Physically-based Modelling

Mass, Force, and Time

Young-Min Kang
Tongmyong University

Mass

- Mass
  - Resistance to being accelerated by force
  - accumulated density (kg/m^3)
    - integration of density

$$m = \int \rho dV$$

Centre of Mass

- theory

$$x_c = \frac{\int x_0 dm}{m}$$

$$y_c = \frac{\int y_0 dm}{m}$$

$$z_c = \frac{\int z_0 dm}{m}$$

$$c_g = \frac{\sum (c_{g_i} m_i)}{m}$$

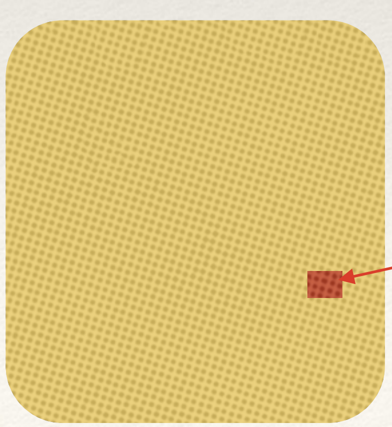
- computation $\longrightarrow$

$$x_c = \frac{\sum x_i m_i}{\sum m_i}$$

$$y_c = \frac{\sum y_i m_i}{\sum m_i}$$

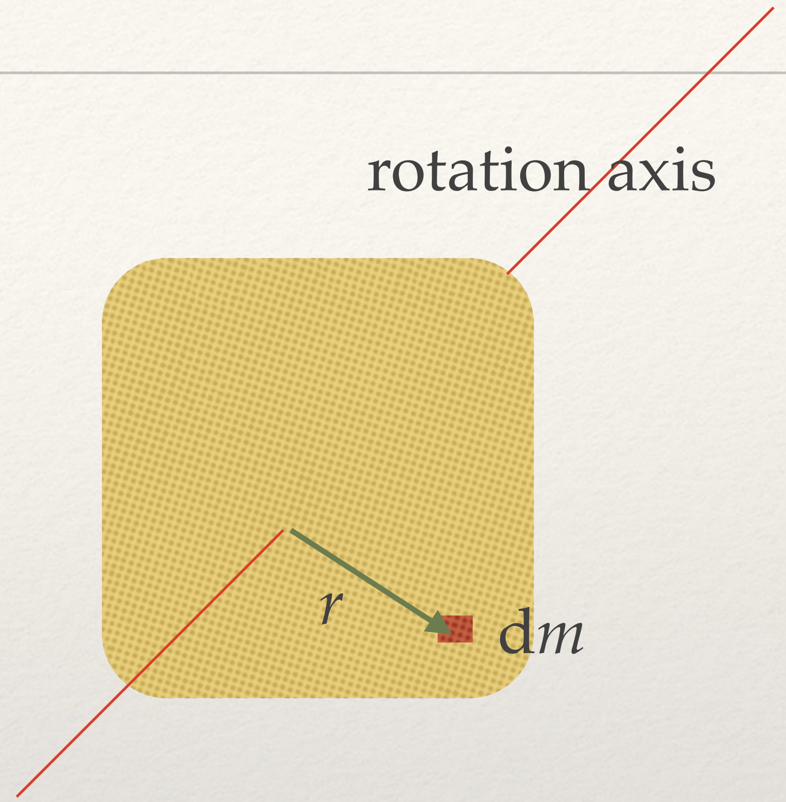
$$z_c = \frac{\sum z_i m_i}{\sum m_i}$$

V



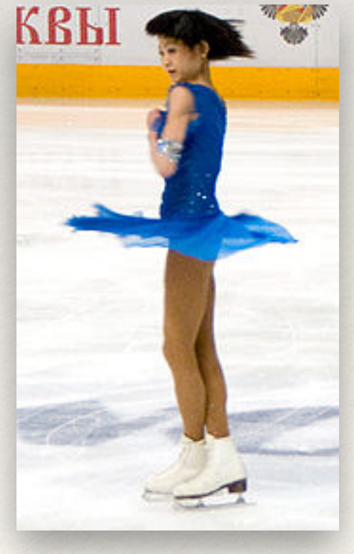
dm at (x_0, y_0, z_0)

Moment of Inertia

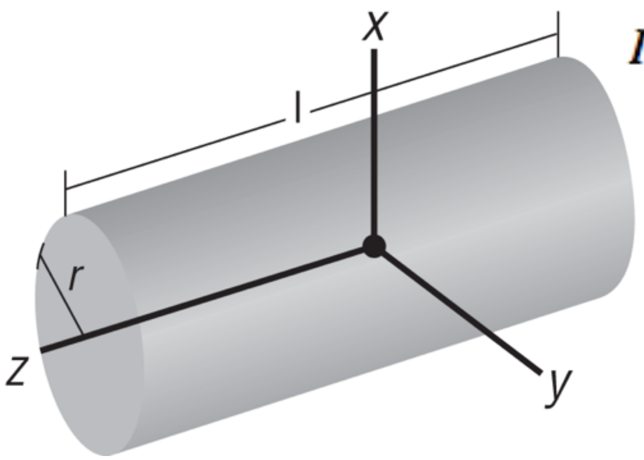


- angular mass
  - rotation axis is important
  - resistance to angular acceleration along the axis

$$I = \int_m r^2 \mathrm{d}m$$

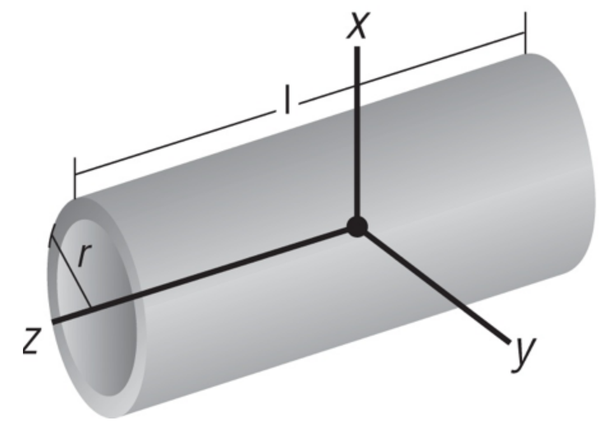


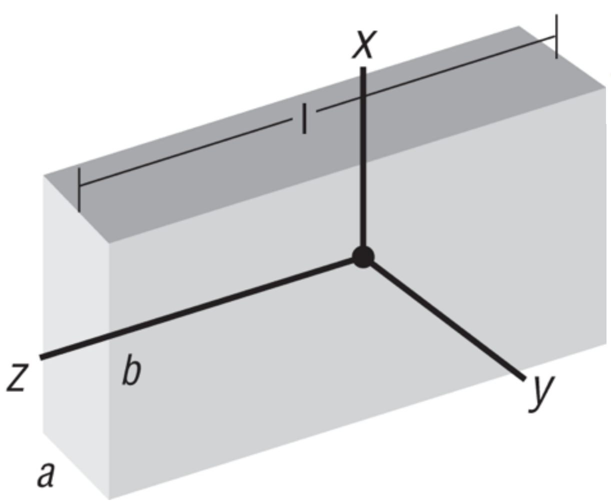
Examples



$$I_{xx} = I_{yy} = (1/4)\, mr^2 + (1/12)\, ml^2 ;\ I_{zz} = (1/2)\, mr^2$$

$$I_{xx} = I_{yy} = (1/4)\, mr^2 + (1/12)\, ml^2 ;\ I_{zz} = (1/2)\, mr^2$$

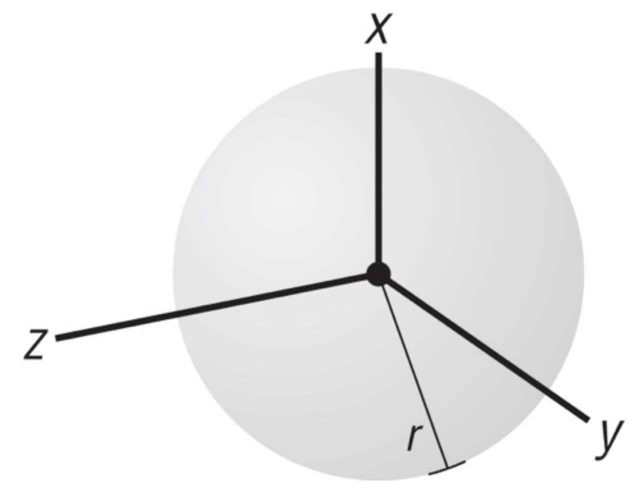




$$I_{xx} = I_{yy} = (1/4)\, mr^2 + (1/12)\, ml^2 ;\ I_{zz} = (1/2)\, mr^2$$

sphere $I_{xx} = I_{yy} = I_{zz} = (2/5)\, mr^2$

spherical shell $I_{xx} = I_{yy} = I_{zz} = (2/3)\, mr^2$



Newton's 2nd law of motion

- force = mass x acceleration

$$\mathbf{f} = m\mathbf{a}$$

$$\sum f_x = ma_x$$
$$\sum f_y = ma_y$$
$$\sum f_z = ma_z$$

- total force $\mathbf{f}_{total}$
    - sum of all the exerted forces $$\mathbf{f}_{total} = \sum_{i=1}^{n} \mathbf{f}_i$$
    - acceleration is determined by the total force

$$\mathbf{a} = \frac{\mathbf{f}_{total}}{m}$$

Linear Momentum and its Derivative

- Linear Momentum: **G**
  - mass x velocity
  - $\mathbf{G} = m\mathbf{v}$
- Derivative of Linear Momentum with respect to time

$$\frac{d\mathbf{G}}{dt} = \frac{dm\mathbf{v}}{dt} = m\frac{d\mathbf{v}}{dt} = m\mathbf{a}$$

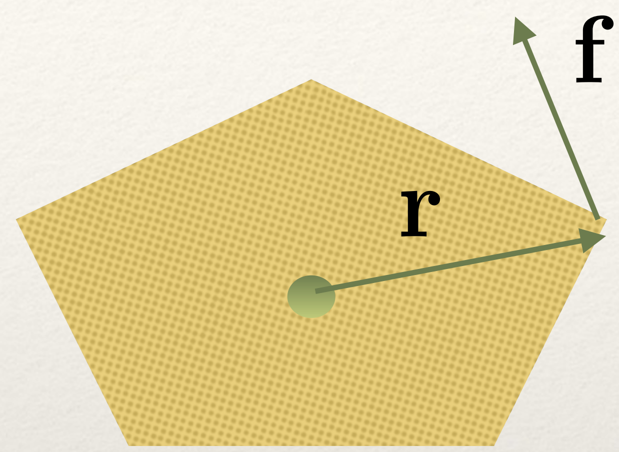
  - therefore,

$$\frac{d\mathbf{G}}{dt} = \sum \mathbf{f}$$

Rotational Motion

- Torque: τ
  - "force" in rotation

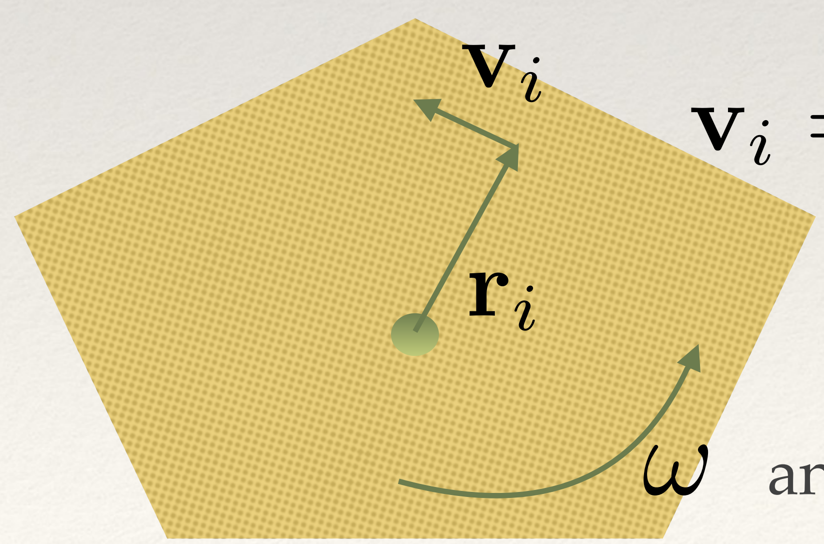
$$\tau = \mathbf{r} \times \mathbf{f}$$



- angular momentum
  - sum of moments of momentum of all particles

$$\mathbf{H} = \sum \mathbf{r}_i \times m_i \mathbf{v}_i$$

$$\mathbf{H} = \sum \mathbf{r}_i \times m_i (\omega \times \mathbf{r}_i)$$



$$\mathbf{v}_i = \omega \times \mathbf{r}_i$$

angular velocity

Angular Momentum = A. Mass x A. Velocity

- Linar Momentum: **G**
  - $\mathbf{G} = m\mathbf{v}$
- Angular Momentum

$$\mathbf{H} = \sum \mathbf{r}_i \times m_i(\omega \times \mathbf{r}_i)$$

$$\mathbf{H} = \int \omega \mathbf{r}^2 dm$$
$$= \omega \int \mathbf{r}^2 dm$$
$$= \omega \mathbf{I} = \mathbf{I}\omega$$

Inertia Tensor
(Angular Mass)

Angular Velocity

Derivative of Angular Momentum

- d**G** / dt = total force
- d**H** / dt = total torque

$$\frac{d\mathbf{H}}{dt} = \frac{d\mathbf{I}\omega}{dt} = \mathbf{I}\frac{d\omega}{dt} = \mathbf{I}\alpha$$

$$\longrightarrow \quad \sum \tau = \mathbf{I}\alpha \quad \longrightarrow \quad \alpha = \mathbf{I}^{-1}\sum \tau$$

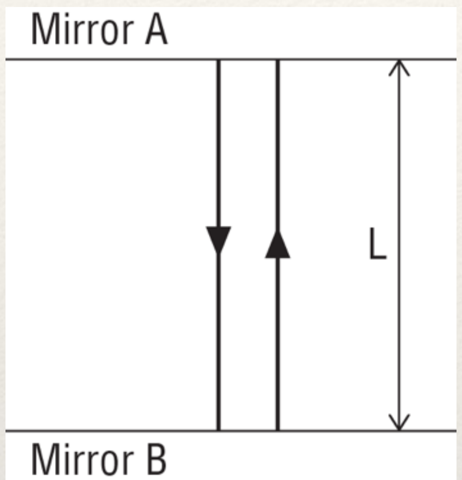
Tensor

- Tensor
  - mathematical expression that has magnitude and direction
  - its magnitude may not be unique depending on the direction
  - typically used to represent properties of materials where these properties have different magnitudes in different directions.
- Isotropic vs. Anisotropic
  - isotropic: properties are the same in all direction
  - anisotropic: properties vary depending on direction
- Moment of Inertia
  - inertia tensor (in 3D)
  - nine components to fully describe it for any arbitrary rotation. (3x3 matrix)
  - property of the body that varies with the axis of rotation.

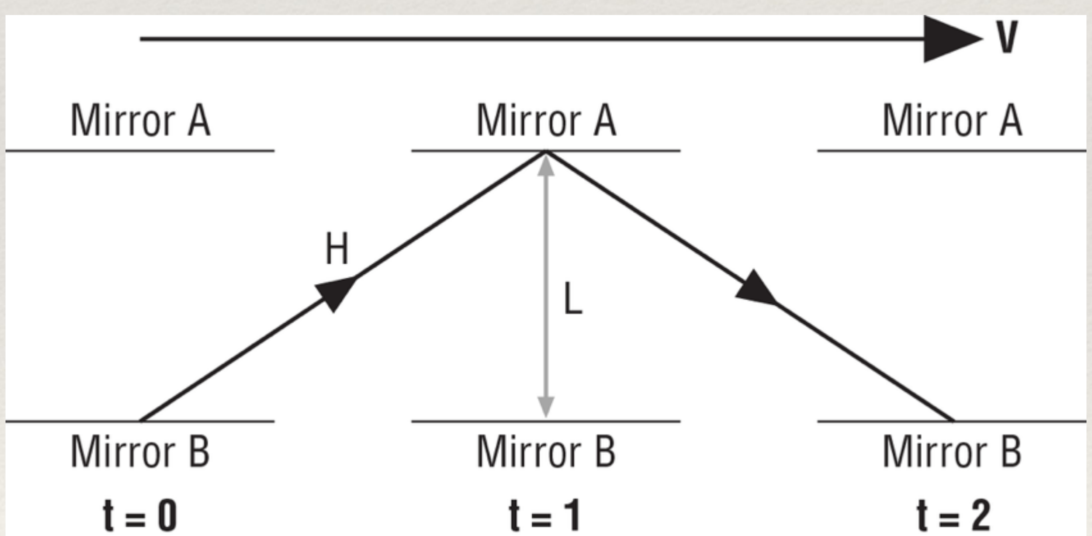
Time

- Classic Physics
  - Time is “constant”
- Modern Physics
  - Time is “variable”
  - “Speed of Light” is constant: c
  - c = 299,792,458 m/s

Relativistic Time



$$c = 2L/\Delta t$$



mirrors are travelling in a spaceship

in the spaceship

$$c = 2L/\Delta t_s$$

on earth

$$c = 2H/\Delta t_e$$

if c is constant
time should be DIFFERENT

Time Dilation

- Simple geometry

$$H^2 = L^2 + \mathbf{v}^2 \Delta t_e^2 \quad \Longrightarrow \quad \left(\frac{c\Delta t_e}{2}\right)^2 = L^2 + \left(\frac{\mathbf{v}\Delta t_e}{2}\right)^2$$

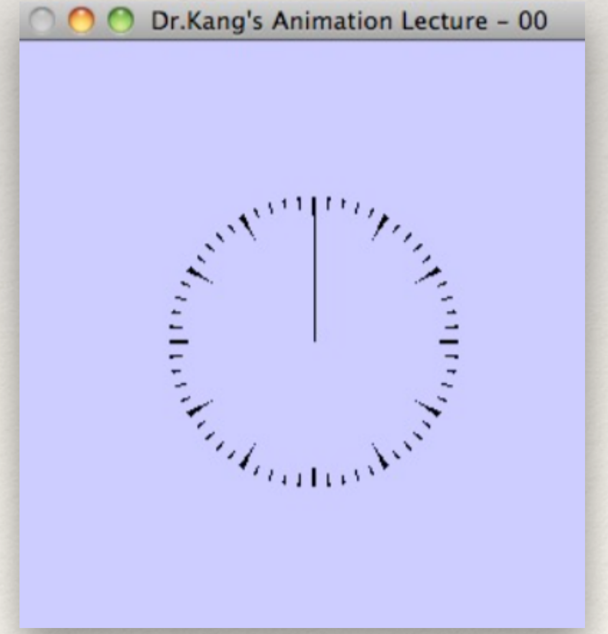
- time dilation

$$\boxed{\begin{aligned} 4L^2 &= c^2 \Delta t_e^2 - \mathbf{v}^2 \Delta t_e^2 \\ 4L^2 &= (c^2 - \mathbf{v}^2)\Delta t_e^2 \\ \frac{4L^2}{c^2} &= (1 - \frac{\mathbf{v}^2}{c^2})\Delta t_e^2 \end{aligned}} \quad \Longrightarrow \quad \boxed{\begin{aligned} \frac{2L}{c} &= \sqrt{1 - \frac{\mathbf{v}^2}{c^2}}\Delta t_e \\ \Delta t &= \sqrt{1 - \frac{\mathbf{v}^2}{c^2}}\Delta t_e \end{aligned}}$$

$$\Delta t_e = \frac{1}{\sqrt{1 - \frac{\mathbf{v}^2}{c^2}}}\Delta t$$

Time and Animation

- Animation
  - change over time
- Computer Animation
  - computing the physical state in accordance with "time"
- We must measure the time
  - Stop watch is needed
  - Let's make our own stop watch for animation



Stop Watch (header)

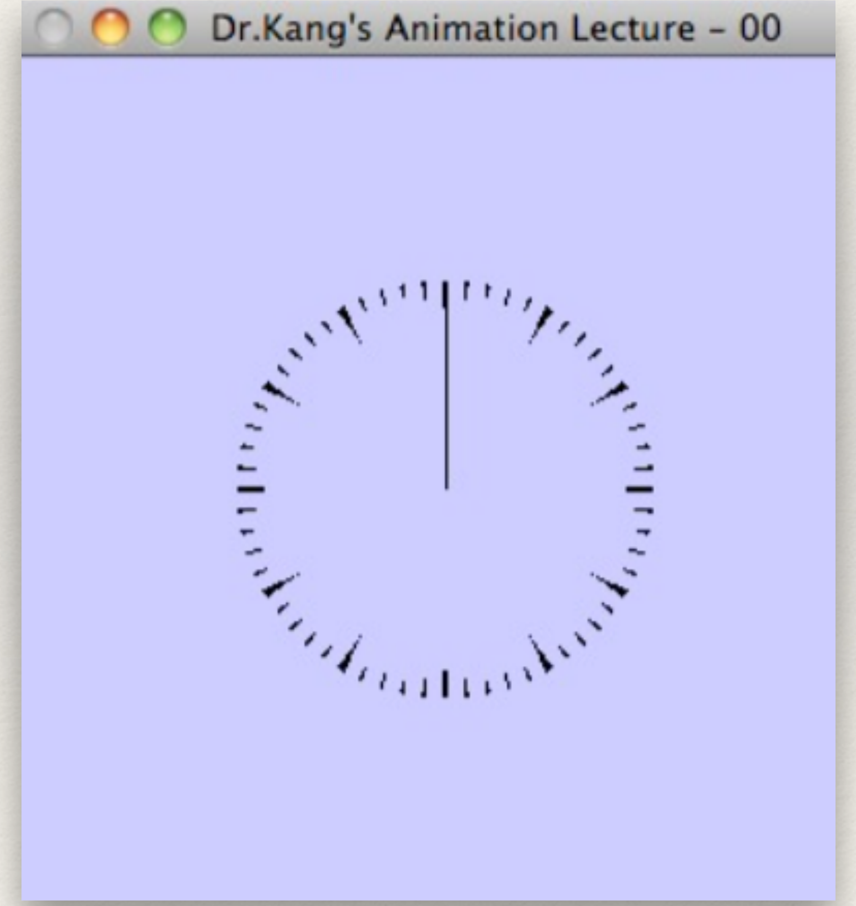
```
#ifndef _STOPWATCH_YMKANG_H
#define _STOPWATCH_YMKANG_H

#ifdef WIN32   // Windows system specific
#include <windows.h>
#else          // Unix based system specific
#include <sys/time.h>
#endif

class StopWatch {
#ifdef WIN32
    LARGE_INTEGER frequency;                         // ticks per second
    LARGE_INTEGER startCount;                        //
    LARGE_INTEGER endCount;                          //
#else
    timeval startCount;                              //
    timeval endCount;                                //
#endif
    double startTimeInMicroSec;
    double endTimeInMicroSec;
public:
    StopWatch();
    void start();               // start StopWatch and record time to "startCount"
    void stop();               // stop StopWatch and record time to "endCount"
    double getElapsedTime();    // return the elapsed time at the last stop since the last start (microsec)
};

#endif
```

Stop Watch (implementation)

```
/*
 *  StopWatch.cpp
 *  Young-Min Kang
 *  Tongmyong University
 *
 */

#include "StopWatch.h"

StopWatch::StopWatch() {
#ifdef WIN32
    QueryPerformanceFrequency(&frequency);
    startCount.QuadPart = 0;
    endCount.QuadPart = 0;
#else
    startCount.tv_sec = startCount.tv_usec = 0;
    endCount.tv_sec = endCount.tv_usec = 0;
#endif
    startTimeInMicroSec = endTimeInMicroSec = 0.0;
}

void StopWatch::start() {
#ifdef WIN32
    QueryPerformanceCounter(&startCount);
#else
    gettimeofday(&startCount, NULL);
#endif
}

void StopWatch::stop() {
#ifdef WIN32
    QueryPerformanceCounter(&endCount);
#else
    gettimeofday(&endCount, NULL);
#endif
}

double StopWatch::getElapsedTime(){
#ifdef WIN32
    startTimeInMicroSec = startCount.QuadPart * (1000000.0 / frequency.QuadPart);
    endTimeInMicroSec = endCount.QuadPart * (1000000.0 / frequency.QuadPart);
#else
    startTimeInMicroSec = (startCount.tv_sec * 1000000.0) + startCount.tv_usec;
    endTimeInMicroSec = (endCount.tv_sec * 1000000.0) + endCount.tv_usec;
#endif
    return endTimeInMicroSec - startTimeInMicroSec;
}
```

Visualise Your Stop Watch

- Implement your stop watch



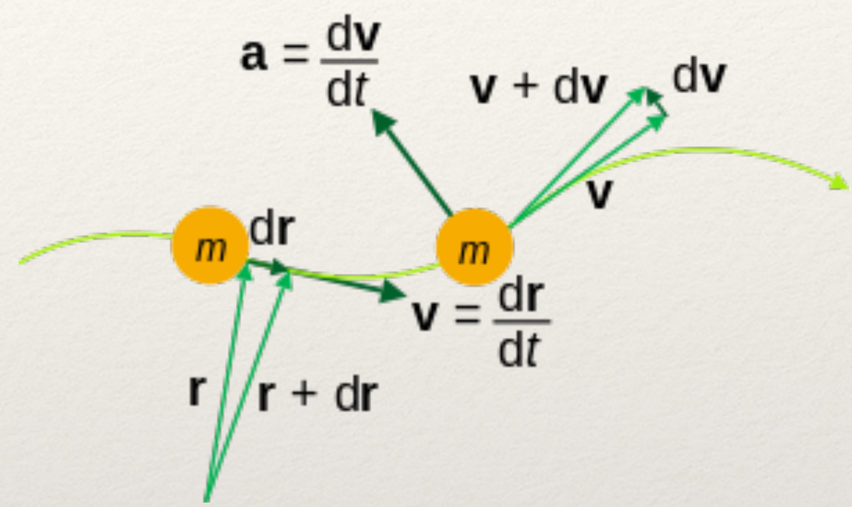
Physically-based Modelling

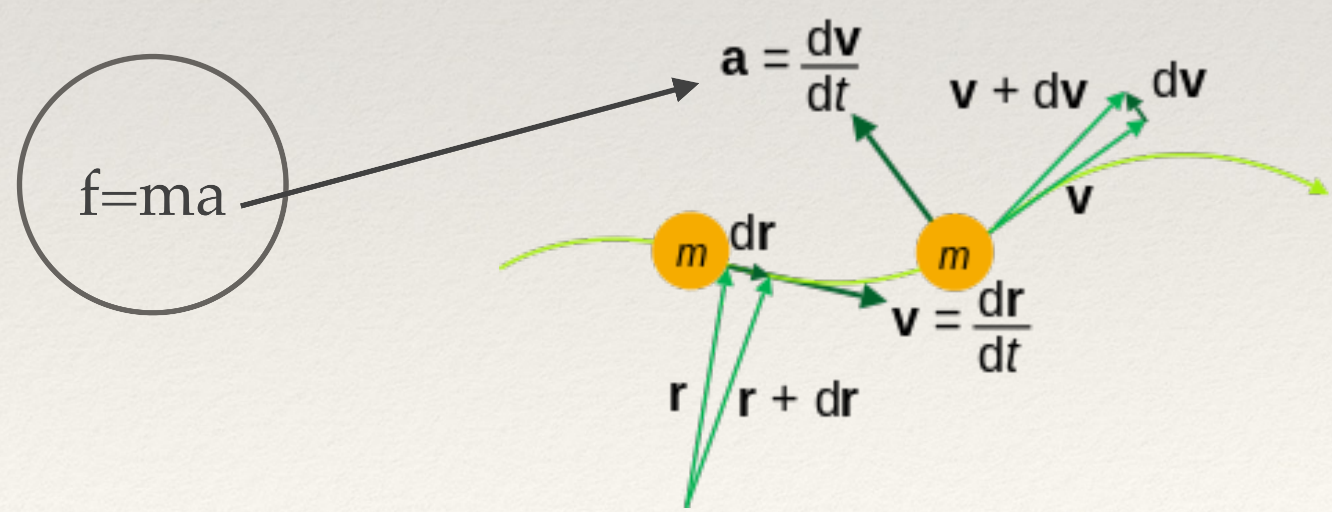
Kiematics

Young-Min Kang
Tongmyong University

Kinematics

- Kinematics vs. Dynamics
    - kinematics
        - force is not considered
        - location, velocity, and acceleration are considered as
            - functions of time
            - x(t), v(t), a(t)
    - dynamics
        - force plays the most essential role
            - compute f(t)
            - a(t) = f(t)/m
            - v(t+dt) += a(t)dt
            - x(t+dt) += v(t+dt)dt





Statics, Kinetics, Kinematics and Dynamics

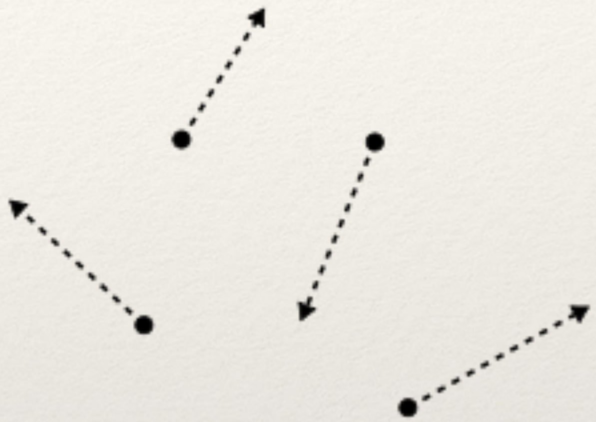
- Statics
  - the study of equilibrium and its relation to **forces**
- Kinetics
  - the study of motion and its relation to **forces**
- Kinematics
  - the study of observed motions without regard for circumstances causing them

classical mechanics

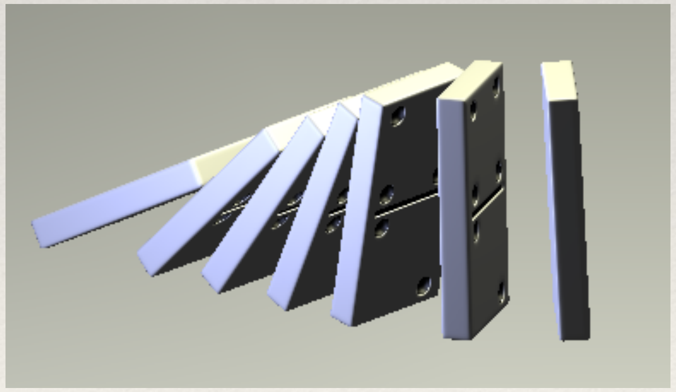
- Dynamics
  - Statics + Kinetics

Particle and Rigid Body

- Particle
  - very very tiny object with mass
    - volume can be ignored
  - rocket trajectory analysis
    - the volume of the rocket can be ignored
    - rocket can be regarded a particle



- Rigid body (= idealised solid body)
  - object with mass and volume
  - the shape does not change in any condition
  - valid animation of rigid body = rotation and translation



Velocity

- velocity
  - a vector
    - speed: the magnitude of velocity
    - velocity = (speed + direction)
  - direction of velocity
    - moving direction
  - magnitude of velocity
    - ratio of moving distance to time

$$\mathbf{v} = \frac{\Delta \mathbf{s}}{\Delta t}$$

ratio of displacement to time interval

Instantaneous Velocity

- Reduce the time interval
  - the more precise velocity at the moment
  - if the time interval approaches to 0
    - instantaneous velocity

$$\mathbf{v} = \lim_{\Delta t \to 0} \frac{\Delta \mathbf{s}}{\Delta t} = \frac{d\mathbf{s}}{dt}$$

displacement

- integration of velocity

$$\int \mathbf{v} dt = \int d\mathbf{s}$$

  - integration from t1 to t2

$$\int_{t_1}^{t_2} \mathbf{v} dt = \int_{s(t_1)}^{s(t_2)} d\mathbf{s}$$

  - equivalent to the displacement

$$\int_{t_1}^{t_2} \mathbf{v} dt = \mathbf{s}(t_2) - \mathbf{s}(t_1) = \Delta\mathbf{s}$$

- If we know the velocity
  - we can predict where the particle will be located in the future

$$\mathbf{v} = \lim_{\Delta t \to 0} \frac{\Delta \mathbf{s}}{\Delta t} = \frac{d\mathbf{s}}{dt}$$

$$\mathbf{v} dt = d\mathbf{s}$$

Acceleration

- Average acceleration
  - ratio of velocity change to the time interval

$$\mathbf{a} = \Delta\mathbf{v}/\Delta t$$

- Instantaneous acceleration

$$\mathbf{a} = \lim_{\Delta t \to 0} \Delta\mathbf{v}/\Delta t = d\mathbf{v}/dt$$

- Integration of acceleration $\mathbf{a}dt = d\mathbf{v}$
  - velocity change

$$\int_{t_1}^{t_2} \mathbf{a}dt = \int_{\mathbf{v}(t_1)}^{\mathbf{v}(t_2)} d\mathbf{v} = \Delta\mathbf{v}$$

Constant Acceleration

- Simple problem for Kinematics
  - example of constant acceleration: gravity
  - gravitational acceleration
    - magnitude: 9.81 m/s^2
    - direction: downwards (0,-1,0)
- We can easily "integrate the acceleration" to compute
  - velocity change at every time
  - this makes it possible to know the velocity at every time: v(t)

Gravitational Acceleration

- gravitational acceleration: g
  - velocity change and the integration of acceleration

$$\int_{\mathbf{v}(t_1)}^{\mathbf{v}(t_2)} d\mathbf{v} = \int_{t_1}^{t_2} \mathbf{g} dt$$

$$\mathbf{v}_2 - \mathbf{v}_1 = \mathbf{g}(t_2 - t_1)$$

  - If we know every thing at time t_1
    - We can easily predict the velocity at time t_2

$$\mathbf{v}_2 = \mathbf{g}t_2 - \mathbf{g}t_1 + \mathbf{v}_1$$

    - if $t_1=0$ ($t_2=t$)

$$\mathbf{v}_2 = \mathbf{v}_1 + \mathbf{g}t$$

Velocity as f(Displacement)

- Another differential equation

$$\mathbf{v}\frac{d\mathbf{v}}{dt} = \frac{d\mathbf{s}}{dt}\mathbf{a}$$

$$\mathbf{v}d\mathbf{v} = \mathbf{a}d\mathbf{s}$$

- Integration $\int_{\mathbf{v}_1}^{\mathbf{v}_2} \mathbf{v}d\mathbf{v} = \int_{\mathbf{s}_1}^{\mathbf{s}_2} \mathbf{a}d\mathbf{s}$

$$\frac{1}{2}\mathbf{v}^2\Big|_{\mathbf{v}_1}^{\mathbf{v}_2} = \mathbf{a}\mathbf{s}\Big|_{\mathbf{s}_1}^{\mathbf{s}_2} = \mathbf{g}\mathbf{s}\Big|_{\mathbf{s}_1}^{\mathbf{s}_2}$$

- Relation between velocity and displacement

$$\frac{1}{2}({\mathbf{v}_2}^2 - {\mathbf{v}_1}^2) = \mathbf{g}(\mathbf{s}_2 - \mathbf{s}_1)$$

$${\mathbf{v}_2}^2 = 2\mathbf{g}(\mathbf{s}_2 - \mathbf{s}_1) + {\mathbf{v}_1}^2$$

Location

- Velocity and displacement

$$\mathbf{v}dt = d\mathbf{s}$$
$$(\mathbf{v}_1 + \mathbf{g}t)dt = d\mathbf{s}$$

- Integration

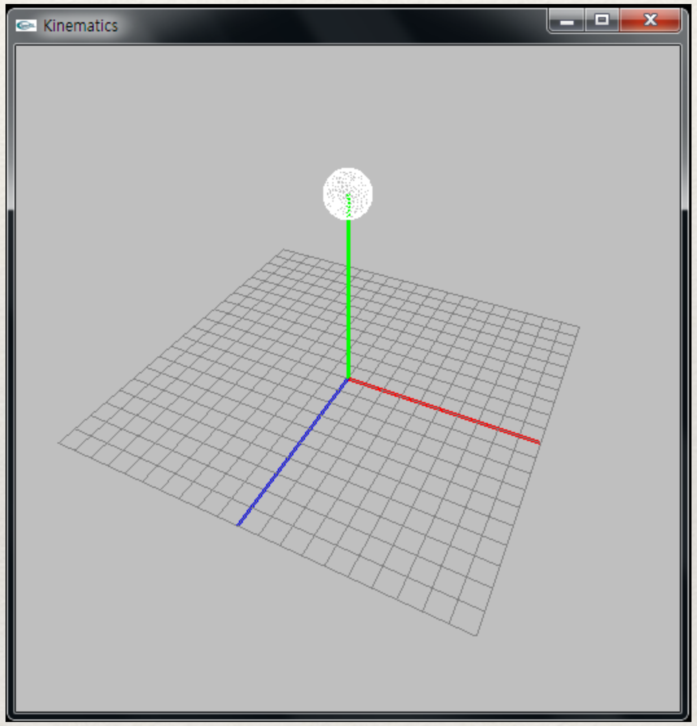
$$\int_0^t (\mathbf{v}_1 + \mathbf{g}t)dt = \int_{\mathbf{s}_1}^{\mathbf{s}_2} d\mathbf{s}$$
$$v_1 t + \frac{1}{2}\mathbf{g}t^2 = \mathbf{s}_2 - \mathbf{s}_1$$

- Location at time t

$$\mathbf{s}_2 = v_1 t + \frac{1}{2}\mathbf{g}t^2 + \mathbf{s}_1$$

Kinematic simulation

```
#include "KinematicsSimulator.h"

CKinematicSimulator::CKinematicSimulator() : CSimulator() {}


void CKinematicSimulator::init() {
    initialLoc.set(0,1,0);
    initialVel.set(0,3,0);
    gravity.set(0.0, -9.8, 0.0);
    currentLoc = initialLoc;
    particle.setPosition(currentLoc[0], currentLoc[1], currentLoc[2]);
    particle.setRadius(0.1);
}

void CKinematicSimulator::doBeforeSimulation(double dt, double currentTime) {

}

void CKinematicSimulator::doSimulation(double dt, double currentTime) {
    currentLoc = initialLoc
        + currentTime*initialVel
        + (0.5 * currentTime * currentTime)  * gravity ;
    particle.setPosition(currentLoc[0], currentLoc[1], currentLoc[2]);
    particle.drawWithGL();
}


void CKinematicSimulator::doAfterSimulation(double dt, double currentTime) {

}
```



https://github.com/dknife/201501Lec_GamePhysics/tree/master/GPcode01_03_KinematicParticleExplosion

Physically-based Modelling

Forces

Young-Min Kang
Tongmyong University

Force

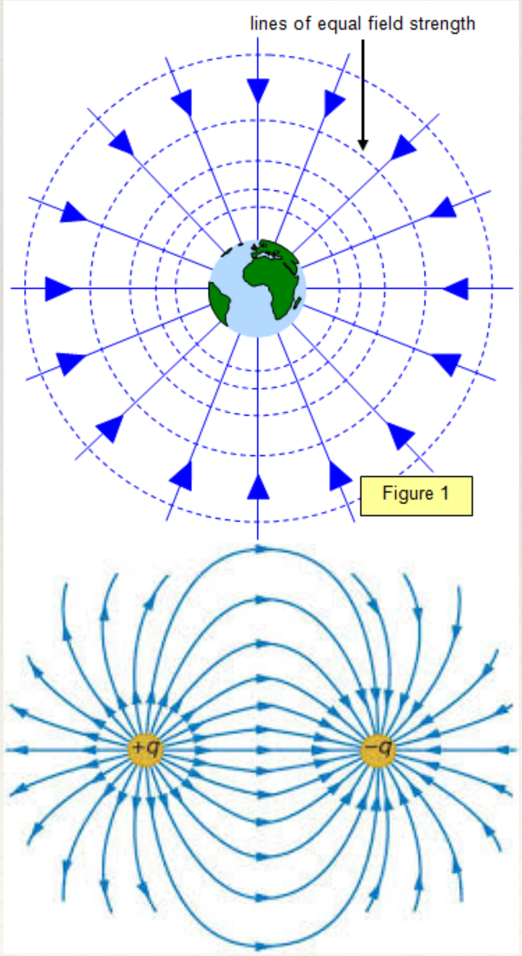
- Limitation of kinematics
  - difficult to handle “various changing forces”
- Dynamics
  - force-based understanding of motions
- What is force?
  - something that causes motion

Concepts

- Force field : example - gravitational force
- Friction: a contact force that resists motion
- Fluid dynamic drag: resistance to objects moving in fluid
- Pressure: force per area
- Buoyancy: force pushing objects “upwards” when immersed in a fluid
- Springs and dampers: elastically ties objects
- Force applies torque to an object, and make it “rotate.”

Force field

- Force field
  - a vector field indicating the force exerted by on object on another
- Very good example
  - gravitational force field
  - electromagnetic force field



Gravitational force field

- universal gravitational force $$|\mathbf{f}_u| = Gm_1m_2/r^2$$
  - G: gravitation constance $6.673 \times 10^{-11}(N \cdot m^2)/kg^2$
  - r: distance between masses
  - m_{1,2}: mass of each objects
- gravity on Earth
  - mass of Earth: $5.98 \times 10^{24}kg$
  - radius of Earth: $6.38 \times 10^6 m$
  - gravitational acceleration

$$\frac{Gm_{earth}}{r^2} \simeq \left(\frac{6.673 \times 5.98}{6.38^2}\right) \times 10m/s^2 \simeq 9.8034m/s^2$$

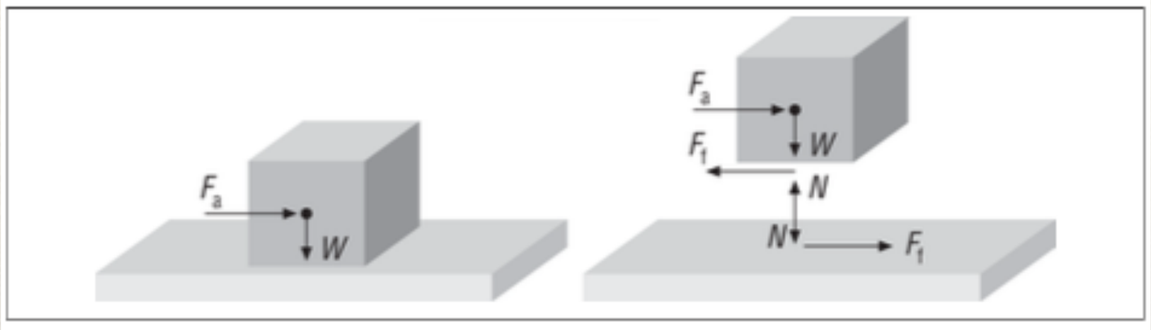
friction

- resistance force due to the contacting surfaces
  - contact force
  - normal force: N is important
- two friction
  - static friction: maximum friction

$$|\mathbf{f}_{max}| = \mu_s \mathbf{N}$$

  - kinetic friction

$$|\mathbf{f}_k| = \mu_k \mathbf{N}$$



Coefficients of friction

- Well known surfaces

Surface condition	M_s	M_u	% difference
Dry glass on glass	0.94	0.4	54%
Dry iron on iron	1.1	0.15	86%
Dry rubber on pavement	0.55	0.4	27%
Dry steel on steel	0.78	0.42	46%
Dry Teflon on Teflon	0.04	0.04	—
Dry wood on wood	0.38	0.2	47%
Ice on ice	0.1	0.03	70%
Oiled steel on steel	0.10	0.08	20%

Fluid dynamic drag

- similar to friction
  - friction is also important component of drag
  - but, it's not the only one
- viscous drag for "slow-moving" objects (laminar)
  - $f = -C v$
- for "fast-moving" objects (turbulence)
  - $f = -C v^2$

Pressure

- Pressure is not force
  - pressure = force per unit area
  - $F = PA$ (force = pressure x area)
  - $P = F / A$
- Pressure is important in simulating
  - boats, hovercrafts,…

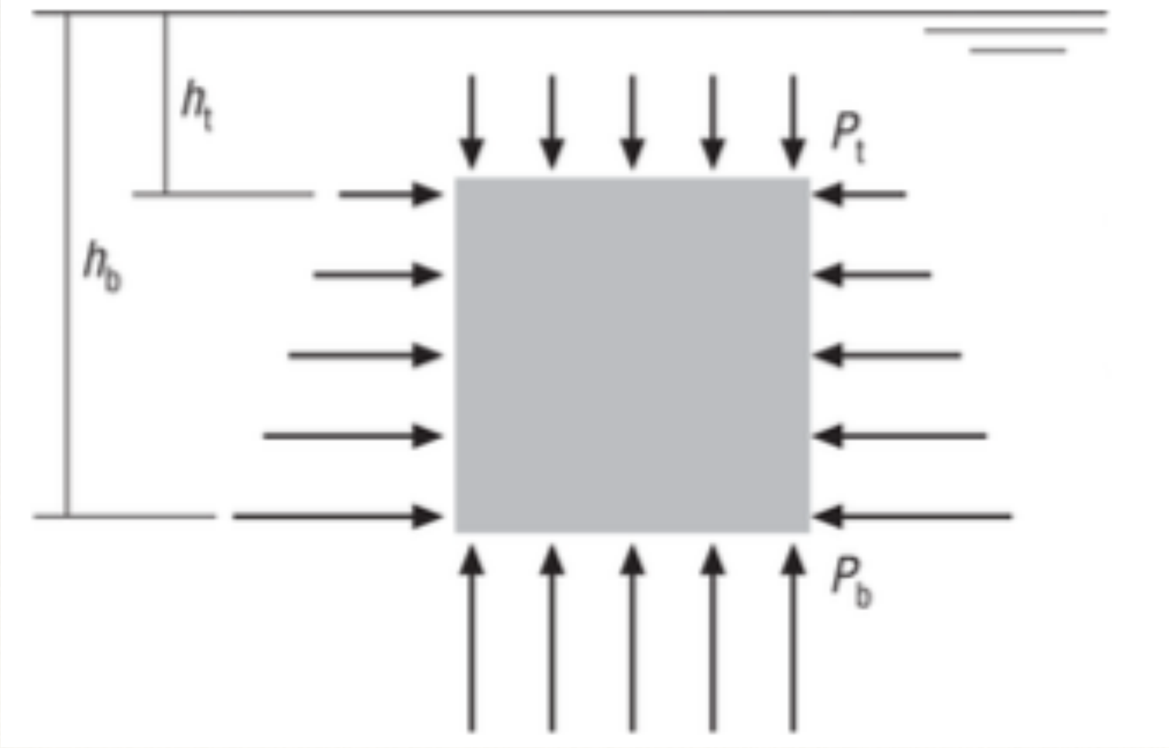
Buoyancy

- Different pressure in fluid
- Horizontal net force = 0
- Vertical net force = bottom force - top force
- F = PA
- Pressures: function(density, gravity)
  - on top surface

$$\mathbf{P}_t = \rho \mathbf{g} h_t$$

  - on bottom surface

$$\mathbf{P}_b = \rho \mathbf{g} h_b$$



Buoyancy

- forces $$\mathbf{f}_t = \mathbf{P}_t A_t = \rho \mathbf{g} h_t s^2$$
  $$\mathbf{f}_b = \mathbf{P}_b A_b = \rho \mathbf{g} h_b s^2$$

- difference

$$\begin{aligned}
\mathbf{f}_b - \mathbf{f}_t &= \rho \mathbf{g} h_b s^2 - \rho \mathbf{g} h_t s^2 \\
&= \rho \mathbf{g} (h_b - h_t) s^2 \\
&= -\rho \mathbf{g} s^3 \\
&= -\rho \mathbf{g} V \quad (V : volume)
\end{aligned}$$

Spring force

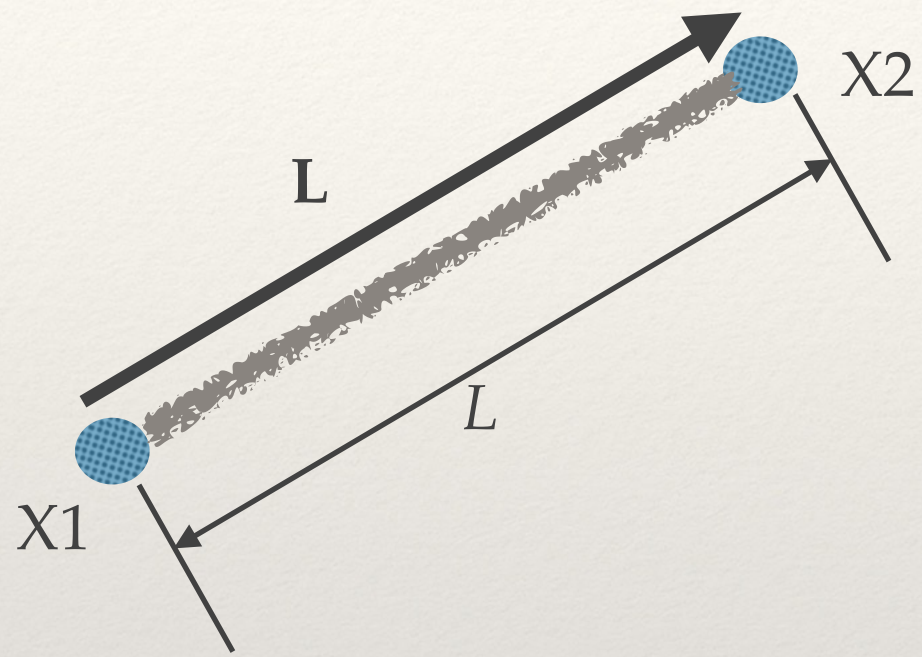
- Hooke's law
  - force needed to extend or compress a spring by a distance x is proportional to that distance
    - f = -k x
    - k: spring constant
- rest length of spring:
- current length of spring:
  - force magnitude: $|\mathbf{f}| = k_s(L - r)$
  - force direction: obj1 and obj2 are linked and located at x1 and x2.
    - $\dfrac{\mathbf{x}_1 - \mathbf{x}_2}{|\mathbf{x}_1 - \mathbf{x}_2|}$ and $-\dfrac{\mathbf{x}_1 - \mathbf{x}_2}{|\mathbf{x}_1 - \mathbf{x}_2|}$

Damper

- Springs do not oscillate forever
  - energy dissipation
  - simple model
    - damping force

$$\mathbf{f}_d = k_d(\mathbf{v}_1 - \mathbf{v}_2)$$

Spring and Damper

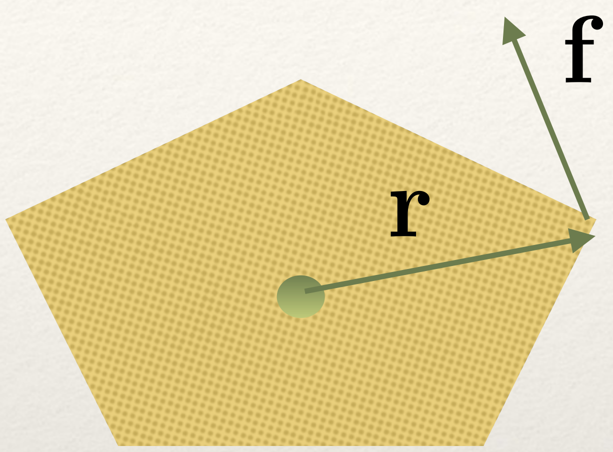


$$\mathbf{f}_1 = -(k_s(L - r) + k_d(\mathbf{v}_1 - \mathbf{v}_2) \cdot \frac{\mathbf{L}}{L})\frac{\mathbf{L}}{L}$$

$$\mathbf{f}_2 = -\mathbf{f}_1$$

Force and Torque

- Force
  - causes linear acceleration
- Torque
  - causes angular acceleration
- Torque: τ
  - a vector
    - magnitude
      - how quickly the angular velocity is changed
        - | r x f |
    - direction
      - rotational axis = (r x f) / | r x f |



$$\tau = \mathbf{r} \times \mathbf{f}$$