Genesis: MPM – multibody interaction

Aug. 7, 2026

マルチボディの蝶と MPM.Liquid の連成計算．翅は N*N の剛体で弾性体を表現している．空気は，MPM.Liquid の水の密度と体積剛性を変更することで近似している．粘性は無し．粒子法なので精度は期待できず．力学もあいまい．蝶の翅がクラップ＆フリングしている間の空気の流れ（流速ベクトル）だけでも可視化できればいい方．

(1) 蝶の構成

．3 剛体のボディ

．N*N の 4 枚翅（左前翅，左後翅，右前翅，右後翅）

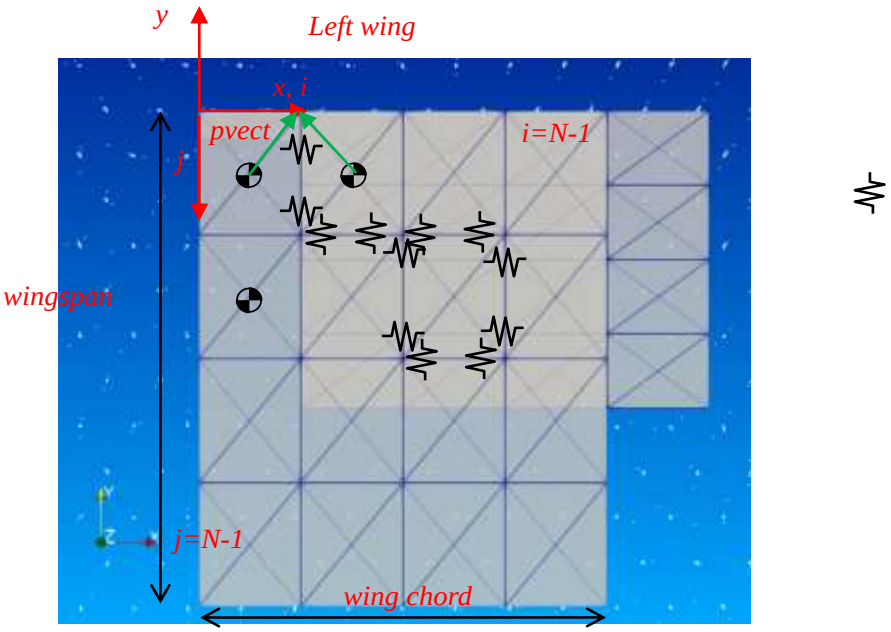
．それぞれの剛体は，morphs.Box（直方体）で生成→CAD の obj ファイルでもできるが計算負荷がかかる．Box-Box の衝突計算に対して専用 API を持っている．

(2) 結合

．剛体同士は，各直方体の頂点同士が並進バネと並進ダンパで結合されている．

．前縁の剛体同士は，3 軸の姿勢がトルクばねとトルクダンパで結合されている．

．胸（ボディの真ん中）と翅根元の前方半分（<N/2）が並進バネと並進ダンパで結合されている．



バネダンパ反力は以下のとおり．自然長は 0．ダンパはバネ方向とは独立になっている．式の単純化と高速化のため．Ri,j は翅剛体の姿勢行列．

$$\mathrm{f}_{i,j} = \mathrm{k}\left(R_{i+1.j}p_{i+1,j} - R_{i.j}p_{i,j}\right) + \mathrm{c}\left(R_{i+1.j}v_{i+1,j} - R_{i.j}v_{i,j}\right)$$

$$\mathrm{f}_{i+1,j} = -\mathrm{f}_{i,j}$$

これによって発生する重心周りのトルクは，

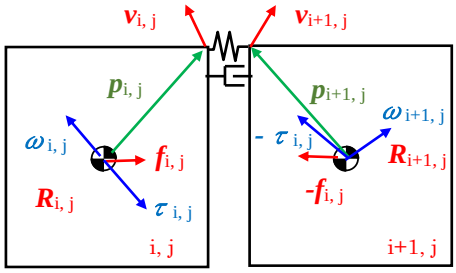
$$\mathrm{T}_{i,j} = p_{i,j} \times f_{i,j}$$

$$\mathrm{T}_{i+1,j} = p_{i+1,j} \times f_{i+1,j}$$

姿勢を合わせるためのトルク反力は，以下の通り．

$\tau_{i,j} = \mathrm{k}R_{i+1,j}{R_{i,j}}^T + \mathrm{c}(\omega_{i+1,j} - \omega_{i,j})$

$\tau_{i+1,j} = -\tau_{i,j}$



(3) 運動

前翅，後翅を以下の目標値ではばたかせている．制御は角度で PD.

$\varphi_f(\mathrm{t}) = A_f \sin(2\pi f t)$

$\varphi_h(\mathrm{t}) = A_h \sin(2\pi f t)$

$\dot{\varphi}_f(\mathrm{t}) = 2\pi f A_f \cos(2\pi f t)$

$\dot{\varphi}_h(\mathrm{t}) = 2\pi f A_h \cos(2\pi f t)$

疑似的にクラップ&フリングさせるために，A_f=60[deg], A_h=50[deg]にして前翅の方が大きく羽ばたくようにしてある．なお，はばたきモーメントを与えているのは，i=j=0 の剛体のみ．反モーメントは真ん中のボディ．

上記より，目標の姿勢行列 R_{target} は，i=j=0 の剛体の姿勢行列を R_{00} として，

$R_{target,f} = \mathrm{R}_{f00}\mathrm{R}_{\mathrm{x}\varphi f}$

$R_{target,h} = \mathrm{R}_{f00}\mathrm{R}_{\mathrm{x}\varphi \mathrm{h}}$

ここで，Rxa は，x 軸で a 度回転の行列．目標角速度は

$\omega_{target,f} = \mathrm{R}_{f00}[\dot{\varphi}_f(\mathrm{t}), 0, 0]^T$

$\omega_{target,h} = \mathrm{R}_{h00}[\dot{\varphi}_h(\mathrm{t}), 0, 0]^T$

となる．よって制御量は，

$u_f = \mathrm{k}\left(R_{target,f}{\mathrm{R}_{f00}}^T\right) + \mathrm{c}(\omega_{target,f} - \omega_f)$

$u_h = \mathrm{k}\left(R_{target,h}{\mathrm{R}_{h00}}^T\right) + \mathrm{c}(\omega_{target,h} - \omega_h)$

(4) 実行と可視化

genesis がインストールされているディレクトリで以下を実行してアクティベイトし，

> source genesis/bin/activate

実行ファイルがあるディレクトリに移動し，以下を実行する．

> python3 butterfly**.py

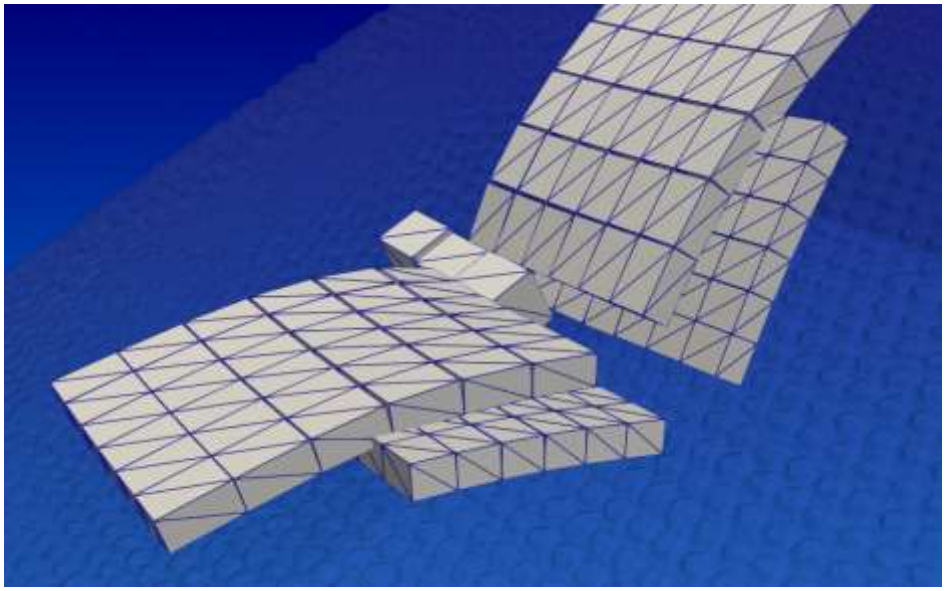
以下で可視化できる

cfd.pvsm

もしくは,
result/scene.pvd

粒子の点データの体積データ化は，Gaussian resampling
同様に，PointVolumeInterpolation をつかう．

morphs.Box の立方体同士の衝突は，判定が甘く，めり込む．移動量は，1 辺の 1%以下がデフォルト．
デフォルトの接触判定は AABB: Axis-Aligned Bounding Box

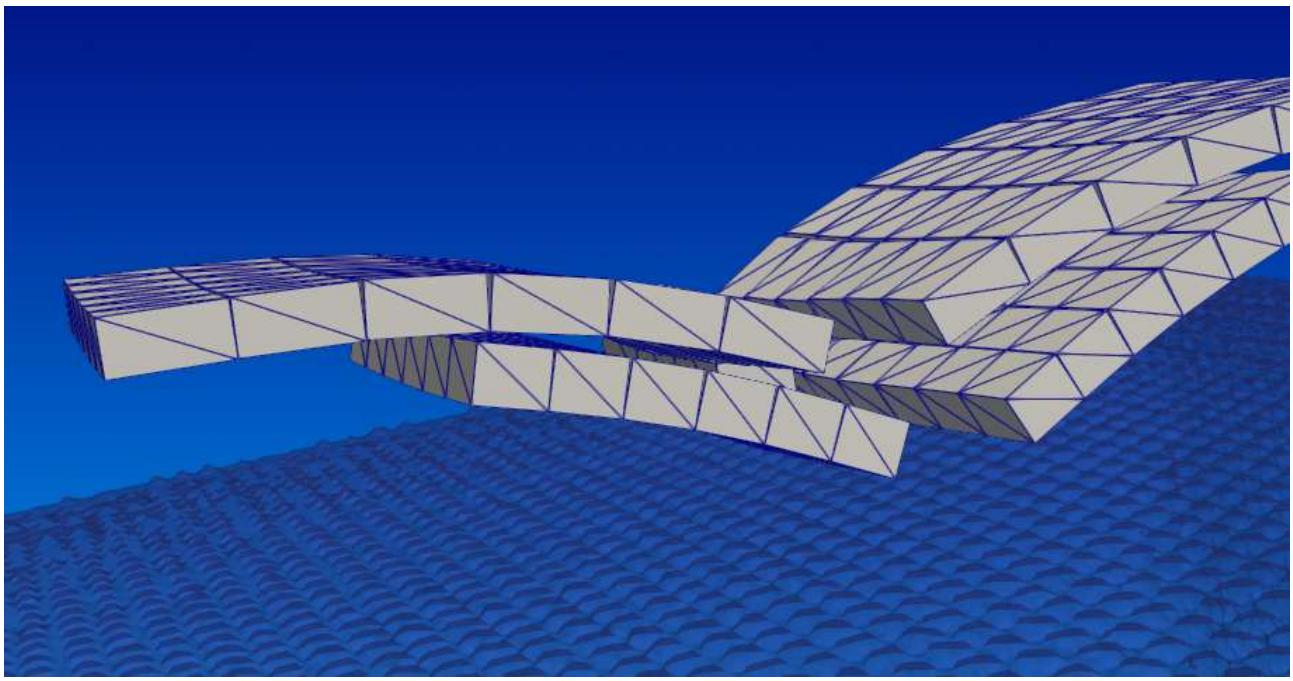


Box-Box 判定にするには,

```
dt = 1.0e-3
substeps = 4
h = dt / substeps
scene = gs.Scene(
        sim_options=gs.options.SimOptions( dt=dt, substeps=substeps, ),
        rigid_options=gs.options.RigidOptions(
                dt=dt,
                # 直方体専用の接触判定
                box_box_detection=True,
                # 面接触では複数接触点を使用
                enable_multi_contact=True,
                # 接触拘束ソルバ
                constraint_solver=gs.constraint_solver.Newton,
                # 既定値 50 より増やす
                iterations=100,
```

```
        ls_iterations=50,
        # 最初は 4 サブステップ程度の時定数
        constraint_timeconst=4.0 * h, ), )
```

constraint_timeconst = 0.001　でめり込まなくなった．直方体 1 辺の 0.1%ぐらいを許容しているような意味合い．



なお，初期位置において，剛体と粒子が重なっている部分は，以下の関数で粒子の方を非アクティブにして，計算から除外している．
deactivate_mpm_particles_inside_rigid_boxes

また，MPM は，SPH とちがって，計算空間の境界面を実際の壁より 3 格子分外側にしている．つまり，実際の物理空間が L^3 の領域だとすると，MPM の計算空間は(L+dx)^3 にしている．dx は格子幅．

さらに，剛体の肉厚（2mm）に対し，MPM 格子のサイズ（5mm）が大きいため，粒子が翅を突き抜けないように，CPIC を true にしているが，この計算が極めて重い．この設定で，これを true にしただけで計算時間が 10 倍以上になる．

enable_CPIC= False, # True, # for body-particle collision

．自己衝突の制御
以下により，翅を構成している剛体同士の接触判定はしないようにしている．例えば，左の前翅は，自分とは衝突判定せず，左の後翅と，右の前翅と衝突判定する．
左の前翅：
contype　　=0x0001,

conaffinity=0xFFFE,

右の前翅

contype　　=0x0002,

conaffinity=0xFFFD,

とすると，下 4bit で表すと，左の前翅同士では，

	左の前翅 I 番目剛体	左の前翅 J 番目剛体
contype	0001	0001
conaffinity	1110	1110
論理積	0000	0000
判定	False	False
衝突判定	False	

左の前翅と右の前翅では，

	左の前翅 I 番目剛体	右の前翅 J 番目剛体
contype	0001	0010
conaffinity	1101	1110
論理積	0001	0010
判定	True	True
衝突判定	True	

になる．判定は，論理和．どちらかが真なら真．最終的には，衝突する翅だけ判定するようにしている．例えば，左前翅は，右後翅とは衝突しない．

```
            wingLF[body_index] = scene.add_entity(
                material=Material,
                morph=gs.morphs.Box(                # Mesh
                    pos=(       i*wingFchord/N+wingFchord/N/2+CenterX,      -j*wingFspan/N-wingFspan/N/2-bodyWidth/2, wingwingDis/2.0 ),
                    size=( wingFchord/N, wingFspan/N, wingThickness ),
                    fixed=False,
                    collision=True,

                    contype    =0x0001, # 0b0000000000000001
                    conaffinity=0xFFFE, # 0b1111111111111110

                ),
```