Playing a Game: The Ecology of Designing, Building and Testing Games as Educational Activities

Alexander Repenning and Clayton Lewis
University of Colorado at Boulder
Boulder, CO, USA
[ralex, clayton]@cs.colorado.edu

problem

- general lack of authentic, project-oriented courses involving teams of students to produce non-trivial artifacts
- Early curriculum focuses on highly isolated skills typically reducing the notion of a project to throw away programs resulting from individually implemented textbook algorithms
- Team-based projects of open-ended problems are only found at senior levels
- often these senior projects result in negative “educational” experiences conveying how not to organize a project

There is a need to understand and to support a complex ecology involving instructors, university students, K-12 students and external design knowledge in order to be able to successfully build these games in a relatively short amount of time

Why are games good for education?

The process of making games (not to be confused with playing games) is a rich design experience because it requires the understanding and the ability to synthesize aspects of many domains:

- Computer science (algorithms, data structures, real-time processing…)
- Art and media (images, 3D models, animations, sound, video…)
- Educational domains (math, geometry, science…)

Playing game…early beginnings

- Object Oriented Design was a dreaded course by most faculty because students expectations varied widely
  - Half like to hands-on, project oriented course
  - Half like to have theory only (e.g., UML diagrams)
- Problem: Hard to balance theory and practice
  - Practice: Textbook examples often feel contrived and/or plain boring
  - Theory: internalization requires experience

Sneak game design into Object Oriented Design course

- Why? > 30 % of students indicated that playing games was one of their main reasons to enter a computer science program
- Approach: use rapid prototyping tool (AgentSheets) to have student design and build a game per week
- Provide description of classic arcade games

Step 1: provide game description

Try to find original game descriptions

Space Invaders

Space Invaders was an arcade video game designed and programmed by Toshihiro Nishikado and originally manufactured by Taito; it was licensed for production in the U.S. by the Midway Manufacturing division of Bally. Released in 1978, it ranks as one of the most influential video games ever created. Though simplistic by today's standards, it (along with other contemporary games such as *Pac-Man*) was one of the forerunners of modern video gaming.

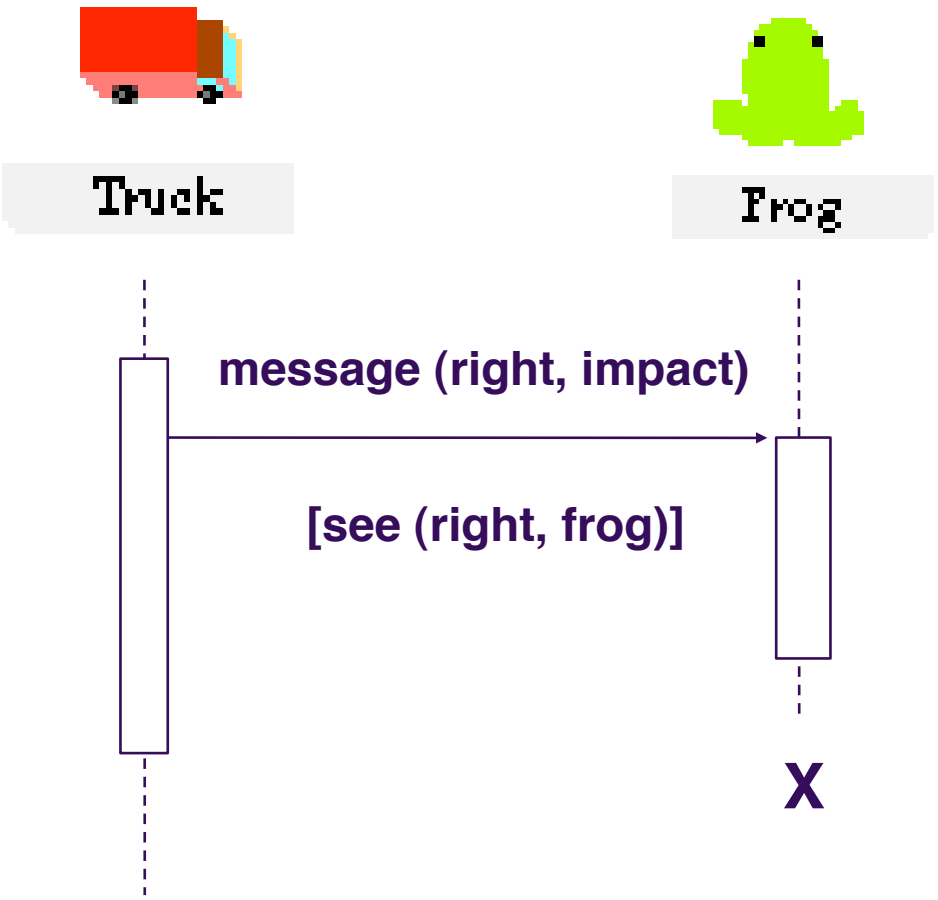
http://en.wikipedia.org/wiki/Space_Invaders

Step 2: game design

Use traditional software engineering + game design specific approaches (e.g storyboarding)

example sequence diagram

Situation: Truck hits frog

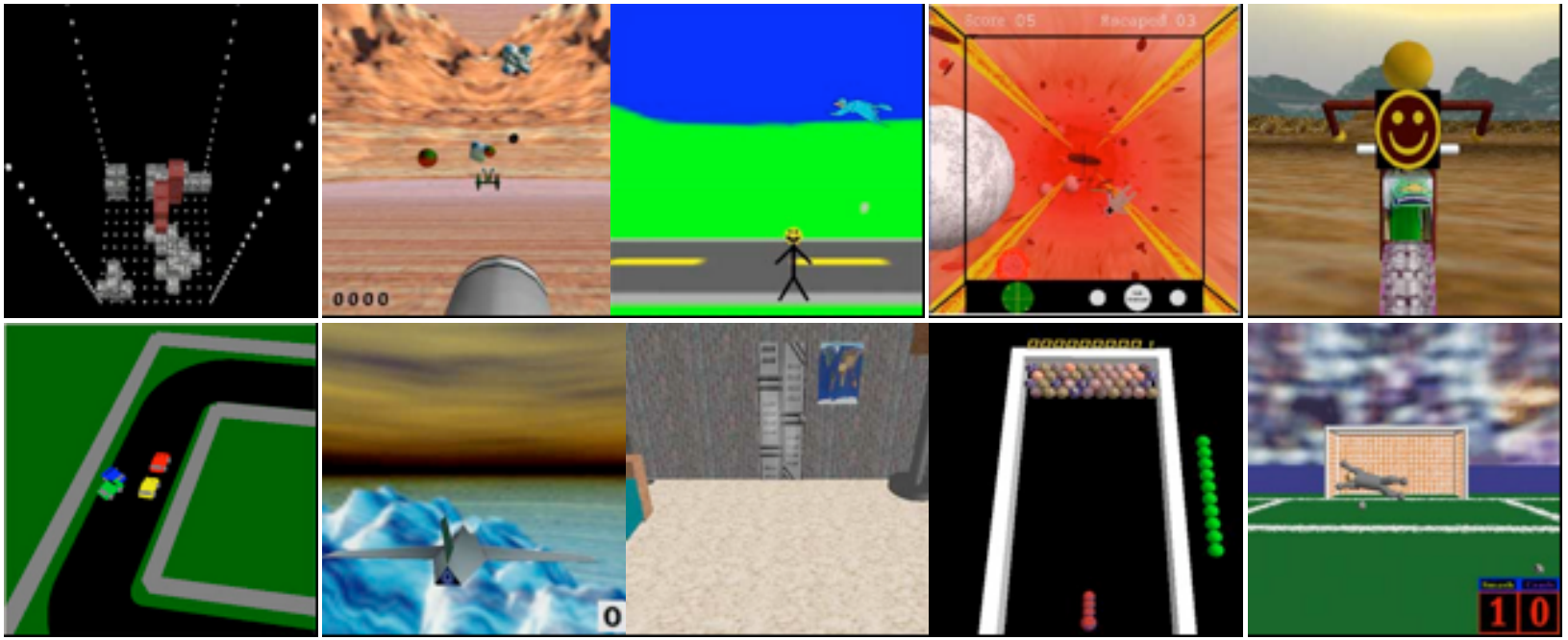


Truck sees a frog to the right

Sends impact message to frog

Frog looks like dead frog and after a little while disappears

example games



Mostly positive experience

- Students put in more time than asked for
- Created pretty sophisticated games including AI
- Initially ~90% “cheated” by first making the game and THEN creating the design documents

Scaling up: Games -> Educational Games

- The process of making games can be highly educational, BUT
- How can we have students build games that are educational?

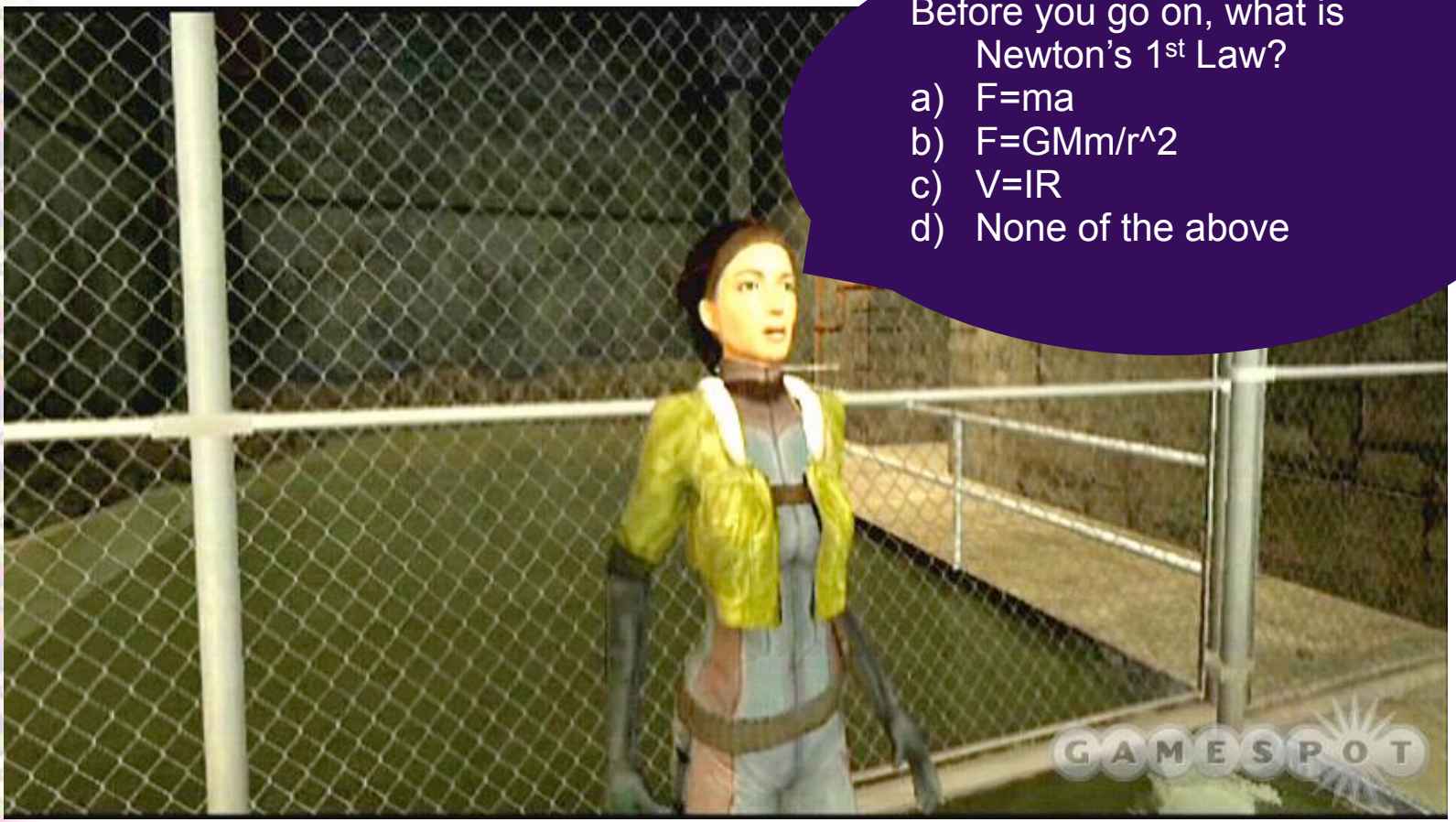
4 challenges

challenge 1

Establish meaningful connections between engagement and learning using Engagement/Learning continuum

- ***Educational Design*** (Learning → Engagement): Educational design's main objective is learning. This design process starts with learning but gradually adds elements of engagement. A popular design approach used in education is *backward design* (Wiggins and McTighe, 2000).
- ***Game Design*** (Engagement → Learning) Game design is highly focused on motivational aspects such as engagement and fun (Koster, 2004). Most games have clever scaffolding mechanisms built in (Gee, 2004) allowing their users to gradually solve more complex problems. However, these mechanisms are typically used to learn about using the game and not about some educational topic. Most game design

Example of bad connection

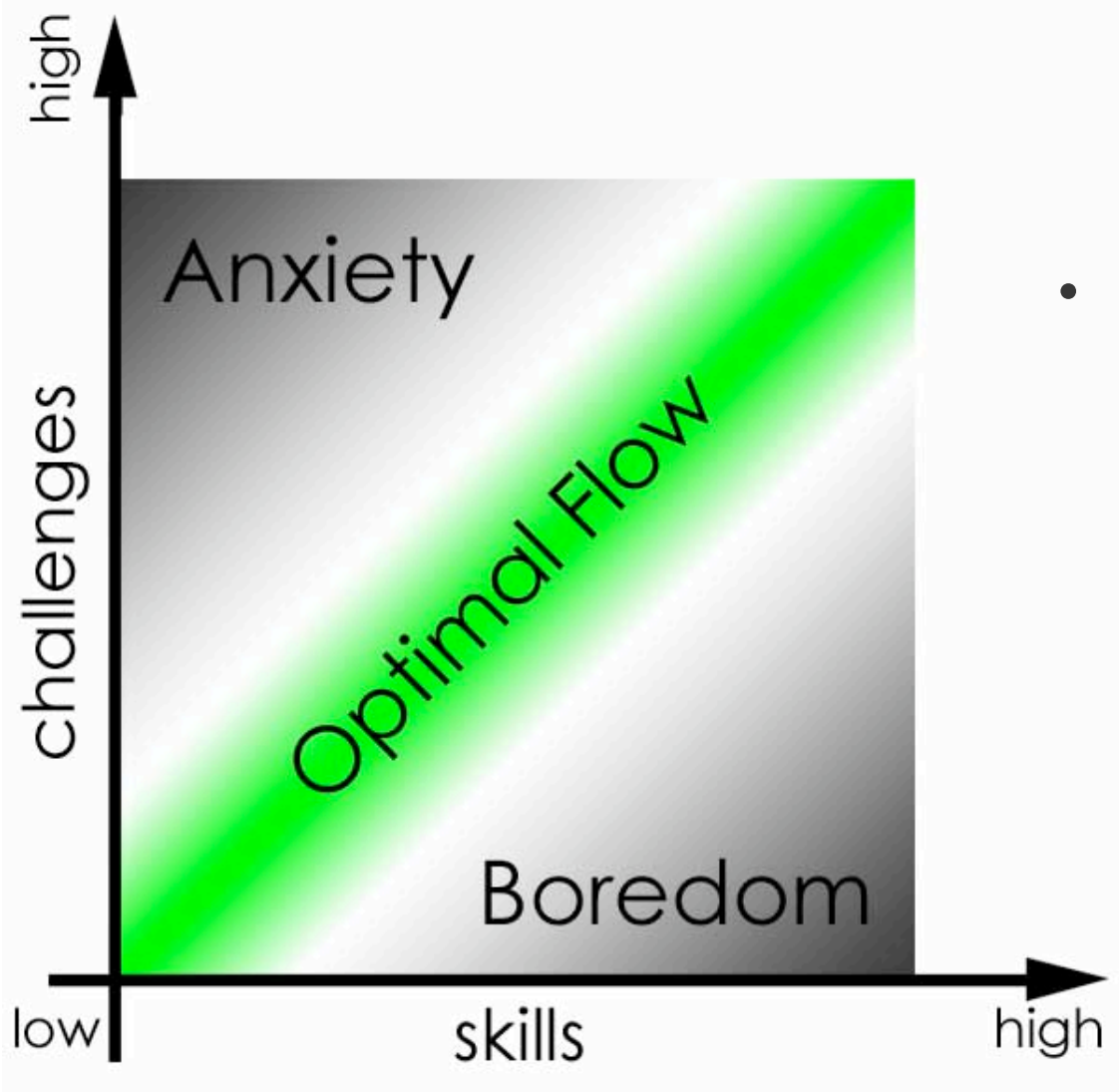


Example of good connection?

- Typing of the Dead

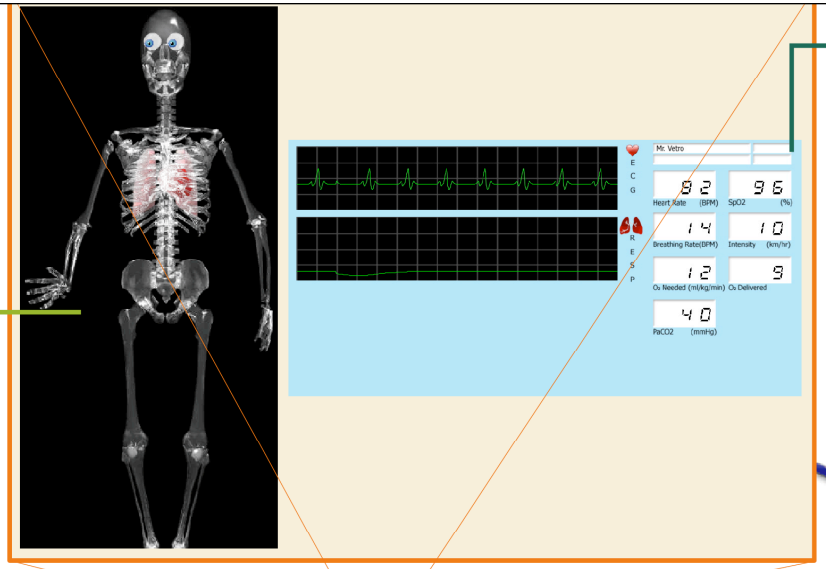
http://dreamcast.ign.com/articles/165/165674p1.html

optimal flow



- Good games keep players in the flow by balancing skills and challenges
- Motor skills such as typing can measured: game becomes adaptive by adjusting challenges
  - More complex words
  - Less time to react

In an engaging educational activity, the server runs a collective simulation portraying Mr. Vetro, a simulated human being with a collection of simulated organs that are distributed on handhelds. The server gathers data from client simulations and serves as a simulation coordination and visualization tool.



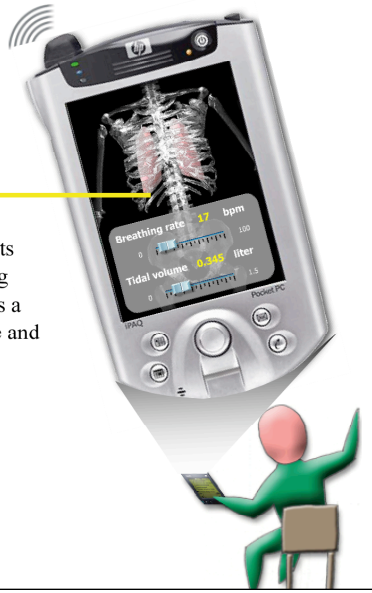
A Life Signs Monitor keeps track of Mr. Vetro's vital signs and displays them in the form of graphs or numerical values. ECG, heart rate, breathing rate, oxygen saturation, and oxygen delivered to tissue are some of the physiological variables.

The teacher orchestrates the educational activity by assigning the control of different organs of Mr. Vetro to groups of students, giving them tasks to complete as a team, monitoring progress, and facilitating classroom discussions.

With a wireless network, the handhelds send data to the server.

The collective simulation is projected to the entire class and therefore serves as a classroom discussion tool.

In a simulation running on a handheld, students control the lungs of Mr. Vetro by varying lung parameters (breathing rate and tidal volume) as a response to changing conditions such as exercise and smoking.



Another group of students controls the heart of Mr. Vetro by varying heart parameters such as heart rate and stroke volume to adjust to changing conditions such as increased exercise intensity.

challenge 2

- Providing diverse background information
- underestimated the need for good background materials in earlier versions of the course
- overestimated our students' ability to pick up the needed concepts from readings in the literature
- Hard to sell education concepts such as learning theory (Act-R) to computer science students
- we didn't provide enough examples of worked-out analyses to supplement the presentation in the notes

challenge 3

- Preparing CS students to interact with K12 students and teachers
- Collaboration between university and K-12 students needs to be heavily scaffolded since few university students have any experience in user centered design approaches involving actual contact with users
  - Many computer science students put up quite some resistance to go to schools
  - They are not used to the concept of a “user”

challenge 4

- Making games is hard. Options we explored:
  - ***3D from Scratch***. Especially to computer science students there is a natural affinity to use high end programming tools. 3D games with complex rendering are often considered the holy grail of engagement.
  - ***3D with Game Engines***. Game Engines are software packages that will substantially leverage the design and implementation of 2D/ 3D games. In some sense they can be considered a middleware layer between the low-level 3D APIs and the game application.
  - ***Gamelets***: Simple Web-Based Game Building Tools. In lieu of the likely complexity emerging from 3D games we have explored the notion of so called Gamelets as simple versions of games. A Gamelet has a complexity comparable to classic arcade games such as Pacman.