RJ-011

赤外線反射光を用いたポインティングデバイスの開発
Development of a Pointing Device using Infrared Light Reflection

土江田 織枝† 宮尾 秀俊‡
Orie Doeda Hidetoshi Miyao

1. はじめに

近年，講義などでは，パソコンの画面をスクリーンに投影し，その投影された画面を使って説明を行うことが多い．その際，発表者は，レーザポインタや指示棒などで着目点を指しながら説明を行う．しかし，パソコンを操作したい場合，レーザポインタや指示棒から直接パソコンを操作することは通常できないため，机上のマウスを操作するために，その場所へ移動しなければならず，説明が中断してしまう．そこで，マウスを使わずに離れた場所からパソコンを操作できるデバイスがあると便利である．これを実現するために先行研究では，デバイスを把持する方法では，レーザポインタを入力に用いる手法[1]や，赤外線 LED 付きのペンを用いた手法[2]などが提案されている．また，デバイスを持たずに直感的にディスプレイとのインタラクションを可能にするポインティングインタフェースとして，単眼カメラで撮影したユーザの指差し画像を重回帰モデルとして扱った手法[3]等が提案されている．

本研究では，ユーザがデバイスを持たず，大掛かりなセットも使わずに，スクリーン上を指差す程度の動作でマウス操作を行うことができる，直接的かつ直観的なポインティングデバイスの開発を目的とする．光を利用したデバイスを考えた場合，何らかの形で発光または撮像する装置が必要となる．このような装置をユーザ自身が持って操作することは，ポインティング操作の妨げになると考える．そこで，本システムでは，指に赤外線反射板をつけ，赤外線カメラ側から赤外線を照射し，指に反射した赤外線をカメラで撮影することにより，指の動きを追跡する Lee の手法[4]を拡張し，指によってコンピュータを遠隔操作することができるポインティングデバイスを提案する．これにより，装置自体がユーザの動作を妨げることがないようにした．なお，本システムは，ユーザの指が指示棒やレーザポインタの役割をするため，ユーザの手の届く範囲の比較的小規模なスクリーン画面での使用を想定している．

2. システムの試作

本システムの構造図を図 1 に示す．本システムでは赤外線を使用する．赤外線投光器からの赤外線をスクリーン上に投光する．ユーザの人差し指と中指に付けた反射シートに反射した赤外線を，CMOS センサーで感知し，スクリーン上の指の位置を検出する．赤外線の反射光の感知とスクリーン上の指の位置座標の取得には，Wii リモコンに搭載されている CMOS センサーを使用した．CMOS センサーで取得した反射光の数と，その位置座標情報は，Wii リモコンからパソコンへ Bluetooth を用いて無線伝送し，パソコン上ではそのデータを元にマウス操作を実現する処理を行った．

反射光の感知が正しく行えるように，赤外線投光器の中央部に Wii リモコンを設置した．プログラムは C#言語で開発し，Wii リモコンの制御にはフリーソフトとして公開されている WiimoteLib[5][6]を使用した．

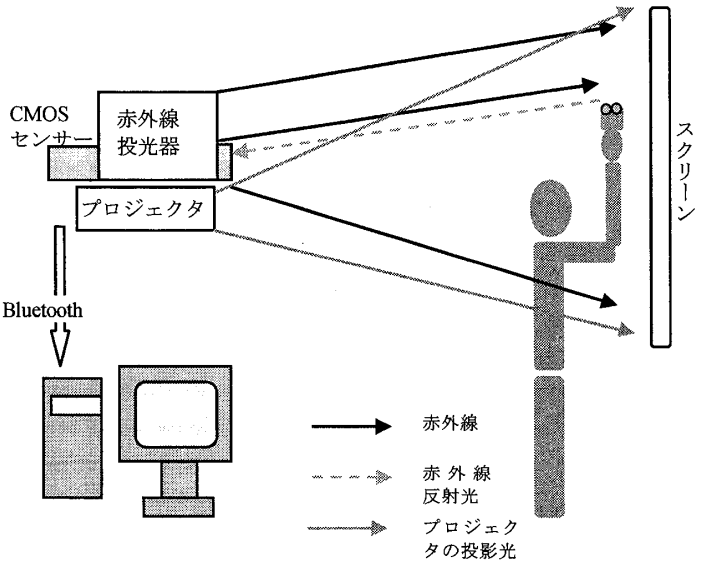


図 1 システムの構成

2.1 赤外線投光器

1 台につき，赤外線発光ダイオード 56 本と，抵抗 8 本を使用して約 2520mW の光出力がある赤外線投光器を作製した．

2.2 CMOS センサー

Wii リモコンに搭載された CMOS センサーは低解像度の CMOS でビデオカメラのようなカラー画像を取得するものではなく，視界に入った赤外線の光の強度の重心位置を複数点取得することができるデバイスである．通常のビデオカメラが 1 秒間に 30-60 枚程度の撮影を行っているのに対し，この CMOS センサーは秒速 200 フレーム以上取得処理を行うことができる[7]．このことからも Wii リモコンの CMOS センサーが非常に高性能であり，本システムのように，指の位置をリアルタイムにカーソルの位置へと対応させるためには，適したデバイスである．

また，CMOS センサーからは，赤外線の座標の位置を取得することができるため，画像処理などを行う必要がないため計算量が少なくて済む．

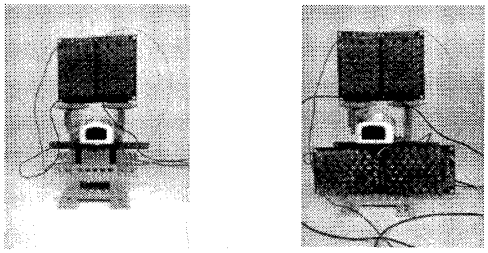
† 釧路工業高等専門学校,Kushiro National College of Technology
‡ 信州大学工学部,Faculty of Engineering, Shinshu University

3. 赤外線の反射に関する実験

本システムは，赤外線の反射光を使用する．そのため，反射シートから CMOS センサーまでの距離をなるべく長く取れると良い．そこで，投光器の数や，反射シートの種類や大きさにより，受光可能距離がどのように変化するか，実験を行った．受光可能距離は，Wii リモコン内の赤外線受光強度の設定を変更し，調整することも可能であるが，受光感度のレベルを高くすると，室内の窓越しに太陽から発せられる赤外線にも過敏に反応してしまう．本システムは太陽光などを遮断した暗室での使用ではなく，窓から太陽光の入る照明を付けた状態の室内環境での使用を想定しているため，Wii リモコン内の受光強度による受光可能距離の調整は行わないこととした．つまり，デフォルト値の WiiLevel3 を使用している．

3.1 投光器の台数

赤外線投光器が 2 台と 4 台使用（図 2）で受光可能な距離にどのような変化があるか，実験を行った．実験では，2 本の指の爪に反射シートを付け（図 3），投光器から照射された赤外線をその反射シートで反射し，その反射光を CMOS センサーで受光させた．この際，パソコンディスプレイ上に，複数の受光位置を異なる色のマーカー点で描画するプログラムを作成し，これを用いて受光距離の確認を行った(図 4)．本システムでは，1 本または 2 本の指先位置を正確に感知でき，かつ余計な光を感知しない状態を見つけたい．このため，プログラムでは，感知した光源を 4 つまで表示するようにし，赤外線を感知しすぎない適切な環境についても調べた．結果は投光器が 2 台のときに比べて 4 台では 20cm 受光可能距離が長くなり 1m30cm だった．投光器の数が増えたことにより若干，受光可能距離が長くなることがわかった．そこで，本システムでは 4 台の投光器を用いることにした．



2 台　　4 台

図 2　赤外線投光器

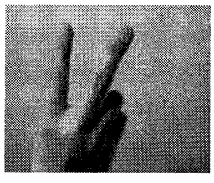


図 3　反射シートを爪に付けた様子

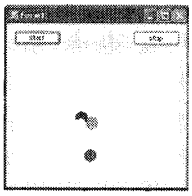


図 4　反射光の感知のプログラム実行画面

3.2 反射シートの種類と大きさ

反射シートには，入射した光がそのまま光源の方に戻るように反射する性質の再帰性反射シート(図 5)と，入射した光を正反射に近い角度で反射する性質の鏡面反射シート(図 6)がある．本システムでは投光器の場所に Wii リモコンを設置するので，再帰性反射シートを使用した．

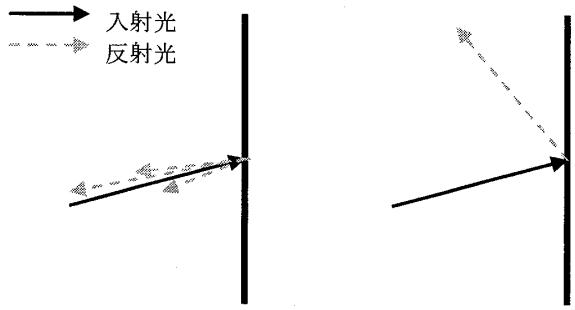


図 5　再帰性反射　　**図 6　鏡面反射**

再帰性反射シートには，高輝度シートと普通のシートがある．一つの反射シートから複数の反射光が出ては支障があるため，その点も考慮してシートの大きさを決定する必要がある．実験は，3.1 節と同様に行った．15mm × 18mm の大きさは，指先に付けて邪魔にならない大きさであり，これ以上大きくすると複数の光を反射してしまうため，この大きさをシートの最大の大きさとした．

表 1　シートの種類と大きさによる反射距離の違い

シートの大きさ	15mm x 18mm	7mm x 15mm	5mm x 12mm
高輝度シート	2m50cm	1m50cm	1m20cm
普通のシート	1m60cm	1m10cm	80cm

表 1 の実験結果から，最長の受光距離を示した 15mm ×18mm の高輝度反射シートを使用することとした．

3.3 反射シートサック

当初，爪に反射シートを貼ることを想定していたが，爪に貼れる大きさは表 1 の 5mm ×12mm が限界であり，この大きさでは反射距離が短いため使えない．また，爪に貼るだけでは手の向きが限定されてしまうので非常に使い難い．

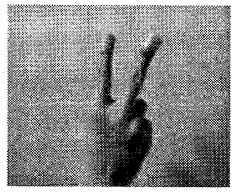


図 7　指サック

手や指の角度に関係なく安定して反射光を生成できるように図 7 のように，シートを円柱状にした指サックを用いることとした．これにより，ほぼどの方向からも 15mm ×18mm の大きさのシートが見えることになる．

4. マウス操作への対応

スクリーン上の指の動作をパソコンのマウス操作へ対応させる処理について説明する．

4.1 カーソルの位置座標

スクリーンとして扱いたい範囲のスクリーン上の 4 隅の座標の値から，内分比を使ったキャリブレーション処理を行うことで，スクリーン上の指の位置を，パソコンの画面上の位置へと変換した．キャリブレーションのためのスクリーンの 4 隅の座標を図 8 の点 A から点 D とし，点 u はカーソルの位置を表している．

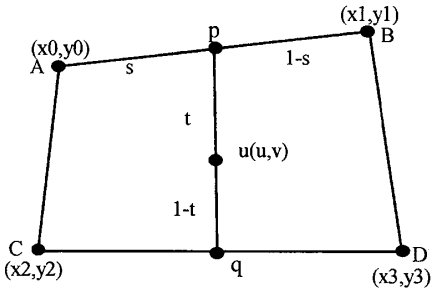


図 8　座標と内分比

線分 AB に対する内分比を s:(1-s)とし，点 u の線分 pq に対する内分比を t:(1-t)とすると点 u は，

$$(u, v) = (1-t)\vec{p}+t\vec{q} \quad (1)$$

となり，内分比を使って点 p と点 q を表すと，

$$\vec{p} = ((1-s)x0+sx1, (1-s)y0+sy1)) \quad (2)$$

$$\vec{q} = ((1-s)x2+sx3, (1-s)y2+sy3)) \quad (3)$$

となり，(1)式～(3)式より，
t を u で解くと，

$$t = (u-\{(1-s)x0+sx1\})/(1-s)(x2-x0)+s(x3-x1) \quad (4)$$

となり，t を v で解くと，

$$t = (v-\{(1-s)y0+sy1\})/(1-s)(y2-y0)+s(y3-y1) \quad (5)$$

となる．
s の値は，(4)式と(5)式より，2 次元方程式 $as^2+bs+c=0$ の形とし，$0 \leqq s \leqq 1$ の条件を満たす解の公式から求めることができる．

4.2 ディスプレイ座標への変換

4.1 節で求めた t と s から，スクリーン上のカーソル位置の座標点 u を，パソコンのディスプレイ上のカーソルの座標点 p として変換する(図 9)．

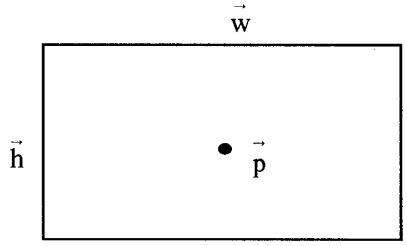


図 9　ディスプレイ座標

点 $p(p_x,p_y)$の座標の値は，

$$\vec{p} = \vec{sw} + \vec{th} \quad (6)$$

(6)式で求めることができる．点 p の x 座標と y 座標の値は，

$$p_x = sw_x + th_x \quad (7)$$

$$p_y = sw_y + th_y \quad (8)$$

(7)式と(8)式で求めることができる．
w と h はそれぞれディスプレイの横幅，縦幅の大きさを表し．$\vec{w}=(w_x,w_y)$, $\vec{h}=(h_x,h_y)$として，表すことができるが，ディスプレイでは，$\vec{w}$は y 軸の値をもたず，$\vec{h}$ は x 軸の値をもたないため$\vec{w}=(w_x,0)$, $\vec{h}=(0,h_y)$となるので，(7)式と(8)式は，

$$p_x = sw_x + 0 \quad (9)$$

$$p_y = 0 + th_y \quad (10)$$

となる．

4.3 マウスイベントの実装

マウスイベント処理は，WiimoteLib でも API が用意されていないため，Win32 プラットホーム SDK の Windows ユーザインタフェースサービスの DLL に含まれる API などを使用した．例として，マウスカーソルを移動するコードを以下に示す．

```
System.Windows.Forms.Cursor.Position =
                    new System.Drawing.Point(x,y)
                                    //カーソルの移動
```

x と y は，4.2 節(9)式，(10)式より求められた値を代入する．
マウスの左ボタンが押されたときの，マウスイベントを発行させるコードを以下に示す．

```
Input[0].mi.dwFlags = 0x0002;        //左ボタンダウン
SendInput(1,input,Marshal.Sizeof(input[0]));
                                     //マウスイベント送信
```

「0x0002」は，WinUser.h に記述されている左ボタンが押されたことを表すマウスイベント定数である．

5. 動作の仕様

本システムでは，離れたところからマウスで行える操作を全て実現することを目的としている．ここでは，あまり用いない右クリックの操作は除外して，マウスの基本操作となる「カーソルの移動」とマウス左ボタンを用いた「クリック」・「ドラッグ」・「ダブルクリック」を実装する．

5.1 カーソルの移動

カーソルの移動は，指一本で行う（図 10）．指の場所にスクリーンのカーソルが移動する．白の矢印はカーソルを表わし，黒の矢印は指の移動を表わす．

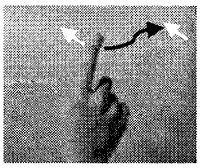
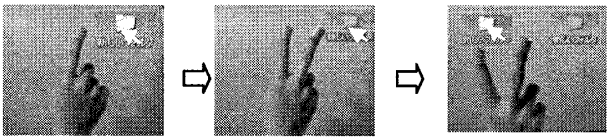


図 10　カーソル移動の動作

5.2 クリックとドラッグ

マウスのクリックの動作は指二本で行う．二つの反射光の位置は動作の判断には使っていないため，二本の指を付けても，離してもどちらでも動作は変わらない．この状態で指を移動するとドラッグの操作を行う．



アイコン上にカーソルを移動　　クリック　　指を移動してドラッグ

図 11　クリックとドラッグの動作

5.3 ダブルクリック

クリック及びドラッグの状態で停止のまま反射シートを一旦見えない状態にしてから指を元に戻すとダブルクリックの操作となる．

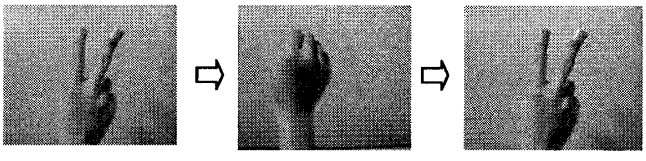


図 12　ダブルクリックの動作

5.4 指の動作とカーソルの座標の対応について

スクリーン上に一本目の指が存在するとき（反射光が一つだけ感知されたとき）には，その反射光の位置がカーソルの位置となる．スクリーン上に二本目の指が存在するとき（二つ目の反射光を感知したとき）には，二つ目の反射光の位置にカーソルを移動させることはせずに，二つの反射光を感知した状態だけを覚えておく．指が，一本→二本の状態に変移し，手がスクリーン上の違う場所に移動するとカーソルの移動を行う．二本の状態のときにカーソルの位置にアイコンがある場合にはドラッグする．この際，どちらが一つ目でどちらが二つ目に出現した指からの光かを区別してマークしていない．そこで，本システムでは，二つの反射光を感知した場合は，より左側の反射光（上下に指が配置されている場合はより下側の反射光）の位置をカーソル位置とすることにした．このため，ドラッグの途中で，カーソルが追随する指が入れ替わることが考えられるが，操作に大きな支障はきたさないと考えた．

5.5 手及び指の向きについて

図 10 から図 12 までのマウス操作の写真では，手は正面を向いている．しかし，反射シートの指サックは円柱状になっているため，手や指の向きに制約はない．実際に本システムを用いる場合，図 13 のような手の向きで使うことが多かった．

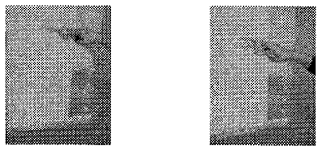


図 13　操作中の手の向き

6. 評価実験と考察

本システムを試用し，評価を行った．実装した機能の他に必要となる機能や，動作の改善点などについて検討した．

6.1 本システム

図 14 は，本システムの概観である．スクリーンやプロジェクタ，赤外線投光器などのシステム構成全体を撮影する都合上，実際よりもスクリーンとプロジェクタの間隔が 1m ほど狭くなっているため，プロジェクタから投影された画面が小さくなっている．実際にはプロジェクタとスクリーンは約 2.5m 離れており，スクリーン画面の大きさは，横 916mm 縦 615mm であった．

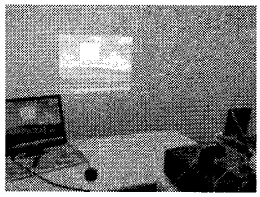


図 14　システムの概観

6.2 被験者

ほぼ毎日パソコンを利用する 14 名を被験者とし，立場の違いによる，本システムの評価の違いについても検証するため，被験者を下記の 2 グループに分類した．

・日常的に，スクリーン画面を使い講義などを行う立場の 7 名（A グループ）
・スクリーン画面を見て講義などを受ける立場の 7 名（B グループ）

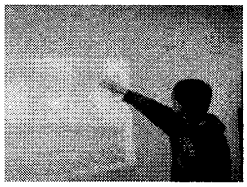
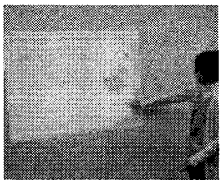
6.3 評価方法と評価内容

被験者には，本システムの使い方を説明後，Web ブラウザの起動や閲覧，リンクページの移動，描画ソフトによる描画，アイコンの移動や内容の表示・起動，コンピュータでよく行う作業等，自由に操作してもらった．

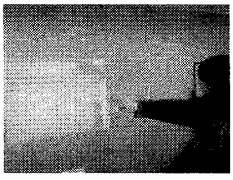
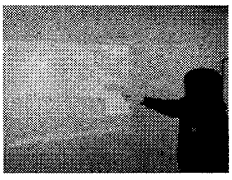
評価の内容は，カーソルの移動やドラッグ，クリック操作について，「使いやすい・使い難い・普通」の選択肢で行った．また，本システムを操作して従来のマウス操作が必要なシステムと比較して便利な点や改善すべき点についても回答を依頼した．

6.4 被験者による試用の様子

図 15 は，評価中の被験者の様子である．本システムは，スクリーン上の着目点を直接スクリーン上に指し示す，いわゆる指示棒を使う時のように，スクリーン上に指を付けて使用する（図 15(a)）ことを想定していた．しかし，投光器の前方で CMOS センサーが反射光を受光できる距離に指があるならば，通常の操作は正しく実行できるので，図 15(b)のように，スクリーンから指を離して使用することも可能であるため，レーザポインタのように，着目点をスクリーンから離れた場所からも指すことができる．試用の結果，スクリーンからかなり離れた位置からも，スクリーン上のパソコン操作が可能となるため，被験者のほとんどが，スクリーンから離れた位置での操作に興味を示し操作を行っていた．

(a)　指をスクリーン上に付けて使用

(b)　指をスクリーンから離して使用

図 15　評価の様子

6.5 結果と考察

図 16 の結果から，指を移動することで操作を行うことができる「カーソル移動とドラック操作」については，「使い難い」との回答が 21%であり，「普通」は 64%，「使いやすい」は 14%であった．次に「クリック操作」については，図 17 より，「使い難い」は 21%，「普通」は 50%，「使いやすい」は 28%であった．この結果から，本システムのマウスの操作を行う動作については，使い難いとの回答率が低いため，ほぼ問題ないものと考えられる．

次に，スクリーンの内容をマウスで変更する従来のシステムとの比較については，図 18 より，A グループでは「使いやすい」・「普通」との回答が 100%で，「使い難い」との回答者はいなかった．しかし，B グループでは，「使いやすい」・「普通」との回答が 42%であり，「使い難い」との回答が上回った．この結果について，B グループの被験者からは，従来のシステムを用いるのは，主にプレゼンテーションを行う場合であり，その際は，パソコン付近でマウスを操作しながら，レーザポインタを用いてスクリーンに着目点を表示することが多いため，離れた場所から，複雑なマウス操作を行う必要性をあまり感じていない，との意見が多数あった．一方，A グループの被験者からは，講義などで従来のシステムを頻繁に使用している．その際，レーザポインタを用いずにスクリーンを直接指示棒や手を使って指し示す場合が多く，パソコン付近から離れていることが多い．また，講義ではプレゼンテーションのみでなく，パソコン上での実際の操作を例示しなければならない場合も数多くある．このような背景から，パソコン付近に移動してマウス操作をしなければならない従来のシステムに対して不便を感じている，との意見が多かった．以上の理由から，比較実験で被験者 A と B の間に差が生じたと考えられる．

本システムを試用した意見として，A グループの被験者からは，従来のシステムの使い難さを普段感じているため，本システムの反射シートサックを指に付けるだけでスクリーン上の内容を変更できる点，マウスの場所を考えずに自由に動ける点，スクリーンから離れた場所からも操作が可能な点が使いやすいとの好評な意見が得られた．また，B グループの被験者からは，本システムを講義などで使用できると，ユーザ（教員）がスクリーンやパソコンのそばにいなくても良くなるため，聴衆（学生）のところからスクリーンの内容を変更しながら説明が行えるので，ユーザと聴衆のコミュニケーションを行いながら講義ができるといった意見があった．

この結果からも，本システムが従来のシステムに比べ，使い勝手が良いと評価できる．しかし，被験者からは，カーソル位置を正確に指定できるようにし，マウスのような安定した動作に近づけると，もっと使いやすくなるとの意見が多かった．

7. おわりに

本研究では，スクリーンに資料を提示し，それを使った講義の際に生じる，机上のマウスによるパソコン操作を排除するため，赤外線反射光を用いたポインティングデバイスの開発と試作を行った．本システムでは，ユー

ザが特別なデバイスを持たず，スクリーン上を指差す程度の動作でマウス操作を実現し，直接的かつ直観的にパソコンの操作を行うことができる．被験者による評価の結果から，従来のようにマウスを意識することなく使うことができる点や，反射シートサックを指につけるだけでシステムをユーザが意識せずに使うことができる点など好評な意見が多く聞かれた．一方，カーソルの動きにマウスのような正確さと安定性が実現されていないため，狭い範囲でのカーソルの動きについて，従来のマウスに比べると使い難いとの意見があった．今後は，マウス動作の判断に，指のジェスチャーの情報も利用することで，使いやすさの面も改善されると考えられる．また，スクリーンから離れて使用できることが，被験者には大変好評だったので，投光器の位置や，邪魔に思えるスクリーン上の影の利用も考えながら改良を進める予定である．

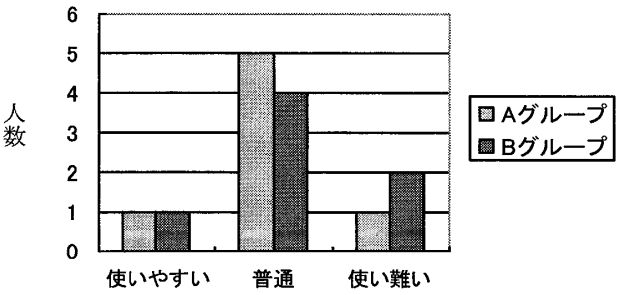


図 16　カーソル移動とドラッグ操作について

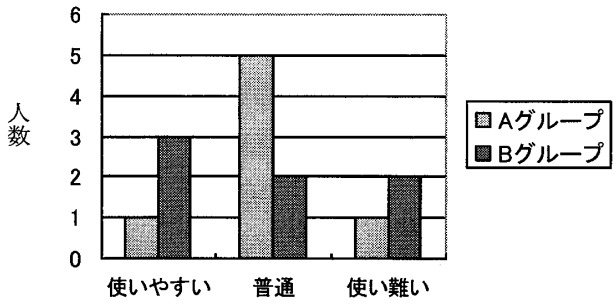


図 17　クリック操作について

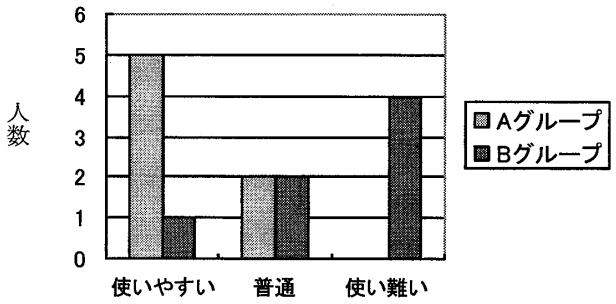


図 18　従来のシステムを使った時との
使いやすさの比較

参考文献

[1]久松孝臣,岩淵志学, 三末和男, 田中二郎,“大画面向けインタフェースへのレーザポインタの応用”,2005 年度第 19 回人工知能学会全国大会論文集 (CD-ROM),2005,巻:19th,頁:3D1-01(2005)
[2]Johnny Chung Lee, “Low-Cost Multi-point Interactive Whiteboards Using the Wiimote”, http://johnnylee.net/projects/wii. (accessed 2010-03-17)
[3]新谷晃市,間下以大,清川清,竹村治雄, “大画面ポインティングシステムのための回帰モデルによる単眼画像からの指差し位置の推定”,情報処理学会研究報告.CVIM,[コンピュータビジョンとイメージメディア],Vol.2009-CVIM-167,No.33(2009)
[4]Johnny Chung Lee, “Tracking Your Fingers with the Wiimote”, http://johnnylee.net/projects/wii. (accessed 2010-03-17)
[5]Brian Peek,“BrianPeek.com”, http://www.brianpeek.com/.(accessed 2010-03-17)
[6]CodePlex, “CodePlex”, “Project Hosting for Open Source Software”, http://www.codeplex.com/.(accessed 2010-03-17)
[7]白井暁彦,小坂崇之,くるくる研究室,木村秀敬. “WiiRemote プログラミング”,オーム社,(2009)