11.1 ロボット実験 1

本実験では、後期実験で用いるロボットの使い方を学ぶ。後期のすべての課題は、ロボットを動作させることが前提である。コンテストに向けて、本稿をよく理解し、応用できる力を身に付けてほしい。

本実験の達成目標を以下に示す。

- ロボットを移動させることができる
- ロボットのユーザボタン、LED、ブザー、超音波センサを動作させることができる
- ロボットから PC へデータを無線通信できる

11.1.1 ロボットの仕様

後期実験で用いるロボットを図 11.1.1 に示す。ロボットは、以下の部品で構成されている。

- Zumo ロボット（Pololu 社 Zumo Robot for Arduino）
  - 左右 2 つのモータを制御するモータドライバ、加速度センサ、地磁気センサ、カラーセンサ、超音波センサ、LED、ブザー、ユーザボタンが搭載されており、Arduino でコントロールする。Zumo ロボットの概要を図 11.1.2 に示す。
- Arduino UNO
  - Zumo ロボットのコントローラとして機能する。ユーザは Arduino にプログラムを書込み、Zumo ロボットを動作させる。
- XBee シールド
  - ロボットに無線通信を行う機能を追加する。

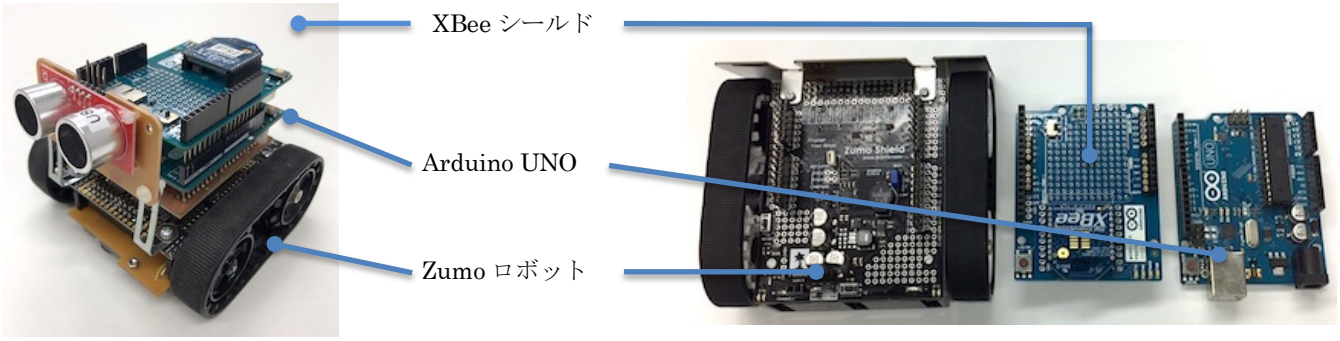

図 11.1.1：ロボット

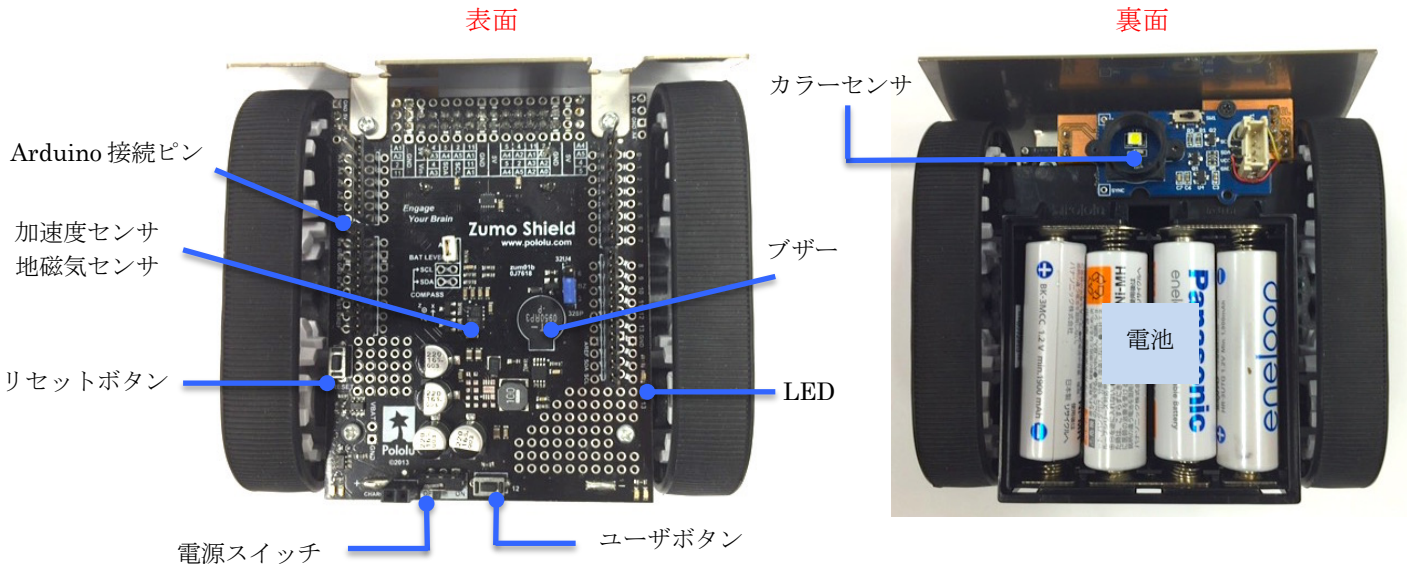

図 11.1.2：Zumo ロボット

11.1.2 Zumo ロボットのピン割り当て

Zumo ロボットは、Arduino の特定のピンを使用する。Zumo ロボットが使用するピンは、プログラマが別の用途に使用することはできない。表 11.1.1 にピン割り当てをまとめる。

表 11.1.1：Zumo ロボットのピン割り当て

ピン番号	機能	関連
0(RX) 1(TX)	無線通信	Serial クラス
3	ブザー	tone 関数、Timer2(MsTimer2)
7, 8	モータの回転方向の制御	ZumoMoters クラス
9, 10	モータの速度制御	ZumoMotors クラス
12	ユーザボタン	digitalRead 関数
13	LED	digitalWrite 関数
A4(SDA) A5(SCL)	カラーセンサ、加速度センサ、地磁気センサ	Wire クラス

例えば、3 番ピンは Zumo ロボットのブザーに割り当てられており、ブザーを tone 関数で鳴動させる処理と、3 番ピンを使う別の処理（3 番ピンに接続した LED を光らせるなど）を同時に実行することはできない。なお、3 番ピンはタイマ割込み（MsTimer2）でも内部的に使用されている[1]ので、ブザーとタイマ割込みを同時に動作させることはできない。使用するピンが競合した場合、動作が不安定になったり、どちらかが機能しなくなることがある。

11.1.3 取り扱いの注意

Zumo ロボットを破損しないこと。高所からの落下や踏みつけ、配線のショートなどに細心の注意を払うこと。Zumo ロボットは非常に高価であり、次年度のシステム実験でも使用する。自己の不注意により破損した場合、弁償の対象となる。特に、次のような事故をおこさないよう注意せよ。

- 高所からの落下。プログラム実行時、突然モータが動作して机から落下しないように気をつけること。
- 踏みつけ。床で動作させる際は、周りに人がいなことを確認すること。埃や水が付着するため、床での動作は推奨しない。
- 回路のショート。基盤のはんだ部分を直接手で触れたり、基盤が抜けかけた状態で動作させないこと。通電する際は、基盤同時がしっかり接続されているか確認すること。
- 基盤を無理に脱着することによるピンの曲がり。特別な理由がない限り、基盤をロボットから抜かないこと。ピンが曲がったり、はんだ付けした箇所が破損する。

[1] MsTimer2 は 3 番と 11 番ピンを使用する

＜演習 11.1.1＞ロボットのモータを回転させて、移動させてみよう。

（1）まず、シリアル通信の設定を確認しておく。XBee シールドの **SERIAL SELECT スイッチ**が USB 側（ロボットの正面側）になっているか確認する。MICRO 側（ロボットの背面側）になっているならば、USB 側に変更する。このとき、ロボットから XBee シールドが抜けないよう気をつけること。

- SERIAL SELECT スイッチによって XBee の接続先を切り替えることができる。スイッチを MICRO 側にすると無線通信が有効になり、USB 側にすると無効となる。Arduino にプログラムを書き込む際には、スイッチを USB 側にしておく必要がある。詳細は、通信実験 1 のテキストを参照せよ。

~~（2）本実験では、ライブラリを用いてモータを動作させる。manaba から ZumoMotors.zip をダウンロード＆展開し、libraries フォルダにコピーせよ。すでに Arduino IDE を起動している場合、Arduino IDE を再起動してライブラリを読み込む必要がある。~~

（3）プログラム 1 を Arduino IDE で作成する（まだ実行しないこと！）

プログラム 1

```
#include <ZumoMotors.h>          //モータライブラリの読み込み

ZumoMotors motors;               //ZumoMotors クラスのインスタンス生成

void setup() {}
void loop() {
  //前進
  motors.setLeftSpeed(100);      //左モータの速度 100（-400～400 に設定可。正なら前転）
  motors.setRightSpeed(100);     //右モータの速度 100（-400～400 に設定可。正なら前転）
  delay(1000);                   //1 秒間前進
  //停止
  motors.setLeftSpeed(0);        //左モータの速度 0（0 なら回転しない）
  motors.setRightSpeed(0);       //右モータの速度 0（0 なら回転しない）
  delay(1000);                   //１秒間停止
  //後進
  motors.setLeftSpeed(-100);     //左モータの速度-100（負なら後転）
  motors.setRightSpeed(-100);    //右モータの速度-100（負なら後転）
  delay(1000);                   //１秒間後進
  //停止
  motors.setLeftSpeed(0);        //左モータの速度 0
  motors.setRightSpeed(0);       //右モータの速度 0
  delay(1000);                   //１秒間停止
}
```

プログラム 1 は、ロボットを 1 秒間前進→ 1 秒間停止→ 1 秒間後進→ 1 秒間停止する動作を繰返す。

ZumoMotors クラスの主なメソッドを以下に示す。

```
void setLeftSpeed(speed)
  Zumo ロボットの左モータの速度を speed に設定する。speed は-400 から 400 までの範囲を整数で指定する。正値のときモータは前転し、負値なら後転する。 0 なら回転しない。
void setRightSpeed(speed)
  Zumo ロボットの右モータの速度を speed に設定する。他は上記と同じ。
```

（4） ロボットと PC を USB ケーブルで接続する。
（5） プログラムを「検証」→「マイコンボードに書き込む」
（6） USB ケーブルを Arduino から抜く。
（7） ロボットの電源スイッチを ON にする。
（8） ロボットが 1 秒間隔で前後に移動することを確認せよ。ロボットが机から落下しないように注意すること！
（9） リセットスイッチを押すと、プログラムが再スタートすることを確認せよ。
（10） ロボットの電源スイッチを OFF にして動作を止める。

＜演習 11.1.2＞ロボットのユーザボタン、LED、ブザーを動作させてみよう。
（1） XBee シールドの SERIAL SELECT スイッチが USB 側になっていることを確かめる。
（2） プログラム 2 を Arduino IDE で作成する（まだ実行しない！）

プログラム 2

```
const int buzzerPin = 3;              //ブザーは 3 番ピン
const int buttonPin = 12;             //ユーザボタンは 12 番ピン
const int ledPin = 13;                //LED は 13 番ピン

void setup() {
  pinMode(buttonPin, INPUT_PULLUP); //ユーザボタンが OFF のとき HIGH, ON のとき LOW になるよう設定
  pinMode(ledPin, OUTPUT);          //13 番ピンを出力モードに設定
}

void loop() {
  while (digitalRead(buttonPin));   //ユーザボタンが ON になるまで待機（無限ループ）
  digitalWrite(ledPin, HIGH);       //LED を点灯
  tone(buzzerPin, 440);             //ブザーを 440Hz で鳴らす
  delay(500);                       //500 ミリ秒間 LED の点灯とブザーの鳴動を継続
  digitalWrite(ledPin, LOW);        //LED を消灯
  noTone(buzzerPin);                //ブザーを止める
}
```

プログラム 2 は、ユーザボタンを押すとブザーが鳴り、LED が光るプログラムである。while ループによって、ユーザボタンが押されるまで待機状態になる。
（3） ロボットと PC を USB ケーブルで接続する。
（4） プログラムを「検証」→「マイコンボードに書き込む」。エラーがあれば修正。
（5） USB ケーブルを Arudino から抜く。
（6） ロボットの電源スイッチを ON にする。
（7） ユーザボタンを押すと、ブザーが鳴り、LED が点灯することを確認せよ。1 回動作すると、while ループによってボタンの入力待ちとなる。再度ユーザボタンを押し、同じ動作を繰返すことを確認せよ。
（8） ロボットの電源スイッチを OFF にする。

ユーザボタンは「基礎実験 2 p.35 演習 2.2.3」で学んだタクトスイッチと同様に、Arduino でプログラミングすることで任意の機能を実装できる（LED も同様）。
ブザーとタイマ割り込み（MsTimer2）は共に Arduino の 3 番ピンを使用する。ブザーを使用するとタイマ割り込み（MsTimer2）が使用できなくなることに注意せよ。

＜演習 11.1.3＞ロボットのユーザボタンを、ライブラリを使って動作させてみよう。

（1）XBee シールドの SERIAL SELECT スイッチが USB 側になっていることを確かめる。

~~（2）manaba から ZumoButton.zip をダウンロード＆展開し、libraries フォルダにコピーせよ。すでに Arduino IDE を起動している場合、 Arduino IDE を再起動してライブラリを読み込む必要がある。~~

（3）プログラム３を Arduino IDE で作成する（まだ実行しない）。

プログラム３

```
#include <Pushbutton.h>              //ボタンライブラリの読み込み

const int buzzerPin = 3;             //ブザーは 3 番ピン
const int ledPin = 13;               //LED は 13 番ピン

Pushbutton button(ZUMO_BUTTON);      //Pushbutton クラスのインスタンスを生成

void setup() {
  pinMode(ledPin, OUTPUT);           //13 番ピンを出力モードに設定
  button.waitForButton();            //ユーザボタンが押されるまで待機
  tone(buzzerPin, 440);              //ブザーを 440Hz で鳴らす
  delay(500);                        //500ms 鳴らす
  noTone(buzzerPin);                 //ブザーを止める
}

void loop() {
  if (button.isPressed() == true) { //ボタンが押されている状態なら
    digitalWrite(ledPin, HIGH);      //LED を点灯
  } else {                           //ボタンが押されていなかったら
    digitalWrite(ledPin, LOW);       //LED を消灯
  }
}
```

プログラム３を実行すると、ユーザボタンが押されるまで待機する。ユーザボタンを押すと 500ms 間ブザーを鳴らして、loop()に入る。loop()内では、ユーザボタンが押されている間、LED が光る（ユーザボタンを離すと LED が消える）。

Pushbutton クラスの主なメソッドを表 11.1.2 以下に示す。

表 11.1.2 Pushbutton クラスの主なメソッド

void waitForButton( )	ユーザボタンが押し込まれた後、離されるまで待機
void waitForPressed( )	ユーザボタンが押し込まれるまで待機
void waitForReleased( )	ユーザボタンが離されるまで待機
boolean isPressed( )	ユーザボタンが押されているなら true，それ以外は false を返す。
boolean getSingleDebouncedPress()	ユーザボタンが押された瞬間の 1 回だけ true，それ以外は false を返す。

getSingleDebouncedPress()が true になるのは、ユーザボタンが OFF から ON になった瞬間の 1 回のみである。よって、ユーザボタンを長押ししても、 1 回だけ true が返ってくる。

isPressded()は、ユーザボタンを押している間も true を返す。短く一度だけ押したつもりでも、loop()の繰り返しによって、何度も true が返ってくることに注意しなければならない。これらは、用途によって使い分けるとよい。

ライブラリを用いることで、ユーザボタンの細かな動きに対する処理を簡単に扱うことができる。以後の実験では、ライブラリを用いても、用いなくても、どちらでもよい。

（4）ロボットと PC を USB ケーブルで接続する。
（5）プログラムを「検証」→「マイコンボードに書き込む」。エラーがあれば修正。
（6）USB ケーブルを Arudino から抜く。
（7）ロボットの電源スイッチを ON にする。
（8）ユーザボタンを押すと、500ms 間ブザーが鳴り、さらにユーザボタンを押し続けている間、LED が光ることを確認せよ。
（9）ロボットの電源スイッチを OFF にする。

＜演習 11.1.4＞ロボットの超音波センサを使ってみよう。
（1）XBee シールドの SERIAL SELECT スイッチが USB 側になっていることを確かめる。
（2）プログラム４を Arudino IDE で作成する（まだ実行しない）。

プログラム４

```
#include <ZumoMotors.h>          //モータライブラリの読み込み

const int trig = 2;              //Trig ピンを Arduino の 2 番ピンに
const int echo = 4;              //Echo ピンを Arduino の 4 番ピンに
unsigned long interval;          //Echo のパルス幅(μs)
int distance;                    //距離(cm)
ZumoMotors motors;               //ZumoMotors クラスのインスタンス生成

void setup() {
  Serial.begin(9600);            //シリアル通信を 9600bps に初期化
  pinMode(trig, OUTPUT);         //trig を出力ポートに設定
  pinMode(echo, INPUT);          //echo を入力ポートに設定
}

void loop() {
  digitalWrite(trig, HIGH);      //10μs のパルスを超音波センサの Trig ピンに出力
  delayMicroseconds(10);
  digitalWrite(trig, LOW);

  interval = pulseIn(echo, HIGH, 23068);     //Echo 信号が HIGH である時間(μs)を計測
  distance = 340 * interval / 10000 / 2;     //距離(cm)に変換
  Serial.println(distance);                  //距離をシリアルモニタに出力

  if (distance < 10) {           //距離が 10cm 未満なら停止
    motors.setLeftSpeed(0);
    motors.setRightSpeed(0);
  } else {                       //距離が 10cm 以上なら前進
    motors.setLeftSpeed(100);
    motors.setRightSpeed(100);
  }

  delay(60);  //trig が HIGH になる間隔を 60ms 以上空ける（超音波センサ HC-SR04 の仕様）
}
```

プログラム４は、ロボットから対象物までの距離が 10cm 以上の間ならば直進し、10cm 未満になると停止するものである。超音波センサの使い方は、センサ実験１で学んだ方法と同じである。

（3）超音波センサのピンと Arduino のピンを表 11.1.3 のとおりにジャンパワイヤ（オス-メス）で接続する。

表 11.1.3 超音波センサと Arduino の接続

超音波センサのピン	Arduino のピン
Vcc	5V
Trig	2
Echo	4
GND	GND

（4）ロボットと PC を USB ケーブルで接続する。
（5）プログラムを「検証」→「マイコンボードに書き込む」。エラーがあれば修正。
（6）ロボットから USB ケーブルを抜く。
（7）ロボットの電源スイッチを ON にする。
（8）ロボットの正面に壁（手やノートなど）を設置して、10cm 手前でモータが停止することを確認せよ。シリアルモニタを開いて、計測距離の確認をすること。
（9）ロボットの電源スイッチを OFF にする。

＜演習 11.1.5＞ロボットを走行させながら、データを PC へ無線通信し、Processing で描画してみよう。XBee による無線通信も、前期の通信実験１で学んだ方法と同様に行える。
（1）XBee シールドの SERIAL SELECT スイッチが USB 側になっていることを確かめる。
（2）プログラム５を Arduino IDE で作成する。

プログラム５

```
#include <ZumoMotors.h>          //モータライブラリの読み込み

ZumoMotors motors;               //ZumoMotors クラスのインスタンス生成

void setup() {
  Serial.begin(9600);
}
void loop() {
  //前進
  Serial.write(0);               //0 を送信
  motors.setLeftSpeed(100);      //左モータの速度 100（-400～400 に設定可。正なら前転）
  motors.setRightSpeed(100);     //右モータの速度 100（-400～400 に設定可。正なら前転）
  delay(1000);                   //1 秒間前進
  //後進
  Serial.write(1);               //1 を送信
  motors.setLeftSpeed(-100);     //左モータの速度-100（負なら後転）
  motors.setRightSpeed(-100);    //右モータの速度-100（負なら後転）
  delay(1000);                   //１秒間後進
}
```

（3） ロボットと PC を USB ケーブルで接続する。
（4） プログラムを「検証」→「マイコンボードに書き込む」。エラーがあれば修正する。
（5） XBee シールドの SERIAL SELECT スイッチを MICRO 側に変更する。
（6） ロボットと PC を接続している USB ケーブルを抜く。
（7） XBee ドングルを PC と接続する。
（8） プログラム 6 を Processing で作成する。シリアルポートのパス（第 2 引数）は適宜変更すること。
（9） Processig で[RUN]ボタンを押す。エラーがあれば修正する。
（10） ロボットの電源スイッチを ON にする。
（11） ロボットの前進・後進に同期して、Processing の表示が GO, BACK に変化することを確認せよ。
（12） Processing で[Stop]ボタンを押す。
（13） ロボットの電源スイッチを OFF にする。

ロボットから PC への無線通信は、前期で学んだ方法と同様である。コンテストでは、ロボットの状態を PC で描画するタスクがあるので、前期の通信実験 1 の資料とプログラム 5 、 6 を参考にするとよい。

プログラム 6

```
import processing.serial.*;

Serial port;
int state;

void setup() {
  size(400, 200);              //幅 400px, 高さ 200px のウインドウを生成
  port = new Serial(this, "/dev/ttyACM0", 9600);  //Serial クラスのインスタンスを生成
}

void draw() {
  background(0);               //背景色を黒に
  textSize(100);               //文字の大きさを 100 に
  textAlign(CENTER, CENTER);   //文字の配置をウインドウの中心に
  if (state == 0) {
    text("GO", 200, 100);      //前進中なら GO と表示
  } else {
    text("BACK", 200, 100);    //後進中なら BACK と表示
  }
}

// シリアルポートにデータが到着するたびに呼び出される割り込み関数
void serialEvent(Serial p) {
  state = p.read();
}
```

＜課題 11.1.1＞次の仕様を満たすプログラムを作成せよ。

- ロボットをその場で「3 秒間時計回り→0.5 秒間停止→ 3 秒間反時計回り→0.5 秒間停止」を繰返す。

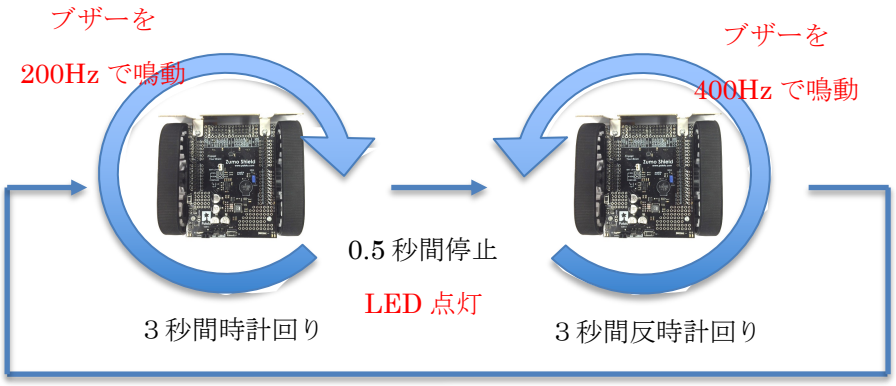

図 11.1.3：ロボットの動作

- 上記 1 セットの動作を繰り返すたびに、時計回りと反時計回りの速度が 50 ずつ増加するようにせよ。初回の速度は 100 とする。ただし、200 を超えたら 100 へ戻せ。つまり、100→150→200→100→...のように回転速度を変えよ。
- 最初にユーザボタンを 1 回押すと動作がスタートするようにせよ（ユーザボタンが押されるまで動かない）。
- ロボットが停止中は LED を光らせよ。時計回り中はブザーを 200Hz で鳴らし、反時計回り中は 400Hz で鳴らせ。

ロボットが机から落下しないよう注意！

＜課題 11.1.2＞次の仕様を満たすプログラムを作成せよ。

- ロボットと壁（手やノートなど）までの距離が常に 10cm に保たれるように移動するロボットを作成せよ。
- つまり、ロボットから壁を遠ざけると近づき、壁を近づけると遠ざかる動きをする。
- 最初にユーザボタンを 1 回押すと動作がスタートするようにせよ（ユーザボタンが押されるまで動かない）
- ロボットから壁までの距離を無線通信し、Processing で表示せよ。表示形式は見やすく理解しやすいものを考えよ。

＜発展課題 11.1.3＞次の仕様を満たすプログラムを作成せよ。

- 四角形、三角形、円を描くようにロボットを移動させ続けよ。
- ユーザボタンを押すたびに、四角形→三角形→円のように動作が切り替えられるようにせよ。

よくある質問を以下にまとめる。

- プログラムが正しいのにエラーがでる
  - ➢ XBee シールドの SERIAL SELECT スイッチが適切に設定されていない。無線通信をする際は MICRO 側に、無効にするには USB 側にする。プログラムを書き込む際も USB 側にしておく必要がある。
- ロボットが移動しない
  - ➢ ロボットの電源スイッチが ON になっているか確認せよ。
- モータの速度を左右同じにしているのにまっすぐ進まない
  - ➢ モータの個体差によるものである。個体差を考慮して左右の速度を設定する必要がある。

【レポート】2019 年 10 月 3 日出題、2019 年 10 月 10 日 12:50 締切

レポート 11.1.1

課題 11.1.1 で作成したプログラム（Arduino）を報告せよ。
プログラムにはコメントを詳細に記述すること。

レポート 11.1.2

課題 11.1.2 で作成したプログラム（Arduino と Processing の両方）を報告せよ。
プログラムにはコメントを詳細に記述すること。
Processing で描画した画面のスクリーンショットも示せ。

発展課題レポート 11.1.3

課題 11.1.3 で作成したプログラムを報告せよ。
図形を描く様にロボットを移動させるには、任意の角度だけ回転させる必要がある。これをどのように実現したか、そのアルゴリズムを解説せよ。
プログラムにはコメントを詳細に記述すること。

使用部品一覧

部品	個数	備考
Zumo ロボット	1	
XBee ドングル	1	
単 3 電池	4	
充電器	1	
Arduino UNO	1	
USB ケーブル	1	

参考図書

[1] Pololu Zumo Chassis User's Guide, https://www.pololu.com/docs/0J54