平成20年度

筑波大学第三学群情報学類

卒業研究論文

題目
Rough Selecting: リモートポインティングと方向キー操作を組み合わせたアイコン選択手法

主専攻　　情報科学主専攻

著者　　藤原 仁貴

指導教員　志築文太郎　高橋伸　三末和男　田中二郎

要　旨

ユーザが画面から離れた場所において操作を行う大画面向けポインティングデバイス（リモートポインティングデバイス）の研究や開発が行われている．しかし，これらのデバイスには，ユーザがデバイスのボタンを押した時にデバイスがぶれる，ポイント位置を一カ所に止めておく事が難しい，ユーザにとって細かい操作を行いづらいといった問題が存在する．そのため，これらのデバイスによりユーザが小さなアイコンや密集して重なり合ったアイコンからターゲットを正確に選択する事は難しく，またユーザにとって煩わしい．

本研究では，リモートポインティングと方向キー操作を組み合わせたターゲット選択手法の提案と実装を行った．この手法では，ターゲットの候補をリモートポインティングにより大まかに絞込み，その後方向キーを用いてターゲットを選択する，というアプローチを取る．また非形式的な実験を行い，この手法が小さいアイコン，密集して重なり合ったアイコンの選択における時間の短縮と精度の向上に有用であるか検証を行った．

目 次

図 目 次

第1章　背景と問題点

1.1　研究の背景

大型液晶ディスプレイや大型プラズマディスプレイ，プロジェクタ等の大画面が安価になり，これらは様々な場所に広く普及している．家庭では，図 1.1 に示す AQUOS [1] のような，薄型，大型，高解像度な液晶テレビや，大型プラズマテレビが設置されている事が多い．これらの大型テレビは，従来のようにテレビ番組やビデオ，DVD の視聴に利用される他，近年では PC のディスプレイとしても利用されるようになった．大型テレビ上において，Web やフォトアルバムの閲覧等を家族で行うといった楽しみ方が一般的となりつつある． テレビ用 Web インタフェースには，10 フィート UI と呼ばれるインタフェースの概念が取り入れられ，リモートコントローラ（リモコン）のボタンを用いて Web 閲覧の操作が行われる．

図 1.1: 室内に設置された大型液晶テレビ

しかし，10 フィート UI には，以下に挙げる問題が存在する．

- 大量のメニューの中からターゲットを選択する事に時間がかかり，操作回数が増加する
- メニューが整列されている（又は番号等により順序付けられている）必要があるため，レイアウトに制限が加わる

[1] http://www.sharp.co.jp/aquos/

このため 10 フィート UI は，フォトアルバムのサムネイルによる写真一覧の閲覧機能等の，大量にオブジェクトが存在するアプリケーションや，都市や店舗等を表すアイコンが散在しているような地図アプリケーション等の，位置情報に意味を持つアプリケーションには向かない．ターゲットの選択にかかる時間やレイアウトの自由度の観点から，これらのようなアプリケーションには，ポインティングデバイスを用いた位置の指示による直接選択が望ましい．

大画面向けポインティングデバイスとして，ジャイロマウス，レーザポインタ等，ユーザが画面から離れてポインティングを行うためのデバイス（以降リモートポインティングデバイスと呼ぶ）が利用される．図 1.2 にリモートポインティングのイメージを示す．ユーザはリモートポインティングデバイスを手に持ち，空中で操作を行う事によってポインティングを行う．

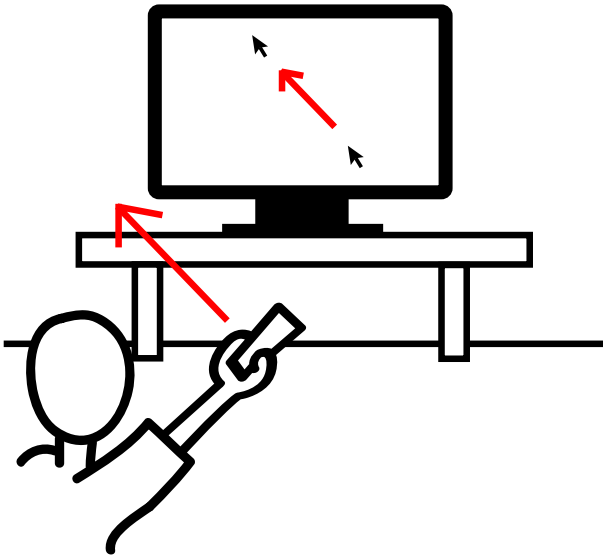

図 1.2: リモートポインティングのイメージ

しかしこれらのデバイスには，以下に挙げる問題が存在する．

- ユーザがデバイスのボタンを押した時に，デバイスがぶれる
- ポイント位置を一カ所に止めておく事が難しい
- 細かい操作を行う事が難しい

また，地図のように，位置に意味を持つレイアウトには，以下に挙げる問題が存在する．

- 地図上に小さなアイコンが密集し重なりあって表示される事が多い

従って，リモートポインティングデバイスによって，そのようなアイコンの選択を正確に行う事が難しい．

1.2 研究の目的

本研究では，大画面上に表示された，多数の小さく折り重なって表示されたアイコンの中から，素早く，正確に目的のアイコンを選択するための手法を提案する．複数のデバイスを比較し，大画面上のアイコンを選択する事に適したデバイスを考察する．そして，そのデバイスの特徴を活かした選択手法を設計し，実装する．

1.3 10 フィート UI とその問題点

10 フィート UI とは，ユーザが 10 フィート (約 3m) 画面から離れて利用する事を前提とした，文字やボタン等が大きく設計されたユーザインタフェースの概念である．図 1.3 に AQUOS の 10 フィート UI を例として示す．10 フィート UI は，リモコンによる操作に適したユーザインタフェースの概念として提唱された．10 フィート UI は，Windows XP Media Center Edition[2]や XMB[3]にも採用され，大画面の普及と共に徐々に家庭に広がりつつある．

図 1.3: リモコンによる 10 フィート UI の操作例

10 フィート UI は，リモコンのボタンによる操作が前提となっているユーザインタフェースの概念である．10 フィート UI には，2 つの問題が存在する．1 つ目は，メニューのターゲットである項目を選択するまでに時間がかかり，操作回数が増加するという問題である．この問題のため，ボタンによるメニューの項目選択を基本としたインタフェースは，大量の項目の中から目的の項目を選択する事を苦手とする．2 つ目は，メニューは，整列される（又は順序づけられる）必要があり，レイアウトに制限が加わる，という問題である．10 フィート UI を取り入れたユーザインタフェースは，メニューは階層的になるように設計される．つまり，

[2]http://www.microsoft.com/japan/windowsxp/mediacenter/default.mspx
[3]http://manual.playstation.net/document/jp/ps3/current/basicoperations/xmb.html

ボタンによるメニューの項目選択を採用すると，メニューの構成やレイアウトに制限が加わるのである．

図 1.4 は，図 1.5 のような構造を持つ，全国主要都市の天気情報閲覧用 10 フィート UI の例である．これらの図を用いて，現在閲覧しているメニューとは別のメニューにあるサブメニューをユーザが選択したい時について，前述の 2 つの問題の例として説明する．天気情報の構造は，図 1.5 に示すような階層的な構造により設計される．ここで，土浦の天気を知りたい時に，ユーザが行うメニューの移動について，図 1.5a に示す．全国というメニューでは，ユーザは各地方を選択する．ユーザが関東地方を選択すると，閲覧中のメニューが全国から関東地方に移る．関東地方にはサブメニューとして各県があり，ユーザは関東地方の各県を選択する．この中からユーザが茨城県を選択すると，閲覧中のメニューが茨城県に移る．同様にして，ユーザは土浦を選択して，土浦の天気を閲覧する．もし，ユーザが土浦の天気を閲覧した後，他地方の都市である姫路の天気を閲覧したくなった場合，図 1.5b に示すように，ユーザはまず全国のメニューに戻ってから，近畿地方を選択し，兵庫県を選択し，姫路を選択する必要がある．

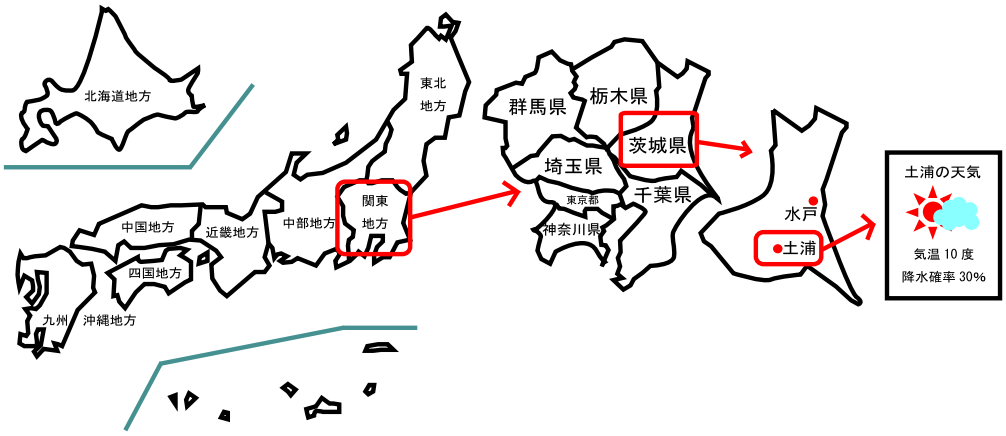

図 1.4: ボタン操作による天気情報の選択例

このように，階層構造をとるユーザインタフェースでは，ユーザに要求される操作の回数が多くなってしまう．また，あるメニューに存在するサブメニューは，ユーザがそのメニューを選択するまで表示されない．このため，全メニューのうち一部しか画面に表示出来なくなる．このため，10 フィート UI を取り入れたユーザインタフェースでは，たくさんのメニューを画面に表示する事が出来ない．地図では位置情報が重要になるため，この事は問題である．

上記の問題の解決策は，すべてのメニューを一度に表示し，直接選択する事である．ユーザに要求される操作の回数は，メニューを直接選択する方が，階層的なメニューにおける選択よりも少なくなる．また，地図では，都市を直接選択する方法により，ユーザは全国の各都市を一度に見渡す事が可能になり，レイアウト上の制限がなくなる．地図上の都市の直接選択は，その位置を指示するための手法により実現可能になる．

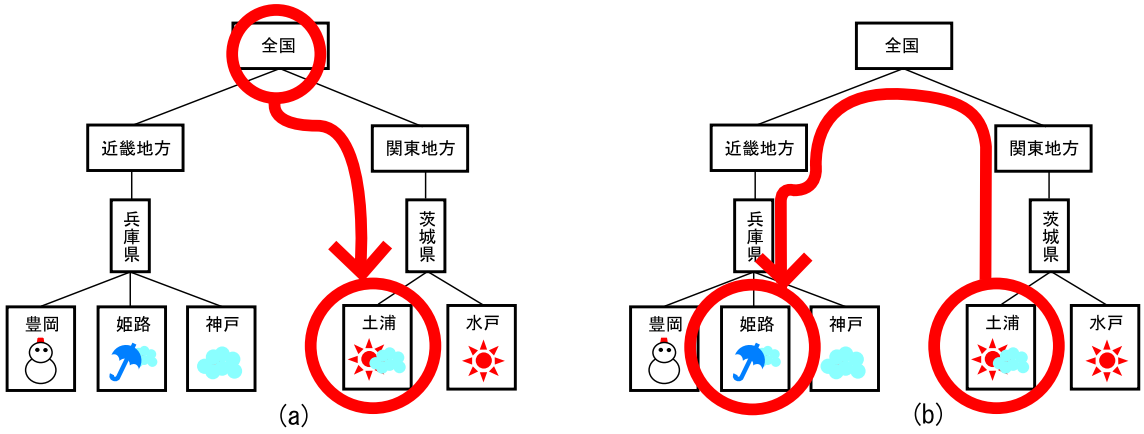

図 1.5: 別のサブメニューにある情報の閲覧操作例

1.4 大画面とポインティングデバイス

画面の任意の場所を指示するための入力デバイスとして，ポインティングデバイスを挙げる事が出来る．

本節では，ポインティングデバイスの分類と，大画面向けポインティングデバイスについての考察，並びに大画面向けポインティングデバイスの問題点について述べる．

1.4.1 ポインティングデバイスの分類

絶対ポインティングと相対ポインティング

絶対ポインティングとは，ポイント位置を指定する事によってポインティングを行う事である．絶対ポインティング用デバイスとして，スタイラスペン，指等が挙げられる．例えば，スタイラスペンや指の先をタッチパネルに接触させると，その接触した位置がポイント位置となる．

相対ポインティングとは，デバイスを動かす事によるデバイスの移動量を入力とし，現在のポイント位置を移動量に応じて移動させてポインティングを行う事である．一般的な相対ポインティング用デバイスの例として，マウス，トラックパッド，トラックボール等が挙げられる．例えばマウスの場合，マウスをユーザが動かすと，現在のポイント位置が，ユーザがマウスを動かした方向と距離に応じて動く．

連続的ポインティングデバイスと離散的ポインティングデバイス

スタイラスペン，マウス等，ユーザがデバイスそのものを動かす事によってポインティングを行うデバイスを，本論文では連続的ポインティングデバイスと定義する．連続的ポインティングデバイスでは，ユーザはデバイスの位置や動かし方によって，それに応じたポイン

ティングを行う事が出来る．一方ボタン入力のみにより操作を行うデバイスでは，ユーザによるボタンの押下に応じてポイント位置を移動させる事が出来る．このようなポインティングデバイスを本論文では離散的ポインティングデバイスと呼ぶ．

1.4.2 リモートポインティングデバイスとその問題点

大画面は，画面の物理的サイズが大きいため，離れた所からの利用に適している．このため，画面に手が届かない場所からポインティングを行える事が望まれる．また，大画面を利用する状況では，机のような，デバイスを置くための水平な台をいつでも利用出来るとは限らない．すなわち，いつでもマウスを利用出来るとは限らない．そのため，画面から離れた所より，ユーザが手に持って空中でポインティングを行うのに適したデバイスである，リモートポインティングデバイスが研究及び開発されている．

以下では，前節において述べた分類を踏まえた上で，複数の大画面向けリモートポインティングデバイスについて，その特徴と問題点について考察を行う．

レーザポインタ

Dan R. Olsen らは [3] において，レーザポインタを用いた大画面向けリモートポインティング手法を提唱した．この手法では，画面に当たったレーザスポットの位置を，そのままポイント位置とする手法である．レーザポインタにより，ポイント位置そのものを指定するため，この手法は絶対ポインティングであり，また連続的ポインティングであると言える．レーザは，直進性に優れており，レーザポインタが少しでも動くと，画面に当たっているレーザスポットも同様に動く．そのため，手ぶれの影響を大変受けやすく，またユーザにとって細かいポインティング操作が難しいデバイスである．

この手法でのクリックは，ユーザがレーザスポットを動かさずに，クリックしたい所に留めておく事により行う．従って，手ぶれの影響を受けやすいレーザポインタでは，ユーザの目的の位置において正確にクリックを行う事が困難である．

Brad A. Myers らは [2] において，レーザポインタによるポインティングでの，画面からの距離と手ぶれとの関係について述べた．彼らは，ユーザが画面から 10 フィート (3m) 離れ，画面上の目的の位置にレーザスポットを止めておく時，平均約 0.3 インチの範囲でレーザスポットが震える事を述べた．

ジャイロマウス

図 1.6 に示すジャイロマウスは，別名エアーマウスとも呼ばれ，ユーザが手に持って空中でデバイスを動かす事により，ポインティングを行うデバイスである．ジャイロマウスは，ジャイロセンサにより角速度を算出し，得られた角速度に応じてポイント位置を移動させる，相対かつ連続的ポインティングデバイスである．

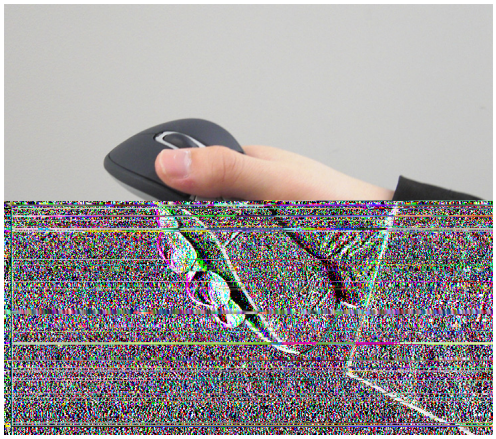

図 1.6: ジャイロマウスの例

図 1.7: リモートコントローラの例

図 1.8: 4 方向キーの例

ユーザが自身の意図通りにポインティングを行うための適切な角速度でジャイロマウスを動かす事は難しいため，細かい操作を行いづらく，ユーザが目的の位置にポイント位置を正確に移動させる事が難しい．また，ジャイロマウスのボタンをクリックする際，ボタンを押し込んだ時にデバイスが大きく動いてしまう．従って，ジャイロマウスもレーザポインタと同様に，ユーザの目的の位置において正確にクリックを行う事が難しいデバイスである．

リモートコントローラ (リモコン)

図 1.7 に示すリモコンは，メニューの項目を選択するために用いる以外に，ポインティングに用いる事が出来る．ユーザがリモコンの方向キーを押し下げると，その方向へポイント位置を移動させる事が出来る．このためリモコンは，相対かつ離散的ポインティングデバイスに分類する事が出来る．ポインティングに，デバイスの移動量や位置等の物理量を利用しないため，リモコンは手ぶれの影響を受けないポインティングデバイスである．そのため，リモ

図 1.9: 密集し選択しづらいアイコンの例

コンを利用する事により，ユーザは画面の任意の位置を正確にポイントする事が出来る．Sooらは [12] において，リモコンの方向キーを用いたポインティング手法について述べた．しかし，リモコンでは，ユーザはポイント位置の移動が一操作につき一定であるため，ユーザはポイント位置の移動のために何度も操作を行わなければならない．また，リモコンでは，ポイント位置の移動方向も制限される．例えば，リモコンの方向キーが，図 1.8 に示すような 4 方向キーの場合は，ユーザは上下左右方向にしかポイント位置を移動させる事が出来ない．このため，現在のポイント位置から離れた場所をポイントする場合，リモコンを用いると大変時間がかかる．また，ユーザが斜め方向にポイント位置を移動させたい場合，縦方向の移動と横方向の移動を組み合わせてポイント位置を移動させなければならない．この操作はユーザにとって煩わしい．

上記のように，連続的ポインティングデバイスには，ポインティング時に手ぶれの影響を受けやすいという問題や，細かい操作を行う事が難しいという問題（操作性の問題）が存在する．また，離散的ポインティングデバイスには，ポイント位置の移動に時間がかかるという問題や，操作回数の点において問題が存在する．

1.5 密集した小さなアイコンと Fitts の法則

図 1.9 は，Google マップ[4]により，つくば市中心部の飲食店を検索した結果のスクリーンショットである．地図上に，小さなアイコンが折り重なって表示されている．地図上では，こ

[4]http://maps.google.co.jp/

のように位置情報に意味を持つレイアウトが必要になる．レイアウトはそのままにこれらのアイコンを順序づけると，その順序とレイアウトの視覚的対応が取れず，方向キーによるターゲットの選択には向かない．そのため，地図上に表示されたアイコンの選択は，ポインティングにより行う事が望ましい．

しかし，これらのアイコンの中には，他のアイコンの下に位置し，ほとんど表示されていない物が存在する．手ぶれの影響や操作性の問題により，リモートポインティングにより，このようなアイコンを選択する事は難しい．

ターゲットを選択するまでのカーソル移動時間 T は，D をカーソルとターゲットとの距離，W をターゲットのサイズとすると，Fitts の法則 [6, 11] から，以下の式によって表される．

$$T = a + b\log(\frac{D}{W} + c) \tag{1.1}$$

この式 1.1 より，ターゲットのサイズが小さくなると，ターゲットの選択に要する時間が大きくなる．他のアイコンの下に位置するアイコンは，そのアイコンの本来のサイズよりも，選択出来る領域が小さくなってしまう．このため，選択にかかる時間が増大する．さらに手ぶれの影響，操作性の問題が加わるため，その時間はより顕著に増加すると考えられる．また，アイコンが密集している場合，ユーザが選択操作を誤ると，他のアイコンを誤って選択してしまう可能性が高くなる．意図通りにアイコンを選択出来ない事は，ユーザにとって煩わしい．

第2章　Rough Selecting の提案

我々は，前章で挙げた問題を解決するために，[16] において，リモートポインティングと 4 方向キー操作を組み合わせたアイコン選択手法 Rough Selecting の提案を行った．Rough Selecting は,「リモートポインティングを利用してターゲットの候補を大まかに絞込み，その後方向キーを利用してターゲットを選択する」というアプローチを取る．これにより，ユーザはアイコンの選択を確実に行う事が出来る．

2.1　リモートポインティングと方向キー操作

Rough Selecting では，リモートポインティングの速さと，方向キー操作の正確さの両方を取り入れるため，リモートポインティングと方向キー操作の両方を組み合わせる事によってアイコンの選択を行う．そのため，リモートポインティングと方向キー操作の両方を行う事が出来るデバイスが必要である．

リモートポインティングは，前章において述べたように，手ぶれの影響を受けやすく，また細かい操作が難しい．言い換えると，正確なポインティングが難しい，という事である．この問題を解決するために，Rough Selecting では，リモートポインティングの役割を，ターゲットの位置を大まかに絞り込む事とした．ポイントされた位置付近にあるアイコンをターゲットの候補（候補アイコンと定義する）とし，その中から，ターゲットを方向キー操作により選択する．

2.2　エリアカーソルによる候補アイコンの絞り込み

Rough Selecting では，大まかに候補を絞り込むために，エリアカーソル [1, 10] を利用する．図 2.1 に示すように，エリアカーソルは，円や四角等の形状をしたカーソルである．エリアカーソルを用いる事により，エリアカーソルの内部に存在するアイコンを選択する事が出来る．図 2.1 では，エリアカーソルに一部分でも含まれているアイコンが，濃い青でハイライトされている．ユーザはエリアカーソルの内部に存在するアイコンを候補アイコンとして確定する．Aileen Worden らは [1] において，Paul Kabbash らは [10] において，エリアカーソルがアイコンの選択時における選択時間のパフォーマンス向上に役立つ事，特にターゲットの大きさが小さい時に大きな効果をもたらす事を示した．この事から，折り重なったアイコンの中から，他のアイコンの下に位置し選択出来る領域が小さくなったアイコンをユーザが選択する時，エリアカーソルの利用により選択時間におけるパフォーマンス向上が期待出来る．

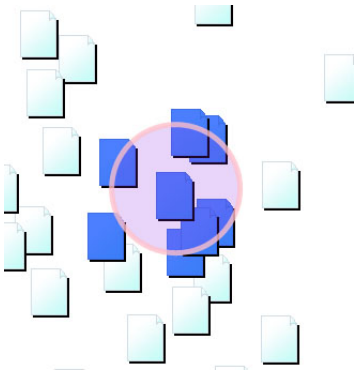

図 2.1: エリアカーソルによる候補アイコンの絞り込み

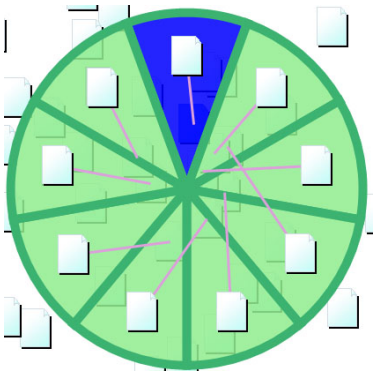

図 2.2: 候補テーブルへの候補アイコンの再配置

2.3 候補テーブルと候補の再配置

エリアカーソル内に候補アイコンが複数存在する時，ユーザはその中からターゲットを選択する必要がある．ユーザが方向キーを利用して候補アイコンからターゲットを選択出来るように，システム側が候補アイコンを順序付け，並べ替える．Rough Selecting では，一時的に並べ替えた候補アイコンを配置するため，図 2.2 に示すパイメニュー [5] 型インタフェースを導入する．これを候補テーブルと呼ぶ．ユーザがアイコンの選択操作を行うと，エリアカーソルの中心位置を中心として候補テーブルが表示される．そして，候補アイコンが候補テーブルに再配置される．

パイメニュー型インタフェースを導入する事によって，システムは候補アイコンを円周上に再配置する．このため，システムは，候補アイコンを方向キー操作に適した一次元的な順序付けを行う事が出来る．また，パイメニュー型インタフェースにおける本来のメニュー選択方法は，候補テーブルの中心から見た，カーソルの移動方向による選択方法である．パイメニュー型インタフェースでは，この選択方法により，ユーザは素早いメニュー選択を行う事が出来る．図 2.3 に，移動方向によるアイコン選択の例を示す．Rough Selecting においても，方向ボタン操作によるターゲット選択の他に，候補テーブルの中心から見たカーソルの移動方向によるターゲットの選択もサポートし，ユーザの好みに応じて使い分けられるようにする．ユーザが円状に配置されたアイコンの中からターゲット選ぶ場合，選択時間の点で有利であるため [15]，ユーザがリモートポインティングに慣れている場合は，より高速なアイコンの選択を行う事が可能になるためである．

2.4 Rough Selecting によるアイコン選択の手順

本節では Rough Selecting によるアイコン選択の手順について，図 2.4 に示す例を用いて述べる．この例では，ピンク色にハイライトされたアイコンをターゲットとする．

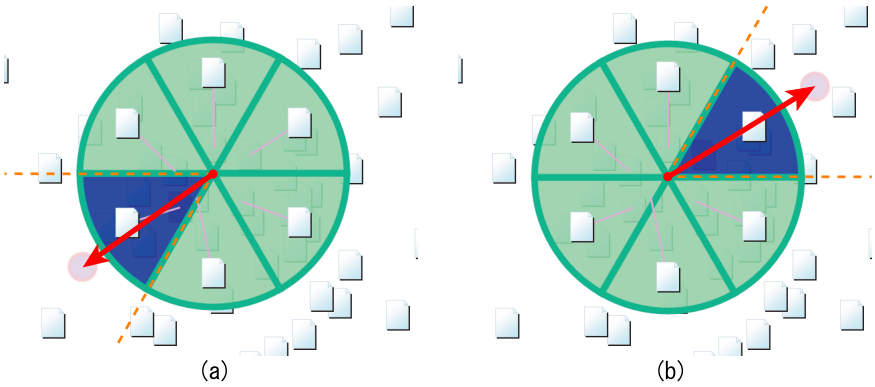

図 2.3: カーソルの移動方向を利用したターゲットの選択

まずユーザは，リモートポインティングにより，エリアカーソルをターゲットがエリアカーソルの内部に含まれるように移動させる (図 2.4a → b)．ユーザがデバイスのアイコン選択用ボタンを押し下げると，図 2.4c に示すように，候補テーブルが表示され，その上に候補アイコンが再配置される．この時，指示部分（図 2.4c, d に示す，候補テーブル上の青い部分）は 1 つの候補アイコンの下にある．次にユーザは，図 2.4d に示すように，この指示部分をデバイスの方向キーを用いてターゲットまで移動させる．この操作で用いる方向キーは，左右方向のキーのみである．ユーザが右方向のキーを押すと，図 2.4d に示すように，指示部分が右回りに進んでいく．左方向キーの場合は，指示部分は左回りに進んでいく．ターゲットまで指示部分を移動させ終わったら，ユーザは今まで押し下げていたアイコン選択用ボタンを離す．これでターゲットの選択は完了である．また，前節において述べた通り，ユーザは方向キーを用いずにリモートポインティングのみで，候補テーブルの中心から見たカーソルの方向によるターゲットの選択を行う事も出来る．

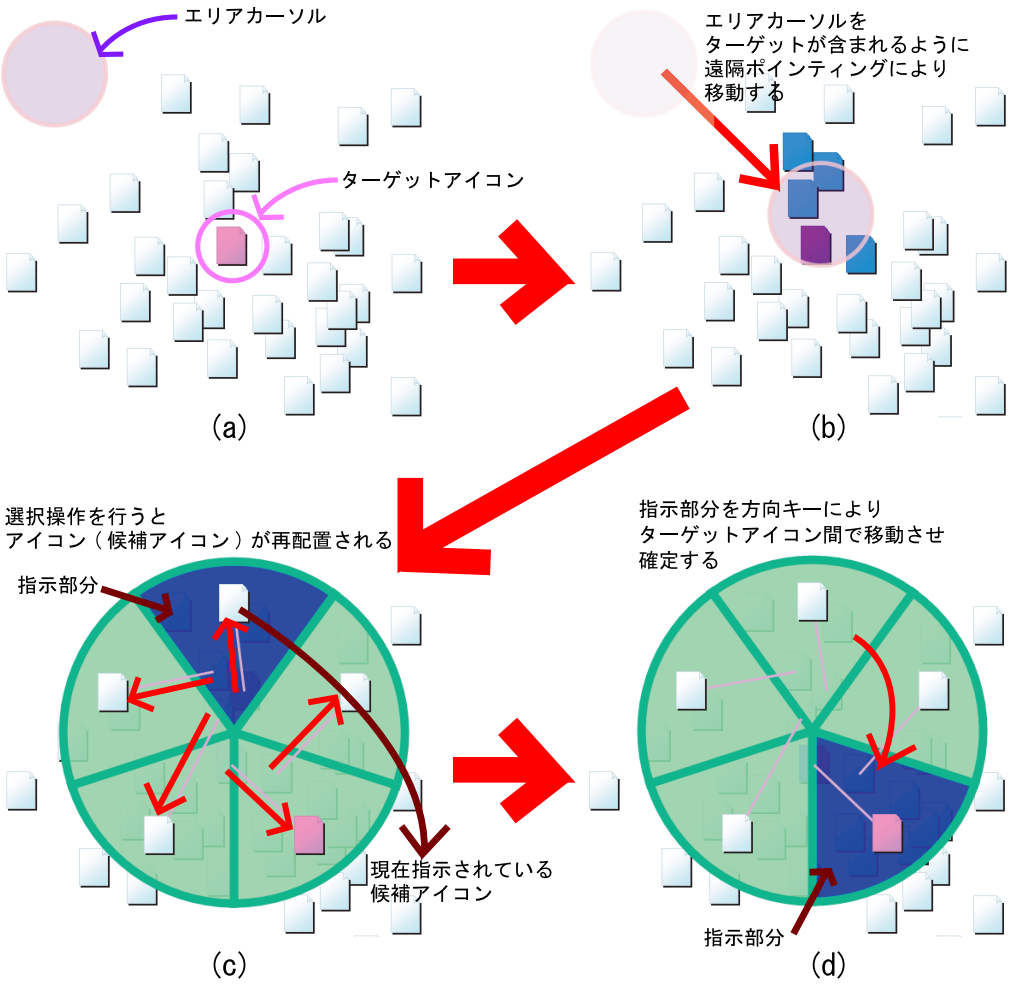

図 2.4: Rough Selecting によるアイコン選択の手順

第3章　実装

第 2 章で提案した Rough Selecting のプロトタイプシステムの実装を行った.

3.1　プロトタイプシステムの設計

3.1.1　入力デバイスに対する要求

Rough Selecting 実現のためには，1 つの入力デバイスに，以下の 4 つの事が要求される.

- リモートポインティングが可能である事
- 方向キーが搭載されている事
- アイコン選択操作機能を割り当てる事が出来るボタンが搭載されている事
- 上記の操作を同時に行う事が出来るボタン配置になっている事

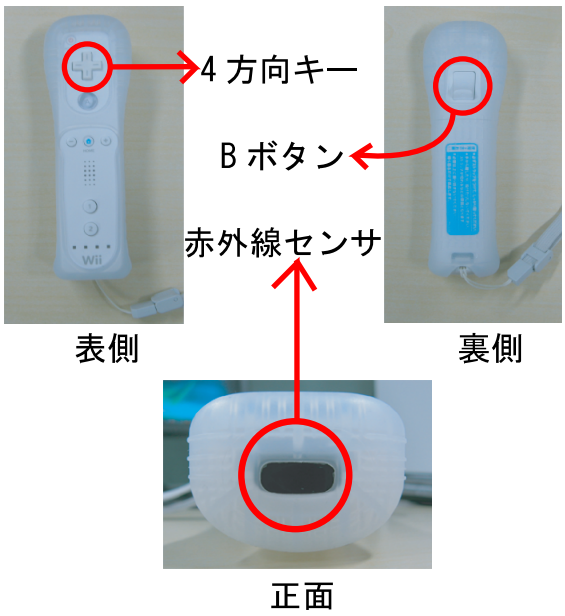

図 3.1: Wii リモコンと本システムでの利用部分

プロトタイプシステムでは，上記の要求を全て満たすデバイスとして，図 3.1 に示す Nintendo Wii リモコン[1]を採用する．Wii リモコンでは，図 3.1 に示す赤外線センサと後述のセンサーバーを用いる事によって，リモートポインティングが可能である．また Wii リモコンには表側に 4 方向キーが搭載されている．さらに，デバイスの裏側（4 方向キーの裏側）に搭載されている B ボタンをアイコン選択操作機能用のボタンとして割り当てる事により，リモートポインティングとアイコン選択操作，方向キー操作とアイコン選択操作を同時に行う事が出来る．

3.1.2 プロトタイプシステムのソフトウェア設計

Rough Selecting は，主に入力部，処理部，出力部の 3 つの部分から成る．

入力部

入力部は，入力デバイスから送信されてくる入力を解析し，イベントを発生させる部分である．ポインティングに関するイベントと，ボタンに関するイベントを発生させる．また，入力部は発生させたイベントを処理部へ送信する．

発生させるイベントは，マウスやキーボードに関するイベントとして発生させるのが適切であると考えた．入力部と処理部を分離出来るため，入力部を書き換えるだけで Wii リモコン以外の入力デバイスに対応出来るからである．

Rough Selecting の機能を実現させるために，以下で説明するイベントが必要であると考察した．

MouseMove イベント ユーザがデバイスを動かし，ポイントしている位置 (エリアカーソルの中心位置) を変化させた時に，入力部が発生させるイベントである．入力部はマウスが動かされた時に発生するイベントとして，イベント発生時のポイント位置座標と共に MouseMove イベントを処理部に送信する．

MouseDown イベント ユーザがデバイスのアイコン選択用ボタンを押し下げた時に，入力部が発生させるイベントである．入力部はマウスの左ボタンが押し下げられた時に発生するイベントとして MouseDown イベントを処理部へ送信する．

MouseUp イベント ユーザが押し下げていたリモートポインティングデバイスのアイコン選択用ボタンを離した時に，入力部が発生させるイベントである．入力部はマウスの左ボタンが離された時に発生するイベントとして MouseUp イベントを処理部へ送信する．

KeyDown イベント ユーザがリモートポインティングデバイスの方向キーを左右どちらかに押し下げた時に，入力部が発生させるイベントである．方向キーを押し下げた向きの情報を処理部に送信する．入力部は，キーボードの左向き矢印キー又は右向き矢印キーが押し下げられた時に発生するイベントとして，左右どちらのキーが押されたかという情報と共に，KeyDown イベントを処理部へ送信する．

[1] http://www.nintendo.co.jp/wii/features/wii_remote.html

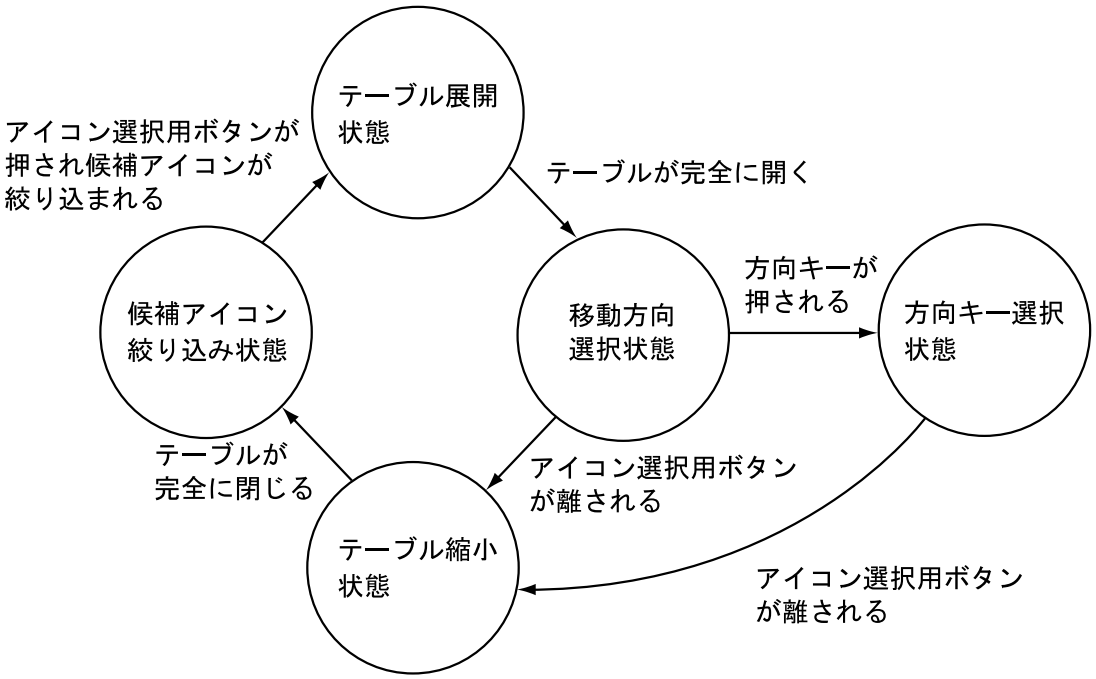

図 3.2: 処理部の状態遷移図

処理部

処理部は，入力部から送信されてきたイベントをもとに，アイコンが選択されたかどうかの判定，再配置先の計算，候補テーブルの指示部分の移動等を行う部分である．

処理部は，以下で説明する 5 つの状態に遷移する．図 3.2 に状態遷移図に示す．各状態によって，処理部が行う処理が異なる．

候補アイコン絞り込み状態 システム起動時の初期状態である．処理部の状態が候補アイコン絞り込み状態の時は，ユーザはエリアカーソルの操作とエリアカーソルによる候補アイコンの絞り込みを行う事が出来る．

処理部は，入力部から MouseDown イベントを受け取ると，その時点でエリアカーソル内に存在するアイコンを探索する (包含判定)．そして，候補テーブルの位置と，候補アイコンの再配置先を計算する．その後，処理部の状態が次に説明するテーブル展開状態へ遷移する．

テーブル展開状態 候補テーブルを展開する状態である．処理部は，候補配置テーブルの半径を段階的に大きくしていく．同時に候補アイコンを再配置先へ移動させる．候補アイコンが再配置先へ到達すると，処理部の状態は，次に説明する移動方向選択状態へ遷移する．

移動方向選択状態 再配置された候補アイコンの中から，ユーザが目的のアイコンを選択する

状態である．処理部が角度選択状態の時，ユーザはカーソルの移動方向による候補テーブルの指示部分の指定と，アイコン選択用ボタンによるアイコンの選択を行う事が出来る．処理部がこの状態の時には，ユーザは，前節において述べた，候補テーブルの中心部分から見たカーソルの方向を用いた選択によりターゲットを選択する．

方向キー選択状態 再配置された候補アイコンの中から，ユーザが目的のアイコンを選択する状態である．ユーザは方向キーによる指示部分の移動によりターゲットを選択する事が出来る点が，移動方向状態との違いである．

テーブル縮小状態 候補テーブルを縮小する状態である．処理部は，候補テーブルの半径を段階的に小さくしていく．同時に候補アイコンを元の位置へ移動させる．候補テーブルの半径が 0 になり，候補アイコンが元の位置へ到達すると，処理部の状態は，候補アイコン絞り込み状態へ遷移する．

出力部

出力部は，アイコン，カーソル，候補テーブルを描画する部分である．ユーザへ視覚的なフィードバックを与えるための処理を行う部分である．

3.2 プロトタイプシステムの作成

3.2.1 開発環境

Visual C# と Wii リモコン用ライブラリ WiimoteLib[2] を使用してプロトタイプシステムの実装を行った．また，プロトタイプシステムの実行には Microsoft .NET Framework がインストールされた Microsoft Windows XP 以上の OS が必要である．

3.2.2 ハードウェア構成

作成したプロトタイプシステムでは，ハードウェアとして PC，ディスプレイ, Wii リモコン，及び図 3.3 に示す自作センサーバーを利用した．ハードウェア構成を図 3.4 に示す．センサーバーは Nintendo Wii[3] で利用されるデバイスであり，センサーバーの左右両端に一か所ずつ 赤外線 LED が搭載されている．Wii リモコンの赤外線センサを用いてこの LED が発する赤外線からデバイスが向いている方向に関する情報を取得し，その値を用いる事によりポインティングが可能である．センサーバーを自作した理由は，Wii に用いられているセンサーバーの赤外線 LED は，輝度が低く，約 5m ユーザがセンサーバーから離れるとポインティングを行う事が難しくなるためである．プロジェクタとスクリーンを用いた大型の画面を，5m

[2]http://www.codeplex.com/WiimoteLib
[3] http://www.nintendo.co.jp/wii/

以上離れて利用する事も視野に入れたいと考えているため，より高輝度な赤外線 LED を用いてセンサーバーを自作した．Wii リモコンとセンサーバーを用いる事によって，ユーザは，リモートポインティングと方向キー入力を併用してアイコンを選択する事が出来る．

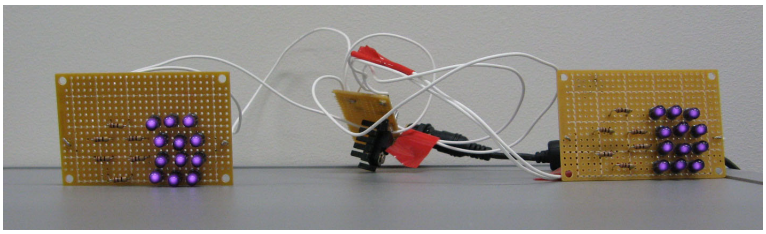

図 3.3: 自作センサーバー

PC と Wii リモコンの通信には，Bluetooth アダプタ (Planex BT-Mini2EDRW) を利用した．

3.2.3 ソフトウェア構成

プロトタイプシステムのソフトウェアは，図 3.5 に示す各部分により構成される．

Wii リモコン部 Wii リモコン部は，Bluetooth アダプタのデバイスドライバと，WiimoteLib である．Wii リモコン部は，入力部の一部を構成する．Wii リモコンの状態が変化した時 (赤外線センサの値が変化した時，ボタンが押された時，ボタンが離された時)，変化した時点での赤外線センサとボタンの状態がイベント発生部へ送られる．

イベント発生部 イベント発生部は，処理部に対してイベントを発生させる部分であり，入力部の一部を構成する．イベント発生部は，Wii リモコン部から送られてきた Wii リモコンの状態と，イベント発生部に保存してある Wii リモコンの以前の状態を比べる．そして Wii リモコンの状態の変化に応じたイベントを発生させ，処理部に対してイベントを送信する．

包含判定部 包含判定部は，アイコンの包含判定を行う部分であり，処理部の一部を構成する．包含判定部は，包含判定の結果が真であったアイコン (候補アイコン) をリストにし，リストを再配置先計算部へ送信する．

再配置先計算部 再配置先計算部は，包含判定部により得られた候補アイコンの再配置先を計算する部分であり，処理部の一部を構成する．再配置先計算部は，候補アイコンの再配置先を計算した後，候補アイコンのリストをターゲットアイコン選択部へ送信する．

ターゲットアイコン選択部 ターゲットアイコン選択部は，候補テーブル上の候補アイコンの中から，ユーザが選択したターゲットを選択状態にする部分であり，処理部の一部を構成する．

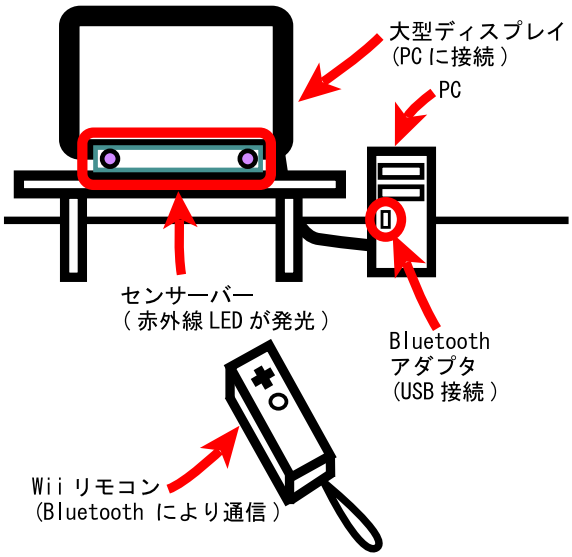

図 3.4: ハードウェア構成

表示制御部 表示制御部は，エリアカーソル，アイコン，候補テーブル，候補アイコンの描画を行う部分であり，表示部を構成する．

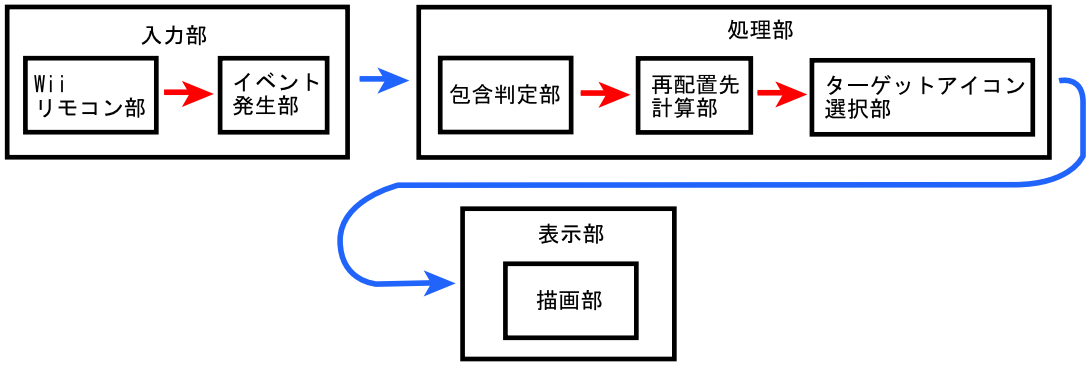

図 3.5: ソフトウェア構成

3.2.4 ソフトウェアの処理

候補アイコン再配置位置の計算

図 3.6 に示す例のような座標系における，候補アイコン再配置位置（再配置位置）の計算について述べる．ここでは，プロトタイプシステムのソフトウェアにおける，画面の左上座標 $(0,0)$ を，この座標系の原点 o とする．

まずソフトウェアは，候補テーブル（円C）の中心位置を，エリアカーソルの中心位置に設定する．この位置を，$o'(o'_x, o'_y)$ とする．エリアカーソルにより，ユーザが n 個の候補アイコンを確定したとする．この場合，ソフトウェアは再配置位置を n 個計算する必要がある．再配置位置は，$n = 1$ の時，図 3.6 に示すように，円 C の中心 o' である．また，ここで k 番目の再配置位置を $P_k(p_{kx}, p_{ky})$ とすると，$n = 2$ の時，再配置位置は，図 3.7 に示すように，線分 P_1P_2 の両端である．$n \geq 3$ の時，再配置位置は，図 3.8 に示すように，半径 r の円 C に内接した，正 n 角形の頂点である．この時，再配置先の座標は，以下の式により求められる．但し，$n \geq 1$ とする．また，$k = 1, 2, \cdots, n$ とする．

$n = 1$ の時

$$p_{kx} = o'_x \tag{3.1}$$

$$p_{ky} = o'_y \tag{3.2}$$

$n = 2$ の時

$$p_{kx} = o'_x + r(-1)^{k-1} \tag{3.3}$$

$$p_{ky} = o'_y \tag{3.4}$$

$n \geq 3$ の時

$$p_{kx} = o'_x + r\sin(\frac{2(k-1)\pi}{n}) \tag{3.5}$$

$$p_{ky} = o'_y - r\cos(\frac{2(k-1)\pi}{n}) \tag{3.6}$$

なお，$n = 1$ の時もアイコンを再配置する事にした理由は，$n \geq 2$ の時とユーザに対するフィードバックを統一するためである．$n \geq 2$ の時は，どの候補アイコンをどの再配置先へ移動させるかを決める必要がある．

まず，図 3.7, 3.8 に示すように，線分 $o'P_1$ を基準として時計回りに，o' と各候補アイコンの中心 $i_k(i_{kx}, i_{ky})$ を結ぶ線分 $o'i_k$ の角度 θ_k を求める．x_k, y_k, z_k を図 3.9 のように定めると，これらは以下の式によって計算出来る．

$$x_k = i_{kx} - o'_x \tag{3.7}$$

$$y_k = o'_y - i_{ky} \tag{3.8}$$

より，

$$z_k = \sqrt{x_k^2 + y_k^2} \tag{3.9}$$

従って，θ_k は以下の式により求める事が出来る．

$$\theta_k = \arccos(\frac{z_k^2 + y_k^2 - x_k^2}{2z_k y_k}) \tag{3.10}$$

各アイコン全ての θ_k が求まれば，これらを基準として小さい順に，各アイコンの再配置先を，P_1 から P_n まで順に割り当てていく．

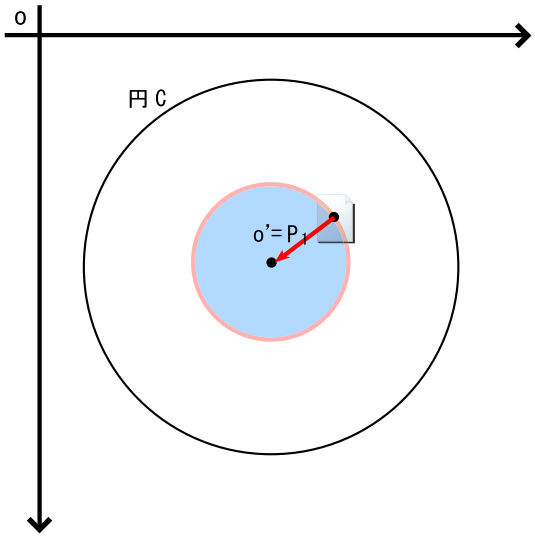

図 3.6: 候補再配置位置の計算 ($n = 1$ の時)

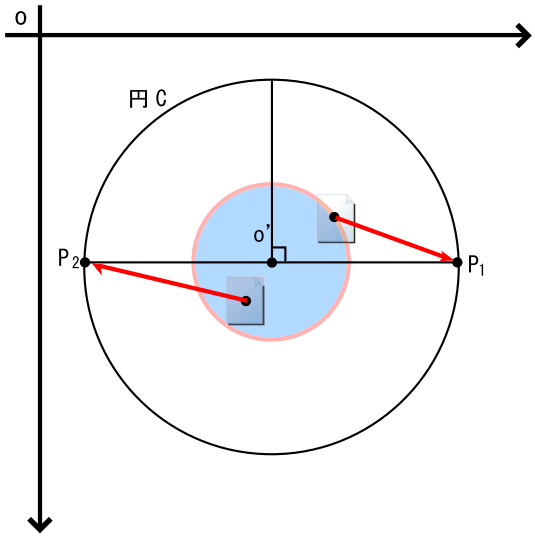

図 3.7: 候補再配置位置の計算 ($n = 2$ の時)

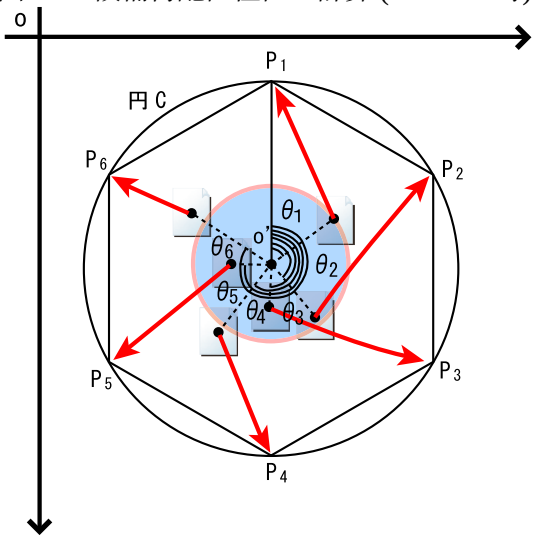

図 3.8: 候補再配置位置の計算 ($n \geq 3$ の時)

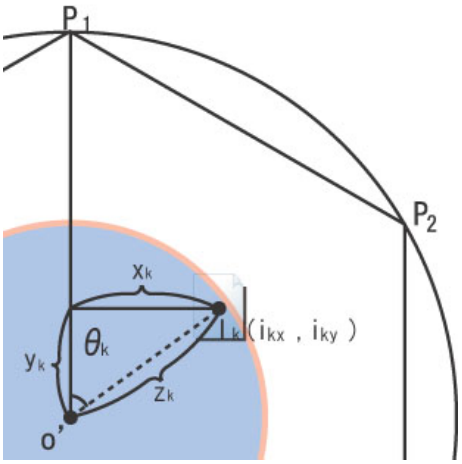

図 3.9: θ_k の計算

第4章　予備実験

Rough Selecting が，リモートポインティングによる密集し重なり合ったアイコンの選択に，選択時間やエラー率の点において有用である事を調べるために，実験を行う予定である．本格的な実験計画を立てる前に，実験の方針を立てるため，非形式的な予備実験を行った．

4.1　実験の目的

本実験では，リモートポインティングにより得られたポイント位置をそのままアイコンの選択に利用する場合と，Rough Selecting を利用する場合とを比較する．ユーザが密集し重なり合ったアイコンを選択する場合，選択にかかる時間やエラー率の点においてどちらが有用であるか調べ，今後実験計画を立てる際の参考にする事が目的である．Rough Selecting では，ポイント位置は候補を大まかに絞り込む事にしか利用しないため，上記のような場合において有益であると考えられる．

また Rough Selecting がユーザに細かい操作を要求せず，ユーザが神経を使う事なくターゲットを選択出来るかどうかを同時に調べる．

4.2　実験方法

4.2.1　被験者

被験者は，筆者自身と，同じ研究室の学生の計 2 名であった．年齢は 22 歳及び 23 歳であった．また，両人とも右利きであった．筆者はプロトタイプシステムを使い慣れていた．もう一人の学生は，プロトタイプシステムの使用経験が 1，2 度あった．

4.2.2　実験環境

本実験では，ディスプレイとして Panasonic の 50V 型プラズマディスプレイ（TH-50PH50）を使用した．また，Windows Vista Business Service Pack1 がインストールされた，Intel(R) Core(TM)2 Duo E6550(2.33GHz×2) 及び 4GB のメモリを搭載した PC を使用した．実験用プログラムは，プロトタイプシステムを改造して作成した．そのため，入力デバイスとして Wii リモコンと自作センサーバーを用いた．センサーバーの 赤外線 LED 間距離は，15cm とした．図 4.1 に実験環境のイメージを示す．

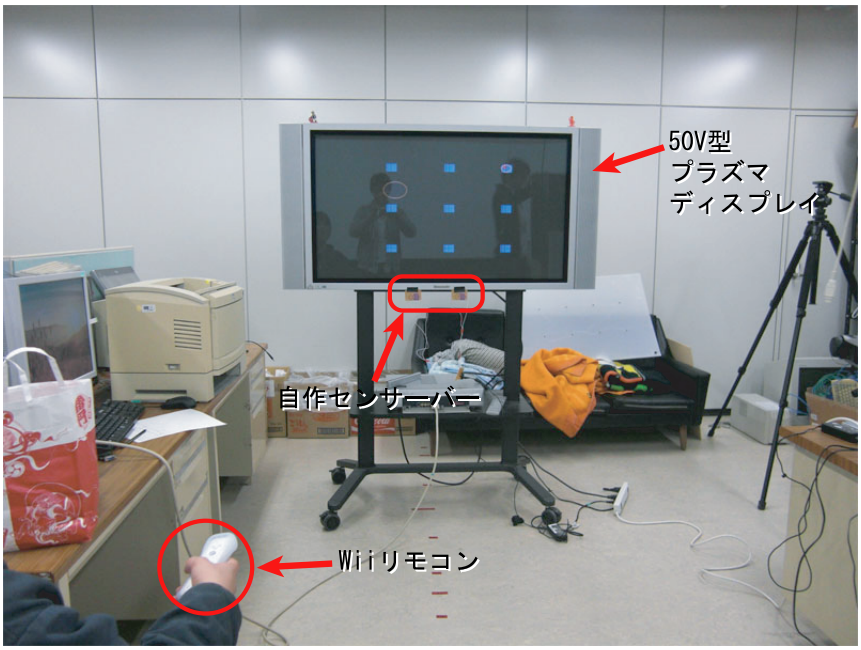

図 4.1: 実験環境のイメージ

4.2.3 実験のデザイン

図 4.5 のように，1280×1024 pixel の画面上に，32×32 pixel のアイコンが表示される．図 4.2 に示す手順によって並べられたアイコンを，重なり合ったアイコンのセット（アイコンセット）とする．アイコンセットセットが図 4.5 のように，縦横それぞれ 244 pixel 間隔で，9 セット表示される．

独立変数は 2 種類である．1 つ目は，アイコンの面積に対する，他のアイコンと重ならずに見えている面積の率 ($\frac{1}{16}$,$\frac{1}{8}$,$\frac{1}{4}$,$\frac{1}{2}$,1 の 5 通り: 図 4.3) である．これを面積率と呼ぶ．2 つ目は，アイコンの選択手法（P, AP, RS, ARS の 4 通り）である．

P(Pointer) リモートポインティングにより得られたポイント位置を，そのままアイコンの選択に利用する手法である．図 4.4 に示すマウスカーソルを被験者が絶対ポインティングを行ってターゲットの上に移動させ，アイコン選択用ボタンを押す事によってアイコンを選択する．

AP(ArrowKey_Pointer) Wii リモコンの 4 方向キーを用いた相対ポインティングにより得られたポイント位置を，そのままアイコンの選択に利用する手法である．AP は，ポインティングの手法以外は P と同様である．

RS(RoughSelecting) アイコンの選択に Rough Selecting を利用する手法である．RS では，被験者は，パイメニューにおける項目選択のように，候補テーブルからのターゲット選択

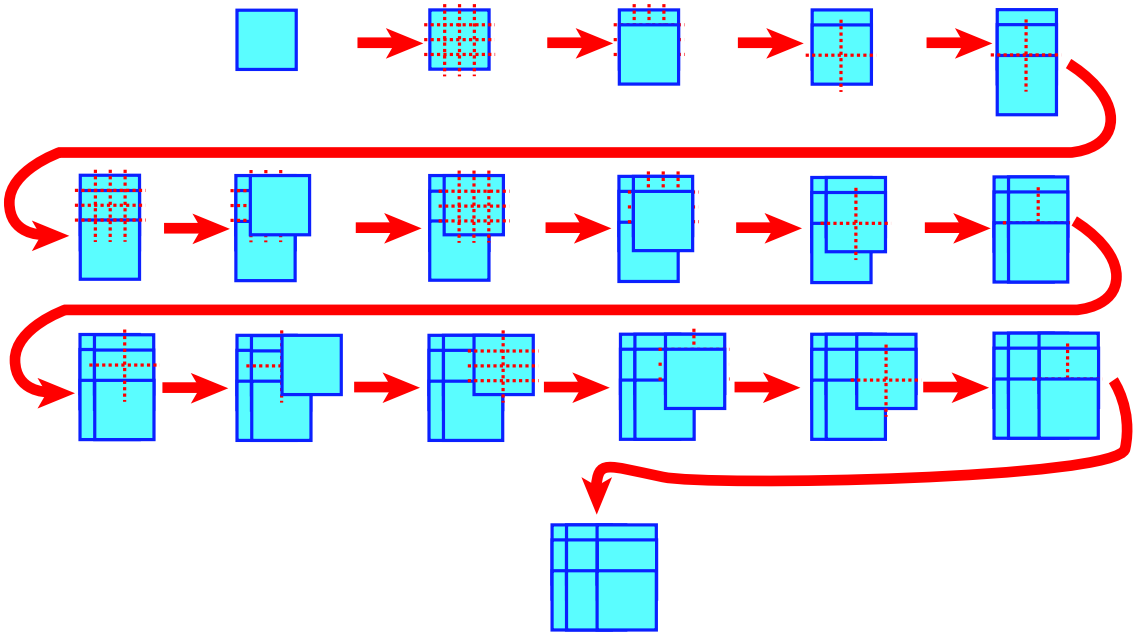

図 4.2: アイコンセットの作成の流れ

$\frac{1}{16}$	$\frac{1}{8}$	$\frac{1}{4}$
$\frac{1}{8}$	$\frac{1}{4}$	$\frac{1}{2}$
$\frac{1}{4}$	$\frac{1}{2}$	1

図 4.3: アイコンセット内の各アイコンの面積率

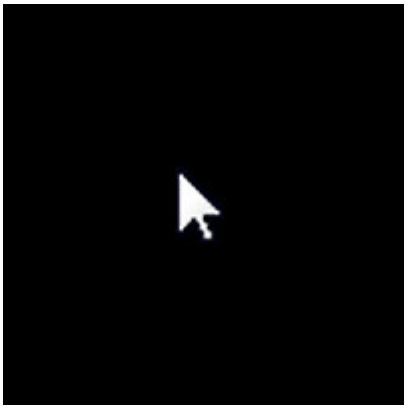

図 4.4: P, AP の手法で用いたマウスカーソル

を，候補テーブルの中心から見たカーソルの移動方向により行う．なお，エリアカーソルの半径は 50 pixel であった．

ARS(ArrowKey_RoughSelecting) アイコンの選択に Rough Selecting を利用する手法である．RS との違いは，候補テーブルからのターゲット選択に，方向キーを用いる点である．この手法でも，エリアカーソルの半径は 50 pixel であった．

被験者は，20 通り全ての条件下において，81 個のアイコンを指示された順番に 1 回ずつ選択していく．我々は，被験者がアイコンを選択するのにかかる時間と，正しくアイコンを選択出来るまでに行ったアイコン選択操作の回数を測定した．

4.2.4 実験の手順

被験者は，ディスプレイから 10 フィート離れた所に設置された椅子に，自由な姿勢で座った．デバイスの持ち方については特に指示しなかった．

図 4.6 に示すように，実験は画面中央に紫色の矩形が表示されている状態で始まる．81 個のアイコンのうち 1 つが，図 4.6 のように赤く表示され，また円で囲まれて強調される．このアイコンがターゲットである．

被験者が画面中央の矩形上にポイント位置を合わせ，そこでアイコン選択用ボタンを押すと，最初のターゲットを選択するまでの時間の測定（msec 単位で測定）が開始される．同時に，画面上から紫色の矩形が消える．被験者は，カーソルを移動し，ターゲットを選択する．被験者が正しくターゲットを選択出来れば，最初の測定が終了する．そして，他のアイコンにターゲットが移り，このアイコンを選択するまでの時間の測定が開始される．被験者は同様に選択操作を繰り返していく．被験者が 81 個のアイコンを 1 回ずつ選択し終わると，実験は完了である．ターゲットの提示される順番は，どの選択手法においても同じであった．その順番は，連続して同一のアイコンアイコンセット中のアイコンがターゲットに成らないよう制約を加え，疑似乱数により決定した．同一のアイコンセットから連続してターゲットを選ぶと，ユーザはエリアカーソルを移動させずにアイコンを選択出来てしまうからである．各手法毎に被験者は，面積率が $\frac{1}{16}$ のアイコンを計 9 回，$\frac{1}{8}$ のアイコンを計 18 回，$\frac{1}{4}$ のアイコンを計 27 回，$\frac{1}{2}$ のアイコンを計 18 回，1 のアイコンを計 9 回選択する．

実験を開始する前に，被験者は操作に慣れるまで実験の練習を行った．

また，実験中に被験者が述べたコメントや，筆者が気付いた点した．

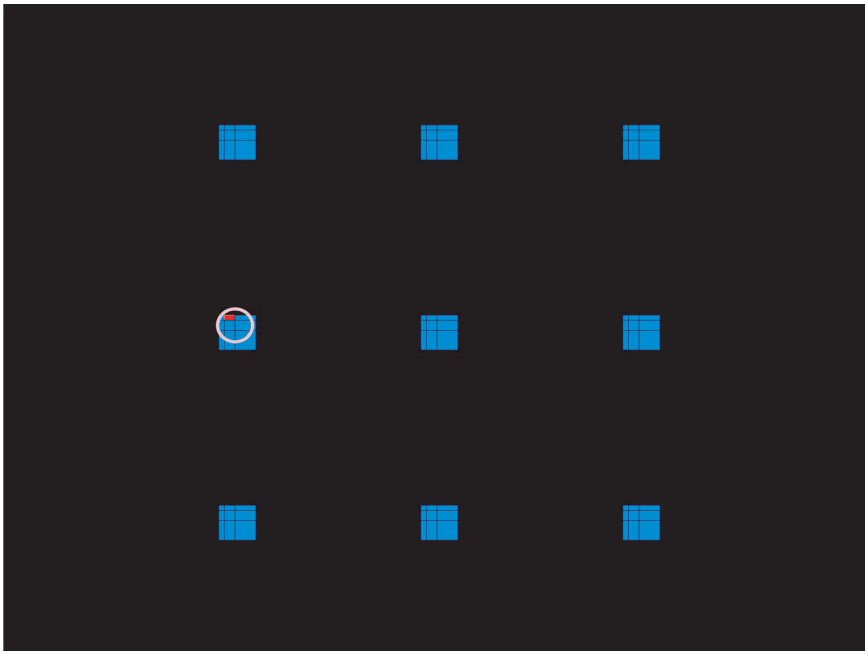

図 4.5: デザインに基づいて配置されたアイコンセット

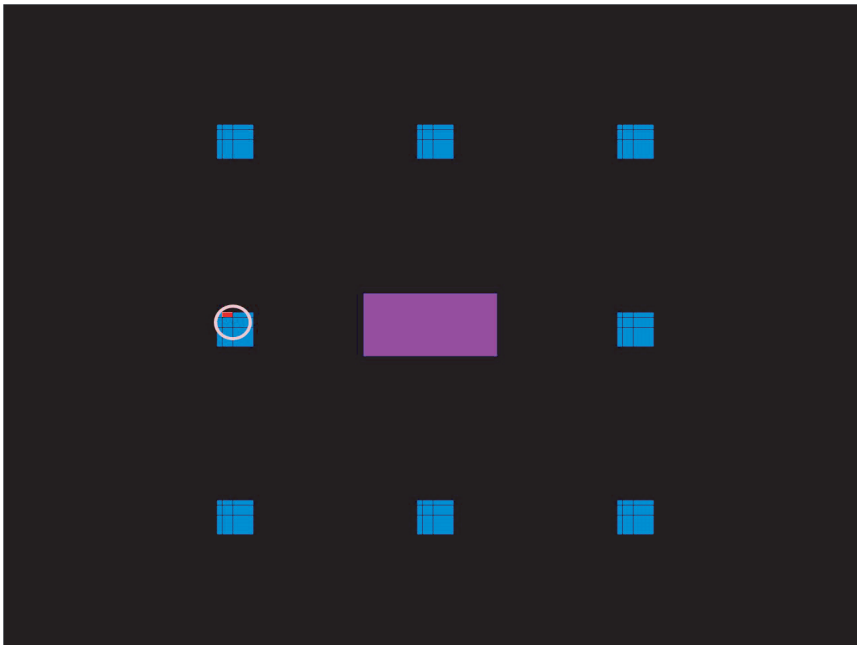

図 4.6: 測定開始前の状態

第5章　予備実験の結果と議論

実験結果を基に，平均選択時間とエラー率について分析を行い，また問題点について考察を行った．

5.1　平均選択時間

図 5.1 は，実験結果から求めた，選択手法及び面積率毎のアイコンの選択時間の平均と標準偏差である．

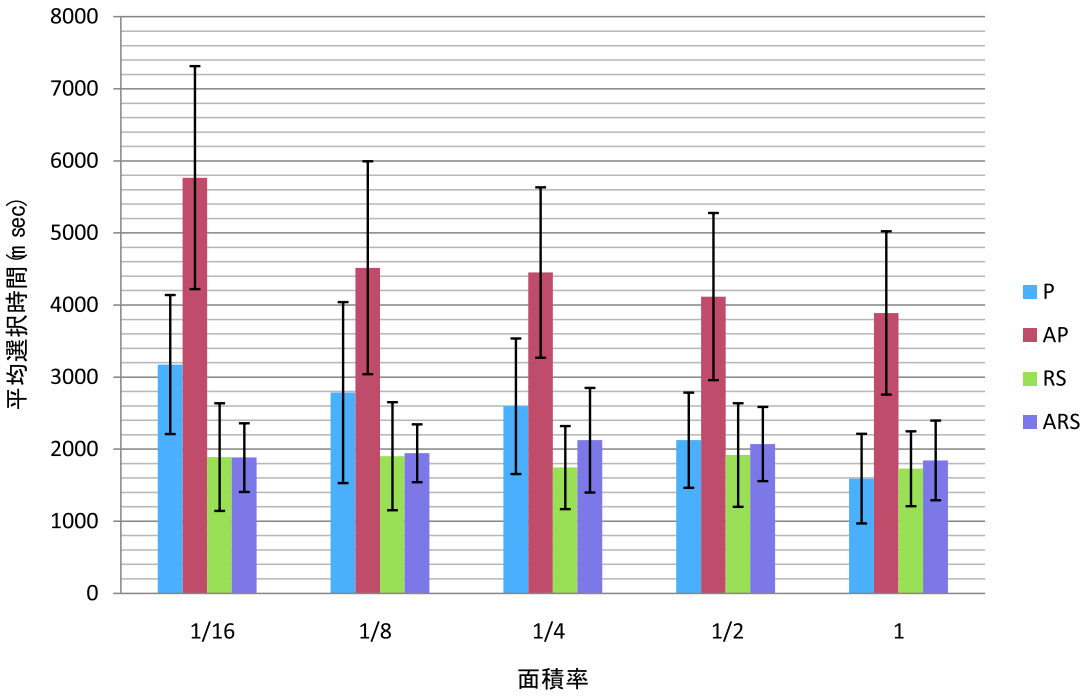

図 5.1: 選択手法及び面積率毎のアイコン選択時間の平均と標準偏差

面積率とアイコン選択手法を要因として二元配置分散分析を行った結果，面積率 ($F(4,68) = 8.32, p < 0.01$)，アイコンの選択手法 ($F(3,51) = 196.20, p < 0.01$) 及び面積率とアイコン選択手法の交互作用 ($F(12,204) = 4.06, p < 0.01$) において有意に差が見られた．アイ

コン選択手法の単純主効果を検定したところ，面積率 $\frac{1}{16}(F(3,204) = 60.84, p < 0.01)$, $\frac{1}{8}(F(3,204) = 24.13, p < 0.01)$，$\frac{1}{4}(F(3,204) = 31.45, p < 0.01)$，$\frac{1}{2}(F(3,204) = 31.52, p < 0.01)$，$1(F(3,204) = 36.18, p < 0.01)$ 全てにおいて有意に差があった． LSD 法を用いた多重比較を行った結果，面積率 $\frac{1}{16}$ では，AP は P よりも平均選択時間が有意に大きく，P は RS, ARS よりも有意に大きかった．RS, ARS の間には有意な差は見られなかった $(MSe = 989977.95, p < 0.05)$．面積率 $\frac{1}{8}(MSe = 1065700.88, p < 0.05)$,$\frac{1}{4}(MSe = 700983.78, p < 0.05)$,$\frac{1}{2}(MSe = 490709.23, p < 0.05)$,$1(MSe = 590163.2462, p < 0.05)$ では，P, RS, ARS の間には有意な差は見られず，これらと AP の間には，AP の選択時間が有意に大きかった．以上の分析結果より，RS, ARS は，面積率のどのような条件下においても，アイコン選択時間が P, AP よりも有意に小さいか，P とほぼ等しいという事が分かった．Rough Selecting は，一度候補アイコンを確定し，その中からターゲットを選択するという 2 段階の選択手法であるため，RS, ARS は，P よりもアイコンの選択に要求される操作の回数が多い．そのため当初は，面積率が大きい時 RS, ARS は P よりも平均選択時間が大きいのではないかと予測をしていた．しかしアイコンの面積率が 1.0 の時も，P と RS, ARS の間で有意な差が見られなかった事から，面積率が大きい時も Rough Selecting が利用出来る事が分かった．これは，Rough Selecting で用いているエリアカーソルの効果が大きいと考えられる．

面積率の単純主効果を検定したところ，P($F(4,68) = 10.00, p < 0.01$), AP($F(4,68) = 4.79, p < 0.01$), ARS($F(4,68) = 4.78, p < 0.01$) には有意差が見られた．しかし，RS($F(4,68) = 2.05, p < 0.1$) では有意差が見られなかった．この事から，RS では面積率とアイコン選択時間との間に関係がない事が分かった．RS では，面積率に関わらずアイコン選択時間がほぼ一定であるため，RS はアイコン選択時間の点で安定した手法であると言える．当初は，ARS も面積率の単純主効果に有意差は現れないと考えていたが，実際には現れた．予想と反した結果になった原因は，Wii リモコンの方向キーと，B ボタン（アイコン選択用ボタン）の位置関係にあると思われる．被験者から,「B ボタンを押しながらの方向キー操作は行いづらい」という意見が得られた．実験中被験者を観察していると，被験者が方向キー操作を行っている時，誤って B ボタンを離してしまう事があった．ARS の方が標準偏差が低い時と高い時があり，安定していない事も，この事が理由であると考えられる．この問題を解決するためには，デバイスのボタン配置を改良すればよいと考えられる．

P では，面積率が小さくなる程標準偏差が増加する傾向があった．この事から P は，面積率が小さくなる程，ユーザが意図通り確実にアイコンを選択出来ずに何度も選択操作を行い直す事が多くなり，確実に選択を行う事が難しくなる，という事が言える．AP では，常に標準偏差は大きかった．これは，AP では 4 方向キーによりポインティングを行ったため，ターゲットが現在のポイント位置の縦又は横方向に存在する時と，斜め方向に存在する時で，選択時間に差が出たと考えられる．

5.2 エラー率

図 5.2 は，実験結果から求めた，選択手法及び面積率毎のエラー率である．

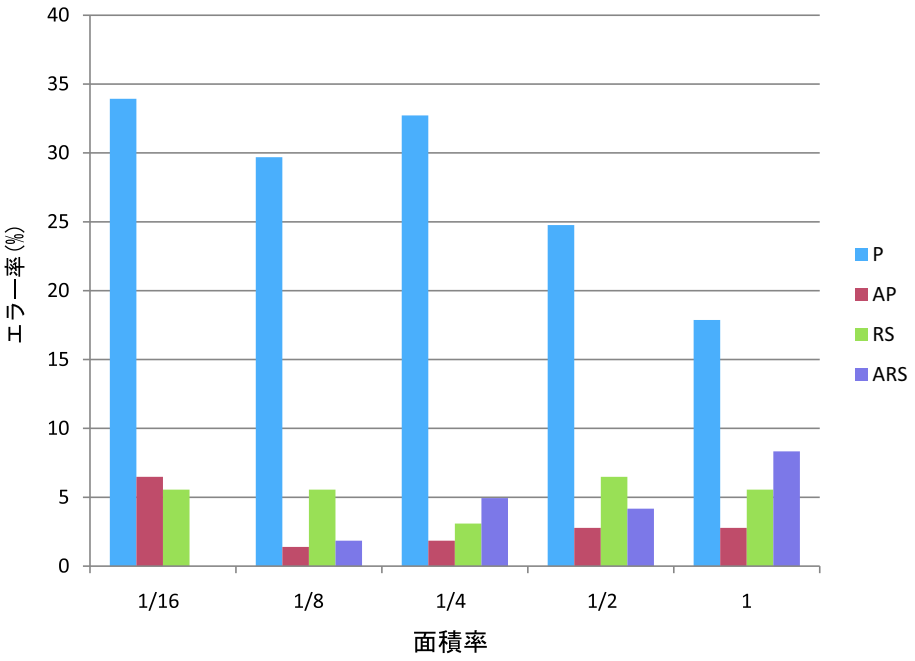

図 5.2: 選択手法及び面積率毎の平均エラー率

P のエラー率は，他の手法と比べて非常に大きかった．この事より，Rough Selecting は絶対ポインティングを利用したリモートポインティングによりポイント位置を得，その位置に存在するアイコンを選択する，という手法よりも有用である事が分かった．AP, RS, ARS では，エラー率に大きな差が見られなかった．これらの手法では，試行によってエラー率にばらつきがあった．これは，実験により得たデータ数が少なかった事が原因であると考えられる．今後本格的に実験を行う際には，より多くの被験者を募って実験を行うべきである．

5.3 被験者から得られた意見及び筆者が感じた点

以下に，被験者から得られた意見，実験中に気付いた点及び筆者が実験を行ってみて感じた点を挙げる．

- **P に関して**
  - デバイスを持つ手に疲労感を覚えた．
  - 両手を用いデバイスを支えてポインティング及び選択操作を行おうとしていた．
  - 意図通りにターゲットを選択出来るない事が多かった．
- **AP に関して**
  - ターゲットを選択するのに時間がかかり，煩わしかった．

- **RS に関して**
  - 大まかな操作によりターゲットの選択を行う事が出来た.
- **ARS に関して**
  - 候補テーブルの指示部分が下の方にある時，方向キーの左右どちらを押せば良いか分からなかった.
  - 操作に慣れればアイコンの選択にかかる時間が速くなりそうである.
- **その他**
  - 候補アイコンの数と選択時間との関係や，エリアカーソルのサイズと選択時間の関係を調べる必要がある.

以上より，Rough Selecting が，細かい操作をユーザに要求せず，ユーザが神経を使わずに小さな，又は密集したアイコンを選択するのに有益である可能性を見いだす事が出来る．今後は，本格的な実験により，この事を確かめていく.

なお，方向キーによる候補アイコンの選択は，エラー率の点や被験者の意見から考えて，改善の余地が見受けられる．今後は，方向キーによるアイコン選択手法のさらなる改良を行う必要がある.

第6章　関連研究

大画面向けターゲット選択手法において，選択時間のパフォーマンスを上げるため，以下の観点に基づく手法の研究が行われている．

- Fitts の法則に基づいてターゲットまでの距離やターゲットのサイズを短くする手法
- リモコンに付いているボタンの操作に注目した手法

6.1　ターゲットまでの距離やターゲットのサイズを短くする手法

大画面上における，指やスタイラスペンを用いた直接ポインティングでは，ユーザの腕や手を動かす量が大きくなり，ユーザの負担が増加するだけでなく，ターゲットの選択にかかる時間が増加する．また，高解像度ディスプレイにおいて，ユーザが画面の端にあるカーソルを，画面の反対側にあるターゲットへ移動させるような場合，カーソルの移動時間が増加する．このような問題を解決するため，Fitts の法則 (式 1.1) に基づき，D を減少させたり，W を増加させる事によりターゲット選択時間の減少を図る様々な研究がなされている．Tovi Grossman らは [14] において，エリアカーソルをベースにしたターゲット選択手法について述べた．この手法は，常にエリアカーソルに一番近いアイコンが選択可能となるように，システムがエリアカーソルの半径を動的に変化させ，擬似的に W を減少させる事によりターゲットの選択に要する時間の短縮を行う．小林らは [8] において，複数のカーソルを利用したターゲット選択手法について述べた．マウスの動きに追随して動く，等間隔に並べられた複数のカーソルの内，ターゲットに一番近いカーソルをユーザが見定め，それをターゲットの選択に用いる事によって，ターゲットの選択に要する時間の短縮を図るものである．この手法では D を減少させる．浅野らは [13] において，カーソルをジャンプさせる事によるターゲット選択時のカーソル移動時間短縮手法について述べた．システムが，カーソルのピーク速度を利用して，ユーザが画面のどの位置にカーソルを移動しようとしているのかを予測し，その位置へシステムがカーソルをジャンプさせる．これよってカーソルの移動距離を減少させる．Patrick Baudisc らは [9] において，カーソルの移動方向により，ユーザが選択しようとしているターゲットの候補を予測し，それらの候補をマウスカーソルの近くへ一時的に移動させる事により D を減少させる手法について述べた．

しかし，これらの手法は，リモートポインティングを利用する事が想定に入れられていない．リモートポインティングデバイスにこれらの手法をそのまま適用しても，デバイスの操作性の問題や，手ぶれの影響の問題を大きく受ける事に変わりはないものと考えられる．リ

モートポインティングを想定に入れたターゲット選択手法は数が少ないのが現状である．また，カーソルの移動方向や移動速度により，カーソルの移動先やターゲットを予測する手法は，予測に失敗する事があり，この事はユーザにとって煩わしい．

デバイスの操作性の問題や，手ぶれの影響を考慮に入れた研究として，Edward Tse らの [4] が挙げられる．Edward Tse らは，リモートポインティングによるアイコンの選択手法として，バブルカーソル [14] を用いて候補を確定し，ターゲットに付加された色情報を発声する事により選択を行う手法の開発と評価を行った．しかし，特徴となる情報をすべてのアイコンに付加しなけらばならないという問題や，ターゲットの特徴を発声する必要がある事から，利用出来る場面や場所が限られるという問題がある．これに対し我々の手法は，アイコンに特徴となる情報を付加する必要がない．また，全ての操作をユーザの手により行う．このため，大画面環境ならばどのような環境でも利用する事が出来る．

6.2 リモコンによるターゲット選択手法

前田らは [17] において，リモコンを用いたターゲット選択手法について述べた．ターゲットに,「↑」「↓」「→」「←」のような，矢印文字を付加する．選択可能なターゲットが 4 種類以上の場合は，これらの文字を組み合わせて，各ターゲットが一意に定まるように文字を付加する．ユーザがターゲットを選択するためには，ターゲットに付加されている矢印文字通りに，ユーザが方向キーを押す，という手法である．

しかしこの手法は，Web ページ上に存在するリンク等，ターゲットの数が限られた場合を対象としている．ターゲットが多数存在する場合，ユーザが行う方向キー操作の回数が多くなり，操作が煩わしくなる．

第7章　まとめと今後の課題

本研究では，リモートポインティングと方向キー操作を組み合わせた大画面向けアイコン選択手法「Rough Selecting」の提案を行った．これにより，ユーザに要求されるリモートポインティングの精度が低くなるため，特にターゲットのサイズが小さい又は密集して重なり合っている時，ターゲット選択タスクの精度が向上し，またターゲット選択時間のパフォーマンスを向上出来る．また，Wii リモコンを入力デバイスとして採用したプロトタイプシステムの設計と実装を行った．さらに，非形式的な実験を行い，パフォーマンス向上に有益である可能性を見出した．今回行った実験環境では，アイコンが小さい程，又は他のアイコンの下に位置し見えている面積が小さい程ターゲットの選択にかかる時間のパフォーマンスの点において有益である事を示した．

今後は，候補テーブルの改良を行いたいと考えている．方向キーの上下方向，斜め方向も利用したより素早く確実なターゲット選択を手法へ改良していきたい．また，今回行った実験の結果を基に，本格的な被験者実験を行い，より多くのデータを収集する．そして，本手法の有用性を確かめていく．

謝辞

本論文の執筆に当たって，指導教員である田中二郎先生，志築文太郎先生をはじめ，三末和男先生，高橋伸先生には丁寧なご指導を頂き，またご助言を頂きました．心より感謝致します．また，志築文太郎先生には日頃からきめ細かいご指導を頂き，大変お世話になりました．厚くお礼申し上げます．田中研究室のメンバーにはたくさんのアドバイスを頂き，また研究以外の部分でも大変お世話になりました．この場を借りてお礼申し上げます．

最後に，大学生活でお世話になったすべての方に感謝いたします．ありがとうございました．

参考文献

[1] Aileen Worden, Neff Walker, Krishna Bharat and Scott Hudson. Making Computers Easier for Older Adults to Use: Area Cursors and Sticky Icons, Proceedings of the 15th annual SIGCHI conference on Human factors in computing systems (CHI 1997), pp.266-271 ,1997.

[2] Brad A. Myers, Rishi Bhatnagar, Jeffrey Nichols, Choon Hong Peck, Dave Kong, Robert Miller and A. Chris Long. Interacting at a Distance: Measuring The Performance of Laser Pointers and Other Devices, Proceedings of the 20th annual SIGCHI conference on Human factors in computing systems (CHI 2002), pp.33-40, 2002.

[3] Dan R. Olsen Jr. and Travis Nielsen. Laser pointer interaction, Proceedings of the 19th annual SIGCHI conference on Human factors in computing systems (CHI 2001), pp.17-22, 2001.

[4] Edward Tse, Mark Hancock and Saul Greenberg. Speech-filtered Bubble Ray: Improving Target Acquisition on Display Walls, Proceedings of the 9th international conference on Multimodal interfaces (ICMI 2007), pp.307-314, 2007.

[5] Jack Callahan, Don Hopkins, Mark Weisert and Ben Shneiderman. An empirical comparison of pie vs. linear menus, Proceedings of the 7th annual SIGCHI conference on Human factors in computing systems (CHI 1989), pp.95-100, 1989.

[6] Johnny Accot and Shumin Zhai. Beyond Fitts' law: models for trajectory-based HCI tasks. Proceedings of the 15th annual SIGCHI conference on Human factors in computing systems (CHI 1997), pp.295-302, 1997.

[7] Martin Hachet, Joachim Pouderoux, Florence Tyndiuk and Pascal Guitton. "Jump and refine" for rapid pointing on mobile phones, Proceeding of the 25th annual SIGCHI conference on Human factors in computing systems (CHI 2007), pp.167-170, 2007.

[8] Masatomo Kobayashi and Takeo Igarashi. Ninja cursors: using multiple cursors to assist target acquisition on large screens, Proceeding of the 26th annual SIGCHI conference on Human factors in computing systems (CHI 2008), pp.949-958, 2008.

[9] Patrick Baudisch, Edward Cutrell, Dan Robbins, Mary Czerwinski, Peter Tandler, Benjamin Bederson and Alex Zierlinger. Drag-and-Pop and Drag-and-Pick: techniques for accessing remote screen content on touch- and pen-operated systems, Proceedings of Ninth IFIP TC13 International Conference on Human-Computer Interaction (Interact 2003), pp.57-64, 2003.

[10] Paul Kabbash and William Buxton. The “Prince” Technique: Fitts’ Law and Selection Using Area Cursors, Proceedings of the 13th annual SIGCHI conference on Human factors in computing systems (CHI 1995), pp.273-279, 1995.

[11] Paul M. Fitts. The Information Capacity of the Human Motor System in Controlling the Amplitude of Movement, Journal of Experimental Psychology, pp.381-391, vol.47, 1954.

[12] Soo Chul Lim, Ji Hyea Han, Min Young Jo, Eun Mi Jeon, Woo Sik Choi, Chang Geun Song, Song Yong Sim and Sung Woo Kim. Effective interaction techniques for moving cursor using a remote control, CHI ’04 extended abstracts on Human factors in computing systems, pp.1542, 2004.

[13] Takeshi Asano, Ehud Sharlin, Yoshifumi Kitamura, Kazuki Takashima and Fumio Kishino. Predictive Interaction using the Delphian Desktop, Proceedings of the 18th annual ACM symposium on User interface software and technology (UIST 2005), pp.133-141, 2005.

[14] Tovi Grossman and Ravin Balakrishnan. The Bubble Cursor: Enhancing Target Acquisition by Dynamic Resizing of The Cursor’s Activation Area, Proceedings of the 23th annual SIGCHI conference on Human factors in computing systems (CHI 2005), pp.281-290, 2005.

[15] 越澤勇太, 日浦慎作, 佐藤宏介. ポインティングによる多数項目からの選択インタフェースの設計と評価, インタラクション 2008, pp.101-108, 2008.

[16] 藤原仁貴, 志築文太郎, 田中二郎. Rough Selecting: 直接指示とボタン入力を組み合わせたアイコン選択手法, 情報処理学会 第 71 回全国大会 (2009 年 3 月発表予定), 2009.

[17] 前田篤彦, 稲垣博人, 阿部匡伸. 矢印タグを利用したテレビ用ウェブブラウザ操作方式の開発, 電子情報通信学会 2008 年総合大会, pp.264, 2008.