Transportation CPS Safety Challenges

Prof. Phil Koopman
koopman@cmu.edu

And teams in ECE and NREC

Electrical & Computer ENGINEERING

NATIONAL ROBOTICS NREC ENGINEERING CENTER

CRUSHER

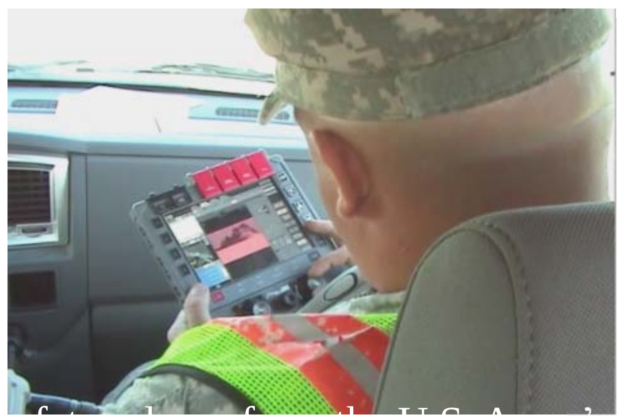
APD (Autonomous Platform Demonstrator)

TARGET GVW: 8,500 kg
TARGET SPEED: 80 km/hr

Approved for Public Release. TACOM Case #20247 Date: 07 OCT 2009

Example: RunTime Safety Monitor

- Dedicated, trusted hardware to monitor behaviors
  - Invariants to describe “safe” behaviors
    - For example: vehicle speed < speed limit
  - State machines to account for system operating modes
    - Different invariants are active in different modes (e.g., “stop” vs. “run”)
  - Emergency shutdown sequencing if any invariant is false



Also, Safety Shutdown Box for CHIMP

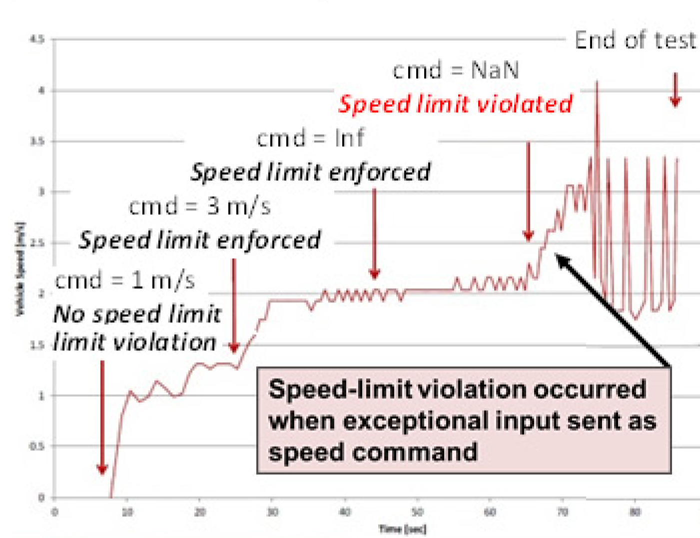
- CMU Highly Intelligent Mobile Platform
- DARPA Robotics Challenge Trials Dec. 2013

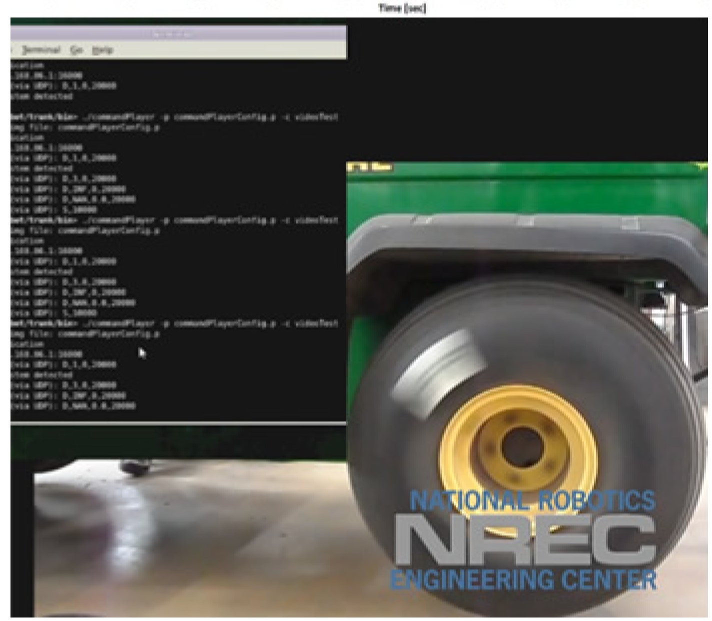
Stress Testing of Autonomous Systems

- Stress testing robots & autonomous vehicles
  - Web Search: “ASTAA NREC”

Distribution Statement A - Approved for public release; distribution is unlimited. NAVAIR Public Affairs Office tracking number 2013-74, NREC internal case number STAA-2012-10-23

RECbot Speed Limit Tests





Coming Soon To A Road Near You

Traditional Safety Approaches

- Elevators
  - Building codes describe required mechanisms
  - Electromechanical safeties (avoid trusting SW)

- Rail systems
  - Dual redundant hardware protection systems
  - Rigorously developed software EN-50126/8/9
    - Customers typically require these standards
    - “Safety net” architecture minimizes critical SW
  - Fail-stop approach – shut down if unsafe

Traditional Safety Approaches – 2

- Aviation
  - Do-178 and other FAA standards
  - Federal certifying agency (FAA)
    - Testing + examination of how system is designed
  - Fail operational; significant redundancy
- Automotive
  - NHTSA does not proactively certify safety
    - FMVSS don’t really address SW safety
  - MISRA Guidelines ➔ ISO 26262 safety standard
  - Some redundancy; tough cost constraints
    - Steering & brakes must fail (partially) operational

Why HW Safety Is Difficult

- "Safe" might be 1e-9/hr catastrophic failures
    - (It is easy to argue cars must be safer than that)
    - Single fatalities at perhaps 1e-7/hr (probably less)
  - Simplex hardware tends to fail at 1e-5 to 1e-6/hr
    - Cosmic rays result in bit flips (yes, really!)
    - Other things go wrong at about this rate
  - Thus, need **redundancy** to be safe
    - No single point failure end-to-end in the system
    - Takes some effort to get redundant components to properly synch.
- **<u>Infeasible to test</u>** to 1e-9/hr
  - Need testing time 3x-10x longer than failure rate

Why SW Safety Is Difficult

- Testing Software does not make it safe
  - See previous slide about testing duration
    - How do you know all SW corner cases tested?
  - Proving correctness is not enough for safety either
    - How do you know your requirements are correct?
    - Have you proven correctness under all fault conditions?
- Software safety requires process + testing
  - Follow standards (e.g., ISO 26262)
    - List of practices to follow based on criticality of SW
    - Need to ensure development process quality is there
  - Testing checks you really did it right
    - Testing is not “debugging” – test for absence of bugs
  - Generally, adaptive/robot software doesn’t fit the mold for existing SW safety

Autonomy Validation Challenges

- Specifying safety
  - Need to artfully select safety requirements as less than 100% of full system functionality
  - Need a realistic role for human operator
- Unconstrained environments
  - Uncontrolled, unpredictable urban roadways
  - Can inductive-based algorithms cover enough of the corner cases to be good enough?
- Trusting validation
  - How do you know your own system is really safe?
  - How do you know someone else's system is really safe when you cooperating with it?