ヒューマンインタフェース 65－15
音声言語情報処理 10－15
（1996. 3. 1）

視線を用いたユーザインタフェース開発環境: EyeTk

高木 啓伸

hironobu@is.s.u-tokyo.ac.jp
東京大学理学系研究科情報科学専攻
東京都文京区本郷 7-3-1

すばやく移動させることができる視線は、入力デバイスとして有望である。そこで、これまでに視線を入力デバイスとして利用する手法が開発されてきたが成功した用途は障碍者用インタフェースなどに限られている。その理由の一つとして入力である視線と入力ではない視線の判別が難しい点が指摘されている。そこでこの問題点を明確にし、判別する手法の開発とそのための実験をおこなう必要がある。また、判別をする必要のない新たな視線の利用法の開発をおこなう必要がある。しかし、このような開発や実験をおこなうのは困難な作業であった。本研究ではこのような実験や開発を支援する環境として EyeTk を開発した。また、EyeTk を利用してサンプルアプリケーションの開発、視線に関する実験を行ない、EyeTk が有効であることを確認した。

Development Environment of CHI with Eye-movements: EyeTk

Hironobu TAKAGI

hironobu@is.s.u-tokyo.ac.jp
Department of Information Science, Faculty of Science, the University of Tokyo
3-1 Hongo, 7-chome, Bunkyo-ku, Tokyo, 113 Japan

Eye-movements are promising as an input device, and many efforts have been made to develop methods using eye-movements. These approaches succeeded, however, only in such domains as for disabled people. For one reason, it was pointed out that distinction between eye-movements for input and not for input was difficult, it was necessary to develop methods for this distinction, but experiments for developing algorithms was hard to execute. In this research, developing sample applications and experimenting on eye-movements, I confirmed that the EyeTk is useful.

1 はじめに

視線には入力デバイスとして次のような利点がある。

1) すばやく移動させることができる。

2) 高速に、長時間動かしても疲労しない。

3) 両手がふさがっていても入力できる。

この特徴を生かして、視線を入力デバイスとして利用する手法が開発されてきた([Jacob 90],[Hansen 95])。

それらは、通常の入力デバイスを利用できない環境、たとえば四肢に障碍を持つ方が利用する場合や、他に作業をおこなっていて視線以外を用いることができないような場合には有効であったが、一般の入力デバイスとしてはさまざまな問題があった。

その問題の一つとして入力である視線と入力ではない視線の判別が難しい点があげられる。

たとえば画面上のボタンを一定時間（判定時間と呼ぶ）注視することでそのボタンを「押す」ことができるというアプリケーションを考える。いま、ボタンを「見る」ための短い注視が入力であると判定されてボタンが押されてしまったとする。正しく判定をするためには判定時間を長くすればよいが、その場合ユーザは「押す」ために長い時間一点を注視しなければならず疲労してしまう。逆に、入力である長い注視が入力であると判定できなかった場合に判定時間を短くすると、入力ではない注視が入力と判定されてしまい、ユーザに負担をかけてしまう。

このように、ある視線が入力であると判定する基準を厳しくする（例では判定時間を長くする）と不自然な視線（例では長い注視）をユーザに強いることになり、緩くする（例では判定時間を短くする）と入力ではない視線が入力であると判定されてしまい、どちらもユーザに負担をかけることになる。

この問題を解決するためにはユーザに負担をかけないで、入力であるどうかを判別する手法の開発を行なう必要があり、さまざまな手法についてユーザへの負担を測定し評価しなければならない。そのため、多くの実験をおこなわねばならない。また、判定をするのでなく判定を必要としない新たな視線の利用手法の開発を行なう必要がある。しかし、このような開発や実験をおこなうのは困難な作業であった。

本研究では判定手法や新たな視線の利用手法の開発とそのための実験を支援する環境としてEyeTkを開発した。EyeTkはシステムの一部としてTcl/Tkインタプリタをもち(図1)、視線で操作可能なGUI (Graphical User Interface) アプリケーションをTcl/Tkスクリプトで容易に開発することができる。また、操作過程の視線を保存、再生、解析する機能をもっており実験の各ステップを支援することができる。EyeTkを用いてサンプルプログラムの開発、視線に関する実験をおこないEyeTkが有効であることを確かめた。

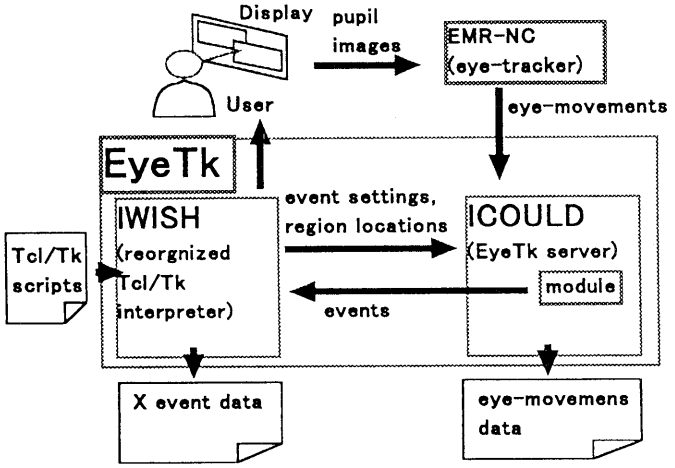

図1. EyeTkシステム

2 関連研究

視線を入力デバイスとして利用する場合、もっとも考慮すべき問題として“Midas touch problem”が指摘されている[Jacob 90]。Midasとは手で触れるものすべてを黄金に変えてしまう力を持つ神話上の怪物のことである。同じ視線が「入力」としての意味と「見る」意味とを持つことによる問題点をMidasの手に例えてこう呼んでいる。前述した入力の判定に関する問題はこの問題の一部であると考えている。

Jacob[Jacob 90]らの研究ではこの問題に対処するために、他の入力デバイスと組み合わせる手法を提案している。そのためのシステムとして、視線を入力として利用することができるユーザインタフェース管理システムを開発している。また、Hansen[Hansen 95]は、入力の判定問題に対して、判定する前にユーザに知らせる手法を提案している。これはボタンへの注視時間を絵の変化としてユーザに提示するという手法である。これらの研究では、インタフェーのを開発する過程での実験支援が考慮されていない点で本研究と異なっている。

3 EyeTk

3.1 EyeTkの概要

次の2点を目標にしてEyeTkを開発した。

1) 入力の判定手法の開発とそのための実験を支援する環境

2) 新たな視線の利用法の開発支援と実装のための基盤

1)の実験は対象となる判定手法を用いたアプリケーションを構築し、その時の視線を分析するという順序で進める予定である。そこで、次の機能が必要である。

a) 対象となる判定手法を組み込んだGUIアプリケーションを構築する機能

b) 実験の各ステップ（刺激提示、データ保存、再生、解析）で実験者を支援する機能

また、2)を実現するためには次の機能が必要である。

c) 新たな手法に関する実験を行なうための、上記b)の機能

d) 新たな手法で実際に動作するシステムを構築するための基盤としての機能

a)の要求に対して、Tcl/Tkインタプリタを拡張し、視線で操作できるGUIアプリケーションを開発する機能を実装した(図2、3.2節)。
また、b),c)の要求に対して、次の機能を実装した(3.3節)。

- Tcl/Tkを用いて刺激提示をおこなう機能
- 視線データ、X windowのイベントデータを保存する機能
- 保存したデータを用いて実験の再生をおこなう機能
- 画面の一部に領域を設定しその領域へ注目時間を測定する機能

最後のd)の要求に対して、複雑な視線の解析を必要とする手法にも対応できるように、視線データ解析部とインタフェース管理部を分離して実装した(3.4節)。

3.2 GUIアプリケーション構築機能

本節ではEyeTkを用いてGUIアプリケーションを構築する方法を、視線入力キーボードの例を用いて説明する。

構築した視線入力キーボードは視線を用いてアルファベットを入力することができるアプリケーションである。画面上（図2）のボタンを一定時間注視することでその文字を入力することができる

以下にプログラムの主要部(Aボタンを表示する部分)とその解説を示す。フォント、色、大きさの指定は省略した。

プログラム

```
1: text .textfield
2: button .button_A -text "A"
3: eye bind "enter .frame1.button_A 0.5sec"
        {
            .textfield insert end A;
            .button_A flash;
        }
4: button .end -text "END"
5: eye bind "fix .end 0.5sec"
        {
            eye stop;
            exit;
        }
6: pack .button_A .end .textfield
7: eye start
```

解説

1: アルファベットを入力する文字領域の指定。
2: A ボタンの指定。
3: 視線で A ボタンを「押す」ための指定。A ボタンを 0.5 秒間注目すると文字領域に "A" が入力される。
4: 終了ボタンの指定。
5: 視線で A ボタンを「押す」ための指定。終了ボタンを 0.5 秒間注目すると視線の計測をやめ、アプリケーションを終了する。
6: ウィンドウの設定。
7: 視線の計測開始。

視線入力キーボードが単純なプログラムで実現できていることがわかる。プログラムのうち、視線に関係する部分は 3, 4, 5, 7 行目である。このうちもっとも重要な 5 行目を次に解説する。

"eye bind" は視線で操作する方法を指定する関数である。この関数は 2 つの引数をとる。

1 つ目の引数(ここでは "'enter .end 0.5sec'") では入力であると判定する条件を指定する。ここでは ".end の領域に視線が 0.5 秒間とどまっていたら" という条件が指定されている。

2 つ目の引数(ここでは "'eye stop; exit'") では条件が満たされた場合に、実行される Tcl/Tk スクリプトを指定する。ここでは "視線の測定を終了し、アプリケーションを終了する。" というプログラムが指定されている。

このように判定する条件と処理が容易に指定できる。

このアプリケーションを実際に何人かで使用したところ、前述した問題点、入力である視線と入力ではない視線の判別の失敗が生じた。例えば、あるボタンを探す視線が入力と判定されてしまい異なった文字が入力されてしまうことが生じた。そのため、視線を常に移動させ続ける必要があり、ユーザに負担がかかった。また、入力しやすい条件(「押す」と判定されるまでの注目時間)は人によって異なった。今後、どのような条件が適しているか、また、その個人差について実験を行なうことを考えている。

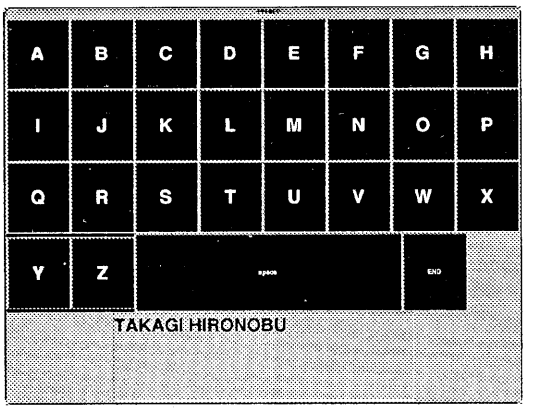

図 2. 視線入力キーボード

3.3 実験支援機能

3.3.1 機能の概要

視線に関する実験の各ステップにおいて実験者を支援するために次の機能を実装した

- 刺激提示

  3.2 節で述べた GUI アプリケーション構築機能を用いて提示する。

- データ保存機能

  実験中の視線データと X window イベントを保存することができる。

- 実験再生機能

  保存されたデータをもちいて視線データの再生、X window イベントの再送をおこなうことで実験を再生することができる。

- データ解析機能

  EyeTk では画面上にいくつかの領域を設定し、これらの領域のうちどれをどれだけの時間注目したか測定し、その結果を保存することができる。

3.3.2 実験例

EyeTkが実験支援環境として有効であることを確認するために我々が提案している視線情報を推測の基情報として用いる手法[Takagi 95b]に関する実験をおこなった。

提案している手法:

推測型インタフェース[Cypher 93]において現在用いられている推測の基情報、例えばコマンドなどのユーザの入力やディスプレイへの表示内容、だけでは推測できないユーザの意図を視線と他の情報をくみあわせることで推測するという手法を提案している。

実験の目的:

ドローイングエディター上で、現在用いられている推測の基情報だけではユーザの意図を推測できない場面において、視線から意図を推測可能であることを確認することが本実験の目的である。

想定場面:

ユーザが目測で整列させたとき、ユーザの意図した整列を推測し、自動的に整列させる機能を持つドローイングエディタを考える。このときユーザが図形を図3 aの位置から図3 bの位置に移動させたとする。この時、ユーザの意図が次のいずれかである可能性が高い。

- 2つの図形の右端を揃える。
- 2つの図形の中心を揃える。
- 2つの図形の左端を揃える。

しかし、いずれの場合もユーザの入力（この場合マウスの移動情報）と表示内容（図形とその位置）には違いがないので意図を推測できないが、視線には3種の場面で違いがあることが予想される。

実験手法:

すでに存在する図形をドラッグして移動させることしかできないドローイングエディタをEyeTkで作成し(図4)、それを刺激として用いて次の作業をおこなうよう被験者に指示した。

- 整列作業（右端、左端、上端、下端）
- 中心揃え作業（水平のみ）

解析手法:

実験を再生しながら、図5に示す図形上の領域それぞれに関してドラッグ途中の注目時間を測定した。

結果:

結果として、整列作業、中心ぞろえ作業のおのおのについて次のことがわかった。

整列作業に関して

図形を細く移動しているときには、ユーザが移動している図形のあわせようとしてる辺上で目標の図形に近い頂点に注目している時間が他のどの頂点より有意に長い。

たとえば左辺を揃えている場合図5 aのNWの頂点を有意に長い時間注目する。右辺を揃えている場合はNEの頂点を有意に長い時間注目する。

中心揃え作業に関して

図形を細く移動しているときには、中心の領域を両端の領域に比べて有意に長い時間注目する。

中心の領域とは図5 bにおけるN2Sの事である。縦方向の領域のうちN2Sへ視線が、NW2SW、NE2SEと比較して、有意に長かった。

このように一連の実験の各ステップにおいてEyeTkが有効であることが確認できた。

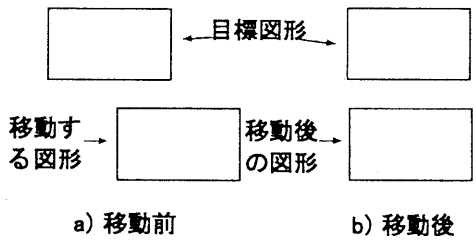

図3．実験例

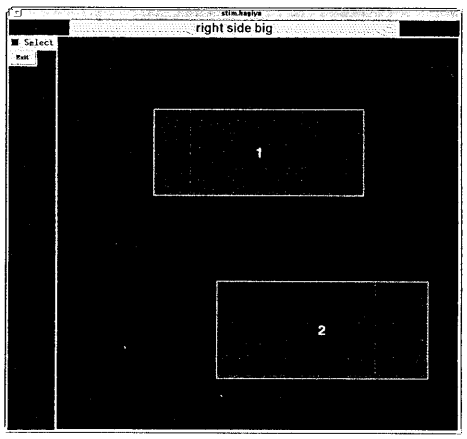

図4．刺激

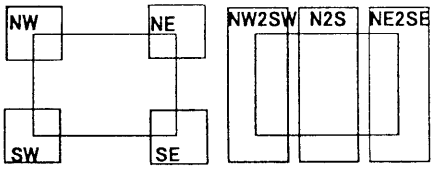

図5．領域

3.4 EyeTkの構造

EyeTkは3.2、3.3節に述べた機能をもち、将来的により複雑な処理に対応できねばならない。そこで処理速度が問題となる。ユーザがリアルタイムに操作できる必要があると同時にリアルタイムに視線の解析をおこなえねばならない。そこで、視線データ解析部(ICOULD)とユーザインタフェース管理部(IWISH)を分離して実装し、負荷を分散させた(図1)。

動作例として「0.5秒間ボタンを注視したらそのボタンを押す」場合を考える。IWISHは処理を解析して条件の種類（この場合ある領域への注視）とその引数（この場合0.5秒間）をICOULDへ送る。ICOULDは送られてきた条件の判定を行なうモジュールを内部で生成する。視線データが入ってくるたびに複数のモジュールがある場合はすべてを評価し、条件を満たしたモジュールはIWISHにイベントを発行する。IWISHはイベントを受け取ると指定された処理（この場合ボタンを押す）をおこなう。

このように処理を分散させたことにより現在実装している機能についてはリアルタイムで処理することができる。将来、処理速度が問題になった場合には両者を別のマシンで実行して負荷を分散させる。

4 今後の課題とまとめ

視線を入力デバイスとして用いる手法の開発とそのための実験を支援する環境としてEyeTkを開発した。また、EyeTkを利用して視線に関する実験をおこない有効性を確かめた。

現在のIWISHでは操作可能なwidgetがボタンなどに限られているため、他のwidgetも対応させる必要がある。また、現在のICOULDでは注視点計算をおこなえないため、これを必要とする手法を利用することができない。今後これらの実装を進めるとともに実験をおこなうことを考えている。

謝辞

本研究はNTT基礎研究所の施設をお借りしておこなっています。斉藤康己主幹研究員と吉川厚主任研究員、大野健彦研究員、およびメンタルプロセスグループ諸氏の、施設の利用、議論等多岐にわたる御協力に対し謝意を表します。また議論、ご指導をいただいた東京大学萩谷昌己教授、および萩谷研究室の諸氏に謝意を表します。

参考文献

[Cypher 93] Allen Cypher,eds: Watch What I Do : Programming by Demonstration,Allen Cypher ed,The MIT Press,1993

[Hansen 95] J.P. Hansen,W. Andersen,P. Roed: Eye-Gaze Control of Multimedia Systems,In. Symbiosis of Human and Artifact,Y. Anzai,K. Ogawa,H. Mori,eds, Elsevier Science B. V.,1995

[Osaka 93] 苧坂・中溝・古賀： 眼球運動の実験心理学、名古屋大学出版会、1993

[Otsuki 94] 大槻 説乎：知識科学とメディア技術に基づく知的教育支援、人工知能学会研究会資料 SIG-IES-9401-3,1994

[Takagi 95a] 高木・吉川：プログラミング教育支援のための視線情報の解析法、教育システム情報学会第２０回全国大会資料、1995

[Takagi 95b] 高木：視線を用いた推測型インタフェースとITS～視線から思考がわかるか？～、HUTET,vol6、1995

[Jacob 90] Jacob, R.J.K.,: What You Look At Is What You Get:Eye Movement-based Interaction Techniques,Proc. ACM CHI'90,pp 11-18,1990