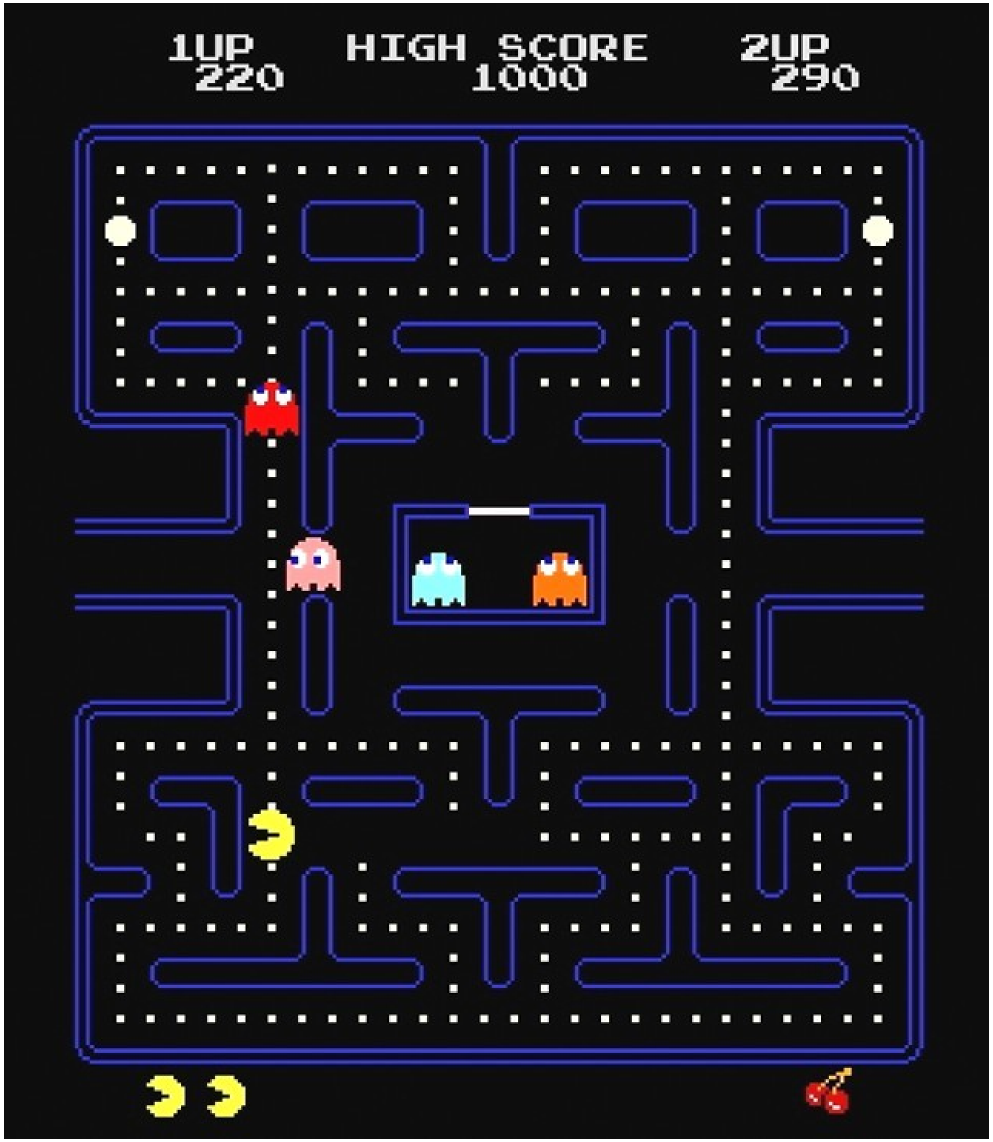
- Physics in games.
  - All games have "physics".
  - No game has "real physics".
- Different basic physics implementations.
  - Frame-synced.
  - Variable timestep.
  - Fixed timestep.
- Physics engines.

ALL GAMES HAVE "PHYSICS"



All games have "physics"

- All games have simplified physics.
  - Input, time, output.
  - Input causes reaction in game state.
  - Game state is represented in some way.

- All games are, deep down, turn-based.
  - Some turns just happen to take 10ms..

No game has "real physics"

- All game-physics simulations are limited.
- The more sophisticated physics you have, the more development issues you will face.
    - Exploding physics etc.
- On the other hand, some degree of "realistic" physics makes things more intuitive.

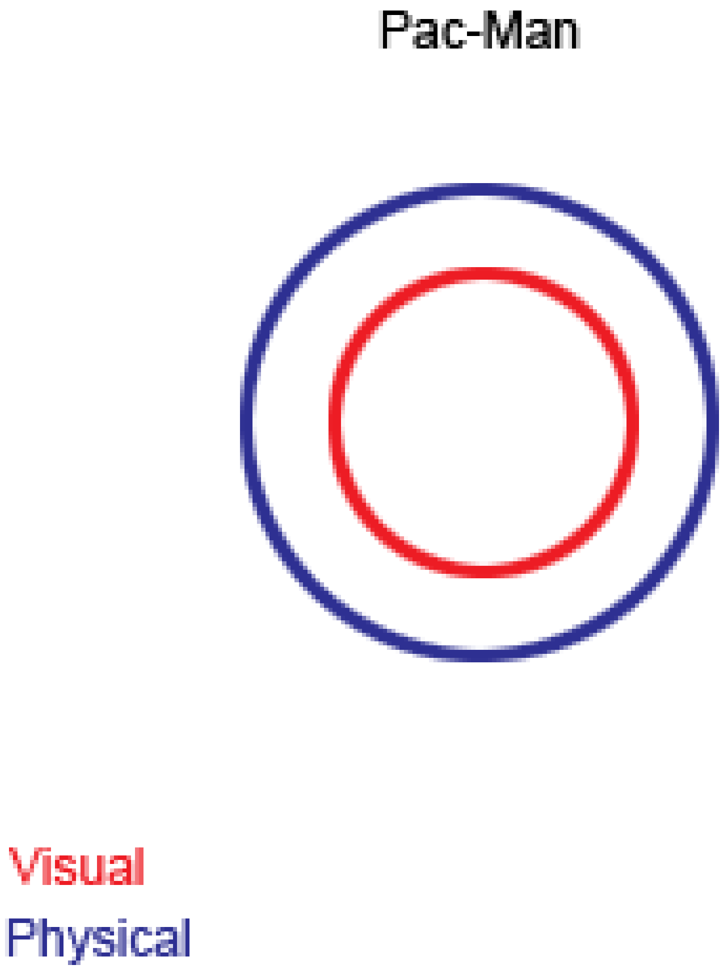
- The most important thing:

fun > realism

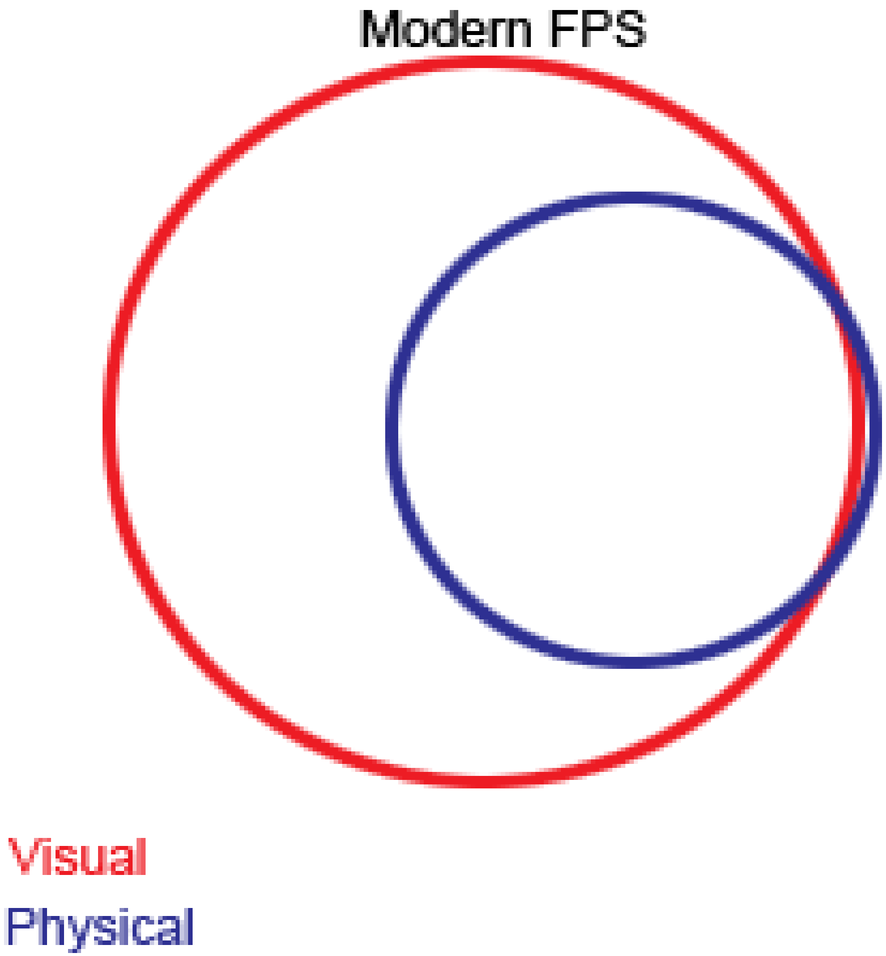
On Physics vs Representation

- Games have internal "physics" state.
  - This is what's "real".
- Player is shown (visual, aural, force feedback) some kind of representation of this.
  - This should, but never is, what the actual state is.

ON PHYSICS VS REPRESENTATION

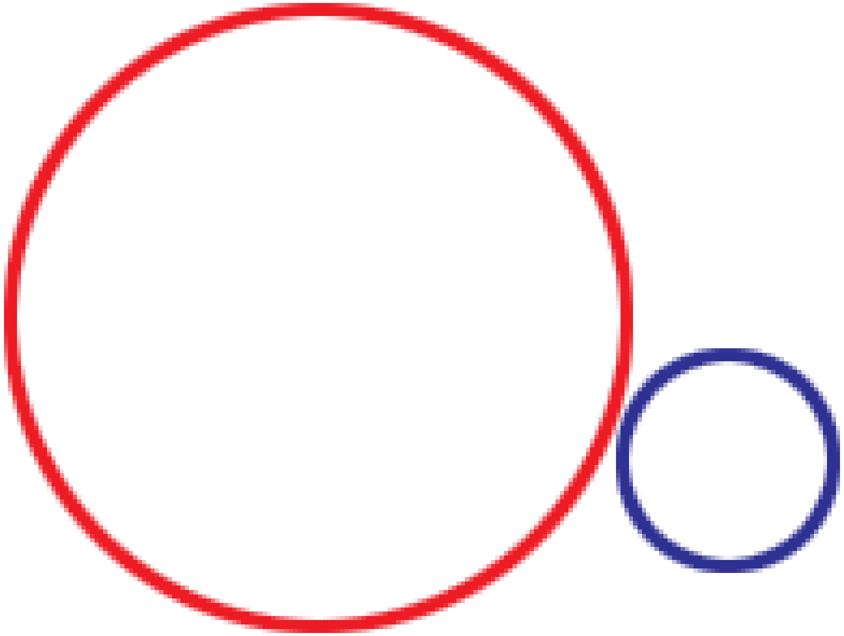


ON PHYSICS VS REPRESENTATION



On Physics vs Representation

Dragon's Lair



Visual

Physical

Frame-Synched Physics

- Physics is synchronized to display update rate.
- Used in old, limited, and sometimes in modern systems "pushing the limits".
- May have issues with NTSC vs PAL refresh rates, etc.
- Mostly a curiosity (for you, anyway).
- In practise, physics has a fixed amount of ms to be calculated.

Non-frame-synched physics

- Physics is calculated 0-n times per rendered frame.
- Basically how all games are made.
- No fixed time limit for physics.
  - Physics may (and in many cases will) affect framerate.

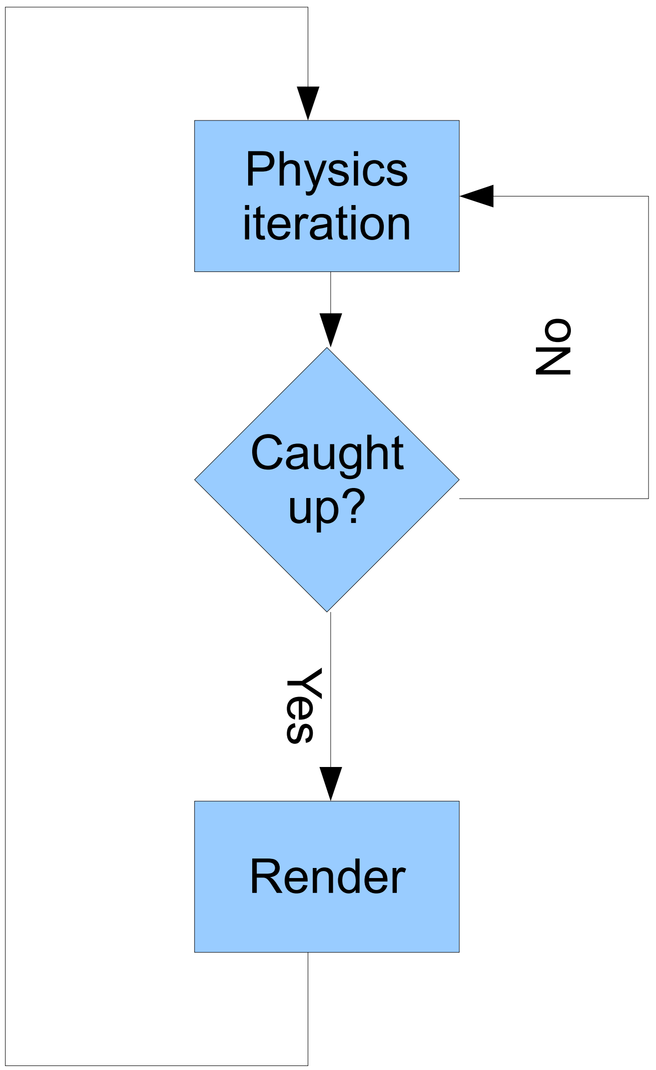
Variable Timestep

- Calculate time since last physics update.
- Use time as a multiplier in physics calculations.
- Pros:
  - Physics updated once per render cycle.
  - May be light, function well in low FPS scenarios.
- Cons:
  - Non-deterministic.
  - Very complex calculations (integration).
  - Bug-prone.

Fixed Timestep

- Run physics simulation at steady n Hz.
  - In practise: run physics several times per render cycle, as needed.
- Pros:
  - Deterministic.
  - Stable.
  - Simplified math (dt=1).
- Cons:
  - May be heavier, issues in low-FPS situations.

Fixed Timestep



Physics Engines

- For more sophisticated physics, many engines are available.
  - Even for free, except..
    - Learning a physics engine takes time.
    - Learning to fix issues with physics engines takes time.

- If you consider using a physics engine, remember:

fun > realism

PHYSICS AS THE GAME

- Creating simple physics simulation may result in a game:
  - Rope physics.
  - Spring physics.
  - "Sand" physics.
  - etc.

HOMEWORK

- Grab your physics book.
- Pick at least three equations.
- Write a short description how you could use it in a game (or as a game).