小型二足ロボットのスケート運動

-RealSenseを用いた機体揺動角度の取得および蹴り動作の改善-

杉内研究室
19NA118　　貝原 輝

研究背景

人型二足ロボットの主な移動手段である二足歩行にはいくつかの問題点が存在する

問題点

- 移動速度が遅い
- エネルギー効率が悪い

解決策：スケート運動

- 歩くよりも移動速度が向上
- 単純な動作で移動できるので省エネルギー

ロボットでの安定したスケート運動の実現

研究課題

現在の課題

- 滑走時の転倒
- 意図しない方向への旋回

原因 の一つとして

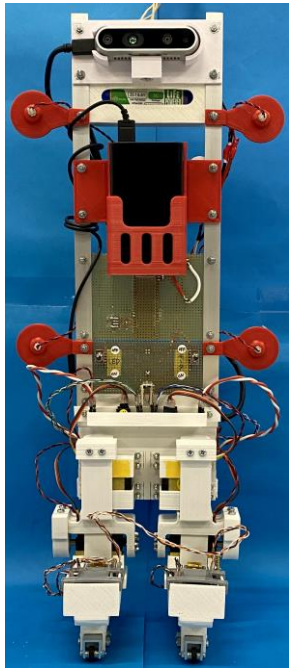
滑走時の機体の揺動を制御できていない

解決策

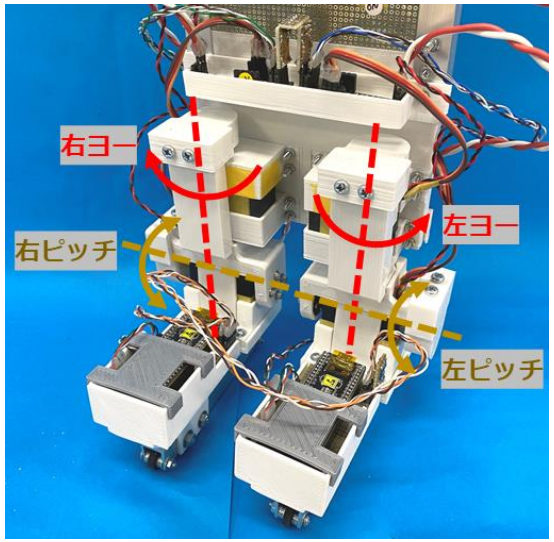
RealSenseを搭載して滑走時の機体の揺動角度を取得しその測定データをフィードバックとしてモータの蹴り角度を決定する蹴り動作

開発したロボット

Penguinkun4号機	
質量 [g]	2181
全長 [mm]	550
機体材質	アルミニウム、PLA樹脂
メインマイコン	Raspberry Pi4
コントローラマイコン	PSoC4、PSoC5LP
サーボモータ	SAVOX SC-0251MG
電源（マイコン）	Power Bank CBD/Q6 10000mAh
電源（モータ）	HYPERION Li-Fe 6.6V 2100mAh
RGB-Dカメラ	RealSense D435



Penguinkun4号機



各モータ位置と軸

RealSense D435

RealSenseを用いた角度測定方法（１）

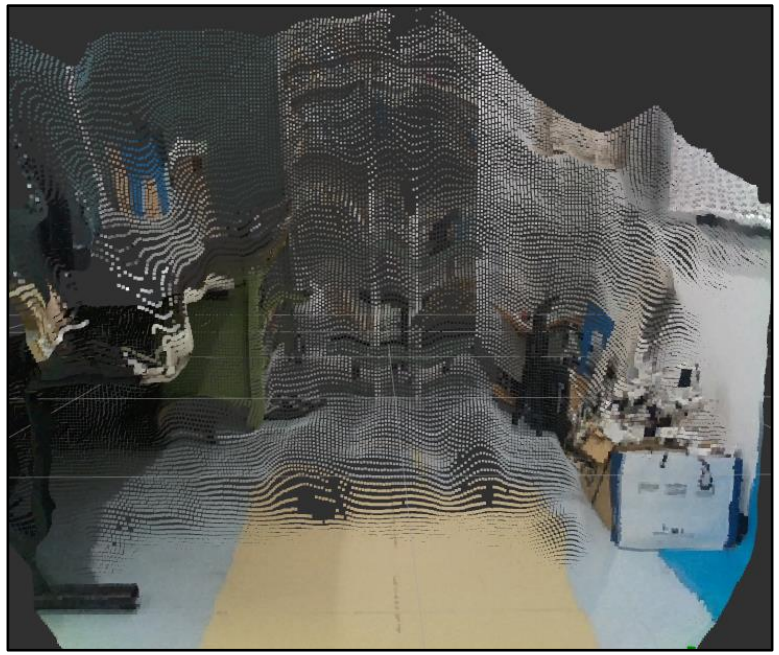
・RealSenseを用いた角度測定方法

⇒ 床の法線を推定し、滑走時に法線の傾きの変化から機体揺動角度を算出

手順①：PointCloudデータの取得

RealSenseで床を含む機体前方を撮影してPointCloudデータを取得する

PointCloudはPCL(Point Cloud Library)で扱うことができる

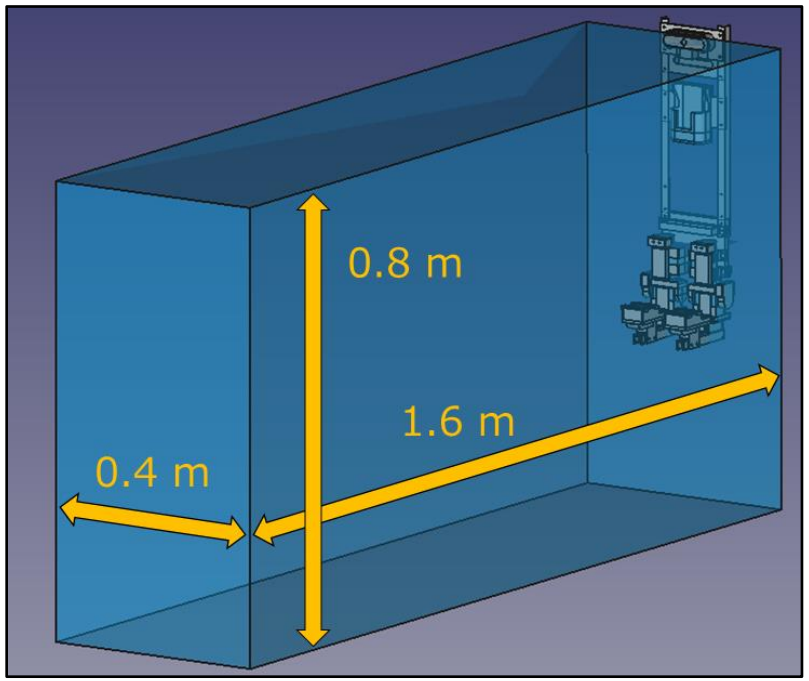


PointCloudデータ

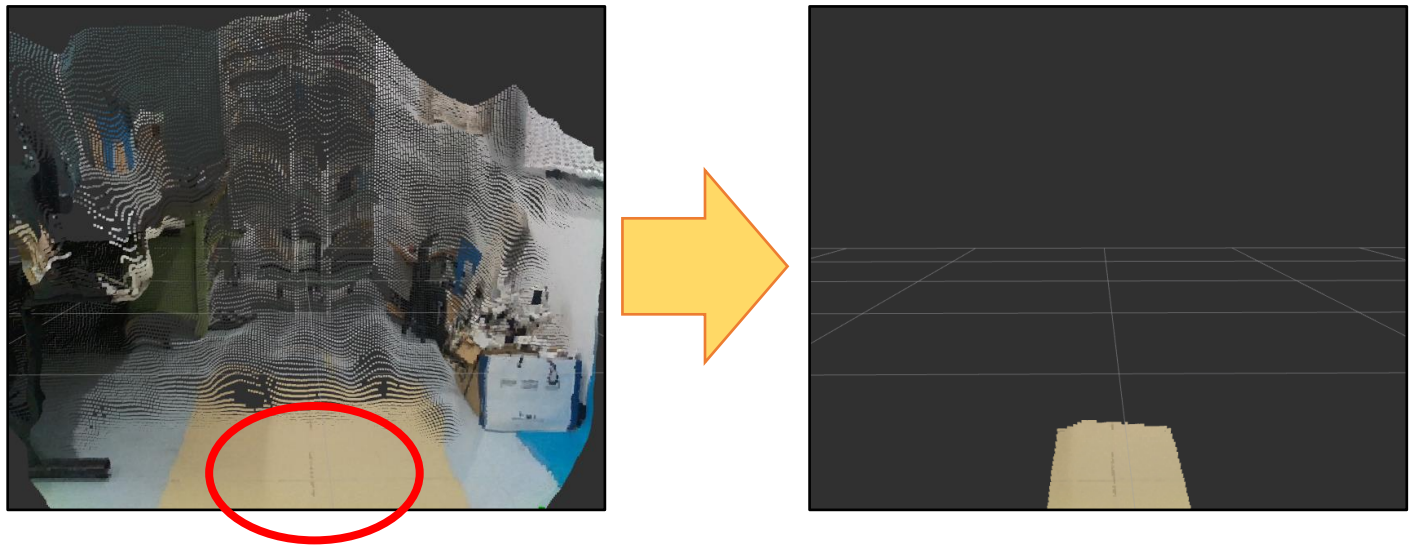
RealSenseを用いた角度測定方法（２）

手順②：PointCloudのフィルタリング処理

- PassThroughFilterでのフィルタリング
  設定した範囲外を除去して床の部分だけを残す



PassThroughFilterの範囲

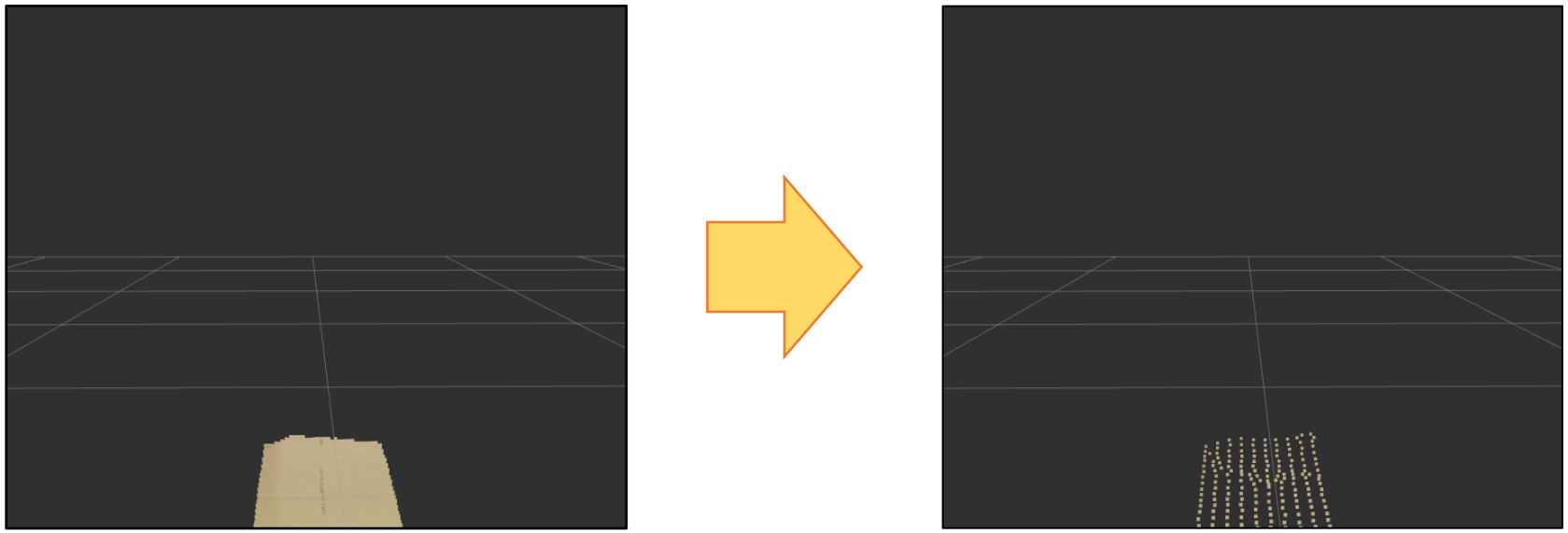


PassThroughFilterでの処理結果

RealSenseを用いた角度測定方法（３）

- VoxelGridFilterでのフィルタリング

  設定した間隔で点群を間引いてダウンサンプリングし
  データの軽量化を行う



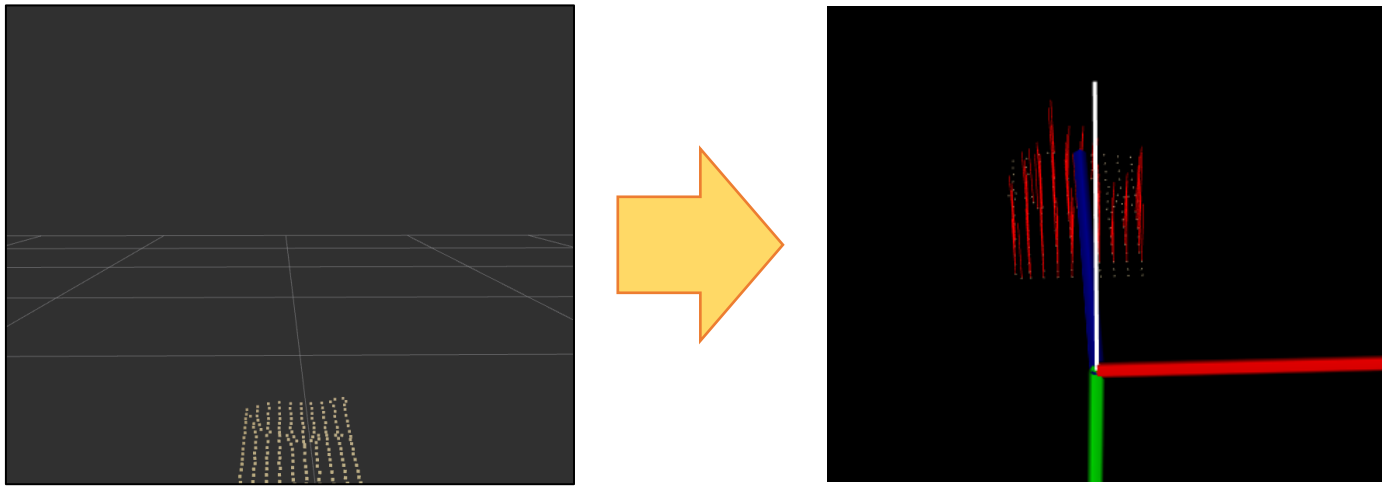
VoxelGridFilterでの処理結果

RealSenseを用いた角度測定方法（４）

手順③：PointCloudからの法線推定

PCLに実装されている法線推定ライブラリを用いて法線推定を行う

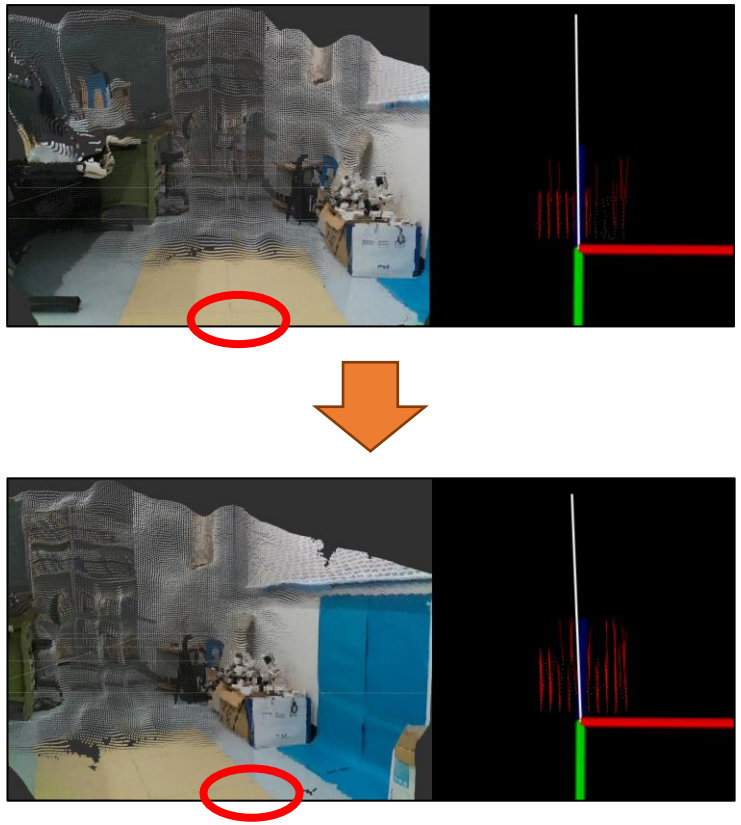
- Kd木という空間分割データ構造を用いてPointCloudの一つ一つの点に対して最近傍探索を行い、局所的な平面を作成
- 作成した局所平面の垂直方向を法線とする



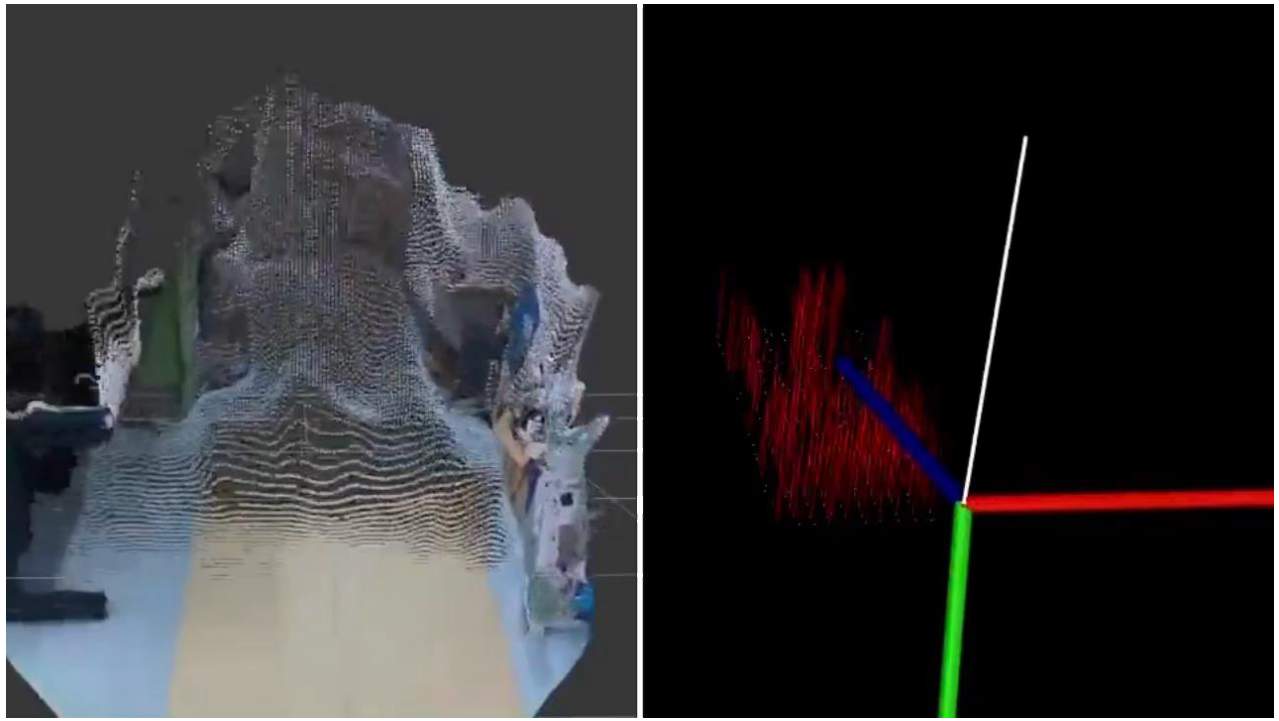
PCLでの法線推定結果

RealSenseを用いた角度測定方法の注意点

RealSenseがヨー軸まわりに回転した際は床の法線に変化が現れないため角度を算出することができない



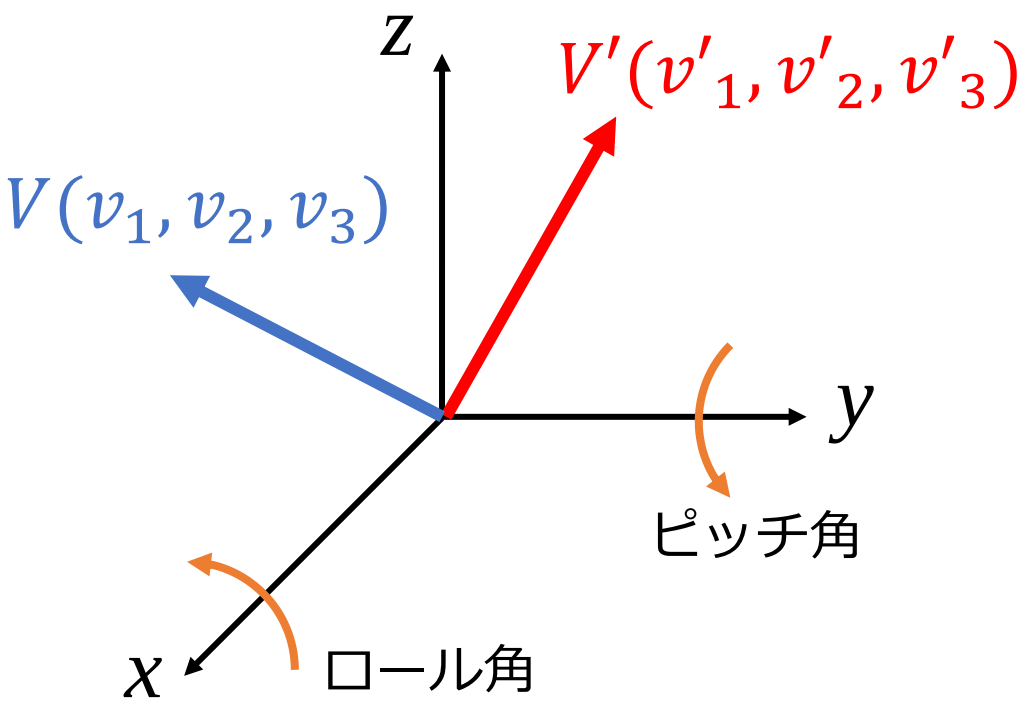
ヨー軸まわりに回転した様子



法線推定の様子

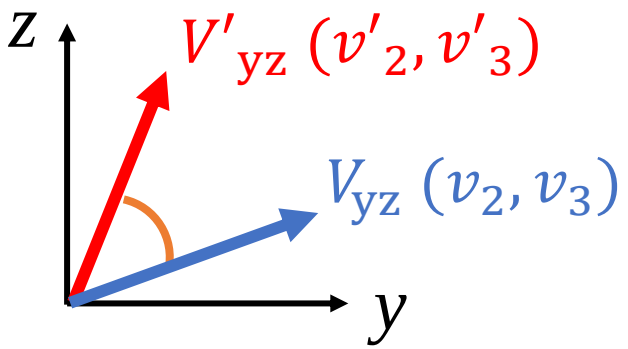
床法線からの角度算出方法

床の法線がVからV'へ変化



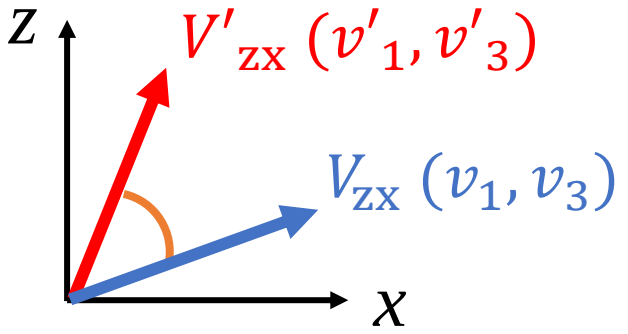
ロール角

VとV'をそれぞれyz平面に射影したベクトルV_{yz}とV'_{yz}間の角度を算出



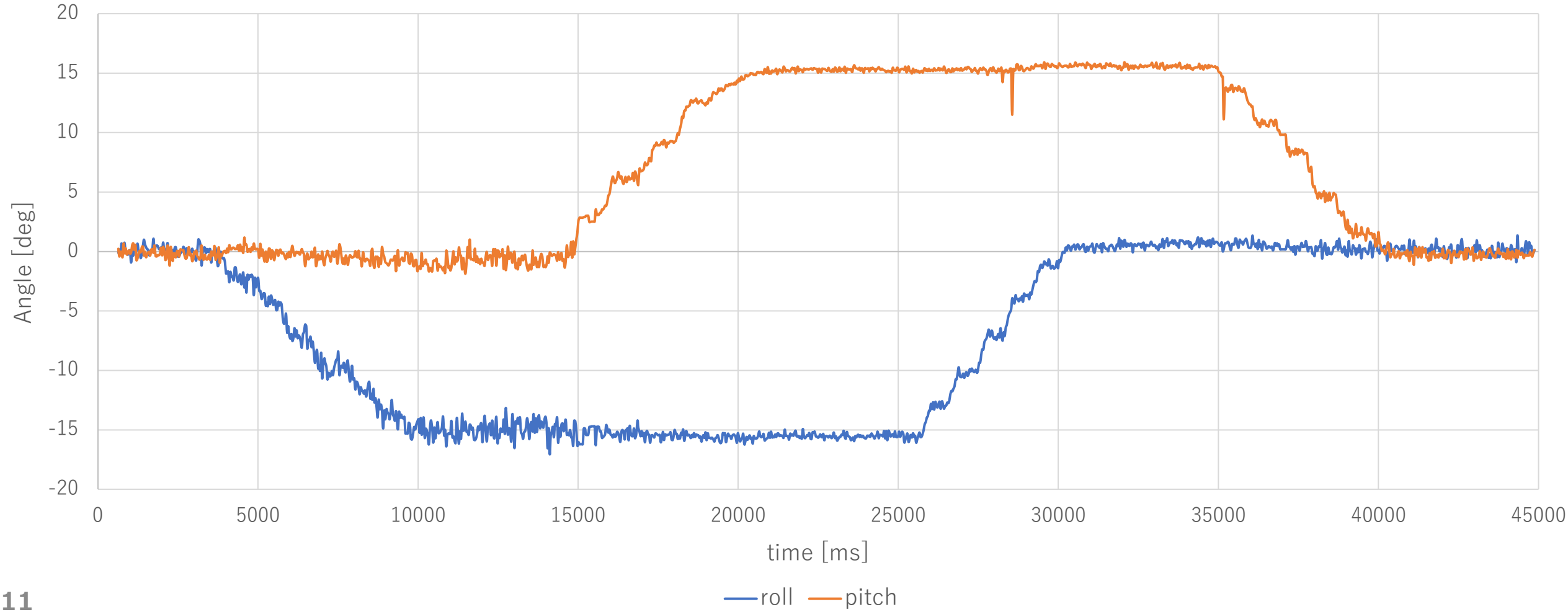
ピッチ角

VとV'をそれぞれzx平面に射影したベクトルV_{zx}とV'_{zx}間の角度を算出



RealSenseを用いた角度測定の精度

RealSenseを雲台に取り付けてロール角-15 deg、ピッチ角+15 degに動かしたときのグラフ

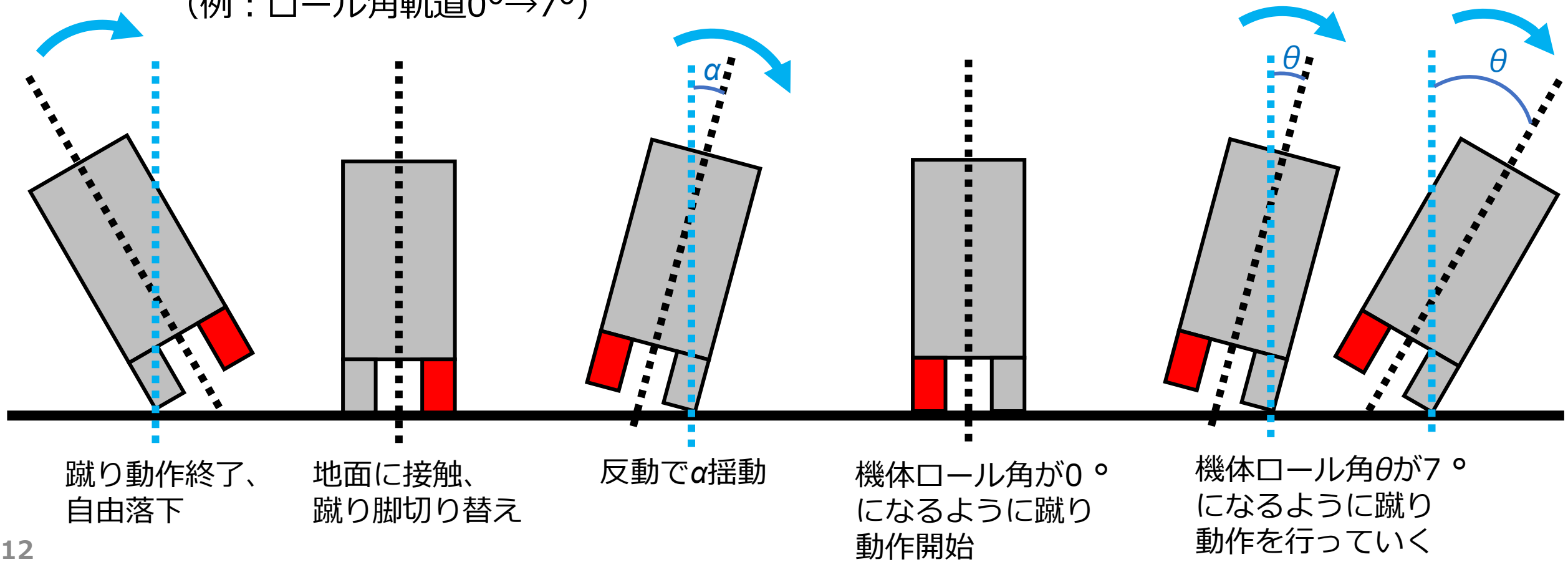


従来と本研究の蹴り動作の違い（１）

従来の蹴り動作　事前に機体ロール角軌道モデルを設定

➡ 設定したロール角軌道を機体が再現するように蹴り動作を行う

（例：ロール角軌道0°→7°）

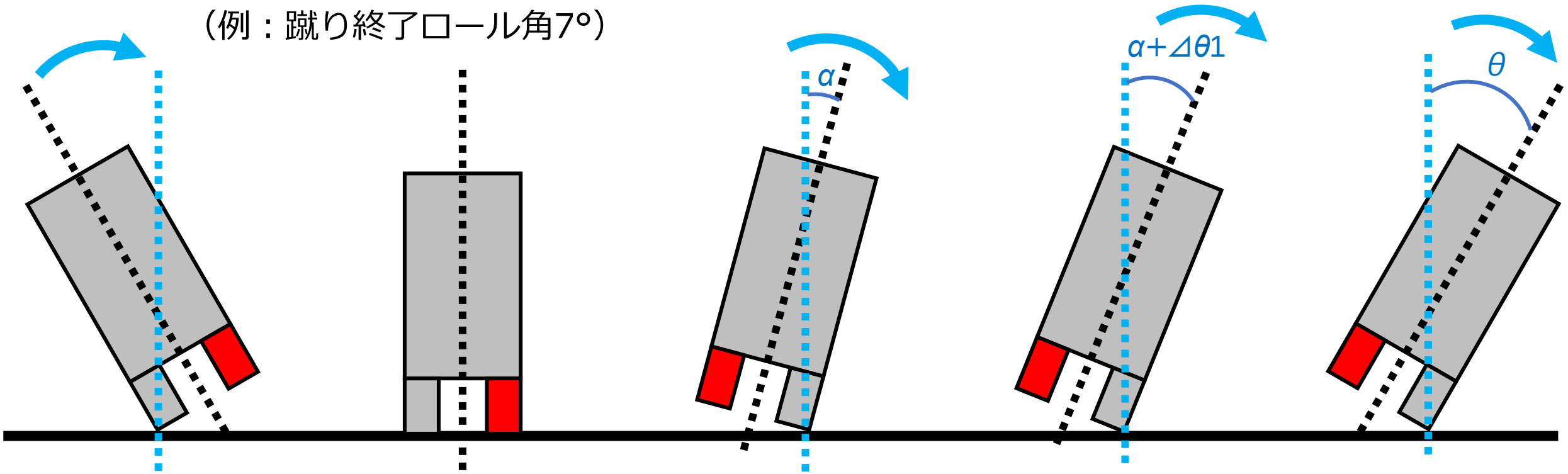


従来と本研究の蹴り動作の違い（２）

本研究の蹴り動作　滑走時の機体ロール角から蹴り角度を決定

➡ 機体ロール角が蹴り終了ロール角に到達すると蹴り動作を終了

（例：蹴り終了ロール角7°）



蹴り動作終了、
自由落下

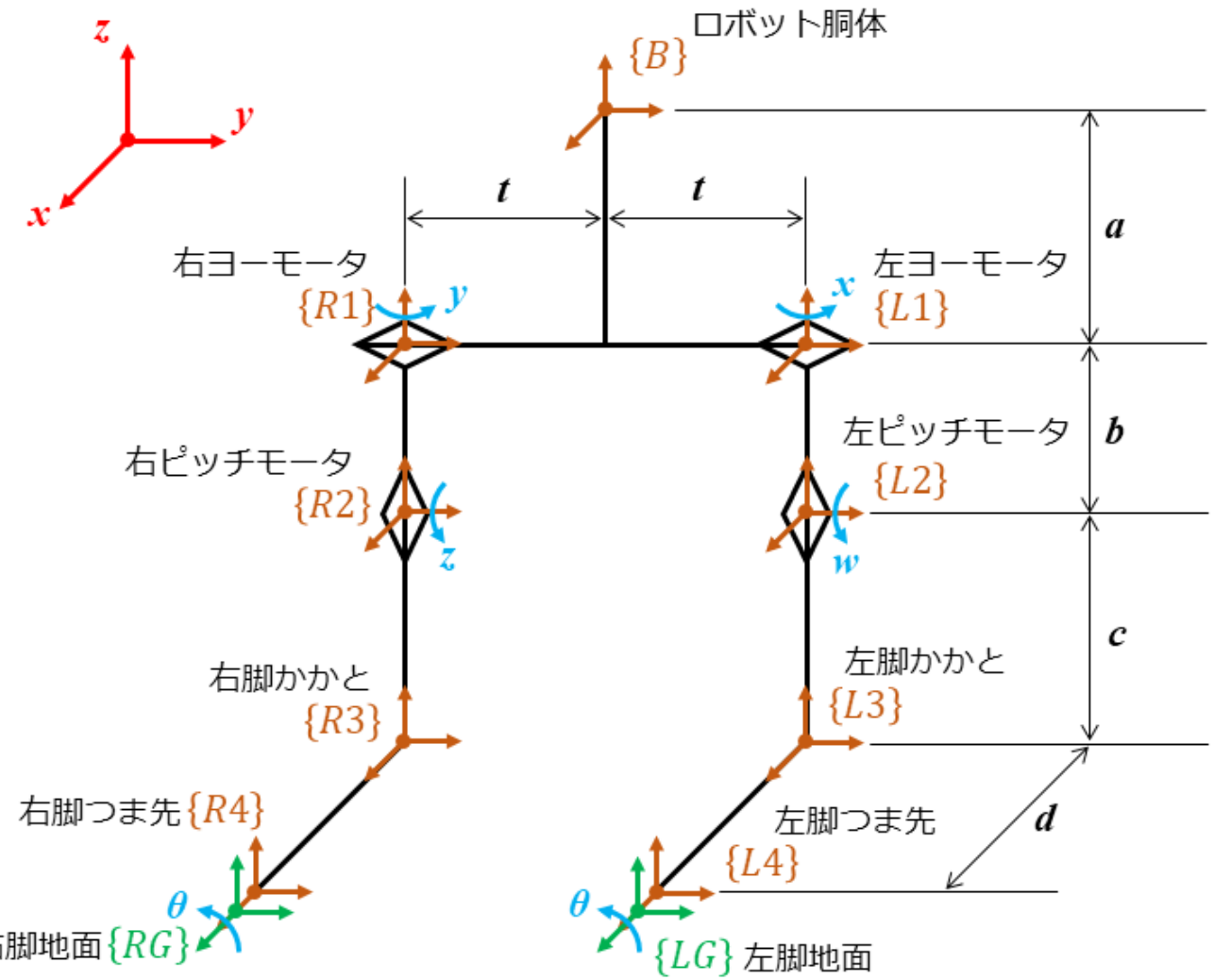
地面に接触、
蹴り脚切り替え

反動でα揺動、ここから機体ロール角がα+⊿θ1となるように蹴り動作を行う

蹴り動作を続けていき機体ロール角θが設定した蹴り終了ロール角(7°)に到達すると蹴り動作終了

蹴りピッチ角度の計算方法（１）

ロボットのリンク座標系



支持脚ロール角から蹴り脚の蹴りピッチ角度を決定

右脚蹴りの場合

$$ {}^{LG}_{R4}\boldsymbol{T} = \begin{bmatrix} & & & p_x \\ & \boldsymbol{R} & & p_y \\ & & & p_z \\ 0 & 0 & 0 & 1 \end{bmatrix} $$

蹴り動作時に蹴り脚つま先が常に地面と接触する蹴り動作

$$ p_z = 0 $$

蹴りピッチ角度の計算方法（２）

右脚蹴りの場合の $p_z = 0$

$$\sin z = -\frac{\cos z[d\{(\sin x\cos y - \cos x\sin y)\sin\theta - \sin w(\sin x\sin y + \cos x\cos y)\cos\theta\} + c\cos w\cos\theta] + 2t(\cos x\sin\theta + \sin w\sin x\cos\theta) - c\cos\theta}{c\{(\cos x\sin y - \sin x\cos y)\sin\theta + \sin w(\sin x\sin y + \cos x\cos y)\cos\theta\} + d\cos w\cos\theta}$$

この式を整理すると蹴り脚(右脚)ピッチ角度zは

$$z = \mathrm{atan2}(\sin z, \cos z)$$

変数として支持脚(左脚) ロール角θ、支持脚ピッチ角w、
支持脚ヨー角x、蹴り脚ヨー角yが含まれる

として求めることができる

RealSenseの動作周波数

RealSenseの動作周波数は使用環境によって異なる

- Raspberry Pi4に接続した状態では30 Hz動作
- 高性能な外部PCに接続した状態では60 Hz動作

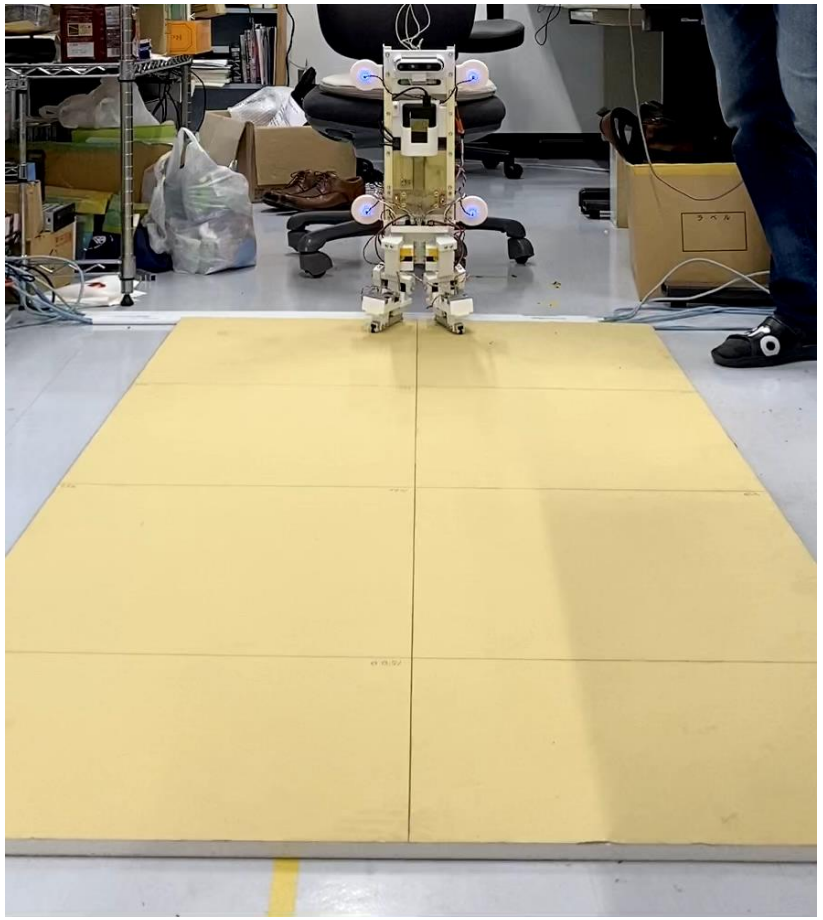
60 Hz動作で本研究の蹴り動作が実現可能か検証

↓

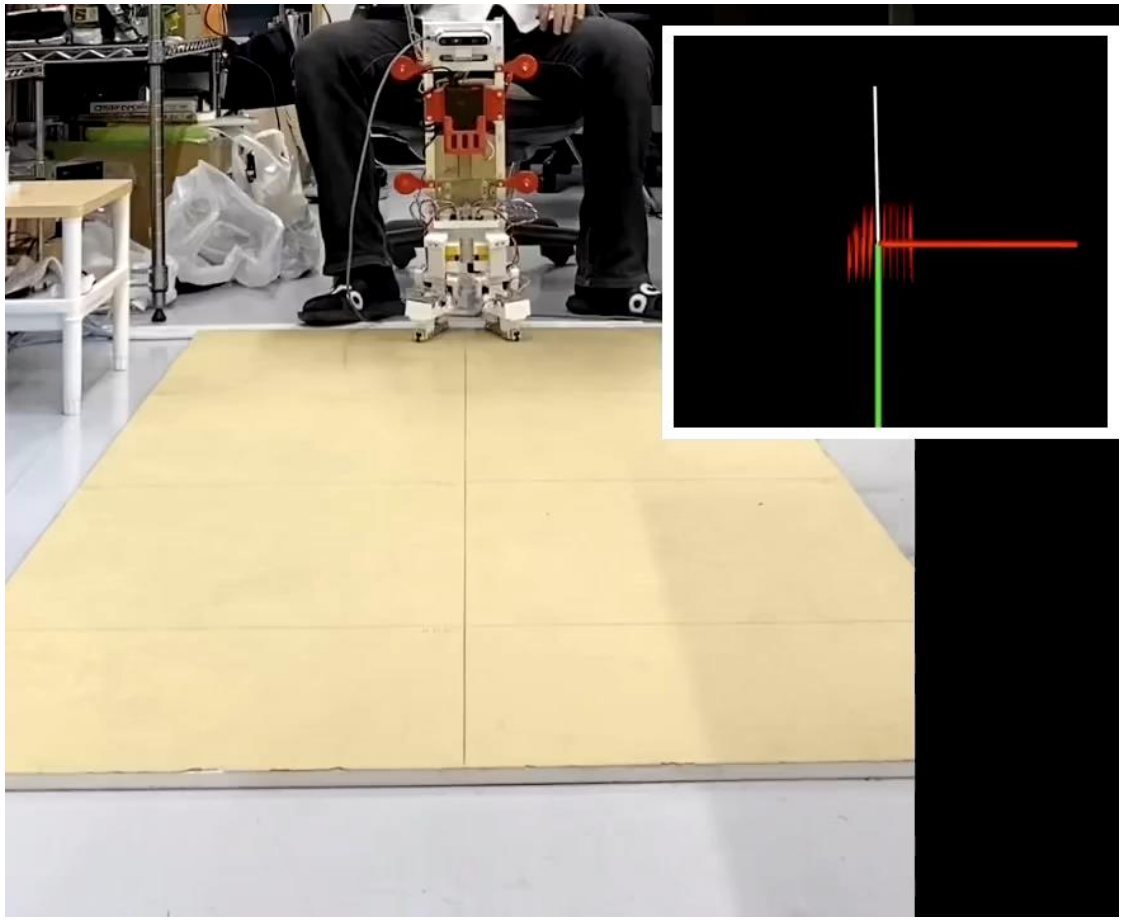
30 Hz動作での滑走実験

60 Hz動作での滑走実験（１）

滑走の様子の比較

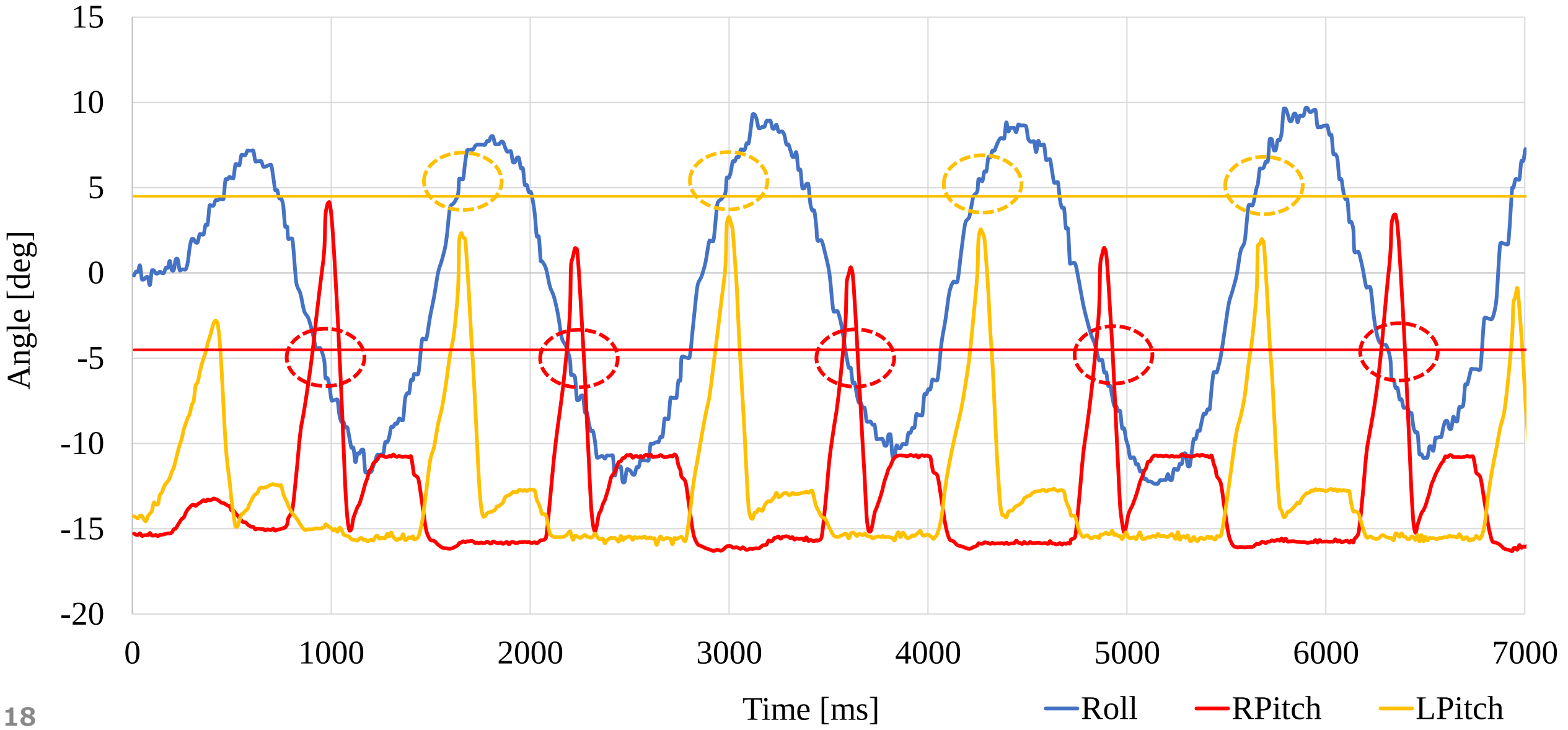


従来の蹴り動作

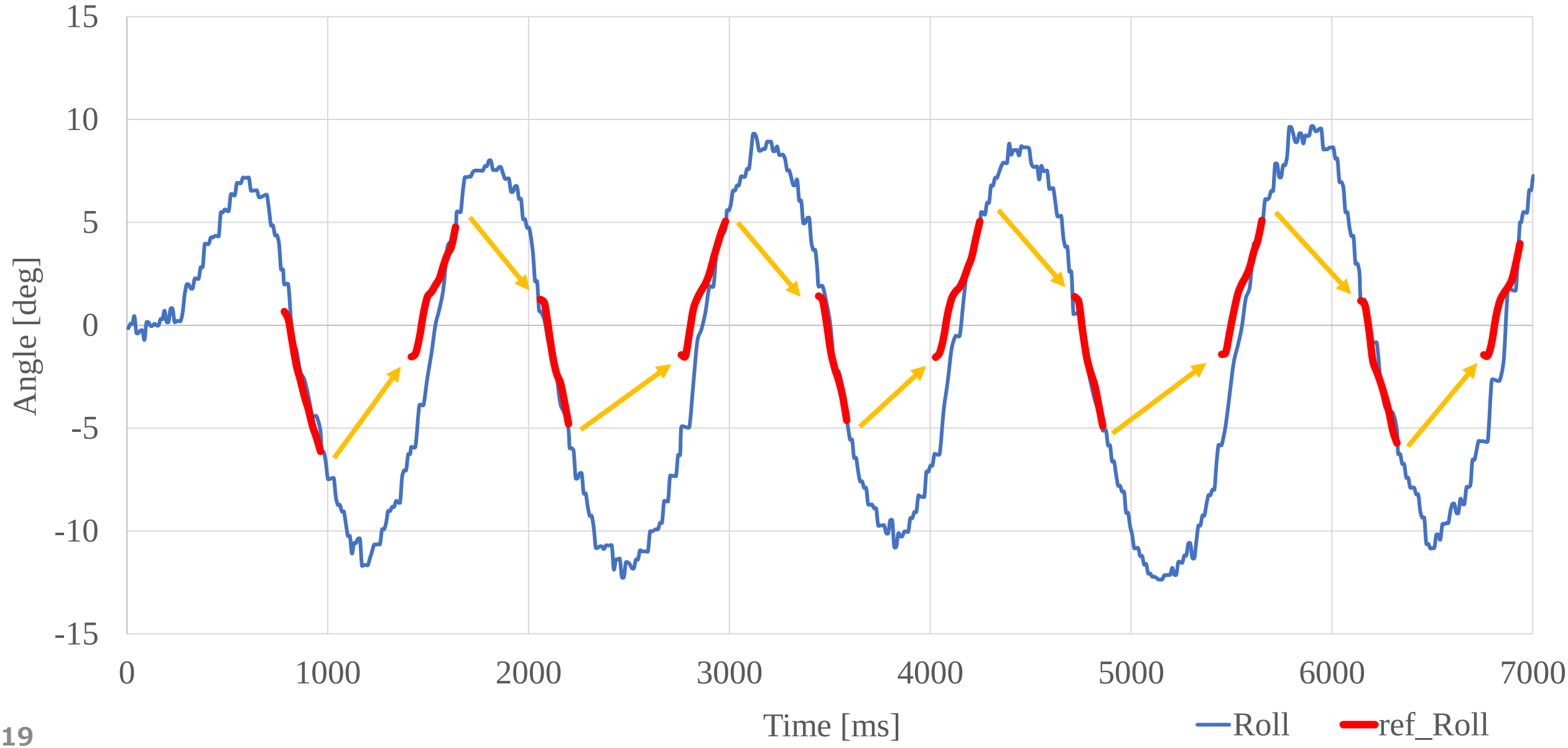


本研究で考案した蹴り動作(60 Hz動作)

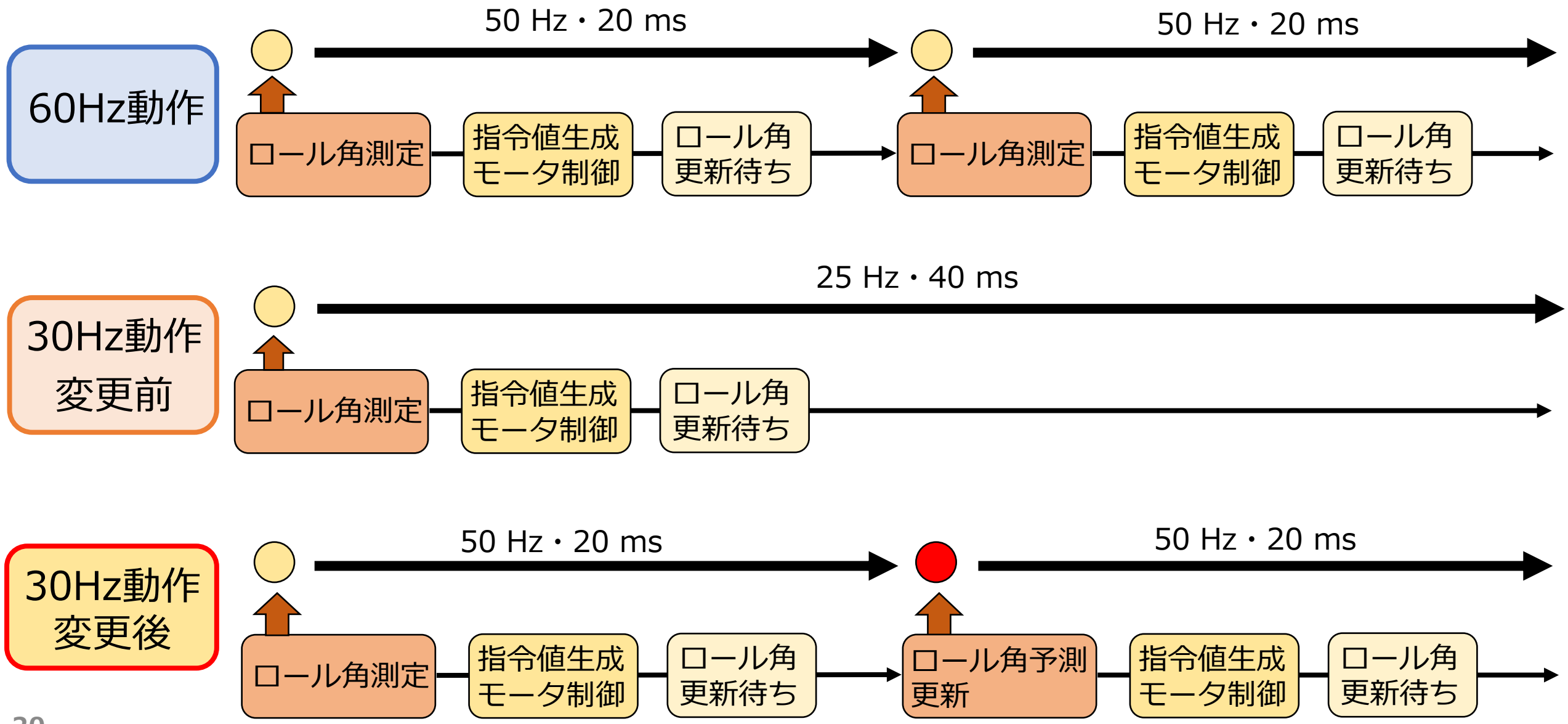
60 Hz動作での滑走実験（２）



60 Hz動作での滑走実験（３）

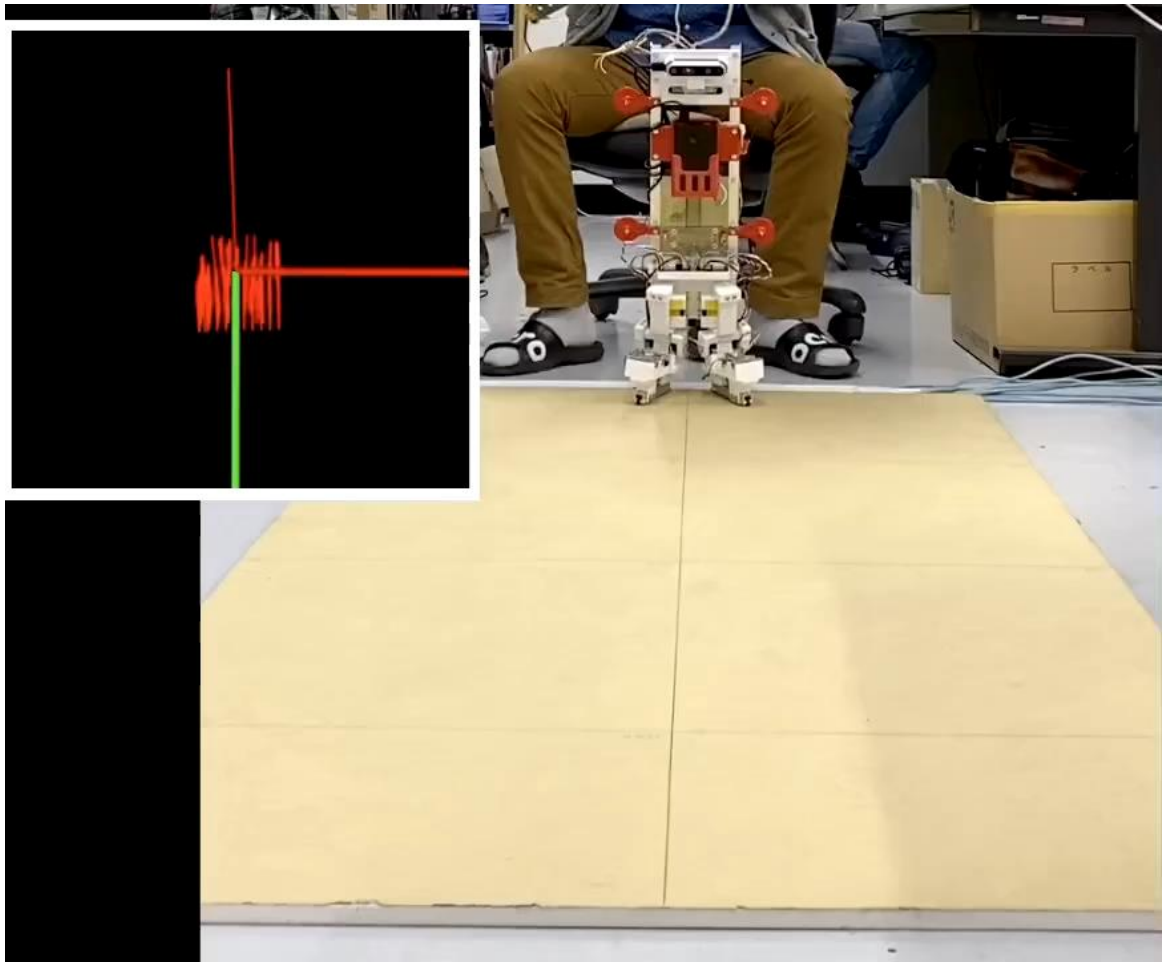


30 Hz動作でのプログラム変更点



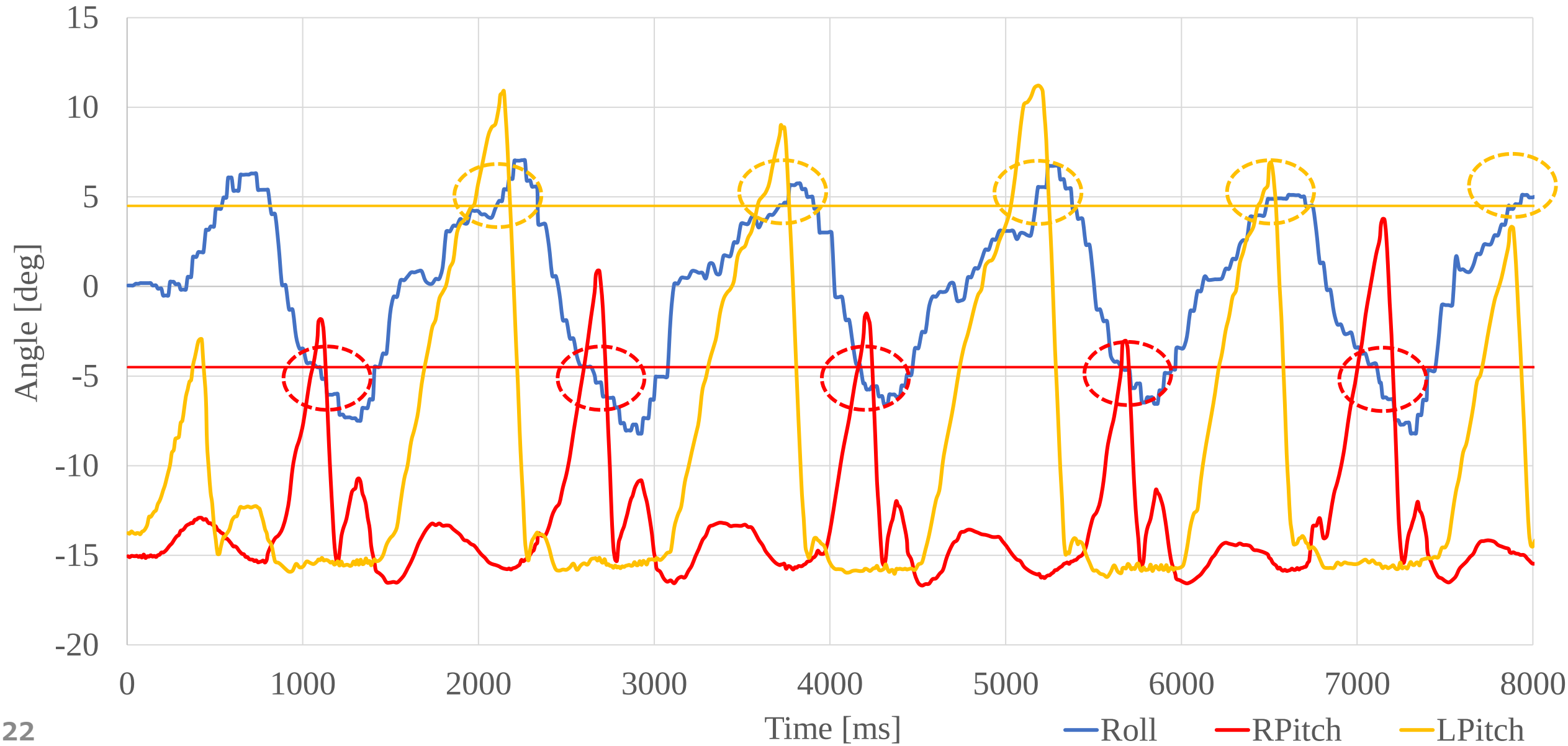
30 Hz動作での滑走実験（１）

滑走の様子

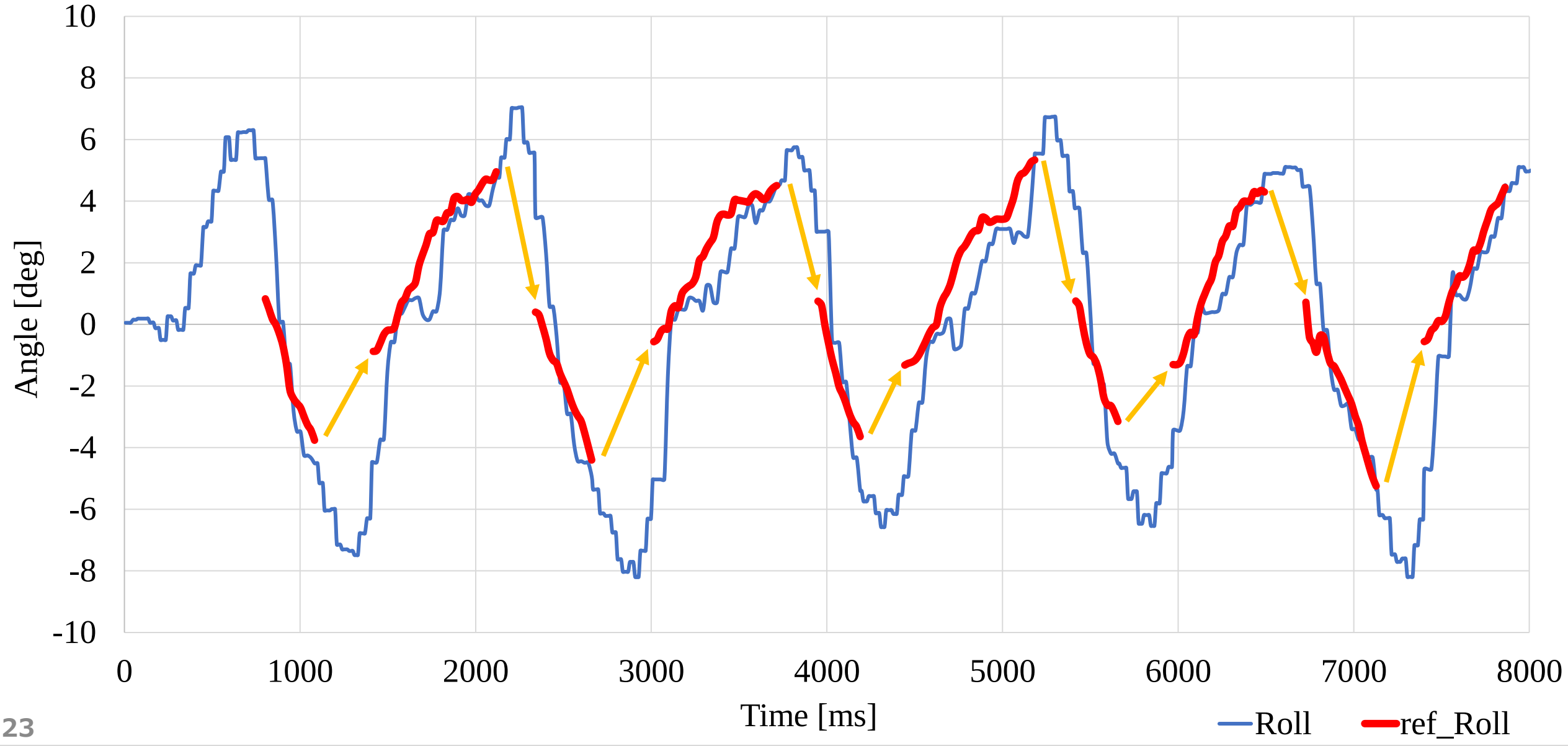


本研究で考案した蹴り動作(30 Hz動作)

30 Hz動作での滑走実験（２）



30 Hz動作での滑走実験（３）



結論

- 本研究で新たに考案した蹴り動作では従来よりも滑走が滑らかになり、滑走時の揺動エネルギーを利用して効率よく揺動運動を持続できるようになったことで滑走距離も長くなった

- 考案した蹴り動作では滑走時の機体ロール角を制御するにはまだ十分ではなく、より正確に制御するためには機体ロール角だけでなく機体ロール角速度も考慮して蹴りモータ指令値を生成する必要がある