音声認識と拡張現実技術を用いたモバイル型スイカ割りゲーム

13N3003　石川 大輔　　指導教員　岩月 正見教授

1. 背景と目的

近年、ゲーム業界の技術の進歩は著しく、その中でも、AR や VR という視覚に現実味を帯びさせる技術が注目を集めている。このような技術の登場により、ゲームの中の世界と現実の世界が徐々に近づき、交わり始めている。しかしながら、現在提供されている AR や VR のコンテンツでも、ゲームを操作するためにコントローラーやキーボードなどのインターフェイスが使われていることがほとんどである。このようなインターフェイスを工夫することで、より現実味を帯びたコンテンツを生み出せる可能性を感じたことが本研究の背景である。そこで本研究では、近年、マウスやキーボードに比べ、より自然で高速な入力が可能なインターフェイスとして、PC やスマートフォン、タブレットなどに搭載されている音声認識技術を AR や VR に組みこむことでゲームの世界のモノを現実の世界のモノのように自分の声で扱うことができるようにすることでゲームの世界を現実の世界に近づけることを目的とする。

2. 概要

本アプリケーションは、キャラクタをユーザの音声によって誘導し、スイカ割りをするゲームであり、Android OS を搭載したモバイル端末で動作する。ユーザは、スイカ割りをするキャラクタに指示を送る人間になり、後述の図 1 のように、目隠ししたキャラクタを誘導してスイカを割らせるという内容となっている。本アプリケーションは、スタートシーン、ゲームプレイシーン、結果表示シーンの 3 つのシーンで構成されている。アプリケーション起動時にまずスタートシーンが表示され、スタートボタンをタップすることでゲームプレイシーンへと遷移する。ゲームプレイシーンでは、音声認識機能を駆使し、目隠しをしたキャラクタに「右」「左」「逆」などの指示を送り、スイカのある位置まで誘導し、「叩け」という合図を発することでスイカを割らせる。その後、スイカを割ることができたかどうかによって、成功、または失敗の結果表示シーンへと遷移する。

図 1　ゲームプレイシーン

3. 開発環境

3.1　Unity5.4.3

Unity は Windows や Mac OS X 上で動作し、数多くのプラットホームに対応した 3D・2D ゲームを製作することができる統合開発環境を内蔵したゲームエンジンである。現在では Windows、Mac OS X、Android、iOS、家庭用ゲーム機などのプラットホームで動作するアプリケーションを製作することができ、今後も対応プラットホームは増えていく見込みである(文献 1)。近年ではゲーム以外にも、デジタルアートや災害仮装シミュレータといった幅広い分野にも応用されている。本アプリケーションでは、3 つのゲーム上のシーンを作るために使用している。

3.2　Vuforia6.2.6

Vuforia とは PTC 社が提供する AR 製作用ライブラリである。認識精度が高いことでも知られ、平面のマーカーだけではなく、立体のマーカー認識、クラウドでも認識、カメラからマーカーが離れた際の追従認識など、様々な形で AR の機能を簡単に実装できる(文献 2)。

3.3　Android SDK

Android SDK とは、PC 等で Android 向けソフトウェアを開発するための開発環境である。Google 社が無償で公開しているもので、Windows 版、Mac OS X 版、Linux 版がある。Android OS を搭載したスマートフォンやタブレット端末で動作するプログラムを開発するために必要なソフトウェアをひとまとめにしたパッケージで、コンパイラやデバッガ、ライブラリ、デバイスドライバ、エミュレータなどで構成される(文献 3)。本研究では、アプリケーションを Android OS 搭載端末で動作させるために Android SDK を使用している。

3.4　Watson Developer Cloud Unity SDK0.14.0

Watson は、IBM が開発した質疑応答システム・意思決定支援システムである。『人工知能』と紹介されることもあるが、IBM は Watson を、自然言語を理解・学習し人間の意思決定を支援する『コグニティブ・コンピューティング・システム』と定義している(文献 4)。本研究では、Watson が提供する API の 1 つである『Speech to Text』を、音声認識技術として使用している。

4. システム概要

4.1　音声認識

本アプリケーションでは、音声認識機能部に、先述の通り、Watson Developer Cloud Unity SDK に含まれている API の 1 つである『Speech to Text』を使用している。図 2 にこのアプリケーションの音声機能部図解を示す。

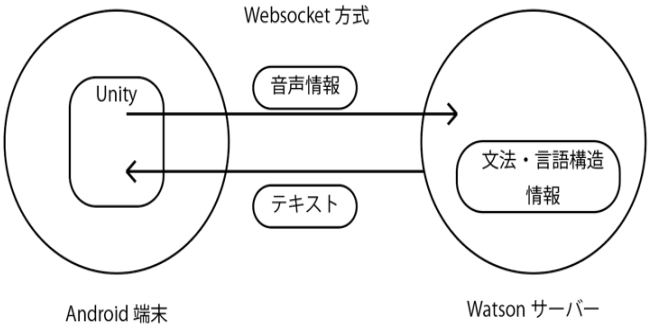

図 2　音声認識機能部システム図解

図 2 の通り、Android 端末にあらかじめ搭載されているマイクを使用し入力された音声データは、一度 Watson のサーバに送られる。Watson のサーバ内では、Watson があらかじめ保有している文法や言語構造に関する情報と、送られてきた音声情報の文法や言語構造に関する情報を比較し、音声情報をテキストに変換し、送り返している。なお、このときの通信規格には、WebSocket 通信が使用されている。

4.2 キャラクタ制御

まず、本アプリケーションの一連の流れをフローチャートとして図 2 に示す。

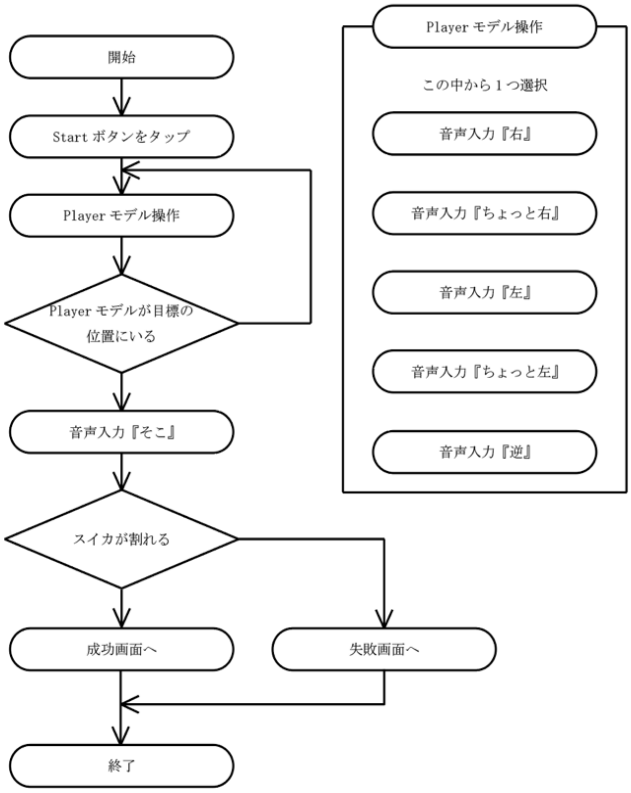

図 3　フローチャート

本アプリケーションでは、キャラクタモデルを誘導するために、図 3 のフローチャート中に記述しているように、下記の 6 つの音声コマンドを用意している。

①「右」
②「左」
③「ちょっと右」
④「ちょっと左」
⑤「逆」
⑥「そこ」

上記①～⑤までは、キャラクタモデルの進行方向を制御する音声コマンドであり、⑥はスイカを割るための音声コマンドである。これらはそれぞれ条件分岐によって制御されており、音声入力の中に含まれているワードで処理をコマンドから選んで実行する。また。ゲームの難易度調整のための、キャラクタの進行スピード、進行方向変更時に何秒間停止するかなども、フレームレートを計算して制御している。

4.3 スイカ挙動制御

今回、スイカモデルを割るときの動作を自然にするために割れる前のスイカモデルと割れた後のスイカモデルを切り替えている。キャラクタモデルが、スイカを割るコマンドを実行すると、手に持っている棒を振り下ろすアニメーションが実行される。このとき、棒がスイカに付与しているコライダーに触れた時、割れたスイカモデルを 2 つ重ね合わせたものを割れる前のスイカモデルと入れ替える。また、触れた時点でのスイカモデルとキャラクタモデルの位置関係を取得し、その位置情報から、スイカの割れる方向を自然な方向へ制御している。

5. 考察

5.1 結果

本研究の目的である、音声認識技術を AR に組み込むことで、ゲームの世界を現実の世界に近づけるという点はクリアできたのではないかと考えている。今回、音声認識と AR 技術をうまく利用できるモバイルゲームとしてスイカ割りゲームを選んだが、外野からの声でモデルを誘導するというシチュエーションがうまく再現されたと考えている。

5.2 課題

音声認識を行うフェーズで、誤認識は少ないものの、音声をテキストで返すまでのタイムラグがかなり長くなってしまうという問題点がある。このようなタイムラグは、ゲーム性の低下を招きかねない。この遅延は音声認識がサーバサイドで行われるために発生していると考えられるが、通信速度の向上やサーバ配置の広域化などによってこの点を改善できれば、さらに現実の世界に近づいたアプリケーションになることが期待できる。

6. 展望

考えられる展望としては下記のような点が挙げられる。

まず、複数端末で同じゲームを行えるようにすることを考えている。本アプリケーションは Android OS を搭載した端末 1 台でのプレイとなるため、複数人で体験を共有することには不向きである。複数人での体験の共有がさらに現実性を持たせることにつながるのではないかと考えている。

つぎに、VR への移植である。現在の VR 市場では音声認識技術を使用したコンテンツはあまり出回っていない印象を受ける。本研究を通して、音声認識技術は、VR の現実性を高めるための手段としてもう有効であると考えられるため、積極的に、VR 版にも取り組んで行くべきであると思われる。

参考文献

1) Unity(ゲームエンジン) - Wikipedia, https://ja.wikipedia.org/wiki/Unity_(ゲームエンジン),(参照 2017-01-16).
2) Unity でも使える無料 AR ライブラリ Vuforia の基礎知識とライセンス登録, http://www.atmarkit.co.jp/ait/articles/1508/24/news025.html,(参照 2017-01-16).
3) Android SDK とは|アンドロイド SDK|Android Software Development Kit, http://e-words.jp/w/Android_SDK.html,(参照 2017-01-16)
4) IBM の「Watson」を Unity で使ってみた-のしメモアプリ開発ブログ, http://noshipu.hateblo.jp/entry/2016/05/29/003111,(参照 2017-01-16).