# BAB 3
# METODOLOGI PENELITIAN

## 3.1 Waktu dan Tempat Penelitian

Pada penelitian mengenai Sistem Pendeteksi Api Menggunakan Sensor Amg8833 *IR Thermal Camera* Pada Robot MR.COOL MK7dilakukan pada :

Waktu : 18 Februari s.d selesai

Tempat : Laboratorium Mikrokontroler dan Robotika Teknik Elektro Universitas Muhammadiyah Yogyakarta.

## 3.2 Alat dan Bahan

Terdapat beberapa alat dan bahan yang dibutuhkan dalam penelitian mengenai pembuatan Sistem Pendeteksi Api Menggunakan Sensor Amg8833 *IR Thermal Camera* Pada Robot MR.COOL MK7. Berikut adalah peralatan dan bahan yang diperlukan.

3.2.1 Bahan

**Tabel 3.1** Bahan

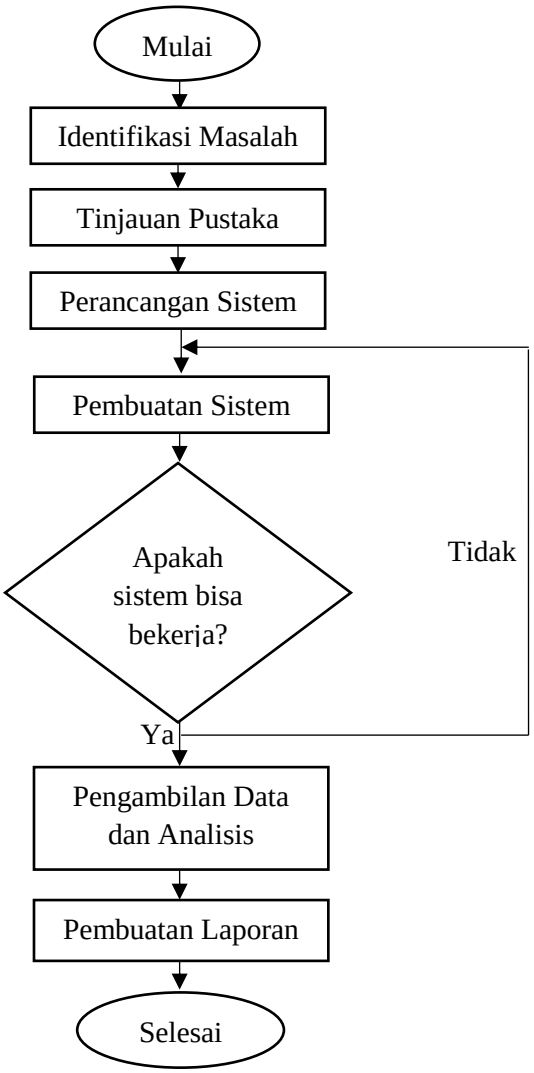
| No | Bahan | Jumlah (buah) |
|---|---|---|
| 1 | Arduino mega 2560 pro | 1 |
| 2 | Downloader K125R | 2 |
| 3 | Sensor AMG8833 | 1 |
| 4 | Stm32 | 1 |
| 5 | LCD alphanumeric 16x2 | 2 |
| 6 | LCD Oled | 1 |
| 7 | Kabel Jumper | Secukupnya |
| 8 | USB Serial | 1 |
| 9 | PCB lubang | 1 |
| 10 | Pin Header Male | 2 strip |
| 11 | Pin Header Female | 2 strip |
| 12 | PCB lubang | 1 |
| 13 | korek | 1 |
| 14 | Tenol | secukupnya |
| 15 | Lilin | secukupnya |

3.2.2 Alat

**Tabel 3.2** Peralatan

| Perangkat Keras | Perangkat Lunak |
|---|---|
| Komputer | Arduino IDE 1.8.7 |
| Solder | Code Vision 3.12 |
| Atraktor | Proteus 8.6 |
| Bor | Fritzing |
| Multimeter | Terminal.exe |

## 3.3 Diagram Alir Prosedur Penelitian




**Gambar 3.1** Diagram Alir Prosedur Penelitian

Dalam pembuatan tugas akhir ini terdapat beberapa kegiatan, dapat dilihat pada diagram alir Gambar 3.1 penelitian ini mulai dari identifikasi masalah, perencanaan, perancangan sistem dan pengambilan data. Untuk penjelasan diagram alir prosedur penelitian adalah sebagai berikut :

a. Identifikasi Masalah

   Tahap ini adalah tahap awal memulai penelitian yaitu mengidentifikasi masalah dalam sistem pendeteksi pada robot MR.COOL MK7.

b. Tinjauan Pustaka

   Pada tahap ini merupakan tahap pengumpulan informasi, referensi dan studi literatur mengenai masalah atau kekurangan pada beberapa penelitian yang terkait sebelumnya. Semua informasi yang dicari dapat berasal dari karya tulis, artikel ilmiah, jurnal, thesis dan disertasi dengan topik yang sama.

c. Perancangan Sistem

   Perancangan adalah tahap paling penting dan tahap awal dari pembuatan sebuah alat. Karena pada tahap ini kita dapat mengetahui apa saja alat dan bahan yang dibutuhkan supaya sistem dapat berjalan sesuai dengan yang diinginkan. Pada penelitian ini di bagi menjadi beberapa perancangan yang dilakukan yaitu perancangan sistem, perancangan perangkat lunak, dan perancangan perangkat keras.

d. Pembuatan Sistem

   Tahap ini merupakan tahap pengerjaan sistem pendeteksi api dengan merealisasikan apa yang sudah dirancang. Di mulai dari rancang bangun perangkat lunak hingga rancang bangun perangkat keras.

e. Pengujian Sistem

   Tahap pengujian sistem adalah tahap uji coba setiap perangkat yang ada, baik perangkat keras maupun perangkat lunak. Uji coba ini bertujuan untuk mengetahui setiap perangkat atau bagian yang ada pada sistem ini berkerja dengan baik atau tidak sesuai dengan tujuan yang diinginkan.

f. Pengambilan Data dan Analisis

   Tahap pengambilan data dilakukan setelah semua sistem dapat berjalan dengan baik dan sesuai dengan tujuan. Data yang diambil adalah data nilai

suhu yang dibaca sensor per-*pixel* sehingga mendapatkan posisi *absolut* titik api. Seluruh data yang diperoleh kemudian dianalisis untuk mengetahui kelebihan dan kekurangan dari sistem pendeteksi api menggunakan sensor AMG8833 ini.

## 3.4 Deskripsi Sistem

Sistem yang akan dirancang pada penelitian ini disajikan dalam bentuk blok diagram perancangan sistem.

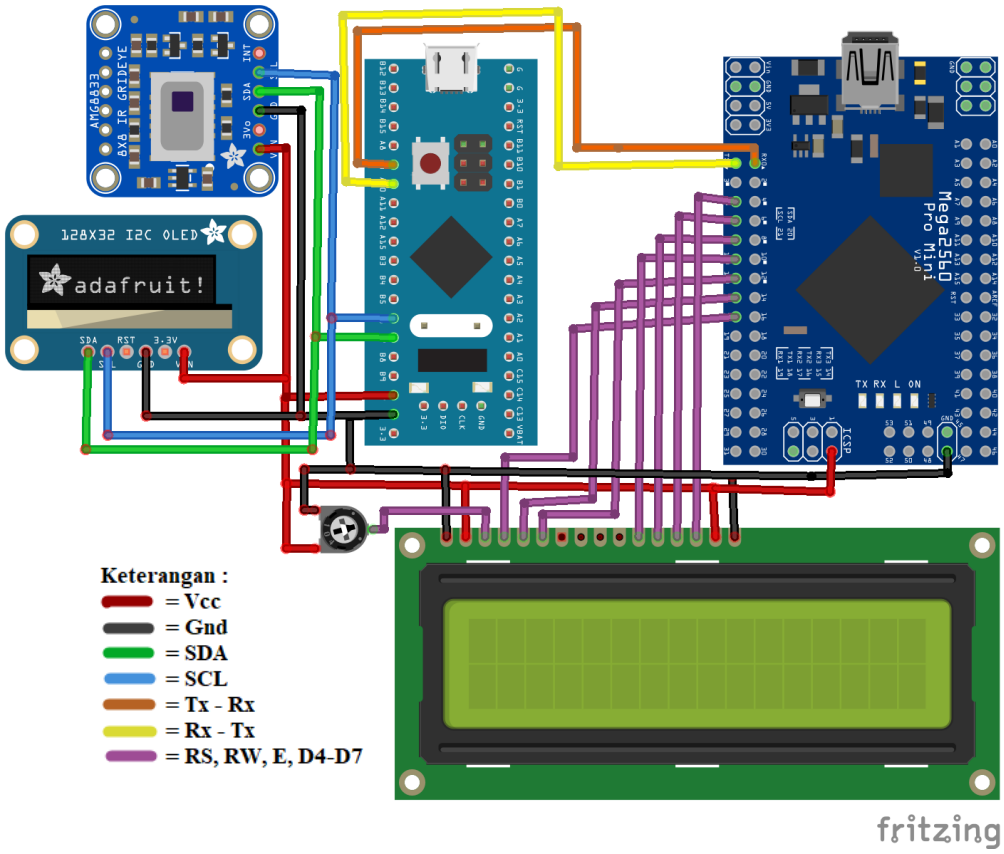

**Gambar 3.2** Blok Diagram Sistem Deteksi Api

Pada penelitian ini dilakukan sebagai salah satu cara untuk memaksimalkan sistem pendeteksi api pada robot MR.COOL MK7. Dari gambar 3.2 rancangan tersebut pada bagian awal adalah sensor AMG8833 yang berfungsi sebagai inputan. AMG8833 akan membaca suhu termal pada setiap *pixel*-nya, hasil pembacaan suhu termal yang didapat diakses dengan mikrokontroler STM32 melalui komunikasi I2C. Pada mikrokontroler STM32 data hasil pembacaan diolah menggunakan metode *bubble sort* dan *weighted average*. Setelah dilakukan pengolahan data maka didapatkan sebuah output nilai posisi titik api. Nilai parameter ini lah kemudian dikirim secara serial menuju mikrokontroler utama pada robot MR.COOL MK7, sehingga robot dapat mengetahui posisi dari titik api.

## 3.5 Perancangan Perangkat Keras

Perancangan perangkat keras merupakan perancangan skematik sistem pendeteksi api yang akan dibuat dengan menggunakan komponen utama berupa sensor AMG8833, *board* mikrokontroler STM32F103C8T6 dan Arduino mega 2560 pro yang merupakan mikrokontroler utama yang berada di robot MR.COOL MK7. Pada pembuatan skematik sistem pendeteksi api ini diperlukan sebuah *software* pendukung. *Software* yang dipakai adalah Fritzing,

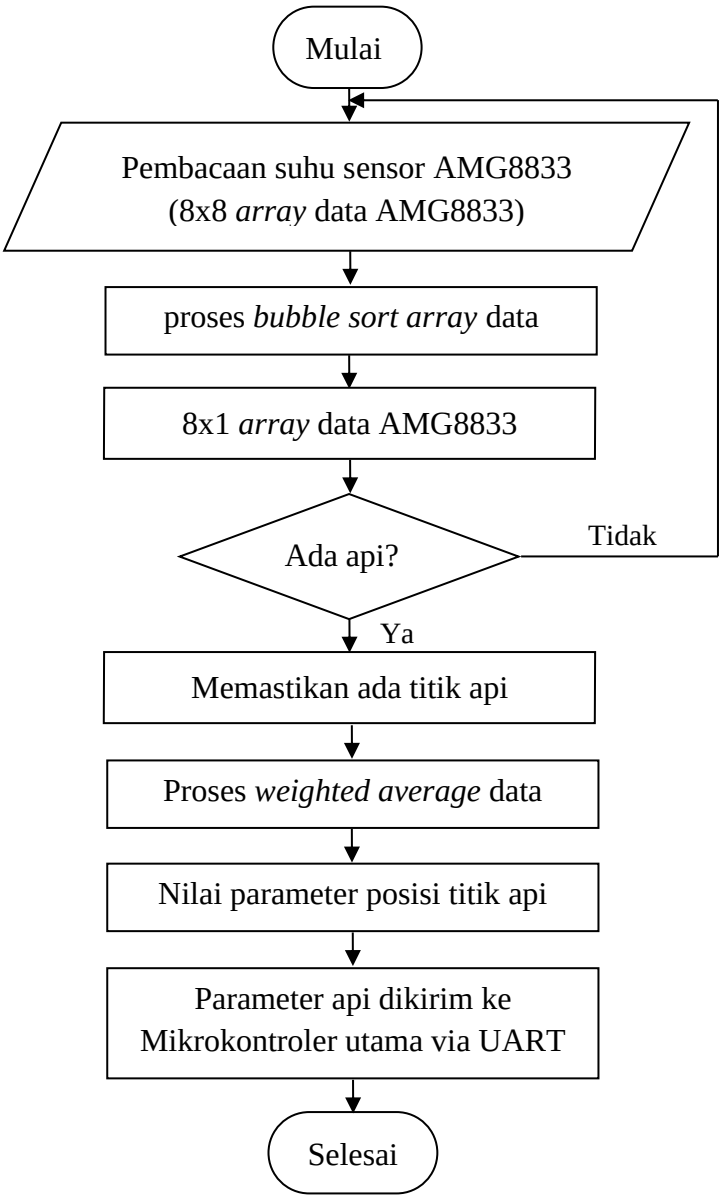
Fritzing merupakan *Software open source* yang berguna untuk pembuatan desain PCB, perancangan skematik, dan dapat mempermudah perancangan *wiring* pada *hardware.* Perancangan skematik sistem pendeteksi posisi api yang akan dibuat ditunjukan pada Gambar 3.3.




**Gambar 3.3** Perancangan Skematik Sistem Pendeteksi Api

## 3.6 Perancangan Perangkat Lunak

Perancangan perangkat lunak merupakan perancangan program mikrokontroler pada *board* STM32F103C8T6 yang berfungsi mengolah data yang didapat dari sensor amg8833 dan perancangan program pada *board* mikrokontroler utama yang berada di robot MR.COOL MK7 yang menerima data sensor hasil pengolahan *board* STM32F103C8T6. Pemrograman dilakukan menggunakan arduino IDE dan Code Vision. *Flowchart* perancangan perangkat lunak dapat dilihat pada gambar 3.4 dihalaman selanjutnya.



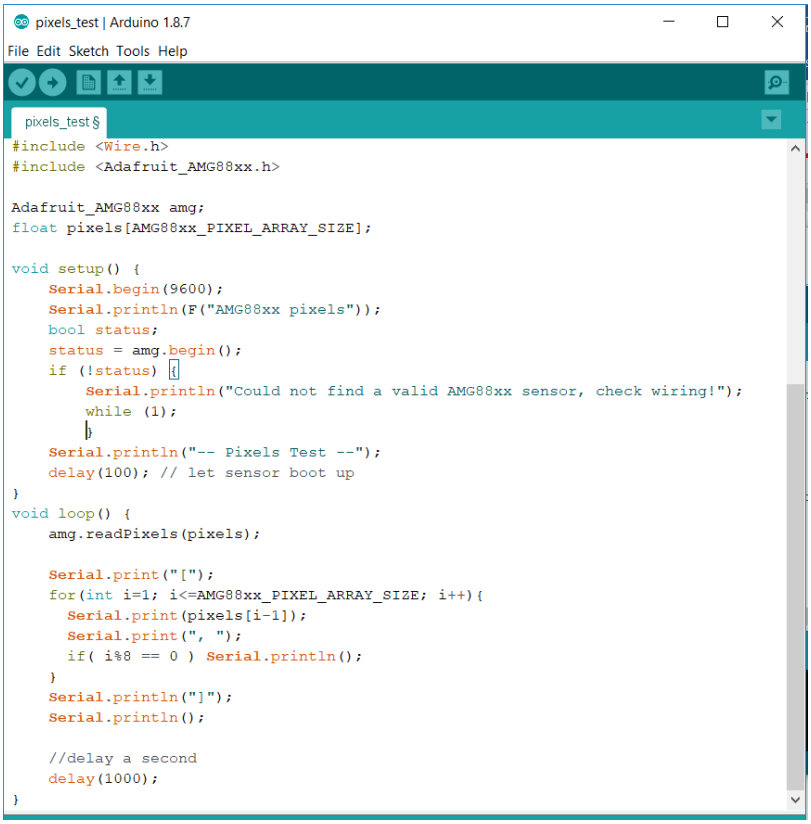

**Gambar 3.4** *Flowchart* Perancangan Perangkat Lunak

Berdasarkan *flowchart* perancangan perangkat lunak pada gambar 3.4, program untuk sistem deteksi posisi api terdiri dari beberapa tahap dari mulai pembacaan suhu dengan sensor AMG8833 sampai didapatkan nilai parameter posisi titik api, untuk lebih jelasnya akan dijelaskan sebagai berikut:

3.6.1 Program Pembacaan Suhu Sensor AMG8833

Program utama pada sistem pendeteksi api ini adalah program pembacaan suhu yang dilakukan oleh sensor AMG8833. Program pembacaan suhu sensor AMG8833 ini menggunakan *library* program

AMG8833 yang sudah *compatible* dengan Arduino IDE yang didapatkan dari internet. Https://Github.Com/Adafruit/Adafruit_AMG88xx Dengan menggunakan Arduino IDE pembacaan suhu sensor AMG8833 cukup menggunakan *file* program *example* yang berjudul '*pixel_test'* yang terdapat pada library 'Adafruit_AMG88xx.h'. Berikut adalah program dari pembacaan suhu sensor AMG8833 menggunakan Arduino IDE, dapat dilihat pada gambar 3.5.




**Gambar 3.5** Program Pembacaan Suhu Sensor AMG8833

Berdasarkan program pada gambar 3.5, untuk memanggil data output hasil pembacaan sensor AMG8833 dapat dilakukan dengan menuliskan kode program seperti pada gambar 3.6.

```
amg.readPixels(pixels);
```

**Gambar 3.6** Kode Program memanggil hasil pembacaan AMG8833

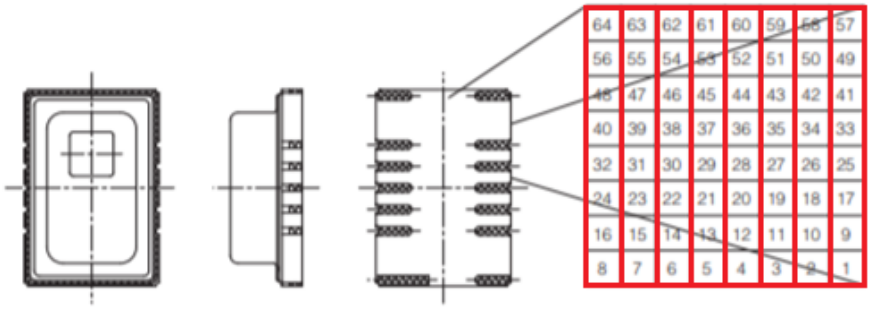
Setelah memanggil output data pembacaan menggunakan kode program seperti pada gambar 3.6 maka sensor AMG8833 akan mengirimkan hasil pembacaan dalam sebuah *array* yang berisi 64 data nilai suhu hasil pembacaan per-*pixel* sensor AMG8833. Data nilai yang didapat dari hasil pembacaan AMG8833 ini sudah dalam satuan °C (derajat celcius).

### 3.6.2 Program Pengolahan Data Hasil Pembacaan AMG8833

Pengolahan data hasil pembacaan AMG8833, perlu dilakukan dengan tujuan utama yaitu untuk mengetahui letak absolut dari posisi titik api dan untuk mempermudah transmisi data via UART menuju mikrokontroler utama yang terletak pada robot MR.COOL MK7. Pengolahan data yang dilakukan dengan 2 metode yaitu menggunakan metode *bubble sort* dan metode *weighted average*. Untuk penjelasan tentang kedua metode tersebut adalah sebagai berikut :

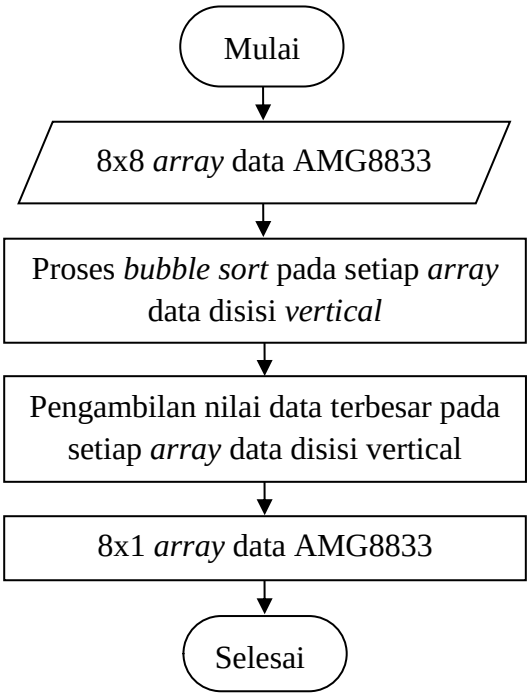
a. Metode *Bubble Sort*

Ketinggian posisi titik api atau lilin yang digunakan pada pertandingan KRPAI berukuran 18 – 22 cm. Sensor AMG8833 mempunyai *pixel* dengan susunan 8 *(pixel horizontal)* x 8 *(pixel vertical).* Penggunaan *Bubble sort* bertujuan untuk menggurutkan data hasil pembacaan pada bagian sisi *vertical*. Dapat dilihat di gambar 3.7 yang dilingkari dengan garis merah adalah nilai *pixel* yang terletak pada sisi *vertical* yang sama.




**Gambar 3.7** *Mapping Pixel* Disisi *Vertical*

Setelah data diurutkan dengan *bubble sort*, maka dapat diketahui nilai pembacaan terbesar disetiap sisi *vertical*. Dengan metode ini dapat meringkas data, dari yang tadinya 64 data menjadi 8 data saja dengan hanya mengambil nilai data terbesar pada setiap *pixel* disisi *vertical* yang sama. Sehingga *array* data yang tadinya 8 *(pixel horizontal)* x 8 *(pixel vertical)* berubah menjadi 8 *(pixel horizontal)* x 1 *(pixel vertical).* Untuk *flowchart* penggunaan metode *bubble sort* dapat dilihat digambar berikut.




**Gambar 3.8** *Flowchart* Penerapan Metode *Bubble Sort*

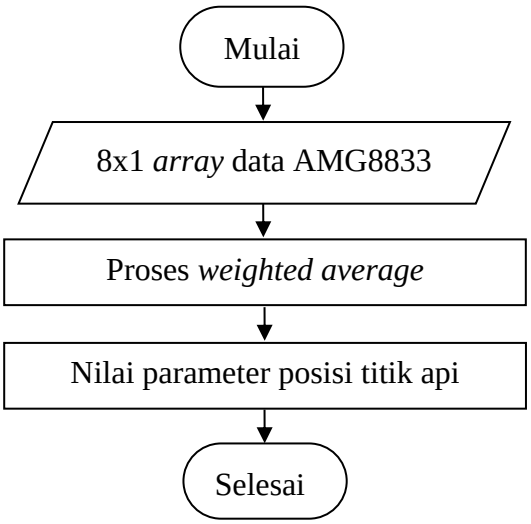
Kode program pada penggunaan metode *bubble sort* ini dibuat dalam bentuk 2 buah fungsi. Fungsi yang pertama adalah fungsi dari metode *bubble sort* itu sendiri yang berrtugas untuk mengurutkan atau untuk *sorting* data. Kemudian, fungsi yang kedua merupakan fungsi yang bertugas untuk mengelompokan *array* data input sebelum dilakukan *sorting* data dan bertugas untuk memilih data yang terbesar setelah *sorting* data selesai. Untuk kode program lengkapnya dapat dilihat pada gambar 3.9 berikut.

```
void sort(int s[], int size) {
    for(int r=0; r<(size-1); r++) {
        for(int o=0; o<(size-(r+1)); o++) {
                if(s[o] > s[o+1]) {
                    int t = s[o];
                    s[o] = s[o+1];
                    s[o+1] = t;
                }
        }
    }
}
void bubble_Sort(){

    int sortValues1[8] = { pixel[0], pixel[8], pixel[16], pixel[24], pixel[32], pixel[40], pixel[48], pixel[56]};
    int sortValues2[8] = { pixel[1], pixel[9], pixel[17], pixel[25], pixel[33], pixel[41], pixel[49], pixel[57]};
    int sortValues3[8] = { pixel[2], pixel[10], pixel[18], pixel[26], pixel[34], pixel[42], pixel[50], pixel[58]};
    int sortValues4[8] = { pixel[3], pixel[11], pixel[19], pixel[27], pixel[35], pixel[43], pixel[51], pixel[59]};
    int sortValues5[8] = { pixel[4], pixel[12], pixel[20], pixel[28], pixel[36], pixel[44], pixel[52], pixel[60]};
    int sortValues6[8] = { pixel[5], pixel[13], pixel[21], pixel[29], pixel[37], pixel[45], pixel[53], pixel[61]};
    int sortValues7[8] = { pixel[6], pixel[14], pixel[22], pixel[30], pixel[38], pixel[46], pixel[54], pixel[62]};
    int sortValues8[8] = { pixel[7], pixel[15], pixel[23], pixel[31], pixel[39], pixel[47], pixel[55], pixel[63]};

    sort(sortValues1,8);
    sort(sortValues2,8);
    sort(sortValues3,8);
    sort(sortValues4,8);
    sort(sortValues5,8);
    sort(sortValues6,8);
    sort(sortValues7,8);
    sort(sortValues8,8);

    sort_pixel[0] = sortValues1[7];
    sort_pixel[1] = sortValues2[7];
    sort_pixel[2] = sortValues3[7];
    sort_pixel[3] = sortValues4[7];
    sort_pixel[4] = sortValues5[7];
    sort_pixel[5] = sortValues6[7];
    sort_pixel[6] = sortValues7[7];
    sort_pixel[7] = sortValues8[7];

}
```

**Gambar 3.9** Kode Program Penerapan Metode *Bubble Sort*

b. Metode *Weighted Average*

Data yang diperoleh setelah diproses menggunakan metode *bubble sort* masih berupa *array* data 8x1, supaya robot lebih mudah mengetahui posisi titik api sekaligus memudahkan pengiriman data via UART ke mikrokontroler utama maka 8x1 *array* data hasil dari proses *bubble sort* diberi nilai pembobotan masing – masing dengan metode *weighted average*. Nilai pembobotan yang diberikan pada setiap nilai data berbeda – beda besarnya antara data satu dengan data lainnya, hal ini supaya nanti output nilainya sesuai dengan yang diharapkan. Untuk *flowchart* penerapan metode *weighted average* dapat dilihat digambar 3.10 berikut.




**Gambar 3.10** *Flowchart* Penerapan Metode *Weighted Average*

Berdasarkan *flowchart* penerapan metode *weighted average* pada gambar 3.10, 8x1 *array* data kemudian diberi pembobotan dengan menggunakan metode *weighted average* akan menghasilkan sebuah nilai yang akan dijadikan sebuah parameter untuk menentukan posisi dari titik api. Kode program untuk penerapan metode ini dapat dillihat pada gambar 3.11 dibawah.

```
t_bobot=(long)((long)0*sort_pixel[0]+(long)25*sort_pixel[1]
+(long)50*sort_pixel[2]+(long)75*sort_pixel[3]+(long)125*sort_pixel[4]
+(long)150*sort_pixel[5]+(long)175*sort_pixel[6]+(long)200*sort_pixel[7]);

t_data =(long)sort_pixel[0]+sort_pixel[1]+sort_pixel[2]+sort_pixel[3]
+sort_pixel[4]+sort_pixel[5]+sort_pixel[6]+sort_pixel[7];

pos_Api=t_bobot/t_data;
```

**Gambar 3.11** Kode Program Penerapan Metode *Weighted Average*

### 3.6.3 Program memastikan ada titik api

Pada saat salah satu *pixel* mendeteksi titik api pada jarak yang cukup jauh, selisih antara hasil pembacaan *pixel* yang mendeteksi titik api dengan *pixel* yang tidak mendeteksi titik api terlalu sedikit. apabila *array* data *pixel* tersebut langsung diproses ke metode *weighted average*, maka selisih nilai posisi api saat didepan dengan disisi serong kiri atau kanan terlalu sedikit. Sehingga untuk mengatasi permasalahan

tersebut dibuat sebuah program untuk memastikan titik api. Program berupa suatu percabangan dengan logika persyaratan *pixel* yang mendeteksi titik api dan lebih besar dari *pixel* lain, maka nilai hasil pembacaan *pixel* tersebut ditambah dengan nilai 100. Untuk lebih jelasnya dapat dilihat pada gambar 3.12 kode program dibawah.

```
void Call_Api(){

  if ((sort_pixel[0]>sort_pixel[2])&&(sort_pixel[0]>sort_pixel[3])&&(sort_pixel[0]>sort_pixel[4])&&
     (sort_pixel[0]>sort_pixel[5])&&(sort_pixel[0]>sort_pixel[6])&&(sort_pixel[0]>sort_pixel[7])&&
     (sort_pixel[0]>=c_Api))sort_pixel[0]=sort_pixel[0]+100;
  if ((sort_pixel[1]>sort_pixel[3])&&(sort_pixel[1]>sort_pixel[4])&&(sort_pixel[1]>sort_pixel[5])&&
     (sort_pixel[1]>sort_pixel[6])&&(sort_pixel[1]>sort_pixel[7])&&(sort_pixel[1]>=c_Api))
     sort_pixel[1]=sort_pixel[1]+100;
  if ((sort_pixel[2]>sort_pixel[0])&&(sort_pixel[2]>sort_pixel[4])&&(sort_pixel[2]>sort_pixel[5])&&
     (sort_pixel[2]>sort_pixel[6])&&(sort_pixel[2]>sort_pixel[7])&&(sort_pixel[2]>=c_Api))
     sort_pixel[2]=sort_pixel[2]+100;
  if ((sort_pixel[3]>sort_pixel[0])&&(sort_pixel[3]>sort_pixel[1])&&(sort_pixel[3]>sort_pixel[5])&&
     (sort_pixel[3]>sort_pixel[6])&&(sort_pixel[3]>sort_pixel[7])&&(sort_pixel[3]>=c_Api))
     sort_pixel[3]=sort_pixel[3]+100;
  if ((sort_pixel[4]>sort_pixel[0])&&(sort_pixel[4]>sort_pixel[1])&&(sort_pixel[4]>sort_pixel[2])&&
     (sort_pixel[4]>sort_pixel[6])&&(sort_pixel[4]>sort_pixel[7])&&(sort_pixel[4]>=c_Api))
     sort_pixel[4]=sort_pixel[4]+100;
  if ((sort_pixel[5]>sort_pixel[0])&&(sort_pixel[5]>sort_pixel[1])&&(sort_pixel[5]>sort_pixel[2])&&
     (sort_pixel[5]>sort_pixel[3])&&(sort_pixel[5]>sort_pixel[7])&&(sort_pixel[5]>=c_Api))
     sort_pixel[5]=sort_pixel[5]+100;
  if ((sort_pixel[6]>sort_pixel[0])&&(sort_pixel[6]>sort_pixel[1])&&(sort_pixel[6]>sort_pixel[2])&&
     (sort_pixel[6]>sort_pixel[3])&&(sort_pixel[6]>sort_pixel[4])&&(sort_pixel[6]>=c_Api))
     sort_pixel[6]=sort_pixel[6]+100;
  if ((sort_pixel[7]>sort_pixel[0])&&(sort_pixel[7]>sort_pixel[1])&&(sort_pixel[7]>sort_pixel[2])&&
     (sort_pixel[7]>sort_pixel[3])&&(sort_pixel[7]>sort_pixel[4])&&(sort_pixel[7]>sort_pixel[5])&&
     (sort_pixel[7]>=c_Api))sort_pixel[7]=sort_pixel[7]+100;

}
```

**Gambar 3.12** Program Memastikan Titik Api

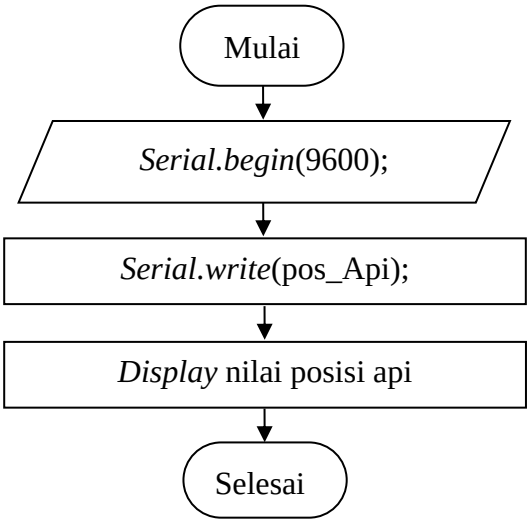
### 3.6.4 Program Komunikasi Data Via UART

Program komunikasi data Via UART merupakan program pengiriman data parameter nilai posisi titik api dari *board* mikrokontroler STM32F103C8T6 menuju ke mikrokontroler utama yang terdapat pada robot MR.COOL MK7. Program komunikasi via UART dibagi menjadi 2, yaitu program *transmitter* (pengirim) dan program *receiver* (penerima).

a. Program *Transmitter*

Program *transmitter* pada sistem deteksi ini diprogram menggunakan Arduino IDE. *Transmitter* akan selalu mengirimkan data posisi titik api baik saat sistem mendeteksi api maupun saat sistem tidak mendeteksi api. Program *transmitter* diawali dengan

*Serial.begin*(9600); yang menandai bahwa komunikasi dimulai dan dengan menggunakan *baudrate* 9600. Selanjutnya, data posisi titik api dikirim melalui perintah *Serial.write*(pos_Api); data akan dikirim terus – menerus ke *receiver* atau ke robot MR.COOL MK7, sehingga robot selalu mengetahui posisi lilin secara *realtime.* Untuk lebih jelasnya dapat dilihat pada *flowchart* gambar 3.13 berikut.

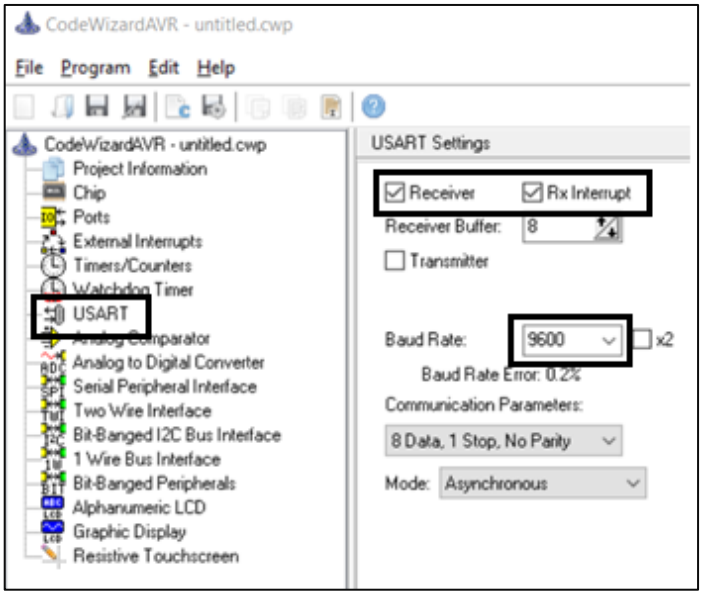



**Gambar 3.13** *Flowchart* Program *Transmitter*

b. Program *Receiver*

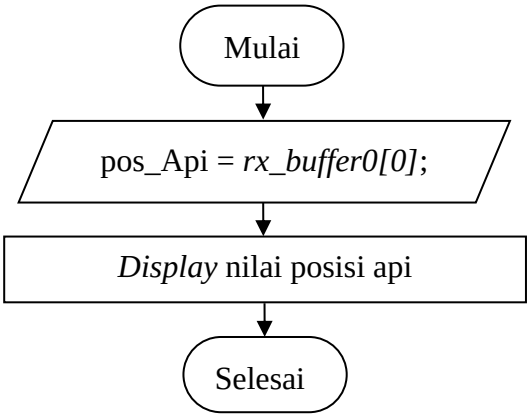
Program *receiver* pada sistem deteksi ini diprogram menggunakan Code Vision AVR, penggunaan Code Vision AVR dikarenakan penyesuaian terhadap mikrokontroler utama robot MR.COOL MK7 yang keseluruhan programnya sudah terlebih dahulu diprogram menggunakan Code Vision AVR. *Receiver* akan selalu menerima data posisi titik api baik saat sistem mendeteksi api maupun saat sistem tidak mendeteksi api. Program *Receiver* diawali dengan melakukan *setting* USART pada CodeWizardAVR. *Setting* dilakukan pada menu bar USART, dengan memberikan centang pada pilihan *receiver* dan *rx interrupt*. Selanjutnya pada bagian *Baudrate* disetting 9600, nilai ini menyesuaikan dengan nilai

*baudrate* yang digunakan pada program *transmitter*. *Setting* USART untuk lebih jelasnya dapat melihat pada gambar 3.14.




**Gambar 3.14** *Setting* USART *receiver* CodeWizardAVR

Data posisi titik api yang dikirim oleh *transmitter* akan selalu diterima dan kemudian dimasukan pada *variable* pos_Api menggunakan perintah pos_Api = rx_buffer0[0]; untuk lebih jelasnya dapat dilihat pada *flowchart* gambar 3.15 berikut.

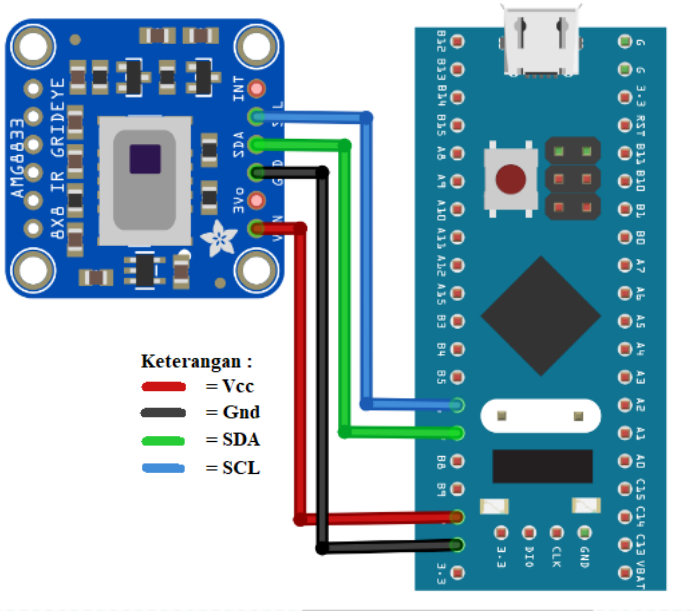



**Gambar 3.15** *Flowchart* Program *Receiver*

### 3.7 Perlakuan Pengujian Sensor AMG8833 dan *Board* STM32F103C8T6

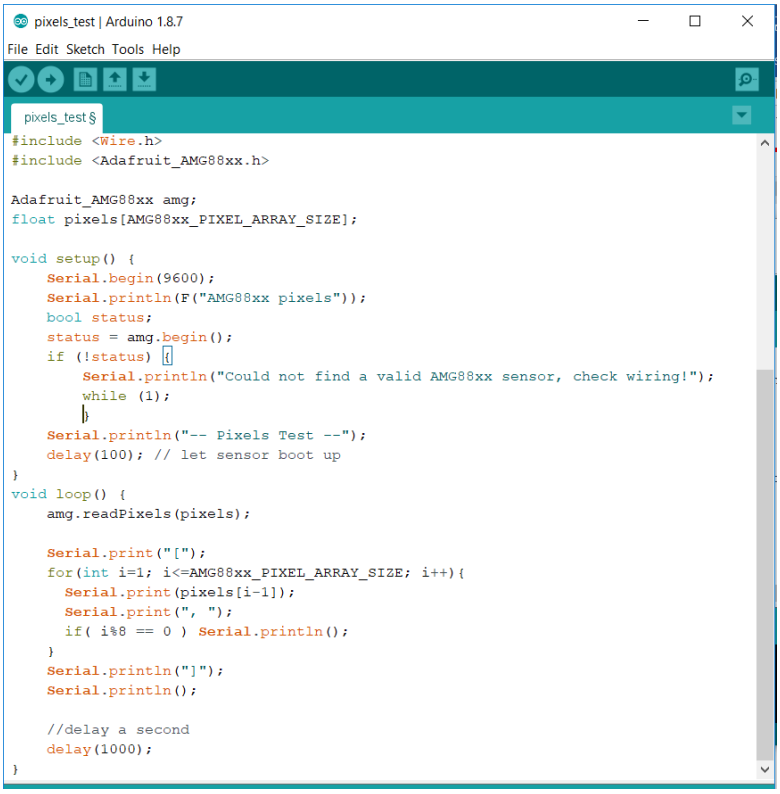
Tahap pengujian diperlukan untuk menguji apakah sensor AMG8833 dan *board* STM32F103C8T6 dapat berkerja sesuai

fungsionalnya. Selain itu, pengujian ini juga dilakukan pada 2 buah tingkat pencahayaan yang berbeda yaitu pada tingkat pencahayaan standart dan tingkat pencahayaan lebih terang. Pengujian tersebut bertujuan untuk mengetahui apakah nilai pembacaan sensor terpengaruh terhadap gelap – terangnya cahaya atau tidak. *Wiring diagram* untuk pengujian ini dapat dilihat pada gambar 3.16 berikut.




**Gambar 3.16** *Wiring Diagram* Pengujian AMG8833 dan STM32F103C8T6

Program yang digunakan untuk pengujian sensor AMG8833 dan *board* STM32F103C8T6 adalah program pixel_test yang terdapat pada *examples* dari *library* Adafruit AMG88xx Library. program pixel_test di*upload* ke *board* STM32F103C8T6 menggunakan *software* Arduino IDE. Berikut adalah kode program pixel_test untuk pengujian sensor AMG8833 dan *board* STM32F103C8T6 yang ditunjukkan pada gambar 3.17.




**Gambar 3.17** Program Pengujian AMG8833 dan STM32F103C8T6

## 3.8 Perlakuan Pengujian Program Pengolah Data

### 3.8.1 Perlakuan Pengujian Program Metode *Bubble Sort*

Pengujian program pengolah data dengan metode *bubble sort* dilalukan dengan tujuan untuk mengetahui apakah program atau perangkat lunak yang sudah dibuat dapat berfungsi sesuai yang diinginkan atau tidak.

Pengujian dilakukan dengan memberikan input pada program *bubble sort* yang telah dibuat, kemudian hasil *bubble sort* akan langsung keluar dan ditampilkan pada serial monitor. Ketentuan input untuk pengujian ini adalah input harus berupa *array* data yang berisi 8 buah nilai angka yang besar – kecil nilai angkanya boleh bervariasi.

Pengujian program metode *bubble sort* ini dilakukan dengan menggunakan Arduino IDE. Kode program pada pengujian ini dapat dilihat pada gambar 3.18 berikut.

```
int sortValues1[8] = { 25, 1, 10, 3, 11, 17, 12, 13 };
void setup(){
  Serial.begin(9600);
}
void loop() {
  sort(sortValues1,8);
  Serial.print("Output Array: ");
  for(int r=0; r<8; r++) {
     Serial.print(sortValues1[r]);
     Serial.print(",");}
  Serial.println();
  delay(1000);
}
void sort(int s[], int size) {
    for(int r=0; r<(size-1); r++) {
        for(int o=0; o<(size-(r+1)); o++) {
                if(s[o] > s[o+1]) {
                    int t = s[o];
                    s[o] = s[o+1];
                    s[o+1] = t;
                }
```

**Gambar 3.18** Program Pengujian Metode *Bubble Sort*

### 3.8.2 Perlakuan Pengujian Program Metode *Weighted Average*

Pengujian pengolah data dengan metode *weighted average* diperlukan agar dapat diketahui apakah hasil perhitungan program atau perangkat lunak yang telah dibuat, sesuai dengan perhitungan manual atau tidak.

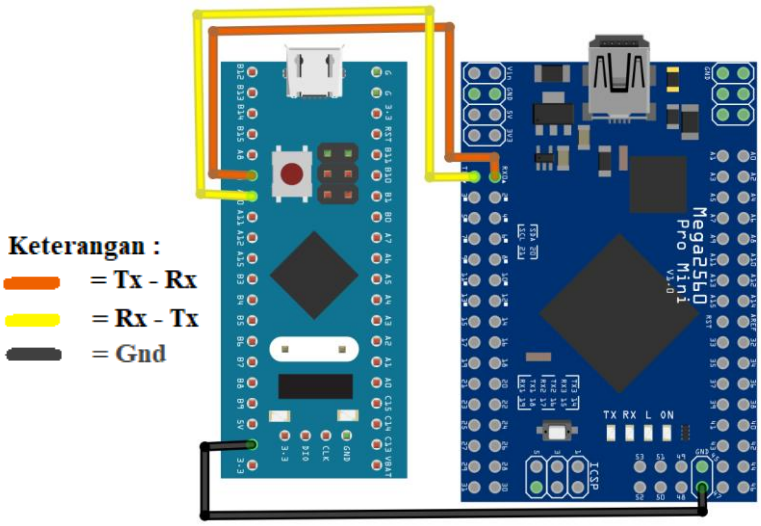
Pengujian ini dilakukan menggunakan Arduino, kemudian hasil dari pengujian akan dibandingkan dengan perhitungan manual yang telah hitung menggunakan persamaan yang sudah ada. Kode program untuk pengujian program *weighted average* ini dapat dilihat pada gambar 3.19 berikut.

```
long t_bobot, t_data, pos_Api;
int sort_pixel[8] = { 200, 28, 29, 30, 29, 28, 27, 28};
void setup() {
  // put your setup code here, to run once:
    Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
    t_bobot=(long)((long)0*sort_pixel[0]+(long)25*sort_pixel[1]
    +(long)50*sort_pixel[2]+(long)75*sort_pixel[3]+(long)125*sort_pixel[4]
    +(long)150*sort_pixel[5]+(long)175*sort_pixel[6]+(long)200*sort_pixel[7]);
    t_data =(long)sort_pixel[0]+sort_pixel[1]+sort_pixel[2]+sort_pixel[3]
    +sort_pixel[4]+sort_pixel[5]+sort_pixel[6]+sort_pixel[7];
    pos_Api=t_bobot/t_data;
    Serial.print("output weighted average =");
    Serial.print(pos_Api);Serial.println();
    delay(1000);
}
```

**Gambar 3.19** Program Pengujian Metode *weighted average*

### 3.9 Perlakuan Pengujian Transmisi Data Melalui Protokol UART

Pengujian transmisi data melalui protokol UART dilakukan dengan tujuan untuk mengetahui apakah ada data yang *error* saat transmisi atau tidak. Pengujian transmisi data dilakukan dengan menggunakan STM32F103C8T6 sebagai *transmitter* dan Arduino mega 2560 pro sebagai *receiver* yang akan menerima data yang dikirim. *wiring diagram* pengujian transmisi data melalui protokol UART dapat dilihat pada gambar 3.20 berikut.




**Gambar 3.20** *Wiring Diagram* Pengujian Transmisi Data

Pada pengujian transmisi data ini, STM32F103C8T6 sebagai *transmitter* akan mengirimkan data angka dari 1 sampai 255 secara berurutan menuju Arduino mega 2560 pro. *Software* pemrograman yang digunakan untuk memprogram STM32F103C8T6 yang berfungsi sebagai pengirim data atau *transmitter* diprogram menggunakan Arduino IDE. Kode program yang digunakan pada bagian *transmitter* untuk pengujian transmisi data ini adalah sebagai berikut:

```
int a=0;

void setup() {
  Serial.begin(9600);
  Serial1.begin(9600);
}
void loop() {
  if (a <= 255) {
   Serial1.write(a);
   a++;
  }
}
```

**Gambar 3.21** Program Transmisi Data Bagian *Transmitter*

Berbeda dengan bagian *transmitter* yang di program menggunakan *software* Arduino IDE. pada bagian *receiver*, *board* Arduino mega 2560 pro diprogram menggunakan *software* code vision AVR. penggunaan code vision AVR dikarenakan mikrokontroler utama robot MR.COOL MK7 yang keseluruhan programnya sudah terlebih dahulu diprogram menggunakan code vision AVR. Kode program untuk *receiver* yang digunakan untuk pengujian transmisi data ini adalah sebagai berikut:

```
#include <mega328p.h>
#include <stdio.h>
#include <delay.h>

// Alphanumeric LCD functions
#include <alcd.h>

// Declare your global variables here

int pos_Api;
// USART Receiver interrupt service routine
interrupt [USART_RXC] void usart_rx_isr(void)
{
char status,data;
status=UCSR0A;
data=UDR0;
if ((status & (FRAMING_ERROR | PARITY_ERROR | DATA_OVERRUN))==0)
   {
   rx_buffer0[rx_wr_index0++]=data;
   pos_Api = rx_buffer0[0];
#if RX_BUFFER_SIZE0 == 256
   // special case for receiver buffer size=256
   if (++rx_counter0 == 0) rx_buffer_overflow0=1;
#else
   if (rx_wr_index0 == RX_BUFFER_SIZE0) rx_wr_index0=0;
   if (++rx_counter0 == RX_BUFFER_SIZE0)
      {
      rx_counter0=0;
      rx_buffer_overflow0=1;
      }
#endif
   }
}
```

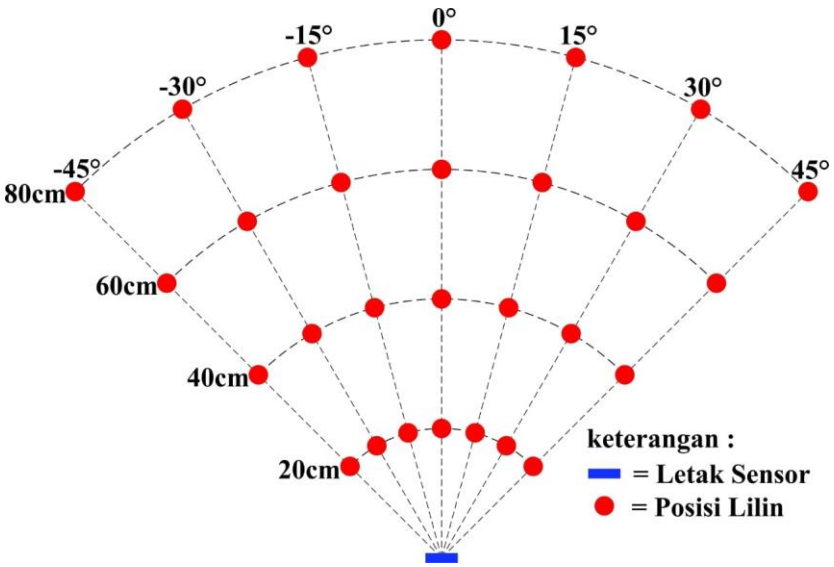
**Gambar 3.22** Program Transmisi Data Bagian *Receiver*

### 3.10 Perlakuan Pengujian dan Analisis Sistem Deteksi Posisi Titik Api

Pengujian sistem deteksi posisi titik api merupakan pengujian untuk mengetahui kemampuan sistem pendeteksi ini dalam mendeteksi posisi titik api baik terhadap rentang sudut pembacaan serta terhadap jarak pembacaan. Selain untuk mengetahui kemampuan sistem deteksi posisi titik api, pengujian ini juga dilakukan untuk mendapatkan acuan parameter nilai untuk menentukan posisi titik api.

Dalam pengujian ini yang dibutuhkan adalah sistem deteksi yang telah dibuat dengan komponen berupa sensor AMG8833, mikrokontroler STM32F103C8T6 dan mikrokontroler Arduino mega 2560 pro. Selain itu, juga dibutuhkan busur derajat serta penggaris untuk mengukur sudut dan jarak peletakan titik api.

Pengujian sistem deteksi posisi titik api ini dilakukan dengan meletakan lilin atau titik api secara bergantian pada sudut dan jarak yang berbeda – beda terhadap sistem deteksi. Untuk *visualisasi* posisi pengujian dapat dilihat pada gambar 3.23. Parameter nilai yang dihasilkan dari setiap pengujian ini, akan dijadikan nilai acuan untuk menentukan posisi titik api.




**Gambar 3.23** *Visualisasi* Pengujian Sistem Deteksi Posisi Api Tampak Atas