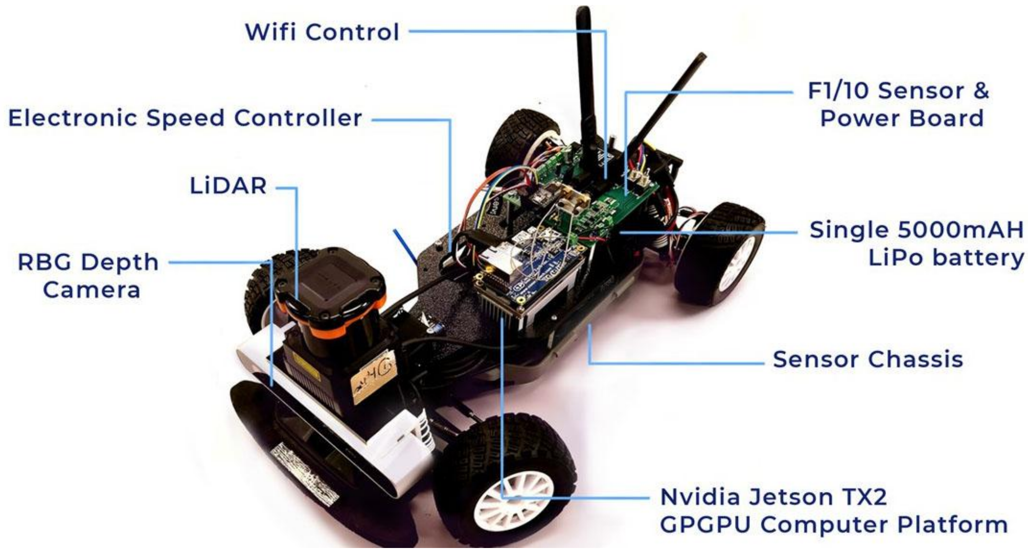
Autonomous Racing CARS



Meeting Times: Tuesday/Thursday (9-12am)

F1/10 is a 1/10th scale autonomous racecar platform that is 10x the fun! This is a hands-on course that will teach you the basics of perception, planning and control in a fun approach. Students who completed this course have gone on to pursue exciting careers in autonomous systems. This course is appropriate for seniors and graduate students.

Build

Build an autonomous race car from scratch! Starting from a stripped-down 1/10th-scale mechanical chassis capable of 40mph, add the sensors, the compute boards, and the electronic components needed for autonomous driving. In the process you will learn about different sensing modalities, their pros and cons, PID tuning, and principles of embedded computing.

Code

Now that you have the hardware ready, you will learn how to program an autonomous robot to turn the raw sensory data into actionable information, and ultimately into navigation and control commands for your car. Your code will leverage ROS capabilities, the widely popular OS for robotic applications.

Race!

Your team will compete against others for completing the track in the fastest time. In the second half of the class your team will independently develop racing algorithms that will give you the edge over the others. We will guide you in learning navigation algorithms such as pure pursuit and probabilistic roadmaps.

Instructors and TAs

Radu Grosu, radu.grosu@tuwien.ac.at

F110-Team website: http://f1tenth.org

Schedule

Note that the lecture topics are subject to change, based on class progress and class interest. The lectures are going to be every Tuesday and Thursday from 9-12am.

- Module A: Introduction to ROS, F110 and simulator
  - Course overview, Team Formation, LIPO, Safety
  - Notions of correctness, intro to LIDAR-safety assist, time-to-collision, Ray Marching/casting
  - Pose representation and coordinate transformation
  - Laplace-domain dynamics, PID for wall following
- Module B: Reactive methods and Race
  - VESC tuning
  - Reactive methods, follow the Gap and variants
  - Race preparation
  - Race day
- Module C: Mapping and localization
  - Localization: Scan matching I
  - Localization: Scan matching II
  - Localization: Particle filter
  - LAM: Cartographer
  - Pure pursuit
  - Putting everything together, maps
  - Race preparation
  - Race day
- Module D: Planning
  - Moral decision making
  - Race-line optimization
  - Rapidly expanding random trees (RRT)
  - Model-predictive control (MPC)
- Module E: Learning and Vision
  - Computer vision I
  - Computer vision II
  - Reinforcement learning I
  - Reinforcement learning II
- Module F: Grand Prix
  - Invited lecture
  - Project demos
  - Race preparation
  - Final race day
  - Final project documentation submission

Course grading and evaluation

[50%] Labs
[30%] Competition performance (mid-semester and end-of-semester) and public communication

[5%] Competition document: a 10-page document summarizing your approach to the competition (software architecture, algorithms, hardware, tests, etc), examples of performance results, etc.
[5%] Peer review: an anonymous evaluation of your work performed by your teammates.
[10%] Participation in class and TA evaluation

Evaluation:

For most labs, we will plug your code into a pre-set test (i.e., a benchmark). The test should run with your code in it. You will be told what the test is in the assignment, so you can make sure that your code at least runs. If you don't find the test description in the assignment, tell us! We will examine the performance of your car. On some labs, we will ask you for a code walk-through, or even code tracing. Software quality (e.g., comments, code structure, modularity, etc) counts towards the grade.

A lab typically lasts one week, unless otherwise specified. It is assigned on the first lecture of the week, and due *before* the first lecture of the following week. We will download all submitted solutions before the lecture. We will also release the solution code at the same time that we download your submissions, so there is really no room for late submissions. You will use the released solutions, hereafter referred to as the *reference implementation,* in the following labs. This way, everybody starts every lab from the same baseline code that we know functions well.

Lab logistics

Most labs have a skeleton code on which to base your solution. The skeleton code will be pushed to the following *public* git repository:

https://github.com/cps-arc/spring2020-skeletons
Git clone this repository. To get a new lab, just do a git pull. Sounds like gibberish? Here's a git tutorial - take 2 hours to follow it. You need command-line git installed on your machine:
https://www.atlassian.com/git/tutorials/setting-up-a-repository

We will release the reference implementation to the following private git repo, to which the instructors will give you access:
https://github.com/cps-arc/spring2020-solutions

Some labs ask you to work with ROS bags. We will provide these bags on a shared Google drive. We will be using Github classroom to provide you with a private github repo in which to do your development work as a team. **Your submission will be considered the last commit your team did before the deadline.** Some assignments are individual and not group-based. Those will not be code development. You can upload those assignments to Canvas as usual.

If we decide to do code tracing with you, it will be done at the beginning of the first guided lab session that follows the deadline. Thus if the deadline is on Monday at 4pm and your first guided lab session after that is Monday at 5pm, that will be the session which we start with code tracing and explanation. This way the code is still fresh in your minds.

Pre-requisites

The most important technical pre-requisite is good programming skills in C++ and Python. You will be coding or reading code in both languages. Python is easy to learn if you don't already know it, but you will have to do that on your own time. You will also need knowledge of frequency transform concepts (e.g., Fourier or Laplace), basic matrix algebra and differential equations.

IT

Use the course TISS support to ask and answer questions about course content. Instructors may chime in from time to time. Remember that a "private" question on TISS can be read by all the instructors and TAs.

Use the course TISS to submit your lab solutions: code (in a single folder for each lab), solutions to paper-and-pencil, 3-mins videos for your labs, etc. Also, use it to communicate with the instructors concerning logistics. If you're registered for the class, the course is on you Canvas dashboard: canvas.upenn.edu

The F1/10 Reference Manual (August 2018 version) is available on www.f1tenth.org . It has lots of good information. Use it! :)