January 27 -1

Graphics Intro

Administrivia
- Processing
- JavaScript

Discuss conversation about what is a good game / game design except...

What does it have to do with Tech?

Why do we need tech for Games?
Why does better tech ⇒ better games?
Why is games tech different than other tech?

Tech in Games: (why)

Tech-centric games
- UI / input device & design around
  Flashy graphics & show off
  Physics puzzles

Create complexity
Create richness

Immersive world
Set up story ⇒ why?
Detail i.e. realism ⇒ or "worldism"

Procedural creation

Provide effective interaction / feedback ⇒ FAST AI

Complexity of Systems - network (massive # of players, responsiveness)
Resources (assets, code, ...)
Game Tech vs Regular CS

Resource Limited - cost conscious vs. richness, complexity, ...
- development costs / resources can grow
- limited platforms (fixed) - consoles
- how to live within constraints

Interactive
- latency, bandwidth, frame-rate
- performance (fast)
  - also communication, storage, pre-computation

Correctness
- looks good consistent
- simulation, visually, ....
- fake it (big stuff off-screen)

Asset Rich
Multi-Disciplinary

Focus on Fun - faster / better / more doesn't always mean more fun
Why Graphics?
- Historically bottleneck
- Just repainting screen @ framerate
- Computationally intense
- Obvious improvements
- No limits to needs (complexity scales arbitrarily)
- Positive feedback: games → hardware → ideas
- Hard to do without
- Rapid evolution: CPU is just faster

Various Tradeoffs
See 2007 Notes

Key Ideas:

Approximation
Preparation
Amortization
Avoidance (don’t draw)

Pipelining / parallelism (exploit hardware) - Caches / memory hierarchy
Caching / pre-fetching / prediction

Making Flooding Faster
- Sub-linear drawing (if zoomed in)
- O(n^2) neighborhood tests
  - n-body methods vs. dynamism