

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

ユーザ定義手書きジェスチャの高速かつ軽量な
認識アルゴリズム

山路 大樹

修士（工学）

（コンピュータサイエンス専攻 学位プログラム）

指導教員 志築 文太郎

2017年3月

概要

目 次

第1章 序論	1
1.1 背景	1
1.2 目的	2
1.3 貢献	4
1.4 本論文の構成	4
第2章 関連研究	5
2.1 手書きジェスチャ認識アルゴリズム	5
2.2 ユーザ定義に特化した手書きジェスチャ認識アルゴリズム	6
2.3 手書きジェスチャ認識を可能にするツールキット	6
2.4 \$-Family Recognizer	6
2.5 手書きジェスチャの評価研究	8
2.6 本研究の位置づけ	8
第3章 ユーザ調査	9
3.1 アプリケーションユーザへの手書きジェスチャの調査	9
3.2 被験者	9
3.3 調査手順	9
3.4 調査結果	11
第4章 \$1 アルゴリズム	13
4.1 特徴	13
4.2 4つのステップ	14
4.2.1 リサンプル	14
4.2.2 向きと大きさの正規化	15
4.2.3 位置の正規化	16
4.2.4 類似度を高くするための最適な角度の選定	16
4.3 類似度計算	18
4.4 リミテーション	18
第5章 \$V アルゴリズム	20
5.1 \$V アルゴリズムが目指す特徴	20
5.2 \$V アルゴリズムのアイディア	21

5.2.1	大きさ，向き，位置に関して識別可能にする方法	21
5.2.2	1次元の手書きジェスチャを認識する方法	21
5.2.3	学習データの保持の方法	21
	ジェスチャグループの作成が認識速度の低下を防ぐ理由	22
	ジェスチャグループの作成が認識率の低下を防ぐ理由	22
	ジェスチャグループごとの特徴量の選定	23
5.3	認識に用いる特徴量を選定した時の認識率と認識速度の実験	24
5.3.1	ジェスチャの特徴量と類似度の定義	24
	大きさ	24
	向き	25
	位置	25
5.3.2	ジェスチャグループの作成方法	25
5.3.3	ジェスチャグループにおける認識に用いる特徴量の選定方法	26
5.3.4	ジェスチャの認識方法	26
5.3.5	被験者	28
5.3.6	実験機器	28
5.3.7	実験手順	28
	ジェスチャの取得	28
	認識率と認識速度の測定	29
	N-best List の 1 番目と 2 番目の差と類似度の測定	30
5.3.8	実験結果と考察	30
	認識率	30
	認識速度	30
	N-best Lists の 1 番目と 2 番目の差	31
	ジェスチャが正しく認識された時の類似度	32
	ジェスチャが正しく認識された時の類似度の最小値	32
5.3.9	議論	33
5.4	最適な重み付けのための実験	35
5.4.1	重み付けの手順	35
5.4.2	実験結果	37
5.5	ジェスチャの認識	38
第 6 章	評価実験	42
6.1	実験設計	42
6.2	実験結果と考察	42
6.2.1	認識率	42
6.2.2	認識速度	43
6.2.3	\$V, \$1, Rubine の認識速度	43
6.2.4	DTW の認識速度	43

6.2.5 識別性能の結果	44
N-best Lists の 1 番目と 2 番目の差	45
ジェスチャが一致した時の類似度	45
ジェスチャが一致した時の類似度の最小値	45
第 7 章 アプリケーション例	48
第 8 章 議論	49
第 9 章 結論	50
謝辞	51
参考文献	52

図 目 次

1.1 ストロークの形状と書き順は同じであるが、大きさ (e), 向き (a) (b) (d), 位置 (c) が異なる、単一ストロークからなる手書きジェスチャの例	3
3.1 \$1において用いられる、一般的にスマートフォンやタブレット端末などのタッ チパネルへの手書き入力やペン入力において良く用いられる、単一ストローク からなる手書きジェスチャの例	10
3.2 調査において6人の被験者 (P1 ~ P6) から得られた、用いたい手書きジェスチャ の一覧	12
4.1 Each step in the \$1 algorithm process	14
4.2 Each step in the \$1 algorithm process	15
4.3 Each step in the \$1 algorithm process	15
4.4 Each step in the \$1 algorithm process	16
4.5 Each step in the \$1 algorithm process	16
4.6 Each step in the \$1 algorithm process	17
4.7 Each step in the \$1 algorithm process	18
4.8 Each step in the \$1 algorithm process	19
5.1 大きさ、向き、位置を特徴量として認識のために用いた場合に、入力データと 学習データが一致しない例	23
5.2 ジェスチャグループの学習データ間の類似度 (Sts, Sto, Stp) を求める方法、こ の場合、(0.8, 0.1, 0.4) となる	27
5.3 実験に用いたスマートフォンのスクリーンショット。緑色のエリアがジェスチャ が入力されるエリアである	29
5.4 各手法における、被験者ごとの認識率	31
5.5 各手法における、被験者ごとの認識速度	32
5.6 各手法における、被験者ごとの N-best Lists の 1 番目と 2 番目の差	33
5.7 各手法における、被験者ごとのジェスチャが正しく認識された時の類似度 . .	34
5.8 各手法における、被験者ごとのジェスチャが正しく認識された時の類似度の最 小値	35
5.9 The procedure to decide the optimal weight values	36
5.10 The procedure to decide non optimal weight values	37

5.11	The screen shot on the smartphone. The green area is the input area	39
5.12	The screen shot on the smartphone. The green area is the input area	40
5.13	The screen shot on the smartphone. The green area is the input area	41
6.1	The screen shot on the smartphone. The green area is the input area	43
6.2	The screen shot on the smartphone. The green area is the input area	44
6.3	The screen shot on the smartphone. The green area is the input area	45
6.4	The screen shot on the smartphone. The green area is the input area	46
6.5	The screen shot on the smartphone. The green area is the input area	46
6.6	The screen shot on the smartphone. The green area is the input area	47

第1章 序論

本研究においては，ユーザが独自に定義した手書きジェスチャを高速に認識し，かつどのような開発環境においても実装可能な軽量なアルゴリズムを示す．本章においては，まず初めに研究背景として既存の手書きジェスチャ認識手法とその課題について述べる．次にその課題を解決すべく本研究の目的について述べ，最後に本論文の構成を述べる．

1.1 背景

スマートフォンやタブレット端末のディスプレイへの入力手法として，ペンや指を用いた手書きジェスチャが多く採用されている．特にそれらを入力として用いるようなアプリケーションをプロトタイピングする環境において，手書きジェスチャをアプリケーションに組み込む際に求められることとして，[Ret94]において述べられているように，手書きジェスチャ認識を実現するためのパターン認識に関する専門的な知識 [HTH00, ABS04, SD05, CB05, Pit91, Cho06, Rub91a, AW10] がなくとも，素早く実装できること(すなわちアルゴリズムが簡潔であるということ)，素早く手書きジェスチャ入力のテストができるここと(すなわちあまり学習データを必要としないこと)ことなどが挙げられる．同時に，ジェスチャ認識であるため，認識率が高いこと，認識速度が速いことも求められる．また，手書きジェスチャであるため，入力されるジェスチャは毎回微妙に形状が異なるジェスチャが入力される可能性が高いため，たとえ形状が微妙に異なったとしても意図したジェスチャとして正しく認識されるようなロバスト性の高い認識アルゴリズムであることも求められる．また，学習データをあまり必要としないということは，アプリケーションユーザが独自に手書きジェスチャを定義することが可能なシステムを実現することにもつながり，これも手書きジェスチャを入力として用いるようなアプリケーションを開発する多くの開発者によって求められていることの1つである．

\$1 [WWL07] は，まさにこれらの要件を満たす手書きジェスチャ認識アルゴリズムであり，「アルゴリズムが簡潔である，少ない学習データにおいて高い認識率を示す，認識速度が速い，ロバスト性が高い」といった特徴を持ち，單一ストロークからなる手書きジェスチャを認識可能なアルゴリズムである．\$1 は，現に，ActionScript, Python, C#, C++, Objective-C, Java, Java ME, and JavaScript(全て URL 引用する)といった言語において実装されている．そのため，多くのソフトウェア開発者にとって，手書きジェスチャ認識を自身のシステムに組み込むことが容易となった．特に手書きジェスチャ認識のためのライブラリが存在しないようなプロトタイピング開発環境において，その存在価値は非常に高い．また，\$1 を改良した多くのアルゴリズムは\$-Family Recognizer [AW10, RSH11, Li10, AW12, HS12, VAW12, TL15] として知ら

れ，\$1 の持つ「アルゴリズムが簡潔である」という特徴を維持しつつ，認識率や認識速度の向上，識別可能なジェスチャの種類の増加といった観点から改良が試みられた．ペンや指を入力として用いるようなインターフェースが普及し，それらを用いたソフトウェア開発者が増加している今日において，これらアルゴリズムは，手書きジェスチャ認識を自身のシステムに容易に組み込むための手段として必要とされており，アルゴリズムの開発/改良の機運が高まっている．これまで開発されてきた\$-Family Recognizer は，ジェスチャを構成するストロークの，大きさ，向き，位置に関して，そのいずれかあるいはすべてについて不变になるようなアルゴリズムを採用することにより，不变にしたものについてロバストなジェスチャ認識を実現した．つまり，手書きジェスチャの書き順が同じでありかつ手書きジェスチャを構成するストロークの形状さえ類似していれば，ストロークの大きさ，向きやストロークが入力された位置などが多少異なっても，同じ形状と書き順の手書きジェスチャとして認識されることを示す．その結果，認識率の向上が実現された．それらを不变にしない，つまり特徴量として認識のために用いるということは，その特徴量について識別できるため，識別可能な手書きジェスチャが増加することにつながるが，不变にしない特徴量についてはロバストではなくなるため，認識率の低下を招く恐れがある．例えば，Rubine Classifier [Rub91b] は 13 もの手書きジェスチャの特徴量を用いることにより手書きジェスチャを認識し，手書きジェスチャを構成するストロークの形状と書き順は同じであるが，大きさ，向き，位置に関して異なるジェスチャ(図 1.1)を識別することが可能である．しかしながら，認識に用いる特徴量が多いため，結果的に認識率の低下を招いている．

認識速度の観点でいえば，DTW [Tap82, SC07] は，HCI 分野において，ジェスチャ認識をするためによく採用されているアルゴリズムであり，単純にストロークを構成する点の距離を比較するのみであるため，こちらも，手書きジェスチャを構成するストロークの形状と書き順は同じであるが，大きさ，向き，位置に関して異なるストロークを識別することは可能である．しかしながら，単純なアルゴリズムによって認識率を向上させるために，非常に計算量が大きいといった問題点がある．

しかしながら，第 3 章において示す事前調査により，これまでに述べたきたような，ストロークの形状と書き順は同じであるが，大きさや向きや位置が異なる単一ストロークからなる手書きジェスチャ(図 1.1)をアプリケーションユーザが入力として用いることを要望していることが導かれた．これらは既に述べたように既存の\$-Family Recognizer において識別できないため，これらの手書きジェスチャを入力として用いたいソフトウェア開発者は既存の\$-Family Recognizer を認識アルゴリズムとして用いることができない．また，これらを識別可能な Rubine classifier や DTW などを採用した場合は，認識率や認識速度において十分なパフォーマンスを得られない可能性がある．

1.2 目的

本研究の目的は\$1 を拡張し，単一ストロークからなる手書きジェスチャに対し，ストロークの特徴量である，大きさ，向き，位置に関して識別可能としながらも，\$1 と比較し認識率

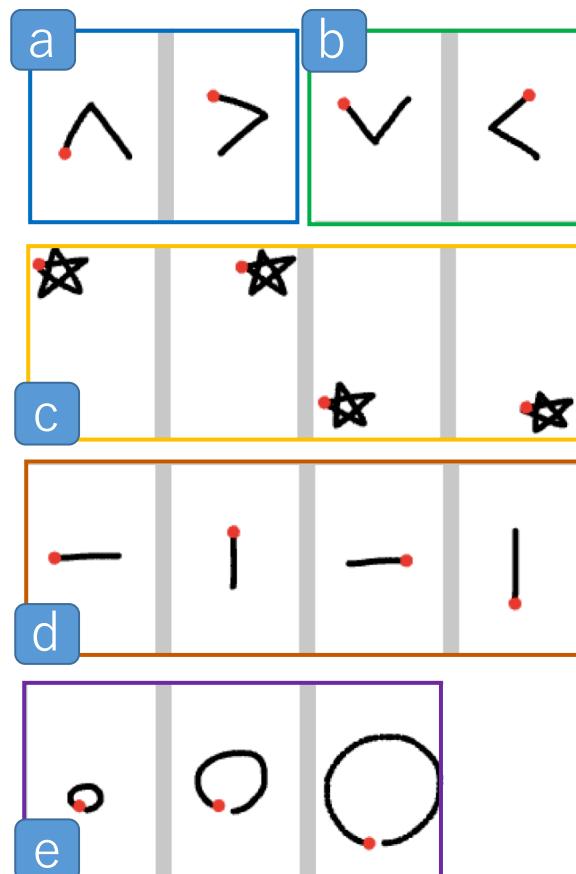


図 1.1: ストロークの形状と書き順は同じであるが、大きさ (e), 向き (a) (b) (d), 位置 (c) が異なる、單一ストロークからなる手書きジェスチャの例

の低下と認識速度の低下を最小限に抑えることを実現した\$-Family Recognizer 開発することである。我々はこの、ストロークの大きさ、向き、位置に関して “V”ariant (不变) なストロークを認識する\$-Family Recognizer を\$V と名付けた。\$V は\$1 と同様、簡単なアルゴリズムによって構成され、どのような開発環境においても実装可能であることも目標としている。また、\$1 と同様、少ない学習データにおいて高い認識率を示すことも目標としており、これは、アプリケーションユーザが手書きジェスチャを独自に定義することが可能であることを示している。また、大きさ、向き、位置に関して識別可能な既存アルゴリズムと比較することによって、\$V の有用性を示すことも本研究における目的とする。

1.3 貢献

本研究における手書きジェスチャ認識アルゴリズム\$V の貢献を以下に示す。

- 手書きジェスチャを構成するストロークの形状と書き順は同じであるが、大きさ、向き、位置に関して異なる手書きジェスチャ識別することが可能なアルゴリズムを開発した。
- 少ない学習データにおいて認識率を向上させるためのアルゴリズムを開発した。
- 認識速度を向上させるためのアルゴリズムを開発した。
- どのような開発環境においても実装可能なアルゴリズムを開発した。

1.4 本論文の構成

第 1 章では、研究背景と目的を述べた。第 2 章では、関連研究を述べる。第 3 章では、本研究の動機にもなった、アプリケーションユーザが入力として用いたい手書きジェスチャの調査について述べる。第 4 章では、\$V の拡張元である\$1 アルゴリズムについて述べ、第 5 章において、\$V アルゴリズムの詳細について述べる。第 6 章では、\$V のアルゴリズムとしての性能評価実験について述べる。第 7 章では、\$V を用いたアプリケーション例を述べる。第 8 章では、\$V アルゴリズムの今後の展望について議論する。第 9 章では、本研究の結論を述べる。なお、付録 A に\$V アルゴリズムの擬似コードを、付録 B に第 3 章のユーザ調査に用いた調査同意書を、付録 C に調査について説明する際に用いた説明書を、付録 D に第 6 章の評価実験に用いた実験同意書を示す。

第2章 関連研究

本章においては、本研究に関する研究について述べる。本研究では、ユーザ定義手書きジェスチャを高速に認識し、かつどのような開発環境においても実装可能な軽量なアルゴリズムを提案している。また、ストロークの大きさ、向き、位置に関して識別可能な\$-Family Recognizerである。よって、本研究の関連研究は、一般的な、手書きジェスチャ認識アルゴリズムの研究、ユーザ定義に特化した手書きジェスチャ認識アルゴリズムの研究、手書きジェスチャ認識を可能にするツールキット、\$-Family Recognizerに関する研究、手書きジェスチャの評価に関する研究に分類される本章においては、それらの研究について述べた後、最後に本研究における手書きジェスチャ認識アルゴリズムである\$Vの位置づけについて述べる。

2.1 手書きジェスチャ認識アルゴリズム

文字、ストロークの形状、手書きジェスチャなどの認識は、長く広く研究されている分野であり、多くのアルゴリズムにより実現されてきた、finite state machines [HTH00] (有限オートマトン)は、有限個の状態と遷移と動作の組み合わせからなる論理モデルであり、ある「状態」において、何らかのイベントや条件によって別の状態へ「遷移」することを繰り返すことで最終的な認識結果を導く。高い認識精度を示すためには、より詳細なモデルの定義が必要となる。Hidden Markov Models (HMMs) [ABS04, SD05, CB05] (隠れマルコフモデル)は、観測された出力の系列から、内部の状態系列を統計的に推測するためのアルゴリズムである。neural networks [Pi91] は脳機能の特性を計算機上に応用したアルゴリズムであり、大量の学習によってモデルを最適化し、多次元量のデータで線形分離不困難な問題に対して比較的小さい計算量で良好な解を得ることができる。feature-based statistical classifiers [Cho06, Rub91a] は、大量の学習データによる特徴量をもとに学習データをクラスタリングし、より低次元な認識モデルを生成するためのアルゴリズムである。ad hoc heuristic recognizers [AW10, WS03] は、「限定的な」認識アルゴリズムであるため、事前に定義されたジェスチャのみ認識することができます。アプリケーション実行時において、新たな学習データを追加した場合に、新たなヒューリスティック関数を定義しなければならないため、アプリケーションユーザが独自にジェスチャを定義することができない。template matching [KS05, KZ04] は、主に画像処理として用いられ、学習データと入力データの画像をそれぞれ走査し、画像上の各位置における類似度を算出するアルゴリズムであり、手書き文字にも応用されている。

これらアルゴリズムはオンライン文字認識及びオフライン文字認識に双方においてしばしば用いられるアリゴリズムである。オンライン文字認識とは、ディスプレイなどにペンや指

などによって入力された文字を認識する技術の総称であり，オフライン文字認識とは，紙に書かれた文書イメージを光学スキャンし，そのイメージを自動的にコンピュータで処理可能なテキストデータに変換する技術の総称である．しかしながら，これらアルゴリズムは，高い認識精度を示す認識モデルを生成するために膨大な数の学習データが必要であり，素早く手書きジェスチャ入力のテストしたい場合において不向きであるだけでなく，アプリケーションユーザが独自にジェスチャを定義する上で実用的であるとは言えない．また，これらアルゴリズムを実装することは，本分野に精通していない開発者にとって困難である．

2.2 ユーザ定義に特化した手書きジェスチャ認識アルゴリズム

Rubine classifier [Rub91b] や Dynamic programming (DTW) [Tap82] などは，少ない学習データにおいてジェスチャ認識可能なアルゴリズムであるが，Rubine classifier はアルゴリズムに用いられる数式が複雑である上，学習データが少ない場合は認識率が高いとは言えない．また，DTW はアルゴリズムが簡潔であるが，計算量が非常に大きいという問題点がある．計算量を改善した Fast DTW [SC07] が開発されたが，アルゴリズムは複雑になっており，プロトタイピング環境開発向けとは言い難い．また，Rubine は多くある特徴量を適切に選ばないと，認識率が低下するという欠点があり，特徴量の選定が難しい．このように認識に用いる特徴量を単に増やすことは，それについて識別できることにつながるが，ロバスト性に欠ける欠点もある．

2.3 手書きジェスチャ認識を可能にするツールキット

プロトタイピング環境向けに開発できるように，ジェスチャ認識を簡単に開発可能なツールキットも開発された．SATIN [HL00] はペンベースのユーザインタフェースであり，ジェスチャ認識のモデルを手書きによって定義することができ，ジェスチャ認識の開発を容易にしたツールキットである．Henry et al. [HHN90]，Landay and Myers [LM93] によるツールキットは～，Amulet toolkit [MMM⁺97] は～がある．これらは，開発を手助けするのに非常に強力であるが，対応可能な開発環境が決まっており，自身の環境に適用できない場合がある．

2.4 \$-Family Recognizer

\$1 [WWL07] は，プロトタイピング環境において実装可能であり，少ない学習データにより，高い認識率を示すアルゴリズムであるため，ユーザが独自にジェスチャを定義するジェスチャ認識システムを開発することが可能である．プロトタイプ開発者は，自身のシステムに，簡単な数式のみを含む，およそ 100 行からなるアルゴリズムを追加するのみによって，図のような单一ストロークからなるジェスチャ認識を行うことが可能である．\$1 は geometric template matching approach を用い，candidate gesture と template gesture の対応する点のユークリッド距離が最小となるような最適な角度を探索することによってジェスチャ認識を行っている．しか

しながら，ストロークを，大きさ，向き，位置に不变にしているため，それらの特徴量に依存するようなストロークを認識することができないため，そのようなストローク認識を行いたい開発者は\$1 を自身のシステムに採用することができない。\$1 が識別/認識することができないストロークを識別/認識するために，これまで\$1 を拡張したアルゴリズムが開発されてきた。\$N [AW10] は，複数のストロークからなるジェスチャを認識することを可能にし，識別可能なストロークを大幅に増やすことに成功した。\$N は，ストロークを複数の单一ストロークに分割し，それぞれの单一ストロークを\$1 の手法によって認識した。また，自動的に all possible permutations of a multistroke を計算することによって，ストロークの向きや書く順番にロバストな認識も可能にした。Quick\$ [RSH11] は，\$1 の改良であり，Hierarchical Clustering によって，対応する点のユークリッド距離が最小となる candidate gesture と template gesture の組み合わせを効率的に探索し，認識速度を高速化することに成功した。Protractor [Li10] も，\$1 に対し，認識速度の面において改良し，最適な角度を探索する際に，GSS [PTVF92](pp. 397-402) を用いた\$1 とは異なり，a closed-form solution を用いることによって，より高速に探索することを可能とした。\$N-Protractor [AW12] は，\$N に対し，Protractor の手法を用いることによって，より高速にかつ，より正確に複数のストロークからなるジェスチャを認識することを可能にした。1 cent Recognizer [HS12] は，\$1 よりも高速であり，アルゴリズムも非常に単純であるため実装が容易であるが，認識/識別可能なジェスチャの種類や，認識率の観点から見るとあまり実用的であるとは言えない。\$P [VAW12] は，ストロークを構成する点を Point Cloud として扱うことによって，\$N-Protractor よりも，memory や execution costs の点において効率的なアルゴリズムを示した。Penny Pincher [TL15] は，ストロークを構成する点間のベクトルを用いることによって，これまでの\$-Family Recognizer と比べて，より高速にかつ，正確に認識することを可能にした。

これらの\$-Family Recognizer は，2D のストローク認識をすることが可能であり，\$1 に対し，アルゴリズムを簡略化したり，認識速度を高速化したり，認識率を高くしたり，認識できるストロークの種類を増やしたりするなどして，繰り返し改善してきた。

しかしながらこれらのアルゴリズムは，ストロークの特徴量である，大きさ，向き，位置に関して，そのいずれかあるいはすべてについて不变になるようなアルゴリズムを採用することにより，それらの特徴量についてロバストなジェスチャ認識を実現し，その結果認識率や，認識速度の向上を実現してきた。それらを不变にしないことは，その特徴量について識別できるようになることにつながるが，不变にしない特徴量についてはロバストではなくなるため，認識率の低下や認識速度の低下を招く恐れがある。例えば，1 cent Recognizer はストロークを構成するすべての点の中心座標から，それぞれの点へのユークリッド距離のみを特徴量とし，candidate gesture と template gesture の特徴量の差が最小となるストロークを探査しているため，ストロークの形状は同じであるが，大きさ，向き，位置に関して異なるストロークを識別することは可能である。しかし，それぞれの特徴量についてロバストでないため，それぞれの特徴量について微妙に異なる場合，認識できないことが多々あり，結果的に認識率の低下を招く。認識速度の観点でいえば，DTW は，単純にストロークを構成する点の距離を比較するのみであるため，こちらも，ストロークの形状は同じであるが，大きさ，向

き，位置に関して異なるストロークを識別することは可能であるが，認識率を向上させるために，非常に計算量が大きい．

2.5 手書きジェスチャの評価研究

2.6 本研究の位置づけ

第3章 ユーザ調査

本章においては、本研究の動機にもなった、アプリケーションユーザが入力として用いたい手書きジェスチャの調査内容とその結果について述べる。

3.1 アプリケーションユーザへの手書きジェスチャの調査

単一ストロークからなる手書きジェスチャ認識を用いたシステムが、これまでにも多く開発されてきた。その中で、\$1 は、[HL00, LM93, LNHL00]において用いられているよう、一般的にスマートフォンやタブレット端末などのタッチパネルへの手書き入力やペン入力において良く用いられる、単一ストロークからなるジェスチャを抜粋し(図 3.1)，それらについて認識率を計測した。[それぞれについて詳しく書く](#)。しかしながら、ユーザ定義ジェスチャを用いた研究 [Vat12, BNLH11, WMW09, SMSJ⁺15] の中には、単一ストロークからなるジェスチャであり、かつ、ストロークの形状は同じであるが、ストロークの向きや位置の違いを利用したジェスチャを入力に用いる場合があった。

そこで、普段スマートフォンを利用する場面、およびスマートフォンを入力デバイスとし PC を操作する場面を想定した時、どのようなアプリケーションに対し、どのような手書きジェスチャを入力として用いたいかを事前調査した。

3.2 被験者

被験者は普段からスマートフォンを使用している大学院生の男性 6 名である。年齢は 21 ~ 27 歳(平均 23.8 歳)であり、全員右利きであった。6 名の被験者の中にはコンピュータサイエンスあるいはユーザインターフェースを専攻している人が 4 名存在し、残りの 2 名は、社会工学を専攻している男性と、エンジニアとして働いている男性であった。被験者は全員日頃からスマートフォンを使用していた。

3.3 調査手順

我々はまず、被験者に調査の目的を説明した。その後、普段スマートフォンを利用する場面、およびスマートフォンを入力デバイスとし PC を操作する場面を想定した時、どのようなアプリケーションに対し、どのような手書きジェスチャを入力として用いたいかを 20 以上考えるよう指示し、自身のスマートフォンに対し実際に入力しながら考えるよう指示した。そ

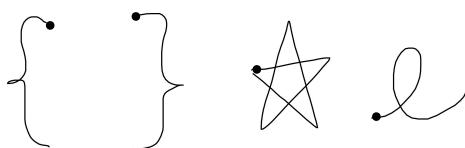
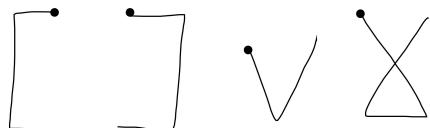
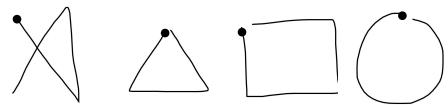


図 3.1: \$1において用いられる，一般的にスマートフォンやタブレット端末などのタッチパネルへの手書き入力やペン入力において良く用いられる，単一ストロークからなる手書きジェスチャの例

の際，ジェスチャを入力する姿勢は次の3つの姿勢(姿勢1，姿勢2，姿勢3)から，実際に自分がそのジェスチャを入力として用いるときに入力する姿勢として1つ選んでもらった．

1. 机にスマートフォンをおいて，利き手の人差し指によって入力する．
2. 利き手でスマートフォンを握りながら，同じ利き手の親指によって入力する．
3. 利き手とは反対の手でスマートフォンを握りながら，利き手の人差し指によって入力する．

被験者は，全ての姿勢を座って入力するよう指示された．また，入力ジェスチャとして，単一ストロークからなるよう指示した．

入力として用いたい手書きジェスチャを1つ考えるたびに，我々は付録[何の付録か](#)に示す紙にそのジェスチャをボールペンによって書くよう指示した．またそれに加え，そのジェスチャを入力した姿勢(1~3)，そのジェスチャをどのアプリケーションに対して用いたいのかも同時に書くよう指示した．

3.4 調査結果

図3.2は、6人の被験者から得られた手書きジェスチャー一覧である。

図3.2において、例えば音楽再生アプリケーションにおいて、早送り、巻き戻し、音量の上げ下げ、といったような前後への移動や、値の上げ下げなど、対になるような操作に対して、向きや大きさの違いを用いて入力する要望があった。また、ブラウザアプリケーションにおいて、位置によって登録先のブックマークを変える、ページ内における表示位置をジェスチャの入力位置に対応させる、といったように、位置の違いを操作対象先の違いに割り当てたり、現在操作しているものの位置に対応付けるといった要望があることがわかった。

また、片手で操作する場面(姿勢2)においては、難しいストローク操作ができないため、極力簡単なストロークを用い、かつ、大きさ、向き、位置の特徴量を利用して入力する要望があることも分かった。[ここら辺もうちょい詳しく書く？](#)

\$Vは、図3.2が示すような、ジェスチャの形状や書き順は同じでも、大きさ、向き、位置に関して異なる手書きジェスチャを認識することが可能なアルゴリズムであり、今回調査を行った被験者の要望を満たすことのできる手書きジェスチャ認識である。

P 1	
P 2	
P 3	
P 4	
P 5	
P 6	

図 3.2: 調査において 6 人の被験者 (P1 ~ P6) から得られた、用いたい手書きジェスチャの一覧

第4章 \$1アルゴリズム

\$V は \$1 の拡張であるため、本章において \$1 アルゴリズムを述べる。

4.1 特徴

ユーザが入力した手書きジェスチャは、図 4.1 のように複数の点によって構成され、すでに登録された手書きジェスチャと、それぞれの点を比較することによって、どの手書きジェスチャと一致しているかが判別される。しかしながら、これらの手書きジェスチャを構成する複数の点は、入力に用いられるハードウェアやソフトウェアに依存した速度によってサンプリングされる。それらに加え、人間によって入力される手書きジェスチャはばらつきがあるため、入力される手書きジェスチャを構成する点の数は入力されるたびに異なる。そのため、ユーザが入力した手書きジェスチャと、すでに登録された手書きジェスチャを比較するにあたり、手書きジェスチャを構成する点どうしを単純に比較することは困難であると言える。

例えば、図 4.1 の手書きジェスチャは、手書きジェスチャを構成する点の数が異なる。それだけでなく、ジェスチャ自体の大きさも異なる。こういった問題点は、手書きジェスチャ認識においてジェスチャを比較する上で、1 つのハドルとなっている。これらの手書きジェスチャによって生じる問題点に対処しつつ、高い認識率を示し、かつどのような開発環境においても実装可能なアルゴリズムを実現するために、\$1 は以下のようないくつかの基準に従うようなアルゴリズムの実現を目指した。

- ハードウェアやソフトウェアのセンシング及び入力する速度などによって変わるサンプリングされる点の数の違いに対してロバストであること。
- 手書きジェスチャの大きさ、向き、位置に不变な認識をすること。
- 数学的な高度な知識やテクニックを必要としないこと（例えば、逆行列、微分、積分など）
- 少ないコードによって実装できること。
- 認識速度が速いこと。
- ソフトウェア開発者やアプリケーションユーザが、独自に手書きジェスチャを定義できること。

- N-best list に関して，高い識別能力を示すスコアを示すこと .
- 図 3.1 のような单一ストロークからなる手書きジェスチャを認識するにあたり，HCI 分野において多く用いられる既存の複雑な手書きジェスチャ認識アルゴリズムと比べても，高い認識率を示すこと .

ここで，N-best list とは，N 個の学習データそれぞれに対する入力データとの類似度を降順に並べたものであり，N-best list の 1 番目と 2 番目のスコアの差が大きいほど，高い識別能力を示していると言える .

次に，これらの基準に従うような \$1 のアルゴリズムを述べる . アルゴリズムは大きく 4 つのステップから構成される .

4.2 4 つのステップ

前節において述べた基準を満たすために，入力データ及び学習データは 4 つのステップを経た後に比較される . 4 つのステップは，リサンプル，向きの正規化，大きさと位置の正規化，類似度を高くするための最適な角度の選定からなる .

4.2.1 リサンプル

前節において述べたように，手書きジェスチャを構成する点の数は，ハードウェアやソフトウェアのセンシング及び入力する速度などによって変わる . 特に，入力速度の違いによる点の数の違いは顕著である (図 4.1) .

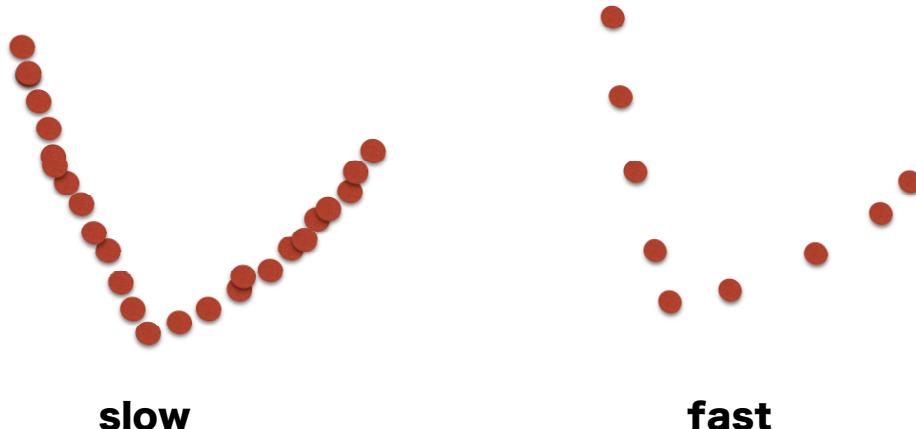


図 4.1: Each step in the \$1 algorithm process

点の数が違うことにより，入力データと学習データの手書きジェスチャを構成する点を互いに比較することが困難となっている . そこで，図 4.2 に示すように N 個の等間隔に並ぶ点

にリサンプルすることとする。N個の点にリサンプルすることは、生のデータを扱うことと比べて正確なデータを扱っているとは言えず認識精度が落ちる可能性があるが、入力データと学習データ双方の手書きジェスチャの点の数が等しくなるため、容易に互いの対応する点を比較できる。

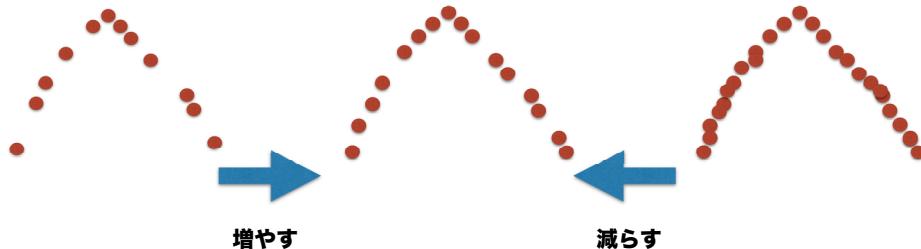


図 4.2: Each step in the \$1 algorithm process

リサンプルすることは、既存の手書きジェスチャ認識アルゴリズムにおいても用いられている [PS00, TSW90, KZ04, ZK03, Tap82]。

また、 $32 \leq N \leq 256$ の時、 $N=64$ の場合に認識率、認識速度双方において高いパフォーマンスが得られることも\$1において報告されている。

4.2.2 向きと大きさの正規化

リサンプルされた手書きジェスチャの向きを“indicative angle”として定義する。indicative angle とは図 4.3 のように、手書きジェスチャの書き始めの一一番最初の点の座標、手書きジェスチャを構成する点全てによる中心座標、0 度方向によって形成される角度である。そして、indicative angle の沿って、全てのジェスチャを回転させる。これにより、ジェスチャは向きに正規化される。

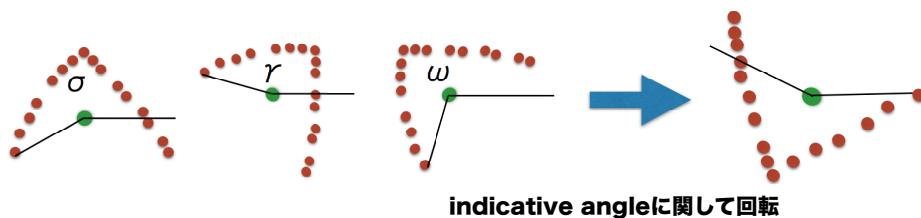


図 4.3: Each step in the \$1 algorithm process

向きによって正規化されたジェスチャの大きさを“boundingbox”として定義する。boundingbox とは図 4.4 のように、ジェスチャに隣接するような矩形である。全てのジェスチャについて、boundingbox をある一定の大きさの矩形に正規化することによって、ジェスチャは大きさに正規化される。

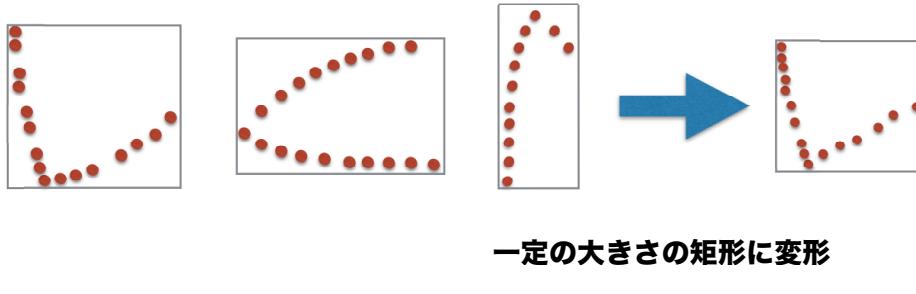


図 4.4: Each step in the \$1 algorithm process

4.2.3 位置の正規化

向きと大きさによって正規化されたジェスチャの位置をジェスチャを構成する全ての点による中心座標として定義する(図 4.5)。全てのジェスチャについて $(0, 0)$ へと移動させることによって、位置に正規化される。

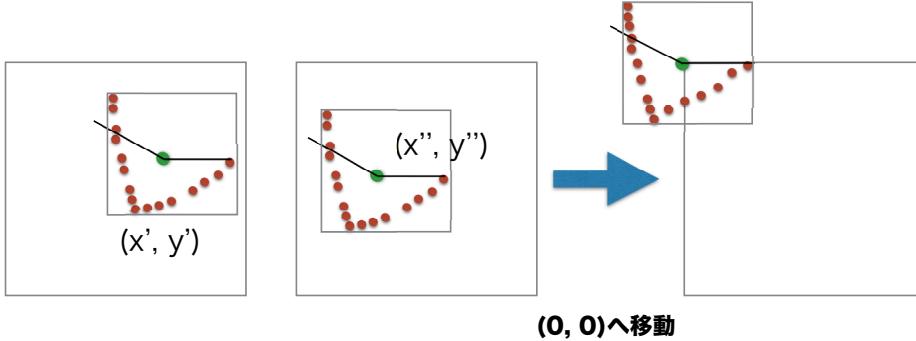


図 4.5: Each step in the \$1 algorithm process

4.2.4 類似度を高くするための最適な角度の選定

リサンプルされた 2 つの手書きジェスチャ構成する点を 1 つずつ対応する点どうし比較するにあたり、1 度ずつ手書きジェスチャを回転させながら、最も類似度が高くなるような角度を見つけた上で、類似度を算出する方法 [KS05] がある。しかしながらこの方法は認識のために膨大な時間を要することになる。全ての学習データの数が 30 くらいの場合そのような方法でも十分な速度によって認識することが可能であるが、\$1 は黄金分割探索 [PTVF92] を用いることによって最も類似度が高くなるような角度を求める。黄金分割探索とは、单峰関数(極値が 1 つしかない関数)において、極値を求めるための方法(局所探索法)のうち効率的な方法の 1 つであり、極値が存在することが自明な範囲において極値を逐次的に求める方法である。

例えば図 4.6 のように、 $f(x)$ の関数があり、極小値 $f(x')$ を求める時に、 x_1 と x_3 の間に極値が存在することが自明な時に、その範囲内に存在する $f(x_2)$ を求める。この時 x_2 は $(x_2 - x_1)$

: $(x_3 - x_2)$ が黄金比 ($1 : \frac{1+\sqrt{5}}{2}$) となるように設定する。これが黄金分割探索と言われる所以であり、常に 3 点 (この場合 x_1, x_2, x_3) が存在する。その後広い区間 (この場合 x_2 と x_3 の間) において、同様に黄金比によって分割し新たな $f(x_4)$ を得る。この時、 $f(x_{4a})$ ならば、極小値は x_1 と x_4 の間に存在するため、新たな 3 点は x_1, x_2, x_4 となる。 $f(x_{4b})$ ならば、極小値は x_2 と x_3 の間に存在するため、新たな 3 点は x_2, x_4, x_3 となる。このように、極小値が存在する範囲を徐々に狭めていくことによって、効率よく極値を求めることができる。

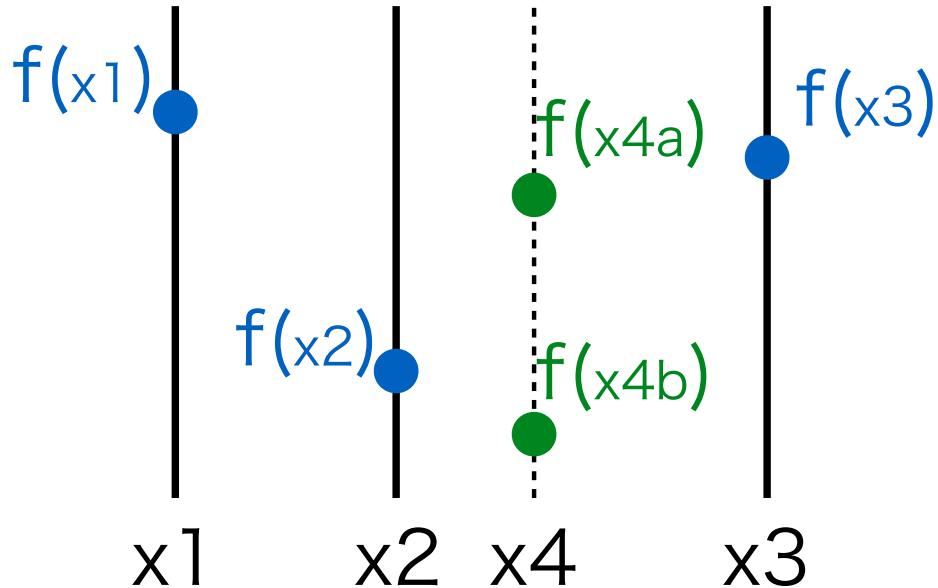


図 4.6: Each step in the \$1 algorithm process

\$1 の場合、480 個の手書きジェスチャにおいて、+45 度の範囲において極小値が存在することが発見され、極小値が存在する範囲が 2 度になるまで黄金分割探索を行う。この時、入力データに類似する学習データが存在する場合あるいは存在しない場合においても、10 ステップ後には極小値が求められることが発見された。

局所探索法の 1 つである山登り法(引用)を黄金分割探索の代わりに用いる場合、480 個の手書きジェスチャにおいて、類似するジェスチャの場合およそ 7.2 ステップ後に極小値を求めることができるが、類似しないジェスチャの場合はおよそ 53.5 ステップ後に極小値が求められることが発見された。つまり学習データが 10 ズつ存在する 16 種類の手書きジェスチャ=160 個のジェスチャにおいて、黄金分割探索は $160 \times 10 = 1600$ ステップ必要であるのに対し、山登り法は $7.2 \times 10 + 53.5 \times 150 = 8097$ ステップ必要であり、およそ 80.2% もの計算量の節約となっている。

このように、類似度を高くするための最適な角度を黄金分割探索によって効率的に求めることによって、認識速度の向上を実現している。

以上の 4 ステップをまとめ、入力データと学習データそれぞれを比較するまでの過程を図 4.7 に示す。

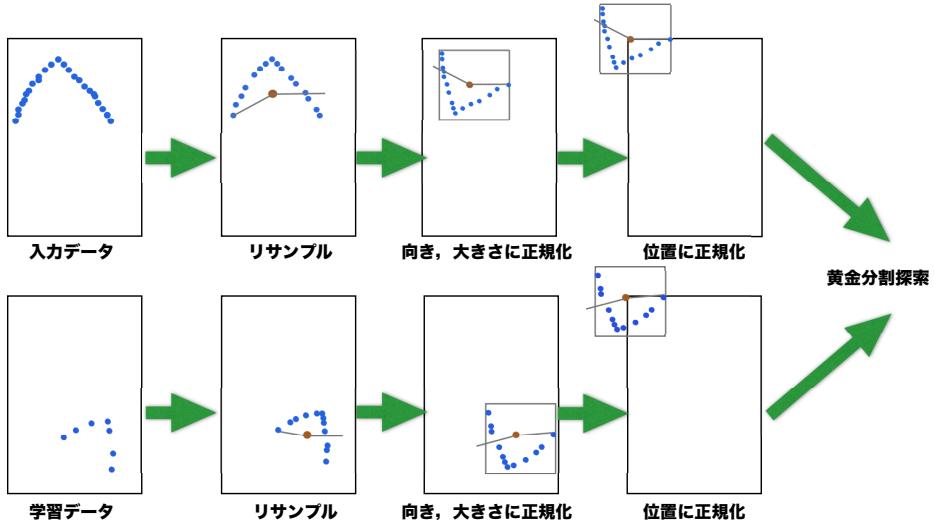


図 4.7: Each step in the $\$1$ algorithm process

4.3 類似度計算

前節までの 4 ステップによって得られた入力データと学習データの最終的な類似度は式 (1) によって表される .

$$score = 1 - \frac{d}{\frac{1}{2}\sqrt{size^2 + size^2}} \quad (4.1)$$

ここで , d は対応する点どうしのユークリッド距離の全ての点に関する平均値であり , $size$ は大きさに正規化するに用いられる矩形(正方形)の 1 辺の長さである .

4.4 リミテーション

$\$1$ は , 手書きジェスチャを , 大きさ , 向き , 位置に正規化することによってアルゴリズム全体を簡潔化したり , それぞれについてロバストな認識を可能にし認識率の向上を実現した . しかしながら , このことが原因により , 幾つかのリミテーションが存在する .

- 手書きジェスチャを大きさ , 向き , 位置によって識別しない .
- 直線のような 1 次元の手書きジェスチャを認識することができない .
- 手書きジェスチャを入力する速度による識別ができない .

「手書きジェスチャを大きさ , 向き , 位置によって識別しない」ことに関しては , それぞれについて正規化しないようにアルゴリズムを変えることによって識別することが可能となる . しかしながら , 正規化しないことにより , 正規化しないものに関してはロバスト性が低下するため , 結果的に認識率の低下を招く恐れがある .

「直線のような1次元の手書きジェスチャを認識することができない」ことに関しては、向き、大きさに関して正規化した際に図4.8のように、ストロークを構成する各点が分散してしまうためである。

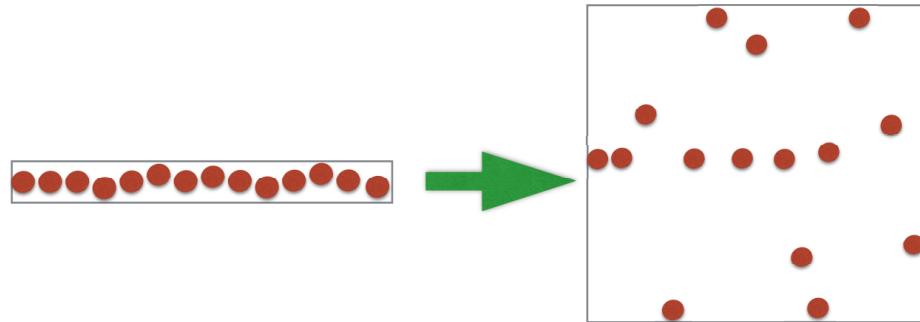


図4.8: Each step in the \$1 algorithm process

「手書きジェスチャを入力する速度による識別ができない」ことに関しては、入力速度の要素を特徴量として用いることによって識別可能となる。例えば、Rubine [Rub91b] は速度を特徴量として用いている。このように認識に用いる特徴量が複数存在する場合、用いる特徴量を適切に選択できれば、つまり、認識に本当に必要な特徴量のみを用いて、認識には必要のない特徴量を選択できれば、認識率の高い手書きジェスチャ認識アルゴリズムとなる。しかしながら、一般的に手書きジェスチャ認識アルゴリズムに関する深い知識がない限り、そのような適切な選択は困難である。\$1は、識別可能な特徴量が少ないが、そのようなわざわしさを一切省いたアルゴリズムとなっている。

第5章 \$Vアルゴリズム

本章において，\$1を拡張した\$Vアルゴリズムを述べる．

5.1 \$Vアルゴリズムが目指す特徴

\$Vアルゴリズムが目指す特徴を以下に示す．

- \$1の特徴を維持すること．つまり，
 - ハードウェアやソフトウェアのセンシング及び入力する速度などによって変わるサンプリングされる点の数の違いに対してロバストであること．
 - 数学的な高度な知識やテクニックを必要としないこと（例えば，逆行列，微分，積分など）
 - 少ないコードによって実装できること．
 - 認識速度が速いこと．
 - ソフトウェア開発者やアプリケーションユーザが，独自に手書きジェスチャを定義できること．
 - N-best listに関して，高い識別能力を示すスコアを示すこと．
 - 図3.1のような単一ストロークからなる手書きジェスチャを認識するにあたり，HCI分野において多く用いられる既存の複雑な手書きジェスチャ認識アルゴリズムと比べても，高い認識率を示すこと．
- 前項目を満たした上で，形状や書き順が同じ手書きジェスチャを大きさ，向き，位置に関して識別可能にすること．

これまで述べてきたように，一般的に，特徴量を不变にすることによってその特徴量についてロバスト性が向上する．しかしながら，不变にしない場合，つまり，認識に用いる特徴量として扱う場合，ロバスト性が低下し，結果的に認識率の低下を招く恐れがある．つまり，\$1アルゴリズムを踏襲した上で，大きさ，向き，位置を特徴量として用い，それらに関して識別可能にすることとは，\$1と比べて，認識率が低下すると言える．以上を踏まえ，\$1に比べて，認識率や認識速度を著しく損なうことなく，図1.1のような，手書きジェスチャの形状と書き順は同じでも，大きさ，向き，位置が異なるジェスチャを識別するアルゴリズムを実現することが\$Vの目指すところである．

5.2 \$V アルゴリズムのアイディア

ここでは，\$V アルゴリズムのアイディアを述べる。

まず，形状や書き順が同じ手書きジェスチャを大きさ，向き，位置に関して識別可能にする方法について述べる。次に，\$1において認識できなかった1次元の手書きジェスチャを認識する方法について述べる。そして，\$V アルゴリズムの最大の特徴である，学習データの保持の方法について述べる。

5.2.1 大きさ，向き，位置に関して識別可能にする方法

\$1 アルゴリズムを活用し，ジェスチャを大きさ，向き，位置によって識別可能にするための方法として以下の2つが考えられる。

1. 単純にリサンプリングした点のみによって判別する(正規化しない)。
2. 正規化した上で，それぞれを特徴量として用いる。

1.の場合，リサンプリングしただけの実質生データのまま比較するため，大きさ，向き，位置によって識別可能となる。しかしながら，手書きジェスチャの場合，アプリケーションユーザの入力は毎度微妙に異なることが予想される。そのため，類似したジェスチャにおいても，類似度が低くなり，ロバスト性が低下する恐れがあるとともに，認識されたか否かを判別するための類似度の閾値の設定が困難になることが予想される。2.の場合，ジェスチャを正規化するためロバスト性は維持され，その上で，大きさ，向き，位置を特徴量として用いるため，1.の場合と比べて，類似度が低くなりづらくなると予想される。そこで，\$V は2.の方法を用いることとする。

5.2.2 1次元の手書きジェスチャを認識する方法

\$V は\$1 とは異なり，1次元の手書きジェスチャを認識できるようにした。これは\$N[]において実装されているため，\$Nにおいて用いられている方法を用いる。具体的には，\$1において大きさを正規化する際に用いられる，手書きジェスチャに隣接する矩形の縦の長さが横の長さに対し(あるいはその逆が)0.3未満の長さであれば，その手書きジェスチャは大きさに正規化しない。これにより，1次元の手書きジェスチャを認識することが可能となる。

5.2.3 学習データの保持の方法

\$V は学習データの保持の方法において特徴がある。

\$V は学習データが追加されるたびに，ジェスチャの形状と書き順が同じ学習データを同じグループに分類する。ここでは形状と書き順が同じジェスチャを識別可能な\$1 アルゴリズムを用いている。この形状と書き順に従って分類されたジェスチャを“ジェスチャグループ”と

名付ける。ジェスチャグループの例を図??に示す(あるいは図1.1もジェスチャグループの例である)。

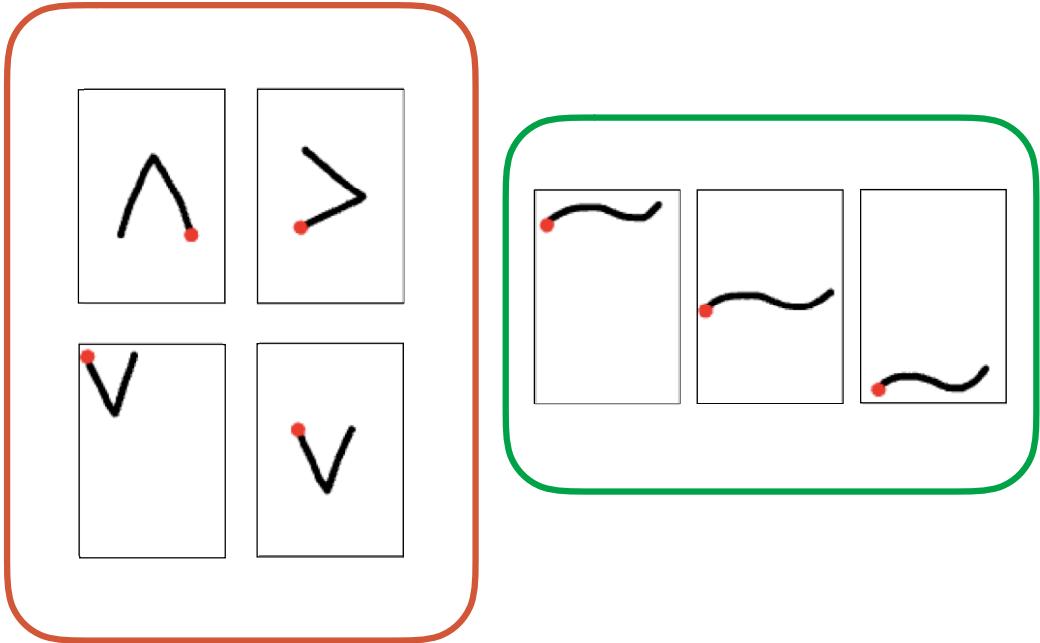


図5.1: ジェスチャグループの例

このようにジェスチャグループを作成する理由は2つある。

- 認識速度の低下を防ぐため。
- 認識率の低下を防ぐため。

それについて理由を述べる。

ジェスチャグループの作成が認識速度の低下を抑える理由

大きさ、向き、位置を特徴量として認識に用いる場合、それについての類似度計算を行うこととなる。これを全てのジェスチャについて類似度計算を行った場合、認識速度が低下する要因となる。 $\$V$ の目的は、ジェスチャの形状と書き順が同じであるが、大きさ、向き、位置に関して識別可能にすることである。そこで、形状と書き順が同じジェスチャが集まつたジェスチャグループを作成し、ジェスチャグループ内に存在する学習データのみに対し、大きさ、向き、位置の類似度計算をする(図??)。一般的に、認識速度は、認識に用いる特徴量を増やした場合、増やさない場合と比べて、学習データの数に比例してその差が大きくなるが、同一ジェスチャグループ内のみに対し認識に用いる特徴量を増やすことによって、全体的な認識速度の低下を防ぐことが可能となると考え、我々は「ジェスチャグループを作成し、

ジェスチャーグループ内に存在する学習データのみに対し、大きさ、向き、位置の類似度計算をすると認識速度の低下を抑えることができる」という仮説を立てた。

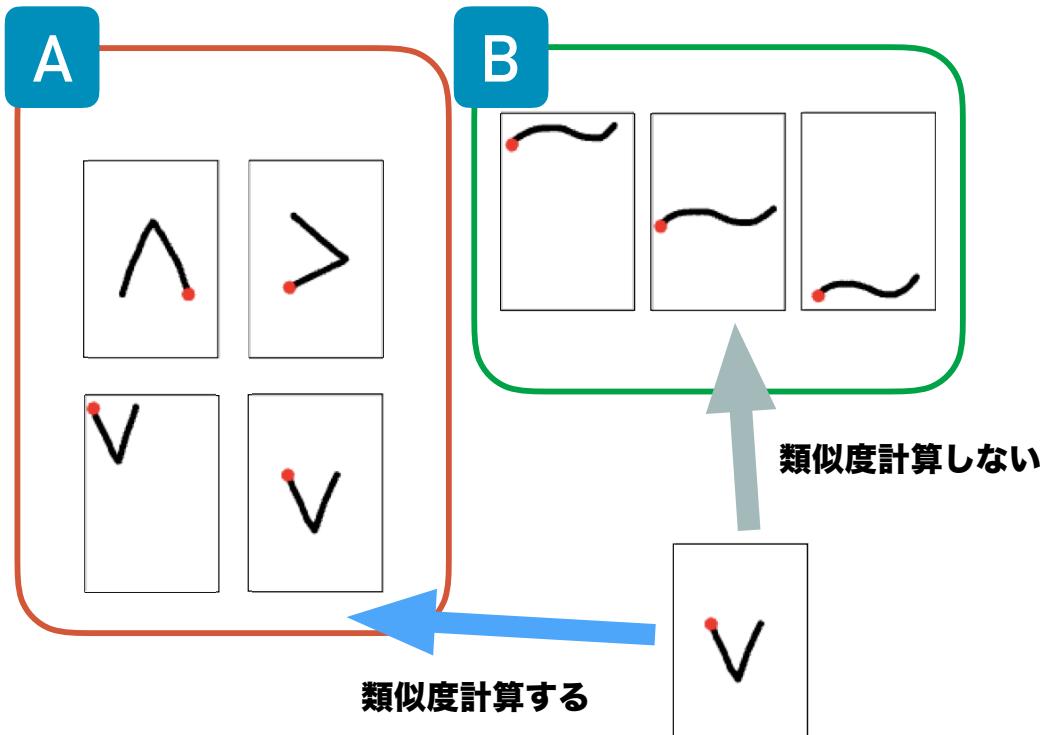


図 5.2: ジェスチャーグループの例

ジェスチャーグループの作成が認識率の低下を抑える理由

大きさ、向き、位置の特徴量を認識のために全てのジェスチャに対し用いた場合を考える。例えば、図 5.1A の場合について考える。学習データ(図 5.1A')が図 5.1A の右下のジェスチャと一致させようと入力されたとする。しかし、この 2 つのジェスチャは大きさが異なるため、大きさを認識のための特徴量として用いている限り類似度が低下し認識率が下がることが考えられる。また、これはロバスト性の低下につながる。

図 5.1B の場合について考える。学習データ(図 5.1B')が図 5.1B の右のジェスチャと一致させようと入力されたとする。しかし、この 2 つのジェスチャは向きが異なるため、向きを認識のための特徴量として用いている限り類似度は低下し認識率が下がることが考えられる。また、これはロバスト性の低下につながる。

図 5.1C の場合について考える。学習データ(図 5.1C')が図 5.1C のジェスチャと一致させようと入力されたとする。しかし、この 2 つのジェスチャは大きさ、向き、位置が異なるため、大きさ、向き、位置を認識のための特徴量として用いている限り類似度は低下し認識率が下がることが考えられる。また、これはロバスト性の低下につながる。

これまでに述べてきたが、大きさ、向き、位置を認識のための特徴量として新たに用いることによって、それについてロバスト性が低下、つまり、認識率の低下を招く恐れがある。図 5.1 はその例である。

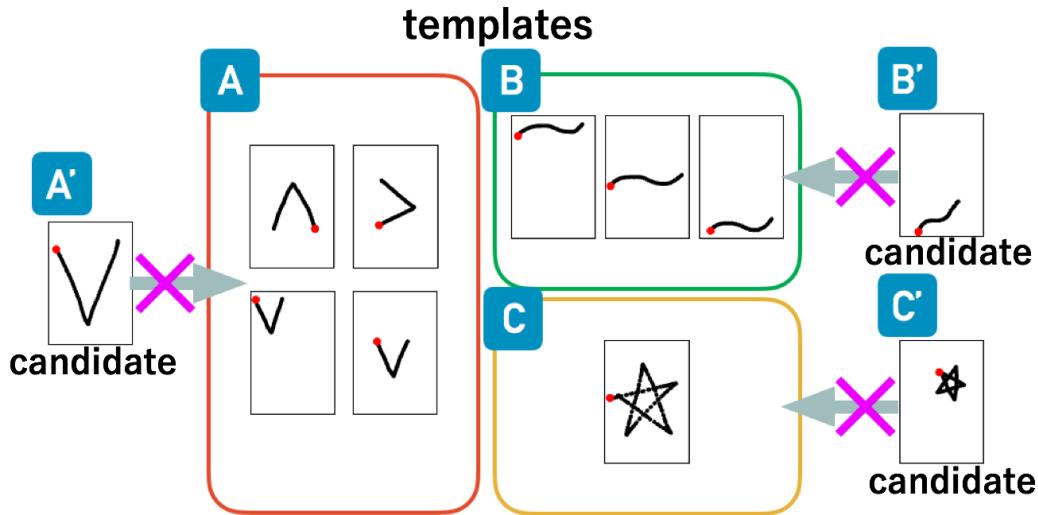


図 5.3: 大きさ、向き、位置を特徴量として認識のために用いた場合に、入力データと学習データが一致しない例

そのため、\$V では、大きさ、向き、位置を認識のための特徴量として単に用いるのではなく、ジェスチャグループごとに、大きさ、向き、位置のうちどの特徴量を認識に用いるか選ぶ、つまり、識別するために必要な特徴量を選ぶという処理を施すことによって、認識率の低下を防ぐことが可能となると考えた。

ジェスチャグループごとの特徴量の選定

ジェスチャグループごとに、大きさ、向き、位置のうちどの特徴量を用いるかを選ぶための方法を述べる。

\$V の目的は、ジェスチャの形状と書き順が同じであるが、大きさ、向き、位置に関して識別可能にすることである。つまり、同一ジェスチャグループにおいて、大きさ、向き、位置のいずれかあるいはすべてが異なるジェスチャを識別できれば良い。

ここで、図 5.1A の場合について考える。図 5.1A のジェスチャグループには、ジェスチャの大きさはどの学習データも同じであるが、向きや位置が異なるジェスチャが存在している。つまり、これらを識別するためには、向き、位置を特徴量として認識に用いることが必要となる。反対に、大きさは特徴量として認識に用いる必要がない。

図 5.1B のジェスチャグループには、ジェスチャの大きさや向きはどの学習データも同じであるが、位置が異なるジェスチャが存在している。つまり、これらを識別するためには、位

置を特徴量として認識に用いることが必要となる。大きさや向きは特徴量として認識に用いる必要がない。

図 5.1C のジェスチャグループには、1 種類のジェスチャしか存在していない。つまり、識別のためにいずれの特徴量も認識に用いる必要がない。

このようにして、ジェスチャグループに存在する学習データの種類によって、認識に用いる特徴量を選ぶ、つまり、ある特徴量については認識のために特徴量として用いないことによって、その特徴量について不变にし、ロバスト性を維持する。これにより認識率の低下を防ぐことが可能となる。

以上を踏まえ我々は、「同一ジェスチャグループ内において、他の学習データと類似している特徴量は、認識のための特徴量として用いなければ、認識率の低下を抑えることができる」という仮説を立てた。

5.3 認識に用いる特徴量を選定した時の認識率と認識速度の実験

前節までにおいて述べてきたように、ジェスチャグループを作成し、ジェスチャグループ内に存在する学習データのみに対し、大きさ、向き、位置の類似度計算をすると認識速度の低下を抑えることができるという仮説と、同一ジェスチャグループ内において、他の学習データと類似している特徴量は、認識のための特徴量として用いなければ、認識率の低下を抑えることができるという仮説を検証するための実験を行った。ここではまず、実験を行うにあたって、大きさ、向き、位置、それぞれの特徴量と類似度の定義、ジェスチャグループの作成方法、ジェスチャグループにおける認識に用いる特徴量の選定方法をそれぞれ具体的に述べてから、実験の内容を述べる。

5.3.1 ジェスチャの特徴量と類似度の定義

ジェスチャの大きさ、向き、位置を特徴量として用いた時の、それぞれの特徴量と類似度の定義を示す。

大きさ

ジェスチャを構成するストロークの点を内包するように隣接した矩形の面積をジェスチャの大きさとして定義する(図??)。そして、2つのジェスチャの類似度 S_s を式 5.1 によって定義する。

$$S_s = \begin{cases} \frac{S'}{S}(S > S') \\ \frac{S}{S'}(S' > S') \end{cases} \quad (5.1)$$

ここで、 S は入力データの矩形の面積、 S' は学習データの矩形の面積である。

$$S = w * h$$

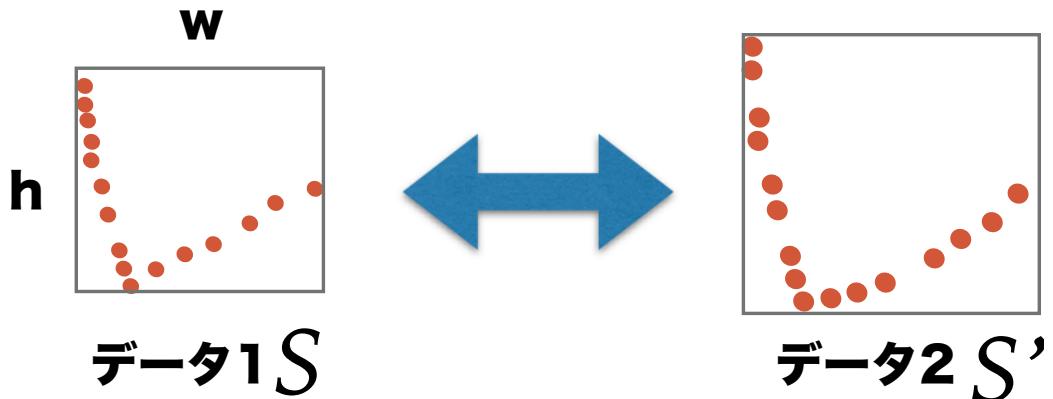


図 5.4: ジェスチャグループの例

向き

ジェスチャを構成するストロークの始めの点の座標，全サンプル点の中心座標，その中心座標から右横に延長した線(0度)によって形成させる角度をジェスチャの向きとして定義する(図??)。そして，2つのジェスチャの類似度 S_o を式 5.2 によって定義する。

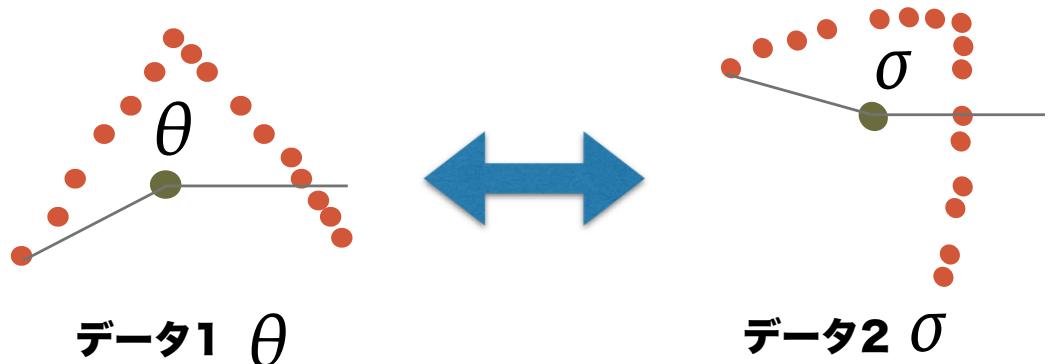


図 5.5: ジェスチャグループの例

$$S_o = 1 - \frac{|\theta - \sigma|}{\pi} \quad (5.2)$$

ここで， θ は入力データの向き， σ は学習データの向きである。

位置

ジェスチャを構成するストロークのすべての点の中心座標をジェスチャの位置として定義する(図??)。そして、2つのジェスチャの類似度 Sp を式 5.3 によって定義する。

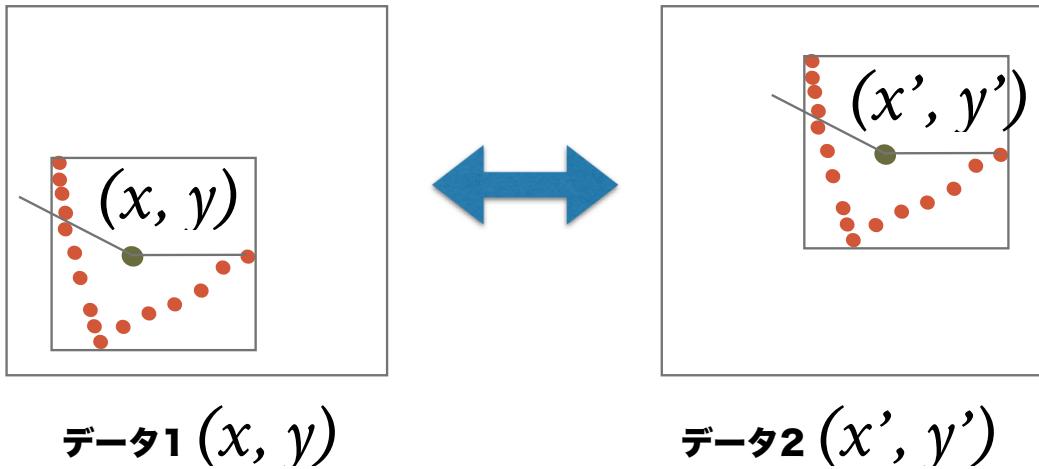


図 5.6: ジェスチャグループの例

$$Sp = 1 - \frac{\sqrt{(x - x')^2 + (y - y')^2}}{\sqrt{Width^2 * Height^2}} \quad (5.3)$$

ここで、 (x, y) は入力データの位置、 (x', y') は学習データの位置であり、Width は入力領域全体の横、Height は縦の長さである。

5.3.2 ジェスチャグループの作成方法

ジェスチャの形状と書き順が同じ学習データが集まったグループである、ジェスチャグループを作成するための手順を以下に示す。まず、学習データが新たに追加される場面を考える。

A. 同じ名前の学習データが存在する場合(図??)

新たに追加された学習データは同じ名前の学習データと一緒に保管される。この時、その名前を持つ学習データの、大きさ、向き、位置の特徴量は、それぞれの特徴量の算術平均によって表される(図??)。

ここで、図??において、 S_{new} , θ_{new} , P_{new} は、同じ名前の学習データにおいて、大きさ、向き、位置の特徴量を示している。

B. 同じ名前の学習データが存在しない場合

\$1 アルゴリズムを用いることによりジェスチャの形状と書き順を判断し、

$$S_{new} = \frac{S + S'}{2}$$

$$\theta_{new} = \frac{\theta + \theta'}{2}$$

$$P_{new} = \left(\frac{x + x'}{2}, \frac{y + y'}{2} \right)$$

図 5.7: 同じ名前の学習データが追加された時に，それぞれの大きさ，向き，位置の特徴量を (S, θ, P) ， (S', θ', P') とした時の，そのジェスチャの新たな特徴量の求め方

- (a) 同じ形状と書き順のジェスチャグループが存在しない場合 (図??)
新たに追加された学習データは新しい，ジェスチャグループとして保管される．
- (b) 同じ形状と書き順のジェスチャグループが存在する場合
そのジェスチャグループに保管される．

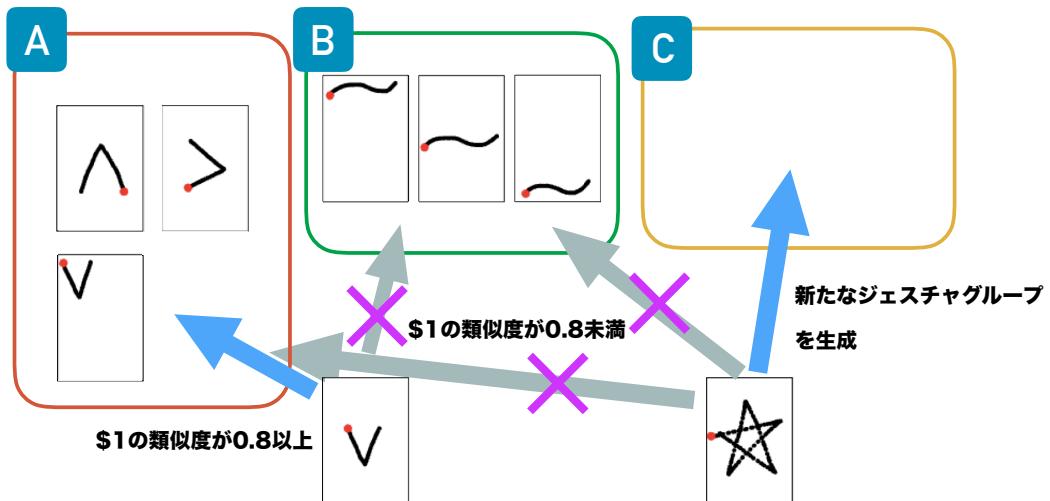


図 5.8: ジェスチャグループの例

ここで，新たに追加された学習データが，既存のジェスチャグループに対し形状と書き順が同じかどうかを判別する方法は，既存のジェスチャグループ内の学習データ 1 つ 1 つと比較することによって行う．今回は\$1 による類似度が 0.8 を超えたときに，形状と書き順が同じであると判断した．これは\$1 アルゴリズムにて，類似度の N-best List の 1 番目と 2 番目の

差が0.2以上であることを利用している。ここで、類似度のN-best Listとは、N個の学習データそれぞれに対するテストデータとの類似度を降順に並べたもの、1番目と2番目の差が大きいほど識別性能が高いことを示す。もし1つでも形状と書き順が同じジェスチャが存在すれば、そのジェスチャが存在するジェスチャグループに保管される。なお、保管候補のジェスチャグループが複数存在する場合は、最も類似度が高かったジェスチャが存在するジェスチャグループに保管される。

5.3.3 ジェスチャグループにおける認識に用いる特徴量の選定方法

我々は、同一ジェスチャグループ内において、他の学習データと類似している特徴量は、認識のための特徴量として用いなければ、認識率の低下を防ぐことができるという仮説を立てた。ここで、同一ジェスチャグループ内における他の学習データとの類似度を“ジェスチャグループの学習データ間の類似度”とし、その定義を以下のように定める。

まず、同一ジェスチャグループ内において、学習データを2つずつ抽出し、それぞれのペアについて学習データどうしの大きさ、向き、位置の類似度を式5.1～式5.3を用いて求める。そして、各ペアにおける類似度を比較した時に、それぞれの特徴量について、最小となった類似度を、その形状グループの学習データ間のそれぞれの特徴量についての類似度とする。

例えば、図5.2左に示すジェスチャグループの場合、同一ジェスチャグループ内に4つの学習データが存在する。そのため、2つずつ抽出した場合の学習データの組み合わせは $4C_2=6$ 通り存在する(図5.2右)。それぞれのペアについて、学習データどうしの大きさ、向き、位置の類似度はそれぞれのペアにおいて図5.2右示されている。この時それぞれの類似度の最小値は、大きさは0.8、向きは0.1、位置は0.4となるため、このジェスチャグループの学習データ間の類似度は(大きさ、向き、位置) = (0.8, 0.1, 0.4)となる(図5.2左)。ここで、類似度の最小値を用いる理由は、類似度が小さいということは、その特徴量に関して類似していないことを示し、類似していない学習データの組み合わせが1つでも存在すれば、その特徴量について識別するために、認識のための特徴量として用いる必要があると考えたからである。

以上のようにして得られたジェスチャグループの学習データ間の類似度を用いて、そのジェスチャグループが認識に用いる特徴量を選定する。本実験においては、類似度が0.2を下回った特徴量を、認識に用いる特徴量として選定した。

5.3.4 ジェスチャの認識方法

入力データの認識方法の手順は2つに分かれている。

1. \$1アルゴリズムを用いることにより、どのジェスチャグループに属するか判別する。この時、学習データを追加する時と同様、ジェスチャグループ内のすべての学習データに対し類似度を求め、0.8を超えた場合あるいは、0.8を超えるジェスチャが複数存在する場合は、最も類似度が高いジェスチャが存在するジェスチャグループに属すると判別する。

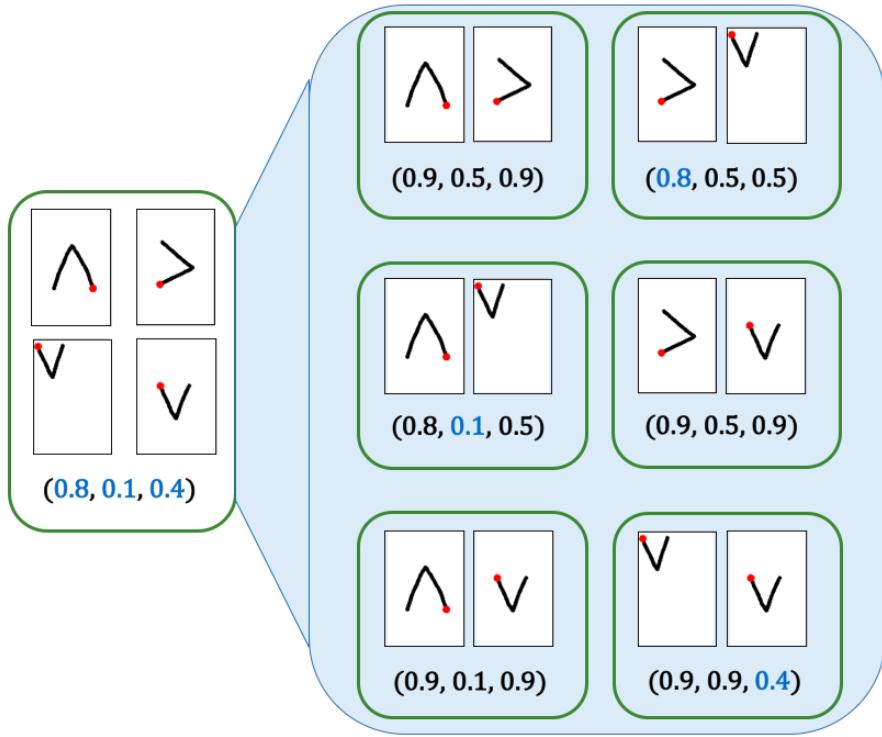


図 5.9: ジェスチャグループの学習データ間の類似度 (Sts , Sto , Stp) を求める方法 , この場合 ,
(0.8, 0.1, 0.4) となる .

2. 1. によって判別されたジェスチャグループ内において , どのジェスチャと最も類似しているかを判別する .
2. において , 類似度 S_{final} は以下の式によって求められる .

$$S_{final} = \frac{S_{cs} \times s + S_{co} \times o + S_{cp} \times p}{n} \quad (5.4)$$

$$s = \begin{cases} 1(S_{ts} < 0.2) \\ 0(else) \end{cases} \quad (5.5)$$

$$o = \begin{cases} 1(S_{to} < 0.2) \\ 0(else) \end{cases} \quad (5.6)$$

$$p = \begin{cases} 1(S_{tp} < 0.2) \\ 0(else) \end{cases} \quad (5.7)$$

ここで， S_{cs} は入力データと学習データの大きさの類似度， S_{co} は入力データと学習データの向きの類似度， S_{cp} は入力データと学習データの位置の類似度を示しており， S_{ts} は学習データ間の大きさの類似度， S_{to} は学習データ間の向きの類似度， S_{tp} は学習データ間の位置の類似度を示している．また，n は 1 となる s ， o ， p の数であり， $0 \leq n \leq 3$ を満たす正の整数である．なお， $n = 0$ の時， S_{final} は計算せず，手順 1.において，最も類似度が高いジェスチャを一致するジェスチャとして判別する．

5.3.5 被験者

被験者は，ユーザ調査において協力してもらった 6 名である（男性 6 名，21～27 歳（平均 23.8 歳），全員右利き）．

5.3.6 実験機器

実験には，入力端末として iPhone5 を用い，実験における入力領域は $1.94'' \times 3.18''$ であり，解像度は 640×1036 である（図 5.3 における緑色の部分）．

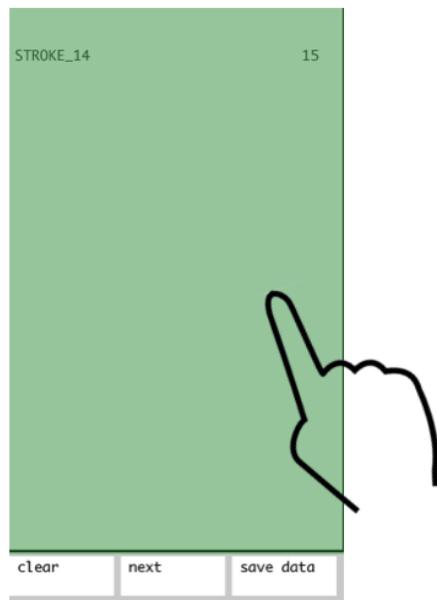


図 5.10: 実験に用いたスマートフォンのスクリーンショット．緑色のエリアがジェスチャが入力されるエリアである．

5.3.7 実験手順

ジェスチャの取得

我々はまず、被験者に実験の目的を説明した。その後、ユーザ調査において記入してもらつた紙を見ながら、自身が入力したそれぞれのジェスチャを入力するよう指示した。この、それぞれの被験者ごとのジェスチャを“ジェスチャセット”とする。ジェスチャは図5.3における緑色の領域部分にジェスチャを入力するよう指示した。その際、そのジェスチャを入力するときの姿勢が紙に書かれているため、それに従い入力するよう指示した。それぞれのジェスチャセットにおいて、ジェスチャには、ジェスチャ番号がSTROKE_1のようにして割り振られており、図5.3に示すように、画面左上に入力すべきジェスチャが表示される。タスクの1試行は被験者が1つのジェスチャを入力するまでである。被験者はランダムに選択されたそれぞれのジェスチャを1回ずつ入力し、自身のジェスチャセットに含まれるジェスチャ全てを入力し、これを1セッションとした。これを10セッション行った。被験者によって入力すべきジェスチャの数は異なるが、いずれの被験者においても20以上のジェスチャを入力する(20~24個のジェスチャ、平均22個)。したがって、被験者は平均して計220試行(22ジェスチャ×10セッション)行った。ジェスチャが思うように入力できなかつた場合には、何度も書き直し可能とした。

認識率と認識速度の測定

本実験において被験者は6人であるため、ジェスチャセットは6つ存在する。実験は各被験者が自身のジェスチャセットについてのみ行うためユーザ依存となる。それぞれのジェスチャは10個ずつあり、学習データをランダムにE個選ぶ。その際のジェスチャグループの決め方と、ジェスチャグループの学習データ間の類似度の計算方法は、5.3.2節及び5.3.3節に示したとおりである。学習データを追加し終わった後、入力データを残りの $10 - E$ 個のジェスチャからランダムに1個選び、認識率と認識速度を測定した(10分割交差検定)。これをそれぞれのジェスチャにつき100回行った。本実験は $E = 1 \sim 5$ とした。また、認識率はジェスチャセットにおける認識率の100回平均を学習データごとに測定し、認識速度は、ジェスチャ1つを認識し終わるまでに要した時間のジェスチャセットに含まれる全ジェスチャの平均値の100回平均であり、テストに用いるジェスチャをランダムに選ぶ過程は含まれていない。

N-best List の1番目と2番目の差と類似度の測定

認識率と認識速度に加え、類似度のN-best Listの1番目と2番目の差及び類似度の測定も行う。ここで、類似度のN-best Listとは、ある入力データが、全ての種類の学習データに対してどれくらい類似しているかを示すものであり、1番目と2番目の差が大きいほど識別性能が高いことを示す。また、本実験においては、正しく認識されたジェスチャの類似度の平均値の100回平均、正しく認識されたジェスチャのうち、類似度が最も低かったものの100回平均も測定した。

5.3.8 実験結果と考察

それぞれの被験者ごとの認識率，認識速度，N-best Lists の 1 番目と 2 番目の差，ジェスチャが正しく認識された時の類似度，ジェスチャが正しく認識された時の類似度の最小値を示す。

上記の結果について，正規化せずにリサンプリングした点のみによってジェスチャを比較する場合及び，ジェスチャグループごとに特徴量を認識用いるが選定しない手法，つまり，どのジェスチャグループに対しても大きさ，向き，位置の特徴量を類似度に用いる手法についても併せて示す。

認識率

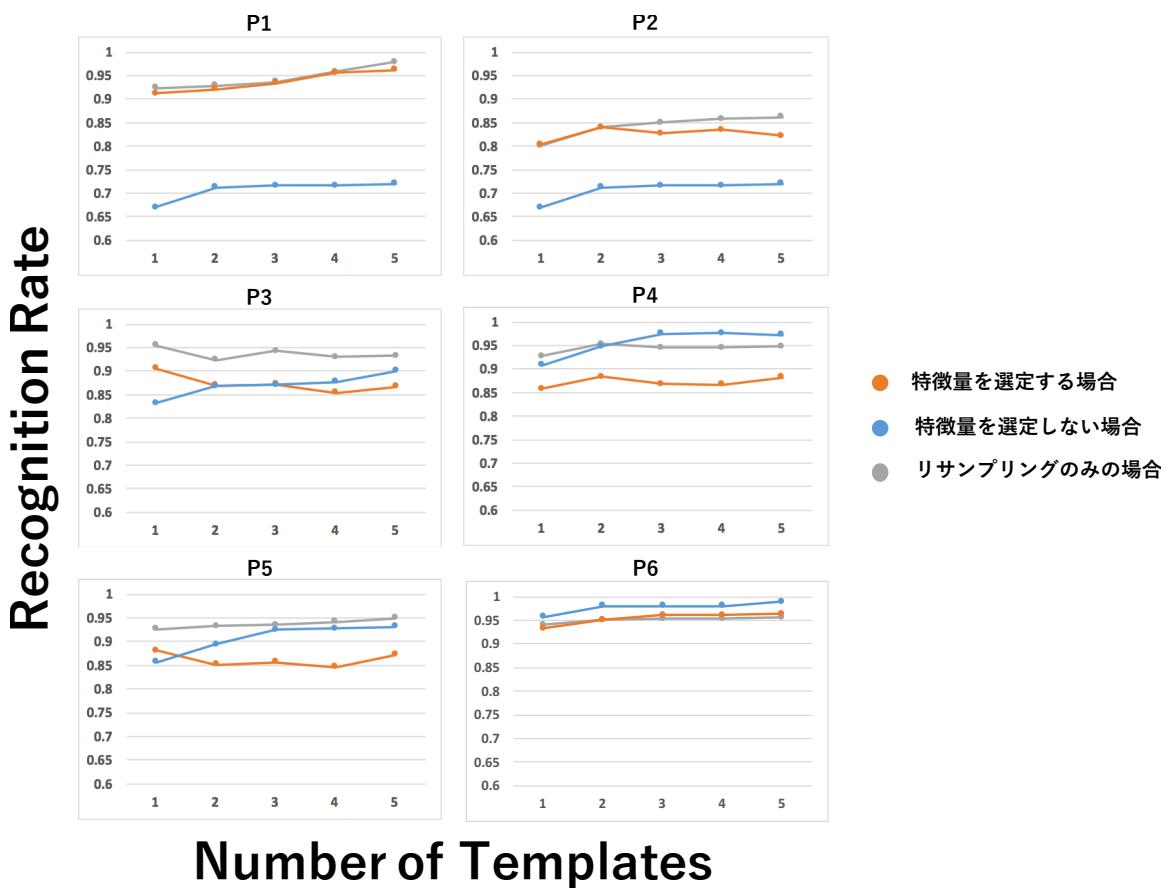


図 5.11: 各手法における，被験者ごとの認識率

特徴量を選定する手法及び特徴量を選定しない手法は，どの被験者のジェスチャセットにおいても認識率は高かった。リサンプリングのみの場合は，ジェスチャセットによっては認識率は低かった。また，リサンプリングのみの手法は，学習データの数に比例して認識率が

高くなつたが，特徴量を選定する手法及び特徴量を選定しない手法において，認識率は必ずしも学習データの数と認識率は比例しなかつた。

これは，リサンプリングのみの手法は，学習データが増えるほど，入力データに類似する学習データが存在する可能性が高くなるが，特徴量を選定する手法及び特徴量を選定しない手法は，同じジェスチャの学習データを追加するたびに，大きさ，向き，位置それぞれの特徴量を算術平均するため，必ずしも入力データに類似する学習データが存在する可能性が高くなるとは言えないからである。

認識速度

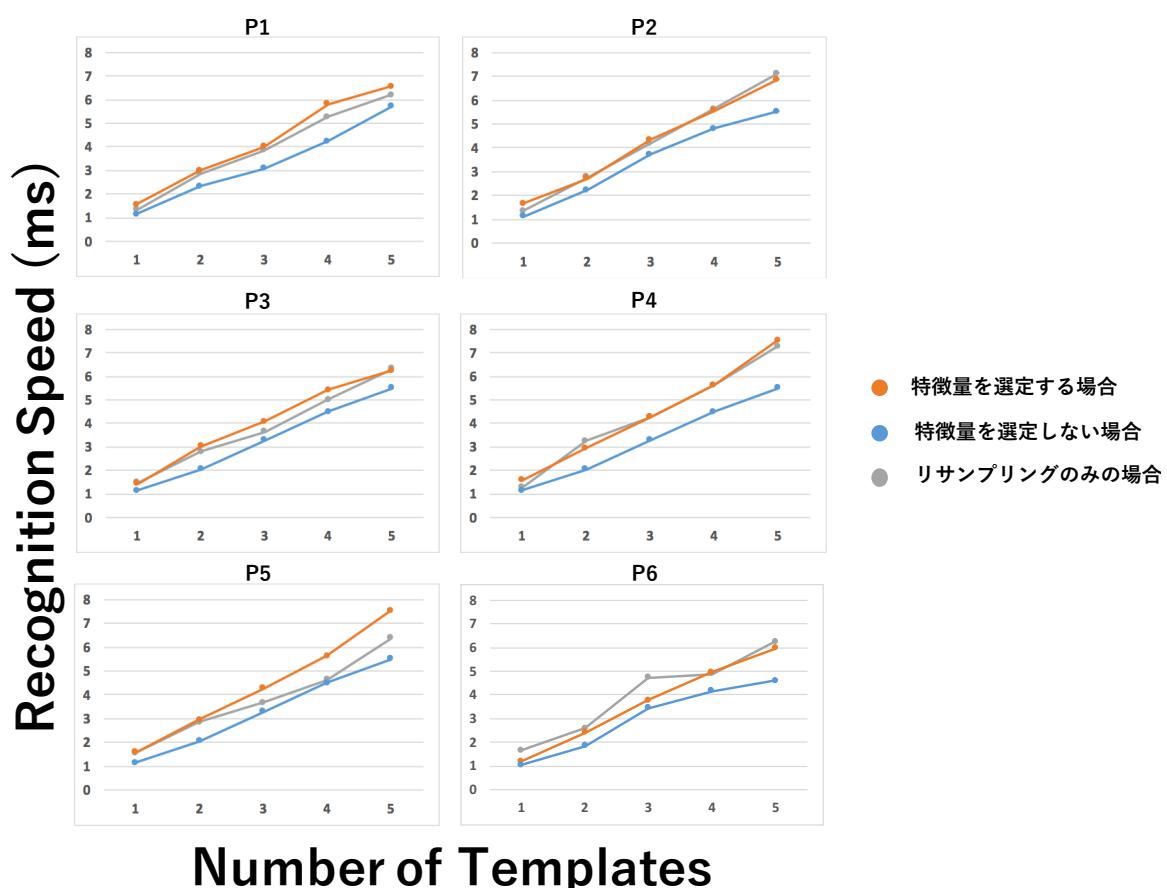


図 5.12: 各手法における，被験者ごとの認識速度

いずれの手法においても，認識速度はほとんど変わらなかつた。リサンプリングのみの手法は，正規化処理を行わないので，最も認識速度は速くなつた。特徴量を選定しない手法は，正規化したのち，特徴量を選定するための処理を行わないので次に認識速度が速くなつた。特徴量を選定する手法は正規化に加え，特徴量を選定する処理を行う必要があるため最も認識速度

が遅くなった。また、どの場合においても認識速度は学習データの数に比例して遅くなった。

N-best Lists の 1 番目と 2 番目の差

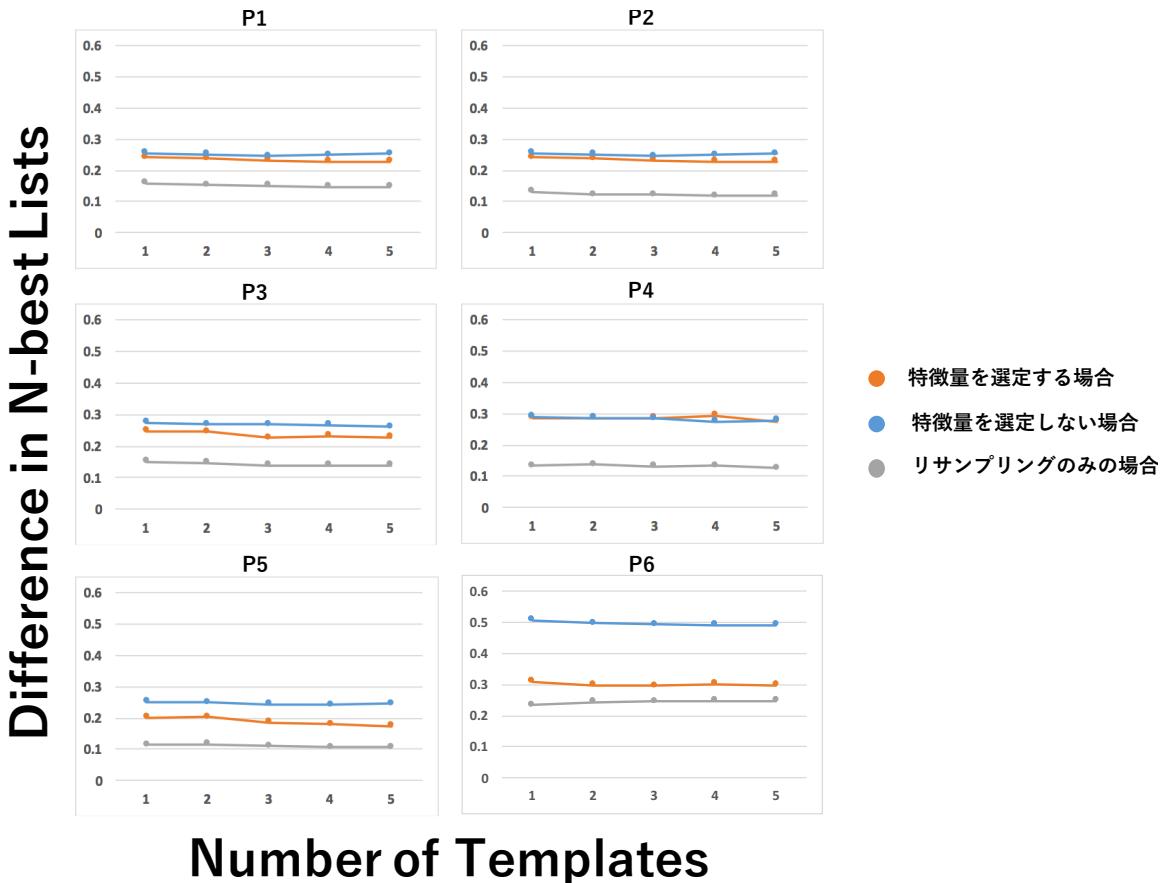


図 5.13: 各手法における、被験者ごとの N-best Lists の 1 番目と 2 番目の差

N-best Lists の 1 番目と 2 番目の差は、リサンプリングのみの手法、特微量を選定する手法、特微量を選定しない手法の順に大きくなかった。リサンプリングのみの場合においては、正規化しないため、類似するジェスチャとそれ以外のジェスチャにおいて、類似度の差が大きくなりやすくなったと言える。特微量を選定する手法及び特微量を選定しない手法は、正規化するため、類似度の差が大きくなりづらいが、特微量を選定する手法は、ジェスチャグループの学習データ間の類似度が小さい特微量のみ認識に用いているため、類似度の差が比較的大きくなかったと言える。それに対し、特微量を選定しない手法は識別に必要のない特微量も認識に用いるため、例えば、向きが異なるが位置がほぼ同じであるジェスチャを識別しようとする場合に、位置の類似度を加味してしまうため、類似度の差が小さくなつたと言える。

ジェスチャが正しく認識された時の類似度

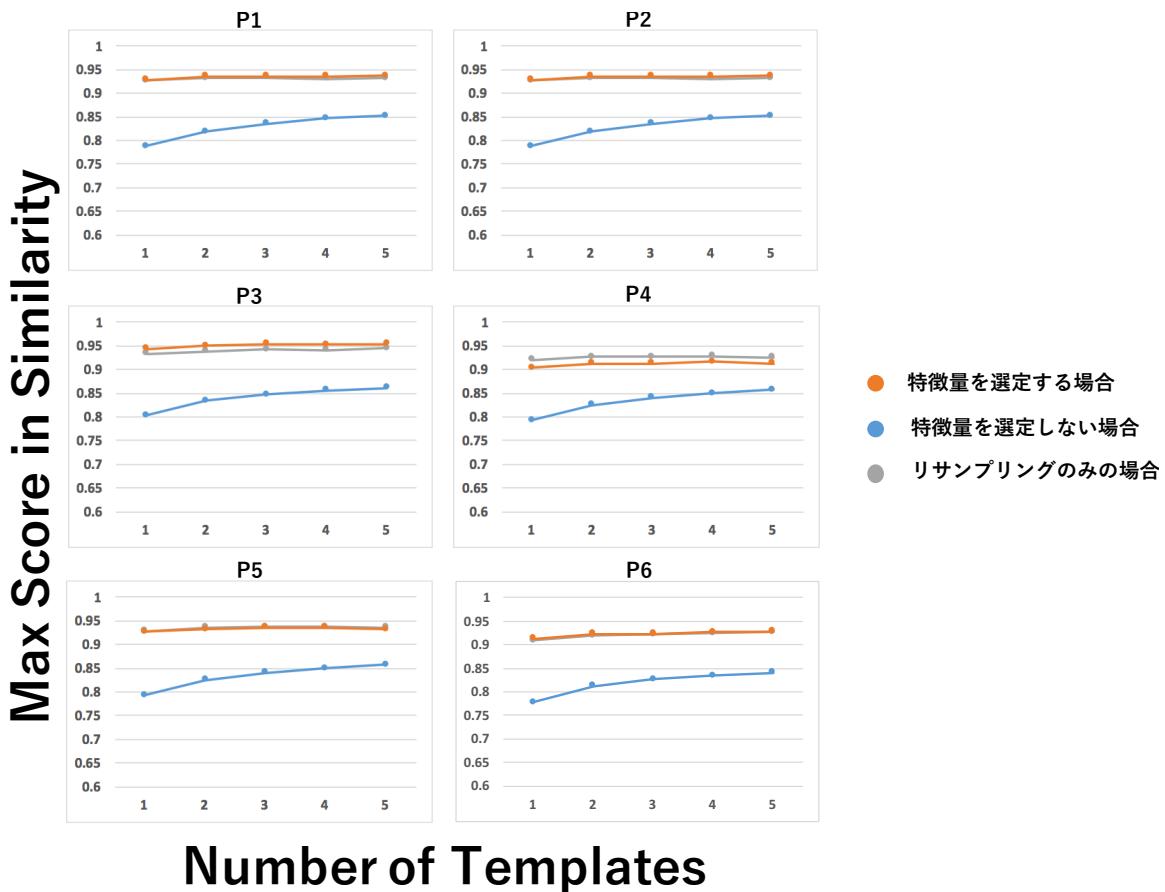
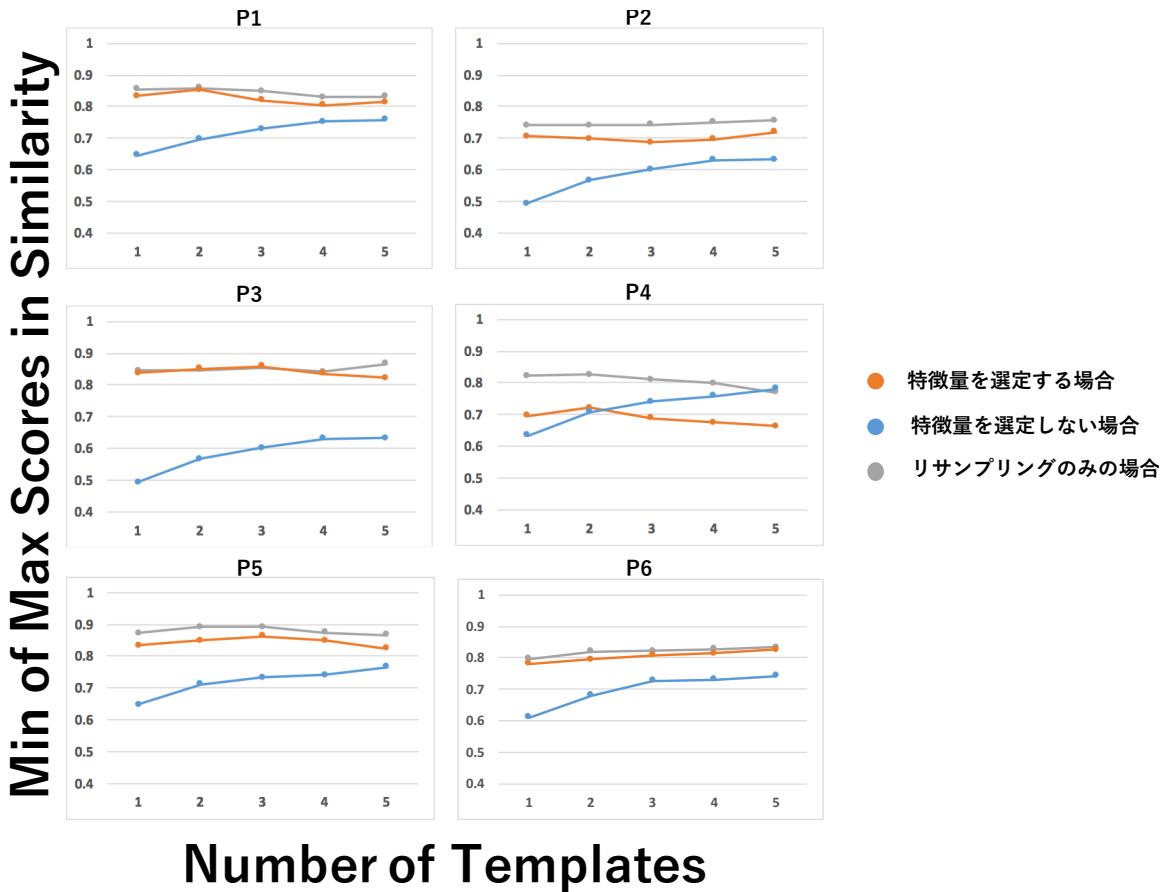


図 5.14: 各手法における、被験者ごとのジェスチャが正しく認識された時の類似度

特徴量を選定する手法及び特徴量を選定しない手法は、ジェスチャが一致した時の類似度の平均値は高くなり、ほぼ同じような結果となった。特徴量を選定しない手法は、特徴量を選定する場合と比べて類似度が小さくなると予想されたが、このような結果となった。これは、本実験がユーザ依存であったため、被験者の書くジェスチャの再現率が高かったことが原因であると言える。リサンプリングのみの手法は被験者の書くジェスチャの再現率が高くとも、正規化しないため類似度が低くなったと言える。

ジェスチャが正しく認識された時の類似度の最小値

特徴量を選定する手法及び特徴量を選定しない手法は、ジェスチャが一致した時の類似度の最小値は高くなかった。また、リサンプリングのみの手法は低くなかった。これらの要因は、ジェスチャが一致した時の類似度と同様であると考えられる。



Number of Templates

図 5.15: 各手法における、被験者ごとのジェスチャが正しく認識された時の類似度の最小値

5.3.9 議論

特徴量を選定する手法及び特徴量を選定しない手法はいずれも、認識率、認識速度のスコアは高く、特徴量を選定する手法は N-best Lists の 1 番目と 2 番目の差も大きくなった。以上を踏まえ、ジェスチャグループを作成し、ジェスチャグループ内に存在する学習データのみに対し、大きさ、向き、位置の類似度計算をすると認識速度の低下を抑えることができるという仮説及び、同一ジェスチャグループ内において、他の学習データと類似している特徴量は、認識のための特徴量として用いなければ、認識率の低下を抑えることができるという仮説はある程度正しいと言える。それらに加え、特徴量を選定することによって、識別性能が向上するということも言える。

しかしながら被験者によっては、認識率及び N-best Lists の 1 番目と 2 番目の差のスコアは低くなかった。この原因について考察する。

特徴量を選定する手法において、それぞれの特徴量は、認識に用いられるか用いられないかの二通りに分類され、閾値を設けることにより判別してきたが、ランダムに選ばれる学習

データによっては、同じジェスチャグループであったとしても、認識に用いられる特徴量が異なる場合があった（閾値によって二通りのいずれかに分類されてしまうため、閾値の設定も難しいといった問題もある）。

また、例えば図5.2において、向き、位置が認識に用いる特徴量として選ばる可能性が高いが、向きは位置に比べ、類似度が小さい組み合わせが存在するため、向きの方が位置よりも識別するための特徴量としてより考慮されるべきではないのかという疑問や、それぞれの特徴量による類似度を、同じ尺度において扱うことができるのかという疑問があった（例えば、大きさの類似度0.9と向きの類似度0.9は、同じくらい類似していると言えるのかなど）。

そこで、これらの問題点に対し、それぞれの特徴量を、認識に用いられるか用いられないかの二通りに分類するのではなく、特徴量に重み付けをすることによって解決しようと試みた。例えば、図5.2の場合、それぞれの特徴量に対する重みの和が1となる場合、これまでには特徴量を用いるか用いないかであったため、（大きさ、向き、位置）の重みが（0, 0.5, 0.5）であったところを、（0.1, 0.6, 0.3）といった具合にする。このように重みを導入することによって、認識に用いる特徴量をより高い尤度によって決定することを試みた。

そこで、我々は、認識率及びN-best Listsの1番目と2番目の差のスコアを向上させる「最適な重み」を求ることとした。

5.4 最適な重み付けのための実験

あるジェスチャグループが認識に用いるべき特徴量が、ジェスチャグループの学習データ間の類似度と関係していることは、これまで述べてきた通りである。そこで我々は、ジェスチャグループの学習データ間の類似度をもとに、認識率及びN-best Listsの1番目と2番目の差のスコアを向上させるためのそれぞれの特徴量に対する最適な重み付けの方法を実験的に求めることとした。

5.4.1 重み付けの手順

重み付けの手順を述べる。

学習データは、追加されるたびに、\$1アルゴリズムを用いてジェスチャグループとして保管される。その際のジェスチャグループの決め方と、ジェスチャグループの学習データ間の類似度の計算方法は、5.3.2節及び5.3.3節に示したとおりである。

全ての学習データを追加し終わった後、入力データを入れていく。まず\$1アルゴリズムによりどの形態のジェスチャであるかを判定する。これは5.3.4節において述べた方法と同じである。その後、ジェスチャグループにおける大きさ、向き、位置のそれぞれの特徴量に対し、最適な重みを求める。その方法を図5.9、図5.10に図示する。まず、入力データと学習データの類似度を求める。その後大きさ、向き、位置のそれぞれの特徴量に対する重みを、それぞれの和が1となるようにそれぞれ0.1ずつ変化させていく。そして、重みを先ほど求めた類似度に式5.8に則って乗算することによって、最終的な類似度を求める。ここで、式5.8におい

て， S_{cs} は入力データと学習データの大きさの類似度， S_{co} は入力データと学習データの向きの類似度， S_{cp} は入力データと学習データの位置の類似度を示しており， W_s は大きさの重み， W_o は向きの重み， W_p は位置の重みを示している。この時，ジェスチャが一致した時の類似度の平均値が 0.9 以上，N-best List の 1 番目と 2 番目の差が 0.2 以上の時のそれぞれの特徴量に対する重みを最適な重みとして記録する。これを各被験者から得られた各ジェスチャセットごとに行う。

このようにして，各ジェスチャセットから得られる，それぞれのジェスチャグループ内の学習データ間の類似度と，最適な重みをセットにして記録することによって，どのような特徴を持つジェスチャグループの場合に，どのような重み付けをすることが望ましいかを考察する。

$$S_{final} = S_{cs} \times W_s + S_{co} \times W_o + S_{cp} \times W_p \quad (5.8)$$

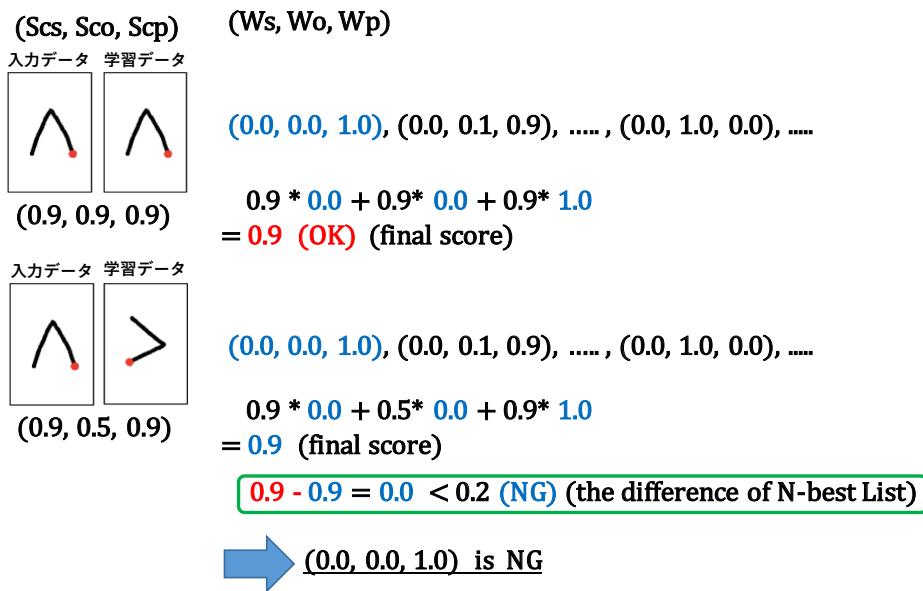


図 5.16: 最適な重みが決定されるまでの手順

5.4.2 実験結果

最適な重み付けのための実験結果を示す。

図 5.11 は，ジェスチャグループ内の学習データ間の類似度と大きさの最適な重みの関係，図 5.12，ジェスチャグループ内の学習データ間の類似度と向きの最適な重みの関係，図 5.13 は，ジェスチャグループ内の学習データ間の類似度と位置の最適な重みの関係を被験者ごとの結果示している。ここで， S_{ts} は学習データ間の大きさの類似度， S_{to} は学習データ間の向きの類似度， S_{tp} は学習データ間の位置の類似度を示している。

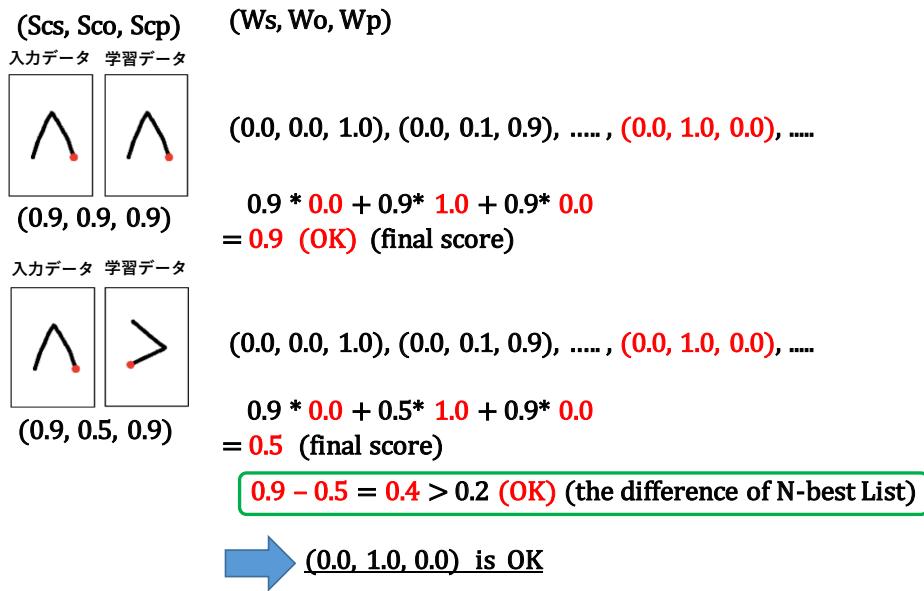


図 5.17: 最適「でない」な重みが決定されるまでの手順

ある類似度の時の最適な重みは複数存在するため、プロットされているデータは、その平均値と標準偏差である。

それぞれについて、我々はジェスチャグループ内の学習データ間の類似度と最適な重みを関係式によって表すこととした。関係式は、それぞれのグラフにおいて青色の線によって示されており、グラフの右上に式が示されている。

この関係式の求め方について述べる。

まず、実験から得られたジェスチャグループ内の学習データ間の類似度と最適な重みの関係の近似式を求める。この時、両対数グラフにおいて、およそ直線で表せられることがわかった。つまり、これらの関係は累乗近似曲線に近似できることがわかった。しかしながら、近似された累乗近似曲線に対し値が離れている元データが多く存在するため、近似式がどれほど元データを表すものになっているかを示す決定係数 R^2 は、どのグラフにおいても低くなった。そこで我々は、累乗近似曲線に近似できることを手掛かりに、近似式を以下のように定めた。

$$W = \frac{1}{S} \times \frac{1}{S} \quad (5.9)$$

ここで、W は重み、S は類似度を示す。

そして、 S を 5~30 まで 5 ずつ増やし、その時に求められる重みを元に認識率を測定したところ、図 5.11 ~ 図 5.13 において示されるような近似式において、認識率が高くなることがわかった。

5.5 最適な重みを用いたジェスチャの認識

以上を踏まえ、最適な重みを用いたジェスチャ認識が可能となる。

まず、学習データを追加する際に、ジェスチャグループ内の学習データ間の類似度を元に最適な重みを求める。この際、図 5.11～図 5.13において示される近似式を元に重みを求める。これは、以下の式に統合される。

$$W_s = \frac{1}{90} (5.5 + 3.5 \frac{S_{to} + S_{tp}}{S_{ts}}) \quad (5.10)$$

$$W_o = \frac{1}{30} (5.0 + 3.0 \frac{S_{ts} + S_{tp}}{S_{to}}) \quad (5.11)$$

$$W_p = \frac{1}{30} (5.0 + 3.0 \frac{S_{ts} + S_{to}}{S_{ip}}) \quad (5.12)$$

ここで、 S_{ts} は学習データ間の大きさの類似度、 S_{to} は学習データ間の向きの類似度、 S_{tp} は学習データ間の位置の類似度を示している。

次に、実際に入力データを認識させる手順を述べる。

1. \$1 アルゴリズムを用いることにより、どのジェスチャグループに属するか判別する。この時、学習データを追加する時と同様、ジェスチャグループ内のすべての学習データに對し類似度を求め、0.8 を超えた場合あるいは、0.8 を超えるジェスチャが複数存在する場合は、最も類似度が高いジェスチャが存在するジェスチャグループに属すると判別する。
2. 1. によって判別されたジェスチャグループ内において、どのジェスチャと最も類似しているかを判別する。

この際、ジェスチャグループ内において、入力データと学習データの類似度 (S_{cs} , S_{co} , S_{cp}) を求め、学習データを追加した際に式 5.10～式 5.12 によって求めた最適な重みを用い、式 5.8 によって最終的な類似度を求める。この類似度が最も高かった時のジェスチャが判別されるジェスチャとなる。

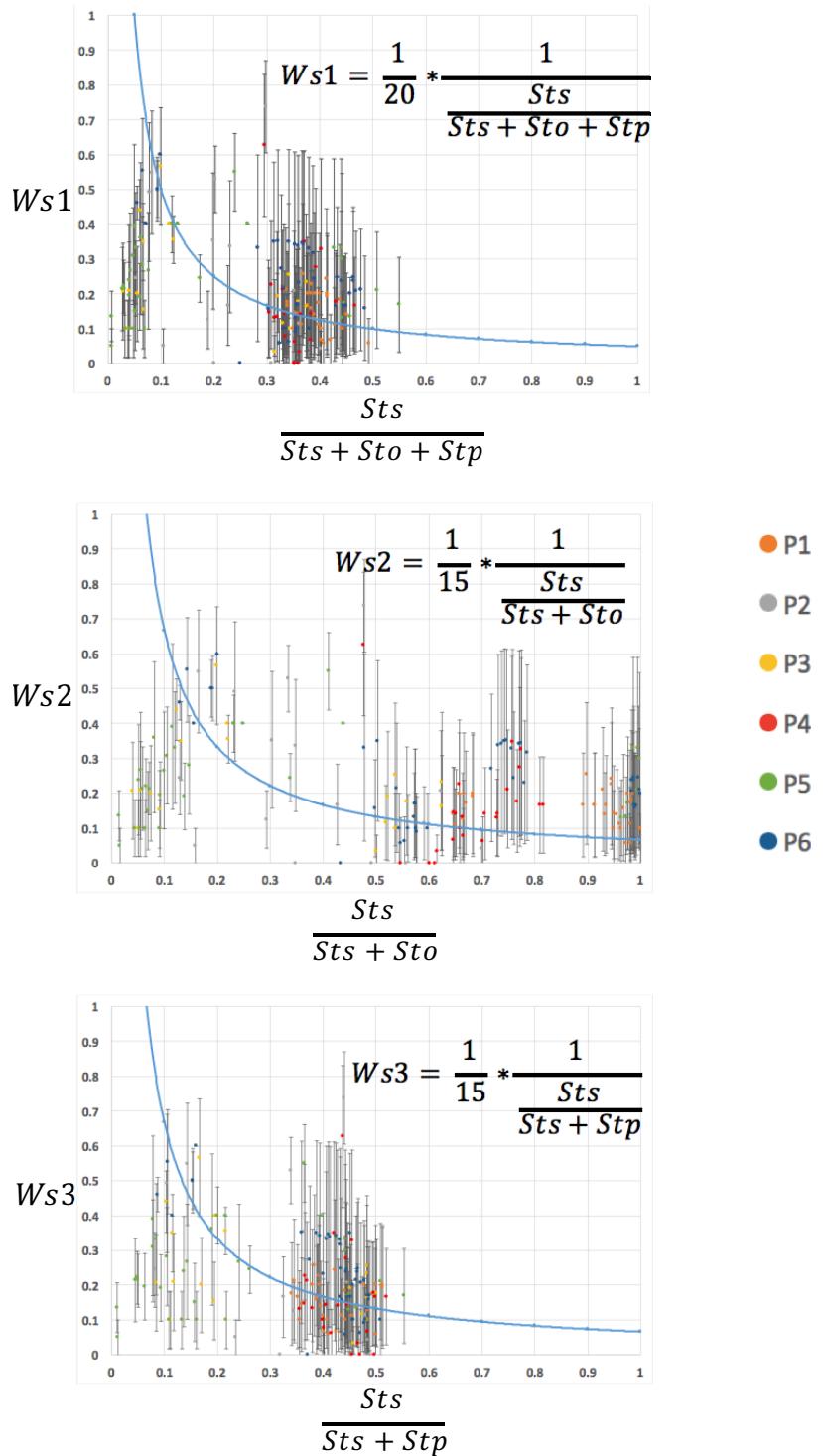


図 5.18: ジェスチャグループ内の学習データ間の類似度と大きさの最適な重みの関係

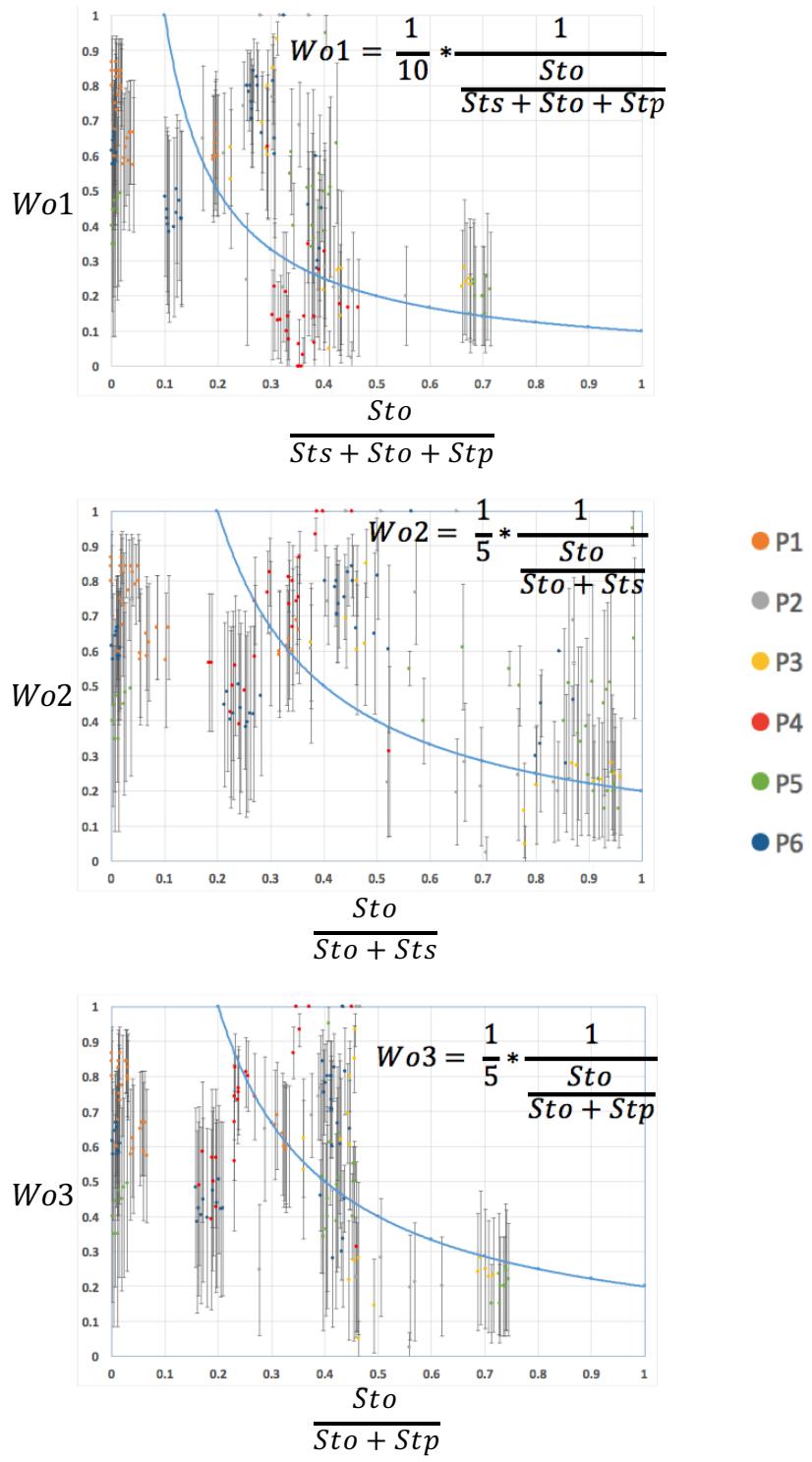


図 5.19: ジェスチャグループ内の学習データ間の類似度と向きの最適な重みの関係

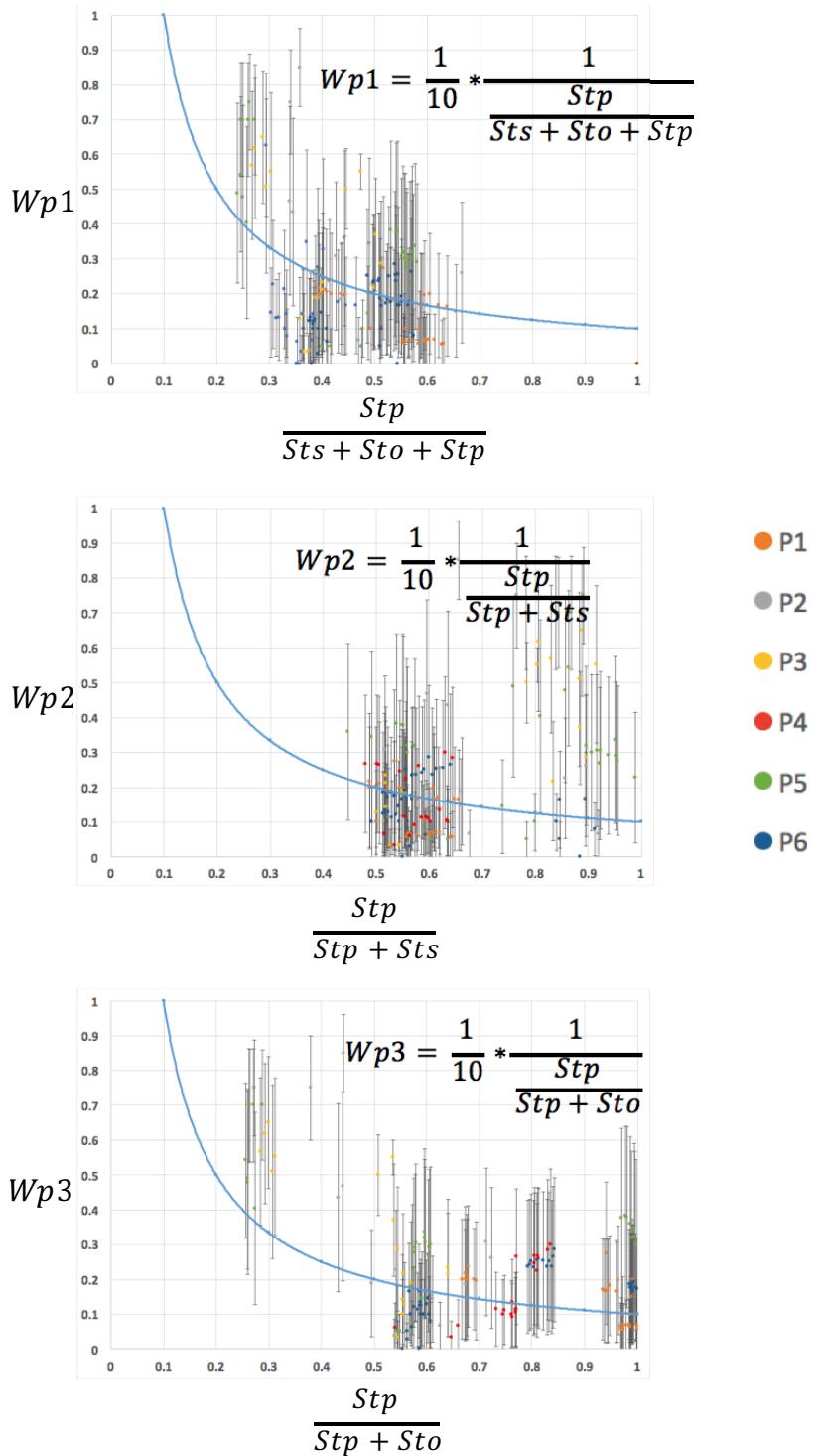


図 5.20: ジェスチャグループ内の学習データ間の類似度と位置の最適な重みの関係

第6章 評価実験

本章において、最適な重み付けをする場合の\$Vアルゴリズムの性能評価実験を述べる。

6.1 実験設計

\$Vの認識率、認識速度、類似度のN-best Listの1番目と2番目の差、ジェスチャが一致した時の類似度、ジェスチャが一致した時の類似度の最小値を測定することによってアルゴリズムの性能を評価した。また、認識率、認識速度に関しては、\$Vの拡張元である\$1、大きさ、向き、位置に関して識別可能なRubine、DTWと比較した。

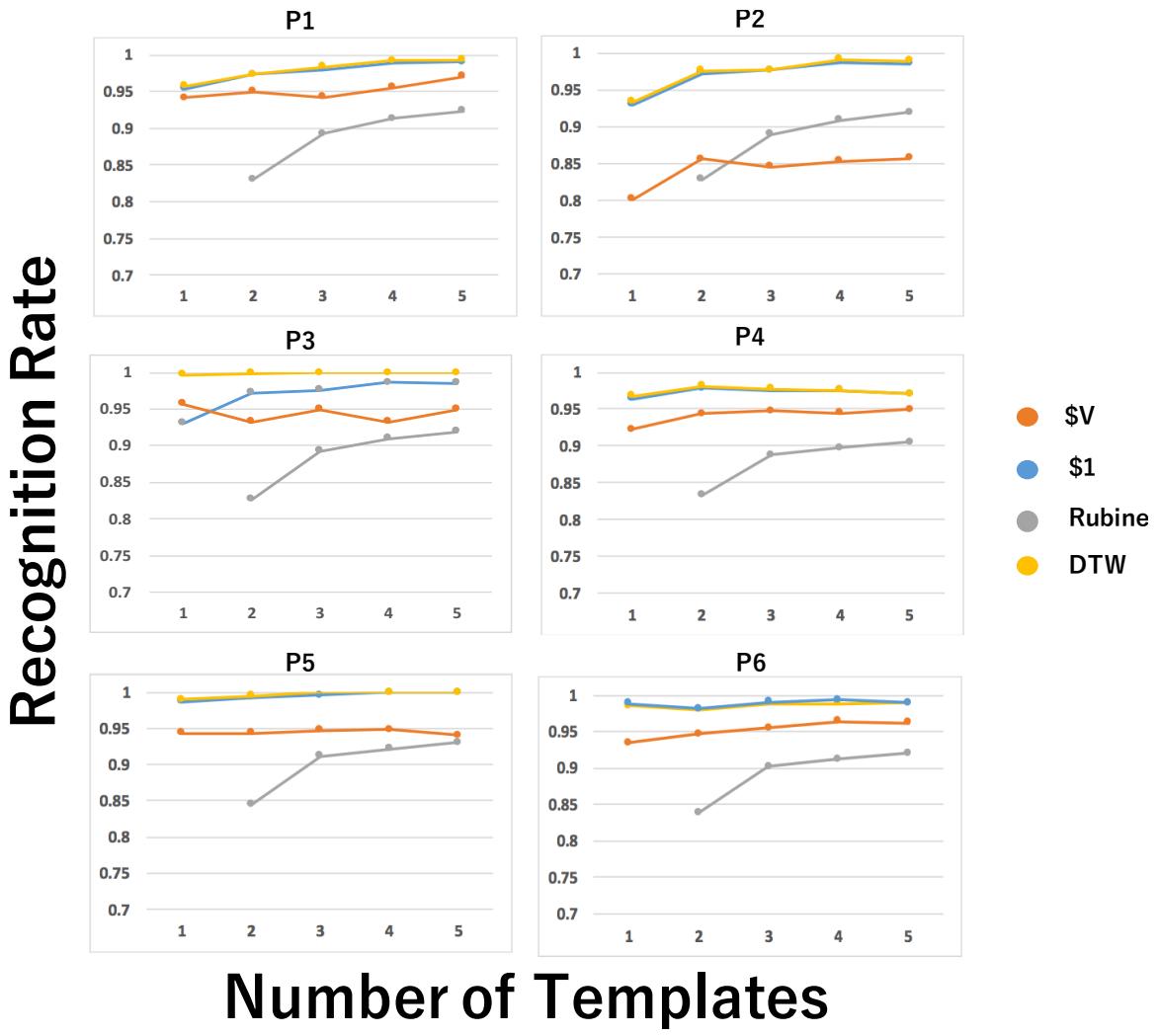
実験用いたジェスチャは、前章において用いたジェスチャを用いる。また、それぞれの測定方法については、前章において述べた通りである。

6.2 実験結果と考察

6.2.1 認識率

それぞれの被験者の平均は、\$Vは93% (SD=4.55)、\$1は98% (1.04)、DTWは98% (0.86)、Rubineは88% (2.14)であった。\$Vは\$1とDTWに対し、有意に低かった($p < 0.01$)が、被験者P1, P3, P4, P5, P6に関しては認識率の平均は95% (1.27)であり、\$1と比べて認識率の低下を最小限に抑えることに成功した。また、重み付けをしない場合と比べても、全被験者について認識率は向上した。また、Rubineに対してP1, P3, P4, P5, P6に関しては有意に高かった($p < 0.001$)。

P2の認識率が低かったのは、P2は別の名前のジェスチャで、似たような形状のジェスチャが複数存在したため、識別が困難になったことが要因であると考えられる。また、\$Vは学習データに比例して認識率が高くなるとは言えない。これは、同じジェスチャの学習データを追加するたびに、大きさ、向き、位置それぞれの特徴量を算術平均するため、必ずしも入力データに類似する学習データが存在する可能性が高くなるとは言えないからである。しかしながら、ほとんどの被験者において、少ない学習データにおいて高い認識率を示し、学習データが1つの場合における、全ジェスチャセットの認識率の平均は91.2% (SD=0.07)、学習データが2つの場合は92.2% (0.04)、学習データが3つの場合は92.7% (0.05)、学習データが4つの場合は93.3% (0.04)、学習データが5つの場合は93.2% (0.05)となった。



Number of Templates

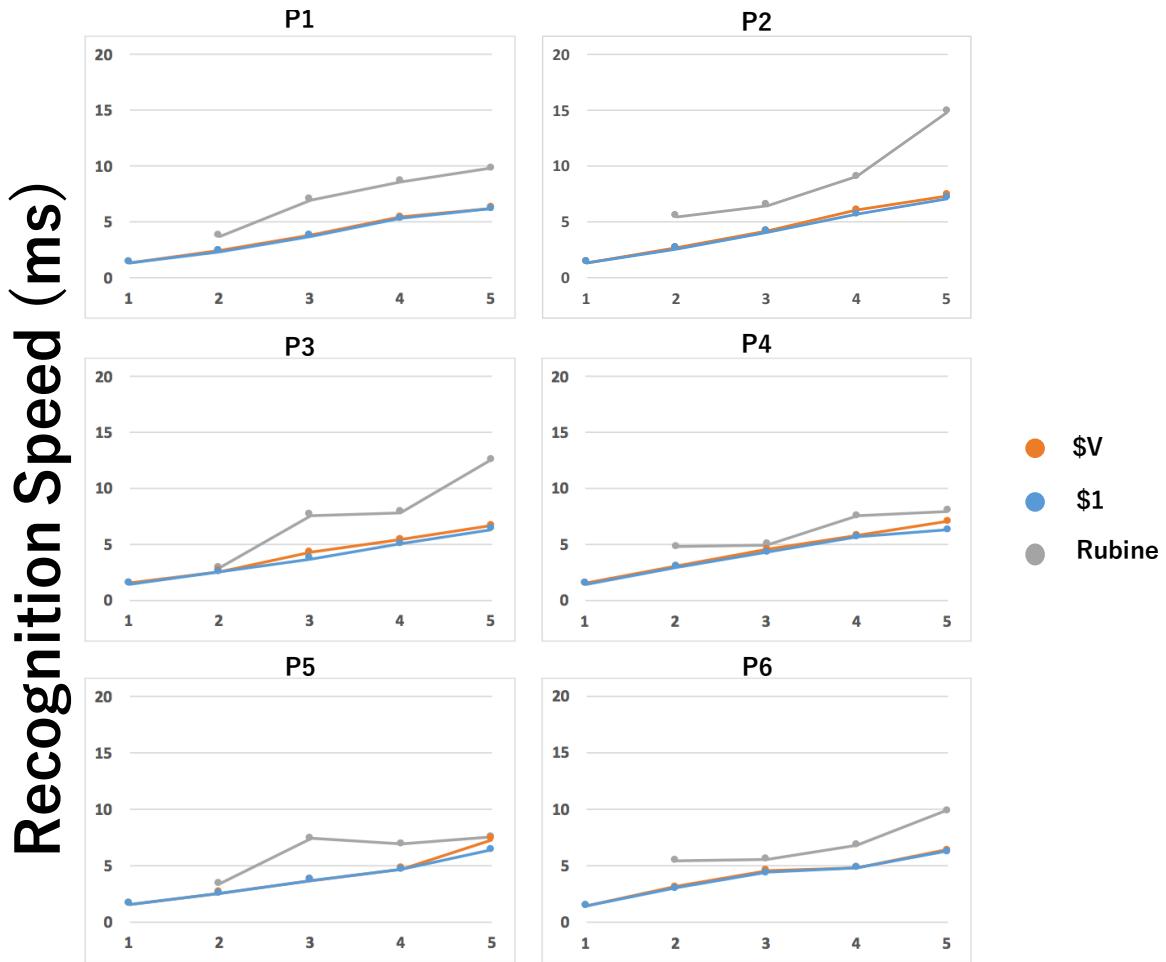
図 6.1: The screen shot on the smartphone. The green area is the input area

6.2.2 認識速度

6.2.3 \$V, \$1 , Rubine の認識速度

6.2.4 DTW の認識速度

\$V の認識速度は、全被験者の平均が、学習データの数が 1 つの場合 2.6ms (SD=0.2)、学習データの数が 2 つの場合 3.5ms (0.3)、学習データの数が 3 つの場合 4.1ms (0.3)、学習データの数が 4 つの場合 4.9ms (0.3)、学習データの数が 5 つの場合 6.1ms (0.4) となり非常に速いと言える。また、\$1 と認識速度に有意差はなく ($p < 0.001$)、Rubine と比べると有意に速かった ($p < 0.005$)。これらより、\$1 と比べて認識速度の低下を抑えることに成功した。DTW の認識速度は図 6.4 に示すように非常に遅く、\$V の認識速度は DTW のおよそ 100 分の 1 であつ



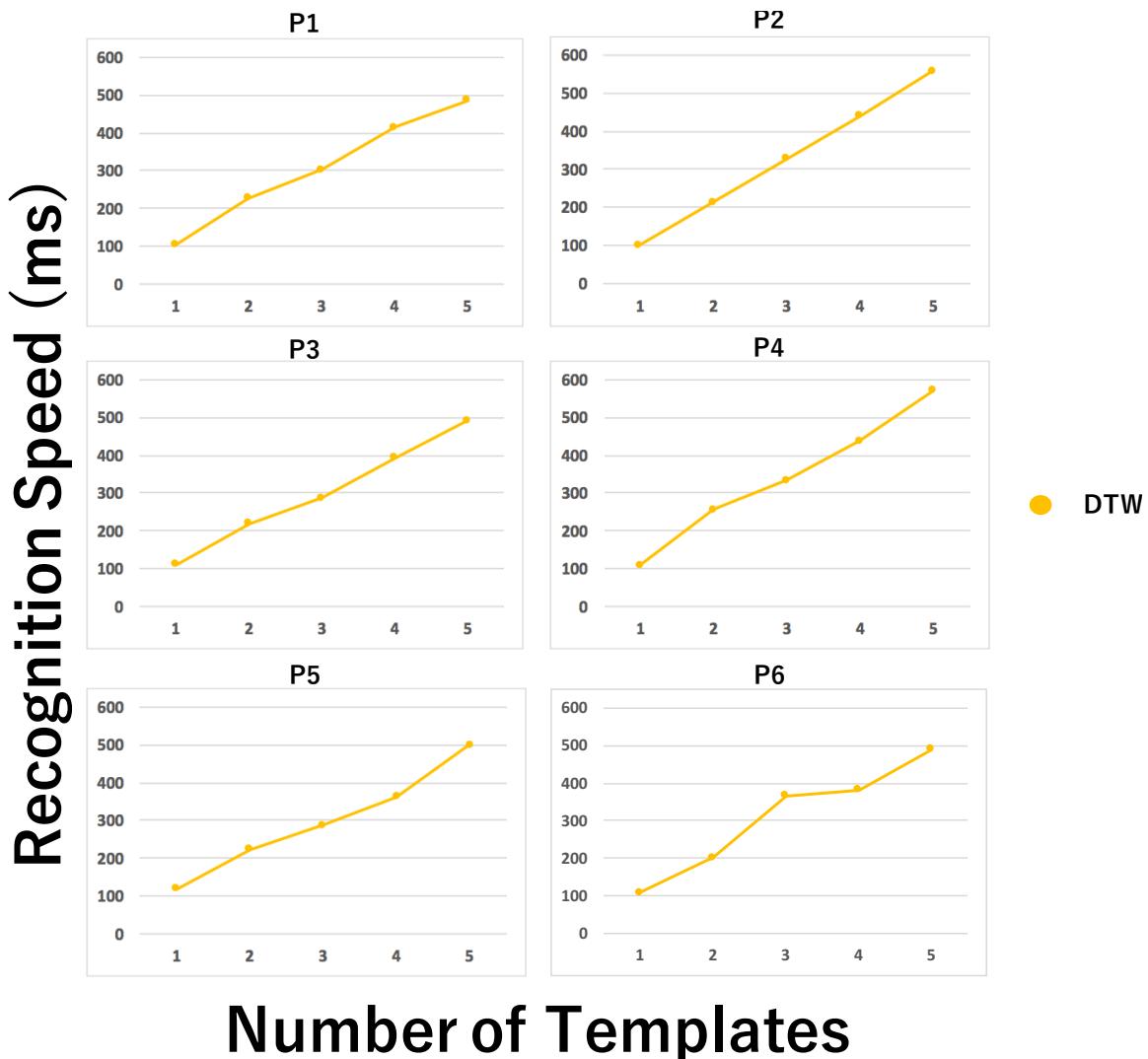
Number of Templates

図 6.2: The screen shot on the smartphone. The green area is the input area

た。また、学習データに比例して認識速度が遅くなることも分かった。

6.2.5 識別性能の結果

\$V の識別性能の結果を、N-best Lists の 1 番目と 2 番目の差及びジェスチャが一致した時の類似度及びジェスチャが一致した時の類似度の最小値を示すことによって述べる。



Number of Templates

図 6.3: The screen shot on the smartphone. The green area is the input area

N-best Lists の 1 番目と 2 番目の差

ジェスチャが一致した時の類似度

ジェスチャが一致した時の類似度の最小値

N-best Lists の 1 番目と 2 番目の差について，全ジェスチャセットの平均値はおよそ 0.24 (SD = 0.1) となった．また，ジェスチャが一致した時の類似度の平均値は 0.94 (SD=0.04) となり非常に高いと言えるが，ジェスチャが一致した時の類似度の最小値は被験者によってばらつきが生じ，平均値はおよそ 0.83 (0.14) であった．また，N-best Lists の 1 番目と 2 番目の差は，重み付けをしない場合と比べて有意に高く ($p < 0.01$)，識別性能が向上したと言える．しか

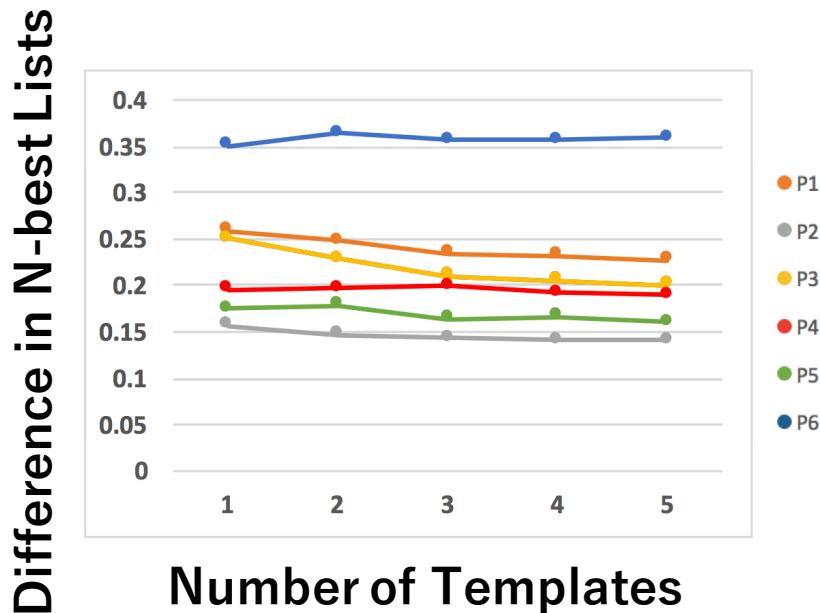


図 6.4: The screen shot on the smartphone. The green area is the input area

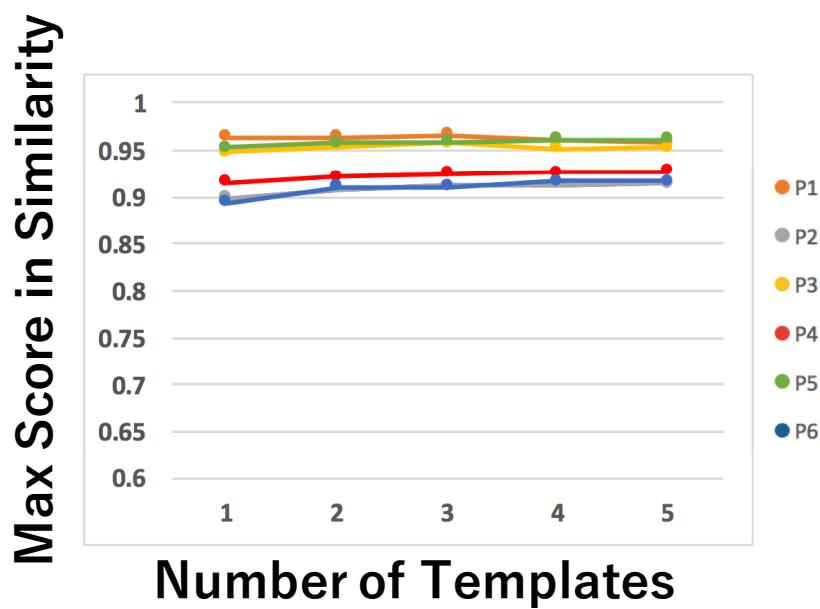


図 6.5: The screen shot on the smartphone. The green area is the input area

しながら、ジェスチャが一致した時の類似度の最小値は、重み付けをしない場合と比べて有意に低かった ($p < 0.01$)。これは、ジェスチャグループによっては、重み付けが最適でない場合があり、その場合類似度が低くなるからであると考えられる。よって、ジェスチャが一致した時の類似度の平均値が高いことから、重み付けは多くのジェスチャグループにおいて適

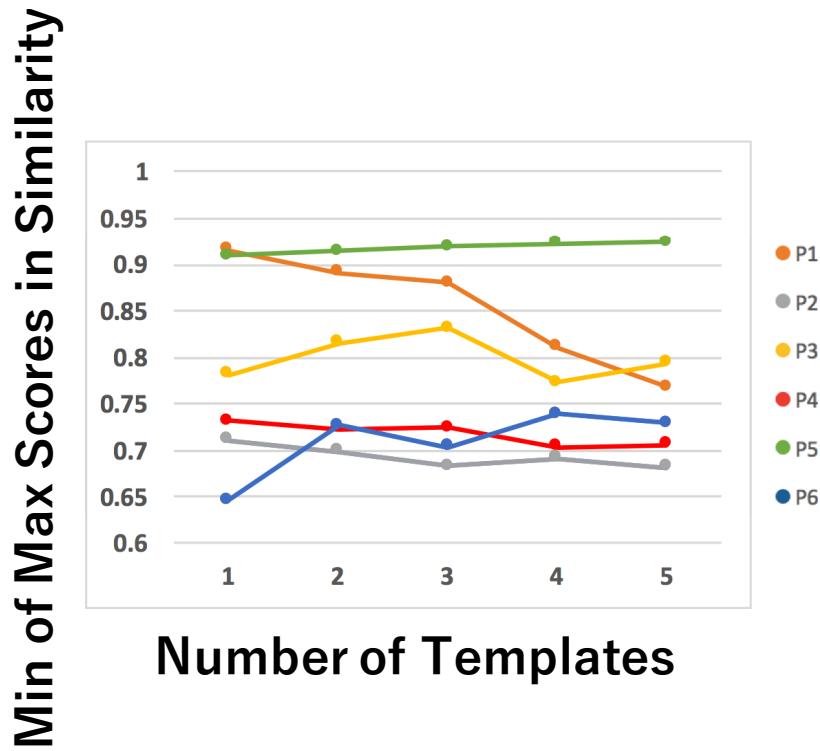


図 6.6: The screen shot on the smartphone. The green area is the input area

用可能であるが，適用することによって類似度が低下する場合もあることがわかった．また，ジェスチャが一致した時の類似度の最小値の平均はおよそ 0.65 以上であり，N-best Lists の 1 番目と 2 番目の差の平均値は 0.15 以上であることから，認識のための類似度の閾値を 0.5 とすることによって，ジェスチャが一致するか否かを判別することが可能となる．

以上を踏まえ，ジェスチャグループを作成し，ジェスチャグループ内に存在する学習データのみに対し，大きさ，向き，位置の類似度計算をすると認識速度の低下を防ぐことができるという仮説及び，同一ジェスチャグループ内において，他の学習データと類似している特徴量は，認識のための特徴量として用いなければ，認識率の低下を防ぐことができるという仮説は検証され，\$V\$ は．形状や書き順が同じ手書きジェスチャを大きさ，向き，位置に関して識別可能なアルゴリズムであると言える．

第7章 アプリケーション例

第8章 議論

第9章 結論

謝辞

参考文献

- [ABS04] Derek Anderson, Craig Bailey, and Marjorie Skubic. Hidden Markov Model Symbol Recognition for Sketch-based Interfaces. 2004.
- [AW10] Lisa Anthony and Jacob O. Wobbrock. A Lightweight Multistroke Recognizer for User Interface Prototypes. In *Proceedings of Graphics Interface 2010*, GI '10, pp. 245–252, Toronto, Ont., Canada, Canada, 2010. Canadian Information Processing Society.
- [AW12] Lisa Anthony and Jacob O. Wobbrock. \$N-protractor: A Fast and Accurate Multi-stroke Recognizer. In *Proceedings of Graphics Interface 2012*, GI '12, pp. 117–120, Toronto, Ont., Canada, Canada, 2012. Canadian Information Processing Society.
- [BNLH11] Andrew Bragdon, Eugene Nelson, Yang Li, and Ken Hinckley. Experimental Analysis of Touch-screen Gesture Designs in Mobile Environments. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI '11, pp. 403–412, New York, NY, USA, 2011. ACM.
- [CB05] Xiang Cao and Ravin Balakrishnan. Evaluation of an on-line adaptive gesture interface with command prediction. In *Proceedings of Graphics Interface 2005*, GI '05, pp. 187–194, School of Computer Science, University of Waterloo, Waterloo, Ontario, Canada, 2005. Canadian Human-Computer Communications Society.
- [Cho06] Mi Gyung Cho. A New Gesture Recognition Algorithm and Segmentation Method of Korean Scripts for Gesture-allowed Ink Editor. *Information Sciences*, Vol. 176, No. 9, pp. 1290–1303, May 2006.
- [HHN90] Tyson R. Henry, Scott E. Hudson, and Gary L. Newell. Integrating Gesture and Snapping into a User Interface Toolkit. In *Proceedings of the 3rd Annual ACM SIGGRAPH Symposium on User Interface Software and Technology*, UIST '90, pp. 112–122, New York, NY, USA, 1990. ACM.
- [HL00] Jason I. Hong and James A. Landay. SATIN: A Toolkit for Informal Ink-based Applications. In *Proceedings of the 13th Annual ACM Symposium on User Interface Software and Technology*, UIST '00, pp. 63–72, New York, NY, USA, 2000. ACM.

- [HS12] J. Herold and T. F. Stahovich. The 1&Cent; Recognizer: A Fast, Accurate, and Easy-to-implement Handwritten Gesture Recognition Technique. In *Proceedings of the International Symposium on Sketch-Based Interfaces and Modeling*, SBIM ’12, pp. 39–46, Aire-la-Ville, Switzerland, Switzerland, 2012. Eurographics Association.
- [HTH00] Pengyu Hong, Matthew Turk, and Thomas S. Huang. Constructing Finite State Machines for Fast Gesture Recognition. In *In Proc. 15th ICPR*, pp. 691–694, 2000.
- [KS05] Levent Burak Kara and Thomas F. Stahovich. An image-based, trainable symbol recognizer for hand-drawn sketches. *Comput. Graph.*, Vol. 29, No. 4, pp. 501–517, August 2005.
- [KZ04] Per-Ola Kristensson and Shumin Zhai. Shark2: A large vocabulary shorthand writing system for pen-based computers. In *Proceedings of the 17th Annual ACM Symposium on User Interface Software and Technology*, UIST ’04, pp. 43–52, New York, NY, USA, 2004. ACM.
- [Li10] Yang Li. Protractor: A Fast and Accurate Gesture Recognizer. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’10, pp. 2169–2172, New York, NY, USA, 2010. ACM.
- [LM93] James A. Landay and Brad A. Myers. Extending an Existing User Interface Toolkit to Support Gesture Recognition. In *INTERACT ’93 and CHI ’93 Conference Companion on Human Factors in Computing Systems*, CHI ’93, pp. 91–92, New York, NY, USA, 1993. ACM.
- [LNHL00] James Lin, Mark W. Newman, Jason I. Hong, and James A. Landay. DENIM: Finding a Tighter Fit Between Tools and Practice for Web Site Design. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’00, pp. 510–517, New York, NY, USA, 2000. ACM.
- [MMM⁺97] Brad A. Myers, Richard G. McDaniel, Robert C. Miller, Alan S. Ferrenny, Andrew Faulring, Bruce D. Kyle, Andrew Mickish, Alex Klimovitski, and Patrick Doane. The Amulet Environment: New Models for Effective User Interface Software Development. *IEEE Transactions on Software Engineering*, Vol. 23, No. 6, pp. 347–365, June 1997.
- [Pit91] James A. Pittman. Recognizing Handwritten Text. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’91, pp. 271–275, New York, NY, USA, 1991. ACM.

- [PS00] Réjean Plamondon and Sargur N. Srihari. On-line and off-line handwriting recognition: A comprehensive survey. *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 22, No. 1, pp. 63–84, January 2000.
- [PTVF92] William H. Press, Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. *Numerical Recipes in C (2Nd Edition): The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA, 1992.
- [Ret94] Marc Rettig. Prototyping for Tiny Fingers. *Communications of the ACM*, Vol. 37, No. 4, pp. 21–27, April 1994.
- [RSH11] J. Reaver, T. F. Stahovich, and J. Herold. How to Make a Quick\$: Using Hierarchical Clustering to Improve the Efficiency of the Dollar Recognizer. In *Proceedings of the Eighth Eurographics Symposium on Sketch-Based Interfaces and Modeling*, SBIM ’11, pp. 103–108, New York, NY, USA, 2011. ACM.
- [Rub91a] Dean Rubine. Specifying Gestures by Example. *SIGGRAPH Computer Graphics*, Vol. 25, No. 4, pp. 329–337, July 1991.
- [Rub91b] Dean Rubine. Specifying Gestures by Example. In *Proceedings of the 18th Annual Conference on Computer Graphics and Interactive Techniques*, SIGGRAPH ’91, pp. 329–337, New York, NY, USA, 1991. ACM.
- [SC07] Stan Salvador and Philip Chan. Toward Accurate Dynamic Time Warping in Linear Time and Space. *Intelligent Data Analysis*, Vol. 11, No. 5, pp. 561–580, October 2007.
- [SD05] Tevfik Metin Sezgin and Randall Davis. Hmm-based Efficient Sketch Recognition. In *Proceedings of the 10th International Conference on Intelligent User Interfaces*, IUI ’05, pp. 281–283, New York, NY, USA, 2005. ACM.
- [SMSJ⁺15] Shaikh Shawon Arefin Shimon, Sarah Morrison-Smith, Noah John, Ghazal Fahimi, and Jaime Ruiz. Exploring User-Defined Back-Of-Device Gestures for Mobile Devices. In *Proceedings of the 17th International Conference on Human-Computer Interaction with Mobile Devices and Services*, MobileHCI ’15, pp. 227–232, New York, NY, USA, 2015. ACM.
- [Tap82] C. C. Tappert. Cursive Script Recognition by Elastic Matching. *IBM Journal of Research and Development*, Vol. 26, No. 6, pp. 765–771, November 1982.
- [TL15] Eugene M. Taranta, II and Joseph J. LaViola, Jr. Penny Pincher: A Blazing Fast, Highly Accurate \$-family Recognizer. In *Proceedings of the 41st Graphics Interface Conference*, GI ’15, pp. 195–202, Toronto, Ont., Canada, Canada, 2015. Canadian Information Processing Society.

- [TSW90] C. C. Tappert, C. Y. Suen, and T. Wakahara. The state of the art in online handwriting recognition. *IEEE Trans. Pattern Anal. Mach. Intell.*, Vol. 12, No. 8, pp. 787–808, August 1990.
- [Vat12] Radu-Daniel Vatavu. User-defined Gestures for Free-hand TV Control. In *Proceedings of the 10th European Conference on Interactive tv and video*, EuroiTV ’12, pp. 45–48, New York, NY, USA, 2012. ACM.
- [VAW12] Radu-Daniel Vatavu, Lisa Anthony, and Jacob O. Wobbrock. Gestures As Point Clouds: A \$P Recognizer for User Interface Prototypes. In *Proceedings of the 14th ACM International Conference on Multimodal Interaction*, ICMI ’12, pp. 273–280, New York, NY, USA, 2012. ACM.
- [WMW09] Jacob O. Wobbrock, Meredith Ringel Morris, and Andrew D. Wilson. User-defined Gestures for Surface Computing. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’09, pp. 1083–1092, New York, NY, USA, 2009. ACM.
- [WS03] Andrew Wilson and Steven Shafer. Xwand: Ui for intelligent spaces. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’03, pp. 545–552, New York, NY, USA, 2003. ACM.
- [WWL07] Jacob O. Wobbrock, Andrew D. Wilson, and Yang Li. Gestures Without Libraries, Toolkits or Training: A \$1 Recognizer for User Interface Prototypes. In *Proceedings of the 20th Annual ACM Symposium on User Interface Software and Technology*, UIST ’07, pp. 159–168, New York, NY, USA, 2007. ACM.
- [ZK03] Shumin Zhai and Per-Ola Kristensson. Shorthand writing on stylus keyboard. In *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems*, CHI ’03, pp. 97–104, New York, NY, USA, 2003. ACM.