モーションシミュレータ 仕様書

1. 動機

モーション制御に関する組み込みシステムの開発に携わっています。
「制御理論から実機械への応用まで」を、きちんと実践できるようにしたい。
そのための学習として、このモーションシミュレータを作成しました。

2. コンセプト

組み込み制御システムのフローに基づく、離散時系列の制御シミュレータ。
（指令計算 ⇒ 操作量計算 ⇒ 物体の運動変化 ⇒ フィードバック）

2.1 特徴

1. 「MotionFlow（モーション制御フロー）」クラスを中心に概念整理した、現実に即した分かりやすいオブジェクト指向設計
2. カスタマイズ可能（JSONファイルでパラメータ設定）
3. 拡張可能（系統的に設計したPythonパッケージ構成とバージョン管理）

2.2 用途

1. 組み込み制御システムの検証
2. 制御工学の学習

3. 使用例

Jupyter Notebookで実行する例を示す。

3.1 シミュレーション実行

```
import pandas as pd          # パッケージのインポート（pandas 表形式データ処理）
import tkmotion as tkm       # パッケージのインポート（モーションシミュレータ）

# モーション制御フローのインスタンスを作成
flow = tkm.MotionFlow()
# 離散時系列の設定をロード（デフォルト Δt=100[μs]、継続時間=6[s]）
flow.load_discrete_time()
# モーションプロファイルの設定をロード（台形速度）
flow.load_motion_profile(filepath="tkmotion/prof/default_motion_prof_config.json",
```

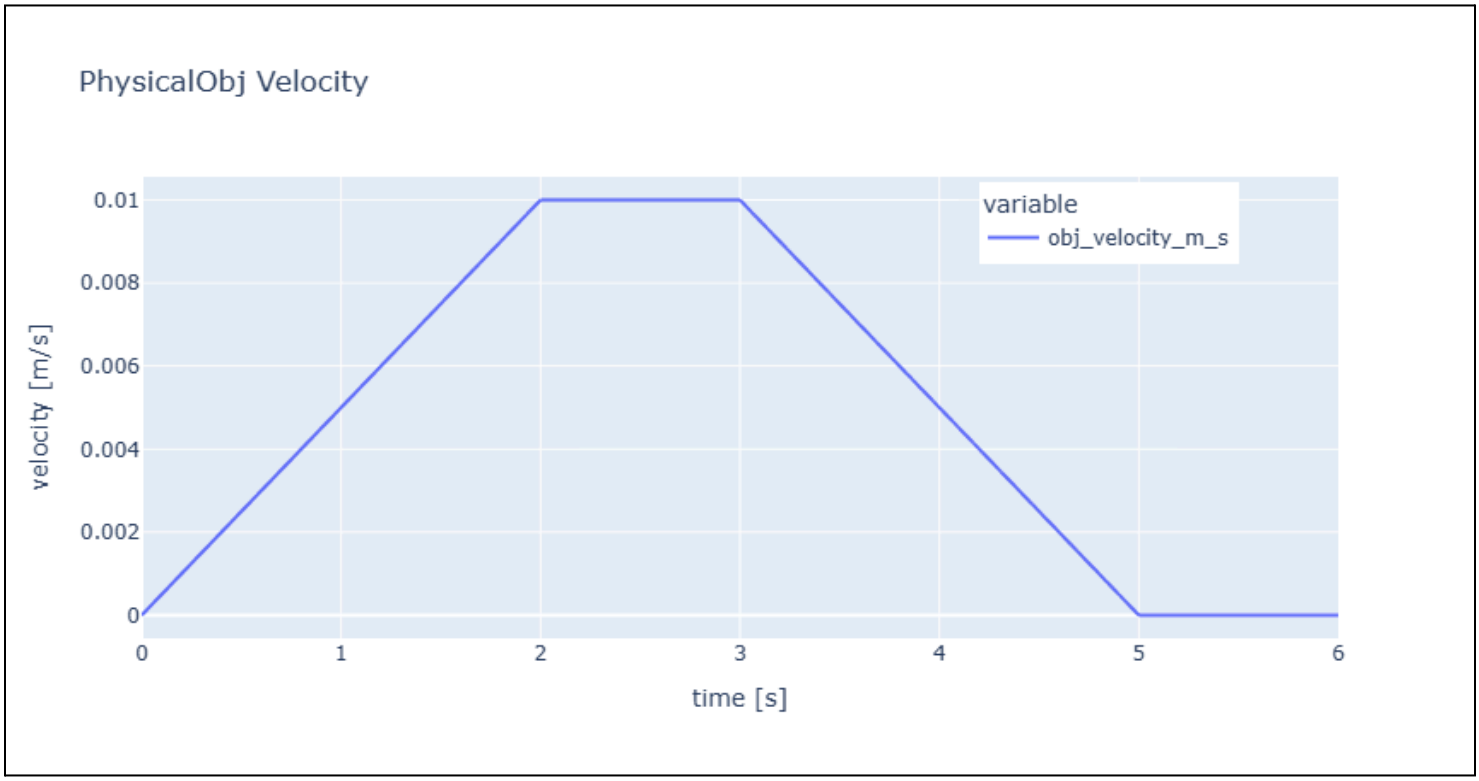
```
prof_index=1)
# コントローラの設定をロード（PIDコントローラ）
flow.load_controller(filepath="tkmotion/ctrl/default_controller_config.json",
ctrl_index=1)
# プラントの設定をロード（デフォルト 質量を持つ物体）
flow.load_plant(filepath="tkmotion/plant/default_plant_config.json", plant_index=0,
phyobj_index=0)
# 制御フローを実行。結果をpandas.DataFrame形式で得る
df = flow.execute()
```

3.2 実行結果をグラフで確認

3.2.1 物体の速度変化

DataFrameから「経過時間 "time_s"」と「物体速度 "obj_velocity_m_s"」を取り出し、
グラフ描画パッケージ plotly.express でlineプロットを作成する。

```
import plotly.express as px
fig = px.line(df, x="time_s", y=["obj_velocity_m_s"], title="PhysicalObj Velocity")
fig.update_layout(
    xaxis=dict(title="time [s]"),
    yaxis=dict(title="velocity [m/s]"),
    legend=dict(yanchor="top", y=0.99, xanchor="left", x=0.7))
fig.show()
```

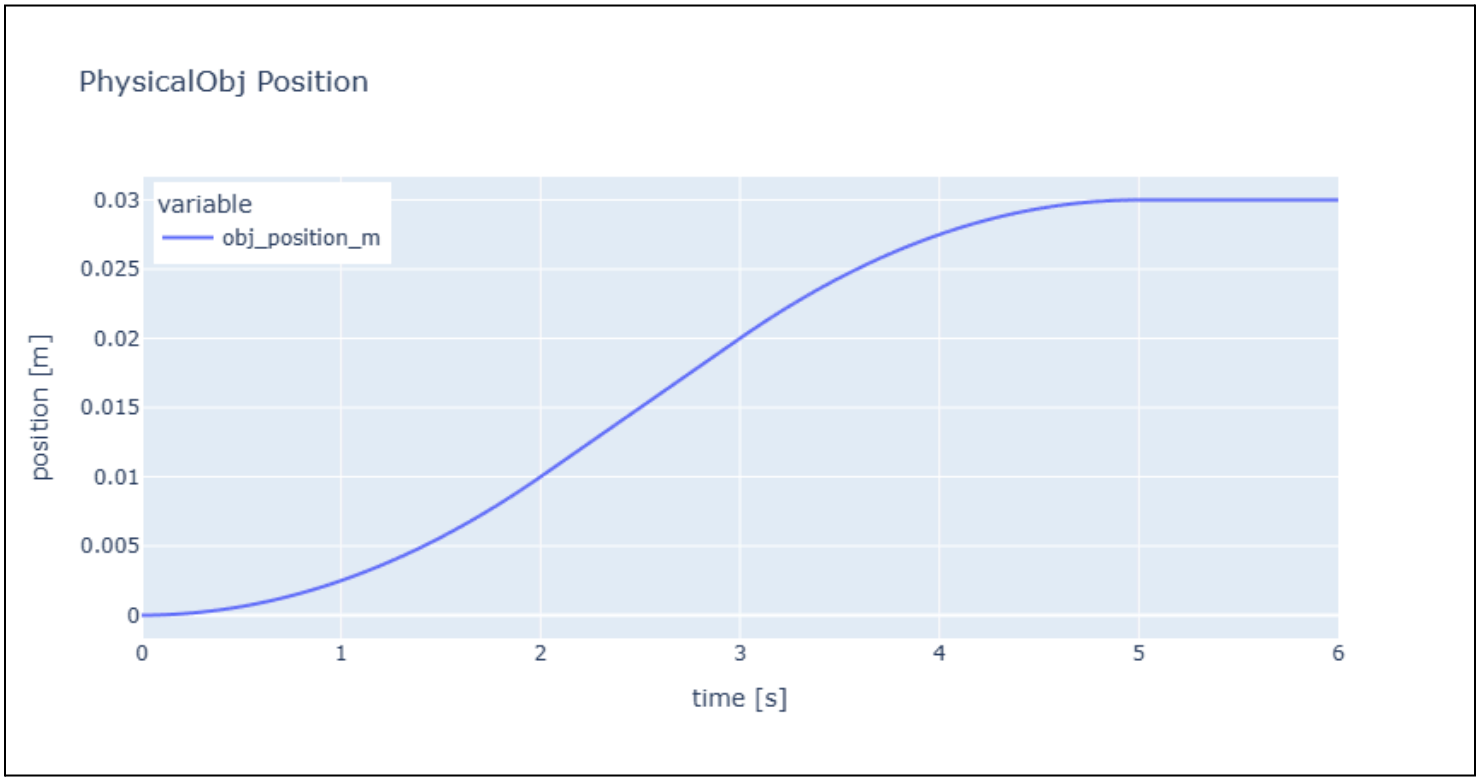


3.2.2 物体の位置変化

DataFrameから「経過時間 "time_s"」と「物体位置 "obj_position_m"」を取り出し、

グラフ描画パッケージ plotly.express でlineプロットを作成する。

```
import plotly.express as px
fig = px.line(df, x="time_s", y=["obj_position_m"], title="PhysicalObj Position")
fig.update_layout(
    xaxis=dict(title="time [s]"),
    yaxis=dict(title="position [m]"),
    legend=dict(yanchor="top", y=0.99, xanchor="left", x=0.01))
fig.show()
```



4. 設計

4.1 Pythonパッケージ（ディレクトリ）とモジュール（.pyファイル）の構成

```
tkmotion/
      __init__.py  # ディレクトリをPythonパッケージとして扱わせるためのスクリプトファイル
      flow/  # motion flow / モーション制御フロー
            __init__.py
            motion_flow.py
      time/  # time series / 時系列
            __init__.py
            discrete_time.py
            default_discrete_time.json  # 離散時系列 デフォルト設定
      prof/  # motion profile / モーションプロファイル
```

```
        __init__.py
        motion_profile.py
        default_motion_prof.json  # モーションプロファイル デフォルト設定
    ctrl/  # motion control / モーション制御
        __init__.py
        controller.py
        default_controller.json  # コントローラ デフォルト設定
    plant/  # plant (control target) / プラント (制御対象)
        __init__.py
        plant.py
        physical_object.py
        default_plant.json  # プラント デフォルト設定
    util/  # utility / ユーティリティ
        __init__.py
        utility.py
```

4.2 モジュールとクラス

4.2.1 パッケージ: `tkmotion.flow`

motion flow / モーションフロー

モジュール	ver	クラス	責務	主要特性
motion_flow.py	0.2.0	MotionFlow (モーション制御フロー)	モーション指令の流れを司り、 シミュレーションを実行	load_discrete_time() load_motion_profile() load_controller() load_plant() execute()

MotionFlow.execute() の戻り値は、シミュレーション結果のpandas.DataFrame

```
result_df = pd.DataFrame(
    {
        "time_s": time_list,                    # 経過時間 [s]
        "cmd_velocity_m_s": cmd_vel_list,       # 指令速度 [m/s]
        "cmd_position_m": cmd_pos_list,         # 指令位置 [m]
        "velocity_error_m_s": vel_error_list,   # 速度偏差 [m/s] (指令速度 - プラント速度)
        "position_error_m": pos_error_list,     # 位置偏差 [m] (指令位置 - プラント位置)
        "force_N": force_list,                  # 操作量(推力) [N]
        "obj_acceleration_m_s2": obj_acc_list,  # プラント加速度 [m/s^2]
        "obj_velocity_m_s": obj_vel_list,       # プラント速度 [m/s]
        "obj_position_m": obj_pos_list,         # プラント位置 [m]
    }
)
```

4.2.2 パッケージ: tkmotion.time

time series / 時系列

モジュール	ver	クラス	責務	主要特性
discrete_time.py	0.2.0	DiscreteTime (離散時間)	間隔Δtの離散時系列を表現	get_time_step_generator()
↑	↑	DiscreteTimeLoader (離散時間ローダ)	設定 (jsonファイル) を読み込み、 DiscreteTimeクラスのインスタンスを生成	load()

default_discrete_time.json # 離散時系列 デフォルト設定

```
[  # 拡張性を考慮し、リスト [] と 辞書 {} を組み合わせる
    {
        "discrete_time": [
            {
                "version": "0.1.0",   # 設定バージョン (major.minor.patch)
                "time_step_us": 100,  # 時間ステップ [μs]
                "duration_s": 6.0     # 継続時間 [s]
            }
        ]
    }
]
```

4.2.3 パッケージ: tkmotion.prof

motion profile / モーションプロファイル

モジュール	ver	クラス	責務	主要特性
motion_profile.py	0.2.0	MotionProfile (モーションプロファイル)	モーションプロファイルのベースクラス。 モーションプロファイルの共通特性を定義	calculate_cmd_vel_pos()
↑	↑	TrapezoidalMotionProfile (台形モーションプロファイル)	台形状に変化する速度(加速・定速・減速)から 指令速度・指令位置を計算する	↑
↑	↑	ImpulseMotionProfile (インパルスモーションプロファイル)	インパルス状の指令値を出力する	↑
↑	↑	StepMotionProfile (ステップモーションプロファイル)	ステップ状の指令値を出力する	↑
↑	↑	MotionProfileLoader (モーションプロファイルローダ)	設定 (jsonファイル) を読み込み、 MotionProfileクラスのインスタンスを生成	load()
↑	↑	VelocityZeroOrMinusError (速度ゼロ・マイナスエラー)	設定速度がゼロ、またはマイナスの例外を表す	-
↑	↑	AccelerationZeroOrMinusError (加速度ゼロ・マイナスエラー)	設定加速度がゼロ、またはマイナスの例外を表す	-
↑	↑	MovingLengthZeroError	設定移動量がゼロの例外を表す	-

		(移動距離ゼロエラー)		

default_motion_prof.json # モーションプロファイル デフォルト設定

```
[  # 拡張性を考慮し、リスト [] と 辞書 {} を組み合わせる
    {
        "motion_profile": [
            {
                "version": "0.1.0",          # 設定バージョン (major.minor.patch)
                "type": "trapezoid",         # モーションプロファイルの種類
                "max_velocity_m_s": 0.01,    # 最大速度 (定速度) [m/s]
                "acceleration_m_s2": 0.005,  # 加速度 [m/s^2]
                "length_m": 0.03             # 移動距離 [m]
            },
            {
                "version": "0.2.0",
                "type": "default"  # デフォルトプロファイル (テスト用)
            },
            {
                "version": "0.2.0",
                "type": "impulse",
                "impulse_vel_m_s": 1000,          # インパルス指令速度 [m/s] (テスト値)
                "impulse_pos_m": 1000,            # インパルス指令位置 [m] (テスト値)
                "impulse_on_timestep_count": 1,   # インパルスON タイムステップ数 [-]
                "delay_s": 1.0                    # 遅延時間 [s]
            },
            {
                "version": "0.2.0",
                "type": "step",
                "step_velocity_m_s": 0.02,  # ステップ指令速度 [m/s] (テスト値)
                "step_position_m": 0.01,    # ステップ指令速度 [m] (テスト値)
                "delay_s": 1.0              # 遅延時間 [s]
            }
        ]
    }
]
```

4.2.4 パッケージ: tkmotion.ctrl

motion control / モーション制御

モジュール	ver	クラス	責務	主要特性
controller.py	0.2.0	Controller (コントローラ)	コントローラのベースクラス。 コントローラの共通特性を定義	reset() calculate_force()
↑	↑	PIDController (PIDコントローラ)	PID計算を実行するコントローラ ・Proportional 比例 ・Integral 積分	kvp, kvi, kvd kpp, kpi, kpd reset()

			・Derivative 微分	calculate_force()
↑	↑	ImpulseController (インパルスコントローラ)	インパルス状の操作量を出力する	calculate_force()
↑	↑	StepController (ステップコントローラ)	ステップ状の操作量を出力する	calculate_force()
↑	↑	ControllerLoader (コントローラローダ)	設定 (jsonファイル) を読み込み、 Controllerクラスのインスタンスを生成	load()

default_controller.json # コントローラ デフォルト設定

```
[  # 拡張性を考慮し、リスト [] と 辞書 {} を組み合わせる
    {
        "controller": [
            {
                "version": "0.0.1",      # 設定バージョン (major.minor.patch)
                "type": "PID",           # コントローラの種類
                "kvp_N_(m_s)": 10000.0,  # 速度比例ゲイン [N/(m/s)]
                "kvi_N_(m_s)": 1000.0,   # 速度積分ゲイン [N/(m/s)]
                "kvd_N_(m_s)": 100.0,    # 速度微分ゲイン [N/(m/s)]
                "kpp_N_m": 1000.0,       # 位置比例ゲイン [N/m]
                "kpi_N_m": 500.0,        # 位置積分ゲイン [N/m]
                "kpd_N_m": 10.0          # 位置微分ゲイン [N/m]
            },
            {
                "version": "0.2.0",
                "type": "default"  # デフォルトコントローラ（テスト用）
            },
            {
                "version": "0.2.0",
                "type": "impulse",
                "impulse_force_N": 1000,         # インパルス推力 [N]
                "impulse_on_timestep_count": 1,  # インパルスON タイムステップ数 [-]
                "delay_s": 1.0                   # 遅延時間 [s]
            },
            {
                "version": "0.2.0",
                "type": "step",
                "step_force_N": 2.0,  # ステップ推力 [N]
                "delay_s": 1.0        # 遅延時間 [s]
            }
        ]
    }
]
```

4.2.5 パッケージ: tkmotion.plant

plant (control target) / プラント (制御対象)

モジュール	ver	クラス	責務	主要特性
plant.py	0.2.0	Plant (プラント)	制御対象を表現する	physical_obj
↑	↑	PlantLoader (プラントローダ)	プラントの設定 (jsonファイル) を読み込み、 Plantクラスと、Plantに属する各クラスのインスタンスを生成	load()
physical_object.py	0.2.0	PhysicalObject (物理オブジェクト)	現実の物体を表現する	mass acc, vel, pos reset() apply_force()
<<将来>> motor.py	将来	Motor (モータ)	モータを表現する (最大トルク・出力遅れ・逆起電力をシミュレーションに反映)	-
<<将来>> encoder.py	将来	Encoder (エンコーダ)	エンコーダを表現する。 (分解能や信号ジッタをシミュレーションに反映)	-

default_plant.json # プラント デフォルト設定

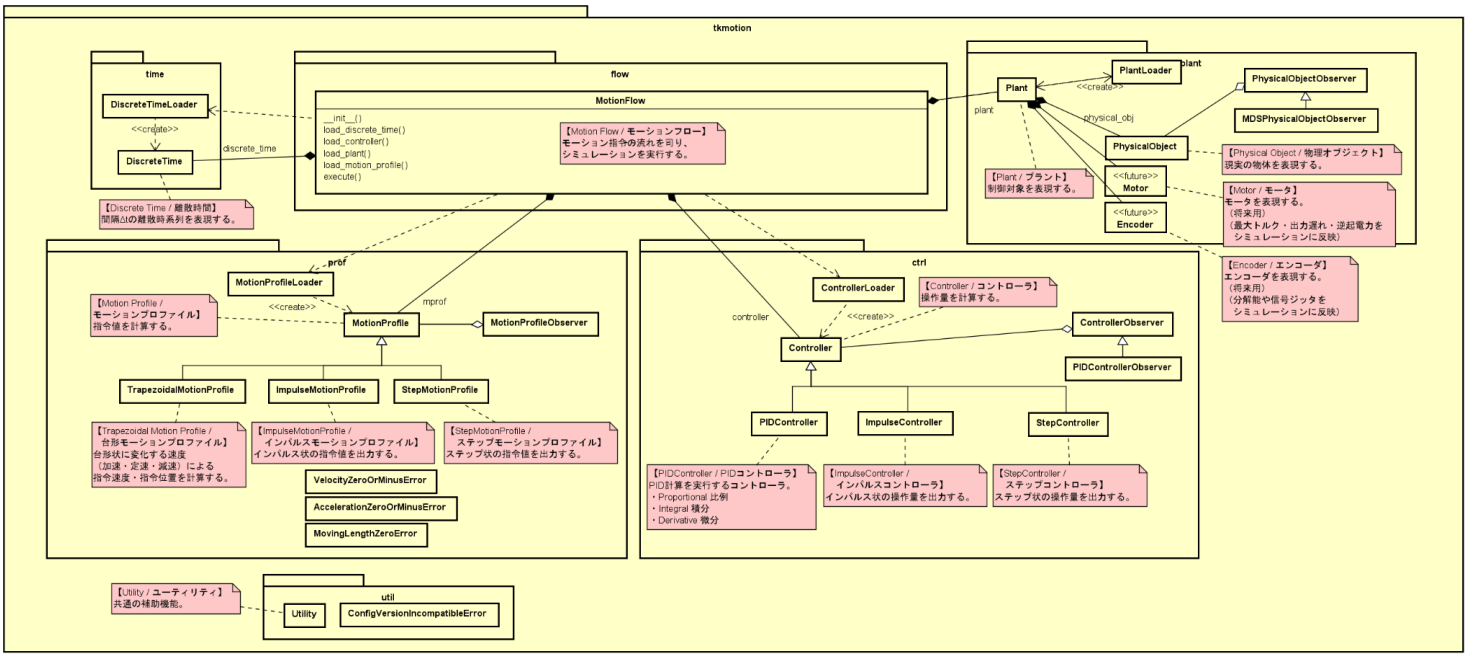
```
[  # 拡張性を考慮し、リスト [] と 辞書 {} を組み合わせる
    {
        "plant": [  # プラント
            {
                "version" : "0.1.0",   # 設定バージョン (major.minor.patch)
                "physical_object" : [  # 物理オブジェクト
                    {
                        "version" : "0.1.0",   # 設定バージョン (major.minor.patch)
                        "mass_kg" : 2.0        # 質量 [kg]
                    }
                ]
            }
        ]
    }
]
```

4.2.6 パッケージ: tkmotion.util

utility / ユーティリティ

モジュール	ver	クラス	責務	主要特性
utility.py	0.1.0	Utility (ユーティリティ)	共通の補助機能を定義する	is_config_compatible()
↑	↑	ConfigVersionIncompatibleError (設定バージョン非互換エラー)	設定バージョンに互換性がない例外を表す	-

4.3 クラス図（概要）



4.4 クラス図 詳細情報（Plant UML）

```
@startuml
class Motor <<future>>
class Encoder <<future>>
class PhysicalObject {
  +grav_acc_m_s2 : float
  +module_version : string
  +config_version : string
  +mass : float
  +acc : float
  +prev_acc : float
  +vel : float
  +prev_vel : float
  +pos : float
  +prev_pos : float
  +__init__() : void
  +get_config() : dict
  +get_observer() : PhysicalObjectObserver
  +reset() : void
  +set_state() : void
  +apply_force() : void
}
class DiscreteTime {
  +module_version : string
  +config_version : string
  +dt : float
  +duration : float
  +__init__() : void
```

```
  +get_config() : dict
  +get_time_step_generator() : generator
}
class MotionFlow {
  +module_version : string
  +__init__() : void
  +load_discrete_time() : void
  +load_controller() : void
  +load_plant() : void
  +load_motion_profile() : void
  +execute() : void
}
class MotionProfileLoader {
  +module_version : string
  +__init__() : void
  +load() : void
}
class DiscreteTimeLoader {
  +module_version : string
  +__init__() : void
  +load() : void
}
DiscreteTimeLoader ..> DiscreteTime
class MotionProfile {
  +module_version : string
  +config_version : string
  +type : string
  +cmd_vel : float
  +cmd_pos : float
  +__init__() : void
  +get_config() : dict
  +get_observer() : MotionProfileObserver
  +calculate_cmd_vel_pos() : tuple
}
MotionProfileLoader ..> MotionProfile
MotionFlow *-- "+ mprof" MotionProfile
Plant *-- "+ physical_obj" PhysicalObject
MotionFlow *-- "+ discrete_time" DiscreteTime
class TrapezoidalMotionProfile {
  +V : float
  +A : float
  +L : float
  +dir : float
  +Ta : float
  +T : float
  +Tc : float
  +__init__() : void
  +calculate_cmd_vel_pos() : tuple
}
```

```
MotionProfile <|-- TrapezoidalMotionProfile
class Plant {
  +module_version : string
  +config_version : string
  +__init__() : void
  +get_config() : dict
}
MotionFlow *-- "+ plant" Plant
class PlantLoader {
  +module_version : string
  +__init__() : void
  +load() : void
}
PlantLoader ..> Plant
MotionFlow ..> PlantLoader
class Controller {
  +module_version : string
  +config_version : string
  +type : string
  +vel_error : float
  +pos_error : float
  +vel_error_cumsum : float
  +pos_error_cumsum : float
  +vel_error_diff : float
  +pos_error_diff : float
  +force : float
  +__init__() : void
  +get_config() : dict
  +get_observer() : ControllerObserver
  +reset() : void
  +calculate_force() : float
}
MotionFlow *-- "+ controller" Controller
class ControllerLoader {
  +module_version : string
  +__init__() : void
  +load() : void
}
ControllerLoader ..> Controller
class PIDController {
  +kvp : float
  +kvi : float
  +kvd : float
  +kpp : float
  +kpi : float
  +kpd : float
  +vel_error : float
  +pos_error : float
  +vel_error_cumsum : float
```

```
  +pos_error_cumsum : float
  +vel_error_diff : float
  +pos_error_diff : float
  +__init__() : void
  +get_observer() : PIDControllerObserver
  +reset() : void
  +calculate_force() : float
}
Controller <|-- PIDController
Plant *-- Motor
Plant *-- Encoder
class Utility {
  +get_module_version() : void
  +is_config_compatible() : boolean
}
class ConfigVersionIncompatibleError
class VelocityZeroOrMinusError
class AccelerationZeroOrMinusError
class MovingLengthZeroError
class ImpulseController {
  +p_force : float
  +on_timestep_count : int
  +delay_s : float
  +__init__() : void
  +reset() : void
  +calculate_force() : float
}
Controller <|-- ImpulseController
class ImpulseMotionProfile {
  +p_vel : float
  +p_pos : float
  +on_timestep_count : int
  +delay_s : float
  +__init__() : void
  +calculate_cmd_vel_pos() : tuple
}
MotionProfile <|-- ImpulseMotionProfile
class StepMotionProfile {
  +s_vel : float
  +s_pos : float
  +delay_s : float
  +__init__() : void
  +calculate_cmd_vel_pos() : tuple
}
MotionProfile <|-- StepMotionProfile
class StepController {
  +s_force : float
  +delay_s : float
  +__init__() : void
```

```
  +reset() : void
  +calculate_force() : float
}
Controller <|-- StepController
class MDSPhysicalObject {
  +damper : float
  +spring : float
  +spring_balance_pos : float
  +static_friction_coeff : float
  +dynamic_friction_coeff : float
  +__init__() : void
  +get_observer() : MDSPhysicalObjectObserver
  +calc_char_values() : tuple
}
PhysicalObject <|-- MDSPhysicalObject
class PhysicalObjectObserver {
  +physical_obj : PhysicalObject
  +__init__() : void
  +reset() : void
  +observe() : void
  +get_observed_data() : dict
}
class MDSPhysicalObjectObserver {
  +physical_obj : MDSPhysicalObject
  +__init__() : void
}
PhysicalObjectObserver <|-- MDSPhysicalObjectObserver
class ControllerObserver {
  +module_version : string
  +controller : Controller
  +__init__() : void
  +reset() : void
  +observe() : void
  +get_observed_data() : dict
}
class PIDControllerObserver {
  +controller : PIDController
  +__init__() : void
}
ControllerObserver <|-- PIDControllerObserver
ControllerObserver o-- Controller
PhysicalObjectObserver o-- PhysicalObject
class MotionProfileObserver {
  +module_version : string
  +profile : MotionProfile
  +__init__() : void
  +reset() : void
  +observe() : void
  +get_observed_data() : dict
```

```
}
MotionProfileObserver o-- MotionProfile

note right of MotionFlow : 【Motion Flow / モーションフロー】\nモーション指令の流れを司り、\nシミュレーションを実行する。
note right of DiscreteTime : 【Discrete Time / 離散時間】\n間隔Δtの離散時系列を表現する。
note right of MotionProfile : 【Motion Profile / モーションプロファイル】\n指令値を計算する。\n（デフォルト指令値としてゼロを出力）
note right of Controller : 【Controller / コントローラ】\n操作量を計算する。\n（デフォルト操作量としてゼロを出力）
note right of Plant : 【Plant / プラント】\n制御対象を表現する。
note right of PhysicalObject : 【Physical Object / 物理オブジェクト】\n現実の物体を表現する。
note right of Motor : 【Motor / モータ】\nモータを表現する。\n（将来用）\n（最大トルク・出力遅れ・逆起電力を\n シミュレーションに反映）
note right of Encoder : 【Encoder / エンコーダ】\nエンコーダを表現する。\n（将来用）\n（分解能や信号ジッタを\n シミュレーションに反映）
note right of PIDController : 【PIDController / PIDコントローラ】\nPID計算を実行するコントローラ。\n・Proportional 比例\n・Integral 積分\n・Derivative 微分
note right of TrapezoidalMotionProfile : 【Trapezoidal Motion Profile /\n 台形モーションプロファイル】\n台形状に変化する速度\n（加速・定速・減速）による\n指令速度・指令位置を計算する。
note right of Utility : 【Utility / ユーティリティ】\n共通の補助機能。
note right of ImpulseController : 【ImpulseController / \n インパルスコントローラ】\nインパルス状の操作量を出力する。
note right of ImpulseMotionProfile : 【ImpulseMotionProfile / \n インパルスモーションプロファイル】\nインパルス状の指令値を出力する。
note right of StepMotionProfile : 【StepMotionProfile / \n ステップモーションプロファイル】\nステップ状の指令値を出力する。
note right of StepController : 【StepController / \n ステップコントローラ】\nステップ状の操作量を出力する。
note right of MDSPhysicalObject : 【Mass-Spring-Damper Physical Object /\n 質量・ダンパ・ばね物理オブジェクト】\n現実の物体を表現する。
@enduml
```

---

5. バージョン管理

5.1 ポリシー

Python Packaging User Guide バージョニング (https://packaging.python.org/ja/latest/discussions/versioning/) の「セマンティックバージョニング」を採用する。

バージョン番号変更の規則（上記 Python Packaging User Guide より）

- APIの変更で互換性を失う時には major 番号、
- 後方互換性を保ったままで新機能を追加する場合には minor を、そして
- 後方互換性を維持したままのバグ修正の場合には patch を増加させる。

5.2 代表バージョン

tkmotionライブラリ代表バージョン = motion_flowモジュールのバージョン（このライブラリの中心モジュール）

6. 開発環境

6.1 Pythonバージョン

> python --version

Python 3.13.1

6.2 Pythonパッケージ

> pip freeze

affine==2.4.0
anyio==4.9.0
argon2-cffi==23.1.0
argon2-cffi-bindings==21.2.0
arrow==1.3.0
asttokens==3.0.0
async-lru==2.0.5
attrs==25.3.0
babel==2.17.0
beautifulsoup4==4.13.4
black==25.1.0
bleach==6.2.0
branca==0.8.1
certifi==2025.4.26
cffi==1.17.1
charset-normalizer==3.4.1
click==8.1.8
click-plugins==1.1.1.2
cligj==0.7.2
colorama==0.4.6
comm==0.2.2
contourpy==1.3.2
cycler==0.12.1
debugpy==1.8.14
decorator==5.2.1
defusedxml==0.7.1
et_xmlfile==2.0.0
executing==2.2.0
fastjsonschema==2.21.1
feedparser==6.0.12
flake8==7.2.0
folium==0.20.0
fonttools==4.57.0
fqdn==1.5.1

geographiclib==2.0
geopandas==1.1.1
geopy==2.4.1
h11==0.16.0
html5lib==1.1
httpcore==1.0.9
httpx==0.28.1
idna==3.10
ipykernel==6.29.5
ipython==9.2.0
ipython_pygments_lexers==1.1.1
isoduration==20.11.0
jedi==0.19.2
Jinja2==3.1.6
joblib==1.4.2
json5==0.12.0
jsonpickle==4.1.1
jsonpointer==3.0.0
jsonschema==4.23.0
jsonschema-specifications==2025.4.1
jupyter-events==0.12.0
jupyter-lsp==2.2.5
jupyter_client==8.6.3
jupyter_core==5.7.2
jupyter_server==2.15.0
jupyter_server_terminals==0.5.3
jupyterlab==4.4.1
jupyterlab_pygments==0.3.0
jupyterlab_server==2.27.3
kiwisolver==1.4.8
lxml==6.0.0
MarkupSafe==3.0.2
matplotlib==3.10.1
matplotlib-inline==0.1.7
mccabe==0.7.0
mistune==3.1.3
mypy==1.15.0
mypy_extensions==1.1.0
narwhals==2.0.0
nbclient==0.10.2
nbconvert==7.16.6
nbformat==5.10.4
nest-asyncio==1.6.0
networkx==3.5
notebook_shim==0.2.4
numpy==2.2.5
opencv-python==4.12.0.88
openpyxl==3.1.5
overrides==7.7.0
packaging==25.0

pandas==2.2.3
pandocfilters==1.5.1
parso==0.8.4
pathspec==0.12.1
pillow==11.2.1
platformdirs==4.3.7
plotly==6.2.0
prometheus_client==0.21.1
prompt_toolkit==3.0.51
psutil==7.0.0
pure_eval==0.2.3
pyarrow==21.0.0
pycodestyle==2.13.0
pycparser==2.22
pyflakes==3.3.2
Pygments==2.19.1
pyogrio==0.11.0
pyparsing==3.2.3
pyproj==3.7.1
python-dateutil==2.9.0.post0
python-json-logger==3.3.0
pytz==2025.2
pyvis==0.3.2
pywin32==310
pywinpty==2.0.15
PyYAML==6.0.2
pyzmq==26.4.0
rasterio==1.4.3
referencing==0.36.2
requests==2.32.3
retrying==1.3.4
rfc3339-validator==0.1.4
rfc3986-validator==0.1.1
rpds-py==0.24.0
scikit-learn==1.6.1
scipy==1.15.2
Send2Trash==1.8.3
setuptools==80.1.0
sgmllib3k==1.0.0
shapely==2.1.1
six==1.17.0
sniffio==1.3.1
soupsieve==2.7
stack-data==0.6.3
terminado==0.18.1
threadpoolctl==3.6.0
tinycss2==1.4.0
tornado==6.4.2
traitlets==5.14.3
types-python-dateutil==2.9.0.20241206

typing_extensions==4.13.2
tzdata==2025.2
uri-template==1.3.0
urllib3==2.4.0
wcwidth==0.2.13
webcolors==24.11.1
webencodings==0.5.1
websocket-client==1.8.0
xyzservices==2025.4.0

7. ライセンス

履歴

日付	ライブラリVer	内容	作成者
作成中	0.3.0	質量・ダンパ・ばねモデル	Takayoshi Matsuyama
2025/12/7	0.2.0	インパルス関数、ステップ関数	Takayoshi Matsuyama
2025/12/1	0.1.0	台形速度PID制御シミュレーション	Takayoshi Matsuyama