Vol.2019-SLDM-189 No.15
Vol.2019-EMB52 No.15
2019/11/14

SNS とクラウドを利用したセキュリティカメラシステムの開発

江頭拓也 [1]　米橋健太 [1]　冨山宏之 [1]　孟林 [1]　谷口一徹 [2]

概要：セキュリティカメラシステムは、オフィスや公共施設だけでなく、一般家庭でも広く導入されている。多くのセキュリティカメラシステムは、カメラを制御するための専用アプリケーションをインストールしなければならない。そこで、本研究ではクラウドと SNS に基づくセキュリティカメラシステムを開発した。開発したセキュリティカメラシステムは、カメラを制御するための専用アプリケーションを必要としない。 代わりに、UI として Twitter を使用する。また、AI ベースの顔認証に Amazon Web Service クラウドシステムを利用する。

キーワード：Amazon Web Service，Twitter，セキュリティカメラシステム

1. はじめに

現在普及している多くのセキュリティカメラは、動画をローカルストレージに保存し、必要に応じて、ユーザが動画を後で確認している。最近のセキュリティカメラは IoT (Internet of Things)技術の進歩により、インターネットに接続され、撮影された動画はクラウドに保存する。ユーザは、スマートフォンなどのモバイルデバイスや PC を使用して、カメラの映像をリアルタイムで視聴することも、保存した動画を後から視聴することも可能である[1][2][3]。最近では、AI (Artificial Intelligence)テクノロジの進歩により、セキュリティカメラは不審者をリアルタイムで自動的に検出することも可能である[4] [5] [6] [7] [8]。検出されると、インターネットを通じて警備会社またはカメラの所有者に警告が送信される。

IoT 技術に基づくセキュリティカメラシステムのほとんどは、カメラ映像の視聴やカメラの制御を行うアプリケーションソフトウェアを必要としている。その為、ユーザは、スマートフォンや PC にアプリケーションソフトウェアをインストールする必要がある。ソフトウェアが特定の種類のデバイス、たとえば Android デバイスのみに提供される場合、カメラのユーザは Android ユーザに制限される。一方で Android デバイス、iPhone、Windows PC などのさまざまなデバイス向けのアプリケーションソフトウェアの開発には時間とコストがかかってしまう。

本研究では、クラウドと SNS を利用したホームセキュリティカメラシステムを開発した。開発したセキュリティカメラシステムには、大きくふたつの特徴が存在する。ひとつ目はシステムの UI に Twitter を使用している点である。ふたつ目は顔認証に AWS (Amazon Web Service)を使用している点である。ひとつ目について、Twitter は世界で最も人気のある SNS のひとつであり、Android や iPhone などのさまざまなデバイスでの利用が容易である。また、Twitter はインターネットブラウザを搭載した PC でも利用できる。

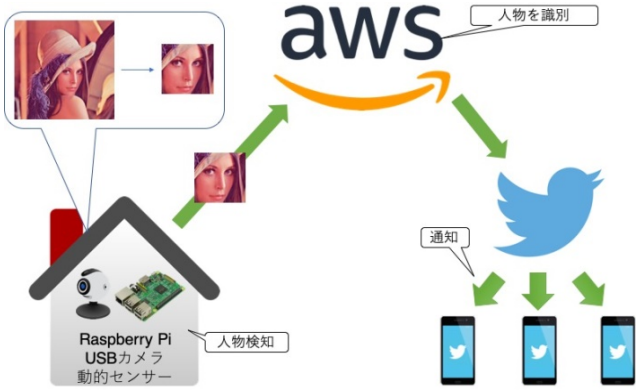

図 1　システムの概要図

この Twitter を利用した UI は、ユーザの利便性を大幅に向上させる。ふたつ目について、本研究のシステムは、AWS が提供している高精度な AI に基づく顔認証を利用している。

本論文は以下のように構成される。第 2 章では開発したホームセキュリティカメラシステムの概要を、第 3 章ではその詳細について説明する。第 4 章では、システム全体の流れを説明し、第 5 章ではリアルタイムの評価を行う。第 6 章は本論文のまとめを行う。

2. システムの全体像

開発したセキュリティカメラシステムの概要を図 1 に示す。カメラシステムは、カメラデバイス、AWS、Twitter、およびスマートフォンなどのユーザデバイスで構成される。

ここでカメラデバイスは、小型のコンピューターボードである Raspberry Pi3、赤外線人感センサ、および Web カメラの 3 つから構成される。カメラデバイスは、玄関のドアに向かって家の中に設置されると想定している。人がドアから入ると、赤外線人感センサが人を検出し、Raspberry Pi3 に接続されている Web カメラが撮影を開始する。 その後、撮影された人の顔部分がトリミングされ、AWS へ送信される。次に、送信された画像は AWS で事前に登録されている顔画像と照合され、人物の特定処理が実行される。ここ

1 立命館大学
Ritsumeikan University.
2 大阪大学
Osaka University

Vol.2019-SLDM-189 No.15
Vol.2019-EMB52 No.15
2019/11/14

図 2　撮影された 10 枚の画像

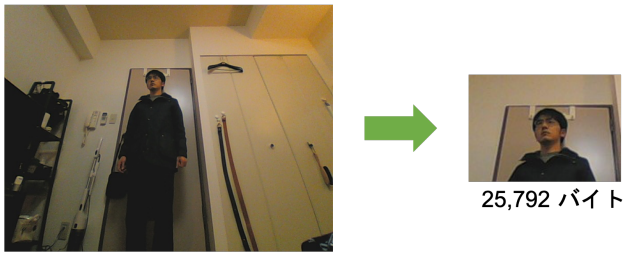

図 3　トリミングによるデータサイズの削減

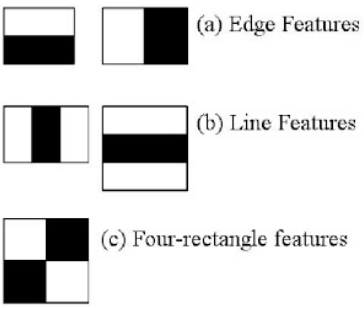

図 4　Haar-like 特徴分類器の例 [9]

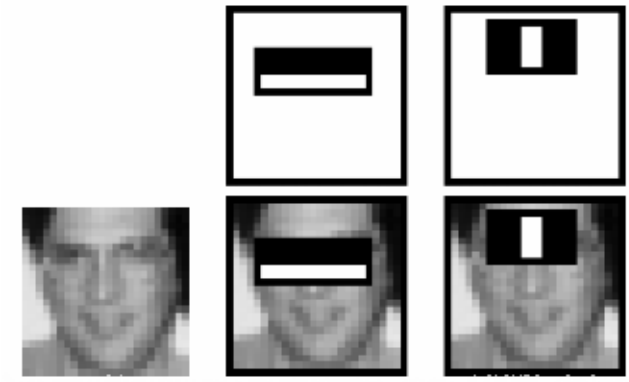

図 5　Haar-like を目の部分に重ね合わせた検出結果[9]

で不審者か否かが判定される。その結果は Twitter を介してユーザへ通知が行われる。また、新しい人物の登録も Twitter 上で行える。

3. 各システムでの詳細な動作

本章では、セキュリティカメラシステムの詳細について説明を行う。

(1) カメラデバイス

2 章で簡単に説明したように、カメラデバイスは Raspberry Pi3 ボード、赤外線人感センサ、および安価な Web カメラの 3 つで構成されている。赤外線人感センサと Web カメラは、それぞれ GPIO と USB ポートを介して Raspberry Pi3 ボードに接続されている。カメラデバイスは家の中に設置され、Web カメラは入り口のドアに向けられる事を想定としている。

まず、人がドアから入ると、赤外線人感センサが人を検出する。次に、図 2 に示すように、Web カメラは連続して 0.5 秒間隔で 10 枚の写真を撮影する。ここで 10 枚の写真を撮影する理由は、数枚の写真では人物の顔がはっきりと捉えられないことが多く、顔の検出には不十分であるからである。

撮影が終了すると、10 枚の写真について、Raspberry Pi3 ボード上で自動的に顔検出のプログラムが実行される。顔検出には、OpenCV を使用している。OpenCV は Haar-like 特徴を用いてブースティングされた分類器のカスケードを使用し顔の検出を行う。Haar-like 特徴を用いた分類とは、明暗で構成されたパターンで物体の検出を行う手法のことである。明暗のパターンの例を図 4 に示す。図 5 のようにさまざまな Harr-like 特徴分類器のパターンを局所的に重ね合わせ、検出対象の識別を行っていく。

図 3 に示すように、OpenCV により検出された顔は画像のサイズを縮小するためにトリミングが行われる。トリミングを行うことで画像のサイズが小さくなり、ネットワークトラフィックが軽減される。その後、画像ファイルは Raspberry Pi3 ボードから AWS に送信される。1 枚の写真に複数の人物が写っている場合、全ての顔が検出される。

(2) クラウドとしての AWS

本研究のカメラシステムでは、AWS をクラウドプラットフォームとして使用している。AWS は、エンジニアが高品質のプログラムを簡単に作成できるように、コンピューティングリソースとストレージリソースだけでなく、豊富なライブラリセットも提供されている。その中で本研究では、Amazon S3、Rekognition、AWS Lambda の 3 つを使用する。Amazon S3 (Amazon Simple Storage Service)は、AWS のストレージサービスである。カメラデバイスから送信された 10 枚の顔写真は Amazon S3 に保存される。Rekognition は画像の情報を分析するサービスである。人物のデータを事前に登録しておき、そのデータを用いて人物の特定を行う。今回は Rekognition のさまざまな分析機能のうち人の顔の比較機能を用いる。この機能を利用するには先述した AWS のクラウドストレージである Amazon S3 にあらかじめ人物の画像データを登録しておく必要がある。AWS CLI のコマンドで Rekognition に顔の比較をリクエストすると、Rekognition は比較画像を分析し、人物の特徴量を抽出する。あらかじめ登録していた顔と一致した場合は、画像と共に一致率を返す。

Vol.2019-SLDM-189 No.15
Vol.2019-EMB52 No.15
2019/11/14

図 6　顔の通知結果

図 7　顔の登録を行うダイレクトメッセージ

AWS Lambda は、AWS のコンピューティングプラットフォームである。新しい画像が Amazon S3 に保存されると自動的に顔認証を行う Lambda プログラムを開発した。開発した顔認証プログラムは、新しく受信した顔画像を事前にデータベースに登録されている顔のいずれかと一致するかどうかを判断する。一致した場合、新しい画像と一致率を返す。

(3) UI としての Twitter

本研究のカメラシステムは、Twitter を通じて、人が家に侵入したという事実を家族に通知する。この Twitter を使用する通知システムにはいくつかの利点が存在する。 1 つ目の利点は、ユーザがシステムの管理する為の専用のアプリケーションソフトウェアのインストールを必要としない点である。 Twitter は最も人気のある SNS の 1 つであり、多くの人々が既にデバイスに Twitter をインストールしており、Twitter の使用には慣れている。また、Twitter は Android スマートフォン、iOS スマートフォン、Windows PC、Linux PC など、さまざまなデバイスとプラットフォームの Web 上で動作する。２つ目の利点は、Twitter のアカウントを所持している家族全員が同時に情報を受信できる点である。

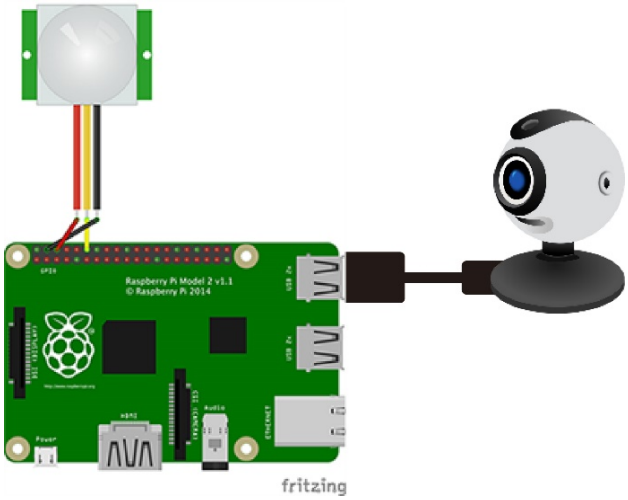

図 8　Raspberry Pi と周辺機器

AWS の顔認証プログラムによって、受信した写真の顔がデータベースに事前登録された人物であると判断されると、プログラムは Twitter に図 6 のようなメッセージを投稿する。投稿されたメッセージには、人物の名前と一致率が写真とともに表示される。一方で顔認証プログラムが写真の顔が事前登録された人物と一致しないと判断すると、プログラムは Twitter に警告メッセージを投稿し、図 6 に示すように、未知の人物が検出されたことを示す。

開発したカメラシステムにおいて、Twitter は、AWS からユーザへ通知する機能だけでなく、AWS のデータベースに新しく顔を登録する機能も備えている。顔の新規登録のプロセスを図 7 に示す。未登録の人物が初めて家に来た場合、その人物がカメラシステムによって未知の人物と判断され、ユーザに Twitter でアラートメッセージが投稿される。複数のユーザの内の 1 人が BoT アカウント（Raspberry Pi からなるカメラデバイスのアカウント）へのダイレクトメッセージでツイートを共有すると、ユーザの元へ BoT アカウントから応答を受け取る事が出来る。応答に対して、人の名前を入力すると、共有されたツイートに添付されている画像の人物が新たに AWS 上にある顔認証プログラムのデータベースに追加される。

この登録方法の利点とはユーザにとって難しい動作はなく、まるで会話をする様にシステムを管理する事が出来る点である。

Vol.2019-SLDM-189 No.15
Vol.2019-EMB52 No.15
2019/11/14

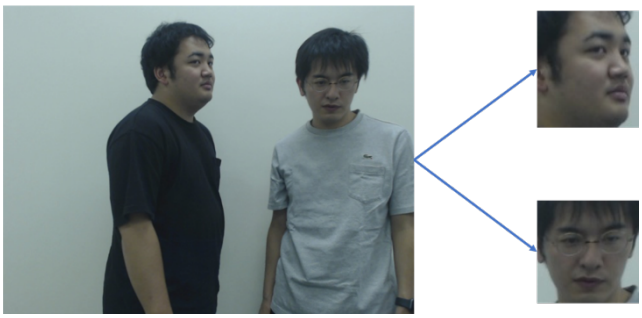

図 9　複数人が写っている場合のトリミング

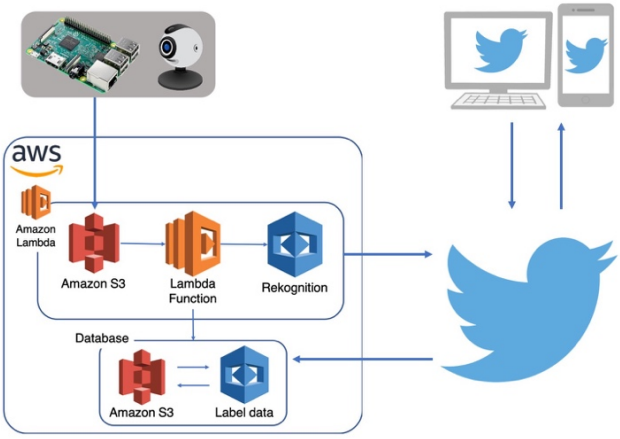

図 10　顔の登録を行う概要図

4. システムの動作の流れ

(1) Raspberry Pi での顔検出

Raspberry Pi3、赤外線動的センサ、Web カメラからなるカメラデバイスはドアに向かって設置されている。Web カメラ、赤外線動的センサはそれぞれ USB、GPIO で接続されている。その様子を図 8 に示す。プログラムを実行している状態はデフォルトで待機状態である。その状態でドアから人が侵入すると、赤外線動的センサが反応する。センサが反応すると写真を撮影するプログラムが実行される。そのプログラムでは、0.5 秒間隔で 10 枚の画像が連続して撮影される。その後、撮影された 10 枚の写真は順番に 1 枚ずつ顔が映っているか検証される。その検証には OpenCV の Haar-like 特徴分類器の中で haarcascade_frontalface_alt.xml という正面の顔を検出する特徴器を使用する。撮影された写真から顔が検出された場合、その顔の部分のみをトリミングする。

ここでトリミングを行う事はふたつのメリットが存在する。ひとつ目は、ネットワークトラフィックの軽減することであり、ふたつ目は、複数人の顔検出に対応することである。ネットワークトラフィックの軽減に関しては，図 3 の例において，画像のサイズが 86,678 バイトから 25,792 バイトへと減少している。これによって Raspberry Pi3 から

図 11　トリミングを行った時の通知結果

AWS の Amazon S3 へとアップロードする時のネットワークトラフィックが軽減される。図 9 に示す通り、1 枚の画像で複数人検出された場合は、それぞれの顔がトリミングされ、別々の画像ファイルとして保存する。

(2) AWS での顔の検出

トリミングされた画像は AWS の Amazon S3 へ送信される。Amazon S3 は AWS のサービス間の連携に適している。Amazon S3 へファイルがアップロードされると、それをトリガとして、Lambda Function にあらかじめ作成しておいたプログラムの処理が開始する。本研究では、「拡張子が.jpg のファイルがアップロードされる」という事象を、Lambda Function が開始されるトリガとする。また、あらかじめ作成しているプログラムが、顔の認証および Twitter を介しての通知を行う。その流れを図 10 に示す

AWS での顔認証には Rekognition を利用する。Rekognition は Amazon S3 に保存されている画像をデータベースと照合する事で顔の識別を行う。Amazon S3 の画像データと Rekognition のラベルデータをリンクさせているものをデータベースと呼んでいる。Lambda のプログラムは、顔の認証結果として名前と一致率を、Twitter の API を利用して Twitter のサーバへその情報を送信する。

(3) Twitter での通知

Twitter のサーバに送信された Rekognition の識別結果は画像と共に Twitter のタイムラインに表示される。複数人の顔が写った図 9 に対する顔認証の結果を図 11 に示す。図 11 では「知らない人」と結果が表示されているが、図 6 のようにあらかじめ顔を AWS のデータベースに登録行っていると、名前とその一致率を表示する。

ユーザはあらかじめ、顔の識別結果がツイートされる BoT アカウントをフォローしておく必要がある。一度、フォローすれば、随時 Raspberry Pi が顔を検出し AWS へ送信される度にツイートされた識別結果の通知を受け取る事が出来る。

(4) Twitter での顔の登録

Vol.2019-SLDM-189 No.15
Vol.2019-EMB52 No.15
2019/11/14

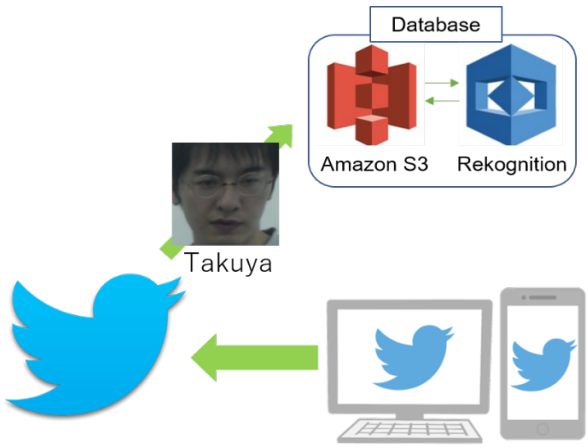

図 12　顔の登録を行う概要図

表 1　リアルタイム性の評価[秒]

	Raspberry Pi の処理	AWS の処理	Twitter の通知	合計
1 回目	21	2	2	25
2 回目	24	1	1	26
3 回目	17	2	1	20
4 回目	18	1	1	20
5 回目	19	1	1	21
6 回目	24	1	2	27
7 回目	19	1	1	21
8 回目	16	1	1	18
9 回目	19	1	2	22
10 回目	22	1	2	25
平均	20	1	2	23

Twitter で表示された Rekognition の識別結果が誤検出、または、未検出の場合、Twitter のダイレクトメッセージで顔の登録を行うことができる。図 11 のような BoT アカウントからの画像付きツイートを、BoT アカウントに対して、ダイレクトメッセージで共有する。BoT アカウントはダイレクトメッセージを受け取ると、「名前を入力してください」と人物名の入力を促すメッセージを返信する。それに対して、ユーザは名前を入力する。ここで入力された名前と共有された画像のデータは、AWS 上の Amazon S3 と Rekognition からなるデータベースへとアップロードされる。画像は Amazon S3 へ保存され、名前はラベルデータとして Rekognition へ保存される。これらの処理も、顔の識別時と同様に、あらかじめ作成している Lambda Function 上のプログラムにより行われる。顔登録の流れは図の 12 に示す。

データベースに保存されると、以降の Rekognition での識別ではダイレクトセージで共有された画像との照合結果がツイートされるようになる。

5.　リアルタイム性の評価

本章では、開発したセキュリティカメラシステムのリアルタイム性について議論する。

本システムの主な流れは、主として、以下の 3 つの工程に分けられる。

- Raspberry Pi3 上での撮影、人検出、AWS への送信
- AWS 上での顔認証、Twitter サーバへの送信
- Twitter サーバからのユーザへの通知

なお、顔の登録に関してはリアルタイムの重要性は薄い事からこの章では割愛する。

5.1　カメラデバイスでの処理時間の評価

ここでの工程は更に以下の 4 つに細分化される。

- 侵入者に対して画像を 10 枚撮影
- 10 枚の画像の OpenCV による顔検出
- 顔画像のトリミング
- Amazon S3 へトリミングされた画像のアップロード

4 つの工程を 10 回繰り返し、その平均の算出結果を表 1 にしめす。前提条件として、撮影された 10 枚の画像の中で前半の 5 枚の中に顔が写っているとする。平均時間は 20[s]以下ということが分かる。

5.2　AWS から Twitter までの処理時間の評価

この工程には更に以下の 2 つに細分化される。

- Rekognition での顔識別
- 識別結果を Twitter にアップロードする

処理時間の測定は、カメラデバイスから S3 に画像ファイルがアップロードされてから、Twitter へ識別結果のアップロードが完了するまでとしている。この処理はすべて AWS の Lambda Function で行われる処理である。10 回の測定結果は表 1 に示す。AWS は非常に計算能力が高い事が測定結果からも伺える。10 回の平均処理時間は 1[s]以下であることが分かる。

5.3　Twitter の通知までの処理時間の評価

この工程には以下の 1 つのみ存在する

- Twitter のタイムラインに表示

Twitter のタイムラインに表示させるのは識別の結果と識別に使用された画像である。なお、AWS での Log データと Twitter で投稿された時間から処理時間を測定している。他と同じく、10 回の平均を算出している。平均処理時間は 1[s]以下となっている。

5.4　合計値での評価

5.1 章から 5.3 章までの工程の中で AWS、Twitter で行われる処理は平均値がどちらも 1 秒未満であり、非常にリアルタイム性に優れていると言える。これはネットワークトラフィックを軽減するための、トリミング処理の影響も大きいと考えられる。

しかし、Raspberry Pi 上での平均処理時間は 20 [s]以下で

Vol.2019-SLDM-189 No.15
Vol.2019-EMB52 No.15
2019/11/14

ある。本研究のシステムでのリアルタイム性という点において、ボトルネックとなっている。

6. まとめ

本稿では、開発したホームセキュリティカメラシステムについて説明を行った。 本研究のカメラシステムは、UIとして世界的に人気のある Twitter を採用している。これはつまり、カメラシステムを Android スマートフォン、iOS スマートフォン、Windows PC、Linux PC などのさまざまなデバイスで利用する事が可能となる。ユーザは、カメラシステムを監視および制御するために、デバイスにアプリケーションソフトウェアをインストールする必要がない。またセキュリティカメラシステムにおいて顔認証とは重要な点の１つである。本研究では顔認証に、AI に基づいた AWS の Rekognition を利用している。

今後の計画として、LINE、Weibo、Facebook などの他の SNS のシステムの拡張を予定している。これらの SNS は、一部の地域では Twitter よりも人気がある為である。また、リアルタイム性の向上と検出精度の向上を行うために OpenCV ではなく YOLO というライブラリの導入を行う予定である。YOLO による機械学習を利用し、10 枚の写真を撮影せずとも顔の検出を行う事を目標としている。

参考文献

[1] P. Ramaswamy, “IoT Smart Parking System for reducing Green House Gas Emission.” In *Proc. of International Conference on Recent Trends in Information Technology* (ICRTIT), 2016.

[2]V. Patchava, H. B. Kandala, P Ravi “A Smart Home Automation technique with Raspberry Pi using IoT.” In *Proc. of International Conference on Smart Sensors and Systems* (IC-SSS), 2015.

[3]S. Purbaya, D. W. Sudiharto, C. W. Wijiutomo “Design and implementation of surveillance embedded IP camera with improved image quality using gamma correction for surveillance camera.” In *Proc. of International Conference on Science and Technology - Computer* (ICST), 2017.

[4]C. M. Parmar, P. Gupta, K. S. Bharadwaj, S. S. Belur “Smart work-assisting gear.” In *Proc. of IEEE 4th World Forum on Internet of Things* (WF-IoT), 2018.

[5]A. Saba, Nagarathna “IoT based energy efficient security system.” In *Proc. of 3rd International Conference on Applied and Theoretical Computing and Communication Technology* (iCATccT), 2017.

[6]H. R. Lee, C. H. Lin, W. J. Kim “Development of an IoT-based visitor detection system.” In *Proc. of International SoC Design Conference* (ISOCC), 2016.

[7]I. Aydin, N. Adnan, Othman “A new IoT combined face detection of people by using computer vision for security application.” In *Proc. of International Artificial Intelligence and Data Processing Symposium* (IDAP), 2017.

[8]S. A. I. Quadri, P. Sathish “IoT based home automation and surveillance system.” In *Proc. of International Conference on Intelligent Computing and Control Systems* (ICICCS), 2017.

[9]A. Mordvintsev, *Haar Cascades を使った顔検出,* 2013. http://labs.eecs.tottori-u.ac.jp/sd/Member/oyamada/OpenCV/html/py_tutorials/py_objdetect/py_face_detection/py_face_detection.html [Accessed 2019].