2022 年度 修士論文

論文題目
非視認状態でのタッチパネル利用を想定した多様な操作インタフェースの提案

(Various operation interfaces on capacitive touch screen in non-visual state)

指導教員
舟橋　健司　准教授

名古屋工業大学大学院　工学研究科　工学専攻
情報工学系プログラム
2021 年度入学　33414035 番

加藤　馨

目 次

第 1 章　はじめに **1**

第 2 章　逆タップ **6**

第 3 章　切替スワイプ **10**
3.1　切替スワイプの判定の概要 10
3.2　切替スワイプの判定の実装 11
3.3　切替スワイプの実行の処理 13
3.4　切替スワイプの比較実験 15
3.4.1　実験準備 15
3.4.2　実験 1 の方法 16
3.4.3　実験 1 の結果と考察 16
3.4.4　実験 2 方法 19
3.4.5　実験 2 の結果と実験 1, 2 の考察 19

第 4 章　支持タップ **22**
4.1　支持タップの判定の概要 22
4.2　支持タップの判定の実装 24
4.2.1　逆タップインタフェースへの実装 26
4.2.2　通常タップと支持タップを併用できる実装 28
4.3　支持タップの比較実験 28
4.3.1　実験準備 28
4.3.2　実験の方法 30
4.3.3　実験結果と考察 33

第 5 章　むすび **35**

謝辞 **37**

参考文献 **38**

発表論文リスト **40**

第1章　はじめに

現在, スマートフォンやタブレットなどのタッチパネル式のデバイスが普及している. 総務省の令和 3 年版情報通信白書によると, 2020 年の世帯におけるスマートフォンの保有割合は 86.8%でパソコンの 70.1%を上回っており, 年々増加傾向にある [1]. 銀行の ATM や鉄道駅の自動券売機などの公共性の高い機械へのタッチパネル式デバイスの導入が進み,　カーナビゲーションへの適用も多くなり, さらに教育現場でのスマートフォンやタブレット PC の利用が推進されるなど, 我々の生活に深く関わってきている [2][3].

タッチパネルの操作検出方式には大きく分けて感圧（加圧）式と静電容量式の 2 つがある. 感圧式は, フィルム面とガラス面に 2 枚の透明電極膜を少しだけ隙間を設けて貼り付けてある. フィルム面の透明電極膜が押されてガラス面の透明電極膜に接触し電圧が変動することで, その接点の位置を検出する. 身近なものでは小売業等の POS 端末（キャッシュレジスター）や, 任天堂株式会社のゲーム機である「Nintendo DS」などに搭載されている. 感圧式の長所は, 構造が単純であるため低コストで製造でき, 消費電力が比較的抑えられている点である. また, 入力がフィルム面への圧力なため, 指だけではなくペンや手袋をしたままの指でも操作が可能である. 短所は, フィルムと 2 枚の透明電極膜により画面の表示品質が低下する点, 耐久性や耐衝撃性が比較的低い点や, 画面サイズが大きくなるほど接点の検出精度が低くなる点である. 一方静電容量式は, ガラス面に電極として透明導電膜である ITO（Indium Tin Oxide）膜が貼り付けてある. 指がこの ITO 膜に触れる, もしくは近づくことで発生する微弱な電流, つまり静電容量の変化をセンサーが感知することで, その点の位置を検出する. この静電容量式のタッチパネルは Apple 社の「iPhone」に搭載されたことで急速に普及し, 近年のスマートフォンやタブレットはほぼ静電容量式タッチパネルが採用されている. 長所は, 水やホコリには反応せず耐久性や耐衝撃性

が高い点や, 感圧式と比べて画面の表示品質が高い点である. また, 正確な多点検出（マルチタッチ）が可能であり, ピンチイン, アウトや回転などの 2 本以上の指で行うジェスチャー操作にも適している. 短所は, 指や静電容量式に反応する専用のタッチペンでしか操作ができないため普通のペンや手袋をしたままの指で操作できない点や, 近くに金属筐体がある場合にその影響を受けやすい点である. また, Apple 社の iPhone6s から iPhoneX まで搭載されていた 3D Touch 機能は画面を押し込む操作が可能である [4]. 3D Touch には画面を軽く押し込む Peek 操作と強く押し込む Pop 操作の 2 つのアクションが用意されており, この 2 つの操作を組み合わせることでタップの回数を減らしスマートフォンとのインタラクションをよりシンプルで素早いものにしている. なお, この 3D Touch 機能を利用するには特別なハードウェアが必要である.

タッチパネル式デバイスは通常, 液晶画面に表示されているボタンを指やペンなどで触れて, その後その指やペンを離す（タップ）ことで操作される. したがってこれらのデバイスにはパソコンや従来型の携帯電話に搭載されているキーボードや通話ボタンのような物理的なボタンは搭載されていない場合が多い. 物理的なボタンが搭載されているデバイスは, そのボタンそのものやそれに付いている突起（例えばキーボードの「F」, 「J」キーや, 電話の「5」）, ボタンとボタンとの境目などを見ずに触って認識, 区別することができる. しかし, 物理的なボタンが搭載されていないタッチパネル式デバイスでは, 凹凸のない画面を触ってもそれがどのボタンなのか, そもそも触れている箇所がボタンなのか認識, 区別することができない. このように触覚的な手掛かりがないため画面を視認せずに思い通りの操作をすることは困難であり, 画面を一瞥しただけで以後画面を見ずに操作をすることも難しい. 厚生労働省による平成 28 年生活のしづらさなどに関する調査によると, 日本全国で在宅の視覚障害者は 31.2 万人おり [5], その中でもスマートフォンを利用する人は増えている [6][7]. 画面を視認できない視覚障害者にとってスマートフォンやタブレットの利用は困難であると考えられ, 視覚障害者が有効に活用できるスマートフォンやタブレットを利用した支援システムに関する研究も行われている [8][9][10]. しかし視覚障害者にとって使いづらい点はいまだ多くあるであろうと考えられる. 例えば全盲者の場合は音声出力機能による補助があったとしても触覚的特徴がないためタッ

チパネル操作が難しいだろう. また, 全盲者及びロービジョン者はタッチパネル上の意図しない箇所をタッチしてしまうことがあるかもしれない.

そこで当研究室では, 広く普及している静電容量式タッチパネルを対象に, 前述のような問題を解決する新しい操作インタフェースを提案している. 先行研究では, 画面を見なくても操作可能な「逆タップ（iTap：inverse tap)」, 逆タップを連続で行う「ダブル逆タップ」という新しい操作方法を提案している [11]. 加えて逆タップを 2 本の指で行う「2 本指逆タップ」も提案している [12][13]. 指が画面から離れている場合だけでなく指が画面に触れている状態も基本的に無操作状態であると考え, 指が画面に触れている状態から操作箇所で離し, 再度同じ箇所に触れることでタップと判定する. なお, 図 1.1 のようにあらかじめ画面上に物理的な突起物を張り付けておく. また指を画面に触れてすぐに離した場合や, 触れてその後触れたまま動かした場合は通常のタップやスワイプ操作と判定しない. これによりユーザは画面を見ずとも突起物の触覚的特徴で操作箇所を判別しタップ操作を行うことができる. しかし, 物理的に画面に張り付けられている突起物の場所や形状はユーザが記憶しておく必要がある. また, この突起物の数は操作画面の操作箇所の数と同じ数を想定しているため, 複雑なアプリケーションの実現のために操作箇所を増やし突起物の数も増やすことはユーザにとって負担になるであろう. そこで本論文ではまず, スワイプ操作を想定した新たな操作方法「切替スワイプ (sSwipe：switch swipe)」を提案する. 通常のタッチパネル式デバイスにおけるスワイプ操作は, 画面上に触れている指を上下左右に滑らせる操作方法であり, 基本的に画面上のどこであっても操作可能である. このように画面レイアウトに依存しない操作方法を提案することで, 画面上の操作箇所の数が限られる逆タップインタフェースの補助をする.

ところで, 前述したように逆タップを利用すればタッチパネル式デバイスを画面を見ずに操作することが可能である. 晴眼者でも運転中のタッチパネル式空調操作パネルを, 一瞥したのちに容易に操作できるだろう. しかし, 逆タップは操作箇所を把握する際に指で支持することができるが, 揺れている車内においては指を一瞬離して再度触れる時に操作したい箇所からずれてしまうかもしれない. 走行中のタッチパネル利用を補助するユーザインタフェースとして A New Car UI というアイデアがある [14]. 指の本数や指同士の間隔を変えることで違った操作を可能にして, 画面

を見ずともエアコンの温度や音量の調節を行うことを容易にしている. しかしこのアイデアでも, 一次的にすべての指を離す必要がある. そこで本論文ではまた, 指での支持を前提とした新しい操作方法「支持タップ（hTap：hold tap）」も提案する. 1 本の指で操作箇所の把握と支持をし, もう 1 本の指でタップ操作を行う支持タップを提案することで, 揺れている状態かつ見えていない状態でのタッチパネル利用をより容易にする.

以下, 第 2 章では先行研究で提案されている逆タップの概略を説明をする. 第 3 章では切替スワイプのアルゴリズムを提案するとともに, その実装方法, 全盲状態を想定した逆タップ操作のみ可能なインタフェースとの比較実験について説明する. 第 4 章では支持タップのアルゴリズムを提案するとともに, その実装方法, 全盲状態かつ揺れている状態を想定した通常タップ及び逆タップとの比較実験について説明する. 第 5 章ではまとめと今後の課題について述べる.

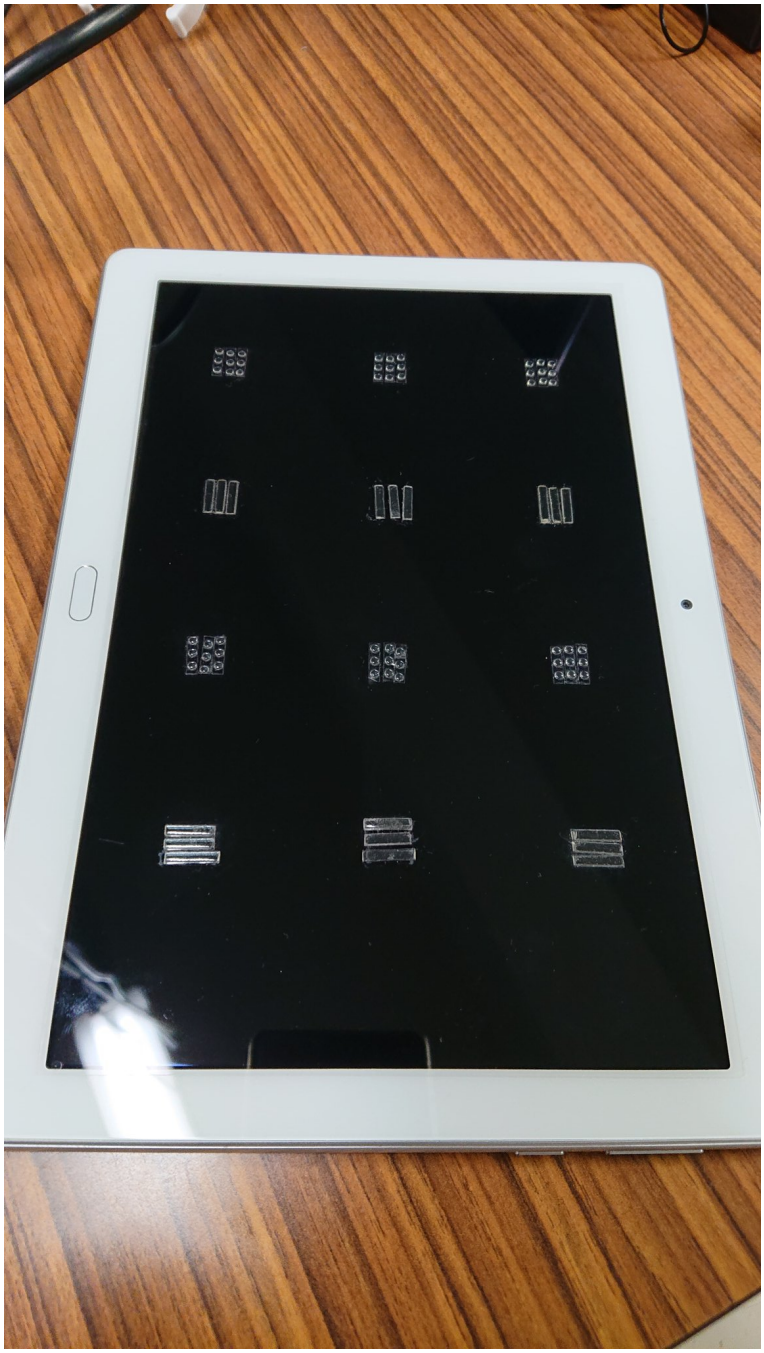


図 1.1: 突起物のついたタッチパネル

第2章　逆タップ

本章では, 先行研究 [11] で提案している逆タップの概略について説明する. タッチパネル式デバイスは基本的に画面上の操作箇所に指を触れてその後離す（タップ）ことで操作される. そのため, パソコンのキーボードや従来型の携帯電話の通話ボタンのような物理的なボタンが備わっていない場合が多く, 画面を見ずに操作することや, 画面を一瞥したのちに画面を見ずに操作することが困難である. そこで逆タップという新しい操作方法を提案している. まず前提として, タッチパネル上に物理的に点字のような突起物を付けておく. また, 指が画面に触れている状態は無操作状態と認識し, 指が画面に触れてすぐに離した場合や, 触れてその後触れたまま動かした場合はいわゆる通常のタップやスワイプ操作と判定しないこととする. その上でユーザが以下の動作を行った時に逆タップと判定する.

1. 指で画面に触れる
2. 画面に触れたまま指を動かし, 操作したい箇所を見つける
3. タップしたい箇所で指を画面から離す
4. 一定時間内に画面の同じ箇所に触れる

画面に指を触れてから離す通常のタップ操作とは逆に, 画面に触れている指を離してから再度触れた場合にタップと判定するため, 逆タップと呼ぶ. 先の前提とこの操作方法により, 画面を見ずに手探りで操作したい場所を探し, 突起物の触覚的特徴で操作箇所を判別し操作することが出来る. また, 合わせてダブル逆タップ操作についても考慮している. 逆タップを短時間に 2 連続で行うことでダブル逆タップの操作と判定する. さらに 2 本指逆タップ操作についても提案している [12][13]. 逆タップを 2 本の指で行うことで 2 本指逆タップの操作と判定する. 図 2.1 に逆タップ操作のアルゴリズムフローを示す. 実装には AndroidStudio の onTouchEvent を用いてい

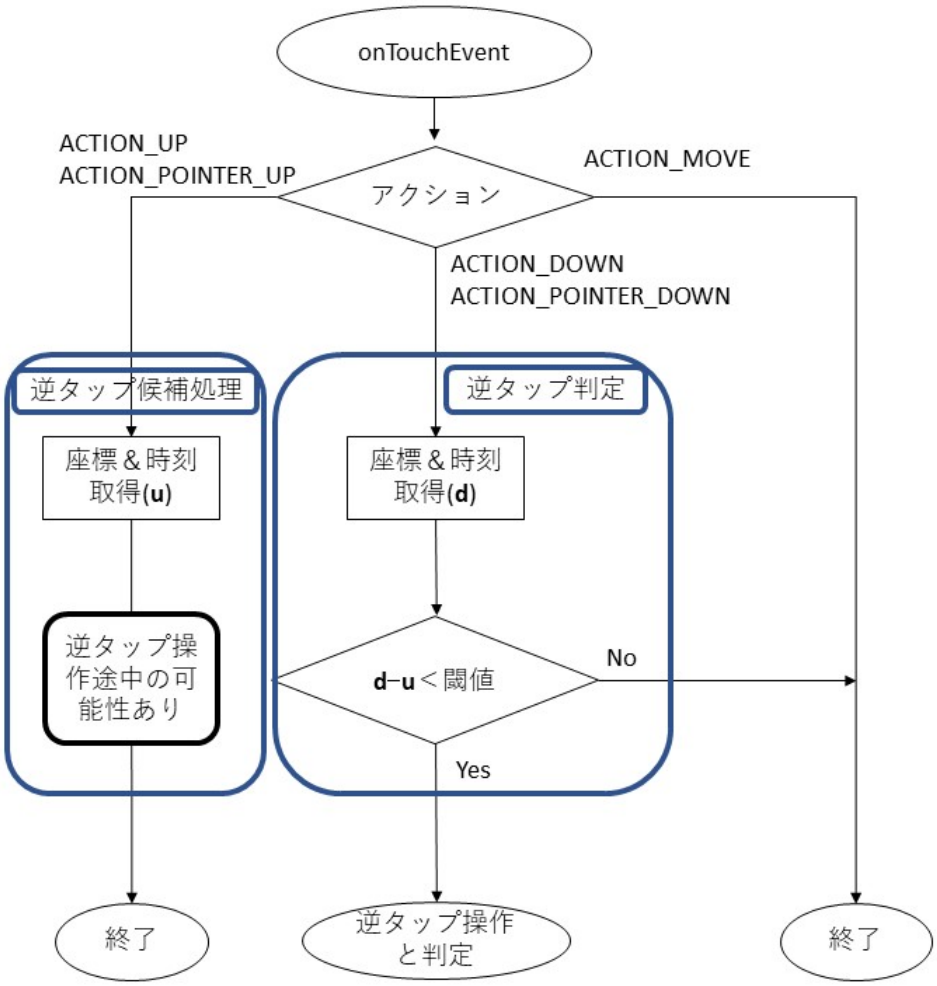


図 2.1: 逆タップのフローチャート (文献 [11] より引用)

る. イベントには種類があり, これをアクションと呼ぶ. アクションの種類には以下のものがある.

- ACTION_DOWN: 1 箇所もタッチしていない状態でタッチした時のイベント
- ACTION_UP: 1 本の指でタッチしている状態でタッチしなくなった時のイベント
- ACTION_POINTER_DOWN: すでに 1 箇所以上でタッチしている状態で追加タッチした時のイベント
- ACTION_POINTER_UP: 複数の指でタッチしている状態で, そのうち 1 つの指を離した時のイベント
- ACTION_MOVE: タッチ中に指を動かしたときのイベント

ACTION_UP, ACTION_POINRER_UP は画面から指が離れた時のアクションであり, 大域変数 $\mathbf{u}$ に指が離れた座標と時刻を保存する. ACTION_DOWN, ACTION_POINTER_DOWN は指が画面に触れた時のアクションであり, 大域変数 $\mathbf{d}$ に指が触れた座標と時刻を保存する. ACTION_MOVE は指が画面上を動いた時のアクションであるため, なにも行わない. 画面に指が触れた時の座標と時刻が, 前回離した時の座標と時刻と近ければ, 逆タップ操作と判定する. 逆タップの詳細, ダブル逆タップの実装方法については文献 [11] を, 2 本指逆タップの実装方法については文献 [12], [13] を参照されたい.

文献 [11] で行われた全盲者を想定した通常のタップとの比較実験では, 被験者がアイマスクをした状態での通常タップ操作は間違った操作箇所での操作が多く見られたのに対し, 逆タップ操作では間違った操作箇所での操作はなかった. よって全盲者を想定した場合に通常のタップ操作よりも逆タップ操作の方が正しく操作できることが確認できた. また文献 [12][13] で行われた全盲状態を想定した通常のタップと 2 本指逆タップとの比較実験においても, 2 本指逆タップの方が正しい操作箇所に対して操作が可能であることが確認できた. しかし, 全ての操作を 2 本指逆タップで行うように指示をしてあり, 被験者も 2 本指逆タップを行ったと申告しているにも関わらず実際には 1.5 割程度の操作は 1 本指の逆タップと判定されていた. これにはタッチ

パネルに張り付けた突起物の上は指の接触判定精度が低いことや, 被験者の指の間隔が狭いとタッチパネルの分解能によって 1 点と認識されてしまうことなどのハードウェア的な問題もあるが, 逆タップ（2 本指逆タップ）という通常のタップとは異なった操作方法を正しく行うためには多少の慣れが必要があることも示唆される.

第3章　切替スワイプ

前章で説明した逆タップ, ダブル逆タップ, 2 本指逆タップは操作箇所 1 箇所のみを対象とする操作である. 操作箇所には触覚的特徴として点字のような突起物が貼り付けてあり, 全盲状態のユーザはこの触覚的特徴を手探りで探し, 操作箇所を判別することが可能である. そのためユーザは, この突起物の画面上の位置や形状を記憶しておく必要がある. この突起物の数は画面上の操作箇所の数と同じ数を想定しているため, 1 つの画面上でできる操作の数を増やすために操作箇所を増やすことは同時に突起物の数を増やすこととなる. それはユーザにとっての負担になるだろう. ところで, タッチパネル式デバイスにおけるスワイプ操作は, 画面に指を触れた状態でその指を上下左右に滑らせる操作方法であり, すなわち基本的に画面上のどこであっても操作可能なため画面上のレイアウトに依存しない操作方法である. しかし, 画面に指が触れている状態も基本的に無操作状態であること前提とする逆タップ（ダブル逆タップ, 2 本指逆タップを含む）インタフェースでは, 通常のスワイプをそのまま実装することはできない. そこで本論文ではまず, 新しい方法として切替スワイプを提案する [15]. 具体的には, 操作を開始する点で触れている指を他の指に入れ替え（切り替え）, 入れ替えた指で従来のスワイプ操作をし, 操作を終了する点で入れ替えていた指を元に戻すことで, 逆タップ操作とスワイプ相当の操作を同時に実現する. 言い換えれば, 触れている指を他の指に切り替えた場合に従来のスワイプ操作を許容することで, 逆タップインタフェースでスワイプ操作に相当する操作を可能にする.

3.1　切替スワイプの判定の概要

ユーザが, 以下の動作を行った時に切替スワイプ操作と判定する.（図 3.1）

0. 1 本の指で画面に触れている＜無操作状態＞（図 3.1（a））

1. ＜スワイプ開始＞

   1.1. 2 本目の指を画面上の 1 本目の指の近傍に触れる（図 3.1（b））

   1.2. 一定時間以内に 1 本目の指を離す（図 3.1（c））

2. 2 本目の指を操作したい方向へ画面に触れたまま動かす＜スワイプ動作＞（図 3.1（d））

3. ＜スワイプ終了＞

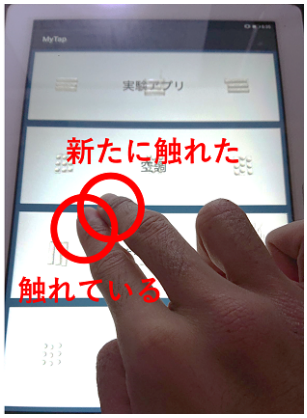
   3.1. 1 本目の指を画面上の 2 本目の指の近傍に触れる（図 3.1（e））

   3.2. 一定時間以内に 2 本目の指を離す（図 3.1（f））

3.2　切替スワイプの判定の実装

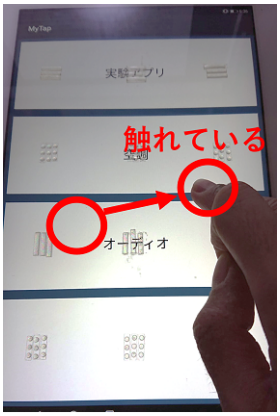
本節では, 切替スワイプの判定の実装について説明する．実装には AndroidStudio の onTouchEvent を用いる．onTouchEvent については第 2 章で解説した通りである．図 3.2 に切替スワイプのアルゴリズムフローを示す．なお, 逆タップ候補処理, および逆タップ判定処理については第 2 章および文献 [11] を, 2 本指逆タップ候補, および 1 本指逆タップ候補の処理については文献 [12][13] を参照されたい．このフローは先行研究で提案している逆タップ（ダブル逆タップ, 2 本指逆タップ含む）操作も考慮されている．フロー内で時刻と座標を格納しておく **u, d** は大域変数とし, 初期値はともに $-\infty$（x, y 座標, および時刻をプログラムにおける最小値）としておく．定数である **thershold** は閾値であり, 事前に値を決めておく．また **touchedPointCount** は画面に触れている指の数を示す．画面に触れている指の数は onTouchEvent の getPointerCount() 関数で取得可能である．まず, タッチパネル上で指が触れる (ACTION_DOWN または ACTION_POINTER_DOWN), または指が離れる (ACTION_UP または ACTION_POINTER_UP), 指が触れながら動く (ACTION_MOVE) のいずれかのアクションが起こると, onTouchEvent が開始され, アクションの判定をする．ここで ACTION_DOWN, ACTION_POINTER_DOWN 時には, 大域変数 **d** に現在指が触れた時点の座標と時刻が格納される．ACTION_UP,

(a) 1 本の指で触れている

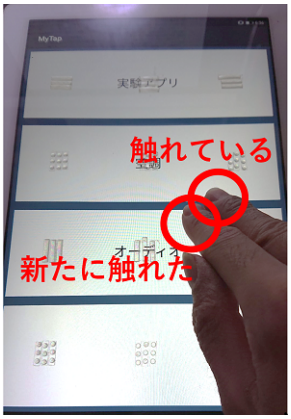


(b) 2 本目の指を 1 本目の指の近傍に触れる

(c) 一定時間以内に 1 本目の指を離す



(d) スワイプ動作



(e) 1 本目の指を 2 本目の指の近傍に触れる

(f) 一定時間以内に 2 本目の指を離す

図 3.1: 切替スワイプ操作

ACTION_POINTER_UP 時には, 大域変数 **u** に現在指が離れた時点の座標と時刻が格納される.

切替スワイプの候補, および判定処理には指が画面から離れた（ACTION_UP, ACTION_POINTER_UP）時に移行する. ただし, フロー中の逆タップ候補処理で 2 本指逆タップ候補としている場合は, 2 回連続で onTouchEvent において画面から指が離れたことを意味し, 切替スワイプ候補となり得ないのでそのまま終了する. 大域変数 **swipeflag** はその時点において切替スワイプ候補であるか否かを示す. 分岐条件「$\mathbf{u} - \mathbf{d} < \mathbf{threshold}$ & $\mathbf{u} \neq \mathbf{d}$ & $\mathbf{touchedPointCount} > 1$」は, 前回指が触れた時刻と座標と, 今回指が離れた時刻と座標の差が閾値未満であり, また触れた指と離れた指が別の指であり, かつ画面に触れられている指の数が 2 本以上であることを示している. 言い換えると, 元々指が触れている近傍に別の指が触れられ, その直後に元々触れられていた指が離れたという意味である. これが満たされない場合は処理を終了する. これが満たされる場合は, 次の分岐条件「$\mathbf{swipeflag} = 1$」で切替スワイプ候補であるかどうかを判別する. 「$\mathbf{swipeflag} = 1$」ではない, つまり切替スワイプ候補ではない場合は, 「$\mathbf{swipeflag} := 1$」と切替スワイプ候補であるとして処理を終了する. 「$\mathbf{swipeflag} = 1$」である, つまり切替スワイプ候補である場合は, 「$\mathbf{swipeflag} := 0$」とし, 切替スワイプの実行処理, すなわちアプリケーションなどでの画面や音量の変更処理へと移行する.

3.3　切替スワイプの実行の処理

通常のスワイプ操作は, 主に指を滑らせた方向や距離によって異なった実際の処理が行われる. 切替スワイプでは前述した切替スワイプ候補・判定処理内の分岐条件「$\mathbf{u} - \mathbf{d} < \mathbf{threshold}$ & $\mathbf{u} \neq \mathbf{d}$ & $\mathbf{touchedPointCount} > 1$」が満たされた時点での大域変数 **d** の座標が, 切替スワイプを開始した位置の座標か, あるいは終了した位置の座標となる. 次の分岐条件「$\mathbf{swipeflag} = 1$」が満たされず, 「$\mathbf{swipeflag} := 1$」とした場合の大域変数 **d** の座標は, 切替スワイプを開始した位置の座標となる. 分岐条件が満たされ, 「$\mathbf{swipeflag} := 0$」とした場合の大域変数 **d** の座標は, 切替スワイプを終了した位置の座標となる. これらの座標を記録し, 開始した位置の座標と終了

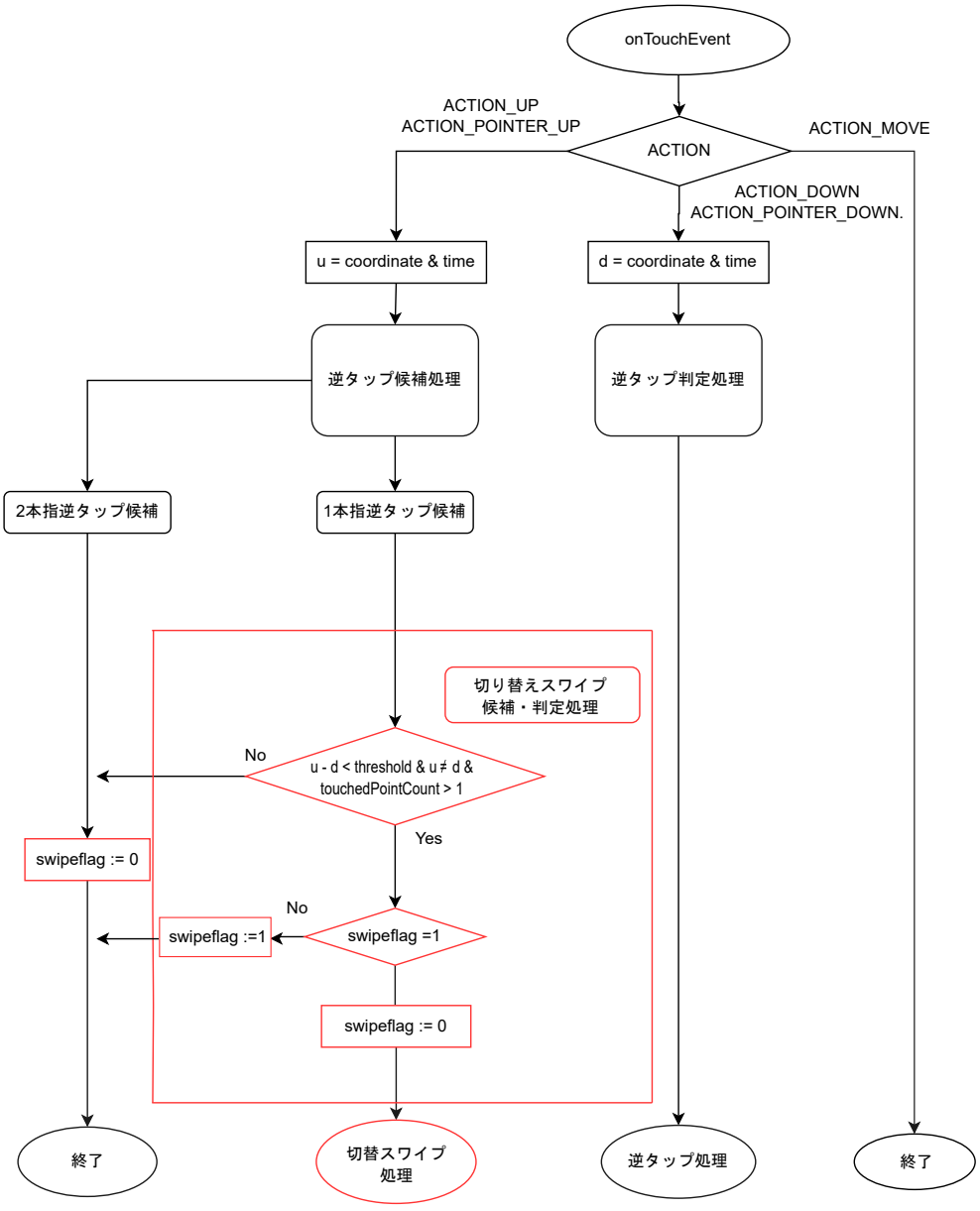


図 3.2: 切替スワイプのアルゴリズムフロー

した位置の座標で計算をすることで, 切替スワイプをした方向, 距離を把握することができる. また, 切替スワイプ候補（**swipeflag** = 1）であり, かつ画面に触れている指が動いている場合（スワイプ動作中）に, その動いている指の座標と切替スワイプを開始した時の座標を ACTION_MOVE 時の処理として計算することで, 切替スワイプ操作の方向や距離に応じたアプリケーションなどでの画面や音量の変更処理を実現する.

3.4　切替スワイプの比較実験

提案する切替スワイプにより, 非視認状態でも操作可能なインタフェースのユーザビリティ向上を確認する. 全盲状態でのオーディオパネルの操作を想定した, 逆タップ操作と切替スワイプ操作との比較実験を行った.

3.4.1　実験準備

まず, 電話やテンキーを参考に画面を図 3.3 に示すようなボタン領域に分けた. また各ボタンに点字状の突起物を張り付けた. この突起物の触覚的特徴により被験者は目が見えていない状態でも手探りで自分が画面上のどこを触っているか判別できる. 実験はオーディオパネルを模した画面での曲の変更や音量の変更を想定している. 曲は 3 曲, 音量は 15 段階で変更が可能である. 曲と音量の変更は, 逆タップ操作の場合は対応するボタンを逆タップすることで, 切替スワイプ操作の場合は上下左右方向（画面上であれば操作する場所は問わない）へ切替スワイプすることで行うことができる. 画面 2 行目の 1, 3 列目のボタンを逆タップ操作により, もしくは左右方向への切替スワイプ操作により曲を変更することができる. また, 4 行目の 1, 3 列目のボタンを逆タップ操作により, もしくは上下方向へ切替スワイプ操作により音量を変更することができる. 実験に用いた 10.1 インチのタブレットでは, 左右方向へ 64mm 以上切替スワイプ操作をすることで曲が変更され, 上下方向へ 16mm 切替スワイプ操作をするごとに音量が 1 段階変わる. この実験では曲や音量の変更に伴い実際にタブレットから音声が流れるため, 被験者は目が見えていない状態であっても操作ができたことを確認できる. 実験を始める前に被験者に逆タップ操作と切

替スワイプ操作の操作方法を説明し, タブレットの操作練習時間を設けた. これは逆タップ操作と切替スワイプ操作に慣れてもらい, またボタンの配置と突起物の配置, 曲の番号とその曲を記憶してもらうことを目的としている.

3.4.2　実験 1 の方法

被験者は 20 代男性 7 名であり, その全てが晴眼者である. この実験は全盲者を想定しているため, 被験者にはアイマスクをしてもらった. 実験 1 の手順を以下に示す.

1. 被験者はアイマスクを着用する
2. 被験者は以下の手順を 1 分間できる限り繰り返す
   (a) タブレット操作タスク (曲, 音量の変更) を受ける
   (b) タスクを実行
   (c) タスクを実行できたと感じたら合図
3. タスク通り実行できた回数を記録する

この手順を, タブレット操作タスクを逆タップ操作のみで行う場合と切替スワイプ操作のみで行う場合との 2 回行った. タブレット操作タスクの音量の変更は, 15 段階の内の具体的な値にしてもらうのではなく, 大きく, 小さくのように抽象的な指示をしている. またこの手順を終えた後に被験者には自由記述のアンケートに回答してもらった. 図 3.4 に実験の様子を示す.

3.4.3　実験 1 の結果と考察

タブレット操作タスクを逆タップのみで行った場合と, 切替スワイプ操作のみで行った場合の 1 分間でのタブレット操作タスクの達成回数を表 3.1 に示す. 表 3.1 を見ると, 切替スワイプ操作のみで操作した場合の方が 1 分間でのタスク達成回数の平均が多い結果となった. しかし, それは約 0.14 回とわずかなものであり達成回数が大きく向上したとは言えない結果となった. また, 自由記述のアンケートの結果を以下に示す.

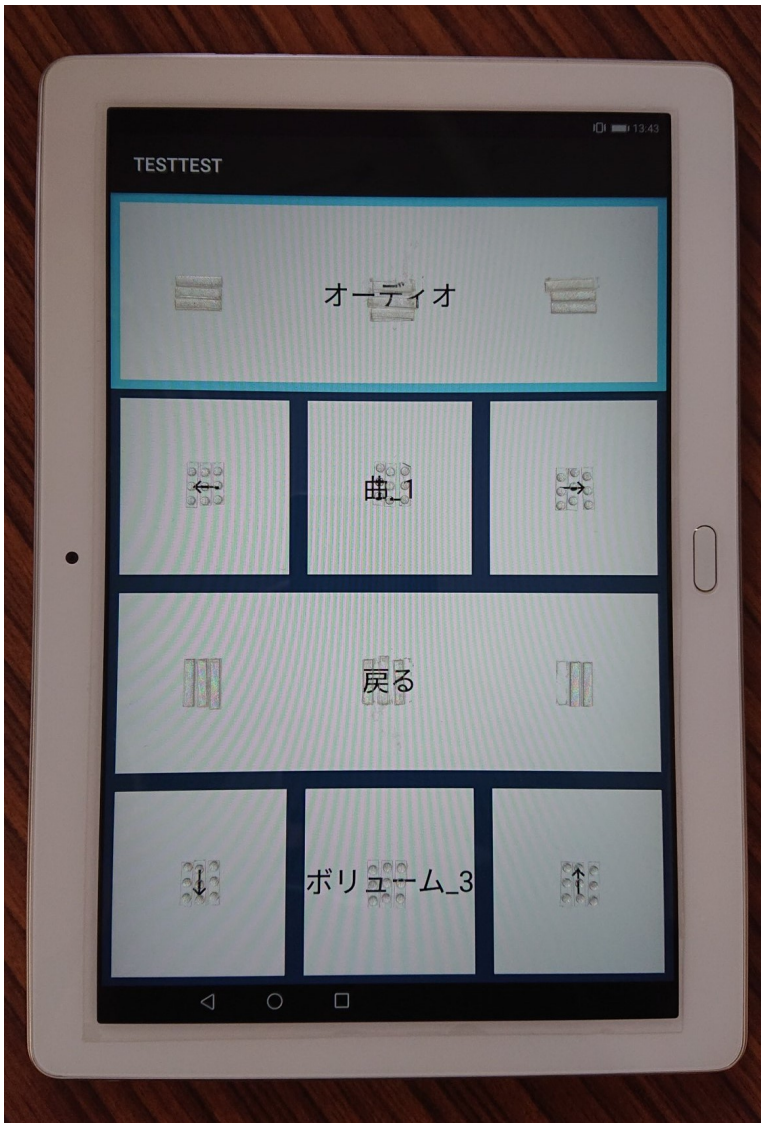


図 3.3: オーディオパネルを模した操作画面

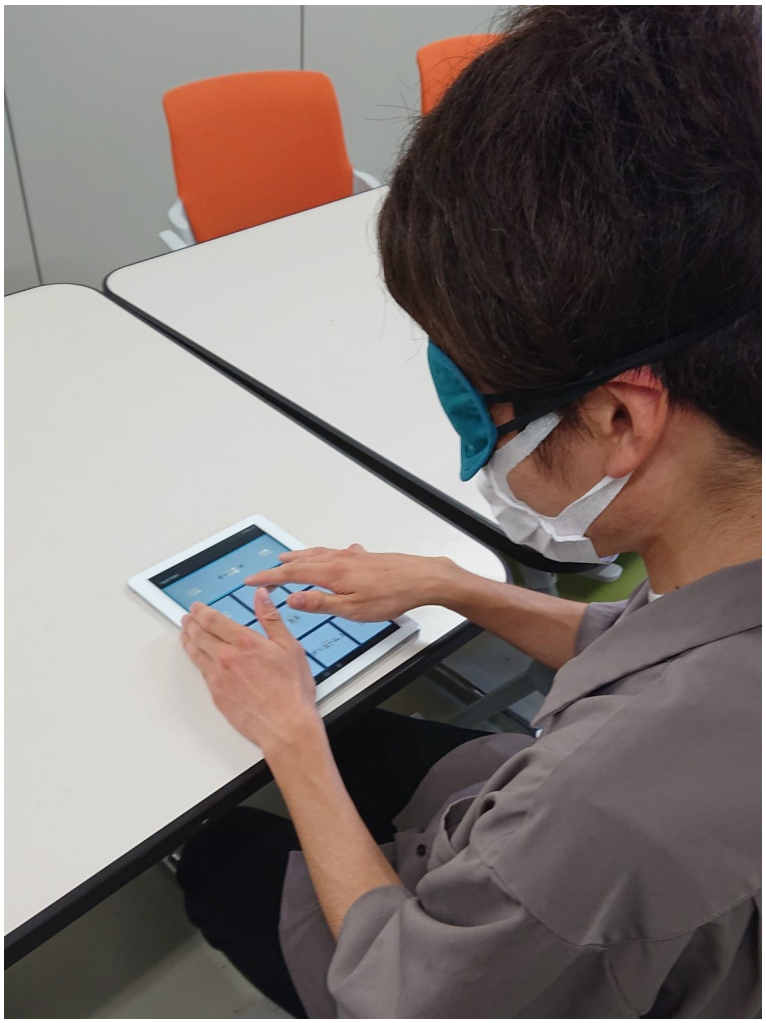


図 3.4: 切替スワイプ実験の様子

- 音量の操作は切替スワイプ操作の方がやりやすい
- 曲の変更は逆タップ操作の方がやりやすい

自由記述のアンケートの結果によると, 音量の操作は切替スワイプ操作の方がやりやすく, 曲の変更は逆タップ操作の方がやりやすいことが分かった. この実験のアプリケーションでは曲数は 3 曲, 音量は 15 段階で変更が可能である. よって変更する量が大きい程, 逆タップ操作では操作回数が多くなる. しかし, 切替スワイプ操作では画面上の指を大きく滑らせることで変更できるため, より優位なのではないかと考えられる.

3.4.4　実験 2 方法

実験 1 の結果を受けて, 3.4.2 項の実験手順におけるタブレット操作タスクを音量の変更のみに限定した実験 2 を以下の手順で行った.

1. 被験者はアイマスクを着用
2. 被験者は以下の手順を 10 回繰り返す
   (a) タブレット操作タスク (音量を上げる, 下げる) を受ける
   (b) タスクを実行
   (c) タスクを実行できたと感じたら合図
3. 10 回達成するまでの時間を記録

3.4.5　実験 2 の結果と実験 1, 2 の考察

タブレット操作タスク（音量の変更）を逆タップのみで行った場合と, 切替スワイプ操作のみで行った場合の 10 回タブレット操作タスクを達成するまでに要した時間を表 3.2 に示す. 表 3.2 を見ると, 切替スワイプ操作のみで操作した場合の方が逆タップ操作のみの場合より平均で約 9 秒短かった. よって変更する量が多い場合, 例えば実験で行ったように音量を変更する場合や, シークバー等を操作する場合においては逆タップ操作よりも提案する切替スワイプ操作の方が優位であることが確認でき

た. また, この実験を被験者はアイマスクを着けて全盲状態で行っていたが, 操作のミスなどは見受けられなかったため目が見えていない状態でも正しく操作可能であることも確認できた.

表 3.1: 実験 1（曲・音量の変更）の結果

被験者	逆タップのみ	切替スワイプのみ
A	7 回	6 回
B	8 回	8 回
C	7 回	7 回
D	7 回	9 回
E	6 回	6 回
F	8 回	7 回
G	8 回	9 回
平均	7.29 回	7.43 回

表 3.2: 実験 2（音量の変更のみ）の結果

被験者	逆タップのみ	切替スワイプのみ
A	64.7 秒	53.9 秒
B	59.6 秒	53.4 秒
C	61.8 秒	54 秒
D	63.5 秒	52.7 秒
E	67.8 秒	56.4 秒
F	65.3 秒	53.3 秒
G	58.1 秒	54.5 秒
平均	63 秒	54 秒

第4章　支持タップ

先行研究 [11] で提案している逆タップを利用すればタッチパネル式デバイスを画面を見ずに操作することが可能である. 晴眼者でも, 運転中にタッチパネル式空調操作パネルなどを一瞥したのちに容易に操作できるだろう. そして, 逆タップは操作箇所を把握する際に指で支持をすることができる. しかし, 揺れている車内においては指を一瞬離して再度触れる時に操作したい箇所からずれてしまうかもしれない. 物理的なボタンであれば, 操作する指以外の指でボタン以外の場所に触れて手を支持することで, あるいは一度ボタンの上に指を置いてからボタンを押すことで, 揺れている場合でも操作したいボタンを操作することができる. そこで本論文ではまた, 指での支持を前提とした新しい操作方法として支持タップを提案する. 支持タップは, まず画面に触れている 1 本の指で操作箇所の把握と支持をし, 他の指でその近傍をタップする操作方法である. 指が触れている状態は基本的に無操作状態とする本研究のインタフェースにおいて, 既に指が触れている場所の近傍へのタップ（逆タップではなく通常のタップ）を支持タップと判定する. 既に触れている指が支えとなり揺れている場合にも操作が容易になる.

4.1　支持タップの判定の概要

ユーザが, 以下の動作を行った時に支持タップ操作と判定する.（図 4.1）

1. 指で画面に触れる（図 4.1（a））
2. 画面に触れたまま指を動かし, 操作したい箇所を見つける（図 4.1（b））
3. 画面に触れている指の近傍（すなわち, 操作したい箇所）に指を触れる（図 4.1（c））
4. 一定時間以内に 3 で触れた指を離す（図 4.1（d））

(a) 指で触れる

(b) 指を動かし操作箇所を見つける

(c) 他の指で触れている指の近傍に触れる

(d) 直近で触れた指を離す

図 4.1: 支持タップ操作

4.2　支持タップの判定の実装

本節では, 支持タップの判定の実装について説明する. 実装には切替スワイプと同様に AndroidStudio の onTouchEvent を用いる. onTouchEvent については第 2 章で説明した通りである. 図 4.2 に支持タップのアルゴリズムフローを示す. なお, このフローでは説明を簡単にするために, 先行研究で提案している逆タップ操作, 前章で提案した切替スワイプ操作は考慮していない. フロー内で時刻と座標を格納しておく **u, d, d2, m** は大域変数とし, 初期値はともに $-\infty$（x, y 座標, および時刻をプログラムにおける最小値）としておく. なお, これらの大域変数に「[**c**]」とついている場合はその変数の座標要素を表す. 大域変数 **holdflag** はその時点において支持タップ候補であるか否かを示し, 初期値は 0 である. 定数である **threshold**, **threshold2** は閾値であり, 事前に値を決めておく. また **touchedPointCount** は画面に触れている指の数を示す. 前章と同様にタッチパネル上でいずれかのアクションが起こると, onTouchevent が開始され, アクションの判定をする. ここで ACTION_DOWN, ACTION_POINTER_DOWN 時には, **d** の時刻と座標が **d2** にバックアップされ, **d** に現在指が触れた時点の座標と時刻が格納される. つまり, このシステムでは過去 2 回の指がタッチパネルに触れた時点のその座標と時刻を保存している. ACTION_UP, ACTION_POINTER_UP 時には, **u** に現在指が離れた時点の座標と時刻が格納される. ACTION_MOVE 時には, **m** に現在指がタッチパネル上を動いた時点の座標と時刻が格納される.

まず, 支持タップの候補処理には指が画面に触れた（ACTION_DOWN, ACTION_POINTER_DOWN）時に移行する. 分岐条件「(**d**[**c**] - **d2**[**c**] $<$ **threshold** $\|$ **d**[**c**] - **m**[**c**] $<$ **threshold**) && **touchedPointCount** $>$ 1」は, 前々回指が触れた座標と今回指が触れた座標との差が閾値未満, もしくは指が画面上を動いて停止した座標と今回指が触れた座標との差が閾値未満であり, かつ画面上に触れられている指が 2 本以上であったことを示す. 言い換えると, 既に指が触れられている近傍に指が触れたという意味である. これが満たされない場合は「**holdflag** := 0」とすることで支持タップ候補でないとして処理を終了する. これが満たされる場合は「**holdflag** := 1」とすることで支持タップ候補であるとして処理を終了する. 次に, 支持タップ判

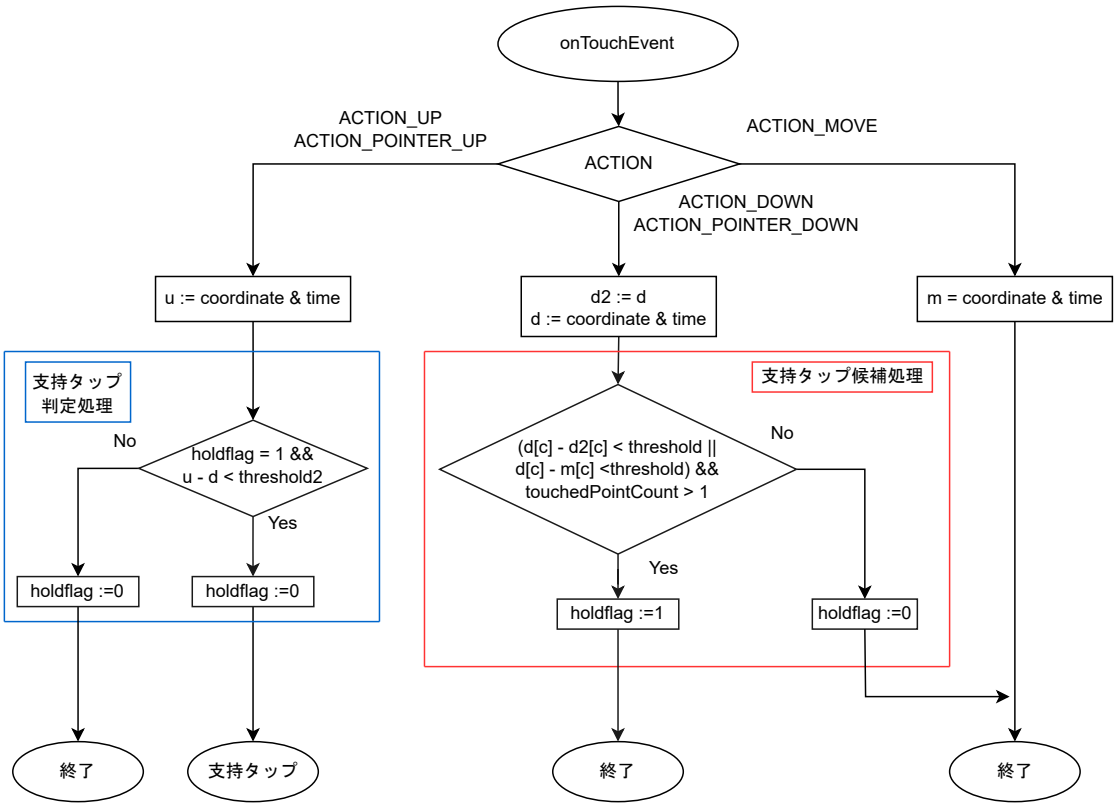


図 4.2: 支持タップのアルゴリズムフロー

定処理には画面に指が離れた時（ACTION_UP, ACTION_POINTER_UP）に移行する．分岐条件「**holdflag** = 1 && **u** - **d** ＜ **threshold2**」は支持タップ候補であり, 前回指が触れた座標と時刻と今回指が離れた座標と時刻の差が閾値未満, つまりいわゆる通常のタップが行われたことを示す．これが満たされない場合は処理を終了する．これが満たされる場合は支持タップと判定し, アプリケーションなどでの実行処理へ移行する．

この支持タップは, 当初の方針通り, 非視認状態を想定している．また, 同様に非視認状態を想定している逆タップと併用することも可能である．まず次の副節では, 逆タップ, 切替スワイプに加えて支持タップを実装する際の説明をする．一方で, 支持タップの操作動作は通常タップと同じ「触れて離す」である．そこで次に続く副節では, 「指が触れている状態は基本的に無操作状態」という前提のもとで, 通常タップと支持タップを併用できる（通常スワイプと逆タップはできない, 切替スワイプも併用可能だが説明を簡単にするため省略）実装について述べる．

4.2.1　逆タップインタフェースへの実装

支持タップを逆タップインタフェースに実装する際の説明をする．図 4.3 は逆タップインタフェースに実装した際のアルゴリズムフローである．第 4.2 節で説明した支持タップ候補処理内の分岐条件に「2 本指逆タップ候補 != 1」を追加している．連続で 2 回, 互いに近い指が画面から離れてそれらの指が再度画面に触れた場合は 2 本指逆タップを優先する．その後, 片方の指を離した場合は支持タップを行わない．また, 先の分岐条件で支持タップ候補ではなかった（**holdflag** := 0）時に, 従来の逆タップ判定処理へ移行する．近くに指が既に触れている場所での逆タップ操作は判定しない．また, このフローには前章で提案した切替スワイプ操作も含まれている．切替スワイプ操作を含む場合は, 指が触れてその後その場で離れる場合と, 指が触れてその後その近傍にあった指が離れる場合を区別しなければならない．そのため閾値 **threshold2** をより厳密に設定する必要がある．

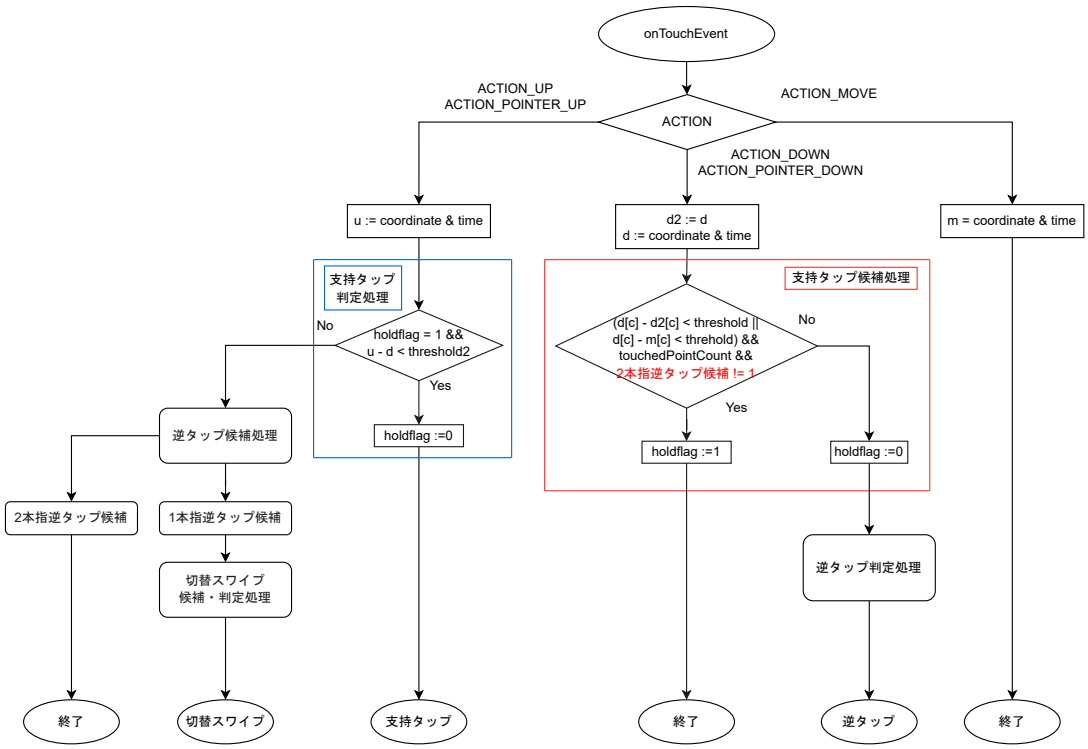


図 4.3: 逆タップインタフェースへ支持タップを実装したフロー

4.2.2　通常タップと支持タップを併用できる実装

本項では「指が触れている状態は基本的に無操作状態」という前提のもとで, 通常タップと支持タップを併用できる実装について説明する. 図 4.4 はこの実装の際のアルゴリズムフローである. なお, 通常のスワイプやフリック, 逆タップ, 切替スワイプは考慮していない. この実装では onTouchEvent の他に, スレッド処理を行う場合に用いられる Java の Handler クラスの postDelayed と removeCallbacks も用いる. postDelayed は第一引数で指定した Runnable オブジェクトの処理を, 第二引数で指定した時間 [ミリ秒] の後に実行する. removeCallbacks は postDelayed で実行しようとした処理を取り消すことができる. **threshold3** はこの postDelayed で設ける遅延時間であり, 事前に値を決めておく. まず, 画面上の指の数が 1 本の状態で指を離した（ACTION_UP）時には postDelayed を用いて遅延して通常のタップ操作と判定する. この遅延時間の間に再度画面に指が触れられた場合は removeCallbaks を用いて遅延実行しようとしていた通常のタップ操作の判定を取り消す. これは, 支持タップをしようと画面に触れている指を動かしている最中に, (通常のタップをしたいわけではないのに) 指が画面から離れてしまった場合にそれをタップと判定しないようするためである. 複数の指で画面に触れている状態から指を離した（ACTION_POINTER_UP）時には第 4.3 節で説明した支持タップの条件を満たす場合のみ支持タップと判定し, それ以外の場合は処理を終了する.

4.3　支持タップの比較実験

提案する支持タップが, 非視認状態かつ揺れている状態でも操作可能であるかどうかを確認する. 全盲状態での揺れている空調パネルの操作を想定した, 通常タップ操作, 逆タップ操作と支持タップ操作との比較実験を行った.

4.3.1　実験準備

切替スワイプ実験の時のように, 画面を 12 のボタン領域に分けた（図 4.5）. また各ボタンに点字状の突起物を張り付けた. 実験は空調操作パネルを模した画面において, 温度や風量, 風向きの変更を行うことを想定している. 2, 3, 4 行目の 1, 3 列目

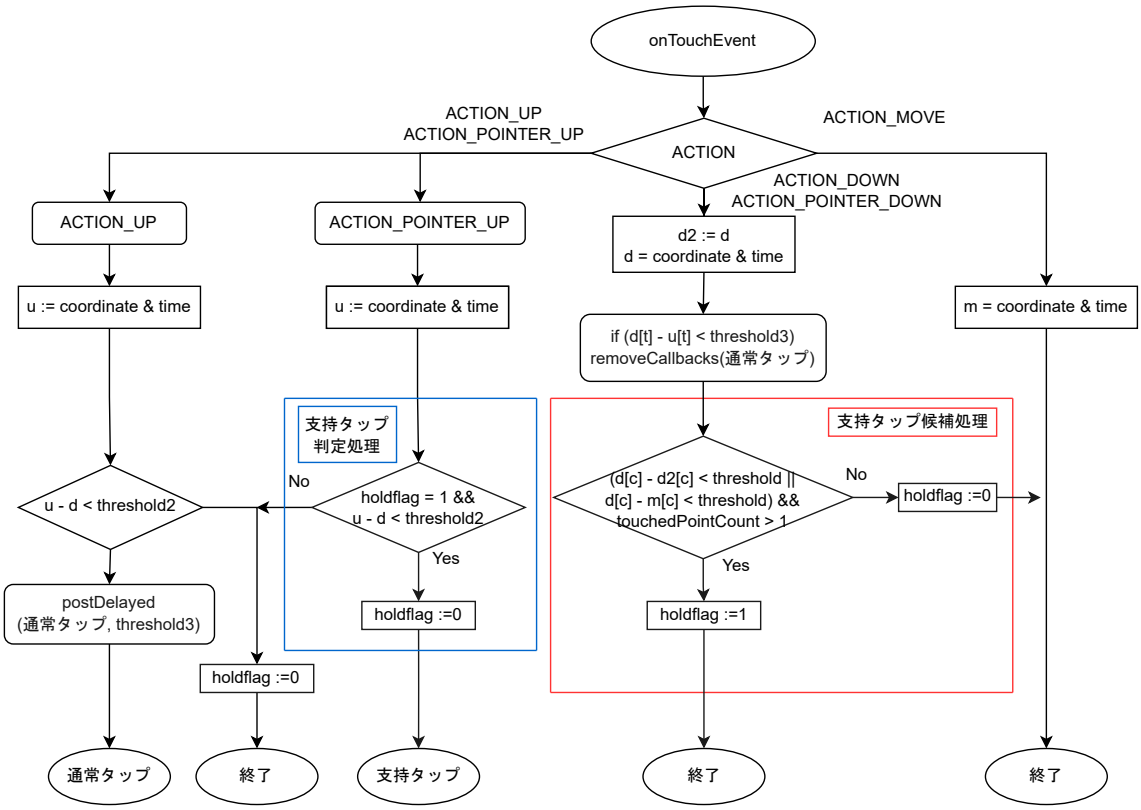


図 4.4: 通常タップと支持タップを併用できる実装のフロー

のボタンは通常タップ操作, 逆タップ操作, 支持タップ操作の実験において, それぞれの操作で動作する. それ以外のボタンには機能を割り当てていない. なお, 何らかの操作が行われた場合はバイブレーションにより通知し, 被験者は画面を見ずとも操作ができたと判別できる. 揺れの再現には Lego Mindstorms Education NXT [16] で組み立てた簡易的な実験装置（図 4.6 上）とゴムボールを用いた. この実験装置には前方に 1 つ, 後方に 2 つのタイヤが付いており, 後方の 2 つのタイヤは機械的に回転させることができる. この 2 つのタイヤを 1 方は前転もう 1 方は後転と, 互い違いの回転を高速に切り替えることで実験装置自体が振動する. 全盲状態の被験者はこの振動する実験装置の上に乗せたタブレットを, さらに腕をゴムボールに乗せた上で操作する（図 4.6 下）. なお, 椅子のひじ掛けにひじを置くことは禁止している. 実験を始める前に被験者に逆タップ操作と支持タップ操作の操作方法を説明し, タブレットの操作練習時間を設けた. これは逆タップ操作と支持タップ操作に慣れてもらい, またボタンの配置と突起物の配置を記憶してもらうことを目的としている.

4.3.2　実験の方法

被験者は 20 代男性 7 名であり, その全てが晴眼者である. この実験は全盲者を想定しているため被験者にはアイマスクをしてもらった. 実験の手順を以下に示す. またこの実験は揺れている状態も想定しているため, 手順中のタブレット操作タスクは全て前項で説明した揺れている状態で行う.

1. 被験者はアイマスクを着用する
2. 被験者は以下の手順を 1 分間できる限り繰り返す
   (a) タブレット操作タスク (温度や風量, 風向きの変更) を受ける
   (b) タスクを実行
   (c) タスクを実行できたと感じたら合図
3. タスク通り実行できた回数を記録する

この手順を, タブレット操作タスクを通常タップのみで行う場合, 逆タップ操作のみで行う場合と支持タップ操作のみで行う場合との 3 回行った. またこの手順を終え

図 4.5: 空調パネルを模した操作画面

図 4.6: （上）実験装置,（下）操作時のイメージ

た後に被験者には自由記述のアンケートに答えてもらった．図 4.7 に実験の様子を示す．

4.3.3　実験結果と考察

それぞれの操作方法に対する 1 分間でのタブレット操作タスクの成功回数と失敗回数を表 4.1 に示す．通常タップ操作の場合は成功の割合が約 19%と低く, 先行研究 [11] で行われた全盲状態のみを想定した実験では約 66%程度成功していたことに鑑みると, 全盲状態に加えて揺れている状態での通常タップ操作はかなり難しいことが確認できた．逆タップ操作の場合と支持タップ操作の場合とを比較してみると, 逆タップ操作の成功割合は約 59%だったのに対し, 支持タップ操作の成功割合は 91%であり, 全盲状態及び揺れている状態においては支持タップ操作の方が有効であることが確認できた．また, 自由記述のアンケートの結果を以下に示す．

- 支持している指が突起物上にずっとあるため, 場所がわからなくなることが無い
- 通常のタップ操作の方がやりやすい

逆タップ操作においてタブレット操作タスクを失敗するケースを分析する．逆タップをするために指を離して再度触れた時に突起物の位置からずれてしまっていた．そのため自分の思っている場所とは違う場所から次のタブレット操作タスクを行ってしまった．すなわち結果的に操作箇所を間違えてしまっていた．一方で支持タップは, タップをする際にも支持している指を突起物上に置いておくことができるので前述したような操作箇所の間違いが少なかったのだろう．アンケート結果 2 つ目の「通常のタップ操作の方がやりやすい」は離す, 触れるという逆タップ操作よりも最終的な判定は通常のタップと同じである支持タップ操作の方がやりやすいという意味だろう．逆タップ操作は新しい操作方法であり使いこなすには多少の慣れが必要であったが, 支持タップは普段から慣れているタップ操作で行うことができるため慣れの影響が少なかったと考えられる．

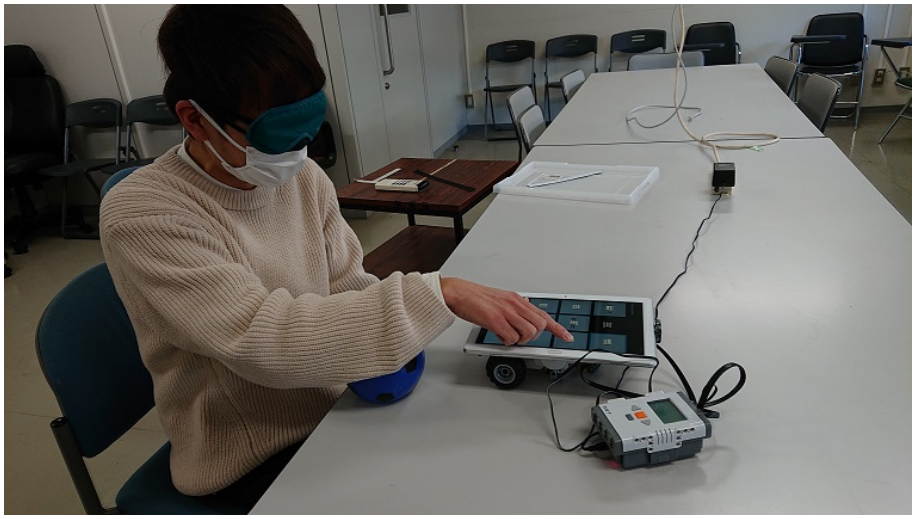


図 4.7: 支持タップ実験の様子

表 4.1: 支持タップ実験の結果

被験者	通常タップのみ		逆タップのみ		支持タップのみ	
	成功回数	失敗回数	成功回数	失敗回数	成功回数	失敗回数
A	2 回	7 回	4 回	4 回	7 回	0 回
B	3 回	10 回	2 回	7 回	6 回	2 回
C	2 回	8 回	5 回	2 回	6 回	0 回
D	0 回	9 回	5 回	3 回	6 回	1 回
E	3 回	6 回	6 回	0 回	8 回	0 回
F	1 回	8 回	3 回	5 回	7 回	0 回
G	2 回	7 回	6 回	1 回	7 回	1 回
平均	1.9 回	7.9 回	4.4 回	3.1 回	6.7 回	0.6 回
成功割合	19%		59%		91%	

第5章　むすび

本論文では, 当研究室で提案している非視認状態を想定したタッチパネル操作インタフェースを対象に新しい操作方法を提案した. まず, スワイプ操作を元にした切替スワイプを提案した. 切替スワイプは操作を開始する場所で指を入れ替えた後で従来のスワイプ操作をし, 操作を終了する場所で指を元に戻す操作方法である. 切替スワイプは画面上のどこでも操作可能な操作方法である. 被験者実験で実際に非視認状態で操作可能であることが確認できた. また, 音量の変更やシークバー調整などの処理を切替スワイプの実行処理とすることが有用であることも示唆された. 次に支持タップという操作方法を提案した. 支持タップは画面に触れている（支持している）指の近くで通常のタップ操作を行う操作方法である. 支持タップも非視認状態で操作可能である. また, 画面に触れている指が手全体を支持することで, 揺れている状態にも比較的正確に操作が可能である. 支持タップは逆タップインタフェースの他, 通常タップと併用する実装も可能である. 被験者実験で非視認かつ動揺状態では, 逆タップ操作よりも支持タップ操作の方が正しく操作可能であることが確認できた.

非視認状態でも利用しやすいタッチパネルインタフェースの実現を目指し, 先行研究 [11][12][13] では, 逆タップ, ダブル逆タップ, 2 本指逆タップという操作方法を提案している. これらの操作方法は操作箇所 1 箇所のみ対象とする操作方法である. 提案した切替スワイプは画面上どこでも操作可能な, 画面レイアウトに依存しない操作方法である. 切替スワイプを提案することにより, 非視認状態を前提としたタッチパネルインタフェースでより複雑なアプリケーションの設計が可能となるだろう. また, 提案した支持タップを利用することで, 走行中の車内のような揺れている状態での操作がより容易になるだろう. 今後も操作方法が過剰になりすぎないよう熟慮しつつ, 非視認状態でのタッチパネル利用がより容易になるようなインタフェース

について検討していきたい.

謝辞

本研究を進めるにあたって, 日頃から多大なご尽力を頂き, ご指導を賜りました名古屋工業大学, 舟橋健司 准教授, 伊藤宏隆 助教に心から感謝いたします. 最期に, 本研究に多大なご協力を頂きました舟橋研究室諸氏に心から感謝いたします.

参考文献

[1] 総務省 令和 3 年版情報通信白書 情報通信機器の保有状況.
https://www.soumu.go.jp/johotsusintokei/whitepaper/ja/r03/html/nd242110.html

[2] 森 博, 田近 一郎, 杉江 晶子, “タブレット PC を活用したマルチメディア教育の試み”, 名古屋文理大学紀要, 12 号, pp. 97-104, 2012.

[3] 柳田 久弥, “授業におけるスマートフォンと授業支援システムの利用に関する一考察”, 富士大学紀要, 48 巻, pp. 87-95, 2016.

[4] Adopting 3D Touch on iPhone - Documentation Archive -Apple Developer.
https://developer.apple.com/library/archive/documentation/UserExperience/Conceptual/Adopting3DTouchOniPhone/

[5] 厚生労働省 平成 28 年生活のしづらさに関する調査：結果一覧.
https://www.mhlw.go.jp/toukei/list/seikatsu_chousa_c_h28.html

[6] Tatsuya Watanabe, Hirotsugu Kaga, Makoto Kobayashi, Kazunori Minatani, “Touchscreen Text Entry Methods Used by Blind and Low Vision Users”, ITE Trans. on MTA, Vol. 7, No. 3, pp. 134-141, 2019.

[7] Carl Hlladay Abraham, Bert Boadi-Kusi, Enyam Komla Amewuho Morny, Prince Agyekum, “Smartphone usage among people living with severe visual impairment and blindness”, Assistive Tchnology, Vol. 34, Issue 5, pp. 611-618, 2021.

[8] 嶋村 幸仁, “スマートフォンによる視覚障害者誘導システムに関する研究”, 筑波技術大学テクノレポート, 24(1), 113-114, 2016.

[9] 山下 晃弘, 佐藤 佳, 佐藤 俊太, 川口 正太郎, 松林 勝志, “UHF 帯 RFID を用いた視覚障がい者向け歩行者ナビゲーションシステムの開発と展示会への適用”, 情報処理学会論文誌コンシューマ・デバイス＆システム（CDS）, 7(1), pp. 34-43, 2017.

[10] Hsuan-Eng Chen, Yi-Ying Lin, Chien-Hsing Chen, I-Fang Wang, “BlindNavi: A Navigation App for the Visually Impaired Smartphone User”, CHI EA’15: Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems, pp. 19-24, 2015.

[11] Kenji Funahashi, Hayato Maki, Yuji Iwahori, “Novel tap operation on capasitive touch screen for people with visual impairment”, ICAT-EGVE 2020 Posters and Demos, pp. 3-4, 2020.

[12] 加藤 馨, 牧 隼人, 舟橋健司, “非視認状態を想定した新たなタップ操作インタフェースの多様化の提案”, NICOGRAPH2021 講演論文集, S-10, 2021.

[13] Kyo Kato, Hayato Maki, Kenji Funahashi, “Diversification of a novel tap operation interface for people with visual impairment”, NICOGRAPH International 2022, p.101, Poster, 2022.

[14] A New Car UI - Matthaeus Krenn, https://matthaeuskrenn.com/new-car-ui/

[15] 加藤 馨, 舟橋 健司, “非視認状態でのタッチパネル利用を想定した新しいスワイプ操作インタフェース”, 日本バーチャルリアリティ学会第 27 回大会講演論文集, 2D1-4, 2022.

[16] レゴ® マインドストーム® について. https://www.lego.com/ja-jp/themes/mindstorms/about

.

発表論文リスト

1. 加藤 馨, 牧 隼人, 舟橋健司, “非視認状態を想定した新たなタップ操作インタフェースの多様化の提案”, NICOGRAPH2021 講演論文集, S-10, 2021.
2. Kyo Kato, Hayato Maki, Kenji Funahashi, “Diversification of a novel tap operation interface for people with visual impairment”, NICOGRAPH International 2022, p.101, Poster, 2022.
3. 加藤 馨, 舟橋 健司, “非視認状態でのタッチパネル利用を想定した新しいスワイプ操作インタフェース”, 日本バーチャルリアリティ学会第 27 回大会講演論文集, 2D1-4, 2022.