Teaching Autonomous Systems at $1/10^{th}$-scale: Design of the F1/10 Racecar, Simulators and Curriculum

Abhijeet Agnihotri[*][†]
agnihota@oregonstate.edu
Oregon State University
Corvallis, Oregon

Matthew O'Kelly[*]
mokelly@seas.upenn.edu
University of Pennsylvania
Philadelphia, Pennsylvania

Rahul Mangharam
rahulm@seas.upenn.edu
University of Pennsylvania
Philadelphia, Pennsylvania

Houssam Abbas
houssam.abbas@oregonstate.edu
Oregon State University
Corvallis, Oregon

ABSTRACT

Teaching autonomous systems is challenging because it is a rapidly advancing cross-disciplinary field that requires theory to be continually validated on physical platforms. For an autonomous vehicle (AV) to operate correctly, it needs to satisfy safety and performance properties that depend on the operational context and interaction with environmental agents, which can be difficult to anticipate and capture. This paper describes a senior undergraduate level course on the design, programming and racing of $1/10^{th}$-scale autonomous race cars. We explore AV safety and performance concepts at the limits of perception, planning, and control, in a highly interactive and competitive environment. The course includes an ethics-centered design philisophy, which seeks to engage the students in an analysis of ethical and socio-economic implications of autonomous systems. Our hypothesis is that $1/10^{th}$-scale autonomous vehicles sufficiently capture the scaled dynamics, sensing modalities, decision making and risks of real autonomous vehicles, but are a safe and accessible platform to teach the foundations of autonomous systems. We describe the design, deployment and feedback from two offerings of this class for college seniors and graduate students, open-source community development across 36 universities, international racing competitions, student skill enhancement and employability, and recommendations for tailoring it to various settings.

CCS CONCEPTS

• **Computer systems organization** → **Embedded and cyber-physical systems**; *Robotic autonomy*; • **Artificial intelligence** → Robotic planning; • **Robustness** → Safety critical systems.

[*]Both authors contributed equally to this research.
[†]The work was conducted while the author was at Oregon State University.

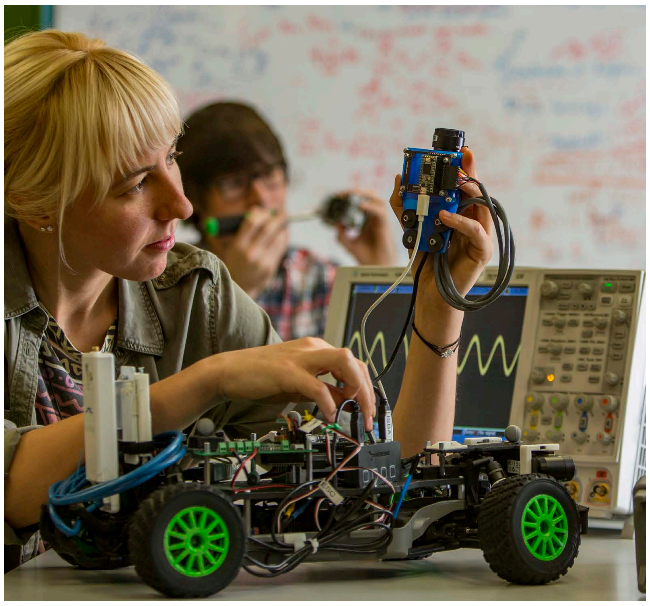


Figure 1: Our 1\10th-scale open-source autonomous vehicle platform enables experience-driven training for perception, planning, control and ethical embodied intelligence

KEYWORDS

Autonomy; Open Curriculum;

1 INTRODUCTION

Autonomous vehicles (AVs) are emerging as integral components of the US economy. The successful development of AV technology has the potential to fundamentally improve the efficiency and safety of the transportation and logistics industry. Enabling such a transformation requires creating reliable, safe autonomous vehicles through the contributions of system builders with a diverse set of engineering, human factors and computer science skills. Traditional treatments of robotics education often emphasize the mechanical, electrical, and computational aspects of system design in isolation. In contrast, autonomous systems, which contain software-based embodied intelligence, should be analyzed via the quality of the decisions and actions of the machine in the world. This paper describes the design of an autonomous systems course based on a $1/10^{th}$-scale open-source autonomous racecar (http://f1tenth.org). The curriculum innovation is centered on the design and deployment of scaled autonomous cars that are accessible, safe to experiment with, easy to customize, can be shared widely, and that can support a wide range of research and education tasks in autonomous vehicles. Prior to developing this course, our goals were:

(1) To design a $1/10^{th}$-scale autonomous racecar for teaching the foundations of perception, planning control and embodied intelligence of real AV operation effectively. This was realized through the operation of AV algorithms and code in the $1/10^{th}$-scale AV simulator and on the physical racecar.
(2) Understand the design principles for safe and responsible decision making in autonomous systems. This will be evaluated by using a risk-based analysis of ethical dilemmas and encoding of moral principles across a variety of driving scenarios.
(3) Build a sustainable community with wide and deep research questions in safe autonomous systems, that cannot otherwise be answered without such a physical platform. This will be evaluated by measuring the increase in number of partners who adopt the platform in their institution.
(4) Develop the technical skills necessary through hands-on laboratory exercises and international racing competitions which implement and deploy core AV algorithms. This will be evaluated by the rate of course graduates gaining employment in the autonomous systems industry.

2 EXISTING EDUCATIONAL MATERIAL

Robotics is often both the entry-point (*e.g.* [5, 9, 16]) and capstone in modern computer science education (*e.g.* [14, 18]). Presentations of advanced concepts in autonomy are fundamentally interdisciplinary and reward maturity in computational and mathematical concepts which form the basis across undergraduate engineering curricula. Still, simple concepts can yield surprising insight and motivation; thus, a variety of related courses have been developed for audiences from high school students to advanced undergraduates.

For high-school level students First Robotics [16] and the MIT Beaverworks programs stand out [9]; these courses primarily focus on software development and simple reactive autonomy concepts. Students do not directly consider the challenges related to underactuated robotic systems, planning, and modern state-estimation.

First year engineering students also often participate in a sequence of introductory courses which includes a robotics component (*e.g* [5]). As with the high-school level initiatives these efforts have well-documented success in attracting and retaining students; however, the material they present is necessarily limited by the background of the students. Advanced concepts such as simultaneous localization and mapping (SLAM), model-predictive control, and computer vision techniques are not addressed, and the robots are simplified both to reduce cost and lessen students' exposure to problem requiring excessive background.

Among the more advanced robotics systems courses, [3, 10] focus on aerial robotics. These curricula cover fundamental concepts such as coordinate transformations and planning; however, the few interactions with real hardware utilize external motion capture

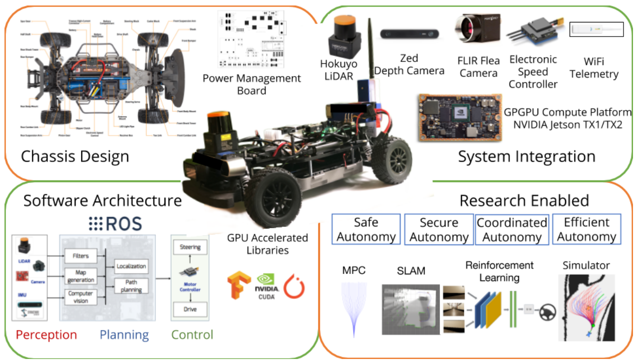


Figure 2: F1/10^{th} is an integrated platform with mechanical design, sensors, power systems, and a full-AV stack. It enables research and education in safe and efficient autonomous systems.

systems, and non-standard robot programming tools, significantly limiting their applicability to industry challenges.

There are several other autonomous vehicle platforms available, ranging from low-cost, simplistic options such as [8, 14] to platforms which exceed \$20,000 [7]. Duckietown [17] is another closely related course which focuses on ground robots; in contrast to the work presented in this paper, the course requires 34 support staff (more than the number of students) and utilizes differential drive robots and an external localization system. The current generation of the F1/10 hardware costs approximately \$2400 and will become cheaper with each design iteration.

The platform developed for [4, 15] is most similar to ours; however, it remains more expensive, less performant due to physical layout, and includes sparse documentation for non-MIT users. While our course is inspired by the excellent work in [4] we specifically aim to lower the barrier to entry by providing comprehensive simulation tools, documentation, course materials, an actively monitored support forum, and a reduced-cost vehicle.

3 CREATING A NEW COURSE

In light of the existing curricula, we decided to design a new course that teaches how to build an *entire* autonomous system, complementing existing courses focused on one element of autonomy. Because car-like robots are one of the simplest non-trivial systems in robotics and a major focus of industry, we use $1/10^{th}$-scale Ackermann steered AVs to have dynamic similitude to full-scale vehicles. Our course design was guided by the following principles:

(1) *Students must get a minimally functioning car as early as possible*, rather than wait until the end of the course. This generates and sustains their excitement and interest.
(2) *The class will be team-based and inter-disciplinary.* The breadth of the course and reality of post-graduate opportunities in the field dictate the need for teamwork, cross-cutting background knowledge, and familiarity with collaboration tools.
(3) *The lectures' focus is on algorithms.* The students will inevitably deal with lower-level concerns like message formats, low-level vehicle control, computation loads, sensor constraints, etc. This is good, and is part of the system integration learning outcomes, but the lectures focus on algorithms.

(4) *The content is mathematically rigorous.* We must avoid the impression that things work by endless parameter tuning. Mathematically sound designs must first be implemented in the $1/10^{th}$-scale AV simulator and then transferred to the car.
(5) *This will be a hands-on class* and lectures serve as preparation for the lab sessions (a.k.a. practice laps).
(6) *Ethical issues arising from embodied autonomy must be covered with equal importance*, as they are not usually covered in a generic class on engineering or professional ethics.
(7) *The class must be accessible to advanced undergraduates working in teams with graduate students*, as they might join the strong job market upon graduation.

3.1 The Hardware Platform

In order to satisfy *principle 1*, the course material includes a new class of high-performance autonomous racing cars, that are $1/10^{th}$-scale of real Formula1 cars and can reach a top speed of 50mph (see Fig. 3.) Each vehicle includes a sensor suite (LIDAR, stereo camera, inertial measurement unit), actuators (electronic speed controller), computation platform (powerful Nvidia Jetson TX2 GPGPU), and on-board power management system. The platform is easy to assemble in 2 hours.

The vehicle utilizes widely available 4-wheel drive chassis based on the Traxxas Rally $1/10^{th}$-scale radio controlled car with Ackermann steering. This platform was selected because it is small enough to run indoors, and its high-performance motivates non-trivial algorithmic challenges.

In order to make the base car suitable for the course several modifications were necessary. First, the stock electronic speed controller is replaced with an open source device which can be tuned to provide closed loop control of motor RPM and actuate a steering servo [19]. Furthermore, recent versions of the Traxxas vehicle utilize a brushed DC motor which we replace with a brushless motor to improve state estimation and performance. Finally, a power distribution board (used to power the onboard compute and sensors) as well as mounting plate design are provided; all aspects of these additional parts are open source and can be fabricated by students or purchased from vendors.

The course hardware includes a modern embedded system on a chip (SoC) which contains a conventional multicore ARM CPU in addition to a powerful GPU: the Nvidia Jetson TX2 or Xavier [5, 6].

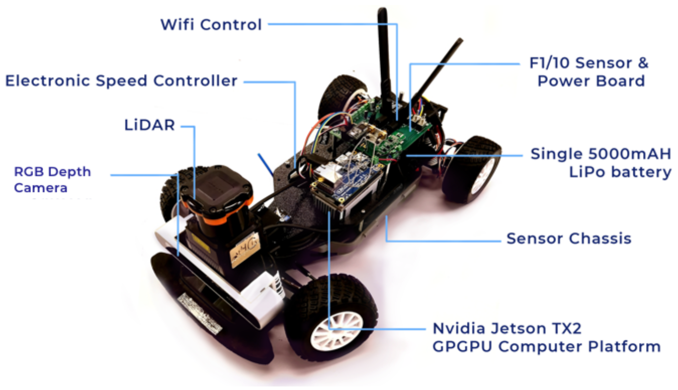

Wifi Control
Electronic Speed Controller
LiDAR
RGB Depth Camera
F1/10 Sensor & Power Board
Single 5000mAH LiPo battery
Sensor Chassis
Nvidia Jetson TX2 GPGPU Computer Platform

Figure 3: $1/10^{th}$-scale AV sensors, actuators and computation

Similar hardware is used in full-scale autonomous vehicles sold today (*e.g.* [12]), thus exposing students to realistic performance constraints. This design decision enables the deployment of machine learning components on the vehicle. The SoC is capable of running a customized Linux kernel with full GUI and GPU support easing students' transition from traditional development tool chains. Detailed instructions for configuring the operating system, hardware interfaces, and software packages are provided in the publicly available 110-page course manual at http://f1tenth.org.

Lastly, the vehicle must have sensors to perceive its operating environment. The main sensor is a planar laser scanner (or LIDAR) which can capture range measurements. The LIDAR enables the vehicle to implement reactive obstacle avoidance strategies, estimate odometry, create maps, and localize. Due to the operating environment (typically corridors with few features) we supplement the LIDAR measurements with odometry information from the electronic speed controller [19]. Optionally, for the semester-version of the course, which includes computer vision, an Intel RealSense RGB-Depth camera provides additional sensing modalities.

3.2 AV Software Stack

Following principles 1 and 3, the skeleton code that each team receives has 2 components:

(1) Utility code
(2) Robot Operating System (ROS)-based skeleton code.

Utility code consists of items such as drivers for the Electronic Speed Controller (ESC), camera and LIDAR. The ROS-based skeleton code contains the interfaces that students directly interact with to link the application layer of a driving strategy with the low level sensor observations and actuation interfaces. As the emphasis of this course is to develop algorithms for autonomous driving, the skeleton code lets teams focus on writing their own logic inside pre-built ROS nodes. This minimizes the time spent handling tedious coding logistics so students can focus on the mathematic ‘and algorithmic basics for perception, planning and control (see Sec. 3.5 and principle 4).

3.3 The $1/10^{th}$-scale Racecar Simulator

In accordance with principles 3, 5, and 7, the course includes a Software-in-loop (SIL) simulator which serves as a replacement for the vehicle and its sensors. Navigation algorithms developed for the car can run inside the simulator without any modification. The response of the vehicle to control commands is captured using the single track model described in [1]. Synthetic sensors mimic odometry information from the ESC and laser scans from the Hokuyo LIDAR. The same message types and ROS publisher-subscriber mechanisms are used to communicate the sensor observations enabling SIL testing. The SIL-oriented simulator uses RViz, the built-in ROS visualization tool.

This simulator is used in the course as a rigorous testing sandbox for the students, prior to racing their vehicles. Students can use different maps in simulation, and obstacles can be added and removed in real time to test the code they generate for each module of the course. With the use of a Docker container for the simulator and accompanying development environment can be deployed on most Linux, MacOS, and Windows-based computers [2, 13]. This eases

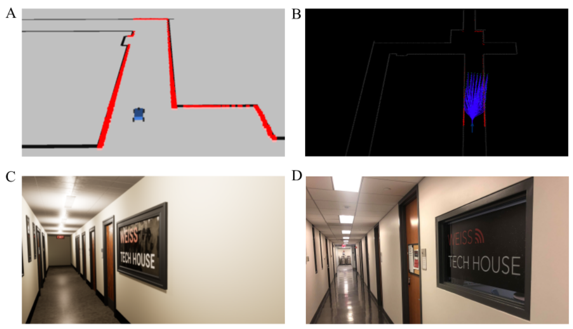


Figure 4: F1/10 simulators: (A) Software-in-Loop sim used for course (B) Cloud-based Open-AI sim for machine learning research (C) Computer vision sim (D) Reality

the transition from traditional desktop development environments to the embedded system on-board the vehicle. While working with the hardware is always preferable, the simulator is an adequate replacement that enables individuals without access to the hardware to complete the assignments.

3.4 Team Formation

Following principles 2 and 7, the course was open to both undergraduates and graduate students pursuing degrees in electrical and computer engineering, computer science, mechanical engineering or robotics. Teams of 3 to 4 students were formed while keeping a balance of undergraduate and graduate students, and a balance of majors, in each team.

Class pre-requisites were *per-team*. Namely, every team needed to have strong programming skills, and working knowledge of ordinary differential equations and numerical linear algebra. Knowledge of ROS, the Laplace transform, Kalman filtering, and optimization were an advantage. Thus on a team, the strong coder did not necessarily know Bayesian filtering and vice versa. We did expect all students to contribute to the code, and tested for that in the labs.

One of the course objectives was to introduce all students, regardless of background, to the entire autonomy stack. It was a challenge to balance of topics between perception, planning, and control. While computer science (CS) students were generally more familiar with perception (e.g., computer vision) and software engineering elements; Electrical (EE) and Mechanical Engineering (MechE) students are typically more familiar with planning and control. CS students generally had more programming experience than MechEs, while EEs span the range. To this effect, we opted for a sequence starting with perception, then planning, and finally control, as shown in Table 1 (note that some topics are multi-week).

3.5 Course Modules

Table 1 lists the course modules of the semester-long version, and which modules were retained in the quarter-long version. They are divided into Perception, Planning, and Control, in accordance with the major categories of algorithms running on an autonomous robot. In place of exams, we conducted three races during the 6th, 10th and 15th week. For the first race, teams used reactive methods for wall-following and obstacle avoidance with the real-time LiDAR data. For the second race, teams use map-based methods with SLAM and pure-pursuit navigation. For the third race, teams developed advanced planners and controllers for optimized racing and overtaking strategies. With each race, the vehicle speed and agility increased significantly.

3.6 Lectures for Lab Preparation

The course has eight labs and the lectures are used to provide the mathematical and algorithmic background for the lab exercises. The lesson plan balances the teaching of algorithms which the students implement from scratch (within the skeletons), and the use of software they can download as ROS packages. As an example, building a *reliable* race-worthy car in 10-15 weeks requires the use of advanced perception techniques, such as SLAM (Simultaneous Localization And Mapping). Modern implementations of SLAM are infeasible for most students to develop in two weeks. Initially, this appears incongruous with the goal of developing a systems level understanding of autonomy.

The course aimed to strike a balance in the modules which generate this tension of learning from scratch or using an off-the-shelf optimized package. First, lectures included substantial mathematical exercises in the topics of control theory, rigid body transformations, computer vision, and scan matching. As teams demonstrated their understanding of the core material, industrial-quality implementations of the algorithm were made available.

For example, *the scan matching algorithm*, a procedure that computes the car's movement by aligning consecutive LIDAR measurements, is a key component of most modern SLAM implementations. Thus, two lectures are dedicated to the motivation, mathematical formalization, and solution, of the scan matching problem. The students then implement the conceptual lecture material on their cars from scratch. Because scan matching utilizes concepts from linear algebra and optimization that recur within future material we also include rigorous mathematical exercises following *principle 4*. Finally, using this experience as a basis, we contrast their implementation withs state-of-the-art approaches. Only then, is the use of a state-of-the-art SLAM package allowed.

3.7 Labs on AV Foundations

The eight lab exercises are a key-component of the experiential learning presented in the curriculum presented in this paper. The lab exercises typically first require the students to develop their algorithms in the simulation environment. The systems engineering component and parameter tuning aspects are particularly easy to assess and improve upon in the simulator. To emphasize the need for rigor, the more substantial labs required a 2-page write-up in which the teams explain the rationale for their algorithmic solutions and the process by which they selected parameters.

While the ROS assignments were team-based in the first offering of the class, we made them individual in the 2nd offering. This encouraged *all* team members to contribute code during the class, and avoided having the expert coders becoming the only coders on the team, as happened on a couple of teams in the first offering.

3.8 Ethics in autonomy research

Human-scale autonomy raises thorny ethical issues. Some of the best-known examples involve 'trolley problems' (should the AV injure its passengers or pedestrians when given no other choice?),

Table 1: Course modules for 15-week class, and which were retained for quarter-long version (N/Y = only lab was retained)

Category	Lectures	Corresponding Practice Lab	Retained?
AV Basics	Selection of teams and introduction	ROS tutorials and Monitor node creation	Y
AV Basics	ROS basics	Assembling the car and manual control	N/Y
Perception	1/10th-scale Hardware	LiDAR Gap finding (sim and car)	Y
Perception	Pose transformations and ROS /tf	–	Y
Perception	Localization: Scan matching	Scan matching (sim and car)	Y
Perception	SLAM and state-of-the-art packages	Run SLAM and localization on car	**N/Y**
Control	Basics of Laplace analysis and PID	ESC tuning and Wall-following (sim and car)	Y
Planning	Pure Pursuit	Implement Pure pursuit	Y
Planning	–	Mid-semester race	**N**
Ethics	Moral decision making	Readings and response essays	Y
Control	Model-Predictive Control	Implementation in simulator	Y
Vision	Computer Vision and near-pose estimation	Two algorithms in simulation	Y
Machine Learning	Reinforcement Learning	Behavioral cloning of human driver (sim and car)	Y
Special topics	Industry case studies	–	**N**
	–	Final Grand Prix	Y

but they extend to more fundamental and broader issues: (1) Who or what is morally responsible for the decisions made by an autonomous system? (2) How can responsibility be programmed with Robot Safety laws? (3) How do context and function affect the question of designing ethical behavior? The students were asked to read essays from moral philosophers and technologists (e.g., from [11]), and discuss them in groups. Then the class met for 6 hours of discussions guided by the instructor, after which every student wrote two response essays, one to each set of readings. The desired outcome is a student reasonably equipped to tackle AS ethical issues on the job because they have thought about them *prior* and know how to further develop their analysis.

4 DEPLOYMENT, TUNING, AND EVALUATION

We offered the class thrice: the first and third time at an Ivy League university on a semester schedule (15 weeks), and the second time at a large public university on a quarter schedule (10 weeks). Because we had hardware for four teams only, the first offering enrolled 16 students (2 undergrads and 14 grads, from Robotics and CS); the second offering enrolled 13 students (7 advanced undergrads and 6 grads, from CS, ECE and MechE); and the third offering included 18 students (4 undergrads and 14 grads from MechE, Robotics and EE). This was a popular course with 3X more students on the waitlist.

Because of the small number of students in each class, and the fact that we evolved the design in response to first offering experiences, we felt that quantitative studies would be statistically meaningless at best, and misleading at worst. Instead, we resorted to rich one-on-one conversations with the students halfway through the course and after the grades had been finalized. We also asked the same questions through an anonymous online form to students who, for whatever reasons, did not want to or could not have a face-to-face conversation. Finally, the end-of-term course evaluations were collected and reported on below where relevant.

4.1 Evaluations

All students who enrolled, completed the course. For the first two offerings, 62.5% of respondents found the balance of topics between Perception, Planning and Control adequate. However, about 25% reported that tuning the car's performance took too much time away from improving their algorithms. This was a foreseen concern when designing the course: the real world provides noisy data, slippery driving surfaces and a changing environment. One of the class' objectives is to demonstrate the limits of parameter tweaking and the need for rigor, and in that sense the students' feedback confirms that the lesson was learned. However, the feedback also suggested to further constrain experimentation and provide avenues for moving past that stage.

Close to 75% of respondents considered the lab reports were beneficial, albeit in different and sometimes unexpected ways: one student found the reports were "important for collaboration. If there was no report, it would be too easy for each person to work individually [...] The reports force you to come together and understand the big picture.". Another used the writing to "put effort into the formal derivation of the algorithms". For yet another student, the reports were "a good place to [...] make sure what we did made sense". Importantly, "the reports seemed "casual" enough to make us not worry too much about how we wrote them and just make sure what we were writing made sense."

All respondents valued the interdisciplinary constitution of the teams. E.g. one responded that "It was essential - we would not have succeeded without someone who had a background in mechanical engineering and other physical sciences". Conversely, "Programming is my weakness, so it was great having CS teammates willing to answer questions and help me troubleshoot." Finally and importantly, communication did not seem to be an issue: "our team had 1 MechE, 2 ECE, and 1 CS. There was no difficulty communicating across disciplines ".

The CS students did struggle with the mathematical exercises, especially on Laplace transforms. Others thought they were disconnected from that week's programming assignment. Thus, there is a need to better tie in the mathematics to the implemented algorithms. Most students in the 10-week class did feel pressed for time. At this point we feel the PID module can be further reduced in a 10-week offering, e.g., by dropping the ESC tuning lab.

The lecture sessions: While the course spans a range of topics, students felt the lectures provide both a basis for understanding the topic and were sufficiently deep to guide them through the lab exercises. While students found the implementation of algorithms from scratch or from a research paper challenging, they were satisfied with the time given to complete the labs. Overall, students found that several of the lecture topics covered job interview questions they had been asked recently.

The lab sessions: Students from the first offering expressed that some lab sessions were not informative because they received little feedback (outside of the grading rubric) before moving on to the next lab. To remedy this, in the second (quarter-long) offering, the first hour of the three hour lab session was dedicated to team demonstrations of their past and existing results. Teams were then advised by all other teams and the instructors on their progress and roadblocks were discussed and shared.

4.2 Feedback to in-class ethics discussions

The student response to the ethics discussions was overwhelmingly positive. In our face-to-face conversations, several expressed the wish that they had been invited to think of these matters in other classes as well. Some appreciated having a theoretical framework (from moral philosophy) in which to turn their instincts into an analysis. One student related their own personal experience interning in a military lab and pondering the ethicality of the technology they were developing. The response essays we received varied in focus and depth. A majority veered towards generalities rather than engage more closely with the readings. In a second offering we would reduce the number of readings from 6 to at most 2, and dedicate more time to a close reading of the essays.

4.3 Course Learning Outcomes

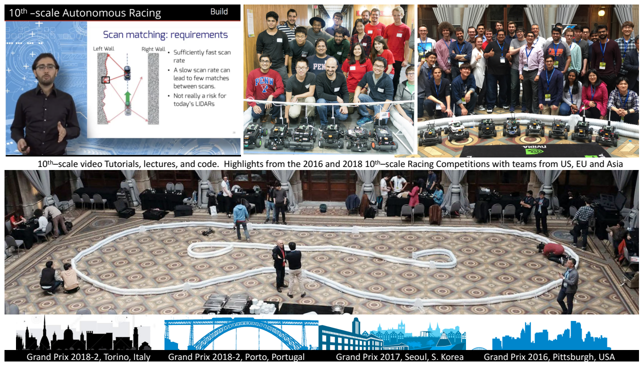
In the third offering, we introduced the F1/10 simulator. On average, teams spent over 80% of their development time in simulation and only 20% with tuning the platform and to capture maps. This demonstrated a significant ease-of-use in completing the same labs with less frustration through development iterations and testing. As the accuracy of the vehicle model in the simulator is relatively good, bridging the simulation to reality gap required little effort.

4.4 Limitations

This course is limited in its depth of exploring individual modules of autonomous driving. While there is a whole class of algorithms for mapping an unknown environment, we stick to discussing these approaches while using an existing package for the purpose to race autonomously. To mitigate this challenge, one approach could be to follow up with an advance version of this course, where students who have already taken this course will get to implement advance algorithms on the cars. Current cost of cars is around \$2400 and puts a limit on the class size. We hope, by open sourcing the car's design, it will increase adoption of the course in other institutions and help in optimizing the cost of the updated car design.

5 COMMUNITY ADOPTION

A major motivation for the creation of the course was to build a community around teaching and research in autonomous systems. We have thus made most of the material open-source at http://f1tenth.org and it has been taught in several other institutions. A typical usage is to offer a subset of the material as a module


10th –scale Autonomous Racing
Build
Scan matching: requirements
Left Wall
Right Wall
Sufficiently fast scan rate
A slow scan rate can lead to few matches between scans.
Not really a risk for today's LIDARs
10th–scale video Tutorials, lectures, and code. Highlights from the 2016 and 2018 10th–scale Racing Competitions with teams from US, EU and Asia
Grand Prix 2018-2, Torino, Italy
Grand Prix 2018-2, Porto, Portugal
Grand Prix 2017, Seoul, S. Korea
Grand Prix 2016, Pittsburgh, USA
Figure 5: F1/10 community outreach and competitions

inside a class on Real-Time Systems (as done at UT Austin) or Automotive Systems (as done at Clemson University). An undergraduate semester-long version of the class was also developed at the University of Virginia, with a much longer introduction to ROS (6 weeks vs 1 week in the semester-long version and none at all in the quarter-long version). Another typical usage of the material is for self-study as part of an Independent Study course, or by students who plan to compete in the F1/10 International Grand Prix. The F1/10 Grand Prix, organized bi-annually, are an important vector for spreading the material and for building the community of F1/10 users and contributors. Over the past three years, we have conducted 5 Grand Prix at tier-1 conferences (in Pittsburgh, Porto, Turin, Montreal and New York) based on this platform. In each competition, an average of 10 teams (generally 4 persons each) follow the instructions on the project website to build the same reference design. The community growth has been steady with more than 36 domestic and 22 international university and small company groups who have adopted the platform for their AV R&D efforts.

The community has contributed to the F1/10 AV stack by demonstrating different designs and algorithms in each race. For example, gap-finding, one of the modules in the course, was first demonstrated by a competitor in the Porto Grand Prix. Pure pursuit, another class module, was demonstrated in the Turin Grand Prix.

6 CONCLUSIONS

This paper described the design of an autonomous systems course and community based on a $1/10^{th}$-scale open-source autonomous racecar platform for project-based training in the fundamentals of perception, planning, control and ethical embodied intelligence. We started with four hypotheses and after three course offerings we have the following findings: (1) The $1/10^{th}$-scale platform was able to seamlessly operate full-scale AV code and algorithms, making it a low-cost, available and safe platform to learn and test AV algorithms. (2) Through the $1/10^{th}$-scale simulator and platform, teams encoded moral constraints in their implementations and this provided a good foundation for discussions on AV ethics and risks. (3) A growing $1/10^{th}$-scale racecar community exists and is sustained by two annual Autonomous Racing Competitions. (4) Students who participated in the course obtained employment in industry at Tesla, NVIDIA, Honda, Aurora, Nuro, Rivian, Zoox, Built Robotics, and Drone Racing League.

REFERENCES

[1] Matthias Althoff, Markus Koschi, and Stefanie Manzinger. 2017. CommonRoad: Composable benchmarks for motion planning on roads. In *2017 IEEE Intelligent Vehicles Symposium (IV)*. IEEE, 719–726.

[2] Carl Boettiger. 2015. An introduction to Docker for reproducible research. *ACM SIGOPS Operating Systems Review* 49, 1 (2015), 71–79.

[3] K Daniilidis and J Shi. 2019. Robotics: Perception. https://www.coursera.org/learn/robotics-perception

[4] Andrew Fishberg, Trevor Henderson, Ken Gregson, Zachary Dodds, Don Krawciw, Christine Nicholls, and Sertac Karaman. 2019. Autonomous RACECAR: 1/10-scale Autonomous Car Platform for Research and Education in Robotics. In *2019 IEEE International Conference on Robotics and Automation (ICRA) Tutorial*. IEEE.

[5] DJ Frank, KL Kolotka, AH Phillips, M Schulz, C Rigney, AB Drown, R Stricko, K Harper, and R Freuler. 2017. Developing and Improving a Multi-Element First-Year Engineering Cornerstone Autonomous Robotics Design Project. In *Proceedings of the 2017 American Society for Engineering Education Annual Conference*.

[6] Dustin Franklin. 2017. NVIDIA Jetson TX2 Delivers Twice the Intelligence to the Edge. *NVIDIA Accelerated Computing| Parallel Forall* (2017).

[7] Brian Goldfain, Paul Drews, Changxi You, Matthew Barulic, Orlin Velev, Panagiotis Tsiotras, and James M Rehg. 2019. AutoRally: An Open Platform for Aggressive Autonomous Driving. *IEEE Control Systems Magazine* 39, 1 (2019), 26–55.

[8] Eloy Irigoyen, Ekaitz Larzabal, and Rafael Priego. 2013. Low-cost platforms used in control education: An educational case study. *IFAC Proceedings Volumes* 46, 17 (2013), 256–261.

[9] Sertac Karaman, Ariel Anders, Michael Boulet, Jane Connor, Kenneth Gregson, Winter Guerra, Owen Guldner, Mubarik Mohamoud, Brian Plancher, Robert Shin, et al. 2017. Project-based, collaborative, algorithmic robotics for high school students: Programming self-driving race cars at MIT. In *2017 IEEE Integrated STEM Education Conference (ISEC)*. IEEE, 195–203.

[10] V Kumar. 2019. Robotics: Aerial Robotics. https://www.coursera.org/learn/robotics-flight/

[11] Patrick Lin, Keith Abney, and George A Bekey (Eds.). 2014. *Robot Ethics: The Ethical and Social Implications of Robotics*.

[12] Hayk Martirosyan, Adam Bry, Matthew Donahoe, Abraham Bachrach, and Justin Michael Sadowski. 2018. Unmanned aerial image capture platform. US Patent App. 15/235,513.

[13] University of Pennsylvania. 2020. F1tenth Simulator Docker. https://hub.docker.com/r/f110penn/f110sim

[14] Liam Paull, Jacopo Tani, Heejin Ahn, Javier Alonso-Mora, Luca Carlone, Michal Cap, Yu Fan Chen, Changhyun Choi, Jeff Dusek, Yajun Fang, et al. 2017. Duckietown: an open, inexpensive and flexible platform for autonomy education and research. In *2017 IEEE International Conference on Robotics and Automation (ICRA)*. IEEE, 1497–1504.

[15] Marco Pavone. [n. d.]. AA 274: Principles of Robotic Autonomy. http://asl.stanford.edu/aa274/

[16] Natalie Rusk, Mitchel Resnick, Robbie Berg, and Margaret Pezalla-Granlund. 2008. New pathways into robotics: Strategies for broadening participation. *Journal of Science Education and Technology* 17, 1 (2008), 59–69.

[17] Jacopo Tani, Liam Paull, Maria T Zuber, Daniela Rus, Jonathan How, John Leonard, and Andrea Censi. 2016. Duckietown: an innovative way to teach autonomy. In *International Conference EduRobotics 2016*. Springer, 104–121.

[18] Sebastian Thrun. 2019. Self Driving Car Engineer Nanodegree. https://www.udacity.com/course/self-driving-car-engineer-nanodegree--nd013

[19] Benjamin Vedder. [n. d.]. Vedder Electronic Speed Controller. https://vesc-project.com/documentation