Raspberry Piとカメラを用いた社内郵便物通知システム
―小型ボードコンピュータと画像処理の応用―

ICT営業本部
佐々木　達也

概要

組み込み用途のMPUの発展に伴い、手のひらサイズの安価なコンピュータが各種販売されており、その処理能力の向上も著しい。消費電力が小さく、簡単な画像処理を行う程度の計算能力を持つに至っているため、カメラを接続して様々な応用が可能である。それらコンピュータの中でも最もユーザー数が多いものの一つであるRaspberry Piと画像処理技術を用いて、筆者は簡単な実務を支援するシステムを構築した。社内宛ての郵便物が集まる棚を一定の時間間隔で撮影した画像について、区画ごとに状態の変化を時系列で分析し、変化が検知された際に担当者に電子メールを送信して通知するシステムである。検知精度を向上させる工夫や様々なノイズへの対策、さらにRaspberry Piを安定して動作させるための工夫などをまとめる。

1. はじめに

社内宛ての郵便物が到着したというイベントを自動検知するシステムを構築するに至った経緯は以下のとおりである。

弊社では、社内宛ての郵便物は一度メール室と呼ばれている部屋に集まり、宛先部署に応じて区切られた棚の中に一時的に格納される。その様子は図1に示すとおりである。

一日に一回、この棚の中の郵便物は配達業務担当者によって取り出されて宛先の各フロアへと配送されている。また、庶務担当者はいつでも自らが担当する部署宛ての郵便物をメール室の棚から持ち帰ることができる。

庶務担当者は実際にメール室に行って棚を目視確認するまでは、取り出すべき郵便物が有るか無いかを知る方法がなかった。弊社は17階建てのビル内に執務フロアが展開されているため、特に高層階の庶務担当者にとって、確認作業のわずらわしさが問題となっていた。

最初の解決策として、Webカメラで棚を撮影して社内LANで公開することにより、どのフロアからもリアルタイムで棚の状況を把握可能にした。これにより、郵便物の効率的な配送が可能になった。

図1　メール室の棚と社内宛て郵便物

しかし、Webカメラによる画像配信でも、庶務担当者がWebブラウザを開いて棚画像を目視確認するまでは、郵便物の有無が分からないという欠点があった。そこで、郵便物が棚に入ったというイベントを検出して、庶務担当者にメールで通知するというシステムの構築を行ってみた。本論文では、この郵便物到着自動検知システムの構成とその検知精度、さらに実際に運用してみて判明した様々な外的ノイズへの対策について報告する。

2. 郵便物到着検知方法の検討

郵便物到着検知に画像処理の利用を選択した根拠を述べる。

郵便物が棚に置かれたことを検知する方法として以下の二つの方法を検討した。

- 棚に物理的なセンサを設置する方法

● 画像処理を利用する方法

物理的なセンサを設置する方が確実な検知を期待できるが、今回は監視区画の数が36ヵ所もあり、すべての区画にセンサを設置するのは現実的ではなかった。そこで、カメラを設置するだけで実行可能な画像処理を利用する方法を採用した。

3. システム構成

郵便物到着検知システムを構成するハードウェアとソフトウェアの各コンポーネントについて説明する。

3.1 小型コンピュータRaspberry Pi

メール室には様々な荷物が出入りし、PCなどの機器を設置する空間的余裕はなかった。そこで、画像取得・画像処理・イベント検知・メール通知までをすべて実行可能であるような、なるべく小型のコンピュータが必要であった。Raspberry Pi[1]はまさにそのような要件を満たすコンピュータの一つである。

Raspberry PiのOSには、豊富なソフトウェア資産を有するRaspbian[2]を選択した。RaspbianはDebian LinuxをRaspberry Pi用にカスタマイズしたOSで、Debian Linuxで利用可能なパッケージの多くを利用できる。

3.2 カメラ

検知性能を比較する目的で、二種類のカメラを検証した。

3.2.1 Raspberry Pi専用カメラボード

Raspberry Piのために開発されたカメラボードで、最大で2592×1944の静止画像の取得が可能である。Raspbian OSでは簡単な前準備で動画および静止画の取得が可能である。

3.2.2 USBカメラ

一般的なUSBカメラの一例として、実売価格が千円程度と安価で、最大1280×720の静止画像の取得が可能なLogicool HD Webcam C270を選択した。

Raspbian OSでは追加のドライバなどは不要で、そのままRaspberry PiのUSBポートに接続するだけで動画および静止画の取得が可能である。

3.3 ソフトウェア

システム構成に用いた主なソフトウェアは以下のとおりである。

3.3.1 raspistill

Raspberry Pi専用カメラから静止画を取得する。

3.3.2 fswebcam

USBカメラから静止画を取得する。

3.3.3 OpenCV[3]

OpenCV固有の機能はほとんど用いておらず、画像のピクセルデータの取得に利用した程度である。

3.3.4 郵便物到着検知プログラム

詳細については次章で述べる。

4. 郵便物到着検知プログラム

郵便物の到着を検知するプログラムの手法と構成を述べる。

4.1 郵便物到着検知の概要

図1に示したように、カメラの映像には棚の全体が収められている。棚の中の各区画を本論文ではボックスと呼ぶことにする。一番右上のボックスは5階の○○部署、その下のボックスは6階の××部署といったように、各ボックスは社内の部署にそれぞれ割り当てられている。

本システムでは15秒に一回、このカメラからの最新画像を取得して郵便物到着検知処理を行う。あるボックスに郵便物が入ると、当然画像上の相当する領域には変化が起こり、到着以前の画像とは異なる画像が取得される。この画像の差異を機械的に検出することで、郵便物到着の検知が可能となる。

ただし、これだけでは、①空のボックスに郵便物が入った変化(IN)、②既に郵便物の入っているボックスに新たな郵便物が入った変化(UPDATE)、③ボックス内の郵便物が回収されて空になった変化(OUT)を区別することができない。とくに今回の要件では、③のOUTについては通知を上げぬよう明確に区別する必要がある。このため、最新状態における郵便物の有無も加えて判定することとした。郵便物有無の判定は、空の状態のボックスを撮影したテンプレート画像を用意しておき、それと最新画像との差異を検出することで行う。

上記の処理は、各ボックスについて個別に行われる。つまり、取得した画像から各ボックスに相当する領域を切り出したのちに、ボックスごとに変化や差異の検出を行い、各種判定、到着時の通知までを行うこととなる。

なお、以降この論文では、上記のIN、UPDATE、OUTすべてを含めたボックス内の郵便物の状態の変化を郵便物変化と呼び、郵便物到着(IN、UPDATEを指し、OUTは含めない)と区別する。

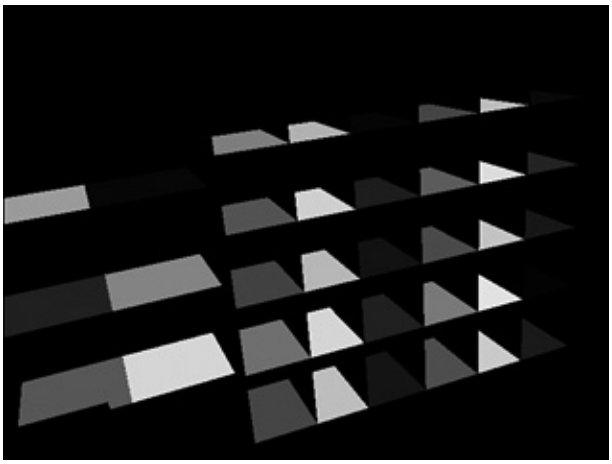
4.2 各ボックス監視領域の切り出し

まずは画像の中の各ボックスの監視領域を切り出す処理が必要である。エッジ抽出の後に自動的に四辺形を認識するなどの方法も考えられるが、今回はより簡便な手法を用いた。

まず、図1の棚の画像をもとにして図2のようなマスク画像を作成する。

図はモノクロでやや分かりにくいが、黒を背景とし、各ボック

図2　マスク画像



スの底面に相当する部分をそれぞれ異なる色で塗りつぶした画像がマスク画像である。マスク画像はレイヤー編集機能を持つ一般的なドローソフトで簡単に作成することができる。

このマスク画像をプログラム開始時に読み込み、各ボックスに応じて、色分けされた各領域のピクセル位置の集合を紐付ける。例えば、一番右上のボックス底面を(r, g, b)成分が(0, 0, 200)の色で塗りつぶしていたとすると、処理時には、マスク画像上で(0, 0, 200)の色情報を持つピクセル位置の集合を参照することで、このボックスの監視領域の切り出しを行うことができる。

4. 3 画像間の差異検出

2画像間の変化検出には様々な計算手法が存在するが、今回採用したのはZNCC(Zero−mean Normalized Cross Correlation:　ゼロ平均正規化相互相関)[4]である。本論文ではZNCCによる画像 **P**, **Q**間の相関係数を$R(\mathbf{P},\mathbf{Q})$と表記する。$R(\mathbf{P},\mathbf{Q})$の定義を式(1)に示す。

式(1)　$R(\mathbf{P},\mathbf{Q})$

$$R(P,\ Q)=\frac{\sum_i\left\{(P(i)-\overline{P})(Q(i)-\overline{Q})\right\}}{\sqrt{\sum_i\left\{(P(i)-\overline{P})^2\right\}\cdot\sum_i\left\{(Q(i)-\overline{Q})^2\right\}}}$$

各記号の意味は以下のとおりである。

- $P(i)$:　画像 **P**の位置iのピクセルの輝度
- $Q(i)$:　画像 **Q**の位置iのピクセルの輝度
- $\overline{P}$:　監視領域全体での、画像 **P**の輝度の平均値
- $\overline{Q}$:　監視領域全体での、画像 **Q**の輝度の平均値

この計算は、各ボックス監視領域のピクセル集合についてそれぞれ行う。つまり、各ボックスに対してそれぞれ$R(\mathbf{P},\mathbf{Q})$が計算される。なお、監視領域をそれぞれのボックスについて個別に切り出す手法は4. 2節で述べたとおりである。

式(1)の$R(\mathbf{P},\mathbf{Q})$の値は、監視領域において画像 **P**と画像 **Q**が完全に一致する場合に1となり、相関が全く無ければ0、ネガとポジがちょうど反転した関係ならば−1となる。ZNCCを用いることにより、領域全体の輝度平均が考慮され、全体的な明るさの変化がこの数値に及ぼす影響を抑えることができる。実際にシステムを運用してみると、画像の全体的な明るさ変化は頻繁に起こり、ZNCCが効果的にその影響を除去することがわかった。

4. 4 郵便物到着の検知

新しい画像を取得するごとに、各ボックス監視領域について、最新時刻における画像 **I**(n)と、前回取得時の画像 **I**(n−1)を比較した相関係数$Rseq=R(\mathbf{I}(n),\mathbf{I}(n-1))$を算出する。

原理的には、$Rseq$の値が1に近ければ、2画像間で変化がない=郵便物変化は生じていない、逆に値が1よりある程度小さければ、画像的に変化が生じた=郵便物変化が起きた、と判定することができる。ただし、これは画像上の変化をすべて郵便物変化によるものと仮定した場合の話であり、実際には郵便物をボックスに入れる人物などの障害物による変化であることが多い。そのため、後に述べるような、映り込みによるノイズを除去する仕組みが必要となる。

4. 5 テンプレート画像の用意

最新状態における郵便物の有無を検出するために、すべてのボックスを空にした状態のテンプレート画像を用意する。

図3がテンプレート画像である。検知精度を向上させる目的で、各ボックス底面には青い布テープを線状に貼りつけて、マーカーとして利用している。

4. 6 郵便物有無の検知

郵便物変化の検知と同様、画像を取得するごとに、各ボックス監視領域について、最新画像 **I**と、テンプレート画像(以下、記号 **O**)を比較する。$R(\mathbf{I},\mathbf{O})$の値が1に近ければ、**I**が**O**

図3　テンプレート画像

とほぼ同じ画像である=郵便物がない、逆に値が1よりある程度小さければ、**I**が**O**と画像的に異なる=郵便物がある、と判定することができる。

郵便物有無の検知に関しても、郵便物変化の検知と同様に、障害物の映り込みによるノイズが発生する。しかし、実際に郵便物有無の判定結果が必要となるのは変化が検知された瞬間のみである。郵便物変化の判定の段階でノイズが除去されていれば、変化を検知した時点での映り込みは起こっていないものとしてよいので、ここで改めて考慮する必要はない。

4.7 ノイズとその対策

メール室には様々な人物が出入りし、カメラの視野の中に映り込む。

図4は映り込みの起こった画像の一例で、作業中の人物が中央付近のボックスを覆い隠してしまっている。この映り込みによって、中央付近のボックスの*Rseq*は大きく減少する。このような映り込みによる数値の変動を郵便物の変化とは見なさぬよう、ノイズとして無視する仕組みが必要である。

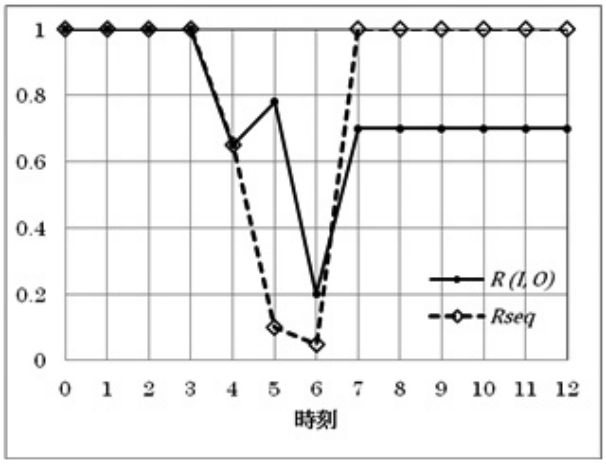
図4　カメラ視野内への人の映り込み

プログラム内で実際に用いたノイズ除去アルゴリズムはやや複雑なので、ノイズ除去の基本的な考え方を以下に示す。

図5はあるボックスの*Rseq*と$R(\mathbf{I},\mathbf{O})$の時系列変化の一例である。横軸は15秒間隔の測定時刻tを表し、縦軸はRの値を表す。時刻t=0から時刻t=3までは*Rseq*、$R(\mathbf{I},\mathbf{O})$ともにほぼ1で、この間はボックスが空のまま変化しなかったことを示している。時刻t=4から時刻t=6まではともに激しい変動があり、これは人が棚の前で作業をしていることによる。時刻t=7から先は値が安定し、*Rseq*は再び1付近に、$R(\mathbf{I},\mathbf{O})$は1よりも低い値になった。これにより、監視領域の画像の変動が落ち着き、郵便物が入っている状態になったと判断できる。

すなわち、このような変動が画像に起こっている間は郵便物変化の判定を保留すればよい。変動の有無は*Rseq*の継続的な値をもとに、画像取得のたびに判定する。変動が始まる

図5　*Rseq*と$R(\mathbf{I},\mathbf{O})$の時系列変化例



前の画像 **Before**をメモリ上に保持しておき、変動が落ち着いた後の画像 **After**との比較$R(\mathbf{Before},\mathbf{After})$を*Rseq*の代わりとして用いることで、映り込みによるノイズを除いた郵便物変化の検知を実現している。

このノイズ除去の仕組みのため、郵便物がボックスに入ってからメールによるアラートが届くまでにタイムラグが発生することは避けられない。しかし、発生するタイムラグは高々数分程度なので、実用上問題となることはない。

4.8 各判定に用いる閾値

- 郵便物変化：$R(\mathbf{Before},\mathbf{After})$が1に近いか否か
- 郵便物有無：$R(\mathbf{I},\mathbf{O})$が1に近いか否か
- 映り込みを考慮するための画像の変動の有無

などの判定にあたり、いくつかの閾値を用いている。

例えば、$R(\mathbf{I},\mathbf{O})$の値が1に近いか否かを分けるための閾値として、$D=0.92$を設定している。4.6節で述べたように、$R(\mathbf{I},\mathbf{O})\geqq D$ならば郵便物無し、$R(\mathbf{I},\mathbf{O})<D$であれば郵便物有りと判定される。

各閾値を決定するには、実際に様々な状態の画像を用いて各ボックスの$R(\mathbf{I},\mathbf{O})$、*Rseq*などを計算し、実態と比較しながら最適な値を探さねばならない。

こうして求めた閾値の組を、すべてのボックスで共通に用いており、現在のところ良好な結果を得られている。監視領域の大きさや監視ターゲットの性質が著しく異なるような場合は、ボックスごとに個別の閾値を設定する必要が生じるかもしれない。

5. カメラ種別による検知率の比較

3.2節で述べたとおり、二種類のカメラでシステムを稼動させた。カメラによって郵便物変化の検知率に差が生じたか否か、結果を示す。

5.1 検知率の定義

信号検出理論の標準的な定義に従って、システムの出力を表1のように四つの事象に分類して集計した。

表1　システムの出力結果の分類

	システムが郵便物変化を検知した	システムが郵便物変化を検知しなかった
実際に郵便物が変化した(信号あり)	ヒット(A)	ミス(B)
実際には郵便物変化がない(信号なし)	フォールスアラーム(C)	コレクトリジェクション(D)

ここで、ヒット率とフォールスアラーム率を、表1の記号A～Dを用いて以下のように定義する。

ヒット率=Aの判定数/(Aの判定数+Bの判定数)

フォールスアラーム率

=Cの判定数/(Cの判定数+Dの判定数)

ヒット率は正しく郵便物変化を検出できた割合で、フォールスアラーム率は郵便物変化が生じていないにも関わらず誤って検知してしまった割合である。

5.2 結果

ある一日のシステムの検知結果について、二種類のカメラのヒット率とフォールスアラーム率を表2に示す。これはすべてのボックスの結果を総合したものであり、各結果を変化種別(IN, UPDATE, OUT)によって分類した。この日の郵便物変化の総件数は、3種の変化を合わせた224となる。

表2　結果

※括弧内は実際の発生件数	Raspberry Pi専用カメラ		USBカメラ	
	ヒット率	フォールスアラーム率	ヒット率	フォールスアラーム率
IN(46)	95.6%	0 %	100%	0 %
UPDATE(127)	95.3%	0 %	94.49%	0 %
OUT(51)	100%	0 %	100%	0 %
合計(224)	96.4%	0 %	96.8%	0 %

結果からわかるとおり、カメラ種別による検知性能の差は見られなかった。各種判定用の閾値はすべてRaspberry Pi専用カメラからの画像を用いて設定したものであったが、同じ値の組をそのまま使ってUSBカメラでも同様に良い結果が得られたのは予想外であった。

なお、今回、INをUPDATEと誤判定するなどといったフラグ種別の誤りは発生しなかった。

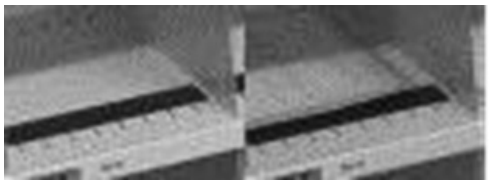
5.3 郵便物検知の失敗例

表1のBに分類された誤りの例を以下に示す。

5.3.1 マーカーを覆い隠さない例

図6は、変化前(左)から変化後(右)で大きな封筒状の郵便物がボックスに入ったにも関わらず、検知できなかった例である。図からわかるとおり、到着した郵便物がマーカーを覆い隠さなかったために、画像として変化が小さく、郵便物変化と見なされなかった。この種の誤りを防ぐ方法は以下の二つが考えられる。

図6　マーカーを隠さない郵便物



- マーカーを覆い隠すように郵便物を置く
- マーカーを増やす

5.3.2 輝度変化が小さい例

図7は、変化前(左)から変化後(右)で小さな封筒状の郵便物がボックスに入ったにも関わらず、検知できなかった例である。輝度の変化が小さく、人間が見ても差が分かりにくい変化で、この種の誤りを防ぐのは困難である。

図7　輝度変化が小さい郵便物

6. システム安定稼動の工夫

本システムを安定して稼動させるには注意すべき点がある。ここではRaspberry Piとカメラのそれぞれについて注意点をまとめる。

6.1 Raspberry Piの安定稼動

Raspberry Piを安定して長期間動作させるには以下の事項に特に注意する必要がある。

6.1.1 十分な電力を供給できる電源

Raspberry Piには専用の公式電源が存在せず、ユーザーが独自に用意することになる。Raspberry Piの公式ページによると、最低でも700mAの電流を供給できる電源が必要とされている[5]。定格電流が1A以上、できれば2Aの電源を用いると安定した動作を期待できる。USB2.0規格に準じた500mA給電の電源製品が多いので選定時には注意が必要である。

筆者は定格2Aの電源を用いたが、専用カメラボードを接続しても、USBカメラを接続しても、特に動作が不安定になることはなかった。ただし、両者を同時に接続した際に不安定な挙動を示したことがあったため、同時接続が必要な場合は外部給電が可能なUSBハブを介して接続するなどの対策が必要である。

6.1.2 SDカードへの書き込みの抑制

Raspberry PiはSDカードにOSをインストールして動作させる。SDカードには書き込み回数の上限があり、上限を超えて書き込むとファイルが破損する可能性がある。よって、なるべく無駄な書き込みを行わないことが長期の安定運用につながる。

本システムでは15秒に一回カメラ画像を取得して保存しておく必要があるため、少なくとも15秒に一回の書き込みが発生する。これをそのままSDカードに書き込むとSDカードの寿命を縮めてしまう。そこで、Linuxのtmpfs（仮想メモリファイルシステム）[6]を利用する構成に変更した。これは、メモリ上に仮想的なファイルシステムを作成する仕組みで、アプリケーションからは通常のファイルシステムと区別せず使えるが、書き込みはSDカードではなくメモリ上だけに行われる。筆者が用いた画像データは640ピクセル×480ピクセルのカラー画像で、およそ300KBであったのでtmpfsとして1MB程度を確保すれば十分画像を保持できた。Raspberry Piのメモリは512MBあるので、tmpfsとしての減少分は問題を引き起こすものではない。tmpfsを利用する構成で、現時点で6ヵ月以上の安定稼動が続いている。

さらに発展させ、aufsとfsprotectを利用して、SDカードをリードオンリーで運用するという手法もあるが、本システムでは通知用のメールアドレス変更などを容易に行う目的で、リードオンリーの運用は行わなかった。

SDカードの劣化に備えて、定期的にバックアップを取得しておく必要があるのは言うまでもない。

6.2 カメラのずれ対策

本手法ではテンプレート画像として郵便物が存在しない棚の画像を利用している。そのため、カメラの位置や方向がずれると、郵便物が存在しない状態でも各ボックスの$R(\mathbf{I},\mathbf{O})$の値が小さくなり、郵便物が有ると誤って判定してしまうことが増える。これを避けるために、カメラの設置位置と固定方法には注意が必要である。

6.2.1 カメラ設置位置

カメラの設置位置は、人や物が接触することのない、適度に高い場所を選ぶと良い。高い場所に置くことで接触による位置ずれを防ぐとともに、カメラ視野内のボックス底面の視面積が大きくなるという利点もある。ボックス底面の視面積が大きくなると、各種のRを計算する際に用いるピクセル数が増加するため、検知精度が向上する。

6.2.2 カメラ固定方法

カメラを高い位置に設置しても、何らかの衝撃が加わってカメラの位置や方向がずれる可能性は排除できない。そのため、単にカメラを置くだけでは固定が十分とは言えない。重量のあるカメラ保持台などを利用し、さらにその保持台を棚にテープで強固に固定するなどの対策を施すのが望ましい。

7. 課題

本システムの課題をまとめる。

7.1 イベント通知方法

郵便物到着が検知されたというイベントを、すべて電子メールで通知する構成には改善の余地がある。まず、庶務担当者にとって、郵便物の到着をなるべく早く知りたい場合と、そうでない場合がある。知る必要のない状況で通知メールが送られてくるのは無駄で邪魔である。そこで、郵便物の到着を知りたいときだけメールによる通知が届くような、イベント通知登録の仕組みを用意すると良いと考えている。例えば、夕刊が届くことだけを通知して欲しい場合は、夕刊が届くと予想される時間帯にのみ通知されるように登録しておくことで、興味あるイベントに関する通知だけを選択して受け取ることが可能になる。

7.2 カメラ位置ずれへの対応

6.2節でも述べたとおり、本システムではカメラの位置ずれへの対応が不足している。筆者がシステムを運用した際には3回の位置ずれが発生し、その都度テンプレート画像の撮り直しを行い、カメラの固定方法を徐々に改善していった。多少のずれなら自動的に画像を補正して処理を行うような仕組みを組み込めば、より安定したシステムになると思われる。

7.3 閾値の設定作業の簡易化

4.8節で述べたとおり、閾値を決定するためには、2000枚程度の画像を用いて、各種判定の結果が実態と合うような適切な値の組を探す作業が必要である。この作業を簡易化するツールを開発すれば、より簡単にシステムの初期設定を済ませることができるようになると思われる。

8. おわりに

本論文で紹介したシステムの検知ターゲットは棚の中の郵便物であったが、検知手法は郵便物に特化したものではないため、他の検知ターゲットにも適用可能であると考えられる。共有物の有無の検知や、消耗品の欠品検知など、応用の範囲は広い。ハードウェアは安価で簡単に入手でき手軽に始められるので、様々なモノの検知を試みていただきたい。業務

の効率化だけではなく、新たなビジネスのアイディアが生まれるかもしれない。そのような活動に本論文が少しでも役立つことが筆者の願いである。

9. その他技術との関連

本論文では小型ボードコンピュータと画像処理の応用について詳しく議論したが、その他にも以下のような技術について開発・検証を行っている。

・BLE（Bluetooth Low Energy）
・EnOcean（エナジーハーベスティング無線センサネットワーク）
・Intel Edison（無線内蔵超小型ボードPC）

これらは、いわゆるIoT（Internet of Things）の構成要素としていずれも重要なもので、今後の新サービス創出にとって不可欠である。これら技術を応用したサービスについてアイディアがある方は筆者にご一報いただきたい。

参考文献

1）http : //www.raspberrypi.org/（2014年12月1日 現在）
2）http : //www.raspbian.org/（2014年12月1日現在）
3）http : //opencv.org/（2014年12月1日現在）
4）J. P. Lewis : Fast Template Matching, http : //www.scribblethink.org/Work/nvisionInterface/vi95_lewis.pdf（2014年12月1日現在）
5）http : //www.raspberrypi.org/faqs#powerReqs（2014年12月1日現在）
6）Michael Kerrisk : Linux プログラミングインタフェース, オライリージャパン（2012）