ヒューマンインタフェース　95－6
コンピュータビジョンと
イメージメディア　129－6
（２００１．９．１３）

視点による情報提示インタフェースの試作と評価

成田 智也　渋谷 雄　中村 重雄　物部 文彦　辻野 嘉宏
京都工芸繊維大学

ActionView は，ビデオキャプチャを利用した非接触インタフェースで，小さい面積のディスプレイを通して大量の情報を覗き込むことができる．ActionView は窓メタファを用い，ユーザは望む情報を，視点を移動することによりディスプレイ上に表示させることができる．覗き込むという動作は，人の日常の行動であるため，ActionView の操作方法を習得するのは容易である．ActionView を用いることにより，ディスプレイ面積よりも大きな面積の情報が提示可能であると考えられる．本稿では，実際に ActionView の面積拡大効果の有効性，ならびに奥行きの利用を考え，これらに関しての評価を行なった．

Introduction and experimental evaluation of view-controllable information display

Tomoya Narita, Yu Shibuya, Shigeo Nakamura, Fumihiko Monobe and Yoshihiro Tsujino[*1]
Kyoto Institute of Technology

Abstract – In this paper, a novel interface, named ActionView, is introduced and evaluated experimentally. ActionView is a sort of non-touch interface using video captured image and is aiming to establish an interaction environment in which the user can look around the large information field through the small display. In ActionView, the window metaphor is applied for the operation manner. It is easy to learn and understand how to use ActionView because its interaction method depends on the ordinary manner in our daily life. ActionView is evaluated experimentally, and it is found that ActionView is usable to look around one large information sheet through the small display.

1.　はじめに

情報機器の進歩により，その扱う情報量はますます多くなってきている．情報提示面積に制限のある情報機器において大量の情報を提示するには，提示方法に工夫が必要となってくる．そこで，メニュー項目をアイコン等で表す，情報をスクロールする，必要に応じて情報を拡大・縮小する，多次元空間を用いて表現するなどの方法がとられている[1][2][3][4][5][6][7].

本研究では，情報を窓越しに覗き込むことができるように提示することにより，あたかも情報提示面積が大きくなったかのようにみせる情報提示システム「ActionView」を提案する．ActionView において，ユーザは情報提示装置（ディスプレイ）を窓というメタファとしてとらえ，ディスプレイを覗き込む視点をかえることにより，情報の見え方をコントロールすることができる．ActionView をもちいることにより，小型情報機器においても大量の情報を提示することが可能となる．

2.　窓メタファ

2.1　窓メタファとは

ユーザが，窓越しに覗き込むようにして情報を閲覧することができる情報提示方法を，窓メタファを用いた情報提示と呼ぶ．ディスプレイを窓としてとらえることにより，ユーザはディスプレイ越しの情報を覗き込むことができることを暗黙のうちに知ることができる．覗き込むという動作は日常の自然なものであり，特に学習しなおす必要はない．

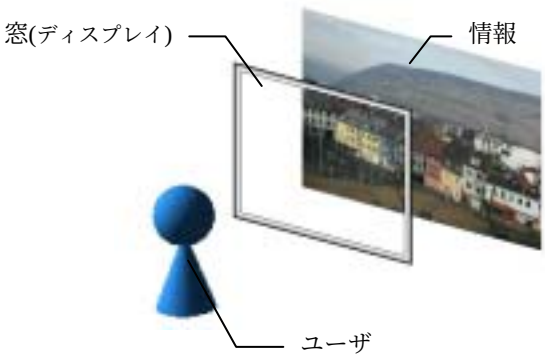


図 1 窓メタファのとらえ方

通常のディスプレイへの情報提示は，ディスプレイ表面に情報を提示する．ディスプレイに投影されている情報が，一度に提示することができる情報の全てであり，インタフェースの設計者は，必要とする情報をディスプレイ一画面にうまく納めなければならない．

窓メタファを用いた情報提示では，図 1 に示すように，情報を，それを提示するディスプレイの奥に存在するかのように提示する．よって，提示する情報の面積はディスプレイと同じである必要はなく，より大きなものを提示することができる．ディスプレイに投影される情報の位置および大きさの見え方は，ユーザの視点移動によって変化する．無論，同時に提示することができる情報量は，ディスプレイの解像度が同じである以上は変わらないが，覗きこむことにより視点を変える無意識の動作により隠れた情報を見ることができるため，より多くの情報を提示することができると言える．

2.2 覗き込み

ユーザは，ディスプレイを窓として，ディスプレイ枠を窓枠としてとらえる．窓枠に隠れて見えない情報があると，ユーザは視点をずらすことによりその情報を見ようとする．その際，視点の移動によりずれる情報の移動量は，窓からの距離が遠ければ遠いほど大きくなる．そこからユーザは情報までの距離をとらえることができる．

複数の情報を違う距離に配置すると，それぞれは違った量のずれかたをする．つまり，窓枠と同様，ユーザは手前の情報に隠れた奥の情報を視点の移動によって見ることができるようになる．つまり，情報を平面的に配置するのではなく，奥行きをもたせて 3 次元的に配置して提示することができる．

3. ActionView

3.1 ActionView の概要

窓メタファを用いて，ディスプレイ越しに情報を覗き込むことができる情報提示システムが ActionView である[8]．情報をシートとして，情報空間に 3 次元的に配置する．一見，従来のオーバーラップドウィンドウシステムのようにはなるが，各情報までの距離が定義されているところで異なる．通常，遠い距離に置かれた情報は小さく表示されることになる．

ActionView の実現には，視点を検出する必要があるが，システムの使用状況を制約しないため，ユーザに視点検出用の特殊なデバイスを装着することを避ける．そこで，ビデオカメラをディスプレイの近傍に装着し，ビデオカメラからキャプチャした画像からユーザの視点を検出することで，非接触なインタフェースを実現している．

3.2 システム構成

ActionView を実現するためのハードウェア構成は，情報を提示するためのディスプレイ，視点を検出するためのビデオカメラ，および，視点位置の計算とそれに基づく情報の描画処理を行うためのコンピュータである．CPU については，MMX Pentium 266MHz 程度の処理能力で動作させることが可能である．

ソフトウェアとして，ActionView は情報提示ウィンドウ，情報管理シート，視点検出の 3 つのモジュールからなる．情報提示ウィンドウは，ひとつ，または複数の情報管理シートに対して視点検出によって得た情報をもとに，各シートの表示位置と大きさを計算し，その情報を各シートに渡す．各シートは，これを元に各々がもつ情報を描画を行う．情報管理シートの立場からは，特には ActionView を意識する必要はない．視点検出モジュールについては次に述べる．

3.3 視点検出

ActionView においての視点情報とは，ディスプレ

図 2 視点検出において輝度を採取する部位

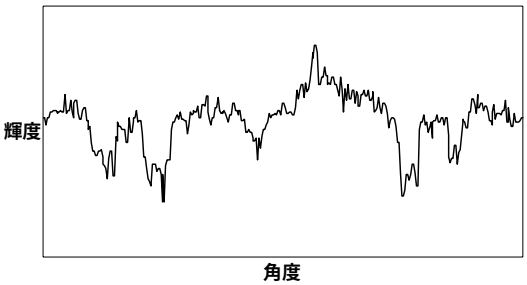


図 3 眉間の周りの輝度波形

イとユーザの視点位置（眉間の位置）との位置関係のことをいう．位置関係は，ディスプレイから視点位置までの角位置，および距離で定義される．視点検出によって求める必要があるのは，この視点情報である．

ユーザの視点は，ビデオカメラからの画像のみによって検出する．人の顔において，図 2 ように，眉間を中心として両目，両眉を通る円周上の輝度を採取すると図 3 のような波形が得られる．この波形と，あらかじめ人間の同位置において採取しておいた輝度波形（テンプレート）との相関係数を調べることによって視点位置を特定する．テンプレートは，実際に ActionView を使用するユーザのもの，また使用する環境と同じ環境で採取しておく方が，視点検出の精度は向上する．

ActionView の視点検出によって，ユーザの視点のカメラの視野内での角位置，およびカメラからの距離を算出する．実際にはカメラとディスプレイは近くに設置されているので，視点検出の検出結果はディスプレイからの角位置と距離に近似している．

ユーザの視点の角位置は，カメラからの入力画像におけるユーザの視点の直交座標そのものが，またユーザの視点の距離は，検出した両目の距離，つまり図 2 における円の直径から求まる．

視点検出には，処理速度が要求される．視点検出に時間がかかると，検出毎のユーザの視点の移動量が大きくなる．視点検出を高速に行うことにより，直前のユーザの視点位置と現在の位置との相関性が高くなり，視点検出の精度が向上する．また視点検出に時間がかかると，情報を出力する際の即時性にも悪影響を及ぼし，ユーザのしぐさに対するディスプレイ上のフィードバックが不自然なものとなる．

3.4 情報提示

ActionView では，情報をシートという 2 次元の長方形領域に提示する．このシートには，面積および 3 次元の位置が定義されており，この値とユーザの視点座標を元に，窓，すなわちディスプレイ上の座標，および大きさが計算される．

実世界的な提示を行う ActionView においての計算方法は次のようになる．ユーザの視点検出によって得られる情報は，図 4 に示すように，カメラを中心としたユーザの視点の極座標，θ，d である．シートの位置の直交座標を L,D であらわした際，ディスプレイ上に表示する際の座標は(1)のようになる．

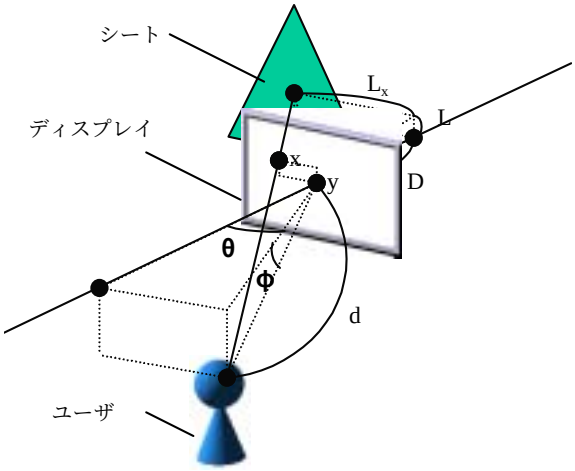


図 4 シート提示位置の算出

$$\begin{cases} x = \dfrac{Dd\sin\theta - L_x d\cos\phi\cos\theta}{D + d\cos\phi\cos\theta}, \\ y = \dfrac{Dd\sin\phi - L_y d\cos\phi\cos\theta}{D + d\cos\phi\cos\theta} \end{cases} \quad (1)$$

またシートの表示する大きさは，元の大きさに対して次式の倍率となる．

$$\frac{d\cos\theta}{\left(D + d\cos\phi\cos\theta\right)} \quad (2)$$

4. ActionView による情報提示面積の拡大

4.1 実験の目的

ActionView により，情報を提示するシート面積をディスプレイの面積より大きく設定することができる．ユーザはこのシートをディスプレイ越しに覗きこむわけであるが，ここでは ActionView により，ユーザにとって実際に情報提示面積拡大の効果が得られるかを考察する．

一般に大きなウィンドウを用いて情報を探すほうが，より効率的に情報を見つけ出すことができると考えられる．これは，狭いウィンドウを通して暗中模索の状態で情報を探し出すことにより，無駄な試行が増え，探索の効率が落ちるためである．ウィンドウが大きくなると，目標の対象物が情報の可視領域に入る可能性が高くなり，探索効率が高くなる

と考えられる．本実験においては，ウィンドウサイズは不変であっても，ActionView を用いることにより情報提示面積が大きくなっているかどうかを，この前提をもとに調査する．

4.2 実験環境

実験に用いたハードウェアは，カメラつきノートパソコン SONY VAIO-PCG C1R である．被験者は椅子に座って机の上のパソコンを操作する．必要とあれば，ユーザはパソコンを持ち上げ，手にとって操作することができる．

4.3 実験タスク

一枚の日本地図があり，被験者にはその一部分がウィンドウを通して見えている．ウィンドウのサイズは 160×120[pix]であり，ディスプレイ中央上部の一部分に提示される．地図の大きさは 2124×1938[pix]である．つまり，被験者は同時に地図全体の 1／200 程度の部分しか閲覧することができない（図 5 を参照）．そこで，被験者は上下左右のカーソルキーを用いて地図を 40[pix]ずつスクロールさせることができる．その方向は縦横方向のみであり，斜め方向にスクロールさせることはできない．

はじめ，北海道の一部がウィンドウを通して見えていて，ここが出発点となる．実験開始とともに，地図のランダムに選ばれた都道府県の場所に旗が一本立つ．被験者は，カーソルキーを用いて地図をスクロールさせながら，たてられた旗を探し出し，その都道府県名を記録する．都道府県名は旗の横に文字で記されているため，被験者の地図に対する知識は必要ない．このタスクを 20 回繰り返し，そのときカーソルを押した回数，および発見に要した時間を記録する．なお，キーを押しつづけるときに発生するリピートは，リピート回数分だけキー押下回数に加算する．見つける旗の大きさは 32 x 32[pix]である．

上記の実験を，通常の情報提示，および ActionView を用いた情報提示について行う．通常の情報提示では，ウィンドウ内にウィンドウの大きさの情報が提示される．ActionView を用いた情報提示では，160×120[pix]のウィンドウ越しに 320×240[pix]のシートが提示され，シート上に地図情報が提示されている（図 6 を参照）．つまり，シートには通常ディスプレイの 4 倍の面積の情報が提示されていることになる．この提示方法で，ActionView を使用しないシステムと同じようにカーソルキーによりスクロールを行なう．

4.4 結果

ここに 3 人の被験者の実験結果を示す．被験者 1，

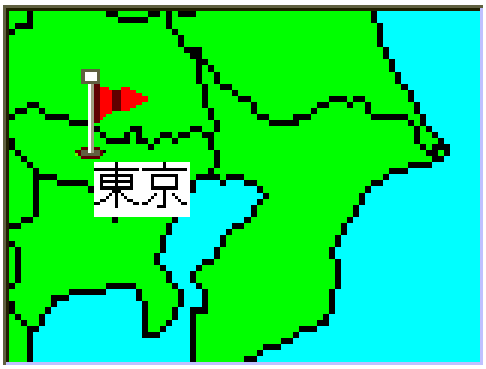


図 2 地図提示ウィンドウ

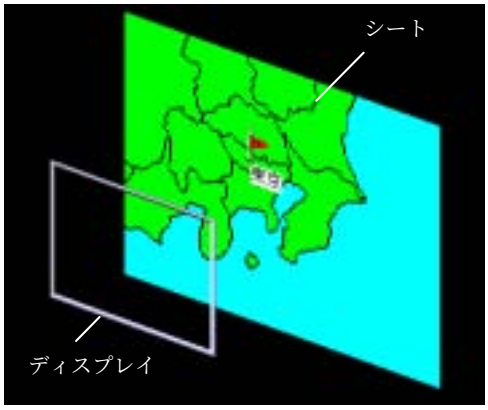


図 2 ActionView を用いた地図提示

および被験者 2 はコンピュータ，および ActionView に多少の知識と経験があり，被験者 3 は ActionView は全く初めてであった．

グラフの横軸は実験開始点である北海道から，旗の立っている都道府県までの距離である．地図のスクロールは縦横方向のみであるため，距離 $d[pix]$ は次式で定義する．

$$d = \Delta x + \Delta y \quad [pix] \qquad (3)$$

ただし，Δx は北海道と旗の立っている都道府県との横方向の差異，Δy は縦方向の差異である．

全ての被験者は，旗の見落としをしないようにジグザグ状に捜索を行った．図 7 のグラフ中の打鍵数の理想値は，全く寄り道なしで目標を発見することができた際の打鍵数である．理想値を下回る打鍵数が存在するのは，ウィンドウの大きさにより，旗を中心にスクロールしなくてもよいことによるアドバンテージである．たまに打鍵数が極端に多いもの

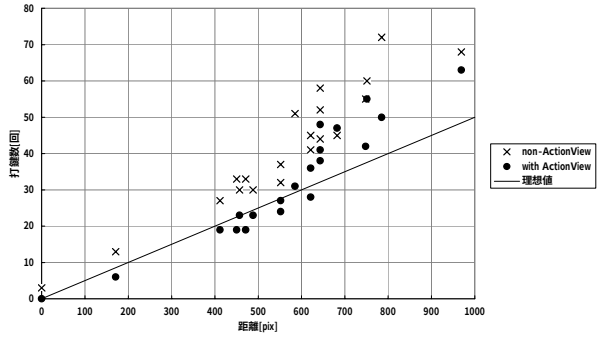
が存在するのは，見落としにより後戻りして探索を行ったことによる．

図 8 の所要時間は北海道から探索を開始し，旗を見つけたとして Enter キーを押すまでに所要した時間である．

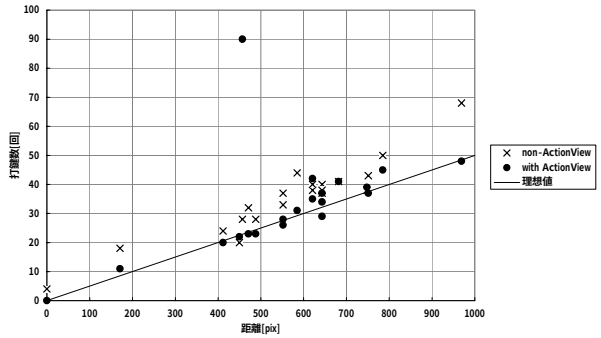
4.5 考察

図 7 から，ActionView を用いることにより一般的に打鍵数が減少することがわかる．被験者は，視点を移動することにより，スクロールして新しく提示された部分を確認していた．しかし，被験者 3 は ActionView に不慣れであり，実験の際ほとんど視点を移動させなかった．結果的に，視点検出のミスによる操作性の低下により無駄な操作が増え，逆効果が見られる．ActionView を使用するには，ある程度の慣れが必要なことがわかる．

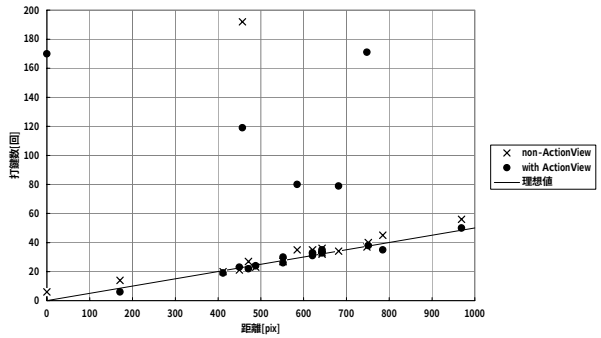
図 8 から発見に所要した時間は，ActionView を用いる場合とそうでない場合とではほとんど差が見られなかった．逆に打鍵数と同く，ActionView の操作性の低下から時間が余計にかかっている部分がみられる．しかし，今回実験に用いたカーソルキーはノートパソコンのキーボードであり，面積もキーストロークもあり比較的操作性がよいものであると考え



(a) 被験者 1

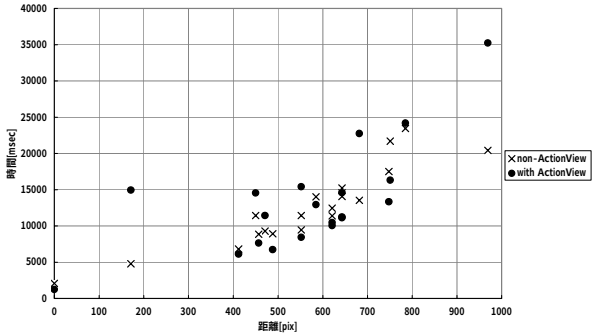


(b) 被験者 2

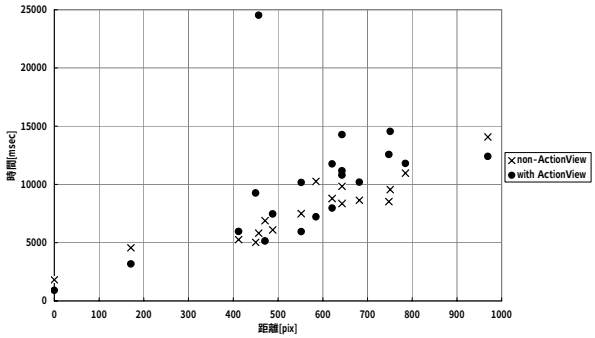


(c) 被験者 3

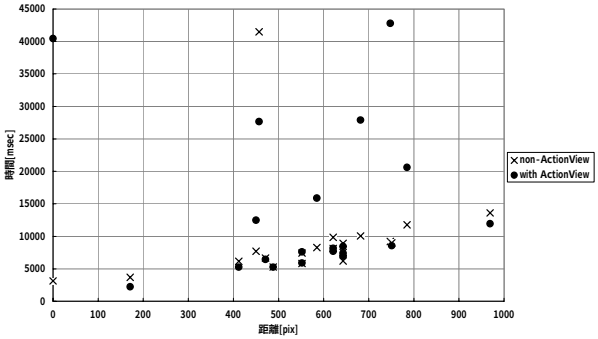
図 7 目標までの距離と打鍵数



(a) 被験者 1



(b) 被験者 2



(c) 被験者 3

図 8 目標までの距離と発見に所要した時間

られる．小型の携帯端末で，このような連打の利くボタンを搭載することができない場合，打鍵数の増加は操作時間の増加を招く．よって，本実験におけるシステムでは操作時間には ActionView の効果は得られなかったが，使用するハードウェアによっては効果が考えられる．

5. ActionView による奥行き情報の利用

5.1 実験の目的

ここでは，ActionView においてシートを 3 次元配置した場合のの有効性について考察を行う．

メニューにおいて，項目が増えると目的の項目を探すのが困難になるため，関連のある項目ごとにまとめて階層性を持たせるのが一般的である．ユーザは，上の階層の項目の名前，アイコンなどからその下の階層に含まれる項目を憶測し，下の階層に移って項目を選択する．あらかじめ上の階層の項目を選択する前に下の階層が見えていることは，この憶測の失敗を減らすことが考えられ，階層性のあるメニュー項目では有意と考えられる．そこで，階層性のあるメニューにおいて，メニューの下の階層の項目の提示に奥行きを用いた．

5.2 実験環境

実験に用いたハードウェアは，カメラつきノートパソコン SONY VAIO-PCG C1R である．被験者は椅子に座って机の上のパソコンを操作する．必要とあれば，ユーザはパソコンを持ち上げ，手にとって操作することができる．

5.3 実験タスク

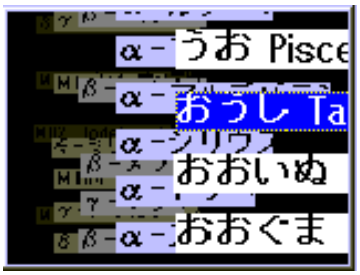
3 種類の提示方法による，階層性のあるリストメニューから目的の項目を探し出すタスクを，４章における実験と同じ被験者におこなってもらった．

実験では簡略のため，一段のみの階層性をもつ情報源として，星座，恒星および星雲を用いた．上の階層の項目は，88 個のうち適当と選んだ 26 個の星座であり，下の階層の項目はその星座に含まれる恒星，星雲である．ひとつの星座には，0～11 個の項目が含まれる．

上記の情報源を，List View，Tree View，ActionView を用いて探索を行う．それぞれに対して同等のタスクを与え，その際要した時間と，キーの打鍵数を測定した．

(a) List View

(b) Tree View



(c) ActionView

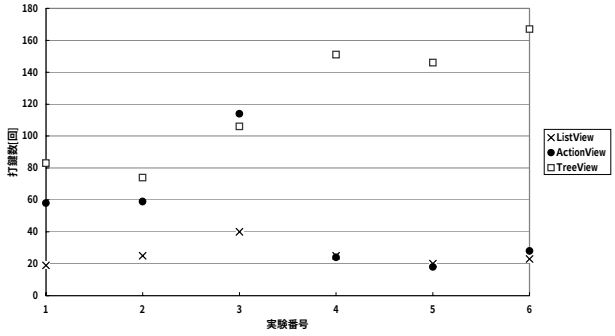
図 9 実験に用いた提示方法

本実験においては，どの星座にどの星があるといったような被験者の星に関する知識は全くない．よって，実験はリストから目的の項目を探し出すだけの単純な行為である．星を見つけ出すタスクとしては，2 種類存在する．ひとつは目的の星が何座にあるかわからない状態で探す場合，もうひとつは，目的の星がどの星座にあるかわかった状態で探す場合である．実験番号と，これらの種類との関係を表 1 に示す．表 1 での上の階層の項目の場所が未知のものが前者であり，既知のものが後者である．

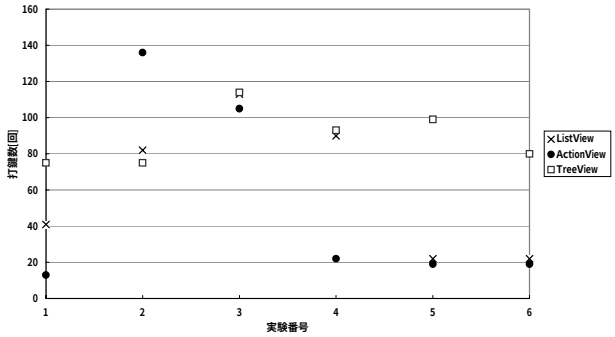
5.3.1 List View によるメニュー

表 1 実験番号とその内容

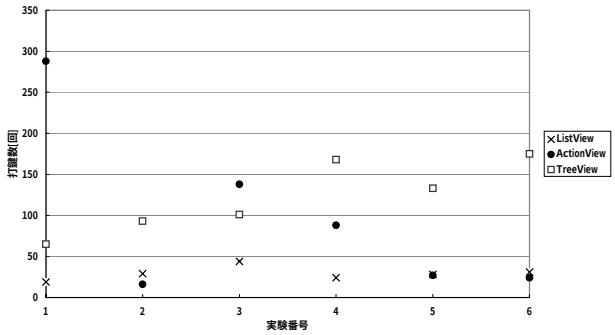
実験番号	1	2	3	4	5	6
上の階層の項目の場所	未知	未知	未知	既知	既知	既知
リスト中での項目の位置	6	8	14	22	18	19～22



(a)被験者 1

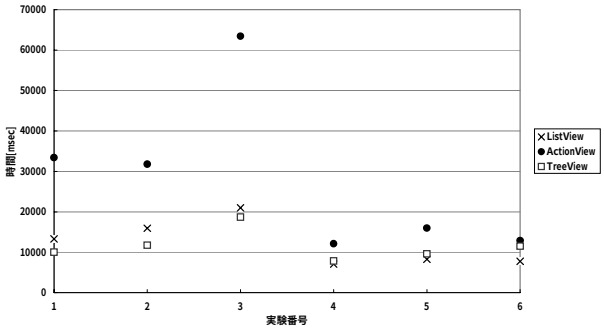


(b)被験者 2

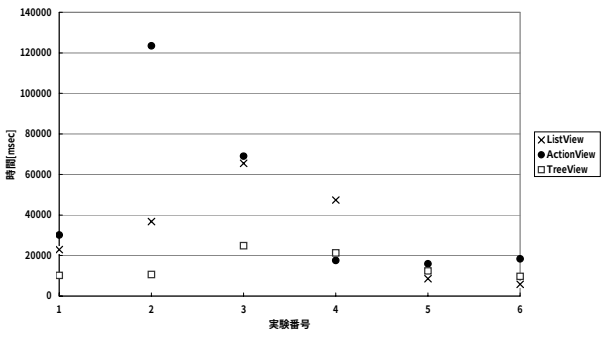


(c)被験者 3

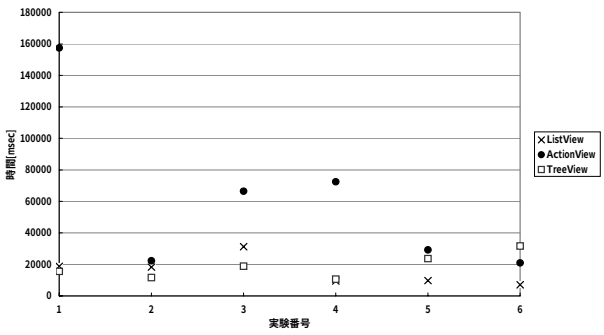
図 10 項目を発見するまでの打鍵数



(a)被験者 1



(b)被験者 2



(c)被験者 3

図 11 項目を発見するのに要した時間

図 9(a)のようなメニューである．ユーザはカーソルキーの上下を用いてカーソルを動かし，エンターキーを押してその下の階層へ移ることができる．下の階層から上の階層へ戻る際は，最上部に用意された「戻る」という項目を選択する．

List View において，ユーザは，上の階層の項目を選択するまで，下の階層の項目を一切見ることはできない．よって項目がうまく整理されていて，上の項目により下の項目の分類がユーザにわかりやすければよいインタフェースであると考えられる．

5.3.2 Tree View によるメニュー

図 9(b)のようなメニューである．項目ははじめ全て展開されてツリー構造として提示されている．ユーザはカーソルキーの上下を用いてカーソルを動かし，カーソルの左キーで展開された項目を畳み，右キーで畳まれた項目を展開する．

Tree View では，上の階層，下の階層とも全てが画面に提示されている．しかし，全ての項目が提示されるため，項目の数が増えるため，下のほうに存在する項目を選択するまでのカーソルキーの打鍵数

は多くなると推定できる．

5.3.3　ActionView によるメニュー

図 9(c)のようなメニューである．メニュー項目が隙間を開けて縦に並んでおり，下の階層の項目が存在する場合は，左奥の方向に順に項目が並んで見えている．奥の項目は，手前の項目に覆い隠され，必ずしも全てが確実に見えるわけではないが，視点を変えることによりずらして見ることが可能となるものもある．項目は奥へ行くほど薄暗く提示され，項目の奥行きを知ることができる．

操作方法は ListView とほぼ同等であり，下の階層の項目が見えているか見えていないかという点で両者は異なる．

5.4　結果

実験４と同じ被験者 3 人の実験結果を図 10，11 に示す．グラフの横軸は，表１における実験番号である．目的の項目の場所がわかっている場合（実験番号 4〜6），ListView が最短の時間，および最小の打鍵数となっている．TreeView では，一度に提示される項目の数が増えるため打鍵数が著しく増える．そのため，所要時間は ListView より長い．

目的の項目の場所がわかっていない場合（実験番号 1〜3）でも，項目の数から TreeView の打鍵数は多くなる．しかし，所要時間に関してはいちいち下の階層の項目に移る必要のない TreeView が一番短い．

ActionView は上のどちらに関しても ListView に近い値ではあるが, いずれにおいても 傾向は似ているものの ListView よりも悪い値となった．特に実験番号 1〜3 では，打鍵数は ListView より多くなり，所要時間も多くかかっている．つまり，本実験での提示方法では奥行き情報の利用による効果は得られなかったと言える．

5.5　考察

被験者らによると，下の階層の見え方の不確からしさから，項目を探す際奥に見える情報はあまり参考にできず，結局 ListView と同じ操作をしてしまったようである．つまり，ActionView で提示することができる奥行き情報は利用されなかった．原因としては，奥の方の項目はほぼ見ることができない，上の項目と下の項目との関連がわかりにくいなどのようである．項目が文字であるため，全ての部分が見えないと捜索の際の助けにはなりにくいようである．

また，視点検出の精度および処理の負荷から，項目選択の操作性が悪化し，かつ複雑な画面表示のために ListView よりも悪い結果となったと考えられる．

6.　おわりに

二つの実験を通して，ActionView において若干の情報提示面積の拡大による効果は得られた．今回の実験でのシステムでは，情報の 3 次元配置による奥行きの利用は実用までの効果は得られなかったが，より項目を見やすく工夫することにより，平面的に情報を配置するよりも多くの情報を配置することが可能であると考えられる．今後さらに提示する情報および，その提示方法について検討が必要である．

今回の実験では，システムがもうひとつ使いにくいという意見が多かったが，ActionView は使っていて面白いという声もあり，アトラクションとしておおいに可能性がある．

参考文献

[1] B.B.Mandelbrot: The fractal-based method for controlling information display, ACM Trans. On Information Systems, Vol13, No.3, pp.305〜323 (1995)

[2] Manojit Sarkar and Marc H.Brown: Graphical Fisheye views of graphs, CHI ’92 Conference Proceedings, pp.83〜91(1992)

[3] G.G.Robertson, J.D.Mackinlay and S.K.Card: ConeTrees: Animated 3D visualization of hierarchical information, Proceedings of the ACM Conferencre on Human Factors in Computing Systems (CHI ’91), pp.189〜194, ACM Press (1994)

[4] 塩澤他：「切り取り操作による柔軟な情報選択ができる WWW 視覚化」；情報処理学会 HI 研究報告資料 97-HI-72-11, pp.61〜66

[5] 椎尾：「Scroll Display：超小型情報機器のための指示装置」；情報処理学会 HI 研究報告資料 97-HI-71, pp.91〜98

[6] 増井他：「携帯端末に適した情報表示・操作方式の検討」；情報処理学会 HI 研究会資料 99-HI-82-2, pp.25〜30

[7] 加藤他：「携帯端末向け小画面表示/片手操作 UI の提案と試作」；情報処理学会 HI 研究会資料 2000-HI-91-4, pp.7〜12 (2000)

[8] 成田 他：「ユーザの視点による視野コントロールが可能な提示システム」；ヒューマンインタフェース学会研究報告集 Vol.3 No.3, pp.11〜14 (2001)