Know the Unknowns: Addressing Disturbances and Uncertainties in Connected and Autonomous Vehicles

Qi Zhu, Associate Professor
Northwestern University

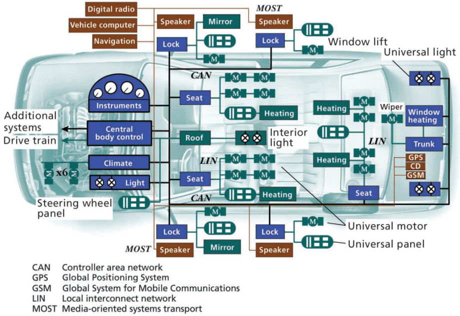
IEEE VTS Chicago Webinar

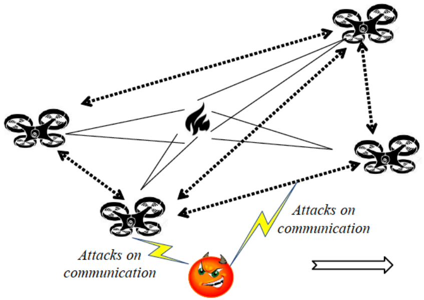
August, 2020

Northwestern

IDEAS Lab (DEsign Automation of Intelligent Systems)

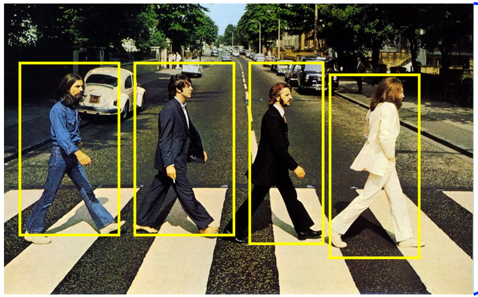
Goal: Create automated, rigorous, and systematic methods, tools, and algorithms for the design, validation, update, and adaptation of intelligent systems.





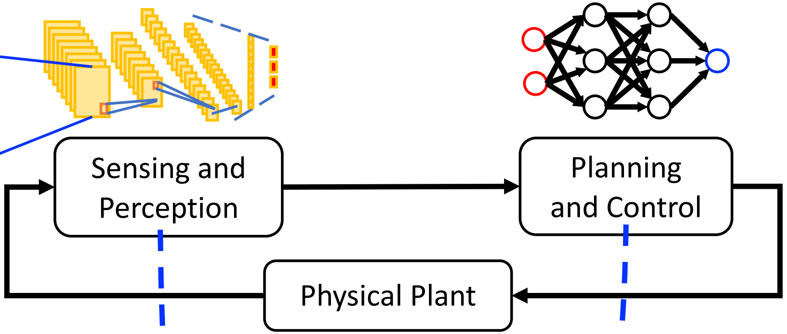
Connected and Autonomous Vehicles (CAVs) and Challenges

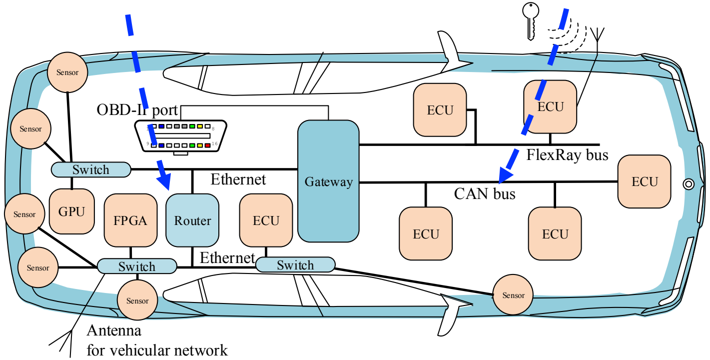
Physical Environment



- Dynamic environment
- Functional complexity
  - Scale and features
  - Machine learning
- Architectural complexity
  - Multicore CPUs, GPUs, …
  - Federated -> Integrated
- Stringent requirements
  - Safety critical and time critical

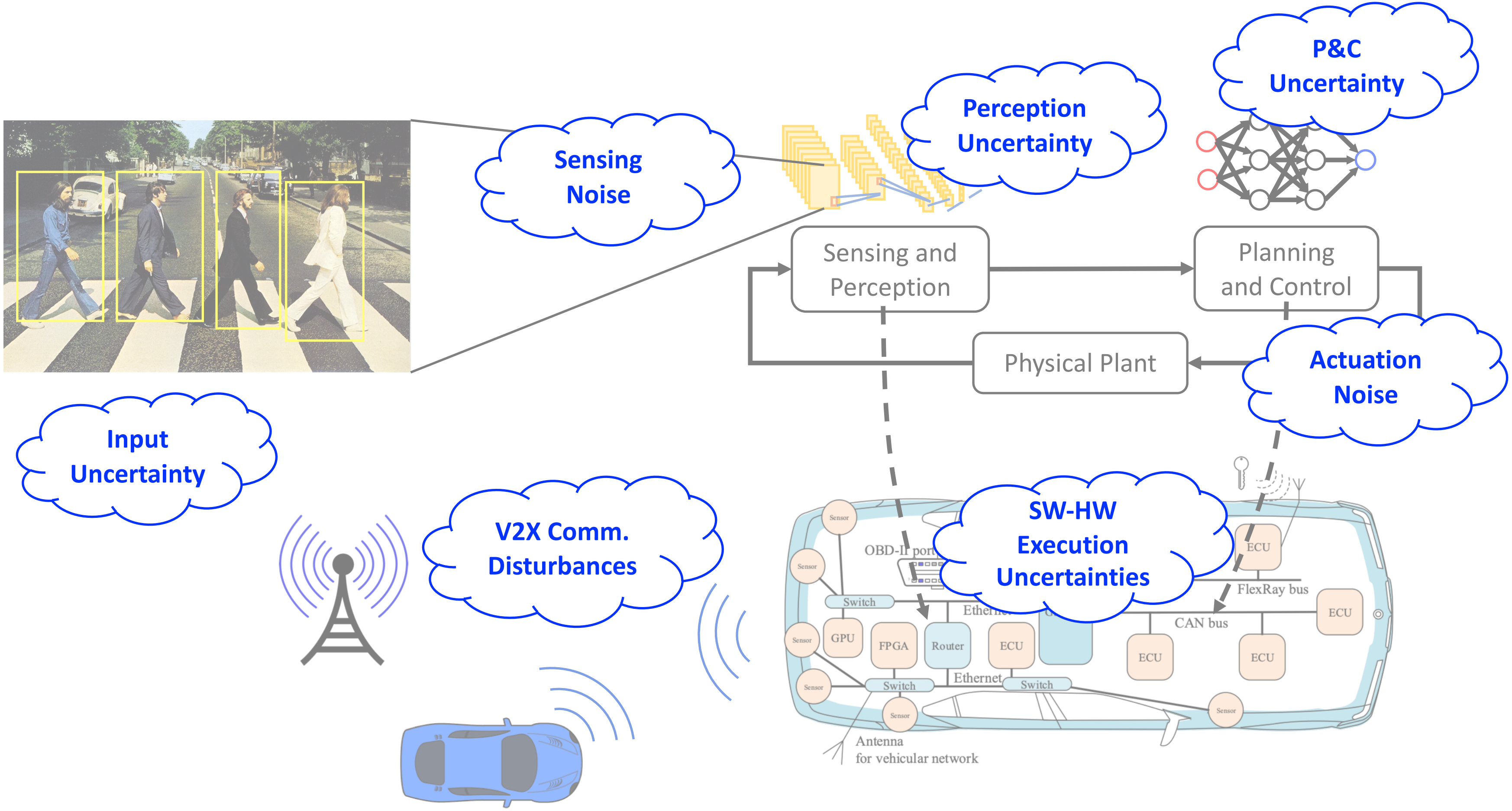
Algorithms (Functionality)



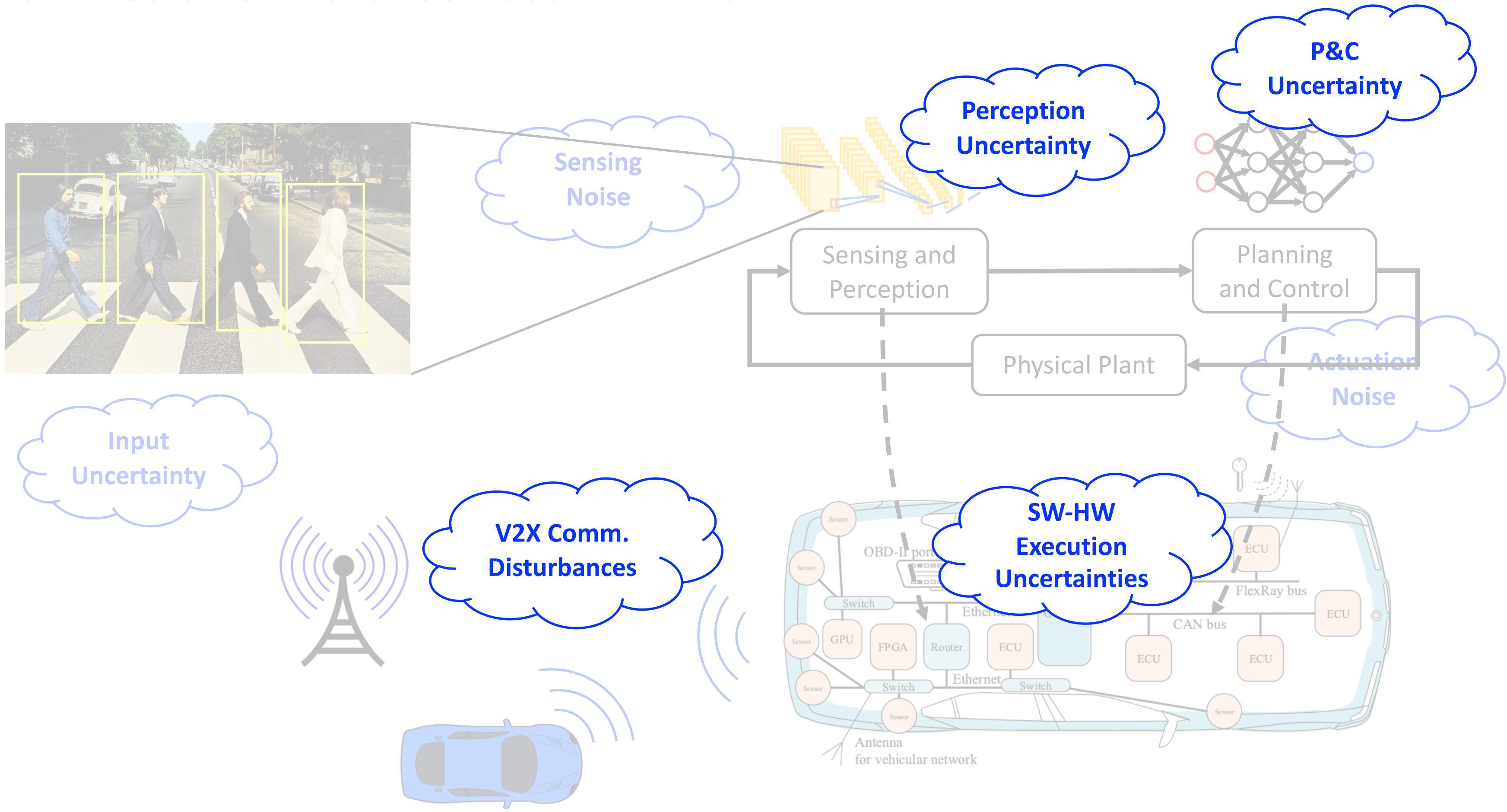


Cyber Platform (SW-HW Architecture)

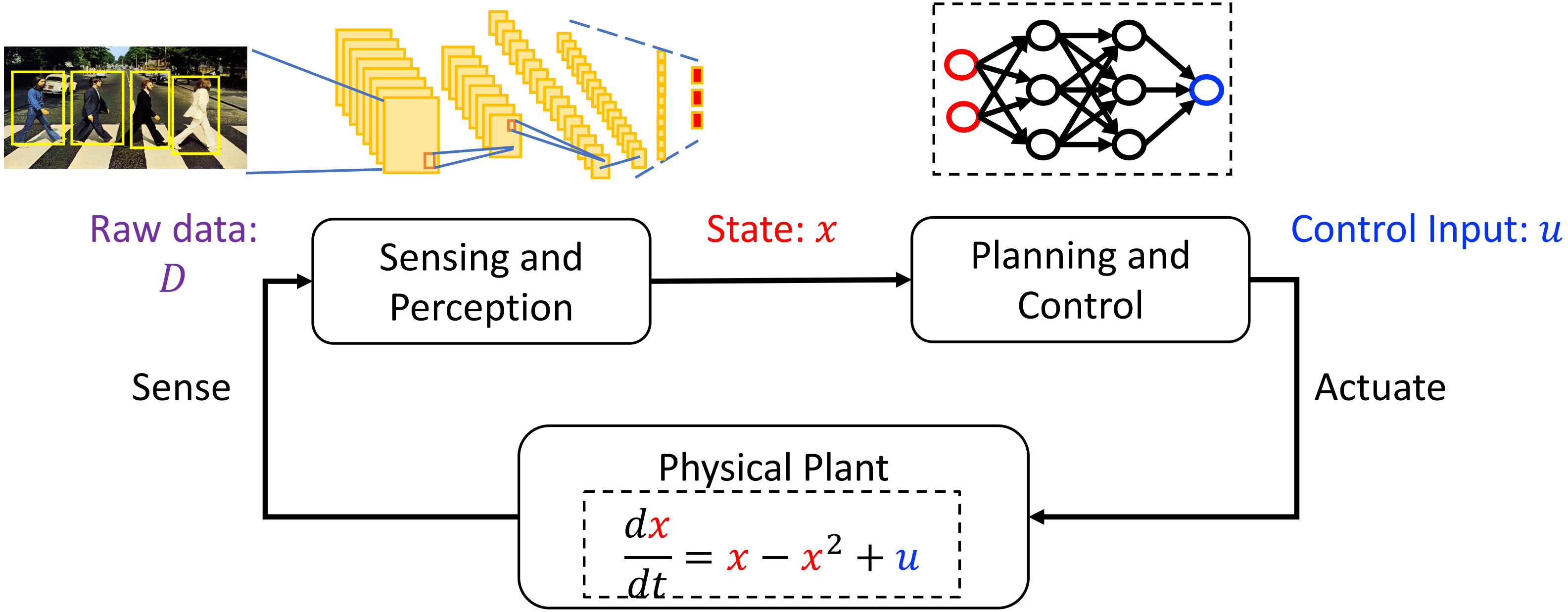
Uncertainties and Disturbances in CAVs



Uncertainties and Disturbances in CAVs

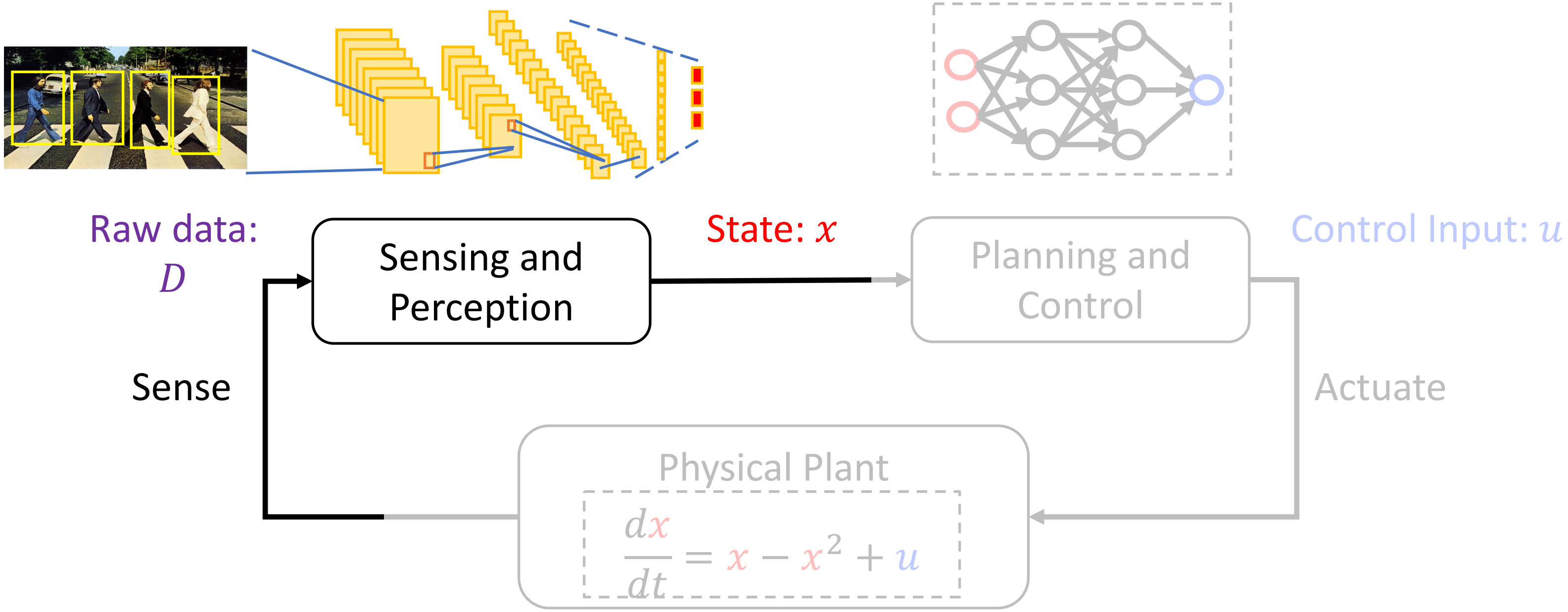


Addressing Uncertainty in Neural Networks



- S&P: output range analysis -> guarantees against adversarial examples
- P&C: reachability analysis -> safety verification of neural-network controlled systems

Addressing Uncertainty in Perception Neural Networks



- S&P: output range analysis -> guarantees against adversarial examples
- P&C: reachability analysis -> safety verification of neural-network controlled systems

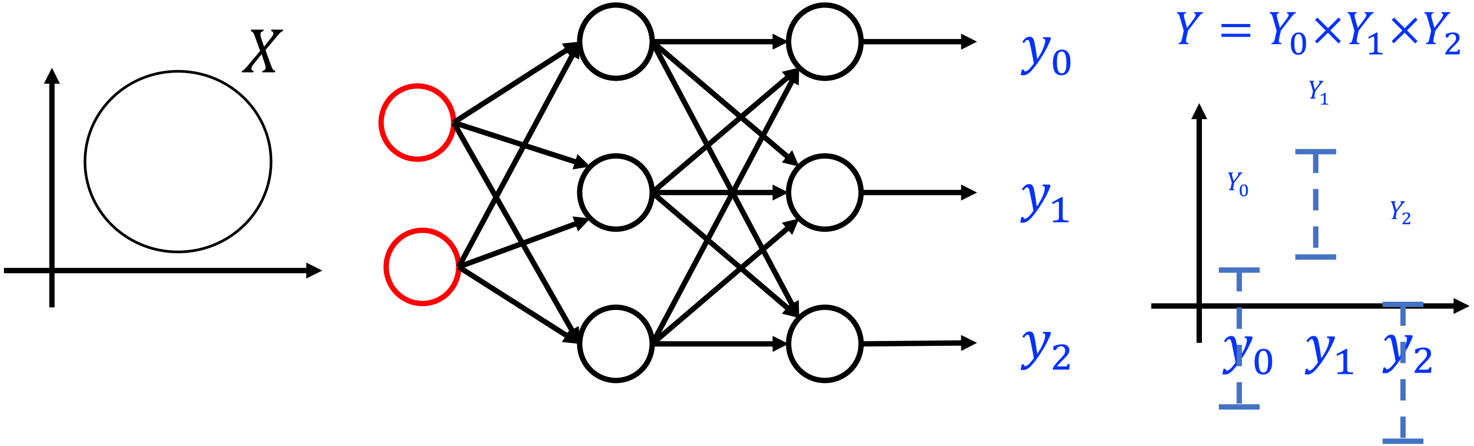
Adversarial Attacks to Perception Neural Networks

Adversarial examples: An adversarial example is an instance with small, intentional feature perturbations that cause a machine learning model to make a false prediction.

Adversarial examples for AlexNet [Szegedy et. al, 2013]. All images to the left are correctly classified. The middle column shows the (magnified) errors added to the images. The produced images to the right all categorized (incorrectly) as ‘Ostrich’.

[C. Szegedy, et al. "Intriguing properties of neural networks”. arXiv preprint arXiv:1312.6199, 2013.]

Output Range Analysis of Neural Networks



Definition: Given a neural network f, and a compact input set X, compute $Y = f(X)$ or its over-approximation (a tight superset that contains the compute $f(X)$) [Dutta et. al, 2017].

[S. Dutta S, et al. "Output range analysis for deep neural networks". arXiv preprint arXiv:1709.09130, 2017.]

Our Approach: *Divide*

For nonlinear operations in a neural network, i.e., activation functions,

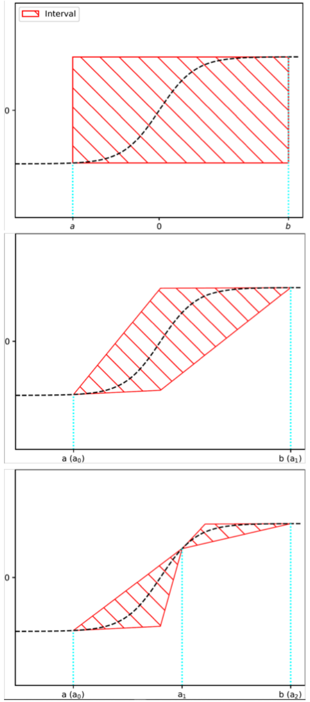
Interval relaxation

↓

LP relaxation

↓

MILP relaxation



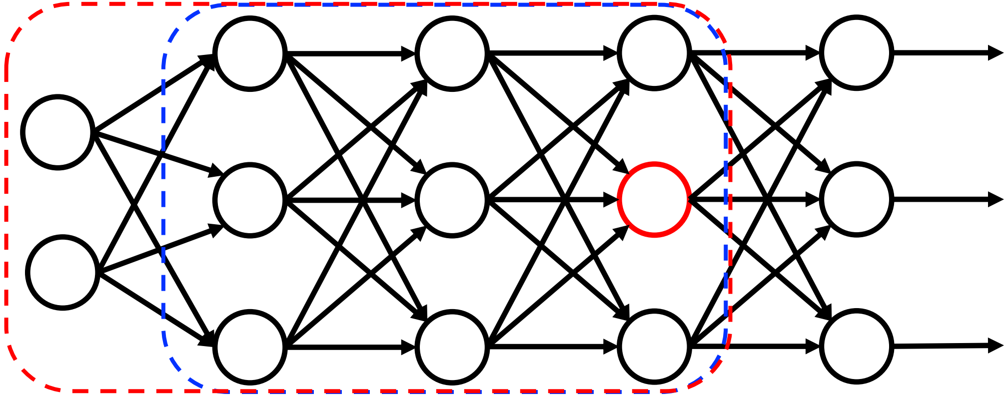
Obtain the interval relaxation for the input of each neuron by IBP and efficient SIP (e.g., ERAN).

Based on the left/right derivative, obtain the polytope (LP) relaxation for a given input interval obtained by interval relaxation.

Partition the input interval, and obtain the multi-polytope (MILP) relaxation based on the LP relaxation formulation.

An example of tanh activation function

Our Approach: *Slide*



$$\bar{x} = \min x_3[2]$$

Subject to

$$\begin{cases} f_1(\color{red}{x}, x_1, \omega) \leq 0 \wedge \color{red}{x} \in X \\ f_2(x_1, x_2, \omega_1) \leq 0 \wedge x_1 \in X_1 \\ \dots \\ f_n(x_{n-1}, \color{blue}{y_0,} \omega_{n-1}) \leq 0 \wedge x_{n-1} \in X_{n-1} \end{cases}$$

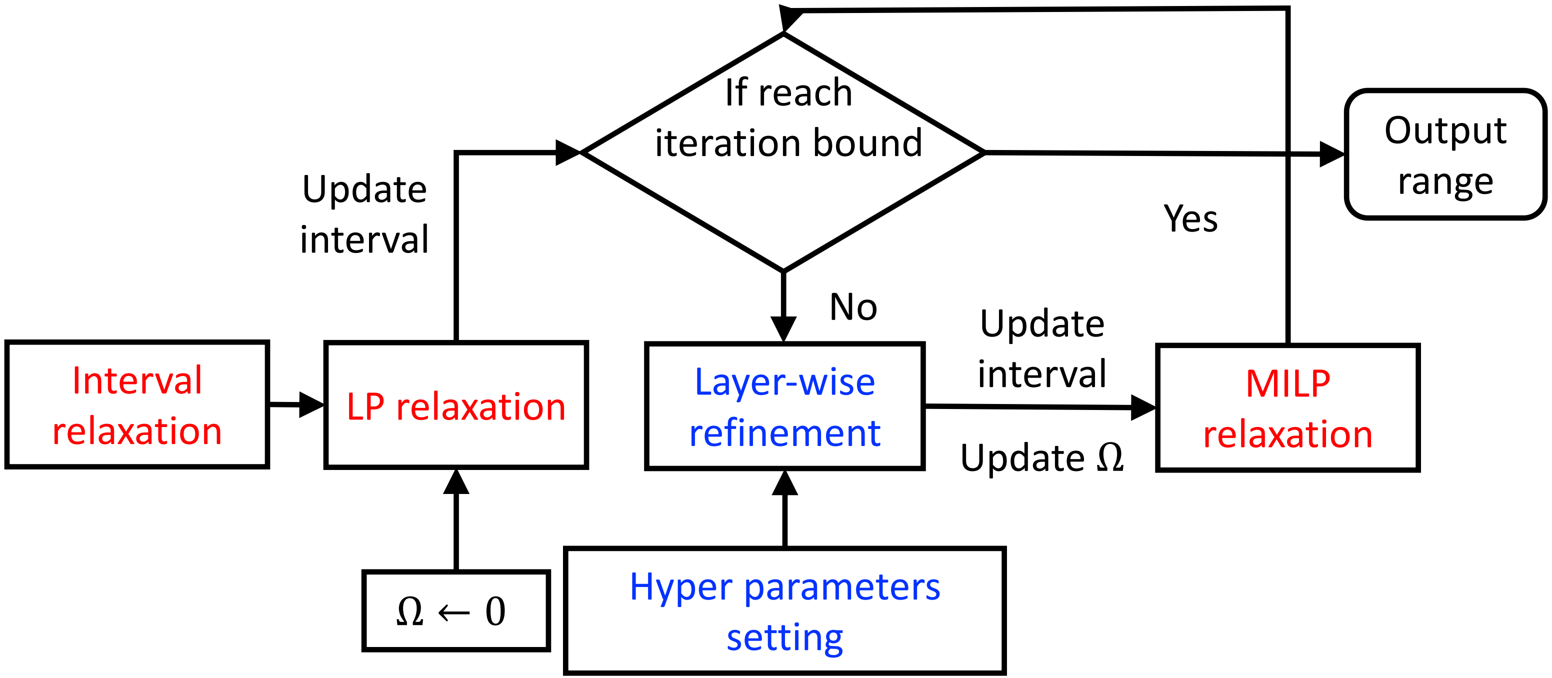
$$\bar{x}' = \min x_3[2]$$

Subject to

$$\begin{cases} f_2(x_1, x_2, \omega_1) \leq 0 \wedge x_1 \in X_1 \\ \dots \\ f_n(x_{n-1}, \color{blue}{y_0,} \omega_{n-1}) \leq 0 \wedge x_{n-1} \in X_{n-1} \end{cases}$$

Validity: $\bar{x}' \leq \bar{x}$.

Our *LayR* Tool for Output Range Analysis: Divide and Slide



- Divide: For each neuron, divide the input space to refine the over-approximation.
- Slide: Perform layer-wise refinement with a sliding-window based method.

[C. Huang, et al. "Divide and Slide: Layer-Wise Refinement for Output Range Analysis of Deep Neural Networks". EMSOFT, 2020.]

Comparison with NNV

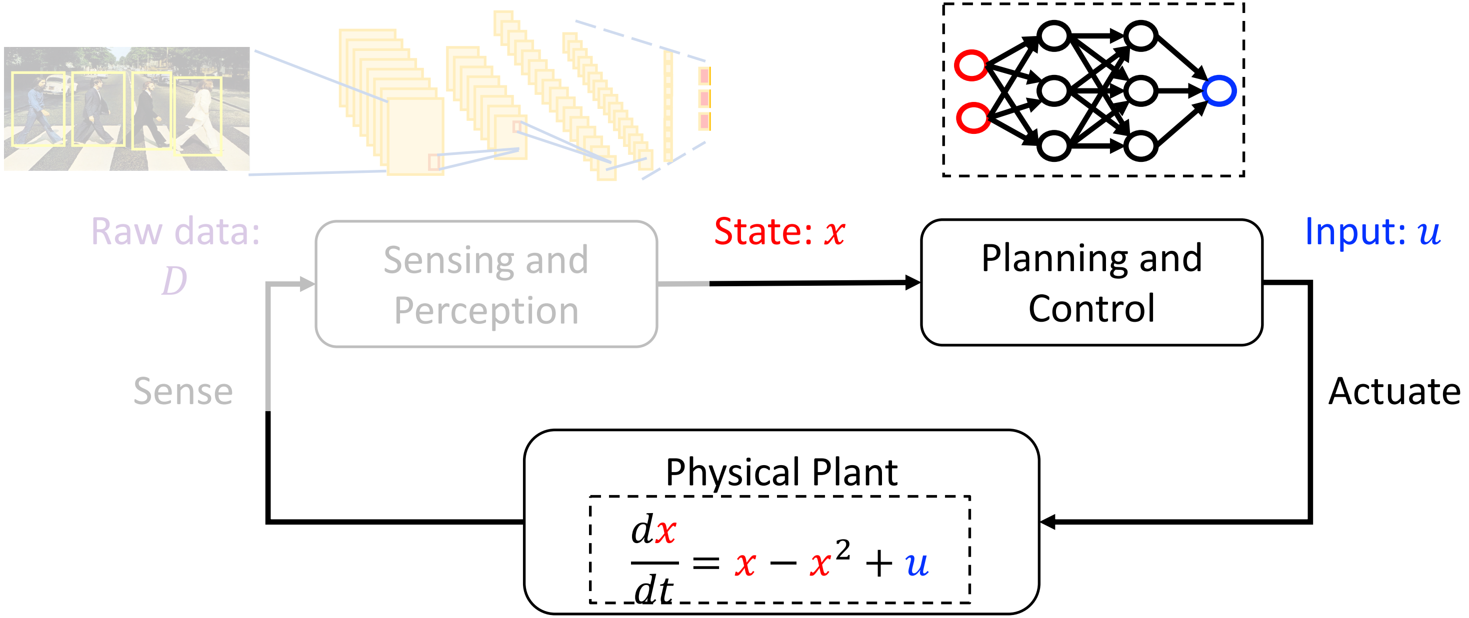
Compared with NNV *[Hoang-Dung, et. al, 2020]*

#[1]	Input set	NNV		*LayR*	
		Range	Time (s)	Range	Time (s)
I	MNIST-1	12.44	6	2.85	1068
	MNIST-2	12.78	7	1.52	925
	MNIST-3	30.36	7	22.10	976
	MNIST-4	12.64	7	2.41	1057
II	MNIST-1	10.50	11	2.24	1200
	MNIST-2	12.43	11	4.96	1656
	MNIST-3	28.44	12	25.44	1274
	MNIST-4	14.13	11	0.75	2663
III	MNIST-1	7.74	3456	2.29	3078
	MNIST-2	6.72	1782	3.07	3124
	MNIST-3	6.26	5954	2.05	3113
	MNIST-4	4.61	1404	2.38	3113

[1] On the remaining settings including MNIST IV-V and CIFAR VI-VII, NNV exceeded a timeout limit of 24 hours while the longest running time of our tool among these benchmarks was around 5 hours on the same machine. Thus we do not have the range comparison for those cases here.

- *LayR* shows 10.55% (Network II, NNIST-3) to 94.69% (Network II, NNIST-4) improvement on output range estimation over NNV.
- LayR has a much slower runtime increase wrt. the increase of neural network size, when compared with NNV.

Addressing Uncertainty in Neural Network Controlled Systems

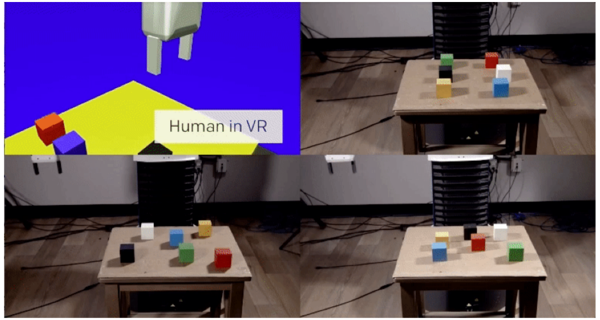


- S&P: output range analysis -> guarantees against adversarial examples
- P&C: reachability analysis -> safety verification of neural-network controlled systems

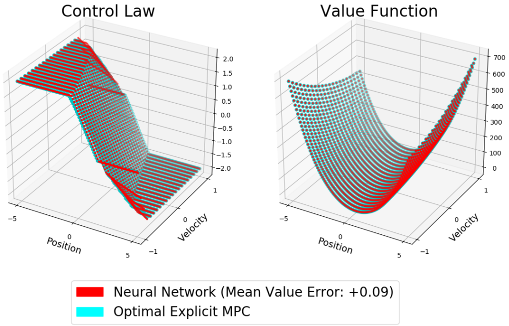
Neural Network Controlled Systems (NNCS)

- (Deep) reinforcement learning
- (End-to-end) Imitation learning
- Approximating MPC

"Best" AI agent trained using Reinforcement Learning (20% higher score than humans)



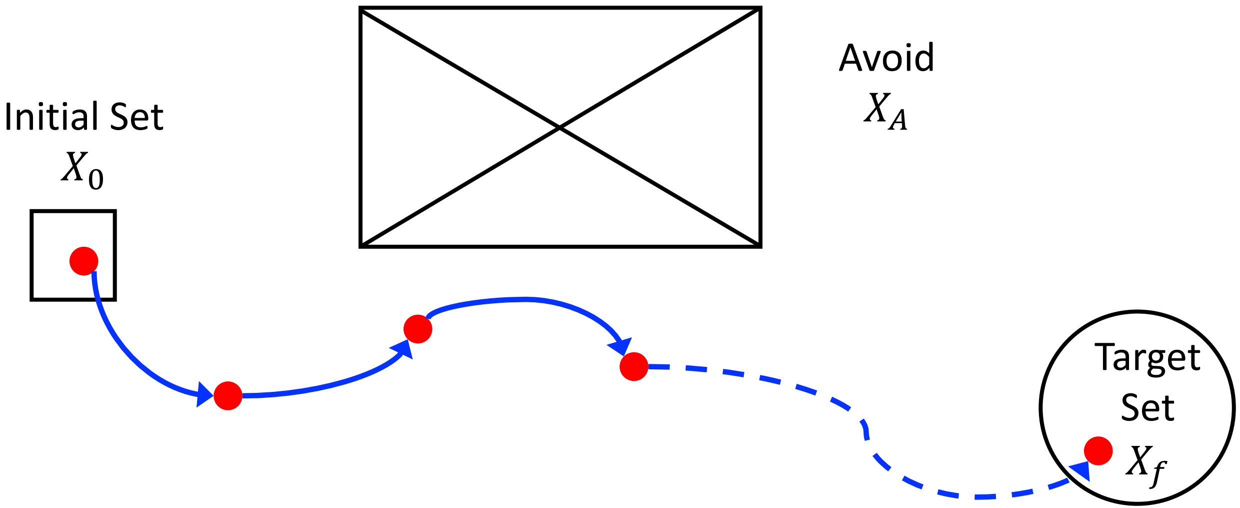
[Duan, et al. 2017]



[Chen, et al. 2018]

[Codevilla, et al. 2017]

Reachability Analysis of NNCS



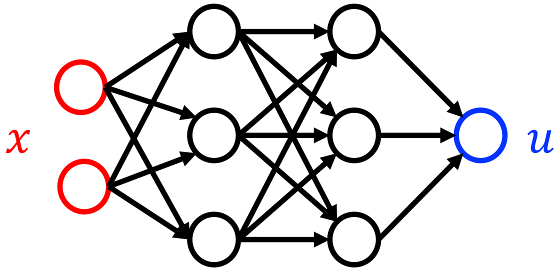
Reach-Avoid: Starting from *any* state in X_0, decide if the NNCS will reach a state in X_f at time $t \geq 0$ while avoiding X_A.

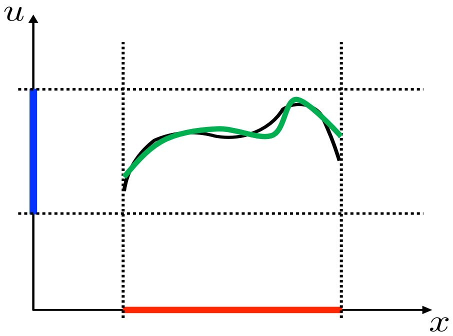
Our *ReachNN* Tool for Reachability Analysis of NNCS

- Key idea: use *Bernstein Polynomials* to approximate the NN controller.

$$B_{f,d}(x) = \sum_{\substack{0 \le k_j \le d_j \\ j \in \{1,\cdots,m\}}} f\left(\frac{k_1}{d_1}, \cdots, \frac{k_m}{d_m}\right) \prod_{j=1}^{m} \left(\binom{d_j}{k_j} x_j^{k_j} (1 - x_j)^{d_j - k_j} \right)$$

- Advantage: works for any Lipschitz continuous NN (ReLU, tanh, sigmoid, combination of them, etc.)



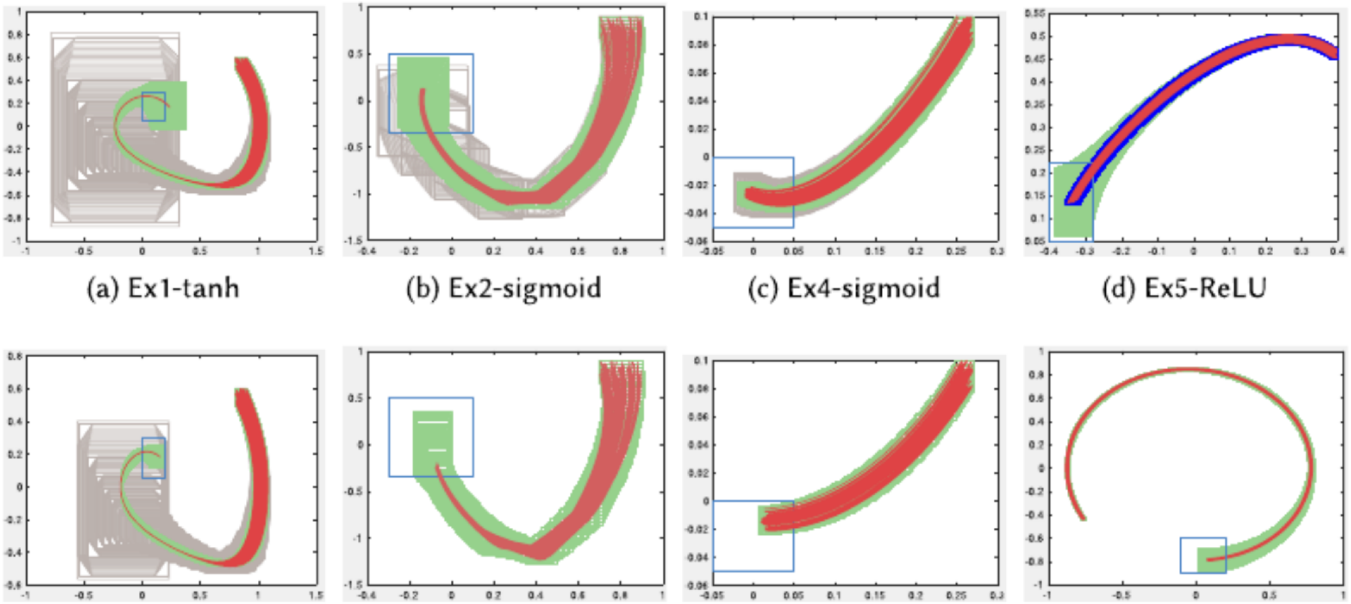


Bernstein Polynomial Approximation

$f(x) \in \{u \mid u = B_{f,d}(x) + \epsilon\}, \forall x \in Xi$

[C. Huang, et al. "ReachNN: Reachability analysis of neural-network controlled systems". EMSOFT, 2019.]

Comparison with Others

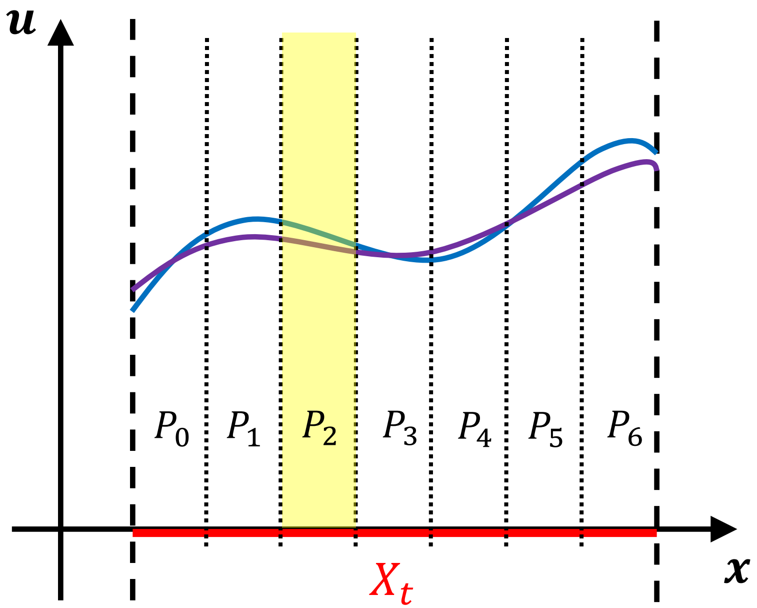


(a) Ex1-tanh (b) Ex2-sigmoid (c) Ex4-sigmoid (d) Ex5-ReLU

(e) Ex1-sigmoid (f) Ex2-ReLU-tanh (g) Ex4-ReLU-tanh (h) Ex6-ReLU-tanh

Flowpipes for the selected examples: Red curves denote the trajectories of x_1 and x_2 of the system simulated from sampled states within the initial set. Green rectangles: ReachNN [Huang, et. al, 2019], gray rectangles: Verisig [Ivanov, et. al, 2019], navy rectangles: Sherlock [Dutta, et. al, 2019].

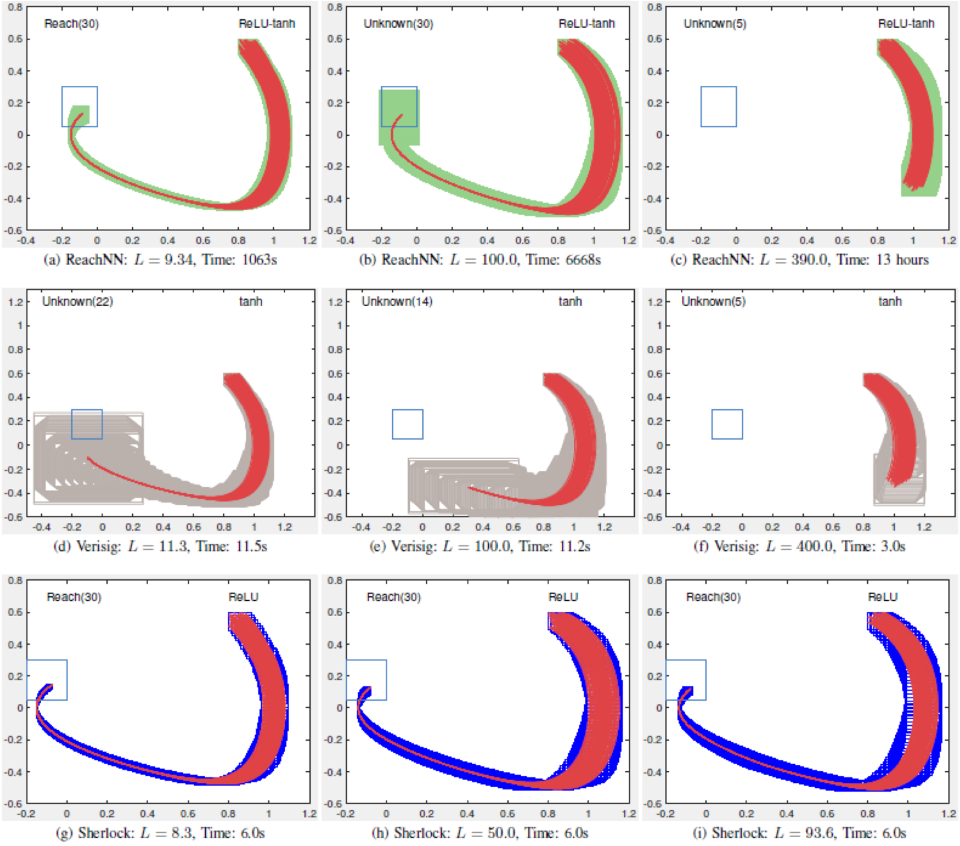
ReachNN:* Parallel Computing for Error Estimation



- Approximation error estimation is a key step in *ReachNN* and time-consuming. *ReachNN** improves it with a partitioned approach and parallel execution on GPUs.

[J. Fan, et al. "ReachNN: A Tool for Reachability Analysis of Neural-Network Controlled Systems". ATVA, 2020.]*

Make a Neural Network more Verification Friendly

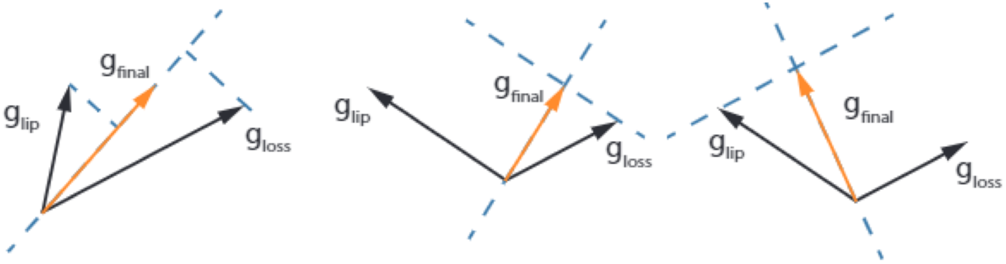


(a) ReachNN: $L = 9.34$, Time: 1063s
(b) ReachNN: $L = 100.0$, Time: 6668s
(c) ReachNN: $L = 390.0$, Time: 13 hours
(d) Verisig: $L = 11.3$, Time: 11.5s
(e) Verisig: $L = 100.0$, Time: 11.2s
(f) Verisig: $L = 400.0$, Time: 3.0s
(g) Sherlock: $L = 8.3$, Time: 6.0s
(h) Sherlock: $L = 50.0$, Time: 6.0s
(i) Sherlock: $L = 93.6$, Time: 6.0s

- Evaluate the impact of Lipschitz constant on three NNCS verification tools: ReachNN, Verisig, and Sherlock.
- Large Lipschitz constant may make verification harder: e.g., uncontrollable approximation error (Fig. f), excessively long computation time (Fig. c).
- Retrain neural networks to reduce Lipschitz constants while maintaining control performance.

Knowledge Distillation: Dual-Objective Optimization

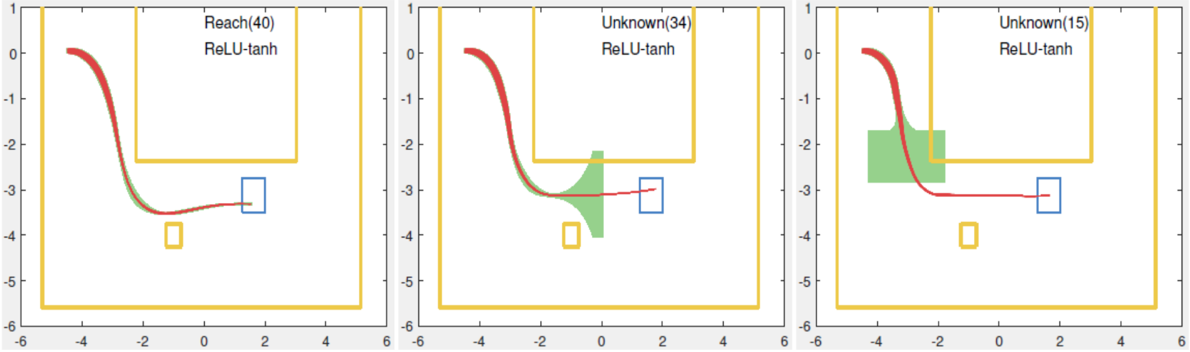
- Regression error J_{loss}: Error between the original network and the retrained network.
- Lipschitz constant error J_{lip}: Difference between the current Lipschitz constant and a target value.



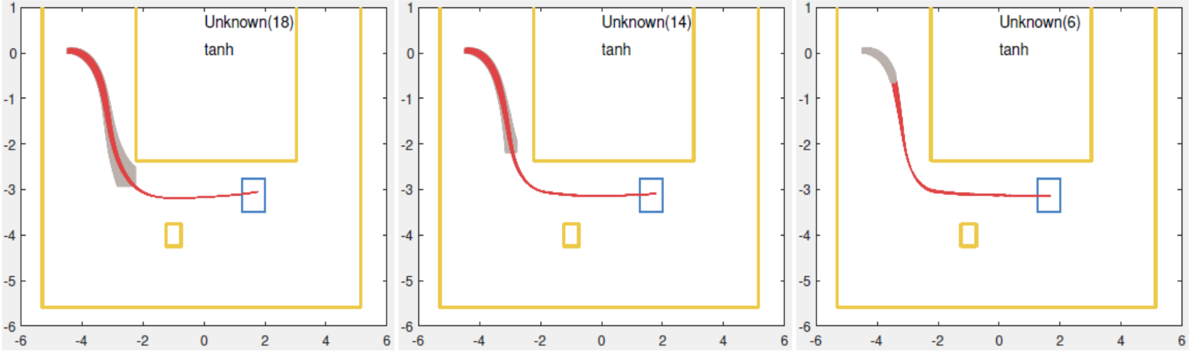
(a) $g_{loss} \cdot g_{lip} > 0$ (b) $g_{loss} \cdot g_{lip} < 0$ (c) $g_{loss} \cdot g_{lip} < 0$ (reprioritization)

[J. Fan, et al. "Towards Verification-Aware Knowledge Distillation for Neural-Network Controlled Systems". ICCAD, 2019.]

Effect of Knowledge Distillation for Smaller Lipschitz Constant



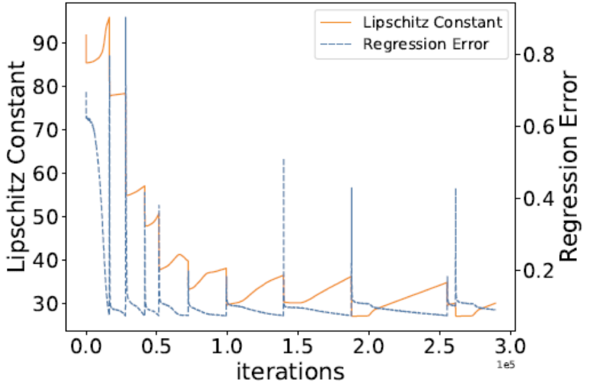
(a) ReachNN: $L = 13.1$, Time: 4657s (b) ReachNN: $L = 100.0$, Time: 13765s (c) ReachNN: $L = 244.3$, Time: 13 hours



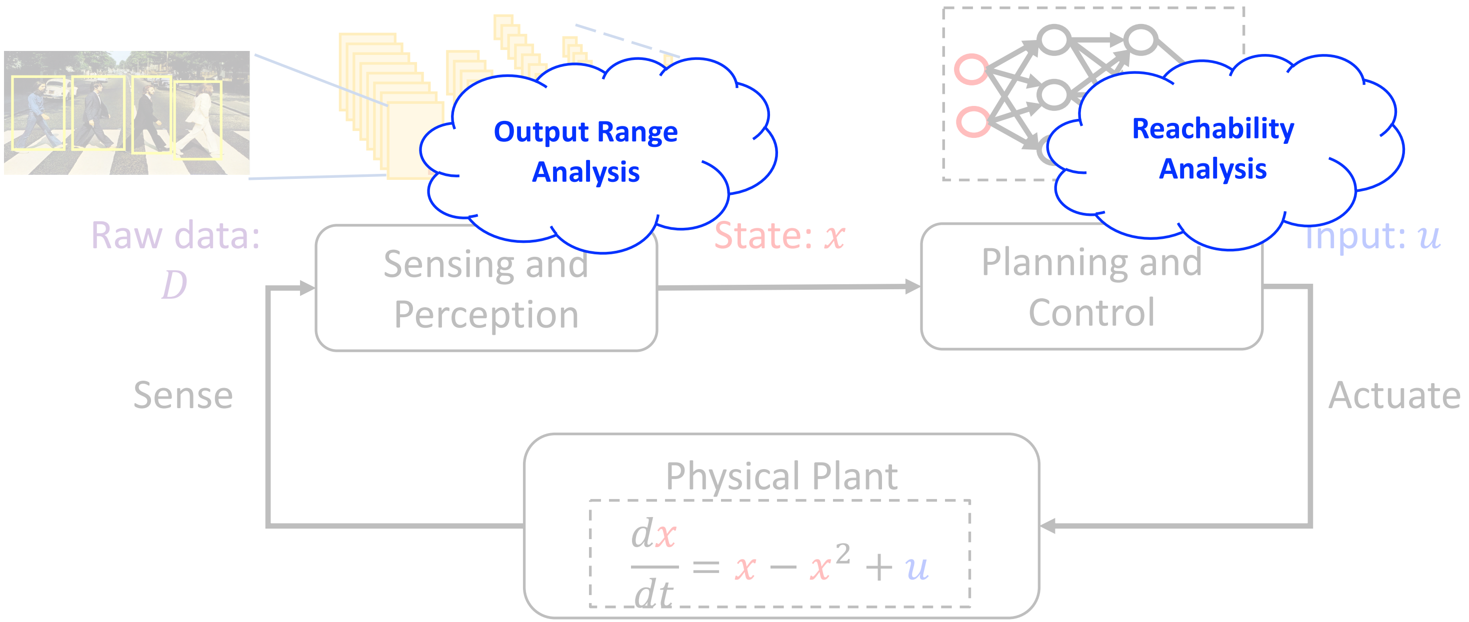
(d) Verisig: $L = 17.1$, Time: 12s (e) Verisig: $L = 150.4$, Time: 12s (f) Verisig: $L = 348.6$, Time: 15s

← Smaller Lipschitz constant: more steps the tools can verify!

↓ The fluctuations reflects the effect of our dual-objective gradient descent approach (eventually it converges).



Recap: Uncertainty in Neural Networks

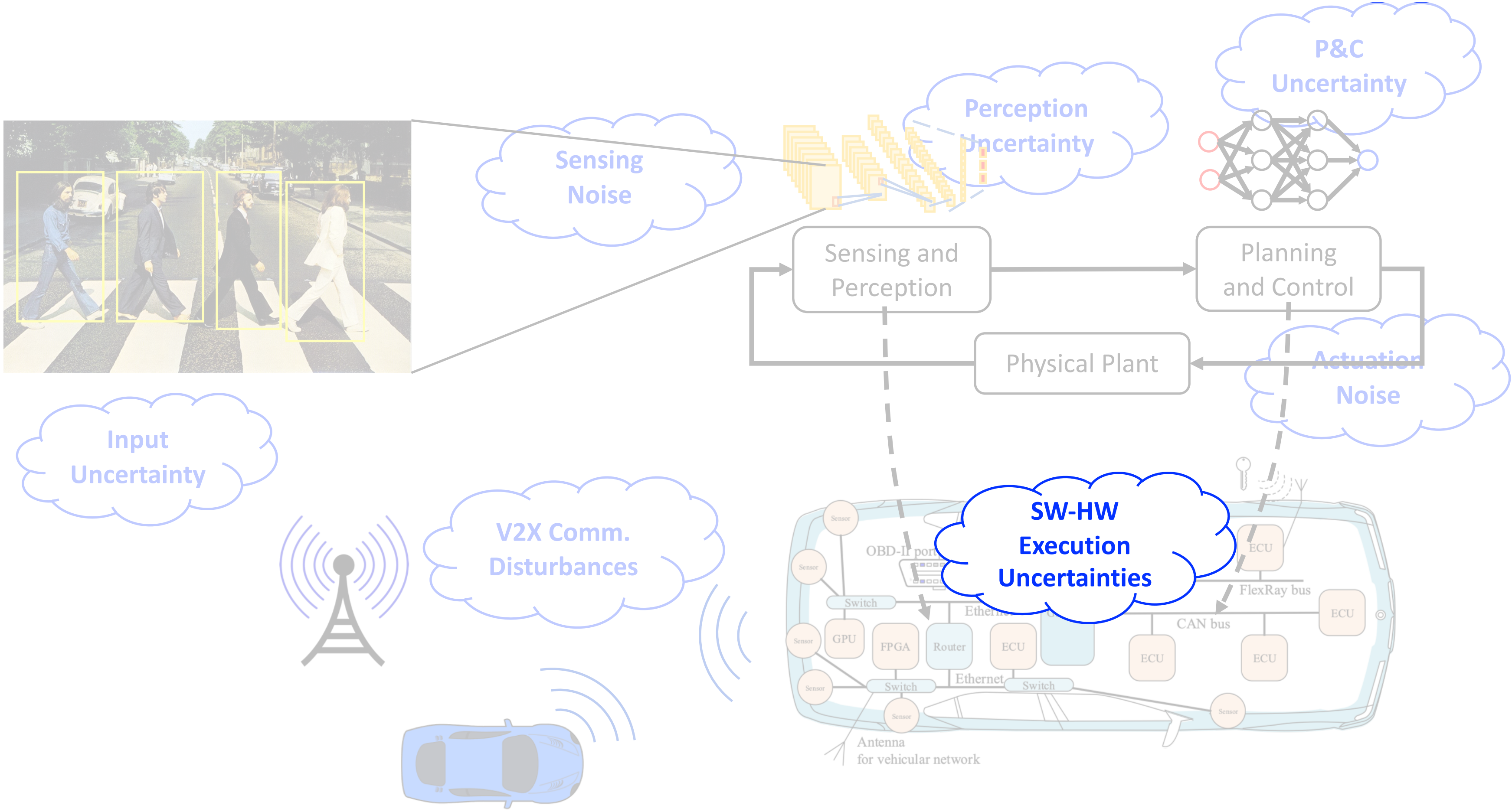


- *LayR:* Output range analysis (guarantees against adversarial examples).
- *ReachNN*:* Reachability analysis (safety verification of neural-network controlled systems).

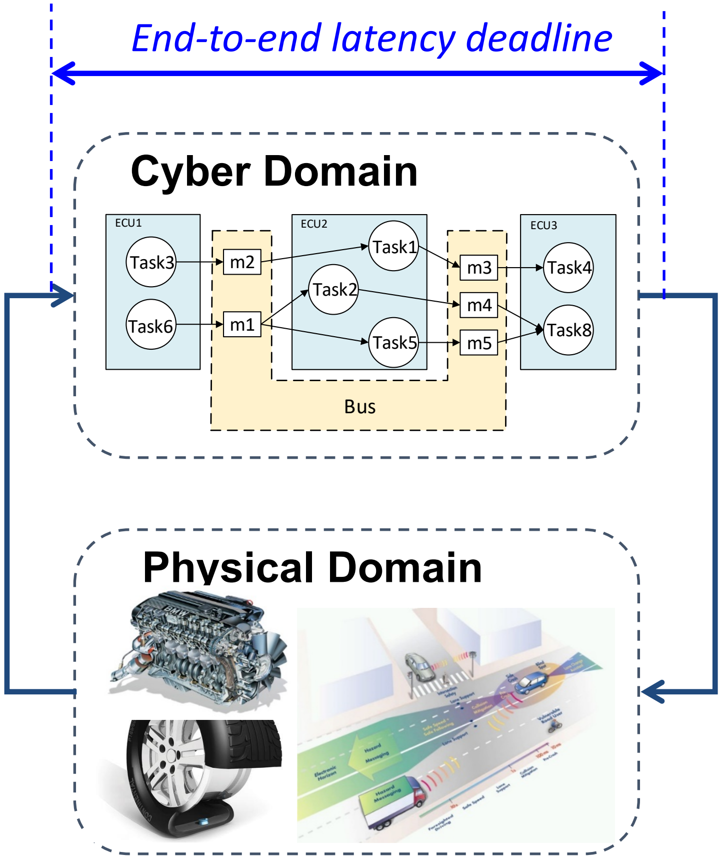
[C. Huang, et al. "Divide and Slide: Layer-Wise Refinement for Output Range Analysis of Deep Neural Networks". EMSOFT, 2020.]

[C. Huang, et al. "ReachNN: Reachability analysis of neural-network controlled systems". EMSOFT, 2019.]

Uncertainties and Disturbances in Automotive CPS

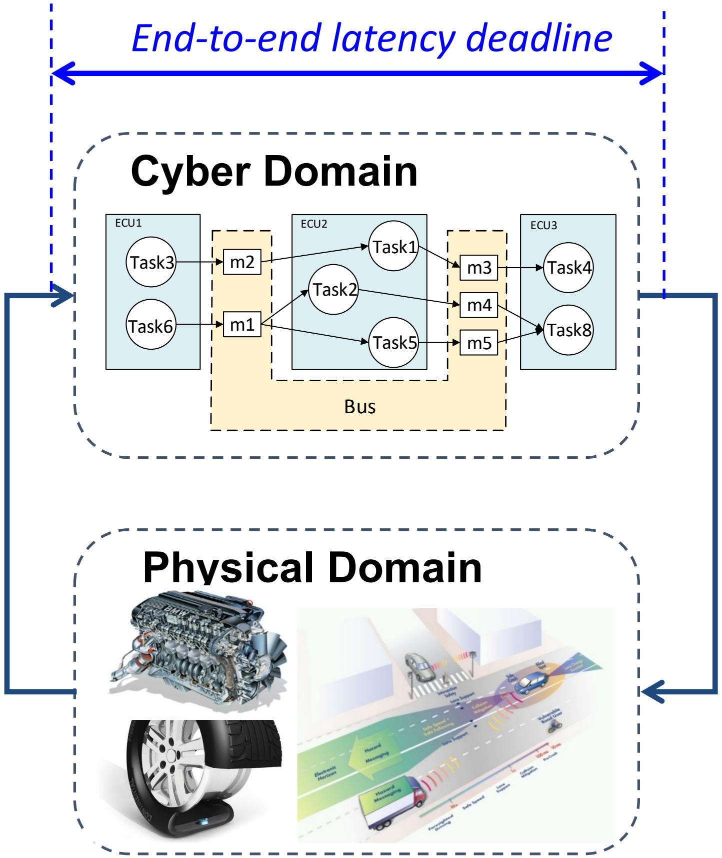


Execution Uncertainties in Cyber (SW-HW) Platform



- Uncertainties/disturbances on operations of computation, communication, storage.
- Various types of execution uncertainties: timing violations, transient errors, malicious attacks, etc.
- The effect of many execution uncertainties is missing deadlines.

Conventional Paradigms for Setting Deadlines (Timing Constraints)

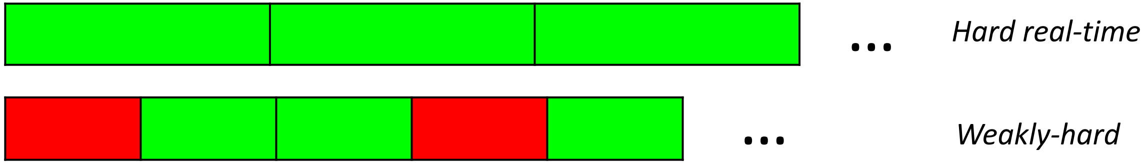


- Hard deadlines
  - Cannot be violated in any circumstance
  - Often require over-conservative worst-case analysis, and lead to infeasible designs or over-provisioning
  - Increasingly hard (*pun-intended*) due to complex function/architecture and uncertain environment

- Soft deadlines
  - Can be violated anytime
  - Cannot provide deterministic guarantees on system properties

Weakly-hard Paradigm for Capturing and Reasoning Uncertainties

- Many system (control, sensing, network) functions can tolerate certain degrees of deadline misses.
- (m,K) constraint: at most m deadline misses among any K consecutive activations *[G. Bernat, et al., 2001]*.



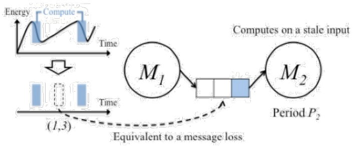
- More flexible than hard real-time; more deterministic guarantees than soft real-time; more general than both.

- Design-time retrofitting: leveraging the allowed *slack* from weakly-hard constraints for adding new functionality/features or fixing existing ones.

- Run-time adaptation: property reasoning and guarantees in challenging environment under timing/fault disturbances.

Key Questions for Weakly-hard Paradigm

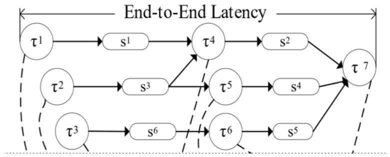
Functional Layer



Can functional/extra-functional properties hold under deadline misses?

["Formal Verification of Weakly-hard Systems", HSCC, 2019.]
["SAW: A Tool for Safety Analysis of Weakly-hard Systems", CAV, 2020]

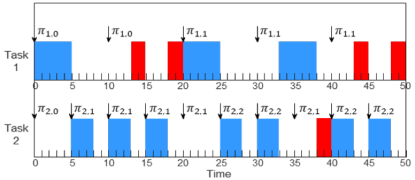
Software Layer



Is the system schedulable under weakly-hard constraints?

A number of approaches in the literature

OS Layer



What OS support is needed?

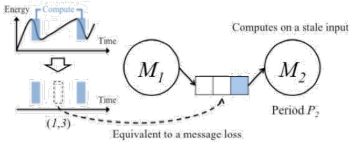
["Job-Class-Level Fixed Priority Scheduling of Weakly-Hard Real-Time Systems", RTAS, 2019.]

How to set weakly-hard constraints for driving system design and adaptation?

["Security-driven Codesign with Weakly-hard Constraints for Real-time Embedded Systems", ICCD, 2019.]
["Opportunistic Intermittent Control with Safety Guarantees for Autonomous Systems", DAC, 2020.]
["Leveraging Weakly-hard Constraints for Improving System Fault Tolerance with Functional and Timing Guarantees", ICCAD, 2020.]

Key Questions for Weakly-hard Paradigm

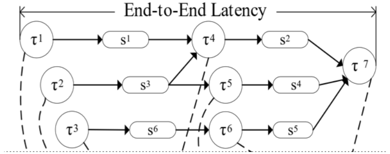
Functional Layer



Can functional/extra-functional properties hold under deadline misses?

["Formal Verification of Weakly-hard Systems", HSCC, 2019.]
["SAW: A Tool for Safety Analysis of Weakly-hard Systems", CAV, 2020]

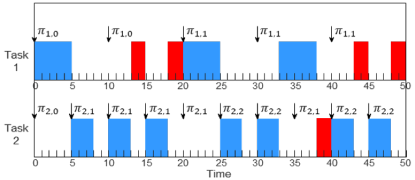
Software Layer



Is the system schedulable under weakly-hard constraints?

A number of approaches in the literature

OS Layer



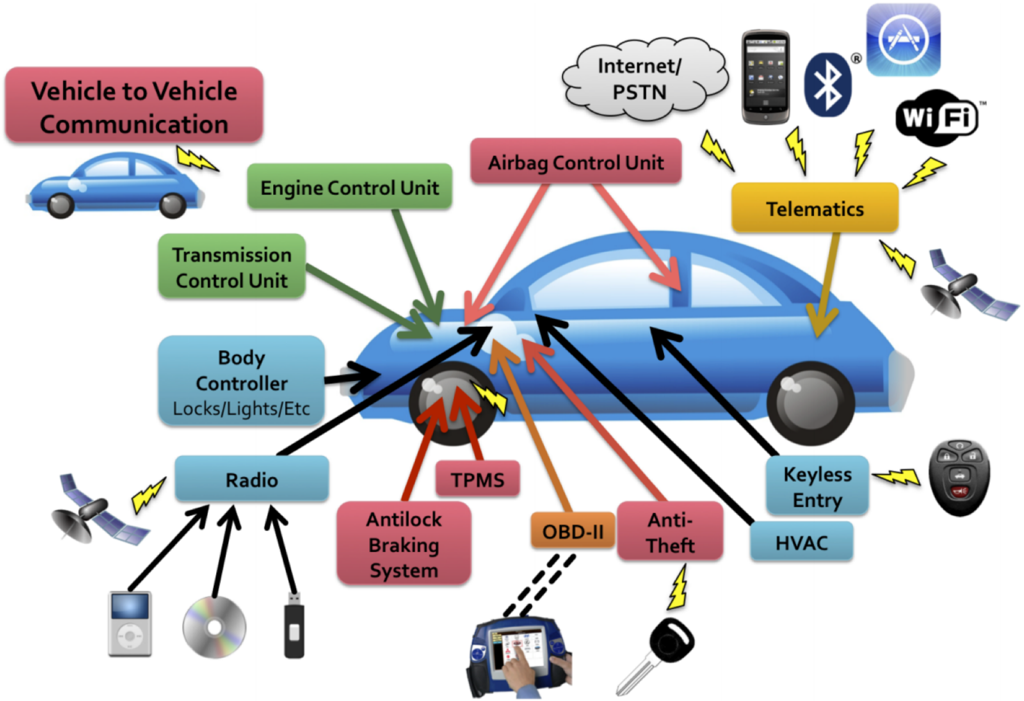
What OS support is needed?

["Job-Class-Level Fixed Priority Scheduling of Weakly-Hard Real-Time Systems", RTAS, 2019.]

How to set weakly-hard constraints for driving system design and adaptation?

["Security-driven Codesign with Weakly-hard Constraints for Real-time Embedded Systems", ICCD, 2019.]
["Opportunistic Intermittent Control with Safety Guarantees for Autonomous Systems", DAC, 2020.]
["Leveraging Weakly-hard Constraints for Improving System Fault Tolerance with Functional and Timing Guarantees", ICCAD, 2020.]

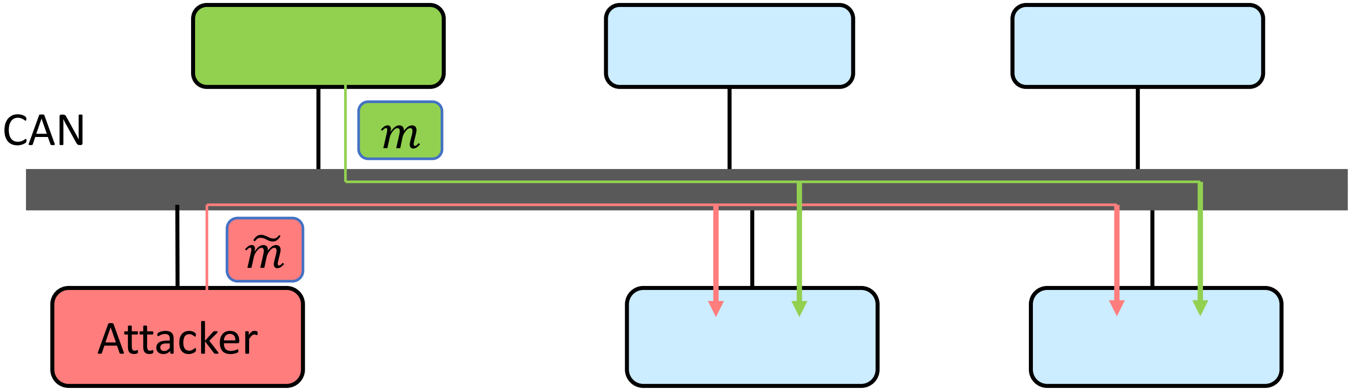
Security Challenges for Automotive Electronic Systems



[Figure Source: S. Checkoway, et al. "Comprehensive Experimental Analyses of Automotive Attack Surfaces". USENIX Security Symposium, 2011.]

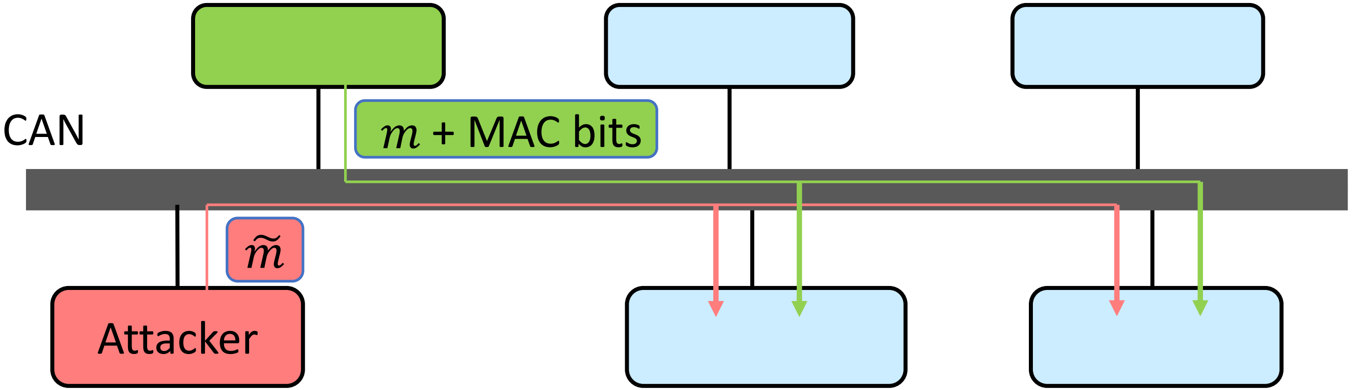
- Various interfaces expose security vulnerabilities.
- Drastic increase of automotive software further exacerbates the problem.

Security Challenges for Automotive Electronic Systems



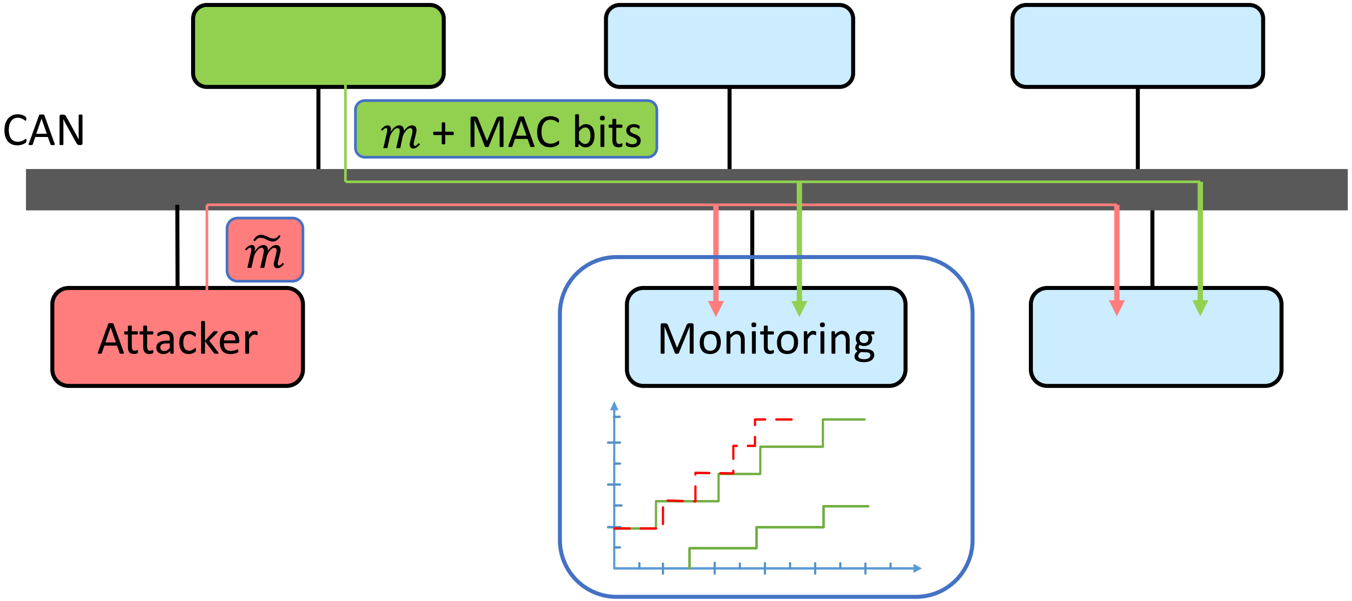
- Lack of built-in security mechanisms in CAN
  - Broadcast messages -> lack of privacy
  - Priority-based scheduling -> DOS attack
  - No message authentication -> masquerade or replay attack

Addressing Security Challenges



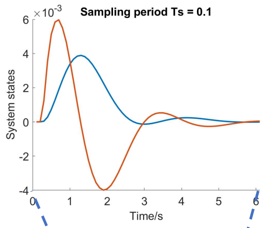
- Lightweight authentication
  - Defend against masquerade and reply attacks
  - Limited resources and timing violations make it infeasible in many cases (*[Lin, et al., TODAES, 2015]*)
  - Even for next-generation Ethernet-based protocols, timing is still a issue.

Addressing Security Challenges

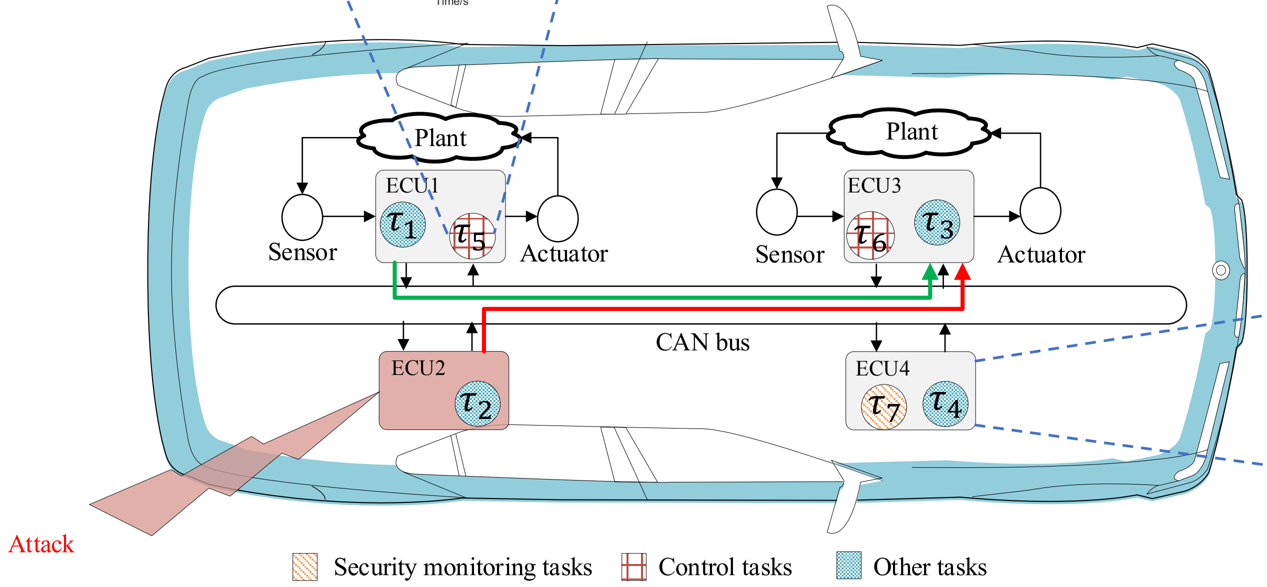


- Lightweight authentication
- Intrusion detection (e.g., by monitoring message streams) – also hard to deploy because of resource limitations and timing constraints

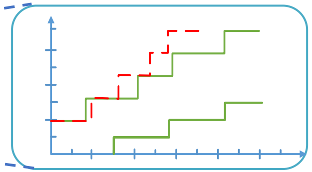
Leveraging Weakly-hard Constraints to Improve Vehicle Security



Allowing deadline misses for certain control tasks based on weakly-hard constraints – safety verified!



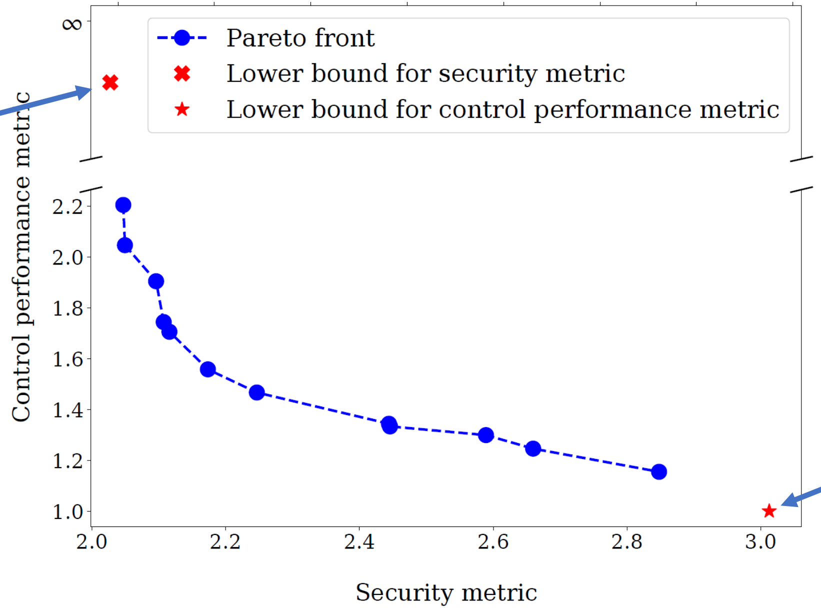
Adding security monitoring tasks with the slack obtained from control tasks



[H. Liang, et al. “Security-driven Codesign with Weakly-hard Constraints for Real-time Embedded Systems”. ICCD, 2019.]

Leveraging Weakly-hard Constraints to Improve Vehicle Security

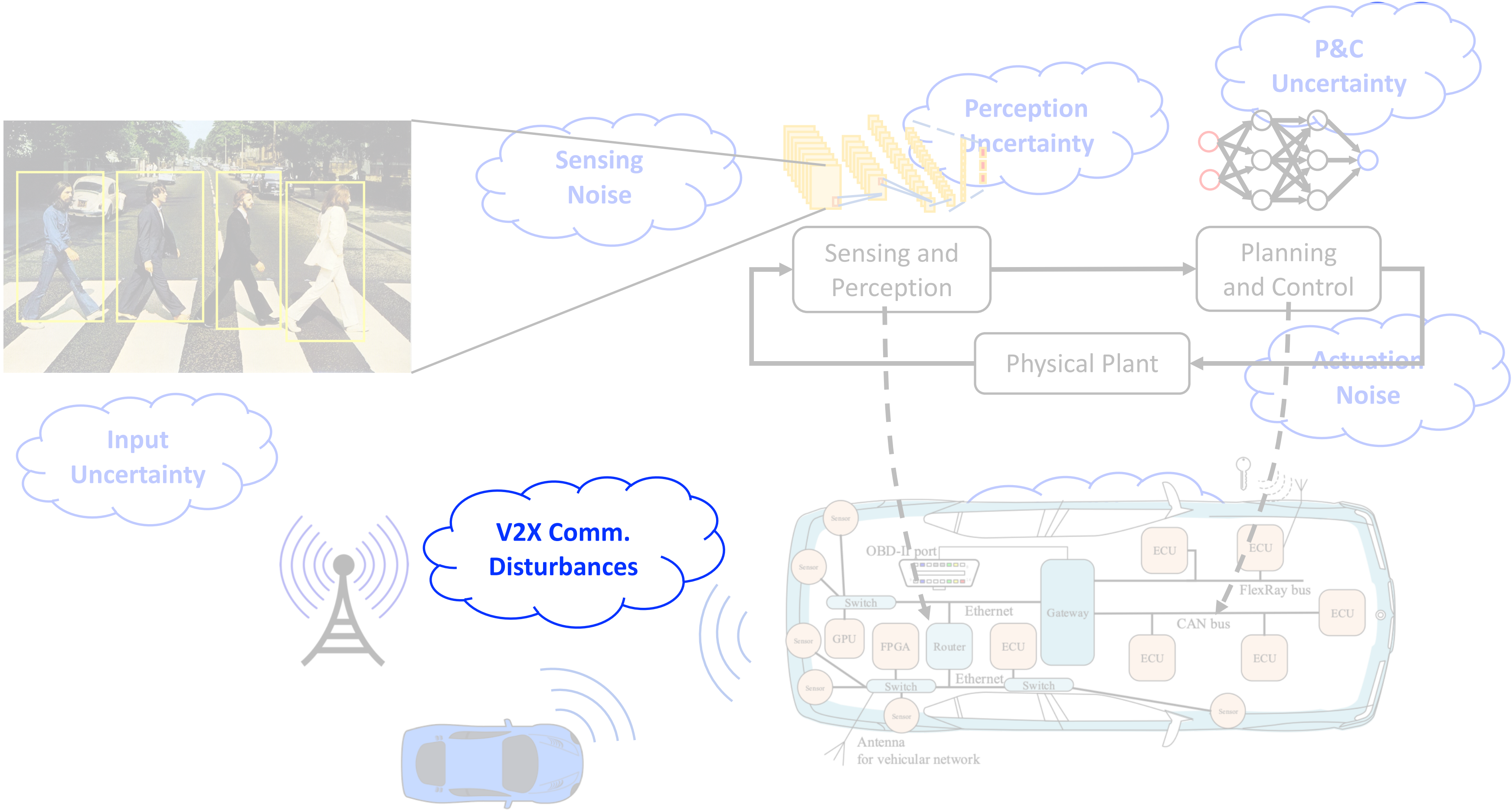
Control tasks become unstable



No deadline miss

Trade-off between security and control performance

Uncertainties and Disturbances in Automotive CPS



Connected Vehicle Applications based on Vehicular Ad-Hoc Network

- Vehicles communicate with each other and infrastructure.
- Share information such as speed, location, acceleration, etc.
- Beyond single-vehicle autonomous driving.
- Many applications in safety, environment, mobility, etc.

V2I Safety

Red Light Violation Warning
Curve Speed Warning
Stop Sign Gap Assist
Spot Weather Impact Warning
Reduced Speed/Work Zone Warning
Pedestrian in Signalized Crosswalk Warning (Transit)

V2V Safety

Emergency Electronic Brake Lights (EEBL)
Forward Collision Warning (FCW)
Intersection Movement Assist (IMA)
Left Turn Assist (LTA)
Blind Spot/Lane Change Warning (BSW/LCW)
Do Not Pass Warning (DNPW)
Vehicle Turning Right in Front of Bus Warning (Transit)

Agency Data

Probe-based Pavement Maintenance
Probe-enabled Traffic Monitoring
Vehicle Classification-based Traffic Studies
CV-enabled Turning Movement & Intersection Analysis
CV-enabled Origin-Destination Studies
Work Zone Traveler Information

Environment

Eco-Approach and Departure at Signalized Intersections
Eco-Traffic Signal Timing
Eco-Traffic Signal Priority
Connected Eco-Driving
Wireless Inductive/Resonance Charging
Eco-Lanes Management
Eco-Speed Harmonization
Eco-Cooperative Adaptive Cruise Control
Eco-Traveler Information
Eco-Ramp Metering
Low Emissions Zone Management
AFV Charging / Fueling Information
Eco-Smart Parking
Dynamic Eco-Routing (light vehicle, transit, freight)
Eco-ICM Decision Support System

Road Weather

Motorist Advisories and Warnings (MAW)
Enhanced MDSS
Vehicle Data Translator (VDT)
Weather Response Traffic Information (WxTINFO)

Mobility

Advanced Traveler Information System
Intelligent Traffic Signal System (I-SIG)
Signal Priority (transit, freight)
Mobile Accessible Pedestrian Signal System (PED-SIG)
Emergency Vehicle Preemption (PREEMPT)
Dynamic Speed Harmonization (SPD-HARM)
Queue Warning (Q-WARN)
Cooperative Adaptive Cruise Control (CACC)
Incident Scene Pre-Arrival Staging Guidance for Emergency Responders (RESP-STG)
Incident Scene Work Zone Alerts for Drivers and Workers (INC-ZONE)
Emergency Communications and Evacuation (EVAC)
Connection Protection (T-CONNECT)
Dynamic Transit Operations (T-DISP)
Dynamic Ridesharing (D-RIDE)
Freight-Specific Dynamic Travel Planning and Performance
Drayage Optimization

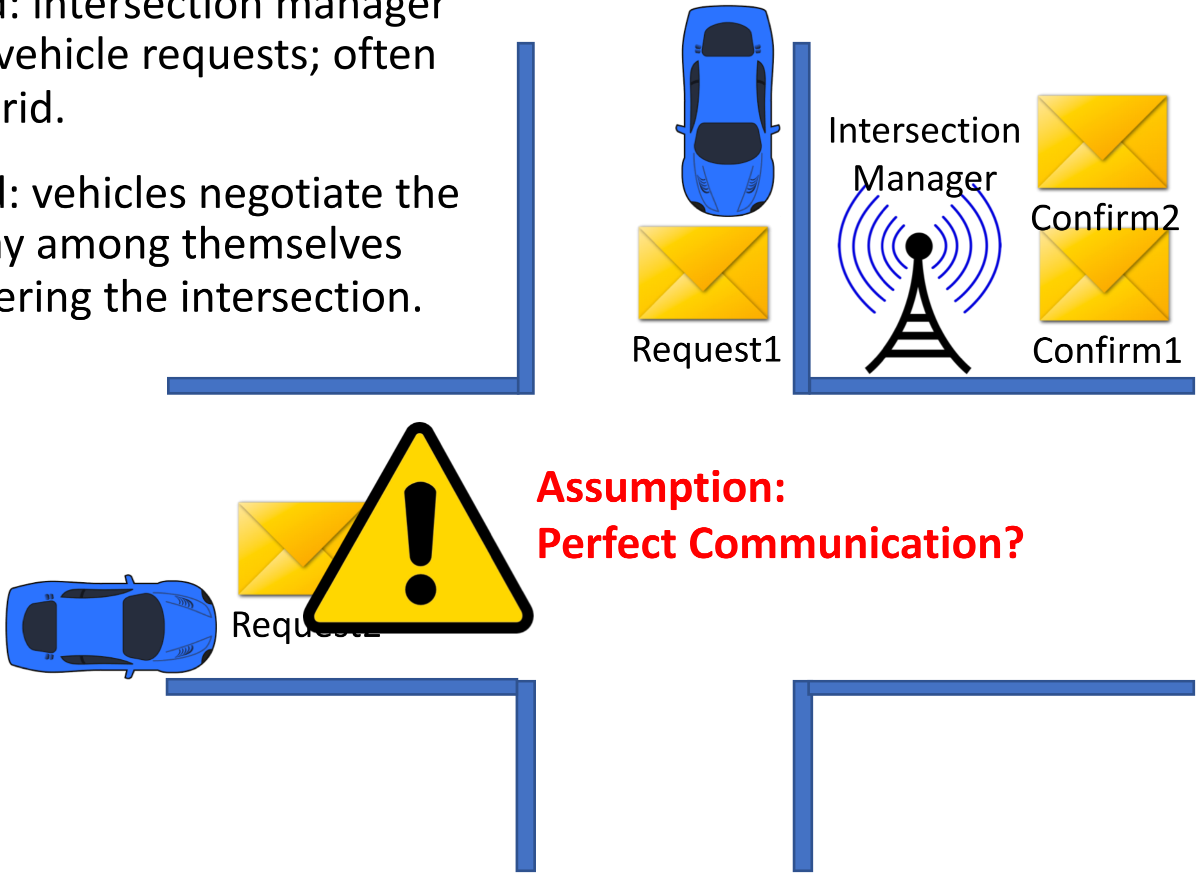
Smart Roadside

Wireless Inspection
Smart Truck Parking

(US DOT)

Autonomous Intersection Management

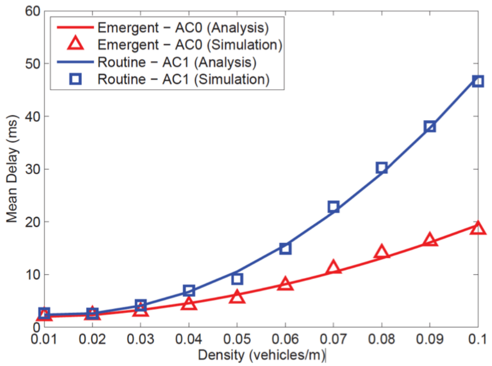
- Centralized: intersection manager schedules vehicle requests; often based on grid.
- Distributed: vehicles negotiate the right-of-way among themselves before entering the intersection.



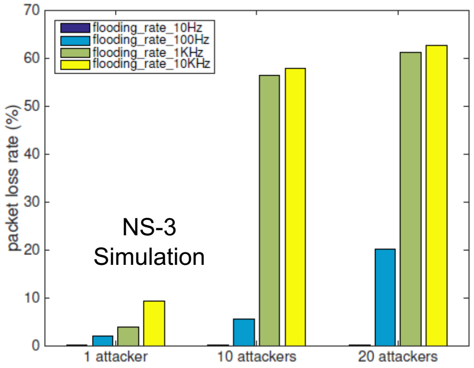
Communication Challenges

- Packet delay and loss
  - DSRC MAC & PHY layer: IEEE 802.11p.
  - Susceptible to significant communication delay and packet collision/loss in crowded traffic.
  - Much worse under jamming/flooding attack.

- Previous intersection management techniques
  - Lack consideration of packet delay/loss.
  - May lead to deadlocks or unsafe situations.
  - May have liveness issues.



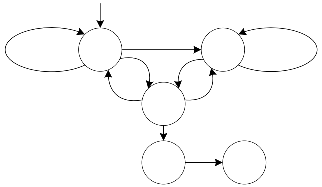
[Y. Yao, et al., "Delay analysis and study of IEEE 802.11 p based DSRC safety communication in a highway environment". INFOCOM, 2013.]



50 vehicles, Road length 300m, Transmission power 26dBm

Our Delay-Tolerant Protocol and Design Tools

Protocol Design and Modeling



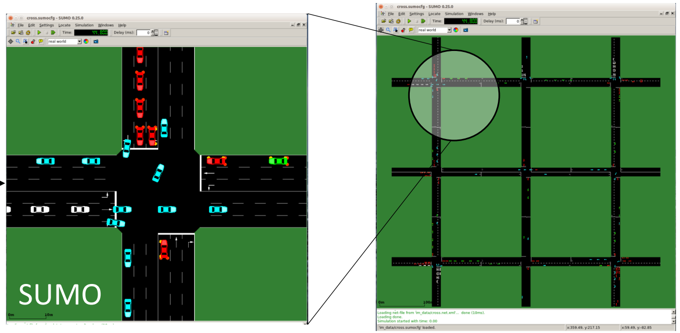
ALGORITHM 3: Intersection Manager

```
Input: currentTime
if receive Request then
    if Ruquest.sender in confirmed[] then
        continue;
    else if Request.sender in inbox[] then
        update Request;
    else
        Store Request
for all vehicle in confirmed[] do
    if vehicle not enter intersection then
        if receive Cancel or [T_L, T_H] expires then
            delete corresponding vehicle in confirmed[];
    else
        if vehicle leaves intersection then
            delete corresponding vehicle in confirmed[];
if currentTime is at manager period then
    if Inbox[].size() == 0 then
        return;
    V_c = front vehicle with earliest arriving time in Inbox[];
    if V_c conflicts with vehicles in confirmed[] then
        return;
    else
        confirmQueueStartWith(V_c);
        for all Request in Inbox[] do
            if Request.sender not conflict with V_c and Request.sender is front then
                confirmQueueStartWith(Request.sender);
```

Simulation

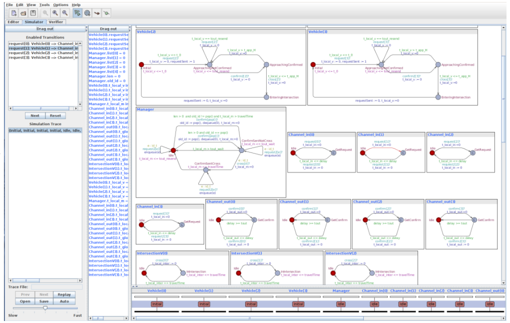
Protocol Simulator

TraCI API



Verification

Timed Automata Models



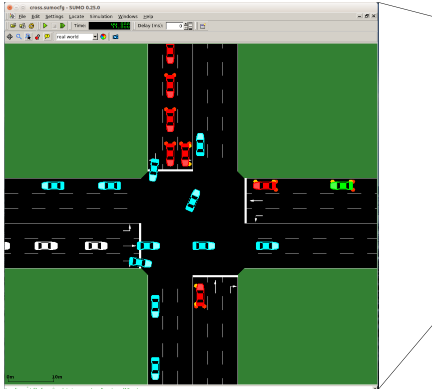
UPPAAL

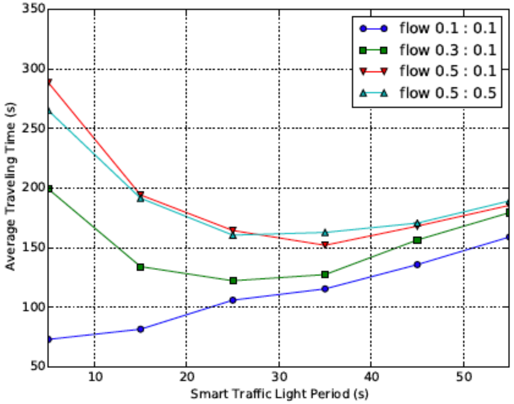
[B. Zheng, et al., "Design and Analysis of Delay-Tolerant Intelligent Intersection Management". ACM TCPS, 2019.]

Verified Properties of Delay-Tolerant Protocol

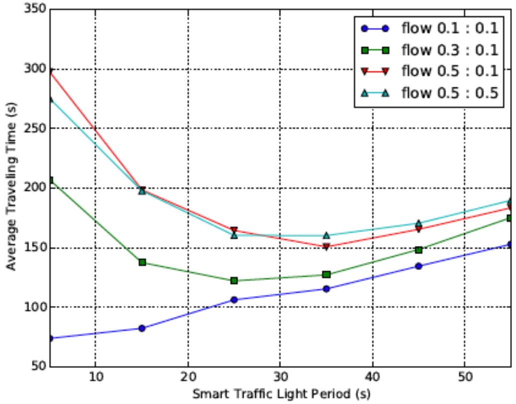
- Guarantee safety even when delay exceeds the estimated bound (considering packet loss/resend).
- Guarantee deadlock-free and liveness if delay is always within the bound.
- Better performance (short traveling time) when delay can be accurately bounded.

Performance Evaluation w/ SUMO-based Simulation



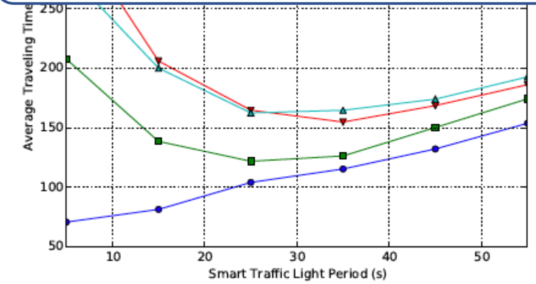


(a) Performance of basic back-pressure control

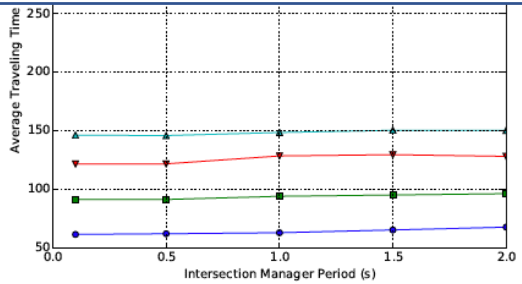


(b) Performance of capacity-aware back-pressure control

Our intelligent intersection design significantly outperforms smart traffic lights under all normal traffic patterns.

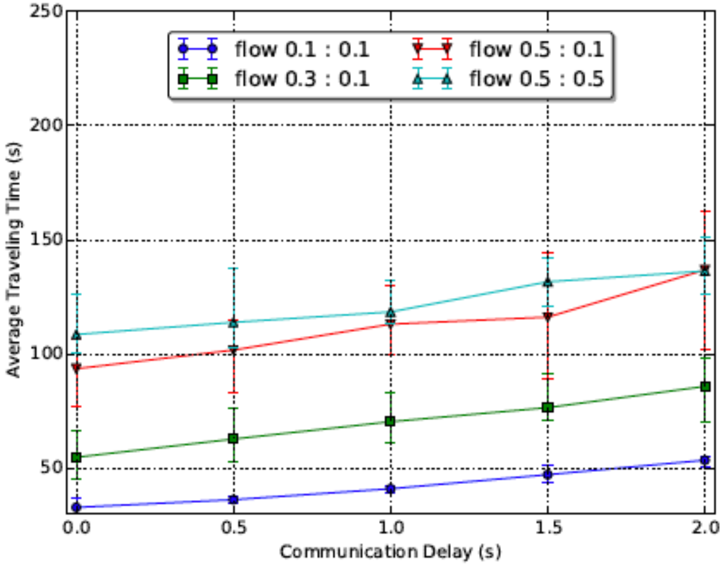


(c) Performance of adaptive max-pressure control

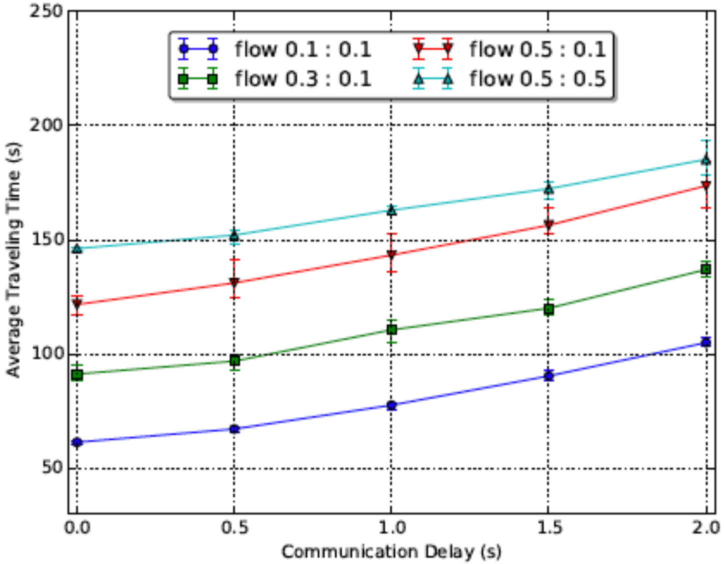


(d) Performance of our intelligent intersection design

Impact of Delay on Intersection Performance



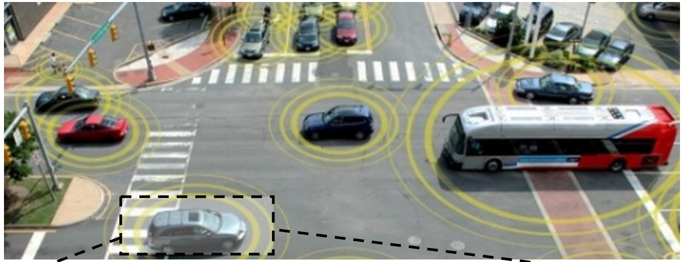
(a) Single intersection



(b) Nine intersections

- Performance degrades with increasing communication delay.
- System-level analysis provides guidelines for lower-layer designs.

CONVINCE: Cross-Layer Design and Validation Framework for Next-Generation Connected Vehicles

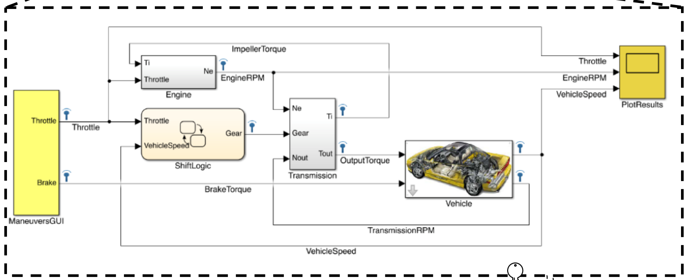


V2X and Self-Driving Applications

Application-level verification, validation and certification

- Functional v/vwith timing consideration
- V2X for autonomous driving
- Vehicle network modeling

Constraints on V2X timing, safety, security, ...

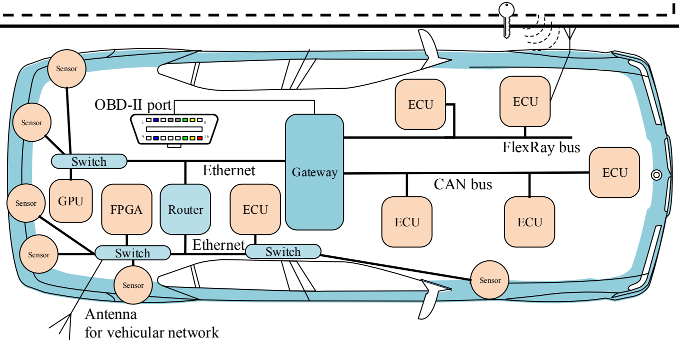


Autonomous Vehicle Software Architecture

Software architecture modeling, synthesis and validation

- Holist task generation and mapping from functional model
- End-to-end timing analysis

Constraints on in-vehicle timing, resource, dependability, ...

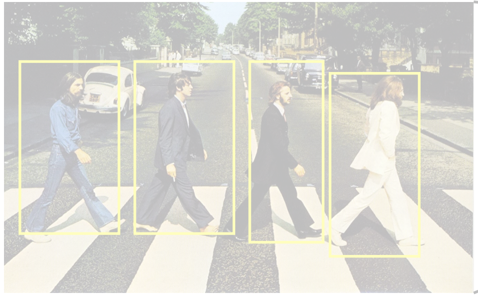


Autonomous Vehicle Hardware Architecture

Hardware architecture modeling and exploration

- Heterogeneous multicore architecture modeling (CPU, GPU, FPGA, Accelerators, ...)
- Efficient architecture exploration

Summary: Addressing Uncertainties and Disturbances in CAVs

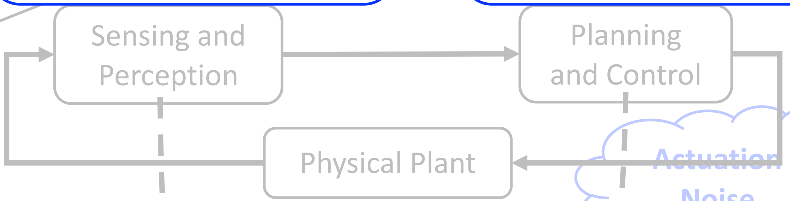


Sensing Noise

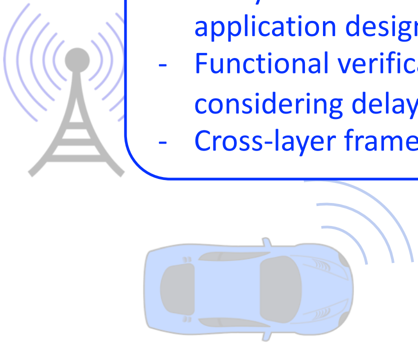
Input Uncertainty

Perception Uncertainty
- Output range analysis (LayR) for addressing NN adversarial attacks

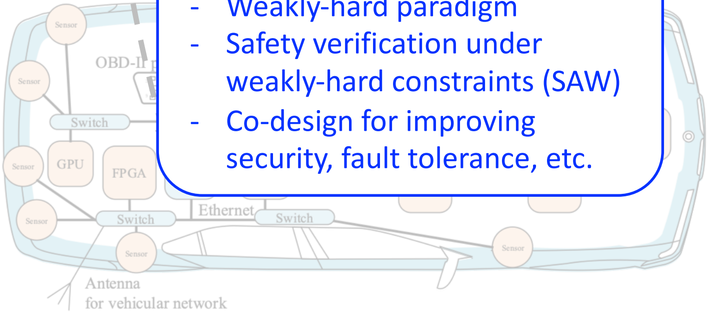
P&C Uncertainty
- Reachability analysis (ReachNN*) for NN safety verification



V2X Comm. Disturbances
- Delay-tolerant V2X application design
- Functional verification considering delay
- Cross-layer framework

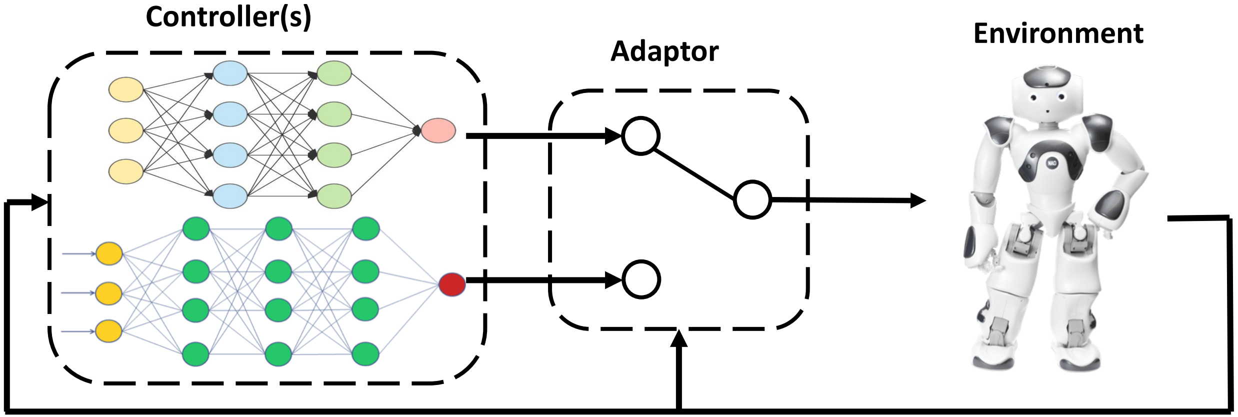


SW-HW Execution Uncertainties
- Weakly-hard paradigm
- Safety verification under weakly-hard constraints (SAW)
- Co-design for improving security, fault tolerance, etc.



Future Direction: Runtime Adaptation with Safety Assurance

- Different controllers may have different strengths and limitation – some have better performance, some are more robust.
- Design an adaptor to switch among multiple controllers, including NN controllers and model-based ones, to accommodate changing environment and missions.
- The key is to provide safety guarantees while doing so.

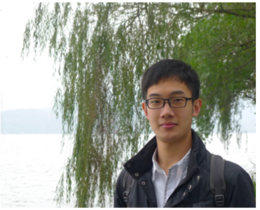


[Y. Wang, et al. "Energy-Efficient Control Adaptation with Safety Guarantees for Learning-Enabled Cyber-Physical Systems ". ICCAD, 2020.]
[C. Huang, et al. "Opportunistic Intermittent Control with Safety Guarantees for Autonomous Systems". DAC, 2020.]

Our Group

Chao Huang (Postdoc):
ML, Formal methods

Hengyi Liang (PhD):
CAVs, MBD, Security

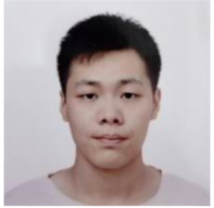


Zhilu Wang (PhD):
MBD, CAVs

Shuyue Lan (PhD):
Embedded Vision

Shichao Xu (PhD):
ML for CPS

Xiangguo Liu (PhD):
CAVs, Control



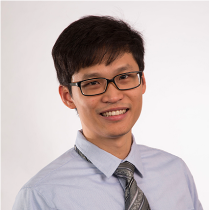
Yixuan Wang (PhD):
ML for CPS

Ruochen Jiao (PhD):
CAVs, MBD

Lixu Wang (PhD):
ML, Security

Our Sponsors

Our Collaborators



Wenchao Li (BU)

Hyoseung Kim (UCR)

Xin Chen (Dayton)

Rolf Ernst (TUB)

Samarjit Chakraborty (UNC)

Chung-Wei Lin (NTU)

Relevant Publications

Output range analysis of neural networks

- Chao Huang, Jiameng Fan, Xin Chen, Wenchao Li and Qi Zhu, “Divide and Slide: Layer-Wise Refinement for Output Range Analysis of Deep Neural Networks”, 20th ACM International Conference on Embedded Software (EMSOFT’20), 2020.

Reachability analysis and safety verification of neural network controlled systems

- Jiameng Fan, Chao Huang, Xin Chen, Wenchao Li and Qi Zhu, “ReachNN*: A Tool for Reachability Analysis of Neural-Network Controlled Systems”, 18th International Symposium on Automated Technology for Verification and Analysis (ATVA’20), 2020.
- Jiameng Fan, Chao Huang, Wenchao Li, Xin Chen and Qi Zhu, “Towards Verification-Aware Knowledge Distillation for Neural-Network Controlled Systems”, 38th ACM/IEEE International Conference on Computer-Aided Design (ICCAD’19), 2019.
- Chao Huang, Jiameng Fan, Wenchao Li, Xin Chen and Qi Zhu, “ReachNN: Reachability Analysis of Neural-Network Controlled Systems”, 19th ACM International Conference on Embedded Software (EMSOFT’19), 2019.

Weakly-hard paradigm

- Hengyi Liang, Zhilu Wang, Ruochen Jiao and Qi Zhu, “Leveraging Weakly-hard Constraints for Improving System Fault Tolerance with Functional and Timing Guarantees”, 39th ACM/IEEE International Conference on Computer-Aided Design (ICCAD’20), 2020.
- Chao Huang, Kai-Chieh Chang, Chung-Wei Lin and Qi Zhu, “SAW: A Tool for Safety Analysis of Weakly-hard Systems”, 32nd International Conference on Computer-Aided Verification (CAV’20), 2020.
- Hengyi Liang, Zhilu Wang, Debayan Roy, Soumyajit Dey, Samarjit Chakraborty and Qi Zhu, “Security-driven Codesign with Weakly-hard Constraints for Real-time Embedded Systems”, 37th IEEE International Conference on Computer Design (ICCD’19), 2019.
- Chao Huang, Wenchao Li and Qi Zhu, “Formal Verification of Weakly-Hard Systems”, 22nd ACM International Conference on Hybrid Systems: Computation and Control (HSCC’19), 2019.
- Chao Huang, Kacper Wardega, Wenchao Li and Qi Zhu, “Exploring Weakly-hard Paradigm for Networked Systems”, ACM/IEEE Design Automation for CPS and IoT (DESTION’19), 2019.
- Hyunjong Choi, Hyoseung Kim and Qi Zhu, “Job-Class-Level Fixed Priority Scheduling of Weakly-Hard Real-Time Systems”, 25th IEEE Real-time and Embedded Technology and Applications Symposium (RTAS’19), 2019.

Relevant Publications

Connected vehicle safety and security

- Xiangguo Liu, Neda Masoud and Qi Zhu, "Impact of Sharing Driving Attitude Information: A Quantitative Study on Lane Changing", IEEE Intelligent Vehicles Symposium (IV'20), 2020.
- Bowen Zheng, Chung-Wei Lin, Shinichi Shiraishi and Qi Zhu, "Design and Analysis of Delay-Tolerant Intelligent Intersection Management", ACM Transactions on Cyber-Physical Systems (TCPS), Vol. 4, No. 1, November, 2019.
- Ahmed Abdo, Sakib Md Bin Malek, Zhiyun Qian, Qi Zhu, Matthew Barth and Nael Abu-Ghazaleh, "Application Level Attacks on Connected Vehicle Protocols", 22nd USENIX International Conference on Research in Attacks, Intrusion and Defenses (RAID'19), 2019.
- Bowen Zheng, Chung-Wei Lin, Hengyi Liang, Shinichi Shiraishi, Wenchao Li and Qi Zhu, "Delay-Aware Design, Analysis and Verification of Intelligent Intersection Management", 3rd IEEE International Conference on Smart Computing (SMARTCOMP'17), 2017.

Runtime Adaptation

- Yixuan Wang, Chao Huang and Qi Zhu, "Energy-Efficient Control Adaptation with Safety Guarantees for Learning-Enabled Cyber-Physical Systems", 39th ACM/IEEE International Conference on Computer-Aided Design (ICCAD'20), 2020.
- Chao Huang, Shichao Xu, Zhilu Wang, Shuyue Lan, Wenchao Li and Qi Zhu, "Opportunistic Intermittent Control with Safety Guarantees for Autonomous Systems", 57th ACM/IEEE Design Automation Conference (DAC'20), 2020.
- Shuyue Lan, Zhilu Wang, Amit Roy-Chowdhury, Ermin Wei and Qi Zhu, "Distributed Multi-agent Video Fast-forwarding", ACM Multimedia (MM'20), 2020.
- Shuyue Lan, Rameswar Panda, Qi Zhu and Amit K. Roy-Chowdhury, "FFNet: Video Fast-Forwarding via Reinforcement Learning", 30th IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR'18), 2018.

Others

- Qi Zhu and Alberto Sangiovanni-Vincentelli, "Codesign Methodologies and Tools for Cyber–Physical Systems", Proceedings of the IEEE, Vol. 106, No. 9, September, 2018.
- Sanjit Seshia, Shiyan Hu, Wenchao Li and Qi Zhu, "Design Automation of Cyber-Physical Systems: Challenges, Advances, and Opportunities", IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems (TCAD), Vol. 36, No. 9, 2017.

Thank you

qzhu@northwestern.edu