大画面環境におけるポインティングハンドジェスチャによるインタラクション手法の研究

中村　卓

システム情報工学研究科

筑波大学

２０１３年　３月

概要

画面から離れた場所から操作は，画面全体を見渡しながら操作を行うことができるため，近年広く普及してきた大画面環境におけるインタラクション手法として有効な手法とのひとつである．離れた場所からのインタラクション手法のひとつにハンドジェスチャを利用した手法がある．ハンドジェスチャを利用することで，マウスなどのデバイスを利用するための水平な台などのスペースを必要とせず，駅などの公共の場やリビングなどより様々な場面で利用することが可能である．

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様々な方法があるが，利用するジェスチャによっては，操作に必要なジェスチャの数が増大したり，そのジェスチャ自体が複雑なためにユーザにとって利用しづらい手法になってしまう恐れがある．本研究では，様々なハンドジェスチャの中から手の動きに応じてポインタが移動するポインティングハンドジェスチャに注目した．ポインティングハンドジェスチャでは，利用するハンドジェスチャはポインタの移動のためのジェスチャのみであり，ポインタの動きがそのままユーザへのフィードバックとなるため，ユーザはポインタの位置にのみ注意すればよく，手の状態をあまり気にすることなく操作することができる．そのため，本研究では，このポインティングハンドジェスチャを利用した新しいインタラクション手法を作成した．新しいインタラクション手法を設計するにあたり，オブジェクトの選択手法やポインティングハンドジェスチャのためのメニューインタフェースを新たに作成した．

ポインティングハンドジェスチャでインタラクションを行う際，オブジェクトの選択などの操作も必要になるが，これらの操作は全てポインタの移動を利用した行う必要がある．オブジェクトを選択は様々なインタラクションを行う上で不可欠であり，一定時間静止したり，特定の箇所を交差するなど様々な方法がある．しかし，これらの手法は手ブレなどの影響を受けやすかったり，違うターゲットを選択しやすかったりするなどの問題がある．そこで，これらの問題を解決する手法としてクロッシング手法を利用したダブルクロッシングを作成した．ダブルクロッシングを利用することで，他の手法よりも素早く選択でき，かつ誤選択なども少なくすることができる手法となっている．

オブジェクトの選択手法以外に，ポインティングハンドジェスチャに合わせたメニューインタフェースの作成も重要である．従来のメニューインタフェースでは，メニュー項目が小さかったり，密集していたりするためポインティングハンドジェスチャには不向きである．ポインタの移動軌跡を利用する方法も考えられるが，ユーザの思い通りの軌跡を描くことが難しいため，誤操作などが増大するなどの問題が発生してしまう．本研究では，パレットインタフェースの枠組みを新たに作成し，そのプロトタイプとして，C.Palette を作成した．パレットインタフェースでは，ポインタの周囲にメニューを表示させることで，大画面環境における視線の移動を減少させ，かつメニュー操作を行いたい時にすぐに利用できるようにした．また，C.Palette では，メニューをユーザの嗜好や作業内容に合わせて変更できるように，XML とインタフェースビルダーを利用することで自由に作成できるようにした．

目 次

第 1 章　序論 **1**
1.1　本研究の目的 2
1.2　論文構成 3

第 2 章　大画面とのインタラクション **4**
2.1　既存のインタラクション手法 4
2.2　ハンドジェスチャによらないインタラクション 5
2.3　タッチパネルの利用インタラクション 6
2.4　ハンドジェスチャインタラクション 6
2.5　ハンドジェスチャの種類 7
2.5.1　静的ジェスチャ 8
2.5.2　動的ジェスチャ 8
2.5.3　混合ジェスチャ 9
2.5.4　ポインティングハンドジェスチャ 10
2.6　手の認識 11
2.7　ポインティングハンドジェスチャにおける選択手法 12
2.7.1　ウェイティング 12
2.7.2　クロッシング 13
2.8　大画面におけるメニューインタフェース 13
2.8.1　既存メニューの利用 14
プルダウンメニュー 14
ポップアップメニュー 15
2.8.2　新しいメニューインタフェースの利用 15

第 3 章　ダブルクロッシング **18**
3.1　ダブルクロッシングの設計 18
3.2　ダブルクロッシングを用いたインタラクション 19
3.3　実装 19
3.3.1　指先の認識とポインタの移動 19
3.3.2　クロッシングの認識 21
3.4　ポインティングハンドジェスチャにおける選択手法の評価 21
3.4.1　実験内容 22

3.4.2 リンクターゲットとメニューターゲット 22
3.4.3 実験環境 23
3.4.4 計測方法 23
3.4.5 実験手順 23
3.4.6 予備実験 24
ウェイティング 24
ダブルクロッシング 25
3.4.7 実験結果 25
3.4.8 被験者からのコメント 27
3.5 考察 27
3.5.1 リンクターゲット 28
3.5.2 メニューターゲット 29
3.5.3 選択難易度 29
3.5.4 選択時間に影響を及ぼすパラメータについて 29
ウェイティング 31
シングルクロッシング 31
ダブルクロッシング 32
3.6 誤選択率に影響を及ぼすパラメータについて 34
3.6.1 ウェイティング 34
3.6.2 シングルクロッシング 34
3.6.3 ダブルクロッシング 35
3.6.4 大画面環境での利用 36

第 4 章 パレットインタフェース 37
4.1 設計指針 37
4.2 パレットインタフェースの基本機能 38
4.2.1 パレットインタフェースの移動 38
4.2.2 メニューの切り替え 39
4.2.3 表示/非表示の切り替え 39
4.3 表示するメニューの管理 39
4.4 C.Palette 40
4.4.1 ヘッダー 40
4.4.2 パレットウィンドウ 41
4.4.3 メニューラベル 41
4.4.4 クリックラベル 42
4.4.5 ウィンドウの移動 43
境界領域による移動 43
クリックラベルによる移動 44
4.4.6 パレットウィンドウの切り替え 44

4.4.7 実装 44
4.5 パレットウィンドウの作成 45
4.5.1 パレットウィンドウ作成の流れ 46
パレットウィンドウの設定 46
メニューラベルの作成 47
4.5.2 クリックラベルの設定 48
4.5.3 メニュー操作の割り当て方法 49
4.6 パレットインタフェースに関する実験 49
4.6.1 実験内容 49
4.6.2 インタフェースの詳細 50
4.6.3 実験環境 51
4.6.4 実験手順 51
4.6.5 実験結果 52
4.6.6 被験者からのコメント 52
4.6.7 考察 53
固定メニュー 53
パレットインタフェース 54

第 5 章 まとめ **55**
5.1 本研究の貢献 55
5.2 今後の展望 56
5.2.1 ポインタの移動を利用した選択手法に関する展望 56
5.2.2 ポインティングハンドジェスチャ向けメニューインタフェースに関する展望 56

謝辞 **57**

参考文献 **58**

著者論文リスト **63**

付録：C.Palette の XML の文書形式 **65**

図 目 次

1.1 家庭における大画面の将来像 1
1.2 離れた場所からのインタラクション例 2

2.1 静的ジェスチャの例 7
2.2 動的ジェスチャの例 8
2.3 ポインティングハンドジェスチャの例（左：手の動き，右：ポインタの動き） 10
2.4 左：ウェイティング，右：(シングル）クロッシング 12
2.5 プルダウンメニュー 14
2.6 ポップアップメニュー 15
2.7 移動方向と回転角度を利用したメニューインタフェース 16
2.8 （a）指先の移動方向による選択，(b）指先の回転角度による選択 17

3.1 （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 18
3.2 大画面ポインティングシステムの構成 20
3.3 LED デバイス（左）と装着した様子（右） 20
3.4 クロッシングの認識 21
3.5 実験におけるブラウザとメニューの配置 22
3.6 リンクターゲットの選択時間と誤選択率 25
3.7 メニューターゲットの選択時間と誤選択率 25
3.8 誤選択の有無による選択時間の違い（リンクターゲット） 26
3.9 誤選択の有無による選択時間の違い（メニューターゲット） 26
3.10 リンクターゲットにおける ID と選択時間の分布（上：ウェイティング，左下：シングルクロッシング，右下：ダブルクロッシング） 27
3.11 メニューターゲットにおける ID と選択時間の分布（上：ウェイティング，左下：シングルクロッシング，右下：ダブルクロッシング） 28
3.12 ウェイティングに関する選択時間の分布（左：ターゲットの幅と距離，右：ターゲットの高さと距離） 30
3.13 パラメータ毎のウェイティングの選択時間の分布（上：距離ごとの分布，左：ターゲットの幅による分布，右：ターゲットの高さによる分布） 30
3.14 シングルクロッシングに関する選択時間の分布 32
3.15 シングルクロッシングの距離による選択時間の分布 32

3.16 ダブルクロッシングに関する選択時間の分布 33
3.17 ダブルクロッシングの距離による選択時間の分布 33
3.18 シングルクロッシングに関する誤選択率（左：最も近い他のターゲットまでの距離と誤選択率，右：指定されたターゲットまでの距離と誤選択率） 34
3.19 ダブルクロッシングの他のターゲットまでの距離毎の誤選択率の分布 35

4.1 パレットインタフェースにおけるメニューの移動イメージ 38
4.2 C.Palette の概観 40
4.3 メニューラベル 41
4.4 クリックラベル 41
4.5 クリックラベルの移動と選択動作 42
4.6 境界領域 43
4.7 メニューウィンドウ 45
4.8 プロパティウィンドウ 46
4.9 パレットウィンドウ作成過程 47
4.10 固定メニューの配置 50
4.11 パレットインタフェースの配置 50
4.12 タスク 1 の画面の配置 51
4.13 タスク 2 の画面の配置 52
4.14 タスク 1 の実験結果 53
4.15 タスク 2 の実験結果 54

表 目 次

3.1 各静止時間の選択時間と誤選択率 24
3.2 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 25

第1章　序論

近年，ディスプレイ技術の発達により，ディスプレイはより大型になり，価格も減少してきている．また，複数のプロジェクタを利用した壁サイズの大画面も登場してきている．

大画面環境の発達に伴い，様々な場所に設置され，様々な情報が表示されている．例えば，駅に設置されている場合，商品の広告を流したり，駅周辺の地図や観光情報を表示している [66, 35]．また，大学に設置されている大画面では，掲示板の様に利用され，休講情報など大学に関する情報が常に表示され続けている．

家庭にも 50 インチ以上の大画面が広く普及し，テレビなどに利用されている．近い将来，家庭の大画面はテレビ以外にも広く利用されると考えられる．図 1.1 のように様々な家電がネットワークに接続され，ユーザの目に見えない状態で存在する．

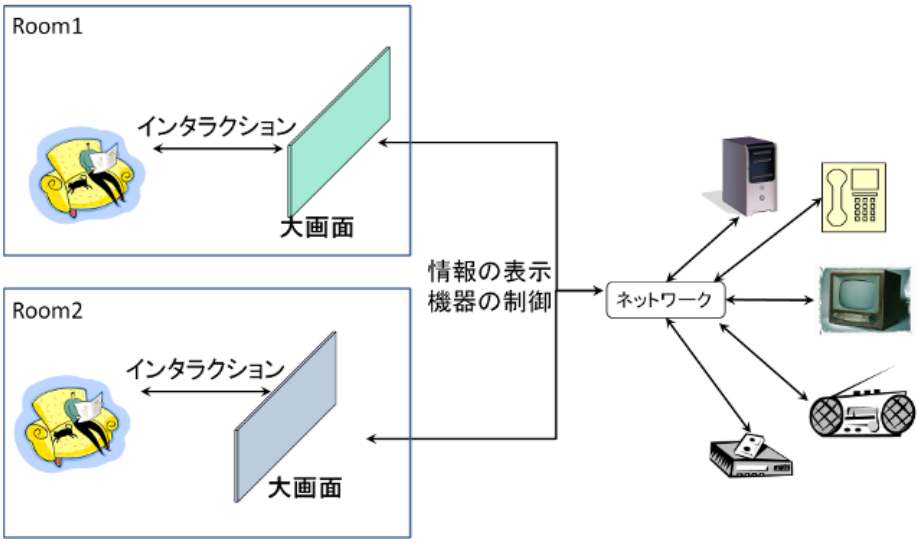


図 1.1: 家庭における大画面の将来像

大画面を利用する利点として，ひとつに従来のサイズのディスプレイと比べ，より大量の情報を一度に表示することが可能である．また，各場所に設置してある大画面をネットワークに接続して管理することで，一元管理することができ，情報の更新などもより容易になる．

しかし，その一方で実際に設置されている大画面の多くは，情報を見せるためだけに設置されていることが多く，その情報を見る側からの操作は難しいのが現状である．その理由として，ひとつは大画面がおかれる環境は従来のようなパーソナルコンピュータ（PC) を利用

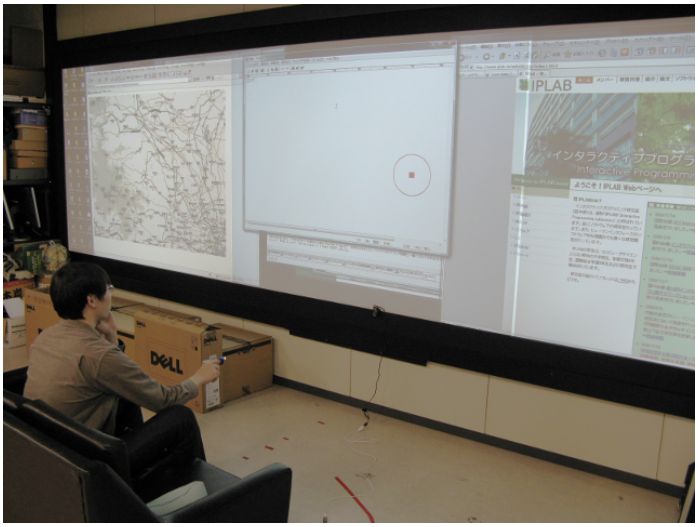


図 1.2: 離れた場所からのインタラクション例

してきた環境とは異なり，必ずしも水平な台などに設置されているわけではないため，PC 環境で利用されてきたマウスやキーボードを利用することができず，従来のインタラクション手法を利用することが難しい点が挙げられる．また，タッチパネルを利用することも挙げられるが，タッチパネルは設置コストがかかり，手の届かない範囲のインタラクションが困難であるなどの問題が発生する．このように，従来のインタラクション手法などを利用することが難しため，新たなインタラクション技術が必要になってくる．そのために，図 1.2 のように画面から離れた場所からインタラクションを行う必要があると考えられる．

離れた場所からのインタラクション手法の一つにハンドジェスチャを利用した手法がある．しかし，既存のハンドジェスチャインタラクション手法の多くは，インタラクション手法が複雑であり，そのインタラクションを利用するためには事前に練習を必要とするため，初見のユーザには利用することが難しい．複雑なインタラクション手法は必要な設備なども大がかりになりがちであるため，コスト面でも問題が生じ，ユーザへの身体的負担も増大してしまう恐れがある．このような問題により，ハンドジェスチャによるインタラクションは利用できる環境が制限されたり，ユーザが利用を敬遠する恐れがあるため，これらの問題を解決したインタラクション手法が必要になる．

1.1 本研究の目的

本研究では，大画面環境において，様々な環境でもユーザ側が大画面とインタラクションを行うことが可能な新しいインタラクション手法の設計を行う．そのために，本研究ではハンドジェスチャの中でもポインティングハンドジェスチャを利用したインタラクション手法に着目し，新たなインタラクション手法の設計とメニューインタフェースの作成を行う．本

研究では，主に家庭などにある大画面上で作業を行うことを想定し，大画面から 3m ほど離れた場所からインタラクションを行うことを前提としたインタラクション手法を作成することにする．その際，ハンドジェスチャを利用することによって生じる問題点を解決し，ユーザへの身体的な負担などを少なくする．また，一つ一つの操作を素早く行うことができ，連続操作も容易であるようなインタラクション手法の作成し，初見のユーザでもすぐに利用することができるような手法を目指す.

1.2 論文構成

本論文では，第 2 章で，まず既存のインタラクション手法の問題点やハンドジェスチャインタラクション，ハンドジェスチャにおけるメニューインタフェースなどについて述べる．第 3 章でポインティングハンドジェスチャとポインタを用いた選択手法であるダブルクロッシングについて説明し，既存手法との比較や既存手法を含めた各手法の特徴について考察した．第 4 章で，ポインティングハンドジェスチャ向けにパレットインタフェースの枠組みについて述べ，その試作インタフェースである C.Palette について述べる．

第2章　大画面とのインタラクション

2.1　既存のインタラクション手法

大画面環境において，従来のマウスやキーボードを利用したインタラクションがそのまま利用されることが多い．しかし，マウスやキーボードは元々は従来のデスクトップ環境での利用を想定したインタラクション手法であるので，画面が大きくなることで，マウスの移動量が増大したりするため，大画面環境で利用するには不向きなインタラクション手法であるといえる．また，駅などの公共の場に設置された大画面を操作しようとした場合，マウスを利用するための机などとの水平で広いスペースを確保することが難しく，利用可能な場所が限られてしまう．キーボードの操作などに慣れていない場合，手元の状態を確認しながら操作を行わなければならず，どうしても画面から視線を離さなければならず，視線の移動が増えてしまう．ほかの問題点として，その視線移動の際にポインタなどを見失ってしまう恐れがあり，それらを見つけるために時間がかかり，操作に余計時間がかかってしまう恐れがある．Ninja Cursor[31] のように複数のポインタを用意してそれらをひとつのマウスで操作・選択する手法もあるが，操作が特殊であったり，どのポインタが選択状態であるかを確認したりする必要があるため，視線移動の増大などの問題は解決されない．

マウスやキーボードを利用しない手法としては，リモートコントローラ（リモコン）などを利用する手法も考えられる．リモコンを利用することで，離れた場所から操作することが可能であり，利用に際してスペースを必要としない．また，種類によってはポインタなどの操作を行うことができ実際に画面とのインタラクションをマウス同様におこうことができるため，キーボードやマウスを利用するよりも大画面に適していると考えられる．例えば，LGではリモコンを利用して大画面とのインタラクションを行う商品を実際に販売している[1]．しかし，リモコンを利用した場合の問題点として，ボタンの数が多く手元を見ながらでないと操作が難しい．また，種類によってボタンの配置などが異なるため，どのリモコンでも同じように操作ができるというわけではない．

そのため，大画面環境では，既存のインタフェースによらない新しいインタラクション手法が必要である．新しいインタラクションでは，大画面を見渡しながら操作したり，操作中も作業している領域から目を離さずに操作ができる必要がある．また，大量の情報やメニューに対して素早く安定した操作ができるべきである．このような新しいインタラクション手法として，スティックなどのオブジェクトを利用したハンドジェスチャによらない手法や，タッチパネルを利用したインタラクション手法，ハンドジェスチャを用いたインタラクション手

[1] http://www.lg.com/jp/tv-audio-video/tv-accessories/LG-AN-MR300.jsp

法などが考えられる．

2.2 ハンドジェスチャによらないインタラクション

ハンドジェスチャを利用しないインタラクション手法には，様々なオブジェクトにセンサを取り付けてそれらを利用する手法や，レーザポインタで場所を指し示したり携帯をカメラに向けて操作を行う手法などがある．

オブジェクトを利用したインタラクション手法では，タッチパネルなどとは異なり，オブジェクトを手で直接持ち，そのオブジェクトに応じたジェスチャを利用してインタラクションを行う [4, 5]．例えば，棒の場合は振ったり傾けたりすることで，ヘッドフォンの場合は装着したり横を叩くなどといった動作をジェスチャとして利用する．また，VisionWand[14] では，棒状の専用のデバイスを作成し，そのデバイスを用いることで画面とのインタラクションを行っている．オブジェクトを利用した場合，そのオブジェクトにあったアプリケーション（例えば，ヘッドフォンの場合は音楽プレーヤなど）では，ジェスチャと操作が結びつきやすいため有効ではある．しかし，大画面の操作はある特定のアプリケーションに特化した操作ではないため，オブジェクトを用いたジェスチャと大画面の操作が結びつきにくい．そのため，ユーザが操作に慣れるのに時間がかかる恐れがあり，操作性が低下する恐れがある．また，インタラクションの際はオブジェクトなどにセンサを取り付け，そのオブジェクトを常に手元に置いておく必要があるため，画面の操作に適していない．

携帯を利用する手法では，例えば，Miyaoku ら [38] は携帯のディスプレイから特定の色差信号をカメラに認識させることで，大画面とのインタラクションを行うことができる手法を開発した．この手法では，携帯電話があれば誰でも大画面とのインタラクションが可能になるが，色差信号を作成・送信するための専用のアプリケーションをユーザが予め用意する必要がある．ユーザ側に特定のデバイスやアプリケーションを用意させる場合，事前に準備させることによりユーザ側の負担が大きくなる恐れがある．また，ユーザ側にアプリケーションを用意させる場合，システム側がアップデートなどにより新しいバージョンのアプリケーションが必要になった場合には，ユーザはアプリケーションをアップデートしなければならないなど，ユーザへの負担がより増大してしまうなどの問題が発生する．

レーザポインタでインタラクションを行う手法もいくつかあり [16, 55, 25]，これらの手法では，画面をレーザポインタで直接指し示し，その位置をシステムが認識してポインタを移動させるシステムとなっている．レーザポインタで直接画面上の一点を指し示すため，ポインタを動かしやすいが，手ブレなどの影響を受けやすく，ポインタがぶれやすいといった問題がある．また，画面のサイズが大きすぎる場合，一台のカメラで大画面を撮ることが難しくなり，複数台必要になり，設置やカメラのキャリブレーションが難しく，環境光の影響を受けやすいなどの問題も発生し，利用可能な場所が限られてしまう．

このように，オブジェクトなどを利用したインタラクション手法は，利用する機器や環境，特定のアプリケーションが必要などといった制限を多く受けてしまい，利用可能な場面が限られてしまう．本研究では，より少ない制限で様々な環境で利用することが可能であるイン

タラクション手法を作成するために，タッチパネルやハンドジェスチャに着目した．

2.3 タッチパネルの利用インタラクション

タッチパネルを利用したインタラクションでは，画面に直接触れることで操作をするため，選択したいオブジェクトなどを簡単に選択することができ，ユーザへのフィードバックが分かりやすいなどの利点がある．タッチパネルを利用したインタラクション手法には，さまざまな手法があり多くの研究が行われている [41, 56, 65]．例えば，Benko ら [9] は，複数の指を利用してディスプレイの一部を拡大したり操作したりするために，マルチタッチを利用したインタラクション手法を設計した．

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．タッチパネルは液晶ディスプレイなど用いた 60 インチ程度の大画面上では有効ではあるが，壁面サイズの大画面では問題が発生する．タッチパネルの性質上，ユーザは画面に密着した状態で操作を行わなければならないため，ユーザの作業領域付近の画面の状態を確認することができても，大画面全体を把握することが難しい．また，画面に直接触れて操作するため，手の届かない範囲を操作することも難しい．手が届かない範囲の操作手法については，いくつか研究が行われている [10, 18]．例えば，Frisbee[29] や drag-and-pop[8] は，作業領域から遠い場所にあるオブジェクトを手元に持ってきたり，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．

2.4 ハンドジェスチャインタラクション

離れた場所からのインタラクション手法のひとつとして，ハンドジェスチャを利用する手法がある．身体を利用したジェスチャは，日常生活中の非言語的コミュニケーションの一つとしてよく利用されており，そのジェスチャを操作に利用することは自然なことと言える．特に，手は人間の身体の中でも最も動かしやすい部位のひとつであり，多彩な表現が可能である．ハンドジェスチャインタラクションでは，この手のジェスチャとコンピュータの操作を結びつけることで，様々なことを行うことができる．

ハンドジェスチャインタラクションの特徴として，手の動きを利用するため，水平な台などのスペースを必要とせず，カメラなどを利用して手を認識させることで離れた場所からの操作が可能である点が挙げられる．離れた場所から操作が行えるため，画面の一部に注目して操作するだけでなく，画面全体を見渡しながら操作したりすることも可能である．ハンドジェスチャの数は非常に多く，様々な表現が可能であるため，様々な種類のインタラクション手法を設計可能である．ハンドジェスチャを利用した大画面とのインタラクション手法は，1980 年代の MIT Media Lab．の put-that-there[11] をはじめ，1990 年代には IBM の Dream Space[34] などの様々な研究 [7, 30] が行われており，2000 年代に入ってからも Microsoft Research[45] など，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きている [7, 30, 37, 12]．また，大画面以外にも，拡張現実や図形描画のためのインタラクション手法 [59, 63, 46] に利用するなど，様々な場面で利用することができる．

ハンドジェスチャを利用することで，本研究が目指すインタラクション手法を作成することができ，かつ様々な場面でも利用可能ではないかと考えた．そこで，本研究では，このハンドジェスチャを利用したインタラクション手法に注目し，これらを用いたインタラクション手法の作成を試みた．

2.5　ハンドジェスチャの種類

ハンドジェスチャは先に述べたとおり，数が非常に多く，ハンドジェスチャインタラクション手法を設計するにあたり，どのようなジェスチャを利用するかが非常に重要となってくる．一口にハンドジェスチャといっても，種類や数が非常に多く，操作と結びつきにくいジェスチャを利用してしまうとユーザが覚えづらく，使いにくいインタラクション手法となってしまう恐れがある．また，ハンドジェスチャは自由度が高い反面，いくらでも複雑なジェスチャを作成することができるため，利用するジェスチャは慎重に選ぶ必要がある．

ハンドジェスチャを利用した手法を作成するにあたり，まずはハンドジェスチャの種類や特徴を知る必要がある．ハンドジェスチャは大まかに,「静的ジェスチャ」,「動的ジェスチャ」,「混合ジェスチャ」,「ポインティングハンドジェスチャ」の 4 種類に分けることができると考えられる．

図 2.1: 静的ジェスチャの例

2.5.1　静的ジェスチャ

静的ジェスチャは，図 2.1 のように手を握ったり，開く，ピースするなど特定の手の形のみで構成されるジェスチャである [44]．静的ジェスチャでは，手の位置や動きなどを利用せず，そのときの手の形のみを利用するため，カメラなどの認識装置の範囲内であればどこでもインタラクションを行うことが可能である．静的ジェスチャに操作を割り当てることで，ショートカットキーのように利用することができるが，動きを利用しないため，ポインタの移動などオブジェクトを動かすなどの操作にはあまり向かない．

図 2.2: 動的ジェスチャの例

2.5.2　動的ジェスチャ

動的ジェスチャは，静的ジェスチャと異なり，手の形を利用せず，図 2.2 のような手の動きや大きさ，移動速度，軌跡などをジェスチャとして利用するハンドジェスチャである [26, 47, 2]．Mafhuiurr らは，指先の動きを認識し，その動きに応じて様々な操作を実行するインタラクション手法を提案している [43]．また，Cheng らは，仮想的なタッチスクリーンを作り，そのスクリーンに指が触れることでタッチパネルのような操作を行っている [17]．

動的ジェスチャでは，操作が全て手の動きのみで決まるため，手が握られているか開いているかということを気にすることなく操作することができる．手を動かすことで操作を行うため，ポインタやオブジェクトの移動など動かす操作を行いやすい．ほかの操作（例えば，音楽プレーヤーの再生 など）も特定の手の動きを検出して，その動きに対応した操作を実行することで実現している．また，画面のスクロールや値を増減させるなどの操作も円を描き続けるなどのジェスチャで繰り返して行うことも可能である．

その一方，オブジェクトの選択についても，動きなどを用いて操作を行う必要がある．また，ペンやタッチパネルを利用する場合と異なり，きれいな軌跡を描くことが難しいため，ユーザが意図した軌跡を描けないことによって正しい操作が選択されない恐れがある．

2.5.3 混合ジェスチャ

混合ジェスチャは，静的ジェスチャと動的ジェスチャを組み合わせてハンドジェスチャを作成し，そのジェスチャに操作を割り当てることで様々な操作を行う．2 種類のジェスチャを組み合わせるため，他のハンドジェスチャと比べても種類が非常に多く，より複雑なハンドジェスチャが利用可能である．

前の 2 つのジェスチャと比べ，より様々な操作を行うことが可能であり，混合ジェスチャを利用したインタラクション手法は様々な手法が提案されている [15, 24, 50, 63]．例えば，木村ら [64, 68] 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，オブジェクトを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．Vogel ら [49] は手の動きでポインタを動かし，特定のジェスチャを行うことでターゲットの選択などを行うことができるインタラクション手法を設計した．また，このインタラクション手法では，手の形状によってポインタの動きが変化する．また，Malik ら [36] は ID タグのついたタッチパッドを利用して離れた場所から両手によるジェスチャインタフェースを設計した．ユーザはタッチパッドの領域を切り分けることにより，大まかなポインタ移動と細かい移動を使い分けることが可能であり，特定のジェスチャによって様々なの操作を実行することができる．

混合ジェスチャによるインタラクションでは，手の動きなどを利用してポインタの移動を行い，手を握ったり手を前に押し出す，指を曲げるなどのジェスチャを用いてオブジェクトの選択を行う．もしくは，指差し [54] を利用してターゲットを指示して選択する手法が大半である．しかし，これらの選択手法では，選択動作時に手の大きさや形状が変化し，手の重心の移動などもみられる．そのため，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．指さしによる選択についても，どこを指さしているのかをシステムが正確に把握することは困難であるため，正確な位置を指すことは難しい．そのため，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．他のハンドジェスチャと比べ，利用するジェスチャが複雑になりやすいため，インタラクション手法自体も複雑になりやすい．また，それによりユーザが操作に必要なジェスチャを覚えるのに時間がかかったり，慣れるまでに時間がかかったりする恐れがある．そのため，混合ジェスチャでインタラクション手法を設計する際には，利用するハンドジェスチャの数や種類に気をつける必要がある．

混合ジェスチャによるインタラクションは，テレビなどの家電などにも利用されており，東芝 [27] や日立 [53] 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 などをすでに市販している．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が，ハンドジェスチャの数が多く覚えにくいなどの問題がある．

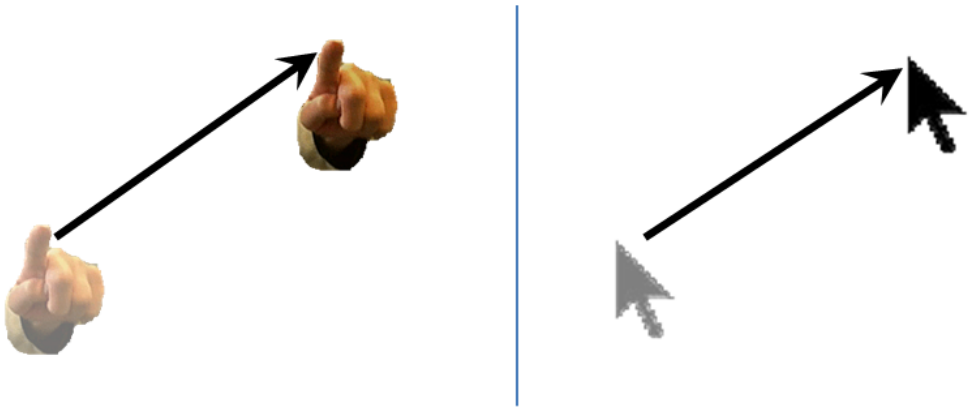


図 2.3: ポインティングハンドジェスチャの例（左：手の動き，右：ポインタの動き）

2.5.4　ポインティングハンドジェスチャ

この方式では認識率が問題になることが多い手の形状推定などは用いず，指先の位置に対応するポインタの動きだけを入力として利用し，ユーザは画面に表示されるポインタの位置を見ながら操作を行う [48]．例えば，図 2.3 の左図の様にユーザが手や指先を移動させた場合，ポインタは図 2.3 の右図の様にその動きに合わせて移動する．そのため，自分が正しいジェスチャをしているかどうかが初心者にも分かりやすいと考えられる．また，ポインタの位置だけでなく，手形状自体（握るとか開くとか）を操作に用いると，手形状を変化させる際にポインタの位置もずれてしまうという問題が起こりやすいが，ポインティングハンドジェスチャでは指先の動きだけをジェスチャとして利用するのでこの問題も起きにくい．

このジェスチャは利用者の手の動きとポインタの動きが対応しているため，ハンドジェスチャの中で分かりやすいジェスチャの一つであるため，初見のユーザでもすぐに利用することが可能である．大画面環境で離れた場所からインタラクションでは，画面が大きく，ユーザと大画面が離れているため，手元を確認しながらの操作が難しが，ポインティングハンドジェスチャでは，ポインタの動きがフィードバックとしてユーザに与えられるため，ユーザは手元を見ずに画面のみを見ながら操作することができる．

本研究で想定する環境での操作に適しており，アプリケーションの操作についても，専用のメニューを用意する場合であってもメニュー選択時にキーボードイベントやマウスイベントをエミュレートしてキーボードショートカットを呼び出すことで，既存のアプリケーションの操作を行うことが可能である．また，Windows などの既存の OS で用いられているグラフィカルユーザインタフェース（GUI）上での操作が可能なため，既存のアプリケーションとの相性もよい．そのため，ポインティングハンドジェスチャのみを利用することで初見の利用者でもすぐに使うことができるインタラクション手法が設計できるのではないかと考えた．

ポインティングハンドジェスチャでインタラクション手法を設計する場合，ポインティング操作のみでメニューをどのようにして選択するかという問題を解決しなければならない．しかし，メニュー選択を行う場合，マウスの左クリックに相当する操作が必要となる．そこで，

本研究では特定のポインティング動作でメニューの選択を行うことを試みた．

2.6 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 [40, 67, 60] 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．データグローブなどの大掛かりな機器を装着せずに認識する方法として，様々な色のついたカラーグローブを装着させる方法も提案されている [51]．この手法では，認識した各色がどの位置にあるかによって手の位置や形を推定し，認識している．このようなカラーグローブを利用することで設備が効果になる問題は解決できるが，利用するカラーグローブ自体は特殊なものであるため，利用可能な場面が限られてしまうといった問題は解決されない．

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 [20, 28, 57]．これらの手法では，肌色の領域や輪郭を抽出して，パターンや学習データなどとマッチングさせることで認識を行う．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 [19] らは手に何も装着せずに，三次元の手の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の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[2]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[3]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．本研究では，インタラクション手法に重点を置いているため，実験などで認識のブレ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，Kinect 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，本研究も将来的には Kinect などの安価なセンサによる素手での指先認識を行うつもりである．

[2] http://www.xbox.com/ja-JP/kinect
[3] https://leapmotion.com/

2.7　ポインティングハンドジェスチャにおける選択手法

Windows などの既存のグラフィカルユーザインタフェース（GUI）等を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（例えば，マウスの左クリック など）が必要である．そのため，ポインティングハンドジェスチャでは，どのようなポインタの動きを利用して選択操作を実現するかはインタラクション手法を設計する上で非常に重要になってくる．ユーザに分かりやすいインタラクション手法を実現するためには，複雑なポインタの動きによる選択ではなく，簡単な動きで行えるべきである．また，同時に意図しない選択や誤選択を防ぐことができるような動きを定義する必要がある．本研究では，これらの問題を解決するためにウェイティングとクロッシングと呼ばれる手法に着目した．

2.7.1　ウェイティング

ウェイティングは，図 2.4 の左のようにポインタなどをある特定の場所で一定時間静止させることで選択を行う手法であり，ペンベースインタフェースやレーザーポインタなどで直接指し示すことによって操作を行うインタラクション手法などでもよく利用される．この手法は，ポインタを一カ所に留めて奥だけであるため，ポインティングハンドジェスチャにおいて最も単純な選択手法のひとつである．初めてのユーザであっても，すぐに利用することができるインタラクション手法を作成することができる．

ポインティングハンドジェスチャでは，手を空中にかざした状態で操作を行うため，選択の度に手を空中で留める必要がある．空中で手を留める場合，手を動かしている時よりも手ブレが起きやすく，それに伴いポインタもぶれやすくなる．また，ウェイティングは静止時間によって操作性が大きく変わる恐れがあるため，適切な静止時間を設定しなければ，意図しない選択が増えてしまったり，手ブレによって中々選択することができないなどの問題が起こりかねない．

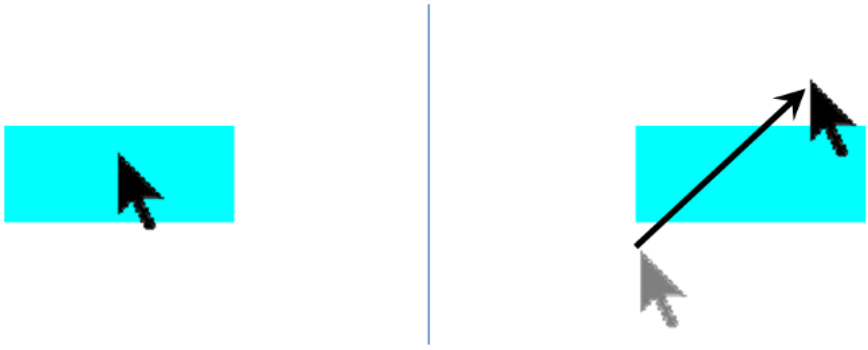


図 2.4: 左：ウェイティング，右：（シングル）クロッシング

2.7.2 クロッシング

クロッシング [1] 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．この手法は，元々はペンベースインタフェースでメニュー選択などを素早く行うために設計された手法である．本研究ではポインタの移動の軌跡を利用することによって，ペンベースインタフェースにおけるこれらの手法をハンドジェスチャでも利用できるのではないかと考え，ポインティングハンドジェスチャでクロッシングを利用することでウェイティ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．クロッシングは図 2.4 の右図のようにポインタなどがターゲットと交差するだけの簡単な操作であり，初見のユーザでもすぐに利用できる．クロッシングはウェイティングと異なり，待つ時間が存在しないため，ウェイティングよりも素早い選択が期待できる．

クロッシングは，ハンドジェスチャにあわせて作られたインタラクション手法ではないため，いくつか問題が発生する恐れがある．例えば，ペンベースインタフェースでは，ペンをタブレットなどから離すという動作が存在するが，ポインティングハンドジェスチャではそれに相当する操作が存在しない．また，ペンベースインタフェースで利用する場合と異なり，軌跡を途中で切ってまた別の場所から軌跡を描き始めることが難しい．それにより，ポインティングハンドジェスチャでは，他のオブジェクトを選択しないためにポインタを迂回させる必要があり，オブジェクトの存在に気づいていない場合には誤選択を起こしてしまう恐れがある．ハンドジェスチャの特性に合わせて改良する必要がある．そこで，本研究ではクロッシングをハンドジェスチャ向けに改良した手法として「ダブルクロッシング」を設計した．ダブルクロッシングの詳細については，第 3 章で説明する．

2.8 大画面におけるメニューインタフェース

ポインティングハンドジェスチャでは，ポインタの動きを利用した選択手法以外に，メニューインタフェースにも着目する必要がある．ポインティングハンドジェスチャでは，様々なポインタの軌跡を定義し，それらを GUI の操作に割り当てることでも様々な操作が可能である．しかし，そのためには必要な軌跡の数が膨大な数になり，ユーザが覚えにくいなどの問題から，インタラクション手法として利用しにくいものとなってしまう恐れがある．

この問題を解決するための方法として，メニューインタフェースとの併用が挙げられる．メニュー操作と併用することで，操作に必要な軌跡を少なくすることができ，またユーザはどのような操作が可能であるかを視覚的に確認することができる．そのため，ポインティングハンドジェスチャとメニューインタフェースの併用はインタラクション手法として適切であると思われる．利用するメニューインタフェースについては，既存の GUI のメニューを利用するか，専用のメニューを別個作成して利用する方法がある．

2.8.1　既存メニューの利用

一般的な WIMP インタフェースにおけるメニューとして，プルダウンメニューやポップアップメニューが挙げられる．プルダウンメニューやポップアップメニューでは，メニューがリストのように並んで表示され，マウスの左クリックなどのような選択操作を行うことでメニューを選択して実行する．ハンドジェスチャを利用する場合も同様にポインタの移動とメニューの選択という 2 種類のジェスチャが必要になる．

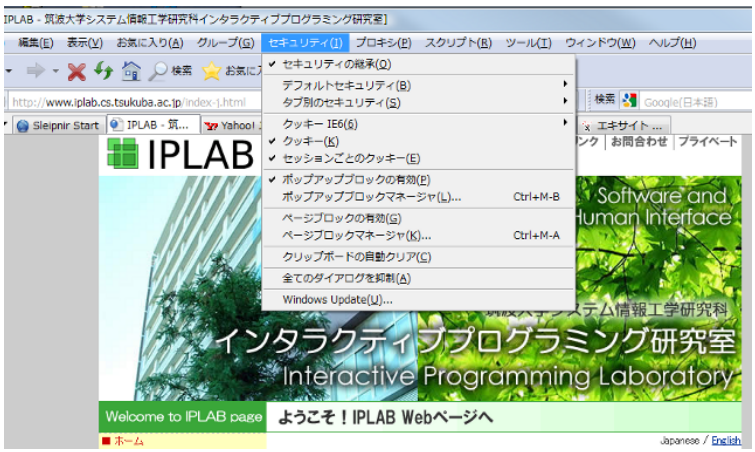


図 2.5: プルダウンメニュー

ポインティングハンドジェスチャにおいて，メニューの選択操作は，新たに別のハンドジェスチャを用いるか，ポインタを一定時間静止させるなどの特定のポインタの移動によって行う必要がある．しかし，新しいハンドジェスチャを用いた場合，ユーザは 2 種類以上のジェスチャを使い分けることが必要になり，利用に際して訓練も必要になってくる．2 種類以上のジェスチャを利用することで，システム側の誤認識やユーザの意図しない動作を招く恐れがあり，操作に支障をきたす恐れがある．他の問題として，既存のメニューインタフェースは一つ一つのメニューが小さく，並んで配置されている．そのため，ポインタの位置がブレやすいポインティングハンドジェスチャを用いて選択するには難しく，誤選択が起こる恐れがある．また，プルダウンメニューやポップアップメニューにはそれぞれいくつかの問題が存在する．

プルダウンメニュー

プルダウンメニューの場合，先に述べた問題に加え，図 2.5 のようにメニューの位置が固定である．大画面環境では，画面の大きさから作業領域とメニューとの距離が離れてしまう場合が多く，ポインタの移動量が増大し，作業領域とメニューとの間の行き来が多くなるなどの問題がある．ポインティングハンドジェスチャでは，ポインタの位置がユーザの視線の位置と考えられるため，ポインタの移動量の増大に伴い視線の移動も増大し，ユーザへの肉体

的な負担も増え，作業効率も悪くなる恐れがある．これらの問題から，プルダウンメニューのように表示される位置が固定されるメニューインタフェースは，大画面環境には適さないと考えられる．

図 2.6: ポップアップメニュー

ポップアップメニュー

図 2.6 のようなポップアップメニューは，メニューを呼び出すことで，現在作業中の領域の近くにメニューを表示することができる．そのため，プルダウンメニューと比べ，ポインタや視線の移動は少なくなる．しかし，ポインティングハンドジェスチャでは，ポインタで円を描くなどのメニューを呼び出すための動きが新たに必要となる．そのため，メニュー選択の度に呼び出し操作を行わなければならず，時間のロスが大きくなる．ポインティングハンドジェスチャでは，常に軌跡を描き続け，またきれいな円などを描くことが難しいため，利用者が意図しない呼び出しを行ってしまったり，反対に必要なときにすぐに呼び出せないなどの問題が発生する．

2.8.2 新しいメニューインタフェースの利用

既存の GUI メニューを利用しない手法として，パイメニュー [13, 21] などの特殊なメニューインタフェースの利用も考えられる [61, 42, 52]．例えば，CrossY[3] は，クロッシングを利用して一ストロークで複数の項目を決定する（例えば，線種や太さ，色などを一ストロークで

選択）メニューインタフェースを作成している．また，FlowerMenu[6] では，特定の軌跡を描くことで，その軌跡に対応した操作が実行されるメニューインタフェースを作成し，また軌跡のガイドを逐次出すことでどのような操作が可能であるかをユーザに視覚的に表示している．

ハンドジェスチャとこれらのメニューを組み合わせた手法も研究されており，例えば Sören らは，マーキングメニュー [32] をハンドジェスチャを組み合わせて家電の操作を行うインタフェースを作成してる [33]．また，前野らは，大画面環境におけるハンドジェスチャ向けのメニューインタフェースとして，数種類のジェスチャとメニューインタフェースの組み合わせを試し，大画面のためのメニューのデザインなどについての考察を行っている [58]．

これらのメニューインタフェースの多くはポップアップメニューのように特定の動作をしたときに呼び出され，ポインタの動きでメニュー選択が可能であるため，大画面環境では既存の GUI メニューを利用するよりも選択を行いやすい．しかし，ポインタの動きがぶれてしまった際の影響を受けやすく，それによる誤選択が起こりやすいという問題がある．ポインティングハンドジェスチャでは，ポインタのブレ以外に，ユーザの描いた軌跡と実際に描いた軌跡とでは差が大きいという問題があるため，描いた軌跡に対する誤認識が多いといった問題が発生するため，ペンベースインタフェースで有効な手法がポインティングハンドジェスチャでも有効であるとは限らない．

本研究でも以前に FlowMenu[23, 22] と指先の移動軌跡を利用したメニュー選択手法を実装した [62] が，手ぶれなどによる誤選択が多く，また操作自体がかなり特殊であったため，最初の内は訓練が必要であるなどの問題があった．また，ポップアップメニューの場合と同様に呼び出すためのジェスチャが必要になるため，ポップアップメニューと同様の問題も発生する．

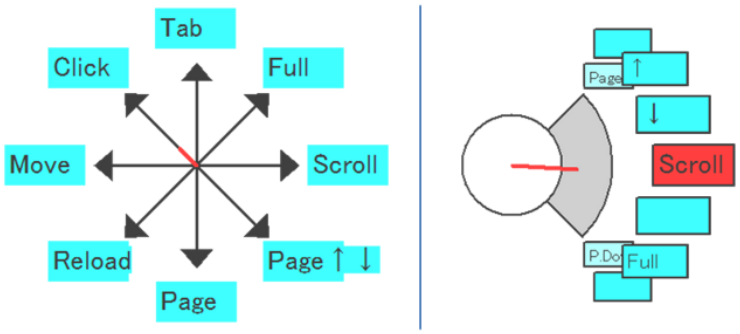


図 2.7: 移動方向と回転角度を利用したメニューインタフェース

本研究でも以前，FlowMenu を利用して，図 2.7 のような指先の移動方向や回転角度を利用したメニューインタフェースを作成したことがある [39]．このインタフェースでは，図 2.8 のように指先を動かすことで，その移動方向にあったメニューを選択したり，回転角度によってメニューも回転し，別のメニューを選択できるようになっていた．しかし，このようなメニューインタフェースでは，メニュー選択に必要な過程が特殊であったため，ユーザは利用するために練習する必要があり，また慣れないうちはゆっくりと手を動かして操作すること

が多かったため，操作効率があまりよくなかった．

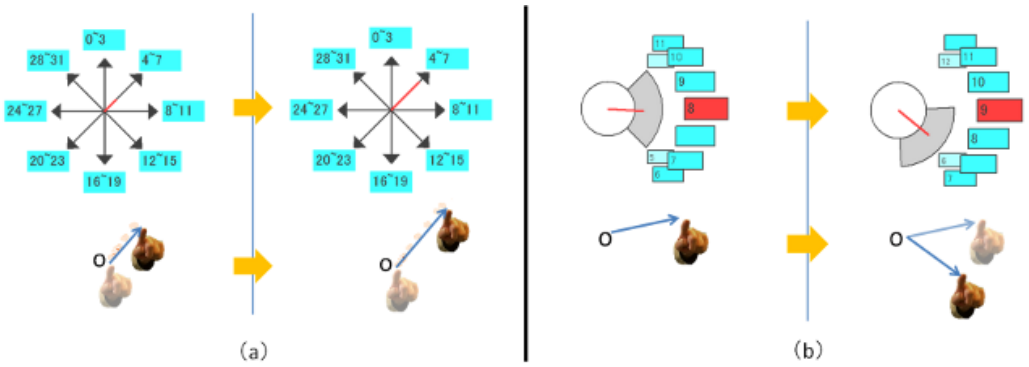


図 2.8: （a）指先の移動方向による選択，(b）指先の回転角度による選択

本研究では，これらの問題点を解決するために，ポインティングハンドジェスチャに合わせた新しいメニューインタフェースの枠組みを作成し，その枠組みに沿ったメニューインタフェースの作成を試みた．その新しいメニューインタフェースとして，本研究では新たにパレットインタフェースを提案し，そのための枠組みを作成した．また，この枠組みに即したメニューインタフェースとして C.Palette を作成した．パレットインタフェースを利用することで，必要なメニューに素早くアクセスすることができ，またポインタの移動量を軽減することができるため，ユーザへの負担も少なくなる．メニューの選択手法もダブルクロッシングを利用することで，選択したいメニューを素早くかつ簡単に行うことが可能である．パレットインタフェースと C.Palette については，第 4 章で説明する．

第3章　ダブルクロッシング

本研究では，クロッシング手法をベースとしたポインティングハンドジェスチャのための新しい選択手法としてダブルクロッシングを作成した．ダブルクロッシングは，1 回のクロッシングで生じる問題を解決し，かつよりポインティングハンドジェスチャに適した選択手法となっている．本章では，ダブルクロッシング手法の基本原理やその実装方法について説明するほか，既存手法のウェイティングなどと比較し，どの場面でどのような手法が適しているか考察を行った．また，実験を行った際に，各手法で選択時間などに影響を及ぼす要因についても考察した．

3.1　ダブルクロッシングの設計

通常のクロッシング手法（シングルクロッシング）では，図 2.4 の右のように一回だけ交差する．ダブルクロッシングでは，図 3.1 のように短時間に同じオブジェクトを 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）にボタンを押すよう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しない，長い時間作業を行っても疲れにくいという利点もある．

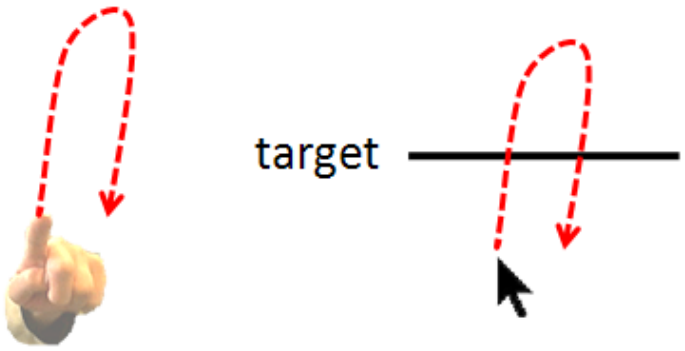


図 3.1:（左）ダブルクロッシングの際の手の動き（右）ポインタの動きとダブルクロッシング

3.2 ダブルクロッシングを用いたインタラクション

ダブルクロッシングとポインティングハンドジェスチャを利用したインタラクションは，基本的には選択可能なオブジェクトやメニュー項目に対してクロッシング判定用の領域を用意し，その領域に対してダブルクロッシングを行った際にそのオブジェクトを選択する．判定用の領域については，線であれば十分であり，またどこに設定しても基本的には問題はなく，既存の GUI をベースにして利用することができる．既存の GUI をベースにすることで，様々な既存のアプリケーションに対して，大きく手を加えなくてもポインティングハンドジェスチャで利用することができ，アプリケーションをポインティングハンドジェスチャ専用に新たに作る必要がない．

しかし，既存の GUI はオブジェクトによっては，小さかったり，オブジェクト同士が密集して存在することがあるため，ポインティングハンドジェスチャではこれらをそのまま利用した場合，誤選択などを起こす恐れがある．そのため，既存の GUI をただそのまま利用するのではなく，選択可能なオブジェクトを制限したり，選択しづらいオブジェクトに対して，何らかの補助をすることで，誤選択を減少させる工夫が必要である．本研究では，この問題に対して，既存の GUI を操作できるポインティングハンドジェスチャ向けのメニューインタフェースを新たに作成することで解決を試みている．このインタフェースについては，第 4 章で述べる．

3.3 実装

本研究で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 3.2 のようになっている．ユーザは，指先に図 3.3 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 Logitec 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 × 480 ピクセルで，フレームレートは 30fps 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，操作の実行およ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 2.83GHz，RAM:4.0GB，HDD:500GB，OS:Windows Vista であった．

3.3.1 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閾値処理によって行っている．なお，予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．

$$
\begin{aligned}
px &= \frac{cx}{camWidth} screenWidth + screenLeft \\
py &= \frac{cy}{camHeight} screenHeight + screenTop
\end{aligned}
$$

$camWidth$ 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．

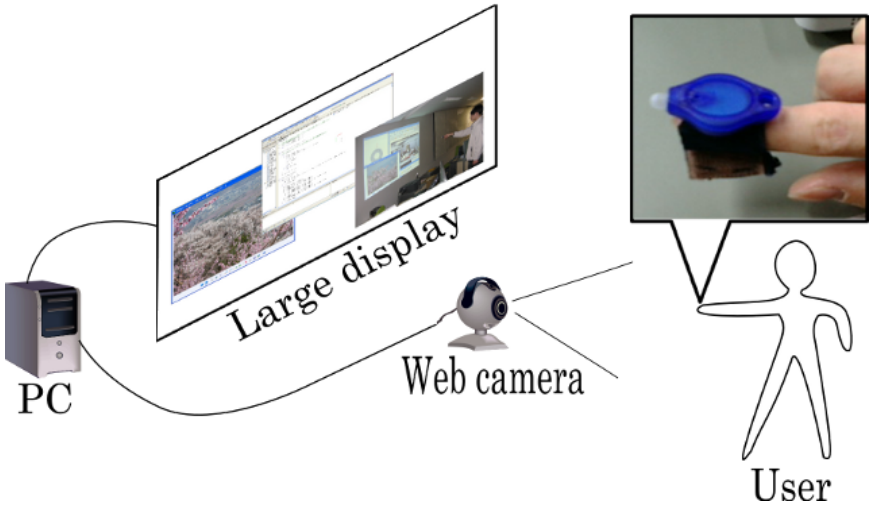


図 3.2: 大画面ポインティングシステムの構成

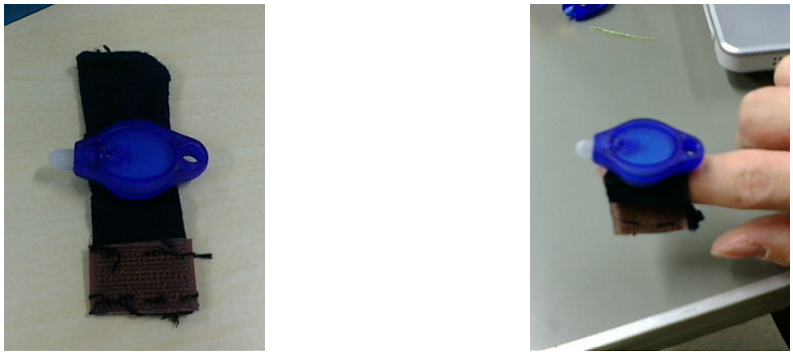


図 3.3: LED デバイス（左）と装着した様子（右）

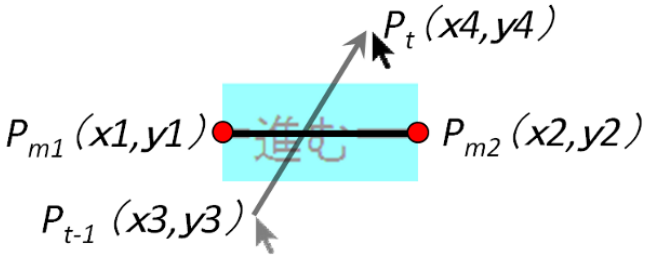


図 3.4: クロッシングの認識

3.3.2 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，2 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 3.4 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．

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

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．

3.4 ポインティングハンドジェスチャにおける選択手法の評価

Accot らの実験 [1] では，ペンストロークによるクロッシングとタッピングによるターゲット選択の比較を行っている．この実験では，クロッシングはタッピングとほぼ同じ速度で選択することができ，また選択操作の難しさについてもタッピングとほぼ同じであることが示されている．この結果から，ペンインタフェースにおいては，クロッシングはタッピングなどの代替手段として利用することができることが示されている．

本実験では，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

図 3.5: 実験におけるブラウザとメニューの配置

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などを確かめるための評価実験を行い，既存手法の代替手法として利用できるか考察を行った．

3.4.1 実験内容

本実験では，図 3.5 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）と右上にある矩形のメニュー（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

3.4.2 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，実際の研究室のホームページ[1] を解析し，ページ上にあるリンクをターゲットとした．そのため，リンクターゲットは，サイズや配置に大きく差がある．小さ

[1]http://www.iplab.cs.tsukuba.ac.jp/index-j.html

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，縦もしくは横に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．ウェブ上のリンクの位置の解析については，VC++と IAccessible インタフェースを用いて解析した．

メニューターゲットは，図 3.5 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，リンクターゲットとは異なり，大きさは統一し，整列して配置した．このターゲットの大きさは，全て 100 × 50 ピクセルとなっており，メニューターゲット同士は，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 3.5 のように全部で 10 個用意した．メニューターゲットは Windows のデスクトップ上に透明なウィンドウは最前面に配置し，その透明ウィンドウにメニューターゲットを半透明にして表示した．なお，図 3.5 のメニューは不透明であるが，実際には，Windows のデスクトップ上に半透明で表示されていた．

3.4.3 実験環境

実験は，横 2.6m ×縦 1.5m の大画面（解像度 1920 × 1080 ピクセル）で，被験者はこの画面から 3m 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 1m 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 3.5 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．被験者が座った状態で軽く腕を上げたときの位置を中心とした 100cm × 50cm の矩形領域内で指先を動かすことでポインタは大画面全域を移動することができた．ポインタの移動精度についても，実験で用いた縦 15 ピクセル程度のターゲットでも問題なくポインタを移動させることができるぐらいの移動精度があった．

3.4.4 計測方法

本実験では,「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」の 3 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，ウェイティ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．

3.4.5 実験手順

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取ることができた．

表 3.1: 各静止時間の選択時間と誤選択率

	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 %

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，被験者毎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．

3.4.6 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,「ウェイティング」と「シングルクロッシング」の 2 つの手法を比較のために用いたが，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．本実験では，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．

ウェイティング

ウェイティングの静止時間に関する結果は，表 3.1 の通りである．表 3.1 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

表 3.2: 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

	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 %

昇してしまう．本実験では，この結果を踏まえ，選択時間が早く，且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．

ダブルクロッシング

クロッシング間隔の上限に関する結果は，表 3.2 の通りである．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．本実験では，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．

3.4.7 実験結果

各手法ごとの選択時間と誤選択率の結果は図 3.6 と図 3.7 の通りとなった．また，手法毎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ティ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ティングとシングルクロッシング，シングルクロッシン

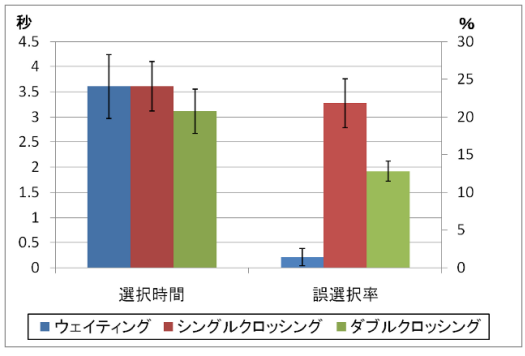


図 3.6: リンクターゲットの選択時間と誤選択率

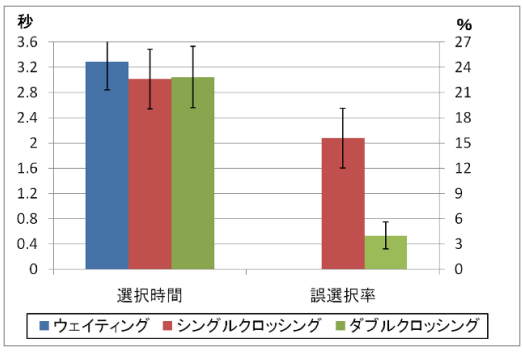


図 3.7: メニューターゲットの選択時間と誤選択率

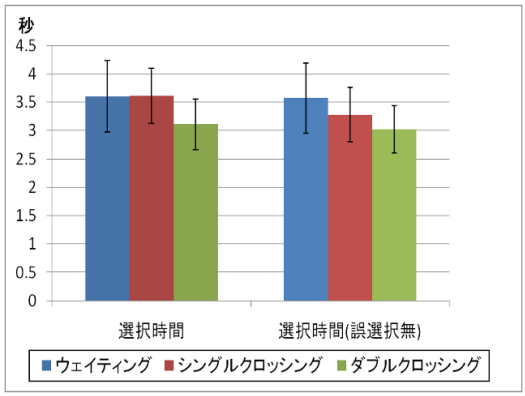


図 3.8: 誤選択の有無による選択時間の違い（リンクターゲット）

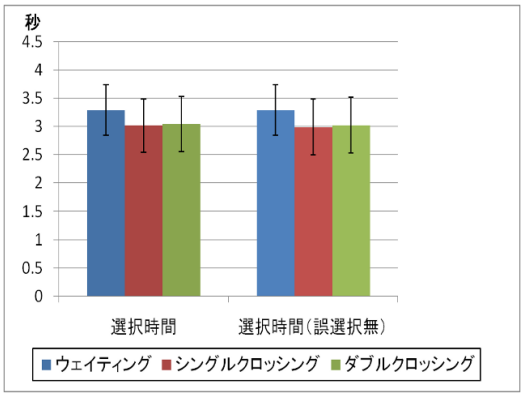


図 3.9: 誤選択の有無による選択時間の違い（メニューターゲット）

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，3 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，ウェイティ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，ウェイティ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，選択時間もウェイティングとほとんど同じであったが，誤選択が無かった場合の選択時間はウェイティングよりも約 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，ウェイティ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) 毎にマッピングを行った．図 3.10 と図 3.11 は，各ターゲットについて手法毎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，ウェイティングで $t = 0.3167 \times ID + 2.3858 (R^2 = 0.145)$，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，ウェイティングで $t = 0.699 \times ID + 1.1676 (R^2 = 0.2059)$，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．

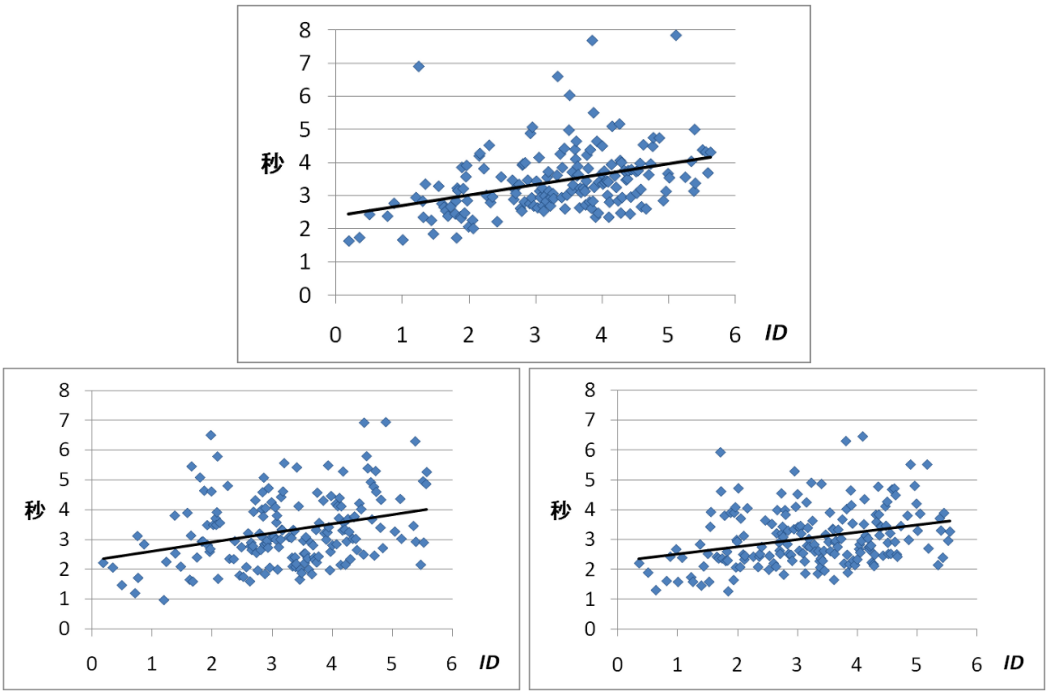


図 3.10: リンクターゲットにおける ID と選択時間の分布（上：ウェイティング，左下：シングルクロッシング，右下：ダブルクロッシング）

3.4.8 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，空中に手を留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かったとコメントした．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 2 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ことができるが，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かったというコメントが多く得られた．

3.5 考察

本実験では，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．

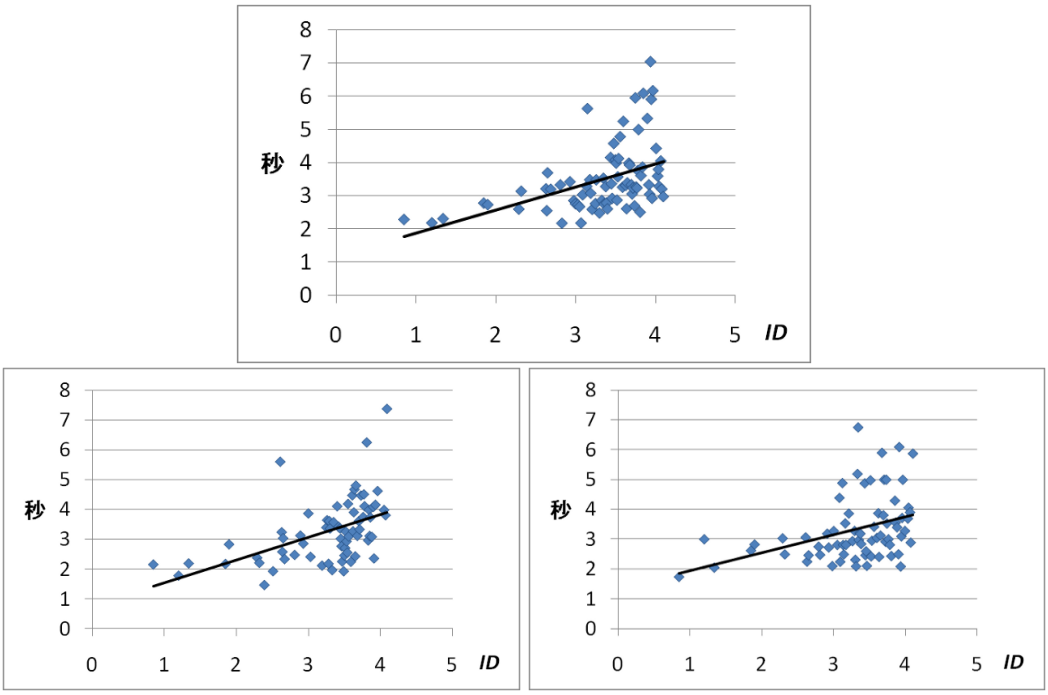


図 3.11: メニューターゲットにおける ID と選択時間の分布（上：ウェイティング，左下：シングルクロッシング，右下：ダブルクロッシング）

3.5.1 リンクターゲット

ウェイティングは，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おこらず，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 2 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ブレ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度々見られ，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ティングよりも早く，誤選択率もウェイティ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．

3.5.2 メニューターゲット

ウェイティングは，他の 2 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，指先の認識時のブレや手ぶれによるポインタがあまり大きくブレなかったため，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なかったため，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ブレ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の関する誤選択の内 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ブレ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．

3.5.3 選択難易度

図 3.10 と図 3.11 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 3.10 と図 3.11 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 3.10 と図 3.11 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．

3.5.4 選択時間に影響を及ぼすパラメータについて

前節で，ID による分布から考察を行ったが，各手法において，どのようなパラメータが影響を及ぼすかについては判断が難しい．そこで，各手法について，パラメータ毎の分布やいくつかの仮定などから，どのようなパラメータが選択時間に影響を及ぼすのかを考察した．

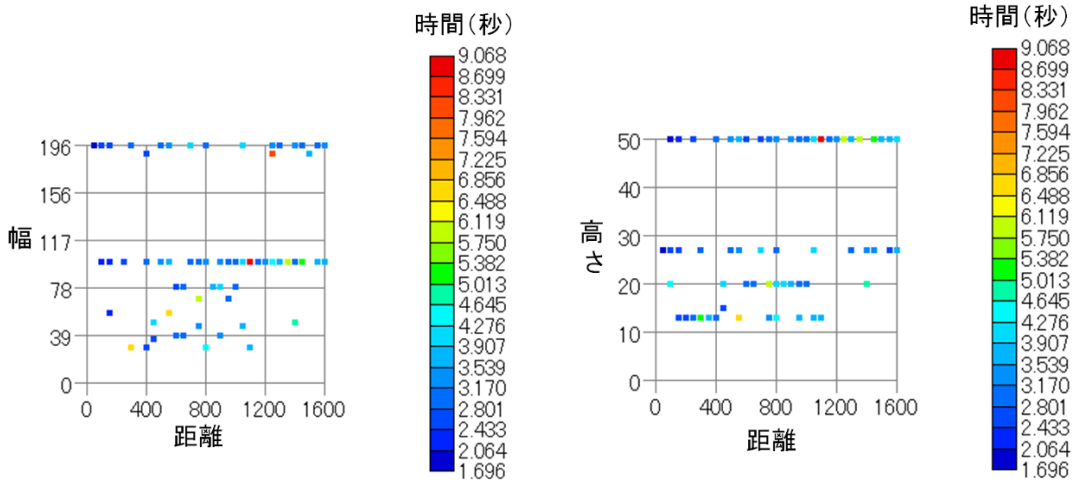


図 3.12: ウェイティングに関する選択時間の分布（左：ターゲットの幅と距離，右：ターゲットの高さと距離）

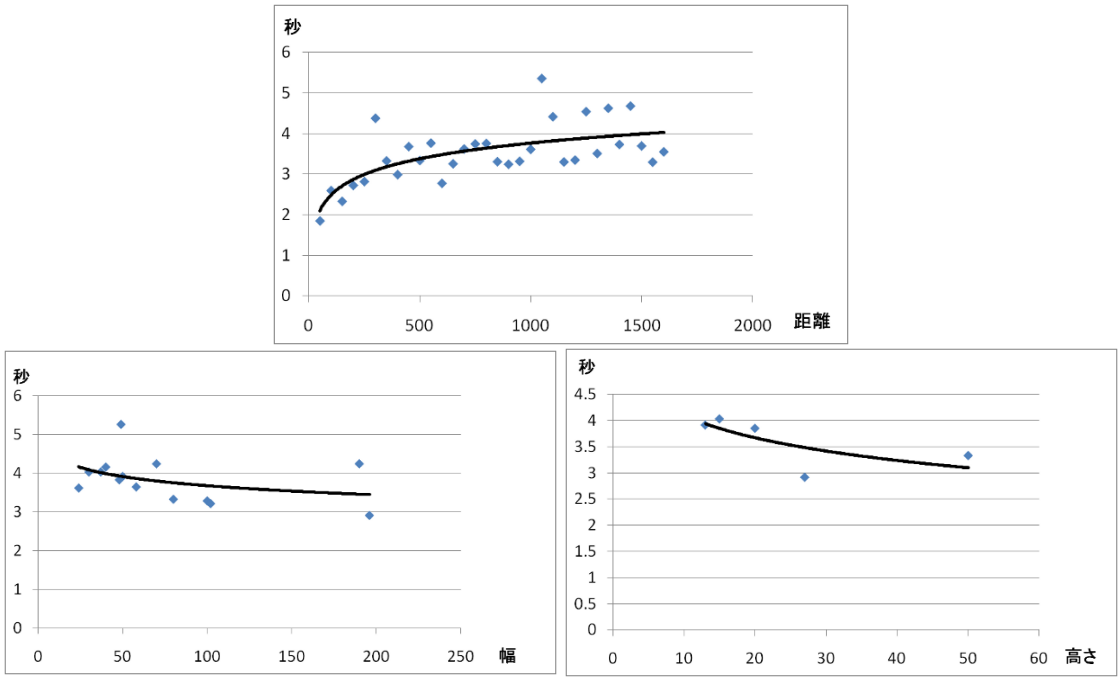


図 3.13: パラメータ毎のウェイティングの選択時間の分布（上：距離ごとの分布，左：ターゲットの幅による分布，右：ターゲットの高さによる分布）

ウェイティング

ウェイティングでは，ポインタの移動以外に手ブレなどによるポインタのブレの影響から，ターゲットまでの直線距離（距離）以外にターゲットの幅か高さのどちらかが影響を及ぼしていると考えた．図 3.12 は，ターゲットの幅（高さ）と距離ごとにまとめ，プロットした図である．距離は，前のターゲットの中心から次のターゲットの中心までの直線距離とした．この距離は 1 ピクセル単位で非常に細かい値が多く，等距離である組み合わせは少数であったため，50 ピクセル単位でまとめている．また 1 つのデータしかなかった組み合わせについては，信頼が置けないデータと仮定して除外している．図 3.12 の各点の色がその組み合わせにおける平均選択時間を示しており，青い点は時間がかからなかったことを示しており，赤い点ほどより時間がかかっていることを示している．ウェイティングの場合，どちらの組み合わせでも，距離が長い場合やや時間がかかっていた傾向が見られたものの選択時間に大きな違いが見られかった．

ウェイティングについて，距離・幅・高さの各パラメータ毎の傾向を詳細につかむために，一つ一つのパラメータに対して選択時間の分布図を得た．図 3.13 は，その結果である．図 3.13 より，距離の増加に応じて選択時間は増加する傾向があり，ターゲットの幅では増加や減少の傾向は見られずにバラバラであり，ターゲットの高さではやや減少する傾向が見られた．それぞれの結果に対して，回帰分析を行ったところ，ターゲットのまでの距離については，$T = 0.5575log(D) - 0.0894(R^2 = 0.4338)$ （T は選択時間，D はターゲットまでの距離），ターゲットの幅に関しては $T = -0.453log(W) + 5.253(R^2 = 0.1371)$ （W はターゲットの幅），ターゲットの高さに関しては $T = -0.625log(H) + 5.542(R^2 = 0.5026)$（$H$ はターゲットの高さ）という結果が得られた．

パラメータごとの分布図や回帰曲線から，ウェイティングの選択時間はターゲットまでの距離やターゲットの高さに影響を受けやすいといえる．また，本実験では，全てのターゲットについて（ターゲットの幅）＞（ターゲットの高さ）であったため，ターゲットの幅や高さの内，小さい値の方に影響を受けやすいと考えられる．しかし，この 2 つのパラメータについては，相関がやや弱い．その理由として考えられることとして，手ブレなどによるポインタのブレが影響している可能性がある．ポインタが一定値以上ブレてしまった場合，ウェイティングの静止時間はリセットされてしまい，やり直す必要がある．このやり直しは，時々起こってしまうため，距離や大きさが同じターゲットでも選択時間の差が大きく出てしまい，その影響により各パラメータと選択時間との相関があまり高くなかったのではないかと考えられる．

シングルクロッシング

シングルクロッシングでも，ウェイティングと同様にターゲットまでの距離が選択時間に影響を及ぼしていると考えた．ターゲットの幅については，一定の長さを超えると選択のしやすさはあまり差がないのではないかと考えられ，距離よりも影響を与えないのではないかと考えた．なお，クロッシングでは，ウェイティング異なり交差するための線の長さが重要

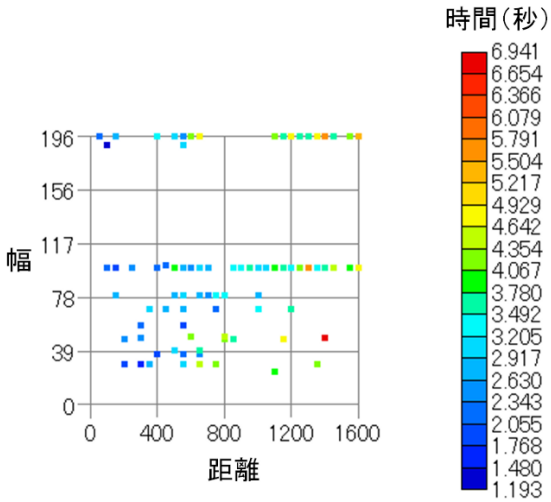


図 3.14: シングルクロッシングに関する選択時間の分布

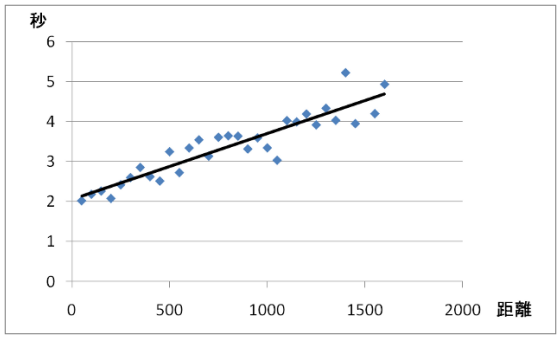


図 3.15: シングルクロッシングの距離による選択時間の分布

であるため，ターゲット高さは検討する必要はない（本実験では，線の長さは全てターゲットの幅であったため）．そこで，ターゲットまでの距離と幅の各組み合わせで平均選択時間を調べた．その結果をプロットした図が図 3.14 である．距離やプロットの方法は図 3.12 の時と同様である．

シングルクロッシングの場合，ウェイティングの場合と異なり，距離が長くなるにつれ，選択時間がより長くなっていく傾向が見受けられ，距離が選択時間に影響を及ぼしている可能性が見受けられる．幅については，幅が大きくなっても選択時間が短くなるような傾向は図 3.14 からは見受けられず，あまり影響を及ぼしていない可能性が高い．

シングルクロッシングは，距離に応じて選択時間が長くなる可能性があるため，距離ごとに平均選択時間をとり，プロットした．図 3.15 は，その結果である．図 3.15 より，距離が長くなるにつれ，選択時間が増加していることが見て取れる．また，この結果に対して回帰分析を行ったところ，$T = 0.0016 * D + 2.0523 (R^2 = 0.8488)$ という回帰直線が得られた．(T は選択時間，D はターゲットまでの距離である）．このように距離に対して高い相関が得られたため，シングルクロッシングにおいては，ターゲットまでの距離が選択時間に強い影響を及ぼしているといえる．

ダブルクロッシング

ダブルクロッシングもシングルクロッシングと同様にポインタの移動距離が選択時間に影響を及ぼしていると仮定した．ターゲットの幅については，ダブルクロッシングのしやすさの問題から，ある程度の長さまでは選択時間に影響を及ぼすが，ある長さを超えた場合，影響がほとんどなくなると仮定した．そこで，ダブルクロッシングについても，ターゲットま

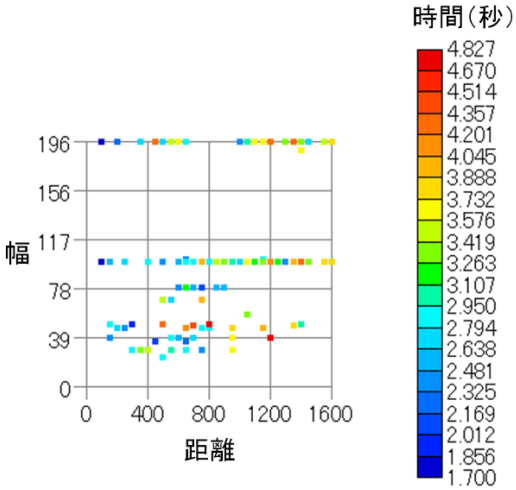


図 3.16: ダブルクロッシングに関する選択時間の分布

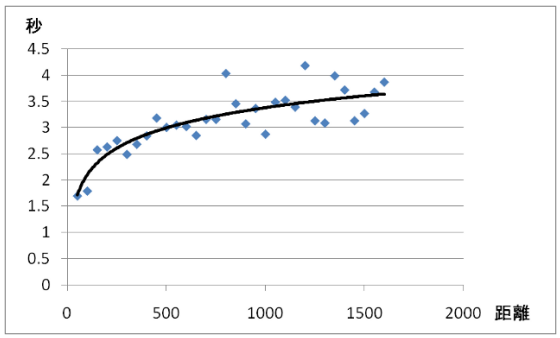


図 3.17: ダブルクロッシングの距離による選択時間の分布

での距離と幅の各組み合わせで平均選択時間を調べた．その結果をプロットした図が図 3.16 である．距離やプロットの方法は図 3.12 の時と同様である．

図 3.16 より，シングルクロッシングのときと同様に距離の増加に応じて，選択時間が増加する傾向が見受けられた．ダブルクロッシングの場合，シングルクロッシングよりも増加量については緩やかであり，またターゲットの幅によってばらつきが見られた．この分布図より，ダブルクロッシングについても，距離が選択時間に影響を及ぼしていると考えられる．ターゲットの幅による選択時間の分布については，ばらつきがあるだけで，分布の傾向が見られたわけではないため，ターゲットの幅はあまり影響を与えないのではないかと思われる．

ダブルクロッシングについても，距離毎に選択時間をプロットした．プロットした図は図 3.17 である．図 3.17 より，距離の増加に応じて選択時間は増加しているが，増加傾向はシングルクロッシングのときよりも緩やかであることがわかる．この結果に対して，回帰分析を行ったところ，$T = 0.5565log(D) - 0.466(R^2 = 0.7146)$ であった（T は選択時間，D はターゲットまでの距離である）．ターゲットの幅で同様に分類して回帰分析したところ，$T = -0.15135log(D) + 0,367(R^2 = 0.0444)$ となり，相関は高くなかった．このように，ターゲットまでの距離に対してやや高い相関が見られたため，シングルクロッシング動揺にこの距離が選択時間に影響を及ぼしているといえる．回帰曲線が対数であったため，シングルクロッシングとは異なり，その影響はやや緩やかであるといえる．また，ターゲットの幅については，選択時間に影響を及ぼさないため，2・30 ピクセル程度の小さな幅であっても選択を行うには十分であるといえる．

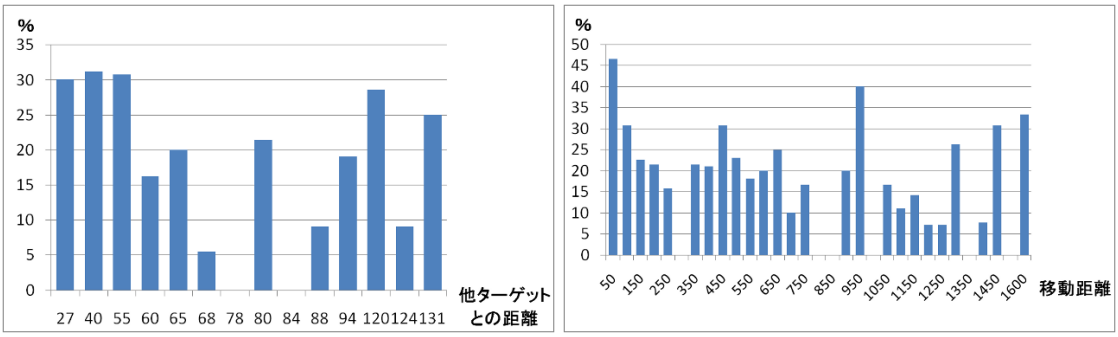


図 3.18: シングルクロッシングに関する誤選択率（左：最も近い他のターゲットまでの距離と誤選択率，右：指定されたターゲットまでの距離と誤選択率）

3.6　誤選択率に影響を及ぼすパラメータについて

誤選択率についても，選択時間と同様にどのようなパラメータが影響を及ぼすかそれぞれの手法について考察を行った．

3.6.1　ウェイティング

ウェイティングについては，他の 2 つの手法と異なり，実験全体を通じても誤選択は 7・8 回程度しか起こらなかった．これは被験者が平均で 1 回弱間違えた程度である．そのため，この結果から移動距離やターゲットの大きさとは無関係であり，ウェイティングに関しては特にパラメータによる影響はないと考えられる．

3.6.2　シングルクロッシング

シングルクロッシングは，ポインタを移動させて選択させるため，近くに他のターゲットが存在すると誤ってそのターゲットを選択してしまう恐れがある．また，ポインタの移動経路上に他のターゲットが存在する場合もその場所を迂回する必要があるため，気がつかない場合には誤選択が起こる恐れがある．そこで，最も近い他のターゲットまでの距離と指定されたターゲットまでの距離の 2 つのパラメータで誤選択率の分布を調べた．

図 3.18 はその分布図である．最も近い他のターゲットの距離は指定されたターゲットの中心から他のターゲットの中心までの距離のうち最も短い距離である．また，指定されたターゲットまでの距離は，前のターゲットの中心から次のターゲットの中心までの直線距離であり，50 ピクセル単位でまとめている．また，誤選択率は（各距離における誤選択回数）/（各距離における全選択回数）× 100（%）とした．

図 3.18 より，他のターゲットまでの距離が 60 ピクセル以下の場合はいずれも 3 割以上と高いが，他のターゲットまでの距離が長くなるほど誤選択率があまり減少していないことが分かる．そのため，他のターゲットまでの距離が短い場合は誤選択が起こりやすいと言えるが，

誤選択を防ぐためには，ターゲット同士をかなりの距離離す必要があると考えられる．また，指定されたターゲットまでの距離の場合も同様に，誤選択率が低かった距離もあるが，距離が長くても誤選択率に大きな変化は見られない．そのため，移動距離については，誤選択率にあまり関係ないと考えられる．

3.6.3　ダブルクロッシング

ダブルクロッシングの場合，シングルクロッシングと同様にポインタを移動させて選択手法であるため，誤選択は他のターゲットまでの距離が短いほど発生しやすいと考えられる．その一方で，シングルクロッシングの場合と異なり，ある程度間が空いていれば誤選択はあまり起こらなくなると考えられる．また，シングルクロッシングのように他のターゲットを迂回する必要が無いため，ポインタの移動距離は関係ないと考えた．

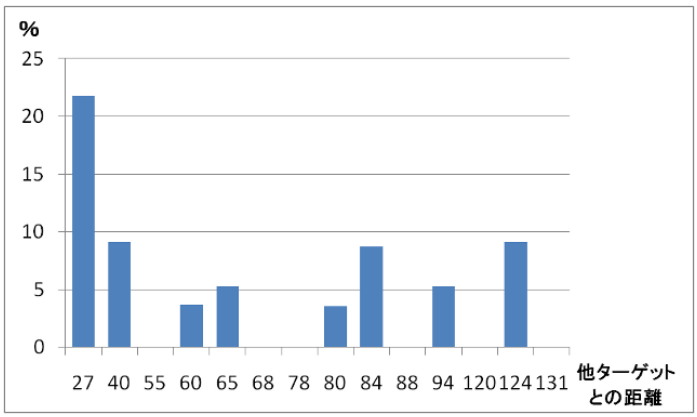


図 3.19: ダブルクロッシングの他のターゲットまでの距離毎の誤選択率の分布

図 3.19 は，最も近い他のターゲットまでの距離毎の誤選択率の分布である．図 3.18 の場合と同様に指定されたターゲットと他のターゲットとの距離のうち最も近い距離でまとめ，誤選択率の計算方法も同様に行っている．最も近い距離である 27 ピクセルのみ，誤選択率が 20 %を超えているが，他の距離では，10 %以下という結果であった．また，本実験のダブルクロッシングの誤選択の 6 割近くがこの距離で起きていた．40，84，124 ピクセルで 8〜9 %起きているが，実際には誤選択は 1〜2 回しか起こっていなかった．そのため，ダブルクロッシングでは，誤選択率は他のターゲットまでの距離の影響を受けるが，ある程度離れていれば，誤選択は著しく減少すると考えられる．どこまで離れていれば十分であるかは，本実験のみでは断言することは難しいが，図 3.19 の結果より 50 ピクセル程度離れていれば十分であると考えられる．

3.6.4 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．ウェイティングについては，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，選択時間は早いが，誤選択が多いため，タッチパネルやペンベー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．

第4章　パレットインタフェース

ポインティングハンドジェスチャとクロッシングを利用して実際にアプリケーションの操作を行う方法のひとつに，既存の GUI をそのまま選択して操作する方法がある．この方法では，例えばウェブを操作する場合，ウェブのリンクやスクロールバー,「戻る」などのボタン全てにクロッシングの判定線を設置することで操作を行う．しかし，この方法では，リンクなどで判定線が密集する場所がでたり，判定線が短くなる場所ができたりする．そのため，クロッシングが難しい場所がでたり，意図しないクロッシングが行われたりしてしまう．

本研究はクロッシングのための GUI 部品を集めた専用のメニューインタフェースを用意し，そのインタフェースを通じて操作を行うことにした．また，操作を行う際，ユーザはポインタの周囲に常に視線があると考え，ポインタの周囲にメニューを表示することでユーザの負担を減少できるのではないかと考え，パレットインタフェースを新たに作成した．

パレットインタフェースは，既存の GUI インタフェースに重ね合わせて利用することを想定している．そのため，専用のアプリケーションを個別に作成する必要がなく，既存のアプリケーションをそのまま利用することができる．また，大画面環境で利用するにあたり，次節のような設計指針をたて，この指針に従ってインタフェースの作成を行った．

4.1　設計指針

大画面環境では，ポインタの位置がユーザの視線の位置とほぼ同位置であると考えられる．ポインタの位置がユーザの視線の位置であるため，ポインタの移動が増大すれば，それに伴い視線の移動も増大してしまうと考えられる．本研究は，ユーザの視線移動を減少させるために，メニューをユーザの見える場所に表示するべきであると考えた．

ポインティングハンドジェスチャでは，マウスのクリックに相当する操作が存在しないため，ポップアップメニューを必要なときに呼び出す場合，呼び出しのための専用のハンドジェスチャが必要になる．これはマウスなどを利用する場合と異なり，ユーザに煩わしさを与えかねない．そこで，パレットインタフェースでは，より早く効率よく操作が行えるようにするため，メニューを常にポインタの周囲に表示し，不必要な場合にのみメニューを隠すようにする．

メニューの表示については，離れた場所からの操作のため，メニューは遠くからでも分かるように大きく，かつ密集させずに表示させることによってメニューを見やすくする．その反面，大きく表示する必要があるため，一度に沢山のメニューを表示してしまうと作業中のアプリケーションの画面が見づらくなり作業の妨げになる恐れがある．そのため，操作に必

要なメニューのみを表示し，状況に応じたメニューの切り替えや，メニューを自由に作成するための枠組みも必要になる．

ポインティングハンドジェスチャでは，マウスのボタンやキーボードに相当する機器を利用しない．そのため，パレットインタフェースにおけるメニューは，コピーや貼り付けなど一般的に言われているメニュー操作に加え，画面のスクロールやクリックなどマウスのホイールやキーボードを利用して行う操作についても扱う．

4.2 パレットインタフェースの基本機能

パレットインタフェースは，先に述べた設計指針を基に，大画面環境で利用しやすいインタフェースにするために,「パレットインタフェースの移動」,「メニューの切り替え」,「表示/非表示の切り替え」という 3 つの基本機能が必要になると考えた．

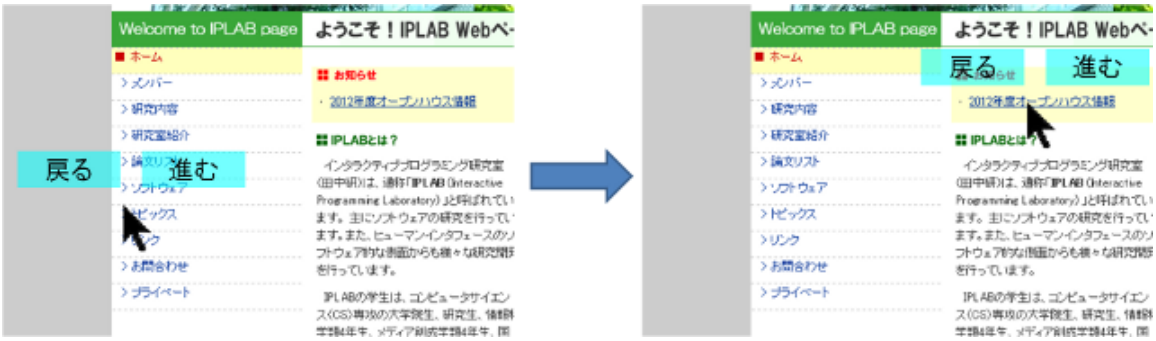


図 4.1: パレットインタフェースにおけるメニューの移動イメージ

4.2.1 パレットインタフェースの移動

「パレットインタフェースの移動」は，メニューをユーザの視界の近くに常に表示させるためにポインタの周囲に常にメニューを表示させるための機能である．メニューの位置が固定である場合，通常の画面での利用と比較して，大画面では作業領域とメニューとの間での視線の移動量が増加し，疲れやすい．また，WIMP インタフェースにおいて，ポインタの位置はユーザの視線の位置とほぼ同位置であると考えられる．そのため，ポインタの移動に追従してパレットインタフェースも自動的に移動させることにより，ユーザの視界に常にメニューインタフェースを表示することができる．

例えば，図 4.1 の左図のようにポインタの周囲に「戻る」や「進む」のメニューが表示され，ポインタが図 4.1 の右図のようにポインタが右上に移動した時，そのポインタの動きに合わせて 2 つのメニューも同時に移動するといった方法が考えられる．このような移動をさせるだけでもポインタとメニュー間の視線の移動を減らすことができ，ユーザへの負担が軽減されることが期待できる．

4.2.2　メニューの切り替え

一度にたくさんのメニューを表示することは難しいため，複数のアプリケーションの操作やより多くのメニューを行うには状況に応じて表示するメニューを切り替える必要がある．メニューの切り替え方法として，例えば，コンテキストメニューのように現在の状況をシステムが自動で認識して，表示するメニューの内容を切り替えるなどの方法がある．

しかし，ひとつのアプリケーションのためのメニューの数が増えれば，メニューの見やすさや作業のしやすさの問題から，一度に全てを表示することは難しく，同じアプリケーション内でもメニューを切り替えを行う必要があるため，自動で切り替える以外にユーザの任意のタイミングで手動で切り替えることができるようにする必要もある．ユーザが手動で切り替えられるようにするためには，メニュー表示切り替え用の専用メニューを個別に用意する必要がある．

4.2.3　表示/非表示の切り替え

パレットインタフェースは常に既存の GUI 上に表示されているため，操作状況によってはユーザの作業の妨げになってしまうかもしれない．そのため，必要に応じてウィンドウの透明度を上げたり，最小化するなどしてユーザの使用の妨げを減少させる機能が必要である．

表示の切り替え方法としては，ユーザの任意のタイミングで行う方法とユーザの作業状況をシステム側が判断して自動で切り替える方法の 2 種類が考えられる．ユーザの任意のタイミングで行う場合，通常のメニューとは別に最小化用の専用メニューを用意し，そのメニューを選択することで実現する．自動で切り替える場合には，ポインタの動きなどを調べ，その活動状況に応じて，メニューを表示するか，非表示にするかを判断するなどの手法が考えられる．

4.3　表示するメニューの管理

ひとつのアプリケーションでも表示するメニューを切り替える必要があるため，表示するメニューについても全て自動作成したり，管理することは難しい．また，近くに大量のメニューを一度に表示することにより，ユーザが作業しているウィンドウの内容が見づらくなったり，大量に表示されたメニューから必要なメニューを探し出すことによる煩わしさによって作業を妨害しかねない．その上，作業の内容によって必要なメニューが異なったり，ユーザによってはよく利用するメニューが異なったりすることも考えられる．

パレットインタフェースでは，これらの問題を考慮するために，一度に表示するメニューの内容をユーザが自由に決められるべきであると考えた．ユーザが自由にメニュー項目を作成できるようにすることで，様々な状況に柔軟に対応することができる．表示するメニューを切り替える必要があるため，メニューを作成する際は，一度に表示するメニュー毎に作成する必要があり，それらを作成管理するための枠組みも必要になる．

4.4 C.Palette

本研究では，前節で述べた要件を基にパレットインタフェースのプロトタイプである C.Palette（Customizable Palette Interface）を作成した．表示するメニューは XML を利用することにより，利用者の嗜好や利用場所などに合わせてメニューの配置などを自由に変更することができる．C.Palette の概観は図 4.2 の通りであり，既存の GUI 上にオーバーレイ表示されている．

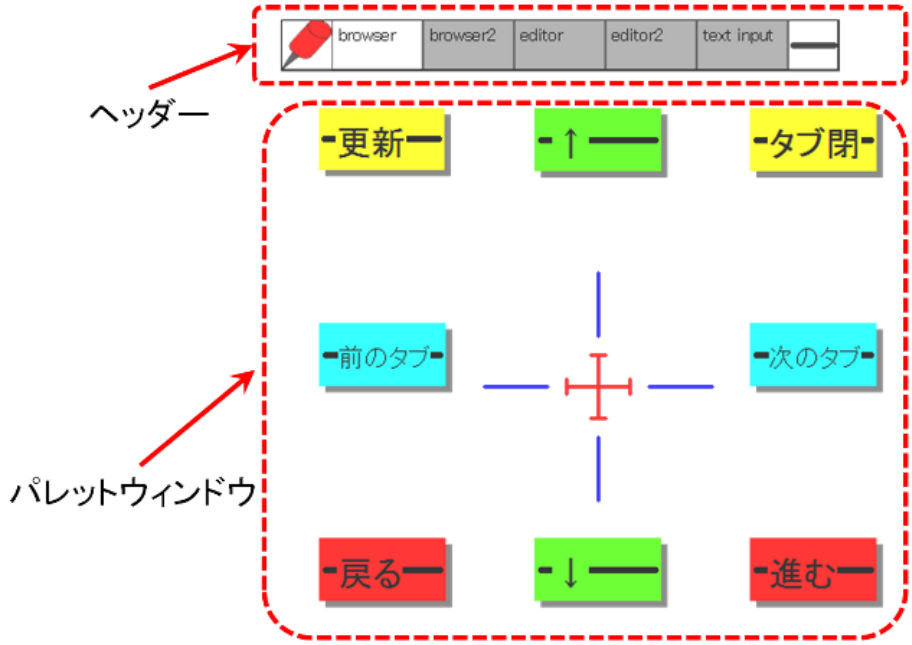


図 4.2: C.Palette の概観

このインタフェースは基本的には常にポインタの周囲に表示され，いくつかのメニューラベルが表示されている．

C.Palette は図 4.2 のように「ヘッダー」と「パレットウィンドウ」と呼ばれる 2 つの部品から構成される．

4.4.1 ヘッダー

ヘッダーは，中央部分の「ウィンドウタブ」，一番左にある「ピンアイコン」，一番右の「最小化アイコン」から構成され，パレットウィンドウの切り替えやパレットインタフェースの固定など主にパレットインタフェースに関する操作を行うことができる．ヘッダーはパレットインタフェースを起動した際に自動的に生成される．

図 4.3: メニューラベル

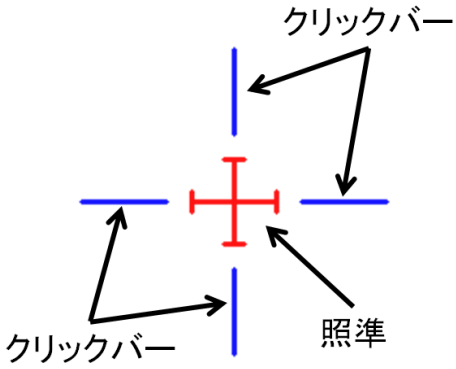


図 4.4: クリックラベル

4.4.2 パレットウィンドウ

パレットウィンドウは図 4.3 の「メニューラベル」と図 4.4 の「クリックラベル」から構成され，ユーザはこれらのラベルを選択することでインタラクションを行うことができる．パレットウィンドウは，先に挙げた問題からあまり大きくすることができず，一度に全てのメニューを表示することが難しい．そのため，C.Palette では複数のパレットウィンドウから構成される．各パレットウィンドウには，例えば異なったアプリケーションのメニューを配置したり，一つのアプリケーションに対するメニューを複数のパレットウィンドウに用意したりすることができる．パレットウィンドウは，ヘッダー中のウィンドウタブで管理されており，ひとつのウィンドウタブがひとつのパレットウィンドウを示している．

4.4.3 メニューラベル

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 4.3 のようなメニューラベルを用意し，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 4.3 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てられた操作を実行することができる (図 4.3 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)．一度ダブルクロッシングした直後に次の動作に容易に移ることができるため，同じ操作を何度も繰り返して行うことも容易にできる．

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 4.3 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に進むといったページを一つ進ませるような操作が割り当てられている．割り当てる操作は次に進むなどの一つのメニュー操作だけではなく，印刷の両面印刷の設定 (「プリンタのプロパティを開く」→「両面印刷を設定」→「プリンタのプロパティを閉じる」) など複数のメニュー操作を割り当て，一回のメニュー

ラベルの選択で行うことも可能である．また，指先を何回も上下させて何回もダブルクロッシングを行った場合は，行った回数だけ同じ操作が実行される．

4.4.4　クリックラベル

本研究では，選択する位置を合わせる動作と実際に選択を実行する動作を分けることで利用者が意図した場所に正確に選択の動作を行うことができるようにした．そのために，図 4.4 のようなクリックラベルを作成した．クリックラベルはマウスの左クリックに相当する操作を模倣し，クリックバーをダブルクロッシングしたときに赤い十字の中央でクリックの操作が実行される．

クリックラベルは，選択位置の調整と選択の実行の 2 種類の役割を明確に分けるために，2 つの部品から構成される．一つは「照準」とよばれる中心にある赤い十字であり，もう一つは照準の周囲にある 4 本の青い線分の「クリックバー」である．照準で選択位置を調整し，クリックバーは選択の実行を行う．照準の周囲にクリックバーを用意することによって，照準を合わせてすぐにクリックの動作が行えるようにした．また，メニューラベルにクリックの操作を割り当てることも可能であり，メニューラベルを選択して実行した場合も照準の中心部で選択が実行される．

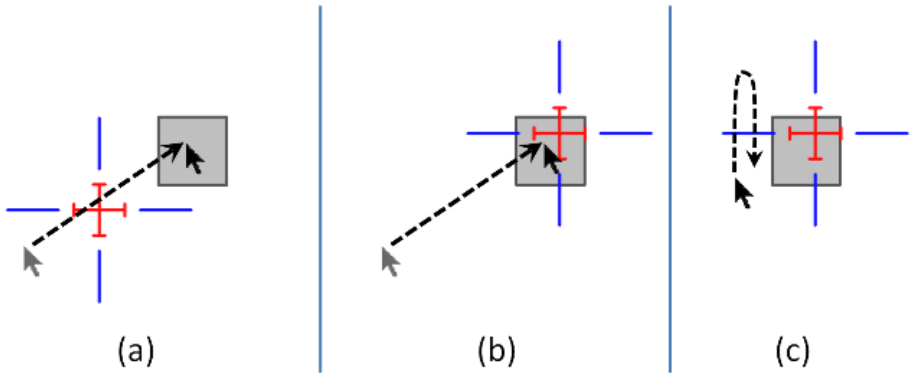


図 4.5: クリックラベルの移動と選択動作

照準はメニューラベルやクリックバーとは異なり，図 4.5(a) のようにポインタが横切ろうとした場合，その横切ろうとした動きに合わせてクリックラベルもポインタと一緒に移動する（図 4.5(b) の状態になる）．そのため，ユーザは照準を横切ろうとする動作を利用することで，選択位置の調整を行うことができる．なお，図 4.5(c) のようにクリックバーをダブルクロッシングすることで，照準の中心にあるオブジェクトを選択できる．ポインタが照準を横切ろうとするとき以外は，照準はポインタの移動の影響を直接受けないため，選択操作実行時に位置がぶれず，ユーザが任意の位置でクリックの操作を行うことができる．

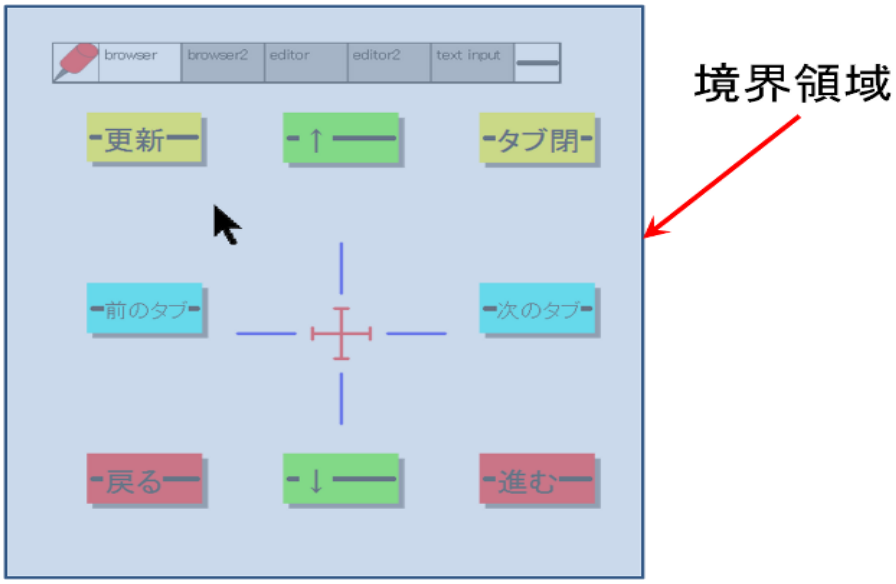


図 4.6: 境界領域

4.4.5　ウィンドウの移動

C.Palette では，常にポインタの周囲に表示するというパレットインタフェースの要件を満たすために,「境界領域による移動」と「クリックラベルによる移動」という 2 種類の移動方法を作成した．この 2 つの方法を利用することにより，メニューインタフェースを常にポインタの周囲に表示することを実現した．なお，ピンアイコンを選択した場合，その場所に固定させることができる．

境界領域による移動

パレットインタフェースでは，常にポインタの周囲に表示し，かつポインタの移動に応じてインタフェースを移動させる必要がある．そのため，C.Palette では，図 4.6 のような境界領域を設定することで，ポインタの移動にあわせてインタフェースを移動させる仕組みを実現している．

C.Palette の位置を固定していない限り，ポインタは常にこの境界領域内に存在し，C.Palette もポインタの周囲に表示されている．もしポインタがこの境界領域を超えて移動しようとした場合，境界領域を超えた量と方向に合わせて C.Palette を移動させ，境界領域も同様に移動した量と方向の分だけ移動する．なお，この境界領域は必ず C.Palette 全体を内包するような領域となっており，パレットウィンドウの切り替えに応じて，境界領域の大きさも変化する．

クリッククラベルによる移動

クリッククラベルを利用してオブジェクトを選択する際，先に述べたとおり，照準を横切るように動かすことでオブジェクトに照準を合わせることができる．このとき，表示されているパレットウィンドウの位置関係を保つために，クリッククラベルの移動に合わせて C.Palette 全体も同じように平行移動する．境界領域についても，移動量に応じて平行移動する．

4.4.6 パレットウィンドウの切り替え

パレットウィンドウを切り替える方法として，一つはアプリケーションなどが切り替わった際にシステムがフォアグラウンドになったアプリケーションに合ったパレットウィンドウに自動的に切り替えるようにした．自動的に切り替えることで，コンテキストメニューのように利用しているアプリケーションに合わせて表示するメニューを切り替えることが容易である．しかし，一つのアプリケーションに対して複数のパレットウィンドウが存在する場合，ユーザの作業状況を正確に把握して自動的に切り替えさせることは難しい．そこで，もう一つの手法としてヘッダーにあるウィンドウタブを選択することによって，表示するパレットウィンドウをユーザが任意のタイミングで切り替えることができる．

パレットウィンドウはウィンドウの大きさや境界領域の大きさがそれぞれで異なるため，パレットウィンドウを切り替えた際に境界領域も選択したパレットウィンドウに合わせて変化する．これにより，ユーザは必要に応じてメニューを自由に切り替えることができる．

4.4.7 実装

C.Palette は，Windows 環境で C++で実装した．C.Palette のウィンドウはデスクトップ上の最前面に透明なウィンドウを作成し，そのウィンドウ上にメニューアイコンなどを描画することで実現している．C.Palette の透明度についても，作成したウィンドウの透明度を設定することで実現している．また，IAcessible インタフェースを利用することで，作業中のウィンドウの情報を取得・解析することでオブジェクトやウェブのリンクの位置や大きさを取得している．取得した情報を利用してクリッククラベルをオブジェクトに合わせるときの補助や利用するパレットウィンドウの自動変更を実現している．

メニューラベルにはフォアグラウンドのアプリケーションに対する操作をショートカットキーを用いて割り当てられている．割り当てる操作はブラウザの次に進むなどの一つのメニュー操作だけではなく，印刷の両面印刷の設定 (「プリンタのプロパティを開く」→「両面印刷を設定」→「プリンタのプロパティを閉じる」) など複数のメニュー操作を割り当てることで，一回の選択で行うことも可能である．

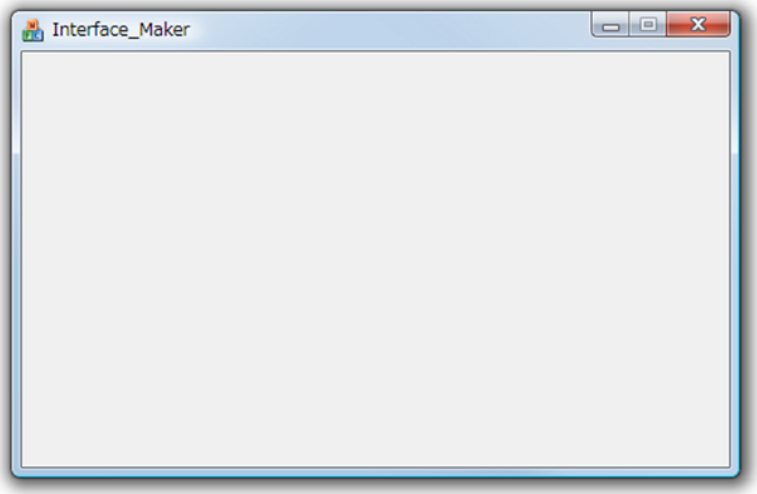


図 4.7: メニューウィンドウ

4.5 パレットウィンドウの作成

C.Palette では，複数のパレットウィンドウを作成するために，パレットウィンドウに関する情報を全て XML で記述し，その XML ファイルを読み込ませることでパレットウィンドウを作成することにした．ひとつの XML ファイルに対して，ひとつのパレットウィンドウが生成され，XML ファイルを編集することでユーザの嗜好などを反映させることができる．

C.Palette では，最初は XML を決めた書式に則って直接記述することによってパレットウィンドウを作成していた．しかし，XML ファイルを直接記述して作成する場合，メニューラベルなどのレイアウトの把握が難しく，記述量も多いため，一つのパレットウィンドウを作成するためにかなりの時間がかかってしまう．また，XML の知識や記述に必要なタグを覚えてい泣ければ，パレットウィンドウを作成することができない．そこで，ユーザがパレットウィンドウを簡単に作成できるインタフェースビルダーを作成した．

このインタフェースビルダーは図 4.7 のパレットウィンドウを作成するためのメニューウィンドウと図 4.8 のパレットウィンドウや各ラベルの情報を設定するプロパティウィンドウからなる．この 2 つのウィンドウを利用してパレットウィンドウの作成が行われる．図 4.7 中に表示されている内容が実際に作成されるパレットウィンドウの見た目となる．そのため，XML の直接記述する場合と比べ，実際に作成されるパレットウィンドウを確認しながら作業することができるため，様々なウィンドウを簡単に作成することができる．

プロパティウィンドウは，作成したメニューラベルの設定を行うためのプロパティとクリックラベルに関するプロパティ，そしてパレットウィンドウ全体の設定を行うためのプロパティの 3 つに分けられる．プロパティウィンドウ上には選択したメニューラベルやクリックラベ

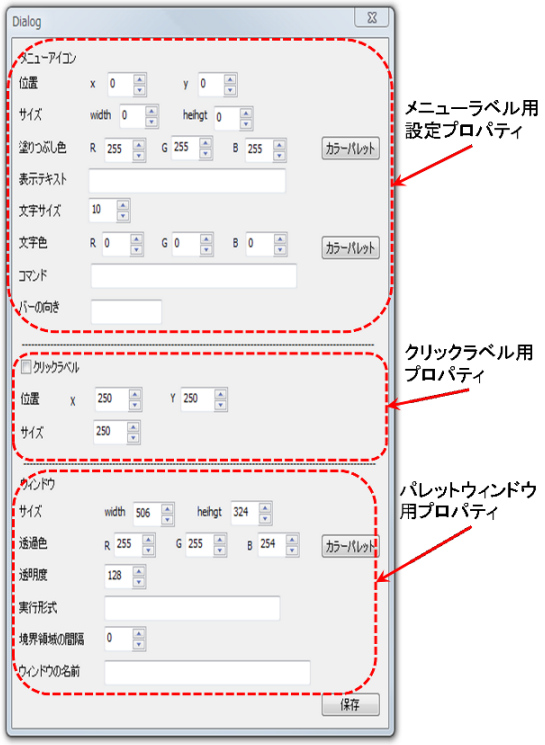


図 4.8: プロパティウィンドウ

ルの情報やパレットウィンドウに関する情報が表示され，これらの情報を編集することでラベルなどの大きさを変更することが可能である．

4.5.1 パレットウィンドウ作成の流れ

このインタフェースビルダーを用いて例えば図 4.2 のあるようなパレットウィンドウを作成する場合,「パレットウィンドウの設定」,「メニューラベルの作成」,「クリックラベルの設定」の 3 つの作業を行うことで作成していく．

パレットウィンドウの設定

パレットウィンドウを作成する場合，まずパレットウィンドウの大きさを決める必要がある．パレットウィンドウの大きさは，メニューウィンドウの縁をドラッグすることで変更することができ，変更したウィンドウのサイズは図 4.8 のパレットウィンドウ用プロパティのサイズの欄に反映される．また，図 4.8 のサイズの項目の値を直接編集することでも設定することができ，大きさを微調整することができる．

パレットウィンドウは大きさを決めた後に，透明度や実行形式などを設定する必要がある．パレットウィンドウの大きさ以外に「透過色と透明度」,「実行形式」,「境界領域の間隔」,「ウィンドウの名前」を設定することができ，これらの項目は全て図 4.8 の各項目を直接編集することで行う．境界領域の間隔については，パレットウィンドウの端から何ピクセル大きくして覆うようにするかを決める．ウィンドウの名前はヘッダーのウィンドウタブに表示されるテキストである．なお，これらの各項目は修正が必要になった際に随時行うことができる．

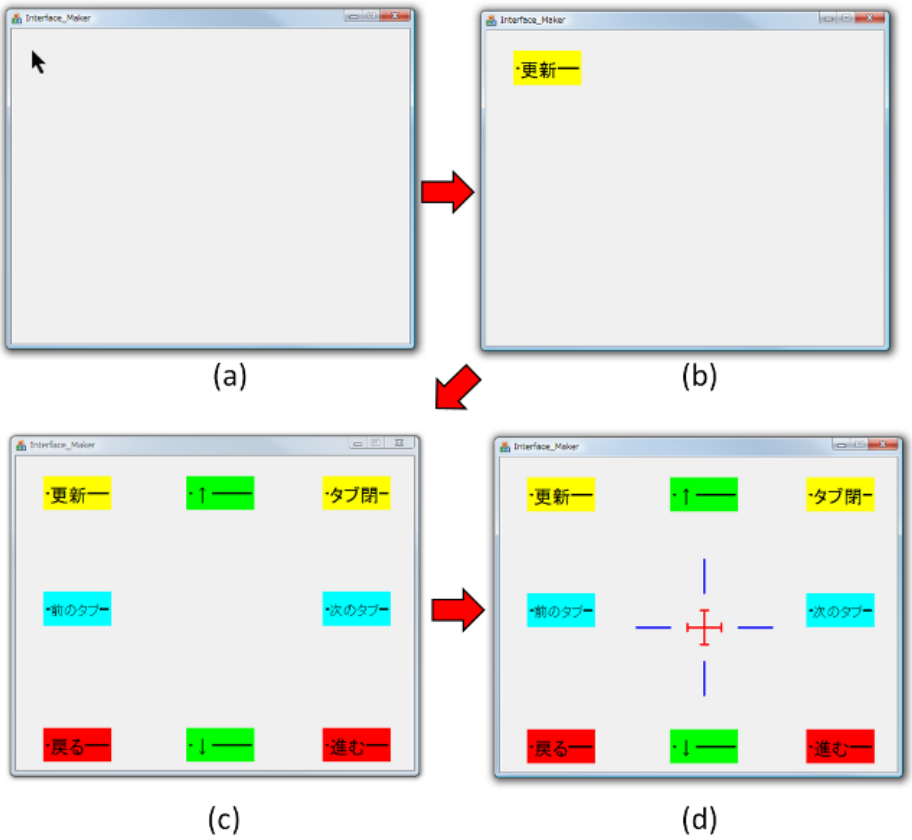


図 4.9: パレットウィンドウ作成過程

メニューラベルの作成

パレットウィンドウの設定が終了したら，次にメニューラベルを作成していく．メニューラベルを作成する場合，まずメニューラベルの大きさを決め，次に実行形式など細かい項目を設定していく．

例えば，図 4.9 の (a) のポインタのある場所に「更新」というラベルを作成したい場合，まずポインタのある場所でドラッグを開始することで，その場所を基準として新しいメニューラベルが作成される．メニューラベルはペイントソフトで矩形を描画するように，ドラッグ

している間は，大きさを自由に変更することができる．そして，ドラッグを終了した時点でそのときの大きさでメニューラベルが作成される．

作成した直後のメニューラベルは矩形があるのみで，ラベルの色や表示テキストなどが設定されていない．これらの項目は，図 4.8 中のメニューラベル用プロパティの各項目を編集することで設定する．新しいメニューラベルの各パラメータはプロパティウィンドウ中に反映される．設定できる項目として,「位置とサイズ」,「塗りつぶしの色」,「表示するテキストに関する情報」,「コマンド」,「クロッシング用のバーの向き」がある．色の設定については，直接 RGB 値を入力しても設定できるが，実際の色がつかみにくいため，カラーパレットのボタンを選択することで，カラーチューザが起動し，実際に表示される色を確認しながら設定することもできる．これらの項目は編集することで，メニューウィンドウ中のメニューラベルに反映され，全ての項目を編集することで，図 4.9 の (b) のような状態となる．

この作業を繰り返すことで必要なメニューラベルを作成することができるが，途中で既に作成したメニューラベルの配置などの修正が必要になることがある．既に作成したメニューラベルを修正する場合，まず変更したいメニューラベルを選択する．メニューラベルは，メニューウィンドウ中のメニューラベルをクリックすることで選択される．選択したメニューラベルは，ドラッグすることで位置を動かすこともできる．選択されたメニューラベルの情報は図 4.8 のメニューラベル用プロパティの各項目に反映され，反映された情報を修正することで各項目の修正を行うことができる．不必要なラベルを削除したい場合には，そのラベルを選択した後にデリートキーを押すことで削除することができる．この作成と修正作業を繰り返すことで，図 4.9 の (c) のようなパレットウィンドウが作成される．

4.5.2 クリッククラベルの設定

クリッククラベルについては，本インタフェースメーカではひとつだけ配置できるようにした．また，クリッククラベル自体の形や大きさはある程度決まっているため，設定は全て図 4.8 のクリッククラベル用プロパティの各項目を直接編集することで行う．

クリッククラベルを配置する場合，まずクリッククラベル用プロパティにあるチェックボックスにチェックを入れることでクリッククラベルが表示される．クリッククラベルは「位置」と「サイズ」を設定することで配置する.「位置」はクリッククラベルの中心（照準の中心）の設定を行い,「サイズ」はクリッククラベルの全体の大きさを決めることができる．これらの情報を設定することで，メニューウィンドウの方にも反映される．クリッククラベルが不必要になった場合には，チェックボックスのチェックを外すことで削除することができる．

図 4.9 の (c) の状態にクリッククラベルを配置した様子が図 4.9 の (d) でなる．この作成したパレットウィンドウを保存し，C.Palette に読み込ませることで，図 4.2 のように実際に利用することができる．保存については，図 4.8 の右下にある保存ボタンを押すことで自動的に XML 形式に変換され，XML ファイルとして保存される．

4.5.3 メニュー操作の割り当て方法

メニューラベルの選択の際に行われる操作は，基本的には，キーボードのキーイベントをエミュレートし，アプリケーションのショートカットを実行することで操作が行われている．そのため，図 4.8 のコマンドには，実際にショートカットなどに使用するキーボードのキーを入力する．また，複数のキーの同時押しさせる場合，半角スペースで区切って入力することで実現している．例えば，コピーの操作を割り当てる場合は,「CTRL C」と記述することで,そのメニューラベルにコピーの操作を割り当てることができる．

メニューラベルには複数のショートカットキーを割り当てることで，複数の操作を一度の選択で行うことができる．複数のショートカットキーを割り当てる場合，記述する際に括弧でくくることで複数の操作を割り当てることが可能になる．メニューラベル選択時のショートカットの実行は左側から順番に行われる．例えば，コマンドに「(CTRL C) (CTRL V)」と記述したメニューラベルを選択した場合，まずコピーの操作が行われ，コピーの操作が実行された後に貼り付けの操作が実行される．

4.6 パレットインタフェースに関する実験

本実験では，作成したパレットインタフェースについて，大画面環境において常にユーザの周囲に表示されることが有効であるかを検証するための実験を行った．本実験では，周囲に表示されることにより作業の効率化を図ることができるかや，表示に対する煩わしさについて調べた．

4.6.1 実験内容

本実験では，ひとつのウィンドウ上での作業と 2 つのウィンドウを互いに行き来する作業の 2 つ状況を想定し，次の 2 つのタスクを行った．

- ブラウザを利用して研究室のトップページ[1]から指定した個人のページへと移動する（タスク 1）
- あるウィンドウに表示された情報の各項目を別のウィンドウの対応する項目にコピーする（タスク 2）

タスク 1 では，ひとつのウィンドウ上で全ての作業を行い，リンクの選択や画面のスクロールなどを利用して作業を行う．タスク 2 では，2 つのウィンドウを行き来して，コピー＆ペーストといった同じ作業を繰り返す作業となっている．

これらのタスクでは，2 種類のメニューインタフェースを利用して行ってもらった．ひとつは，画面上のある場所に固定したメニューインタフェースである．もうひとつは，パレットインタフェースとして，C.Palette を利用した．

[1] http://www.iplab.cs.tsukuba.ac.jp/index-j.html

図 4.10: 固定メニューの配置

4.6.2　インタフェースの詳細

固定メニューで用いたメニューの詳細は図 4.10 の通りである．タスク 1 では，10 個のメニューラベルを用意し，タスク 2 ではメニューラベルは 3 個用意した．メニュー同士は矩形の端からそれぞれ 30 ピクセルずつ離れており，実行線の間隔は垂直方向に 80 ピクセル離して配置した．また，メニューのサイズは全て 100 × 50 ピクセルで統一した．配置については，できるだけ素早く選択でき，かつ操作中に他のメニューラベルを誤選択しない程度に空けた配置にした．固定メニューのみでは，リンクの選択などの操作を行うことが難しいため，図 4.10 のメニューラベル群とは別にクリックラベルを用意した．このクリックラベルは，ポインタの移動には追従せず，照準をポインタで押すように動かすことでのみ移動するようにした．

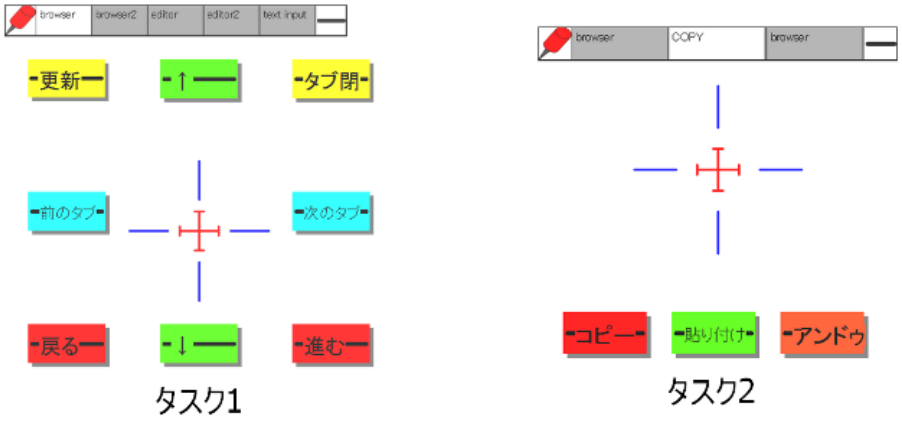


図 4.11: パレットインタフェースの配置

パレットインタフェースの場合は，図 4.11 の通りである．パレットインタフェースの場合も同様に，タスク 1 で 10 個，タスク 2 で 3 個のメニューラベルを用意した．メニューラベル同士は，タスク 1 で 80 ピクセル以上，タスク 2 で 30 ピクセル離れていた．配置については，固定メニューの場合のと異なり，メニューラベルが作業領域を見づらくしないようにある程度分散させて配置した．なお，パレットインタフェースの場合もメニューラベルは全て 100 × 50 ピクセルとした．

メニューラベルの内容については，タスク 1 では,「上下のスクロール」,「ページの更新」,「タブを閉じる」,「タブの切り替え」,「ページの進む／戻る」を用意した．タスク 2 では,「コピー」,「貼り付け」,「アンドゥ」の 3 種類を用意した．

図 4.12: タスク 1 の画面の配置

4.6.3 実験環境

実験の環境は，プロジェクタ 3 台を利用した大画面（横約 6m ×縦約 2m，解像度 3840 × 1024 ピクセル）で，被験者には画面から約 3m 離れた場所に座ってもらった．タスク 1 では，図 4.12 のように画面中央の 3 分の 1 の領域を占めるようにブラウザを配置した．タスク 2 では，図 4.13 のように画面中央の 3 分の 1 の領域にコピー元となる情報が表示されたウィンドウを配置し，画面左の 3 分の 1 を占める領域にコピー先となるウィンドウを配置した．固定メニューの場合は，図 4.12 や図 4.13 のように画面の右上の方に配置した．

指先の認識方法や利用したカメラについては，第 3.3 節と同じものを利用した．ただし，利用した PC のみ異なり，実験に利用した PC は DELL 社製の PRECISION T3400 で，CPU:intel Core2 Duo 2.16GHz，RAM:2.0GB，HDD:250GB，OS:Windows Vista であった．指先の認識，ポインタの移動，大画面への出力，およびメニューの表示・処理は全てこの 1 台のみで行った．

4.6.4 実験手順

被験者は，研究室の学生 8 名とし，以下の手順で行った．

1. 利用してもらう手法についての説明を行い，被験が慣れるまで自由に利用してもらった．
2. 被験者が操作手法を十分に理解したところで，タスク 1 を 2 回行ってもらった．

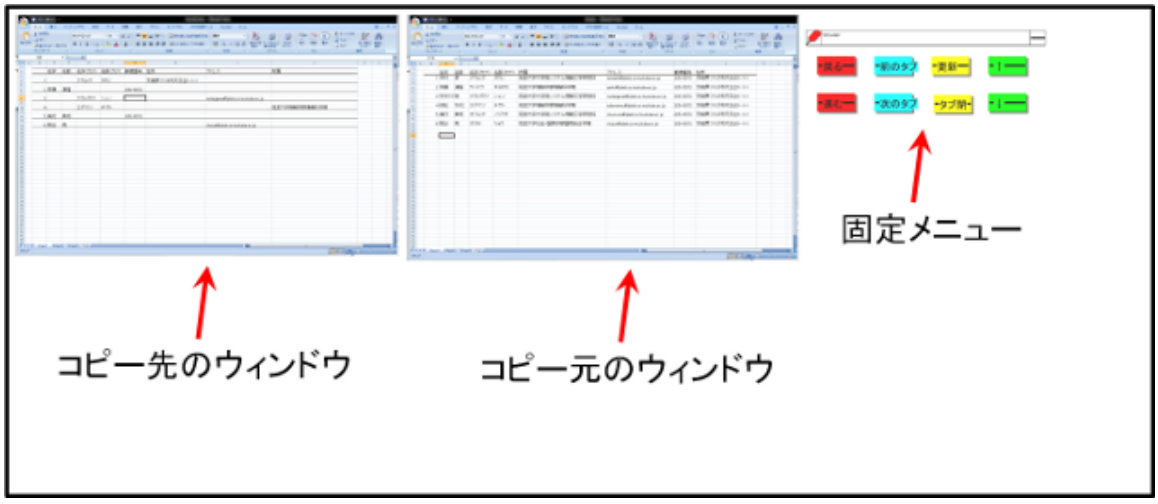


図 4.13: タスク 2 の画面の配置

3. タスク 2 を 2 回行ってもらった.

4. 被験者に使いやすさなどに関するコメントをもらった.

5. 他の手法でも 1. から 4. までの作業を同様に行った.

タスク間では被験者は自由に休憩を取ることができ，被験者が十分だと思ったところで次のタスクに移ってもらった．利用してもらうメニューインタフェースの順番については，慣れなどによる影響などを考慮し，被験者毎に使う順番を変えることで，カウンターバランスをとった.

4.6.5 実験結果

タスク 1 の操作時間と移動量の平均は図 4.14，タスク 2 の操作時間と移動量の平均は図 4.15 の通りである．グラフ中の誤差バーは標準誤差である．手法による操作時間の差については，タスク 1 で約 10 秒，タスク 2 で約 20 秒の差が見られた．移動量については，タスク 1 で約 3000 ピクセル，タスク 2 で約 25000 ピクセルとタスク 2 では倍近い差が見られた.

得られた結果について，t-検定を行った．操作時間については，タスク 1・2 共に 5 %水準で有意差が見られた（$p < 0.05$）．移動量については，タスク 1・2 共に 1 %水準で有意差が見られた（$p < 0.01$）.

4.6.6 被験者からのコメント

6 人の被験者が固定メニューの場合操作領域とメニューとの間の視線の移動が多くて操作が大変であったとコメントした．また，固定メニューでは作業領域の内容を隠すことはなかっ

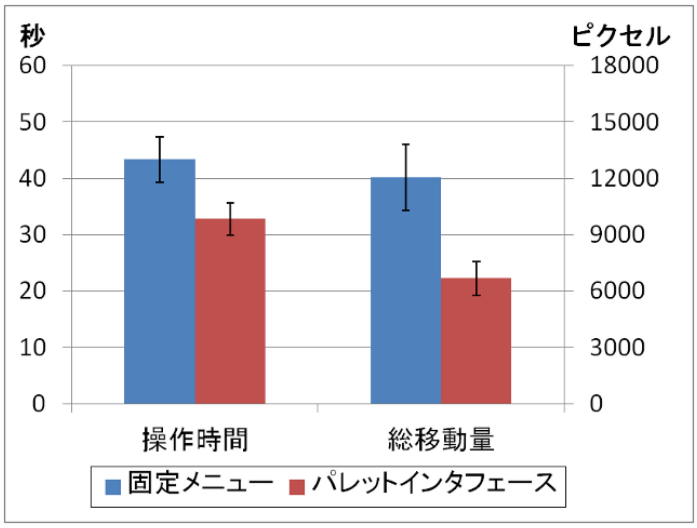


図 4.14: タスク 1 の実験結果

たが，2 カ所を行き来しなければならず，メニュー操作時に操作が正しく行われたかの確認が難しかったというコメントも得られた．

パレットインタフェースについては，被験者全員がメニューが常に追従してくるため，視線の移動が少なくメニュー操作が楽であったと答えた．その一方で，ポインタの周囲に常に表示されているため，2 人の被験者が作業領域が見づらかったと答えた．しかし，作業領域の見づらさについては，気にならないと答えた被験者も 2 人いた．照準を合わせた後にメニュー操作を行おうとしたとき，ポインタが境界領域からはみ出ることで照準がずれてしまったと答えた被験者もいた．

4.6.7 考察

固定メニュー

固定メニューでは，広域にわたって操作するほどポインタの移動量が増加した．タスク 1 では，メニューと操作領域との往復は最低 1 回で済んだため，操作時間や移動量はタスク 2 のような大きな開きはなかった．しかし，タスク 2 では，この往復作業が増加し，また作業領域からメニューまでの距離も増加したため，より大きな差となったものと思われる．また，メニューが操作領域を見にくくすることはなかったが，メニューが常に同じ位置にあったため，被験者は常に視線を移動させながら操作し，選択する度に一度視線を作業領域に移すといった行動が見られた．このような行動も，操作により時間がかかった要因のひとつであると考えられる．

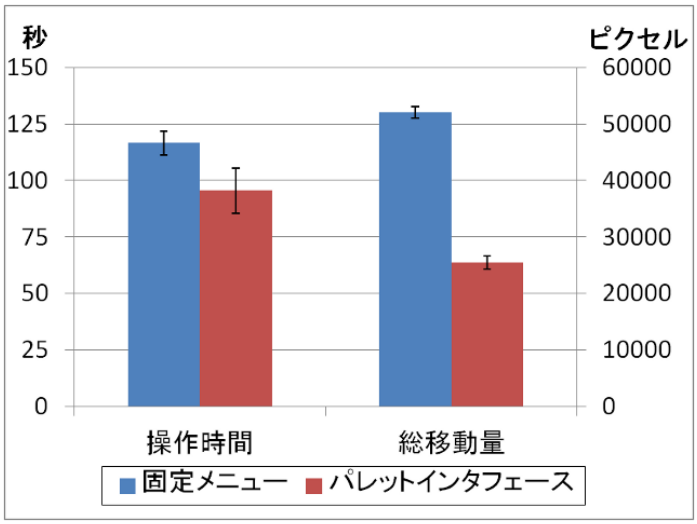


図 4.15: タスク 2 の実験結果

パレットインタフェース

パレットインタフェースは，固定メニューよりも操作時間・ポインタの移動量共に優れた結果となったが，これは，ポインタの移動や視線の移動が少なくなったことが一番の要因であると考えられる．被験者の作業の様子を観察したところ，作業中は視線はポインタの周囲から大きくずれることがなく，被験者はメニュー操作後にすぐに照準の移動に移ったり，照準を合わせてすぐにメニューを選択したりすることができた．

照準を合わせる際も，境界領域を利用した移動で大まかな位置を合わせた後，照準を使って微調整することが多く見られた．照準のみでは，ゆっくり動かすことが多かったが，境界領域による移動では素早く動かす様子が見られたため，固定メニューと比べ，照準を合わせる時間も全体的に短くなった．

その一方で，照準を合わせること自体は問題なく行われたが，その後メニューの選択時にポインタが境界領域をはみ出してしまい，メニュー全体が移動してしまい，それにより照準が動いてしまうことが度々見られた．被験者は，その度に照準の位置を修正しなければならず，操作に余計な時間をかけてしまっていた．この問題については，照準による移動後，もしくはメニューの 1 回目のクロッシング行われた時は，一定時間は境界領域による移動を行わないようにすることで対処することができると考えられる．

第5章　まとめ

本研究では，大画面向けの新しいインタラクション手法として，手の動きによってポインタが移動するポインティングハンドジェスチャを利用した手法を設計した．インタラクション手法を設計するにあたり，オブジェクトの選択手法やポインティングハンドジェスチャ向けのメニューインタフェースの作成を行った．

ポインティングハンドジェスチャによるオブジェクトの選択手法として，クロッシングを利用したダブルクロッシングを作成した．ダブルクロッシングを利用した選択手法では，指先の動きのみで素早く選択することができるため，初見のユーザでもすぐに利用することができることが期待できる．また,「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」の 3 つの手法について，選択時間や誤選択率などを測定し，各手法の特徴などの考察を行った．ウェイティングは選択に時間がかかるが，誤選択が少ない手法であり，選択時間には移動距離やターゲットの大きさの影響を受けやすかった．シングルクロッシングは，選択時間こそ比較的早かったものの，誤選択が多く，移動距離やターゲット同士の距離の影響を受け，大画面環境での利用は難しい結果となった．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．選択時間や誤選択率は，シングルクロッシングと同様に移動距離やターゲット同士の距離の影響を受けやすい．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ティングを利用することが適していると思われる．

ダブルクロッシング以外に，ポインティングハンドジェスチャ向けのメニューインタフェースとしてパレットインタフェースの枠組みを作成した．パレットインタフェースは，ユーザのポインタの周囲に常に表示されるため，操作に必要なメニューをすぐに利用することができ，ユーザの視線移動の減少などを軽減した．また，パレットインタフェースの試作インタフェースとして C.Palette を作成した．C.Palette では，ユーザの嗜好などを反映できるようにするために，XML とインタフェースビルダーを利用することで，メニューを自由に作成することができる．パレットインタフェースと固定メニューに関して比較実験を行い，操作時間の短縮やポインタの移動量が少なくなったことを確認した．

5.1　本研究の貢献

ハンドジェスチャを利用した研究において，本研究は既存の GUI をベースとした上で，ポインティングハンドジェスチャとダブルクロッシングなどを利用することで，少ない動きで

様々な操作を可能にした．ダブルクロッシングについては，既存手法であるクロッシングの単純な改良ではあるが，ポインティングハンドジェスチャにおける問題点を改善し，また実験によりダブルクロッシングと既存の選択手法との差異や各手法の特徴などを明確にした点などが本研究の貢献であると考えられる．ポインティングハンドジェスチャ用のメニューインタフェースについては，専用のメニューインタフェースの枠組みを作成したが，メニューインタフェース内のメニュー項目を自由に作成できるようにすることで，既存の GUI のアプリケーションを利用できるようにした．既存のハンドジェスチャインタラクション手法のようにその手法に合わせた専用アプリケーションを作成する必要が無く，既存のアプリケーションをそのまま利用することができるため，既存の GUI 環境との互換性を高く保持したインタラクション手法を可能にした．その一方で，ユーザの作業領域の近くにメニューを重畳表示するため，作業領域が時々見づらくなってしまうなどの問題点があることも明らかにした．

5.2 今後の展望

今後の展望としては，ポインタの移動を利用した選択手法に関する展望と，ポインティングハンドジェスチャ向けメニューインタフェースに関する展望の 2 つについて述べる．

5.2.1 ポインタの移動を利用した選択手法に関する展望

本研究では，ダブルクロッシングが選択速度と安定性を兼ね備えた手法であることを確認したが，密集したオブジェクトを選択しづらいといった問題があった．一方，ウェイティングは時間がかかっても安定して選択することができる手法であった．そのため，どちらか片方を利用してオブジェクトを選択させるのではなく，2 つの選択手法を組み合わせることで，それぞれの利点を生かしたオブジェクトの選択を行えることが期待できる．2 つの手法を組み合わせるにあたり，どのような状況でどちらの手法を利用すべきかを決める必要があるため，より様々な状況下で 2 つの手法を試行して決めていく必要がある．この線引きをはっきりさせることが，今後の研究として望まれる．

5.2.2 ポインティングハンドジェスチャ向けメニューインタフェースに関する展望

本研究では，ポインタの移動やダブルクロッシングを利用したパレットインタフェースを作成し，ポインタの周囲に常にメニューを表示することで，ユーザの作業の効率を上げることが期待できることを確認した．しかし，ポインタに追従させることでユーザの作業領域を隠しやすいため，メニューが作業の妨げとならないような表示方法を工夫する必要があるなど，改善すべき点が残されている．また，メニューを作成するにあたり，XML の記述やビルダーによる作成は，作業前に行う必要があり，作業中にメニューの配置や内容を入れ替えたい場合には対応が難しい．そのため，作業中でも入れ替えができるようなメニューの交換機能などを追加することで，より柔軟にユーザの嗜好に対応することができることが期待される．

謝辞

本論文を進めて行くにあたり，筑波大学システム情報系情報工学域の田中二郎教授，ならび高橋伸准教授には，多くの御助言をいただきました．また，本研究を進めるにあたり，長期にわたって，研究の進め方，実験方法，論文執筆に関して多大なるご指導をいただきました．ゼミなどでは，三末和男准教授，志築文太郎准教授から様々な助言をいただきました．

本論文の審査では，田中二郎教授，葛岡英明教授，福井幸男教授，秡川友宏准教授，高橋伸准教授に審査していただきました．また審査では，的確なご意見やご指摘をいただき，本論文をまとめることができました．

博士後期課程に進むにあたり，グローバル COE プログラ “サイバニクス：人・機械・情報系の融合複合” に研究補助員として教育・研究プログラムに参加させていただきました．本プログラムでは，経済的支援以外にも，異分野の研究に関する知識などを得る機会をいただきました．また，本プログラムの拠点リーダである筑波大学大学院システム情報工学研究科知能機能システム専攻の山海嘉之教授をはじめ，本プログラムに関わった多くの先生方，活動の補助をしていただいた事務局スタッフの方々には多大なご支援をいただきました．

本研究を進めるにあたり，田中研究室の皆様には，ゼミなどを通じて研究に関する助言や研究活動の支援・協力をしていただきました．特に，田中研究室ユビキタスチームの皆様には毎日の研究活動のご支援をいただきました．

最後に，本研究を進めていくにあたり，協力してくださった皆様に心より感謝申し上げます．ありがとうございました．

参考文献

[1] Accot, J. and Zhai, S.: More than dotting the i's - foundation for crossing-based interfaces, *CHI' 02*, pp. 73–80 (2002).

[2] Aoki, R., Karatsu, Y., Ihara, M., Maeda, A., Kobayashi, M. and Kagami, S.: Gesture Identification Based on Zone Entry and Axis Crossing, *HCII 2011 Part II*, pp. 194–203 (2011).

[3] Apitz, G. and Guimbretiere, F.: CrossY: A Crossing-Based Drawing Application, UIST '04, pp. 3–12 (2004).

[4] Atia, A., Takahashi, S. and Tanaka, J.: Coin Size Wireless Sensor Interface for Interaction with Remote Displays, *HCI International 2007*, pp. 733–742 (2007).

[5] Atia, A., Takahashi, S. and Tanaka, J.: Smart Gesture Sticker: Smart Hand Gestures Profiles for Daily Objects Interaction, *ICIS 2010*, pp. 482–487 (2010).

[6] Bailly, G., Lecolinet, E. and Nigay, L.: Flower menus: a new type of marking menu with large menu breadth, within groups and efficient expert mode memorization, *AVI '08*, pp. 15–22 (2008).

[7] Baudel, T. and Beaudouin-Lafon, M.: CHARADE: Remote Control of Objects using Free-Hand Gestures, *Communications of the ACM Vol.37 No.7*, pp. 28–35 (1993).

[8] Baudisch, P., Cutrell, E., Robbins, D., Czerwinski, M., Tandler, P., Bederson, B. and Zierlinger, A.: 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).

[9] Benko, H., Wilson, A. D. and Baudisch, P.: Precise Selection Techniques for Multi-Touch Screens, *CHI '06*, pp. 1263–1272 (2006).

[10] Bezerianos, A. and Balakrishnan, R.: The vacuum: facilitating the manipulation of distant objects, *CHI '05: Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 361–370 (2005).

[11] Bolt, R. A.: "Put-that-there": Voice and gesture at the graphics interface, *SIGGRAPH '80*, pp. 262–270 (1980).

[12] Bragdon, A. and Ko, H.-S.: Gesture select:: acquiring remote targets on large displays without pointing, *CHI '11*, ACM, pp. 187–196 (2011).

[13] Callahan, J., Hopkins, D., Weiser, M. and Shneiderman, B.: An Empirical Comparsion of Pie vs. Linear Menus, *CHI '88*, pp. 95–100 (1988).

[14] Cao, X. and Balakrishnan, R.: VisionWand: interaction techniques for large displays using a passive wand tracked in 3D, *UIST '03*, pp. 173–182 (2003).

[15] Chen, Q., Malric, F., Zhang, Y., Abid, M., Cordeiro, A., Petriu, E. M. and Georganas, N. D.: Interacting with Digital Signage Using Hand Gestures, *ICIAR 2009*, pp. 347–358 (2009).

[16] Chen, X. and Davis, J.: Multi-user laser-based interaction on large tiled displays, *Displays*, pp. 205–211 (2002).

[17] Cheng, K. and Takatsuka, M.: Estimating virtual touchscreen for fingertip interaction with large displays, *OZCHI '06*, pp. 397–400 (2006).

[18] Collomb, M., Hascoët, M., Baudisch, P. and Lee, B.: Improving drag-and-drop on wall-size displays, *GI '05*, pp. 25–32 (2005).

[19] Dhawale, P., Masoodian, M. and Rogers, B.: Bare-hand 3D gesture input to interactive systems, *CHINZ '06*, pp. 25–32 (2006).

[20] Dominguez, S. M., Keaton, T. and Sayed, A. H.: Robust Finger Tracking for Wearable Computer Interfacing, *PUI 2001*, pp. 1–5 (2001).

[21] Fitzmaurice, G., Matejka, J., Khan, A., Glueck, M. and Kurtenbach, G.: PieCursor: Merging Pointing and Command Selection for Rapid In-place Tool Switching, *CHI '08*, pp. 1361–1370 (2008).

[22] Guimbretière, F., Martin, A. and Winograd, T.: Benefits of merging command selection and direct manipulation, *ACM Trans. Comput.-Hum. Interact.*, Vol. 12, No. 3, pp. 460–476 (2005).

[23] Guimbretiére, F. and Winograd, T.: FlowMenu: combining command, text, and data entry, *UIST '00*, pp. 213–216 (2000).

[24] Gustafson, S., Bierwirth, D. and Baudisch, P.: Imaginary interfaces: spatial interaction with empty hands and without visual feedback, *UIST '10*, pp. 3–12 (2010).

[25] Hisamatsu, T., Shizuki, B., Takahashi, S. and Tanaka, J.: A novel click-free interaction technique for large-screen interfaces, *APCHI2006*, pp. (CD–ROM) (2006).

[26] Hu, K., Canavan, S. and Yin, L.: Hand Pointing Estimation for Human Computer Interaction Based on Two Orthogonal-Views, *ICPR 2010*, pp. 3760–3763 (2010).

[27] Ike, T., Kishikawa, N. and Stenger, B.: A Real-Time Hand Gesture Interface Implemented on a Multi-Core Processor, *MVA2007*, pp. 9–12 (2007).

[28] Jia, Y., Li, S. and Liu, Y.: Tracking pointing gesture in 3D space for wearable visual interfaces, *HCM '07*, pp. 23–30 (2007).

[29] Khan, A., Fitzmaurice, G., Almeida, D., Burtnyk, N. and Kurtenbach, G.: A remote control interface for large displays, *UIST '04*, pp. 127–136 (2004).

[30] Kjeldsen, R. and Kender, J.: Toward the Use of Gesture in Traditional User Interfaces, *FG'96*, pp. 151–156 (1996).

[31] Kobayashi, M. and Igarashi, T.: Ninja Cursors: Using Multiple Cursors to Assist Target Acquisition on Large Screens, *CHI '08*, pp. 949–958 (2008).

[32] Kurtenbach, G. and Buxton, W.: The limits of expert performance using hierarchic marking menus, *CHI '93*, pp. 482–487 (1993).

[33] Lenman, S., Bretzner, L. and Thuresson, B.: Using marking menus to develop command sets for computer vision based hand gesture interfaces, *NordiCHI '02*, pp. 239–242 (2002).

[34] Lucente, M., Zwart, G.-J. and George, A. D.: Visualization Space: A Testbed for Deviceless Multimodal User Interface, *Intelligent Environments '98*, pp. 87–92 (1998).

[35] Lucero, A., Aliakseyeu, D. and Martens, J.-B.: Funky wall: presenting mood boards using gesture, speech and visuals, *AVI '08*, pp. 425–428 (2008).

[36] Malik, S., Ranjan, A. and Balakrishnan, R.: Interacting with Large Displays from a Distance with Vision-Tracked Multi-Finger Gestural Input, *UIST'05*, pp. 43–52 (2005).

[37] Meiguins, B. S., do Carmo, R. M. C. and et al., L. A.: Multidimensional Information Visualization Using Augmented Reality, *VRCIA '06*, pp. 391–394 (2006).

[38] Miyaoku, K., Higashino, S. and Tonomura, Y.: C-Blink: A Hue-Difference-Based Light Signal Marker for Large Screen Interaction via Any Mobile Terminal, *UIST '04*, pp. 147–156 (2004).

[39] Nakamura, T., Takahashi, S. and Tanaka, J.: One-finger Interaction for Ubiquitous Environment, *ICIS 2010*, pp. 267–272 (2010).

[40] Nickel, K. and Stiefelhagen, R.: Pointing Gesture Recognition based on 3D-Tracking of Face, Hands and Head Orientation, ICMI '03, pp. 140–146 (2003).

[41] Olwal, A., Feiner, S. and Heyman, S.: Rubbing and Tapping for Precise and Rapid Selection on Touch-Screen Displays, *CHI '08: Proceedings of the SIGCHI conference on Human factors in computing systems*, pp. 295–304 (2008).

[42] Perlin, K.: Quikwriting: Continuous Stylus-based Text Entry, *UIST 1998* (1998).

[43] Rahman, A. M., Hossain, M. A., Parra, J. and El Saddik, A.: Motion-path based gesture interaction with smart home services, *MM '09*, pp. 761–764 (2009).

[44] Rautaray, S. S. and Agrawal, A.: A novel human computer interface based on hand gesture recognition using computer vision techniques, *IITM '10*, pp. 292–296 (2010).

[45] Robertson, G., Czerwinski, M., Baudisch, P., Meyers, B., Robbins, D., Smith, G. and Tan, D.: The Large-Display User Experience, *IEEE Comput. Graph. Appl.*, Vol. 25, No. 4, pp. 44–51 (2005).

[46] Schall, G., Mendez, E. and Schmalstieg, D.: Virtual redlining for civil engineering in real environments, *ISMAR '08*, pp. 95–98 (2008).

[47] Schick, A., van de Camp, F., Ijsselmuiden, J. and Stiefelhagen, R.: Extending touch: towards interaction with large-scale surfaces, *ITS '09*, pp. 117–124 (2009).

[48] Černeková, Z., Malerczyk, C., Nikolaidis, N. and Pitas, I.: Single camera pointing gesture recognition for interaction in edutainment applications, *SCCG '08*, pp. 121–125 (2008).

[49] Vogel, D. and Balakrishnan, R.: Distant Freehand Pointing and Clicking on Very Large, High Resolution Displays, *UIST'05*, pp. 33–42 (2005).

[50] Wachs, J. P., Kölsch, M., Stern, H. and Edan, Y.: Vision-based hand-gesture applications, *Commun. ACM*, Vol. 54, No. 2, pp. 60–71 (2011).

[51] Wang, R. Y. and Popović, J.: Real-time hand-tracking with a color glove, *ACM SIGGRAPH 2009 papers*, SIGGRAPH '09, pp. 63:1–63:8 (2009).

[52] Zhao, S., Agrawala, M. and Hinckley, K.: Zone and polygon menus: using relative position to increase the breadth of multi-stroke marking menus, *CHI '06*, pp. 1077–1086 (2006).

[53] 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 2009 年 9 月号，Vol.91 No.09，pp. 48–53 (2009).

[54] 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，情報処理学会論文誌 Vol.43 No.12，pp. 3675–3684 (2002).

[55] 小山慎哉，小野寺光，葛岡英明，山崎敬一：レーザポインタ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).

[56] 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).

[57] 西村托一，向井理朗，　岡隆一：白黒動画像からの形状特徴を用いたジェスチャのスポッティング認識システム，電子情報通信学会論文誌 D-II Vol.J81-D-II No.8，pp. 1812–1821 (1998).

[58] 前野恭平，藤田誠司，木村朝子，柴田史久，田村秀幸：ジェスチャ操作を活用する広視野電子作業空間のためのメニューデザインの検討，情報処理学会研究報告 2007-HCI-125 (3)，pp. 17–24 (2007).

[59] 蔵田武志，興梠正克，加藤丈和，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，映情学技報, VIS2001-103,Vol.25,No.85，pp. 43–52 (2005).

[60] 天田泰亨，鈴木基之，後藤英昭，牧野正三：動作位置の変化に頑健なジェスチャ認識，電子情報通信学会技術研究報告. PRMU, パターン認識・メディア理解 Vol.99 No.448，pp. 39–46 (1999).

[61] 佐藤大介，志築文太郎，三浦元喜，田中二郎：Popie：フローメニューに基づく日本語入力手法，情報処理学会論文誌, Vol.47, No.7，pp. 2305–2316 (2006).

[62] 中村　卓，佐藤大介，高橋　伸，志築文太郎，田中二郎：手の軌跡と FlowMenu を組み合わせたインタラクション手法，WISS 2007 予稿集，pp. 181–182 (2007).

[63] 中村聡史，塚本昌彦，西尾章治朗：Protractor: 機能の可視化に注目した両手操作可能な図形描画システム，WISS2000 予稿集，pp. 1–6 (2000).

[64] 木村朝子，鶴田剛史，酒井 理生他：広視野電子作業空間に関する考察とシステム試作〜マイノリティ・リポート型 I/F とその発展形，インタラクション 2005 論文集，pp. 143–150 (2005).

[65]　陳欣蕾，小池英樹，中西泰人，佐藤洋一，小林貴訓：机型実世界指向システムにおける両手による直接操作の評価，*WISS 200*, pp. 215–216 (2000).

[66] 深澤哲生，福地健太郎，小池英樹：壁型ディスプレイを用いた非接触対話型電子広告システム，WISS 2006 予稿集，pp. 39–46 (2006).

[67]　畠直志，岩井儀雄，谷内田正彦：動き情報と情報圧縮を用いたロバストなジェスチャ認識手法，電子情報通信学会論文誌 D-II Vol.J81-D-II No.9，pp. 1983–1992 (1998).

[68] 木村朝子，柴田史久，鶴田剛史他：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，情報処理学会論文誌，Vol. 47，No. 4，pp. 1327–1339 (2006).

著者論文リスト

本研究に関する論文

論文誌

- 中村 卓，高橋 伸，田中 二郎”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-”, 電子情報通信学会論文誌 Vol.J96-D, NO.4, Apr.2013（掲載予定）.

査読付き国際会議

- Takashi Nakamura, Shin Takahashi, and Jiro Tanaka ”Double-crossing: a new interaction technique for hand gesture interfaces”, Proceedings of the 8th Asia-Pacific Conference on Computer-Human Interaction (APCHI2008), pp.292-300, Seoul, Korea, July 6-9, 2008.
- Takashi Nakamura, Shin Takahashi, and Jiro Tanaka ”One-finger Interaction for Ubiquitous Environment”, Proceedings of 9th IEEE/ACIS international conference on computer and information science(ICIS2010), pp.267-272, Yamagata, Japan, August 18-20, 2010.

その他の論文

- 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” , 情報処理学会第 70 回大会, 2008 年 3 月.

本研究に関連しない論文リスト

論文誌

- 高橋 伸，中村 卓，田中 二郎”漫画的手法を用いたライブカメラ画像上へのプレゼンス情報の表示，” コンピュータソフトウェア, Vol.24, No.3, 2007, pp. 29-40.

査読付き国内会議

- 高橋 伸, 岩淵志学, ジャッキーノ ヤン, 山田徹, 久松孝臣, 中村卓, 土持幸久, 金春明, 田中二郎”ライブカメラ画像を用いたプレゼンス情報の表示手法.” , 第 13 回インタラクティブシステムとソフトウェアに関するワークショップ (WISS 2005), 2005 年 12 月, pp.15-18.

査読無し国内会議

- 中村卓，高橋伸，田中二郎”ハンドジェスチャを用いた公共大画面向けインタフェース”, マルチメディア、分散、協調とモバイル（DICOMO2006）論文集, 情報処理学会, 2006 年 7 月, pp.833-836.

その他の論文

- 中村卓，高橋伸，田中二郎”Hands-Popie:両手の動きを利用した日本語入力手法” 第 14 回インタラクティブシステムとソフトウェアに関するワークショップ (WISS2006), 日本ソフトウェア科学会, 2006 年 12 月, pp.151-152.
- 中村卓，佐藤大介，高橋伸，志築文太郎，田中二郎”手の軌跡と FlowMenu を組み合わせたインタラクション手法” 第 15 回インタラクティブシステムとソフトウェアに関するワークショップ (WISS2007), 日本ソフトウェア科学会, 2007 年 12 月, pp.181-182.
- 中村卓，南竹俊介，鈴木茂徳，林恵理奈，竹林綾子，高橋伸，田中二郎”FestiCam：賑やかさを伝えるライブカメラ映像への情報提示手法”, 第 17 回インタラクティブシステムとソフトウェアに関するワークショップ (WISS2009), 日本ソフトウェア科学会, 2009 年 12 月, pp.119-120.
- 中村卓，高橋伸，田中二郎”手の移動方向と回転ジェスチャを用いたインタラクション手法” インタラクション 2010, 情報処理学会, 2010 年 3 月.(DVD-ROM).
- 中村卓，高橋伸，田中二郎”HMD を用いた拡張現実感技術向け物体情報の視覚的表現・比較手法”, 情報処理学会第 72 回大会, 2010 年 3 月 (CD-ROM).
- Takashi Nakamura, Shin Takahashi and Jiro Tanaka ”Object information visualization and comparison technique using augmented reality”, 4th International Workshop on Cybernics (IWC2011) , 4 Pages(CD-ROM), Tokyo, Japan, March 9th, 2011

付録：C.PaletteのXMLの文書形式

基本構成

C.Paletteでは，interface要素をルートとした構造となっており，その下に，パレットウィンドウに関するWindow，メニューラベルに関するMenuIcon，クリックラベルに関するClickLabelの3つの要素から構成される．それぞれの要素はさらに以下の属性を持っている．

- Window
  - width
    ウィンドウ全体の横幅（整数値）.
  - height
    ウィンドウ全体の高さ（整数値）.
  - transColor
    透明処理を行うための色の指定．指定した色がウィンドウの背景色として指定され，その色は完全に透明になる．ほかの色は指定された透過率で透過される．transColorはさらにr，g，bの3つの属性を持ち，0〜255の整数値で指定する．
  - transparency
    透過率の指定．0〜255の整数値.
  - execute-type
    メニューラベルなどの選択方法の指定．この属性値が“waiting”であればウェイティング，“single-crossing”であればシングルクロッシング，“double-corssing”であればダブルクロッシングとなる．それ以外の値や指定がない場合は全てダブルクロッシングとして処理される．
  - gap
    境界領域の設定（0以上の整数値）．C.Paletteのウィンドウの縁からこの属性値だけ離れた場所に境界領域を作成する．
  - tag-name
    ヘッダーのタグに表示する文字列.
- MenuIcon

- position
  メニューラベルを表示する位置（左上の点）．x と y の属性を持ち，パレットウィンドウ上の位置座標を指定する（0 以上の整数値）．
- size
  メニューラベルの大きさ．width と height の属性を持っている．0 以上の整数値で指定する．
- fillColor
  メニューラベルの塗りつぶしの色．r，g，b の 3 つの属性を持ち，0〜255 の整数値で指定する．
- text
  メニューラベルの文字表示に関する要素で以下の 3 つの属性を持つ．
  - name
    メニューラベルに表示する文字列．
  - fontSize
    文字の大きさ．整数値で指定するが，実際の文字の大きさはワードパットなどにおける文字サイズの 10 分の 1 の値のサイズと同等になる．
  - fontColor
    文字列の色の指定．r，g，b の 3 つの属性を持ち，0〜255 の整数値で指定する．
- command
  そのメニューラベルが選択された際に実行されるコマンド．コマンドの詳細は後述する．
- bar-direction
  クロッシング判定用のバーの向き．属性値が “vertical” の場合はメニューラベルの中央を垂直方向に横切るように判定線が作成される．“horizontal” またはそれ以外の値の場合はメニューラベルの中央を水平に横切るように判定線が作成される．

- ClickLabel
  - position
    クリックラベルの位置を指定し，その位置にクリックラベルの照準の中心が来るように配置する．x と y の属性を持ち，パレットウィンドウ上の位置座標を指定する（0 以上の整数値）．
  - size
    クリックラベルの大きさを指定する．縦横ともこの大きさで作成される．なお，大きさは左側（上側）のクリックバーの左端（上端）から右側（下側）のクリックバーの右端（下端）までであり，この属性値を基にシステム側が照準やクリックバーを自動生成する．

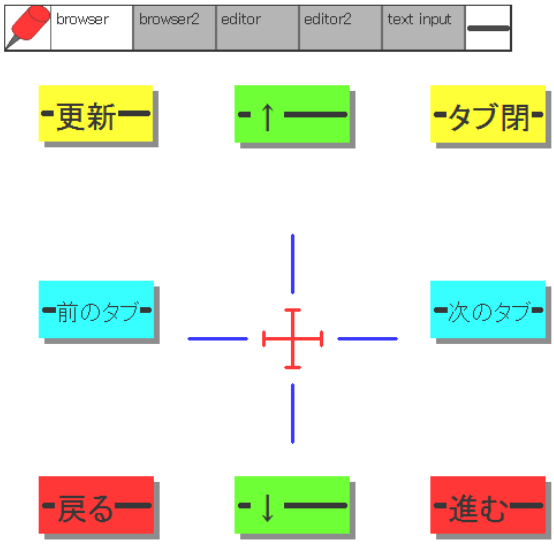
実行コマンドについて

MenuIcon の子要素である command の属性値を設定することで，メニューラベルのコマンドを実行することができる．属性値は “CTL C” などと入力することで決められるが，コマンドの作成に当たっていくつか規則がある．一つ目は，入力可能な操作はキーボードのキーと一部のマウス操作である．二つ目は，半角スペースで区切ることで，複数のコマンドを同時に行うことができる（キーボードショートカットキーと同様の操作が可能）．三つ目は，半角括弧で区切ることで，複数の操作をひとつのコマンドとしてまとめることができる（例えば，“(CTL C) (CTL V)” とすることで CTL+C と CTL+V の 2 つの操作にできる）．三つ目の規則については，ひとつも括弧がない場合はそこに書かれている内容を全てひとつのコマンドとして扱われる（全て同じタイミングでボタンが押される）．

利用可能な操作については，以下のとおりである．なお，全て半角で記述し，大文字・小文字は関係ない．また，マウスのクリック操作に関しては，クリックラベルの照準の中心で実行され，クリックラベルがない場合は何も起こらない．

- a〜z，0〜9 ：それぞれに該当するキー
- f1〜f12 ：ファンクションキー
- esc，escape ：エスケープキー
- shift ：シフトキー
- tab ：タブキー
- ctrl, control ：コントロールキー
- alt ：alt キー
- space ：スペースキー
- UP，DOWN，LEFT，RIGHT ：上下左右キー
- enter，return ：リターンキー
- delete ：デリートキー
- backspace，bs ：バックスペースキー
- pageup，pagedown ：ページアップ，ページダウンキー
- home ：ホームキー
- l_click，s_click ：マウスの左クリック（シングルクリック）
- d_click ：マウスの左クリック（ダブルクリック）

- r_click ：マウスの右クリック
- m_click ：マウスの中クリック



XML サンプル

例えば，上図のようなブラウザ用のパレットウィンドウを作成しようとした場合，このウィンドウを作成するための XML ファイルは以下のようになる．

```
<?xml version="1.0" encoding="Shift_JIS" standalone="yes"?>
<Interface>
    <Window>
        <width>500</width>
        <height>500</height>
        <transColor>
            <r>255</r>
            <g>255</g>
            <b>254</b>
        </transColor>
        <transparency>168</transparency>
        <excute-type>double-crossing</excute-type>
        <gap>30</gap>
        <tab-name>browser</tab-name>
```

```
</Window>
<ClickLabel>
    <position>
        <x>250</x>
        <y>250</y>
    </position>
    <size>225</size>
</ClickLabel>
<MenuIcon>
    <position>
        <x>200</x>
        <y>30</y>
    </position>
    <size>
        <width>100</width>
        <height>50</height>
    </size>
    <fillColor>
        <r>128</r>
        <g>255</g>
        <b>0</b>
    </fillColor>
    <text>
        <name>↑</name>
        <fontSize>200</fontSize>
        <fontColor>
            <r>0</r>
            <g>0</g>
            <b>0</b>
        </fontColor>
    </text>
    <command>PAGEUP</command>
    <bar-direction>horizontal</bar-direction>
</MenuIcon>
<MenuIcon>
    <position>
        <x>370</x>
        <y>30</y>
```

```
    </position>
    <size>
        <width>100</width>
        <height>50</height>
    </size>
    <fillColor>
        <r>255</r>
        <g>255</g>
        <b>0</b>
    </fillColor>
    <text>
        <name>タブ閉</name>
        <fontSize>200</fontSize>
        <fontColor>
            <r>0</r>
            <g>0</g>
            <b>0</b>
        </fontColor>
    </text>
    <command>CTRL F4</command>
    <bar-direction>horizontal</bar-direction>
</MenuIcon>
<MenuIcon>
    <position>
        <x>370</x>
        <y>200</y>
    </position>
    <size>
        <width>100</width>
        <height>50</height>
    </size>
    <fillColor>
        <r>0</r>
        <g>255</g>
        <b>255</b>
    </fillColor>
    <text>
        <name>次のタブ</name>
```

```
        <fontSize>150</fontSize>
        <fontColor>
            <r>0</r>
            <g>0</g>
            <b>0</b>
        </fontColor>
    </text>
    <command>CTRL TAB</command>
    <bar-direction>horizontal</bar-direction>
</MenuIcon>
<MenuIcon>
    <position>
        <x>30</x>
        <y>30</y>
    </position>
    <size>
        <width>100</width>
        <height>50</height>
    </size>
    <fillColor>
        <r>255</r>
        <g>255</g>
        <b>0</b>
    </fillColor>
    <text>
        <name>更新</name>
        <fontSize>200</fontSize>
        <fontColor>
            <r>0</r>
            <g>0</g>
            <b>0</b>
        </fontColor>
    </text>
    <command>F5</command>
    <bar-direction>horizontal</bar-direction>
</MenuIcon>
<MenuIcon>
    <position>
```

```
            <x>30</x>
            <y>200</y>
        </position>
        <size>
            <width>100</width>
            <height>50</height>
        </size>
        <fillColor>
            <r>0</r>
            <g>255</g>
            <b>255</b>
        </fillColor>
        <text>
            <name>前のタブ</name>
            <fontSize>150</fontSize>
            <fontColor>
                <r>0</r>
                <g>0</g>
                <b>0</b>
            </fontColor>
        </text>
        <command>CTRL SHIFT TAB</command>
        <bar-direction>horizontal</bar-direction>
    </MenuIcon>
    <MenuIcon>
        <position>
            <x>30</x>
            <y>370</y>
        </position>
        <size>
            <width>100</width>
            <height>50</height>
        </size>
        <fillColor>
            <r>255</r>
            <g>0</g>
            <b>0</b>
        </fillColor>
```

```
    <text>
        <name>戻る</name>
        <fontSize>200</fontSize>
        <fontColor>
            <r>0</r>
            <g>0</g>
            <b>0</b>
        </fontColor>
    </text>
    <command>ALT LEFT</command>
    <bar-direction>horizontal</bar-direction>
</MenuIcon>
<MenuIcon>
    <position>
        <x>200</x>
        <y>370</y>
    </position>
    <size>
        <width>100</width>
        <height>50</height>
    </size>
    <fillColor>
        <r>128</r>
        <g>255</g>
        <b>0</b>
    </fillColor>
    <text>
        <name>↓</name>
        <fontSize>200</fontSize>
        <fontColor>
            <r>0</r>
            <g>0</g>
            <b>0</b>
        </fontColor>
    </text>
    <command>PAGEDOWN</command>
    <bar-direction>horizontal</bar-direction>
</MenuIcon>
```

```
    <MenuIcon>
        <position>
            <x>370</x>
            <y>370</y>
        </position>
        <size>
            <width>100</width>
            <height>50</height>
        </size>
        <fillColor>
            <r>255</r>
            <g>0</g>
            <b>0</b>
        </fillColor>
        <text>
            <name>進む</name>
            <fontSize>200</fontSize>
            <fontColor>
                <r>0</r>
                <g>0</g>
                <b>0</b>
            </fontColor>
        </text>
        <command>ALT RIGHT</command>
        <bar-direction>horizontal</bar-direction>
    </MenuIcon>
</Interface>
```