「マルチメディア，分散，協調とモバイル
(DICOMO2013) シンポジウム」 平成25年7月

デスクトップ上の画面変化に基づく取り消し操作の可視化手法

坂本 有沙[1]　片山 拓也[1]　寺田 努[1,2]　塚本 昌彦[1]

概要：GUI においてユーザの直前の作業内容を取り消す操作は，あらゆるアプリケーションに導入されており，広く普及している．しかし，従来の取り消し操作は，瞬時に操作が実行されるため取り消された部分を見逃したり，作業を一時的に中断した場合や複数の作業を並行して行っている場合に，ユーザが取り消された内容を把握できない状況が生じる．これまでにも取り消し操作を支援する研究が行われているが，それらはいずれもアプリケーションに依存し，汎用的なものは存在しない．そこで本研究では，アプリケーションに依存しない取り消し操作の可視化手法を提案する．提案手法では，取り消し操作に伴う視覚的変化に着目し，デスクトップ上の画面変化から取り消し操作の対象を特定する．本稿では，取り消し範囲と取り消し内容，ディスプレイの表示領域外での取り消し操作の明確化のために強調表示機能と取り消し内容提示機能，取り消し通知機能の実装と評価，考察を行う．また，アプリケーションが保持している過去の UI 操作履歴を用い，アプリケーションに依存しない Redo 機能の実装を行う．

A Method for Visualizing Undo Operations based on Changes on Desktop Screen

Arisa Sakamoto[1]　Takuya Katayama[1]　Tsutomu Terada[1,2]　Masahiko Tsukamoto[1]

1. はじめに

パーソナルコンピュータで利用されている GUI (Graphical User Interface) では，一般に取り消し操作によって簡単に直前の作業を取り消せる．しかし，この取り消し操作は瞬時に起こるため，取り消された対象の位置が分からない，あるいは位置が分かっても取り消された内容を把握できない場合がある．この問題に対し，Microsoft 社のオフィススイートである Office では，「元に戻す」ボタン横のプルダウンメニューを展開すれば，取り消す対象が履歴として提示される．また Adobe 社の画像編集ソフトウェアである Photoshop では，操作履歴を常時視認できるヒストリウィンドウを備えている．これらの機能により，ユーザは取り消される対象を取り消し操作実行前に確認でき，所望の操作までのすべての操作が一括で取り消せる．しかし，これらの機能は，ソフトウェアごとに提示方法が異なるためユーザを困惑させたり，「入力」や「フォントの色」のように操作の表現が簡略化されているため，取り消される作業内容を正確に把握することが難しい．また，ユーザの意図した取り消し操作が行われなかった場合に変化した箇所を見逃したり，取り消し部分がディスプレイ領域外に存在する場合に取り消し操作が実行されたことに気付かない可能性がある．

上記で述べた問題点以外にも，任意の操作のみの取り消しができないという問題や，Google 社の Google Document のように複数のユーザによって編集されるドキュメントに対して，他ユーザの編集の取り消しが行えないという問題点がある．これらの問題点を解決するために，選択的な取り消しを実現した研究 [1] や，取り消し操作にグラフィカルな操作履歴を用いた研究 [2]，共同編集可能な環境での取り消し操作を可能にする研究 [3] などが行われている．しかし，これらは専用のアプリケーションに依存した設計になっており，汎用的に用いることができない．

1 神戸大学大学院工学研究科
Graduate School of Engineering, Kobe University
2 科学技術振興機構さきがけ
PRESTO, Japan Science and Technology Agency

そこで本研究では，アプリケーションに依存しない取り消し操作の可視化手法を提案する．提案手法は，一般に取り消し操作実行時には視覚的変化が伴うという特徴をもとに，デスクトップ上の画面変化を画像マッチング技術を用いて検出し，人の目では見逃してしまう可能性のある小さな変化を含んだ取り消し操作の対象を特定する．構築したシステムは，どこが修正・変更されたか分かりにくい，取り消された操作の詳細が分からない，ディスプレイ領域外での取り消し操作に気付かない，という 3 つの問題点を解決するため，取り消しされた点を強調表示する機能，取り消された入力系列を表示する機能，画面外の取り消しを通知する機能を備える．また，取り消し操作を可視化するために蓄積した操作履歴を活用し，アプリケーションに依存しない拡張やり直し（Redo）機能を提案する．

本稿は以下のように構成されている．2 章で関連研究について述べ，3 章で取り消し操作に伴う画像変化について論じる．4 章で提案手法について述べ，5 章で評価について述べる．6 章で拡張アプリケーションについて述べ，最後に 7 章で本稿のまとめを行う．

2. 関連研究

通常の取り消し操作において，Microsoft 社 Office など多くのアプリケーションでは過去の操作を順に遡って取り消しを実行するため，途中の一部の操作を選択的に取り消せない．それに対して，Adobe 社 Photoshop で実現されているような任意の操作のみの取り消しを可能にする技術は選択的取り消しと呼ばれる．選択的取り消しを用いることで，ユーザはそれまでの操作の実行順序にとらわれず，任意の操作を取り消せる [4]．TRIBASE と呼ばれるプラットフォームで実装された選択的取り消しでは，取り消し対象が取り消し可能かを判断し，すべての操作の関係性を壊さないように配慮している [5]．また，インタラクティブな GUI の開発環境である Amulet[6] では，コマンドオブジェクトを階層化することで，操作を選択的に取り消せるだけでなく，その操作を再利用し同じオブジェクトに対して再実行したり，他のオブジェクトに対して新しく実行できる [7].

この他にも，テキストエディタや表計算ソフトなどに限定されるが，ユーザが選択した領域内で行われた操作のみを取り消すという Regional Undo[8] と呼ばれる手法がある．この手法では，取り消しを適用したい領域を選択できるため，ユーザが想定している対象に対して取り消し操作を行える．また，専用の電子ホワイトボードを用い，作業領域毎の入力の取り消しややり直しを可能にした Regional Undo/Redo[9] も提案されている．この手法では，同じホワイトボード上の異なる領域で同時に作業を行った場合でも，他方の作業を取り消すことなく，自らの作業領域で行った作業だけを取り消せる．さらに，Flatland[10][11] と呼ばれるシステムをもとに拡張されたホワイトボードは，線や図形などオブジェクトごとの履歴を保存することで，特定のオブジェクトに対する操作のみを対象とした取り消し操作や，複数オブジェクトを対象とした操作の取り消しを可能にする [12].

Microsoft 社 Office や Adobe 社 Photoshop において取り消し可能な操作履歴が確認できることを述べたが，これらの履歴は操作の表現が簡略化されているため，ユーザが望むコマンドや状態を見つけるのは困難である．この問題点を解決するため，操作履歴のグラフィック表示やオブジェクトごとの履歴保存を用いて選択的取り消しを実現する手法が提案されている．西田らは，ペイントやブラウザにおいて，取り消し操作を「過去の状態を回復させる操作」ととらえて，すべての過去の作業状態の遷移をわかりやすく提示するためにヒストリグラフを用いている [13]．グラフのノードは作業状態を示し，それに続くノードとその間を結ぶエッジは操作を表し，ノードを選択するとそのノードと対応する作業状態を回復できる．増田らは，選択的取り消しを容易かつ有効に利用するためのインタフェースとして，専用のグラフィカル履歴ブラウザを設計している [14]．この履歴ブラウザでは図形オブジェクトごとに操作コマンドをグループ化し，作業画面のスナップショットから成るグラフィカル履歴をフリップコミックで提供している [15]．取り消したい操作の描かれているスナップショットを選択するだけで希望の操作が取り消される．この時，ユーザは操作の内容や効果を気にすることなく，視覚的に対象のオブジェクトの状態を理解できる．また，コマンドをグループ化したことで，特定のオブジェクトを選択するとそのオブジェクトに関連する作業内容のみを提示するので，取り消したい操作を検出する手間も減少する．

複数のユーザの共同作業においては，操作を行っていないユーザによって取り消し操作が行われると一貫性が崩れてしまうので，適切な修正を加える必要がある．そこで，Vitter は，非線形の取り消し操作が可能な DistEdit[20] ツールキットにおける US&R モデル [21] を用いて，取り消しを行わない部分は現在の状態を保ったまま，取り消しを行う部分のみを取り消すことで，他のユーザの操作によって生じた作業状態を考慮しながら個人的に操作した部分を取り消すことができる手法 [22] を提案している．また，Weiss らの手法 [23] は，Compensation と呼ばれるフレームワークを用いて複数ユーザの操作の一貫性を維持しながら，共有ドキュメントでの取り消し操作を実行できる．

1 章で述べた問題点を解決するために上記の研究が行われているが，これらは専用のアプリケーションに依存した設計である．そこで，汎用的に用いることのできるシステムの提案が必要となる．

また，操作履歴を活用し，過去の状態を回復する研究がいくつか行われている．代表的なものに，暦本が提案する

Time-Machine Computing[16] が挙げられる．この手法では，デスクトップ環境に限定されるが，そこでの変更履歴を時系列で蓄積し，任意時点でのコンピュータの状態を再現できる．Time-Machine Computing を参考に，次の 2 つの研究が行われている．大江らは，時系列での操作履歴の他にデスクトップ全体の状態を撮りため，任意の操作時の状態を閲覧可能にしたインタフェースである Chronicle[17] から着想を得て，UnReT́[18] を提案している．この手法は，記録したマウス操作の軌跡を視覚的に表現し，取り消したい部分の軌跡をなぞることで，なぞられた箇所までのマウス操作を取り消せる．しかし，上記の研究は回復したい操作とその対象が同一のアプリケーションに限定される．一方で，大島らは，簡単なマウス操作でデスクトップの任意の領域をくり抜き，反時計回りのドラッグ操作によって時間を巻き戻し閲覧するツール Rewind-ow[19] を提案している．この手法は，デスクトップ全体の状態を一秒間隔でキャプチャし，そのキャプチャ画像を基に領域の特定や過去の状態の閲覧を可能にしているが，振り返った過去の状態を回復できない．そこで，過去の操作履歴を活用し，アプリケーションに依存せず，操作を回復できるシステムが必要だと考えられる．

3. 取り消し操作に伴う画像変化

本研究では，取り消し操作に伴ってデスクトップ上に視覚的変化が起こることに着目し，デスクトップの画像変化を利用した汎用的な取り消し操作の可視化手法を提案する．提案手法の設計にあたり，以下の 9 つのアプリケーションにおける取り消し操作時の挙動を調査した．今回の調査には，Windows7 搭載の Lenovo ThinkPad X220 (CPU: Core i5 2.50GHz，メモリ: 4.00GB) を用いた．

- Microsoft Office PowerPoint 2007（以下, PowerPoint）
- Microsoft Office Word 2007 （以下，Word）
- Microsoft Office Excel 2007 （以下，Excel）
- ペイント
- エクスプローラ
- メモ帳
- Mozilla Firefox ver. 9.0.1 （以下，Firefox）
- Mozilla Thunderbird （以下，Thunderbird）
- Microsoft Visual Studio 2008（以下，Visual Studio）

各アプリケーションにおいて取り消し操作が可能な操作の一部を**表 1** に示す．常に視覚的変化があるものを○で示し，ユーザが視覚的変化を確認するために条件があるものを△で示す．

表 1 の操作の種類において，テキスト編集はテキストの入力や修正，切り取り，貼り付けを示し，図形編集は図形の移動や削除，複製，挿入，位置やサイズの変更，回転を示す．どのアプリケーションにおいても，テキスト編集や図形編集の他に，コピー・切り取り・貼り付けといった汎用的な機能など，ほぼ同様の操作を取り消し対象にしていることが分かる．しかし，PowerPoint や Word や Excel におけるレイヤ変更では，ユーザが視覚的変化を確認するための条件が存在する．例えば，他のオブジェクトと重なっていないオブジェクトに対して表示レイヤの設定を行う場合や，より前面にあるオブジェクトと重なっていないオブジェクトのレイヤを最前面に切り替える場合は視覚的変化を確認できない．また，通常は視覚的変化を伴う操作であっても，取り消し操作を実行する部分がディスプレイの領域外に存在する場合も視覚的変化を確認できない場合がある．例えば，対象のウィンドウの一部がディスプレイの領域外にあり，ディスプレイ領域外の操作を取り消す場合，PowerPoint，Word，Excel，ペイントにおいては画面上部のツールバーもしくはタイトルバーに視覚的変化が発生する場合があるが，それ以外のアプリケーションではディ

表 1 各アプリケーションにおける取り消し可能な操作

アプリケーション	操作の種類	視覚的変化
PowerPoint	デザイン変更	○
	図形編集	○
	レイヤ変更	△
	スライド編集	○
	ページ設定	○
	アニメーション変更	○
	テキスト編集	○
Word	インデント変更	○
	行間変更	○
	テキスト編集	○
	図形編集	○
	レイヤ変更	△
	箇条書きの変更	○
	段落番号の編集	○
	フォント変更	○
	ワードアート変更	○
	配置変更	○
Excel	セル内テキスト編集	○
	セル編集	○
	グラフ編集	○
	レイヤ変更	△
ペイント	線の描画	○
	塗りつぶし	○
	消しゴムでの消去	○
	図形編集	○
エクスプローラ	ファイル管理	○
メモ帳	テキスト編集	○
Firefox	テキスト編集	○
	Google カレンダー編集	○
	ブックマーク管理	○
Thunderbird	テキスト編集	○
	メール移動	○
	メール削除	○
Visual Studio	テキスト編集	○
	ツール編集	○

スプレイ領域外の操作を取り消した場合，ユーザは視覚的変化を確認できない．さらに，スクロールバーを伴う広範囲にわたるコンテンツにおいて，非表示領域で取り消し操作が行われる場合，PowerPoint や Word，Excel，メモ帳，Visual Studio では，取り消し操作と同時に取り消し部分が確認できるよう自動でスクロールする．それに対して，ペイントや Thunderbird では取り消し操作は行われるが表示領域には移動しない．ユーザが取り消し操作に伴う視覚的変化を確認できない場合，取り消された対象が把握できないだけでなく，取り消し操作が行われたかどうかが把握できない．さらに，ユーザが取り消し操作が行われていないと勘違いをしてしまい，複数回の取り消し操作を実行してしまう問題も発生する．そのため，視覚的に変化があるが，ユーザがそれを確認できない場合でも取り消し操作が行われたことをユーザに知らせる機構が必要になる．

以上の調査結果より，大部分の取り消し操作は視覚的変化を伴うことから，画像差分を用いた取り消し操作の認識によって，アプリケーションやプラットフォームに依存しない，汎用的な取り消し操作の機能拡張が行えると考えられる．一方，画面外にウインドウが存在する場合など単純に画像差分を用いるだけではうまくいかない場合が存在するため，そのような場合でも取り消し内容を把握できる機能を実現する必要がある．

4. 提案手法

本研究では，取り消し操作に伴ってアプリケーションウィンドウに視覚的変化があることを利用し，どこが修正・変更されたか分かりにくい，取り消された操作の詳細が分からない，ディスプレイの領域外での取り消し操作に気付かないという従来の取り消し操作の問題点に対して，以下の 3 つの機能を提案する．

(1) 強調表示機能: ユーザの取り消し操作によって変化した部分に視覚的効果を与えて表示

(2) 取り消し内容提示機能: 取り消し操作によって取り消された入力内容をユーザに提示

(3) 取り消し通知機能: ディスプレイ領域外で取り消し操作が起こったことをユーザに通知

その際，本研究では取り消し操作のトリガとして，取り消しによく用いられているショートカットである Ctrl キーと Z キーの同時押下を用いる．ここで，すべての提案手法において取り消し操作を検知するために提案システムはキーボード入力のフックを行い，Ctrl キーと Z キーが同時に押下された際に，上記の 3 種類の機能を使って取り消し操作を可視化する．以下，それぞれの機能の詳細を述べる．

図 1 赤枠表示モード

4.1 強調表示機能

取り消し部分を明確にするための強調表示機能として，取り消し部分を赤い枠で強調する赤枠表示モードと，取り消し操作を滑らかに提示する滑らか表示モードを実装した．これらの機能によって，取り消し操作によって変化した箇所の特定を容易にすると同時に，取り消された作業内容をユーザが把握しやすくなる．

赤枠表示モード

提案手法では，Ctrl キーと Z キーが同時に押下された瞬間のスクリーンショットを取り消し操作前のディスプレイ状態として取得し，Ctrl キーもしくは Z キーどちらかが離されてから 1 秒後のスクリーンショットを取り消し操作後のものとして取得する．次にそれぞれのスクリーンショットにおける各ピクセルの RGB 値を比較し，RGB 値が異なる領域をそれぞれ囲む，最小の四角形を記録する．これらの四角形を取り消し操作が実行された範囲として設定し赤枠を描画すると**図 1** に示すように赤枠が表示され，ユーザに対し取り消し部分を強調して表示できる．

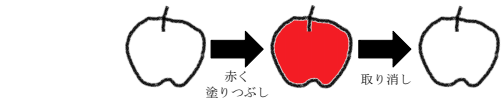
滑らか表示モード

赤枠表示モード同様，Ctrl キーと Z キーが同時に押下された瞬間のスクリーンショットを取り消し操作前のディスプレイ状態として取得し，そのスクリーンショットをディスプレイに重ねて描画する．その描画したスクリーンショットの不透明度を設定することで，次第に取り消し操作実行後のディスプレイが見えるようになる．ここでは不透明度の初期値を 100% に設定した状態から 30 ミリ秒間隔で 1% ずつ減少させていくことで，最終的に不透明度を 0% にし，重ねたスクリーンショットが見えない状態にしている．これによって，通常では**図 2(a)** に示すように実行される取り消し操作が，**図 2(b)** に示すように瞬時に実行された取り消し操作をまるで徐々に実行されているかのようにユーザに提示できる．ここで，左のスクリーンショットが取り消し操作を行う前の状態を表し，右のスクリーンショットが滑らか表示モードが完了した状態，つまり取り消し操作後の状態を表す．

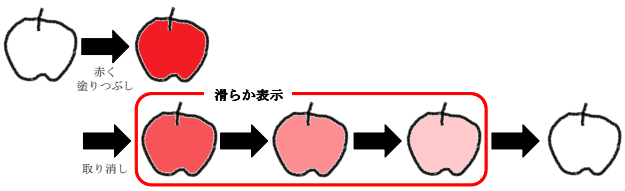
4.2 取り消し内容提示機能

ユーザに取り消された内容を提示することで，ユーザは自身の過去の操作を視覚的に把握できる．

従来の取り消し操作は，コンテンツの変化のみに対応し



(a) 通常の取り消し操作



(b) 提案手法

図 2 滑らか表示モード

ている．そのため，ペイントにおける線の色・太さの選択，文書作成におけるフォントやサイズ・効果の選択など，直接コンテンツに影響を与えない変化には非対応で，過去の状態に戻すことが難しい場合がある．そこで，システムフックを用いて記録したキーボードとマウスの操作履歴と画像差分を用いて，取り消し操作によって戻された期間を特定し，その期間のキーボードとマウスの入力内容をユーザに提示する．

入力の記録

本機能は，キーボードとマウスの入力を提示する．提案システムはそれぞれの入力イベントをフックし，1 秒間隔でその時のスクリーンショットと関連付けて保存する．処理時間短縮のため，60 秒間の入力イベントとスクリーンショットをメモリ上に保持する．また，スクリーンショットは，実行アプリケーションに無関係な画面変化を可能な限り除外するため，アクティブなウィンドウのみを記録する．ここで，記録するキーボードとマウスの以下の入力イベントはそれぞれ以下の通りである．

- キーボード: 各キーの KeyDown イベント
- マウス: MouseDown イベント，MouseUp イベント，MouseMove イベント

なお，マウスの入力はこの 3 種類のイベントを組み合わせることで通常のカーソル移動とドラッグを識別できる．

期間の特定

提案手法は，取り消し操作後のスクリーンショットに類似したスクリーンショットを蓄積した画面遷移の履歴から探索することで，取り消された期間を特定する．比較方法には，それぞれのスクリーンショットに対するヒストグラムにおける，Bhattacharyya 距離の算出を用いる．Bhattacharyya 距離とは，画像間の類似性評価指数として代表的なもので，統計学において，2 つの離散的確率分布間の距離を求める尺度として用いられている指標である．ここで，R は類似度，H_1, H_2 はそれぞれのヒストグラムの成分集合，N はヒストグラムのビンの総数を表す．

(a) 取り消し操作後　(b) 4 秒前

(c) 11 秒前　(d) 16 秒前

図 3 スクリーンショットの比較

表 2 各スクリーンショットの Bhattacharyya 距離

スクリーンショット	R
取り消し操作 4 秒前 (図 3(b))	2.086×10^{-3}
取り消し操作 11 秒前 (図 3(c))	9.153×10^{-4}
取り消し操作 16 秒前 (図 3(d))	0.000

$$R = \sqrt{1 - \frac{1}{\sqrt{\bar{H}_1 \bar{H}_2 N^2}} \sum_I \sqrt{H_1(I) H_2(I)}} \tag{1}$$

$$\left(\because \bar{H}_k = \frac{1}{N} \sum_J H_k(J) \right) \tag{2}$$

式 1 において，$R = 0$ の場合，比較する 2 つのスクリーンショットが一致していることを表す．取り消し操作実行後のスクリーンショットは，取り消された操作を行う前のものと最も類似すると考えられるため，R が最小の時点を操作が取り消された時点とした．ここで，比較時間短縮のため，次のような条件を設けた．

1. $R = 0$ となった場合: その時点を取消し操作実行前とし，比較を終了
2. $R = 0$ とならなかった場合: $R < 0.05$ を満たす，最小の R の時点を取消し操作実行前とし，$R < 0.05$ を満たして以降に $R > 0.05$ となった時点で比較を終了

実行例として，オレンジの実線で囲んだ部分の 3 本の黒い実線を消しゴムで消す作業を取り消した様子を**図 3** に示す．ここで，図 3 のキャプションは取り消し操作の何秒前かを表す．取り消し操作後のスクリーンショット (**図 3(a)**) と取り消し操作前の各スクリーンショット (**図 3(b)**, **図 3(c)**, **図 3(d)**) との Bhattacharyya 距離の値 R を**表 2** に示す．図 3(d) のスクリーンショット比較時に $R = 0$ となり，条件 1 を満たすため，この時点で比較が終了される．この例では，取り消し操作によって 16 秒前まで戻ったと判断する．

内容の提示

上記で得られた取り消し操作によって戻された期間をも

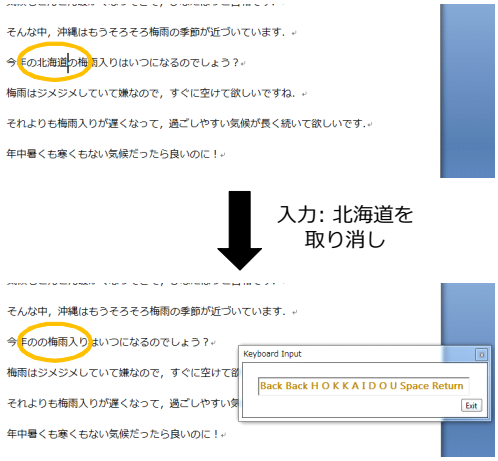


図 4　取り消し内容提示機能: キーイベント

とに，取り消し内容の提示を行う．この時間以降に記録されたスクリーンショットに関連付けられているキーボードとマウスの入力履歴を結合し，得られた一連の入力履歴を**図 4** や**図 5** のように提示する．ここで，図 4 における例はキーボードの入力イベント履歴を提示するもの，図 5 はマウスの入力イベント履歴を提示し，マウスの軌跡を線で提示しているものである．キーボード入力は，打鍵されたキーの系列を提示し，同時に変化した部分を枠で囲って示す．図 4 の場合，オレンジで囲んだ部分において，元々入力されていた文字の削除の BackSpace キーとアルファベットキー，変換の Space キー，確定の Enter キーが入力されたことを提示している．また，マウス入力はドラッグを水色の実線で示し，マウスボタンのダウンイベントがあった点を赤色，アップイベントがあった点をピンク色で示す．図 5 の場合，マウス操作として線の描画開始時と終了時のクリック，描画のためのドラッグ操作があったことを提示している．

4.3　取り消し通知機能

3 章で述べたように，従来の取り消し操作は，ディスプレイ領域外で取り消し操作が行われた場合，操作によるデスクトップ画面の変化を確認できない．また，一部のアプリケーションにおけるツールバーやクイックアクセスツールバーのグラフィックの変化により取り消し操作の実行を確認できるが，その内容については確認できない．そこで，ディスプレイ領域外で取り消し操作が実行された場合のみユーザに通知し，ユーザが任意に取り消された内容を取得できる機能を提案する．本稿では，ウィンドウの点滅により取り消し操作実行を通知することとした．具体的には，Ctrl キーと Z キーが同時に押下された瞬間にアクティブなウィンドウのハンドルを取得し，そのハンドルをもつウィンドウに対して画面外の取り消し操作が行われたことを検出するとタスクバーボタンおよびタイトルバーを点滅させる．

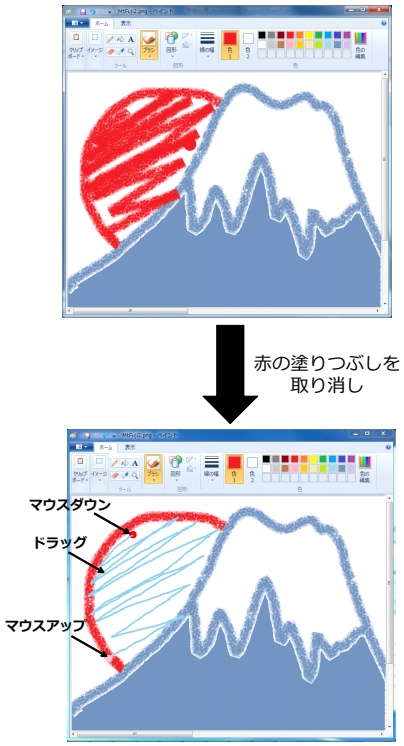


図 5　取り消し内容提示機能: マウスイベント

ディスプレイ領域外のウィンドウをキャプチャするために，非表示領域のスクリーンショットが取得できる Win32 API の PrintWindow 関数を用いた．具体的には，アクティブになったウィンドウのスクリーンショットを，アプリケーションごとに記録し続ける．提案システムは，取り消し操作が行われた際にアクティブなウィンドウのスクリーンショットとその履歴を順次比較し，取り消された位置と内容を特定する．その際，取り消された位置がディスプレイの領域外にある際にはウィンドウを点滅させている．

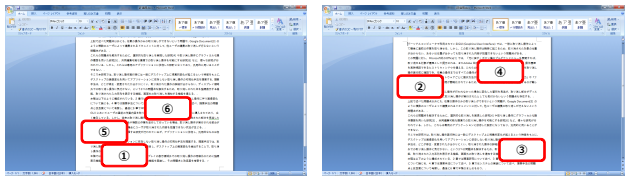
5.　評価

提案システムのプロトタイプを実装し，従来の取り消し操作と強調表示機能，取り消し内容提示機能，取り消し通知機能を実行した取り消し操作との比較実験を行い，提案手法の有効性を評価した．実装と評価の際に，Microsoft Windows7 OS 搭載の Lenovo ThinkPad X220 (CPU: Core i5 2.50GHz，メモリ: 4.00GB) を用い，Microsoft Visual C#2008 により，機能を実現した．

5.1　実験方法

実験では，日常的に利用するアプリケーションにおいて，取り消し操作による視覚的変化の把握が困難であろう状況を設定し，以下に示す従来の取り消し操作と 5 つの提案手法を用いた取り消し操作との把握精度の比較実験を行った．

- 方法 a: 提案手法なし（従来）
- 方法 b: 赤枠表示モード
- 方法 c: 滑らか表示モード



(a) 状況 1　　(b) 状況 2

図 6　比較実験の状況: Word

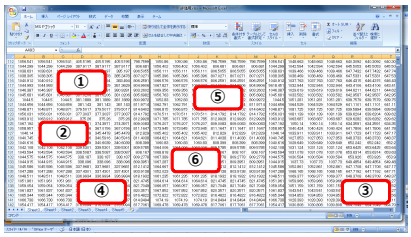
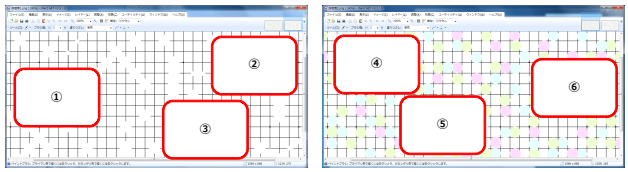
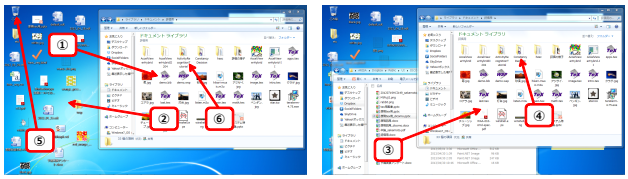


図 7　比較実験の状況: Excel



(a) 状況 1　　(b) 状況 2

図 8　比較実験の状況: ペイント



(a) 状況 1　　(b) 状況 2

図 9　比較実験の状況: エクスプローラ・デスクトップ

- 方法 d: 方法 b と取り消し内容提示機能の併用
- 方法 e: 方法 c と取り消し内容提示機能の併用
- 方法 f: 取り消し通知機能

今回，4 つのアプリケーションにおいて，以下のような操作を取り消し操作の対象とした．

I. Word: 8〜20 文字のテキスト入力
II. Excel: 4〜6 桁のランダムな数字入力
III. ペイント: 点や線の描画
IV. エクスプローラとデスクトップ: ファイル移動

被験者には，各機能の説明を口頭でのみ行い，1 つのアプリケーションにつき，6 つの異なる箇所での作業に対して，Ctrl キーと Z キーの同時押下で取り消し操作を行なってもらう．詳細な取り消し操作実行箇所として，Word では**図 6**，Excel では**図 7**，ペイントでは**図 8**，エクスプローラおよびデスクトップでは**図 9** のように設定した．図 9 における矢印は，ファイル移動の始点から終点を表している．

ここで，取り消し操作の対象の作業は，被験者に見られないように毎回入力する．そして，1 試行終了毎に，被験者に変化部分や変化内容を正確に把握できたかを口頭で回答してもらい，1 つのアプリケーションでの試行終了毎に，方法 a〜方法 e について，取り消し操作が分かりやすかった場合を 5，分かりづらかった場合を 1 とする，5 段階評価でアンケート用紙に回答してもらう．方法 f について，表示領域外での取り消し操作の提示方法として妥当であれば 5，不適であれば 1 とする，5 段階評価でアンケート用紙に回答してもらう．すべての試行終了後，アプリケーション毎に最適な方法を 1 つずつ選択してもらう．全試行を通して，被験者には積極的に提案手法の問題点や改善点，優れた点等を発言・記入してもらった．この時，取り消し操作の対象の作業（①〜⑥）の順序はすべての被験者に対して同様だが，取り消し操作時に実行される機能（方法 a〜方法 f）の順序は被験者により異なる．被験者は 23〜25 歳の取り消し操作を日常的に利用する男性 6 人である．

5.2　実験結果

まず，比較実験の結果について議論する．それぞれのアプリケーションにおける各方法での取り消し操作の位置把握精度を**表 3** に，内容把握精度を**表 4** に示す．ここで，表 3 において，正確な位置を把握できた場合を 2，大まかな位置を把握できた場合を 1，全く把握できなかった場合を 0 とした．また，表 4 において，取り消された内容を正確に把握できた場合を 3，内容の一部は正確に把握できたが，その他は正確に把握できなかった場合を 2，大まかに内容を把握できた場合を 1，全く把握できなかった場合を 0 とした．表 3 の平均値を基に，有意水準 5%の分散分析を行った結果，すべてのアプリケーションにおいて，すべての方法間に有意差がなかった (II: $F(4, 25)=1.01$, $MSe=0.45$, n.s.，III: $F(4, 25)=2.55$, $MSe=0.36$, n.s.，IV: $F(4, 25)=2.12$, $MSe=0.22$, n.s.)．ここで，I において，全被験者がすべての方法で同じ結果となったため，有意差がないことは明らかである．これにより，提案手法は，実行されるアプリケーションが明確であり，画面上に対象のアプリケーションウィンドウしか開かれていない場合の取り消し操作の位置把握に寄与していないと言える．この理由として，実行アプリケーションの限定により，被験者が変更部分の予測を立てられ，今回の実験環境が把握しやすいものであった可能性が高い．そこで今後，画面上に複数のアプリケーションウィンドウが開かれており，実行されるアプリケーションが不明な状態で評価を行う必要がある．また，従来の取り消し操作の仕様により，一部のアプリケーションにおいて位置把握が支援されていることが挙げられる．具体的に，取り消し操作実行時，I においてはカレット，II においては選択中のセルを囲う黒い枠の取り消された位置への移動が起こっていた．一方で，表 4 の平均値を基に，有意水準 5%の分散分析を行った結果，すべてのアプリケーションにおいて有意差があった

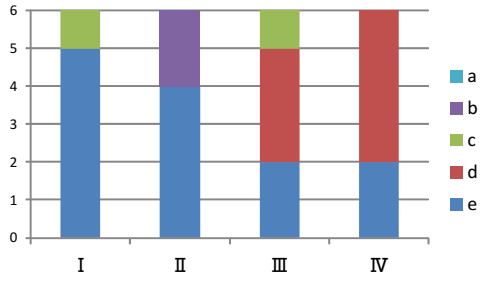


図 10　各アプリケーションにおける最適な提示方法

(I: F(4, 25)=29.15, MSe=0.33, p<.05, II: F(4, 25)=18.67, MSe=0.44, p<.05, III: F(4, 25)=25.40, MSe=0.29, p<.05, IV: F(4, 25)=17.67, MSe=0.30, p<.05)．そこで，各方法間での有意水準 5%の t 検定を行った結果, すべてのアプリケーションにおいて，方法 d は方法 a に対して有意差があった (I: t(5)=7.05, p<.05, II: t(5)=4.39, p<.05, III: t(5)=7.05, p<.05, IV: t(5)=5.48, p<.05)．また，すべてのアプリケーションにおいて，方法 e も方法 a に対して有意差があった (I: t(5)=12.65, p<.05, II: t(5)=8.00, p<.05, III: t(5)=7.05, p<.05, IV: t(5)=7.00, p<.05)．これにより, 取り消された作業内容の視覚的な提示が有効であり, 取り消し操作の把握を支援できると言える．他にも, すべてのアプリケーションにおいて, 方法 b と方法 d (I: t(5)=11.07, p<.05, II: t(5)=7.91, p<.05, III: t(5)=5.97, p<.05, IV: t(5)=7.00, p<.05), 方法 b と方法 e (I: t(5)=17.00, p<.05, II: t(5)=17.00, p<.05, III: t(5)=5.97, p<.05, IV: t(5)=6.71, p<.05)，方法 c と方法 e (I: t(5)=2.91, p<.05, II: t(5)=4.47, p<.05, III: t(5)=5.48, p<.05, IV: t(5)=6.32, p<.05) で有意差があった．I において方法 a と方法 c (t(5)=4.39, p<.05) や方法 b と方法 c (t(5)=3.95, p<.05), II において方法 d と方法 e (t(5)=7.00, p<.05)，III において方法 c と方法 d (t(5)=7.75, p<.05)，IV において方法 a と方法 b (t(5)=2.71, p<.05) や方法 a と方法 c (t(5)=2.74, p<.05) で有意差があった．この結果より，強調表示機能単体での利用より取り消し内容提示機能との併用が有効であり，取り消し内容提示機能の有無の条件が同じ場合，赤枠表示モードより滑らか表示モードが有効だと考えられる．

次に，被験者に行ったアンケート結果について議論する．それぞれのアプリケーションにおける各方法での取り消し操作の分かりやすさを**表 5** に，各アプリケーションでの最適な提示手法に選ばれた，各方法の得票数を**図 10** に示す．

表 5 の平均点を基に，有意水準 5%の分散分析を行った結果，すべてのアプリケーションにおいて有意差があった (I: F(4, 25)=13.56, MSe=0.52, p<.05, II: F(4, 25)=4.60, MSe=0.87, p<.05, III: F(4, 25)=5.31, MSe=1.11, p<.05, IV: F(4, 25)=7.50, MSe=0.64, p<.05)．そこで，各方法間での有意水準 5%の t 検定を行った結果，すべてのアプリケーションにおいて，方法 a に対して方法 c (I: t(5)=11.18, p<.05, II: t(5)=4.57, p<.05, III: t(5)=3.95, p<.05, IV: t(5)=4.47, p<.05)，方法 d (I: t(5)=2.73, p<.05, II: t(5)=3.50, p<.05, III: t(5)=3.95, p<.05, IV: t(5)=3.99, p<.05)，方法 e (I: t(5)=5.00, p<.05, II: t(5)=4.57, p<.05, III: t(5)=5.53, p<.05, IV: t(5)=3.87, p<.05) は有意差があった．方法 a に対して方法 b は，II において有意差がなかった (I: t(5)=3.87, p<.05, II: t(5)=2.00, n.s., III: t(5)=3.84, p<.05, IV: t(5)=3.95, p<.05)．この理由として，例えば，ツールバーの変化や選択中のセルを囲う黒い枠の移動等，取り消し操作による変化部分以外の画面変化に対して赤枠表示が実行され，ユーザの把握を阻害したことが挙げられる．一方で，方法 b に取り消し内容提示機能を併用した方法 d が方法 a に対して有意差があった理由として，変化部分付近に取り消された内容を提示するため，他の赤枠表示に惑わされず，正確な変化部分が把握できたためだと考えられる．他にも，I において方法 b と方法 c (t(5)=6.71, p<.05)，方法 c と方法 d (t(5)=4.39, p<.05)，方法 d と方法 e (t(5)=6.71, p<.05) で有意差があった．この結果より，I において，赤枠表示モードより滑らか表示モードが有効だと考えられる．また，表 5 と図 10 の結果より，被験者が分かりやすいと考える方法と各アプリケーションで最適だと考える手法は異なることが分かった．双方で評価の良かった項目は，I においては方法 c と方法 e, III においては方法 d と方法 e であった．しかし，II においては方法 e に加えて，前者では方法 c，後者では方法 b が優位であった．また，IV においては，方法 d と方法 e に加えて，前者では方法 c が優位であった．今後，追加で聞き取り調査を行い，その原因の明確化が必要だと考えられる．

最後に，方法 f の評価について議論する．比較対象は方法 a のみとし，この方法では表示領域外の取り消し操作に気付けないものと仮定する．それぞれのアプリケーションにおける，表示領域外での取り消し操作の把握精度を**表 6** に，表示領域外での取り消し操作の提示方法としての妥当性のアンケート結果を**表 7** に示す．ここで，表 6 において，表示領域外で取り消し操作があったことに気付けた場合を 1，気付けなかった場合を 0 とした．表 6 の結果より，すべてのアプリケーションにおいて，方法 f は高い把握精度を示した．また，表 7 の結果より，すべてのアプリケーションにおいて，点滅での取り消し操作の通知は不適傾向を示した．この理由として，変化部分の把握に集中しているため点滅に気付かず，表示領域外での取り消し操作だと認識できていなかったことが挙げられる．以上より，表示領域外での取り消し操作の把握を支援する手法が有効であるが，今後点滅以外の手法でより把握を支援できる方法を検討する必要がある．また，この場合，取り消される対象が表示されていないため，よりユーザの理解を助ける内容の提示方法が必要だと考えられる．

今回，実行アプリケーションを軸に評価を行ったが，取

表 3 取り消し操作の位置把握精度

	I				II				III				IV			
	0	1	2	ave.	0	1	2	ave.	0	1	2	ave.	0	1	2	ave.
a	0	0	6	2.00	1	2	3	1.33	1	2	3	1.33	1	2	3	1.33
b	0	0	6	2.00	1	1	4	1.50	0	0	6	2.00	0	2	4	1.67
c	0	0	6	2.00	0	0	6	2.00	2	1	3	1.17	0	0	6	2.00
d	0	0	6	2.00	1	1	4	1.50	0	0	6	2.00	0	0	6	2.00
e	0	0	6	2.00	0	1	5	1.83	0	1	5	1.83	0	1	5	1.83

0: 全く把握できなかった，1: 大まかに把握できた，2: 正確に把握できた

表 4 取り消し操作の内容把握精度

	I					II					III					IV				
	0	1	2	3	ave.	0	1	2	3	ave.	0	1	2	3	ave.	0	1	2	3	ave.
a	5	1	0	0	0.17	5	0	1	0	0.33	4	2	0	0	0.33	3	2	1	0	0.67
b	6	0	0	0	0	5	1	0	0	0.17	2	4	0	0	0.67	0	3	3	0	1.50
c	0	4	0	2	1.67	2	3	0	1	1.00	3	3	0	0	0.50	0	2	4	0	1.67
d	0	0	4	2	2.33	0	1	5	0	1.83	0	0	3	3	2.50	0	0	2	4	2.67
e	0	0	1	5	2.83	0	0	0	6	3.00	0	0	3	3	2.50	0	0	0	6	3.00

0: 全く把握できなかった，1: 大まかに把握できた，2: 一部正確に把握できた，3: 正確に把握できた

り消し操作による変化部分の違いや方法の実行順序の違いによる影響を考慮し，有意差が見られないか今後検証する必要がある．

5.3 考察

プロトタイプを著者らが検証した結果と被験者へのアンケート調査結果から，提案手法の問題点と改善策を考察する．

取り消し操作に対する挙動の差

取り消し操作に対する挙動はアプリケーションごとに異なる．例えば，取り消し操作を連続的に行った場合，多くのアプリケーションでは取り消し可能な操作が存在しなくなるまで取り消し操作を行い，それ以降は何も起こらないが，Excel でのセル内のテキスト編集や，メモ帳におけるテキスト編集では，直前の操作のみ取り消しが可能であり，連続的に取り消すことはできない．提案システムでは，ユーザの入力履歴とスクリーンショットの変化を合わせて保持しているため，ユーザの入力を遡らせるような入力をエミュレートすることで，従来の取り消し操作では取り消せなかった範囲の取り消し操作を実行できる可能性がある．

また，現在の実装では，取り消し内容提示機能において，連続的な取り消し操作の再現に対応できていない．今後，操作履歴を更に遡り，二度目以降の取り消し操作に対応できる機構を検討する必要がある．

可視化方法の改善

赤枠表示モードにおいて，ユーザの操作以外の画面変化もユーザの操作による変化と誤認して，実際の取り消し操作が実行された部分と異なる範囲に赤枠を表示してしまうことがある．今後，これらの問題点の解決方法を検討する．

滑らか表示モードにおいて，変化部分の把握が遅れると変化内容の把握までは困難であった．そこで，次回評価時に赤枠表示機能と併用し，より把握を支援できないかを検証する．

取り消し内容提示機能において，記録した入力情報をユーザが容易に理解できるよう，直観的な提示方法を検討する．

実使用を考慮した実装

現在の実装では，システム実行開始時にユーザが提示に用いる機能をあらかじめ選択する必要があり，実行されるアプリケーション毎に機能を選択できない．また，システム実行中は Ctrl キーと Z キーの同時押下を検出すると，ユーザの要不要に関わらず機能を実行し，ユーザの作業を妨げる恐れがあり，実使用を考慮できていない．これらを解決するため，取り消し操作が実行されるアプリケーションやその操作の範囲，キー入力かマウス入力かの操作の種類，取り消し操作によって遡る期間から，システム側で機能実行の有無や実行する機能を自動で選択できる機構の実装を今後の課題とする．

6. 拡張アプリケーション

ユーザが取り消し操作を実行する際，取り消された作業の条件を一部のみ変更し，再実行したい場合が少なくない．例えば，ペイントソフトにおいて，綺麗な円形を描画したいが，線を何度描画しても上手くいかない場合やせっかく綺麗な円形を描画できたが，サイズや位置を少しだけ変更

表 5 取り消し操作の分かりやすさのアンケート結果

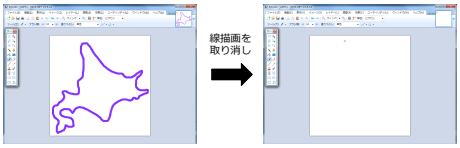
	I						II						III						IV					
	1	2	3	4	5	ave.	1	2	3	4	5	ave.	1	2	3	4	5	ave.	1	2	3	4	5	ave.
a	1	4	1	0	0	2.00	2	3	1	0	0	1.83	2	3	1	0	0	1.83	2	3	1	0	0	1.83
b	0	2	2	2	0	3.00	0	2	2	1	1	3.17	0	0	2	4	0	3.67	0	0	3	3	0	3.50
c	0	0	0	3	3	4.50	0	0	3	2	1	3.67	1	0	2	1	2	3.50	0	0	2	3	1	3.83
d	0	1	4	1	0	3.00	0	1	2	3	0	3.33	0	1	0	1	4	4.33	0	1	0	3	2	4.00
e	0	0	1	1	4	4.50	0	1	1	3	1	3.676	0	0	2	1	3	4.17	0	0	2	3	1	3.83

1: 分かりづらかった，2: 少し分かりづらかった
3: どちらともいえない，4: 少し分かりやすかった，5: 分かりやすかった

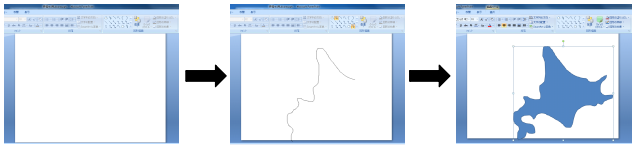
表 6 表示領域外への取り消し操作の把握精度

I			II			III			IV		
0	1	ave.	0	1	ave.	0	1	ave.	0	1	ave.
1	5	0.83	1	5	0.83	2	4	0.67	1	5	0.83

0: 把握できなかった，1: 把握できた



(a) Paint.NET での取り消し操作



(b) PowerPoint での提案手法の実行

図 11 やり直し機能

したい場合が挙げられる．そこで拡張機能として，前述の提案手法で用いた操作履歴を用いて，アプリケーションに依存しないやり直し機能を提案する．

提案機能では，各アプリケーションに備わる固有のやり直し機能のように，取り消された操作をそのまま回復させるのではなく，**図 11** のようなユーザが再現したいウィンドウを指定し，再現する位置やサイズなどのパラメータを変更した上でやり直す．ここで，**図 11(a)** は，Paint.NET での取り消し操作を示し，**図 11(b)** は PowerPoint を指定した場合の提案手法の実行を示す．このように，従来拡縮すると劣化してしまう画像ファイル形式での描画図形の引用が，提案機能を使うと，図形を劣化させることなく，他のアプリケーションで再現できる．

本機能は，取り消し内容提示機能で得られた入力履歴に記録されたキーボードやマウスの入力を基に，キーボードとマウスの入力をエミュレートして指定されたアプリケーション上に操作を再現する．この時，再現するアプリケーションの指定は，指定するアプリケーションのウィンドウクリックで行い，再現する位置やサイズの指定は，取り消し内容提示機能で提示した軌跡描画のマウスドラッグ操作で行う．機能実行のトリガとして，今回ショートカットキーとして割り当てのない Shift キーと Y キーの同時押下を用い，再現条件の指定を行う．

7. おわりに

本研究では，デスクトップ上の画面変化とユーザインタフェースの操作履歴を用いることで，アプリケーションに依存せず，取り消し操作を可視化するシステムを提案した．本稿では，どこが修正・変更されたか分かりにくい，取り消された操作の詳細が分からない，画面外の取り消し操作に気付かないという問題点に対して，強調表示機能，取り消し内容提示機能，取り消し通知機能の実装と評価を行った．評価の結果，提案手法は位置把握の支援では不十分だが，内容把握の支援で有効だと分かった．また，保持している過去履歴を用いた，アプリケーションに依存しない拡張やり直し機能の実装を行った．今後は，自らのプロトタイプ利用結果とアンケート評価結果から明らかになった問題点を改善し，更なる評価を行う．そして，取り消し操作における問題点を解決し，ユーザの操作をサポートする，より快適な取り消し操作を提供する．

謝辞 本研究の一部は，科学技術振興機構戦略的創造研究推進事業 (さきがけ) によるものである．ここに記して謝意を表す．

参考文献

[1] A. G. Cass and C. S. T. Fernandes: Using Task Models for Cascading Selective Undo, *Proc. of the 5th International Workshop on Task Models and Diagrams for Users Interface Design (TAMODIA2006)*, pp. 186–201 (2006).

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

[3] D. Chen and C. Sun: Undoing Any Operation in Collaborative Graphics Editing, *Proc. of the 2001 Inter-*

表 7 方法 f の妥当性のアンケート結果

I						II						III						IV					
1	2	3	4	5	ave.	1	2	3	4	5	ave.	1	2	3	4	5	ave.	1	2	3	4	5	ave.
2	0	2	2	0	2.67	1	1	2	2	0	2.83	2	1	1	2	0	2.50	2	2	0	2	0	2.33

1: 不適，2: やや不適，3: どちらともいえない，4: やや妥当，5: 妥当

national ACM SIGGROUP Conference on Supporting Group Work (GROUP2001), pp. 197–206 (2001).

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

[5] C. Zhou and A. Imamiya: Object-based Nonlinear Undo Model, *Proc. of the 21th Computer Software and Applications Conference (COMPSAC1997)*, pp. 50–55 (1997).

[6] B. A. Myers, R. McDaniel, R. C. Miller, A. S. Ferrency, A. D. Faulring, B. Kyle, A. Mickish, A. Klimovitski, and P. Doans: The Amulet Environment: New Models for Effective User Interface Software Development, *IEEE Transaction on Software Engineering*, Vol. 23, No. 6, pp. 347–364 (1997).

[7] B. A. Myers and D. S. Kosbie: Reusable Hierarchical Command Objects, *Proc. of SIGCHI Conference on Human Factors in Computing Systems (CHI1996)*, pp. 260–267 (1996).

[8] Y. Kawasaki and T. Igarashi: Regional Undo for Spreadsheets, *Proc. of the 17th Annual ACM Symposium on User Interface Software and Technology (UIST2004) Demonstration Abstract*, (2004).

[9] T. Seifried, C. Rendl, M. Haller and S. Scott: Regional Undo / Redo Techniques for Large Interactive Surfaces, *Proc. of SIGCHI Conference on Human Factors in Computing Systems (CHI2012)*, pp. 2855–2864, (2012).

[10] T. Igarashi, W. K. Edwards, A. LaMarca, and E. D. Mynatt: An Architecture for Pen-based Interaction on Electronic Whiteboards, *Proc. of International Working Conference on Advanced Visual Interfaces (AVI2000)*, pp. 68–75 (2000).

[11] E. D. Mynatt, T. Igarashi, W. K. Edwards, and A. LaMarca: Flatland: New Dimensions in Office Whiteboards, *Proc. of SIGCHI Conference on Human Factors in Computing Systems (CHI1999)*, pp. 346–353 (1999).

[12] W. K. Edwards, T. Igarashi, A. LaMarca, and E. D. Mynatt: A Temporal Model for Multi-Level Undo and Redo, *Proc. of the 13th Annual ACM Symposium on User Interface Software and Technology (UIST2000)*, pp. 31–40 (2000).

[13] 西田知博, 林 真志, 辻野嘉宏, 都倉信樹: ヒストリーグラフを用いたアンドゥ機構の提案と評価, 情報処理学会研究報告, ヒューマンインタフェース研究会報告, Vol. 99, No. 69, pp. 67–72 (1999).

[14] 増田尚則, 今宮淳美: Undo 機能をもつグラフィカル履歴ブラウザ設計と視覚的探索方法, 電子情報通信学会論文誌, Vol. J85-D1, No. 8, pp. 798–810 (2002).

[15] C. Meng, M. Yasue, A. Imamiya, and X. Mao: Visualizing Histories for Selective Undo and Redo, *Proc. of the 3rd Asia Pacific Computer and Human Interaction (APCHI1998)*, pp. 459–464 (1998).

[16] J. Rekimoto: Time-Machine Computing: a Time-centric Approach for the Information Environment, *Proc. of the 12th Annual ACM Symposium on User Interface Software and Technology (UIST1999)*, pp. 45–54, (1999).

[17] T. Grossman, J. Matejka and G. Fitzmaurice: Chronicle: Capture, Exploration, and Playback of Document Workflow Histories, *Proc. of the 23th Annual ACM Symposium on User Interface Software and Technology (UIST2010)*, pp. 143–152, (2010).

[18] 大江龍人, 志築文太郎, 田中二郎: 軌跡に基づいた undo/redo インタフェース, 情報処理学会研究報告, ヒューマンコンピュータインタラクション研究会報告, Vol. 149, No.7, pp.1–8, (2012).

[19] 大島裕樹, 宮下芳明: Rewind-ow: 特定領域の時間を巻き戻して閲覧するツール, 情報処理学会, インタラクション 2013 論文 DVD, (2013).

[20] M. J. Knister and A. Prakash: DistEdit: A Distributed Toolkit for Supporting Multiple Group Editors, *Proc. of ACM conference on Computer-Supported Cooperative Work (CSCW1990)*, pp. 343–355, (1990).

[21] J. S. Vitter: US&R: A New Framework for Redoing, *Journal of IEEE Software*, Vol. 1, No. 4, pp. 39–52 (1984).

[22] A. Prakash and M. J. Knister: A Framework for Undoing Actions in Collaborative Systems, *ACM Transaction on Computer-Human Interaction (TOCHI)*, Vol. 1, No. 4, pp. 295–330 (1994).

[23] S. Weiss, P. Urso, and P. Molli: A Flexible Undo Framework for Collaborative Editing, *Technical Report of Institut National de Recherche en Informatique (INRIA)*, No. 6516 (2008).