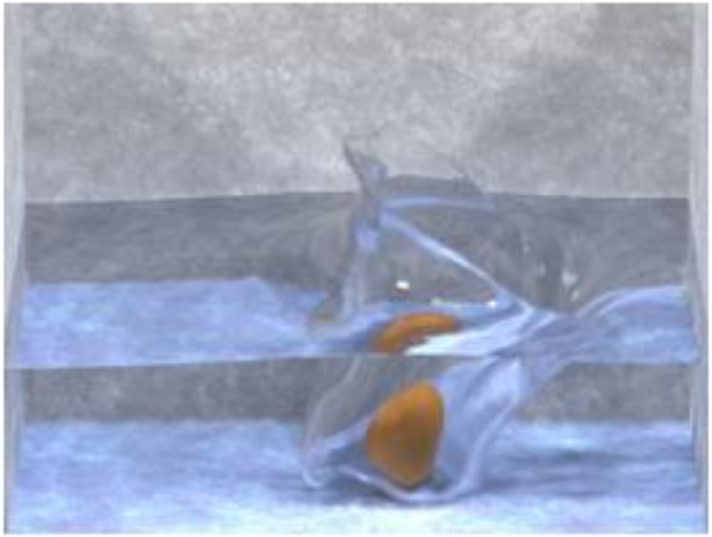
第五章 物理模拟

——基于物理模型的动画技术



基于物理模型的动画技术

- 物体的真实运动取决于其受力状况
- 利用物理模型生成所需要的真实感运动
- 力和力矩以及运动控制技术
  - 力和力矩控制物体的运动,不直观
  - 考虑物体的动力学属性
    - 物体的质量, 摩擦系数,刚度系数…
- 动力学方程的求解问题
  - 快速稳定的数值解法

主要内容

1. 刚体运动动力学方程的建立和求解方法
2. 物体运动时,如何防止物体间的相互贯穿

第五章 物理模拟

——基于物理模型的动画技术

刚体运动动力学方程

刚体运动

粒子的运动

一般刚体的运动

第五章 物理模拟

——基于物理模型的动画技术

粒子运动

- 粒子是一个具有一定**质量**但没有**大小**的**点**
- 由于没有大小，粒子在三维空间的运动只需要考虑**平移运动**，不需要考虑**旋转运动**
- 计算机动画系统中，关心粒子的**位移矢量**

粒子运动

- 粒子的运动方程

$$ma = \sum_{i=1}^{n} F_i$$

$$\left\{ \begin{aligned} \frac{dx(t)}{dt} &= v(t) \\ \frac{dv(t)}{dt} &= a(t) \end{aligned} \right\}$$

粒子运动

粒子运动方程的一般形式

$$m\frac{d^2x(t)}{dt^2}+d\frac{dx(t)}{dt}+hx(t)=\sum_{i=1}^{n}F_i$$

- 其中，***m*,*d*,*h***是三个已知量
- 上述方程是一个矢量方程

粒子运动

- 将粒子在 t 时刻的状态矢量 $y(t)$ 表示为由粒子的位置矢量 $x(t)$ 和速度矢量 $v(t)$ 组成的高维矢量

$$y(t) = \begin{pmatrix} x(t) \\ v(t) \end{pmatrix}$$

$$\frac{dy(t)}{dt} = \frac{d}{dt}\begin{pmatrix} x(t) \\ v(t) \end{pmatrix} = \begin{bmatrix} v(t) \\ \frac{\sum F_i}{m} \end{bmatrix}$$

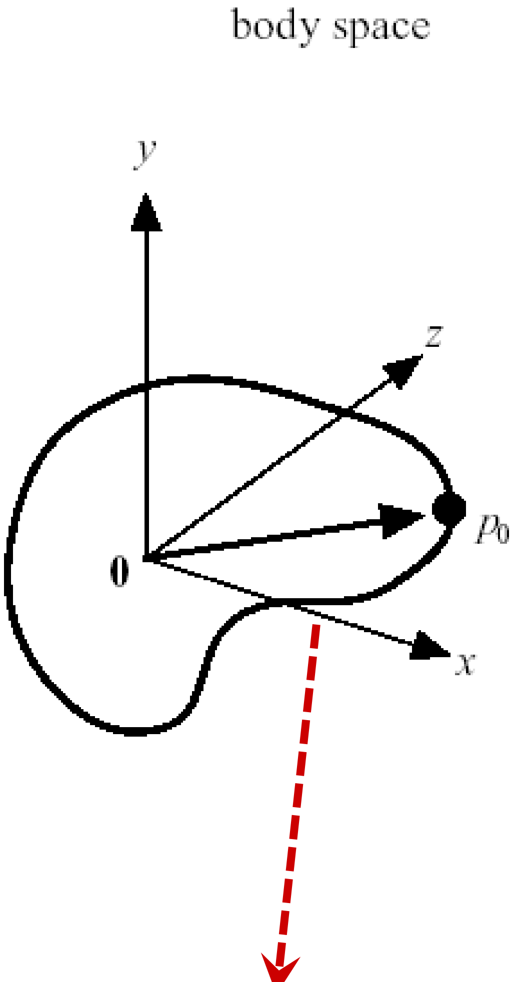
Maya系统将上述方程引入到普通的粒子系统中,成功地模拟了粒子在各时刻的运动物理状态。

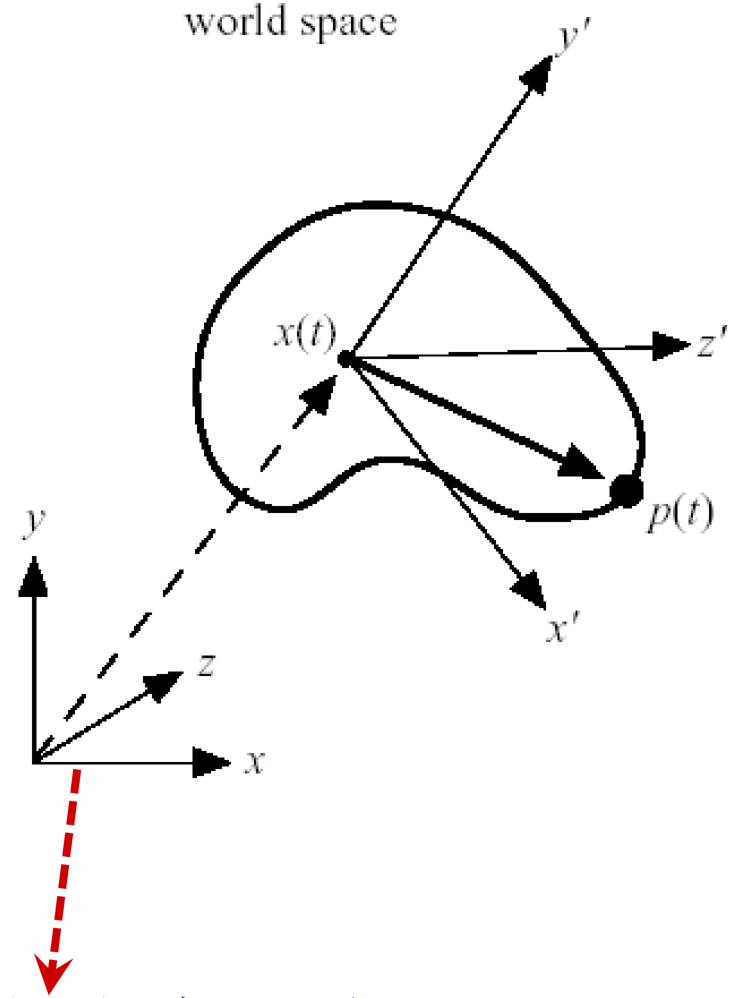
第五章 物理模拟

------基于物理模型的动画技术

一般刚体的运动

- 对于具有一定**体积**的刚体而言，**旋转**运动与**平移**运动同样重要
- 刚体动力学系统中的坐标系
  - **惯性坐标系**
    - 固定在空间中的固定参考系
  - **刚体坐标系**
    - 附着在一个刚体上，随刚体的运动而变化





刚体坐标系　　惯性坐标系

变换矩阵

- 刚体在t时刻的惯性坐标 $Q(t)$ 与其对应的刚体坐标 Q' 的关系

$$Q(t) = R(t) * Q' + X(t)$$

刚体坐标系到惯性坐标系的旋转变换矩阵

Q'相对于刚体坐标系始终相对静止

刚体坐标系到惯性坐标系的平移向量

- 称 $X(t)$ 和 $R(t)$ 为物体的位置和方向

刚体的平移运动

- 一个刚体运动可以被分解为平移和旋转两部分，二者可以独立计算
- 刚体的平移运动等价于其质心粒子的平移运动----线速度 $v(t)$

$$y(t) = \begin{pmatrix} x(t) \\ v(t) \end{pmatrix}$$

$$\frac{dy(t)}{dt} = \frac{d}{dt}\begin{pmatrix} x(t) \\ v(t) \end{pmatrix} = \begin{bmatrix} v(t) \\ \dfrac{\sum F_i}{m} \end{bmatrix}$$

刚体的旋转运动

- 刚体绕通过质心的轴旋转
- 旋转的角速度矢量用 $\omega(t)$ 表示
  - $\omega(t)$ 的方向为旋转轴的方向
  - $\omega(t)$ 的幅度 $|\omega(t)|$ 表示旋转的快慢

角速度 $\omega(t)$ 与方向矩阵 $R(t)$

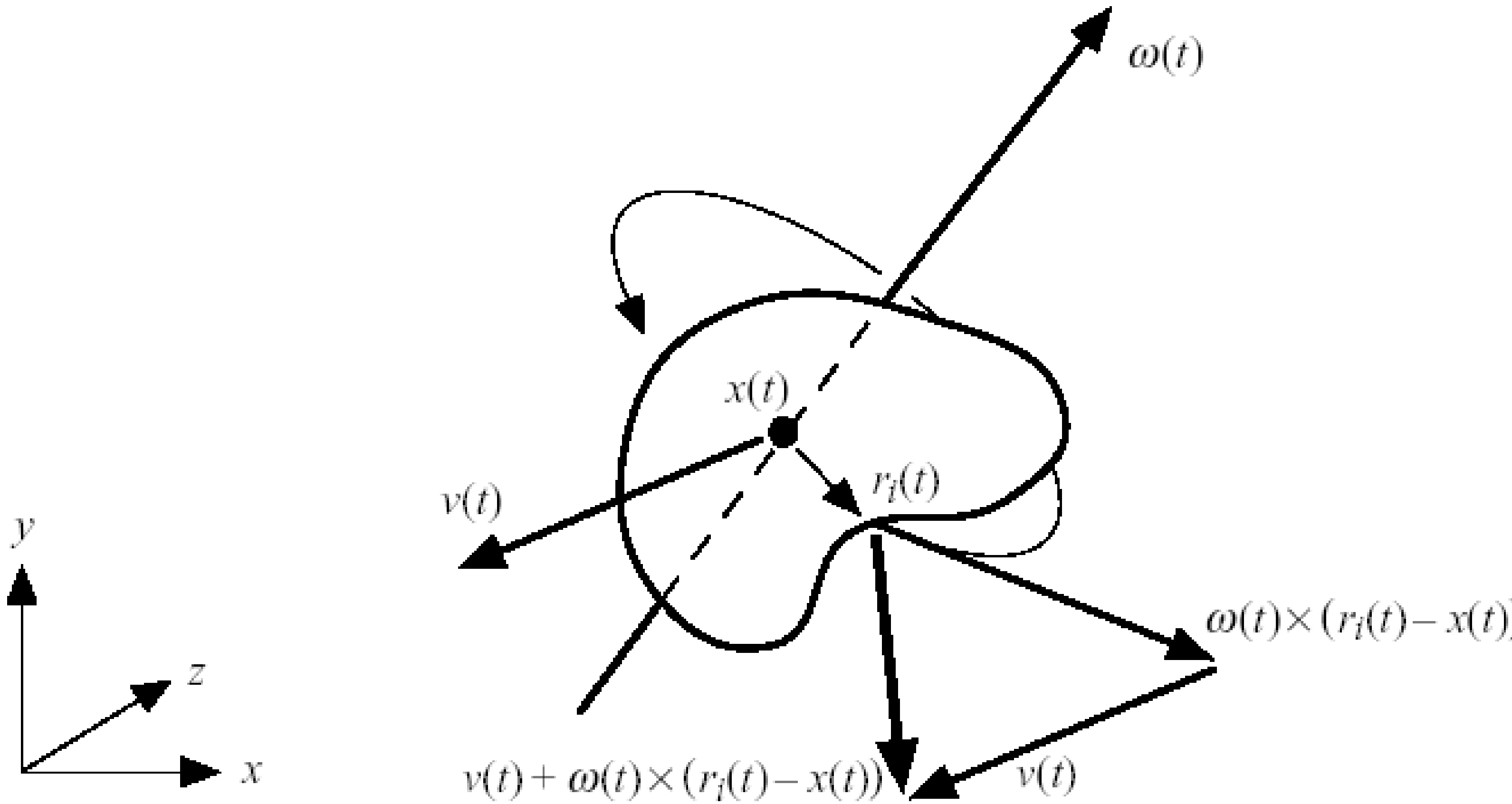
$$\frac{dR(t)}{dt} = \omega^*(t)R(t)$$

$$\omega(t) = (\omega_x(t), \omega_y(t), \omega_z(t))$$

$$\omega^*(t) = \begin{pmatrix} 0 & \omega_z(t) & -\omega_y(t) \\ -\omega_z(t) & 0 & \omega_x(t) \\ \omega_y(t) & -\omega_x(t) & 0 \end{pmatrix}$$

总的运动速度

- 对于刚体上的一点 $r_i(t)$，总的运动速度为



$$\dot{r}(t)=\omega(t)\times(r_i(t)-x(t))+v(t)$$

力矩(Torque)

力矩说明了所有外力在刚体上的分布

$$\tau(t) = \sum \tau_i(t) = \sum (r_i(t) - x(t)) \times F_i(t)$$

$$F(t) = \sum F_i(t)$$

线动量 $P(t)$

$$P(t) = M * v(t)$$

$$\dot{P}(t) = F(t)$$

$$\dot{v}(t) = \frac{\dot{P}(t)}{M}$$

$$\dot{v}(t) = \frac{F(t)}{M}$$

角动量 $L(t)$

$$L(t) = I(t) * \omega(t)$$

惯性张量矩阵

$$\boxed{\dot{L}(t) = \tau(t)}$$

力矩

$$\boxed{\omega(t) = (\omega_x(t), \omega_y(t), \omega_z(t))^T}$$

惯性张量

- 不同质量分布的刚体, 其质心位置不同, 产生的旋转运动也不相同
- 物体的质量分布用惯性张量 I 表示

$$I = \begin{bmatrix} \int (y^2+z^2)dm & -\int xydm & -\int xzdm \\ -\int xydm & \int (x^2+z^2)dm & -\int yzdm \\ -\int xzdm & -\int yzdm & \int (x^2+y^2)dm \end{bmatrix}$$

每个积分均在整个刚体上进行

状态向量

- **为统一求解, 将刚体在 t 时刻的状态向量 $y(t)$ 置为由其质心位置 $x(t)$, 旋转矩阵 $R(t)$ 及其线动量 $P(t)$, 角动量 $L(t)$ 组成的高维矢量**

$$y(t) = \begin{bmatrix} x(t) \\ R(t) \\ P(t) \\ L(t) \end{bmatrix}$$

$$\frac{dy(t)}{dt} = \begin{bmatrix} v(t) \\ \omega^{*}(t)R(t) \\ F(t) \\ \tau(t) \end{bmatrix}$$

第五章 物理模拟

——基于物理模型的动画技术

运动方程的数值解

运动方程的数值解

- 每个微分方程都可能有一组解, 需要从中确定一个特定的解
- 为了唯一地确定微分方程的解, 需要增加一些约束条件
  - 初值约束
    - 粒子的初始位移和初始速度
    - 刚体运动中, 自由度的位置和速度的初始值
  - 边界约束

一阶常微分方程的求解

$$\begin{cases} \dfrac{dX}{dt} = f(X,t) \\ X(t_0) = X_0 \end{cases}$$

$X(x_1, x_2, \ldots, x_n)$是一个 ***n*** **维向量**

- 假设该初值问题的**精确解**为 $G(t)$
- 计算机动画中，需要的是 $G(t)$ 在某些**时刻**的值

$$h = \frac{t_n - t_0}{n} \qquad [t_0, t_n] \qquad t_i = t_0 + i * h$$

第五章 物理模拟

------基于物理模型的动画技术

一阶常微分方程的求解

- **显式求解方法**
  - 仅仅利用已知的函数*f*, 求解方法很直接
- **隐式求解**
  - 利用需求解的函数*X*, 需要用迭代过程求解

显式求解方法

- 欧拉方法
  - ❖最古老、最著名的常微分方程数值解法
  - ❖利用泰勒展开式中的一阶项
  - ❖方法简单, 计算效率高, 但精度不高
- Runge-Kutta算法
  - ❖ 利用泰勒展开式中的高阶项提高计算精度
- 多步法
  - ❖ 不损失计算精度的前提下, 提高计算速度

欧拉方法

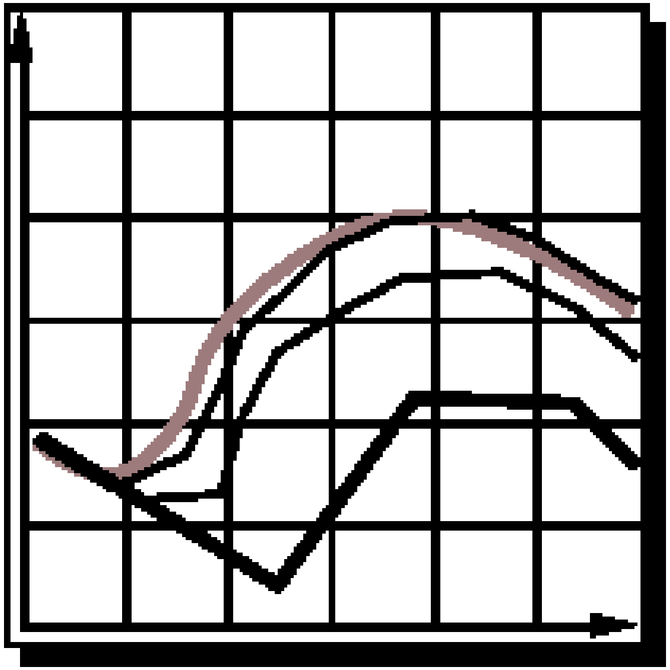
$$G(t_{i+1}) = G(t_i) + h * f(G(t_i), t_i) \boxed{+ O(h^2)}$$

$$\Updownarrow \text{ 递推公式}$$

$$X_{i+1} = X_i + h * f(X_i, t_i)$$

- 当 $f(x,t)$ 变化较为平坦时, 该方法比较有效
- 当 $f(x,t)$ 变化剧烈,甚至不连续时,无法保证精度要求

欧拉方法的缺点:不准确



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

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

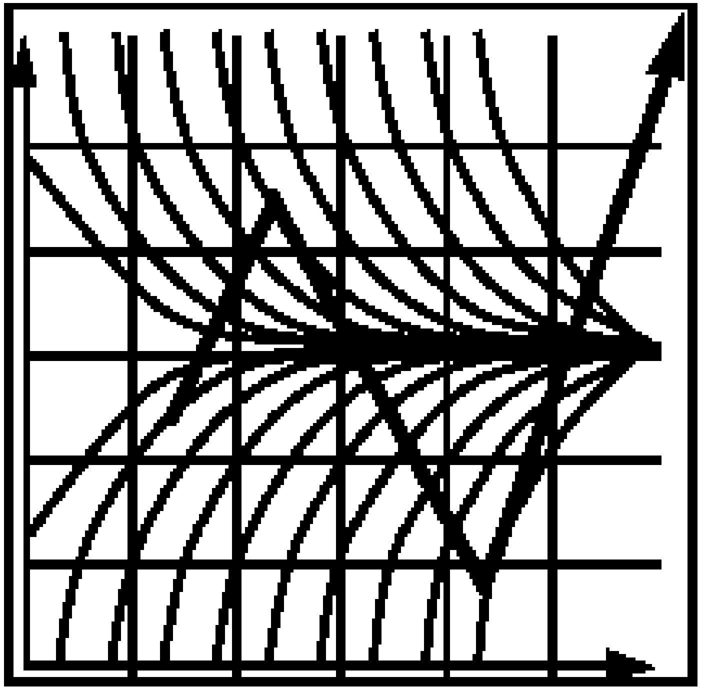
欧拉方法的缺点:不稳定

- 步长过大时, 数值解会发散

$$\dot{x} = -kx \longrightarrow x = e^{-kt}$$

$$X_{i+1} = X_i - h * kX_i = (1 - hk) X_i$$

➢如果步长$h>2/k$, $|\triangle X|>2|X|$, 解发散



欧拉解发散的情况

改进的欧拉方法

- **将均匀步长改为可变步长**
  - 利用泰勒展开式的二次项以及给定的误差控制步长
  - 第 i 步的步长 h_i 满足:

$$h_i = \sqrt{\frac{2\varepsilon}{\dfrac{\delta f(X_i, t_i)}{\delta X}}}$$

Runge-Kutta算法

- 利用泰勒展开式中的高阶项提高计算精度
- 利用迭代计算, 简化高阶导数的计算
- n 阶 Runge-Kutta 算法具有 $O(h^{n+1})$ 的计算精度
- 实际使用时, 常取 **n=4**
- 二阶 **Runge-Kutta** 算法也称为<u>中点法</u>

二阶 Runge-Kutta 算法---中点法

$$X_{i+1} = X_i + h * f\left(X_i + \frac{h}{2} * f(X_i, t_i), t_i\right)$$

$$= X_i + h * f\left(X_i + \frac{h}{2} * \dot{X}_i, t_i\right)$$

中间点处的函数值

四阶 Runge-Kutta 算法

$$X_{i+1} = X_i + \frac{1}{6}K_1 + \frac{1}{3}K_2 + \frac{1}{3}K_3 + \frac{1}{6}K_4$$

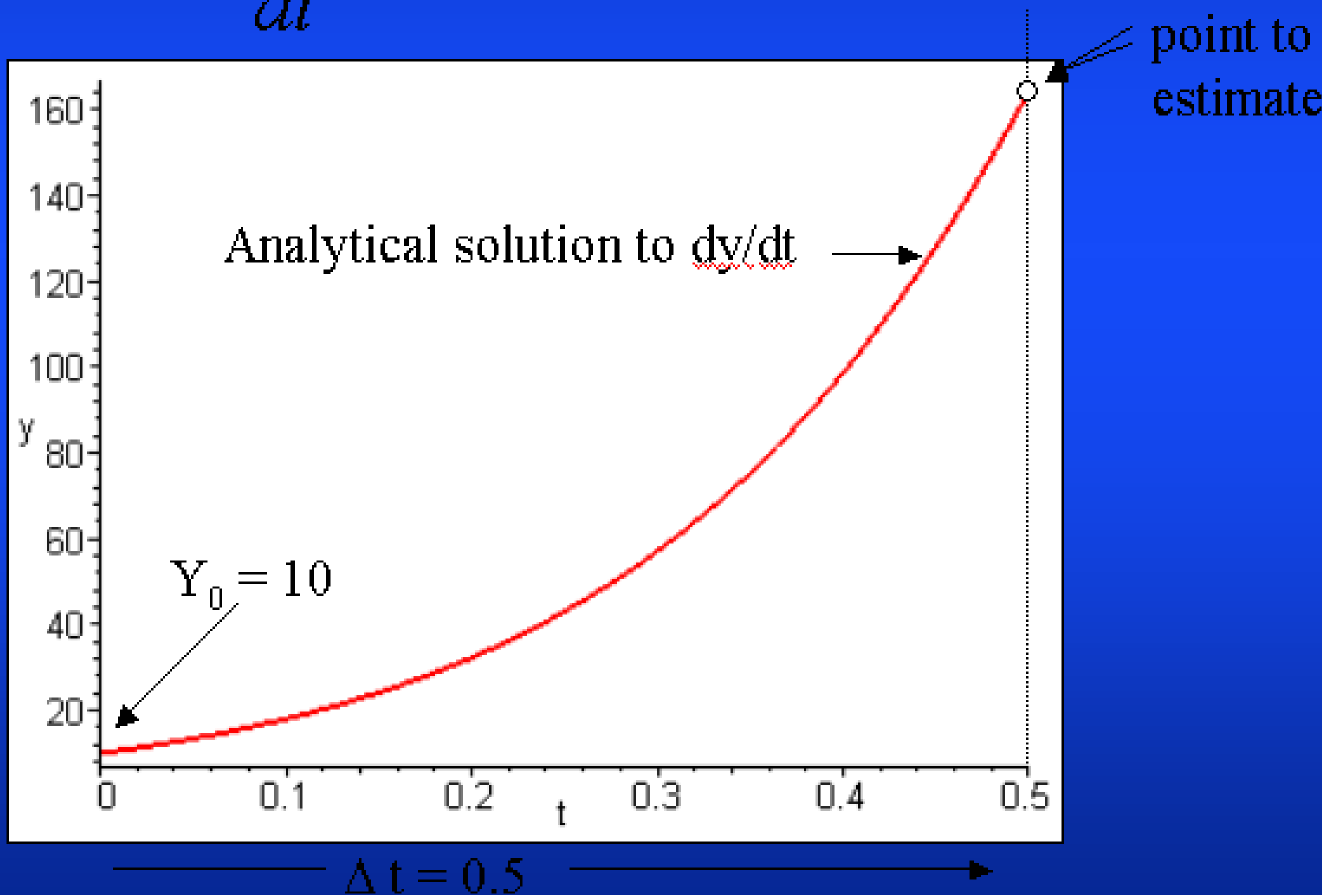
$$K_1 = h * f(X_i, t_i)$$

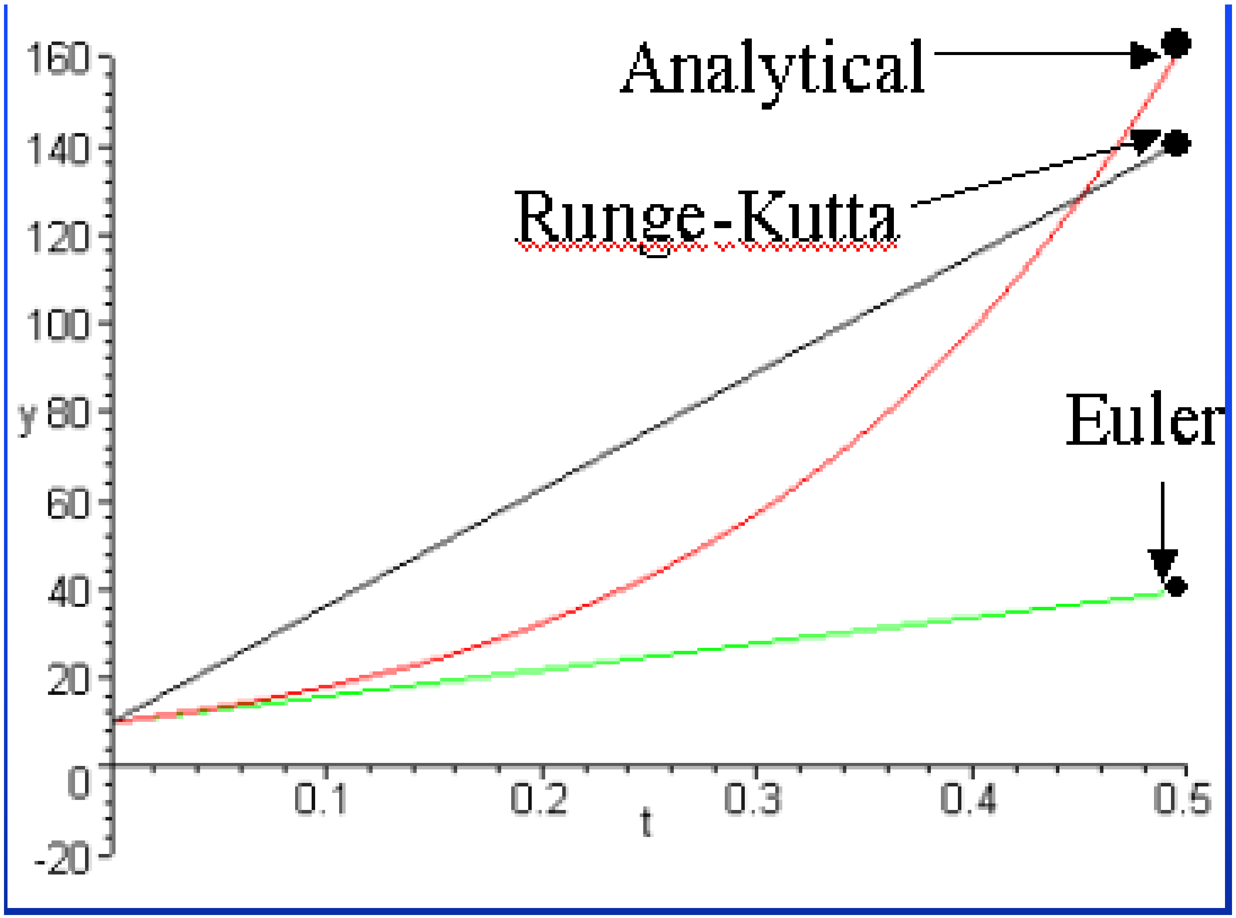
$$K_2 = h * f(X_i + \frac{K_1}{2}, t_i + \frac{h}{2})$$

$$K_3 = h * f(X_i + \frac{K_2}{2}, t_i + \frac{h}{2})$$

$$K_4 = h * f(X_i + \frac{K_3}{2}, t_i + h)$$

$$\frac{dy}{dt} = 6y - .007y^2$$





Analytical
Runge-Kutta
Euler
y
t
160
140
120
100
80
60
40
20
0
-20
0.1
0.2
0.3
0.4
0.5

多步法

- 在四阶 Runge-Kutta 算法中，每迭代一次，需要计算 $f(x,t)$ 函数四次，计算量比较大
- 良好的解决方案需要能充分利用已经计算好的结果来计算下一点，从而保证每一迭代步骤中只需计算一次函数值 $f(x,t)$, 而不损失计算精度, 称此类方法为多步法

多步法

- Adams-Bashforth方法是一种比较常用的**四阶**多步法

$$X_{i+1} = X_i + \frac{h}{24}[55f(X_i,t_i) - 59f(X_{i-1},t_{i-1}) + 37f(X_{i-2},t_{i-2}) - 9f(X_{i-3},t_{i-3})]$$

- 使用Runge-Kutta算法或者其他方法确定**最初的四个**值 X_0, X_1, X_2, X_3

步长的选取

- 在 $f(x,t)$ 比较复杂时, **Runge-Kutta**方法和多步法也需要考虑使用<u>非固定</u>的步长
- 一个使用的步长选取方法:
  - 首先<u>确定等间隔步长 h</u>
  - 每次迭代时, <u>调整 h</u>

调整h的方法

假设所选取的求解方法的**求解精度**为$O(h^{n+1})$

$$e = \| X(t_i + h) - X(t_i + h/2) \|$$

$$h_{new} = h * \left(\frac{e_{\max}}{e}\right)^{\frac{1}{n+1}}$$

- 公式中，e_{max}是用户预先指定的每步迭代允许的<u>最大误差</u>

第五章 物理模拟

——基于物理模型的动画技术

隐式求解技术

- 隐式求解过程比显式求解过程更加**稳定**，但需要的**计算量**更大
- 利用需求解的函数，需要用**迭代**过程求解
  - 隐式欧拉方法
  - 预测校正方法

隐式欧拉方法

隐式欧拉方法

$$X_{i+1} = X_i + h * f(X_{i+1}, t_{i+1})$$

显式欧拉方法

$$X_{i+1} = X_i + h * f(X_i, t_i)$$

隐式欧拉方法

- 利用分段线性函数来逼近 $f(X_{i+1}, t_{i+1})$

$$f(X_{i+1}, t_{i+1}) = f(X_i, t_i) + \frac{\partial f}{\partial X}(X_i, t_i)\Delta X i$$

$$\Delta X_i = X_{i+1} - X_i$$

$$X_{i+1} = X_i + h * f(X_{i+1}, t_{i+1})$$

隐式欧拉方法

$$\Delta X_i = h*[f(X_i, t_i) + \frac{\partial f}{\partial X}(X_i, t_i)\Delta X_i]$$

$$\Delta X_i = (\frac{1}{h}E - \frac{\partial f}{\partial X}(X_i, t_i))^{-1} f(X_i, t_i)$$

单位矩阵

矩阵

隐式欧拉方法

- 隐式欧拉方法需要计算逆矩阵, 因此其<u>计算量</u>比显式欧拉方法<u>大</u>.
- 对于许多实际问题, 需要计算的矩阵为一个稀疏矩阵, 可以在线性时间内计算其<u>逆矩阵</u>, 进而计算出X_{i+1}

隐式欧拉方法

$$\begin{cases} \dfrac{dX}{dt} = \dfrac{d}{dt}\begin{pmatrix} x(t) \\ y(t) \end{pmatrix} = \begin{pmatrix} -x(t) \\ -ky(t) \end{pmatrix} \\ X(t0) = \begin{pmatrix} x0 \\ y0 \end{pmatrix} \end{cases}$$

$$f(X,t) = \begin{pmatrix} -x(t) \\ -ky(t) \end{pmatrix}$$

$$\frac{\partial f}{\partial X} = \begin{pmatrix} -1 & 0 \\ 0 & -k \end{pmatrix}$$

$$\Delta X_0 = \begin{pmatrix} \dfrac{1+h}{h} & 0 \\ 0 & \dfrac{1+k*h}{h} \end{pmatrix}^{-1} \begin{pmatrix} -x_0 \\ -ky_0 \end{pmatrix} = -\begin{pmatrix} \dfrac{h}{h+1}x_0 \\ \dfrac{h}{1+hk}ky_0 \end{pmatrix}$$

$$X_1 = X_0 + \Delta X_0$$

隐式欧拉方法

$$\Delta X_0 = \begin{pmatrix} \frac{1+h}{h} & 0 \\ 0 & \frac{1+k*h}{h} \end{pmatrix} \begin{pmatrix} -x_0 \\ -ky_0 \end{pmatrix} = -\begin{pmatrix} \frac{h}{h+1} x_0 \\ \frac{h}{1+hk} ky_0 \end{pmatrix}$$

$$\lim_{h\to\infty} \Delta X_0 = -\begin{pmatrix} x_0 \\ y_0 \end{pmatrix}$$

$$\lim_{h\to\infty} X_1 = \lim_{h\to\infty}(X_0 + \vartriangle X_0) = X_0 - \begin{pmatrix} x_0 \\ y_0 \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

隐式欧拉方法

- 由极限可以看到, 利用隐式欧拉方法时可以使用充分大的步长来求解运动方程，因此可以使用比较少的迭代步骤求解，这有效地弥补了由于求矩阵逆运算带来的较大的计算量

预测校正法

- 将隐式欧拉方法推广到高阶导数, 对应的方法称为预测校正法(**Predictor-corrector**)
- 常用的**Adams-Moulton**的 **4** 阶预测校正算法的迭代公式

$$X_{i+1} = X_i + \frac{h}{24}[9f(X_{i+1}, t_{i+1}) + 19f(X_i, t_i) - 5f(X_{i-1}, t_{i-1}) + f(X_{i-2}, t_{i-2})]$$

隐式求解技术

- 一般而言，隐式求解技术较显式求解方法稳定，可靠，因此该技术对于复杂的动力学系统方程的求解具有重要意义；
- 数值计算领域，有大量的一阶微分方程数值求解方法。

二阶常微分方程的数值求解

- 许多动力学问题可以归结为二阶常微分方程：

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

- 数值解法
  - ❖差分逼近技术直接求解二阶常微分方程
  - ❖将二阶常微分方程转化为一阶常微分方程

二阶常微分方程的数值求解

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

$$v(t) = \frac{dX}{dt}$$

$$\frac{d}{dt}\begin{pmatrix} X(t) \\ v(t) \end{pmatrix} = \begin{pmatrix} v(t) \\ f(X(t), v(t)) \end{pmatrix}$$

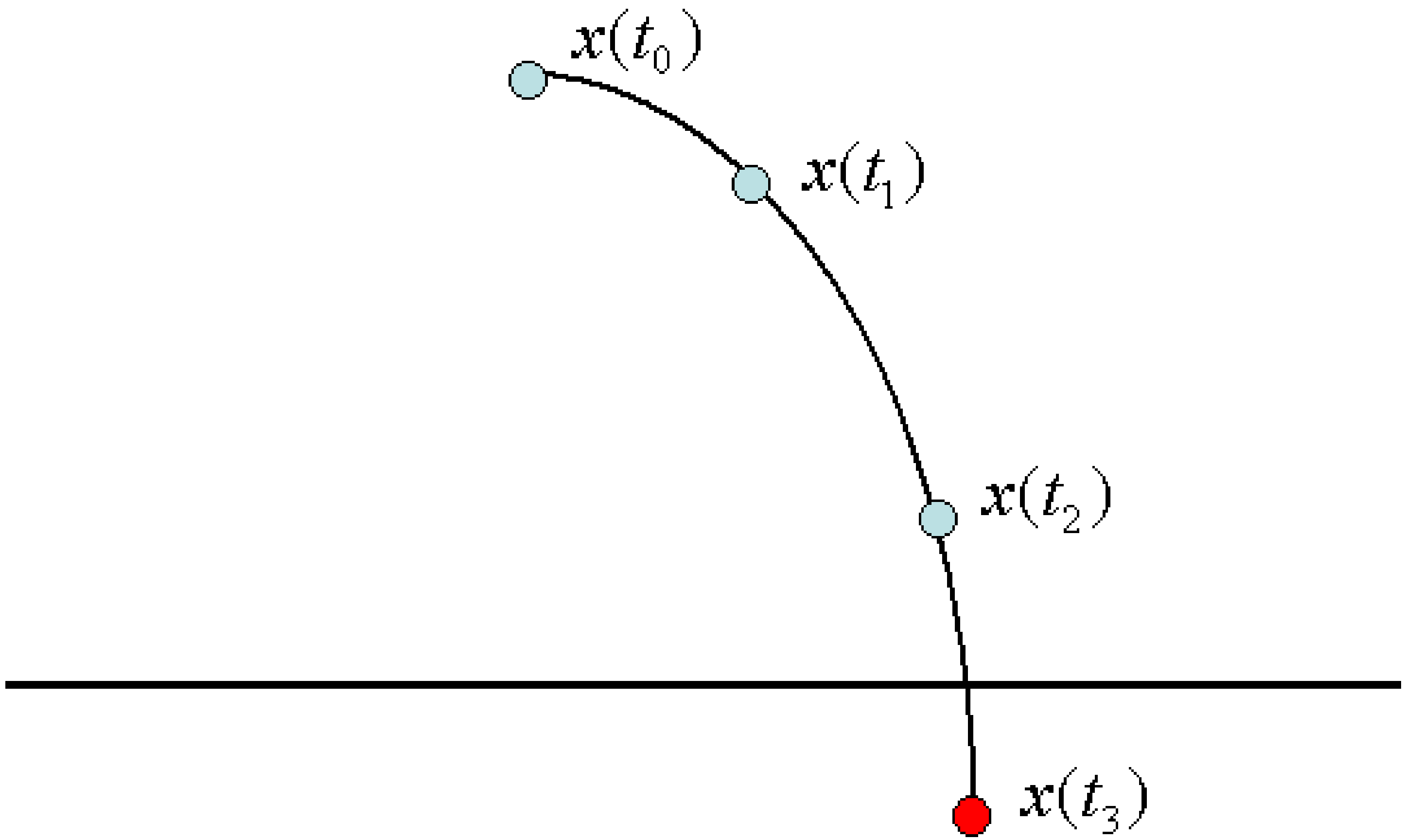
第五章 物理模拟

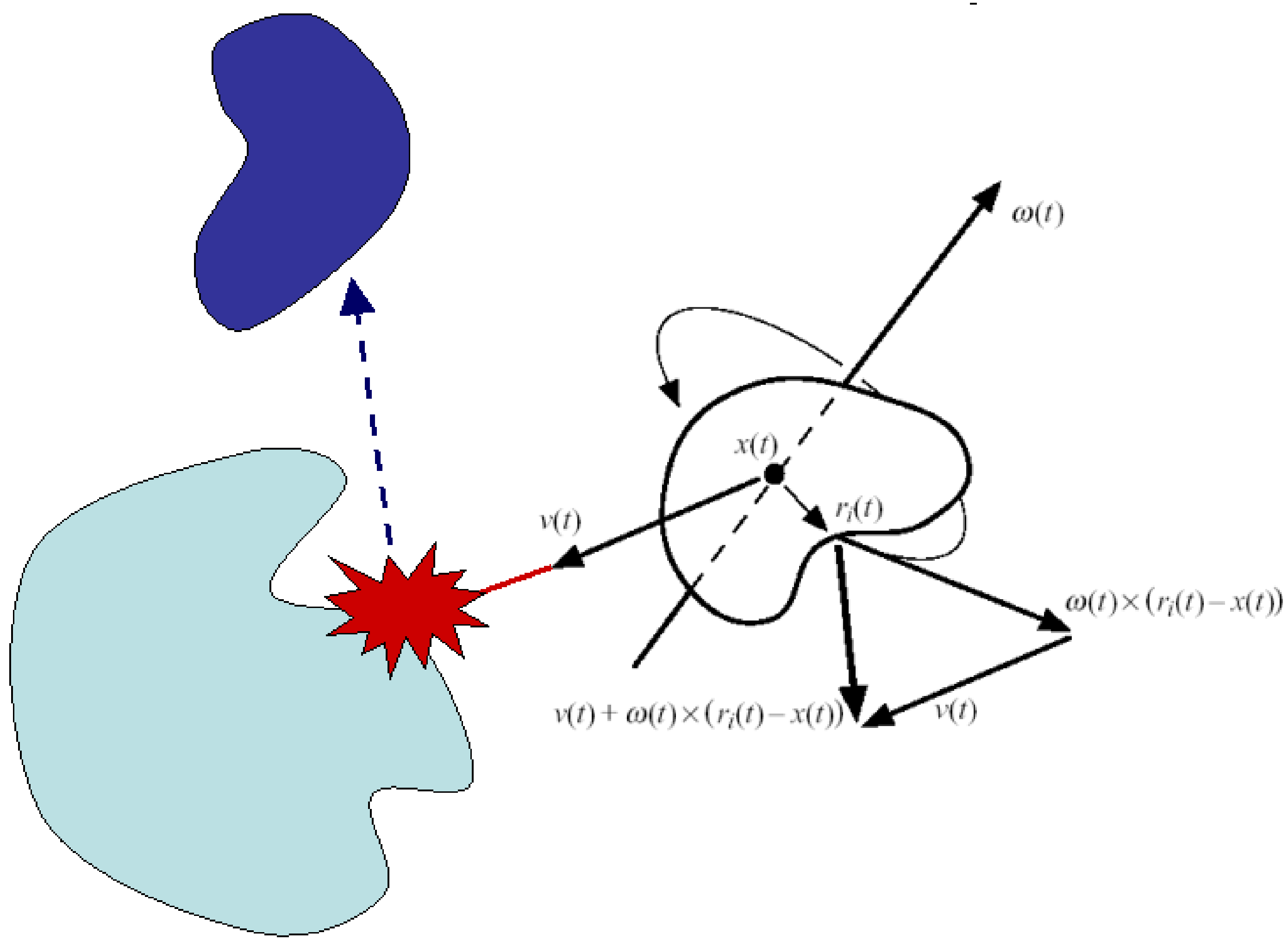
——基于物理模型的动画技术

非贯穿约束动力学

- **Collision detection** –
  determining if, when and where a collision occurs

- **Collision response** –
  calculating the state (velocity, position, …) after the collision


$x(t_0)$
$x(t_1)$
$x(t_2)$
$x(t_3)$



$\omega(t)$
$x(t)$
$r_i(t)$
$v(t)$
$\omega(t)\times(r_i(t)-x(t))$
$v(t)+\omega(t)\times(r_i(t)-x(t))$
$v(t)$

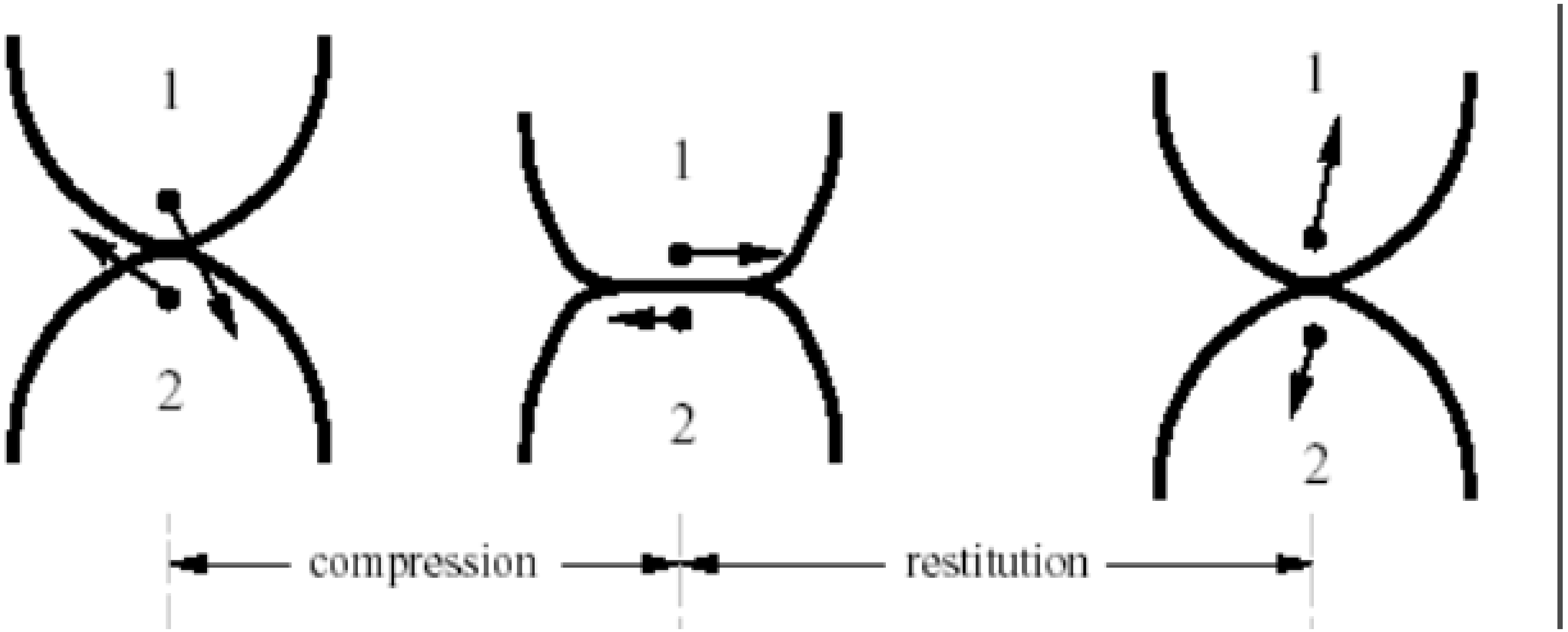
贯穿约束

- 质点掉到地板的过程
  - 质点撞向地板的一瞬间，不允许此质点穿过地板，而且地板在撞击点附近也不发生形变，因此，在碰撞发生的瞬时，粒子与地板发生了接触，导致粒子的速度发生突变

挤压和还原

- 刚体碰撞过程包含两个阶段: 挤压和还原
- 在挤压阶段，两个物体的运动能量转化为变形能量
  - 如果碰撞是完全非弹性的，所有运动能量会损失，因此碰撞后在碰撞面法向上没有相对运动
  - 如果碰撞是完全弹性的，所有的变形能量会在还原阶段转换回运动能量，物体沿碰撞面法向上的运动速度与碰撞前大小相等,方向相反

挤压和还原



第五章 物理模拟

——基于物理模型的动画技术

刚体动力学
中的接触

碰撞接触

支撑接触

碰撞接触(colliding contact)

1. 两个刚体以相向速度在一些点处发生接触
2. 碰撞接触需要瞬时改变相碰的速度
3. 相碰刚体的状态向量在相碰时刻发生速度的不连续变化
4. 碰撞前后分别对应不同微分方程

支撑接触(resting contact)

- 一个刚体在一些接触点处受另一刚体的支撑
- 刚体相碰产生相互作用力称为接触力 **(contact force)**

第五章 物理模拟

——基于物理模型的动画技术

- 碰撞问题可以转化为三个问题:

1. 计算刚体间的碰撞时刻 t_c 和接触点 P
2. 计算碰撞接触时的速度变化
3. 计算防止相互贯穿的接触力

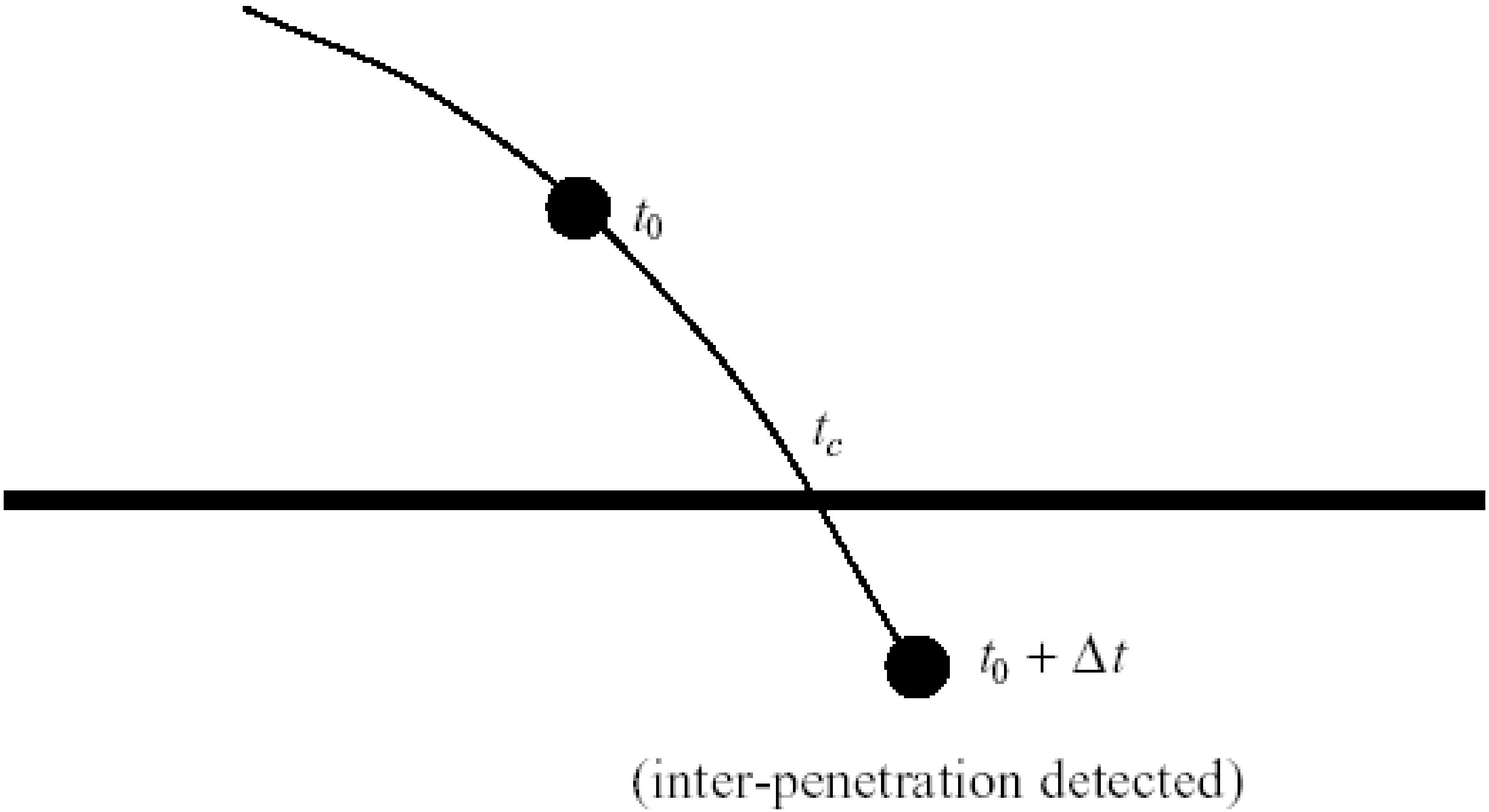
碰撞检测与接触点的计算

- 纯几何问题, 与力学理论无关
- 需要计算刚体相碰的时间 t_c 以及两碰撞刚体的所有接触点

相碰时间 t_c 的计算

- 以一定的时间间隔计算采样时刻 $t_i=t_0+i*\triangle t$ 刚体**A**的位置, 并在每一时刻实施它与其他刚体的碰撞检测.
- 若该刚体在 t_i 时刻与刚体**B**不相碰, 而在 t_{i+1} 时刻两者相贯穿，说明这两个刚体在 $t_c\in(t_i, t_{i+1})$时刻相碰.
- 确定了 $t_c\in(t_i, t_{i+1})$ 后，在一定的误差范围内,可以采用二分法确定 t_c 的具体值.

相碰时间 t_c 的计算



碰撞检测

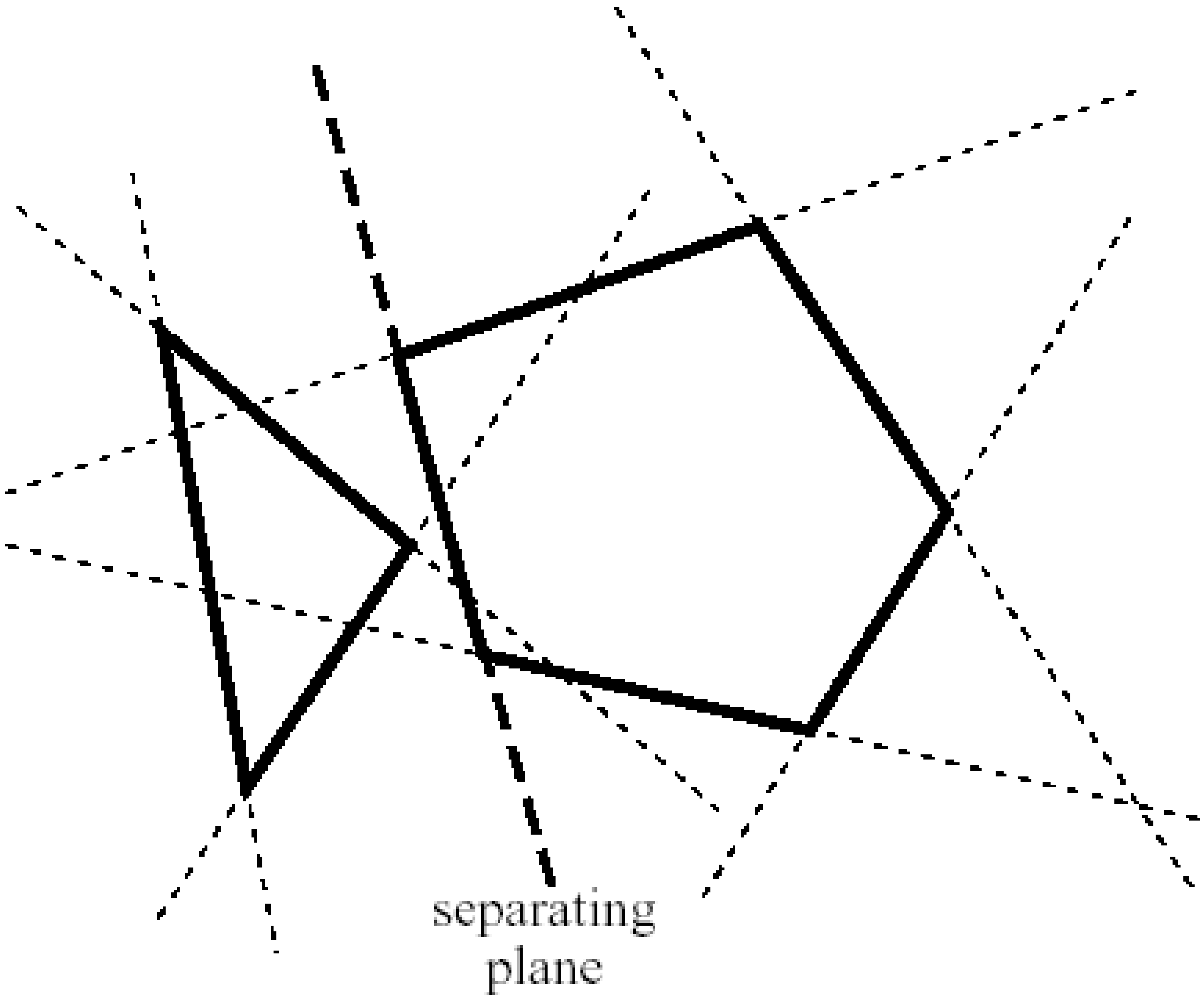
- 为提高碰撞检测效率，在预处理阶段, 对每个物体建立长方体包围盒.
- 碰撞检测问题转换为
  - 给定 n 个刚体的包围盒，快速决定所有相互重叠的长方体包围盒对.
  - 包围盒不相重叠的刚体肯定不相碰;
  - 相重叠的需要进一步处理.

凸多面体的贯穿检测和接触计算

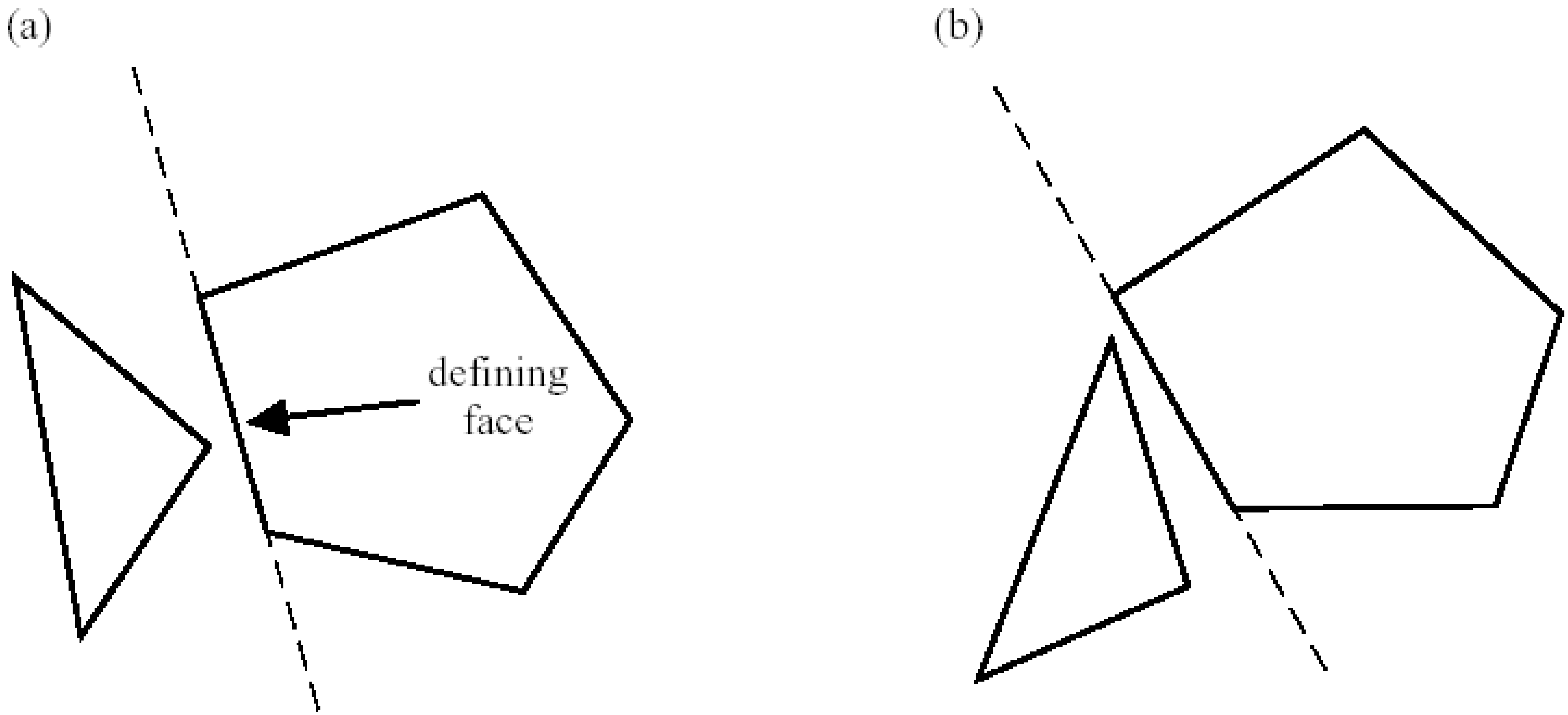
- 两个凸多面体不互相贯穿的充分必要条件:
  - 两个凸多面体之间存在一个分离平面, 使得两个凸多面体分别位于该分离平面的两侧
    - 如果多面体 A 和 B 的所有顶点都位于一个平面的两侧, 则该平面即为 A 和 B 的分离平面
    - 如果两个多面体之间不存在分离平面, 它们肯定相互贯穿

凸多面体的贯穿检测和接触计算

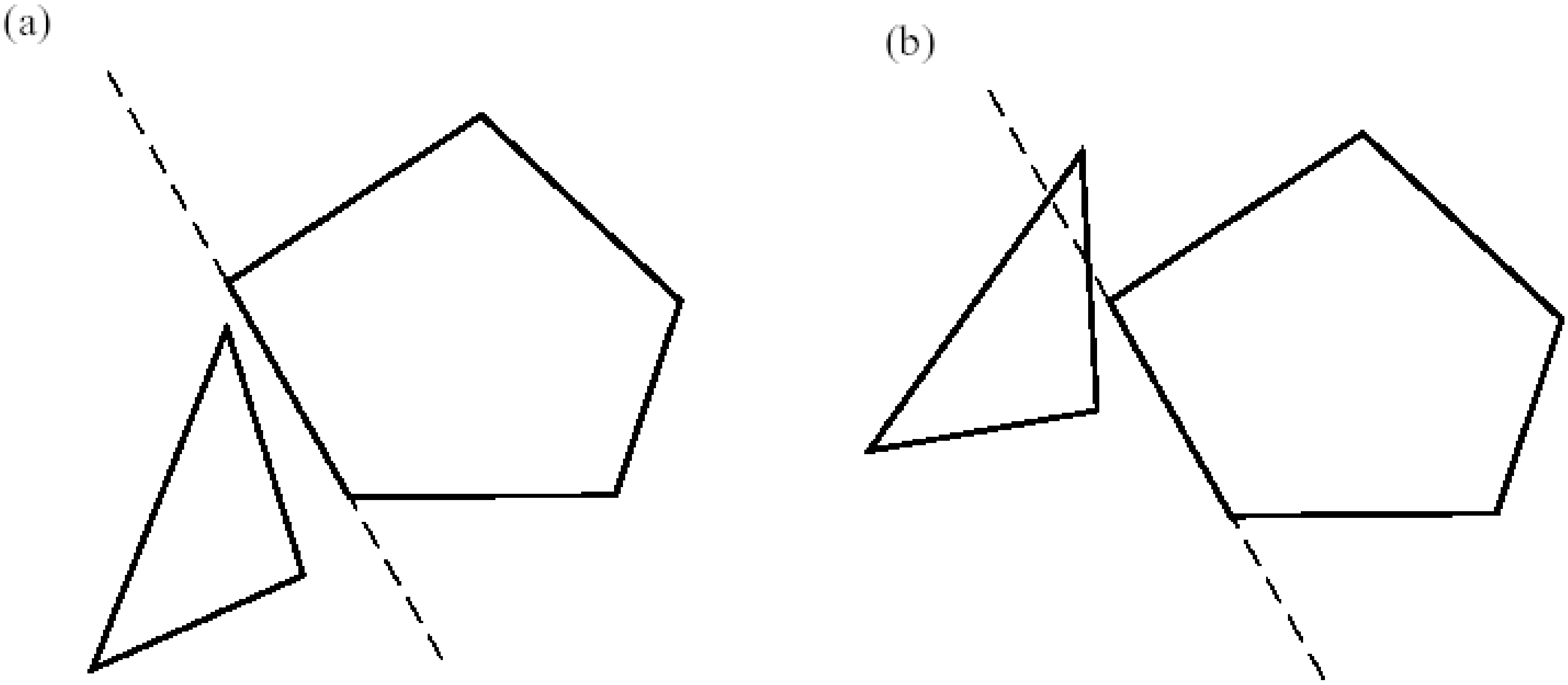
- 如果一对凸多面体是分离的, 下述条件满足时, 分离平面一定存在！
  - 分离平面包含某个多面体的一个面(定义面)
  - 分离平面包含某个多面体的一条边且平行于另一多面体的一条边, 即该分离平面的法向是两条边所形成的向量的叉积(定义边)
  - 可以采用穷举法检测所有可能的面和边的组合是否可以形成一个分离平面



穷举搜索分离面

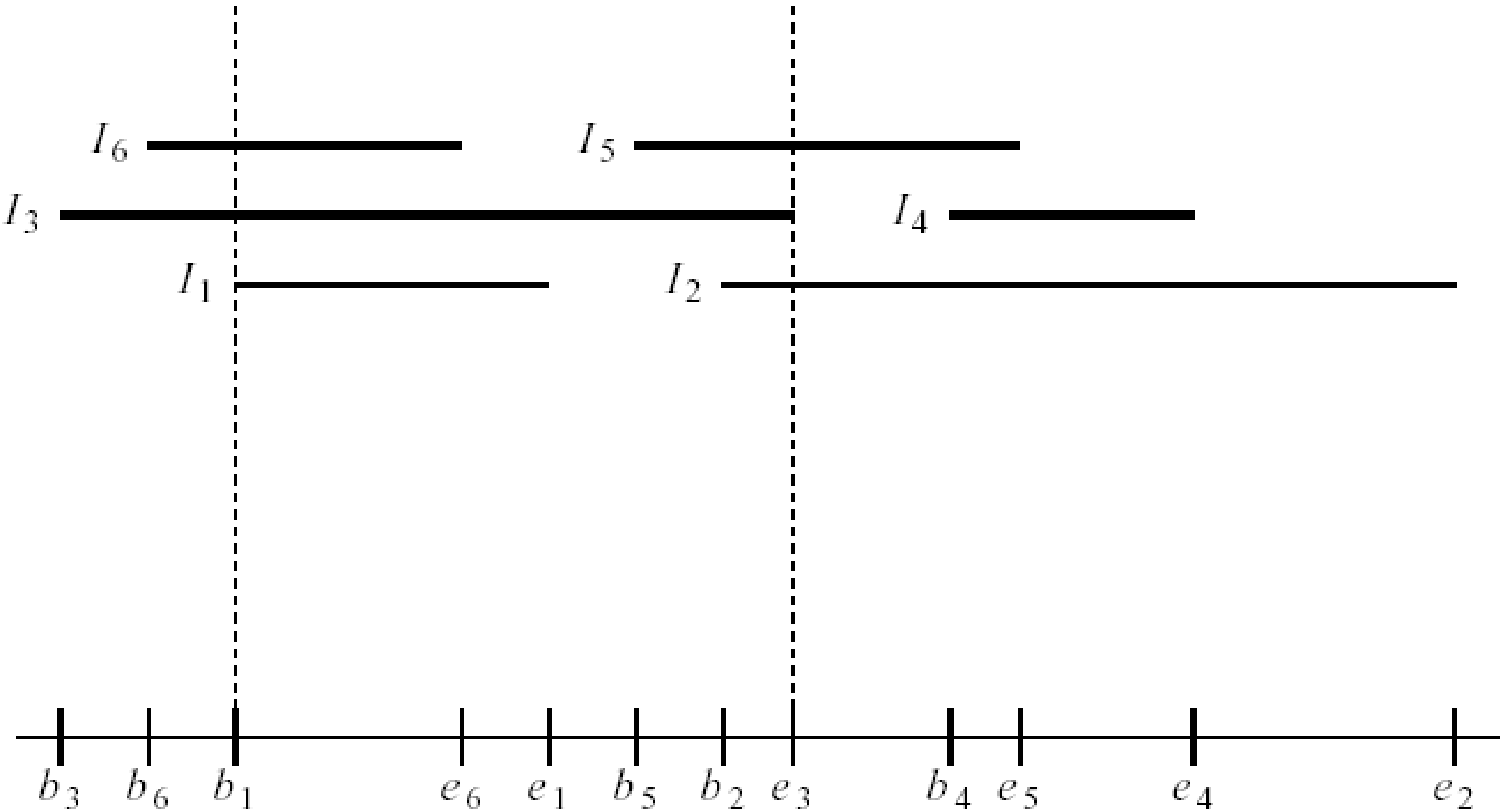


A 时刻的分离平面在 b 时刻仍然有效



A 时刻的分离平面在 b 时刻不再有效，
需要重新计算一个新的分离平面

包围盒的重叠检测算法

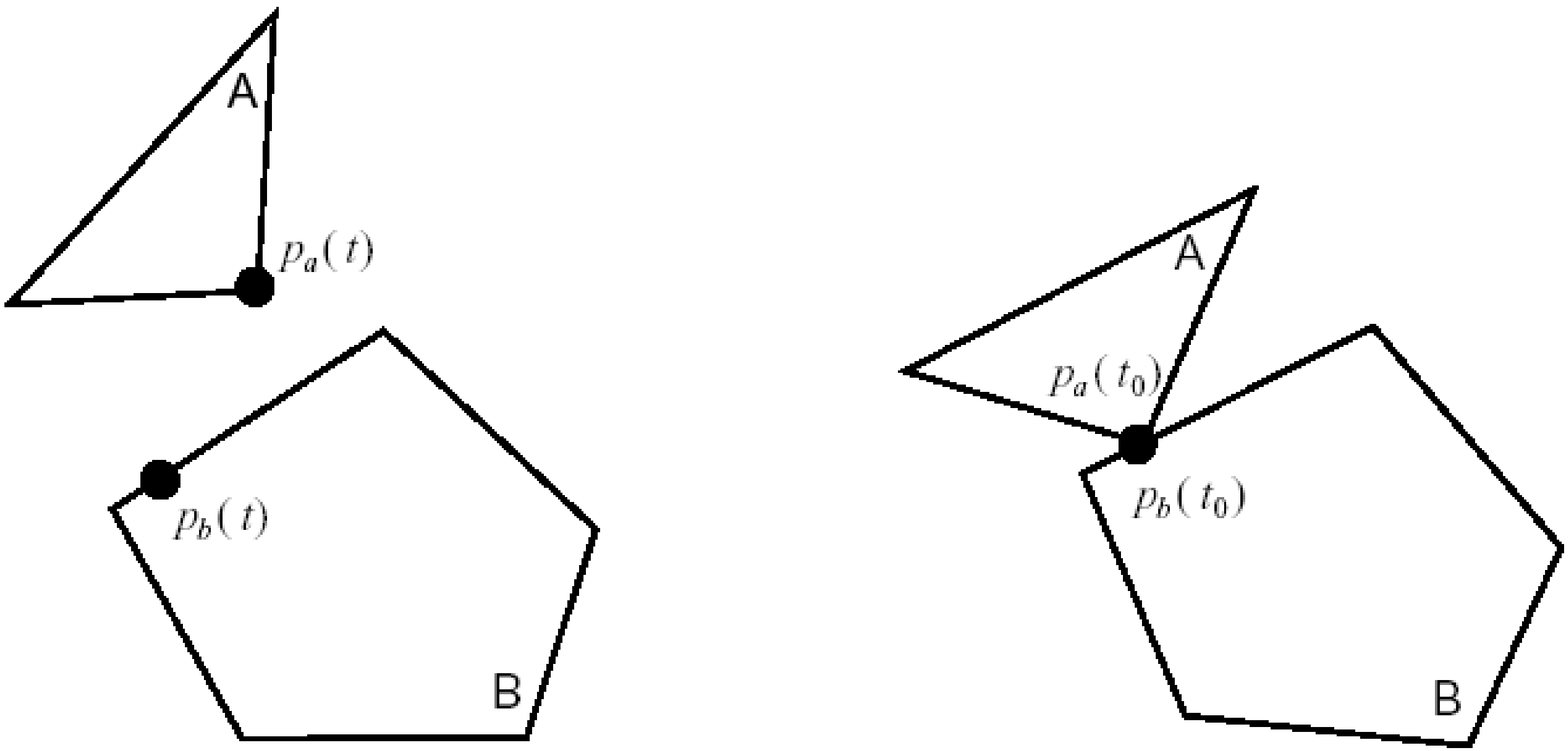


排序----扫描算法（一维情况）

碰撞接触后的速度变化

- 检测某一特定时刻 t_o 物体间的碰撞
- 假设在 t_o 时刻没有物体是相互贯穿的，并且系统已经确定了所有相接触的物体及其接触点
- 多面体间的接触
  - 顶点----面接触
  - 边　----边接触
  - 顶点----顶点接触和顶点----边接触是退化情形，不考虑

碰撞接触后的速度变化



顶点----面接触　$P_a(t_0) = P_b(t_0) = P$

碰撞接触后的速度变化

- 尽管 $P_a(t)$ 和 $P_b(t)$ 在时刻 t_o 重合，但它们的速度可能相差很大
- 可以通过检测速度来判断是否发生了碰撞

$$\dot{p}_a(t_0) = v_a(t_0) + \varpi_a(t_0) \times (p_a(t_0) - x_a(t_0))$$

$$\dot{p}_b(t_0) = \underbrace{v_b(t_0)}_{\text{线速度}} + \underbrace{\varpi_b}_{\text{角速度}}(t_0) \times (p_b(t_0) - x_b(t_0))$$

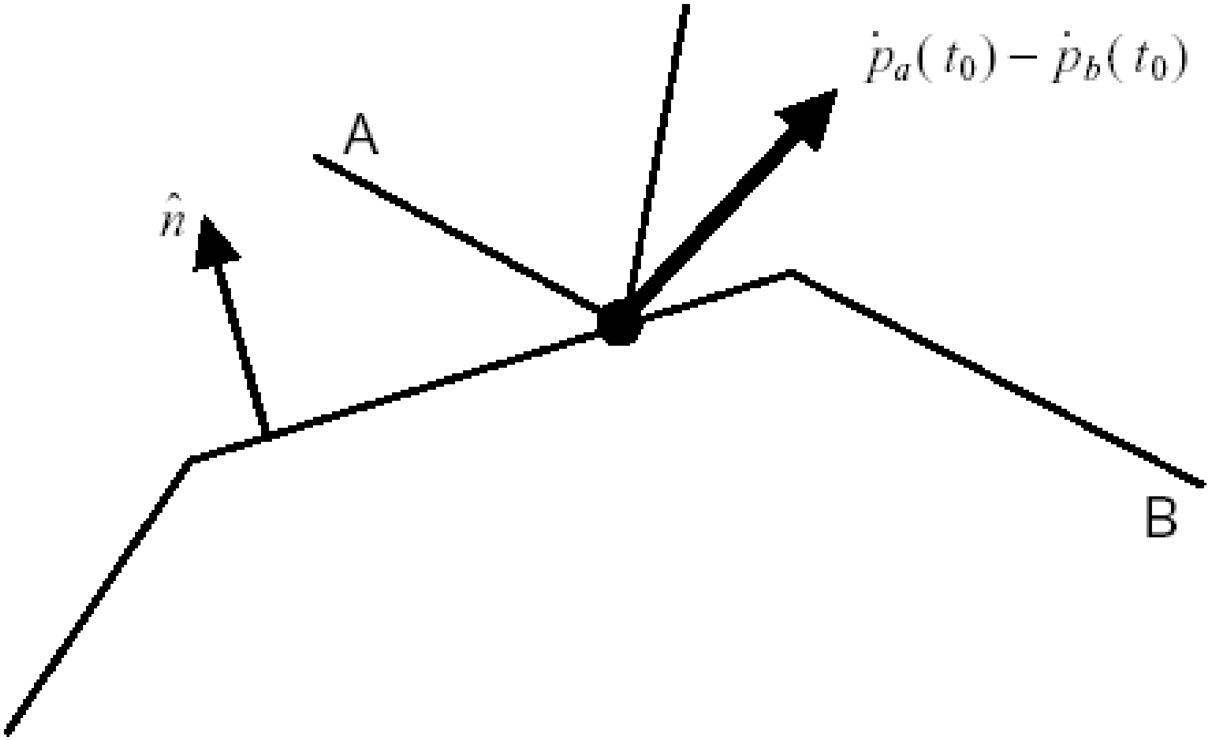
碰撞接触后的速度变化

$$v_{rel} = n(t_0) \bullet (\dot{p}_a(t_0) - \dot{p}_b(t_0))$$

相对速度在 ***n*(*t*$_0$)** 方向的投影

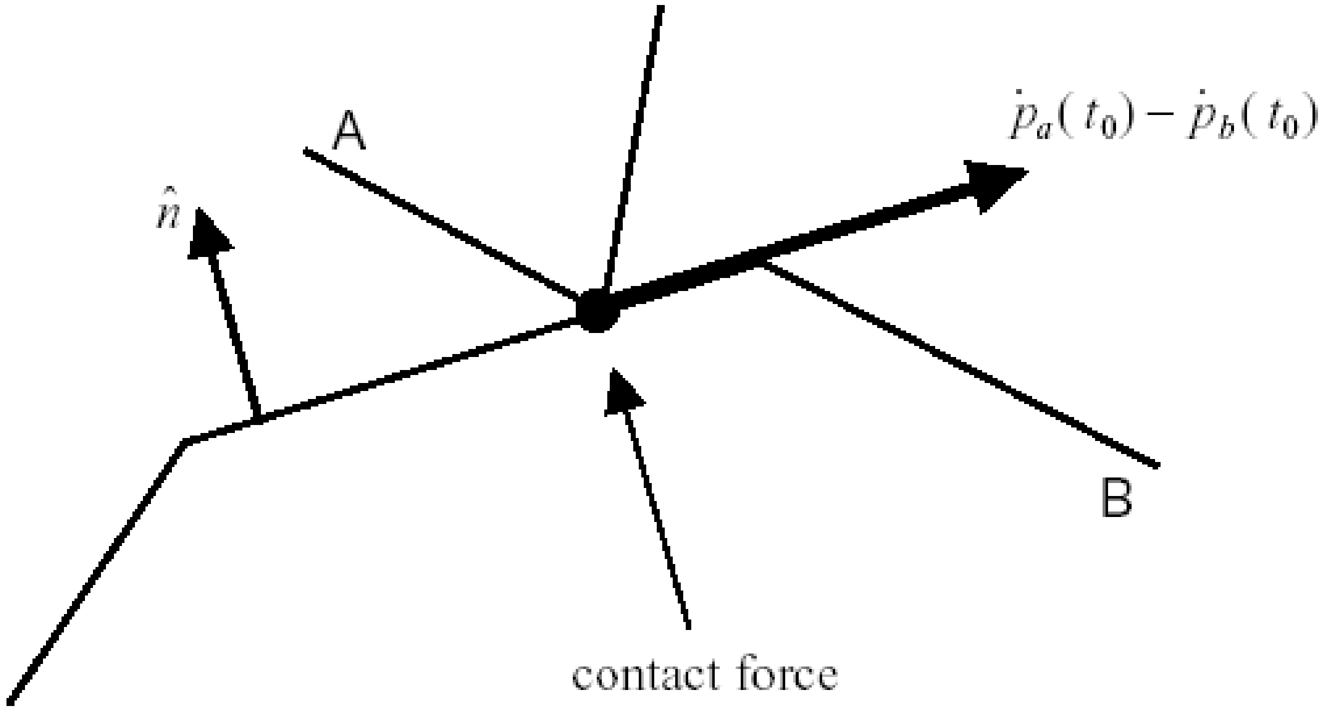
物体 **B** 在接触点处的单位法向方向

接触点消失的情况



如果 v_{rel} 是正值，物体将马上分离，接触点也将在 t_o 时刻后消失. 此时无须任何处理

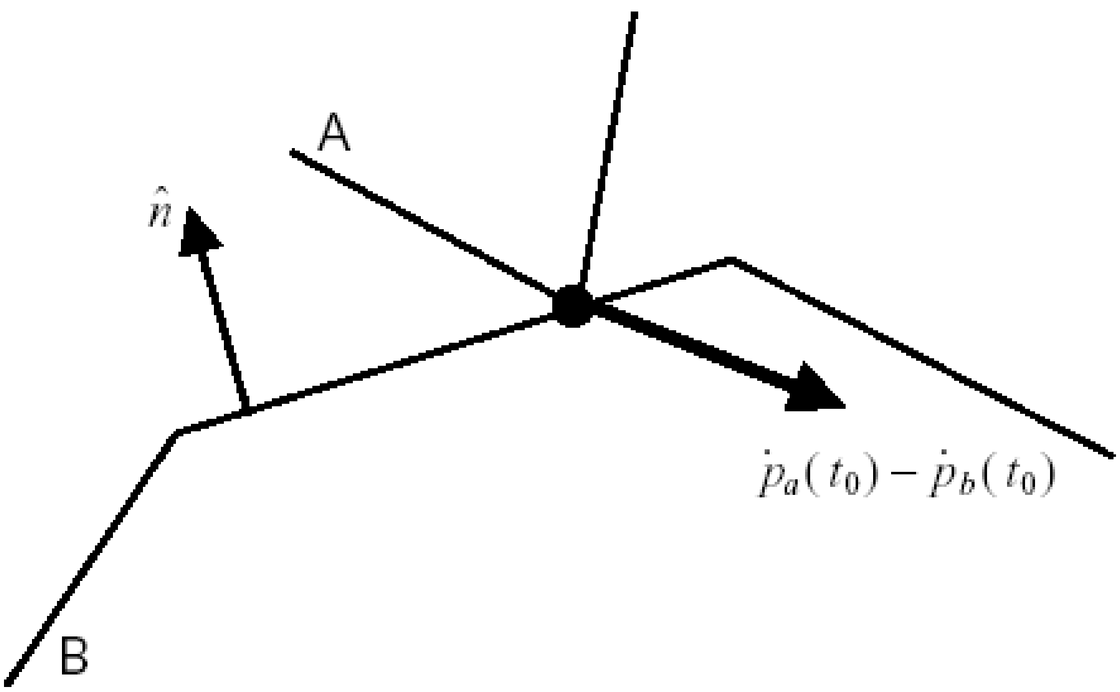
接触点继续保持的情况



如果 v_{rel} 为零，物体在接触点既不前进也不后退，两物体处于支撑接触状态，此时需要计算接触力以避免两个物体向对方方向加速

碰撞接触的情况

如果 v_{rel} 小于零，物体在接触点处的相对速度与法向方向相反且有碰撞接触，如果刚体速度不骤变，将会发生贯穿



计算速度的变化

- 由于惯性的作用，无论作用于接触点处的力有多大，都需要少量的时间来完全停止刚体间的相对运动
- 使用力学中的冲量 *J* 来刻画刚体速度的突变
  - 冲量是一个矢量，它产生物体速度的瞬时改变
  - 一个巨大的力 *F* 在一个很短时间内作用于刚体，产生的冲量为 *J*
  - *J* 等于线动量的变化量

$$F * \Delta t = J$$

$$\Delta v = \frac{J}{m}$$

线速度的变化量

计算速度的变化

- 如果冲量的作用点是 ***P***，冲量 ***J*** 会产生一个冲量力矩

$$\tau_{impulse} = (p - x(t)) \times J$$

在 ***t*** 时刻刚体的中心位置

计算速度的变化

冲量力矩会产生一个角动量的变化△***L***

$$\triangle L = \tau_{impulse}$$

角速度的改变量为

$$I^{-1}(t_0)\tau_{impulse}$$

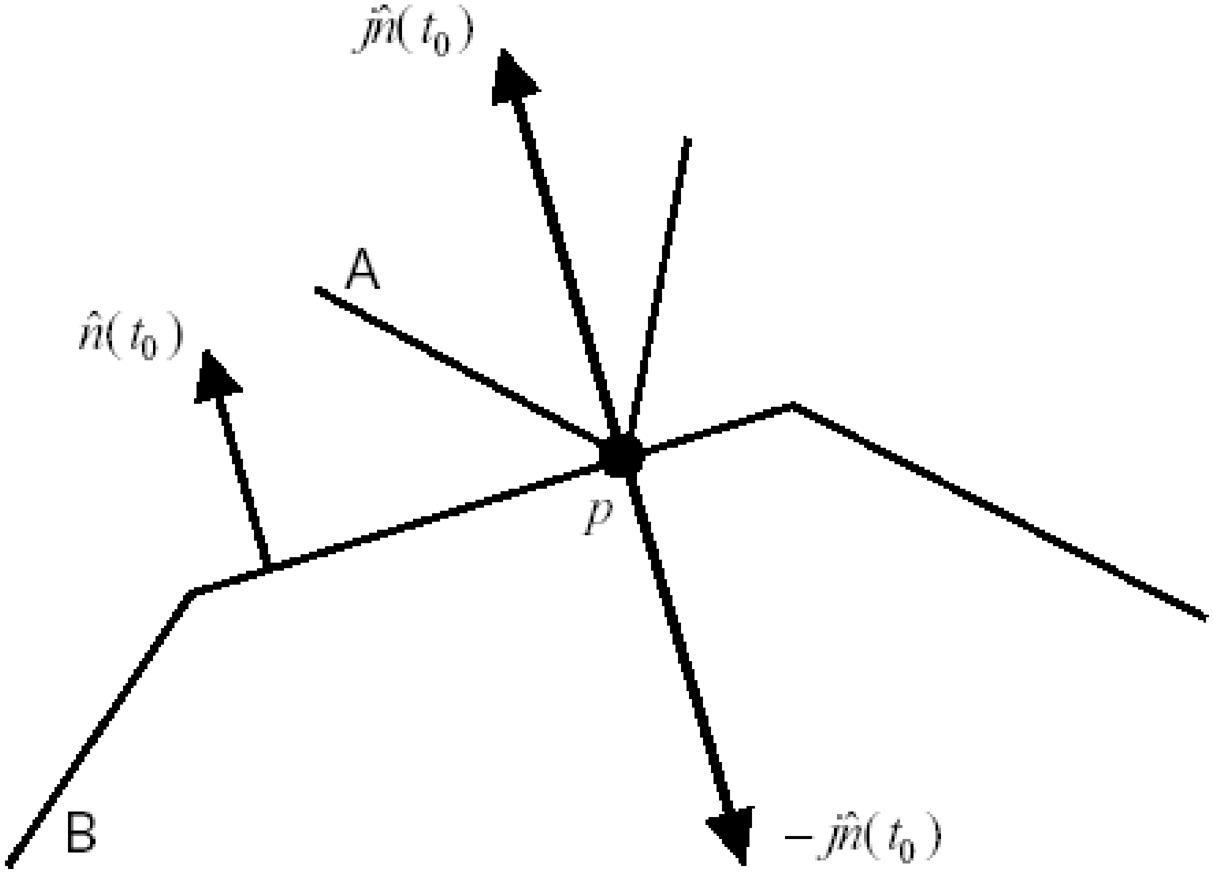
计算速度的变化

- 两个物体相碰时, 可以用冲量来实现速度的改变
- 对于无摩擦的物体, 冲量的方向就是法向

$$J = j * n(t_0)$$

j 是待确定的冲量大小

接触点处的冲量



A 得到的冲量为 $+jn(t_0)$，B 得到的冲量为 $-jn(t_o)$

瞬时相对速度

- 可以用一个碰撞的经验定律计算冲量的大小 j
- $P_a^-(t_0)$ 和 $P_a^+(t_0)$ 分别表示冲量起作用前后物体 A 上接触点的速度
- 碰撞发生前沿法向 $n(t_0)$ 的瞬时相对速度

$$v_{rel}^- = n(t_0).(P_a^-(t_0) - P_b^-(t_0))$$

- 碰撞发生后沿法向$n(t_0)$的瞬时相对速度

$$v_{rel}^+ = n(t_0).(P_a^+(t_0) - P_b^+(t_0))$$

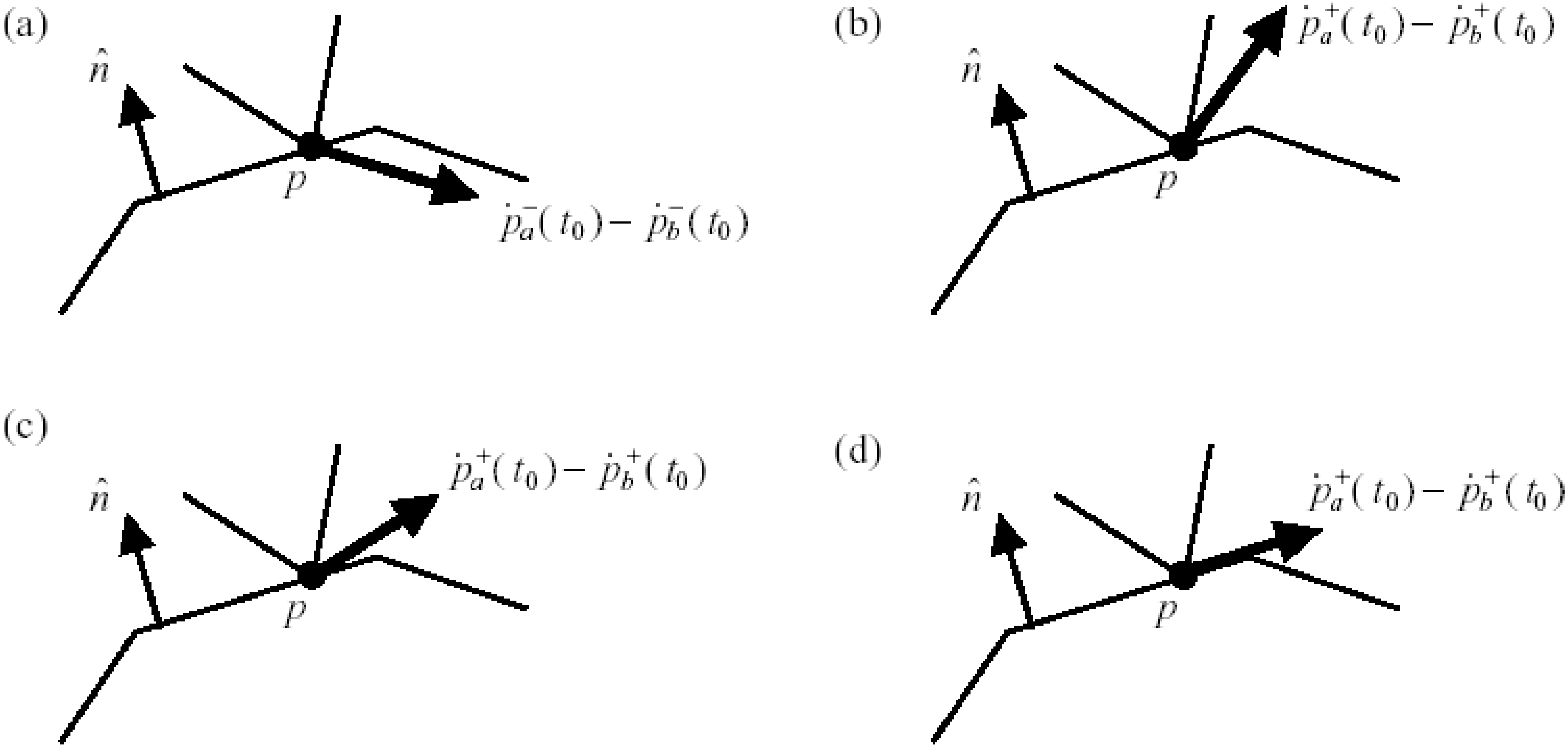
碰撞前后的相对速度

- 无摩擦碰撞的经验定律
  - 描述了冲量强度, 相对速度和还原系数之间的关系

$$v_{rel}^{+} = \varepsilon * v_{rel}^{-}$$

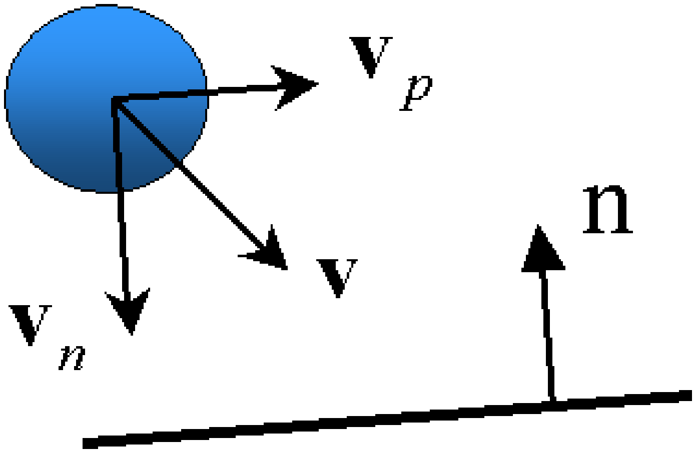
-
  - $0 \le \varepsilon \le 1$
  - 若 $\varepsilon=1$, 碰撞是完全弹性的, 即碰撞过程中能量守恒
  - 若 $\varepsilon=0$, 碰撞是完全非弹性的, 动能最大损失, 接触点为支撑接触

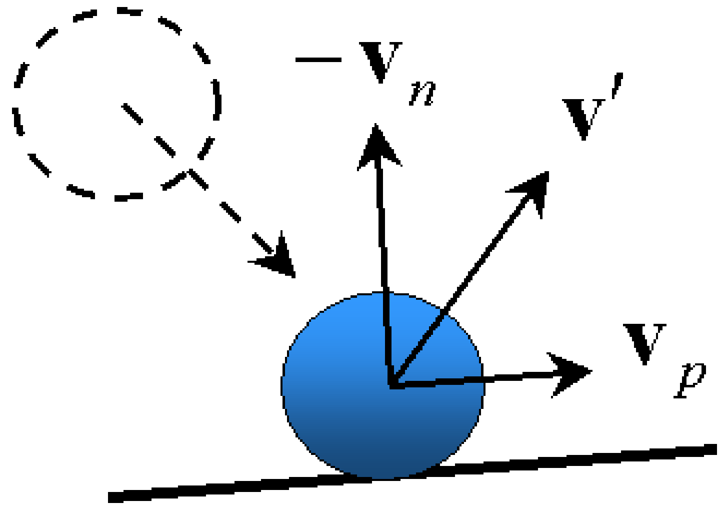
碰撞前后的相对速度



a) 冲量作用前的相对速度 **b) ε=1的碰撞**

c) 0<ε<1的碰撞 **d) ε=0的碰撞**





$$\mathbf{v} = \mathbf{v}_n + \mathbf{v}_p$$

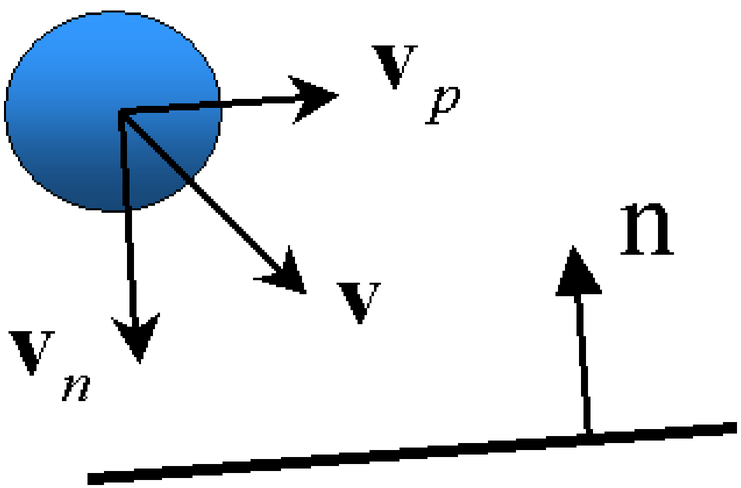
$$\mathbf{v}_n = (\mathbf{v}.\mathrm{n})\mathrm{n}$$

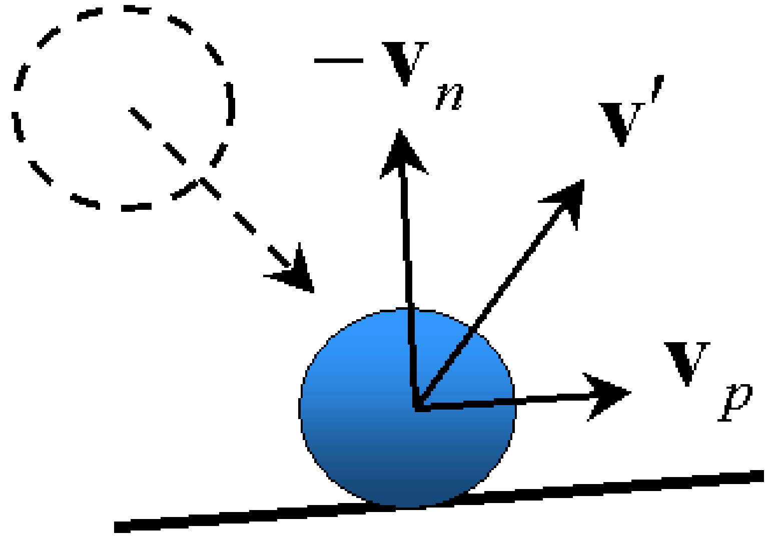
$$\mathbf{v}_p = \mathbf{v} - \mathbf{v}_n$$

$$\mathbf{v}' = \mathbf{v}_p - \mathbf{v}_n$$

What kind of response?

Totally elastic!
No kinetic energy is lost →
response is "perfectly bouncy"





$$\mathbf{v} = \mathbf{v}_n + \mathbf{v}_p$$

$$\mathbf{v}_n = (\mathbf{v}.\mathrm{n})\mathrm{n}$$

$$\mathbf{v}_p = \mathbf{v} - \mathbf{v}_n$$

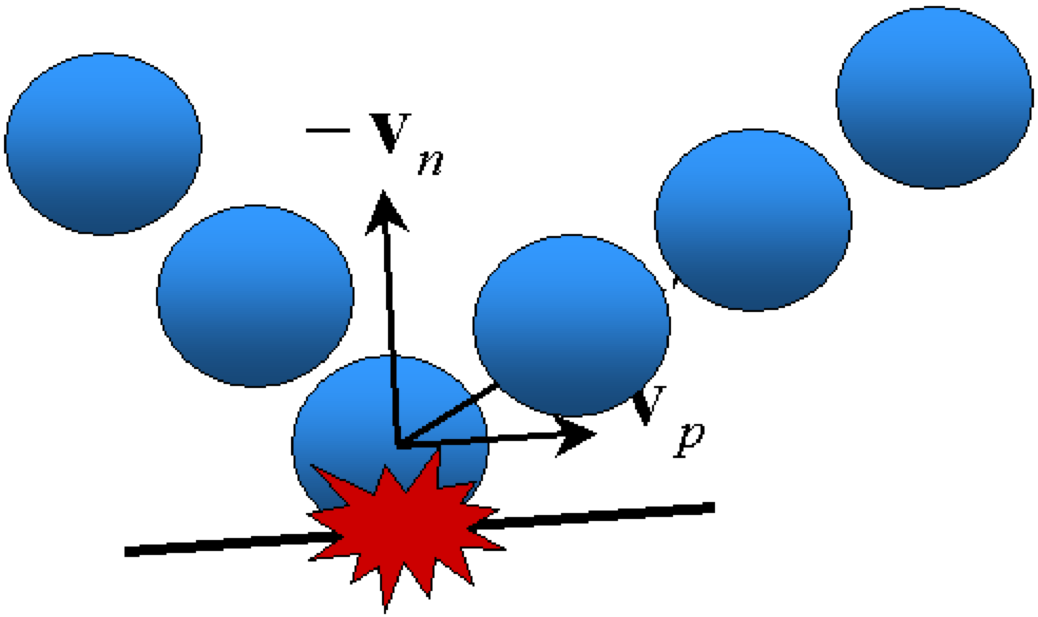
$$\boxed{\mathbf{v}' = \mathbf{v}_p - k\mathbf{v}_n}$$

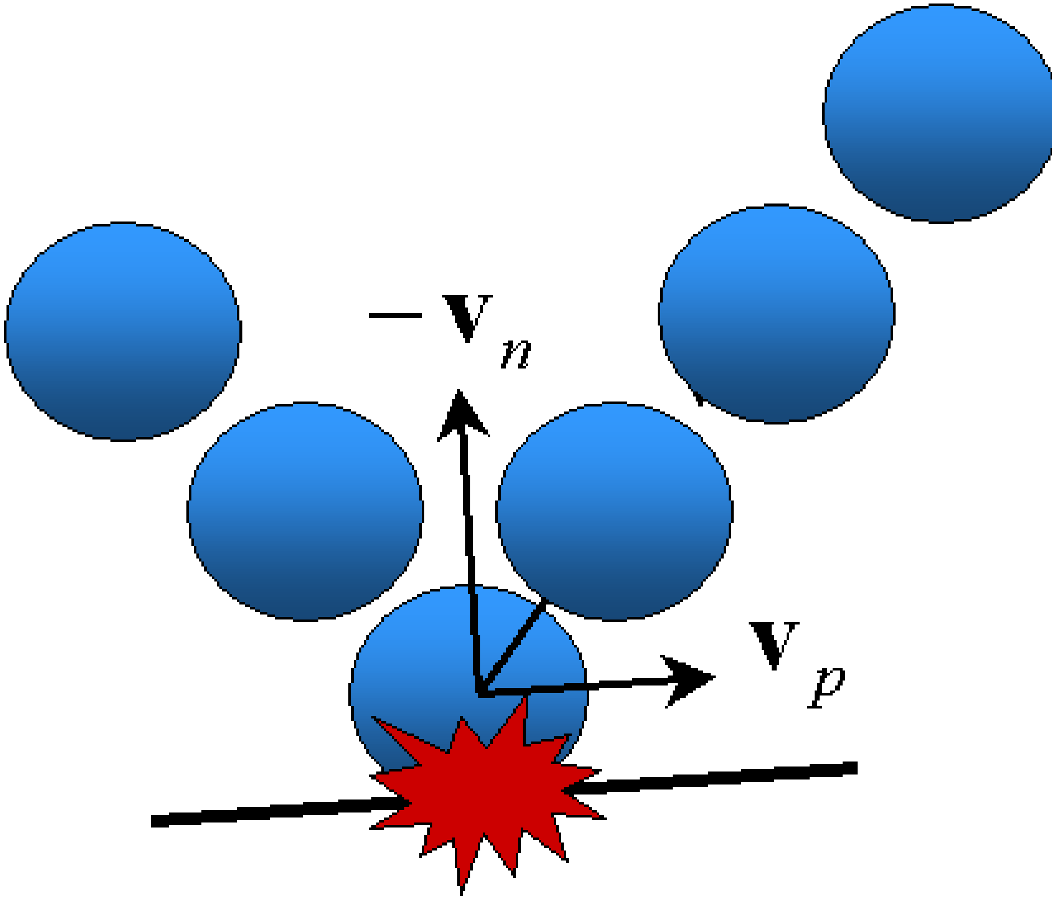
k, in [0,1] → coefficient of *restitution*

Using Coefficient of Restitution:

As k gets smaller → more and more energy is lost → less and less bouncy

$$\mathbf{v}' = \mathbf{v}_p - 0.5\mathbf{v}_n$$



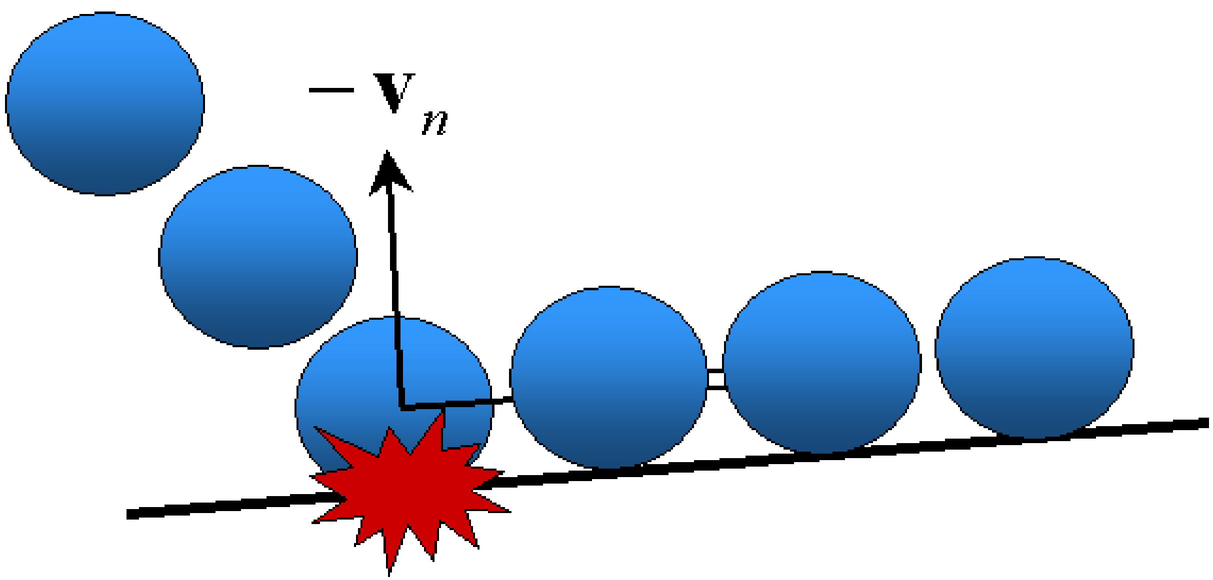


$$\mathbf{v}' = \mathbf{v}_p - \mathbf{v}_n$$

Using Coefficient of Restitution:

As k gets smaller → more and more energy is lost → less and less bouncy

$$\mathbf{v}' = \mathbf{v}_p - 0.0\mathbf{v}_n$$



冲量大小的计算

$$j = \frac{-(1+\epsilon)v_{rel}^-}{\frac{1}{M_a} + \frac{1}{M_b} + \hat{n}(t_0)\cdot\left(I_a^{-1}(t_0)\left(r_a \times \hat{n}(t_0)\right)\right)\times r_a + \hat{n}(t_0)\cdot\left(I_b^{-1}(t_0)\left(r_b \times \hat{n}(t_0)\right)\right)\times r_b}$$

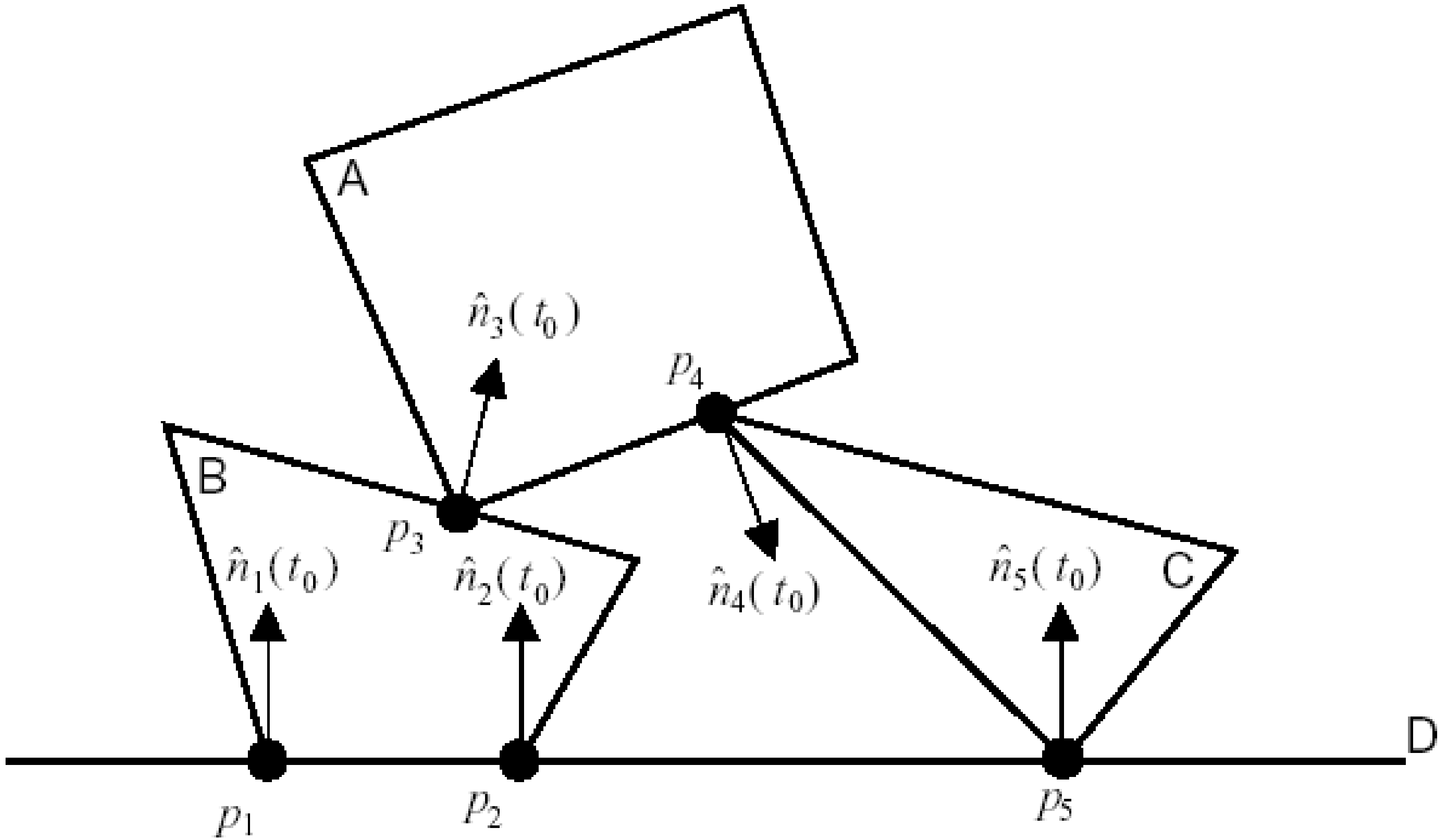
- M_a: 刚体 A 的质量
- $I_a(t_0)$: A的惯性张量矩阵
- 位移 r_a 和 r_b 分别为 $P^- x_a(t_0)$ 和 $P^- x_b(t_0)$
- 上标-和+分别表示碰撞前后的状态量
- ε: 还原系数

第五章 物理模拟

——基于物理模型的动画技术

支撑接触

- 支撑接触时, 接触点处的相对速度为零
- 支撑接触时, 每个接触点处都有一个压力 $f_i * n(t_0)$, 需要计算所有的 f_i
- 由于第 i 个接触点处的压力有可能影响到第 j 个接触点处的刚体, 因此所有的 f_i 必须同时求得



支撑接触及接触点处的法向

支撑接触

- 计算 f_i 需要满足的三个条件：
  - 接触压力必须强到能够阻止两个接触的刚体向对方挤压，从而防止相互贯穿
  - 接触压力必须是相互排斥的，以分离刚体
  - 如果两个刚体开始分离，则接触点处的压力必须为零

距离函数

- 描述 t 时刻两刚体在接触点附近的距离函数 $d_i(t)$

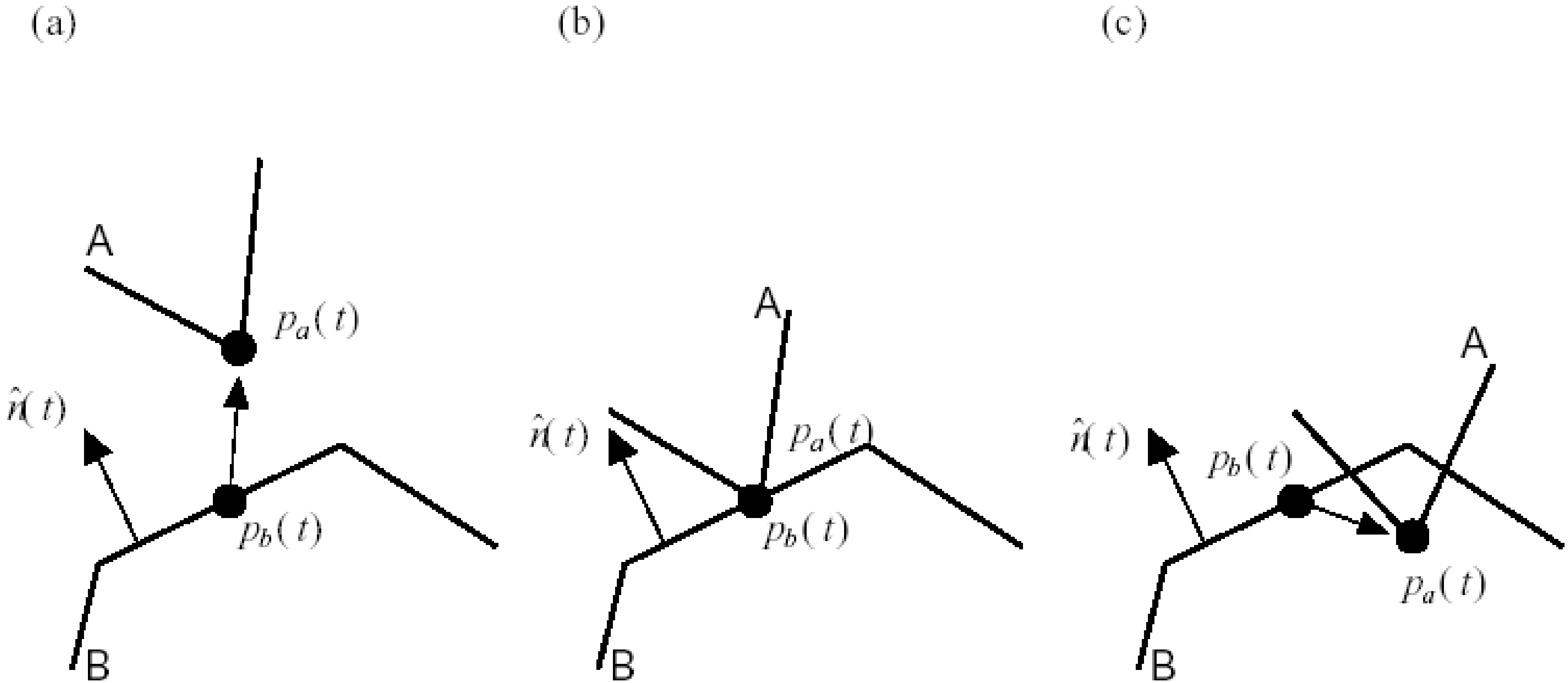
$$d_i(t) = n_i(t).(P_a(t) - P_b(t))$$

若$d_i(t)>0$ 两物体分离

若$d_i(t)=0$ 两物体接触

若$d_i(t)<0$ 两物体相互贯穿

- 为防止贯穿，需要保证对每个接触点，在 $t>t_0$ 时，$d_i(t)>0$



a) $d_i(t) > 0$ **b) $d_i(t) = 0$** **c) $d_i(t) < 0$**

思考题：

1. 粒子运动方程的一般形式是什么？各变量表示什么含义？
2. 一阶常微分方程有哪两种常用的求解方法？其中显式求解有主要有几种方法，各有哪些优缺点？
3. 碰撞问题可以转化为哪三个问题？
4. 碰撞检测可以转化为哪几过程？加速策略有哪些？

结束