卒　業　研　究　報　告　書

平成22年度

研究題目

GUIアプリケーションにおける
マウス入力支援の評価

指導教員　　上　野　秀　剛　　助教

氏　　名　　Hafiyan　Prafianto

平成23年1月27日　提出

奈 良 工 業 高 等 専 門 学 校　　情 報 工 学 科

GUI アプリケーションにおけるマウス入力支援の評価

上野研究室　ハフィヤン

本研究では，マウスポインティングの効率を高めることを目的とし，従来の研究で提案された４種類の入力支援機能を操作予測と組み合わせ，被験者実験で有効性を比較する．

評価する入力支援機能は，（１）D／C比の調整，（２）Object Pointing，（３）ボタンの拡大，（４）Drag-and-Pop の４種類で，それぞれ，操作履歴に基づく操作予測と組み合わせる入力支援機能である．

被験者実験では，被験者に実験用のプログラムを用いてタスクを完了してもらう．実験用のプログラムには４種類の入力支援機能が実装されており，入力支援機能が実装されているときのポインティングにかかる平均時間とポインティングの誤差率を測定し，入力支援機能を評価する．さらに，実験終了後にアンケートで使いやすさに関する複数の項目について被験者に解答してもらう．

被験者実験の結果，ポインティングにかかる平均時間と誤差率については，入力支援機能の間に有意な差は見られなかったが，主観的な評価では，ボタンの拡大が高く評価された．

目次

1　はじめに

現在の計算機では，グラフィカルユーザインタフェース（GUI）上のポインタをマウスで操作する方法が広く用いられている．ポインタをマウスの操作で画面上の特定の位置（目標）に動かす作業をポインティングというが，このポインティングが GUI の操作のほとんどによく行われている．したがって、ポインティングの効率を高めることで，GUI の計算機の利用効率にも大きな影響を与えることができる [1].

多くの研究でポインティングの効率向上を目的とした様々な入力支援機能が提案されている [1]．これらの支援機能の多くは，画面上のボタンをより素早く，正確に押せるように支援することを目的としており，ポインタの近くにあるボタンを拡大したり，ポインタをボタンに近づけたりすることで入力を支援している．

ユーザの操作履歴から次にクリックする目標のボタンを予測し，自動で操作を行う方法や，ポインタを自動で移動させる方法などが提案されている [2]．操作予測を行うことで入力支援を行うことができるが，誤りのない確実な予測は難しいので，ごく一部の操作にしか導入されていない．

ポインティングの目標が予測できれば，その目標に入力支援を集中させれば，ポインティングの効率を向上させる可能性があると考えられる．本研究では，ユーザの操作予測と入力支援機能を組み合わせたときの，さまざまな入力支援機能のパフォーマンス評価を目的とする．

本研究では，オフィススイートや，ウェブブラウザなど多くのソフトウェアで使用されているツールバーに注目する．ツールバーのほとんどの操作は，ツールバー上のボタンをポインティングし，そのボタンをクリックすることで機能を動作させる．ツールバーの広範囲の用途をみると，ツールバーに関する研究は重要だと考えられるので，本研究ではツールバーをクリックするたびに行われているポインティングを対象として実験を行う．

2　入力支援機能

前節で述べたように，GUI では，ポインティングの効率を高めることで，計算機全体の利用率を高めることができる．そのため，現在までの研究ではマウスの入力支援機能が多く提案されている．マウスの入力支援機能の目的は，ポインティングをよりすばやく，正確に行えるようにすることである．

本研究では，現在まで提案されている多くの入力支援機能から本研究の対象であるツールバーに適用しやすいものを選び評価する．

2.1　フィッツの法則とマウスの入力支援機能

フィッツの法則によると，ポインティングの目標がポインタの現在位置より遠いほどポインティングにかかる時間が長い．そして，目標の幅が狭いほどポインティングにかかる時間が長い [3].

数学的に定義すると，ポインティングにかかる時間は Index of Difficulty I_d （目標到達の困難さ）に依存する．この I_d を与える式は，次の式 1 で計算できる [3].

$$I_d = -\log_2\left(\frac{W}{2}\frac{1}{A}\right) \text{bits / response} \tag{1}$$

ただし，W（width）は，動きの方向に測った対象物の幅であり，A（amplitude）は開始点から対象の中心までの距離である．これらの定義を図 2.1 に示す．ただし，図の右側にある灰色の四角はクリックの目標とする．

大体の入力支援機能は，目標とポインタの距離を減らすか，目標の幅を広くすることでポインティングを支援している [1].

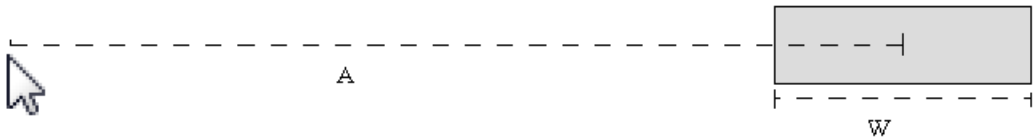


図 2.1　フィッツの法則の計算に用いられる W と A の定義

この方法を使用する例としては，たとえば，Mac OS のドックなどに実装されているのが，ポインタ付近のボタンを拡大する方法がある．図 2.2 に示すように，図の真中のポインタ付近のボタンが拡大されている．この方法では，ポインタ付近のボタンを目標のボタンだと仮定し，その目標のボタンをより簡単にクリックできるようにする．

2.2 マウスの入力支援機能

本研究では，ツールバーのボタンを対象にし関連研究で提案されており，フィッツの法則の解釈を用いる入力支援機能の評価をする．評価される入力支援機能は４つあり，それぞれについて説明する．

2.2.1 D／C比の調整

D／C比は，画面上におけるポインタの移動（Display Motion）距離と入力デバイスの制御（Control Motion）距離との比である [4]．D／C比が高いと，小さい入力デバイスの制御距離だけで画面上におけるポインタの移動距離が大きくなり，逆に，D／C比が低いと小さい入力デバイスの制御距離では画面上におけるポインタの移動距離が小さくなる．すなわち，このD／C比はポインタの相対的な速度を表しているといえる．

西口によると，D／C比を調整することでポインティングにかかる時間を短くすることができる [4]．調整方法は，目標ボタン付近でD／C比を下げる方法がある [1]．

この方法の動作を図 2.3 に示す．今,「保存」のボタンが目標のボタンであるとする．図が示すように，ポインタと「保存」のボタンが離れているときは速度を上げてポイ

図 2.2　Mac OS のドック

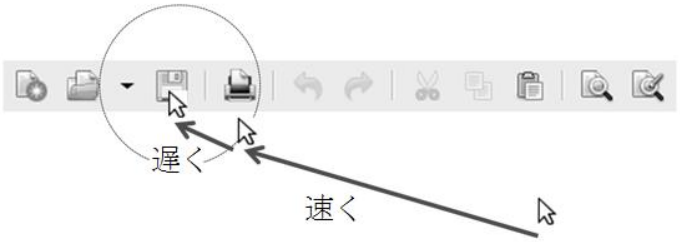


図 2.3　D／C比の調整

ンタをボタンに近づける時間を減らし，ボタン付近ではポインタの速度を下げて正確にクリックできるようにする．

この方法を用いるためには，目標のボタンをあらかじめ予測する必要があるので，予測アルゴリズムを実装する必要がある．

本研究ではこの入力支援機能を評価する．実装の詳細設定については 4.2 節で説明する．

2.2.2 Object Pointing

現在の計算機のディスプレイは，サイズが大きく，解像度が高くなりつつある [2]．そのため，ポインタをクリックの目標に移動させる時間が長くなる [6]．

Guiard らは，ポインタはクリックに反応する領域（object）だけを通るようなシステムを提案している [6]．ポインタがクリックに反応しない領域に移動すると，その反応しない領域の向こうにある，クリックに反応する領域に飛ばす．たとえば，図 2.4 に示すように，「いいえ」のボタンの上からポインタを左に動かすと，ポインタは「はい」のボタンと「いいえ」のボタンの間の領域を通らず，直接に「はい」のボタンの上に飛ぶ．

この方法ではポインティングにかかる時間を短くすることができる．しかし，ポインタがとびとびに動いているので，目がポインタを追いつけなくなるリスクがある．

本研究ではこの入力支援機能を評価する．しかし，目がポインタを追いつけなくなるリスクを考慮し，この方法を少し変換した．実装の詳細設定については 4.2 節で説明する．

2.2.3 **ボタンの拡大**

フィッツの法則によると，ボタンが大きくなるほどそのボタンにポインタを移動させるのにかかる時間が短くなる [3]．したがって，目標のボタンを拡大すれば，ポインティングにかかる時間を短くすることができる．

従来の研究では，拡大されるボタンがポインタ付近のボタンであるとする [1] が，本研究では，予測アルゴリズムを用いてあらかじめ目標のボタンを予測し，そのボタン

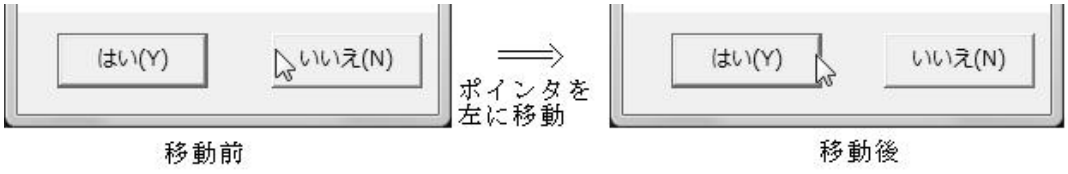


図 2.4　Object pointing でのポインタの移動

だけを拡大する．実装の詳細設定については 4.2 節で説明する．

2.2.4 Drag-and-Pop

ドラッグ・アンド・ドロップはポインティングの操作に似ている．違いは，ドラッグ・アンド・ドロップでは，目標の領域にポインタを移動させる間，マウスボタンを常に押すことである．したがって，ドラッグ・アンド・ドロップのための入力支援機能はポインティングにも用いられる．

Baudisch らは，ドラッグ・アンド・ドロップを対象とした入力支援機能 Drag-and-Pop を提案している [7]．この入力支援機能の動作を図 2.5 に示す．ユーザがマイクロソフトワードのファイルをごみ箱にドラッグ・アンド・ドロップしたいとする．ポインタをファイルの上に移動しマウスの左ボタンを押してからごみ箱にドラッグし始めると，システムがドロップ目標になりうるアイコンの proxy（代理）を作成し，ポインタの近くに置く．続いて，ユーザがごみ箱の proxy にファイルをドロップする．この操作はファイルを実際にごみ箱にドロップすることと同様にファイルを削除することになる．

この方法では，目標をポインタに近づけることで，操作にかかる時間を減らすことができる．しかし，システムがドロップ目標になりうるアイコンの予測がはずれる可能性がある．この場合，ドラッグ・アンド・ドロップを開始するアイコン付近は不要な proxy で混雑するようになる．最悪な場合，ドロップ目標のアイコンの元の位置は

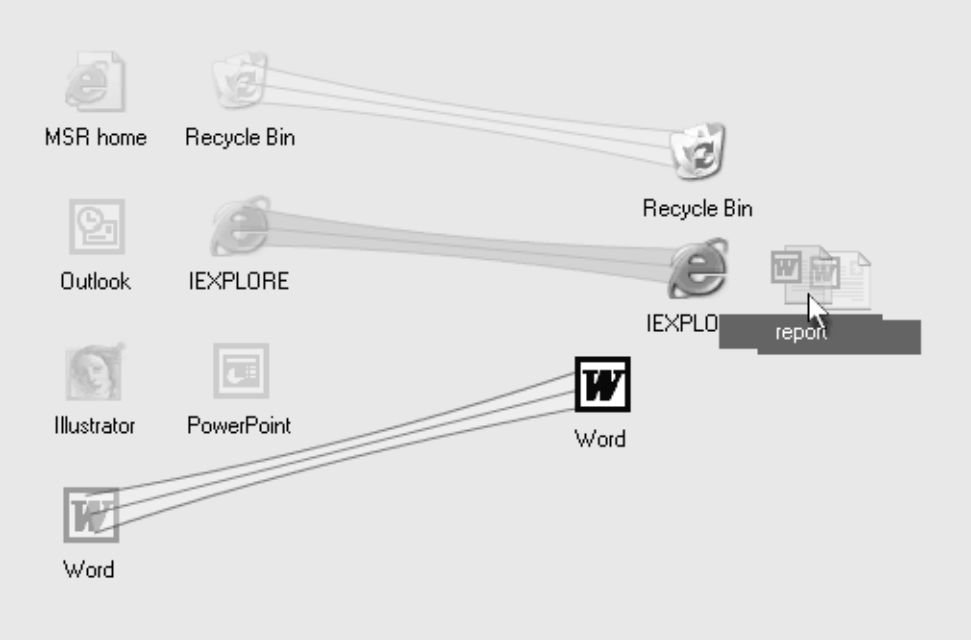


図 2.5　Drag-and-Pop の動作

proxy を置くところにあれば，不要な proxy がその目標のアイコンとかぶるリスクもある [1].

本研究ではこの入力支援機能をポインティングに適用できるような形にし，評価する．しかし，proxy をポインタに近づけすぎると，他のアイコンやボタンとかぶるリスクも考慮し，この方法を少し変換した．実装の詳細設定については 4.2 節で説明する．

3　操作予測

本研究では，2.2 節で説明したマウスの入力支援機能を操作予測と組み合わせて評価する．組み合わせ方法は，操作予測で次にユーザがクリックするボタンを予測し，そのボタンに入力支援を集中させる．

ポインタの軌道や速度などに基づく GUI 向けの操作予測のアルゴリズムは提案されているが [2]，そのアルゴリズムはかなり難しく，プログラムとして実現するにはかなりコストがかかると考えられる．一方、比較的簡単なコマンドラインインターフェイス（CLI）向けの操作予測のアルゴリズムも提案されているので、本研究ではその CLI 向けのアルゴリズムを GUI に適用できるような形に変換し用いる．この章では，そののアルゴリズムについて説明する．

3.1　操作履歴による操作予測

栢沼と萩原は，CLI 向けのユーザ操作の予測アルゴリズムを提案している [5]．この予測アルゴリズムは，次に入力されるコマンドを予測する．

提案システムの実行画面を図 3.1 に示す．通常のシェルのウィンドウの上に，提案システムの支援情報提示用ウィンドウがある．このウィンドウに，次に入力される可能性の高いコマンドが表示される．支援情報提示用のウィンドウで表示されるコマンドの最大数を**候補数**と呼ぶ．それらのコマンドは予測アルゴリズムで予測された結果で，ユーザがシェルウィンドウ上でホットキーを入力することで予測結果に基づく補完機能を呼び出すことができる．

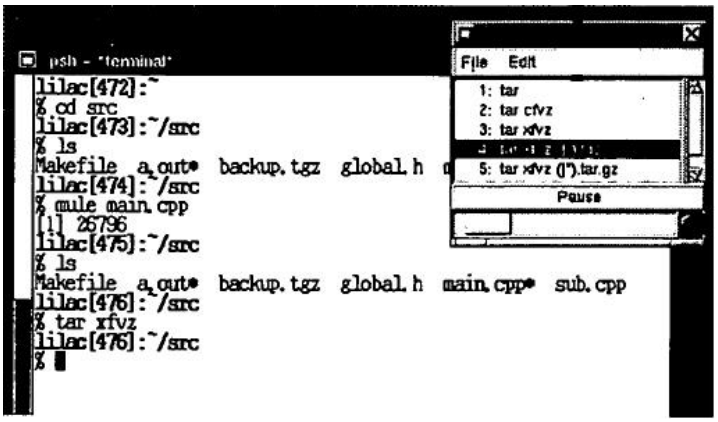


図 3.1　支援情報提示ウィンドウ

図 3.2 に，補完機能の動作を示す．ホットキーを 1 回入力することで，カーソル下に補完候補が一覧表示される．さらにホットキーを続けて入力することで，上位の補完候補から順番にコマンドラインに順番に取り込むことが可能である．

3.2 予測アルゴリズム

このアルゴリズムは，ユーザが入力したコマンドの履歴から**予測ルール**という，コマンドの入力する順を生成する．

表 3.1 に，UNIX システム上で pLaTeX のドキュメントをコンパイルして PDF ファイルを出力するまでにユーザが入力するコマンド履歴の例を示す．ここで C_{-i} は，現時点から i ステップ前に実行されたコマンドを表すものとする．

UNIX では，一つの目的を達成するために，複数のコマンドを入力する必要がよくある．たとえば，pLaTeX のドキュメントをコンパイルして pdf の結果表示するために，TEX のファイルをコンパイルするコマンド platex と，DIV ファイルの中間結果

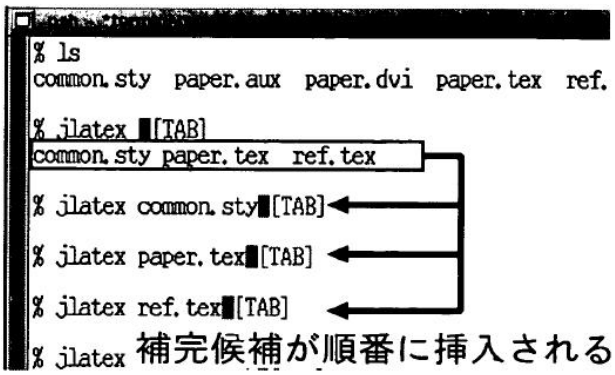


図 3.2　提案システムの入力補完システムの動作

表 3.1　サンプル操作履歴

履歴番号	コマンドライン	
C_{-7}	%platex	main.tex
C_{-6}	%dvipdfmx	main.dvi
C_{-5}	%xpdf	main.pdf
C_{-4}	%cd	..
C_{-3}	%vi	index.tex
C_{-2}	%platex	index.tex
C_{-1}	%dvipdfmx	index.dvi

を PDF に変換するコマンド dvipdfmx と，PDF ファイルを表示するコマンド xpdf の３つのコマンドが順番に入力されるというコマンド入力のパターンがある．このような入力パターンは，操作履歴で何回も現れる可能性が高い．提案されたシステムは，操作履歴から何回も現れる入力パターンを予測ルールとして抽出する．

システムが利用する予測ルールは，**前提条件**と**結果**からできている．たとえば，前の例の platex → dvipdfmx → xpdf の入力順では，platex の次に dvipdfmx が入力されると次に xpdf が入力されるといえるので，platex と dvipdfmx は予測の前提条件で，xpdf は予測の結果になる．

システムが抽出した予測ルールの例を図 3.3 に示す．この場合，システムは，（1）platex と dvipdfmx は前提条件で，予測結果は xpdf であるという予測ルールと，（2）mule と g++ は前提条件で，予測結果は a.out であるという予測ルールを抽出したことを示す．

コマンドの具体的な予測手法の例を図 3.4 に示す．アルゴリズムは操作履歴の中の

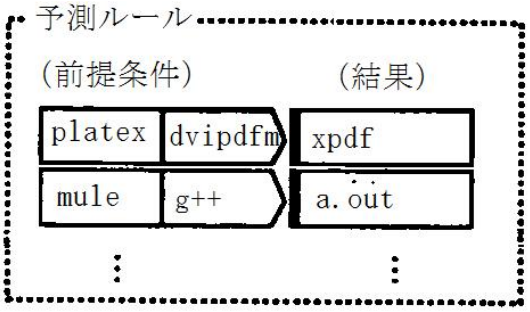


図 3.3　予測に用いる予測ルール

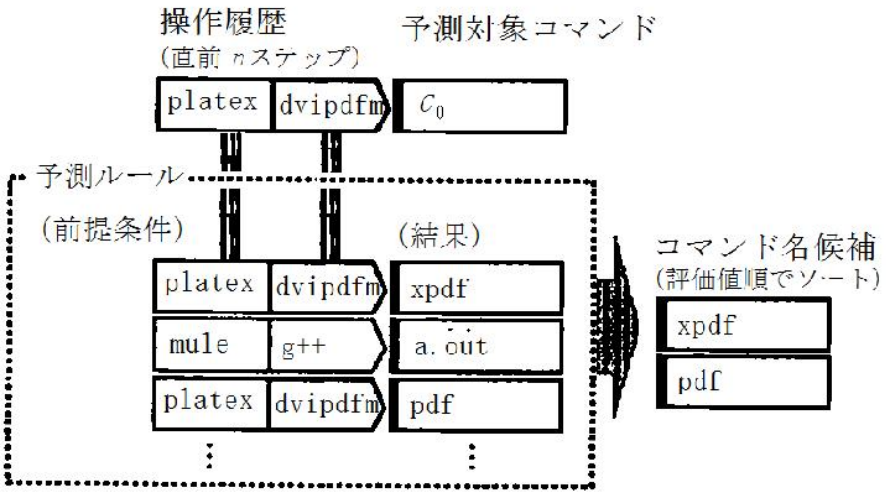


図 3.4　コマンド名の予測

直前 n ステップのコマンド系列をすべての予測ルールの前提条件と比較する．直前 n ステップのコマンド系列と予測ルールの前提条件が成り立つ予測ルールがあれば，その予測ルールの結果を予測アルゴリズムの予測結果となり，候補として出力する．ここで、n は**予測の次数**と呼ぶ．n が小さい場合、複数の前提条件が成り立つ場合がある．このとき，予測候補のコマンド名が一意に決まらない．そうしたら，複数の候補の間に，評価値を用いて順序づけを行う．予測ルールの評価値について次に説明する．

各予測ルールのデータには，固有のパラメータと表価値が格納され，新しいコマンドラインが入力されるたびに更新される．固有パラメータは評価値の計算に用いられる．固有のパラメータを表 3.2 に示す．

このアルゴリズムでは，固有パラメータから評価値 e を計算する方法を次式に定義した．

$$e = \alpha \cdot f(t-\theta) \cdot g(m) \tag{2}$$

ここで，$f(t-\theta)$ は，経過ステップ数 t に対する単調減少関数であり，このアルゴリズムでは次式のように定義した．

$$f(x) = \begin{cases} 1, & x \leq 0 \\ \frac{1}{x+1}, & x > 0 \end{cases} \tag{3}$$

$f(t-\theta)$ の項は, 時間が経過する毎に予測ルールの有効性が薄れていく．一方, $g(m)$ は，出現頻度 m に対する単調増加関数であり，このアルゴリズムでは次式のように定義した．

$$g(m) = \sqrt{m} \tag{4}$$

これらの $f(t-\theta)$ 項と $g(m)$ 項に加え，予測ルール自体の信頼度 α を考慮して評価値 e を定めることで，複数の支援情報で相対的な順位付けを行う指標として使うことができる．

この予測アルゴリズムの手順を次に説明する．

表 3.2　予測ルールの固有パラメータ

α	信頼度（=正解回数/出現回数)
θ	平均周期（=総歴総数/出現回数)
t	最後に出現してから経過したステップ数
m	コマンドの出現回数

1. 初期の予測ルール群の生成
   ファイルに保存された履歴からパターンを抽出する．
2. 1〜n ステップ目のユーザ入力
   参照すべき直前のコマンド系列が存在しないのでユーザに手作業でコマンドを入力してもらう．
3. (n+1) ステップ目以降のコマンドライン予測
   直前 n ステップのコマンドから、予測を行い，ユーザに提示する．
4. (n+1) 回目以降のユーザ入力
   ユーザが候補に含まれているコマンドを選択，もしくは入力したいコマンドがなければ手作業で入力する．
5. 評価値の計算
   直前の予測結果と実際にユーザが入力したコマンドを比較し、すべての予測ルールの固有パラメータと評価値を更新する．予測が当たったルールの評価値が増加し、それ以外のルールの評価値は不変あるいは減少する．
6. 新規の予測ルール群の獲得
   最新のコマンドラインから、新しいルールが現れたらそのルールを獲得する．
7. 予測ルール群の再構成
   ステップ６で獲得したルールを含み、更新したすべてのルールの評価値にしたがい、予測ルール群を再構成する．評価値が小さい予測ルールを削除する．
8. ステップ３へ戻る

3.3 アルゴリズムの特徴

前述のアルゴリズムを本研究で用いるに当たり，考慮すべきことがある．

- **予測アルゴリズムの正答率は候補数と入力されたコマンドラインの数に依存する**
  ユーザが候補に含まれているコマンドを選択し入力すれば予測が成功であると定義する．この場合，**予測正答率**は次式に定義される．

$$\text{予測正答率} = \frac{\text{予測が成功したコマンド数}}{\text{コマンド総数}} \tag{5}$$

  栢沼と萩原が行った評価実験をの結果，予測正答率は入力されたコマンド数が大きくなるほど，予測正答率が高くなる傾向がある．入力されたコマンド数は時間の経過で増加するので，時間の経過で予測正答率も高くなる傾向がある．そして，候補数が大きいほど，予測正答率も高くなる．

- **予測をしたり候補を表示したりするタイミングは *CLI* 向けである**
  栢沼と萩原のシステムは，コマンドラインを入力する際に，予測を行い候補を支援情報提示ウィンドウで表示する．
  本研究でこのアルゴリズムを GUI に用いると，CLI での，予測をしたり候補を表示したりするタイミングはそのまま用いることができないため，GUI でのタイミングを考慮する必要がある．
- **元に戻す（*undo*）機能があるとそのままで用いることができない**
  CLI でのコマンド入力には元に戻す機能がない．一方，一般の GUI には，元に戻す機能がある．元に戻す機能がある場合，操作履歴のデータ構造も異なる．

4　実験方法

本研究では，入力支援機能を評価するために，入力支援機能が実装されたプログラムを用いて被験者実験を行う．被験者には，実験用プログラムを用いてタスクを完了してもらうことで，実験用プログラムに実装された入力支援機能を評価することができる．

以下，実験用プログラムとそのプログラムに実装された入力支援機能と予測アルゴリズムについて説明する．次に，実験で実際に計測し評価されるものを説明する．最後に，実験の流れについて説明する．

4.1　実験用プログラム

実験のために評価対象の入力支援機能を実装したプログラムを作成する．このプログラムは，Microsoft 数式のようなプログラムで，ツールバーのボタンをクリックすることで数式を書く数式エディタである．数式エディタでは，ユーザの操作が限られており，予測が容易である．また，数式エディタでは，本研究で対象としているツールバー上の記号をクリックすることで入力するのが一般的である．

プログラムのスクリーンショットを図 4.1 に示す．ツールバーには４３個のボタンがあり，ボタンの大きさはすべて 32 × 32 ピクセルである．タスク表示部は，被験者

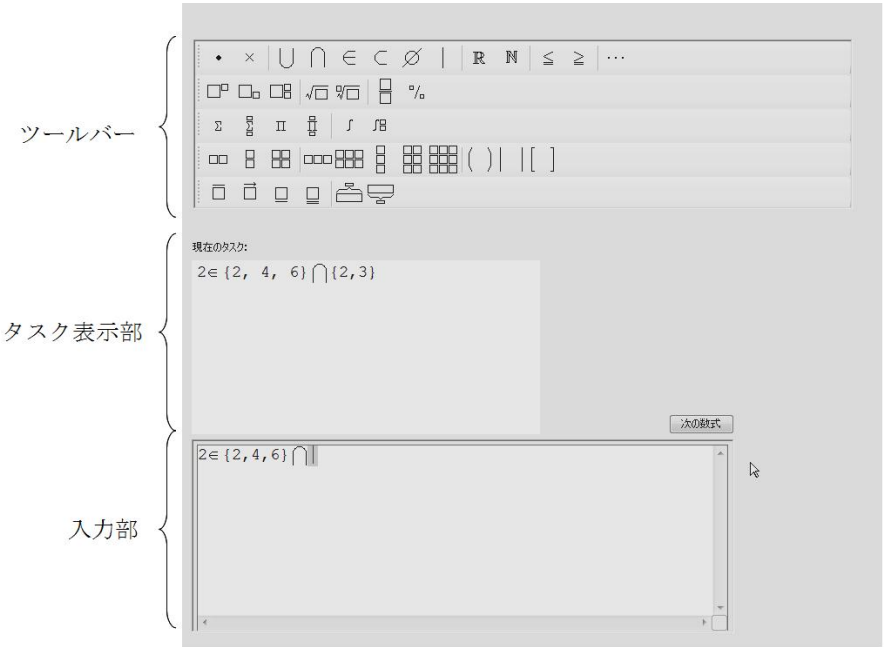


図 4.1　実験用プログラムのスクリーンショット

に書いてもらう数式を表示する．入力部では，記号の挿入位置を決めて，キーボードの入力またはツールバーのボタンをクリックすることで文字や記号を入力する．

プログラムのツールバーは，入力支援機能の実装を除いて何も工夫を入れていない普通のツールバーである．ツールバーが使いにくいほど，入力支援機能の実装の効果が現れると考えられるので，ツールバーを意図的に使いにくくする．たとえば，Microsoft 数式のような数式エディタと違って数式をグループ分けしたりしない．

このプログラムは Microsoft Visual C# 2010 Express の環境で開発された．コントロール数は９４個あり，コードは２２９８行からできている．その中に，１４０個の関数が含まれている．

プログラムの利用の簡潔な使用方法について次に説明する．

1. 挿入位置の決定
   プログラムを開始すると，まず，数式入力部分をクリックして文字や記号の挿入位置を決める必要がある．
   挿入位置を決めないと，キーボードを打つこともツールバーをクリックすることもできない．
2. キーボードで入力
   キーボードを用いて挿入位置に文字を入力する．
3. ツールバーをクリックして記号を挿入
   ツールバーをクリックすることで記号を入力する．このプログラムでは，ツールバーをクリックするとプログラムは次の挿入位置がわからなくなるので，再び挿入位置を決める必要がある．
4. 次の数式に移る
   入力部の右上にある「次の数式」のボタンをクリックすると，タスク表示部で次の数式を表示し，入力部がリセットされ，次の数式を被験者に書いてもらう．

このプログラムには「元に戻す」・「繰り返し入力」機能が実装される．「元に戻す」の機能は，キーボードの Ctrl+Z で呼び出す．「繰り返し入力」の機能は，キーボードの Ctrl+Y で呼び出す．

4.2 入力支援機能の実装

1. Slow
   この方法は，D／C比を調整することでポインティングを支援している．

図 4.2 に示すように，目標ボタンを囲める辺の長さ５０ピクセルの正四角形の領域で，D／C比をそれ以外の領域の半分にした．

ボタンがクリックされると，目標ボタン付近の領域でのD／C比をそれ以外の領域と等しくなるように上げる．

2. Blackhole

この方法は，Object Pointing の方法を少し修正した方法である．

ポインタを目標ボタンに動かすと，一瞬で目標ボタンに飛ばさず，目標ボタンに向けて秒速 1000.0 ピクセルでポインタを自動的に動かす．ただし，図 4.3 に示すように，手作業でポインタを 100 ミリ秒以上連続で目標ボタンの方向に向けて動かしてから自動的にポインタを目標まで動かす．

途中でマウスを手で動かすと，自動的な動きを止めるが，ポインタをボタンの方向に近づけると，何度でも自動的な動きが再開される．

ボタンがクリックされると，ボタンの方向にポインタを動かしても自動的に動かなくなる．

3. Big

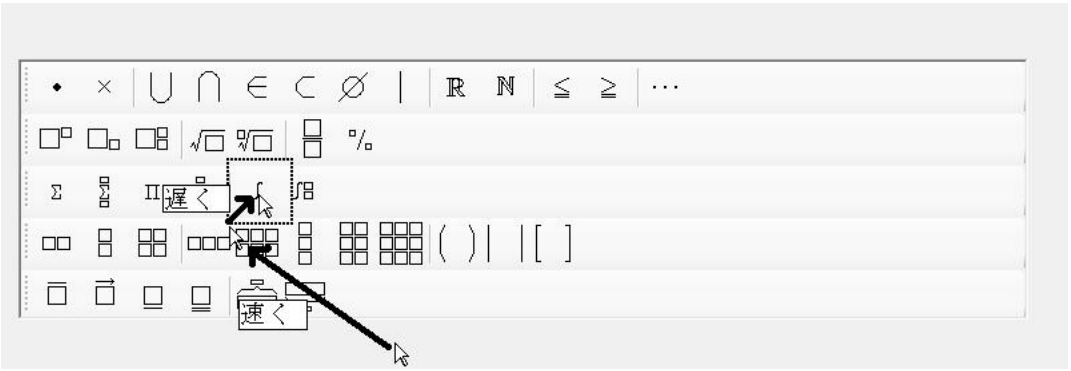


図 4.2　Slow

図 4.3　Blackhole

目標のボタンを拡大することでより簡単に目標のボタンをクリックできるようにする．

図 4.4 に示すように，ボタンを辺の長さ４８ピクセルの正四角形に拡大する．

ボタンがクリックされるとボタンの大きさを元に戻す．

4. Stalker

この方法は，Drag-and-Pop の方法を少し修正した方法であり，動作を図 4.5 に示す．

目標のボタンをポインタの近くまで移動させるが，ボタンの動きはツールバーの範囲内に限定し，ツールバー内の他のボタンとかぶらないように他のボタン

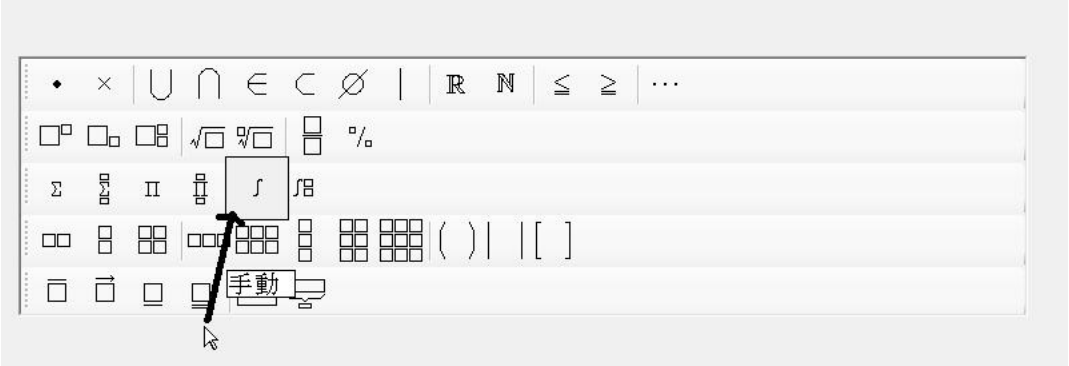


図 4.4　Big

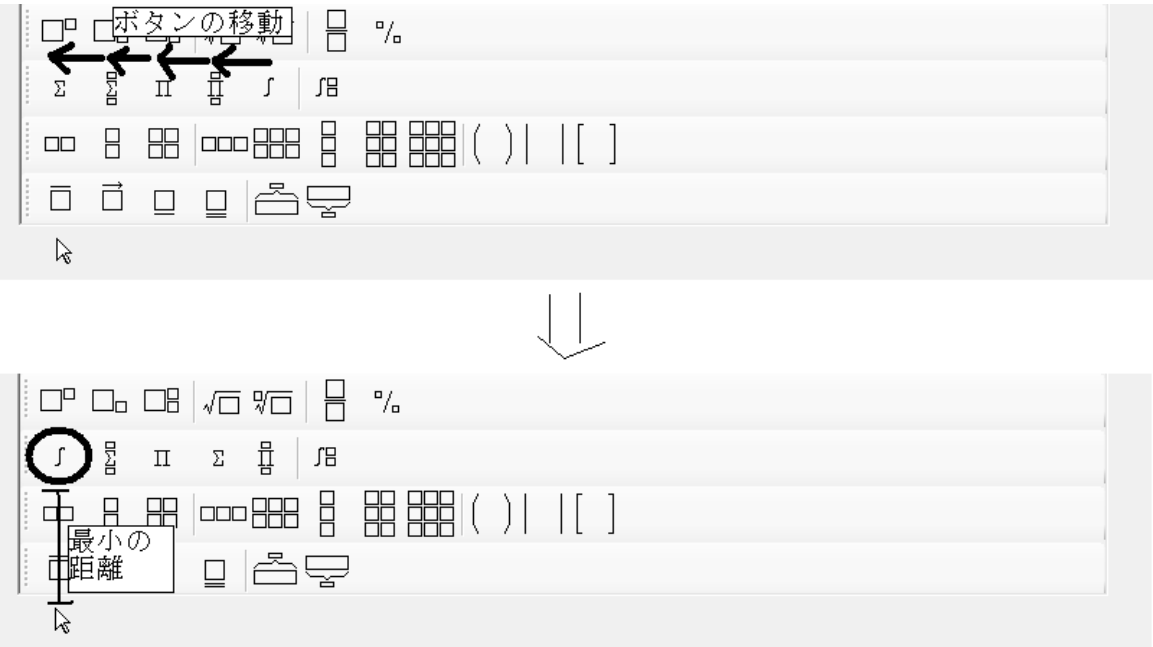


図 4.5　Stalker

も目標のボタンにスペースを譲るように動く．
図 4.5 に示すように，目標ボタンを含むツールバーは，目標ボタンとポインタとの距離を最小にするように，ツールバー範囲内のボタンの順を置き換える．ボタンの位置は 200ms ごとに更新される．
ボタンがクリックされるとすべてのボタンを元の位置に戻す．

4.3 予測アルゴリズムの実装

本研究では，3 章で説明したアルゴリズムを GUI に適用する．適用方法は，コマンド入力の変わりに，ツールバーのボタンのクリックを予測の対象とし，ユーザが次にツールバーのどのボタンをクリックするかを予測する．

予測アルゴリズムを実行するタイミングは，記号挿入位置を決定した時点である．予測が完了し目標ボタンを予測できれば，入力支援を開始する．また，実験用のプログラムでは「元に戻す」機能があるので，「元に戻す」機能があっても利用できるようにアルゴリズムを修正した．

4.4 評価対象

本研究では，入力支援機能の評価によく用いられる次の 3 つのパラメータを用いる [8]．

1. *Movement Time*
   ポインティングにかかる時間であり，短いほど素早くクリックできることを示すので望ましい．本研究では，記号の挿入位置を決定してからボタンがクリックされるまでに，ポインタの動きにかかる時間（ミリ秒）で求められる．ただし，ポインタを動かさない時間は含まない．
2. *Rate of Error*
   ユーザがタスクの完了に必要のないボタンをクリックした割合で，値が小さいほど正確にボタンがクリックできることを示すので望ましい．目標ボタン以外をクリックした回数／ボタンをクリックした回数で求められる．
3. **主観的な使いやすさ**
   実験終了後のアンケートで使いやすさに関する複数の項目について解答してもらう．被験者は，それぞれの入力支援機能に対して，４段階の評価で入力支援機能を評価してもらう．

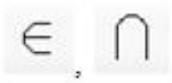
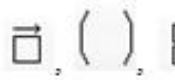
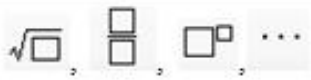
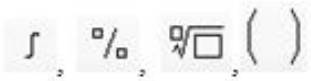
その３つのパラメータ以外に，予測の正答率とそれぞれのパラメータとの相関関係があるかどうかを調べるために，予測の正答率も計測する．

4.5　実験の流れ

本研究の実験では，１０名の被験者が協力してくれた．それぞれの被験者に６つのタスクを完了してもらう．ただし，最初のタスクは予測のための操作履歴を収集するためのもので，評価対象としない．

それぞれのタスクの内容は，８つの数式を書くことである．数式の形は表 4.1 に示すような形で，数や変数名はそれぞれのタスクごとに異なる．

表 4.1　実験のタスク

数式	押す必要のあるボタン
$2 \in \{2,4,6\} \cap \{2,3\}$	
$\{2\} \subset \{x \mid x \leqq 3, x \in \mathbb{R}\}$	
$\vec{u} = \binom{1}{0}; \vec{u} \cdot \vec{u} = 1$	
$\begin{pmatrix}2\\5\\4\end{pmatrix} \times \begin{pmatrix}1\\1\\2\end{pmatrix} = \begin{pmatrix}6\\0\\-3\end{pmatrix}$	
$\begin{vmatrix}1 & 1\\0 & 2\end{vmatrix} - \begin{vmatrix}3 & 0\\0 & 2\end{vmatrix} = -4$	
$\sqrt{1+0.2} = 1 + \frac{0.2}{2} - \frac{0.2^2}{8} + \cdots$	
$\int \sqrt[3]{u}\,du = {}^{3}/_{4}\left(u\sqrt[3]{u}\right) + C$	
$\overline{X^2} = \frac{\sum_{k=1}^{100} x_k^2}{100}$	

それぞれの被験者のタスクに実装される入力支援機能の割り当てを表 4.2 に表す．ただし，*none* は，入力支援機能が適用されないことを表す．None には操作予測が含まれているが，入力支援機能は適用されないので，予測結果は用いられない．

それぞれのタスクに対して，movement time と rate of error と予測正答率を計測する．タスクが開始される前に，現在どの入力支援機能が適用されているかを知らせるメッセージが表示される．その例として，図 4.6 に stalker の入力支援機能（目標ボタンが動的にポインタに近づく方法）が実装されるタスクが開始される前に表示されるメッセージである．

実験（全６つのタスク×８つの数式）に必要な時間は一名当たり約３０分である．

表 4.2　入力支援機能の割り当て

被験者番号	入力支援機能					
	タスク 1	タスク 2	タスク 3	タスク 4	タスク 5	タスク 6
1	None	None	Slow	Blackhole	Big	Stalker
2	None	Slow	Blackhole	Stalker	None	Big
3	None	Blackhole	Stalker	Big	Slow	None
4	None	Big	None	Slow	Stalker	Blackhole
5	None	Stalker	Big	None	Blackhole	Slow
6	None	None	Slow	Blackhole	Big	Stalker
7	None	Slow	Blackhole	Stalker	None	Big
8	None	Blackhole	Stalker	Big	Slow	None
9	None	Big	None	Slow	Stalker	Blackhole
10	None	Stalker	Big	None	Blackhole	Slow

図 4.6　タスクを開始する前に表示されるメッセージ

実験後に，被験者にアンケートを回答してもらう．このアンケートで，各入力支援機能を主観的に評価する．それに加えて，被験者の数式エディタの利用経験の有無は結果に影響を与えるかどうかを調べるために，被験者の数式エディタの利用経験の有無に関する項目もアンケートに載せる．

5 実験結果

5.1 Movement Time

各入力支援機能の movement time を表 5.1 に表す．また，データを箱ひげ図にしたグラフを図 5.1 に示す．この箱ひげ図の箱はデータの第 1 四分位点（箱の下の端）と中央値（箱の中の水平な線）と第 3 四分位点（箱の上の端）を表す．上のひげは第 3 四分位点と外れ値以外の最大値を結ぶ．一方，下のひげは第 1 四分位点と外れ値以外の最小値を結ぶ．外れ値は丸印（○）で表す．ただし，外れ値は，箱との最小の距離

表 5.1 Movement Time

入力支援機能	平均	中央値
none	109.04	101.42
slow	115.06	115.75
blackhole	107.08	109.00
big	102.11	95.01
stalker	121.38	110.69

*単位はすべてミリ秒（ms）

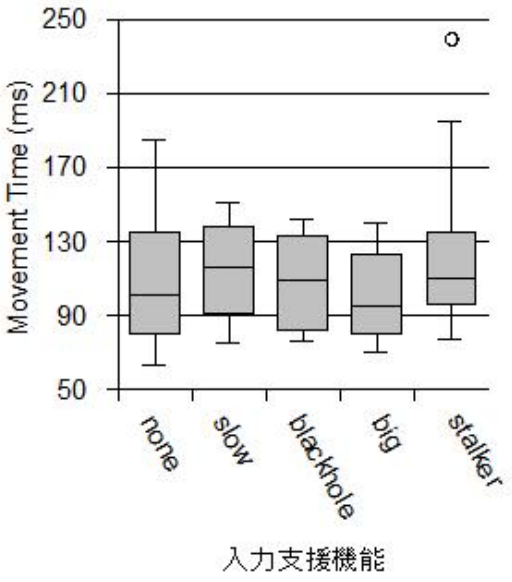


図 5.1 Movement Time

が箱の長さの 1.5 倍以上のデータであると定義する．

表 5.1 に示す movement time の平均を見ると，blackhole と big は入力支援機能なしに比べて短かったが，slow と stalker は逆に長かった．一方，図 5.1 に示す movement time の中央値を見ると，入力支援機能なしに比べて blackhole だけが短かった．Blackhole と slow，stalker の movement time の中央値は入力支援機能なしに比べて長かった．

それらの差は有意であるかどうかを調べるために，両側 t 検定を行った．t 検定の結果を表 5.2 に表す．表では，t 値と p 値を示す．p 値は危険率を示す．本研究では，有意水準を 0.05 とする．つまり，$p < 0.05$ であれば有意差とする．

表 5.2 に示すように，４つ入力支援機能の movement time と入力支援機能なしの movement time には有意差は見られなかった．つまり，実際には，４つの入力支援機能の平均 movement time は，入力支援機能なしの平均 movement time と差がない可能性が高い．

表 5.2　入力支援機能間の Movement Time

	Slow	Blackhole	Big	Stalker
None	$t =$ 0.430 $p =$ 0.672	$t =$ 0.142 $p =$ 0.889	$t =$ 0.505 $p =$ 0.620	$t =$ 0.675 $p =$ 0.508
Slow		$t =$ 0.696 $p =$ 0.496	$t =$ 1.144 $p =$ 0.268	$t =$ 0.381 $p =$ 0.708
Blackhole			$t =$ 0.447 $p =$ 0.660	$t =$ 0.870 $p =$ 0.396
Big				$t =$ 1.179 $p =$ 0.254

5.2 Rate of Error

各入力支援機能の rate of error を表 5.3 に表す．また，データを箱ひげ図にしたグラフを図 5.2 に示す．表 5.3 に示す rate of error の平均を見ると，blackhole と stalker は入力支援機能なしに比べて低かったが，big と slow は逆に高かった．一方，図 5.2 に示す rate of error の中央値を見ると，入力支援機能なしに比べて slow と stalker が高かった．Blackhole と big の rate of error の中央値は入力支援機能なしと同じく 0 であった．

それらの差は有意であるかどうかを調べるために，両側 t 検定を行った．t 検定の結

表 5.3 Rate of Error

入力支援機能	平均	中央値
none	2.22	0.00
slow	2.89	2.50
blackhole	2.19	0.00
big	2.45	0.00
stalker	1.70	1.14

*単位はすべてパーセント（%）

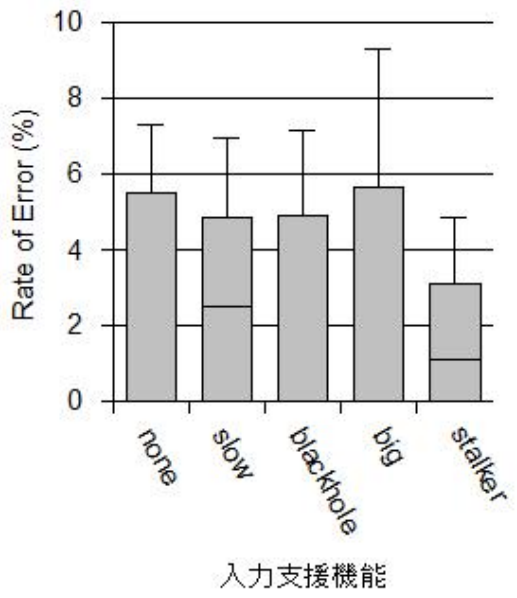


図 5.2 Rate of Error

果を表 5.4 に表す．表 5.2 と同じく，表 5.4 で t 値と p 値を示す．p 値は危険率を示し，有意水準を 0.05 とする．

表 5.4 に示すように，４つ入力支援機能の rate of error と入力支援機能なしの rate of error には有意な差は見られなかった．つまり，実際には，４つの入力支援機能の平均 rate of error は，入力支援機能なしの平均 rate of error と差がない可能性が高い．

5.3 主観的な評価

主観的な評価の結果を表 5.5 に表す．Big の入力支援機能は他の入力支援機能に比べて高く評価され，表 5.5 に示すように「悪い」以下を評価した回答はなかった．

表 5.4　入力支援機能間の Rate of Error

	Slow	Blackhole	Big	Stalker
None	$t =$ 0.534 $p =$ 0.600	$t =$ 0.024 $p =$ 0.981	$t =$ 0.150 $p =$ 0.882	$t =$ 0.440 $p =$ 0.665
Slow		$t =$ 0.587 $p =$ 0.564	$t =$ 0.323 $p =$ 0.750	$t =$ 1.191 $p =$ 0.249
Blackhole			$t =$ 0.177 $p =$ 0.861	$t =$ 0.437 $p =$ 0.668
big				$t =$ 0.572 $p =$ 0.575

表 5.5　主観的な評価

入力支援機能	回答数				平均評価*
	とても悪い	悪い	良い	とても良い	
None	0	4	6	0	2.6
Slow	0	5	5	0	2.5
Blackhole	2	2	3	3	2.7
Big	0	0	3	7	3.7
Stalker	2	4	3	1	2.3

*数値： とても悪い=1； 悪い=2；良い=3； とても良い=4

入力支援機能の間の主観的な評価の差は有意であるかどうかを調べるために，マン・ホイットニーの U 検定を行った．U 検定の結果を表 5.6 に表す．表に U 値と Z 値と危険率 p 値を示す．

表 5.6 に示すように，big と他の入力支援機能の評価の差は有意であった．つまり，他の入力支援機能の間の関係はいえないが，ボタンを big の主観的な評価がもっとも高い確率が非常に高い．

アンケートの自由記述で，１０人中３人が big はボタンを拡大することで視覚的にわかりやすい，と書いていた．これは big の利点であると考えられる．それに比べて，slow には，どれが目標ボタンかを表すかわかりにくい．また，blackhole と stalker は自動的な動きが含まれているので，ユーザの感覚にずれが生じる可能性がある．システムによる自動的な動きは入力を妨げる可能性もあるので，使いにくいと感じるユーザもいる．

5.4 被験者の経験の有無で分割した結果

アンケートの結果を見て，本研究の実験で参加した１０名の被験者のうち，２名は数式エディタを使用した経験があり，２名とも Microsoft Word 2007 の数式エディタを利用した．

表 5.6　入力支援機能間の主観的な評価

	Slow	Blackhole	Big	Stalker
None	$U =$ 45.0 $Z =$ 0.438 $p =$ 0.661	$U =$ 45.0 $Z =$ 0.403 $p =$ 0.687	$U =$ 9.0 $Z =$ 3.342 $p =$ 0.001***	$U =$ 39.0 $Z =$ 0.904 $p =$ 0.366
Slow		$U =$ 42.5 $Z =$ 0.601 $p =$ 0.548	$U =$ 7.5 $Z =$ 3.425 $p =$ 0.001***	$U =$ 42.5 $Z =$ 0.616 $p =$ 0.538
Blackhole			$U =$ 24.0 $Z =$ 2.134 $p =$ 0.033*	$U =$ 39.0 $Z =$ 0.861 $p =$ 0.389
Big				$U =$ 11.0 $Z =$ 3.104 $p =$ 0.002**

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$

被験者の経験の有無でデータを分けて分析を行った．経験の有無で分けた各入力支援機能の movement time，rate of error と主観的な評価を表 5.7 に表す．経験者の movement time を見ると，big は入力支援機能なしに比べて短かったが，blackhole と slow,stalker は長かった．経験者の rate of error を見ると，big は入力支援機能なしに比べて低かったが，blackhole と slow,stalker は高かった．経験者の主観的なを見ると，big は入力支援機能なしに比べて高かったが，blackhole と slow,stalker は低かった．

一方，非経験者の movement time を見ると，big と blackhole，slow は入力支援機能なしに比べて短かったが，stalker は長かった．経験者の rate of error を見ると，blackhole と stalker は入力支援機能なしに比べて低かったが，big と slow は高かった．経験者の主観的なを見ると，big と blackhole は入力支援機能なしに比べて高かったが，slow と stalker は低かった．

表 5.7　経験の有無で分けた評価結果

入力支援機能	Movement Time (ms)		Rate of Error (%)		主観的な評価	
	経験者	非経験者	経験者	非経験者	経験者	非経験者
None	86.57	118.67	0.90	2.79	2.50	2.63
Slow	117.35	114.08	2.44	3.09	2.00	2.63
Blackhole	93.18	113.03	2.38	2.11	1.50	3.00
Big	75.62	113.46	0.00	3.50	3.00	3.88
Stalker	93.41	136.54	1.67	1.95	1.00	2.63

それらの差は有意であるかどうかを調べるために，movement time と rate of error に対して両側 t 検定，主観的な評価に対してマン・ホイットニーの U 検定を行った．すべての検定の有意水準を 0.05 とする．Movement time について，経験者のデータの検定結果を表 5.8 に，非経験者のデータの検定結果を表 5.9 に示す．表に示すように，経験者の big の movement time と slow の movement time の間に有意差が見られたが，非経験者の big の movement time と slow の movement time の間に有意差が見られなかった．経験者の big の movement time と slow の movement time の間の差を除き，有意差が見られなかった．

表 5.8　経験者の入力支援機能間の Movement Time

	Slow	Blackhole	Big	Stalker
None	$t =$ 2.305 $p =$ 0.083	$t =$ 0.452 $p =$ 0.675	$t =$ 0.905 $p =$ 0.417	$t =$ 0.460 $p =$ 0.670
Slow		$t =$ 2.193 $p =$ 0.093	$t =$ 5.722 $p =$ 0.005**	$t =$ 2.113 $p =$ 0.102
Blackhole			$t =$ 1.855 $p =$ 0.137	$t =$ 0.018 $p =$ 0.987
Big				$t =$ 1.811 $p =$ 0.144

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$

表 5.9　非経験者の入力支援機能間の Movement Time

	Slow	Blackhole	Big	Stalker
None	$t =$ 0.248 $p =$ 0.809	$t =$ 0.321 $p =$ 0.754	$t =$ 0.324 $p =$ 0.752	$t =$ 0.624 $p =$ 0.544
Slow		$t =$ 0.067 $p =$ 0.948	$t =$ 0.045 $p =$ 0.965	$t =$ 0.871 $p =$ 0.401
Blackhole			$t =$ 0.034 $p =$ 0.974	$t =$ 0.952 $p =$ 0.360
Big				$t =$ 0.343 $p =$ 0.987

Rate of error について，経験者のデータの検定結果を表 5.10 に，非経験者のデータの検定結果を表 5.11 に示す．表に示すように，rate of error については，有意差がまったく見られなかった．

表 5.10　経験者の入力支援機能間の Rate of Error

	Slow	Blackhole	Big	Stalker
None	$t = 0.534$ $p = 0.600$	$t = 0.921$ $p = 0.409$	$t = 0.582$ $p = 0.592$	$t = 1.000$ $p = 0.374$
Slow		$t = 0.022$ $p = 0.984$	$t = 1.732$ $p = 0.158$	$t = 0.473$ $p = 0.661$
Blackhole			$t = 1.000$ $p = 0.374$	$t = 0.283$ $p = 0.791$
Big				$t = 2.000$ $p = 0.116$

表 5.11　非経験者の入力支援機能間の Rate of Error

	Slow	Blackhole	Big	Stalker
None	$t = 0.179$ $p = 0.861$	$t = 0.406$ $p = 0.692$	$t = 0.355$ $p = 0.729$	$t = 0.665$ $p = 0.519$
Slow		$t = 0.700$ $p = 0.498$	$t = 0.231$ $p = 0.821$	$t = 1.039$ $p = 0.319$
Blackhole			$t = 0.782$ $p = 0.449$	$t = 0.295$ $p = 0.773$
Big				$t = 1.039$ $p = 0.319$

主観的な評価についての経験者のデータの検定結果を表??に示す．表に示すように，経験者の主観的な評価については，有意差がまったく見られなかった．

表 5.12　経験者の入力支援機能間の主観的な評価

	Slow	Blackhole	Big	Stalker
None	$U = 1.0$ $Z = 1.000$ $p = 0.317$	$U = 0.5$ $Z = 1.225$ $p = 0.221$	$U = 1.0$ $Z = 1.000$ $p = 0.317$	$U = 0.0$ $Z = 1.633$ $p = 0.102$
Slow		$U = 1.0$ $Z = 1.000$ $p = 0.317$	$U = 0.0$ $Z = 1.732$ $p = 0.083$	$U = 0.0$ $Z = 1.732$ $p = 0.083$
Blackhole			$U = 0.0$ $Z = 1.633$ $p = 0.102$	$U = 1.0$ $Z = 1.000$ $p = 0.317$
Big				$U = 0.0$ $Z = 1.732$ $p = 0.083$

主観的な評価についての非経験者のデータの検定結果を表 5.13 に示す．表に見えるように，非経験者の big の主観的な評価と他の入力支援機能の間に有意差が見られた．

経験者と非経験者の検定結果の間に相違点がいくつかあることから，被験者の選び方によってこの実験の結果が異なってくる可能性があると示した．しかし，この実験では，経験者の 2 名という非常に少ない人数しかいなかったので，経験者の人数を増やすとまた異なる結果が出る可能性もある．

表 5.13　非経験者の入力支援機能間の主観的な評価

	Slow	Blackhole	Big	Stalker
None	$U = 45.0$ $Z = 0.438$ $p = 0.661$	$U = 45.0$ $Z = 0.403$ $p = 0.687$	$U = 9.0$ $Z = 3.342$ $p = 0.001^{***}$	$U = 39.0$ $Z = 0.904$ $p = 0.366$
Slow		$U = 42.5$ $Z = 0.601$ $p = 0.548$	$U = 7.5$ $Z = 3.425$ $p = 0.001^{***}$	$U = 42.5$ $Z = 0.616$ $p = 0.538$
Blackhole			$U = 24.0$ $Z = 2.134$ $p = 0.033^{*}$	$U = 39.0$ $Z = 0.861$ $p = 0.389$
Big				$U = 11.0$ $Z = 3.104$ $p = 0.002^{**}$

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$

5.5 予測正答率

平均予測正答率とタスク番号の関係を図 5.3 に示す．タスクは番号順に完了されているので，図 5.3 は予測正答率の時間推移ともいえる．タスク 2 の予測正答率は他のタスクの予測率に比べて低いので，予測正答率は時間の経過で高くなる傾向があることが確認できた．

予測正答率は入力支援機能の性能にどんな影響を与えるかを調べるために，movement time・予測正答率の相関関係と rate of error・予測正答率の相関関係を調べるために，相関係数を計算する．その結果を表 5.14 に表す．ただし，r は相関係数を示し，p は両側 t 検定により計算された相関係数の有意性を示し，p が小さいほど $r \neq 0$ の確率が高い．表 5.14 に示すように，予測正答率と rate of error の間に相関関係がある確率が高い（$p < 0.05$）．$r < 0$ なので，予測正答率が高いほど rate of error が低いことを表す．

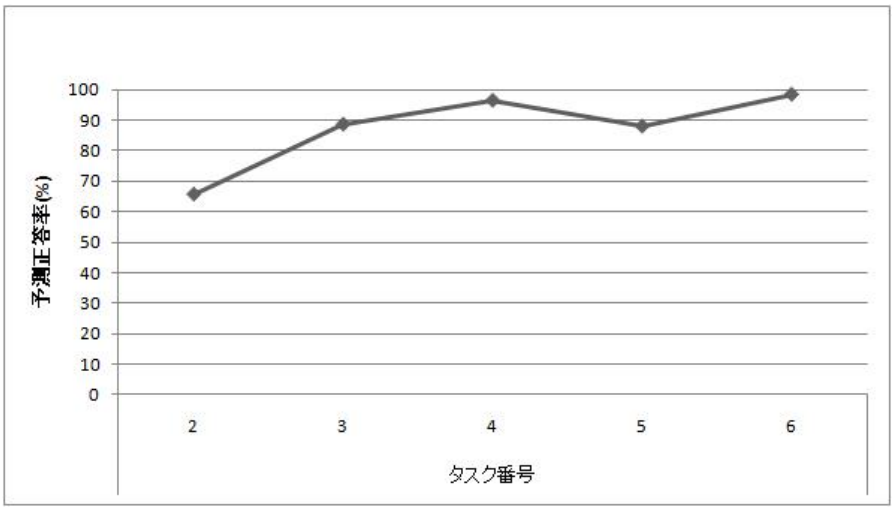


図 5.3　予測正答率とタスク順序

表 5.14　予測正答率と他の変数の相関関係

	Movement Time	Rate of Error
予測正答率	$r =$ -0.129 $p =$ 0.373	$r =$ -0.352 $p =$ 0.012*

*** $p < 0.001$; ** $p < 0.01$; * $p < 0.05$; $^{\#}p < 0.1$

6　考察

6.1　結果のまとめ

実験の結果から，本研究の実験で評価された予測アルゴリズムと組み合わせた４つの入力支援機能とも，ポインティングの効率にはあまり効果がないようで，入力支援機能なしの状態に比べてあまり変わらない．平均を見ると，movement time で big は最も短く，rate of error で stalker は最も低かったが，t 検定の結果を見ると，いずれも，入力支援機能なしの状態との差は有意であるとは考えにくい．

一方，主観的な評価から見れば，big の入力支援機能は他の入力支援機能に比べて高く評価された．Big が主観的に高く評価される理由はこのように考えられる．ユーザがツールバー上のボタンをクリックするために，まず，ユーザがそのボタンの位置を探す．ボタンを見つけてから，マウスを動かして，そのボタンにポインティングをする．

Big を実装すると，操作予測によってユーザが次にクリックする可能性が高いボタンを拡大し，見つけやすくする．予測が正答であれば，ユーザが拡大されたボタンを探す時間を減らすことで入力を支援することができる．予測が正答でなくても，big は他のボタンのクリックをあまり妨げないはずである．

Movement time の計測では，マウスを動かしてからの時間だけを計測し，ボタンを探す時間を考慮しないので，big を実装しても big の movement time と入力支援機能なしの movement time の間に有意差が見られなかったが，おそらく，ボタンを探す時間も考慮すれば，有意差が見られるかもしれない．

本実験の結果を考えて，movement time と rate of error には入力支援機能なしとの差が見られなくても，big の主観的な評価が入力支援機能なしの場合に比較して高いことより，ツールバーに big を実装すると効果があると考えられる．

6.2　入力支援機能の改善

既存の入力支援機能をツールバーに適用しプログラムに実装するとき，入力支援機能の詳細パラメータを著者の判断で決めたので，パラメータの設定に用いられた値は最適であるとは限らなかった．今後，それらのパラメータについてさまざまな条件で実験を行い最良の値を調査することが考えられる．

それぞれの入力支援機能の考察を次に加える．

1. Slow

   この方法で，著者の判断で決めたパラメータは，D／C比が下がる領域（ボタ

ン付近の辺の長さ 50 ピクセルの正四角形）と，その領域でのD／C比の減少量（50%）であった．実験の結果を見ると，有意な差が見られなかったとはいえ，slow の平均 movement time は入力支援機能なしの状態に比べて長かった．D／C比が下がる領域では，より正確なポインティングができるように，ポインタの速度を遅くしたが，遅くしすぎた可能性もある．そのため，その領域でのD／C比を 50% 以上にすれば，movement time が短くなるかもしれない．

2. Blackhole
   この方法で，著者の判断で決めたパラメータは，マウスが自動的に動くまでに手動で動かす必要のある時間（100 ms）と，自動的な動きの速度（1000 ピクセル/s ）であった．実験の結果，特に，図 5.1 を見ると，入力支援機能なしの movement time のデータは，100 ms より短いデータも存在した．これは，マウスが自動的に動かされる前にすでにポインティングを完了していたケースもあることを示しているので，マウスを手動で動かす時間を短くする必要があると考えられる．そして，movement time をより短くするために，自動的な動きの速度を上げることも考えられる．
3. Big
   この方法で，著者の判断で決めたパラメータは，拡大されたボタンの大きさ（辺の長さ 48 ピクセルの正四角形）であった．実験の結果を見ると，big の rate of error はかなり高かった．この rate of error を低くするために，拡大されたボタンの最大の大きさをより大きくすることが考えられる．
   拡大されたボタンの最大の大きさを大きくすれば，フィッツの法則によると，movement time もさらに短くなるはずである．しかし，予測がはずれる場合も考えて，目標ボタンの周りのボタンのクリックを妨げないように目標ボタンを拡大する．
4. Stalker
   この方法で，著者の判断で決めたパラメータは，ボタンの位置を変換する時間間隔（200 ms）であった．図 5.1 を見ると，入力支援機能なしの movement time のデータは，200 ms 以上のデータはなかった．ポインティング中にボタンの位置は変換されない可能性があると考えられる．ボタンの位置を変換する時間間隔を短くすると，より有用なボタンの位置変換が行える可能性があり，movement time が短くできると考えられる．

実験結果によると，予測正答率が高いほど rate of error が低い．この実験では，１０人の被験者が完了したタスクの全体の平均予測正答率は 87.50% である．より予測正答率が高い予測アルゴリズムを使用すれば，rate of error を低くすることができる

と考えられる.

7　おわりに

本研究では，ツールバーを対象とするマウスポインティングの効率を向上することを目的とし，マウスの入力支援機能４種類と既存の予測アルゴリズムを組み合わせ被験者実験を通じて評価した．

評価した入力支援機能は，（１）slow，（２）blackhole，（３）big，（４）stalker の４種類であった．それぞれ，ツールバーのポインティングに適用できるように修正し，操作履歴に基づく操作予測と組み合わせ，数式エディタのプログラムに実装し評価を行った．

評価に用いるパラメータは，（１）movement time，（２）rate of error，（３）主観的な評価で各入力支援機能を被験者実験と実験終了後のアンケートで評価した．

被験者実験の結果，movement time と rate of error の平均を見れば，movement time で big は最も短く，rate of error で stalker は最も低かったが，t 検定を行った結果，どの入力支援機能でも，入力支援機能が実装されない状態に比べて有意な差が見られなかった．一方，主観的な評価から見れば，big は他の入力支援機能と入力支援機能なしの場合に比べて高く評価され，有意差も見られた．全体的に見れば，movement time と rate of error には入力支援機能なしとの差が見られなくても，big の主観的な評価が入力支援機能なしの場合に比較して高いことより，ツールバーに big を実装すると効果があると考えられる．

本研究では，入力支援機能の実装で，多くのパラメータを著者の判断で決定し実験を行った．今後，それらのパラメータについてさまざまな条件で実験を行い最良の値を調査することが考えられる．

謝辞

本論文を書くことや本研究を進めることに，多くの方々の力なしでは完成させることができません．ここで，感謝の言葉を伝えたいと思います．ありがとうございました．

الحمد لله ربّ العلمين

指導教員の上野秀剛助教から，研究に関するガイダンスから論文のチェックまでお忙しいところいろいろお世話になりました．本当にありがとうございます．

査読教員の松村寿枝講師には，査読のコメントやガイダンスなどをいただきました．それらがとても参考になりまして，ありがとうございます．

同じく査読教員の内田眞司講師には，査読のコメントやガイダンスなどをいただきまして，ありがとうございます．

本研究の両方の卒研発表会で，山口智浩教授に予測正答率が与える影響について質問をくださいまして非常に参考になりました．ありがとうございます．

中間発表に，実験用プログラムの開発に関心を見せた浅井文男教授に，例を申し上げます．

同研究室の皆様，いろいろなアドバイスや相談をくださいまして，ありがとうございます．

そして，被験者実験にご協力した皆様，お忙しいところ参加してくださってありがとうございます．

参考文献

[1] Ravin Balakrishnan:“‘ Beating ’Fitts ’law: virtual enhancements for pointing facilitation, ”International Journal of Human-Computer Studies, Vol. 61, pp. 857—874, (2004).

[2] Takeshi Asano, Ehud Sharlin, Yoshifumi Kitamura, Kazuki Takashima and Fumio Kishino:“ Predictive Interaction using the Delphian Desktop, ”UIST ’05 Proceedings of the 18th annual ACM symposium on User interface software and technology, pp. 133—141, (2005).

[3] Fitts,P.M.: “ The information capacity of the human motor system in controlling the amplitude of movement, ” Journal of Experimental Psychology, 47, pp. 381—391, (1954).

[4] 西口宏美：“ 脳性麻痺者のＧＵＩ画面でのマウスポインタ操作の効率化の支援方策に関する一考察：Ｄ／Ｃ比の調整による作業時間値の短縮について, ”日本経営工学会論文誌, Vol. 59, No. 5, pp. 411—417, (2008).

[5] 栢沼正司, 萩原将文：“ 履歴に基づく予測ルールを用いたコマンドライン予測インタフェース, ”日本ファジィ学会誌, Vol. 14, No. 5, pp. 482—490, (2002).

[6] Guiard, Y., Blanch, R. and Beaudouin-Lafon, M.: “ Object pointing: a complement to bitmap pointing in GUIs, ”Proceedings of Graphics Interface, pp. 9—16, (2004).

[7] 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. , ” Proceedings of Interact, pp. 57—64. (2003).

[8] Afke Donker and Pieter Reitsma: “ Young children ’s ability to use a computer mouse, ” Computers & Education, Vol. 48, pp. 602—617, (2007).