AI/IoTセンサのしくみを知ろう（応用編）

徳島大学技術支援部常三島技術部門
技術専門職員　辻　明典　博士（工学）
E-mail: a-tsuji@is.tokushima-u.ac.jp

2018年12月1日（土）

講座内容

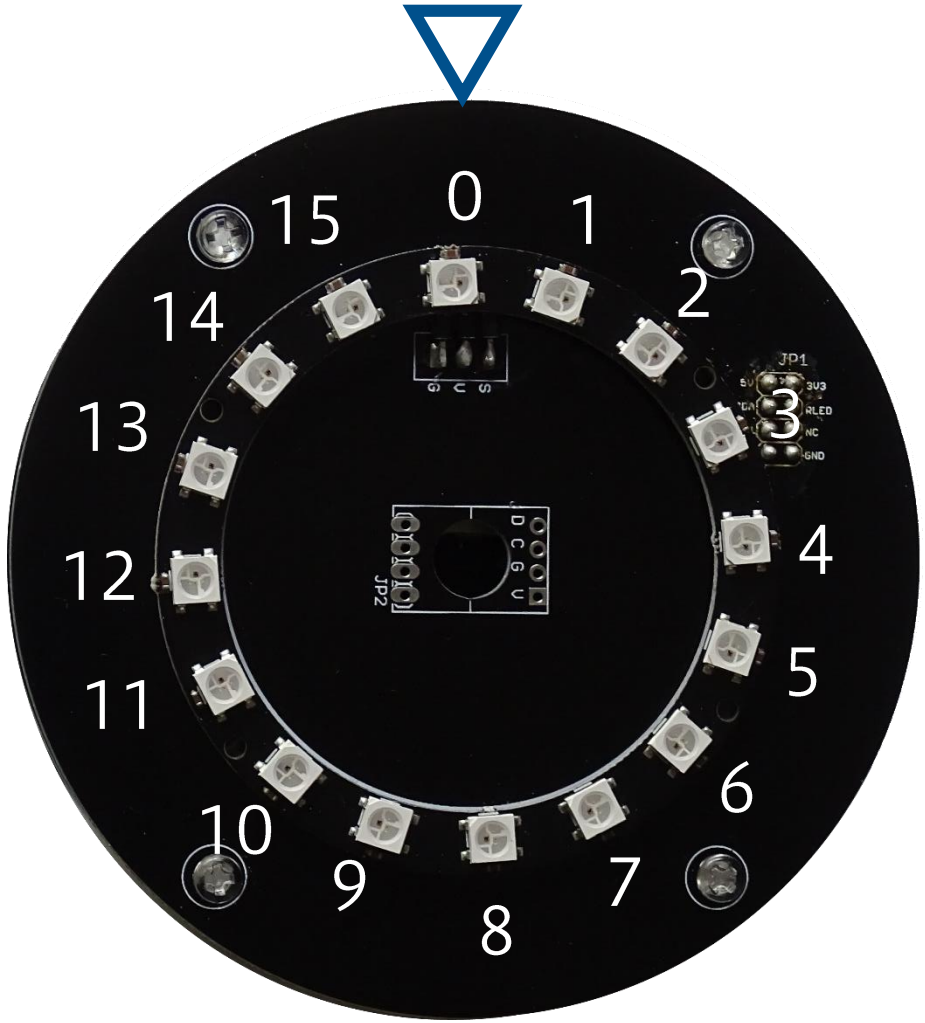
- 講師：辻　　明典（徳島大学技術支援部）
  桑折　範彦（徳島大学名誉教授）
  川上　博　　（徳島大学名誉教授）
- 土曜日：10:00～11:30
- 日程：

① 10 / 5　概要，環境設定，配布部品の確認
② 10 / 12 復習
③ 10 / 19 ロボットのモーター1（基本動作）
④ 10 / 26 ロボットのモーター2（応用動作）
⑤ 11 / 30 ロボットの制御1
（モーター，センサの協調動作）
⑥ 11 / 9　ロボットのセンサ1
（フォトリフレクタ）
⑦ 11 / 16　ロボットのセンサ2
（ライントレース1）
⑧ 12 / 7　ロボットの制御2
（ライントレース2）
⑨ 12 / 14　ロボットの制御3
（迷路課題）

Ex0701：フルカラーLED

- ライブラリ：FastLED
- 光らせるLEDの位置
  - leds[n]： n番目(n=0,1,2,…15)
- フルカラーLED 16個
- 光らせる色
  - CRGB(赤, 青, 緑)
  - CHSV(色相, 彩度, 明度)
  - それぞれ，0～255

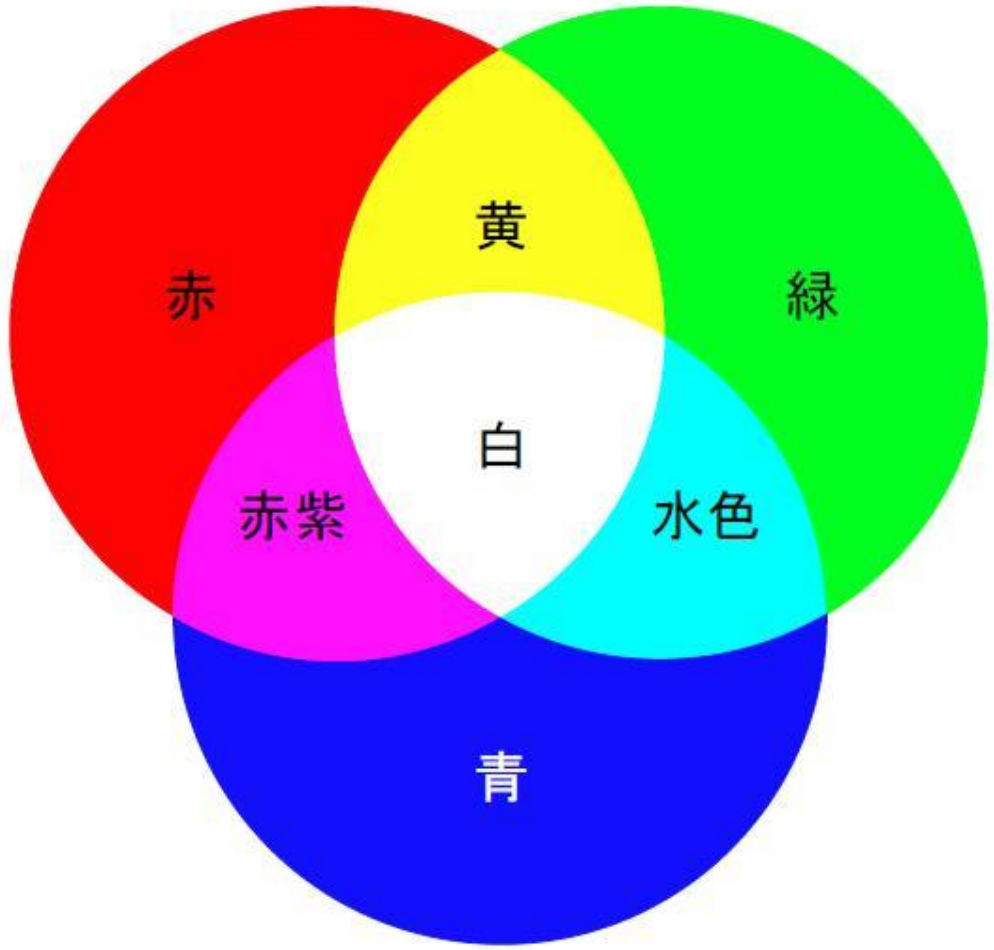
(例1) LEDの4番目を緑に点灯
```
leds[4] = CRGB(0, 255, 0);
FastLED.show();
```
(例2) LEDの3番目を色相(100)
```
leds[3] = CHSV(100, 255, 255);
FastLED.show();
```



光の三原色

光の三原色
・赤(Red)
・緑(Green)
・青(Blue)
の重ね合わせ（加法混色）

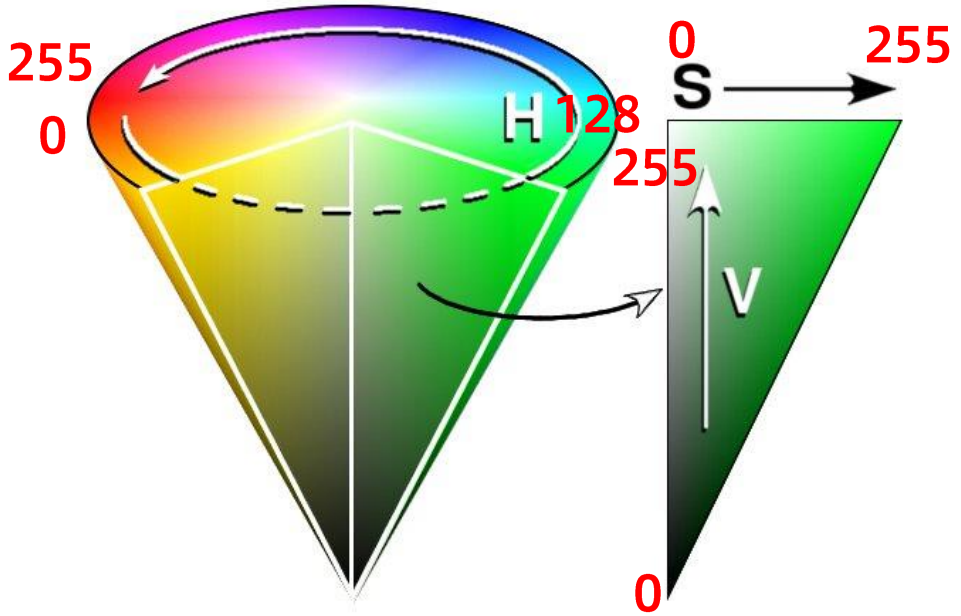
CRGB(R, G, B);
R値：0 – 255 赤の明るさ
G値：0 – 255 緑の明るさ
B値：0 – 255 青の明るさ



HSV表色系

色相（H: Hue）
　色の種類
彩度（S: Saturation）
　色の鮮やかさ
明度（V: Value, Brightness）
　色の明るさ

CHSV（h, s, v）;
h値：0 – 255
s値：0 – 255
v値：0 - 255



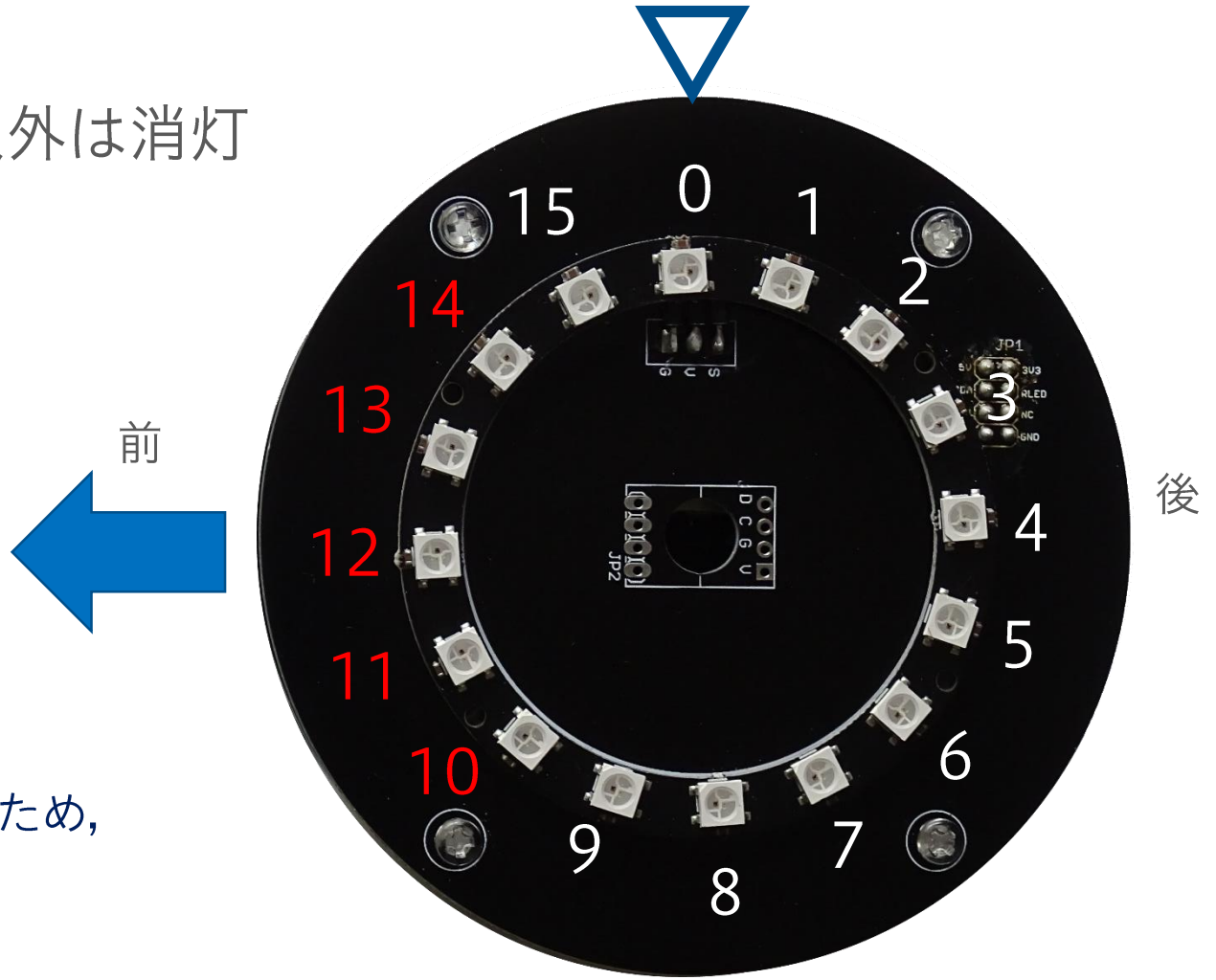
HSVの関係

Ex0702：フォトリフレクタの反応をLED表示

- LED10～14の5個を使用
- pr_bin[]が1のときに点灯，それ以外は消灯

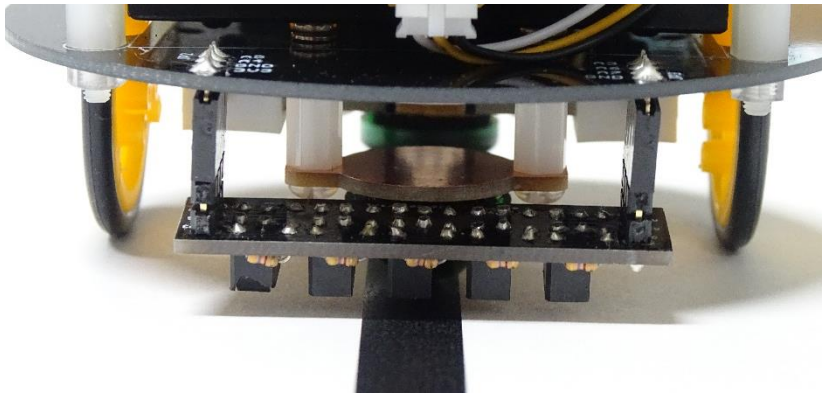
```
for (int i = 0; i < NUM_PR; i++) {
  if (pr_bin[i] == 1) {
    leds[i+10] = CRGB(255, 0, 0); // 点灯(赤)
  } else {
    leds[i+10] = CRGB(0, 0, 0); // 消灯
  }
  FastLED.show(); // 表示を更新
}
```

leds[i+10]は，LEDの10番目から14番目を使うため，
10～10+NUM_PR (10,11,12,13,14,15)



Ex0703：フォトリフレクタの自動校正

- ロボットをライン上に配置
- スイッチを押す
- ロボットが左右に動く
- すべてのセンサがラインを通過
- フォトリフレクタの自動校正が完了



左右に回転して，ラインをトレース
黒，白のコントラストを調べる

フォトリフレクタの自動校正時の動作

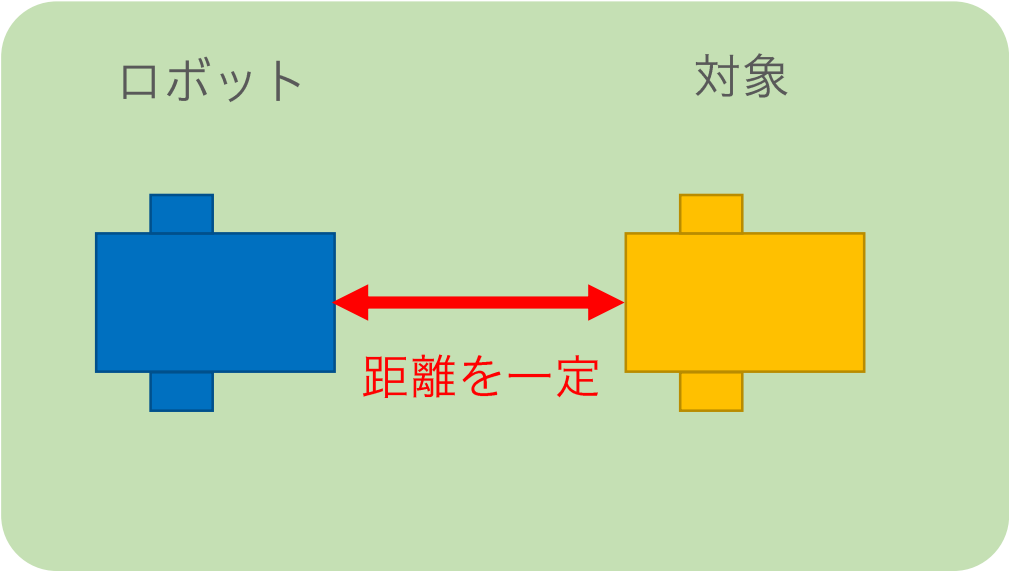
- ラインの上を左右に回転して，ラインをトレース
- 校正の動きを作る

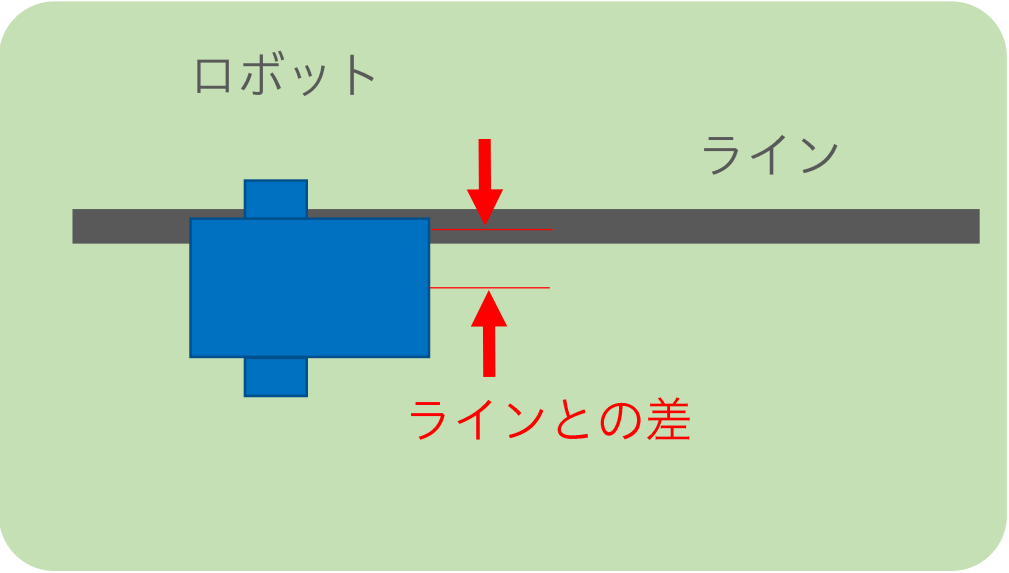
```
void pr_calib() {
  for (int i = 0; i < 80; i++) {
    if ( (i > 10 && i <= 30) || (i > 50 && i <= 70) ) {
      motor(sp, sp, HIGH, LOW); // 右回転
    } else {
      motor(sp, sp, LOW, HIGH); // 左回転
    }
    pr_minmax(); // 最小・最大を更新
    delay(40);
  }
  motor(0, 0, LOW, LOW); // 停止
}
```

iの値：ロボットの動き
0-10：左回転
10-30：右回転
30-50：左回転
50-70：右回転
70-80：左回転

ロボット制御

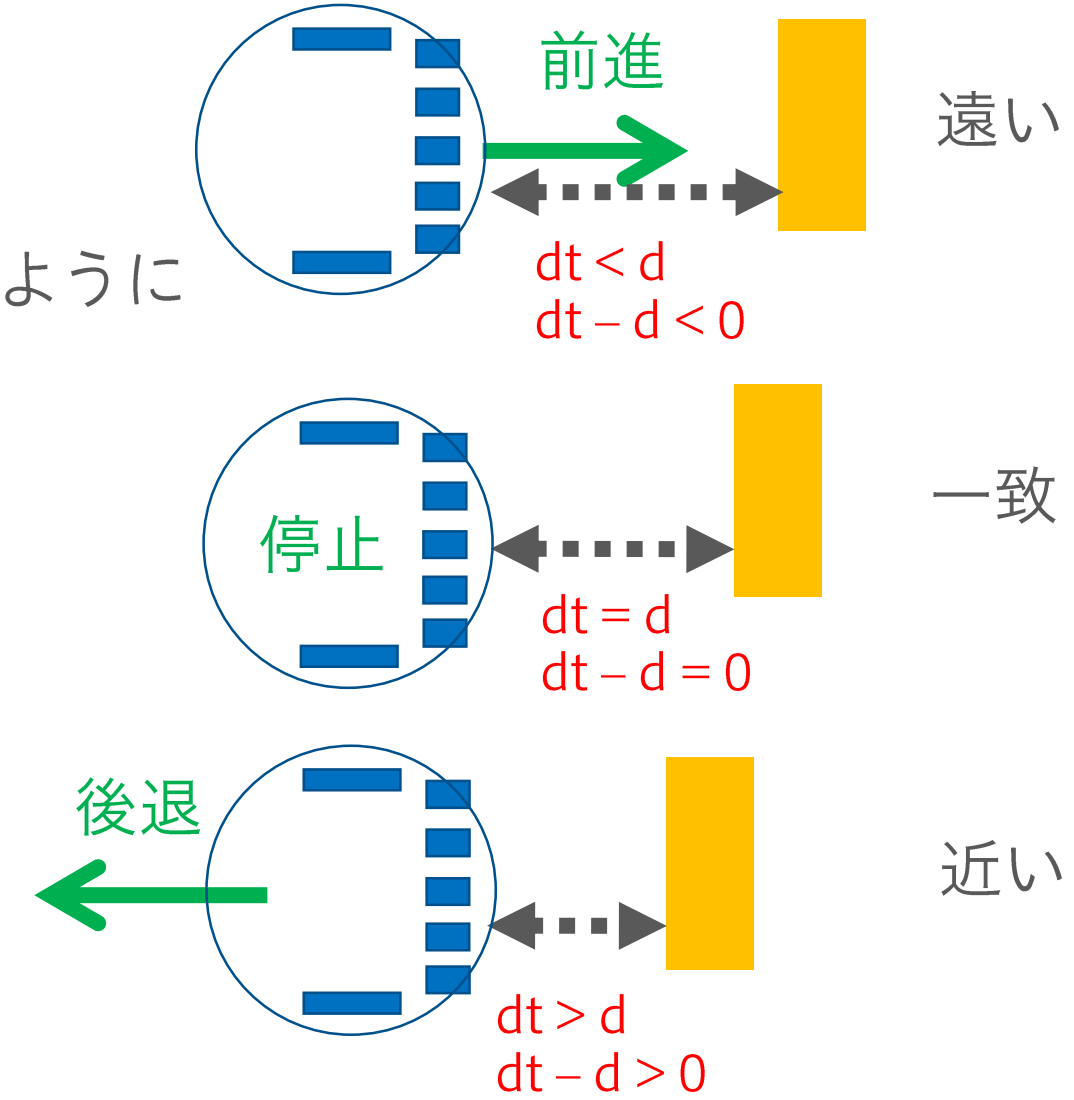
① 目標の設定（マイコン）
② 現在の状態を知る(センサ)
③ 目標と現在の状態との「差」を知る（マイコン）
④ 「差」があれば無くなるよう調整（モーター）





距離を一定に保つ

- ロボットと対象との距離を一定に保つ
  - 目標距離・・・・・dt [mm]
  - 計測距離・・・・・d [mm]
- 目標の距離と計測距離が0[mm]になるように
  モーターを制御
  - e = dt – d → 0 [mm]
- 目標距離：dt = 100 [mm]
- 計測距離：d = 0[mm]～200[mm]



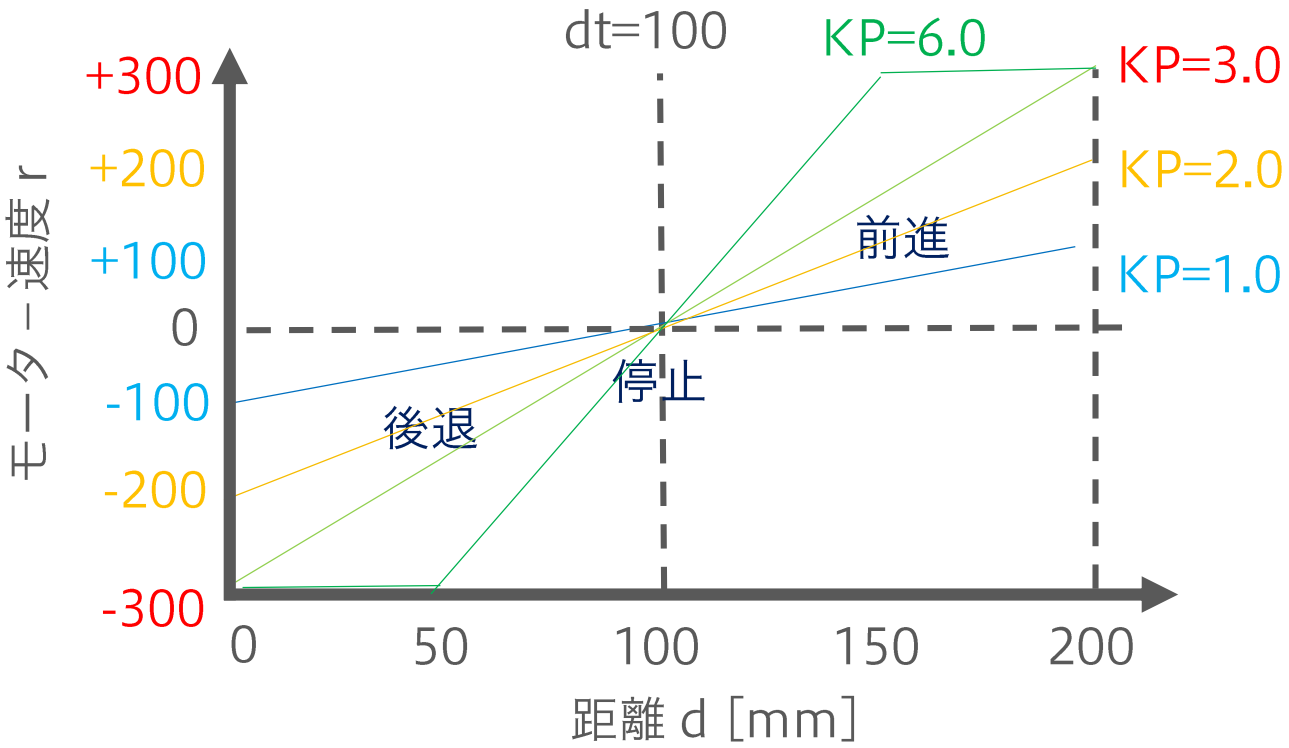
Ex0704：距離を一定に保つ

モーター速度の制御

$r = KP * (dt - d)$

KP：比例定数，dt：目標距離

d：現在距離



比例（Proportional）制御：
目標値と計測値との差に比例して操作量を制御

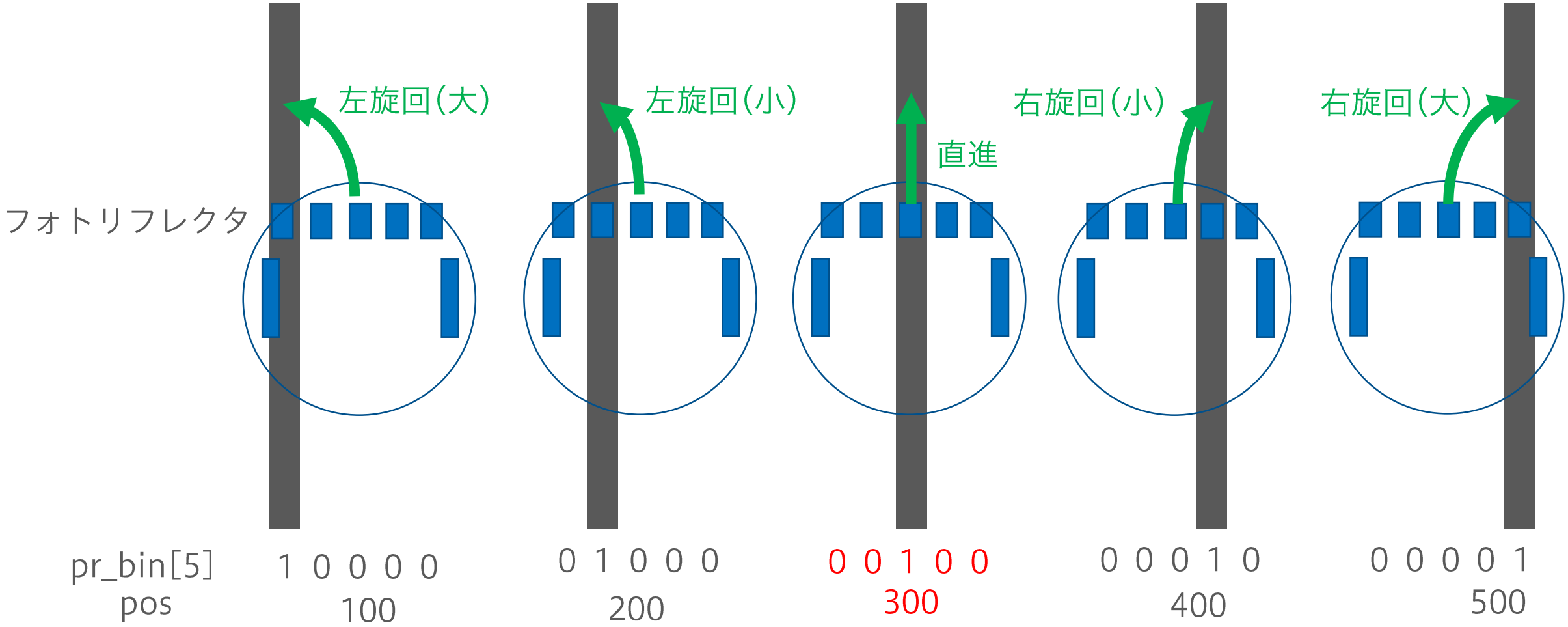
KPを変更してロボットの動作を確認

ロボットの前進と後退：
前進：r＜0：motor(-r, -r, FWD, FWD);
停止：r＝0：motor(0, 0, FWD, FWD);
後退：r＞0：motor(r, r, BWD, BWD);

※ Ex0406のif文を使った場合と比べてみよう

ライントレース

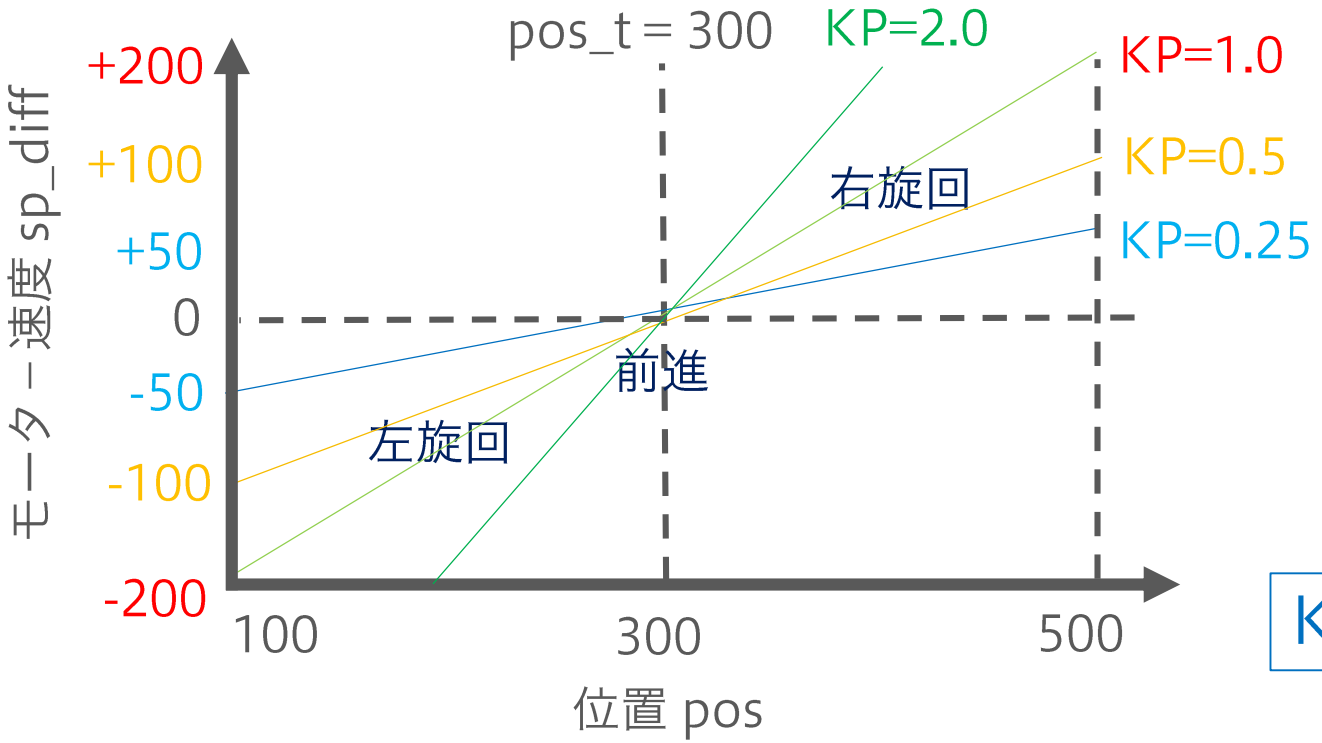
▶ロボットがラインの中央を走るように制御



Ex0705：ライントレース

- 目標位置：pos_t = 300
- 計測位置：pos（100～500）
- モーターの速度：sp_l, sp_r（0～1023）
- 目標位置pos_tと計測位置posの差が小さくなるようモーター速度を変化

比例（Proportional）制御：
目標値と計測値との差に比例して操作量を制御



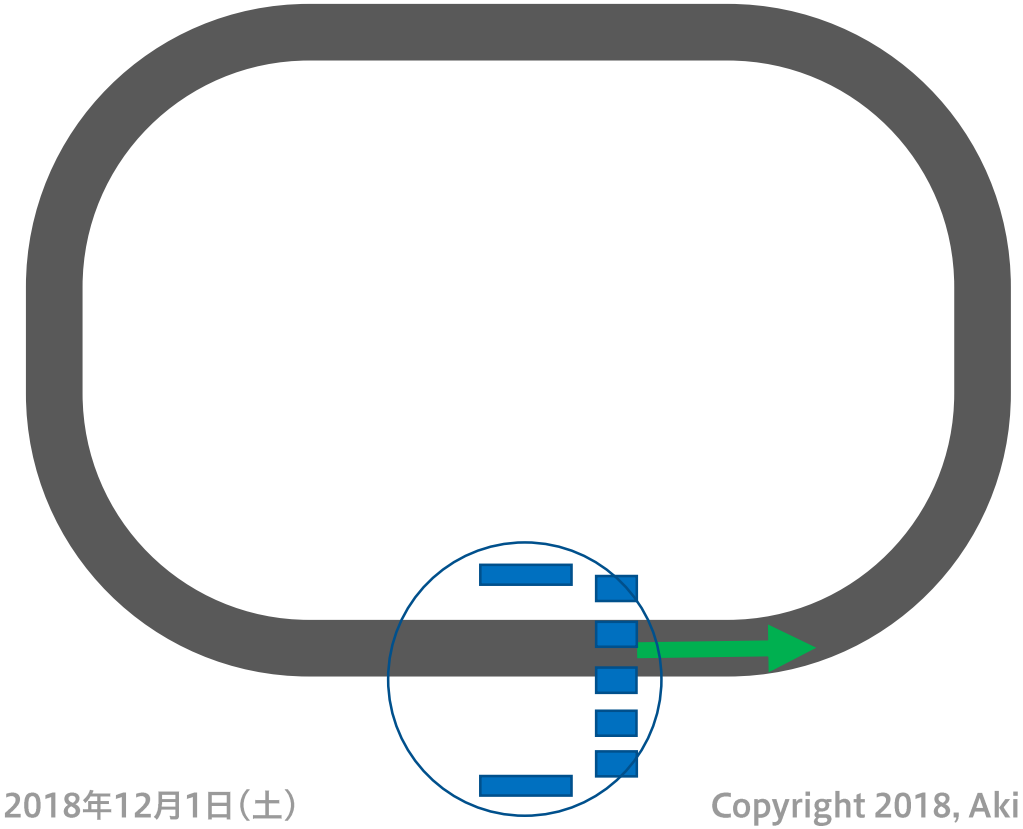
目標との差：sp_diff = KP * (pos_t – pos)
KP：比例定数
pos_t：目標位置
pos：計測位置

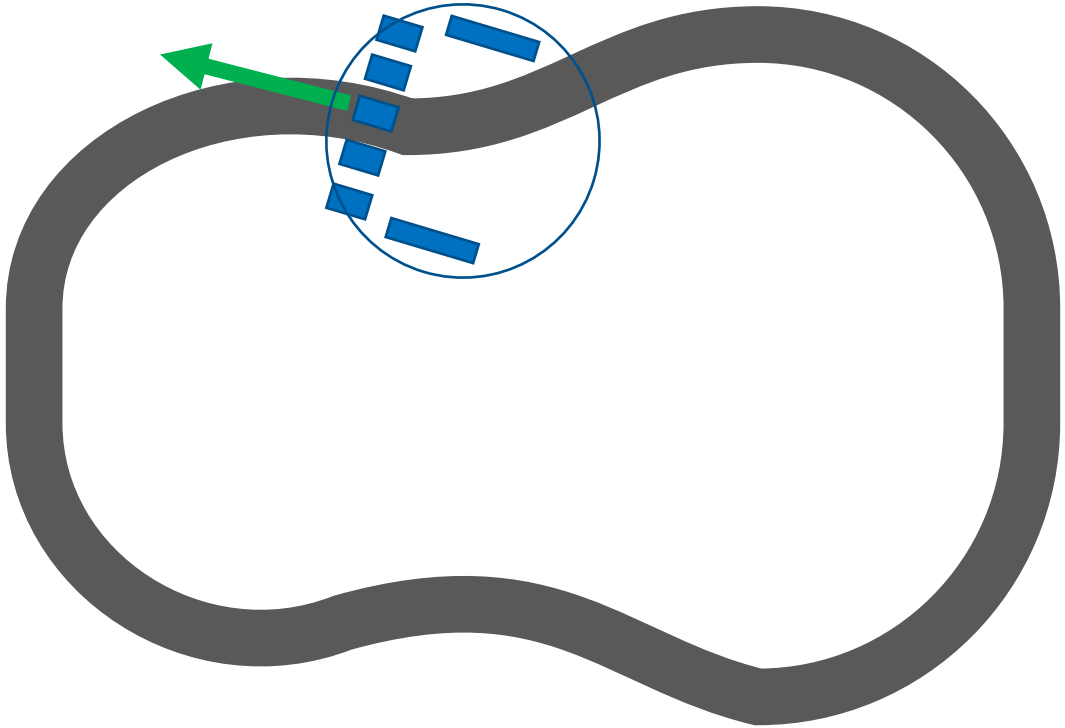
左モーター：sp_l = sp − sp_diff
右モーター：sp_r = sp + sp_diff
sp: モーターの基本速度

KPを変更してロボットの動作を確認

コースを周回

- モーターの基本速度(sp)，比例定数 (KP) を変更
- コースを左回り，右回りに脱線せずスムーズに走行できるか確認





付録：Ex0706: 自動校正（EEPROM）

- 必要な時だけ，自動校正を行う
- 自動校正の開始
  - （1）モーターの電源をON
  - （2）スイッチボタンをONのままマイコンの電源をON

```
// 起動時に校正の有無を確認
 if (digitalRead(SW_PIN) == LOW) {
   sw_calib = 1; // 校正をON
  for (int i = 0; i < 3; i++) { // 3回白色点滅
    leds[0] = CRGB(255, 255, 255);
    FastLED.show();
    delay(200);
    leds[0] = CRGB(0, 0, 0);
    FastLED.show();
    delay(200);
  }
 }
```

```
// 自動校正開始
if (sw_calib == 1) {
 sw_calib = 0;
 pr_calib(); // フォトリフレクタの自動校正
 eeprom_write(); // 校正値をEEPROMに書き込み
} else {
 eeprom_read(); // 校正値をEEPROMより読み込み
}
```

付録：Ex0707：ライントレースと回転動作

- ラインを検出している間はライントレース
- ラインをはみ出すと180度左回転して方向転換