Game Physics

GPR350, Fall 2019
Daniel S. Buckstein

Integration
Week 2

License

Locomotion Systems & Integration

- Physics: continuous vs. discrete integration
  - Fundamental calculus review
- Integration methods for locomotion
  - 1st order
  - 2nd order
  - Displacement
  - Other methods

Basic Calculus

- Calculus is the mathematical study of *change*

- Derivative: the rate of change of a dependent variable as an independent variable changes

Basic Calculus

- Derivatives:
- Example: for the equation $y = 2x - 3$, the dependent variable is *y* and the independent variable is *x*
- We control *x*, the function tells us what *y* is
- As *x* changes, *y* will have a constant rate of change: 2

Basic Calculus

- Derivatives:

- We say that

$$\frac{dy}{dx} = \frac{d}{dx}(2x - 3) = 2$$

- There are rules to solve derivatives quickly, but this is not what we are concerned with…

Basic Calculus

- Derivatives:

- In the context of animation, the derivative of some function ‘f’ with respect to time ‘t’

$$\frac{df}{dt}$$

Basic Calculus

- Application of calculus: ***physics***
- The change in *distance* over time is called speed (scalar)
- The change in *position* over time is called velocity (vector)
- The change in *speed* or *velocity* over time is called acceleration

Basic Calculus

- Application of calculus: ***physics***
- Position as a function of time: $x(t)$
- Velocity as a function of time: $\dot{x}(t)$ or $v(t)$
- Accel. as a function of time: $\ddot{x}(t)$ or $a(t)$

Basic Calculus

- Application of calculus: ***physics***
- Position as a function of time: $x(t)$
- Velocity as the derivative of position:

$$v(t) = x'(t) = \frac{dx}{dt}$$

- Acceleration as the derivative of velocity:

$$a(t) = v'(t) = \frac{dv}{dt} = x''(t) = \frac{d^2x}{dt^2}$$

Basic Calculus

- Core formula in calculus: *Difference quotient*
- Average rate of change:

$$m_{\text{secant}} = \frac{\Delta f(x)}{\Delta x} = \frac{f(x) - f(a)}{x - a}$$

- Instantaneous rate of change:

$$m_{\text{tangent}} = \lim_{x \to a} \frac{f(x) - f(a)}{x - a}$$

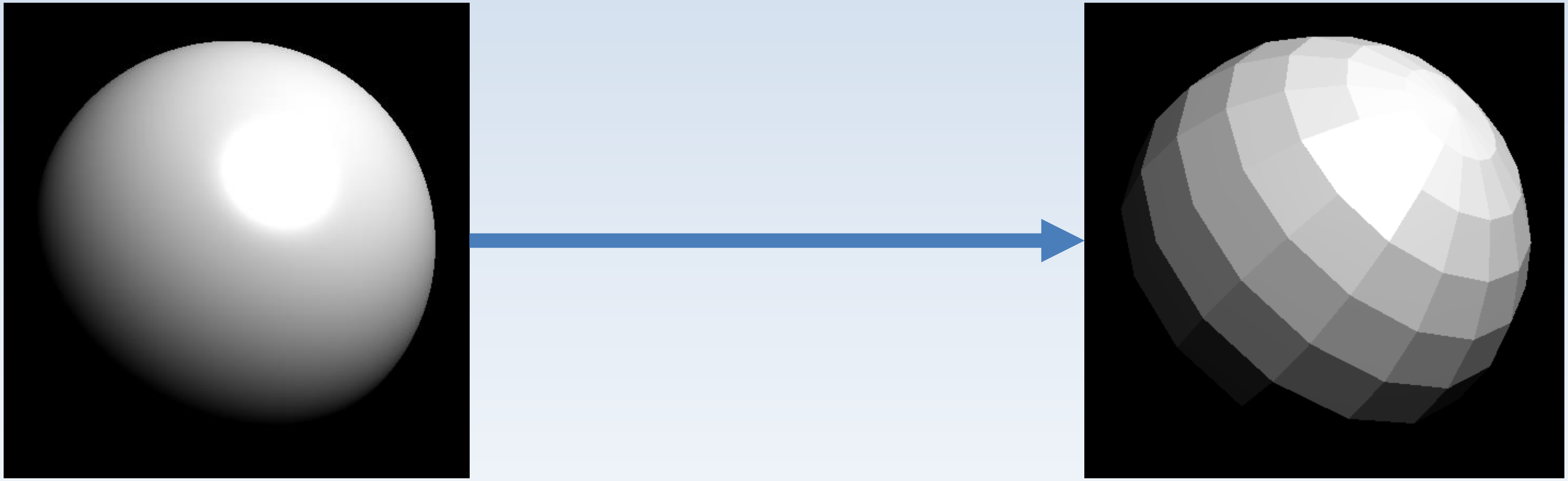
$$\equiv \lim_{\Delta x \to 0} \frac{f(x + \Delta x) - f(x)}{\Delta x}$$

Basic Calculus

- Note the limit: $\lim_{\Delta x \to 0}$
- As the change in our independent variable approaches zero, the closer we get to ***instantaneous***
- This is the ultimate goal of physics engines to make things as accurate as possible: ***minimize the independent variable (time step)***

Basic Calculus

- Also note: Physics world != Graphics world



Continuous, "physical" definition of a sphere:

$$f(r, \theta, \varphi)$$
$$\{\, r > 0; \quad 0 \le \theta \le 2\pi; \quad 0 \le \varphi \le \pi \,\}$$

Discrete samples give us explicit vertices!!!

$$x = x_0 + r\cos\theta\sin\varphi$$
$$y = y_0 + r\sin\theta\sin\varphi$$
$$z = z_0 + r\cos\varphi$$

Locomotion Systems

- Animation: bringing life to inanimate things
- The illusion of life

- ***Kinematics***: the *study of motion*
- How things move, description of motion
- Why things move
- What makes them move

Locomotion Systems

- ***Locomotion***: the movement of an object, method of getting from point A to point B
- Requires effort, actual physical movement
- Animation: the illusion of life… not the same…
- It would be very awkward if a human just slid across the ground in an upright position…
- …or did not move at all…
  - https://www.youtube.com/watch?v=13YlEPwOfmk

Locomotion Systems

- Key diff between physics and animation:
- ***Interpolation vs. Integration***
- All disciplines pivot around the concept of the ***derivative*** or rate of change: $x' = f'(t)$
- In physics, we *integrate* the derivative to go from one known state to the next unknown
- In animation, we *interpolate* between known states to emulate change over time

Basic Calculus

- Video games revolve around the concept of *integration*
- The *integral* is fundamentally the opposite of the derivative
- If the rate of change is known, we add it to the previous value of the function we want to find a value for!!!

Locomotion Systems

- Intro integration (continuous):

$$\frac{d}{dt}x(t) = x'(t) = v(t) \;\rightarrow\; \int v(t)\,dt = x(t)$$

- *x* is position, *v* is velocity, *t* is time (indep. var.)
- *Integral* just means *repeatedly add in small steps* of dt
- …we're not interested in continuous math…

Locomotion Systems

- ***Numerical integration*** (discrete method):

$$\underbrace{x_{t+dt}}_{\substack{\text{next}\\\text{position*}}} = \underbrace{x_t}_{\substack{\text{current}\\\text{position}}} + \underbrace{v_t}_{\substack{\text{current}\\\text{velocity}}} \overbrace{dt}^{\text{differential: change in time (independent var.)}}$$

- Using functions (says the same thing):

$$x(t + dt) = x(t) + v(t)dt$$

**Position* is usually '*x*' in physics, but with numerical integration, the formula can apply to anything!

Locomotion Systems

- Can also write our position formula like this:

$$x_t = x_{t-dt} + v_{t-dt}dt$$

$$x(t) = x(t - dt) + v(t - dt)dt$$

- Exactly the same, just interpreting it as using *last known values* to compute *current values*!

Locomotion Systems

- Graphical example:

$$x_{t+dt} = x_t + v_t dt$$

$$x_0 = (0, 0)$$

$$v_0 = (2, 1)$$

$$x_{0+0.2} = x_0 + v_0(0.2)$$

$$x_{0.2} = (0, 0) + (0.4, 0.2)$$

$$x_{0.2} = (0.4, 0.2)$$

Let's assume our time step each update is one fifth of a second:

$$dt = 0.2$$

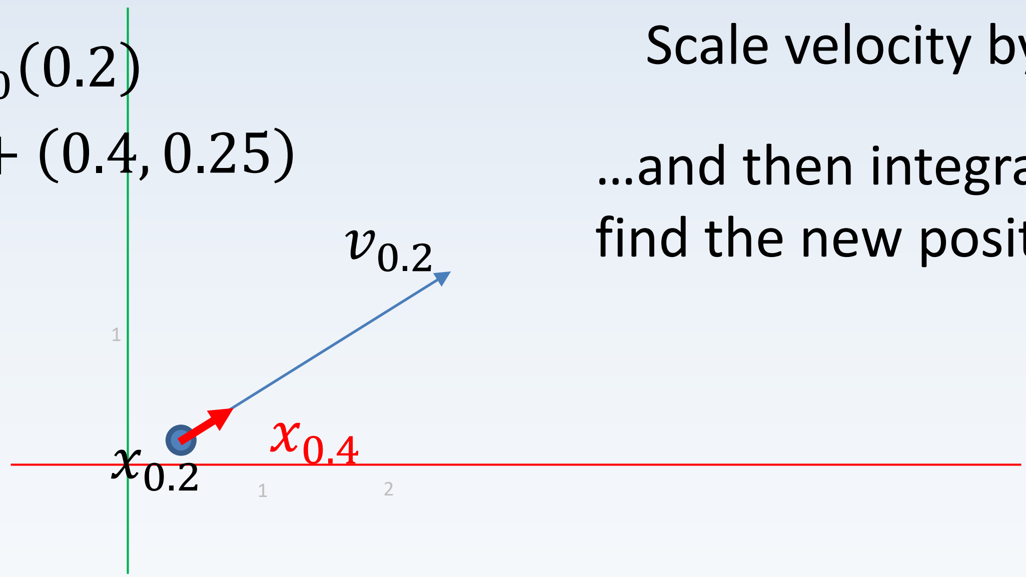
Scale velocity by *dt*

…and then integrate it to find the new position

Locomotion Systems

- Graphical example:

$$x_{t+dt} = x_t + v_t dt$$
$$x_{0.2} = (0.4, 0.2)$$
$$v_{0.2} = (2, 1.2)$$
$$x_{0.2+0.2} = x_0 + v_0(0.2)$$
$$x_{0.4} = (0.4, 0.2) + (0.4, 0.25)$$
$$x_{0.4} = (0.8, 0.45)$$



Let's assume our time step each update is one fifth of a second:
$$dt = 0.2$$

Scale velocity by *dt*

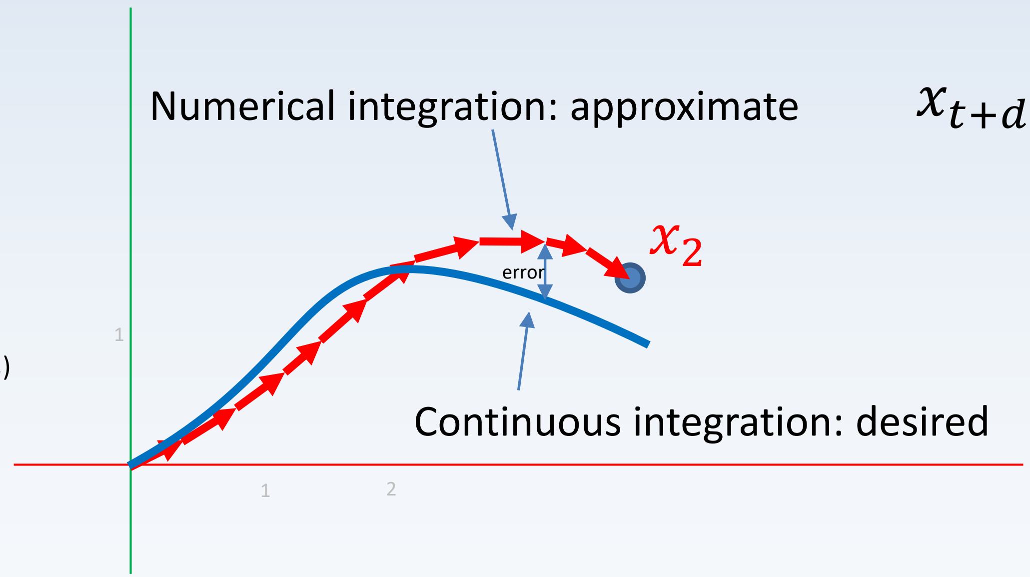
...and then integrate it to find the new position

Locomotion Systems

- Graphical example:
- PROBLEM:
- This is an *approximation*!!!

When $t = 2$ →

(after 10 steps when $dt = 0.2$)



$$x_{t+dt} = x_t + v_t dt$$

$$x_t = \int v_t dt$$

Locomotion Systems

- This is “first-order numerical integration”
- It is an *approximation* of what the curve should look like, reconstructed…
- …but mathematically there is a lot of *error*!
- First-order because we are integrating *only the first derivative* of position (velocity)
- How do we get a more accurate result and minimize error???

Locomotion Systems

- ***Second-order numerical integration***:

$$\frac{d}{dt}v(t) = v'(t) = x''(t) = a(t)$$

$$\int a(t)\,dt = v(t)$$

$$\int\int a(t)\ dt^2 = x(t)$$

Locomotion Systems

- ***Second-order numerical integration***:
- *Acceleration* integrates into *velocity* the same way *velocity* integrates into *position*:

$$v_{t+dt} = v_t + a_t dt$$

or

$$v_t = v_{t-dt} + a_{t-dt} dt$$

- Can be substituted into our position formula:

$$x_{t+dt} = x_t + v_t dt$$

Locomotion Systems

- Results in second-order formula:

$$x_{t+dt} = x_t + (v_{t-dt} + a_{t-dt}dt)dt$$

$$\boldsymbol{x_{t+dt} = x_t + v_{t-dt}dt + a_{t-dt}dt^2}$$

- Problem: we need to know current *and* previous values!!!

Locomotion Systems

- ***Displacement***:
- Kinematic formula for displacement: an application of “changing position”

$$\boldsymbol{s_t = \frac{1}{2}(v_t + v_{t+dt})}$$

- The average of two subsequent velocities

Locomotion Systems

- **Integrating displacement**:
- Step 1: Replace *velocity* in first-order equation with *displacement*:

$$x_{t+dt} = x_t + \boldsymbol{\frac{1}{2}}(\boldsymbol{v_t} + \boldsymbol{v_{t+dt}})dt$$

Locomotion Systems

- **Integrating displacement**:
- Step 2: Substitute formula for ***next velocity***:

$$x_{t+dt} = x_t + \frac{1}{2}(v_t + [\boldsymbol{v_{t+dt}}])dt$$

Equivalent to first-order for *velocity*: $\longrightarrow$ $\boldsymbol{v_{t+dt} = v_t + a_t dt}$

$$x_{t+dt} = x_t + \frac{1}{2}(v_t + [\boldsymbol{v_t + a_t dt}])dt$$

Locomotion Systems

- **Integrating displacement**:
- Step 3: Expand and simplify:

$$x_{t+dt} = x_t + \frac{1}{2}(\boldsymbol{v_t} + \boldsymbol{v_t} + a_t dt)dt$$

$$= x_t + \frac{1}{2}(\boldsymbol{2v_t} + a_t dt)dt$$

$$\boldsymbol{x_{t+dt} = x_t + v_t dt + \frac{1}{2} a_t dt^2}$$

Locomotion Systems

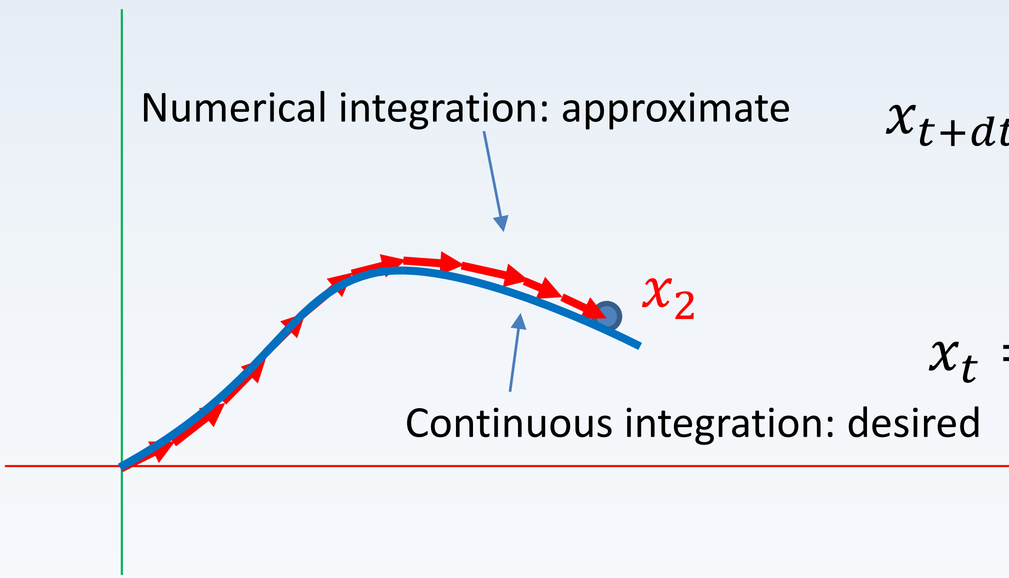
- **Integrating displacement:**

$$x_{t+dt} = x_t + v_t dt + \frac{1}{2} a_t dt^2$$

$$v_{t+dt} = v_t + a_t dt$$

When $t = 2$ →

(after 10 steps when $dt = 0.2$)



$$x_{t+dt} = x_t + v_t dt + \frac{1}{2} a_t dt^2$$

$$x_t = \int v_t dt = \int \int a_t \; dt^2$$

Locomotion Systems

- ***Data structure***:
- Something that stores our positional data…

```
struct Particle
{
    vec3 position, velocity, acceleration;
};
```

- Possible update functions:

```
void particleUpdatePos1stOrder( Particle *p, float dt );
void particleUpdatePos2ndOrder( Particle *p, float dt );
void particleUpdatePosDisplace( Particle *p, float dt );
```

Locomotion Systems

- ***Euler's method***:
- The simple integration methods are instances of a greater set of problems: *Euler's method*

$$\frac{dx}{dt} = f(x_t)$$

← fancy way of describing the *velocity of x*

$$\boldsymbol{x_{t+dt} = x_t + f(x_t)dt}$$

(psssssst… it's a more general form of first-order!)

Locomotion Systems

- ***Euler's method***:
- Category of "one-step integration methods"
- …what about "two-step" or "multi-step"?
- $x_{t+dt} = \cdots$
- $x_{t+2dt} = \cdots$
- $x_{t+3dt} = \cdots$
- Simple example: 2nd-order integration

Locomotion Systems

- ***Adams-Bashforth methods***:

$$x_{t+dt} = x_t + f(x_t)dt$$ ← literally Euler's method

$$x_{t+2dt} = x_{t+dt} + \left[\frac{3}{2}f(x_{t+dt}) - \frac{1}{2}f(x_t)\right]dt$$

$$x_{t+3dt} = x_{t+2dt} + \left[\frac{23}{12}f(x_{t+2dt}) - \frac{4}{3}f(x_{t+dt}) + \frac{5}{12}f(x_t)\right]dt$$

…etc.

Locomotion Systems

- ***Adams-Bashforth methods***:
- Two-step example using discrete values:

$$\frac{dx}{dt} = f(x_t) \rightarrow f(x_t) = v_t$$

$$x_{t+2dt} = x_{t+dt} + \left[\frac{3}{2} v_{t+dt} - \frac{1}{2} v_t\right] dt$$

$$\boldsymbol{x}_{\boldsymbol{t+dt}} = \boldsymbol{x}_{\boldsymbol{t}} + [\boldsymbol{3v}_{\boldsymbol{t}} - \boldsymbol{v}_{\boldsymbol{t-dt}}]\frac{\mathbf{1}}{\mathbf{2}}\boldsymbol{dt}$$

- Data structure needs to store more info…

Locomotion Systems

- ***Runge-Kutta methods***:
- Have fun.
    - Actually though, we’ll talk about it later.
- Neat helpful system: “function” data structure
- Stores a “state” and a pointer to its “derivative”
- “Integrator” function is an algorithm that solves the state using derivative state!

The end.

- Questions? Comments? Concerns?