透過型HMDにおけるフリック入力の検討

大岡 湧汰[†1] 入江 英嗣[†1] 坂井 修一[†1]

概要：近年の AR 技術の進歩に伴い，特殊な入力機器が不要な透過型 HMD による高度な AR アプリケーションが実現可能となってきたが，そのようなデバイスではほとんどの場合，文字入力が非常に難しいという問題がある．これらのデバイスで QWERTY キーボードを用いて日本語入力するのは，入力速度・正確性のどちらも気軽に使えるレベルには達していない．この問題に対する解決策の一つとして音声入力があるが，VR デバイスとは違い公共の場での利用が想定される AR デバイスでは使えない場面も多い．他の解決策として，スマートフォンと AR デバイスを連携してスマートフォン経由で文字入力をする手法も提案されているが，ハンズフリーで作業する必要がある場合には利用できない．本論文では，指で空中をフリックするジェスチャ入力をひらがなの入力に，AR 環境では有用とされるダイヤルジェスチャを変換候補の選択に，それぞれ用いることで高速かつ正確に入力できる，新たな入力手法を提案する．HoloLens 上に提案手法を具体的に実装したソフトウェア「AirFlickey」を開発した．AirFlickey の各機能の概要と詳細な実装を説明した上で，フリック入力と QWERTY キーボード入力の入力速度・エラー率・ユーザビリティを計測する実験の詳細を説明する．被験者を対象とした試用評価では，先行研究と比較し入力速度の面で 3 倍程度の向上，ユーザビリティ評価の面で 1 段階の向上が確認された．

Flick Input for See-Through Head Mounted Display

YUTA OOKA[†1] HIDETSUGU IRIE[†1] SHUICHI SAKAI[†1]

1. はじめに

近年，HoloLens[1] を始めとして，Magic Leap[2]，nreal light[3]，XR-1[4] と，様々な AR デバイスが発表・発売されている．これらのデバイスは依然として高価であり，現在は依然として一般家庭に普及していないが，VR デバイスが普及した際と同様に，将来的には安価になり普及していくと予想されている．中でも，HoloLens は入力にコントローラを必要としないという特徴があり，このような素手で操作できる透過型のヘッドマウントディスプレイ (以下 HMD と略す) に対する需要は，工業用デバイスという用途だけでなく，手に接触するデバイスを必要としないという衛生上の利点もあいまって，高まる傾向にある．

ところが，このようなデバイスはほとんどの場合，文字入力が非常に難しいという問題を抱えている．HoloLens においてもそれは同様で，デフォルトでは英語入力と同じ QWERTY キーボードを用いて日本語入力するしかない (図 1)．QWERTY キーボードの中で入力したい文字に視線を合わせてクリックすることを正確に繰り返すのには労力を要する．

加えて，キーの文字サイズが小さいことから，視力が低い人にとってこのキーボードで入力するのは難しい．そうでなくとも，キーの総数が多いために各キーの領域が小さく，視線が少しずれるだけで間違って入力してしまうこと

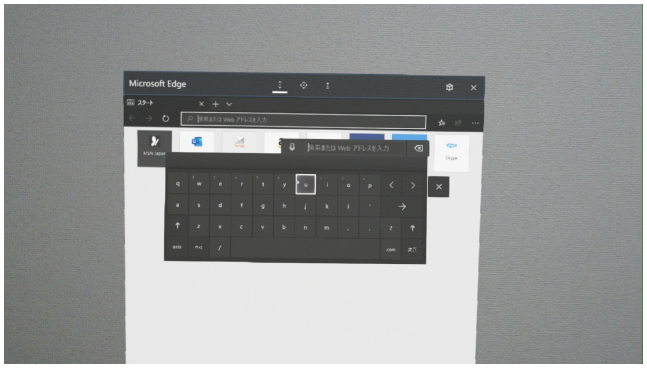

図 1 HoloLens におけるデフォルトの日本語入力

†1 現在，東京大学大学院情報理工学系研究科

が多々ある．

これらの問題に対する解決策の一つには音声入力があるが，VR デバイスとは違い公共の場での利用が想定される AR デバイスでは使えない場面も多い．それ以外には，スマートフォンと HoloLens を Wi-Fi で連携して，スマートフォンから HoloLens に入力するという手法も提案されている [5]．しかし，工業用デバイスという用途では，ハンズフリーで作業する必要がある場合も多く，その場合には利用できない．

また，AR 環境ではなく VR 環境における文字入力手法であれば，従来から様々なものが提案されている．従来の入力手法とは異なる特殊な手法や，QWERTY キーボードを用いた入力手法，また比較的少数だがフリック入力に着目した手法も提案されている [6][7][8]．ところが，AR 環境における文字入力手法は VR 環境のそれと比較してあまり多く提案されていない．そもそも，VR 環境ではジェスチャーや音声による情報伝達が可能であるため，文字入力の必要性自体が低い．対して AR 環境では，将来的に一般家庭に普及することも想定すると，周囲に知られたくない文章を入力する場合もあるため，文字入力の必要性は高い．将来的に両手 10 本の指を認識できるデバイスが開発されたとしても，現時点で QWERTY 入力よりフリック入力の方が得意であるという人が一定数存在する [9] だけでなく，フリック入力キーボードには，キー数が少ない上にキー配置を覚えなくともすぐに使えるようになる，というメリットがあるため，フリック入力に対する需要は存続する．

加えて，QWERTY キーボードの全体を視野に入れるように表示すると一つのキーが小さくなってしまうが，フリック入力キーボードには，キーボード全体を視野に入れるように表示しても一つのキーを大きくでき，選択時にミスが発生しにくいというメリットもある．

また，日本語入力にはひらがなの入力だけではなく，変換候補の選択という操作も必要である．頭の向きや視線を使って変換候補を選択する場合には，カーソルを頭や目を使って動かしてからクリックするという操作が必要である．しかし，先に述べたように選択範囲が小さいため，QWERTY キーボードと同様に視線が少しずれるだけで間違って選択されてしまうことが多々ある．

この問題に対する解決策として，丸山らによるダイヤルジェスチャを用いた入力手法 [10] を変換候補の選択に組み込むことを提案する．彼らは，ダイヤルジェスチャを用いるとユーザはスクロールしやすいという結論を導いている．

本研究の主目的は，スマートフォンでよく利用されるフリック入力の透過型 HMD への導入を検討することである．代表的な素手で操作できる透過型 HMD として HoloLens を選択した．また，ひらがなの入力にフリック入力を用いた上で，変換候補の選択にはダイヤルジェスチャ入力を用いることで，最大限快適な入力インターフェイスを設計することを目指す．

本論文は，全 6 章で構成する．第 2 章では関連研究としてまずフリック入力について述べ，次に VR 環境におけるフリック入力インターフェイスと AR 環境におけるダイヤルジェスチャ入力について述べたのちに，本研究の位置づけを明らかにする．第 3 章では AirFlickey の概要と詳細な実装について述べる．第 4 章では提案手法の有用性を確認するための実験方法・結果とその評価を述べる．第 5 章では提案手法に関する考察を施し，第 6 章で結論を導き，今後の展望について述べる．

2. 関連研究

本章では，フリック入力について詳しく述べたのち，関連研究として VR 環境におけるフリック入力及び AR 環境におけるダイヤルジェスチャ入力について説明する．最後に，本研究の位置づけを明らかにする．

2.1 フリック入力

フリック入力とは，特定のキーをタップしたのちに弾く（フリックする）ことによって文字を入力する方法である．1998 年に Apple Newton 用に開発された Hanabi[11](**図 2**) が草分けで，2008 年に iPhone[12] に採用された (**図 3**) ことで急速に普及した．

具体的には，そのままタップすれば「あ段」を，左，上，右，下とフリックすれば，それぞれ「い段」,「う段」,「え段」,「お段」を入力することができる．このように，フリック入力は 1 文字＋ 4 方向で 5 つの文字が 1 セットとして扱える入力方法なので,「あいうえお」と 5 段存在する日本語とはとても相性がよい．実際，海外でフリック入力を用いているユーザはほとんど存在しないが，日本ではフリック入力しているスマートフォンユーザが大半である．

スマートフォンにおけるフリック入力がここまで日本

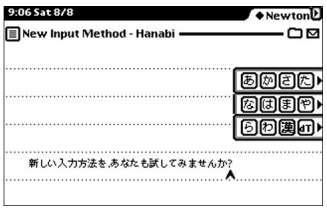

図 2 Hanabi のインターフェイス [11]

図 3 iPhone でのフリック入力 [12]

に浸透した理由として，上で述べたもの以外にもある．QWERTY キーボードは小さなスマートフォンに埋め込むにはキー数が多すぎる上に，スマートフォン上に 10 本の指を置くことは不可能であることである．同様の理由で，視界が狭く 10 本の指を認識できない HoloLens においては，スマートフォンと同じようにフリック入力に優位性がある．

ただ，将来的には AR デバイスも視界が広くなり，10 本の指を認識できるようになると言われており，その場合 AR 上の QWERTY キーボードが現在の物理キーボードと同程度の快適さで入力できるようになる可能性もある．しかし，フリック入力キーボードには，QWERTY キーボードと比べキー数が少ない上にキー配置を覚えなくともすぐに使えるようになる，というメリットがあるため，ライトユーザーには利用され続けると予想できる．

2.2 VR 環境におけるフリック入力

小澤ら [6] は，VR 環境において「あ段」を分散して配置し，特定のキーをつまんで引っ張る（フリックする）操作で入力する方法を提案している．これは本研究における操作ととても似通っているように見えるが，本研究とは VR と AR という点で異なる上に，視線の先にあるキーを遠くからフリックするのではなくキーに触れてフリックするという点で異なる．

福仲ら [7] は，スマートフォンと同様にタップしてフリックする入力方法を，Leap Motion を用いて視覚フィードバックとともに実現している．これは本研究の操作とは異なっているだけでなく，VR と AR という観点で異なり，視線を使わないという点でも異なる．

また喜多ら [8] は，VR 環境においてキーボード入力とフリック入力のどちらが適しているかを調べ，VR 環境においてはキーボード入力の方が優れているという結論を導いているが，最後に「被験者はキーに触れるという感覚を掴みかねている様子を観察できた」と繋げている．これはキーに直接触れる形式のインターフェイスによって生まれた問題であり，視線の先にあるキーをフリックする本研究においては当てはまらない．また，彼らは「フィードバックがなかったことが原因」とも述べていたため，本研究では視覚・聴覚フィードバックを加えている．

2.3 AR 環境におけるダイヤルジェスチャ入力

丸山ら [10] は，空中に円を描くダイヤル式ジェスチャ入力において，描いた円弧の角度を入力として用いる文字入力手法を提案している．たしかに AR 環境において，ダイヤルジェスチャによる入力の優位性は大きいが，新たな入力手法をユーザが習得するまでに時間がかかるというデメリットもある．また，丸山らはダイヤルジェスチャ入力をスクロール操作に用いると多くのユーザがスムーズに利用できるが，文字入力に用いると入力速度が比較的遅くなってしまうと結論付けている．そのため，本研究においてはひらがな入力の際にはフリック入力を用い，変換候補の選択時にはダイヤルジェスチャ入力を用いることで高速かつ正確に入力できるようにする．

2.4 本研究の位置づけ

我々の知る限り，今までにフリック入力が AR 環境に導入された例はない．加えて，VR 環境においても視覚・聴覚フィードバックをともに実装した例や，変換候補の選択にダイヤルジェスチャを用いるという手法を利用した例なども未だ存在しない．ゆえに，本研究は既存の入力手法であるフリック入力とダイヤルジェスチャ入力をうまく組み合わせて最大限使いやすい日本語入力インターフェイスを設計することを目指す．

3. AirFlickey

3.1 AirFlickey の概要

現在の透過型 HMD は英語圏向けに作られている上，技術として発展途上であることも相まって HMD 環境における日本語入力の手法は未だ確立されていない．本研究では，スマートフォンでの日本語入力でよく用いられるフリック入力を透過型 HMD に導入することで，慣れ親しんだ入力手法を用いれば入力速度を高め，ユーザー体験も向上できるのではないかという仮説を立てた．

先行研究においては，フリック入力を HMD 環境に実装する際に，スマートフォンですでに実装されている聴覚フィードバックを加えていなかったり，キーをクリックしてドラッグする操作をうまく HMD 環境に特化した手法に落とし込むことができていなかったりしていた．本研究では，AirTap と呼ばれる人差し指と親指をくっつける動作をクリックとし，人差し指と親指を離す動作をドラッグ動作の終了と定義することで，HMD 環境において使いやすいフリック入力の導入を検討する．

フリック入力をひらがなの入力に，後述するダイヤルジェスチャ入力を変換候補の選択に用いるインターフェイスを導入するため，「AirFlickey」というソフトウェアを HoloLens 用に開発した．ちなみにこの命名は，このソフトウェアが空中 (Air) をフリック (Flick) するキーボード (Keyboard) であるため，これらの単語を結合したことによる．この項では，AirFlickey の概要をフリック入力機能と日本語変換機能の 2 つに分けて述べる．

3.1.1 フリック入力機能の概要

図 4 は，「な」キーを下にフリックする前のキーボードとフリック直後のキーボードの，実際に表示される画面である．AirTap されたキーは暗く変化し，フリック方向にキーが飛び出るようなインターフェイスになっている．これが使用感を向上させるための視覚フィードバックであ

る．また，文字入力時には HoloLens のデフォルト文字入力音が再生されるような聴覚フィードバックも加えた．色合いもシステムのデフォルトキーボードのそれと似せたことによって，システムのデフォルトである QWERTY キーボードからさほど使用感を変化させずにフリックキーボードを実装できた．

また，フリック入力の詳細な仕様は iOS のそれに合わせた．具体的には，「や」を左にフリックすると “「” が，右にフリックすると “」” が，「わ」を右にフリックすると “！” が入力される．これによって，今までスマートフォンでフリック入力を利用してきたユーザが，HoloLens においても違和感を抱かずに入力できるようになる．

3.1.2 日本語変換機能の概要

図 5 は，AirFlickey においてひらがなを入力したあとに表示される変換候補を選択するために AirTap をした直後の画面と，ダイヤルジェスチャをしている間の画面である．変換ボタンを AirTap したあとダイヤルジェスチャを始めると，画面上部の変換候補のうち一つが黄色でハイライトされ，そのままダイヤルジェスチャを続けると変換候補のハイライトが一つずつ隣に移動するようになっている．そして，AirTap 状態を解除することで変換を確定することができる．

3.2 AirFlickey の詳細な実装

3.2.1 フリック入力機能の詳細な実装

AirFlickey の実装には手の状態と絶対情報を含んだ生データを提供する Unity の API を利用している．Unity の API には以下の 5 つがある．

- SourceDetected：手を認識した瞬間に呼ばれる（複数の手に対応）
- SourceLost：一度認識した手を見失った瞬間に呼ばれる
- SourcePressed：クリックした瞬間に呼ばれる
- SourceReleased：クリックを解除した瞬間に呼ばれる
- SourceUpdated：手の位置が更新されるたびに呼ばれる

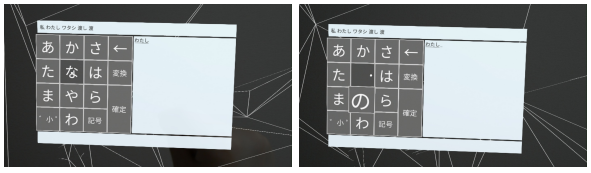

図 4 AirFlickey におけるフリックの様子

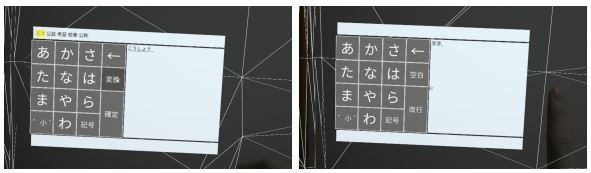

図 5 AirFlickey におけるダイヤルジェスチャの様子

本研究ではこれらの API の中で，SourceDetected 以外の 4 つを用い，以下のようなステップでフリック入力を実現している．

(1) SourcePressed が呼ばれた時に視線の先にあるキーオブジェクト（MRTK で取得可能）と手の 3 次元絶対座標を記憶しておく．
(2) SourceUpdated が呼ばれるたびに「この時点で SourceReleased が呼ばれたと仮定したときに入力されるべき文字」を取得する（後述するフリック判定を用いる）．
(3) SourceReleased ／ SourceLost が呼ばれると，クリックが解除されたと判断し，後述するフリック判定を行う．

フリックしたかどうか，したのであればどの方向（上，下，右，左）にフリックしたのか，という「フリック判定」は，以下のように行っている (**図 6**)．

(1) クリックを開始した時点での手の絶対座標からみた，クリックを終了した時点での手の絶対座標の三次元ベクトルを，ユーザの顔の向きと垂直な平面に正射影した二次元ベクトルを考える．
(2) 1. のベクトルを (x,y) としたとき，ベクトル (x,y) の大きさが 0.02 未満の場合はフリックしなかったと判定する（この値は開発中の試行錯誤によって導いた）．
(3) 2. でフリックしたと判定された場合，ベクトル (x,y) とベクトル (1,0) のなす角を計算し，その値で以下のように場合分けする．

- 右: $0° \leq \theta < 45°, 315° \leq \theta < 360°$
- 上: $45° \leq \theta < 135°$
- 左: $135° \leq \theta < 225°$
- 下: $225° \leq \theta < 315°$

3.2.2 日本語変換機能の詳細な実装

入力した文字列のうち，確定していない文字列を変換対象の文字列とする．変換対象の文字列が更新されるたび，Google の日本語変換 API である「Google CGI API for Japanese Input」[13] にリクエストを送る．そしてその結果をリアルタイムにキーボードの上部に表示するようにしている．

そして，「変換」ボタンを AirTap した直後から，手の絶

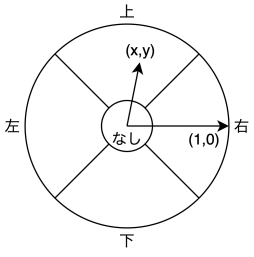

図 6 フリック判定

対座標の動きから描こうとしている円を推定し始める．推定された円を 8 分割し，そのうち一つぶん右回転すると次の候補に，左回転すると前の候補にフォーカスされる．AirTap を解除するとフォーカスされた変換候補が確定され，変換対象文字列と置き換えて確定される．

4. 評価環境

4.1 実験方法

被験者 6 人に，まず Desktop Note[14] というメモ帳アプリで文字入力を行ってもらい，入力時間とミス数を計測したのちに，先述した AirFlickey で文字入力を行ってもらい，同様に入力時間とミス数を計測することで，ひらがなの入力にフリック入力を用いる手法の評価を行った．被験者は全員右利き，被験者 1 と 2 は女性，それ以外は男性で，被験者 3 と 4 は HoloLens を利用したことがあると回答し，被験者 5 はフリック入力を利用したことがないと回答した．

課題の単語を読み上げ，図 4 の右側に表示されている文字入力エリアにひらがなで入力してもらうことで入力速度とエラー率を計算した．本実験ではひらがな入力に関してのみ行い，フリック入力キーボードとしての有用性を調べるため，変換操作や確定操作は行っていない．

課題の単語は全部で 8 個あり，すべてタイピング速度の測定で実際に用いられているものである．フリック入力とフルキーボード入力での入力速度差を正確に調べるため，両方のキーボードで同じ単語を入力してもらった．実際に入力してもらった単語は以下の 8 つである．

- たんさん
- おりょうり
- こたつ
- すぺしゃる
- でんき
- だっしゅつ
- えいかいわ
- ろうどう

ユーザビリティ評価のため，2 つのキーボードで入力してもらったあとに，被験者全員に主観的評価を行うためのアンケート調査を行った．アンケートの質問文には，ユーザビリティ調査によく用いられる SUS(System Usability Scale)[15] を用いた．SUS とは，1（全くそう思わない）から 5（とてもそう思う）の 5 段階から選択する質問を 10 個用意することでユーザビリティの受け止められ方を評価する指標であり，奇数番目にポジティブな質問，偶数番目にネガティブな質問を配置するのが一般的である．以下に実際に行ったアンケートの質問の一覧を示す．

- このシステムを使いたいと思う
- このシステムは不必要なほど複雑だと感じた
- このシステムは使うのが簡単だと思う
- このシステムには一貫性のないところが多々あったと思う
- このシステムの様々な機能はうまく統合されていると感じた
- このシステムを使うことができる人は限られると思う
- たいていの人がこのシステムをすぐに使えるようになるだろうと思う
- このシステムがとても使いにくいと感じた
- このシステムを使うのに自信があると感じた
- このシステムを動かす前に多くのことを学ぶ必要があった

被験者には，AirTap を含めた HoloLens の操作方法とデフォルトキーボードの入力方法・AirFlickey の入力方法を 3 分ほど説明し，一連の動作を行えるまで練習してもらった．その後，以下の 2 単語を高速かつ正確に入力してもらうことで，本番前の練習を行った．

- いそっぷ
- かくれる

その後，8 個の単語を Desktop Note 内の QWERTY キーボードで入力するセッションを行ってもらった．HoloLens の操作による腕の疲労を考慮し，5 分間の休憩を挟んだのち，今度は AirFlickey 内でのフリック入力キーボードで同じ 8 単語を入力してもらった．実験は全て屋内で行った．

4.2 評価項目

4.2.1 入力速度

入力速度の指標としては一般的に WPM(Words Per Minute) が用いられるが，日本語入力の場合，1 単語の文字数にばらつきがある上，ローマ字入力とフリック入力でひらがな一文字を入力するためにクリックする必要のあるキー数が違うため，新たに CPM(Characters Per Minute) を「1 分間に正確に入力できるひらがな数」と定義し，これを用いることにする．

CPM の計算方法は以下の式で表される．ただし C は入力した文字数，S は入力にかかった秒数を表している．

$$CPM = \frac{C}{S} \times 60$$

各被験者の各単語ごとの CPM を計算したところ，**図 7** のようになった．

また，各被験者の全単語の CPM の平均を計算すると**図 8** のようになる．

なお，全被験者の平均 CPM は，QWERTY キーボードは 20.3，フリック入力キーボードは 26.5 であった．また，全被験者の中で最高の CPM は，QWERTY キーボードは 32.9，フリック入力キーボードは 41.6 であった．

また，本実験における CPM は 1 分間に正確に入力できるひらがなの文字数であるが，この値を一般的に使われる入力速度の指標である WPM と比較することを考える．WPM は「1 分間に正確に入力できる英単語数」という定

図 7　各被験者の各単語ごとの CPM

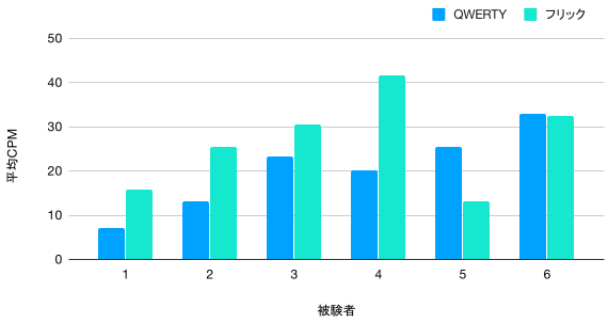

図 8　各被験者の全単語の平均 CPM

義であるが，これはローマ字入力を前提とした指標である．一般に，一英単語の平均文字数は 5 であると知られており，ひらがな 1 文字はローマ字入力で換算すると平均して英字 2 文字分であるから，以下の概算が可能である．

$$WPM = CPM \times \frac{2}{5}$$

ゆえに，全被験者のフリック入力における平均 CPM を WPM に換算した値は，10.6 となる．

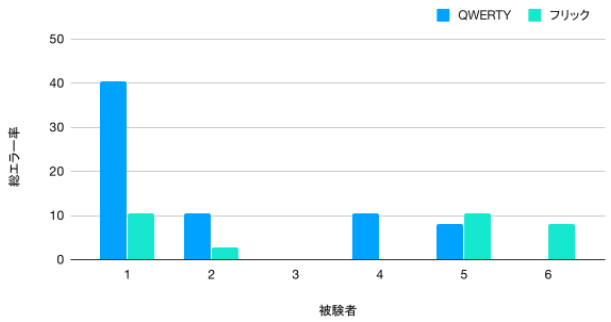

図 9　各被験者の全単語を通した総エラー率

表 1　アンケートの結果

質問	全くそう思わない	あまりそう思わない	どちらでもない	まあそう思う	とてもそう思う
使いたい	0	1	1	3	1
不必要なほど複雑	2	3	1	0	0
使うのが簡単	1	1	1	1	2
一貫性のないところがある	3	3	0	0	0
うまく統合されている	0	1	2	1	2
使うことができる人は限られる	1	4	0	1	0
すぐに使えるようになる	0	2	0	2	2
とても使いにくい	1	2	2	1	0
使うのに自信がある	1	0	1	2	2
多くのことを学ぶ必要がある	3	1	1	1	0

4.2.2　エラー率

Soukoreff ら [16] は，入力には以下の 4 種類が存在すると述べている．

- Correct (C): 正しく入力された文字
- Incorrect-not-fixed (INF): 間違えて入力されたが，気付かれなかった文字
- Incorrect-fixed (IF): 間違えて入力され，バックスペースによって訂正された文字
- Fix (F): バックスペースキーの入力

本実験では被験者による入力が課題単語と一致していることを確認するまでが計測対象であるため，INF は存在しない．ゆえに，Soukoreff ら [16] によって提唱された総エラー率 TER(Total Error Rate) は以下の式で表される．

$$TER = \frac{INF + IF}{C + INF + IF} \times 100 = \frac{IF}{C + IF} \times 100$$

各被験者ごとの全単語に対する総エラー率を計算すると図 9 のようになる．

なお，全被験者の平均総エラー率は，QWERTY キーボードは 11.6%，フリック入力キーボードは 5.3%であった．また，全被験者の中で最高の総エラー率は，QWERTY キーボードは 40.3%，フリック入力キーボードは 10.5%であった．

4.2.3　ユーザビリティ

アンケートで得られた結果を表 1 に示す．SUS スコアは得られた結果を用いて以下の式で求められる．ただし，a_n は n 番目の回答 (1〜5) である．

表 2　SUS スコアによる判定の基準

SUS スコア	グレード	付加される評価
> 80.3	A	Excellent
$68 - 80.3$	B	Good
68	C	Okay
$51 - 68$	D	Poor
< 51	E	Awful

表 3　被験者 5 を除いたアンケートの結果

質問	全くそう思わない	あまりそう思わない	どちらでもない	まあそう思う	とてもそう思う
使いたい	0	1	0	3	1
不必要なほど複雑	2	2	1	0	0
使うのが簡単	1	1	0	1	2
一貫性のないところがある	2	3	0	0	0
うまく統合されている	0	1	2	0	2
使うことができる人は限られる	1	4	0	0	0
すぐに使えるようになる	0	1	0	2	2
とても使いにくい	1	2	1	1	0
使うのに自信がある	0	0	1	2	2
多くのことを学ぶ必要がある	3	1	1	0	0

$$SUS = 2.5 \times \left\{ \sum_{k=1}^{5} (a_{2k-1} - 1) + \sum_{k=1}^{5} (5 - a_{2k}) \right\}$$

本実験での SUS スコアの平均は 70.0 であった．SUS スコアの基準を表 2 に示す．本実験の SUS スコアは一般の標準平均 68.0 を上回り，B グレード，Good 判定となる．

また，透過型 HMD でフリック入力を利用するのはスマートフォンでフリック入力を利用したことがあるユーザのみであると考えれば，フリック入力を利用したことがない被験者 5 を除いて SUS スコアを算出しても問題ない．その場合，以下のようなアンケート結果となる．この抽出した結果を元に SUS スコアを算出すると，74.5 となる．

5.　評価結果

5.1　フリック入力機能

5.1.1　入力速度

まず，透過型 HMD におけるフリック入力の入力速度について考察する．フリック入力に全く慣れていない被験者 5 を除いた全員が，HoloLens においてフリック入力を用いることで，QWERTY 入力と比較してより早く入力できていた．しかし，フリック入力を使ったことがない被験者 5 は，ほぼすべての単語に関して QWERTY 入力の方が早く入力できた．実験前からある程度予想していたが，要するにフリック入力をスマートフォンですでに慣れている人にとっては透過型 HMD におけるフリック入力も使いやすく，そうでない人にとっては使いにくいと考察できる．また，全被験者の入力速度を WPM に換算した値は 10.6 であり，AR 環境におけるダイヤルジェスチャを用いた英文入力の速度を測定した丸山ら [10] の記録である 3.62 を大幅に超える値であり，先行研究と比較し提案手法は入力速度の面で大きく改善できた．

5.1.2 エラー率

エラー率に関しては，QWERTY キーボードでの総エラー率が際立っている．これと比較すると，フリック入力の総エラー率は非常に低い．

丸山ら [10] によるダイヤルジェスチャ入力に関する実験の 1 セット目と比較するとエラー率は提案手法の方が低い結果となった．ただ，この値はその実験における 3 セット目のエラー率よりも高い．これは，本実験が 1 セットしか行われず，HoloLens に慣れていない被験者が十分に操作に慣れる前に実験を終えてしまったことが原因であり，実際 HoloLens を過去に利用したことがある被験者 3 と 4 はフリック入力でのミスはない．ゆえに，このエラー率はデバイスへの慣れとともに低下していく可能性がある．

5.1.3 ユーザビリティ

ユーザビリティに関するアンケート調査の結果，丸山ら [10] によるダイヤルジェスチャ入力の SUS スコア (63.93) より十分に高い値 (70.0) を得ることができた．また，この値は SUS スコアの標準平均 68.0 を上回るため，よいインターフェイスであると結論付けられる．また，フリック入力を利用したことがない被験者 5 を除いて算出した SUS スコアは 74.5 であり，この値を先行研究と比較すると大幅な改善である．

5.2 日本語変換機能

日本語変換にダイヤルジェスチャ入力を用いることの有用性は実験に基づいて示すことができなかったが，丸山ら [10] による研究によって AR 環境におけるスクロール操作は一般的にダイヤルジェスチャを用いると安定すると結論付けられている上，実際，変換候補を選択する操作は正確かつ高速に行えたため，十分に有用である．

6. おわりに

本研究では，透過型 HMD においてフリック入力をひらがな入力に，ダイヤルジェスチャ入力を変換候補の選択に利用する入力手法を提案し，AirFlickey という HoloLens 用フリック入力ソフトウェアを開発した．フリック入力機能に関するユーザ実験の結果，ひらがな入力速度は CPM26.5，WPM10.6 を達成し，総エラー率は 5.3%，SUS スコアは 70.0(補正後 74.5) を達成した．エラー率は丸山ら [10] によるダイヤルジェスチャ入力と同等であるものの，WPM と SUS スコアは大幅に改善することができた．

今後の課題として，提案手法の一部である，変換候補の選択においてダイヤルジェスチャを用いる手法の実験による評価を行うことが挙げられる．また，本研究の実験を同じ被験者に対して 3 回行うことでより HoloLens の操作に慣れさせ，エラー率が下がるかどうかの実験も今後行いたい．

提案手法は，HoloLens のような 10 本の指が認識できない透過型 HMD においてはもちろん有用であると結論付けられるが，将来的に 10 本の指が認識できる HMD デバイスが登場しても，QWERTY 入力と比べフリック入力をより得意とするユーザからの需要は少なからず存在する．そのため，提案手法は現在の透過型 HMD に限定されたものではなく，今後登場するあらゆる HMD デバイスに適用可能である．

参考文献

[1] Microsoft. HoloLens. https://www.microsoft.com/ja-jp/hololens/hardware. (Accessed on 02/06/2020).

[2] Magic Leap. https://www.magicleap.com/. (Accessed on 02/06/2020).

[3] nreal light. https://www.nreal.ai/. (Accessed on 02/06/2020).

[4] Varjo. XR-1. https://varjo.com/products/xr-1/. (Accessed on 02/06/2020).

[5] sumihiro. HoloKeyboard for iOS. https://github.com/sumihiro/HoloKeyboard. (Accessed on 02/06/2020).

[6] 宗馬小澤, 猛梅澤, 範高大澤. J-036 空中におけるつまむ動作を用いた効率的な文字入力の検討 (J 分野:ヒューマンコミュニケーション＆インタラクション, 一般論文). 情報科学技術フォーラム講演論文集, Vol. 14, No. 3, pp. 389–390, aug 2015.

[7] 伊織福仲, 浩然謝, 一乗宮田. VR 環境におけるフリック入力形式インタフェースの開発. Technical Report 3, 北陸先端科学技術大学院大学, 北陸先端科学技術大学院大学, 北陸先端科学技術大学院大学, mar 2019.

[8] 喜多修太郎, 小倉加奈代, 高田豊雄ほか. LeapMotion を用いた VR 上での文字入力手法の検討. 研究報告ヒューマンコンピュータインタラクション (HCI), Vol. 2019, No. 21, pp. 1–7, 2019.

[9] 長澤直子. 大学生のスマートフォンと pc での文字入力方法. コンピュータ＆エデュケーション, Vol. 43, pp. 67–72, 2017.

[10] 丸山桂史. ダイヤル式ジェスチャによるインタラクション手法の提案. 2019.

[11] Newton Japan. Hanabi. https://newtonjapan.com/hanabi/. (Accessed on 02/06/2020).

[12] Apple. iPhone. https://www.apple.com/jp/iphone/. (Accessed on 02/06/2020).

[13] Google. Google CGI API for Japanese Input. https://www.google.co.jp/ime/cgiapi.html. (Accessed on 02/06/2020).

[14] ssz666. Desktop note. https://www.microsoft.com/ja-jp/p/desktop-note/9p5rdm9t8cr3?activetab=pivot:overviewtab. (Accessed on 02/06/2020).

[15] J. BROOKE. Sus : A quick and dirty usability scale. *Usability Evaluation in Industry*, pp. 189–194, 1996.

[16] R. Soukoreff and I. MacKenzie. Metrics for text entry research. p. 113, 01 2003.