribute) invariance
    - Closeness in feature space



Input-Output Verification



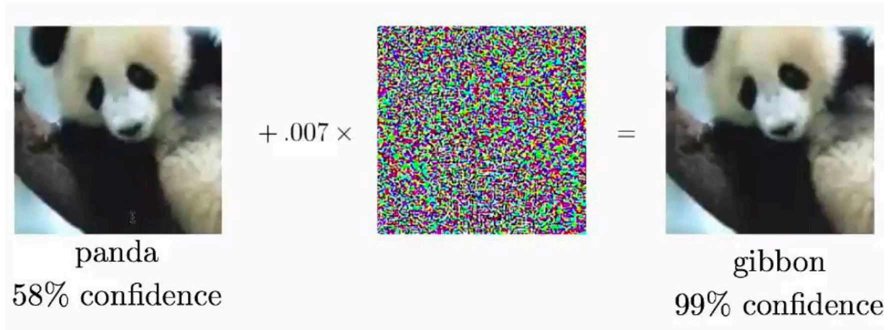
Decision Making?



Feedback controller

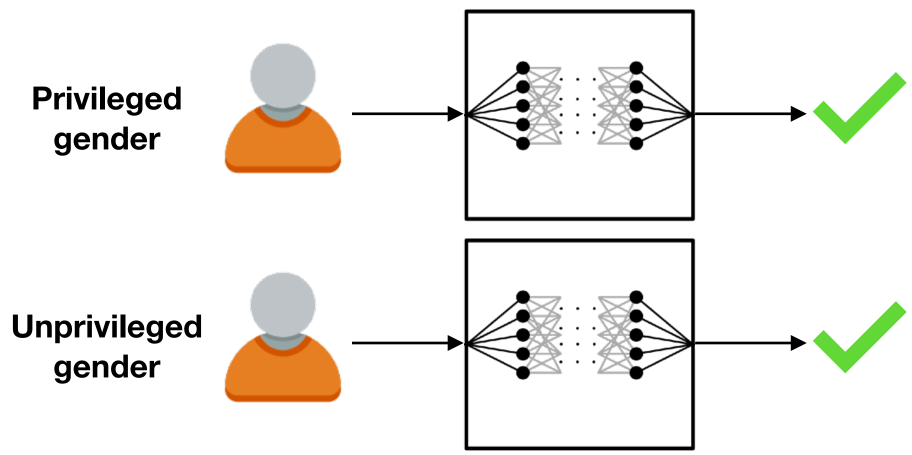
Lyapunov/Barrier certificate:

- Train a NN controller along with a stability/safety certificate.



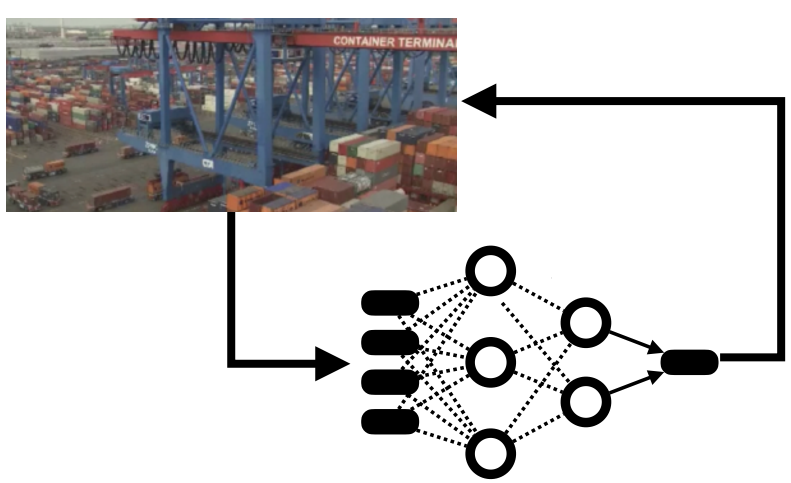
Adversarial robustness:

- The attacker can not fool the detector.
- The Out-of-Distribution Detector (OOD) is robust to bounded noise.



Fairness

- Similar individuals are to be treated similarly by the decision model

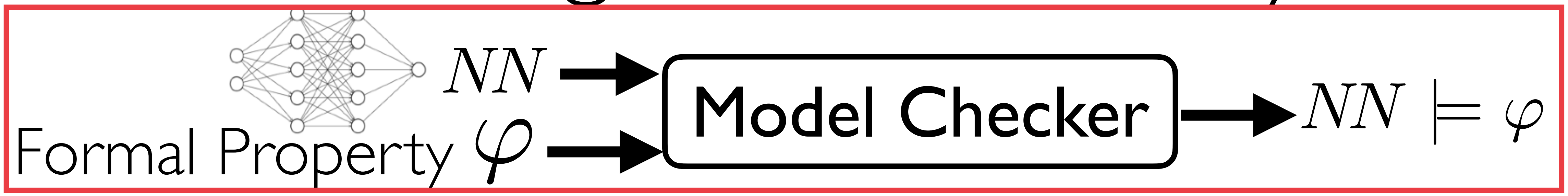


Lyapunov/Barrier certificate:

- Train a NN controller along with a stability/safety certificate.

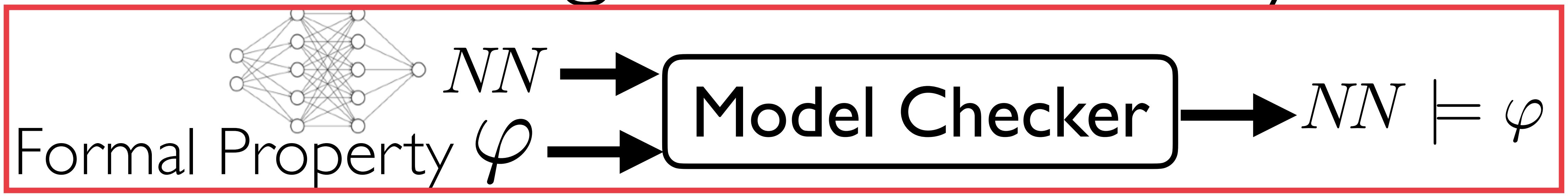
$$\left\{ x \in \mathbb{R}^{k_0} \mid x \in P_x \wedge \mathcal{N}\mathcal{N}(x) \notin P_y \wedge \left(\bigwedge_{\ell=1}^{m} h_\ell(x, \mathcal{N}\mathcal{N}(x)) \leq 0 \right) \right\} = \emptyset$$

Design-for-Verifiability



- Formal verification of NNs is NP-hard.

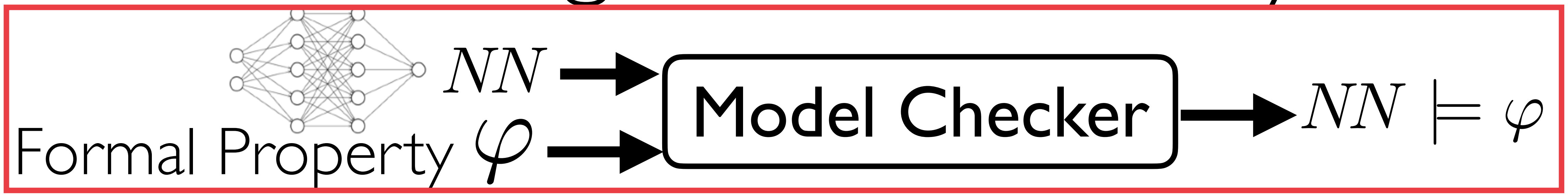
Design-for-Verifiability



- Formal verification of NNs is NP-hard.

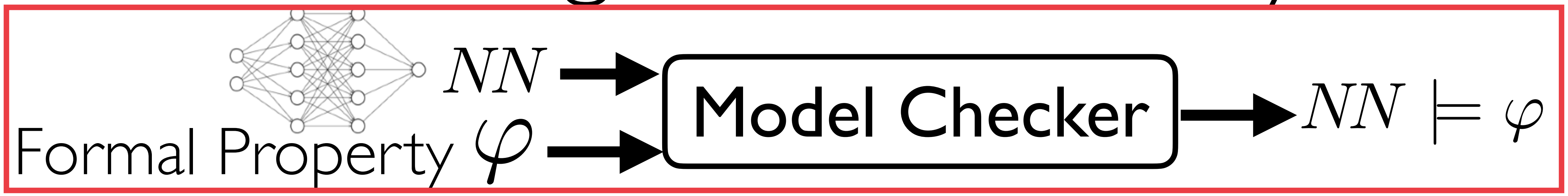
- Are all NNs "equally" hard?

Design-for-Verifiability



- Formal verification of NNs is NP-hard.
- Are all NNs "equally" hard?
- Can we find NNs with special structure/semantics that lead to "fast" verification?

Design-for-Verifiability



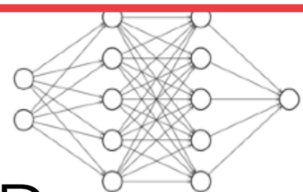
- Formal verification of NNs is NP-hard.

- Are all NNs "equally" hard?

- Can we find NNs with special structure/semantics that lead to "fast" verification?

- Can we replace the ReLU activation non-linearity with one that is amenable to "fast" verification?

Design-for-Verifiability



NN → Model Checker → $NN \models \varphi$

Formal Property φ →

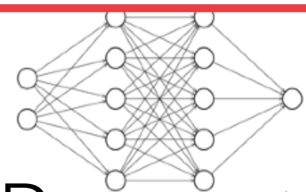
Verify **“easier” ReLU-NN architectures**
(NN structure/semantics)

Two-Level Lattice (TLL) NNs are **verifiable in polynomial time***

(* in the number of neurons)

J. Ferlez and Y. Shoukry, "Bounding the Complexity of Formally Verifying Neural Networks: A Geometric Approach," CDC 2021.

Design-for-Verifiability



NN → **Model Checker** → $NN \models \varphi$

Formal Property φ → **Model Checker**

Verify **"easier" ReLU-NN architectures**
(NN structure/semantics)

Two-Level Lattice (TLL) NNs are
verifiable in polynomial time*

(* in the number of neurons)

Theorem:
Any CPWA function can be rewritten as:

$$f(x) = \max_{1 \leq i \leq M} \min_{j \in s_i \subseteq \{1,\ldots,N\}} \ell_j(x)$$

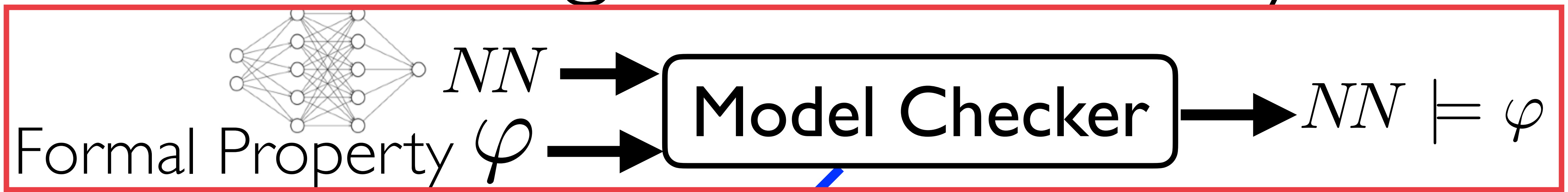
which is known as the two-level lattice representation.

J. M. Tarela and M.V. Martínez. Region configurations for realizability of lattice Piecewise-Linear models. Mathematical and Computer Modeling, 1999.

$N = \#$ local linear functions

$M = \#$ unique order regions

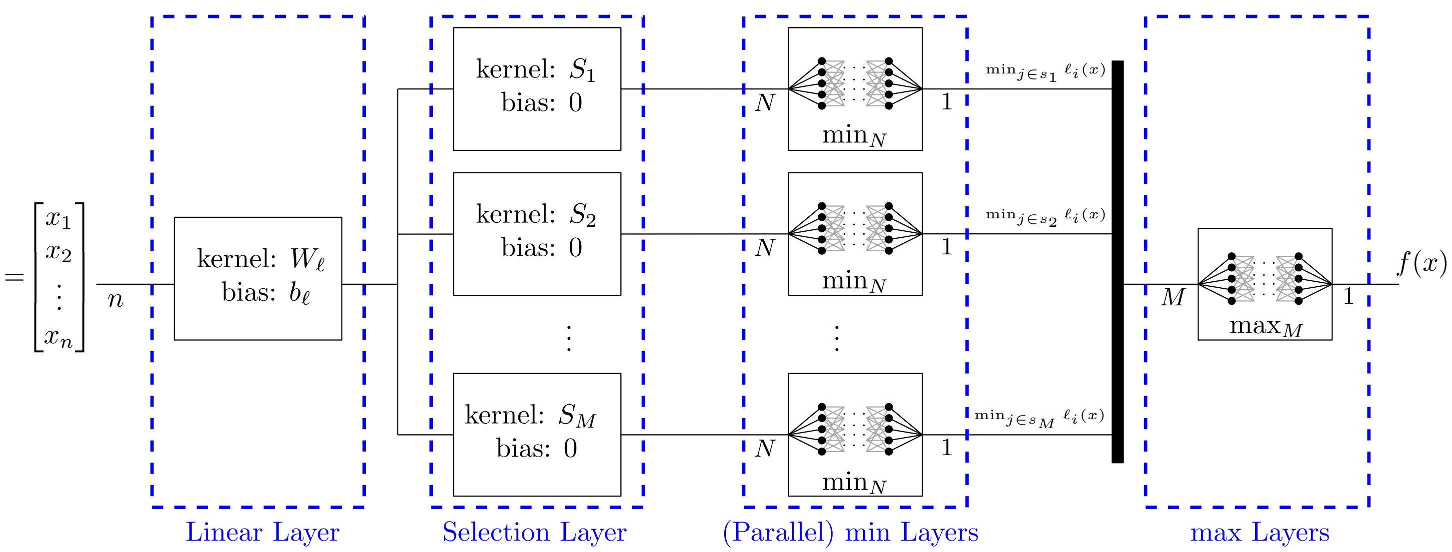
Design-for-Verifiability



Verify **"easier" ReLU-NN architectures** (NN structure/semantics)

Two-Level Lattice (TLL) NNs are **verifiable in polynomial time***

(* in the number of neurons)



Theorem:
Any CPWA function can be rewritten as:

$$f(x) = \max_{1 \leq i \leq M} \min_{j \in s_i \subseteq \{1,\ldots,N\}} \ell_j(x)$$

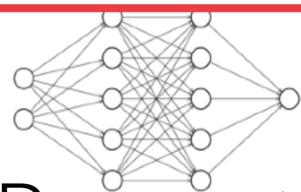
which is known as the two-level lattice representation.

J. M. Tarela and M.V. Martínez. Region configurations for realizability of lattice Piecewise-Linear models. Mathematical and Computer Modeling, 1999.

$N = \#$ local linear functions

$M = \#$ unique order regions

Design-for-Verifiability



NN → **Model Checker** → $NN \models \varphi$

Formal Property φ → **Model Checker**

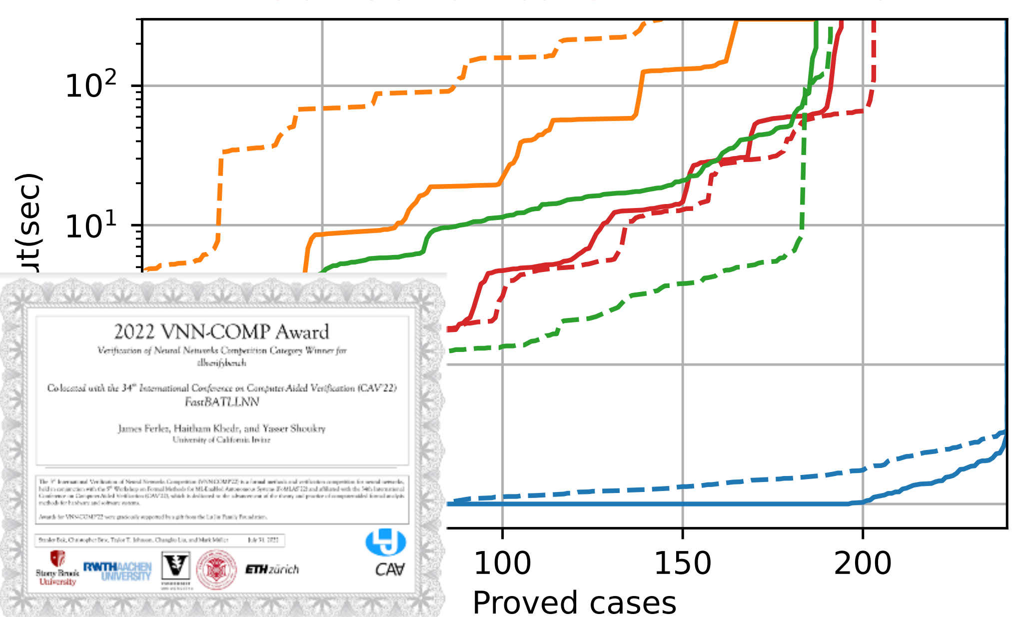
Verify **"easier" ReLU-NN architectures** (NN structure/semantics)

Two-Level Lattice (TLL) NNs are **verifiable in polynomial time***

(* in the number of neurons)

Verify **"structured" properties** (use NN structure/semantics)

FastBATLLNN: Fast Box-like constraints of TLL NNs



J. Ferlez and Y. Shoukry, "Bounding the Complexity of Formally Verifying Neural Networks: A Geometric Approach," CDC 2021.

J. Ferlez, H. Khedr, and Y. Shoukry, "FastBATLLNN: Fast Box Analysis of Two-Level Lattice Neural Networks," HSCC 2022.

Design-for-Verifiability

NN → **Model Checker** → $NN \models \varphi$

Formal Property φ → **Model Checker**

Verify **"easier" ReLU-NN architectures** (NN structure/semantics)

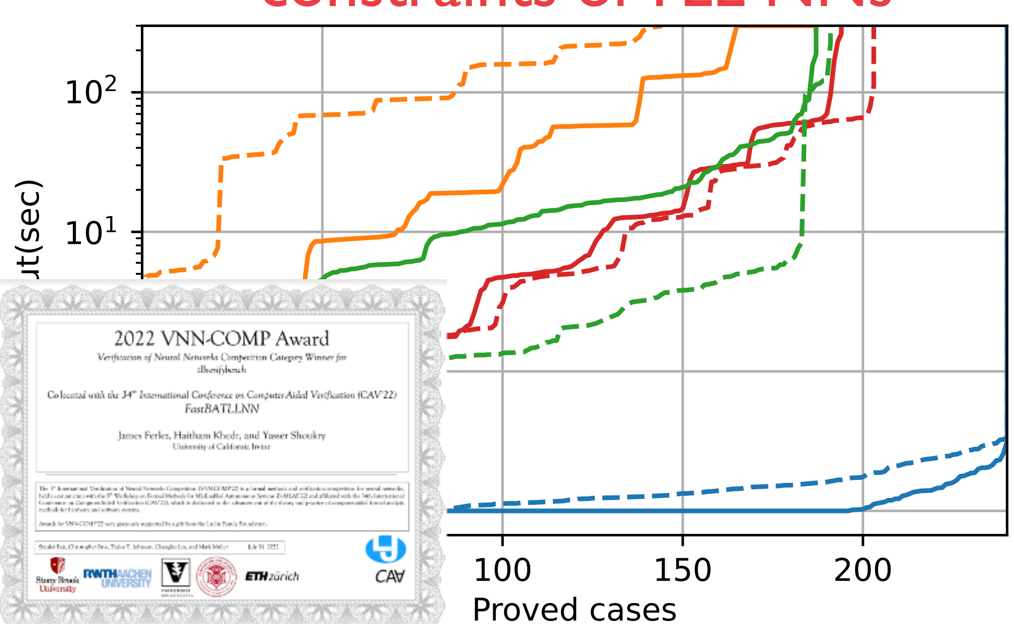
Verify **NNs with "easier" activation units** (use nice properties of other nonlinear functions)

Two-Level Lattice (TLL) NNs are **verifiable in polynomial time***

(* in the number of neurons)

Verify **"structured" properties** (use NN structure/semantics)

FastBATLLNNN: Fast Box-like constraints of TLL NNs



Design-for-Verifiability

NN → **Model Checker** → $NN \models \varphi$

Formal Property φ → **Model Checker**

Verify **"easier" ReLU-NN architectures** (NN structure/semantics)

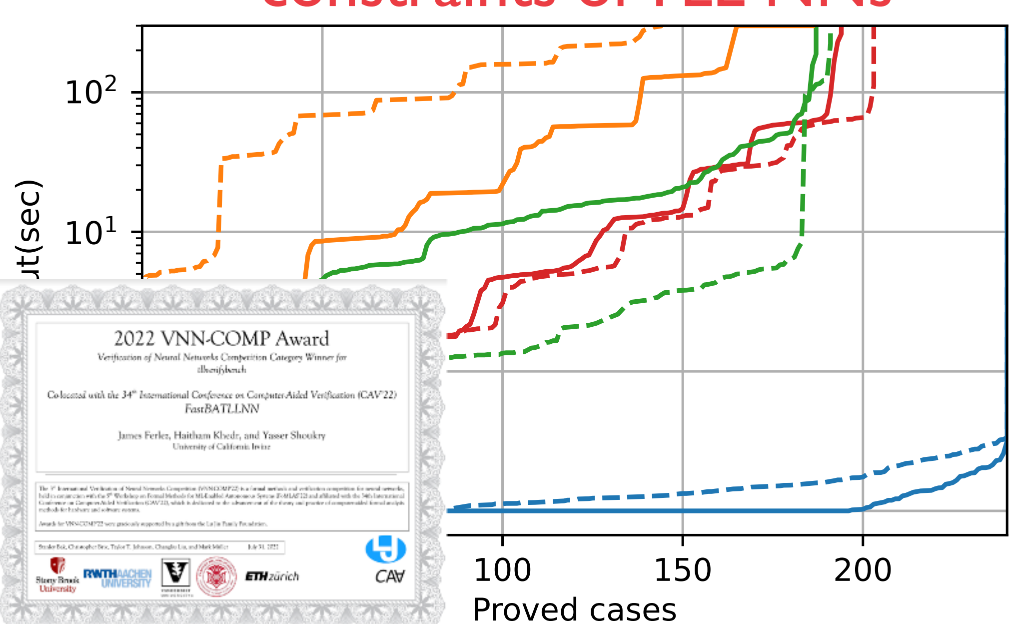
Verify NNs with **"easier" activation units** (use nice properties of other nonlinear functions)

Two-Level Lattice (TLL) NNs are **verifiable in polynomial time***

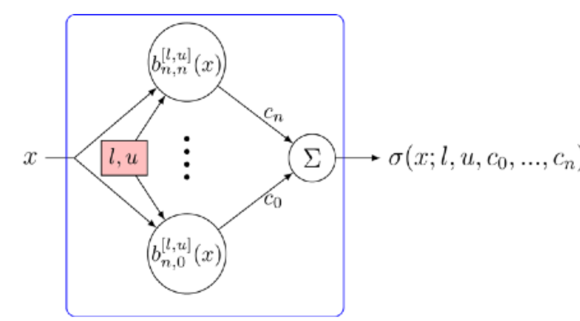
(* in the number of neurons)

Verify **"structured" properties** (use NN structure/semantics)

FastBATLLNNN: Fast Box-like constraints of TLL NNs



Bernstein Polynomials enjoy several "nice" properties (enclosure of range and subdivision)



$$\sigma_n^{[l,u]}(x) = \sum_{k=0}^{n} c_k b_{n,k}^{[l,u]}(x), \quad x \in [l, u],$$

$$b_{n,k}^{[l,u]}(x) = \frac{\binom{n}{k}}{(u-l)^n}(x-l)^k(u-x)^{n-k}$$

Design-for-Verifiability

NN → **Model Checker** → $NN \models \varphi$

Formal Property φ →

Verify **"easier" ReLU-NN architectures** (NN structure/semantics)

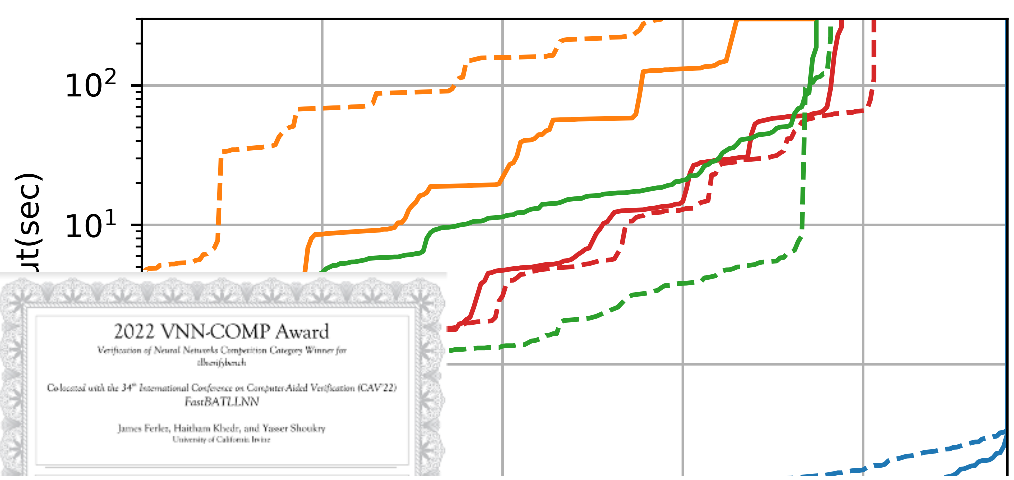
Verify NNs with **"easier" activation units** (use nice properties of other nonlinear functions)

Two-Level Lattice (TLL) NNs are **verifiable in polynomial time***

(* in the number of neurons)

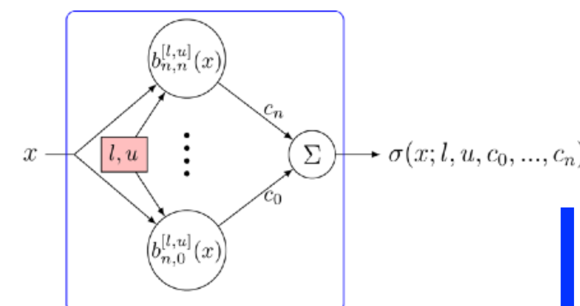
Verify **"structured" properties** (use NN structure/semantics)

FastBATLLNNN: Fast Box-like constraints of TLL NNs



H. Khedr and Y. Shoukry, "DeepBern-Nets: Taming the Complexity of Certifying Neural Networks using Bernstein Polynomial Activations and Precise Bound Propagation," AAAI 2024.

Bernstein Polynomials enjoy several "nice" properties (enclosure of range and subdivision)

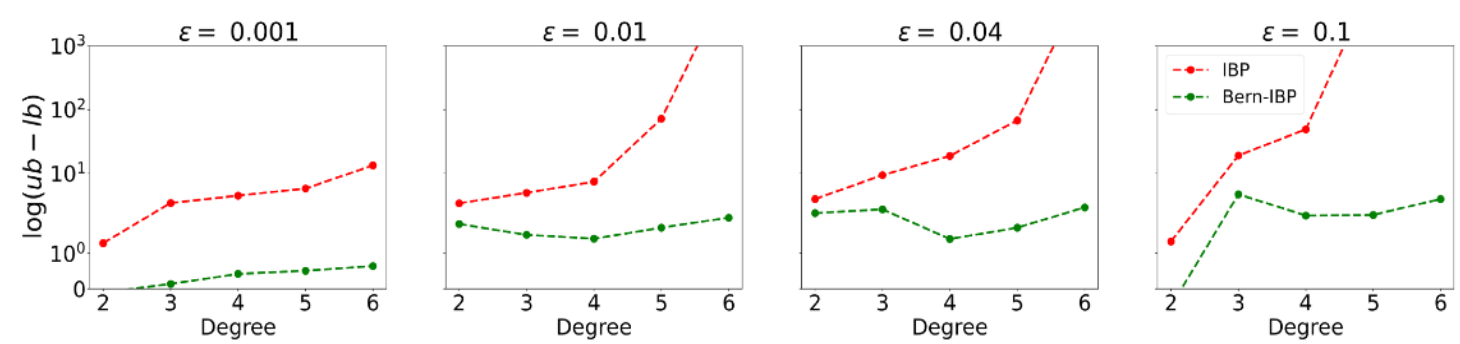


$$\sigma_n^{[l,u]}(x) = \sum_{k=0}^{n} c_k b_{n,k}^{[l,u]}(x), \quad x \in [l, u],$$

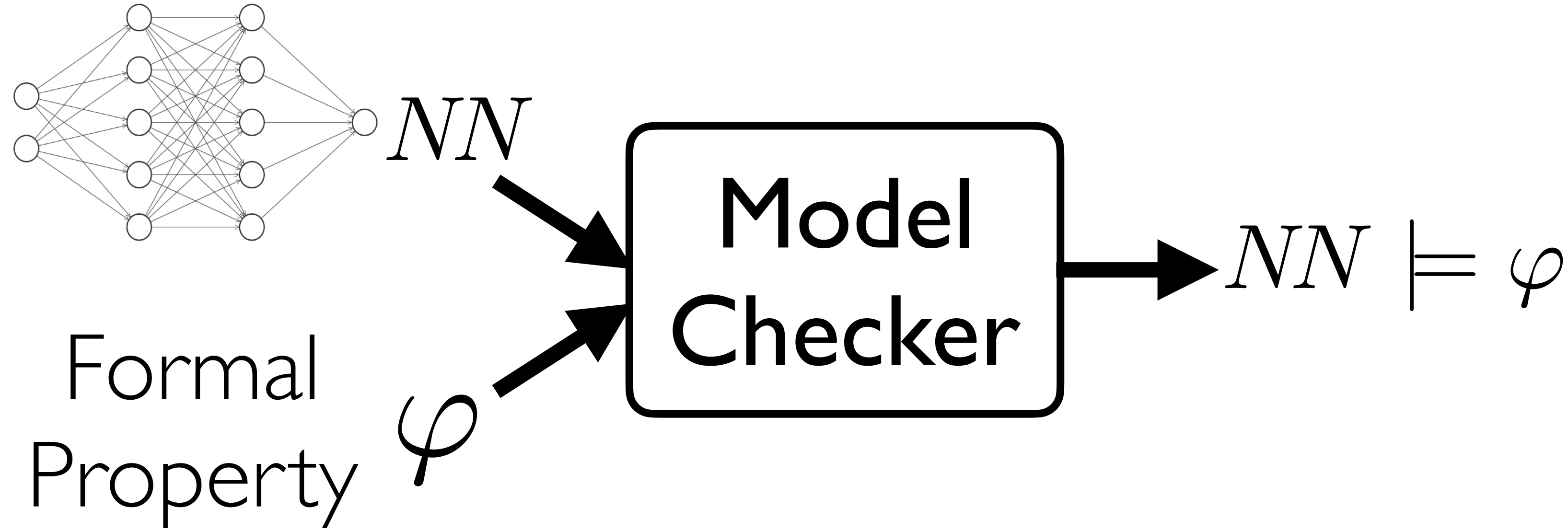
$$b_{n,k}^{[l,u]}(x) = \frac{\binom{n}{k}}{(u-l)^n}(x-l)^k(u-x)^{n-k}$$

Deep Bern-Nets = Precise Bound Propagation

Order	$\epsilon = 0.001$ IBP	$\epsilon = 0.001$ Bern-IBP	$\epsilon = 0.01$ IBP	$\epsilon = 0.01$ Bern-IBP	$\epsilon = 0.04$ IBP	$\epsilon = 0.04$ Bern-IBP	$\epsilon = 0.1$ IBP	$\epsilon = 0.1$ Bern-IBP
2	-20.16	-16.63	-42.72	-16.56	-83.7	-22.22	-71.33	-8.25
3	-96.55	-12.16	-205.09	-14.02	-34962.84	-22.91	-2302369792	-137.07
4	-3550.07	-10.15	-56758.56	-13.72	-1.09065E+15	-9.23	-8.24695E+24	-23.03
5	-1345.89	-11.78	-2.2861E+35	-12.93	-inf	-8.68	-inf	-18.11
6	-109130.05	-12.24	-inf	-17.03	-inf	-30.47	-inf	-72.53



Formal Verification Tools for NN Analysis

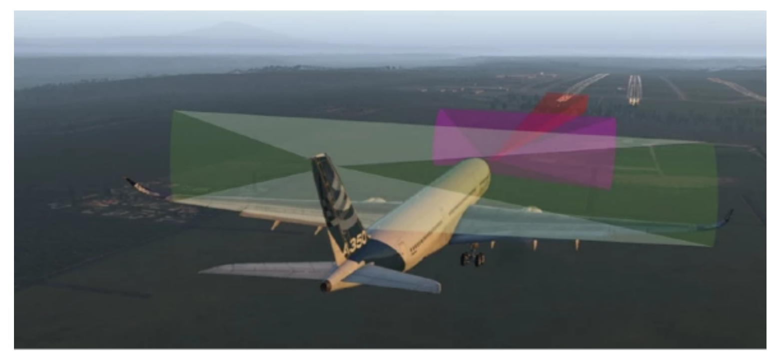


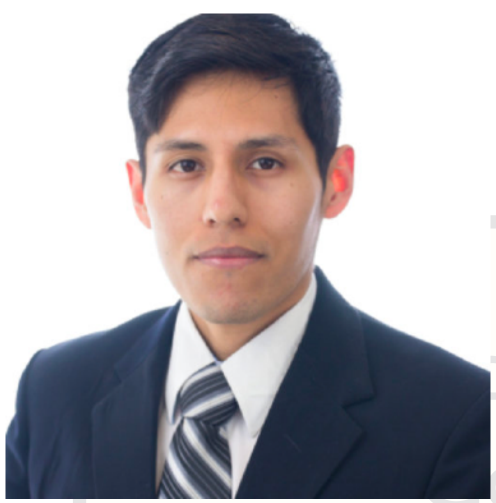
Assured NN-based Perception

Assured NN-based Control

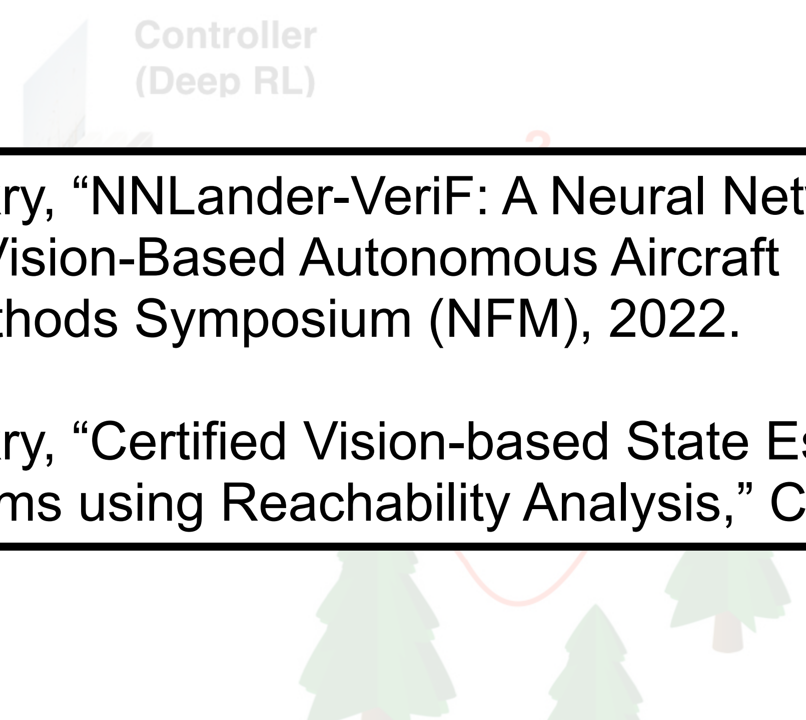
Formal Verification Tools for NN Analysis

Assured NN-based Perception





Ulices Santa Cruz Leal

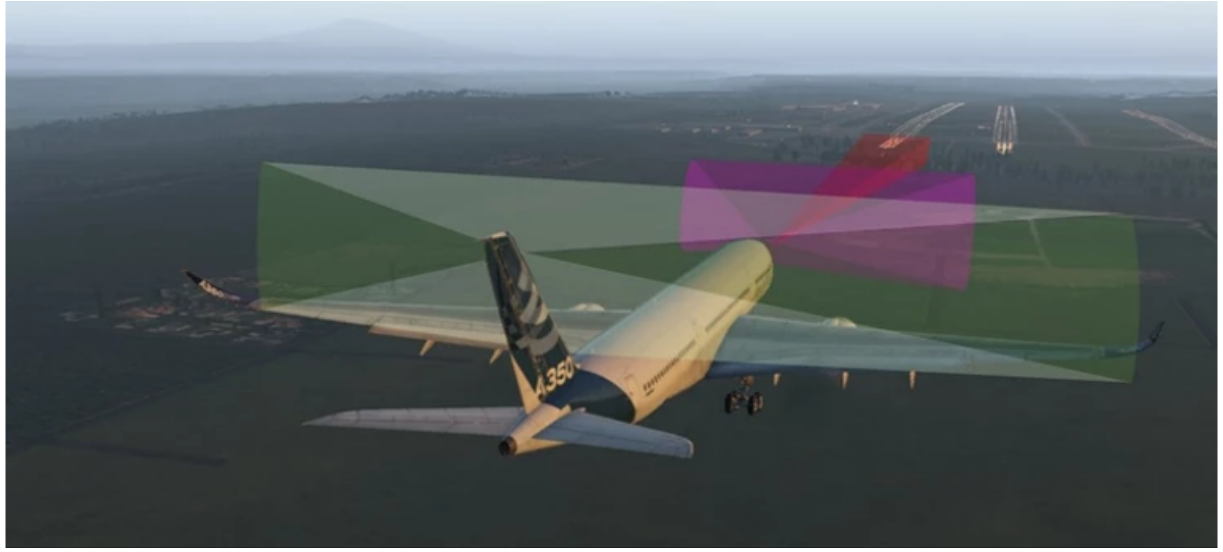


U. Santa Cruz and Y. Shoukry, “NNLander-VeriF: A Neural Network Formal Verification Framework for Vision-Based Autonomous Aircraft Landing,” NASA Formal Methods Symposium (NFM), 2022.

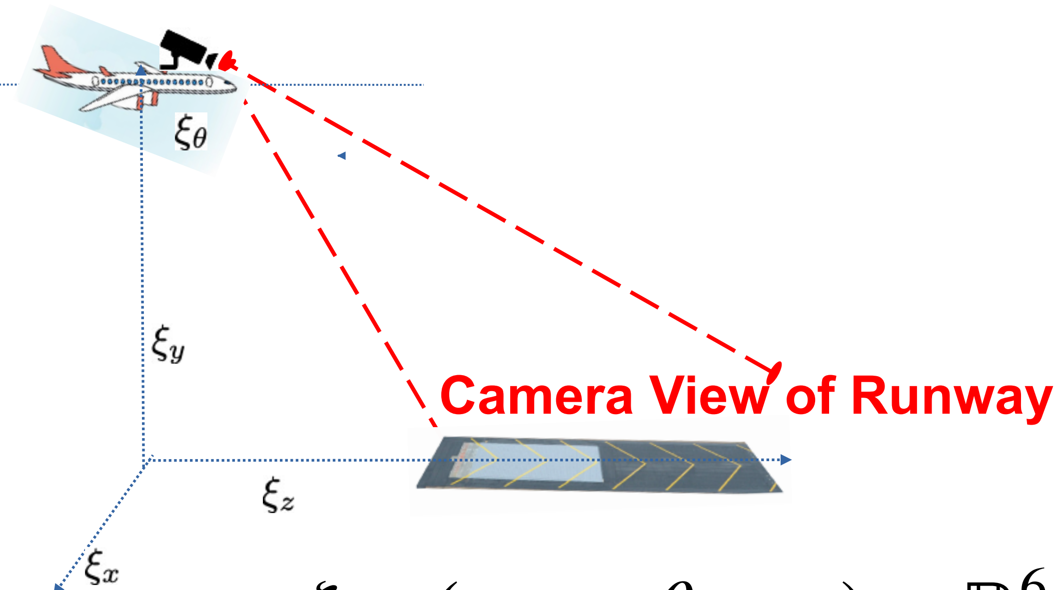
U. Santa Cruz and Y. Shoukry, “Certified Vision-based State Estimation for Autonomous Landing Systems using Reachability Analysis,” CDC 2023.

Assured NN-based Perception

- Can we train NNs with provable guarantees in terms of:
  - Ability to detect certain objects?
  - Ability to estimate the location of these objects?

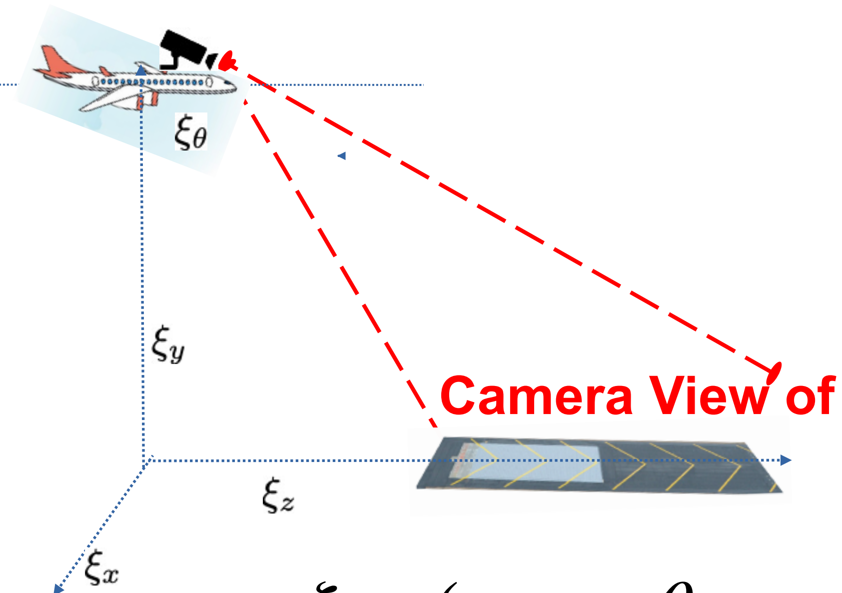


Assured NN-based Perception



$$\xi = (x, y, z, \theta, \psi, \rho) \in \mathbb{R}^6$$

Assured NN-based Perception

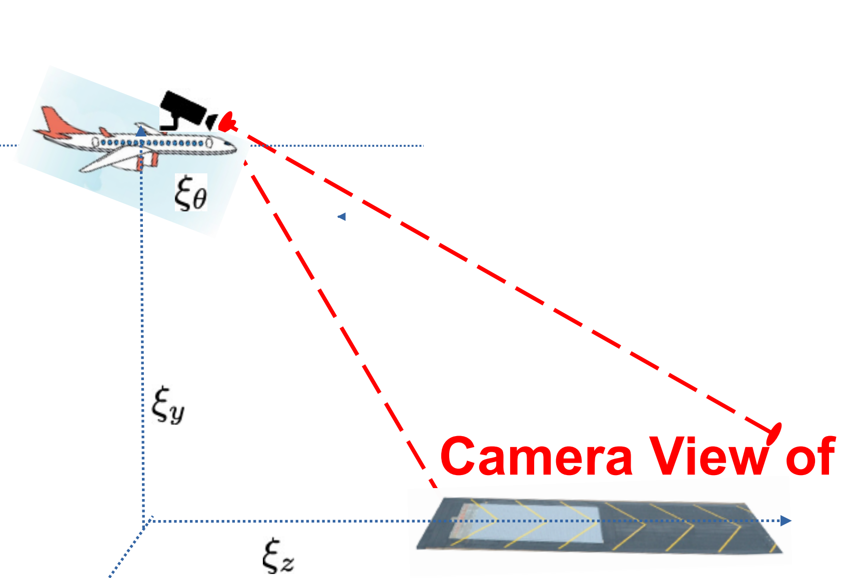


$$\xi = (x, y, z, \theta, \psi, \rho) \in \mathbb{R}^6$$

Image from Camera

$$I(\xi) \in \{0, 1\}^{a \times b}$$

Assured NN-based Perception



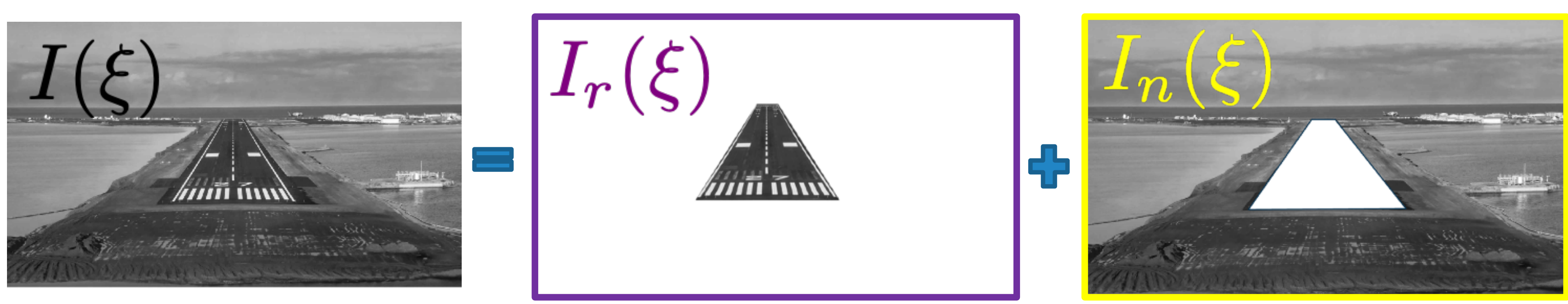
$$\xi = (x, y, z, \theta, \psi, \rho) \in \mathbb{R}^6$$

Image from Camera

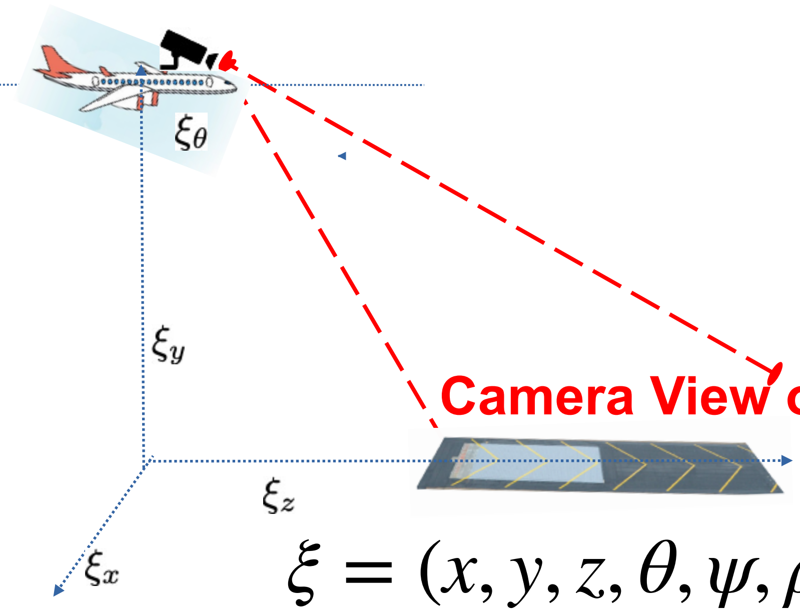
$$I(\xi) \in \{0,1\}^{a \times b}$$

$$I(\xi) = I_r(\xi) + I_n(\xi)$$

Original Image = **Image of Runway** + **Image of Other Objects**



Assured NN-based Perception



Camera View of Runway

$$\xi = (x, y, z, \theta, \psi, \rho) \in \mathbb{R}^6$$

Image from Camera

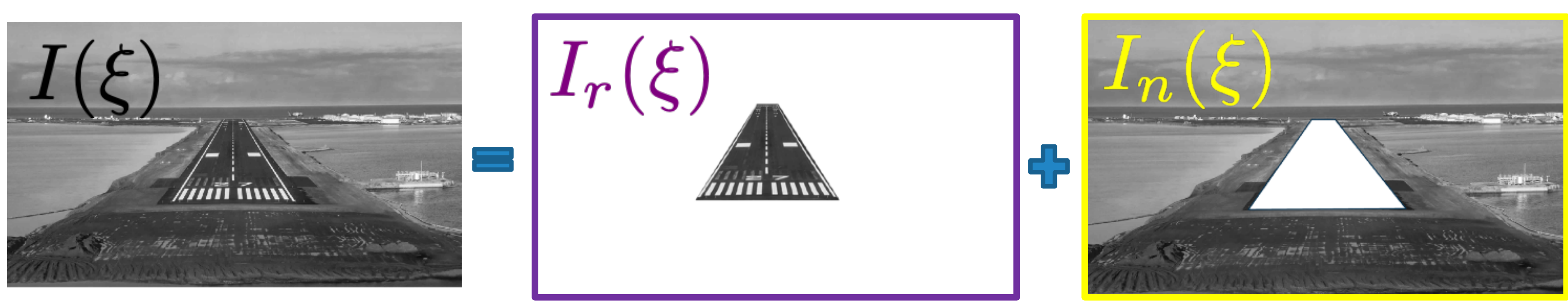
$$I(\xi) \in \{0,1\}^{a \times b}$$

State Estimate

$$\hat{\xi} = (\hat{x}, \hat{y}, \hat{z}, \hat{\theta}, \hat{\psi}, \hat{\rho}) \in \mathbb{R}^6$$

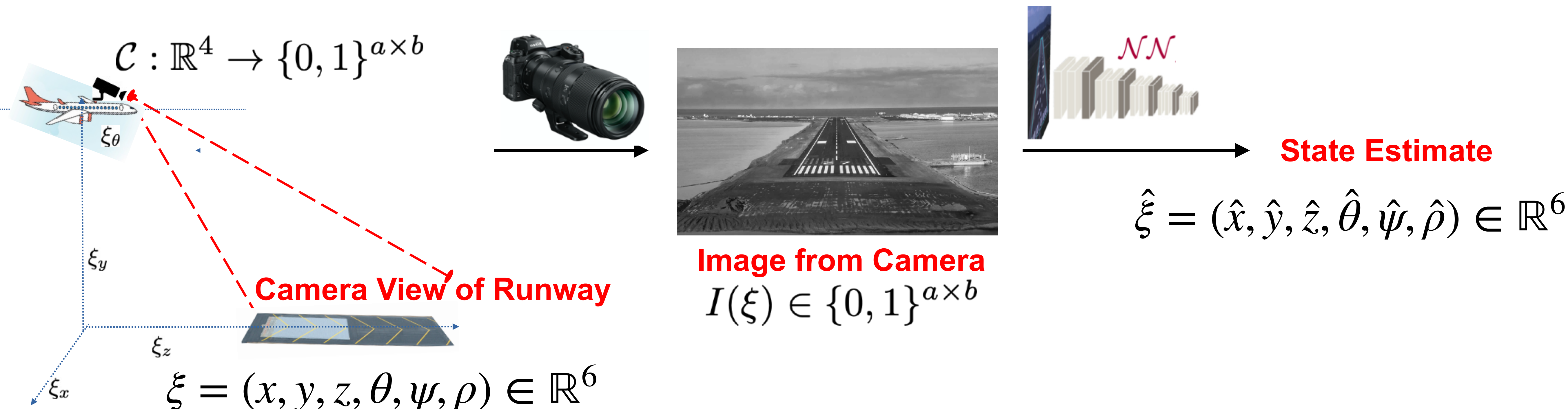
$$I(\xi) = I_r(\xi) + I_n(\xi)$$

Original Image = **Image of Runway** + **Image of Other Objects**



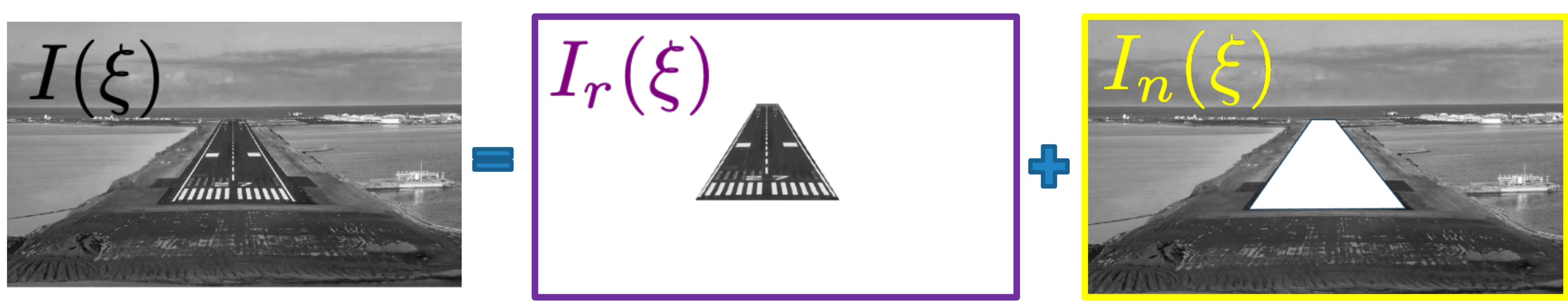
CPS Lab Resilient

Assured NN-based Perception



$$I(\xi) = I_r(\xi) + I_n(\xi)$$

Original Image = Image of Runway + Image of Other Objects



Given: A camera image $I(\xi) = I_r(\xi) + I_n(\xi)$

Given: User defined error $\epsilon > 0$

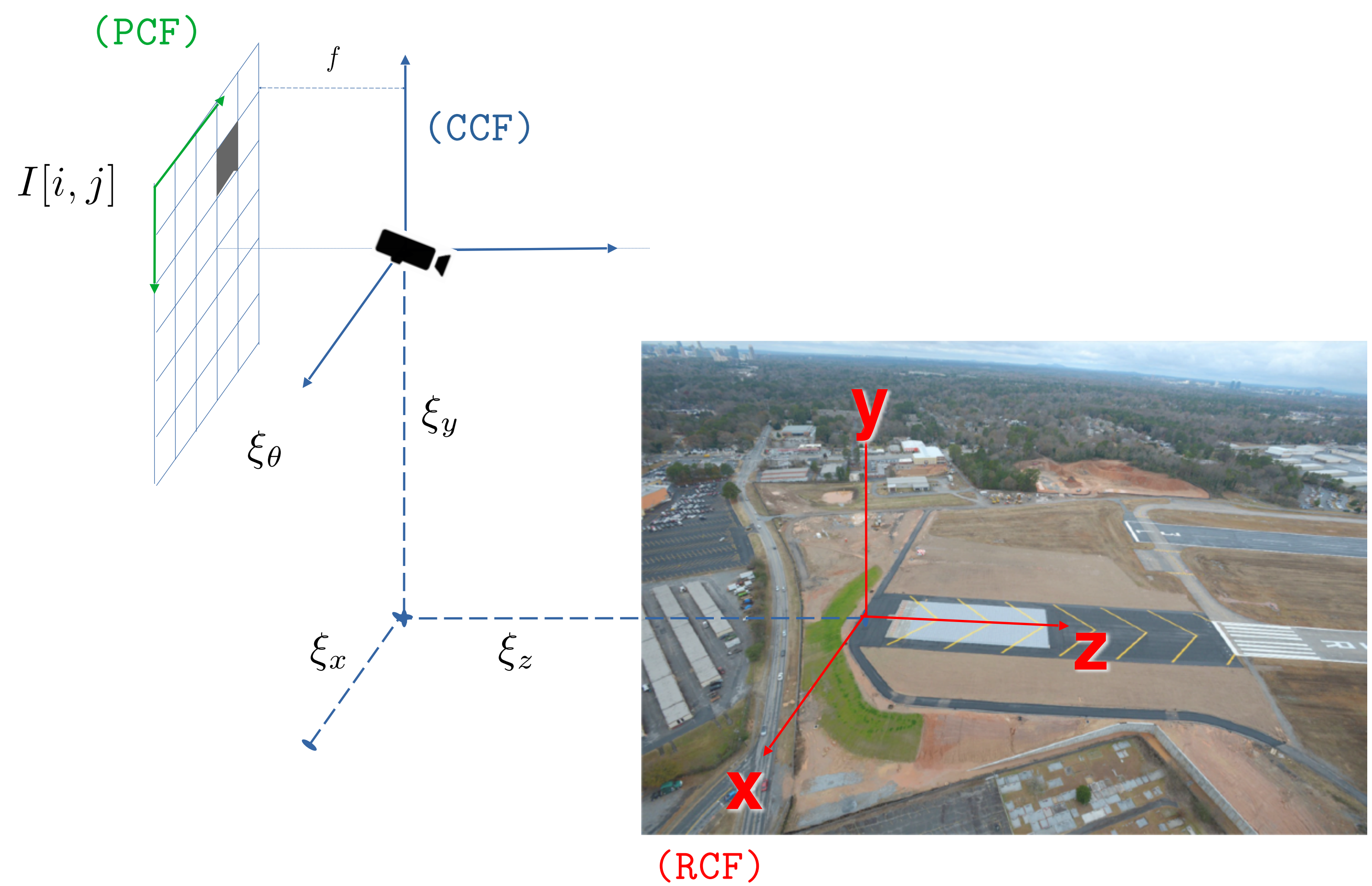
Design: NN Estimator $\hat{\xi} = \mathcal{NN}(I)$ such that $||\xi - \hat{\xi}|| \leq \epsilon$

Assured perception

Assured NN-based Perception

Image Formation Process

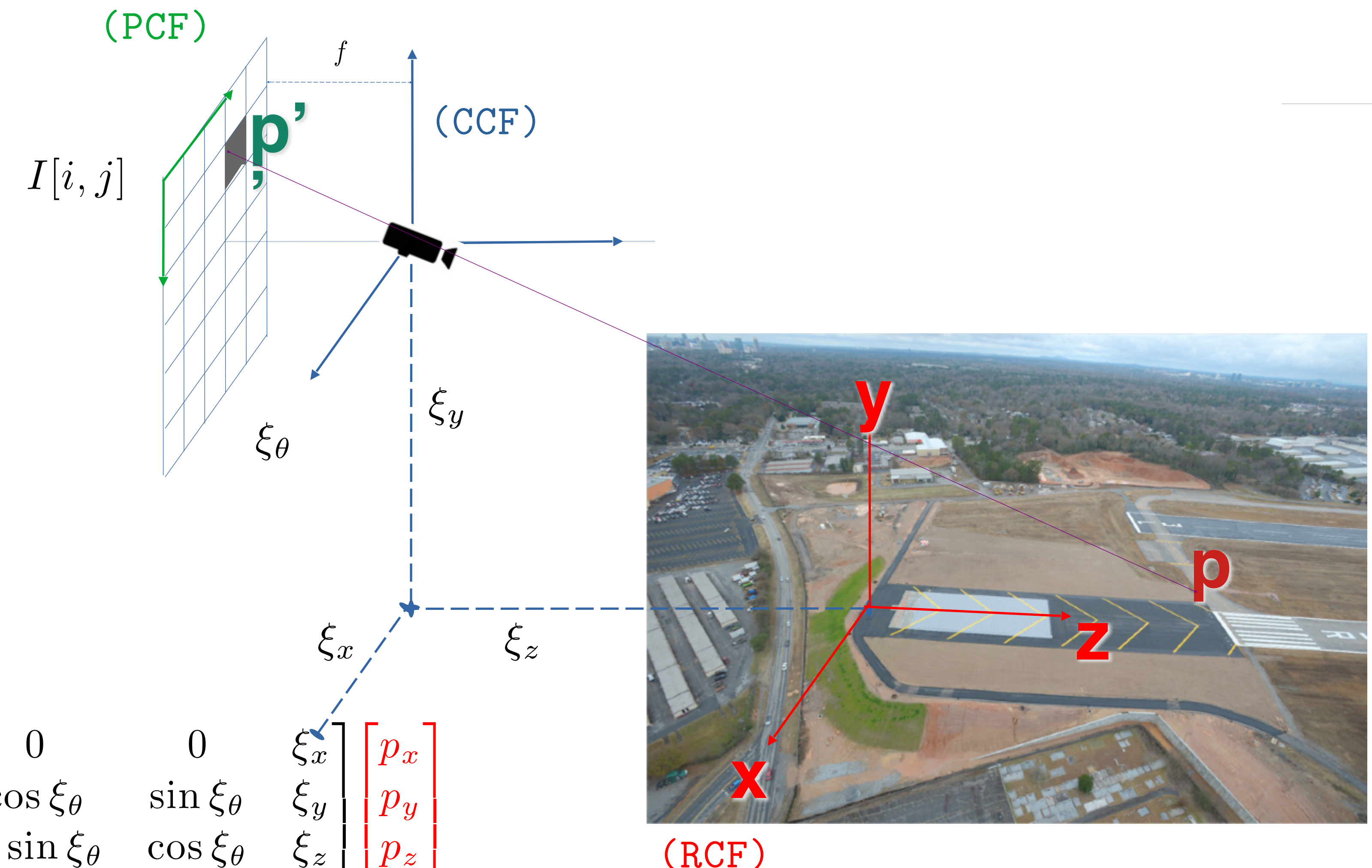
$C : \mathbb{R}^6 \rightarrow \{0,1\}^{a \times b}$



Assured NN-based Perception

Image Formation Process

$C : \mathbb{R}^6 \rightarrow \{0,1\}^{a \times b}$



$$\begin{bmatrix} q_{x_{\text{PCF}}} \\ q_{y_{\text{PCF}}} \\ q_{z_{\text{PCF}}} \end{bmatrix} = \begin{bmatrix} \rho_w & 0 & u_0 \\ 0 & -\rho_h & v_0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} f & 0 & 0 & 0 \\ 0 & f & 0 & 0 \\ 0 & 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 & \xi_x \\ 0 & \cos\xi_\theta & \sin\xi_\theta & \xi_y \\ 0 & -\sin\xi_\theta & \cos\xi_\theta & \xi_z \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} p_x \\ p_y \\ p_z \\ 1 \end{bmatrix}$$

p

$$\text{p''} = (p''_{x_{\text{PCF}}}, p''_{y_{\text{PCF}}}) = \left(\left\lfloor \frac{q_{x_{\text{PCF}}}}{q_{z_{\text{PCF}}}} \right\rfloor, \left\lfloor \frac{q_{y_{\text{PCF}}}}{q_{z_{\text{PCF}}}} \right\rfloor \right)$$

$$I[i,j] = \begin{cases} 1 & (p''_{x_{\text{PCF}}} == i-1) \wedge (p''_{y_{\text{PCF}}} == j-1) \wedge \text{visible} \\ 0 & \text{otherwise} \end{cases}$$

$$\text{visible} = \begin{cases} \text{yes} & |p''_{x_{\text{PCF}}}| \le \frac{W}{2} \vee |p''_{y_{\text{PCF}}}| \le \frac{H}{2} \\ \text{no} & \text{otherwise} \end{cases}$$

f: focal length

ρ_w, ρ_h: pixels/image size

u_0, v_0: image size scale

$$\rho_w = \frac{\text{WP}}{\text{W}} \qquad \rho_h = \frac{\text{HP}}{\text{H}}$$

Assured NN-based Perception

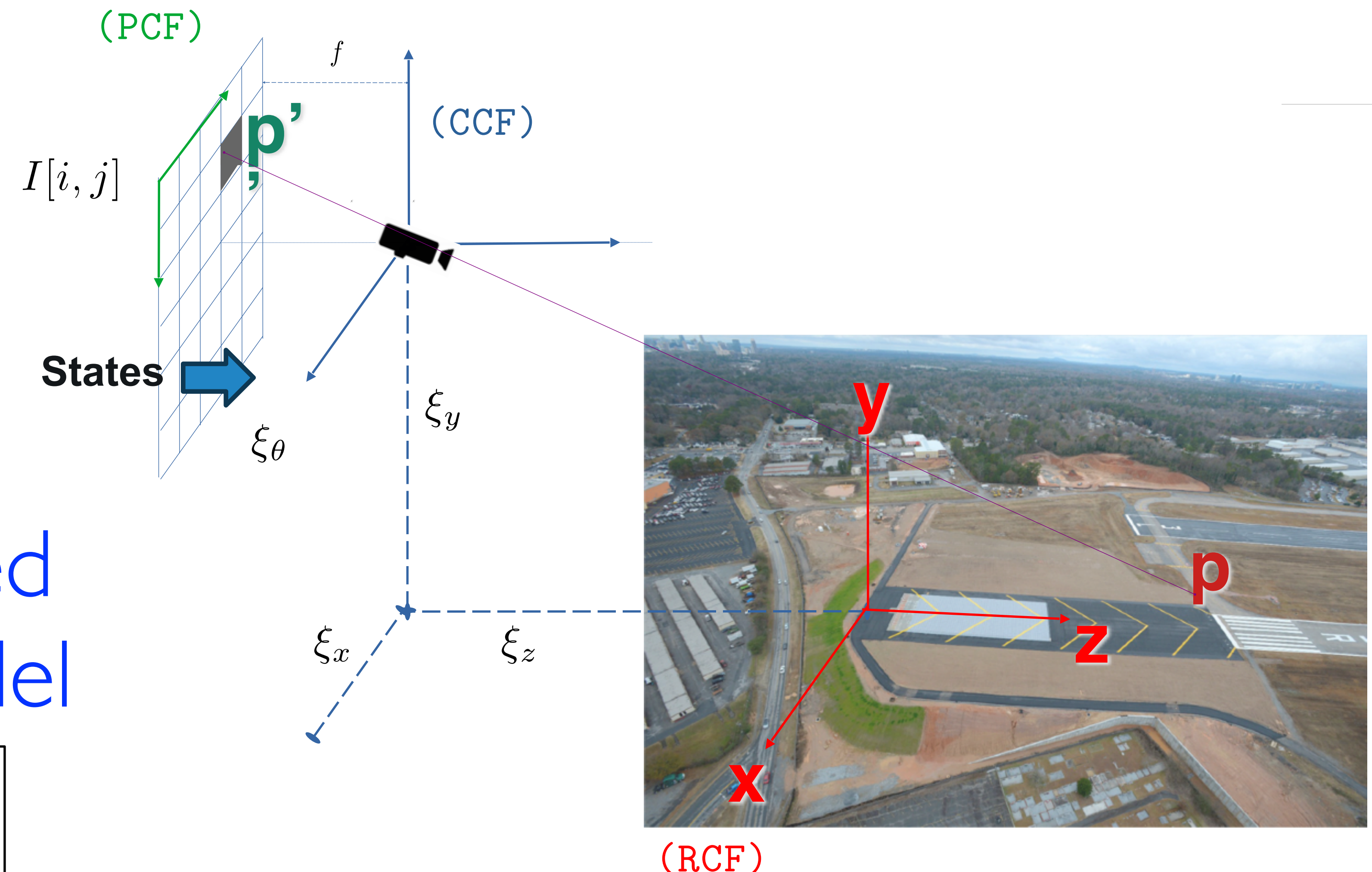
Santa Cruz, U., Shoukry, Y. (2021). NNLander-VeriF: A Neural Network Formal Verification Framework for Vision-Based Autonomous Aircraft Landing . In: NASA Formal Methods Conference

Image Formation Process

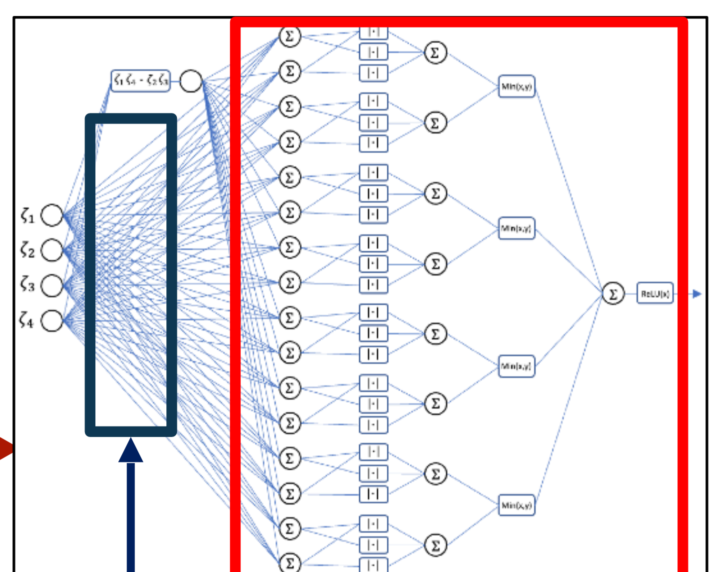
$C : \mathbb{R}^6 \rightarrow \{0,1\}^{a \times b}$

=

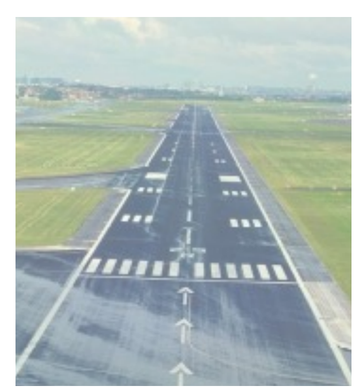
Geometry-based Generative Model



Position, angles ξ



Image



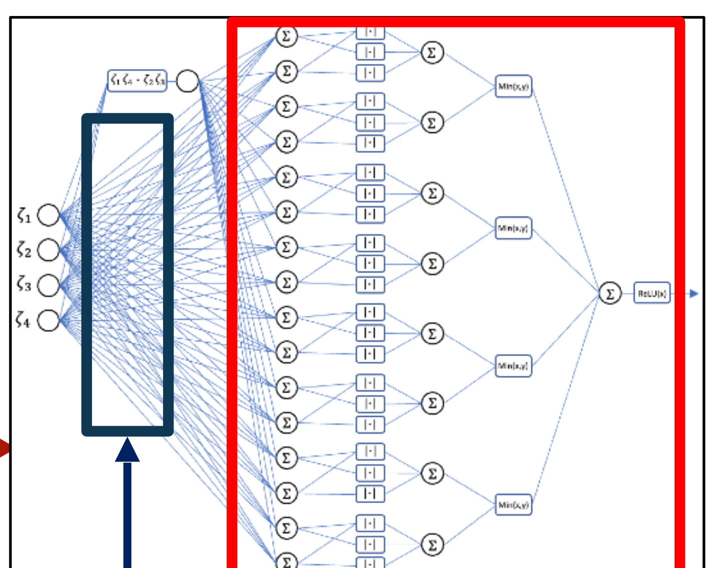
Physical Parameters (Trainable)

Geometrical Parameters (Non-Trainable)

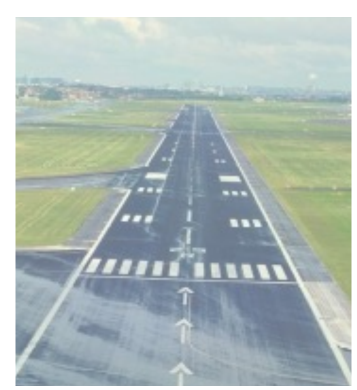
Assured NN-based Perception

Geometry-based Generative Model

Position, angles ξ



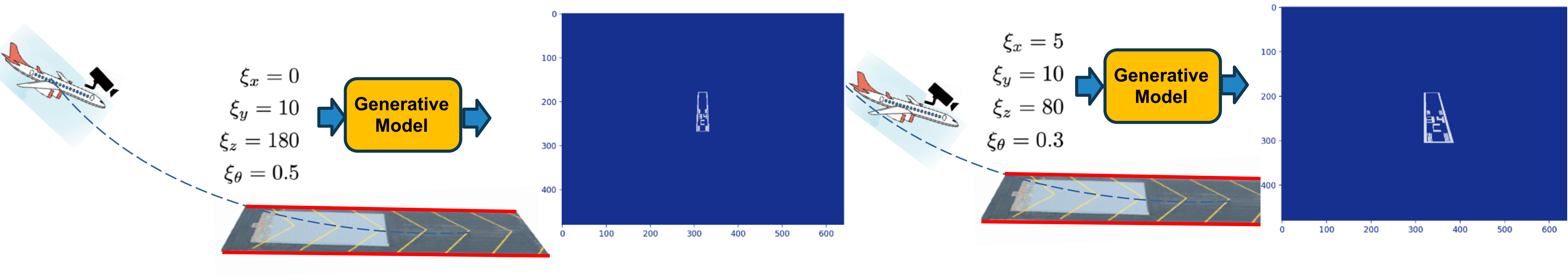
Image

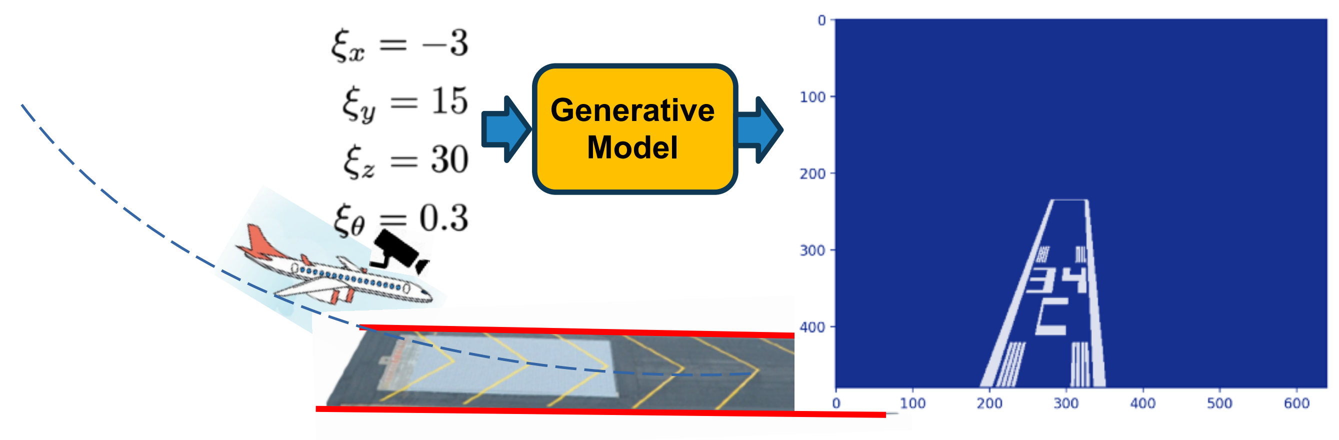


Physical Parameters (Trainable)

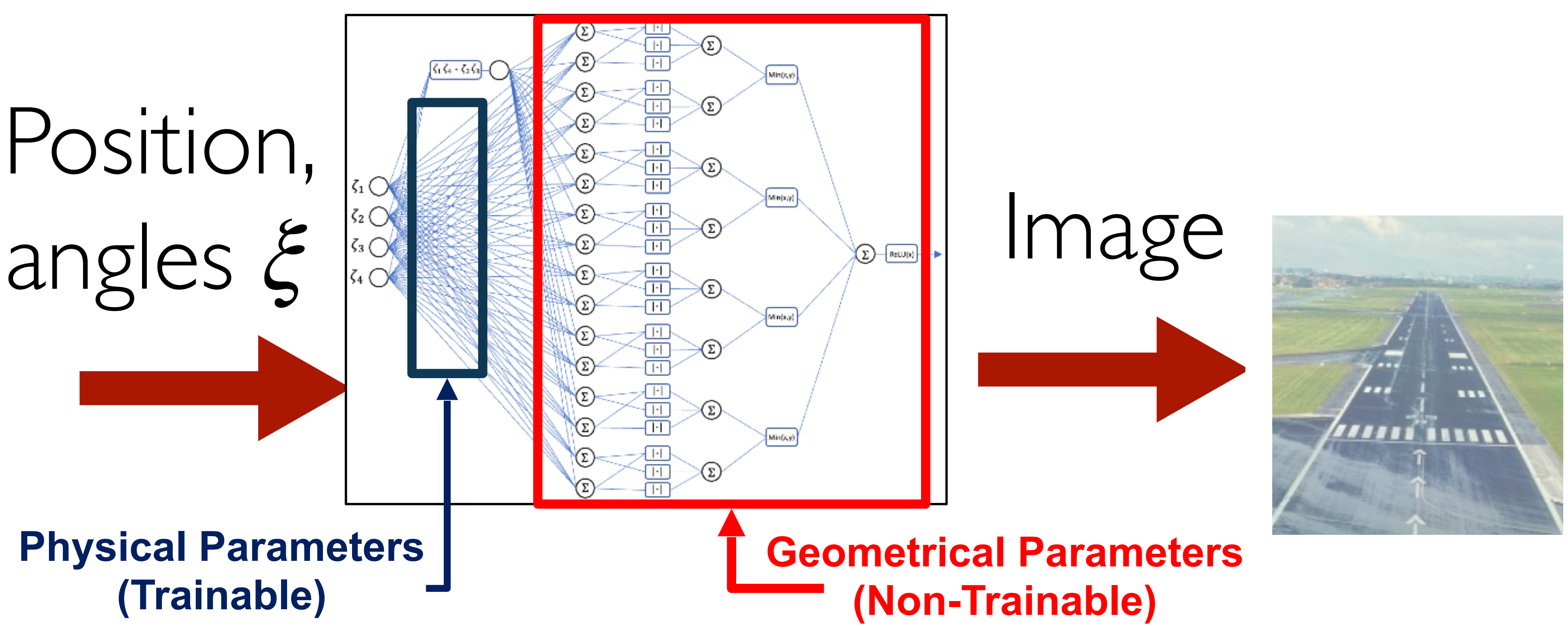
Geometrical Parameters (Non-Trainable)

Assured NN-based Perception

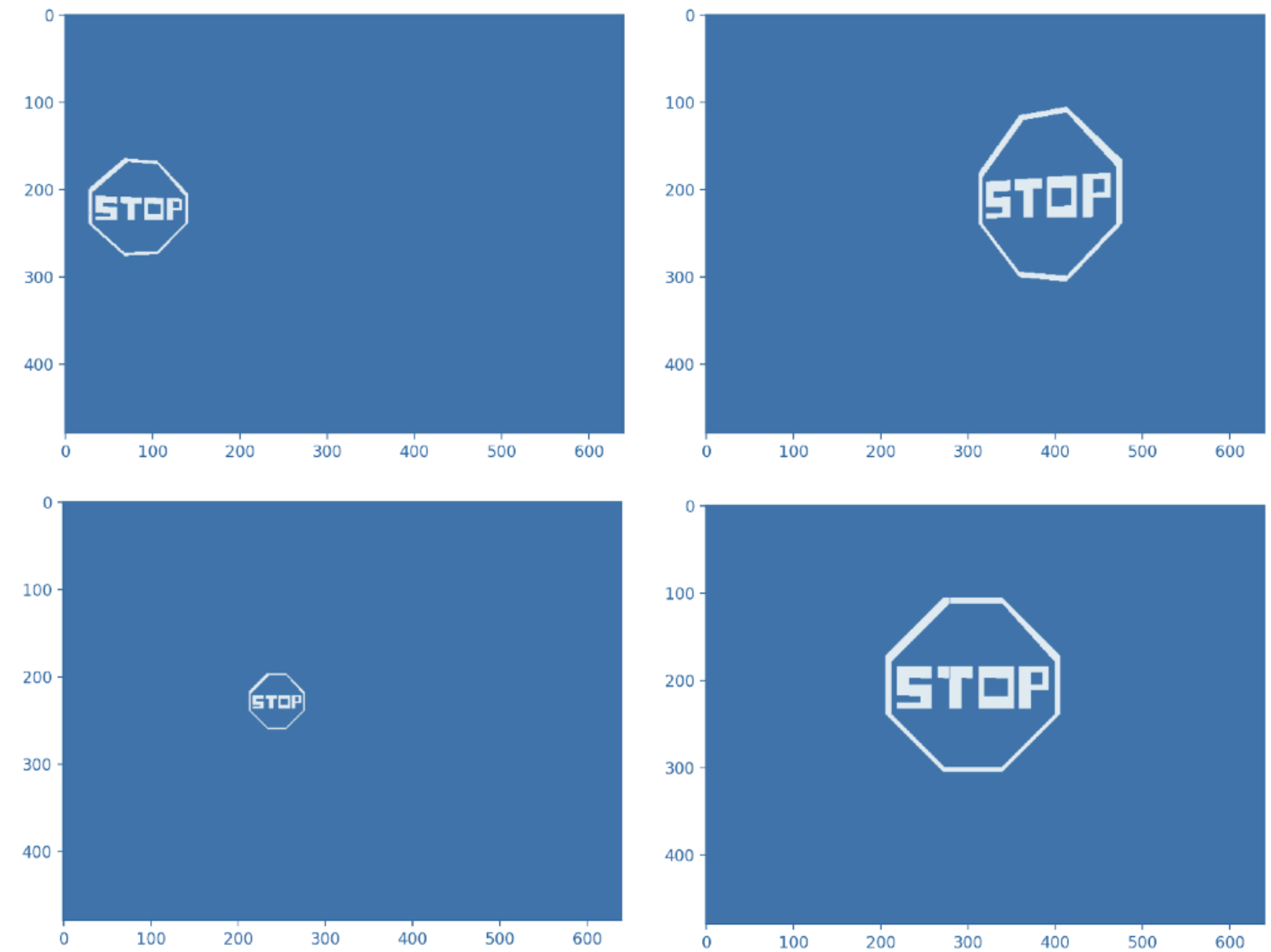




Geometry-based Generative Model

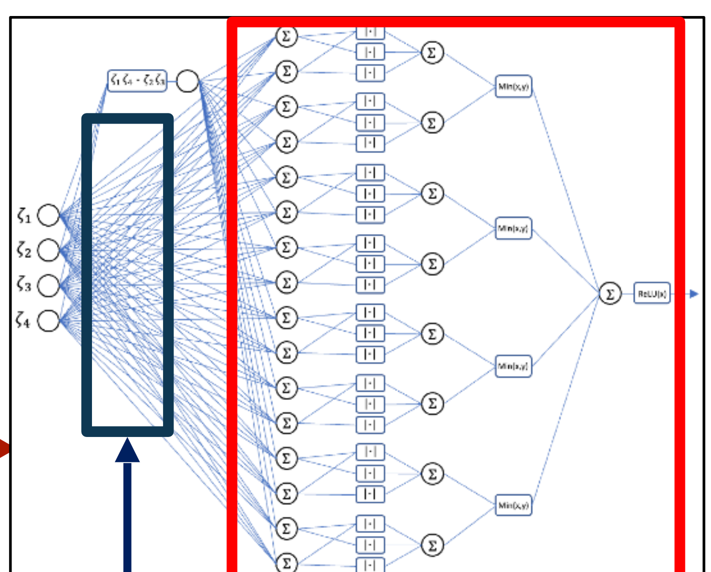


Assured NN-based Perception



Geometry-based Generative Model

Position, angles ξ



Physical Parameters (Trainable)

Geometrical Parameters (Non-Trainable)

Image

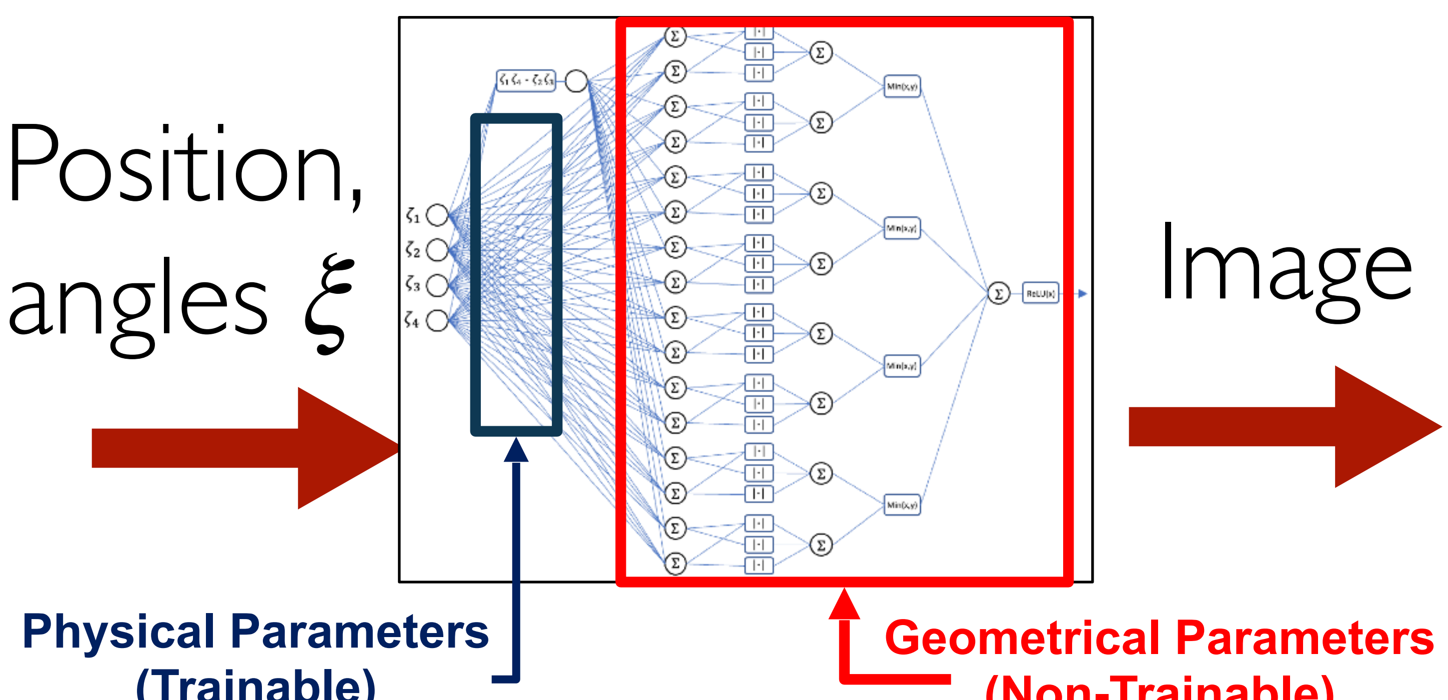
Assured NN-based Perception

Theorem (Informal Version)

For any 2D object that can be formed as unions and intersection of polytopes, then the Geometry-based Generative Model (GGM) Neural Network is equivalent to the Pin-hole camera model, i.e.,

$$I_o(\xi) = GGM_o(\xi)$$

Geometry-based Generative Model



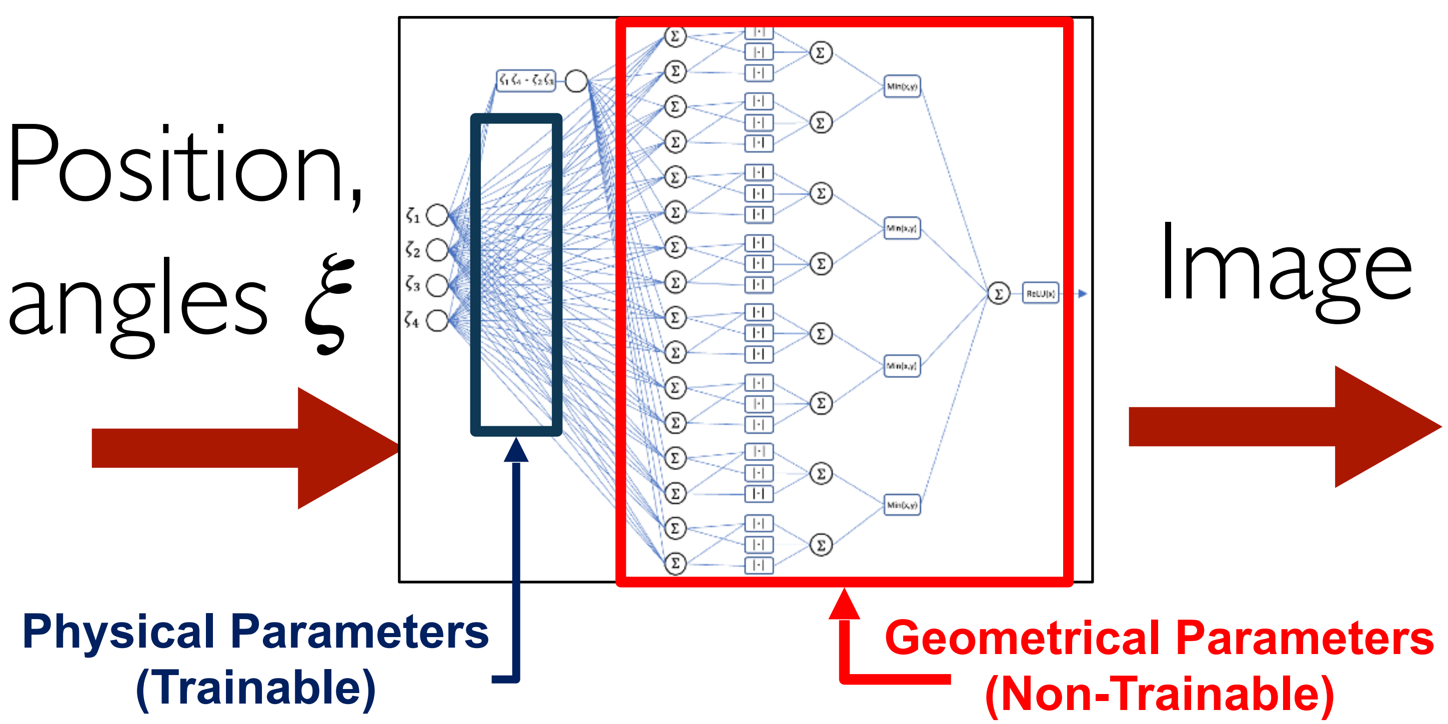
Assured NN-based Perception

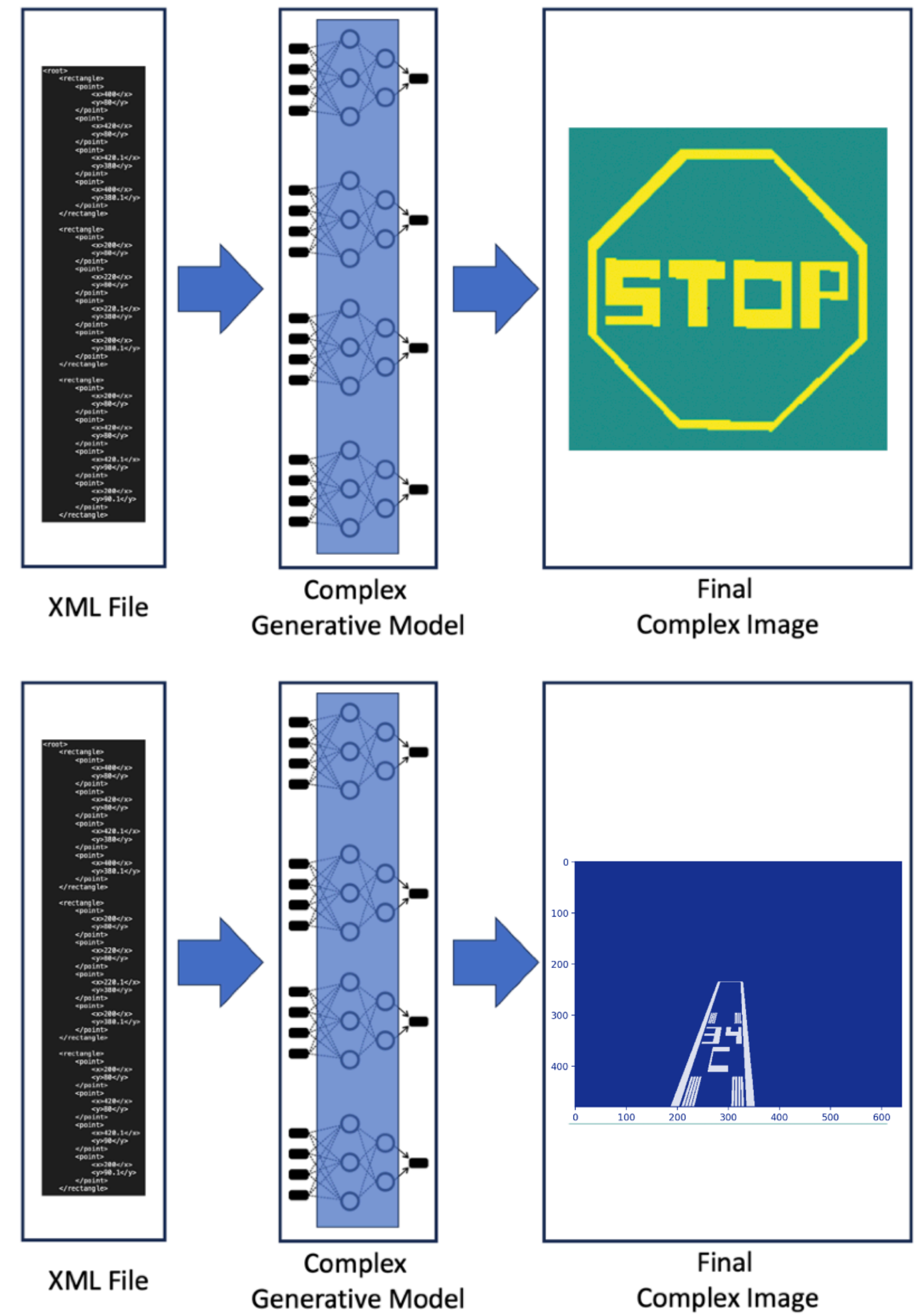
Theorem (Informal Version)

For any 2D object that can be formed as unions and intersection of polytopes, then the Geometry-based Generative Model (GGM) Neural Network is equivalent to the Pin-hole camera model, i.e.,

$$I_o(\xi) = GGM_o(\xi)$$

Geometry-based Generative Model





Assured NN-based Perception

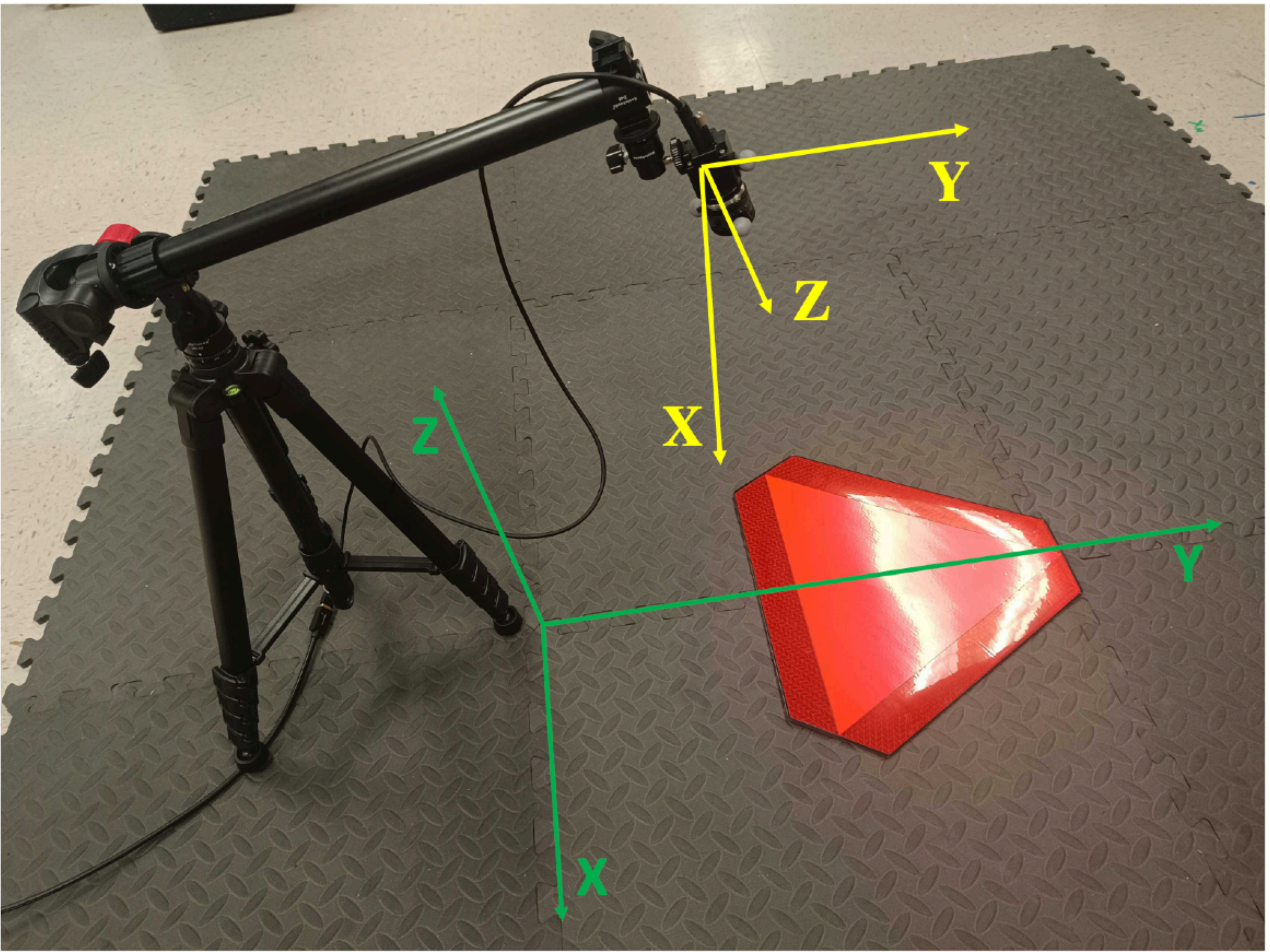
Theorem (Informal Version)

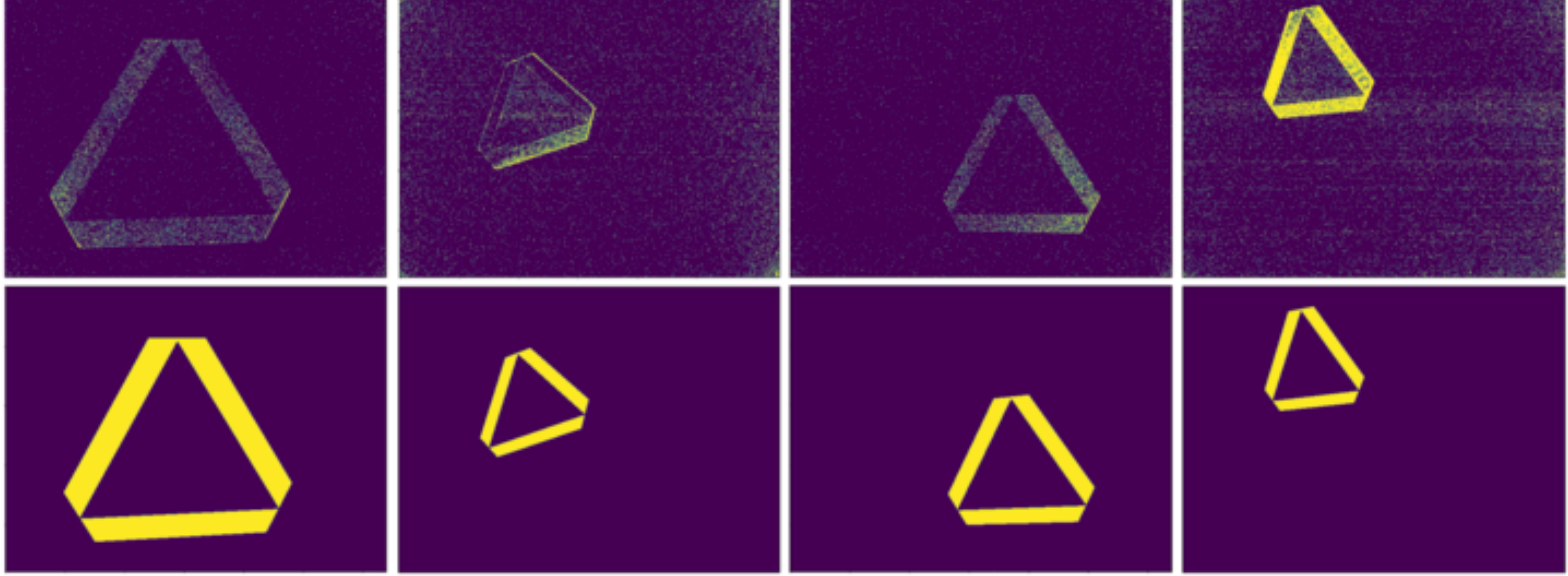
For any 2D object that can be formed as unions and intersection of polytopes, then the Geometry-based Generative Model (GGM) Neural Network is equivalent to the Pin-hole camera model, i.e.,

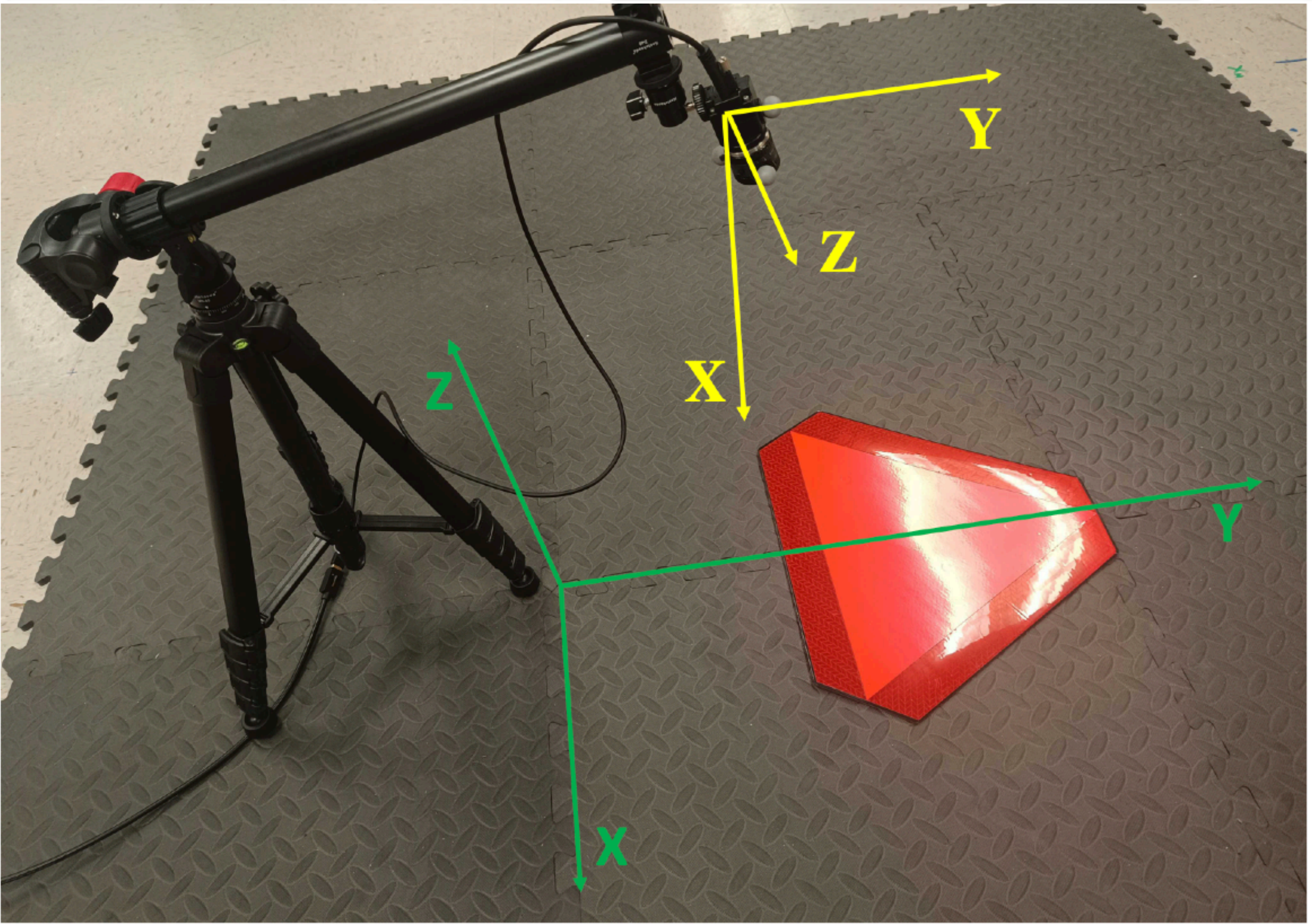
$$I_o(\xi) = GGM_o(\xi)$$

Ground Truth states (Vicon Cameras)

SilkyevCam Event Based Camera







Y
Z
X
Z
Y
X

Assured NN-based Perception

Theorem (Informal Version)

For any 2D object that can be formed as unions and intersection of polytopes, then the Geometry-based Generative Model (GGM) Neural Network is equivalent to the Pin-hole camera model, i.e.,

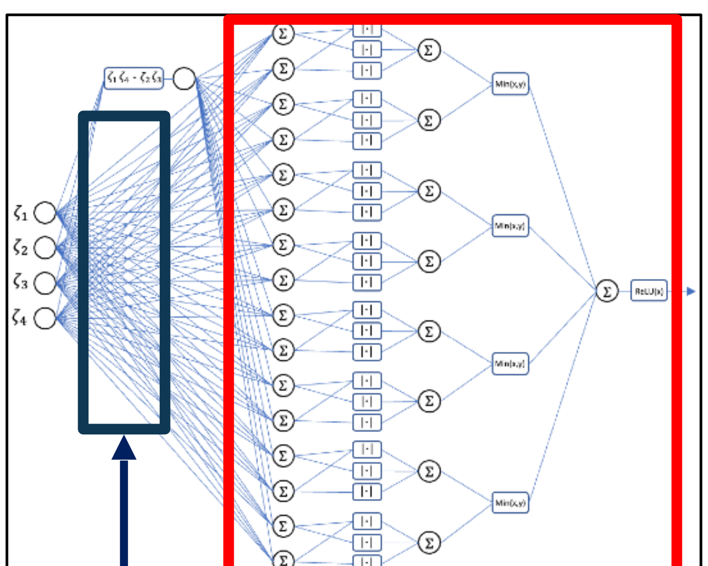
$$I_o(\xi) = GGM_o(\xi)$$

Can we design certified "object detectors"?

Can we design certified "state estimators"?

Geometry-based Generative Model

Position, angles ξ



Image

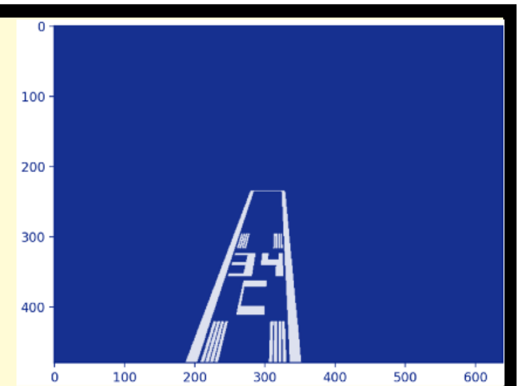
Physical Parameters (Trainable)

Geometrical Parameters (Non-Trainable)

Assured NN-based Perception

Case 1: Ideal Image

$$I(\xi) = GGM_o(\xi)$$

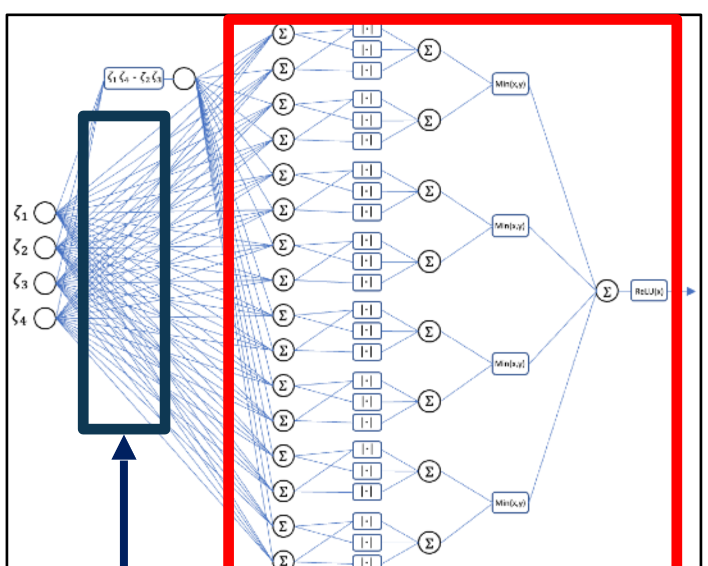


Can we design certified "object detectors"?

Can we design certified "state estimators"?

Geometry-based Generative Model

Position, angles ξ



Image

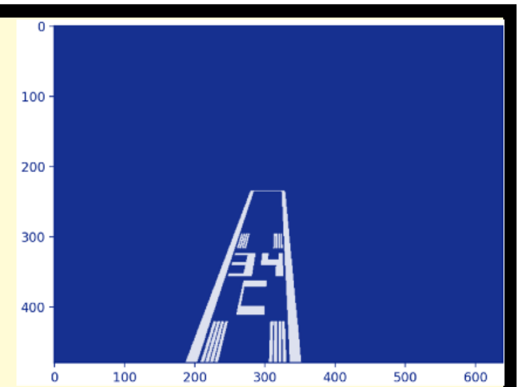
Physical Parameters (Trainable)

Geometrical Parameters (Non-Trainable)

Assured NN-based Perception

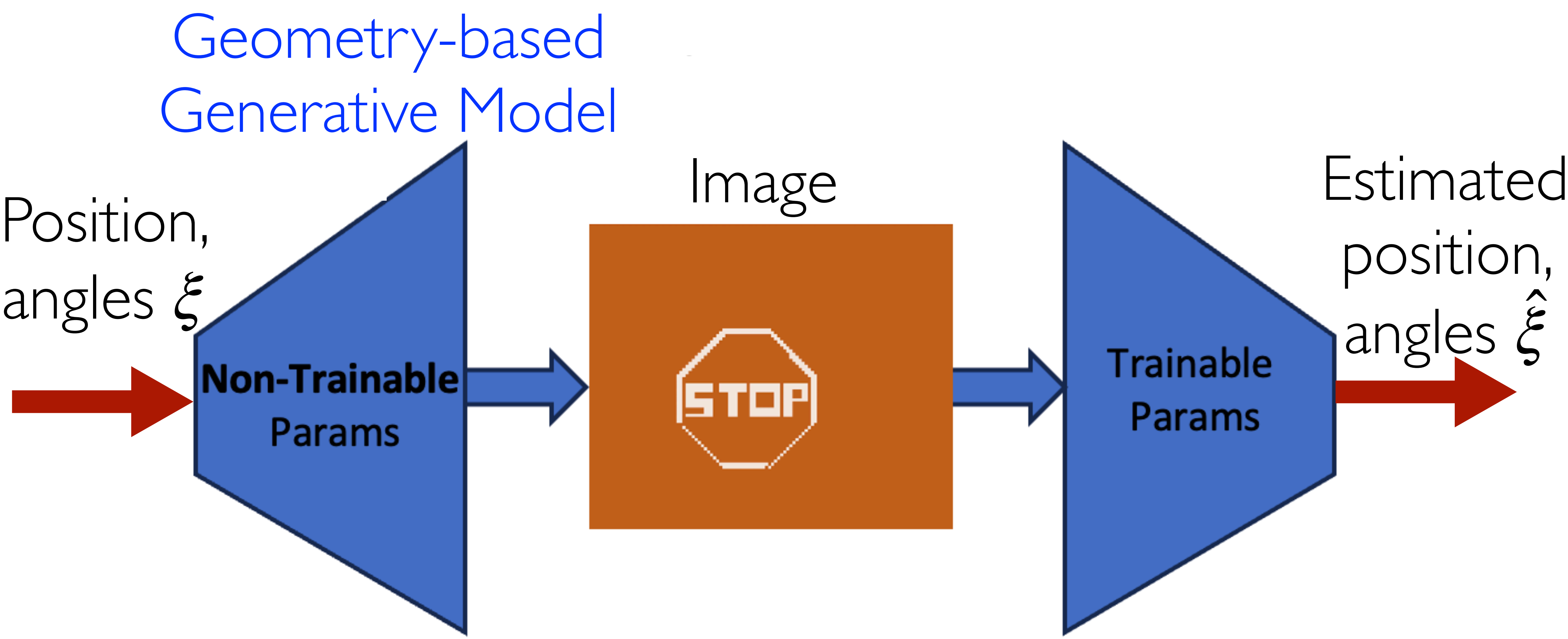
Case 1: Ideal Image

$$I(\xi) = GGM_o(\xi)$$



Can we design certified "object detectors"?

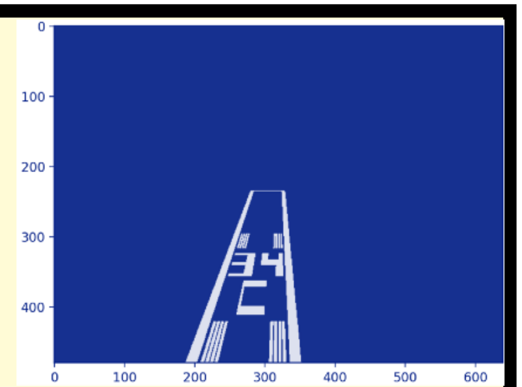
Can we design certified "state estimators"?



Assured NN-based Perception

Case 1: Ideal Image

$$I(\xi) = GGM_o(\xi)$$

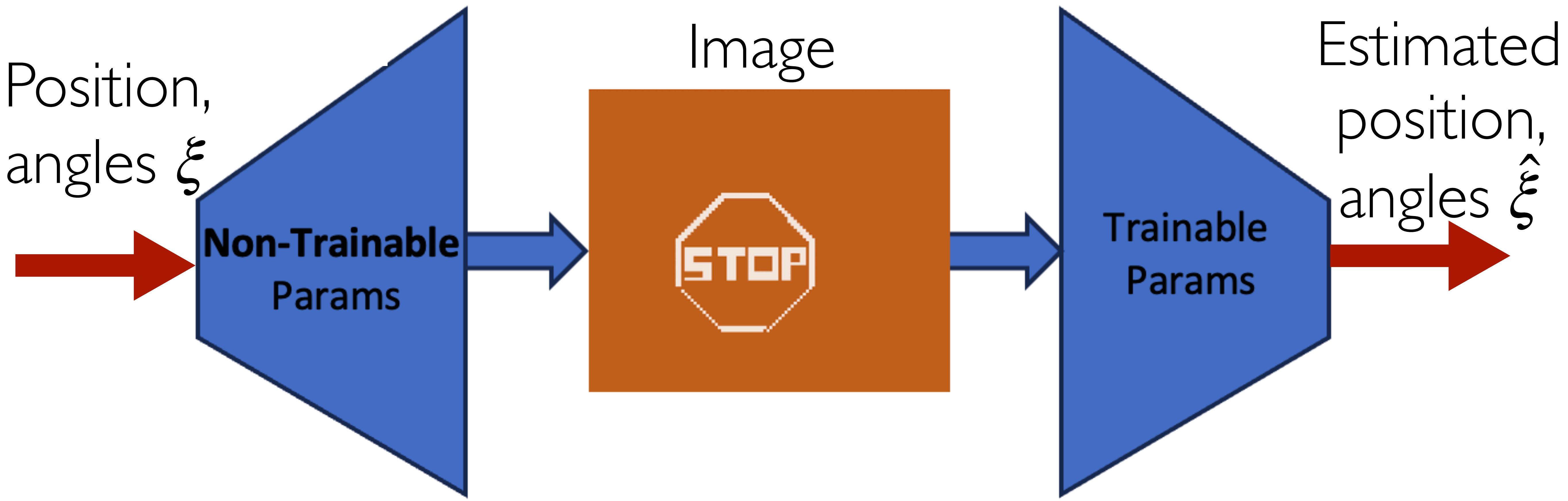


$$\|\hat{\xi} - \xi\| \leq \eta L_{NN} + e_{max}$$

Can we design certified "object detectors"?

Can we design certified "state estimators"?

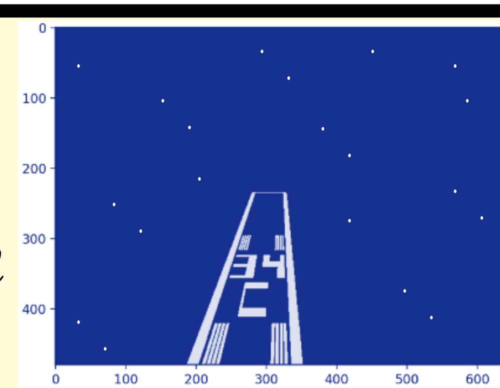
Geometry-based Generative Model



Assured NN-based Perception

Case 2: Limited Noise

$$I(\xi) = GGM_o(\xi) + I_n$$



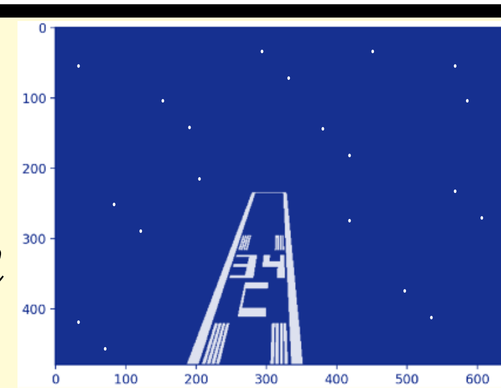
Can we design certified "object detectors"?

Can we design certified "state estimators"?

Assured NN-based Perception

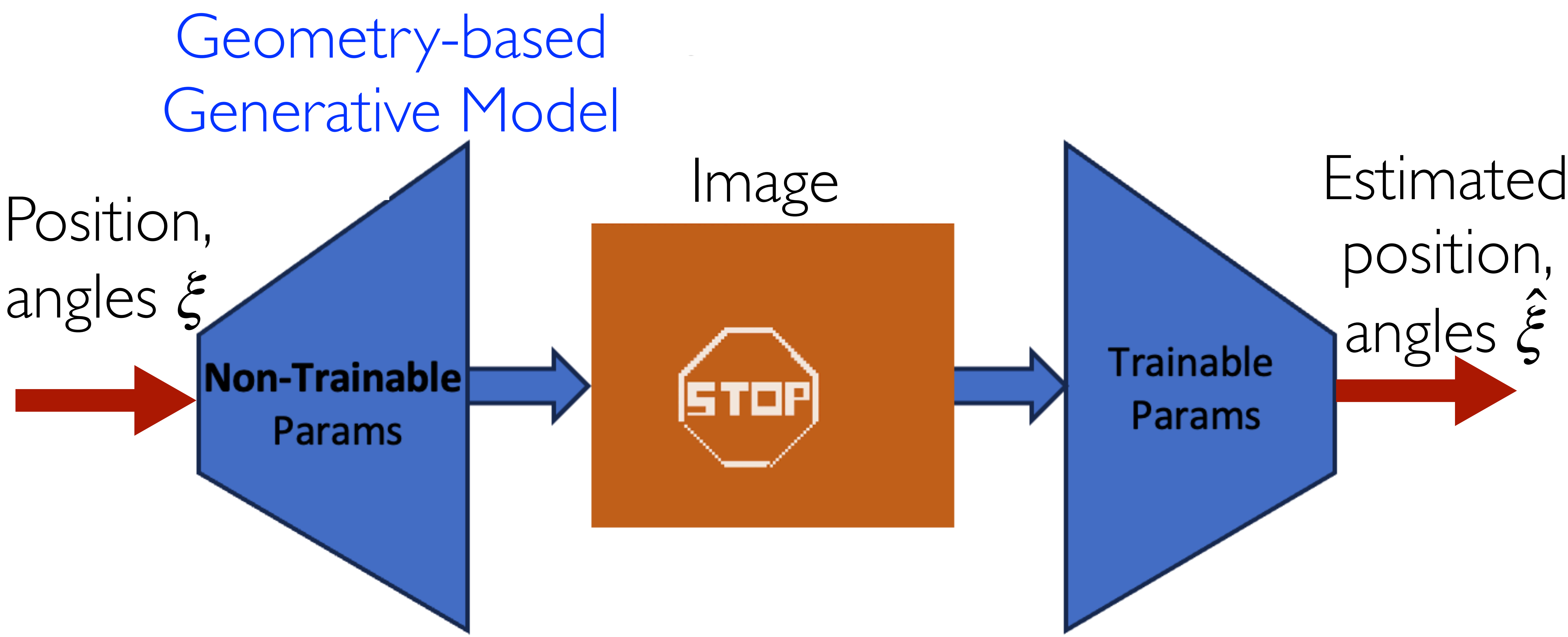
Case 2: Limited Noise

$$I(\xi) = GGM_o(\xi) + I_n$$



Can we design certified "object detectors"?

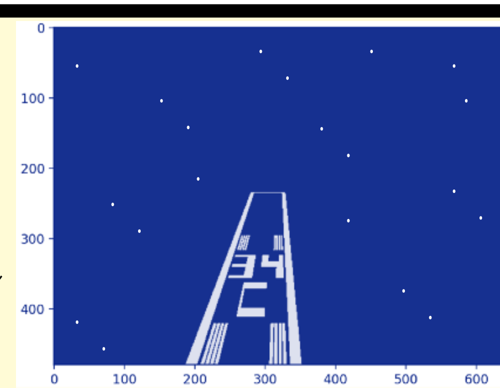
Can we design certified "state estimators"?



Assured NN-based Perception

Case 2: Limited Noise

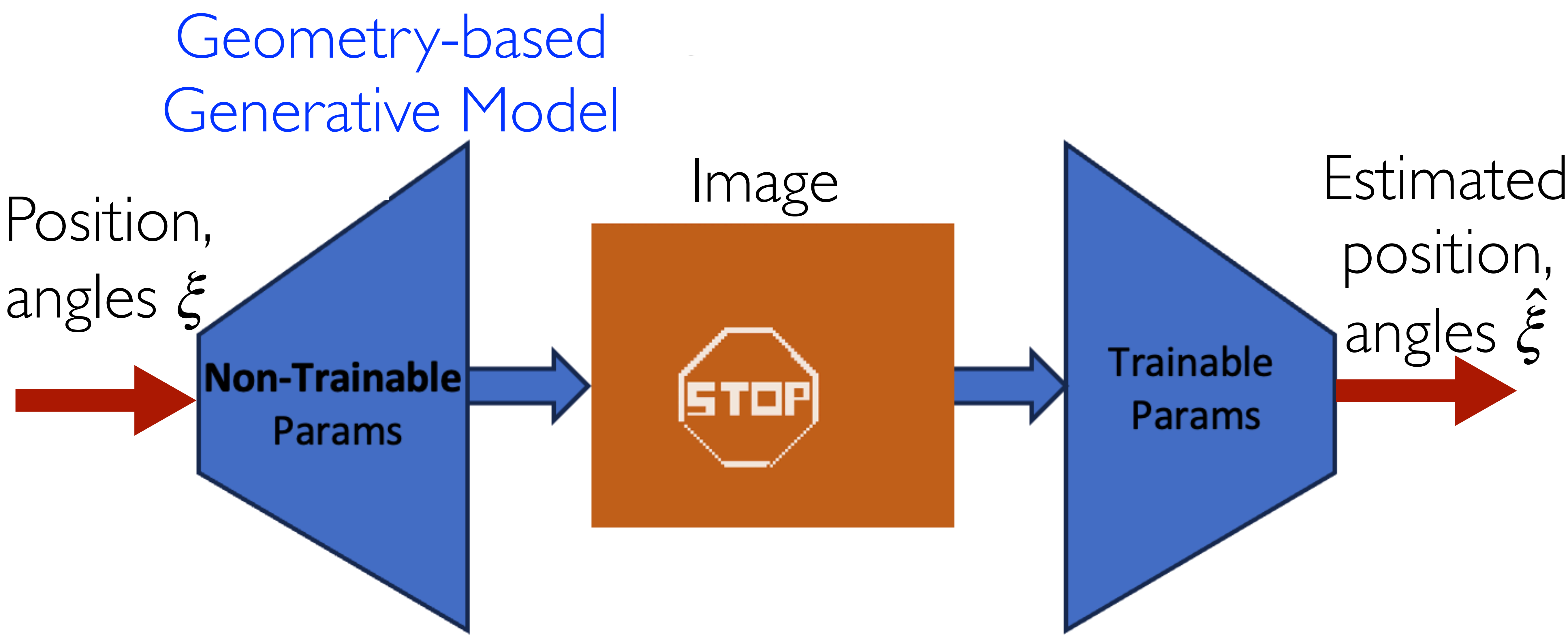
$$I(\xi) = GGM_o(\xi) + I_n$$



$$\|\hat{\xi} - \xi\| \leq \eta L_{NN} + e_{max} + L_{NN}\|I_n\|$$

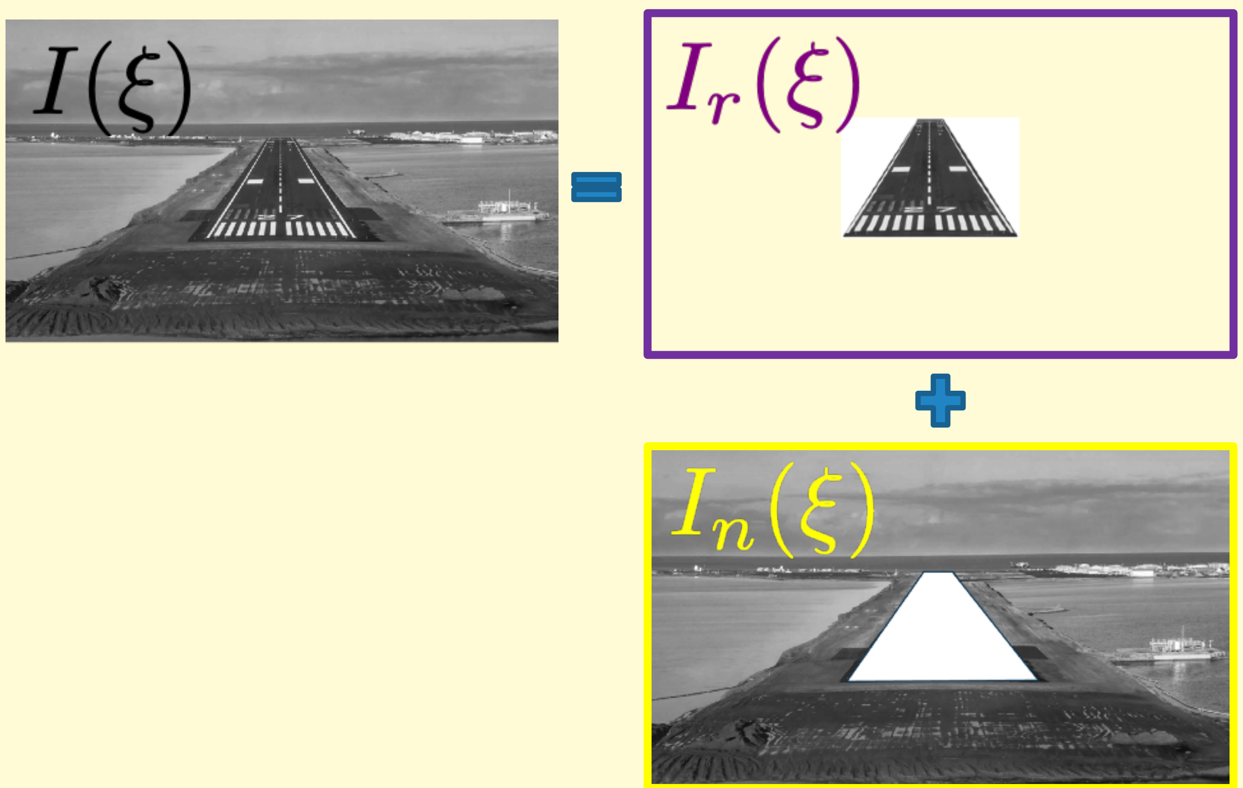
Can we design certified "object detectors"?

Can we design certified "state estimators"?



Assured NN-based Perception

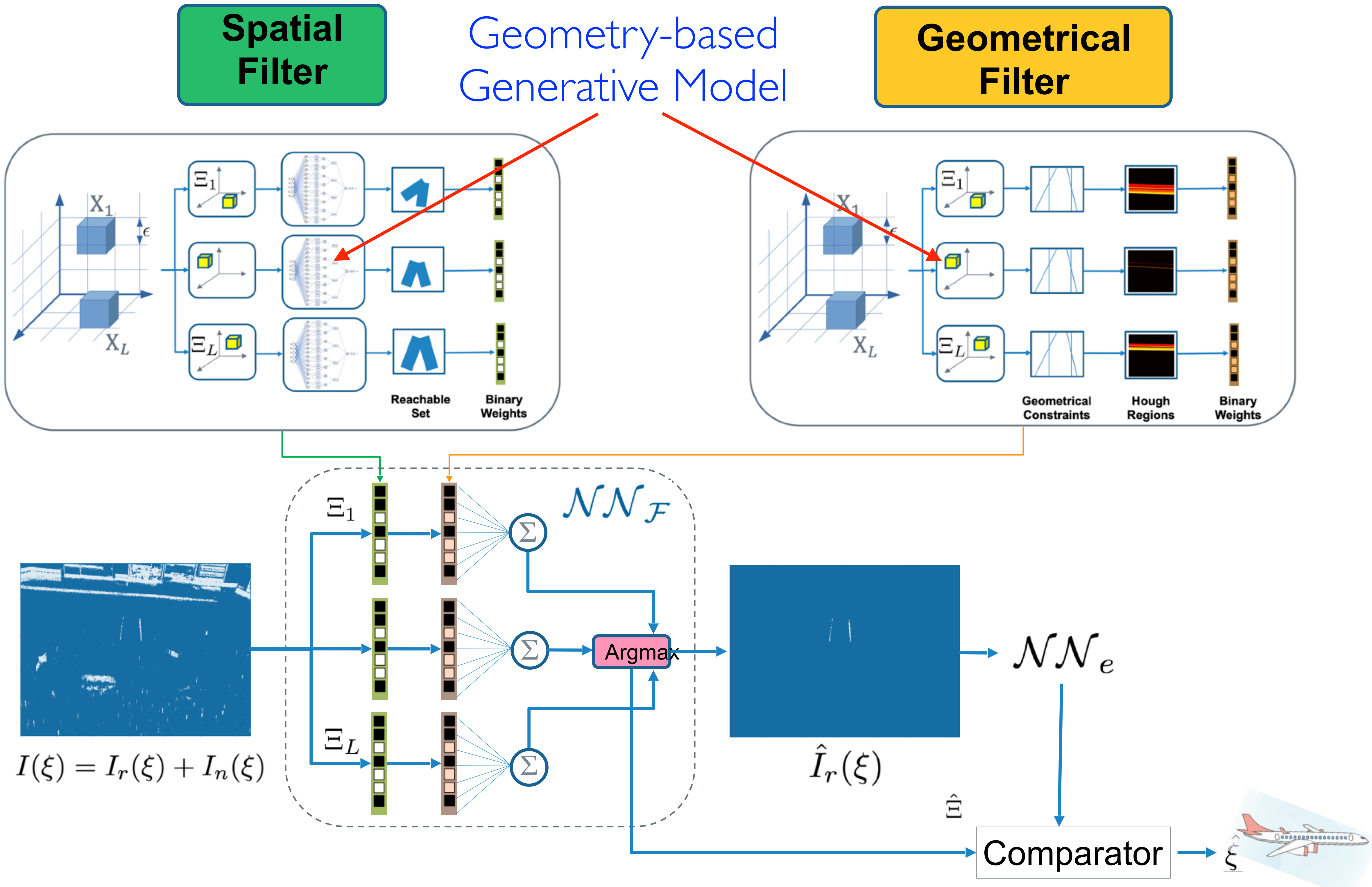
Case 3: Cluttered Noise Image



Can we design certified "object detectors"?

Can we design certified "state estimators"?

Assured NN-based Perception



CPS Lab Resilient

Assured NN-based Perception

Theorem (Informal Version)

Given:

- A camera image: $I(\xi) = I_r(\xi) + I_n(\xi)$
- Partitioning of the state space: $\Xi_1, \ldots, \Xi_l$

Under the following assumptions:

(i) $I_n(\xi) \notin \{\mathcal{NN}_r(\xi) | \xi \in \Xi\}$

(ii) $\forall \xi \in \Xi^*. [I_n(\xi) \otimes \mathcal{NN}_r(\xi) = \mathbf{0}_{a,b}]$

The following holds:

$$\hat{\Xi} = \Xi^*$$

$$\hat{I}_r = I_r(\xi)$$

$$||\xi - \hat{\xi}|| \leq 4 L_h \delta$$

Where:

$$(\hat{\Xi}, \hat{I}_r) = \mathcal{NN}_F(I(\xi))$$

Other objects can not be generated by the same geometric generative model of the runway, **i.e., other objects not look like a target runway.**

Other objects does not appear in the neighborhood of the runway

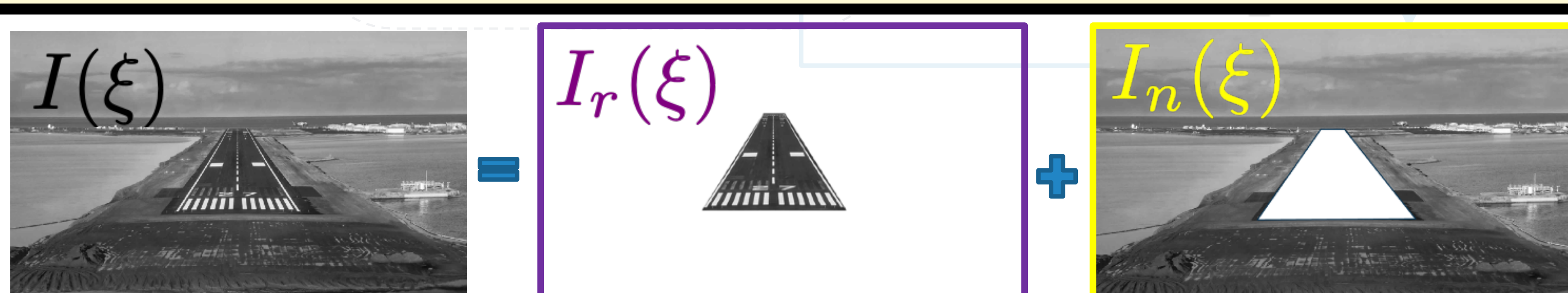
NN output:

- The partition where the state belongs
- Filtered image estimate.

Bound:

L_h Lipschitz constant of Generative Model

δ Radius of the infinity ball used to partition the state space



CPS Lab Resilient

Assured NN-based Perception

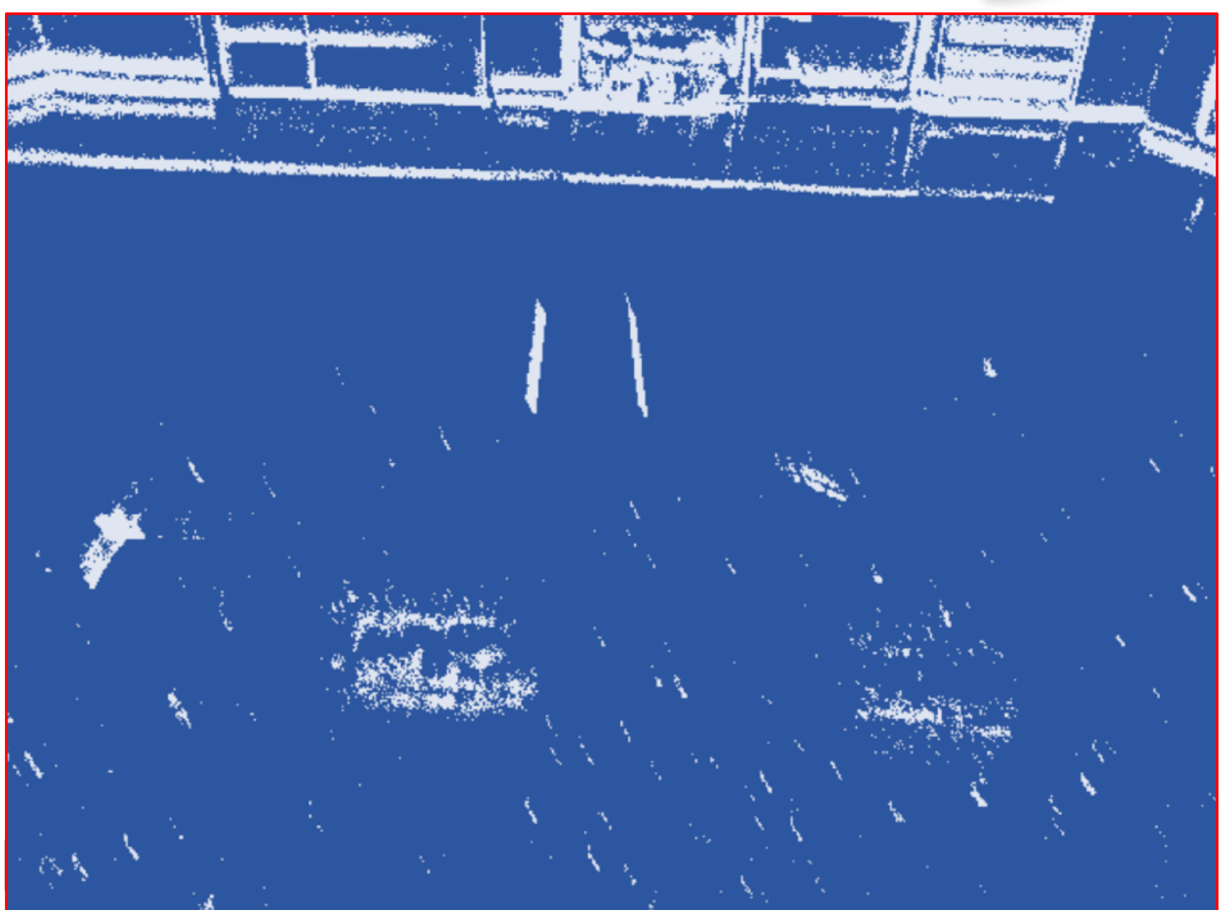
SilkyevCam Event Based Camera

Model	PPS3MVCD (PROPHESEE)
Image size	Type 3/4 " (Diagonal 12mm)
Module effective pixels	VGA (640 (H) x 480 (V))
Pixel size	15um x 15um
Typical Latency	200us

Ground Truth states (Vicon Cameras)

$$\|\xi - \hat{\xi}\| \leq 0.5$$

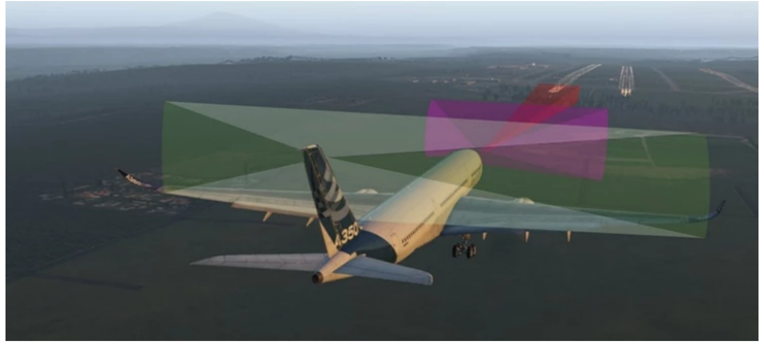
Original Video



Filtered Video

Formal Verification Tools for NN Analysis

Assured NN-based Perception



Assured NN-based Control

Xiaowu Sun

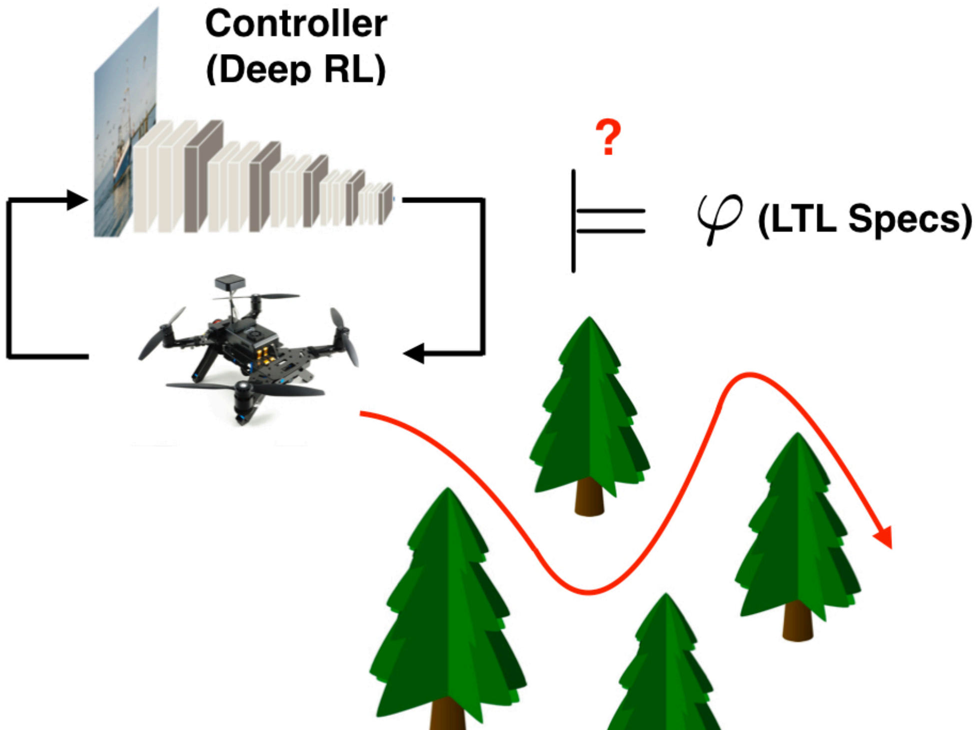
X. Sun and Y. Shoukry, "Neurosymbolic Motion and Task Planning for Linear Temporal Logic Tasks," T-RO, submitted, arXiv 2022.

X. Sun, W. Fatnassi, U. Santa Cruz, and Y. Shoukry, "Provably Safe Model-Based Meta Reinforcement Learning: An Abstraction-Based Approach," CDC 2021.

X. Sun and Y. Shoukry, "NNSynth: Neural Network Guided Abstraction-Based Controller Synthesis for Stochastic Systems," CDC 2022.

Assured NN-based Perception

Assured NN-based Control



Assured Meta Learning for LTL Tasks

Given: a nonlinear dynamical system $x^{(k+1)} = f(x^{(k)}, u^{(k)}) + g(x^{(k)}, u^{(k)})$

The nominal model f is assumed to be black-box model, e.g., a simulator or a neural network.

The unknown model-error g is assumed to be bounded and can be learned by Gaussian Process regression.

Assured Meta Learning for LTL Tasks

Given: a nonlinear dynamical system $x^{(k+1)} = f(x^{(k)}, u^{(k)}) + g(x^{(k)}, u^{(k)})$

Objective: train a neural network-based controller $u^{(k)} = \mathcal{NN}(x^{(k)})$

such that the closed-loop system satisfies safety and liveness specifications:

$$\mathcal{NN}, \mathcal{X}_{\text{init}}^{\mathcal{W}} \models \phi_{\text{safety}}^{\mathcal{W}} \wedge \phi_{\text{liveness}}^{\mathcal{W}},$$

The nominal model f is assumed to be black-box model, e.g., a simulator or a neural network.

The unknown model-error g is assumed to be bounded and can be learned by Gaussian Process regression.

Assured Meta Learning for LTL Tasks

Given: a nonlinear dynamical system $x^{(k+1)} = f(x^{(k)}, u^{(k)}) + g(x^{(k)}, u^{(k)})$

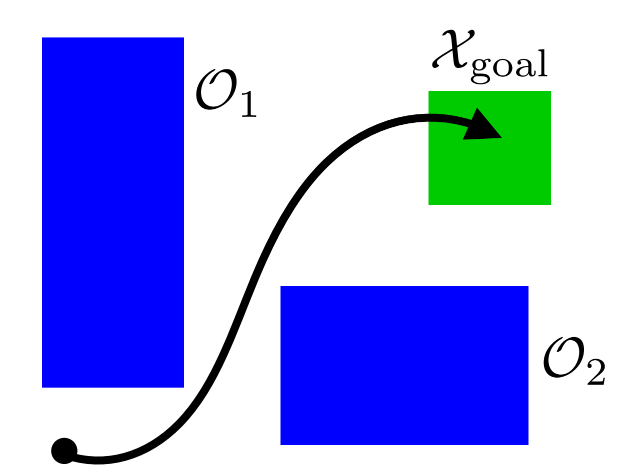
Objective: train a neural network-based controller $u^{(k)} = \mathcal{NN}(x^{(k)})$

such that the closed-loop system satisfies safety and liveness specifications:

$$\mathcal{NN}, \mathcal{X}_{\text{init}}^{\mathcal{W}} \models \phi_{\text{safety}}^{\mathcal{W}} \wedge \phi_{\text{liveness}}^{\mathcal{W}},$$

The nominal model f is assumed to be black-box model, e.g., a simulator or a neural network.

The unknown model-error g is assumed to be bounded and can be learned by Gaussian Process regression.



$$\xi_{x_0,\mathcal{NN}} \models \phi_{\text{safety}}^{\mathcal{W}} \iff \forall k \in \mathbb{N},\ \xi_{x_0,\mathcal{NN}}(k) \notin \mathcal{O}_1 \cup \mathcal{O}_2$$

$$\xi_{x_0,\mathcal{NN}} \models \phi_{\text{liveness}}^{\mathcal{W}} \iff \exists k \in \{1, \ldots, H\},\ \xi_{x_0,\mathcal{NN}}(k) \in \mathcal{X}_{\text{goal}}$$

Assured Meta Learning for LTL Tasks

Given: a nonlinear dynamical system $x^{(k+1)} = f(x^{(k)}, u^{(k)}) + g(x^{(k)}, u^{(k)})$

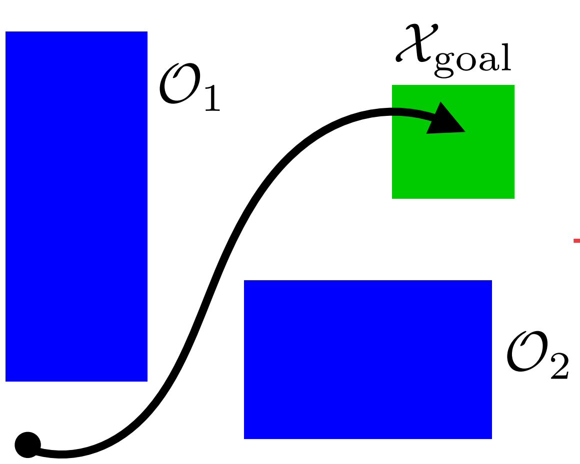
Objective: train a neural network-based controller $u^{(k)} = \mathcal{NN}(x^{(k)})$

such that the closed-loop system satisfies safety and liveness specifications:

$$\mathcal{NN}, \mathcal{X}_{\text{init}}^{\mathcal{W}} \models \phi_{\text{safety}}^{\mathcal{W}} \wedge \phi_{\text{liveness}}^{\mathcal{W}},$$

The nominal model f is assumed to be black-box model, e.g., a simulator or a neural network.

The unknown model-error g is assumed to be bounded and can be learned by Gaussian Process regression.



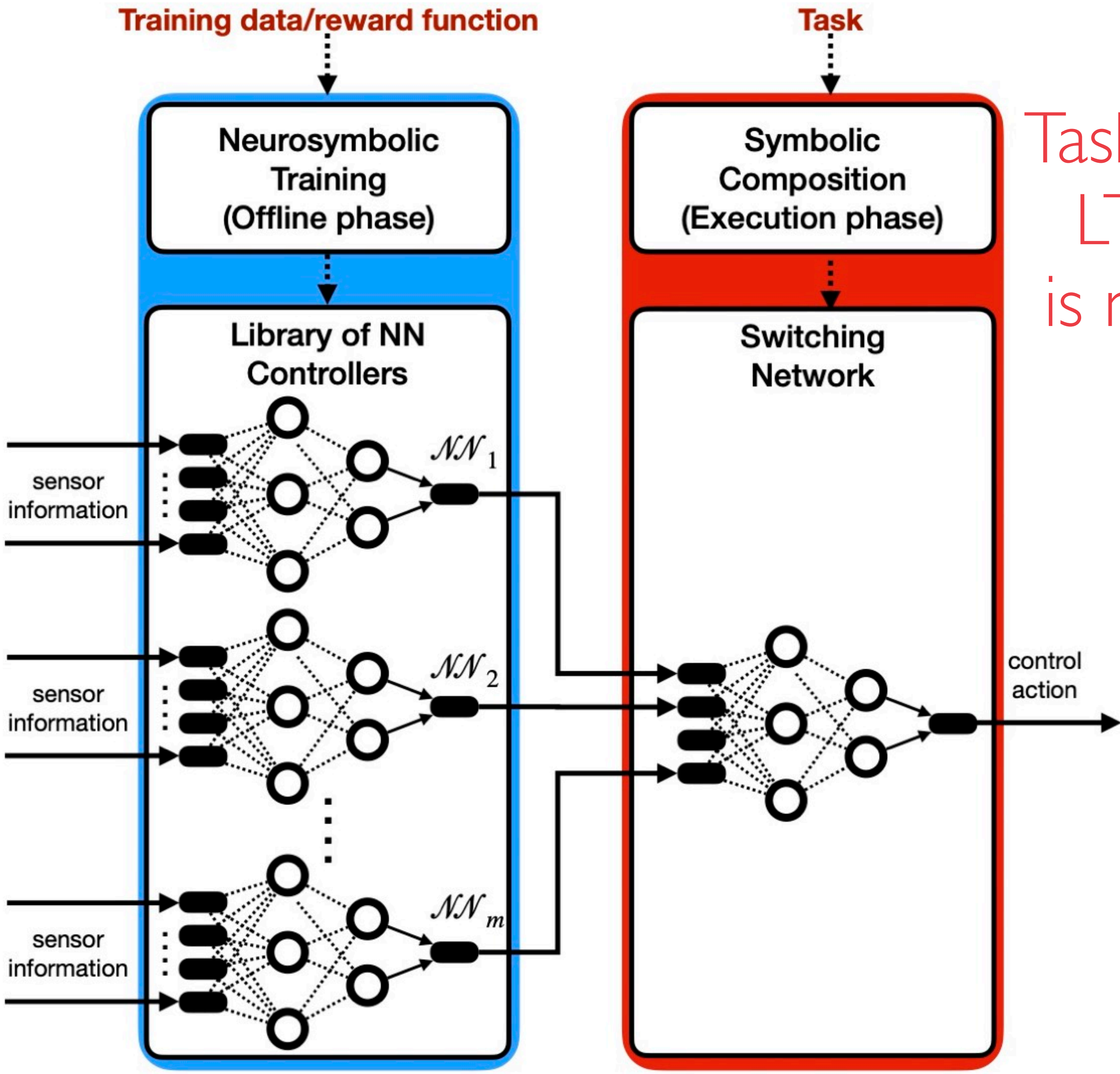
$$\xi_{x_0,\mathcal{NN}} \models \phi_{\text{safety}}^{\mathcal{W}} \iff \forall k \in \mathbb{N},\ \xi_{x_0,\mathcal{NN}}(k) \notin \mathcal{O}_1 \cup \mathcal{O}_2$$

$$\xi_{x_0,\mathcal{NN}} \models \phi_{\text{liveness}}^{\mathcal{W}} \iff \exists k \in \{1, \ldots, H\},\ \xi_{x_0,\mathcal{NN}}(k) \in \mathcal{X}_{\text{goal}}$$

Task = {workspace, obstacles, LTL mission, model error}
is not known during training

Assured Meta Learning for LTL Tasks

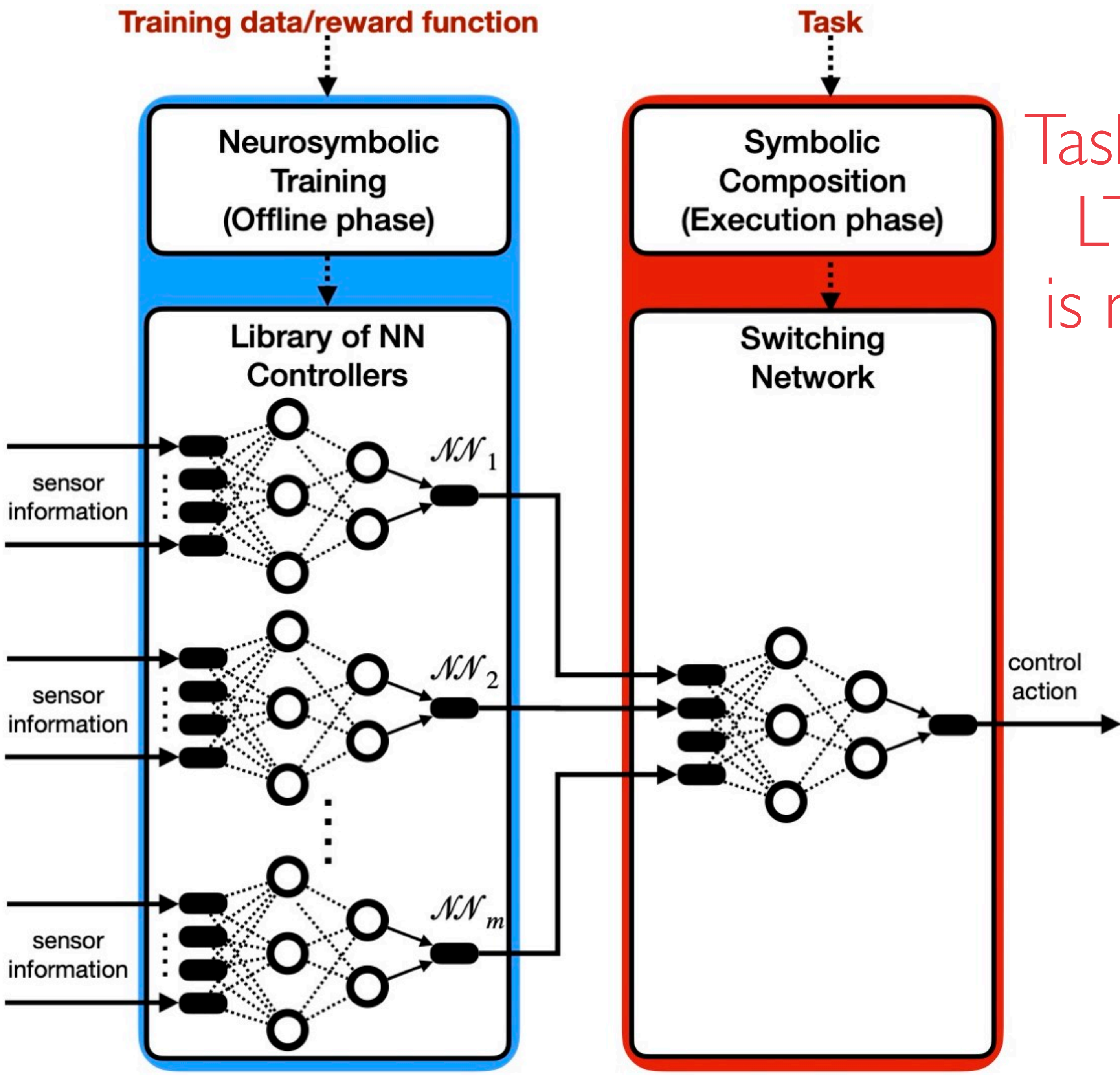
Train a *finite* library of NNs offline to satisfy *infinitely* many tasks at runtime



Task = {workspace, obstacles, LTL mission, model error} is not known during training

Assured Meta Learning for LTL Tasks

Train a *finite* library of NNs offline to satisfy *infinitely* many tasks at runtime

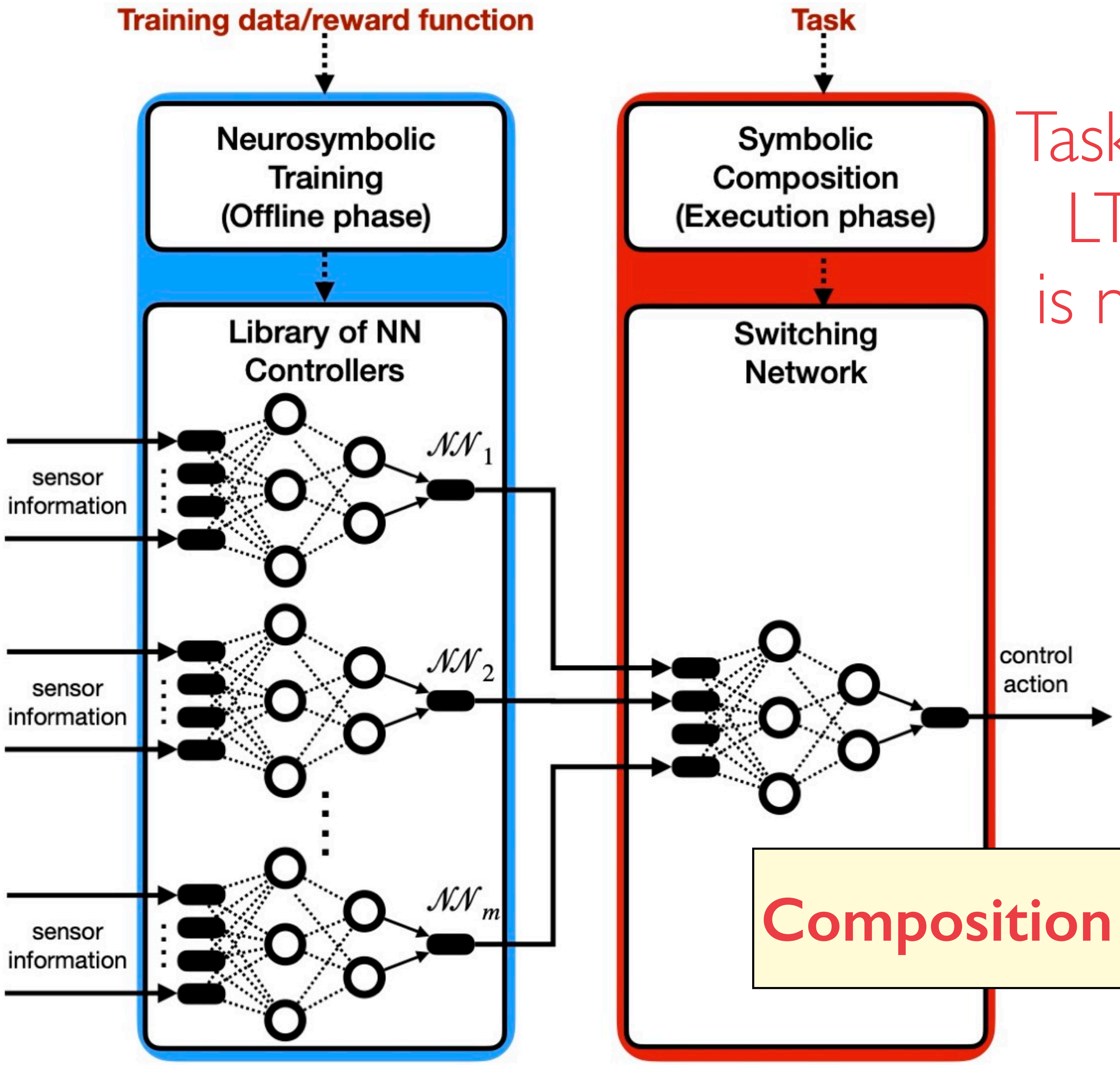


Task = {workspace, obstacles, LTL mission, model error} is not known during training

Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

Train a *finite* library of NNs offline to satisfy *infinitely* many tasks at runtime



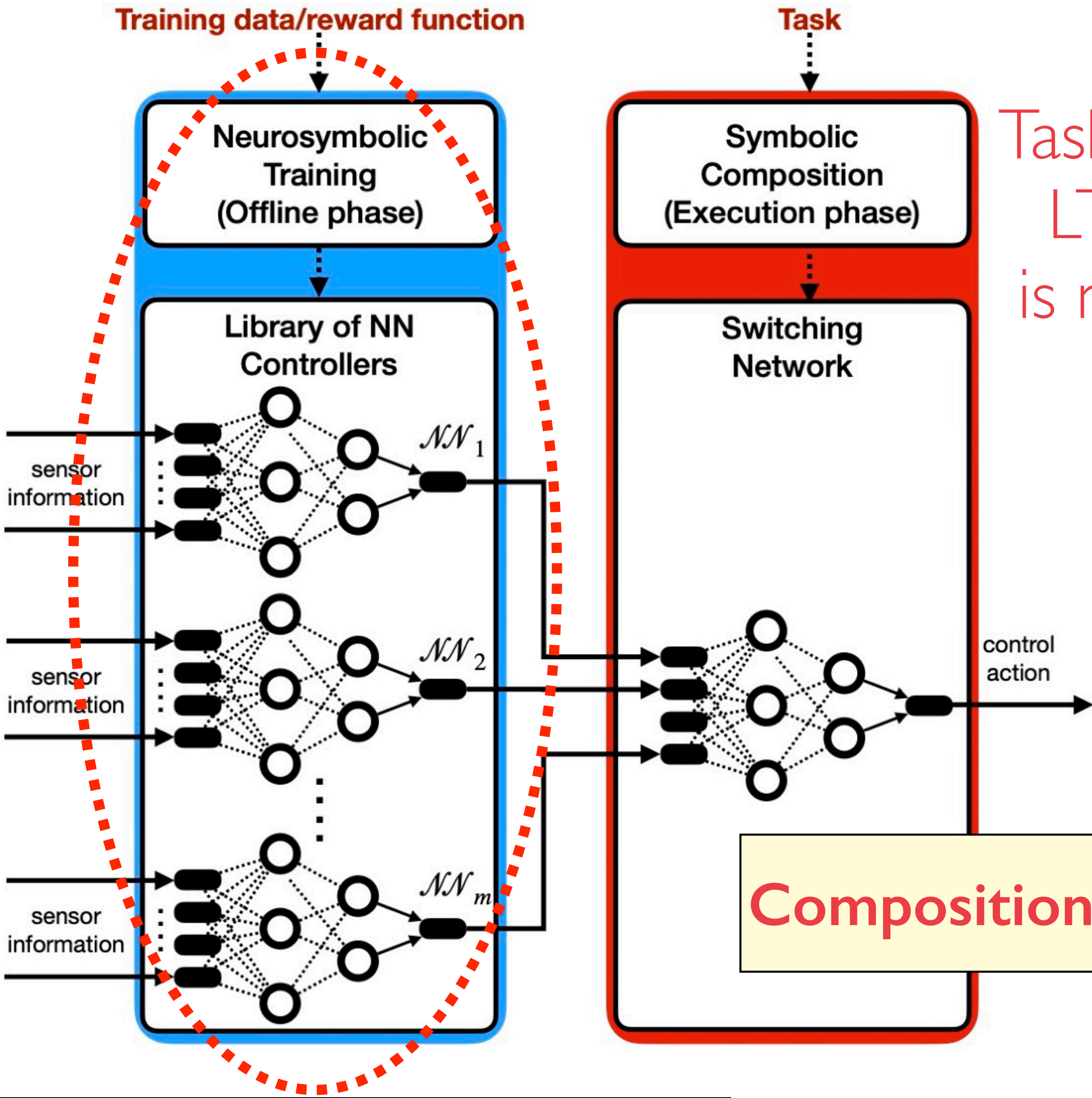
Task = {workspace, obstacles, LTL mission, model error} is not known during training

Composition of NNs is provably correct

Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

Train a *finite* library of NNs offline to satisfy *infinitely* many tasks at runtime



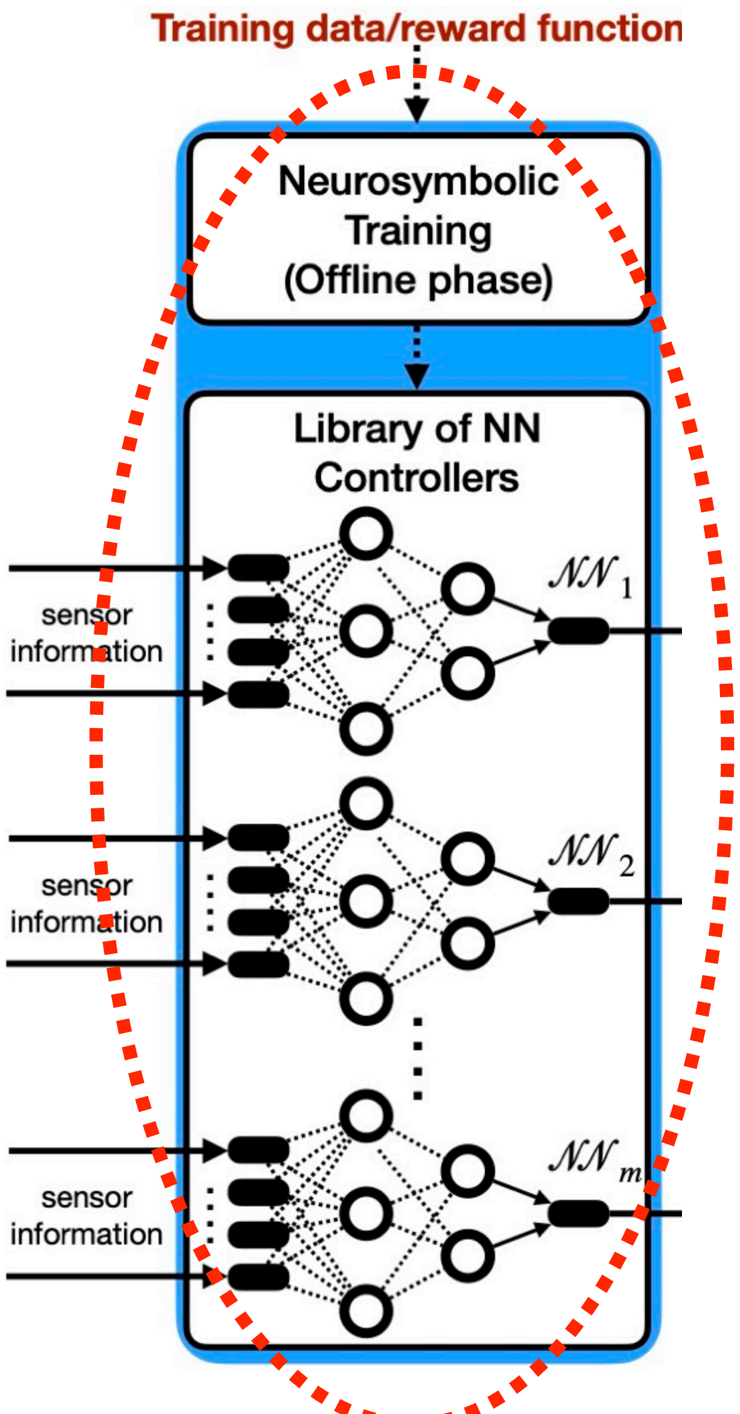
Task = {workspace, obstacles, LTL mission, model error} is not known during training

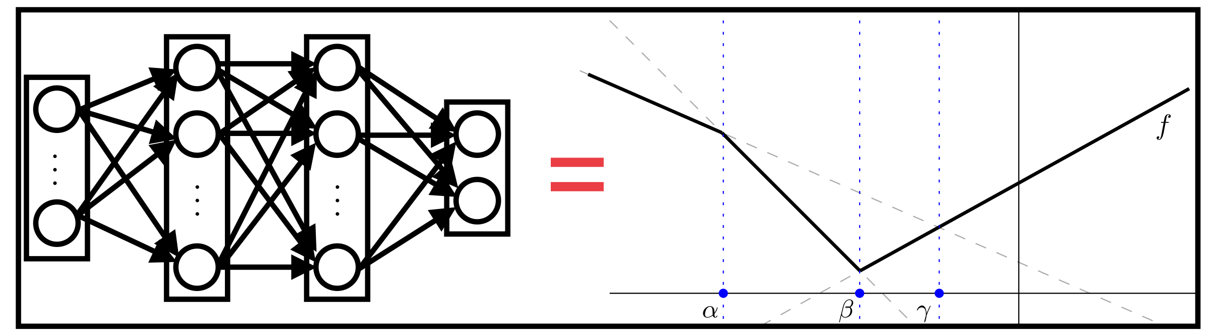
Composition of NNs is provably correct

Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

Train a ***finite*** library of NNs offline to satisfy ***infinitely*** many tasks at runtime



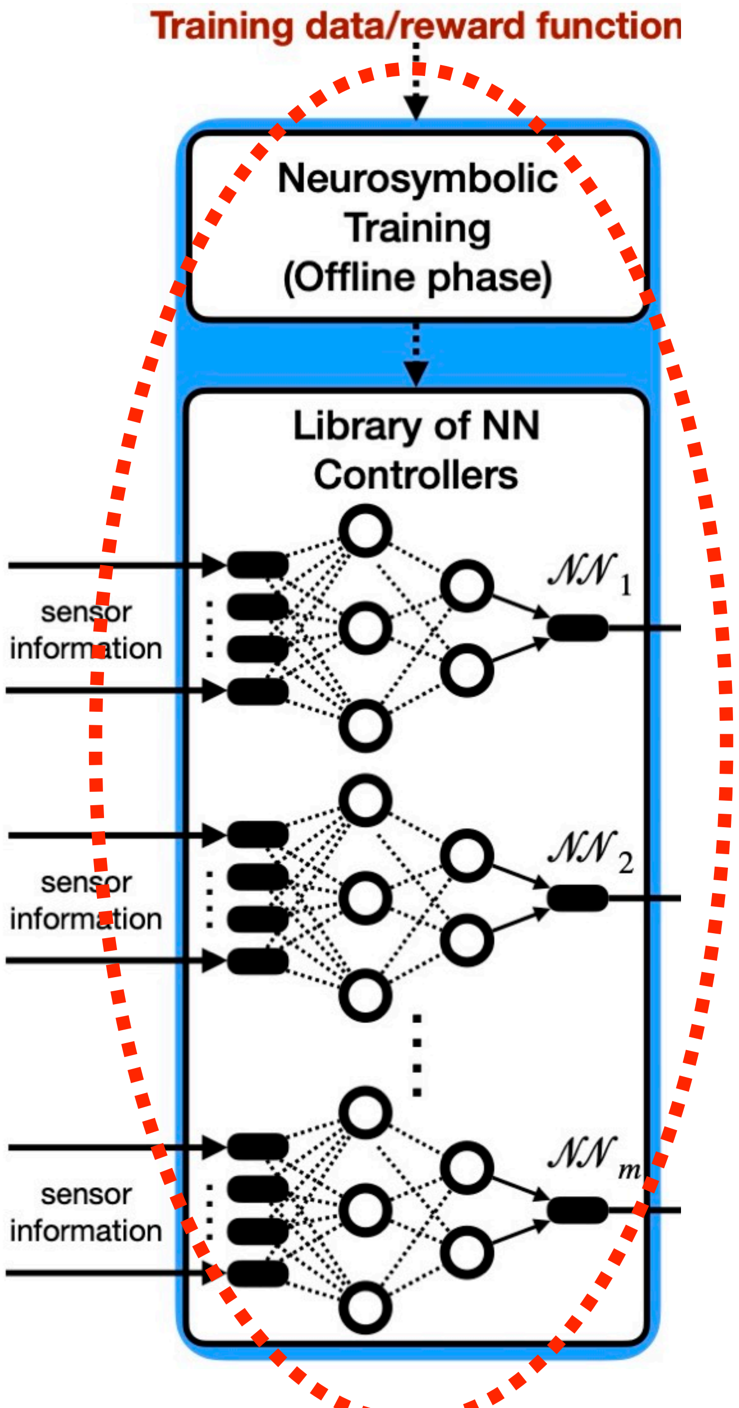


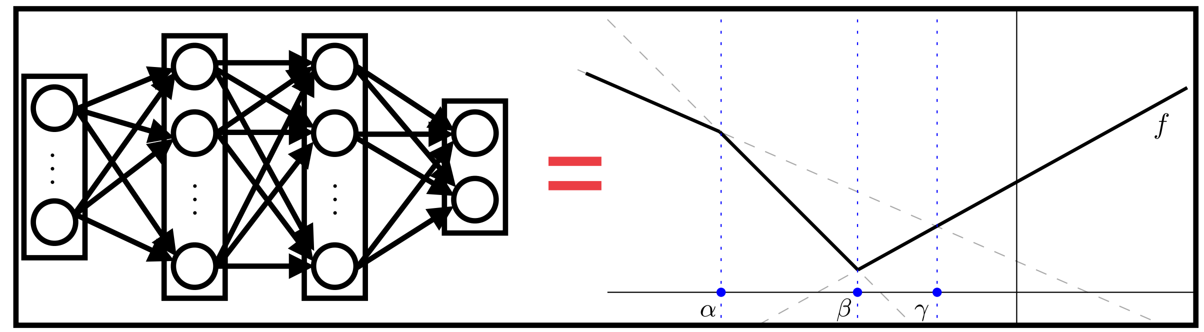
NN = Continuous Piece-Wise Affine (CPWA) functions

Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

Train a ***finite*** library of NNs offline to satisfy ***infinitely*** many tasks at runtime





NN = Continuous Piece-Wise Affine (CPWA) functions

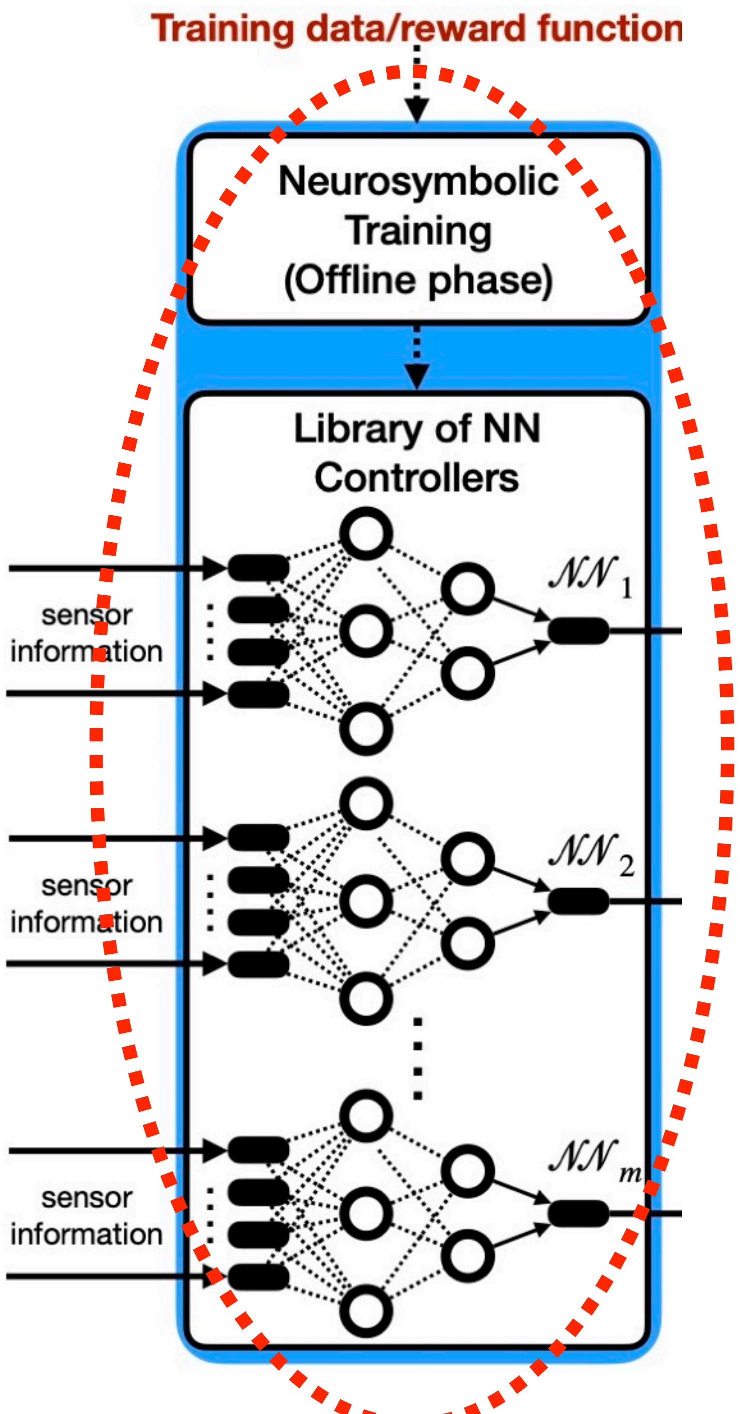
$$u^{(t)} = K_i x^{(t)} + b_i$$

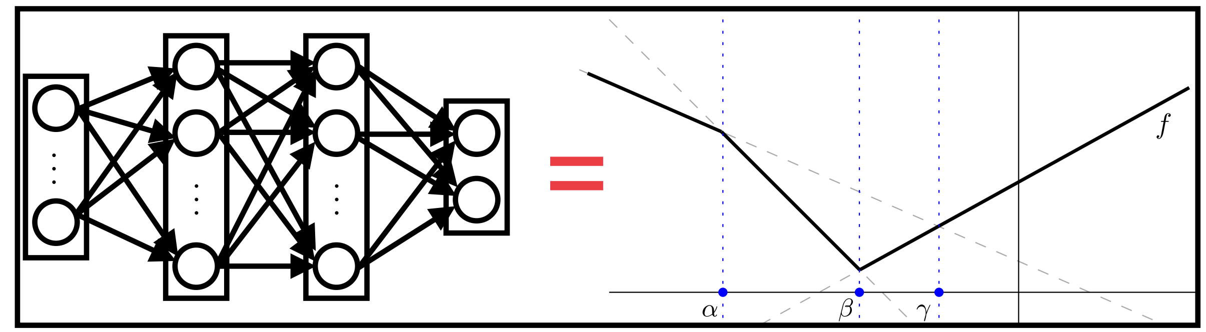
$$\mathcal{P} = \{(K, b) \mid K \in \underbrace{\mathcal{K}}_{\text{polytopic}}, b \in \underbrace{\mathcal{B}}_{\text{polytopic}}\}$$

Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

Train a ***finite*** library of NNs offline to satisfy ***infinitely*** many tasks at runtime





NN = Continuous Piece-Wise Affine (CPWA) functions

$$u^{(t)} = K_i x^{(t)} + b_i$$

$$\mathcal{P} = \{(K, b) \mid K \in \underbrace{\mathcal{K}}_{\text{polytopic}}, b \in \underbrace{\mathcal{B}}_{\text{polytopic}}\}$$

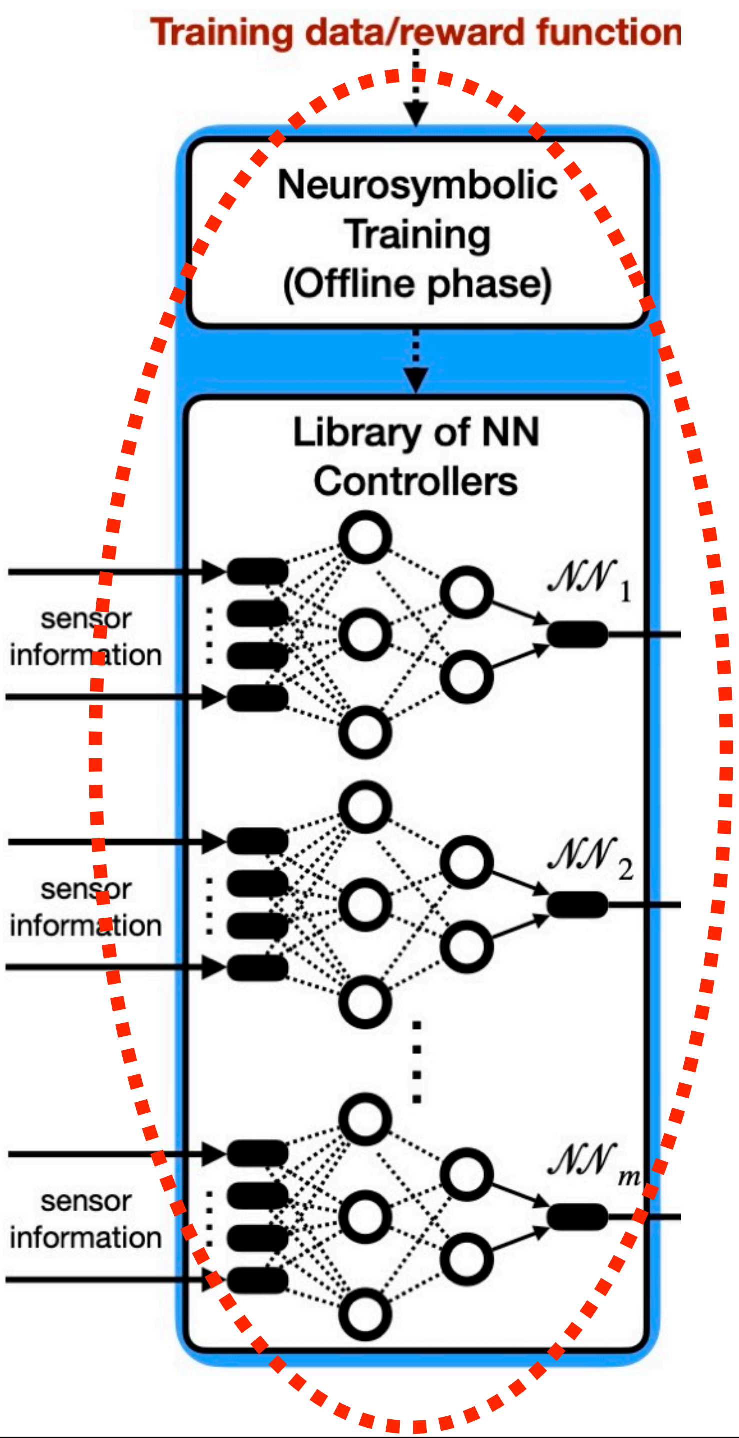
Controller Partitions:

$$\mathbb{P} = \{P_1, P_2, \ldots, P_m\}$$

Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

Train a *finite* library of NNs offline to satisfy *infinitely* many tasks at runtime



$$\mathcal{P} = \{(K, b) \mid K \in \underbrace{\mathcal{K}}_{\text{polytopic}}, b \in \underbrace{\mathcal{B}}_{\text{polytopic}}\}$$

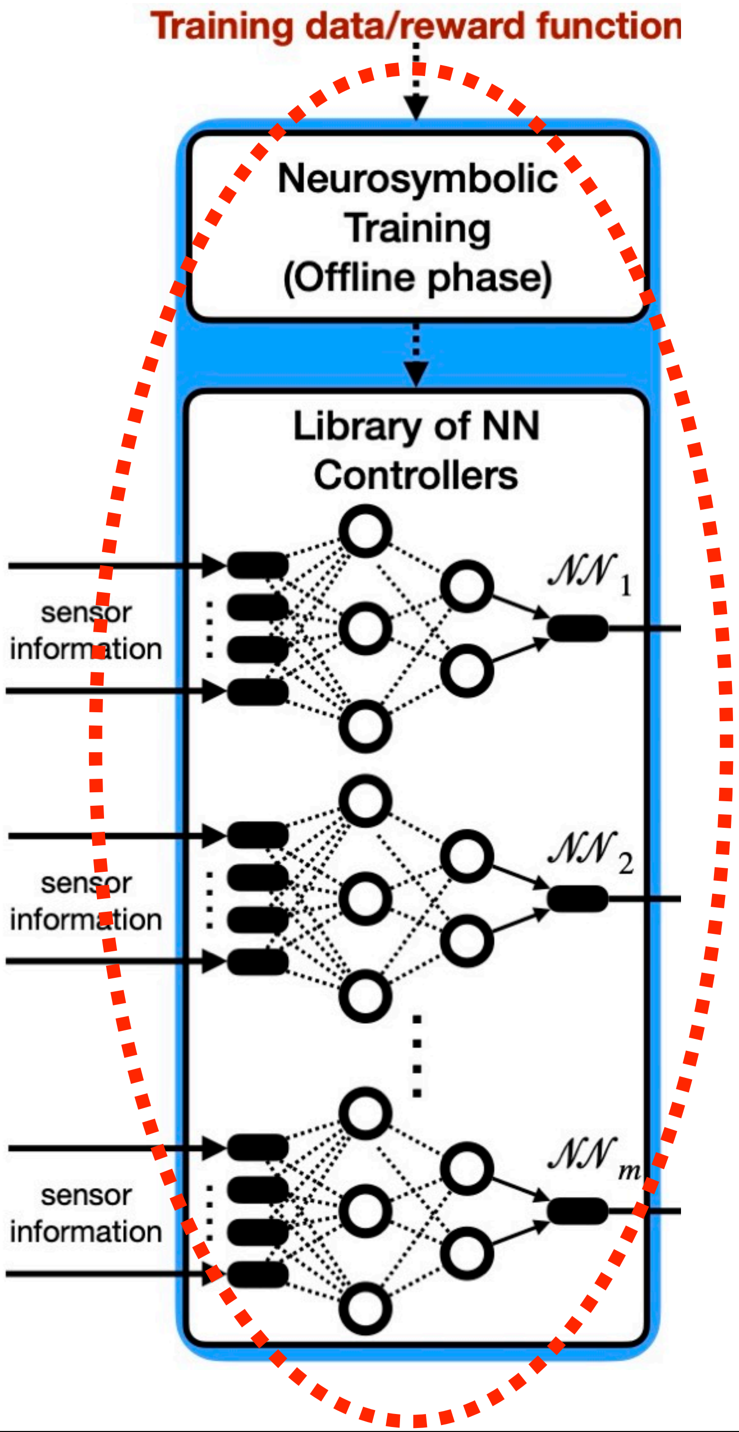
Controller Partitions:

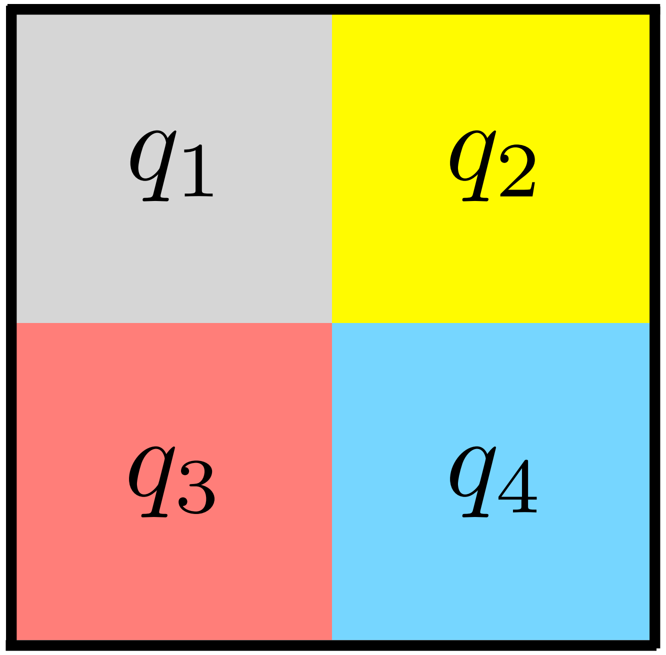
$$\mathbb{P} = \{P_1, P_2, \ldots, P_m\}$$

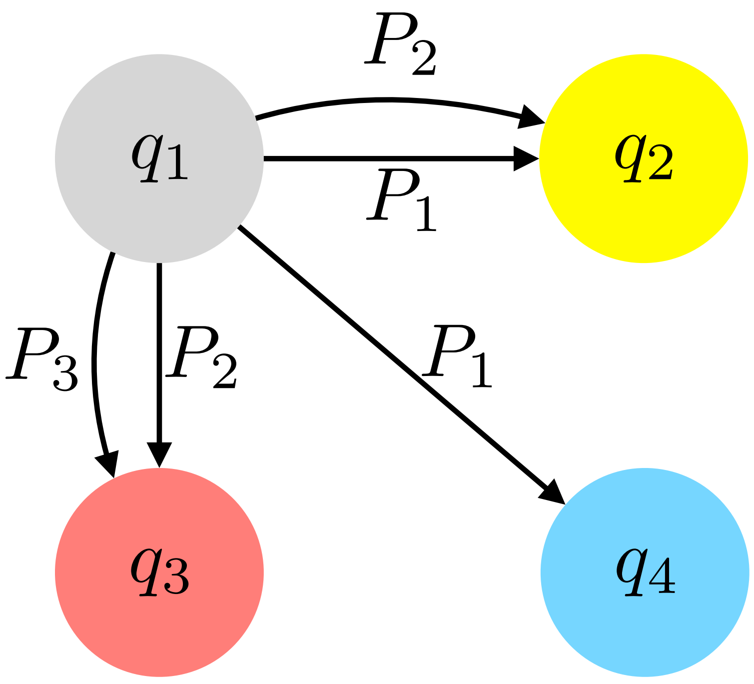
Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

Train a *finite* library of NNs offline to satisfy *infinitely* many tasks at runtime







$$\mathcal{P} = \{(K, b) \mid K \in \underbrace{\mathcal{K}}_{\text{polytopic}}, b \in \underbrace{\mathcal{B}}_{\text{polytopic}}\}$$

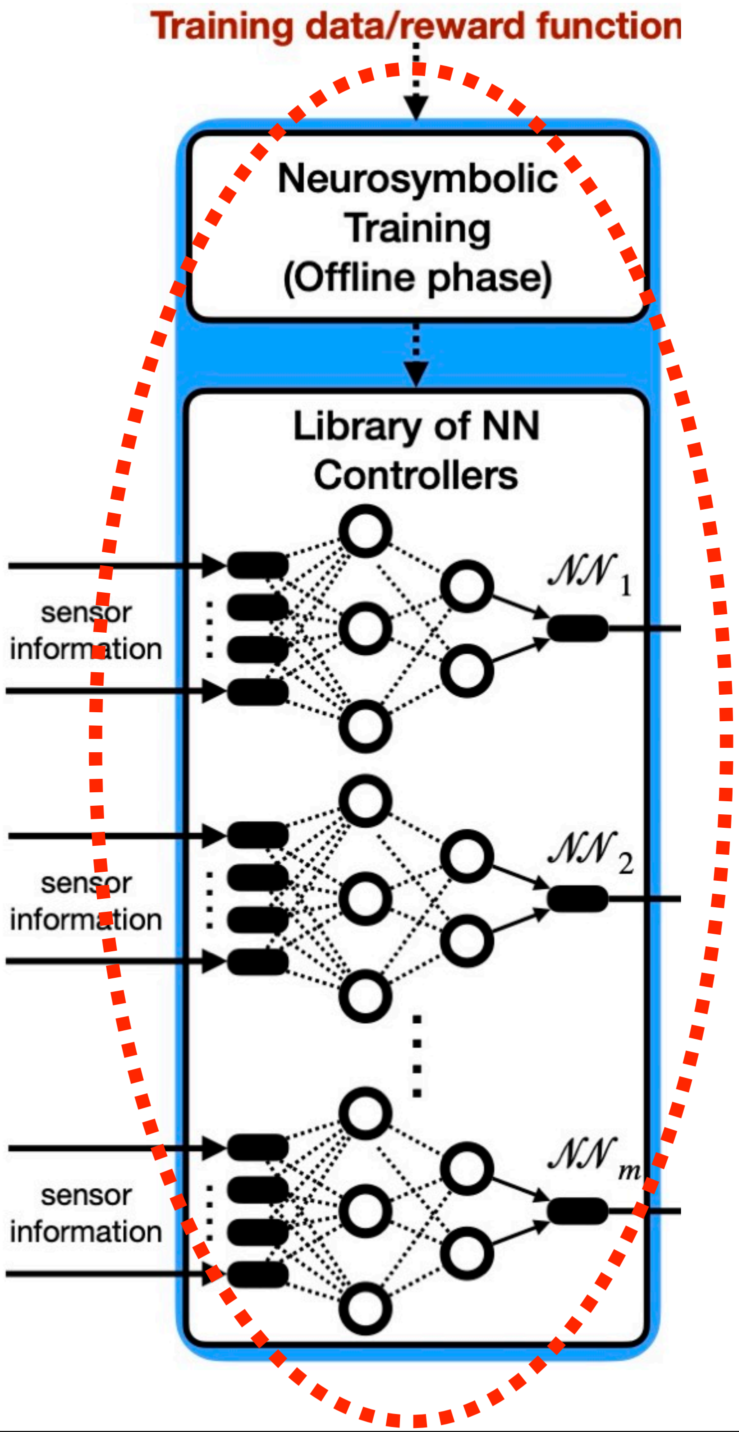
Controller Partitions:

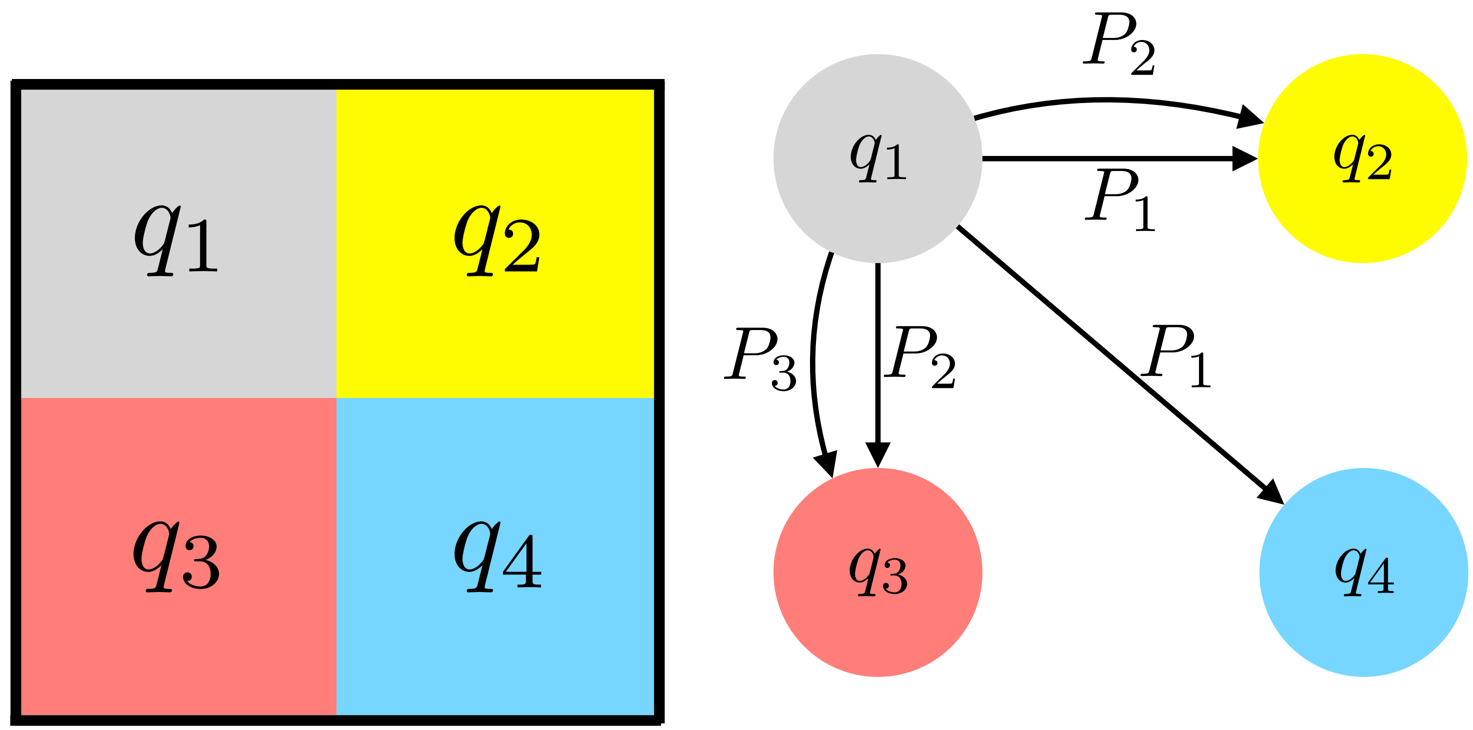
$$\mathbb{P} = \{P_1, P_2, \ldots, P_m\}$$

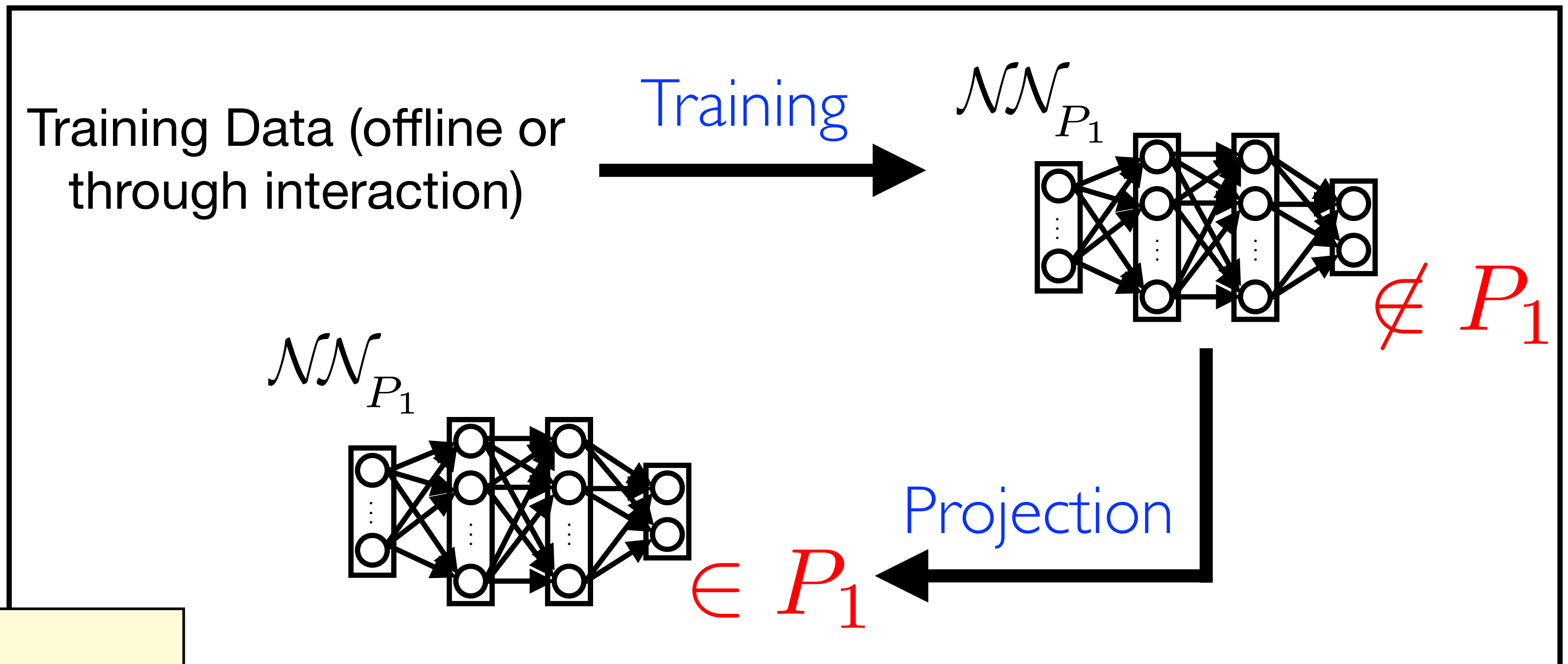
Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

Train a ***finite*** library of NNs offline to satisfy ***infinitely*** many tasks at runtime



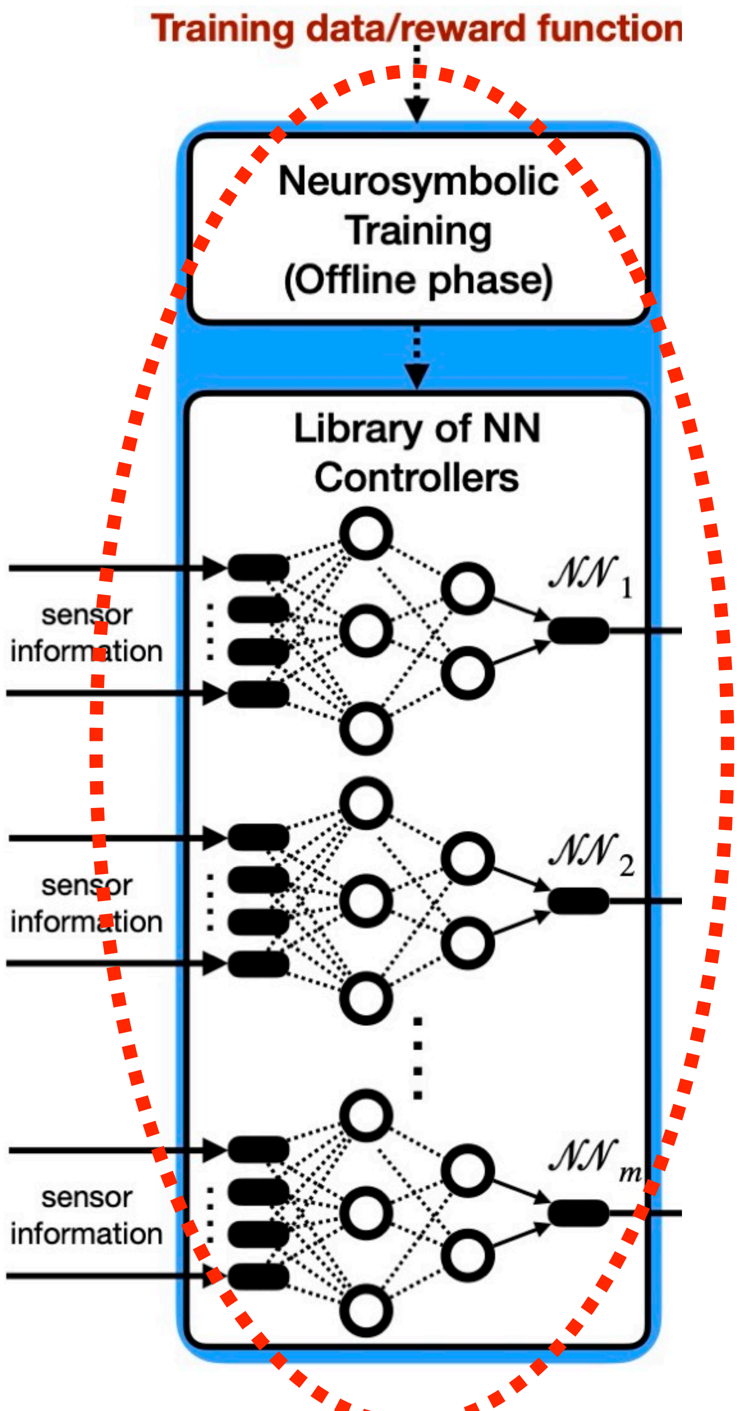




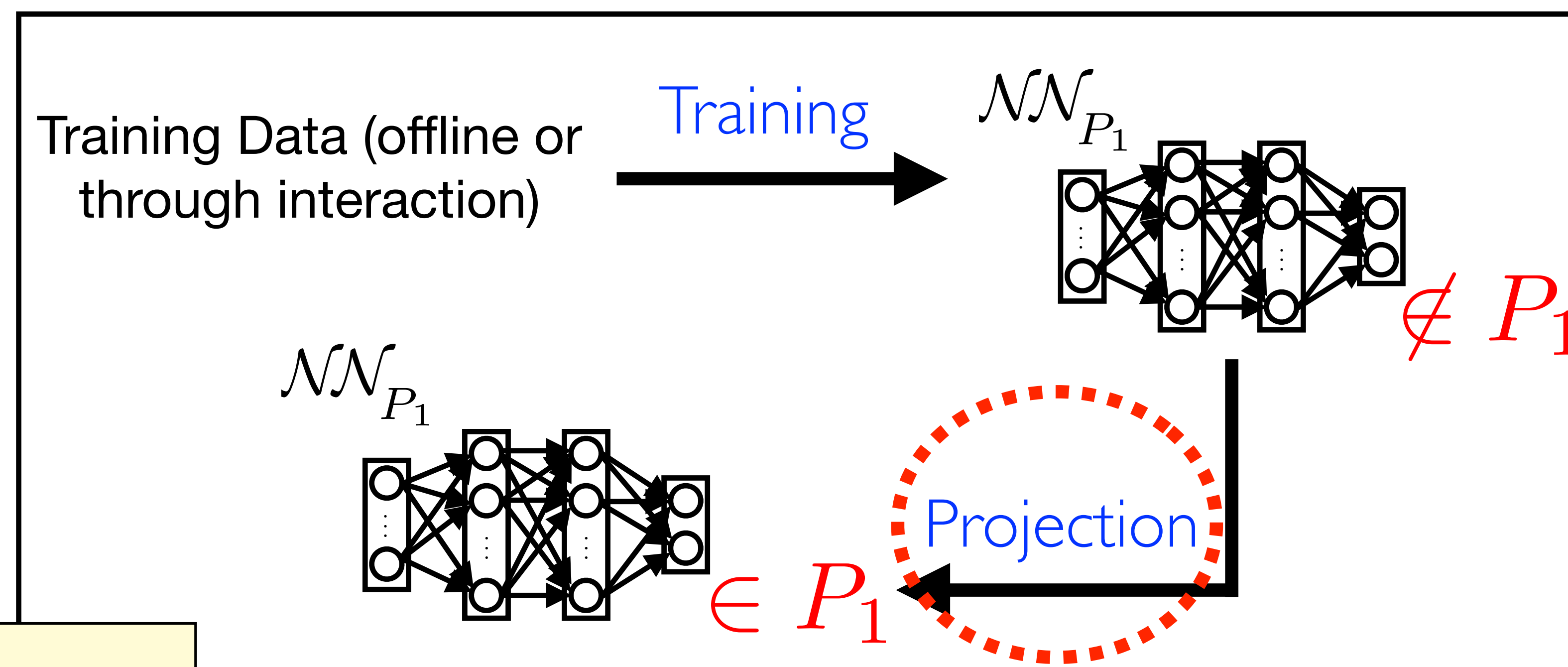
Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

Train a ***finite*** library of NNs offline to satisfy ***infinitely*** many tasks at runtime



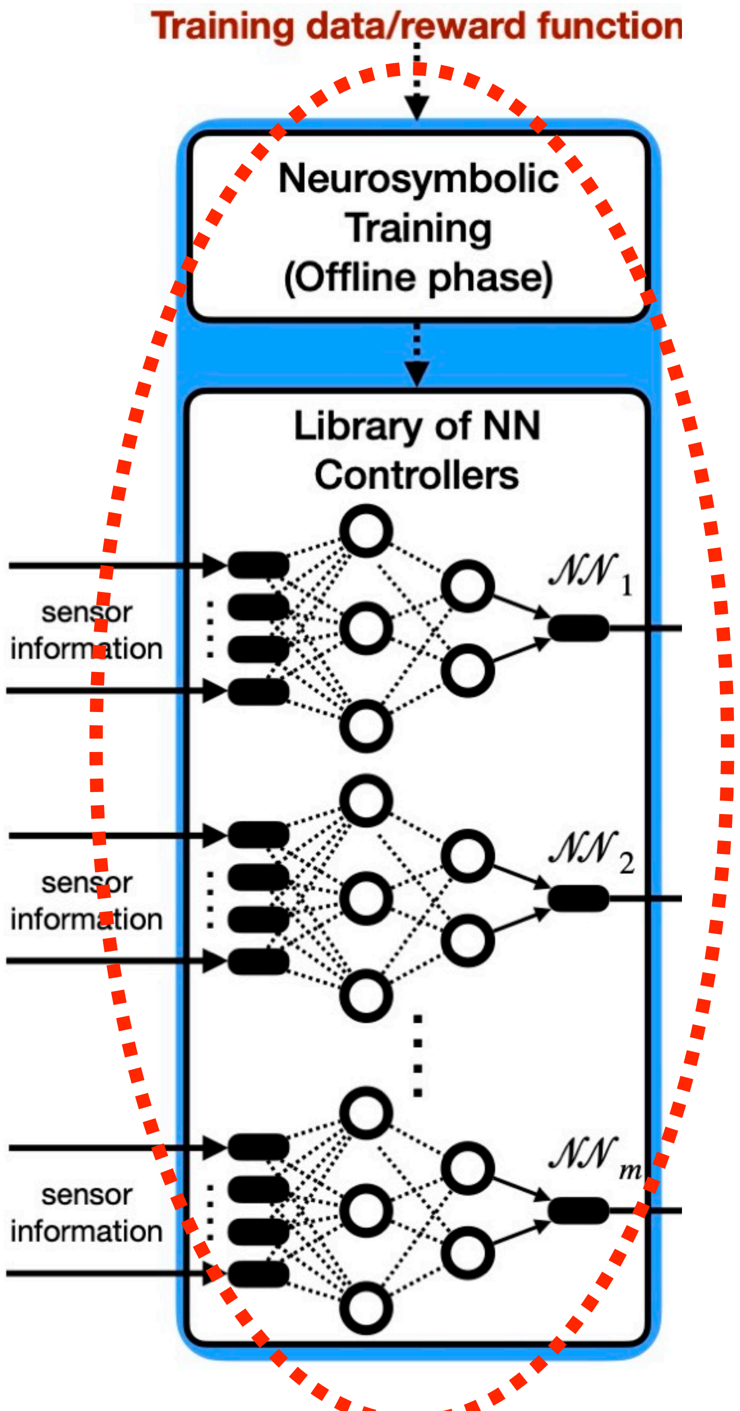
NN Weight Projection:



Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

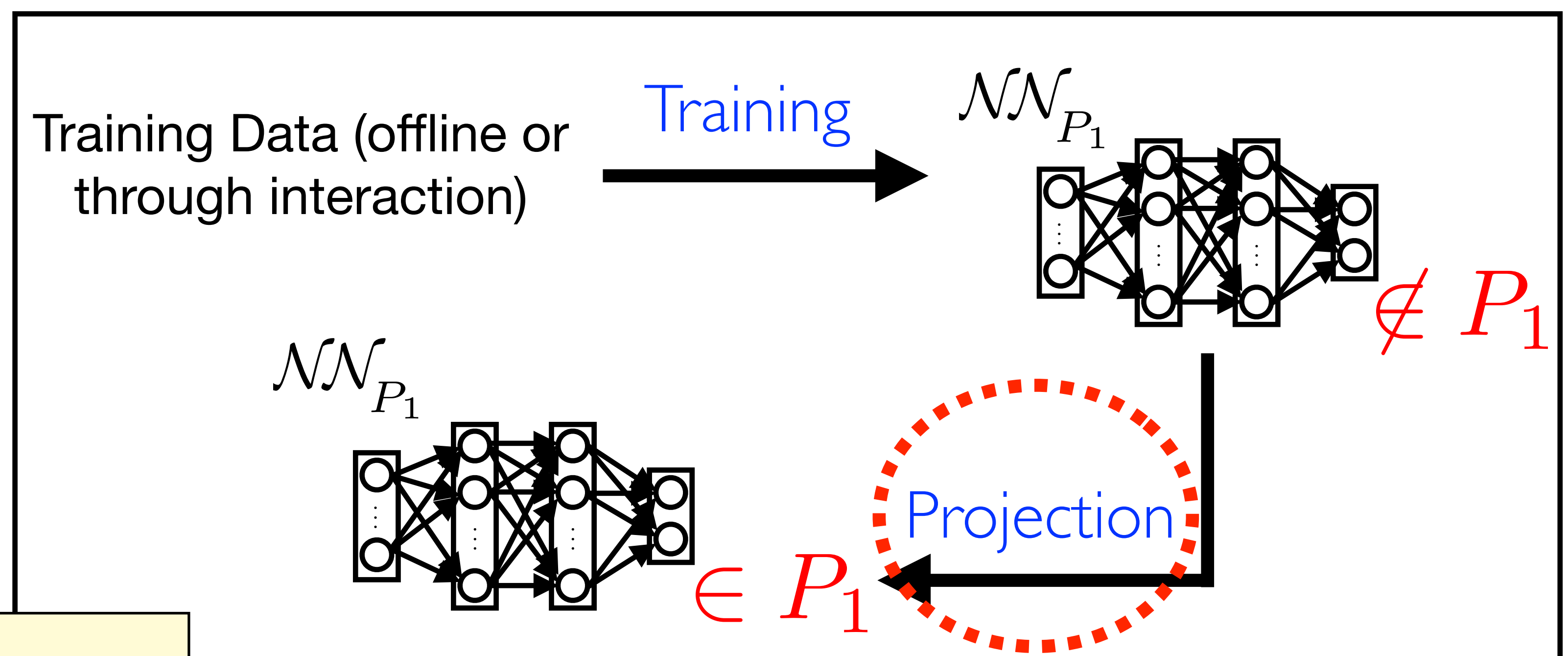
Train a *finite* library of NNs offline to satisfy *infinitely* many tasks at runtime



NN Weight Projection:

$$\underset{\widehat{W}^{(F)},\widehat{b}^{(F)}}{\operatorname{argmin}} \max_{x \in q} \|\mathcal{NN}_{\widehat{\theta}}(x) - \mathcal{NN}_{\theta}(x)\|_1$$

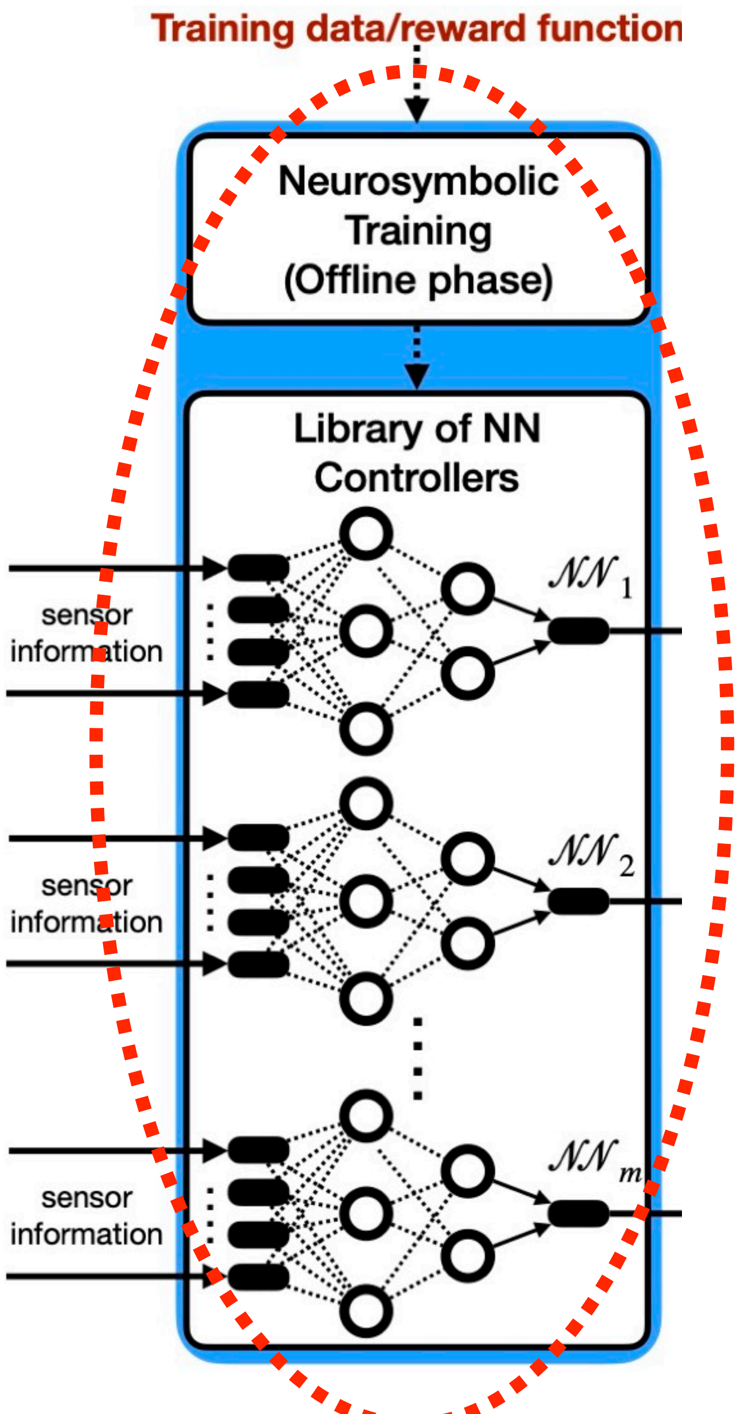
$$\text{s.t. } (\widehat{K}_i, \widehat{b}_i) \in P, \ \forall \mathcal{R}_i \in \{\mathcal{R} \in \mathbb{L}_{\mathcal{NN}_\theta} \mid \mathcal{R} \cap q \neq \emptyset\}$$



Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

Train a *finite* library of NNs offline to satisfy *infinitely* many tasks at runtime

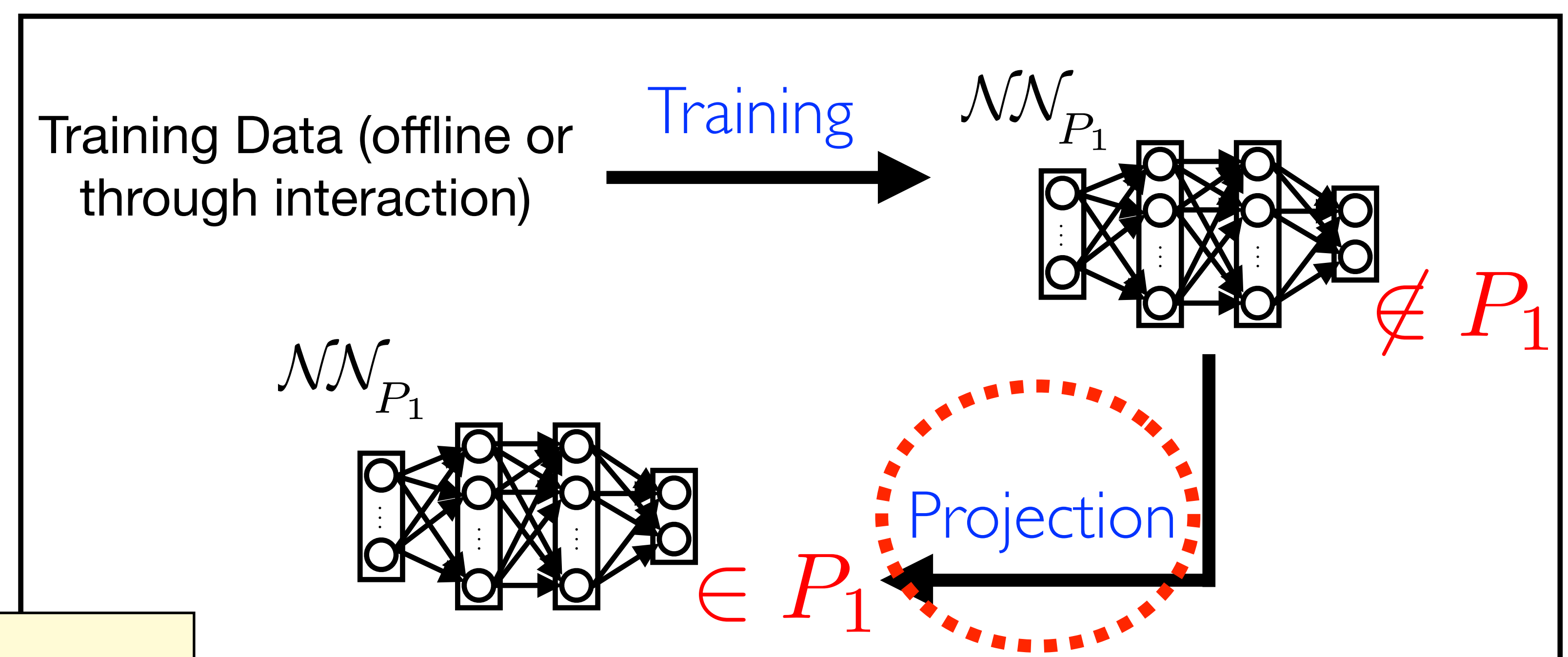


NN Weight Projection:

$$\underset{\widehat{W}^{(F)},\widehat{b}^{(F)}}{\operatorname{argmin}} \; \max_{x\in q} \|\mathcal{NN}_{\widehat{\theta}}(x) - \mathcal{NN}_\theta(x)\|_1$$

$$\text{s.t. } (\widehat{K}_i,\widehat{b}_i)\in P,\ \forall \mathcal{R}_i \in \{\mathcal{R}\in \mathbb{L}_{\mathcal{NN}_\theta} \mid \mathcal{R}\cap q \neq \emptyset\}$$

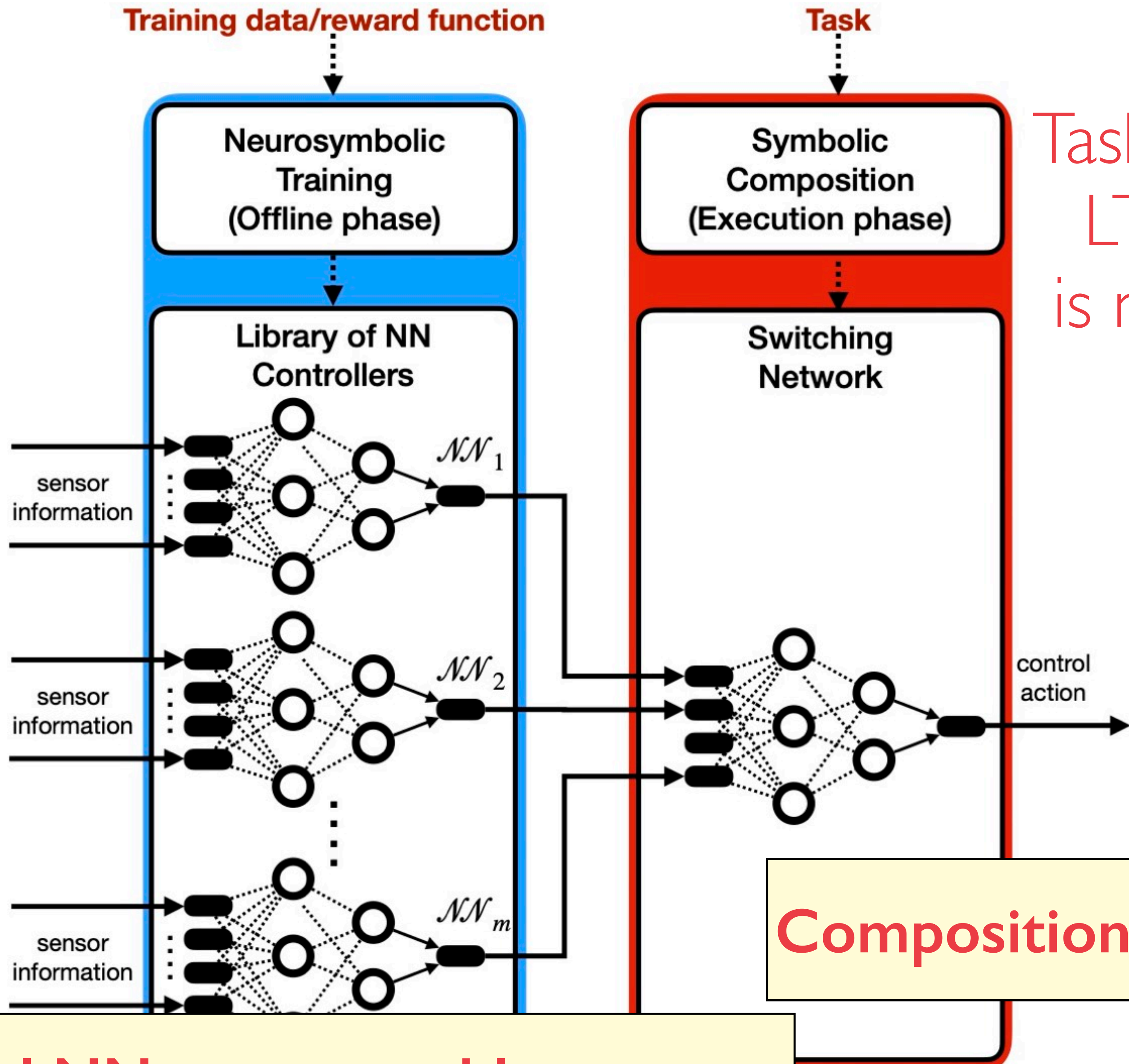
- Linear program.
- The change by projection $\max_{x\in q} \|\mathcal{NN}_{\widehat{\theta}}(x) - \mathcal{NN}_\theta(x)\|_1$ can be upper bounded.



Individual NNs are provably correct

Assured Meta Learning for LTL Tasks

Train a *finite* library of NNs offline to satisfy *infinitely* many tasks at runtime



Task = {workspace, obstacles, LTL mission, model error} is not known during training

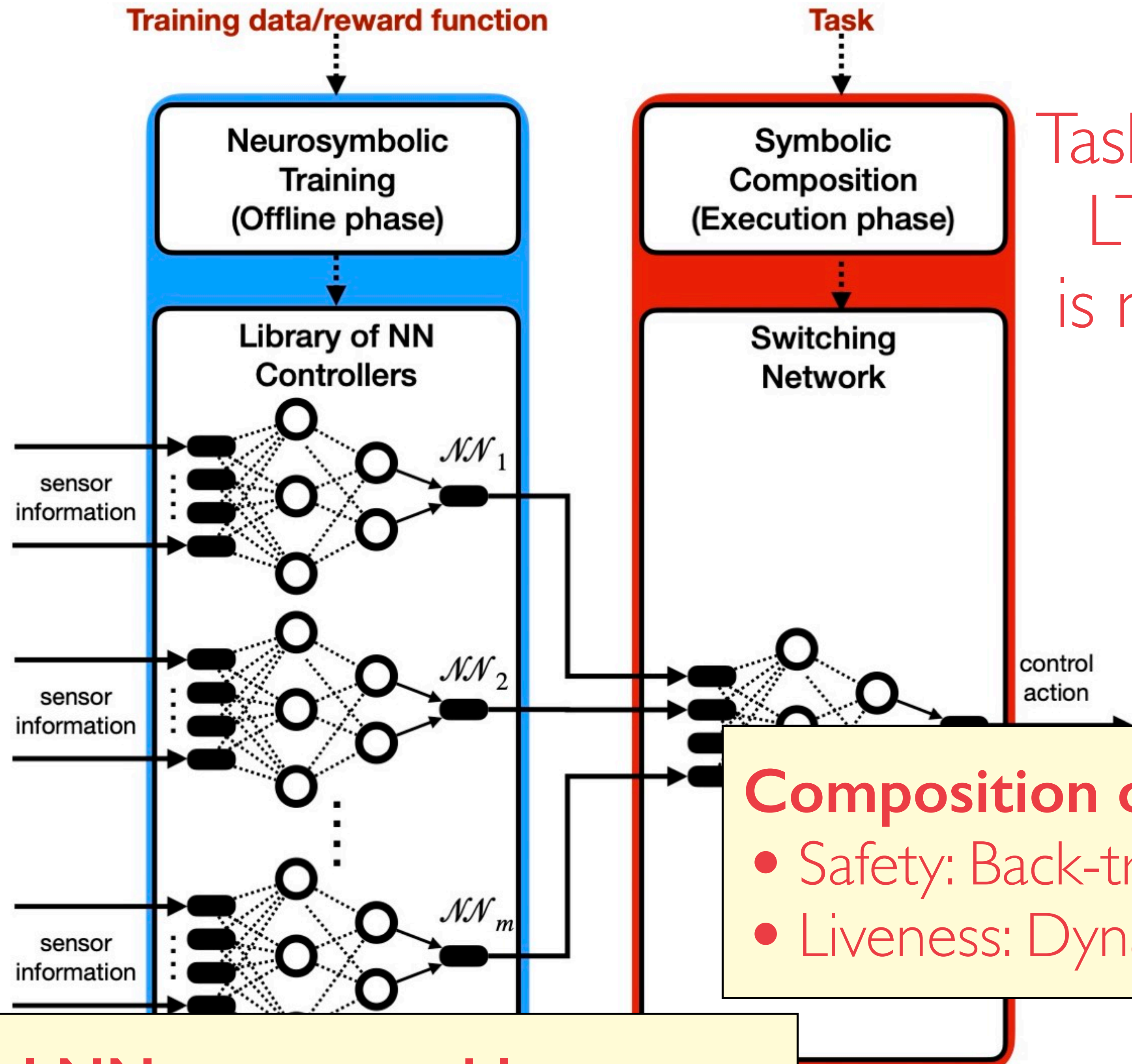
Composition of NNs is provably correct

Individual NNs are provably correct

- Construct finite MDP
- NN-Weight-Projection Training

Assured Meta Learning for LTL Tasks

Train a *finite* library of NNs offline to satisfy *infinitely* many tasks at runtime



Task = {workspace, obstacles, LTL mission, model error} is not known during training

Composition of NNs is provably correct

- Safety: Back-tracking
- Liveness: Dynamic Programming

Individual NNs are provably correct

- Construct finite MDP
- NN-Weight-Projection Training

Assured Meta Learning for LTL Tasks

Theorem (informal):

Consider the nonlinear system $x^+ = f(x,u) + g(x,u)$. Let $\mathfrak{NN}$ be the library of neural networks trained using the projected neural network training algorithm. For any arbitrary task

$\mathcal{T}$ = (workspace, error in dynamics, LTL specifications)

Then:

Activation map — space of CPWA functions (ReLU NNs)

$$\left|\Pr\left(\mathcal{NN}_{[\mathfrak{NN},\Gamma]} \models \varphi\right) - \max_{P\in\mathbb{P}} \Pr(P \models \varphi)\right| \le HZ\Delta^{\mathcal{NN}}$$

(i.e., $\mathcal{NN}_{[\mathfrak{NN},\Gamma]}$ can generalize to any task, if the task is achievable)

Individual NNs are provably correct

- Construct finite MDP
- NN-Weight-Projection Training

Assured Meta Learning for LTL Tasks

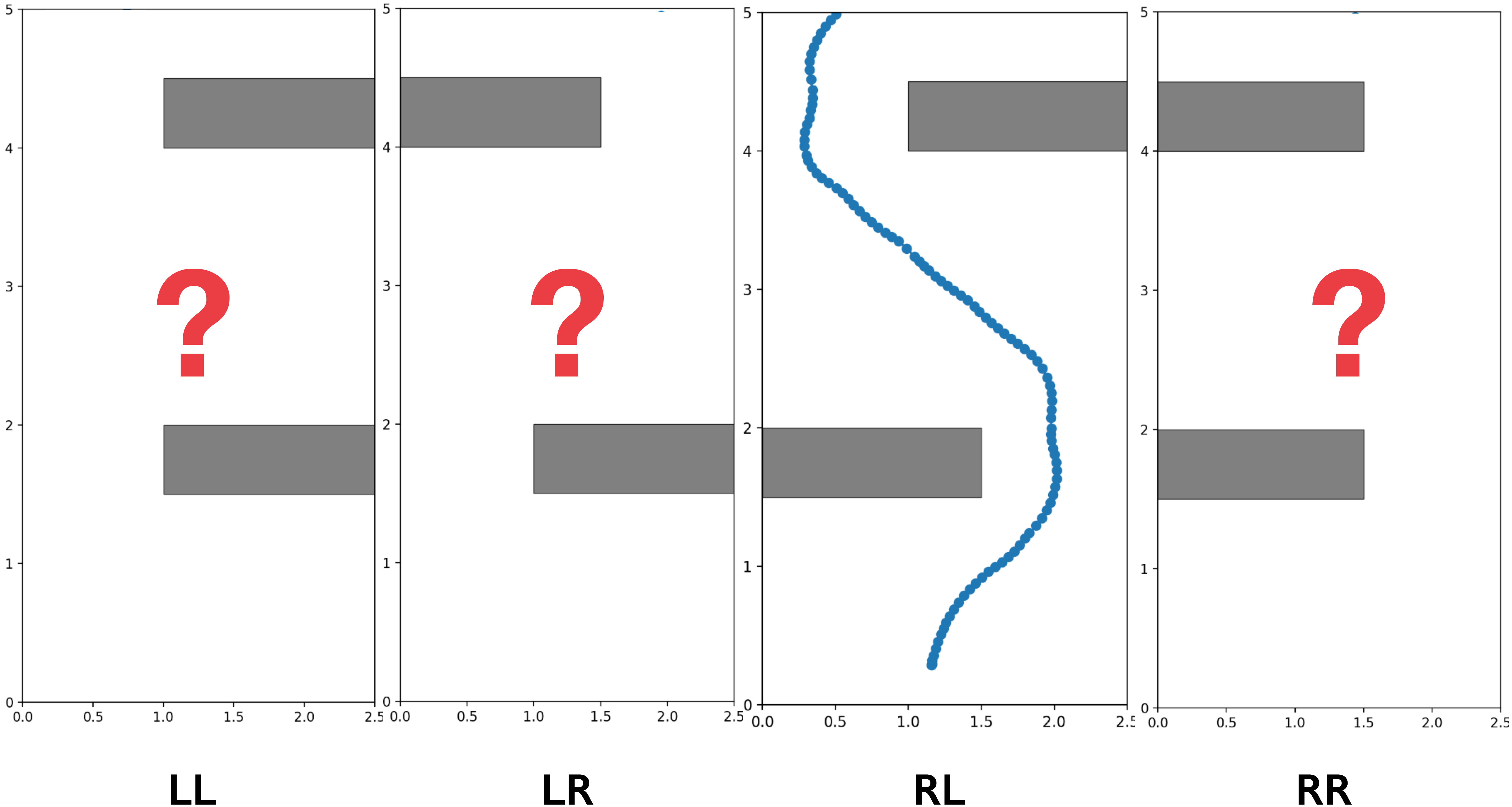
Practical Considerations:

- Do we need to train a full NN library $\mathfrak{NN}$?
  - No, we can use a partial library + formal transfer learning
  - We can obtain the same theoretical guarantees
- Can we used data collected from previous tasks to accelerate the framework?
  - Yes, expert data can be used to better train the NN library
  - It can also be used to accelerate the construction of the symbolic model

Individual NNs are provably correct

- Construct finite MDP
- NN-Weight-Projection Training

Comparison against Meta-RL

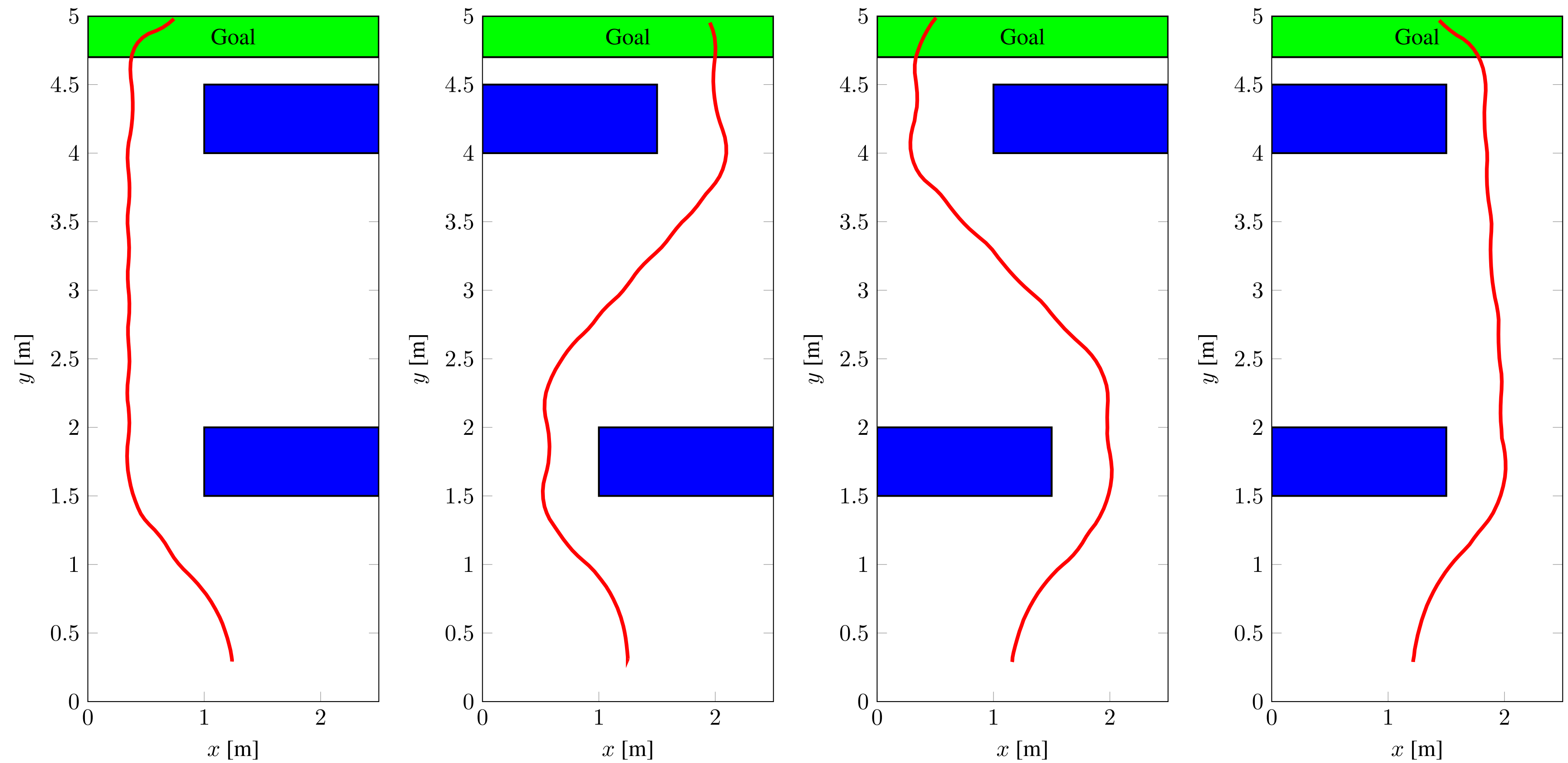


Trajectories will belong to different homotopy classes

Cao, Z., Kwon, M. and Sadigh, D., 2021. Transfer reinforcement learning across homotopy classes. *IEEE Robotics and Automation Letters*, *6*(2), pp.2706-2713.

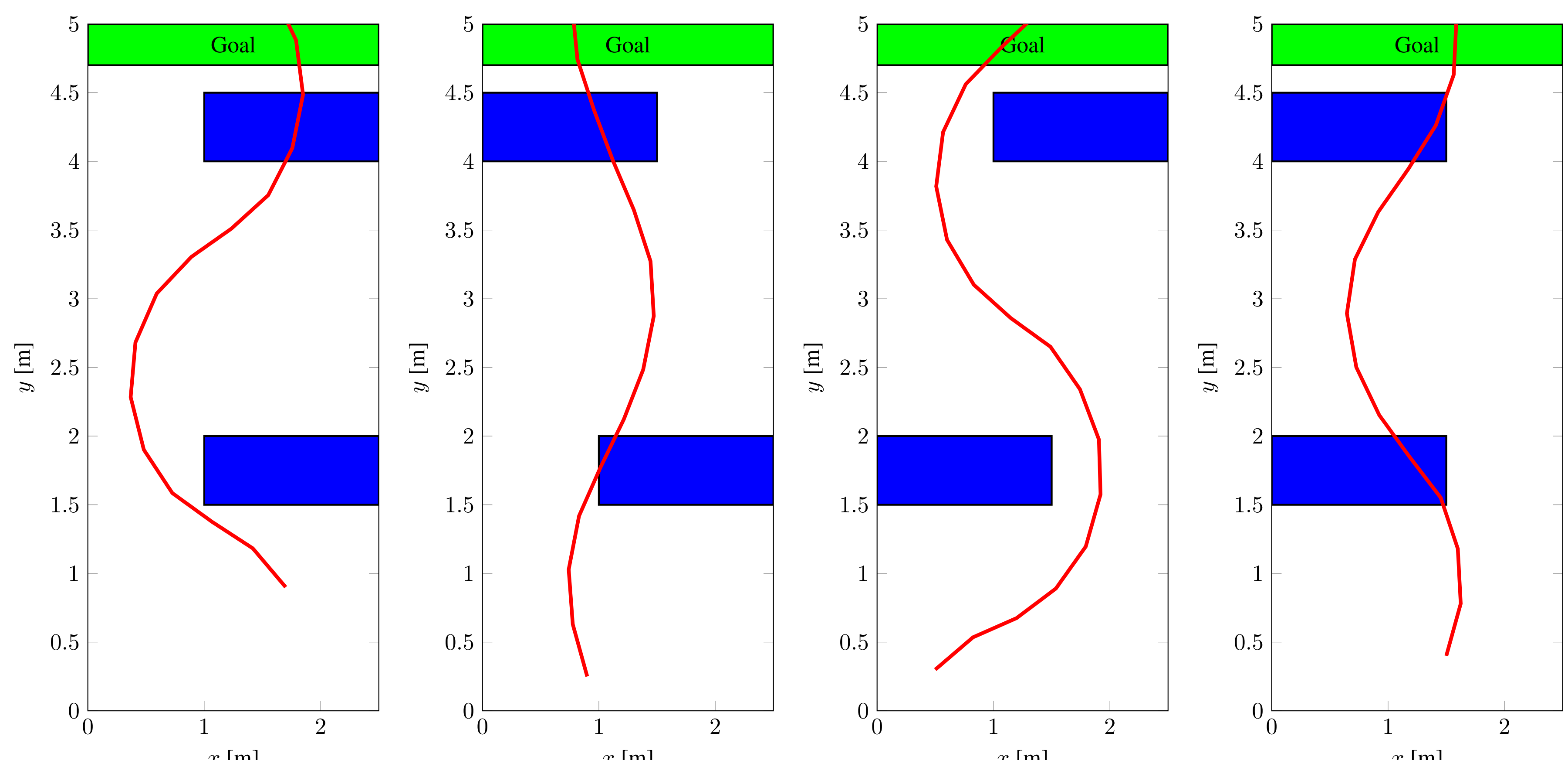
Resilient CPS Lab

Neurosymbolic RL Training

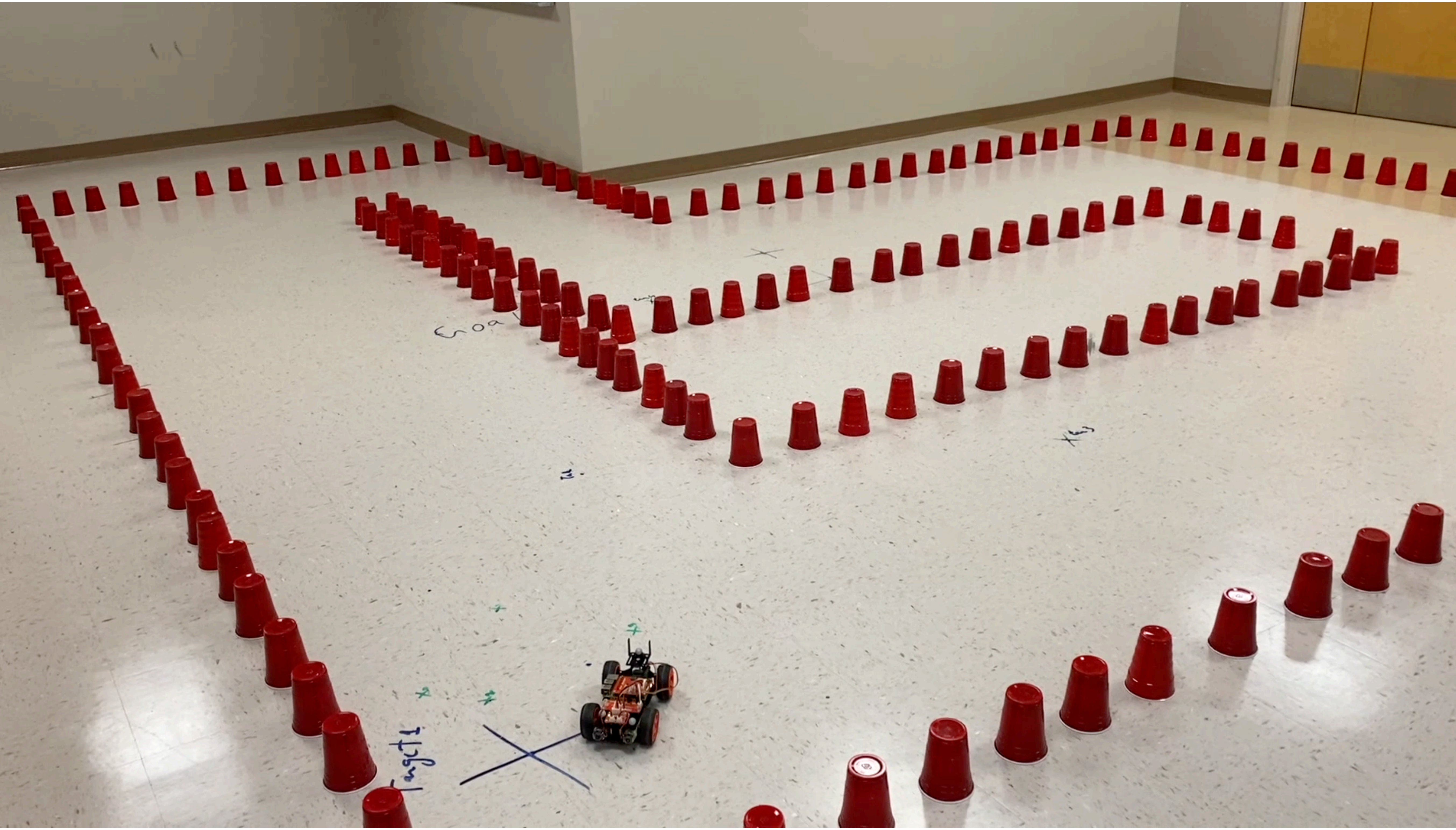


Off-policy metaRL algorithm (PEARL)

Rakelly, K., Zhou, A., Finn, C., Levine, S. and Quillen, D., 2019, May. Efficient off-policy meta-reinforcement learning via probabilistic context variables.
In *International conference on machine learning* (pp. 5331-5340). PMLR.

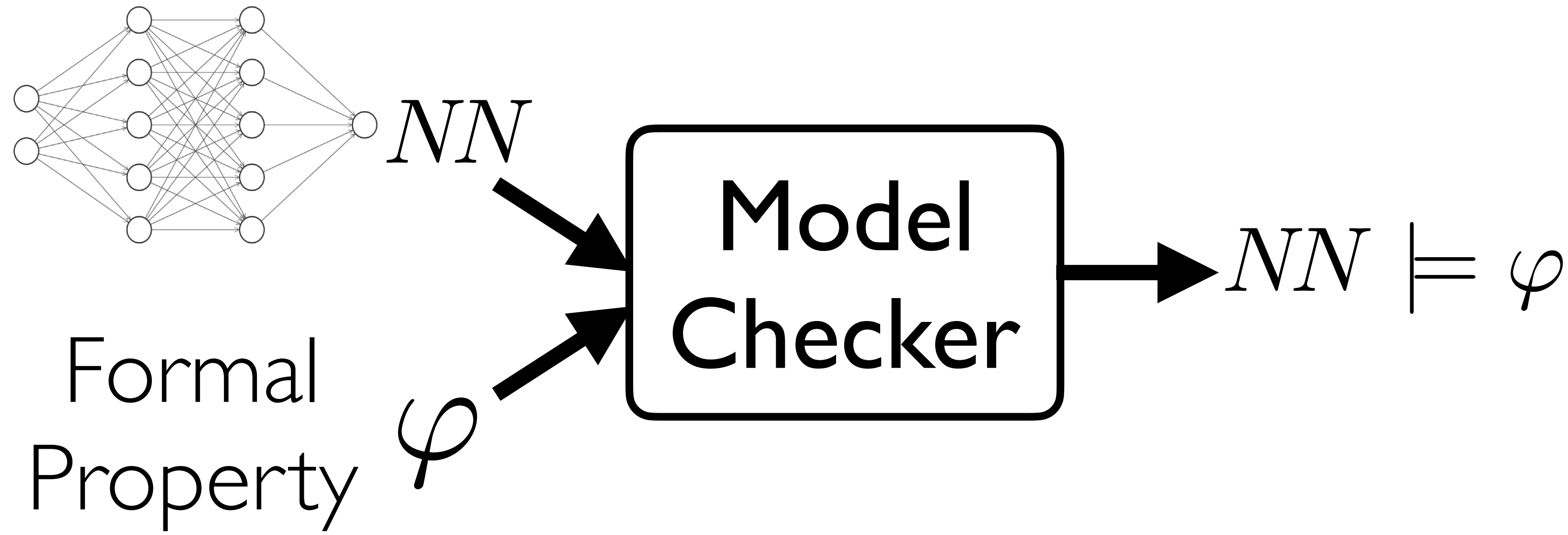


Xiaowu Sun and Yasser Shoukry, **"Neurosymbolic Motion and Task Planning for Linear Temporal Logic Tasks,"** T-RO, submitted.



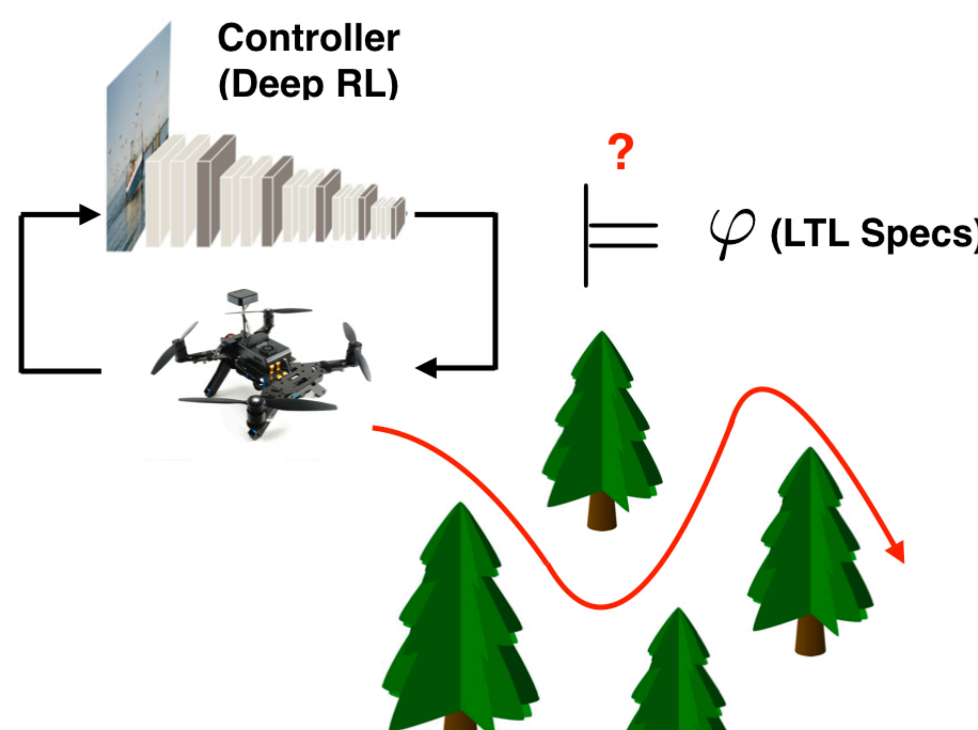
Xiaowu Sun and Yasser Shoukry, **"Neurosymbolic Motion and Task Planning for Linear Temporal Logic Tasks,"** T-RO, submitted.

Formal Verification Tools for NN Analysis

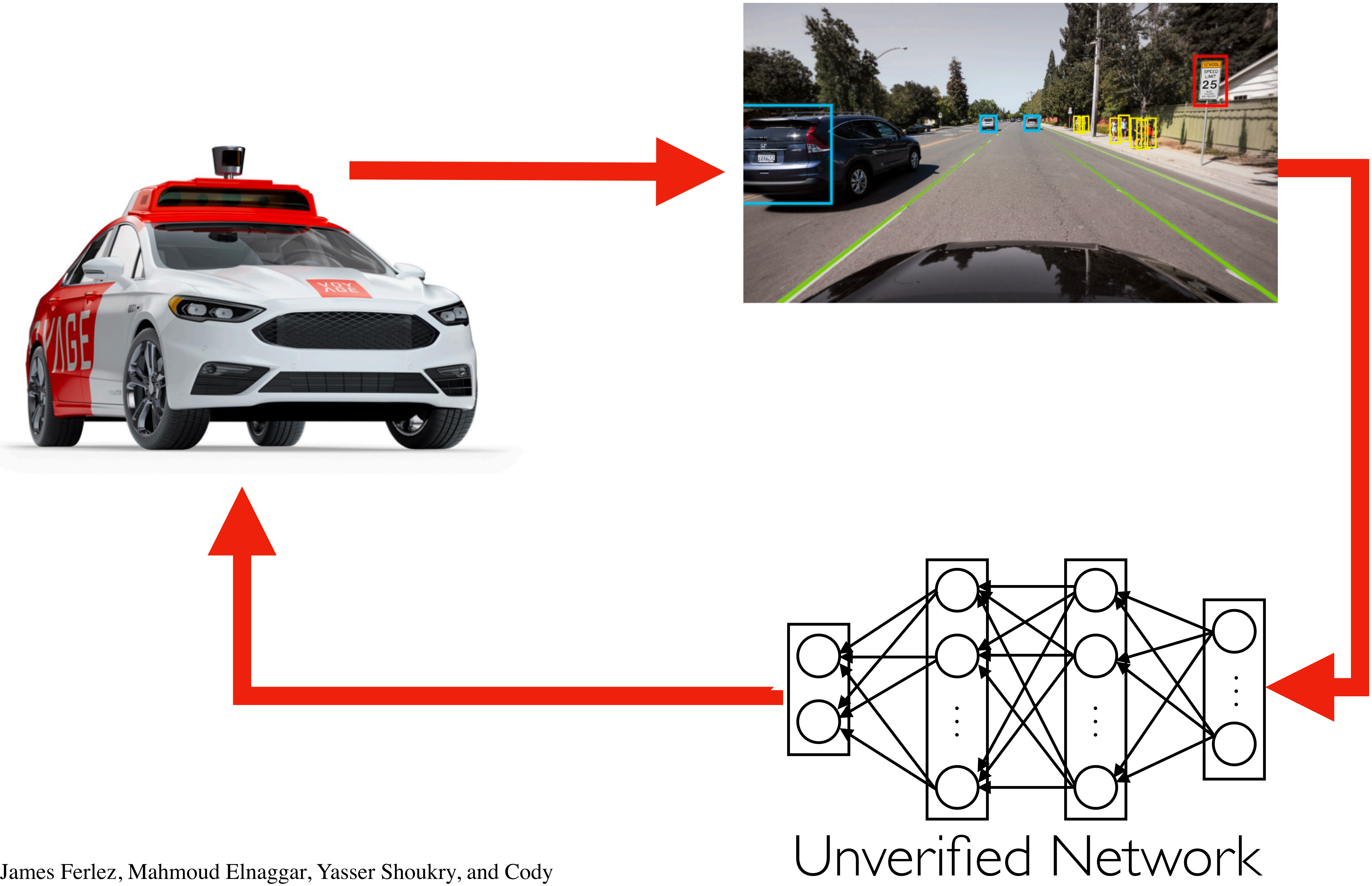


Assured NN-based Perception

Assured NN-based Control

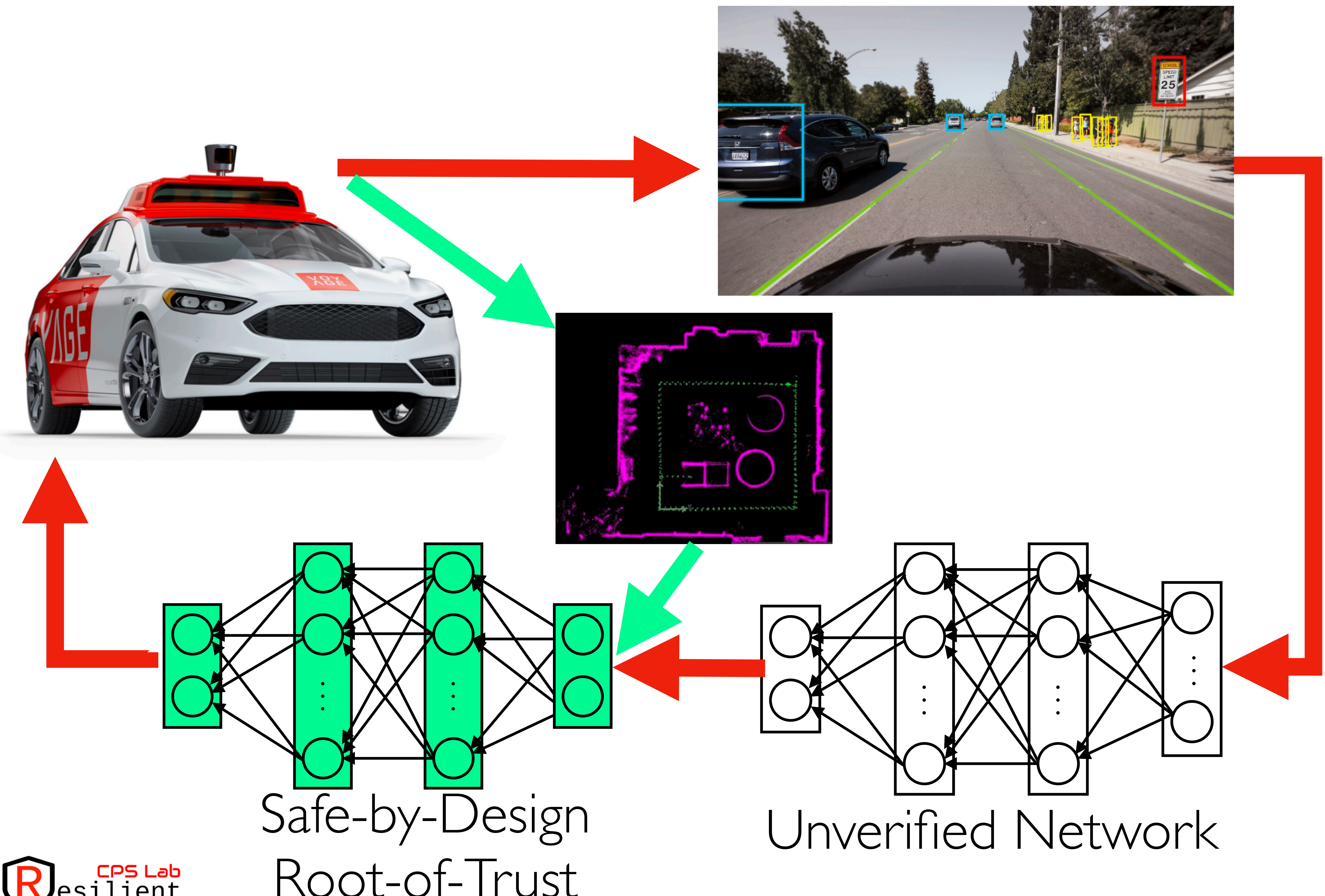


Synthesis of NN-based Safety Filters



James Ferlez, Mahmoud Elnaggar, Yasser Shoukry, and Cody Fleming, **[“ShieldNN: A Provably Safe NN Filter for Unsafe NN Controllers,”]** arXiv 2022.

Synthesis of NN-based Safety Filters



Synthesis of NN-based Safety Filters

James Ferlez, Mahmoud Elnaggar, Yasser Shoukry, and Cody Fleming, **"ShieldNN: A Provably Safe NN Filter for Unsafe NN Controllers,"** arXiv 2022.

Synthesis of NN-based Safety Filters

ROBUSTNESS DEMO #1

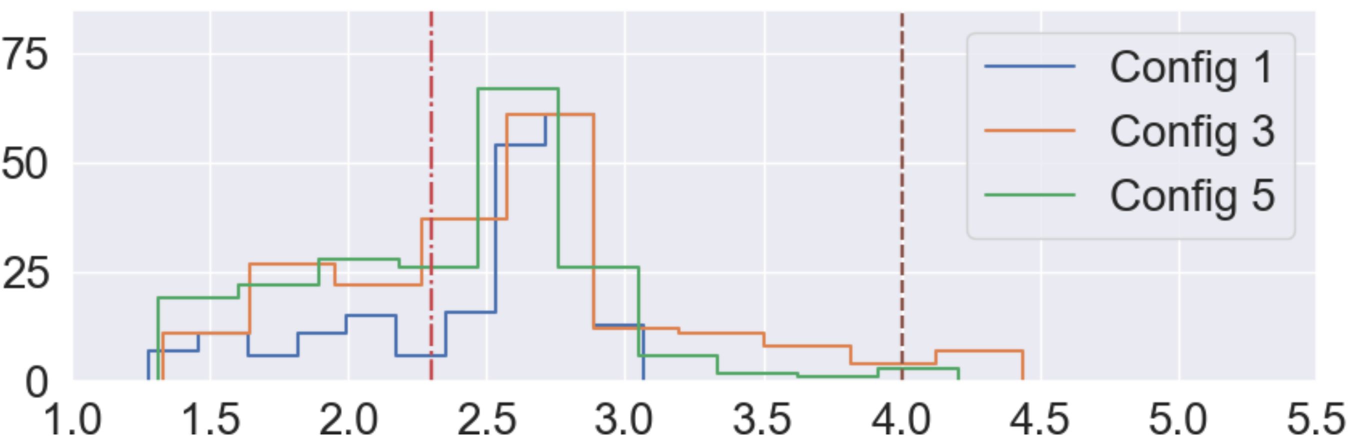
Agent #2

Agent #3

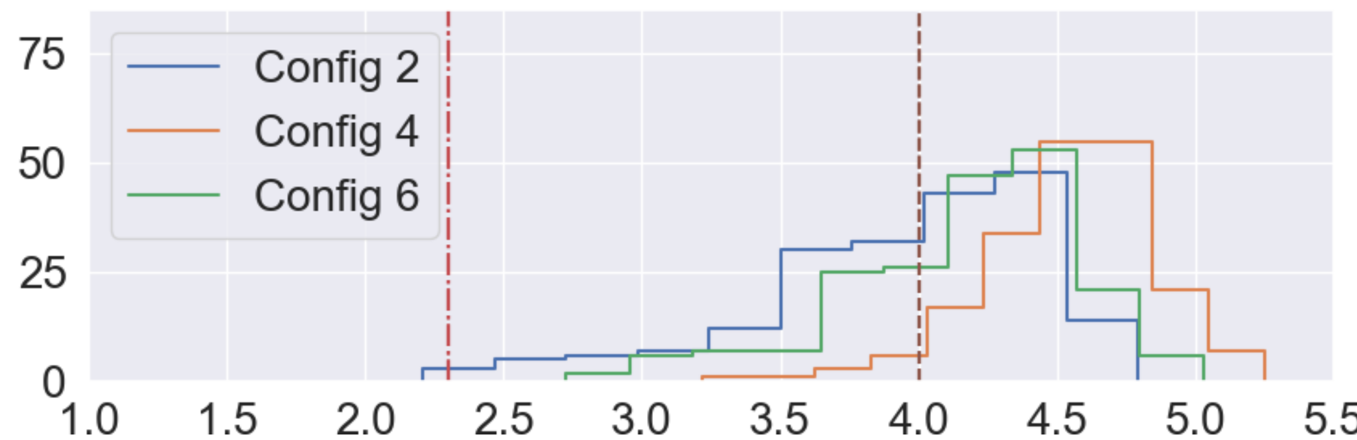
ShieldNN ON

James Ferlez, Mahmoud Elnaggar, Yasser Shoukry, and Cody Fleming, **"ShieldNN: A Provably Safe NN Filter for Unsafe NN Controllers,"** arXiv 2022.

Synthesis of NN-based Safety Filters



Without Root-of-Trust Network



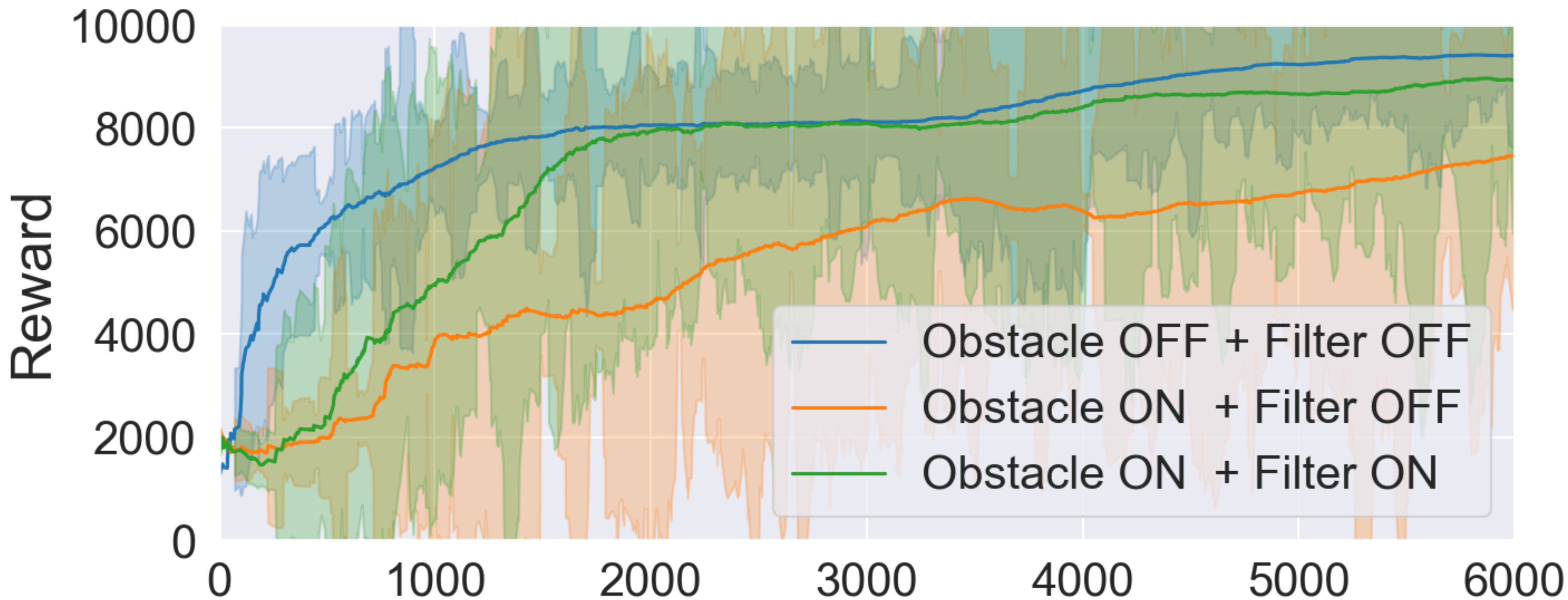
With Root-of-Trust Network

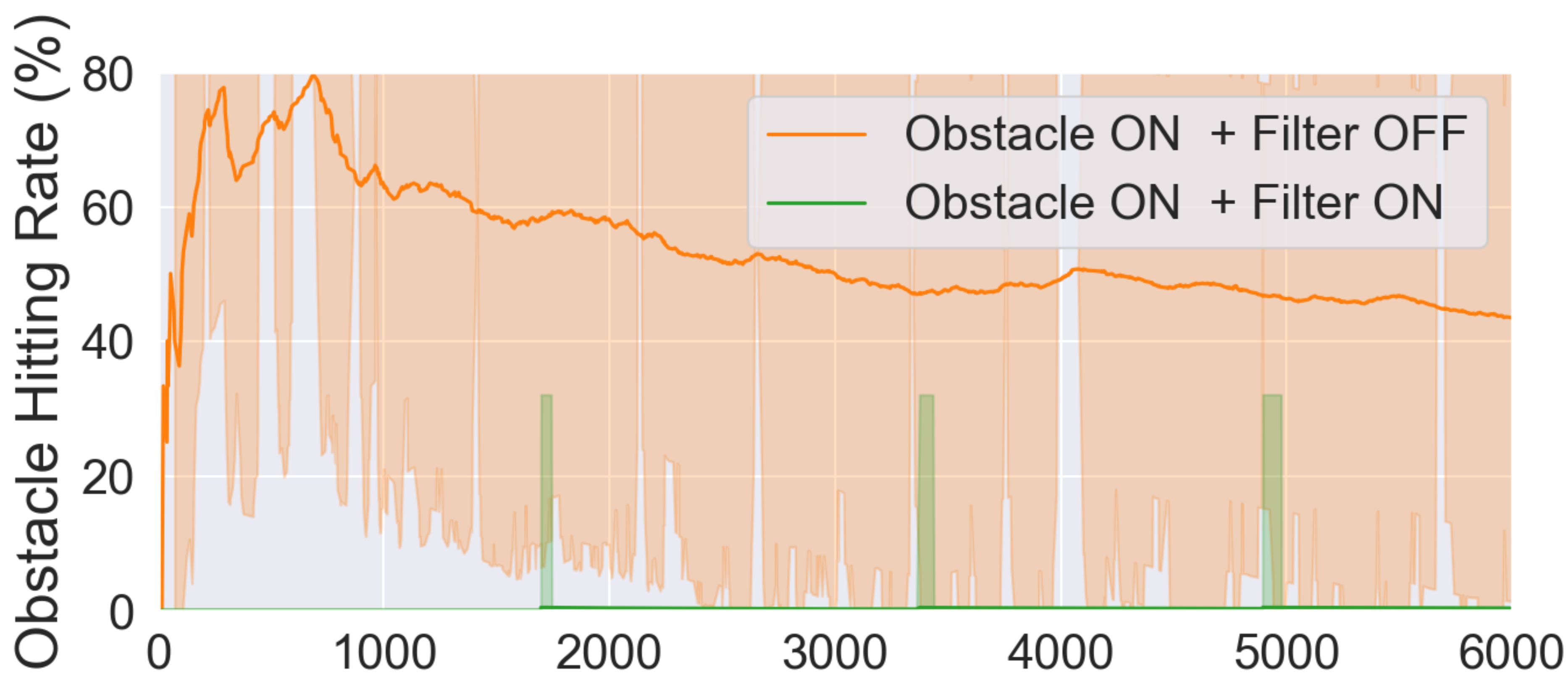
	Training		Testing	Experiment 1		Experiment 2	
Config	**Obstacle**	**Filter**	**Filter**	**TC%** [1]	**OHR%** [2]	**TC%** [1]	**OHR%** [2]
1	OFF	OFF	OFF	7.59	99.5	27.53	79.5
2	OFF	OFF	ON	98.82	0.5	98.73	0.5
3	ON	OFF	OFF	94.82	8.5	71.88	34
4	ON	OFF	ON	100	0	100	0
5	ON	ON	OFF	62.43	44	50.03	60
6	ON	ON	ON	100	0	100	0

[1] TC% := Track Completion % [2] OHR% := Obstacle Hit Rate %

James Ferlez, Mahmoud Elnaggar, Yasser Shoukry, and Cody Fleming, **"ShieldNN: A Provably Safe NN Filter for Unsafe NN Controllers,"** arXiv 2022.

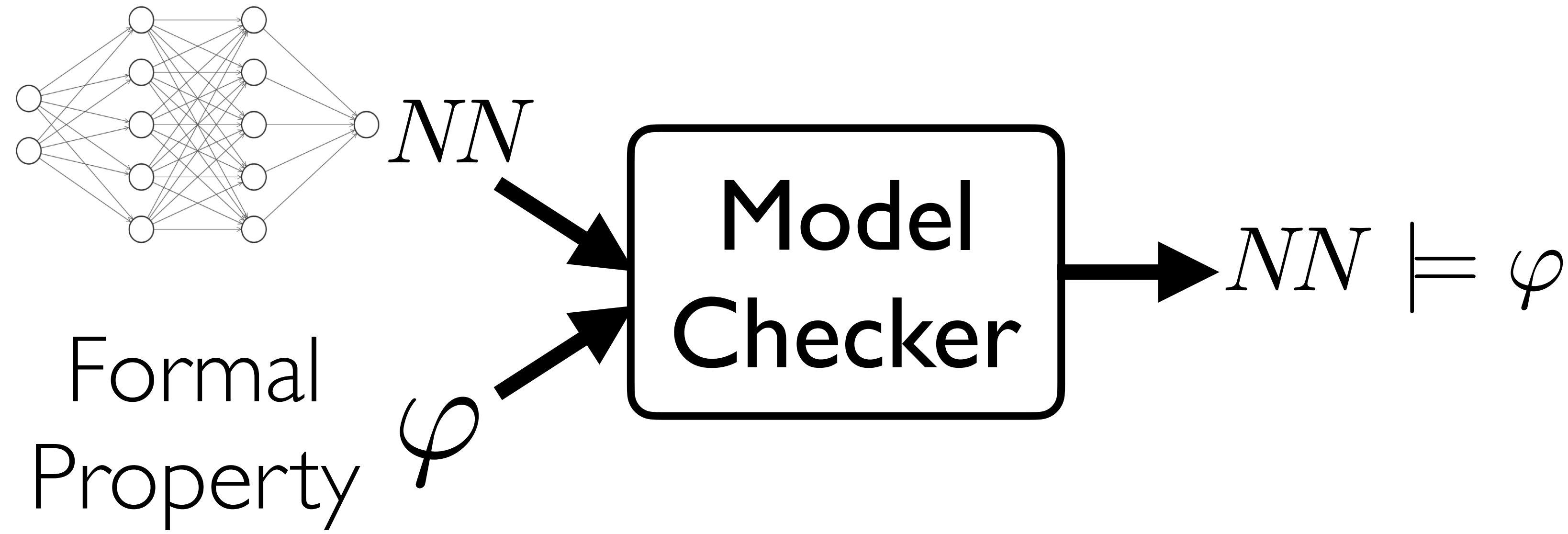
Synthesis of NN-based Safety Filters





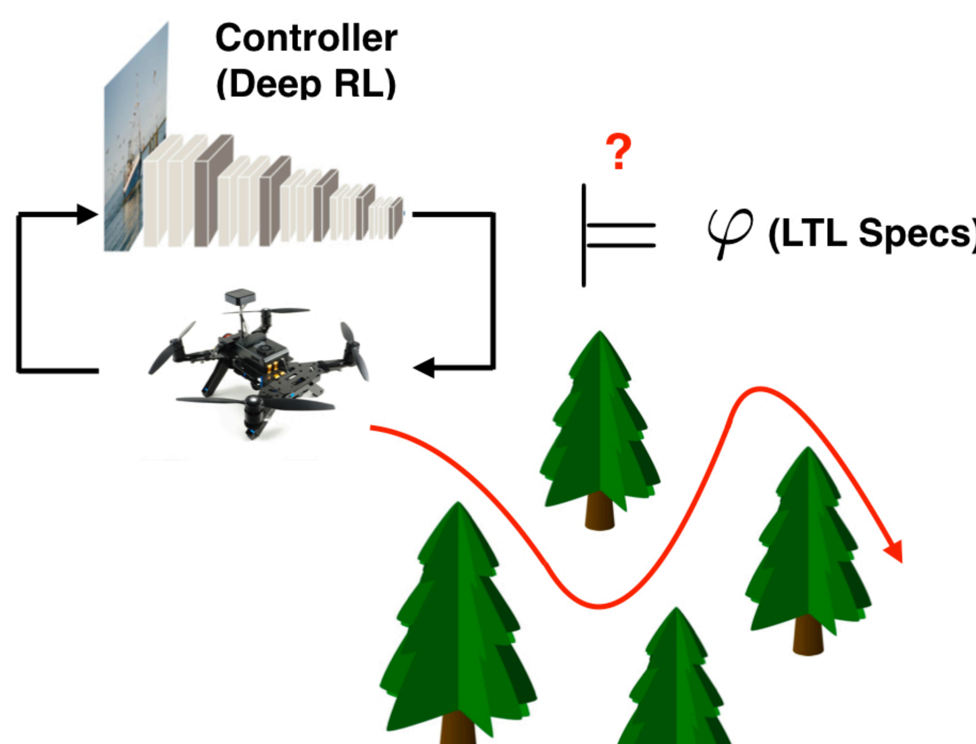
James Ferlez, Mahmoud Elnaggar, Yasser Fleming, **"ShieldNN: A Provably Safe N Controllers,"** arXiv 2022.

Formal Verification Tools for NN Analysis



Assured NN-based Perception

Assured NN-based Control



Resilient Cyber-Physical Systems Lab!

NORTHROP GRUMMAN

C3.ai Digital Transformation Institute

CPS Lab Resilient