

情報メディア実験A

物理エンジンを使った アプリケーション開発

筑波大学情報学群
情報メディア創成学類
藤澤誠

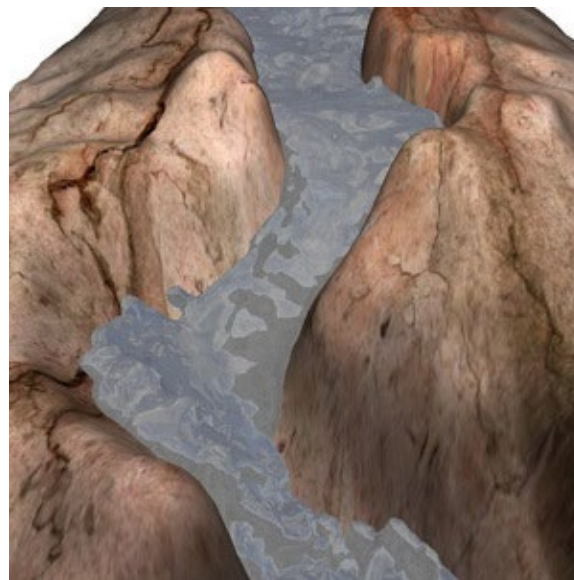
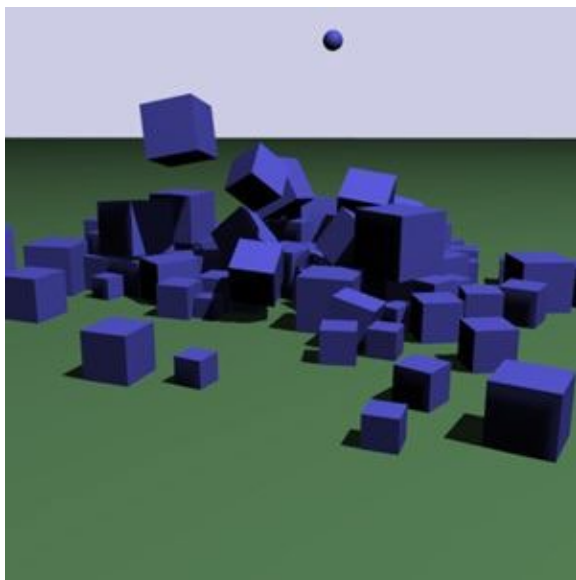
実験の目的

物理シミュレーションエンジンを利用したアプリケーションが作れるようになる

- 教員による説明回+講義ページの説明を理解して練習問題を解く(実装する)
- 物理エンジンを用いた簡単なアプリケーションを作成

物理シミュレーションとは？

- 物体の物理運動をリアルに再現するための手法
→ CG作成によく使われる
 - 質量・速度・摩擦・風といったものを
物理法則に基づきシミュレーション

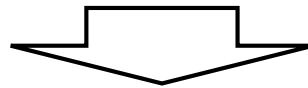


物理エンジンで使う数学

- ニュートンの運動方程式

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

$$\text{力[N]} = \text{質量[kg]} \times \text{加速度[m/s}^2\text{]}$$



- 物体に力を加えたときにどれぐらい加速するかが分かる
- 物体を描画するときに必要なのは位置 x

物理エンジンで使う数学

- ニュートンの運動方程式

$$\mathbf{F} = m\ddot{\mathbf{x}}$$

$\ddot{\mathbf{x}}$: \mathbf{x} の時間2階微分($\frac{d^2\mathbf{x}}{dt^2}$)

⇩ 1階微分×2に分解

$$\ddot{\mathbf{x}} = \frac{d\dot{\mathbf{x}}}{dt} = \frac{d\mathbf{v}}{dt},$$

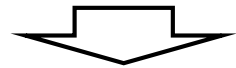
$$\dot{\mathbf{x}} = \mathbf{v} = \frac{d\mathbf{x}}{dt}$$

物理エンジンで使う数学

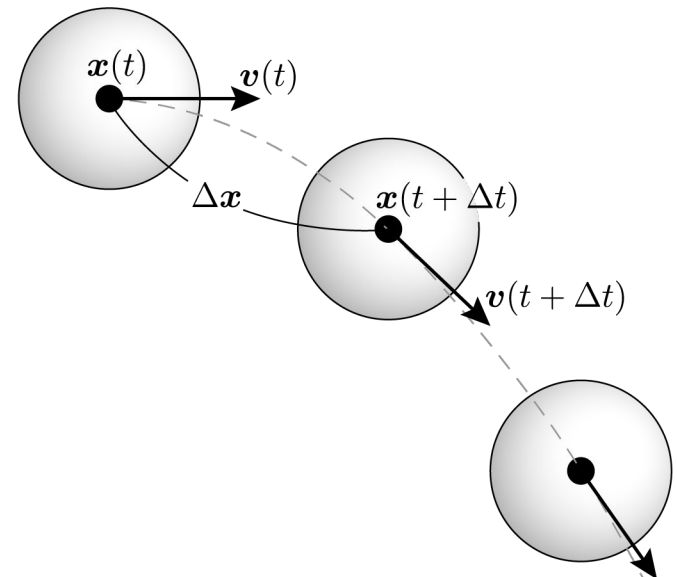
- 時間積分の離散化

$$\mathbf{v}(t) = \frac{d\mathbf{x}}{dt} = \left(\frac{dx}{dt}, \frac{dy}{dt}, \frac{dz}{dt} \right)$$

位置 \mathbf{x} の時間変化



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



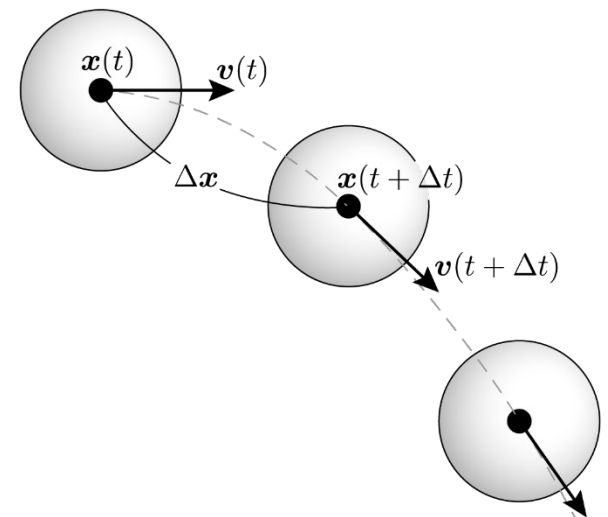
物理エンジンで使う数学

- 時間積分の離散化

$$\Delta x = \int_t^{t+\Delta t} v(t) dt$$

速度が t から $t + \Delta t$ の間一定の値だと仮定

$$x(t + \Delta t) = x(t) + v(t)\Delta t \quad : \text{前進オイラー法}$$



物理エンジンで使う数学

- ここまでのまとめ
 - 物体を動かすためには力 F を与える
 - F から運動方程式で加速度，加速度から前進オイラー法で速度 \Rightarrow 位置と計算
 - 時間ステップ幅 Δt を与える必要がある
 - 精度は Δt が小さいほど良くなる(計算時間は大きくなる)

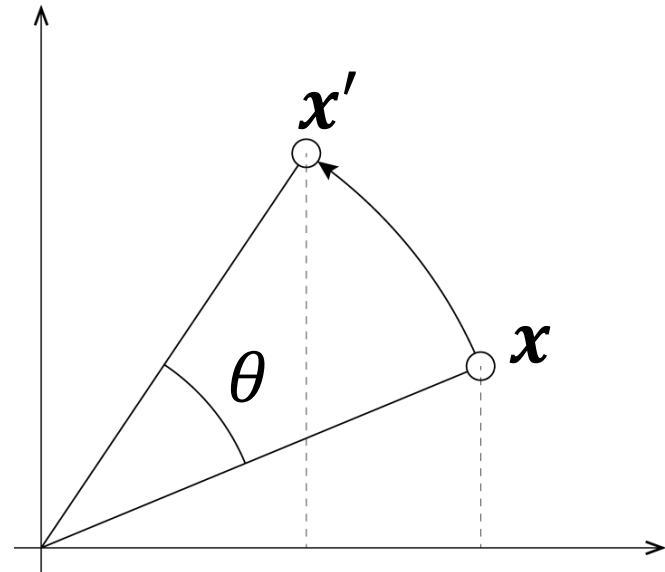
物理エンジンで使う数学

- 行列を使った回転の表現
2次元の回転 \Rightarrow 角度 θ で表現可能

$$R = \begin{pmatrix} \cos \theta & -\sin \theta \\ \sin \theta & \cos \theta \end{pmatrix}$$



$$x' = Rx$$



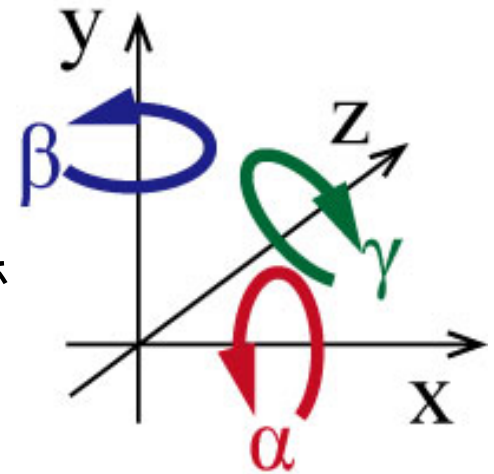
物理エンジンで使う数学

- 行列を使った回転の表現
3次元の回転⇒?

- オイラー角

x, y, z 軸周りの回転角 (α, β, γ)
で回転を表現

- それぞれに回転行列 R_x, R_y, R_z があり, それを合成することで1つの行列で回転を表現



計算が大変, 1つの回転に9つの要素が必要

([ジンバルロック](#)の問題もある)

物理エンジンで使う数学

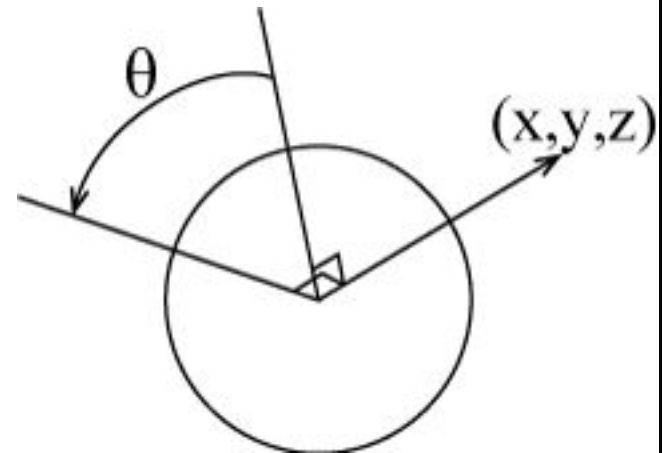
■ 四元数

実数部(スカラー部)1つ,虚数部(ベクトル部)3つを持つ4次元数

$$\mathbf{q} = (\mathbf{v}, s) = i v_x + j v_y + k v_z + s$$

ベクトル部を回転軸, スカラー部を回転角と考えると任意の回転を表現可能

$$\mathbf{q} = \left(\underbrace{x \sin \frac{\theta}{2}, y \sin \frac{\theta}{2}, z \sin \frac{\theta}{2}}_{\text{ベクトル部}}, \underbrace{\cos \frac{\theta}{2}}_{\text{スカラー部}} \right)$$



物理エンジンで使う数学

- 四元数を使った回転の表現

$$\boldsymbol{x}' = \boldsymbol{q} (\boldsymbol{x}, 0) \boldsymbol{q}^{-1}$$

- 四元数で覚えておくこと

- 単位四元数

$$\boldsymbol{q} = (0, 0, 0, 1)$$

- 逆回転を表す四元数(共役四元数)

$$\boldsymbol{q}^{-1} = (-\boldsymbol{v}, s)$$

- 回転行列と相互変換可能

物理エンジンで使う数学

- 四元数 $q = (v_x, v_y, v_z, s)$ から回転行列への変換

$$R = \begin{pmatrix} 1 - 2v_y^2 - 2v_z^2 & 2v_xv_y - 2sv_z & 2v_xv_z + 2sv_y \\ 2v_xv_y + 2sv_z & 1 - 2v_x^2 - 2v_z^2 & 2v_yv_z - 2sv_x \\ 2v_xv_z - 2sv_y & 2v_yv_z + 2sv_x & 1 - 2v_x^2 - 2v_y^2 \end{pmatrix}$$

- ほとんどの物理エンジンは四元数を扱うクラス (bulletだとbtQuaternion)で姿勢等を指定できるが、描画側(OpenGL)は回転行列を使っている
⇒ 回転行列への変換が必要となる

物理エンジンで使う数学

- ニュートンの運動方程式(回転版)

$$\mathbf{T} = I \dot{\boldsymbol{\omega}}$$

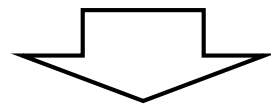
トルク[N・m] = 慣性テンソル×角加速度[rad/s²]

- 角速度 $\boldsymbol{\omega}$ は角度 θ or 四元数 q の時間微分
- 慣性テンソルは回転版質量のようなもの
 - 回転のしにくさを表す
 - 同じ質量でも形状やまわす軸でまわしやすさは変わる
 - 物理エンジンにはこれを計算するための関数が用意されている

物理エンジンで使う数学

- 回転版の前進オイラー法(四元数使用)

$$\boldsymbol{\omega} = \frac{\partial \boldsymbol{\theta}}{\partial t}, \dot{\boldsymbol{\omega}} = \frac{\partial \boldsymbol{\omega}}{\partial t}$$



角度 $\boldsymbol{\theta}$ を四元数 \boldsymbol{q} にして離散化

$$\boldsymbol{q}(t + \Delta t) = \boldsymbol{q}_w \boldsymbol{q}(t)$$

$$\boldsymbol{\omega}(t + \Delta t) = \boldsymbol{\omega}(t) + \dot{\boldsymbol{\omega}}(t)\Delta t$$

並進運動版 : $\boldsymbol{x}(t + \Delta t) = \boldsymbol{x}(t) + \boldsymbol{v}(t)\Delta t$

物理エンジンで使う数学

- 1ステップあたりの姿勢変化 q_w について
 - 1ステップあたりの姿勢変化量： $\omega\Delta t$
⇒ これを四元数変化量に変換すれば良い
 - ベクトル $\omega\Delta t$ をオイラー角 (α, β, γ) と考えて,

$$q_w = q_x q_y q_z = \begin{pmatrix} \sin \frac{\alpha}{2} \cos \frac{\beta}{2} \cos \frac{\gamma}{2} - \cos \frac{\alpha}{2} \sin \frac{\beta}{2} \sin \frac{\gamma}{2} \\ \cos \frac{\alpha}{2} \sin \frac{\beta}{2} \cos \frac{\gamma}{2} + \sin \frac{\alpha}{2} \cos \frac{\beta}{2} \sin \frac{\gamma}{2} \\ \cos \frac{\alpha}{2} \cos \frac{\beta}{2} \sin \frac{\gamma}{2} - \sin \frac{\alpha}{2} \sin \frac{\beta}{2} \cos \frac{\gamma}{2} \\ \cos \frac{\alpha}{2} \cos \frac{\beta}{2} \cos \frac{\gamma}{2} + \sin \frac{\alpha}{2} \sin \frac{\beta}{2} \sin \frac{\gamma}{2} \end{pmatrix}^T$$

と変換できる.

GLFWによるOpenGLアプリケーション

- 本実験テーマではGLFWライブラリによってアプリケーションを構築しています.
 - OpenGLはグラフィック用APIなのでGUI関係は別に実装する必要がある
⇒ GLUT,GLFWなど
 - GLFWはOpenGLのための最低限のGUI環境しか提供しないので, 本実験は別途ImGuiというものをを用いている.
 - 実験サンプルはすべてGLFWベースですが, 別のを使っても構わない(ただし自己責任で)

実際のコードを見せながら構造を解説します.