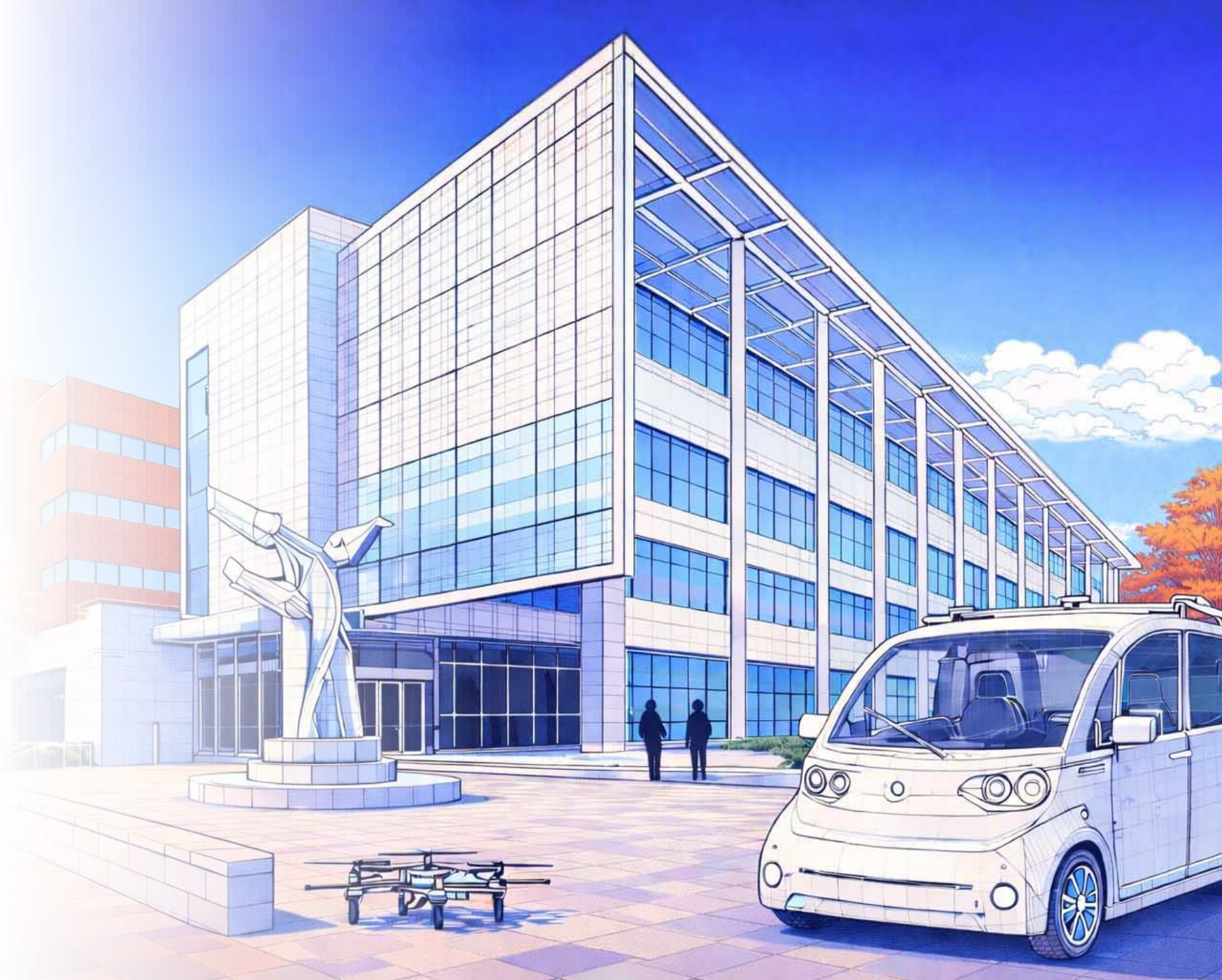
Outline

Motivation

Administrivia

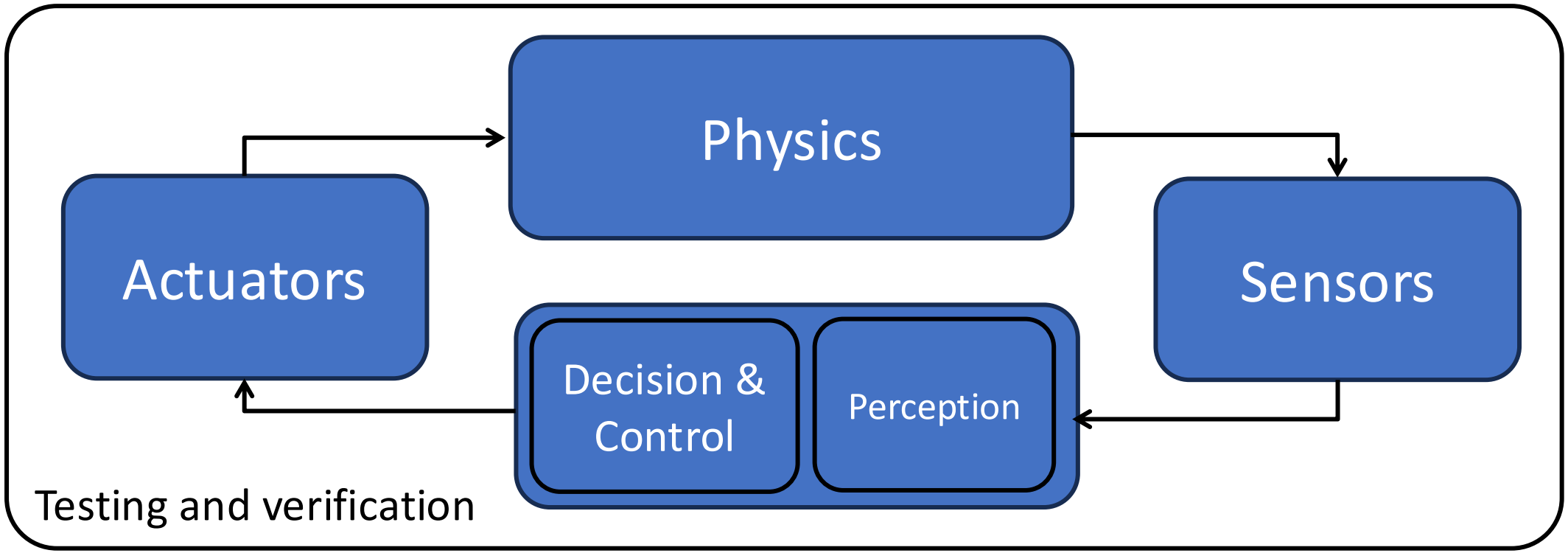
Introduction to Safety



Principles of Safe Autonomy ECE 484

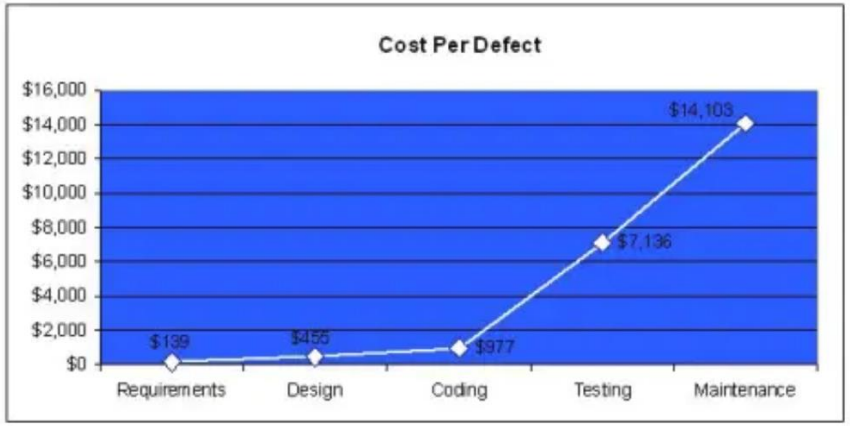
Autonomy is a frontier where perception, learning, control, and verification meet the real world and ECE484 is where you begin to shape it.

ECE484 develops the foundational principles behind autonomy



Cost of unreliability in autonomous systems

- Therac-25 radiation therapy machine delivered overdoses because of software bug which resulted in 6 fatalities.
- Elaine Herzberg was killed by self-driving Uber prototype in Tempe, Arizona in March 2018.
- Data conversion error caused the **$500M Ariane 5 rocket** to veer off course and explode shortly after launch.
- GM's Cruise autonomous vehicle unit shut down its San Francisco robotaxi fleet after crashes in 2023.
- Cost of defects grow exponentially with time of discovery



Capers Jones, Software Assessments, Benchmarks, and Best Practices, Addison-Wesley, 2000

Limitations of testing for reliability and safety

Testing is an important and essential method for checking correctness of systems, but

"Testing can be used to show the presence of bugs, but never to show their absence!''

--- Edsger W. Dijkstra

Amount of testing required for autonomous systems can be prohibitive

- Probability of a fatality caused by an **accident per one hour of human driving** is ~ 10^{-6}
- Assume that for AVs this has to be 10^{-9}
- Data required to guarantee a probability of 10^{-9} fatality per hour of driving is proportional to its inverse, **10^9 hours, 30 billion miles**
- Multi-agent, open system, with human interactions => cannot be simulated offline to generate data
- Any change is software means tests must be rerun

On a Formal Model of Safe and Scalable Self-driving Cars by *Shai Shalev-Shwartz, Shaked Shammah, Amnon Shashua, 2017 (Responsibility Sensitive Safety)*

Mathematically Checking truthfulness of statements

The ultimate standard for truth: A theorem with a proof

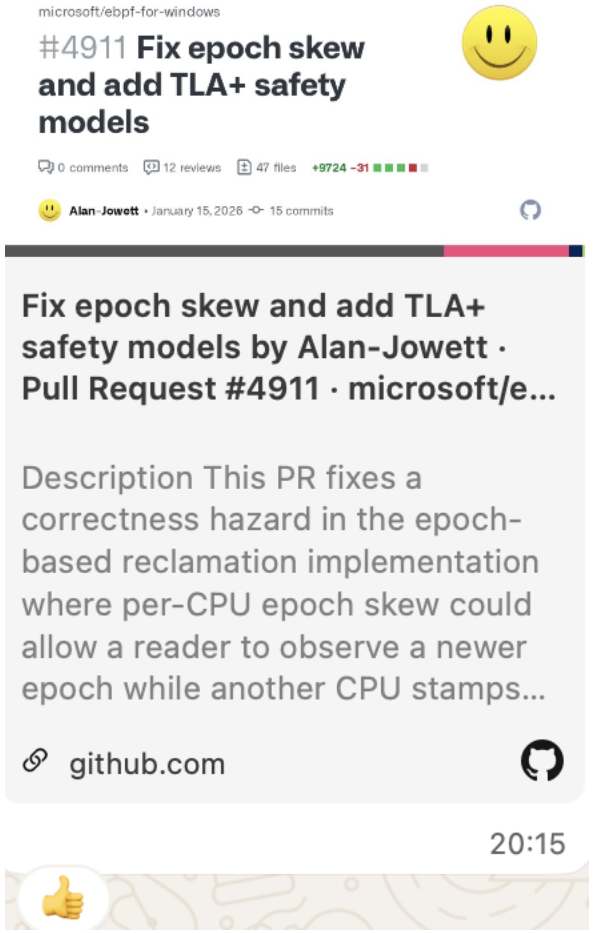
Formal verification: *The science of proving or disproving truth of statements asserting correctness of systems*

Proofs are used at scale in Amazon, Meta, Microsoft, NASA, …

"In 2017 alone the security team used deductive theorem provers or model checking tools to reason about cryptographic protocols, hypervisors, boot-loaders, firmware, garbage collectors, and network designs." Byron Cook, Amazon

In MP0 you will use a tool called Verse to verify collision avoidance of UAVs

To prove theorems about system safety, first we need precise assumptions about the its behavior and environment --- this is a model



Byron Cook at FLoC 2018
https://www.youtube.com/watch?v=JfjLKBO27nw

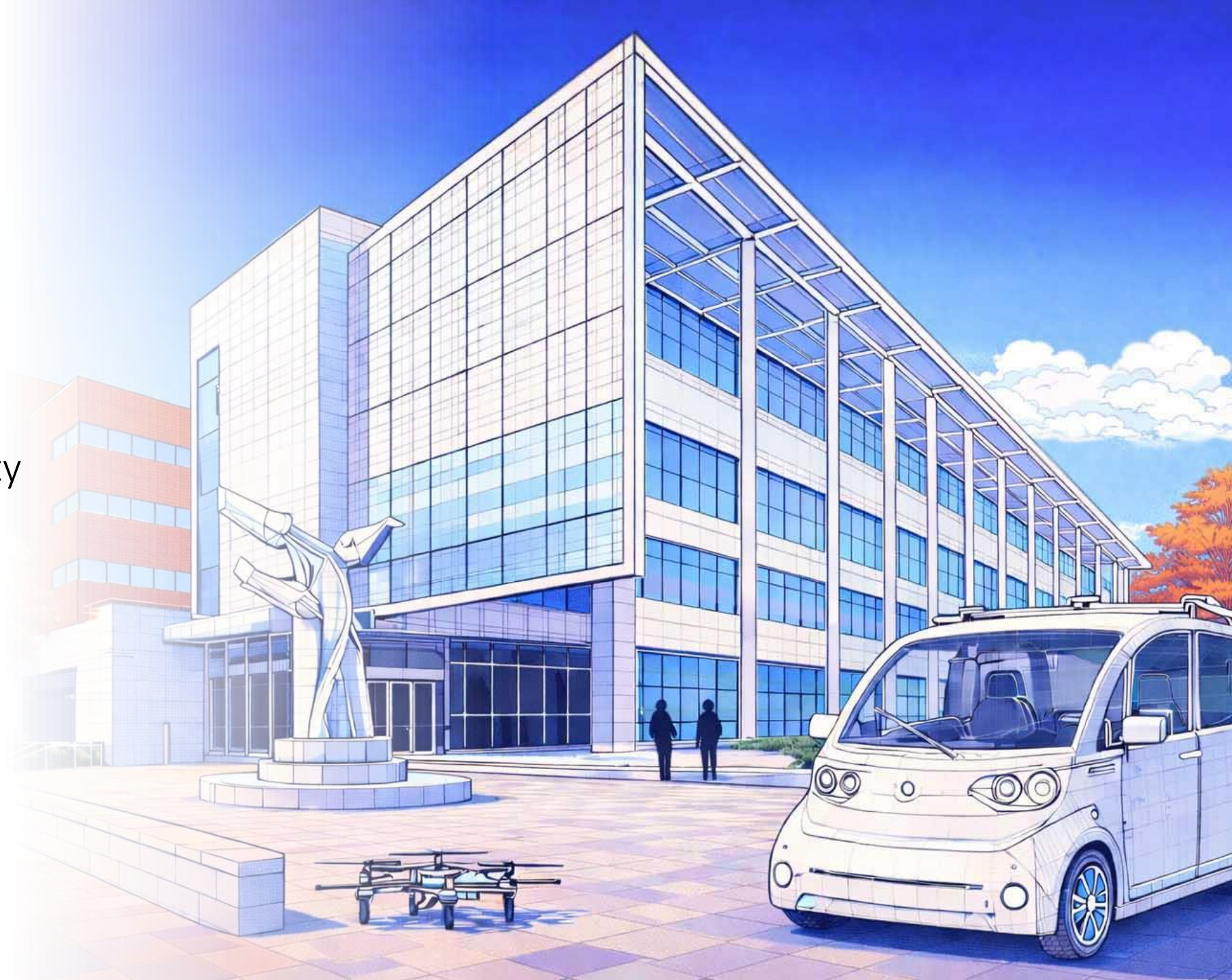
Outline

Motivation

Administrivia

Introduction to Safety

- Models
- Requirements
- Proofs



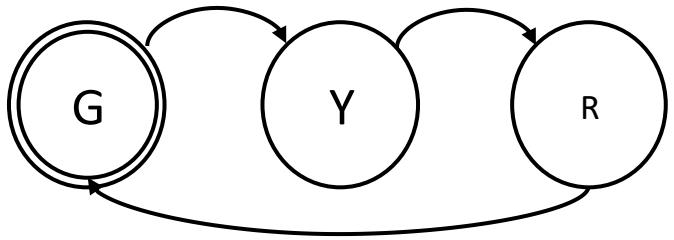
Automata or state machine models

An automaton A is defined by a triple $\langle Q, Q_0, D \rangle$, where

- Q is a set of states
- $Q_0 \subseteq Q$ is a set of initial states
- $D \subseteq Q \times Q$ is a set of transitions

An execution of A is a finite or infinite sequence $q_0, q_1, \ldots$ such that $q_0 \in Q_0$ and $(q_i, q_{i+1}) \in D$

Example: Traffic light automaton



- $Q = \{G, Y, R\}\ Q_0 = \{G\}$
- $D = \{(G, Y), (Y, R), (R, G)\}$

Execution of traffic light $G, Y, R, G, Y, R \ldots$ infinite even though finite state

Infinite State Automata and Testing

Example 2

- $Q = \mathbb{N}\ Q_0 = \{n_0\}$
- $\left(n, \frac{n}{2}\right) \in D$ if n is even $(n, 3n+1) \in D$ if n is odd

Deterministic automaton because each state $q \in Q$ has a unique next state

Deterministic automata have a single execution

Execution from $n_0 = 6$ is 6, 3, 10, 5, 16, 8, 4, 2, 1, 4, 2, 1 …

Execution from $n_0 =$ 7, 22, 11, 34, 17, 52, 26, 13, 40, 20, 10, 5, 16, 8, 4, 2, 1, . . .

Testing question. For any n_0 does the execution end in the 4-2-1 loop?

This is a well-known open problem in mathematics called the Collatz conjecture

Testing and verification can be hard for simple deterministic automata

Requirements and Counter-examples

Requirements define what the system must and must not do

Example: “Car stays within speed limit”

Autonomous car: “Ego should not collide with lead car”

Collatz: “Every number eventually ends in the 4-2-1 cycle”

A requirement defines a set ***R*** of allowed executions

An execution α that is not in the set ***R*** is a *counter-example*

$R_{\text{eventually-1}} = \{\alpha \mid \exists k\ \alpha_k = 1\}$

An automaton A satisfies a requirement R if ***all*** executions of A satisfies R

Whether the Collatz automaton satisfies the requirement $R_{\text{eventually-1}}$ for all initial conditions remains an open problem, although no counter-example has been found up to 2^{70}

This is an example of a verification problem

Verification problem

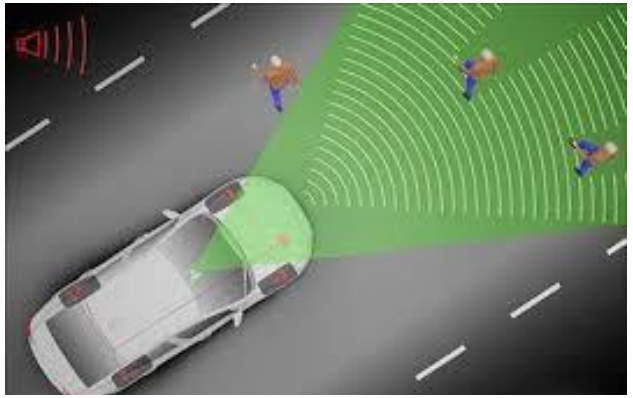
Verification problem: Given an automaton A and a requirement R, check whether all executions of A satisfy R or find a counter-example

Testing or checking individual executions can help find counter-examples but cannot show that there is no counter-example

Verification can be hard because

- |Q| is finite but large and testing may require visiting all the states (e.g., Collatz)
- |Q| is small but the number of executions is very large
- |Q| may be infinite and D may be nondeterministic --- typical in autonomy

Example: Automatic Emergency Braking (AEB)



Car must brake to maintain safe gap with lead vehicle/pedestrian

There is no standard for checking correctness of AEB

Future: Every code commit in github from an AEB engineer, **proves a theorem** establishing A satisfies R_{gap}

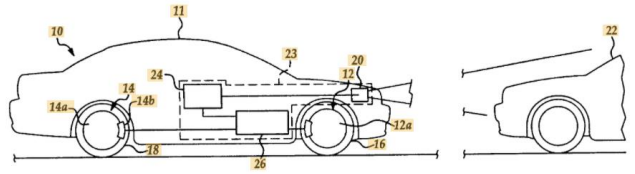


Figure 1

www.google.com › patents

US20110168504A1 - Emergency braking system - Google ...

Jump to **Patent** citations (18) - US4053026A * 1975-12-09 1977-10-11 Nissan Motor Co., Ltd. Logic circuit for an **automatic braking** system for a motor ...

www.google.com › patents

US5170858A - Automatic braking apparatus with ultrasonic ...

An automatic braking apparatus includes: an ultrasonic wave emitter provided in a ... Info: **Patent** citations (13); Cited by (7); Legal events; Similar documents; Priority and ... US6523912B1 2003-02-25 **Autonomous emergency braking** system.

www.google.com › patents

DE102004030994A1 - Brake assistant for motor vehicles ...

B60T7/22 **Brake**-action initiating means for **automatic** initiation; for initiation not ... Info: **Patent** citations (3); Cited by (9); Legal events; Similar documents ... data from the environment sensor and then **automatically** initiates **emergency braking**.

www.google.com.pg › patents

Braking control system for vehicle - Google Patents

An **automatic emergency braking** system for a vehicle includes a forward viewing camera and a control. At least in part responsive to processing of captured ...

www.automotiveworld.com › news-releases › toyota-ip...

Toyota IP Solutions and IUPUI issue first commercial license ...

Jul 22, 2020 - ... and validation of automotive **automatic emergency braking** (AEB) ... and Director of **Patent** Licensing for Toyota Motor North America. "We are ...

insurancenewsnet.com › oarticle › patent-application-tit...

Patent Application Titled "Multiple-Stage Collision Avoidance ...

Apr 3, 2019 - No assignee for this **patent** application has been made. ... **Automatic emergency braking** systems will similarly, also, soon be required for tractor ...

Automaton model of AEB

Automaton $A = \langle Q, Q_0, D \rangle$

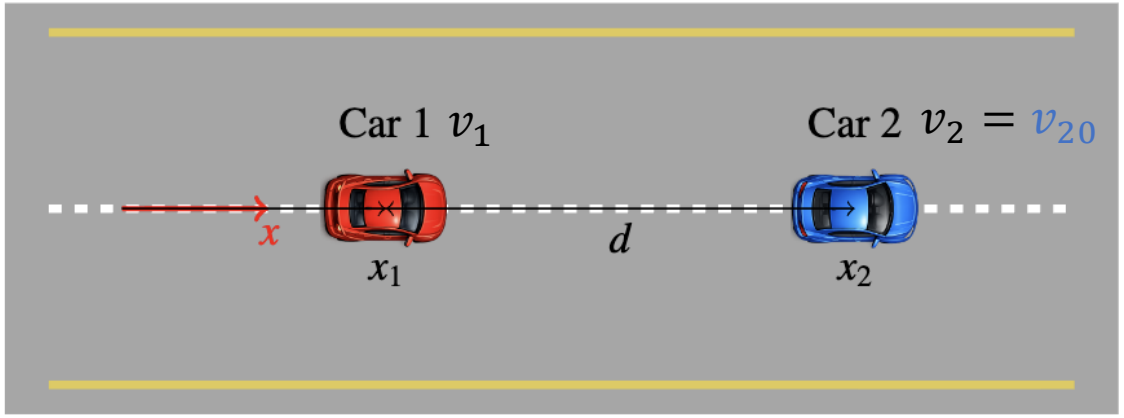
- Q: $[x_1, x_2, v_1] \in \mathbb{R}^3$
- $Q_0 = \{[x_1 = x_{10}, x_2 = x_{20}, v_1 = v_{10}]\}$
- $D \subseteq Q \times Q$ written as a program

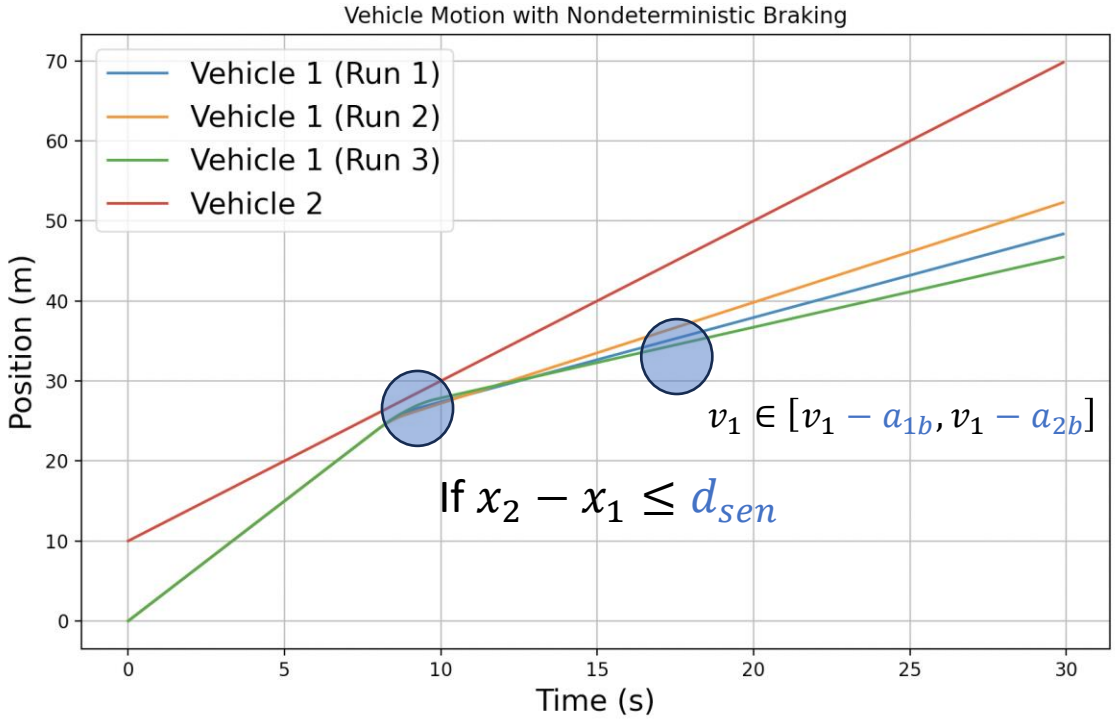
  If $x_2 - x_1 \leq d_{sen}$

  $v_1 \in [v_1 - a_{1b}, v_1 - a_{2b}]$

  $x_2 = x_2 + v_2$

  $x_1 = x_1 + v_1$





Automaton model of AEB

Automaton $A = \langle Q, Q_0, D \rangle$

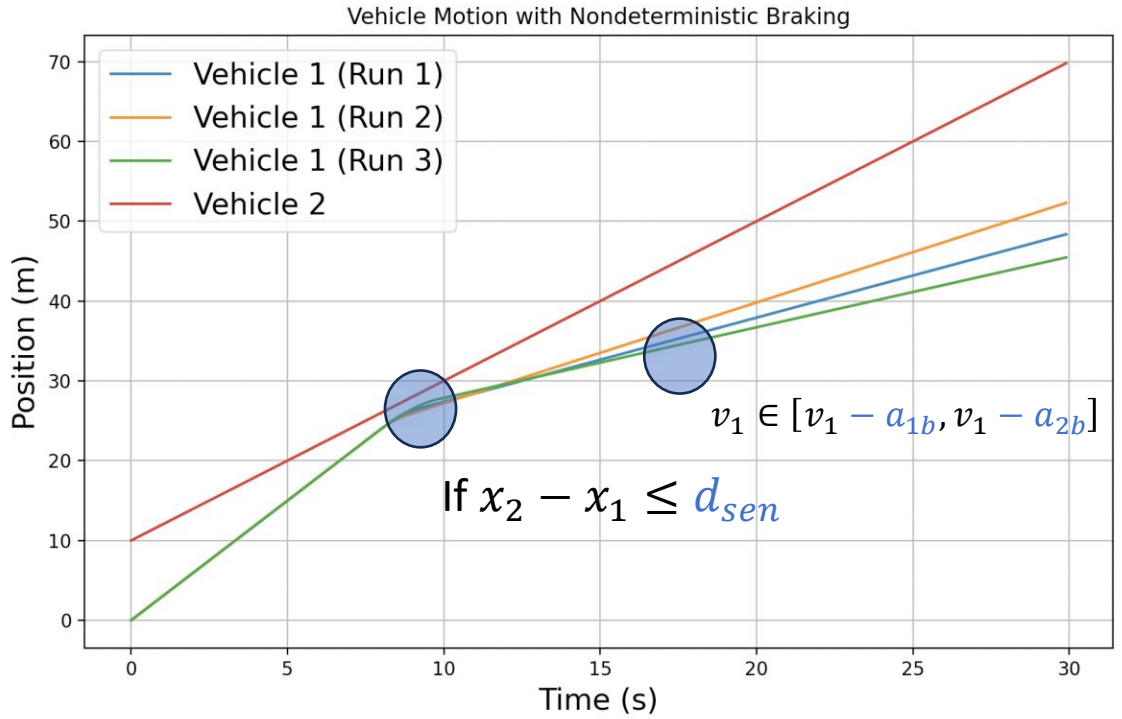
- $Q: \mathbb{R}^3; \boldsymbol{q} \in Q\ \boldsymbol{q}.x_1, \boldsymbol{q}.x_2 \in \mathbb{R}$
- $Q_0 = \{\boldsymbol{q} \mid [\boldsymbol{q}.x_1 = x_{10}, \boldsymbol{q}.x_2 = x_{20}, \boldsymbol{q}.v_1 = v_{10}]\}$
- $(\boldsymbol{q}, \boldsymbol{q}') \in D$ iff

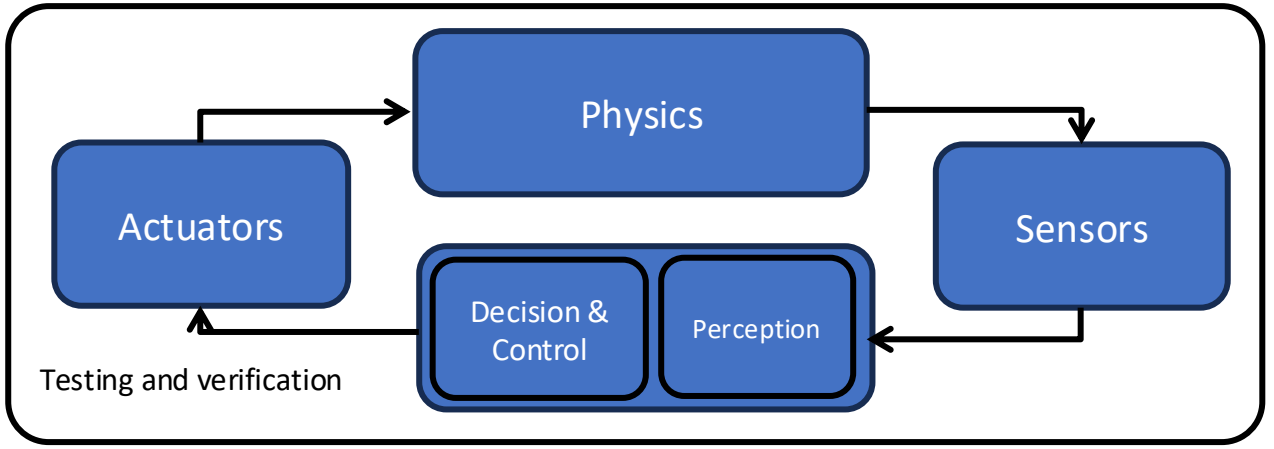
  If $\boldsymbol{q}.x_2 - \boldsymbol{q}.x_1 \leq d_{sen}$

  $\boldsymbol{q}'.v_1 \in [\boldsymbol{q}.v_1 - a_{1b}, \boldsymbol{q}.v_1 - a_{2b}]$

  $\boldsymbol{q}'.x_2 = \boldsymbol{q}.x_2 + \boldsymbol{q}.v_2$

  $\boldsymbol{q}'.x_1 = \boldsymbol{q}.x_1 + \boldsymbol{q}.v_1$





What did we miss in the AEB model?

If $x_2 - x_1 \leq 2.0$

$v_1 \in [v_1 - a_{1b}, v_1 - a_{2b}]$

else $v_1 = v_1$

$x_2 = x_2 + v_2$

$x_1 = x_1 + v_1$

- Acceleration, friction in dynamics
- Uncertainty in sensing
- Uncertainty in lead vehicle behavior
- Rear vehicle

"All models are wrong, some are useful."

Safety and liveness requirements

$R_{gap} = \{\alpha \mid \forall i\ \alpha_i.x_2 > \alpha_i.x_1\}$ non-zero gap $U_{gap} = \{\boldsymbol{q} \mid \boldsymbol{q}.x_2 - \boldsymbol{q}.x_1 \leq 0\}$

$R_{sp-lim} = \{\alpha \mid \forall i\ \alpha_i.v_1 \leq 70\}$ speed limit $U_{sp-lim} = \{\boldsymbol{q} \mid \boldsymbol{q}.x_1 \geq 70\}$

$R_{catch-up} = \{\alpha \mid \exists i\ 2 > \alpha_i.x_2 - \alpha_i.x_1 > 1\}$ catch eventually

A safety requirement is a requitement that every states along all executions should stay in certain good states

Equivalently, a safety requirement says that no execution of A ever reaches a bad or unsafe states

R_{gap} and R_{sp-lim} are examples of safety requirements with U_{gap} and U_{sp-lim} as the corresponding unsafe sets

$R_{catch-up}$ is not a safety requirement; it is an example of a liveness / progress requirement

A liveness requirement says that along every execution eventually some good state is reached

Summary

- Testing alone is inadequate---in theory and practice
- Automaton executions define system behaviors
- Requirements define desired or correct behaviors
- Verification proves that automaton satisfies requirements or finds counter-examples
- Safety requirements say that *no execution ever* hits unsafe states
- Next: Verification of safety requirements