視線によるテキストウィンドウの自動スクロール

大和正武[†] 門田暁人[†] 高田義広[†,☆]
松本健一[†] 鳥居宏次[†]

本稿では，視線によるテキストウィンドウの自動スクロール方式を提案し，提案方式による自動スクロールが可能なテキストブラウザを用いた評価実験の結果について述べる．従来の視線を入力とするインタフェースでは，視線情報をポインティングデバイスの代替として利用しており，メニューやアイコンの選択のために一時的にではあるが作業対象から視線を外す必要があった．これに対して提案する方式では，ウィンドウ内での視線の移動に着目する．作業者はウィンドウ内で視線を移動させるだけで，表示したいテキスト部分をウィンドウ中央部に移動させ，表示させた状態でスクロールを停止させることができる．評価実験の結果から，テキストのスクロールにより文字列を検索するという作業（タスク）においては，視線による自動スクロールはキーボード操作によるスクロールと比較して同程度かそれ以上に有効であることが分かった．この場合，ウィンドウ中央から注視点までの垂直方向の距離をスクロール速度に比例させる制御方式が，他の方式に比べ優れていることが確認できた．

Scrolling the Text Windows by Looking

MASATAKE YAMATO,[†] AKITO MONDEN,[†] YOSHIHIRO TAKADA,[†,☆]
KEN-ICHI MATSUMOTO[†] and KOJI TORII[†]

This paper proposes a method and implementation for a system to automatically scroll a text window by tracking where a user is looking at. A user of the system can access a portion of the displayed document without using keyboard or mouse. Existing eye-tracking methods, which use eye-gaze as a substitute for pointing devices, require a user to "look at" menus and icons to manipulate a window display. In contrast, our proposed method uses eye movement within a window to control its display. A position of a user's gaze within a text window determines how to scroll the window, what portion of the text to display, and where to stop. Our user study demonstrates that the effectives of scrolling using this method is as good as one with conventional keyboard control as respects string search by scrolling. In particular, we have found that a controlling mechanism to determine the speed of scrolling in terms of the distance of the gaze point from the center of the window was both qualitatively and quantitatively superior to other control mechanisms in the string search by scrolling.

1. はじめに

計算機画面や画面内のウィンドウに何らかの情報を表示するソフトウェア（テキストエディタ，ウェブブラウザ等）の多くは，ユーザからの入力に応じて表示内容をスクロールする機能を持つ．スクロール機能は，同時に表示可能な情報量を増やすことはできないが，大きさや解像度に制限のある計算機画面やウィンドウで，より大量の情報を扱うことを可能にする．たとえば，テキストエディタが同時に表示可能な行数は，一般に，たかだか数十行であるが，数百行，数千行のプログラムファイルを編集することがスクロール機能により可能となっている．

表示内容をスクロールさせるには，通常，キーボードからスクロールのためのコマンドを入力したり，マウスを操作してスクロールバーのクリックやドラッグを行う必要がある．しかし，情報を読み書きしている最中にコマンド入力やマウス操作を頻繁に行うと，作業者の思考が中断され，作業効率の低下や作業者への高い負荷を招く恐れがある．ページ単位でのスクロールや指定した文字列の存在箇所までスクロールさせるコマンドによって，スクロール操作の回数や時間を短くすることは可能であるが，同時に，スクロール操作をより繁雑にする．

本稿では，キーボードやマウスによる操作を必ずし

† 奈良先端科学技術大学院大学情報科学研究科
Graduate School of Information Science, Nara Institute of Science and Technology
☆ 現在，オムロン株式会社
Presently with OMRON Corporation

も行わなくても，作業者が読み書きしたいと考える情報がウィンドウ内に表示されるインタフェースの実現を目指して，視線によるテキストウィンドウの自動スクロール方式を提案する．具体的には，視線追跡装置によって作業者の視線を計測し，視線とテキストウィンドウとの交点（注視点）の座標に基づいて縦スクロールの方向と速度を決定する．注視点がテキストウィンドウ上部にある場合には下方向へのスクロールを，下部にある場合には上方向へのスクロールを行う．また，スクロールの速度（あるいは加速度）は，ウインドウの垂直方向の中心と注視点との距離に比例させる．これにより，ウィンドウ内に読みたいと思うテキストが存在する場合はもちろんのこと，ウィンドウ外にテキストがある場合でも，作業者はウィンドウ内で視線を移動させるだけで，テキストをウィンドウ中央部に表示させ（移動させ），スクロールを停止させることができる．

キーボードやマウスの代わりとして視線を入力とするインタフェース（視線インタフェース）の研究はさかんに行われている[1)~3),5),6)]．その多くは視線によるメニュー選択に関するものであるが，Jacob[2)]は，視線によってテキストのスクロールを制御する方式を提案している．彼が提案する方式では，あらかじめ用意された特定のアイコンに視線が向けられるとスクロールが開始し，視線がテキストに戻るとスクロールは停止する．したがって，ポインティングデバイスによる入力に代わるものとして視線を利用しているといえる．アイコンへの視線の移動は，表示されているテキストから一時的にではあるが視線を外すことを意味し，作業効率を低下させる可能性が大きい．これに対して，本稿で提案する方式では，ウィンドウ内で視線を移動させるだけでスクロールを制御することができ，表示されているテキストから視線を外す必要はない．特に，テキストを先頭から順に読む作業においては，「読む」という動作に追従して自動的にテキストがスクロールすることになる．

本稿の以降の構成は次のとおりである．2章では，視線による自動スクロールの概略を示し，必要な機能を列挙する．3章では，ウィンドウ上の注視点座標をパラメータとして，スクロールの向きや速度を制御する具体的な方式を4つ提案する．4章では，視線による自動スクロール機能を持つシステムとして試作したテキストブラウザについて述べる．5章では，試作したテキストブラウザを用いて行った自動スクロール機能の評価実験について説明し，その結果を示す．最後に6章では，まとめと今後の課題を述べる．

2. 視線による自動スクロール

視線による自動スクロールとは，「作業者（コンピュータユーザ）が作業対象中の見たい，あるいは，読みたいと考える部分が表示用ソフトウェアの情報表示部（ウィンドウ）の中心に表示されるよう，作業者の注視点の座標に基づいて，表示用ソフトウェアのスクロール制御を自動的に行う」ことである．ここで，「注視点」とは作業者の視線とウィンドウとの交点を意味する．「スクロール制御」とは，スクロールの方向，および，速度を決定することである．また，「作業対象」としては，プログラムリスト，ワープロ文書，画像等を，「表示用ソフトウェア」としては，テキストエディタ，ウェブブラウザ等が考えられる．

本稿では，簡単のため，作業対象としてプログラムリストを，表示用ソフトウェアとしてはテキストのみ表示可能なソフトウェア（テキストブラウザ）を，それぞれ考える．また，スクロールの方向は，縦方向のみを考える．したがって，ウィンドウの中心とは，テキストブラウザの情報表示部（テキストウィンドウ）の垂直方向の中心線（以降では，単に「中心線」と呼ぶ）であり，この中心線と注視点の距離，すなわち，注視点の垂直座標の値（変位）によってスクロールの方向と速度が決定されることになる．

視線を向けるという自然な動作によって，見たい，あるいは，読みたいと考えるテキスト（目標テキスト）をウィンドウの中心へと移動させるためには，注視点が中心線より上部にある場合には下方向に，下部にある場合には上方向に，それぞれスクロールさせればよい（**図1**参照）．目標テキストがウィンドウ内にある場合，目標テキストに作業者が視線を向けるとスクロールによって目標テキストは中心線に近づき，それを作業者が目で追うことにより注視点も中心線に近づく．比較的短い時間で，目標テキストと注視点がともに中心線上に位置し，スクロールは停止する．目標テキストがウィンドウ外にある場合，（スクロールが停止しないように）中心線より少し離れたウィンドウの上部，あるいは，下部に作業者が注視点をとどめておけば，スクロールによって目標テキストがウィンドウ内に現れる．現れた目標テキストを作業者が見つけ，視線を向け続ければ，目標テキストと注視点はやはり中心線上に移動し，スクロールは停止する．

「視線による自動スクロール」の利用者からすれば，次のように視線を移動するだけで，見たい，あるいは，読みたいと考えるテキストがウィンドウ中央に表示されることになる．

- 目標テキストをウィンドウ内に見つけた場合には，目標テキストに視線を向ける．
- 目標テキストがウィンドウ内に見つからない場合は，目標テキストがウィンドウ内に現れるまで，ウィンドウの上部，または，下部に視線をとどめる．

「視線による自動スクロール」を実現するためには，次の4つの機能が必要となる（図**2** 参照）．

(1) 視線追跡機能：作業者の視線の向きを精密に測定する．作業者の頭部を固定しない場合には，頭部の向きや位置も測定する必要がある．スクロールを円滑に行い，停止させるためには，測定誤差が1～2度以下で，1秒あたり10回程度の測定性能は必要である．

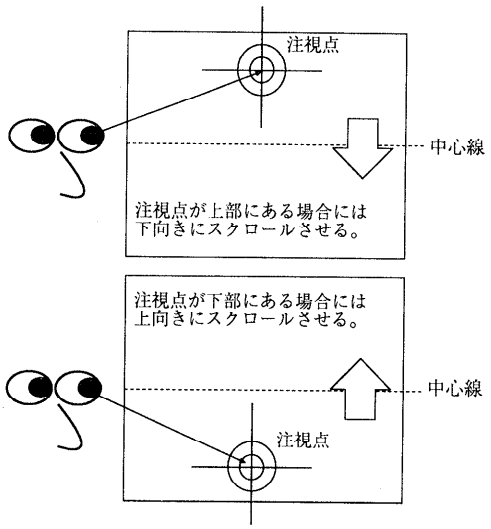


図1 視線による自動スクロール
Fig. 1 Scrolling windows by looking.

(2) 注視点算出機能：ウィンドウの位置，および，「(1) 視線追跡機能」により得られた視線の向きと作業者の頭部の向きや位置から，注視点を算出する．

(3) スクロール制御機能：「(2) 注視点算出機能」により得られた注視点の座標に基づいて，スクロールの向きと速度を決定する．

(4) 表示機能：作業対象を表示し，「(3) スクロール制御機能」で決定されたスクロールの向きと速度に基づいて実際にスクロールを行う．

3. スクロール制御方式

2章で述べたように，視線を向ける目標テキストをウィンドウの中心へと移動させるためには，注視点が中心線より上部にある場合には下方向に，下部にある場合には上方向に，それぞれスクロールさせればよい．

スクロール速度の制御では，

(1) 目標テキストのウィンドウの中心への迅速な移動
(2) スクロールのスムーズな停止

の2点が重要である．本稿では，(1) を実現する制御方式として「速度方式」と「加速度方式」の2方式を提案する．さらに，(2) を実現するために「速度3分割方式」と「加速度3分割方式」の2方式を追加し，合計4つの制御方式を提案する（**表1** 参照）．なお，名称に統一性を持たせるため，以降では「速度方式」を「速度2分割方式」と，「加速度方式」を「加速度2分割方式」と，それぞれ呼ぶ．

3.1 速度2分割方式と加速度2分割方式

目標テキストをウィンドウの中心へすばやく移動させるには，中心線と注視点の距離，すなわち，注視点の垂直座標の値（変位）が大きくなるほど，スクロール速度を大きくすればよい（**図3** 参照）．「速度2分

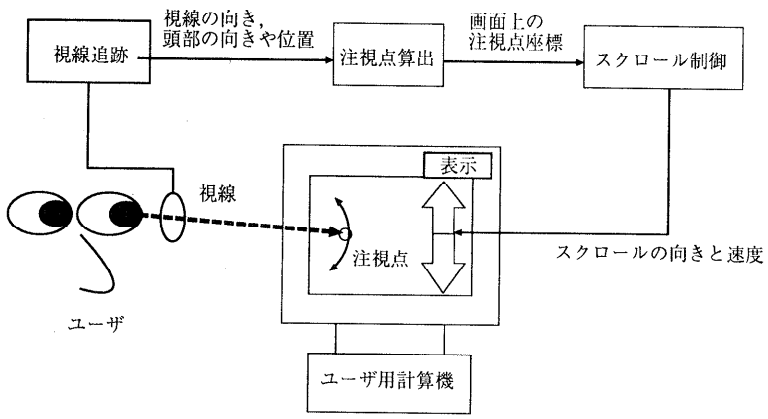


図2 自動スクロールに必要な機能
Fig. 2 Functions for automatic scrolling.

表1 スクロール速度の計算方式
Table 1 Calculation methods for speed of scrolling.

方式	計算式	試作したテキストブラウザでの値
速度 2分割	$v(t) = -M_v e(t)$	$M_v = 3$
速度 3分割	$v(t) = \begin{cases} -M_v\left(e(t)+\frac{1}{n}\right) & e(t) < -\frac{1}{n} \\ -M_v\left(e(t)-\frac{1}{n}\right) & e(t) > \frac{1}{n} \\ 0 & \text{その他} \end{cases}$	$M_v = 6$, $n = 6$
加速度 2分割	$v'(t) = -M_a e(t) - Rv(t)$	$M_a = 3$, $R = 1$
加速度 3分割	$v'(t) = \begin{cases} -M_a\left(e(t)+\frac{1}{n}\right) - Rv(t) & e(t) < -\frac{1}{n} \\ -M_a\left(e(t)-\frac{1}{n}\right) - Rv(t) & e(t) > \frac{1}{n} \\ -Rv(t) & \text{その他} \end{cases}$	$M_a = 6$, $R = 1$, $n = 6$

$e(t)$：時刻 t における注視点の変位（単位はページ）.
$v(t)$：時刻 t におけるスクロールの速度（単位はページ/秒）.
$v'(t)$：時刻 t におけるスクロールの加速度（単位はページ/秒^2）.
M_v：最大速度を調整するための比例定数.
M_a：最大加速度を調整するための比例定数.
R：抵抗（摩擦）の大きさを表す定数（単位は1/秒）.
n：中心線付近の領域の幅を表す定数（単位はページ）.

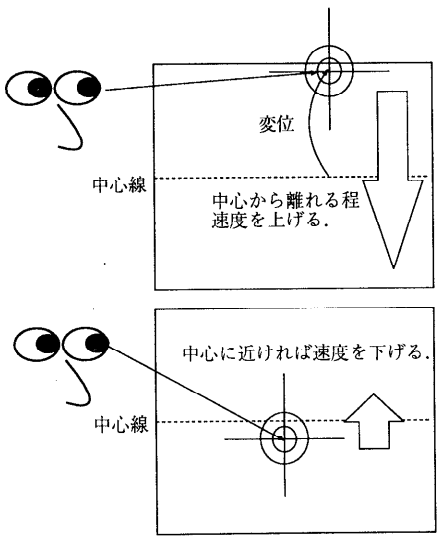


図3 変位によるスクロール速度の制御
Fig. 3 Controlling the speed of scrolling based on distance.

割方式」は，スクロール速度を変位に比例させる制御方式である．スクロール速度の算出式は次のとおりである．

$$v(t) = -M_v e(t) \qquad (1)$$

ここで，$e(t)$ は時刻 t における注視点の変位（単位はページ）を，$v(t)$ は時刻 t におけるスクロール速度（単位はページ/秒）を，それぞれ表す．ページとはテキストウィンドウの縦方向の長さである．また，M_v は最大速度を調整するための比例定数である．なお，中心線よりウィンドウ下方向の変位や速度は正の値で，上方向の変位や速度は負の値で，それぞれ表すものとする．

「速度2分割方式」では，注視点が大きく移動すると，スクロール速度もそれに比例して大きく変化し，作業者の目がスクロールに追従できない可能性がある．変位をスクロールの加速度に比例させれば，スクロール速度の変化を緩やかにすることができる．ただし，スクロールの加速度を単に変位の定数倍にすると，注視点が中心線をまたがず，変位が正，または，負の値のままでいる間は速度の絶対値は単調に増加し続け，やがて，作業者の目がスクロールに追従できなくなる．「加速度2分割方式」では，スクロールの加速度を変位に比例させるとともに，速度に比例した抵抗（摩擦）を表す項により，スクロール速度の絶対値が際限なく大きくなるのを防ぐ．「加速度2分割方式」におけるスクロール速度の算出式は次のとおりである．

$$v'(t) = -M_a e(t) - Rv(t) \qquad (2)$$

ここで，$e(t)$，$v(t)$ は式(1)と同じである．また，M_a は最大加速度を調整するための比例定数である．R は抵抗（摩擦）の大きさを表す定数である．

3.2 速度3分割方式と加速度3分割方式

「速度2分割方式」では，変位が0とならない限り，すなわち，ウィンドウの中心線に作業者が正確に視線を向けない限り，スクロールを停止させることはできない．「加速度2分割方式」では視線をさらに微妙に移動させなければ，速度，加速度をともに0とすることができず，スクロールは停止しない．このことは，スクロールにおける新たな負荷を作業者に課す可能性があるばかりでなく，視線の測定精度が悪ければ，視線による自動スクロールそのものが実現困難となる．

そこで，ウィンドウを3つの領域に分割し，中心線付近の領域では，変位による速度や加速度を0とする方式（3分割方式）を提案する（**図4** 参照）．具体的には，閾値を設定し，変位の絶対値が閾値以下であれば，変位による速度や加速度を0とする．変位を速度に比例させる「速度3分割方式」では，スクロール速度を次式により算出する．

$$v(t)=\begin{cases} -M_v\left(e(t)+\dfrac{1}{n}\right) & e(t)<-\dfrac{1}{n} \\ -M_v\left(e(t)-\dfrac{1}{n}\right) & e(t)>\dfrac{1}{n} \\ 0 & \text{その他} \end{cases} \quad (3)$$

ここで，$e(t)$，$v(t)$，M_v は式 (1) と同じである．また，n は中心線付近の領域の幅を表す定数である．ここでは，中心線付近の領域の幅は中心線の上下で同じとしている．

同様に，変位を加速度に比例させる「加速度3分割方式」では，スクロール速度を次式により算出する．

$$v'(t)=\begin{cases} -M_a\left(e(t)+\dfrac{1}{n}\right)-Rv(t) & e(t)<-\dfrac{1}{n} \\ -M_a\left(e(t)-\dfrac{1}{n}\right)-Rv(t) & e(t)>\dfrac{1}{n} \\ -Rv(t) & \text{その他} \end{cases} \quad (4)$$

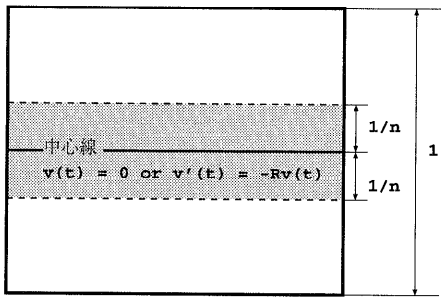


図4 ウインドウ領域の3分割
Fig. 4 Three regions of window.

ここで，$e(t)$，$v(t)$，M_a，R は式 (2) と同じである．また，n は式 (3) と同じである．

4. 自動スクロール機能を持つテキストブラウザ

試作したテキストブラウザでは，2章で示した，「視線による自動スクロール」に必要な4つの機能を次のように実現している[7]．

(1) 視線追跡機能

作業者の視線の向き，および，頭部の向きや位置の測定には，米国 Applied Science Laboratories 社製の Eye Tracker 4100H と Magnetic Head Tracker を用いる．Eye Tracker 4100H は，作業者の頭部に装着するヘルメット型の部分と据置き型の部分とからなる（**図5** 参照）．測定誤差は1〜2度以下で，1秒あたり10回程度の測定が可能である．

(2) 注視点算出機能

注視点の算出はソフトウェアで行う．ソフトウェアは Linux 上で C++で開発した．Ethernet を介して実時間で送られてくる測定データを受け取り，注視点を算出し，算出結果を Ethernet を介して実時間で送信することができる．注視点の算出精度は，作業者にも依存するが，平均誤差 10 mm 前後である．なお，注視点の算出精度を高めるためのキャリブレーション用ソフトウェアもあわせて開発した．

(3) スクロール制御機能

スクロール制御はソフトウェアで行う．ソフトウェアは Linux 上で C++で開発した．3章で述べた4つの方式での制御が可能である．スクロール速度や加速度の計算式中の定数値としては，表1 の最右列に示

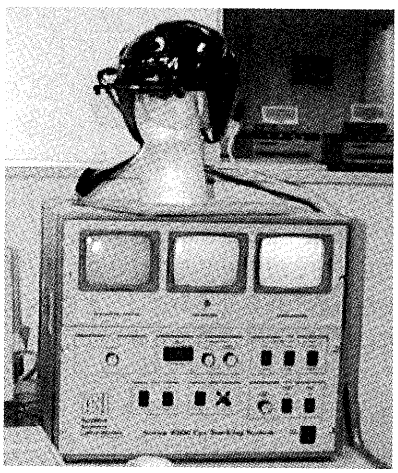


図5 視線追跡装置
Fig. 5 Eye tracking device.

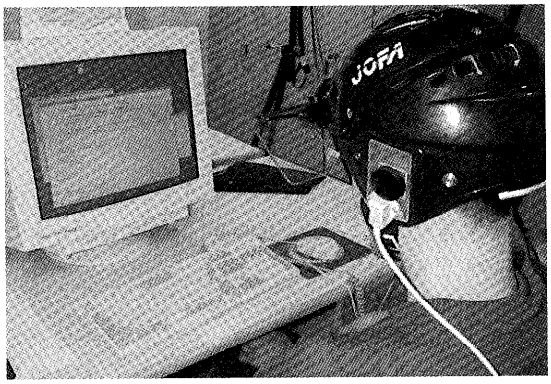


図6 試作したテキストブラウザの利用の様子
Fig. 6 Prototype of Text Browser.

す値を用いている．特に，「3分割方式」では，変位による速度や加速度を0とする領域の幅を $2n = 1/3$（ページ）とし，また，ウィンドウの上端，あるいは，下端から中央に向かって1/6ページの位置に注視点があるときに，変位による速度が1ページ/秒に，加速度が1ページ/秒^{2}となる．

(4) 表示機能

表示機能は，テキストブラウザとしてX-Windowを用いて実装した．テキストブラウザはテキストのみ表示可能で，表示部（テキストウィンドウ）の大きさは横80文字，縦25行である．スクロールは縦方向のみ可能であるが，スクロール制御ソフトウェアからの指示を受け取り，1ドット単位でテキストをスクロールさせることができる．なお，スクロール操作は，キーボードからも可能である．Eye Tracker 4100Hを装着してテキストブラウザを利用している様子を図**6**に示す．

5. 評価実験

5.1 概　　要

「視線による自動スクロール」の有効性を確かめるため，4章で述べたテキストブラウザを用いて，スクロール方式の比較実験を行った．実験では，スクロールを頻繁に行う必要のある作業（タスク）を，テキストブラウザを用いて被験者に行ってもらい，被験者へのインタビュや実験データに基づいて，スクロール方式間での使用感や作業効率の違いを評価した．

被験者には次のような指示を与えた．

- 与えられたCプログラムをテキストブラウザ上で読み，プログラム中に含まれているすべてのprintf文をできるだけ早く発見してください．
- printf文を発見するごとに，その旨を声を出して実験実施者に報告してください．

表2 タスクで用いたプログラム
Table 2 Program files used in tasks.

プログラム	P1	P2	P3	P4	P5
行数（LOC）	236	252	255	218	271
printf文の個数	2	3	2	3	3

- すべてのprintf文を発見したと思ったら，作業の終了を声に出して宣言してください．

なお，与えられたCプログラムに含まれているprintf文の個数は，タスク開始時に被験者に知らせておいた．また被験者が作業の終了を宣言した時点で，タスクは終了したと見なした．

タスクで用いるCプログラムとして，P1，P2，P3，P4，P5の5つを用意した．各プログラムの行数と含まれるprintf文の個数を**表2**に示す．表2からも分かるように，用意したプログラムの行数は，実験で用いたテキストブラウザが一度に表示できる行数（25行）のおよそ10倍となっている．また，含まれるprintf文の個数が2，3個と少ないため，被験者はテキストブラウザのスクロール機能を使い，注意深くプログラムを見ていく必要がある．

被験者は5人（A，B，C，D，E）で，いずれも奈良先端科学技術大学院大学の教官または学生である．被験者はすべて，C言語によるプログラムの読み書きができ，UNIXのテキストエディタ（muleやvi等）を日常的に使用している．したがって，テキストブラウザやそのスクロール機能，タスク，タスクで用いるCプログラム等については，簡単な説明と練習だけで，被験者はそれらを正しく理解することができた．

5.2 タスク実行

タスク実行に先立って，視線追跡装置のキャリブレーションと練習タスクを，被験者ごとに行った．キャリブレーションは筆者らが開発したソフトウェアを用いて，ウィンドウ上での注視点の平均測定誤差が20 mm未満になるまで行った．練習タスクは，テキストブラウザやスクロール方式に被験者が慣れ，被験者にタスクを正しく理解してもらうために行った．練習タスクで用いたプログラムの行数は267行，含まれるprintf文は5個であった．なお，必要に応じて練習タスクは繰り返し行い，練習タスク後に再度キャリブレーションを実施した．

キャリブレーションと練習タスクを終えた5人の被験者にはそれぞれ，P1～P5を対象として計5回のタスクを行ってもらった．ただし，各タスクは異なるスクロール方式で行ってもらった．すなわち，「速度2分割方式による自動スクロール」，「速度3分割による自動スクロール」，「加速度2分割方式による自動

スクロール」，「加速度3分割による自動スクロール」，そして，「キーボード操作によるスクロール」の5方式でそれぞれ1回ずつタスクを行ってもらった．被験者がタスクで用いたスクロール方式とプログラムの対応関係を**表3**に示す．表3からも分かるように，同一プログラムに同一スクロール方式が2回以上用いられないように，また，同一プログラムに同一被験者が2回以上タスクを行わないように割り当てた．

各タスクにおいては，タスク終了までに要した時間（タスク時間），および，誤りの発生回数を計測した．ここで，タスクにおける誤りの発生とは，スクロールによってウィンドウ内に現れたprintf文に被験者が気づかず，視線が向けられないままウィンドウ外に移動した場合を指す．また，すべてのタスク終了後，スクロール方式の違いによる使用感や作業効率の差，タスク中に発生した誤りの原因等について，被験者へのインタビュを行った．

5.3 実験結果

被験者へのインタビュの結果を**表4**にまとめる．表4の評価項目のうち（2），（3）は，インタビュの中で被験者が述べた感想において，スクロール方式によってその優劣がはっきりと分かれた項目を列挙している．表中の○は比較的優れていることを，×は比較的劣っていることを，それぞれ表す．ただし，図6に示したように，実験に用いた視線追跡装置はヘルメット型であり，自動スクロールのために日常的に使用するのは現実的ではない．インタビュにおいては，「視線追跡装置が小型軽量化され，装着感もほとんどなく，作業者の動作も制限しない」という状況を仮定したうえで回答してもらった．各タスクで計測したタスク時間と誤り発生回数を**表5**にまとめる．表5において，＊はタスク中に誤りが発生したことを表す．いずれのタスクおいてもたかだか1回の誤りしか発生しなかった．以上の実験データに基づき，以下では「視線による自動スクロール」の有効性を定性的，および，定量的に評価する．

表3 被験者への対象プログラムとスクロール方式の割当て
Table 3 Assignment of program files and scrolling methods to each subject.

スクロール制御方式	被験者				
	A	B	C	D	E
速度2分割	P5	P2	P4	P3	P1
速度3分割	P2	P3	P1	P5	P4
加速度2分割	P3	P4	P2	P1	P5
加速度3分割	P4	P1	P5	P2	P3
キーボード	P1	P5	P3	P4	P2

(1) 定性的評価

表4の評価項目（1）の結果から分かるように，スクロールの制御方式にかかわらず，「視線による自動スクロール」は，「キーボード操作によるスクロール」と同程度かそれ以上の作業効率が得られると感じる被験者がほとんどであった．特に，速度2分割方式が最も使いやすいという意見が多かった．

スクロールの制御方式による違いを見てみると，まず，評価項目（2）の結果から分かるように，加速度方式は視線移動に対する反応が悪いと感じる被験者が多かった．これは，加速度方式が，スクロール速度の変化を緩やかにする反面，スクロール開始時にはスクロール速度を徐々にしか大きくできないことに起因し

表4 スクロールの計算方式の比較
Table 4 Comparison among the methods of calculating scroll speed.

評価項目	スクロール制御方式			
	速度2分割	速度3分割	加速度2分割	加速度3分割
(1) キーボード操作と比較した作業効率	○	○	○	○
(2) 反応の良さ	○	○	×	×
(3) 速度変化の自然さ	○	×	○	×

表5 実験データ
Table 5 Data of experiments.

方式	タスク時間（秒）						誤り回数
	A	B	C	D	E	平均	
速度2分割	27	38	28	45	45*	36.6	1
速度3分割	73*	61*	35	24*	36*	45.8	4
加速度2分割	32	37	75*	36	50	46.0	1
加速度3分割	30	87*	33	70*	36	51.2	2
キーボード	55	59	44	36	95	57.8	0

（「*」は，誤りが1回発生したことを示す．）

ていると思われる．

評価項目（3）の結果からは，3分割方式においてスクロール速度の変化が不自然であると感じる被験者が多かったことが分かる．これは，3.2節で述べたように，スクロールの停止を容易にするために，変位による速度や加速度を0とする領域を中心線付近に設定したためと考えられる．すなわち，「視線の移動」が必ずしも「スクロール速度の変化」とならないことが，被験者をかえって混乱させる結果になったと思われる．

(2) 定量的評価

表5のデータを用いて，スクロール制御方式間におけるタスク時間の母平均の差を統計的に検定した．その結果，「速度2分割法による自動スクロール」のタスク時間のみが「キーボード操作によるスクロール」のタスク時間より小さいと有意水準5%でいえることが分かった．また，その他の制御方式による自動スクロールのタスク時間については，「キーボード操作によるスクロール」のタスク時間と有意な差はないことが分かった．これらの結果は，文字列検索における被験者の定性的評価と一致する．すなわち，前述の「スクロールの制御方式にかかわらず，視線による自動スクロールは，キーボード操作によるスクロールと同程度かそれ以上の作業効率が得られた」，および，「自動スクロールでは速度2分割法が最も使いやすかった」と一致する．

タスク中の誤り回数についても同様に検定した結果，「速度3分割法による自動スクロール」の誤り回数のみが「キーボード操作によるスクロール」の誤り回数より大きいと有意水準5%でいえることが分かった．また，その他の制御方式による自動スクロールの誤り回数については，「キーボード操作によるスクロール」の誤り回数と有意な差はないことが分かった．

スクロールのスムーズな停止を考慮した速度3分割法では，タスク実行により多くの時間を要し誤りが多発した．その原因は，中心付近に停止の領域を設けた分，スクロール速度を加減するための領域が狭くなりスクロール制御が困難になった点にあると思われる．しかも本実験では文字列検索をタスクとしており，スクロールをスムーズに停止させる必要がほとんどなかった．「プログラム中の誤り（バグ）の内容を調べる」といった，目標テキストが必ずしも明確でなく，テキストを熟読する必要のあるタスクでは，速度3分割法の優位性が示される可能性がある．また3つの領域やその境界を提示することにより，速度3分割法におけるスクロール速度の変化の不自然さが解決されタスク時間や誤り回数が改善されるかもしれない．

以上の評価結果をまとめると，視線追跡装置を装着することそのものの負荷を無視し，文字列検索というタスクに限定すると，視線による自動スクロールは，キーボード操作によるスクロールと比較して同程度かそれ以上に有効であるといえる．同様に，文字列検索というタスクに限定した場合，4つのスクロール制御方式の中では速度2分割方式が定性的にも定量的にも優れているといえる．

6. まとめ

本稿では，キーボードやマウスによる操作を必ずしも行わなくても，コンピュータユーザが読み書きしたいと考える情報がウィンドウ内に表示されるインタフェースの実現を目指して，視線によるテキストウィンドウの自動スクロール方式を提案した．また，4つのスクロール制御方式を考案し，それらの制御方式による自動スクロールが可能な「テキストブラウザ」を試作した．さらに，試作したテキストブラウザを用いた実験を行い，視線によるテキストウィンドウの自動スクロールの有効性を，主にキーボード操作によるスクロールとの比較により，定性的，および，定量的に示した．特に，文字列検索というタスクに限定すると速度2分割方式は，統計的検定によっても，キーボード操作との有意な差が認められ，考案した4つのスクロール制御方式の中で最も優れていることが分かった．

今回得られた結果は，キーボードやマウスに代わるポインティングデバイスとして，視線の利用範囲が広がったことを意味する．1つは，キーボードやマウスの操作が身体的に困難なユーザへの適用である．すなわち，文字どおり，キーボードやマウスの代わりにポインティングデバイスとして視線を利用し，場合によっては，キーボードやマウスをまったく使わないことも視野に入れた利用形態である．もう1つは，キーボードやマウスと視線を併用し，コンピュータ操作を分担する利用形態である．

たとえば，マルチウィンドウ環境において，あるウィンドウ上の長文テキストをスクロールによって参照しながら，それとは別のウィンドウ上でテキストを入力する，といった作業を行いたいとする．従来のキーボードとマウスだけのインタフェースでは，テキストのスクロールと入力を同時に行うことはできない．すなわち，キーボード操作でスクロールを行う場合には，テキスト入力を中断する必要がある．マウス操作でスクロールを行う場合でも，スクロールさせるウィンドウをマウスで選択する必要があり，別ウィンドウでのテキスト入力はできない．これに対して，キーボードで

テキスト入力を，視線でスクロールを，といった分担が可能になれば，スクロールのためにテキスト入力を中断する必要がなくなり，「読みながら書く」といった作業を円滑に行うことができるようになる可能性がある．

別の例として，描画ソフトウェアにおいて表示ウィンドウには収まらない大きな絵を描く作業を行いたいとする．マウスのみを用いて描画する場合，マウスカーソルが表示ウィンドウの端に到達するたびに，作業を中断してスクロールバーにマウスを移動させる必要がある．もし，視線によるスクロールが可能であれば，マウスでは描画作業のみを行えばよく，作業の中断も起こらない．

ただし，キーボードやマウスと視線の併用を具体化するためには，文献4)で指摘されている「視線インタフェースにおける選択過程と取得過程の混在の問題」を解決する必要がある．前述の例では，「ウィンドウを選択する視線の動き」と「テキストをスクロールさせながら読むための視線の動き」，および，「絵を描くためにマウスカーソルを追尾する視線の動き」と「スクロールさせるための視線の動き」を区別する技術の開発が必要となる．また，デバイスごとの操作分担が可能となるように，表示用ソフトウェアやウィンドウシステムを新たに開発したり，改良したりする必要がある．さらに，操作分担が可能なユーザインタフェースが準備できても，操作分担がかえってユーザを混乱させる可能性もある．今後は，ユーザによる作業や操作分担のモデル化を含め，操作分担を効率良く，しかも，ユーザにとって使いやすいものとするための検討を進めていく予定である．

参考文献

1) 江崎信也，海老澤嘉伸，杉岡 明，小西 学：ビデオ式注視点検出によるコミュニケータによる瞬きを用いた高速メニュー選択法，1997年電子情報通信学会情報・システムソサイエティ大会講演論文集 (1997).
2) Jacob, R.: What you look at is what you get: Eye movement-based interaction techniques, *Proc. ACM CHI '90 Human Factors in Computing System Conference*, pp.11–18 (1990).
3) 大野建彦：階層メニュー選択における視線の利用，情報処理学会研究報告，ヒューマンインタフェース 71-13 (1997).
4) 大野建彦：視線インタフェースにおける選択過程と取得過程の識別，日本ソフトウェア科学会WISS '97，尾内理紀夫（編），インタラクティブシステムとソフトウェア V (1997).
5) 高木啓伸：視線を用いたユーザインタフェース開発：EyeTk，情報処理学会研究報告，ヒューマンインタフェース 65-15 (1996).
6) 高木啓伸：視線からのユーザ情報の解析—翻訳支援システムを題材として，日本ソフトウェア科学会WISS '97，尾内理紀夫（編），インタラクティブシステムとソフトウェア V (1997).
7) 大和正武，高田義広，鳥居宏次：視線追跡によりプログラムエディタを自動スクロールさせる方式の比較，電子情報通信学会技術研究報告，Vol.SS97-16 (1997).

(平成10年 6 月 8 日受付)
(平成10年12月 7 日採録)

大和 正武

平成9年立命館大学理工学部情報工学科卒業．現在，奈良先端科学技術大学院大学博士前期課程に在学中．ユーザインタフェースの研究に従事．

門田 暁人（正会員）

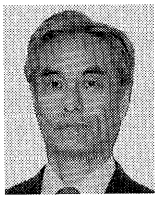
平成6年名古屋大学工学部電気学科卒業，平成10年奈良先端科学技術大学院大学博士課程修了．同年奈良先端科学技術大学院大学助手．博士（工学）．プログラム理解の分野に興味を持つ．

高田 義広（正会員）

平成元年大阪大学基礎工学部情報工学科卒業，平成6年同大学大学院博士課程修了，同年奈良先端科学技術大学院大学助手．平成10年オムロン（株）入社．工学博士．ソフトウェア開発における人的要因の研究に従事．電子情報通信学会，日本音響学会各会員．

松本 健一（正会員）

昭和60年大阪大学基礎工学部情報工学科卒業．平成元年同大学大学院博士課程中退．同年同大学基礎工学部情報工学科助手．平成5年奈良先端科学技術大学院大学助教授．工学博士．ソフトウェアにおける計測環境，ソフトウェア品質保証の枠組に関する研究に従事．電子情報通信学会，IEEE 各会員．



鳥居　宏次（正会員）

昭和37年大阪大学工学部通信工学科卒業．昭和42年同大学大学院博士課程修了．同年電気試験所（現電子技術総合研究所）入所．昭和50年ソフトウェア部言語処理研究室室長．昭和59年大阪大学基礎工学部情報工学科教授．平成4年奈良先端科学技術大学院大学教授．工学博士．ソフトウェア工学の研究に従事．電子情報通信学会，日本ソフトウェア学会，人工知能学会，ACM，IEEE各会員．