Teaching Statement

Robert LiKamWa, roblkw@rice.edu

My teaching philosophy is that learning is only effective when students "connect the dots" for themselves. The proverbial dots refer not only to textbook concepts, but also to real-world applications and research adventures. My teaching goal, then, is to provide academic concepts and engineering projects in a context beyond the classroom. Applied to instruction, this philosophy manifests in three pedagogical principles: (*i*) build hands-on experiences into the curriculum through practical application; (*ii*) illuminate relationships between concepts, encouraging students to bridge connections between topics, and (*iii*) illustrate the history of the course material, leaning on motivations that lead to the formulation on adoption of engineering concepts. By applying these three principles, I hope to encourage students to pursue research opportunities, industrial internships, and outside projects to integrate into the engineering world.

Curriculum Development

Depending on the curricular need, I am interested in teaching several core courses in Computer Science and Computer Engineering, including Operating Systems, Computer Architecture, Digital Logic, Data Structures, Algorithms, Computer Vision, and Embedded Systems.

I intend to update the standard Computer Architecture curriculum to the technological and economic push towards System-on-Chip (SoC) integration. Led by its prominence in mobile systems, SoC adoption has changed fundamental principles of Computer Architecture [1], especially for memory and cache architectures. While multi-processor architectures traditionally adopt a unifying L3 cache, this is replaced with a memory-mapped Network-on-Chip (NoC) interconnect subsystem for SoCs. This in turn has forced a redistribution of the memory hierarchy; a larger L2 cache becomes tightly-coupled across homogeneous CPUs for improved performance and efficiency. Furthermore, due to the NoC latency implications of fetching data due to a missed cache access, recent designs have adopted hardware managed coherency between L2 caches of CPU and GPU cores to promote cache hits. SoC-FPGA experimental platforms, such as Xilinx Zynq boards, will provide hands-on experience studying these and other trends of modern computer architecture.

Similarly, the swift advancement of mobile operating systems creates a need to update Operating Systems curriculum [2]. I will include course content for power management mechanisms, covering OS abstractions, e.g., clock frequency scaling, and program abstractions, e.g., power state access control. I will also include recent developments in mobile service frameworks, which provide secure, efficient library service through inter-process communication. The course will also cover mobile I/O interfaces through device drivers and device trees. Building OS exercises on the Android Open Source Project, as inspired by [3], will allow students to investigate the industrial implementation of modern operating systems concepts.

In addition, I would be interested in developing and teaching the following systems courses:

System Support for Computer Vision: This course will highlight the challenges of practically implementing computer vision on workstations, mobile, and wearable devices. The course will explore techniques to enhance the latency and bandwidth performance of vision execution, including an investigation of acceleration through vector-based, multi-threaded, and SIMD parallelism. Students will then learn techniques to reduce the energy consumption of vision executions, e.g., spatiotemporal subsampling, cloud offloading, and hardware acceleration. Lectures will cover state-of-the-art materials and tools (OpenCV in iOS and Android, imaging pipelines of modern SoCs) to connect students with a understanding of current systems. Homework will involve a hands-on implementation of the performance and efficiency strategies, culminating in a creative course project to design a high-performance and/or energy-efficient vision application.

Mobile Systems: This course will study the theory and implementation of mobile systems design. Students will learn several levels of mobile systems design, including: mobile I/O, app development frameworks, system services, sensor platforms, network connectivity, battery usage, and System-on-Chip architecture. Using Android and iOS as examples, the course will maintain relevance to contemporary mobile and wearable systems. Homework will involve userspace-level and kernel-level development with an emphasis on operating systems support (as opposed to app development).

Research Mentorship

Teaching the mentality of the research pursuit at the graduate and undergraduate level is challenging. While technical skill is important, the defining characteristic of impactful researchers is in how they connect their ideas to advance academia and industry. As such, through prior mentorship of younger graduate students and undergraduates, I've found that I must identify research topics that will energize the students to form such connections. A student who is doing research by checking boxes will be far less effective than one truly motivated to solve a meaningful challenge. Meaningful challenges encourage students to persevere through tough spots and relentlessly gain necessary skills. Fortunately, in the field of continuous mobile vision, this has been easy; providing computers with sight inspires the imagination (and work ethic) of the students I've mentored, which has only strengthened my own passion for continuous mobile vision.

Impactful researchers also communicate effectively with the outside world. During my time with the Rice Center of Engineering Leadership, I created speaking opportunities and presentation coaching for my peer graduate students to articulate their research interests. The speaking students reported that the platform to preach actually motivated them to pursue their research with more vitality, understanding the importance of what they were doing. But more importantly, it also provided them the skills to effectively convey complex topics to a wide audience, giving them the confidence to pursue research connections to academic and industrial collaborators.

In all, my teaching philosophy encourages students to form connections from student-to-topic, and topic-to-world. In doing so, I hope to equip and inspire students to be the next generation of thought leaders, pioneering new technology in academia and industry.

References

[1] John L. Hennessy and David A. Patterson. *Computer Architecture, Fifth Edition: A Quantitative Approach*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 5th edition, 2012.

[2] Abraham Silberschatz and Peter B Galvin. *Operating System Concepts*. Wiley, 9th edition, 2013.

[3] Jeremy Andrus and Jason Nieh. Teaching operating systems using android. In *Proceedings of the 43rd ACM technical symposium on Computer Science Education*, pages 613–618. ACM, 2012.