多様経験型講座

ロボトレースカー講座

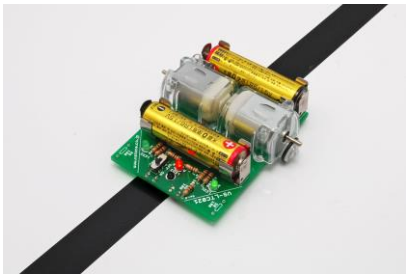
熊本高等専門学校　高専ハカセ塾
2022　9/17(土) 講座
2022 10/22(土) 試走会
2022 10/23(日) 大会参加

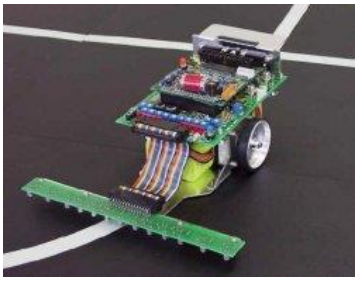
目次

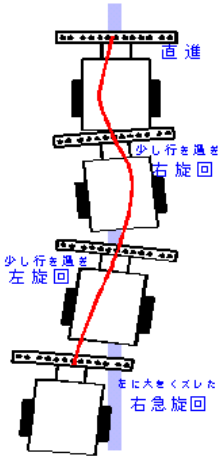
1. ライントレースロボットとは
   （ロボトレース競技について，ルール）
2. 演習に使うロボットの説明
3. ライントレースの動作原理
4. Arduino（開発環境）について
5. ロボットを使った演習
6. ロボットのプログラムについて

ラインとレースロボットとは

- 白い床に黒線，または黒い床に白線で引かれたライン上をトレースして走るロボット．
- 線の有無を検知する光センサと，移動用の車輪を有する．
- ロボットの入門用によく使われるが，一定のルールで走行速度を競う各種レース競技がある．





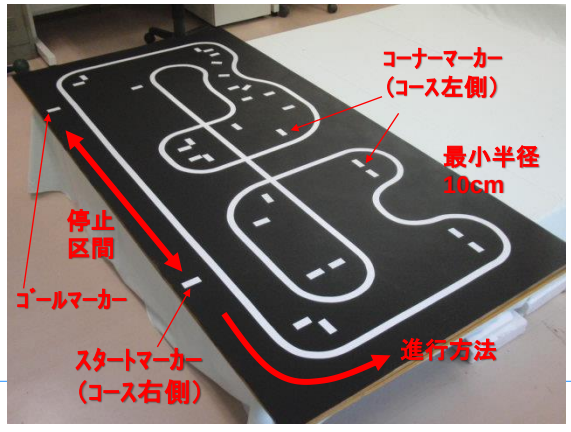


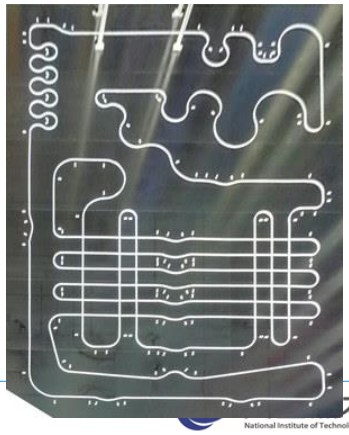
国立熊本高専
National Institute of Technology, Kumamoto College

3

ロボトレース競技とは

迷路を解いて走行する時間を競うマイクロマウスは，30年以上続く草分け的なロボット競技．
大会では一定のルールで構成されたコースで速度を競うロボトレース競技も実施されている．
・黒地に白のラインをトレースするロボット競技
・スタート・ゴールマーカーを検知し範囲内で自動停止しなければならない
・コースの曲率が変化する部分にマーカーあり
・5分の競技時間内に3回の走行が可能．最短の走行時間を競う．
・ソフトウェアの工夫によりタイム短縮が可能（ログ走行，ライン上に機体があればよい）

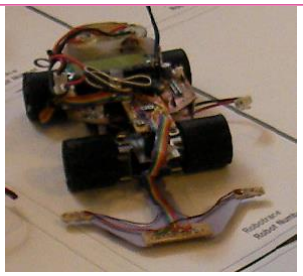




競技コース例

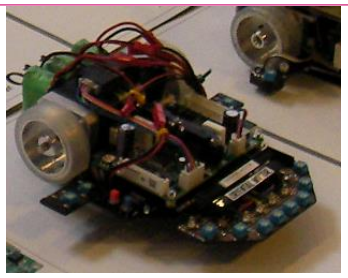
4

ロボットの構造



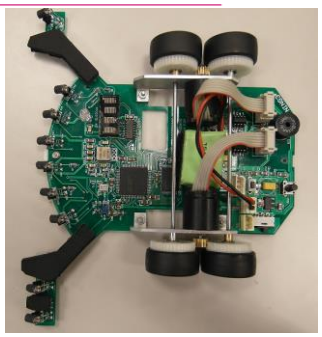
・DCモータ駆動
・サーボによるステアリング
・ステアリングとセンサが連結

利点：ラインの追従性がよい
欠点：小径ターンのステアリングに機械的な限界



・ステッピングモータ駆動
・ステアリングなし
・デジタルセンサ

利点：制御容易
欠点：高イナーシャ，加速性能
・旋回性能が悪い



第30回大会優勝者のロボット

・エンコーダ内蔵DCモータ駆動
・変形4輪駆動
・ステアリングなし
・アナログセンサ？

利点：軽量・低イナーシャ，速度制御性が良い，グリップが良い
欠点：制御が難しい

競技のポイント！
・直線の最高速度
・最小ターン半径10cmを走りぬける旋回性
・ショートカットで減速させない走行

国立熊本高専 National Institute of Technology, Kumamoto College

マイクロマウス九州地区大会に出場しよう！

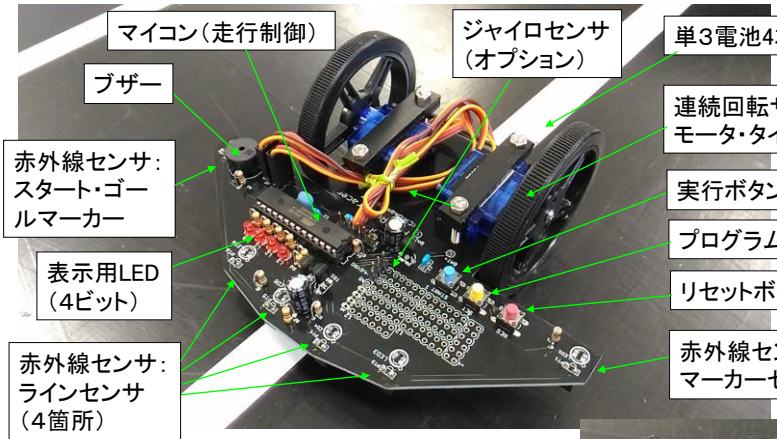
2022年10月23日にマイクロマウス九州地区大会が熊本高専にて開催されます．競技は3種目，
・マイクロマウス競技
・クラシックマウス競技
・ロボトレース競技
今回演習するライントレースロボットはロボトレース競技のコースをルールに沿って走ることができる機体です．

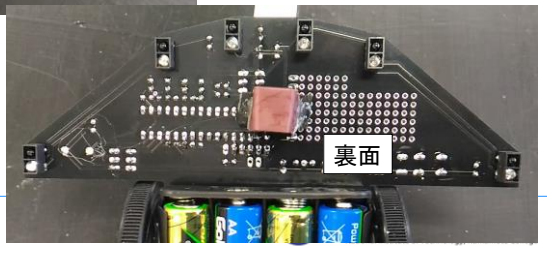
9/17・・・原理説明と簡単な調整
10/22・・・調整と試走会
10/23・・・上位の人は九州地区大会に出場
（他の人は，大会とオープンキャンパスを見学）
https://sites.google.com/site/mousekyushu/2022/

競技のポイント！
・直線の最高速度
・最小ターン半径10cmを走りぬける旋回性
・ログ走行，ショートカットで減速させない走行

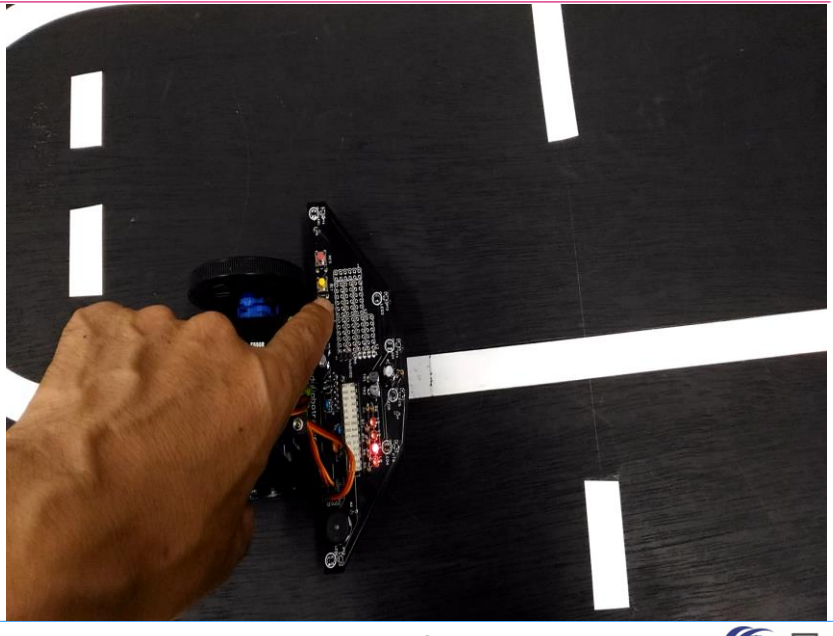
国立熊本高専 National Institute of Technology, Kumamoto College

ロボトレーサのハードウェア

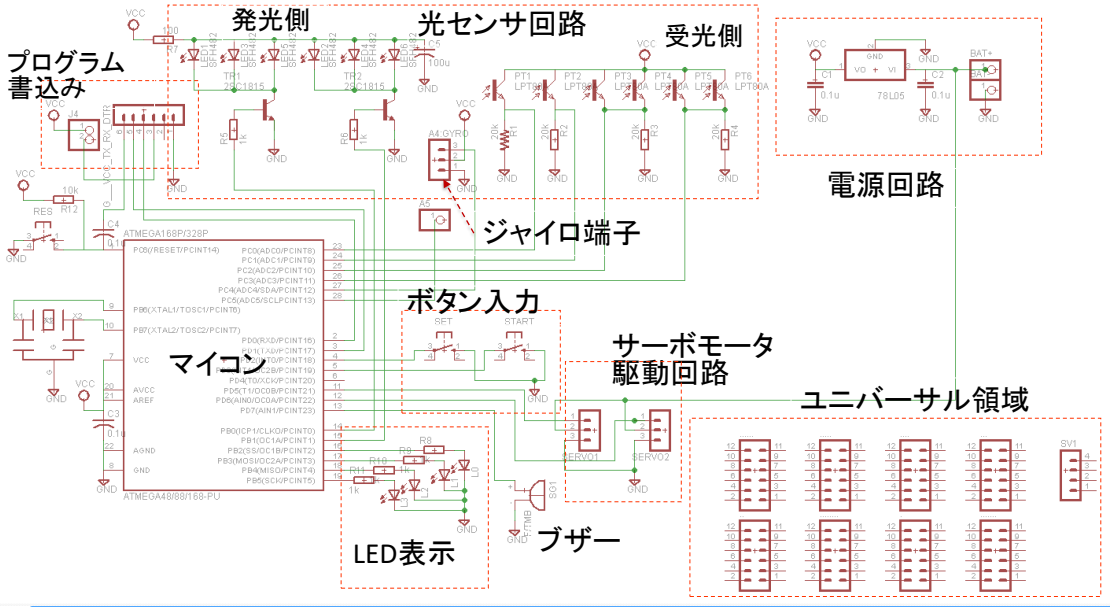




教材のロボトレーサの動画

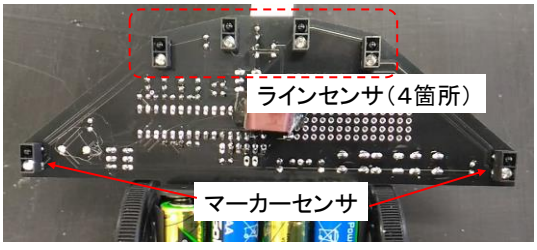


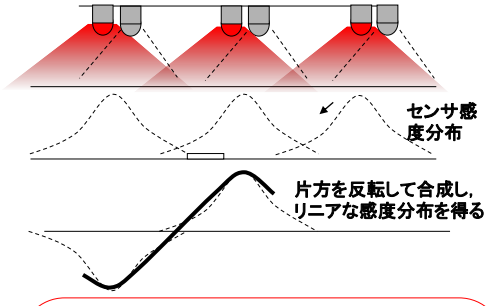
ロボトレーサの回路



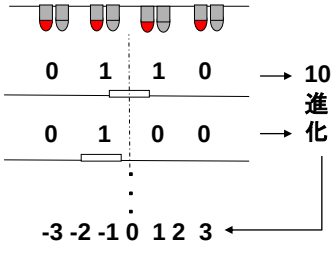
国立熊本高専 National Institute of Technology, Kumamoto College

ラインセンサについて



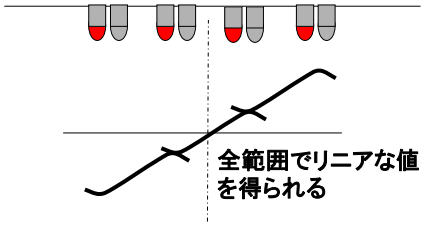


デジタル処理



配列を使って位置を-3から3の数値に変換

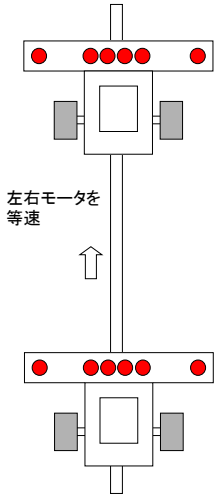
アナログ処理



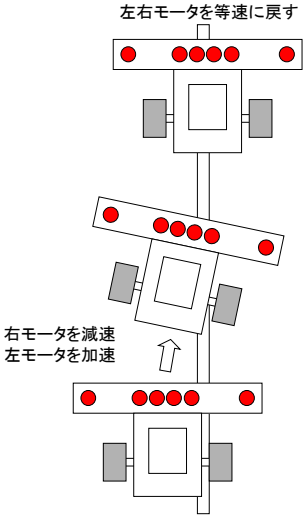
国立熊本高専 National Institute of Technology, Kumamoto College

ロボトレーサの走行制御（1）

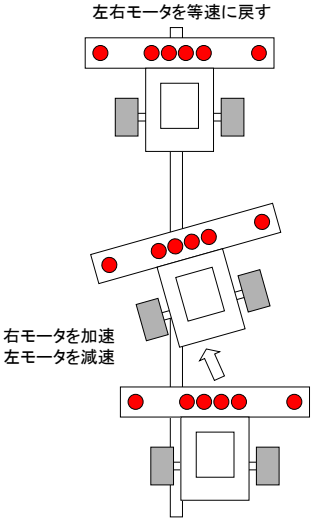
ライン上



ラインより左にある時



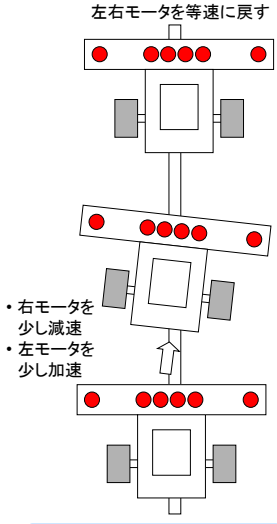
ラインより右にある時



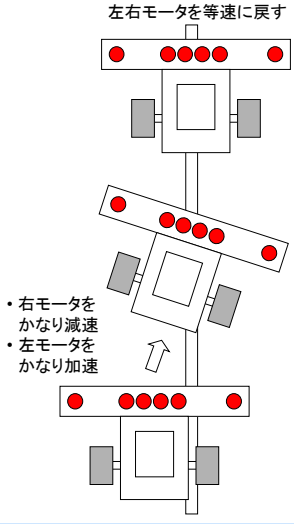
国立熊本高専 National Institute of Technology, Kumamoto College

ロボトレーサの走行制御（2）

ラインより少し左にある時



ラインよりかなり左にある時



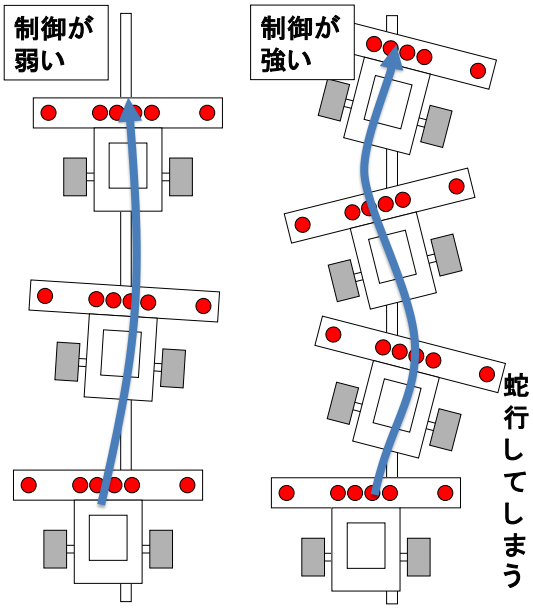
左タイヤの速度
＝前進速度＋ラインからの位置×比例係数

右タイヤの速度
＝前進速度ーラインからの位置×比例係数

比例・・・Proportional，　P制御

国立熊本高専 National Institute of Technology, Kumamoto College

ロボトレーサの走行制御（3）



左タイヤの速度
＝前進速度＋ラインからの位置×比例係数
　ー変化量×微分係数

右タイヤの速度
＝前進速度ーラインからの位置×比例係数
　＋変化量×微分係数

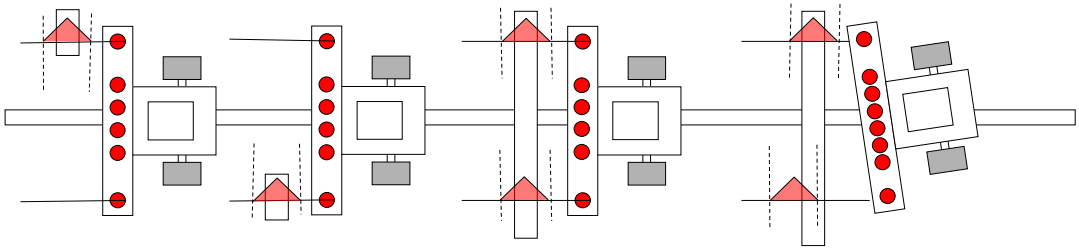
比例・・・Proportional，　P制御
微分・・・Differential，　D制御

両方合わせて，　PD制御

もし左右のタイヤの速度に最初から差がある時は，速度差を小さくした方が良い
積分・・・Intergral，I制御

全部合わせて，　PID制御

マーカーの検出

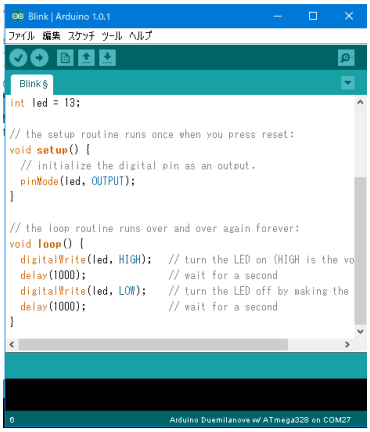


片側のマーカーを検知した瞬間に判断するのではだめ．
なぜなら，交差点を斜めに入った時にもマーカーだと検知してしまう．

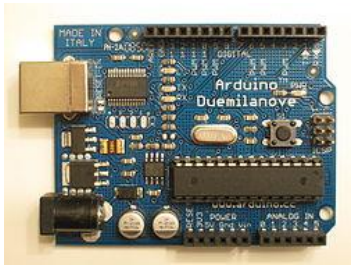
マーカーの終わりに判断をする．
・・・・片方のマーカーが消える前にもう一方のマーカーを検知したら交差点．
　　もう一方が検知でき成れば，それはマーカー．

Arduinoとは？

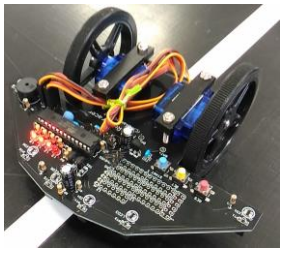
Arduino（アルドゥイーノ もしくは アルデュイーノまたはアルディーノ）とは、（ハードウェアの）「Arduinoボード」、および（ソフトウェアの）「Arduino IDE」から構成されるシステムである。IDEで作成したプログラムをコンパイル・デバッグ等し、それをArduinoボードに転送，実行できる.



Arduino IDE(開発環境)
https://www.arduino.cc/よりダウンロード可能



Arduinoボード



ArduRobotracer

オープンソースのハードウェアなのでボードの自作も可能，多くの互換ボードあり，ハカセ塾で使うロボトレーサーもArduino互換のボード

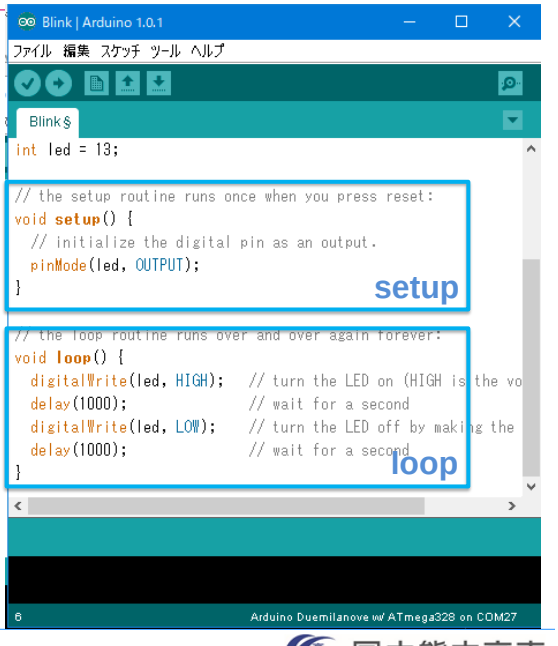
国立熊本高専 National Institute of Technology, Kumamoto College

Arduinoのスケッチ

基本的に処理の流れは2つ.

setup・・・最初に1度実行
loop ・・・処理を繰返し実行

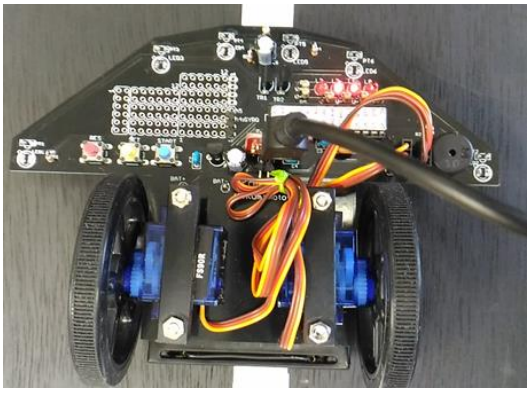
```
int led = 13;     // 13番ピンをLEDに

void setup() {
pinMode(led, OUTPUT);    // 13番は出力ピン
}

void loop() {
  digitalWrite(led, HIGH);   // LEDを点灯
  delay(1000);               // 1秒待つ
  digitalWrite(led, LOW) ;   // LEDを消灯
  delay(1000);               // 1秒待つ
}
```

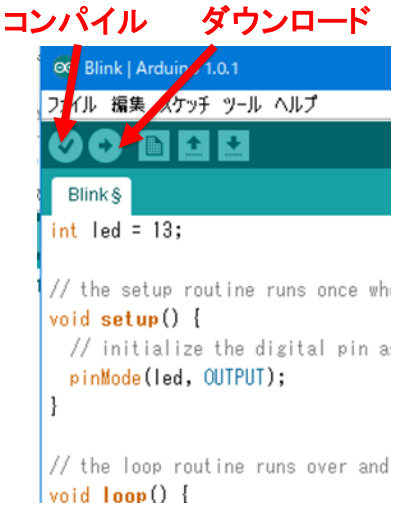


国立熊本高専 National Institute of Technology, Kumamoto College

Arduinoのコンパイル，ダウンロード

(1) USB-シリアル変換ケーブルでPCとロボトレーサーを接続
(2) ボードの種類を選択（Arduino Deumilanove）
(3)ボードを認識されるシリアルポートを選択
(4) ダウンロード（コンパイルだけも可）





http://www.musashinodenpa.com/arduino/ref/

演習１

```
//スイッチ説明
//赤：リセットボタン，　黄色：セットボタン，　青：スタートボタン

// 演習１
// 連続回転サーボのオフセットを調整する.
// サーボモータは 0-180 の値で回転する.
// 90が中央値で，これより大きいと正回転，小さいと逆回転
#define OFFSETL 90  // サーボモーターのオフセット
#define OFFSETR 90  // サーボモーターのオフセット
```

【解説】
プログラムでは中央値になっていもサーボモーターの初期設定の誤差で
モーターが回りだしてしまうことがある.

- プログラムのオフセットの値（90）を少し増やすか減らすようにプログラムを書き換えて書込みする.
- サーボモーターの設定ボリュームをドライバーで調整する.

演習２

```
// 演習２
// モーターを回転させてみる.
// セットボタンで0-3までプログラムを選んでスタートボタンを押す（LEDに2進数表示）
//  0:前進，1:バック，2:右回転，3:左回転
```

【解説】
セットボタン（黄）を押すとプログラムが変わる．プログラム番号がLEDに表示．リセットボタン（赤）で0に戻る．
プログラム番号を選んでスタートボタン（青）を押すと，
1秒間モーターが回る．

0:前進　・・・・　0000　　0が消灯，1が点灯
1:バック　・・・・　0001
2:右回転・・・・　0010
3:左回転・・・・　0011

10進数：　2進数
0: 0000
1: 0001
2: 0010
3: 0011
4: 0100
5: 0101
6: 0110
7: 0111
8: 1000
9: 1001
10: 1010
11: 1011
12: 1100
13: 1101
14: 1110
15: 1111

演習3

```
// 演習3
//  スピードを変えてみる
#define TSPD 20     // モーターテストプログラムの回転速度
```

【解説】
初期値20を書き換えるとモーターの回転速度が変わる.
書き変えた後はマイコンに書込み.
最低は0，最高は90まで.（マイナスの値を入れると逆回転）

演習4

```
// 演習4
// 真っ直ぐ走らないない時は，回転速度を補正する
#define KR 1.0      // 右モーターの回転速度の補正値
#define KL 0.9      // 左モーターの回転速度の補正値
```

【解説】
左右モーターにはわずかに差があり，同じ速度で回転させても真っ直ぐ前進しない．差が大きい場合は，プログラムで補正することができる.
モーターに回転を与える時にここで設定した補正値をかけて回転させている.

演習5

// 演習5
// ラインセンサの確認
// プログラム5を選んでスタートすると白線の位置がLEDとPCに出力される.
// 左端で－3，中央で0，右端で3になる．値が滑らかに変わるように次の演習でしきい値を調整

【解説】
4つのラインセンサの検知状態が対応するLEDに表示される.
しきい値より強く反射光を検出したらLEDをON,弱ければOFF.
PCではシリアルモニタを開いておくと，値が連続して表示される.

演習6

// 演習6
// 個々の光センサを調整する.
// プログラム4を選んでスタートすると，PCにセンサの値が表示される.
// センサ黒い板の上だと値が小さく，白線の上だと反射光により値が大きくなる.
// 白線の上だと判断する境目の値（しきい値）を修正する.
#define CTH　300　// コーナーマーカーのしきい値，SGマーカーより低くして，交差点の検出につかう.
#define SGTH　500　// スタート，ゴールマーカーのしきい値
#define S3TH　100　// センサ3のしきい値
#define S4TH　100　// センサ4のしきい値,
#define S5TH　500　// センサ5のしきい値
#define S6TH　500　// センサ6のしきい値

【解説】
スタート・ゴールマーカーが間違って検知してしまうとロボットが止まってしまうのでしきい値を少し高めに設定．PCではシリアルモニタを開いておく.

演習7

```
// 演習7
// 速度可変のライントレース走行(フリーラン)
// プログラム6を選んでスタートするとライントレース走行を行う.
// 速度の初期値は1，セットボタンを押すたびに速度が増加する．速度はLEDに
2進数で表示.
#define FR0VC  1   // 速度の初期値
#define FR0KP  3   // 比例係数の初期値
#define FR0KD  0   // 微分係数の初期値
```

【解説】
速度の初期は1なので最初はロボットは動かない．黄色ボタンを何度か押すと動き出す.
速度が遅いと蛇行する.
ちょうどよくライントレースする速度がある.
速すぎるとコースアウトしてビープ音がなる.

演習8

```
// 演習8
// 比例係数可変のライントレース走行(フリーラン)
// プログラム7を選んでスタートするとライントレース走行を行う.
// 比例係数の初期値は1，セットボタンを押すたびに比例係数が増加する．比
例係数はLEDに2進数で表示.
#define FR1VC  10  // 速度の初期値
#define FR1KP  1   // 比例係数の初期値
#define FR1KD  0   // 微分係数の初期値
```

【解説】
速度の初期は10で走り出すが，比例係数が小さいのでいきなりコースアウトしてしまう．ロボットを床に落としてしまわないように注意.
黄色ボタンを何度か押すとライントレースをはじめる.
比例係数が小さいと大回り
ちょうどよくライントレースする比例係数がある.
比例係数が大きすぎると蛇行してしまう.

演習9

// 演習9
// 微分係数可変のライントレース走行(フリーラン)
// プログラム8を選んでスタートするとライントレース走行を行う.
// 微分係数の初期値は0，セットボタンを押すたびに微分係数が増加する．微分係数はLEDに2進数で表示.
// ただし，原理通りに上手くはいかない.
#define FR2VC 10 // 速度の初期値
#define FR2KP 5 // 比例係数の初期値
#define FR2KD 0 // 微分係数の初期値

【解説】
センサの値の時間的な変化を得たいが，センサの値が-3~3の7段階しか変化しないために精度が悪い．ここでは変則的にセンサの値が1変化する時の時間を割って微分としている．そのためD（微分）制御は上手く働かないかもしれない．上手く働けば，ちょうど良い微分係数の時に蛇行がおさまる.

演習10

// 演習10
// ライントレース競技走行，1回目
// プログラム9を選んでスタートすると競技走行を行う.
// 右側のスタートマーカーを検知して低いビープ音，
// 左側のコーナーマーカーを検知して高いビープ音，
// 交差点はマーカーを無視して直進，
// ゴールマーカーを検知して0.5秒後にストップ.
// 安全にゴールできる速度での安定走行を狙う.
#define CR1VC 10 // 速度の初期値
#define CR1KP 2 // 比例係数の初期値
#define CR1KD 0 // 微分係数の初期値

【解説】
競技では3回の走行が許される.
いろいろなコースを安定して走れる係数を探そう.

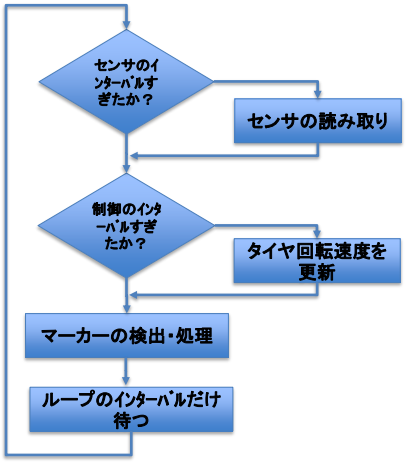
演習11，12

```
// 演習11
// ライントレース競技走行，2回目
// プログラム10を選んでスタートすると競技走行を行う.
// より高速にゴールを狙う.
#define CR2VC  15  // 速度の初期値
#define CR2KP  3   // 比例係数の初期値
#define CR2KD  1   // 微分係数の初期値

// 演習12
// ライントレース競技走行，3回目
// プログラム11を選んでスタートすると競技走行を行う.
// より高速にゴールを狙う.
#define CR3VC  20  // 速度の初期値
#define CR3KP  5   // 比例係数の初期値
#define CR3KD  2   // 微分係数の初期値
```


演習13

```
// 演習13
// 高度な設定を行うことができる.
// 光センサを読み取る時間間隔を調整できる. 高速走行ではセンサ読み取りが間に合わない時がある.短すぎると誤動作するので注意.
#define SENSINT 5   // センサーのインターバル(ms)
// モーターの制御を行う時間間隔を調整できる.
// 機械的な動きに合わせて調整すると良くなる. 早すぎてて遅すぎてもだめ.
#define CTRLINT 20  // 制御のインターバル(ms)
// マーカーの検出処理を繰り返す時間間隔を調整できる. 誤作動を防ぐために適当な時間が必要.
#define LPINT 2     // ループのインターバル(ms)
```

ライントレース走行の処理



全プログラム(定数の定義)

```
//--------------------------------------------------------------------------
//  演習用ロボトレーサ(ArduRobotracer), タイマーを使わない
// バージョン                                  葉山清輝（熊本高専）
// 2019.9.2-14 ver1.0 完成
//--------------------------------------------------------------------------
#define OFFSETL 90  // サーボモーターのオフセット
#define OFFSETR 90  // サーボモーターのオフセット
#define TSPD 20     // モーターテストプログラムの回転速度
#define KR 1.0      // 右モーターの回転速度の補正値
#define KL 0.9      // 左モーターの回転速度の補正値
#define CTH   300   // コーナーマーカーのしきい値，SGマーカー
                    //より低くして，交差点の検出につかう.
#define SGTH  500   // スタート，ゴールマーカーのしきい値
#define S3TH  100   // センサ3のしきい値
#define S4TH  100   // センサ4のしきい値,
#define S5TH  500   // センサ5のしきい値
#define S6TH  500   // センサ6のしきい値
#define FR0VC  1   // 速度の初期値
#define FR0KP  3   // 比例係数の初期値
#define FR0KD  0   // 微分係数の初期値
#define FR1VC  10  // 速度の初期値
#define FR1KP  1   // 比例係数の初期値
#define FR1KD  0   // 微分係数の初期値
#define FR2VC  10  // 速度の初期値
#define FR2KP  5   // 比例係数の初期値
#define FR2KD  0   // 微分係数の初期値
#define CR1VC  10  // 速度の初期値
#define CR1KP  2   // 比例係数の初期値
#define CR1KD  0   // 微分係数の初期値
#define CR2VC  15  // 速度の初期値
#define CR2KP  3   // 比例係数の初期値
#define CR2KD  1   // 微分係数の初期値
#define CR3VC  20  // 速度の初期値
#define CR3KP  5   // 比例係数の初期値
#define CR3KD  2   // 微分係数の初期値
#define SENSINT 5   // センサーのインターバル(ms)
#define CTRLINT 20  // 制御のインターバル(ms)
#define LPINT 2     // ループのインターバル(ms)
// ブザー ------------
#define BEEPH tone(BZ,1000,100)  // 1kHz
#define BEEPL tone(BZ,800,1000)  // 800Hz
// ピンアサイン----------
#define SET_SW    2     // セットスイッチ，デジタル入力
#define START_SW  3     // スタートスイッチ，デジタル入力
#define LEDOUT1 8    //ラインセンサ用LED出力()
#define LEDOUT2 9    //ラインセンサ用LED出力()
#define MOTOR_L  5  // 連続回転サーボ出力L
#define MOTOR_R  6  // 連続回転サーボ出力R
#define BZ 7    //ブザー用出力
// センサ入力ピン番号
#define SENS_C  A0  // マーカーセンサ1,コーナーマーカー
#define SENS_SG A1  //  マーカーセンサ2,スタート･ゴールマーカー
#define SENS_34 A2  // ラインセンサ3,4
#define SENS_56 A3  // ラインセンサ5,6
#define GYRO_IN A4  // ジャイロセンサ
#define SENS_OP A5  // センサオプション
```


全プログラム(変数の定義，LED 表示)

```
// 変数定義----------
int pmode=0;          // プログラムモード(メインメニューで分岐)
volatile int sensC, sensCB;  // ラインセンサ右，赤外線点灯前（以下同様）
volatile int sensSG, sensSGB;  // ラインセンサ中央
volatile int sens3, sens3B;  // ラインセンサ左
volatile int sens4, sens4B;  // ラインセンサ左
volatile int sens5, sens5B;  // ラインセンサ左
volatile int sens6, sens6B;  // ラインセンサ左
volatile int ledSW=0;
volatile int sensD,sensDout,sensDoutB;
volatile int sensCtime=100;
volatile long tchange=0;

// センサ値->ライン位置の返還行列，index6が中央で0,それ以外も0，全センサOFFで99
const int sensArray[16]={99,3,1,2,-1,0,0,0,-3,0,0,0,-2,0,0,0};

#include <Servo.h>
Servo motorL;
Servo motorR;


// LED表示 ------------------------------------------------------------
void dispLED(byte n)
{
 PORTB &= B11000011;                // LEDを一旦消す
 PORTB |= (n<<2);                   // LED表示
}
```

全プログラム(赤外線センサ読取り)

```
//-------------------------------------------------------------------------
// 赤外線センサー読み取り，並列接続のセンサを左右交互に読み取る，
2回読んで更新
void sensors() {
  // 赤外線LED消灯時のセンサの読み
  sensCB=analogRead(SENS_C);
  sensSGB=analogRead(SENS_SG);
  sens3B=sens4B=analogRead(SENS_34);
  sens5B=sens6B=analogRead(SENS_56);

  // ledSWを反転しながら並列のセンサを交互に読む
  ledSW=!ledSW;
  if(ledSW){
    digitalWrite(LEDOUT1, HIGH);    // 赤外線LED点灯
    delayMicroseconds(20);          // ちょっと待つ
    sens3=analogRead(SENS_34)-sens3B;  // 赤外線点灯前後の
    sens5=analogRead(SENS_56)-sens5B;  // 差で読む
    sensC=analogRead(SENS_C)-sensCB;
    digitalWrite(LEDOUT1, LOW);    // 赤外線LED消灯
  } else {
    digitalWrite(LEDOUT2, HIGH);    // 赤外線LED点灯
    delayMicroseconds(20);          // ちょっと待つ
    sens4=analogRead(SENS_34)-sens4B;
    sens6=analogRead(SENS_56)-sens6B;
    sensSG=analogRead(SENS_SG)-sensSGB;
    digitalWrite(LEDOUT2, LOW);    // 赤外線LED消灯
  }

  // 各センサのON-OFFをラインとの位置の整数値，
  // －3から3までに変換
    sensD=0;
    if (sens3>S3TH ) sensD |= 0x08; else sensD &= ~(0x08);
    if (sens4>S4TH ) sensD |= 0x04; else sensD &= ~(0x04);
    if (sens5>S5TH ) sensD |= 0x02; else sensD &= ~(0x02);
    if (sens6>S6TH ) sensD |= 0x01; else sensD &= ~(0x01);
    sensDoutB=sensDout;
    sensDout=sensArray[sensD];
    if (sensDoutB != sensDout){
      if(sensDoutB>sensDout) sensCtime=millis()-tchange;
      else sensCtime=tchange-millis();
      tchange=millis();
    }
  }
```


全プログラム(センサ・モータテスト)

```
//テストプログラム ----------------------------------------------------
// 全赤外線センサテスト  ----------------------------------------
void testSensors(){
  while(digitalRead(START_SW)==HIGH){
    // センサー読み取り
    sensors(); delay(25);
    sensors(); delay(25);
    // センサ値をシリアル出力
    Serial.print("C:");Serial.print(sensC);
    Serial.print(" SG:");Serial.print(sensSG);
    Serial.print(" PT3:");Serial.print(sens3);
    Serial.print(" PT4:");Serial.print(sens4);
    Serial.print(" PT5:");Serial.print(sens5);
    Serial.print(" PT6:");Serial.println(sens6);
  }
}
// ライン位置の出力  ----------------------------------------
void testSensD(){
  while(digitalRead(START_SW)==HIGH){
    // センサー読み取り
    sensors(); delay(25);
    sensors(); delay(25);
    dispLED(sensD);
    // ライン位置をシリアル出力
    //Serial.print("D:"); Serial.print(sensD);
    Serial.print("POS:"); Serial.println(sensDout);
  }
}
```

```
//---------モーターテスト----------------------------------------------------
void testMotorF(){   // 前進
  motorL.write(OFFSETL+TSPD*KL);
  motorR.write(OFFSETR-TSPD*KR);
  delay(1000);
}
void testMotorB(){   // バック
  motorL.write(OFFSETL-TSPD*KL);
  motorR.write(OFFSETR+TSPD*KR);
  delay(1000);
}
void testMotorRturn(){   // 右ターン
  motorL.write(OFFSETL+TSPD*KL);
  motorR.write(OFFSETR+TSPD*KR);
  delay(1000);
}
void testMotorLturn(){   // 左ターン
  motorL.write(OFFSETL-TSPD*KL);
  motorR.write(OFFSETR-TSPD*KR);
  delay(1000);
}
```

全プログラム（ライントレース）

```
//フリーラン，パラメータ可変のライントレース----------------------------------
void freeRun(int fm, float vC, float kP, float kD ){
// fm=0: 速度可変，fm=1: 比例係数可変，fm=2:微分係数可変
// vC:中心速度，　kP:比例係数，　kD:微分係数，　kI:積分係数(未使用)
float vp;
int vR,vL;
int frun=0;　// スタート，ゴールマーカーの処理のための状態遷移
int markerC=0; // コーナーマーカー読み取り
int markerSG=0; // スタート･ゴールマーカー読み取り
long tms;　// ゴールを超えて静止するまでの時間ms.
long tsens;　// センサ読み取りの間隔の監視タイマー.
long tctrl;　// 制御の時間間隔の監視タイマー.

    tsens=tctrl=millis();  // タイマー初期化
    sensors(); delay(15); sensors(); delay(15); // センサー初期化
    if (fm==0) dispLED(vC);
    if (fm==1) dispLED(kP);
    if (fm==2) dispLED(kD);

    while(digitalRead(START_SW)){

      if (digitalRead(SET_SW)==0){　　// SETスイッチが押された時パラメータを変える
        while(digitalRead(SET_SW)==0);  // SETを離すまで待つ
        if (fm==0){vC++; dispLED(vC); }
        if (fm==1){kP++; dispLED(kP); }
        if (fm==2){kD++; dispLED(kD); }
      }

      // センサー読み取り
      if (millis()>(tsens+SENSINT)){
        tsens=millis();
        sensors();  // 片側ずつ読むので2回呼び出して全部更新
      }
      // センサはずれたときはコースアウトとしてBEEP
      if (sensDout==99){ BEEPL; sensDout=0; }
      Serial.println(sensCtime);

      // モーター制御
      if (millis()>(tctrl+CTRLINT)){
        tctrl=millis();

        vp=(float)sensDout*kP-10.0/(float)sensCtime*kD;
        vR=OFFSETR-vC+vp*KR;
        vR=constrain(vR,0,180);
        vL=OFFSETL+vC+vp*KL;
        vL=constrain(vL,0,180);

        motorL.write(vL);
        motorR.write(vR);
      }

      delay(LPINT);  // マーカー処理のインターバル
    }

    // モータ停止
    motorL.write(OFFSETL);
    motorR.write(OFFSETR);
}
```


全プログラム(競技走行)

```
//ライントレース走行----------------------------------------------------
void run1(float vC, float kP, float kD ){
// vC:中心速度，　kP:比例係数，　kD:微分係数，　kI:積分(未使用)
float vp;
int vR,vL;
int frun=0;　// スタート，ゴールマーカーの処理のための状態遷移
int markerC=0; // コーナーマーカー読み取り
int markerSG=0; // スタート･ゴールマーカー読み取り
long tms;　// ゴールを超えて静止するまでの時間ms.
long tsens;　// センサ読み取りの間隔の監視タイマー.
long tctrl;　// 制御の時間間隔の監視タイマー.

    tsens=tctrl=millis();  // タイマー初期化
    sensors(); delay(15); sensors(); delay(15); // センサー初期化

    while(digitalRead(START_SW)){
      // センサー読み取り
      if (millis()>(tsens+SENSINT)){
        tsens=millis();
        sensors();  // 片側ずつ読むので2回呼び出して全部更新
      }

      // モーター制御
      if (millis()>(tctrl+CTRLINT)){
        tctrl=millis();

        vp=(float)sensDout*kP-10.0/(float)sensCtime*kD;
        vR=OFFSETR-vC+vp*KR;
        vR=constrain(vR,0,180);
        vL=OFFSETL+vC+vp*KL;
        vL=constrain(vL,0,180);

        motorL.write(vL);
        motorR.write(vR);
      }

      // センサはずれたときはコースアウトとして終了
      if (sensDout==99){ BEEPL; break; }

      // コーナーマーカー処理
      if (markerC==0 && sensC>CTH) markerC=1;　// Cマーカーに乗った時
      if (markerC==1 && sensSG>SGTH) markerC=2;　// SGマーカーで交差点
      if (markerC==2 && sensC<CTH) markerC=0;　// 交差点はフラグを戻す
      if (markerC==1 && sensC<CTH) {　　// Cマーカーを超えた時の処理
        markerC=0;
        BEEPH;

      }

      // スタート･ゴールマーカー処理
      if (markerSG==0 && sensSG>SGTH) markerSG=1; // マーカーに乗った時
      if (markerSG==1 && sensC>CTH) markerSG=2;　// SGマーカーで交差点
      if (markerSG==2 && sensSG<SGTH) markerSG=0; // 交差点はフラグを戻す
      if (markerSG==1 && sensSG<SGTH) {　　// SGマーカーの処理
        markerSG=0;
        BEEPL;
        frun++;  // SGマーカー通過ごとに1足す
        if (frun==2) {frun=3; tms=millis(); BEEPL;} // 2回目がゴール，タイマーセット
      }
      if (frun==3 && millis()>(tms+500)) break;　// ゴールしたら0.5秒で停止

      delay(LPINT);  // マーカー処理のインターバル
    }

    // モータ停止
    motorL.write(OFFSETL);
    motorR.write(OFFSETR);
}
```

全プログラム（setup）

```
//---------------------------------------------------------------------
// セットアップ（最初に一度だけ実行）
//---------------------------------------------------------------------
void setup()
{
  DDRB |=B00111111;     // PB0,1: センサLED, PB2-5: 表示用LEDで出力
せってい,
                  // PB6,7はクリスタルなので入力設定
  pinMode(SET_SW, INPUT);
  pinMode(START_SW, INPUT);
  digitalWrite(SET_SW, HIGH);
  digitalWrite(START_SW, HIGH);

  pinMode(LEDOUT1, OUTPUT);
  pinMode(LEDOUT2, OUTPUT);
  digitalWrite(LEDOUT1, LOW);    // 赤外線LED消灯
  digitalWrite(LEDOUT2, LOW);    // 赤外線LED消灯

//  pinMode(BZ, OUTPUT);
//  digitalWrite(BZ, LOW);    // 赤外線LED消灯

  motorL.attach(MOTOR_L);
  motorR.attach(MOTOR_R);
  motorL.write(OFFSETL);
  motorR.write(OFFSETR);

  // シリアル通信スタート
  Serial.begin(9600);
}
```


全プログラム(loop)

```
//---------------------------------------------------------------------
// ループ（繰り返し実行）
//---------------------------------------------------------------------
void loop(){

 dispLED(pmode);                  // プログラムモードをLEDに2進表示

 while (digitalRead(START_SW)==HIGH){   // スタートスイッチが押されたら指定プログラムを実行
   if (digitalRead(SET_SW)==LOW) {      // セットスイッチが押されたらプログラムモードを進める
     delay(10);                   // チャタリング防止
     while(digitalRead(SET_SW)==LOW);
     delay(10);                   // チャタリング防止
     pmode++; if (pmode>15) pmode=0;  // モードを増やす
     dispLED(pmode);              // LEDに2進表示
     BEEPH;
   }
 }
 // スタートスイッチが押されたら処理プログラムへ分岐
 dispLED(0);
 delay(500);                      // 0.5秒待つ

 switch(pmode){                   // 選択されたプログラムモードのプログラムを実行
   case  0:  testMotorF(); break;
   case  1:  testMotorB(); break;
   case  2:  testMotorRturn(); break;
   case  3:  testMotorLturn(); break;

   case  4:  testSensors(); break; // 全センサーのテスト
   case  5:  testSensD(); break;  // センサ位置のテスト

   case  6:  freeRun(0,FR0VC,FR0KP,FR0KD); break;  // 速度可変のライントレース
   case  7:  freeRun(1,FR1VC,FR1KP,FR1KD); break; // 比例係数可変のライントレース
   case  8:  freeRun(2,FR2VC,FR2KP,FR2KD); break; // 微分係数可変のライントレース
   case  9:  run1(CR1VC,CR1KP,CR1KD); break;      // 1回目 ライントレース走行
   case  10: run1(CR2VC,CR2KP,CR2KD); break;      // 2回目 ライントレース走行
   case  11: run1(CR3VC,CR3KP,CR3KD); break;      // 3回目 ライントレース走行
 }
 motorL.write(OFFSETL);
 motorR.write(OFFSETR);
 delay(500);                      // 0.5秒待つ
}
```