

WS におけるユーザ操作を記録・再現するツール

ORRIX : 技術的現状と課題

宮内 崇 大森 晃

takamiya@ms.kagu.sut.ac.jp ohmori@ms.kagu.sut.ac.jp

東京理科大学 工学部

コンピュータ作業の過程を記録・再現できるツールがあれば、プログラミング過程の分析、GUI の評価などへの利用が考えられる。我々は、UNIX-WS で標準的に使用されているウィンドウシステムである X-Window 用アプリケーションウィンドウに対するユーザの操作を記録し、後からその操作をアプリケーションウィンドウ上で再現するツール ORRIX (Operation Recorder and Reproducer on unIX) を開発中である。本稿では、その技術的現状と課題について述べる。

A Recording and Reproducing Tool, ORRIX, of User's Operations on WS Environment

Takashi Miyauchi Akira Ohmori

Faculty of Engineering, Science University of Tokyo

If there exists a computerized tool that can record computer operations and reproduce them, we can use it for analysis of programming process, evaluation of GUI, and so on. We are developing a tool, ORRIX (*Operation Recorder and Reproducer on unIX*), which records user's operations of some application windows for X-Window (*X-Window is a standard Window System of UNIX-WS*) and then reproduces them on the application windows. In this manuscript, we report its present technical state and problem.

1. はじめに

コンピュータ作業の操作の過程を記録・再現できるツールがあれば、次のような利用法が考えられる。

- ・プログラミング過程の分析への利用
エディタに対するユーザの操作を記録することで、ユーザごとのプログラミングスタイルの分析などに利用することができる。
- ・インタフェース評価への利用
あるインタフェースに対して、ユーザがどのような操作を行うのかを記録することで、どの機能をよく使うのか、ボタンの配置などは適切であるかといった分析に利用することができる。
- ・定型作業への利用
繰り返し行わなければならない作業に対して、一度作業を記録すれば、次からはその作業を再現するだけでよいので、作業者の負担を軽くすることができ、しかも作業を確実に行うことができる。
- ・デモンストレーションへの利用
アプリケーションソフトウェアのデモンストレーションにおいては、デモンストレーションとしたい操作を記録すれば、その操作の再現をデモンストレーションとして利用することができるので、VTR を使用したり、デモンストレーション用のプログラムを別途作成する必要がない。

ところで、操作履歴からユーザ操作の解析をおこなう研究がいままでにいくつかなされている。鈴木ら[1]は、学生がどのようにエディタ Emacs を使用し、プログラミング演習をしているかを調べている。ここでは、Emacs Lisp が常に最後のキー入力の100文字を保持していることを利用し、1分に一度、キー入力を調べて新しく入力された部分を採用している。しかし1分に一度では、1分間に100文字より多くの入力があった場合、それらは切り捨てられてしまう。またキー入力の順はわかるが、

それが Emacs のどの位置で入力されているかがわかりにくく、マウスの移動した位置や、ドラッグがどこからどこまで行われているかを記録することができないと思われる。また、これは操作の再現を意図していない。一方、西田ら[2]および森ら[3]はウィンドウマネージャを利用してユーザの操作履歴を収集し、操作を再生するツールを試作している。これはユーザが日常の作業を行っている際の操作履歴を取得してユーザの行動分析を行うことを目的としている。操作履歴の記録はウィンドウマネージャが起動された瞬間（ログイン）からはじまり、ウィンドウマネージャの終了（ログアウト）で終わる。ユーザ操作の再生はツールのウィンドウ上で行われ、再生速度の変更や操作履歴のスキップ、バック、操作系列の検索を行うことができ、入力デバイスに関するアナライザも備えている。しかし、ユーザがアプリケーションに対してどのような操作をするかについて分析する場合、ツールのウィンドウ上で操作を再生するよりも実際のアプリケーションウィンドウ上で操作を再現するほうがより有用であると考えられる。

そこで我々は UNIX で標準的に使用されているウィンドウシステムである X-Window 用のアプリケーションウィンドウに対するユーザの操作を記録し、後からその操作を当該アプリケーションウィンドウ上で再現するためのツール ORRIX (Operation Recorder and Reproducer on UNIX) を開発中である。本稿では、その技術的現状と課題について述べる。

2. X-Window システム

2.1 X-Window システムとは[4][5]

X-Window システムは現在 UNIX 上で標準的に使用されている GUI システムである。

X-Window システムでは、各種の処理はクライアントとサーバの2つのプロセスで分担して行われる（図1）。

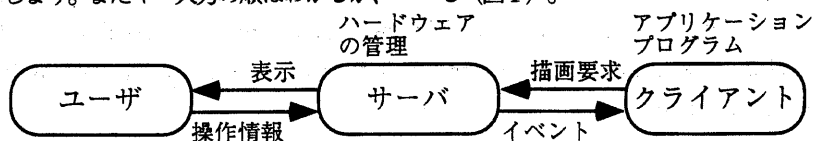


図1 クライアントとサーバの関係

クライアントは計算と処理の主体であり、サーバは表示とハードウェアの管理の主体である。クライアントとサーバの間ではXプロトコル（クライアントとサーバ間のコミュニケーションのための基本メッセージ）として、リクエスト、リプライ、イベント、エラーの4種類の情報が交わされる。各メッセージの転送方向は決まっていて、リクエストは、クライアントからサーバへ、その他は、サーバからクライアントへ転送される。このうちイベントはキー入力やマウス操作などのサーバ側で非同期に発生する情報をクライアントに通知するものである。クライアントはサーバからイベントが送られてくるのを、イベント待ちの無限ループ（イベントループ）に入って待ち続け、そしてイベントを受信して初めて各種の処理を実行する。このような処理をイベント駆動という。X-Windowのソフトウェアはすべてイベント駆動であるので、ユーザの操作を記録するには、イベントを取得して記録すればよいと考えられる。

クライアントはウィンドウを窓口としてイベントを受け取る。クライアントの持つウィンドウをアプリケーションウィンドウと呼び、その中にはウィジェット（メニューやプッシュボタンといったクライアントのインタフェースを構成する部品を示すもの）もウィンドウとして存在する。

ウィンドウは階層構造によって構成され、ウィンドウの上下関係は、親ウィンドウ、子ウィンドウとして表現される。各ウィンドウには、生成される度にサーバによりウィンドウIDという個別の識別番号が付けられる。

2.1 xwininfo コマンド

xwininfo は、ウィンドウに関するさまざまな情報を表示するユーティリティである[6]。コマンドライ

ンから起動し、情報を得たいウィンドウをコマンド行で指定するか、マウスでクリックすると、ウィンドウID、ウィンドウ名、ウィンドウの座標などの情報が出力される。

2.2 InputOnly ウィンドウ[7][8]

ウィンドウのクラスには、InputOutput と InputOnly がある。大多数のウィンドウは InputOutput クラスに属し、ウィンドウに対する入力と表示の出力の両方を行うことができる。一方、InputOnly ウィンドウは目に見えず、背景や境界線を持たない。また、ウィンドウに対しての入力は行いが出力は行わない。この種のウィンドウの目的は、マウスやキーボードからのイベント処理および配送のために画面の各領域を管理分割することである。

3. 操作の記録と再現の方式

3.1 イベントの取得

以下に述べるように、ORRIX はアプリケーションウィンドウが受け取るイベントを横取りすることで、キー入力やマウス操作などのユーザの操作情報を取得する。

ORRIX は記録時、対象アプリケーションウィンドウを指定すると、xwininfo コマンドを起動し、その出力結果から対象アプリケーションウィンドウの最上位のウィンドウのウィンドウIDとウィンドウ名を取得する。そして対象アプリケーションウィンドウの最上位のウィンドウから、階層構造にしたがって、すべてのウィンドウに対して、InputOnly ウィンドウを子ウィンドウとして作成する（図2）。

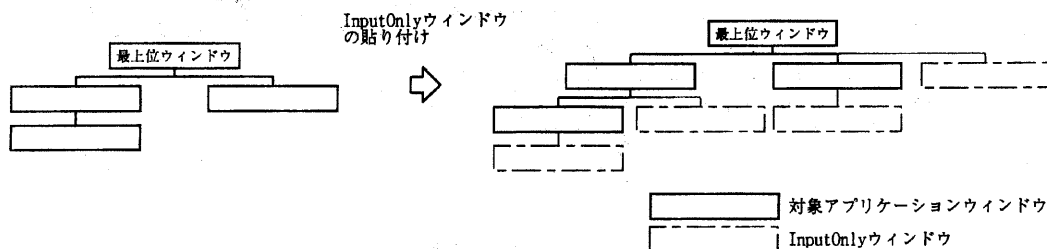


図2 InputOnly ウィンドウの作成

つまり、対象アプリケーションウィンドウの上を透明な InputOnly ウィンドウが覆うことになる。これにより本来対象アプリケーションウィンドウへ送られるはずのイベントは対象アプリケーションウィンドウではなく、InputOnly ウィンドウに送られるようになり、ORRIX がイベントを取得できるようになる（図3）。

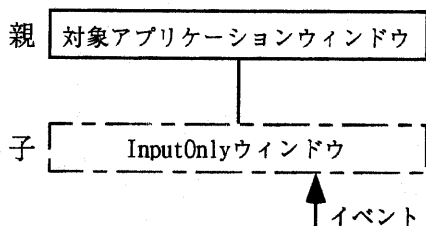


図3 イベントの取得

3.2 操作の記録

InputOnly ウィンドウへ送られたイベントは、ORRIX が取得してファイルに格納する。これによってユーザの操作を記録する。その後、イベントはORRIX によって対象アプリケーションウィンドウへ送られる（図4）。これはイベントが横取りされたため、そのままでは対象アプリケーションウィンドウにイベントが伝達されず、アプリケーションが動作しなくなってしまうためである。この処理は、透明な InputOnly ウィンドウを使用し、ワークステーションはあたかも InputOnly ウィンドウが存在しないかのように動作するため、ユーザからはわからない。よってユーザからは、ユーザがアプリケ

ーションウィンドウを直接操作しているようにみえる。

3.3 操作の記録形式

ORRIX が取得したユーザの操作情報（イベントデータ）はテキストファイル形式でファイルに格納される。よってエディタなどでイベントデータを編集することが可能である。

イベントデータの例を図5に示す。イベントデータは1行単位で格納されており、それぞれは1つのイベントの情報に対応している。各行の最初の数字はイベントの種類を表すものであり、2が KeyPress イベント、3が KeyRelease イベント、4が ButtonPress イベント、5が ButtonRelease イベント、6が MotionNotify イベントを表しており、それにつづく項目はイベントの発生したウィンドウ、座標などを表している。

3.4 操作の再現

操作の再現は、記録時に取得しておいた対象アプリケーションウィンドウのウィンドウ名をもとにして、xwininfo コマンドを起動して自動的に対象アプリケーションウィンドウを特定し、記録時と同様に InputOnly ウィンドウの貼り付けを行う。そして、イベントが格納されたファイルからイベントを読み出して InputOnly ウィンドウの親ウィンドウである対象アプリケーションウィンドウに送信することで、ユーザの操作を再現する。

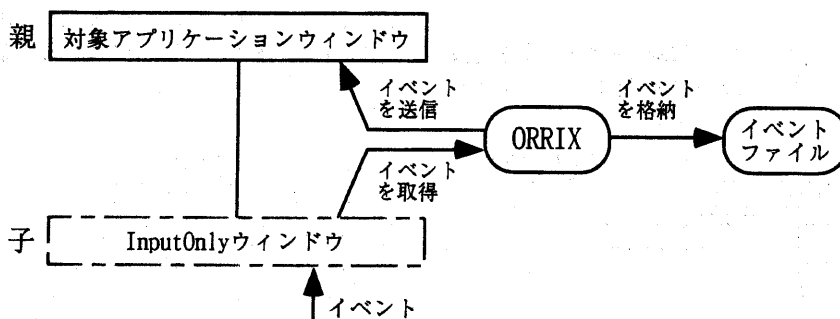


図4 操作の記録

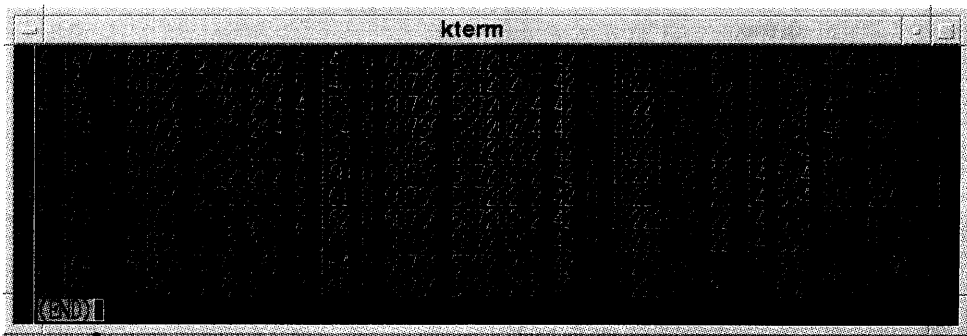


図5 イベントの記録形式

4. ORRIX の現状

4.1 ORRIX の機能的特徴

ORRIX には以下のような機能的特徴がある。

- ・ 指定したアプリケーションウィンドウだけへの操作を記録・再現する。
- ・ 対象アプリケーションウィンドウと独立して機能する。
- ・ 対象アプリケーションウィンドウをマウスで指定する。
- ・ ウィンドウ名の異なる複数のアプリケーションウィンドウに対応する。
- ・ 記録時と再現時で対象アプリケーションウィンドウのウィンドウ ID が異なる場合に対応する。

4.2 記録できる操作

ORRIX が記録できるユーザ操作 (イベント) は、

- ・ キー入力 (KeyPress、KeyRelease イベント)
- ・ マウスボタンの操作 (ButtonPress、ButtonRelease イベント)
- ・ マウスの移動操作 (MotionNotify イベント)

である。それ以外の操作は、現段階では記録・再現することはできない。また、メニューに対する操作も記録・再現することができない。これはメニューを構成しているガジェットがウィンドウ ID を持たないため、InputOnly ウィンドウを子ウィンドウと

して作成することができないためである。InputOnly ウィンドウを利用してユーザの操作を記録・再現する ORRIX にとって、メニューへの対応は現状では非常に困難である。

4.3 対応できるアプリケーション

ORRIX はイベントの送信を XSendEvent によって行っている。よって ORRIX は

- ・ XSendEvent によるイベント受信を許可しているアプリケーション
- ・ デフォルトでは XSendEvent によるイベント受信を禁止しているがオプションでイベント受信を許可できるアプリケーション (kterm など)

に対応している。これまでに動作を確認したのは editres、kterm などである。UNIX で広く使われているエディタ mule に対しての操作の記録・再現は完全にはできなかったが、オプションを -nw として kterm 上で起動させ、さらに kterm のメインオプションのメニューで Allow SendEvents を選択して XSendEvent によるイベント受信を許可することにより、その操作を記録・再現することが可能である。なお ORRIX はルートウィンドウに対する操作を記録・再現することはできない。

4.4 ORRIX の操作方法と動作

ORRIX のインタフェースは GUI (図6) であり、操作をマウスのみで行うことができる。

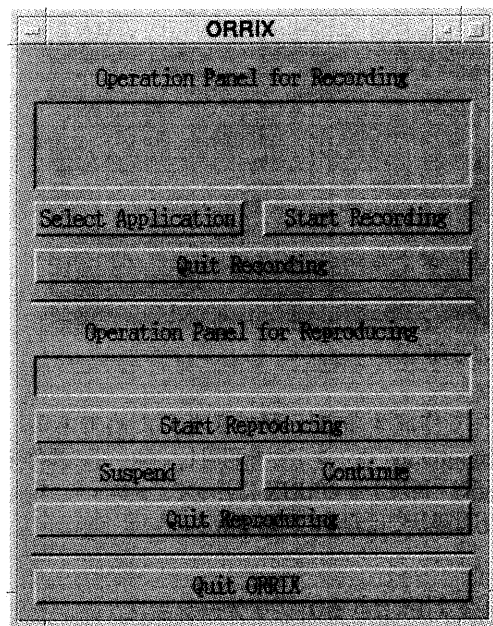


図6 ORRIX のインタフェース

4.4.1 記録時の操作

- (1) **Select Application** ボタンをクリックする。
これにより ORRIX は、`xwininfo` を起動し、アプリケーションウィンドウを選択できるようにする。
- (2) 記録対象として指定するアプリケーションウィンドウをクリックする。
これにより ORRIX は、`xwininfo` で対象アプリケーションウィンドウの最上位のウィンドウのウィンドウ ID とウィンドウ名を取得し、**InputOnly** ウィンドウの作成を行う。
- (3) 他のアプリケーションウィンドウも記録対象として追加するなら(1)と(2)の操作を繰り返す。
- (4) 記録を開始するなら **Start Recording** ボタンをクリックする。
これにより ORRIX は、イベントループに入りイベントが送信されるのを待つ。
- (5) 記録対象アプリケーションウィンドウ上で記録したい操作を行う。
この間、ORRIX はイベントデータをファイル

に書き込む。

- (6) 記録の終了は **Quit Recording** ボタンをクリックする。
これにより ORRIX は、対象アプリケーションウィンドウに貼り付けていた **InputOnly** ウィンドウを破壊する。その後もアプリケーションウィンドウに対する操作は可能である。

4.4.2 再現時の操作

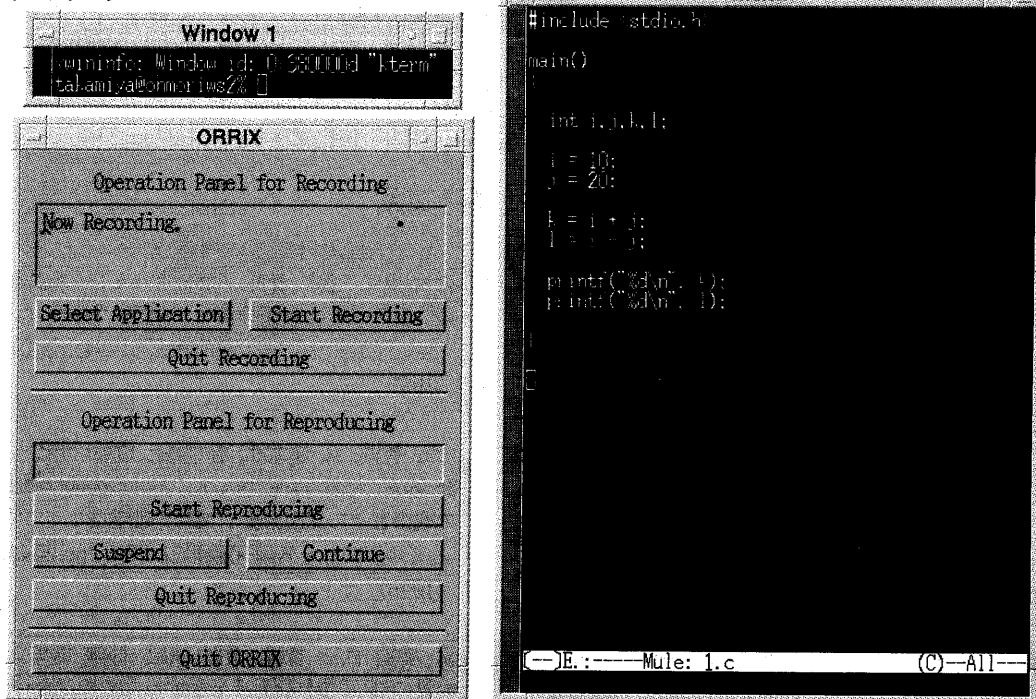
対象アプリケーションが起動されていない場合は、まずそれらを起動してから以下の操作を行う。

- (1) **Start Reproducing** ボタンをクリックする。
これにより ORRIX は、記録時に取得したウィンドウ名をもとに対象アプリケーションウィンドウを自動的に特定し、記録時と同様に **InputOnly** ウィンドウの作成を行う。ファイルからイベントを読み出し、操作を 1 秒間隔で再現する。
- (2) 再現を一時停止するなら **Suspend** ボタンをクリックし、再現の続行は **Continue** ボタンをクリックする。再現の中止は **Quit Reproducing** ボタンをクリックする。
再現中止後はアプリケーションウィンドウに対して通常の操作が可能となる。
- (3) 再現終了のメッセージ「**Reproducing end**」が表示されたら、**Quit Reproducing** ボタンをクリックする。
これにより、アプリケーションウィンドウに対して通常の操作が可能となる。

4.5 ORRIX の動作画面

図7に ORRIX の記録時と再現時の動作画面を示す。対象は `mule` を `kterm` 上で起動したものである。2つは同じアプリケーションウィンドウであるが、記録を取った後に起動しなおしているの、ウィンドウ ID が異なっている。つまり、記録時と再現時で別のウィンドウに同じ内容が描画されている。つまり、操作の記録と再現ができている。

記録時



再現時

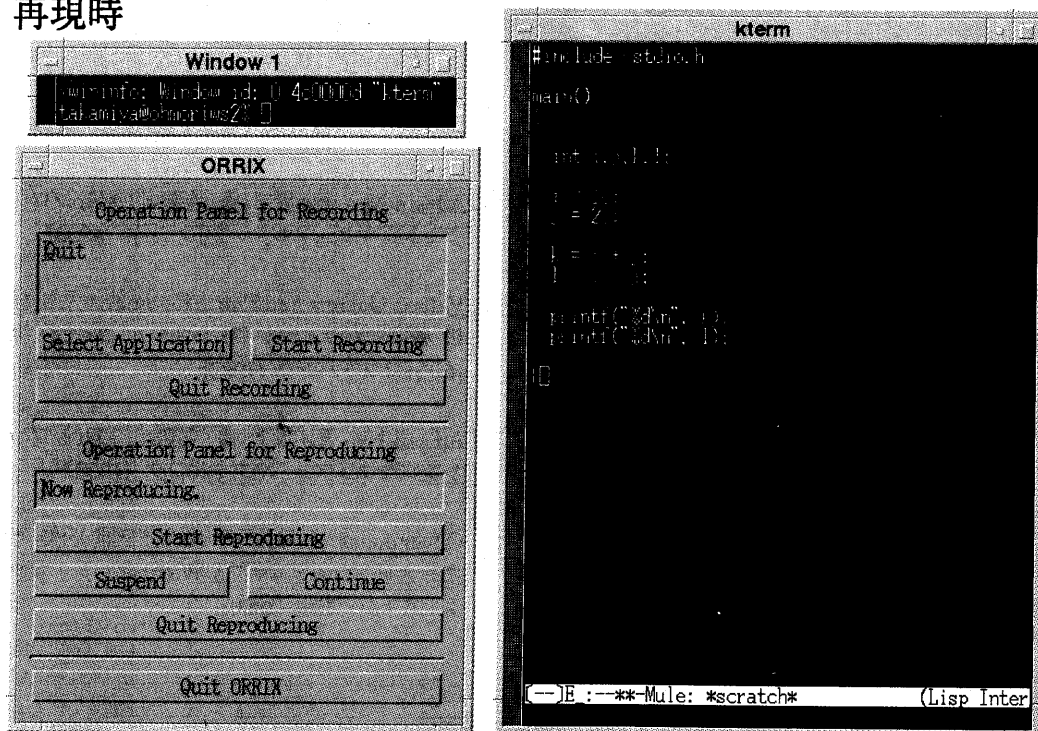


図 7 ORRIXの動作画面

5. まとめ

本稿では X-Window 用アプリケーションウィンドウに対するユーザの操作を記録・再現するツール ORRIX の現状について述べた。

今後取り組むべき課題には、以下のような事項がある。

・再現する操作の指定

記録した操作を再現するときに、ある特定の操作だけを指定して再現できるように改良していきたい。

・再現時間の調整

ORRIX は操作の再現を 1 秒間隔で行っている。これだと操作をじっくり観察するにはよいが、操作すべてをざっと流して見るには時間がかかり過ぎてしまう。よって操作の再現を実時間に近い状況で行えるように改良していきたい。また、再現時間（再現の間隔）を変更できるように改良していきたい。

・記録・再現できるイベントの追加

ORRIX で記録・再現可能な操作（イベント）は現在のところキー操作（KeyPress, KeyRelease）、マウスのボタン操作（ButtonPress, ButtonRelease）、マウスの移動操作（MotionNotify）である。しかしユーザの操作によって発生するイベントは他にも存在するので、それらのイベントを記録・再現するように改良をしていきたい。

・記録開始後の記録対象の追加

ORRIX は対象アプリケーションウィンドウを指定してからユーザの操作を記録しているが、記録開始後に新たに記録対象ができた場合にも、それを追加し、操作を記録・再現できるようにしていきたい。

・同名のアプリケーションウィンドウの識別

再現時、ORRIX はウィンドウ名をもとにして対象アプリケーションウィンドウを特定しているた

め、同じ名の複数のアプリケーションウィンドウに対する操作を記録して、再現する場合に 1 つのアプリケーションウィンドウ上にすべての操作が再現されてしまう。これに対する解決方法については今後検討していきたい。

参考文献

- [1] 鈴木 悦子, 小川 貴英: "プログラミング操作記録による学生の意図分析", 情報処理学会ヒューマンインタフェース研究会報告 60-1, pp.1-8 (1995)
- [2] 西田 智博, 森 慎太郎, 森 孝弘, 斎藤 明紀, 辻野 嘉宏, 都倉 信樹: "GUI における実操作履歴の取得とその意図分析", 情報処理学会ヒューマンインタフェース研究会報告 60-4, pp.25-32 (1995)
- [3] 森 孝弘, 西田 智博, 斎藤 明紀, 都倉 信樹: "大量の GUI 操作履歴を分析するための走査・再生ツール", 情報処理学会ヒューマンインタフェース研究会報告 69-1, pp.1-8 (1996)
- [4] 木下 凌一/林 秀幸 著: X-Window Ver.11 プログラミング[第2版], 日刊工業新聞社 (1993)
- [5] Adrian Nye 著/坂下 秀 他監訳: Xlib プログラミングマニュアル, ソフトバンク (1993)
- [6] Valerie Quercia and Tim O'Reilly 著/大木 敦雄 監訳: X-ウィンドウ・システム ユーザ・ガイド, ソフトバンク (1993)
- [7] Adrian Nye 著/坂下 秀 他監訳: Xlib リファレンスマニュアル, ソフトバンク (1993)
- [8] Oliver Jones 著/西村 亨 監修/三浦 明美・ドキュメントシステム訳: X-Window ハンドブック, アスキー出版 (1990)