スマートフォンのセンサを利用したパソコンの遠隔操作

沼津工業高等専門学校
s23131

2023 年 12 月 6 日

概要

マウスが手元にないときに何とかする方法を考案した。AppInventor を用いて Android アプリ開発をし、スマートフォンのセンサーの情報をパソコンに送る方法を用いた。なお、パソコン側は Python で開発した。

1 はじめに

AppInventor とはマサチューセッツ工科大学が開発した Android アプリ作成向けブロックプログラミング言語である。最初から Kotlin や Java などの言語を用いて開発するのはハードルが高かったので、分かりやすいブロックプログラミング言語を選択した。

2 主な仕様

- 方位センサを用いてスマホの傾けた方向・角度を取得し、コンピューターでデータを加工した後その数値をもとにカーソルを移動させる。(傾ければ傾けるほどカーソルの速さが速くなる)
- キーボードによるキー入力も受け付ける。
- スマートフォンの所定の箇所を指で操作することで画面をスクロールすることができる。

3 方位センサを選択した理由

AppInventor では力学的センサのうち、方位センサの他に加速度計 (重力加速度を計算に入れる) と角速度計の値が取得できる。しかし、加速度計は端末の向きによる加速度の補正が困難であること、角速度計はカーソルを動かし続けるためには手首を数回転させないとまともにつかえないため人類には不可能だと判断したので、方位センサを選択した。

4 実装方法

この節ではパソコン側の実装とスマートフォン側の実装方法とで分けて解説する。

4.1 スマートフォン側

重要なところのみ取り上げる。

4.1.1 センサー取得部分

ジャイロセンサーの取得は、Sensors の OrientationSensor を用いて行った。該当部分の主なコードは以下のとおりである。

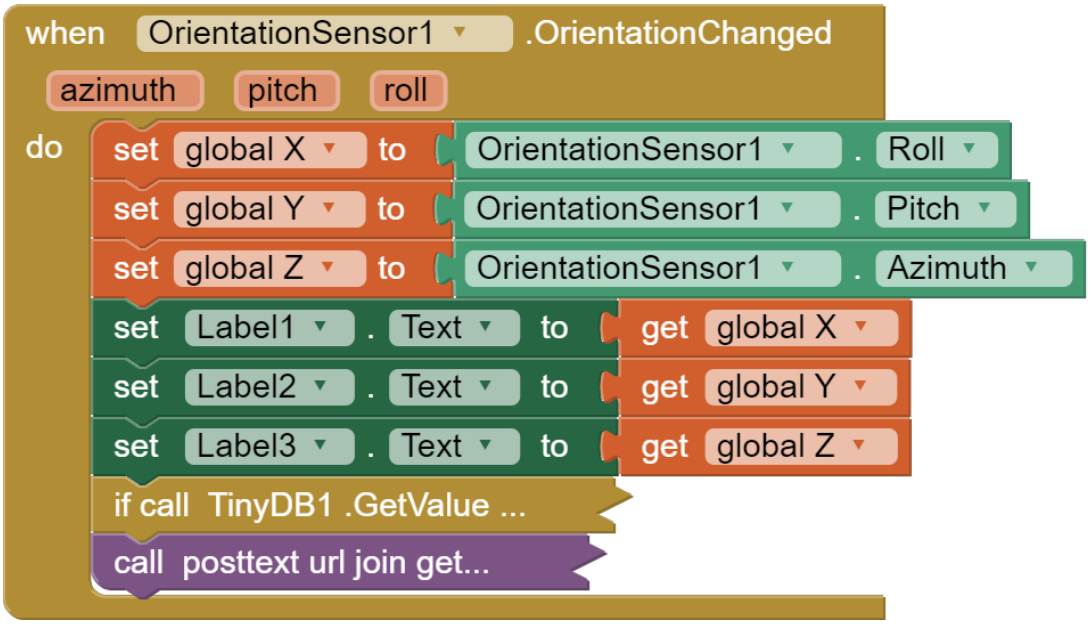


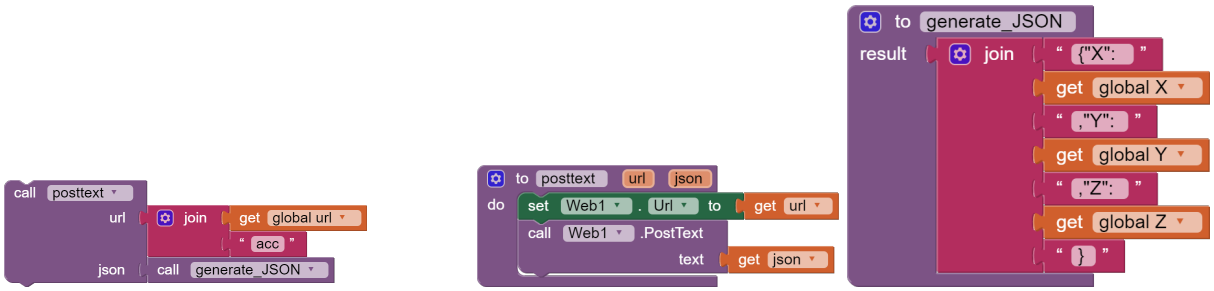
図 1: センサの値の取得

センサの値を取得するコードは、方位センサの値が更新されたとき実行されるようにしている。方位センサーはスマートフォンが静止していても若干変わるためほぼ無限ループであるので仕様上の問題はない。
スクロール実装部分はセンサを用いていないため、詳しくは解説しないが、指でスクロールした分だけパソコンの画面がスクロールされるようになっている。注:set Label ～ の 3 行はデバッグ用コードであるため無くても動く。call posttext ～は情報送信部分の解説で取り上げる。

4.1.2 情報送信部分

情報の送信は Connectivity の Web を用いて HTTP リクエストをコンピュータに向けて送信する。
該当部分の主なコードは以下のとおりである。

図 2: 初期化時の設定



(a) POST 部 **(b)** posttext の中身 **(c)** generate_JSON の中身

図 3: 送信部分

図 1 で URL と RequestHeaders を設定している。図のように配列で指定すればリクエストを送信できるようになる。スクロール量の送信は、最後の行で行っている。なお、図 1 の 2 ブロック目で url タグのデータがない場合、自動で URL を指定しているが、これは URL 設定画面を作ったのでそこを適切な URL に変更してから元の画面に戻れば変更された URL に指定しなおされる。

4.2 パソコン側

Flask を用いて受け取り部分を実装した。必要な設定はすべて割愛し、重要な部分だけ取り上げる。

4.2.1 受け取り部分

ソースコード 1: 受け取り部分

```
1 @app.route("/acc", methods=["POST"])
2 def acc():
3     global scroll_enable_flag
4     set_position(request.json["X"],request.json["Y"])
5     if scroll_enable_flag==1:
6         scroll_(request.json["Z"])
7     return "<p>acc</p>"
```

ここで 5,6 行目はスクロールに関する箇所である。
このコードの他に、マウスの左右ボタンのクリック命令や方向リセット命令 (スマートフォンの方向も受け取る) などが受け取れる設定をしてある。

4.2.2 データ加工部分

ソースコードだと非常に長いので、どのように処理しているか手順を述べる。
データの加工に、リセット時の挙動も深くかかわるのでそれも取り上げる。
カーソルの位置リセット時の挙動は以下のとおりである。

1. スマートフォンの傾きを取得して保存する。(非常に重要)
2. カーソルを中心付近まで移動させる

また、データの加工の手順は以下のとおりである。

1. リセット時に取得した基準の方向からどのくらい傾いているかを計算する。
2. 微妙な傾きを無視するために 1 度未満の傾きを無視する。
3. 1 度以上傾けたとき、急に移動させないようにするため、角度から 1 度を引く。(なお、閾値は変数で管理しているため、1 度以上にすることも可能。)
4. 大きく傾けたら速く、小さく傾けたら遅く移動させるため、移動量の絶対値を修正し、符号を戻す。

3. を図示すると以下のようになる。(x 軸が傾けた量で y 軸が移動量である。)

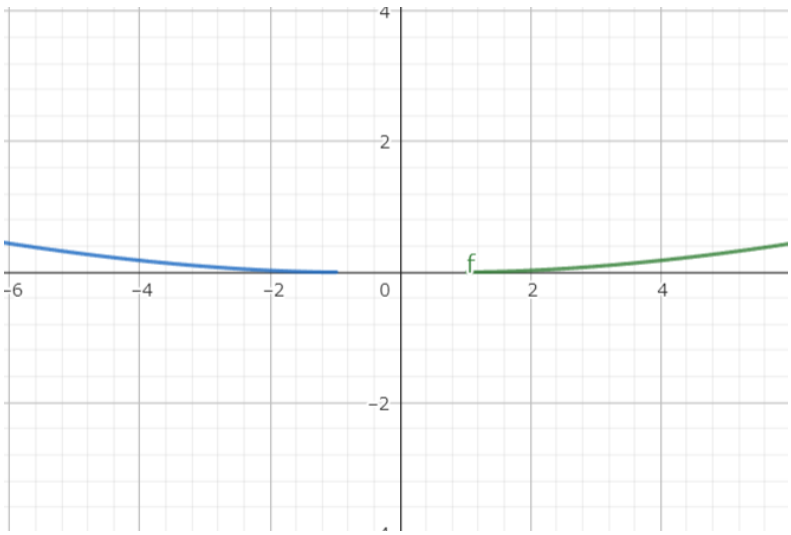


図 4: 手順 3 の図示

このようにすることで閾値を超えた瞬間に急にカーソルが動くということを防いでいる。

4 の「絶対値を修正」で用いた式は以下のとおりである。(3. で出した値を A, 修正後を A' とする)

$$A' = (0.125A)^{1.75}$$

式中の数値は変数によって管理しているため、アプリ側で適切な設定画面と情報受け取り側で適切な動作を実装すればアプリからの変更も可能になる。

5 終わりに

今回は AppInventor を用いたが、Kotlin などのテキスト形式のプログラミング言語で実装し、直線加速度計を用いて実際のマウスのような挙動にできるようにもしていきたい。また、マウスを使わずにセットアップする方法を確立したい。