［原著論文］

タブレット端末による非接触咀嚼検出アプリの開発

廣瀬　明依，孫　氷玉，宮中　大，早川　吉彦

北見工業大学工学部情報システム工学科　〒090-8507　北海道北見市公園町 165 番地
（2016 年 3 月 22 日受付，2016 年 6 月 18 日最終受付）

Development of contactless chewing detection application using tablet-type PC device

Mei HIROSE, Bing-Yu SUN, Dai MIYANAKA and Yoshihiko HAYAKAWA

Department of Computer Science, Faculty of Engineering, Kitami Institute of Technology
165, Koen-cho, Kitami, Hokkaido 090-8507 Japan
（Received on March 22, 2016. In final form on June 18, 2016）

Abstract : Dysfunctions of mastication and swallowing are caused by the growth in the mixed dentition period, post-operative symptoms in surgical procedures, missing tooth/teeth, simply aging, etc. There is a dietary report on the reduction of food intake due to prolonged chewing. Therefore, we tried to develop a noncontact chewing cycle analyzer, which works on tablet PCs. After a face area is captured, the top far-left pixel was tracked to measure the whole-face transition and two pixels at the captured mouth angle were picked up to measure the optical flow method. The mouth movement in the period of food intake, mastication and swallowing, was recorded. The movement showed kinds of wave forms, which sometimes contained irregular forms. But the compensation using the whole-face movement clearly showed the repeated mastication. Due to the limitation of the calculation loading in tablet PCs, the range of practically measurable fps was 8 to 9 in the maximum 30. We reduced the size of captured image and the number of pixels for the optical flow analysis, and chose the effective edge-extraction filter for the faster calculation. The data captured at 8 fps were enough to record wave forms of the mouth movement. Such non-contact chewing cycle analyzer is of value to evaluate the mastication dysfunction and recovery.
Keywords : Face recognition, Chewing cycle, Pattern matching, Optical flow, Mastication

1．はじめに

咀嚼（そしゃく，chewing,mastication または bite）とは，食物を摂食（せっしょく，food intake）後に嚥下（えんげ，swallowing）に適した大きさにするために噛み砕き，唾液と混ぜる口の働きとされている．これは食事の際に最も主要で，食味・食感を味わう上でも大切な動作である．私たち現代人の食事はいわゆる軟食化が進んでいると言われている．これにより咀嚼回数が減少し，顎関節症や肥満症などの問題が生じているとされる．日本咀嚼学会では，1 口 30 回の咀嚼を目安としている[1]．Smit HJ, et al．は，1 口あたり10回の咀嚼と35回の咀嚼では満腹を感じるまでの食事時間は 2 倍になったが，食事量は減少したと報告した[2]．このことから，体重を減らす方法いわゆるダイエット方法としても咀嚼は注目されている．

一方，人が咀嚼能力を失うにはいくつかの要因が考えられる．乳歯から永久歯への交換期，歯科矯正治療中の患者，歯列において複数の歯の欠損を有する者，顎口腔領域や耳鼻咽喉科領域における外科手術の術後患者，脳血管障害や精神神経障害等の患者および高齢者が挙げられる[3]．また，より一般的に老化も原因と考えられる．このような場合では，ひとつの食塊（bolus）を摂食，咀嚼，嚥下するまでの動作が連続してスムーズに行えるかが日々の食生活に大きく影響する．したがって，一連の機能が解析できると対象者の特徴が把握できると思われる．また，口に入れる食塊の性状によって必要とされる咀嚼回数も変わるため，咀嚼能力回復のためのリハビリテーションのために，食塊の大きさ・硬さを工夫する研究もある[4]．そのような研究にも咀嚼機能解析は利用できると考えられる。

これまでの咀嚼に関する研究では，ビデオで撮影した後にカウントする方法[5]や筋電図を用いる方法[6]などが用いられている．斎藤ら[5]は，ビデオ撮影後の肉眼的観察と左右咬筋に電極を付着させて測定した筋電図のデータを照らし合わせて，咀嚼の回数を測定している．虫本ら[6]は，より多くの電極で咬筋だけでなく小さな口腔周囲筋の筋電図を得て，いわゆる「のどぼとけ」（被験者は男性のみ）に加速度センサーをつけて計測している．どれも計測と解析に時間がかかり，頭頸部あるいは顎口腔領域に何らかの装置を身に着けなければならい．

さて，宮中大らは，web カメラを用いた咀嚼回数のリアルタイム計測システムの開発を試みた[7]．この研究では，ディスクトップ PC を使い，web カメラで撮影した画像に OpenCV ライブラリ[8]の顔検出を利用し，検出された顔領域の座標を用いて鼻や口の各顔器官周辺の領域を指定した．そして，顔の各器官の中で局所的な動きがほぼない鼻領域と，口と顎の領域の 2 つの画像を作成しエッジ抽出などの画像処理を施して行った．咀嚼の検出は，オプティカルフロー・ファンクションを利用し，両領域の画像中における特徴点の移動量の差異を用いて行った．このようにして，リアルタイムかつ非接触で咀嚼の回数を計測するシステムの実現を目指した．

そこで本研究では，非接触でより手軽に咀嚼を検出できるよう，持ち運べる Android タブレット PC 端末でのアプリケーションの開発を目指した．当然，ディスクトップ PC とタブレット PC 端末では，計算処理パフォーマンスが大きく異なる．画像認識で多くの特徴点を得てそれに画像処

理を施す時間はなく，固有の工夫が必要である．そこで，タブレット PC 端末のカメラ画像から OpenCV ライブラリの顔自動検出を利用し，検出された顔画像の座標から口唇領域を指定する．エッジ抽出法とオプティカルフロー・ファンクションを利用して，画像中の特徴点の移動を検出する．下顎の開閉運動によって繰り返される移動の波形をグラフにすることで，咀嚼動作における口唇・口腔領域の移動量とその変化や個人差などを見てとることができるアプリを開発した．

なお，この研究は，北見工業大学における「人を対象とする研究倫理審査」の承認（No. 1008）を得ている．

2. 方　法

2.1　システム構成

Android OS をプラットフォームとするタブレット PC 端末である Nexus 7（ME 571-32G，APQ 8064 QuadCore 1.5 GHz，メモリ 2 GB，Android OS Ver. 4.4.3，Google Inc.）を検証デバイスとし，開発環境 AndroidStudio を使用した．言語は Java と android NDK を用いて C＋＋を使用した．さらに，オープンソースの画像処理・画像認識ライブラリ OpenCV Ver. 2.4.11 for Android[8]とグラフ描画ライブラリ AchartEngine[9]を使用した．Windows PC 上の開発環境 AndroidStudio を使用し，USB インタフェースケーブルでタブレット PC をつないだ．シミュレータ・モードで WindowsPC で仮想実行しながら開発し，実機タブレット PC，Nexus 7 で検証した．

2.2　アプリケーションの流れ

本アプリケーションは，3 つのアクティビティ，「メニュー画面」，「咀嚼の検出」，および「グラフ表示」から構成される（Fig.1）．

2.2.1　メニュー画面

アプリケーションを開始した際に最初に表示するのがメニュー画面である（Fig.1(a)）．「Detection Start」をタップすることで，咀嚼の検出へ移行する．

2.2.2　咀嚼の検出

「咀嚼の検出」画面では，タブレット PC のフロントカメラから得たリアルタイム画像（30 fps）を表示する（Fig.1(b)）．キャプチャ画像に画像処理を施し，咀嚼の検出を行う．画面右下の「Graph」ボタンをタップすることで処理を終了し，グラフ表示へ移行する。ここで得た移動量を記録したリストをここに渡すことになる。

2.2.3　咀嚼波形のグラフ表示

「グラフ表示」画面では，「咀嚼の検出」画面から受け取ったデータをグラフ化し表示する（Fig.1(c)）．横軸にフレーム数，縦軸にピクセル数をとったグラフである．フレーム数は経過時間の指標であり，ピクセル数は口角付近の特徴点の垂直移動量である．なお，ディスプレイ・キャプチャーをカラー反転し，白黒二値化して表示している．

2.3　画像処理の流れ

本アプリケーションにおける処理の流れを（1）～（7）に示す．

（1）顔領域の描画→（2）エッジ抽出→（3）口唇領域の画像作成→（4）特徴点抽出→（5）移動量算出→（6）咀嚼検出→（7）グラフ描画

これらの画像処理と画像認識のプロセスについて説明する．

2.3.1　顔領域の描画

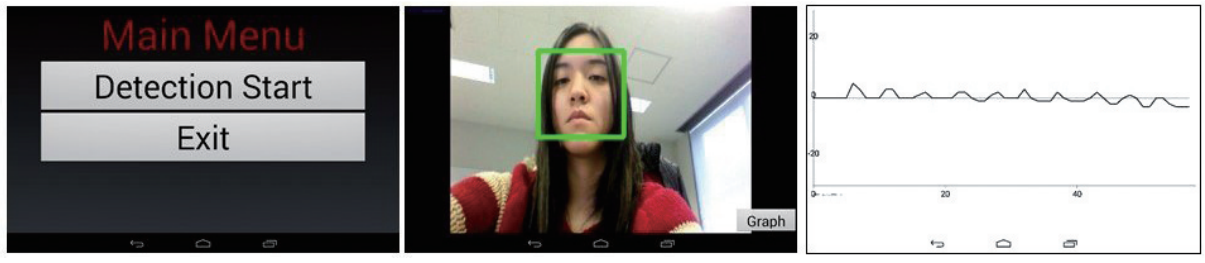
同じ研究室において以前に行われ 2012 年に発表された研究である阿部らの方法を参考に，OpenCV ライブラリのオブジェクト検出を用いて顔検出を行った[10]．この方法では，画像からエッジ，線，中心周辺の特徴量を検出し，テンプレートと類似する特徴量とそうでない領域を分類する．それぞれ異なる特徴量を検出する分類器により領域を順次分類していくことにより，物体を正確に検出することができる．今回は OpenCV の顔のカスケードフィルタファイル「haarcascade_frontalface_alt 2」を用いた．

なお，処理速度を高めるためにキャプチャ用のディスプレイ画素数を 240×320 pixel にして行った．タブレット PC 端末 Nexus 7 の最大の画素数である 1920×1200 pixel で行った場合，約 1.5 fps で動作した。1 秒間に 1 回以上の咀嚼が見込まれるため，この fps ではサンプリングが粗すぎて咀嚼動作を解析できないと考えて，画素数を下げてフレームレートを上げることにした。解像度を下げた場合のフレームレートは，後述するようなそのほかの工夫もあるが 7.84～9.15 fps であった。

顔領域を検出した際の画像を Fig.2(a) に示す．緑色の四角で囲われた部分が検出した顔領域である．リアルタイムで認識するため，顔が動けば領域も移動する．ここでは，この領域の左上座標と幅，高さのパラメータ（Fig.2(a) における矢印で示す．）を得ることができる．顔が 2 つ以上検出された場合はより大きい（つまり，手前に位置する）顔領域をひとつだけ認識するようにした．

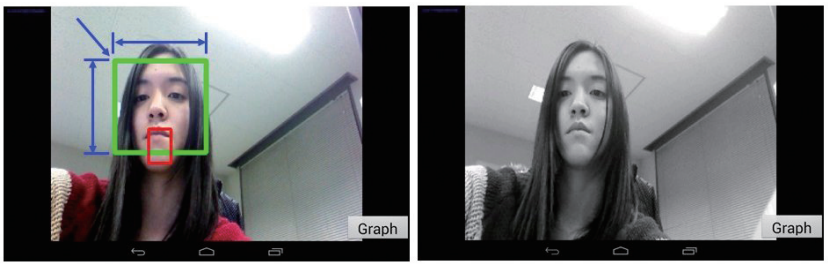
2.3.2　エッジ抽出

特徴点は濃度変化の強い部分に現れる．そこで，エッジ部分を強調するための画像処理を行った．人は咀嚼する際



(a)　(b)　(c)

Fig.1 Display examples at three-activities : menu display (a), front-camera capture for mastication detection (b, detected face area as green square) and graphic display (c, X-axis : elapsed time as the number of frames captured, Y-axis : vertical mouth movement in pixel. The inversion in color and the binarization in black and white were done on the display capture.)



(a) (b)

Fig.2 Face-area detection (a, green square), and the gray-scale image (b). Coordinates of the top-left corner and the height and width of the face area were recorded. Red rectangular area (a) is the mouth area detected, as described in 2.3.3.

に唇が上下に動くので，口部分のエッジの強調を行った．Fig.2(b) のように，グレイスケール画像で処理を行った．OpenCV ライブラリのエッジ強調がグレイスケール画像を対象としているためである．過去の研究では，Laplacian フィルタ（Fig.3(a)）で行った[7]．同ライブラリでエッジ強調に用いられる方法は，他に Sobel フィルタ（Fig.3(b)）と Canny フィルタ（Fig.3(c)）がある[11][12][13]．同ライブラリのデフォルト設定により，3×3 マトリクスで，Sobel フィルタは x 方向と y 方向で行った．

より高速に，はっきりとしたエッジ部分の抽出を行えるように，3 つの処理について検討した．各処理方法による処理速度の違いを Table 1 に示す．処理速度は，「(1) 顔領域の描画」から「(6) 咀嚼検出」までの一連の画像認識・処理を 30 fps の動画に対して施した場合，ここに表した数値まで低下することを示している．ただ単にフィルタの種類の違いのみを反映しているものではない．この処理速度の数値は，開発環境 AndroidStudio のもと，シミュレータ・モードで仮想実行している WindowsPC 上でログを取ることで得られる．この処理速度は，他の処理も同時に行っているためカメラとの距離などに依存して変化する．3 つの処理はどれも処理速度は咀嚼という動作の速度と比べると影響のない速度であった．つまり，咀嚼が検出可能な速度（fps）を実現したものである。

各フィルタにおいて差はあるが，処理速度は 7.84～9.15 fps であった．実機タブレット PC，Nexus 7 上で，特に口唇全体あるいは口角付近におけるエッジの抽出状態に注目して肉眼的に観察すると，Laplacian は抽出が弱く特徴点の抽出も不安定になった．Canny は処理速度が速い印象だが不安定で特徴点が同じ場所で得られない場合があった．Sobel は Laplacian よりはっきりとした抽出ができていた．異なるエッジ抽出フィルタが処理速度と効果にこのような違いを生じさせる根拠は追及していないが，本研究では，より鮮明に抽出される Sobel フィルタを用いることにした．

2.3.3 口唇領域の画像作成

咀嚼の際に大きく動く範囲である下唇から顎の領域を指定し，その部分以外を黒く塗りつぶす処理を行った（Fig.4(a)）．検出された顔領域における口の位置は万人共通としてほぼ決まっているとして座標を指定した．指定した座標を Table 2 に示す．カラー画像における口唇領域は Fig.2(a) における赤い四角のようになる．

2.3.4 特徴点抽出

OpenCV ライブラリの goodFeaturesToTrack を使用し特徴点を抽出しその座標を取得した．本研究では指定する特徴点の数を 2 つとした．処理速度の向上を図るために，特徴点の数は結果的に 2 点まで減らさざるを得なかったが，それでも咀嚼の波形が描かれることを結果に示す．Fig.4(b) では，口の両端部分（口角，oral angle）の特徴点に白い点が打たれている．

2.3.5 移動量算出

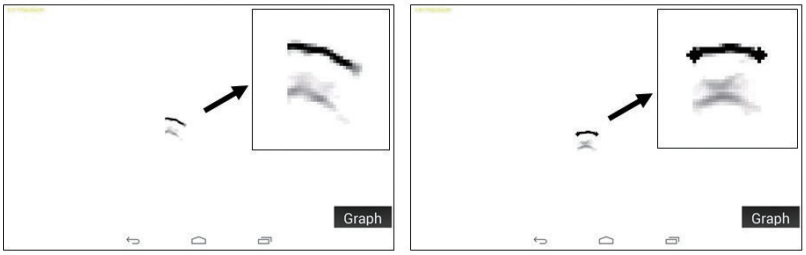
OpenCV ライブラリの calcOpticalFlowPyrLK を使用しオプティカルフローの算出を行う．オプティカルフローとは，時間連続な画像を利用して物体の動きをベクトル集合で表したものである[11, 13]．本研究では，物体の動きをフロー

(a) (b) (c)

Fig.3 Differences in edge-detection filters on the Fig.2(c) image. Images were processed by Edge-enhanced filters such as, Laplacian (a), Sobel (b) and Canny (c). The inversion in color was done on the display capture.

Table 1 Comparison of edge-enhanced filters

Filter	Speed (fps)	Characteristic remark
Laplacian	7.95～8.66	Edge-enhancement is weak. Feature extraction is unstable.
Sobel	7.84～9.15	Edge-enhancement is much clear than Laplacian.
Canny	8.59～9.15	Speed is relatively high, but Feature extraction is unstable.



(a) (b)

Fig.4 A mouth area extracted (a). Two featuring points set at oral angle (b). The inversion in color was done on the display capture. Zoomed images are shown on the top right.

Table 2 Coordinate dimensions to be allocated as a mouth in each recognized and captured face

region	Upper left corner		Lower right corner	
	X-dimension	Y-dimension	X-dimension	Y-dimension
Face	X	Y	X＋W	Y＋H
Mouth	X＋3/8 W	Y＋6/8 H	X＋5/8 W	Y＋9/8 W

ベクトルで表すのではなく特徴点の情報を取得するのを目的としているため，ベクトルではなく座標を返すcalcOpticalFlowPyrLKを利用した．

特徴点の垂直方向の移動量平均を算出する計算式を以下に示す．

$$D_{i,y,f}=P_{i,y,f}-P_{i,y,f-1}\quad(i=1,2)$$

$$D_{i,y,f,ave}=(D_{1,y,f}+D_{2,y,f})/2$$

$P_{i,y,f}$は，口角付近の特徴点iのフレームfにおける y 座標である．$P_{i,y,f-1}$は，そのひとつ前のフレーム$f-1$における y 座標である．したがって，$D_{i,y,f}$はその差，特徴点の垂直方向の移動量になる．特徴点は 2 点だけとなったので，その平均$D_{i,y,f,ave}$は，単にふたつの特徴点の垂直方向の移動量の和を 2 で割った値となる．

2.3.6 咀嚼検出

過去の研究では，口腔領域と同時に鼻領域も保持し，それぞれの動きの平均の差をとることで咀嚼動作と顔の移動の判別をしていた[7]．しかし，Android 端末では，口腔と鼻の 2 枚の画像を処理すると，その速度が遅くなり顎の動きを記録するのに十分な fps 値を得ることができなかった．

そのため，トラッキング・ポイントを絞ることとした。咀嚼動作と顔の移動動作における特徴点（Fig.4(b)）に示す，左右口角の2 点），顔の左上座標，顔の高さの数値（ともに Fig.2(a)）における矢印で示す．幅の値は使用しない）を取得した．顔の左上座標の移動量と顔の高さの変化量の和が±5（−5～＋5）pixel の場合を咀嚼動作，それ以上動いた場合は顔全体の動きとした．グラフで表す場合には，この±5（−5～＋5）pixel を 0 で表す．こうすることで，顔の動きの時にはより大きな振幅の差が出て，咀嚼との違いをはっきりさせることができた．

2.3.7 グラフ描画

フレーム毎の移動量の平均値と顔の移動量を示す値をリストに格納し，撮影終了後にグラフ描画行う（Fig.1(c)）．グラフは AChartEngine ライブラリ[9]を用いて描画した．縦軸は口角付近の特徴点の垂直方向の移動量の平均，横軸は時間ではなく得られたデータの順序で，単位はフレーム数になっている．

図中（Fig.1(c)）の太い黒色の折れ線は，口角付近の特徴点の垂直方向の移動量である．特徴点はふたつになったので，その平均値である．移動量はピクセル数で表している．キャプチャしている画像は 240×320 pixel なので，その中でこれくらい変化していることを示している．図中の細い黒色の線は，「2.3.6 咀嚼検出」に書いたように顔の左上座標の移動量と顔の高さの変化量の和を表していて，その値が±5（−5～＋5）pixel の場合はこの例のようにフラットになっている．このフラットになっている領域が，顔全体の動きが少なく咀嚼動作を捉えているところである．

Android 端末では，画像処理の重さ（必要な計算時間が長いと重い）によって，フレーム毎に fps の値がほぼ 7～9 の範囲で変化する．そのため，一定の時間間隔ではデータが返ってくるわけではない．たとえば，平均 8 fps であるとすれば，横軸の 0～40 は約 5 秒間を意味する．細い黒色の線が連続で 0 を示しているところが顔の動きがないことを表している．

3. 結　果

以上の画像認識・画像処理方法で咀嚼検出を行い，咀嚼の記録・解析を行った結果を以下の図に示す．

Fig.5 は（1）食べ物を口に入れる動作、(2）咀嚼動作、(3）連続での咀嚼動作（10 回）を行った時のグラフである．ここでは，摂食と咀嚼が計測波形の高さの違いで表現できていることが確認できた．細い黒色の線が連続で 0 を

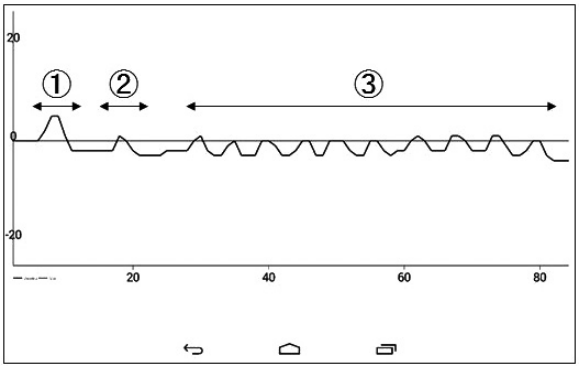


Fig.5 An example of (1) food intake, (2) a single mastication (chewing), and (3) multiple mastication cycle (10 times). The inversion in color and the binarization in black and white were done on the display capture.

示しているので，顔の動きがほぼなく咀嚼のみの動きを捉えていると判断する．

Fig.6 は，咀嚼動作と顔を動かす動作を交互に行ったものである．±5（−5〜+5）pixel を 0 で表すことによって，この細い黒色の線がフラットの部分が顔の動きがなく咀嚼を行っている時間であることを表すことができた。この細い黒色の線を加えることで、咀嚼と顔の動きの判断，区別ができていることが確認できた。

Fig.7 の 4 つのグラフは，実際に食物を食べた際のグラフである．どれも 1 口サイズのチョコレート，おかき，チョコレート菓子である．細い黒色の線が連続で 0 を示しているので，顔の動きがほぼなく咀嚼のみの動きを捉えていると判断する．図中に示した丸数字は，①摂食，②咀嚼，③嚥下動作であると考えられる．それぞれで，ほぼ常にこの 3 つの動作を確認できたが，Fig.7(d) のように，嚥下動作が確認しにくいときもあった。これは，咀嚼しながら飲み込んでいるため，咀嚼と嚥下の違いが判断しにくくなっていると考えられる．

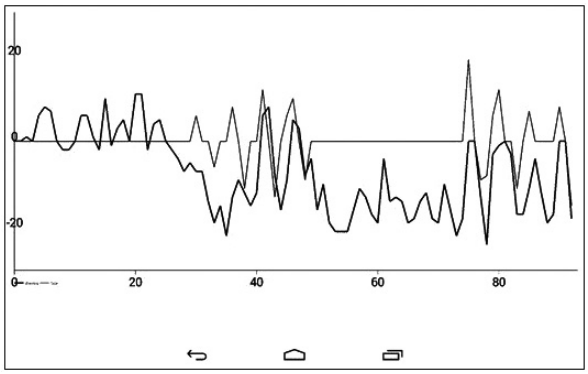


Fig.6 An example of the repetition of mastication (chewing) and entire facial movement. The flat thin gray line shows the mastication period and the waved thin gray line means the facial movement period. The inversion in color and the binarization in black and white were done on the display capture.

4. 考　察

本研究では，Android タブレット端末での咀嚼検出と解析のための波形描出を実現した．当初処理に時間がかかり 1.5 fps 程度でしか解析できなかったが，約 8〜9 fps までの速い処理を実現した．咀嚼のような生体の機能・運動には個人差を起こす様々な要因が考えられ，咀嚼波形を描くための最低限必要なサンプリング（fps）はわからないが，このふたつの値の間にあると思われる．

また，咀嚼の波形を示している図の横軸は，時間を表しているわけではない．8〜9 fps で画像フレームをキャプチャして画像認識・処理を行った結果として，フレーム数をカウントしている．過去に私たちの研究室で行われた研究[11]は，「顔自動認識技術」を出発点にして瞬き（まばたき）波形を描けるようにした．今回は，同じ出発点から咀嚼波形を描いたが，タブレット PC の低いパフォーマンスから特徴点の数がわずかふたつになった．ふたつでも咀嚼波形が描けることを示せたが，ふたつだけならオプティカルフローを使うこともないという議論もあるかもしれない．しかし，タブレット PC のパフォーマンスはより高くなると思われるので，このままの手法の方が将来的に参考になると思われる．

Fig.5 では，摂食と 1 回ないし連続の咀嚼の違いが波形データとして表示できていることを確認できた．Fig.6 からもわかるように，細い黒色の線を加えることで，咀嚼と顔の動きの判断がしっかりとできていることも確認できた．

Fig.7 に示したように，検出された波形のほとんどで，摂食，咀嚼，嚥下の各動作を区別できる波形を確認できた．しかし，Fig.7(d) のように，嚥下の動作が確認しにくい

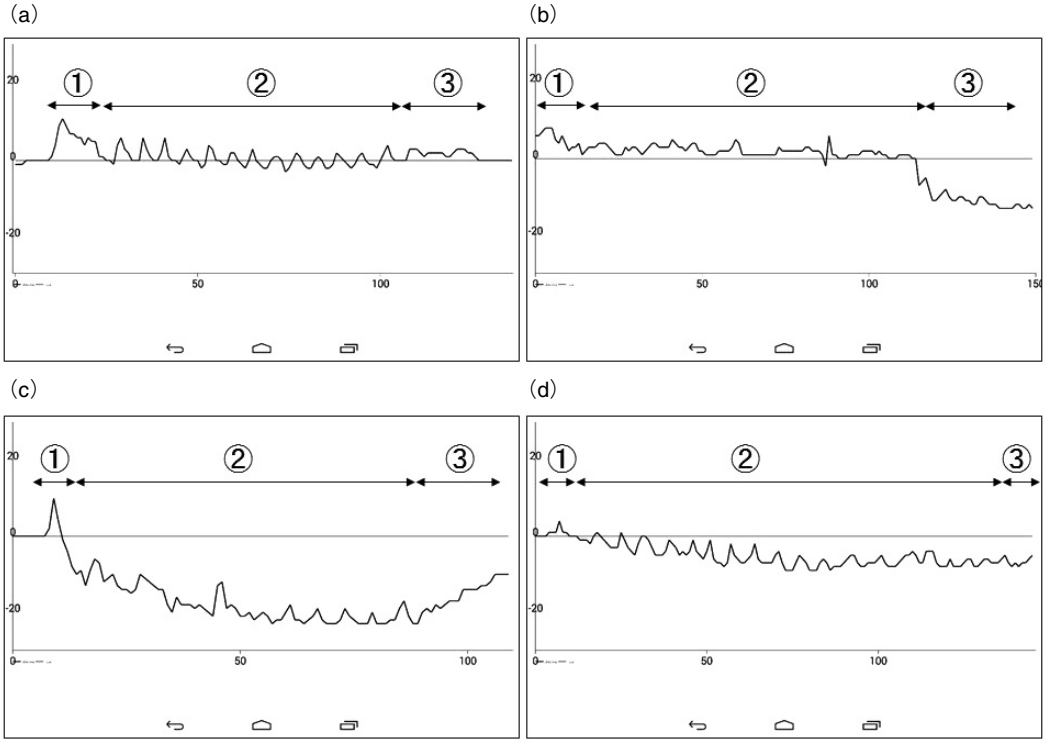


Fig.7 Examples of (1) food intake, (2) mastication (chewing), and (3) swallowing. Foods are, a piece of chocolate ((a) and (b)), a piece of rice cracker (c) and a piece of chocolate biscuit (d). The inversion in color and the binarization in black and white were done on the display capture.

ケースもあった．これは，咀嚼しながら飲み込んでいるため，違いが判断しにくくなっていると考えられる．チョコレートは温かさで溶けてくると口腔内の歯，歯肉（歯齦），舌，軟口蓋などの組織にへばりつき，唾液を絡めてそれを取る際に口唇が動くため，読み取りにくいグラフになることがあった．また，Fig.7 の各グラフのように，摂食時＋方向に大きく動き，咀嚼を始めると右下がり傾向になることが多かった．さらに，嚥下の際は咀嚼時と違う位置で咀嚼より小さい動きをすることが多いことが確認できた．

このようなデータを多く集めることで，個人の癖や個人間の差異，食品の性状に依存した差異などを見てとることができると考えられる．このアプリケーションでは，顔が動いているときに咀嚼はしていないと考えているので，顔をあまり動かさないという条件が必要である．また，顔を動かすと処理速度が遅くなることも確認した．加えて，タブレット端末との距離によって顔サイズが変わり，口唇領域の設定など以降の処理に影響した．今回は，タブレット PC を手に持って腕を伸ばしで，その程度の距離で実現できるシステムを開発したが，顔のサイズや口唇領域の設定など，その場に合わせてカスタマイズするような機能があれば，より正確なデータが得られると思われる．

また，今回はY座標（縦方向）のみの動きの平均をとったが，柔らかいものを食べるとき等は，口を横に動かす（すりつぶす）動作が大きくなるため，X 軸（横方向）の移動量も調べれば，より細かな動作が見えるようになるだろう。さらに，そうすることで読唇などの認識にも応用できるのではないかと考えられる．

歯科医療の専門家や臨床家は，ただ単に垂直移動しているだけではない下顎運動経路，習慣側・非習慣側という左右非対称性，X 線ビデオ撮影（VF：Ｖideo-Fluorography）でわかる舌や咽頭・喉頭部の解剖学的構造の動きとの関係等に注目する必要を訴えるだろう．ビデオで撮影した後にカウントする方法[5]や筋電図を用いる方法[6]は，計画的に行われた臨床的研究であって，ビデオ撮影後の肉眼的観察，左右咬筋やもっと小さな口腔周囲筋に電極を付着させて測定した筋電図のデータ，いわゆる「のどぼとけ」（被験者は男性のみ）に加速度センサーをつけた測定が行われている．

今回の私たちの咀嚼波形が観察できるシステムは，過去に私たちの研究室で行われた研究[11]が，瞬き波形を描けるようにしたのと同じように「顔自動認識技術」を出発点にして開発した．「顔自動認識」は文献[11]で引用しているように，21 世紀になってからの技術である．性別推定，年齢推定，個人認識などの分野で進歩しているが，瞬き波形，瞬きカウンタ，咀嚼波形，咀嚼カウンタのような生体現象の計測・解析に応用した例は他にはないと考えている．

今回はどこにでも持ち運んで測定可能なタブレット PC システムにこだわった．それは，咀嚼機能の維持が脳卒中のリハビリや老人介護で注目されているからである．PC パフォーマンス次第で顔自動認識を出発点とする詳細な摂食・咀嚼・嚥下の初期を解析するシステムが開発できる可能性がある．エッジ検出フィルタについて詳述したが，システム全体の処理速度を律速しているのはそこではなく，画像のサイズや特徴点の数，顔の動きに影響される顔自動認識やオプティカルフローのアルゴリズムと思われる．

結論として本研究は，Android タブレット端末から得た画像から，顔の検出，口唇の検出，特徴点抽出，オプティカルフローの算出，咀嚼検出およびその結果を表示するアプリケーションを開発した．このアプリケーションでは，一口の摂食，咀嚼，嚥下までの動作を波形データとして確認することができた．

参考文献

[1] 日本咀嚼学会：日本咀嚼学会からの発信（http://sosyaku.umin.jp/info/file/info01.pdf），website : http://sosyaku.umin.jp/（2016 年 3 月 18 日アクセス）

[2] Smit HJ, Kemsley EK, Tapp HS, Henry CJ. : Does prolonged chewing reduce food intake? Fletcherism revisited, Appetite 57(1), 295-298, 2011.

[3] 日本障害者歯科学会：障害者歯科とは（http://www.kokuhoken.or.jp/jsdh-hp/html/user/），website : http://www.kokuhoken.or.jp/jsdh-hp/html/（2016年3月18日アクセス）

[4] 日本摂食嚥下リハビリテーション学会：嚥下調整食学会分類 2013
（http:// www. jsdr. or. jp / wp-content / uploads / file / doc / classification2013-manual.pdf），website: http://www.jsdr.or.jp/（2016 年 3 月 18 日アクセス）

[5] 斎藤やよい：ビデオ観察法による食行動に関する研究 ―観察方法と食事摂取スタイル―，民族衛生，61(5)，276-284，1995.

[6] 虫本栄子，田中久敏，古山智成：開・閉口筋筋電図による嚥下動作の評価法，日本補綴歯科学会雑誌，44(2)，292-299，2000.

[7] 宮中大，千葉優輝，早川吉彦：オプティカルフローを用いた咀嚼回数のリアルタイム計測, 電子情報通信学会技術報告，Vol.113, IE2013-13, PRMU2013-6, MI2013-6, 31-34，2013.

[8] 画像処理ライブラリ，OpenCV，http://opencv.org/（2016 年 3 月 18 日アクセス）

[9] グラフ描画ライブラリ，AChartEngine，https://github.com/ddanny/achartengine（2016 年 3 月 18 日アクセス）

[10] 阿部恒介，董建，早川吉彦：瞬き波形を検出するための画像処理による VDT 作業時における瞬き回数の計測，Medical Imaging Technology（日本医用画像工学会雑誌），30(2)，65-72，2012.

[11] 白鳥則郎監修，大町真一郎，陳謙，大町方子，宮田高道，長谷川為春，早川吉彦，加瀬澤正，塩入諭著：画像処理，第 7 章特徴抽出，7.1 エッジ抽出，第 9 章動画像処理，9.1 オプティカルフロー，共立出版，2014.

[12] OpenCV プログラミングブック制作チーム著：OpenCV 2 プログラミングブック，マイナビ，2012.

[13] OpenCV リファレンスマニュアル，エッジ検出（Sobel, Laplacian, Canny）http://opencv.jp/opencv2-x-samples/edge_detection，オプティカルフロー
http://opencv.jp/sample/optical_flow.html#optflowPyr（2016 年 3 月 18 日アクセス）