インタラクティブアニメーションの分析・再利用のためのツール

大島 裕樹[†1]　宮下 芳明[†1†2]

本稿ではソフトウェア上で用いられているインタラクティブアニメーションを分析し，再利用するためのツールを提案する．多くのソフトウェアではユーザの操作に対するアニメーションが用意されており，優れた挙動はユーザに心地よい操作感をもたらしている．提案システムでは既存のアニメーションをキャプチャし，フレーム毎に挙動を分析することでインタラクティブアニメーションの記録を行う．記録したアニメーションはユーザのプログラムに組み込むことができ，心地良いと感じた挙動を再利用することができる．アニメーションの記録はゲームや映画といったコンテンツにも行うことができ，既存キャラクターの挙動を再現したゲーム制作や，コンテンツ内の架空インタフェースの再現なども可能になる．

A Tool for Analysis and Reuse of Interactive Animation

YUKI OSHIMA [†1]　HOMEI MIYASHITA[†1†2]

We propose a tool for analyzing and reusing interactive animations that have been used on the software. In many software, there are several animations for operation, and high quality animations give us comfortable feelings for operations. Our proposed system, records animation by capturing the animation and analyzing the operation in each frame. The recorded animations can be set in the user's program, and the operation that we feel comfortable can be used again. They can be used for content such as games and movies, and therefore we can produce the game where the operation of existing characters is reproduced as well as fictional interface can be reproduced.

1. はじめに

コンピュータ上での操作にはアニメーションが付き物である．マウスカーソルの挙動においても，ドキュメントや Web ページでスムーズにスクロールして見えるのも，アニメーションによるものである．また，ゲームコンテンツを始め，キー入力に応じて画面内でオブジェクトがアニメーションするコンテンツも数多く存在する．GUI による操作が大半を占める現在のコンピュータ環境では，このようなアニメーション表現はいたるところに存在する．本稿では，そういったユーザの操作に対応して実行されるアニメーションを「インタラクティブアニメーション」と呼ぶ．

本稿で提案するのは，インタラクティブアニメーションを記録して分析し，自身のプログラムに再利用するためのツールである．インタラクティブアニメーションは，ユーザの操作感に大きな影響を与える．操作対象を見失わないようにオブジェクトの軌跡を表示したり，実行されている内容が理解しやすいように動作で示したりと，心地よい操作感を提供するためにインタラクティブアニメーションは欠かせない．しかし多くのユーザが心地よいと感じるアニメーションを一から作り出すのは難しい. 提案システムは，既存の優れたインタラクティブアニメーションを分析することで，プログラムにおけるアニメーション実装を促進する．GUI の操作やゲームコンテンツからインタラクティブアニメーションを記録・分析することで，自作プログラムへの再利用を実現する(図 1).

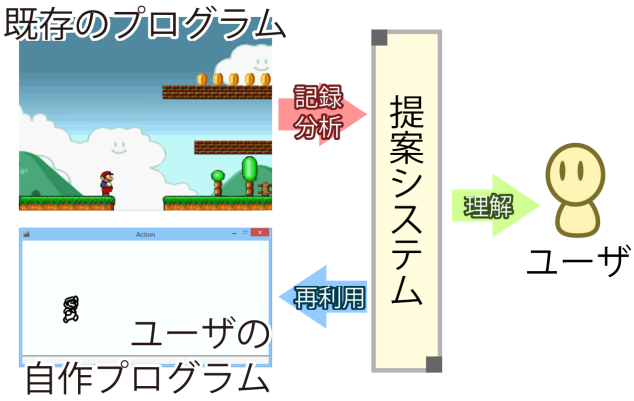

図 1　提案システム概要

Fig. 1　Overview of the proposed system

1.1 分析支援の必要性

インタラクティブアニメーションを伴う GUI が広く使用される現状に対して，より良い GUI を構築するためにはユーザの操作についての理解が重要である[1][2].そして理解を深めるためにはまず既存の環境に触れる必要がある．GUI にアニメーション技法を取り入れる思想は古くから存在しており，アニメーションの導入により動作に関する情報を簡潔に伝えることができるとされている[3]．インタラ

†1 明治大学大学院理工学研究科新領域創造専攻ディジタルコンテンツ系
Program in Digital Contents Studies, Program in Frontier Science and Innovation, Graduate School of Science and Technology, Meiji University
†2 独立行政法人科学技術振興機構, CREST
JST, CREST

クティブアニメーションの進化の過程で，どの操作にどのアニメーションを割り当てるべきか，改善が行われてきた[4]．こうして過去から評価され，現在まで参考にされてきたインタラクティブアニメーションには，ユーザに心地よい操作感をもたらす要素が含まれている．しかし，それらのアニメーションは多くのユーザにとって「完成されたプログラム」として見る機会しか無く，どのようなアルゴリズムで動作しているかは想像するしかなかった．提案システムが目指すのは，そのようなブラックボックス化したインタラクティブアニメーションを，プログラマが再利用できるように記録して分析することである．

本稿ではコンピュータの操作に加えて，ゲームコンテンツをインタラクティブアニメーションの分析対象として議論する．操作に反応して画面内のキャラクターなどが動くという点で，ゲームコンテンツはインタラクティブアニメーションが多量に用いられたコンテンツと言える．ゲームコンテンツを制作する場合にも，既存作品の動きを模倣することは珍しくないが，直にソースコードを見る機会はほとんどない．そのためGUIと同様にゲームコンテンツの分析も価値のあることである．

ゲームコンテンツにおけるキャラクターの挙動はコンテンツの質に直結する．移動速度やジャンプの挙動，キー入力に対する反応速度などを適切に調整しなければ，操作していて心地のよいゲームコンテンツは生まれない．Bret Victorによるコンセプトビデオ[5]においては，プログラムのパラメータに応じてゲーム内のキャラクター挙動を推測する手法が示されており，ゲームコンテンツにおけるパラメータ調整の重要さと，既存環境での調整の難しさを物語っている．ストレスを感じない移動速度はどの程度か，適切なジャンプ力はどれくらいか，これらの議論は過去のゲーム作品の中でも行われてきたはずのことである．そして，多くの人に親しまれている過去の名作におけるキャラクターの挙動は，それらの議論に対する一つの解答であり，困難な調整の末に完成されたパラメータとして大きな価値がある．

過去の優れたGUIやゲームコンテンツに対して，インタラクティブアニメーションを分析することは，より良いコンテンツを生み出すために必要不可欠である．本稿の提案システムは，アニメーションの分析に対して有用に働くと考えている．

1.2 再利用支援の必要性

プログラムを分析することによって，インタラクティブアニメーションの実装に必要なパラメータを推測する作業が省略され，試行錯誤の手間を軽減できる．しかし，これでは分析結果をユーザ自信がプログラムに流し込んでいるだけであり，ソースコードを記述する手間自体は軽減できていない．記録したインタラクティブアニメーションは，それ自体が一つのモジュールである．プログラムをユーザの入力と，それに対するアニメーションの集まりと考えれば，インタラクティブアニメーションをモジュールとして再利用することで，実装の手間を削減することができる．

提案システムは分析と合わせて，過去のインタラクティブアニメーションを記録し，蓄積する機能を持つ．元々プログラミングで過去のソースコードを再利用する場面は多い．本稿での提案はこれまでソースコード単位で行われていた再利用を，インタラクティブアニメーションという括りで行うものである．提案システムによって手間が削減される他，プログラミング初学者の支援にもつながると考えられ，分析と再利用による新しいプログラミングの手順をも実現する．

2. 関連研究

プログラムの分析を試みた研究として，AndrewらのWhylineがある[6].これはプログラミングにおけるデバッグ作業の効率化を目的としたシステムで，実行時の動作に対して質問を選ぶことで，ソースコードの該当箇所を示すものである．本稿で提案する手法は，ソースコードにアクセスできないプログラムに対しての使用を想定しているが，実行時の動作を元に分析するという思想は一致している．

GUI改善のための分析ツールとしては，森らの研究が挙げられる[7]．この研究では操作分析のために大量の履歴を記録し，再生する機能を備えている．操作内容は経過時間を横軸としたグラフ上に記録をしており，時間軸に対応して操作内容記録は本稿の提案システムにも用いられている．

プログラミング支援の観点から，外部環境の動作を利用した開発環境の提案が多数存在する．カメラから動作を取り込み，実装に活用した研究としてKatoらのVisionSketchが挙げられる[8]．これは動画の指定領域の変化を記録し，画像処理を行うシステムであり，処理の結果をプログラムに組み込むことで簡単に動画や画像をベースにしたプログラムを実装できるものである．本稿の提案システムでは，GUIやゲームコンテンツを記録の対象にしているが，対象物の特定領域に対して記録を行うという点で共通している．また機能の一つに，指定領域内の様子を一列に並べてタイムラインのように表現するものがあり，領域内の変化を時間軸に対応させて視覚化する手法は提案システムでも取り入れているものである．

同様にカメラからの入力を扱った開発環境としてPicodeがある[9]．Picodeではロボットの姿勢制御のように，ソースコードから実行内容を全く連想できない事態に対して，写真を組み込むことで情報の視覚化を試みている．

FukahoriらのCapStudioでは，実行画面のプレビューを用意し，リアルタイムにパラメータを調整する開発環境を実現している[10]．本稿の提案システムは過去のインタラクティブアニメーションを，そのまま記録して再利用するものであるが，記録元のパラメータを直接利用することが最

適とは限らない．パラメータを変化させた際の挙動を確認する上で，CapStudioで取り入れられた対話可能なプレビューの導入は有効であり，提案システムにも機能のひとつとして取り入れている．

JeremyらのPhysInkは，ゲームデザイナーなどにとって日常的に行われている物理的な動作の記述を支援するツールである[11]．PhysInkでは物体の動作はタイムライン上で管理されており，特定の時間に移動した後に軌跡を書き足すことでそれ移行の挙動を変更でき，時間を指定して後の挙動をシミュレーションする思想は提案システムでも取り入れている．提案システムではフレーム毎の内容を，入力状況と合わせてタイムライン上に配置している．提案システムではフレームに対して入力を設定することで，挙動のシミュレーションを行うことを実現する．

3. 提案システム

提案するシステムは，インタラクティブアニメーションを分析して，システム内に記録する部分と，アニメーションをユーザが作成したプログラムへ組み込む再利用部分に分かれる．ここでは，それぞれの機能についての詳細を述べる．

3.1 分析する領域の指定

インタラクティブアニメーションの分析は，ディスプレイ内の指定領域に対して行う．領域の指定には，領域指定フレームを使用する．フレームは一般的なウインドウと同じように移動とサイズ変更が可能となっており，ユーザはフレームを分析したい場所に移動し，必要に応じて大きさを調整することができる．また，指定した領域はプレビューとしてシステム上に表示される(図 2).

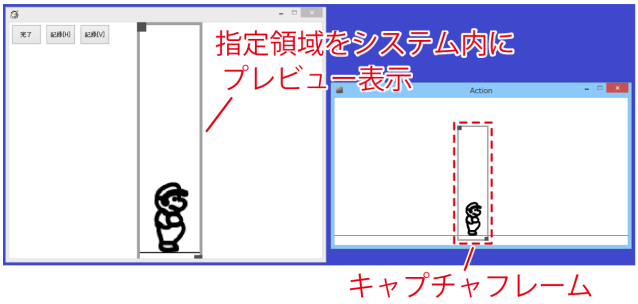

図 2　キャプチャフレームとプレビュー表示

Fig. 2　Capture frame and preview display

3.2 分析範囲の一列配置記録

アニメーションを分析するためには，実際の挙動を観察する必要があるが，正確に把握するためには連続して動いている様子を見るだけでは不十分である．提案システムでは，指定領域内でキャプチャした画像を，一列に並べることで動作を記録する．上下に移動するアニメーションに対しては横に並べ，左右に移動するアニメーションに対しては縦に並べることで，アニメーションをフレーム毎の軌跡として記録することができる(図 3).

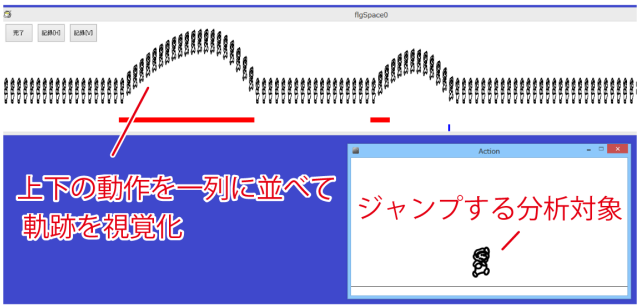

図 3　一列に配置してのアニメーション記録

Fig. 3　Animation record in one array

一列に並べることによって，ユーザは対象のフレーム間での挙動をより正確に知ることができる．等速で移動しているのか，フレーム毎に加速度はどれくらいかなどを視覚的に確認することができる．画像としての記録の他に，フレーム間の画像差分から対象が移動した方向と距離，加速度を数値として記録しており，アニメーションを自作プログラムに再利用する際にはこの数値を用いて近似した関数を生成する．

同系統のアニメーションの差を検証する際には，静止画による比較が有効である(図 4)．GUI であれば，OS の違いやソフトウェアの違いによる微妙な差を検証することができ，ゲームコンテンツであれば運動性能の比較や，アクションに要する時間の差などを検証することができる．

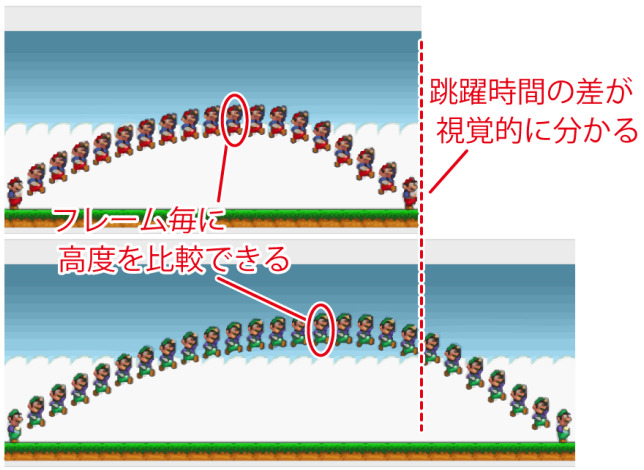

図 4　静止画による比較が有効なケース

Fig. 4　A valid case comparing to the still image

3.3 一方向に限定されないアニメーションの記録

ゲームコンテンツにおけるダッシュジャンプのように，縦方向と横方向のアニメーションが組み合わさった動作に対しては，フレーム毎に記録する位置を調整することで対応をする．ユーザはアニメーションする範囲を予め予想し

てキャプチャフレームの範囲を決定する．全体キャプチャ範囲の他に，分析対象に合わせて 1 フレーム辺りの記録サイズを指定することができ，キャプチャ時には 1 フレーム毎に隣り合う領域を記録して，軌跡を作成することができる．(図 5).

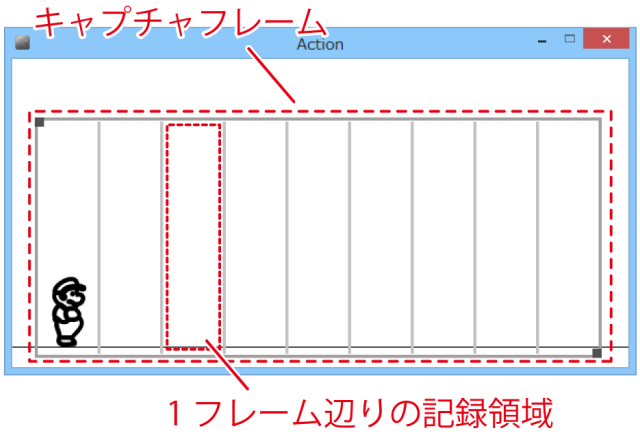

図 5　動作方向が限定できない場合のキャプチャフレーム
Fig. 5　Capture frame in the case of unidentifiable directions

ダッシュジャンプの例を考えると，跳躍開始地点と着地地点を含むようにキャプチャフレームを配置すれば良い．この場合横方向の移動は等速とは限らず，フレーム毎に多少記録位置がずれるが，これはシステム側で調整することができる．ユーザの手間は増えるが，複雑に速度を変化させるアニメーションに対しても，一方向の成分のみを抽出した記録を行える．

3.4　入力状況の記録

インタラクティブアニメーションは，ユーザの操作に対して行われるアニメーションである．挙動を再現するためには入力と動作の時間関係が重要であり，分析を行う上でユーザの入力状況の把握は必要不可欠である．提案システムでは，アニメーションの記録中に特定キーの入力を監視し，入力があったフレームが分かるようにキャプチャ画像と並べて表示する(図 6). ユーザは分析に必要なキーを指定することで，視覚的に入力状況とアニメーションの関係性を容易に確認することができる．

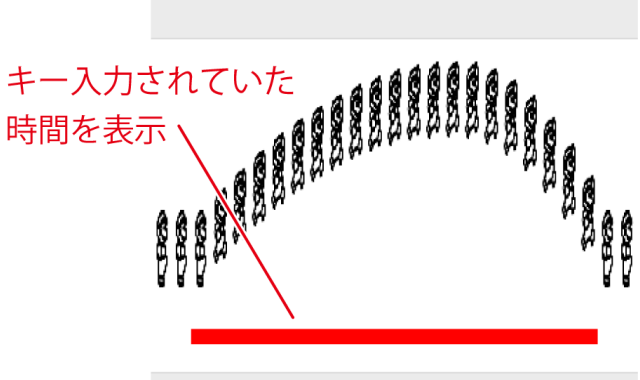

図 6　アニメーションに対する入力状況の表示
Fig. 6　Input status display for the animation

3.5　アニメーションの確認及びシミュレーション

記録したインタラクティブアニメーションは，システム上で自由に再生することができる．この機能は連続した静止画ではなく，実際の動作を確認したい場合に有効である．さらに，記録した様子をそのまま再生するだけではなく，入力状況をユーザが指定してのシミュレーションを行うこともできる．例えば多くのアクションゲームの場合，ジャンプボタンを押す長さで跳躍の高さが変化する．予めボタン入力の有無によるジャンプ高度の差を分析しておけば，フレーム単位で入力状況を指定することで，挙動の違いをシミュレーションで確認することができる．

3.6　分析内容を用いたプログラム制作

提案システムでは，アニメーション制作に特化した開発環境を用意しており，分析したインタラクティブアニメーションを使ってのプロトタイププログラミングが可能になっている．操作対象となる最低限のオブジェクトはシステム側で用意してあり，ユーザはオブジェクトに対してどのインタラクティブアニメーションを実行させるかを選ぶだけで簡単なプログラムを実装できる．

提案システムが用意した環境では実装できないプログラムのために，外部環境で分析したアニメーションを再利用する手段も用意してある．記録されたパラメータを，外部環境に合わせた関数として提示することで，他の開発環境でも分析したインタラクティブアニメーションに近い挙動のものを作成することができる．

4.　ツール使用例

ここでは，提案システムが有効活用できるインタラクティブアニメーションについて紹介する．

4.1　一部ソフトウェアのスムーズなスクロールの分析

Word2013 や Excel2013 など，一部のソフトウェアではページアップ・ダウンキーを押した際にスムーズなスクロールのアニメーションが実行される．このアニメーションに対して，加速具合や最高速の分析を行うことができる．

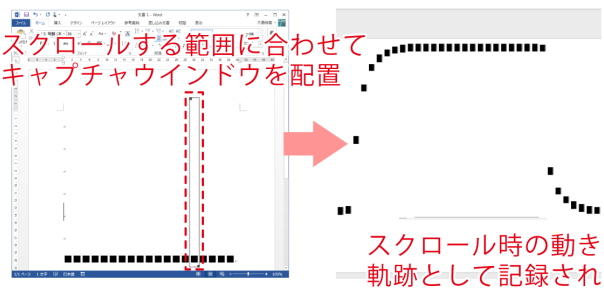

図 7　縦方向スクロールの記録
Fig.7　Record of the vertical scrolling

分析を行う際は，記録後にユーザが閲覧しやすいようなファイルを用意すれば良い．例えば，Word を対象にアニメーションの記録を行うならば，区切り線だけを描画しただけのファイルに対してキャプチャ範囲を指定すれば良い（図 7）．この状態でページアップ・ダウンキーを入力することで，アニメーション時の挙動が視覚化できる．

4.2 アクションゲームにおける左右移動の分析

アクションゲームにおける左右移動の挙動は，作品毎に差がある．特に分かりやすいのがキャラクターの移動速度が等速の場合と，徐々に加速しながら移動する場合の比較である．ここでは両者の違いを提案システムで比較する手順を考える．

左右移動の分析なので，キャプチャウインドウは横長に配置して記録を開始する．この時方向キーの入力状況を合わせて記録することで，入力を開始した瞬間からの挙動が比較できる(図 8)．

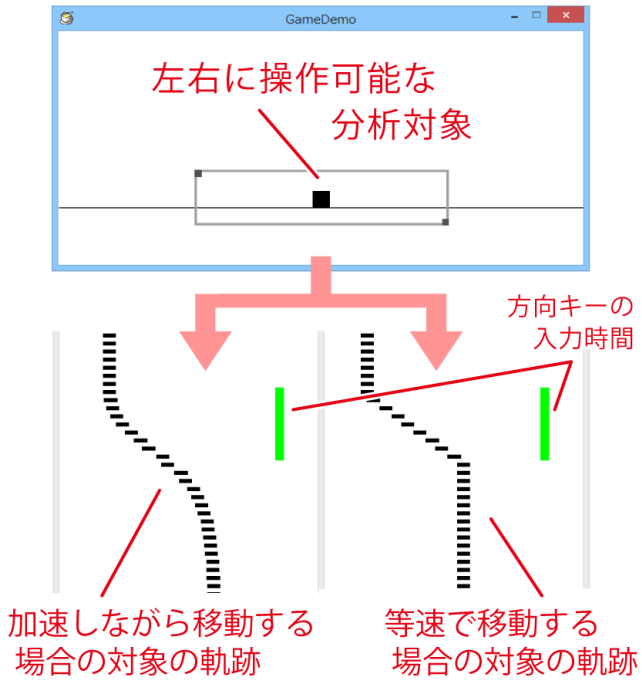

図 8　左右移動の分析と比較

Fig. 8　Analysis and comparison of right-left movement

加速しながら移動する場合と，等速で移動する場合では，入力時間が同じでも挙動に差が生じることが読み取れる．加速しながらの移動では短い入力で細かな座標の調整ができ，繊細なアクションを要求するゲームではこちらの手法が向いていると分析できる．一方等速で移動する場合は入力時間と移動距離は比例関係にあり，入力と同時に最高速度に達することから，ストレスを感じさせないスムーズな動作が実現できると分析できる．

4.3 空中ジャンプの分析

多くのゲームコンテンツに取り入れられているジャンプアクションを拡張するものとして，空中ジャンプがある．これはボタンを押すタイミングによって，跳躍の高度や距離を伸ばしたり，着地までの時間を稼いだりと奥の深い要素である．これに対し，シミュレーションを交えて空中ジャンプを最も活用できる入力のタイミングについて分析を行う．

最初にジャンプ入力時の挙動と，空中ジャンプ入力時の挙動をシステムでそれぞれ記録する．ここから空中ジャンプでの最大跳躍高度を調べたい場合，ジャンプの頂点で空中ジャンプの入力を行うようにシミュレーションすれば良い．一方，最大跳躍距離を調べたい場合は，複数のタイミングで入力した空中ジャンプのシミュレーションし，着地までに要したフレーム数を比較すればよい．

コンテンツ内のアクションの限界を分析することで，アクションに対する地形の設定などを読み取ることができ，どこまでシビアな入力をプレイヤーに要求しているかなど，考察の助けになる．

5. 課題と展望

提案したシステムは，フレーム毎に分析対象を並べることで挙動を視覚的に分かりやすく記録し，他の挙動との比較にも優れた分析形態を採用した．この手法は画像を並べる都合上，上下と左右の同時に動作するアニメーションに対しては最適な手段とは言い難い．二方向の組み合わさったアニメーションに対しては，広くキャプチャ範囲を指定することによる記録法を提案したが，複雑な動きになるほど，記録時のユーザの調整作業が増えてしまい，効率化のためにも改善の必要がある．

また，入力状況の記録についても改善が必要である．今回はキーボードからの入力を記録することで，ジャンプ時のキー入力時間や，ダッシュボタン入力の有無を把握することができたが，これだけでは GUI 操作に対する分析に対応できないケースが多い．これは多くの GUI に関するアニメーションではマウスカーソルによる操作が関わっており，単純に入力状況を把握するだけでは不十分であるためである．GUI におけるインタラクティブアニメーションを分析する場合，マウスの座標や特定の点からの距離，マウスホイールの値など，分析に役立つ情報を記録する手段が必要とされる．視覚的に分析しやすくするという提案システムの目的に対して，これらの数値情報をユーザが理解しやすい形に視覚化することが今後の課題となる．展望としてはここまでに挙げた問題点の改善により，提案システムを幅広いインタラクティブアニメーションに対して，有用な分析ツールへと発展させていきたい．

6. おわりに

本稿で提案したシステムは，二次元空間におけるインタラクティブアニメーションを分析するものであるが，GUIにおけるアニメーションを分析する思想は今後も必要とされていくと考える．コンピュータが一般のユーザに普及して以来，その性能は大きく向上してきたが，GUI に関しては根本的には共通する思想があるのではないだろうか．二次元空間に配置されたウインドウやアイコンに対して操作するという慣れ親しまれた環境が継続する限り，過去のインタラクティブアニメーションを活用する機会は必ず存在し，加えて近年増加しているタッチ操作の影響もあり，心地よい操作感の重要性は増していくと考える．

ゲームコンテンツに対しては，GUI とは異なり直接最先端のコンテンツへの再利用が難しい．今やゲームコンテンツは三次元空間上で操作されるのが当たり前であり，一見すると過去の名作の動作を分析する価値は薄くも見える．しかし，動作を直接利用することは難しくとも，過去の作品に含まれる動作の分析には意味がある．心地よいジャンプや快適な移動速度, 入力に対する適切な反応速度などは，環境が変わっても重要な要素であり，分析からそれらを理解することで，最先端での実装にも活かすことができる．

より良いインタラクティブアニメーションを生み出すためには，過去に評価されてきた動作を分析することが重要である．その分析に提案システムは貢献することができると考える．

参考文献

1) Sarah A. Douglas, Anant Kartik Mithal. The effect of reducing homing time on the speed of a finger-controlled isometric pointing device. CHI '94 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp.411-416, 1994.
2) I. Scott MacKenzie, William Buxton. Extending Fitts' law to two-dimensional tasks. CHI '92 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp. 219-226, 1992.
3) Bay-Wei Chang, David Ungar. Animation: from cartoons to the user Interface. UIST '93 Proceedings of the 6th annual ACM symposium on User interface software and technology, pp.45-55, 1993.
4) Patrick Baudisch, Desney Tan, Maxime Collomb, Dan Robbins, Ken Hinckley, Maneesh Agrawala, Shengdong Zhao, Gonzalo Ramos. Phosphor: explaining transitions in the user interface using afterglow effects. UIST '06 Proceedings of the 19th annual ACM symposium on User interface software and technology, pp.169-178, 2006.
5) Bret Victor. Inventing on Principle. http://vimeo.com/36579366.
6) Andrew J. Ko, Brad A. Myers. Debugging reinvented: asking and answering why and why not questions about program behavior. ICSE '08 Proceedings of the 30th international conference on Software engineering, pp.301-310, 2008.
7) 森孝 弘, 西田 知博, 斉藤 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. 情報処理学会研究報告. HI, ヒューマンインタフェース研究会報告 vol.96, pp.1-8, 1996.
8) Jun Kato, Takeo Igarashi. VisionSketch: integrated support for example-centric programming of image processing applications. GI '14 Proceedings of the 2014 Graphics Interface Conference, pp.115-122, 2014.
9) Jun Kato, Daisuke Sakamoto, Takeo Igarashi. Picode: inline photos representing posture data in source code. CHI '13 Proceedings of the SIGCHI Conference on Human Factors in Computing Systems, pp.3097-3100, 2013.
10) Koumei Fukahori, Daisuke Sakamoto, Jun Kato, Takeo Igarashi. CapStudio: an interactive screencast for visual application development. CHI '14 Extended Abstracts on Human Factors in Computing Systems, pp.1453-1458, 2014.
11) Jeremy Scott, Randall Davis. Physink: sketching physical behavior. UIST '13 Proceedings of the 26th annual ACM symposium on User interface software and technology, pp.9-10, 2013.