ReSonAte: A Runtime Risk Assessment Framework for Autonomous Systems

Charlie Hartsell, Shreyas Ramakrishna, Abhishek Dubey, Daniel Stojcsics, Nagabhushan Mahadevan, and Gabor Karsai

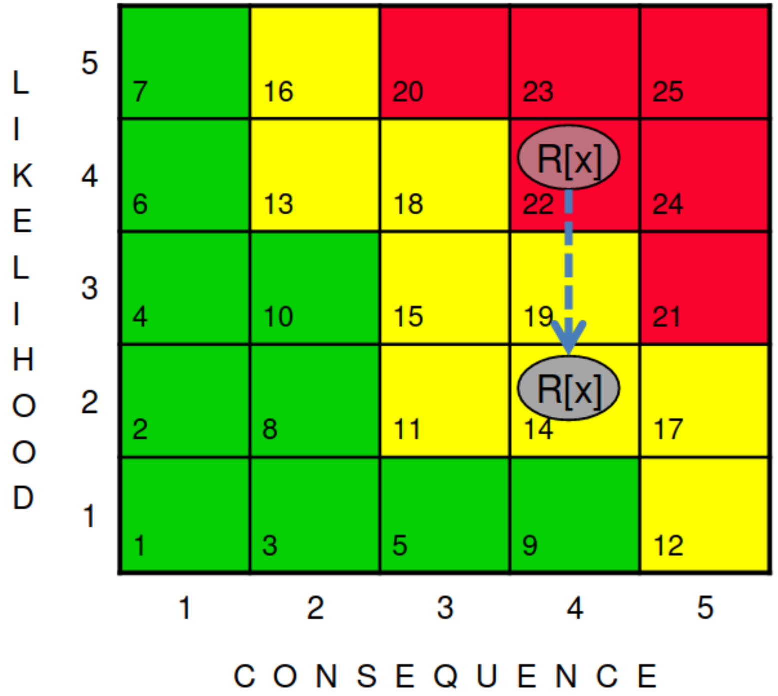
Vanderbilt University

Tel (615) 343-7472 Fax (615) 343-7440
1025 16th Avenue South|Nashville, TN 37212
www.isis.vanderbilt.edu

VANDERBILT UNIVERSITY

Risk Reduction Approach

- *"Safety Risk Management (SRM) has become a core component of safety assurance case in UAS domain. An acceptable safety case must provide information that outlines all hazards, risks associated and procedures to mitigate the risk. "* - (FAA(8900.1 CHG 625))
- Risk defined as product of **severity** and **probability** of consequences. Popularly represented as Risk matrix
- Hazard identification, probability estimation, risk reduction
- Conventional risk management use static, design-time risk estimates
  - Covers range of expected operating environments
- Static estimate insufficient for highly autonomous systems
  - Constantly dealing with potentially hazardous events in dynamic environment
  - No human operator for handling unexpected events



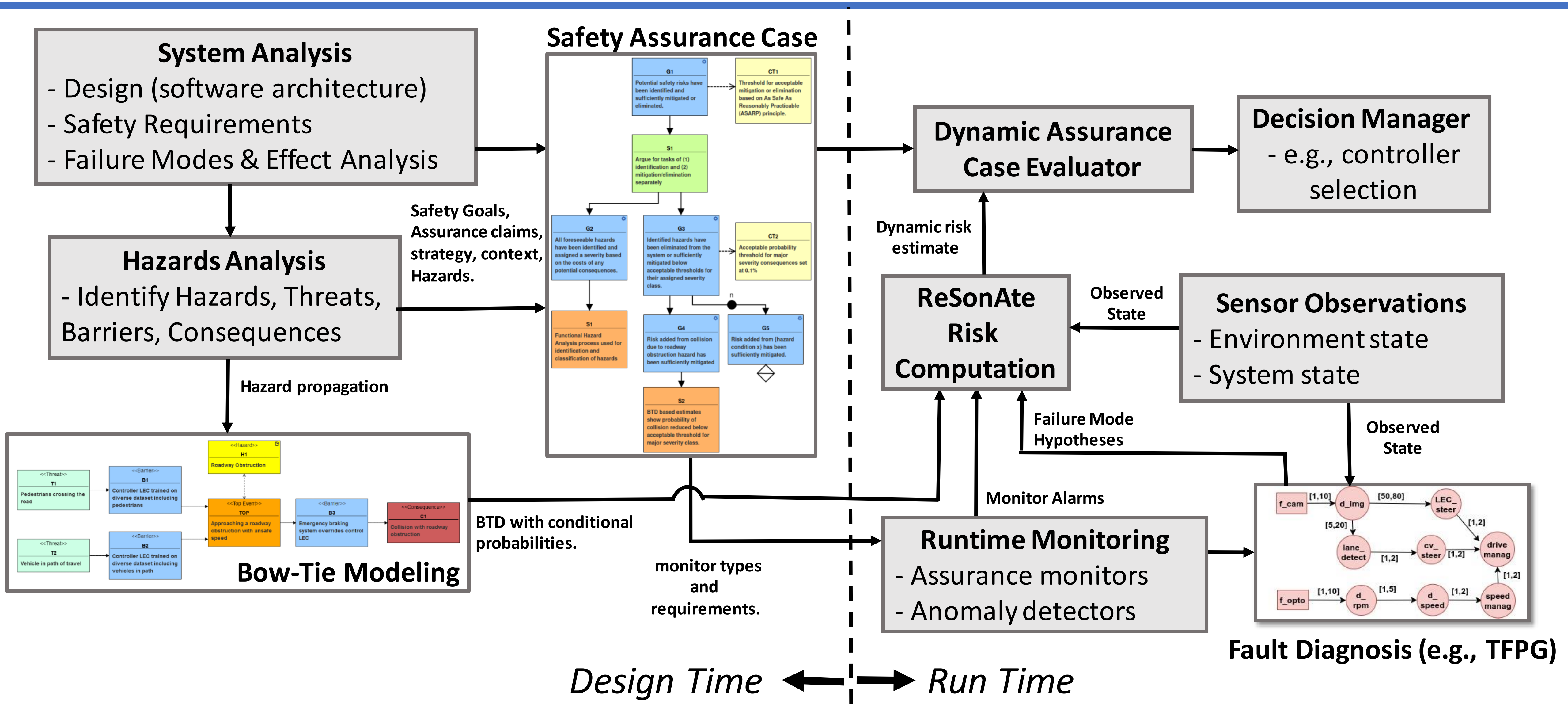
Risk matrix reproduced from NASA's "Risk Review Template"

ReSonAte Framework and Contributions

- We introduce ReSonAte, a framework that calculates *probability* of consequences **dynamically**
  - Probabilities change based on the state of the system and environment
  - Likelihood of events, effectiveness of control strategies
  - *Severity* remains static for now

- **Specific Contributions** of ReSonAte are:
  1. We use an extended Bow-Tie Diagram (BTD) to describe possible hazard propagation paths and make design-time measurements of the conditional relationships between hazard rates and the state of the system and environment
  2. We use the conditional relationships at run-time along with state observations from multiple sources to dynamically estimate system risk
  3. We implement ReSonAte for an Autonomous Vehicle (AV) example in the CARLA simulator and an Unmanned Underwater Vehicle (UUV) example in the Gazebo/UUV-sim simulator

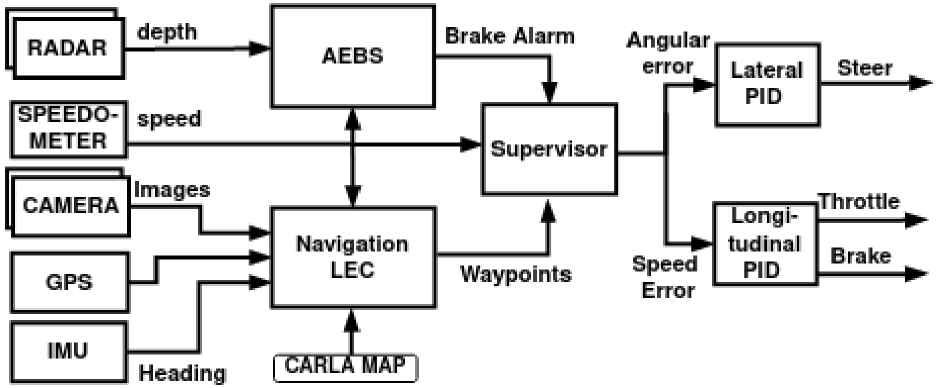
System Risk Management using ReSonAte



Target Platform

- ReSonAte is intended for Autonomous Systems.
- These systems often utilize Learning Enabled Components (LECs)
- LEC – Component where behavior is learned from data instead of explicitly defined (e.g. Machine Learning)
- Out-of-distribution problem - LECs are shown to predict erroneously on data that is not in the training data
- Assurance Monitor - Component for OOD data detection [1]

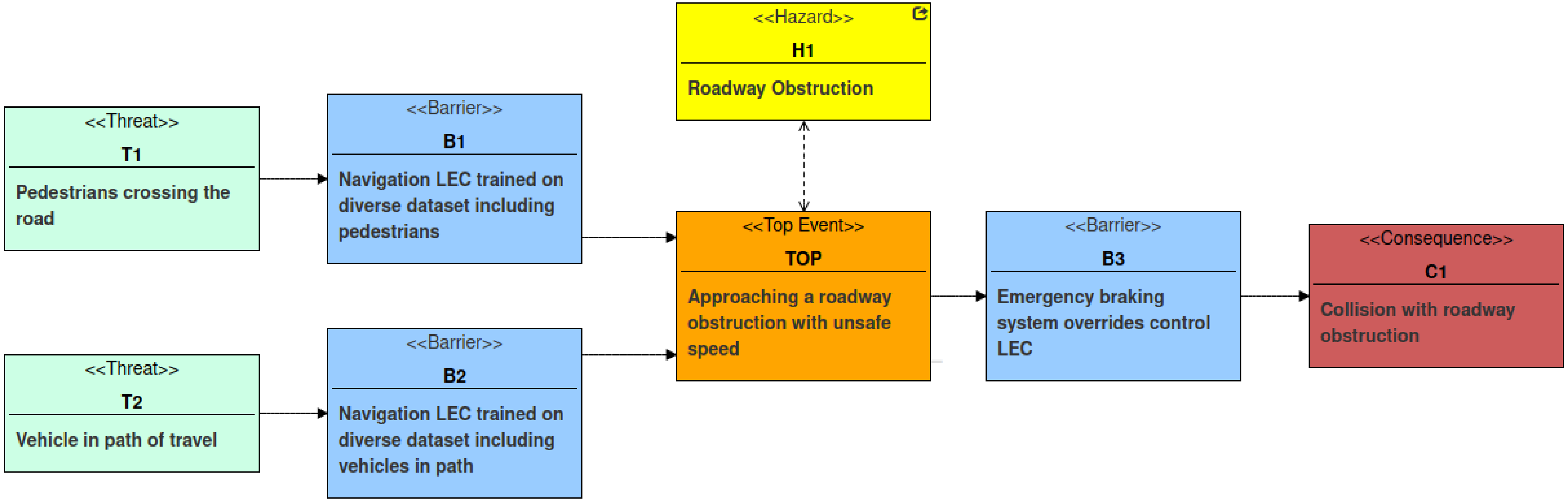
Running Example: An Autonomous Vehicle (AV) operating in CARLA simulation under varying weather and sensor faults.



System block diagram of AV

[1] Sundar, V. K., Ramakrishna, S., Rahiminasab, Z., Easwaran, A., & Dubey, A. (2020, May). Out-of-Distribution Detection in Multi-Label Datasets using Latent Space of β-VAE. In *2020 IEEE Security and Privacy Workshops (SPW)* (pp. 250-255). IEEE.

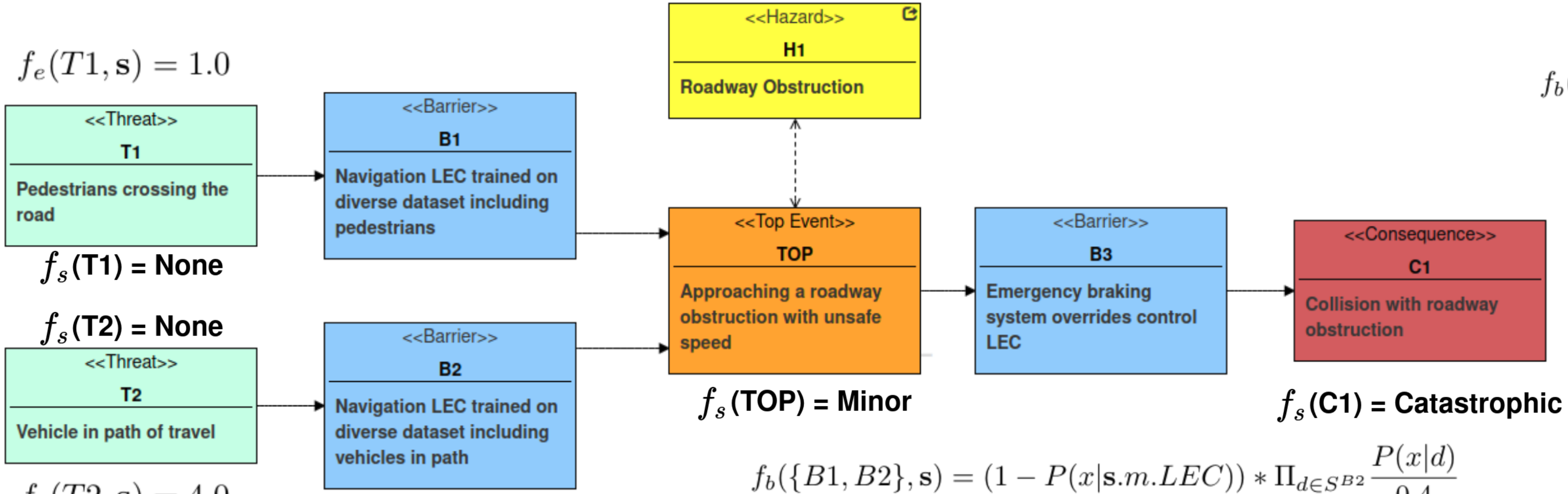
Bow-Tie Diagram



- **BowTie Diagram (BTD)** - graphical language for modeling hazards present in a system, how certain events may escalate the risk from these hazards, and the mitigation strategies for preventing escalation.
    - **Hazard –** Condition/event that can lead to harm
    - **Events –** Threat, Top Event, Consequence
    - **Barrier –** Hazard control strategies
- Each barrier reduces the probability of further hazard propagation
- BTDs are proactively used for computing design-time risk

BTD Conditional Probability Estimation

$$f_e(T1, \mathbf{s}) = 1.0$$



$$f_e(T2, \mathbf{s}) = 4.0$$

$$f_b(B3, \mathbf{s}) = \begin{cases} 0.0 & radar\_failure \in \mathbf{s}.F \\ 0.833 & \mathbf{s}.e.precip \leq 0.2 \\ 0.700 & 0.2 < \mathbf{s}.e.precip \leq 0.4 \\ 0.714 & 0.4 < \mathbf{s}.e.precip \leq 0.6 \\ 0.042 & 0.6 < \mathbf{s}.e.precip \leq 0.8 \\ 0.056 & \mathbf{s}.e.precip > 0.8 \end{cases}$$

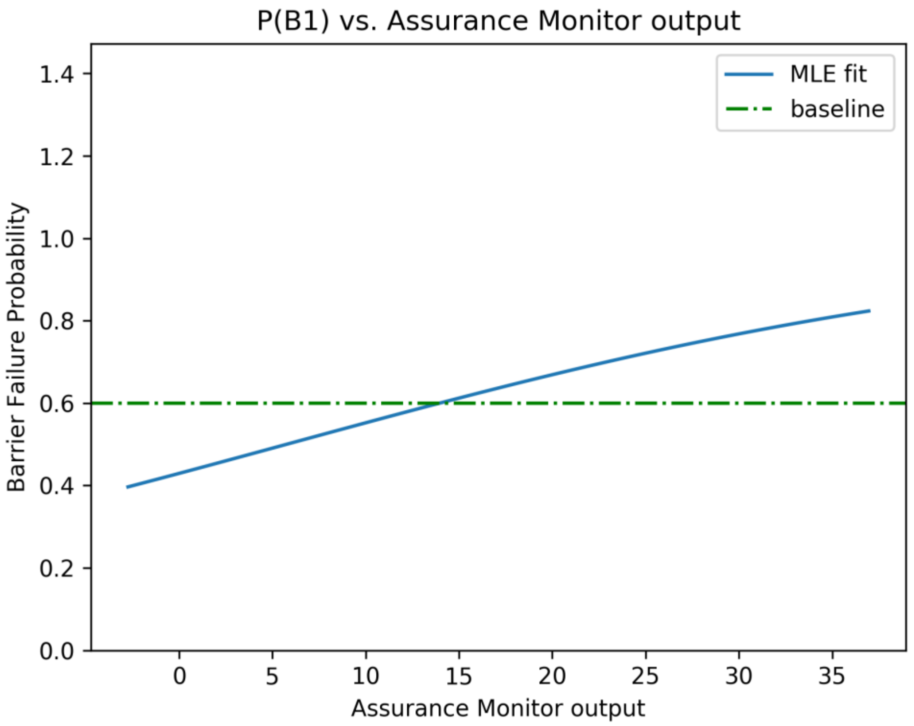
$$f_b(\{B1, B2\}, \mathbf{s}) = (1 - P(x|\mathbf{s}.m.LEC)) * \Pi_{d \in S^{B2}} \frac{P(x|d)}{0.4}$$

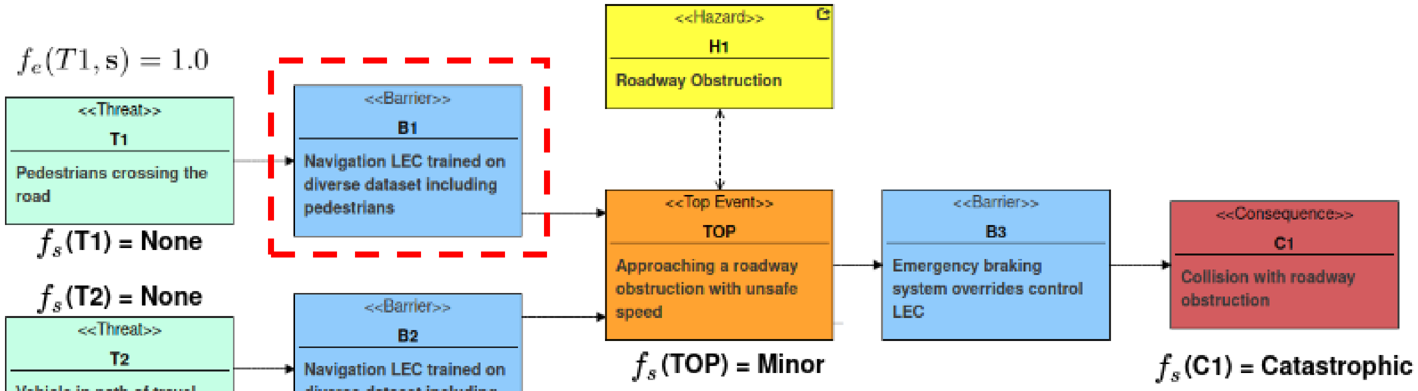
$$P(x|\mathbf{s}.m.LEC) = (1 + e^{-0.049*(\mathbf{s}.m.LEC - 5.754)})^{-1}$$

d	$P(x\|d)$	$P(x\|\bar{d})$
Center Cam Blur	0.351	0.417
Left Cam Blur	0.378	0.409
Right Cam Blur	0.270	0.443
Center Cam Occlude	0.342	0.421
Left Cam Occlude	0.395	0.404
Right Cam Occlude	0.395	0.404

- f_e(T1/T2, **s**) - expected frequency of threats given state **s**
- f_b(B3, **s**) - probability of B3 preventing propagation from TOP to Consequence given state **s**
- f_s(event, **s**) - assigned severity level for each event
- P(x|s.m.LEC) - probability of B3 conditional on output of the LEC assurance monitor

BTD Conditional Probability Example



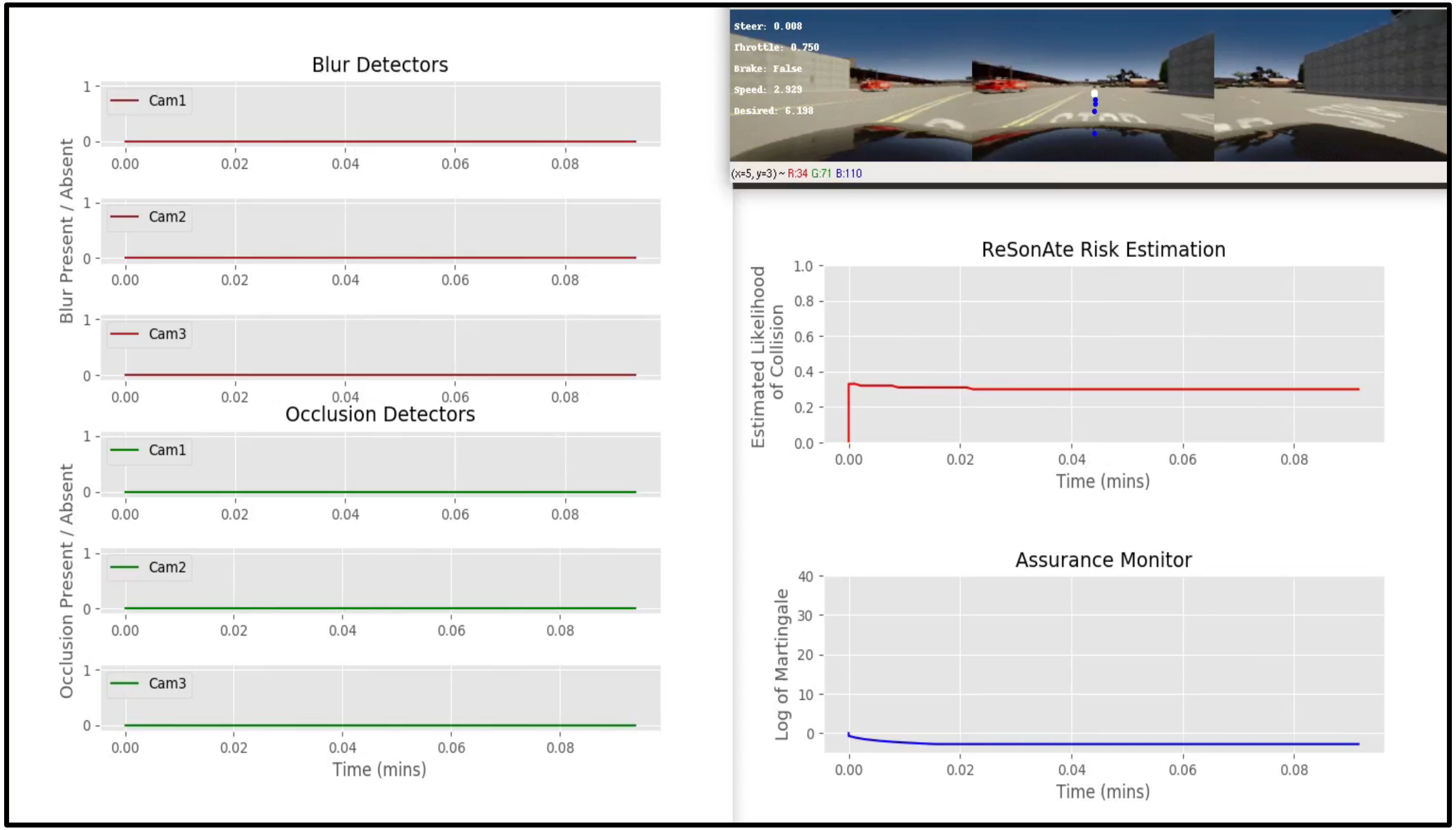


$$f_b(x = (B1, B2), s) = (1 - P(x|s.m.LEC)) * \Pi_{d \in S^{B2}} \frac{P(x|d)}{0.4}$$

$$P(x|s.m.LEC) = (1 + e^{-0.049*(s.m.LEC - 5.754)})^{-1}$$

- Probabilities estimated from simulation data
- Fit conditional Poisson distribution to observed data using Maximum Likelihood Estimation
- *Current*: Specific scenarios allow each barrier to be treated in isolation
- *Future*: Continually improve estimates from operational data

Camera Fault – Occluded Image



Operating Environment

- Weather (cloud= 0.0, precipitation = 0.0).
- Traffic = 120 (high)
- Town type = 3

Sensor Faults:

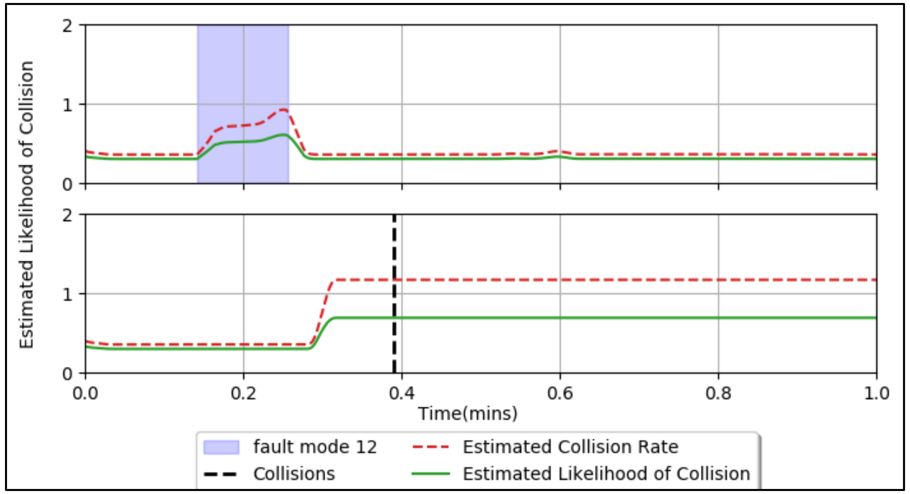
- Occluded Camera Images.

Runtime Monitors:

- **Blur Detectors** – detect image blurriness. (1 – present, 0 - Absent)
- **Occlusion Detectors** – detect image occlusions. (1 – present, 0 - Absent)
- **Assurance Monitor** – detect shifts in operating environment.
  - Martingale increases with adverse changes in environment (e.g., brightness)

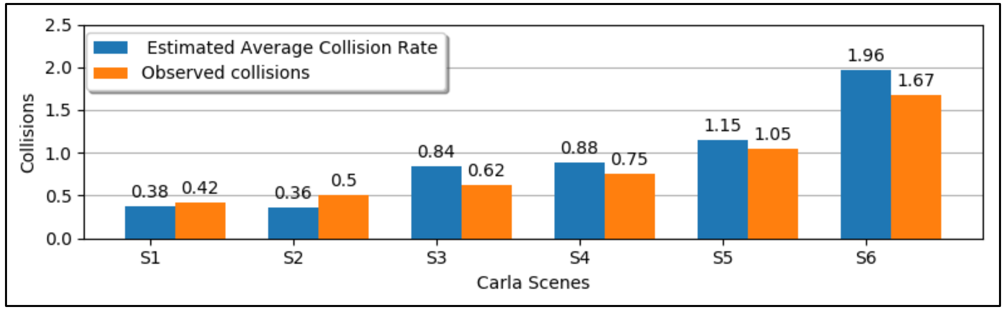
Validation with CARLA Simulation

- We ran 608 separate simulations from 46 CARLA scenes with different weather patterns.
- Collision is probabilistic event – modeled by Poisson distribution.
- Dynamically estimate the hazard rate $\boldsymbol{\lambda}$, then compute probability of collision in next t time units as 1 - e^(-$\boldsymbol{\lambda} t$)



ReSonAte estimated collision rate and likelihood of collision for:

- (Top) nominal scene intermittent camera fault (occlusion)
- (Bottom) nominal scene with adverse environment (excessive brightness).



Estimated average collision rates vs. Actual collisions across 6 validation scenes. Average of 20 simulation runs for each scene.

Scene	Cloud	Precipitation	Deposit	Faults
S1	0.0	0.0	0.0	0
S2	5.0	5.0	5.0	0
S3	0.0	0.0	0.0	1 or 2
S4	25.0	25.0	25.0	1 or 2
S5	50.0	70.0	30.0	1 or 2
S6	90.0	75.0	80.0	1 or 2

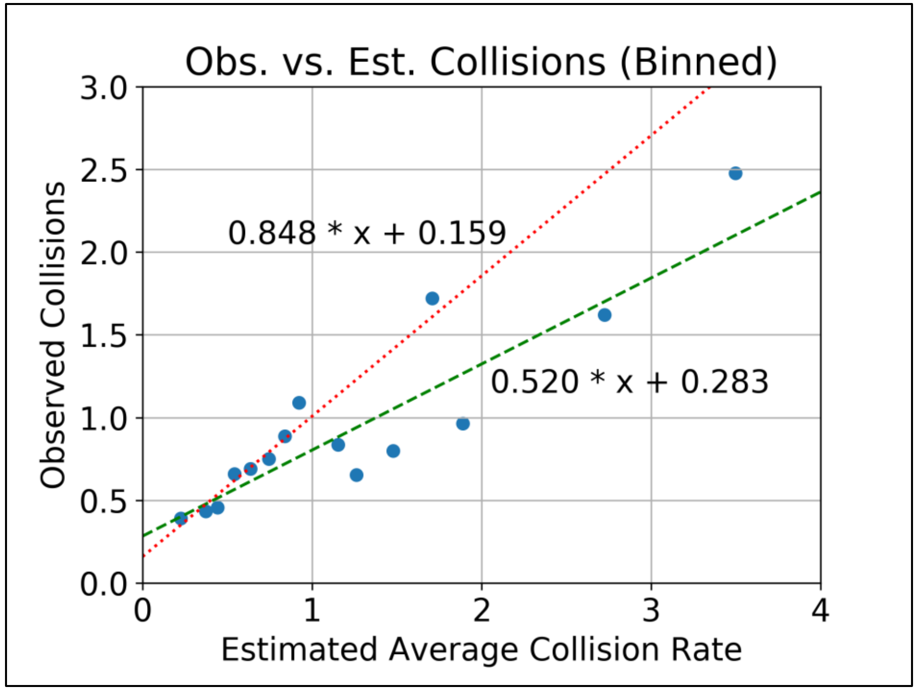
Scene descriptions for S1-S6

Validation with CARLA Simulation

- Likelihood analysis shows dynamic estimates outperforms best static estimate
  - Log likelihood of observed outcomes: <u>–710 vs –741</u>
- Strong correlation between the estimated collision rates and observed collisions.
  - Spearman rank correlation: <u>0.943</u>

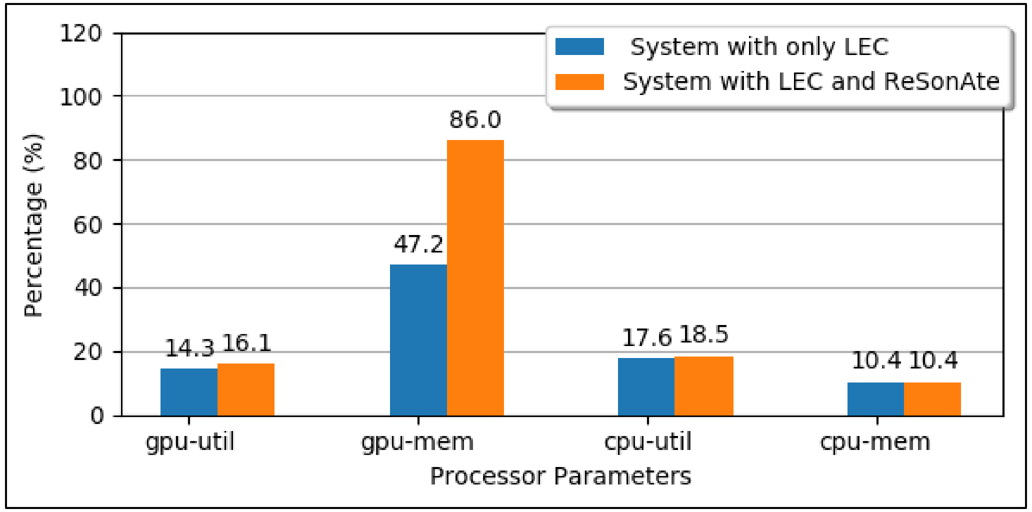
- The green line is a least square fit on the binned collision rate estimates of all the 608 simulation runs
  - Our estimates show an overestimation of the collision rates when the hazard rate is > 1
- The red line is a least square fit on the binned collision estimates when the hazard rate is low (0, 1)

- Considering the risk score as a binary classifier with a nominal risk threshold (Δ=0.4).
- ReSonAte had an F1-score of 76% in identifying scenes with a risk greater than Δ.

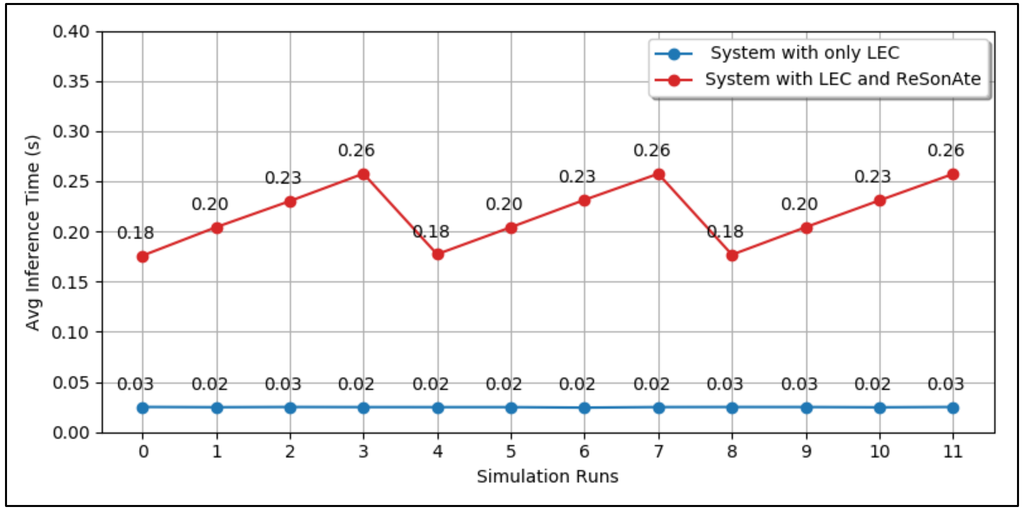


Overhead of the ReSonAte Framework

- Host machine: 32 AMD Ryzen Threadripper 1950X 16-Core Processor, 4 NVIDIA Titan Xp GPU's, 128 GiB memory and Ubuntu 18.04. (1 GPU used for experiments, ReSonAte risk calculations single-threaded)



Processor CPU and GPU Utilization of System with only LEC and System with LEC and ReSonAte

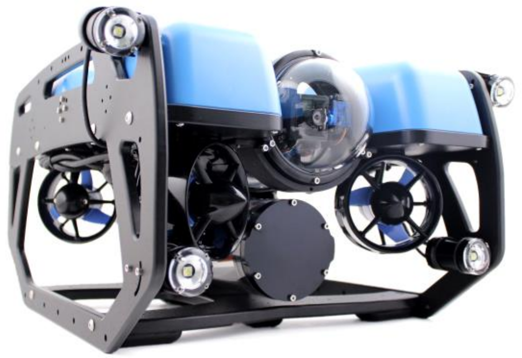


Inference times of System with only LEC and System with LEC and ReSonAte

- Increased GPU usage due to Assurance Monitor (deep learning based) and very minimally by the other Anomaly detectors (OpenCV based)
- Risk calculations take only 0.3 milliseconds
- ReSonAte increases the inference times, largely due to addition of runtime monitors.

Example2 - BlueRoV2

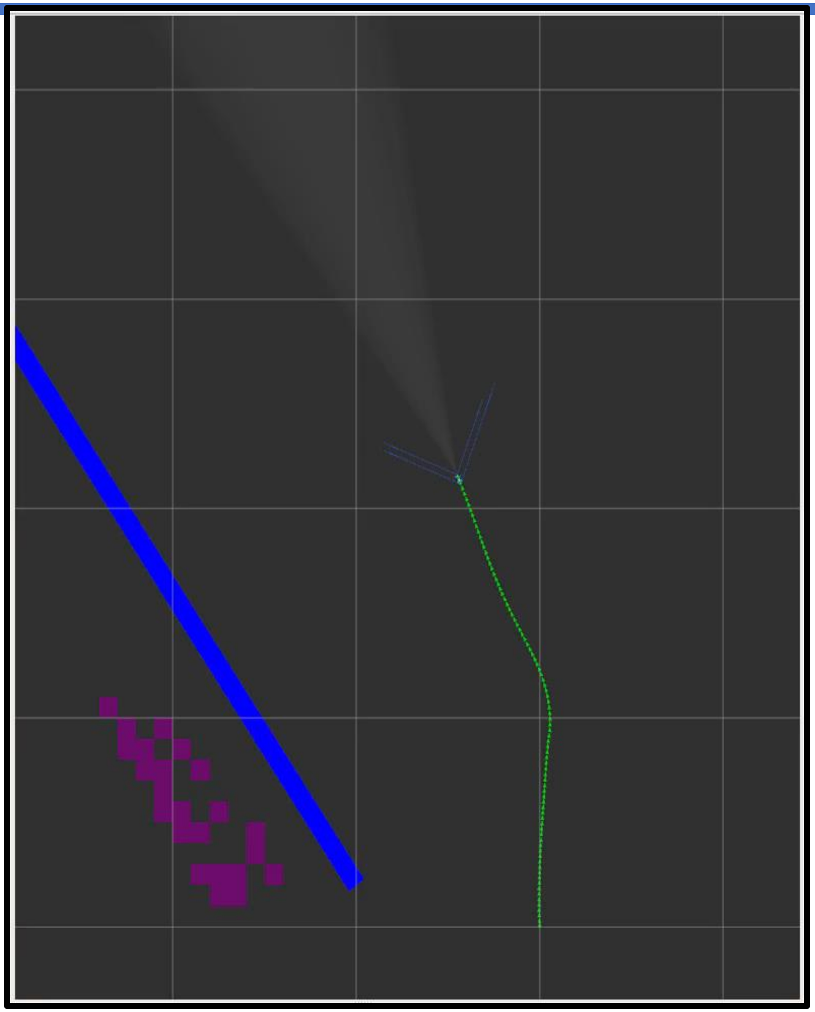
BlueROV2[1]



Autonomous operations:
- Pipe tracking
- Waypoint following
- Return to home
- Loiter
- Emergency surfacing

Potential Hazards:
- Static obstacle collision
- Dynamic obstacle collision
- Pipe loss

Example: UUV performing pipe tracking and obstacle avoidance in Degraded Conditions
- **Degradation** – One or more thruster degradation
- **Obstacles** – Static obstacles of varying size and dynamic obstacles (e.g. nearby shipping traffic)
- **Contingency Plan** – Thruster reallocation, return to home, and loiter mode.
- Demo available at https://github.com/scope-lab-vu/resonate



BlueRov2 Operation in nominal mode

1. https://bluerobotics.com/store/rov/bluerov2/

Conclusion and Future Work

- ReSonAte provides a technique for dynamically estimating risk posed by identified hazard conditions
- Probability of threats and effectiveness of barriers subject to change based on system state
- ReSonAte's risk calculations require minimal computational resources and time, making it suitable for resource-constrained and real-time cyber-physical systems
- Future Work:
  - Further application to UUV and hardware testbeds such as F1/10 cars
  - Inclusion of runtime verification
  - Addition of uncertainty in conditional probability calculations
  - Prediction of future risk based on expected future states
  - Recovery/Contingency actions
  - Continually improve conditional probabilities at runtime