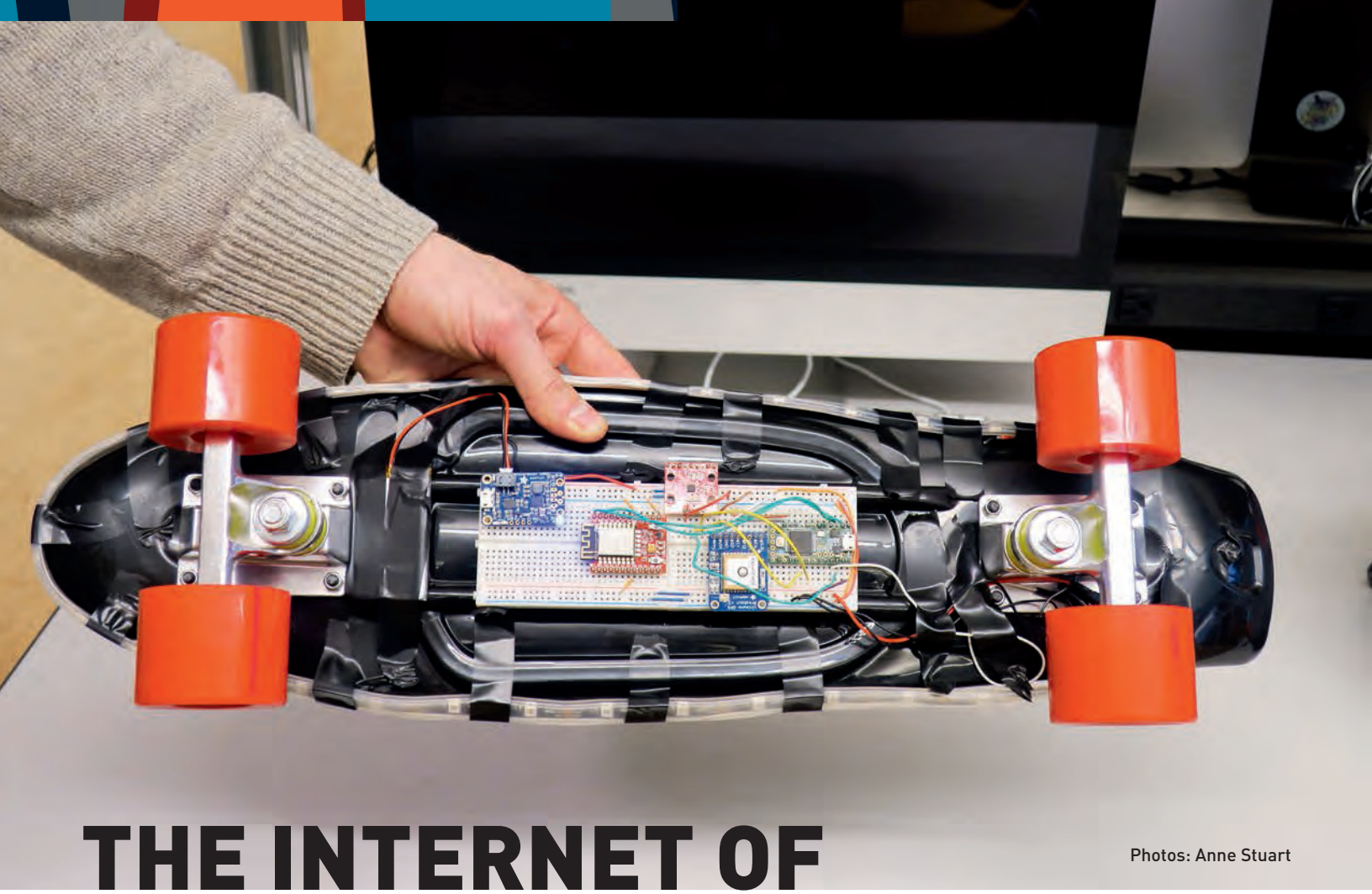


Photos: Anne Stuart

THE INTERNET OF (PLAY) THINGS

In two popular courses, students learn how to program mobile sensors — and savor the connections they make.

By Alison F. Takemura | EECS

Invisible messages fly through the room. With laptops open and microcontrollers at the ready, students are learning how to code programs that encrypt their messages, typed on the microcontrollers called Teensies, so that only their lab partners can read them.

"It's like spy stuff," says freshman Brandon Kramer on an April afternoon. Except for one thing: they're not exactly dishing in state secrets. His lab partner, sophomore Abby Bertics, divulges that the main topic of their messages has been food.

Kramer and Bertics are taking the introductory course 6.S08 (Interconnected Embedded Systems). Only in its second year, the course is already extremely popular. Last year, 160 students pre-registered, and the instructors ran a lottery to accept 40, said Joel Voldman, professor of Electrical Engineering and Computer Science (EECS). This year 240 students signed-up, but the course could only enroll 180.

Voldman believes the course's appeal stems in part from its hands-on approach. Week by week, students learn how to piece together the same interconnected components you'd find in a Fitbit: microcontrollers, gyroscopes, accelerometers, magnetometers, Wi-Fi chips, and so on.

"I wish I had had something like this when I was a freshman," says Joe Steinmeyer, an EECS lecturer and co-creator of the course. "There's so much stuff out there now. You can go on the Web and just buy these really amazing parts, like a GPS unit or Wi-Fi unit — and you can do a lot with them. But I think many students coming in don't know where to begin."

This course and a more advanced one (6.S062, Mobile and Sensor Computing), which also started last year, both help make students more familiar with using and integrating wirelessly connected devices — the so-called "Internet of Things." The emphases, however, are different. The introductory 6.S08 focuses on building a device, which entails

putting together hardware and controlling it by programming in languages C++ and Python. The advanced 6.S062 allows students to build more sophisticated software that harnesses data from wireless sensors, such as mobile phones, for different applications. In both courses, students receive the opportunity to apply the skills they learn from lectures and lab exercises to a final, open-ended project. Some submissions are delights, and others, significant novel contributions.

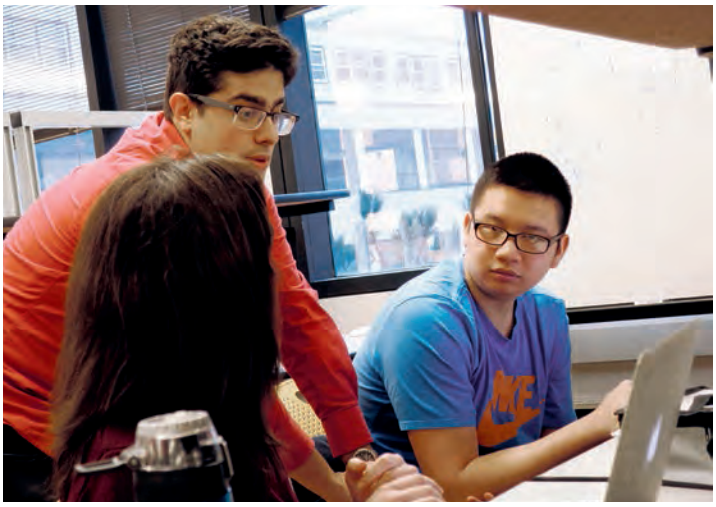
6.S08: Embedded fun

"Before the class, I had no idea how to wire anything," said Jenny Xu, a sophomore who took the course last year and is now a lab assistant (LA). She loves developing games in her own free time, so, in her final project, she and a partner put together the hardware and software for their own multi-player trivia game. All players could buzz in on their own consoles, the augmented Teensies used in the class, and scores would appear on a leaderboard. It was a rewarding challenge to make a game she could play with friends, Xu says.

Last year, sophomore Kenneth Collins and a partner created a device to answer a proverbial college-student question: Where can you find free food? The device interacted with a server that would go to your Gmail and look at free-food e-mails," he says. Now an LA, he wants to go a little further with the project, possibly putting it on display in his dormitory, as something of a community service.

One of the instructors' favorite projects from last year was a skateboard that could give its rider directions. Through use of GPS, the skateboard's sides would light up to indicate which way to turn at intersections. The board, which the instructors still have, glows beautifully with blue LEDs.

Completing a final project is a winning feature of the course for Carissa Gadson, a sophomore and LA. "A lot of freshmen asked me, 'Should I should take it? Is it worth it?' I always say yes, because [at the end] you have something technical that you can explain really well." Recruiters were surprised that she had worked with databases and servers, along with C++ and Python, she says. "I think that really helped with getting my internship last summer."



"How do you build a network of devices that communicate with each other? Sensors may need to last for months at a time in a remote environment, how can you ensure their batteries don't run down really fast?"

—EECS Professor Sam Madden, cofounder of 6.S062.

Voldman wants the course to empower students for situations ranging from Undergraduate Research Opportunities Program projects to issues that might come up at work. "We'd like them to have the confidence that, if they approach a UROP or job and their employer says, 'We have this equipment, and we need you to put an embedded system around it to control it or get the data off it,' they feel like, 'Yeah, I can do that,'" he says.

"The field is changing so fast," Steinmeyer adds. "A lot of future engineering is going to be engineering across multiple systems." It's not going to be just one clean sandbox language like Python, for example, but Python, Javascript, and HTML, all mashed together, he says: "We're really trying to give students experience integrating across multiple environments."

At the time of this writing, current students haven't gotten to the projects yet, but they're still enthused. "There aren't many classes where you're just building cool things all the time," Bertics says. "It's my favorite class this semester."

6.S062: The software side

Geared toward juniors and seniors, 6.S062 looks at how to use mobile sensors from a higher level than 6.S08: through software.

Software is what allows users to extract data from local and remote sensors — increasingly ubiquitous technologies. "So much of what's happening in the world today involves sensing and phones and mobile devices," says Sam Madden, professor of EECS and course co-founder. "There really are some specialized techniques that people should know if they're working in those environments." One technique is the Viterbi algorithm to solve a Hidden Markov model, which can be applied to a seemingly simple problem: how to figure out which road a car travelled along, using only a trace of GPS signal collected from the car's driver.

"This problem can actually be very challenging," Madden says. He flips open his laptop and pulls up a video of red dots moving along streets in Boston. Each dot is a GPS signal, most likely coming from a phone. "You squint at it, and you're like, 'Oh yeah, these match onto the roads pretty well.' But then you see there's weird stuff." He zooms in and dots are sometimes on sidewalks or in the Charles River. Some areas have too little data to tell whether there's actually a road there or whether the results are being confounded by a rogue pedestrian.

When Madden changes the video's focus to the Massachusetts Turnpike running beneath Boston's Back Bay neighborhood, the GPS dots drop out — because GPS doesn't work indoors or underground. And then in Copley Square, the dots get rowdy as GPS signals reflect off the Hancock Tower.

Road mapping this kind of real data is one of many problems students can tackle in the class, Madden says. "There's multi hop communication. How do you build a network of devices that communicate with each other? Sensors may need to last for months at a time in a remote environment, how can you ensure their batteries don't run down really fast?"

Tackling these problems gives students a foundation to build final projects of their own design — a satisfying experience, says Natasha Consul, a senior who took the course in 2016. "When you have that freedom and independence, that's when you learn the most," she says.

Three students — Geronimo Mirano, now a master's of engineering (MEng) student, and Eric Lau and Harihar Subramanyam, who have both since graduated — solved a tricky problem for their final project. GPS works well outdoors, but because it doesn't penetrate walls, the team created a solution for indoor travel, such as in a mall or a museum. Their software used noisy video data (from stationary surveillance cameras) and inertial data (from an individual's phone) to help people find their bearings. Madden and course co-founder Hari Balakrishnan, the Fujitsu Professor of Electrical Engineering and Computer Science, call those results exciting. "We encouraged them to publish," Madden says.

Mirano has been too busy with his master's work to pursue a paper at the moment. But the class left a lasting impression. It showed him there were "really cool problems to be found in these technologies," and that the same issues pop up in a variety of systems, he says. For example, Fitbit and Google maps both grapple with inertial data to figure out trajectories. The robots he programs in his research must similarly extract useful, low-bandwidth information from noisy, high-bandwidth sensors, he says, adding: "It's all connected."

New Computer Science Minor Attracts Students from Across the Institute

Sixty-six students from 14 MIT departments have declared that they will minor in computer science, taking advantage of a new offering that debuted this past fall.

Designed to enable students to earn credentials in computer science while they pursue other majors, the computer science minor ensures that graduates learn the fundamentals of programming, algorithms, and discrete mathematics. To complete the minor, students must take six subjects in Course 6, including four required courses in the fundamentals and two electives, at least one of which must be at an advanced level, typically within either artificial intelligence and/or theoretical computer science.

"The six courses in the minor program provide a thorough-going introduction to computer science," says Chris Terman, the undergraduate officer for MIT's Department of Electrical Engineering and Computer Science (EECS).

Terman noted that before the minor was developed, the only way to earn computer science credentials at MIT was to major in 6-2 (Electrical Engineering and Computer Science), 6-3 (Computer Science and Engineering), 6-7 (Computer Science and Molecular Biology) or 18C (Mathematics with Computer Science). The Minor in Computer Science is open to all undergraduates except those in courses 6-1 (Electrical Science and Engineering), 6-2, 6-3, 6-7, 7 (Biology), and 18C.

Students enrolled in the new minor thus far hail from a wide range of departments at MIT, including Aeronautics and Astronautics, Civil and Environmental Engineering, Mechanical Engineering, Mathematics, Economics, and Management.

As of spring 2017, there were 27 sophomores, 26 juniors, and 12 seniors in the degree program. However, more people may actually be pursuing the minor, because students may wait until their senior year to declare a minor, Terman notes.

Overall, the launch of the new minor has been a success, Terman says: "Use of online advising and progress-monitoring has helped keep the administrative burden low, so the minor program is serving a new cadre of students without adding substantially to the department's advising and teaching load."

For more information on the new computer science minor, visit eecs.mit.edu/csminor.