

物理シミュレータの MuJoCo で、マルチボディとしてモデル化されるシリアルリンクの Cobotta と、CAD のデータを FEM による弾性体としてモデル化される魚の数値計算を行う。

(1) CAD の msh フォーマット化

stl は、surface モデル（2 次元三角形で構成される表面データ）なので、MuJoCo フォーマットでは中空モデルになる。中実モデルにするためには、中を埋めた四面体要素（Tetrahedron）に変換しなければならない。と思ったが、ステップフォーマットファイルの方が情報量が多く確実なので、こちらを変換する。

・ Gmsh のインストール方法は以下

2026 年 9 月現在、apt からインストールされるのは、version 4.12.1. でも最新ではなく、optimize オプションが使えない。

```
> sudo apt update
> sudo apt install gmsh
> gmsh --version
```

よって、apt 版を使わず、公式 Linux バイナリ版（4.15.2）をインストールする。
まず、CPU を調べる。

```
> uname -m
```

x86_64 と表示されたら、以下を実行する。

```
> curl -fLO https://gmsh.info/bin/Linux/gmsh-4.15.2-Linux64.tgz
> sudo tar -xzf gmsh-4.15.2-Linux64.tgz -C /opt
> sudo ln -sfT /opt/gmsh-4.15.2-Linux64/bin/gmsh /usr/local/bin/gmsh
> hash -r
```

その後、確認

```
> which gmsh
> gmsh -version
> gmsh -info
```

正しくインストールされていることが分かる。

```

kikut@kikut3:/tmp$ which gmsh
/usr/local/bin/gmsh
kikut@kikut3:/tmp$ gmsh -version
4.15.2
kikut@kikut3:/tmp$ gmsh -info
Version      : 4.15.2
License      : GNU General Public License
Build OS     : linux64
Build date   : 20260324
Build host   : gmsh.info
Build options: 64Bit ALGLIB[contrib] ANN[contrib] Bang Blas[petsc] Blossom Cgns DIntegration Dlopen GmHex Eigen[contrib] Fltk Gmsh[contrib] Hxt Jpeg Kibipack Lapack[petsc] LinuxJoystick MathEx[contrib] Med Mesh Netis[contrib] Png Popen Netge n Nii2mesh ONELAB ONELABMetamodel OpenCASCADE OpenCASCADE-CAF OpenGL OpenMP OptHom PETSc Parser Plugins Png Post QuadMes hingTools QuadTri Solver TetGen/BR TinyXML2[contrib] Untangle Voro++[contrib] WinalumUntangler Zlib tinyobjloader
FLTK version : 1.3.11
PETSc version: 3.14.4 (real arithmetic)
OCC version  : 7.8.1
MED version   : 4.1.0
Packaged by   : geuzaine
Web site      : https://gmsh.info
Issue tracker : https://gitlab.onelab.info/gmsh/gmsh/issues

```

GUI からでなく、プロンプト上から行うのが確実なので、いったん以下の二つの geo ファイル：

step_to_surface_stl.geo

stl_to_tet_msh.geo

を作る。前者は、step ファイルから表面形状の stl ファイルを取り出し、後者は stl から中実のテトラ要素を作り出す。step ファイルは mm 単位で出力されているので、一旦 1/1000 にして m 単位にして処理している。品質の悪いメッシュを作らないために、いろいろ試行錯誤している。

. step_to_surface_stl.geo

```
SetFactory("OpenCASCADE");
```

```
// STEP は mm 単位なので、読み込み時に m 単位へ変換する
```

```
Geometry.OCCScaling = 0.001;
```

```
Merge "flyingfish.stp";           // 読み込みファイル。ここを書き換える
```

```
// 中間 STL 用の細かい表面メッシュ：0.5～1.0 mm
```

```
Mesh.CharacteristicLengthMin = 0.0005;
```

```
Mesh.CharacteristicLengthMax = 0.001;
```

```
Mesh.Algorithm = 6;
```

```
Mesh.SaveAll = 1;
```

```
Mesh 2;                          // 2 次元
```

```
Save "surface.stl"; // 出力ファイル
```

. stl_to_tet_mesh.geo

```
// STEP の微小トポロジーを使わず、STL 外皮を再パラメータ化して再メッシュする
```

```
Merge "surface.stl";           // 入力ファイル
```

```
// 40° 以上の実際の稜線は残し、滑らかな面上の不要な STEP seam は跨ぐ
ClassifySurfaces{40 * Pi / 180, 1, 1, 180 * Pi / 180};
CreateGeometry;
```

```
Surface Loop(1) = Surface{:};
Volume(1) = {1};
```

```
// 最終メッシュ : 0.5~5 mm
Mesh.CharacteristicLengthMin = 0.0005;    // 0.5 mm
Mesh.CharacteristicLengthMax = 0.005;     // 5m
Mesh.Algorithm = 6;
Mesh.Algorithm3D = 1;
Mesh.Optimize = 1;
Mesh.OptimizeThreshold = 0.3;
Mesh.OptimizeNetgen = 1;
```

```
// MuJoCo 用 : MSH 2.2、ASCII、一次四面体のみ
Physical Volume("softbody") = {1};
Mesh.MshFileVersion = 2.2;
Mesh.Binary = 0;
Mesh.ElementOrder = 1;
Mesh.SaveAll = 0;
```

```
Mesh 3;                // 3 次元
RenumberMeshNodes;
RenumberMeshElements;
Save "flyingfish.msh"; // 出力ファイル. ここを書き換える
```

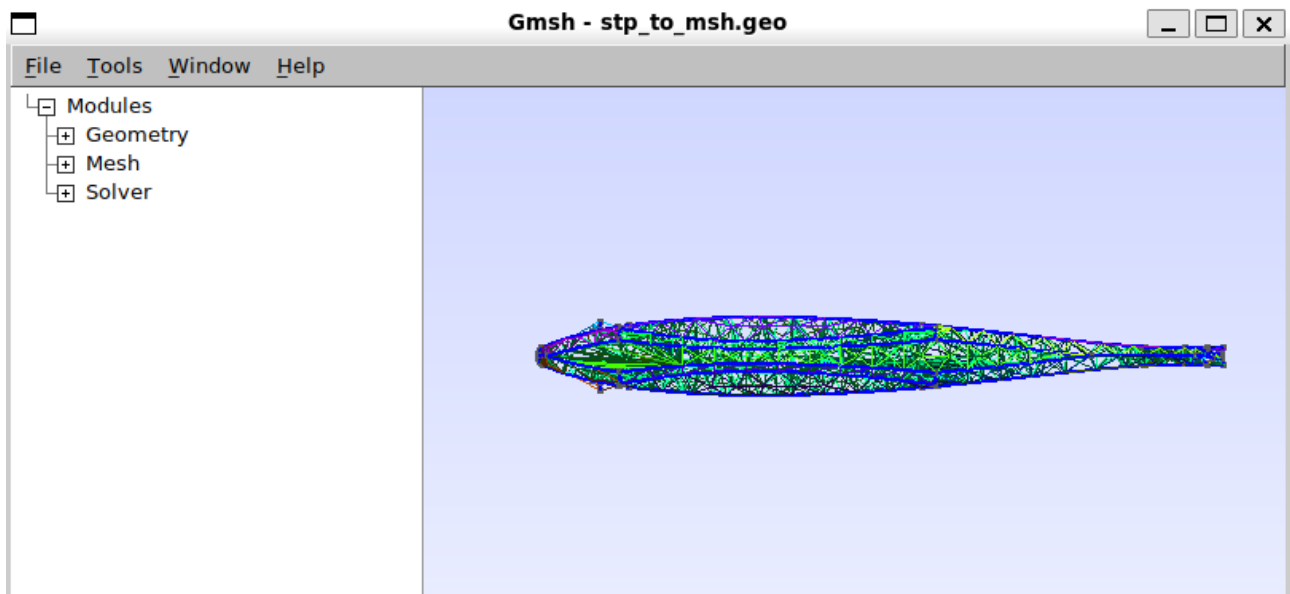
以下を実行する.

```
> gmsh step_to_surface_stl.geo
```

GUI が立ち上がり, 中間ファイル surface.stl が出力される. 単位を 1/1000 にした関係上, 恐ろしく小さく表示されるので, 見たければ拡大する. いったん Quit して, 以下を実行する.

```
> gmsh stl_to_tet_msh.geo
```

flyingfish.msh ファイルが出力される.



ファイルは、CAD ディレクトリにコピーしておく。

CAD/flyingfish.msh

(2) 実行

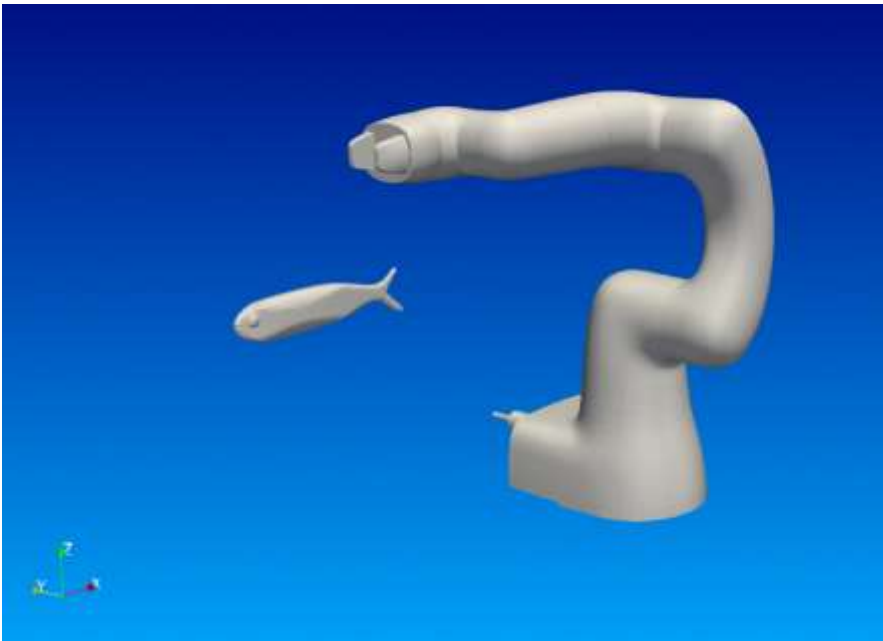
物理シミュレータとしては MuJoCo であるが、Genesis に入っている MuJoCo からでも、Taichi にインストールした MuJoCo からでも実行できる。ここでは、Genesis に入れた MuJoCo から実行してみる。まずは、アクティベート

```
> source genesis/bin/activate
```

その後、本体のコードがあるディレクトリに移動し、実行

```
> python3 cobottaDynamics.py
```

STL というディレクトリに計算結果が出力されるので、paraview で可視化する。



(3) XML ファイル

```
<mujoco model="cobotta">
  <compiler angle="radian" meshdir="." autolimits="true"/>

  <option timestep="0.000002" integrator="implicitfast" impratio="1" noslip_iterations="0">    //
  時間刻み [s]
</option>

  <default>
    <default class="cobotta">
  ...
    </default>
  </default>

  <asset>
    <material class="cobotta" name="white" rgba="1 1 1 1"/>
    <material class="cobotta" name="off_white" rgba="0.901961 0.921569 0.929412 1"/>
    <material class="cobotta" name="black" rgba="0.25 0.25 0.25 1"/>

    <!-- Collision meshes -->
    <mesh name="base00_c" file="CAD/base00.stl"/>
    <mesh name="link01_c" file="CAD/link01.stl"/>
    <mesh name="link02_c" file="CAD/link02.stl"/>
    <mesh name="link03_c" file="CAD/link03.stl"/>
    <mesh name="link04_c" file="CAD/link04.stl"/>
    <mesh name="link05_c" file="CAD/link05.stl"/>
    <mesh name="hand_c" file="CAD/hand06.stl"/>
    <mesh name="finger00_c" file="CAD/finger00.stl"/>
    <mesh name="finger01_c" file="CAD/finger01.stl"/>

    <!-- Visual meshes -->
    <mesh name="base00" file="CAD/base00.obj"/>
    <mesh name="link01" file="CAD/link01.obj"/>
    <mesh name="link02" file="CAD/link02.obj"/>
    <mesh name="link03" file="CAD/link03.obj"/>
    <mesh name="link04" file="CAD/link04.obj"/>
```

```

<mesh name="link05" file="CAD/link05.obj"/>
<mesh name="hand06" file="CAD/hand06.obj"/>
<mesh name="finger00_nogroup" file="CAD/finger00_nogroup.obj"/>
<mesh name="finger01_nogroup" file="CAD/finger01_nogroup.obj"/>
</asset>

<worldbody>
  <light name="top" pos="0 0 2" mode="trackcom"/>

  <body name="base00" pos="0 0 0" euler="0 0 0" childclass="cobotta">

    // cobotta の定義は略

  </body>

  <!-- 弾性体を支持する床 --> // 床
  <geom name="floor"
    type="plane"
    size="2 2 1"
    friction="0.8 0.02 0.001"
    rgba="0.8 0.8 0.8 1"/>

  // 四角形要素にした CAD モデルを弾性体モデルとして読み込むところ
  <flexcomp name="flyingfish" type="gms" file="CAD/flyingfish.msh"
    pos="-0.2 0.2 0.2" mass="0.10" radius="0.003"
    rgba="0.25 0.55 0.9 1">
    <contact selfcollide="none" internal="false" condim="3" friction="0.8 0.02 0.001"
solref="0.001 1" solimp="0.95 0.99 0.001"/>
    <edge damping="1"/>
    <elasticity young="5e4" poisson="0.3" damping="0.01"/>
  </flexcomp>

</worldbody>

<actuator>
  .....
</actuator>

<keyframe>
  <key name="home" qpos="0 0 0 0 0 0 0" ctrl="0 0 0 0 0 0 0"/>
</keyframe>
</mujoco>

```

solref="0.005 1"の意味は？

solref="0.005 1" は MuJoCo の接触拘束の応答設定です。

- 0.005 : 時定数 0.005 秒 (5 ms)。接触による食い込みをこの速さで戻そうとします。小さいほど硬い接触になります。
- 1 : 減衰比 1.0。臨界減衰で、跳ね返りや振動を抑えつつ最短で安定させる設定です。

つまり「5 ms 程度で食い込みを解消し、振動しにくい接触」です。

なお、timestep="0.00001" (10 μ s) なら 5 ms は十分大きく、MuJoCo の最小時定数制限にもかかりません。弾性体をより柔らかく接触させたいなら第一値を大きく、硬くしたいなら小さくします。

`solimp="0.95 0.99 0.001"`の意味は？

`solimp="0.95 0.99 0.001"` は接触拘束の「インピーダンス（実効的な硬さ・効き具合）」を、接触の食い込み量に応じてどう変えるかの設定です。

- `0.95` : 接触開始直後のインピーダンス
- `0.99` : 食い込みが増えたときの最大インピーダンス
- `0.001` : その遷移幅。食い込み量 1 mm 程度で `0.95 → 0.99` に変化します

つまり、接触直後からかなり硬く、1 mm ほど食い込むとさらに硬くなる設定です。`solref="0.005 1"` と組み合わせると、比較的硬く安定した接触になります。