

空飛ぶクルマ高精度リアルタイムシミュレーション 技術構築



What is **Flying Cars**?

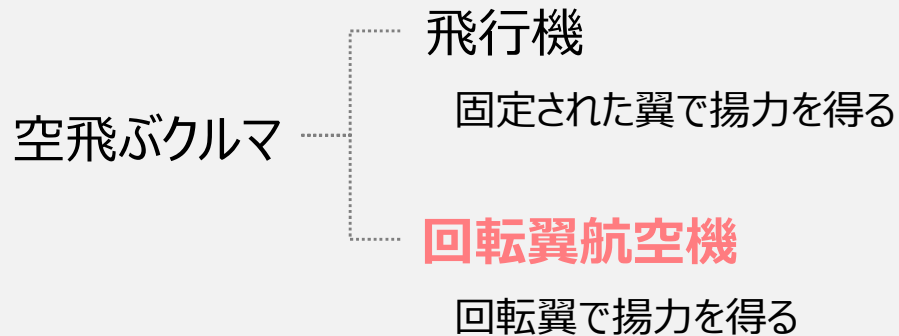
複数のロータを有した垂直離着陸可能なモビリティ



<https://clipset.com/toyota-coche-volador-skydrive/>

What is Flying Cars?

■ 空飛ぶクルマの分類



マルチロータ型



機体を傾けて揚力・推力を得る

<https://clipset.com/toyota-coche-volador-skydrive/>

リフトクルーズ型



揚力用と推力用のロータが別々

<https://jbpress.ismedia.jp/articles/-/77748>

ベクタードスラスト型



ロータを傾けて揚力・推力を得る

<https://www.jobyaviation.com>

What is Flying Cars?

空飛ぶクルマによる社会的メリット

01 経済成長

- ・ 人とモノの移動を活発化できる
- ・ 保険, レジャーなど**新たなサービス**が生まれる

**2050年市場規模
180兆円以上**

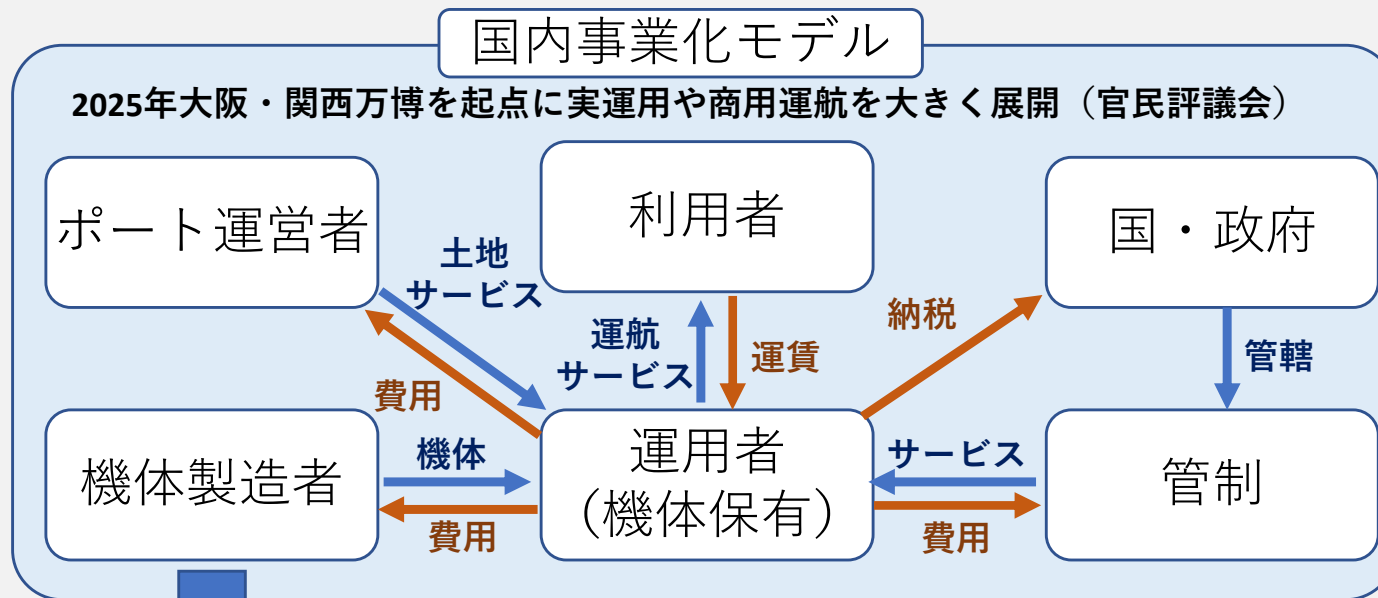
02 都市間移動

- ・ **渋滞解消**として大都市での飛行が期待される
- ・ 救急や災害での緊急対応が期待される

03 自動運転

- ・ **パイロットなし**での飛行が可能
- ・ 自律飛行により航空機同士の衝突を防ぐ

Background



機体の要素技術/課題

センサー

- ・姿勢制御用選定
- ・視覚認識用開発

ローター

- ・機体制御
- ・静音対策

フレーム形状

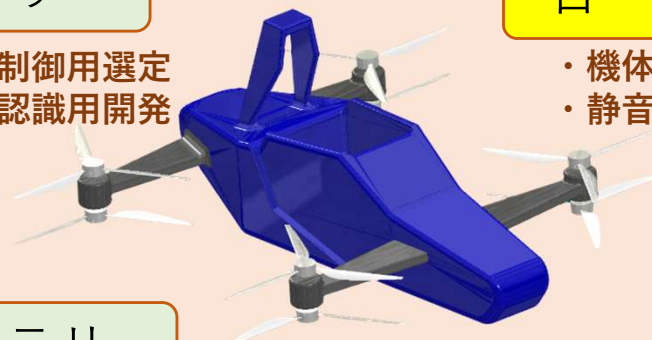
- ・空力性能評価

バッテリー

- ・航続距離
- ・軽量化（高密度化）

フレーム材料

- ・軽量化/強度



単体評価可能

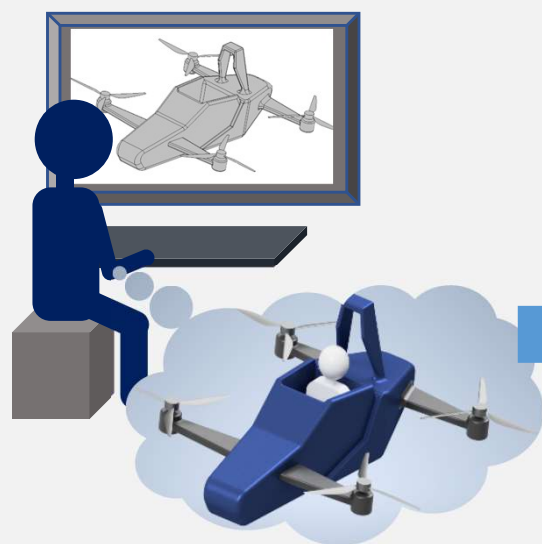
飛行による評価必要

◆ 空飛ぶクルマの開発現状



設計変更の度に再製作が必要・莫大な時間とコストを要する

◆ 実験レス開発への期待



シミュレーションによる
仮想実験



機体製作

最終試験のみ



試験飛行

開発スピードの向上

コストの低減

安全性の確保

◆ デジタルフライングカー（空飛ぶクルマ飛行を再現できる仮想空間）

空力モデル

+

剛体モデル

+

制御モデル

空力：移動する機体周りの流れ場計算

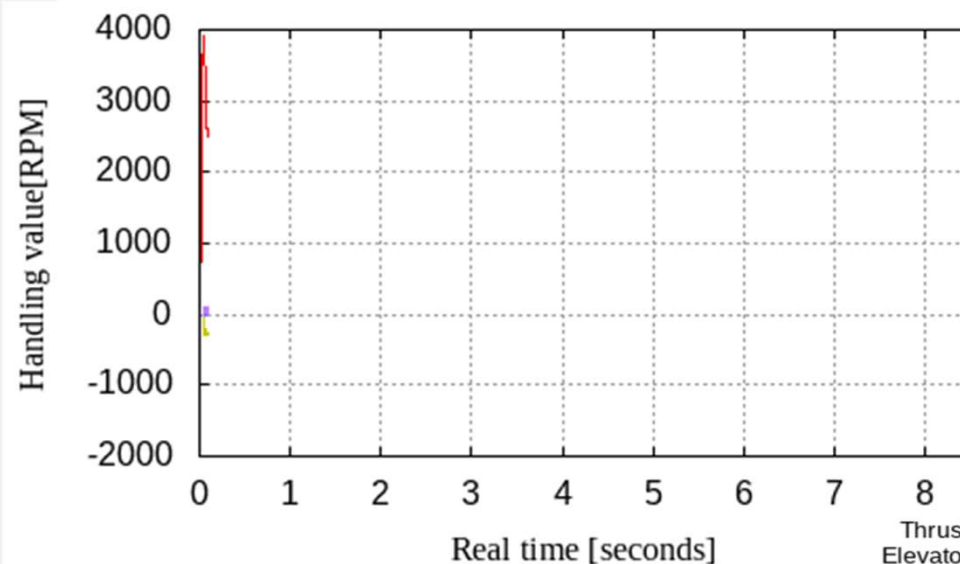
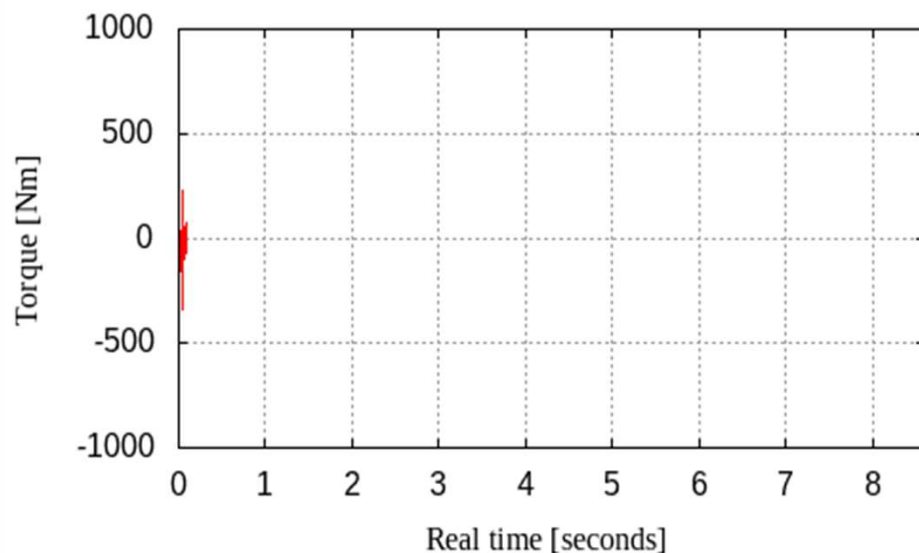
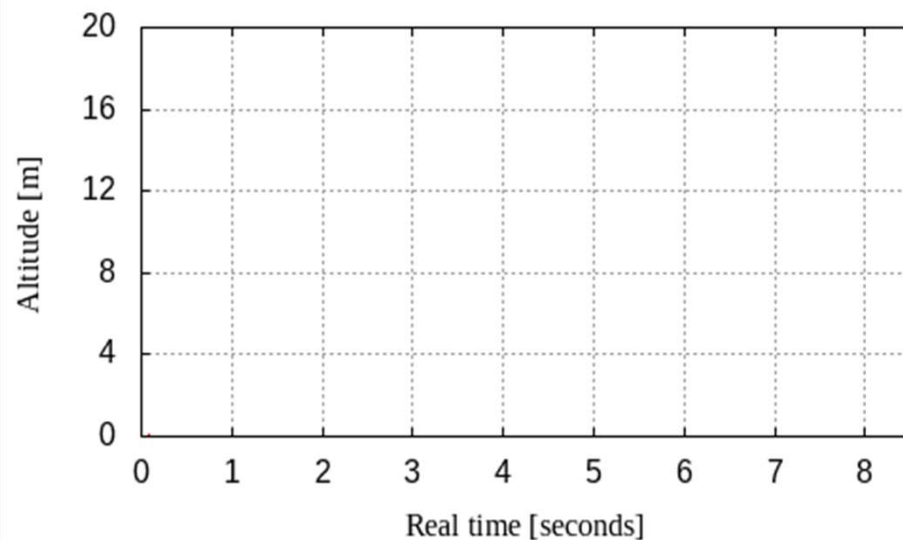
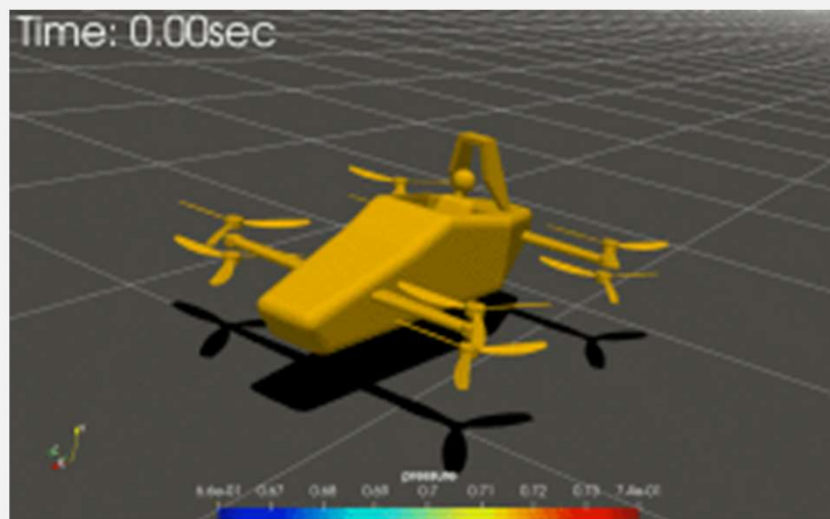
剛体：運動方程式を用いて機体の回転・移動量計算

制御：目標値に向けてロータの回転数計算

□ 設定値 : 飛行高度, 機体速度

■ 得られるデータ : ロータ回転数, 揚力, 飛行軌跡, 姿勢角度

◆ デジタルフライングカー : 2023年3月にプロトタイプが完成



Thrust —
Elevator —
Aileron —
Rudder —

■ 「デジタルフライングカー」による業界に対する期待できる波及効果

- 数千回におよぶ想定飛行試験を短時間で実施し確認できる。
- 飛行に関する予測を正確に実施できる。
- 国内事業化モデルにおける波及効果が期待できる。

国内事業化モデルにおける波及効果

ポート運営者

- ・ポートの安全対策
- ・土地利用の最適化

機体製造者

- ・ **機体の多様化**
- ・ **輸出モデル対策**

シミュレーションを新環境への開発へ応用し、高度仕様や寒冷地仕様、海外を想定した砂漠仕様等あらゆる機体開発

利用者

- ・ **飛行状況可視化**

フライトサービス：シミュレーションによる計算結果をリアルタイムで可視化

運用者 (機体保有)

- ・ **自動運転補助**
- ・ 気象予測を勘案した高精度ナビゲーション
- ・ RNPの検証

国・政府

- ・ **運用ルールの制定**

許可の有無判断：シミュレーションにより運用時の安全レベルを予測しルールへフィードバック

管制

- ・ **精密な飛行予測による事故回避管制**
- ・ 離着陸管制最適化
(安全な離陸間隔の見積)

Numerical Approach – デジタルフライングカーの計算手法 –

✓ 支配方程式

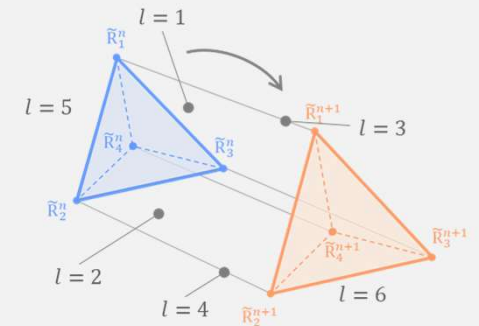
$$\frac{\partial \mathbf{q}}{\partial t} + \frac{\partial \mathbf{E}}{\partial x} + \frac{\partial \mathbf{F}}{\partial y} + \frac{\partial \mathbf{G}}{\partial z} = 0$$

$$\mathbf{q} = \begin{bmatrix} \rho \\ \rho u \\ \rho v \\ \rho w \\ e \end{bmatrix}, \mathbf{E} = \begin{bmatrix} \rho u \\ \rho u^2 + p \\ \rho uv \\ \rho uw \\ u(e + p) \end{bmatrix}, \mathbf{F} = \begin{bmatrix} \rho v \\ \rho uv \\ \rho v^2 + p \\ \rho vw \\ v(e + p) \end{bmatrix}, \mathbf{G} = \begin{bmatrix} \rho w \\ \rho uw \\ \rho vw \\ \rho w^2 + p \\ w(e + p) \end{bmatrix}$$

✓ 非構造移動格子有限体積法(UMGFV法)

4次元時空間に対する保存則を満足するように離散化

$$\mathbf{q}^{n+1}(\tilde{n}_t)_6 + \mathbf{q}^n(\tilde{n}_t)_5 + \sum_{l=1}^4 (\mathbf{q}^{n+1/2} \tilde{n}_t + \mathbf{E}^{n+1/2} \tilde{n}_x + \mathbf{F}^{n+1/2} \tilde{n}_y + \mathbf{G}^{n+1/2} \tilde{n}_z)_l = 0$$

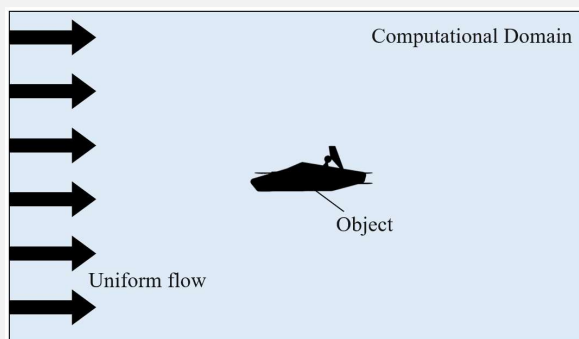


Numerical Approach – デジタルフライングカーの計算手法 –

✓ 移動計算領域法(MCD法)

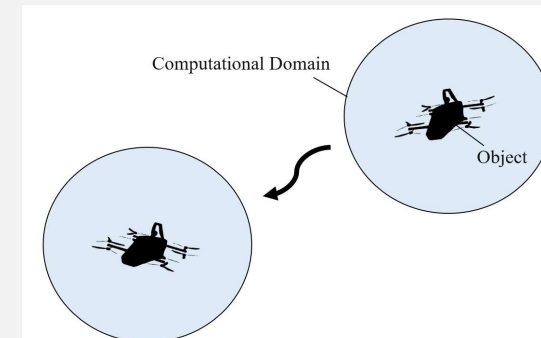
- ・ 制約をもたない自由な運動表現
- ・ 計算コストの削減

従来法



等速直線飛行のみ計算可能

MCD法

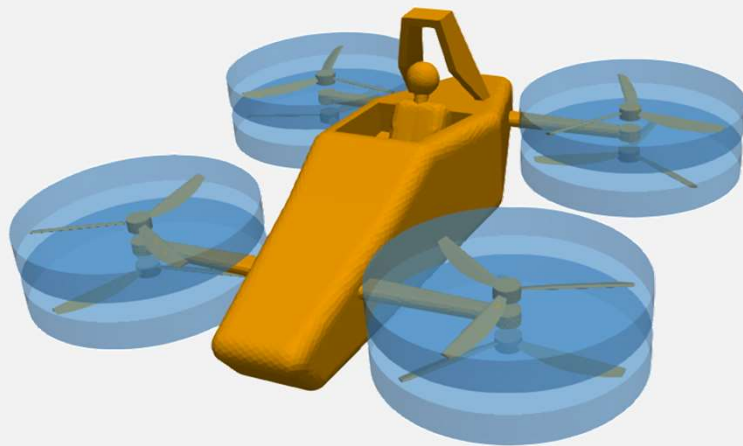
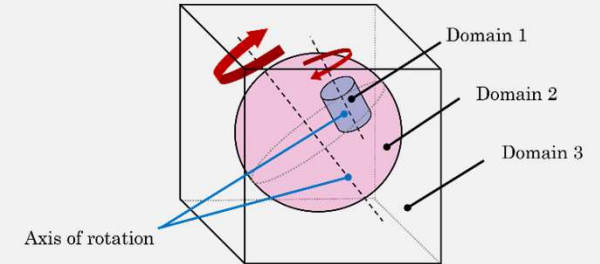


離着陸・加減速・旋回・失速墜落が計算可能

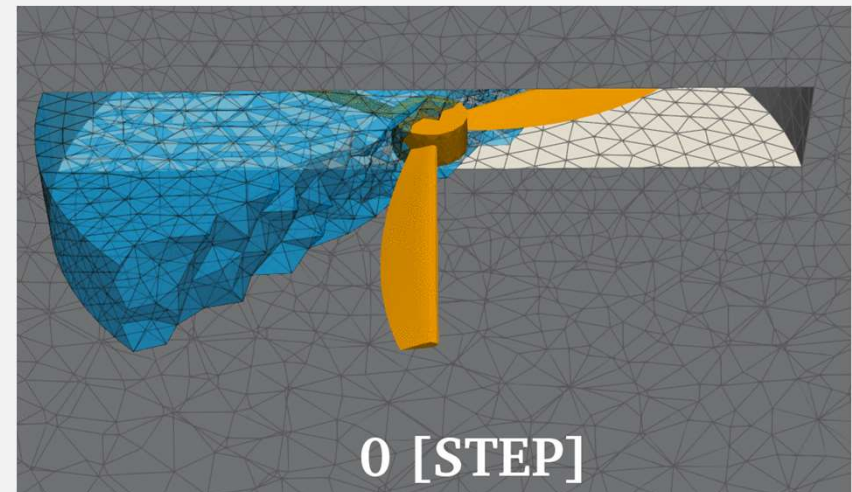
Numerical Approach – デジタルフライングカーの計算手法 –

✓ 多軸スライディングメッシュ法

- 複数の回転軸を用いたスライディングメッシュ法，運動表現の自由度を大幅UP
- ロータ部は機体の動きとは異なるため **別の計算領域** を必要とする



本研究で用いたスライディングメッシュ



スライディングメッシュでのロータ回転

Numerical Approach – デジタルフライングカーの計算手法 –

✓ 流体 – 剛体 連成計算

ニュートンの運動方程式 と オイラーの回転方程式 から機体の動きを計算する

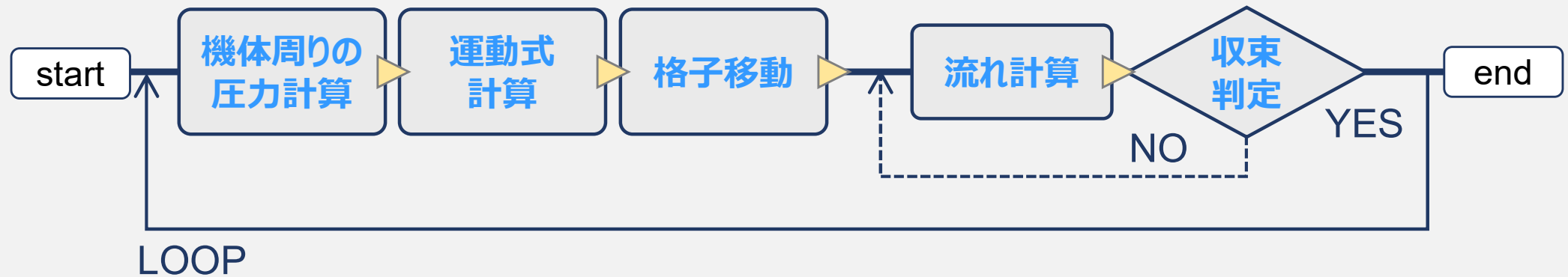
$$m \frac{d^2 \mathbf{r}}{dt^2} = \mathbf{F}$$

$$I \frac{d\boldsymbol{\omega}}{dt} + \boldsymbol{\omega} \times I \boldsymbol{\omega} = \mathbf{T}$$

m : 質量
 I : 慣性テンソル

\mathbf{r} : 位置ベクトル
 $\boldsymbol{\omega}$: 角速度ベクトル

\mathbf{F} : カベクトル
 \mathbf{T} : トルクベクトル



Numerical Approach

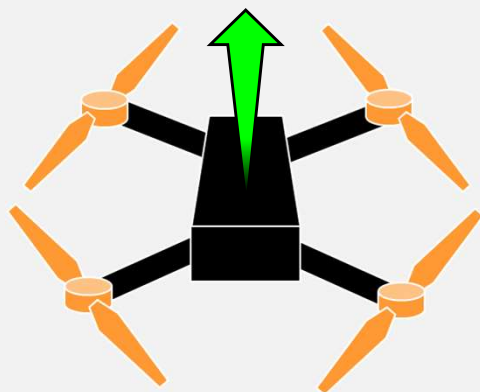
ex. マルチローター機



 : 機体移動方向

姿勢制御

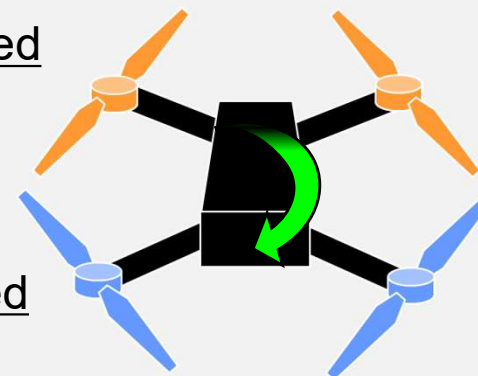
機体姿勢に対して4つのパラメータから制御を適用する



全てのロータに同程度の回転数を配分

High Speed

Low Speed



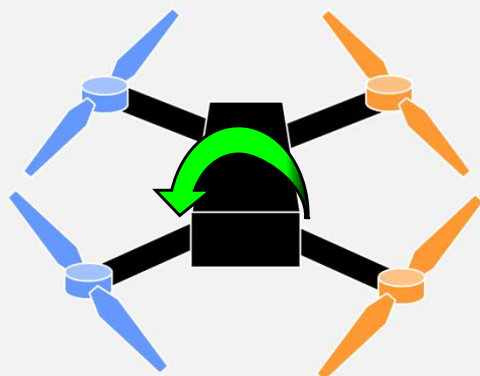
機体前後方向で回転数を変える

Altitude

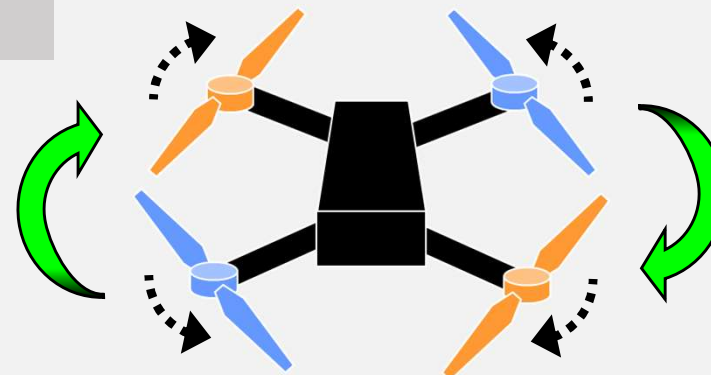
Pitch

Roll

Yaw



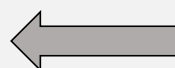
機体左右方向で回転数を変える



ロータの回転方向によって回転数を変える

目標値

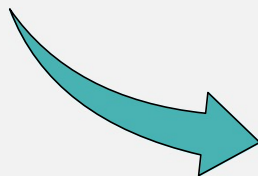
- ・機体座標前進速度
- ・高度
- ・機体姿勢角度



現在の機体速度・角度

入力

上昇速度
Pitch角速度
Roll角速度
Yaw角速度



機体伝達関数モデル

$$\frac{v_z(s)}{u_{Throttle}} = \frac{K_T}{ms}$$

$$\frac{q(s)}{u_{Pitch}(s)} = \frac{LK_T}{I_y s}$$



$$\frac{p(s)}{u_{Roll}(s)} = \frac{LK_T}{I_x s}$$

$$\frac{r(s)}{u_{Yaw}(s)} = \frac{K_A}{I_z s}$$

PID制御

$$h(t) = K_p e(t) + K_i \int e(t) dt + K_d \frac{de(t)}{dt}$$

制御器

ロータ
回転数取得

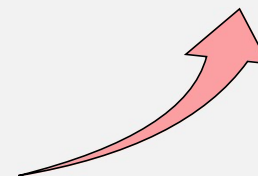


Mixing処理



操作量

Throttle
Pitch
Roll
Yaw



計算事例（１）

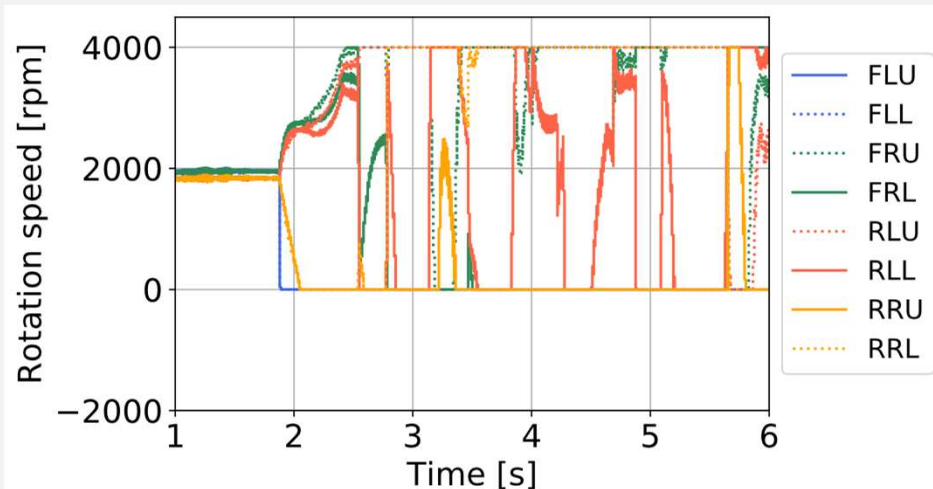
ホバリング時のローター停止

⇒設計思想に含まれない
ローター逆回転を指示

従来法による制御



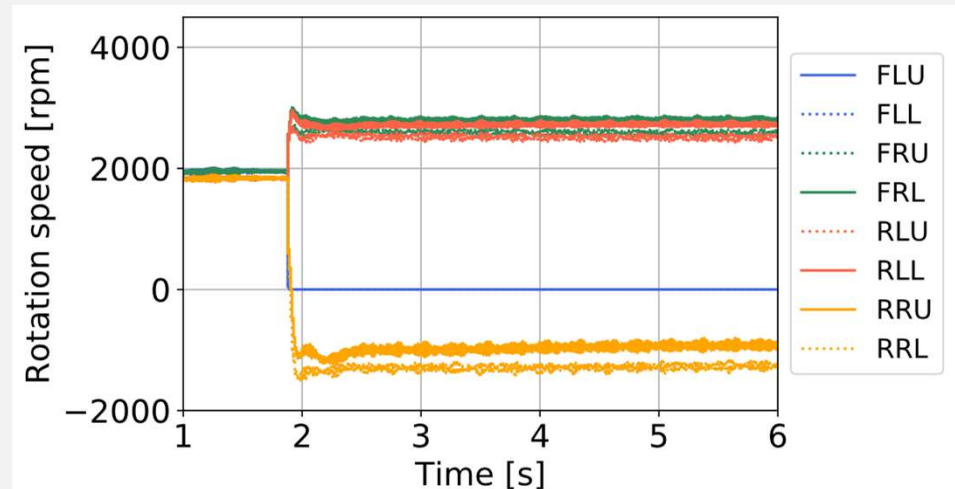
0 [rpm] to 4000 [rpm]



AIによる自動制御



逆回転



計算事例（2）

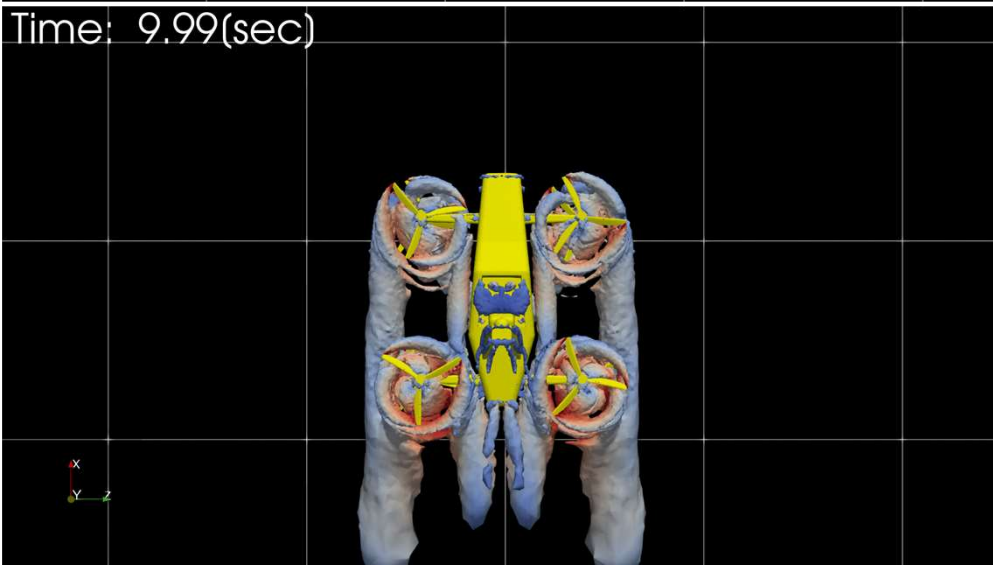
飛行中のローター停止（AI自動制御）

✓ Stopping rotor : FLU and FLL

Time: 9.99(sec)



Time: 9.99(sec)

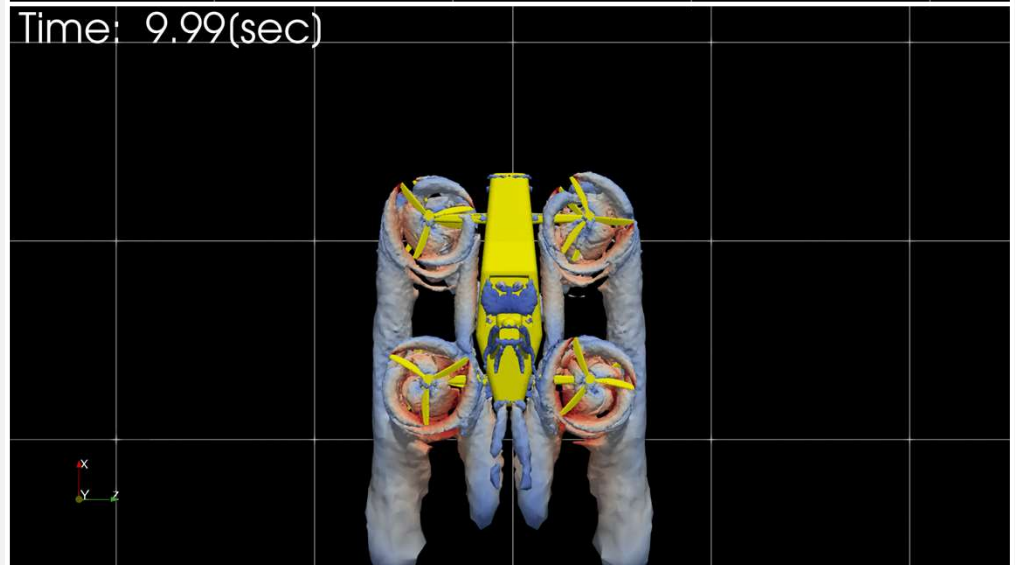


✓ Stopping rotor : RLU and RLL

Time: 9.99(sec)



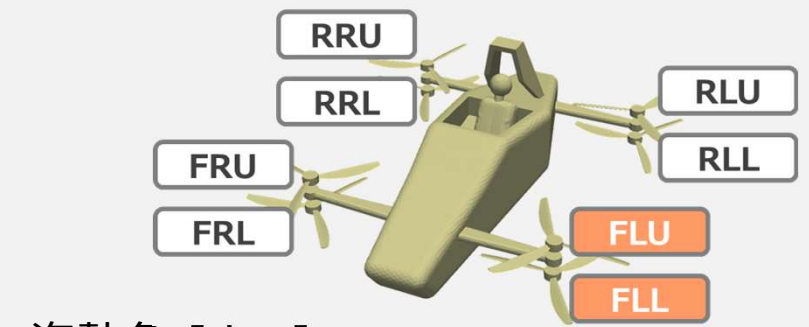
Time: 9.99(sec)



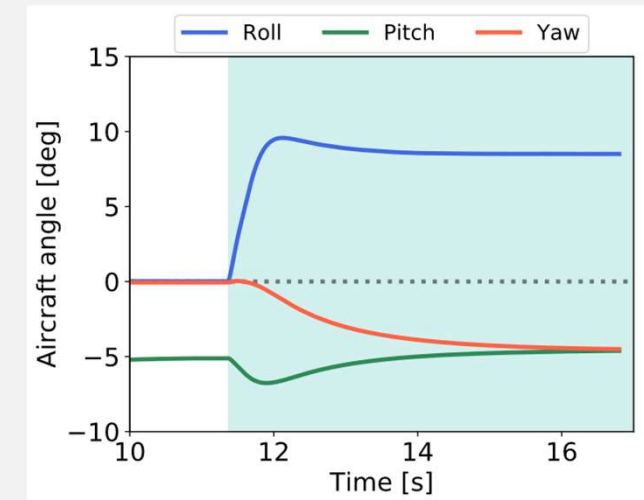
計算事例（２）

飛行中のローター停止（AI自動制御）

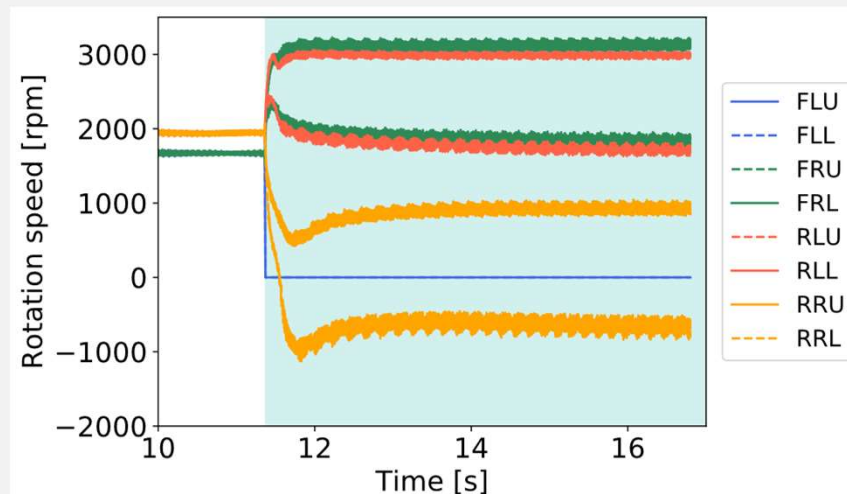
✓ Stopping rotor : FLU and FLL



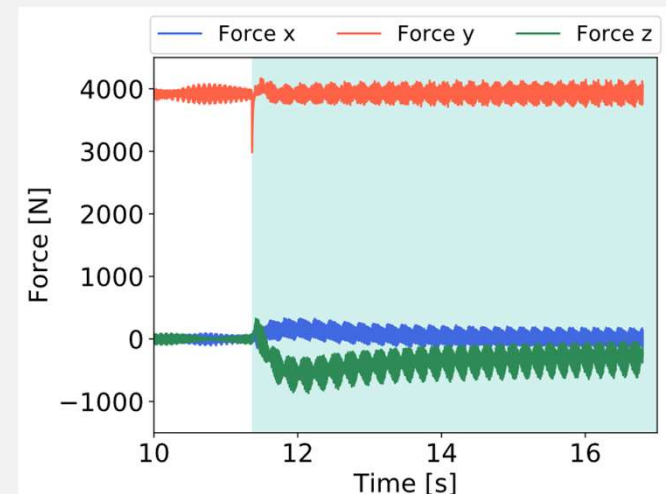
姿勢角 [deg]



ローター回転数[rpm]



機体にかかる力 [N]



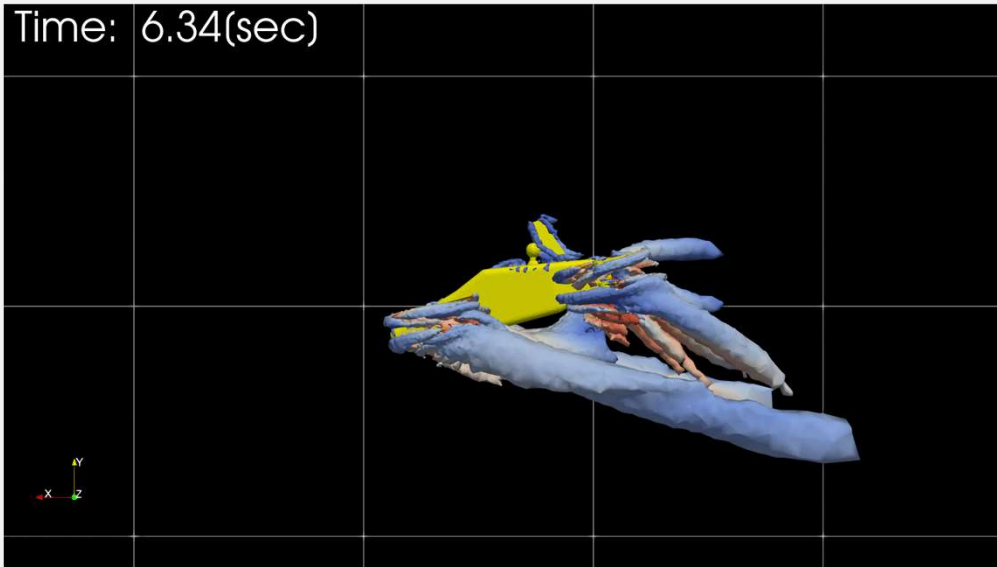
計算事例 (3)

角速度変化による減速性能比較

15 [deg] $\pi / 8$ [rad/s]

Velocity magnitude
0.0e+00 0.05 1.0e-01

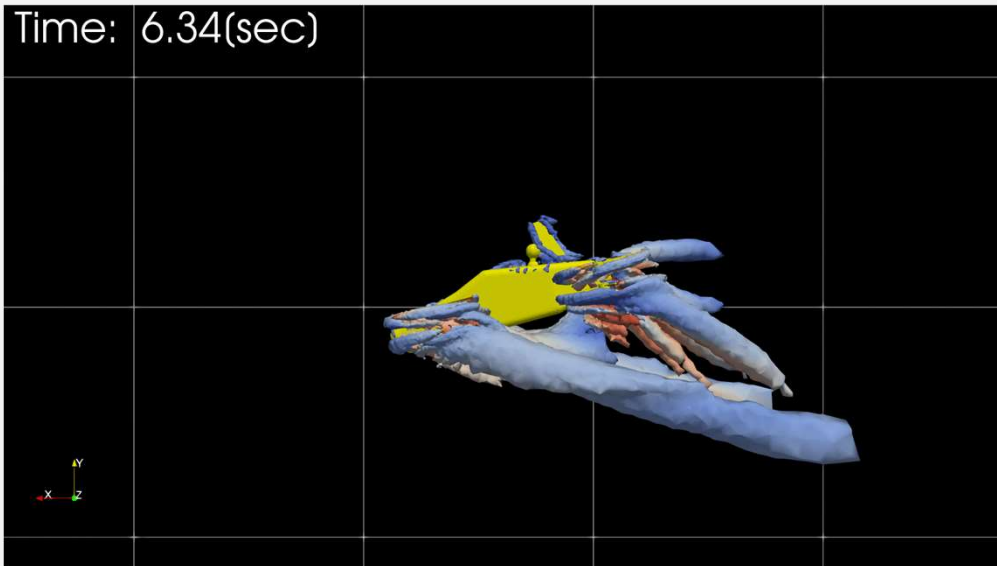
Time: 6.34(sec)



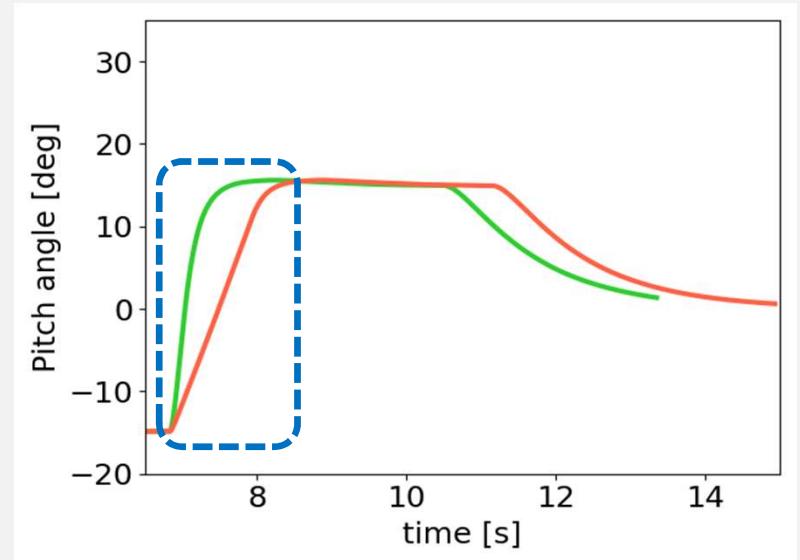
15 [deg] $\pi / 2$ [rad/s]

Velocity magnitude
0.0e+00 0.05 1.0e-01

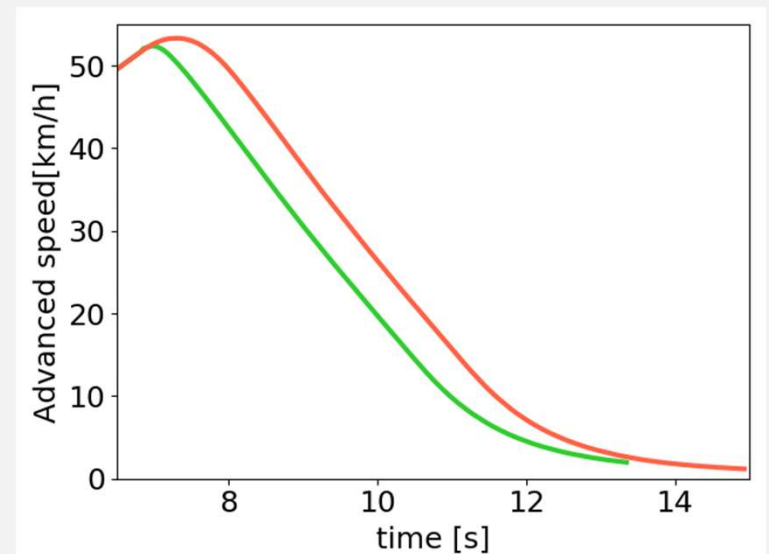
Time: 6.34(sec)



● 機首角度 [deg] ● $\pi/2$ ● $\pi/8$



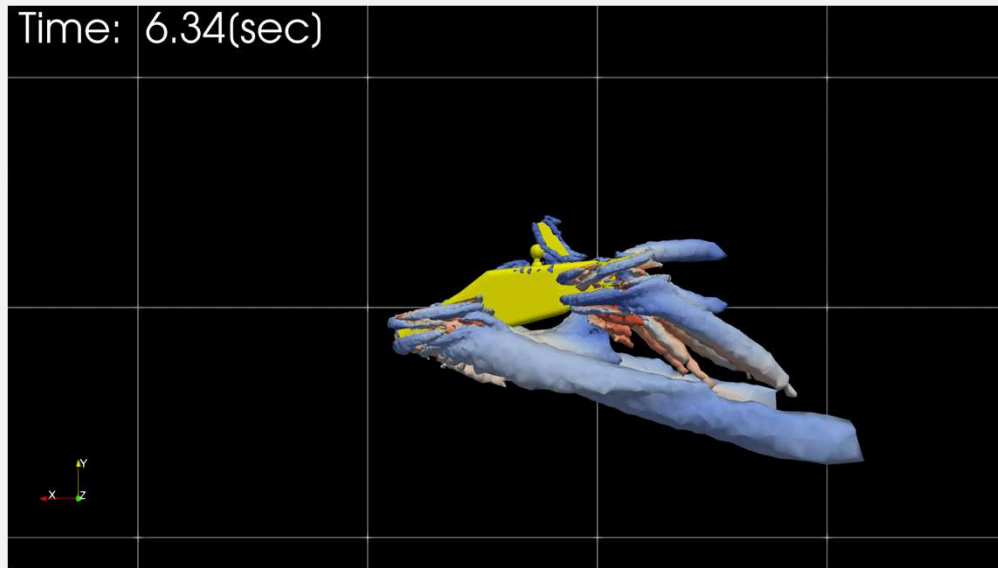
● 前進速度 [km/h] ● $\pi/2$ ● $\pi/8$



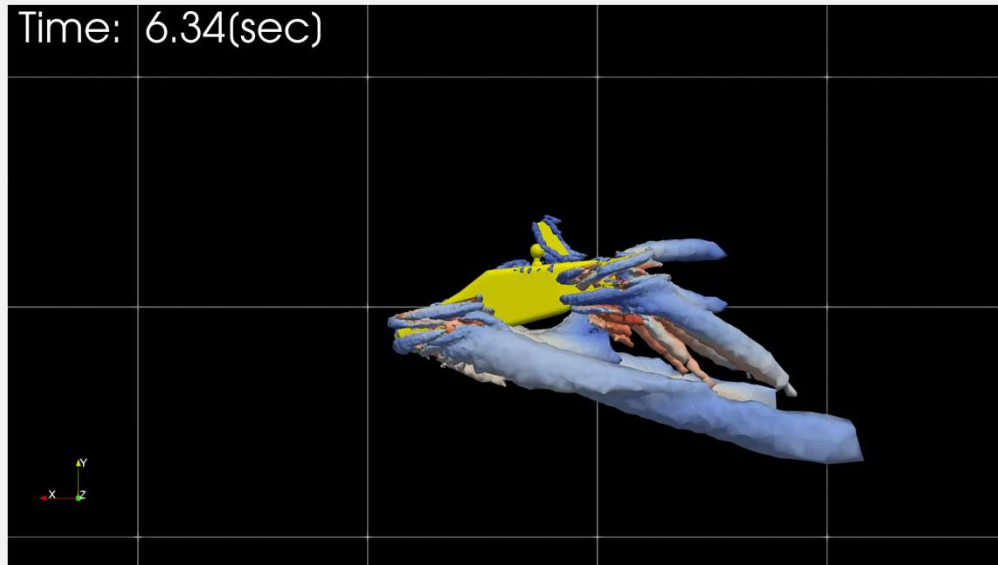
計算事例 (3)

角度変化による減速性能比較

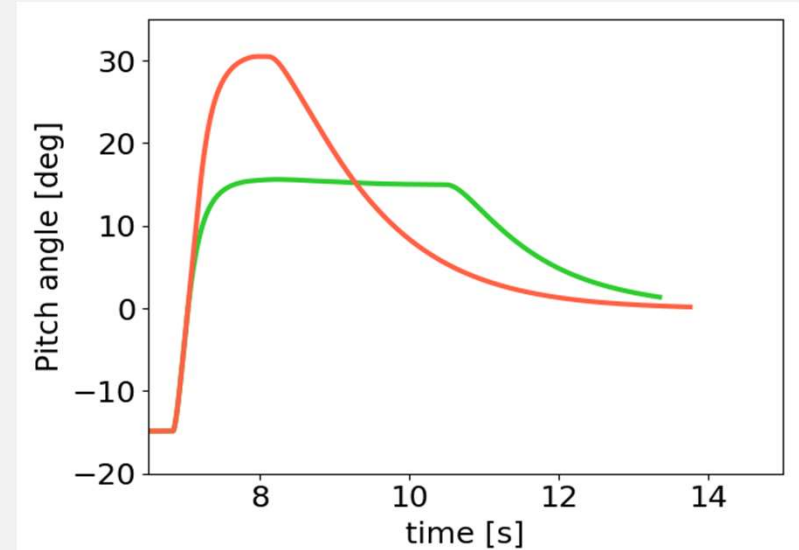
15 [deg] $\pi / 2$ [rad/s]



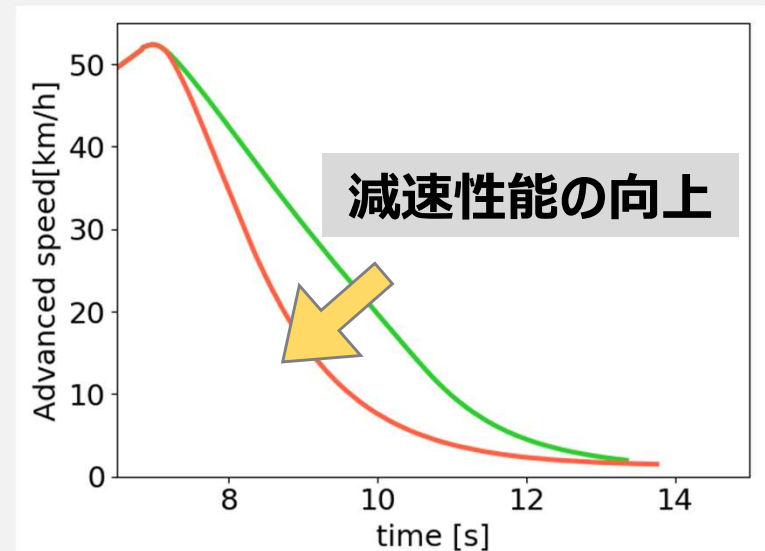
30 [deg] $\pi / 2$ [rad/s]



● 機首角度 [deg] ● 15 ● 30



● 前進速度 [km/h] ● 15 ● 30



高精度リアルタイムシミュレーションの概要

- ・ 計算速度を実時間以上に向上させることでリアルタイムシミュレーションを達成
- ・ 基本は**機械学習**による高速流体計算による

