Vol.2012-HCI-149 No.7
2012/7/19

軌跡に基づいたundo/redoインタフェース

大江龍人[1,a)]　志築文太郎[2]　田中二郎[2]

概要：我々は軌跡に基づく undo/redo インタフェースを示す．ユーザはマウスを使って形状を指定することにより，同様の形状を持つ操作まで undo/redo することが可能である．その様な操作が複数存在する場合には，選択的に undo/redo することも可能である．本稿では本インタフェースのインタラクション手法，及び応用的な使用方法を示す．

キーワード：Undo/redo，軌跡，履歴，なぞる，直接操作，デスクトップインタフェース，GUI イベント，UnReŤ

Undo/Redo by Trajectory

Tatsuhito Oe[1,a)]　Buntarou Shizuki[2]　Jiro Tanaka[2]

Abstract: We show a trajectory based undo/redo interface. Using the interface, a user traces actions' trajectories shown in a desktop. As a result, undo/redo operations are performed with rapidly selection of a target from histories. In this paper, we show interaction techniques using the interface and advanced usages of the interface. In addition, we show a prototype's implementation based on a preliminary study.

Keywords: Undo/redo, Trajectories, History, Tracing, Direct Manipulation, Desktop Interface, GUI Events, UnReŤ

1. 序論

多くのアプリケーションにおいて，ユーザの操作は「履歴 (History)」として保存され，undo/redo 操作に用いられる．この履歴には図 1a に示す様に，単一の時間軸に沿ったリニアな履歴モデルが一般的に採用されており，ユーザの操作が履歴として積み重ねられる．この履歴を用いてユーザが undo/redo 操作を行う場合，図 1b の様に undo/redo 用のコマンド (以降，ショートカット) を 1 度ずつ入力する．この場合にユーザが履歴中のある操作に到達するには，ショートカットを複数回入力する必要があるため，操作を完了させるためにショートカットの入力回数分の時間を要する．

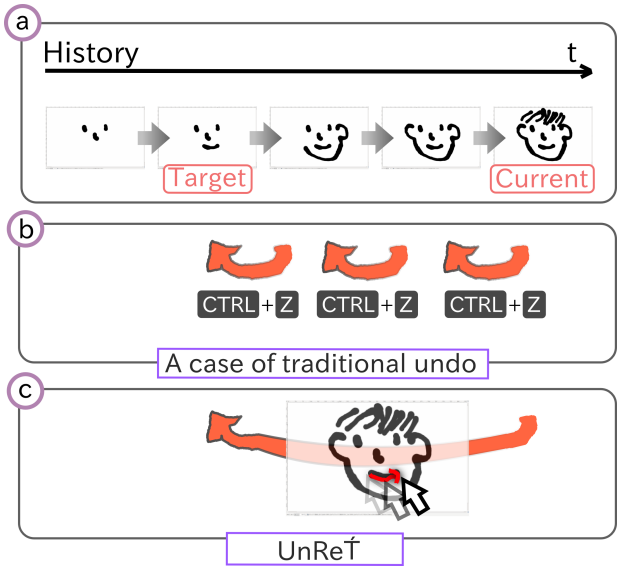

図 1　リニアな履歴モデルにおける従来の undo と UnReŤ を用いた undo．a: リニアな履歴モデル．b: ショートカットを用いた undo．c: UnReŤ を用いた undo．

Fig. 1　Traditional undo and UnReŤ's undo in linear history model.

[1] 筑波大学大学院システム情報工学研究科コンピュータサイエンス専攻
Department of Computer Science, Graduate School of Systems and Information Engineering, University of Tsukuba
[2] 筑波大学システム情報系
Faculty of Engineering, Information and Systems, University of Tsukuba
[a)] tatsuhito@iplab.cs.tsukuba.ac.jp

素早く undo/redo を行うために，履歴から任意の操作を直接指定すると undo/redo 操作が行われるインタフェースがこれまでに示されてきた．これらのインタフェースには，「テキストやスクリーンショットを用いて履歴を視覚化するインタフェース」と，「undo/redo の操作範囲を局所化するインタフェース」の 2 種類がある．それぞれの具体例は以下に示すものとなる．

テキストやスクリーンショットを用いた履歴の視覚化
テキストを用いた履歴の視覚化では，履歴の操作リストがテキスト表示される．例えば，プレゼンテーションアプリケーションである LibreOffice Impress[*1]では，「ページの挿入」,「見出しを移動」,「透過性のあるビットマップを移動」等の，スライドの編集操作のテキストリストが表示される．
スクリーンショットを用いた履歴の視覚化では，履歴のスクリーンショットリストが表示される．例えば，ドローイングアプリケーションである GIMP[*2]では，過去のドローイングのスクリーンショットリストが表示される．

undo/redo の操作範囲の局所化　この例としては，テキストエディタである Emacs[*3]や DistEdit [1] の undo/redo 操作が挙げられる．これらのエディタにおいて，ユーザは undo/redo が適用される操作範囲をあらかじめ指定し，その範囲内において undo/redo 操作を行うことができる．これらエディタの履歴モデルには，特定の操作範囲が undo 可能である regional undo モデル [2] や，単一操作が undo 可能である selective undo モデル [3] が用いられる．

これら 2 つのインタフェースはそれぞれ利点と欠点を持つ．前者のインタフェースでは，操作履歴の概観を素早く閲覧可能である．しかしながら，履歴が膨大になった場合，局所的な履歴点の選択に時間を要する．それに対し後者のインタフェースでは局所的な undo/redo 操作が可能であるが，一般的に用いられるリニアな履歴モデルは採用されない．それゆえ，適用可能なアプリケーションが限定される問題がある．

我々は，リニアな履歴モデルを採用する一般的なアプリケーションに広く適用可能な，操作を直接指定する undo/redo インタフェースの有効性調査を目的としている．そのために「軌跡」に基づいた undo/redo インタフェース (Undo/Redo by Trajectory，以下「*UnReTí*」と呼ぶ) を開発している．ここで軌跡とは drag，press，release から構成されるマウス操作である．UnReTí を用いてユーザは，デスクトップに表示されるマウスの軌跡をなぞり undo/redo 操作を行う．UnReTí を用いて undo 操作を行っている様子を図 1c に示す．図 1c では，ユーザは口を描いた軌跡をなぞることにより，それを描いた時点まで undo 操作を行っている．

[*1] http://www.libreoffice.org/
[*2] http://www.gimp.org/
[*3] http://www.gnu.org/software/emacs/

2. 関連研究

UnReTí は，リニアな履歴モデルを採用する一般のデスクトップアプリケーションにおいて，過去の軌跡をなぞることにより undo/redo 操作が行われるインタフェースである．このインタフェースに関連する研究は，履歴モデル，テキストやスクリーンショットを用いた履歴の視覚化，デスクトップの拡張が挙げられる．

2.1 履歴モデル

1 節において示した様に，undo/redo の操作範囲を局所化しユーザに undo/redo 操作を行わせる履歴モデルがこれまでに示されてきた．

Berlage [3] や Myers ら [4] は，過去の単一操作を選択した undo/redo 操作が可能である selective undo モデルを示した．この selective undo モデルと比較し操作範囲がさらに広い履歴モデルとして，Kawasaki ら [2] の regional undo モデルがある．これらの履歴モデルは，その操作範囲が空間的に局所化される特徴を有する．そのため応用としては単一ユーザが用いるインタフェースの他に，複数ユーザの協調作業環境におけるユーザ毎の undo/redo 操作 [5], [6] が存在する．

これらの履歴モデルは，リニアな履歴モデルを採用する一般的なアプリケーションに広く適応することが出来ない．対して UnReTí ではマウス操作の軌跡を履歴として保存し，それに基づいて undo/redo が可能な実装を採用しているため，特定アプリケーションに依存せず広く適用可能である．

2.2 テキストやスクリーンショットを用いた履歴の視覚化

テキストやスクリーンショットを用いて履歴を視覚化するインタフェースの研究がこれまでに行われてきた．これらの研究において最も単純な視覚化は，テキストやスクリーンショットのリストを視覚化するものとなる．例えば Meng ら [7] の研究では，スクリーンショットリストを用いて selective undo [3] が視覚化される．

単純な視覚化と比較し，さらに履歴の付加情報をユーザに提示し，閲覧性を向上させる試みがなされている．Kurlander ら [8] の研究では，GUI 操作時の前後のスクリーンショットを組にしたリストを用いて視覚化し，ユーザに操作の文脈を提示する．また，Nakamura ら [9] はスクリーンショット上に GUI 操作を重畳表示させ視覚化し，単純なスクリーンショットリストと比較し履歴の検索速度を向上させた．Vratislav [10] も検索速度を向上させるために，履歴のテキストリストの表示に Fisheye Menus [11] を適用さ

せた．

これらの研究では視覚化されたリストを閲覧・選択して undo/redo 操作が行われる．対して UnReT では，既にデスクトップ画面上に表示される軌跡を直接なぞり undo/redo 操作が行われる．そのため，UnReT ではリストの閲覧時間が短縮され，より素早く undo/redo 操作が可能であると我々は考える．

2.3 デスクトップの拡張

UnReT はデスクトップを操作するマウスの軌跡を履歴として撮りため，通常の undo/redo 操作を拡張させるインタフェースである．このシステムに関連する研究として，デスクトップの履歴を拡張する研究や，デスクトップのマウス操作を拡張する研究が挙げられる．

デスクトップの履歴拡張　デスクトップにおけるユーザの操作や状態を記録し，ユーザを支援するインタフェースがこれまでに提案されてきた．Rekimoto [12] はデスクトップ上のファイルを時系列に保存整理するために，任意時点でのデスクトップ環境の状態を再現可能にする Time-Machine Computing を示した．また，Kelly ら [13] はデスクトップ上の操作履歴を保存し，任意のアプリケーションにおいて操作されたファイルを視覚提示する Desktop History を示した．さらに Grossman ら [14] は，操作履歴の他にデスクトップ全体の映像を撮りため，任意の操作時における映像を閲覧可能であるインタフェースを示した．

これらのインタフェースでは撮りためた操作は視覚提示するために用いられるが，我々のインタフェースにおいては，デスクトップにおけるユーザの操作を拡張させるために操作が撮りためられる．

デスクトップのマウス操作拡張　デスクトップにおけるユーザのマウス操作を拡張させる研究がなされている．Appert ら [15] はカーソルと操作対象との間にバネを定義し，それを用いてユーザにマウスの逆操作をさせることにより，マウス操作の undo とキャンセルが可能である Dwell-and-Spring を示した．また Kobayashi ら [16] は，マウスを用いた異なるディレクトリ間のファイル移動を容易にするために，ファイルを投げる様な移動操作が可能である Boomerang を示した．さらに Chapuis ら [17] は，デスクトップ上の要素にユーザ定義のマークを関連付け，関連付けられた要素を素早く操作可能にする UIMarks を示した．

3. 軌跡に基づいた undo/redo インタフェース

UnReT は，マウス操作の軌跡をなぞることにより undo/redo 操作を行うインタフェースである．この UnReT のインタラクションは以下のユーザ入力とシステム出力から構成される．

ユーザ入力　図 1c の様に画面に表示されている軌跡をなぞる．

システム出力　なぞられた箇所まで undo/redo 処理を行う．

システムはこのインタラクションを支援するために，「マウス軌跡の視覚表示機能」，「undo/redo 操作の下見機能」，「類似軌跡がなぞられた場合のスクリーンショットリスト表示機能」のユーザ補助機能を有する．本節では，最初に UnReT を用いた基本操作を詳述し，さらにユーザ補助機能を具体的に説明する．

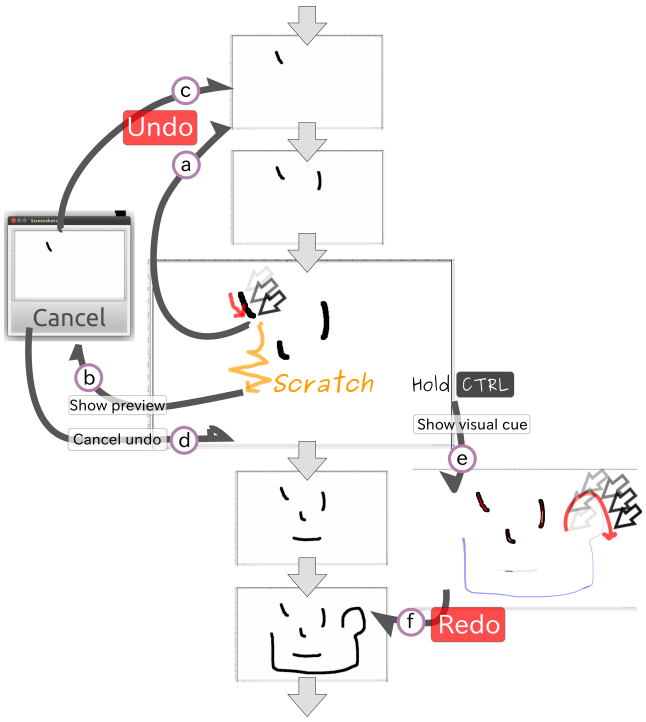

図 2　UnReT を用いた undo/redo の一連のインタラクション．a: マウスを用いて軌跡をなぞり undo を行う．b: なぞる際に Scratch 操作を用い下見を行う．c と d: undo の適用可否を選択する．e と f: Ctrl 長押しによる redo 対象の軌跡表示後，軌跡をなぞり redo を行う．

Fig. 2　A flow of undo/redo's interactions using UnReT.

3.1 基本操作

UnReT を用いた基本操作は，図 2a に示される undo 操作と，図 2e 及び図 2f に示される redo 操作である．図 2 を例にし，それぞれの基本操作におけるユーザの操作例を以下に説明する．

undo 操作　ユーザは右目を描いた箇所まで undo 操作を行いたいと考えたとする．まずユーザは UnReT を用いて，Ctrl を押しながら右目の軌跡をなぞる．その結果，右目を描いた箇所まで undo される (図 2a)．

redo 操作　ユーザは，輪郭を描いた箇所まで redo 操作を行いたいと考えたとする．まずユーザは Ctrl を長押し

し，redo 対象となる軌跡群を視覚表示させる (図 2e)．その後ユーザは Ctrl を押しながら視覚表示された輪郭の軌跡をなぞり，その結果，輪郭を描いた箇所まで redo される (図 2f)．

3.2 ユーザ補助機能

UnReŕ の基本操作を補助するために以下の機能をシステムは有する．

軌跡の視覚表示 Ctrl を長押しすると図 3a に示す様にマウス軌跡が視覚表示される．視覚表示される軌跡の色として現在の実装では，undo 対象の軌跡は赤色，redo 対象の軌跡は青色を採用している．また，図 3b に示す様に，現在の履歴から遠い操作ほど表示色は濃くなる．遠い操作ほど表示色を濃くした理由は，現在の履歴から近い操作程ユーザはその軌跡を記憶していると考えたためである．ユーザが記憶していない遠い操作ほど視覚表示が必要であると我々は考え，遠い操作の表示色を濃くした．なお，この表示機能は図 2e に示す様に，「操作対象の軌跡が視認できない」redo 操作時には必要不可欠な機能である．

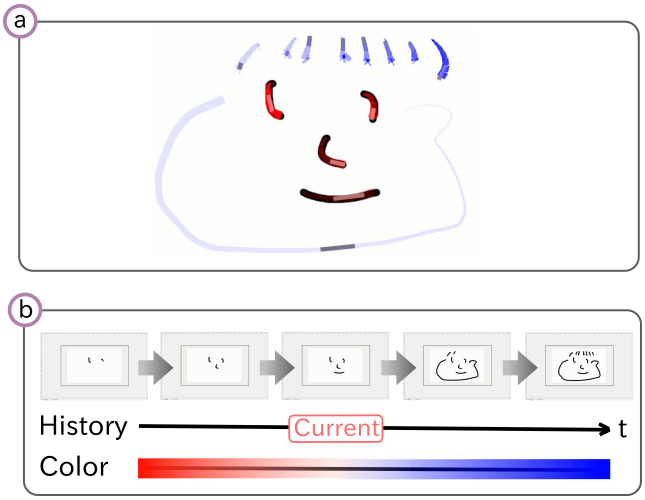

図 3 a: Ctrl を長押しするとマウス軌跡の視覚表示が行われる．undo 対象の軌跡は赤色，redo 対象の軌跡は青色を用いて表示される．b: 表示色は現在の履歴点から遠いほど濃くなる．

Fig. 3 Show visual ques by holding down a Ctrl key.

undo/redo 操作の下見 undo/redo 操作は履歴中の特定操作をキャンセル又は削除するために多く用いられる他，特定の履歴点におけるアプリケーションの状態を確認するためにも用いられると我々は考える．我々は後者の使用方法を支援するための機能として undo/redo 操作の下見機能を実現した．図 2 に示す様に，ユーザは undo/redo 操作のためのなぞり入力中にジグザグにマウスを Scratch 操作することにより，図 2b の様にスクリーンショットによる状態確認画面を表示させる．ユーザはその後，スクリーンショットを選択し図 2c の様に undo を行える．なお図 2d の様に，本機能は UnReŕ による undo/redo のキャンセル操作に応用することが可能である．

類似軌跡がなぞられた際のスクリーンショットリスト表示 類似軌跡とは似ているマウス軌跡を指し，例えば図 4a に示される重ね書きされた軌跡群となる．この様な軌跡群は，ドローイングソフトを用いストロークを重ねて描く場合に多く出現するものである．UnReŕ を用い図 4a の軌跡群をなぞると，図 4b の様に undo/redo の適用候補がスクリーンショットリストにて表示され，ユーザはこれら候補から選択を行う．

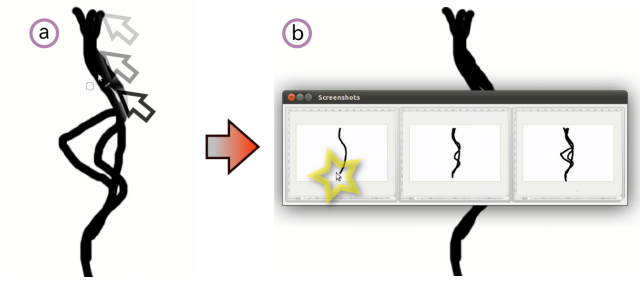

図 4 類似軌跡がなぞられた際のスクリーンショットリスト表示．a: 類似軌跡をなぞる．b: undo/redo の適用候補がスクリーンショットリストにて表示され，ユーザはこれら候補から選択を行う．

Fig. 4 Show a screenshots' list in case of tracing similar trajectories.

4. プロトタイプ実装

UnReŕ はマウス軌跡の記録 (図 5a)，マウス軌跡のマッチング (図 5b)，undo/redo 処理 (図 5c) から構成される．現在これらの機能はプログラミング言語 Python[*4]を用いて実装されており，Linux 環境において動作する．

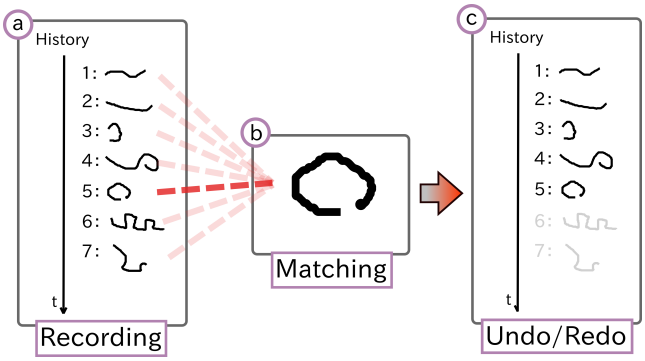

図 5 UnReŕ はマウス軌跡の記録，マウス軌跡のマッチング，undo/redo 処理から構成される．

Fig. 5 UnReŕ consists of recoding, matching, and undo/redo processing.

[*4] http://www.python.org/

4.1 マウス軌跡の記録

デスクトップの任意の場所においてマウスが press，drag，release 操作された場合，デスクトップにおける座標群が図 6 の様に軌跡として履歴に記録される．また記録時には，ユーザ補助機能に使用されるスクリーンショットも同様に保存される．

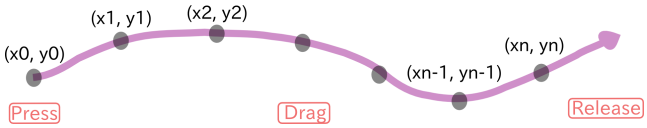

図 6 マウスの press，drag，release 時の座標群が軌跡として記録される．

Fig. 6 Points of press, drag, and release using a mouse are recorded as a trajectory.

4.2 マウス軌跡のマッチング

Ctrl を押しながらマウス軌跡が入力されると，入力された軌跡と履歴内の軌跡群とのマッチングが行われる．この処理では，入力された軌跡と履歴内の各軌跡との座標群の類似度が算出される．我々はこの処理に，図 7 に示す UnReT́ 用に修正された DP マッチングを用いた．この DP マッチングでは，入力された軌跡 (図 7b，図 7c) と履歴内の軌跡 (図 7a) との類似度を計算する際に，press された点同士が距離 d より遠い場合は計算が行われない．この実装により，入力された軌跡と履歴内の軌跡群「全て」との類似度計算が回避され，その結果マッチング速度が向上する．なお現在の実装では，距離 d は経験的に 30px と定められている．

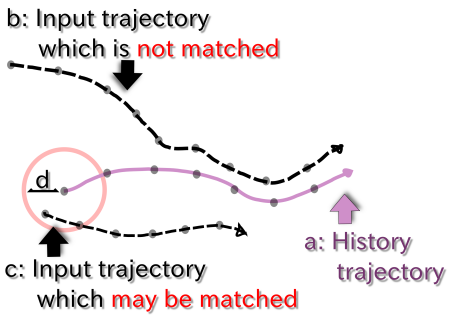

図 7 UnReT́ 用に修正された DP マッチング．Ctrl を押しながら入力された軌跡 (b，c) と履歴内の軌跡 (a) との類似度を計算する際に，press された点が空間的に遠い場合は計算が行われない．

Fig. 7 Modified DP matching for UnReT́.

4.3 undo/redo 処理

undo/redo 処理では，入力軌跡と最も類似する履歴内の軌跡に至るまで，システムが複数回 undo/redo のキーボードショートカットを入力する．この undo/redo 処理により，複数回の undo/redo 操作がシステム側から自動的に行われる．

なお，アプリケーションごとに undo/redo コマンドは異なるものである．例えば，redo のキーボードショートカットは Ctrl+y や Shift+Ctrl+z 等となる．さらに，アプリケーションによっては undo/redo のキーボードショートカットが存在しない場合も有り得る．これらの理由から，UnReT́ を一般的なアプリケーションに広く適用可能にするためには，アプリケーションごとの undo/redo コマンドの差異を吸収する必要がある．

このため，我々はアプリケーション毎に undo/redo の処理内容を登録できる機能を実装した．この機能により以下の処理内容が登録可能である．

キーボードショートカットによる undo/redo 処理 この処理内容では，登録されたキーボードショートカットによる undo/redo 処理が行われる．

undo/redo 処理をしない この処理内容では，高類似度の軌跡が Ctrl を押しながら入力された場合においても，undo/redo 処理が行われない．

マウスの逆操作による undo/redo 処理 この処理内容では，システムが履歴内の軌跡を release，drag，press 順にマウス操作する．具体的に説明すると，図 6 の (x_n, y_n) が press 点となり，(x_{n-1}, y_{n-1}),...,(x_1, y_1) と軌跡とは逆順に drag され，最終的に (x_0, y_0) においてマウスが release される．なお，この undo/redo 処理はマウスの逆操作を用いて undo とキャンセルが可能である Dwell-and-Spring [15] から着想を得ている．

この処理内容の登録機能を用いることにより，例えば「undo 対象が GIMP である場合 Ctrl+z のキーイベントをウィンドウに送信する」の様な登録が可能となる．

5. 本インタフェースの応用

本稿では，一般的なアプリケーションに広く適用可能な，操作を直接指定し undo/redo を行う UnReT́ の有効性調査を目的としている．その調査を行うために，我々は様々な環境や操作に UnReT́ を適用させ試用し，そして UnReT́ が有効となり得る場面を探った．本節では UnReT́ を用いた適用環境と応用操作を述べた後，それらに基づいた考察を述べる．

5.1 適用環境

図 8 に示す様に「マウスとキーボードを用いた環境」，「スタイラスを用いた環境」，「携帯情報端末のタッチインタフェースを用いた環境」のそれぞれの環境において，本論文の第一著者が UnReT́ を試用した．なお，全ての環境において UnReT́ は GIMP を対象として試用された．以下にそれぞれの環境毎のセットアップや操作方法を述べ，さらに，それぞれの利点や欠点を述べる．

マウスとキーボードを用いた環境 (図 8a) この環境におい

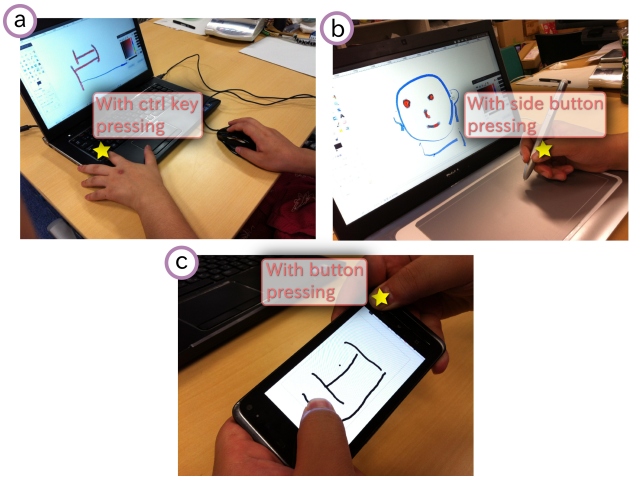

図 8 UnReT́ の試用を行った環境．a: マウスとキーボードを用いた環境．b: スタイラスを用いた環境．c: 携帯情報端末のタッチインタフェースを用いた環境．

Fig. 8 Environments for UnReT́'s trial using.

ては，図 8a★ の箇所に左人差し指を添え Ctrl ボタンを押しながら，右手のマウスを用いて UnReT́ を試用した．この環境において UnReT́ が機能することは確認できたが，履歴点が時間的に近い数回の undo/redo 操作を行う際は，UnReT́ を用いずに左手を用いてキーボードショートカットを直接入力した方が有効であった．

スタイラスを用いた環境 (図 8b) この環境ではスタイラスのみを用い，図 8b★ の箇所に右手親指を添え，スタイラスのサイドボタンを押しながら UnReT́ を試用した．第一著者は日常的にスタイラスを全く用いないが，それにも関わらずスタイラスの方がマウスとキーボードよりも操作しやすかった．この要因の 1 つとしては，スタイラスの方がマウスよりも軌跡がなぞりやすいインタフェースであることが挙げられる．Accot ら [18] は幅を持たせた直線や円の軌跡をなぞった際のパフォーマンスを，Steering Law [19] に基づき入力デバイスごとに評価した．その評価結果によると，直線より複雑な円の軌跡を描くタスクにおいて，マウスよりスタイラスの方が高いパフォーマンスが発揮された．この理由から軌跡を直接なぞり undo/redo 操作が行われる UnReT́ は，軌跡をなぞりやすいスタイラスとの相性が良いと我々は考える．

携帯情報端末のタッチインタフェースを用いた環境 (図 8c) この環境においては，図 8c★ に存在するボタンを右手親指を用いて押しながら，左手を用いて軌跡を描き UnReT́ を試用した．なお，この環境は UnReT́ が稼働する Linux サーバに，携帯情報端末から VNC クライアントを用いてリモートログインすることにより実現した．タッチインタフェースを用いた操作は，マウスやスタイラスと比較し直接的に軌跡をなぞった操作が可能である．そのため，タッチインタフェース全般に適用可能であり，携帯情報端末よりも大きいタッチインタフェースにも応用できると我々は考える．もし仮に大画面タッチインタフェースに UnReT́ を適用する場合は，複数ユーザでの使用が想定されるため，selective 又は regional undo の様な局所的に適用される undo/redo [5], [6] が必要となるだろう．

また，現在の試用では指により対象軌跡が隠れる Fat Finger Problems [20] や，図 8c の様に携帯情報端末のボタンを押しながら操作しなければならない問題が生じた．後者のボタン押し問題に関して，Loregian ら [21] は「携帯情報端末において undo ボタンが提供されて欲しいか」をオンラインフォームやメーリングリストを用いてアンケート調査した．その結果，得られた 133 の回答の 28.6%が Yes，48.9%が No，25.6%が Other であり，Yes と Other を回答した人の多くは「undo コマンドを手近に持ちたい」と答えた．この調査を受け携帯情報端末のタッチインタフェースにおいて UnReT́ を用いる場合は，ボタンを押しながら操作するのではなく，1 回の指のストロークのみを用いて操作を行う方が優れていると我々は考えた．具体的にこの操作を設計する場合は，タップの強さを認識する Forcetap [22] や，タップ時の指の接触箇所を認識する TapSense [23] 等の研究が参考になる．

5.2 応用操作

我々は様々なアプリケーション上において UnReT́ を試用した．この試用においても，5.1 節と同様に本論文の第一著者が UnReT́ を試用した．本節では各アプリケーションにおける，UnReT́ の各応用操作をそれぞれ以下に示す．

ドローイング操作 UnReT́ を GIMP におけるドローイング操作に応用させた．その結果，図 9 に示す通常 GIMP では不可能である「色を変更させる前」まで undo/redo 操作を行う技法を発見した．その技法におけるユーザの操作手順を図 9 を用いて説明する．まず，ユーザは図 9a を操作の出発点として，色の変更 (図 9b) と描画操作を行う (図 9c)．その後，ユーザが Ctrl を長押しすると色変更時の軌跡が表示され (図 9d★)，ユーザはその軌跡をなぞることにより色変更前まで undo 操作を行うことが可能である (図 9e)．

また本論文の第一著者は，研究室内におけるゼミや学内セミナー発表において，UnReT́ と GIMP を用いた操作のデモンストレーションを行った．その際に得られたコメントとして「なぞった軌跡のみを undo したい」，「ここまで消えてしまったのか！となるため，そもそもリニアな履歴モデルが嫌いである．UnReT́ を他の履歴モデルには対応しないのか」等のコメントが得られた．

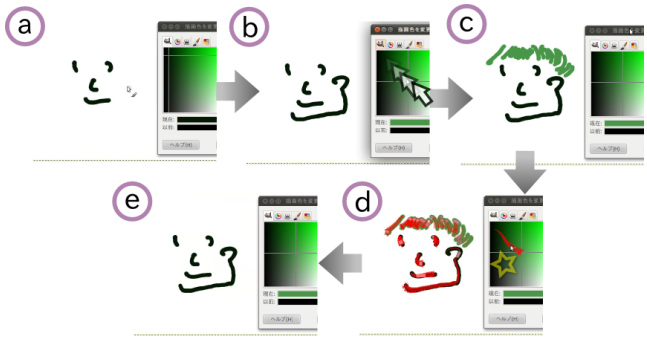

図 9 通常 GIMP では不可能である「色を変更させる前」まで undo/redo 操作を行う．a: 初期状態．b: 色の変更．c: 変更色を用いた描画操作．d: Ctrl を長押しすると ★ の様に色変更時の軌跡が表示される．e: 色を変更させる前まで undo 操作が行われる．

Fig. 9 Undo/redo before “changing a color”, which is not capable in GIMP typically.

undo/redo のためのマーキング操作 undo/redo のためのマーキング操作とは，UIMarks[17] の様に GUI 操作のためのショートカットをユーザ自身が定義可能な操作である．具体的に 3D モデラである blender[*5] にマーキング操作を適用させた図 10 を例にし以下に説明する．まずユーザは，アプリケーションの操作対象外の画面領域にマウスを用いてマークを付ける (図 10a)．その後 3D モデルの変形操作を行い (図 10b)，次いで Ctrl を長押しするとマークした軌跡が表示される (図 10c★)．ユーザはその軌跡をなぞることにより，自身が定義したマークまで undo/redo 操作を行う (図 10d)．

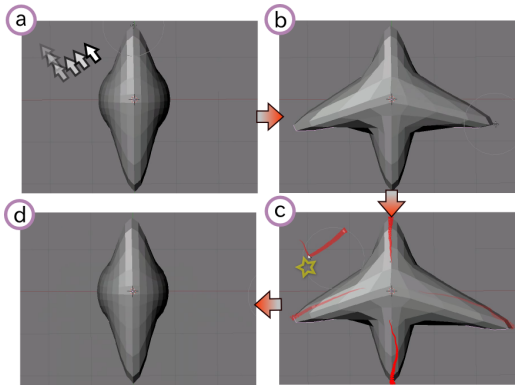

図 10 undo/redo のためのマークをユーザが定義する．a: blender の操作対象外の領域にマークを付ける．b: 3D モデルを変形させる．c: Ctrl を長押しすると ★ の箇所に定義したマークの軌跡が表示される．d: マーキングした時点まで undo/redo する．

Fig. 10 A user defines a mark for undo/redo.

アイコン移動操作 4.3 節に示したマウスの逆操作による undo/redo 処理を用いることにより，一般的に支援されないデスクトップにおけるアイコン移動操作の undo/redo が可能となった．この操作において，まずユーザはアイコンを移動させる (図 11a)．その後 Ctrl を長押しするとアイコン移動の軌跡が表示され (図 11b)，ユーザはその軌跡をなぞることにより，一回のなぞり操作を用いて複数回移動の undo/redo を行う (図 11c)．なおこの様な操作は Sikuli [24] においても，画面スクリーンショットを用いた例示プログラミングにより実現されているが，本応用はプログラミング不要かつ「なぞり」のみにより操作可能である利点がある．

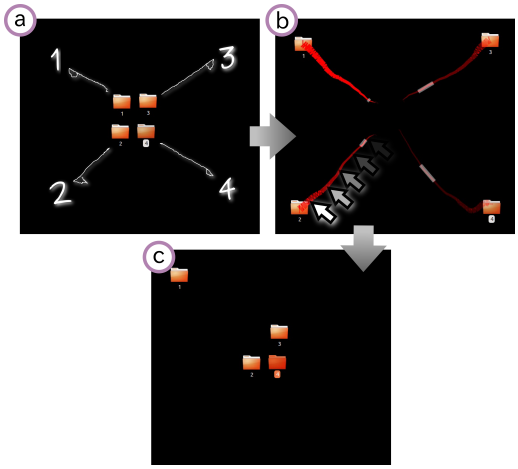

図 11 アイコン移動操作の undo/redo を行う．a: 1，2，3，4 の順にアイコンを移動させる．b: 2 番目に移動させた軌跡をなぞる．c: 2，3，4 の移動操作が undo される．

Fig. 11 Undo/redo icons’ dragging.

チェックボックス選択操作 アイコン移動操作と同様に，本操作もマウスの逆操作による undo/redo 処理により実現された機能である．この操作において，まずユーザは上から順にチェックボックスを選択し (図 12a)，その後 Ctrl を押しながら Ford のチェックボックスをクリックすると (図 12b)，Ford 入力時まで undo/redo される (図 12c)．

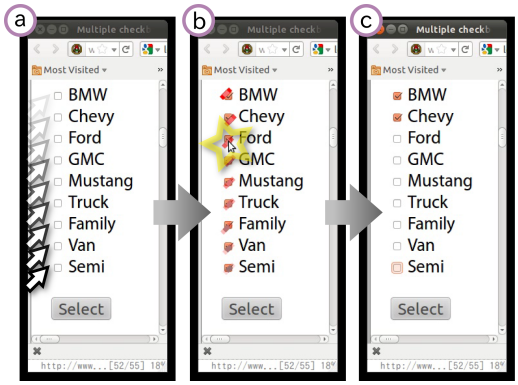

図 12 チェックボックス選択操作の undo/redo を行う．a: 上から順にチェックボックスを選択する．b: Ctrl を押しながら Ford のチェックボックスをクリックする．c: Ford が入力された時点まで undo/redo される．

Fig. 12 Undo/redo checkboxes’ selections.

[*5] http://blender.jp/

5.3 考察

我々は 5.1 節において，大画面タッチインタフェースに対して UnReŕ が応用可能であるアイデアを示した．さらに 5.1 節のドローイング操作において，「なぞった軌跡のみを undo したい」，「UnReŕ を他の履歴モデルには対応しないのか」の様なユーザの意見があることを示した．これらのアイデア・意見に着目すると，単一操作のみ undo/redo する selective undo の様な操作が UnReŕ に必要であると我々は考える．

UnReŕ を用いた selective undo を実現するために，Archer ら [25] が示した script undo を参考に実装する予定である．script undo において，$A_1, ..., A_n$ によって表される操作履歴中の $A_i (1 \leqq i \leqq n)$ 操作のみを undo する場合，$A_n, ..., A_{i+1}, A_i$ までは通常の undo により操作が打ち消され，その後 $A_{i+1}, ..., A_n$ の操作が script により復元される．その結果，selective undo の様な単一操作のみの undo/redo が実現できる．我々は script undo における $A_{i+1}, ..., A_n$ の操作復元を，履歴中の軌跡の再現とすることにより，UnReŕ を用いた selective undo が実現可能であると考える．この機能を実装することにより，「なぞった軌跡のみを undo したい」要求に答えることが可能である．

6. 結論と今後の課題

本稿では，一般的なアプリケーションに広く適用可能な，操作を直接指定し undo/redo を行うインタフェースの有効性調査を目的とした．そのために，我々は「軌跡に基づいた undo/redo インタフェース (UnReŕ)」を用いたインタラクション手法を示した．さらに我々は，様々な環境や操作に UnReŕ を適用させ試用し，そして UnReŕ が有効となり得る場面を探った．

今後我々は，5.3 節にて示した UnReŕ を用いた selective undo を実装することにより，UnReŕ の有効性をさらに調査していきたい．

参考文献

[1] Knister, M. J. and Prakash, A.: DistEdit: A Distributed Toolkit for Supporting Multiple Group Editors, in *Proc. CSCW 1990*, pp. 343–355, ACM (1990).

[2] Kawasaki, Y. and Igarashi, T.: Regional Undo for Spreadsheets, in *Proc. UIST 2004 Demostration Abstract*, ACM (2004).

[3] Berlage, T.: A Selective Undo Mechanism for Graphical User Interfaces Based On Command Objects, *ACM Transactions on Computer-Human Interaction*, Vol. 1, No. 3, pp. 269–294 (1994).

[4] Myers, B. A., Mcdaniel, R. G., Miller, R. C., Ferrency, A. S., Faulring, A., Kyle, B. D., Mickish, A., Klimovitski, A. and Doane, P.: The Amulet Environment: New Models for Effective User Interface Software Development, *IEEE Transactions on Software Engineering*, Vol. 23, No. 6, pp. 347–365 (1997).

[5] Seifried, T., Rendl, C., Haller, M. and Scott, S.: Regional Undo/Redo Techniques for Large Interactive Surfaces, in *Proc. CHI 2012*, pp. 2855–2864, ACM (2012).

[6] Shao, B., Li, D. and Gu, N.: An Algorithm for Selective Undo of Any Operation in Collaborative Applications, in *Proc. GROUP 2010*, pp. 131–140, ACM (2010).

[7] Meng, C., Yasue, M., Imamiya, A. and Mao, X.: Visualizing Histories for Selective Undo and Redo, in *Proc. APCHI 1998*, pp. 459–464, IEEE (1998).

[8] Kurlander, D. and Feiner, S.: A Visual Language for Browsing, Undoing, and Redoing Graphical Interface Commands, in *Visual Languages and Visual Programming*, pp. 257–275, Plenum Press (1990).

[9] Nakamura, T. and Igarashi, T.: An Application-Independent System for Visualizing User Operation History, in *Proc. UIST 2008*, pp. 23–32, ACM (2008).

[10] Vratislav, J.: Cascading undo control, in *Bachelor Thesis*, pp. 1–52, Czech Tecnical University, Prague Faculty of Electrical Engineering (2008).

[11] Bederson, B. B.: Fisheye Menus, in *Proc. UIST 2000*, pp. 217–225, ACM (2000).

[12] Rekimoto, J.: Time-Machine Computing: a Time-centric Approach for the Information Environment, in *Proc. UIST 1999*, pp. 45–54, ACM (1999).

[13] Kelly, S. U. and Davis, P. J.: Desktop History: Time-based Interaction Summaries to Restore Context and Improve Data Access., in *Proc. INTERACT 2003*, pp. 204–211, IOS Press (2003).

[14] Grossman, T., Matejka, J. and Fitzmaurice, G.: Chronicle: Capture, Exploration, and Playback of Document Workflow Histories, in *Proc. UIST 2010*, pp. 143–152, ACM (2010).

[15] Appert, C., Chapuis, O. and Pietriga, E.: Dwell-and-Spring: Undo for Direct Manipulation, in *Proc. CHI 2012*, pp. 1957–1966, ACM (2012).

[16] Kobayashi, M. and Igarashi, T.: Boomerang: Suspendable Drag-and-Drop Interactions Based on a Throw-and-Catch Metaphor, in *Proc. UIST 2007*, pp. 187–190, ACM (2007).

[17] Chapuis, O. and Roussel, N.: UIMarks: Quick Graphical Interaction with Specific Targets, in *Proc. UIST 2010*, pp. 173–182, ACM (2010).

[18] Accot, J. and Zhai, S.: Performance Evaluation of Input Devices in Trajectory-based Tasks: An Application of The Steering Law, in *Proc. CHI 1999*, pp. 466–472, ACM (1999).

[19] Accot, J. and Zhai, S.: Beyond Fitts' Law: Models for Trajectory-Based HCI Tasks, in *Proc. CHI EA 1997*, pp. 250–250, ACM (1997).

[20] Siek, K. A., Rogers, Y. and Connelly, K. H.: Fat Finger Worries: How Older and Younger Users Physically Interact with PDAs, in *Proc. INTERACT 2005*, pp. 267–280, Springer (2005).

[21] Loregian, M.: Undo for Mobile Phones: Does Your Mobile Phone Need an Undo Key? Do You?, in *Proc. NordiCHI 2008*, pp. 274–282, ACM (2008).

[22] Heo, S. and Lee, G.: Forcetap: Extending the Input Vocabulary of Mobile Touch Screens by Adding Tap Gestures, in *Proc. MobileHCI 2011*, pp. 113–122, ACM (2011).

[23] Harrison, C., Schwarz, J. and Hudson, S. E.: TapSense: Enhancing Finger Interaction on Touch Surfaces, in *Proc. UIST 2011*, pp. 627–636, ACM (2011).

[24] Yeh, T., Chang, T.-H. and Miller, R. C.: Sikuli: Using GUI Screenshots for Search and Automation, in *Proc. UIST 2009*, pp. 183–192, ACM (2009).

[25] Archer, J. E., Jr., Conway, R. and Schneider, F. B.: User Recovery and Reversal in Interactive Systems, *ACM Transactions on Programming Languages and Systems (TOPLAS)*, Vol. 6, No. 1, pp. 1–19 (1984).