Building a Better Jump

J. Kyle Pittman
@PirateHearts
Co-founder, Minor Key Games
Hi

- I’m Kyle
- Hi Kyle

2007-2013

2013-20XX
Motivation

- Avoid hardcoding, guessing games
- Design jump trajectory on paper
- Derive constants to model jump in code
Motivation

● Has this ever happened to you?

● There’s GOT to be a better way!!
Assumptions

- Model player as a simple projectile
- Game state
  - Position, velocity integrated on a timestep
  - Acceleration from gravity
- No air friction / drag
Gravity

- Single external force
- Constant acceleration over time

\[ f''(t) = g \]
Integration

- Integrate over time to find velocity

\[ \int g \, dt = gt + v_0 \]
Integration

- Integrate over time again to find position

\[ \int (gt + v_0) \, dt = \]

\[ \frac{1}{2} gt^2 + v_0 t + p_0 \]
Projectile motion

\[ f(t) = \frac{1}{2} gt^2 + v_0 t + p_0 \]

- Textbox Physics 101 projectile motion
- Understand how we got here
Parabolas

- Algebraic definition
  - $f(x) = ax^2 + bx + c$
- Substituting

  $x \rightarrow t \quad b \rightarrow v_0$
  $a \rightarrow \frac{1}{2} g \quad c \rightarrow p_0$

  $f(t) = \frac{1}{2} gt^2 + v_0 t + p_0$
Properties of parabolas

- Symmetric
Properties of parabolas

- Geometric self-similarity
Properties of parabolas

- Shaped by quadratic coefficient \( a \rightarrow \frac{1}{2} g \)
Design on paper
Design on paper

\[ f(t) \]

\[ h \]

\[ 0 \quad t_h \quad 2t_h \quad t \]
Maths

- Derive values for gravity and initial velocity in terms of peak height and duration to peak

\[ f(t) = \frac{1}{2}gt^2 + v_0 t + p_0 \]

\[ f'(t) = gt + v_0 \]

\[ f''(t) = g \]
Initial velocity

Solve for $v_0$:

$$f'(t) = gt + v_0$$
$$f'(t_h) = 0$$
$$0 = gt_h + v_0$$
$$v_0 = -gt_h$$
Gravity

Known values:

\[ v_0 = -gt_h \]
\[ p_0 = 0 \]

Solve for \( g \):

\[
\begin{align*}
    f(t) &= \frac{1}{2}gt^2 + v_0t + p_0 \\
    f(t_h) &= h \\
    h &= \frac{1}{2}gt_h^2 + v_0t_h + p_0 \\
    h &= \frac{1}{2}gt_h^2 + (-gt_h)t_h + 0 \\
    h &= -\frac{1}{2}gt_h^2 \\
    g &= \frac{-2h}{t_h^2}
\end{align*}
\]
Back to init. vel.

Solve for $v_0$:

$$v_0 = -gt_h$$

$$g = \frac{-2h}{t_h^2}$$

$$v_0 = -\left(\frac{-2h}{t_h^2}\right)t_h$$

$$v_0 = \frac{2h}{t_h}$$
Review

\[ v_0 = \frac{2h}{t_h} \]

\[ g = \frac{-2h}{t_h^2} \]
Time → space

- Design with x-axis as distance in space
- Introduce lateral (foot) speed
- Keep horizontal and vertical velocity components separate
Parameters

- Foot speed \( v_x \)
- Time to peak \( t_h \)
- Distance to peak \( x_h \)
Time $\rightarrow$ space
Time $\rightarrow$ space
Maths

- Rewrite gravity and initial velocity in terms of foot speed and lateral distance to peak of jump
Maths

\[ v_0 = \frac{2h}{t_h} \]

\[ g = \frac{-2h}{t_h^2} \]

\[ t_h = \frac{x_h}{v_x} \]

\[ v_0 = \frac{2hv_x}{x_h} \]

\[ g = \frac{-2hv_x^2}{x_h^2} \]
Review

\[ v_0 = \frac{2hv_x}{x_h} \]

\[ g = -\frac{2hv_x^2}{x_h^2} \]
Breaking it down

- Real world: Projectiles always follow parabolic trajectories.
- Game world: We can break the rules in interesting ways.
- Break our path into a series of parabolic arcs of different shapes.
Breaks

- Maintain continuity in position and velocity
  - Trivial in implementation
- Choose a new gravity to shape our jump
Fast falling
Variable height jumping

Position

Velocity / Acceleration
Double jumping
Integration

- Put our gravity and initial velocity constants to use in practice
- Integrate from a past state to a future state over a time step
Integration

Current state

Next state?

position

velocity

acceleration

acceleration

timestep $\Delta t$
Euler

- **Pseudocode**
  
  \[
  \text{pos} += \text{vel} \times \Delta t \\
  \text{vel} += \text{acc} \times \Delta t
  \]

- Easy
- Unstable
- We can do better
Runge-Kutta (RK4)

- The “top-shelf” integrator.
- No pseudocode here. :V
- Gaffer on Games: “Integration Basics”
- Too complex for our needs.
Velocity Verlet

- Pseudocode
  
  \[
  \begin{align*}
  \text{pos} & \texttt{ += vel} \Delta t + \frac{1}{2} \text{acc} \Delta t^2 \\
  \text{new_acc} & = f(\text{pos}) \\
  \text{vel} & \texttt{ +=} \frac{1}{2} (\text{acc} + \text{new_acc}) \Delta t \\
  \text{acc} & = \text{new_acc}
  \end{align*}
  \]
Observations

- Similarity to projectile motion formula
- What if our acceleration were constant?
- We could integrate with 100% accuracy
Assuming constant acceleration
Assuming constant acceleration

- Pseudocode
  
  \[
  \text{pos} += \text{vel} \cdot \Delta t + \frac{1}{2} \text{acc} \cdot \Delta t \cdot \Delta t \\
  \text{vel} += \text{acc} \cdot \Delta t
  \]

- Trivially simple change from Euler
- 100% accurate as long as our acceleration is constant
Near-constant acceleration

- What if we don’t change a thing?
- The error we accumulate when our acceleration does change (versus Velocity Verlet) will be:
  - $\Delta\text{acc} \times \Delta t \times \Delta t$
  - Acceptable?
The takeaway

- Design jump trajectories as a series of parabolic arcs
- Can author unique game feel
- Trust result to feel grounded in physical truths
Questions?

- In practice: *You Have to Win the Game* (free game PLAY IT PLAY MY THING)
- The Twitters: [@PirateHearts](http://minorkeygames.com)
- [http://minorkeygames.com](http://minorkeygames.com)
- [http://gunmetalarcadia.com](http://gunmetalarcadia.com)