Formal Methods for Lab-Based MOOCs: Cyber-Physical Systems and Beyond

Sanjit A. Seshia

UC Berkeley

Joint work with:

Edward A. Lee, Jeff. C. Jensen, Alexandre Donzé, Garvit Juniwal, Andy Chang

UC Berkeley & NI

CPSGrader.org

CPS-Ed 2014

November 4, 2014

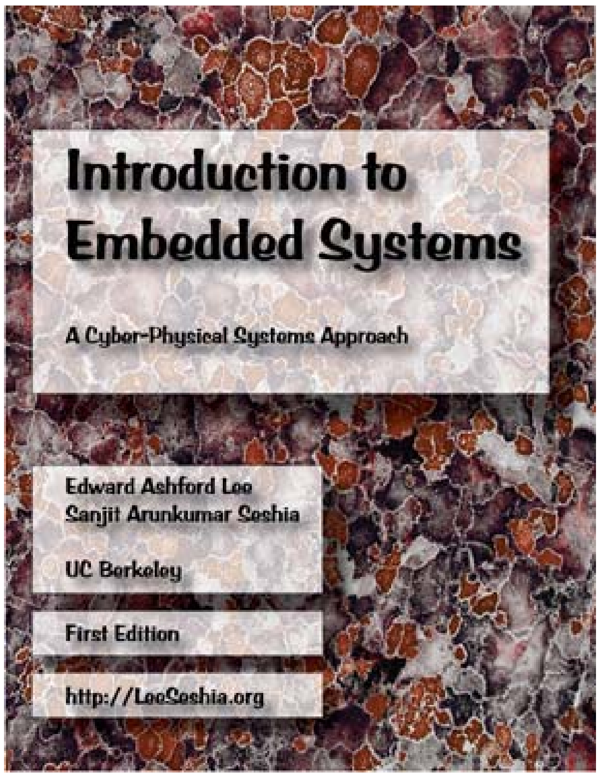
Massive Open Online Courses (MOOCs)

edX coursera UDACITY

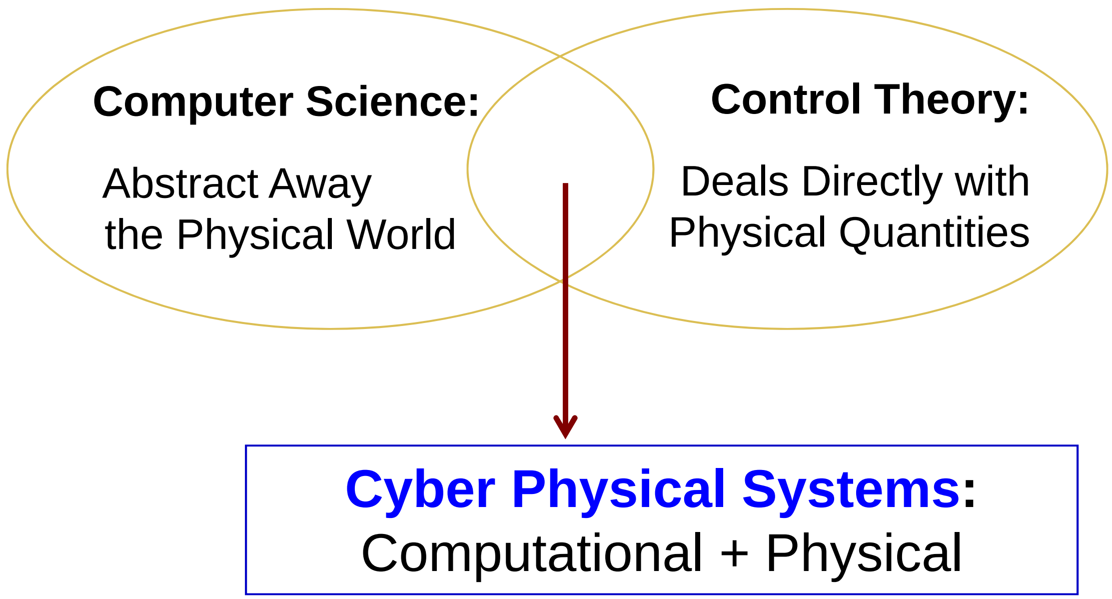
Courses from universities world-wide available to any one with an Internet connection

EECS 149: Introduction to Embedded Systems
UC Berkeley

This course introduces the *modeling, design and analysis* of *computational systems that interact with physical processes*.



http://leeseshia.org/

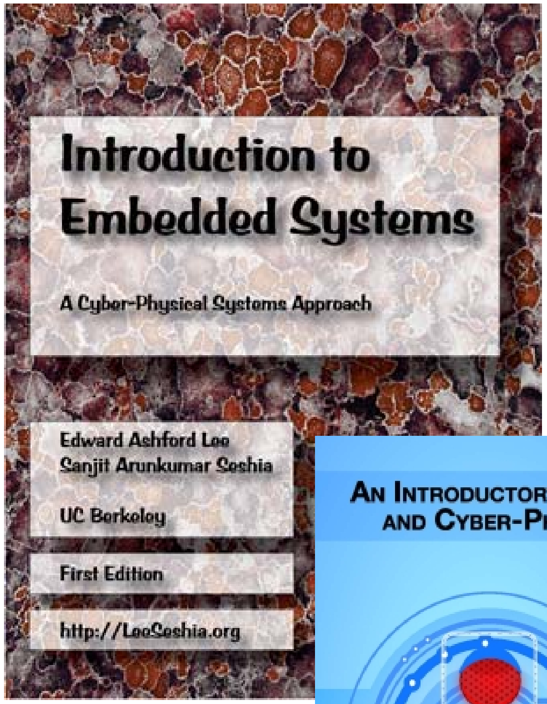


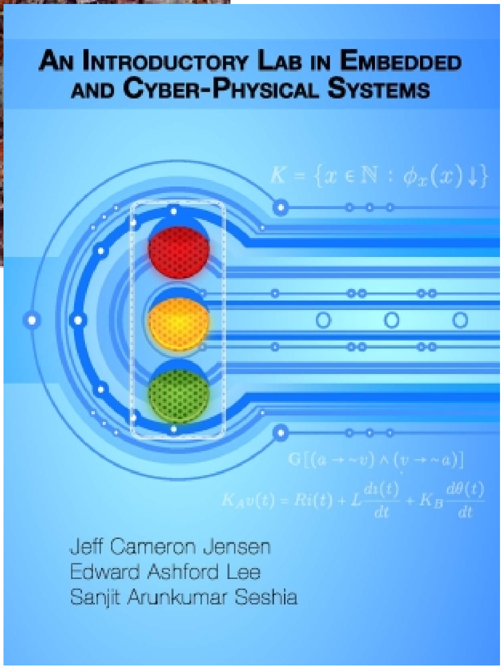
On-campus course gets somewhat diverse enrollment (EE/CS, ME, CE, …)

The Core Learning Experience: Exercises and Labs

- Textbook Exercises:
  – High-level modeling with FSMs, ODEs, temporal logic, etc.
  – Programming in various languages (C, LabVIEW, etc.)
  – Algorithm design and analysis (scheduling, verification, etc.)
- Laboratory (6 weeks)
- Capstone design project (12 weeks)

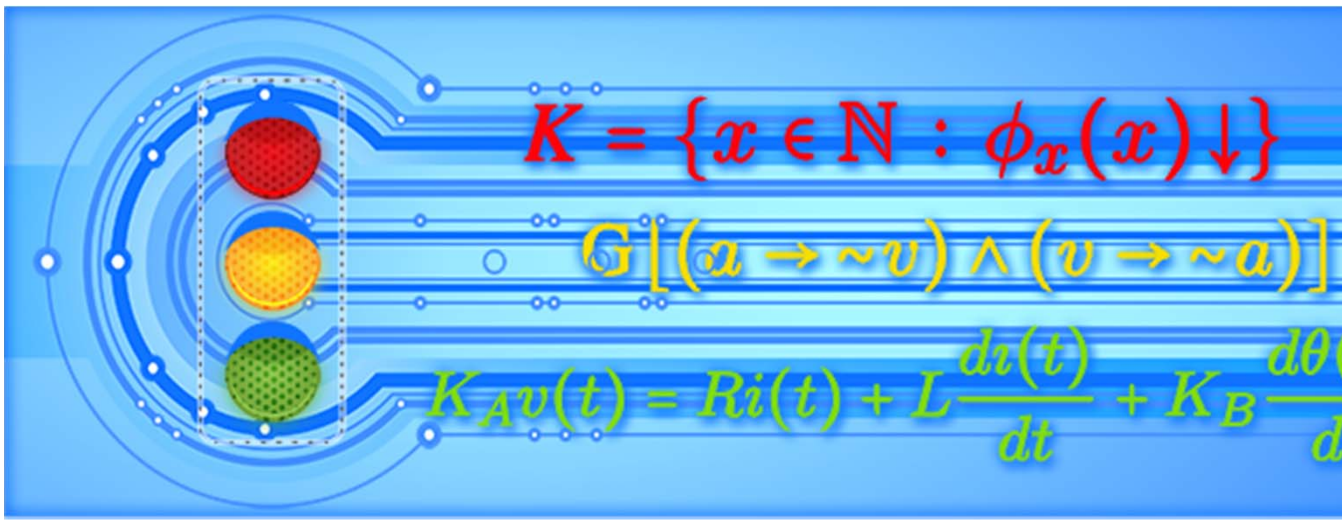
➢ **How to extend this experience to a MOOC version of EECS 149?**





EECS149.1x: Cyber-Physical Systems

- MOOC offering on edX: May 6 to June 24, 2014
- Berkeley-NI collaboration
- Virtual lab software for CPS: CyberSim
- First course to employ **formal methods** in auto-grader: CPSGrader



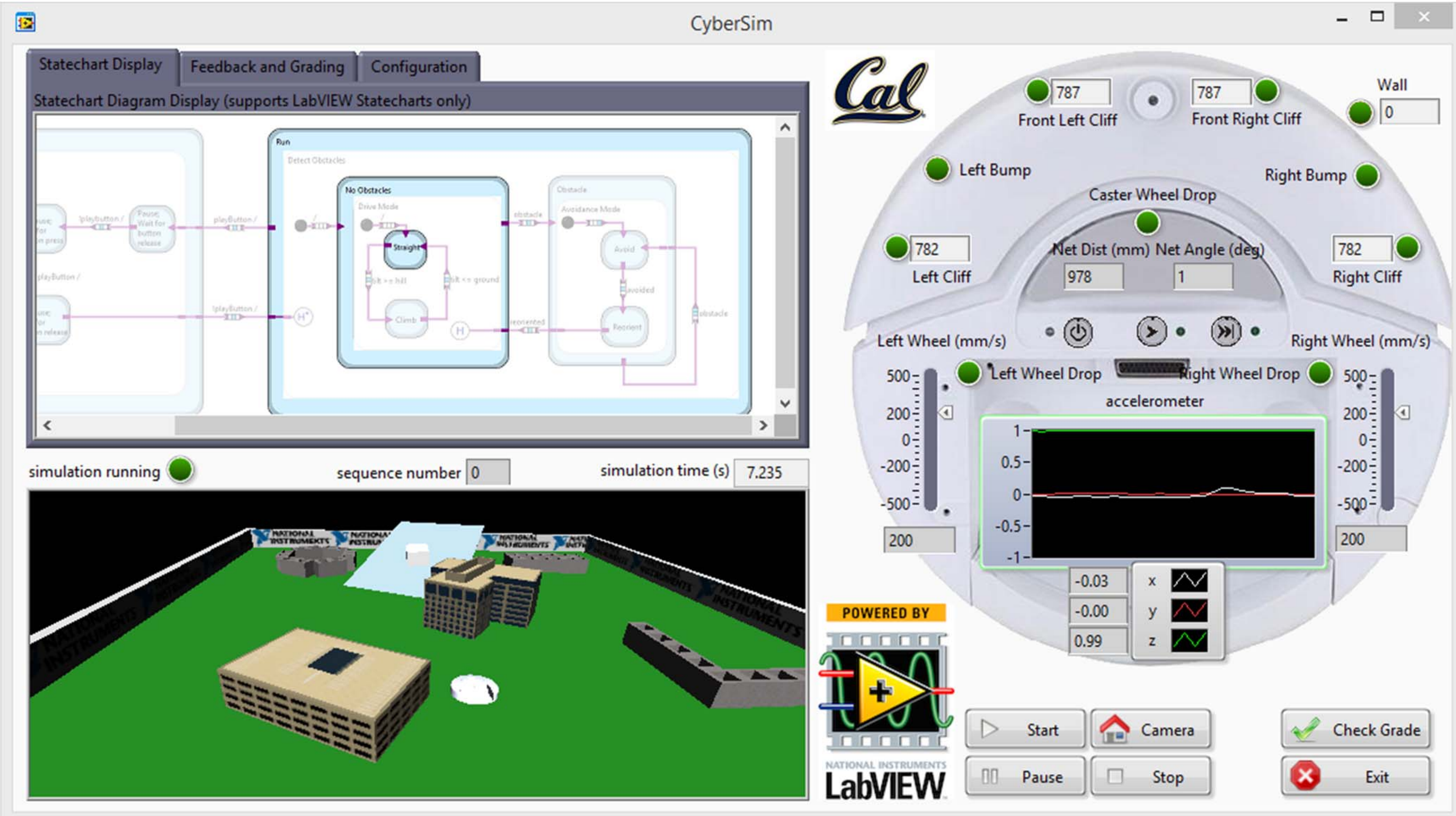
Roadmap for Rest of this Talk

- CyberSim + CPSGrader
  - NI Robotics Simulator + UC Berkeley Auto-Grader
  - Demo
- The EECS149.1x Experience
  - Statistics, Survey Results, Feedback
- Future Directions

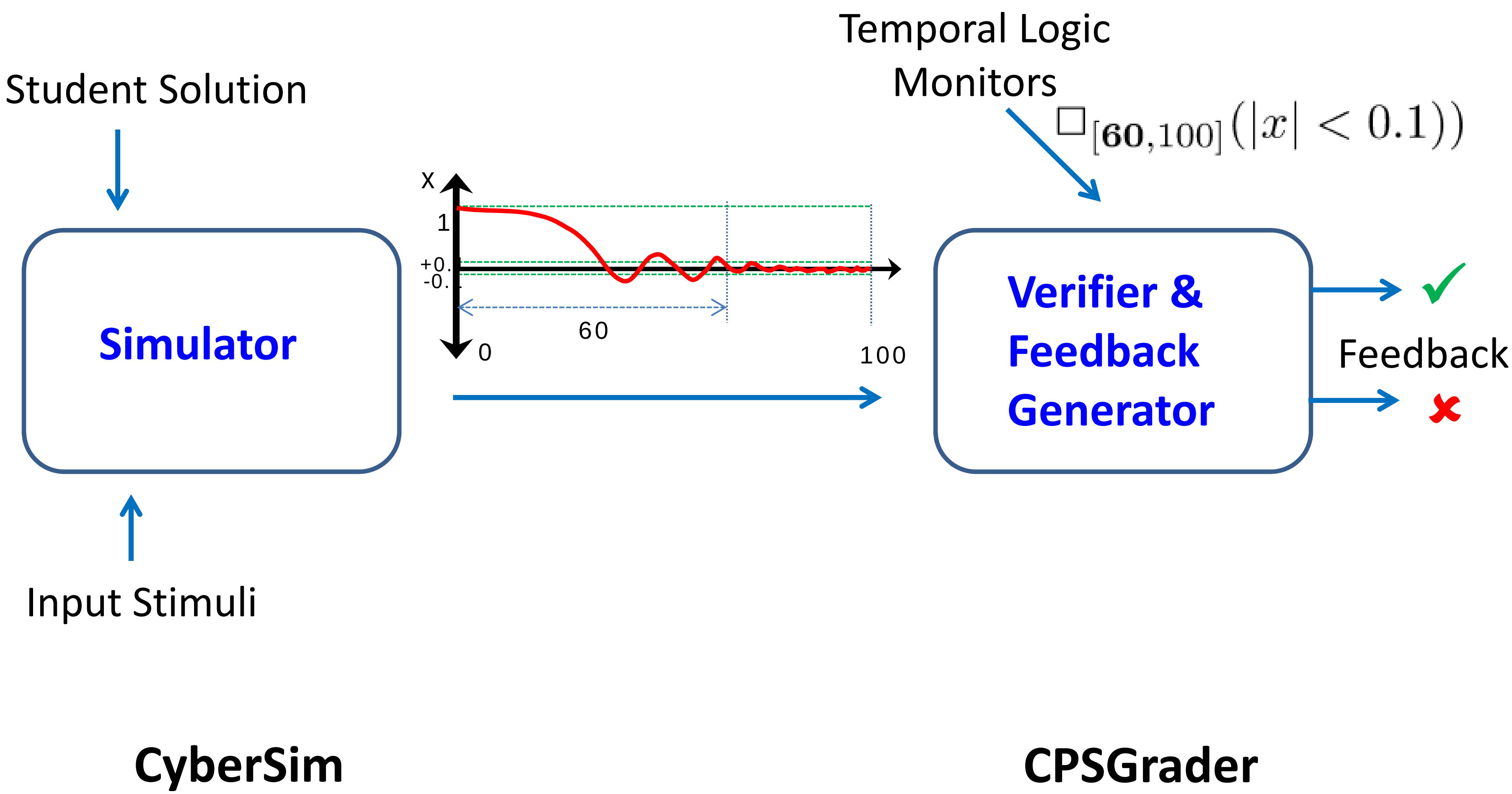
On-Campus Lab Assignment: The "Hill-Climbing" Robot

Goal: Online Virtual Lab with learning experience "comparable" to On-Campus Real Lab

Virtual Lab Assignment (Demo)



Components


Student Solution
Simulator
Input Stimuli
Temporal Logic Monitors
$\square_{[\mathbf{60},100]}(|x| < 0.1))$
x
1
+0.
-0.
0
60
100
Verifier & Feedback Generator
Feedback
CyberSim
CPSGrader

CPSGrader: Auto-Grading and Feedback Generation

[Juniwal, Donze, Jensen, Seshia, EMSOFT 2014]

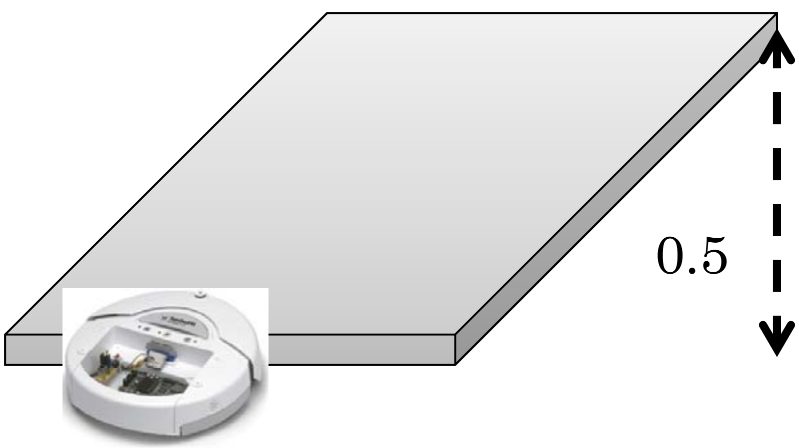
- *Auto-grading* = verification + debugging

- Employ *Simulation-based (run-time) verification*
  - get simulation trace
  - monitor *signal temporal logic* properties
  - localize *faulty* behavior

Fault Detection

- *Environment*: Arena composed of obstacles and hills
- *Monitor*: Signal Temporal Logic formula that captures presence of fault in a trace
- *Test*: Environment + Monitor

A test is "triggered" by a controller if the fault property holds on the simulation trace in the environment.



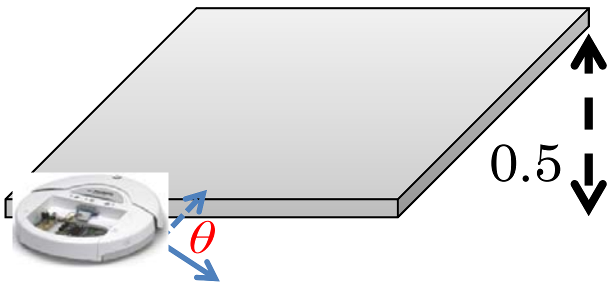
$$+ \qquad G_{[0,60]} \mathsf{pos}.z < 0.4$$

Technical Challenge

- Grading should be robust to variations in environment and student solutions.
  - Obstacle placement; hill incline & height
  - Different wheel speeds; strategies.

- Introduce parameters in environment and STL formula.
- Creating *temporal logic test benches* = solving a *parameter synthesis* problem.

Synthesis of Test Benches



Test Bench

$$G_{[0,\tau]}\text{pos}.z < h$$

+

$$[\theta \mapsto (-90, 90), \tau \mapsto (60, +\infty), h \mapsto (0, 0.4)]$$

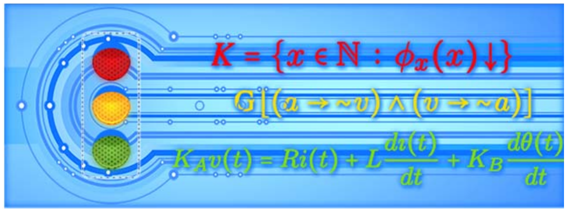
- Need to synthesize subset of parameter space that ONLY matches faulty solutions
  - Tedious to do manually!
- Coming up with *reference faulty/good solutions* is easier

- Goal: Synthesize fault subspace from reference controllers

Roadmap for Rest of this Talk

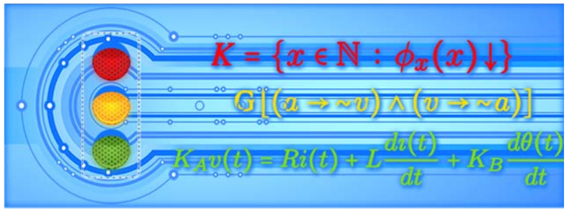
- CyberSim + CPSGrader
  - NI Robotics Simulator + UC Berkeley Auto-Grader
  - Demo
- The EECS149.1x Experience
  - Statistics, Survey Results, Feedback
- Future Directions

EECS149.1x: Basic Statistics



- 6-7 weeks
- 49 lectures, 10 hours 50 minutes of video
- 6 weekly lab assignments
  - 1 LabVIEW and Dev Tools tutorial
  - 1 Memory Architectures “lab”
  - 4 Virtual Lab exercises:
    - Week 1: Navigation, programming in C
    - Week 2: Hill climb, programming in C
    - Week 3: Navigation, programming in LabVIEW
    - Week 4: Hill climb, programming in LabVIEW
  - Hardware track optional

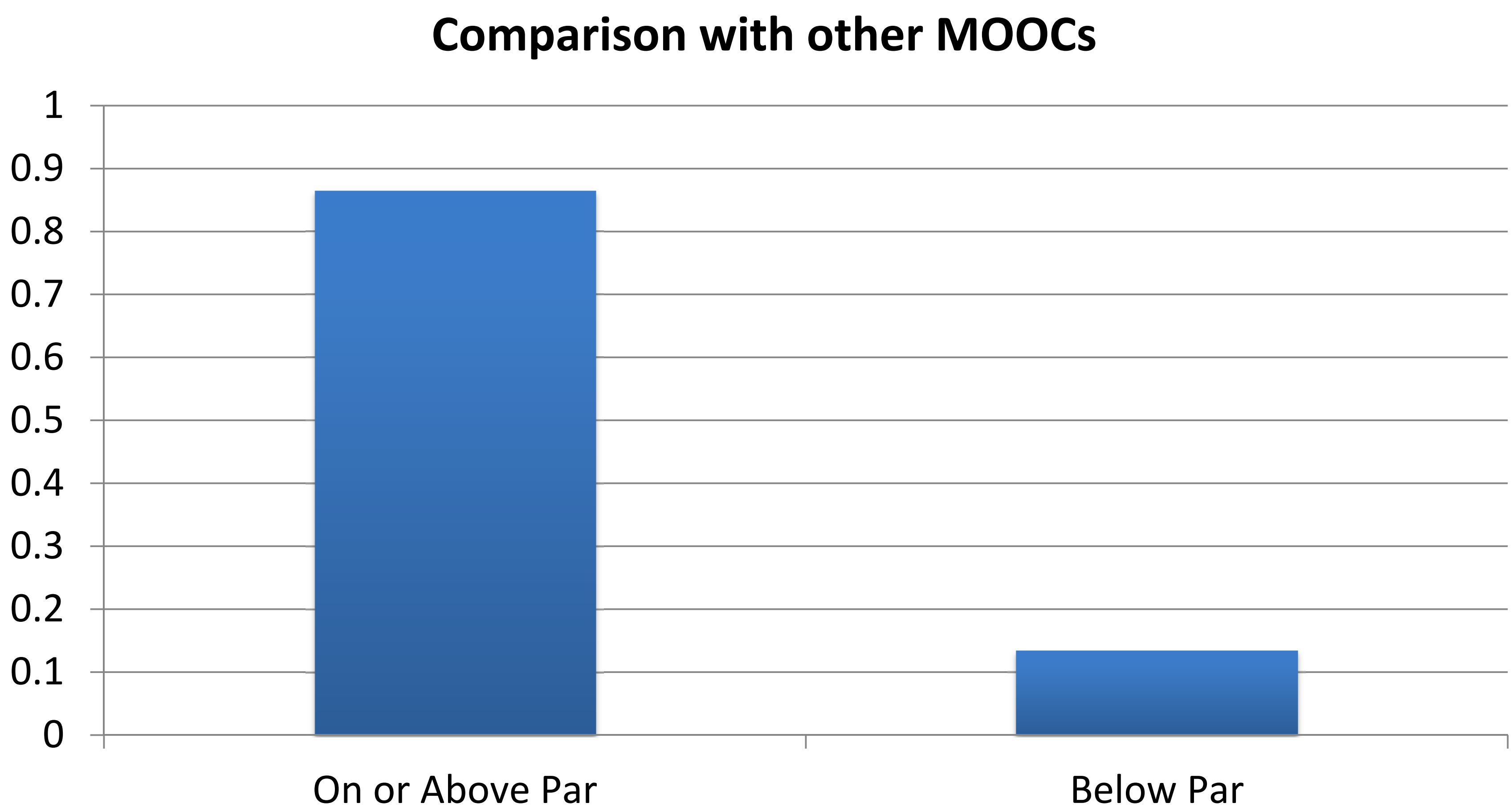
EECS149.1x: Basic Statistics



- 6-7 weeks
- 49 lectures, 10 hours 50 minutes of video
- 6 weekly lab assignments

- Peak Enrollment: 8767
- Largest number submitting any lab: 2213
- Number scoring more than 0: 1543
- Number who passed: 342 (4% of peak enrollment)

Student Survey



54% of students had taken 3 or more (other) MOOCs already

EECS149.1x: LabVIEW Stats

(About 200-300 survey respondents)

- Prior Experience: 59% NEW to LabVIEW

- LabVIEW vs. C for the labs:
  - LabVIEW was superior: 26%
  - LabVIEW was equally capable: 56%
  - LabVIEW was inferior: 18%

- Repeating lab in LabVIEW after doing it in C:

  73% felt it is a good thing
  - teaches different concepts and skills

Hardware Track: When deploying to the *real robot*, did you modify your solution from the simulator?

>90%

of controllers that passed the Virtual Lab auto-grader worked on the real robot with no or minor modifications

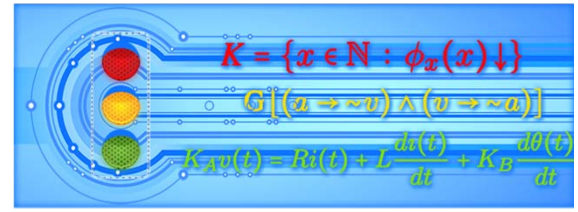
Students Reporting
Auto-grader Feedback as Useful:

86%

EECS149.1x: Lecture Modules

1. Introduction to CPS
2. Memory Architectures
3. Interrupts
4. Modeling Continuous Dynamics
5. Sensors and Actuators
6. Modeling Discrete Dynamics
7. Extended and Hybrid Automata
8. Composition of State Machines
9. Hierarchical State Machines
10. Specification & Temporal Logic

Survey on Lecture Modules

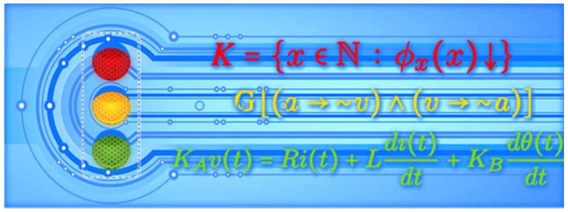


1. Introduction to CPS
2. Memory Architectures (1)
3. Interrupts (2)
4. Modeling Continuous Dynamics
5. Sensors and Actuators (1)
6. Modeling Discrete Dynamics
7. Extended and Hybrid Automata
8. Composition of State Machines (3)
9. Hierarchical State Machines (2)
10. Specification & Temporal Logic (3)

(1) Top Theory Topic

(1) Top Lab-Relevant Topic

Conclusion



- EECS149.1x: A first step towards enabling Lab-based MOOCs
  - Useful for growing enrollments on campus too!
- CPSGrader architected to be reusable for other courses
  - Circuits
  - Robotics
  - Mechatronics
  - …
- Formal Methods can offer much to Education in Science and Engineering
  - Virtual Science & Engineering Labs with built-in Auto-Grading can broaden participation

Future Directions

- Partial and Extra Credit
  - Quantitative semantics of Signal Temporal Logic
  - Quantitative satisfaction of temporal formulas
- Frequency-Domain Properties
  - Time Frequency Logic
- Online/Incremental Algos for Run-Time Verification
- Machine Learning for New “Unknown” Faults
- …