Media Science

2018 年度　卒　業　論　文

スマートフォンゲームにおける
圧力変化を使用したタッチパッド UI に関する研究

指導教員：渡辺大地 准教授

メディア学部 ゲームサイエンス プロジェクト
学籍番号　M0115062
衛藤　凌

2019 年 2 月

2018 年度　　卒　業　論　文　概　要

論文題目

スマートフォンゲームにおける
圧力変化を使用したタッチパッド UI に関する研究

メディア学部 学籍番号：M0115062	氏名	衛藤　凌	指導教員	渡辺大地 准教授

キーワード	UI、スマートフォン、圧力、ゲーム操作、ゲームパッド、押し間違い

スマートフォンゲームの中でタッチパッド UI という実際のゲームパッドを模した UI を使用したゲームがある。タッチパッド UI は実際のゲームパッドを模しているため操作を理解しやすいという利点がある。タッチパッド UI のボタンはタッチ操作を使用するのが一般的であるが、タッチ操作でボタンを押す位置がずれて押せなかったり、また押そうとしたボタンの隣のボタンを押したと判断する Fat Finger 問題が発生することがある。タッチの問題に対して圧力を使用した手法は多く存在するが、圧力を使用したもののほとんどが強く押すか弱く押すか、または圧力に応じて効果が変わるといったものである。こういった研究はタッチ操作のエラーの対処としては良いが、強く弱く押すの 2 パターンではゲームのボタンとしては少なく、圧力の変化に応じて効果が変わるものでは実際のゲームパッドにない動作であり、タッチパッド UI の理解しやすい点を阻害する。本研究では圧力を使用してタッチパッド UI の利点を崩さない UI として任意の強さに押し込みつつ、任意方向に指を傾けることでそれに対応したアクションを起こすという方法を用いたタッチパッド UI を提案する。指を押し込むことでアクションを起こすというのは実際のゲームパッドにおけるボタンを押すという動作と似ているためタッチパッド UI の利点を損なわず、また圧力を使用することによってタッチ操作に関する問題の解消になると考えた。評価実験として従来のゲームに使われている一般的なタッチパッド UI を模した UI を制作し、その UI との比較検証を行った。その結果従来の UI のほうが押し間違いが少ないという結果となった。また実験より人によって押し間違いの結果に差があり、人によって得手不得手が大きく使い手が限られる UI ということが分かった。

目 次

図 目 次

第 1 章

はじめに

1.1　研究背景と目的

iPhone や Android といったスマートフォンはその普及率を伸ばし、博報堂 DY メディアパートナーズのメディア環境研究所の調査 [1] では 15 歳から 69 歳のスマートフォン所有率は 8 割近くを示していた。またそのスマートフォンを使用して遊ぶことが出来るアプリケーションゲームは数多く存在する。その中でタッチパッド UI を使用したゲームが存在する。タッチパッド UI とは実在するゲームパッドコントローラーを模し、移動する為のスティック、決定・攻撃などのコマンド操作するためのボタンからなる UI である。タッチパッド UI を使用しているゲームの例として三國無双斬 [2]、FAITH[3] がある。図 1.1 は一般的なタッチパッド UI のイメージである。

図 1.1　一般的なタッチパッド UI のイメージ

タッチパッド UI は実際のゲームパッドを模して作られているため、直感的なインターフェースデザイン [4] の要素である視覚的メタファや操作のメタファといった直感的にイメージしやすい点において優秀である。またスマートフォンゲームはタッチパネルを使用して遊ぶのが一般的であり、タッチパッド UI のボタンに関しても同様である。しかしタッチ操作でボタンを押す場合、押す位置がボタンとずれていて押せていなかったり、またタッチパッド UI の性質上右にボタンが寄りやすく、押そうと思っていたボタンの隣のボタンを押してしまう Fat Finger 問題 [5] が発生しやすい。従来のタッチパッド UI はよく押すボタンをより大きく設定するのが一般的である。これによってよく押すボタンを連打した際などに押し間違えが起きにくくするようになっている。こういったタッチ操作の問題に対して谷ら [6] の研究ではタップ位置を補正することで入力精度の向上を図った。また黒澤ら [7] の研究では一つの指で操作する際のボタンの余白が与える影響について調査した。しかしこれらの研究はタッチ操作をする以上タッチ操作でのミスは避けられない。また指の圧力を使用してこの問題を解消したゲームも存在する。iphone6/6s からの機能である 3D Touch[8] は指の圧力を取得することが可能である。それをゲームに使用した例として Warhammer 40000:Freeblade[9]、Asphalt8[10] がある。Warhammer 40000:Freeblade では

3D Touch を使用し、画面を押す強弱で武器を使い分けることが出来る。Asphalt8 では弱く押すと車をその方向にカーブさせることができ、強く押すことでブーストボタンを使用し速く走ることができる。どちらも 2 つのボタンを使い分けるのに使用され、それらのボタンの変更でいちいち指を離す必要がなく、タップミスをするということがない。しかし従来の 3D Touch を使用した UI では押しが強いか弱いかでの使用しかしておらず、ゲームの機能が増えて、UI が増えることになった場合タッチに関する問題が起きる可能性がある。また圧力を使用した研究として Heo らは、Forcetap[11] において端末に内蔵されている加速度センサの値を利用して押下圧を推定してタップと強タップを使い分ける手法を提案した。同じく Heo らは ForceDrag[12] において強い圧力をかけることによってドラックモードに切り替える手法を提案した。Brewster ら [13] は強く押すことで大文字、弱く押すことで小文字の入力が可能なキーボードを提案した。Yong らは ForceClick[14] において押下圧を用いたクリック手法を提案しており、画面から指を離さずに連続して押すことが可能なボタンを示した。これらの研究では圧力は強いか弱いかしか使用しておらずこれをゲームに使用する場合、3D Touch を使用したゲームと同じようにゲームの機能が増えた場合タッチに関する問題が起きてしまう。また McCallum ら [15] の研究では 3 段階、または 4 段階の押下圧を識別することによって 1 つのキーで 3，4 種類の文字を入力することを可能にした。Zhong ら [16] の研究では指の圧力に応じて移動するカーソルを用いることで 1 つのキーのみでの文字入力が可能なキーボードを提案した。Suzuki ら [17] および Miyaki ら [18] は圧力を使用したズームイン、ズームアウト手法を提案した。これらの研究は圧力を細かく使い分ける必要がある。ゲームで使用する場合、強く押すボタンと弱く押すボタンの間に中間の圧力を押すボタンがあると、そのボタンを押す場合強すぎても弱すぎても押せず、その中間の圧力を保持する必要があるため、素早い操作を求められる場合などにボタンを押すのが難しく、押し間違いの原因となる。またボタン一つで多くの操作を使い分けるというのは実際のゲームパッドにはなく、タッチパッド UI の利点である理解しやすい点を損ねてしまう。

本研究では圧力変化を利用したタッチパッド UI を提案する。タッチパッド UI の理解しやすい点と圧力を使用して押し間違いを抑制するという 2 つの利点を踏まえ、圧力を使用したより実際のゲームパッドに近い UI を目標とした。本手法では指の圧力と指が最初に画面に触れた位置と現在の指の位置を取り、現在の指の位置が画面に最初に触れた位置からどちらに動いたかという情報を取得、そこから指の現在の圧力に応じてアクションを起こす。これをタッチパッド UI の要素であるスティック、ボタンに適用した。ボタンを押すという実際のゲームパッドにおける動作を圧力情報で再現することで、タッチパッド UI の理解しやすい点を阻害しなくなり、またボタンから指を離さないことでタッチに関する問題を解決することが出来ると仮説を立てた。また従来のタッチパッド UI ではタッチ操作の押し間違えの対策としてよく押すボタンを大きくしていたが、本手法をタッチパッド UI に適用する際はタッチによる押し間違えを考慮する必要がない、また一つのボタンを大きく設定した場合ほかのボタンが極端に押しにくくなっていしまう可能性があるので本研究では実際のゲームパッドのボタン配置を模した形とした。図 1.2 は実際のゲームパッドの配置のイメージである。

図 1.2　実際のゲームパッドの配置のイメージ

そこから提案手法と現在のゲームで使用しているような配置の UI を用意し、比較検証を行い、押し間違えの回数などの評価を行った。その結果従来の UI のほうが押し間違いが少ないという結果となった。また評価実験用のゲームの結果では従来の UI のほうが優秀な結果となった。よっ

て本手法は改善の必要があることが分かった。

1.2　本論文の構成

本論文の構成を述べる。2 章では本研究の手法を述べる。3 章では検証内容と結果を述べる。そして 4 章ではまとめを述べる。

第 2 章

提案手法

本章では本研究の提案手法について述べる。第 2.1 では提案手法の概要について述べる。第 2.2 では提案手法を踏まえてスティックを作成、第 2.3 では提案手法を踏まえてボタンを作成する。

2.1 提案手法の概要

2.1.1 UI の表示について

タッチパッド UI は基本的にスマートフォンを横に持ち、左側にスティック、右側にボタンという配置になり、左手の親指でスティック、右手の親指でボタンを操作する。本手法ではそれに従ってそれぞれの指が画面に最初にタッチした位置にそれぞれの UI を表示する。この際扱うデータは以下の 2 つである。

- 指が最初に画面にタッチした位置
- 指の現在の位置

ここでの指が最初にタッチした位置は 1 回のタッチごとの指が最初にタッチした位置であり、指を離すと UI も消え、再度タッチした位置を最初にタッチした位置として UI を表示する。この際原点をデバイスの左下とし、画面は x 軸がデバイスの長辺で右が正方向、y 軸をデバイスの短辺

として上が正方向とする。スティックとボタンの表示方法に関して、指が画面に最初にタッチした指の位置と使用するデバイスの画面の横幅をもとに決定する。この際のデバイスの画面の横幅とはデバイスの長辺の事を指す。指が最初に画面にタッチした位置の x 座標を F_x, 画面の横幅を S_x とした場合、画面半分より左側ならスティックを表示する。その際の式は式 (2.1) となる。画面半分より右側ならボタンの UI を表示する。その際の式は式 (2.2) となる。また指が 2 本感知されている場合まず画面にタッチした順番を取得し、1 番目にタッチした指に関しては上記の方法でどちらの UI を表示するかを決定する。2 番目にタッチした指に関しても同様に決定するのだが、2 番目にタッチした指は 1 番目にタッチした指とは違う UI のほうをタッチしなければならない。例として 1 番目の指で式 (2.1) を満たしていた場合、2 番目の指は式 (2.1) を満たしても UI は表示されず、式 (2.2) を満たした場合のみ UI が表示する。図 2.1 はこの例を図にしたものである。

$$F_x < \frac{S_x}{2} \tag{2.1}$$

$$F_x > \frac{S_x}{2} \tag{2.2}$$

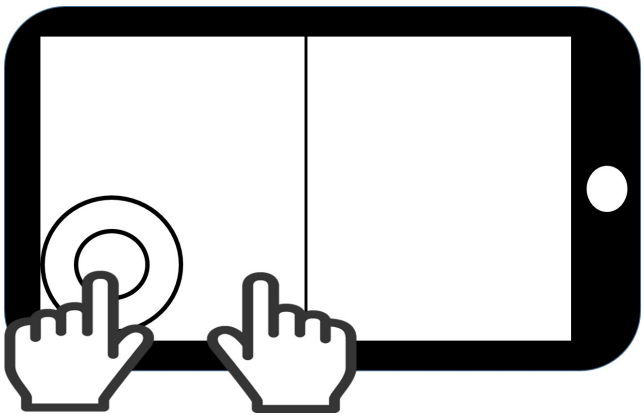

図 2.1　UI を表示しない例

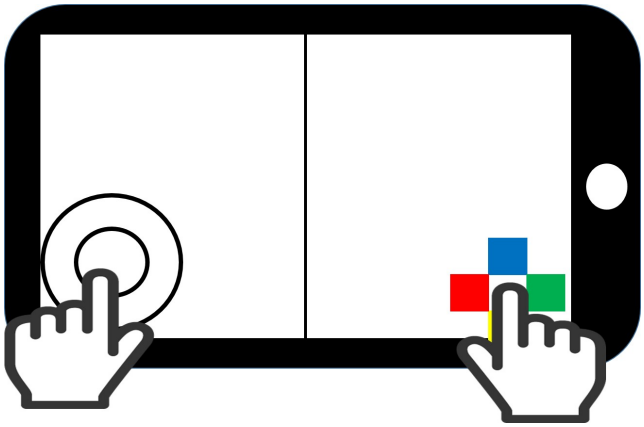

図 2.2　UI を表示する例

2.1.2　提案手法の流れ

提案手法を実装するに際し、以下のデータを取得する。

- 指の圧力
- 指が最初に画面にタッチした位置からの指を動かした方向

指が最初に画面にタッチした位置からの指を動かした方向について、指が最初に画面にタッチした位置を中心に現在の指の位置がどちらに離れているかを計測して取得する。具体的には指が最初に画面にタッチした位置を始点とし、指の現在の位置を終点としたベクトルを取得する。さらにこれを正規化することによって指が最初に画面にタッチした位置から指を動かした方向を取得する。これによって得た指が最初に画面にタッチした位置から指を動かした方向と現在の指の圧力の値を参照し、それに応じたアクションを起こすというのが提案手法の流れである。図 2.3 は提案手法のイメージである。

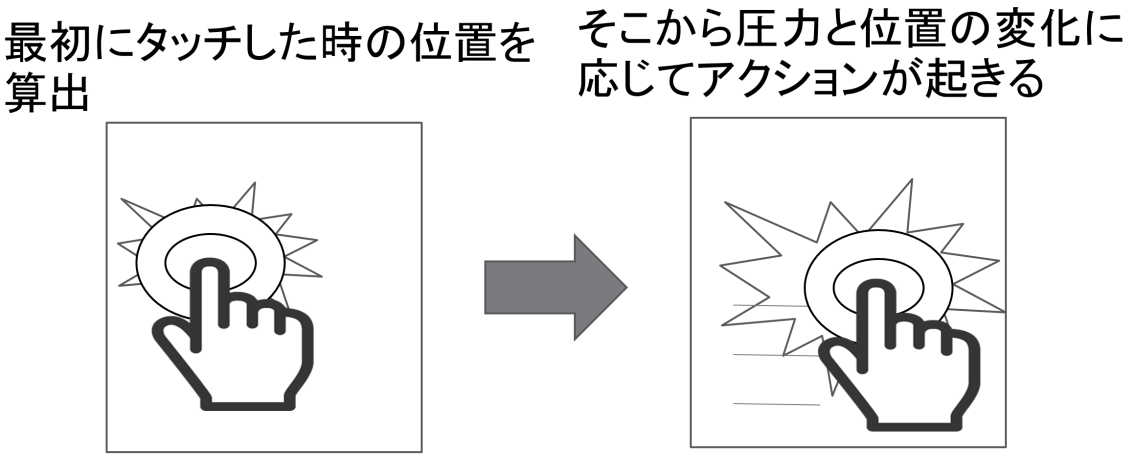

図 2.3　提案手法のイメージ

2.2　スティックの作成

スティックはキャラクターの移動に使用する。キャラクターの移動に関して必要な情報は以下の 2 つである。

- キャラクターの移動方向
- キャラクターの移動速度

これらを提案手法で扱うデータに当てはめる。まずキャラクターの移動方向は指が最初に画面にタッチした位置からの指を動かした方向を使用して決定する。その具体的な方法としては指が最初に画面にタッチした位置からの指を動かした方向のベクトルを取得する。指が画面に最初にタッチした位置を $\mathbf{L}$、現在の指の位置を $\mathbf{M}$、指が最初に画面にタッチした位置からの指を動かした方向の単位ベクトルを $\mathbf{D}$ として式 (2.3) となる。

$$\mathbf{D} = \frac{\mathbf{L} - \mathbf{M}}{|\mathbf{L} - \mathbf{M}|} \tag{2.3}$$

そのベクトルの向きにキャラクターの移動方向を決定する。例として右にキャラクターを移動させたい場合、指が最初に画面にタッチした位置から指を右に動かすことでその方向にキャラクターの動きを指定することが出来る。図 2.4 はキャラクターの移動方向の取得の流れである。

図 2.4　キャラクターの移動の向きの取得

次にキャラクターの移動速度は現在の指の圧力を使用して決定する。具体的な方法としてはキャラクターを速く動かしたい場合は指の圧力を強くし、遅く移動させたい場合は指の圧力を弱くすることでその操作ができる。圧力に関してデバイスによって取得可能な圧力の最大値が変わる。本研究で使用した環境では「０」～「6.666667」までの圧力の値を取得することが可能だった。ゲームで使用する場合圧力の値に任意の変数をかけ調整する。

以上の情報をもとにキャラクターの移動ベクトルを $\mathbf{P}$、現在の指の圧力を f、ゲームで使用する場合の圧力の値の調整用の変数を a とした場合、キャラクターの移動の式は (2.4) となる。

$$\mathbf{P} = af\mathbf{D} \tag{2.4}$$

2.3　ボタンの作成

ボタンは決定、攻撃など各ボタンを押すことによってそれらのボタンに割り振られた行動を行う。ボタンにおいて必要な情報は以下の２つである。

- どのボタンを選択しているか

- ボタンを押しているかどうか

これらを提案手法で扱うデータに当てはめる。まずどのボタンを選択しているかは指が最初に画面にタッチした位置からの指を動かした方向を使用して決定する。またどのボタンを選択しているかというのは提案手法では画面から指を離さない関係上、タッチした状態でそれぞれのボタンの識別を行う。識別の方法としては上記に記したように、指が最初に画面にタッチした位置からの指を動かした方向を使用し、任意の方向に指を動かすとそのボタンを選択したという判定になる。選択できるボタンの数に関しては 4 種類としている。具体的なボタンの判定の分け方として実際のゲームパッドを模して上下左右でボタンを配置し、それぞれの判別として指が最初に画面にタッチした位置の x 座標を R_x、指が最初に画面にタッチした位置の y 座標を R_y、指の現在の位置の x 座標を r_x、y 座標を r_y とした場合、左は式 (2.5)、右は式 (2.6)、上は式 (2.7)、下は式 (2.8) の場合そのボタンを選択する。また図 2.5 はボタンの判別を画像で表したものである。中心を指が最初に画面にタッチした位置として、それぞれの色の範囲の中に現在の指の位置が入った場合それぞれに対応したボタンを選択する。

$$r_y - R_y > r_x - R_x \cap r_y - R_y < -(r_x - R_x) \tag{2.5}$$

$$r_y - R_y < r_x - R_x \cap r_y - R_y > -(r_x - R_x) \tag{2.6}$$

$$r_y - R_y > r_x - R_x \cap r_y - R_y > -(r_x - R_x) \tag{2.7}$$

$$r_y - R_y < r_x - R_x \cap r_y - R_y < -(r_x - R_x) \tag{2.8}$$

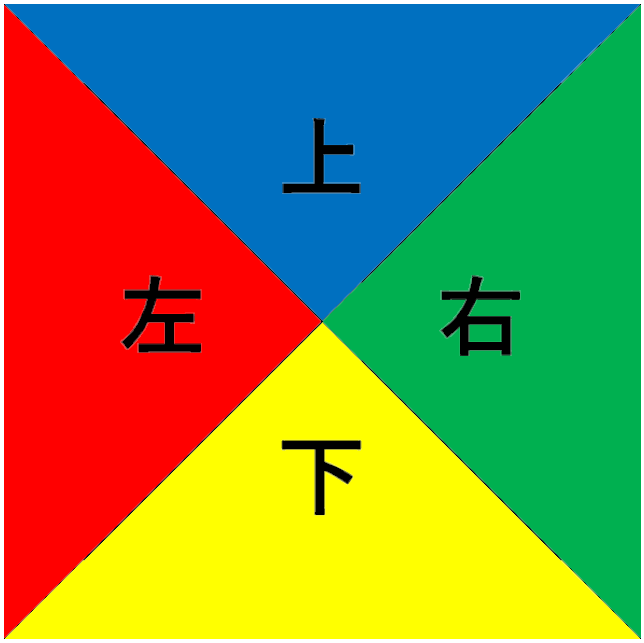

図 2.5　ボタンの選択

次にボタンを押しているかどうかは指の現在の圧力によって決定する。ボタンを押しているかどうかのしきい値として、デバイスでとれる圧力の最大値の半分より強く押している場合ボタンを押したと認識する。デバイスでとれる圧力の最大値を f_{max} とすると式 (2.9) を満たした場合に押していると認識する。本研究で行った環境だと、しきい値は「3.333333」となった。

$$f > \frac{f_{max}}{2} \tag{2.9}$$

第 3 章

評価と分析

本章では評価実験の方法と実行結果、また結果の考察を行う。第 3.1 では評価実験の方法と実行結果を行い、第 3.3 では実験の考察を行う。

3.1　評価方法

本手法の有用性を検証するために本研究では被験者に本手法の UI と一般的なタッチパッド UI を参考に作成した従来の UI を使用した後述の評価実験用のゲームをプレイしてもらった。被験者にはまず片方の UI についてと評価実験用のゲームの説明とその UI の練習を 5 分程度行った後、評価実験用のゲームをプレイしてもらい、敵の色と違う色を押した回数、ボタンから赤からその他に変わった時の指定のボタンを押すまでの秒数、ゲームクリアまでの時間を計測した。またその後同じように、もう片方の UI についての説明とその UI の練習を 5 分程度行った後、同じゲームをプレイしてもらい、敵の色と違う色を押した回数、ボタンから赤からその他に変わった時の指定のボタンを押すまでの秒数、ゲームクリアまでの時間を計測した。また提案手法のみ画面から指を離した回数、従来の UI のみ UI 以外の場所をタッチ回数を計測した。

3.1.1　実装した環境について

実装についてはスマートフォンに関しては iPhone8 を使用した。図 3.1 は実際に使用した実機である。また開発ツールに関しては Unity[19] を使用した。

図 3.1　使用した iPhone8

3.1.2　評価実験用のゲームについて

評価に使用したゲームについて述べる。評価実験用のゲームの内容はボタンを押すことでそれに対応した色の弾が出る。それを敵に当てて倒すというものである。敵は hp という値を持ち、初期値は 30 でプレイヤーがボタンを押して出た弾に当たった場合その値を-1 する。hp が 0 になった時点で敵を倒したと判定する。また敵の色に応じてあたる弾が変わり、同じ色の弾しか当たらないようになっている。敵の色は hp によって変わる。表 3.1 は hp による色の変化を表にしたも

のである。

表 3.1　hp に応じた色の変化

hp	敵の色
30〜21	赤
20	緑
19〜11	赤
10	青
9〜0	赤

こういった色の変化にした理由として、タッチミスの要素としてボタンを連打している場面と、その他のボタンを押す場面という 2 つの状況を作るという意味合いでこのような色の変化とした。

またプレイヤーの出す弾はボタンによって異なる。提案手法の UI では実際のゲームパッド UI で主に攻撃ボタンとして使用されている左のボタンをもっとも使う赤とし、それ以外は上のボタンを青、右のボタンを緑、下のボタンを黄と割り振った。従来の UI では赤を中心に置き、青をその上、緑を左上、黄を左下に配置した。図 3.2、図 3.3 はそれぞれ提案手法と従来の手法のボタンの配置図である。

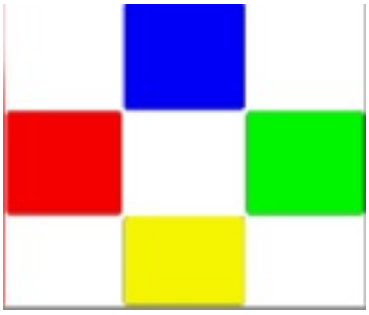

図 3.2　提案手法のボタンの配置

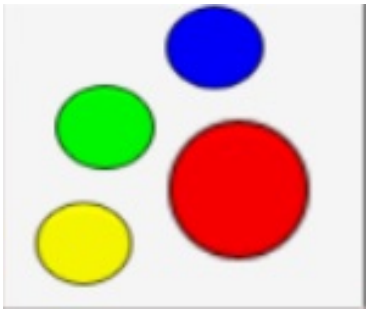

図 3.3　従来のボタンの配置

また図 3.4 と図 3.5 はそれぞれ提案手法と従来の UI での評価実験の実際の様子である。

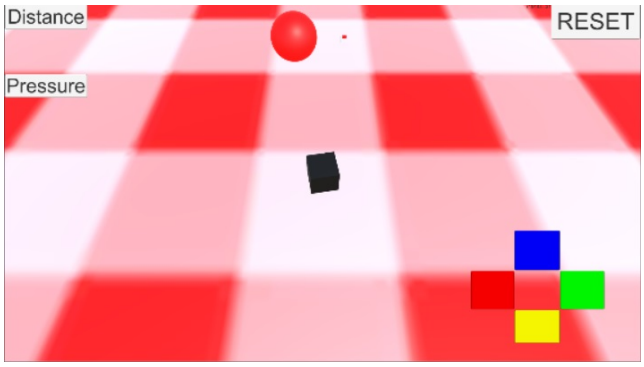

図 3.4　提案手法の UI の様子

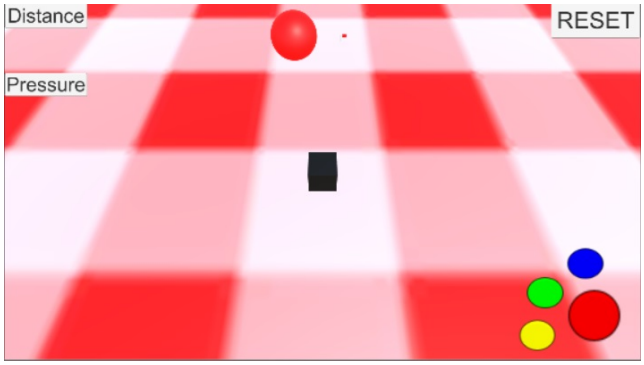

図 3.5　従来の UI の様子

3.2　実験結果

実験結果を示す。被験者としては 20 代の男性 8 名に対して検証を行い、以下のデータを表にまとめる。

- 被験者番号
- 敵の色と違う色を押した回数
- 提案手法のみ画面から指を離した回数、
- 従来の UI のみ UI 以外の場所をタッチした回数
- ボタンが赤からその他に変わった時の指定のボタンを押すまでの秒数
- 敵を倒すまでの時間

この際順序効果をなくすため、被験者番号に応じて使用する UI の順番を変え、前半 1〜4 までが本手法から、後半 5〜8 までが従来の手法からという形にした。

3.2.1　押し間違えについて

表 3.2 は本手法と従来の UI での間違えて違うボタンを押した回数と本手法の画面から指を離した回数、従来の UI でのボタン以外の部分を押した回数である。

表 3.2　押し間違えの回数

被験者番号	本 UI ボタン	本 UI 離した	従来の UI ボタン	従来の UI ボタン以外
1	1	0	1	6
2	0	0	2	4
3	6	3	0	0
4	10	4	0	3
5	0	0	0	0
6	2	0	0	4
7	1	9	0	0
8	1	2	0	9
平均	2.65	2.25	0.38	3.25

また表 3.3 は押し間違えの合計として、表 3.2 の本手法の間違えて違うボタンを押した回数と画面から指を離した回数を合計したものと従来の UI の間違えて違うボタンを押した回数とボタン以外の部分を押した回数を合計したものである。結果より従来の UI のほうが押し間違えが少ないという結果となった。

表 3.3　押し間違えの合計

被験者番号	本 UI	従来の UI
1	1	7
2	0	6
3	9	0
4	14	3
5	0	0
6	2	4
7	10	0
8	3	9
平均	4.88	3.63

表 3.3 の結果より、従来の UI のほうが本手法の UI よりも押し間違いが少ないという結果となった。

3.2.2　敵の色が赤から切り替わった場合のボタンを押すまでの時間

表 3.4 は本手法と従来の UI での敵が緑になってから緑のボタンを押すまでの時間で表 3.5 が敵が青になってから青のボタンを押すまでの時間を示したものである。また敵が緑になってから緑のボタンを押すまでの時間 (秒) を green(秒)、敵が青になってから青のボタンを押すまでの時間 (秒) を blue(秒) として表示している。

表 3.4　敵の色が赤から切り替わった場合の緑のボタンを押すまでの時間

被験者番号	本 UI green(秒)	従来の UI green(秒)
1	1.13	1.33
2	1.38	1.15
3	0.94	1.39
4	1.40	1.26
5	1.79	1.49
6	1.79	1.33
7	1.75	1.81
8	1.50	1.41
平均	1.34	1.27

表 3.5　敵の色が赤から切り替わった場合の青のボタンを押すまでの時間

被験者番号	本 UI blue(秒)	従来の UI blue(秒)
1	2.51	2.13
2	1.11	1.68
3	2.96	1.64
4	1.71	1.86
5	1.25	1.68
6	0.99	1.67
7	2.48	1.56
8	1.96	1.92
平均	1.87	1.77

表 3.4 と表 3.5 より本手法の UI と従来の UI で敵の色が赤から切り替わった場合のボタンを押すまでの時間に大きな差はないという結果となった。

3.2.3　敵を倒すまでの時間

表 3.6 は本手法と従来の UI での敵の hp を 0 にするまでの時間である。それぞれの敵を倒すまでの時間 (秒) を表示した。

表 3.6　敵を倒すまでの時間

被験者番号	本 UI 時間 (秒)	従来の UI 時間 (秒)
1	22.17	20.05
2	23.71	25.00
3	20.83	21.78
4	22.09	20.21
5	29.02	22.75
6	34.69	23.83
7	29.39	23.71
8	36.02	20.48
平均	27.24	22.23

この表 3.6 より従来の UI のほうが、本手法より早く敵を倒すことが出来るという結果となった。

3.3　考察

押し間違えの回数と敵を倒すまでの時間から読み取れる結果として、押し間違えの回数は従来手法のほうが少ないという結果となった。また本手法での押し間違えは被験者によって、押し間違えの量に大きな差があることが分かった。敵を倒すまでの時間は、従来の UI のほうが早いという結果となった。しかし被験者によっては、本手法のほうが早く敵を倒すことができた人もいた。これらの事から本手法は人による得手不得手が大きく、使い手が限られる UI だと考察できる。

第 4 章

まとめ

本研究では、圧力変化を利用したタッチパッド UI を提案し、Unity を使用して本手法のタッチパッド UI と検証用のゲームを作成し、従来の UI との比較を行った。検証結果として、従来の UI のほうが押し間違いが少ないという結果となった。また被験者によって本手法の押し間違いの回数が大きく変わった。また従来の UI のほうが評価実験用のゲームにおいては優秀な結果となった。しかし評価実験用のゲームにおいて本手法のほうが優秀な結果の場合も存在した。

今後の展望として本研究で行った実験から本手法の問題点を挙げる。まずボタンを押す際やボタンを押している状態から押していない状態にするときにしきい値を超えず、その判定が継続する問題がある。これは人によって押しやすい圧力のしきい値が違うことが原因だと推測する。よって圧力のしきい値を動的にし、使用者によって押しやすいしきい値に変更できるようにすることが望ましい。またボタンを押している判定から押していない判定にしようとしたときに誤ってボタンから指を離す問題がある。この問題は被験者によって発生するかどうかが大きく分かれた。よってこの問題が起こす被験者と起こさない被験者によって、どのような傾向があるかを調査する必要がある。押し間違いにおいて本手法のほうでは指を離した回数についての結果も合計したがこの押し間違いは圧力に慣れていないため起こったものだと考えられ、本手法の操作に慣れた場合、この押し間違いの結果は減少すると考察する。また本手法において実験で行ったゲー

ムは 1 種類のみであったため、今回のゲームにない操作をした場合だと有意な結果が出る可能性がある。また今回はスティックに関しては評価をしていない。スティックに関しての検証も必要である。上記を踏まえてより多くの人が使いやすい UI とするべく手法を改善する必要がある。

なお本研究は、芸術科学会 NICOGRAPH2018 における”スマートフォンゲームにおける圧力変化を使用したタッチパッド UI に関する研究”[20] として発表した内容を含む。

謝辞

本研究を行うにあたり、ご指導いただいた渡辺先生と阿部先生に心より感謝いたします。また研究室の皆様、研究に協力してくださった方々にも深く感謝いたします。

参考文献

[1] ガベージニュース. http://www.garbagenews.net/archives/2170355.html. 参照:2018.12.21.

[2] 三國無双斬公式サイト. https://mobile.nexon.co.jp/musouzan. 参照:2018.12.22.

[3] Faith 公式サイト. http://mobile.nexon.co.jp/faith. 参照:2018.12.22.

[4] 井上勝雄, 広川美津雄, 岩城達也, 加島智子. 直感的なインターフェースデザインの 10 原則の提案. 日本デザイン学会第 62 回研究発表大会, Vol. 62, pp. 238–239, 2015.

[5] Katie A. Siek, Yvonne Rogers, and Kay H.Connelly. Fat finger worries: how older and younger users physically interact with pdas. *In Proceedings of the 2005 IFIP TC13 international conference on Human-Computer Interaction*, pp. 267–280, 2008.

[6] 谷尭尚, 山田誠二. タッチパネルにおける UI デザインを考慮した操作特性モデルを用いたタップ座標補正. HAI シンポジウム 2014, Vol. 6, pp. 98–103, 2014.

[7] 黒澤敏文, 久野祐輝, 小森谷大介, 志築文太郎, 田中二郎. タッチ UI におけるボタンの余白の大きさが操作に与える影響. 研究報告ヒューマンコンピュータインタラクション, Vol. 16, pp. 1–7, 2014.

[8] 3D Touch - iOS - Apple Developer. https://developer.apple.com/ios/3d-touch/. 参照:2018.12.19.

[9] PIXELTOYS GAMES. https://www.pixeltoys.com/games/. 参照:2018.12.22.

[10] asphalt8. http://www.gameloft.com/asphalt8/?lang=jp. 参照:2018.12.24.

[11] Seongkook Heo and Geehyuk Lee. Forcetap:Extending the Input Vocabulary of Mobile Touch Screens by Adding Tap Gestures. *In Proceedings of the 13th International Conference on Human Computer Interaction with Mobile Devices and Services*, pp. 113–122, 2011.

[12] Seongkook Heo and Geehyuk Lee. ForceDrag:using pressure as a touch input modifier. *Proceedings of the 24th Australian Computer-Human Interaction Conference*, pp. 489–493, 2017.

[13] Stephen A. Brewster and Michael Hughes. Pressurebased Text Entry for Mobile Devices. *In Proceedings of the 11th International Conference on HumanComputer Interaction with Mobile Devices and Services*, pp. 9:1–9:4, 2009.

[14] Sangeon Yong, Edward Jangwon Lee, Roshan Peiris, Liwel Chan, and Juhan Nam. ForceClicks:Enabling Efficient Button Interaction with Single Finger Touch. *In Proceedings of the Eleventh International Conference on Tangible*, pp. 489–493, 2017.

[15] David C. McCallum, Edward Mak, Pourang Irani, and Sriram Subramanian. Pressure Text:Pressure Input for Mobile Phone Text Entry. *In Proceedings of the 2009 CHI Conference Extended Abstracts on Human Factors in Computing Systems*, pp. 4519–4524, 2009.

[16] Mingyuan Zhong, Chun Yu, Qian Wang, Xuhai Xu, and Yuanchun Shi. ForceBoard:Subtle Text Entry Leveraging Pressure. *In Proceedings of the 36th Annual ACM Conference on Human Factors in Computing Systems*, pp. 528:1–528:10, 2018.

[17] Kenji Suzuki, Ryuuki Sakamoto, Daisuke Sakamoto, and Tetsuo Ono. Pressure-sensitive

Zooming-out Interfaces for One-handed Mobile Interaction. *In Proceedings of the 20th International Conference on HumanComputer Interaction with Mobile Devices and Services*, pp. 30:1–30:8, 2018.

[18] Takashi Miyaki and Jun Rekimoto. GraspZoom: Zooming and Scrolling Control Model for Single-handed Mobile Interaction. *In Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*, pp. 11:1–11:4, 2009.

[19] Unity official site. `https://unity3d.com/jp`. 参照:2018.12.17.

[20] 衛藤凌, 阿部雅樹, 渡辺大地. スマートフォンゲームにおける圧力変化を使用したタッチパッド UI に対する研究. 芸術科学会 NICOGRAPH, 2018.