ロボット博士養成講座

ロボティクスプロフェッサーコース

オムニホイールロボット①

第2回

オムニホイールロボットを操縦する

講師用

目　　次

○　授業開始にあたって

　授業のはじめは、着席させ、大きな声であいさつしてから始めます。

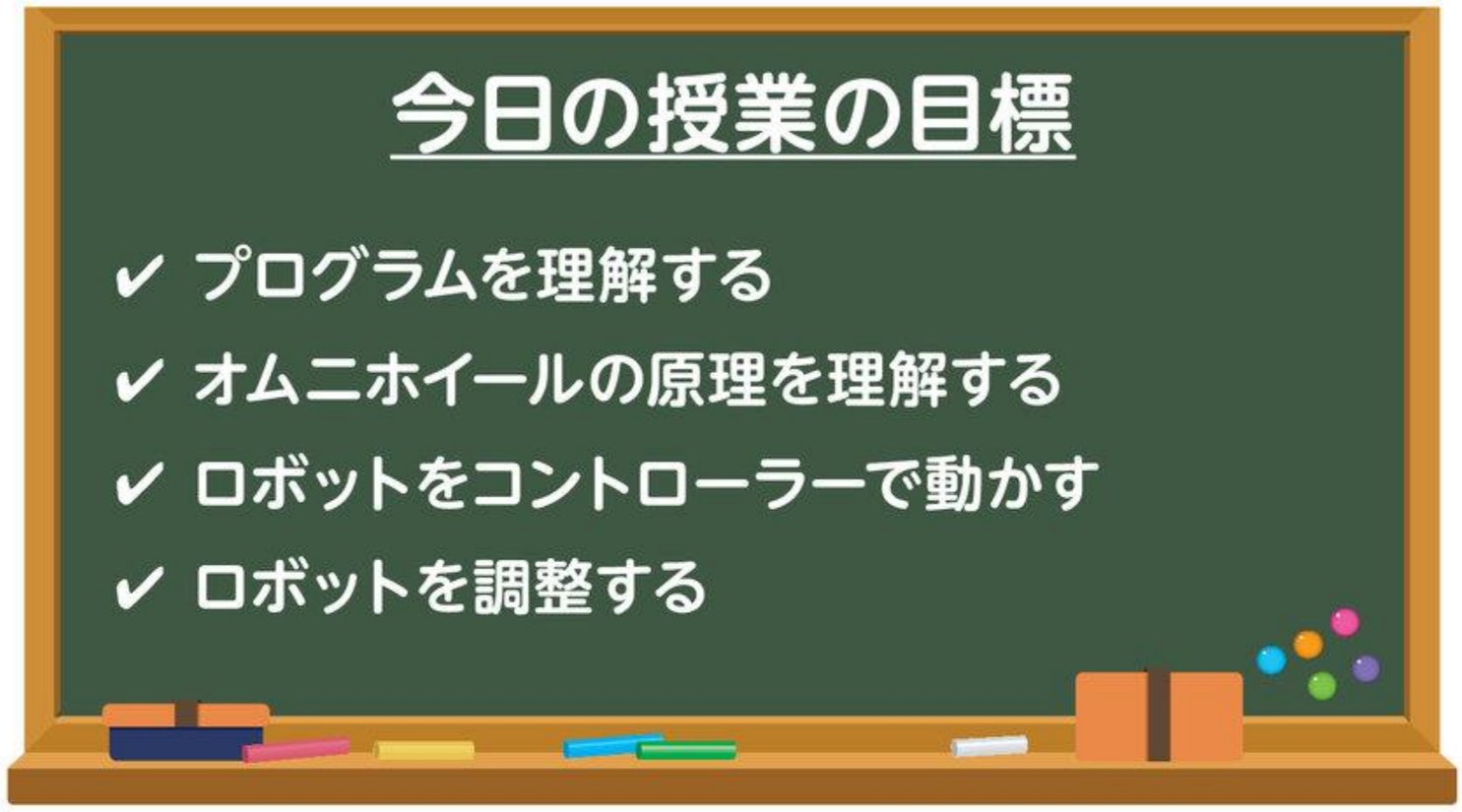
○　今回の目標を黒板に予め書いておきます。

　（授業の目標を明確化することは大変重要なことですので、生徒によく理解させます）

目安時間は授業時間 120 分のうち、休憩 10 分程度を取ることを想定しています。
生徒の進捗状況により、休憩時間などを調整して授業を行ってください。

0. オムニホイールロボットを操縦（そうじゅう）する（目安５分）

0.0. 「オムニホイールロボットを操縦（そうじゅう）する」でやること



オムニホイールロボット、第２回です。
前回の授業では、最後に少しだけプログラムを書き込み、ロボットを動かしてみましたね。
今回はまず、このプログラムのつくりを学習し、思い通りに動き回らせることができるようにしてみましょう！

また、後半ではロボットにさらにパーツを追加し、コントローラーを使ったラジコン操作を楽しんでみましょう！

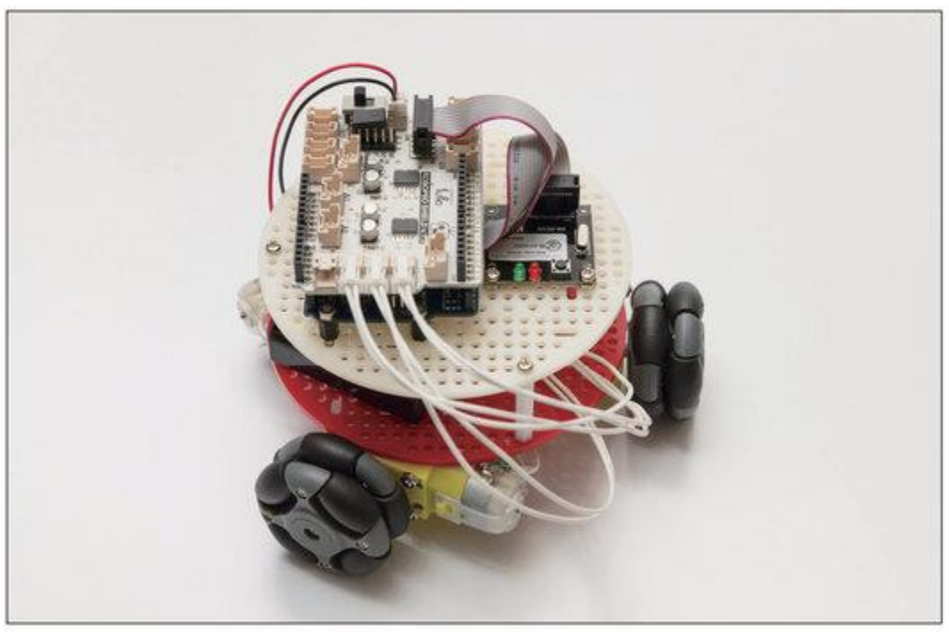
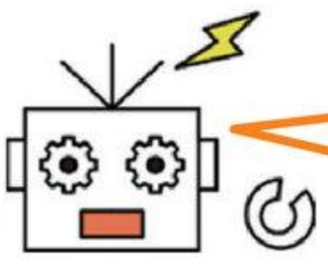


図 0-0　オムニホイールロボット



かっ、勘違いしないでよね！
べつにコントローラーに命令されたから動くわけじゃないんだからネ！

0.1. 必要なもの

前回作ったロボットと、以下のパーツを準備しておきましょう。

USB ケーブル 1	リボンケーブル 1	コントローラー 1
		

図 0-1　次回必要なもの

講

前回製作したオムニホイールロボットの状態（パーツが外れていないか、配線が適切にされているか、など）を確認してからはじめてください。特にギアドモーターの配線がロボプロシールドの MC0、MC1、MC2 のコネクターへ正しく接続されているかは確実に確認させてください。

1．オムニホイールロボットを走らせる（目安30分）

1.0.　モーターに命令を出す

前回の最後にも登場した、モーターの動作確認用のプログラムを再び書き込んでみましょう。

プログラムの書き込み
RoboticsProfessorCourse1 > PreCourse > Motor0

MC0 のモーターだけが回転しますね。
今回は、このプログラムの中身を読みとって改造してみましょう！

プログラム「Motor0」より抜粋

```
//----------------------------------------------------------------------
// MC0のモーターコネクターに接続されたモーターを回すプログラム
//----------------------------------------------------------------------
// Copyright (C) Future Robotics Technology Center All Right Reserved
//----------------------------------------------------------------------
#include <RPlib.h>

// おまじない(ピン接続設定)
RPmotor mc(MC0);      // MC0につながっているモーターを指定する

void setup(){
}

void loop(){
    mc.rotate(100);  // 100の速度で正方向に回す！（最大255）
}
```

プログラムの中で // という記号よりも右側、画面上で文字がグレーになっている部分はプログラムとして実行されない、タイトルや説明書きの部分です。
今回は、以下の部分にのみ注目してみましょう。

```
RPmotor mc(MC0);

(中略)

void loop(){
    mc.rotate(100);
}
```

やってみよう！

プログラム「Motor0」内の黄色の部分を次のように書きかえ、ロボットの動きがどう変わるか確認しよう！
① MC0 を MC1 や MC2 に書きかえる。
② 100 を 50 や 200 に書きかえる。

①の書きかえでは回転するモーターが変わりました。また、②の書きかえではモーターの回転が遅くなったり速くなったりしましたね。
この部分の意味をしっかり理解すれば、思い通りの動きをさせることができそうです。

1.1. プログラムを読みとき、書きかえる

それぞれ、どんな命令だったのか確かめてみましょう。

1）モーターの名前を決める

プログラム「Motor0」より抜粋

```
RPmotor mc(MC0);          // MC0につながっているモーターを指定する
```

まず、上の一行です。 MC0 の部分が「命令するモーターはどれか」を指定しているのはなんとなくわかったのではないでしょうか。
もう少し詳しく見てみます。

命　　令 「RPmotor ●● (△△);」
実行内容：△△につながれているモーターに、「●●」という名前をつける
使 い 方：RPmotor mc(MC0);
　　// MC0 につながれているモーターに、「mc」という名前をつける

プログラミングにかぎった話ではありませんが、何かにお願いや命令をするときにはまず、相手に呼びかけるところからスタートしますよね。つまり「相手の名前」が必要なのです。「鏡よ鏡、世界で一番うつくしいのは誰か教えておくれ……」という命令をしたければ、事前に「お前のことは『鏡』と呼ぶよ」ということをハッキリさせておかなければならないわけです。

RPmotor からはじまる行は、まさにこの「名前を決める」という部分にあたります。
今回は「 MC0 につながっているのは『mc』という名前のモーターだよ！」というのを、ロボットに教えてあげています。このとき MC1 や MC2 のモーターには名前がついていないので、命令を出すこともできなかったわけです。

では、『mc』というステキな名前をつけてもらった MC0 モーターは、どんな命令を出されたのでしょうか。
残りの行を見ていきましょう。

2）モーターを回転させる

プログラム「Motor0」より抜粋（ばっすい）

```
void loop(){
    mc.rotate(100);        // 100の速度で正方向に回す！（最大255）
}
```

void loop() という文がありますね。
プログラムというのは基本的に上から一行ずつ実行していき、一番下まで実行し終えたら終了、という流れになっているのですが、この void loop() 命令を使うと波かっこ（ {} ）に囲まれた部分をずっとくり返すようになります。
つまり、その中にある mc.rotate(100); というのが「ずっと続けたい動き」の命令にあたることがわかります。

命　　令 「●● .rotate(△△);」
実行内容：●●という名前のモーターを、△△の速度で回転させる
使 い 方：mc.rotate(100);　　// モーター「mc」を、速度 100 で回転させる

速度の部分は255までの整数を使うことができます。0を使うと「モーターの回転を止める」という命令になります。
また、数字に「-（マイナス）」をつけて mc.rotate(-100); などとすると、モーターの回転方向が逆になります。

3）プログラムを書きかえる

「名前を決める」「回転させる」という2種類の命令文を学びました。
「モーターに命令を出す」ためには、まず「名前を決める」という処理が必要になるわけですね。
モーターの名前は、自分の好きに決めてしまって構(かま)いません。2文字目からは数字も使えます。ただしひらがなやカタカナ、漢字などは使えません。
名前を決める行を何行も書いて、複数(ふくすう)のモーターに別々の名前を付けていけば、3つのモーターそれぞれに命令を出すこともできます。実際にやってみましょう。

ステップアップ

プログラム「Motor0」を書きかえ、MC0 、MC1 、MC2 のモーターが全て速度200で回転するようにしてみよう！
成功すると、オムニホイールロボットがその場でぐるぐる回るはずだよ！

講

以下のようにプログラムを書きかえます。

```
RPmotor mc0(MC0);
RPmotor mc1(MC1);
RPmotor mc2(MC2);

void setup(){
}

void loop(){
    mc0.rotate(200);
    mc1.rotate(200);
    mc2.rotate(200);
}
```

名前を決める行、モーターの回転命令の行、ともに3行ずつ必要です。
3つのモーターの名前がすべて「mc」だと呼び分けができませんので、「mc0」「mc1」「mc2」と名前を分けています（もちろん、これ以外の名前でも問題ありません）。

チャレンジ課題

さらに「Motor0」を書きかえ、ロボットがまっすぐ前（円形ボードの四角印があるほう）に進むようにしてみよう！

講

MC0 を停止させ、MC1 と MC2 の2つを同じ速さで回転させると前進します。
ただし MC1 の速度にマイナスをつけ、逆回転させてください。
この課題は第3回でより詳しく実施するので、時間が余る場合のみ実施してください。

２．オムニホイールロボットの仕組み（目安 15 分）

2.0.　オムニホイール

オムニホイールロボットはタイヤやモーター、ギアなどが組み合わさって動いています。今から一つひとつ見ていき、全体の仕組みを理解しましょう。

１）オムニホイールの構造（こうぞう）と特徴（とくちょう）

オムニホイールは、外周にタル型のローラーが並んだヘンテコな車輪ですね。このタル型のローラーが自由に回転することで、ふつうの車輪とは異なり、横方向にすべることができます。そしてこのタル型ローラーと車輪全体の回転とのコンビネーションによって、前後左右どの方向へも移動することができます。
しかしそのためには工夫が必要です。オムニホイールは、ふつうの車輪のように車輪全体が回転することで、前後方向への力を得て進むことはできます。しかし、横方向に力を加えても、タル型のローラーが回転してツルツルとすべってしまいます。

図 2-0　オムニホイールの構造（こうぞう）

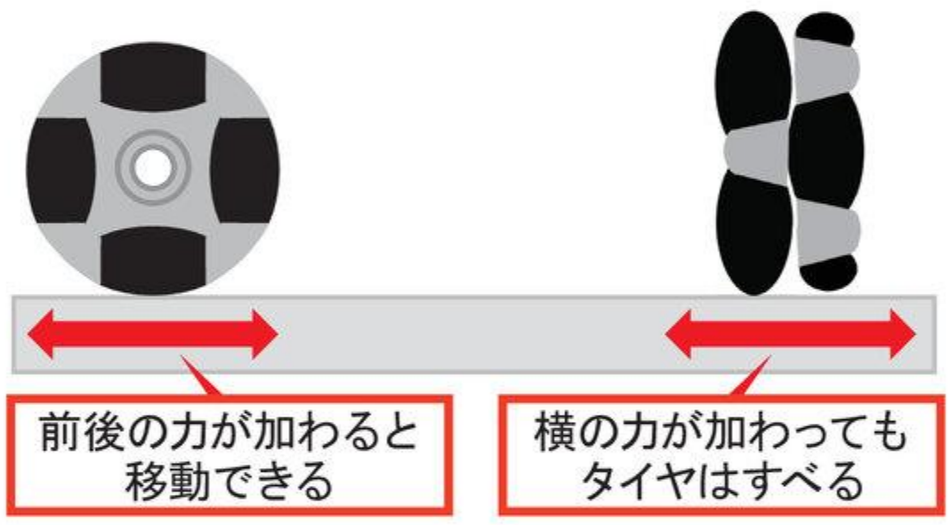


図 2-1　オムニホイールの特徴（とくちょう）

２）オムニホイールの配置

そこで、ホイールの配置が動き方のカギになります。今回作ったロボットにはオムニホイールが 120° 間隔（かんかく）で 3 個ついています。しかし、自動車のように 4 輪が配置されていると、全方向には進めないのです。

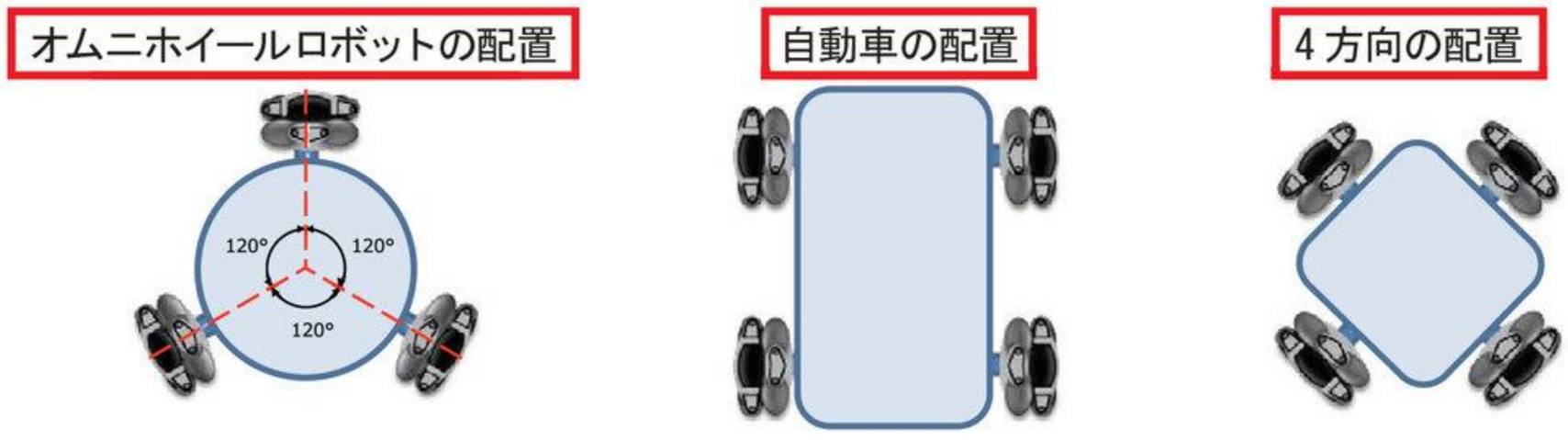


図 2-2　オムニホイールの配置

講　時間に余裕があれば、他にどのような配置がよいか考えさせてみてください。

2.1. モーター

モーターは、電気の力で中心の軸を回転させる機械です。
ロボプロで使われているモーターは「ギアドモーター」といいます。名前の通り、黄色いケースの中には歯車（ギア）が入っていて、モーターと接続されています。

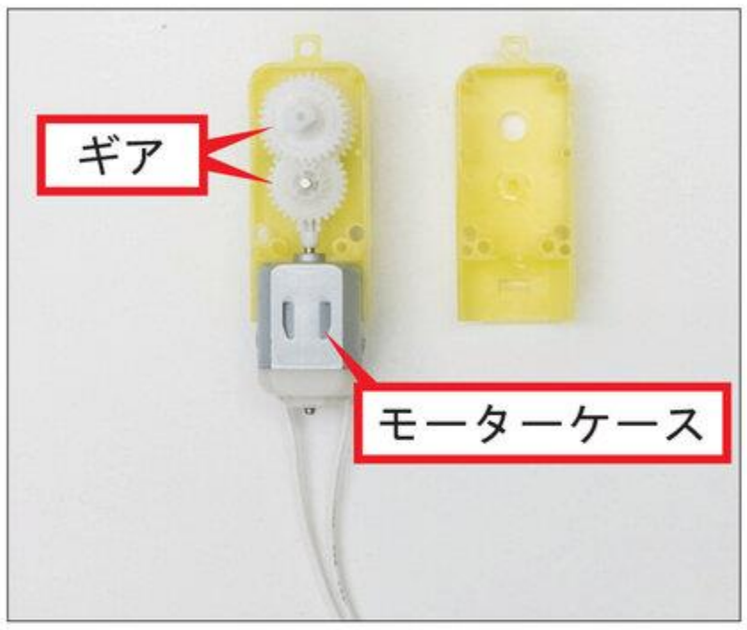


図 2-3　ギアドモーターの中身

もともとこのモーターは速く回転するのは得意なのですが、かわりに回転力が弱いという欠点がありました。歯車をつなぐことで回転速度は落ちますが、回転する力を強めることができます。これによって、モーターよりも大きなオムニホイールを回転させ、地面を走ることができるのです。

POINT

ギアドモーターに回転命令を出しているのに、回転しないことがあります。
これは、モーターの回転力が弱すぎて、ギアを動かすことができていないときに起こります。
その場合はモーターをもっと速く回転させるよう、命令を出しなおしてみましょう。

モーターの回転速度は電気の強さ（電圧の大きさ）で決まるので、モーターにかける電圧を調整することで、思い通りの速さで回転できるようにしています。
オムニホイールロボットの場合、この電圧の調整はマイコンボード上の小型コンピューター（マイクロコンピューター、略してマイコン）が行っています。

コラム　モーターと発電機

モーター（電気モーター）はより正確にいえば「電気のエネルギー」を「ものが動くエネルギー」に変える機械をさします。磁石の近くで導線に電気を流すと、導線を動かすような力が発生します。一般的なモーターは、この力をうまく利用して軸を回転させています。
ちなみに、モーターと似たしくみですがモーターとは正反対に「ものが動くエネルギー」を「電気のエネルギー」に変える「発電機」という機械もあります。ハンドルをぐるぐる回すと点灯する非常時用のライトや、タイヤが回転しているときだけ点灯する自転車用のライトを見たことはありませんか。
家庭で使われる電気の大半は発電所で発電されていますが、やはり同じしくみで生み出されたものがほとんどです。そう考えると、モーターの技術は今の生活には欠かせないものだと感じられますね！

３．コントローラーで操縦（そうじゅう）する（目安 35 分）

さて、モーターの回転をあやつってロボットを動かせるようになりました。
こんどはコントローラーを使って、ロボットを「操縦（そうじゅう）」できるようにしてみましょう。

3.0.　コントローラーとロボットをつなぐ

１）無線受信モジュールの接続

まず、オムニホイールロボットに取り付けておいた無線受信モジュールをロボプロシールドと接続し、コントローラーと通信できるようにします。
接続には、リボンケーブルを使います。無線受信モジュールのさし込み口と、ロボプロシールドの CN9 コネクタにケーブルを取り付けましょう。
リボンケーブルの赤いラインが、無線受信モジュール側は「Vstone」というロゴマークの方を、ロボプロシールド側は「ROBOPRO SHIELD v.1.2」という文字がある方を向くようにします。
図 3-0 の写真を参考にしましょう。

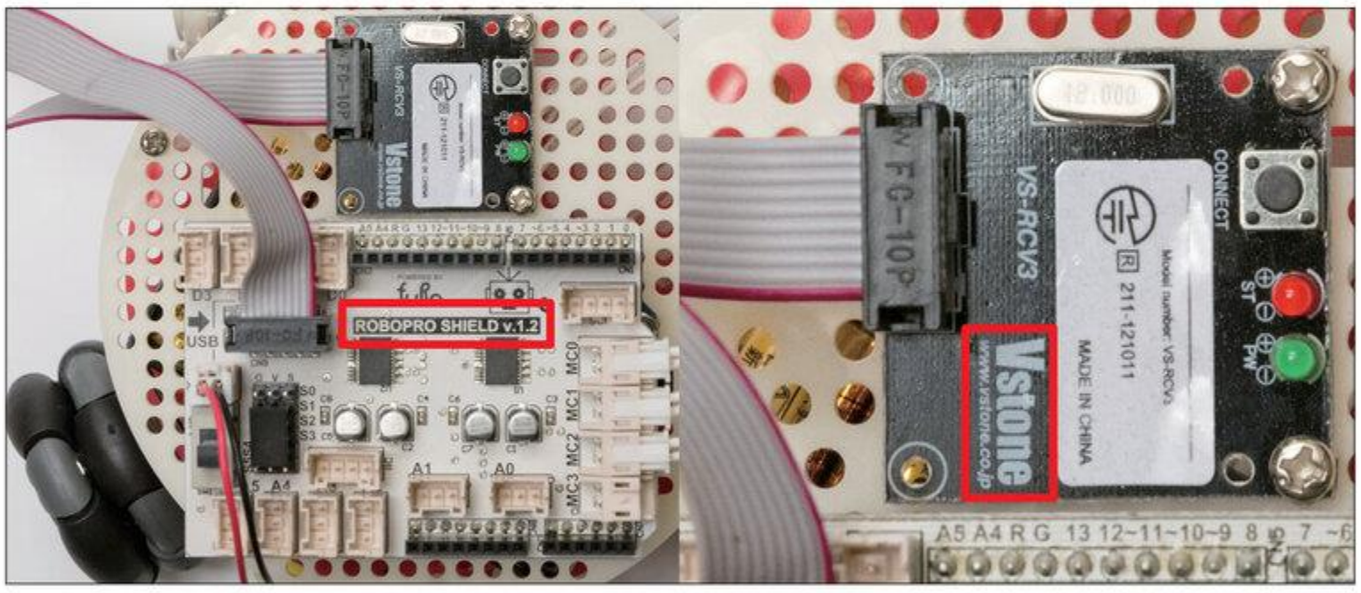


図 3-0　無線受信モジュールの接続

正しく接続されていれば、オムニホイールロボットの電源を入れたとき、無線受信モジュール上の 2 色の LED がつきます。赤色 LED は点滅（てんめつ）（ついたり消えたりする）、緑色 LED は点灯（てんとう）（つきっぱなし）になれば成功です！

講　前のプログラムが残っていると、電源を入れた時に、オムニホイールロボットが高速で動く場合があるので注意するよう促しましょう。

2）プログラムの書き込み

コントローラーで操縦するためには、プログラム内にコントローラーを使うための命令を書く必要があります。
以下のプログラムを書き込んでみましょう。

プログラムの書き込み
RoboticsProfessorCourse1 > OmniWheelRobot2 > Adjust

ただし、プログラムを書き込んだだけではまだ操縦ができません。

3）ペアリング

たくさんのコントローラーとロボットが無線で接続されていると、「目の前のロボットを操縦したいのに、棚の中にしまっていたロボットが動きだしてしまった！」や、「隣の席のクラスメイトがコントローラーを操作したら、自分のロボットが動きだして机から落下してしまった！」といった状況が起こってしまい、危険ですよね。
これを防ぐため、コントローラーは1つの無線受信モジュールにしか信号を送れず、また無線受信モジュールは1つのコントローラーからしか信号を受け取れない、という「ペア」を組ませます。この作業を「ペアリング」といいます。

POINT

●ペアリングのやり方
まず、無線受信モジュールの黒い「CONNECT」ボタンをおします。
次に、コントローラーの裏面の「POWER」を ON にし、表面の「ANALOG」ボタンをおします。
・無線受信モジュールの赤色 LED が、点滅から点灯に変わる
・コントローラーの緑色 LED と赤色 LED が、点滅から点灯に変わる
上記の状態になれば、ペアリング成功です。

●コントローラー使用時の注意点
コントローラーはしばらく放っておくと、自動的に接続が切れて、無線受信モジュールの赤色 LED が点滅に変わり、コントローラーの LED は消灯します。その場合はコントローラーの「START」ボタンをおせば、再びペアリングの状態になります。
※コントローラーの箱に入っている説明書も参考になります。

講　1人ずつ順番にペアリングをしましょう。みんなが同時にペアリングを行うと、混線してしまい、となりの人のロボットと自分のコントローラーがペアになってしまうこともあります。もしそうなったら、手順をはじめからやり直してください。

3.1. ロボットを動かしてみる

これで、ロボットとコントローラーをつなぐことができました。
コントローラーの△ボタン、○ボタン、×ボタンをおすと、ロボットがそれぞれ前、右ななめ前、右に移動していきます。実際に操縦してみましょう。

図 3-1　プログラム「Adjust」の操縦方法

どの方向に動かすときも、複数のモーターが回転していますね！
ということは、先ほどあつかった mc1.rotate(100); などの命令が、モーターの数だけ使われているはずです。

しかし、プログラムを見てみるとどうもおかしいですね。
ためしに、×ボタンをおしてロボットが右に移動するときの命令を見てみましょう。

プログラム「Adjust」より抜粋

```
else if(ps2x.Button(PSB_BLUE)){
    omniBot.move(64, 0, 0);          // 右へ移動
}
```

「ボタンがおされたことを判定する」という命令も書かれていますが、いったん無視します。
今回は「ロボットの動き」を指示している、黄色の部分だけを見てみましょう。
omniBot.move(64, 0, 0); という、見なれない命令が登場していますね。
実は、これも「モーターを回転させる」という命令なのです。ただ、命令の出しかたがさっきの mc1.rotate(100); と少しちがいます。

命　　令　**「omniBot.move(●● , △△ , ××);」**
動作内容：オムニホイールロボットが、指定の向きに移動するようモーターを回転させる
使 い 方：omniBot.move(64, 0, 0);
　　// 64 の速度で右へ移動するようにモーターを回転させる

`mc1.rotate(100);` では「前進させるにはこのモーターをこの向きで回転させて…」と考える必要がありましたが、今回は「どう移動するか」を直接指示できていますね。
●●の部分に横移動の速度を、△△の部分に前後移動の速度を、××の部分に回転速度を入れます。なお、今回の速度は0～100の範囲で入力しましょう。「-（マイナス）」をつければ逆方向に移動します。

やってみよう！

プログラム「Adjust」の `omniBot.move(64, 0, 0);` の数字を好きに書きかえ、×ボタンをおしたときの動きがどう変わるか確かめてみよう！
2か所以上の値を0でない数にすると、動きを組み合わせることもできるよ！

⚠注意！
ロボットが机などから落下しないように気をつけましょう！

講 たとえば前後移動の速度と回転速度をともに正の数にすると、右にカーブしながら進むようになります。

3.2. プログラムの数値を調整する

オムニホイールロボットは組み立てたときのモーターのちょっとした角度のズレや、モーターそのものの回りぐあいのちがいなどが原因で、同じ命令を与えてもロボットごとに少しずつ進み具合がことなります。
「×ボタンをおしたのに右よりちょっと上にずれて進んでいく！」などの状況になったとしたら、この「個体差」が原因です。

人間の手で、個体差がまったく出ないようなロボットを作るのはとても難しいです。そのためオムニホイールロボットは、出た個体差に合わせて調整ができるようなプログラムになっています。

プログラム「Adjust」より抜粋

```
RPomniDirect omniBot(1.0f, 1.0f, 0.9f, 10.0f);   // オムニホイール調整用パラメータ
                     ななめ  前①  前②   横
```

かっこの中の4つの数値を変更すると、同じボタンをおしたときでも移動のしかたが少しずつ変わります。

POINT

・△ボタンをおしたときの動きを調整するとき
ロボットが右にずれていくときは「前①」、左にずれていくときは「前②」の値を、0.1 ずつ小さくしてみる。
変化が大きすぎるときは、値の書きかえを 0.05 ずつに変えてみる。

・○ボタンをおしたときの動きを調整するとき
「ななめ」の値を、ロボットが前よりにずれていくときは大きく、横よりにずれていくときは小さくする。
まずは 0.1 ずつ上げ下げしましょう。

・×ボタンをおしたときの動きを調整するとき
「横」の数値を、ロボットが前よりにずれていくときは大きく、後ろよりにずれていくときは小さくする。
まずは 2.0 ずつ変化させ、変化量が小さいようなら 10.0 ずつの変化に切りかえてみましょう。

少しずつ値を書きかえてはロボットに書き込み、ちょうどよい値を探します。
前、ななめ、横すべて調整できたら、そのときの値をメモしておきましょう。
今後オムニホイールロボットで登場するプログラムには、基本的にこの値を書き込む場所があります。プログラムを新たに書き込む際には、調整値を書きかえてからにしましょう。

やってみよう！

ちょうどよい調整値がみつかったら、下の解答らんにメモしておこう！

omniBot

講 ４つの調整値にはマイナスの数も使えます。特に「横」の調整値は、マイナスも活用して大幅に値を変化させた方がよいケースもあります。

4．さらに細かく操縦（そうじゅう）する（目安15分）

4.0. 操縦（そうじゅう）用プログラム

プログラム「Adjust」ではボタンをおしたらロボットが移動しました。ただ、移動方向がかぎられていて、「操縦（そうじゅう）」とは言えませんでしたね。
今度は、全ての方向に自由に移動させられる操縦（そうじゅう）プログラムを書き込（こ）んでみましょう。調整値を忘れずに書きかえておきましょう。

プログラムの書き込み
RoboticsProfessorCourse1 > OmniWheelRobot2 > Remote1

コントローラーの左アナログスティックを倒すと、同じ方向にオムニホイールロボットが移動します。
興味のある人はプログラムの中身も読んでみましょう。先ほどと同じように omniBot.move(); の命令が使われています。
今回はスティックの倒し具合に応じてロボットの動きを変化させなければならないので、速度をあらかじめ指定しておくことができません。そのため xx や yy といったアルファベットを使っています。

講
左アナログスティックを横に倒していくと xx 、縦に倒していくと yy の部分に入る値が変化していきます。
このように、プログラム内にはアルファベットなどの「文字列」を書いておき、あとで数値に置きかえるテクニックがあります。この文字列を「変数」と呼びますが、詳しくは別の回で解説します。

4.1. 高速モード・低速モードを追加する

さらに、特定のボタンをおしている間だけ動きが速くなったり、おそくなったりするプログラムがあります。

プログラムの書き込み

RoboticsProfessorCourse1 > OmniWheelRobot2 > Remote2

コントローラーの L1 、または R1 ボタンをおしているあいだは「高速モード」になり、ロボットの移動速度が上がります。
また、L2 、または R2 ボタンをおしているあいだは「低速モード」になり、ロボットの移動速度が下がります。

「高速モード」や「低速モード」もすべて omniBot.move(); を使っていますが、使われている数字が少しずつことなります。興味のある人は確認してみたり、数字の部分を別の数字に変えてみたりしてもかまいません。

講

xx / 2 は、変数 xx を 2 で割る、つまり xx ÷ 2 であることを表します。「2」の部分は分母にあたるので、この数値を大きくしていくほど分数全体は小さな値になり、速度は下がります。
omniBot.move(); の速度の最大値は 100 ですが、スティックを最大まで倒すと xx や yy は 128 になります。限界速度を超えてしまわないよう、2 などで割って小さな値に直しているのです。

5．まとめ（目安 5 分）

今回の内容を通して、ホイールやモーターのしくみ、ロボットを動かすときのプログラムの役割を理解できたでしょうか？
プログラムに関しては、便利な命令 `omniBot.move();` を使ってオムニホイールロボットの操縦(そうじゅう)もしました。実際には、3 個のモーターにどのような速度で回転して欲しいかを `omniBot.move();` が、指定しているのです！
次回は、3 個のモーターをどのように動かすと、オムニホイールロボットがどのように動作するのかについて、より具体的に見ていくことにします。

≪次回必要なもの≫
次回は、今回使ったロボットと以下のパーツを持ってきてください。

USB ケーブル　1	コントローラー　1
	

図 5-0　次回必要なもの

講

○　以下の授業の目標を再確認します。
　・プログラムを理解する
　・オムニホイールの原理を理解する
　・ロボットをコントローラーで動かす
　・ロボットを調整する
○　今回の授業で学んだ感想や面白かったことなどを、生徒から聞いてみましょう。
○　次回テーマは「力の合成と並進運動」であることを告知します。