J.D. ZAMFIRESCU-PEREIRA • TEACHING STATEMENT

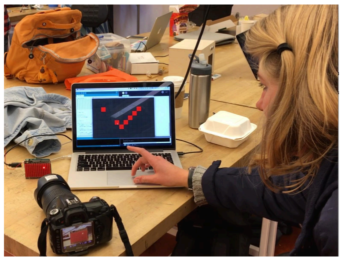
I am excited and qualified to teach and mentor students in HCI, its intersections with AI, NLP, and Design, and broadly across computing and information.

I love to teach hands-on, project-based courses, drawing out and helping channel students' curiosity, creativity, and their own full selves into their projects. I spent a decade developing (and teaching) an introductory programming sequence for artists and designers at California College of the Arts, where I spearheaded a new minor in Computational Practices.

Berkeley undergraduates Jefrey N. and Chelsey C. use servomotors and a distance sensor to make an environment-aware walking robot.

CLASSROOM & MENTORING EXPERIENCE

For over two decades, I have taught programming and electronics, interaction design, and physical computing courses to students from a broad diversity of backgrounds. With my support, my students have made teleoperated robots, electric guitars with lasers for "strings", rotary phones for getting advice from ChatGPT, touch-reactive textiles, contributed to works at the Venice Biennale, and founded startups that have raised millions, including one I now consult for that is saving lives in hospital operating rooms. I've taught at the undergraduate and masters level, in classes ranging from 15 to 1200+ students, at MIT, Cornell Tech, Berkeley, and California College of the Arts—as a TA (MIT, Cornell, Berkeley), lecturer (Berkeley, CCA), and professor of the practice (CCA). I receive consistently high ratings on evaluations in the courses I run. At CCA, I consistently received evaluations in the range of 3.8-3.9+ out of 4.0 (dept. avg. 3.6/4.0), and one representative student comment is "J.D. is easily one of the best teachers I have had at this school." At Berkeley's Jacobs Institute, I averaged 6.4 / 7 (dept. avg. 6.2); students find me approachable, meeting them where they are, and supportive of experimental ideas: "JD is a fantastic instructor, I would totally take more classes from him", "I felt like I could always ask questions and would not be talked down upon", and "The instructor was open to more experimental ideas than other engineering courses I have taken."


CCA undergraduate Bibi B. shows a long-exposure photograph of a "slow screen" that stochastically illuminates only one pixel at any given time.

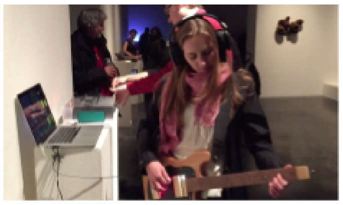
During my PhD at Berkeley, I have also mentored three junior PhD, three Master's, and thirteen undergraduate students, leading to six publications with mentees and two more currently under review. My undergraduate and Master's mentees have gone on to grad programs at Berkeley and Stanford, and landed jobs at established companies and exciting startups.

I draw on my research wherever I can to support student learning. For example, my work on the 61A-Bot, an LLM-powered homework helper for Berkeley's largest introductory CS class, draws on my LLM research—and has shifted how that course is taught: students complete homework with less unproductive struggle, letting TAs focus on concept learning with less time spent debugging.

CCA undergraduate Rob C. uses an Arduino and touch sensors to make a "synthesized" sax.

TEACHING PHILOSOPHY & APPROACH

My teaching is grounded in three main concepts: active learning, hands-on projects, and developing skills for self-directed learning. I aim to create inclusive course environments that address a broad diversity of backgrounds and readiness across prerequisite topics—through "challenge" problems for advanced students, and grounding in real-world questions and concerns. I introduce core concepts via in-class demonstrations and workshops coupled with my own purpose-built software (e.g., Rudy, 61A-Bot), and then to let students drive deeper knowledge acquisition through projects—my students have thrived when engaged in a project they feel a strong internal motivation to pursue. Asa Hillis, a furniture design student in my "Interface" physical computing class, brought twin passions for woodworking and music to create a laser air guitar (right). Comfortable with the basics of code and driven to make the lasers work, Asa dug deep into coordinating the thresholds for light sensors with Max/MSP's sound synthesis—a basic knowledge and a deep dive into a single, intrinsically-motivating topic.


CCA undergraduate Asa H.'s guest plays his laser air guitar in Interface—a physical computing class.

MENTORING PHILOSOPHY & APPROACH

My mentoring approach is grounded in getting students enough background to be productive, having them contribute to an existing project to gain familiarity with the research process, and finally identifying an area of curiosity and capability for the student and transitioning to a student-led project. Heather Wei, my undergrad mentee, initially contributed to our successful "affordances of prompting LLMs" work. I then learned of our mutual interest in CS education, and with my support she has developed a new pipeline for grounding models of student knowledge acquisition in the raw text of course materials—work she had **accepted as a poster to SIGCSE '25.**

In my experience, students vary widely in their mentorship needs. Heather quickly developed confidence in being self-directed, though she initially applied "classroom standards" of success (i.e., all problems are solvable and not solving them is failure) to research—a standard unachievable for any interesting work, where the whole point is to tackle problems without obvious solutions. Understanding where and how we have the ability to make contributions that are meaningful, and when we do not, is a lesson that applies across many future student career paths. My history in industry helps me speak directly to these and other questions students have about academic and nonacademic paths post-college or -MS—experience that I've brought to my peer PhD students too. Several mentees have confided that our conversations were instrumental in steering them away, or towards, a path they were unsure about. I will bring these experiences to my PhD students as well.

FUTURE COURSES & RESEARCH CONNECTIONS

I bring experience and enthusiasm, and look forward to contributing through new courses, or by co-teaching or joining in a faculty rotation for existing courses. I am prepared to teach:

HCI Research Seminar: Theory + Practice (upper division / graduate course)

A rapid-fire introductory seminar focused on classic and current HCI literature. Reading consists of five or six papers per week. A term-long group project focuses on familiarity with HCI research methods, with a late-breaking-work or demo conference submission-level expectation for PhD students. Different iterations go deep on specific focus areas as student demand permits.

Human-Computer Interaction Design + Implementation (undergraduate)

I have taught interaction design courses on topics ranging from UI Design, Interactive Device Design, and Physical Computing. These courses span topics from needfinding and user research to prototyping with physical computing platforms like Arduino and/or web technologies like React or p5.js, depending on broader curricular needs. Technical competencies include event handling, managing input/output, working with device/AI/frontend APIs (e.g., integrating off-the-shelf ML libraries). Projects focus on developing students' prototyping and user testing capabilities, grounded in goals, scenarios, and values. ChatGPT has led to rapid recent increases in the ability to rapidly prototype across this set of courses.

Entrepreneurship / Startup Studio (upper division / graduate course)

Case studies and best practices around identifying market opportunities, mixing technology-focused and customer-focused approaches to isolate startup-appropriate problems, and venture financing. Strong focus on "pounding the pavement" and prototyping with a purpose. I bring experience spearheading commercial initiatives as well as non-profit, educational, and academic initiatives to this course; not all new technology requires a venture-backed model to succeed, alternative versions of this course could focus on how traditional lean and customer-centric approaches can apply outside of business too.

Introduction to Computing (undergraduate)

I have deep training in foundational CS and really enjoy teaching these courses. I have many ideas about how to teach these topics in an LLM-code-synthesis world—and have already contributed to CS 61A through my research in chatbots for student assistance. In particular, reorienting these courses around understanding code rather than writing it, making judicious use of code synthesis models to support students' conceptual development through personalized, interest-oriented challenges and projects, bespoke walkthroughs of parallel problems, and special-purpose "canned solution" training supporting students' recognition of sound design practices.

Computational Cognitive Support for Design (upper division / graduate course)

This research seminar, informed by my own research, would explore the intersection of AI, cognition, and creative processes, including research, engineering and scientific discovery. Covers the recent literature on how computational tools can augment computing broadly, with implications for HCI and across CS. Includes deep reading of foundational psychology and communications (including Clark and others) and enough NLP to be dangerous (datasets, training and model architectures, search strategies).