

EDUC 236X – CS 402

Beyond Bits and Atoms

New Technologies for Creative Learning

Prof. Paulo Blikstein

General Information

School of Education EDUC 236X

School of Engineering CS 402

Spring 2011

Friday, 10:00 – 12:50pm, Room TBD

Lab

This course has a lab session (EDUC-211X/CS-402L) for which you should register as well. In very special cases, we will allow students to take only the lab or only the lecture sessions, but that has to be discussed before the start of the quarter. For more information on the lab session, see the lab syllabus on the website. Lab session happens on Fridays (2:15-5:05pm) in the fabrication lab (CERAS 102).

Teaching team

Paulo Blikstein (paulob@stanford.edu)

Assistant Professor of Education and (by courtesy) Computer Science

Director, Transformative Learning Technologies Lab and Stanford Makers' Club

520 Galvez Mall - CERAS 232 - tel. (650) 736-0966

Office hours: Tuesday, 4-5:45pm (CERAS 102 or 232)

TA: TBD

TA Office hours: TBD

Course Web Site & email lists

<http://beyondbitsandatoms.stanford.edu>

Email list for course instructors: bbafaculty@lists.stanford.edu

Email list for all course members (students and faculty): bba2011@lists.stanford.edu

Course Abstract

This course is a hands-on practicum in evaluating, designing and building technology-enabled curricula, tools, and learning environments. We will use many rich software toolkits and state-of-the-art prototyping technologies (3D printers, laser cutters, routers, robotics, sensors) designed for novices to get their “hands dirty” designing educational software, educational toolkits, educational toys, and tangible user interfaces. A special focus of the course will be to design low-cost, appropriate technologies, particularly for urban school in the US and abroad.

After completing this course, you should be able to:

- Understand constructivism, constructionism, computational literacy and thinking, critical pedagogy, and why they are crucial for designing next-generation learning tools.
- Design sophisticated objects and tangible interfaces using tools such as 3D printers, laser cutters, 3D scanners, and microcontrollers (see lab syllabus).
- Put together theory and design skills to create educational software/hardware at the prototype level, avoiding common design errors.
- Assess and critique technology-rich learning products.
- Design technology-enabled activities that take advantage of the computational medium, considering the knowledge domain, age group, context, and deployment situation.
- Interview and do user testing with children and young adults.
- Implement technologies in real-world classrooms.
- Change the world, eventually.

I can't program or do anything technical. Should I enroll?

No previous programming or technology background is assumed. Really. Even if you have never programmed in your life, you can still enroll! This class is a good fit for people both with and without a technical background. Students with a technical background will learn how to use their skills to design meaningful tools for learning—and not just cool “gadgets” disconnected from learning theory. Students from the humanities, art or psychology will learn how to make their ideas come to life—a toy, a piece of software, a tangible human-computer interface, or a technology-rich curriculum.

This class will emphasize authoring projects using Logo-inspired languages. Logo is a computer programming language designed explicitly for use by children and is in use in large numbers of schools – common implementations of Logo are Microworlds, Scratch, NetLogo, StarLogo, ModKit, and LogoBlocks.

Even if you might not intend to be an educational software designer yourself, it is the reality of today— and more so, of tomorrow— that should inform your choice to become educated about the promise of technology in education. Courses similar to this have been taught around the world and people with no programming experience have done extremely well. The class is structured for non-programmers to succeed. However, programming and prototyping does take time and you will be expected to devote substantial time to it, just like any other new skill. You are strongly encouraged to get help from your fellow students through the class email list as well as from the TAs. The TAs will hold weekly office hours designed especially for technical and programming support.

In addition to projects, there will be weekly readings. Typically, two papers per week. There is a considerable literature that we will not have time to read this term, so you will find an extensive additional bibliography at the end of this syllabus. You may find some of these readings to be useful to you in completing the final project.

Course Description

Technology has been the “next big thing” in education for the past 50 years – first with television, videotapes, computers, the internet, and mobile phones. Countless conferences, books, national plans, and international initiatives have promised better student performance, motivating courses, better teacher training, lower costs, or more equity.

Why hasn't technology in education lived up to its hype? Answering that question is one of the goals of this course. Part of that answer is that, for the most part, technology designers often do not know much about the learning & cognitive sciences, and educators, entrepreneurs, and educational researchers are not fluent enough with the new technologies to become designers, or at least educated consumers of advanced technologies. Educational technologies are always years behind the bleeding-edge of technology. Also, not much has been done in the direction understanding the role of cultural differences in the uses of technology across communities and countries. The result is that most

solutions are either too technocentric, ignoring cognitive and cultural issues, or do not use technology to its fullest potential.

In this course, we will both learn how to study, evaluate *and* design technologies for learning, with a special focus on cultural appropriateness. Towards that end, we will have five main kinds of activities:

- **Lectures and class discussions:** here we will get in touch with the theoretical foundations in the cognitive science, developmental psychology, human-computer interaction, critical pedagogy, constructivism, and constructionism.
- **Product reviews:** in class, we will review 10-20 existing software and hardware products (free and commercial) for education, and learn how to critique and evaluate them.
- **Invited speakers** from the MIT Media Lab, Berkeley, Northwestern, Nokia Research, and foreign universities will tell students about the cutting-edge of technological design for learning.
- **Contact with children:** at least one assignment will require interviewing or testing a design with children.
- **Lab course:** using state-of-the-art prototyping machines (such as laser cutters and 3D printers) students will have an immersive, hands-on design experience, and become fluent in several prototyping technologies.

The main theoretical backbone of the course is the **constructionist learning** design perspective. This perspective, first named by Seymour Papert and collaborators at MIT, and greatly influenced by the work of Jean Piaget, is very influential in the learning sciences today. *The constructionist approach starts with the assumption that teaching cannot successfully proceed by simply transferring knowledge to students' heads.* Skillful teaching starts with the current state of knowledge of the student. In order for students to learn effectively, they need to construct the knowledge structures for themselves. In the spirit of constructionism, we will engage in our own construction of artifacts in this class and, through this activity, explore and evaluate the design of kits and tools intended to enable learners to construct their own motivating and powerful artifacts. We will do this by constructing both physical and virtual artifacts and by engaging in reflective discussion of both the artifacts themselves and the tools used to

construct them. Another important influence in this course is the work of Paulo Freire on **critical pedagogy**. Freire states that the school curriculum has to depart from the learners' cultural context and values, and emphasizes the importance of designing learning environments in which agency, social empowerment, and emancipation are crucial components.

In the final project, students will put all of this together by designing and implementing a constructionist learning environment, with critical pedagogy inspirations, using any combination of the tools and machines studied during the course.

Software/hardware platforms we will use

We will use three main platforms in this course:

- **Scratch:** a multimedia, easy-to-use graphical programming language inspired by the Logo perspective. It is used by hundreds of thousands of students worldwide for the creation of programs, games, multimedia, and stories.
- **NetLogo:** a multi-agent version of Logo, this language is tuned for constructing models of dynamic systems. It is useful for creating models of ecological systems, chemical systems, economic trade, social behavior, etc.
- **GoGo Board:** a low-cost, open-source robotics-and-sensing interface for educational use. It can be used for building robots, working prototypes (a new system for trash recycling or water purification, for example), and environmental sensing (for example, doing research on water quality or pollution). We will construct devices that have sensors and motors and can interact with objects in the world. We will also look at **Arduino** boards.
- **Other platforms:** depending on students' final projects, we will also use other platforms such as **Reactivision** (for tangible table-top interfaces and digital surfaces), **Netduino**, **Propeller**, **Cubelets**, **Topobo** and **LEGO NXT** (robotics), **Lilypad** (wearable computing), **Fritzing** (electronics design), and languages such as **Processing** and **Alice**.

Besides these packages, the class includes a software review assignment in which we look at a number of other packages, including : TableTop, Genscope, Biologica, Zoombinis, SimCalc, HubNet,

ChemSense, Fathom, MediaMoo, Moose Crossing, CSILE, Hypergami, Stagecast Creator, Vehicles, RelLab, Interactive Physics, the Sims, Impromptu, Geometer's Sketchpad, MatLab, Squeak, Boxer, Model-It, STELLA, MyWorld, Hyperscore, Sugar (One Laptop Per Child), etc.

Summary of Requirements

This course is designed to be somewhere between a class and a working group. We will work together to make sense of readings, and, for most of the class projects, you will be working in small groups. The requirements for everyone are:

- Set up your personal blog for the course, and keep detailed logs of your projects, with rich media (videos, pictures).
- Keep up with the readings and participate in class.
- Attend the lab sessions and complete the lab assignments.
- Complete and present (mostly group) assignments.
- Review one product and present your review in class.
- Design and implement your final project, and give a presentation during the last week of the course.

Can I miss a class?

Due to the project-based nature of the class, and the number of group projects, attendance is very important. The standard policy is that, if you have a very good justification, and if you email us *in advance*, you can miss *up to one class*. If you anticipate missing more than one class (for example, you already know you will be away for two weeks), you should not enroll.

About the Final Project

The final project is to design and implement a constructionist learning environment. There are (at least) three alternatives for this project:

1) Standalone Educational Software (scaffolding in software)

Design and implement a constructionist educational software. This option would involve writing a design specification for the software that describes what it is for, who it serves, why it is needed, why it

is best done in software, etc. Subsequent to receiving feedback on the design specification you will need to start working on a functional specification of the software itself and then embark on implementing it. You are free to use any authoring tools you like to implement the software as long as you make a good argument for their being well matched to the task. Suggested educational software genres are: a simulation game, a microworld, a collaborative role-play.

2) Software/hardware-embedded curriculum (scaffolding in curricular materials)

Design and implement an educational activity that has a computationally embedded component. In this option, you could use one of the software environments used in this course: Microworlds Logo, Scratch, GoGo Boards, NetLogo, Arduinos, or others (for example: an iPhone/Android development platform).

As above, you would begin with a design specification. Depending on the design, you may or may not require a functional specification – it could be a curriculum flow specification instead. You would then go on to construct the software and/or tangible interfaces that form the kernel of the activity, flesh out the curricular materials that accompany the software/hardware and write up a paper that describes one person's path through the activity.

3) Toolkit

Design and implement a constructionist toolkit (special preference will be given to low-cost toolkits). Examples of a toolkit are a robotics kit, a science kit, an environmental investigation kit, a construction kit for physics, a storytelling environment, etc. This option would involve writing a design specification for the toolkit that describes what it is for, who it serves, why it is needed. Subsequent to receiving feedback, you will write a functional specification: components that the toolkit will have, technologies will you use to create it, cost/fabrication issues, examples of projects that students can create with it, and what learning outcomes are expected. You are free to use any authoring tools and prototyping machines.

For some students, the final project could take a different direction, such as designing a (computational) model of how students behave in a classroom, using NetLogo. If you're interested in this or other options, talk to us.

Important dates for the final project

- The first project design specification (1 page) is due by April 29th 2011.
- The second project design specification is due by May 6th 2011.
- The final project functional specification (or curricular flow specification) is due by May 13th 2011.
- The beta version of the final project software/hardware is due May 27th 2011.
- Final projects will be presented on June 3rd 2011 during a special event. You are welcome to invite friends and/or relatives to attend.
- The final project documents (paper, video, photos, blog post) are due by June 6th 2011.

Grading

All assignments will be graded as either complete or incomplete. If an assignment is judged incomplete, you will have an opportunity to complete it or redo it the following week. Incomplete assignments not redone will impact your final grade. You will also be assessed on your class participation both in class and on-line – your weekly reading reactions and comments on the blog, and your contribution to the discussions in class.

Overall, the final project will be worth 50% of the final grade, assignments will be worth 30%, and 20% will be given for reading reactions and class participation.

Readings

There will be no printed course reader. All materials will be online as PDFs. As a result, we will save one third of a tree over the ten weeks, and a lot of electrical energy. This will make us a bit less guilty for using so much energy and materials in the lab sessions.

Assignments (further details will be given each week)

- **Reactions to readings (weekly):** Every week, you have to post reactions to the papers read for that week. As a rule, you will choose one idea from each and write 1–2 paragraphs. For classes with multiple papers, we will indicate the papers about which

you should write a reaction, and we will try to keep it the maximum at two. You are supposed to react to your other people's reactions too.

- **“Gears” essay:** a short essay about an crucial object in your childhood, which influenced your intellectual development (in the spirit of Papert’s “The Gears of My Childhood” from his “Mindstorms” book)
- **Sensor-activated hypermedia Project:** in groups, create a hypermedia Project in Microworlds or Scratch to present something personal about each person in the group. The pieces for each person should connect in some interesting way (for example, a quilt in which each person is a piece, a story, or a cartoon), and should use sensors as triggers.
- **NetLogo assignment 1:** Choose a model from the NetLogo Models Library and add a new feature (with or without sensors).
- **NetLogo assignment 2:** Create a NetLogo model from scratch (with or without sensors).
- **Small toolkit:** build a toolkit to teach something simple to a perfect stranger.
- **Design for a real child:** you will have to interview a child and build her/his “dream” toy.
- **Software/hardware/toolkit/toy review:** an in-depth review of educational software or toy. A list of qualified items will be given in class. Students are expected to present their review to the class. You can use, extend, revise, or re-create the taxonomy of interactive software from the readings.

Schedule

Class	Class
Class 1	Constructionism: Introduction
Class 2	Computational literacy and computational thinking.
Class 3	Educational fabrication: bits, atoms, and kids
Class 4	Constructivism: Piaget.

Class 5	Critical pedagogy, cultural relevance: Paulo Freire, Illich.
Class 6	Complexity theory, NetLogo: Wilensky
Class 7	Computational Literacy and Epistemological Pluralism: diSessa, Papert, Turkle.
Class 8	The maker revolution, a new age in tangibles for learning, One Laptop per Child.
Class 9	Overview, future directions, the school of the future.
Class 10	Final projects

Readings (required readings in **bold**)

Every week, on average, we will have two papers to read.

Class	Class
Class 1	<ul style="list-style-type: none"> ▪ Papert, Seymour (1980). <i>Mindstorms</i>. New York: Basic Books ▪ Papert, S. (1991). <i>Situating Constructionism</i>. Constructionism, Ablex Publishing Corporation.
Class 2	<ul style="list-style-type: none"> ▪ Papert, S. & Solomon, C. (1971). <i>20 Things to do with a computer</i>. ▪ Kay, A. (1991). <i>Computers, Networks, and Education</i>. Scientific American, Sept. 1991, pp. 138-148. ▪ Papert, S. (2000). <i>What's the big idea: Towards a pedagogy of idea power</i>. IBM Systems Journal, vol. 39, no. 3-4. ▪ Resnick, Mitchel. (2001). Closing the fluency gap. Communications of the ACM. ▪ (optional) Feurzeig, W. et al. (1970) Programming-languages as a conceptual framework for teaching mathematics.
Class 3	<ul style="list-style-type: none"> ▪ Eisenberg, M. (2002). <i>Output Devices, Computation, and the Future of Mathematical Crafts</i>. International Journal of Computers for Mathematical Learning, 7(1), 1-43. ▪ Resnick, M. et al. (1998) <i>Digital Manipulatives: New Toys to Think With</i>. CHI 98. ▪ Eisenberg, M. (2003). <i>Mindstuff: Educational Technology Beyond the Computer</i>. Convergence. ▪ Eisenberg, M. (2007). <i>Pervasive Fabrication: Making Construction Ubiquitous in Education</i>. Fifth Annual IEEE International Conference on Pervasive Computing and Communications Workshops.

	<ul style="list-style-type: none"> ▪ Raffle, H. (2004). <i>Topobo for Tangible Learning</i>.
Class 4	<ul style="list-style-type: none"> ▪ Papert, S. (1999, March 29). <i>Papert on Piaget</i>. Time Magazine, special issue on “The Century's Greatest Minds,” 105. ▪ Selections from Jean Piaget ▪ Abrahamson, D., & Wilensky, U. (2005). <i>Piaget? Vygotsky? I'm game!: Agent-based modeling for psychology research</i>. Paper presented at the annual meeting of the Jean Piaget Society. ▪ Smith, E. R., & Conrey, F. C. (2007). <i>Agent-based modeling: A new approach for theory building in social psychology</i>. <i>Personality and social psychology review</i>, 11, 87-104.
Class 5	<ul style="list-style-type: none"> ▪ Selections from the Pedagogy of the Oppressed, by Paulo Freire. ▪ Blikstein, P. (2008) Travels in Troy with Freire: Technology as an Agent for Emancipation. in Noguera, P. & Silva, C. A. (eds.). <i>Freire and the Possible Dream</i>. Sense Publishers, Rotterdam. ▪ Cavallo, D., Blikstein, P. et al. <i>The City that We Want: Generative Themes, Constructionist Technologies and School/Social Change</i>. In Proceedings from the IEEE International Conference on Advanced Learning Technologies, Finland, September 2004.
Class 6	<ul style="list-style-type: none"> ▪ Wilensky, U. (2001). <i>Modeling Nature's Emergent Patterns with Multi-Agent Languages</i>. Paper presented at the Eurologo 2001, Linz, Austria. ▪ Wilensky, U., & Resnick, M. (1999). <i>Thinking in Levels: A Dynamic Systems Approach to Making Sense of the World</i>. Journal of Science Education and Technology, 8(1), 3-19. ▪ Blikstein, P. & Wilensky, U. (2007). <i>Bifocal modeling: a framework for combining computer modeling, robotics and real-world sensing</i>. Paper presented at the annual meeting of the American Educational Research Association (AERA 2007), Chicago, USA. ▪ Kelly, K. (1994). <i>Out of control: The rise of neo-biological civilization</i>. Reading, MA: Addison Wesley. (chapter 1)
Class 7	<ul style="list-style-type: none"> ▪ Turkle, S., & Papert, S. (1990). <i>Epistemological Pluralism</i>. Signs, vol. 16, no. 1 ▪ diSessa, A. (2000). <i>Changing Minds: Computers, Learning, and Literacy (Selections)</i>. MIT Press.
Class 8	<ul style="list-style-type: none"> ▪ Papert, S. (1987). Computer criticism vs. technocentric thinking. <i>Educational Researcher</i>, 16(1), 22-30. ▪ Pea, R. D. (1987d). Programming and problem-solving: Children's experiences with Logo. In T. O'Shea & E. Scanlon (Eds.), Educational computing (An Open University Reader). London: John Wiley & Sons. (Also Technical Report No. 12, Bank Street College, Center for Children and Technology, April 1983).

	<ul style="list-style-type: none"> ▪ Selections from Cuban, L. (2001). <i>Oversold and Underused: Computers in the Classroom</i>. Cambridge: Harvard University Press.
Class 9	<ul style="list-style-type: none"> ▪ Readings from the IDC 2009-2011 conference (TDB) ▪ Caitlin Kelleher and Randy Pausch, <i>Lowering the Barriers to Programming: A Taxonomy of Programming</i>. ACM Computing Surveys, Vol. 37, No. 2, June 2005, pp. 83–137
Class 10	<ul style="list-style-type: none"> ▪ Final project presentations

Lab Fee

- Charging a lab fee is the typical policy in lab courses at Stanford and other schools. The lab fee (\$95) will be used to buy shared materials use during lab sessions, including basic maintenance, and one microcontroller kit for each student. We will have a limited supply of acrylic, wood, 3D printer powder & binder, sensors, motors, and robotics stuff. The lab fee is not meant to support final projects, though student are welcome to use leftover materials from the appropriate bins. If students want to use materials that go above and beyond what the lab fee covers, they should be ordered directly, or reimbursed/replaced to the lab.
- Can I buy the materials myself and not pay the lab fee? *No*. We have tried it and it doesn't work at all. Some materials can only be bought in bulk (such as 3D printer powder), and it is impossible to control individualized materials in the lab.
- The lab fee is due on week 3. Please write a check to Stanford University.