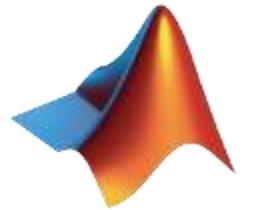


【三菱電機シーケンサ×MATLAB/Simulink】 人手作業の自動化/自律化を効率的に実現する ～ロバストなアルゴリズム・システム開発とは～

MathWorks Japan
アプリケーションエンジニアリング部



×



Agenda

1.自動化/自律化におけるトレンドと課題

2.ロバストなシステム開発のためのワークフロー

- MATLAB/Simulinkとは？
- 三菱電機シーケンサとの連携機能
- 両社の強みを生かしたロバストな開発フロー

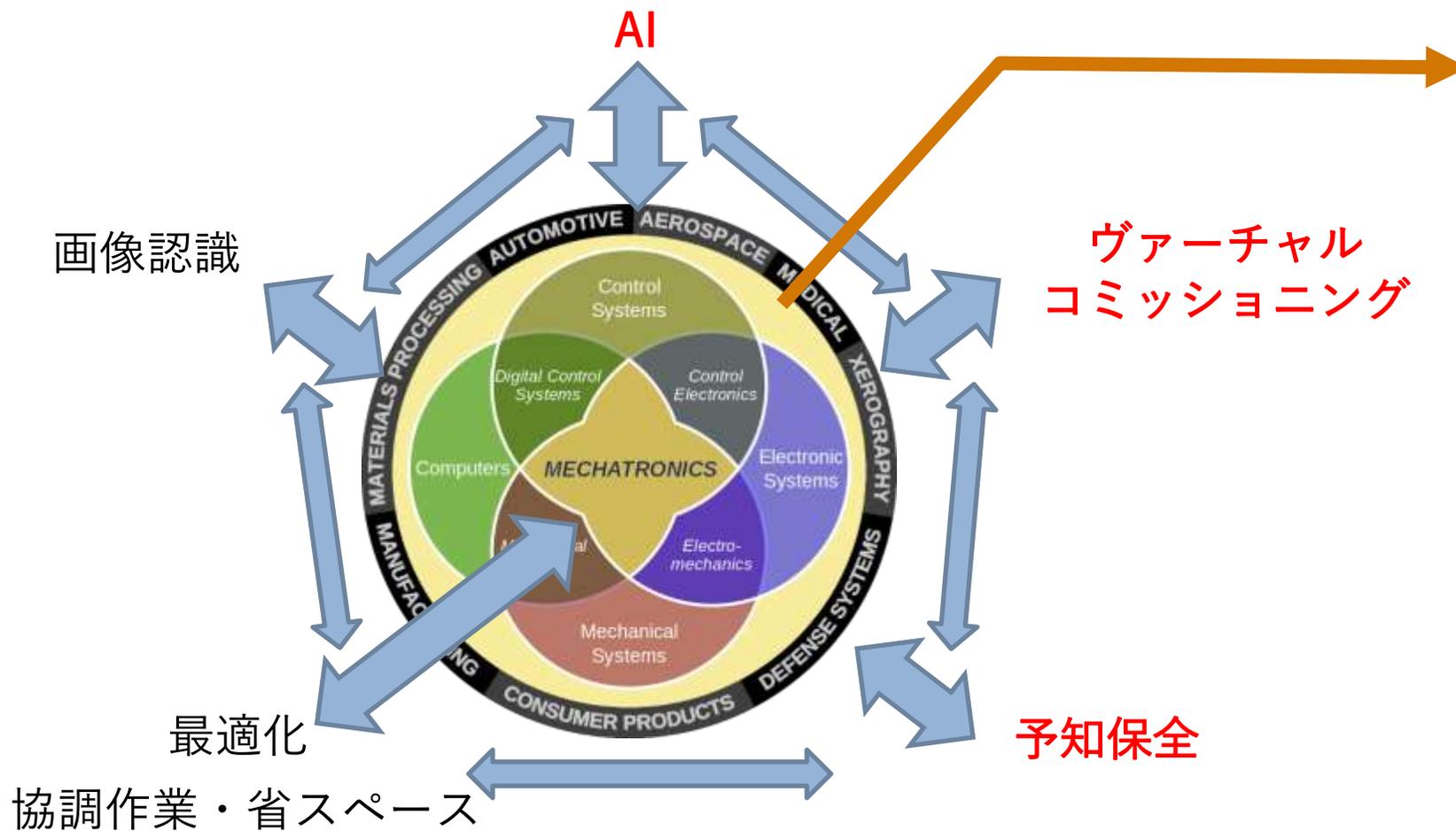
3.人手作業の自律化を想定したシステム開発例

トレンド: 高度・デジタル技術への期待

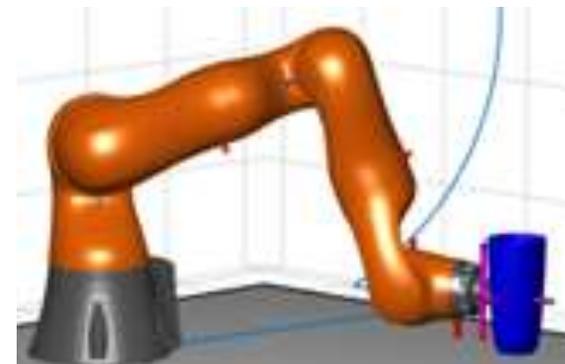
複雑・高度化したシステム開発

複合領域にまたがるシステムに更に新たな技術を付加

デジタル化 = モデルの活用



ヴァーチャル
コミッションング



デジタルツイン



自動化 ⇒ 自律化

自動化/自律化の例題：ピッキングタスク

デルタロボットの軌道計画



ロボット & 環境



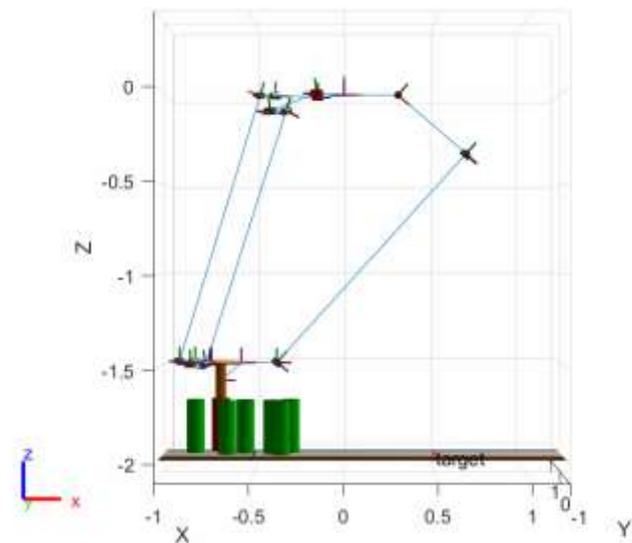
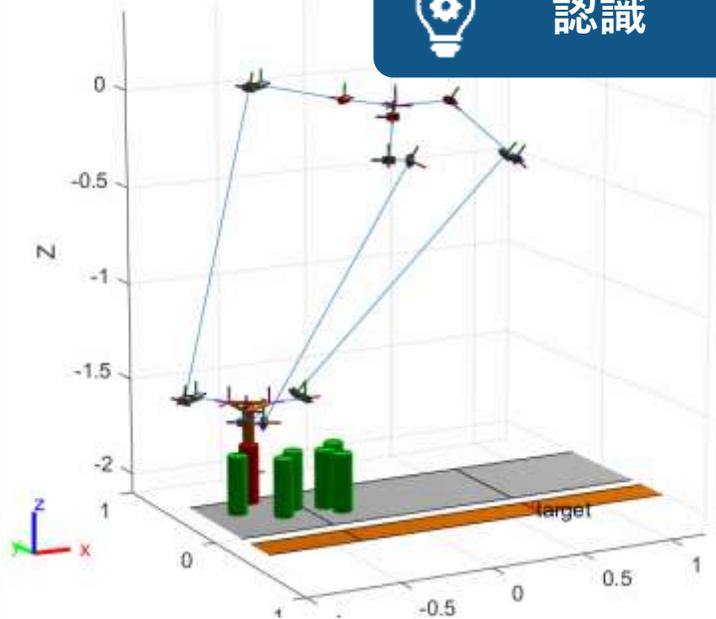
KRONES



センシング



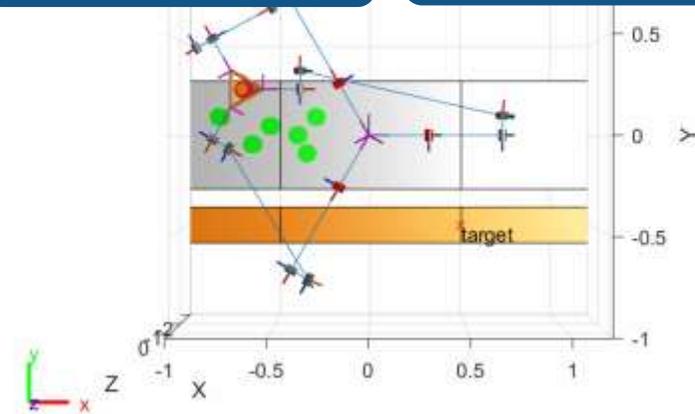
認識



計画 & 判断



制御



自律システム開発における課題とは？



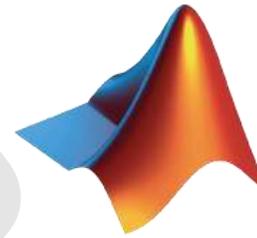
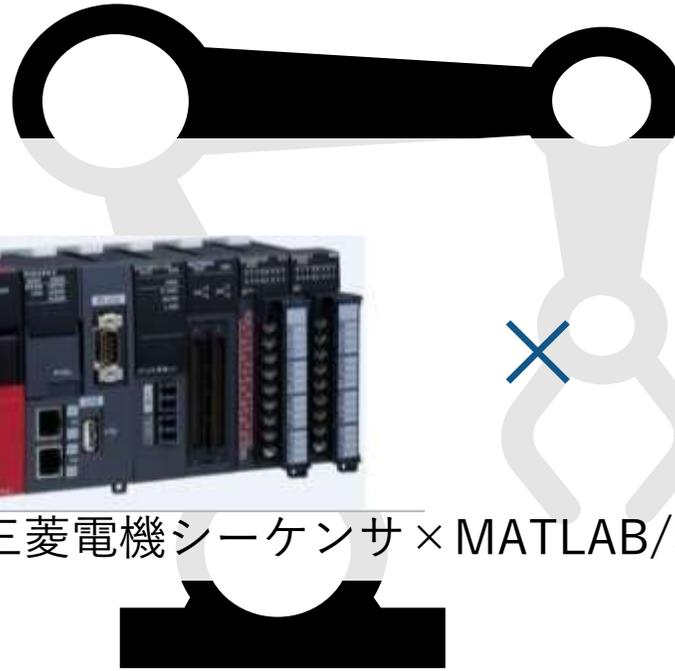
複合領域の
専門性



高度な
アルゴリズム



【三菱電機シーケンサ × MATLAB/Simulink】



開発環境



システムの
安全性



Agenda

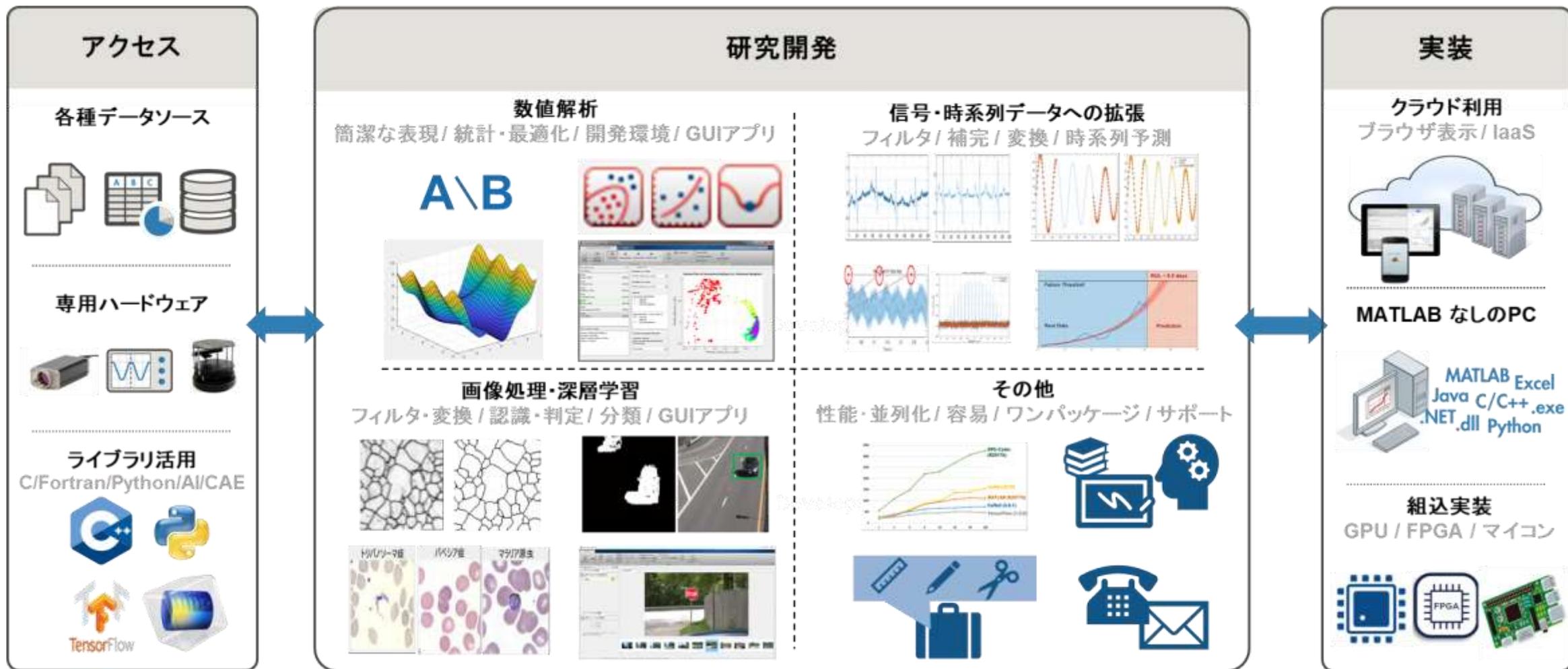
1.自動化/自律化におけるトレンドと課題

2.ロバストなシステム開発のためのワークフロー

- MATLAB/Simulinkとは？
- 三菱電機シーケンサとの連携機能
- 両社の強みを生かしたロバストな開発フロー

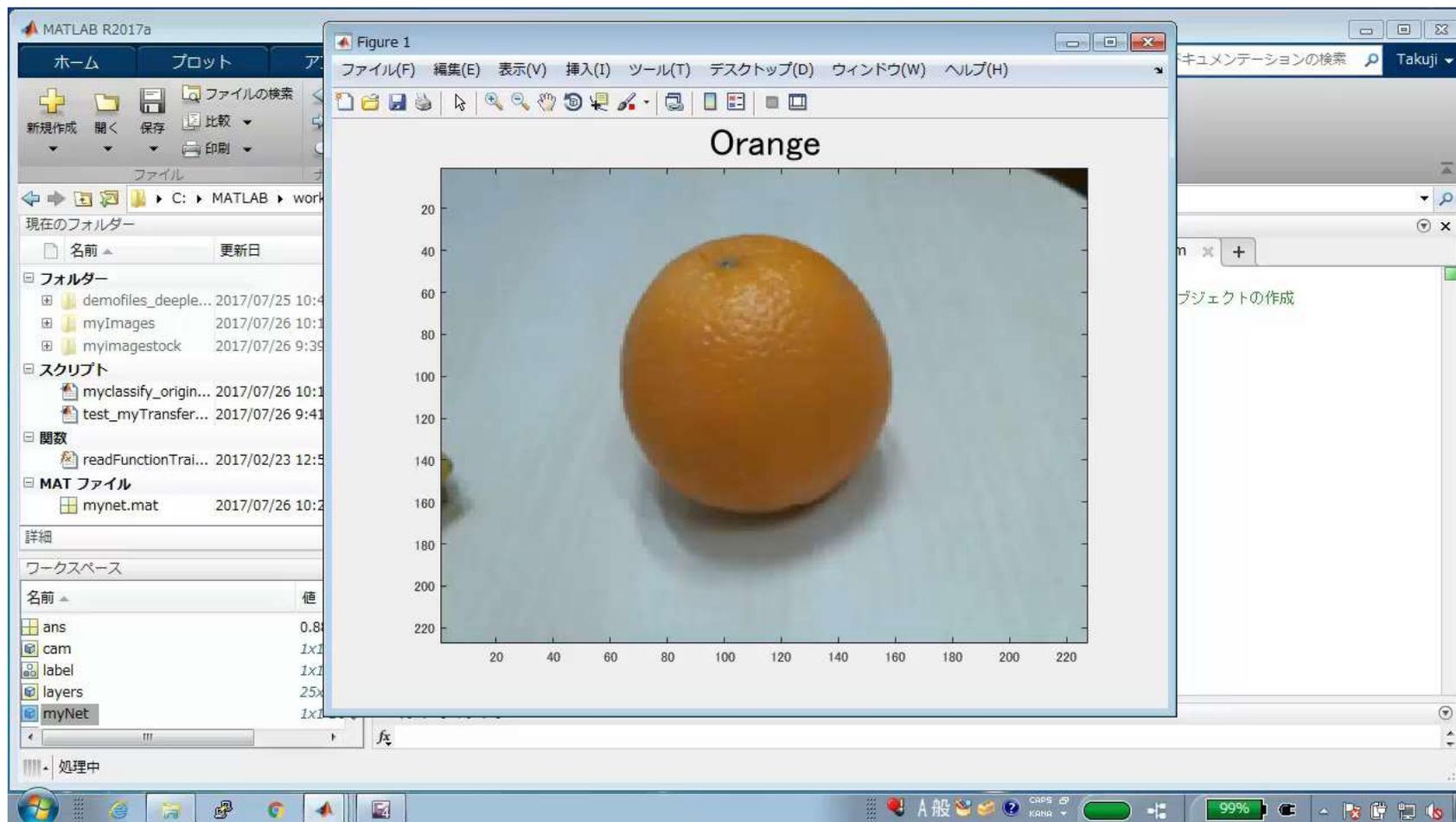
3.人手作業の自律化を想定したシステム開発例

MATLABが研究開発を加速



ディープラーニングによる物体認識

ディープラーニング：10行でできる転移学習 ～画像分類タスクに挑戦～



学習した種類：

- オレンジ
- みかん
- グレープフルーツ(ルビー)
- グレープフルーツ(ホワイト)
- レモン

学習画像数：各 20 枚

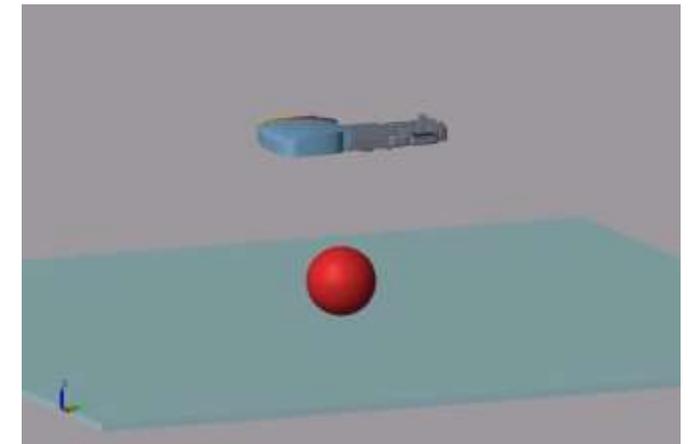
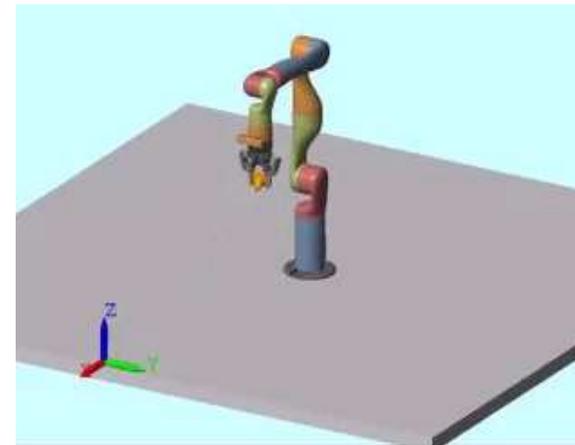
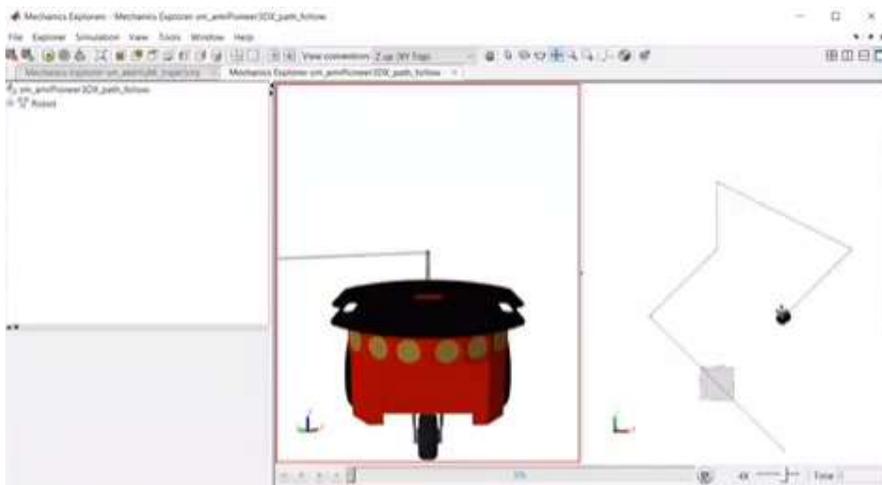
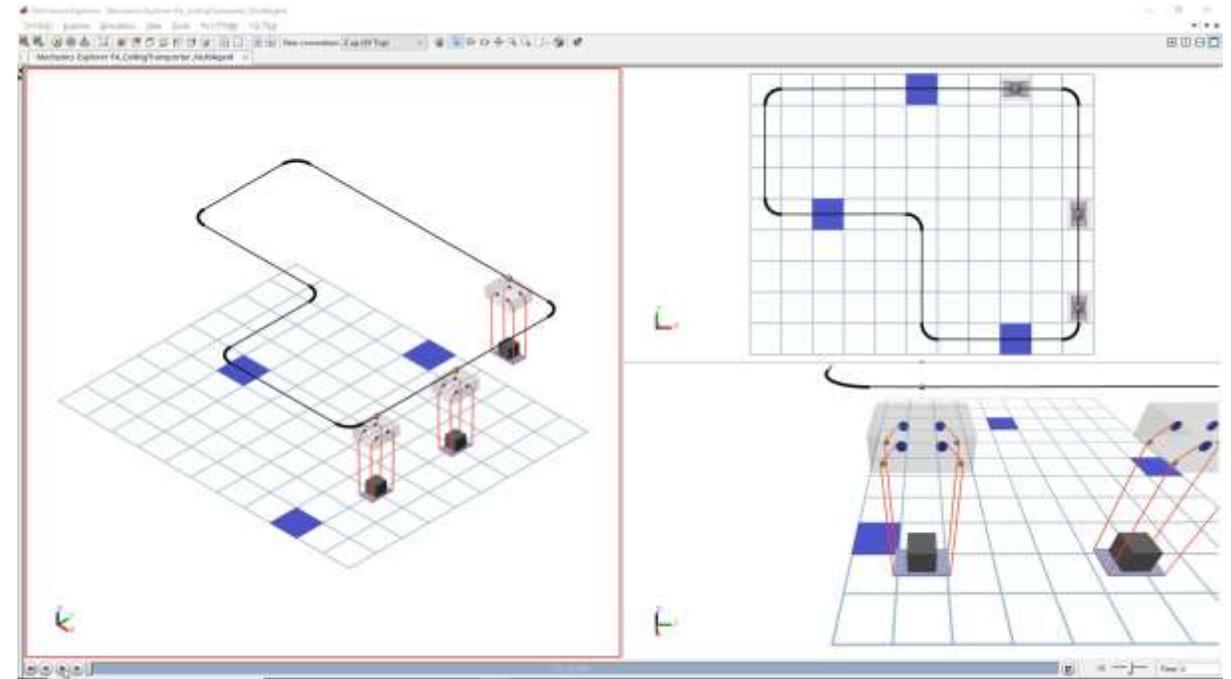
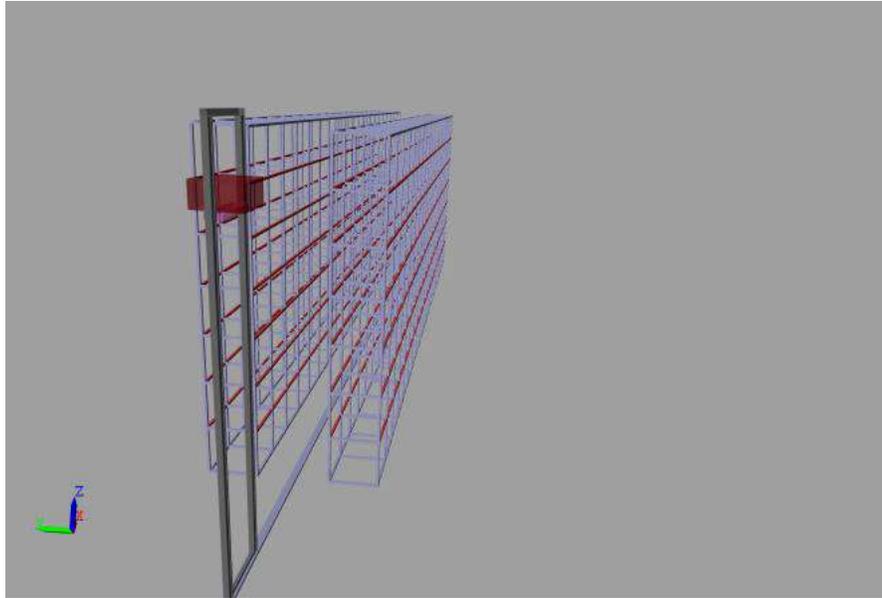
- 要件を満たすPC&MATLAB環境
 - 学習済みAlexnet
 - 画像セット
- で10行のコーディングで始められます

<https://www.mathworks.com/videos/deep-learning-transfer-learning-in-10-lines-of-matlab-code-1503069304524.html>

Simulinkとは？



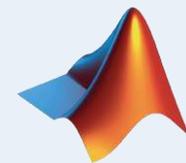
関連サンプルデモ



連携機能を活用した自律システムのロバストな開発フロー

STEP1.机上検討

MATLAB/Simulinkを活用して
アルゴリズムの選定や初期の検証を加速

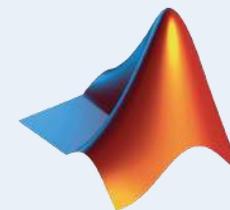


STEP2.プロトタイプ

三菱電機シーケンサ×MATLAB/Simulink
ロバストな試験環境を早期に構築

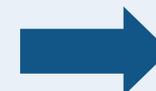
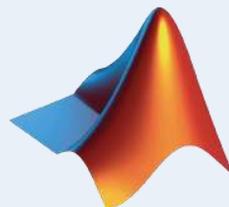


Ethernet通信等



STEP3. 実運用

コード生成技術によりアルゴリズムをシームレスに展開



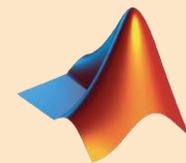
コード生成
実装



連携機能を活用した自律システムのロバストな開発フロー

STEP1.机上検討

MATLAB/Simulinkを活用して
アルゴリズムの選定や初期の検証を加速

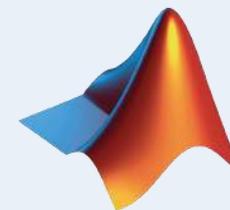


STEP2.プロトタイプ

三菱電機シーケンサ×MATLAB/Simulink
ロバストな試験環境を早期に構築

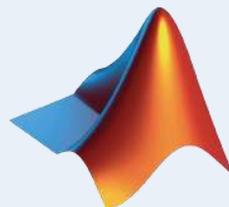


Ethernet通信等



STEP3. 実運用

コード生成技術によりアルゴリズムをシームレスに展開



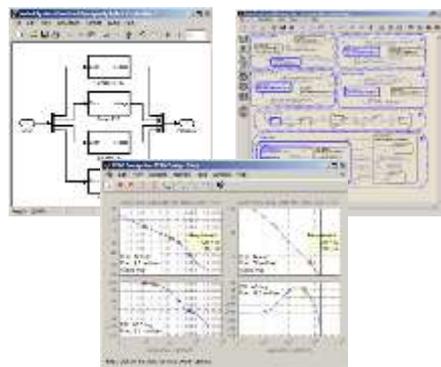
コード生成
実装



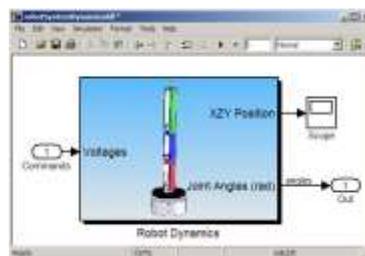
STEP1.机上検討

MATLAB/Simulinkを活用してアルゴリズムの選定や初期の検証を加速

仮想環境



ソフトウェアモデル

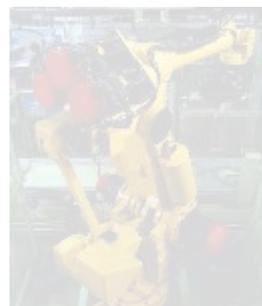


ハードウェアモデル

実環境



制御装置



制御対象

高度なアルゴリズムを早期に”机上検証”

- ✓ ライブラリや例題の活用
- ✓ UI・可読性の高い統合開発環境

シミュレーションによる”コスト削減”

- ✓ 試作・実験回数を削減
- ✓ 大規模試験の仮想化
- ✓ メカ・エレキ・制御含めたトレードオフ設計を促進

⇒ MATLAB/Simulinkの各種機能がシミュレーションの活用を支援・促進

STEP1.机上検討

MATLAB/Simulinkを活用してアルゴリズムの選定や初期の検証を加速

The image displays two software windows. The left window is Simulink Stateflow, showing a stateflow chart for a robot arm system. The chart is divided into four main sections: BeltIn, Robot, Gripper, and BeltOut. Each section contains several states and transitions, with transitions labeled with conditions and actions. The right window is Mechanics Explorers, showing a 3D simulation of the robot arm in a virtual environment. The simulation includes a base, joints, and a gripper mechanism. The interface includes a menu bar, a toolbar, and a status bar at the bottom.

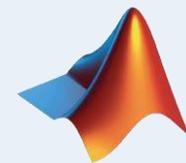
Stateflow Chart Details:

- BeltIn (1):**
 - Empty (entry: BeltIn_En = 0;)
 - Transition: [BeltIn_Box == 1]
 - On (entry: BeltIn_En = 1;)
 - Transition: [BeltIn_LC == 0]
 - BoxReady (entry: Robot.GetBox, BeltIn_En = 0;)
 - Transition: [BeltIn_LC == 1]
 - WaitClear (after: delayBeltClear.sec)
- Robot (2):**
 - StartHome (entry: Way = 0;)
 - GetBox (entry: Gripper.OpenGrip, Way = 1;)
 - Transition: after(delayGripBox.sec)
 - BeltIn (entry: Gripper.GripBox)
 - Transition: MoveBox
 - GoBeltOut (entry: Way = 2;)
 - Transition: after(delayDropBox.sec)
 - BeltOut (entry: Gripper.DropBox)
 - Transition: GoHome
 - GoingHome (entry: Way = 3;)
 - Transition: after(delayClose.sec)
 - Home (entry: Gripper.Close)
- Gripper (3):**
 - Closed (entry: Grip = 0;)
 - Transition: OpenGrip
 - Open (entry: Grip = 1;)
 - Transition: GripBox
 - Tighten (entry: Grip = 2;)
 - Transition: after(delayMoveBox.sec)
 - Grip (entry: Robot.MoveBox)
 - Transition: DropBox
 - Release (entry: Grip = 1;)
 - Transition: [after(delayShipBox.sec)] {BeltOut.ShipBox}
- BeltOut (4):**
 - Empty (entry: BeltOut_En = 0;)
 - Transition: [BeltOut_Box == 1]
 - WaitRelease
 - Transition: ShipBox
 - On (entry: BeltOut_En = 1;)
 - Transition: [BeltOut_LC == 0]
 - BoxReady (entry: Robot.GoHome, BeltOut_En = 0;)
 - Transition: [BeltOut_LC == 1]
 - WaitClear (after: delayBeltClear.sec)

連携機能を活用した自律システムのロバストな開発フロー

STEP1.机上検討

MATLAB/Simulinkを活用して
アルゴリズムの選定や初期の検証を加速

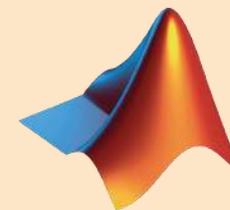


STEP2.プロトタイプ

三菱電機シーケンサ×MATLAB/Simulink
ロバストな試験環境を早期に構築

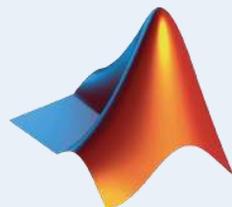


Ethernet通信等



STEP3. 実運用

コード生成技術によりアルゴリズムをシームレスに展開

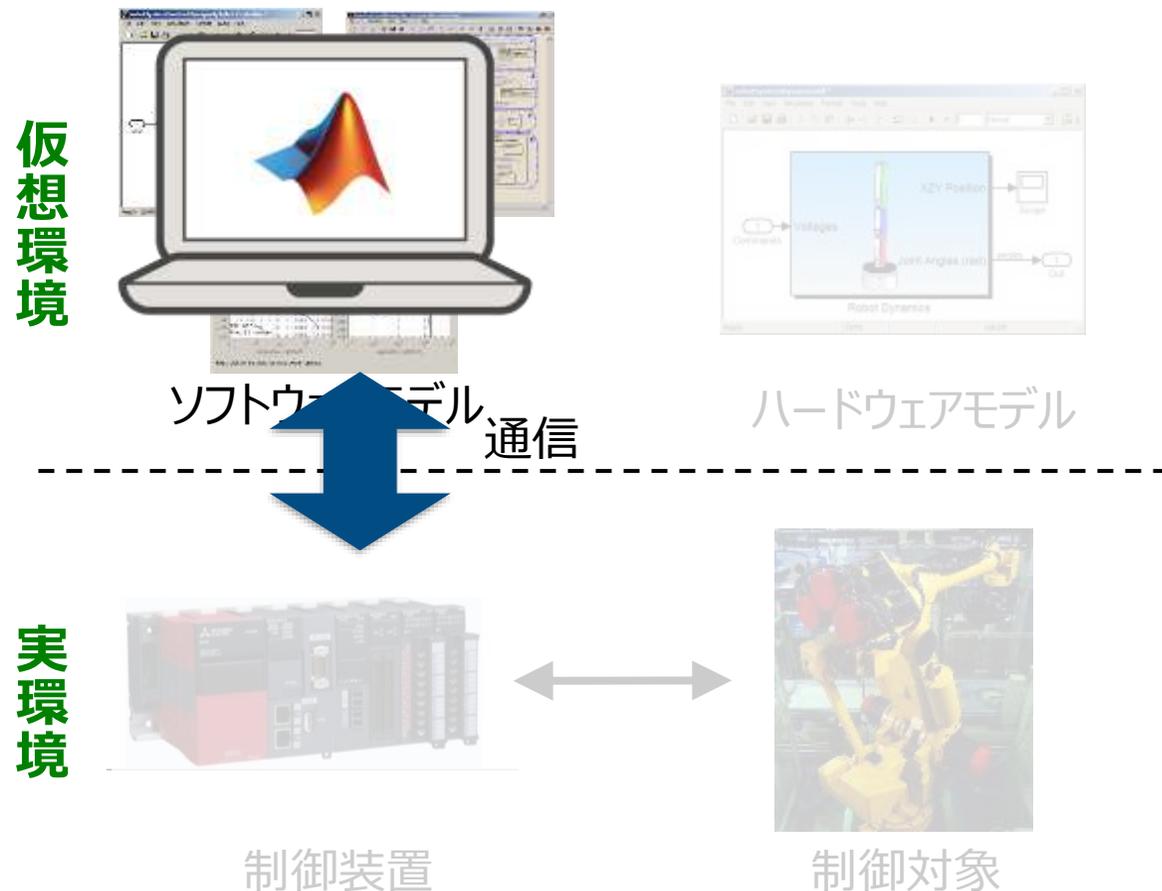


コード生成
実装



STEP2.プロトタイプ

1)MATLAB/Simulinkとシーケンサの通信ユニットを介した連携



高度なアルゴリズムを早期に”実機検証”

- ✓ PC上のMATLABを直接接続
- ✓ リアルタイム専用ハード(Speedgoat)の接続にも対応可能

三菱電機シーケンサの”柔軟な接続性”

- ✓ Ethernetを始めとした柔軟な接続性
- ✓ 設備で利用されている制御装置をそのまま活用可能

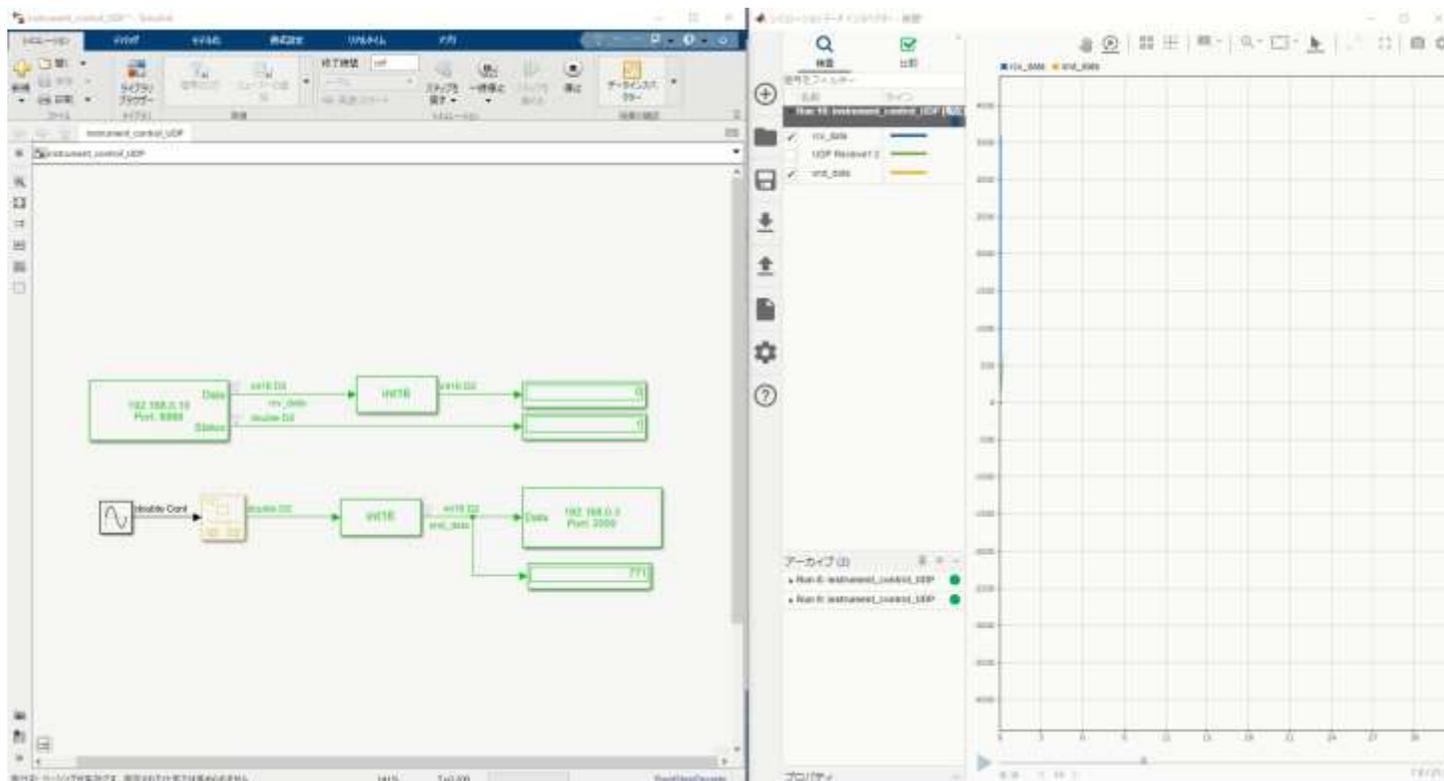
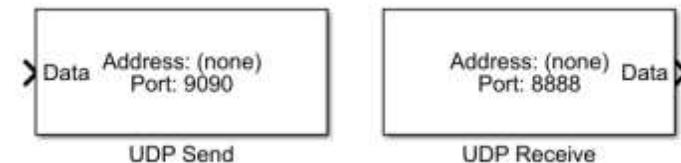
⇒MATLAB/Simulink、三菱電機シーケンサそれぞれ提供する柔軟なインターフェースにより、簡単に接続が可能。

STEP2.プロトタイプ

1) MATLAB/Simulinkとシーケンサの通信ユニットを介した連携



Ethernet (UDP)経由での
サイン波のループバック



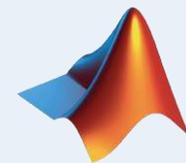
- ✓ Ethernet接続の基本的な設定だけで簡単に接続

※Ethernetユニット設定例
UDP通信(オープン待機)
ポートペアリングON
固定バッファ
無手順
生存確認なし

連携機能を活用した自律システムのロバストな開発フロー

STEP1.机上検討

MATLAB/Simulinkを活用して
アルゴリズムの選定や初期の検証を加速

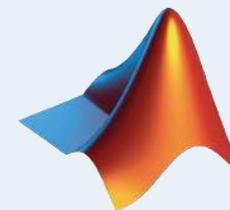


STEP2.プロトタイプ

三菱電機シーケンサ×MATLAB/Simulink
ロバストな試験環境を早期に構築

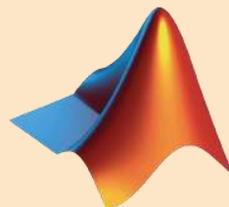


Ethernet通信等



STEP3. 実運用

コード生成技術によりアルゴリズムをシームレスに展開

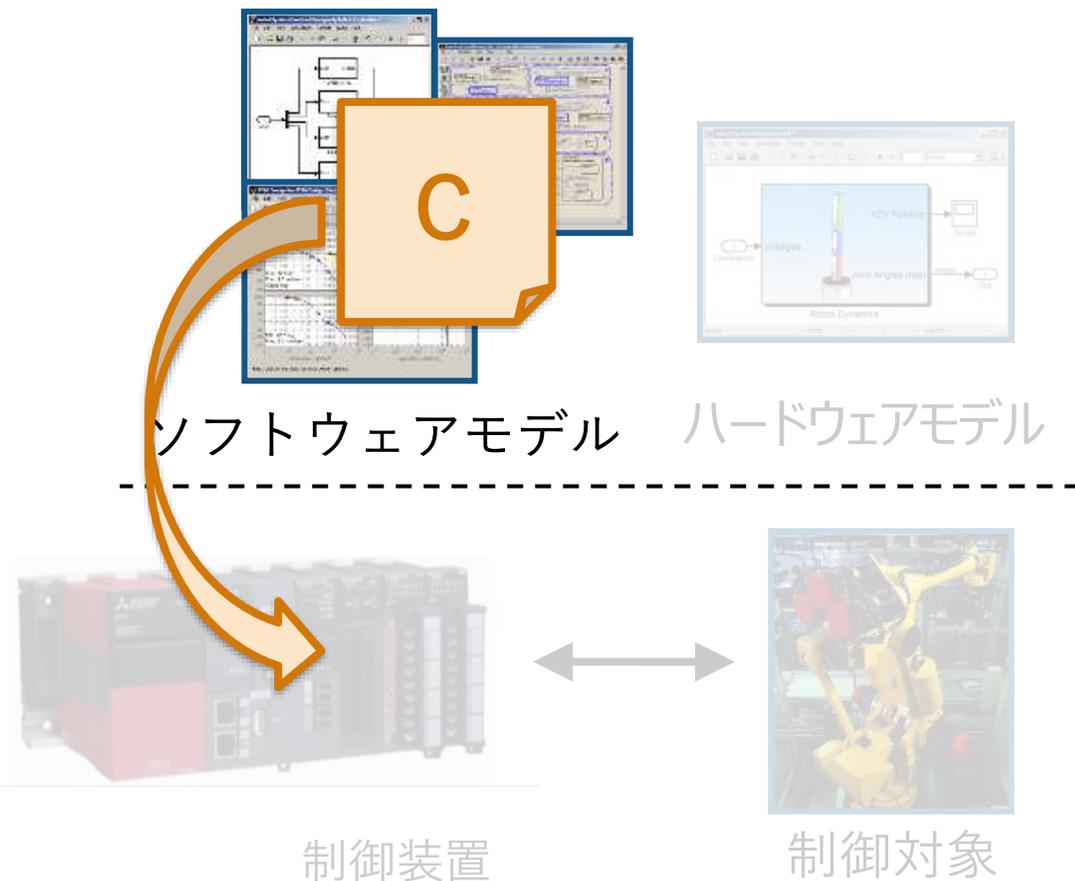


コード生成
実装



STEP2.プロトタイプ ⇒ STEP3.実運用

2)MATLAB/SimulinkのCコード生成機能を通じた C言語対応ユニットへのプログラムの実装



MATLAB/Simulinkからの自動Cコード生成

- ✓ STEP1,2で検証したアルゴリズムをシームレスに活用
- ✓ 自動コード生成機能を活用して、コーディングの手間やミスを最小化

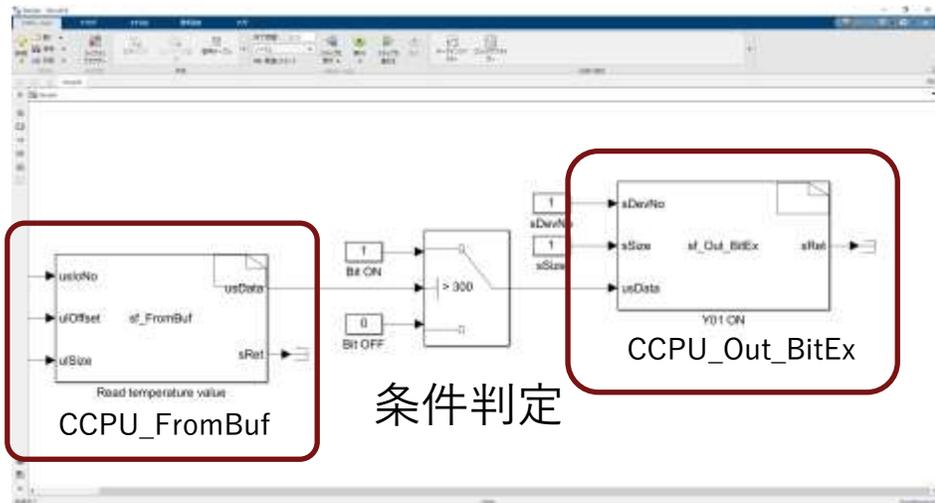
三菱電機シーケンサC言語対応ユニットの活用

- ✓ Simulinkブロックから生成されたC言語を直接利用可能

⇒MATLAB/SimulinkのCコード生成機能
三菱電機シーケンサのC言語対応ユニット
の協調によりシームレスな実装フローを実現

STEP2.プロトタイプ ⇒ STEP3. 実運用

2) MATLAB/SimulinkのCコード生成機能を通じた C言語対応ユニットへのプログラムの実装



C
自動生成



三菱電機シーケンサ専用命令
を含む状態で自動生成

CCPU_FromBuf

変数宣言

条件判定

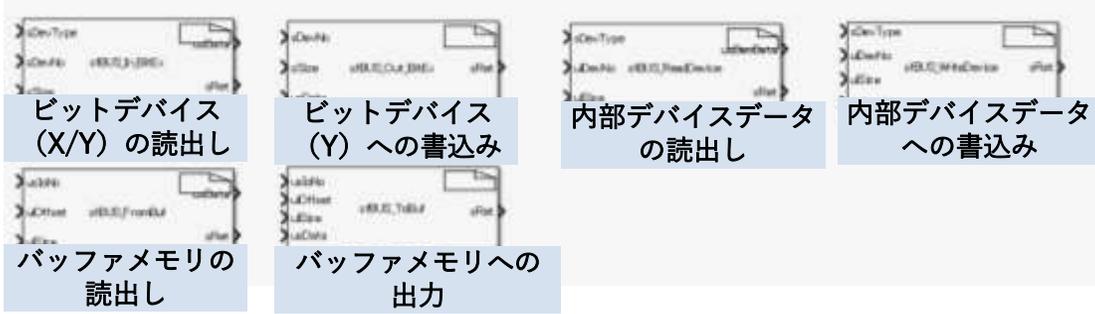
CCPU_Out_BitEx

```

25
26 #include "Se
27 #include "Se
28
29 /* Block sig
30 B_Sample_T Sample_B;
31
32 /* Real-time model */
33 static RT_MODEL_Sample_T Sample_M_;
34 RT_MODEL_Sample_T *const Sample_M = &Sample_M_;
35
36 /* Model step function */
37 void Sample_step(void)
38 {
39     /* S-function (sf_FromBuf): 'sRootz/Read temperature value' incorporates:
40     * Constant: 'sRootz/uOffset'
41     * Constant: 'sRootz/uSize'
42     * Constant: 'sRootz/usIoNo'
43     */
44     sf_FromBuf_Outputs_wrapper(&Sample_P.usIoNo_Value, &Sample_P.uOffset_Value,
45     &Sample_P.uSize_Value, &Sample_B.Readtemperaturevalue_01,
46     &Sample_B.Readtemperaturevalue_02);
47
48     /* Switch: 'sRootz/Switch' */
49     if (Sample_B.Readtemperaturevalue_01 > Sample_P.Switch_Threshold) {
50         /* Switch: 'sRootz/Switch' incorporates:
51         * Constant: 'sRootz/Bit_ON'
52         */
53         Sample_B.Switch = Sample_P.BitON_Value;
54     } else {
55         /* Switch: 'sRootz/Switch' incorporates:
56         * Constant: 'sRootz/Bit_OFF'
57         */
58         Sample_B.Switch = Sample_P.BitOFF_Value;
59     }
60
61     /* End of Switch: 'sRootz/Switch' */
62
63     /* S-function (sf_Out_BitEx): 'sRootz/Y01ON' incorporates:
64     * Constant: 'sRootz/sDevNo'
65     * Constant: 'sRootz/sSite'
66     */
67     sf_Out_BitEx_Outputs_wrapper(&Sample_P.sDevNo_Value, &Sample_P.sSite_Value,
68     &Sample_B.Switch, &Sample_B.Y01ON);
69
70
71
72
73
74
75
76
77
78
79
80
81
82
83
84
85
86
87
88
89
90
91
92
93
94
95
96
97
98
99
100

```

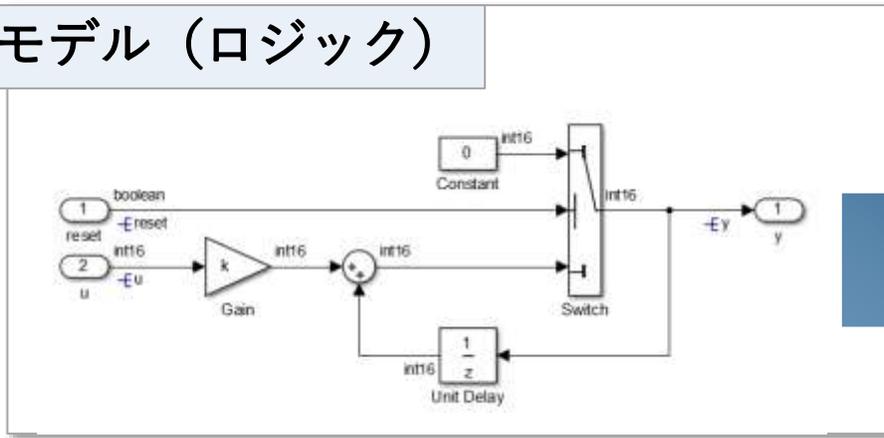
Simulink Library for MELSEC C言語コントローラ
<http://www.mitsubishielectric.co.jp/fa/products/cnt/plceng/download/library/>



ご参考：Simulink PLC Coder™： モデルからPLC用STコードを自動生成します

※ラダー図は特定のPLC IDEのみ対応

モデル (ロジック)



コード
自動生成

STコード

```

FUNCTION_BLOCK rst_cntr
VAR_INPUT
  ssMethodType: SINT;
  reset: BOOL;
  u: INT;
END_VAR
VAR_OUTPUT
  y: INT;
END_VAR
VAR
  UnitDelay_DSTATE: INT;
END_VAR

CASE ssMethodType OF
0:
  UnitDelay_DSTATE := 0;
1:
  IF reset THEN
    y := 0;
  ELSE
    y := DINT_TO_INT(INT_TO_DINT(2 * u) +
      INT_TO_DINT(UnitDelay_DSTATE));
  END_IF;
  UnitDelay_DSTATE := y;
END_CASE;
END_FUNCTION_BLOCK
  
```

コード仕様

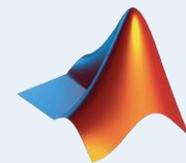


- ターゲットPLC IDE
- 変数/定数属性
 - 名前、記憶クラス、データ型
- テストベンチ用コード有無

連携機能を活用した自律システムのロバストな開発フロー

STEP1.机上検討

MATLAB/Simulinkを活用して
アルゴリズムの選定や初期の検証を加速

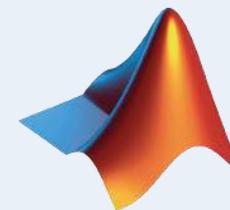


STEP2.プロトタイプ

三菱電機シーケンサ×MATLAB/Simulink
ロバストな試験環境を早期に構築



Ethernet通信等

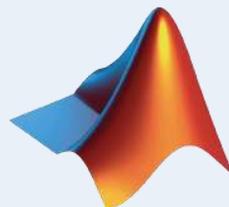


実運用のその前に



STEP3. 実運用

コード生成技術によりアルゴリズムをシームレスに展開

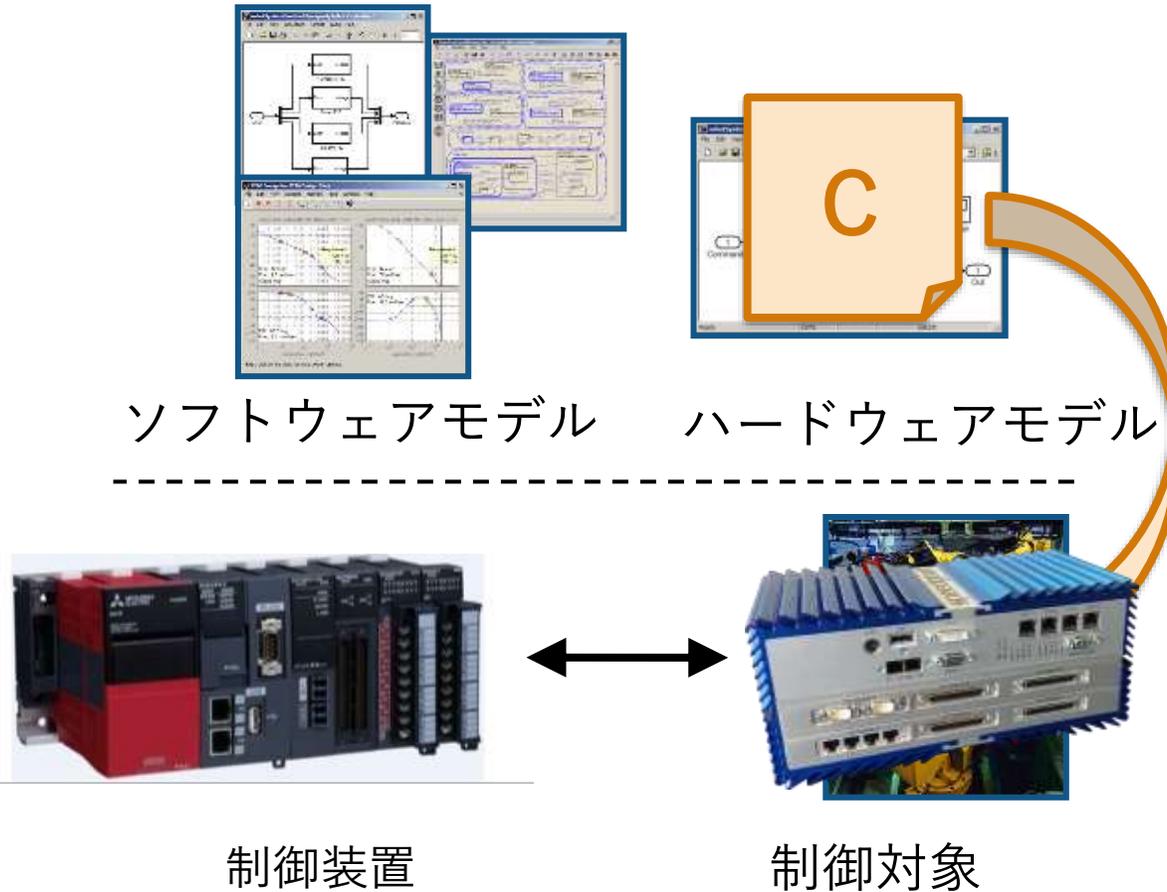


コード生成
実装



STEP2.プロトタイプ ⇒ STEP3.実運用

実機レス開発を促進するHardware In the Loop



制御対象モデルを実機の代替として活用

- ✓ 実機模擬用ハードに制御対象モデルをコードし実装
- ✓ テスト対象はハンドコード/自動生成コードどちらもOK
- ✓ 本格運用前の仮試験や故障・異常系処理のテストに有用
- ✓ 特に大規模設備や、危険な試験にて適用効果大

⇒MATLAB/SimulinkのCコード生成機能
ハードウェアモデルを再利用して、
実機レス試験をさらに加速

Agenda

1.自動化/自律化におけるトレンドと課題

2.ロバストなシステム開発のためのワークフロー

- MATLAB/Simulinkとは？
- 三菱電機シーケンサとの連携機能
- 両社の強みを生かしたロバストな開発フロー

3.人手作業の自律化を想定したシステム開発例

連携機能を活用したロバストな開発フロー～実践～

STEP1.机上検討

MATLAB/Simulinkを活用して
アルゴリズムの選定や初期の検証を加速

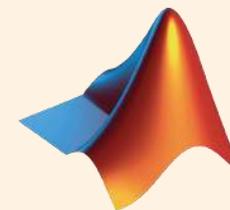


STEP2.プロトタイプ

三菱電機シーケンサ×MATLAB/Simulink
ロバストな試験環境を早期に構築

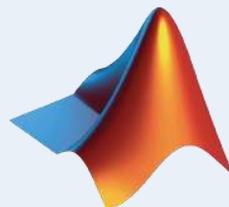


Ethernet通信等



STEP3. 実運用

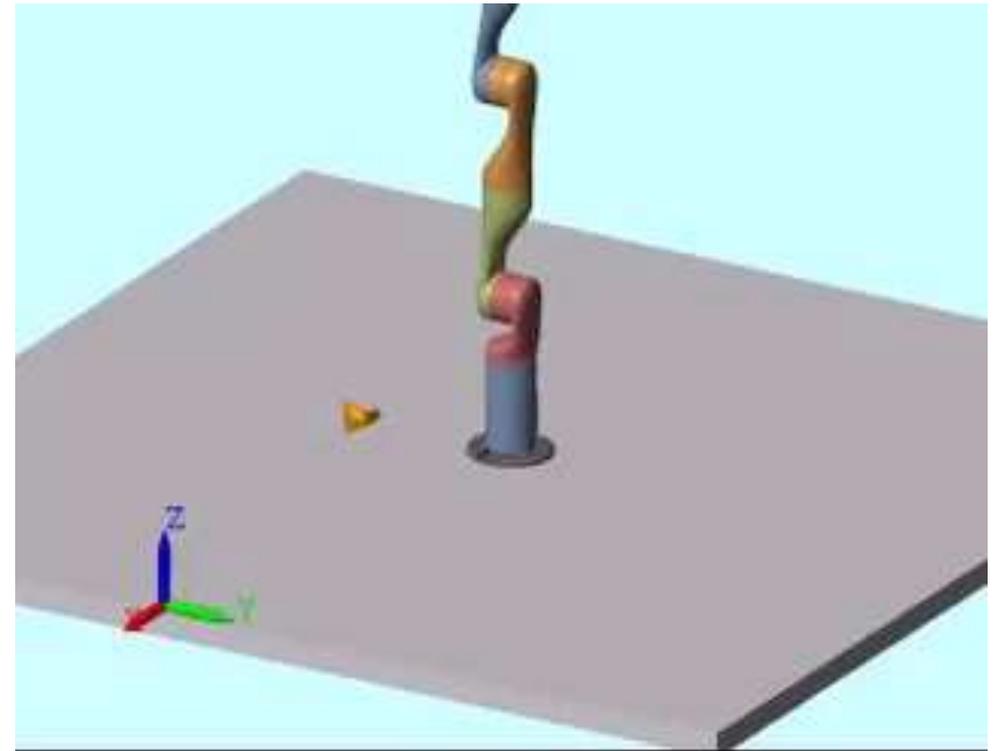
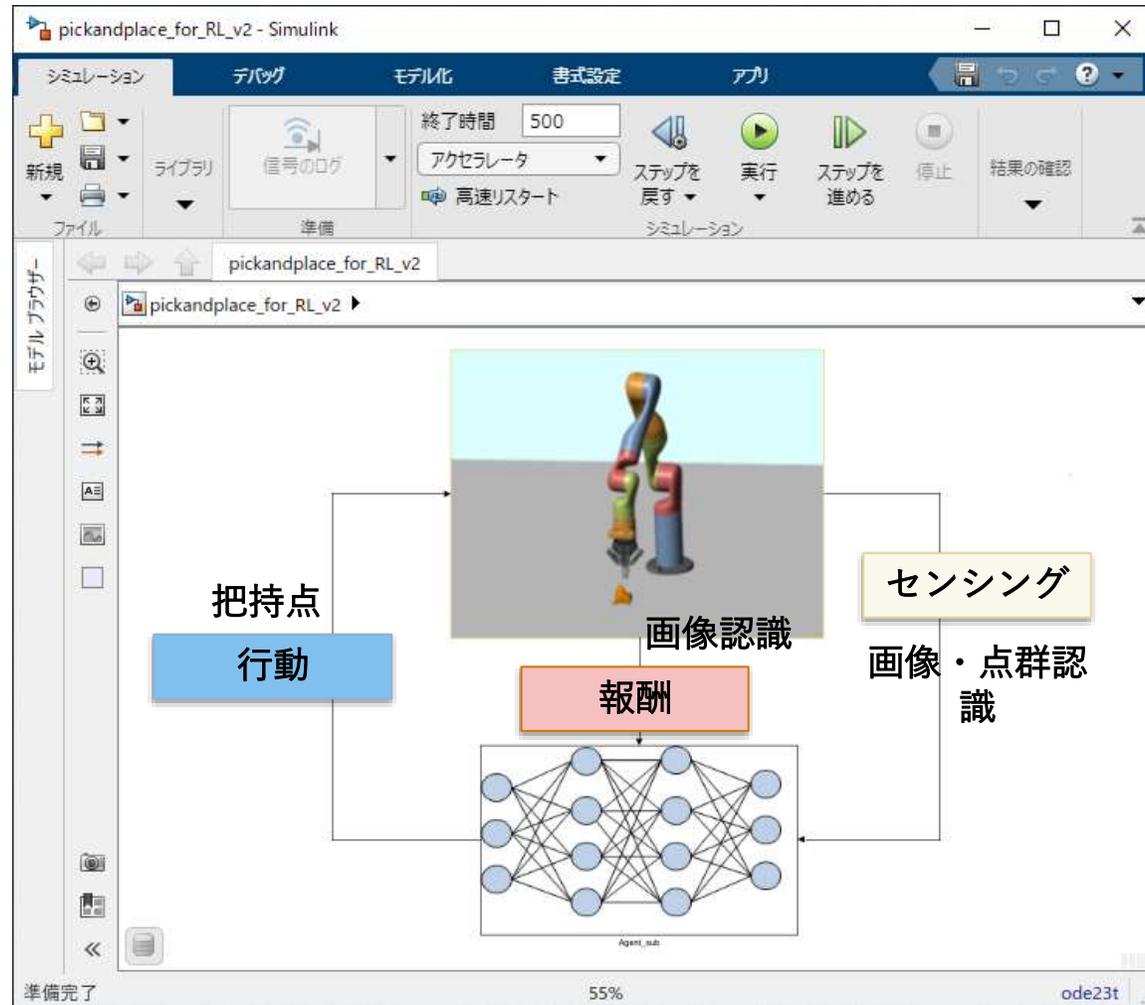
コード生成技術によりアルゴリズムをシームレスに展開



コード生成
実装

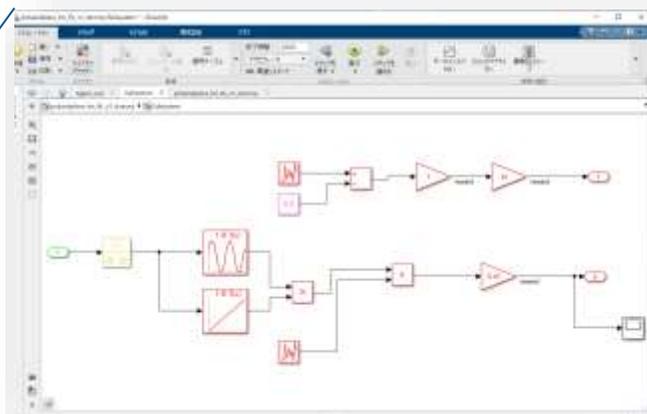
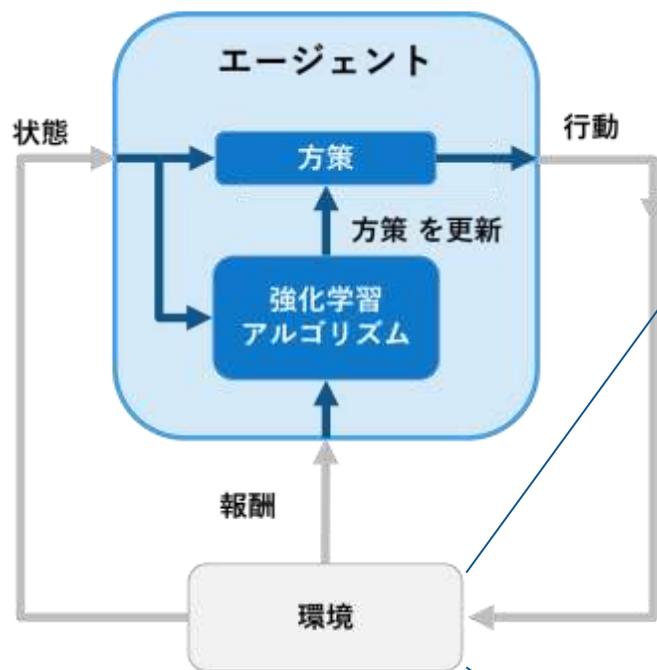


自律システムの例(強化学習によるチューニング)



高度自律システム開発 : Reinforcement Learning

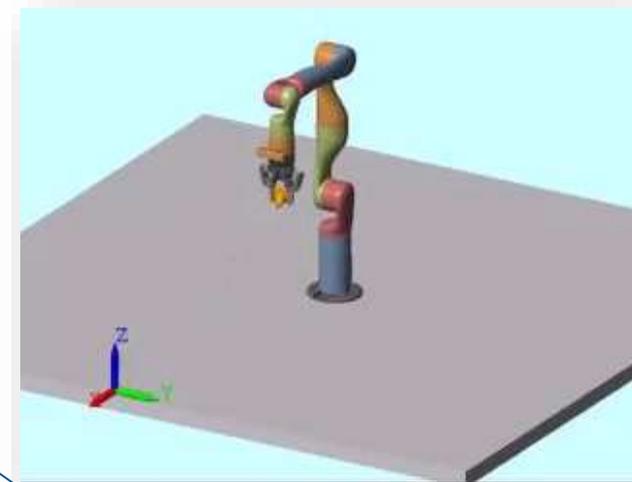
モデルの詳細度とパラメータ設計



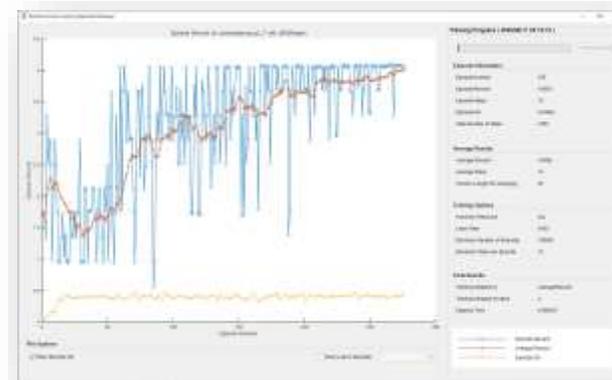
類似問題(高速)



パラメータ設計検討



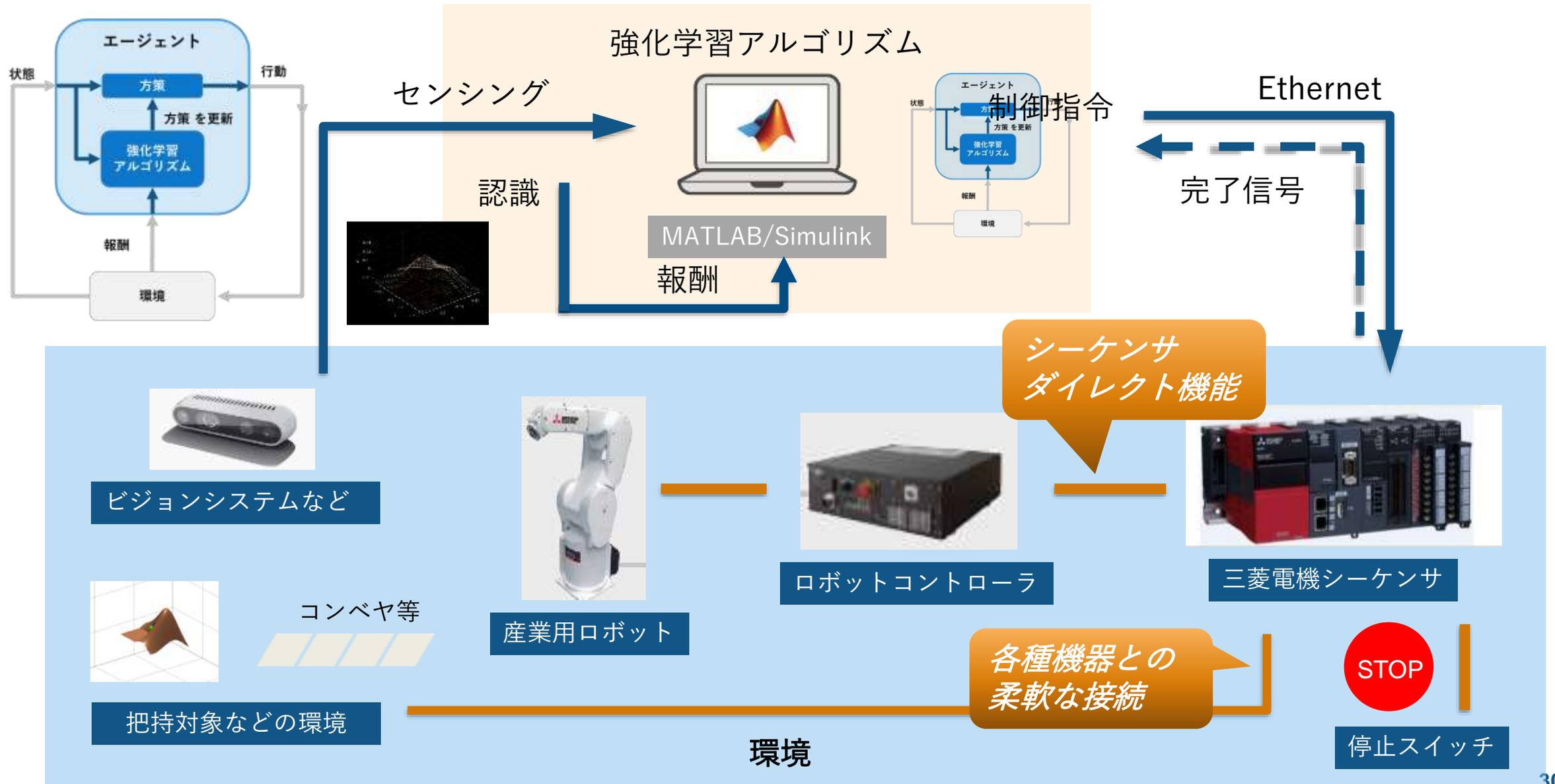
厳密な問題



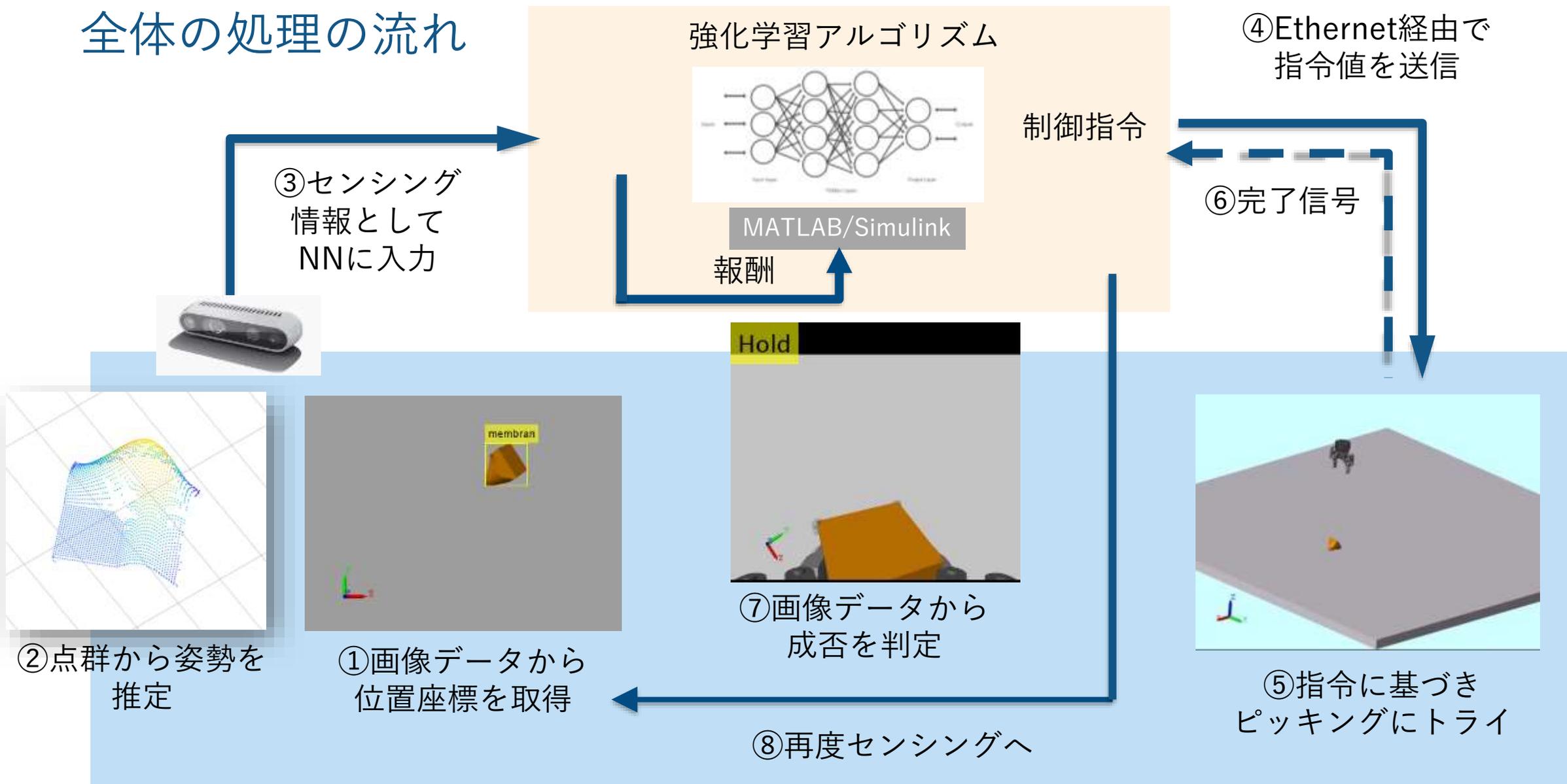
実問題へのトライ

試験設備のシステム構成案

画像は三菱電機FAサイトより引用
<http://www.mitsubishielectric.co.jp/news/2017/0302-b.html>



全体の処理の流れ



ポイント

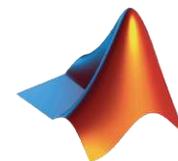
STEP2.プロトタイプ

三菱電機シーケンサ × MATLAB/Simulink
ロバストな試験環境を早期に構築

MATLAB/Simulinkのアルゴリズム開発の強みと、
柔軟な接続性など三菱電機シーケンサの強みを活用



×

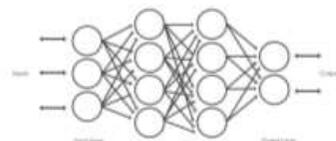


学習したネットワークの実運用

STEP3.実運用

コード生成技術によりアルゴリズムをシームレスに展開

-学習したNeural networkなどの高度アルゴリズムのコード生成および実装に対応



強化学習
認識



コード生成
実装



まとめ：自律システムにおける課題とは？



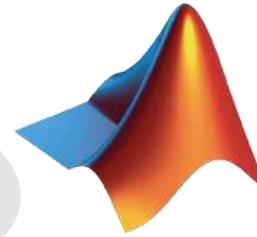
複合領域の
専門性



開発環境



×



【三菱電機シーケンサ × MATLAB/Simulink】



高度な
アルゴリズム



システムの
安全性



Accelerating the pace of engineering and science

© 2022 The MathWorks, Inc. MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.