『ロボットをつくろう WS 第 3 回』資料
1.1 版

2014/9/26(金) 10:00～12:00　徳島大学 工学部知能情報工学科 C 棟 1F 実験室

辻　　明典

目次

1. ロボットの構成

1.1. 基本構成

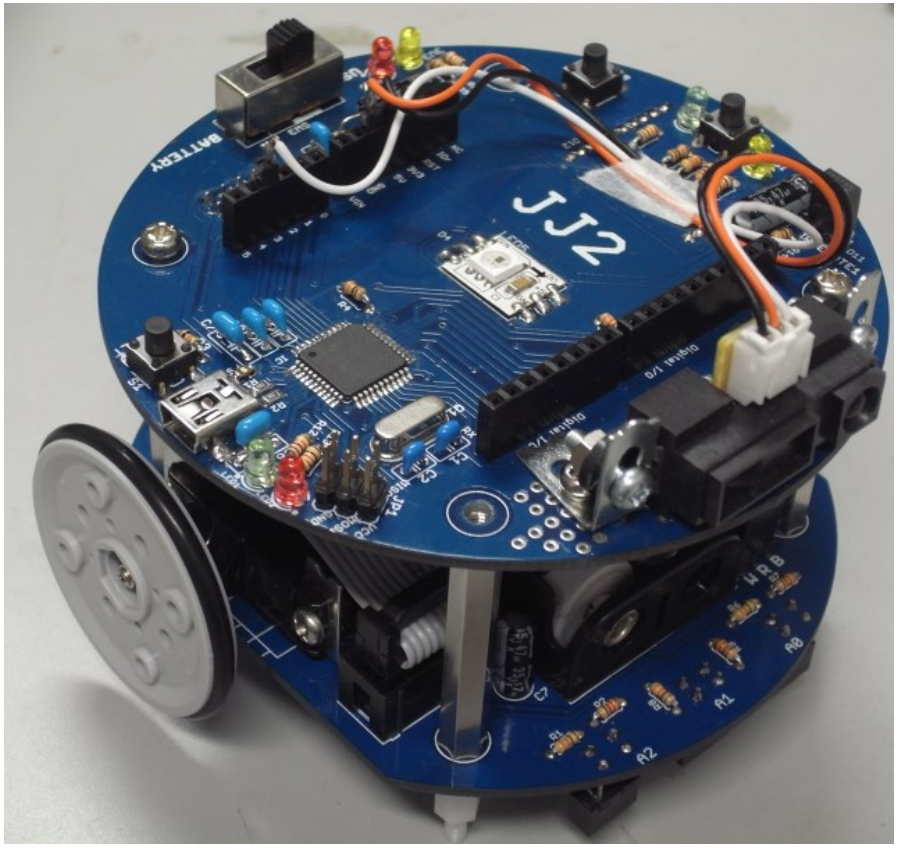
ロボットは，Arduino 互換マイコンボード，モータボードより構成される．ロボットの制御には，マイコン（AVR 社 ATmega32u4）マイコンを用いる．

(1) Arduino 互換ボード

① ATmega32u4 マイコン (AVR 社)
② フルカラーLED（1 線シリアル接続）
③ ブザー（圧電素子）（裏面）
④ 赤外線受信センサ（リモコン）
⑤ Arduino 互換拡張ソケット
⑥ 無線モジュール(ZigBee, Bluetooth)（オプション）

(2) モータドライバボード

① サーボモータ 2 個
② フォトインタラプタ 3 個



2. ロボットのモデリング

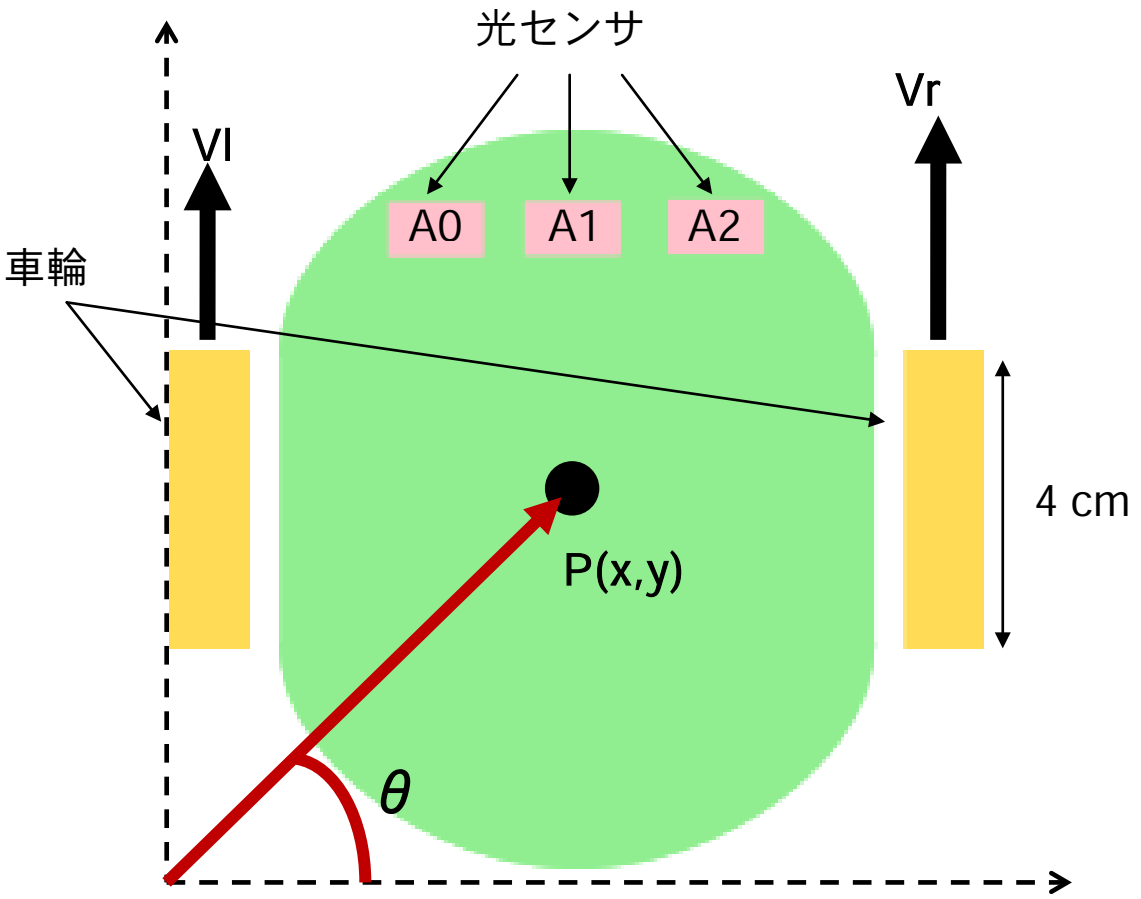
2.1. ロボットの構成要素

入力（センサ）

- **光センサ　　3 個**
- 無線モジュール　1 個
- 赤外線センサ　　1 個

出力（アクチュエータ）

- **サーボモータ　　2 個**
- ブザー　　1 個
- フルカラーLED　1 個



モデリング例：

車輪の直径：R

ロボットの幅：L

左右のモータ速度：Vl,　Vr (cm/s)

ロボットの角度：θ

X 方向の速度：Vx (cm/s)

Y 方向の速度：Vy (cm/s)

現在のロボット位置：P(x, y)

現在の目標の位置：P(xt, yt)

次の目標の位置: P(xd, yd)

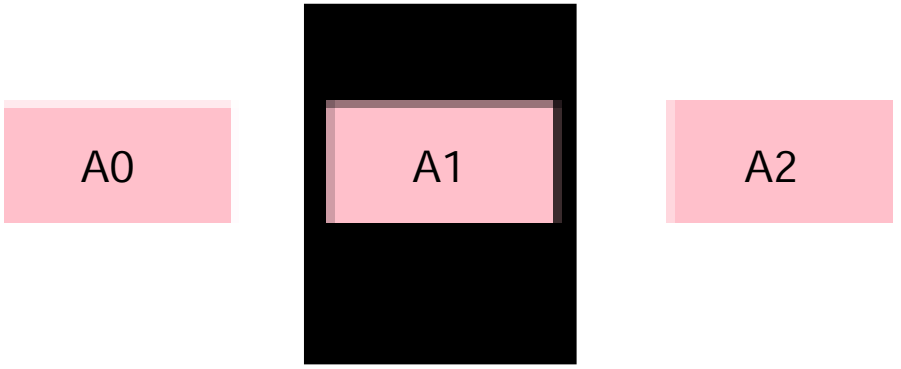
3. ロボットの制御

3.1. 光センサの入力

ロボットのセンサにより周囲の環境を判別して行動する．光センサにより，ロボットの足元の状態を調べることができる．

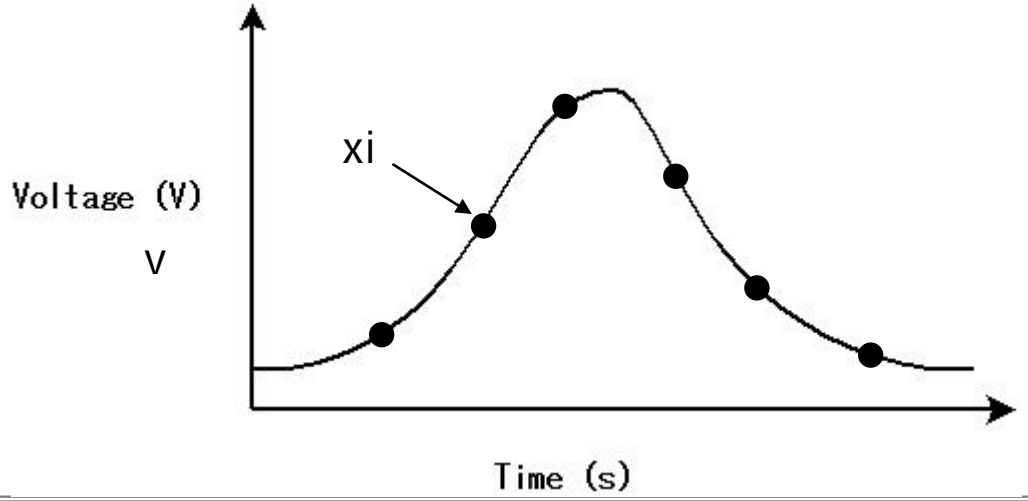
方法：
① 光センサより得られる値 xi をシリアルモニタで調べる．
　ーライン上，ライン外

※光センサは，マイコンのアナログ入力（10 ビット AD 変換器）に接続されており，電圧 V (0 ～ 5.0V) に対応する値 xi , i=0,1,2,…に対応する 0～1023 の値が得られる．
0V のとき 0, 5.0V のとき 1023.



ライン

② TeraTerm で光センサのデータを取得する．センサの特性をプロットする．

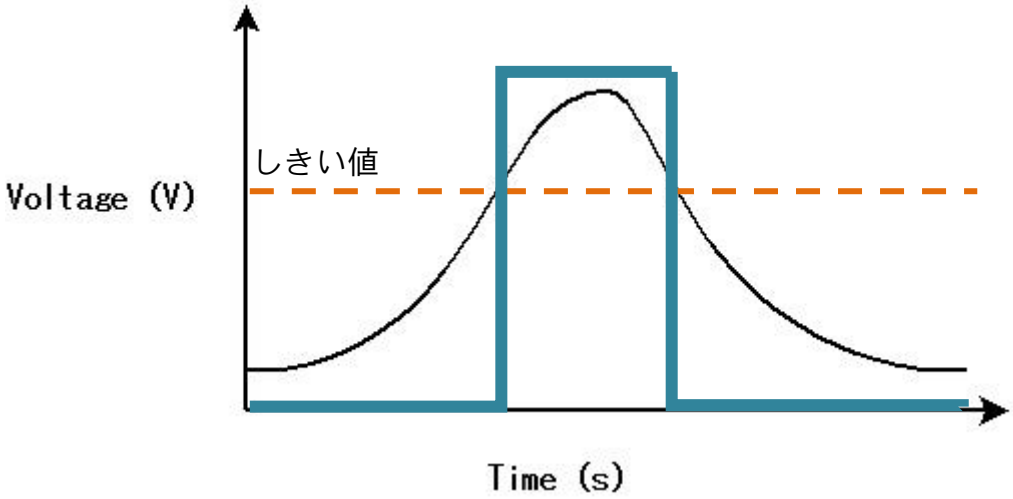


3.2. ラインの検出

方法：
①しきい値(vth)を決めて，センサより得られた値 x0_i (xn_i, i= 0, 1, 2, …) が，しきい値より大きな値ならライン上(1)，それ以下ならライン外(0)とする．

```
x0 = analogRead(A0);
if (x0> vth) st0 = 1;
else         st0 = 0;
```

ロボットの状態（ライン上にあるかないか）として，2 状態 st0={0,1}を定義する．
センサの数に合わせ，状態(sti, i=0,1,2,…)を定義できる．
② ロボットの状態 st にあわせ，ロボットを動作 act させる．



課題 1. センサ(1 個)がライン上にあるかないかを判別する関数 check_line を作成する．

int check_line(int x);

引数：センサの値(x)
返り値： 状態(st)

考察：ロボットがライン境界を横切るときに(1), (0)が繰り返され，ロボットの動きが不安定になることがある.

→ センサより得られた値 xi, i=0,1,2…の N 個分を平均する（ローパスフィルタ）.

y = (x0 + x1 + x2, …, xN-1) / N

→ 平均化したデータと通常のデータをプロットして比較してみる.

（例）センサより得られた値 5 個分の平均をとり結果 y を出力する.

入力信号列 xi ：1, 2, 3, 10, 5, 6, 7, 8, 9, 10, …

サンプル数 N =5

出力：y

入力：1, 2, 3, 10, 5

平均：(1 + 2 + 3 + 10 + 5) / 5;

出力：4 (4.2)

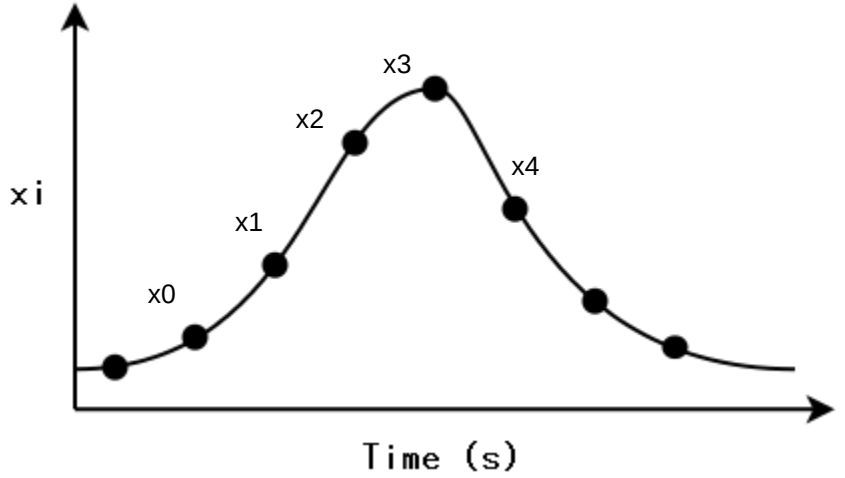
入力：2, 3, 10, 5, 6

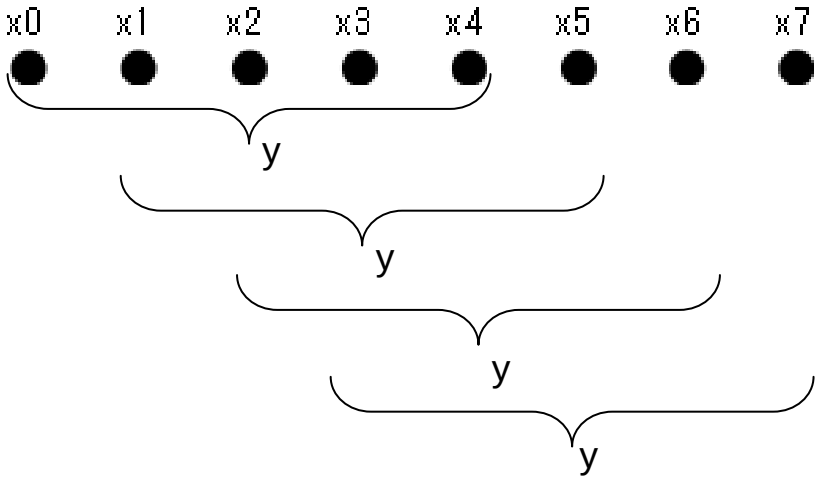
平均：(2 + 3 + 10 + 5 + 6) / 5;

出力：5 (5.2)

コード例：関数名 int lowpass(int x)

```
const int N=5; int buf[N]; int y;
buf[0] = x; // buf[0] に最新のデータを格納
y = 0;
for (i=0; i<N; i++) {
   y+= buf[i];
}
y /= N;
for (i=N-1; i>0; i--) { // データを 1 個ずつ移動
     buf[i] = buf[i-1];
}
```





3.3. ロボットの行動

センサより得られた情報に基づいてロボットのモータを動かす.

課題 2. センサ（1 個）の状態に応じて，ロボットを動かす．任意のセンサ（1 個）より得られた値を x0 としてラインを検出する．ロボットの状態 st0 を 0, 1 （2 状態）として，次の表を完成させ，ロボットを動作させる.

センサの状態 st0	ロボットの動作 action
0	ライン探索 line_search
1	ライン発見 line_found

・ロボットの動作として，ラインの内側を走らせる．ラインに沿って走らせる.
・ライン探索例：その場で回転してラインが見つかったら終了
while (1) { x = analogRead(A0); st0 = check_line(x); if (st0 == 1) break; else { servo(45,135); delay(30); }
ただし，ラインから大幅にはずれると・・・
・プログラムの流れ
センサの値取得→（フィルタ処理）→現在のロボットの状態判別→ロボットの行動

課題 3. 任意のセンサ（2 個）の値 x0, x1 からロボットの状態 st0,st1 を決定し，各状態に合わせた行動 act に対応する表を作成してロボットを動作させる.

st0	st1	Action
0	0	
0	1	
1	0	
1	1	

考察：
・ラインの濃さ，太さ，紙質が変わった場合でも安定して動作させるようにはどうすれば良いか.

課題 4. 任意のセンサ（3 個）の値 x0, x1,x2 からロボットの状態 st0,st1,st2 を決定し，現在の状態に合わせた行動 act に対応する表を作成してロボットを動作させる.（ヒント：コースに合わせ，成立しない条件を除外する.）

st0	st1	st2	Action
0	0	0	
0	0	1	
0	1	0	
0	1	1	
1	0	0	
1	0	1	
1	1	0	
1	1	1	

A0　　A1　　A2

課題 5. ラインセンサの情報に基づいて PID 制御によるモータの制御を行う．ラインセンサ（3 個,L, C, R）の値 x0, x1,x2 をライン反応を 1，反応なしを 0 として，{x0,x1,x2} = {0/1,0/1,0/1}．各センサに対応する重みを wi として，センサの重み付けした出力を $xi*wi^T$ を求めて位置情報(position)=誤差(error)とする．ここで，{w0,w1,w2 } ={-N,0,N}．中央ラインの走行を制御目標(Center=0)として，Position が Center に近づくよう制御器（controller）にて調整する．

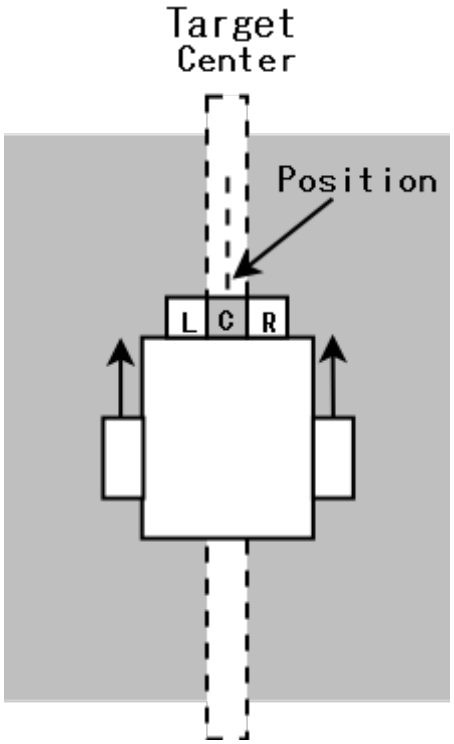
比例制御：Kp * error・・・d_prop

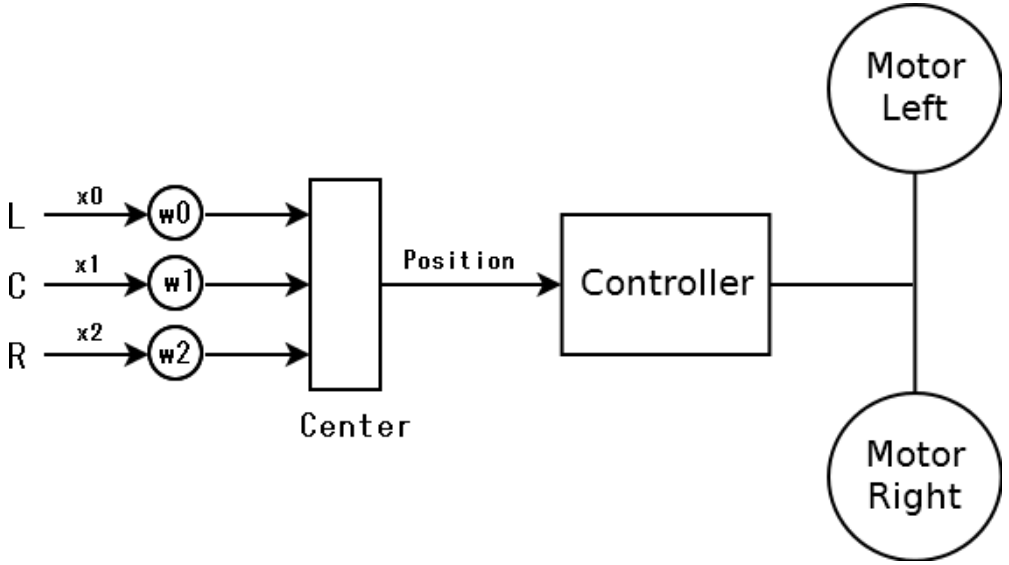
積分制御：Ki * (d_p rop– d_prop_past)

微分制御：Kd * (error – error_past)

制御出力：out = Kp*error + Ki (d_prop – d_prop_past) + Kd * (error – error_past)

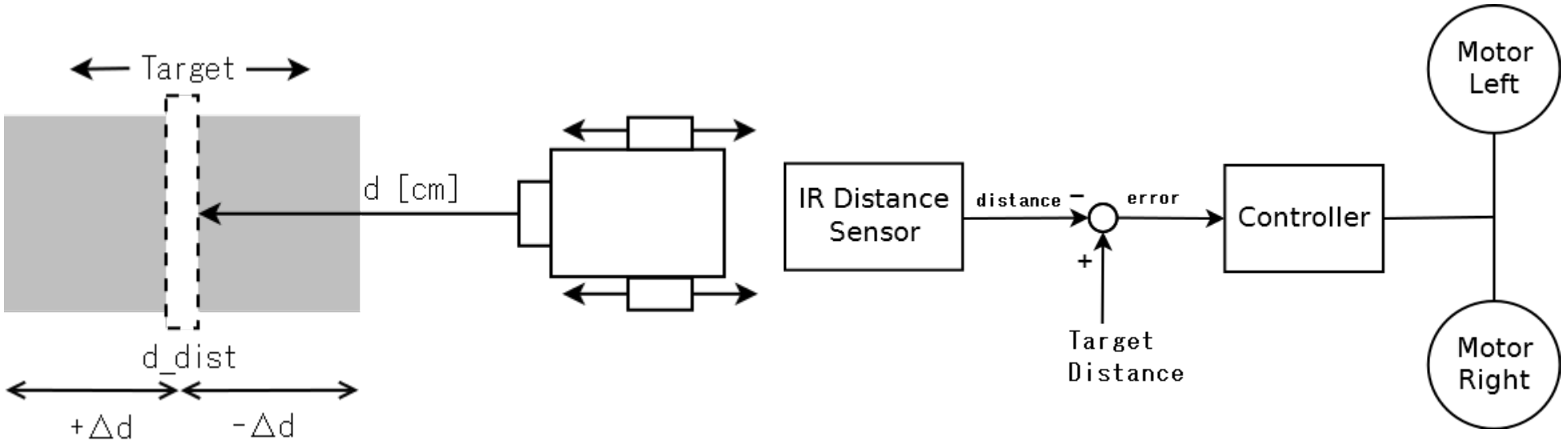
・ロボットに制御を導入して走行コースを走らせ，Kp, Ki, Kd パラメタの変化により走行がどのように変化するかを実験せよ．
ロボットのラインの直線，コーナーの挙動を見る．





課題 6. 距離センサの情報に基づいて PID 制御によるモータ制御を行う．距離センサより得られるデータから距離(d)を算出する（データシート参照）．目標距離(d_dst)を設定して，ロボットと対象（障害物）との距離が目標距離となるように制御する．ロボットの追従範囲 d_dst±Δd 内に入ったときに制御を動かす．対象と目標距離との差：error = d_dst-d が小さくなるようにモータ制御を行う．このとき，モータは前後に移動させて目標との距離を調整する．

・PID 制御のパラメタ Kp,Ki,Kd をかえて，ロボットの目標に対する追従動作を確認せよ．



（例）d_dist = 15cm，±Δd=5cm に設定する．つまり，10cm～20cm の範囲に入ったときにターゲット追従制御を行う．
距離センサの値 d=10cm のとき，誤差 e は，e=d_dst – d = 15cm – 10cm = 5cm
距離センサの値 d=20cm のとき，誤差 e は，e=d_dst – d = 20cm – 15cm = -5cm
誤差 e>=0 ときターゲットに近いので後退
誤差 e<0 のときターゲットから遠いので前進

4. マイコンボードのピン配置

4.1. ピン配置

マイコンボード上のピンソケットは，Arduino Leonardo 互換のピン配置である．

(1) 電源ピン

ラベル名	機能	接続先
NC	未接続	未使用
IOR	IO 電源	5.0V
RST	リセット	リセット
3.3V	電源(3.3V)	3.3V
5.0V	電源(5.0V)	5.0V
GND	グランド	GND
GND	グランド	GND
VIN	電圧入力	未使用

(2) アナログピン

ラベル名	機能	接続先	ライブラリ
A0	アナログ入力 0	フォトインタラプタ 0	AnalogRead
A1	アナログ入力 1	フォトインタラプタ 1	AnalogRead
A2	アナログ入力 2	フォトインタラプタ 2	AnalogRead
A3	アナログ入力 3	距離センサ	AnalogRead
A4	アナログ入力 4		
A5	アナログ入力 5		

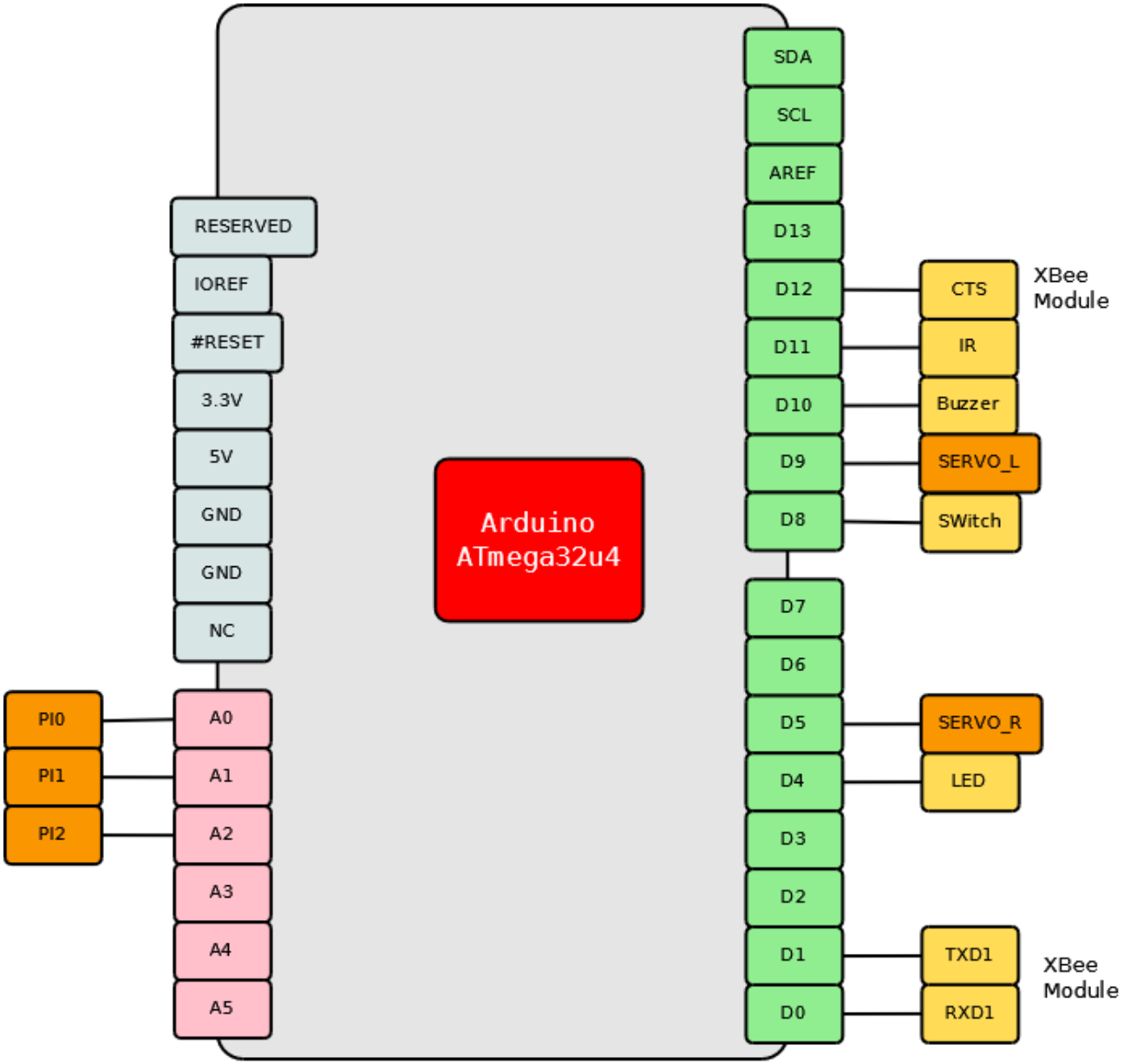
(3) デジタルピン

ラベル名	機能	接続先	ライブラリ
SCL	I^2C 通信のクロック	D3(共用)	Wire
SDA	I^2C 通信のデータ	D2(共用)	Wire
AREF	アナログリファレンス		
GND	グランド	GND	
D13	デジタル入出力 13	赤外線 LED	IRremote(Send)
D12	デジタル入出力 12	XBee(CTS)	XBee
D11	デジタル入出力 11	赤外線受光素子	IRremote
D10	デジタル入出力 10	圧電ブザー	Tone
D9	デジタル入出力 9	サーボモータ左	Servo
D8	デジタル入出力 8	スイッチ	digitalRead

(4) デジタルピン

ラベル名	機能	接続先	ライブラリ
D7	デジタル入出力 7		
D6	デジタル入出力 6		
D5	デジタル入出力 5	サーボモータ右	Servo
D4	デジタル入出力 4	フルカラーLED	FastSPI_LED2
D3	デジタル入出力 3	SCL	Wire(I2C)
D2	デジタル入出力 2	SDA	Wire(I2C)
D1/TXD1	デジタル入出力 1	XBee/BT(RX)	Serial1, XBee
D0/RXD1	デジタル入出力 0	XBee/BT(TX)	Serial1, XBee

4.2. ピン配置図



5. Arduino のスケッチ

5.1. Arduino の準備

1) USB ケーブルを接続して電源スイッチを USB にする.

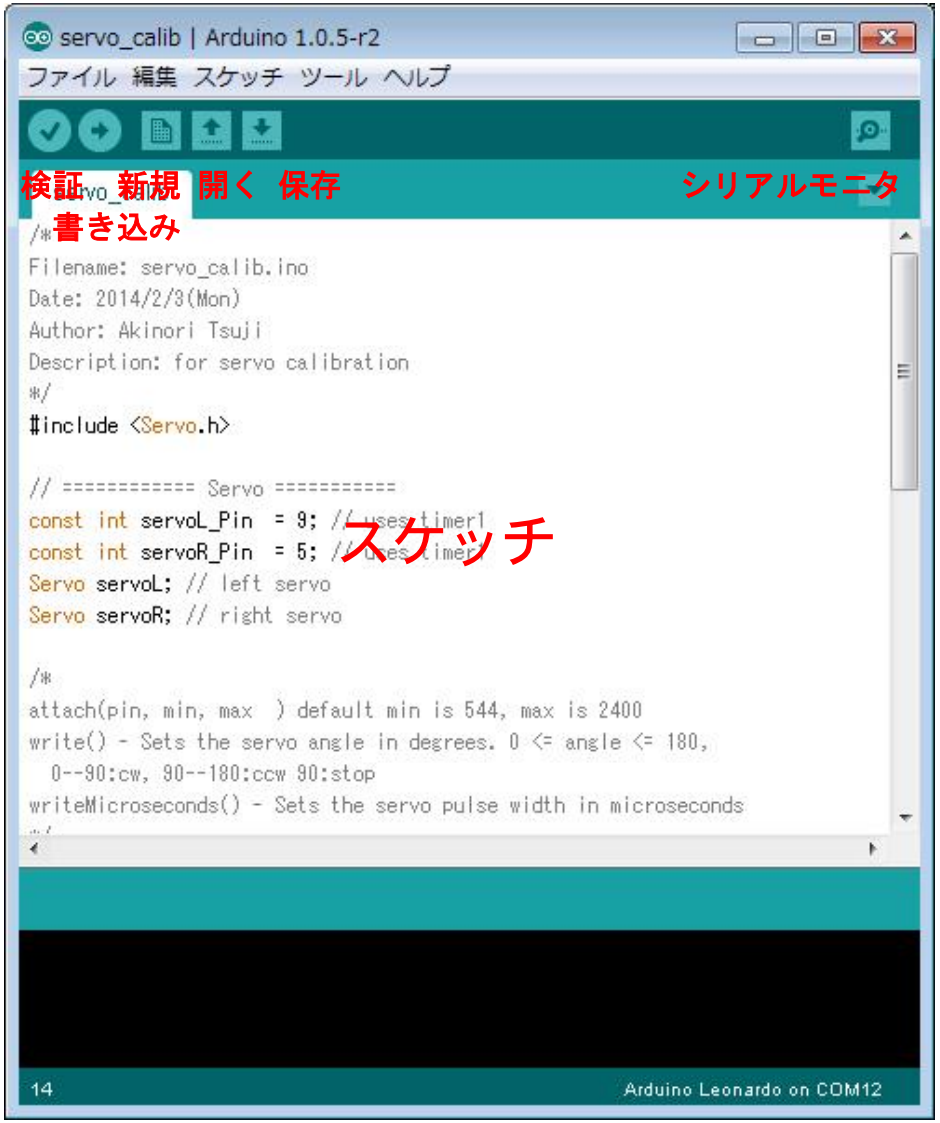
2) Arduino ソフトウェアを起動する.

3) ツール→シリアルポート→　COMxx

4) ツール→ボード→　Arduino Leonardo

※ COMxx が現れない場合には，ボードのリセットスイッチを押す.

※ 書き込み中にエラーが出た場合には 3),4)を確認する.



・ヘルプ

ヘルプ→リファレンス

＊Arduino 基本関数を調べられる.

・スケッチ例

ファイル→スケッチの例

・ライブラリの追加

ドキュメント￥Arduino￥libraries に，libraries.zip を展開（解凍）する．ファイル→スケッチの例にライブラリが含まれているかを確認する.

・ホームページ

http://www.arduino.cc/

6. マイコンの動作確認プログラム一覧

(1) LED led_blink.ino
　LED を点滅させる.
(2) スイッチ button.ino
　ボタンの状態を検出する.
(3) フルカラーLED tape_led.ino
　RGB,HSV で点灯する.
(4) ブザー toneMelody.ino
　音階に合わせて音を鳴らす.
(5) 焦電型赤外線センサ pir_motion.ino
　人の動きを検出する.
(6) 赤外線受信センサ irremote_code.ino, irremote_led.ino
　リモコンのボタンコードを調べる.
　リモコンのボタンコードに合わせて LED を点灯する.
(7) 赤外線 LED irsend_sony.ino
　リモコンのボタンコードを送信する.
(8) サーボモータ servo_calib.ino, servo_angle.ino, servo_cruise.ino, servo_remocon.ino
　サーボモータの基準点（90 度：停止位置）を調べる.
　サーボモータの角度（0 度～180 度）を調べる.
　ロボットをリモコンのボタンコードで制御する.
(9) Bluetooth Low Energy（BLE）モジュール Bluetooth_hm11.ino
　Bluetooth モジュールを他の機器から検出する.（マスタ，スレーブ）
(10) ロボットの制御（ライン検出） servo_pho
　センサ 1 つ，ライン検出で直進，検出なしで探索（回転）
(11) ロボットの制御（ラインセンサ）servo_pid,
　PID 制御によるライントレース
(12) ロボットの制御（距離センサ）dist_control, dist_control_p
　目標の閾値値による制御，目標に対して比例制御
(13) ロボットの制御（ラインセンサ，距離センサ）servo_pid_dist
　ライントレースとターゲット追従の組み合わせ

7. 改編履歴

日時	名前	内容
2014 年 4 月 27 日（日）	辻　明典	新規作成
2014 年 9 月 18 日（木）	辻　明典	1.0 版 ワークショップ
2014 年 9 月 26 日（金）	辻　明典	1.1 版