## 7.2　センサ実験２

前回は超音波センサとサーボモータの動作原理と使い方を学んだ。今回はそれらを統合したアプリケーションシステムを作る。これまでの実験で習得した技術を総合的に応用することが必要となる。

本実験の達成目標を以下に示す。

１週目（前回）：

- 超音波センサの動作原理を説明できる
- 超音波センサを使用できる
- 超音波センサを用いたアプリケーションシステムを作成できる
- サーボモータの動作原理を説明できる
- サーボモータを使用できる
- サーボモータを用いたアプリケーションシステムを作成できる

２週目（今回）：

- 超音波センサとサーボモータを統合したアプリケーションシステムを作成できる

### 7.2.1 システムを設計するときに考えること

Arduino、Processing、素子（LED, 抵抗）、センサ（温度、照度、超音波）、モータを統合して、何らかアプリケーションシステムを作成する際、以降の点について整理した上でプログラミングする必要がある。

### 7.2.2　アナログ／デジタル入出力

電子部品には、アナログ値電圧（0～5V）で動作するものと、デジタル値電圧（5V[HIGH]あるいは0V[LOW]）で動作するものがある。Arduino UNO には、アナログ入力ピン（A0～A5）、デジタル入出力ピン（0～13）、PWM 波を生成する出力ピン[1]（3, 5, 6, 9, 10, 11）がある。アナログ入力ピンはデジタル入出力ピン（A0～A5 が 14～19 に対応）としても利用できる。

システムを設計する際には、使用する電子部品がアナログ値／デジタル値どちらで動作するのか、Arduino のどのピンを使うのかを把握しておく必要がある。アナログ／デジタル入出力の詳細はテキスト「基礎実験３　AD 変換」を参照せよ。

これまでの実験で用いた電子部品について、接続先となる Arduino のピンの種類を表 7.2.1 にまとめる。

表 7.2.1 : 電子部品が使用する Arduino のピン

| 電子部品 | Arduino のピン |
|---|---|
| 照度センサ | アナログ入力 |
| 温度センサ | アナログ入力 |
| 圧電ブザー | PWM 出力 |
| 超音波センサ | デジタル入力、デジタル出力 |
| サーボモータ | PWM 出力 |

[1] ピン番号に~がついている

### 7.2.3　ポーリング／タイマ割り込み／外部割り込み

データ処理を実施するタイミングを制御する方法として、ポーリング、タイマ割り込み、外部割り込みがある。プログラマは、システムの仕様をよく理解した上で、それぞれの方式を適切に選択する必要がある。

表 7.2.2 に、各方式の特徴をまとめる。

表 7.2.2：ポーリングと割り込みの特徴

| 方式 | 長所 | 短所 |
|---|---|---|
| ポーリング | 割り込みを発生できないイベントも監視できる | 正確な時間間隔での処理が保証されない。<br>要求を即座に処理できないことがある。 |
| タイマ割り込み | 正確な時間間隔で要求を処理できる | 頻繁に割り込みが発生すると、割り込みサービスルーチンを呼び出すための無駄な時間（オーバーヘッド）が増える。 |
| 外部割り込み | 不定期に発生する要求を即座に処理できる | |

タイマ割り込みは、Arduino のデジタル 3, 11 番ピンを内部で使用しているので、プログラマはこれらを使ってはいけない（ピンが競合すると動作しなくなったり、不安定になる）。

割り込みの詳細は、テキスト「基礎実験４　割り込み」を参照せよ。

### 7.2.4　シリアル通信

Arduino から Processing へシリアル通信でデータを送信する方式には、Arduino から一定時間毎にデータを送受信する方式（通信方式１）と、Arduino と Processing の間で送受信のタイミングを合わせる方式[2]（通信方式２）がある。

また、1byte の値（0～255）を送受信するのか、1byte 以上の値を送受信するのかで、その方法は異なる。さらに、値を１つだけ送受信するのか、複数を送受信するのかで、方法が異なる。

プログラマは、送受信する値の範囲や数を把握した上で、その方式を適切に選択する必要がある。

Arduino と Processing 間のシリアル通信については、テキスト「可視化実験１」および「可視化実験２」を参照せよ。

[2] ハンドシェイク方式ともいう

## ＜演習 7.2.1＞

超音波センサとサーボモータを統合して動作させる装置を作ろう。
ただし、以下の仕様に従うこと。

**仕様：**

- 対象物までの測定距離が 10cm 未満（<10cm）になったら、サーボモータを 90 度の位置に回転。それ以外のときは、0 度の位置に回転させる。
- 対象物までの測定距離を Processing で可視化する。距離は整数値と長方形の横幅で表現する。10cm 未満（<10cm）になったら、背景色を赤に変える。距離 50cm 以上(>=50cm)は 50 と表示する。

**作成手順：**

1. サーボモータの回転軸に、サーボホーンを 90 度の位置に取り付ける（図 7.2.1）。回転軸が 90 度になっていない場合、前回の演習 7.1.2 を実行し、90 度に回転させた上でサーボホーンを取り付けるとよい。
2. サーボモータのコネクタに、図 7.2.2 のようにシャンパワイヤを接続する。

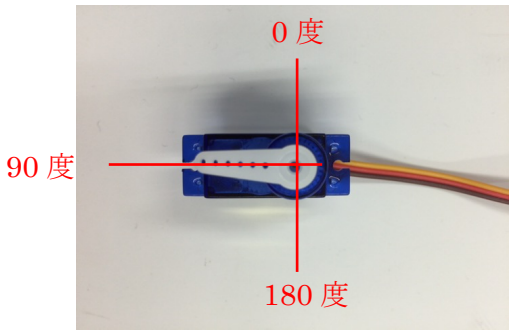


図 7.2.1：サーボホーンの取り付け

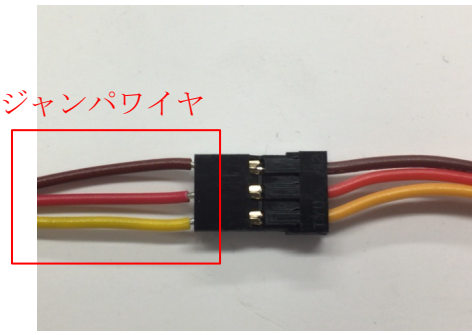


図 7.2.2：コネクタとジャンパワイヤの接続

3. 図 7.2.3 の回路をブレッドボードに組む（Arduino と PC はまだ接続しない）
   - 超音波センサの Trig ピンを Arduino のデジタル出力ピンに接続
   - 超音波センサの Echo ピンを Arduino のデジタル入力ピンに接続
   - 超音波センサの Vcc ピンを Arduino の 5V ピンに接続
   - 超音波センサの GND ピンを Arduino の GND ピンに接続
   - サーボモータの信号線（黄）を Arduino のアナログ出力（PWM）ピンに接続
   - サーボモータの電源線（赤）を Arduino の 5V ピンに接続
   - サーボモータの GND 線（茶）を Arduino の GND ピンに接続

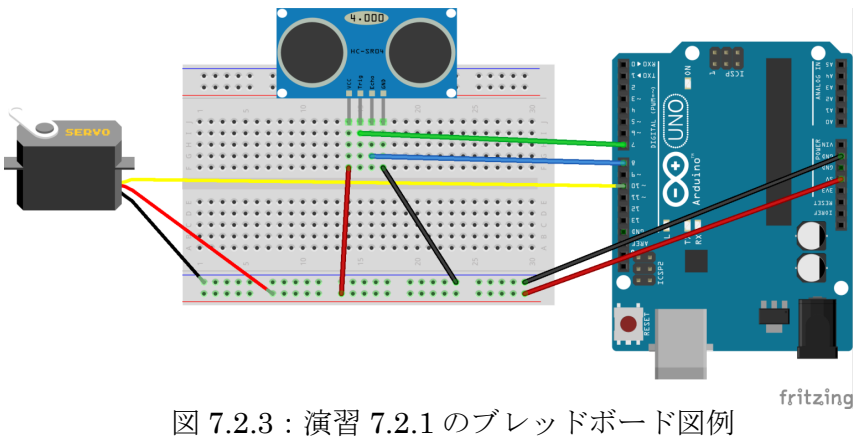


図 7.2.3：演習 7.2.1 のブレッドボード図例

4. 次のプログラム E 7.2.1 を Arduino IDE で作成する。

プログラム E 7.2.1

```
1  #include <Servo.h>
2
3  Servo servo;                    //Servo クラスのインスタンス servo を生成
4  const int trig =  ____;         //Trig ピンに接続する Arduino のピン番号
5  const int echo =  ____;         //Echo ピンに接続する Arduino のピン番号
6  const int servoPin =  ____;     //サーボモータの信号線に接続する Arduino のピン番号
7  unsigned long interval;         //Echo のパルス幅（μs）
8  int distance;                   //距離(cm)
9
10 void setup() {
11   Serial.begin(9600);      //シリアル通信を 9600bps で初期化
12   pinMode(trig, OUTPUT);   //trig を出力ポートに設定
13   pinMode(echo, INPUT);    //echo を入力ポートに設定
14   servo.attach(servoPin); //制御の対象を servoPin に割り当て
15 }
16
17 void loop() {
18   digitalWrite(trig, HIGH); //10μs のパルスを超音波センサの Trig ピンに出力
19   delayMicroseconds(10);
20   digitalWrite(trig, LOW);
21
22   interval = pulseIn(echo, HIGH, 5767);  //Echo 信号が HIGH である時間(μs)を計測
23   distance = 340 * interval / 10000 / 2;  //距離(cm)に変換
24
25   if (0 < distance && distance < 10) {
26     servo.write(90);  //距離が 10cm 未満ならば、サーボモータを 90 度に回転
27   } else {
28     servo.write(0);   //距離が 10cm 以上ならば、サーボモータを 0 度に回転
29   }
30
31   if (0 < distance && distance < 50) {
32     Serial.write(distance);  //距離 50cm 未満ならシリアルポートへ送信 (Processing へ送信)
33   } else {
34     Serial.write(50);        //距離 50cm 以上なら 50 をシリアルポートへ送信
35   }
36
37   delay(60);  //Trig を再度 HIGH にするまでに 60ms 以上の間隔をあける（HC-SR04 の仕様）
38               //この処理をしない場合、動作が不安定になる
}
```

プログラム E7.2.1 の loop()について解説する。

18～20 行目は、超音波センサ（HC-SR04）の Trig ピンに 10μs のパルスを出力している。これによって、超音波センサは送信器から超音波を発射し、対象物に反射して返ってきた超音波を、受信器が受信する。送信から受信までに要した時間（μs）は HIGH レベル信号の長さに対応付けされて Echo ピンから出力される。詳細は、テキスト「センサ実験 1」の図 7.1.4 を参照せよ。

22 行目は、pulseIn()を用いて、Echo 信号が HIGH レベルである時間（μs）を計測している。23 行目は、音速を 340m/s（気温 15℃の時の音速）と仮定し、距離を cm の単位で求めている。

25〜29 行目は、サーボモータを距離に応じて回転させている。
31〜35 行目では、計測値をシリアルポートへ送信している。

5. 次のプログラム E7.2.2 を Processing で作成
6. Arduino と PC を USB ケーブルで接続
7. Arduino IDE で[検証] → [マイコンボードへ書き込む]ボタンを押す
8. Processing で [Run] ボタンを押す
9. 10cm 未満の距離に対象物を置くとサーボモータが 90 度に回転するか？ 10cm 以上の時、0 度に回転するか？ Processing の実行画面で、距離に応じて四角形の幅が伸縮するか確認せよ。
10. Processing で [Stop] ボタンを押す
11. USB ケーブルを抜く。Stop ボタンを押す前にケーブルを抜かないこと！故障の原因になる。

プログラム E 7.2.2

```
1  import processing.serial.*;
2
3  Serial port;
4  int distance;  //距離(cm)
5
6  void setup() {
7    frameRate(60);   //draw()を 1 秒間に 60 回呼び出す（デフォルト 60）
8    size(600, 200);  //幅 600px，高さ 200px のウインドウを生成
9    port = new Serial(this, "/dev/ttyACM0", 9600); //Serial クラスのインスタンスを生成
10 }
11
12 void draw() {
13   if (distance < 10) {
14     background(255, 0, 0);  //距離が 10cm 未満ならば背景を赤で塗りつぶす
15   } else {
16     background(255, 255, 255);        //距離が 10cm 以上ならば背景を白で塗りつぶす
17   }
18
19   translate(0, 100);   //座標軸を右に 0px，下に 100px 移動
20
21   stroke(0, 0, 255);              //図形の枠線の色を青に
22   fill(0, 0, 255);                //図形の内側の色を青に
23   rect(200, -20, distance * 6, 40); //幅 distance * 6px，高さ 40px の四角形を描画
24
25   fill(0);          //テキストの色を黒に
26   textSize(40);     //テキストのサイズを 40pt に
27   text(distance, 100, 20);  //距離 distance をテキストで座標（100，20）に表示
28 }
29
30 //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
31 void serialEvent(Serial p) {
32     distance = p.read();
33 }
```

19 行目では translate()によって座標軸を移動している。translate 関数の書式を以下に示す。

```
translate(x, y)
座標を x px, y px 移動する。
【引数】
x: x 軸方向（右）に移動させるピクセル数
y: y 軸方向（下）に移動させるピクセル数
```

19 行目の translate(0, 100)によって、座標軸は図 7.2.4 のように変わる。

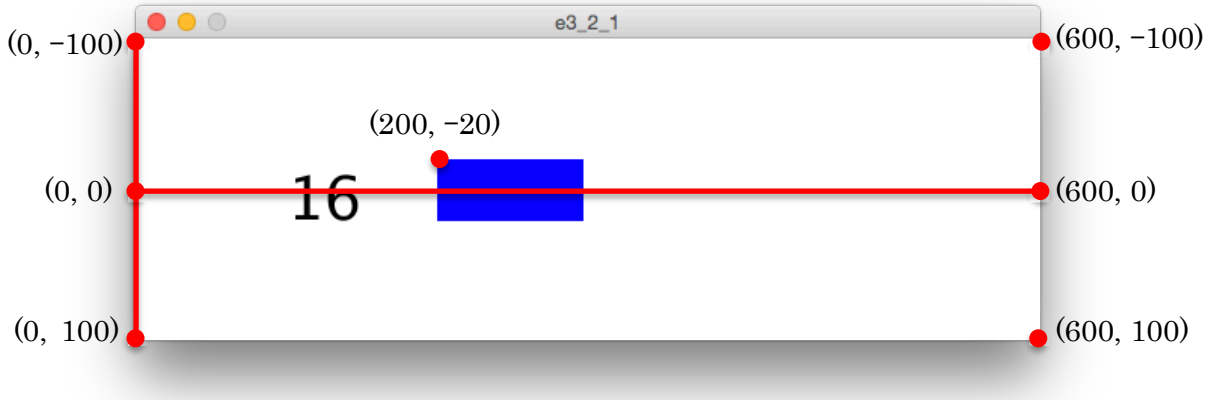


図 7.2.4：座標軸の変換

23 行目では、横幅が測定距離 distance に応じた四角形を描画している。背景が描画される度に、既に s 描画されていた四角形が塗りつぶされて消えるので、四角形の幅が超音波センサの測定距離に応じて伸縮して見えるようになる。

警告：Processing を Run したまま USB ケーブルを抜き差しすると重大な故障の原因となる。
Stop を押してから（Processing の実行画面を閉じてから）、USB ケーブルを抜き差しすること。

**＜課題 7. 2. 1＞**

レーダを作成せよ。ただし、以下の仕様に従うこと。

**仕様：**

- 以下を Arduino で処理する
  - 距離 60cm 以内、角度 180 度の範囲内にある物体までの距離を計測する
  - 距離は、できるだけ正確な時間間隔で計測する
  - 気温を 25℃と仮定して音速を設定する
- 以下を Processing で処理する
  - 計測した物体の位置を 2 次元平面上にプロットする（図 7.2.5）
  - 計測している方向を把握できるように、レーダの向きを描画する（図 7.2.5 の青線）
  - 距離を把握できるように、レーダを中心とする半径 30cm および 60cm の半円と、90 度を示す縦線を常に表示する
  - 0 度～180 度の範囲について物体の位置をプロットし終わったら、すべてのプロットを消し、180 度～0 度の範囲について新たにプロットを開始する。これを繰り返す（図 7.2.6）
- 適当な場所に対象物を置いて、適切に位置が記録されるように工夫せよ

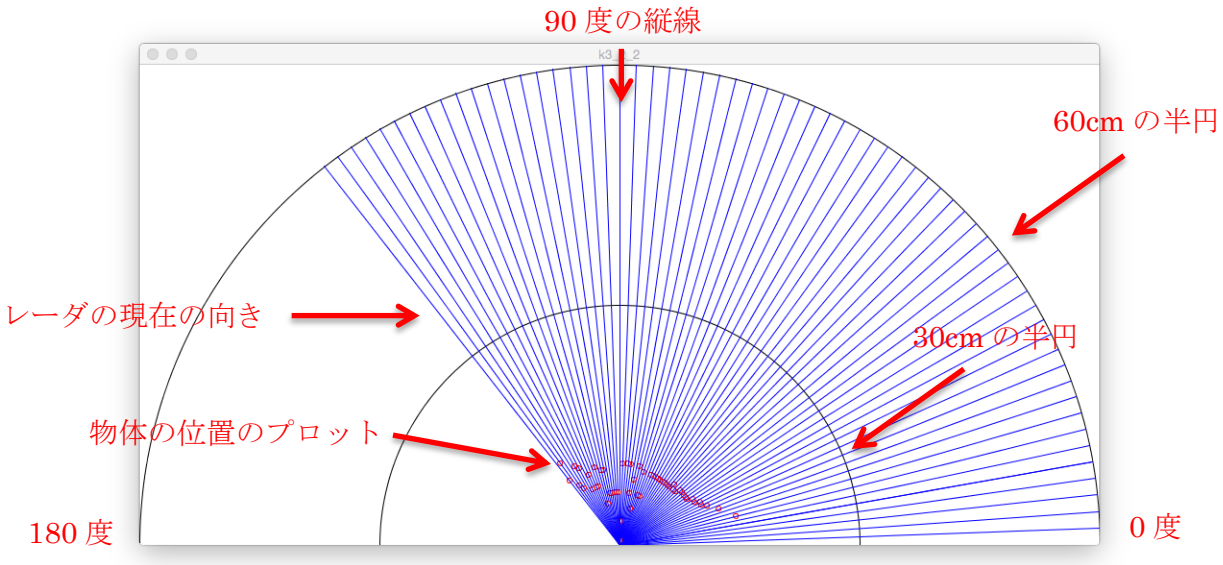


図 7.2.5：レーダによる位置計測

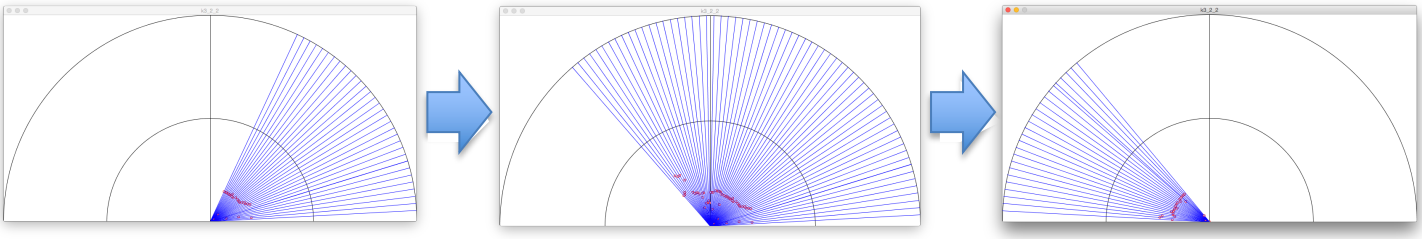

図 7.2.6：0～180 度まで計測したら 180 度から 0 度まで計測

サーボモータに超音波センサを取り付けるには、以下の手順に沿って台座を作成するとよい。

1. スチレン板を 4cm×4cm の正方形に切り取る。これを 4 枚作る（図 7.2.7 (a)）
2. 4 枚をテープで張り合わせて 1 枚の厚板にする（図 7.2.7(b)）
3. サーボホーンを板の側面にテープで貼り付ける（図 7.2.7 (c)）サーボホーンは回転軸から 2 本のホーンが伸びているものが固定しやすい。


図 7.2.7：サーボホーンの取り付け

4. 超音波センサのピンに、ジャンパワイヤ（オス-メス）を接続する（図 7.2.8）。


図 7.2.8：ジャンパワイヤ（オス-メス）の接続

5. 超音波センサと 1 ～ 3 で作成したスチレン板を貼り付ける（図 7.2.9）。ジャンパワイヤのコネクタにテープを貼るとよい。サーボホーンの向きとジャンパワイヤの向きが逆になるようにすること。

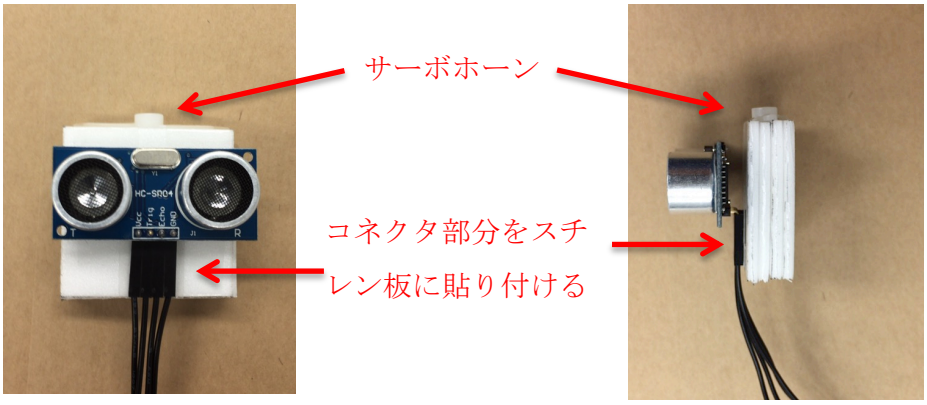


図 7.2.9：超音波センサとスチレン板の取り付け

6. サーボホーンをサーボモータに取り付ける（図 7.2.10）

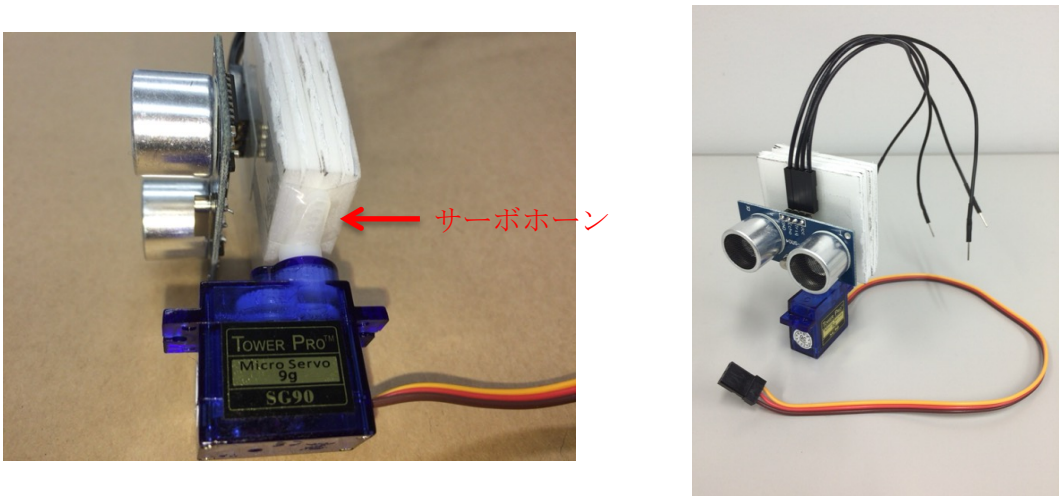


図 7.2.10：サーボホーンの取り付け

7. サーボモータが倒れないように、サーボモータと机をテープで固定する（サーボモータ側面にテープを貼り付けると剥がす際に番号シールも剥がれてしまう。テープを輪にして底面に貼り付けるとよい）。
8. 超音波センサとサーボモータを Arduino の適切なピンに接続する。
9. 仕様を満たす動作をするよう Arduino と Processing でプログラミングせよ。

プログラミングを始める前に、以下の項目について検討し、適切な手段を選択するようにせよ。

- 超音波センサとサーボモータを、Arduino のどのピンに接続するか？
  - アナログ入力／アナログ出力（PWM）／デジタル入力／デジタル出力
    - テキスト「基礎実験２ Arduino 開発環境とディジタル I/O」および本テキストの演習 7.2.1 を参照するとよい。
- 超音波センサによる距離計測を、どのタイミングで行うか？
  - ポーリング／タイマ割り込み
    - テキスト「基礎実験４ 割り込み」を参照せよ。
- Arduino から Processing へ、データをどのようにシリアル通信するか？
  - 通信方式１／通信方式２
  - 1byte 以上／以下
  - 送受信は１つ／複数
    - テキスト「可視化実験１」および「可視化実験２」を参照せよ。

以下については、実際に装置を動作させ、正確に対象物の位置を検出できるよう、最適な値を設定せよ。

- 超音波センサによる距離計測をどれくらいの時間間隔で行うか？
- サーボモータをどれくらいの速さで回転させるか？
- `pulseIn()`のタイムアウト値、`frameRate()`の値、スクリーンのサイズ、プロットの円の大きさ等

距離 d と角度 θ が与えられたとき、直交座標 x, y を求めるには、図 7.2.11 を参考にせよ。

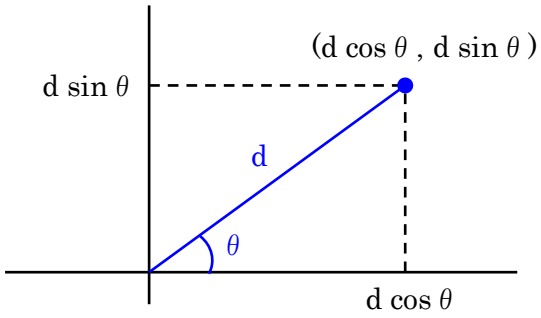


図 7.2.11: 座標の求め方

Processing には、三角関数を求めるためのライブラリ関数が用意されている。

sin(rad)
正弦を求める。
【引数】
rad: 角度。単位はラジアン。
【戻り値】
正弦値（float 型）

cos(rad)
余弦を求める。
【引数】
rad: 角度。単位はラジアン。
【戻り値】
余弦値（float）

角度の単位を「度」からラジアンへ変換するには radians 関数が便利である。

Processing のライブラリ関数については http://tetraleaf.com/p5_reference_alpha/を参照せよ。

radians(deg)
角度の単位を度からラジアンへ変換する。
【引数】
deg: 角度。単位は度。
【戻り値】
角度。単位はラジアン（float）。

**＜課題 7. 2. 2＞**

課題 7.2.1 で作成したレーダは、ノイズによる誤検出が発生することがある（図 7.2.12）。

ノイズを除去する処理を加えたレーダを作成せよ。

ヒント：ノイズと非ノイズの出現位置や分布にどのような違いがあるか？

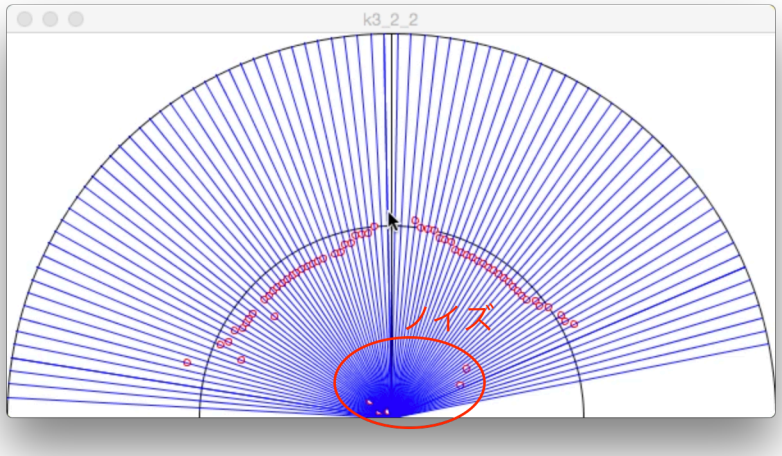


図 7.2.12: ノイズの例

**＜発展課題 7. 2. 3＞**

課題 7.2.2 に、オブジェクトの個数をカウントする機能を追加せよ。

オブジェクトの個数とは、実際に超音波センサの前に置いた物体の数である。例えば、ペットボトルを 3 つ置いていたならば、カウントは 3 である。

0 度～180 度まで計測した後にカウントを表示すればよい（あるいは、0 度～180 度の計測を複数回繰り返した後にカウントを表示してもよい）。

カウントは Processing で描画すること。

ヒント：単一オブジェクト、および、複数オブジェクト間で、計測距離の分布にどのような違いがあるか？

距離が正確に計測できない場合、以下の原因が考えられる。

- 対象物は表面が固く、平面であることが望ましい。布や綿など柔らかく凸凹した物は、超音波を吸収・乱反射してしまい、適切な出力を得られないことがある。
- 他の超音波センサが発射した超音波を、受信機が拾っていることがある。他の超音波センサと距離をあけたり、向きを変えてみるとよい。

**【レポート 7.2（2019 年 7 月 5 日出題、7 月 12 日 12:50 締切）】**

プログラムにはコメントを記述すること。コメントがないものは減点する。

**レポート 7.2.1**

課題 7.2.1 で作成した装置について、以下を報告せよ。

- プログラム（Arduino と Processing）
- 実行結果のスクリーンショット
- 物体の位置を正しく表示できたか？できなかった場合、その理由は何か？どんな改善が考えられるか？について考察せよ。

Ubuntu Linux で「選択中のウインドウのスクリーンショット」をとるには、ウインドウにカーソルをあわせて、Alt + PrintScreen[3]（Alt キーを押しながら PrintScreen キー）を押す。

**レポート 7.2.2**

課題 7.2.2 で作成した装置について、以下を報告せよ。

・プログラム（Arduino と Processing）

・実行結果のスクリーンショット

・どのようなアルゴリズムでノイズを除去したか詳細に解説せよ

・除去できなかったノイズがある場合、なぜ除去できなかったか考察せよ

**発展課題レポート 7.2.3**

発展課題 7.2.3 で作成した装置について、以下を報告せよ。

・プログラム（Arduino と Processing）

・実行結果のスクリーンショット

・どのようなアルゴリズムでオブジェクトをカウントしたか詳細に解説せよ

・正確にカウントできなかった場合、なぜカウントできなかったか考察せよ

**使用部品一覧**

[3] PrtSc, PSc, SRq などとキーボードに表記されていることがある

| 部品 | 個数 | 備考 |
|---|---|---|
| Arduino UNO | 1 | 学生私物 |
| USB ケーブル | 1 | 学生私物 |
| ブレッドボード | 1 | 学生私物 |
| ジャンパワイヤセット | 1 | 学生私物 |
| はさみ | 1 | 7/5 配布&回収 |
| テープ | 1 | 7/5 配布&回収 |
| スチレン板 | 1 | 7/5 配布＆回収 |
| ジャンパワイヤ（オス–メス） | 4 | 7/5 貸出 7/12 回収 |
| 超音波センサ HC-SR04 | 1 | 6/28 貸出 7/12 回収 |
| サーボモータ SG90 | 1 | 6/28 貸出 7/12 回収 |

貸出部品は次週、回収するので、破損や紛失のないように注意して保管・取り扱うこと。

実験により生じたゴミ（剥がしたテープ、スチレン板の小さな断片など）は、各自持ち帰るか、ゴミ箱に捨てること。最寄りのゴミ箱は C10 前にある。

スチレン板の残り、貸し出したテープやハサミなどの工具は、実験終了時に回収するので、持ち帰らないこと。

**参考図書**

[1] システム設計および実験運営委員会:システム設計および実験テキスト,徳島大学工学部知能情報工学科 (2014).