2016 年度（平成 28 年度） 卒業プロジェクト

「自家用車向けパーソナルアシスタントの研究」

“Personal Assistant for Private Vehicles”

増井俊之研究会
学籍番号: 71346711　環境情報学部４年　名児耶均基

論文要旨

近年、希少価値の高まっている旧式のスポーツカー等を狙った自動車盗難事件が増加しており、駐車中の車両盗難やいたずらが所有者の不安材料になっている。所有者が車と常に繋がっているような安心感があればそのような不安は払拭されると考えられる。

本研究では、自家用車向けのパーソナルアシスタントを設計・実装した。これにより、安価でかつ手軽な手法で駐車中の不安を払拭し、パーソナルアシスタントとの SNS（Twitter）上での対話を通して心理的な安心感を生み出すことに成功した。

キーワード： 自動車、セキュリティ、SNS、Twitter、IoT

慶應義塾大学環境情報学部環境情報学科　名児耶均基

目次

第一章　研究背景と目的 5

1.1.　研究背景 5

1.2.　研究目的 5

1.3.　本論文の構成 5

第二章　先行事例および関連研究 7

2.1.　先行事例・関連研究 7

2.1.1　カーセキュリティ分野 7

2.1.2　インフォテインメント分野 7

2.2.　本研究の新規性と予想される効果について 8

第三章　システムの提案と実装 9

3.1.　システムの概要 9

3.2.　システムの構成 10

3.3.　システムの実装 12

3.3.1　Twitter のタイムライン監視 13

3.3.2　カーセキュリティ機能 13

3.3.3　位置情報の参照機能 14

第四章　評価 16

4.1.　評価 16

4.2.　考察 16

第五章　結論 17

5.1.　本研究によって得られた成果 17

5.2.　今後の課題と展望 17

謝辞 18

付録 A（ソースコード） 19

A.1.　twistream.py 19

A.2.　acceralation.py 20

A.3.　location.py 21

参考文献......22

第一章 研究背景と目的

1.1. 研究背景

近年、希少価値の高まっている旧式のスポーツカー等を狙った盗難事件が増加している。イモビライザーと呼ばれる電子的なキー照合システム等の盗難防止装置が付いていない車両やキーの複製が簡単な旧式の車両の所有者にとって、特に駐車中の車両盗難やいたずらが不安材料であると推測される。

旧式のスポーツカーを所有するような車好きの人間にとって、マイカーは言わば家族や恋人のような存在であり、車と常に繋がっているような安心感があれば、そのような不安は払拭されると考えられる。

市販のカーセキュリティ装置は高価であり、取り付けにも専門知識を要する。いたずらや窃盗行為の振動や犯人の接近を検知して警報音を出すような物が多く、万が一車両を盗難されてしまっても為す術がない。また、GPS を用いて愛車の位置情報を参照したり、PHS・3G 等のデータ通信回線を用いて異常を通知したりするものは一部のハイエンドモデルに留まる上、月額料金や位置情報の参照の度に料金を請求されるというのが現状である。

現在は小型で安価なコンピューターやセンサー類が手に入るようになっているため、それらを組み合わせて応用すれば、既存の物と比較して更にスマートでインタラクティブなカーセキュリティ装置を実現できるのではないかと考えた。具体的には、ユーザーと擬人化されたパーソナルアシスタント間で SNS（Twitter）を利用した対話を行うことでこれを実現する。

1.2. 研究目的

本研究では、自家用車用のインターネットに接続された車載パーソナルアシスタント装置を利用することにより、主にいたずらや窃盗行為に対する不安を低減し、車と繋がっているような安心感を生み出すことで、より快適なカーライフを送ることが出来るようにすることが目的である。具体的には、自家用車に、センサー等を接続した小型のコンピューターを設置し、いたずらや窃盗行為を所有者に通知する。また、所有者が擬人化されたパーソナルアシスタントアシスタントと対話することで、装置の盗難通知（セキュリティ機能）をオン・オフしたり、車両の位置情報を取得したりできる。

1.3. 本論文の構成

本稿では、第一章にて本研究の背景と目的について述べた。第二章では、先行事例

および関連研究、この研究の新規性と予想される効果について論ずる。第三章では、具体的なシステムの提案と実装方法を論述する。第四章では、評価および考察を行い、第五章では本研究の総括をする。

第二章 先行事例および関連研究

2.1. 先行事例・関連研究

本研究と関連する自動車におけるセキュリティ分野や、インフォテインメントシステムにおけるパーソナルアシスタントの先行事例や研究を紹介する。

2.1.1 カーセキュリティ分野

市販のカーセキュリティ装置の多くは、前述のように、いたずらや窃盗行為の振動や犯人の接近を検知して警報音を出すような物に留まる。加藤電機「HORNET 370V」「NEW iVIPER」等、一部メーカーの高価なモデルには、GPS と PHS・3G 等のデータ通信回線を用いた、異常の通知機能・位置情報の参照機能を搭載したものがある。

また、セコムの提供する「ココセコム」と呼ばれるサービスにも、車両に機器を取り付けて位置情報を参照できるものがある。

これらのシステムでは通信サービスの月額使用料や位置情報取得に都度料金がかかる。また、機器の取り付けにも専門的な知識と複雑な配線作業が求められるため、導入が難しいのが難点である。そして、装置と対話するようなインタラクティブな機能はなく、無骨な印象であるのが現状である。

2.1.2 インフォテインメント分野

スマートフォンやパーソナルコンピューターの分野で Apple の「Siri」や Microsoft の「Cortana」のような擬人化された一般消費者向けアシスタントボットが増えている[1]ように、自動車のインフォテイメントシステムの分野においてもパーソナルアシスタント機能が増えている。例えば、BMW のインフォテイメントシステムである「iDrive」には、「スピーチ・コントロール」機能があり、自然言語処理や AI を利用したナビゲーションの設定や音楽の選曲を行ったり、タイヤの空気圧やオイルの状態などの車両コンピューターの情報を参照したりすることが可能[2]となっている。また、自動車メーカー以外の後付けの製品では、ユピテル社の GPS&レーダー探知機「Lei03」では、「霧島レイ」というキャラクターが音声案内を通して、ゲーム感覚で安全運転をサポートする[3]という商品も販売されている。

このような、デジタルボットを介した対話型のユーザー体験が増えているのは、デジタルボットが機械の操作を簡単にするだけでなく、まるで人物とコミュニケーションをとっているようなリッチな体験を作り出すからである。

2.2. 本研究の新規性と予想される効果について

本研究の装置では、センサーや通信モジュールを取り付けた小型軽量ボードコンピューターを車内に設置する方式を用いる。この方法では、常時電源を車両側から供給するだけで良いため、既存のカーセキュリティと比較して導入が非常に簡単である。また、車両に依存しない外付けのシステムであるため、旧式の自動車にも導入が簡単であり、手軽にセキュリティを向上させる手法としての利用価値は非常に高いと考えられる。そして、近年増えている MVNO の格安 SIM カードや、「SORACOM Air」等の IoT デバイス向けのデータ通信 SIM を活用すれば、前述のような既存のカーセキュリティと比較して位置情報の取得にかかる通信料金を大胆に減らすことができるだろう。

Twitter 等の SNS プラットフォームを介して自分の車と対話するという手法はかつて存在せず、新規性が高い。この方式には、サーバーを自分で用意することが必要ないというメリットがある。自動車を擬人化し、インターネットと接続し、いつでも車両の位置情報やいたずら・窃盗行為の兆候を知ることのできる機能を持たせることで、ユーザーの不安の低減し、より快適で安心感のあるカーライフを送ることが出来るようになると本研究では期待している。

第三章 システムの提案と実装

3.1. システムの概要

本システムでは、自家用車の車内に小型軽量ボードコンピューターを設置し、車両に異常があれば、その旨をインターネット経由で所有者に通知し、GPS で取得した車両の位置情報を参照できるようにする。

システムと所有者の対話には Twitter を使用して前述の機能を操作する。

今回の装置ではマイカーを擬人化し、パーソナルアシスタントとして愛車の様子を見守るという設定にした。擬人化のイメージをわかりやすくするために、オリジナルキャラクターを用意した。著者の名前である「均基（まさき）」と、著者の所有している車両である「マツダ・RX-8」のエイトを振り、「真咲エイト」と命名した。以下がそのキャラクターと著者の所有している車両の写真である。

図 3.1: 著者の愛車をイメージしたオリジナルキャラクターである「真咲エイト」

図 3.2: 著者の所有している「マツダ・RX-8」

3.2. システムの構成

ハードウェアのメインボードには、小型軽量ボードコンピューターの「Raspberry Pi 3 Model B」を採用した。インターネットに常時接続するために、Huawei 製の USB 接続の 3G データ通信モジュール「EMOBILE GD03W」を用いた。SIM カードには株式会社インターネットイニシアティブの「IIJmio データ通信専用 SIM」を用いた。これにより、通信料が月額税抜き 900 円での運用が可能[4]となる。(2017 年 1 月現在)

車両の位置情報を取得するための GPS モジュールには、U-blox 社の「NEO-6M」を搭載した UART のシリアル通信が可能なモジュールを使用した。また、受信感度を向上させるために GPS モジュールに汎用のアクティブアンテナを接続し、アンテナは車両のダッシュボードに設置した。

車両のいたずら・盗難の兆候を検知するための加速度センサーには、3 軸加速度センサーモジュールの「ADXL345」を使用した。Raspberry Pi と加速度センサーモジュールの通信には I^2C 通信を用いている。

GPS モジュール、加速度センサーモジュールはブレッドボード上に設置し、Raspberry Pi と接続した。GPS アンテナ以外のハードウェアは助手席グローブボックス内に設置したが、加速度センサーで車両の揺れを問題なく取得することができた。

ソフトウェア環境には、Raspberry Pi Foundation の公式オペレーティングシステムシステムである「Raspbian」の最新版、「Raspbian Jessie with PIXEL」を用いた。

そして、ボットソフトウェアの開発には Python 言語を用いている。

ハードウェアへの電源供給は、車両の車内ヒューズボックスからエーモン製「【1542】電源ソケット(ヒューズ電源タイプ)」および、汎用の USB シガーソケットチャージャーを用いて行った。

システムの構成図は以下のようになっている。

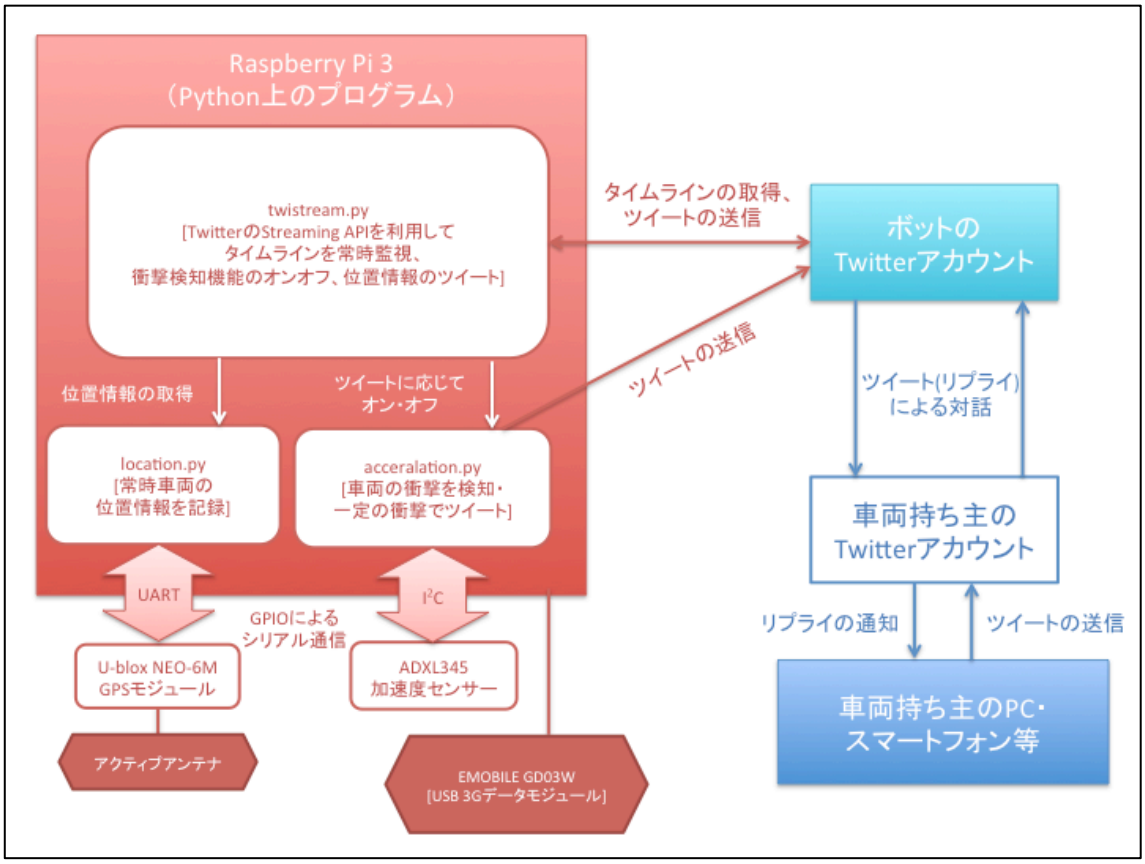

図 3.3: システム構成図

なお、実際のシステムは以下の写真のようになっている。

図 3.4: 助手席グローブボックス内に設置しているハードウェアを取り出して撮影した写真

図 3.5: ダッシュボードに設置した GPS 受信感度強化用の汎用アクティブアンテナ

3.3. システムの実装

まず、パーソナルアシスタント用の Twitter のアカウントとして@raspi_car_sec を作成した。

図 3.6: パーソナルアシスタントが利用するアカウント「真咲エイト (@raspi_car_sec)」の Twitter アカウント

パーソナルアシスタントの機能として 3 つの大きな機能を用意した。以下が各機能の詳細と実装方法である。

3.3.1 Twitter のタイムライン監視

今回のシステムでは、Twitter を介してユーザーと対話するため、Twitter の Streaming API を用いて常時リプライの監視を行っている。それを担うプログラムとして、「twistream.py」というメインプログラムを Raspbian OS の起動時に自動実行している。Python 上で Twitter を扱うライブラリとして「TwitterAPI」(https://github.com/geduldig/TwitterAPI)を用いた。メインプログラムを実行すると、同時に後述の位置情報を常時記録する「location.py」を別のスレッドで実行している。

「twistream.py」では、Twitter のリプライを常時 Streaming API で監視しており、特定ユーザー(車両所有者、今回の場合は著者の大学用の TwitterID である@knps_sfc)のリプライに含まれる特定の文字列を判定して、後述の所定の動作を行うようになっている。

3.3.2 カーセキュリティ機能

カーセキュリティ機能は、愛車がいたずらや窃盗行為の脅威にさらされた時の為の通知機能である。Raspberry Pi に接続された加速度センサー「ADXL345」を用いて、一定の加速度があれば車両の所有者にリプライを送信する。これにより、所有者は素早い初期対応を行うことが可能になる。所有者がボットに「オン」、「ON」、「on」というワードの入ったリプライを送信することで、「acceralation.py」のスレッドが実行され、この機能を有効化することができる。また、所有者が「オフ」、「OFF」、「off」というワードの入ったリプライをボットに送信することでこの機能が無効化される。機能のオン・オフの際にはその旨のツイート(リプライ)が所有者に投稿されるようになっている。Python 上で ADXL345 を扱うライブラリには「ADXLpython」

（https://github.com/pimoroni/adxl345-python）を使用している。ADXL345 上で 0.1 秒の間に 0.1G の加速度変化があると所有者に通知するようにした。これは、センサーの誤検知がなく、ドアを開閉する程度の揺れを取得することが出来る適切なしきい値を測定して決定したものである。

図 3.7: カーセキュリティ機能をオンにした際のリプライツイート

図 3.8: 車両の異常が検知された際のリプライツイート

図 3.9: カーセキュリティ機能をオフにした際のリプライツイート

3.3.3 位置情報の参照機能

「twistream.py」の実行時に自動起動される「location.py」により、Raspberry Pi に接続された GPS モジュール「NEO-6M」の位置情報を 30 秒毎にテキストファイルに書き出している。Raspberry Pi 上で GPS を扱うアプリケーションとして「GPSD」（https://savannah.nongnu.org/projects/gpsd/）を利用した。「twistream.py」で記録された位置情報を読み込み、所有者から「位置」、「どこ」というワードの入ったリプライが来ると、位置情報の埋め込まれたツイートを時間と共に所有者宛に呟く。位置情報のツイートには Twitter Rest API エンドポイントのパラメーターである、「lat」に緯度の情報を、「long」に経度の情報を、「display_coordinates」に「true」をそれぞれ付加することで正確な位置情報を投稿している。

図 3.10.1: 位置情報を参照した際のリプライツイート

図 3.10.2: 図 3.10.1 の赤枠をタップし、位置情報の詳細をツイッタークライアントで確認した様子。車両の位置がピンで示されている。

第四章 評価

4.1. 評価

本研究のシステムを著者が実際にテストした結果、いつでも自分の愛車の異常を知ることが出来ること、そして位置情報の確認が出来ることによる確かな心理的な安心感を得ることが出来た。

今回のシステムのハードウェアの制作にかかるコストは、ボードコンピューターや各種センサー、モジュール等を合わせても 15,000 円弱で実現することが出来た。(中古パーツを含む) そのため、市販のカーセキュリティシステムよりも導入にかかるコストが安いことが分かった。そして、配線作業も複雑ではないため、車両への設置も 10 分程度で終わらせることが出来た。

セキュリティの機能をオン・オフする度、位置情報を確認する度に、オリジナルキャラクターである「真咲エイト」に和まされた。擬人化された自分の愛車だけのオリジナルキャラクターがアシスタントをしてくれるという優越感と人間らしさに由来する安心感があった。

Twitter を用いてボットと対話する手法は、ボットアカウントに特定のキーワードを含めたリプライを送るだけで良いため、専用のアプリケーション等が必要なく、著者のような Twitter のヘビーユーザーには使いやすい方法であった。また、Twitter のクライアントの通知機能によって車両の異常がスマートフォンのプッシュ通知で分かることも非常に有益であった。

また、GPS の位置参照機能を用いることで、広い駐車場で愛車の位置を知ることができるのも結果的に有用な機能であった。

4.2. 考察

本装置を用いることで、常に愛車の位置情報や異常を知ることができるため、万が一、愛車の盗難やいたずらをされてしまった際に素早い初期対応が出来ることは非常に魅力的かつ有用であろう。しかしながら、毎回車両を離れる際と戻ってくる際にセキュリティ機能を手動でオン･オフすることが少し面倒であったのは事実であり、改善すべき点であった。

前述のように安価で導入の簡単なカーセキュリティとしては非常に有効な手段ではあるが、システムの存在が車外から見て分かるものではないため、いたずらや窃盗を行う者に対する抑止力にはなっていなかった。異常時のアラーム等の警報があると、被害の低減に繋がったのではないかと感じた。

第五章 結論

5.1. 本研究によって得られた成果

安価で手軽に実装できるカーセキュリティとしての小型ボードコンピューターの車載は効果的である。

窃盗などを物理的に防ぐことは出来ないが心理的な安心感が一番大きな効果であった。

Twitter を用いたシステムとの対話はシンプルで使いやすく、プッシュ通知の機能もあるため、スマートフォン等のモバイルデバイスを利用する時代の車載ボットシステムとの対話手段として有用であると考えられる。

単なるカーセキュリティは機械的で味気ないが、擬人化したパーソナルアシスタントとのコミュニケーションを通すことで、ユニークかつ魅力的ものとなった。愛車を擬人化し、パーソナライズがされたキャラクターは、アバターのように特に愛着を持つことができる。パーソナルアシスタントとしての車載システムは、人間らしさによる安心感と自分専用であるという優越感があることが分かった。

5.2. 今後の課題と展望

本研究のパーソナルアシスタントは予め用意された単純なメッセージの対話しかできないため、機械学習や AI の機能をパーソナルアシスタントに組み込み、更にリッチな対話をすることが出来れば、ユーザーはもっと楽しむことができるのではないかと感じた。また、車両側のボードコンピューター等と統合し、様々な情報を参照したり、車両のあらゆる操作（例えば、車外からエアコンをオンにしたり、ナビゲーションの目的地をセットする等）が出来たりするとユーザーも便利であろう。

今回の実装では、愛車にいたずらや盗難の兆候があるときのみボットから自発的にリプライを送るが、例えば、一定時間以上車に乗らないと定期的に愛車が自分にメッセージを送ること等があれば、更なる愛車との一体感が生まれ、もっと車に乗りたいという気分にさせるのではないかと感じた。

前述のように手動でのセキュリティ機能のオン・オフが面倒であったため、Bluetooth の Beacon を用いた近接検知での自動的なセキュリティのオン･オフが出来れば更に便利で快適だったと考えられる。

謝辞

増井俊之教授、そして増井俊之研究室の院生、学部生の皆様方には大変お世話になりました。本研究以外にもあらゆるアドバイスを賜りましたことを、心から感謝しております。

また、「真咲エイト」の擬人化に協力していただいた、春野すずらん(@latte_suzuran)氏、本当にありがとうございました。

付録 A（ソースコード）

A.1. twistream.py

```
# -*- coding: utf-8 -*-

from TwitterAPI import TwitterAPI
import os
import datetime
import time
import threading
from adxl345 import ADXL345
from acceralation import Accera
from location import GetLocation

CONSUMER_KEY =
CONSUMER_SECRET =
ACCESS_TOKEN_KEY =
ACCESS_TOKEN_SECRET =

api = TwitterAPI(CONSUMER_KEY,
                 CONSUMER_SECRET,
                 ACCESS_TOKEN_KEY,
                 ACCESS_TOKEN_SECRET)

def Stream():
        r = api.request('user')
        for msg in r:
                now = datetime.datetime.today().strftime("[%Y/%m/%d %H:%M:%S]")
                ripu = msg['user']['screen_name'] if 'user' in msg else msg
                naiyou = msg['text'] if 'text' in msg else msg
                print(naiyou)
                if ripu == 'knps_sfc' and u'オン' in naiyou or 'ON' in naiyou or 'on' in naiyou:
                        a = Accera()
                        t = api.request('statuses/update',{'status':'@knps_sfc セキュリティをオンにしたよ！ '+now})
                        print('セキュリティがオンになりました')
                if ripu == 'knps_sfc' and u'オフ' in naiyou or 'OFF' in naiyou or 'off' in naiyou:
                        a.stop()
                        t = api.request('statuses/update',{'status':'@knps_sfc セキュリティをオフにしたよ！ '+now})
                        print('セキュリティがオフになりました')
                if ripu == 'knps_sfc' and u'位置' in naiyou or u'どこ' in naiyou:
                        print(naiyou)
                        f = open('ido.txt','r')
                        g = open('keido.txt','r')
                        ido = f.read()
                        keido = g.read()
                        t = api.request('statuses/update',{'status':'@'+ripu+u' ここにいるよ！！ '+now, 'lat':str(
                            ido), 'long':str(keido), 'display_coordinates':'true'})
                        print(now)
                        print(t.status_code)


def TwiStream():
        while True:
                try:
                        Stream()
                except:
                        print("Unable to connect to Twitter. Wait a minute and reconnect...")
                        time.sleep(60.0)
                        continue

threading.Thread(target=GetLocation).start()
TwiStream()
```

A.2. acceralation.py

```
# -*- coding: utf-8 -*-
from TwitterAPI import TwitterAPI
from adxl345 import ADXL345
import time
import datetime
import threading

CONSUMER_KEY =
CONSUMER_SECRET =
ACCESS_TOKEN_KEY =
ACCESS_TOKEN_SECRET =

api = TwitterAPI(CONSUMER_KEY,
                 CONSUMER_SECRET,
                 ACCESS_TOKEN_KEY,
                 ACCESS_TOKEN_SECRET)

class Accera():
        def __init__(self):
                self.stop_event = threading.Event()
                self.thread = threading.Thread(target = self.target)
                self.thread.start()

        def target(self):
                adxl345 = ADXL345()
                axes = adxl345.getAxes(True) #initialize

                time.sleep(5.0)

                while not self.stop_event.is_set():
                        adxl345 = ADXL345()
                        axes = adxl345.getAxes(True)
                        xold = axes['x']
                        yold = axes['y']
                        zold = axes['z']

                        time.sleep(0.1)

                        axes = adxl345.getAxes(True)
                        x = axes['x']
                        y = axes['y']
                        z = axes['z']

                        xdif = x-xold
                        if xdif < 0:
                            xdif = -xdif

                        ydif = y-yold
                        if ydif < 0:
                            ydif = -ydif

                        zdif = z-zold
                        if zdif < 0:
                            zdif = -zdif

                        now = datetime.datetime.today().strftime("[%Y/%m/%d %H:%M:%S]")

                        if xdif > 0.1 or ydif > 0.1 or zdif > 0.1:
                            print "Detected Acceralation!"
                            t = api.request('statuses/update',{'status':'@knps_sfc 車両の異常を検知したよ!(>w<) '+now})
                            time.sleep(60.0)

                        print('実行中')

        def stop(self):
                self.stop_event.set()
                self.thread.join()
```

A.3. location.py

```
#! /usr/bin/python
# -*- coding: utf-8 -*-
import os
from gps import *
from time import *
import time
import ConfigParser

def Location():
  session = gps()
  session.stream(WATCH_ENABLE)

  keido=""
  ido=""
  for report in session:
      if report['class'] == 'TPV':
        keido = report['lon']
        ido = report['lat']

        f = open('keido.txt','w')
        f.write(str(keido))
        f.close

        g = open('ido.txt','w')
        g.write(str(ido))
        g.close

  print(ido, keido)
  time.sleep(30.0)

def GetLocation():
  while True:
    try:
      Location()
      print('実行中')
    except:
      print('Waiting 60 sec, then reconnect GPS...')
      time.sleep(60.0)
      continue
```

参考文献

[1]: 50 億ドル市場の人工知能、Amazon Alexa、Microsoft Cortana、Apple Siri が変えるビジネスと生活の未来とは - THE BRIDGE（ザ・ブリッジ）. http://thebridge.jp/2016/12/how-amazon-alexa-microsoft-cortana-and-apple-siri-will-help-automate-business .

[2]: ここまで進歩した！BMW7 シリーズの音声認識がすごい理由｜エコカー大戦争！｜ダイヤモンド・オンライン. http://diamond.jp/articles/-/87944 .

[3]: Lei03 - GPS＆レーダー探知機「霧島レイ」モデル. http://lei-kirishima.jp/sp/lei03/ .

[4]: データ通信専用 SIM | IIJmio. https://www.iijmio.jp/hdd/data/ .