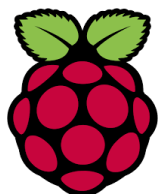


WLOラズパイ倶楽部 Grid-EYEで人感センシング!



2018年3月14日



BLOG

DOWNLOADS

COMMUNITY

HELP

FORUMS

EDUCATION



RASPBERRY PI 3 MODEL B+

1.4GHz 64-bit quad-core processor, dual-band wireless LAN, Bluetooth 4.2/BLE, faster Ethernet, and Power-over-Ethernet support (with separate PoE HAT)

[BUY NOW >](#)[or Buy for Business](#)

本日発表されました!

CPU: ARMv8 QuadCore 1.4GHz
WiFi 802.11b/g/n/ac (2.4GHz/5GHz)
Gigabit Ethernet (USB2.0速度制限)
PoE対応(別売HAT必要)

[GETTING STARTED](#)[SPECIFICATIONS](#)[SOFTWARE & OS](#)[COMPLIANCE](#)

The Raspberry Pi 3 Model B+ is the latest product in the Raspberry Pi range.

- Broadcom BCM2837B0, Cortex-A53 (ARMv8) 64-bit SoC @ 1.4GHz
- 1GB LPDDR2 SDRAM
- 2.4GHz and 5GHz IEEE 802.11.b/g/n/ac wireless LAN, Bluetooth 4.2, BLE
- Gigabit Ethernet over USB 2.0 (maximum throughput 300 Mbps)
- Extended 40-pin GPIO header
- Full-size HDMI
- 4 USB 2.0 ports
- CSI camera port for connecting a Raspberry Pi camera
- DSI display port for connecting a Raspberry Pi touchscreen display
- 4-pole stereo output and composite video port
- Micro SD port for loading your operating system and storing data
- 5V/2.5A DC power input
- Power-over-Ethernet (PoE) support (requires separate PoE HAT)



- Raspberry PiのGPIO I2C端子にGrid-EYEを接続

 - 物理接続

 - Raspbianの設定

- Pythonで値を読んでみる

 - ライブラリの導入

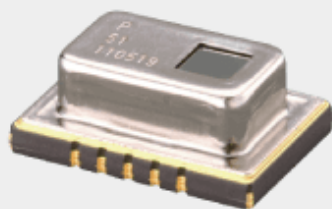
 - 読み込み用スクリプトの導入

- 人感センサーとして試してみる

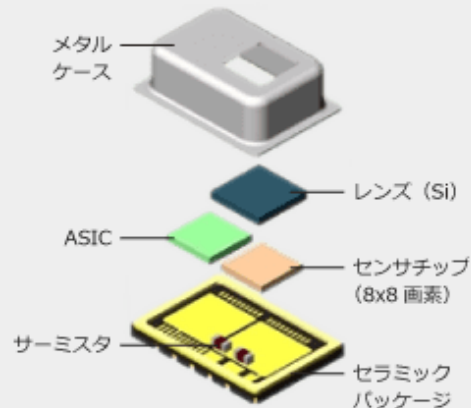
 - センサー用スクリプトの導入

赤外線アレイセンサ『Grid-EYE』とは

非接触での温度分布検知が可能で、家電・産業に幅広く使用されているセンサ

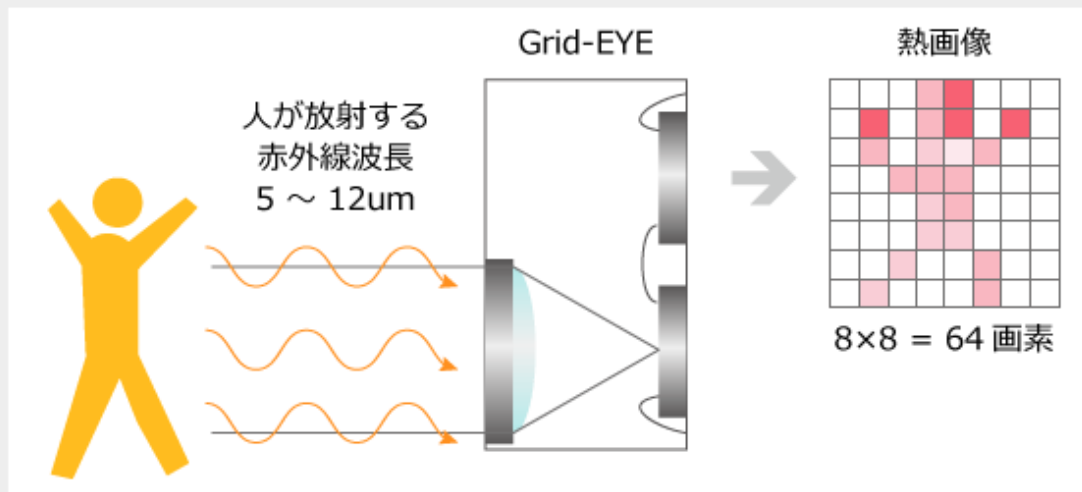


L 11.6mm x W 8.0mm x H 4.3mm



8x8画素で赤外線をつめることで、エリアの温度分布を検知可能

対象の人の動き、存在を検知可能（静止、移動を検知）



<https://industrial.panasonic.com/jp/ds/pr/grid-eye>

- ・8x8(64画素)で2次元温度検知

7m以内、 -20°C ～ $+100^{\circ}\text{C}$ を $\pm 3^{\circ}\text{C}$ の精度で検出

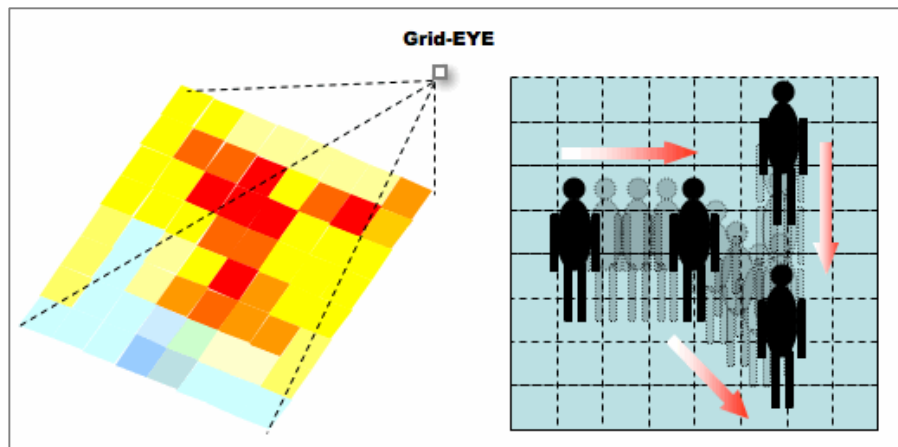
I^2C デジタル出力、最大10フレーム/秒

基準用のサーミスタ温度出力

- ・従来の温度センサーとの違い

非接触(放射赤外線で検出)

静止人体検出、移動方向検出、エリア内温度分布測定が可能



<https://industrial.panasonic.com/jp/products/sensors/built-in-sensors/grid-eye>



・I²C (Inter-Integrated Circuit)

フィリップス社が提唱した周辺デバイスとのシリアル通信の方式で、主にEEPROMメモリICなどとの高速通信を実現する方式です。

<http://www.picfun.com/c15.html>

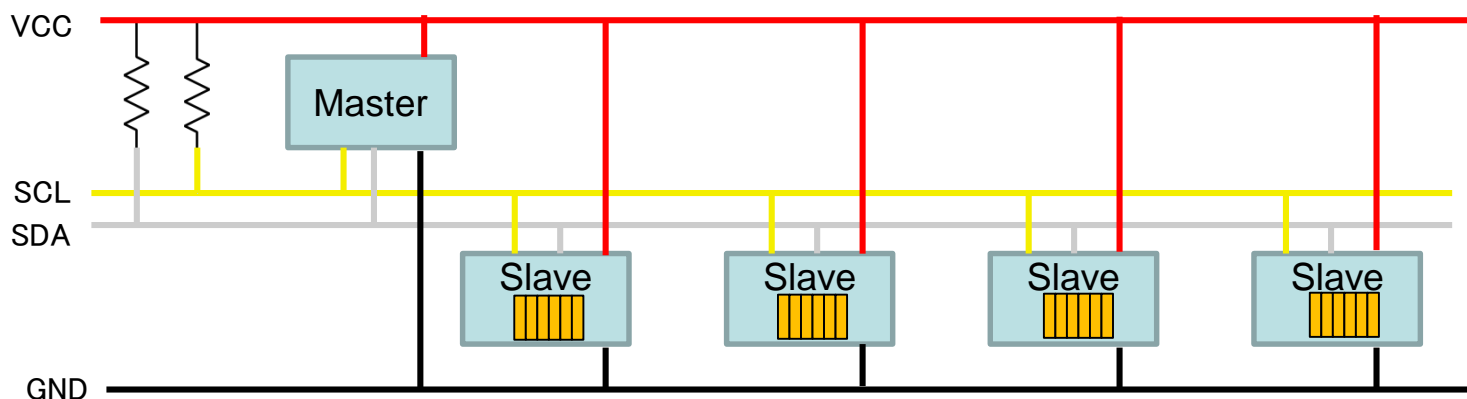
読み方: あい・すくえあ・しー

・特徴

マスタースレーブ: 1台のマスタと1台以上(最大112台)のスレーブを2線の信号線で接続

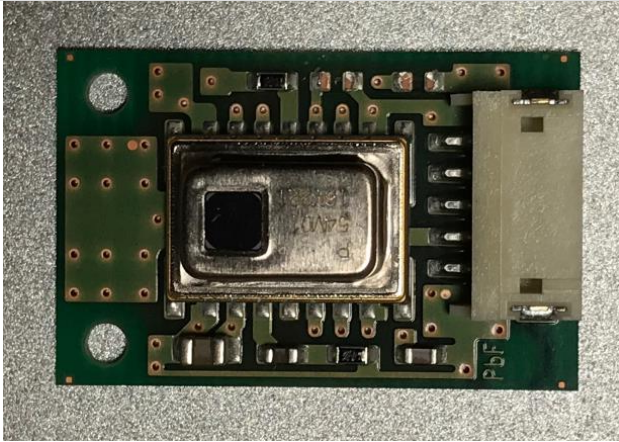
クロック同期通信: 100kbit/s のstandard mode と400kbit/s のFast modeがある

一つのスレーブ内に1バイトのレジスタアドレスを持ち、マスタから指定して読み書き可能

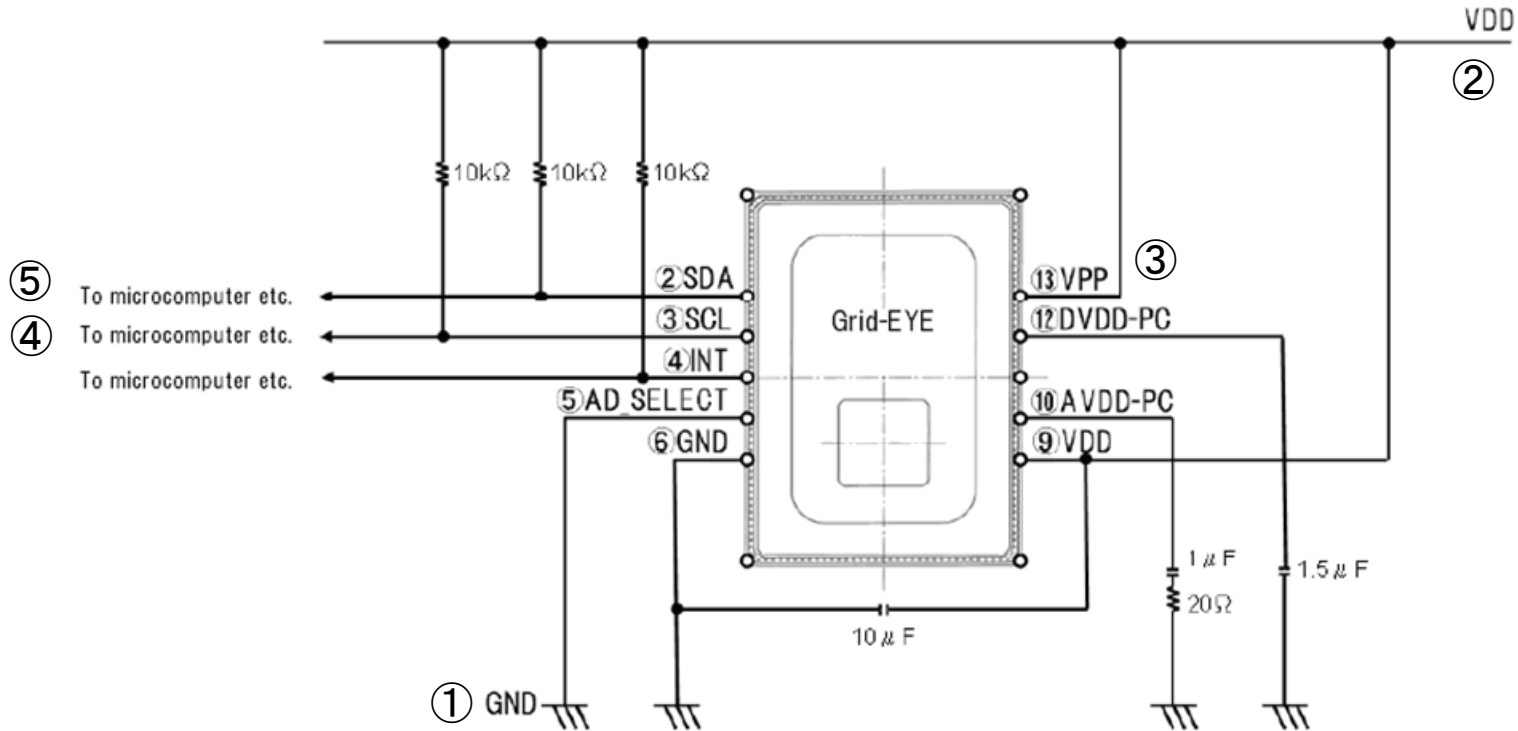


ブレークアウトボード(AIS社提供)

7



SDA	5
SCL	4
VPP	3
VDD	2
GND	1



http://eleshop.jp/shop/g/g122319/

共立電子の店
現在 未ログイン
新規会員登録はこちら
買い物かごは空です...
ログイン
注文方法
よくあるご質問
問合せフォーム
大阪・日本橋店 半生中庭
[シリコンハウス] 店内披露

※表示価格は税込価格です。

ホーム > 【電子部品・半導体】 > センサボード > 温度
eleshop.jp

カテゴリ/大分類から絞り込む
Q
↑ 複数入力例) LED 高輝度 白色 ↑
検索
詳細検索はこちら
■ 機種、その他
■ 未登録商品の注文
■ 商品コード一括注文

送料について
Wonder Pure
共立オリジナルオーディオキット
あの雑誌に掲載されていたアレが欲しい!!
雑誌連動企画

赤外線アレイ・サーモ・センサモジュール (ハッターピン実装済) ■ 数量限定品 ■
AMG88-MODULE-A

web販売価格: ¥4,730
在庫: 57 (現在在庫)
送料区分: 宅配便(ヤマト運輸)
メーカー: 共立電子産業
商品コード: I22319
品名/型番: 赤外線アレイ・サーモ・センサモジュール (ハッターピン実装済) ■ 数量限定品 ■ / AMG88-MODULE-A
登録日: 2018.2.2
数量: 1
買い物かごへ

https://www.switch-science.com/catalog/3395/

SWITCHSCIENCE
OPEN SOURCE HARDWARE SHOP

送料 0 円 ~ 500 円 (税込)
3,000 円または10,000 円以上で送料無料
営業日 14 時までのお支払いで当日発送
VISA JCB Mastercard
PayPal 銀行振込
代引き

商品を探す: amg8834
マイページ ログアウト カート (1 アイテム / 3,500 円)

SWITCHSCIENCE international store
基板製造サービス
スイッチサイエンス PCB

Conta™ サーモグラフィー AMG8833搭載

Panasonic社の 8x8 の赤外線アレイセンサ「AMG8833」を搭載したセンサモジュールです。AMG8833には「Grid-EYE」という愛称がついています。

各素子ごとの温度測定範囲は0℃~80℃です。測定エリアはセンサ正面(上下左右約60度)の四角錐で、このエリアを8x8ピクセルに分割した2次元画像が得られます。

Category: すべて開く
新商品 (131)
スイッチサイエンス製品 (266)
Conta (10)
ベースボード (1)
センサ (9)
ESPr (12)
入門・学習・実験キット (34)
Arduino用シールド (13)
Arduino互換機 (7)
マイコンボード (15)
モジュール (81)
実用 (6)
組立てキット (23)
基板 (25)
部品 (31)
その他 (30)
スイッチエデュケーション製品 (17)
スイッチサイエンスマーケットプレイス (委託商品) (379)
Rapiro (26)
3Dプリンタ (MakerBot) (65)
Arduino (263)
SparkFun (425)
SeedStudio (163)
Adafruit (260)

赤外線が目で省エネ・快適! パナ...

本製品は「Contaシリーズ」の規格に則った製品です。単体での使用はもちろん、Contaベースシールドに搭載することもできます。ピンヘッダははんだ付け済みです。

製品仕様
赤外線アレイセンサ「AMG8833」搭載
入出力は I²C / バスインターフェース
割り込み出力端子
I²C スレーブアドレスは "68h" と "69h" から選択可能 (7ビット表記)
I²C スレーブアドレスの初期値は "68h" (7ビット表記)
8x8 の赤外線アレイセンサにより2次元画像取得可能
検出温度範囲: 0~80℃ (各素子)
人検知が可能な距離: 最長7m

名前: Conta™ サーモグラフィー AMG8833搭載
コード番号: SSCI-033954
SKU#: 3395
送料区分: 150
税込単価: 4,860 円
数量: 1 カートに追加
在庫: 多数
短縮URL: sscl.to/3395
公開日: 2017年12月22日
ツイート
いいね! 11
4.7 ★★★★★
Google
カスタマー

http://git.io/wlopi

https://moosoft.jp/index.php?option=com_content&view=article&id=105&Itemid=140



Adafruit AMG8833 8x8 Thermal Camera Sensor

Build your own mini thermal camera

[Overview](#)

[Pinouts](#)

[Assembly](#)

• [Arduino Wiring & Test](#)

[Arduino Thermal Camera](#)

[CircuitPython Wiring & Test](#)

[Raspberry Pi Thermal Camera](#)

[Downloads](#)

[Featured Products](#)

[Single Page](#)

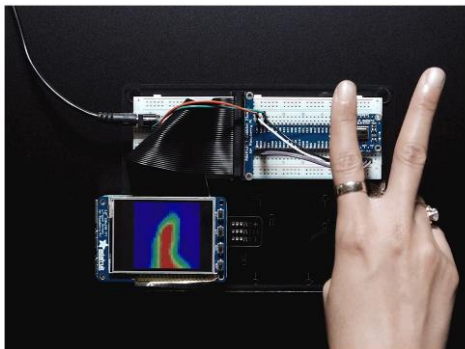
[Download PDF](#)

[Contributors](#)

[Justin Cooper](#)

[SENSORS / TEMPERATURE & HUMIDITY](#) [SENSORS / LIGHT](#)
[SENSORS / CAMERA](#) | [❤](#)

Raspberry Pi Thermal Camera by Dean Miller



The Raspberry Pi also has an i2c interface, and even better has processing capability to interpolate and filter the sensor output. By adding processing power, you can 'turn' the 8x8 output into what appears to be a higher-resolution display.

We're using a PiTFT 2.8" and a Pi Cobbler but the code can be adapted to output to the HDMI display - we're using pygame to draw to the framebuffer.

You can use any Raspberry Pi computer, from Pi A+ to Pi 3 or even a Pi Zero, but we happen to have a Pi 3 on our desk set up already so we're using that.

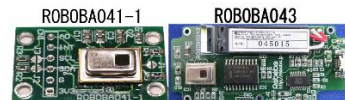


Raspberry Pi 3 - Model B - ARMv8 with 1GB RAM

<https://learn.adafruit.com/adafruit-amg8833-8x8-thermal-camera-sensor/raspberry-pi-thermal-camera>



2次元温度センサ ROBOBA041/ROBOBA043



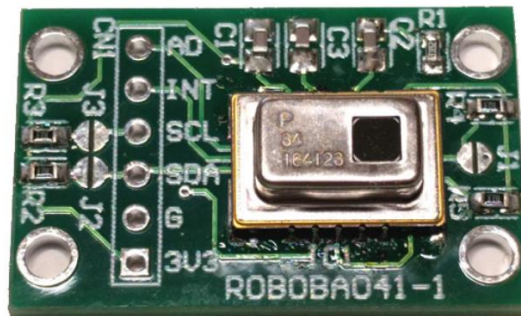
ダウンロードページ: [取扱説明書](#)、[サンプルアプリケーション](#)

センサモジュール販売先: [Amazon](#)

➡ [赤外線アレイセンサ Grid-EYE\(Panasonic 製\)](#) を使った2次元温度センサです。

2017/12/06 ROBOBA041 をバージョンアップしました。

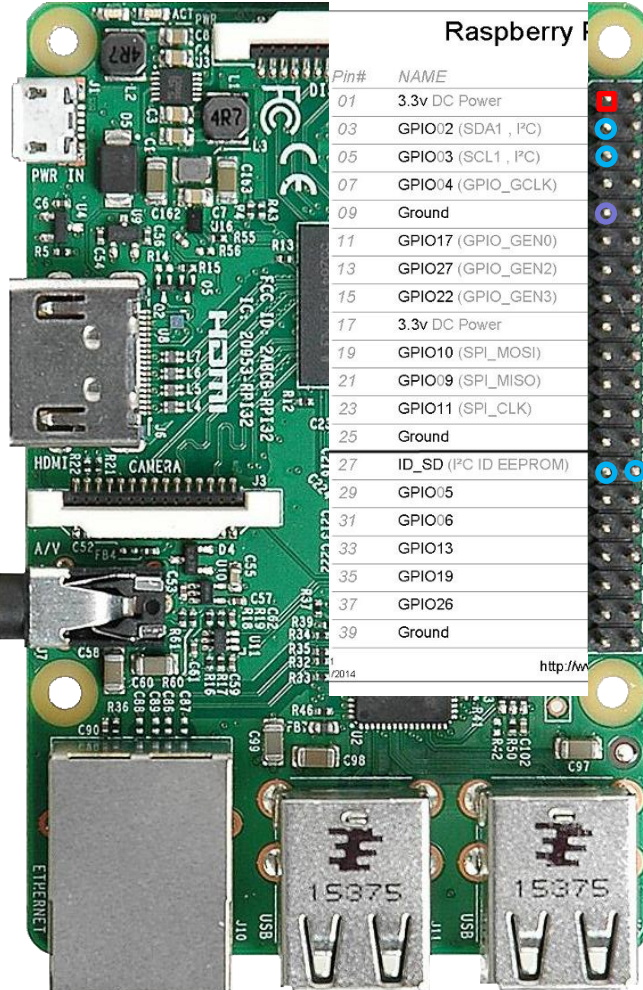
- インターフェース 2.54mmピッチ 6ピンに変更
- センサを最新の AMG8834 に変更
- SDA(J2)、SCL(J3) プルアップ抵抗選択可能(ショートすると 10KΩ プルアップ) 新機能
Arduino 等の内部プルアップの無い MCU 用です。
- アドレス設定 0x69(J1 オープン)、0x68(J1 ショート) デフォルトは 0x69 (旧バージョンと同じ)
- 価格改定: 販売先 [Amazon](#)



2015/10/22 Android アプリケーションをアップデートしました。Android Ver.5、Ver.6 に対応しました。
➡ [ダウンロードページ](#) に公開しました。

- ・2ポートのI²Cを持つが、使えるのはGPIOの3/5pinの1ポート

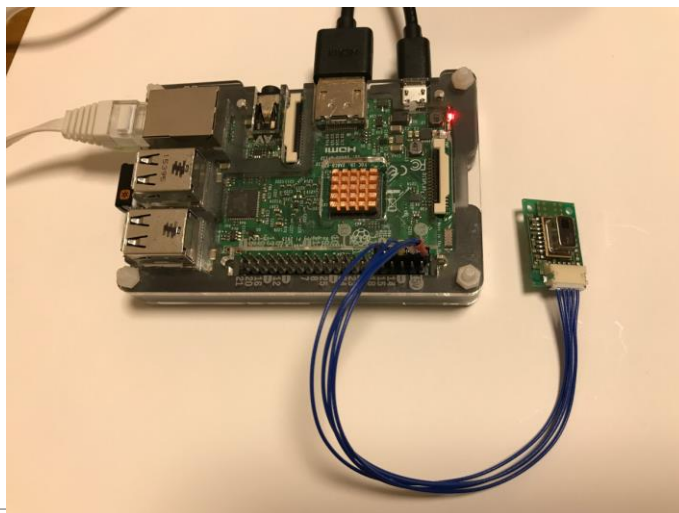
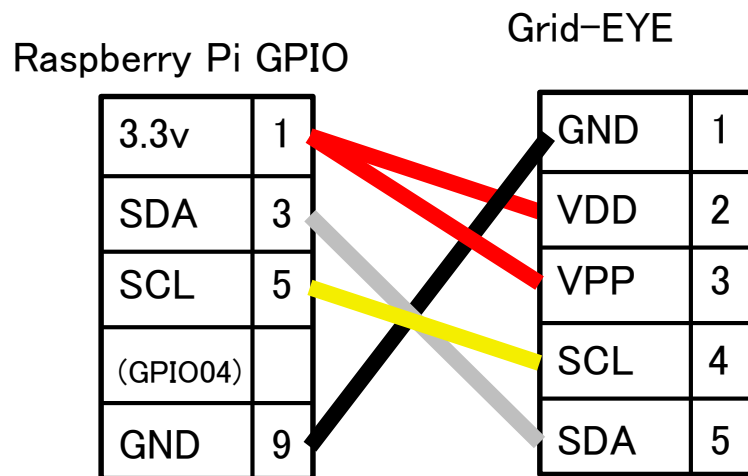
27/28pinはHATのためのEEPROM専用
3.3Vトレラント



The image shows a Raspberry Pi board with a GPIO header overlay. The overlay is a table with two columns: Pin# and NAME. The pins are numbered 01 to 40. The names are: 01: 3.3v DC Power, 03: GPIO02 (SDA1, I²C), 05: GPIO03 (SCL1, I²C), 07: GPIO04 (GPIO_GCLK), 09: Ground, 11: GPIO17 (GPIO_GEN0), 13: GPIO27 (GPIO_GEN2), 15: GPIO22 (GPIO_GEN3), 17: 3.3v DC Power, 19: GPIO10 (SPI_MOSI), 21: GPIO09 (SPI_MISO), 23: GPIO11 (SPI_CLK), 25: Ground, 27: ID_SD (I²C ID EEPROM), 29: GPIO05, 31: GPIO06, 33: GPIO13, 35: GPIO19, 37: GPIO26, 39: Ground, 02: DC Power 5v, 04: DC Power 5v, 06: Ground, 08: (TXD0) GPIO14, 10: (RXD0) GPIO15, 12: (GPIO_GEN1) GPIO18, 14: Ground, 16: (GPIO_GEN4) GPIO23, 18: (GPIO_GEN5) GPIO24, 20: Ground, 22: (GPIO_GEN6) GPIO25, 24: (SPI_CE0_N) GPIO08, 26: (SPI_CE1_N) GPIO07, 28: (I²C ID EEPROM) ID_SC, 30: Ground, 32: GPIO12, 34: Ground, 36: GPIO16, 38: GPIO20, 40: GPIO21.

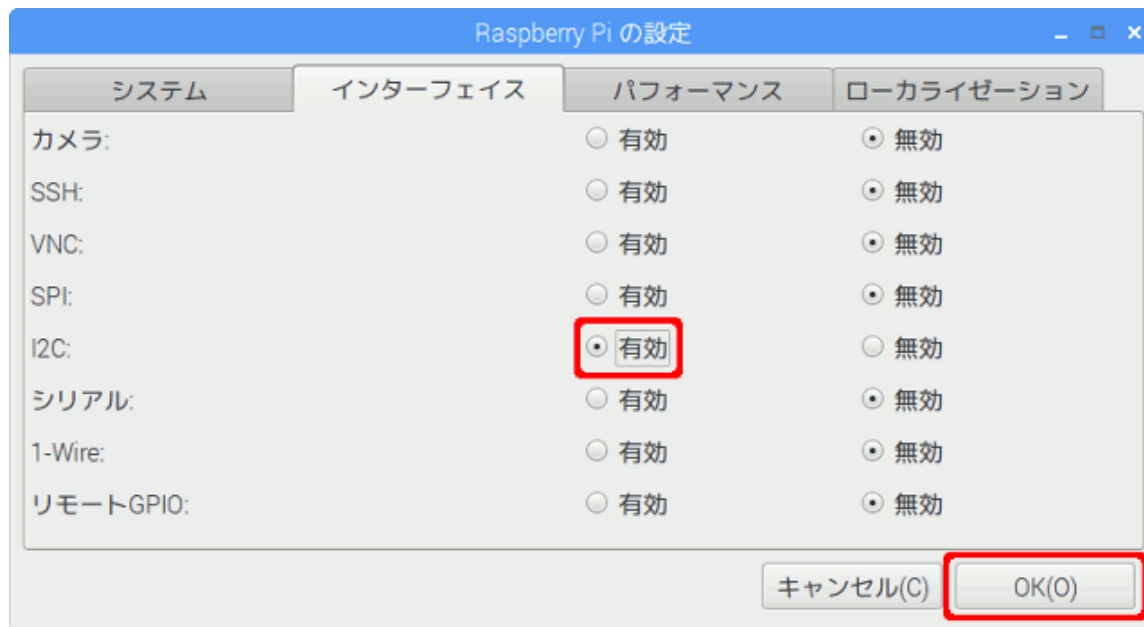
Pin#	NAME	NAME	Pin#
01	3.3v DC Power	DC Power 5v	02
03	GPIO02 (SDA1, I ² C)	DC Power 5v	04
05	GPIO03 (SCL1, I ² C)	Ground	06
07	GPIO04 (GPIO_GCLK)	(TXD0) GPIO14	08
09	Ground	(RXD0) GPIO15	10
11	GPIO17 (GPIO_GEN0)	(GPIO_GEN1) GPIO18	12
13	GPIO27 (GPIO_GEN2)	Ground	14
15	GPIO22 (GPIO_GEN3)	(GPIO_GEN4) GPIO23	16
17	3.3v DC Power	(GPIO_GEN5) GPIO24	18
19	GPIO10 (SPI_MOSI)	Ground	20
21	GPIO09 (SPI_MISO)	(GPIO_GEN6) GPIO25	22
23	GPIO11 (SPI_CLK)	(SPI_CE0_N) GPIO08	24
25	Ground	(SPI_CE1_N) GPIO07	26
27	ID_SD (I ² C ID EEPROM)	(I ² C ID EEPROM) ID_SC	28
29	GPIO05	Ground	30
31	GPIO06	GPIO12	32
33	GPIO13	Ground	34
35	GPIO19	GPIO16	36
37	GPIO26	GPIO20	38
39	Ground	GPIO21	40

- ・「特製ケーブル」で接続。



・I²Cを有効にする

メニュー → 設定 → Raspberry Piの設定 → インターフェイスタブ



※再起動が必要

・パッケージの導入

i2c-tools、python、opencv

```
$ sudo apt install i2c-tools python-smbus libopencv-dev python-opencv
```

- ・i2cdetectコマンドで確認

```
$ i2cdetect -y 1
```

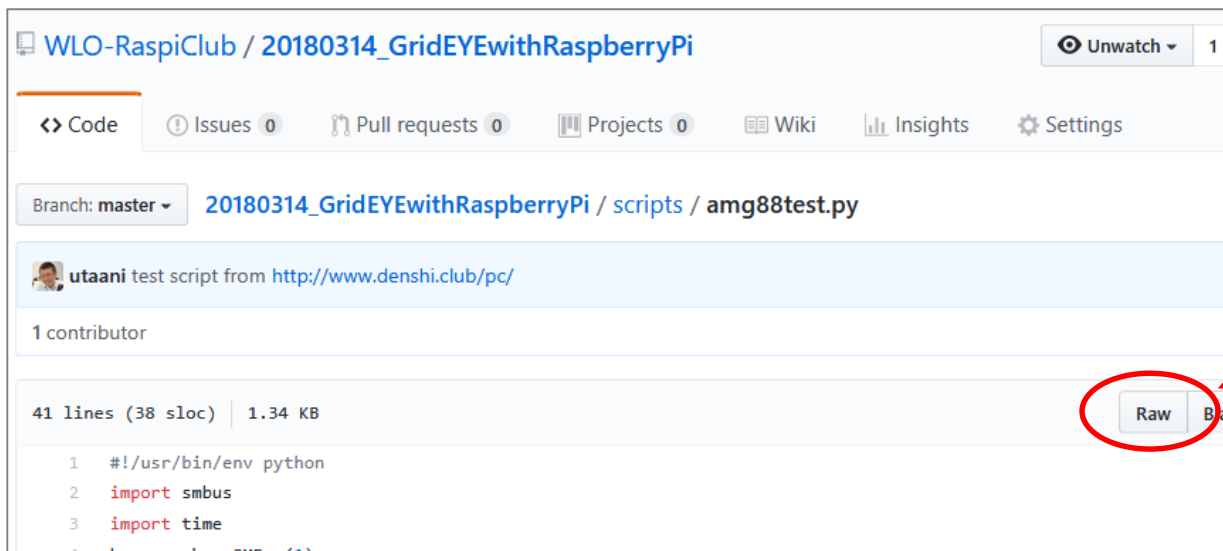
```
pi@raspberrypi:~$ i2cdetect -y 1
   0  1  2  3  4  5  6  7  8  9  a  b  c  d  e  f
00:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
10:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
20:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
30:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
40:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
50:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
60:  --  --  --  --  --  --  --  68  --  --  --  --  --  --  --
70:  --  --  --  --  --  --  --  --  --  --  --  --  --  --  --
pi@raspberrypi:~$
```

認識されたデバイスのアドレスが表示される

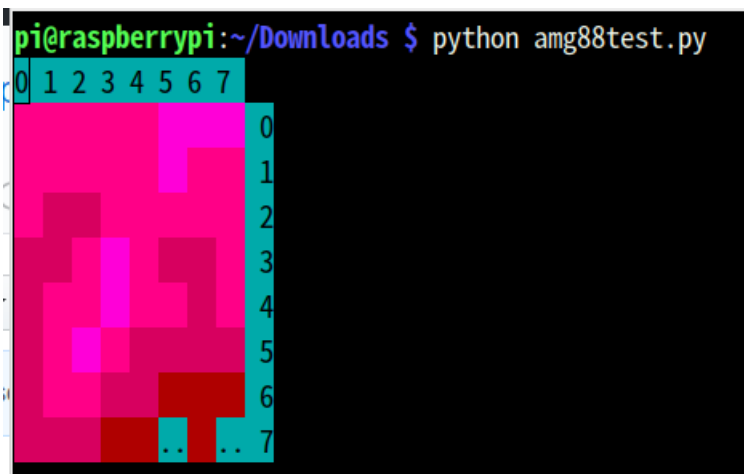
ブレイクアウトボードによって0x68 or 0x69になる
(AD_SELECTピンがGNDの場合は0x68、VDDの場合は0x69になる)

`http://git.io/wlopi`

https://github.com/WLO-RaspiClub/20180314_GridEYewithRaspberryPi/blob/master/scripts/amg88test.py



右クリックで
「名前をつけてリンク先を保存」
でスクリプトを保存



h2>・arms22さんのBlog「なんでも作っちゃう、かも。」で紹介のOpenCVとPythonを使ったサンプル

Raspberry PiでPython(とOpenCV)はじめました。

Posted by arms22 on 2016年01月26日 2 0



仕事ではC/C++を使ったプログラムを書くことが多いのですが、動的型付け言語、所謂スクリプト言語を1つマスターしたPythonを勉強することにしました。

Pythonを選んだ理由は、

- プログラム言語別年収ランキングで一位
国外の話で国内だとあんまり変わらないみたい。orz
- Rubyより速そう
バйнаリパーサーを書いてみただけであまりにも遅いのでPythonならどうだろう？という期待を込めて。
- Raspberry Piにはじめから入っている
環境構築の手間がはぶけて電子工作しつつPythonの勉強ができて一石二鳥。
- OpenCVも使ってみたい
OpenCVは画像処理・画像解析・機械学習用のライブラリ。データサイエンスの分野でPython+OpenCVが流行っているみたいで。

これからPythonをはじめようと思っている方は下記チュートリアルから始めると良いと思います。実際にPythonインタプリタを動かしながら読むとすぐにコードを書けるようになります。

Python チュートリアル - Python 2.7.x ドキュメント

<http://docs.python.jp/2/tutorial/>

今回は前回紹介した2次元温度センサー「Grid-EYE」を使ったサーモグラフィーをRaspberry PiとPythonを使って作り直しました(上は動画)。画面への表示処理にはOpenCVを使っています。

GitHubGist

Search...

All gists GitHub

New gist

arms22 / GridEye.py

Last active 2 hours ago • Report gist

★ Star 0

🍴 Fork 0

Code

Revisions 9

Embed

<script src="https://gist.g" >

Download ZIP

grid_eye_view.py

Raw

```
1  #!/usr/bin/python
2  #-*- coding: utf-8 -*-
3
4  import numpy as np
5  import cv2
6  from GridEye import GridEye
7
8  myeye = GridEye()
9
10 temp_min = 20.
11 temp_max = 30.
12
13 img_edge = 256
14 img = np.zeros((img_edge, img_edge * 2, 3), np.uint8)
15
16 # np.set_printoptions(precision=1)
17
18 while(True):
19     # print 'Thermistor Temp:', myeye.thermistorTemp()
20
21     pixel = np.array(myeye.pixelOut())
22     pixel.resize((8, 8))
23
24     temp_min = temp_min * 0.9 + pixel.min() * 0.1
25     temp_max = temp_max * 0.9 + pixel.max() * 0.1
26     if temp_max < 30:
27         temp_max = 30
28
29     # print 'Pixel Out(Temp):'
30     # print pixel
31
32     pixel = pixel.clip(temp_min, temp_max)
33     pixel = (pixel - temp_min) / (temp_max - temp_min) * 255.0
34     pixel = pixel.astype(np.uint8)
35
36     # print 'Pixel Out(0-255):'
37     # print pixel
```

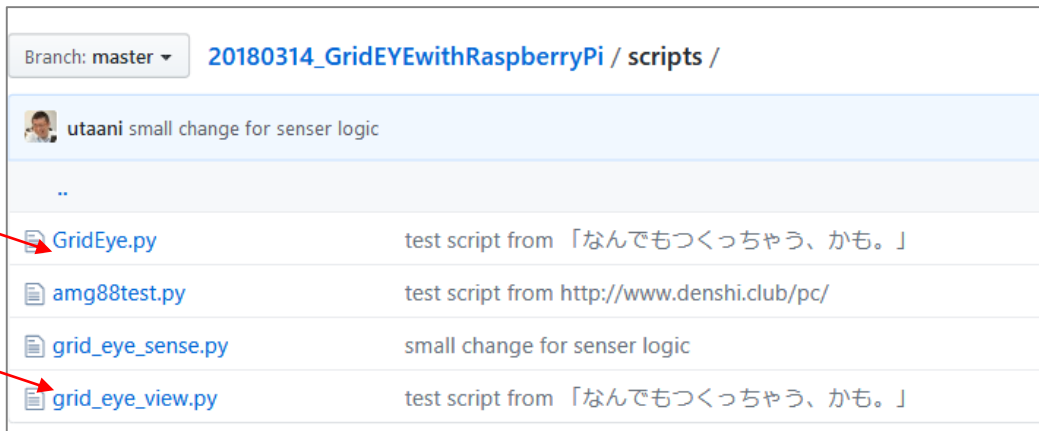
<https://gist.github.com/arms22/e62e682089fe428b1de8>

<http://arms22.blog91.fc2.com/blog-entry-601.html>

<http://git.io/wlopi>

・サンプルスクリプトのダウンロード

https://github.com/WLO-RaspiClub/20180314_GridEYewithRaspberryPi/tree/master/scripts

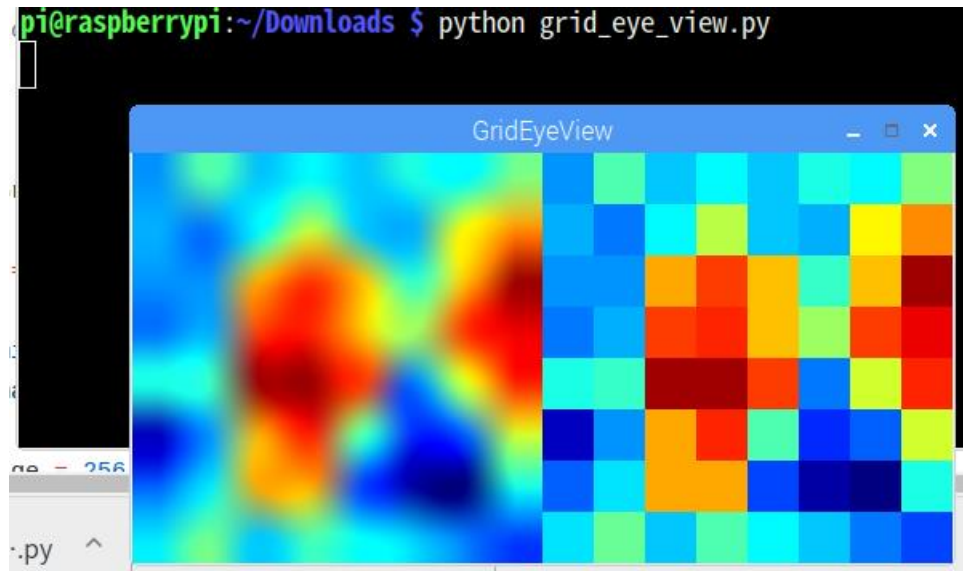


センサードライバ

画面表示



・サンプルスクリプトの実行



・サンプルスクリプトの説明

```
while(True):  
    # print 'Thermistor Temp:', myeye.thermistorTemp()
```

```
    pixel = np.array(myeye.pixelOut())  
    pixel.resize((8, 8))
```

センサーから64回値を読み込み、8x8の2次元配列pixelに格納

```
    temp_min = temp_min * 0.9 + pixel.min() * 0.1  
    temp_max = temp_max * 0.9 + pixel.max() * 0.1  
    if temp_max < 30:  
        temp_max = 30
```

取得した最小値、最大値から表示値用に範囲補正
(最大値を30℃として)

```
    # print 'Pixel Out(Temp):'  
    # print pixel
```

<https://www.learnopencv.com/applycolormap-for-pseudocoloring-in-opencv-c-python/>

```
    pixel = pixel.clip(temp_min, temp_max)  
    pixel = (pixel - temp_min) / (temp_max - temp_min) * 255.0  
    pixel = pixel.astype(np.uint8)
```

補正した範囲に
整数の0-255を割り振る

```
    # print 'Pixel Out(0-255):'  
    # print pixel
```

```
    pixel = cv2.applyColorMap(pixel, cv2.COLORMAP_JET)
```

OpenCVの疑似カラーを割り付ける

```
    # 左側にコピー  
    roi = img[:, :img_edge]  
    cv2.resize(pixel, roi.shape[0:2], roi,  
               interpolation=cv2.INTER_CUBIC)
```

512x256ピクセルの左側に
バイキュービック補間した画像を
貼り付ける

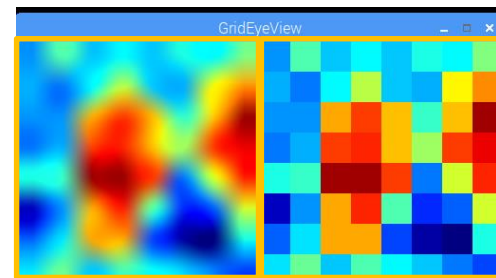
```
    # 右側にコピー  
    roi = img[:, img_edge:]  
    cv2.resize(pixel, roi.shape[0:2], roi,  
               interpolation=cv2.INTER_NEAREST)
```

512x256ピクセルの右側に
最近傍補間した画像を
貼り付ける

```
    cv2.imshow('GridEyeView', img)  
    if cv2.waitKey(100) == 27: # ESC  
        break
```

100ms毎に画面を更新する
ESCキーで終了

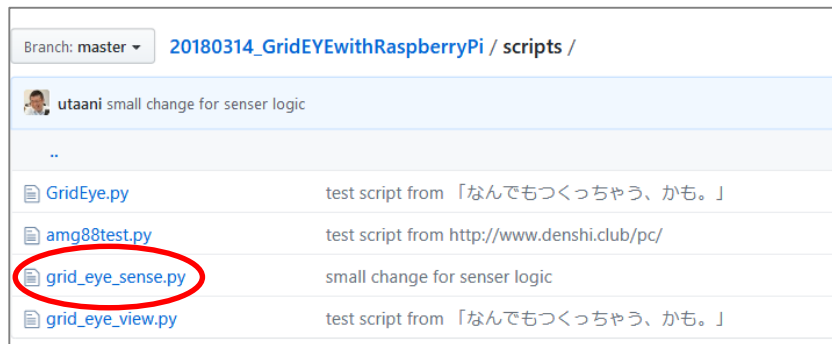
| Value | Name | Scale |
|-------|------------------|-------|
| 0 | COLORMAP_AUTUMN | |
| 1 | COLORMAP_BONE | |
| 2 | COLORMAP_JET | |
| 3 | COLORMAP_WINTER | |
| 4 | COLORMAP_RAINBOW | |
| 5 | COLORMAP_OCEAN | |
| 6 | COLORMAP_SUMMER | |
| 7 | COLORMAP_SPRING | |
| 8 | COLORMAP_COOL | |
| 9 | COLORMAP_HSV | |
| 10 | COLORMAP_PINK | |
| 11 | COLORMAP_HOT | |



バイキュービック補間

最近傍補間

・改造例: 最小値～最大値に差がある場合にのみ標準出力



最小値、最大値の差が10℃以上の場合に人が検出とみなす
→標準出力に温度差を出力

```
while(True):
    # print 'Thermistor Temp:', myeye.thermistorTemp()

    pixel = np.array(myeye.pixelOut())
    pixel.resize((8, 8))

    if pixel.max() - pixel.min() > 10.0:
        print pixel.max() - pixel.min()

    temp_min = temp_min * 0.9 + pixel.min() * 0.1
    temp_max = temp_max * 0.9 + pixel.max() * 0.1
    if temp_max < 30:
        temp_max = 30

    # print 'Pixel Out(Temp):'
    # print pixel

    pixel = pixel.clip(temp_min, temp_max)
    pixel = (pixel - temp_min) / (temp_max - temp_min) * 255.0
    pixel = pixel.astype(np.uint8)

    # print 'Pixel Out(0-255):'
    # print pixel

    pixel = cv2.applyColorMap(pixel, cv2.COLORMAP_JET)

    # 左側にコピー
    roi = img[:, :img_edge]
    cv2.resize(pixel, roi.shape[0:2], roi,
               interpolation=cv2.INTER_CUBIC)
```

- Adafruitのブレイクアウトボードのダウンロードページ

センサーのデータシートやPythonのドライバなど

<https://learn.adafruit.com/adafruit-amg8833-8x8-thermal-camera-sensor/downloads>

- OpenCV 関連

OpenCVはコンピュータによる画像処理(コンピュータビジョン)のためのライブラリ。

<https://opencv.org/>

OpenCV-Pythonチュートリアル

http://labs.eecs.tottori-u.ac.jp/sd/Member/oyamada/OpenCV/html/py_tutorials/py_tutorials.html

ラズパイ3にOpenCV3を簡単に導入

<https://qiita.com/mt08/items/e8e8e728cf106ac83218>

今回は標準パッケージからOpenCV2を導入したが、
最新版では移動方向検知など新機能が追加されている