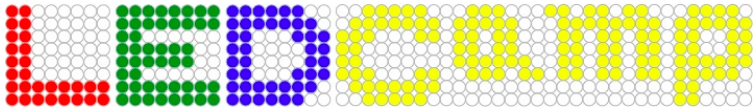
LED-Camp3開発教材の解説 v2

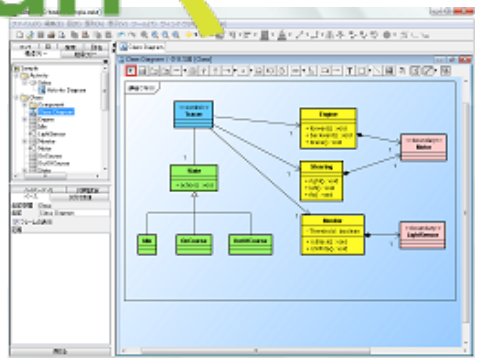


目次

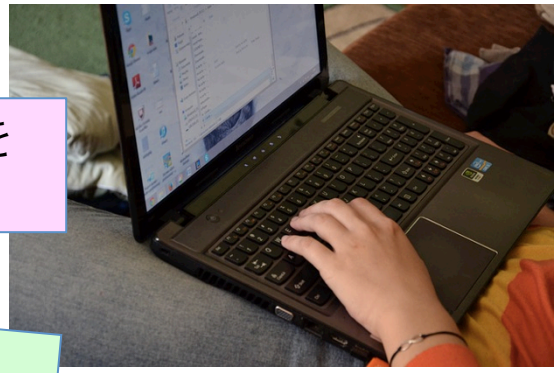
- ハードウェア仕様
  - iRobot Create2
  - GR-SAKURA
  - 超音波距離センサURM37
  - 無線モジュールXBee
- ソフトウェア仕様
  - 各種イベントの仕様と利用方法
- 開発のヒント
  - デバッグとテスト
  - 検出値／制御値の補正

開発の流れ

astah

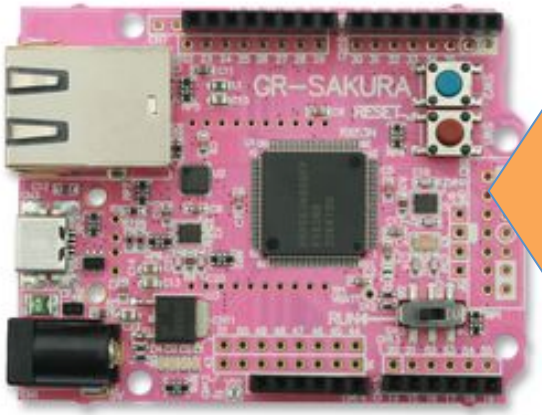


制御ソフトを
モデル設計



コードの
自動生成

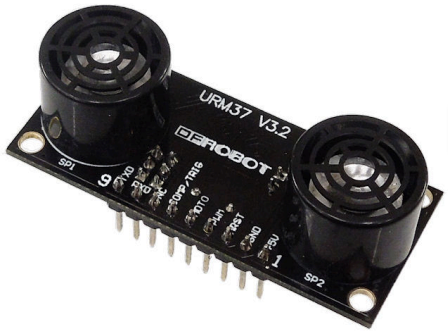
デバッグデータ
の送信



シリアル通信で
動作を制御

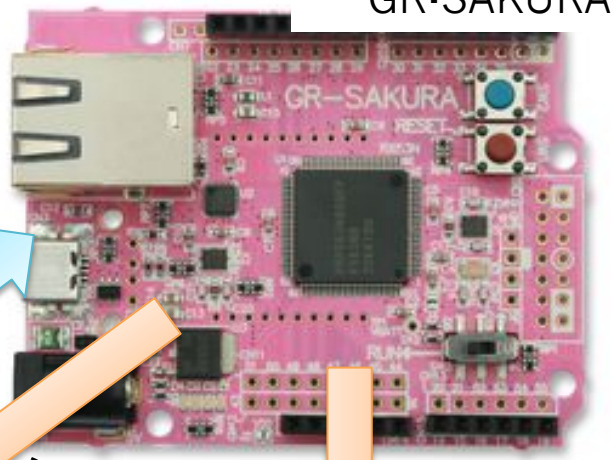
iRobot

開発教材のシステム構成



超音波距離センサ
URM37 V3.2

GR-SAKURA



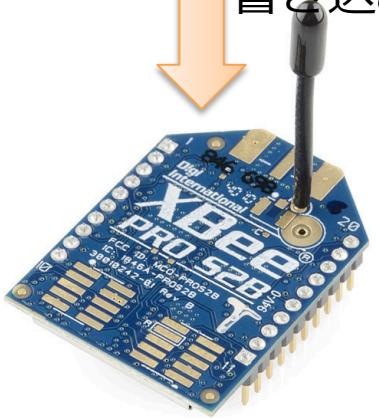
物体との距離

検知系の状態

制御コマンド

デバッグデータ
書き込み

iRobot Create2

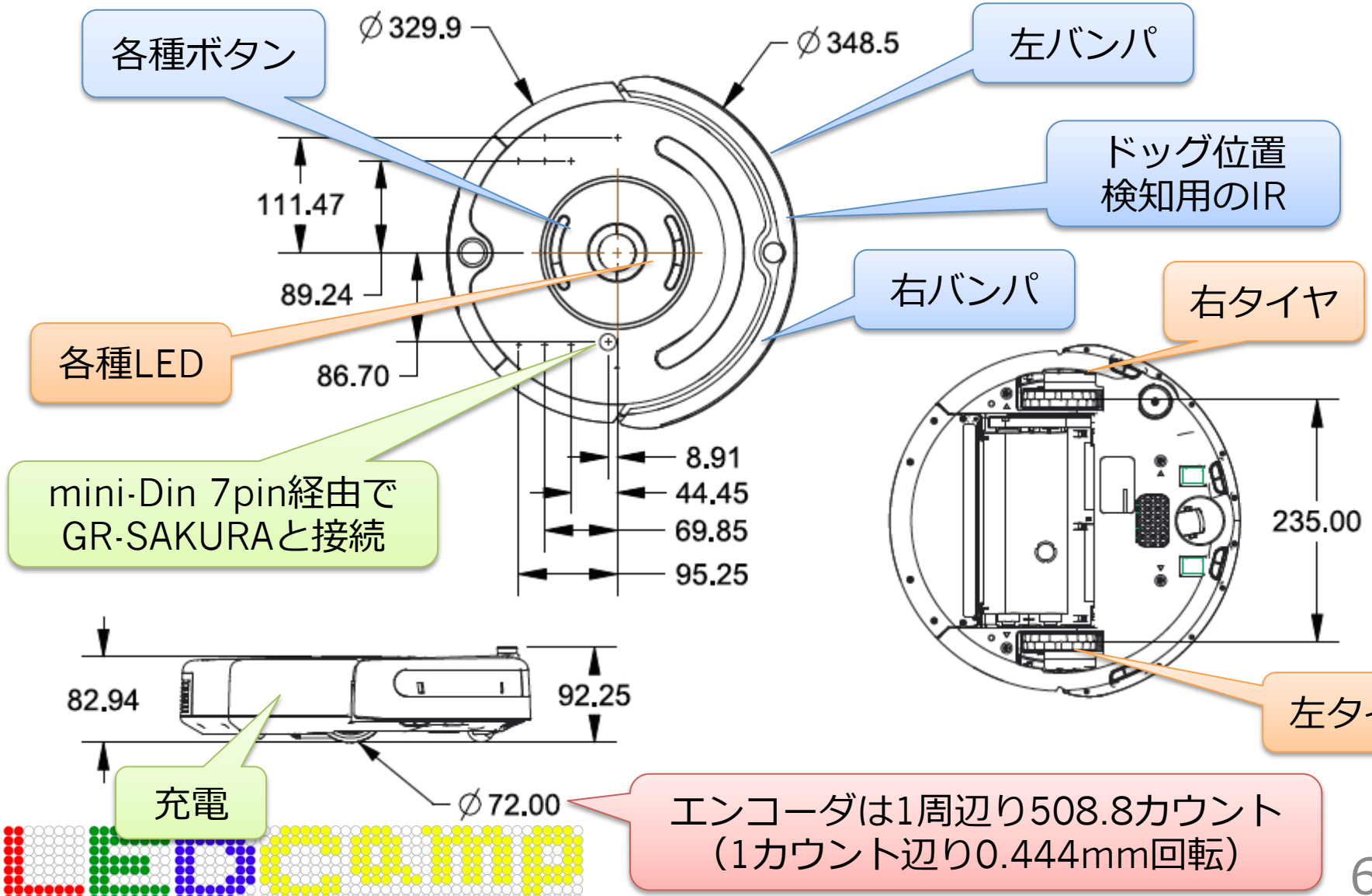


無線モジュール
XBee-Pro ZB S2B

iRobot Create2

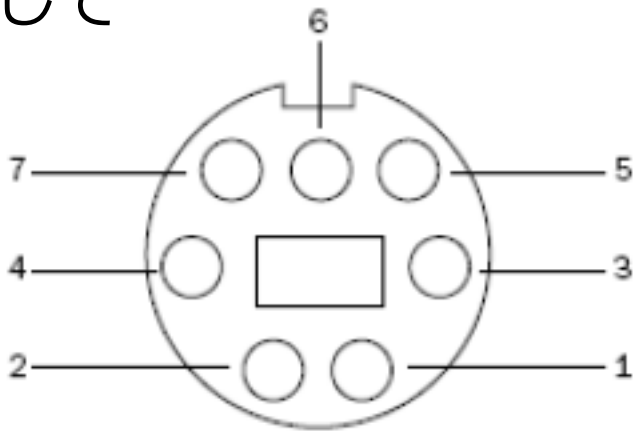
- 掃除機型ロボット
  - Roomba 600とHW・API互換（ブラシを付けたらお掃除可能）
- 主要な検知系
  - 左右バンパ
  - 左右タイヤのエンコーダ
  - IRセンサ（ドッグ位置検知）
  - クリフセンサ
  - ボタン×8
- 主要な制御系
  - 左右タイヤ
  - ブラシ×3
  - LED×4
  - 7セグLED×4桁
  - スピーカー（MIDI形式）

Create2の外観・サイズ [mm]



GR-SAKURAとの接続

- mini-Din 7pinコネクタ
  - 8ビットのデータをOIコマンドとして シリアル通信でやりとり
  - RxD,TxDはGR-SAKURAの Serial1（D0,D1）に接続
    - コネクタシールドを配布, ピン接続は固定
  - ボーレット値は115,200
    - 19,200に変更も可能



Pin	Name	Description
1	Vpwr	Roomba battery + (unregulated)
2	Vpwr	Roomba battery + (unregulated)
3	RXD	0 – 5V Serial input to Roomba
4	TXD	0 – 5V Serial output from Roomba
5	BRC	Baud Rate Change
6	GND	Roomba battery ground
7	GND	Roomba battery ground

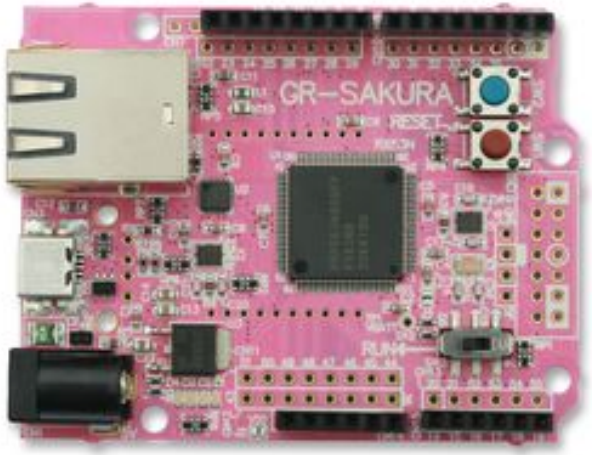
Create2の動作モード

- Off Mode
- Passive Mode
  - 3種類のボタン（Clean, Spot, Dock）に応じて動作する
  - センサ値の取得はできる
- Safe Mode
  - 制御コマンドのやり取りによって，センサ値の取得および動作の制御ができる
  - 落下検知，脱輪または充電時には緊急停止する
- Full Mode
  - 制御コマンドのやり取りによって，センサ値の取得および動作の制御ができる
  - 制御コマンド以外ではバッテリ全放電しないとOffにならないので注意する

Create2API

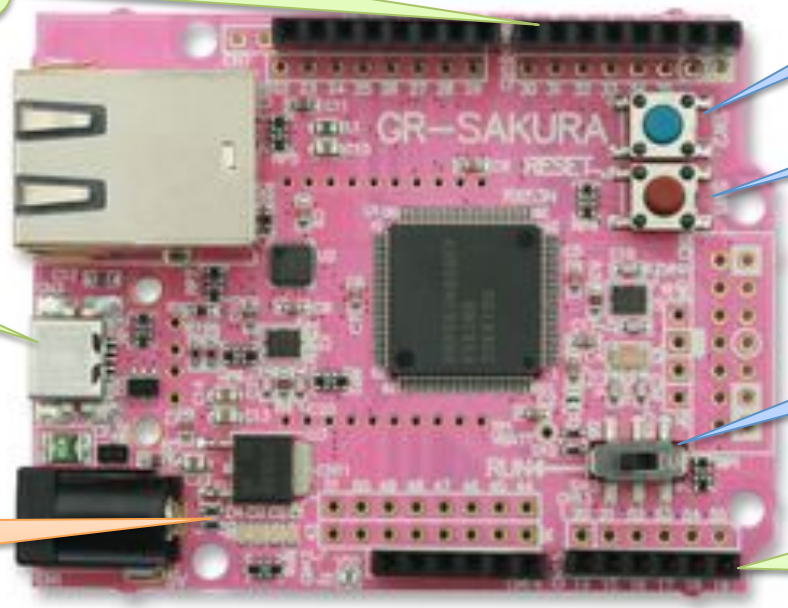
- 詳細は資料集末尾のAPIリファレンスにて
- 初期化・スタート制御系
  - Create OI制御コマンドの通信を開始する
  - 動作モードを変更する
- モータ駆動系
  - Create2の動作（タイヤ回転量）を制御する
  - 単位と外乱の影響に注意する（詳細は後述）
- 本体センサ系
  - 各種検知系の状態や値を取得する
- 本体ボタン・音楽・LED表示系
  - 主にデバッグ用途として使用する

GR-SAKURA



- 「がじぇっとるねさす」から販売されているリファレンスボード
- Arduinoとピン互換有り
  - Arduino：CC BY-SA（表示・継承）で公表されているオープンソース・ハードウェア
- ハードウェア仕様
  - ルネサス社製32ビットCPU RX63N
    - 動作周波数96MHz（単精度FPU，乗除算器内蔵）
  - FlashROM：1MB，RAM：128KB
  - データ用Flash：32KB
  - 動作電圧：5V（CPUは3.3V）
  - デジタルI/Oピン：14本（うち6本はPWM出力可）
  - アナログ入力ピン：6本（デジタルI/Oとしても使用可）
    - 拡張コネクタの追加でRX63Nの100ピンのほとんどを使用可

GR-SAKURAの外観・操作方法



デジタルI/Oピン
右からD0～D13
D0,D1はSerial1と併用

ユーザ
スイッチ

リセット
スイッチ

miniUSB
コネクタ

スライドスイッチ
（RUNになっている
ことを確認する）

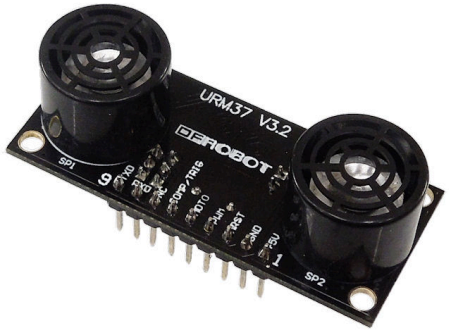
LED
右からLED1～4

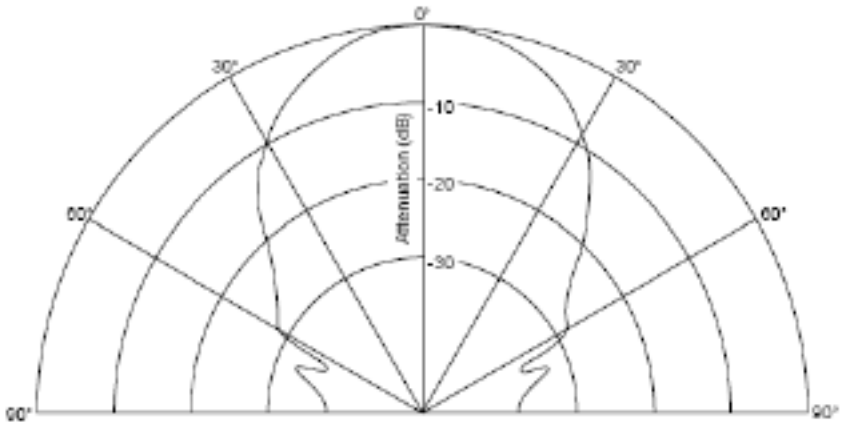
アナログ入力ピン
左からA0～A5

USB接続時にリセットスイッチを押すとPCからマスストレージに見える
バイナリをドラッグ＆ドロップすることでプログラムを書き込める

URM37 V3.2

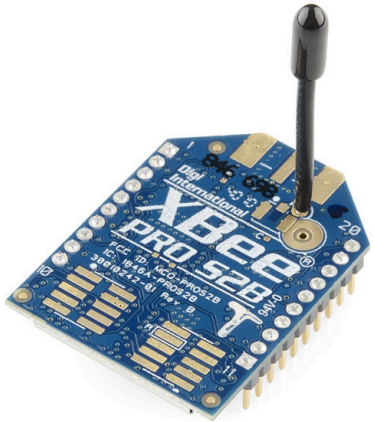
- 超音波の送受信で物体までの距離を測るセンサ
- 主要なスペック
  - 測定範囲：4cm～300cm
  - 分解能：1cm
  - パルス幅の波形を出力
- 注意点
  - 測定可能な範囲は正面だけではない↓
  - APIの返値は0～100cmに制限されている
    - 遠距離だと検出値のぶれが大きくなるため





XBee-PRO ZB S2B

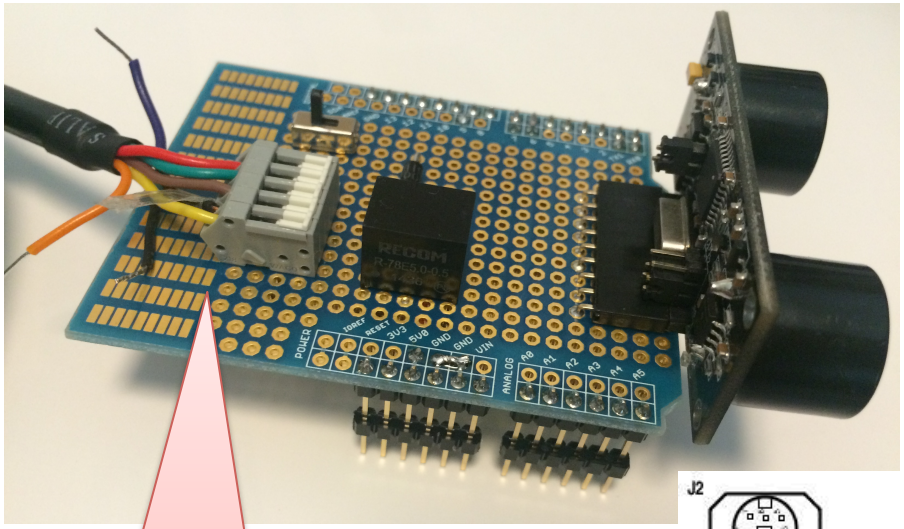
- ZigBeeプロトコルに準拠した無線通信モジュール
  - 技的取得済
- 主な仕様
  - 電源：3.3V
  - 室内通信距離：60m
  - 周波数帯域：2.4GHz

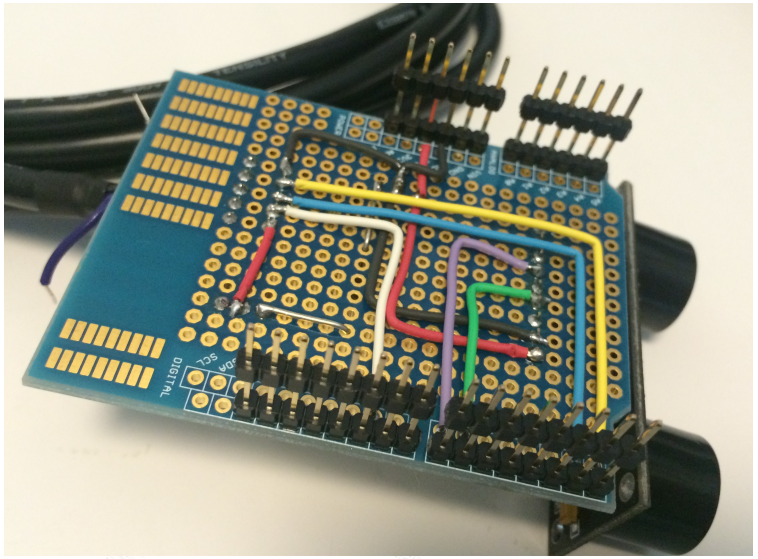


- 今回の設定・使用法
  - チーム毎に2枚のXBeeをペアリング（PAN IDによる）
  - PCとGR-SAKURAのシリアル通信が可能
    - GR-SAKURA側はSerial3に接続
    - 通信ボーレットは38,400

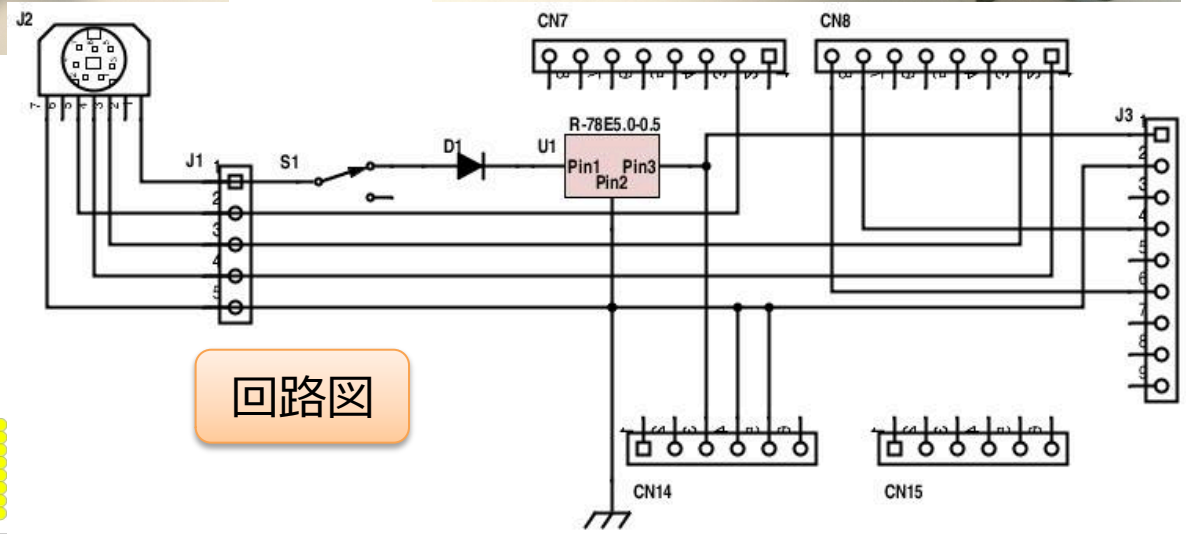
コネクタシールド

- GR-SAKURAとminiDinコネクタと超音波センサを接続するシールドを自作製作



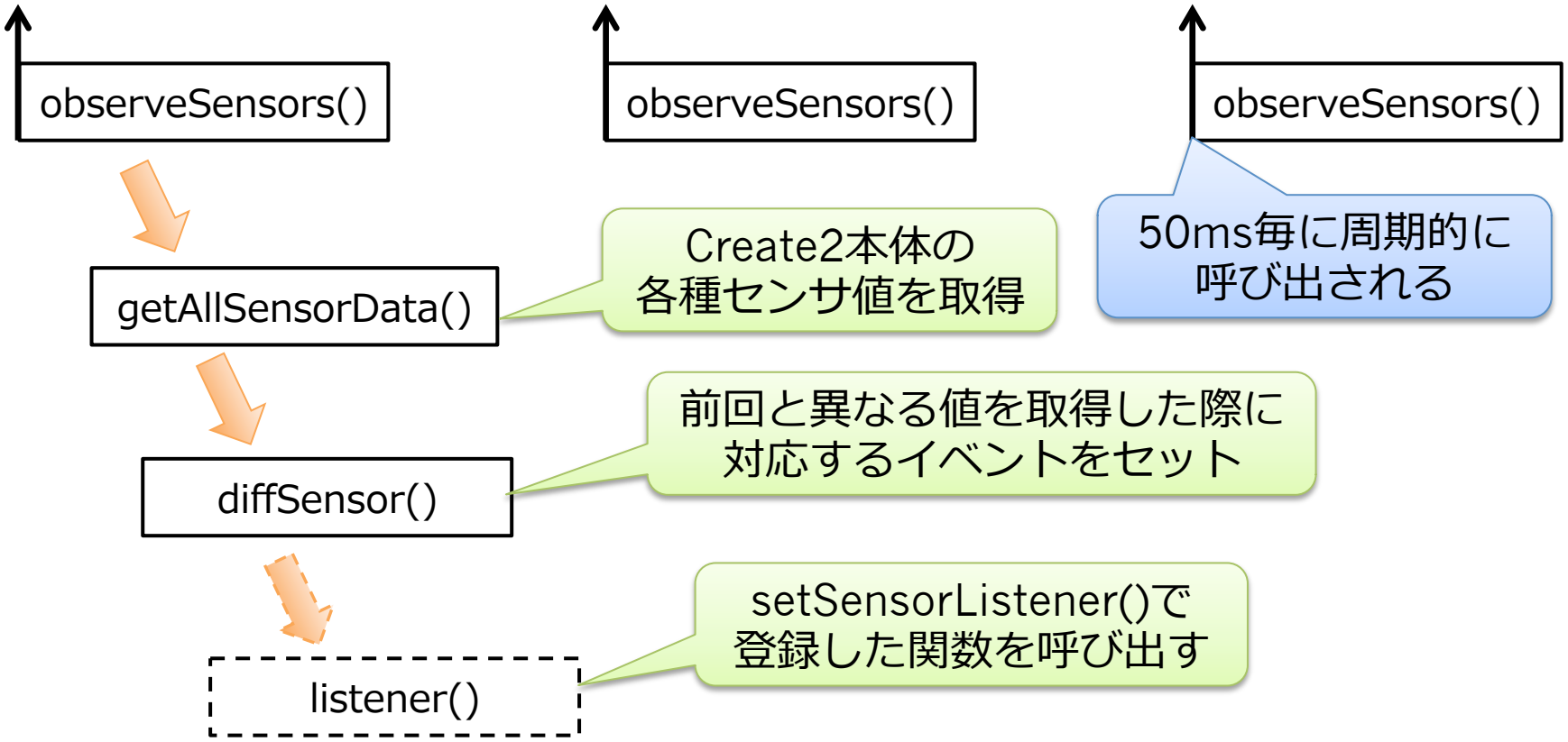


赤・緑・茶・黒・黄の順
黒は2本あるので注意



回路図

ソフトウェア仕様



diffSensor()/setSensorListener()まではMDDツールが雛形として生成
astah*上ではlistener()中身のアルゴリズムを設計する

イベント

資料集から
追記・変更

- diffSensor()内で前回取得した各種センサ値から異なるものに対応したイベントを設定する
- 主なイベント
  - ChangeBumperRight：右バンパの状態変化
  - ChangeBumperLeft：左バンパの状態変化
  - ChangeButtons：本体ボタンの状態変化（ボタン配置は次ページ）
  - ReachDistance：所定の距離への到達
    - setNextDistance()で距離を指定できる
  - ReachAngle：所定の角度への到達
    - setNextAngle()で角度（正の値が反時計回り）を指定できる
  - ReachSonicDistance：所定の超音波センサの取得値への到達
    - setNextSonicDistance()で取得値を指定できる
  - Timeout：所定の時間の経過
    - setNextTimeout()で経過時間を指定できる
    - 生成されるEvents.hではController_Timeoutにリネームされる

開発のヒント

- 適切な計画のもとにテストを実施する
  - 「何を」「どのくらい」試したいかを考える
    - とりあえずやってみて，それができたかどうかをチェックするだけは意味が無い
  - テスト項目が「どのくらい」達成できたか検証する
    - トライアンドエラーに陥らないようにする
- 適切なデバッグを実施する
  - XBee：Serial3　USB：Serial
  - LEDやスピーカーも有効
  - 本体ボタンを用いた遷移も有効

```
enum iRobotButtons {
        B_Clean = 1,
        B_Spot = 2,
        B_Dock = 4,
        B_Minute = 8,
        B_Hour = 16,
        B_Day = 32,
        B_Schedule = 64,
        B_Clock = 128
};
```

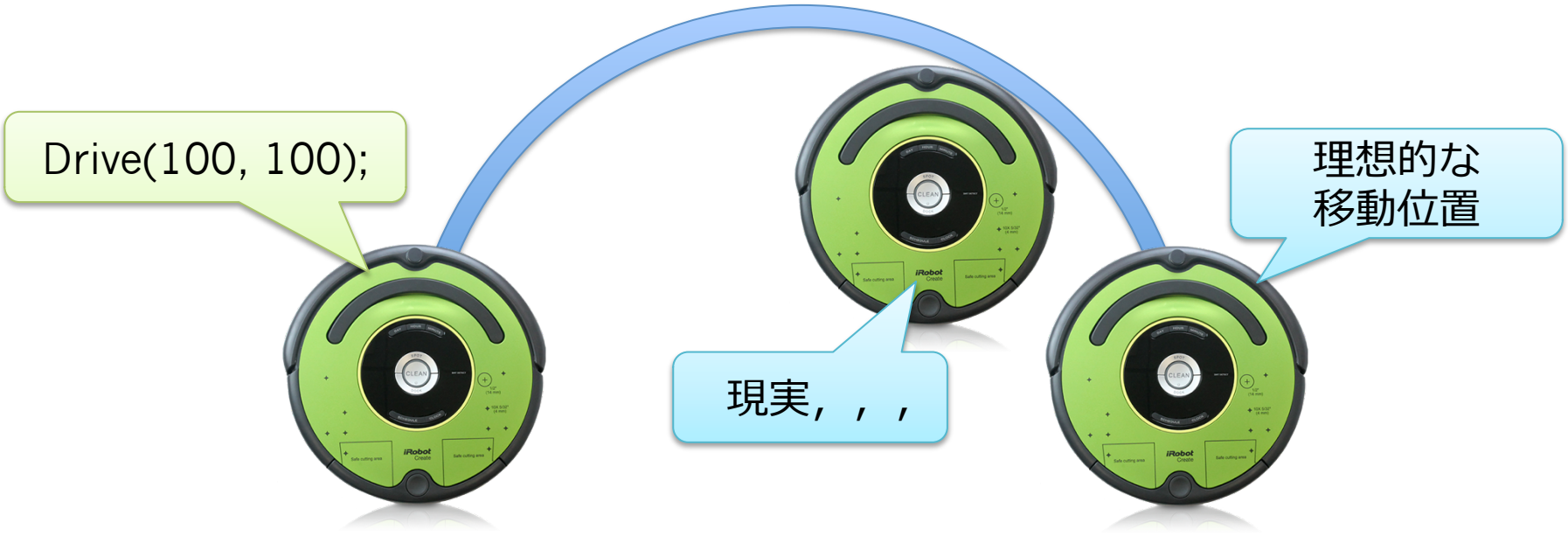
開発のヒント

資料集から
追加

- 複数のセンサ・イベントから目的に応じて
  適切なものを選択して戦略を決める
  - 左右のバンパ
  - 距離や角度の到達，エンコーダ値
  - 超音波センサ
  - タイマの経過時間

  ※ 幾つか組み合わせて戦略の精度を高めるのもアリ
- 理想と現実のぶれを考慮する
  - 次スライド

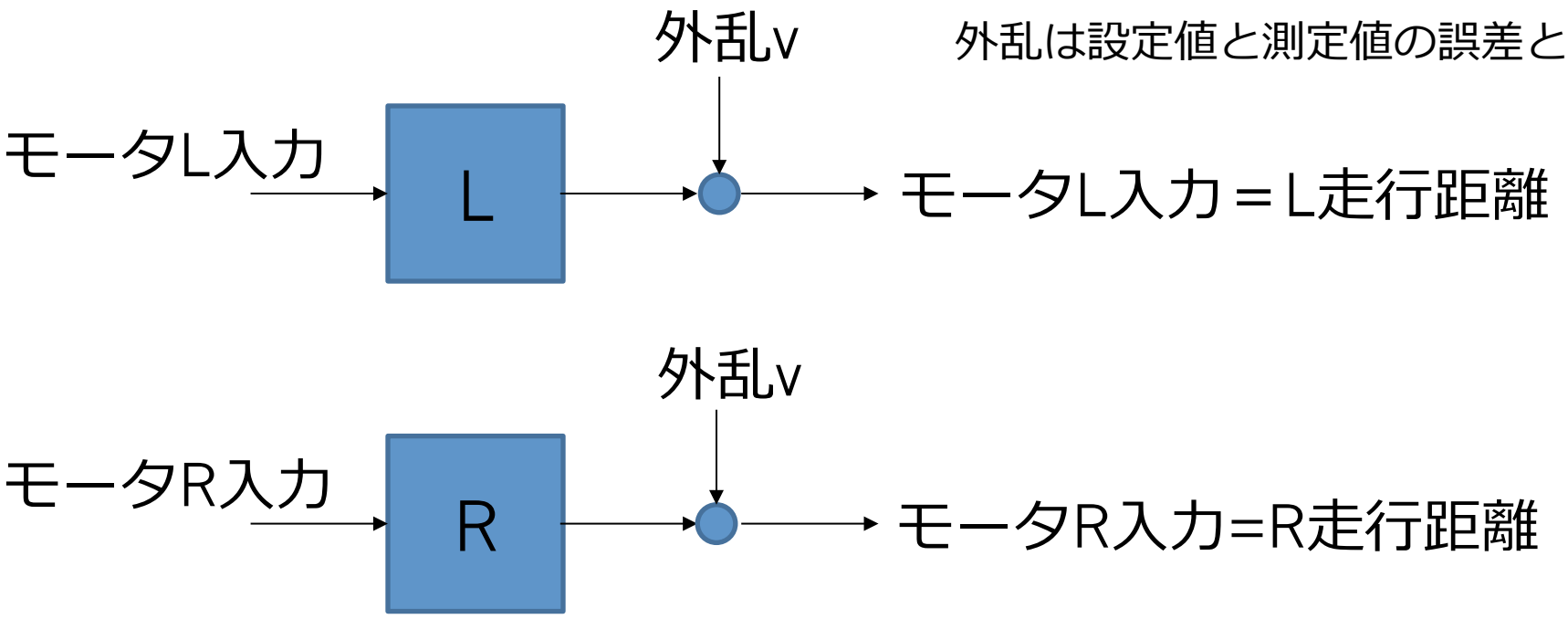
理想とのぶれ



ぶれの要因：
- モータ･エンコーダの個体差
- センサ特性
- 床面の材質
- タイヤの摩耗度 etc, etc, …

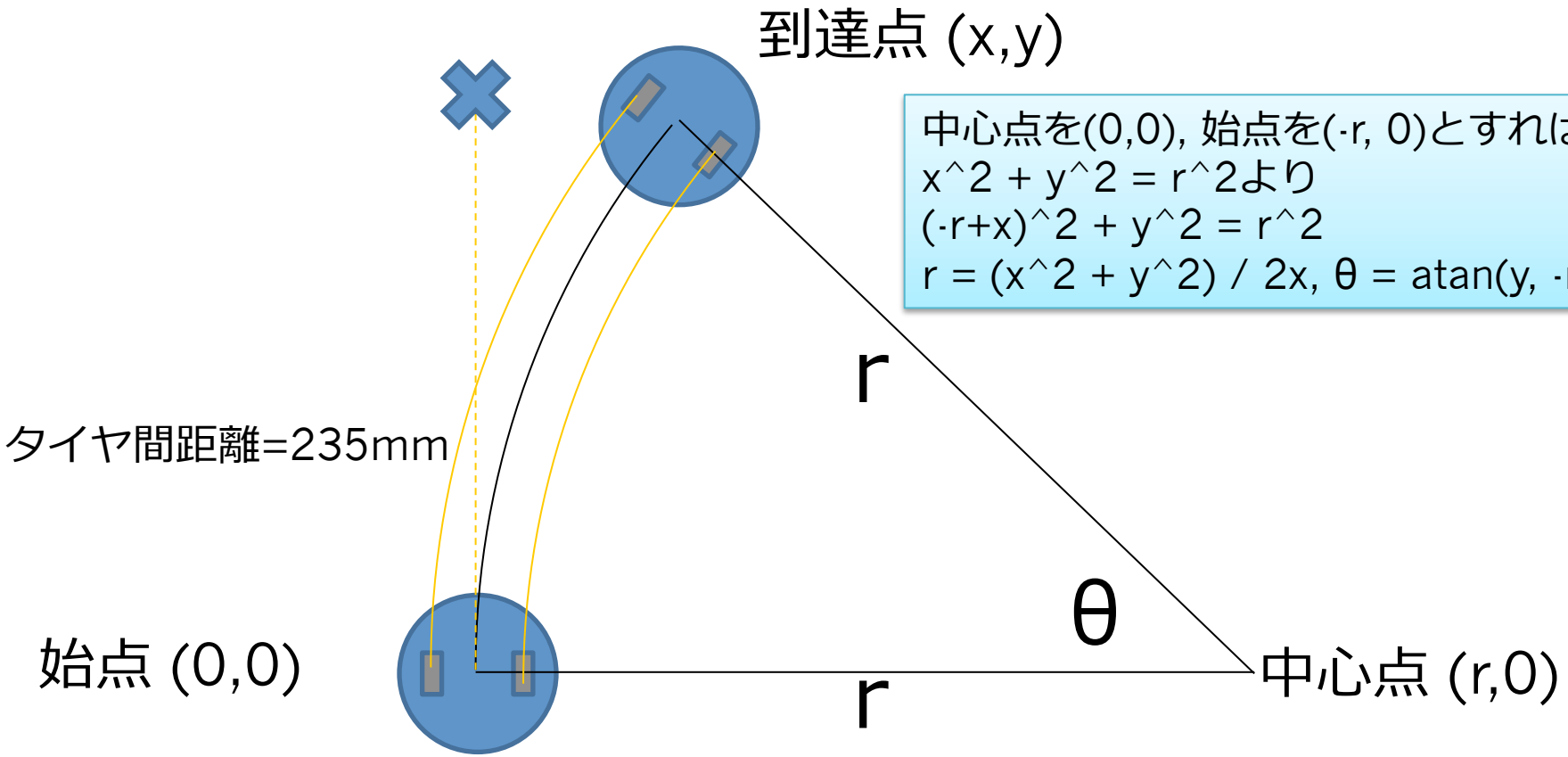
最小二乗法を用いて
パラメータを調整する

左右モータの出力



- L，Rそれぞれのずれの要因になる関数を知りたい
- 逆関数を求められれば，ちゃんと指定した通りの距離を走ってくれるようになるはず

計測

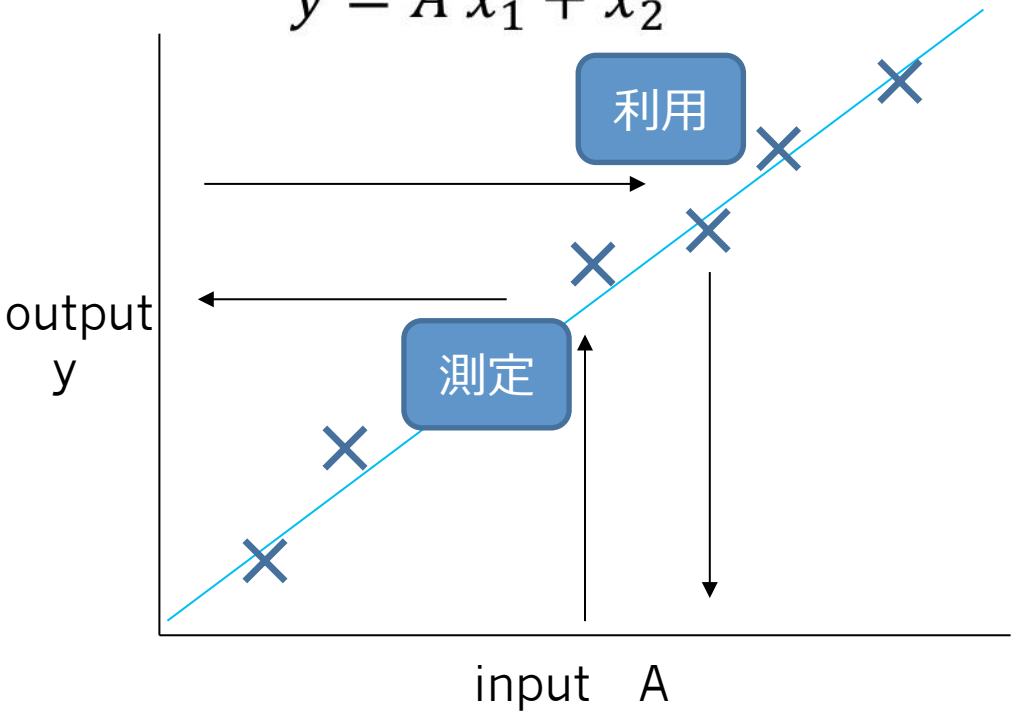


左モータ走行距離 = 2π * (r+L) / θ
右モータ走行距離 = 2π * (r-L) / θ

最小二乗法による逆関数の導出

- 1入力1出力だと？
  - 実際には考慮すべき入力値はまだある

$$y = A\,x_1 + x_2$$



1入力1出力　1次関数

$$x_1 = \frac{N\sum AiYi - \sum Ai \sum Yi}{N\sum Ai^2 - (\sum Ai)^2}$$

$$x_2 = \frac{-\sum Ai \sum AiYi + \sum Ai^2 \sum Yi}{N\sum Ai^2 - (\sum Ai)^2}$$

・逆関数　$A = \dfrac{y - x_2}{x_1}$

理解度の確認

この資料では以下を解説しました．理解度を確認してください．

- 開発教材のシステム構成を理解できましたか？
  - 各モジュールの仕様を理解できましたか？
  - Create2の動作モードやAPIを使いこなせますか？
- ソフトウェア仕様を理解できましたか？
  - MDDツールの生成コードの仕様を理解できましたか？
  - イベントの仕様が理解できましたか？
- 開発のヒントが理解できましたか？
  - 有効なテスト・デバッグ方法を実践できますか？
  - 外部環境とのシステム同定を実践できますか？

分からないところがあれば，質問してください．