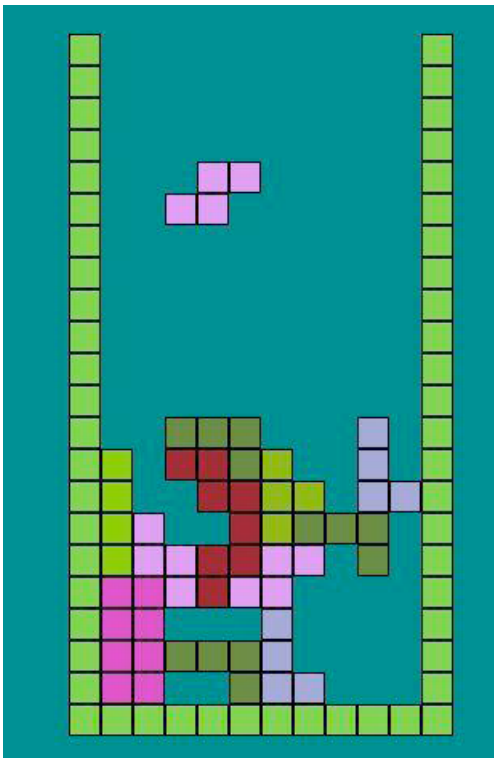
Video Games



Sometimes very simple game concepts can be extremely successful (*e.g.* Tetris)

Thanks: K. Egan, J. Parker, J.Kuffner

Terminology

- Video game - a game that involves interaction with a user interface to generate visual feedback on a video device (traditionally, on a raster display device)
  - the electronic systems used to play a video game are known as platforms;
  - example platforms: video game console, arcade machine, or personal computer
- Computer game – a video game played on a personal computer
- Game controller – the input device used to control a video game
  - varies among platforms: keyboard & mouse, joystick and a button, several buttons and joysticks etc
- Game engine – the core software component of a video game; multiple video games may use the same game engine
  - typically includes a 2D/3D renderer, a scene graph, collision detection and response (aka physics engine), animation, networking, sound etc

Game Engine: Basic Game Loop

```
while (!gameover) {
  get input
  process input
  simulate physics and AI
  redraw
}
```

- As cs1566 graduates, you can all do this
  - Modeler (P04): level file format and (bad) scene data structure
  - Transformer (P03): take care of camera and animation
  - Basic Interaction (P01&4): user manipulation
  - Ray-Object Intersections (P05): basics of collision
  - You can hack together a FPS without too much effort
- How fast does my game loop need to run?
  - 60Hz, preferably (range: 15Hz lowest, 1000Hz for haptics)

Game engines: build, rent or buy?

- Think of the engine of a car
- The game engine…
  - Takes input
  - Computes physics (collisions, projectiles, etc.)
  - Plays sounds, simulates AI
  - Draws stuff!
- A good game engine is independent of the game type; the one we've built in cs1566 is very basic

- Writing a good game engine is hard
  - top commercial game engines: Torque Game Engine, TV3D SDK 6.5, 3D Game Studio
  - top open-source game engines: Crystal Space, Game Blender, The Nebula Device (used in many commercial games), Ogre3D (graphics engine, really)

- Most developers license engines
  - *Doom III*, *Unreal 3, Crysis*
  - Ever notice that a lot of games look alike?
- Well, what's a game then?

  Gameplay and Art

Overview

- Game Engines
  - the Game Loop
- Realism & the Uncanny Valley
- Rendering
- Physics
- AI
- Networking
- Art Pipeline

- Getting a job
- Modeling and Simulation

- References

Genres

- Games - Games
  - MMORPG (Massively-Multiplayer-Online RPG)

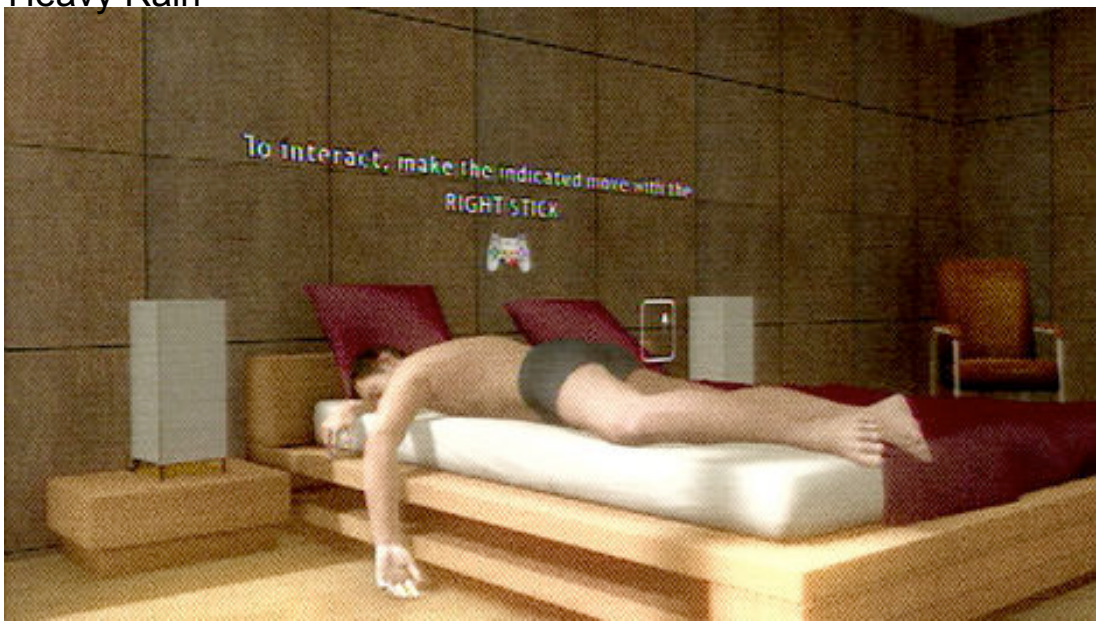
World of Warcraft

- Serious Games
  - provide essential experiences that in real life are too difficult, dangerous, or expensive
  - e.g.: simulations, training, education

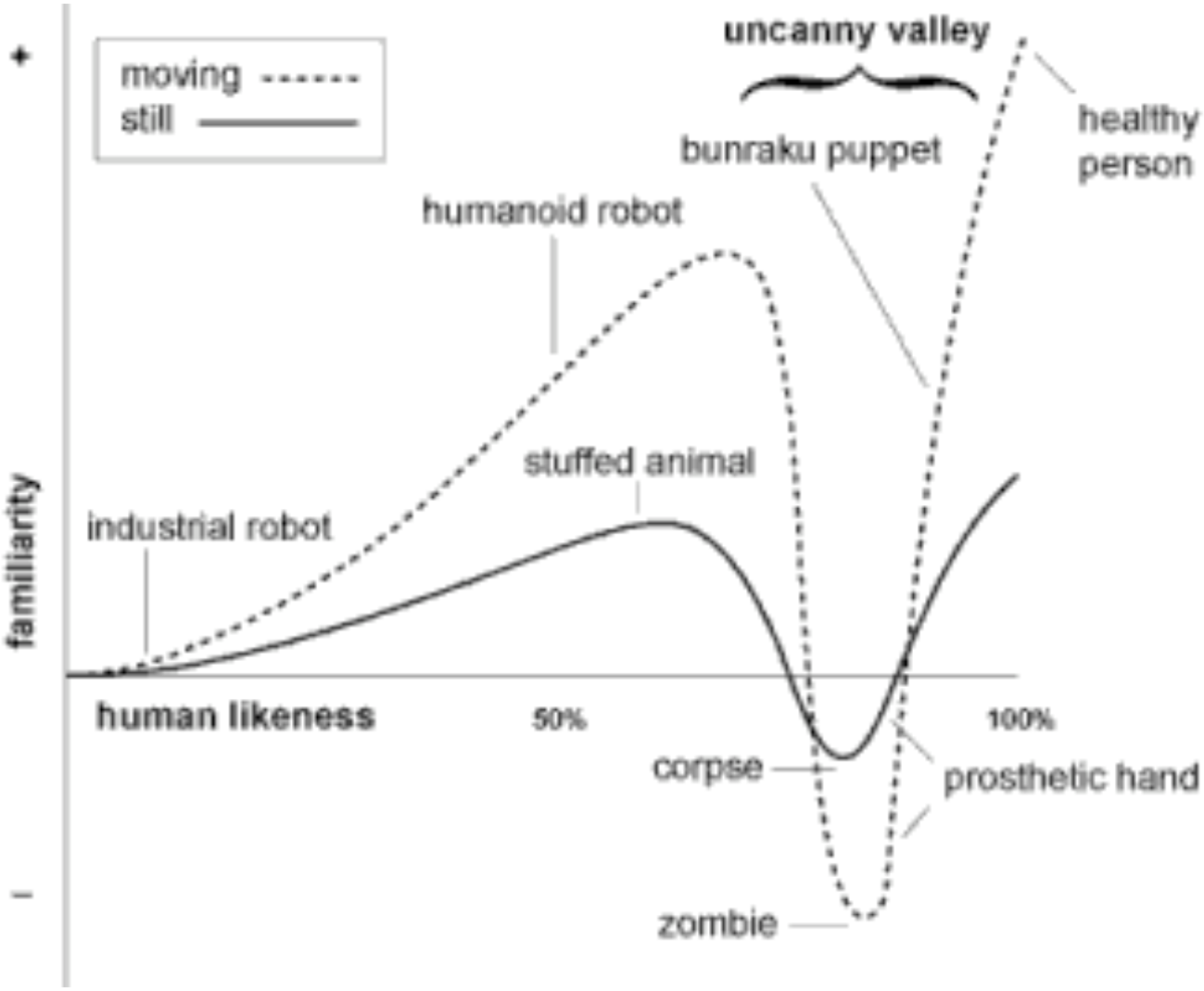
Realism: "If it looks like CG, it is not good CG" (J. Birn)

Images by Hervé Steff http://www.insideko.com/. Rendered with Maxwell Render.

Heavy Rain



But also beware the Uncanny Valley!



- http://spectrum.ieee.org/slideshow/robotics/humanoids/ode-to-the-uncanny-valley

Uncanny Valley

- "Coraline" eyes (less photorealistic, but familiar)
- "Polar Express" eyes (more photorealistic, but scary)

Realistic Rendering

- The Kajiya rendering equation (covered in advanced CG classes) describes this in exacting detail
  - very expensive to compute!
  - previous techniques were different approximations to the full rendering equation
- Photon mapping provides a pretty good approximation to the equation.

- Eric Veach's Metropolis Light Transport is a faster way of sampling the full rendering equation (converge to accurate result of the rendering equation)
  - add subsurface scattering!
  - impossible to get a real-time, high-fidelity global illumination rendering

Rendered using MLT, all light comes from the other room

Subsurface Scattering

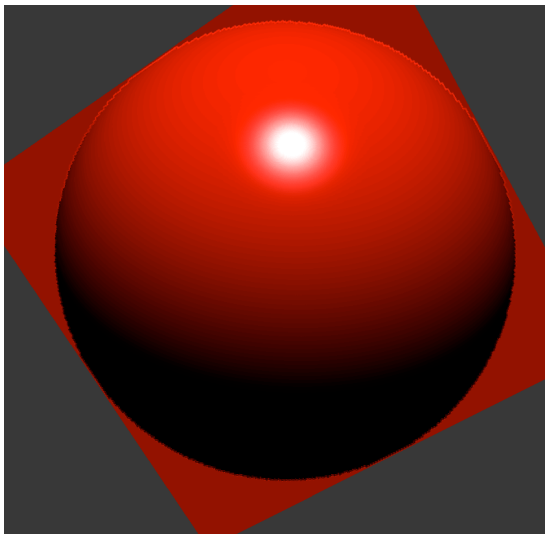
- Advanced technique for rendering translucent materials (skin, wax, milk, etc)
  - light enters material, bounces around, some exits with new properties
  - hold your hand up to a light, you'll notice a "reddish" glow

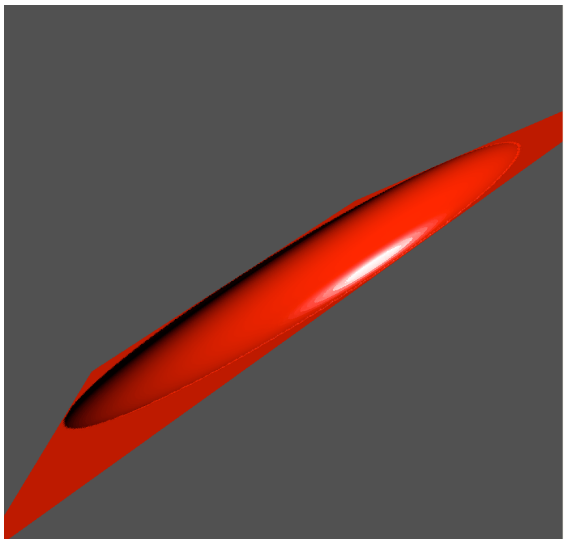
http://graphics.ucsd.edu/~henrik/

Real-time Rendering: Billboards

- Take a quad and slap a texture on it. Now we want it to face the camera. How do we do that?
  - … <your answer here>

- Bread-and-butter of older 3d games and still used extensively today
  - Monsters (think *Doom*)
  - Impostors (LOD)
  - **Text**
  - Faked smoke, fire, explosions, particle effects, halos, etc.

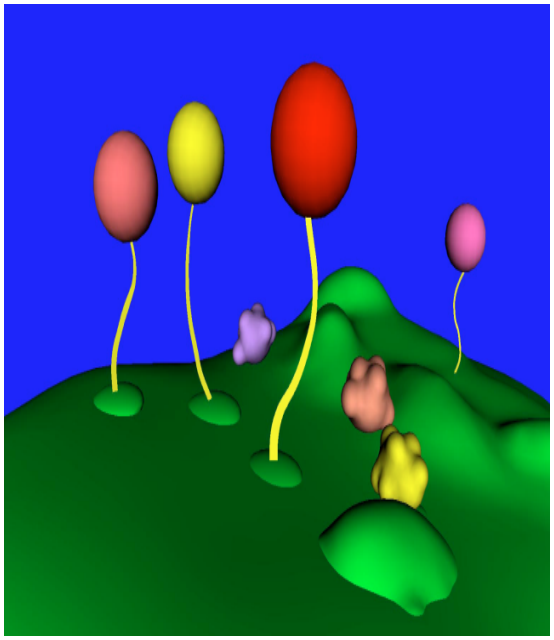




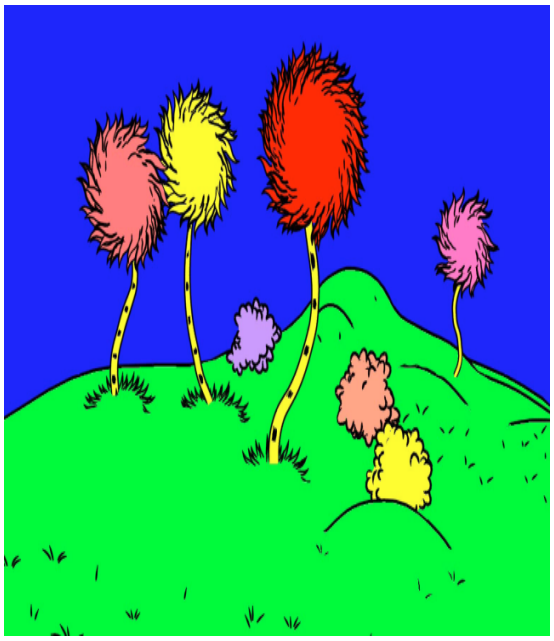
Boo!

Non-Realism in Rendering

- NPR (nonphotorealistic rendering)
- Algorithms to render images which look like they have been drawn



Valkyria Chronicles uses Sega's Canvas graphics engine



Physics: Physically-based

- Physically-based simulations (kinematics for rigid-body motion, dynamics for F = ma)
  - Hugely complex modeling problem
  - expensive, using space-time constraints, inverse kinematics, Euler and Runge-Kutta integration of forces, N^2-body problems. These can take a long time to solve
  - looks *fairly* convincing…but not quite real (yet)

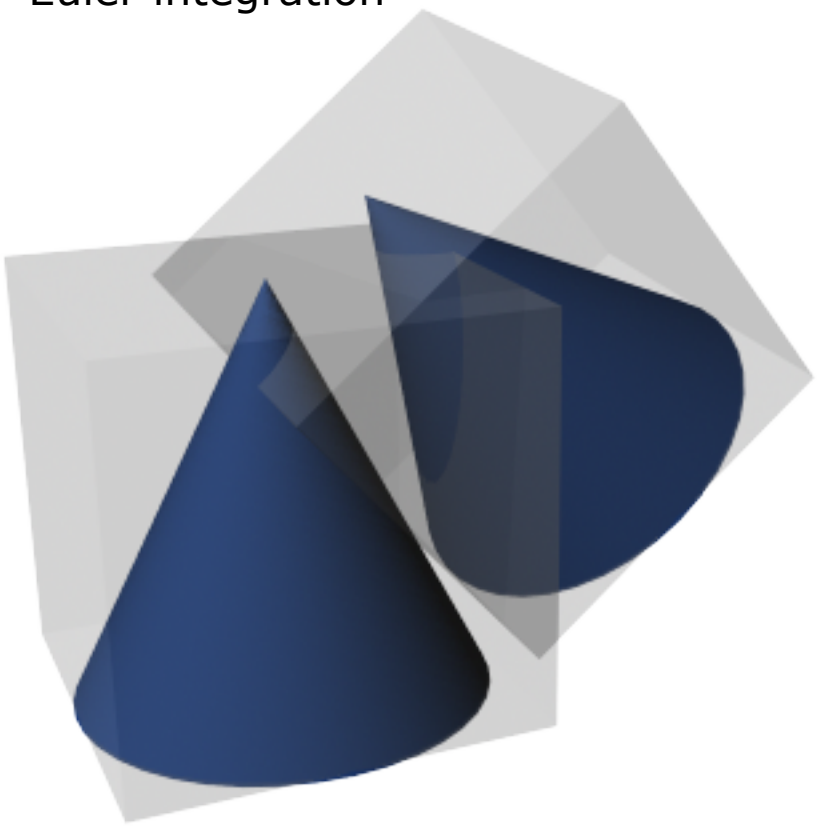
Physics: Motion Capture

- **Motion-capture**
  - capture positions and orientations of motion-trackers over time.
    - Trackers usually attached to joints of human beings performing complex actions.
  - once captured, apply these motions to a computer model
  - useful for a variety of characters with similar joint structure

Physics in Games

The Gracefully Degraded

- Break laws of physics (hopefully imperceptibly)
  - Simplify numerical simulation: consider fewer forces, use bounding boxes instead of precise collision detection, etc.
  - Decrease number of time steps used for Euler integration



Bounding Box

Rag Doll Physics

- Sick of seeing the same death-animation every time? Enter "rag doll physics."

- Real physics: computationally expensive, so lots of approximations (like bounding boxes) are made.
  - Use spatial data structure to speed up collision tests
- Particle systems
- But: most explosions and projectiles are scripted, not simulated
- Realistic physics != good game (*Trespasser*)

AI for Games

- And you thought *graphics* was hacky!
- The Old AI: Simple FSMs, Crash 'n' Turn
- The New AI: Glorified FSMs (scripting, navigation, goals…), A*

- AI continues to get better as people "read the literature" and the GPU lightens the load on the CPU
- Realistic AI != good game

Black and White (2000)

Networking for Games

- Packets *will* get lost: the internet is unpredictable
  - Use non-blocking I/O and multithreading
- TCP/IP is too slow!
  - Too much error correction, and way too generalized
  - Waits for response before sending next packet
  - Almost all games use UDP (User Datagram Protocol) instead, which has no error correction
- UDP does not guarantee reliability or ordering in the way that TCP does
  - datagrams (short messages exchanged by programs) may arrive out of order, appear duplicated, or go missing without notice
  - UDP avoids the overhead of checking whether every packet actually → faster and more efficient
  - time-sensitive applications often use UDP because dropped packets are preferable to delayed packets
- Using UDP, it's up to us to do the error correction
  - no one really had a good solution until… John Carmack with *QuakeWorld* (1996) set up the standard client/server model still used today

The Art Pipeline

- A good engine DOES NOT make a good game

- No game is good without artwork
  - *characters & storyline*
  - *environments*
  - *sound, music*
  - *animations*

The power of a good storyline: the Zork series

- Nowadays, the workflow for developing and integrating artwork into the game makes or breaks a game company
  - This may be one of the major reasons why EA has been so successful
  - EA uses licensed engines for all their games – the majority of the staff streams art content into the games in a very organized manner

Learn How to Work in a Team

- with other CS/CoE-ers: software management (cs1630)
- with Artists, Writers, non CS-ers in general: communication skills (cs1630 and cs1666 - every other year)

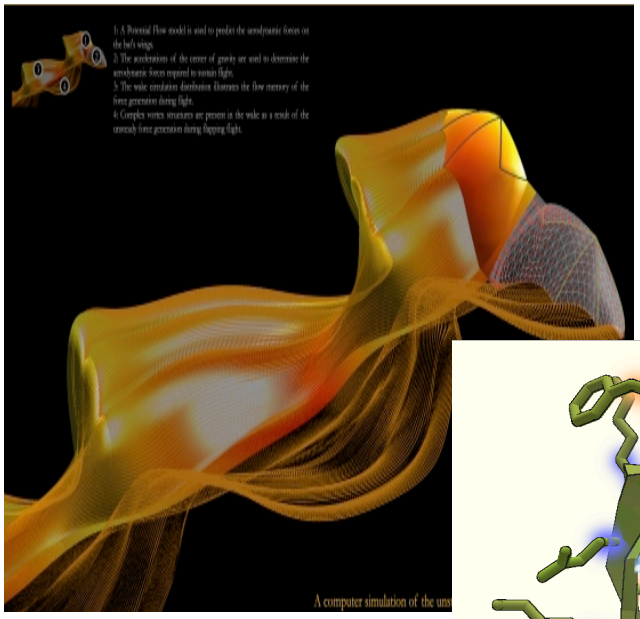
Doom III

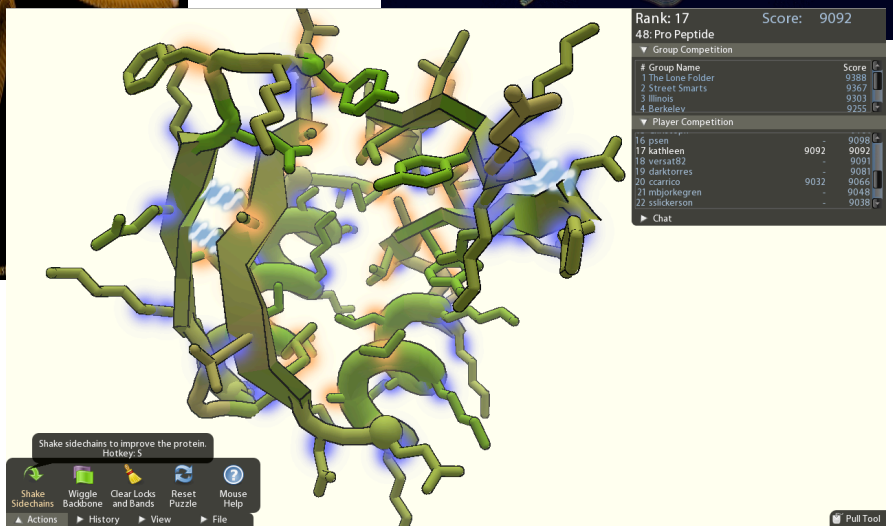
Getting a Job

- Try cs1666 first
- Add what you can: AI, networking
- Learn to work in a team
- Ways to make it easier:
  - find jobs in graphics, not games: user interfaces, data visualization, and education
  - intern, become a slave
- If you *really, really* want a job in games
  - prepare to be poor for a while
  - if you thought the programming workload in cs1566 was heavy… **beware**
- Start an online portfolio: write demos to get a company to look at you (see danieloliphant.com)
  - Your CS1566 code will work great for simple demos but not so much for real demos (i.e. your code isn't going to draw a *Quake* map)
  - Do an directed study; work on your own
  - There are faster ways to draw than glVertex

Modeling and Simulation

- 10 times the size and revenue of the gaming industry





www.fold.it or on youtube: computer game which enables novice users to contribute to important scientific research (protein folding).

- Serious Games
  - e.g.: simulations, training, education
- Talk to me about a capstone project or a research experience appointment

References

- *Real-Time Rendering* Second Edition
  - Becoming the bible for real-time graphics
- *Tricks of the 3d Gaming Gurus: Advanced 3d Graphics and Rasterization*
  - Great if you want to write your own rasterizer
- *Masters of Doom*
- *Game Developer Magazine*
- *Game Programming Gems* and *Graphics Gems* book series
- www.gamasutra.com
- www.flipcode.com
- www.gametutorials.com
- www.gamedev.net
- www.emuunlim.com/doteaters/

Summary

- Game Engines
  - the Game Loop
  - Realism and the Uncanny Valley
  - Rendering
  - Physics
  - AI
  - Networking
  - Art Pipeline

- Getting a job
- Modeling and Simulation

- References