令和4年度卒業研究報告書

心拍センサーを用いたアート作品の改良と機能拡張

情報技術科 古屋　勇翔

昆野　捷冴

指導教員 菅野　研一

目次

[第 1 章 はじめに 3](#_Toc128736438)

[第 2 章 研究概要 4](#_Toc128736439)

[2.1 作品概要 4](#_Toc128736440)

[2.2 システム構成 5](#_Toc128736441)

[第 3 章 開発環境 6](#_Toc128736442)

[3.1 Arduino-IDE 6](#_Toc128736443)

[3.2 Pure Data 6](#_Toc128736444)

[第 4 章 通信 7](#_Toc128736445)

[4.1 Wi-Fi 7](#_Toc128736446)

[4.2 UDP 7](#_Toc128736447)

[4.3 OSC 8](#_Toc128736448)

[第 5 章 ハードウェア 9](#_Toc128736449)

[5.1 ESPr Developer 32 9](#_Toc128736450)

[5.2 LED 11](#_Toc128736451)

[5.3 心拍センサー(脈波センサー) 12](#_Toc128736452)

[5.4 ロジックレベルコンバーター 13](#_Toc128736453)

[5.5 ステレオミニプラグケーブル 14](#_Toc128736454)

[5.6 コネクタ・基板 15](#_Toc128736455)

[第 6 章 ソフトウェア 17](#_Toc128736456)

[6.1 脈波について 17](#_Toc128736457)

[6.1.1 脈波の取得 17](#_Toc128736458)

[6.1.2 値の平滑化 18](#_Toc128736459)

[6.1.3 閾値の設定 19](#_Toc128736460)

[6.1.4 波の検出 20](#_Toc128736461)

[6.1.5 データの保存 21](#_Toc128736462)

[6.1.6 類似度判定 21](#_Toc128736463)

[6.2 LED処理について 25](#_Toc128736464)

[6.2.1 Adafurit\_NeoPixelライブラリ 25](#_Toc128736465)

[6.2.2 光の種類について 25](#_Toc128736466)

[6.2.3 色の処理について 26](#_Toc128736467)

[6.3 Pure Data(以下Pd) 28](#_Toc128736468)

[6.3.1 ライブラリ 28](#_Toc128736469)

[6.3.2 変数 29](#_Toc128736470)

[6.3.3 Wi-FiとUDP接続 30](#_Toc128736471)

[6.3.4 OSC通信 31](#_Toc128736472)

[6.3.5 機能 32](#_Toc128736473)

[6.3.6 主なオブジェクト 32](#_Toc128736474)

[6.3.7 インレット・アウトレット 32](#_Toc128736475)

[6.3.8 パッチの作成方法 33](#_Toc128736476)

[6.3.9 受信と音の出力 36](#_Toc128736477)

[6.3.10 音の種類について 41](#_Toc128736478)

[第 7 章 終わりに 41](#_Toc128736479)

[参考文献 42](#_Toc128736480)

# はじめに

過去の卒業研究で「入力デバイスに水を用いて音を生成するアート作品」[1],「入力デバイスに水を用いて音を出力・制御する楽器の製作」[2],「入力デバイスに水を用いて音と光を出力するシステムの製作」[3]がというものあり,「水と向き合う」というアート作品としての出展歴がある. [4] [5]以前に産技短展で「水と向き合う」に実際に触れ面白いと感じた.見るだけではなく実際に体験できる展示品の方が,面白味があると思ったため,実際に触れて体験できるアート作品を作成したいと考えた.

昨年の卒業研究で「心拍信号を利用したアート作品の制作」[6]というものがあった.展示をするには改善が必要だと聞いた.そこで今回私たちは,改善をし,展示ができるように作成に取り組むことにした.

# 研究概要

## 作品概要

２台の脈波センサーから読み取った値を計算し,類似度を算出する.算出された類似度によって光と音の演出が変わる.全体図は次のようになっている.

LEDはアクリル円筒に貼り付け,PCで音の出力を行っている.

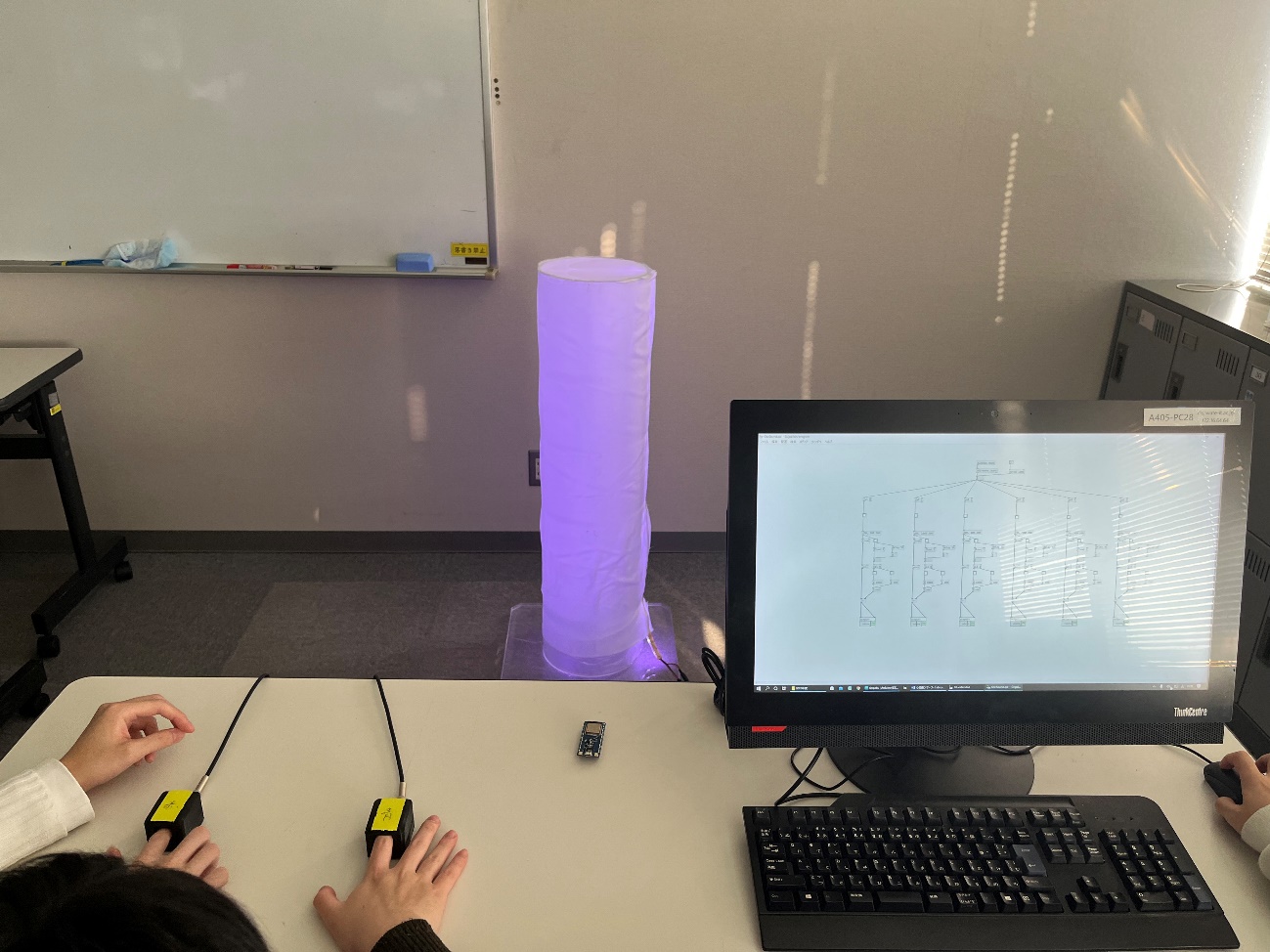


図 　全体図

## システム構成

　主に使用しているものは心拍センサー, ESP32, LED, アクリル円筒, PC,である.

心拍センサーで脈波を取得し,ESP32で値の処理, LEDの制御, PCとの無線通信を行っている.

　PCは音の出力を行うため使用している.



図 2　システム構成図

# 開発環境

## Arduino-IDE

Arduino-IDEはボード上で動作するソフトウェアを開発するために作られた統合開発環境である.ソースコードの編集や,コンパイル,リンク,作成されたオブジェクトのボードへの書込み機能などが提供されている.

## Pure Data

Pure Dataとは,音声,ビデオ,映像処理のためのリアルタイムなグラフィカルプログラミング環境である.フリーソフトであり,どのプラットフォームでも動作し,シンセサイザ,スペクトル解析,三次元映像処理・生成などができる.

Arduinoと組み合わせた自作楽器の製作も可能である.

# 通信

## Wi-Fi

　Wi-Fi(Wireless Fidelity)とは,電波を用いた無線通信により近くにある機器間を相互に接続し,構内ネットワーク（LAN）を構築する技術である.

今回は,ESP32とPC間で通信をするためにWi-Fiを使用している.

## UDP

UDP(User Datagram Protocol)とは, インターネット・プロトコル(IP)を使ったネットワークにおいて,アプリケーション同士が最小限の仕組みでデータを送受信できるトランスポート層のプロトコルである.TCPに比べてプロトコルが簡素なため,信頼性には劣るが,通信の処理にかかるコストが少ないという性質から,リアルタイム性が重要視される通信で使われている

## OSC

OSC (Open Sound Control)とは,電子楽器（特にシンセサイザー）やコンピュータなどの機器において音楽演奏データをネットワーク経由でリアルタイムに共有するためのアプリケーション層のプロトコルである.OSCはTCPかUDPプロトコルを使用して通信される.一般的にはUDPを使用することが多いため,本研究でもUDPを使用する. 　OSCメッセージ（OSC Message）とOSC引数（OSC Argument）の2つに分けられる．OSCメッセージは，送受信するOSCの情報内容をラベリングしたものでOSCの値が何を意味しているのかを表す．メッセージは階層構造を持つことが可能で，階層を「/」で区切って表現する．OSC引数は実際の値を送信する．値は，整数，実数，文字列等様々な型を送受信することが可能である．また，複数の値を一度に送受信することもできる． 　本研究では,心拍数の類似度をデータとして送受信するために使用している. (図 3)

**/temperature 29.0**

**OSC Message OSC Arguments**

図 3　OSCプロトコルの仕様

# ハードウェア

## ESPr Developer 32

ESPr Developer 32はEspressif Systemsの無線通信モジュール（Wi-Fi + Bluetooth）ESP-WROOM-32とUSB-シリアル変換IC FT231XSを搭載した開発ボードである．その他にも3.3 V出力レギュレータやリセットスイッチ，動作モード切替用スイッチが搭載されているので，ESP-WROOM-32 Wi-Fi + Bluetoothモジュールをすぐに使用可能．ESPr Developer 32の仕様は表，ESP-WROOM-32の仕様は表の通りである．

表 1　ESPr Developer 32

|  |  |
| --- | --- |
| 電源 | USB Type-C 5V  電源入力ピン 3.7V~6.0V  電源出力ピン USBまたはVinピンに入力した電圧を出力 |
| 動作電圧 | 3.3V |

表 2 　ESP-WROOM-32

|  |  |
| --- | --- |
| Wi-Fi | 802.11 b/g/n（HT40） |
| Bluetooth | Classic，BLE 4.2（デュアルモード） |
| SRAM | 520KB |
| フラッシュ | 4MB(32Mbit) |
| 動作電圧 | 2.2～3.6V |

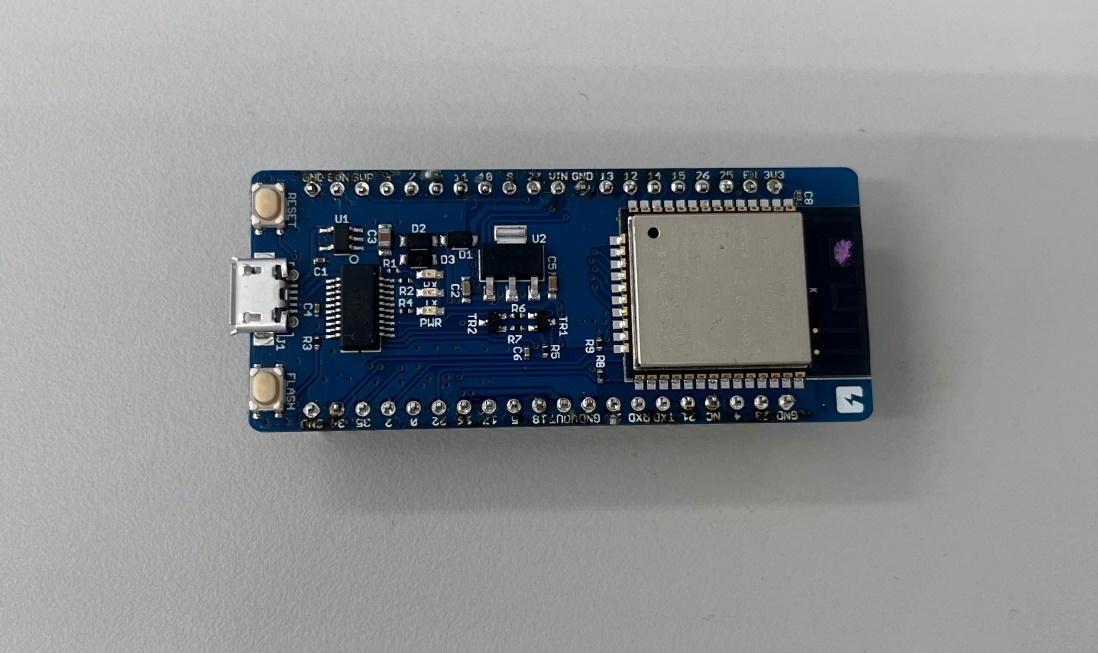


図 4 ESPr Developer 32

## LED

LEDはNeo Pixelを使用した.従来からあるRGBフルカラーLEDは,色は変化はするが,LEDテープライト全体が同時に変化する.

Neo PixelはLED1個ずつにIC（マイコン）チップが搭載されているためLEDの色や明るさを細かくコントロールすることが可能である.今回の研究では画像のように虹色に光らせる演出を取り入れたいためNeo Pixelを使用した.(図 5)



図 5 Neo Pixel

## 心拍センサー(脈波センサー)

指先などの末梢血管のある部分を光センサー部に置いて脈波をアナログ出力するセンサーデバイスである.(図 6)脈波とは心臓が血液を送り出すことに伴い発生する,血管の容積変化を波形としてとらえたものである.(図 7)



図 6　　 心拍センサー(脈波センサー)

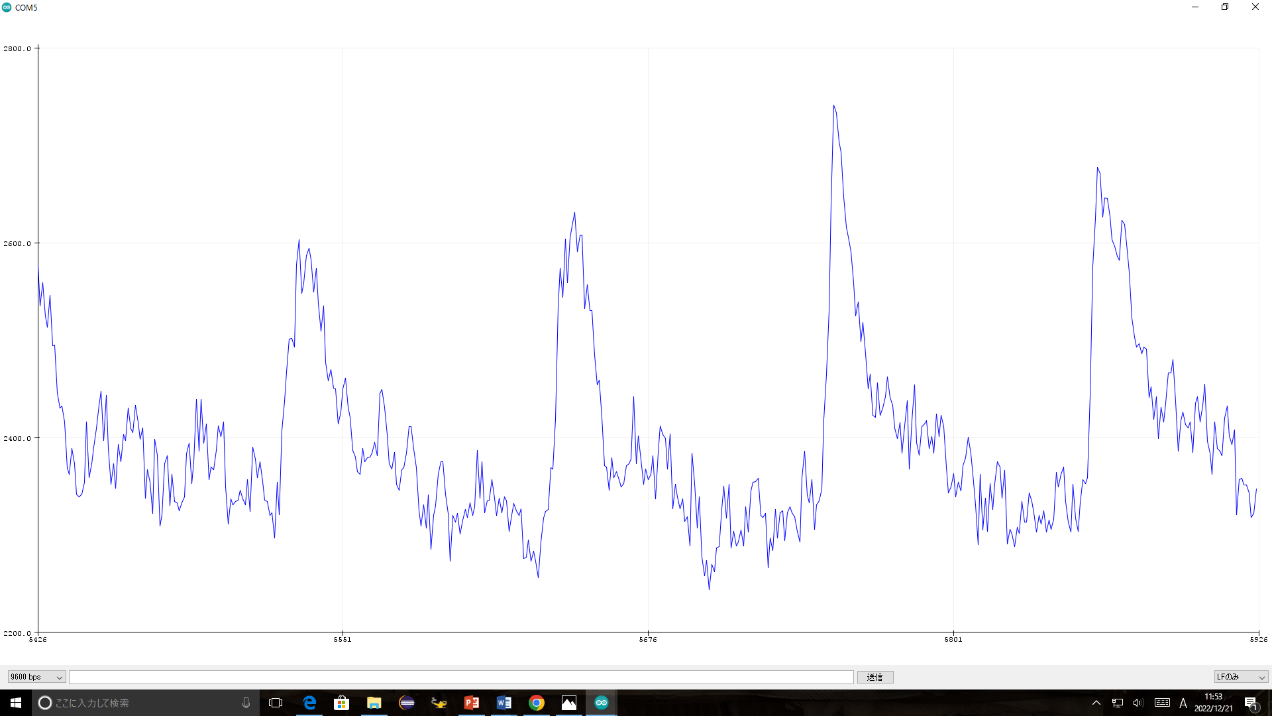


図 7 　脈波

## ロジックレベルコンバーター

昨年度の研究の課題として安定して光らせることができないというものがあった.原因の一つとして,ESP32 とNeo Pixelを直接つないでしまったことにあると考えた.その理由として,ESP32は3.3Vで稼働するのに対して,Neo Pixelは5.0Vで稼働するため直接つないでしまうと稼働することもあるが,制御の安定性に欠けていた.そこで,今回の研究では3.3Vから5.0Vへ変換するロジックレベルコンバーターを使用することにした.(図 8)(図 9)

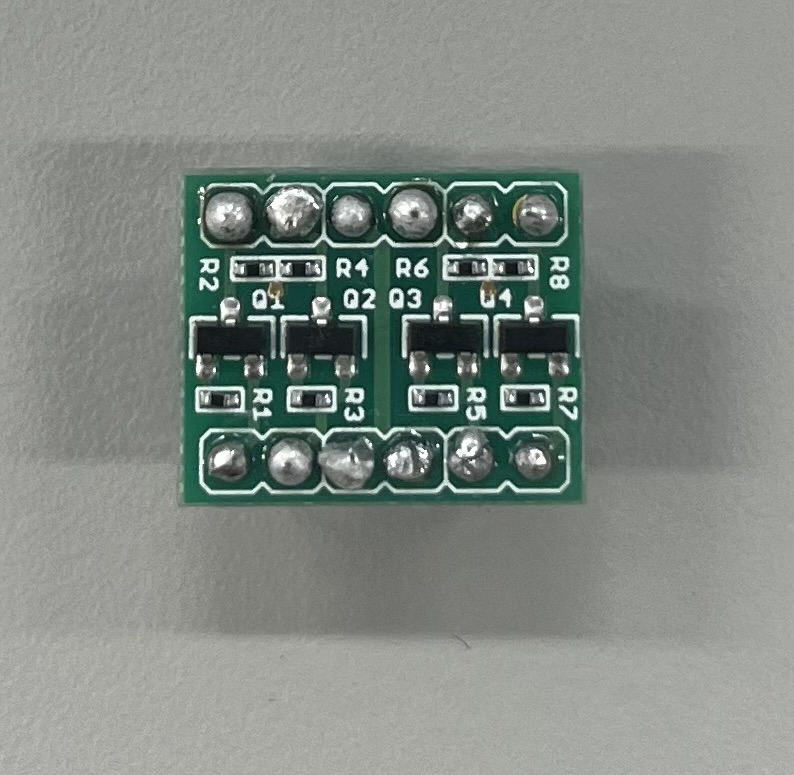


図 8　ロジックレベルコンバーター

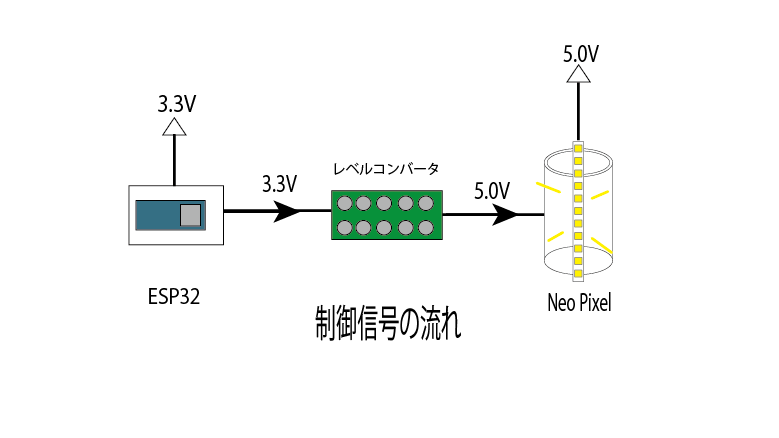


図 9　制御信号の流れ

## ステレオミニプラグケーブル

心拍センサーに電源供給するためにステレオミニプラグケーブルを使用する.(図 10)しかし一部断線していて値を取得しない問題が発生していたため修復した.(図 11)



図 10 ステレオミニプラグケーブル

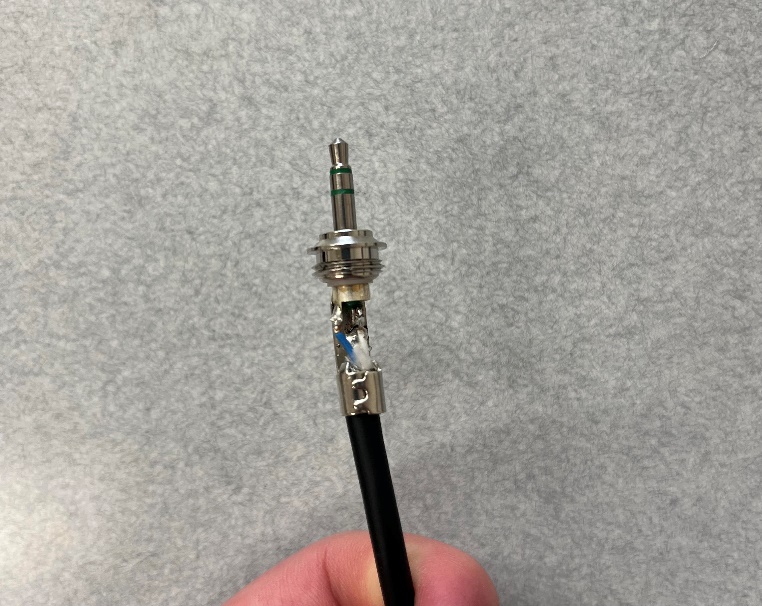


図 11　修復後のステレオミニプラグケーブル

## コネクタ・基板

展示の準備をしやすくするためにコネクタの改良と基板の作成を行った.(図 12)(図 13)(図 14)

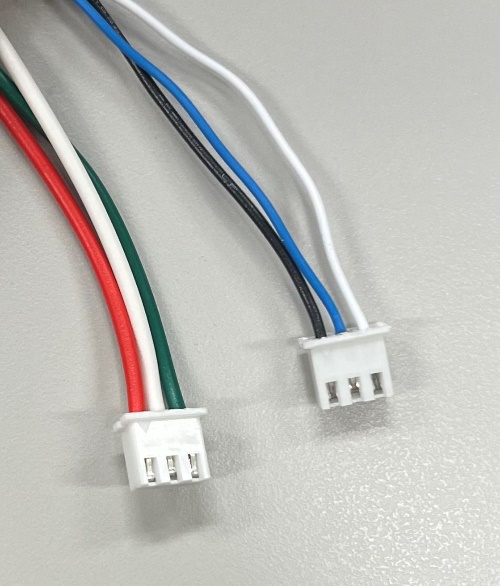


図 12　コネクタ

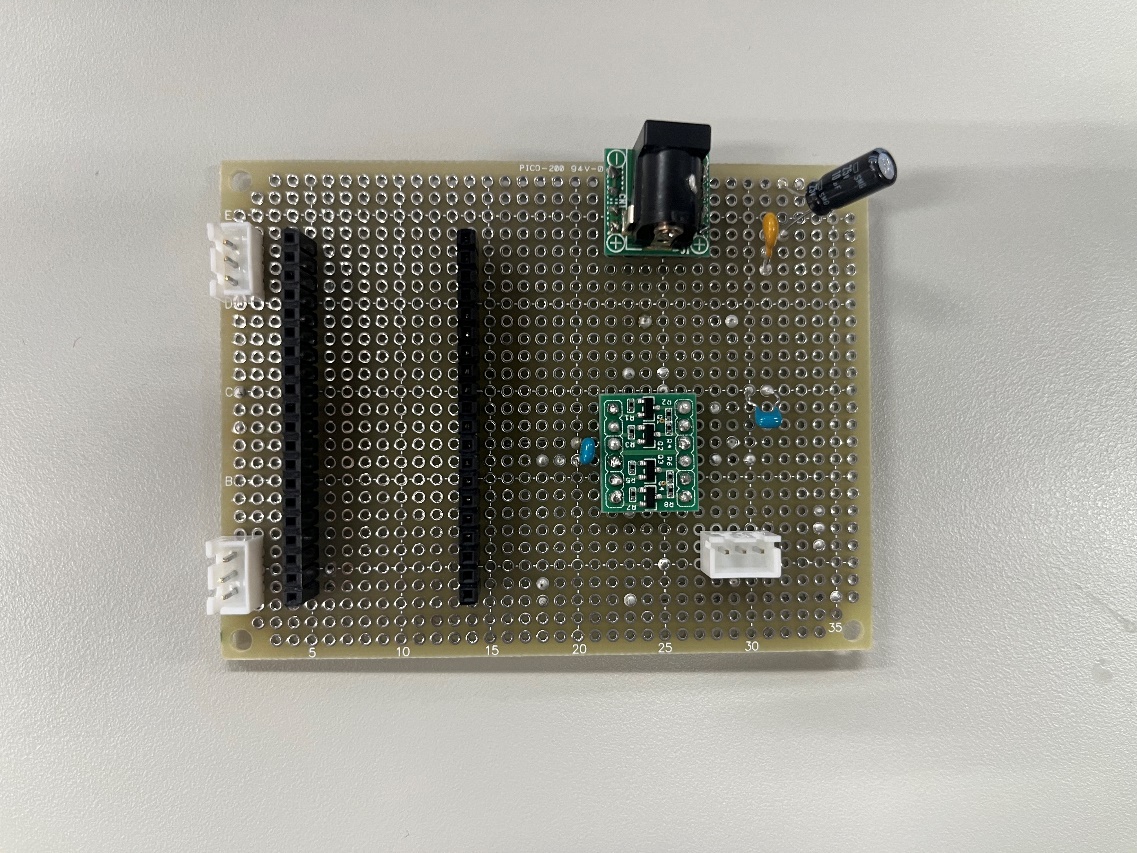


図 13　基板(表)

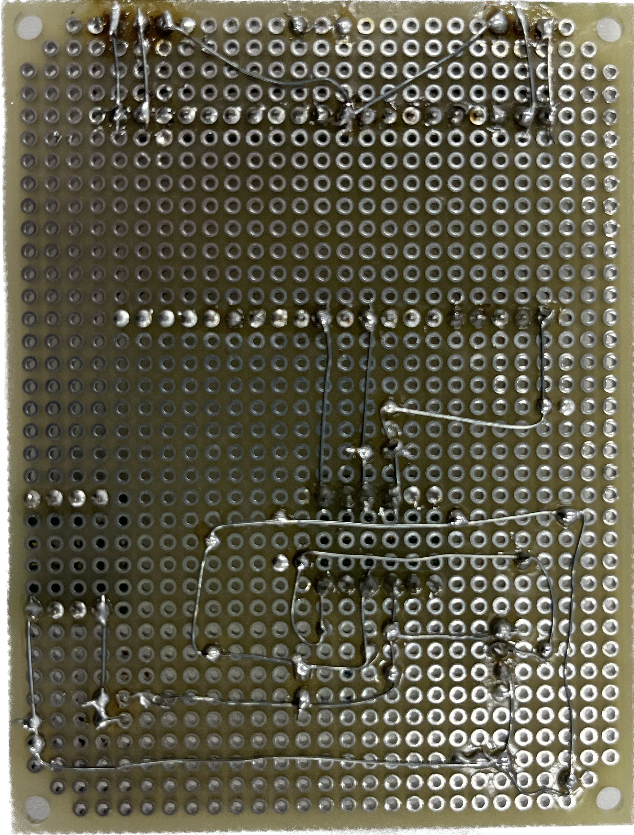


図 14　基板(裏)

今回作成した基板の部品と回路図は下の図のとおりである.(図 15)(図 16)

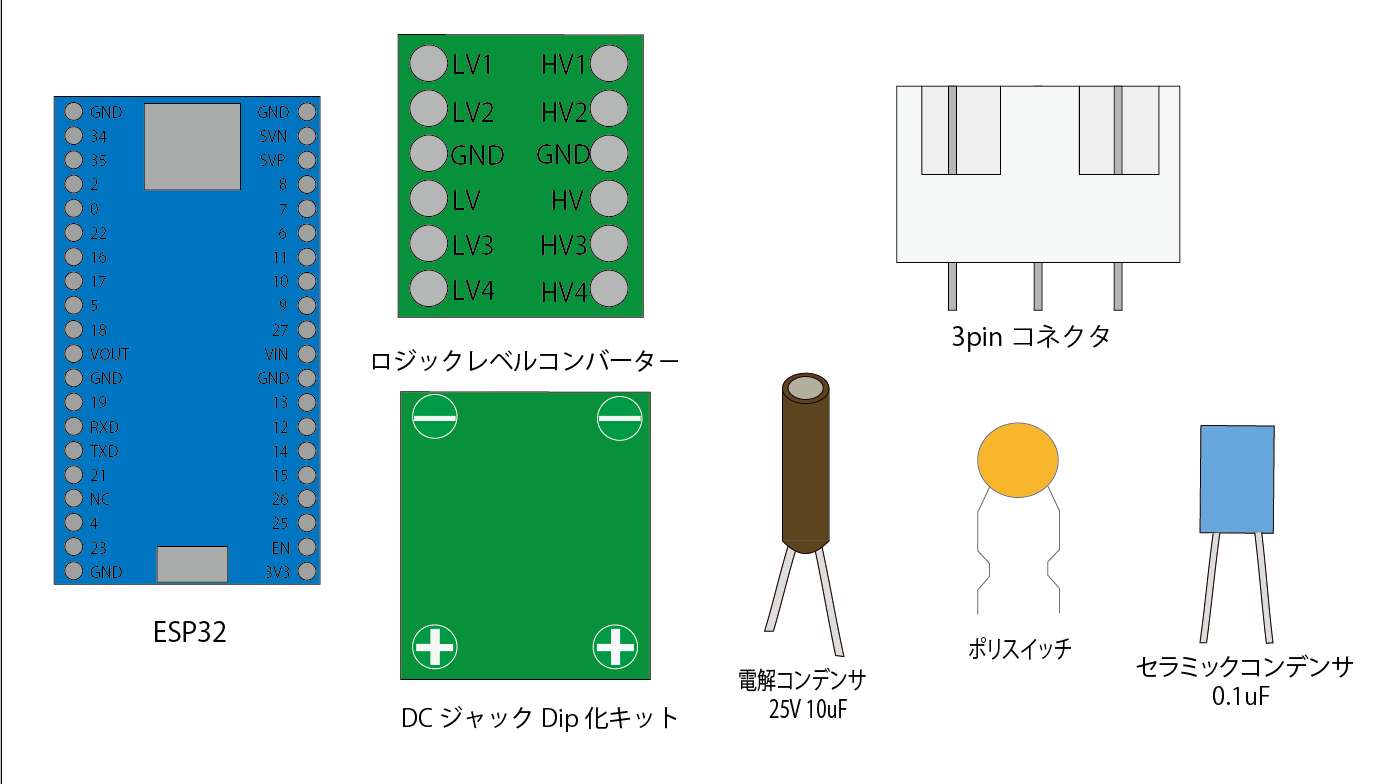


図 15　使用部品



図 16　回路図

# ソフトウェア

## 脈波について

心拍を取得するコードが残っていなかったため昨年の研究の報告書を参考に作成した.[6]

### 脈波の取得

センサーの電源ピン，GNDピン，アナログピンをESP32に接続することで，心拍を0~4095の電圧値で取得することができる．取得にはanalogRead関数を使用する．analogRead関数は指定したアナログピンからの値を読み取る関数である．

アナログピンを34ピンと35ピンに設定している．(図 17)ピンから読み取った値はそれぞれ用意した配列に格納していく.データは５０個格納している.(図 18)

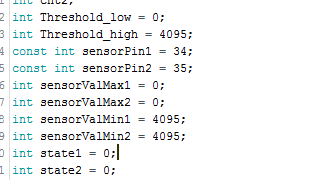


図 17　ピンの選択



図 18 　値の格納

### 値の平滑化

脈波センサーから取得した値(RAWデータ)のままだとばらつきがあるため値を平滑化する必要がある.平滑化をするのに移動平均という方法を使った.

50個のデータの総和を求め,平均を算出している.(図 19)

平滑化前とは平滑化後の図20のようになっている.(図 20)

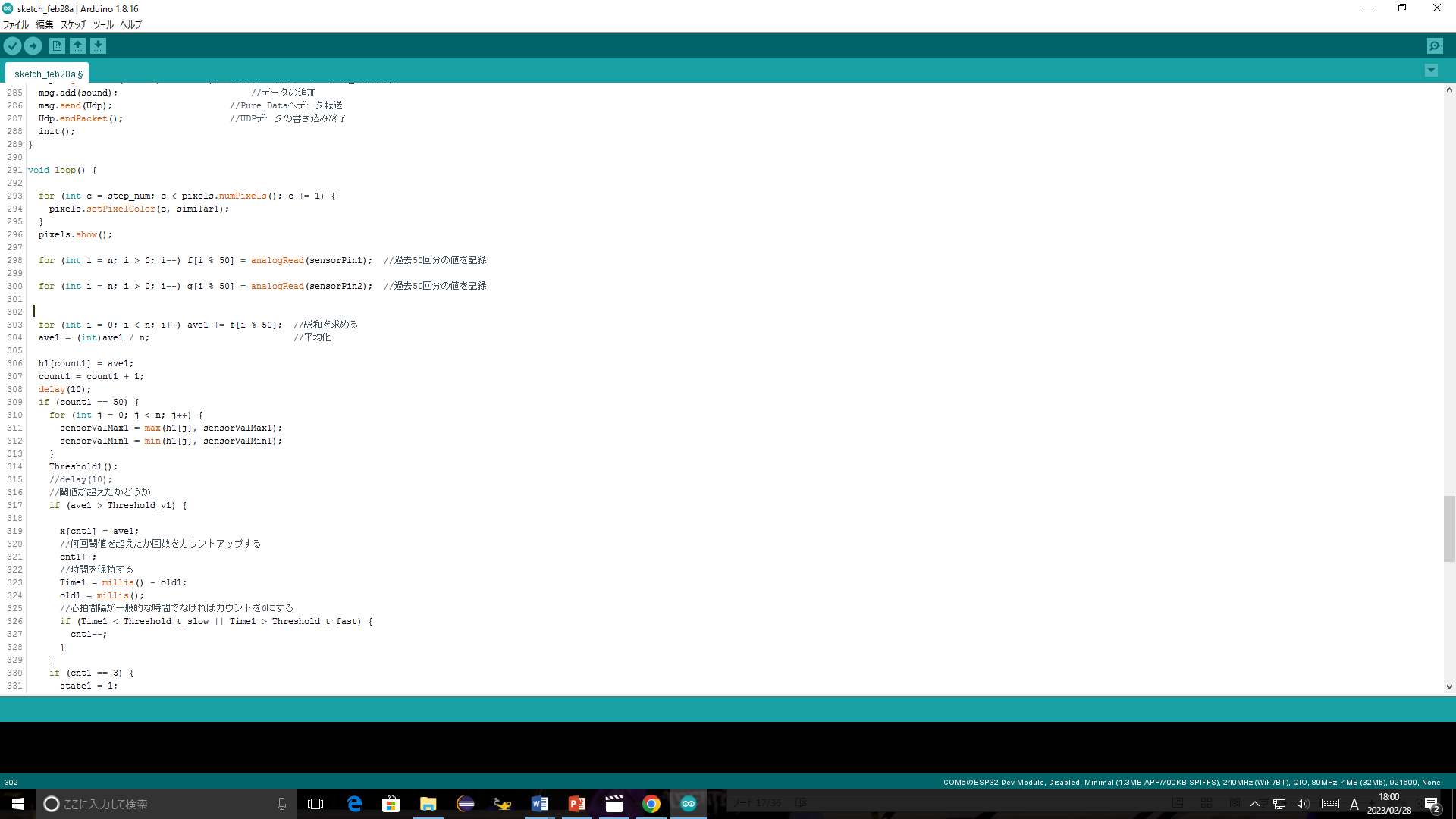


図 19　平滑化処理

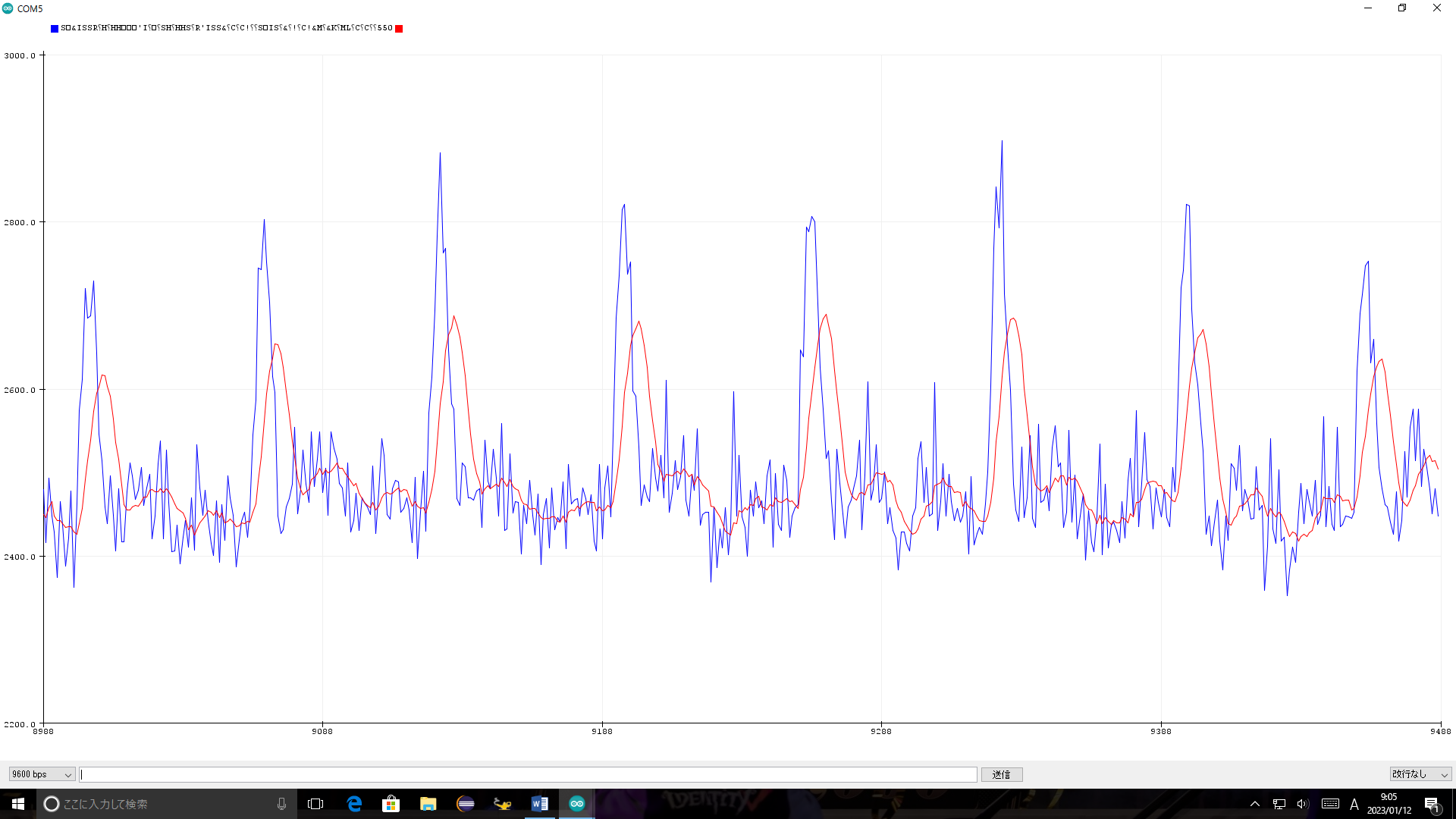


図 20 平滑化前と平滑化後の比較　青：平滑化前　赤：平滑化後

### 閾値の設定

心拍の波とノイズによる波の識別をするために平滑化したデータから閾値を設定し,閾値を超えたら心拍の波と判定する.閾値の設定はデータ50個ごとの最小値と最大値を使用して,波の中間を閾値とするようにした

平滑化したデータを配列に格納し,５０個ごとの最大値,最小値を取得する.

最大値,最小値の取得後,閾値の設定をするThreshold関数を呼んでいる.(図 21)

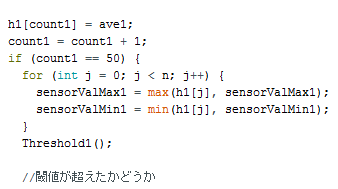


図 21　最大値,最小値の取得

ノイズや指を入れ始めは最大値と最小値の差が大幅に変化するため今回は最大値と最小値の差が100以上1000以下であれば,閾値の設定を行うことにしている.ノイズや指の入れ始めによる値と判断した場合は閾値にセンサーで取得できる最大の値である4095を渡すことで値が超えないようになっている.(図 22)

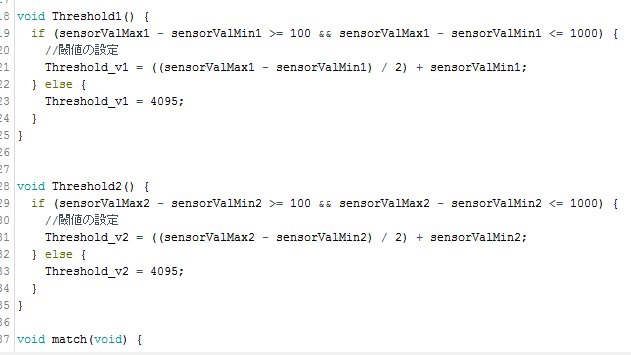


図 22　閾値の設定

閾値を算出する式は最大値をMax，最小値をMinとして，心拍の波の中間Tを次の式より算出し，閾値とした．

### 波の検出

課題として,閾値の設定をしてもノイズの波も心拍の波と判定をしてしまうことがあった.そのため,閾値の設定に加えて波の間隔にも閾値を設定する必要があった.人の心拍の波の間隔は,約0.6～1.2秒なので閾値も0.6～1.2秒に設定した.しかし,波の間隔 0.6～1.2秒でない人の場合ノイズの波と判定され値が取得することができなかった.また,処理速度の関係で時間に多少のタイムラグが発生していた.そこでノイズの波と心拍の波を識別できる範囲の中で間隔に余裕を持たせることにした.今回は1.0秒から3.0秒の範囲に設定している(図 23)

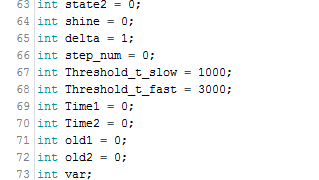


図 23　波の間隔設定

平滑化した値が閾値を超えたら配列に格納する.格納した値の波の間隔がノイズの波と判定した場合はその値は消去する.(図 24)

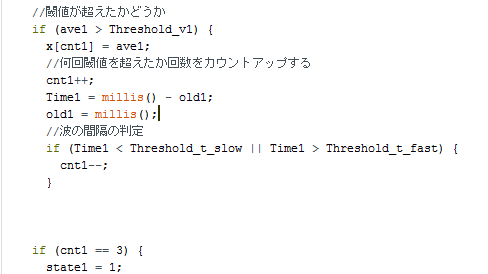


図 24　波の検出

### データの保存

閾値を超えた値を3個取得するとstateという変数に１を渡す.

その後,閾値を超えた値を数える変数,配列の変数,最大値,最小値の初期化をする.(図 25)

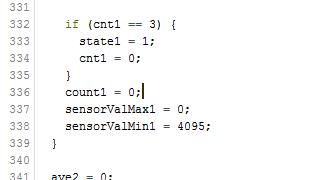


図 25　データの保存

### 類似度判定

センサー二つとも閾値を超えた値を３個取得すると類似度判定をするmatch関数を呼ぶ.(図 26)

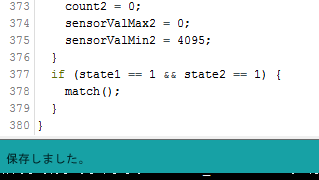


図 26　類似度判定の呼び出し

match関数では類似度算出を行う．類似度の算出にはユークリッド距離を使用した．ユークリッド距離は２点間の距離を算出する方法の一つで，２点が直線的にどれだけ離れているのかを示している． ユークリッド距離は値が小さいほど類似度が高いことを意味する．ユークリッド距離は次の式より求められる．

今回は２つの脈波の２点間の距離を使用した.閾値を超えた値を３個取得すると類似度判定に入るため今回の場合,データの個数は３である.(図 27)

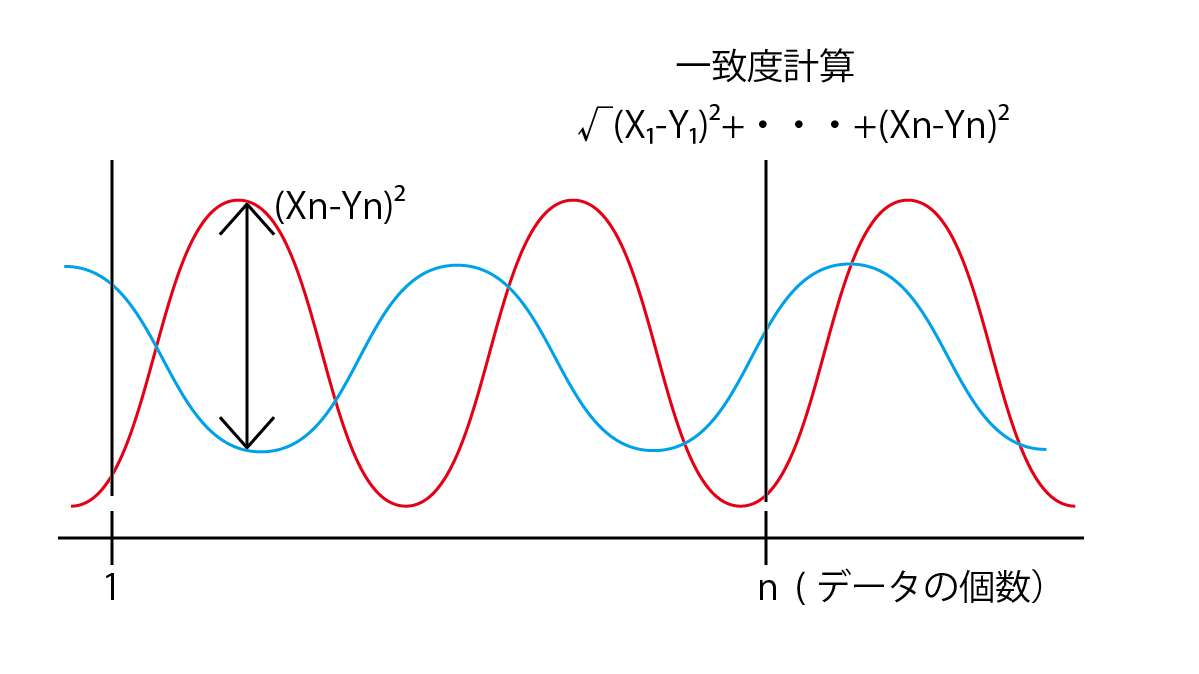


図 27 ユークリッド距離

算出された類似度によって場合分けをするための変数varに値が１~5の段階に分けて,代入される.(図 28)

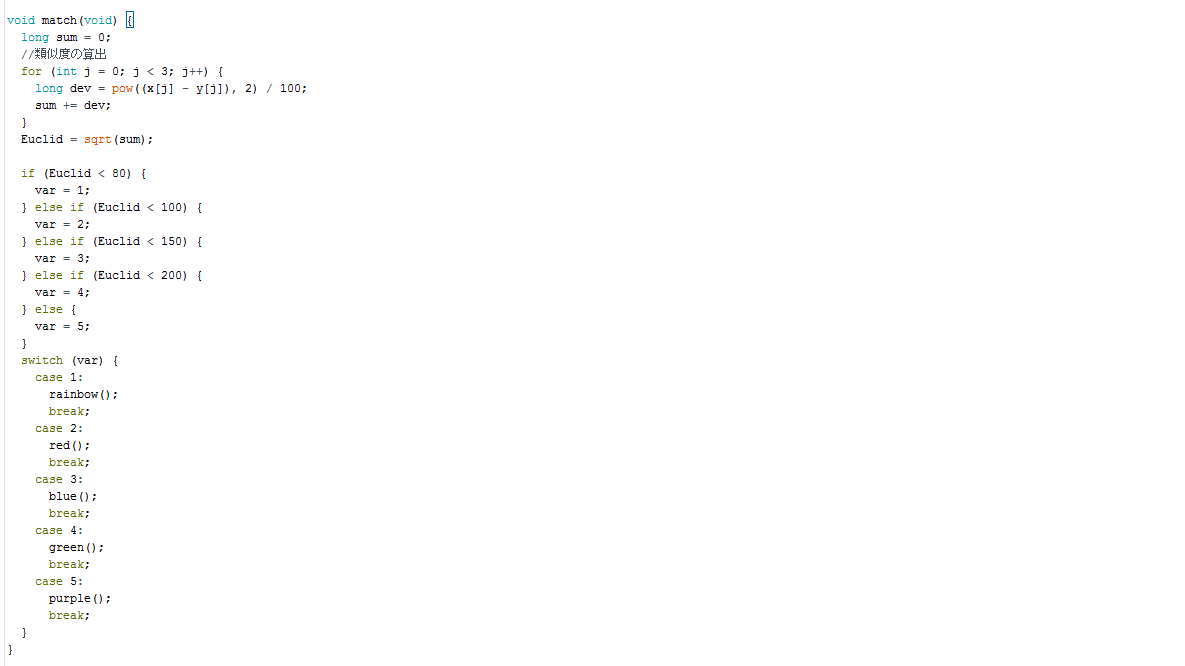


図 28　類似度の算出

varに値が受け渡されると類似度に応じた光と音を出力させる関数を呼び出す.(図 29)

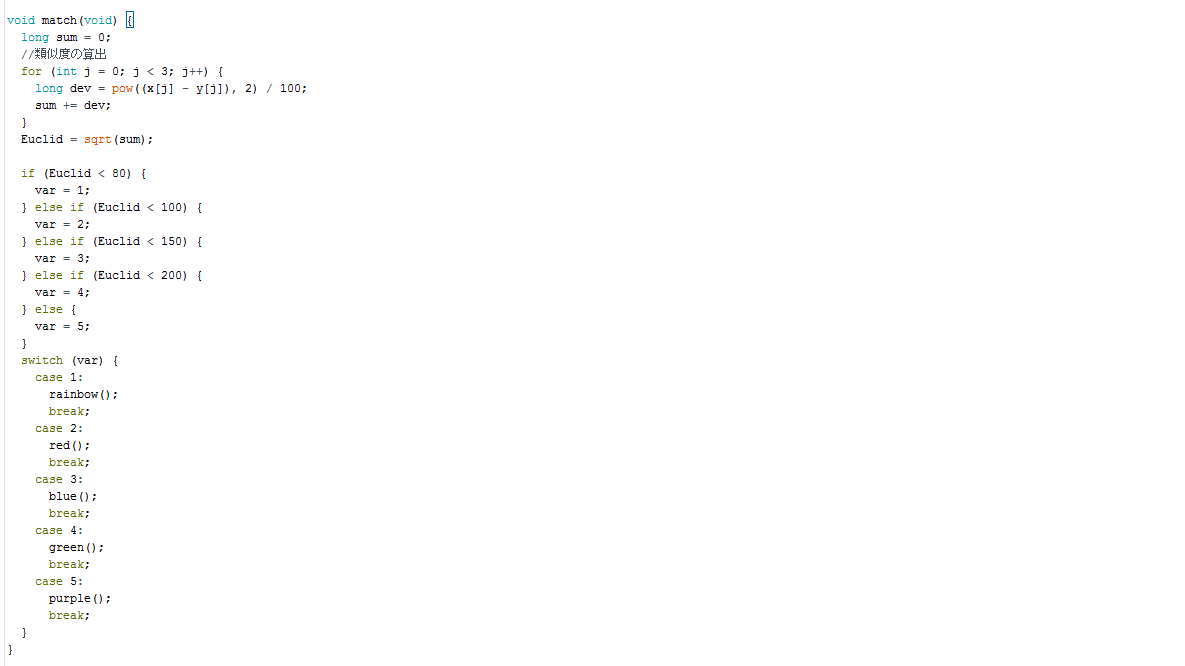


図 29　類似度の場合分け

## 初期化

類似度が決まって,それぞれの点灯と音の出力を5秒間行ったあと,init関数で初期化をする(図 30)

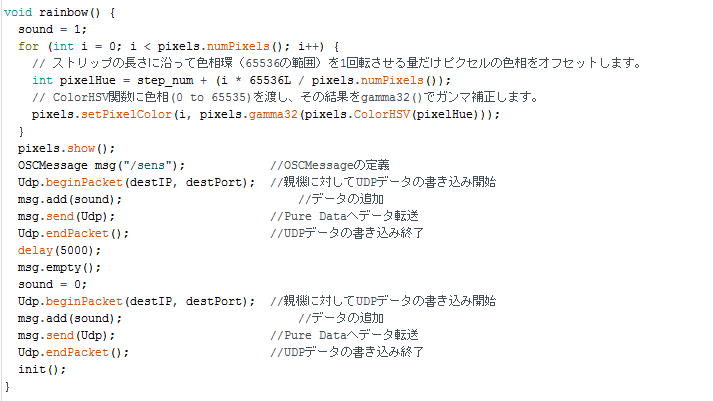


図 30　初期化までの流れ

init関数でRawデータが格納されている配列,無線通信で受け渡す音用の変数,閾値を超えた回数を数える変数,閾値を3回超えた状態を示す変数の初期化をし,点灯も取得中の色に変える.(図 31)

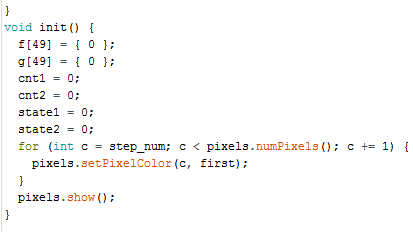


図 31 init関数

LEDと通信の処理については後ほど説明する.

## LED処理について

### Adafurit\_NeoPixelライブラリ

LEDの制御にはAdafurit\_NeoPixelライブラリを使用した．(図 32)

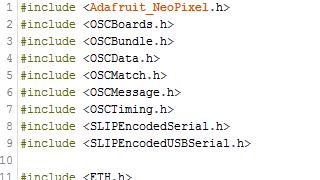


図 32 Adafurit\_NeoPixelライブラリのインクルード

### 光の種類について

今回は５段階の類似度に応じた点灯と脈波を取得中の点灯,計６種類ある.

取得中はこのような点灯をしている.(図 33)



図 　取得中の点灯

類似度に応じた点灯は高い方から虹,赤,青,緑,紫とした.(図 34)

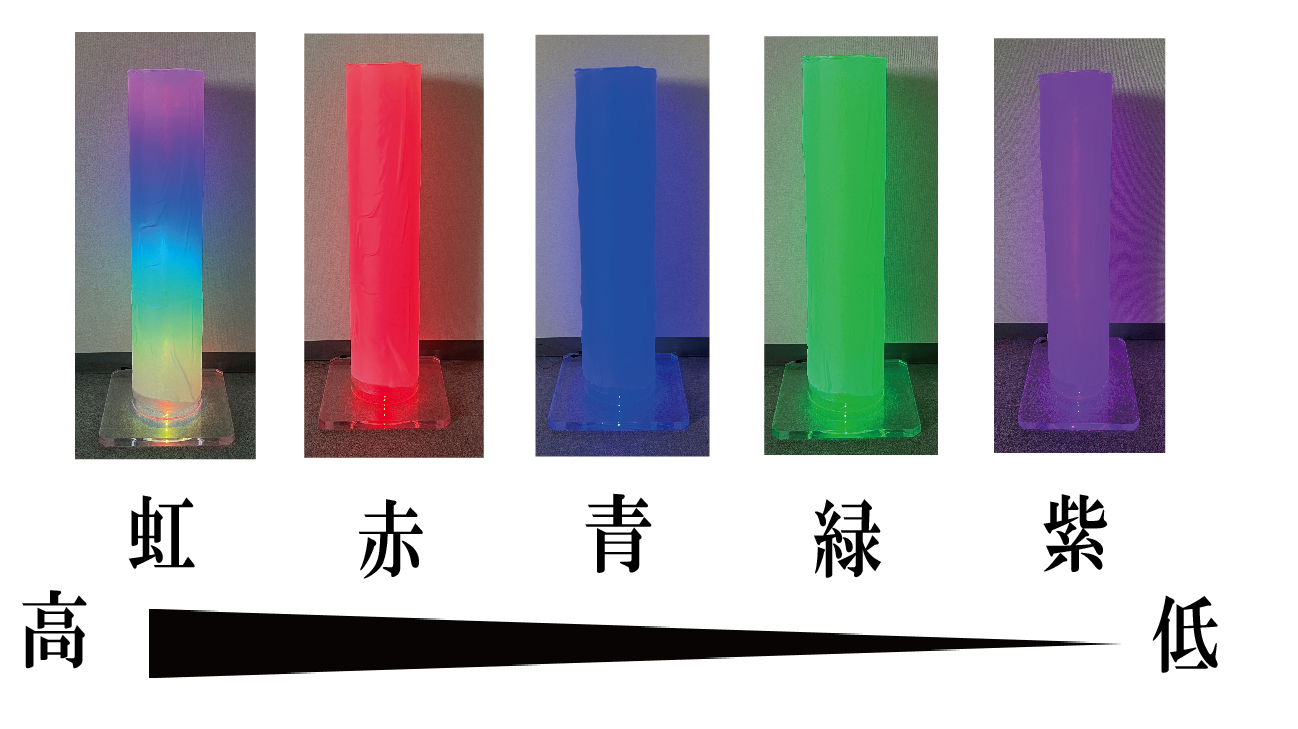


図 34　類似度に応じた点灯

### 色の処理について

#### 虹について

今回使用するNeoPixelの性質を活かしてLED１個ずつに違う色を出力させる処理にしている.

また,虹は制御が不安定で色にばらつきが出てしまうことがあったためガンマ補正の処理も入れている.(図 35)

#### 

図 35　虹色の処理

#### 

#### 虹以外の処理について

虹以外の色の処理はRGB値によって変化させている.(図 36)点灯の処理としては一つずつRGBでしている値を受け渡すようになっている. (図 37)

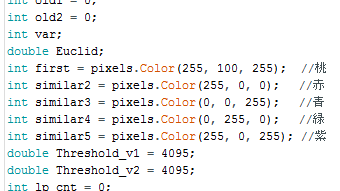


図 36 色の指定

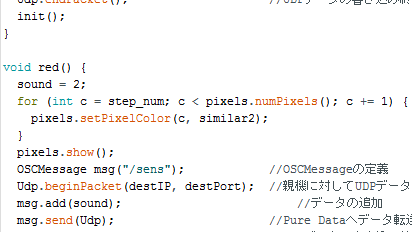


図 37　点灯の処理

## Pure Data(以下Pd)

### ライブラリ

今回の研究では,音の出力は無線通信を使用することにした.ライブラリはOSCとWiFiを使用した.(図 38)

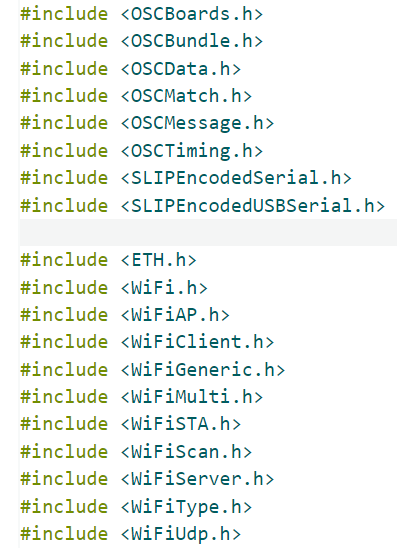


図 38　ライブラリ

### 変数

通信用に用いた変数については以下のとおりである.(図 39)

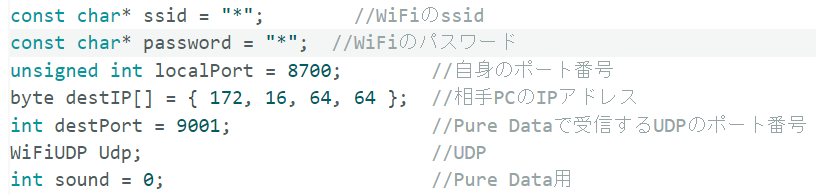


図 39　通信用の変数(ssidとpasswordは非表示)

・ssid：Wi-FiのSSID

・password：Wi-Fiのパスワード

・localPort：ボード側のポート番号

・destIP[]：相手側のIPアドレス

・destPort：Pure Dataで受信するUDPのポート番号

・Udp：UDP通信するための変数

・sound：Pure Data用変数

### Wi-FiとUDP接続

まずWi-Fi接続を開始する.Wi-Fi接続されていない間,シリアルモニタに”.”表示する.Wi-Fi接続に成功したら,シリアルモニタに”Wi-Fi connected.”が表示させる.その後,UDP接続を開始する.(図 40)

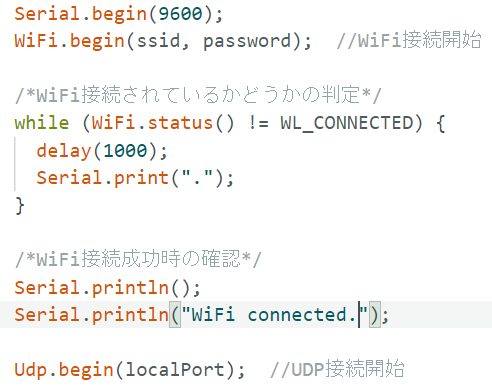


図 40　Wi-FiとUDP接続

### OSC通信

OSC通信については以下のとおりである.(図 41)

・OSC Message型の変数msg(“/sens”)を定義する.

・UDPデータの書き込みを開始する.

・Pure Data用の変数soundをmsgに追加する.

・データをPure Dataへ転送する.

・書き込みを終了する.

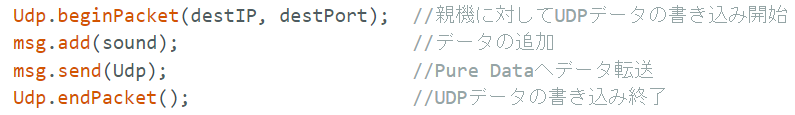


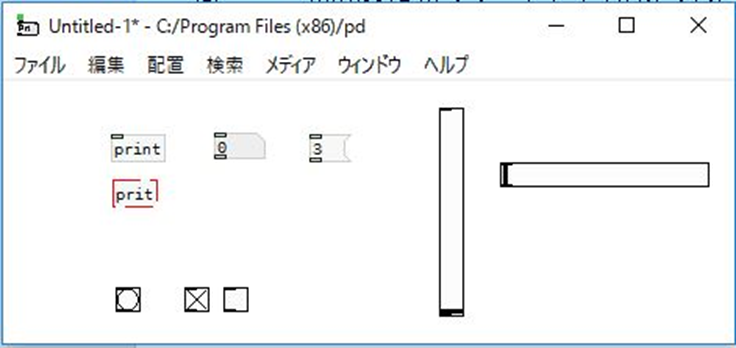
図 41　OSC通信

### 機能

　Pdファイルやプログラムのことをパッチ(patch)と呼ぶ.Pd.プログラミングは,オブジェクトと呼ばれる各々異なる機能を持った[箱]を,パッチコードと呼ばれる[線]で繋いでいくことが基本となる.

### 主なオブジェクト

Pdには様々な種類のオブジェクトが存在する．ここでは主なオブジェクトの説明をする．(図 42)



①

②

③

④

⑤

⑥

図 42 様々なオブジェクト

①オブジェクト：オブジェクト名を入力して設定する箱．オブジェクト名が間違っていると赤い点線になる．

②ナンバーボックス：数値データを出力する．

③メッセージボックス：数値や文字列データ，リストを出力する．

④スライダー（垂直・水平）：設定した範囲内で数値の出力をする．

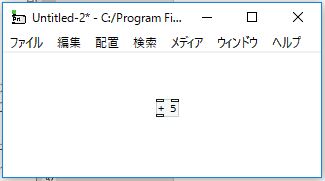
⑤Bangボタン：実行モードでクリックするとBangデータを出力する．

⑥Toggleボタン：チェックのオン/オフで数値データの1/0を出力する．

### インレット・アウトレット

　オブジェクトの上についている「穴」をインレット,下についている「穴」をアウトレットという.インレットはメッセージ,すなわちデータやオブジェクトに対する命令を入力するために使う.また,アウトレットからはオブジェクトの行った処理の結果が出力され,それがほかのオブジェクトへのメッセージとなる.

　インレット,アウトレットは複数ある場合が多いため,それぞれに名前がある.インレットが２つの場合は,左インレット,右インレットと呼ぶ.また３つの場合は,左インレット,中央インレット,右インレットと呼ぶ.まれに,それ以上ある場合は左から第１インレット,第２インレット,第３インレット……と呼ぶ.(図 43)



左インレット

右インレット

アウトレット

図 43 インレット・アウトレット

### パッチの作成方法

作成方法について簡単なパッチを用いて説明する．まず，Pdを起動すると，Pdウィンドウが開かれる．PdウィンドウはPd全体の機能設定や，動作ログやエラーメッセージを表示するためのものである．ウィンドウを閉じるとPd自体が終了してしまう．(図 44)

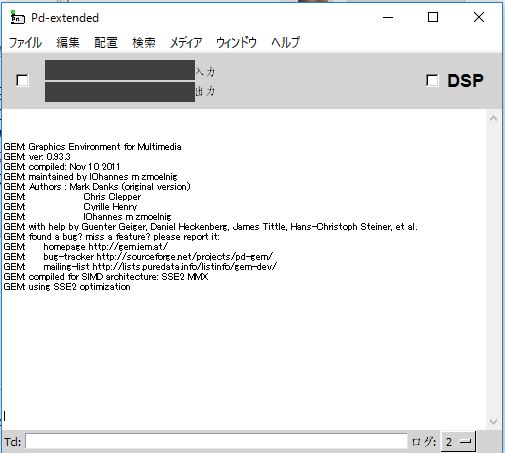


図 44 　Pdウィンドウ

Pdのプログラムは，キャンバスと呼ばれる白地のウィンドウ上に，さまざまな箱を配置し，それらを境界線でつなぐことでデータの流れをプログラムする．ここではサンプルとして，Hello worldと表示するパッチを作成していく．

配置メニューからメッセージボックスを選択し適当な位置に配置する．メッセージボックス内でカーソルが点滅するため，Hello world（以下[Hello world]）と入力する．Pdに入力が終了したことを知らせるためにキャンバスの白地の部分をクリックする．

次に配置メニューからオブジェクトを選択し適当な位置に配置する．オブジェクト内にはprint（以下[print]）と入力し，キャンバスの白地部分をクリックする．(図 45)

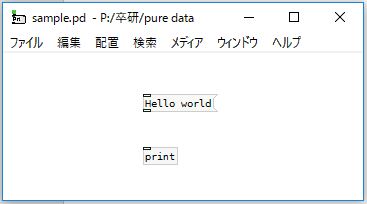


図 45 sample.pd（１）

最後にマウスポインタを[Hello world]の左下にある小さな四角（アウトレット）に合わせクリックし，[print]の左上にある小さな四角（インレット）までドラッグする．マウスボタンを離せば，[Hello world]から[print]へのパッチコードが生成される．(図 46)

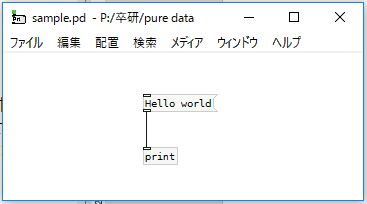


図 46 sample.pd（2）

プログラムを実行するには編集メニューから編集モードを選択し，実行モードに変更する（編集モードの場合カーソルは手のマークになり，実行モードの場合は矢印に戻る）．実行モードにした状態で，[Hello world]をクリックすると，Pdウィンドウのコンソールにprint: Hello worldという行が追加される．(図 47)

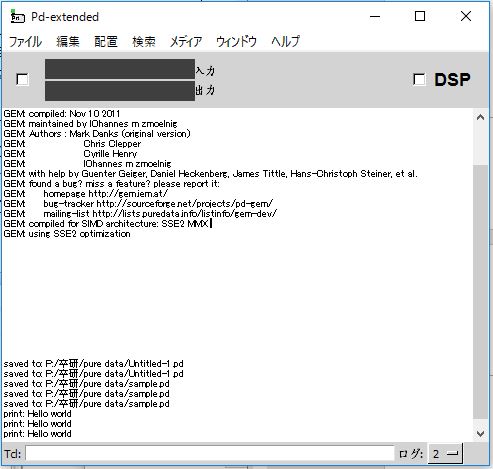


図 47 実行結果

### 受信と音の出力

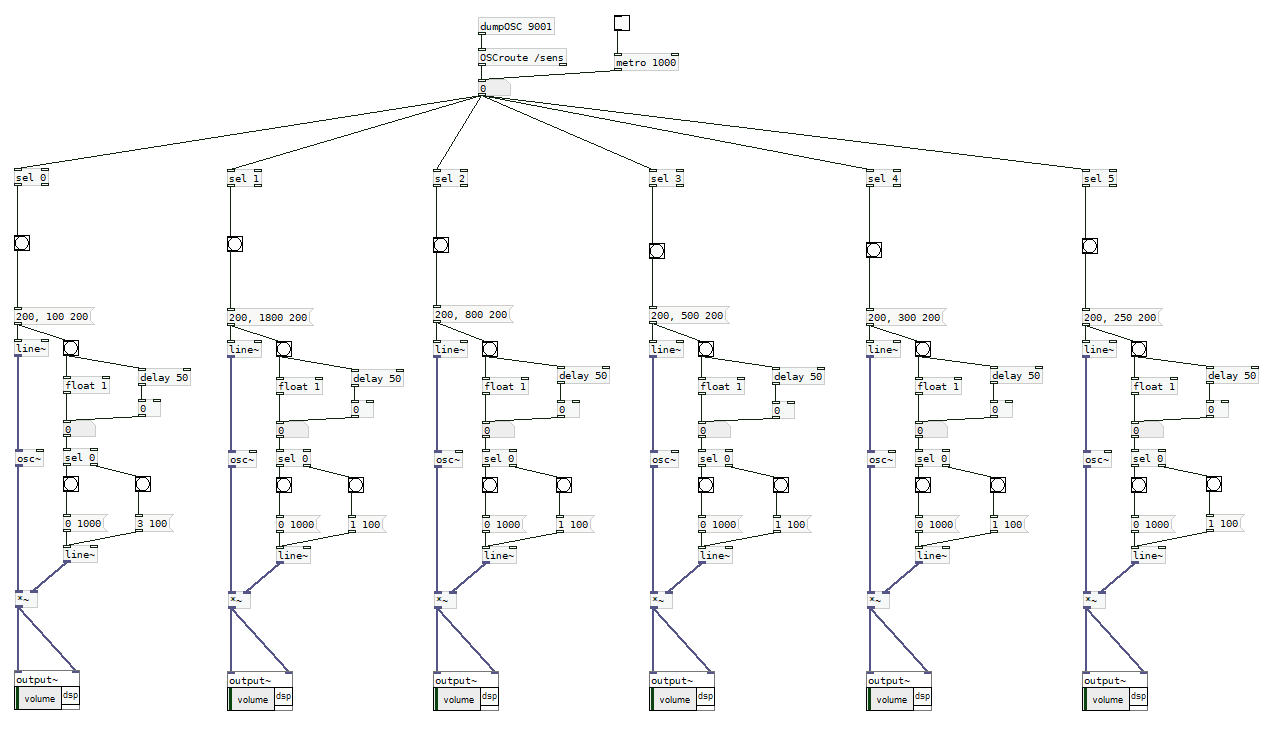


図 48 OscSound.pdパッチ

#### オブジェクト

・[dumpOSC]：指定したポート番号でOSCデータを受信する.

・[OSCroute]：受信したデータの振り分けを行う.

・[metro]：一定間隔でbangを出力する.

・[sel]：インレットからの入力値と引数を比較する.入力値と引数のいずれかが一致する場合は対応するアウトレットからbangを出力し,一致しない場合は１番右のアウトレットから入力値をそのまま出力する.

・[line~]：入力されたfloat値をsignalに変換.所定の時間をかけて徐々にsignalの値を上昇,下降させるときにも使用する.

・[float]：float値を保存し,左インレットにbangを入力された時にその値を出力する.

・[osc~]：周波数を指定して,正弦波を出力する

・[\*~]：左インレットに入力されたsignalの値aと右インレットに入力された引数の値bを乗算し,その結果を出力する

#### パッチ

このパッチに送られてくる値は,0~6のどれかである.測定中の時には,常に0が送られる.そして判定したら,1~5が送られてくる.1~5には類似度判定で出た光り方に対応した音が出る.類似度が高くなるにつれて高い周波数の音が出る.大まかに説明したところで,次にパッチの詳細について部分的に説明していく.

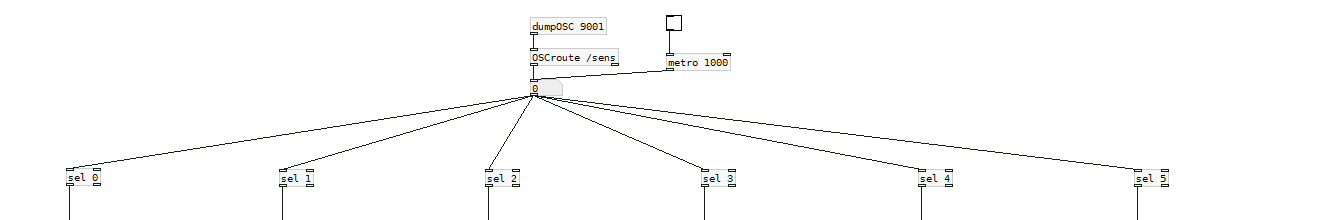


図 49　　OSCデータの受信

　ここでは,OSC通信の中の,データの受信についての処理を行っている.まず,dumpOSCオブジェクトで指定したポート番号のOSCデータを受信する.次に,OSCrouteオブジェクトで受信したデータの振り分けを行う.プログラム内では,/sensという変数に値が振り分けられている.そして,selオブジェクトで/sensの値が指定した数値なら,処理を行う.

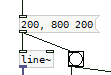


図 50　　周波数変化

ここでは,出力する音の周波数の変化させる処理を行っている.lineオブジェクトでメッセージボックスの第１要素の200hzから第２要素の800hzへ第３要素の200msをかけて徐々に変化させる.

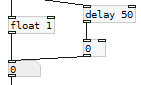


図 51　　音量変化

ここでは,音量を変化させる処理を行っている.floatオブジェクトで指定した数値１をナンバーボックスに出力している.それで,500msたったら,０をナンバーボックスに出力している.

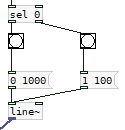


図 52　　ノイズを防ぐ処理

続いての説明に入る.前図の処理だけで音量を変えることができるが,音の再生時と停止時に「ブチッ」というノイズが発生してしまう.よって,それを防ぐ処理を行っている.

図 53　ノイズを防ぐ方法

100ms

100mss

ノイズを防ぐ方法は,音量をいきなり変化させるのではなく,徐々に変化させることである.

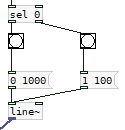


図 54　　ノイズを防ぐ処理

まず,先ほど音量の説明ででたナンバーボックスの値によって,処理が変わる.

値が１の時,selオブジェクトの右側の処理を行い,line~オブジェクトで値を徐々に変化させていく処理を行うことで音量を上げていく.値が０の時,selオブジェクトの左側の処理を行い,line~オブジェクトで値を徐々に変化させていく処理を行うことで音量を下げていく.

#### 音の出力方法

Pdで音の出力を行うには，output~オブジェクトのdspがONになっている必要がある．dspをクリックすることで背景が緑色になり×マークがつく.この状態でvolumeのバーを動かすことで音量の調整ができる．

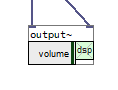


図 55　　output~オブジェクト

### 音の種類について

音も類似度に応じた周波数と脈波を取得中の周波数,計６種類ある.

取得中は100Hzの音を出している.

類似度に応じた周波数は高い方から1800Hz,800Hz,500Hz,300Hz,250Hzとした.(図 56)

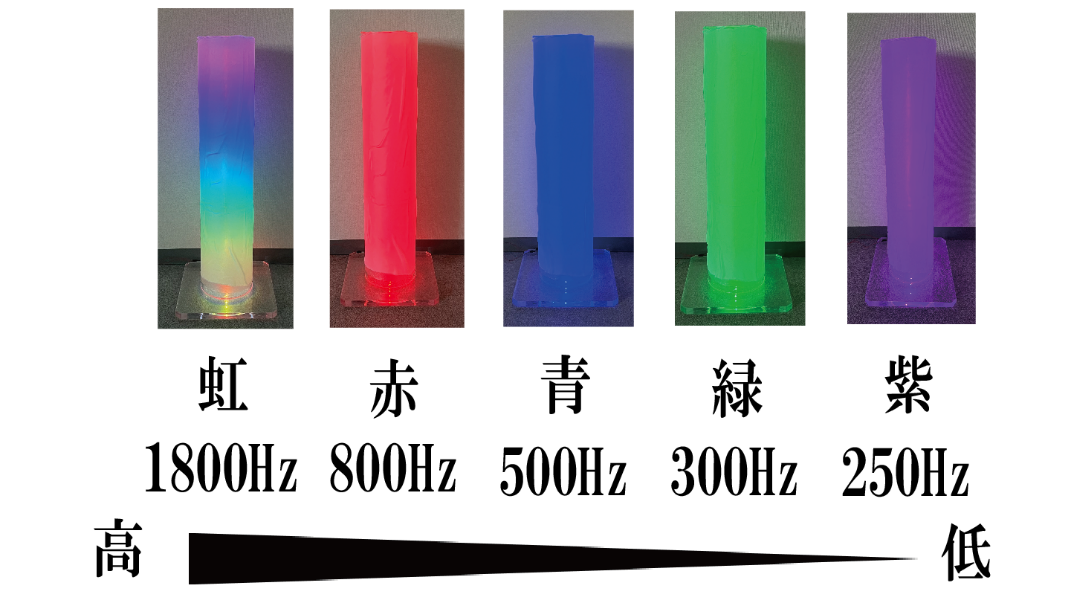


図 56　類似度に応じた周波数

# 終わりに

今回は昨年の研究の引継ぎをし,展示をできる状態を目指し改善と機能拡張を行った.昨年の研究で見つかった接続の問題点を早期に改善することができた.

一方で,コードの修復は報告書に説明がかいてあるにもかかわらず理解・修復に時間を多く費やしてしまい,力不足も実感した.

しかし,機能拡張として音の機能を追加し,体験したひとがより楽しめるアート作品を作成することができた.

「水と向き合う」のようにこれからの産技短のイベントで展示され,実際に触れ,見て楽しんでもらえたらうれしい.

# 参考文献

[1]入力デバイスに水を用いて音を生成するアート作品(産業技術短期大学校矢巾校卒業研究報告書,2016,上山明江,東山真実)

[2] 入力デバイスに水を用いて音を出力・制御する(産業技術短期大学校矢巾校卒業研究報告書,2017,廣藤美緒)

[3] 入力デバイスに水を用いて音と光を出力するシステムの制作(産業技術短期大学校矢巾校卒業研究報告書,2018,田鎖鴻容)

[4] 自由と独立を意味する［independants］を冠した無審査の美術展。仙台から。つづけ。アート。ゴー。

<https://sendai21-independants.com/#gallery>

[5] 「アート＆テクノロジー東北２０１８」コンテスト

<http://www-cg.cis.iwate-u.ac.jp/AT2018/index.html>

[6]心拍信号を利用したアート作品の制作(産業技術短期大学校矢巾校卒業研究報告書,2022,橋本佳太,福田陸斗)

[7]「水と向き合う」

<http://www-cg.cis.iwate-u.ac.jp/AT2018/award.html>

[8]ESP32とNeo PixelフルカラーLEDテープでWi-Fi卓上イルミネーションオブジェを作ってみた

<https://www.mgo-tec.com/blog-entry-led-tape-neopixel-esp32-artnet.html/2>

[9]第30回 OSC通信でArduinoと他のアプリを連携させてみる.(前編)

<https://deviceplus.jp/raspberrypi/entry0030/>

[10]Pure Data Japan　日本のPure Dataユーザのためのフォーラム

<https://puredatajapan.info/?page_id=2>

[11]garretlab　いろいろ試したことを書き留めるページです

<https://garretlab.web.fc2.com/>

[12]Pure Data　チュートリアル＆リファレンス(美山千香氏,株式会社ワークスコーポレーション,2013)

[13]Pd Recipe Book ―― Pure Dataではじめるサウンドプログラミング(松村誠一郎,株式会社ビー・エヌ・エヌ新社,2012)

[14]UDPとは

<https://www.nic.ad.jp/ja/basics/terms/udp.html>