Secure Planning Against Stealthy Attacks via Model-Free Reinforcement Learning

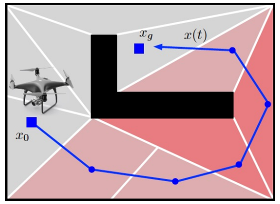
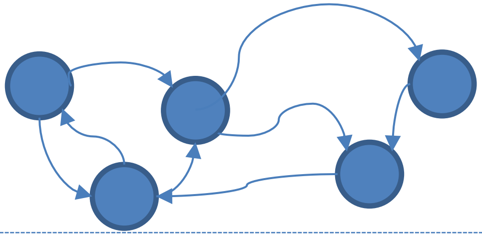
Miroslav Pajic

CPSL@Duke

Department of Electrical and Computer Engineering

Duke University

Security-Aware Autonomy

Control Stack	Control view	Modeling view
Mission Planner	Long-horizon views	
Tactical Planner	Short-horizon views	
Low-level Control	Continuous/discrete control with constraints	$f_r(x(t)) = \int_0^T \varrho(x(t), h(t))dt + \int_0^T \|x(t)\|^2 dt,$ $\min \; f_r(x_r(t)) + f_h(x_h(t))$ $\text{s.t.} \; x_r(t) = x_h(t), \;\; u_r(t) = u_h(t),$
Vehicle		

Adding Resiliency

[ICRA21a, ICRA21b, ICRA20, ICRA19, CAV'19a, THMS19]

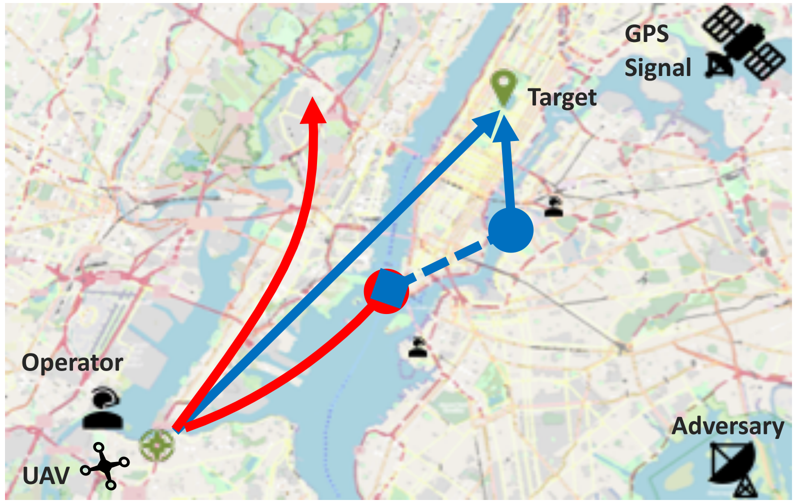
[Automatica21*, TII21, TASE21*, CDC19a, CDC19b, IoTDI19]

[ICML21*, TCPS20, ACC20, AUT21b*, AUT21, AUT18, TECS17, RTSS17, TCNS17a, TCNS17b, CSM17, CDC17, CDC18,…]

Our Goal: Add resiliency to controls across different/all levels of the autonomy stack

Problem Setting

- **Controller**
  - aims to perform a given task in an unknown stochastic environment
  - has a perfect knowledge of the current state
  - has an intrusion-detection system (IDS) that monitors anomalies
  - can detect attacks only when the IDS raises an alarm



- **Attacker**
  - aims to prevent the controller from performing the given task
  - has a perfect knowledge of the current state, the controller strategy and the IDS mechanism
  - can attack on actuators unless detected
  - tends to stay stealthy

Secure Planning Objective

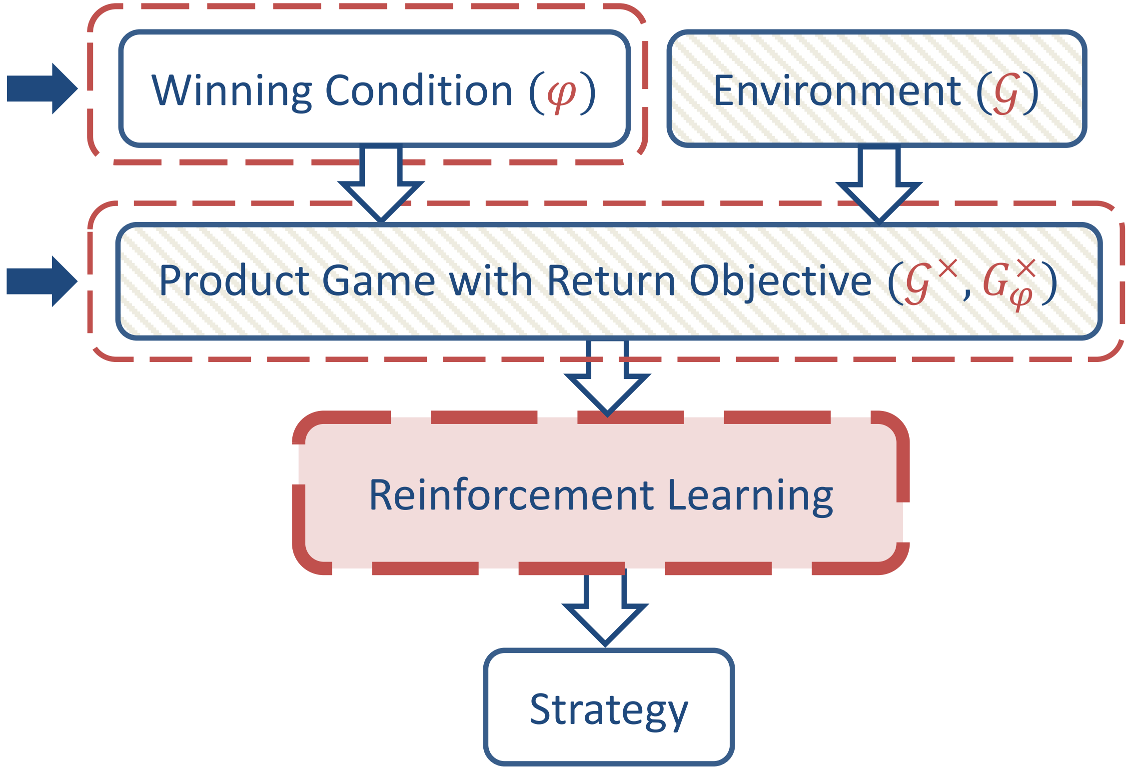
- **Problem:**
  - For a given task and the IDS mechanism, learn an optimal controller strategy resilient to stealthy attacks on actuators

- **Three-Step Solution [1]:**
  - Model the problem as a zero-sum SG $\mathcal{G}$ with an LTL winning condition φ capturing
    - the controller task
    - the IDS mechanism
    - the behavior of stealthy attackers
  - Reduce the LTL objective $argmax_{\mu}\ min_{\nu}\ Pr_{\mu,\nu}(\mathcal{G} \vDash \varphi)$ to a return objective:

$$argmax_{\mu}\ min_{\nu}\ \mathbb{E}_{\mu,\nu}\left[G_{\varphi}^{\times}\right]$$

$$argmax_{\mu}\ min_{\nu}\ \mathbb{E}_{\mu,\nu}\left[\sum_{i=0}^{\infty} \gamma^{i} r_{(i)}\right]$$

  - Learn an optimal controller strategy using a model-free RL



[1] A. K. Bozkurt, Y. Wang, and M. Pajic. "Secure Planning Against Stealthy Attacks via Model-Free Reinforcement Learning". ICRA, 2021, accepted.

LTL Winning Condition

- $\boldsymbol{\varphi}_{\mathbf{TASK}}$:
  - LTL specification of the given task
  - Surveillance Example:

$$\varphi_{\text{TASK}} = \square\lozenge\, \text{region}_1 \wedge \square\lozenge\, \text{region}_2$$

- $\boldsymbol{\varphi}_{\mathbf{IDS}}$:
  - LTL specification of the intrusion detection system
  - A reachability specification satisfied when an attack is detected
  - Attacks can be detected only after reaching the high-alert mode triggered by the anomalies
  - Counting-Based IDS Example:

$$\varphi_{\text{IDS}} = \lozenge\Big(\text{anomaly} \wedge \bigcirc(\text{anomaly} \wedge \bigcirc\lozenge^{\leq T}\text{attack})\Big)$$

    - Two consecutive anomalies triggers the high-alert mode
    - The attacks can be detected during the high-alert mode

- **Winning Condition:** $\boldsymbol{\varphi} = \boldsymbol{\varphi}_{\mathbf{IDS}} \vee \boldsymbol{\varphi}_{\mathbf{TASK}}$:
  - $\neg\varphi = \neg\varphi_{IDS} \wedge \neg\varphi_{TASK}$ reflects the behavior of stealthy attackers
  - Being detected results in losing the game; thus, the attacker always stays hidden
  - The only way for the attacker to win to prevent the controller performing the task

Performing Tasks After Attack Detection

Duke
PRATT SCHOOL of ENGINEERING

- **Satisfaction of φ_{TASK}:**
  - The task needs to be performed even after the attacker is eliminated
  - An attack could prevent performing the task even if it is detected
  - Safety Example:

$$\varphi_{\mathrm{TASK}} = \Box \neg \mathrm{unsafe}$$

  - Recovering from an unsafe state is not possible; although being eliminated the attacker should win the game

- **Allowing for a single attack:**
  - φ_{IDS} can be easily modified to capture such cases
  - An attack after a detected attack satisfies φ_{IDS}
  - Counting-Based IDS Example:

$$\varphi_{\mathrm{IDS}} = \Diamond \Big(\text{anomaly} \wedge \bigcirc \big(\text{anomaly} \wedge \bigcirc \Diamond^{\leq T} (\text{attack} \wedge \bigcirc \Diamond \text{attack}) \big) \Big)$$

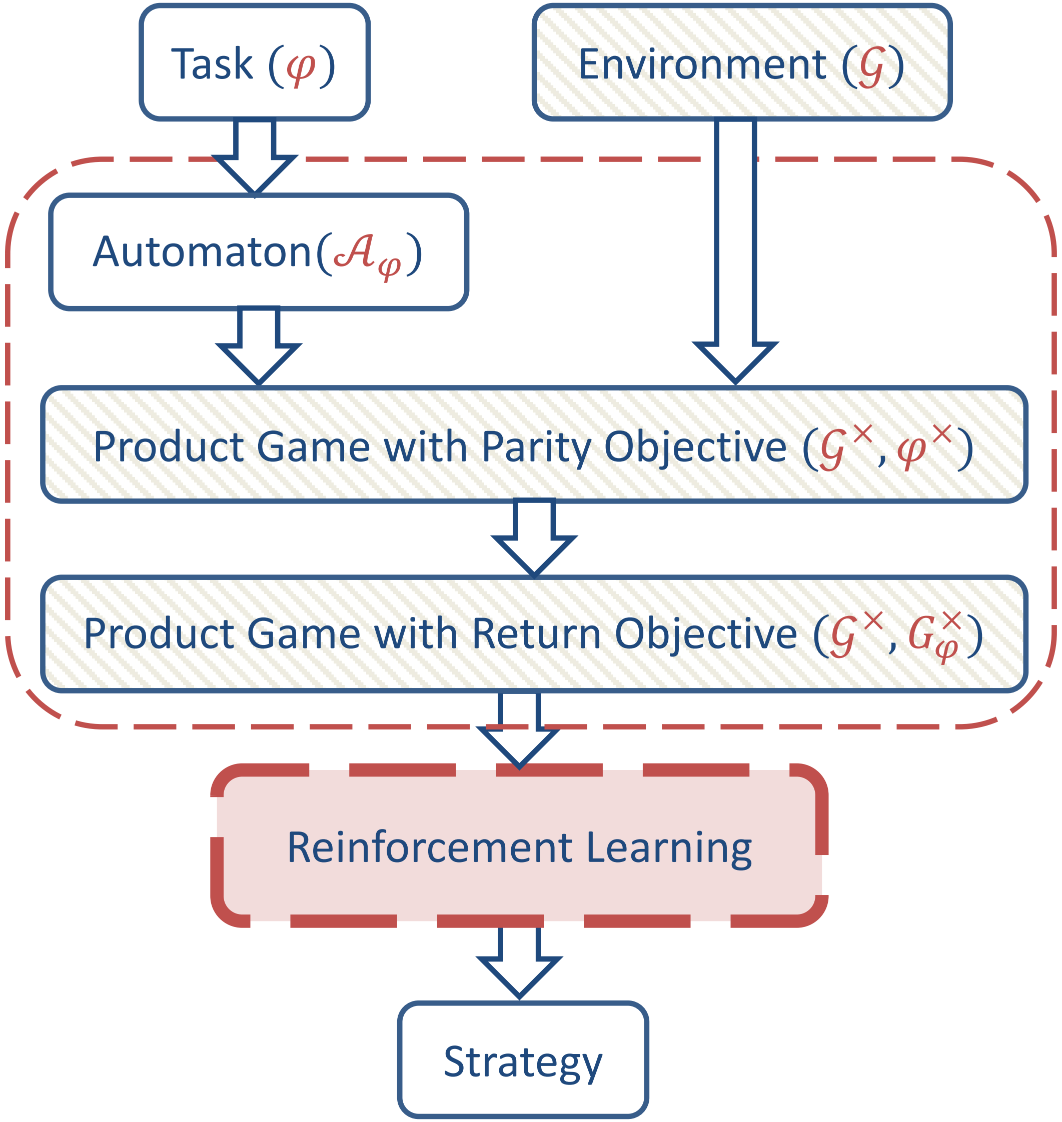
  - Being eliminated is equivalent to not attacking after a detected attack

RL Framework for LTL

- **Reduction Steps:**
  - LTL -> Automaton
  - Product Game Construction
  - Reduction from Parity to Return
  - Model-free Learning

- **Parity to Return I (Multiple Rewards Discount Factors) [2]:**
  - $Pr_{\mu,\nu}(\mathcal{G}^\times \models \varphi^\times) = \lim_{r_\varphi \to 0^+} \mathbb{E}_{\mu,\nu}\left[\sum_{i=0}^{\infty}\left(\prod_{j=1}^{i} \Gamma_\varphi\left(s^\times_{(j)}\right)\right) R_\varphi\left(s^\times_{(i)}\right)\right]$
    - $R_\varphi(s^\times) \coloneqq r_\varphi^{k-Color(s^\times)} \mathbf{1}_{\{Color(s^\times) \text{ is } even\}}$
    - $\Gamma_\varphi(s^\times) \coloneqq 1 - r_\varphi^{k-Color(s^\times)}$

- **Parity to Return Objectives II (Priority Reward Machines) [3]:**
  - $Pr_{\mu,\nu}(\mathcal{G}^\times \models \varphi^\times) = \lim_{\varepsilon_\varphi \to 0^+} \mathbb{E}_{\mu,\nu}\left[\sum_{i=0}^{\infty}\left(1 - \varepsilon_\varphi\right)^i R^\star_\varphi\left(s^\times_{(i)}, \varrho_{(\mathrm{i})}\right)\right]$
    - ε_φ: PRM transition probability
    - $\varrho_{(\mathrm{i})}$: PRM state
    - $R^\star_\varphi$: PRM reward

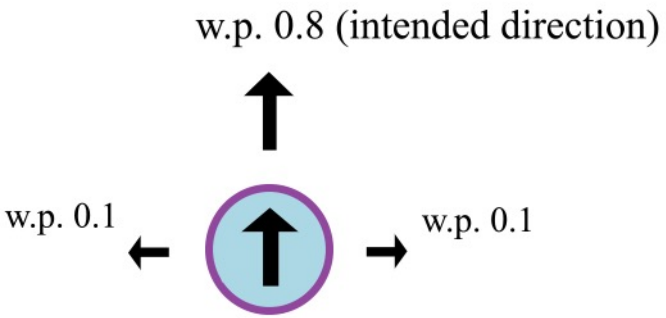


[2] A. K. Bozkurt, Y. Wang, M. M. Zavlanos, and M. Pajic. "Model-Free Reinforcement Learning for Stochastic Games with Linear Temporal Logic Objectives". ICRA, 2021, accepted.
[3] A. K. Bozkurt, Y. Wang, and M. Pajic. "Learning Optimal Strategies for Temporal Tasks in Stochastic Games". 2021, submitted.

Case Studies: Grid Worlds

Duke
PRATT SCHOOL of ENGINEERING

- **Grid World**
  - The agent (i.e., the controller) can take four actions: *North*, *South*, *East*, *West*
  - The agent moves in the intended direction w.p. 0.8 and sideways w.p. 0.2
  - The attacker can override the controller action
  - A movement is called an anomaly if it is not in the intended direction



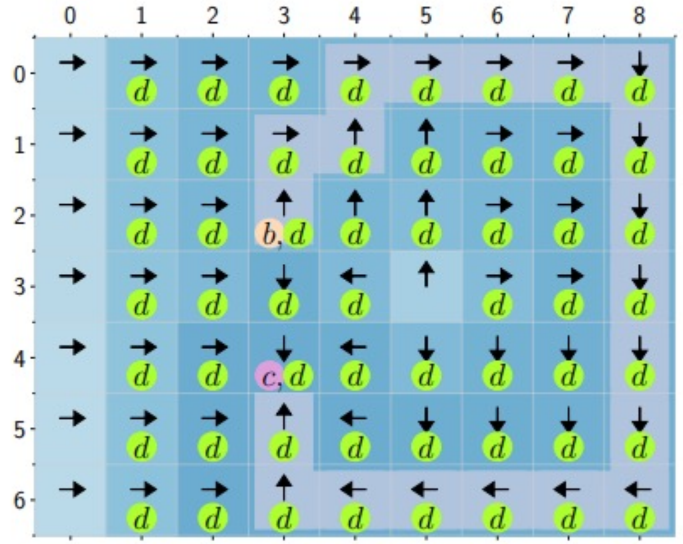
- **IDS:**
  - Two consecutive anomalies triggers the high-alert mode for the next two time steps

$$\varphi_{\mathrm{IDS}} = \Diamond\Big(\text{anomaly} \wedge \bigcirc\Big(\text{anomaly} \wedge \bigcirc\Diamond^{\leq 1}(\text{attack} \wedge \bigcirc\Diamond\text{attack})\Big)\Big)$$

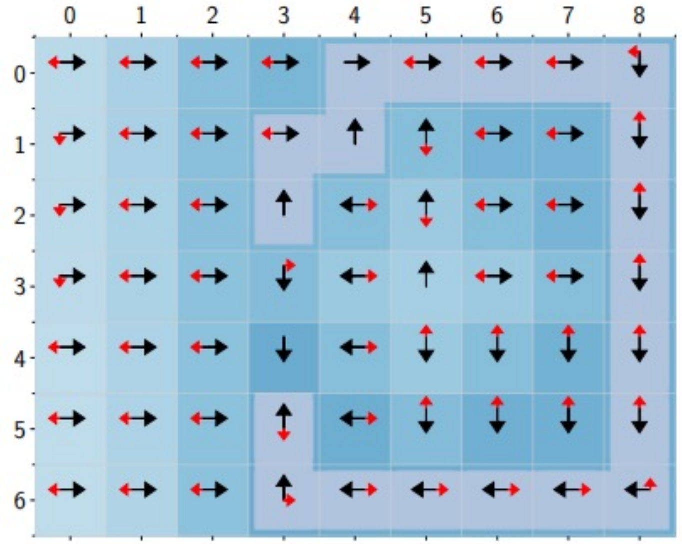
Case Study I: Surveillance

- **Task:**
    - Repeatedly visit a b and a c cell
    - Eventually reach a safe region labeled with d and do not leave

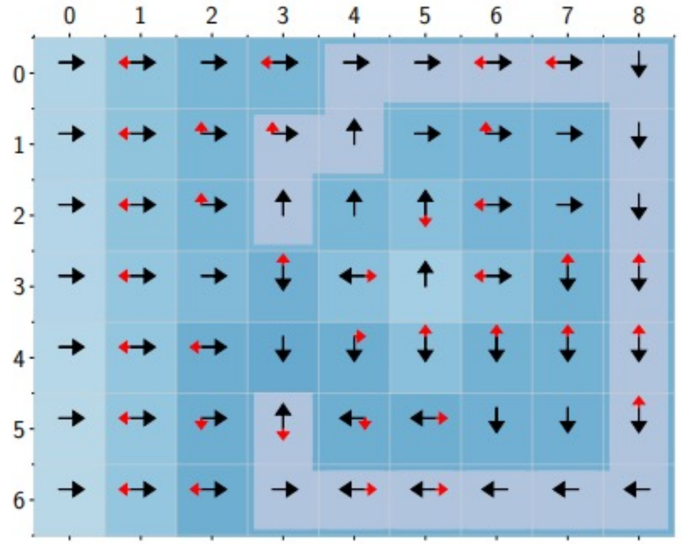
$$\varphi_{\mathrm{TASK}} = \Box\Diamond b \wedge \Box\Diamond c \wedge \Diamond\Box d$$



(a) The controller strategy from b to c and the labels of the cells



(b) The controller and the attacker strategies from b to c before any anomaly



(c) The controller and the attacker strategies from b to c after one anomaly

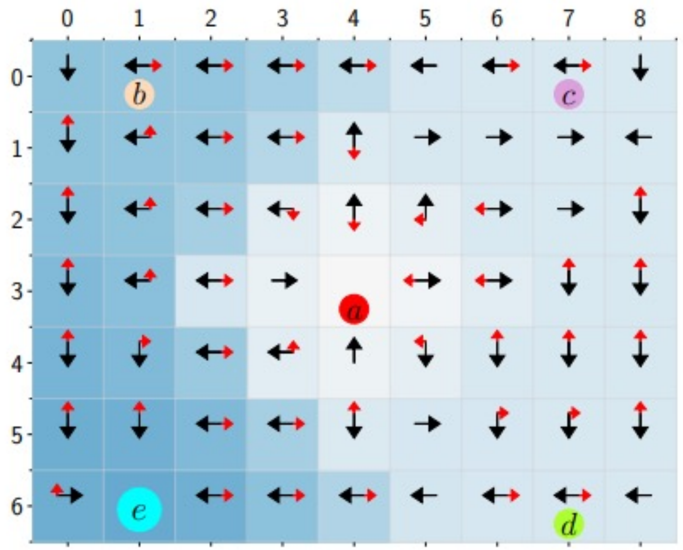
Case Study II: Sequencing

Duke
PRATT SCHOOL of ENGINEERING

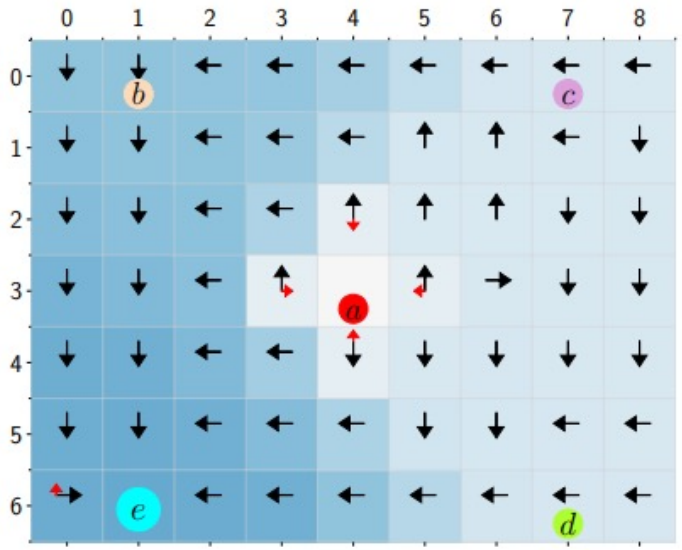
- **Task:**
  - Repeatedly visit a b and a c cell
  - Eventually reach a safe region labeled with d and do not leave

$$\varphi_{\mathrm{TASK}} = \Diamond\Big(b \wedge \Diamond\big(c \wedge \Diamond(d \wedge \Diamond e)\big)\Big) \wedge \Box\neg a$$

(a) The controller strategy from d to e and the labels of the cells



(b) The controller and the attacker strategies from d to e right after an anomaly happens

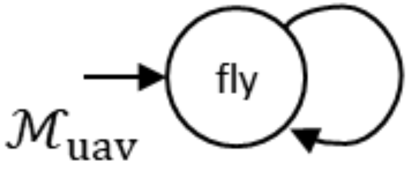


(c) The controller and the attacker strategies from d to e right after an alarm is raised

Attacks on Sensors

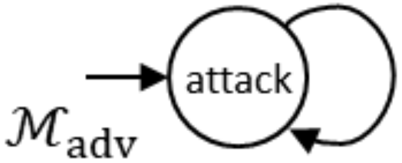
Duke
PRATT SCHOOL of ENGINEERING

UAV Model



$$\begin{array}{c|l} {\color{red}pl = uav} & x_{\mathcal{B}} \coloneqq x_{\mathcal{B}} + \Delta x(d) \\ fly! & d_{\mathcal{B}} \coloneqq d \\ d \in A_{\text{uav}} & pl \coloneqq adv \end{array}$$

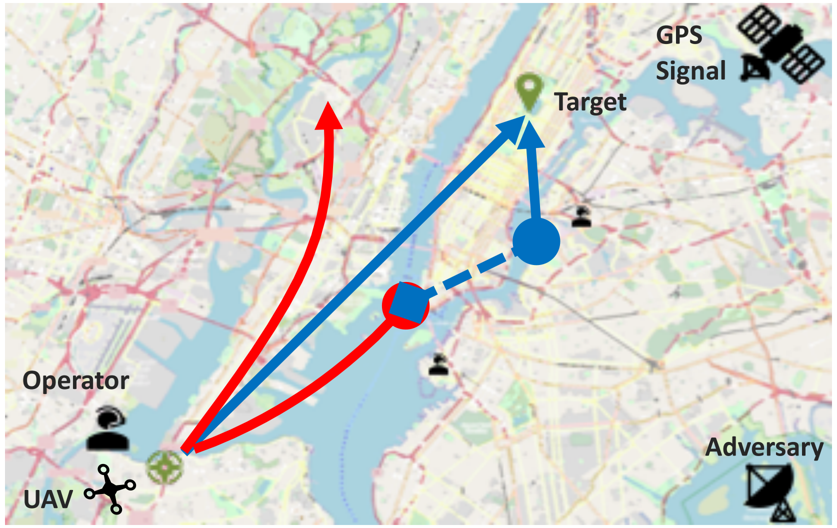
Adversary Model



$pl = adv$

$\boxed{x_{\mathcal{T}} \coloneqq x_{\mathcal{T}} + \Delta x(f(z))}$

$pl \coloneqq as$

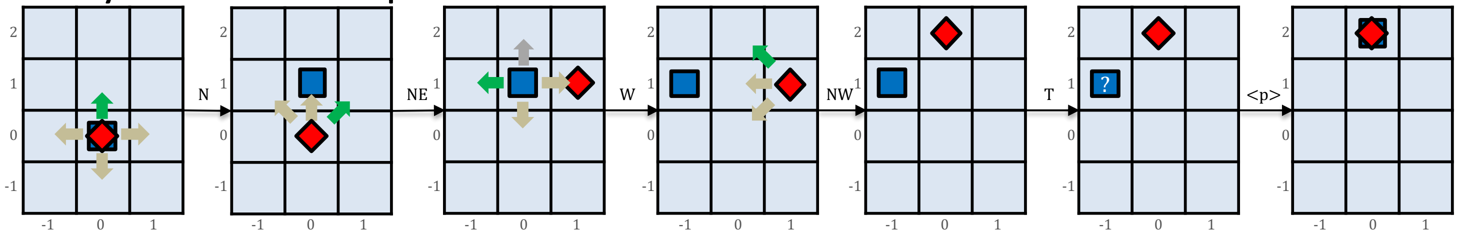


Information inside this box is oftentimes unknown, i.e., **hidden**

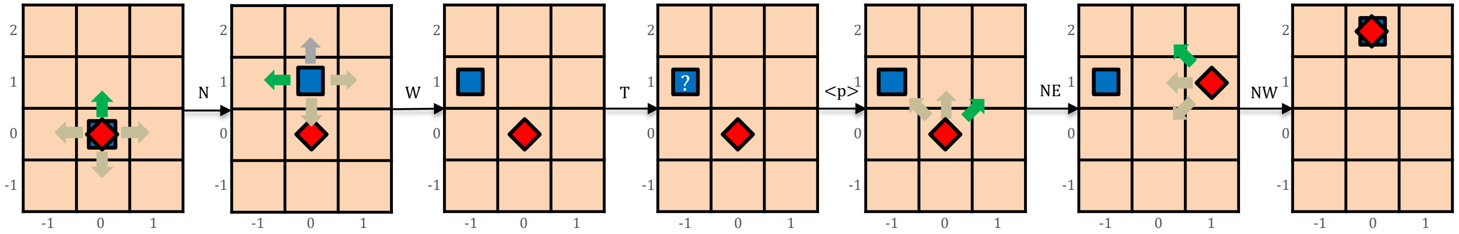
Off-the-shelf model checkers do NOT support hidden variables
Strategies CANNOT be synthesized based on hidden information

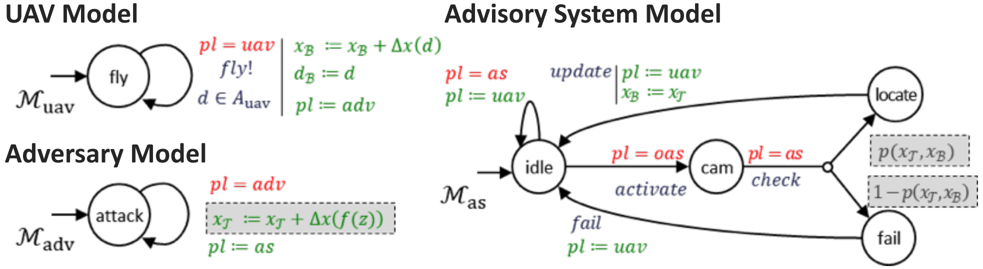
Security-Aware Mission Planning

Delayed Actions Representation

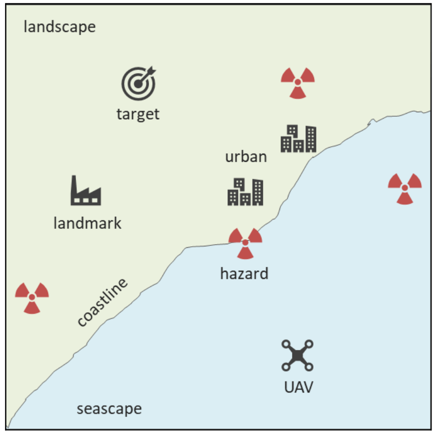


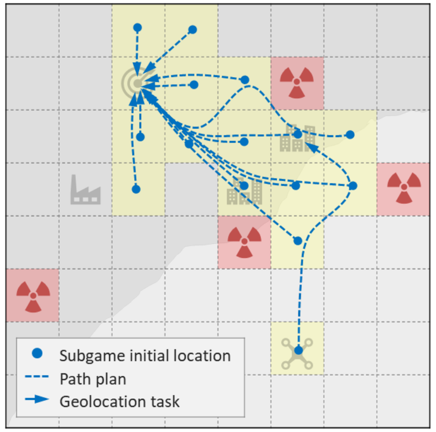
Bisimulation relation

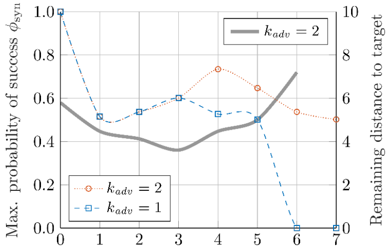




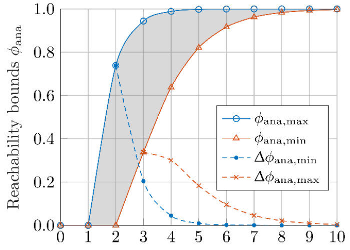
Private Variables Representation





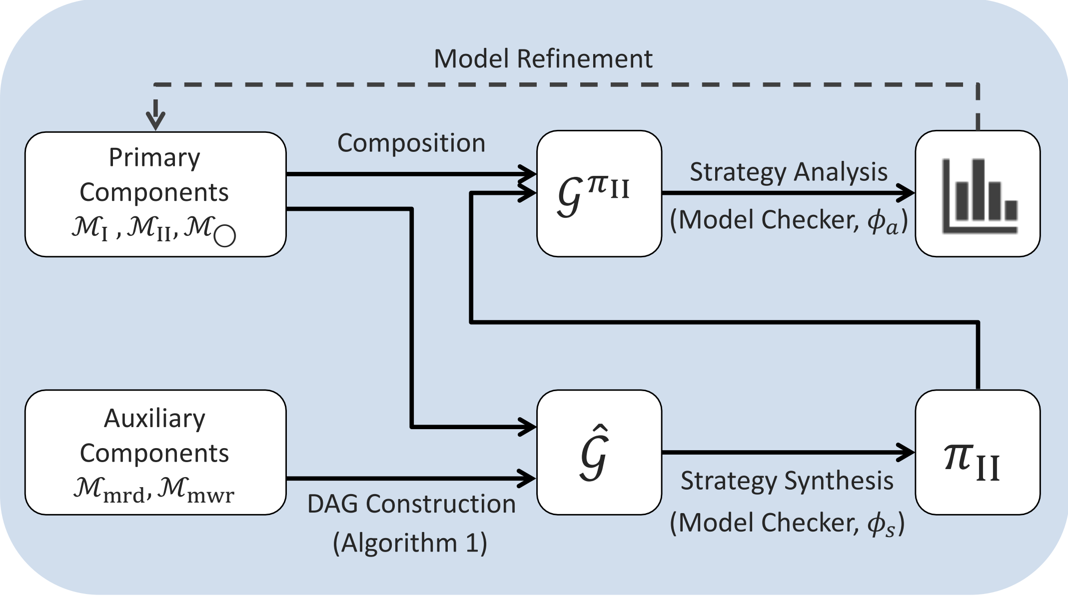


(a) Geolocation task at stage k



(b) Max. no. of geolocation tasks h

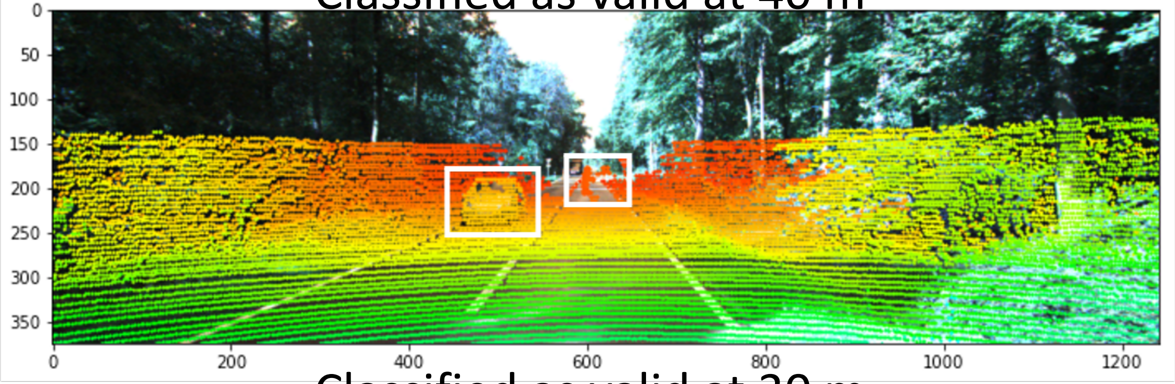
Synthesis Framework [CAV19]



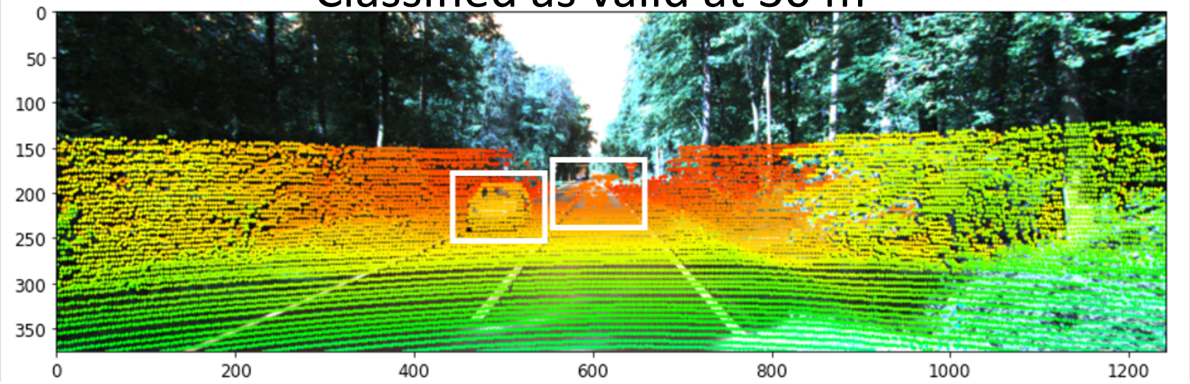
[1] M. Elfar, Y. Wang, and M. Pajic, “Security-Aware Synthesis using Delayed Action Games”, 31st CAV, 2019.

Lidar Attacks – Visualizations in Camera Frame

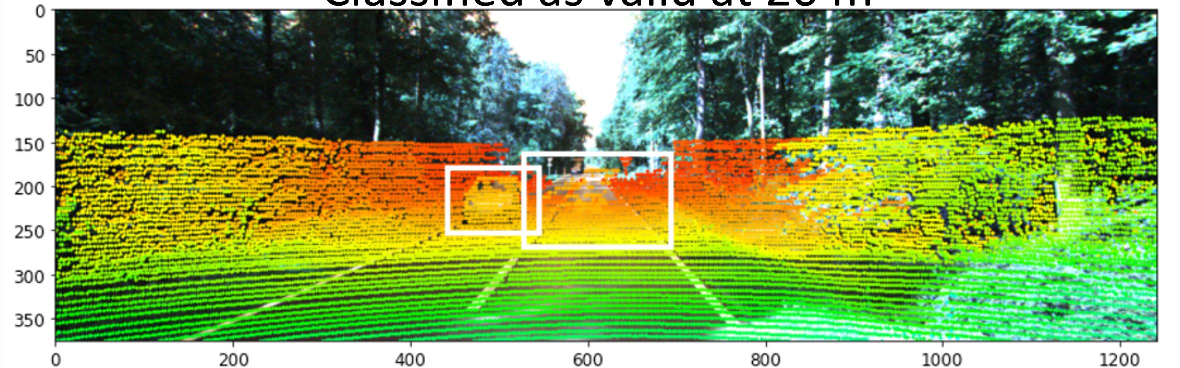
Classified as valid at 40 m



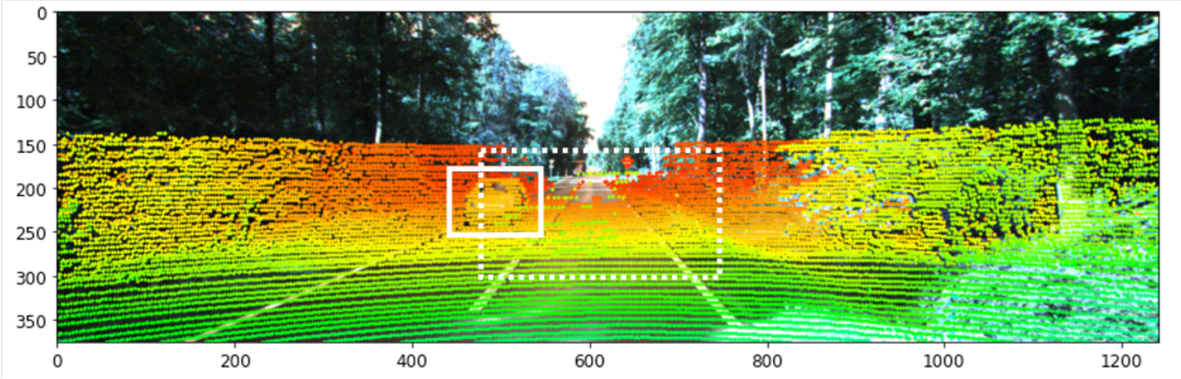
Classified as valid at 30 m



Classified as valid at 20 m

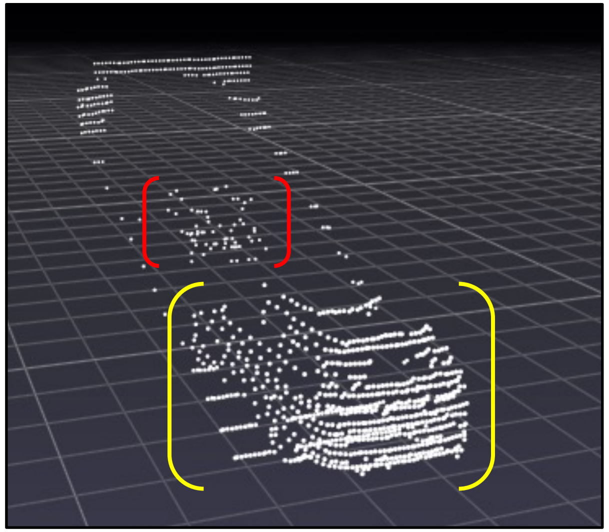


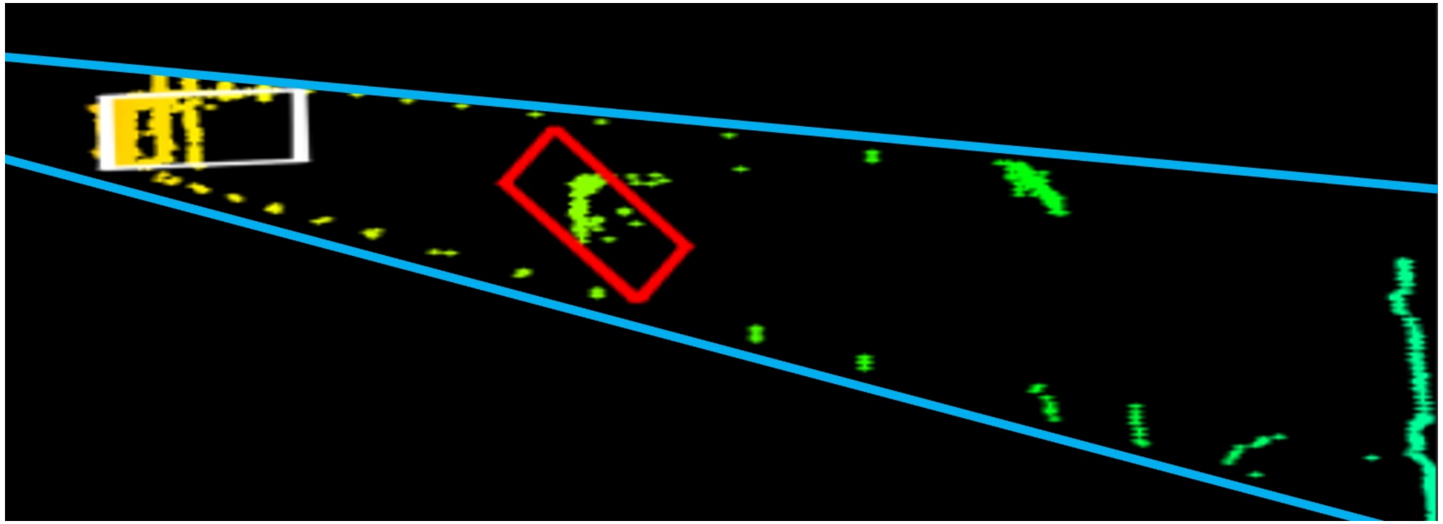
Classified as invalid at 10 m



Attacks on Camera-Lidar Fusion
Frustum Pointnet Vulnerability Example

Duke
PRATT SCHOOL of ENGINEERING



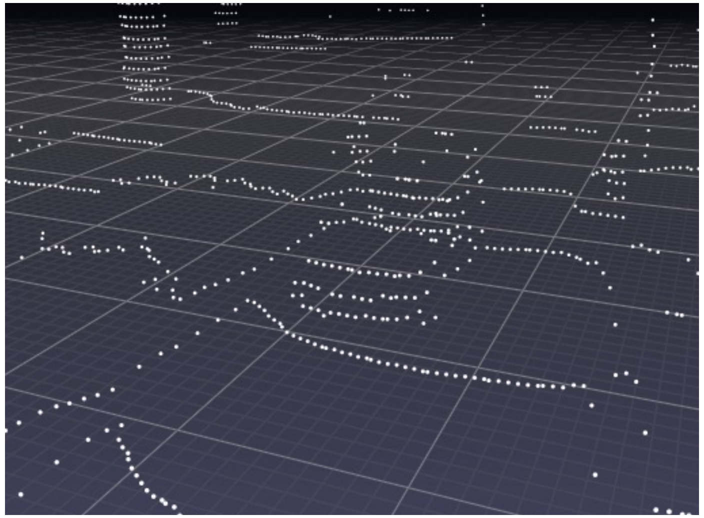


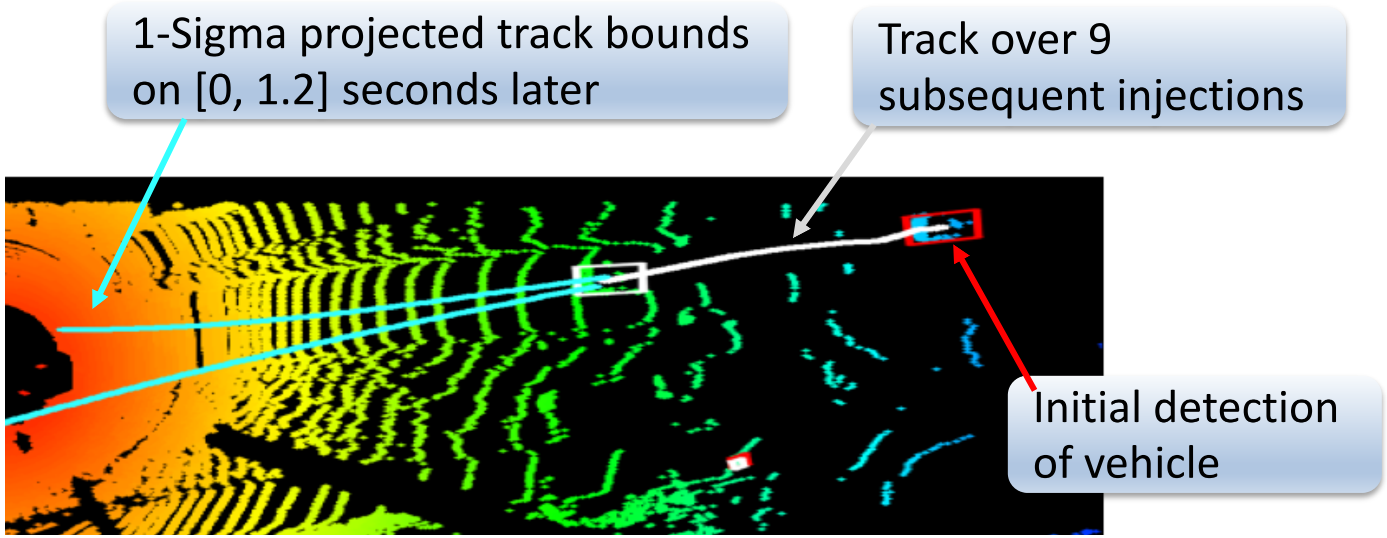
- Injection of just 65 points (bracketed red) can fool frustum pointnet 3D object detection, even against a valid object (bracketed yellow) of 492 points

- An adversary capitalizes on physics-based assumptions that few LiDAR points penetrate physical objects.

Fusion of camera + LiDAR is still vulnerable to attacks with knowledge of the approximate frustum

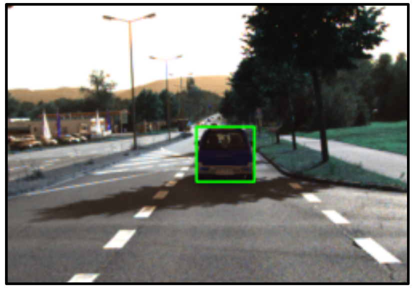
Tracking Case Study – Incoming Vehicle

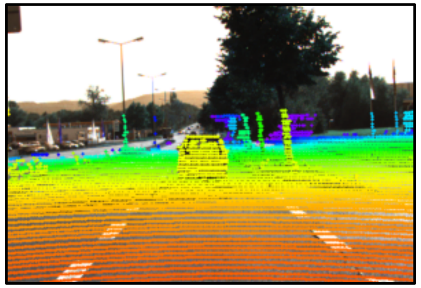


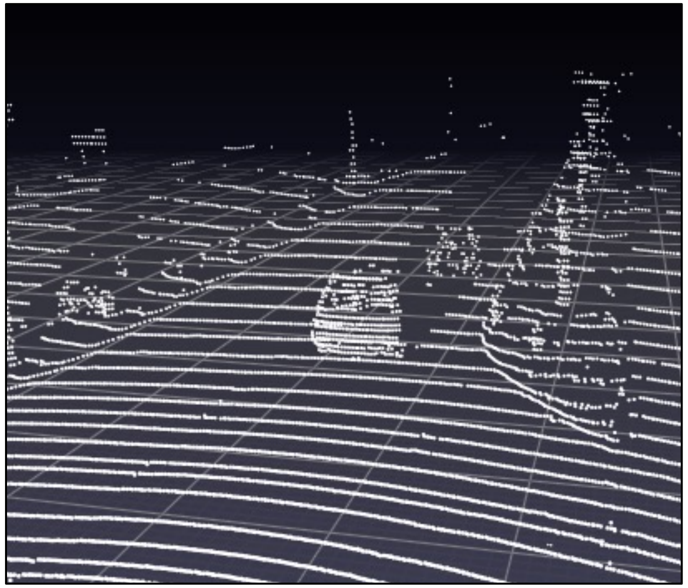


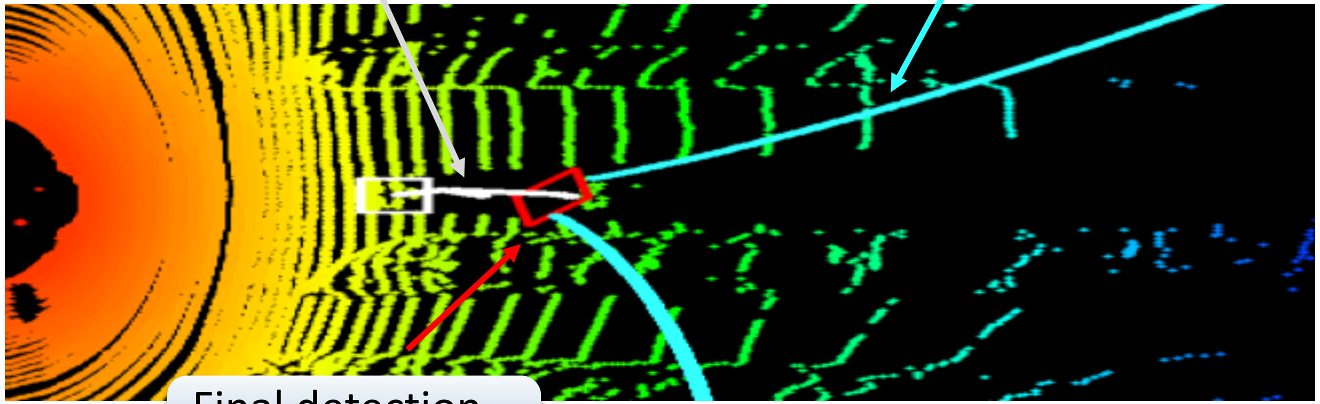
- Move false detections into false target tracking.
- Initial injection is in red box, white line is track history, and white box is ground truth target location.
- False moving target created with a time-to-impact with the host vehicle of just under 1.2 seconds

Tracking Case Study: Vehicle Following









Attack goal: create a false vehicle trajectory moving away from the host vehicle

- resulting in unsafe behavior of the host vehicle.

Thank you

Duke UNIVERSITY | PRATT SCHOOL *of* ENGINEERING