一般社団法人 未来創生 STREAM 教育総合研究所

クリエイティブロボティクスコンテスト 2025

「タイムアタックレース部門競技ガイド」

2025 年 10 月 31 日

クリエイティブロボティクスコンテスト　運営本部

『タイムアタックレース部門 』本番用コースガイド

1. コース詳細

【コース全体のサイズ】134cm×360cm（製作工程の都合でシート全体のサイズが 5cm 程度前後する可能性がありますが、シートに印刷されているコースのサイズには影響ありません）。なお「ユポ紙」にて製作を行います。
【壁サイズ】厚さ 6.2cm×高さ 12cm ※100 円ショップ「ダイソー」の「発泡ミニブロック」を使用します。
色はランダムで、両面テープを用いてシートに貼り付けます。
【内壁素材】ポリプロピレン製シート　※100 円ショップ「ダイソー」の「PP シート　乳白色両面つや消し」を使用します。
ブロック高さのサイズ（12cm）にカットし、両面テープでブロックに貼付けます。
【走行部分の床色】白（一部床面に着色された部分があります。後述）

2. 競技ルール

競技当日に走行するコースは、エントリー時に試作してもらったコース（以下、予選コース）と異なります。
センサーを用いた自律走行によるコース攻略を行うため、事前に本番コースの具体的な形状やサイズは公表しません。ただし、「どのような仕様のコースを準備するか」については以下に記します。
なお、予選時のルールと今回記したルールが競合する場合は、今回記したルールを優先するものとします。

■ルール■

① 予選で使用したロボットを本選までに改良することができます。

② レース毎の乾電池の交換は自由とします。

③ ロボットのサイズは直径 30cm、高さ 50 ㎝の円柱の中におさまるように設計してください。

④ スタート時はロボットの先端がスタートラインより後方になるように設置します。

⑤ スタートの操作はロボプロシールドのスイッチまたはタッチセンサーを使用してください。

⑥ 当日は、各出場者 2 回ずつ走行が可能です。すべての出場者が 1 回ずつ走行したあと、同じ順番でもう一巡走行することになります。1 回目と 2 回目の間でロボットやプログラムの修正を行っても構いませんが、2 回目の順番が回ってきた際に走行が行える状況にない場合は走行順の入れ替え、2 回目走行の中止などの判断を下す場合があります。

⑥ 別紙「禁止規定」に抵触するものを使用した場合は失格となります。

⑦ 大会当日は、開会前など一部の時間で実際に走行するコースを開放し、スタッフ管理の元で練習走行ができます。

■コースの改変ポイント■

① コース壁は予選コースと変更なく、厚さ 6.2 ㎝、高さ 12 ㎝のスチロール製ブロックを使用します。ただし、表面の摩擦をよくする目的から、PP シートを側面全体に貼りつけます（詳細は前述）。

② コース全体のサイズは、予選コースより大きなものとなります。

③ コース内の走行ルート、つまり道の幅は一律 30 ㎝を保ちます。曲がり角や交差点を設ける場合、2 本の道は必ず直角に交わります。

⑥ コース全体を通して、無線コントローラを行ったり、人間側が意図的な動作でセンサーを反応させたりすることでの遠隔操縦は出来ません。

⑦ スタートライン侵入後からゴールエリアまでは、進行方向右側の壁に沿って進めばかならずたどり着ける形になっています（この経路を以下「正規ルート」と呼びます）。

⑧ エリア途中に 1 か所、「分岐エリア」を設けます。分岐エリアには直進方向の正規ルートとは別に、左に曲がることで侵入できる「近道」があります。分岐エリアは床面を赤色に着色します。なお、分岐エリアの数は一つとは限りません。

⑨ 近道を通ると正規ルートより短い距離でゴールにたどり着けます。ただし、近道へは左に曲がることで侵入するため、右手の壁に沿って走行している場合は「分岐エリアを検知し、近道に入るための特別な動き」が必要です。はじめから進行方向左側の壁に沿って進んでいれば容易に近道に侵入できますが、近道内は左手の壁に沿って進むとかえって遠回りになるような形状になっています。

⑩ ゴールエリア地点には、床を緑色に着色した「ゴールエリア」を設けます。ロボットの接地部分がゴールエリアに触れた時点でゴールと判定されます（ゴール後、ロボットを停止させる必要はありません）。

① コース内の直線（曲がり角、分岐エリア、ゴールエリアのいずれでもない部分）のうち 1 か所、床面を青色に着色します。ただし、曲がり角や分岐エリアから 15cm 以内の部分は着色しません。

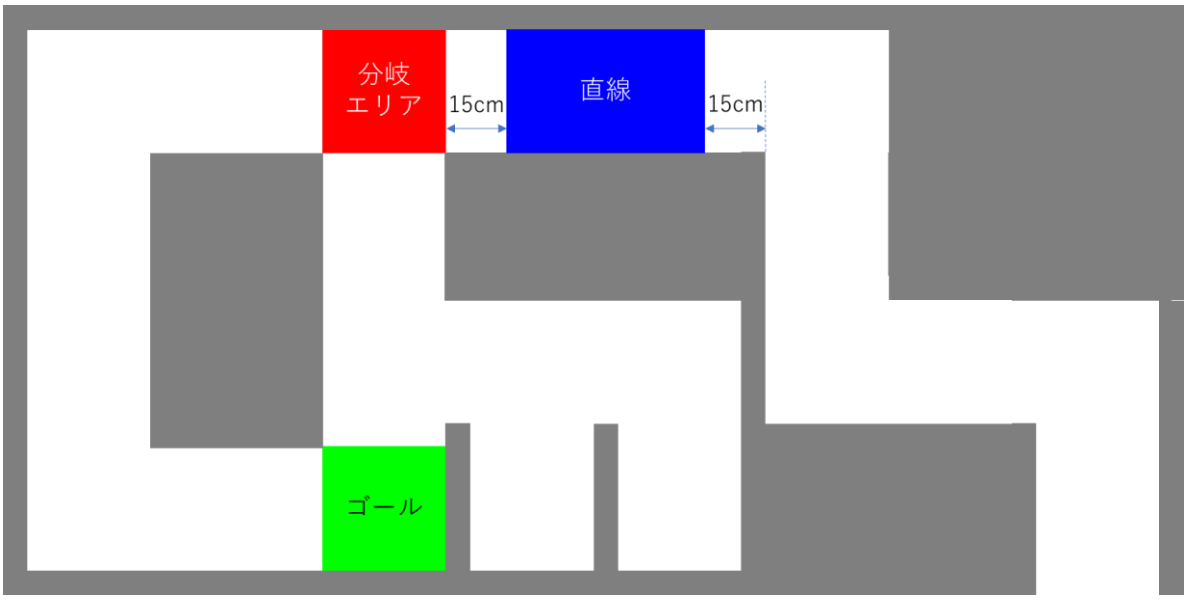
⑪ コース内の壁に接触することは問題ありません。ただし壁を破壊する、移動させるなどの方法を用い、ロボットがコース外を走行した場合は走行中止の判断を下す場合があります。

安定を重視して正規ルートで着実にゴールを目指すか、走行不能やタイムロスになるリスクを承知の上で近道に入るかは、出場者各自の判断に委ねます。

タイムは 2 回走行のうち、より良い記録を採用します。走行中止となった場合はその回のタイムは記録されませんが、ロボットの作りや動作に見られた工夫や、途中まで走行できた場合はその走破距離なども評価の対象となる場合があります。

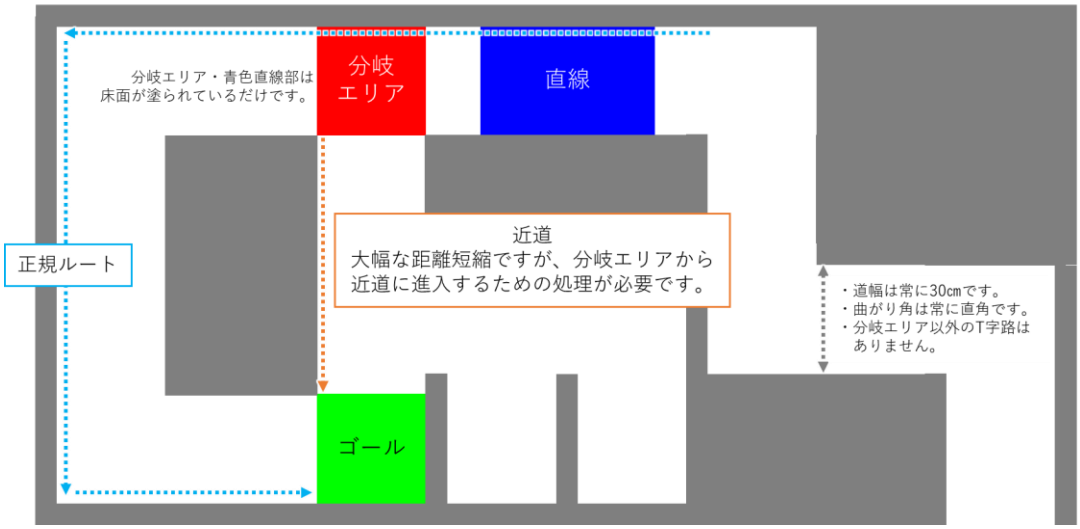
参考のため、これらの改変ポイントを全て取り入れて作成したコースレイアウトの図を以下に示します。

※あくまでルールを理解しやすくするための仮の図であり、当日コースのレイアウトとは形状が異なります。

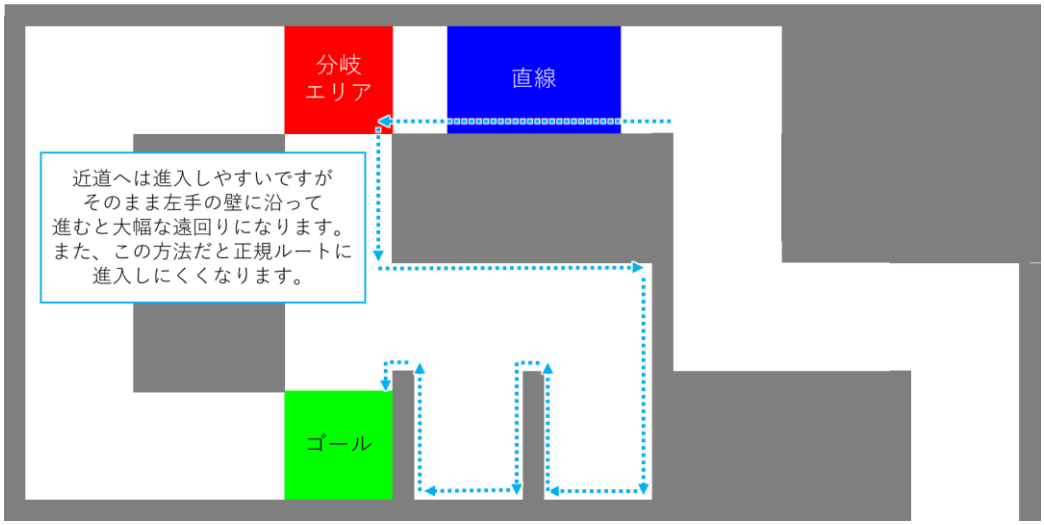


このコースを左右の壁に沿って走行する場合、それぞれ以下のようになります。

後半エリア内で「右手の壁に沿って」進んだ場合



後半エリア内で「左手の壁に沿って」進んだ場合



3. タイムアタックレース部門攻略法

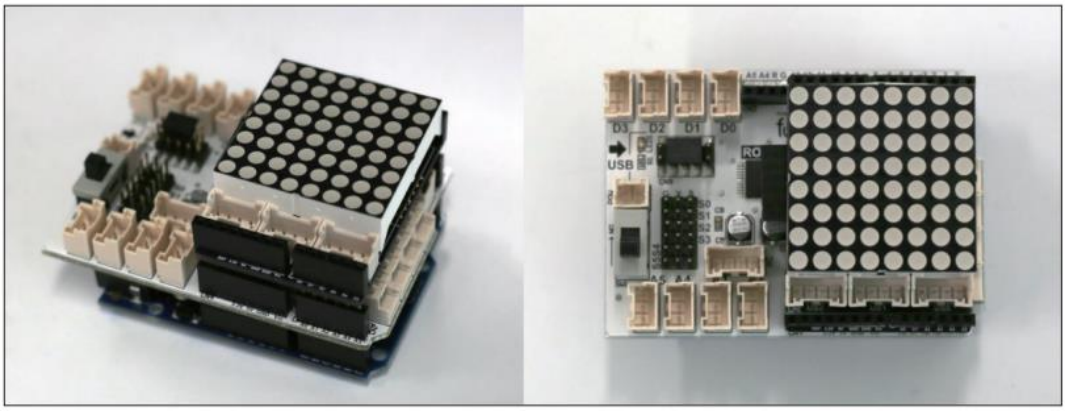
先に記した通り、本番当日では特に後半エリアにおいて「センサーを用いた自律制御」が事実上必須となります。
ロボプロキット内に登場するセンサーは毎月必ず使うものではないため、たとえば同じ 2 年目コースの選手であっても『もう●●センサーについて学んだ選手』と『まだ学んでいない選手』が混在することになります。大会に公正を期すため、ここにコース攻略に利用できる見込みの高い一部のセンサーの使い方を記します。必要に応じて参考にしてください。

1. 超音波距離センサーの使い方

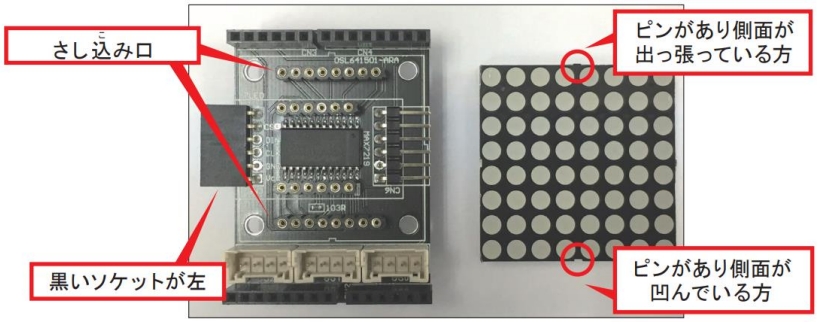
【超音波距離センサーについて】
超音波距離センサーは、人間の耳には聞こえない音（超音波）を発してその反射具合をチェックすることで、前方の障害物の有無や障害物までの距離をはかることができるセンサーです。
「壁からの距離が●cm 以上だったら▲▲する」のような、壁からの距離に応じた条件分岐をするのに役立ちます。

【超音波距離センサーの接続と動作確認】
① 超音波距離センサーをオレンジのセンサーカバーと、白色のセンサーL 字ステイではさみこみ、M3L6 タッピングネジ(B)を 2 本使ってとめます。

② マイコンボードにロボプロシールド、マトリクス LED シールド、マトリクス LED を接続します。

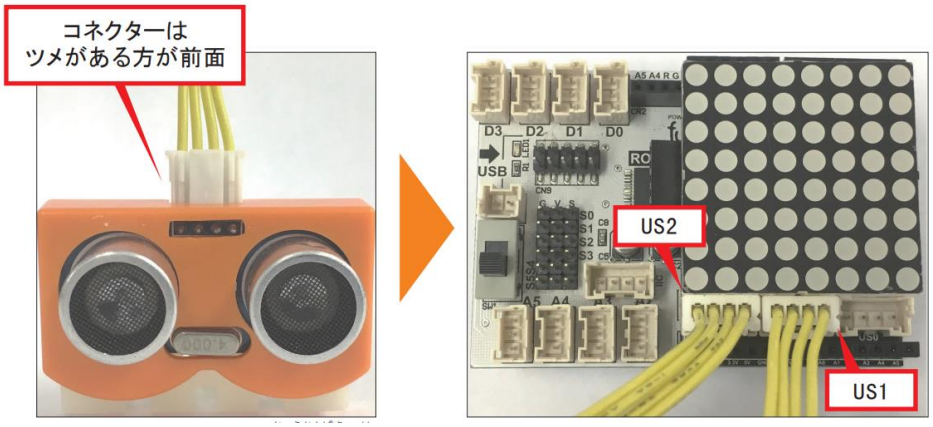


マトリクス LED をマトリクス LED シールドに接続するときには、向きに注意してください。



なおマトリクス LED シールドは超音波距離センサーの接続に必須ですが、マトリクス LED の方は最初の動作確認で必要になるだけなのでそのあとは取り外しても構いません。

③　マトリクス LED シールドと超音波距離センサーを、黄色のセンサーケーブルで接続する。



※画像では US1・US2 の 2 か所にセンサーが接続されていますが、はじめの動作確認は US1 に接続したセンサーのみを使うので接続するセンサーは 1 つで構いません。

！注意！　超音波距離センサーにセンサーケーブルを接続する際には、画像のとおりツメがある方がセンサー前面を向くようにしてください。ケーブルの構造上逆向きにも挿すことができますが、この状態で電源を入れると回路がショートし、発熱・発煙の原因となる場合があります。

④　以下のプログラムを書き込み実行する。
RoboticsProfessorCourse1 ＞ MagicItemB1 ＞ USSTest

超音波距離センサーの動作確認用のプログラムです。
成功すると、マトリクス LED に 2 桁の数字が表示されます。この数字は超音波距離センサーがはかっている「障害物までの距離」を表しています。センサーを手に持って動かしてみると、数字が変化するはずです。なお、単位はcmです。
もしうまく動作しない場合は、以下の方法で問題箇所を特定しましょう。

＜そもそもマトリクス LED に何も表示されない場合＞
マイコンボード、ロボプロシールド、マトリクス LED シールド、マトリクス LED いずれかの不具合である可能性があります。
まずは超音波距離センサーを取り外し、マトリクス LED の動作確認プログラムを実行してみましょう。
RoboticsProfessorCourse1 ＞ MagicItemB1 ＞ MatrixTest
これでもマトリクス LED が光らない場合は、上記の 4 パーツを 1 つずつ別キットのものと交換し、どのパーツに問題があるか特定しましょう。なお、交換用のパーツの確保や問題パーツが特定出来た後の対応は、教室の先生にお願いしてください。

＜マトリクス LED は光るが数値が変化しない場合＞
マイコンボード、ロボプロシールド、マトリクス LED シールド、センサーケーブル、超音波距離センサーのいずれかの不具合である可能性があります。やはり上記のパーツを 1 つずつ別キットのものと交換し、不具合箇所を特定して対応してください。
ただし、超音波距離センサーは「前方に音（超音波）を発して、壁や障害物に当たって跳ね返ってきたものを検知する」という原理です。音が反射しづらい壁や、斜面・曲面に反射させたときなどはうまく検知できない場合もあります。

【超音波距離センサーのプログラムについて】
動作確認のプログラムを使って、超音波距離センサーのプログラムのつくりについて説明します。

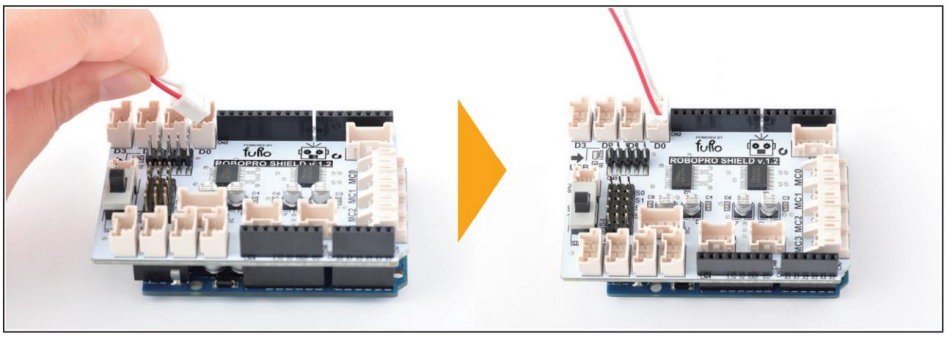
```
7  //ライブラリを取り込む
8  #include <Sprite.h>
9  #include <Matrix.h>
10 #include <RPlib.h>
11
12 Matrix myMatrix = Matrix(11, 13, 1);  //マトリクスLEDを使うときのオマジナイ
13
14 void setup(){
15     myMatrix.setBrightness(8);         //マトリクスLEDの明るさを設定(0-8)
16 }
17
18 void loop(){
19     myMatrix.clear();
20     myMatrix.putd2(0, 0, ussRead(US1));         //マトリクスLEDに2けたの数字を表示
21     delay(100);
22 }
```

・10 行目「#include <RPlib.h>」
超音波距離センサーを使う場合、プログラムの初めにこの宣言が必要です。
ギアドモーターを動作させる（たとえば「mc.rotate(●●;)」の命令を使うときなど）のにも同じ宣言が必要なので、
いずれにせよサンプルプログラム内にはこの宣言が入っている場合が多いです。
・20 行目「ussRead(US1)」
超音波距離センサーが障害物までの距離をはかり、その結果の数値に置きかわります。
「myMatrix.putd2(0, 0, ●●);」という命令は「マトリクス LED に●●という 2 桁の数値を表示する」という意味ですが、
●●の部分に数字ではなく ussRead(US1)を使うことで「距離をはかり、その結果を表示する」という処理になっています。
単に数字を表示するのに使うだけでなく、たとえば「if(ussRead(US1) > 10)」という if 文を作ると「もし障害物までの距離が 10 cmより大きければ～」という条件分岐になります。
カッコ内の「US1」部分を書き換えれば他のコネクタに接続した超音波距離センサーも使えますが、ロボプロシールドを経由してパーツを接続する場合「同時にパーツを繋ぐとうまく動作しないコネクタのペア」が存在するので注意してください。

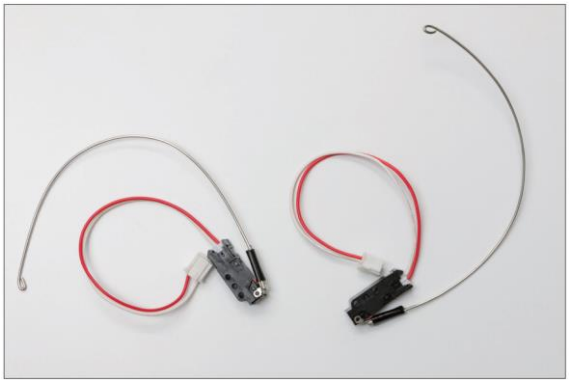
2. タッチセンサーの使い方

【タッチセンサーについて】
タッチセンサーは、「押されている」か「押されていない」どちらの状態かを二択で判定できる小さなスイッチです。
実際に壁・障害物に触れないと反応しませんが、つくりや判定がシンプルで誤動作しにくいのが強みです。

【タッチセンサーの接続と動作確認】
①タッチセンサーのケーブルを、ロボプロシールドの D0 コネクタに接続する。



1 年目キットに含まれている「100 mmビニールチューブ」と「150 mm針金」を使うと、タッチセンサーを触覚のようなイメージで使うこともできます。



② 以下のプログラムを書き込み実行する。
RoboticsProfessorCourse1 ＞ PreCourse ＞ Switch0

書き込みが終わったら、ロボプロシールドの下にあるマイコンボードをのぞきこみながらタッチセンサーを押してみましょう。
タッチセンサーを押している間だけ、マイコンボード上の LED が光れば成功です。

【タッチセンサーのプログラムについて】

```
6  int swPin  = 10;                  //D0にスイッチを接続する
7  int ledPin = 13;                  //本体のLED
8
9  void setup(){
10     pinMode(swPin, INPUT_PULLUP);  //おまじない
11     pinMode(ledPin, OUTPUT);       //おまじない
12 }
13
14 void loop(){
15     if(digitalRead(swPin) == HIGH){ //スイッチが入ったら
16         digitalWrite(ledPin, HIGH); //ledPinに接続されたLEDを光らす
17     }
18     else{
19         digitalWrite(ledPin, LOW);  //ledPinに接続されたLEDを消す
20     }
21 }
```

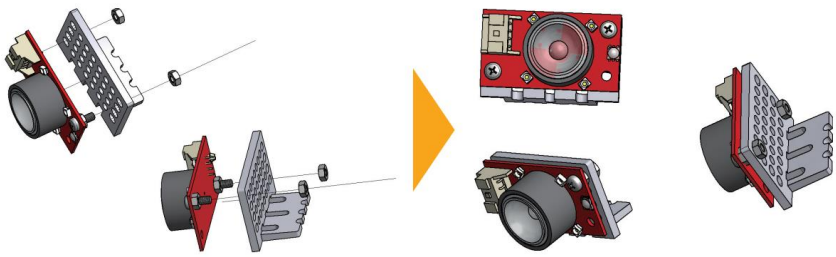
・10 行目「pinMode(swPin, INPUT_PULLUP);」
タッチセンサーを使う場合、void setup()内に必ずこの宣言が必要です。
D0 に接続したタッチセンサーはマイコンボードの「10 番ピン」という部分に繋がるため、初めに「10 番ピンに繋がっているパーツは、信号を受け取るために使うよ」と決めています。「pinMode(10, INPUT_PULLUP);」と書いても同じです。
（「swPin」と書くことができるのは、6 行目で「swPin という変数には 10 という値が入る」と設定しているためです）
D1 にタッチセンサーを接続する場合、D1 コネクタはマイコンボードの「9 番ピン」に繋がっているので、
「pinMode(9, INPUT_PULLUP);」とします。D0/D1 両方にタッチセンサーを繋ぐなら、宣言も 2 行とも必要です。

・15 行目「if(digitalRead(swPin) == HIGH)」
「もしタッチセンサーが押されている（HIGH 状態）なら～」という条件分岐です。こちらも「if(digitalRead(10) ==HIGH)」と書いても構いません。
ちなみに、「タッチセンサーが押されていないなら～」としたい場合は「if(digitalRead(swPin) == LOW)」です。

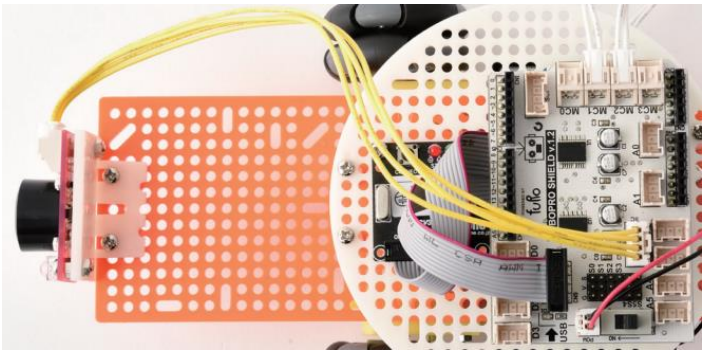
3. カラーセンサーの使い方

【カラーセンサーについて】
カラーセンサーは、その名のとおり色を感知するためのセンサーです。今回の競技では「赤色の床を踏んだら●●する」などといった条件分岐に使われることが多いでしょう。

【カラーセンサーの接続と動作確認】
① ロボットへの取り付け例：カラーセンサーにセンサーL 字ステイを取り付ける。※M3 ナットは計 4 つ。
※ただ動作確認をしたいだけなら、基板へ取り付けず手でセンサーを持っても構いません。



② 黄色のセンサーケーブルを、ロボプロシールドの「IIC」に接続する。



③ ギアドモーターを 2 つ用意し、ロボプロシールドの MC1、MC2 に接続する。

④ 以下のプログラムを書き込み、実行する。
RoboticsProfessorCourse1 ＞ MagicItemB4 ＞ ColorTracer

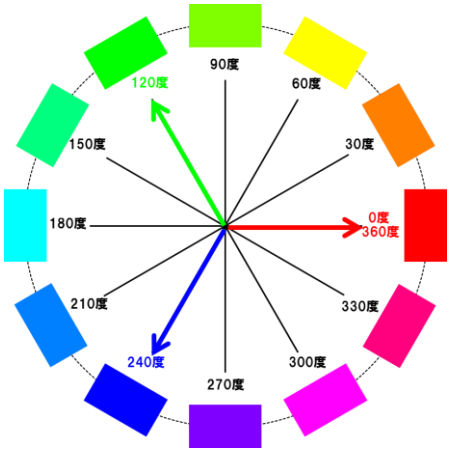
センサーの目の前に青い紙をかざしたときだけ、2 つのモーターが回転したら成功です。
丁度良い距離に紙があるときでないと反応しないので、うまくいかないときはひとまず紙の位置を前後させてみましょう。

【カラーセンサーのプログラムについて】

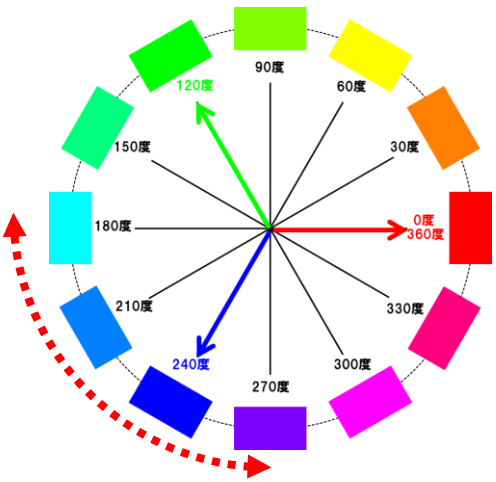
```
7  #include <ColorSensor.h>
8  #include <RPlib.h>
9
10 //おまじない(ピン接続設定)
11 RPmotor mc1(MC1);              //MC1につながっているモーターを指定する
12 RPmotor mc2(MC2);              //MC2につながっているモーターを指定する
13
14 void setup(){
15     ColorSensor.begin(1000, 0); // カラーセンサーの積分時間を指定 x10[ms]   値が大きい
16     ColorSensor.led(3);
17 }
18
19 void loop(){
20     if (ColorSensor.colorRead()){
21         word clear = ColorSensor.colorClear();
22
23         word red   = ColorSensor.colorRed();
24         word green = ColorSensor.colorGreen();
25         word blue  = ColorSensor.colorBlue();
26
27         ColorSensor.convertHSV(red, green, blue);
28         word h = ColorSensor.convertH();
29         word s = ColorSensor.convertS();
30         word v = ColorSensor.convertV();
31
32         if(h > 180 && h < 270){       //カラーセンサーに反応したら前進して近づく
33             mc1.rotate(-50);
34             mc2.rotate(50);
35         }
36         else{                         //カラーセンサーに反応しなかったら何もしない
37             mc1.rotate(0);
38             mc2.rotate(0);
39         }
40     }
41     delay(10);
42 }
```

・7 行目「#include <ColorSensor.h>」
カラーセンサーを使う場合、プログラムの初めにはこの宣言が必要です。
・15 行目「ColorSensor.begin(1000,0);」＆16 行目「ColorSensor.led(3);」
これも、void setup()内に必ず必要な処理です。15 行目の「1000」の部分は、大きくするほど 1 回の感知にかかる時間が長くなりますが、かわりに感知の精度が上がります。16 行目の「3」の部分は、カラーセンサーの周囲にある LED をいくつ点灯させるかの値です。0～7 まで指定でき、数字が大きいほどより多くの LED で目先を照らすことができます。
・20 行目～40 行目の if 文
void loop 内の大半を、青い枠の if 文が占めています。特に前半の青く塗られている部分は、カラーセンサーを使うならばそのままコピペするとよいでしょう。この処理をループの初めに実行することで、ループのたびにカラーセンサーが動作しチェックした色情報を各種変数に入れます。
特に変数「h」に入る値が「目の前の色が何色か」を表しているため重要です。

ロボットに色を判定させるには、「色相環」という考え方を用います。
色の三原色である赤、緑、青の 3 色をベースとして、色を円形に配置して角度で表せるようにしたものです。
赤を 0 度（または 360 度）、緑を 120 度、青を 240 度の場所に配置し、これらの色を混ぜてできる色をさらに 9 色追加し全 12 色とすると下図のような色相環になります。たとえば赤と緑を同量ずつ混ぜると黄色になるため、黄色は 0 度と 120 度のちょうど中間の 60 度に配置されています。この角度の値を「色相」とよび、色を数値で表すのに使います。



このプログラムでは 32 行目に「if(h > 180 && h < 270)」という if 文があります。「h が 180 より大きく、かつ 270 より小さいなら」という意味です。



色相環に照らし合わせてみると、右の図のような範囲になります。
センサーがこの範囲内の色を感知した（つまり、青色周辺だと判断した）ら、MC1＆MC2 のモーターを回転させるという命令になっています。

このように、「●●かつ、△△なら」のように条件を繋げたければ 2 つの式を「&&」で繋ぎます。「●●または、△△なら」としたければ、「&&」のかわりに「||」を使います（SHIFT＋『￥』キーで入力できます）

なお、今回の競技で各種エリアにつかわれる色は、
・分岐エリア：赤色（色相 0 または 360）
・ゴールエリア：緑色（色相 120）
・直線（コース内の 1 か所のみ）：青色（色相 240）
となりますが、実際の会場の照明やセンサーとの距離の影響もあり、色相が上記の値でピッタリ感知できるとは限りません。
当日朝に実際の会場、実際のコースで調整できますがそれほど長い時間は取れないので、色相をすばやく確認し、条件式を書きかえることができるよう、自分なりにプログラムを工夫してきてください。