スマホを使ったIoT防犯カメラの制作

本城　勇岐　　藤原　将也

１．研究概要

Raspberry Piを使用し防犯カメラを制作した。その中でPythonを使用し、カメラで写真を撮影､その撮影した画像をLINEでスマホにデータを送るシステムを作った。これらの活動を通じてRaspberry PiとPythonについての理解を深める。さらにケースを作り、外に置けるようにする。

２．研究の具体的内容

（1）使用したもの

- Raspberry Pi 3 model B
- OpenCV
- LINE Notify
- ディスプレイ
- Webカメラ
- 焦電型赤外線センサーモジュール
- 段ボール

（2）動作の流れ

撮影から通知までの流れを図１に示す。

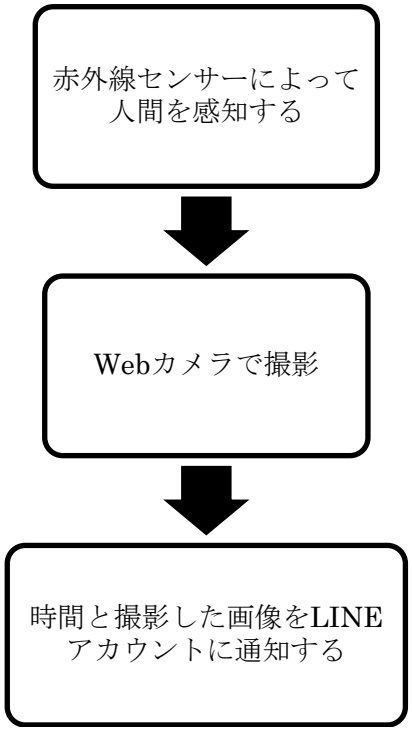

図１　動作の流れ

（3）制作環境の準備

監視カメラを制作するにあたり､一から考え作成するには期間が短いと考え、“そぞら”氏の｢[ラズベリーパイ]監視カメラの作り方｣を参考にしました。

プログラムを始める前に、カメラで撮影した写真の入出力やトリミングやリサイズなどの処理を行うために､OpenCVをインストールした。その次に、数値計算を行うnumpyのインストールとパッケージリストの更新を行った。

```
sudo pip3 install opencv-python=4.5.1.48
```

```
pip install –U numpy
```

```
sudo apt update
```

```
sudo apt install libatlas3-base
```

（4）カメラ映像の表示

プログラムによる動作や内容を理解するために、手始めにカメラの映像をそのまま表示するだけのプログラムを作成した（図２）。

動作としては一コマ分のカメラ画像を読み込み、表示する。これをWhile文によって繰り返すようにしている。

途中で動作を終わらせられるように、Esc キーを入力すると終了するようになっている。

```
1       import cv2                      #openCVを使えるようにする
2
3       camera=cv2.VideoCapture(0)      #カメラチャンネルを指定する
4
5       While True:                     #繰り返し処理開始
6         ret,frame=camera.read()       #1コマ分のカメラ画像を読み込む
7         if not ret:
8           break
9
10      cv2.imshow("Frame",frame)       #カメラ画像を表示する
11      key=cv2.waitkey(1)
12
13                                      # Escキーを入力されたら画面を閉じる
14      if key==27:
15        break
16
17      camera.release()
18      cv2.destroyAllwindows()         #画像表示の終了
```

図２　映像表示のプログラム

（５）画像の保存

次に画像保存のプログラムを作成した（図３）。ファイル名を固定して保存するだけだと、元の画像が上書きされてしまいデータが残らない。そのため、データ名を撮影日時にすることで、個別に保存できるうえに、赤外線センサーが感知した時、要するに侵入者が来た時刻を記録することができる。

```
1       import cv2
2       import datetime
3
4       dt_now=datetime.datetime.now()    #現在の時刻を取得
5       file_name=dt_now.strftime('%Y年%m月%d日%H時%M分%S秒')
6
7       cap=cv.2VideoCaputure(0)
8       ret,frame=cap.read()
9       cv2.imwrite(file_name+'.jpg',frame)  #.jpgファイルで保存する
10      cap.release()
```

図３　保存のプログラム

（６）人感センサーの応用

人感センサーの検出範囲が広すぎたり、感度が高すぎたりすると関係ないところで反応したり、うまく動作しなかったりすることが多々あった。そのために、センサーの足にある調整ボリュームで細かく調整した（図４）。

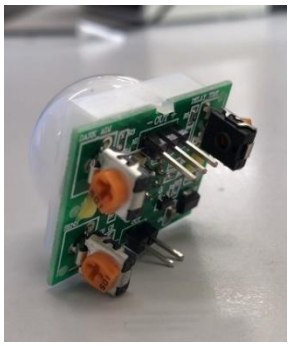

図４　赤外線センサーの調節ネジ

（７）人感センサーによる撮影と通知

これまでのプログラムを応用し、センサーに反応すると同時に撮影､そしてLINEによって通知するプログラムを制作した（図５）。

参考元のプログラムだと、人を感知している間、連続的に撮影してしまうため、感知した時と感知しなくなった時だけ撮影するようにした。

```
import RPi.GPIO as GPIO
import cv2
import time
import datetime
import requests

#LINEメッセージ送信の関数
saved_sense=GPIO.LOW
def send_message(Discovery_time):
    url = "https://notify-api.line.me/api/notify"
    token = "トークンをここに入力"
    headers = {"Authorization" : "Bearer "+ token}
    files = {'imageFile': open("image.jpg", "rb")}
    message = "{Discovery_time,"侵入者あり")
    payload = {"message" :  message}
    r = requests.post(url, headers = headers, params=payload, files=files)

#センサーを使う準備
GPIO_PIN = 18

GPIO.setmode(GPIO.BCM)
GPIO.setup(GPIO_PIN,GPIO.IN)

while True:
    sense=GPIO.input(GPIO_PIN)
    if saved_sense=sense;
        if sense==GPIO.HIGH;
        print("1")

        #検出時間の取得
        dt_now = datetime.datetime.now()
        Discovery_time = dt_now.strftime('%Y年%m月%d日%H時%M分%S秒')
        print(Discovery_time)

        #カメラ画像を保存する
        cap = cv2.VideoCapture(0)
        ret, frame = cap.read()
        cv2.imwrite("image.jpg", frame)
        cap.release()

        #LINEメッセージ送信
        send_message(Discovery_time)

        #10秒待機
        time.sleep( 10 )
        saved_sense=sense

    else:
        #センサー未検出時の処理
        print("0")
        saved_sense=sense
        time.sleep( 1 )

else:
 time.sleep(1)
```

図５　撮影と通知のプログラム

（８）自動起動について

通常のままだと、ラズベリーパイを起動した後、プログラムを開き、手動で実行をしなければいけない。そのため、systemd を使って指定したプログラムを起動時に実行できるようにした。起動するプログラムの概要とどこに保存されているかの設定を新しいファイルに保存した。

sudo systemctl start line.service

#自動起動のシステムのスタートをかけることができる（start を stop に変えると自動起動を停止することができる。

ps –ef | grep python3

#現在の自動起動のシステムが動いているか、状態を見ることができる。

（９）収納箱の作成

防犯カメラとして作成するにあたって、家の中ではなく、外に置くことを想定した。そこで、ラズベリーパイやカメラなどの部品を外にさらすと問題が生じると考えた。防犯カメラと気づかれにくいように、段ボールで収納箱を作った。メンテナンス性の確保のために箱の上面を開閉できるようにした。

上の段に USB カメラと赤外線センサーを配置し、下段にラズベリーパイを配置した。

図６　収納箱の正面

図７　収納箱の内側

（１０）完成（岡工祭での展示）

撮影した写真に検出日時を添付し,LINE で通知する（図８）という当初の目的を達成することができた。完成品を岡工祭で展示し、実際に防犯カメラとして稼働させた。大きな失敗はなく、やり遂げることができたが、細かな問題点も見つかった。

図８　実際の通知画面

（１１）問題点

岡工祭で実際に稼働させてみた結果、赤外線センサーの感度に問題があり、検出が遅れてうまくいかないことがあった。試験段階では、検出の範囲と感度の両方とも良いバランスで調整できていた。しかし、実際に稼働させてみると、人が複数人出入りしたり、留まったりすることで感知にずれが見えた。

検出の範囲、感度をより細かく計測し、防犯カメラを設置する場所と場面に適正な設定を考える必要がある。

３．研究のまとめ

今回製作した防犯カメラは概ね納得のいくものとなった。また、プログラムをする中で困難なこともあったが、横山先生を含めた三人で協力していくことによって乗り越えることができた。それと同時にラズベリーパイや Python について実際に手を動かし、考えることで深く学ぶことができた。

また、年間を通して活動することの難しさを感じた。最初に立てた年間計画（図１０）を考えると、実際は全く余裕のない日程となってしまった。二人の作業の分担や細かなノルマの設定を丁寧にするべきだった。

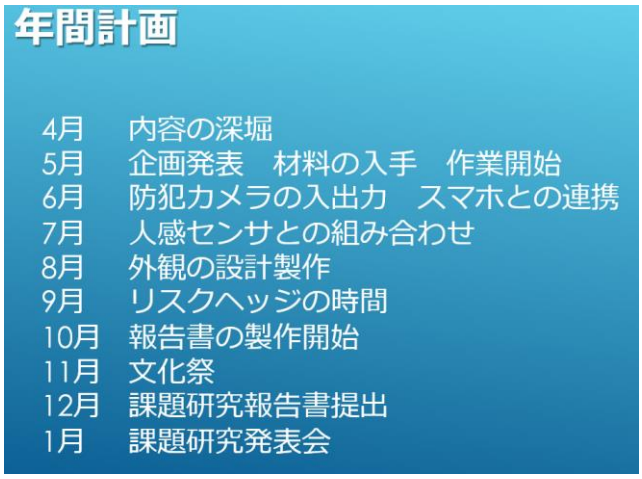

図１０　年間計画

感想（藤原）

今回、ラズベリーパイや Python に触れることができたのはとても貴重な経験になった。ラズベリーパイには多くの可能性があると考えており、これから先の情報分野でも広く活用されるだろう。Python も同様に広く利用されている。大学入学後も使用すると思うので、今回の経験を生かしていきたい。

感想（本城）

今回、課題研究を通してラズベリーパイと Python を勉強出来てよかった。今回の作業だと、写真を撮り LINE に送るプログラムはあまり苦労せず作れたが、このプログラムを自動で起動させるシステムを作るのにとても苦労した。しかし先生に協力してもらい無事完成した。このことから自分たちの経験不足を実感出来た。社会人になると Python を使う機会が増えると思うので、たくさん経験を積んでいきたいと思った。

参考文献

[ラズベリーパイ]監視カメラの作り方（sozorablog）
https://sozorablog.com/camera_shooting/

Raspberry Pi4 起動時に指定したプログラムを実行させる（パソコン工房 NEXMAG）
https://www.pc-koubou.jp/magazine/52061