平成 27 年度

筑波大学情報学群情報科学類

卒業研究論文

題目 入力インタフェース化した机での端末操作システムの開発

主専攻　ソフトウェアサイエンス主専攻

著者名　　富田 洋文

指導教員　　田中 二郎　高橋 伸　志築 文太郎

要　旨

机の上で多くの仕事を作業する人は多い. 例えば, 机の上で絵を描く, 設計図を見ながら物を作る, 参考書を見ながら問題を解くといったものがある. そのような場面の中で, PC やタブレット端末の画面に参考資料や図を表示しながら作業をすることがある. しかしディスプレイに表示してある参考資料や図を変更, 拡大などの操作をしたい時に, 机の上で作業をしながら画面の操作をするのは簡単ではない. そこで本研究では, 机の上で作業をするユーザが作業中でも PC やタブレット端末の操作を簡単に行える入力インタフェースシステムを考案する. PC やタブレット端末を操作するインタフェースはいくつかあるが, 我々は作業する手の認識による入力インタフェースを用いることを考える. 本研究では机自体を入力インタフェースにすることで目的を達成すると考えた. 本研究ではシステムを考案し, 机を入力インタフェースにするための装置の作成とその装置の予備実験, そして装置を用いた手の位置や動きなどの情報の取得を行った. 今後は装置の改善, 入力インタフェース化した机を用いた PC やタブレット端末を操作するアプリケーションの開発, そして本システムの評価実験を行う予定である.

目次

図目次

表目次

第1章　序論

1.1　机の上での作業

机の上で多くの仕事を作業する人は多い. 例えば, 机の上で絵を描く, 設計図を見ながら物を作る, 参考書を見ながら問題を解くといったものがある. そのような場面の中で, PCに接続しているディスプレイやタブレット端末の画面に参考資料や図を表示しながら作業をすることがある. 絵を描く場合ではディスプレイに参考にする風景の写真を映すことがあり, 電子回路での回路の配線を行う際にはんだごてを手で持ちながらディスプレイに映る配線図を見ながら作業することもある. このようにディスプレイを活用して机の作業を支援するケースは多い.

1.2　机の上で作業する際の問題点

ディスプレイに表示する参考資料や図の変更, 拡大といった操作をする際に, 作業中に画面の操作をするのは簡単ではない. 手にペンやはんだごてのような作業用の道具を手で持ちながらディスプレイに映る画面を操作することが難しいからである. 操作するのには入力インタフェースが必要となるが, 入力インタフェースは主にキーボードやマウスといったものがある. しかし, 絵を描くといった作業の際に直接使わないこれらの入力インタフェースは作業スペースから外される. そういった場合, 画面の操作をしたい時に作業する手を止めてキーボードやマウスまで手を移動させなければならない.

1.3　本研究の目的

本研究では, 机の上で作業をするユーザが作業中でも PC やタブレット端末の操作を簡単に行える入力インタフェースを用いたシステムを考案し, どのユーザがどの作業場所でも本システムを利用できることを目指す.

1.4　本論文の構成

次章以降の本論文の構成は以下の通りである. 第 2 章では本研究を行うにあたってどのような入力インタフェースを用いるのかを述べる. 第 3 章では、本研究の全体のシステムについて述べる. 第 4 章ではシステムの実装として装置の作成や得られるデータの処理について述べる. 第 5 章では本システムを用いたアプリケーション例について述べる. 第 6 章では関連研究について述べる. 第 7 章では本研究の結論を述べる.

第2章　本研究で扱う入力インタフェース

2.1　入力インタフェース

PC やタブレット端末を操作するインタフェースはいくつかあり, キーボードやマウス, 音声認識によるインタフェース, 目の動きをトラッキングし視線によるインタフェースなどがある[7].音声や視線によるインタフェースは手で作業しながらでも扱えるメリットがあるが, 端末周りの環境音やユーザごとの顔の形の違いなどによって音声や視線の認識が難しい場合がある.

そういったインタフェースがある中で我々は PC やタブレット端末を操作するために作業する手の認識による入力インタフェースを用いることにした. 手の認識によるインタフェースは作業する手を使うということがあるが, 操作するという面ではユーザに親しみやすくそして直感的である.

2.2　手を認識する入力インタフェース

机の上の手を検知するのに先行研究では様々な装置を用いて手の認識を行った.カメラを用いた画像処理による手の認識[1]や Microsoft Kinect v2 や Leap Motion を用いて手の位置認識やジェスチャ認識を行った研究もある[3].他にも机の中にタッチディスプレイを設置しタッチ認識を行ないその評価を行った研究[2]もある.また音響反射を機械学習させタッチ認識を行う研究もある[9].

2.3　本研究で用いる入力インタフェース

カメラや Microsoft Kinect v2 を用いた手の認識は,カメラを設置するだけで手を認識することが出来るが,部屋の照明や背景と手の色の関係で認識しづらい状況が生まれる.机での作業では手に手袋を着けたり道具を持ったりするので認識が難しくなるのではと考えられる. Leap Motion は机の上に設置して手の位置や手のジェスチャを認識するが,装置自体が作業スペースに入ったり,装置と手の間に障害物があると認識できなかったりする.したがって,これらの装置を用いた場合,センサに合わせた作業スペースを作ることになる.

本研究の目的に合う入力インタフェースを次のように考える.

・ユーザが作業するスペースに影響を与えない
・部屋の環境に依存しない
・手の状態に関係なく使える
・作業場所を変えても装置を設置しやすい

そこで本研究では目的に合う入力インタフェースとして机を考えた.

2.4　　机入力インタフェース

机自体をタッチ検出できる入力インタフェースにすることで机の周りの環境を変えることなく作業を行うことができ,作業中でも手を大きく動かす必要がなくなる.PCやタブレット端末の操作を行いたい場合は,机の上の手の位置を検出し,その位置データを用いてそれらの操作を行う.

今回は机の変更にも対応するため,手の検出が可能な机を作るのではなく既存の机に手を認識させる装置を付ける形式とする.そして装置を机の裏に設置し.机の上の手を認識する.これによりあたかも机が入力インタフェース化したように見える.

第3章　本研究のシステム設計

3.1　本研究のシステムの概要

本研究で行うシステムの流れは次のとおりである.システム図を図１に示す.

1. 机を入力インタフェース化する
2. 机の上に手をあてる
3. 手の位置を検出するためのセンサデータを取得
4. PC やタブレットの画面を操作

本論文では机を入力インタフェース化させる装置の作成とそれを用いた画面操作アプリケーションを中心に述べる.

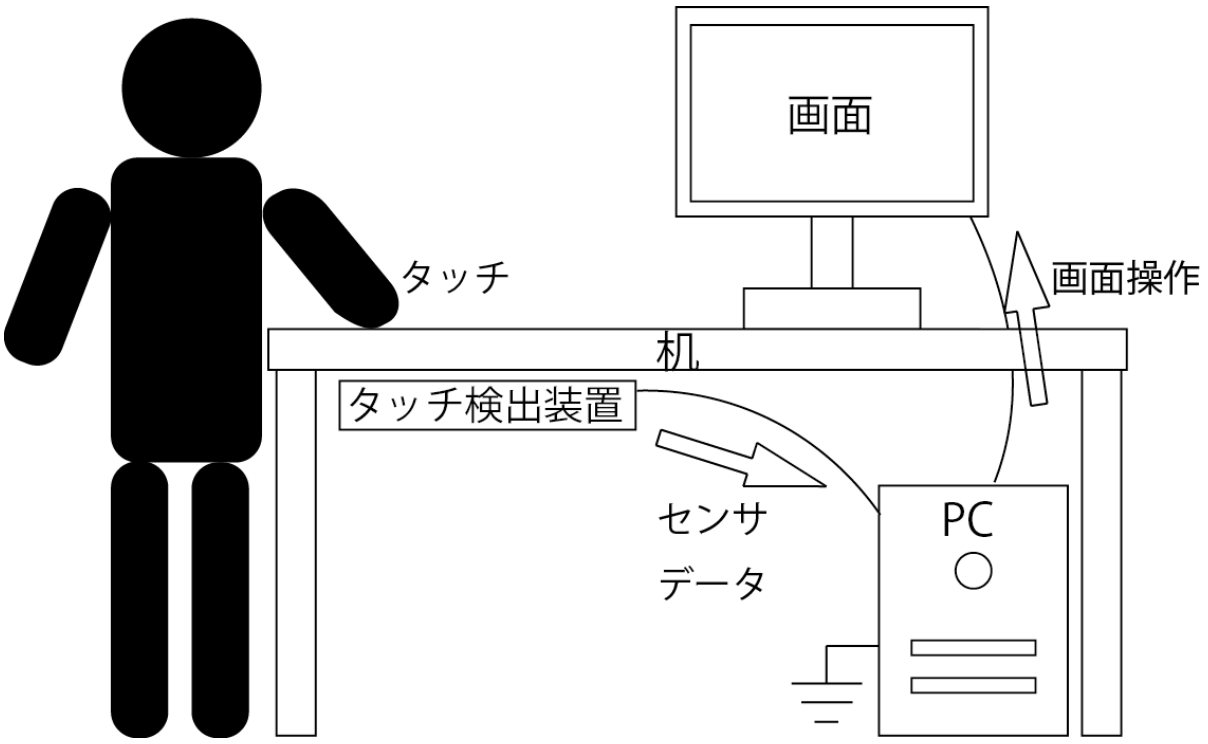


図 1 机にタッチしてから画面を操作するまでの流れ

3.2　机の入力インタフェース化

机の入力インタフェース化を行うのに机の上の手を認識するセンサを作成する.人間の持つ静電容量を利用したタッチセンサを用いて,手とセンサの間に机がある状態でもタッチ検出を可能にする.1 つのセンサによるタッチ検出だけでなく,複数のタッチセンサを用いる.そ

れぞれのセンサのデータを PC やタブレット端末といったホストコンピュータに送ることで,机にタッチしたかどうか,手の位置の推定,手の動いた方向などを計算し机の上の手に関する情報を得ることができる.また複数のタッチセンサやデータ送信回路を含んだ装置をテープ等で貼り付けるだけで設置が可能となるので机の変更に伴う再設置が簡易化し,ユーザへの装置の設置に対する負担がほとんどない.

3.3 画面操作アプリケーション

入力インタフェース化した机を用いたアプリケーションを開発する.例えば,絵や図をディスプレイに映し作業中の手を机の上で動かすだけで画像の切り替えを行うアプリケーションを考える.これはディスプレイに風景写真を表示しながら机の上のノートに絵を描く,または勉強するために参考資料や問題をディスプレイに表示しノートにそれらの解答を書くといった場面を想定する.これにより作業中の手のまま机の上でスワイプ動作をするだけで写真の切り替えや拡大表示が出来るアプリケーションができると思われる.

第4章　　手を認識する装置の実装

一般の机をタッチ可能な入力インタフェースとして使うためのタッチ検出装置を作成した(図2). この装置はタッチセンサとそれを固定するアクリル板, そしてPCとデータを送受信する回路で構成されている. この装置を机の裏に設置してホストコンピュータに接続することで装置のセンサ値をそのホストコンピュータに送信し, 端末側でタッチ判定や机の上の手の位置などの情報を得ることが出来る.

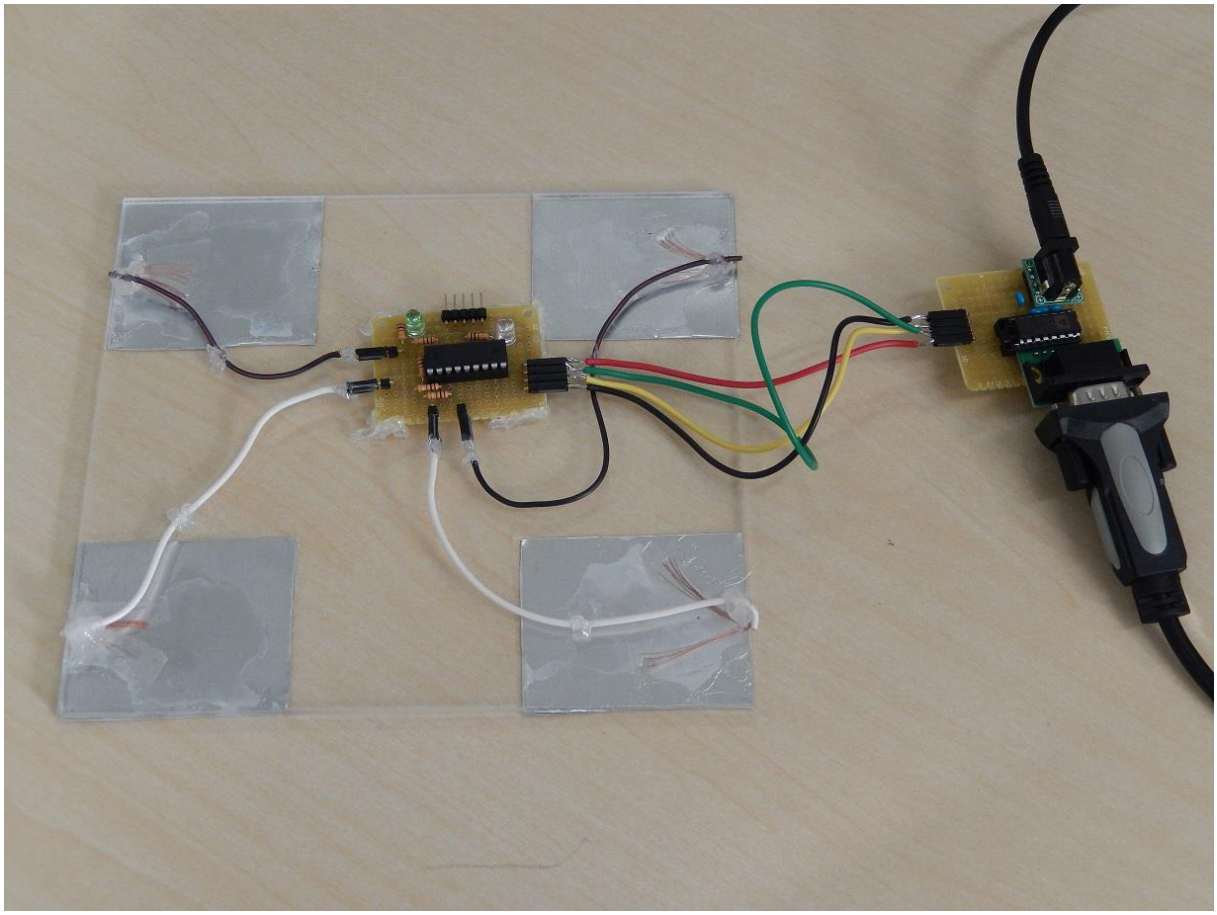


図2 タッチ検出装置

この章では机の上の手を認識する装置について述べる. タッチセンサの仕組みや回路の概要の説明, タッチセンサの予備実験の実施, センサ値を用いた机の上の情報(タッチ判定, タッチ位置推定, 手の動いた方向)を取得する方法を述べる.

4.1　　タッチセンサの仕組み

4.1.1　　RC 積分回路

電気回路において電源と抵抗とコンデンサが直列につながっている回路をRC積分回路という.これは入力電圧の時間積分を出力する回路である.回路図は図3のようになり電源を入れるとコンデンサの両端に電圧が発生するが,この電圧が上がるまたは下がる速さは抵抗値とコンデンサの静電容量値によって変化する.抵抗値または静電容量値を増やせば増やすほど,コンデンサの両端電圧値の上がるまたは下がる速さは遅くなる。波形の例を図4に示す.Voは入力電圧,Vは出力電圧である

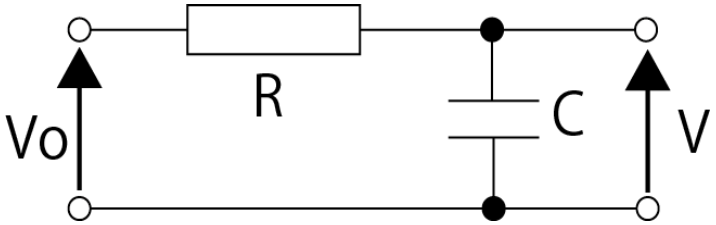


図3 RC積分回路の回路図

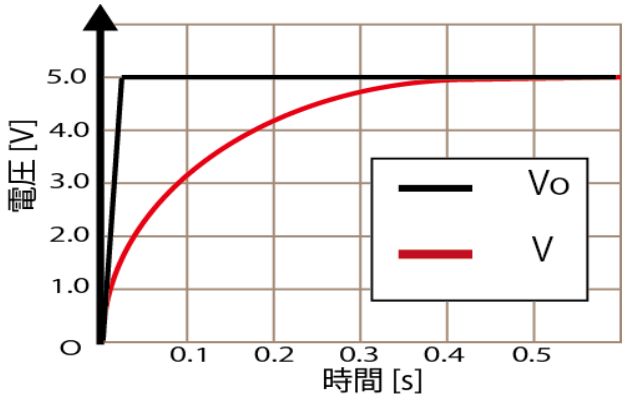


図4 RC積分回路の波形の例

抵抗値をR[Ω],コンデンサの静電容量値をC[F],時間をt[s],入力電圧をVo[V],コンデンサの両端電圧値をV[V]としたとき,この積分回路の式は次のようになる.

$$\mathrm{V} = V_o\left(1 - e^{-\frac{1}{RC}t}\right) \qquad (1)$$

この式においてV[V]が閾値電圧 V_T までになるまでの時間を求める式にすると

$$t = \ln\frac{V_o}{V_o - V_T}\, RC \qquad (2)$$

となる.ここで抵抗値が一定とすれば時間tとコンデンサの静電容量値C[F]は比例の関係となる.つまり,時間tを計測することでコンデンサの静電容量値C[F]の大きさが分かる.

4.1.2　　タッチ検出

4.1.1で述べた積分回路において，コンデンサの部分を人間が持つコンデンサに置き換えても同様な電気的特性が生まれる．抵抗に導電体をつけて，それに人間が触ることによって静電容量が発生し式(1)の波形が生まれる．逆に触らなければ静電容量が0に近づくので出力電圧と入力電圧がほぼ同じ波形になる．本研究で作成するタッチセンサはこの特性を利用し，導電体に手が触れているか触れていないかを判別する．今回は導電体としてアルミ材を，電圧の入出力制御や波形の測定するマイクロコンピュータとしてPIC16F1827を使用した．タッチセンサ回路の概略図を図5に示す．

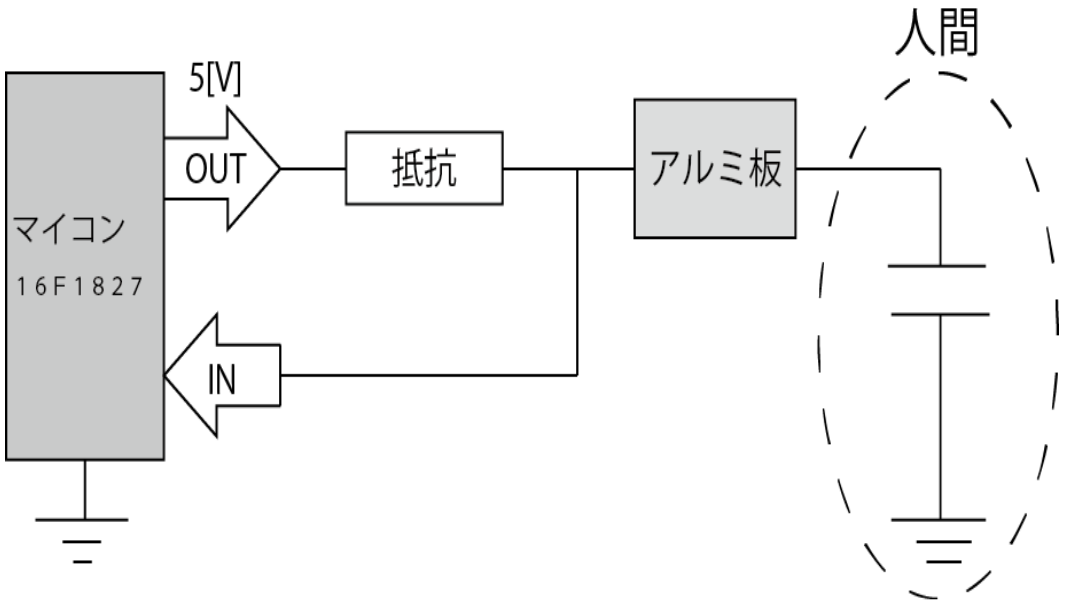


図5　タッチセンサ回路の概略図

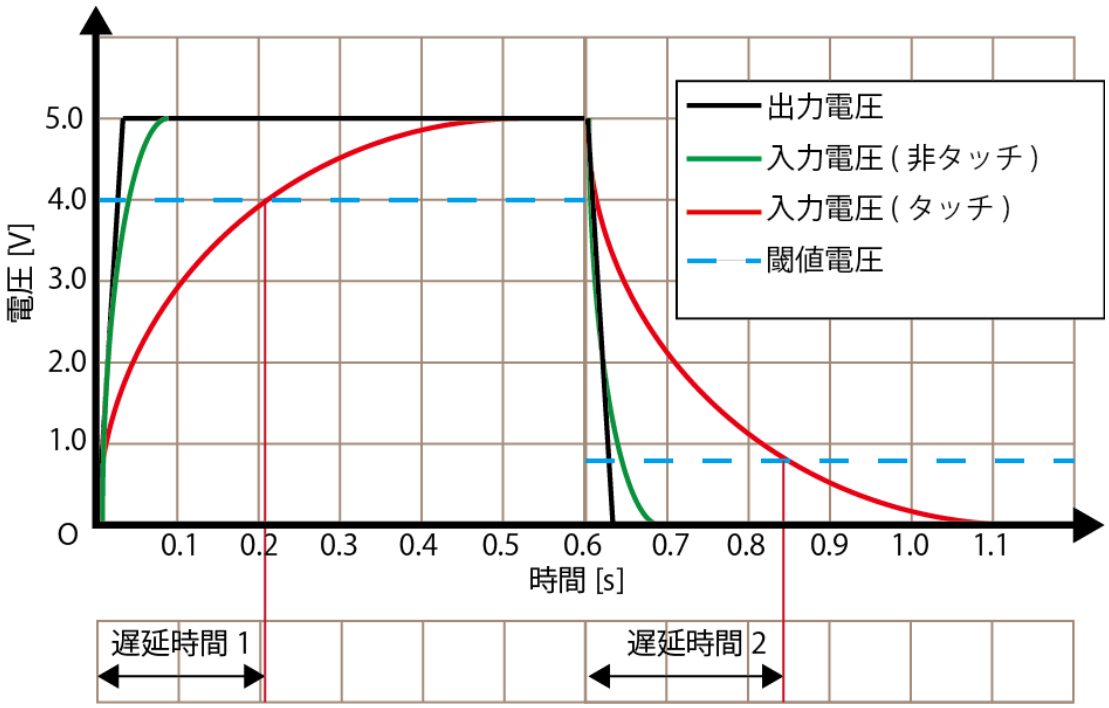


図6　タッチセンサ回路の波形の例

マイコンの出力ピンから電圧をかけたとき,アルミ材に手が触れているかいないかでマイコンの入力ピンの電圧の波形が変わる.波形の例を図6に示す.タッチすると入力側の電圧の上がるまたは下がる速さが遅くなり,遅延時間が発生する.

マイコンでこの遅延時間を測定し,それに応じてタッチしたかどうかを検出する.また,この波形の変化は,抵抗値,アルミ材の面積,アルミ材と手の間の材質と厚さで変わる.今回はアルミ材と手の間に机を挟むのでアルミ材からタッチする人間までの静電容量は次のように求まる. アルミ材の面積をA[m^2],アルミ材と手の間の厚さをd[m],机の誘電率を ε としたときの机の静電容量C_dは次のようになる.

$$C_d = \varepsilon \frac{A}{d} \qquad (3)$$

導電体と手の間に机が挟まっているとすると、(3)で求めた机の静電容量と人間が持つ静電容量は直列につながっていると考えられる.人間が持つ静電容量をC_hとすると合成した静電容量Cは次のようになる.

$$\mathrm{C} = \frac{C_h\, C_d}{C_h + C_d} \qquad (4)$$

(2),(3),(4)式で机の厚さ,机の誘電率,人間の静電容量は不変であると考える.抵抗値,アルミ材の面積といったパラメータを変えることで,アルミ材と手の間に机を挟めた状態でも手を検出できる.したがってタッチセンサのアルミ材を机の裏に貼り付けることで,机の上の手の検出を可能とする.

4.2 遅延時間の測定

遅延時間の測定はマイクロコンピュータを用いて行う.そのマイクロコンピュータ内で遅延測定用の変数を用意し,出力電圧の立ち上がりから入力電圧の立ち上がり,または出力電圧の立ち下がりから入力電圧の立ち下がりまでの間を用意した変数でカウントしていき,このカウントした数を静電容量の大きさとしてホストコンピュータにセンサ値としてデータを送る.今回は時間を直接計測するのではなく,時間の代わりにこのカウント数を遅延時間として表す.カウント数が多ければ多いほど遅延時間は長くなる.しかし実際にカウント数を測定すると,毎回出力されるカウント数に安定性がなく,ばらついた測定結果となってしまう.そのためマイクロコンピュータの中でカウント数の平均値を出して安定した値を出力する.具体的には,複数のカウント数を格納する場所を用意し,値を1つずつ次の格納場所にずらしていきながら、得られたカウント数を1番目の格納場所に入れていく.そして順次格納してあるすべてのカウント数の平均値を計算し出力する.本研究ではこのカウント数を格納するデータの数を40個とした.装置がセンサ値を取得するまでのフローチャートを図7に示す.フローチャートにあるdataはカウント数を格納する変数である.

今回,遅延時間を測定するのに PIC16F1827 を用いた.このマイクロコンピュータの I/0 ピンに図 5 の回路概要図のように抵抗の両端にそれぞれ接続しマイクロコンピュータ内でカウント数を計測する.このマイクロコンピュータの I/0 ピンの立ち上がり電圧は 2 ボルトで立下り電圧は 0.8 ボルトである.またこの I/0 ピンにはセンサ値であるカウント数をホストコンピュータへ送信するための RS232C インタフェース IC と接続してある.今回使用した

RS232C インタフェース IC として ADM3202AN を用いた.

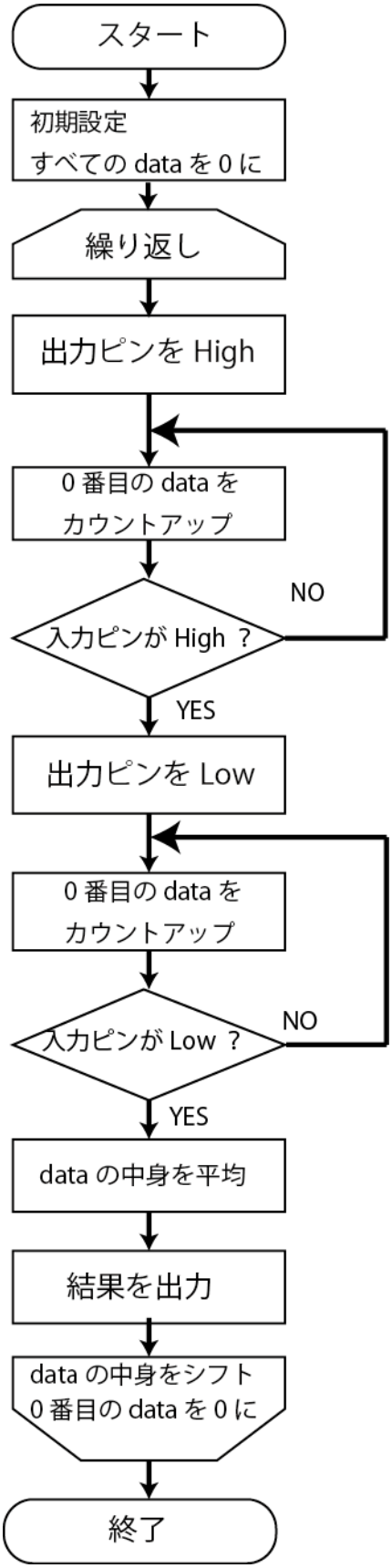


図7　装置がカウント数を取得するまでのフローチャート

4.3　　タッチセンサの予備実験

タッチセンサについて調査するために3つの実験を行った.

(a)タッチした時としていない時の遅延時間カウント数

作成したタッチセンサを設置して実際に机の上をタッチすることで遅延時間がどのように変化したかを調査した.以下にタッチセンサにパラメータを示す.

アルミ材の面積 ： 50[mm]×60[mm]
抵抗値　　： 1.0[MΩ]
机の材質　　： 木材
机の厚さ　　： 30[mm]

タッチした時とタッチしていない時の平均カウント数を表1に示す.

表1 タッチした時とタッチしていない時の平均カウント数

状態	カウント数
非タッチ	9.2
タッチ	13.4

結果,机の上をタッチする前後で値に大きな違いが発生したので,このセンサを用いることでタッチしたかしていないかの判定は可能と言える.

(b) 机の上の手の位置を変えてタッチするとカウント数はどうなるか

作成したタッチセンサを設置して,机の手の位置を移動するとどのような変化が起きるかを調査した.センサは実験(a)と同じものを使用した.手の位置に対してのカウント数を表2に示す.縦軸はタッチしているかいないか、横軸はセンサの中心から手までの距離(D)である.

表2　手の位置に対しての平均カウント数

	D=0[mm]	D=25[mm]	D=50[mm]	D=75[mm]	D=100[mm]
カウント数	13.4	12.2	10.9	9.9	9.6

結果,センサから離れれば離れるほど、ほぼ一定の割合で遅延時間が少なくなった.今回は1つのセンサで行ったが複数のタッチセンサを用いることで机の上の手の位置を推定できるのではと思われる.

(c) センサを4つ設置するとどうなるか

実験(a)で使用したセンサを4つ用意し,それらを60[mm]ずつ離して図8のように設置した.それぞれのセンサを1つのマイコンの入出力ピンに接続し，設置したアルミ材に対応する机の上の部分をタッチした時の遅延時間を計測する.タッチする位置は各アルミ板に対応する4点と装置全体から見て中央の位置の1点で計測した.

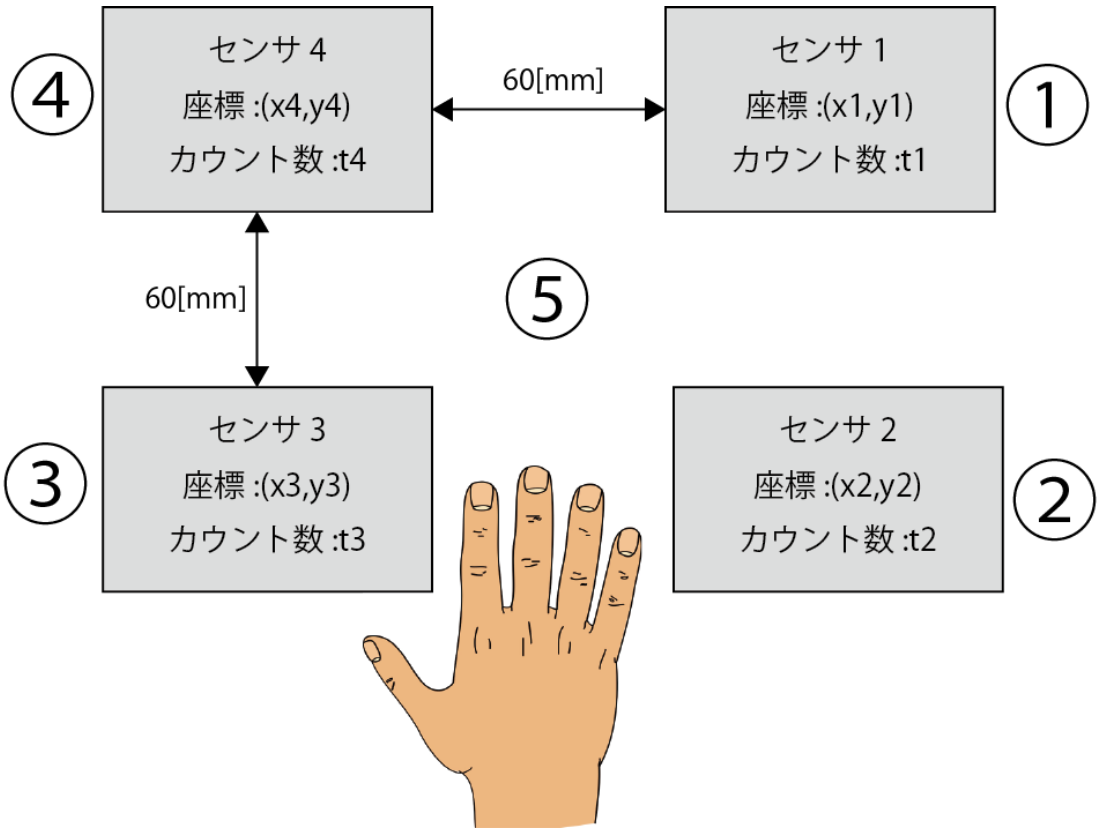


図8　実験(c)のタッチ位置

表3 それぞれの位置に対しての4つのセンサの平均カウント数

	非タッチ時	①	②	③	④	⑤
アルミ板1	9.2	13.3	12.2	10.0	10.8	12.4
アルミ板2	11.4	14.0	16.4	13.3	13.0	15.6
アルミ板3	10.8	11.5	12.0	14.8	12.5	13.0
アルミ板4	13.1	13.8	13.8	15.4	16.2	15.5

結果, 各タッチ点でそれぞれ違うカウント数が得られた. また同じ場所を何度もタッチしても4つのセンサ値は同じような値を出した. これにより4つのセンサ値を用いて手の位置を推定できると思われる.
しかし, タッチしていない時のそれぞれのアルミ板は違う値を出力している. これは同じセンサを使用していてもピンごとの内部回路の個体差によりセンサごとで得るカウント数が同じにならなかったからである.

4.4　センサ値から得られる情報

ここでは装置から送られるセンサ値を用いてホストコンピュータが何を処理するのかを述べる. 前節で述べたようにピンごとのカウント数が違うため, 初めにホストコンピュータ側でセンサ値であるカウント数を正規化し, その後にタッチ判定やタッチ位置推定, スワイプ方向の推定を行う. この 4.4 節ではセンサ値であるカウント数の取得から手の位置やスワイプ方

向の推定までの各処理の詳細について述べる. これらの処理はホストコンピュータ側で処理する. 初めにホストコンピュータ側のフローチャートを図 9 に示す.

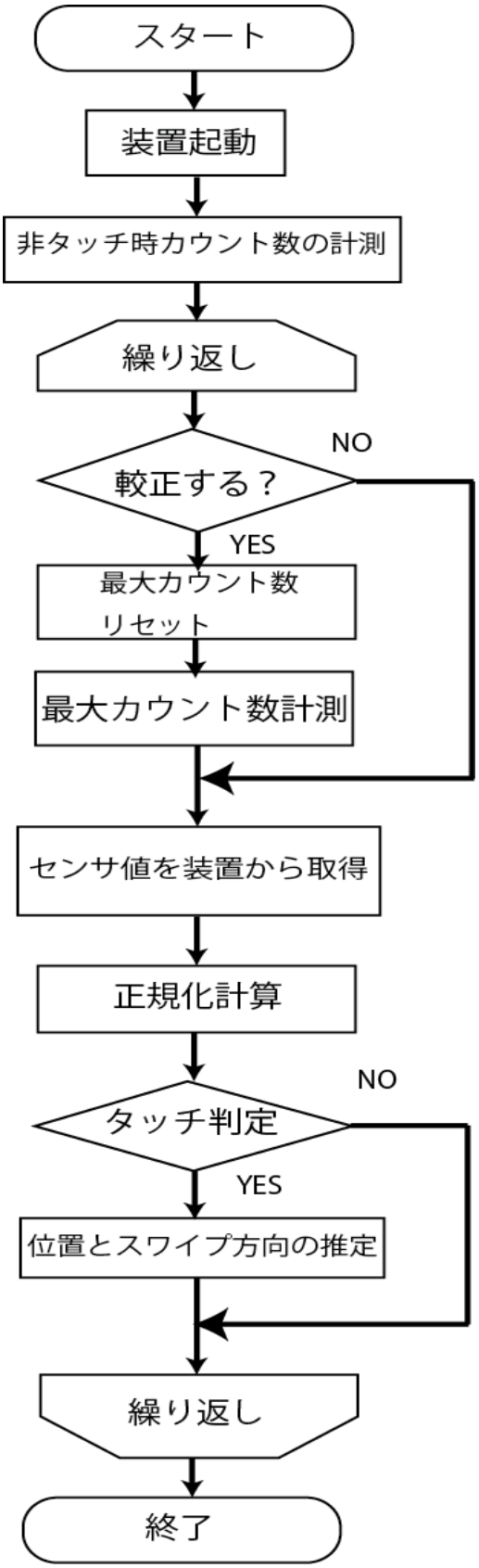


図 9　装置を起動してから手に関する情報を取得するまでのフローチャート

4.4.1 各センサ値の正規化

ピンごとのカウント数の違いが生じるため, これを正規化する. 今回正規化する方法として, 非タッチ時のカウント数とタッチ時の最大カウント数の差をとり, その差をその時のカウント数に割ってセンサごとの割合を計算する. 計算式を次の式(5)に示す. n番目のセンサの非タッチ時のカウント数を$t_{l,n}$, タッチ時の最大カウント数を$t_{h,n}$, その時に計測されたカウント数をt_n, 正規化されたカウント数をp_nとする. (n=1, 2, 3, 4)

$$p_n = \frac{t_n}{t_{h,n} - t_{l,n}} \times 100 \tag{5}$$

正規化されたカウント数を用いることでタッチした位置の推定やスワイプ方向の推定が容易になる.

4.4.2 較正

装置を起動するごとに非タッチ時のカウント数$t_{l,n}$とタッチ時の最大カウント数$t_{h,n}$を較正しなければならない. またユーザを変更した場合でもこれらの値が違うので, ユーザの変更があった場合にもこれらのカウント数を較正しなければならない. しかし較正は簡単に行うことができる. 装置起動時に非タッチ時のカウント数$t_{l,n}$を取得し, その後に机の上に手を置き動かすだけで最大カウント数$t_{h,n}$を取得することでこの問題は解決できる. ユーザの変更があった場合にはその度に最大カウント数$t_{h,n}$をリセットし再度机の上に手を置いて動かしてもらうことで解決する. 机の変更がある場合でも装置を変更する机の裏に設置し同じような処理をすることで較正することができる.

4.4.3 タッチ判定

タッチ位置の推定の前に, タッチ判定の基準をここで定義する. 正規化されたカウント数が変化したときをタッチ判定としてしまうと, タッチをしていないつもりでも手が机に近づくだけでセンサはタッチしたという誤反応を起こしてしまう. そのためそれぞれのセンサで得た正規化されたカウント数の平均がある閾値を越える場合のみをタッチしたということにした.

さらに, 今回はタッチ判定に「タッチしていない」「タップ」「タッチ」の3つの状態を付けた. これは後に述べるタッチの誤作動防止に利用される. これらの状態の閾値を次の表にまとめた. 手の位置などの情報はこの「タッチ」の状態の場合のみで得ることにする.

表4 3つのタッチ状態を判定するための閾値

4つのセンサの平均正規カウント	タッチの状態判定
0～17.5	タッチしていない
17.5～30	タップ
30～	タッチ

4.4.4 タッチ位置推定

ここでは計測した4つのセンサ値を用いて手の位置を推定する. これは4. 4. 3でタッチ判定が「タッチ」の状態だけに限定し, それ以外は手の座標が無いとする. 手の位置を求める方法として、4つのセンサの正規化された平均カウント数を用いてそれらの重心計算を用いることで求める. 手の位置推定座標を(hx. hy)とすると, 座標を求める式は以下のようになる.

$$hx = \frac{x_1 \times t_1 + x_2 \times t_2 + x_3 \times t_3 + x_4 \times t_4}{t_1 + t_2 + t_3 + t_4} \quad (6)$$

$$hy = \frac{y_1 \times t_1 + y_2 \times t_2 + y_3 \times t_3 + y_4 \times t_4}{t_1 + t_2 + t_3 + t_4} \quad (7)$$

位置を推定するのにセンサ1からセンサ4の位置座標を設定しなければならない. 座標全体の大きさを500×400にし, センサ1からセンサ4までの座標を下のように設定した.

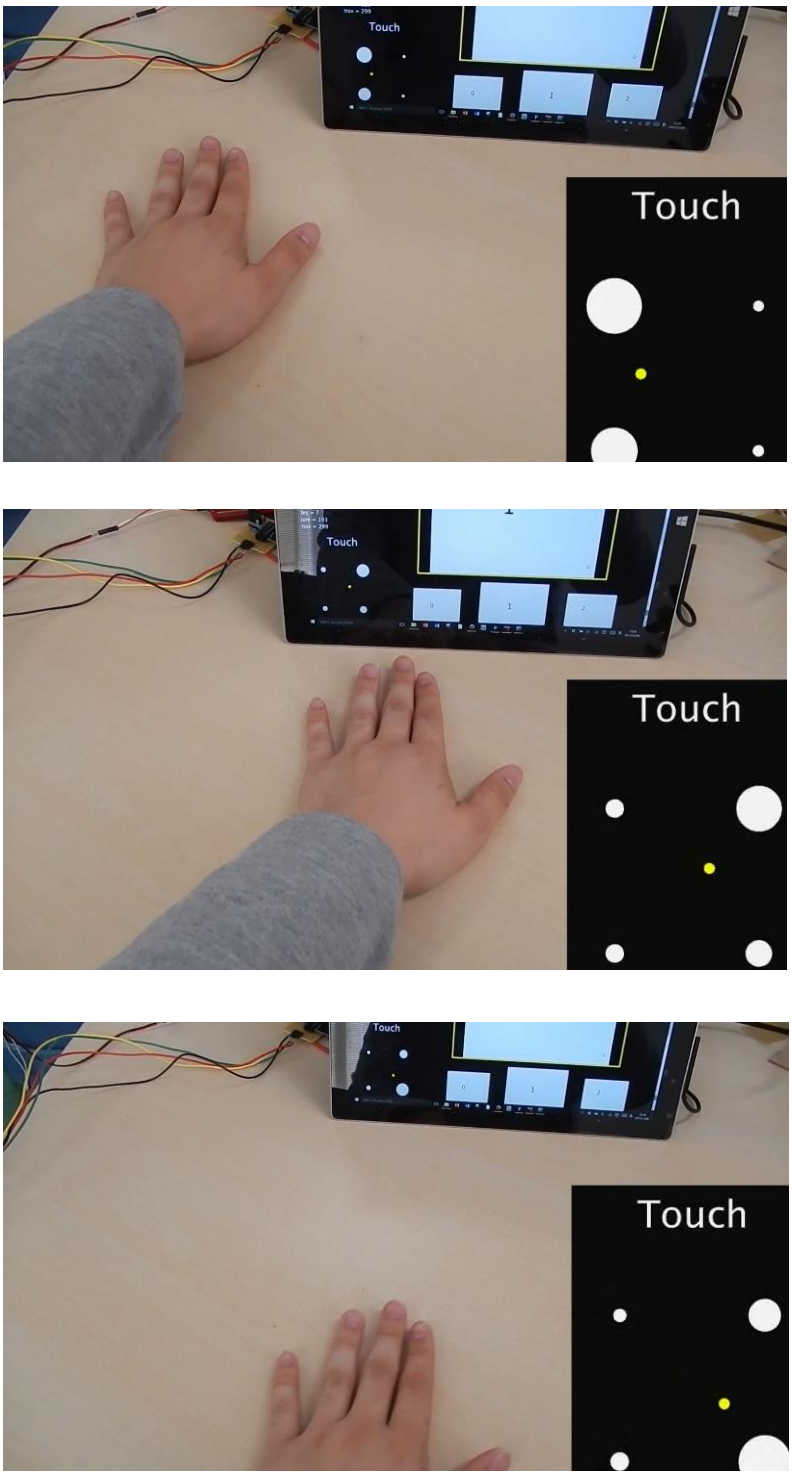
センサ1： $(x_1, y_1) = (150,100)$
センサ2： $(x_2, y_2) = (400,350)$
センサ3： $(x_3, y_3) = (150,350)$
センサ4： $(x_4, y_4) = (400,100)$

これらの数値と4. 4. 1で正規化した4つのセンサ値を式(6)と式(7)に代入することで手の座標位置(hx. hy)を得る.

ここで作成した装置と位置推定の計算式を使って手の位置を推定できるかの実験を行った. 実験の様子を図10に示す. 使用した机は4. 3でセンサの予備実験で使用した机と同じである. 図10の白い円は各センサで得られた正規化されたカウント数の大きさを表す. センサごとで正規化されたカウント数が多いほど白い円の面積が大きくなる. これによりどのセンサがどれだけ反応しているのかが分かる. また黄色い点は上記の計算式によって計算された手の推定位置を示している.

実験の結果, おおよその手の位置の特定を行うことができ, 手を動かすと動かした通りに黄色い点が動いた. しかし手を机に触れた瞬間と離した瞬間に推定位置がずれてしまうことがあった. また, センサの真上に手を置いた場合や装置の外側に手を置いた場合では他のセンサが少しだけ反応していることがあり本来の手の位置とずれている結果もあった. このことから上記の式では装置の場所に限った手の位置の推定が出来るといえる.

手の位置を推定する実験の他に, 机と手の間にノートを挟めて同様の実験を行ったところ, 挟める前と変わらなく手の位置を推定することができた.



：カウント数

：推定座標(hx.hy)

図10 カウント数の重心計算を用いたタッチ位置推定実験の様子

4.4.5 スワイプ判定

位置データを用いて手のスワイプ方向を検出する. 今回はタッチの開始地点と終了地点の 2 点の差を用いてスワイプ方向の判定を行う. タッチ開始地点を(x, y)とし, 終了地点を(x’, y’)とする. 2 点間の x 軸の差と y 軸の差をそれぞれ dx, dy とするとこれらは次の式で表される.

$$dx = x' - x \tag{8}$$

$$dy = y' - y \tag{9}$$

この式で計算された dx と dy の符号と量でスワイプ方向を決める. 今回は上下左右の 4 方向スワイプ方向を求める. スワイプ方向を決める数値を表 5 に示す.

表 5　スワイプ方向の判定をするための閾値

	dx	dy
上スワイプ	-39~39	30 以上
下スワイプ	-39~39	-30 以下
右スワイプ	40 以上	-29~29
左スワイプ	-40 以下	-29~29

ここの数値の範囲外であればスワイプの判定を無しにした. 今回はタッチが開始してからタッチが終了するまでの距離を見るので, 手が机から離れてからのスワイプ判定となる.

実装についての記述は以上である. 最後に本システムでの装置とホストコンピュータ周りのブロック図を図 11 に示す.

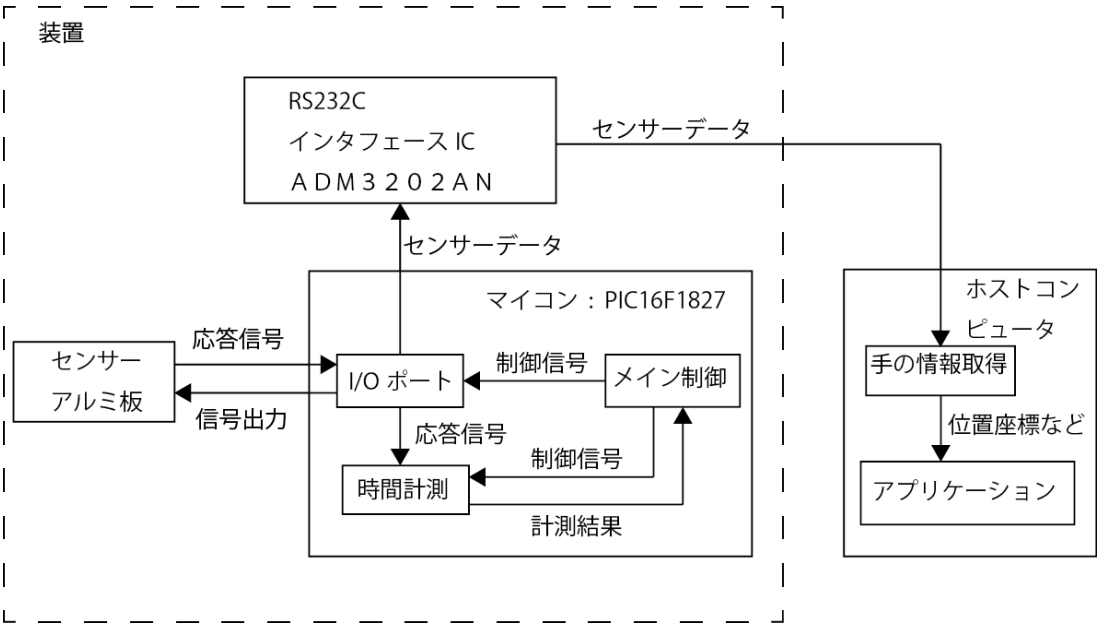


図 11 装置のセンサからホストコンピュータのアプリケーションまでのブロック図

第5章　　手を認識する装置の実装

5.1　　アプリケーション概要

作成した装置を利用して,机の上で作業(例：お絵かき)をしながらでもディスプレイに映る画像の切り替えをするデモを行った.デモ用のソフトウェアはProcessingで作成した.ソフトウェアの画面を図12に示す.

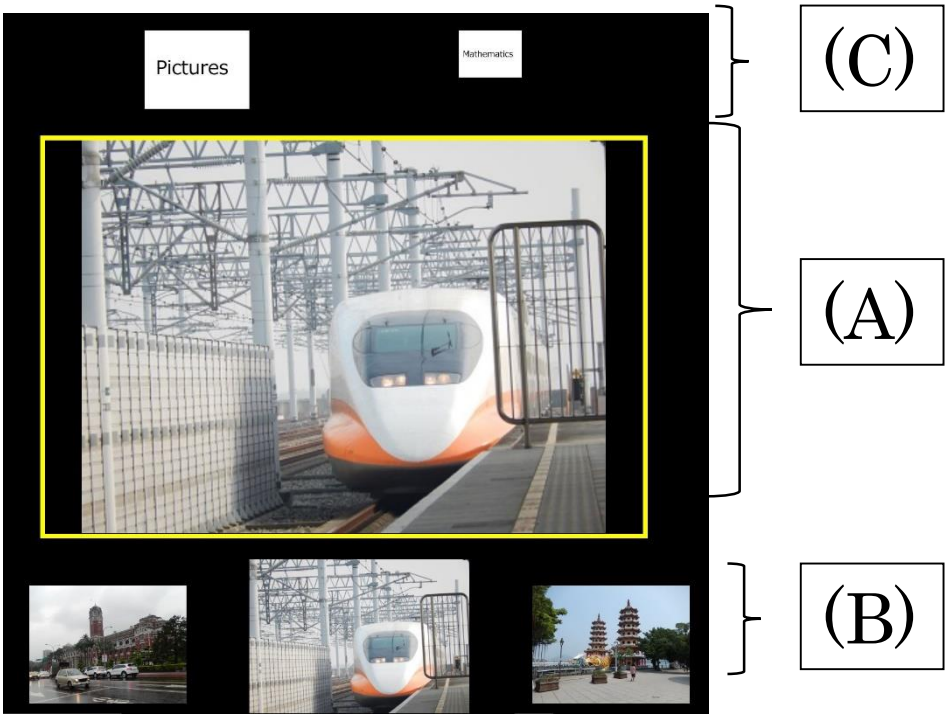


図12 画像切り替えアプリケーションの画面

図12の(A)は今見ている画像を表示する.(B)は現在表示している画像とその前後の画像を表示する.(C)は見ている画像のフォルダを表示する.これらは手の動きに応じて画像の切り替えや画像フォルダの切り替えを行う.例えば,机の上で手を左に動かすと(B)の左側の画像に切り替わる.手を上に動かすと画像フォルダが切り替わる.

また,誤作動防止をするために操作可能トリガを導入した.これは操作をする場合でもないのに手が机に触れて誤って画像の切り替えが行われないようにするためである.この操作可能トリガをオンにすることで画像またはフォルダの切り替えを行える.一通り操作を終えた後に操作可能トリガをオフにすることで誤動作による間違った画像の切り替えを防ぐ.このソフトウェアではそのトリガのオンとオフの切り替えを2回連続のタップで行う.

5.2　　操作方法

1.　装置の取り付け

　ユーザが作業する机の裏に装置を設置する.設置はテープで装置を固定した.固定した後は装置に電源を供給するACアダプタとホストコンピュータと接続するRS232ケーブルを取り付ける.

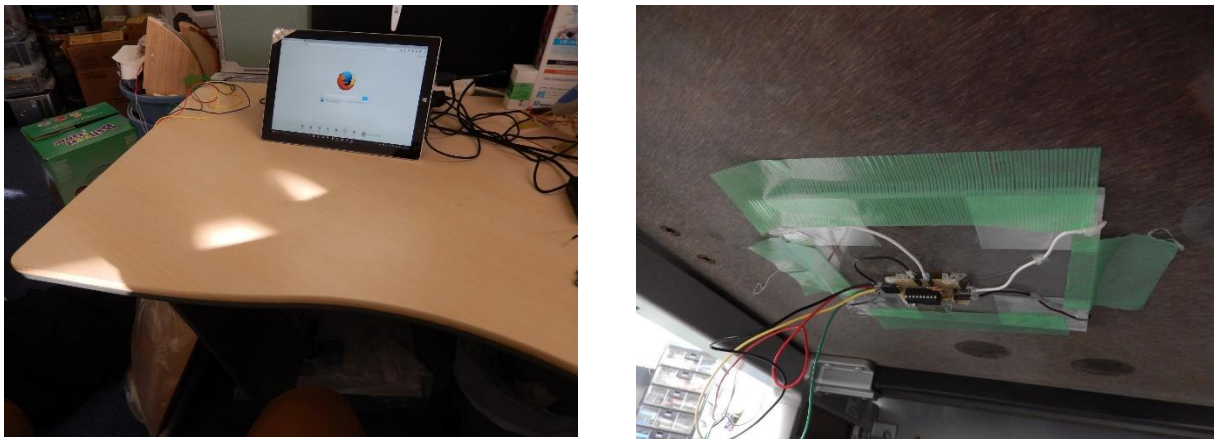


図13 装置を取り付けた時の机の表と裏

2.　専用ソフトウェアと装置を起動

　画像切り替えソフトウェアと装置を起動し,非タッチ時のセンサ値を較正する.

3.　較正

　較正を行うために,今回はソフトウェア上の画面をマウスでクリックまたは画面を直接タッチすることで較正を開始し,その後に机の上を手で触れることでタッチ時のセンサ値を取得し較正が完了する.

4.　操作可能トリガをオン

　机の上を手で軽く2回タッチしソフトウェアが２度のタップ判定が出力されると画像やフォルダの切り替えが可能となる.

5.　画像切り替え

　ソフトウェアを起動すると図12の画面が出て,表示画像モード(図12のA)とフォルダモード(図12のC)があり,黄色い枠で現在どのモードが選択されているのかが分かる.表示画像モードで机の上で手を左右にスワイプ動作をすることで画像の切り替えが行える.フォルダモードで同様に手を左右にスワイプ動作することでフォルダの切り替えを行う.モードの切り替えは上下スワイプすることで行える.そしてモードの切り替えを行うことで黄色い枠の位置が変わる.

6.　操作可能トリガをオフ

　画面操作が終わった際に２度のタップを行うことでトリガがオフになる.

　以下4.から6.を繰り返す.

実際にデモを行ったところ机の上で手を動かすだけで画像を切り替えることができた. 机の上にノートを置き, そのノートに片方の手で絵や文字を書きながらもう片方の手で画像の切り替えを行うこともできた. 途中, ソフトウェアがスワイプ動作の判定を間違えることがあった. 操作を行わない時でも手や腕に装置は反応したが, 大きな動きが無い限り誤作動は起きなかった.

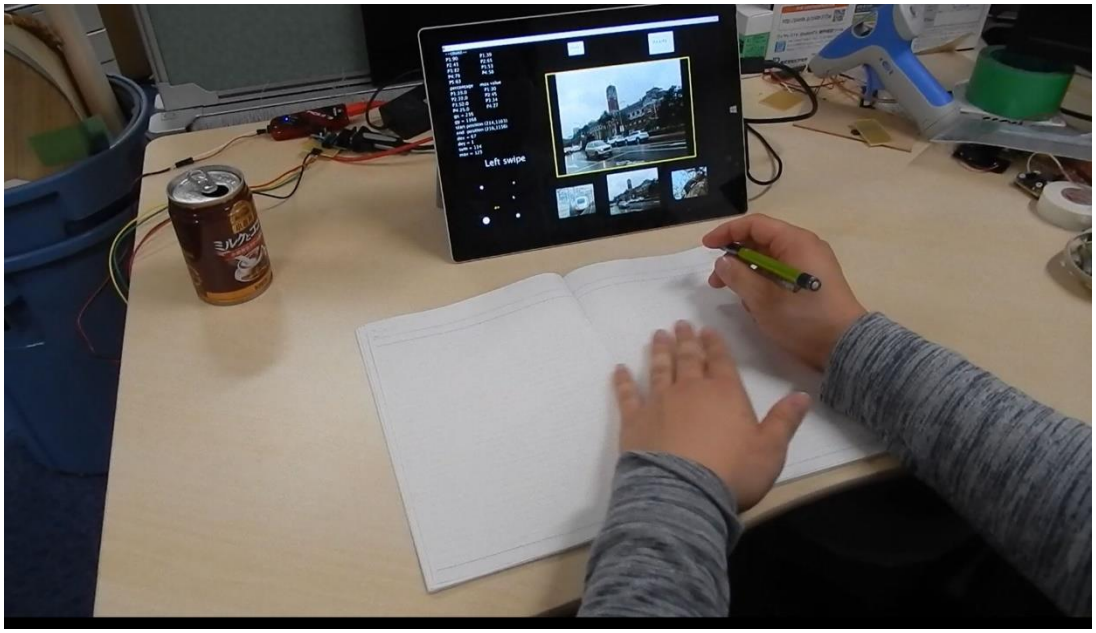


図 14　画像切り替えアプリケーションのデモの様子

この応用例の他にプロジェクタで机に直接映してソフトウェアの中身が同様のデモを行った.

図 15　プロジェクタを用いた本アプリケーションの利用例

第6章　関連研究

机の上での手を用いたインタラクションの研究はある. Wellner はプロジェクタで映像を机に映し, カメラを用いて机の上にある手を認識することで机の上でのインタラクション操作を可能にした[1]. Xiaojun らは,机の中にタッチディスプレイを設置し, デスクトップ PC 上でのタスクの処理を支援するための適度なウィンドウの位置やタッチ操作の調査をする研究を行った[2]. Raphael らは、机とディスプレイが一体化した装置を作成し, 装置全体でのタッチ検出やディスプレイの表現方法についての研究を行った[6]. Ilya らは, 手を検出するために Microsoft Kinect v2 と Leap Motion を使用し机の上の手の位置やジェスチャ認識を行なった[3].

これらの研究ではカメラやタッチディスプレイ, 手を検出する装置を使用しているが, 部屋の環境や机自体の構造に依存してしまう. また, 机を変更する際に周辺の装置の再設置に時間がかかってしまう.

今回、我々は静電容量を用いた机のインタラクションについて述べた。静電容量を用いた机の上でのインタラクションや手の認識の研究はある。

Tobias らは静電容量を用いた NFC によるユビキタスコンピューティングの研究を行った[4]. これは静電容量を計測するためにセンサの受信機を机の裏に, 送信機を机の上の任意の物に取り付けている. これによりこの研究では机の上に計測用の送信装置が必要となる. 我々の研究では操作するユーザ自身には電源を必要としない.他にも静電容量センサを複数用いて手の位置やマルチタッチを認識する研究も行った[5]. この研究は本論文と近い部分があるが, 検出する装置の大きさやコストという面では, 本研究の方ではシングルタッチではあるが片手で持てるサイズであり, かつお金のコストも低く用意する電子部品も手に入れやすい.

第7章　結論

本研究では机を入力インタフェースとして扱えるための装置を作成し机の上で作業をしていても専用のアプリケーションを使用することにより簡単な画像切り替えソフトウェアを利用できた.装置は人間がもつ静電容量を利用し机の上のタッチ検出を可能にした.さらに複数のセンサを用いることでタッチ位置と手の動いた向きを検出することもできた.装置を用いたアプリケーションとしてディスプレイに映る画像を机の上の手の動きで切り替えるアプリケーションを開発した.今回は画像切り替えアプリケーションだけであったが,本システムを利用することで,机の上での作業をしている中でもコンピュータまたはタブレット端末の簡単な操作が可能になった.

今後は, 今回作成した装置やアプリケーションの改良, PC やタブレット端末の操作, それらの評価実験を行う予定である. 装置の改良としては机の変更をする際に机の種類によって手を認識できない場合がある. 解決案としてはセンサ用の回路の抵抗値の変更やアルミ部分の位置や面積の変更することにより, 変更した机に合うセンサになることを考えている. また手の操作に関してタッチの始まりと終わりの位置だけでなくその間のストロークによる入力手法も考えている. さらに今後, 複数の人数での評価実験を行っていく予定である.

謝辞

本研究を行う当たり,指導教員である田中二郎教授,志築文太郎准教授,高橋伸准教授,および嵯峨智先生には研究全体にわたって丁寧なご指導とご助言をいただきまして,心から感謝申し上げます.また Nerf チームの皆様をはじめ,インタラクティブ・プログラミング研究室の皆さまにはゼミはもちろん日頃の研究生活や大学生活を通じて多くのご意見やご助言をいただきました.心から感謝いたします.

最後に,自分の生活を支えてくださった両親,日常生活でお世話になった友人,本研究をご支援くださった皆様に心から感謝申し上げます.

参考文献

[1] Pierre Wellner. The DigitalDesk calculator: tangible manipulation on a desk top display. In Proceedings of the 4th annual ACM symposium on User interface software and technology (UIST '91). ACM, New York, NY, USA, 27-33. 1991.

[2] Xiaojun Bi, Tovi Grossman, Justin Matejka, and George Fitzmaurice. Magic desk: bringing multi-touch surfaces into desktop work. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '11). ACM, New York, NY, USA, 2511-2520. 2011.

[3] Ilya Efanov and Joel Lanir. Augmenting Indirect Multi-Touch Interaction with 3D Hand Contours and Skeletons. In Proceedings of the 33rd Annual ACM Conference Extended Abstracts on Human Factors in Computing Systems (CHI EA '15). ACM, New York, NY, USA, 989-994.2015.

[4] Tobias Grosse-Puppendahl, Sebastian Herber, Raphael Wimmer, Frank Englert, Sebastian Beck, Julian von Wilmsdorff, Reiner Wichert, and Arjan Kuijper. Capacitive near-field communication for ubiquitous interaction and perception. In Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14). ACM, New York, NY, USA, 231-242. 2014.

[5] Tobias Grosse-Puppendahl, Andreas Braun, Felix Kamieth, and Arjan Kuijper. Swiss-cheese extended: an object recognition method for ubiquitous interfaces based on capacitive proximity sensing. In Proceedings of the SIGCHI Conference on Human Factors in Computing Systems (CHI '13). ACM, New York, NY, USA, 1401-1410. 2013.

[6] Malte Weiss, Simon Voelker, Christine Sutter, and Jan Borchers. BendDesk: dragging across the curve. In ACM International Conference on Interactive Tabletops and Surfaces (ITS '10). ACM, New York, NY, USA, 1-10. 2010.

[7] Yanxia Zhang, Jörg Müller, Ming Ki Chong, Andreas Bulling, and Hans Gellersen. GazeHorizon: enabling passers-by to interact with public displays by gaze. In Proceedings of the 2014 ACM International Joint Conference on Pervasive and Ubiquitous Computing (UbiComp '14). ACM, New York, NY, USA, 559-563. 2014.

[8] Ivan Poupyrev, Chris Harrison, and Munehiko Sato. Touché: touch and gesture sensing for the real world. In Proceedings of the 2012 ACM Conference on Ubiquitous Computing (UbiComp '12). ACM, New York, NY, USA, 536-536. 2012.

[9] Makoto Ono, Buntarou Shizuki, and Jiro Tanaka. A rapid prototyping toolkit for touch sensitive objects using active acoustic sensing. In Proceedings of the adjunct publication of the 27th annual ACM symposium on User interface software and technology (UIST'14 Adjunct). ACM, New York, NY, USA, 35-36. 2014.