

## 2 LINE の Messaging API を用いた防犯システムの作成

佐々木 理音

指導教員 石館勝好

### 1. はじめに

今、この時代は SNS が大きな役割を果たしている。私も SNS を利用しているが、去年の夏に無防備な農家の畑に除草剤をまかれたニュースを聞いて、手軽にかつ低コストで今主流の SNS である LINE を用いた防犯システムを作成できないものかと思い、このテーマを考えた。

### 2. システム概要

システム構成図を図 1 に示す。

Raspberry Pi とカメラモジュール、人感センサーを用いて防犯カメラシステムを作成する。そのカメラで入手したデータをもとに LINE BOT からユーザーに通知する。この際、カメラは屋内または玄関など人が出入りする場所に設置し、侵入者を検知したら撮影し、LINE に送信する。

このシステムを作成することにより、無防備な場所に侵入した犯人を特定しやすくなり、複数人に対する画像が通知されるので、抑止効果も期待できる。

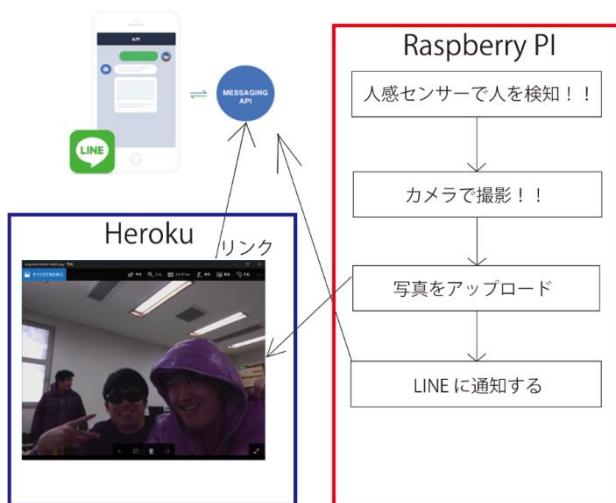


図 1 システム構成図

### 3. 研究概要

#### 3.1 開発環境について

開発環境は、以下の通りである。

- Raspberry Pi 3 Model B
- PIR モーションセンサー
- LINE
- Raspberry Pi カメラモジュール V2
- Messaging API
- Heroku

#### 3.2 LINE と Messaging API

LINE（ライン）は、韓国の IT 企業ネイバーの子会社、LINE 株式会社が提供するソーシャル・ネットワーキング・サービス(SNS)である。

Messaging API は、LINE が提供する LINE の BOT を作成するための API のことである。

#### 3.3 Heroku について

Heroku とは PaaS(Platform as a Service)と呼ばれるサービスで、アプリケーションを実行するためのプラットフォームである。

LINE に画像を送信するには https://で始まる画像リンクが必要になるため、Heroku サーバーが必要になる。

#### 3.4 Raspberry Pi 3 Model B について

Raspberry Pi は、ARM プロセッサを搭載したシングルボードコンピュータである。イギリスのラズベリーパイ財団によって開発されている。



図 2 Raspberry Pi 3 Model B

#### 4. 開発手順

##### 4.1 Heroku アカウント及びプロジェクトの作成

初めに、<https://www.heroku.com> にアクセスし、Sign up をクリックし、サイトに書いてある手順に従って、アカウントとプロジェクトを作成していく。(プロジェクト名：agile-falls-\*\*\*\*\*)

##### 4.2 LINE チャンネルと BOT の作成

この作業は開発者として登録するための作業である。今回は、無料で BOT の全機能が使える

LINE Developer で BOT を作成した。

- ・初めに <https://developers.line.me/ja/> にアクセスする
- ・Messaging API を始めるを押す
- ・プロバイダーの作成
- ・BOT の作成 (今回は Developer Trial で作成)



図 3 LINE の BOT 作成ページ

##### 4.3 Raspberry Pi のネットワーク設定

ネットワークは /etc/network/interfaces ファイルで設定を行う。今回は有線と無線どちらも設定した。

##### 4.4 人感センサとカメラのリンクの設定

- ・人感センサーは Raspberry Pi の GPIO のピンにつなぐ。

Raspberry Pi B Rev 2 P1 GPIO Header			Raspberry Pi B+ B+ J8 GPIO Header			
Pin No.			Pin No.			
3.3V	1	2	5V	1	2	5V
GPIO2	3	4	5V	3	4	5V
GPIO3	5	6	GND	5	6	GND
GPIO4	7	8	GPIO14	7	8	GPIO14
GND	9	10	GPIO15	9	10	GPIO15
GPIO17	11	12	GPIO18	11	12	GPIO18
GPIO27	13	14	GND	13	14	GND
GPIO22	15	16	GPIO23	15	16	GPIO23
3.3V	17	18	GPIO24	17	18	GPIO24
GPIO10	19	20	GND	19	20	GND
GPIO9	21	22	GPIO25	21	22	GPIO25
GPIO11	23	24	GPIO08	23	24	GPIO08
GND	25	26	GPIO07	25	26	GPIO07
			DNC	27	28	DNC
			GPIO05	29	30	GND
			GPIO06	31	32	GPIO12
			GPIO13	33	34	GND
			GPIO19	35	36	GPIO16
			GPIO26	37	38	GPIO20
			GND	39	40	GPIO21

図 4 Raspberry Pi の GPIO

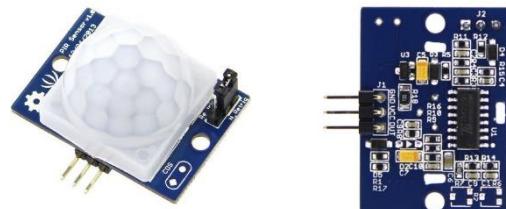


図 5 PIR モーションセンサー

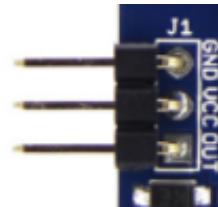


図 6 モーションセンサー拡大図

今回はジャンパーウイヤーで GND は GND、VCC は 3.3V に、OUT は今回 18 番に接続する。  
(GPIO のピンは指定されたところに接続すればどこでもよい)



図 7 カメラモジュール接続

この段階まで終わったら Python3 で PIR モーションセンサーで検知したら撮影するプログラム <camera.py>を書く。

```
<camera.py>
//撮影する画像の縦のサイズを 1024 に設定
PICTURE_WIDTH = 1024
//撮影する画像の縦のサイズを 1024 に設定
PICTURE_HEIGHT = 1024
//保存先ファイルパスを指定する
SAVEDIR = "original"
//撮影の間隔は 30 秒
INTAVAL = 30
//1秒のスリープ時間
SLEEPSIME = 1
//GPIO ピンは 18 番を使用
SENSOR_PIN = 18
//人を検知したら JPEG で画像を保存する
while True:
    if(GPIO.input(SENSOR_PIN) ==
GPIO.HIGH) and (st + INTAVAL <
time.time() ):
        st = time.time()
        filename =
time.strftime("%Y%m%d%H%M%S") + ".jpg"
        save_file = SAVEDIR + filename
        cam.capture(save_file)
        time.sleep(SLEEPSIME)
```

#### 4.5 画像のアップロード

LINE に画像を送信するには Heroku 上に、撮影した画像をアップロードしなければならない。よって、Heroku に以下のような PHP プログラム <upload.php> を配備し、画像を Heroku に画像をアップロードできるようにした。

次に、python でラズパイ内に保存されている画像を [https://agile-falls-\\*\\*\\*\\*.herokuapp.com/images/](https://agile-falls-****.herokuapp.com/images/) 上にアップロードできるようなプログラム <imageupload.py> を作成した。

```
<upload.php>
//images という保存先を指定する
$uploaddir = 'images/';
<imageupload.py>
```

```
//Heroku のアップロード先の URL
url = https://agile-falls-
*****.herokuapp.com/upload.php
//**に撮影済みの画像をアップロードする
files = {'imagefile': open("*****",
"rb")}
```

#### 4.6 LINE への通知

最後に LINE に通知するプログラムを作成した。この際、4.1～4.5までのプログラムを組み合わせた <camera2line.py> を作った。

カメラで撮影した画像のサイズは original 画像 (1024\*1024) に設定していた。LINE は送信できる画像のサイズの上限があり、original 画像は 1024\*1024 まで、preview 画像は 240\*240 である。

今回は 1024\*1024 で画像を撮影するようにしているため、preview 画像は 240\*240 まで縮小してから送るよう設定した。

```
<camera2line.py>
//BOT に送信する際に必要な URL
url='https://api.line.me/v2/bot/message/
push'
//original 画像の保存先パス
SAVEDIR = "original"
//preview 画像の保存先パス
SAVEDIR2 = "preview"
//**に BOT のアクセストークンを設定する
token = '*****'
//**に BOT のユーザーID を設定する
"to": "*****",
//original 画像のアップロード
"originalContentUrl":"https://agilefalls
***.herokuapp.com/images/" + save_file,
//preview 画像のアップロード
"previewImageUrl": "https://agile-falls-
***.herokuapp.com/images/" + revision_file
//画像を縮小する
Image.open(save_file).resize((240,240)).
save(revision_file)
```

#### 4.7 グループ会話の設定

最後は複数人と情報を共有することで犯罪の抑止力にもなると考えグループを作成し、プログラムのユーザーIDの部分にグループIDを設定した。

#### 5. 運用を想定した機能の検討

##### 5.1 手動起動から自動起動へ

今回の防犯システムを作成するにあたり、実際に運用するとなると、今のままでは手動で一から起動しなければならない。

よって、ラズパイを起動したらシステムが自動で起動できるようにする。

##### 5.2 監視のオンとオフ機能

システムをずっと起動していると、毎時間のように画像を送信してしまう。

そこで、BOTのトーク画面を開き、今は撮影しなくてもよいからOFF、これから出かけるからカメラをONにしようというように、メッセージにONかOFFかを送信するだけで、今は撮影し画像を送信すべきかどうかの切り替えができるようになる。

#### 6. システムの実行及び評価

人感センサーで人を検知したら、すぐにカメラで撮影し、撮影した画像をHerokuサーバーにアップロードすることができた。そのURLを利用し、LINEのBOTに画像を送信することができた。よって、今回の卒業研究の課題であるLINEに検知した画像を送信する技術的な問題はすべて解決することができた。



図 8 実際に LINE に送られた画像

#### 7. 終わりに

今回は、無人の場所でも誰が侵入したかわかるような防犯システムを作成した。LINEのBOTに人感センサーで人を検知し画像を撮影し送信できた。後は、運用に向けた機能の追加をしていこう。もしこれからできるならば、BOTを登録したユーザー全員がみれるような設定もしていきたい。

#### 8. 参考文献

「API リファレンス-LINE Developers」

URL:<https://developers.line.me/ja/docs/messaging-api/reference/>

「人感センサーとカメラを Raspberry Pi につなげてみた」

URL:<http://akkun49.blog.fc2.com/blog-entry-77.html>

「ラズベリーパイのドキュメンテーション」

URL:<https://www.raspberrypi.org/documentation/configuration/wireless/access-point.md>

「LINE BOT を作ろう！ Messaging API を使ったチャットボットの基礎と利用例」

著：立花 翔