筑波大学大学院博士課程

システム情報工学研究科修士論文

画面ループを利用した
モバイル端末片手操作手法

土佐 伸一郎

修士（工学）

（コンピュータサイエンス専攻）

指導教員　田中 二郎

2013 年 3 月

概要

本論文では，モバイル端末片手把持時において GUI を円滑に操作可能にする端末操作手法について述べる．

画面操作領域の大きなモバイル端末を片手操作する際には，親指にて触れにくい領域が存在してしまい，その領域の GUI に触れる事が非常に困難になる．

この問題の原因は，現在のモバイル端末のタッチインタフェースが画面上のオブジェクトに直接触れて操作するという指直接操作の操作体系を取っている一方，片手操作時には親指可動領域外に GUI が配置されてしまうためであると我々は考える．そこで我々は現行の指直接操作の操作体系を残したまま，親指可動領域外の GUI を操作可能にするインタラクション手法の開発を行う．モバイル端末の画面全体をループするように移動させることにより，親指可動領域外の GUI を可動領域内に移動させるというアプローチを取る．これにより，片手によるシングルタッチ操作環境において，様々な GUI を円滑に指直接操作可能である．また，提案手法はハードウェアとソフトウェア両方からのアプローチにより操作と結果が自然に対応付いた直観的なインタラクションを実現している．

本手法を適用した実験用アプリケーションをユーザに利用してもらう被験者実験により，本研究で開発した手法が親指可動領域外の GUI を円滑に操作可能にすること，そして本手法がユーザが端末操作を行う上で直感的な手法である事を確認した．

目 次

第 1 章　序論 **1**
1.1　モバイル端末の操作体系 1
1.2　片手操作の重要性 1
1.3　画面操作領域の増大による問題点 2
1.4　本研究の目的 2
1.5　本研究のアプローチ 2
1.6　本研究の貢献 3
1.7　本研究の構成 3

第 2 章　関連研究と本研究の位置づけ **4**
2.1　ソフトウェアによる GUI 操作支援 4
2.2　ハードウェアによる GUI 操作支援 6
2.3　本研究の位置づけ 6

第 3 章　画面ループを利用したモバイル端末片手操作手法 **8**
3.1　片手操作と現行のモバイル端末の GUI 8
3.2　指直接操作を行うためのアイデア 8
3.3　画面ループによる指直接操作 10
3.4　画面ループを生じさせる端末操作 10
3.5　ベルトメタファを用いた端末操作と画面効果の自然な対応 11
3.6　本手法の適用範囲 11
3.7　現行のモバイル端末操作手法との共存 12
3.7.1　応用利用 13

第 4 章　デバイス設計と実装 **15**
4.1　デバイスの設計方針 15
4.2　プロトタイプデバイスの実装 15

第 5 章　ソフトウェアの設計と実装 **18**
5.1　実装環境 18
5.2　ロール操作 18
5.3　画面移動 21
5.3.1　ロール操作以外の画面移動を生じさせる操作 21

5.3.2 ループビュー 22
5.4 Bluetooth による端末間通信 22
5.5 予備実験による実装改善 24
GUI 操作後の画面移動の挙動 24
画面を初期位置に戻す操作方法の変更 25
左右方向の画面ループ 26
5.6 アプリケーションの実装 27

第 6 章 実験 **30**
6.1 実験目的 30
6.2 実験内容 30
6.3 実験結果と考察 33
6.3.1 タスク全体に関して 33
手への負担度 33
GUI の操作性 34
インタラクションの自然さ 35
6.3.2 レイアウトごとに関して 35
6.3.3 1 試行あたりの実行時間 36

第 7 章 今後の課題と発展 **39**
7.1 今後の課題 39
7.2 発展 40

第 8 章 結論 **41**

謝辞 **42**

参考文献 **43**

図 目 次

2.1 特殊なカーソルを用いる手法 5
2.2 縮小スクリーンショットを選択する手法 5

3.1 画面ループ 9
3.2 画面ループによる指直接操作 10
3.3 ロール操作 11
3.4 ベルトメタファの概念図 12
3.5 利用例：Web ブラウザアプリケーションへの適用 13
3.6 左:タブレット端末への応用，右:画面のずれを利用した画面拡張 14

4.1 デバイスの理想イメージ 16
4.2 実装したプロトタイプデバイス 17
4.3 左：プロトタイプデバイスの前面，右：プロトタイプデバイスの背面 17

5.1 背面接触指 19
5.2 無効なロール操作 20
5.3 背面指によるダブルタップ 21
5.4 ループビュー 23
5.5 予備実験時の様子 24
5.6 予備実験用のアプリケーションの画面レイアウト 25
5.7 両面の指を用いたダブルタップ 26
5.8 画面移動受付状態時の上下左右方向への画面移動 28
5.9 ブラウザアプリケーション 29
5.10 画面拡張アプリケーションランチャー 29

6.1 実験用アプリケーションのレイアウト．(a)B レイアウト，(b)CB レイアウト，(c)SB レイアウト，(d)DD レイアウト 31
6.2 プロトタイプ下部のシステムキー 32
6.3 実験の様子 34
6.4 GUI 操作性と手への負担度に関する 5 段階リッカート尺度評価. 左上：タスク全体に関して，右上：B に関して，左下：CB に関して，下中央：SB に関して，右下：DD に関して 35
6.5 ユーザごとのドラッグアンドドロップ試行回数の平均値 36

6.6　レイアウトごとの 1 試行あたりの時間 (msec) 38

第1章　序論

本章では，モバイル端末における片手操作の重要性について述べ，その際に発生する問題点について述べる．更に本研究の目的と目的達成のためのアプローチを述べ，本研究の貢献を述べる．

1.1　モバイル端末の操作体系

フルタッチパネルを搭載したモバイル端末の普及が進んでいる [1]．このような現在のモバイル端末のタッチインタフェースは，画面上に表示されるオブジェクトに指にて直接触れることで端末の操作を行っている (以降，指直接操作)．モバイル端末の操作方法は大きく分けて片手操作と両手操作に分類可能である．本稿で述べる片手操作と両手操作の定義は以下である．

片手操作 片方の手にてモバイル端末を把持し，その手の操作指 (親指) で操作

両手操作 片方の手にてモバイル端末を把持し，もう片方の手の指で操作，もしくは，両手でモバイル端末を把持し両方の手の指で端末を操作

1.2　片手操作の重要性

両手操作ではモバイル端末画面の好きな位置に自由に触れることや，複数の指を利用したマルチタッチ操作を行うことが可能であり，一般的には片手操作よりも快適なモバイル端末操作が可能である．しかし，実際モバイル端末を利用すると両手では操作が出来ない状況が多々存在する．例えば，立ったまま電車で通勤するという状況を考える．このような状況では，片手に鞄を抱えていて，片手のみしか思うように動かせないということがある．この時にもし片手操作が可能ならば，電車内にてモバイル端末上で仕事のメールやドキュメントを確認することが出来る．このように片手操作というのは日常生活では自然に利用されており，モバイル端末を利用する上では欠かすことの出来ない操作体系と言える．さらに Karlson らの調査 [2] では，大多数のユーザが「インタフェースが対応していれば，日常的に片手操作を行いたい」と回答したとされている．このような需要のために，マルチタッチ操作により頻繁に利用される操作方法は，シングルタッチ操作を用いて代替出来るように設計されていることが多い．マルチタッチ操作は 2 本以上の指を同時に利用するので，基本的には両手操作が必要とされる操作方法である．例えば，2 本指にてピンチ操作を行うと画面内容が拡大縮小

するという操作がある．これは一般的には片手で端末を把持して，もう片方の手の 2 本の操作指にてピンチ操作する，あるいは，両手にて把持しそれぞれの手の指を 1 本づつ用いてピンチ操作する必要がある．しかし，こういった操作はシングルタッチのダブルタップという操作に割り当てる事や画面下端に拡大縮小のソフトウェアボタンを表示させる事で代替手段とする事が一般的となっている．そのためゲーム等の一部のアプリケーションを除いて，マルチタッチ操作なしに，片手でのシングルタッチ操作にて十分に利用可能であるアプリケーションは非常に多い．

1.3 画面操作領域の増大による問題点

一方，画面操作領域の大きなモバイル端末の需要が大きくなっており，今後さらに増えるとされている [3]．これにより画面上の情報量が増えたり，画面上のコンテンツを大きく表示できるといった利点がある．しかし，このような画面操作領域の大きなモバイル端末での片手操作を行う際には，ある問題が発生する．それは画面操作領域が大きいために親指で触れにくい領域が存在し，その領域の GUI に触れることが非常に困難になるということである．このような時は親指の長さが物理的に足りないために，両手操作を強いられたり無理な持ち替えを行うことになる．

1.4 本研究の目的

上記の問題の原因は，現在のモバイル端末のタッチインタフェースが画面上のオブジェクトに直接触れて操作するという指直接操作の操作体系を取っている一方，片手操作時には親指可動領域外に GUI が配置されてしまうためであると我々は考える．そこで我々は現行の指直接操作の操作体系を残したまま，親指可動領域外の GUI を円滑に操作可能にするインタラクション手法の開発を目的とする．

1.5 本研究のアプローチ

モバイル端末の画面全体をループするように移動させることにより，親指可動領域外の GUI を可動領域内に移動させるというアプローチを取る．これにより，例えば画面上部に配置されているボタン等の GUI を少ない移動量で画面下部へと移動させることができ，親指で容易に指直接操作することが可能となる．

また，画面全体を移動させる操作としてモバイル端末の両面から親指と人差し指を用いて端末を回転させるように動かす操作であるロール操作を提案，実装する．ベルトコンベアを回転させるようなイメージで操作でき，操作と結果が自然に対応付いた直観的なインタラクションを実現している．さらに，上記のような両面操作を可能とするデバイスを実装する．このデバイス上で本提案手法を適用させたアプリケーションを実装し，本手法の有用性を検証するための評価実験を行う．

1.6 本研究の貢献

親指可動領域外の GUI を操作可能にする研究は存在するが，それらはポインタを用いるといったような間接操作にとどまっていたり，タップ操作のみしか考慮していない限定的な指直接操作であった．本研究では，現行のタッチインタフェースで採用されている指直接操作の操作体系をそのまま利用できるので，親指可動領域外に配置された様々な種類の GUI が操作可能となる．これを実現した点が本研究の第 1 の貢献である．

また，本研究では画面全体がループするという特殊なインタフェースを用いている．その画面ループを生じさせる端末操作としてロール操作を提案，実装した．従来研究ではソフトウェアとハードウェアそれぞれが個々に片手操作を支援する研究が成されていた．本研究では，ソフトウェア側のインタフェースの工夫とハードウェアへの操作という両方からの支援を上手く組み合わせることにより，操作と結果が結びついた直感的なインタラクションを可能とした．これを実現した点が本研究の第 2 の貢献である．

1.7 本研究の構成

本論文は，まず始めに第 2 章にて関連研究の分析を行い，本研究の位置づけについて述べる．第 3 章では親指可動領域外の GUI を操作可能にする，画面ループを利用したモバイル端末片手操作手法について述べる．第 4 章では提案インタラクションを行うのに適切なデバイスの設計，およびび実装について述べる．第 5 章では，第 4 章にて実装を行ったプロトタイプデバイス上で動作するソフトウェアの設計，および実装について述べる．第 6 章では先に述べた手法の有用性についての評価実験とその考察を行う．第 7 章では今後の課題と発展について述べる．そして最後に第 8 章にて結論を述べる．

第2章　関連研究と本研究の位置づけ

本章では，本研究に関連する従来研究を述べる．モバイル端末の片手操作時における GUI 操作を支援する研究について，ソフトウェア側から，そしてハードウェア側から支援する研究について述べる．それら研究と本研究との関連性について議論し，本研究との位置づけを述べる．

2.1　ソフトウェアによる GUI 操作支援

GUI 配置方法を工夫するもの

GUI の配置の仕方を工夫することにより片手操作を支援するものが存在する．Karlson らの AppLens and LaunchTile[4] ではタイル状の GUI を用いる事により，スマートフォンや PDA といった小型端末での片手操作を可能にした．Smoozy[5]，iLunascape[6]，jigtwi[7] はモバイル端末向けのアプリケーションであり，画面下部に頻繁に利用されるボタンを配置しておく事により，片手操作による円滑なアプリケーション操作を可能にしている．親指が容易に動かせる位置である画面右下部，左下部に扇状にメニューを配置するものもある [8][9].

ここまでに述べた従来研究，サービスにおいてはどれも新たな GUI の配置方法を提案している．しかし，現状のモバイル端末の GUI 全てをこれらの提案 GUI のように片手操作向けに変更する事は事実上不可能である．そのため，GUI の配置方法を改善するだけでなく，片手操作向けに設計されていないアプリケーションであっても，片手操作可能にするような操作手法の実現が必要であると考える．

新たな入力手法を提案するもの

GUI の配置方法を工夫するのではなく，新たな操作手法により現行の GUI をより円滑に操作出来るようにする研究もなされている [10]．その中でも片手操作の円滑化を主目的とした研究について述べる．Karlson らの ThumbSpace[11] では，画面のスクリーンキャプチャを縮小したものを親指可動領域内に表示し，その縮小スクリーンキャプチャ上で目的の GUI が表示されている部分にタップし，大まかに選択エリアを指定する．その後は，親指の上下左右のスワイプ操作により選択する GUI のフォーカスをずらして目的の GUI を選択できる．

Roudaut らの MagStick[12] では，指のスワイプと逆方向に移動する特殊なカーソルを用いる事で，親指可動領域外の GUI を選択する事が可能となる．

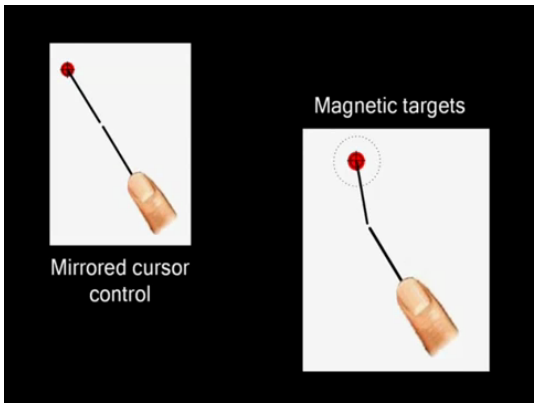

図 2.1: 特殊なカーソルを用いる手法

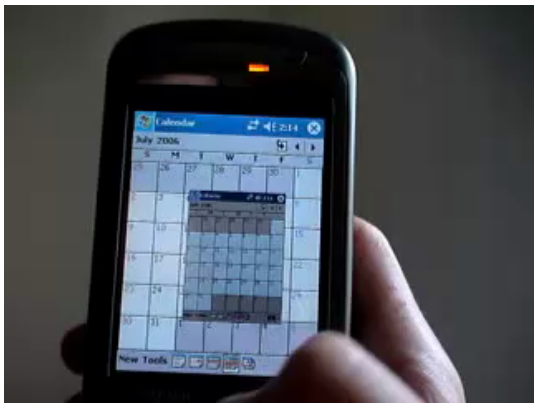

図 2.2: 縮小スクリーンショットを選択する手法

Kim ら [13] はモバイル端末のベゼル領域から親指をスライドさせる，あるいは親指の設置面積が大きくなるように画面にタップしてから親指をスライドさせることにより，画面全体を動かし可動領域外の GUI を親指へと近付けるという手法を取っている．

ここまでに紹介した研究は，片手操作時における GUI 操作の支援を目的としている．しかし，Karlson ら，Kim らの手法においてはカーソルによるポインティングや，GUI へのフォーカスの変更などを用いた，間接的な操作による GUI の選択にとどまっている．現行の Andorid や iOS における GUI の操作は，操作指そのものをポインティングデバイスとし，画面の GUI に直接触れて操作を行う指直接操作という体系をとっている．そのために，画面に別のカーソルを表示し GUI を選択するといったような，指位置と GUI の位置が離れた間接的な操作手法を用いるのは非常に不自然であると考える．本手法では画面上のあらゆる GUI に対して，親指による指直接操作を行える．またこれら従来研究と異なり，GUI の選択だけでなく，タップやスワイプ等の既存のシングルタッチによる操作手法を行う事が可能なので，親指可動領域外の GUI に対してよりリッチな操作を行う事が可能となる．

Kim らの手法では画面全体を動かし，可動領域外の GUI を親指へと近付けている．この手法においては本研究と同様に GUI 自体を動かすというアプローチを取っており，タップなどの指直接操作が可能となっている．しかし，この手法では画面全体を動かすと，画面のベゼルを超えて移動した部分は画面からはみ出してしまい表示されない．つまりそのはみ出した部分に対して操作を行う事は出来ない．そのため，例えば画面の最上部から最下部へとオブジェクトをスワイプで移動させたい場合などには対応することが出来ない．本研究では画面全体を動かす際に画面の上下左右をループさせるという方法を用いている．それにより，あらゆる状況において変わらない操作感で指直接操作を行う事が可能である．

2.2 ハードウェアによるGUI操作支援

本研究のようにモバイル端末に入力装置を付加することによって，モバイル端末操作を支援する研究は数多く存在する [14][15][16][17]．その中でも片手把持時におけるGUI操作を支援しているものについて説明する．

河内谷ら [18] はモバイル端末にスティック上のアナログ入力機構を取り付け，その入力機構を動かす事で画面のGUIへのポインティング可能にした．渡部ら [19] は端末に内蔵されているカメラの上に弾性の突起物を取り付け、突起物を指で操作することによってGUIのポインティングや押し込む操作を実現した．Yuら [20] はタッチパネルを搭載したモバイル端末のベゼル領域にクリップ型の入力装置を装着することによって、タッチパネルの操作拡張を行いGUIへのタッチ操作の代替とする手法を示した．

これらの従来研究では，どれも平面的な実装が不可能な機構を取り付けている．モバイル端末は通常，把持が行いやすいように裏表ともに凹凸の無い形状をしている．これらの特殊な外部機構を取り付ける事によりモバイル端末の把持を困難にしたり，端末の収納の際に不便が生じたりと，日常利用を考えると望ましくない．またYuらの手法のように外部機構が容易に取り外し可能であっても，それらを装着したり常に持ち歩くのは負担である．

背面にタッチセンサを利用するもの

本研究では提案するインタラクションを実現するための外部機構として背面にタッチセンサを取り付けるという方法を採用した．タッチセンサは平面的な実装が可能であり，商品化されているものもある [21][22]．背面にタッチセンサを搭載したモバイル端末への入力手法については近年研究対象として注目されており，数多くの研究がなされている [23][24]．本節では特に背面のタッチセンサを用いる事でGUI操作を支援する研究について紹介する．

Yangら [25][26] は背面の上部にトラックパッドを取り付け，背面指を動かす事により画面上に表示されるカーソルを動かし，GUI操作を支援している．この研究のようにトラックパッドが背面に取り付けられたモバイル端末は近年商品化もされている [27]．しかし，これらトラックパッド操作により片手操作の支援を行っているが，GUIの操作方法はカーソルによるポインティングであり指直接操作ではない．Windorら [28]，Baudischら [29]，Ohtaniら [30] は透過ディスプレイを用いて，背面からのGUIへの指直接入力を可能にしている．しかし，これらはGUI操作時の操作指によるオクルージョン問題に着目して研究を行っており，片手操作に関しては着目していない．そのため，指直接操作にて親指可動領域外のGUIを操作できない状況が存在してしまう．

2.3 本研究の位置づけ

先ず本研究のモバイル端末の操作手法としての位置づけであるが，本研究にて実現する操作手法はソフトウェアとハードウェア両方を組み合わせて支援を行っているものであると言える．本提案手法では画面全体の上下左右をループさせるという手法を用いる．そしてその

画面ループを生じさせる端末操作としてロール操作を用いる．これにより端末自体をベルトコンベアのように見立てて，裏と表からベルトをずらし回転させるようなイメージで操作を行うことができる．このように，ソフトウェア側の工夫と新たな外部機構への入力手法の両方を上手く組み合わせる事により，操作と結果が結びついた直感的なインタラクション手法を実現している．これを実現した点が本研究の新規性である．

次に，モバイル端末の GUI 操作に関しての新規性について述べる．2.1 節にて述べた通り，従来研究においては，GUI の操作においてはカーソル等を用いた間接操作を用いているものが多かった．また一部指直接操作を達成しているものも存在するが，画面の最上部から最下部へとオブジェクトをスワイプで移動させるといった操作は行えず，限定的な指直接操作と言える．一方本研究では，画面ループを用いる事によりあらゆる状況において変わらない操作感で指直接操作が可能にしており，その点が本研究の新規性であると言える．

第3章　画面ループを利用したモバイル端末片手操作手法

本章では，本研究にて開発する，画面ループを利用したモバイル端末片手操作手法について述べる．また，本手法の適用範囲，応用事例について述べる．

3.1　片手操作と現行のモバイル端末の GUI

Karlson らの調査 [2] によると，PDA 右手片手操作時には親指の可動領域外である画面端への操作が困難であり，画面上端，左端の操作が特に困難であるとされている．これは端末の画面操作領域が大きければ大きいほど顕著になると考えられる．

一方，現行の Android[1]，iOS[2]のインタフェースのデザインガイドライン [31][32] によると，画面最上部や左端にメニューやボタンを設置する事は一般的である．また，モバイル端末上のブラウザアプリケーションにて web ページを閲覧することは多く，その web ページの画面上部にリンクやボタンなどの GUI が配置されている状況も多い．以上のように親指の可動領域外とされる画面上部，左端に GUI が配置されている例は数多く存在すると言える．

ボタンのような GUI であればただボタンをタップすれば操作を行うことが可能である．しかし，現行の GUI には操作にスワイプを必要とするものや，ロングタップ，ダブルタップ等を必要とするものが存在している．例えば，シークバーを操作するのにはスワイプ動作が必要であり，画面のアイコン等をドラッグアンドドロップするためにはロングタップを行うことは多い．従来研究 [11][12] のようなポインタを利用した間接操作では，親指可動領域外の GUI を選択 (クリックやタップにあたる操作) することはできるが，先に述べたような指直接操作ならではのスワイプ，ロングタップというようなリッチな操作を行うことは出来ない．本研究ではこの点に問題を感じており，指直接操作という現行のタッチインタフェースの操作体系を残したまま，親指可動領域外の GUI を操作可能にすることを本研究の目的としている．

3.2　指直接操作を行うためのアイデア

指直接操作を行うための根幹的なアイデアとして，筆者はタッチ位置を目標 GUI 付近へと疑似的に動かすのではなく，画面上に表示さている GUI 自体を親指付近へと動かすことを考

[1] http://www.android.com/
[2] http://www.apple.com/jp/iphone/ios/

えた．それにより指直接操作の操作体系を残したまま，GUI を操作可能になると考える．また，親指で届きにくいとされる画面領域は親指とは反対側に位置する．具体的には右手把持時には，画面上部と画面左部であり，左手把持時には画面上部と画面右部である．そこで，本研究では画面全体を上下左右にループさせる手法を提案する．それにより例えば，画面上部に配置されている GUI を下部から，画面左部に配置されている GUI を右部から移動させることができる．この手法により少ない移動量かつ，高速に GUI を任意の位置に移動させて操作を行う事が可能になると考える．また従来研究 [13] のようにループさせずに動かす方法と異なり，画面からはみ出した部分はその反対側から出現するために，操作が不可能になる領域が存在しなくなるというメリットもある．画面のループを活用する手法は Huot らの TorusDesktop[33] においても提案されている．こちらはデスクトップ環境において利用するシステムである．デスクトップ画面の上下そして左右がシームレスに繋がっており，例えばマウスカーソルを画面左部に移動させると，その反対側である画面右部からマウスカーソルが移動してくるというものである．本研究では，タッチインタフェースにおける指直接操作環境においてループを使用しており，マウスカーソルを動かすのではなく，画面全体を動かし対象 GUI を親指に近づけるという手法を取る点で異なっている．

図 3.1: 画面ループ

3.3　画面ループによる指直接操作

本研究では，GUI の移動方法として画面全体を上下左右方向にループさせて動かす手法を取る (図 3.1)．これにより，右手把持状態において最も操作が行いにくいとされる画面上端と左端は，画面の下端と右端から移動してくることとなる．それにより，ユーザは端末の無理な持ち替えをせずに容易に対象 GUI を操作することが可能となると考える．また，指直接操作の操作体系をそのまま利用可能なので，現行のタッチインタフェースにおいて使用されているシングルタッチによるあらゆる操作 (タップ，ダブルタップ，ロングタップ，スワイプ… etc) が可能になる．そのために，ただ GUI をタップして選択するだけでなく，スライダーをスワイプにてスライドさせることも可能である (図 3.2 左)．また，操作対象にロングタップし選択状態にしてからドラッグアンドドロップすることも可能である (図 3.2 右)．これにより，例えば画面下部に配置されたアイコンを画面上部に少ない移動量でドラッグアンドドロップするといったことが可能となる．画面操作領域が大きなモバイル端末を片手操作する際に，画面下部から画面上部にオブジェクトをドラッグアンドドロップすることは本来ならば非常に困難な動作であるが，本手法を用いる事によって把持姿勢を変える事無く，親指が容易に動作可能な位置にて操作を行う事が可能となる．

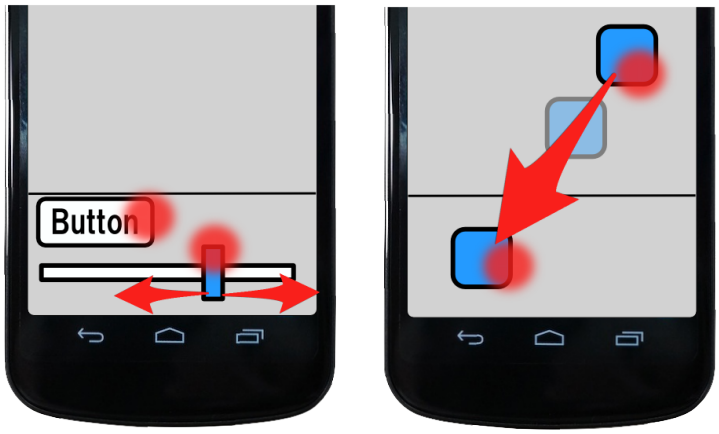

図 3.2: 画面ループによる指直接操作

3.4　画面ループを生じさせる端末操作

本研究では画面ループを発生させる操作方法としてロール操作を提案，実装した．端末の前面と背面に対して，人指し指と親指の相対位置が近づく方向にスワイプする動作をロール操作と呼ぶ (図 3.3)．例えば図 3.3 の左のように，親指と人指し指の相対位置が端末に対して親指：下，人指し指：上の場合，親指は上方向に，人指し指は下方向に動かす操作である．操作可能方向は上下左右の 4 方向である．また，これに近い操作手法は Shen ら [23] の研究でも

提案されているが，本研究では片手のみで行っている点，画面を移動させるトリガーとして用いている点で異なる．

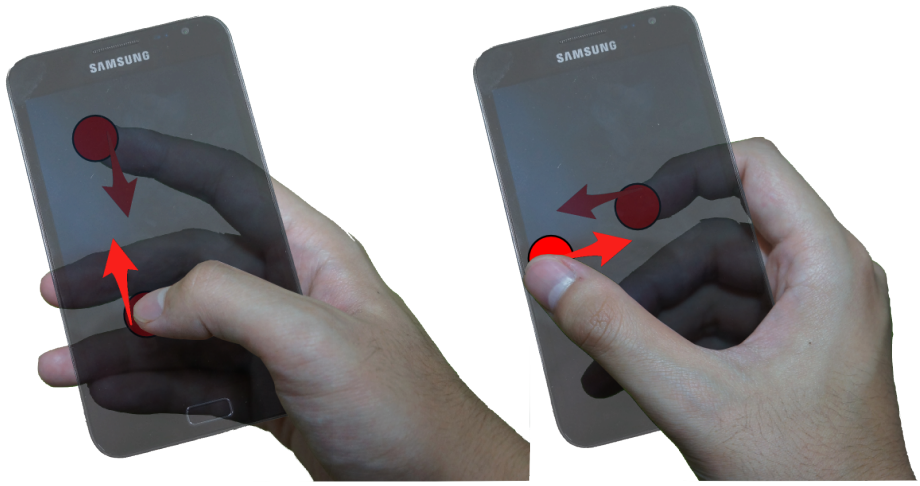

図 3.3: ロール操作

3.5　ベルトメタファを用いた端末操作と画面効果の自然な対応

本研究では両面からのタッチ入力が可能なモバイル端末を用いてインタラクションを行う．従来の片面のみのタッチインタフェースでは，ユーザは二次元平面に対してタッチ操作行っていた．しかし，表と裏の両面に対してタッチ操作を行おうとする場合，ユーザは端末という三次元物体を意識しつつ操作を行う事になる．つまり，単純に表と裏の 2 つの二次元平面に対する操作というよりは，三次元物体に対する操作という感覚でユーザは操作を行うと考える．本研究ではメタファとして三次元物体である「ベルトコンベア」を取り入れた．ベルトメタファの概念図を図 3.4 に示す．デバイス自身を「ベルトコンベア」と見立てて，その両面からベルトをスライドさせ回転させるイメージで操作する事が可能である．また，ベルトは平面がつながって形成されており，ループし続けるものである．そのため，このベルトメタファにより「ロール操作を行うと，画面が回転してループする」という端末操作と画面効果の対応付けは自然であると考える

3.6　本手法の適用範囲

本手法はタッチインタフェース環境において，親指可動領域外にシングルタッチ操作が可能な GUI が配置されている全ての場合において汎用的に適用可能であると考える．具体的には Android や iOS といったようなモバイル端末向けの OS で見られるネイティブ GUI や，web

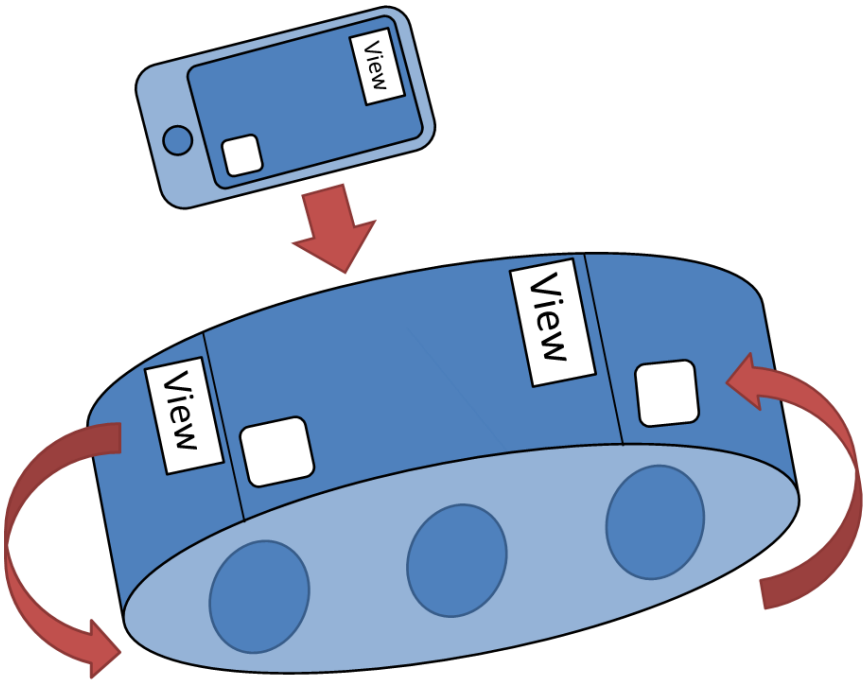

図 3.4: ベルトメタファの概念図

ページにおける GUI に対して適応する事が可能である．また，モバイル端末でブラウザアプリケーションにて web ページを閲覧することは多く，その web ページの画面上部にリンクやボタンなどの GUI が配置されている状況も多々存在する．例えば図 3.5 のような web ブラウザアプリケーションの場合，画面最上部に配置されたネイティブ GUI による固定メニューや web ページ最上部や最左部に配置されたリンクを操作する事は片手操作時には困難である．このような親指の可動領域外に GUI が配置されている際に本手法を適用する事で，親指にて容易に指直接操作することが可能となる．

3.7　現行のモバイル端末操作手法との共存

現行のスマートフォンに代表されるモバイル端末の GUI 操作には，タップ，ダブルタップ，スワイプ等いくつか存在する．これらの操作の内，操作と結果の対応付けが一般的に浸透しているものがある．例えば，画面をダブルタップすることでブラウザの画面が拡大する，スワイプを行うと画面がスクロールしたり，異なる画面に遷移する等である．これらの操作は一般的に浸透しているために，これら操作を行ったときに期待しているものと異なる結果が生じるのはユーザにとって不親切である．本手法で提案するロール操作は両面入力を用いることで，このような既存の操作手法と共存しつつ，競合せずに組み合わせて利用する事が可能であると考える．例えば，写真を閲覧するアプリケーションにて画面上部にメニューが配置されているという場合を考える．ユーザはある画像の閲覧中に画像を拡大するためダブル

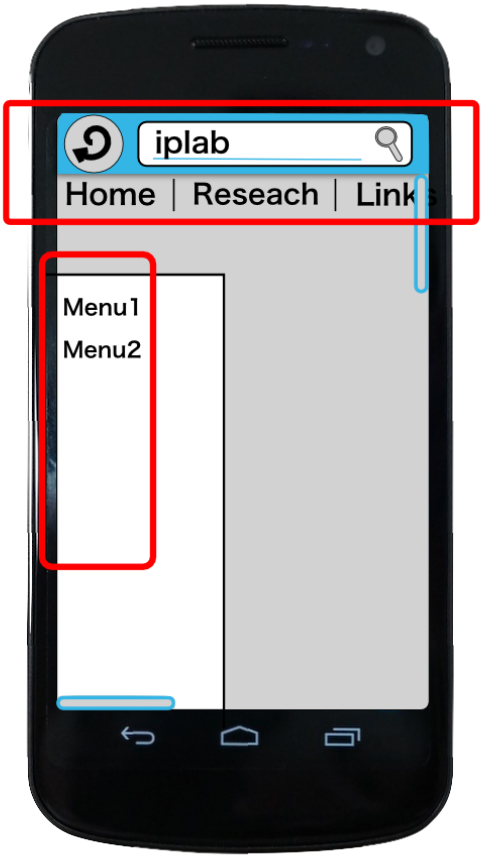

図 3.5: 利用例：Web ブラウザアプリケーションへの適用

タップすると友人が映っているのに気付いた．そこで画面全体をロール操作によってずらし，画面上部のメニューに配置されたボタンをタッチして友人にその画像を送信する，といったような既存の操作手法と自然に組み合わせた利用が可能となる．

3.7.1 応用利用

本手法はスマートフォンに代表されるモバイル端末の片手操作時の操作を円滑にするものであり，両手操作については考慮していない．しかし応用的な利用として，両手によるタブレット端末操作時にも適用可能であると考える．図 3.6 左のように端末を両手で把持した場合，端末上部の GUI に触れるためにはタブレットの持ち替えが発生する．タブレット端末は小型端末と比べ重量があり画面操作領域が非常に大きいので，画面上部へ手を移動した際の腕や手への負担は小さいものではない．しかし，本手法をタブレット端末に応用すると，端末下部を常に把持したまま操作を行う事が可能となると考える．また，ロール操作は画面を

「ずらす」操作であると言える．そのため，ずれた領域分画面を拡張する利用方法も可能だと考える．例えば，図 3.6 右のようにアプリケーション固有のメニューを表示させるという例が挙げられる．

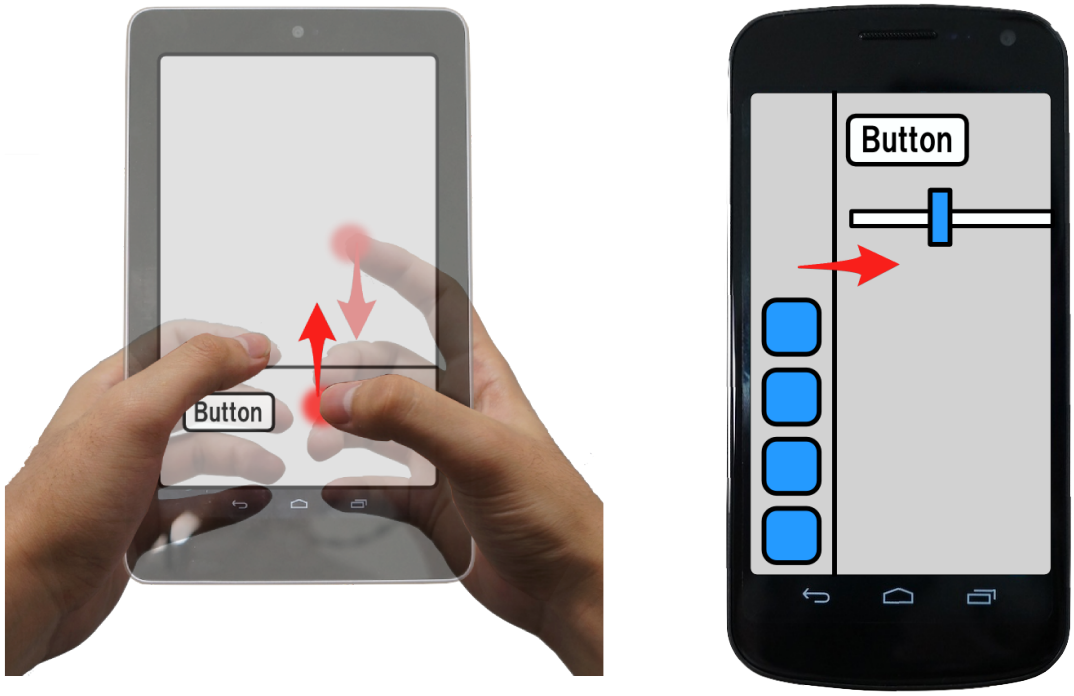

図 3.6: 左:タブレット端末への応用，右:画面のずれを利用した画面拡張

第4章　デバイス設計と実装

本章では，提案手法を実現するためのデバイスの設計を行う．そして，設計をもとに作成したプロトタイプデバイスについて述べる．

4.1　デバイスの設計方針

提案インタラクションは端末前面だけでなく，背面からの入力も必要とする．そしてその背面からの入力は，ユーザが背面指の位置を無理に動かす事無く，端末把持の自然な延長として行うことが可能である事を想定している．ユーザごとに端末の把持姿勢は様々であり，背面接触指の位置も様々である．そのため，ユーザによって異なる背面指の位置を許容出来るように，プロトタイプ背面には可能な限り大きなタッチセンサを搭載する必要があると考える．また，提案インタラクションを行うハードウェア環境として，フルタッチパネルかつ画面操作領域の大きなモバイル端末を操作する想定している．そこで，一般的に大きなモバイル端末に分類される約 4 インチから 5 インチのモバイル端末がデバイス作成に適切だと考える．以上の事から，提案インタラクションを行うデバイスは以下の要件を満たす必要がある．理想デバイスのイメージを図 4.1 に示す．

1. 大サイズのタッチセンサをプロトタイプ背面に搭載
2. 前面の端末はフルタッチパネル端末であり，画面操作領域が約 4 インチから 5 インチの大きさである．

ゲーム機においては，背面にタッチセンサが搭載されたものが販売されている [21]．モバイル端末においても，端末の背面にタッチセンサが搭載されているものはいくつか販売されいてる [27]．しかし，これらの端末では背面の一部にのみトラックパッドが搭載されているのみである．一方，要件 2 を満たす端末は近年数多く登場しており [3]，主流になりつつある．以上から，本研究ではユーザが操作する前面の端末をそのような一般的に販売されている端末を用いて，その端末の背面全体に指接触入力が可能なタッチセンサを取り付けるという方針を取る．

4.2　プロトタイプデバイスの実装

設計方針をもとに，提案インタラクションが可能なプロトタイプデバイスを実装した (図 4.2)．Android 端末を 2 台背中合わせに重ね合わせる事によって，簡易的にプロトタイプデバ

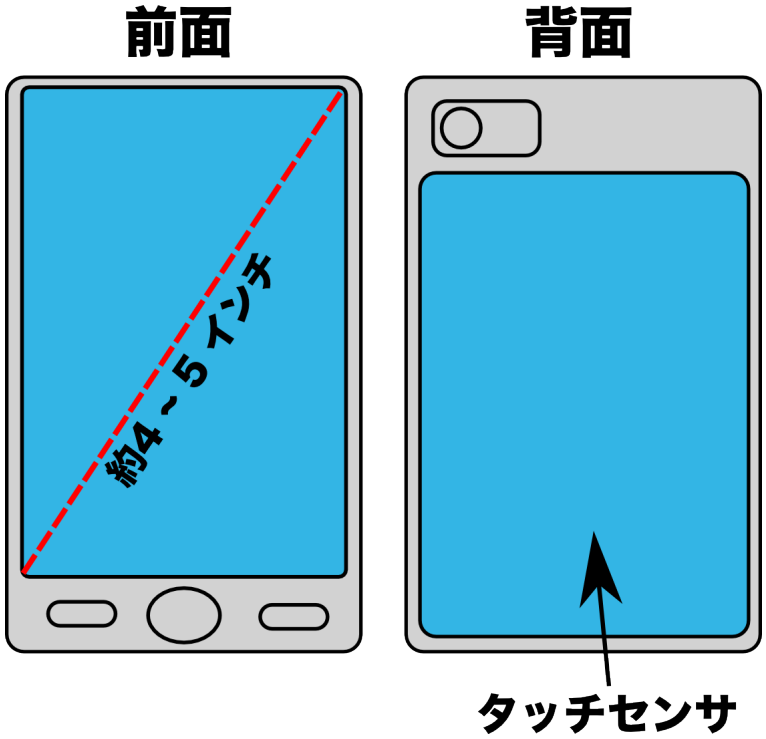

図 4.1: デバイスの理想イメージ

イスを作成した．Android 端末 2 台 (Samsung Galaxy Note 147(H) x 83(W) x 9.7(D) mm 5.3 インチ，Google Nexus S 123.9(H) x 63(W) x 10.9(D) mm 4 インチ) から構成される．プロトタイプのサイズは 147(H) x 83(W) x 21.1(D) mm，重さ 329.8g であった．

本プロトタイプデバイスは右手操作用に作成した．前面の端末が Galaxy Note，背面が Galaxy S である．異なる端末を利用したのは軽量化を図るためである．これらを背中合わせに重ね合わせ，隙間は軽量紙粘土を用いて埋めている．また，重ね合わせる際には図 4.2 のように前面の端末に対して背面の端末を，左端と下端を合わせるように配置した．このように配置した理由は，ユーザの背面指が自然にタッチセンサ上に位置するようにするためである．下端を合わせた理由としては，現行の多くのモバイル端末は端末操作時に頻繁に押下する事となるハードウェアキーやソフトウェアキーが端末下部に配置されているので，ユーザは端末の下部をホームポジションとして把持するためである．左端を合わせた理由は，手の形状から，端末を右手で把持する際の背面指は端末背面左部に位置することになるためである．端末間のデータ送受信には BlueTooth を用いていており，背面指の接触点情報を前面の端末に送信している．

図 4.2 のプロトタイプを養生テープを用いて補強したものが図 4.3 である．図 4.3 の左図ようにユーザが操作する前面の端末は通常のモバイル端末のように操作が可能である．また，図 4.3 の右図のように，前面側から見たときに向かってデバイスの背面左下寄りを覆うようにタッチセンサ領域が存在する．実装に使用した背面端末の Nexus S は端末下部にはバック

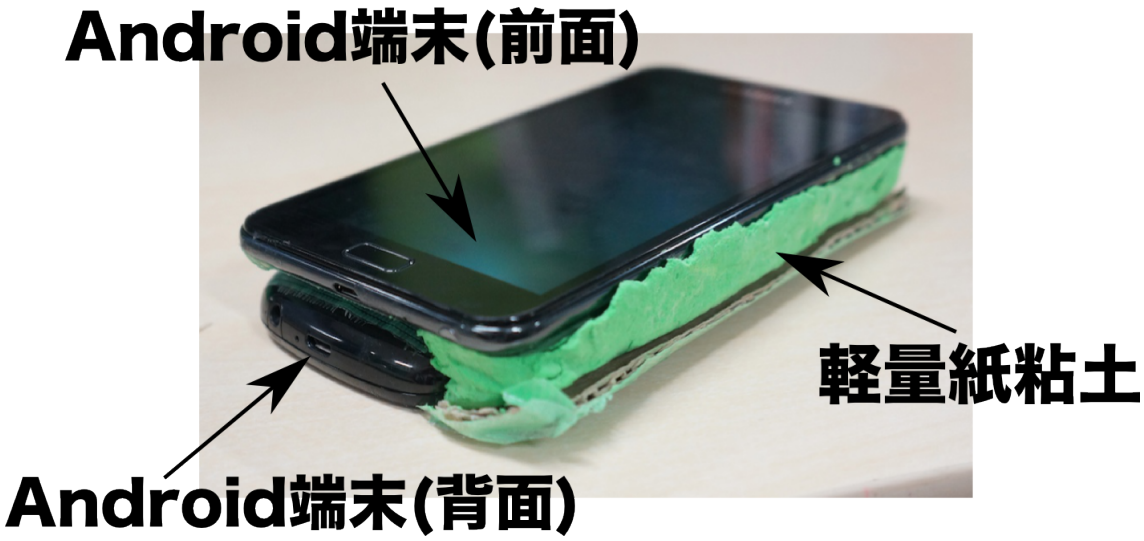

図 4.2: 実装したプロトタイプデバイス

キー，ホームキー等のソフトウェアボタンが配置されており，画面が点灯時には常にそれらが表示されている状態である．そのため，ユーザが本プロトタイプデバイスを把持する際に，背面指によってそれらに触れてしまうということが頻繁に生じた．そのため，プロトタイプデバイス背面下部のソフトウェアボタン表示領域には紙を重ね，薄いカバー状にすることによってソフトウェアボタン押下の誤動作を防止している．

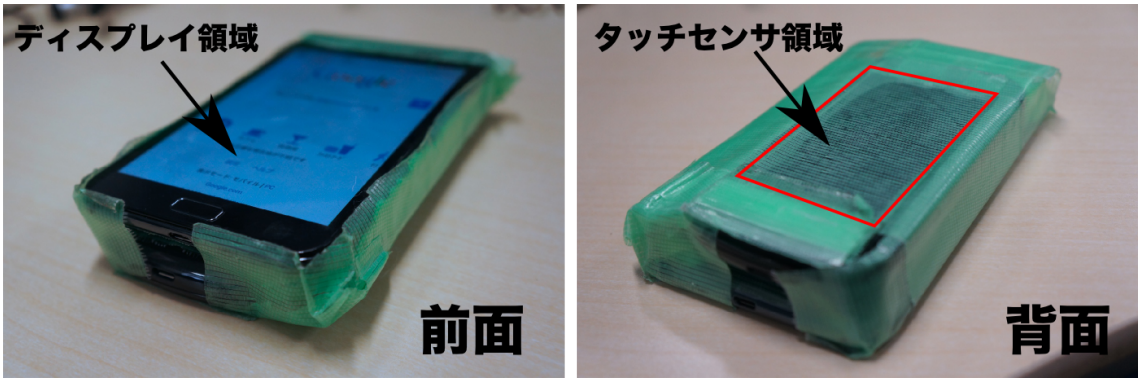

図 4.3: 左：プロトタイプデバイスの前面，右：プロトタイプデバイスの背面

第5章　ソフトウェアの設計と実装

本章では，4 節節にて述べたプロトタイプデバイス上で動作させる，ソフトウェアの設計と実装について述べる．ソフトウェア側では，ロール操作の認識，画面移動を行うためのビューの作成，Bluetooth による指接触情報の通信を行う．また，実際に作成したシステムを被験者に利用してもらいフィードバックを得た．そのフィードバックによる実装の改善についても述べる．

5.1　実装環境

実装環境は以下の通りである．

- プラットフォーム：Android4.0(前面) Android4.1(背面)
- 開発環境：Eclipse Juno(4.2)
- 開発言語：Java

5.2　ロール操作

ロール操作は画面移動を生じさせる操作のことである．ロール操作時には前面を親指，背面を人指し指にて操作することとした．背面の操作指として人指し指を選んだ理由としては，Wobbrock らの研究 [34] によると人指し指は背面とほぼ同様の入力パフォーマンスを持つとされ，背面指の中でも最も動かしやすい指だと考えたためである．背面の接触指の情報を，前面の端末に BlueTooth で送信し続ける．図 5.1 のように，背面には人指し，中指，薬指，小指と最大で 4 本の指が接触する．この背面接触指の中で，人指し指の動作を検知する必要がある．今回の実装では背面接触指の中でも，最上部の接触点を人指し指の接触点と見なす事とした．端末を把持する際に，手の構造上人指し指よりも上部にそれ以外の指が接触する事は通常あり得ないためこのような実装とする．

ロール操作を発生させるためには以下の全ての条件を満たす必要がある．条件は (1) 背面指移動条件，(2) 親指移動条件，(3) 指位置条件の三種類に分かれる．以下で述べる ACTION_MOVE イベントとは Android 端末のタッチイベントにおいて，操作指の移動が発生した際に生じるイベントのことである．

図 5.1: 背面接触指

(1) 背面指移動条件

Xmsec 内に背面最上部指の ACTION_MOVE イベントが N 回連続して発生

(2) 親指移動条件

(1) が満たされてから Ymsec 以内に親指の ACTION_MOVE イベントが発生

(3) 指位置条件

親指の接触位置と背面最上部指の接触位置が適切な位置に存在し，それぞれについて適切な方向に ACTION_MOVE イベントが発生

上述した条件中の X，Y，N の値は背面最上部指の動かしやすさ，つまり端末の大きさや厚さによっても変更されるべきであると考えた．今回作成したプロトタイプデバイスにおいては X:225，Y:50，N:2 と設定した．また，(3) 指位置条件における適切な位置，適切な方向，その際の画面の移動方向を表 5.1 に示す．例えば，親指と背面接触指の相対位置関係が親指：下部，背面接触指：上部であり，親指にて上方向，背面接触指にて下方向の ACTION_MOVE イベントが発生すると画面は上方向に移動する事になる．

背面には基本的に常に指が触れた状態である．そのため，端末の持ち替え等によって背面最上部指の ACTION_MOVE イベントが発生し，背面指移動条件を誤って満たしてしまう可能性がある．そこで上記の (2)(3) の条件により誤動作を軽減できるよう期待し，このような設計とした．(2)(3) を満たさない例を図 5.2 に示す．図のように親指位置が端末前面下部，人指し指が端末背面上部に存在していても，それぞれの指の移動方向が表 5.1 を満たしていないために画面移動は生じない．

画面移動方向は表 5.1 の通り上下左右の 4 方向であり，親指の移動方向と一致する．また，1 度画面が動き出すと (以降，この状態を画面移動可能状態と呼ぶ)，親指の移動のみで画面移動の移動量を変化させることが可能であり，背面指は動かす必要は無い．そして，画面移動

表 5.1: 指位置条件を満たす親指と背面最上部指の適切な位置，方向とその際の画面移動方向

親指		背面最上部指		画面移動方向
位置	方向	位置	方向	
下部	上	上部	下	上
上部	下	下部	上	下
左部	右	右部	左	右
右部	左	左部	右	左

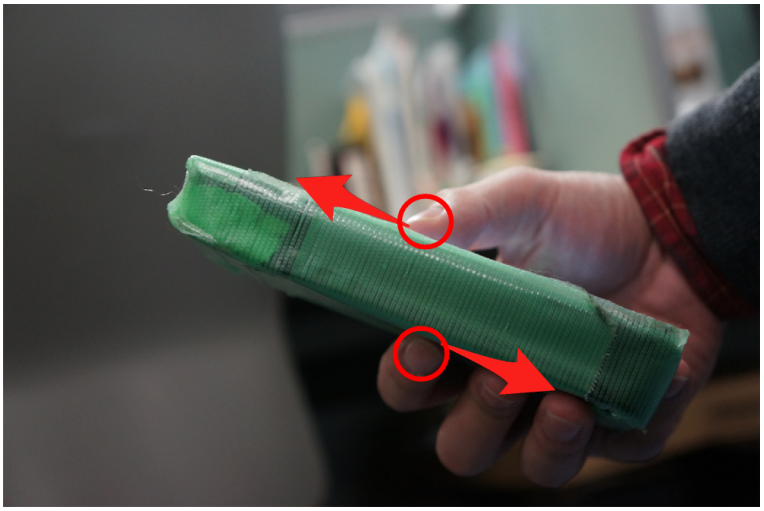

図 5.2: 無効なロール操作

可能状態時に親指を画面から離すとその状態終了し，画面移動は出来なくなる．操作時には，親指のスワイプ動作のみで画面移動量を調整し，触れたい GUI が期待の場所に移動したら親指を画面から離し，GUI を操作することになる．そのため親指を離すことを画面ループの完了と見なすのは自然であると考える．そのためこのような操作設計とする．以降，画面移動可能状態に遷移した回数をロール操作試行回数と呼ぶ．

また，画面移動方向が垂直方向のロール操作を行った場合，親指のスワイプ動作のみにて垂直方向に自由に画面移動させることが可能であるが，水平方向には画面移動させる事はできない．水平方向に画面移動させるためには，再度ロール操作を試行する必要がある．また反対に，画面移動方向が右のロール操作を行った場合，垂直方向には画面移動させる事はできない．

5.3　画面移動

5.3.1　ロール操作以外の画面移動を生じさせる操作

画面移動はロール操作により生じる．画面移動は上下左右の 4 方向であり，上下左右に画面がループしている．移動させるのに必要な操作は 5.2 節の通りである．他に画面移動を生じさせる動作として，画面を初期位置に戻すという動作がある．画面が初期位置に戻る際には，目的の GUI への操作が完了したら，自動的に戻るのが望ましいと考える．さらに，この戻すという操作においても既存の端末操作とは競合しない操作手法が望ましいと考える．そのため，目的の GUI への操作が完了する，もしくは端末背面をダブルタップすると初期位置に元に戻るように実装する．ダブルタップ操作は図 5.3 の操作を 2 回行う事で発生する．

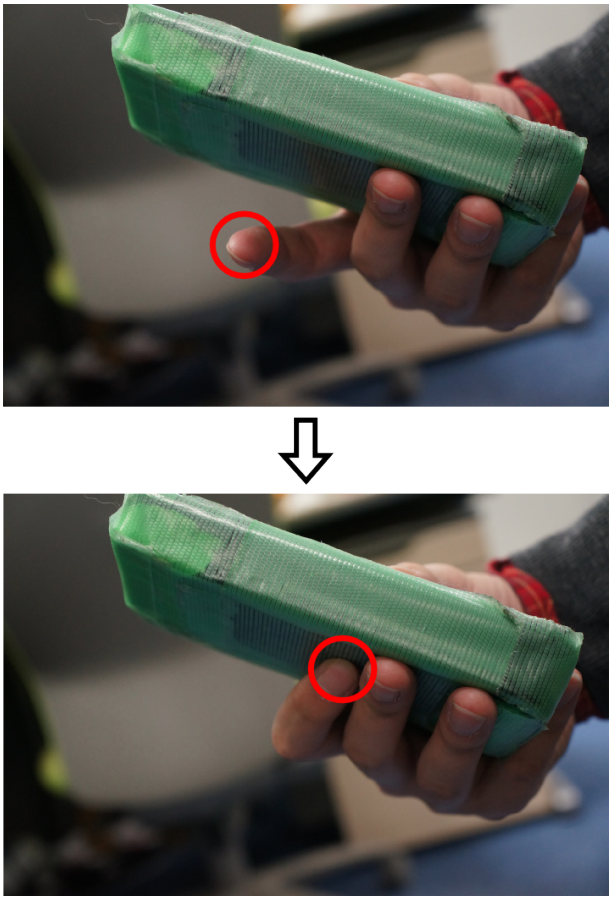

図 5.3: 背面指によるダブルタップ

目的の GUI への操作の完了の判別方法は，GUI の種類によって異なる．例えば，ボタンやリンクをタップすることはそのまま操作の完了と見なすこととした．またスライダーはスワイプ操作が完了し，親指を画面から離した時点で完了と見なすこととした．しかしトグルボ

タンやチェックボックス等の GUI に対する操作の場合，ユーザの操作がどこで完了したのかをシステム側で自動判別するのが困難である．そのため，このような GUI に対する操作の場合は画面を自動で初期位置に戻さず，親指のダブルタップという明示的な操作により画面を戻すように実装する．

また，背面のダブルタップの認識には以下の (1)(2) の条件を満たすように実装を行う．ACTION_DOWN イベントおよび ACTION_POINTER_DOWN イベントとは Android 端末のタッチイベントにおいて，指が画面に接触した際に生じるイベントのことである．ACTION_DOWN イベントはシングルタッチ時に呼ばれ，ACTION_POINTER_DOWN イベントはマルチタッチ時に呼ばれるイベントである．また，式 5.1 中の preX，preY は前回の ACTION_DOWN イベントもしくは ACTION_POINTER_DOWN イベント時の座標値である．

1. Xmsec 以内に ACTION_DOWN イベントまたは ACTION_POINTER_DOWN イベントが 2 回生じる
2. ACTION_DOWN イベントまたは ACTION_POINTER_DOWN イベントの X，Y 座標が式 5.1 を満たす

$$|X - preX| < 50, |Y - preY| < 50 \tag{5.1}$$

5.3.2 ループビュー

ループビューは画面ループを可能にする Android の View である．ループビューは Android の ViewGroup クラス[1]を継承することにより実装している．ループビューの全体像は図 5.4 の a の通りである．ループビューは 4 つの領域から成っており，左上の領域をメイン領域と呼び，それ以外を複製領域と呼ぶ．複製領域にはメイン領域と同様の内容を表示している．画面ループが発生していない状態では，端末の画面にはメイン領域を表示している．ロール操作が発生時にはループビュークラスの scrollTo メソッドを呼び出し，端末画面に映す表示領域をずらすことによりあたかも画面がループしているようにユーザに見せかけている．例えば，b 領域を画面に表示しようとした場合，c 領域を代わりに表示する事によって，ユーザにはループしているように見せかける事が可能である．上下も同様の方法で画面ループを実現する．どの領域で発生した GUI イベントも，すべての領域で共有している．そのために，例えばどれか一つの領域内のチェックボックスがチェック状態になった場合，他領域の対象チェックボックスもチェック状態になる．

5.4 Bluetooth による端末間通信

背面の Android 端末から前面への端末へと接触点情報等のデータを Blutooth 通信により送信する．端末間通信のプロトコルは，背面指最上部指の x,y 座標情報，背面指移動条件を満た

[1]http://developer.android.com/reference/android/view/ViewGroup.html

図 5.4: ループビュー

したか (1)，満たしていないか (0) となっている．背面指移動条件が 1 の場合，親指移動条件，指位置条件を満たしていれば，画面ループが発生する．背面の端末では指接触位置情報を取得すアプリケーションが常に起動した状態である．前面の端末にて対応するアプリケーションを起動しているときのみ，そのアプリケーションが情報を取得する．背面のタッチセンサには最大で 4 本の指が接触する事になるが，今回の実装では背面最上部の接触指を人指し指とみなし，その接触位置座標を前面の端末に送信している．また，前面と背面では異なるデバイスを用いている．そのために，背面端末で取得した座標値を前面端末に渡す際には座標値を解像度，画面操作領域に応じて変更する必要が生じる．また，前面の端末と背面の端末では端末の向きが逆であるために x 座標を変換する必要がある．そのために，今回の実装環境においては以下のように変更を行った．変換前の座標をそれぞれ x，y，変換後の座標をそれぞれ convertedX，convertedY とする．また背面のスクリーン最大横幅 (横向きの解像度) を displayWidth とする．それぞれの式の定数は筆者が実験的に定めたものである．

$$convertedX = (displayWidth - backX) \times 1.2 \tag{5.2}$$

$$convertedY = backY \times 1.3 + 220 \tag{5.3}$$

5.5 予備実験による実装改善

設計をもとに実装を行い，筑波大学システム情報工学研究科コンピュータサイエンス専攻の学生 10 人に試用してもらい，自由記述のコメントを得るという予備実験を行った．予備実験の際には，ビデオカメラにより利用の様子を観察した．予備実験の様子を図 5.5 に示す．利用したアプリケーションは画面の上部にボタン，チェックボックス，シークバーの 3 タイプの GUI が配置されている画面レイアウトのアプリケーションである (図 5.6)．このアプリケーションに本手法を適用した．この予備実験においては，画面のループは上下方向のみ行うようにした．タスクは全部で 24 試行あり，ボタンレイアウト，チェックボックスレイアウト，シークバーレイアウトそれぞれ 8 試行ずつである．画面下部の指示通りに GUI を操作すると，画面下部に Next と表示されたボタンが出現し，それを押下する事で次のタスクへ進む事が可能となる．操作対象の GUI は無作為かつ均等に指定した．タスクの実行順はレイアウトのタイプごとに 8 試行ずつ行った (例: ボタンレイアウト x8 → チェックボックスレイアウト x8 → シークバーレイアウト x8)．

図 5.5: 予備実験時の様子

GUI 操作後の画面移動の挙動

予備実験を行った際に「GUI が自動的に戻るのと戻らない場合があるのに違和感を感じる」(2 名) という声があった．具体的にはボタンやシークバーは操作後に自動的に画面が初期位置

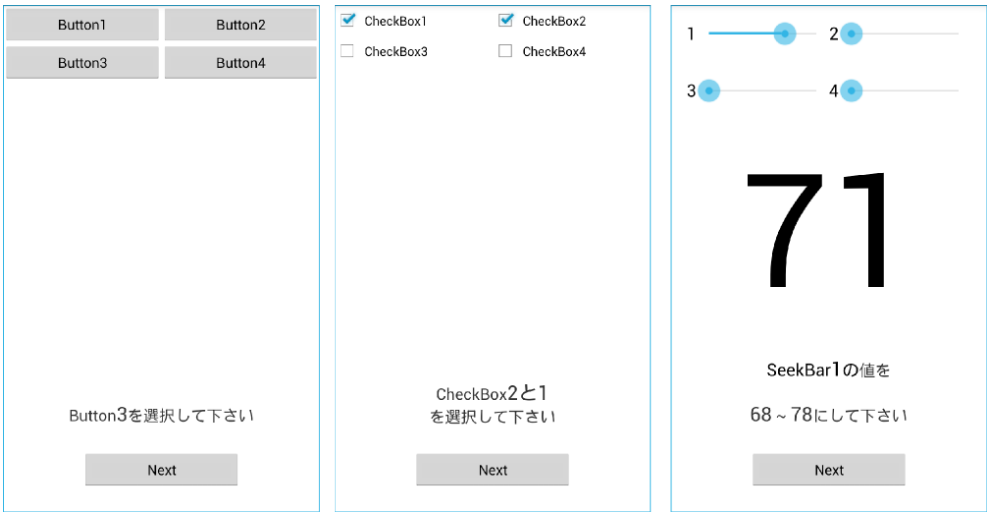

図 5.6: 予備実験用のアプリケーションの画面レイアウト

に戻るが，チェックボックスは戻らないという動作に違和感を感じるユーザが多かったということである．また「ボタン，シークバーは操作完了後に，自動で画面が初期位置に戻るのは使いにくい」(2 名) という意見が得られた．何故使いにくかったのかを分析するために，ビデオカメラにより実験の様子を観察した．その結果，先のコメントは「GUI を連続して操作したい場合に，不快感を感じる」ということであると筆者は結論付けた．例えばボタンタスクでは，指示されたボタンを正しく選択すると，画面下部に Next ボタンが表示され，それと同時に画面がアニメーションし初期位置に戻る．この際に被験者は画面をループさせてタスクを行っているため，画面上の表示では 4 つ配置された GUI のすぐ上部に Next ボタンが表示されることになる．この Next ボタンは親指でも容易に届く位置に配置される事になるので，被験者はすぐにそのボタンをタップし次のタスクに進もうとする．しかし，そのボタンに触れようとすると画面が自動的に初期位置に戻り，選択操作を失敗してしまうために，不快感を感じてしまうということである．チェックボックスについては操作が完了しても，自動で初期位置には戻らないのでそのような意見は得られず，ビデオでも困惑する様子は観察されなかった．これらのことから，GUI 選択後には画面を初期位置に自動的に戻す事はせずに，ユーザの明示的な動作によってのみ戻すように実装を修正した．

画面を初期位置に戻す操作方法の変更

また，利用時の様子を筆者が観察したところ，意図せず背面のダブルタップが発生してしまい，画面が初期位置に戻ってしまうユーザがいたことが確認された．それは，端末の背面には常に親指以外の指が触れている状態のために，少しの端末の持ち替え等によってダブルタップが生じてしまう事があるためである．このことから，背面のダブルタップよりもより誤動作の少ない操作を初期位置に戻る動作に割り当てる必要があると考える．考えられる手

法として，前面の端末画面をダブルタップやロングタップ，あるいは前面の画面のベゼルをスワイプすると言った方法が考えられる，しかしこれらは既存の画面操作の操作手法と競合する可能性があるために，用いるべきではないと考える．他にも，前面の画面にソフトウェアボタンをオーバーレイしそれを選択させるといったような方法も考えられるが．ユーザに任意のソフトウェアボタンを選択させる操作というのは，ユーザの視覚認知的負担が生じさせるために，適切とは言えないと考える．筆者は代替案として表，裏両面からのダブルタップに画面位置を戻すように実装した (図 5.7)．この手法は，既存の操作手法とも競合せず，視覚認知的負担も生じさせず，さらに把持姿勢を大きく変えずに行える動作であるので，ユーザの負担にもならない操作方法であると考える．

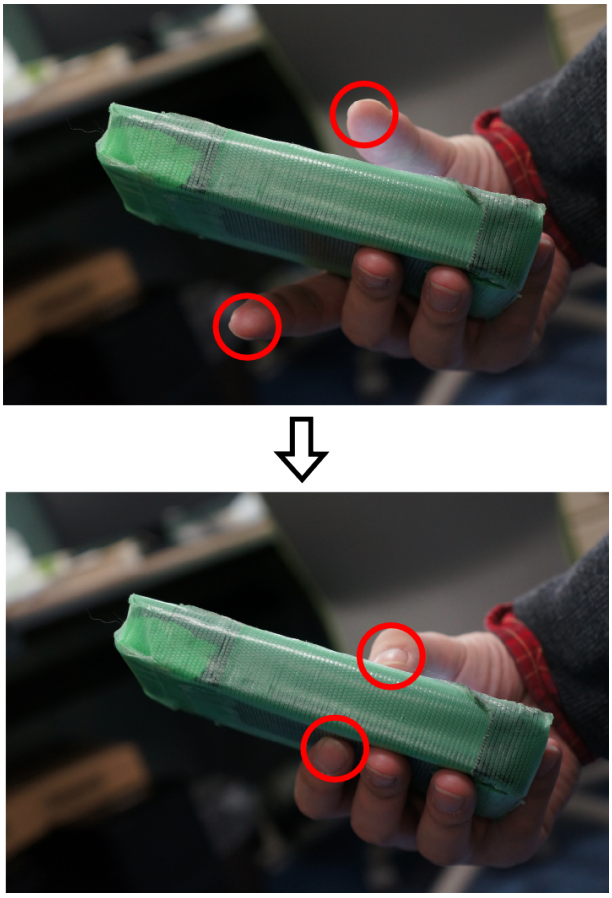

図 5.7: 両面の指を用いたダブルタップ

左右方向の画面ループ

この予備実験においては，上下方向の画面ループのみ行うようにした．それは，上下方向のみのロール操作だけでも十分に GUI は操作可能になるのではないかと考えたためである．実

験の様子を観察したところ，ほとんどの被験者は上下方向のみの画面ループで GUI を操作可能になることが観察された．しかし，やはり画面左端に寄った一部 GUI の操作が困難になる被験者が多数見受けられた．このことから，やはり左右方向の画面ループは必要であると考えられる．加えて,「画面ループの方向は上下だけでなく，同時に左右にも動くと操作が 1 度で済んで行いやすい」(1 人) というコメントがあった．以上を踏まえ，以前の実装に置いては垂直方向の画面移動中は水平方向に画面移動させる事はできなかったが，水平方向にも画面移動可能なように実装を修正した (図 5.8)．これにより基本は上下方向の画面ループを行い，補助的に左右方向に移動させるといったような利用が出来るのではないかと考える．それによりロール操作回数が 1 回のみで目的の GUI を親指可動領域内に近づける事が可能になると考える．

5.6 アプリケーションの実装

本手法を適用したアプリケーションを実装した．画面ループを行うアプリケーションは LoopBaseActivity を継承する必要がある．ここで言う Activity とは Android においてアプリケーションの画面を表すオブジェクトであり，その画面の生成や破棄等のライフサイクルも管理している．この LoopBaseActivity は Activity を継承したものである．LoopBaseActivity では前面の端末へのタッチイベントを常に監視している．また，bluetooth 接続状態の管理を Actibity のライフサイクルに紐づけて行っている．具体的には，この Activity の生成と同時に bluetooth 通信の待ち受け状態となり，背面から接続要求が行われるとコネクションを確立する．この Activity が破棄されたときには同時にコネクションも切断される．このような設計となっているために，画面ループを行うアプリケーションは LoopBaseActivity を継承しさえすれば，タッチイベントの監視や bluetooth 通信の処理は行う必要なく，画面作成を行うのみで実装することが可能である．

ブラウザアプリケーションの様子を図 5.9 に示す．図 5.9 のレイアウトのように，画面操作領域が大きいスマートフォンにおいては画面最上部のブックマーク登録ボタンを押下することは非常に困難である．しかし，本手法を適用することで，画面最上部のメニューバーを画面下部から出現させる事が出来るので，把持姿勢を変える事無く容易にブックマーク登録ボタンを押下する事が可能である．また web ページ最上部，あるいは左端にリンクが貼られている web ページも少なくない．そのような時も同様に本手法を利用する事で，円滑に操作が可能になる．また，応用的な利用として画面拡張メニューの実装を行った (図 5.10)．これはアプリケーションランチャーとなっており，左右方向に画面移動させると，ずれた領域分画面が拡張され，起動可能なアプリケーションアイコンが表示される．そのアプリケーションアイコンをメイン画面にドラッグアンドドロップすることによりそのアプリケーションを起動する事が出来る．

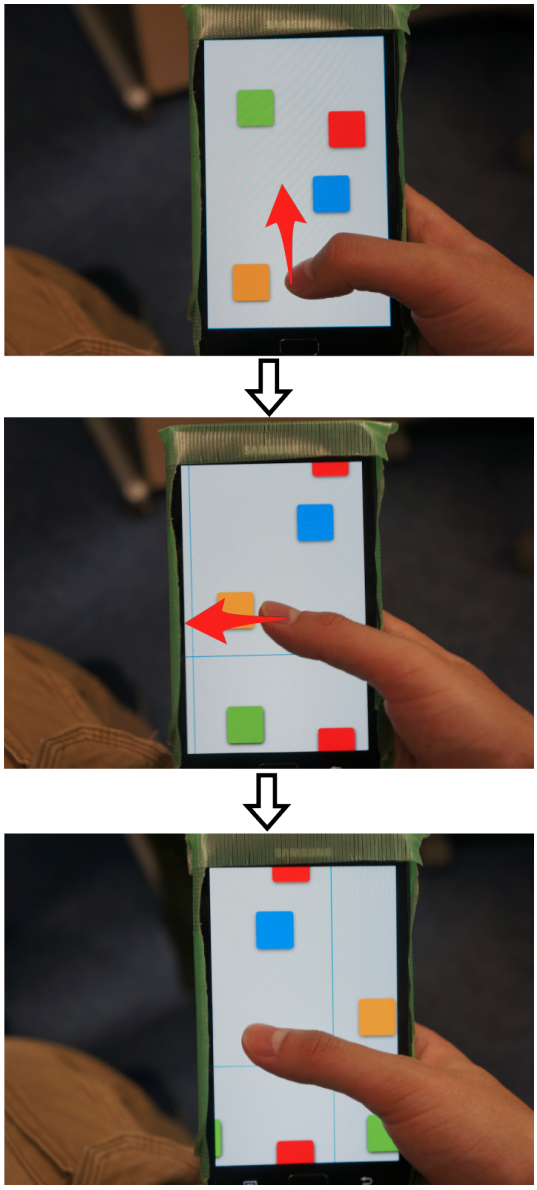

図 5.8: 画面移動受付状態時の上下左右方向への画面移動

図 5.9: ブラウザアプリケーション

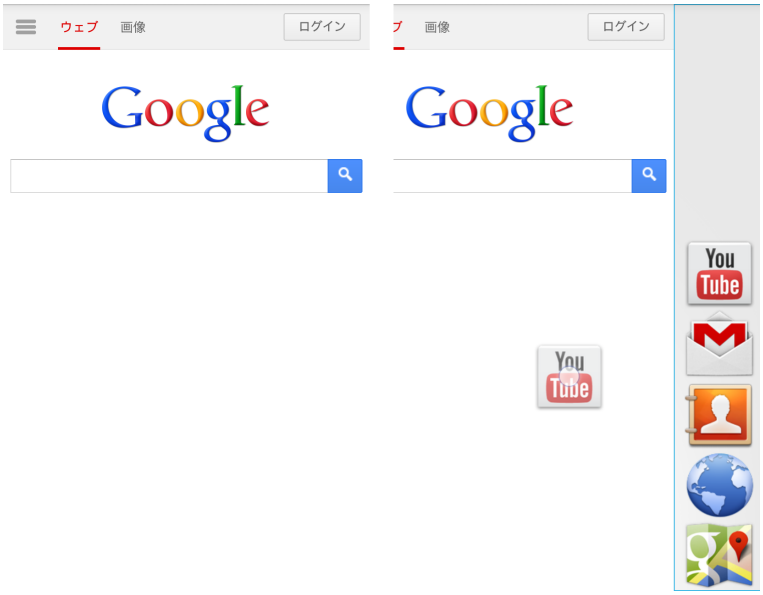

図 5.10: 画面拡張アプリケーションランチャー

第6章　実験

本章では，本研究にて開発した手法についての評価実験について述べる．プロトタイプデバイス上で動作する実験用アプリケーションを作成し，被験者に利用してもらった．本手法を適用した場合と適用しなかった場合について比較することによって評価実験を行った．

6.1　実験目的

画面操作領域の大きなモバイル端末片手操作時において，画面ループを用いる事によって，親指可動領域外の GUI が円滑に操作可能になるかどうかを評価することを実験の目的とする．そこで提案手法の適用時と非適用時を比較する事によって評価実験を行う．比較内容として操作性，負担度，タスク完了時間について比較評価する．

また，提案手法が直感的な操作手法であるかどうかもインタラクションを行う上で非常に重要な要素である．そのため，ロール操作を行うと画面全体が動くという操作と結果の対応付けの自然さについても検証する．

6.2　実験内容

プロトタイプデバイス上で動作する，実験用のアプリケーションを作成し被験者に利用してもらう．実験用のアプリケーションは Android OS 固有の GUI を用いたアプリケーションである．実験後は，操作の負担度，操作性，提案手法的用事における画面ループ操作の自然さに関するアンケートを行う．また，実験を行っている際には 1 試行が完了する時間を計測する．被験者は座った状態で操作を行う．本手法の適用時と非適用時の二つのタスクを比較することにより実験を行う．それぞれを適用条件，非適用条件と呼ぶ．適用条件においては提案手法を利用して実験を行ってもらい，非適用条件においては提案手法を適用せずに実験を行ってもらう．操作方法とプロトタイプデバイスに慣れるために，それぞれのタスクの実験を行う前に練習を行わせ，練習時間は自由とした．練習は本番と同様のレイアウトで行わせる．本手法の練習においては，図 3.3 を印刷した用紙を被験者に見せて手法を説明し，ベルトコンベアの様なイメージで操作すると伝える．被験者が行う内容はプロトタイプデバイス上で動作するアプリケーションの GUI 操作である．本手法の利用環境としては画面上部に GUI が配置されているという状況を想定しているため，そのような状況を満たすアプリケーションを作成した (図 6.1)．左から順に (a) ボタンレイアウト (以降，B)，(b) チェックボックスレイアウト (以降，CB)，(c) シークバーレイアウト (以降，SB)，(c) ドラッグアンドドロップレ

イアウト (以降，DD) となっている．特殊な GUI を除き，一般的な GUI はタップ操作，あるいはスワイプ操作により操作を行う事が可能である．そのため B，チェックボックス，シークバー，ドラッグアンドドロップそれぞれが配置されている 4 種類のレイアウトとした．ボタンとチェックボックスはタッチ操作に関して，シークバー，ドラッグアンドドロップはスワイプ操作に関して検証するために採用する．また，ドラッグアンドドロップを行う際に，オブジェクトを移動可能にするにはロングタップを用いる．

図 6.1: 実験用アプリケーションのレイアウト．(a)B レイアウト，(b)CB レイアウト，(c)SB レイアウト，(d)DD レイアウト

ボタン，チェックボックス，シークバーについては図 6.1 のように画面上部に同タイプの GUI を 4 つ配置し，それらの内一つまたは複数個の GUI を操作するように指示する．1 タスク中で表示される画面レイアウトは 4 種類である．1 つのレイアウトにつき 8 試行い，合計で 32 試行である．それぞれのレイアウトの試行を行う順番は，被験者ごとに無作為である．B については画面上部に表示される 4 つのボタンの内，指定されたボタンをタップすることで完了する．CB は画面上部に表示される 4 つのボタンの内，指定された 2 つをタップし選択状態にすることにより完了する．SB は画面上部に表示される 4 つのシークバーの内，指定された 1 つをスワイプ操作により動かす．動かした量によって画面中央の数値 (以降，シーク値と呼ぶ) が変化し，指定の範囲 (以降，シーク目標範囲と呼ぶ) まで動かしてから親指を画面から離す事で完了する．DD ではドラッグ対象をドロップ領域にドロップする事で完了する．ドラッグ対象をロングタップするとドラッグ対象を移動可能状態に出来る．移動可能状態にスワイプ操作するとドラッグ対象を自由にドラッグ出来る．そしてそのドラッグ対象がドロップエリア半分以上進入するとドロップ領域の色が青から赤へと変化する．その状態で親指を画面から離すとドロップが行われ，操作が完了する．どの種類のタスクも操作が完了すると，画面下部に次のタスクへ進むよう促すテキストが表示され，図 6.2 の赤丸で囲まれた部分であるプロトタイプ右下部のシステムキーを押す事で次の試行へ進む．ユーザにプロトタイプデバイス右下部のシステムキーを押下させ次のタスクに進むように設計した理由は，画面下

側をホームポジションとして把持する事を被験者に促すためである．これは，現行のモバイル端末の多くは端末操作に頻繁に利用する事になるハードウェアキーやソフトウェアキーが画面下部に配置されているため，一般的に端末下側を把持して操作を行う事は一般的であるためである．

図 6.2: プロトタイプ下部のシステムキー

操作対象の GUI はごとに異なる．B においては，均等かつ無作為に操作対象のボタンを指定した．CB についても均等かつ無作為に操作対象のチェックボックスを指定した．SB は均等かつ無作為に操作対象のシークバーを指定した．シーク目標範囲の指定方法は画面左部に配置されているシークバー 1，シークバー 3 についてはシーク目標範囲を 10 20，20 30 とし，画面右部に配置されているシークバー 2，シークバー 4 についてはシーク目標範囲を 20 30，80 90 とした．シーク目標値についてはなるべく操作が困難になるであろう値を設定した．この値は筆者が経験的に決定した．DD では，ドラッグ対象とドロップ領域の組み合わせが 8 つ全て異なる．その組み合わせを表 6.1 に示す．ドラッグ対象とドロップ領域が離れるように配置した．

表 6.1: ドラッグ対象とドロップ領域の組み合わせ

ドラッグ対象	ドロップ領域
左下部	上部
右下部	上部
左上部	下部
右上部	下部
右下部	左部
右上部	左部
左下部	右部
左上部	右部

6.3 実験結果と考察

被験者は 10 人で内一人は女性であり，全員右利きであった．実験中は端末操作の様子を前面と背面から 2 台のビデオカメラ撮影し，観察を行った．実験の様子を図 6.3 に示す．また実験後にアンケートを行った．非適用時と適用時それぞれについて，5 段階のリッカート尺度を用いてタスク全体を通しての GUI の操作性 (1:非常に困難，5:非常に容易)，手への負担度 (1:非常に負担，5:全く負担はない) について回答させた．また，それぞれを選択した理由についても記述させた．B，CB，SB，DD それぞれについても適用時，非適用時について上記と同様に GUI の操作性と手への負担度について回答させた．また,「親指と人指し指をスワイプ操作すると，画面全体がずれる」という操作と結果の対応付けについて，自然であるかどうか (1：非常に不自然，5：非常に自然) を回答させた．最後に提案手法に関して自由記述のコメント欄を設け回答させた．実験中には 1 試行の完了する時間を計測した．

手への負担度と GUI の操作性に関する，5 段階リッカート評価の結果を図 6.4 に示す．図中の左上が実験全体を通した評価，右上が B に関する評価．左下が CB に関する評価．下中央が SB に関する評価．右下が DD に関する評価である．全てのグラフにおいて，縦軸は全被験者の評価点の平均値である．全てにおいて操作性，負担度ともに適用時の方が良い評価が得られた．

6.3.1 タスク全体に関して

手への負担度

タスク全体を通した評価を見ると，非適用条件に関して得られたコメントとして「操作対象 GUI に指を伸ばしたり，端末の持ち替えが頻繁に発生し非常に負担に感じた」(3 名)「端末を落としそうそうになった」(1 名)「端末を手のひらにのせて操作するしか無く，画面を見やすい状態で操作するのが不可能であると感じた」(1 名)「画面左上にタッチするのには苦労した」(1 名) というものがあった．撮影したビデオ観察結果からも，端末を取り落としそうに

図 6.3: 実験の様子

なったり，頻繁に把持姿勢を変えながら苦労して操作する様子が見受けられた．さらに 5 段階評価の結果も，非適用時の手の負担度が約 1.5 と評点が低かった．

しかしながら，適用条件は約 3 と中間評価値であり，少なからず負担を感じてしまっていた事が分かる．適用時に関するコメントを見ると，否定的なものとして「普段慣れている持ち方が出来なかったため負担に感じた」(1 名)「背面指を動かす操作 (ロール操作) が負担に感じた」(2 名)「端末が大きく，重かったため負担であった」(2 名) というコメントがあった．一方肯定的なコメントとして「親指を伸ばす必要がなかったので負担を感じなかった」(1 名)「手法に慣れると少ない手の動きで，ロール操作を行え負担は少なかった」(2 名) という意見があった．以上から，ユーザによっては (ユーザの手の大きさや形，普段の慣れた把持方法によっては)，プロトタイプデバイス上でのロール操作を負担無く行えていたことが分かる．また，今回作成したプロトタイプデバイスは一般的なモバイル端末の約 2 倍の重さ，厚さがある．コメントからも分かるように，この点も負担度合いに大きな影響を与えている事が考えられる．より薄く，軽量なモバイル端末にて本手法を利用する事で，ユーザの端末の把持が行いやすくなり，背面指の操作の負担が軽減される可能性が考えられる．

GUI の操作性

適用時に多く得られた GUI の操作性に関しては否定的な意見は得られなかった．肯定的な意見として「親指の近くに GUI を持って来れるので，容易に操作出来た」(2 名)「変に把持姿勢を変える事無く，容易に操作出来た」(2 名)「普段通りの操作感覚で操作出来た」(1 名) というものがあった．これらの肯定的な評価は図 6.4 の結果にも表れていることが分かる．

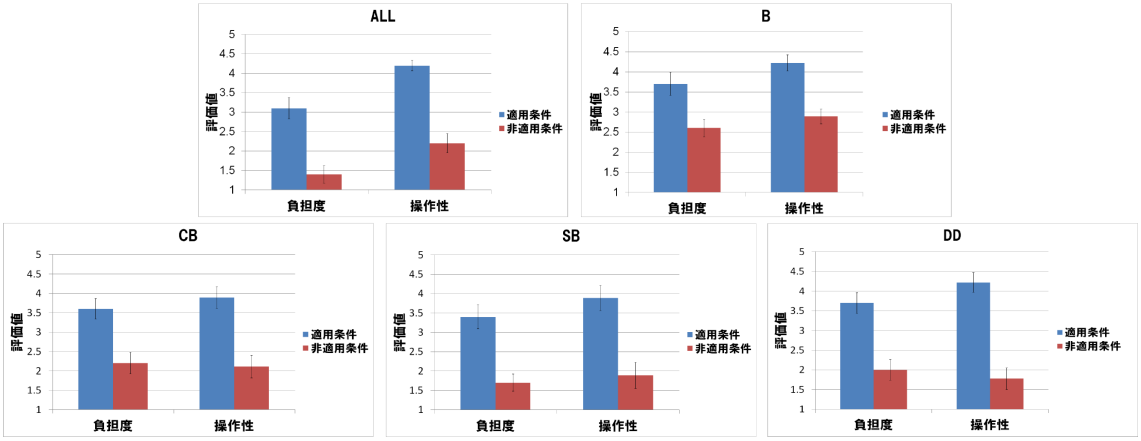

図 6.4: GUI 操作性と手への負担度に関する 5 段階リッカート尺度評価. 左上：タスク全体に関して，右上：B に関して，左下：CB に関して，下中央：SB に関して，右下：DD に関して

インタラクションの自然さ

5 段階のリッカート尺度による評価にて，評価値の平均は 4.2 であった (1：非常に不自然, 5：非常に自然)．得られたコメントとして「ベルトのようなという説明から容易に操作から結果が想像出来た」(1 人),「対応付けが簡単で，分かりやすかった」(3 人) というものを得た．以上のように好意的な評価を得る事ができた．この結果から，操作と結果が対応づいた自然なインタラクションを実現出来ていると言えると考える．しかし一方で,「一度画面をずらした後に，親指だけで動いてしまうのはベルトコンベアを回すイメージとは異なる」(1 人) という否定的なコメントも得た．

6.3.2　レイアウトごとに関して

全てのレイアウトに関して，手への負担度，操作性ともに適用時の方が良い評価を得られた．これらのレイアウトではタップ，スワイプ，ロングタップといった操作を利用し，シングルタッチにて操作可能な一般的な GUI に対して操作を行った．つまり，提案手法適用時に上記の操作を満足に行えたということは，モバイル端末のシングルタッチ操作時のほとんどの状況において一様に円滑な GUI 操作を行う事が可能であることを示唆している．

全てのレイアウトの中でも，特に SB と DD の適用条件と非適用条件の差が開いている事が見て取れる．これらはどちらも操作にスワイプを用いるレイアウトである．まず SB に関してビデオ観察を行ったところ，特に最も左上に配置されたシークバーを操作する際に，親指が思うように動かせずに GUI 操作を満足に行えていない様子が観察された．これは，画面左上部は親指がもっとも届きにくいとされる位置であるために，その位置で親指を左右方向にスワイプすることが非常に困難であるためだと考える．次に DD に関して，ビデオ映像を行ったところ，非適用条件において，ユーザによっては把持姿勢を変えて複数回に分けてドラッ

グアンドドロップをしていたり，親指を無理に伸ばし親指が画面から意図せず離れてしまう様子が観察された．これはドラッグ対象からドロップ領域までの距離が離れているためであると考える．さらにユーザごとのドラッグアンドドロップ試行回数を図 6.5 示す．ドラッグアンドドロップ試行回数とは一度ドラッグ対象を動かし始めてから，親指を画面から離すまでの一連の動作を行った回数である．この回数は実験アプリケーション上で計測した．対応のある有意水準 1%の両側検定の t 検定を行ったところ，非適用条件よりも適用条件の方が有意に DD 試行回数の平均値が小さかった ($t(79) = -3.65$，$p = .0004$)．図 6.5 によると，適用条件においてはほとんどの被験者は一回のドラッグアンドドロップ試行回数でタスクを完了しているのに対して，非適用条件においてはドラッグアンドドロップ試行回数が大きくなっている．このことからもユーザがドラッグ，つまりスワイプ操作を苦労して行っていたことが分かる．以上から，以下の状況が特に操作が困難であり，同時に提案手法がより効果的である状況であると言える．

1. 画面下部から画面上部のように，始点と終点の距離が大きく離れるスワイプ操作を行う状況
2. 画面の左上部のような親指が届きにくい領域におけるスワイプ操作を行う状況

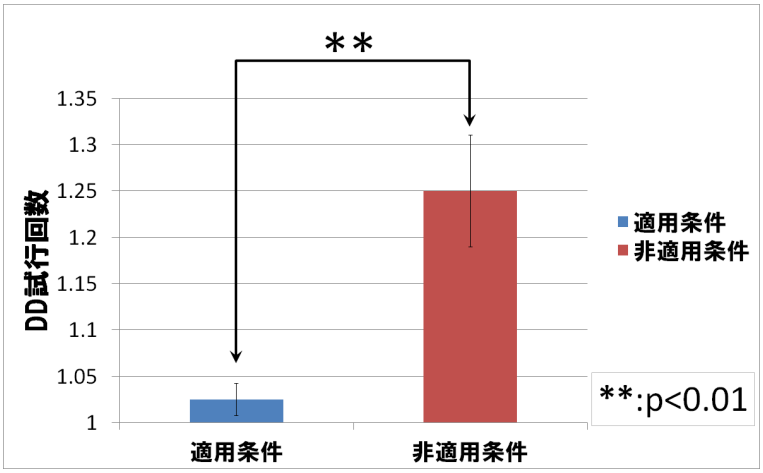

図 6.5: ユーザごとのドラッグアンドドロップ試行回数の平均値

6.3.3　1 試行あたりの実行時間

レイアウトごとの 1 試行あたりの時間を図 6.6 に示す．図中エラーバーは標準偏差を表す．

対応のある有意水準 5%の両側検定の t 検定を行ったところ，ALL において適用条件よりも非適用条件の方が有意に 1 試行あたりの時間が短かった ($t(319) = 2.2$，$p = .02$)．レイアウトごとの比較である B，CB，SB，DD においては有意差は見られなかった．

図 6.6 に示すように，全レイアウトにおいて適用条件よりも非適用条件の方が 1 試行あたりの時間が短い事が分かる．これは画面のループを利用した GUI 操作よりも，端末の把持姿勢の変更や指の伸ばしを用いた GUI 操作の方が，早く操作出来る事を示している．ビデオ観察を行ったところ，主に以下の箇所に時間を取られていることが見て取れた．

1. 画面移動を生じさせるまでの時間
2. 画面移動を行った後に，GUI の位置を自身の操作しやすい位置へと調整する時間

1 を挙げたのは，被験者によって画面移動を生じさせるための操作であるロール操作が上手く認識されず，そこに時間を取られているという状況が観察されたためである．この部分については認識アルゴリズムの調整によって改善出来る部分であると考える．また，このロール操作認識については操作性や手への負担度にも大きな影響を与える部分であると考える．そのためどの程度精度高くロール操作を認識出来るかを今後評価する必要がある．またその際には，モバイル端末の背面には指が常に触れている状態であるために，ロール操作の誤動作についても注意して検証する必要がある．

2 に関しては特に SB 操作時に多く時間を取られていることがビデオにて観察され，図 6.6 においても表れている事が分かる．B や CB ならば操作がタップのみで終わるので，ある程度大雑把な位置に近づけても操作が可能であった． しかし SB においては，シークバー中の「つまみ部分」を左右にスワイプ操作する必要がある．そのため，被験者はより余裕をもって親指を動かせる位置へと操作対象 GUI の位置を調整していたと考えられる．

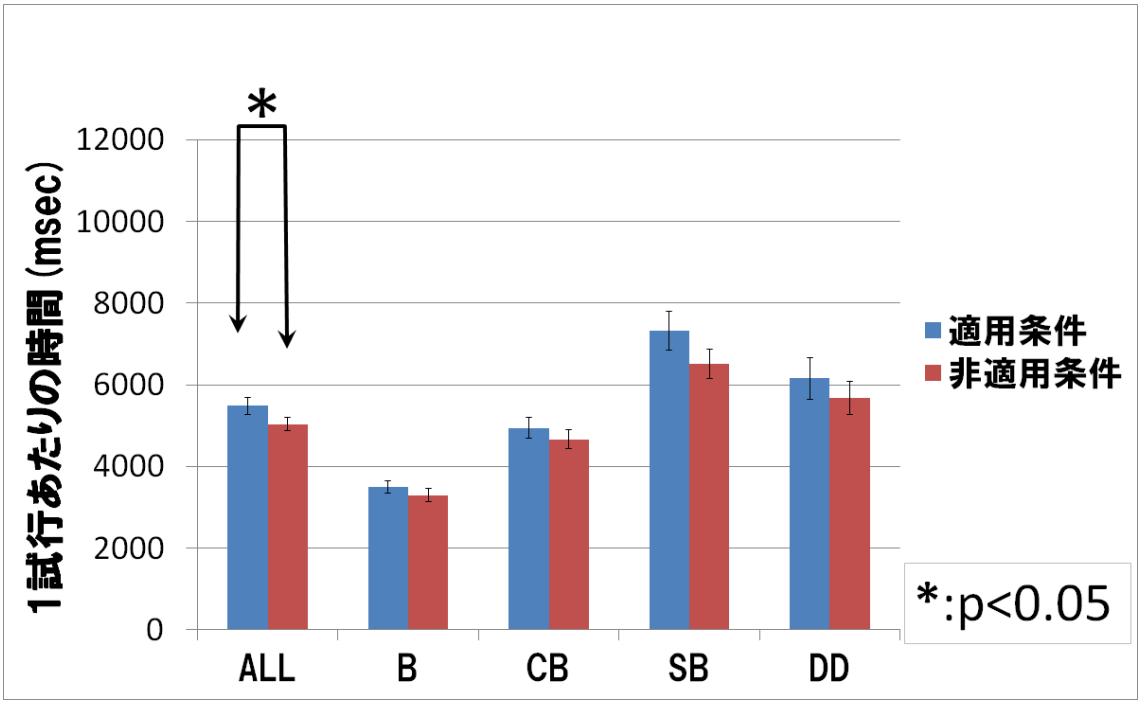

図 6.6: レイアウトごとの 1 試行あたりの時間 (msec)

第7章　今後の課題と発展

本章では，評価実験を通して明らかになった課題と，本研究の発展について述べる．

7.1　今後の課題

デバイスの軽量化，薄型化

現在のプロトタイプデバイスは一般的なスマートフォンと比較して重さ，厚さが約 2 倍となっている．評価実験に置いて，被験者 2 名から「端末が大きく，重かったため負担であった」というコメントが得られたことから，提案手法の操作性，負担度に影響を与えていると考える．現在の実装では，Android 端末を 2 台重ね合わせる事によって実装しているため，これが厚さと重さを増加させる原因となっている．しかし，技術的には市販されているゲーム機 [21] やカバー [22] のように薄く，軽量に平面実装する事が十分に可能である．

より日常利用に即したアプリケーション環境における評価実験

今回行った評価実験では,「操作が行いにくいレイアウト」を連続して操作してもらうという形の評価実験であった．しかし，実際のモバイル端末利用環境においては，ある画面では容易に片手操作できるが，ある画面では片手操作が困難になるといったように，連続してロール操作を行い続ける環境は少ない可能性がある．そのため，今後はより日常的な利用を想定したアプリケーション環境における評価実験を行う必要がある．

ロール操作認識の改善と評価

評価実験おいて，被験者 2 名から「背面指を動かす操作 (ロール操作) が負担に感じた」というコメントが得られた．また，ビデオ観察においても，被験者によってはロール操作の認識が上手くされていなかった．考えられる主な原因としては背面指のスワイプ操作を行えていないことである．これの簡単な改善案としては背面指のスワイプ動作の認識しきい値を甘くする事が考えられる．しかし，デバイスの背面には指が常に触れている事になるために，誤動作の危険性も増す事になる．そのため，ロール操作を行いやすく，かつ誤動作を少なく出来るように認識アルゴリズムを今後改善し，さらに評価実験によってその精度を評価する必要がある．

7.2 発展

今後の発展として，節 3.7.1 でも述べたように，画面操作領域の大きなモバイル端末にとらわれずに，タブレット端末といった大サイズタッチインタフェースへの適用が挙げられる．タブレット端末は画面操作領域が大きく重量もあるために，両手操作を行い，大きな画面をタッチ操作するために手を大きく動かす負担は決して少なくない．本手法を用いる事で，把持姿勢を変える事無く容易に GUI 操作が可能になり負担無くタブレット端末を操作する事が可能になると考える．

また，本研究では画面移動を発生させる操作方法としてロール操作を用いており，それにより操作と結果が自然に対応した直感的なインタラクションを実現している．しかし，このロール操作の利用の仕方によっては十分に片手操作以外の利用にも適用出来ると考える．例えば，ロール操作を画面の裏と表を入れ替える操作と考えて，メイン情報とサブ情報を切り替えるような使い方が可能であると考える．他にも，3D アプリケーション利用時に三次元的な視線移動操作にロール操作を対応付けといった利用も可能であると考える．このようににロール操作に着目する事で，モバイル端末の片手操作だけでなく，幅広い用途に利用可能であると考える．

第8章　結論

本研究ではモバイル端末の片手操作を支援するために，現行の指直接操作の操作体系を残したまま，親指可動領域外の GUI を操作可能にするインタラクション手法の開発を目的とした．そこで，画面ループを利用したモバイル端末片手操作手法を提案実装し，その手法を実現するためのハードウェアとソフトウェアの開発を行った．そして提案手法の評価を行った．

提案手法の有用性の評価実験を，実際に被験者 10 名に使用してもらう事により行った．本手法適用時の手への負担度と GUI 操作性ともに好意的な評価を得ることができ，親指可動域外の GUI を円滑に操作可能にすることができたと言える．また，インタラクションの自然さについても好意的な評価を得られたことから，操作と結果が結びついた自然なインタラクションを実現したと言える．しかし一方，プロトタイプデバイスのサイズや重量に関する問題が指摘された．また，タスク完了時間を比較すると，適用条件よりも非適用条件の方が有位に早かった．これらについては今後実装上の改善の余地がある．

指直接操作の操作体系を残したまま，様々な状況において親指可動領域外の GUI を円滑に片手操作可能にしたこと，画面ループを生じさせる端末操作としてロール操作を提案実装し，操作と結果が結びついたインタラクションが可能としたことが，既存研究には無い，本研究の貢献である．

謝辞

本研究を遂行するにあたり，本当にたくさんの皆様からご助力を頂きました．3 年間の私の研究生活は，先生方，先輩方，後輩，友人と，本当に多くの人から支えて頂いていたのだなと，こうして修士論文の謝辞の筆をとるにあたってひしひしと感じております．ご助力下さった皆様に感謝申し上げます．本当にありがとうございます．

私は NERF チームの 1 期生であり，チームの先輩もいない状態での研究生活が始まりました．そのような時に，田中先生からは個人研究について，そして研究室で学ぶということについて非常に多くのご指導を頂きました．また，田中先生には，大規模情報コンテンツ時代の高度 ICT 専門職業人育成プロジェクトを初めとして，多くの機会を与えて頂きました．積極的にそのような活動に取り組んだ結果，個人研究との両立をしながら対外発表や開発したサービスのリリースといった成果もあげることも出来ました．このような機会を多く与えてくださった田中先生のご指導のおかげで，私自身大きく影響できたと感じております．本当にありがとうございます．また，三末和男先生，高橋伸先生にも，様々な視点から，特に研究発表について多くのご助言を頂くことが出来ました．ありがとうございました．特に NERF と同室であった WAVE チームの指導教員である志築先生には様々なアドバイスを頂きました．ありがとうございました．

NERF チームと WAVE チームのメンバーには日頃から研究に関する議論にお付き合い頂きました．私が学部 4 年生の時は NERF チームの先輩がいなかったために，NERF の先輩代わりとして WAVE チームの先輩から様々な事をご指導いただきました．研究以外でも全てのチームメンバーには大変お世話になりました．皆さんがいたおかげで，毎日充実した研究室生活が送れました．ありがとうございました．

そして何よりも，精神面でも金銭面でもいつも支えてくれた両親の存在がなければ，私はこうして修士論文を執筆することはできませんでした．在学 6 年間常に支えて下さった両親に心から感謝致します，ありがとうございました．最後に，大学生活にてお世話になった全ての人に感謝の意を表し，謝辞を括りたいと思います．ありがとうございました．

参考文献

[1] ディーツーコミュニケーションズ. スマートフォン普及動向調査. http://www.d2c.co.jp/news/2012/20120418-1340.html.

[2] Amy K. Karlson and Benjamin B. Bederson. Understanding single-handed mobile device interaction. Technical report, 2006.

[3] Paul Brown. Smartphone owners want thin devices with larger displays. Technical report, Wireless Device Lab, 2012.

[4] Amy K. Karlson, Benjamin B. Bederson, and John SanGiovanni. Applens and launchtile: two designs for one-handed thumb use on small devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '05, pp. 201–210, New York, NY, USA, 2005. ACM.

[5] MindFree. Smoozy. http://smoozy.jp/.

[6] Lunascape Corporation. ilunascape. http://www.lunascape.jp/ProductSite_deploy/products/android/iLunascape/default.aspx.

[7] jig.jp. jigtwi. http://jigtwi.jp/pc/index_android.html.

[8] PSC. 扇ランチャー. https://play.google.com/store/apps/details?id=com.psc.fukumoto.ArcLauncher.

[9] Panasonic. P-07c. https://www.mydocomo.com/onlineshop/products/smart_phone/P07C.html.

[10] Daniel Vogel and Patrick Baudisch. Shift: a technique for operating pen-based interfaces using touch. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '07, pp. 657–666, New York, NY, USA, 2007. ACM.

[11] Amy K. Karlson and Benjamin B. Bederson. Thumbspace: generalized one-handed input for touchscreen-based mobile devices. In *Proc. INTERACT 2007*, pp. 324–338. Springer, 2007.

[12] Anne Roudaut, Stéphane Huot, and Eric Lecolinet. Taptap and magstick: improving one-handed target acquisition on small touch-screens. In *Proceedings of the working conference on Advanced visual interfaces*, AVI '08, pp. 146–153, New York, NY, USA, 2008. ACM.

[13] Sunjun Kim, Jihyun Yu, and Geehyuk Lee. Interaction techniques for unreachable objects on the touchscreen. In *Proceedings of the 24th Australian Computer-Human Interaction Conference*, OzCHI '12, pp. 295–298, New York, NY, USA, 2012. ACM.

[14] Craig Stewart, Eve Hoggan, Laura Haverinen, Hugues Salamin, and Giulio Jacucci. An exploration of inadvertent variations in mobile pressure input. In *Proceedings of the 14th international conference on Human-computer interaction with mobile devices and services*, MobileHCI '12, pp. 35–38, New York, NY, USA, 2012. ACM.

[15] NTT DOCOMO. Grip ui. `http://docomo-exhibition.jp/cj2012/pc/index.html`.

[16] Mayank Goel, Jacob Wobbrock, and Shwetak Patel. Gripsense: using built-in sensors to detect hand posture and pressure on commodity mobile phones. In *Proceedings of the 25th annual ACM symposium on User interface software and technology*, UIST '12, pp. 545–554, New York, NY, USA, 2012. ACM.

[17] 茂夫平岡, 一伸宮本, 潔富松. Behind touch:携帯電話のための背面・触覚操作インタフェース (入力技法)(¡特集¿「インタラクション:理論, 技術, 応用, 評価」). 情報処理学会論文誌, Vol. 44, No. 11, pp. 2520–2527, nov 2003.

[18] 清久仁河内谷, 浩石川. 超小型機器の「指一本操作」のための入力機構. 全国大会講演論文集, Vol. 54, No. 4, pp. 111–112, 1997.

[19] 佐藤克成渡部陽一. 光学式力測定手法を用いた携帯型タッチパネル端末用入力デバイスの提案. 情報処理学会シンポジウム論文集, No.3, pp. ROMBUNNO.2EXB–12, 2012.

[20] Neng-Hao Yu, Sung-Sheng Tsai, I-Chun Hsiao, Dian-Je Tsai, Meng-Han Lee, Mike Y. Chen, and Yi-Ping Hung. Clip-on gadgets: expanding multi-touch interaction area with unpowered tactile controls. In *Proceedings of the 24th annual ACM symposium on User interface software and technology*, UIST '11, pp. 367–372, New York, NY, USA, 2011. ACM.

[21] Sony Computer Entertainment Inc. Playstation vita. `http://www.jp.playstation.com/psvita/hardware/index.html?hfclick=FTR_psvitahard`.

[22] Canopy. Sensus. `https://www.getsensus.com`.

[23] Erh-li Early Shen, Sung-sheng Daniel Tsai, Hao-hua Chu, Yung-jen Jane Hsu, and Chi-wen Euro Chen. Double-side multi-touch input for mobile devices. In *CHI '09 extended abstracts on Human factors in computing systems*, CHI EA '09, pp. 4339–4344, New York, NY, USA, 2009. ACM.

[24] Katrin Wolf, Christian Müller-Tomfelde, Kelvin Cheng, and Ina Wechsung. Pinchpad: performance of touch-based gestures while grasping devices. In *Proceedings of the Sixth International Conference on Tangible, Embedded and Embodied Interaction*, TEI '12, pp. 103–110, New York, NY, USA, 2012. ACM.

[25] Xing-Dong Yang, Pourang Irani, Pierre Boulanger, and Walter Bischof. One-handed behind-the-display cursor input on mobile devices. In *CHI '09 Extended Abstracts on Human Factors in Computing Systems*, CHI EA '09, pp. 4501–4506, New York, NY, USA, 2009. ACM.

[26] Xing-Dong Yang, Edward Mak, Pourang Irani, and Walter F. Bischof. Dual-surface input: augmenting one-handed interaction with coordinated front and behind-the-screen input. In *Proceedings of the 11th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI '09, pp. 5:1–5:10, New York, NY, USA, 2009. ACM.

[27] MOTOROLA. Motorola charm. `https://motorola-global-portal.custhelp.com/app/product_page/faqs/p/30,6720,7489`.

[28] Daniel Wigdor, Clifton Forlines, Patrick Baudisch, John Barnwell, and Chia Shen. Lucid touch: a see-through mobile device. In *Proceedings of the 20th annual ACM symposium on User interface software and technology*, UIST '07, pp. 269–278, New York, NY, USA, 2007. ACM.

[29] Patrick Baudisch and Gerry Chu. Back-of-device interaction allows creating very small touch devices. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '09, pp. 1923–1932, New York, NY, USA, 2009. ACM.

[30] Tomoko Ohtani, Tomoko Hashida, Yasuaki Kakehi, and Takeshi Naemura. Comparison of front touch and back touch while using transparent double-sided touch display. In *ACM SIGGRAPH 2011 Posters*, SIGGRAPH '11, pp. 42:1–42:1, New York, NY, USA, 2011. ACM.

[31] Google. Android developers — design. `http://developer.android.com/design/index.html`.

[32] Apple. ios human interface guidelines. `http://developer.apple.com/library/ios/#documentation/UserExperience/Conceptual/MobileHIG/Introduction/Introduction.html`.

[33] Stphane Huot, Olivier Chapuis, and Pierre Dragicevic. Torusdesktop: pointing via the backdoor is sometimes shorter. In *CHI'11*, pp. 829–838, 2011.

[34] Jacob O. Wobbrock, Brad A. Myers, and Htet Htet Aung. The performance of hand postures in front- and back-of-device interaction for mobile computing. *Int. J. Hum.-Comput. Stud.*, Vol. 66, No. 12, pp. 857–875, December 2008.