TouchContext: タッチ操作の挙動分析に基づく人のコンテキスト認識

平部 裕子[1,a)]　荒川 豊[1]　安本 慶一[1]

概要：スマートフォンを用いたコンテキスト認識では，加速度センサや GPS を用いた研究が数多く報告されているが，本研究はそうしたセンサの１つとして，スマートフォン上での「タッチ操作の挙動」を利用することを提案し，その値を取得・分析するための Analayzer の開発に関して報告する．さらに，開発したシステムを用いて分析を行い，タッチ操作の挙動が人のコンテキスト認識へ適用できる可能性について議論する．

TouchContext: Human Context Recognition based on Touch Operation Analysis

Yuko Hirabe[1,a)]　Yutaka Arakawa[1]　Keichi Yasumoto[1]

Abstract: This paper proposes to use “touch operation” for recognizing a human context as one of available sensors in a smartphone. We have developped a system for logging and analyzing all the touch operation on a smartphone. And we report some evaluation results that show a possibility to apply a touch context for recognizing a human context.

1. はじめに

近年，センサーネットワークやスマートフォンの普及に伴い，ユーザのコンテキストを取得しサービスとして還元するコンテキストアウェアなサービスやアプリケーションが次々と提案されている [1][2]．よりアウェアネスの高いサービスを提供するためには，偏在する多くのセンサから様々な情報を取得し，解析した上で，ユーザのコンテキストを認識することが重要である．例えば，最も利用されているコンテキストである「位置情報」は，GPS（Global Positioning System）を利用することで容易に取得可能であるが，GPS を利用できない屋内向けに，WiFi を用いた屋内位置推定手法の研究が数多くなされている．そして，近年では，より抽象的なコンテキストの取得に関する研究が進んでいる．例えば，歩く・走る・座る，など「動作情報」を，加速度センサを用いて認識する研究 [4] や，眼球動作に基づいて笑顔を認識する研究 [5] などがある．

このような背景のもと，本研究では，身近にありながらこれまで利用されてきていなかった新しい情報源として，「タッチ操作の挙動」を用いることを提案する．スマートフォンやタブレットの普及により，日常生活においてタッチ操作が増加している．また，タッチパネルや OS（Operating System）の進化により，マルチタッチを用いた UI（User Interface）も一般的になっている．そして，同じアプリケーションでもマルチタッチの頻度や速度などの挙動に差が出る可能性が高い．つまり，図 1 に示すように，タッチ操作の挙動を通じてユーザの感情や個性，操作スキル、姿勢，何を同時にしているか等，より複雑なコンテキストがユーザのタッチ操作を取得し分析することで取得できる可能性が高い．

しかしながら，Android に代表されるモバイル OS（Operating System）では，セキュリティの観点から Sandbox と呼ばれる概念に基づいて設計されており，各アプリケー

[1] 奈良先端科学技術大学院大学
Nara Institute of Science and Technology
8916-5, Takayama, Ikoma, Nara 630–0192, Japan
[a)] hirabe.yuko.ho2@is.naist.jp

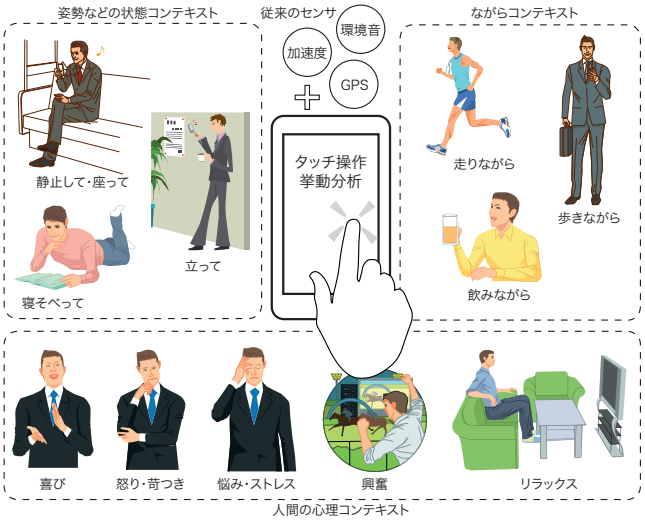

図 1　タッチ操作の挙動からわかるコンテキストの例

ション間は完全に分離されている．すなわち，さまざまなアプリケーションに対して行われるタッチ操作を別のロガーから監視，取得することは一般的には難しい．それに対して，我々は OS から出力されているデバイスイベントファイルに着目し，その一見不可解なログファイルを逆解析することによって，スマートフォン上でのあらゆるのタッチ操作を取得できるのではないかと考えた．

イベントデバイスファイルは，我々の提案の有無にかかわらず，OS によって常時出力されているファイルであり，その逆解析が成功すれば，本コンテキストの利用による電力増大をほとんど招くことなく常時取得可能な情報源となりうる．

そこで本論文では，タッチ操作の挙動という情報が人のコンテキスト認識に利用可能かという検証に先立ち，まず，Android OS 上でのあらゆるタッチ操作を認識可能なロガーの開発について報告する．その後，開発したロガーを用いて，２名の被験者実験を行い，タッチ操作の挙動の違い，および違いからわかることに関して議論する．

2.　関連研究

タッチ操作ではないが，ユーザが Web ページのどのボタンや部分がクリックされやすいかという情報を可視化する商用システムは多く存在し，代表的なものに ClickTale[*1]があげられる．これは，Web ページにスクリプトを埋め込むことで，マウス操作を追跡し，追跡結果を録画する等の機能をもつ．また，スマートフォン向けの ClickTale Touch も提供されており，スマートフォンアプリにおいても同様に使用できる．アプリ開発者がアプリケーション内に実装することで，アプリケーション内の操作状況を取得している．さらにヒートマップを用いてユーザの操作結果を可視化している．しかしながら，ClickTale では，サービス提供

[*1] http://www.clicktale.com/products/clicktale-touch

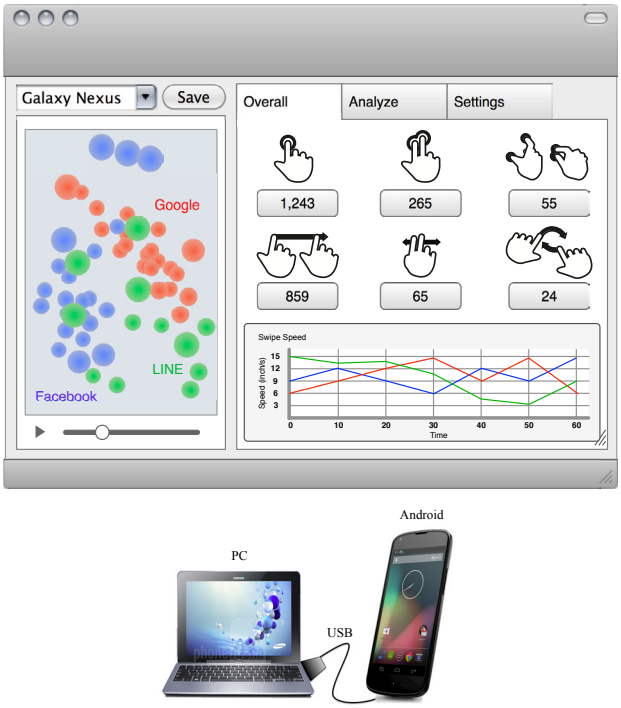

図 2　TouchAnalyzer の構成

者が自らアプリ及びサイト内にスクリプトを埋め込む必要があり，それ以外のアプリケーションではユーザのタッチ操作を取得することはできない．つまり，本研究で行いたいアプリケーションを横断してのユーザのタッチ操作の取得は不可能である．

TouchLogger[6] や Touchalytics[7] という類似した名前の研究があるが，これはタッチ操作そのものを取得するのではなく，タッチ操作で入力するソフトウェアキーボードのキー入力を推定するというセキュリティ分野の研究であり，ソフトウェアキーボードの各キーを押した場合の加速度センサの変化を分析するのみに留まっている．

3.　開発したロガー TouchAnalyzer について

将来的には，スマートフォンの１つのアプリケーションとしてロガーを実装したいと考えているが，今回は，出力されるログの意味を分析することを目標とし，図 2 に示すように，ロガーは PC 上で動作し，スマートフォンは USB 経由で接続される構成とした．TouchAnalyzer は Android SDK の platform-tools に含まれる，Android Debug Bridge（以下 adb と記載）と python を用いて実現する．TouchAnalyzer の分析対象となる AndroidOS は，AndroidOS 全バージョンであるが，実機を用いて確認しているのは，バージョン 4.2.4，4.2.1，4.1.2，4.0.4，4.0.2，2.3.4 である．ロガーの画面は，開発中のイメージであるが，タッチ操作の挙動を記録・再生する機能，アプリケーションごとの色分け機能，各タッチ操作の統計および推移を表示する機能などから構成される．

3.1 タッチ操作ログの取得

Android では，Linux と同様に/dev/input などのデバイスログにタッチイベントに関するデータが出力される．そこで，TouchAnalyzer では，まず/dev/input などのデバイスログに出力されるタッチイベントログを，adb ツールからシェルコマンドを発行し取得する．またこの時，foreground (ユーザが操作している) のアプリケーションのログを取得する．しかしながら，機種によってタッチイベントの出力先や時刻フォーマットが異なる．

具体的には，/dev/input などのデバイスログには，ファルダ event0〜event*まで存在する．ただし「*」の値は数字であり，各機種により event フォルダの存在する数は異なる．さらに，時刻フォーマットは大きく 2 つ存在し，これもまた機種により異なる．我々が確認した各機種とタッチイベントの出力先，時刻フォーマットの違いについてまとめたものを，表 1 に示す．

表 1 2 点タッチした際に出力されたデータ

	機種	時刻フォーマット		event			Android OS
		「-」型	「.」型	1	2	6	
1	Galaxy Nexus		○	○			4.2.1
2	Galaxy S III	○				○	4.0.2
3	Ideapad Tablet	○				○	2.3.4
4	Xperia Arc HD	○			○		4.0.4
5	Nexus 4		○		○		4.2.4
6	Galaxy Note II		○		○		4.1.2
7	GalaxyS II	○			○		4.0.3

以降の章では，Android4.0.2 がインストールされた GalaxyS II のタッチログデータを用いて，タッチイベントデータの解析を行った結果について述べる．

3.2 タッチ操作ログデータの解析

/dev/input などのデバイスログに出力される実際のタッチイベントデータログは，一見してもどのようなタッチ操作が行われたのか不明である．TouchAnalyzer では，代表的なタッチ操作として，図 3 のシングルタッチ・マルチタッチ，図 4 のシングルスワイプ・マルチスワイプ，図 5 のピンチイン・ピンチアウト，図 6 のローテートに関して，デバイスイベントファイルの逆解析を行う．

これらの挙動は，スマートフォンを操作する多くのユーザが一般的によく行うものであり，またスマートフォンのタッチ操作のほとんどを占めている．もしこれらの挙動の識別とユーザが使用しているアプリケーションのデータが同時に取得できれば，ユーザがどのアプリケーションを頻繁に使用し，アプリケーションごとにどのようなタッチ操作を行っているのか，といった情報が取得可能になると考える．これらの情報は，タッチ操作の挙動に対する様々なコンテキストを推定する上で必要になると考える．

3.2.1 タッチイベントデータのフォーマット

実際に出力されたログデータの 1 文を図 7 に示す．出力

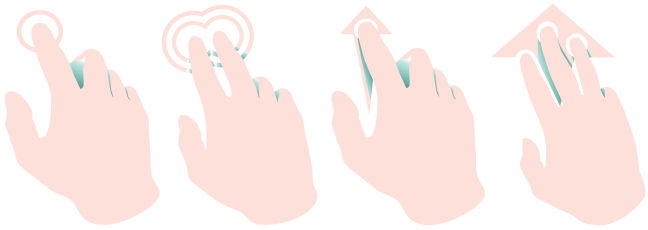

図 3 シングル・マルチタッチ

図 4 シングル・マルチスワイプ

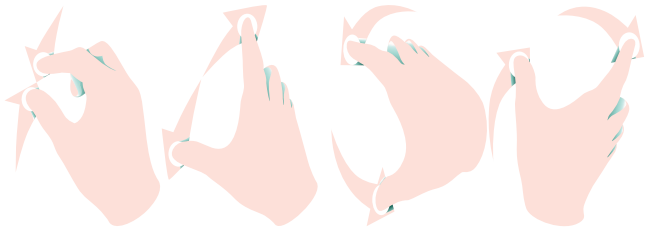

図 5 ピンチイン・ピンチアウト

図 6 ローテート

されるデータは，左から順に，起動経過時間，処理フラグ，種類，値を示している．起動経過時間とその他の出力は，セミコロンとスペースで区切られている．処理フラグ，種類，値の項目はスペースにより区切られている．タッチした際に出力されるデータは 16 進数で表現されている．今回は，処理フラグ，種類，値の項目を，それぞれ 1 列目，2 列目，3 列目として定義した．各列に出力されるデータは 16 進数で表現されている．起動経過時間，処理フラグ，種類，値の項目内容について順に詳しく述べていく．

まず，起動経過時間について述べる．起動経過時間の時刻フォーマットには 2 種類存在し，ハイフン「-」型，コロン「.」型がある．機種ごとの出力の違いについては，表 1 に示している．また，1 列目の処理フラグについて述べる．処理フラグには「0000」と「0003」2 つの値が出力され，それぞれタッチイベントが未処理であるか処理中であるかを表している．

次に，2 列目の種類について述べる．この種類とは，出力されたログデータがどのような値の取得，または処理を行っているかを表している．処理フラグが「0000」の時は，必ず 2 列目に「0000」が出力される．今回この情報は，取得した情報の区切りとして用い「==== ==== ====」と表した．また処理フラグが「0003」の時，つまりタッチイベントが処理されている時は，(a)〜(g) の処理内容が存

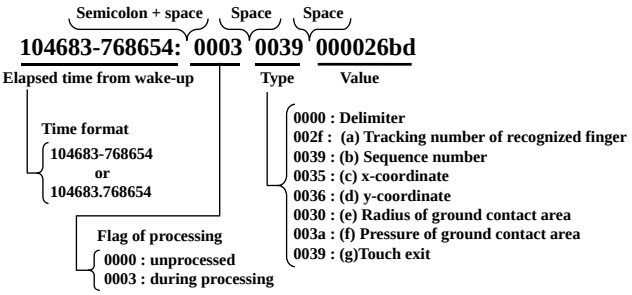

図 7 出力されるログデータの 1 文

在する．2 列目に「002f」が出力される場合には，(a) 認識した指に追跡番号の取得を行っている．また，「0039」が出力された場合には，(b) シーケンス番号と (g) タッチ終了が存在し，この識別には，3 列目の値を用いている．もし 3 列目が「ffffffff」ならば，(g) タッチ終了を示している．その他の値であれば，(b) シーケンス番号の取得を行っている．「0035」が出力された場合は，タッチした場所の (c) x 座標を取得し，「0036」が出力された場合は，タッチした場所の (d) y 座標を取得している．また，「0030」が出力された場合は，タッチした部分，つまり (e) 接地面積の半径を取得したことを示している．GalaxyS II の解像度が WVGA (Wide-VGA) の幅 800px，縦 480px であるため，面積単位は $[\mathrm{rad}\cdot\mathrm{px}^2]$ で表されると考える．「003a」が取得された場合は，タッチした部分の圧力，つまり (f) 接地部分の圧力を示している．圧力単位は [hPa] である．

表 2 2 点タッチした際に出力されたログ

	出力ログ例			ログ解析結果
	列 1	列 2	列 3	タッチイベントに関する
1	0003	002f	0000001	(a) 認識した指に追跡番号
2	0003	0039	000001b1	(b) シーケンス番号
3	0003	0035	00000098	(c) x 座標
4	0003	0036	000001ca	(d) y 座標
5	0003	0030	00000053	(e) 接地面積の半径
6	0000	0000	00000000	==== ==== ====
7	0003	002f	0000000	(a) 認識した指の追跡番号
8	0003	0039	000001b2	(b) シーケンス番号
9	0003	0035	0000015c	(c) x 座標
10	0003	0036	000001b1	(d) y 座標
11	0003	0030	0000005d	(e) 接地面積の半径
13	0003	003a	00000006	(f) 接地部分の圧力
14	0000	0000	00000000	==== ==== ====
15	0003	002f	0000001	(a) 認識した指の追跡番号
16	0003	0039	ffffffff	(g) タッチ終了
17	0000	0000	00000000	==== ==== ====
18	0003	002f	0000000	(a) 認識した指の追跡番号
19	0003	0039	ffffffff	(g) タッチ終了
20	0000	0000	00000000	==== ==== ====

表 2 は，端末画面を 2 点タッチした際，/dev/input/event2 に出力されたログの例とそのログを図 7 の (a)〜(g) を用いて解析した結果である．処理フラグ，種類，値の項目を，それぞれ 1 列目，2 列目，3 列目として定義している．表 2 の出力からわかるように，(a)〜(g) のうち，必ず出力されるものと，そうでないものがある．具体的には，(b)，(c)，(d)，(g) のログデータは必ず出力される．また，(a) はマルチタッチ時のみ出力され，追跡番号が付与される．(e)，(f) は出力される頻度としては低い．この理由としては，センサ の感度が原因であると考える．以降，出力された内容の解析結果 (a)〜(g) を用いて詳述していく．

3.2.2 シングルタッチ・マルチタッチ

タッチした際，シングルタッチかマルチタッチかを判断するために，表 2 に示した (a) 認識した指の追跡番号が使われている．もし (a) が取得されなければ，シングルタッチであると判断される．また，(a) が取得された場合は，認識した指の数だけ (b)〜(f) の値を取得し，認識した指の追跡が行われる．タッチイベントの終了に関しても同様に行われ，(a) で得られた指の追跡番号の分だけタッチイベントの終了が行われる．タッチイベント処理の流れを Algorithm1 として示す．例えば，3 本の指で画面をタッチ

Algorithm 1 シングルタッチ・マルチタッチ時の処理

```
if (a) が取得されない then
  表 2 の (b)〜(f) を取得
else {(a) が取得される }
  for (a) 認識した指の追跡番号がある && 追跡番号が 10 より小さい do
    認識した指の追跡番号ごとに表 2 の (b)〜(f) を取得
  end for
end if
```

した場合は，それぞれの指に追跡番号が割り振られ，それぞれの指において (b)〜(f) の値を取得し追跡していく．またもし仮に，3 本の指で画面をタッチした後，指を 1 本画面から離し，2 本指で操作する場合は，追跡番号から離された指のみを識別し，タッチ終了の処理を行う．そして，残りの 2 本の指の追跡を続ける．認識される指の数は最大で 10 本であることを確認した．TouchAnalyzer による描画結果を図 8 として示す．

3.2.3 シングルスワイプ・マルチスワイプ

スワイプはシングルタッチ・マルチタッチ時の認識方法を応用している．つまり，Algorithm1 を繰り返し，追跡している指ごとに終了判定が行われる（指が離れる）まで連続でタッチデータを取得し続ける．これにより，シングルスワイプ，マルチスワイプを認識している．TouchAnalyzer によるスワイプ描画結果を図 9 として示す．

3.2.4 ピンチ

ピンチにはピンチイン・ピンチアウトが存在し，マルチスワイプの場合でも，特に 2 点スワイプ時に起きる特別な挙動である．追跡している指が離れるまで連続でタッチデータを取得し，追跡していた 2 本の指が離れた場合に，取得した 2 つの終点 P_3，P_4 の間の距離 L1，始点 P_1，P_2 の間の距離 L2 を求め，L1，L2 の大小比較からピンチイン・ピンチアウトを認識している．TouchAnalyzer によるピンチイン描画結果と，始点・終点の関係を図 10 の左として示す．

3.2.5 ローテート

ローテートは 2 点タッチと 2 点スワイプを組み合わせた特別な挙動である．Algorithm1 を繰り返すスワイプ動作の中でも，タッチデータの軌跡が円を描くような場合に認識される．追跡している指が離れるまで連続でタッチデータを取得し，終点 P_3 から始点 P_1，P_2 を結ぶ直線へ垂線を引いた距離を L1，終点 P_4 から始点 P_1，P_2 を結ぶ直線へ垂線を引いた距離を L2 とし，L1 と L2 の長さがある閾値以上であればローテートであると認識する．今回は始

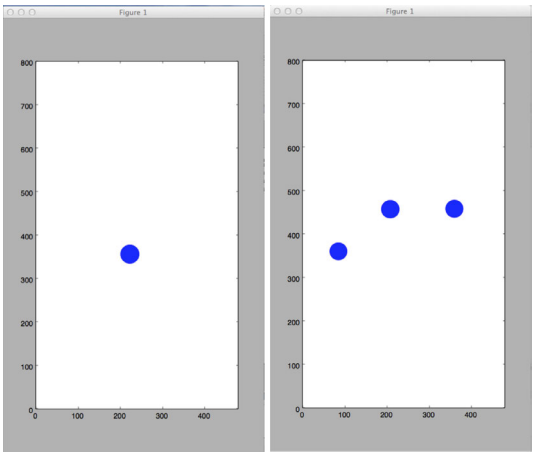

図 8　シングルタッチ・マルチタッチ

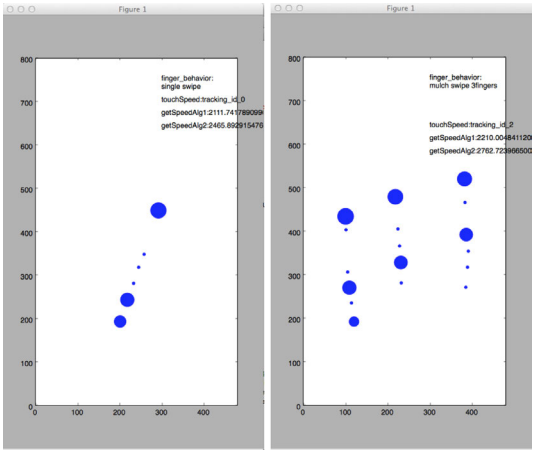

図 9　シングルスワイプ・マルチスワイプ

点 P_1，P_2 を結ぶベクトルを u，始点 P_1 と終点 P_3 を結ぶベクトルを v，始点 P_2 と終点 P_4 を結ぶベクトルを w とし，外積計算から L1 と L2 を求めローテートを認識した．TouchAnalyzer によるローテート描画結果と，始点・終点の関係を図 10 の右として示す．

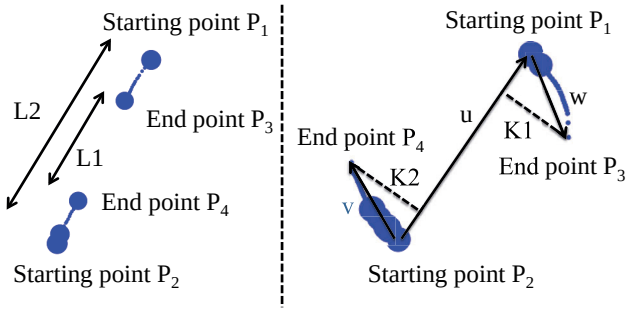

図 10　ピンチとローテートの定義

3.3　タッチ操作の統計分析

各識別した挙動ごとの頻度と平均速度を出力した．頻度は，先に示したタッチ操作の挙動を認識した回数により求める．次に，平均速度について述べる．一般的に速度を求める場合，式

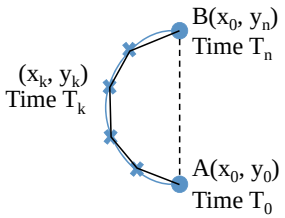

図 11　タッチ操作によく見られるスワイプの軌跡

$$\bar{x_{ab}} = \frac{\sqrt{(x_n - x_0)^2 + (y_n - y_0)^2}}{T_n - T_0} \tag{1}$$

が適用される．しかしながら，タッチ操作の挙動においては，しばしば（特にスワイプ時）図 11 に示すような軌跡が見られる．しかしこの場合，点 A と点 B の x 座標は等しいため，式 1 では適切な速度を取得できない．そこで本研究では，挙動ごとの平均速度を出力するために 2 つの計算式を考案し，TouchAnalyzer に実装した．取得したタッチ操作の挙動として取得した描画点が n 個あった時，求める速度を式 2 と定義した．

$$\bar{x_{ab}} = \frac{\sum_{k=0}^{n-1} \frac{\sqrt{(x_{k+1} - x_k)^2 + (y_{k+1} - y_k)^2}}{T_{k+1} - T_k}}{n - 1} \tag{2}$$

式 2 は，点間隔ごとに速度を出して最終的に取得した点の数から 1 引いた値で割っている．つまり，各点ごとの速度を取得し足した後，全体の速度の平均をその挙動の速度としている．我々が提案する 2 式により，図 11 に示すような軌跡でも速度を取得できると考える．

3.4　タッチ操作の可視化

解析したタッチ操作ログのデータから，タッチ操作の可視化を行った．TouchAnalyzer では，PC に USB 接続した端末のタッチデータをリアルタイムに取得し，取得した x，y 座標を用いて，タッチした場所を点として描画する．また，取得したデータにタッチ面積の半径が含まれる場合には，描画する点を円とし，タッチ面積の半径に合わせて描画する円の大きさが変わるよう設計した．また TouchAnalyzer は，タッチ操作の挙動を取得する際，foreground にあるアプリケーションの情報を横断的に取得している．そして，アプリケーションがユーザのタッチの挙動により切り替わるたび，描画する色を変化させている．

4.　タッチ操作の取得実験

実際に開発したシステムを用い，タッチ操作の取得実験

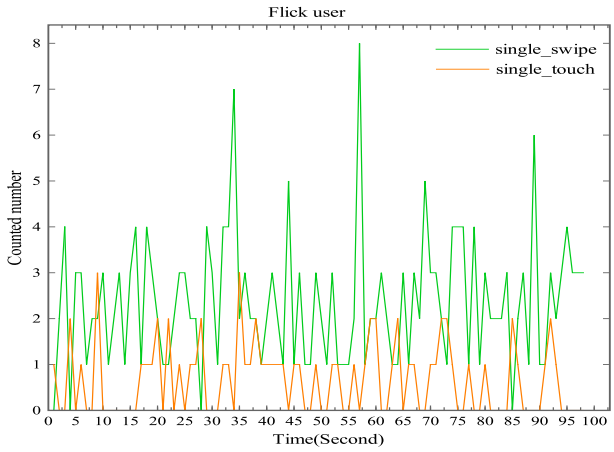

図 12　フリック入力ユーザ

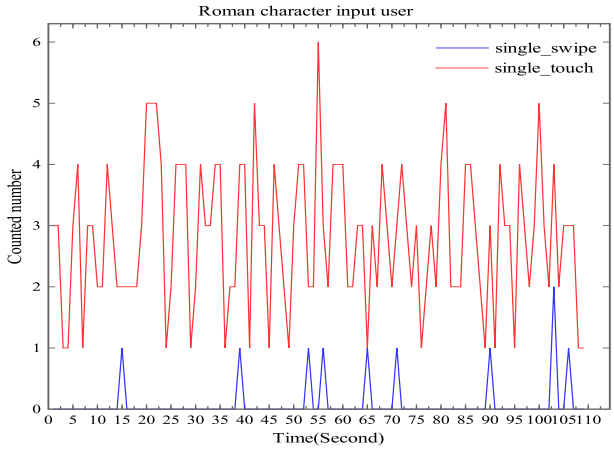

図 13　日本語ローマ字入力ユーザ

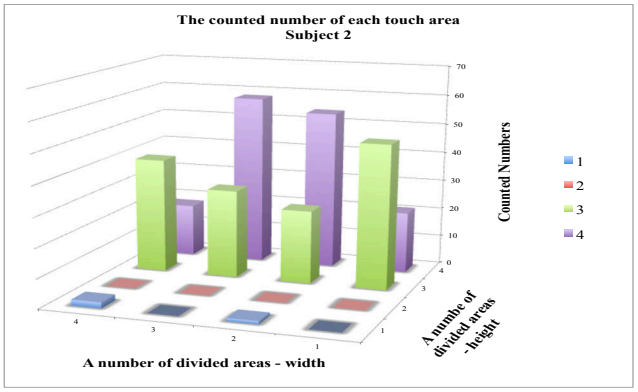

図 14　フリック入力ユーザ

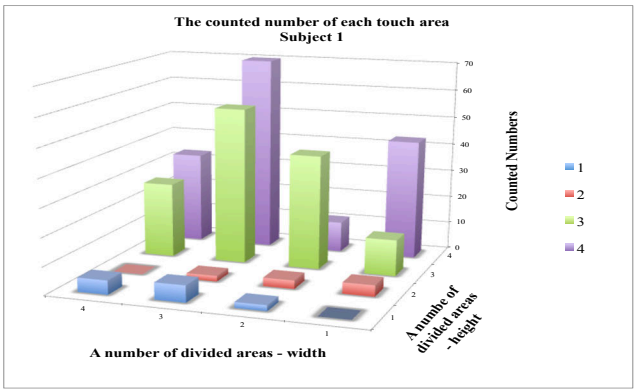

図 15　日本語ローマ字入力ユーザ

を行い，コンテキストの差がタッチ操作に現れるのかを検証した．本実験では，操作慣れによる差をできる限り低減するため，被験者が普段利用している端末とは異なる端末（Galaxy SII）を用いて実験を行った．実験課題は，(1) 用意した実験課題と (2) 3 分間各自の Facebook ページを操作する，という 2 つである．また，以降の節におけるタッチ操作の速度は，定義式 2 を用いて算出された値である．

4.1　文字入力を用いた実験

1 つ目の実験課題は，以下の文章を入力するというものである．この文章は，Yahoo ニュースから取得したものであり，被験者は初めて目にする文章である．この際，英字など普段，日本人が入力に慣れていない文字は含まない文章を選定するとともに，漢字の知識に依存しないようにすべての漢字に対して読み仮名をふった原稿を用意し，印刷された原稿を見ながら入力してもらっている．

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

台風 27 号は 17 日午後 3 時現在、マリアナ諸島にあり、ゆっくりと北北西へ進んでいる。中心気圧は 970 ヘクトパスカル、中心付近の最大風速は 35 メートルで、強い勢力となった。また、中心から 90 キロ以内では風速 25 メートル以上の暴風となっている．

＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝＝

4.1.1　入力方式による違い

スマートフォンで日本語を入力する場合，QWERTY キーボードでローマ字入力をする方法だけではなく，フリック操作により入力する方法などいくつかの入力方法が考えられる．それらの入力にはそれぞれ特徴があると考えて，タッチ操作の挙動と入力方式の関係について分析した．今回の被験者 4 名の場合，フリック入力とローマ字入力の 2 方式のみであり，携帯入力など他の入力手法を利用しているユーザはいなかった．

図 12，図 13 にそれぞれの入力方式におけるシングルスワイプとシングルタッチの頻度について示す．フリック入力のユーザは，シングルスワイプの頻度が多いのと，毎秒現れていることが特徴的である．一方，ローマ字入力の場合はシングルタッチが多く，たまにシングルスワイプが現れる．このシングルスワイプは，変換候補が表示され選択する際に，1 ページ目の所望の候補が表示されず，スワイプでページめくりをしている状態と考えられる．これらの結果から，利用している入力方式は容易に推定可能であると考えられる．

次に，これをデバイスの画面領域を縦・横それぞれ 4×4 に分割し，分割した領域毎にカウントしたものを図 14，図

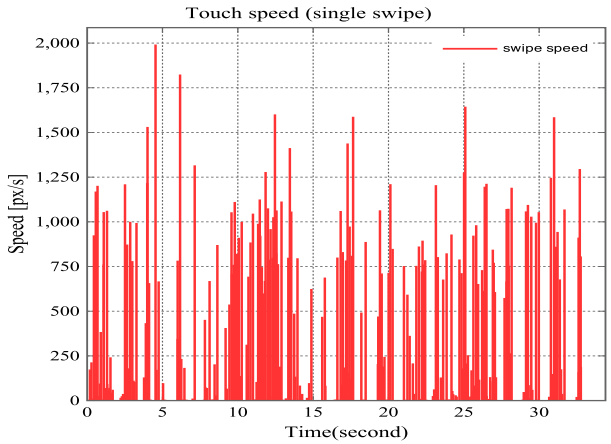

図 16　タッチ速度（シングルスワイプ）

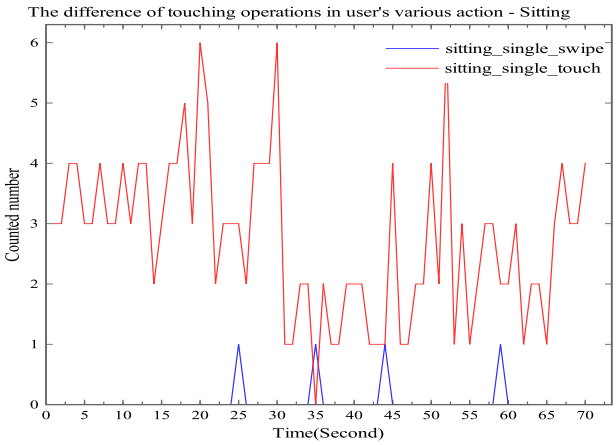

図 17　座りながら

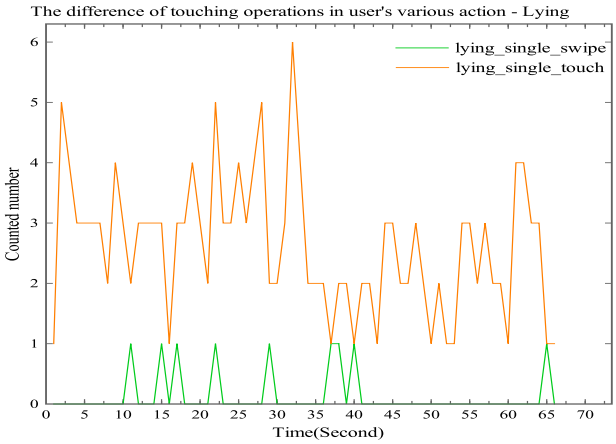

図 18　寝転びながら

15 に示す．これは，携帯電話のスクリーンのどの領域が頻繁に利用されいているかを表しており，グラフ奥がスマートフォンの画面下部に相当する．この図から，明確な差をどのような分析手法で発見するのかは今後の課題であるが，フリック入力は画面下部をまんべんなく使っているのに対して，ローマ字入力では，同じ画面下部であっても偏りが見られることがわかる．

最後に，タッチ速度の変化について示す．今回は，速度によるユーザ間の明確な差を分析することができなかったため，スワイプ入力の被験者 1 名の速度変化を一例として図 16 に示す．この図では，横軸が経過時間を表し，縦軸が速度を表す．フリック入力の度に，速度が上がり，文字選択（シングルタッチ）の際に速度が 0 になると考えられる．今後，速度の分散や入力文字との関係性を分析するとともに，同じフリック入力ユーザでどのような違いが現れるのかに関しても分析していく予定である．

4.1.2　入力姿勢による違い

スマートフォンを操作する場合，ユーザは様々な姿勢で入力することが考えられる．加速度センサはユーザが動作しない場合に反応を示さないため，姿勢の認識を行うことができない．そこでタッチ操作の挙動から様々な姿勢による入力にも各特徴があると考え，タッチ操作の挙動と入力方法の関係について分析を行った．今回は，ユーザが日常生活で一般的に行っている代表的な入力姿勢として，座りながら (図 17)，寝転びながら (図 18)，話しながら (図 19)，歩きながら (図 20) の 4 つの入力姿勢を対象とした．現時点での結果では，入力姿勢による明確な差を分析する事ができなかったが，これより更にタッチ間の時間や分散の関係を用いて入力姿勢の違いによる分析をおこなっていく予定である．

4.2　Facebook を用いた実験

本実験では，被験者に 3 分間各自の Facebook ページを操作してもらり，タッチ操作の種類とその頻度，平均速度について分析を行った．

図 21 は 4 人の被験者のスワイプ操作に関する速度分布である．横軸は各スワイプ操作を表しており，被験者 4 はスワイプ操作回数が最も多かったことを意味している．また被験者 2 と 3 は，スワイプ操作の速度が安定しているのに対して，被験者 1 と 4 は何らかの操作の場合は，高速なスワイプ速度を行っていることがわかる．被験者 4 名の平均速度は，図 22 のようになっているが，これを特徴量としてとらえるには至っていない．今後は，操作状況をビデオ録画するなどして，どのような操作がどのようなスワイプを誘発しているのかを確認しながら，速度とコンテキストの関係を明確にしたいと考えている．

謝辞　本論文の成果の一部は科学研究費補助金挑戦的萌芽研究（課題番号：25540031）の成果である．また，有益な助言をいただいたマイクロソフトリサーチアジア（MSRA）の矢谷浩司氏に感謝する．

5.　おわりに

本研究では，タッチ操作の挙動をコンテキスト推定の 1 つの情報として用いることを最終的な目的として，まずユーザが行っている代表的タッチ操作の幾つかに関して，

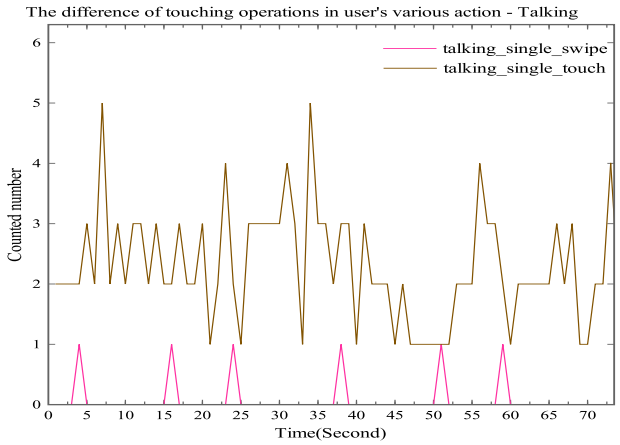

図 19 喋りながら

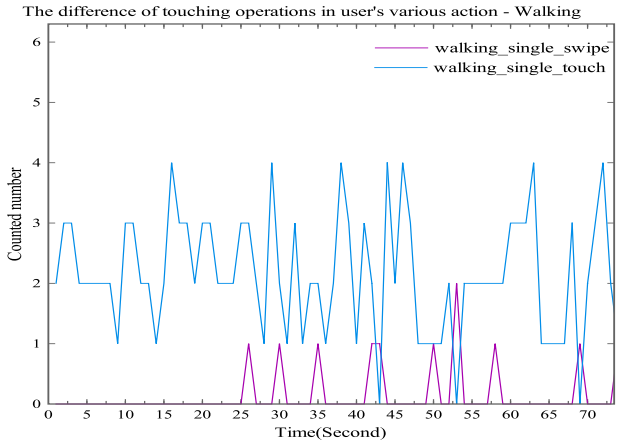

図 20 歩きながら

逆解析アルゴリズムを確立し，タッチ操作の取得・識別を行うロガーは開発した．そして，開発したロガーを用いて，タッチ操作のロギングが可能であることを確認するとともに，タッチ操作の頻度や速度を統計的に表示可能であることを確認した．被験者実験により，同じ端末で同じタスクを行った場合も，人によって操作が違うことが明確になったが，今後は，こうした統計値と人のコンテキストの関係性をより詳しく分析し，どのような定量化手法が適切であるか，どのようなコンテキスト認識に向いているのかなどを明らかにしていきたいと考えている．そして，将来的には，個人の操作スキルに適応した動的 UI や，ユーザの感情を理解するコンシェルジュアプリの開発，またユーザが寄った状態でも入力間違いを起こしづらい文字入力システムといった，これまでにないコンテキストアウェアなサービスへと発展させていきたいと考えている．また，本研究の成果は，ライブラリとして公開し，さまざまな研究において，タッチ操作の挙動というコンテキストを利用可能にしたいと考えている．

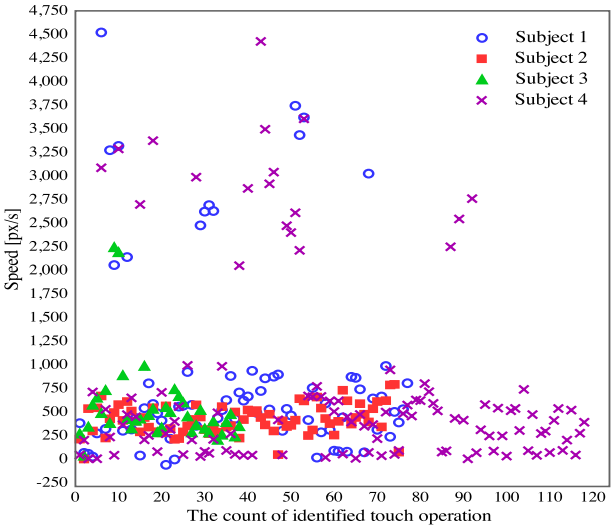

図 21 式 2 から取得した速度データの散布図

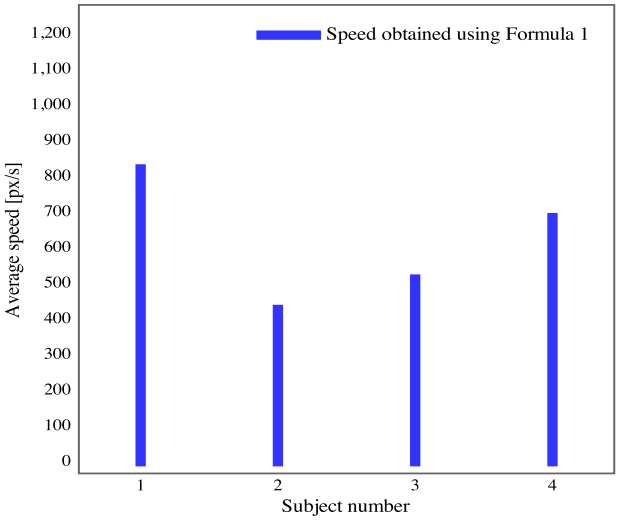

図 22 被験者ごとの全タッチ操作の平均速度

参考文献

[1] GPS Log. http://gpslogapp.com/.

[2] NIKON IMAGE SPACE. http://nikonimagespace.com/.

[3] Kawaguchi, N., Ogawa, N., Iwasaki, Y., Kaji, K., Terada, T., Murao, K., Inoue, S., Kawahara, Y., Sumi, Y. and Nishio, N.: HASC Challenge: gathering large scale human activity corpus for the real-world activity understandings, *Proceedings of the 2nd Augmented Human International Conference*, ACM, p. ¯ 27 (2011).

[4] Kawaguchi, N., Ogawa, N., Iwasaki, Y., Kaji, K., Terada, T., Murao, K., Inoue, S., Kawahara, Y., Sumi, Y. and Nishio, N.: HASC Challenge: gathering large scale human activity corpus for the real-world activity understandings, *Proceedings of the 2nd Augmented Human International Conference*, ACM, p. ¯ 27 (2011).

[5] Fukumoto, K., Terada, T. and Tsukamoto, M.: A smile/laughter recognition mechanism for smile-based life logging, *Proceedings of the 4th Augmented Human International Conference*, ACM, pp. 213–220 (2013).

[6] Cai, L. and Chen, H.: TouchLogger: inferring keystrokes on touch screen from smartphone motion, *Proceedings of the 6th USENIX conference on Hot topics in security*, USENIX Association, pp. 9–9 (2011).

[7] Frank, M., Biedert, R., Ma, E., Martinovic, I. and Song, D.: Touchalytics: On the applicability of touchscreen input as a behavioral biometric for continuous authentication (2012).