活用：マウスホイール

小林 正朋[†]　　五十嵐 健夫[‡]

[†‡]東京大学大学院情報理工学系研究科　[‡]科学技術振興機構さきがけ

[†]kobayash@is.s.u-tokyo.ac.jp　　[‡]takeo@acm.org

1. はじめに

GUI を操作するための最も基本的なデバイスであるマウスに関して、「この 10 年間に起こった最大の進化は？」と尋ねられたなら、多くの人々はホイール機能こそがそれであると答えるはずだ。狭いスクロールバー領域にポインタを移動させなければならないという旧来の仕様は効率が悪く、ホイールの登場は GUI ユーザにとって福音であった。実際、ホイール付きのマウスは誕生後わずか数年足らずで PC の標準装備となり、今日ではほとんどの PC ユーザがホイールを常用している。

このように素晴らしいマウスホイールであるが、現状では充分に活用されているとは言い難い。第一に、ホイールの回転によって実現される操作が単純な等速スクロールに限られているため、長い距離を移動したい場合に困難を伴う。第二に、GUI の操作を総合的に捉えた場合（スクロールは「決定」や「取り消し」ほどには普遍的な操作でないため）左右のボタンと比較してホイールの活用頻度が低くなってしまう。これはマウスにとって大きな損失である。なぜなら、正負の 1 次元値を与えることのできるホイールは、ON/OFF の 2 値のみを持つボタンでは実現できない有用なインタラクションの可能性を秘めているはずだからである。

上記の観点に立ち、本稿は(1)スクロール機能の強化、(2)その他の応用の両面からマウスホイールを捉え直し、前者についてはマウスポインタの座標に基づくスクロールの加速手法を、後者についてはオブジェクトのドラッグ時におけるウィンドウ切替のためにホイールを用いる手法を提案する。

2. ポインタの座標によるスクロール加速

2.1. 既存スクロールシステムの利点と欠点

ホイールの回転に基づくスクロール操作は、ホイールの回転角がそのまま上下への移動距離に対応しているという点で直感的であり、初心者にも容易に理解できる。しかし、スクロールの基本速度は数行単位での移動が可能となるよう低速に調整されているため、ページ単位で大幅にスクロールしたい場面では極めて非効率である。一方、スクロールバーやキーボード上の PgUp/PgDn キーを利用する方法ではマウスの繊細な操作やデバイスの持ち換えを要求されるため、ホイール操作とは別の意味で効率が悪い。また、多くの GUI システムではホイールを押下することでスクロールモードに切り換わり高速なスクロールが可能となるが、マウスの遷移をスクロール速度に対応付けるこの手法は特に高速移動時において制御が難しい。五十嵐ら[2]はスクロール速度に基づき自動的に画面表示をズームアウトさせることでこの問題を解消したが、このように先進的な機構を既存の GUI システムへ単純に組み入れることは困難である。

2.2. 提案するスクロール加速手法

前節に述べたスクロール効率の問題を解消するためには、移動速度の高低に関わらず安定して操作を行うことのできる手法が必要である。そこで、マウスポインタの x 座標値に応じてホイールの単位回転当たりのスクロール量を変更する手法を提案する。すなわち、ウィンドウの中心にポインタを置いてホイールを回転させた場合には標準の速度、右端で回転させた場合には N 倍の速度、左端ならば $1/N$ 倍の速度でスクロールさせる（図 1）。

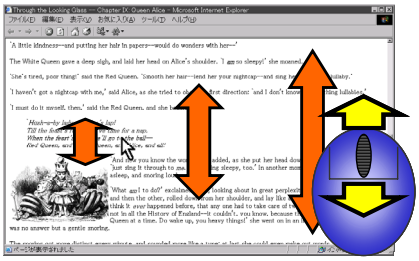

図 1. 右側でホイールを回転させると高速スクロールする

ウィンドウの中心付近で操作を行った場合は従来と同一の挙動を示すため、ユーザはこの手法を

Interactions with Mouse Wheels, Masatomo Kobayashi and Takeo Igarashi, Graduate School of Information Science and Technology, The University of Tokyo, JST PRESTO

自然に受け入れることができる。一方、倍率は x 値によって連続的に変化するため、熟練によって極めて繊細な操作が可能となる。また、スクロールに特化した機構であるホイールのみに依存して動作するため、一般的な GUI システムへ組み入れる際に既存の機能と衝突してしまう危険性は低い。実際、著者らの実装した Windows 用のシステムはメッセージフックによってマウスホイールイベントの⊿値を書き換えているだけであり、ホイール回転によるスクロール操作をサポートする全ての既存アプリケーションに対して有効である。

3. ドラッグ時におけるウィンドウ切替

3.1. 既存ウィンドウ切替手法の特徴と課題

加藤ら[3]はウィンドウの重なり方向を z 軸とし、専用の 1 次元デバイスやマウスホイールの入力を z 値に対応させることで汎用的なウィンドウ切替機構を実現したが、ウィンドウの枚数が多い場合に繊細な操作を要求されるうえ、ホイールのサポートに制約があった。また、一般的な GUI システムにおいてウィンドウ切替操作に困難を伴うのはオブジェクトのドラッグ時のみであり、汎用的なシステムは必ずしも効果的ではないと考えられる。

3.2. 提案するウィンドウ切替手法

我々の提案する手法は、オブジェクトのドラッグ中にホイールを下に回転させることで、マウスポインタの直下にあるウィンドウを一時的に消去するというものである（図 2）。消去されたウィンドウは内部のスタックに保持されており、ホイールを上に回転させることで順に復帰する。また、ドラッグ終了時には全てのウィンドウを復帰する。本手法は、利用場面をオブジェクトのドラッグ時に限定することで通常のスクロール操作との衝突を回避している。この仕様は、一般的な GUI システムにおいてドラッグ中にスクロールするためにはマウスポインタを画面の端に停留させれば良く、また実際に多くのユーザがそうしているため、マウスホイールに別の能力を与えても機能の競合が生じないという見解に基づく。また、我々の手法はホイール入力を z 値として用いる訳ではないため、ウィンドウの総数に関わらず必要最低限の操作でドラッグ作業を完了できる。なお、同様の目的を持つ魅力的な手法として「捲る」メタファによる fold-and-drop[1]があるが、我々はマウスホイールを用いることでさらなる効率の向上を目指した。

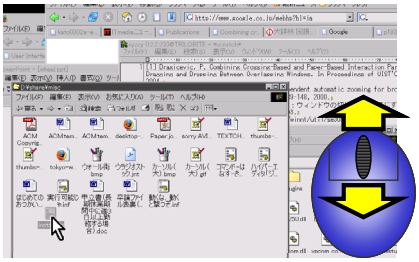

図 2. アイコンをドラッグ中にホイールを上下に回すとポインタ直下のウィンドウをそれぞれ消去/復帰させる

3.3. 非スクロール操作へのホイール活用関連事例

スクロールやウィンドウ切替以外の用途として、Ctrl キーとホイールの同時操作によって表示を拡大/縮小する機能は広く採用されている。また、WheelVol[5]は無変換キーとホイールの同時操作にオーディオデバイスの音量調節機能を割り当てる。他に、塚田ら[4]はホイールの回転によってツール選択を行う手法を提案している。なお、これらのシステムは（本来のスクロール機能と共存するために）修飾キーなどの明示的なモード切替を必要としている。対照的に、我々の手法は有効場面をオブジェクトのドラッグ操作と連動させることで暗黙的なモード切替を行っている。

4. おわりに

本稿は、マウスホイールの特質について考察し、マウスポインタの座標に基づいてスクロールを加速する手法およびオブジェクトのドラッグ時にホイールによってウィンドウを切り換える手法を提案した。今後、ホイールのさらなる可能性を追求し、このデバイスの一層の地位向上を図りたい。

参考文献

[1] Dragicevic, P. Combining Crossing-Based and Paper-Based Interaction Paradigms for Dragging and Dropping Between Overlapping Windows. In *Proceedings of UIST'04*, pp.193-196, 2004.

[2] T. Igarashi, K. Hinckley. Speed-dependent automatic zooming for browsing large documents. In *Proceedings of UIST'00*, pp.139-148, 2000.

[3] 加藤直樹, 小國健. ばらばらウィンドウ：ウィンドウの切り替えを容易にするインタフェース, *インタラクション 2003 論文集*, pp.123-130, 2003.

[4] 塚田有人, 星野剛史. On-Demand Cursor：直感的なカーソル機能選択のためのインタラクション. *WISS2004 論文集*, pp.41-46, 2004.

[5] WheelVol. http://www.vector.co.jp/soft/winnt/util/se308455.html