

装着型センサを用いた 体感型ゲーム制作支援フレームワーク

田 中 宏 平^{†1,*1} 岸 野 泰 恵^{†2} 寺 田 努^{†3,†4}
宮 前 雅 一^{†5} 西 尾 章 治 郎^{†1}

実世界の人の動きを用いてゲームを楽しむ体感型ゲームが注目を集めている。このようなゲームでは、一般的にセンサを用いて人の動作を取得するが、ゲーム制作者はセンサ接続やセンサデータの扱いに関する知識が必要となり製作の敷居が高い。そこで本研究では、体感型ゲーム製作を支援するフレームワークを提案する。本研究では、ゲーム製作に広く用いられている Adobe 社の Flash に着目し、コンテンツ作成者がゲームの入出力として利用したいユーザの動作を選択するだけでゲームの枠組みが作成できるフレームワークを実現する。提案フレームワークのプロトタイプを用いて複数のゲームコンテンツを作成し、体感型コンテンツが容易に作成できることを確認した。

A Framework to Construct Video Games Using Wearable Sensors

KOHEI TANAKA,^{†1,*1} YASUE KISHINO,^{†2}
TSUTOMU TERADA,^{†3,†4} MASAKAZU MIYAMAE^{†5}
and SHOJIRO NISHIO^{†1}

Recently, the games that can be dealt intuitively by motions attract a great deal of attention. In such games, since user motions are captured by wearable sensors, game creators must have the knowledge about sensor hardware and data processing methods of sensors. In this paper, we propose a framework that helps to create contents controlled by motions. Our framework assumes that the contents are created by using Adobe Flash that is popular for creating multimedia contents. Using our tool that supports complex transactions of interactive contents creation, the things that the creator does are the selection of the mappings between contexts and gestures. We confirmed the effectiveness of the proposed framework by creating actual contents.

1. はじめに

近年、ゲーム業界において従来のゲームパッドやジョイスティックなどに加えて、センサ技術を用いた直観的な新しいコントローラが注目を集めている。特に、2006 年冬に任天堂から発売された Wii は、加速度センサ、CMOS センサを備えたコントローラをラケットや剣に見立ててゲームを楽しむことを可能とし、新たなゲーム市場の獲得に成功している。このような直観的なインタフェースは、複雑化するゲーム操作の敷居を下げるだけでなく、実際に行動することによる運動不足の解消など、新たなゲーム利用の側面を形成している。しかし、このようなセンサ技術を用いたゲームを制作するためには、センサの接続に関する知識やセンサデータの処理方法に関する知識が必要となり、従来のゲーム制作者にとってその敷居は高い。また、1 度作成したゲームコンテンツに対して、利用するセンサの変更や新たなセンサの追加を行うことは、コンテンツのプログラムを書き換える必要があるなど、手間を要する。

一方、近年の Web の普及にともない、Web ブラウザ上でアニメーションやゲームを楽しむためのプラットフォームである Adobe Flash¹⁾ が普及している。Flash は、Flash コンテンツを再生するための Flash Player が幅広いプラットフォームに対応しており、内容に対してのファイルサイズが小さいため、インターネットとの親和性が高い。2008 年 6 月の時点で、アメリカ、日本などでインターネットに接続できるパソコンの 99% 以上で利用されており、ほとんどのユーザが Flash コンテンツを再生できる環境にある。また、Flash コンテンツは、ActionScript と呼ばれるスクリプト言語を利用して、ユーザの入力を受け付けたり、画像を操作したりするなどのプログラミングを容易に行えることから、Flash コンテンツの開発環境を用いて簡単なミニゲームなどの制作を行うクリエイタが増加している。

†1 大阪大学大学院情報科学研究科

Graduate School of Information Science and Technology, Osaka University

†2 NTT コミュニケーション科学基礎研究所

NTT Communication Science Laboratories

†3 神戸大学大学院工学研究科

Graduate School of Engineering, Kobe University

†4 科学技術振興機構さきがけ

PRESTO, Japan Science and Technology Agency

†5 ウェストユニティス株式会社

WESTUNITIS Co., LTD.

*1 現在、三菱電機先端技術総合研究所

Presently with Advanced Technology R&D Center, Mitsubishi Electric Corporation

```

WHEN GPS_MOVE
IF CURRENT.Pos == 'station'
THEN DO BROWSER_OPEN('URL')

```

図 1 ECA ルールの例

Fig. 1 An example of ECA rule.

そこで本研究では、Flash を用いた体感型ゲームコンテンツ制作のニーズがあると考え、センサの扱いに関する知識が乏しいゲーム制作者が、センサやアクチュエータを用いた体感型ミニゲームを作成することを支援するフレームワークを提案する。提案フレームワークで実現した体感型コンテンツ作成支援ツールを用い、ゲームコンテンツ作成者が入力として用いるプレイヤーの動作を設定するだけで、体感型ゲーム制作に必要な ActionScript の出力や、センサデータの利用に必要な設定を行え、センサの知識がなくても体感型コンテンツを制作できる。

以下、2 章で本研究の想定環境を定義し、3 章で関連研究を述べる。4 章で提案フレームワークの詳細について説明し、5 章で提案機構を用いたゲームコンテンツ制作について説明する。6 章で考察を行い、最後に 7 章でまとめる。

2. 想定環境

本研究では、センサの接続に関する知識やセンサデータの処理方法に関する知識は十分でないが、Flash コンテンツは制作できる知識があるユーザを想定する。近年、Flash を用いた Web コンテンツが数多く作成されており、このようなユーザは多数存在する。

また本研究では、プレイヤーの動作を取得・定義するために Wearable Toolkit⁶⁾ を用いる。Wearable Toolkit は、イベント駆動型のルール処理エンジンとその関連ツールで構成される状況依存システム作成支援キットである。ルール処理エンジンで処理されるルールは、発生する事象 (Event)、実行条件 (Condition)、実行する動作 (Action) の 3 つを一組とした ECA ルールにより記述する。たとえば、「ユーザが駅に到着したときに、時刻表を表示する」システムを作成する場合には、図 1 に示すように、GPS 電波を受信したときに (WHEN) その値が駅付近であれば (IF)、時刻表の Web ページを表示する (THEN DO) という 1 つのルールで実現できる。さらに、Wearable Toolkit には図 2 に示すような、ユーザの動作や状況を定義するコンテキスト定義ツールが付属している。このツールは、ユーザの動作や状況を認識し、イベントとして簡単に定義するためのツールである。ユーザはツールを用



図 2 コンテキスト定義ツール

Fig. 2 A snapshot of context tool.

いて定義したい動作を実際に行い、質問に数回答えることで、半自動的に動作をイベントとして定義できる。コンテキスト定義ツールは、3 軸加速度センサ、3 軸角速度センサ、温度センサ、GPS、RFID タグリーダを用いた動作認識を実現している。

3. 関連研究

3.1 多種のセンサを扱うツールキット

センサなどの様々なデバイスをプログラムの中で容易に利用するためのツールキットはこれまでも数多く研究、開発されており、すでに多くの製品が販売されている。

Phidgets⁵⁾ はボタンやスライダ、ミニジョイスティック、タッチセンサ、光センサ、RFID リーダなどの各種センサや、サーボモータや LED などのアクチュエータを統一したインタフェースで扱えるようにしたパッケージである。一般発売されており、Java、Visual Basic、C++、Delphi など数多くのプログラミング言語用のライブラリが公開されているため、プログラミングの知識があればセンサを用いた任意のアプリケーションを開発できる。

Teleo⁴⁾ は MakingThings 社の製品で、Phidgets と同様にいくつかのモジュールを組み合わせることで、各種センサやモータを入出力としたアプリケーション開発を支援するパッケージである。C や C++ などで作成したアプリケーションから直接 Teleo を利用するため

の開発キット、音楽/マルチメディア向けの統合開発環境としてメディアアート分野で多用されている Max/MSP 用のオブジェクトも提供されており、センサやアクチュエータを簡単に利用できる。

Gainer¹³⁾ は、既存の各種センサやモータを自由に接続し、活用するためのプラットフォームである。Gainer を用いることで、インタフェースやメディア・インストールの開発を行える。Gainer では、Phidgets や Teleo と異なり、専用のセンサを用いるのではなく、ハードウェアの設計を自分で行うための機構も有しており、自由にセンサやモータを利用できる。Max/MSP や Processing, Flash との連携を行うためのライブラリも提供されており、容易にセンサを用いたコンテンツ開発を行えるようになる。

これらのツールキットを用いることでセンサを用いたアプリケーション開発は容易になるが、これらのツールキットはアプリケーションとセンサの接続を容易にするためのものであり、センサデータの扱いを支援するものではない。たとえば、ある一定の温度を超えたときに動作するといった簡単な利用は容易に行えるが、加速度センサを用いてプレイヤーの動作を認識し、その結果を用いるようなコンテンツの制作は生のセンサデータから動作を認識する必要があるため、センサデータからプレイヤーの動作を認識する計算に関する知識が必要となる。本研究で提案する機構では、ゲーム制作者はセンサの接続だけではなくセンサデータの処理方法に関しての知識も十分ではないと考え、センサデータを用いたユーザの状況や動作の認識までを簡単に活用できるようにすることで、体感型コンテンツ制作時にセンサを意識する必要のない枠組みを実現している。

また、既存のアプリケーションに対して様々なセンサを有する既存のデバイスを連携させるためのツールとして GlovePIE³⁾ がある。このツールでは、簡単なスクリプト記述を用いて、アプリケーションの機能に対してデバイスの持つ機能を割り当てられる。ボタンや角度などその値を割り当てただけであれば容易に連携を行える。

同様に、仮想空間において様々な入力インターフェースを汎用的に利用するために仮想空間プラットフォームを提供している VR Juggler⁸⁾ やライブラリを提供している VRPN⁷⁾ がある。VR Juggler は、様々な入出力デバイス、様々な OS をサポートする仮想空間のミドルウェアを提供しており、容易に体感型仮想空間を構築できる。VRPN は、同じ機能を持つ異なるデバイス間の違いを吸収する仕組みを有しているため、多様なデバイスを簡単に利用できる。

しかし、GlovePIE, VR Juggler や VRPN でもセンサデータの解釈までは支援しておらず、コンテンツ制作者がセンサデータからプレイヤーの動作を認識するアルゴリズムを知る必要があるなど、センサの特徴を意識したコンテンツ制作が求められる。

3.2 カメラを用いたコンテンツ制作ツールキット

ゲームなどのコンテンツにプレイヤーの動作を入力する方法として、カメラを用いる方法がある。実際に Xbox Live ビジョン⁹⁾ や Eyetoy²⁾ など、カメラを用いた体感型ゲームも実現されている。このようなカメラを用いたジェスチャ認識やコンテンツ制作支援の研究も数多く存在しており、GT2K¹⁴⁾ は、環境に設置した、もしくはプレイヤーの装着しているカメラを用いて、カメラの前で行った手の動きを認識し、利用するツールキットである。ARToolkit¹²⁾ や DART¹¹⁾ は、カメラで取得したビジュアルマーカと呼ばれる 2 次元コードとカメラの相対位置を取得し、コンテンツ制作に利用する。

カメラを用いた動作認識は、特定のスペースで個人を対象とした認識としては、体感型ゲームコンテンツ制作には活用できると考えられるが、光の加減によって動作が安定しない、複数人を対象とした認識が困難、カメラの視覚が多数存在するなど、ゲームコンテンツに利用するには課題も多い。

3.3 例示によって動作の定義を行うシステム

実際にユーザが動作を行う例示により動作を容易に定義することを狙った研究として、a CAPpella システムが提案されている¹⁰⁾。しかし、このシステムは動作定義後のプログラミングモデルは言及されておらず、本研究で対象としている Flash コンテンツクリエイタのように既存の開発者を想定したものではない。

4. 提案フレームワーク

提案フレームワークを用いて作成する Flash ゲームコンテンツは、図 3 に示すように、センサやアクチュエータを管理するデバイス管理層と Flash コンテンツが動作する Flash コンテンツ層の 2 層で動作する。図中のコンテキスト定義ツールは、Wearable Toolkit に付属のセンサデータから動作や状況を認識するツールである。Flash Plug-in, および Communication Class は、本研究で実装したもので、後述の Wearable Toolkit と Flash の通信を実際に行う部分である。Flash ゲーム制作者にとってセンサの接続やセンサデータから人の動作を認識することは難しくプログラミングも大変であるが、体感型ゲームに動作を利用する場合、どのような動作が行われているかといったセンサデータを解釈した結果を取得できればコンテンツ制作が可能となる。したがって、提案フレームワークでは、センサデータを解釈するデバイス管理層と Flash コンテンツを動作させる Flash コンテンツ層の 2 層に分けた構成とし、コンテンツ制作者が制作する Flash コンテンツ層からデバイス操作を排除した。なおデバイス管理層では、2 章で述べた Wearable Toolkit を用いてセンサデータ

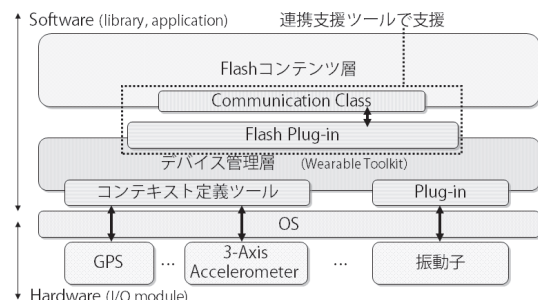


図3 提案機構の位置づけ

Fig.3 Structure of proposed framework.

の解釈、アクチュエータの制御を行う。

このような構成に基づき、本研究では、Wearable Toolkit と Flash の通信処理の実現、およびこれらの通信処理の実装と連携を支援するツールの開発を行った。さらに、Wearable Toolkit のコンテキスト定義ツールではゲームコンテンツに不可欠なプレイヤー動作の信頼度を取得できないため、プレイヤー動作の信頼度も取得するようツールを拡張した。

4.1 2層化に対する議論

提案フレームワークでは、Flash コンテンツ部とデバイス管理層である Wearable Toolkit の2層で動作する。Flash からは体感型ゲーム制作に必要なセンサやアクチュエータを直接利用できないため、実際にデバイスの制御を行うデバイス管理層を加えた2層で動作させる。

デバイス管理層で行う処理について、以下の方法が考えられる。

- (1) デバイスとコンピュータの接続のみを行い、センサデータを Flash にそのまま渡す。
- (2) デバイスとコンピュータの接続、およびセンサデータの解釈を行い、解釈結果をイベントとして Flash に渡す。

(1)の方法は、3章で説明したすべての既存研究で用いられている方法で、すべてのセンサデータを Flash で処理できるため、センサを活用した多様なコンテンツ開発が行える。しかし、センサデータを用いたプレイヤーの動作認識を Flash で行うには、複雑な数値計算を Flash 上で行う必要があり、計算オーバーヘッドが大きい。また、コンテンツ上につねに実行される処理を記述する必要があり、簡単なゲーム制作においても大規模なプログラミングが必要となる。

(2)の方法は、デバイス管理層でプレイヤーの動作認識を行い、動作が認識された際にイベントとして Flash にメッセージを送る。この方法では、Flash コンテンツ上で動作認識を行わないため、コンテンツ制作者の自由度は減るが、簡単な体感型ゲームにおいてはプレイヤーの動作認識アルゴリズムを変更することはない。また、デバイス管理層でセンサデータから動作を認識し、動作の ON/OFF をメッセージとすることで、これまでの Flash ゲーム制作で用いられるボタン入力と同じ仕組みでゲーム制作に利用できるため、Flash ゲーム制作者にとっても親しみやすいと考えられる。

以上の議論から、提案フレームワークでは、(2)のセンサデータの解釈までをデバイス管理層で行う方法を用いる。

4.2 Wearable Toolkit と Flash の通信

本研究では、Wearable Toolkit と Flash コンテンツ間の通信には、Flash コンテンツの呼び出し元となるホストアプリケーションとの通信を行うコマンドを用いる。具体的には、Wearable Toolkit が Flash コンテンツを呼び出すことでホストアプリケーションとなり、通信を行う。この通信方法は、ストリームデータのような連続したデータなどの通信はできないが、簡単なテキストメッセージのやりとりを行う負荷の小さい通信手段である。他の方法として Socket 通信を用いる手段もあるが、この方法ではセンサデータなどの連続データを扱う一方で、つねにデータの到着を監視する必要があるなど簡単なメッセージのやりとりと比較すると手軽ではない。前節で述べた提案フレームワークの場合、前者の簡単なテキストメッセージのやりとりで十分に動作を通信できるため、本研究では前者の通信方法を用いる。

提案フレームワークでは、センサデータからプレイヤーの動作を認識したとき、Flash ゲームコンテンツからアクチュエータを作動させるとき、初期起動時の3種類の状況で通信を行う。初期起動時には、Flash ゲームコンテンツで用いる動作やアクチュエータ操作に必要なメッセージの定義を各層で同期させるため、通信が必要となる。以降でこれらの状況における通信について、処理の詳細を説明する。

4.2.1 Flash コンテンツの起動時の通信

Flash ゲーム起動時に、Flash ゲームコンテンツで利用するプレイヤーの動作メッセージと Flash ゲームコンテンツからアクチュエータを作動させるときに発生するメッセージを Wearable Toolkit に通知する。Wearable Toolkit は受け取った情報をもとに ECA ルールを動的に生成する。ルール生成までの流れを図4に示す。

起動した Flash コンテンツは、図5に示すようなスクリプトを Flash コンテンツの初期

(1) Wearable Toolkit起動

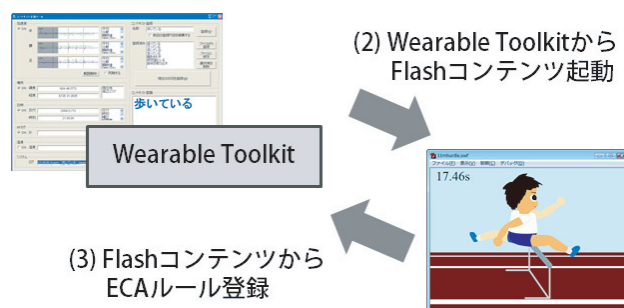


図 4 起動時の動作の流れ

Fig. 4 The flow of initialization.

```

RegisterAction("右手パンチ", OnRightPunch, "右手パンチ", "null");
RegisterEvent("右手振動子作動", "右手振動子を作動", "Number");
function OnRightPunch(Arg:引数の型){
    //ここに右手パンチ認識時の処理を記述
}

```

図 5 初期起動時に呼び出されるスクリプト

Fig. 5 Scripts that called at initialization of flash contents.

起動時に呼び出す。このスクリプトによって、Wearable Toolkit は各 Flash コンテンツがユーザのどのような動作を必要とするのか、Flash からのどのようなメッセージをトリガとしてどのアクチュエータを作動させるのかなどの、ゲームコンテンツごとに異なるパラメータを定義できる。Flash コンテンツ層では Wearable Toolkit への通知と同時に、ユーザの動作認識時に実行する ActionScript との関連付けを行う。図 5 に示す例の場合、“右手パンチ”という動作に対して、OnRightPunch() 関数が関連付けられる。

4.2.2 動作認識時の通信

ゲームコンテンツが起動したら、Flash コンテンツ層とデバイス制御層である Wearable Toolkit は独立して動作を行う。動作認識時の処理の流れを図 6 に示す。

図 6 の (3) の動作送信では、初期起動時に登録された、動作認識メッセージを Flash コンテンツ層へ送信する ECA ルールが処理される。ここで、Wearable Toolkit から Flash コン

(1) センサ群から常時動作を判断

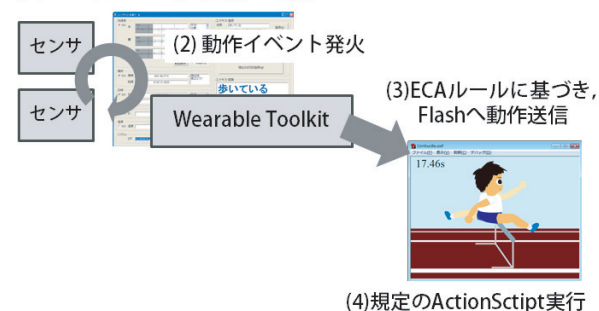


図 6 動作認識時の動作の流れ

Fig. 6 The flow of gesture recognition.

(1) アクチュエータ動作要求



図 7 アクチュエータ動作の流れ

Fig. 7 The flow of communication to actuate devices.

テンツへの通信には、Flash コンテンツ内の特定の変数を書き換える SetVariable() 関数をホストアプリケーションから実行することで行う。Flash コンテンツ側では、書き換えられた変数が変化したときに呼ばれるコールバック関数と実行したい ActionScript を初期登録時に関連付けしており、該当する ActionScript が実行される。図 5 に示す例では OnAction() 関数が実行される。

4.2.3 Flash からアクチュエータを動作させるときの通信

Flash コンテンツからアクチュエータを動作させるためには、ActionScript からホストアプリケーションへテキストメッセージを送る fscommand を実行する。アクチュエータの動作は、図 7 に示す流れで行われる。

図 7 の (2) で実行される動作イベントには、初期登録時に登録したイベント名やその引



図 8 連携支援ツール

Fig. 8 A snapshot of supporting tool for content creation.

数を指定する。

4.3 連携支援ツール

本研究で提案している流れに従って Flash と Wearable Toolkit の通信を行うためには、前節で述べたスクリプトの記述が必要である。このスクリプトは、ある利用したい動作や操作に対して 1 対 1 で定義できるため、本研究ではさらに、そのスクリプト記述の手間をなくす連携支援ツールを実装した。連携支援ツールを用いることで、プレイヤーの動作を利用するためにコンテンツ制作者が記述するスクリプトは、動作認識時に行う ActionScript だけとなる。また、アクチュエータの制御に必要なスクリプトは、1 文の ActionScript を入力するだけで生成できる。連携支援ツールの画面例を、図 8 に示す。ゲームコンテンツに必要な情報を入力し、図中の出力ボタンを押すことで、Wearable Toolkit と Flash コンテンツの通信に必要なスクリプトが記述された状態の Flash プロジェクトファイルが出力できる。

5. アプリケーション実装例

本章では、提案機構を用いてどのような手順でコンテンツ開発を行うかを、図 9 に示す 110 m ハードル競技の体感型ゲームを作成した例を交えて具体的に説明する。

体感型ゲームコンテンツ開発における主な流れは、以下のとおりである。

Step1 コンテンツの設計

Step2 コンテキスト・動作の登録

Step3 Wearable Toolkit のルール定義

Step4 Flash コンテンツの作成

Step5 デバッグ、テスト

提案機構は、従来の Flash コンテンツ制作では必要のなかった Step2, Step3, Step4 を支援する機構である。各ステップについて、以下で詳細に説明する。

Step1 コンテンツの設計

このステップでは、作成するコンテンツの全体像を考える。コンテンツの目的などの通常のコンテンツ制作で考慮する点に加えて、体感型ゲーム制作においては、特にプレイヤーの動作を Flash コンテンツにどのように利用するのか、およびどのようなアクチュエータをいつ利用するのかを明確にする必要がある。

たとえばハードル競技ゲームの場合、画面の合図によってゲームがスタートし、110 m の距離を走りきるまでのタイムを競うという、ゲームの要旨に加え、プレイヤーの「走る」「止まる」という動作を Flash ゲームコンテンツ内のキャラクタの「走る」「止まる」、プレイヤーの「ジャンプ」は、キャラクタの「ジャンプ」に利用するという、ユーザ動作とコンテンツの関連を明確にする。また、キャラクタがハードルに引っかかった際に、振動子を振動させるといったアクチュエータの利用も設計する。

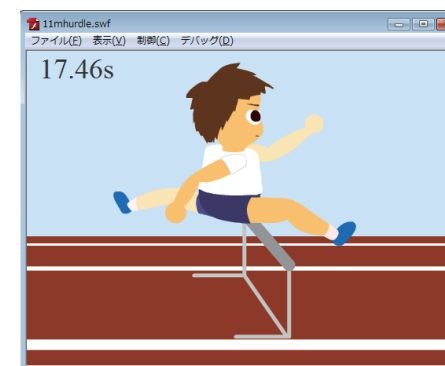


図 9 110 m ハードル競技ゲーム

Fig. 9 A 110 metre hurdles game.



図 10 走るイベントのアクションの定義

Fig. 10 Definition of the action which is executed in Running.

Step2 コンテキスト・動作の登録

Step1 で計画したコンテンツに利用するプレイヤーの動作を、Wearable Toolkit のコンテキスト定義ツールを用いて登録する。コンテキスト定義ツールは、登録したい動作を実際に行うことで例示した後、数回の簡単な質問に答えるだけで、プレイヤーの動作を定義できるツールである¹⁵⁾。さらに、同ツールを用いて実際に正しく動作が認識できるか確認する。

ハードル競技ゲームの例では、本ステップで「走る」「止まる」「ジャンプ」の3つの動作をコンテキスト定義ツールを用いて登録し、正しく認識できていることを確認する。

Step3 Wearable Toolkit のルール定義

Wearable Toolkit の動作は、本研究で実装した連携支援ツールを用いることで容易に定義できる。Step1 で設計したプレイヤーとコンテンツの関連を連携支援ツールに入力すると、動作に必要なルールを生成できる。

具体的にハードル競技ゲームの場合は、「走る」動作を認識した際に、ゲーム上のキャラクターを「走る」に変更するため、図 10 に示すように、ユーザの動作に「走る」を入力し、さらに動作認識時にコンテンツで実行するスクリプトを記述する。ここで、動作認識の確度の閾値を決定することで、動作の認識されやすさと誤認識のされにくさを調節できる。この入力作業により、コンテンツの動作に必要な ActionScript と Wearable Toolkit で用いるルールが生成される。出力される Wearable Toolkit で用いるルールを図 11 に示す。同様

```
DEFINE DO_RUN
WHEN CONTEXT_RECOGNIZED(running)
THEN DO FLASH_DO_RUN()
```

図 11 出力される Wearable Toolkit のルール

Fig. 11 A output rule of Wearable Toolkit.

```
wtkLib.RegisterEvent("FLASH_VIBRATION", "振動子を振動", "Number");

//このコードを適切なところに貼り付けてください
wtkLib.RaiseEvent("FLASH_VIBRATION", "1");
```

```
DEFINE VIVRATION
WHEN FLASH_VIBRATION()
IF NEW.DATA1 == "1"
THEN DO CMN_VIBRATE(1)
```

図 12 出力される ActionScript とルール

Fig. 12 A output ActionScript and a rule.

に、「止まる」「ジャンプ」も定義する。

さらに、Flash コンテンツからアクチュエータを利用する関連付けも定義する。具体的には、振動子を振動させる場合、Wearable Toolkit で利用できるアクチュエータの一覧から振動子を振動させる機能を選択し、その機能に名前を付ける。実際に Flash コンテンツから振動子を動作させる際には、ここで定義した名前のイベントを Flash コンテンツ内から呼び出すことで、その機能が利用できる。この際、出力される ActionScript、およびルールを図 12 に示す。

Step4 Flash コンテンツの作成

ここで、Flash のコンテンツ作成に着手する。Step3 で入力した連携支援ツールの「出力」ボタンを押すことで、Wearable Toolkit と Flash の通信に必要な基本的なスクリプトの定義されたプロジェクトファイルが出力される。コンテンツ制作者は、それをもとに Flash コンテンツの作成を行う。

Flash からのアクチュエータの操作は、操作命令を発行したい場所で前節で定義したイベント名を引数にした RaiseEvent() 関数を呼び出す。ハードル競技ゲームの例では、ゲーム

キャラクターがハードルにぶつかったときにアクチュエータを振動させるため、ハードルとの当たり判定を行うところに、Step3 で出力された RaiseEvent() 関数を挿入する。

Step5 デバッグ、テスト

最後に、デバッグとテストを行う。デバッグは、Wearable Toolkit に付属のデバッグを用いることで、Flash コンテンツを起動した際にアクション、イベントが登録されているかを確認できる。また、コンテンツにおいては動作認識の精度も重要なポイントとなる。コンテキストの認識精度はコンテキスト定義ツールを用いた動作の再定義や連携支援ツールを用いた認識精度の閾値の変更などで調節する。

作成したコンテンツの拡張などで新たに連携を追加で定義する際にも、連携支援ツールを用いることで、ユーザの動作やアクチュエータの操作 1 つ 1 つに対してルールや ActionScript を出力できるため、容易に拡張できる。

6. 考 察

6.1 コンテンツ制作時間の評価

本提案機構を用いることで、どれだけ短時間で体感型コンテンツが制作できるかについて議論する。まず初めに提案機構の動作確認として、キーボード入力で行うブロック崩しゲームを提案機構を用いて体感型ゲームコンテンツとして動作させた。制作では、キーボード入力の代わりにプレイヤーの 2 種類の動作（右手を右にひねる、右手を左にひねる）を用いるよう修正し、ゲームを行えるようにしたところ、1 時間以下で体感型ゲームとして動作させられた。プレイヤーの手をひねる動作をセンサデータから算出するプログラムを記述するためには 1 時間以上の時間を有すると考えられるが、提案機構では既存の動作認識ツールであるコンテキスト定義ツールを用いることで、その時間を短縮できたと考えられる。

次に、体感型コンテンツ制作に対して特別な知識のない Flash コンテンツ作成者が提案機構を利用できるか評価を行った。まず初めに、筆者が下記の 2 つのゲームコンテンツを提案機構を用いて作成した（図 13）。

- ボールに対してタイミング良く手を振るとゲームキャラクターがバットを振って、ホームランが打てるホームラン競争
 - 右手と左手を交互に揺くとゲームキャラクターが泳ぎ、ゴールまでの時間を競う水泳ゲーム
- コンテンツ制作の結果、ホームラン競争に 2 時間、水泳ゲームに 3 時間を要した。しかし、そのほとんどがゲームの画像作成のための時間で、Flash との連携部分の定義に要した時間は各 15 分程度であった。

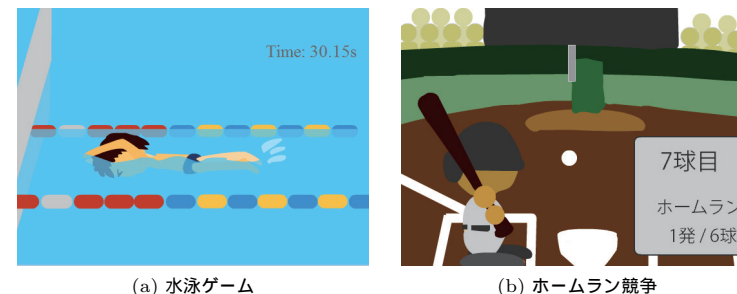


図 13 作成した体感型ゲーム

Fig.13 Created games.

次に、Flash コンテンツ制作歴約 4 年で、日常から趣味で Flash コンテンツを作成する Flash コンテンツ作成に慣れた 25 歳の男性を被験者として、実験を行った。評価実験では、この被験者に対して提案機構、提案ツールの使い方を十分に説明し、実際にサンプルアプリケーションと一緒に作成した後、筆者が作成したものと同一 2 つのコンテンツを作成するよう指示した。

被験者には、さらに実際に筆者が作成したそれぞれのゲームのプロトタイプを見せ、具体的にゲームの完成をイメージできる状態にした。また、ゲームの細かな作りこみや絵の作成には極力時間をかけず、ゲームの完成を急ぐように指示した。実験には筆者も立ち会い、被験者の疑問に答えられる状態で実験を行った。ただし、実験中に得られた質問は、作成するゲームの質に対する質問のみで、実験中に機構の仕組みや操作に対する質問はなかった。また、ゲームで用いる画像データは筆者が用意し、ゲーム作成の負荷を軽減した。

実験の結果、ホームラン競争も水泳ゲームのどちらも約 1 時間で体感型ゲームコンテンツが完成した。ゲーム作成時間の内訳は、初めの 15 分間に連携ツールを用いた Flash プロジェクトファイル作成、残りの 40 分間で画像のレイアウトや ActionScript のプログラミングなどのコンテンツ作成、最後の 5 分程度でプレイヤー動作の登録を行っていた。さらに、ゲームコンテンツ作成後の感想として、「こんなに簡単に体感型ゲームを作成できるとは思わなかった」という肯定的な意見も得られたものの、「連携ツールの使い方が直観的でなく、使いづらい」という意見も得られ、連携ツール GUI の改善には余地があることも分かった。

またゲーム以外のコンテンツとして、神戸ルミナリエのイルミネーション募金箱のディスプレイ部に本システムで作成したコンテンツを実装した。この募金箱コンテンツは、募金の



図 14 募金箱コンテンツ
Fig. 14 A content for collection box.

呼びかけを行うユーザの行動を認識し、ディスプレイ上の募金を呼びかけるキャラクターの動きに反映させるというものである。たとえば、ユーザのお辞儀を認識すると、図 14 に示すように募金箱内のキャラクターもお辞儀をする。募金箱コンテンツは、筆者を含む提案機構を熟知したグループで企画し、実装は筆者が行った。コンテンツ作成は 3 時間程度で完了し、コンテンツ制作者に困難と考えられるセンサの制御部分や Flash コンテンツとの連携部分の作成には、10 分程度しか要さなかった。募金箱コンテンツの作成を通じ、ゲームコンテンツなどユーザの動作認識率がゲームの難易度に直接影響するコンテンツは、認識率に対する認識の閾値の変更などの試行錯誤が必要となるが、ユーザの動作認識率が大きく影響しないコンテンツの作成は特に短時間でできることが分かった。

提案フレームワークを用いずにセンサを活用した体感型コンテンツを作成するには、既存研究で述べた多種のセンサを扱うツールキットを用いるなどして、Flash でセンサデータを処理する必要がある。しかし、多くの Flash コンテンツ制作者はセンサデータから人の動作を認識するアルゴリズムに対する知識がないため、簡単には作成できない。仮に知識がある場合でも、Flash 上でセンサデータから人の動作などを認識するための数値計算処理を記述する必要があり、大変な実装時間を要する。またカメラを用いた動作認識では、死角の影響や複数プレイヤーへの対応が困難であるなどの別の問題が発生するため、一概に比較できない。提案フレームワークでは、別の用途にも用いられている既存の動作認識機構を利用す

ることで、Flash を用いた体感型コンテンツの作成を容易にしておき、明らかに開発時間を短縮しているといえる。

6.2 既存の Flash コンテンツとの差異

キーボードやマウスで操作するコンテンツは、確実に認識できる操作をもとにして設計されているため、そのようなコンテンツの操作をユーザの動作認識に置き換えるだけでは、ユーザが操作の不自由を感じたり、難易度が高いと感じたりすることが分かった。たとえば 110m ハードル走ゲームにおいて、キーボード操作では確実にキャラクタをジャンプさせられるにもかかわらず、プレイヤー動作のジャンプの認識率が 100%ではなく、認識に失敗しハードルを越えられない場合が多く発生する。つまり、既存のゲームではユーザの操作が行われればそれが実行されることが保証されているが、体感ゲーム化することによって、ユーザが操作を行っても認識エラーにより操作が行われていないと判断される可能性があり、このことがユーザにとって、ゲームの難易度が向上している、あるいは、ゲームが操作しにくくなった、という印象を与えていると考えられる。そのため加速度センサなどのセンサ値からジェスチャを認識する必要のあるような体感型コンテンツを制作する際には、ユーザの動作認識が必ずしも正確に行われるわけではないことを十分に理解したうえで、コンテンツを制作する必要がある。ただしこの問題は、本機構による問題ではなく、動作認識や体感型コンテンツそのものに対する課題である。

6.3 幅広い Flash コンテンツへの対応

提案フレームワークは、Flash コンテンツを作るためのプロジェクトファイルの段階で、Wearable Toolkit との通信部分を記入しているため、Flash コンテンツを一から作成する、または、Flash コンテンツのソースファイルが存在する場合にのみ、体感型コンテンツに拡張できる。ただし、Flash コンテンツファイルである swf ファイルは仕様が公開されているため、提案フレームワークを拡張して、Flash コンテンツのプロジェクトファイルがなくても、swf ファイルから体感型コンテンツを作成できるようにする予定である。既存の Flash コンテンツを体感型コンテンツに変換できるようになると、多くの体感型コンテンツを容易に作成できると考えられる。ただし、既存の Flash コンテンツは確実に認識できる操作をもとに設計されているため、前節で述べた認識精度に起因する問題は残る。

7. ま と め

本研究では、センサ技術を簡単に扱う機能を有する Wearable Toolkit と Flash コンテンツを連携させる機構を実現することで、Flash コンテンツ制作者が容易に短時間でセンサ技

術を活用したコンテンツを作成できることを確認した。Wearable Toolkit と連携させることで、センサによって認識されたプレイヤーの動作を利用するだけでなく、振動子などの他のデバイスとの連携も可能になるなど、Flash コンテンツ制作者にとって幅広い選択肢を提供できるものとなった。さらにコンテンツ制作者は、本研究で作成した連携を支援するツールを利用することで Wearable Toolkit との通信も意識する必要がなくなり、新たに知識を増やさなくても体感型ゲームコンテンツを作成できるようになった。

今後は、より多くの Flash コンテンツ制作者に提案機構を利用してもらい、体感型コンテンツ制作時間、およびその手軽さに関する評価を行う予定である。

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）および文部科学省科学研究費補助金基盤研究（A）（20240009）、特定領域研究（21013034）、特別研究員奨励費（19-55371）によるものである。ここに記して謝意を表す。

参 考 文 献

- 1) Adobe Flash CS4 ホームページ .
<http://www.adobe.com/jp/products/flash/>
- 2) EyeToy ホームページ .
<http://www.jp.playstation.com/scej/title/eyetoy/>
- 3) GlovePIE ホームページ .
<http://carl.kenner.googlepages.com/glovepie>
- 4) MakingThings 社ホームページ .
<http://www.makingthings.com/>
- 5) Phidgets ホームページ .
<http://www.phidgets.com/>
- 6) Wearable Toolkit ホームページ .
<http://wearable-toolkit.com/>
- 7) VRPN ホームページ .
<http://www.cs.unc.edu/Research/vrpn/>
- 8) VR Juggler ホームページ .
<http://www.vrjuggler.org/>
- 9) Xbox Live ビジョンホームページ .
<http://www.xbox.com/ja-JP/hardware/x/xboxlivevision/>
- 10) Dey, A.K., Hamid, R., Beckmann, C., Li, I. and Hsu, D.: A CAPpella: Programming by Demonstration of Context-Aware Applications, *Proc. International Conference on Human Factors in Computing Systems (CHI2004)*, pp.33-40 (2004).
- 11) MacIntyre, B., Gandy, M., Dow, S. and Bolter, J.D.: DART: A Toolkit for Rapid

Design Exploration of Augmented Reality Experiences, *Proc. Conference on User Interface Software and Technology (UIST'04)*, pp.197-206 (2004).

- 12) Kato, H., Billingham, M., Poupyrev, I., Imamoto, K. and Tachibana, K.: Virtual Object Manipulation on a Table-Top AR Environment, *Proc. International Symposium on Augmented Reality (ISAR2000)*, pp.111-119 (2000).
- 13) Kobayashi, S., Endo, T., Harada, K. and Oishi, S.: GAINER: A reconfigurable I/O module and software libraries for education, *Proc. International Conference on New Interfaces for Musical Expression (NIME'06)*, pp.346-351 (2006).
- 14) Westeyn, T., Brashear, H., Atrash, A. and Starner, T.: Georgia Tech Gesture Toolkit: Supporting Experiments in Gesture Recognition, *Proc. International Conference on Perceptive and Multimodal User Interfaces (ICMI2003)*, pp.85-92 (2003).
- 15) 寺田 努, 宮前雅一, 山下雅一: Wearable Toolkit: その場プログラミング環境実現のためのイベント駆動型ルール処理エンジンおよび関連ツール, 情報処理学会論文誌, Vol.50, No.6, pp.1587-1597 (2009).

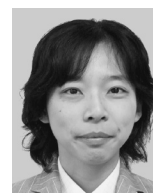
(平成 22 年 1 月 26 日受付)

(平成 22 年 9 月 17 日採録)



田中 宏平（正会員）

2005 年大阪大学工学部卒業。2007 年同大学院情報科学研究科博士前期課程修了。2010 年同研究科博士後期課程修了。同年三菱電機株式会社先端技術総合研究所入社。博士（情報科学）。在学時、ウェアラブルコンピューティング・ユビキタスコンピューティングの研究に従事。



岸野 泰恵（正会員）

2002 年大阪大学工学部卒業。2004 年同大学院情報科学研究科博士前期課程修了。2007 年同研究科博士後期課程修了。同年日本電信電話株式会社入社。博士（情報科学）。ユビキタスコンピューティング、センサネットワークに関する研究に従事。



寺田 努 (正会員)

1997 年大阪大学工学部情報システム工学科卒業．2000 年同大学院工学研究科博士後期課程退学．同年より大阪大学サイバーメディアセンター助手．2005 年より同講師．2007 年神戸大学大学院工学研究科准教授，現在に至る．2004 年より特定非営利活動法人ウェアラブルコンピュータ研究開発機構理事．2004 年には英国ランカスター大学客員研究員を兼務．博士 (工学)．ウェアラブル・ユビキタスコンピューティングの研究に従事．IEEE，電子情報通信学会，日本データベース学会，ヒューマンインタフェース学会の各会員．



宮前 雅一

2001 年大阪大学工学部電子情報エネルギー工学科情報システム工学科目卒業．2003 年同大学院工学研究科情報システム工学専攻博士前期課程修了．2006 年同大学院情報科学研究科マルチメディア工学専攻博士後期課程修了．博士 (情報科学)．同年 (株) 国際電気通信基礎技術研究所知識科学研究所研究員．2008 年ウエストユニティス (株) 入社，現在に至る．ウェアラブル・ユビキタスシステムの研究開発に従事．



西尾章治郎 (フェロー)

1975 年京都大学工学部数理工学科卒業．1980 年同大学院工学研究科博士後期課程修了．工学博士．京都大学工学部助手，大阪大学基礎工学部および情報処理教育センター助教授，大阪大学大学院工学研究科情報システム工学専攻教授を経て，2002 年大阪大学大学院情報科学研究科マルチメディア工学専攻教授，現在に至る．2000 年より大阪大学サイバーメディアセンター長，2003 年より大阪大学大学院情報科学研究科長，その後 2007 年より大阪大学理事・副学長に就任．この間，カナダ・ウォータールー大学，ピクトリア大学客員．データベース，マルチメディアシステムの研究に従事．現在，Data & Knowledge Engineering 等の論文誌編集委員．本会理事を歴任．本会論文賞を受賞．電子情報通信学会フェローを含め，ACM，IEEE 等 8 学会の各会員．