筑波大学 情報学群 情報メディア創成学類

卒業研究論文

柔らかいタッチパネル面に生じるしわを用いた入力認識

坂本 侑一郎

指導教員　志築 文太郎 三末 和男 田中 二郎

2011年2月

概要

従来のタッチパネルにおけるインタラクションは、指の接触した領域の座標のみを用いている単純なものが主である。このような単純な情報の入力以外の多様な入力が可能になれば、タッチパネルにおけるインタラクションの幅も広がると考えられる。

先行研究において、より多様な入力を実現するための、入力面の柔らかいタッチパネルとそれを用いた多様な入力方法が提案された。本研究では、このタッチパネル面に生じるしわを用いて認識する手法を実現した。加えて、本手法での入力の認識率と分解能を調査する実験を行った。実験の結果、入力の認識率の結果は 70%〜95% となった。分解能を調査する実験の結果では、入力値によって入力の成功率と達成時間に違いが生じた。入力認識率の実験結果から、入力方法の個人差を更に調査することが必要であると分かった。分解能を調査する実験の結果から、入力値を制御しやすい範囲が存在することが分かった。

目 次

図 目 次

第1章　序論

1.1　背景

Microsoft Surface[1]のようなテーブルトップシステム、iPhone[2]のような携帯電話・スマートフォン、iPad[3]のようなスレート端末など、様々な電子機器の入力インタフェースとして、タッチパネルが用いられている。それらは、複数の指を用いて入力することも可能であり、2 本の指を閉じたり開いたりすることによる画面の拡大・縮小、2 本の指を回転させることによる画面の回転などのインタラクションが実現されている。そのような従来のタッチパネルにおけるインタラクションは、指の接触した領域の座標のみを用いている単純なものが主である。単純なタッチ以外の多様な入力が可能になれば、タッチインタラクションの幅も広がると考えられる。そこで、本研究の先行研究において、より多様な入力を実現するための、入力面の柔らかいタッチパネルと多様な入力方法が提案された。先行研究において提案された多様な入力は、タッチ、プッシュ、スラスト、ツイストの 4 つである。タッチは、指でタッチパネル面に軽く触れる動作、プッシュは、指をタッチパネル面に対して垂直に強く押す動作、スラストは、指を強く押したままずらす動作、ツイストは、指を強く押したまま指を回転させる動作である。プッシュ、スラスト、ツイストの動作は、それらがどの程度の力で行われたかも認識する。

1.2　本研究の目的

本研究の目的は、先行研究で提案された入力手法を実現することである。また、複数の指による入力も可能にする。

1.3　本論文の構成

2 章では、本研究の関連研究について述べる。3 章では、本研究の先行研究である、入力面の柔らかいタッチパネルについて述べる。4 章では、予備実験を行って得られた知見を述べ、入力の認識手法を提案する。5 章では、実装方法について述べる。6 章では、本手法の入力認

[1] http://www.microsoft.com/surface/
[2] http://www.apple.com/jp/iphone/
[3] http://www.apple.com/jp/ipad/

識の精度について実験結果について述べる。7 章では、6 章で行った実験の結果をもとに、認識手法について議論する。8 章では、本研究のまとめを述べる。

第2章　関連研究

本章では、本研究の関連研究として、タッチパネルのインタラクションにおいて多様な入力を実現する手法と、弾性体の変形を認識して入力に用いる手法についての研究を示す。

2.1　タッチパネルのインタラクションにおいて多様な入力を実現する手法

本研究と同様に、タッチパネルへの入力に指の接触点のみを用いるのではなく、多様な情報の入力を目指す研究がいくつか存在する。

内藤らは、タッチパネルの操作に押し込みを導入する提案した [内藤 09]。タッチパネル面に指を軽く接触させたときと強く接触させたときの指の接触領域の面積の変化を利用している。

堀らは、タッチパネル面に対して奥行き操作感を得る操作を提案した [堀 10]。タッチパネル面に接触させた指の側面を、同じ手の親指で摩擦し、吸い上げる・押し出す動作を行う。このときに生じた摩擦音がタッチパネル面に固体音として伝わる。この固体音をタッチパネル面に装着されたマイクを使って取得し、解析することにより、操作を実現する。

Wang らは、タッチパネルの操作に指の方向を活用することを提案した [WCRI09]。タッチパネル面に接触した指の接触領域の形状が楕円形となることと、指の接触領域の面積が手のひらの方向へ拡大していくことを利用して指の指す方向を取得している。

竹岡らは、指の傾きや方向をタッチパネルのインタラクションに活用する Z-touch を提案した [竹岡 10]。Z-touch は、ディスプレイの周囲に 3 段に設置されたラインレーザーと、ディスプレイ下方に設置された高速カメラで構成される。ラインレーザーの照射と高速カメラでの撮影を同時に行うことにより、ディスプレイ上の空間の指の姿勢を認識する。

2.2　弾性体の変形を認識して入力に用いる手法

Sato らは、透明な弾性体の変形を用いた入力インタフェースである、PhotoelasticTouch を提案した [SMKF09]。PhotoelasticTouch は、液晶ディスプレイ、透明な弾性体、偏光フィルムを装着したカメラで構成される。弾性体に変形があった場合、液晶ディスプレイから発する光が光弾性効果により複屈折を起こす。このことを利用して、弾性体の変形を認識する。弾性体に圧力を加えた位置や面積、方向を計測する。PhotoelasticTouch により、つまむ、揉む、引っ張るなどのインタラクションが可能となる。

Vlack らの GelForce は、弾性体に加わった力の大きさと、その向きを分布として検出することができる触覚センサである [VMK+05]。GelForce は、2 色のマーカが埋め込まれた透明な弾性体と、それを撮影するカメラで構成される。マーカの位置の移動をカメラで検出することにより、弾性体表面に加わる力の大きさと向きを認識することができる。弾性体に力を加えるとマーカーが移動するので、それをカメラで撮影し、その変化を調べて力を感知している。このことにより、弾性体表面に対して、押し込んだり、つまんだりする入力が可能である。主にロボットハンドや拡張現実感に応用されている。

Vogt らは、入力面にゴムを用いた入力デバイスを開発した [VCHF04]。このデバイスは、ドットのパターンが埋め込まれたゴムと、それを撮影するカメラで構成される。入力面に力が加わると、変形したドットが撮影できる。このことを利用して、複数の指による入力とその圧力を認識できる。

2.3 本研究の位置付け

本研究は先に示した関連研究と同様に、タッチパネルにおけるインタラクションにおいて、より多様な入力の実現を目指すものである。関連研究における本研究の違いは、柔らかい入力面を有するタッチパネルの変形を用いて入力を行うことである。本研究で用いる入力面の柔らかいタッチパネルは、入力面にしわが生じる特徴がある。本研究では、従来の弾性体の変形の認識手法にはない、しわが生じることを用いた入力の認識を目的とする。

第3章　本研究の先行研究:入力面の柔らかいタッチパネル

本節では、本研究の先行研究である、入力面の柔らかいタッチパネルについて述べる。3.1節では、入力面の柔らかいタッチパネルのハードウェア構成と、接触検出の手法であるFTIRについて述べる。3.2節では、先行研究において提案された入力の種類について述べる。

3.1　ハードウェア構成

本研究の先行研究において、入力面の柔らかいタッチパネルのプロトタイプが作成された。そのハードウェア構成を図3.1に示す。

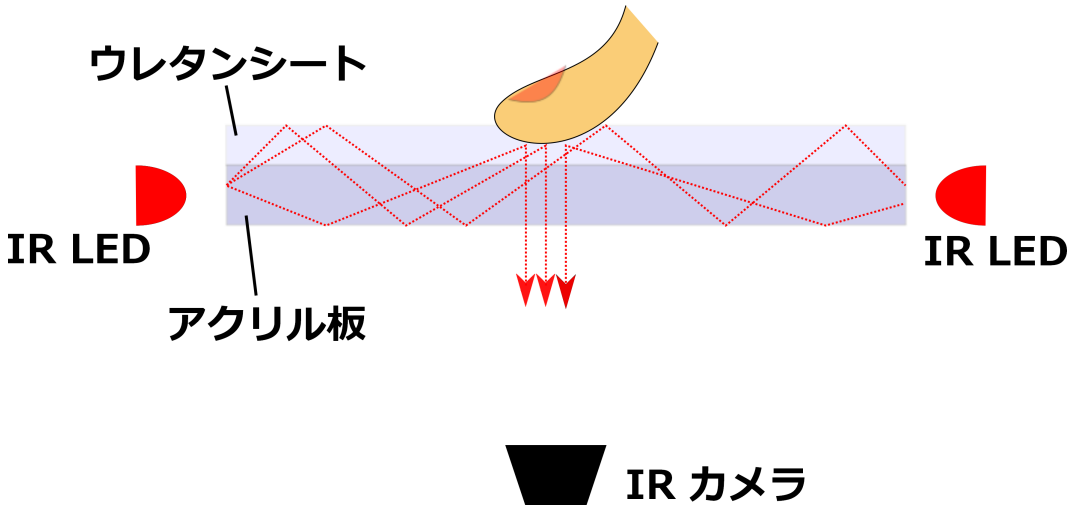

図3.1: ハードウェア構成

入力面の柔らかいタッチパネルは、透明な柔らかいウレタンシートを張り付けたアクリル板と、タッチ検出に用いる赤外線LEDと赤外線カメラで構成される。タッチ検出の仕組みには、Hanによって示されたFTIR(Frustrated Total Internal Refrection)[Han05]の原理を利用する。この仕組みを以下に述べる。

アクリル板の側面から照射された赤外線光は通常アクリル板の中で全反射を起こす。このとき、アクリルに指を接触させることにより、その接触領域においてアクリル内で全反射し

ていた赤外線が拡散反射を起こす。この拡散反射光を、タッチパネル面の下方に設置された赤外線カメラで撮影することにより、タッチパネルへの接触を検出できる。

本研究で用いるタッチパネルの入力面は、Fukumoto の PuyoSheet[Fuk09] を用いている。PuyoSheet はウレタン素材の透明な柔らかいシートである。PuyoSheet をタッチパネル面に用いた時も、先に示した FTIR と同様の現象が発生する。アクリル板側面から照射された赤外線光は、アクリルと PuyoSheet の中で全反射を繰り返す。アクリルのみのタッチパネルと異なる点は、タッチパネル面にしわが生じたときに、しわの生じた領域からも拡散反射が起こることである。このことを利用して、指の接触のみでなく、タッチパネル面に生じたしわも検出できる。

3.2 入力の種類

先行研究で提案された入力の種類は、タッチ、プッシュ、スラスト、ツイストの 4 つである。以下に、各入力の動作と、タッチパネル面の様子について述べる。

タッチ 入力面に軽く指を触れる動作である。図 3.3 に示すように、タッチパネル面に変形は生じない。

プッシュ タッチパネル面に対して垂直に指を押し付ける動作である。図 3.3 に示すように、タッチパネル面にくぼみが生じる。

スラスト タッチパネル面に指を押し付けたまま、その指をずらす動作である。図 3.4 に示すように、指をずらした方向にしわが生じる。

ツイスト ツイストは、タッチパネル面に指を押しつけたまま、指を回転させる動作である。図 3.5 に示すように、指の周囲にしわが生じる。

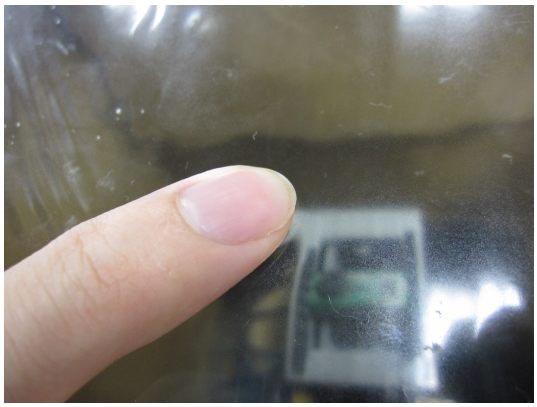

図 3.2: タッチをしたときのタッチパネル面の様子

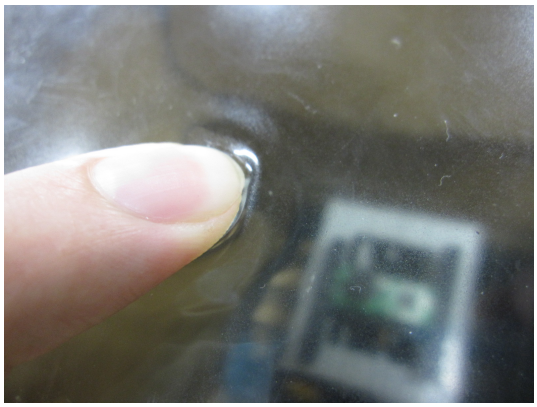

図 3.3: プッシュ動作をしたときのタッチパネル面の様子

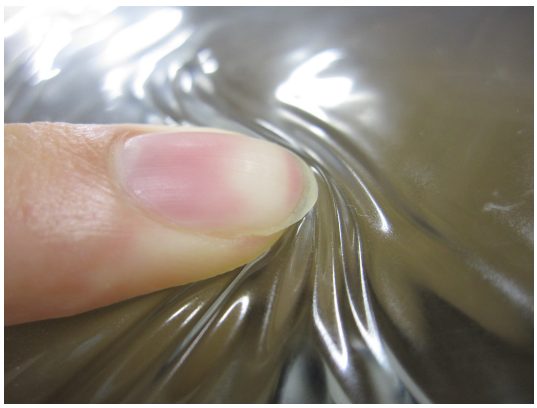

図 3.4: スラストをしたときのタッチパネル面の様子

図 3.5: ツイストをしたときのタッチパネル面の様子

第4章　柔らかいタッチパネル面に生じるしわを用いた入力の認識手法の提案

本章では、柔らかいタッチパネル面に生じるしわを用いた入力の認識の手法を示す。4.1 節では、認識手法を開発するために、各入力を行ったときの特徴を観察し、そこから得られた知見について述べる。4.2 節では、予備実験において得られた知見をもとに、入力認識のアイデアを示す。

4.1　予備実験と考察

認識手法の検討に先立って、提案された入力を行った場合にどのような特徴が得られるかを調べる実装を行った。この実装により、赤外線カメラから読み込んだ画像（以下、元画像）、元画像を 2 値化処理をした画像（以下、2 値化画像）、2 値化画像を縮退処理をした画像、2 値化画像を膨張処理をした画像の計 4 枚の画像を生成し、観察する。その結果、以下の 4 つの特徴が得られた。

1. 輝度値
2. 指の回転
3. しわベクトルの大きさ
4. 円形度

本節では、この 4 つの特徴について述べる。

4.1.1　輝度値

各入力を行ったときに元画像を図 4.1、図 4.2、図 4.4、図 4.5 に示す。まず、明確な違いが表れたのが、指の接触領域の輝度値である。輝度値は画素の明るさを表す値で、0〜255 の値をとる。図 4.1 に示すタッチをした時の指の接触領域は暗く写っているため輝度値は低く、それ以外の入力では高かった。また、図 4.3 に示すように、プッシュは入力面を強く押すほど指の接触領域の輝度値が高くなることが分かった。

図 4.1: タッチをした時の元画像

図 4.2: プッシュをした時の元画像

図 4.3: 図 4.2 より強くプッシュをした時の元画像

図 4.4: スラストをした時の元画像

図 4.5: ツイストをした時の元画像

4.1.2 指の回転

ツイストをした時は指の指す方向が変化し、その他の入力は指の指す方向が変化しない特徴がある。指の指す方向を計算するために、元画像を 2 値化する。2 値化画像からしわの発生している領域を取り除くために、縮退処理を施す。このことにより、指の接触領域に近い形状を図 4.6、図 4.7 のように抽出することができる。このように処理された領域を「指領域」と定義する。

図 4.6: スラストをした時の指領域

図 4.7: ツイストをした時の指領域

この指領域を用いて、式 4.1、式 4.2 に示す式により指の指す方向 ϕ を計算する。

$$M_{pq} = \sum_{x=0}^{w} \sum_{y=0}^{h} (x - \overline{x})^p (y - \overline{y})^q f(x, y)$$

$$\theta = \frac{1}{2} \arctan \frac{2M_{11}}{M_{20} - M_{02}} \tag{4.1}$$

$$\phi(t+1) = \begin{cases} \theta(t+1) & (|\phi(t) - \theta(t+1)| \leq 90^\circ) \\ \theta(t+1) + 180^\circ & (|\phi(t) - \theta(t+1)| > 90^\circ) \end{cases} \tag{4.2}$$

式 4.1 は指領域の重心を通る慣性主軸の傾斜角を意味しているが、方向を一意に決めるものではない。そこで、Wang らの示したアルゴリズム [WCRI09] を適用する。現在の指の方向を決定するためには、式 4.2 に示すように、過去に計算された指の方向を用いる。このことにより、指の方向を安定して計算することができる。以上のように計算された現在と過去の指の方向を比較することにより、指の回転を検出する。

4.1.3 円形度

2 値化画像を膨張処理した画像を図 4.9、図 4.9、図 4.10 に示す。この処理によって得られる領域を「指しわ領域」と定義する。

図 4.8: プッシュをした時の指しわ領域

図 4.9: スラストをした時の指しわ領域

図 4.10: ツイストをした時の指しわ領域

プッシュ、スラスト、ツイストの指しわ領域画像を比較すると、領域の形状の複雑さが異なっている。領域の形状の複雑さを意味する指標として、指しわ領域の円形度を調べる。円形度とは、領域が真円にどの程度近いかを示す度合いであり、領域の形が真円に近づくほどは値が 1 に近づき、複雑な形状ほど 0 に近づく。円形度は、領域の面積 S と周囲長 l を用いて式 4.3 で表わされる。

$$\text{円形度} = \frac{4\pi S}{l^2} \tag{4.3}$$

4.1.4　しわのベクトルの大きさ

スラストとツイストにおけるしわの出方は異なっている。スラストは指をずらした方向のみにしわが発生することに対し、ツイストは指の周囲にしわが発生する。つまり、スラスト時はしわが一方向に偏り、ツイスト時には偏りがない。このことを利用するために、指しわ領域から指領域を差し引いた「しわ領域」を求める。スラストとツイストのしわ領域を図 4.11、図 4.12 に示す。

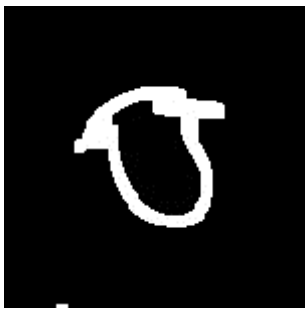

図 4.11: スラストをした時のしわ領域

図 4.12: ツイストをした時のしわ領域

指領域の重心からしわ領域の重心へのベクトルをしわベクトルと定義する。スラスト時の指領域の重心はしわ領域の重心から離れているため、しわベクトルの大きさは大きい。一方ツイストは、しわ領域の重心と指領域の重心はほぼ一致するため、しわベクトルの大きさは小さい。このことを利用し、スラストとツイストを判別する。

4.1.5　まとめ

本節の予備実験により得られた入力に対する特徴を表 4.1.5 にまとめる。指の接触領域の輝度値、指しわ領域の円形度、しわベクトルの大きさ、指の回転を用いて入力を認識する。

表 4.1: 入力に対する特徴

入力	輝度値	円形度	しわベクトルの大きさ	指の回転
タッチ	低い	0.8〜	小さい	なし
プッシュ	強いほど高い	0.8〜	小さい	なし
スラスト	高い	0.5〜0.8	大きい	ほぼなし
ツイスト	高い	0.5〜0.8	小さい	あり

4.2　入力認識のアイデア

前節の表 4.1.5 で示した入力に対する特徴をもとに、入力認識のアイデアを示す。輝度値に注目すると、タッチだけが低く、プッシュ、スラスト、ツイストでは高くなっている。したがって、輝度値を用いてタッチとそれ以外の入力を識別する。次にしわベクトルの長さ、円形度、指の回転量を用いて、プッシュ、スラスト、ツイストを識別する。ここで問題となるのが、どの特徴量も一定値を示さずに、若干の幅があることである。そこで、各入力に対する各特徴量の尤度関数を定めることを提案する。

円形度に注目すると、プッシュ時はしわが発生しないため、指しわ領域は円に近い形状になる。このときの円形度は、0.8 以上となる。スラスト、ツイスト時はしわが発生するため、指しわ領域は複雑な形状になる。このときの円形度は 0.5〜0.8 となる。また、強くスラスト、ツイストをすることによってしわが多く発生すると、指しわ領域の形状はより複雑となり、円形度は下がる。

しわベクトルの大きさに注目すると、プッシュ時はしわが発生しないため、しわ領域と指領域の重心は一致するため、しわベクトルの大きさは小さい。スラスト時は指をずらした方向にしわが多く発生する。よって、しわ領域の重心は、指領域の重心と離れているため、しわベクトルの大きさは大きくなる。プッシュ時は、指領域の重心と、しわ領域の重心がほぼ一致する。その時の値は、概ね 2.0 以下である。よって、しわベクトルの大きさは、小さい。ツイスト時は、しわは指の周りに発生し、1 方向に偏ることが少ない。この時、プッシュと同様に、指領域の重心としわ領域の重心がほぼ一致するため、しわベクトルの大きさは小さい。しかし、しわが発生する箇所に偏りがあるときは、しわベクトルが大きくなる。そのため、プッシュよりも尤度に幅を持たせる。

指の回転量に注目すると、プッシュ時は、単純に指を押しつける動作なので、回転は生じない。よって、回転量は小さい。スラスト時は理想的には回転が発生しないはずであるが、現在の実装では指の方向取得の精度が悪い。そのため、スラストの尤度には幅を持たせる。ツイスト時には、回転が発生するため、回転量の絶対値が小さいときには尤度が低いと判断する。

第5章　柔らかいタッチパネル面に生じるしわを用いた入力の認識手法の実装

本章では、4 章で述べられた手法の実装について述べる。
入力の認識処理全体の流れを図 5.1 に示す。

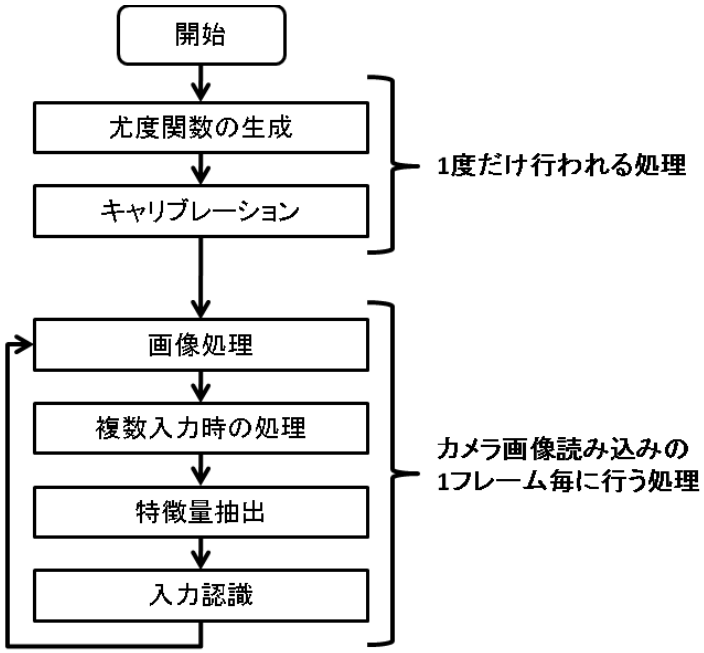

図 5.1: 入力の認識処理全体の流れ

入力を認識するための処理として、プログラム実行開始時に 1 度だけ行われる処理と、カメラ画像を 1 枚読み込むたびに行われる処理がある。プログラム実行直後に 1 度だけ行われる処理として、尤度関数の生成と、タッチパネル座標へ変換するためのキャリブレーションがある。その処理が終わると、カメラ画像読み込みの 1 フレームごとに行う処理がある。カメラから読み込まれた画像を処理し、前章で示された領域を生成する。複数の入力が生じた時は、それらが前フレームにおいて入力された座標をもとに、同一の指による入力を追跡する。また、タッチパネル面にしわが生じることに起因する問題を、ボロノイ分割により解決する。その後、画像処理により得られた領域と前フレームにおいて得られたデータをもとに

して特徴量の抽出を行う。最後に、特徴量と尤度関数を用いて入力の認識を行う。

5.1　尤度関数の生成

尤度関数は、プログラム実行開始時に 1 度だけ行われる。予備実験に基づき作成した各特徴量に対する各入力の尤度関数を図 5.2、図 5.3、図 5.4 に示す。尤度関数は表 5.1.1、表 5.1.2、表 5.1.3 に示す座標データを 3 次スプライン補間することにより生成される。尤度関数の定義域のうち、与えられた座標データの区間外は、座標データの端点の尤度を採用して定数で補外することとした。

ここに定義された尤度関数により、各特徴量に対する各入力の尤度を計算し、認識に用いる。

5.1.1　円形度に対する各入力の尤度関数

円形度に対する各入力の尤度関数を図 5.2 に示す。この関数の定義域は [0.0,1.0] とした。

表 5.1: 円形度に対する各入力の尤度関数を生成するための点列データ

入力	点列データ (円形度, 尤度)
プッシュ	(0.6,0.0),(0.79,0.01),(0.81,0.8),(0.85,0.9)
スラスト	(0.65,0.9),(0.7,0.75),(0.75,0.6),(0.8,0.0)
ツイスト	(0.5,0.5),(0.7,0.7),(0.75,0.8),(0.8,0.0)

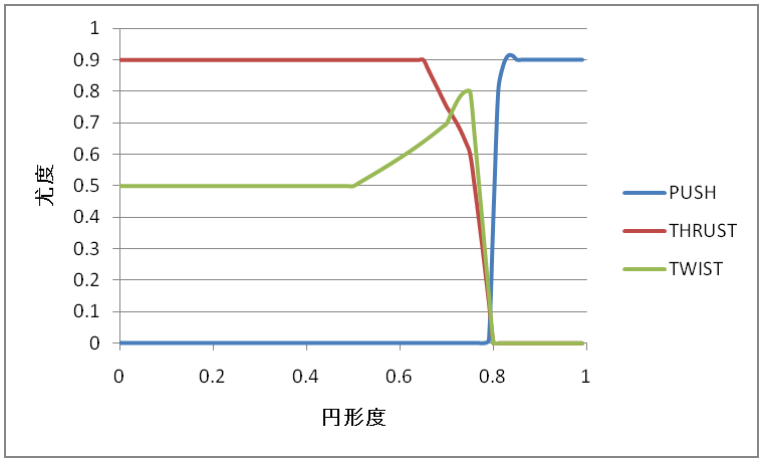

図 5.2: 円形度に対する各入力の尤度関数

5.1.2　しわベクトルの大きさに対する各入力の尤度関数

しわベクトルの大きさに対する各入力の尤度関数を図 5.3 に示す。この関数の定義域は [0.0,30.0] とした。

表 5.2: しわベクトルの大きさに対する各入力の尤度関数を生成するための点列データ

入力	点列データ (しわベクトルの大きさ, 尤度)
プッシュ	(0.0,0.9),(1.0,0.8),(3.0,0.2),(5.0,0.0)
スラスト	(1.0,0.0),(2.0,0.1),(3.0,0.4),(5.0,0.6),(7.0,0.8),(10.0,1.0)
ツイスト	(0.0,0.5),(1.0,0.7),(2.0,0.8),(5.0,0.4),(7.0,0.3),(10.0,0.0)

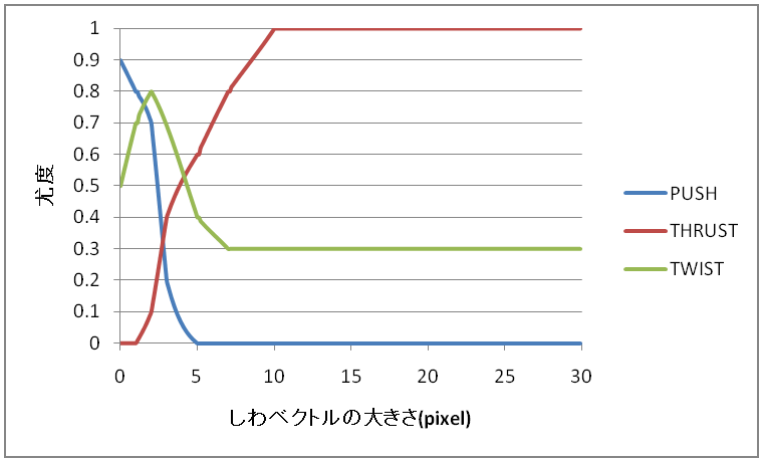

図 5.3: しわベクトルの大きさに対する各入力の尤度関数

5.1.3　回転量に対する各入力の尤度関数

回転量に対する各入力の尤度関数を図 5.4 に示す。この関数の定義域は [-90,90] とした。

表 5.3: 回転量に対する各入力の尤度関数を生成するための点列データ

入力	点列データ (回転量, 尤度)
プッシュ	(-10.0,0.1),(-5.0,0.3),(0.0,1.0),(-5.0,0.3),(10.0,0.1)
スラスト	(-10.0,0.3),(-5.0,0.5),(0.0,1.0),(-5.0,0.5),(10.0,0.3)
ツイスト	(-25.0,0.9),(-15.0,0.8),(-12.0,0.6),(-10.0,0.2),(-5.0,0.0),(0.0,0.0),(-5.0,0.0),(10.0,0.2),(12.0,0.6),(15.0,0.8),(25.0,0.9)

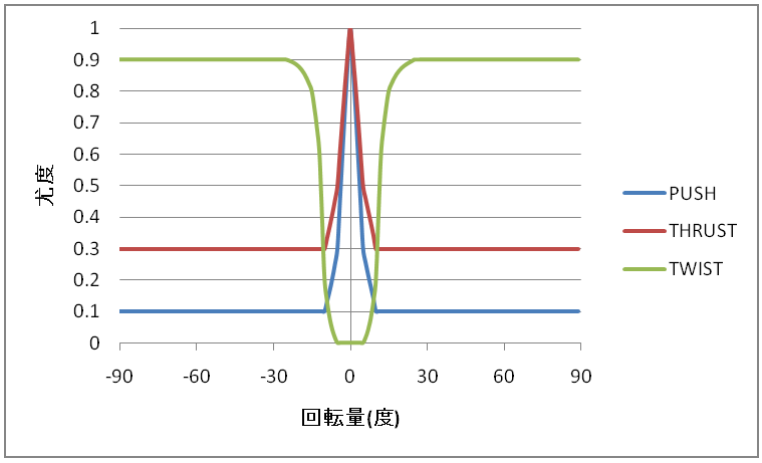

図 5.4: 回転量に対する各入力の尤度関数

5.2 キャリブレーション

カメラの座標と入力面の座標が一致するとは限らない。そこで、キャリブレーションが必要となる。図 5.5 にカメラから読み込まれた画像を示す。図中の赤い印は、カメラ画像におけるタッチパネル面の四隅を示している。この四隅の座標を、タッチパネル面の座標へマッピングする 3×3 の行列を計算する。このことにより、図 5.6 のように画像が変換される。

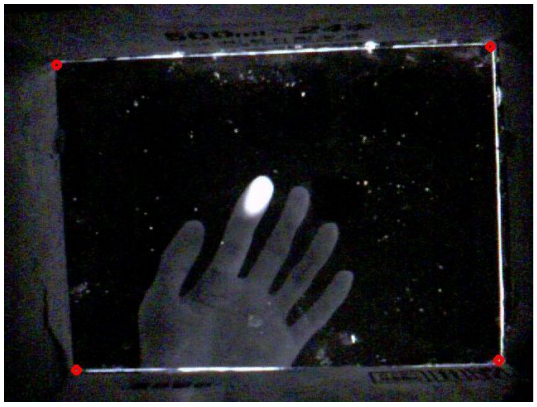

図 5.5: カメラから読み込まれた画像

図 5.6: 変換後の画像

5.3 特徴量抽出のための画像処理

カメラ読み込みの 1 フレーム毎に行う画像処理を図 5.7 に示す。この画像処理で取得する値は、各入力における指の接触領域の重心点、指の指す方向、指の接触領域の平均輝度値、円

形度、しわの重心である。

図 5.7a は、カメラから読み込まれた画像（以下、元画像）をタッチパネル面の座標にキャリブレーションした画像である。図 5.7b は、図 5.7a をグレースケール化した後に平滑化し、背景差分を取った画像である。図 5.7c は、図 5.7b を 2 値化した画像である。2 値化閾値に基づき、閾値以上なら画素値を 255、閾値未満なら画素値を 0 とする。図 5.7d は、図 5.7c に対して縮退処理を行い、その後膨張処理を行った画像である。先に縮退処理を行うことにより、図 5.7c におけるしわの領域を縮小させている。図 5.7e は、図 5.7d における各領域を取りだした画像である。図 5.7d に対して、まずラベリング処理を行う。ラベリングを行った各領域に対して、領域の面積 (ピクセル数) を計算する。面積の閾値の最大値・最小値をあらかじめ設定しておき、この範囲に入らない領域はノイズとして除外する。各指領域について、指の重心座標、指の指す方向、図 5.7b における指領域の平均輝度値を計算する。

図 5.7f は、図 5.7c に対して膨張処理を行い、その後縮退処理を行った画像である。先に膨張処理を行うことにより、図 5.7c におけるしわの領域と指の領域を連結させている。図 5.7g は、図 5.7f における各領域を取りだした画像である。指領域を導出した時と同様の手順により、図 5.7f をラベリング処理し、領域の面積によりノイズとなる除去する。各指しわ領域について、円形度を計算する。

図 5.7h は、図 5.7f から図 5.7d を指し引いた画像である。図 5.7i は、図 5.7h における各領域を取りだした画像である。指領域を導出した時と同様の手順により、図 5.7h をラベリング処理し、領域の面積によりノイズを除去する。

5.4 特徴量抽出

前節で得られた指領域、指しわ領域、しわ領域から、入力の認識に用いる特徴量を計算する。

5.4.1 円形度

指しわ領域から、円形度を計算する。

5.4.2 しわベクトルの大きさ

しわベクトルとは、指領域の重心点 $f(x, y)$ からしわ領域の重心点 $w(x, y)$ への方向ベクトルのことである。その大きさ L は式 5.1 で表わされる。

$$L = \sqrt{(w_x - f_x)^2 + (w_y - f_y)^2} \tag{5.1}$$

5.4.3 指の回転量

指領域から指の指す方向を式 4.1、式 4.2 を用いて計算する。指の回転量は、現在のフレームにおける指の方向と、前フレームで取得した基準となる指の方向との差で定義する。

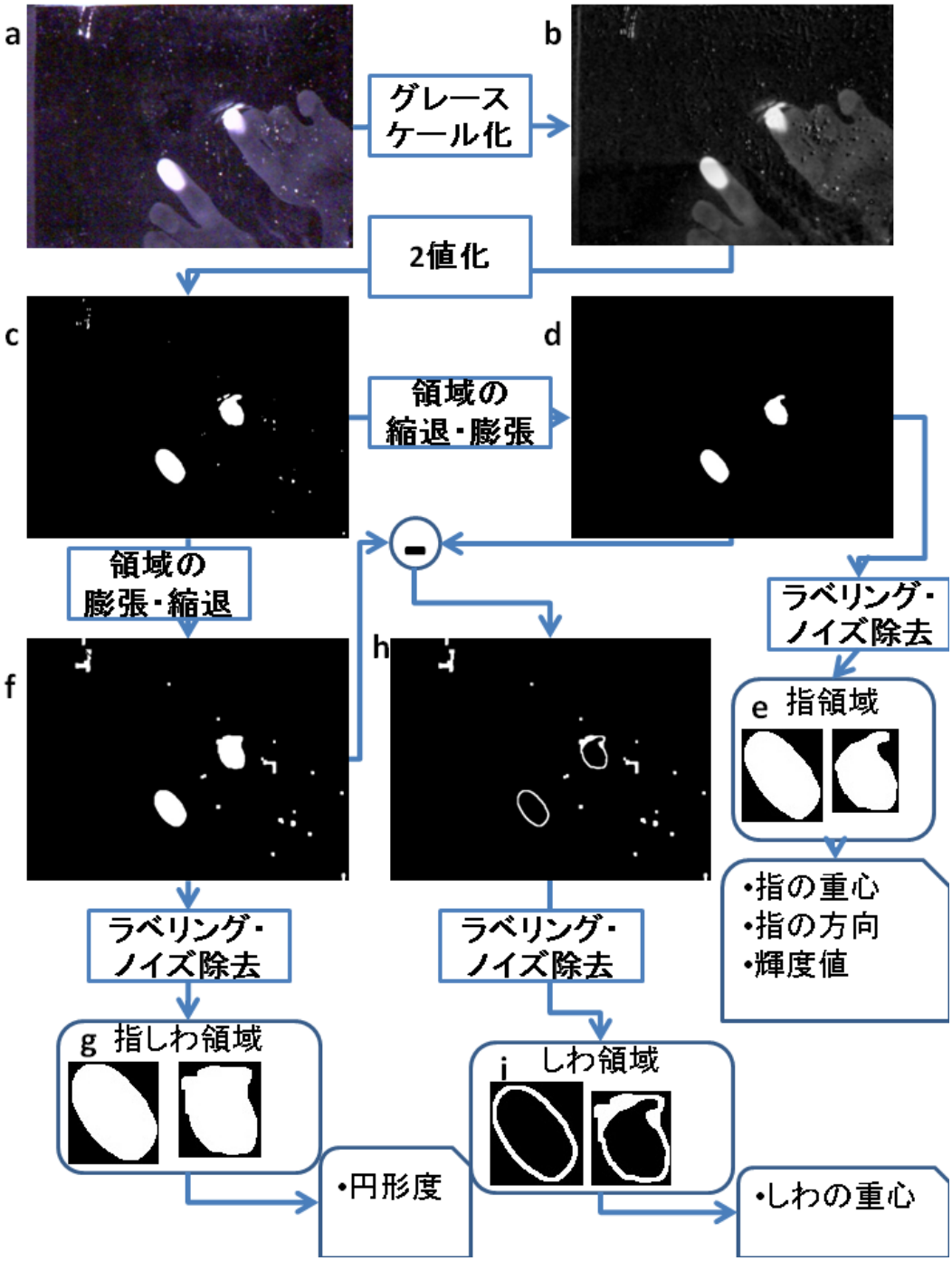
a
グレースケール化
b
2値化
c
領域の縮退・膨張
d
領域の膨張・縮退
ラベリング・ノイズ除去
f
h
e 指領域
•指の重心
•指の方向
•輝度値
ラベリング・ノイズ除去
ラベリング・ノイズ除去
g 指しわ領域
i しわ領域
•円形度
•しわの重心

図 5.7: 画像処理の手順

5.5　複数入力時の処理

本節では、複数の指による入力があった時の処理について述べる。その処理内容は、指領域の重心点の追跡と、近接した接触領域の分割である。

5.5.1　指領域の重心点の追跡

前フレームで検出された指領域の重心点と現在のフレームで検出された指領域の重心点について、すべての 2 点間の距離を計算し、定められた距離以内且つ一番小さかったものを同一の指による継続の入力とみなす。

5.5.2　近接した接触領域の分割方法

入力の認識手法において、2 値化画像を膨張させることによってしわを抽象化し、入力の認識に用いることを考えた。しかし、図 5.8 に示すような近接した複数の入力が発生した場合、問題が発生する。それは、タッチパネル面に生じたしわによって、近接した複数の入力の指しわ領域が、1 つの領域とみなされることである。

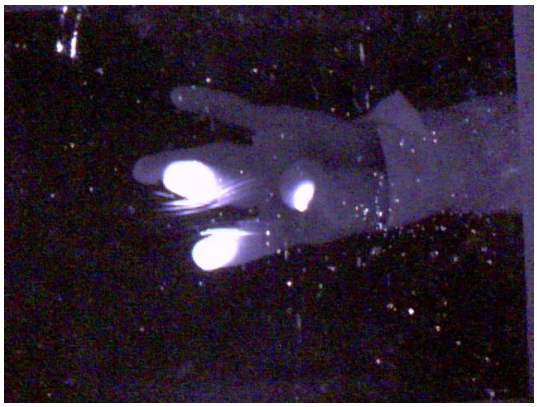

図 5.8: 近接した入力

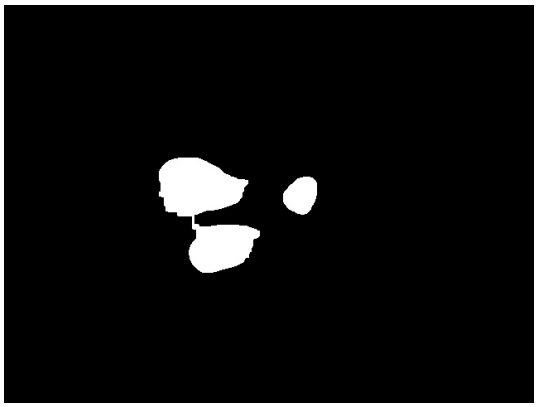

図 5.9: 膨張処理によって 1 つの領域となる例

この解決策として、指領域の重心点を母点としたボロノイ図を作成することにより、指領域を分割する。まず、ボロノイ境界を求めるために、離散ボロノイ図を作成する。離散ボロノイ図とは、画素の集合からなる平面上で考えたボロノイ図である。画素毎に最も近い母点へ所属させることにより離散ボロノイ図を求める。画素の所属毎に画素値を設定し、その結果を図 5.10 に示す。この画像のエッジを検出することにより、ボロノイ境界 (図 5.11) を求める。最後に、指しわ画像からこの境界を差し引くことにより、図 5.12 のように領域が分割される。

図 5.10: ボロノイ図

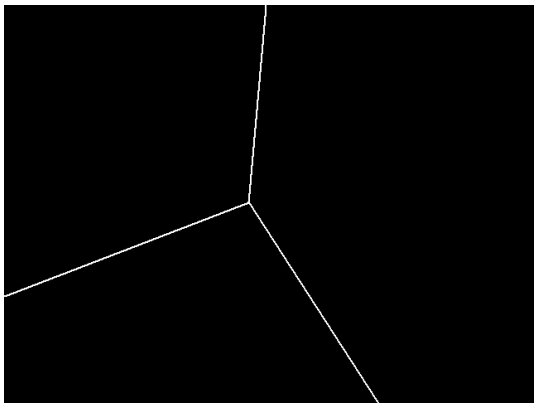

図 5.11: ボロノイ境界

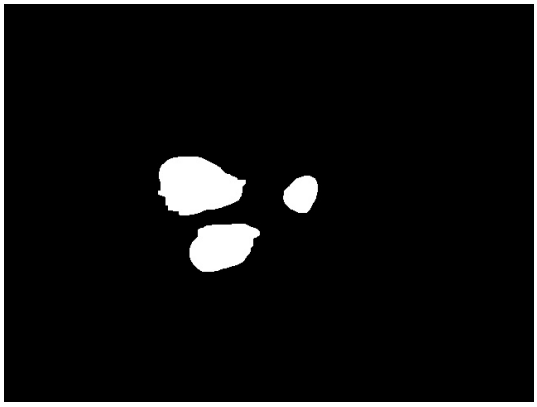

図 5.12: 分割された領域

5.6 入力の認識

プッシュ、スラスト、ツイストの最終的な尤度は、各特徴量に対する各入力の尤度を掛け合わせた値となる。これらの値を利用して、入力の認識を行う。本実装では、揺らぎを防ぐために認識結果の平滑化を行う。入力が検出されてから数フレーム間はタッチパネル面に接触した指が安定しないため、認識結果に揺らぎが生じる。

まず、図 5.13 に示す認識の状態遷移図について説明する。

図 5.13 中の L_{pu}, L_{th}, L_{tw} はそれぞれ、プッシュの尤度、スラストの尤度、ツイストの尤度である。初期状態は、前フレームにおける認識結果である。現フレームで初めて入力が生じた場合、まずはタッチに遷移する。これは、指の回転量の計算に最低 2 フレーム分の指の方向情報が必要なためである。認識に指の回転量を用いないタッチへ遷移することにより、このときに指の方向を計算する。タッチ-プッシュ間の遷移は、輝度値が閾値を越えたかどうか応じて生じる。プッシュ-スラストとプッシュ-ツイスト間の遷移は、尤度値が最大となった入力へ遷移する。スラスト-ツイスト間の遷移は、入力の尤度が最大であることに加えてもう 1

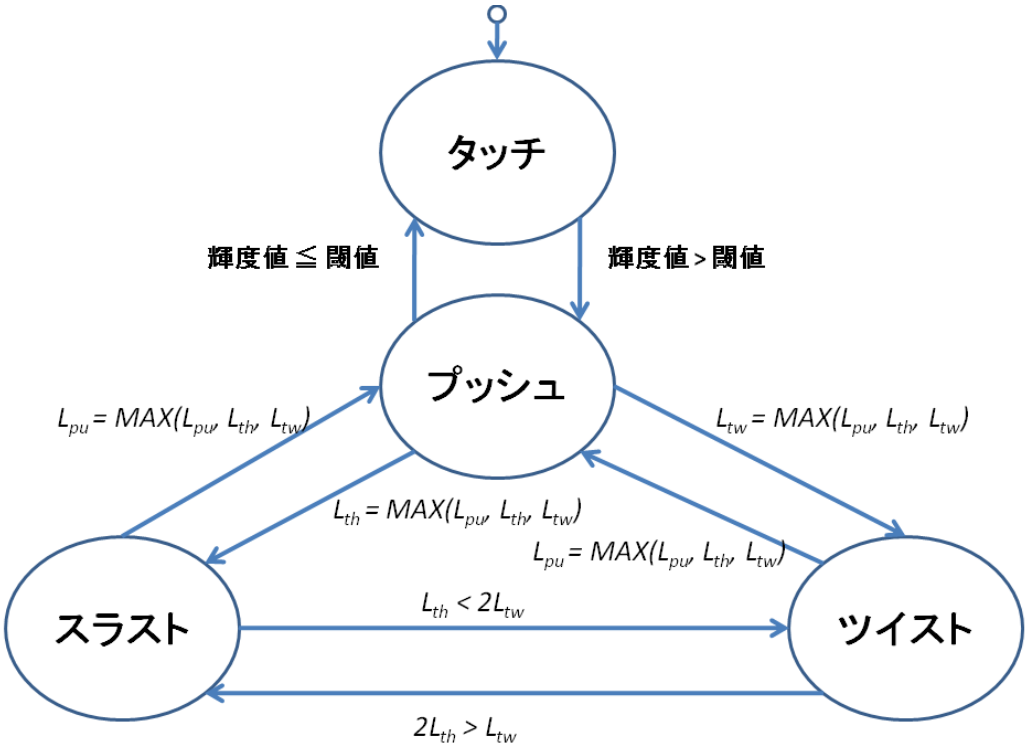

図 5.13: 認識の状態遷移

つ条件を設けた。スラストからツイストへ遷移は L_{tw} が L_{th} の 2 倍以上、ツイストからスラストへ遷移は L_{th} が L_{tw} の 2 倍以上の時とした。これは、スラストとツイスト間の認識結果の揺らぎをより強く抑える効果がある。図 5.13 のエッジに示されている条件に当てはまらない場合は、前フレームの認識結果の状態にとどまる。また、最大の尤度値と 2 番目の尤度値が極めて近い場合は誤差の範囲とみなし、現在の状態から遷移しない。本実装ではこの値を 0.005 と定めた。

上記の条件により遷移した状態は、長さ 3 のキューに保存される。現在の状態と前フレームの認識結果に違いがあった場合、キューの保存されている認識結果がすべて一致するならば遷移した状態を現フレームにおける認識結果に適用する。そうでない場合は、前フレームの認識結果を現フレームの認識結果とする。

第6章　認識精度の実験と実験結果

6.1 節では、入力の認識率を調べる実験と、システムが入力に利用できるパラメータの範囲と人間が入力できる分解能を調べる実験について述べる。6.2 節は、実験結果とその考察である。

6.1　実験内容

本節では、実験の目的とその内容について述べる。

6.1.1　実験概要

本実験の目的は 2 つある。まず 1 つ目は、実験者により指定された入力を被験者が行い、それを認識プログラムが正しく認識するかどうか確かめることである。2 つ目は、実験者により指定された入力を、特定のパラメータを制限した状態で被験者が行い、被験者が入力できる分解能とシステムが入力に利用できるパラメータの範囲を確かめることである。この 2 つの目的のために、6.1.3 節に示す 2 つのタスクを行う。実験中にシステムが認識した結果は、すべてデータベースに保存される。また、実験者は被験者がタスクを行う様子を観察する。

6.1.2　被験者

被験者は 22〜23 歳の男性 5 名であった。

6.1.3　タスク

本節では、入力の認識率を測定するためのタスク（タスク 1）と、システムが入力に利用できるパラメータの範囲と人間が入力できる分解能を測定するタスク（タスク 2）について述べる。なお、被験者は 5 名とも右手を用いて入力を行った。

タスク 1

タスク 1 は、被験者が実験者により指定された入力を行うタスクである。指定された入力を被験者が行い、システムが連続して 1000ms 以上正解を認識し続けたら、成功となり、タ

スクは終了する。指定された入力を被験者が行い、システムが指定の入力を一度認識したが、途中でそれ以外の入力であると認識したら、失敗となる。

本実験では、20 試行 × 4 種類の入力 × 被験者 5 人で、合計 400 回のタスクが行われた。

タスク 2

各入力において被験者に制限が課されるパラメータとその値の範囲を表 6.1.3 に示す。パラメータの範囲は予備実験において実際に観測された範囲を適用した。

表 6.1: 各入力において制限されるパラメータとその範囲

入力	パラメータ	範囲
プッシュ	輝度値	170-235
スラスト	指の重心の移動量	0〜30(pixel)
ツイスト	指の回転量	0〜60(度)

各パラメータは、各入力の強さを疑似的に表している。本実験では、表 6.1.3 に示すパラメータの範囲を 10 段階に区切る。つまり被験者は、プッシュは輝度値を 6.5 刻みで、スラストは指の重心の移動量を 3.0pixel 刻みで、ツイストは指の回転量を 6 度刻みで制御することとなる。

被験者は、入力する力を制御して、オブジェクトを目標の領域内に収めるタスクを行う。被験者には、目標とする領域を示す円と、入力する力に応じて大きさが変化する円が図 6.2 のようにディスプレイに提示される。紫色の円は入力の強さに応じて半径が変化し、現在入力している強さを意味する。緑色の円は目標の最小値と最大値を意味している。被験者は、ディスプレイに表示される視覚フィードバックにより、現在入力しているパラメータ値を確認できる。

図 6.1: 実験の様子

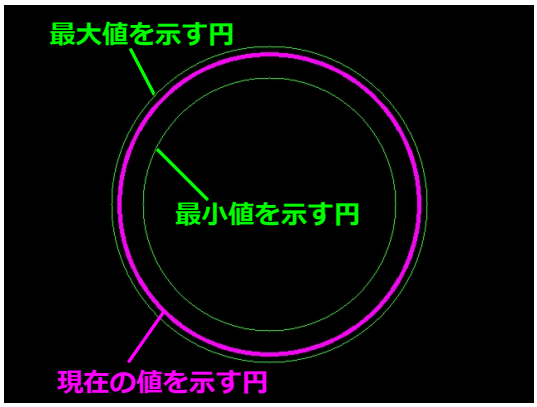

図 6.2: 提示される視覚フィードバック

入力が目標値に収まっている状態を 1000ms 継続することにより、タスクは成功となる。被験者が初めに指を接触させた時から時間を計測し、5000ms 以内にタスクを完了させる。

本実験では、10 段階 × 5 試行 × 4 種類の入力 × 被験者 5 人で、合計 1000 回のタスクが行われた。

6.1.4 実験手順

以下に実験の手順を示す。

1. 実験の目的について説明する
2. 入力方法を説明する
3. 実験者が被験者にタスク開始を合図し、システムは記録を開始する
4. 被験者はタスクを遂行する

6.2 実験結果と考察

本節では、タスク 1 とタスク 2 の実験結果を示し、実験結果の考察を行う。

6.2.1 タスク 1 の結果と考察

各入力の認識結果を表 6.2.1 に示す。

表 6.2: 各入力の成功率

入力	プッシュ	スラスト	ツイスト (時計回り)	ツイスト (反時計回り)
プッシュ	95%	5%	0%	0%
スラスト	0%	70%	29%	1%
ツイスト (時計回り)	4%	14%	81%	1%
ツイスト (反時計回り)	12%	18%	0%	70%

プッシュをプッシュと認識する割合は 95% となり、5% はスラストと認識された。誤認識の理由として、タッチパネル面に利用している PuyoSheet の気泡がノイズとなっていると考えられる。プッシュを認識する際に、指しわ領域の円形度が高いことを用いているが、気泡の影響で円形度が低くなってしまうことがある。PuyoSheet が気泡の入っていない理想的な状態であれば、プッシュの認識率は 100% にできると考えられる。

スラストを認識する割合は 70% となった。プッシュと誤認識されることはなく、29% が時計回りのツイスト、1% が反時計回りのツイストと誤認識された。誤認識の理由は、指領域の

形状のゆらぎである。指領域を楕円形とみなし、その長軸方向を指の方向としている。スラストを行う際に、指の先端だけが接触している状態になることがある。このとき、指領域は円形に近くなり、楕円の長軸を安定して求めることができなくなる。その結果、大きな回転量が生じてツイストの尤度が高くなり、ツイストへの誤認識率が上がったと考えられる。

ツイストにおけるスラストへの誤認識は、指の重心が無意識に移動してしまうことが原因であると考えられる。指を回転させながら指を移動させると、しわが指の移動方向に多く発生してしまう。結果として、しわベクトルの長さは長くなり、スラストと認識されてしまう。反時計回りのツイストのうち、12% がプッシュと認識された。この時のタッチパネル面の様子を実験者が観察した結果、指を反時計回りに回転させたときにしわが生じていなかった。また、時計回りと反時計まわりを比較したとき、反時計回りのほうが誤認識率が高くなった。インフォーマルに収集した被験者の意見からも、反時計回りのほうが動かしづらいという感想が得られた。

6.2.2　タスク 2 の結果と考察

各入力のパラメータを制限した状態における入力の認識結果と考察を以下に述べる。

プッシュの輝度値を制限した状態における入力の認識精度

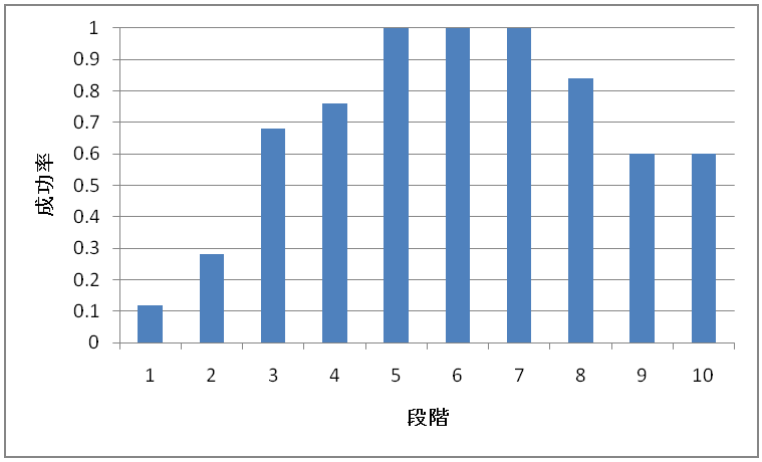

図 6.3: プッシュの各段階における成功率

プッシュの各段階における成功率を、図 6.3 に示す。段階 1 と 2 においては、低い成功率を示したが、段階 5〜7 においては、成功率 100% となった。

プッシュの各段階における成功時のタスク達成時間図 6.4 に示す。段階 1〜3 においては、タスク達成に 2500ms 以上費やしたが、段階 4〜10 においては、2000ms 以下となった。この区間は、強さを素早く制御できると言える。

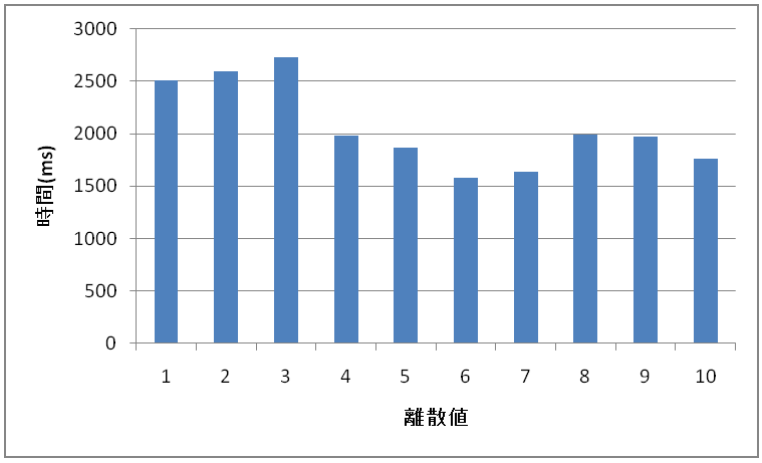

図 6.4: プッシュの各段階における成功時のタスク達成時間

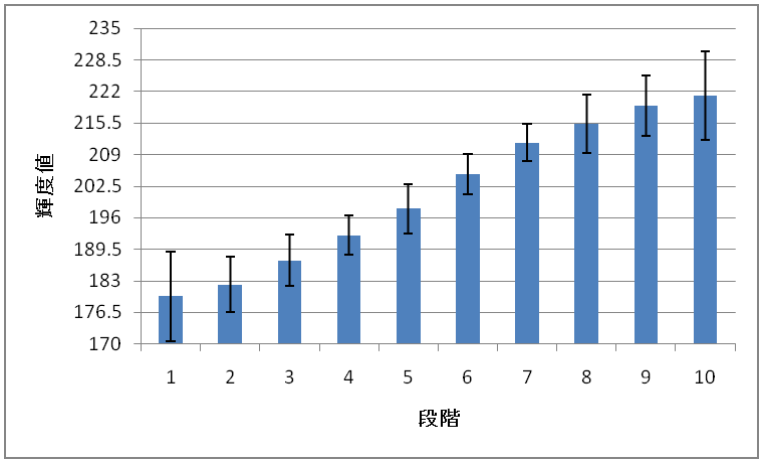

図 6.5: プッシュの各段階における輝度値の平均値と標準偏差

図 6.5 に示すプッシュの各段階における輝度値の平均値と標準偏差から、輝度値が低い段階と高い段階において、標準偏差は高くなり、制御が難しいことを示している。逆に中間の段階では、標準偏差は低くなり、この区間では輝度値を安定させることができると言える。また、段階 1 と 9〜10 については、輝度値の平均値がそもそも目標値に到達しなかった。

これらを総合すると、プッシュの輝度値をプッシュの強さとしてインタラクションに用いるときは、段階 1〜3、4、5、6、7、8〜10 の 6 段階を利用するのが適当であると考えられる。

スラストの指の移動量を制限した状態における入力の認識精度

図 6.6 に、プッシュの各段階における成功率を示す。段階 4 において成功率は 80% を越えたが、そこを中心に段階が下がるもしくは上がるほど成功率は下がった。

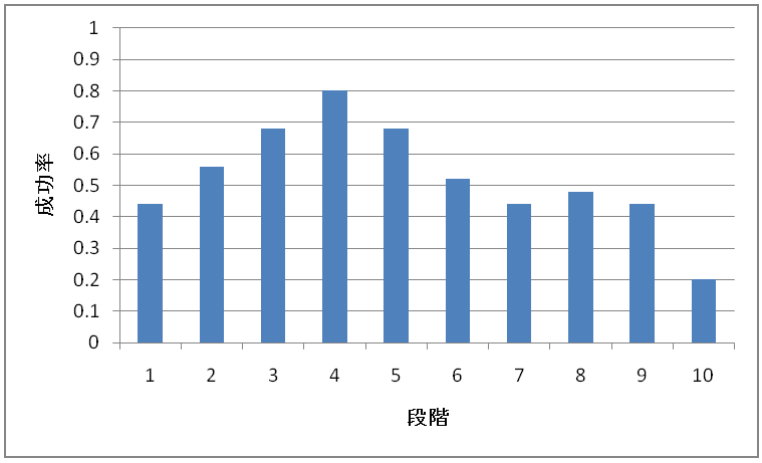

図 6.6: スラストの各段階における成功率

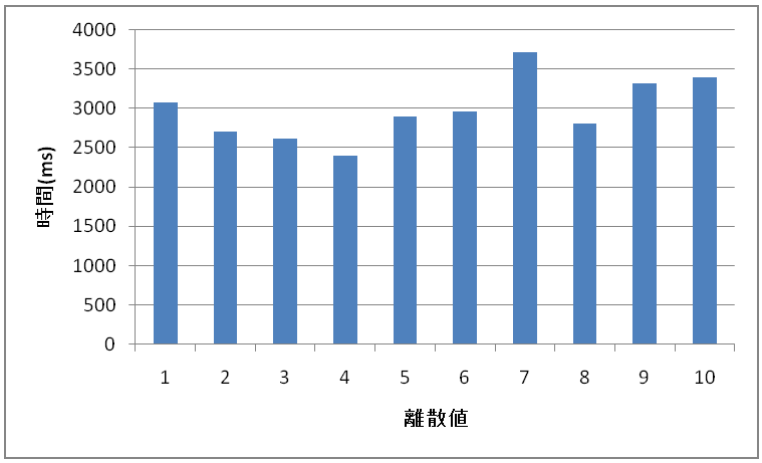

図 6.7: スラストの各段階における成功時のタスク達成時間

また、図 6.7 に示す成功時のタスク達成時間も段階 4 が最小となり、2392ms だった。しかし、段階 4 以外ではすべて 2500ms を上回る結果となった。プッシュと比較すると、全体的に制御に時間がかかったと言える。

スラストの各段階における移動量の平均値と標準偏差を図 6.8 に示す。標準偏差は、段階が上がるごとに大きくなり、指の移動量を大きくしようとするほど制御が難しかったことを意味している。また、平均値が目標値内に収まったのは段階 2 と 3 だけであった。また、全体を通して、平均値が目標値を下回る結果となった。実験者の観察より、以下に理由を推察する。被験者は、視覚フィードバックにより、入力値が目標値を越えたことを察知すると、指をずらす力を弱めることにより入力値を制御しようとする。その際に、被験者が想像している以上に指が移動してしまい、今度は入力値が目標値を下回る、または誤認識が発生する。これ

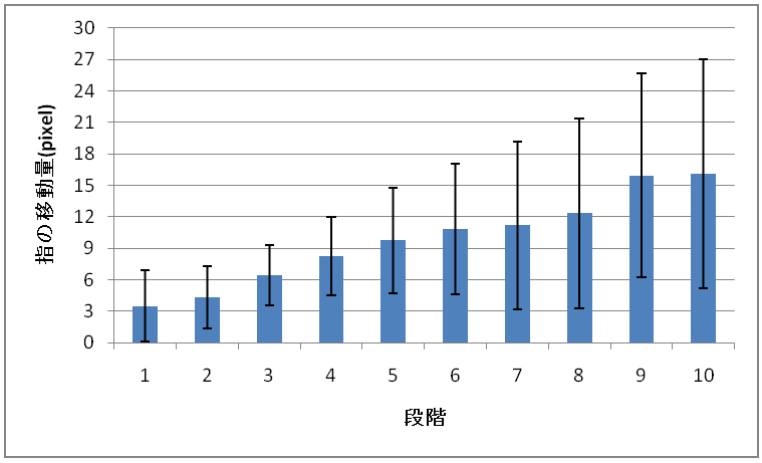

図 6.8: スラストの各段階における移動量の平均値と標準偏差

は、PuyoSheet の弾性によるものであり、弾性力と指をずらす強さを考慮して指を移動させることが難しかったと考えられる。被験者は試行の過程でこのことを学習し、指を少しずつずらすことにより、タスク達成を試みたと考えられる。ゆえに、各段階における指の移動量の平均値は目標値を下回った。

ツイストの指の回転量を制限した状態における入力の認識精度

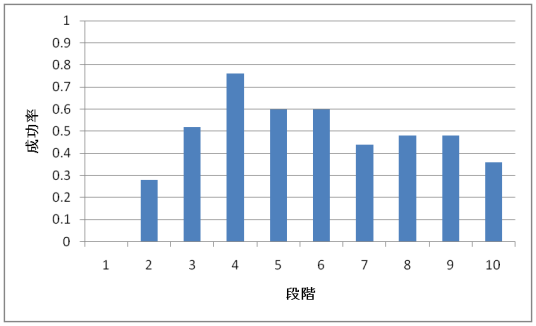

図 6.9: ツイスト (時計回り) の各段階における成功率

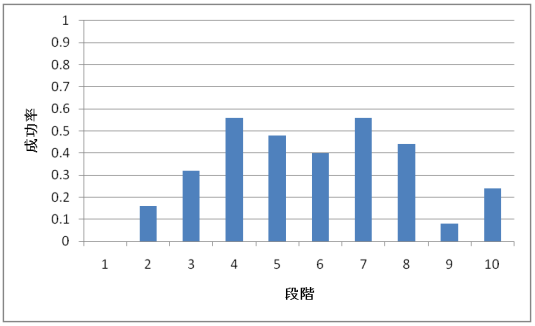

図 6.10: ツイスト (反時計回り) の各段階における成功率

ツイストの各段階における成功率を図 6.9、図 6.10 に示す。表 6.2.1 に示したとおり、反時計回りと比較して時計回りのほうが全体的に高い成功率を示した。一番高い成功率を示したのが、両方とも段階 4 であった。段階 1 は、両方とも成功率 0 となった。段階 1 を角度に換算すると 0〜6 度であり、この角度におけるスラストは、図 5.4 で示した尤度関数からもほとんど出現しないことが予想できる。

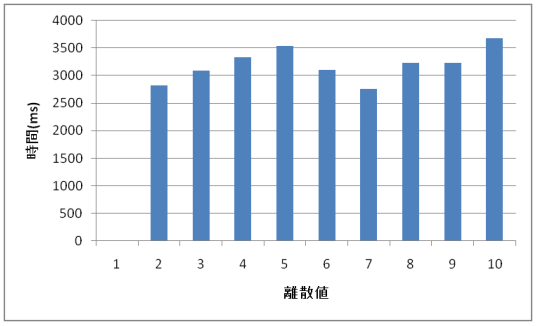

図 6.11: ツイスト (時計回り) の各段階における成功時のタスク達成時間

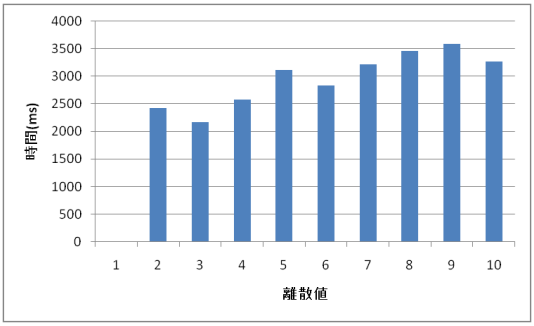

図 6.12: ツイスト (反時計回り) の各段階における成功時のタスク達成時間

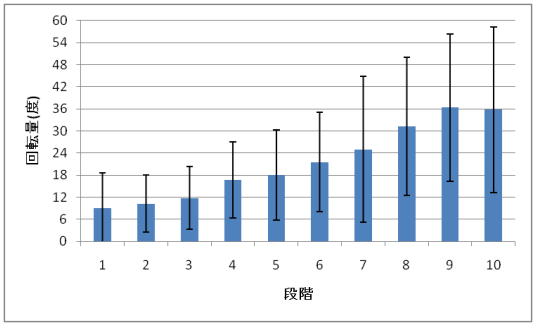

図 6.13: ツイスト (時計回り) の各段階における回転量の平均値と標準偏差

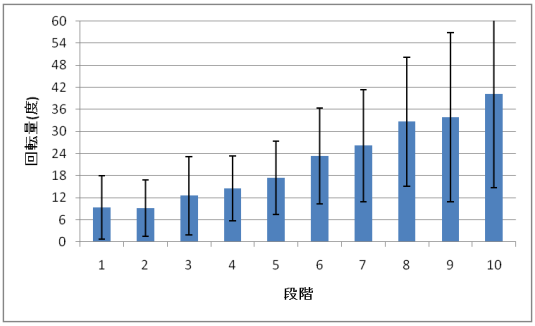

図 6.14: ツイスト (反時計回り) の各段階における回転量の平均値と標準偏差

各段階における成功時のタスク達成時間を図 6.11、図 6.12 に示す。段階 8、9 を除いて、時計回りのツイストのほうが反時計回りのツイストよりタスク達成に要した時間が長かった。ほとんどの段階において、タスク達成に 2500ms 以上を要していることから、スラストと同様に、制御が難しかったと考えられる。

各段階における回転量の平均値と標準偏差を図 6.13、図 6.14 に示す。スラストと同様に、標準偏差は、段階が上がるごとに大きくなり、指の移動量を大きくしようとするほど制御が難しかったことを意味している。また、入力の平均値が目標値を下回る結果も、スラストと同様の理由が当てはまると考えられる。それは、指をねじる強さを強い状態から弱い状態へ制御することが難しいということである。

第7章　議論

本章では、6 章における実験の結果を踏まえて、議論をする。

7.1　提案手法の考察

実験のタスク 1 では、プッシュの認識率が 95%、スラストの認識率が 70%、時計回りのツイストの認識率が 81%、反時計回りのツイストの認識率が 70% という結果になった。

誤認識を生じたタスクにおいては、著者の想定していない、個人差による入力の仕方のばらつきが観測された。例えば、スラストをする時に指が浮く、ツイストの時に指の重心がずれる、またはしわがうまく生じないということが観測された。このことは、回転量、しわベクトルの長さ、円形度それぞれに誤差を与える。この誤差について更に調査し、誤差を許容するような尤度関数の再設計が必要である。

7.2　アプリケーション開発時のインタラクション手法

入力のパラメータを制限した状態での認識精度実験で、各段階における成功率、タスク達成時間、パラメータの平均値と標準偏差を調査した。実験結果より、入力が正確に行えるパラメータ範囲、または入力値を素早く目標値に合わせられるパラメータ範囲が存在することが分かった。よって、柔らかいタッチパネルを利用するアプリケーションを設計する際には、インタラクションの設計が熟考されるべきである。また、本研究で実験した 10 という段階数は、スラストとツイストをするにはかなり細かい指の制御が必要であり、成功率は低く、タスク達成時間には概ね 2500ms 以上を費やした。素早く正確な入力を要求されるアプリケーションを設計するときには、段階数を減らすべきである。なぜなら、成功率と入力の早さは段階数に反比例すると考えられるからである。

ツイストの時計回りと反時計回りでは、入力の認識率に違いが生じた。インタラクションを設計する際に、ツイストの時計回りと反時計回りで逆の操作を割り当てることが考えられる。このときに、2 種類のツイストに同じパラメータ幅の分解能を与えるのではなく、反時計回りのツイストのしにくさを考慮した入力に補正するような処理が考えられる。

7.3　より完成度の高いハードウェアの必要性

本研究で使用した入力面の柔らかいタッチパネルは、まだ試作の段階であり、ハードウェアに起因するノイズが数多く観測された。ノイズの主な原因は、PuyoSheet 内に生じた気泡、PuyoSheet とアクリル板の間に生じた気泡、アクリル板のたわみ、アクリル板のずれである。ノイズは、入力の誤認識を生むだけでなく、ノイズ除去に要する計算時間も増やす結果となる。また、本研究に用いている FTIR のような赤外線反射光を利用したタッチ検出技術は、環境光の影響を受けやすい問題も存在する。本研究を進める上でも、窓から差し込む日光や蛍光灯の影響を受けた。赤外線 LED のパルス発光と同期して赤外線カメラでの撮影を行い、動的にノイズを除去できるシステムを構築することが望まれる。今後は、完成度の高いハードウェアの作成方法を検討し、その上で認識アルゴリズムの開発を進めていくべきである。

第8章　まとめ

本研究では、柔らかいタッチパネル面に生じるしわを用いた入力認識の手法を提案し、認識プログラムの実装を行った。また、入力の認識率と入力のパラメータを制御することにより分解能を測定する評価実験を行った。その結果、プッシュの成功率は95%、スラストは70%、時計回りのツイストは81%、反時計回りのツイストは71% となった。入力の認識率の結果より、個人差による特徴量の違いをより多く観察するための基礎的な実験が必要であると考察した。分解能を測定する実験の結果においては、各段階におけるタスクの成功率と達成時間を計測し、パラメータを制御しやすい区間があることを発見した。また、その結果により、本研究を利用したアプリケーション開発をする際のインタラクション手法について考察した。

今後は、認識手法の改善を行いたいと考えている。そのために、より多くの被験者実験を行い、各個人による入力時の指の動きの違いを観察し、データを収集したい。

謝辞

本論文の執筆にあたって、指導教員である田中二郎先生、志築文太郎先生をはじめ、三末和男先生、高橋伸先生には、研究についてご指導をいただきました。また、NTT ドコモの福本雅朗氏には、本研究のハードウェアに用いた PuyoSheet を提供していただくとともに、研究に関するアドバイスをいただきました。インタラクティブプログラミング研究室の皆様、特に WAVE チームの皆様には公私共にお世話になりました。ここに深く感謝いたします。

参考文献

[Fuk09] Masaaki Fukumoto. Puyosheet and puyodots: simple techniques for adding “button-push” feeling to touch panels. In *Proceedings of the 27th international conference extended abstracts on Human factors in computing systems*, CHI ’09, pp. 3925–3930, New York, NY, USA, 2009. ACM.

[Han05] Jefferson Y. Han. Low-cost multi-touch sensing through frustrated total internal reflection. In *Proceedings of the 18th annual ACM symposium on User interface software and technology*, UIST ’05, pp. 115–118, New York, NY, USA, 2005. ACM.

[SMKF09] Toshiki Sato, Haruko Mamiya, Hideki Koike, and Kentaro Fukuchi. Photoelastictouch: transparent rubbery tangible interface using an lcd and photoelasticity. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, UIST ’09, pp. 43–50, New York, NY, USA, 2009. ACM.

[VCHF04] Florian Vogt, Timothy Chen, Reynald Hoskinson, and Sidney Fels. A malleable surface touch interface. In *ACM SIGGRAPH 2004 Sketches*, SIGGRAPH ’04, p. 36, New York, NY, USA, 2004. ACM.

[VMK+05] Kevin Vlack, Terukazu Mizota, Naoki Kawakami, Kazuto Kamiyama, Hiroyuki Kajimoto, and Susumu Tachi. Gelforce: a vision-based traction field computer interface. In *CHI ’05 extended abstracts on Human factors in computing systems*, CHI ’05, pp. 1154–1155, New York, NY, USA, 2005. ACM.

[WCRI09] Feng Wang, Xiang Cao, Xiangshi Ren, and Pourang Irani. Detecting and leveraging finger orientation for interaction with direct-touch surfaces. In *Proceedings of the 22nd annual ACM symposium on User interface software and technology*, UIST ’09, pp. 23–32, New York, NY, USA, 2009. ACM.

[竹岡 10] 竹岡義樹, 味八木崇, 暦本純一. Z-touch: 指先姿勢インタラクション可能なマルチタッチシステム. WISS2010 論文集. 日本ソフトウェア科学会, 2010.

[内藤 09] 内藤真樹, 志築文太郎, 田中二郎. 赤外線方式タッチパネルにおける接触面積を利用した押し込み操作の基礎検討. 全国大会講演論文集, pp. 173–174. 情報処理学会, 2009.

[堀 10] 堀竜慈, 志築文太郎, 田中二郎. タッチパネル面に伝わる固体音の svm を用いた解析によるスポイト操作の実時間認識. 日本ソフトウェア科学会大会論文集. 日本ソフトウェア科学会, 2010.