ME 327: Design and Control of Haptic Systems
Spring 2020

Lecture 13:
Kinesthetic haptic devices:
1-DoF rendering

Allison M. Okamura
Stanford University

rendering friction

(in one degree of freedom)

surface properties

- typical haptic VEs display general shape very well, but don't feel "realistic"
  - add surface properties to increase realism
- how you do it depends on
  - the surface model
  - complexity of the surface property
  - frequency response of your haptic device

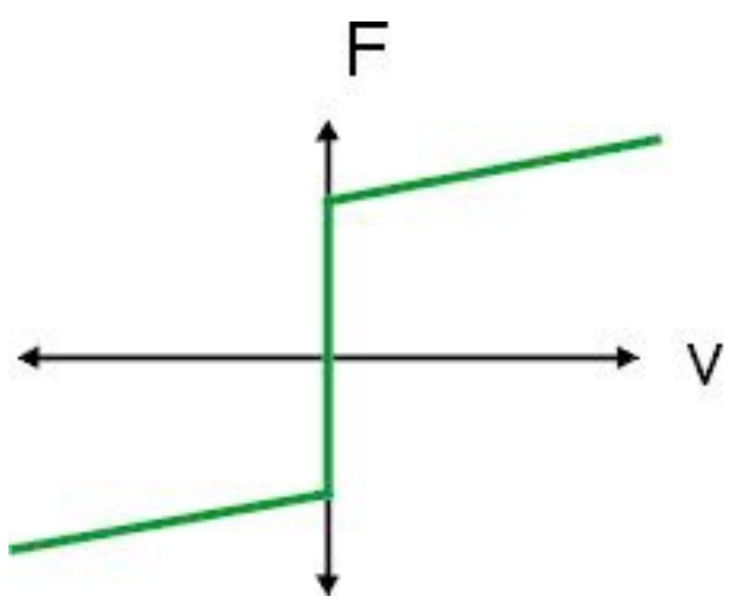
damping for virtual walls

- a pure spring force for a wall may seem to “active”
- add a dissipative term, where b is the damping coefficient
- only damps when going into the wall
- this can also create vibrations upon wall impact

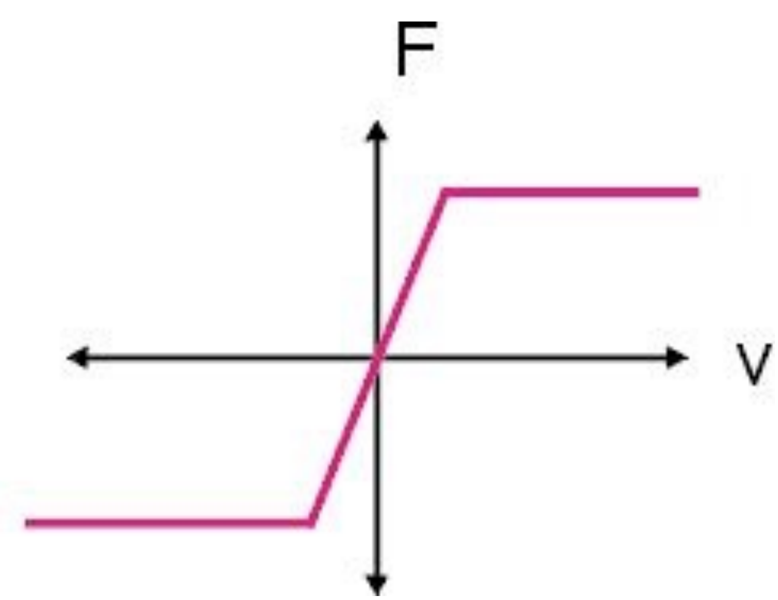
$$F = \begin{cases} k\Delta x + b\dot{x} & \text{for } \dot{x} > 0 \\ k\Delta x & \text{for } \dot{x} < 0 \end{cases}$$

"frictional" damping

- surfaces can feel unnaturally slippery
- friction would help, but it is difficult to implement
- you can add damping to motion parallel to surface



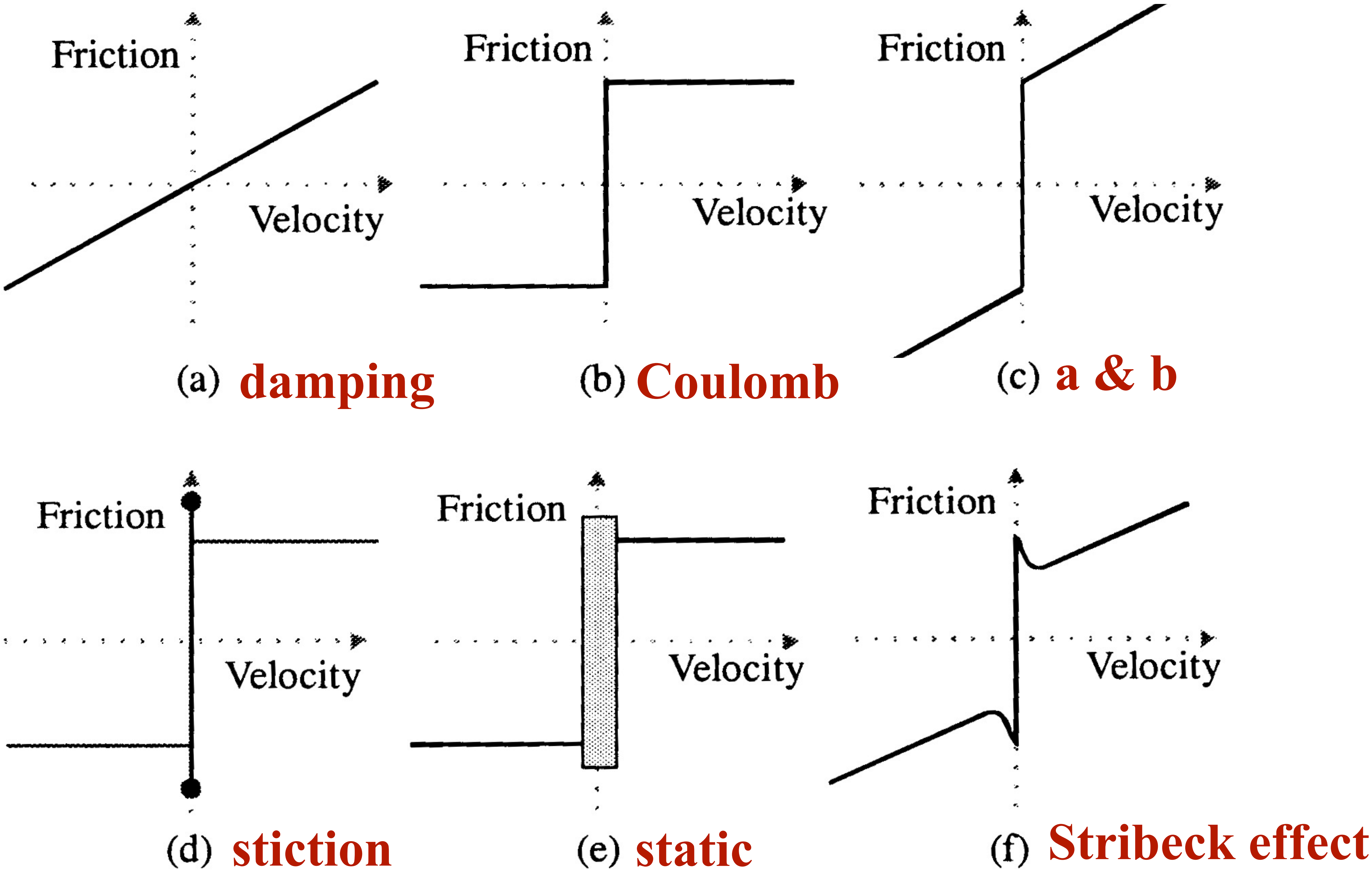
friction



damping

friction display

difficult to render because it is non-linear

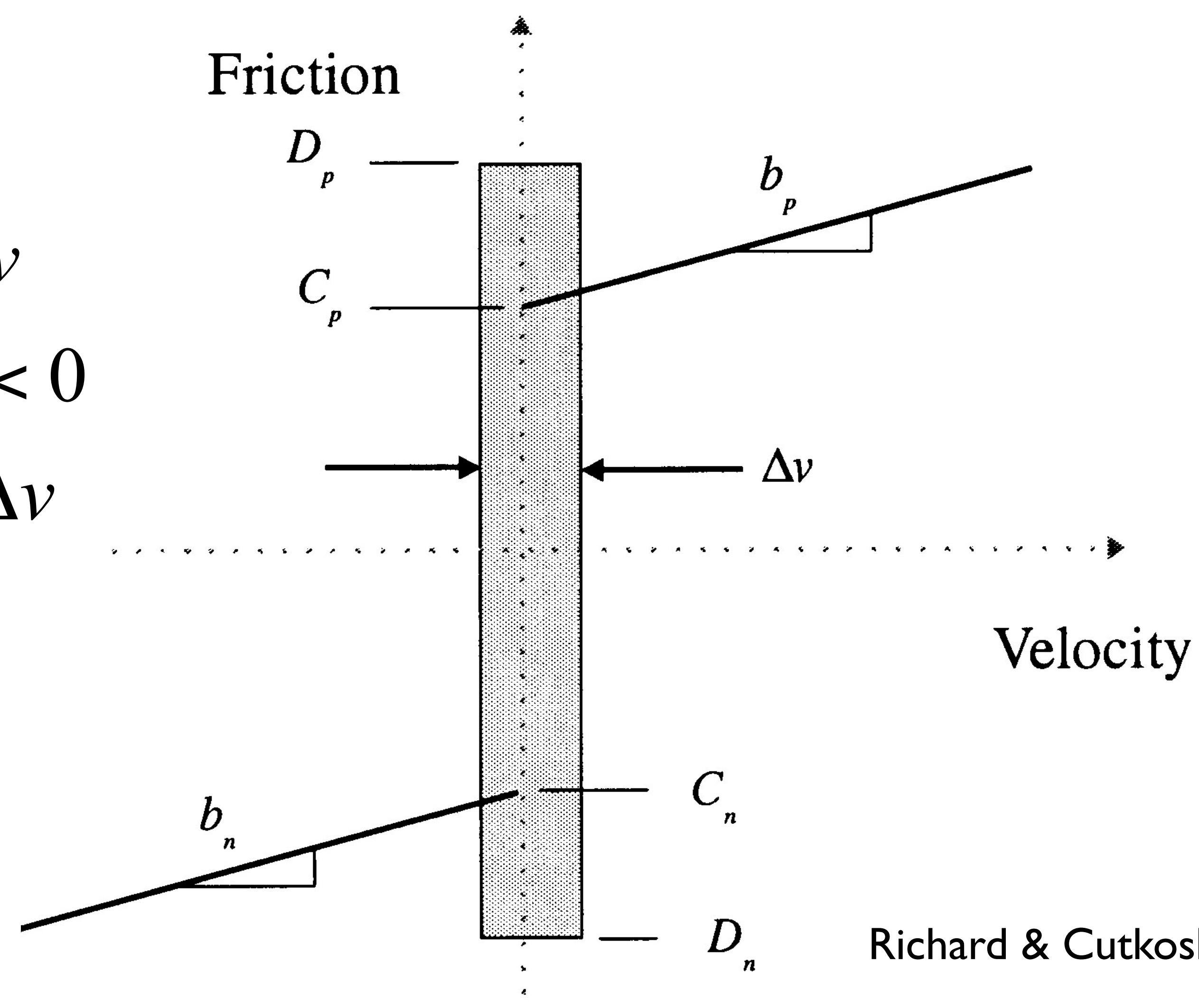


Richard & Cutkosky

Karnopp model

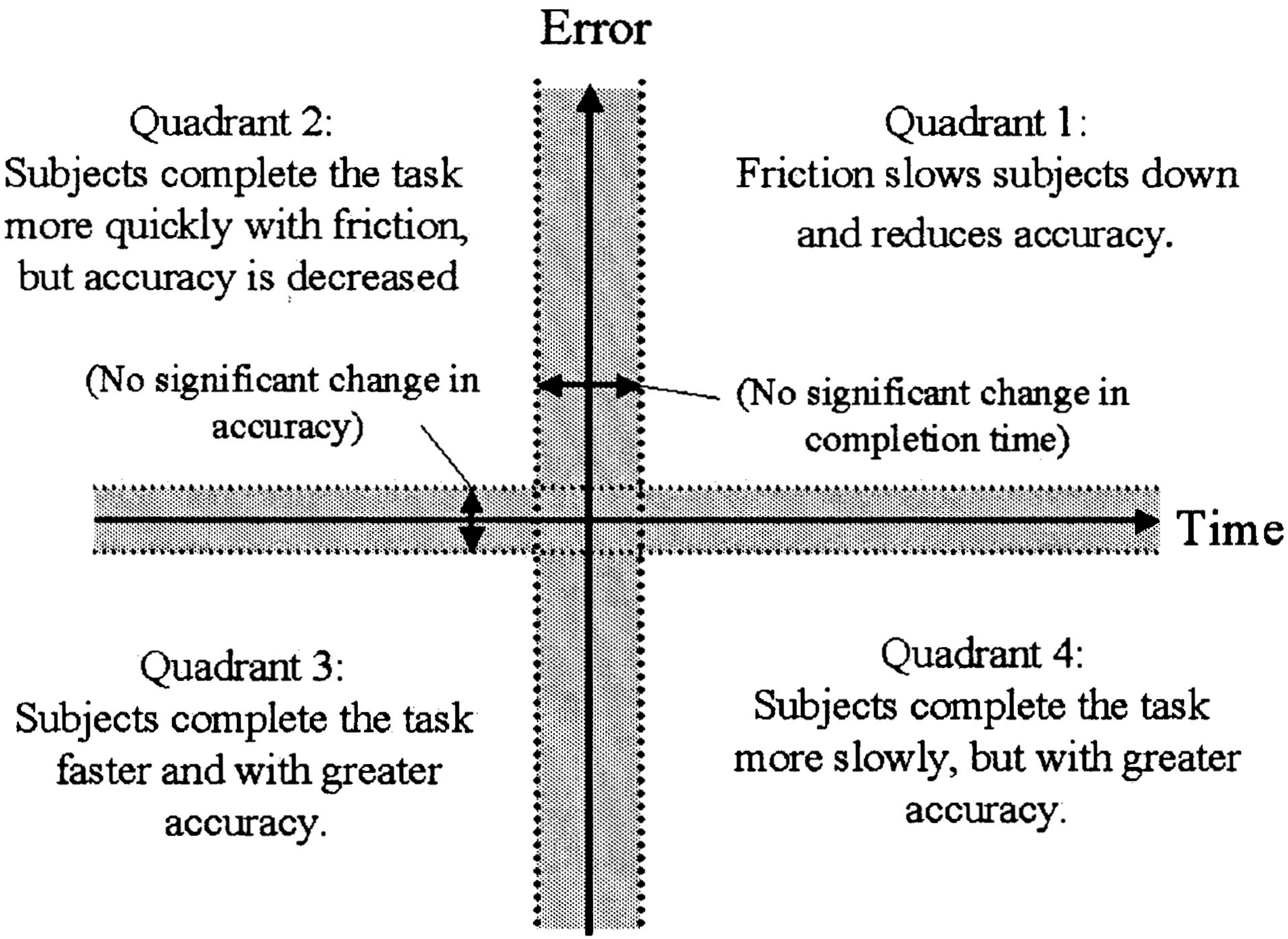
adds viscous damping, Coulomb friction, and static friction

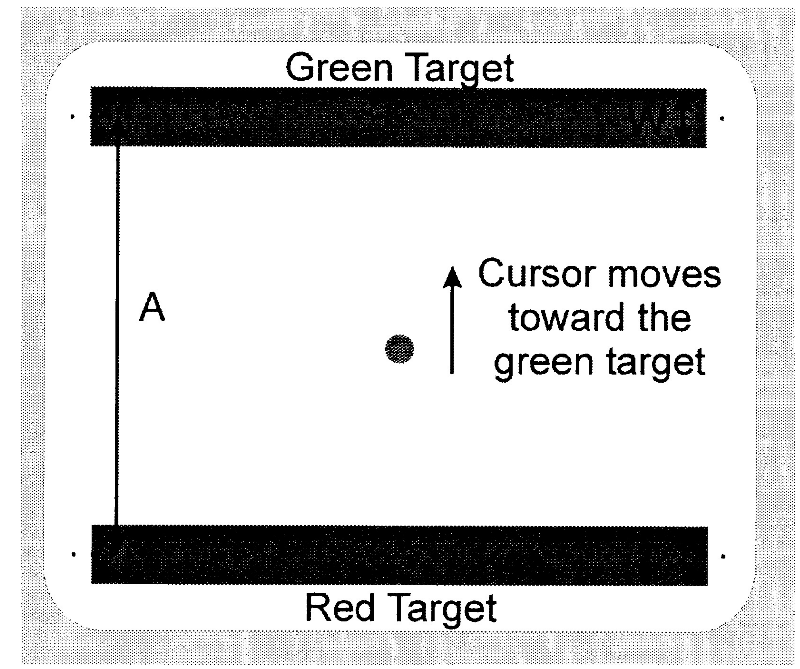
$$F = \begin{cases} C_n \operatorname{sgn}(\dot{x}) + b_n \dot{x} & \text{for } \dot{x} < -\Delta v \\ \max(D_n, F_a) & \text{for } -\Delta v < \dot{x} < 0 \\ \min(D_p, F_a) & \text{for } 0 < \dot{x} < \Delta v \\ C_p \operatorname{sgn}(\dot{x}) + b_p \dot{x} & \text{for } \dot{x} > \Delta v \end{cases}$$



Richard & Cutkosky

aside: friction display evaluation





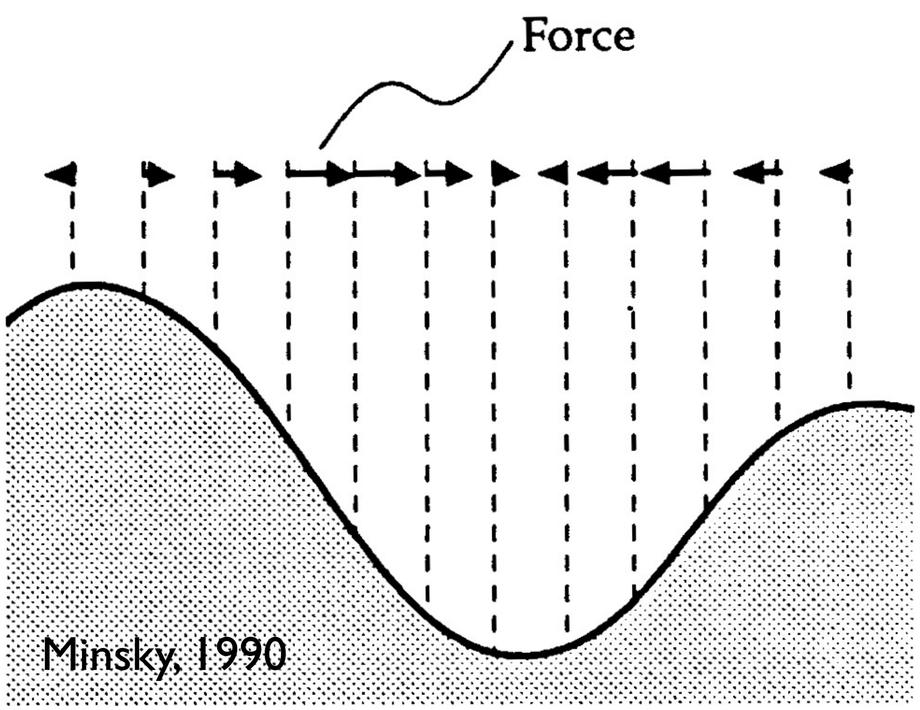
Richard & Cutkosky, 2000

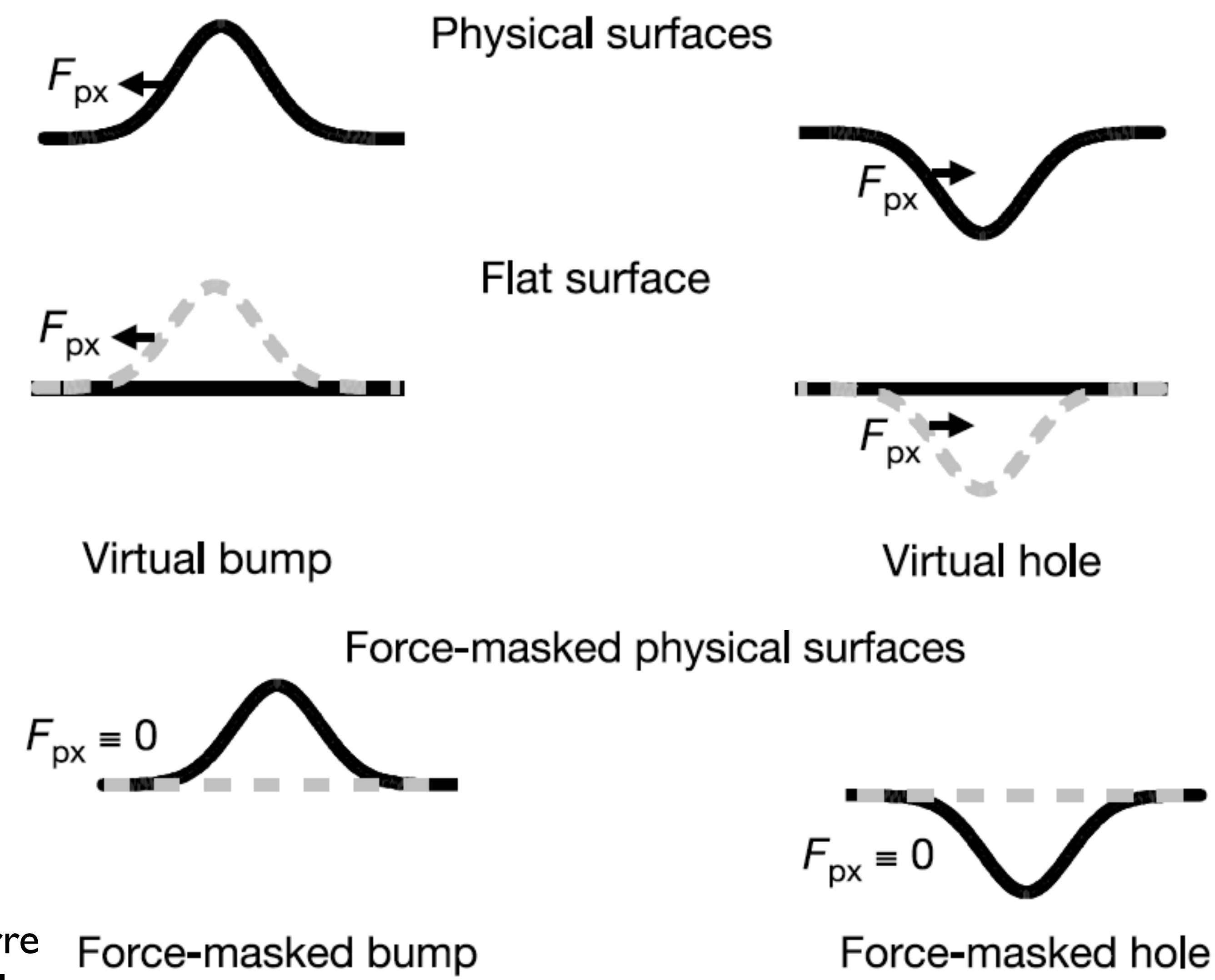
rendering bumps and textures

(in one degree of freedom)

bumps and valleys

- as a user moves "up" a bump, motion is opposed.
- done in 2D, spring force proportional to height of bump
- force information can overcome geometry!





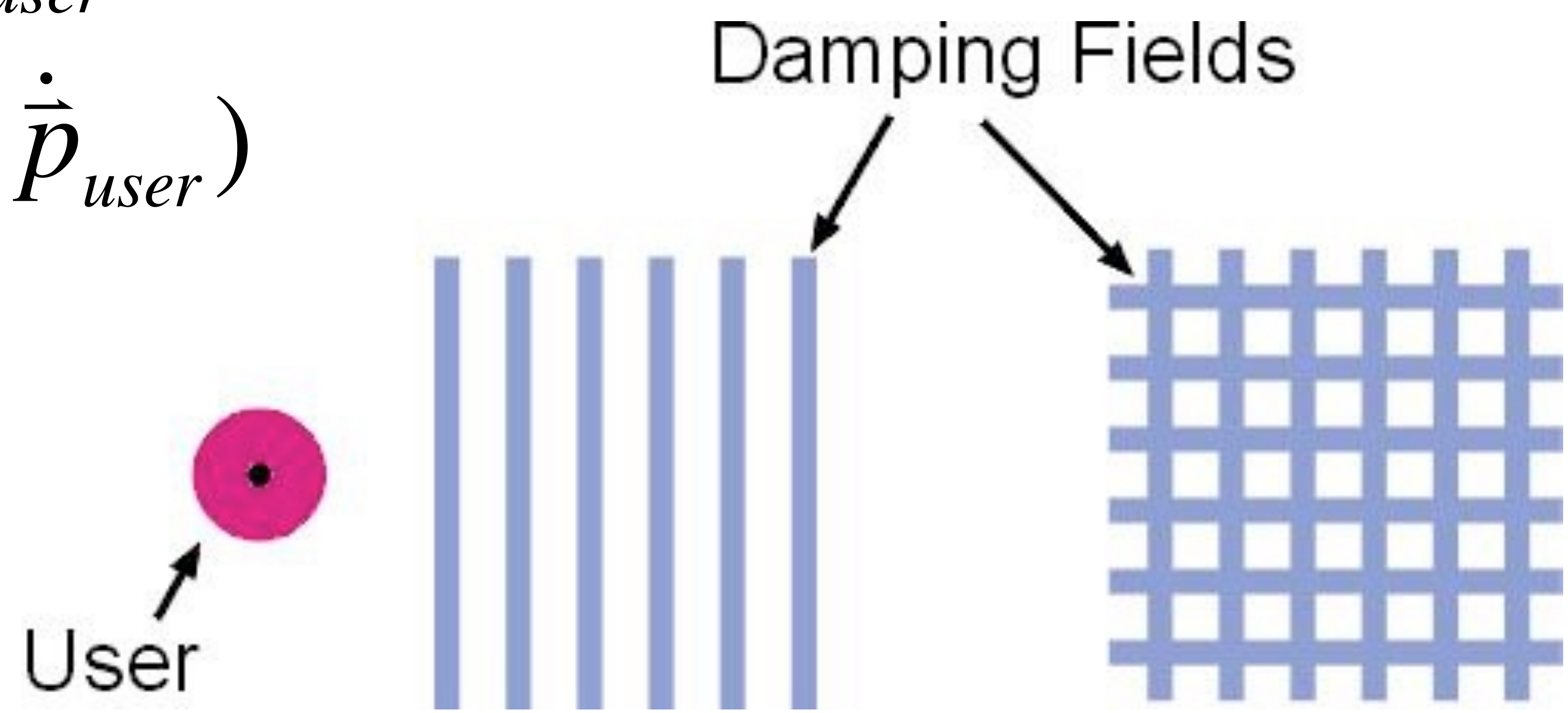
Robles-De-La-Torre & Hayward, 2001

damping textures

if $\vec{p}_{user}$ is inside a damping area

$$F = b\vec{v}_{user}$$

$$(\vec{v}_{user} = \dot{\vec{p}}_{user})$$



note that vibrations occur due to discontinuity in force

simulating and rendering dynamic objects

(in one degree of freedom)

dynamic simulation of "rigid" bodies

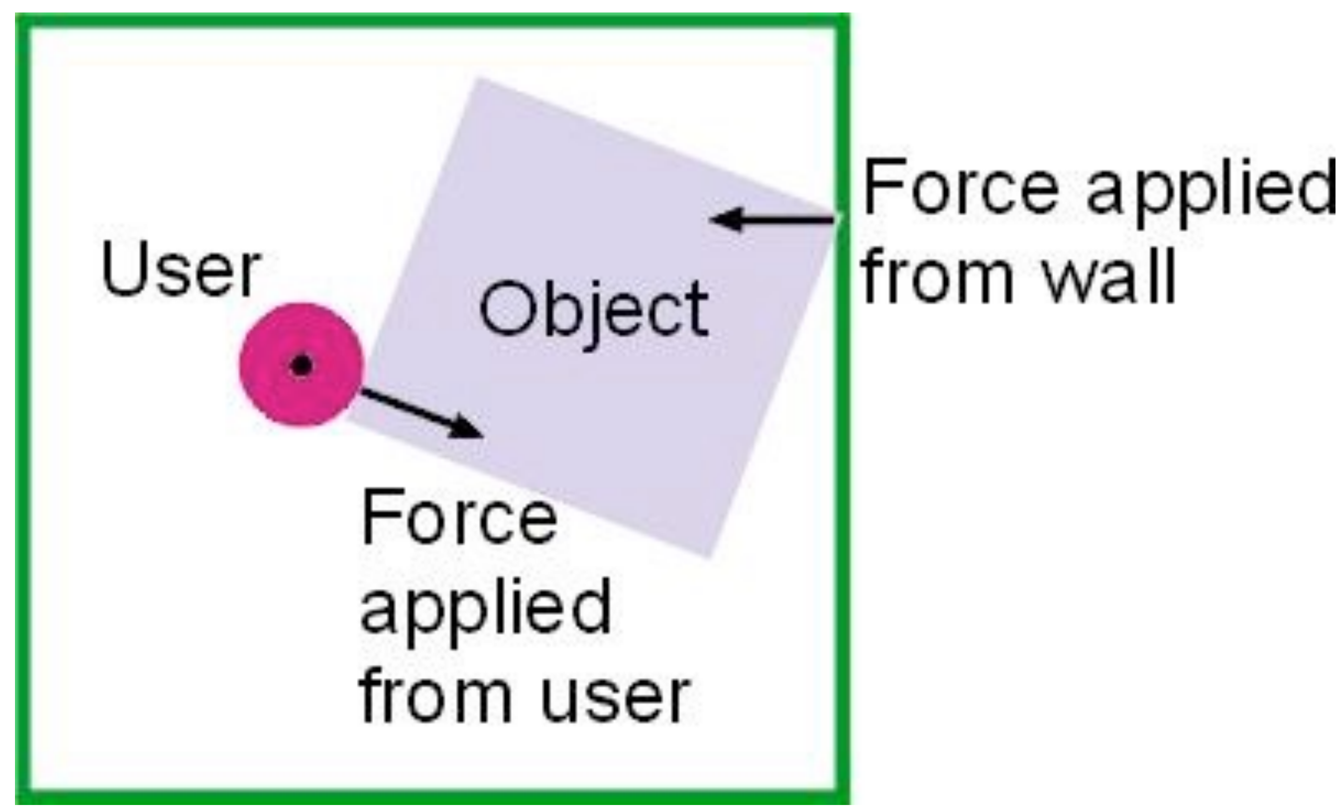
- assumptions
  - you have an impedance-type force-feedback display
  - you are using a linear stiffness model of the surface
- basic approach:
  1. save the state of the moving object
  2. sum the forces on the object
  3. calculate the new state

object state

- the "state" of an object is used to describe its current condition
- made up of variables that will change with time, such as
  - position
  - velocity
  - acceleration
  - rarely: other parameters such as mass, shape, etc. that might be changed by dynamic interaction

calculating forces on an object

- for forces from the user's hand pushing, this is equal and opposite to the force fed back to the user
- for forces from other objects in the VE, use the same idea: force is proportional to penetration



pseudocode

- store last_ObjAcc, last_ObjVel
- ObjForce = Force_from_user
            + Force_from_other_objects

  Force_from_user = k*penetration_distance_user

  Force_from_other_objects = k*penetration_distances_objects

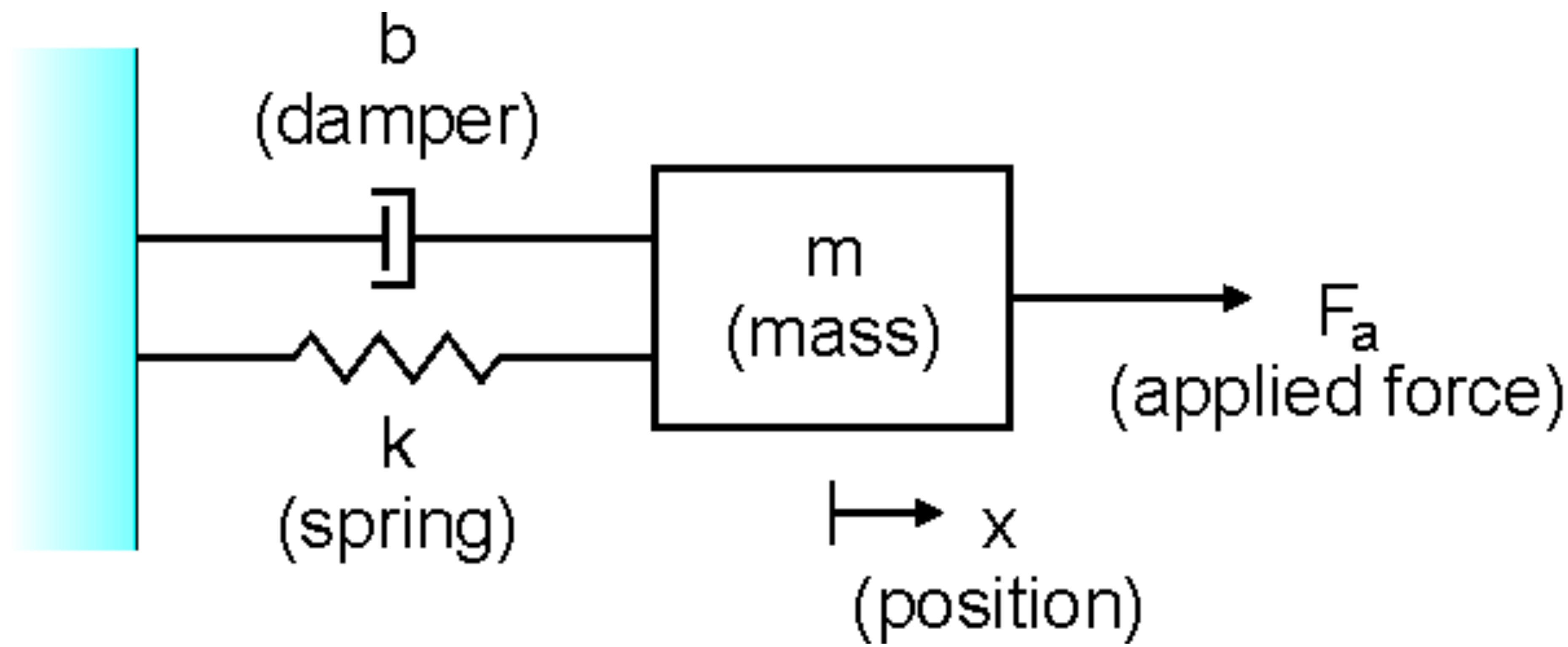
  can add other effects (e.g. damping) as well
- ObjAcc = ObjForce/mass *(F=ma -> a = F/m)*
- integrate using (for example) the trapeziodal rule:
  ObjVel += 1/2(last_ObjAcc + ObjAcc)/sim_freq
  ObjPos += 1/2 (last_ObjVel + ObjVel)/sim_freq

the new state

- now you have a new position, velocity, and acceleration of the object
- use new position for collision detection, penetration and force calculation
- store velocity and acceleration for the next integration
- do the loop again

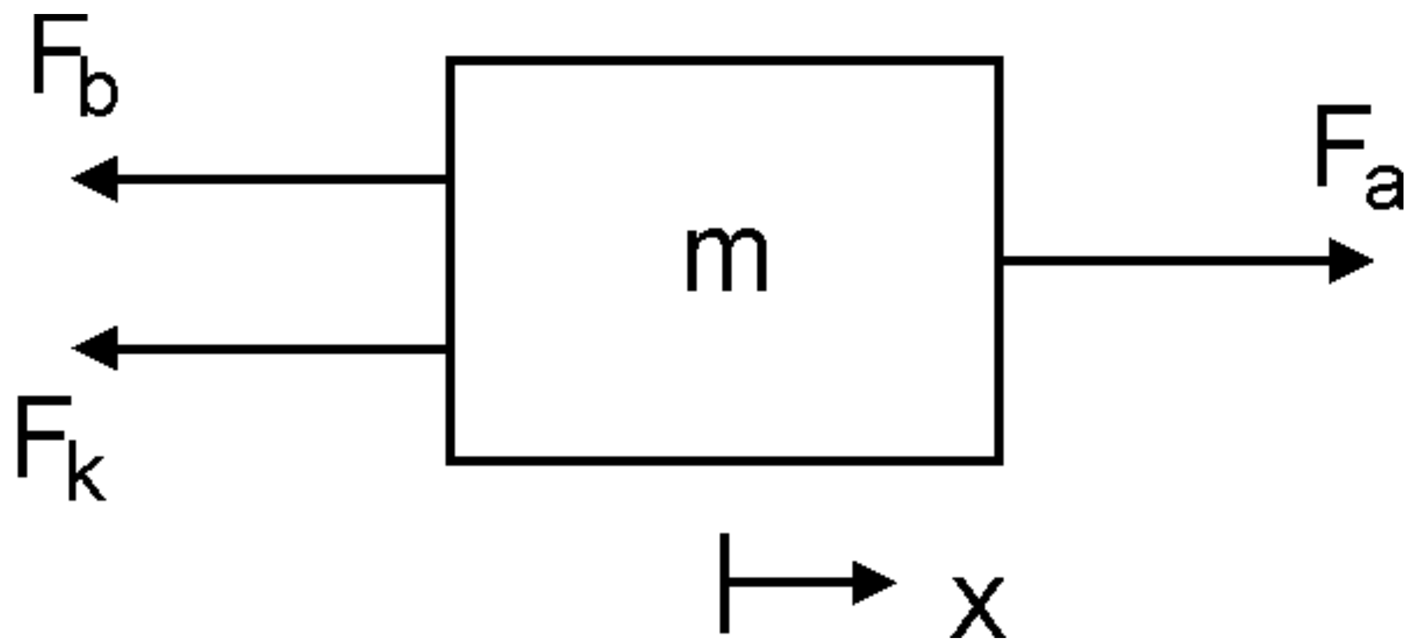
second order dynamic systems

mass-spring-damper



x = 0 at equilibrium

free body diagram



- $F_b = b\dot{x}$, $F_k = kx$
- sum forces, equate to inertia:

$$m\ddot{x} = F_a - F_b - F_k$$

$$m\ddot{x} + b\dot{x} + kx = F_a$$

