

Teaching Statement

Rahul Mangharam

My teaching goal is to make students enjoy learning algorithms for autonomous systems which operate at their extreme limits, eliminate exams (!) and yet get every student to intellectually stretch themselves. The courses I develop are intense, organized, and open source. For example, the F1Tenth Autonomous Racing course and user-generated videos have over 1 million views, 2000+ active users on the f1tenth-teams Slack, and has resulted in over 15 international competitions, workshops, and tutorials. The course material I developed is taught in TU-Vienna, Lehigh, UVA, TU-Munich, Clemson, Oregon State, ETH Zurich, Uni. of Lincoln Nebraska, Carnegie Mellon, UT Austin, KAIST and other universities. In every teaching session, 2/3rd time is for instruction and the remaining is for hands-on tutorials which we do with the students. This incorporates an apprenticeship model which communicates not just the material but the nuances of how to approach the topic conceptually and as a learned skill.

The central thread of my teaching is to approach everything as a system. Unlike traditional courses in signal processing, computer vision or circuits which are taught in isolation, here students get to view all aspects of the system from perception, planning, control, power systems, GPGPU processing, real-time operation, and vehicle dynamics. By creating platforms that are complex enough but not too complex, students get to approach the system as a whole. They then learn to identify how to partition the problem across the different subsystems and build pipelines to accomplish Simultaneous Localization And Mapping (SLAM), Model Predictive Control (MPC), Rapidly exploring Random Trees (RRT) motion planners and understand performance bottlenecks from bottom-up and also across hardware and software boundaries.

Teaching Philosophy:

My teaching goal mirrors my inter-disciplinary approach towards research: I aim to ensure that students cultivate a holistic view of life-critical and safety-critical system development by drawing stronger connections between autonomous systems, embedded systems, control systems, formal methods, and hands-on machine-learning-at-the-edge of platform development. I design courses that have a significant experimental component, where students work in teams to develop sizable distributed embedded systems that integrate concepts across electrical and computer engineering, computer science, bioengineering and mechanical engineering. These include building testbeds for energy-efficient heating/cooling systems, EV powertrain systems, Heart-on-a-Chip medical devices, biofeedback control systems and leader-follower quadrotor aircraft systems¹.

By emphasizing *learning by doing*, I aim to develop students' confidence in deconstructing a design problem, iterating over several inefficient designs, presenting intermediate results, and incorporating creativity in engineering solutions. In following this *design-driven approach*, every student is continually making decisions on how to partition functionality to different sub-systems, how to keep things simple, how to define clean and generic interfaces at hardware/software boundaries and finally how to demonstrate the core ideas within the operational system in an effective way. This emphasis on developing hands-on critical thinking skills forms the bedrock of my teaching approach.

In the rest of this statement, I will summarize the Autonomous Racing, TinyML Tiny Machine Learning, Embedded Systems and Model-based Embedded Systems courses I have developed at Penn and discuss plans for a broader curriculum towards a Department of Autonomy. I will describe my efforts in developing cross-school collaborations within Penn, student and faculty-based teaching events, and establishing Penn as a recognized force across global engineering communities.

A. F1Tenth Autonomous Racing – Learn to Build, Code and Race.

In order to facilitate research and education in autonomous systems, I developed an open-source research platform of high-performance autonomous racing cars that are 1/10th-scale of Formula-1 cars and can reach a top speed of 50mph. F1Tenth [<http://f1tenth.org/>] enables a wide range of machine learning engineering with perception, planning, control and coordination modules. In addition to the platform hardware, my teaching team developed an autonomous vehicle software stack and a set



¹ For a really exciting compilation of ESE350 project videos, see <http://tinyurl.com/ese350videos>

of simulators as plug-and-play replacements for the 1/10th-scale platform itself. The course covers five modules and include three races for evaluation:

1. *Introduction to ROS, F1/10 & the Simulator:*

Introduction to self-driving hardware and full software stack, automatic emergency braking, LiDAR, rigid body transformations, Laplace domain dynamics, and PID control for wall following.

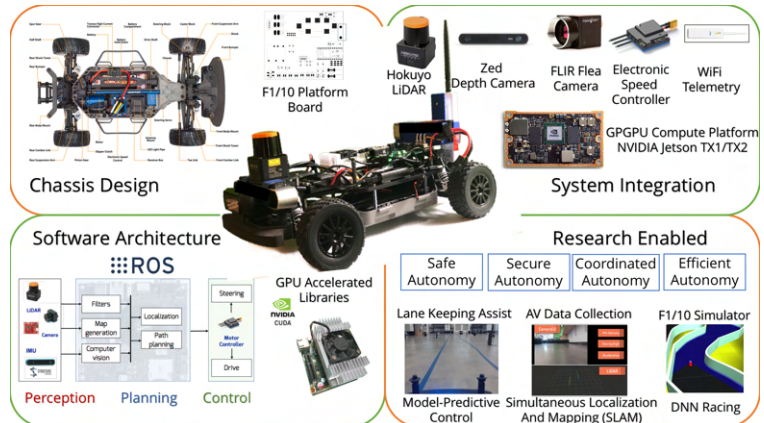
2. *Driving using Reactive Methods:* Build a vehicle, tune the electronic speed controller, and implement reactive driving methods such as 'follow the gap' and complete **Race 1**.

3. *AV Mapping & Localization:* Understand key elements of SLAM by implementing scan matching and particle filters; use Google Cartographer SLAM; implement pure pursuit driving; complete **Race 2** using maps.

4. *AV Planning:* Learn Moral Decision Making for autonomous systems; implement planning with rapidly exploring random trees (RRT) and understanding model-predictive control (MPC) for race line optimization.

5. *Learning & Vision:* Design and implement algorithms for object detection and pose estimation, reinforcement learning and visual feature extraction.

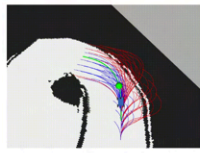
6. *F1/10 Grand Prix: Race 3* will include a project to implement racing strategies which combine fast perception, agile planning, and aggressive control for race optimization.



I deliberately replaced midterms with races. The students spend between 10-14 hours a week on the course and we develop an environment for experimentation, frequent failure and friendly competition. The main reason students take this course is because they want to tie together all the topics they studied in isolation within an applied and competitive context. The course has 8 intense labs covering SLAM, MPC, sampling-based planners, learning-based visual navigation and is followed by a month-long team project on a topic the students choose. These have gone on to be publishable papers in ACC, ICRA and IFAC World Congress. This is a popular course with over 90 students on the waitlist and a quarter of the students continue in the following semester to pursue independent study projects. Several have joined the Ph.D. cohort too.

ESE615 Course Autonomous Racing Spring 2022		Module A: Introduction to ROS, F110 & the Simulator			
		Lab	Date	Topic	Lab
A. Robotics and ROS basics a. Sensors and Mechanics b. Simulator c. Safe AV control basics B. Navigation a. Reactive planning b. Mapping and Localization c. Pure Pursuit planning d. AV Ethics C. AV Race Planning a. Raceline optimization b. RRT Planner c. Model Predictive Control D. Vision and Learning a. Detection and Pose Estimation b. RL E. Hands-on Project a. Drive till you MHz!	1	8/28/2019	Introduction	Lab 1: Intro to ROS	
	2	9/4/2019	Automatic Emergency Braking	Lab 2: Safety Co-Pilot	
	3	9/9/2019	Rigid Body Transformation	Lab 1	
	4	9/12/2019	Laplace domain dynamics, PID	Lab 3: Wall Following	
	Module B: Reactive Methods & RACE!				
	5	9/16/2019	VESC tuning	Lab 3	
	6	9/18/2019	Reactive Methods: Follow the Gap	Lab 4: Follow the Gap	
	7	9/23/2019	Race Prep	Race 1	
	8	9/25/2019	Race Day		Lab 4
	Module C: Mapping & Localization				
	9	10/1/2019	Localization: Scan Matching I	Lab 5: Scan Matching	
	10	10/3/2019	Localization: Scan Matching II		
	11	10/7/2019	Localization: Particle Filter		
	12	10/9/2019	SLAM: Cartographer	Lab 5	
	13	10/14/2019	Pure Pursuit	Lab 6: Pure Pursuit	
	14	10/16/2019	Putting everything together. Maps		
	15	10/21/2019	Race Prep	Race 2	
	16	10/23/2019	Race Day		Written Assignment
	Module D: Planning				
	17	10/28/2019	Moral Decision Making		
	18	10/30/2019	Raceline Optimization		
	19	11/4/2019	RRT	Lab 7: RRT	
	20	11/6/2019	MPC	Written Assignment	
	Module E: Learning & Vision				
	21	11/11/2019	Vision: Detection and Pose Estimation I		
	22	11/13/2019	Vision: Detection and Pose Estimation II	Lab 8: Vehicle Tracking	
	23	11/18/2019	Reinforcement Learning I		
	24	11/20/2019	Reinforcement Learning II	Lab 8	
Module F: F1/10 Grand Prix!					
25	11/25/2019	Invited Lecture			
26	11/27/2019	Project Demos			
27	12/2/2019	Race Prep	Race 3		
28	12/4/2019	Final Race Day		70	

F1TENTH: Research projects for students



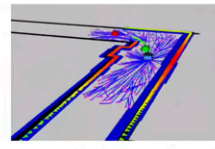
MPC Racing Stack



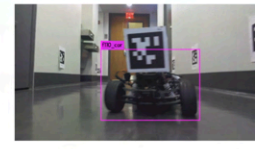
Overtaking & Improvisation



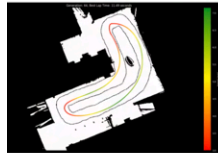
FITENTH Hardware 2.0



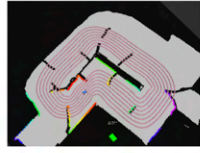
Multi-Vehicle Coordination



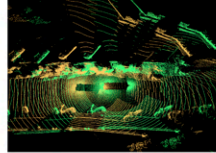
Object Detection



TUNERCAR*



Virtual Racing Front End



Visual Slam



Vision-based Navigation



CAD2CAV

The International F1Tenth Community: F1Tenth has over 80+ university community partners for research and education. In addition to teaching the course at Penn, we have organized 15 international autonomous racing competitions for multi-vehicle racing in top robotics, transportation, and Cyber-Physical Systems conferences. Five more races have been scheduled for 2024 at CPSweek (Hong Kong), ICRA (Japan), IROS (UAE), IEEE Intelligent Vehicles (Korea), ITSC (Canada). The F1Tenth research community has published over 50 papers using the platform for neuromorphic computing, safe ML, learning Koopman operators, multi-friction racing, NN-based localization, etc. F1Tenth is supported by a \$1.5MM NSF CISE Community Research Infrastructure award and industrial partners such as National Instruments, Nvidia, Intel, SICK, etc. Driving at the limits of vehicle performance helps us accelerate the development of safe autonomous vehicles.



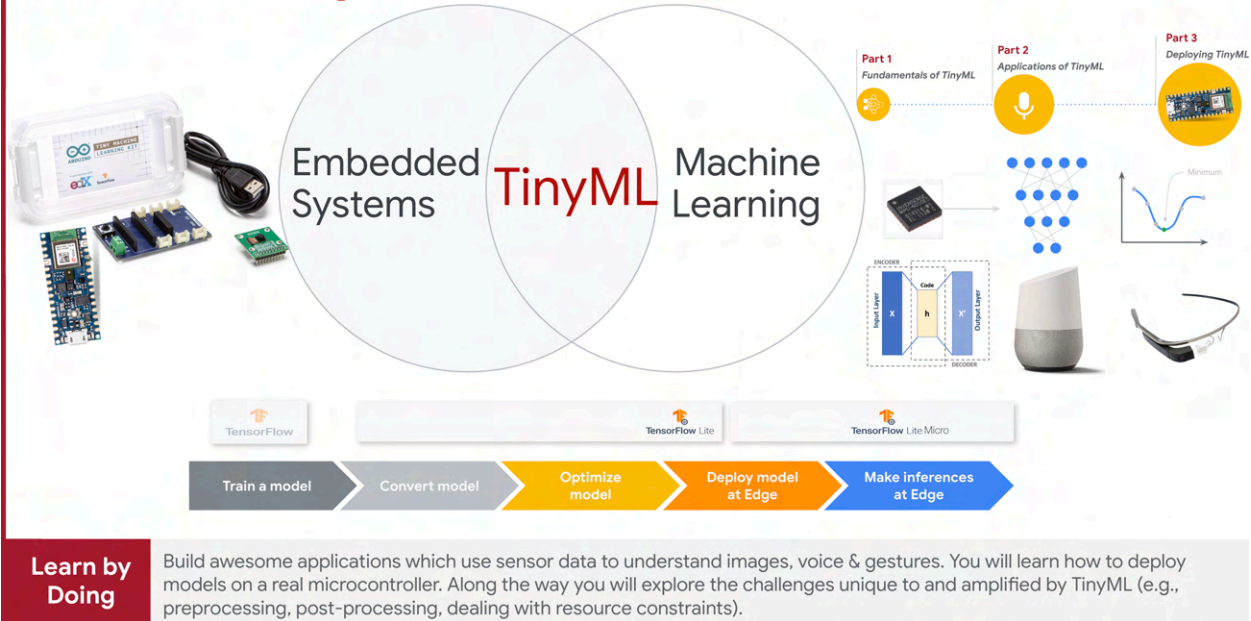
F1/10 COMMUNITY PARTNERS 80+



The 2nd Korean F1Tenth Autonomous Grand Prix was held on October 17-18, 2023. It had 31 teams, and the top prize was \$40,000! F1Tenth involves all top 20 Korean universities and has become a national program.



ESE 3600 TinyML Where Embedded Systems & Machine Learning meet



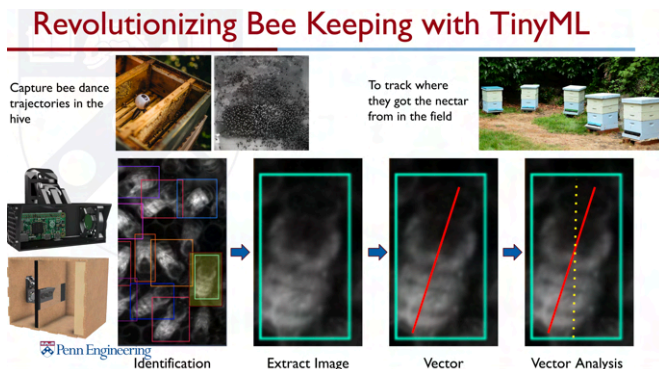
B. Tiny Machine Learning (TinyML) curriculum development:

My first decade of teaching was on real-time embedded systems. However, my research had long migrated to machine learning (ML) on the edge. A natural progression was to develop and teach a course on TinyML, which is an exciting and emerging field at the intersection of embedded ML applications, algorithms, hardware, and software. TinyML differs from mainstream server/cloud-based ML in that it requires not only software expertise, but also embedded-hardware expertise. This course emphasizes hands-on experience with ML training and deployment in tiny microcontroller-based devices with onboard sensors, a camera, and a breadboard with wires—enough to unlock capabilities such as image, sound, and gesture detection.

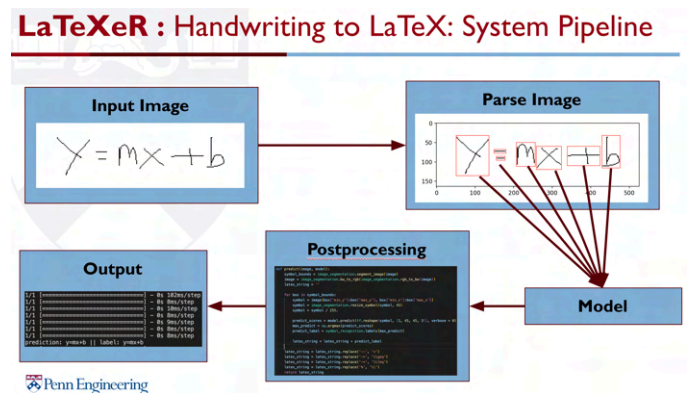
There are no prerequisites besides basic scripting in python but before they know it, students are implementing an entire TinyML application. Students go from learning the fundamentals of ML and embedded systems, building datasets, training, and compressing tiny ML models in TensorFlow, TensorFlow Lite and TensorFlowLite Micro to conceiving and deploying their own TinyML applications.

TinyML has been a hit with students who want a hands-on introduction to ML and want to build complete solutions. They think of this as a first course in both machine learning and embedded systems and they don't need any background in either. Here are two example projects – one team of 2 freshmen designed an embedded camera system to track the trajectories of bees in the hive. This is used by beekeepers to identify the approximate locations the bees acquired the nectar from the fields. Another project, LaTeXER captured handwritten text and converted it to LaTeX. Other projects included IMUs, microphones and other sensors.

Revolutionizing Bee Keeping with TinyML



LaTeXER : Handwriting to LaTeX: System Pipeline



C. Developing Embedded Systems Programs at Penn

Since 2008, I have been part of the founding committee for establishing the embedded systems curriculum at Penn through the development of the Computer Engineering (CMPE) undergraduate program and, separately, the Masters in Embedded Systems (EMBS) program (1st such program in the nation). From 2015-2019, I served as the EMBS Program Director. Both programs have been tremendously popular, with high quality students. Embedded Systems graduates, who were under my supervision, during and after taking my courses, have gone on to pursue graduate studies in Stanford (Kevin Conley and Jeff Kiske), Georgia Tech (Matt Hale), Duke (Ashleigh Thomas), UCLA (Paul Martin), CMU (Utsav Drolia), MIT (Max Li), among others.

Prior to 2014, very few technology companies recruited from the ESE Department and that has changed tremendously with both programs. Students joined SpaceX, Tesla Motors, NASA (Langley), GoogleX Labs, BOSCH Research, Intel, Nvidia, Qualcomm, etc. Students graduating from the Autonomous Systems course were very successful in getting top placements in Tesla AutoPilot group, Honda's and Nvidia's Autonomous Vehicles group, Rivian's electric drivetrain group, Aurora's and Zoox's motion planning and control team, nuro's software infrastructure team, among others. This is a testament that the students are nationally competent with a broad and deep skillset of model-based and data-driven system design.

Through the 120+ undergraduates and 140+ Masters students that I have advised, I have come to appreciate the need to balance theory and skill development. As we refined the curriculum, our goal was to ensure students not only get solid foundations in becoming "Platform Architects" across the computing stack, but also become skilled in "Systems Thinking" for debugging and developing complex life-critical systems.

D. ESE350 Introduction to Embedded Systems

This course introduces the use of microcontrollers, sensors, and actuators in building real-time embedded systems that interact with the physical world. With an approach focused heavily on learning by doing, the labs are designed to be interactive, fun, yet challenging. The students build:

- *Arduino from Scratch* – in the first 2 weeks students build the entire Arduino on a breadboard and are introduced to power management, bootloaders, memory layouts, and circuits with over 36 components.
- *Build a musical instrument with Sensors and actuators* - to learn timers, analog-to-digital conversion interrupts, analog circuits.
- *Interactive Hand-held Pong Game System* – learn to interface touchscreens, LCDs, and motion sensors to make a fun and interactive game system.
- *Balance Bot* - develop a self-balancing bot using stepper motors, motor drivers, an IMU and interface it with a Zigbee wireless remote. Implement controls for stability and position control and race!

The lab exercises begin with well-defined instructions where the students initially program the 16-bit microcontrollers hardware in C at the bare metal. The labs progressively get more open-ended and the students' progress to use programming libraries and eventually work with embedded operating system (using 32-bit ARM multiprocessors). This ensures the basics of hardware-software interfaces are learned and allow the students to gradually abstract away the low-level details for more *system-wide thinking*.

The final 4 weeks feature a project where students are asked to build creatively on the foundations, they have learned by designing and developing an embedded system of their own. Examples of such systems include tele-operated leader-follower quadrotors, body sensor games based on hacked Nintendo/Sega console games, electro-mechanical chess, connect-4 and basketball shootout machines, spherical robots, etc. Projects from ESE350 have won over *16 national and international awards*.

E. ESE519 Real-Time and Embedded Systems

This core graduate course covers the concepts, theory, and tools necessary to understand, design, and build real-time and concurrent embedded systems. The course is spread across five major modules starting with an introduction to networked embedded systems, real-time scheduling theory, concurrent programming and distributed systems theory, life-critical systems, and case studies in embedded system failures. Five labs introduce programming with a real-time operating system (RTOS); networked operation across distributed embedded systems; development of routing protocols; and design of safety-critical systems. Each group is

given a set of wireless embedded nodes, 32-bit embedded multi-processor boards (700MHz and 1GHz) and a variety of sensors and actuators. Final projects have included:

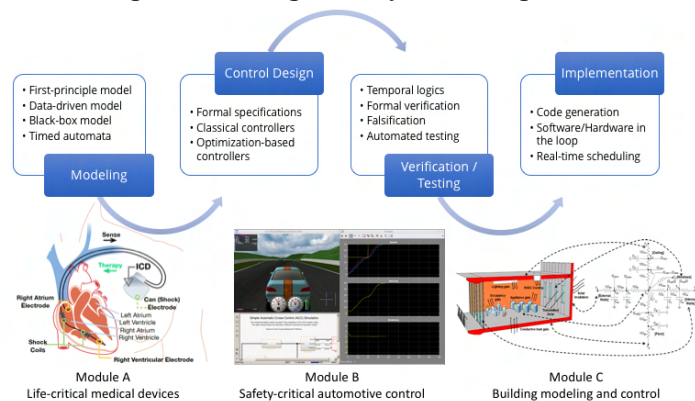
- Energy-efficient building automation systems, which integrate sensing, distributed controls and building automation algorithms for heating and cooling in HVAC systems.
- Automotive embedded systems that integrate control systems for traction, stability, anti-lock braking, and adaptive cruise control with a hardware-in-loop testbed.
- Robot-soccer with multi-robot consensus protocols for defense/offense, using computer vision.
- Implementation of new wireless protocols (ISA100.11a) for industrial control and automation

Student projects from ESE519 have resulted in over 12 international conference publications and demos in venues such as ACM/IEEE Cyber-Physical Systems Week, ACM Embedded Systems Week, ACM Building Systems Symposium, and the IEEE Real-Time Systems Symposium (RTSS).

F. Model-based Systems Curriculum Development

I routinely experiment with new course concepts. To translate our research into teaching, I developed a course on foundations for Model-based Design for Cyber-Physical Systems. This course allows the student to journey from specifications to models to implementations to integration for rigorous system design of controls, computation and communication to answer:

- *What does it take to design and implement life-critical software in an implantable cardiac defibrillator?*
- *How to certify that a car in autonomous cruise control mode will drive itself safely?*
- *How do you develop and tune controls for skyscrapers that have complex interactions with the environment, occupants, and equipment?*



This course focuses on modeling for verification, testing and control of such safety-critical systems. The course is 50% theory covering the foundations of temporal logic, controls, and falsification and 50% practical skill development with the use of industry standard tools in verification, testing and model-based development (MATLAB/Simulink, EnergyPlus, UPPAAL). In three month-long modules, we cover in-depth modeling of implantable cardiac medical device software and systems, testing of advanced driver assistance software in automotive controllers and data-driven modeling and control of buildings. This course provides the foundations and tools for a career focus in model-based design of embedded systems.

Recent awards with students I mentored after they took my courses:

1. **First Prize Autonomous Electric Go-karting Competition** 2023
US Autonomous Karting Series at Purdue University
2. **Winner 10th International F1Tenth Autonomous Racing Competition at ICRA** 2023
International Conference on Robotics and Automation
3. **Winner International JSAE Autonomous Driving Competition** 2022
Japan Society of Automotive Engineers
4. **DASD Best of Session Award** 2020
Drone Conflict Management at the 39th Digital Avionics Systems Conference
5. **SIGCSE 2nd Best Paper Award** for Curricula Initiatives 2020
ACM Technical Symposium on Computer Science Education (SIGCSE)
6. **NeurIPS Best Demonstration Award (Runner-up)** 2019
34th Annual Conference on Neural Information Processing Systems (NeurIPS)