第 4 講 曲げセンサの値を取得する

曲げセンサの値を Arduino を用いて取得する。

1　曲げセンサとは

曲げセンサは、センサ本体を曲げると抵抗値が変化するセンサである。プリント面が外側になるようにして使用する。(反対方向の値が取りたい場合は、2 本組み合わせて使うと良い。)

0 度 ..... 約 10K Ω ↓ (曲げると徐々に抵抗値が増加) ↓ 90 度 ..... 約 3040K Ω

注意!

- 強い力で無理やり折り曲げないように

  薄いフィルム状のセンサであるため、強い力で折りたたんだりすると、癖が付き正しい値が取得できなくなってしまう。

- ブレッドボードに接続するときは必ず根本を持つ

  接続する端子部分は折れやすいので取り扱いに注意すること。

2　回路を組み立てる

回路図

TRY1

回路図を参考にして、曲げセンサと Arduino を接続してみる。必ず、Arduino を PC から取り外した状態で配線すること。

TRY2

Processing の println 文を使って、曲げセンサで取得できる数値を確認する。できなければ前回の講義の資料を参考にしてみよう。使っているセンサが「光」から「曲げ」に変わっただけだ。

3　Processing で画像を表示する

Processing で画像を表示するには、sketch フォルダ (Processing のファイルが入っているフォルダ) 内の data フォルダに使用したい画像をあらかじめ入れておく必要がある。自分でフォルダにコピーしても良いが、メニューバーの Sketch → Add Files... を選択し、使いたいファイル名を選ぶことで、ファイルが data フォルダにコピーされるので覚えておくと便利。

PImage hogeImage; PImage オブジェクトの宣言。void setup() より前の、プログラムの文頭に書く。複数回宣言しておくことで、違う種類の画像をプログラム内で用いることができる。! hogeImage = PImage オブジェクトの名前 loadImage(“image.jpg”); PImage オブジェクトに画像をロードする命令文。void setup() の中に書く。! “image.jpg” = 使用したい画像名 (

.bmp .jpg .png 形式の画像が使用できる。) imageMode(CORNER); 画像を配置する時に基準となる原点を変更する命令文。void setup() 内に書くことが多い。！ CORNER = ！ CORNERS = ！ CENTER 左上が原点 (デフォルト) 左上の x 座標、左上の y 座標、右下の x 座標、右下の y 座標の順に指定= 画像の中心が原点 image(¡¡PImage オブジェクト名¿¿, x, y, wid, hei); アプリケーション画面に画像を表示する命令文。void draw() の中に書く。！ x ！ = ！ 原点の x 座標! ！！ y ！ 2 = ！ 原点の y 座標 2013 年度 春学期授業 インタラクティブ・アート実習 担当:松下 光範! wid ！ = ！ 表示する画像の幅! ！ hei ！ = ！ 表示する画像の高さ

TRY3

幅 640px、高さ 480px の画面に、好きな画像を表示するプログラムを作成する。

Processing のシステム変数を利用する

Processing では、プログラムが実行中に決まる様々なデータを、Processing 自体が宣言した変数の中に収めてプログラム内で使用することができる。これを用いることで、プログラムを簡単に書くことができる。このような Processing があらかじめ用意してくれている変数のことを「システム変数」と呼ぶ。

width！ ウインドウ横幅のピクセル数 height！ ウインドウ縦幅のピクセル数 mouseX！現在のマウスポインタの x 座標 mouseY！現在のマウスポインタの y 座標 pmouseX！前フレームでのマウスポインタの x 座標 pmouseY！前フレームでのマウスポインタの y 座標

TRY4

TRY3 のプログラムを改変して、マウスポインタの位置に画像が表示されるようにする。

¡Hint¿

```
PImage img;
void setup(){
  size(640, 480);
  colorMode(RGB, 256);
  img = loadImage("image001.png");
  imageMode(CENTER);
}

void draw(){
  image(img, mouseX, mouseY, width, height);
}
```

map 関数を利用する曲げセンサの場合、取得できる数値の範囲が狭いため、これをうまく利用するためには、Processing の Calculation 関数の一つの、map 関数を利用すると良い。map(value, low1, high1, low2, high2);！ value! = 変換したい変数の名前 (ex: gainer.analogInput[0]) ！！ low1! ！ = センサで得られる下限値! high1! = センサで得られる上限値! ！ low2! ！ = 変換したい (割り当てたい) 変数の下限値! high2! = 変換したい (割り当てたい) 変数の上限値!

TRY5

TRY4 のプログラムを改造して、曲げセンサを曲げると画像が拡大・縮小するようにする。出来る人は、画像にフェードアウト効果もつけてみよう。

¡Hint¿

```
import processing.gainer.*;
PImage img;
Gainer gainer;

void setup(){
  size(640, 480);
  colorMode(RGB, 256);
  img = loadImage("image001.png");
  imageMode(CENTER);
  gainer = new Gainer(this);
  gainer.beginAnalogInput();
}

void draw(){
  background(255);
  float val = map(gainer.analogInput[0], ???, ???, 0, 1);
  image(img, mouseX, mouseY, ??? * width, ??? * height);
}
```