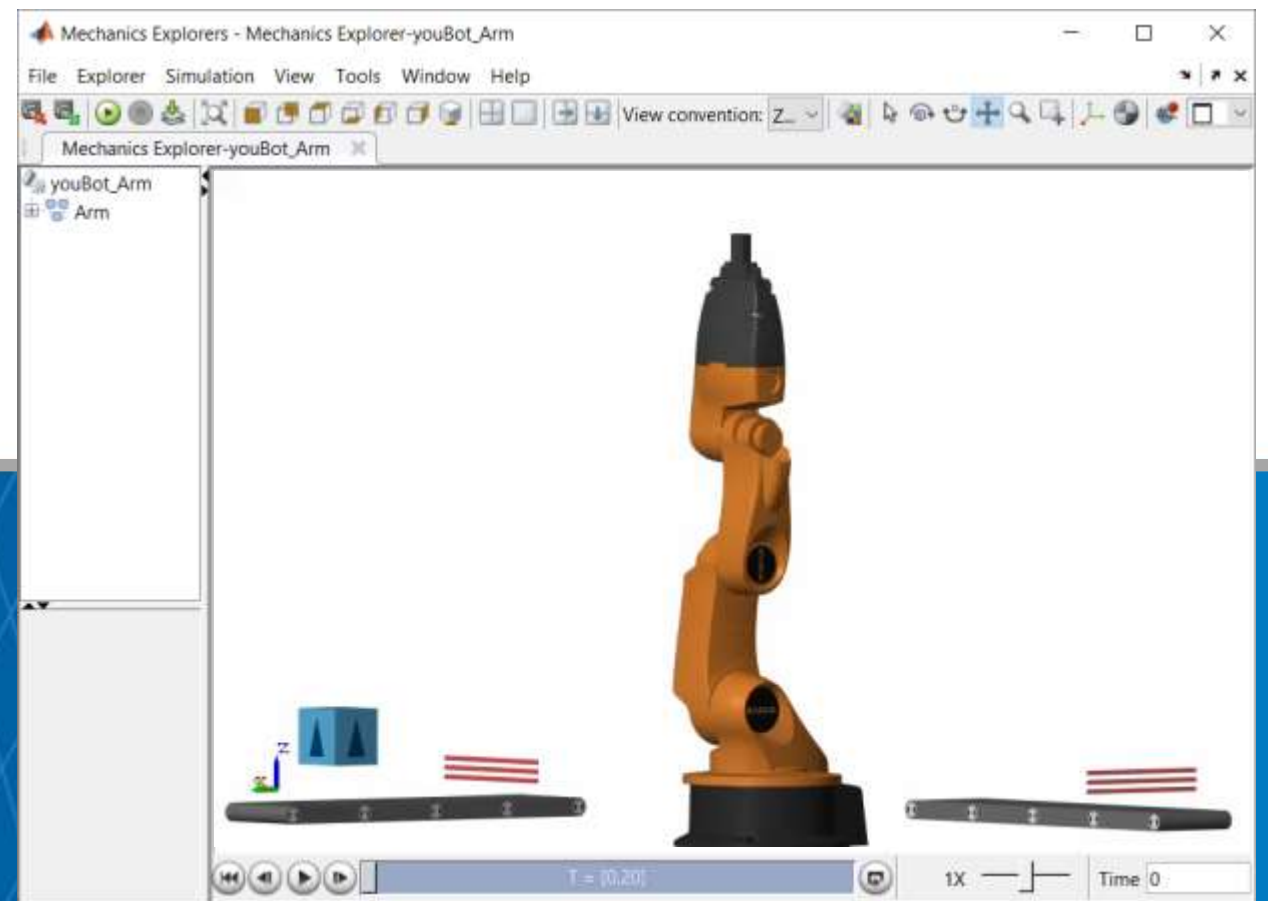
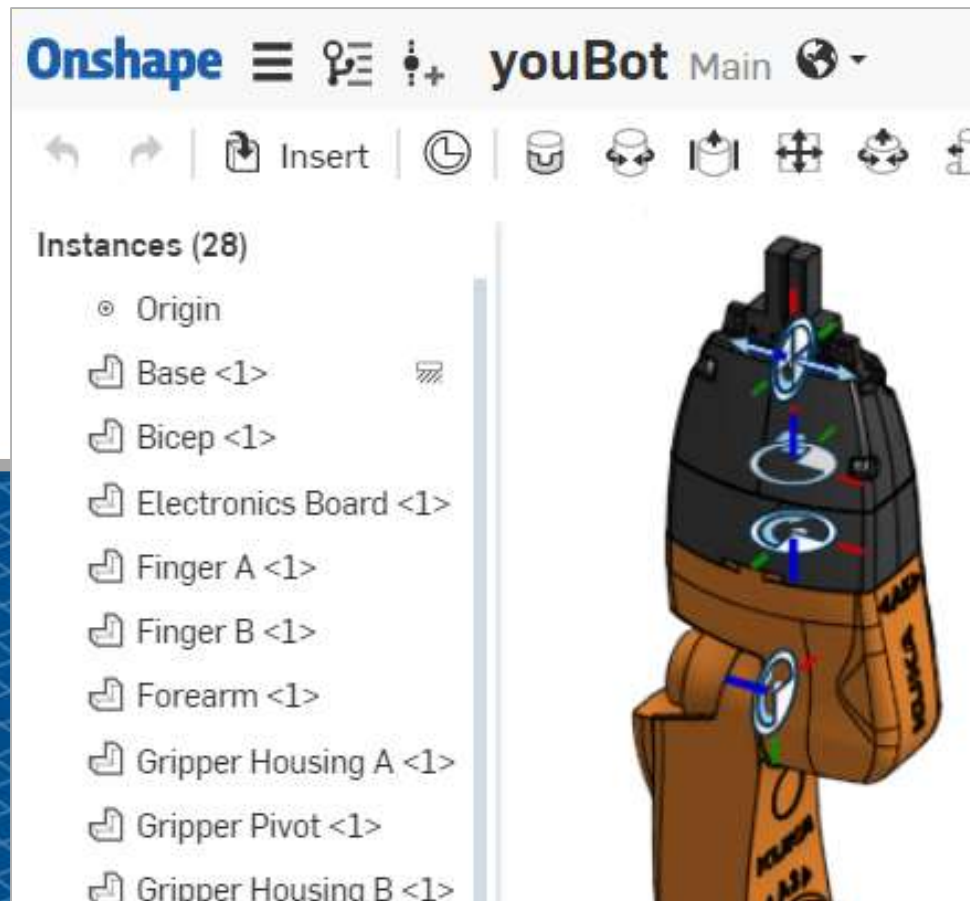
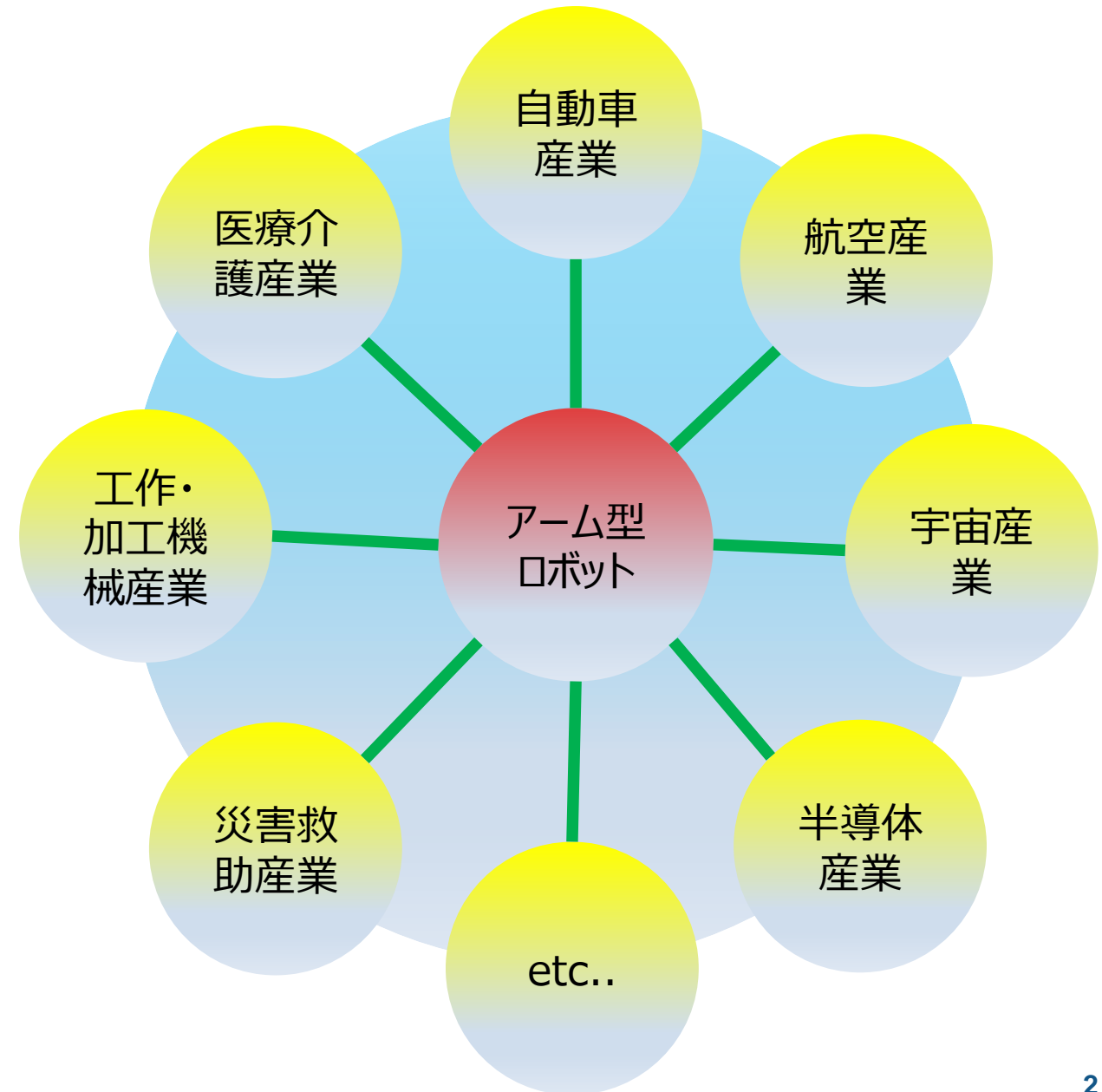


アーム型ロボット開発・導入を支援するMATLAB/Simulink

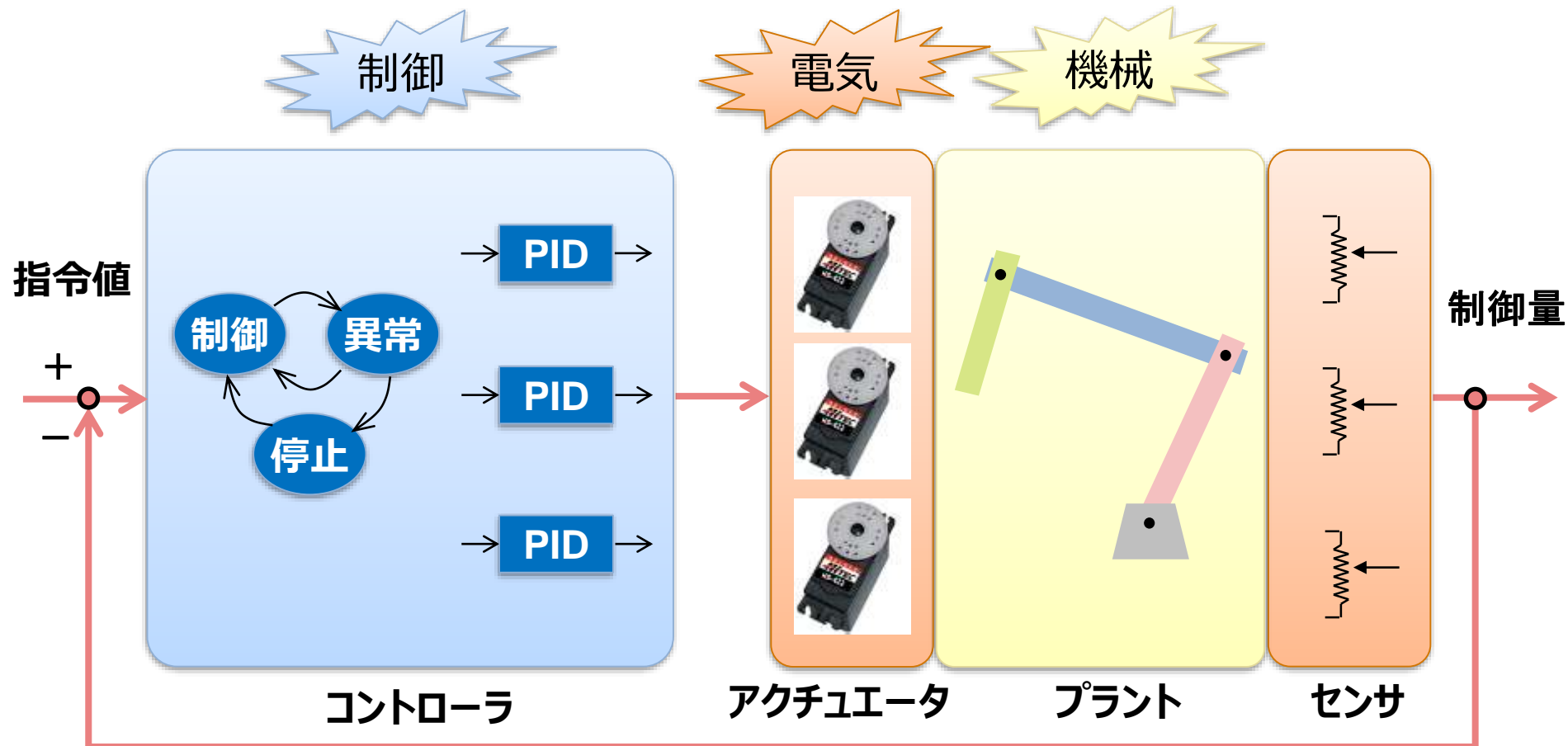


アーム型ロボットの活用産業

産業	用途例
自動車・自動車部品	ジョイント組み立て、ウインドウ搭載支援 等
航空	機体の組み立て支援 等
宇宙	宇宙空間での衛星の組み立て、修理 等
半導体 電子部品	卓上はんだ付け、微小チップ部品の整列 等
災害救助	汚染環境での作業支援 等
医療・介護	手術による執刀支援 等
工作・加工機	工作・加工機械の組み立て等



ロボットの全体構成



制御・電気・機械が一体のシステムで、所望の機能・性能を実現し、さらに安全性が要求される

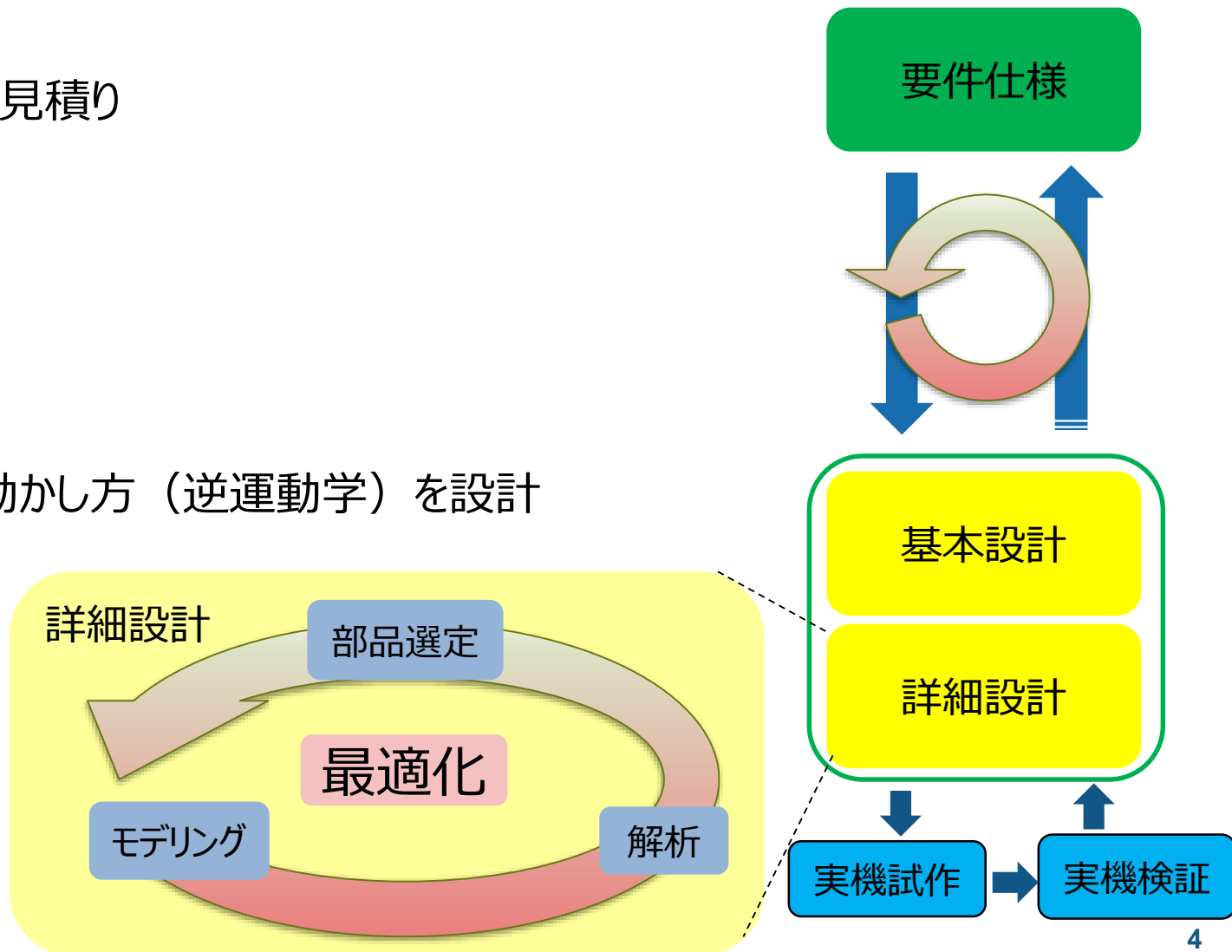
導入・開発における検討事項

■ 導入検討者

- 使用用途の検討
- コスト(消費電力、動作時間、etc) の見積り
- 軌道設計検討 / 動作制約の検討
- 外乱・フェールセーフ動作の検討

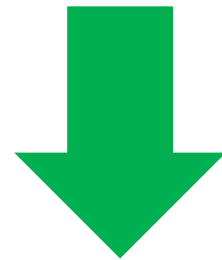
■ 開発者（特に制御設計者）

- エンドエフェクタの軌道に対する関節の動かし方（逆運動学）を設計
- 動作制約を踏まえた最適動作の検討
- フェールセーフ動作の設計



従来の開発における主な課題

- 実機ベースで開発しているため、現物が出来るまで机上検討しかできない。
- システム全体の最適化が出来ていない。
- CAD資産が制御領域で有効活用できていない。

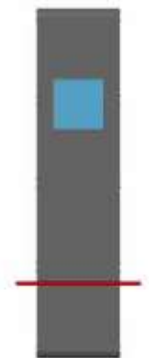


課題

高度・複雑なシステム設計を、設備導入前に効率的に行う必要があるが、現状、システム設計ができておらず、実機ベースで開発している。

設計におけるチャレンジ

システム:

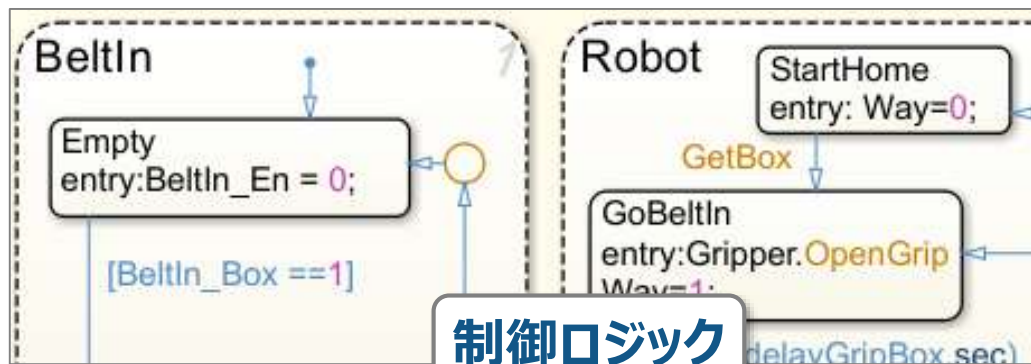


Challenge: モーターを選択し、ロボットとコンベアベルトの制御を定義します。

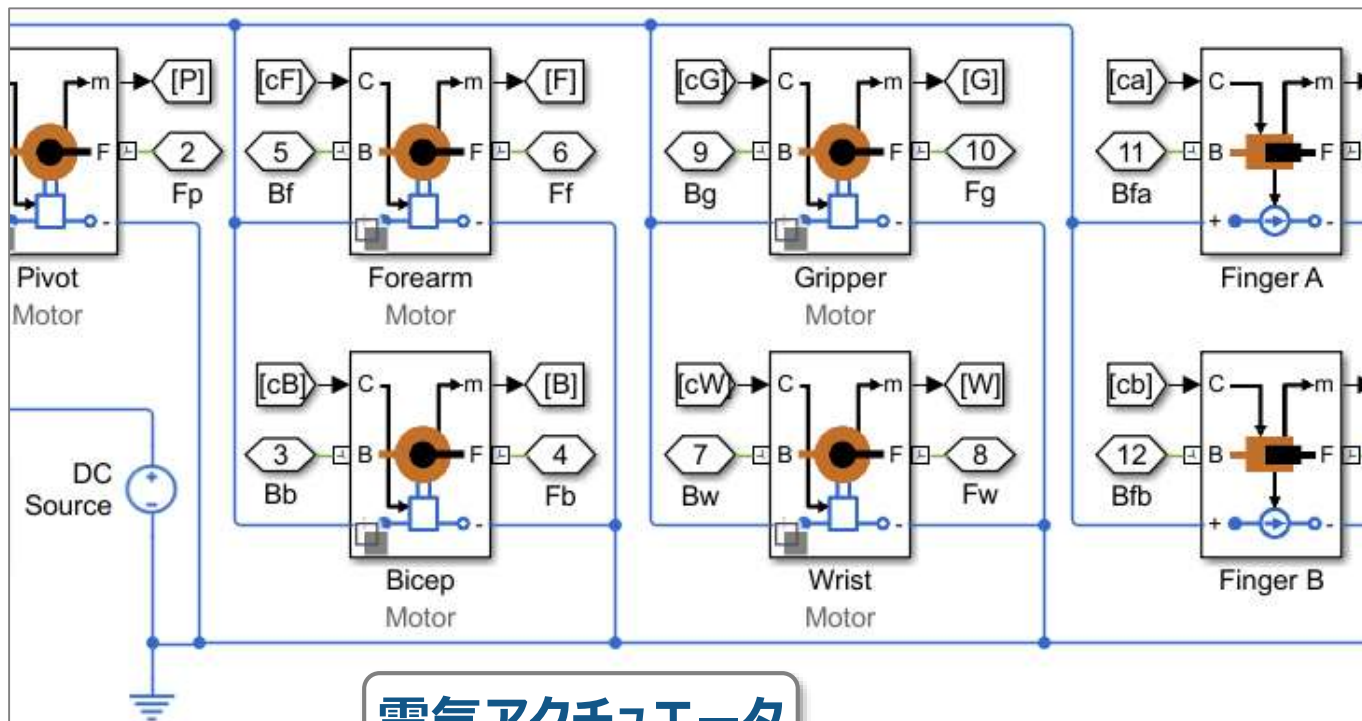
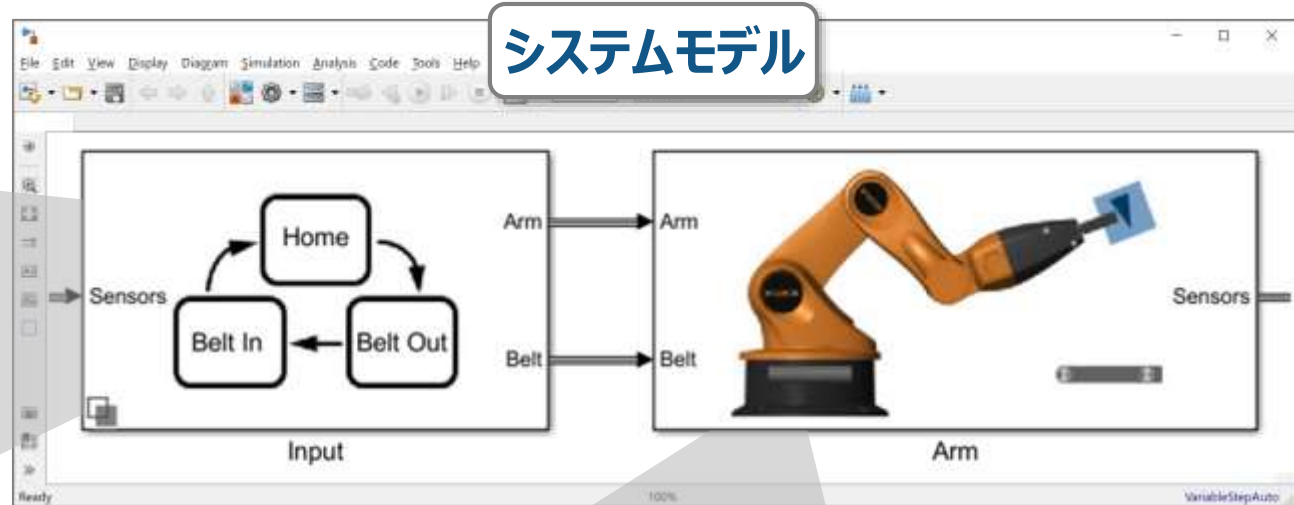
Solution: OnshapeモデルをSimscapeにインポート。シミュレーションを活用してアクチュエータ要件と制御ロジックを定義する。

1. Onshapeモデルのインポート
2. モーター要件の決定
3. 電動アクチュエータの統合
4. 消費電力を最小限に抑える
5. 制御ロジックの開発

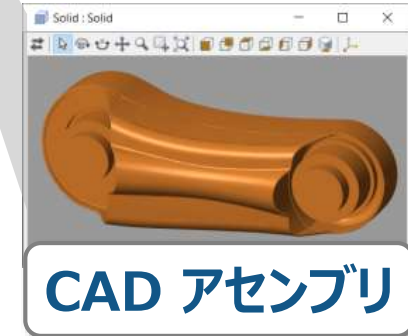
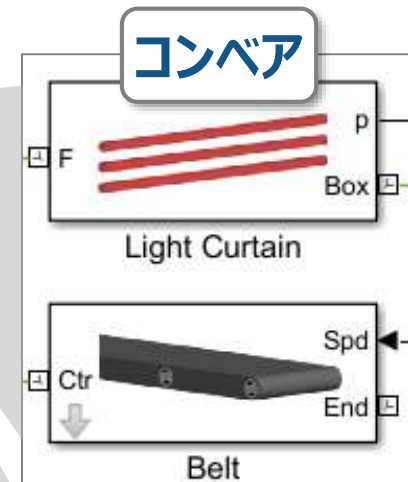
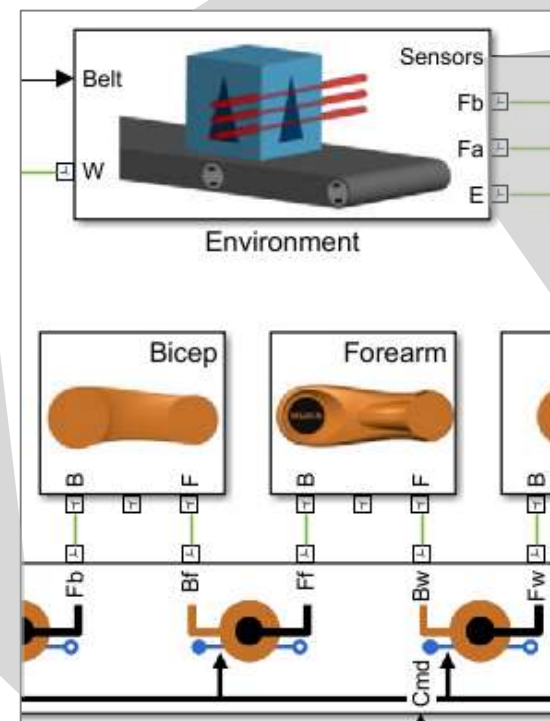
システムモデル



制御ロジック



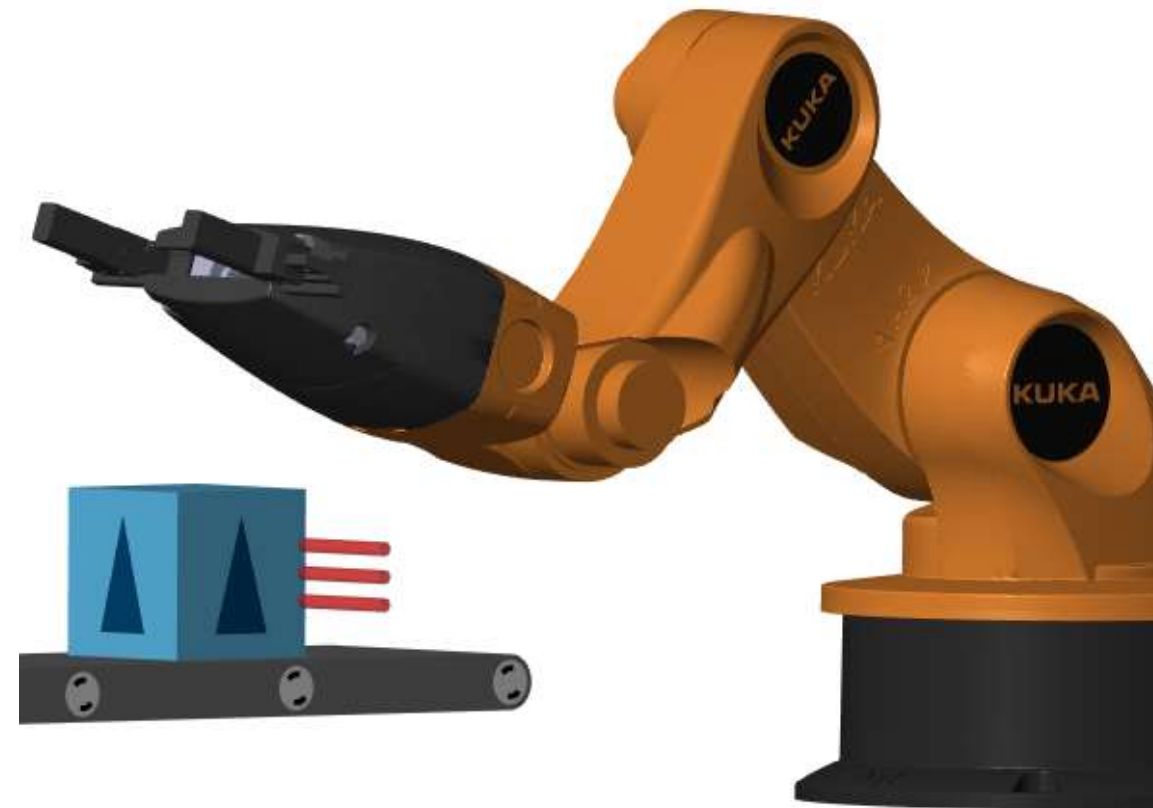
電気アクチュエータ



CAD アセンブリ

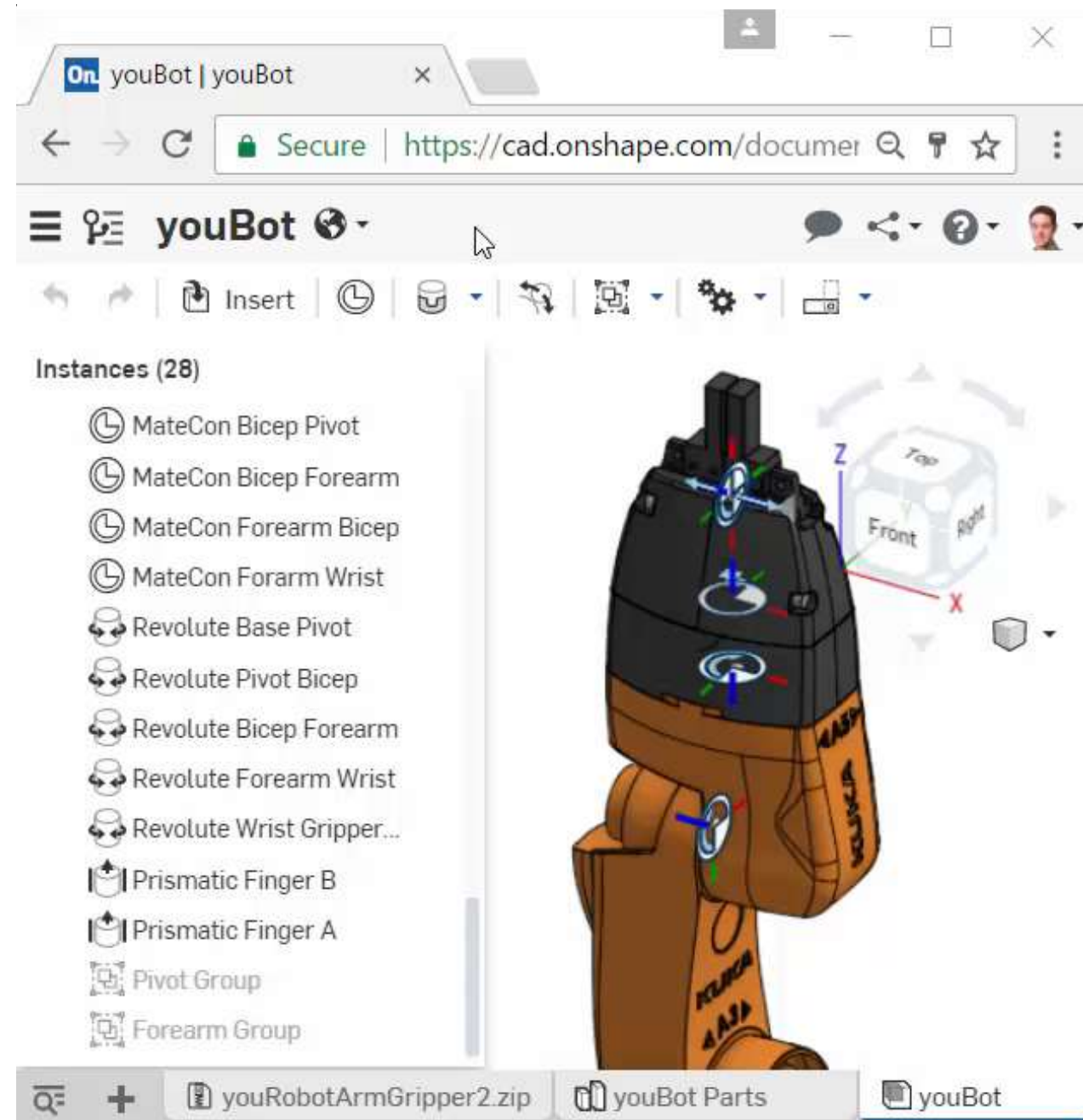
何故、CADとマルチドメインの動的シミュレーションを組み合わせるか？

- **機械設計の回数削減**
 - ∵ **要件が明確**になっているため
- **試作機が制作が減少**
 - ∵ **早期段階で設計ミス**が発見できるため
- **システムコストの削減**
 - ∵ **コンポーネントが大きすぎない**ようにするため
- **システムのダウンタイム短縮**
 - ∵ **仮想試運転**を使用してデバッグ可能なため



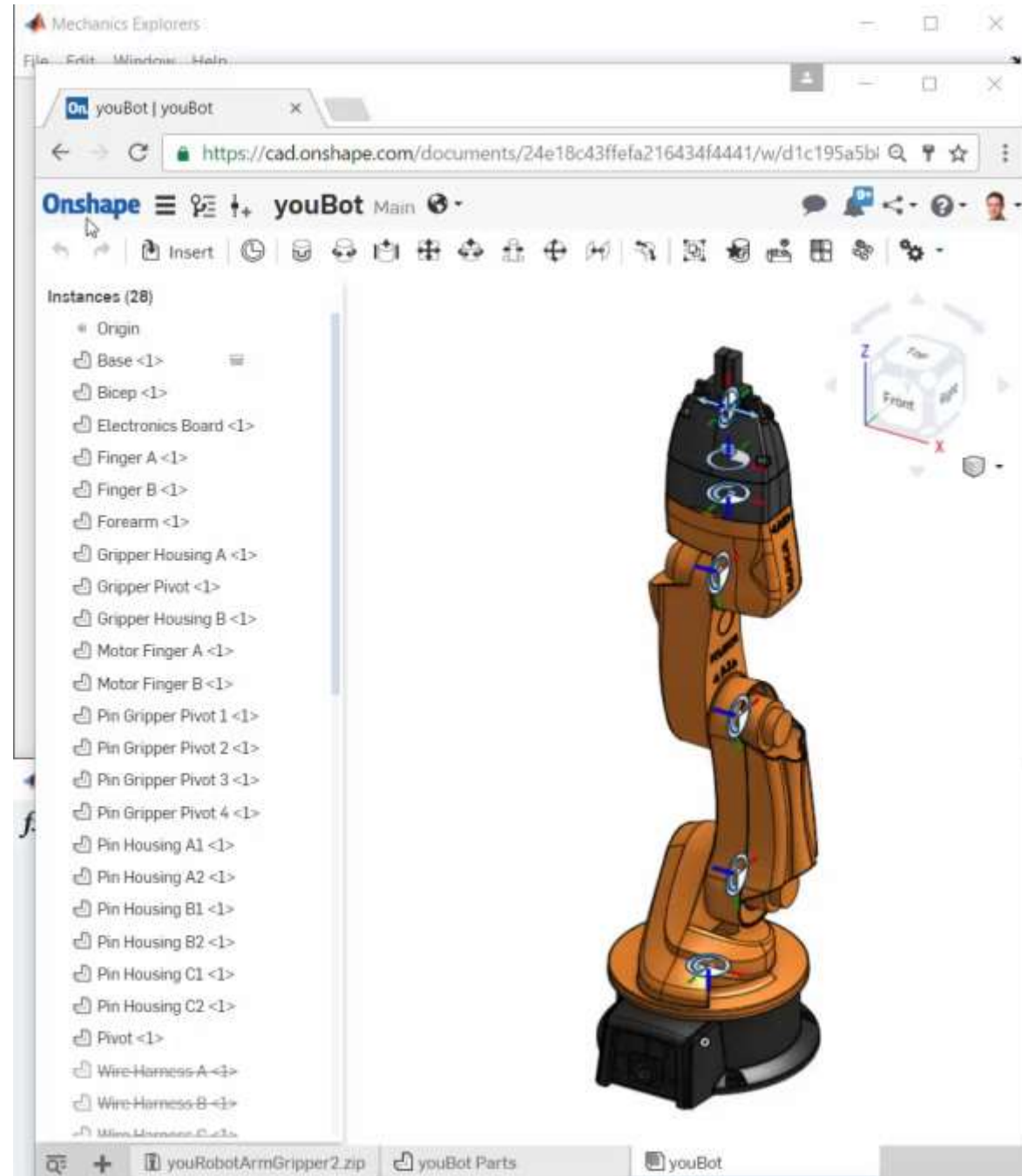
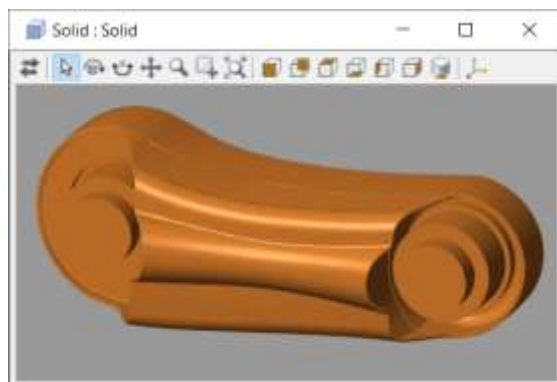
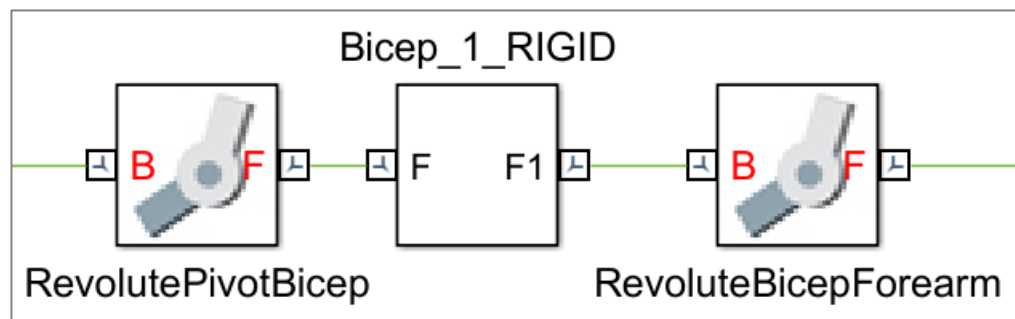
Kuka Robot

- 5自由度、and グリッパ
- Onshapeの主な利点:
ジョイントの直接定義が可能
 - マルチボディ・シミュレーションで使用される制約への正確なマッピング
- システムエンジニアは、機械設計モデルを動的シミュレーション検討に利用できる。



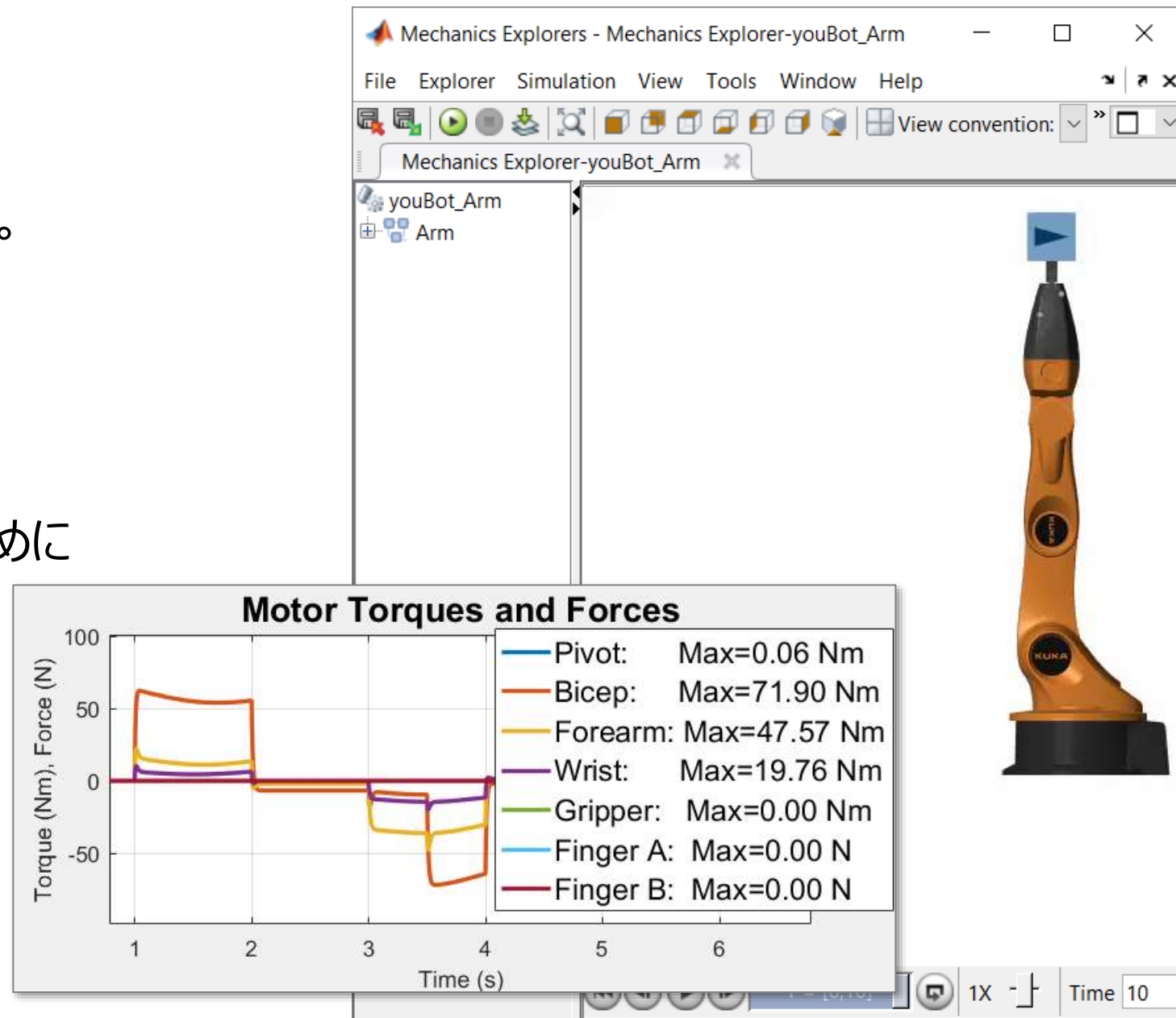
1. Onshapeモデルのインポート

- CADアセンブリから
動的シミュレーションモデルへ変換
 - 質量, 慣性, ジオメトリ, ジョイント



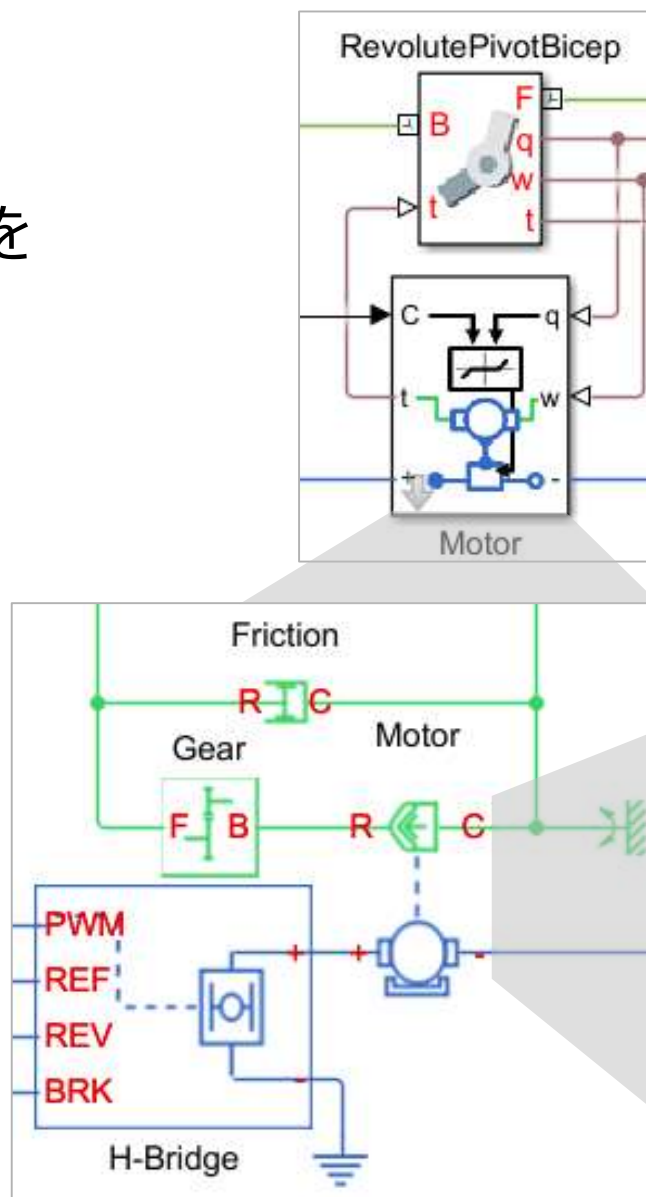
2. モーター要件の決定

- 一連のテストを定義して実行する。
 - 最大(荷重, 速度)
 - 摩擦レベルのワーストケース
 - 全可動域
- 必要なトルクと耐力を計算するために動的シミュレーションを使用
- 設計変更が起きたら、自動テストを実施し再評価する



3. 電動アクチュエータの統合

- モータ、駆動回路、ギア、摩擦を追加
- トルク要件に基づいてモータを選択する
- データシートから直接パラメータを割り当てる



Motor Data

251601

Characteristics

Terminal resistance	Ω	0.978
Terminal inductance	mH	0.573
Torque constant	mNm / A	33.5
Speed constant	rpm / V	285
Speed / torque gradient	rpm / mNm	8.32
Mechanical time constant	ms	11.8
Rotor inertia	gcm ²	135

Electrical Torque

Mechanical

Model parameterization: Circuit parameters

Armature resistance: 0.978 Ohm

Armature inductance: 0.573 mH

Torque constant: 33.5 mN*m/A

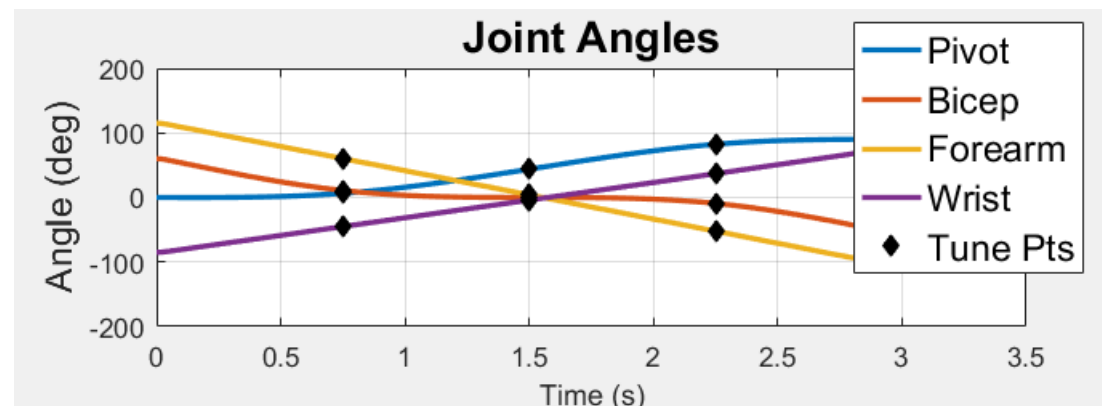
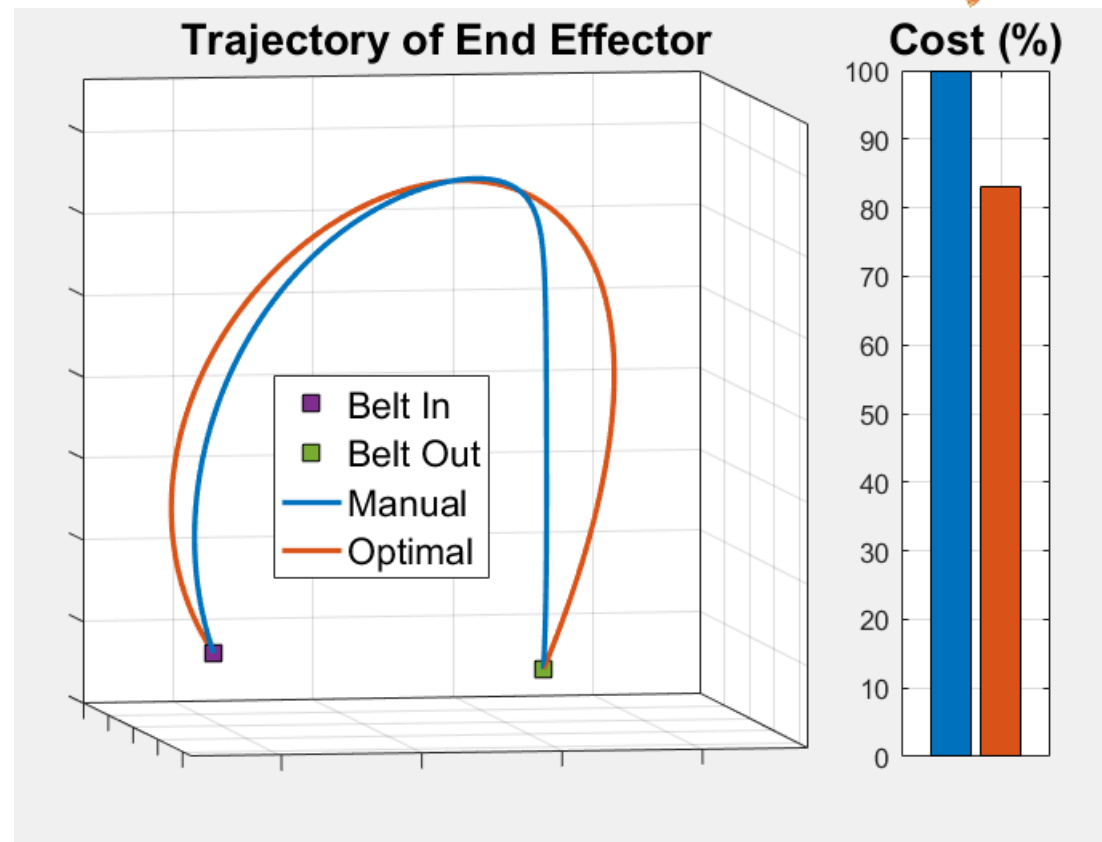
4.消費電力を最小限に抑える

モデル:

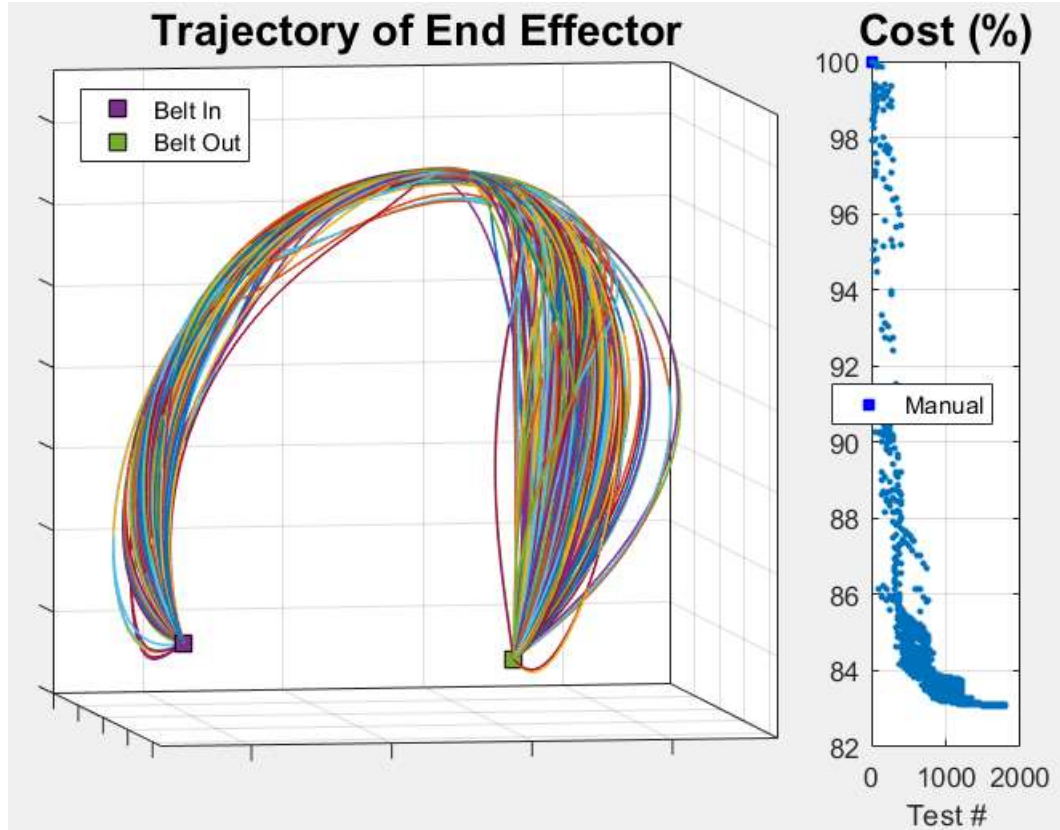


Challenge: 消費電力を最小化するアームの軌跡を特定する。

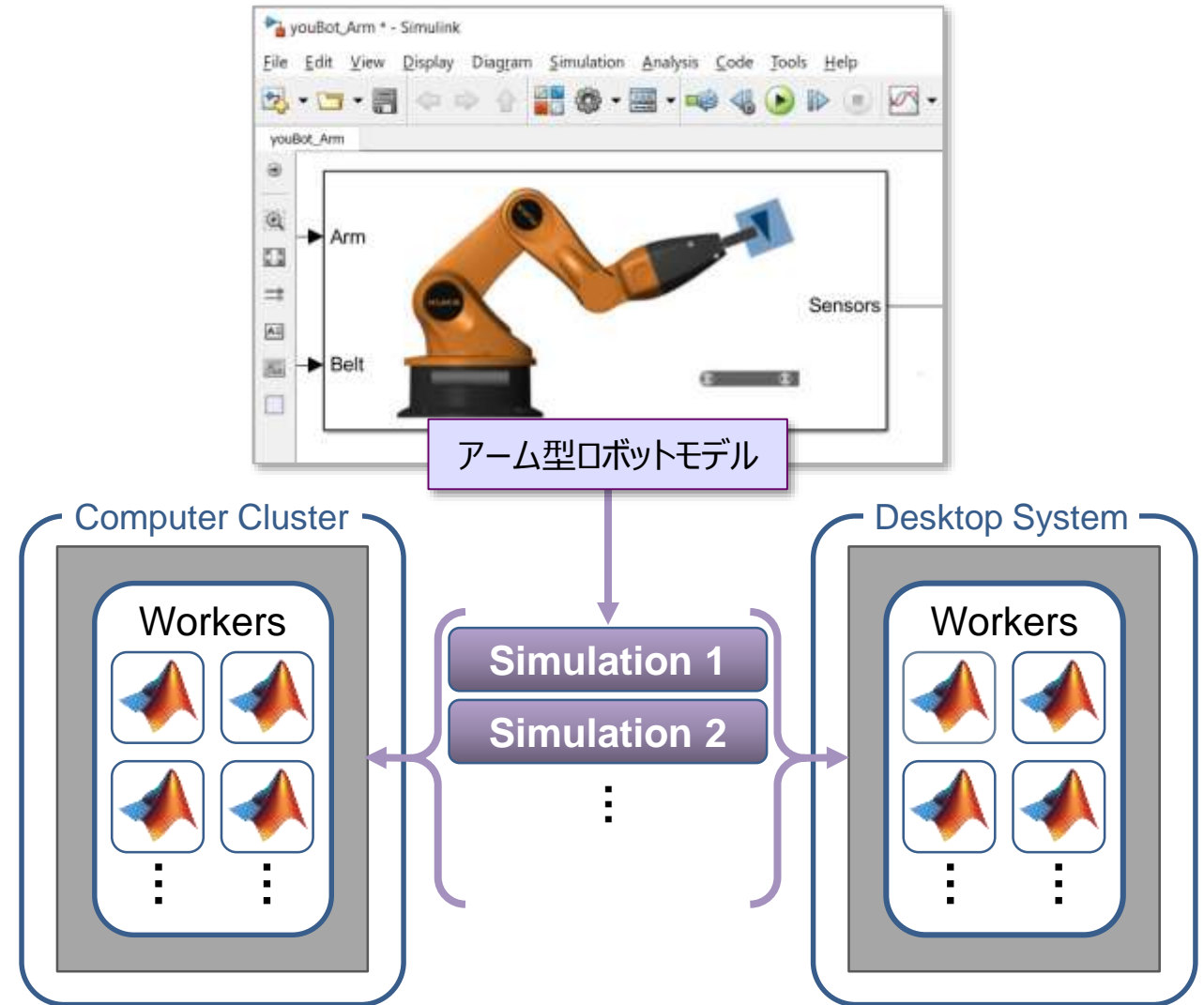
Solution: 動的シミュレーションを使用して消費電力を計算し、最適化アルゴリズムを使用して軌道をチューニングします。



Parallel Computingを使用した 繰り返し設計の加速化



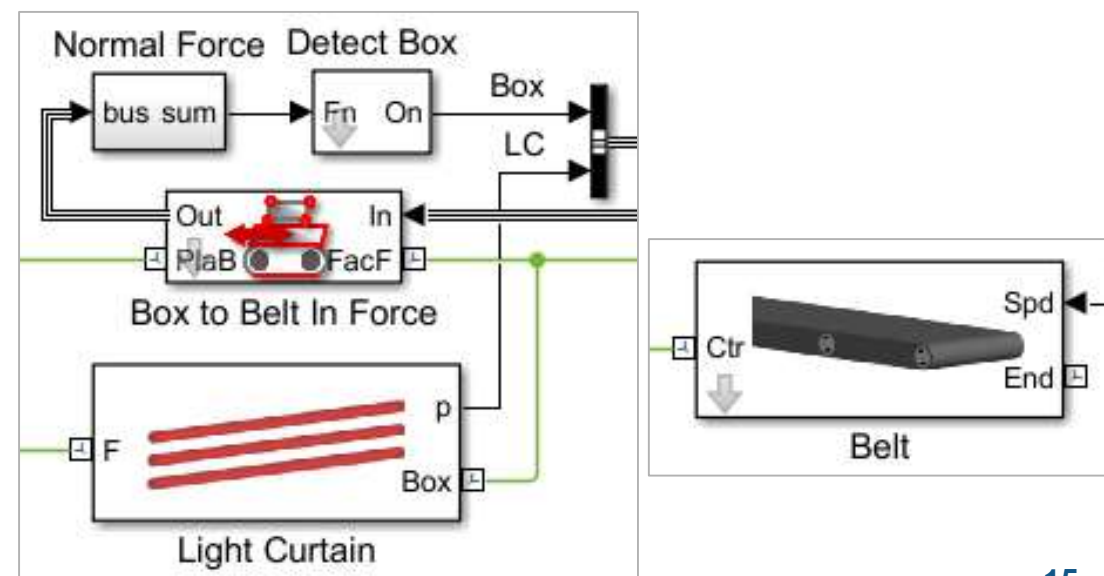
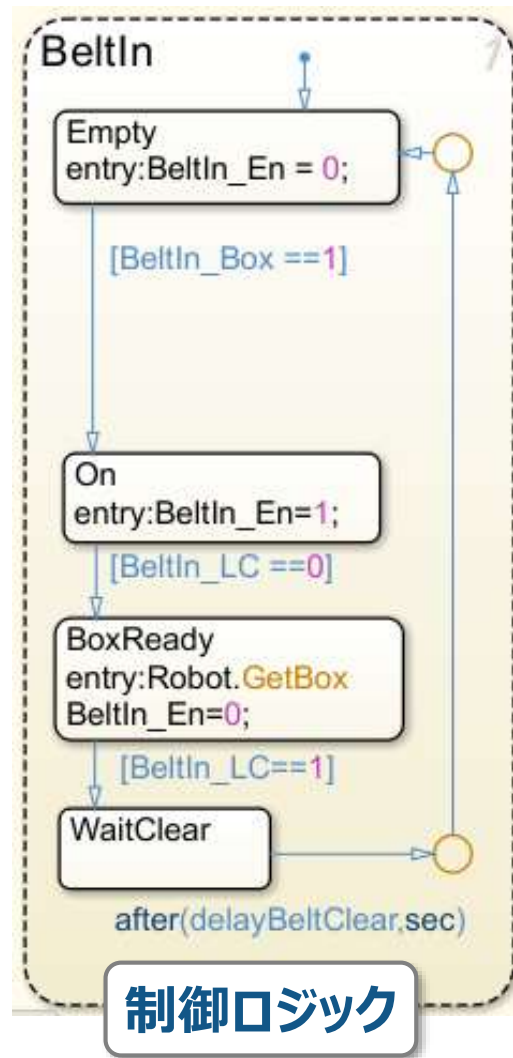
この最適化作業には、
約2000回のシミュレーションが必要でした。



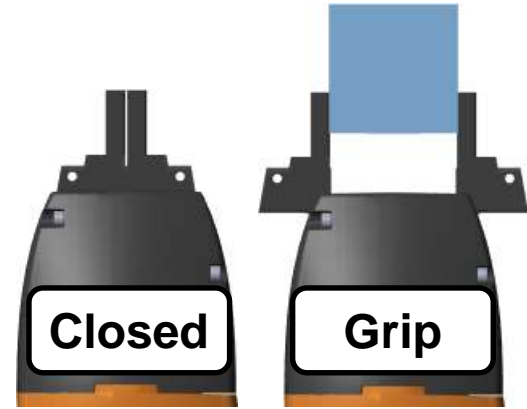
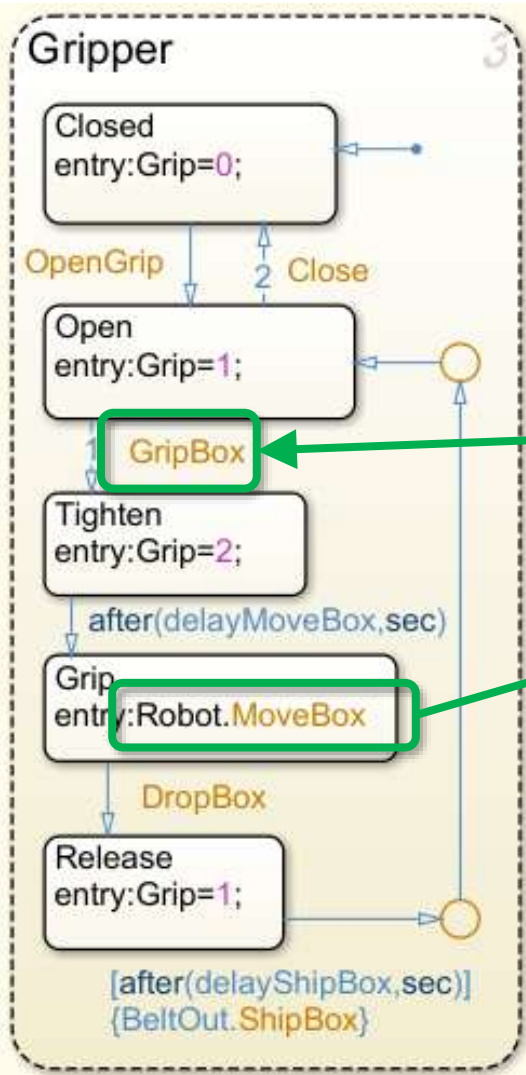
シミュレーションを並列実行すると
テストプロセスが短縮されます。

5. アームとベルトコンベアの制御ロジック開発

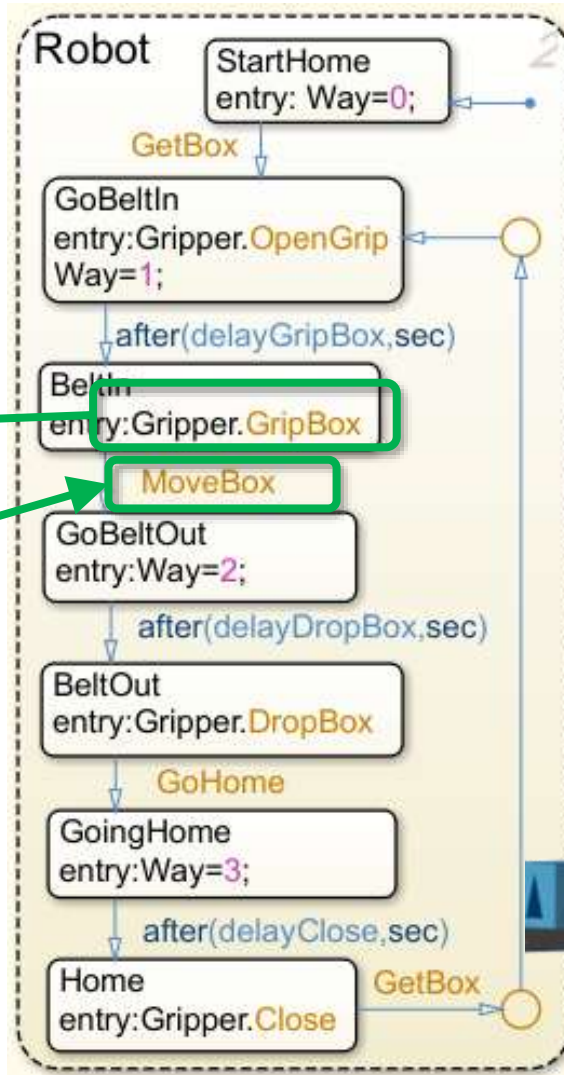
- システムモデル内のイベントを検知します。
- 状態遷移図を使用した論理設計
- システムコンポーネントのモデルを制御するために出力信号を使用します。



5. アームとベルトコンベアの制御ロジック開発



State charts depend on each other



5. アームとベルトコンベアの制御ロジック開発

Stateflow (chart) youBot_Arm/Input/Control/Logic* - Simulink

File Edit View Display Chart Simulation Analysis Code Tools Help

Logic

BeltIn

- Empty entry: BeltIn_En = 0;
- [BeltIn_Box == 1]
- On entry: BeltIn_En = 1;
- [BeltIn_LC == 0]
- BoxReady entry: Robot.GetBox; BeltIn_En = 0;
- [BeltIn_LC == 1]
- WaitClear after(delayBeltClear, sec)

Robot

- StartHome entry: Way = 0;
- GetBox
- GoBeltIn entry: Gripper.OpenGrip; Way = 1;
- after(delayGripBox, sec)
- BeltIn entry: Gripper.GripBox
- MoveBox
- GoBeltOut entry: Way = 2;
- after(delayDropBox, sec)
- BeltOut entry: Gripper.DropBox
- GoHome
- GoingHome entry: Way = 3;
- after(delayClose, sec)
- Home entry: Gripper.Close; GetBox

Gripper

- Closed entry: Grip = 0;
- OpenGrip
- Open entry: Grip = 1;
- GripBox
- Tighten entry: Grip = 2;
- after(delayMoveBox, sec)
- Grip entry: Robot.MoveBox
- DropBox
- Release entry: Grip = 1;
- [after(delayShipBox, sec)] {BeltOut.ShipBox}
- Close

BeltOut

- Empty entry: BeltOut_En = 0;
- [BeltOut_Box == 1]
- WaitRelease
- ShipBox
- On entry: BeltOut_En = 1;
- [BeltOut_LC == 0]
- BoxReady entry: Robot.GoHome; BeltOut_En = 0;
- [BeltOut_LC == 1]
- WaitClear after(delayBeltClear, sec)

Running 100% ode15s

Mechanics Explorers - Mechanics Explorer-youBot_Arm

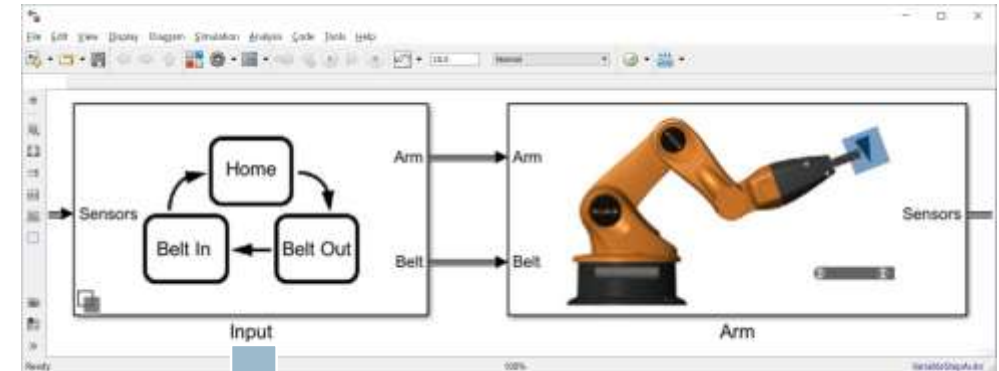
File Explorer Simulation View Tools Window Help

Mechanics Explorer-youBot_Arm

0% 1X Time 0

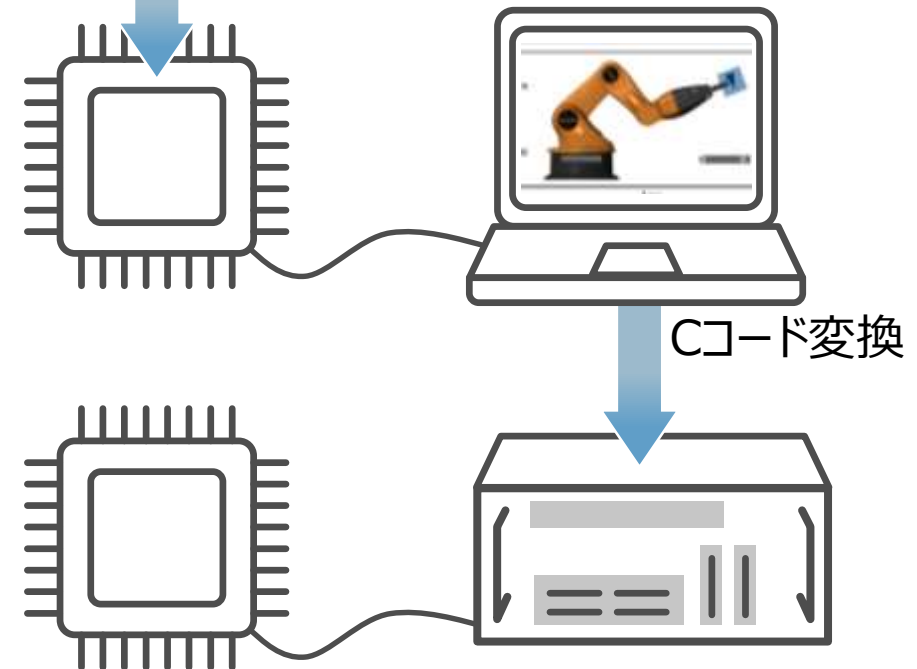
コントローラーのソフトウェアテスト

- アルゴリズムの自動コード生成
 - C Code, IEC 61131-3 Code
- 変換工程で、段階的にテストの効果を確認します。
 - 固定小数点
 - コントローラーのレイテンシ
- 同じプラントモデルの使用
 - 高価なハードプロトタイプ無しでテストが可能



Cコード変換

Processor-in-the-Loop (PIL)



Hardware-in-the-Loop (HIL)

まとめ

- Onshape と MathWorks により、エンジニアは、CAD モデルにマルチドメイン、動的シミュレーションを組み合わせることが出来ます。
- 結果:
 1. 最適化されたメカトロニクスシステム
 2. システム全体の品質向上
 3. 開発サイクルの短縮

