

バブルカーソルの GUI 環境への適用と拡張

重森 晴樹 入江 健一 倉本 到 渋谷 雄 辻野 嘉宏
京都工芸繊維大学

1 はじめに

GUI 環境において、ポインティングタスクは非常に重要な操作であり、その性能が作業効率に与える影響は大きい。そこで、Grossman らはさらなる性能向上を目指し、バブルカーソル^[1]を提案した。これは、円形のエリアカーソル^[2]の半径を近傍の目標までの距離に応じて動的に変化させることにより、カーソルの移動時間を短縮する目標選択手法である (図 1)。

2 GUI 環境での問題と課題

GUI 環境におけるポインティングタスクは**目標選択**と**座標選択**の 2 つに分類できる。しかしながら、バブルカーソルは有用な目標選択手法である反面、面の選択領域を持つため座標選択を行うことができない。

これはドラッグアンドドロップ (DnD) でも問題を引き起こす。DnD は、「掴む」「引きずる」「放す」の 3 つのタスクに分解することができる。また、「掴む」と「放す」のそれぞれには目標選択と座標選択があり、座標選択の「掴む」は複数目標選択のために、座標選択の「放す」は目標の移動先の座標を指定するために必須である。目標選択だけではなくこれらの座標選択も行うためには、タスクに応じてポイントカーソル機能とバブルカーソル機能を切り替えられる必要がある。

また、ポイントカーソルでは常に目標とカーソルが重なった状態で目標を掴むため、直感的に目標を移動できる一方で、バブルカーソルでは目標とカーソルが離れていても掴むことができるため、DnD の直感性が損なわれる可能性がある。

さらに、Grossman ら^[1]の実験は、GUI 環境への適用について十分に考慮されていないと考える。彼らの実験ではすべての目標が真円であるため、実質的に高さと同幅が等しいが、GUI 環境では細長い形状の目標も存在する。また、目標が隙間

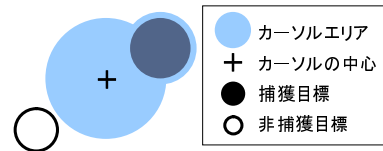


図 1: バブルカーソル

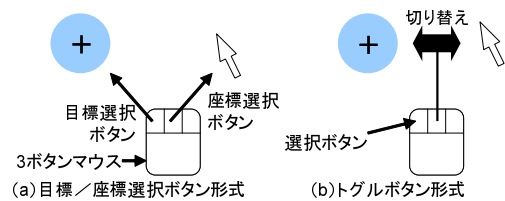


図 2: 機能の切り替え方法

なく配置された場合、目標の実質的な幅を拡大できないためバブルカーソルの性能が低下すると考えられる。しかし、GUI 環境と実験環境の違いが及ぼす影響について評価および分析は行われていない。従って、GUI 環境を考慮した実験環境下で再評価する必要があると考える。

3 GUI 環境のための拡張

我々は、タスクに応じて機能を切り替える方法として、3 ボタンマウスを使用した 2 つの方法を考案し、実装した (図 2)。図 2 (a) の方法では、目標選択ボタンならば捕獲目標を、座標選択ボタンならばバブルカーソルの中心座標を選択できる。また、DnD 中の目標を放すときの機能の切り替えは、一度両方のボタンを押下した後、最後に放した方のボタンに対応する機能として動作させることで実現する。図 2 (b) の方法では、目標を選択するボタンと、トグル式の機能切り替えボタンを使用する。

また、バブルカーソルのための DnD 手法を 3 つ考案し、実装した (図 3)。目標選択の「掴む」と「放す」を行う場合において、図 3 (a) の方法では、DnD の間、目標とカーソルの中心との相対的位置関係を固定する。図 3 (b) の方法では、掴んだ直後にカーソルの中心が目標の中心に移動する。図 3 (c) の方法では、掴んだ直後に目標の中心がカーソルの中心に移動する。なお、図 3 のすべての方

Enhancing the bubble cursor for GUI environment,
Haruki Shigemori, Ken'ichi Irie, Itaru Kuramoto, Yu
Shibuya, Yoshihiro Tsujino
Kyoto Institute of Technology

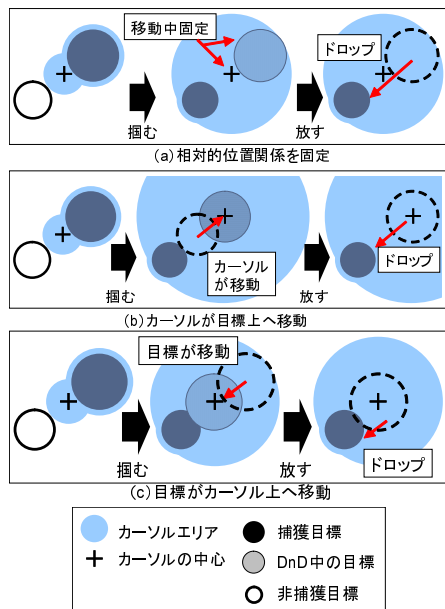


図 3: バブルカーソルでの DnD 手法 (目標選択)

法において、バブルカーソルは掴まれた目標が存在しないかのように拡大、縮小し、ドロップ先の目標を捕獲する。また、バブルカーソルでの座標選択は、ポイントカーソルの先端の代わりにバブルカーソルの中心を用いて行う。よって、座標選択の「放す」は、中心座標に他の目標があるならばその目標へのドロップとして動作し、無いならば掴んだ目標がある位置への移動として動作する。

4 変形バブルカーソル

バブルカーソルを一般化して考えると、大きさが動的に変化するエリアカーソルの捕獲目標選択法は、目標ごとに算出した優先度のうち、最も高い値を持つ目標を選択する手順である。バブルカーソルであれば、目標までの距離だけで優先度が決定されると言える。そこで、さらなる性能向上を目的として別のパラメータを優先度に導入する。

1 つ目に、目標までの距離とカーソルの移動方向を考慮するカーソルを実装した。まず、カーソルから周囲を見回し、他の目標の死角に入らない目標を捕獲候補とする。目標の優先度は、移動方向に近い方向にある捕獲候補ほど、また、距離が近いほど高くする。次に、すべての捕獲候補の方向に伸びる足を考え、優先度の高さを基にして足の長さを決定する。そして、足と足の間には谷ができるように描画すると図 4 ((a) から (c) へ移動する様子) のようなアメーバ状のカーソルになる。このカーソルは移動先にある目標までの実質的な距離を短縮するため、性能向上が期待できる。

2 つ目に、目標までの距離と目標の利用頻度を

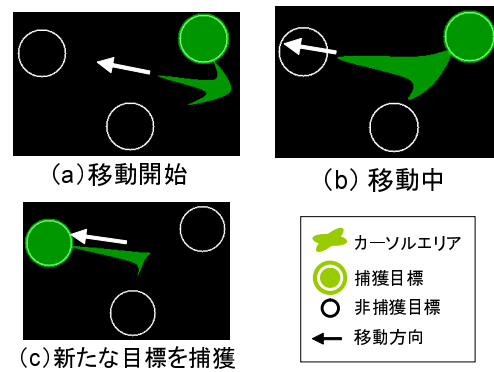


図 4: 移動方向を考慮したアメーバ状カーソル



図 5: 足の骨格だけのアメーバ状カーソル

考慮するカーソルを実装した。利用頻度とは、ある目標が過去に選択された回数である。利用頻度が高い目標ほど捕獲しやすくすることで、一連のポインティングタスクの総移動時間を短縮でき、作業効率の向上が期待できる。移動方向を考慮した場合と同様にこのカーソルもアメーバ状カーソルとし、移動方向の代わりに利用頻度を用いて足の長さを決定する。

なお、図 5 は足の骨格だけを描いた場合である。実質的には、先の滑らかな輪郭を描いた場合と同等であるが、見た目の違いによる影響を調べるため両方を実装した。

5 おわりに

本研究では、バブルカーソルを GUI 環境に導入するために必要な拡張を行った。現在、GUI 環境に近い実験環境で評価し、そのユーザビリティを分析している。また、移動方向や利用頻度を考慮することで移動時間を短縮できる可能性がある手法を考案し、実装した。現在、評価実験を行ってそれらのユーザビリティを分析中である。

参考文献

- [1] Grossman, T. and Balakrishnan, R.: “The Bubble Cursor: Enhancing target acquisition by dynamic resizing of the cursor’s activation area”, Proc. of CHI 2005, pp. 281–290 (2005).
- [2] Kabbash, P. and Buxton, W.: “The “Prince” technique: Fitts’ law and selection using area cursors”, Proc. of CHI’95, pp. 273–279 (1995).