

第6講 応用編: 赤外線センサとモーターを連携させる

赤外線センサを用いて、センサと物体との距離を測定する。

モーターの回転速度を制御する。

測定した距離に応じてモータの回転数を制御する。

1 赤外線センサを用いて距離を測る

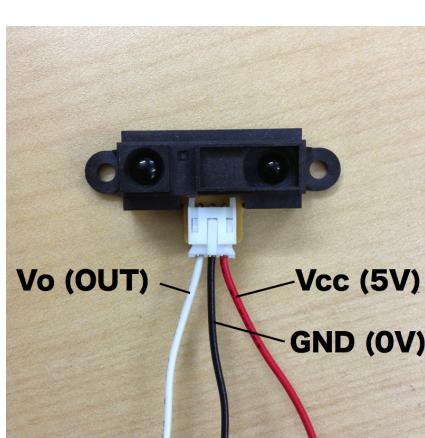
物体との距離を非接触で測定するための方式には「超音波方式」と「赤外線方式」があります。赤外線センサは、(その名の通り)赤外線を用いて距離を測定しています。超音波センサは、物体から人体まで幅広いものの距離を検出することができます。

今回利用する赤外線センサは、大体 10cm ~ 80cm までの距離を測定することができます。赤外線センサは、外光による影響を受けやすいため、使用する環境によっては注意する必要があります。赤外線センサよりも、もう少し長い距離を測定したい場合は、超音波センサを使うと良いでしょう。

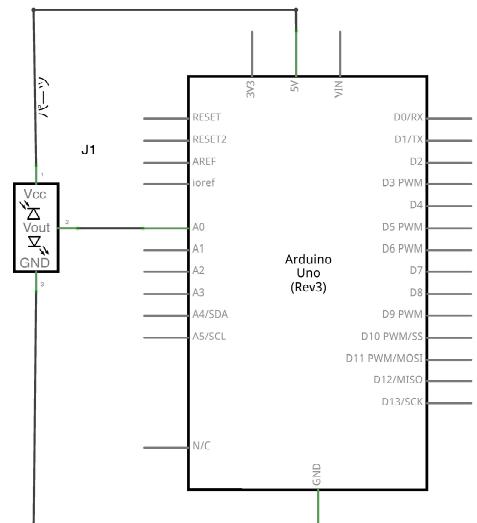
注意!

赤外線センサは、とてもデリケートで配線の向きを間違えただけでも破損してしまいます。
PC に接続する前に配線が正しいかどうか、もう一度確認すること！

回路



赤外線センサ



fritzing

赤外線センサを用いた回路図

プログラム

光センサを圧力センサに置き換えたときのように、プログラムは同じものが使えます。

```
import processing.serial.*;
import cc.arduino.*;

Arduino arduino;
int sensorPin = 0;
PFont font;

void setup() {
    size(300, 300);

    arduino = new Arduino(this, Arduino.list()[0]);
    font = loadFont("Serif-48.vlw"); // Create Fontを忘れずに!
}

void draw() {
    background(255);

    int val = arduino.analogRead(sensorPin);

    textFont(font, 32);
    fill(0);
    text("val:" + val, 32, 32);
}
```

2 センサの値を実際の距離にマッピングする

距離センサと後述する map 関数を組み合わせると、電子ものさしを作ることができます。

準備

距離センサの値を取得して、数値がどのように変化するのか観察してみましょう。また、センサの値が実際の距離とどのように対応付いているのをメモしておいてください。

map 関数

Processing には map 関数という機能が用意されています。

```
float y = map(x, xBegin, xEnd, yBegin, yEnd);
```

map 関数は非常に便利なので覚えておくとよいでしょう。

map 関数は x を範囲 xMin～xMax から別の範囲 yMin～yMax へ変換する関数です。map 関数を利用して、センサの値を実際の距離の計測値に変換してみましょう。

```
void draw() {
    background(255);

    int val = arduino.analogRead(sensorPin);
```

```
// 例、
// 距離が
// 10cmのとき、センサの値が256で
// 80cmのとき、センサの値が768の場合
float distance = map(val, 255, 768, 10, 80);

textFont(font, 32);
fill(0);
text("val:" + val, 32, 32);
text("distance:" + distance, 32, 64);
}
```

3 モーターの回転速度を制御する

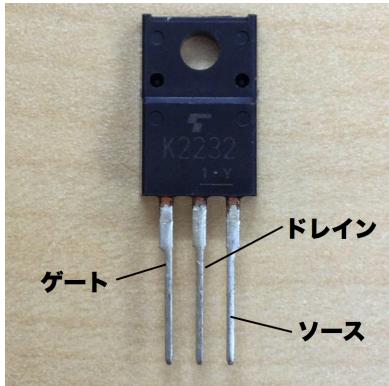
モーターを制御する回路にはいくつか種類がありますが、今回は配線が簡単な電界効果トランジスタ (FET: Field effect transistor) を使用した回路を用います。モーターを回転させるには、比較的大きな電流が必要なため、Arduino の出力だけでは、モータを回転させ続けるパワーが足りません。そのため、外部電源からモーターに電力を供給し、Arduino や Processing で制御するために必要となるのが FET です。FET を用いることで、モーターに流す電流を制御し、モーターの回転数を変化させることができます (電流が多く流れると早く回る)。

用いる部品

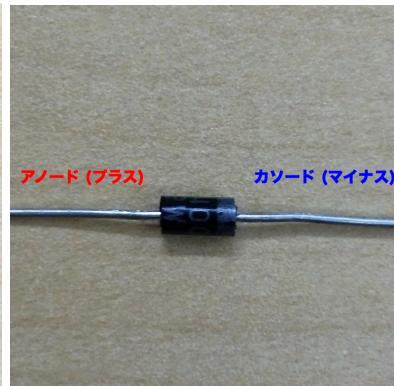
今回は簡単にするために外部電源を使わずにやってみます。各部品が手元にあるか確認してください。



モーター (RE-140RA)

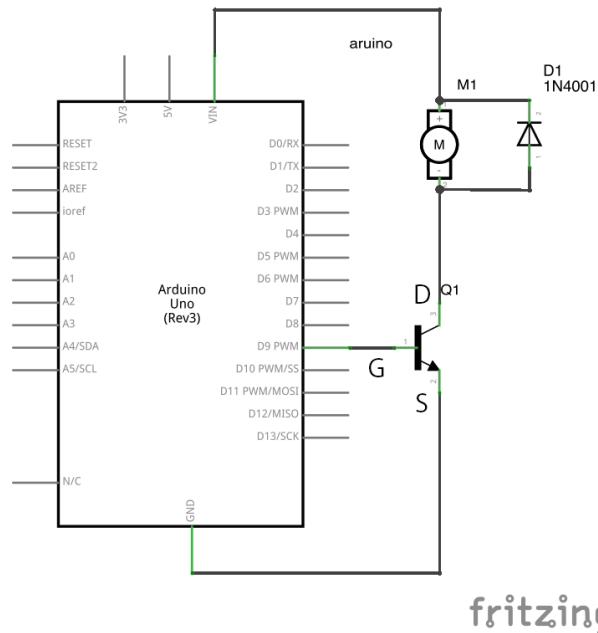


FET (2SK2232)



ダイオード

回路



モーター制御のための回路図

プログラム

```

import processing.serial.*;
import cc.arduino.*;

Arduino arduino;
int motorPin = 9; // モータが接続されたピン

void setup() {
    size(255, 255);
    arduino = new Arduino(this, Arduino.list()[6]);
    arduino.pinMode(motorPin, Arduino.OUTPUT); // モータのピンを出力に
        設定
}

void draw() {
    arduino.analogWrite(motorPin, mouseX); // モータの回転数をセット
}

```

4 赤外線センサからの入力に基づいてモーターを制御する

今回の内容を踏まえて、赤外線センサを用いて取得した距離に基づいて、モーターの回転速度を制御してみよう。

付録

Processing で画像を表示する

Processing では、矩形や円などの単純な図形と同様に画像を表示することができます。画像を表示するにはまず、画像を読み込む必要があります。

準備として、data フォルダに使いたい画像ファイルをコピーしてください。わざわざ data フォルダを開かなくても、Processing のウィンドウにドラッグ&ドロップすると勝手にコピーされます。

```
PImage img;

void setup() {
    size(400, 400);

    img = loadImage('hogehoge.jpg'); // 使いたい画像ファイルの名前を指定
}

void draw() {
    background(255);

    // 画像を表示
    // PImage、x座標、y座標を指定
    image(img, mouseX, mouseY);

    // 幅と高さを指定するとリサイズしてくれる
    image(img, 0, 0, 120, 90);
}
```

Arduino の Input/Output まとめ

Digital Input/Output

```
import processing.serial.*;
import cc.arduino.*;

Arduino arduino;

int inputPin = 0;
int outputPin = 1;

void setup() {
    // COMポートの指定は各々異なるので、
    // デバイスマネージャなどで調べて変更する
    arduino = new Arduino(this, Arduino.list()[0]);

    // inputPinをInputに
    arduino.pinMode(inputPin, Arduino.INPUT);

    // outputPinをOutputに
    arduino.pinMode(outputPin, Arduino.OUTPUT);
}

void draw() {
```

```
// Digital Input
// 戻り値: Arduino.HIGH or Arduino.LOW
int input = arduino.digitalRead(inputPin);

// Digital Output
// outputPinをHIGH(5V)に
arduino.digitalWrite(outputPin, Arduino.HIGH);
// outputPinをLOW(0V)に
arduino.digitalWrite(outputPin, Arduino.LOW);
}
```

Analog Input/Output

```
import processing.serial.*;
import cc.arduino.*;

Arduino arduino;

int inputPin = 0; // A0を使うときは0に
int outputPin = 9; // PWMが使えるピンを選ぶ

void setup() {
    arduino = new Arduino(this, Arduino.list()[0]);

    // Analog Input を使うときは Pin Mode の設定は必要ない
    // arduino.pinMode(inputPin, Arduino.INPUT);

    // outputPinをOutputに
    arduino.pinMode(outputPin, Arduino.OUTPUT);
}

void draw() {
    // Analog Input
    // 戻り値: 0~1023の整数
    int input = arduino.analogRead(inputPin);

    // Analog Output
    // Analogなので、outputする値を0~255で指定
    arduino.analogWrite(outputPin, 0);
    arduino.analogWrite(outputPin, 127);
    arduino.analogWrite(outputPin, 255);
}
```