

# 大画面環境におけるハンドジェスチャの選択手法

## ——ダブルクロッシングの提案と他の選択手法との比較——

中村 卓<sup>†</sup>      高橋 伸<sup>††</sup>      田中 二郎<sup>††</sup>

The Selection Technique of Hand Gesture in Large Screen Environment

——Proposal of Double-Crossing and Comparison with Other Techniques——

Takashi NAKAMURA<sup>†</sup>, Shin TAKAHASHI<sup>††</sup>, and Jiro TANAKA<sup>††</sup>

あらまし 大画面環境で離れた場所からインタラククションを行う手法として、指先の動きを利用してポインタの移動により GUI を操作するポインティングハンドジェスチャがある。この手法では、GUI のアイテムの選択などマウスの左クリックに相当する選択操作を定義する必要がある。そのための選択手法として、ポインタを一定時間静止する「ウェイティング」が考えられるが、手の認識座標のぶれによりポインタの位置がずれやすく、決まった時間だけ静止しなければならないため選択に時間がかかるなどの問題が起きる可能性がある。他の選択手法として、「シングルクロッシング」がある。これはターゲットをポインタで横切ることによって選択を行うが、意図しないターゲットを誤って横切ってしまうおそれがある。我々は、シングルクロッシングをポインティングハンドジェスチャ向けに改良した選択手法として、「ダブルクロッシング」を提案する。また評価実験を行い、この三つの選択手法の選択時間や誤選択率を測定した。その結果、大画面アプリケーションの作成を行う場合、通常はダブルクロッシングを使用し、操作の正確さを必要とする場合はウェイティングを使用するのが適しているとの実験結果を得た。

キーワード ハンドジェスチャ、ウェイティング、シングルクロッシング、ダブルクロッシング

### 1. ま え が き

ディスプレイの低価格化に伴い、大画面が様々な場所に広く普及し、様々な用途に利用されている。このような大画面環境では、図 1 のように離れた場所からインタラククションを行うことで、画面全体を眺めながらの操作ができるため、複数のウィンドウにまたがる作業や沢山の情報を扱うときなどに有効である。しかし、環境によっては、机などの水平なスペースを用意することが難しく、また手元と画面を同時に確認しながらの作業が難しい。そのため、マウスやキーボードによるインタラククションを用いた従来のインタラク

ション手法を行うことが難しく、大画面環境に適したインタラククション手法 [1]～[3] が必要である。

大画面環境で離れた場所からのインタラククション手法として、ハンドジェスチャを利用したインタラククション手法があり、1980 年頃から研究が行われ [4]、現在も様々な手法が研究されている [5], [6]。このハンドジェスチャインタラククションの一つに手や指先の動き

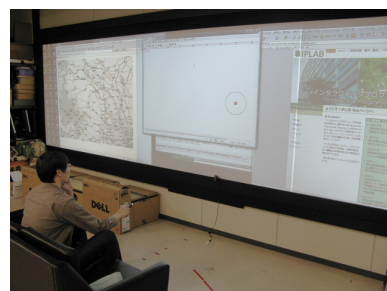


図 1 離れた場所からの大画面とのインタラククションの例

Fig. 1 Example of remote interaction with large display.

<sup>†</sup> 筑波大学大学院システム情報工学研究科コンピュータサイエンス専攻, つくば市

Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba, Tsukuba-shi, 305-8573 Japan

<sup>††</sup> 筑波大学システム情報系情報工學域, つくば市

Faculty of Engineering, Information and Systems, University of Tsukuba, Tsukuba-shi, 305-8573 Japan

に応じて移動するポインタの動きのジェスチャ（ポインティングハンドジェスチャ）がある．全ての操作をこのポインタの移動を利用することで、同じような動きで操作を行うことができる．しかし、Windowsなどの既存のグラフィカルユーザインタフェース（GUI）等进行操作するためにはポインタを動かすだけでなく、アイコンやメニューを選択する操作が必要である．しかし、ポインティングハンドジェスチャ自体にはマウスのクリックに相当する操作がないため、特定のポインタの動きを選択操作として割り当てる必要がある．ポインタの動きを利用した選択手法として、ペンベースインタフェースやレーザポインタを利用した手法[7]などで利用されている手法の利用が考えられる．

ポインティングハンドジェスチャにおける最も簡単なオブジェクトの選択手法として、例えば、選択するオブジェクトの上で一定時間滞留することでポインタ上のオブジェクトを選択する手法（ウェイトイング）がある．しかし、ハンドジェスチャは手の認識座標のぶれによりポインタの位置がずれやすいため、ポインタを一箇所に留めておくことが難しい．また、決まった時間だけ静止しなければならないため、一回の選択に時間がかかってしまうおそれがある．

ペンベースインタフェースで利用される選択手法の一つとして、クロッシング[8]と呼ばれる手法がある．クロッシングはターゲットを横切る軌跡のポインティング動作で選択を行う手法であり、マウスのクリックなどの代替手段として利用できることが知られている．我々はポインティングハンドジェスチャにおいてもクロッシングを利用することでウェイトイングに変わる操作手法を設計できるのではないかと考えた．クロッシングはターゲットと交差するだけで簡単な操作であり、初見のユーザでもすぐに利用できることが期待できる．しかし、ポインティングハンドジェスチャは基本的には指先の動きに合わせて常にポインタが移動するため、意図しないオブジェクトを誤ってクロッシングしてしまうおそれがある．

## 2. ダブルクロッシングの提案

我々は、クロッシングをポインティングハンドジェスチャ向けに改良した選択手法として、ダブルクロッシング[9]～[11]を利用したインタラクション手法を提案する．通常のクロッシング手法（シングルクロッシング）では、図2のように一回だけ交差する．ダブルクロッシングでは、図3のように短時間に同じ

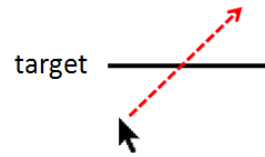


図2 (シングル) クロッシング  
Fig.2 (Single) crossing.

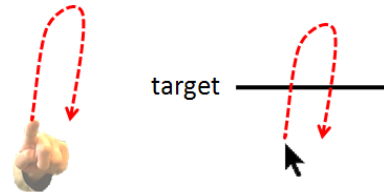


図3 (左) ダブルクロッシングの際の手の動き  
(右) ポインタの動きとダブルクロッシング  
Fig.3 (Left) Hand movement for double-crossing.  
(Right) Pointer movement and double-crossing.

オブジェクトを2回クロッシングすることでそのオブジェクトを選択する手法である．シングルクロッシングと比較して、手ぶれなどによる意図しないダブルクロッシングは非常に少なく、よりロバストな選択手法となるため、ダブルクロッシングはポインティングハンドジェスチャにおけるオブジェクトの選択手法として有効であると考えられる．また、この手法は指を上下（左右）に動かすという簡単な動作で操作を行うことができるため、利用するために特別なトレーニングなどを必要とせず、長い時間作業を行っても疲れにくいという利点もある．

### 2.1 メニューラベルの選択によるコマンドの実行

従来のGUIのメニューは、一列に並んで表示されていたり、一つひとつのメニューが小さいため、大画面環境ではそれらのメニューを確認するのが困難であったり、ポインティングジェスチャでは選択することも難しく、誤選択などを引き起こしやすいなどの問題がある．また、キーボードやマウスを利用しないため、画面のスクロールや文字入力のためのメニューなども用意する必要がある．

我々は、画面のスクロールやコピーなどメニュー選択による操作のために図4のようなメニューラベルを用意し、このメニューラベルをダブルクロッシングで選択することによって必要なメニューを素早く選択できるようにした．メニューラベルには実行線（図4の中央の線）が決められており、それをダブルクロッシングすることによって、そのメニューラベルに割り当

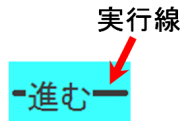


図 4 メニューラベル  
Fig. 4 Menu label.

てられた操作を実行することができる (図 4 中では実行線はラベルによって 2 本に見えるが、実際にはメニューラベルの中央を水平に横切る一本の線となっている)。また、同じメニューを連続で操作する場合、指を上下に振り続けるだけでよいので、連続操作も容易である。

クロッシングのための判定線は、必ずしも横線である必要はなく、縦線や斜め線、ラベルの外枠などを利用しても問題ない。しかし、ユーザが正確にクロッシングを行うことができるように、判定線はある程度の長さが必要である。

メニューラベルにはそれぞれに特定の操作が割り当てられている。例えば、図 4 では、「進む」というラベルであるため、ウェブブラウザにおける次のページを表示するといった操作が割り当てられている。

### 3. 大画面ポインティングシステムの実装

我々が作成した大画面ポインティングシステムの全体の構成は図 5 のようになっている。ユーザは、指先に図 5 のような LED デバイスを装着し、LED の光をユーザの正面に設置したカメラで認識することによってポインタの移動を行う。そして、その際のポインタの移動によってダブルクロッシングを行い、大画面とのインタラクションを行う。ポインタについては、標準よりも大きいポインタを利用しているが、このポインタ自体は、OS に最初から入っているもの (本研究では Windows Aero の特大フォント) を利用している。カメラと LED デバイスは、特殊な物ではなく、市販されている一般的な物を利用している (カメラは Logitech 社の Qcam Pro for notebook, LED デバイスは、100 円ショップなどで売っているものである)。なお、カメラの解像度は  $640 \times 480$  ピクセルで、フレームレートは 30 fps である。

指先の認識、ポインタの移動、ダブルクロッシングの検出、操作の実行及び大画面への表示は全て 1 台の PC で行われる。本研究で利用した PC のスペックは、DELL 社製の Vostro420 で、CPU: intel Core2 Quad

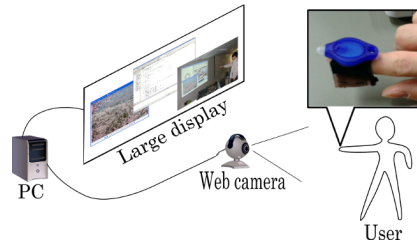


図 5 大画面ポインティングシステムの構成  
Fig. 5 System setting of large display pointing.

2.83 GHz, RAM: 4.0 GByte, HDD: 500 GByte, OS: Windows Vista であった。

#### 3.1 指先の認識とポインタの移動

指先の認識は、LED デバイスの光をカメラを利用して、背景差分と色や輝度によるしきい値処理によって行っている。なお、あらかじめカメラの露光などを絞っておくことで、LED 以外の光を検知しにくくしている。LED の位置の検出については、LED の光として検出された全座標の平均値をとっている。そのため、最終的な検出座標は整数ではなく、小数で算出される。そして、LED の検出座標を  $(cx, cy)$  としたとき、ポインタの移動先の座標  $(px, py)$  は以下の式で算出される。

$$px = \frac{cx}{camWidth} screenWidth + screenLeft$$

$$py = \frac{cy}{camHeight} screenHeight + screenTop$$

$camWidth$  と  $camHeight$  はそれぞれカメラの解像度の幅と高さを、 $screenLeft$  と  $screenTop$  は大画面上の最も左端の値と最も上端の値である。そして、算出された  $(px, py)$  の値に基づいてポインタを移動させる。

本研究の環境では、一回の認識にかかる処理時間は 10 数ミリ秒程度であり、誤差については、手をかざしていることによるぶれがあるが、静止状態ではポインタが 2・3 ピクセル動く程度の誤差であり、揺らぎに関してもほとんど見られない。

#### 3.2 クロッシングの認識

ポインタの移動軌跡と各ラベルとの交差判定については、二つの線分の交差判定に関する数式を用いて、以下のような手順で行われる。まず、図 6 のように時刻  $t$  におけるポインタの座標  $P_t$  と時刻  $t-1$  におけるポインタの座標を  $P_{t-1}$  とし、あるラベルの基準線の両端の座標を  $P_{m1}$  と  $P_{m2}$  とする。次にそれぞれの座標から、 $t1, t2, t3, t4$  を以下の式から算出する。

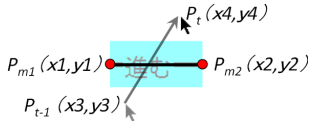


図 6 クロッシングの認識  
Fig. 6 Crossing recognition.

$$\begin{aligned} t1 &= (y1 - y3)(x3 - x4) - (x1 - x3)(y3 - y4) \\ t2 &= (y2 - y3)(x3 - x4) - (x2 - x3)(y3 - y4) \\ t3 &= (y3 - y1)(x1 - x2) - (x3 - x1)(y1 - y2) \\ t4 &= (y4 - y1)(x1 - x2) - (x4 - x1)(y1 - y2) \end{aligned}$$

そして、 $t1 \times t2 < 0$  かつ  $t3 \times t4 < 0$  を満たした場合にそのラベルと交差したと判断する．この式は、2 線分の交差判定を行う一般的な数式である．

シングルクロッシングの際は、この交差判定をそのまま利用することで判定可能である．ダブルクロッシングについては、あるラベルと交差した後に  $T$  ミリ秒以内（クロッシング間隔の上限）にもう一度交差した場合にダブルクロッシングが行われたと判断している． $T$  については、任意に定めて問題ない．

### 3.3 メニューラベルの実装

メニューラベルは、既存の GUI の上に透明なウィンドウを作成し、そのウィンドウ上に作成する．メニューラベルには、あらかじめ利用する既存アプリケーションのコマンドのショートカットキーが割り当てられている．複数の操作を行う場合には、その操作に必要な分のショートカットキーが割り当てられている．選択された際は、そのショートカットキーがエミュレートされ、コマンドが実行される．ショートカットキーのエミュレートにより、Windows などの既存のオペレーティングシステム（OS）で用いられている GUI やアプリケーションをそのまま利用することができる．

## 4. ターゲット選択手法に関する評価実験

Accot らの実験 [8] では、ペンストロークによるクロッシングは、マウスのボタンを利用した選択手法とほぼ同じ速度で選択することができ、選択の難しさについても同程度であることが示されている．我々は、ポインティングハンドジェスチャにおいて、ダブルクロッシングもペンにおけるクロッシング手法と同様にウェイティングなどの既存の手法の代わりとして利用可能であると考えた．そこで、ダブルクロッシングが実際にどのぐらいの速さで選択することができるかを



図 7 実験におけるブラウザとメニューの配置  
Fig. 7 Setting of browser and menus.

確かめるための評価実験を行い、マウスのボタンを利用しない選択手法について考察を行った．

### 4.1 実験タスク

本実験では、図 7 のように配置されたウェブブラウザのリンク（リンクターゲット）と右上にある方形のメニュー（メニューターゲット）のうち指定されたターゲットを選択するタスクを行い、選択時間や誤選択率を取得した．このタスクでは、リンクターゲットを 30 回、メニューターゲットを 20 回、合計で 50 回の選択を行った．なお、リンクターゲットとメニューターゲットを一定の順番で選択するのではなく、ランダムに指示されるターゲットを選択させるようにした．選択手法については、ダブルクロッシング、ウェイティング、シングルクロッシングの 3 種類を用いた．

実験は、以下の手順で行われた．なお、被験者はタスク間に自由に休憩をとることができた．

- (1) 利用する選択手法について説明し、その手法に慣れるまで自由に使ってもらう．
- (2) ターゲットの選択タスクを行う．
- (3) 被験者に使いやすさや疲れやすさなどに関するコメントを得た．
- (4) 他の選択手法で (1) から (3) を行う．
- (5) 全タスクが終了した時点で実験全体に関するコメントを得た．

被験者は研究室の学生 9 名で、指先の軌跡を用いた選択手法には慣れていなかった．そのため、選択手法に対する学習効果などを少なくし、ポインティングの癖などを十分に掴んでもらうために、事前の練習は十分に行ってもらった．また、一人目でダブルクロッシングを先にやった場合、次の被験者はダブルクロッシングを後で行うといったように、被験者ごとに利用する手法の順番を変えることでカウンターバランスを

とり、順番による影響も考慮した。各手法で、合計で (50 回) × (9 名) = 450 回の試行を行った。

#### 4.2 リンクターゲットとメニューターゲット

リンクターゲットは、既存の GUI のアイテムをそのまま利用することを想定としたターゲットである。リンクターゲットは、実際の研究室のホームページ<sup>(注1)</sup>を解析し、ページ上のリンクをターゲットとした。既存の GUI のアイテムをそのまま利用しているため、リンクターゲットのサイズにはばらつきがあり、小さいターゲットで 30 × 15 ピクセル、大きいターゲットで 190 × 50 ピクセルであった。配置についても、密集している場所と分散している場所が存在した。なお、リンクターゲットは全部で 31 個存在した。

メニューターゲットは、リンクターゲットとは異なり、ダブルクロッシングに合わせて我々が配置や大きさを調整したターゲットである。メニューターゲットは、図 7 右上にある水色の矩形のターゲットを用意した。メニューターゲットは、大きさは統一し、全て 100 × 50 ピクセルとした。配置については、メニューターゲット同士を 30 ピクセルずつ離して配置した。メニューターゲットは図 7 のように全部で 10 個用意した。

クロッシングのための実行線については、縦線・横線・斜め線などが考えられる。被験者が正確にクロッシングを行うためにある程度の長さが必要であるが、斜め線を利用した場合、線の方向が左下や右下など複数考えられ、線の方向で選択しにくくなるおそれがある。本実験では、選択手法の評価に重点を置いているため、ある程度の長さや間隔をとるために、ラベル中央を通る横線で統一した。なお、本実験では、全てのターゲットは横長であった。

#### 4.3 実験環境

実験は、横 2.6 m × 縦 1.5 m の大画面（解像度 1920 × 1080 ピクセル）で、被験者はこの画面から 3 m 離れた場所から座った状態で操作を行ってもらった。指先を認識するためのカメラは被験者の約 1 m 手前に設置した。ウェブブラウザを画面の左端から 3 分の 2 を占める大きさで設置し、そのブラウザ上にリンクターゲットを用意し、図 7 のようにそのすぐ右上にメニューターゲットを設置した。実験に利用した PC やカメラ、LED デバイスについては、3. で述べたものと同じものを利用した。

#### 4.4 予備実験

本実験では、ダブルクロッシング以外の手法として、

表 1 各静止時間の選択時間と誤選択率

Table 1 Results of each waiting time.

|          | リンクターゲット |        | メニューターゲット |        |
|----------|----------|--------|-----------|--------|
|          | 平均選択時間   | 誤選択率   | 平均選択時間    | 誤選択率   |
| 1000 ミリ秒 | 3.59 秒   | 0.33%  | 3.45 秒    | 0%     |
| 750 ミリ秒  | 2.96 秒   | 1.31%  | 2.86 秒    | 0.50%  |
| 500 ミリ秒  | 2.77 秒   | 18.70% | 2.50 秒    | 12.66% |

表 2 クロッシング間隔の上限の違いによる選択時間と誤選択率

Table 2 Results of each limit time of interspace of 2nd crossing.

|         | リンクターゲット |       | メニューターゲット |        |
|---------|----------|-------|-----------|--------|
|         | 平均選択時間   | 誤選択率  | 平均選択時間    | 誤選択率   |
| 500 ミリ秒 | 2.61 秒   | 2.45% | 3.19 秒    | 14.52% |
| 350 ミリ秒 | 2.43 秒   | 1.96% | 3.02 秒    | 10.18% |
| 250 ミリ秒 | 2.43 秒   | 3.34% | 3.10 秒    | 8.79%  |

「ウェイティング」と「シングルクロッシング」の二つの手法を比較のために用いた。ウェイティングの静止時間やダブルクロッシングのクロッシング間隔の上限  $T$  は、設定する時間によって選択時間や誤選択率が大きく変わってくる。我々は、予備実験として、これらのパラメータがどのぐらいの時間であれば適切であるか調査した。

予備実験は、5 名の被験者に本実験と同様の環境で時間のパラメータを変化させて試してもらった。ウェイティングの静止時間は 1000 ミリ秒・750 ミリ秒・500 ミリ秒の 3 種類、ダブルクロッシングのクロッシング間隔の上限は 500 ミリ秒・350 ミリ秒・250 ミリ秒の 3 種類で行った。この 5 名については本実験を行っていない。

静止時間に関する結果は、表 1 のとおりである。表 1 より、静止時間が短くなれば選択時間は早くなる。その一方、静止時間が短すぎる場合、誤選択率が著しく上昇してしまう。我々は、この結果を踏まえ、選択時間が早く、かつ誤選択率も低く安定していた 750 ミリ秒をウェイティングの静止時間として用いることにした。

クロッシング間隔の上限に関する結果は、表 2 のとおりである。ウェイティングの静止時間と異なり、 $T$  の値の違いによって結果にあまり大きな差は見られなかった。我々は、250 ミリ秒であれば、意図しないダブルクロッシングを防止しやすいと考え、 $T = 250$  ミリ秒とし、実験で用いることにした。

(注1) : <http://www.iplab.cs.tsukuba.ac.jp/index-j.html>

表 3 各手法の選択時間と誤選択率  
Table 3 Results of each method.

|            | リンクターゲット |               |        | メニューターゲット |               |        |
|------------|----------|---------------|--------|-----------|---------------|--------|
|            | 平均選択時間   | 平均選択時間 (誤選択無) | 誤選択率   | 平均選択時間    | 平均選択時間 (誤選択無) | 誤選択率   |
| ウェイトイング    | 3.60 秒   | 3.57 秒        | 2.76%  | 3.29 秒    | 3.29 秒        | 0%     |
| シングルクロッシング | 3.61 秒   | 3.28 秒        | 21.81% | 3.01 秒    | 2.99 秒        | 15.56% |
| ダブルクロッシング  | 3.11 秒   | 3.02 秒        | 8.94%  | 3.04 秒    | 3.01 秒        | 4.00%  |

#### 4.5 計測方法

本実験では、「ウェイトイング」、「シングルクロッシング」、「ダブルクロッシング」の三つの手法について、どのぐらい選択時間や誤選択率に差が見られるかを計測した。

選択時間は、直前に指定されたターゲットを選択した直後から次の指定されたターゲットを選択するまでの時間とした。誤選択は、指定されたターゲット以外のターゲットを選択した場合を誤選択としてカウントした。また、誤選択した場合は、正しいターゲットを選択できるまで続けてもらった。選択時間は、誤選択も含んだ選択時間以外に、誤選択を起こさなかった試行のみの選択時間の2種類を計測した。本論文では、前者を選択時間、後者を選択時間（誤選択無）と表示する。どちらの選択時間についても、選択完了までの時間であるため、ウェイトイングの静止時間やクロッシングに要した時間も含まれている。

誤選択率は（誤選択した回数）/（全選択回数）×100（%）とした。

#### 4.6 実験結果

各手法ごとの選択時間と誤選択率の結果は表3のとおりとなった。また、手法ごとで選択時間や誤選択率に違いが見られるか分散分析を行った。リンクターゲットについては、選択時間は1%水準で有意差が見られ（ $F(2, 807) = 6.66, p < 0.01$ ）、最小有意差法（LSD法）による多重比較によってウェイトイングとダブルクロッシングの間で5%水準で有意差が見られた。誤選択率についても、1%水準で有意差が見られ（ $F(2, 24) = 25.07, p < 0.01$ ）、LSD法による多重比較によりウェイトイングとシングルクロッシング、シングルクロッシングとダブルクロッシングの間で5%水準で有意差が見られた。メニューターゲットについては、三つの手法で選択時間に有意差が見られなかったが（ $F(2, 537) = 2.06, p > 0.1$ ）、誤選択率は1%水準で有意差が見られた（ $F(2, 24) = 14.18, p < 0.01$ ）。誤選択率についてLSD法を用いて多重比較した結果、ウェイトイングとシングルクロッシング、シングル

クロッシングとダブルクロッシングの間で5%水準で有意差が見られた。

リンクターゲットについては、ウェイトイングの誤選択率が低く、安定して選択が可能であった。シングルクロッシングは、誤選択率が他の手法よりも高く、選択時間もウェイトイングとほとんど同じであったが、誤選択がなかった場合の選択時間はウェイトイングよりも約0.3秒早くなった。ダブルクロッシングは選択時間は全手法で最も早かったが、誤選択率がやや高い結果となった。

メニューターゲットについては、ウェイトイングは他の手法よりも選択時間が遅い結果となったが、誤選択率は0%と最も低かった。シングルクロッシングは選択時間はダブルクロッシングとほぼ同じであったが、誤選択率が15%を超える結果となった。ダブルクロッシングは選択時間が早く、誤選択率も低めであった。

選択時間（ $t$ ）について、ターゲットの大きさや距離によって時間が変わってくるのではないかと考え、フィッツの法則の選択難易度  $ID$ （Index of Difficulty）ごとにマッピングを行った。図8と図9は、各ターゲットについて手法ごとにマッピングした結果である。 $ID$ については、 $ID = \log_2(D/W + 1)$ とした（ $D$ はターゲットまでの最短距離、 $W$ はターゲットの幅）。また、これらについて回帰分析を行った。リンクターゲットでは、ウェイトイングで  $t = 0.3167 \times ID + 2.3858$ （ $R^2 = 0.145$ ）、シングルクロッシングで  $t = 0.3065 \times ID + 2.302$ （ $R^2 = 0.2467$ ）、ダブルクロッシングで  $t = 0.2444 \times ID + 2.2708$ （ $R^2 = 0.0904$ ）となった。メニューターゲットでは、ウェイトイングで  $t = 0.699 \times ID + 1.1676$ （ $R^2 = 0.2059$ ）、シングルクロッシングで  $t = 0.7641 \times ID + 0.7814$ （ $R^2 = 0.2467$ ）、ダブルクロッシングで  $t = 0.6038 \times ID + 1.335$ （ $R^2 = 0.1446$ ）となった。なお、メニューターゲットは全て同じ大きさであるため、いずれの場合も  $W = 100$  である。

#### 4.7 被験者からのコメント

被験者から得られた最も多いコメントとして、ウェ



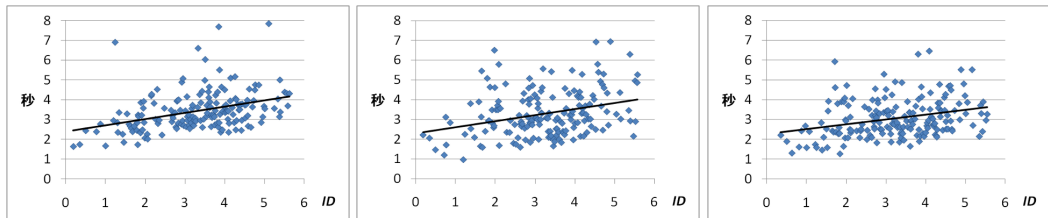


図8 リンクターゲットにおけるIDと選択時間の分布（左：ウェイトイング，中央：シングルクロッシング，右：ダブルクロッシング）

Fig.8 Map of selection time by ID at link targets (Left: waiting, center: single-crossing, right: double-crossing).

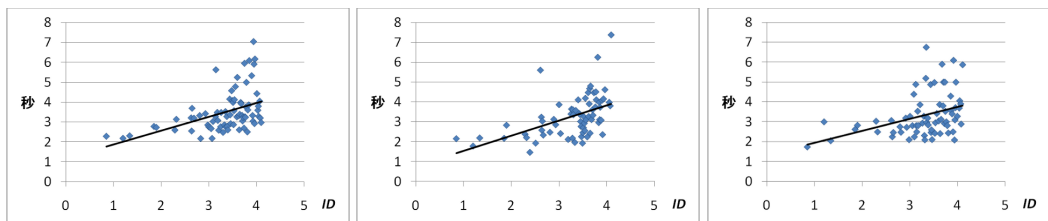


図9 メニューターゲットにおけるIDと選択時間の分布（左：ウェイトイング，中央：シングルクロッシング，右：ダブルクロッシング）

Fig.9 Map of selection time by ID at menu targets (Left: waiting, center: single-crossing, right: double-crossing).

イティングは選択が安定するという意見が多かった。しかし、5名の被験者が、空中に手をとどめる操作があるため、数分の作業でも腕が疲れたとコメントした。

シングルクロッシングについては、被験者全員が誤選択しないために迂回などをしなければならないため、より慎重にかつ神経を使って作業しなければならないとコメントした。一方で、3名の被験者がターゲットまでポインタを移動させた後は他の二つの手法よりも素早く選択ができるとコメントした。

ダブルクロッシングについては、他の手法よりも素早く選択ができるが、ターゲットが密集している場所や小さいターゲットは選択が難しく、それらを選ぶときは慎重にしなければならないというコメントが多く得られた。

#### 4.8 考察

我々は、実験結果について、リンクターゲット、メニューターゲット、選択難易度、大画面環境での利用の観点から考察を行った。

##### 4.8.1 リンクターゲット

ウェイトイングは、ポインタをある場所に静止させるため、選択動作中に他のターゲットを選択することが少なかった。そのため、密集したターゲットであっても誤選択があまり起こらず、時間はかかるが安定し

た操作が可能であった。

シングルクロッシングは他の二つの手法と異なり、誤選択を防ぐために他のターゲットを迂回しなければならない。ポインタの迂回の際、わずかなポインタのぶれによって密集地帯にある他のターゲットを選択することがたびたび見られ、その結果誤選択に大きな影響を及ぼしたと思われる。また、指定されたターゲットを選択しようとした際に、他のターゲットも一回の選択動作で選択してしまうことが多く見られた。

ダブルクロッシングは、実験結果よりウェイトイングよりも早く、誤選択率もウェイトイングよりも高いがシングルクロッシングの半分以下であったため、ターゲットの選択は比較的容易であったと思われる。そのため、クロッシングの回数を2回に増やすことでシングルクロッシングにおける問題を十分に解決できるものと思われる。

##### 4.8.2 メニューターゲット

ウェイトイングは、他の二つの手法よりもやや時間はかかったが、安定した選択ができていた。誤選択率については、メニューターゲット同士は十分な間隔が空いており、指先の認識時のぶれや手ぶれによるポインタがあまり大きくぶれなかったため、ポインタの移動時などに他のターゲットを誤って選択することもし

かったため、誤選択がなかったものと思われる。

シングルクロッシングは選択時間はリンクターゲットよりも十分に早いですが、誤選択率は15%以上と他の手法よりも高い結果であった。リンクターゲットの場合と同様に、精密なポインタの移動を要求されたため、ポインタの移動中にわずかなぶれで他のターゲットを選択してしまうことがあった。メニューターゲットに関する誤選択の内4分の1はポインタの移動中の誤選択であった。そのため、シングルクロッシングの場合には、メニューターゲットの間隔を開けたり、配置を工夫することで誤選択を減少させる必要がある。

ダブルクロッシングは、他の手法とは異なり、選択時間はリンクターゲットの場合と大きな差は見られないが、誤選択率は半分以下に減少している。ダブルクロッシングの場合、ターゲットの選択動作時に他のターゲットを誤って選択することが減少したことが要因であると考えられる。また、ポインタの移動時にポインタのぶれによる誤選択は起こらず、ポインタの移動についても問題なく行うことができた。そのため、ダブルクロッシングでは、メニューラベル同士の間隔は本実験で利用した間隔で十分であったと考えられる。

#### 4.8.3 選択難易度

図8と図9の結果から得られた近似直線を比較すると、リンクターゲットは傾きが小さく切片が大きい。一方、メニューターゲットは傾きが大きく切片が小さい傾向が見られた。図8と図9を重ねてみると、IDが高くなるとどちらのターゲットでもほぼ同様の分布となっていた。しかし、IDの小さいメニューターゲットの場合、選択時間がやや低い傾向にあった。そのため、近似直線に先のような傾向が見られたものと思われる。

メニューターゲットのIDは、移動距離のみに応じて増加する。この傾向は、選択対象が大きく、かつ近くにある場合は選択しやすいことを示していると考えられる。一方、図8と図9のIDが小さい部分を比較すると、リンクターゲットの方がより広範囲に分布している。このことから、距離が短くてもターゲットが小さい場合には選択が難しい傾向にあると思われる。

#### 4.8.4 大画面環境での利用

実際に大画面環境で利用する場合、ダブルクロッシングは、選択時間が短く、比較的安定して選択を行うことができる。そのため、大画面環境でアプリケーションを作成する場合、ダブルクロッシングをインタラクション手法として利用することで選択の早さと安定性

を兼ね備えたアプリケーションの作成ができると思われる。ウェイトिंगについては、安定して選択ができるため、より正確な操作を求められるようなアプリケーションを作成する場合に利用するのが適していると思われる。シングルクロッシングについては、選択時間は早いですが、誤選択が多いため、タッチパネルやペンスインタフェースでは適しているが、メニューの間隔を大きく開けなければならないなどの問題からポインティングハンドジェスチャのような大画面環境での利用は難しいと思われる。

## 5. 関連研究

### 5.1 ハンドジェスチャインタラクション

ハンドジェスチャを利用したインタラクション手法にはタッチパネルを利用したもの[12],[13]など様々な種類が存在する。しかし、タッチパネルを利用した手法の問題点として、大画面環境では操作範囲が手の届く範囲のみしか操作できず、画面全体を見渡しながらの操作も難しい。これらの問題を解決するための手法もいくつか研究が行われている。Frisbee [14] や drag-and-pop [15],[16] は、作業領域から遠い場所にあるオブジェクトを手元にもってきたり、送ったりすることができる。これらの手法は手の届かないオブジェクトにアクセスすることは容易になるが、画面全体を把握しながらの操作という問題は解決されない。

ハンドジェスチャをインタラクション手法として利用する試みは、1980年代のMIT Media Lab. の put-that-there [4]をはじめ、1990年代にはIBMの Dream Space [17]などの様々な研究[18],[19]が行われており、現在でも多くの研究者がハンドジェスチャを利用して離れた場所から操作を行うインタラクション手法の研究を行っている[20]~[22]。例えば、木村ら[23]は、手の動きや形状を組み合わせることで、ポインタを移動させたり、オブジェクトの選択・移動などを行うことができるインタラクション手法を提案している。これらの手法では、オブジェクトの選択に特定のハンドジェスチャを用いる。しかし、選択動作時に手の大きさや形状の変化による手の重心の移動が起こる。これにより、選択のジェスチャを行う前に合わせていたポインタの位置がずれることがあり、意図した場所で選択できないことがある。大きなオブジェクトを選択する分には問題はあまり起こらないが、ウェブのリンクやアイコンなどを選択する際には正しく選択できないことが多い。また、これらのインタラ



クション手法の多くがハンドジェスチャのみを利用して操作を行うため、操作の種類が増加するに従って必要なジェスチャの数も増加し、ユーザが一度に全ての操作を覚えことが難しくなる。その上、ユーザに利用のために訓練の時間を必要とし、ハンドジェスチャ認識のためのシステムも大がかりなものになってしまう。これらの問題から、実環境で利用する際には非常に大きな制限が発生してしまう。

東芝[24]や日立[25]では、ハンドジェスチャを利用して操作を行うテレビやPCなどを既に市販している。これらのハンドジェスチャインタラクション手法では、手を振ったり、回す、押すなどの動作を組み合わせることで操作を行っている。LGはマジックリモコン<sup>(注2)</sup>を利用して画面上のコンテンツなどを選択・操作することができる。市販されているジェスチャインタフェースは、単一の機器を扱う上で比較的少ない機能の操作を行うことを目的として設計されている。本手法は、複数のウィンドウにまたがった作業や、より多くのメニューや情報を扱った作業を想定している。そのため、本手法では、大量の情報やメニューに対して同じような動作で選択でき、かつそれらに対して素早く安定した操作ができるインタラクション手法を目指している。

## 5.2 手の認識

ハンドジェスチャインタラクションにおいて手の位置や形状の認識[26]は不可欠である。データグローブなどの機器を利用することで手の認識は容易になるが、ユーザに手全体を覆うような大がかりな機器を装着させる必要があり、また設備が高価になるなどの問題があるため、利用できる場所が限られてしまう。

ユーザにデータグローブなどのデバイスを利用しない手法として、カメラから画像を取得・解析することで手の認識を行う手法がある[27]~[29]。しかし、これらの手法の多くが、複数のカメラを利用したり、複数のPCで並列処理を行わせたりするため、認識のためのシステムが大がかりになることが多い。そのため、設置コストが高く、公共の場や家庭のリビングなどに設置することが難しい。

設置の手間やコストを削減するためには、認識手法の簡略化や認識アルゴリズムの高速化が考えられる。例えば、Dhawaleらは[30]には手に何も装着せずに、三次元の手動きと形状の組み合わせで画面とのインタラクションを行っている。位置や形状の認識には、作業領域の上部に設置されたカメラから得られた画像

から手の領域を抽出し、その手の位置や大きさの変化から三次元的位置を計測や形状の認識を行っている。この研究では、あまり大がかりな設備をしていないため、設置などのコストはあまりかからないが、この手法は太陽光などの環境の影響を大きく受けるため、認識が安定しないといった問題がある。

近年では、MicrosoftのKinect<sup>(注3)</sup>など家庭でも体の動きをとって、ゲームなどに利用することができるようになってきている。Kinectは市販されているセンサ類の中で比較的安価で購入することができる上に、三次元の計測が可能である。Leap Motion<sup>(注4)</sup>では、指先の動きを非常に高い精度で計測することが可能であり、複数の指先であっても素早く正確に認識することができる。本論文ではインタラクション手法に重点を置いているため、実験などで認識のぶれによる誤操作をより少なくし、かつ高速な認識手法としてLEDを用いた手法を利用した。しかし、これらのデバイスを利用することで、LEDを利用せずに指先の検出を行うことができるため、我々も将来的にはKinectなどの安価なセンサによる素手での指先認識を行うことで、より様々な環境で本手法が利用できるのではないかと思われる。また、複数の指を認識することも可能であるため、出している指先の本数で操作を変更するなどといった操作もできるのではないかと思われる。

## 6. む す び

我々は、大画面環境における離れた場所からのインタラクション手法として「ウェイトイング」、「シングルクロッシング」、「ダブルクロッシング」の三つの手法を提案し実装した。これらの手法では、ポインティングハンドジェスチャにより、指先の動きのみで操作ができるため、初見のユーザでもすぐに利用することが期待できる。

また、我々は、「ウェイトイング」、「シングルクロッシング」、「ダブルクロッシング」の三つの選択手法について、選択時間や誤選択率を測定した。ウェイトイングについては、選択に時間がかかるが、誤選択が非常に少なく、安定した選択手法であった。シングルクロッシングについては、選択時間こそ比較的早かったものの、誤選択が多く、大画面環境での利用は難しい

(注2) : <http://www.lg.com/jp/tv-audio-video/tv-accessories/LG-AN-MR300.jsp>

(注3) : <http://www.xbox.com/ja-JP/kinect>

(注4) : <https://leapmotion.com/>

結果となった。ダブルクロッシングは、他の手法よりも素早く選択することができ、安定して選択ができた。そのため、大画面でアプリケーションの作成を行う場合、通常はダブルクロッシング手法を使用するのが適切であり、操作の正確さが必要とされるようなアプリケーションを使用する場合にはウェイトイングを利用することが適していると思われる。

今後の課題として、一つ目は、ダブルクロッシングのための実行線やクロッシングの状況を示すためのフィードバックの与え方について工夫する必要がある。二つ目として、ダブルクロッシングや大画面の環境に合わせたメニューインタフェースの作成が挙げられる。先に述べた問題を解決したインタフェースを作成すると同時に、特定のアプリケーションに特化するのではなく、様々なアプリケーションに対応することが必要であると思われる。また、ユーザの嗜好などにも柔軟に対応できるべきだと考え、XML などを用いて様々な種類のメニューインタフェースが作成できるようなフレームワークの作成を行っていくつもりである。

## 文 献

- [1] X. Chen and J. Davis, "Multi-user laser-based interaction on large tiled displays," *Displays*, pp.205–211, 2002.
- [2] 蔵田武志, 興梠正克, 加藤文和, 大隈隆史, 坂上勝彦, "ハンドマウスとその応用: 色情報と輪郭情報に基づく手の検出と追跡," *映像学技報*, VIS2001-103, vol.25, no.85, pp.43–52, 2005.
- [3] 小山慎哉, 小野寺光, 葛岡英明, 山崎敬一, "レーザポイントによる遠隔作業指示支援システムの実用可能性," *ヒューマンインタフェースシンポジウム 2000*, pp.435–438, 2000.
- [4] R.A. Bolt, "put-that-there": Voice and gesture at the graphics interface," *SIGGRAPH '80*, pp.262–270, 1980.
- [5] S. Malik, A. Ranjan, and R. Balakrishnan, "Interacting with large displays from a distance with vision-tracked multi-finger gestural input," *UIST'05*, pp.43–52, 2005.
- [6] M. Nancel, J. Wagner, E. Pietriga, O. Chapuis, and W. Mackay, "Mid-air pan-and-zoom on wall-sized displays," *CHI '11*, pp.177–186, 2011.
- [7] T. Hisamatsu, B. Shizuki, S. Takahashi, and J. Tanaka, "A novel click-free interaction technique for large-screen interfaces," *APCHI2006*, p.10 (CD-ROM), 2006.
- [8] J. Accot and S. Zhai, "More than dotting the i's - foundation for crossing-based interfaces," *CHI' 02*, pp.73–80, 2002.
- [9] 中村 卓, 高橋 伸, 田中二郎, "ハンドジェスチャとクロッシングによるインタラクション手法," *情報処理学会全国大会講演論文集第 70 回平成 20 年 (4)*, "4-15"–"4-16", 2008.
- [10] T. Nakamura, S. Takahashi, and J. Tanaka, "Double-crossing: A new interaction technique for hand gesture interfaces," *APCHI2008*, pp.831–838, 2008.
- [11] T. Nakamura, S. Takahashi, and J. Tanaka, "One-finger interaction for ubiquitous environment," *ICIS 2010*, pp.267–272, 2010.
- [12] 小林貴訓, 佐藤洋一, 小池英樹, "Enhanceddesk のための赤外線画像を用いた実時間指先認識インターフェース," *WISS'99*, pp.49–54, 1999.
- [13] H. Benko, A.D. Wilson, and P. Baudisch, "Precise selection techniques for multi-touch screens," *CHI '06*, pp.1263–1272, 2006.
- [14] A. Khan, G. Fitzmaurice, D. Almeida, N. Burtnyk, and G. Kurtenbach, "A remote control interface for large displays," *UIST '04*, pp.127–136, 2004.
- [15] P. Baudisch, E. Cutrell, D. Robbins, M. Czerwinski, P. Tandler, B. Bederson, and A. Zierlinger, "Drag-and-pop and drag-and-pick: Techniques for accessing remote screen content on touch and pen-operated systems," *INTERACT '03*, pp.57–64, 2003.
- [16] M. Collomb, M. Hascoët, P. Baudisch, and B. Lee, "Improving drag-and-drop on wall-size displays," *GI '05*, pp.25–32, 2005.
- [17] M. Lucente, G.-J. Zwart, and A.D. George, "Visualization space: A testbed for deviceless multimodal user interface," *Intelligent Environments '98*, pp.87–92, 1998.
- [18] T. Baudel and M. Beaudouin-Lafon, "Charade: Remote control of objects using free-hand gestures," *Commun. ACM*, vol.37, no.7, pp.28–35, 1993.
- [19] R. Kjeldsen and J. Kender, "Toward the use of gesture in traditional user interfaces," *FG'96*, pp.151–156, 1996.
- [20] D. Vogel and R. Balakrishnan, "Distant freehand pointing and clicking on very large, high resolution displays," *UIST'05*, pp.33–42, 2005.
- [21] A. Bragdon and H.-S. Ko, "Gesture select: Acquiring remote targets on large displays without pointing," *CHI '11*, pp.187–196, ACM, 2011.
- [22] 塚田浩二, 安村通晃, "Ubi-finger: モバイル指向ジェスチャー入力デバイスの研究," *情処学論*, vol.43, no.12, pp.3675–3684, 2002.
- [23] 木村朝子, 柴田史久, 鶴田剛史, 酒井理生, 鬼柳牧子, 田村秀行, "ジェスチャ操作を活用する広視野電子作業空間の設計と実装," *情処学論*, vol.47, no.4, pp.1327–1339, 2006.
- [24] T. Ike, N. Kishikawa, and B. Stenger, "A real-time hand gesture interface implemented on a multi-core processor," *MVA2007*, pp.9–12, 2007.
- [25] 松原孝志, 徳永竜也, 黒澤雄一, 星野剛史, 尾崎友哉, "快適操作を提供するユーザーインタフェース技術," *日立評論*, vol.91, no.9, pp.48–53, 2009.
- [26] R.Y. Wang and J. Popović, "Real-time hand-tracking with a color glove," *ACM SIGGRAPH 2009 papers*, *SIGGRAPH '09*, pp.63:1–63:8, 2009.

- [27] S.M. Dominguez, T. Keaton, and A.H. Sayed, "Robust finger tracking for wearable computer interfacing," PUI 2001, pp.1-5, 2001.
- [28] Y. Jia, S. Li, and Y. Liu, "Tracking pointing gesture in 3d space for wearable visual interfaces," HCM '07, pp.23-30, 2007.
- [29] K. Nickel and R. Stiefelhausen, "Pointing gesture recognition based on 3d-tracking of face, hands and head orientation," ICMI'03, pp.140-146, 2003.
- [30] P. Dhawale, M. Masoodian, and B. Rogers, "Bare-hand 3d gesture input to interactive systems," CHINZ '06, pp.25-32, 2006.

(平成 24 年 8 月 2 日受付, 11 月 7 日再受付)



中村 卓

2006 筑波大・第三学群・情報学類卒.  
2008 同大学院システム情報工学研究科  
コンピュータサイエンス専攻博士前期課程  
了. 現在, 筑波大学大学院システム情報工  
学研究科コンピュータサイエンス専攻博士  
後期課程在学中. ヒューマンコンピュータ

インタラクションやユビキタスコンピューティングに興味をも  
つ. 情報処理学会, 日本ソフトウェア科学会各会員.



高橋 伸

1991 東大・理・情報科学卒. 1993 同大  
大学院理学系研究科情報科学専攻修士課程  
了. 1995 同大学院理学系研究科情報科学  
専攻博士課程中退. 現在, 筑波大学システ  
ム情報系准教授. ユーザインタフェースや  
ユビキタスコンピューティングに興味をも  
つ. ACM, 情報処理学会, 日本ソフトウェア科学会各会員.



田中 二郎 (正員)

1975 東大・理卒. 1977 同大学院理学  
系研究科修士課程了. 1984 米国ユタ大学  
大学院計算機科学科博士課程了. Ph.D. in  
Computer Science. 1985 年から 1988 年  
に (財) 新世代コンピュータ技術開発機構  
において主任研究員として第五世代コン

ピュータ核言語の技術開発に従事. 1993 から筑波大学に勤務.  
現在, 筑波大学システム情報系教授. ユビキタスコンピュ  
ーティングや未来の情報環境に興味をもつ. ACM, IEEE, 日本  
ソフトウェア科学会, ヒューマンインタフェース学会, 人工知  
能学会各会員.