Computer Animation

Lesson 10 - Rigid Body Dynamics

Remi Ronfard, Nov 2019

Principle 11 - Solid drawing

11. Solid Drawing

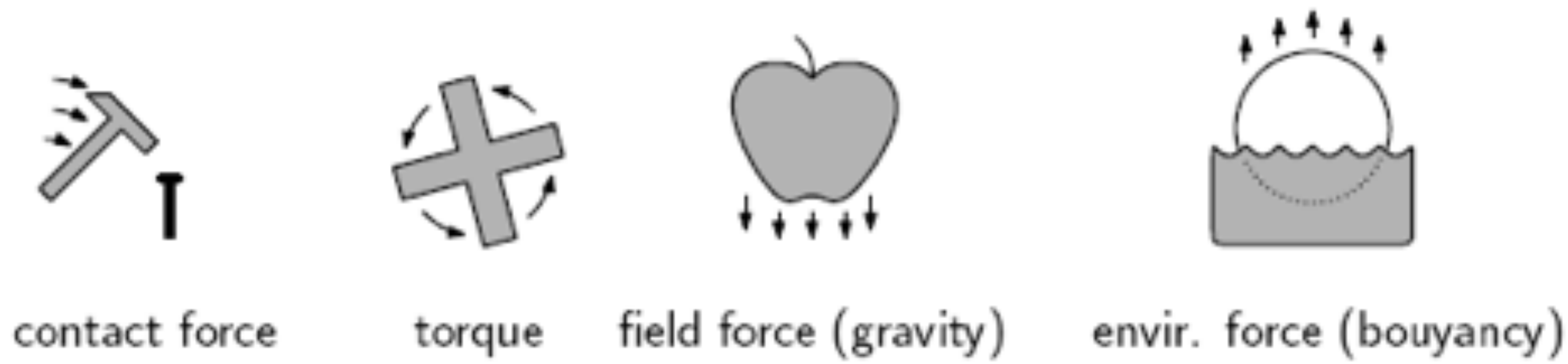
From kinematics to dynamics

- Compute positions, velocities, angles and angular velocities as a function of time
- Integration of Newton's equations of motion

$$\begin{bmatrix} p(t) \\ v(t) \\ q(t) \\ \omega(t) \end{bmatrix} \rightarrow \begin{bmatrix} p(t+\Delta t) \\ v(t+\Delta t) \\ q(t+\Delta t) \\ \omega(t+\Delta t) \end{bmatrix}.$$

Important concepts in dynamics

- Forces



- Center of mass and moment of inertia

Newton's Law and Euler integration

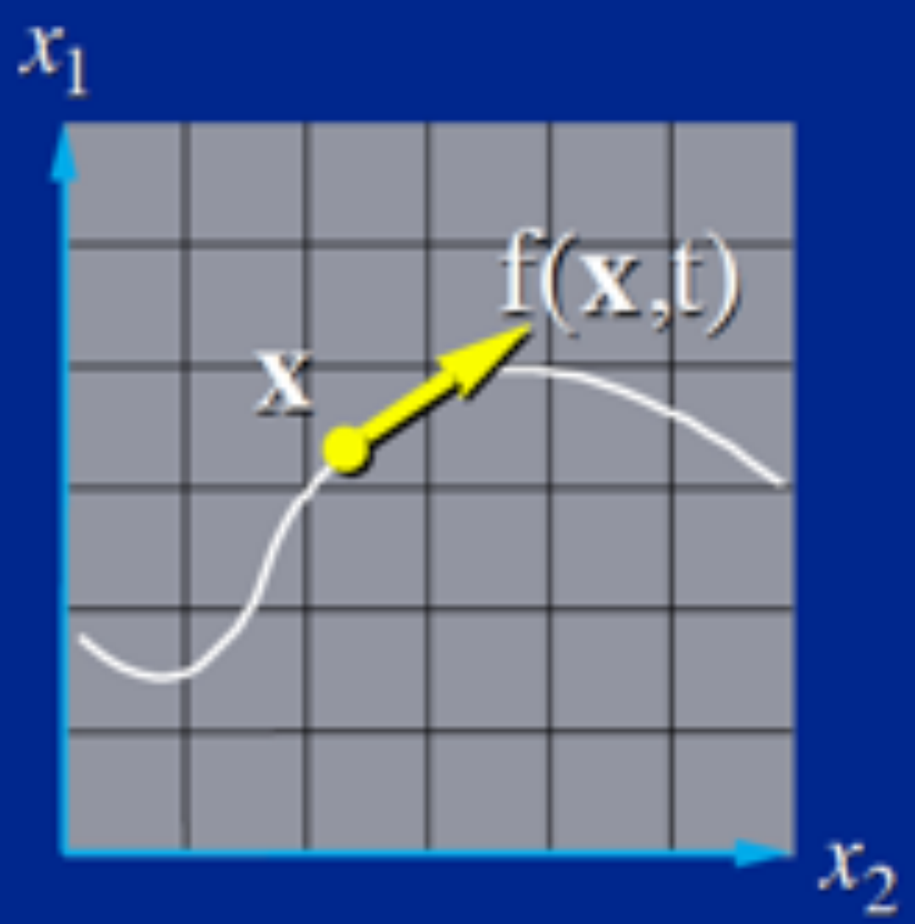
- Euler method, named after Leonhard Euler, is a first-order numerical procedure for solving ordinary differential equations (ODEs) with a given initial value.
- It is the most basic kind of explicit method for numerical integration of ordinary differential equations.

- Start at x0
- Compute acceleration from f = ma (Newton)
- Update velocity and position

$$\mathbf{x}' = \mathbf{x} + \mathbf{v} \cdot \Delta t$$

$$\mathbf{v}' = \mathbf{v} + \mathbf{a} \cdot \Delta t,$$

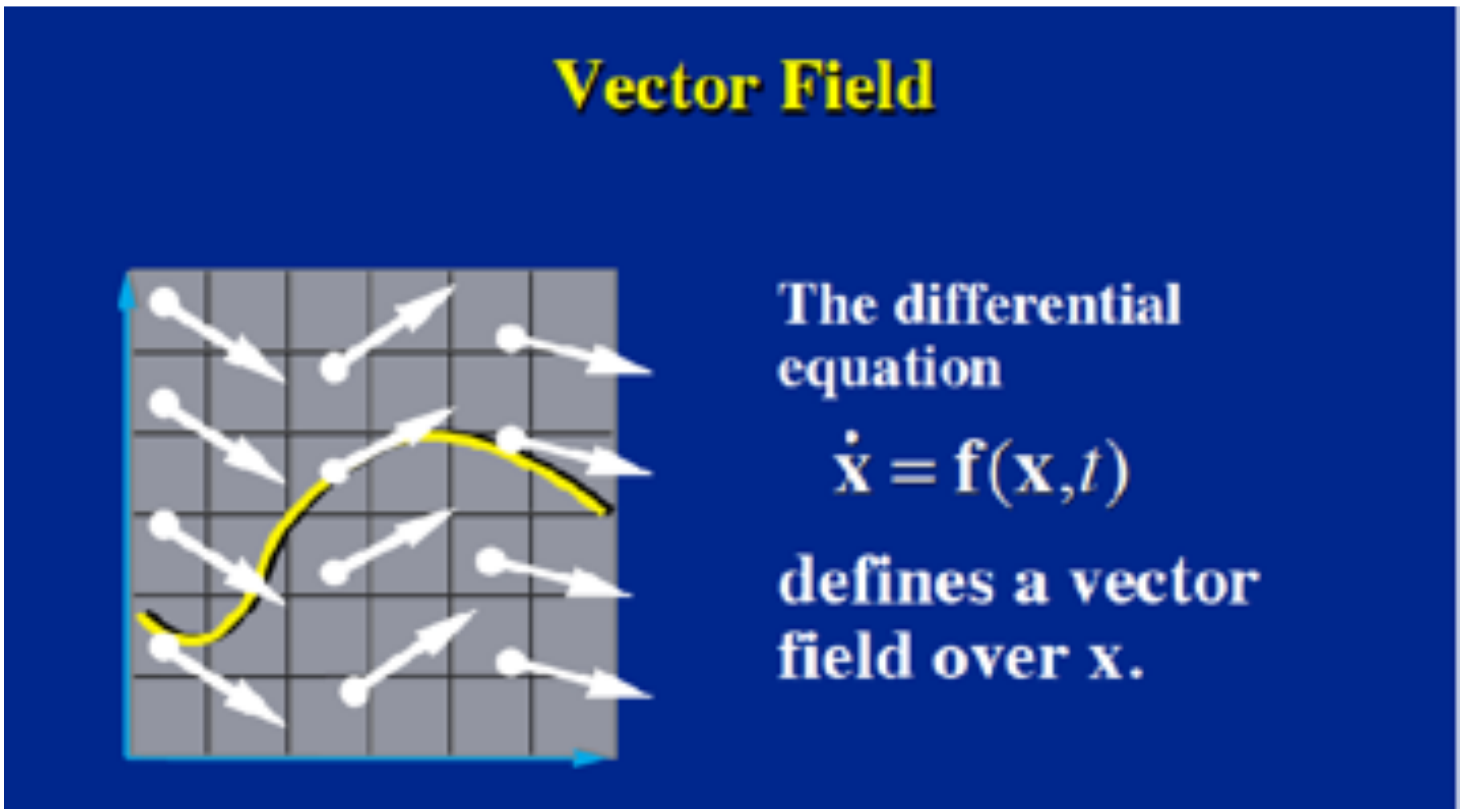
Ordinary differential equations



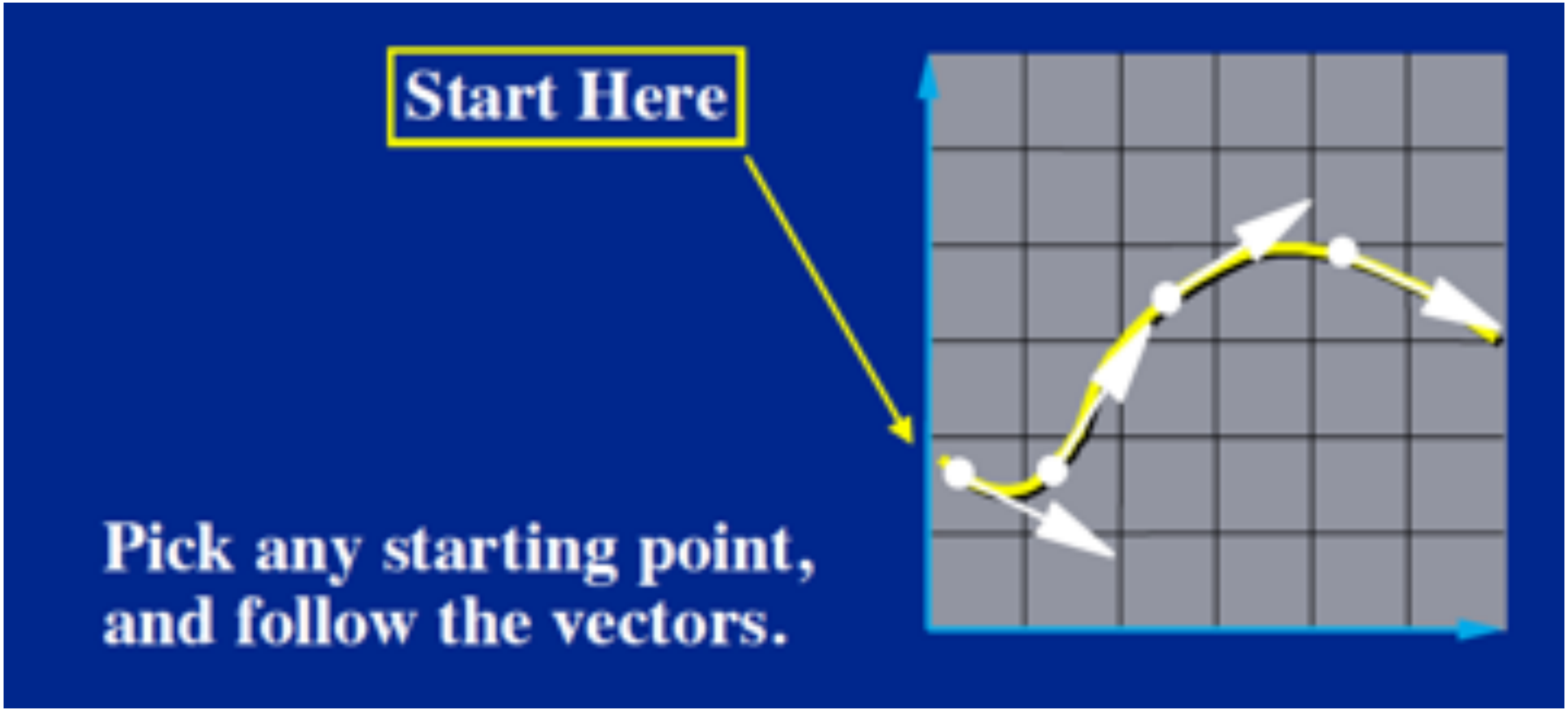
$$\dot{\mathbf{x}} = \mathbf{f}(\mathbf{x}, t)$$

- **x**(t): **a moving point.**
- **f(x,**t**): x's velocity.**

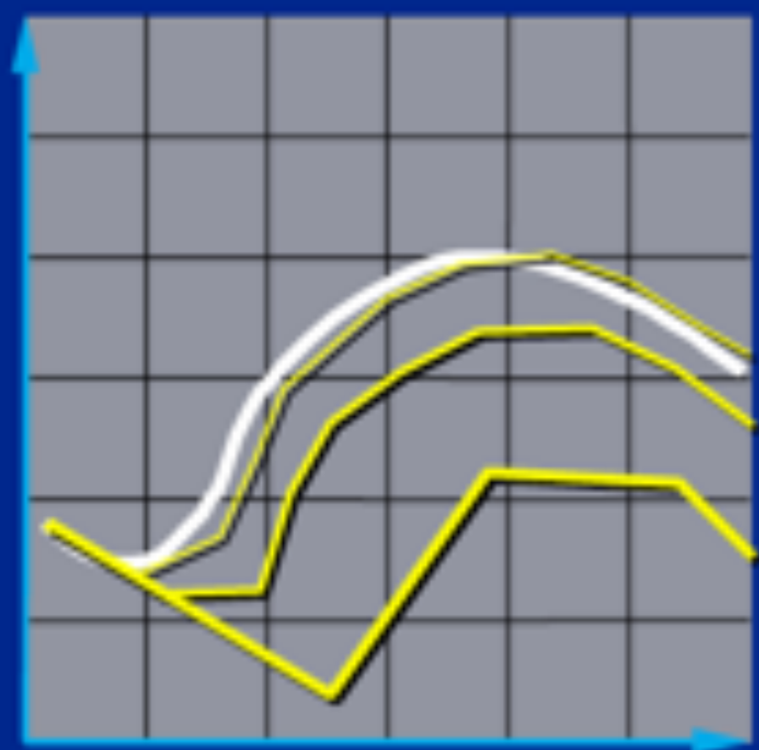
Ordinary differential equations



Ordinary differential equations



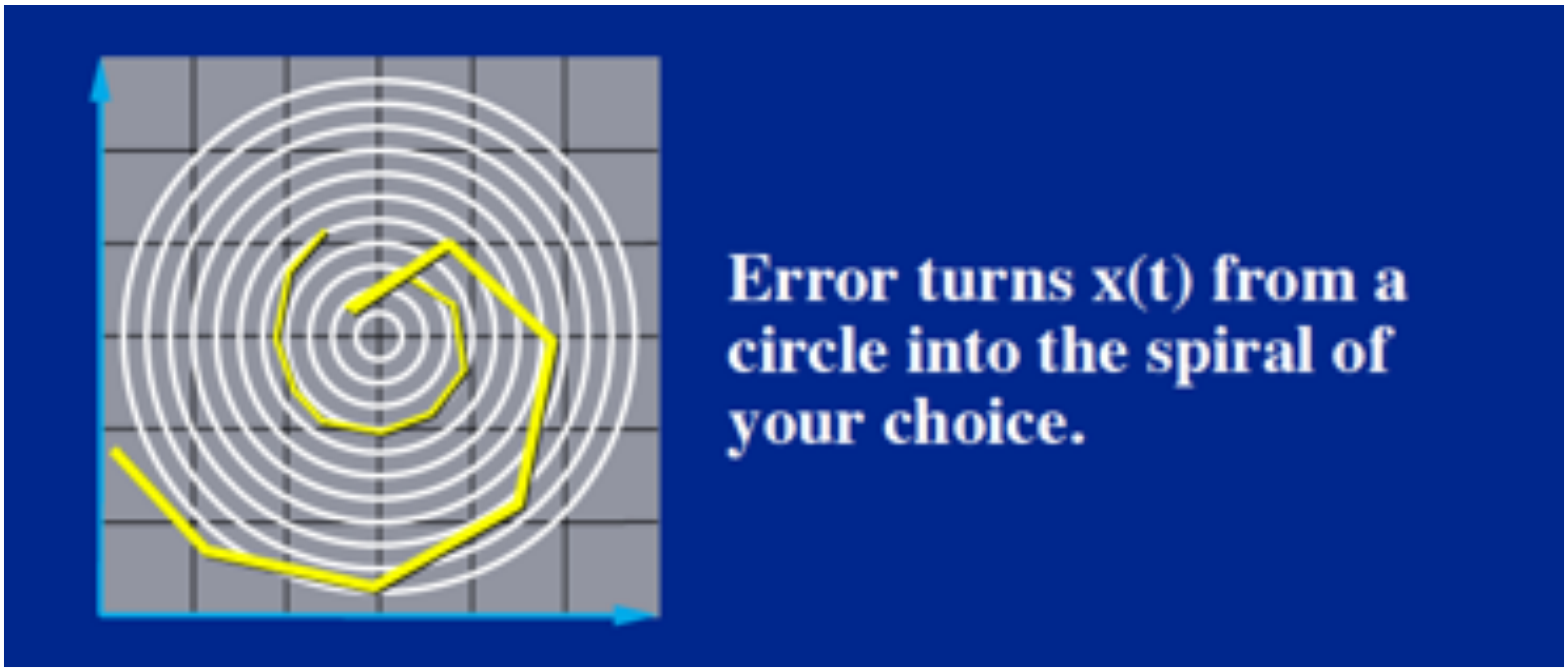
Euler's method



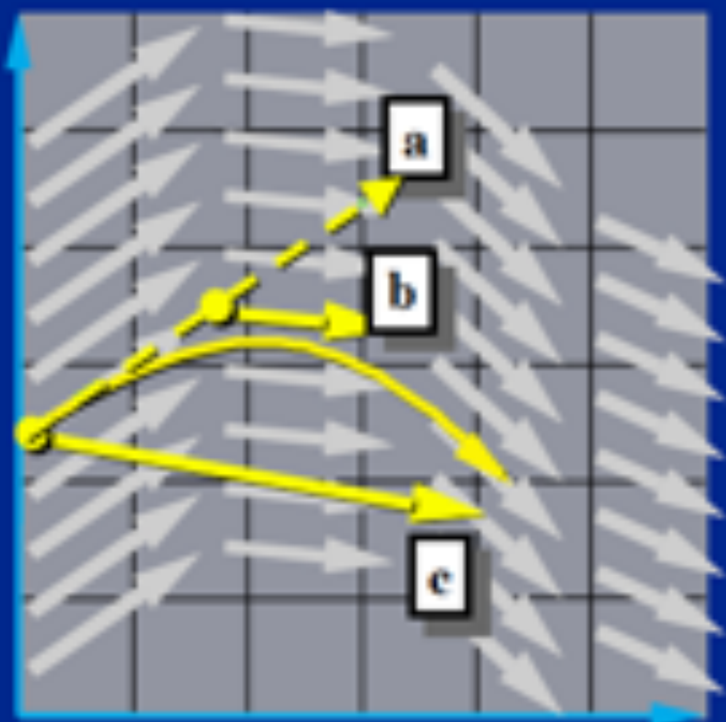
- Simplest numerical solution method
- Discrete time steps
- Bigger steps, bigger errors.

$$\mathbf{x}(t+\Delta t)=\mathbf{x}(t)+\Delta t\,\mathbf{f}(\mathbf{x},t)$$

Euler's method



Midpoint method



a. Compute an Euler step

$$\Delta \mathbf{x} = \Delta t \, \mathbf{f}(\mathbf{x}, t)$$

b. Evaluate f at the midpoint

$$\mathbf{f}_{\mathrm{mid}} = \mathbf{f}\left(\frac{\mathbf{x} + \Delta \mathbf{x}}{2}, \frac{t + \Delta t}{2} \right)$$

c. Take a step using the midpoint value

$$\mathbf{x}(t + \Delta t) = \mathbf{x}(t) + \Delta t \, \mathbf{f}_{\mathrm{mid}}$$

Other integration methods

- Euler's method is *1st Order.*
- The midpoint method is *2nd Order.*
- Just the tip of the iceberg. See *Numerical Recipes* for more.
- Helpful hints:
  - *Don't* use Euler's method (you will anyway.)
  - *Do* use adaptive step size.

Verlet integration

- Verlet integration was used by Carl Størmer to compute the trajectories of particles moving in a magnetic field (hence it is also called Störmer's method) and was popularized in molecular dynamics by French physicist Loup Verlet in 1967.
- It is frequently used to calculate trajectories of particles in molecular dynamics simulations and video games.
- Stability of the technique depends fairly heavily upon either a uniform update rate, or the ability to accurately identify positions at a small time delta into the past.

Verlet integration

- Current position is x
- Remember previous position x*

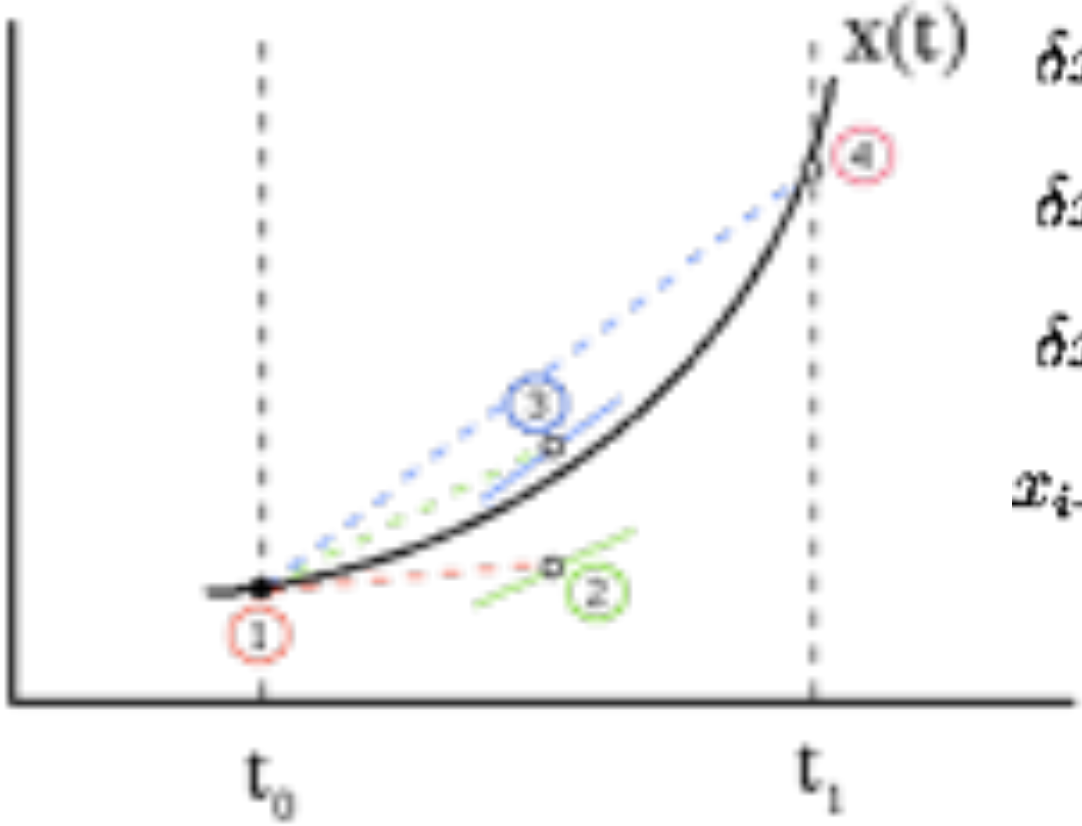
$$x' = 2\mathbf{x} - \mathbf{x}^* + \mathbf{a} \cdot \Delta t^2$$

$$\mathbf{x}^* = \mathbf{x}$$

- Update x and x*

- Advantage : velocity cannot go wrong !
- Applications to particle systems, mass-spring models, rigid and soft bodies

Runge-Kutta integration



$$
\begin{aligned}
\delta x_1 &= \delta t\, f(x_i, t_i),\\
\delta x_2 &= \delta t\, f(x_i + \tfrac{1}{2}\delta x_1, t_i + \tfrac{1}{2}\delta t),\\
\delta x_3 &= \delta t\, f(x_i + \tfrac{1}{2}\delta x_2, t_i + \tfrac{1}{2}\delta t),\\
\delta x_4 &= \delta t\, f(x_i + \delta x_3, t_i + \delta t),\\
x_{i+1} &= x_i + \tfrac{1}{6}\delta x_1 + \tfrac{1}{3}\delta x_2 + \tfrac{1}{3}\delta x_3 + \tfrac{1}{6}\delta x_4 + O(\delta t^5).
\end{aligned}
$$

Physically-based animation

- Vertex x with mass m and forces f
- Newton's equation
  $F = m\,a$
- Integrate to find $x(t+1)$ given $x(t)$ and $v(t)$ or $x(t)$ and $x(t-1)$

- Applications to mass-spring systems, cloth animation, fluid animation
- Smoothed Particle Hydrodynamics (SPH)

Integrating Newton's laws of motion

A Newtonian Particle

- Differential equation: $\mathbf{f} = m\mathbf{a}$
- Forces can depend on:
  - Position, Velocity, Time

$$\ddot{\mathbf{x}} = \frac{\mathbf{f}(\mathbf{x}, \dot{\mathbf{x}}, t)}{m}$$

Second-order motion equation

$$\ddot{\mathbf{x}} = \frac{\mathbf{f}(\mathbf{x}, \dot{\mathbf{x}}, t)}{m}$$

Not in our standard form because it has 2nd derivatives

$$\begin{cases} \dot{\mathbf{x}} = \mathbf{v} \\ \dot{\mathbf{v}} = \mathbf{f}/m \end{cases}$$

Add a new variable, **v**, to get a pair of coupled 1st order equations.

Phase space

$$\begin{bmatrix} \mathbf{x} \\ \mathbf{v} \end{bmatrix}$$

Concatenate **x** and **v** to make a 6-vector: *Position in Phase Space*.

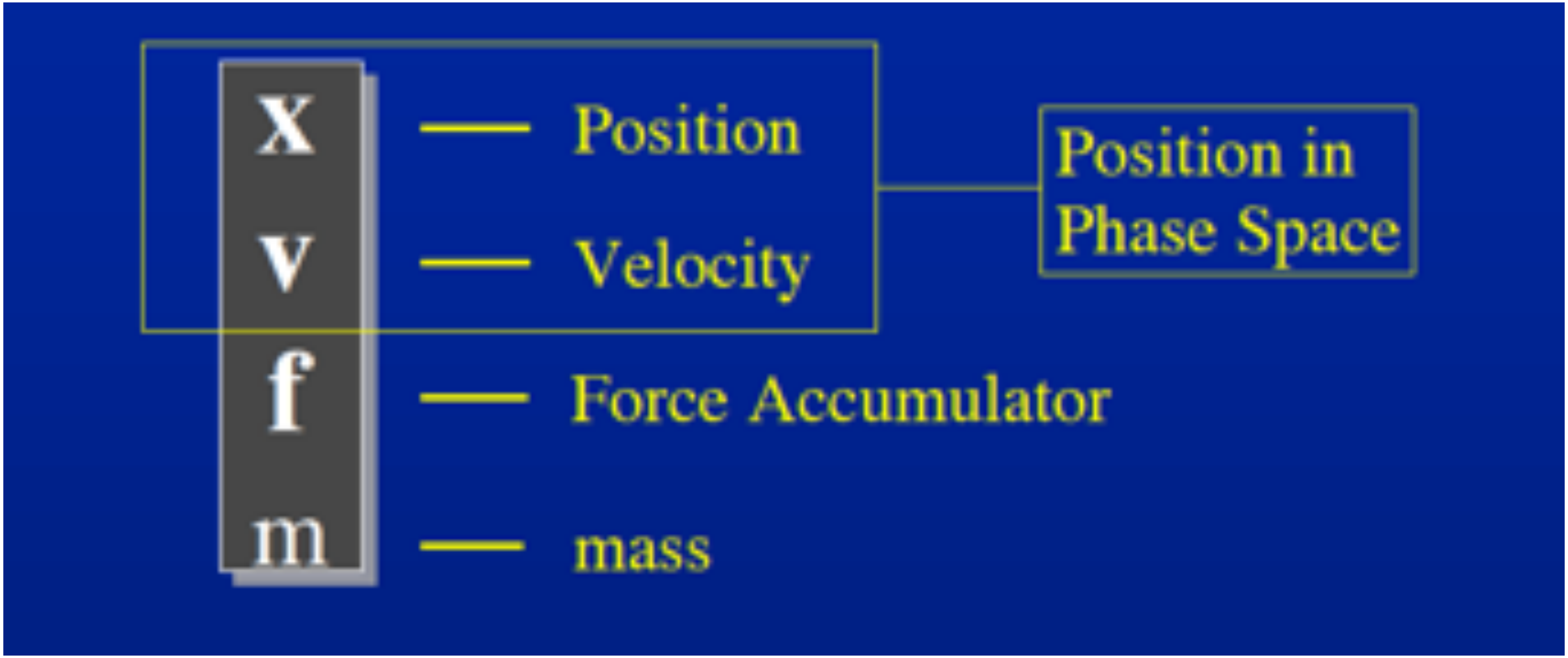
$$\begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{v}} \end{bmatrix}$$

Velocity in Phase Space: another 6-vector.

$$\begin{bmatrix} \dot{\mathbf{x}} \\ \dot{\mathbf{v}} \end{bmatrix} = \begin{bmatrix} \mathbf{v} \\ \mathbf{f}/m \end{bmatrix}$$

A vanilla 1st-order differential equation.

Particle structure



Forces acting on a particle

• Constant	gravity
• Position/time dependent	force fields
• Velocity-Dependent	drag
• n-ary	springs

Rigid body motion

We represent orientation as a rotation matrix[†] $\mathbf{R}(t)$. Points are transformed from body-space to world-space as:

$$p(t) = \mathbf{R}(t)\,p_0 + x(t)$$

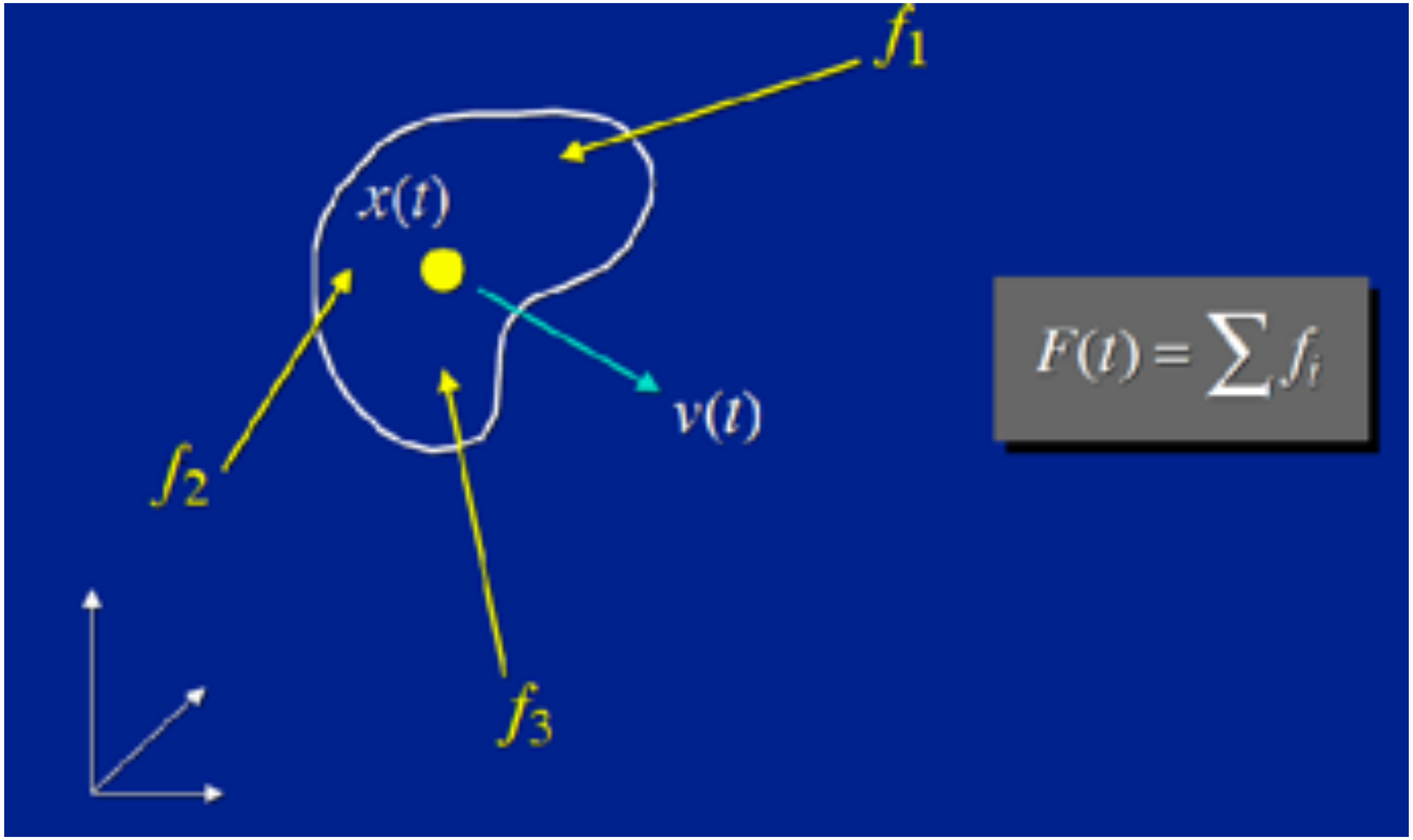
Rigid body motion

$\dot{\mathbf{R}}(t)$ and $\omega(t)$ are related by:

$$\frac{d}{dt}\mathbf{R}(t) = \begin{pmatrix} 0 & -\omega_z(t) & \omega_y(t) \\ \omega_z(t) & 0 & -\omega_x(t) \\ -\omega_y(t) & \omega_x(t) & 0 \end{pmatrix} \mathbf{R}(t)$$

$$= \omega(t)^* \mathbf{R}(t)$$

Rigid body motion



Moment of inertia

$$\mathbf{I}(t) = \begin{pmatrix} I_{xx} & I_{xy} & I_{xz} \\ I_{yx} & I_{yy} & I_{yz} \\ I_{zx} & I_{zy} & I_{zz} \end{pmatrix}$$

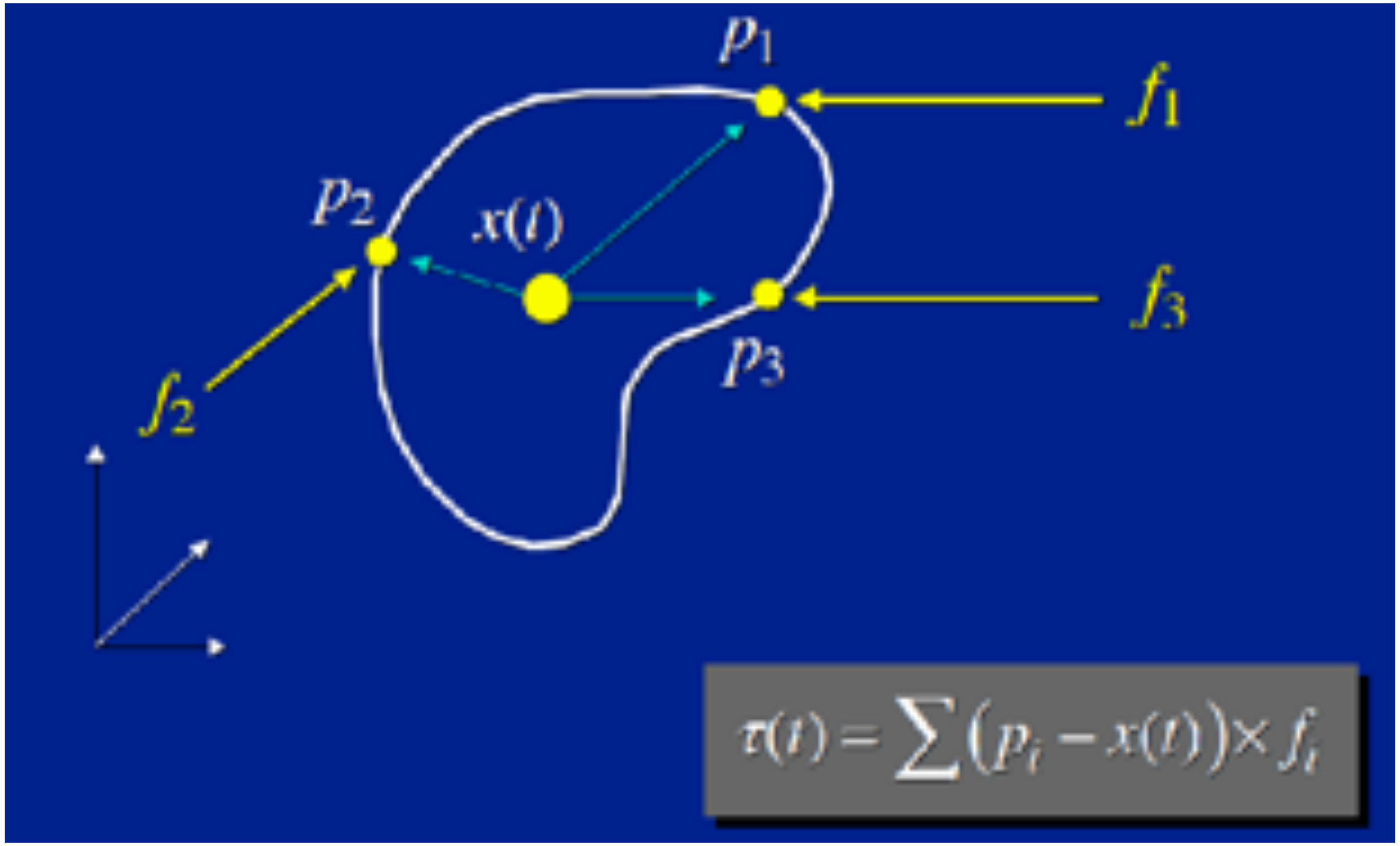
diagonal terms

$$I_{xx} = M \int_V (y^2 + z^2)\, dV$$

off-diagonal terms

$$I_{xy} = -M \int_V xy\, dV$$

Torque



Rigid body equation of motion

$$\frac{d}{dt}\mathbf{X}(t) = \frac{d}{dt}\begin{pmatrix} x(t) \\ \mathbf{R}(t) \\ Mv(t) \\ \mathbf{I}(t)\omega(t) \end{pmatrix} = \begin{pmatrix} v(t) \\ \omega(t)^{*}\mathbf{R}(t) \\ F(t) \\ \tau(t) \end{pmatrix}$$

Paper 10 - Artist-directed dynamics

Artist-Directed Dynamics for 2D Animation

Yunfei Bai[1,2] Danny M. Kaufman[1] C. Karen Liu[2] Jovan Popović[1]
[1]Adobe Research [2]Georgia Institute of Technology

Figure 1: *Artist-Directed Dynamics provides an interactive workflow with sparse keyframing, simulation, and example artwork. After creating an expressive walk with keyframes for hand and feet we add a single keyframe to the neck to introduce overlap (a); add two art examples to transform the motion to a sad walk (b, left); or two alternate poses for a sneak walk (b, right).*