

# Can We Make Machine Learning Safe for Safety-Critical Systems?

Thomas G. Dietterich  
Distinguished Professor Emeritus  
Oregon State University  
@tdietterich (X and BlueSky)  
tgd@cs.orst.edu



**Oregon State**  
University

# Outline

- Part 1: Integrating ML into traditional safety engineering processes
  - Scenario-based data collection
  - Verification-based data collection
  - Risk quantification
- Part 2: ML in open worlds: Safety as Control
  - Detecting anomalies, near misses, and departures from the Operational Design Domain
  - Adaptation strategies
- Part 3: Safety as Continual Redesign
  - Design is never finished
  - Resilient systems are “Poised to adapt”

# Traditional Safety Engineering

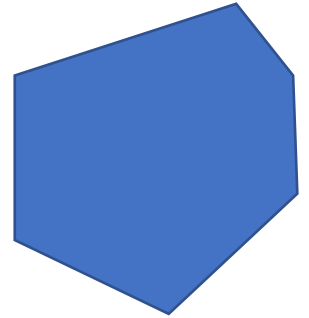
Semi Driving on Freeway

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

# Traditional Safety Engineering

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

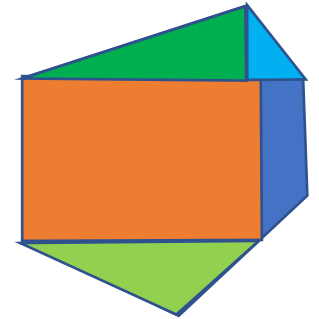
Semi Driving on Freeway



# Traditional Safety Engineering

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

Semi Driving on Freeway



# Traditional Safety Engineering

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

Semi Driving on Freeway



Scenario: Cut in front from left

- Front collision
- Rear collision
- Side collision
- Drive off road

Harms:

- Death
- Severe injury
- Physical damage

# Traditional Safety Engineering

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

Semi Driving on Freeway



Scenario: Cut in front from left

- Front collision
- Rear collision
- Side collision
- Drive off road

Acceptable Risk of Death:  
1 in  $10^8$  hours

# Traditional Safety Engineering

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

Semi Driving on Freeway



Scenario: Cut in front from left

- Front collision
- Rear collision
- Side collision
- Drive off road

Feed forward control

- Minimum following distance
- Decelerate when overtaken
- Decelerate or change lanes when tail-gated



# Traditional Safety Engineering

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

Semi Driving on Freeway



Scenario: Cut in front from left

- Front collision
- Rear collision
- Side collision
- Drive off road

Minimum following distance  
Perception accuracy & reliability  
Braking performance & reliability  
Steering performance & reliability  
Brake light reliability  
Controller response time  
Etc.

# Traditional Safety Engineering

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

Semi Driving on Freeway



Minimum following distance  
Perception accuracy & reliability  
Braking performance & reliability  
Steering performance & reliability  
Brake light reliability  
Controller response time  
Etc.

Component Design  
Controller Design  
Software Development

# Traditional Safety Engineering

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

Semi Driving on Freeway



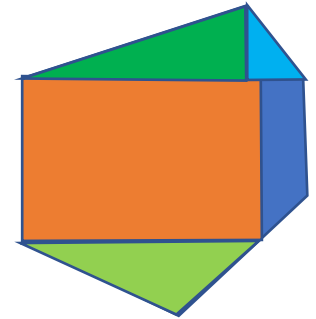
Minimum following distance  
Perception accuracy & reliability  
Braking performance & reliability  
Steering performance & reliability  
Brake light reliability  
Controller response time  
Etc.

Component Design  
Controller Design  
Software Development

# Traditional Safety Engineering

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

Semi Driving on Freeway

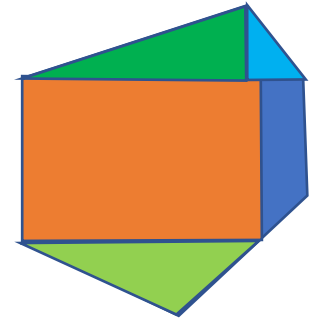


Simulation studies  
Real-world system tests

# Traditional Safety Engineering

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

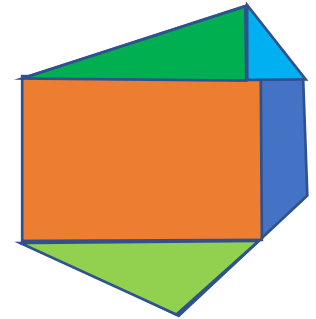
Semi Driving on Freeway



# Traditional Safety Engineering

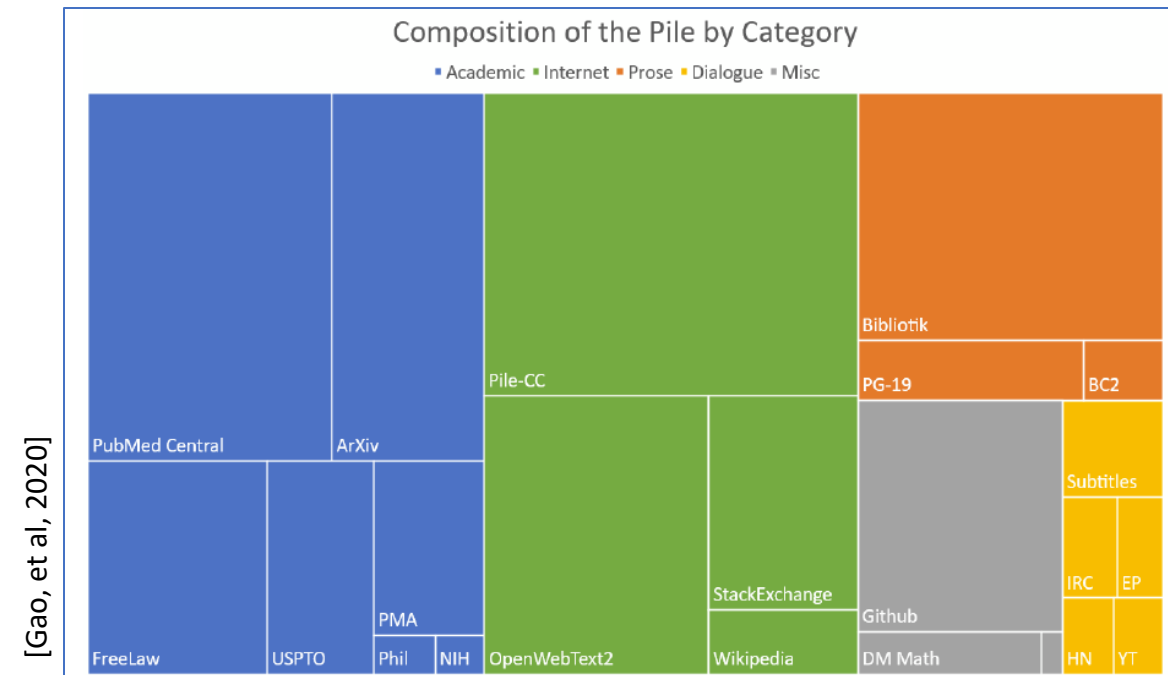
- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- Risk Assessment (likelihood and severity of each harm). Identification of socially acceptable risk
- Mitigation Development: Determine the strategy for achieving socially acceptable risk. Identify the fallback conditions: Minimum Risk Condition
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements. Methods for testing and verifying each safety measure.
- Design and development to meet the requirements
- Verification: The system meets the requirements (safety cases; software testing, model-based verification)
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

Semi Driving on Freeway



# Contrast: Traditional Machine Learning Methodology

- Aggregate data from as many sources as possible
  - Data was often collected for other purposes
  - “Big Data” is “the new oil”



**The Pile: An 800GB Dataset of Diverse Text for Language Modeling**

# Consequences of this Methodology

- No guarantee that the Operational Domain is covered well
- No guarantee that the ML system will learn a model that meets the safety requirements
- Learning Theory only provides statistical guarantees for inputs drawn from the same distribution as the training data
- If the actual distribution in operations concentrates on a region of poor coverage, error can be arbitrarily large/serious

**We need a new methodology**



# Achieving Distribution-Independent Accuracy in Machine Learning Components

- Deliberately collect training data to attain good coverage of all cases (including corner cases)
  - Risk-driven sampling techniques (e.g., [Wang, et al., 2023])
- Verify approximation quality of the learned model
  - Collect additional examples as needed

# Sampling via Surrogate Model Optimization

(also known as Bayesian Optimization)

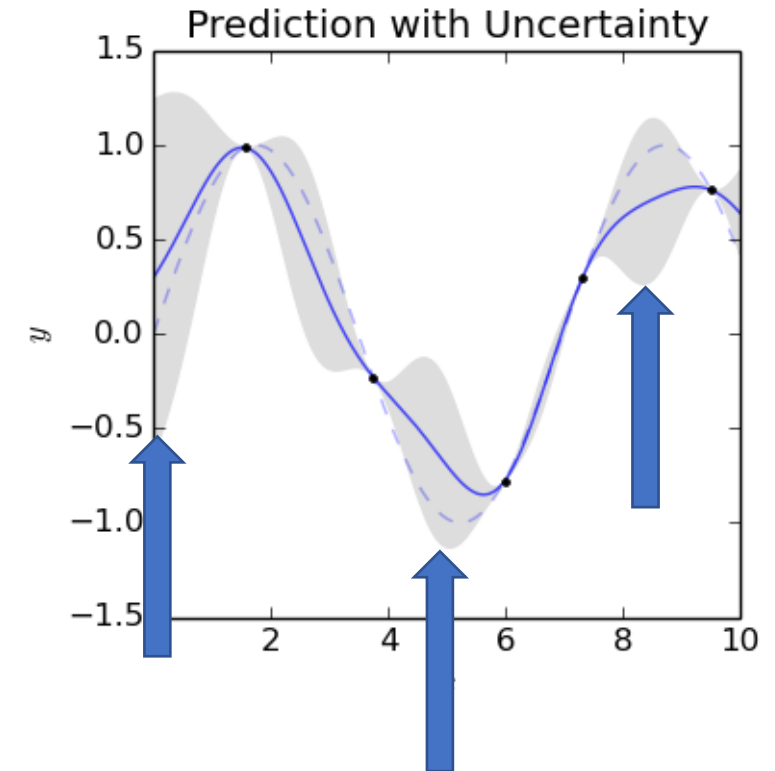
- Collect an initial real-world sample and train the ML component
- Build and validate a simulation model (“digital twin”)

## Repeat

- Fit surrogate model (e.g., Gaussian Process)
  - Provides estimates of “epistemic uncertainty”
- Select a batch of new cases using an “acquisition function”
  - Collect training example for each case using simulation
  - Update the ML component model
  - Update the surrogate model

Until target metrics are attained

- Metrics of interest:
  - Reasonable coverage of the state space
  - Good coverage of hazardous states



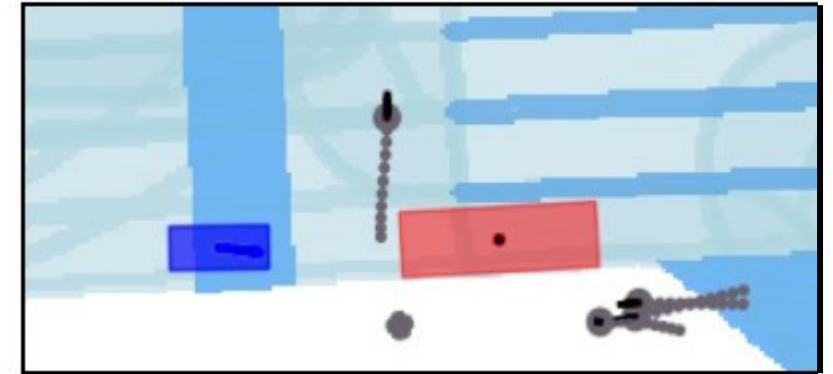
Cdipaolo96 - Own work, CC BY-SA 4.0

# Generating an Adversarial Scenario with AdvSim

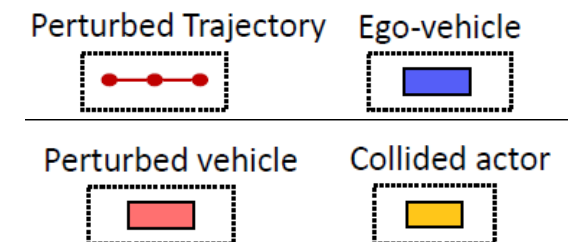
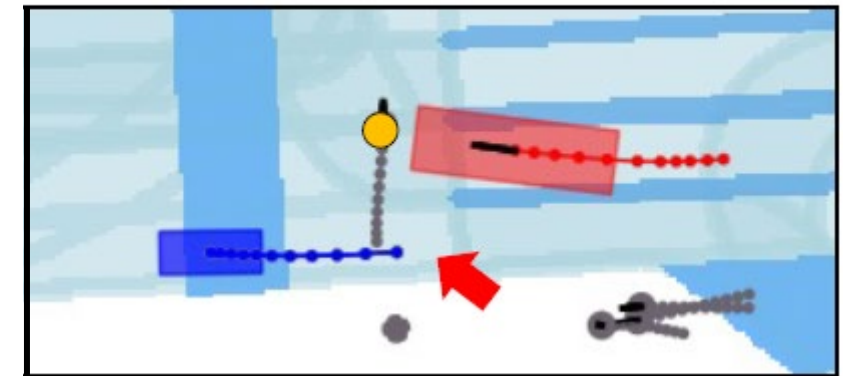
[Wang, et al., 2023]

- Given: original trajectory from expert driver
  - trajectories of all “actors” (vehicles, pedestrians, cyclists), LiDAR data, map
- Select one or more vehicles and perturb their behavior to maximize an adversarial loss for the end-to-end system
  - collisions, law violations, passenger discomfort
  - Perturbation is at the level of a kinematic trajectory (acceleration and curvature)
- Simulate the perturbed LiDAR data
- Run the current end-to-end system
- Score the adversarial loss
- Repeat  $N$  times and keep the perturbation with the largest adversarial loss

Original Scenario

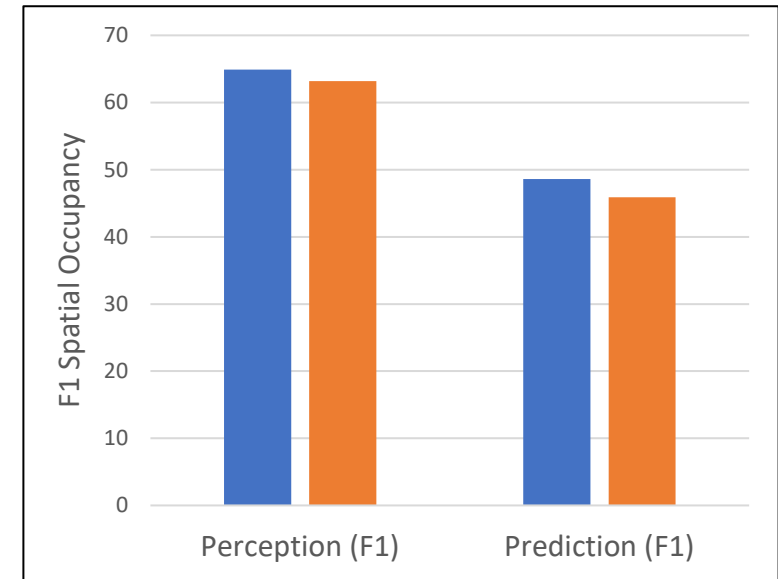
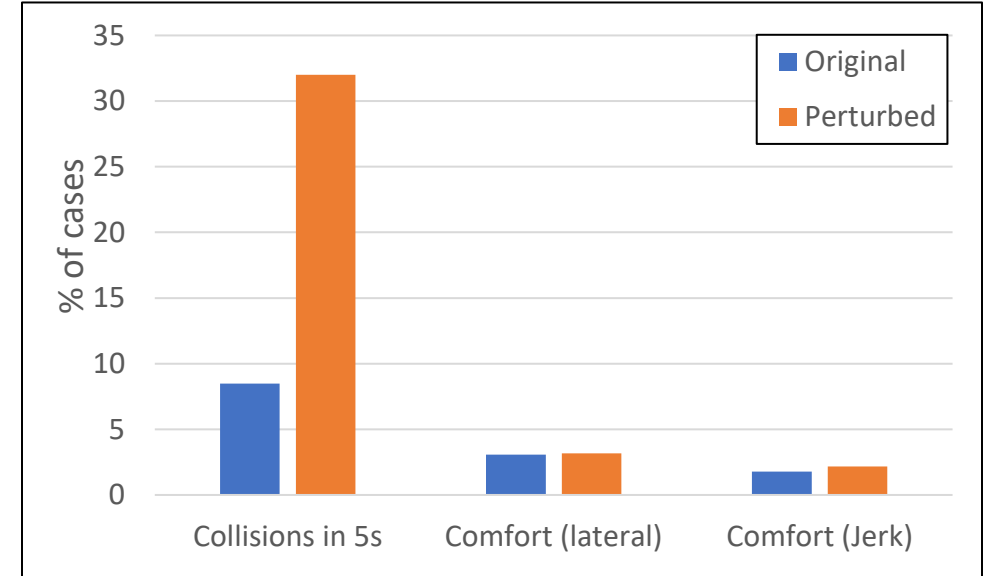


AdvSim Scenario



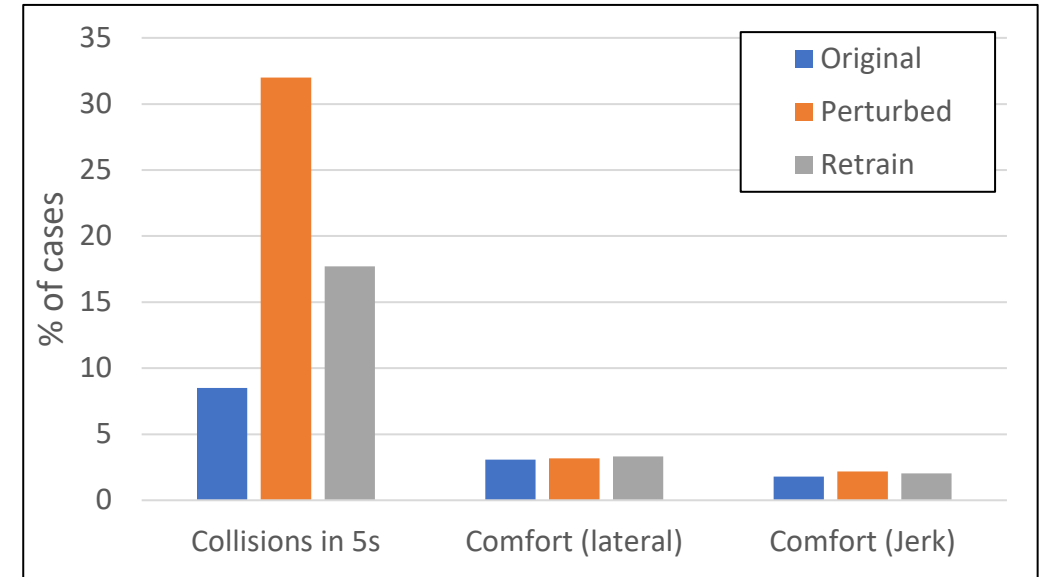
# AdvSim Results

- 376% increase in collisions
- Small increases in discomfort
- Decreases in accuracy of perception and predicted trajectories



# Updating the Model

- Retrain on the original data + adversarial cases
  - Collisions reduced to 17.7%
- 
- To be determined: What improvement is possible with more iterations?

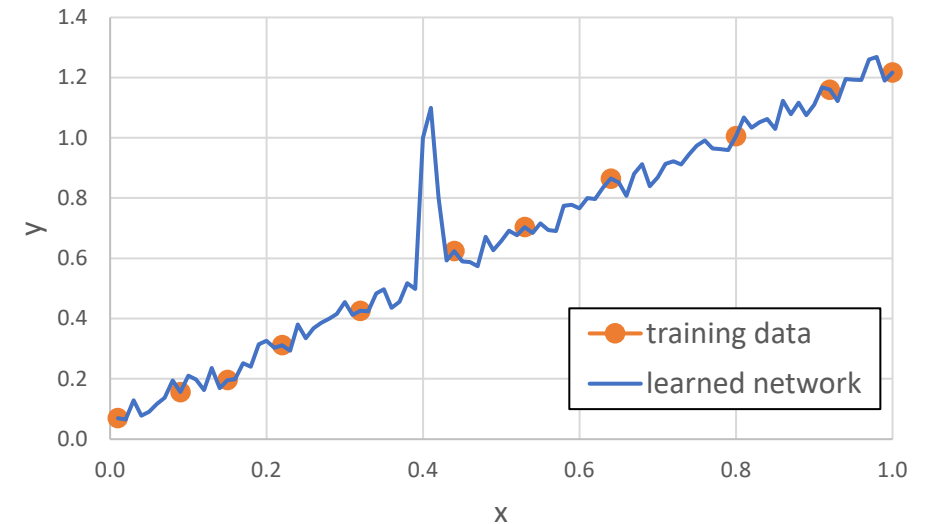


# Notes

- AdvSim operates at the semantic level (agent behavior)
  - Much smaller search space than searching in image space
  - Requires high fidelity simulation of imaging
- Adversarial collision rate is much higher than expected rate under normal driving conditions
  - See below

# Verifying Correct Behavior of ML Component

- How can we gain assurance that the ML system has learned the correct function?
- We have no explicit specification of correctness, but we can detect bad behavior
- Are there regions where the learned function behaves badly?

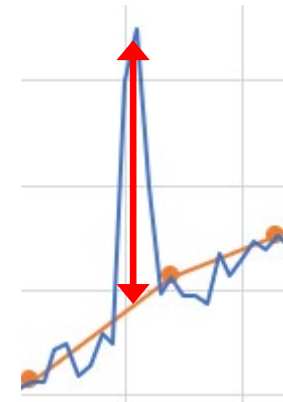
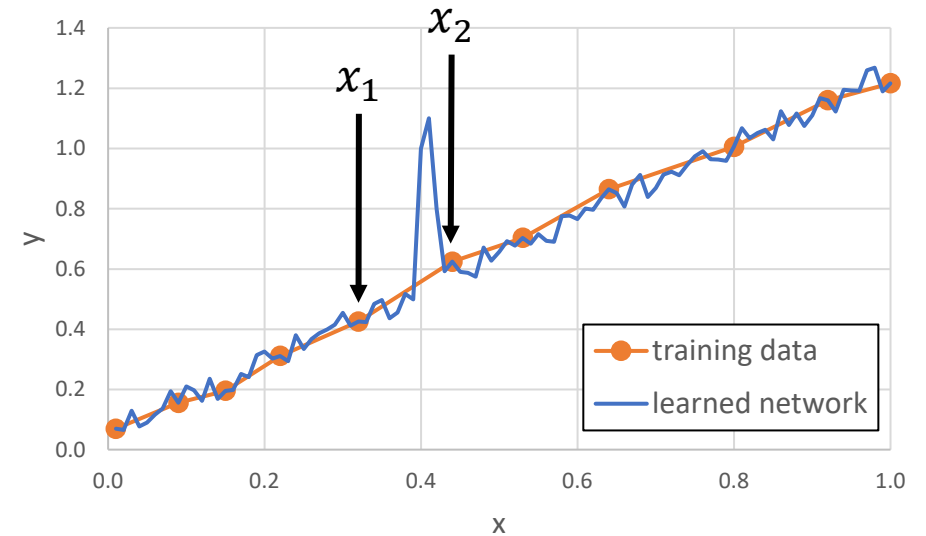


# Proposal: Bound the difference between the fitted function and linear interpolation of the training data

- Consider two adjacent training examples  $x_1$  and  $x_2$
- Let  $\alpha \in [0,1]$

$$\max_{\alpha} \left| \frac{f(\alpha x_1 + (1 - \alpha)x_2)}{\alpha f(x_1) + (1 - \alpha)f(x_2)} \right|$$

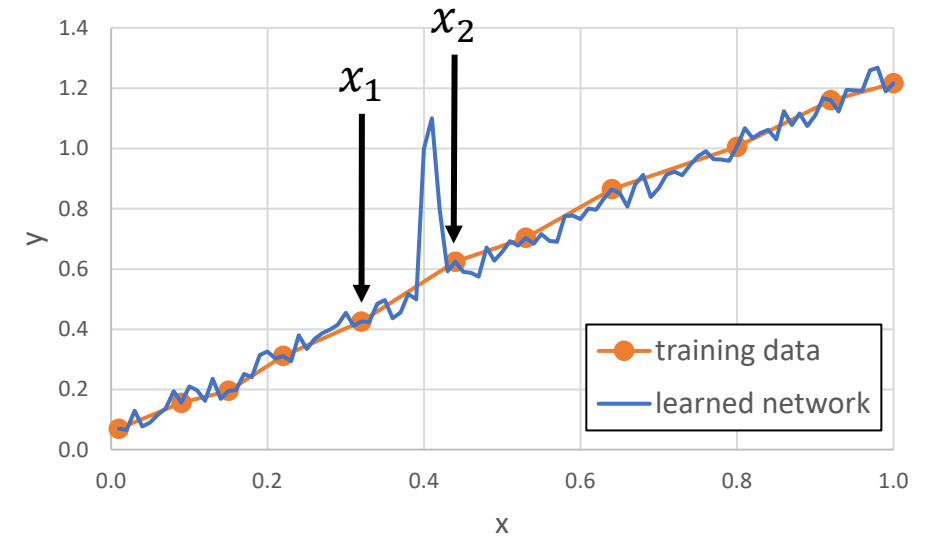
- If this is small, the function behaves well in between the training data
- This can be solved by the methods of [Singh, et al. 2021] (but those may not scale)





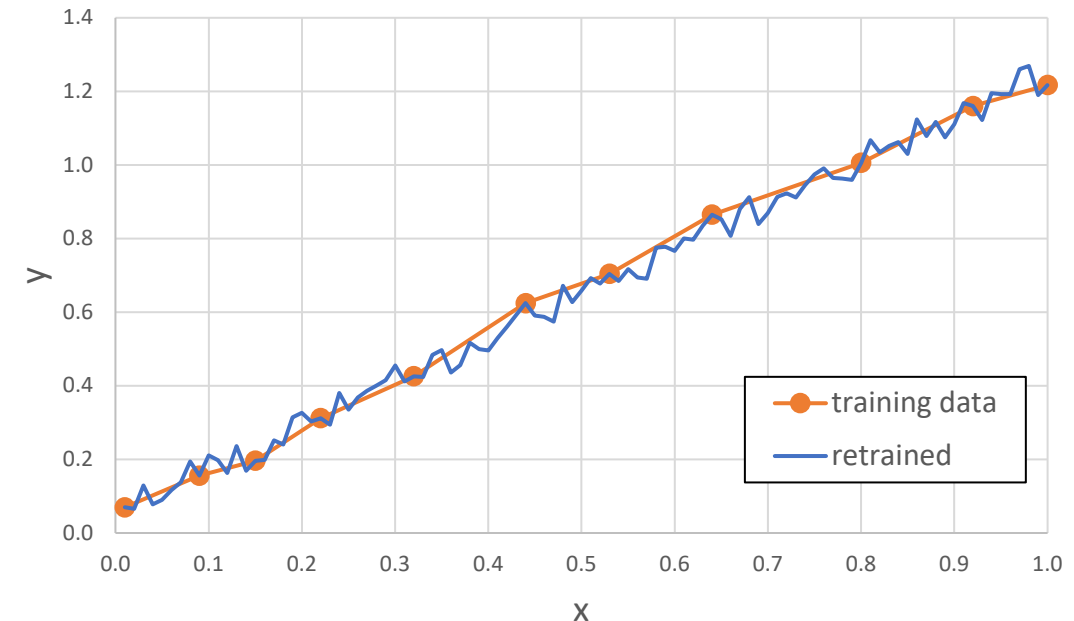
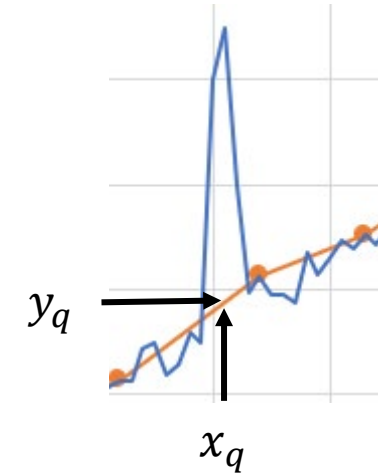
# Interpolating in the Right Space

- This looks great in 1 dimension ...
- We need to interpolate in a semantic space (like AdvSim)
  - For each known training case, identify the  $k$  most similar cases, where  $k \approx$  the number of parameters in a scenario
  - Consider all convex combinations of those  $k + 1$  cases to find the maximum discrepancy between a linear interpolation and the fitted neural network



# Verification-Based Active Learning

- $\alpha_q := \operatorname{argmax}_{\alpha} \left| \frac{f(\alpha x_1 + (1 - \alpha)x_2)}{\alpha f(x_1) + (1 - \alpha)f(x_2)} \right|$
- Generate a new example at
- $x_q = \alpha_q x_1 + (1 - \alpha_q)x_2$
- Obtain  $y_q$
- Retrain the network on  $(x_q, y_q)$
- Repeat until no failure regions can be found



# Estimating Risk = Probability of Harm

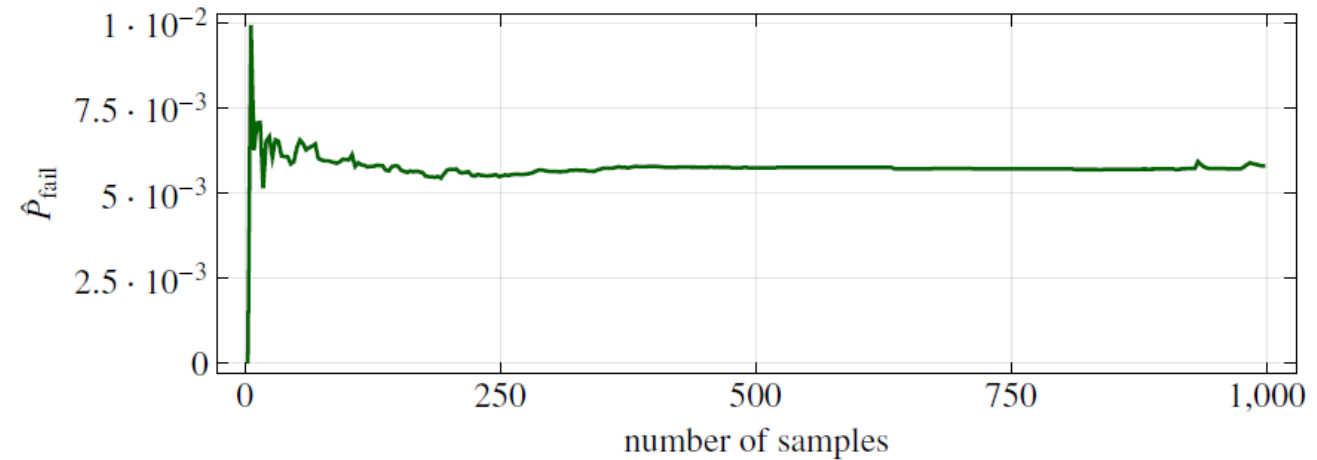
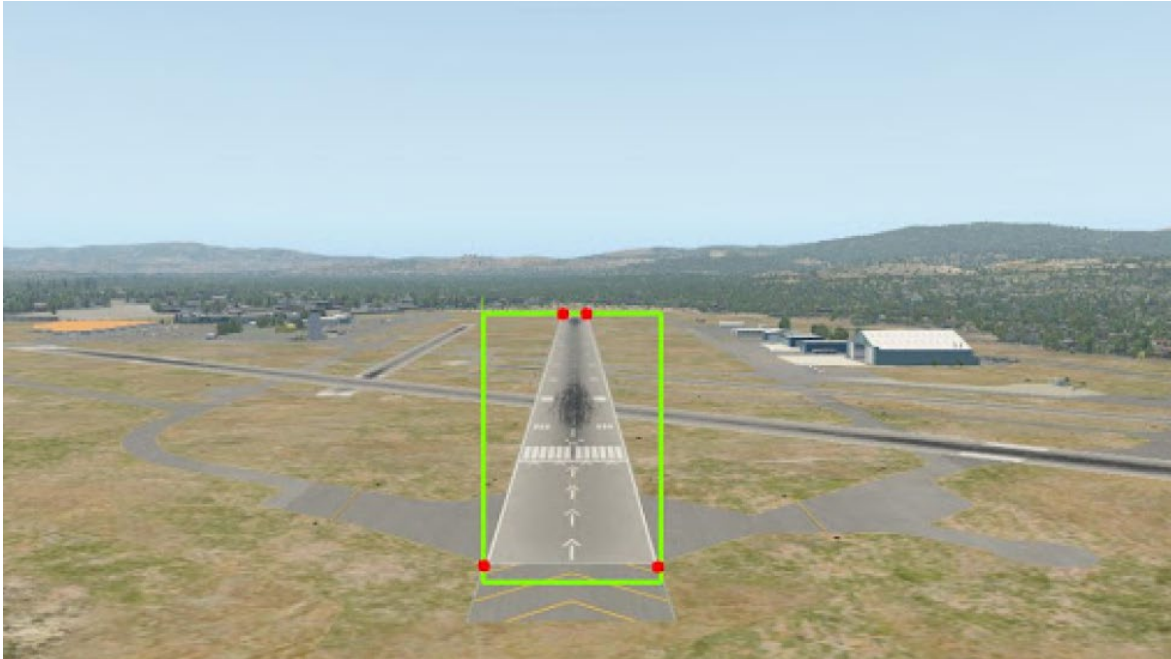
- If we have discarded the data distribution, how can we estimate risk?
- Answer: Simulate system operation and measure the probability of harm
- Challenges:
  - We must be able to simulate system operation
  - Harms are very rare
- Solution:
  - Fit a probabilistic model  $P(s)$  of normal operations
    - Probability of initial states
    - Probability of system behavior
    - Probability of the behaviors of other agents
  - Develop a proposal distribution  $Q(s)$  that greatly increases the probability of harms
    - Reuse our design data?
  - Simulate according to  $Q(s)$
  - Apply importance reweighting  $\frac{P(s)}{Q(s)}$  to each hazardous state  $s$  that is observed

# Bayesian Risk Estimation

[Moss, Kochenderfer, Gariel, Dubois, 2024]

- Apply Surrogate Model Optimization to discover failure regions
- Combine three “acquisition functions”
  - Explore regions of high operational likelihood  $P(x)$  and high epistemic uncertainty
  - Explore regions near the boundaries of failure regions (hazards)
  - Explore the interiors of the failure regions

# Example: Runway Detection



57.2% of samples were in failure regions

$$\hat{p}_{fail} = 5.8 \times 10^{-3}$$

Only 0.6% of Monte Carlo samples are in failure regions

# With these tools, ML can be integrated into the safety engineering process

- Define the intended scope of the system
- Define the operational domain (OD)
- Decompose OD into scenarios
- Hazard Analysis of each scenario
- **Risk Assessment (likelihood and severity of each harm).**
- Mitigation Development: Determine the strategy for achieving socially acceptable risk.
- Safety Requirements Specification: defining the safety requirements for the system including performance and reliability requirements.
- **Design and development to meet the requirements**
- **Verification: The system meets the requirements (safety cases; software testing, model-based verification)**
- Validation: Checking that the system performance meets the application needs
- Maintenance and Monitoring: Ensuring all components are performing at required levels.
- Certification and documentation: Regulatory requirements

N.B. No single validation method suffices to ensure safety. See [Kochenderfer, et al., *Algorithms for Validation* (forthcoming)]

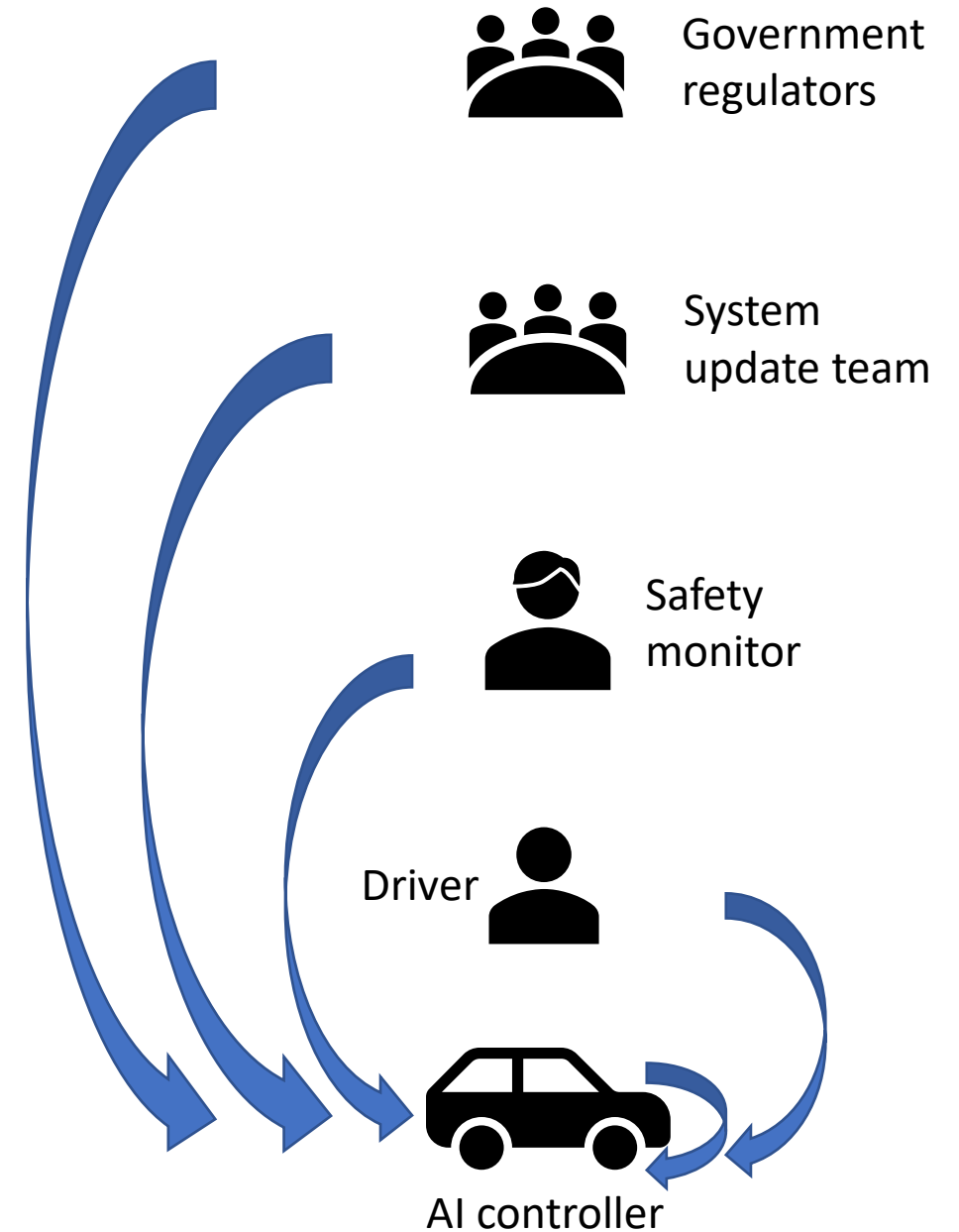
# Outline

- Part 1: Integrating ML into traditional safety engineering processes
  - Scenario-based data collection
  - Verification-based data collection
  - Risk quantification
- Part 2: ML in open worlds: Safety as Control
  - Detecting anomalies, near misses, and departures from the Operational Design Domain
  - Adaptation strategies
- Part 3: Safety as Continual Redesign
  - Design is never finished
  - Resilient systems are “Poised to adapt”

# Systems View of Safety

[Leveson 2011: Engineering a Safer World]

- Safety is a control problem
  - Maintain the safety of the system in the presence of disturbances
- What is the “controller”?
  - The human organizations that build, operate, and maintain it
  - Government regulators
  - Elected officials
- What are the “disturbances”?
  - Budget cuts and staff reductions
    - Systems tend to migrate toward the edges of safety
  - Unknown unknowns
    - Environmental Novelty
- The controller must detect and compensate for these disturbances
  - Today: It is the exclusively the humans who do this
  - Can AI help?





# Unknown Unknowns: Detecting Novel Failure Modes

- Key performance indicators [Weick, et al., 1999]
  - Number of anomalies detected
  - Number of near misses detected
- These provide evidence of novel failure modes *before* they cause harms
- What is the status of AI methods for detecting anomalies and near misses?

# Anomaly Detection in Computer Vision

- Extensively studied for the past 10+ years
  - [Ruff, et al., 2021; Dietterich & Guyer, 2022]
- Advances in deep learning and vision foundation models have produced major improvements
- No method can guarantee to detect all novelty

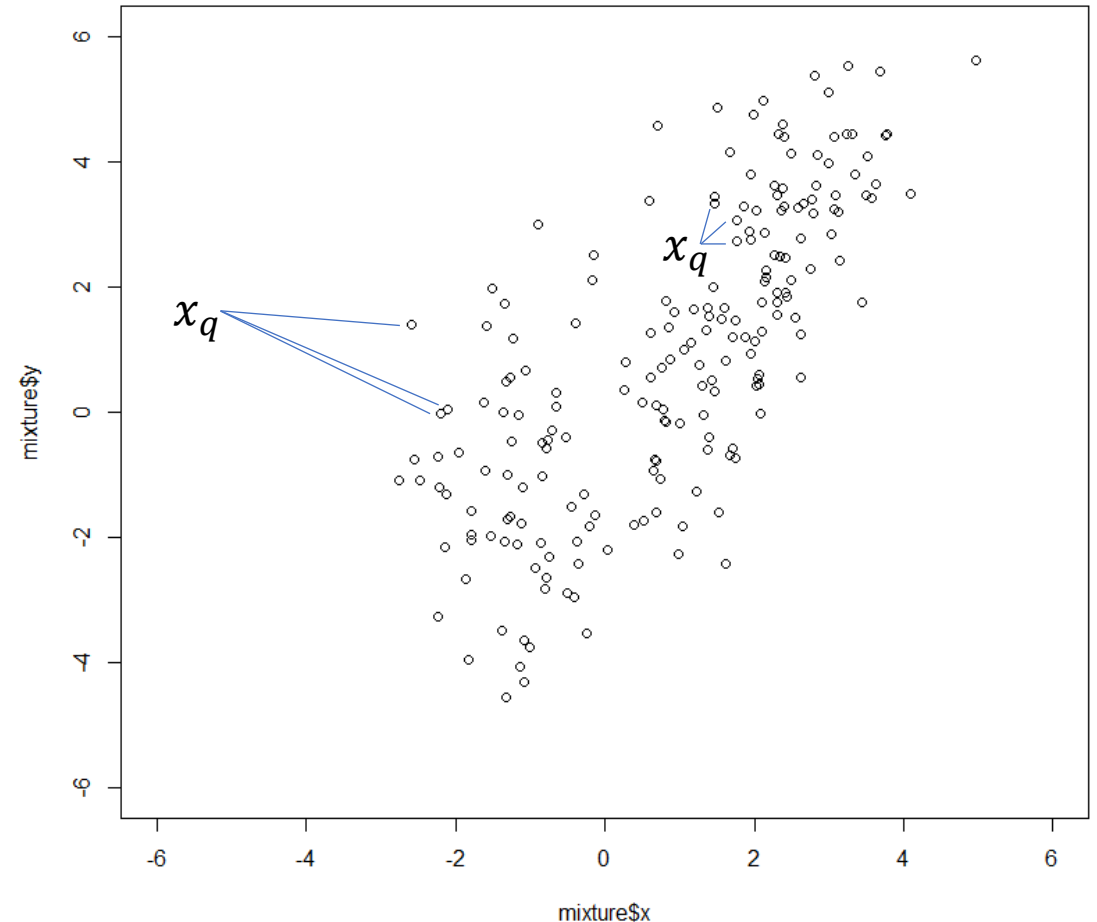


# Detecting Novel Hazards

- Perception Failures
  - Novel objects
  - Novel imaging conditions
  - Insufficient sensors
- Control Failures
  - Near misses
  - Collisions

# Novelty Detection in Machine Learning

- Distance-Based Methods
  - Define a distance  $d(x_i, x_j)$
  - Given a query  $x_q$ , compute
$$\min_{x \in D} d(x_q, x)$$
- Density-Based Methods
  - Fit a probability density  $P(x_i)$
  - Given a query  $x_q$  compute
$$-\log P(x_q)$$
  - Densities are always dependent on distances



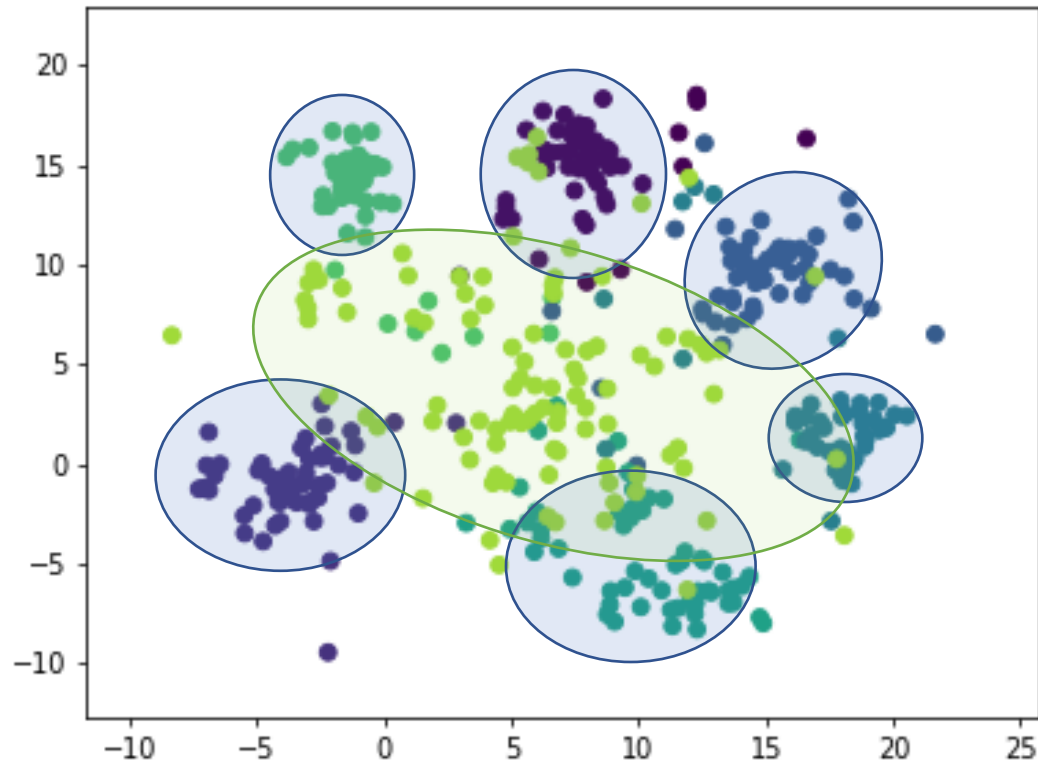
# Deep Anomaly Detection

- An important advantage of deep learning is that it learns its own internal features
  - Euclidean distance in pixel space is not useful
- Problem: It only learns the internal features that it needs for the training task. These features may not separate novel queries  $x_q$  from nominal data

# Experiment:

## Deep Learned Features in Computer Vision

- DenseNet with 384-dimensional latent space.
- CIFAR-10: 6 known classes, 4 novel classes
- Light green: novel classes
- Darker greens: known classes
- Images from known classes are “pulled out” from the center of the space
- Most novel-class images stay toward the center of the space; others overlap with known classes
- Novel images are “inliers”



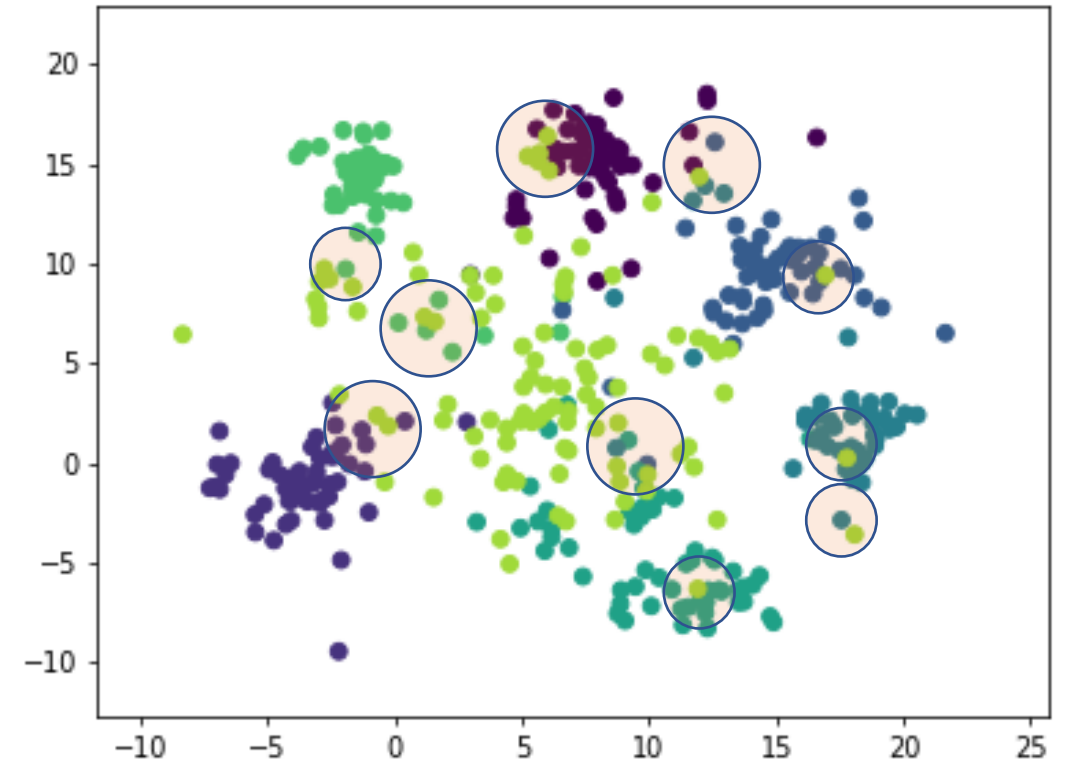
6 Known  
Classes

4 Novel  
Classes

Dietterich & Guyer, 2022

# The Learned Representation is Promising But Not a Complete Solution

- Many novel-class images are mapped into clusters of known images
- ➔ The learned representation can't detect the novelty



# How can we learn better features?

- Foundation Model Approach:
  - Train on all the data we can find
  - Artificially introduce variation through augmentations
    - Rotations, flips, simulated snow, rain, pixel noise, etc.
  - Synthetic data
- The deep representation learns to “see” (represent) the known world
  - A OneWheel will still be novel, but the model should have the right features to represent it and thereby separate it from all known objects

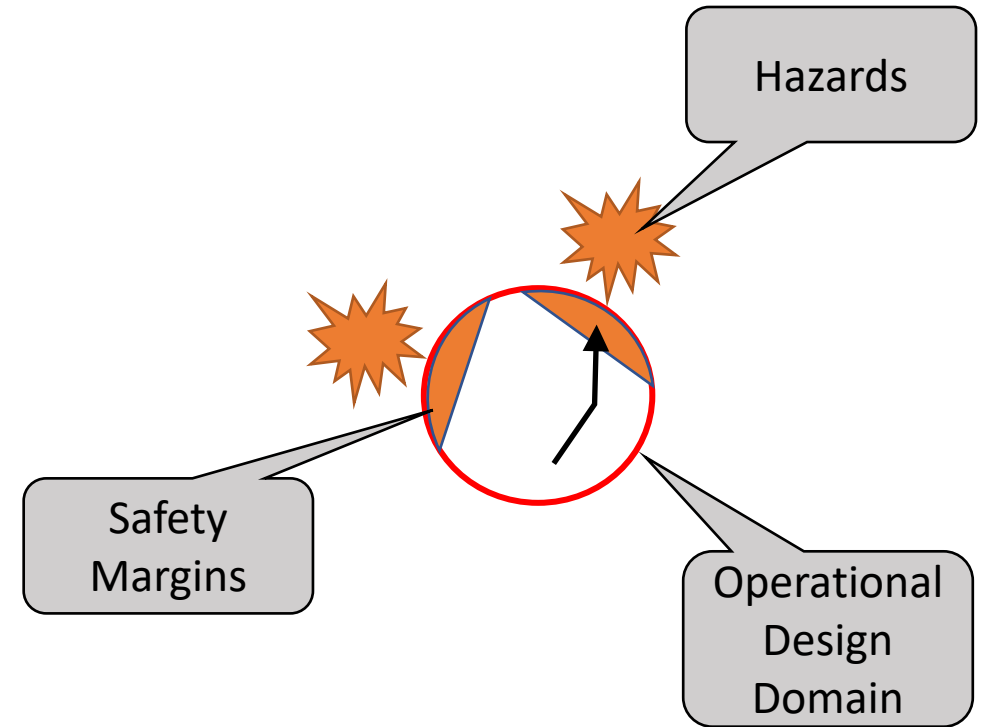
Source: onewheel.com





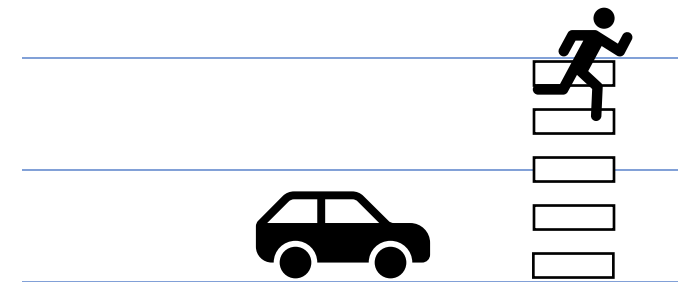
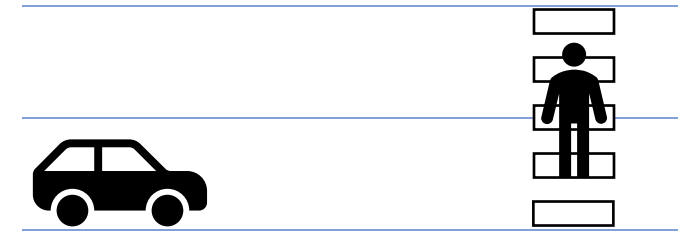
# Detecting Near Misses

- Case 1: Known Hazards
  - During the design process, we have defined hazardous states and introduced margins of safety around them
    - Come too near to another object
    - Extreme steering and braking
  - Design should include sensors to detect when we enter those safety margin regions



# Case 2: Counterfactual Near Misses

- Automatic Vehicle safety conditions
  - At least 2m separation between vehicle and pedestrians, cyclists, stationary obstacles
- Pedestrian sees car coming and jumps out of the way
- Car determines that it met the required 2m separation → “no problem”
- Counterfactual: There would have been a safety violation if the pedestrian had not taken evasive action
- Pearl’s Theory of Causality provides the formal basis for computing counterfactual near misses [Pearl, 2009; Pearl & MacKenzie, 2018]



# Automated Diagnosis and Repair

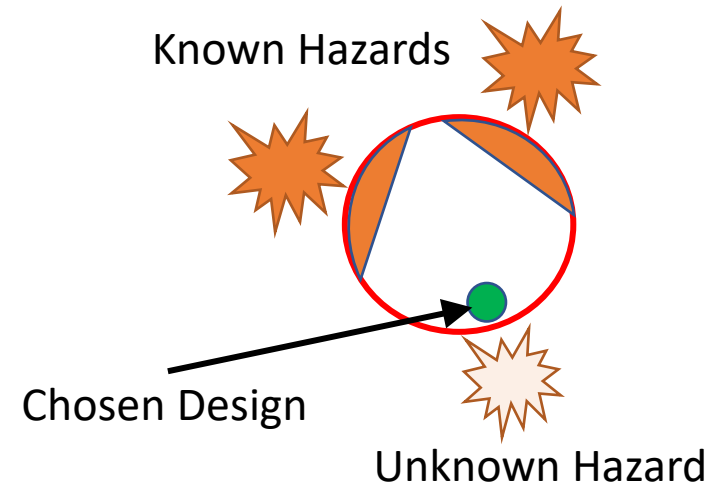
- Given a detected anomaly or hazard, what components contributed and how should they be modified?
- Diagnostic system requires
  - A causal model of the system including information flows
  - Reasoning capability to hypothesize potential contributing components
- Repairs can range from simple retraining of ML components to entire system redesign
  - What repairs can be safely applied by the AI system itself?
    - Adding a new hazard region into the path planner
    - Preparing training data to update the perceptual system and controllers

# Outline

- Part 1: Integrating ML into traditional safety engineering processes
  - Scenario-based data collection
  - Verification-based data collection
  - Risk quantification
- Part 2: ML in open worlds: Safety as Control
  - Detecting anomalies, near misses, and departures from the Operational Design Domain
  - Adaptation strategies
- Part 3: Safety as Continual Redesign
  - Design is never finished
  - Resilient systems are “Poised to adapt”

# Creating Resilient Systems

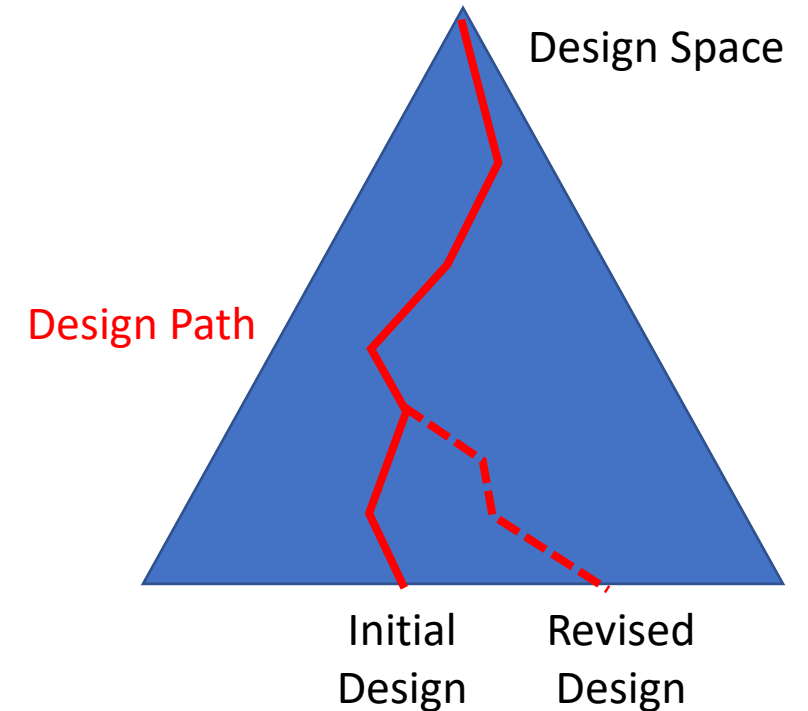
- Engineered systems are “robust yet fragile”
  - Robust to the known hazards
  - Vulnerable to novel failure modes
- Optimization for cost, weight, etc. results in designs near the edge of the feasible region
  - Highly Optimized Tolerances (HOT) theory.  
Carson & Doyle (2002)



# Creating Resilient Systems

- David Woods: A resilient system is one that is “poised to adapt”
  - [Woods, 2024a, 2024b]
- An AI perspective:
  - The entire design process should be regarded as one path through a design space
  - Adaptation requires following new paths through that space
  - Build AI tools for continual design

**The design space and design process should be “kept on standby” so that they can be resumed whenever adaptation is required**



# Summary

- ML and traditional safety engineering: Managing Known Hazards
  - High-fidelity simulation
  - Risk-driven generation of training data and test cases
  - Verification methods to ensure distribution-independent generalization
  - Importance sampling for risk estimation
- Safety as Control
  - Novel hazards as system disturbances
  - KPIs: Anomalies and Near Misses
  - AI methods for anomaly detection are mature
  - AI tools can help:
    - Anomaly detection and near-miss detection
    - Diagnosis and repair
- Safety as Resilience: “Poised to Adapt”
  - The design space and design process “kept on standby” so that they can be invoked whenever adaptation is required

# References

- Carlson, J. M., & Doyle, J. (2002). Complexity and robustness. *Proceedings of the National Academy of Sciences*, 99(Supplement 1), 2538–2545. <https://doi.org/10.1073/pnas.012582499>
- Deng, J., Dong, W., Socher, R., Li, L.-J., Li, K., & Li, F.-F. (2009). ImageNet: A large-scale hierarchical image database. In *2009 IEEE Conference on Computer Vision and Pattern Recognition*. IEEE. <https://doi.org/10.1109/CVPR.2009.5206848>
- Dietterich, T. G., & Guyer, A. (2022). The Familiarity Hypothesis: Explaining the Behavior of Deep Open Set Methods. *ArXiv*, 2203.02486(v1). <http://arxiv.org/abs/2203.02486>
- Kochenderfer, M., Katz, S. M., Corso, A. L., Moss, R. J. (forthcoming) *Algorithms for Validation*. MIT Press. Cambridge, MA. Draft available for comment from the authors. <https://algorithmsbook.com/validation/files/val.pdf>
- Leveson, N. G. (2011). *Engineering a Safer World: Systems Thinking Applied to Safety*. MIT Press. Cambridge, MA.
- Moss, R. J., Kochenderfer, M. J., Gariel, M., & Debois, A. (2023). Bayesian Safety Validation for Failure Probability Estimation of Black-Box Systems. *ArXiv*, 2305.02449(v2), 12–16.
- Pearl, J. (2009). *Causality*. Cambridge University Press.
- Pearl, J., & Mackenzie, D. (2018). *The Book of Why*. Basic Books.
- Ruff, L., Kauffmann, J. R., Vandermeulen, R. A., Montavon, G., Samek, W., Kloft, M., Dietterich, T. G., Mueller, K.-R. (2021). A Unifying Review of Deep and Shallow Anomaly Detection. *Proc. IEEE*, 109 (5): 756-795
- Singh, H., Kumar, P., Torr, P. H. S., & Dvijotham, K. (2021). Overcoming the Convex Barrier for Simplex Inputs. *Advances in Neural Information Processing Systems (NeurIPS 2021)*, 12.
- Verma, A. K., Ajit, S., Karanki, D. R. (2010) *Reliability and Safety Engineering, 2ed*, Springer.
- Wang, J., Pun, A., Tu, J., Manivasagam, S., Sadat, A., Casas, S., Ren, M., & Urtasun, R. (2023). AdvSim: Generating Safety-Critical Scenarios for Self-Driving Vehicles. *ArXiv*, 2101.06549(v4).
- Weick, K., Sutcliffe, K., & Obstfeld, D. (1999). Organizing for high reliability: Processes of collective mindfulness. In R. S. Sutton & B. M. Staw (Eds.), *Research in Organizational Behavior* (Vol. 1, pp. 81–123). Jai Press.
- Woods, D. D. (2024). Resolving the Command – Adapt Paradox: Guided Adaptability to Cope with Complexity. In J. Le Coze & B. Journé (Eds.), *Compliance and Initiative in the Production of Safety* (pp. 73–87). Springer Nature Switzerland. <https://doi.org/10.1007/978-3-031-45055-6>
- Woods, D. D. (2024). Limits of Automata — Then and Now: Challenges of Architecture , Brittleness , and Scale. *Journal of Cognitive Engineering and Decision Making*. <https://doi.org/10.1177/15553434241240203>