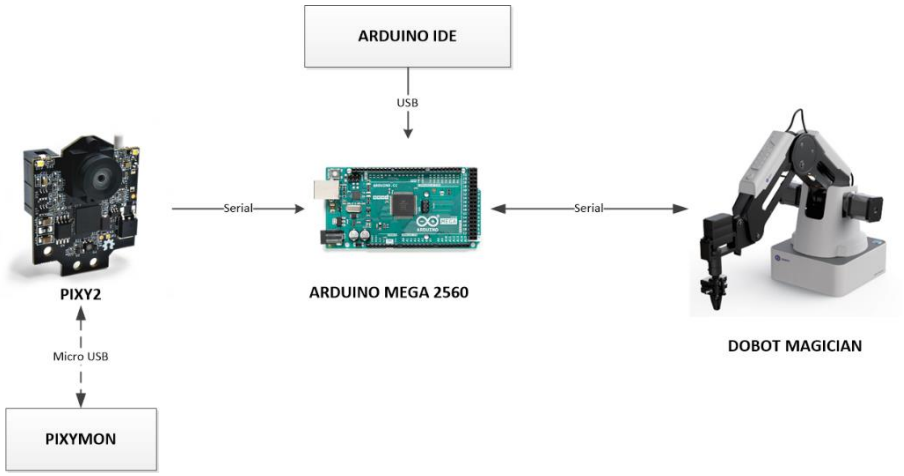
3. PERENCANAAN SISTEM

Pada penelitian ini akan dibuat sistem untuk Dobot Magician yang bertujuan untuk melakukan proses *pick and place* objek dan mengelompokkannya sesuai dengan warna tiap objek. Pengenalan warna objek akan dilakukan dengan bantuan *vision camera* yaitu Pixy2. Penjelasan selanjutnya dapat dilihat pada blok diagram yang ada pada Gambar 3.1.



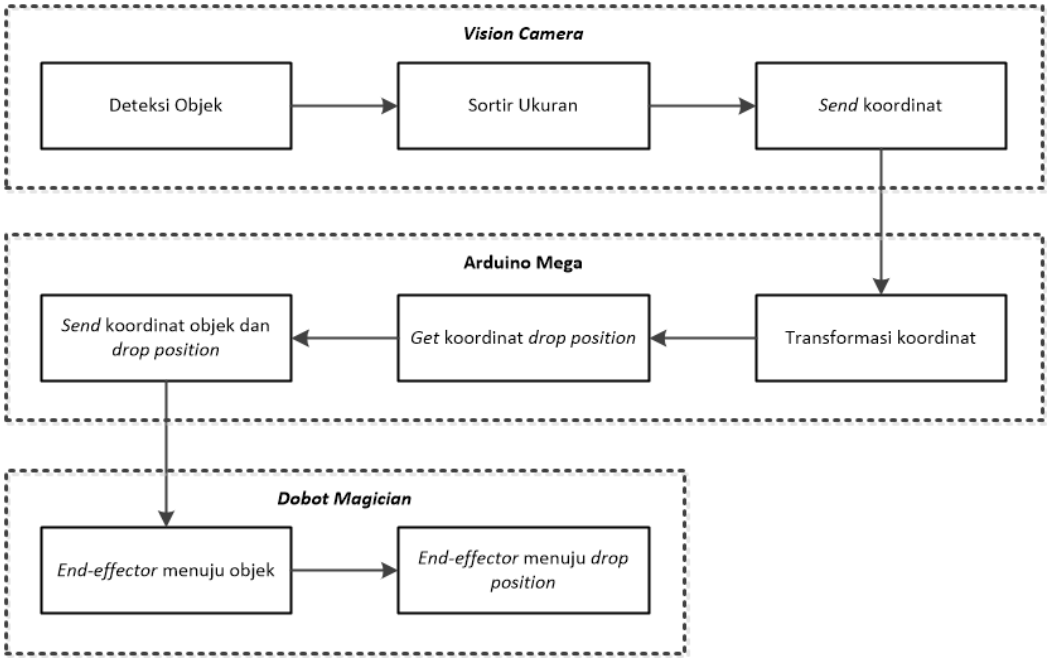
Gambar 3.1 Blok diagram *hardware* sistem penyortiran warna dengan menggunakan *vision camera* Pixy2

Pada Gambar 3.1, dapat dilihat blok diagram sistem penyortiran objek berdasarkan warna ini. Pemrograman Arduino akan dibuat dengan menggunakan *software* Arduino IDE. Setelah selesai program akan di-*upload* pada Arduino Mega Pro. Komunikasi antara Dobot Magician dan Arduino Mega akan menggunakan koneksi serial seperti pada potongan program dibawah. Program ini berfungsi sebagai tempat penyimpanan data sementara untuk seluruh data yang dikirim dan diterima melalui *port* serial. Pada *buffer* untuk TX, semakin besar nilai *buffernya* semakin banyak data yang disimpan sebelum dikirim. Pada RX, *buffer* yang besar dapat menyimpan data yang diterima.

```
#define SERIAL_TX_BUFFER_SIZE 64
#define SERIAL_RX_BUFFER_SIZE 256
```

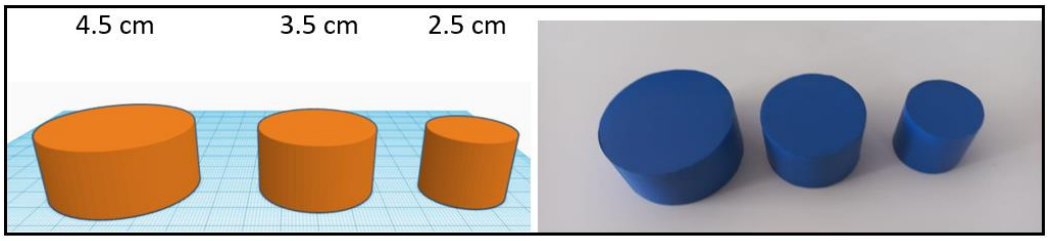
Gambar 3.2 Koneksi serial antara Dobot Magician dan Arduino Mega

Arduino Mega Pro merupakan komponen utama dalam sistem ini karena Arduino Mega Pro akan terhubung pada Dobot Magician dan *vision camera* Pixy2 yang bertugas menangkap dan mengidentifikasi warna pada objek. Kedua alat ini akan terhubung dengan Arduino Mega Pro dengan menggunakan koneksi serial. Pada sistem ini dapat juga ditambahkan *software* PixyMon yang dibuat khusus agar pengguna dapat melihat gambar yang ditangkap kamera. Pada sistem ini, semua pemrograman Dobot Magician akan menggunakan Arduino IDE sehingga tidak menggunakan aplikasi seperti DobotStudio atau DobotLab.



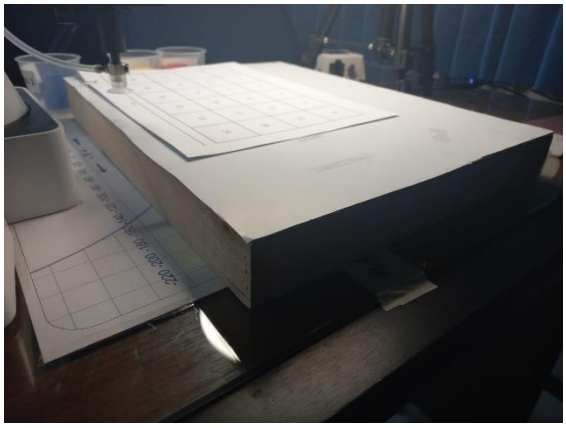
Gambar 3.3 Blok diagram sistem *pick and place* menggunakan *vision camera*

Pada Gambar 3.3 merupakan blok diagram alur dari proses yang terjadi pada saat sistem *pick and place* dijalankan. Pertama, *vision camera* Pixy2 akan mendeteksi objek dan mendapatkan nomor *signatura* dan koordinat semua objek. Setelah itu, Pixy2 akan menyortir objek berdasarkan ukuran, hal ini dilakukan dengan mengalikan panjang dan lebar dari objek. Koordinat objek pertama yaitu objek terkecil kemudian akan dikirimkan ke Arduino Mega. Pada Arduino Mega akan dilakukan transformasi koordinat objek agar sesuai dengan sistem koordinat Dobot Magician. Objek kemudian akan mendapatkan koordinat untuk *drop position* sesuai dengan nomor *signature* dari Pixy2. Semua koordinat kemudian akan dikirimkan ke Dobot Magician untuk dapat menggerakkan lengan robot tersebut.



Gambar 3.4 Objek

Pada Gambar 3.4 di atas merupakan objek yang digunakan pada saat implementasi sistem. Pada saat pengujian sistem, jumlah warna objek yang akan digunakan akan berupa 3 warna berbeda. Terdapat tiga ukuran objek yang akan digunakan dalam penelitian ini. Objek akan berupa barang berbentuk tabung dengan diameter sebesar 2.5, 3.5, dan 4.5 cm. Semua objek akan memiliki tinggi yang sama yaitu 2 cm.



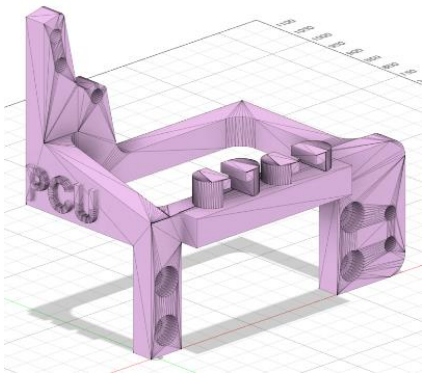
Gambar 3.5 Alas Objek

Pada penelitian ini, terdapat juga alas yang digunakan untuk memperluas jangkauan Dobot Magician. Alas yang digunakan memiliki tinggi sebesar 4 cm dan dapat dilihat pada Gambar 3.5. Dobot Magician memiliki batas pergerakan untuk setiap sendi robotnya. Jika objek berada di luar jangkauan gerak sendi, robot akan berhenti bekerja untuk mencegah kerusakan pada mekanismenya.

3.1 Mounting Kamera Pixy2

Gambar 3.6 Letak *mounting* Pixy2

Dalam mengintegrasikan kamera Pixy2 pada Dobot Magician, diperlukan penggunaan *mounting* agar sistem dapat bekerja dengan baik. Peletakan *mounting* dapat dilihat seperti pada gamber 3.6. *Mounting* ini bertujuan untuk mengamankan kamera pada posisi yang diinginkan. Dengan menggunakan *mounting* yang dirancang khusus untuk Dobot Magician, Pixy2 dapat menangkap gambar objek dengan akurat. Posisi kamera Pixy2 akan diletakkan dekat dengan *end effector* dari lengan robot, dan berada pada posisi yang tidak terhalangi *gripper* atau *suction cup.* Pada Gambar 3.7 merupakan *mounting* yang digunakan dalam pelaksanaan sistem.

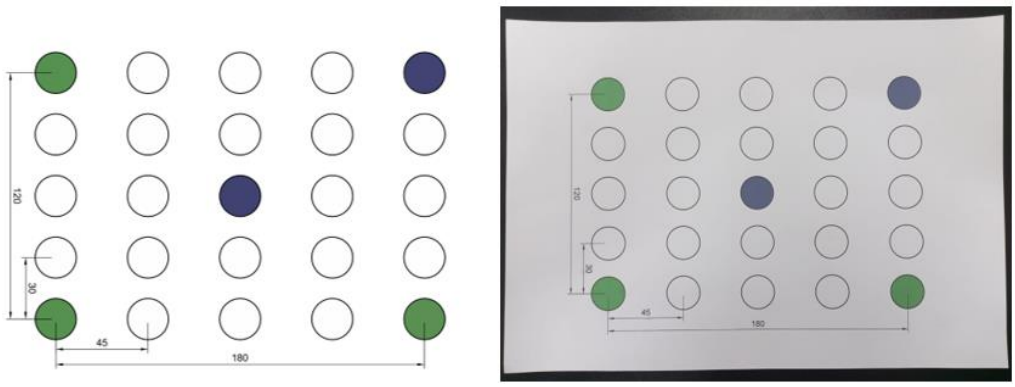


Gambar 3.7 *Mounting* Pixy2 pada Dobot Magician

3.2 Kalibrasi

Sistem koordinat yang dimiliki oleh Dobot Magician berbeda dengan sistem koordinat yang dimiliki Pixy2 maka diperlukan adanya proses kalibrasi. Kalibrasi ini bertujuan agar koordinat bidang yang ditangkap kamera dapat ditransformasikan menjadi sesuai dengan

koordinat pada Dobot Magician. Tanpa ada proses kalibrasi, Dobot Magician tidak dapat menyesuaikan titik koordinat objek dengan benar, hal ini akan menyebabkan kesalahan pada pergerakan lengan robot.



Gambar 3.8 Kertas kalibrasi

Sumber: Robin. (2019). *How to Make a Vision System for the Dobot Magician with the Pixy 2 Camera. https://uptimefab.com/2019/10/02/how-to-make-a-robot-vision-system-with-the-pixy-2-camera.*

Pada proses kalibrasi ini akan menggunakan kertas kalibrasi yang terdiri dari tiga titik. Kertas kalibrasi ini dapat dilihat pada Gambar 3.8. Tujuan dari tiga titik ini adalah untuk menentukan batas-batas bidang yang akan digunakan pada sistem. Titik-titik koordinat ini akan menjadi referensi dari titik awal, sumbu x, dan sumbu y dari bidang. Kamera Pixy2 akan mengidentifikasi tiga titik ini dan akan mengirimkannya pada Arduino. Kemudian pada Arduino, akan terdapat program untuk mentransformasikan koordinat tiga titik ini agar sesuai dengan sistem koordinat pada Dobot Magician. Perhitungan dapat dilihat pada bagian di bawah ini (Robin, 2019).

$$pixyPartY = (207 - pixyPartY) \quad (3.1)$$

Pertama, bidang akan dibalik secara Y-Axis karena pada Pixy2 bagian kanan atas merupakan titik awal Y = 0 dan semakin kebawah nilai Y akan bertambah. Pada Dobot Magician, titik awal Y = 0 berada pada kanan bawah dan semakin ke atas nilai Y akan bertambah. Maka dari itu, bidang akan dibalik dengan cara menguranginya dari jumlah pixel maksimum dari sumbu Y, yaitu sebanyak 207 pixel seperti yang terlihat pada potongan program pada persamaan (3.1). Tahap selanjutnya adalah melakukan scaling dan transformasi koordinat objek. Berikut dibawah ini merupakan persamaan yang digunakan pada proses kalibrasi.

Untuk menghitung *scaling* antara garis X dan Y pertama, akan dicari jarak dari garis X dari sistem koordinat Dobot Magician dan Pixy2. Setelah itu, sumbu X dari sistem koordinat Dobot Magician akan dibagi dengan sumbu X dari sistem koordinat Pixy2 untuk mendapatkan *scaling* sumbu X. Begitu pula untuk menghitung *scaling* sumbu Y. Perhitungan dapat dilihat pada persamaan 3.2 dan 3.3 di bawah ini.

$$scaleX = \frac{\sqrt{((b2X-a2X)^2-(b2Y-a2Y)^2)}}{\sqrt{((b1X-a1X)^2-(b1Y-a1Y)^2)}} \quad (3.2)$$

$$scaleY = \frac{\sqrt{((c2X-a2X)^2-(c2Y-a2Y)^2)}}{\sqrt{((c1X-a1X)^2-(c1Y-a1Y)^2)}} \quad (3.3)$$

Persamaan 3.4 dan 3.5 digunakan untuk mencari panjang jarak titik kalibrasi A dan B dari Pixy2 dan Dobot Magician. lr1 merupakan panjang jarak titik AB dari Pixy2 dan lr2 merupakan panjang jarak titik AB dari Dobot Magician.

$$lr1 = \sqrt{((b1X - a1X)^2 - (b1Y - a1Y)^2)} \quad (3.4)$$

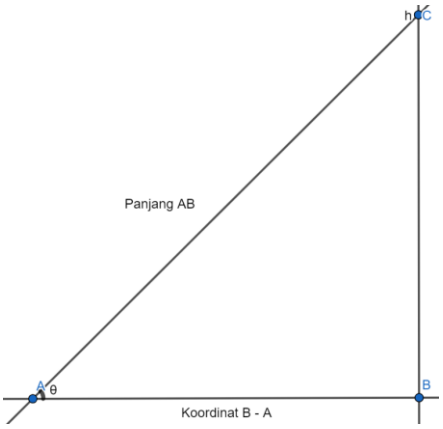
$$lr2 = \sqrt{((b2X - a2X)^2 - (b2Y - a2Y)^2)} \quad (3.5)$$

Hasil akhir dari perhitungan ini adalah untuk mencari dobotPartX dan dobotPartY. Koordinat ini merupakan koordinat objek dari sistem Dobot Magician. Persamaan 3.6 dan 3.7 digunakan untuk melakukan translasi terhadap koordinat X dari titik A sistem koordinat Pixy2.

$$dobotPartX = pixyPartX - a1X \quad (3.6)$$

$$dobotPartY = pixyPartY - a1Y \quad (3.7)$$

Persamaan 3.8 dibawah akan digunakan untuk mengetahui sudut antara kertas kalibrasi dengan kamera Pixy2.



Gambar 3.9 Mencari perbedaan sudut kertas kalibrasi dan Pixy2

Persamaan ini diambil dengan menggunakan rumus cosinus dimana bagian dari sudut dibagi bagian miring dari sudut seperti pada Gambar 3.9. Sehingga akan didapatkan persamaan di bawah ini.

$$angle1 = arccos(\frac{b1X - a1X}{lr1}) \quad (3.8)$$

Kemudian koordinat objek akan dirotasi sesuai dengan sudut yang didapat pada persamaan 3.8. Setelah dilakukan rotasi maka koordinat akan di-*scaling* agar sesuai dengan sistem koordinat Dobot Magician. Perhitungan ini dapat dilihat pada persamaan 3.11 dan 3.12.

$$dobotPartX = (dobotPartX \times cos(-1 \times angle1)) + (dobotPartY \times sin(-1 \times angle1)) \quad (3.9)$$

$$dobotPartY = \big((-1 \times dobotPartX) \times sin(-1 \times angle1)\big) + (dobotPartY \times cos(-1 \times angle1)) \quad (3.10)$$

$$dobotPartX = dobotPartX \times scaleX \quad (3.11)$$

$$dobotPartY = dobotPartY \times scaleY \quad (3.12)$$

Untuk mendapatkan sudut antara Pixy2 dengan sistem koordinat Dobot Magician akan digunakan persamaan yang sama dengan persamaan 3.8. Berikut adalah persamaan yang digunakan untuk mencari besar sudut tersebut.

$$angle2 = arccos(\frac{b2X - a2X}{lr2}) \quad (3.13)$$

Setelah koordinat dari sistem telah di-*scaling* dan telah sesuai dengan sistem koordinat Pixy2, titik koordinat tersebut akan dirotasi kembali agar sesuai dengan sistem koordinat Dobot Magician. Berikut adalah persamaan yang akan digunakan.

$$dobotPartX = (intermediateX \times cos(angle2)) + (intermediateY \times sin(angle2)) \quad (3.14)$$

$$dobotPartY = (-1 \times intermediateX \times sin(angle2)) + (intermediateY \times cos(angle2)) \quad (3.15)$$

Terakhir, koordinat objek akan ditranslasikan dengan titik koordinat A yang dimiliki Dobot Magician.

$$dobotPartX = dobotPartX + a2X \quad (3.16)$$

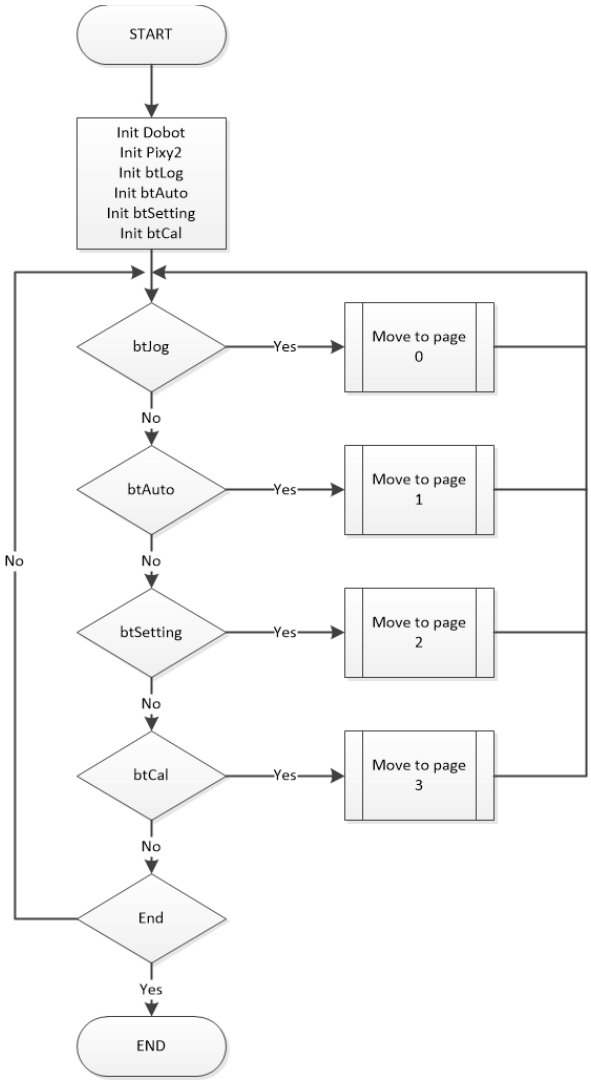
$$dobotPartY = dobotPartY + a2Y \quad (3.17)$$

3.3 *Flowchart* Sistem

```
#include <EEPROM.h>
#include <Nextion.h>
#include <Pixy2.h>
#include <avr/wdt.h>
#include "FlexiTimer2.h"
#include "Protocol.h"
#include "command.h"
#include "stdio.h"
```

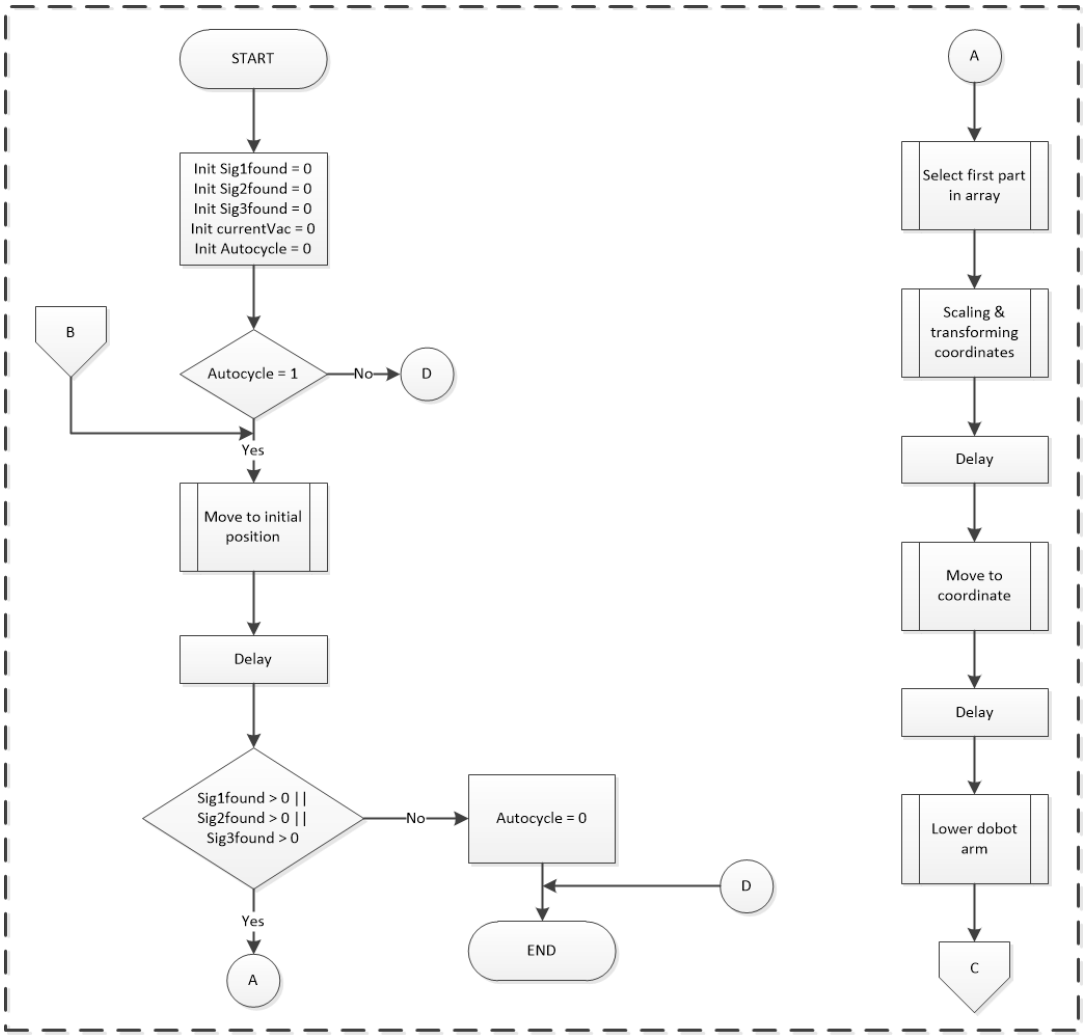
Gambar 3.10 *Library* sistem

Pada potongan kode di atas merupakan *library* yang digunakan pada sistem ini. Terdapat "Pixy2.h" yang diperlukan untuk menjalankan *vision camera* Pixy2. "Nextion.h" digunakan untuk menjalankan HMI yang akan diperlukan. "EEPROM.h" diperlukan untuk mengirimkan dan menerima data dari EEPROM, "avr/wdt.h" digunakan untuk melakukan reset pada arduino. "Stdio.h" merupakan *library* yang digunakan untuk input dan output dalam bahasa C. "Protocol.h" digunakan untuk mengirim dan menerima data melalui *port* serial. "Command.h" berisi deklarasi fungsi-fungsi dan data yang digunakan untuk mengatur parameter Dobot Magician seperti fungsi dari *end effector*, JOG, dan *point to point movement*. Terakhir, "Flexitimer2.h" merupakan *library* yang digunakan untuk *timer* pada Arduino. Semua *library* ini diperlukan agar sistem dapat berjalan dengan baik. Penjelasan sistem selanjutnya akan dijelaskan pada *flowchart* di bawah ini.

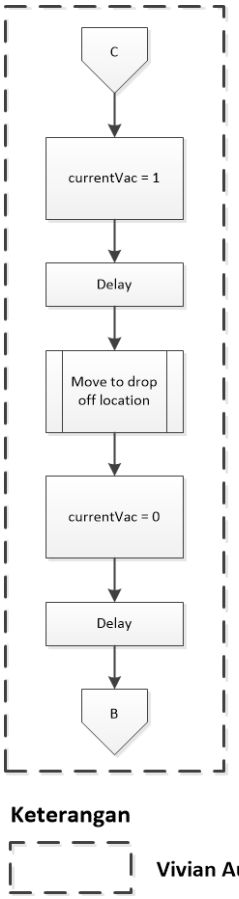


Gambar 3. 11 *Flowchart* sistem

Seperti yang terlihat pada *flowchart* Gambar 3.11, pada sistem akan terdapat beberapa pembagian sistem lainnya. Pada sistem pertama, yaitu Jog merupakan sistem yang digunakan untuk melakukan *manual jogging* pada Dobot Magician. Sistem kedua, yaitu Auto merupakan sistem yang berfungsi untuk melakukan proses *pick and place* objek berdasarkan warna secara otomatis. Pada Setting terdapat pengaturan yang dilakukan untuk menyimpan posisi awal Dobot Magician, posisi Dobot Magician ketika pertama dinyalakan, ketinggian lengan robot ketika melakukan pengambilan objek, serta pengaturan kecepatan pergerakan Dobot Magician. Terakhir, pada Cal terdapat pengaturan yang diperlukan untuk melakukan kalibrasi pada Dobot Magician dan mengatur lokasi *drop-off* objek.



Gambar 3.12 *Flowchart* sistem *pick and place* berdasarkan warna otomatis bagian 1



Gambar 3.13 *Flowchart* sistem *pick and place* berdasarkan warna otomatis bagian 2

Flowchart pada Gambar 3.12 dan Gambar 3.13 ini merupakan sistem yang berada pada bagian Auto yang berfungsi untuk melakukan sistem pemilahan objek berdasarkan warna secara otomatis. Sistem ini hanya dapat bekerja dengan baik apabila kalibrasi telah dilakukan sebelumnya. Sistem akan dimulai apabila *Autocycle* dinyalakan. Pertama-tama, lengan Dobot Magician akan bergerak pada posisi awal yang telah diatur sebelumnya. Posisi ini bertujuan agar Pixy2 dapat menangkap seluruh permukaan bidang datar. Selanjutnya, Pixy2 akan melakukan pengenalan warna. Apabila ada warna objek yang dikenali Pixy2, maka proses akan berjalan sebagaimana mestinya. Terdapat juga *delay* setelah tiap pergerakan untuk memberikan waktu Dobot Magician dalam mengatur posisinya sebelum melakukan tindakan selanjutnya.

Pixy kemudian akan mengurutkan objek berdasarkan ukuran, mulai dari yang terkecil hingga terbesar. Setelah objek telah diurutkan, sistem akan memilih objek pertama yang berada pada *array* yaitu objek terkecil. Koordinat objek ini kemudian akan ditransformasikan agar sesuai dengan koordinat sistem pada Dobot Magician. Bagian ini dijelaskan pada subbab 3.2, yaitu kalibrasi.

Setelah koordinat objek telah didapat, Dobot Magician akan bergerak menuju objek tersebut. Tetapi koordinat pada sumbu Z akan ditambahkan sebanyak 20 unit. Hal ini bertujuan agar lengan Dobot Magician tidak bertabrakan dengan objek lain ketika bergerak menuju objek. Selanjutnya, lengan robot akan turun sesuai dengan tinggi yang telah diatur sebelumnya untuk melakukan pengambilan objek.

Apabila proses telah selesai, *vacuum cup* atau *gripper* Dobot Magician akan menyala untuk mengambil objek. Setelah itu, lengan objek akan bergerak menuju tempat *drop-off* objek berdasarkan warna objek. Warna pertama akan diletakkan pada lokasi pertama, warna kedua pada lokasi kedua dan lain begitu pula selanjutnya. Ketika sampai pada lokasi yang diinginkan, *vacuum cup* atau *gripper* akan mati dan meletakkan objek pada tempatnya. Proses ini akan kembali diulang hingga Pixy2 tidak lagi mendeteksi objek.

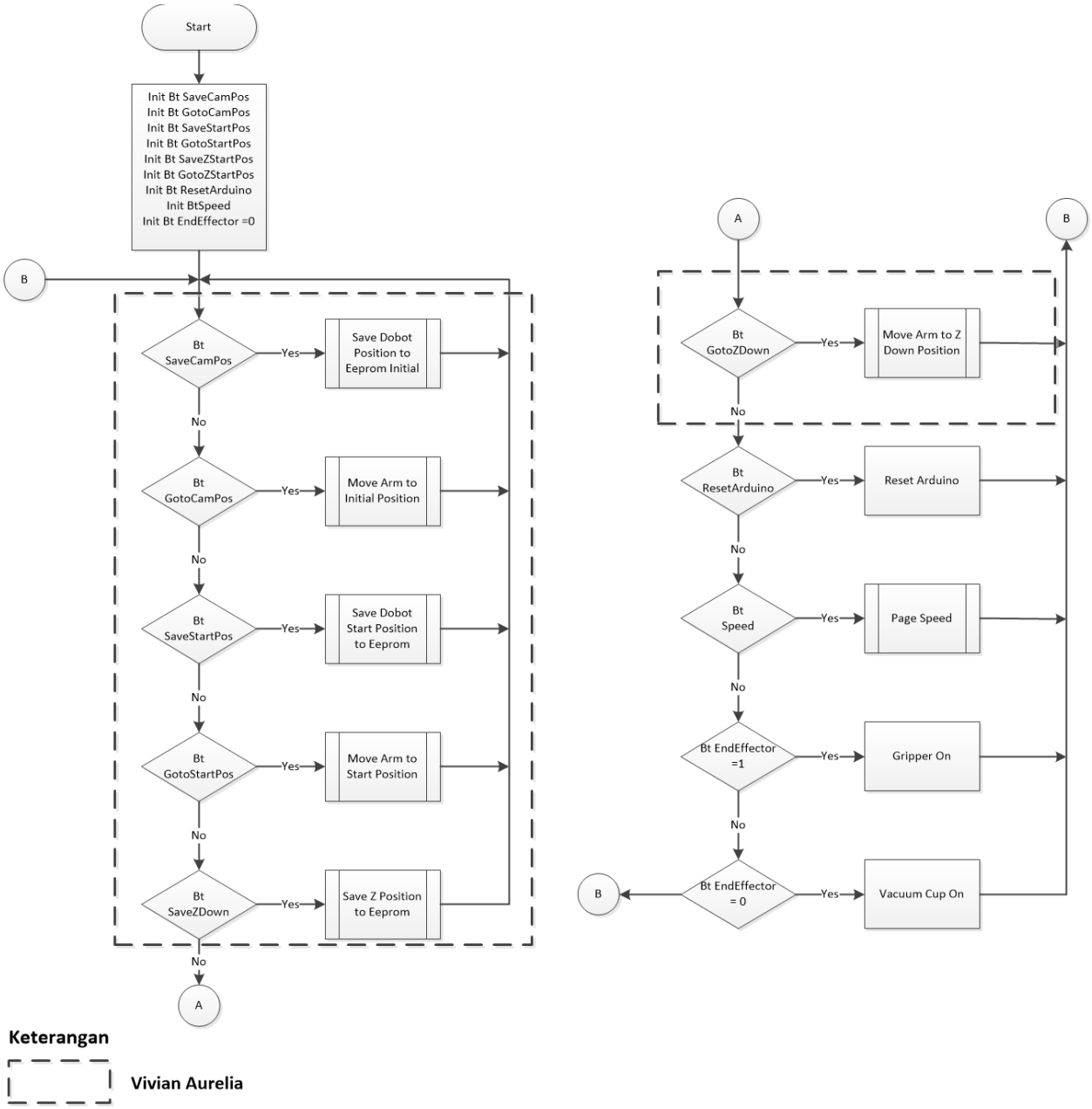
Seluruh perintah untuk menggerakkan lengan robot akan menggunakan *function* bernama MoveArm. Pada *function* tersebut terdapat perintah untuk mengirimkan koordinat ke Dobot Magician. Perintah yang dimaksud adalah gPTPCmd. Perintah gPTPCmd merupakan bagian dari *playback command* yang digunakan Dobot Magician. Perintah ini digunakan untuk pengaturan dan konfigurasi parameter gerak yang relevan, termasuk parameter sendi, parameter sistem koordinat, parameter rasio, dan parameter terkait lainnya (Shenzhen Yuejiang Technology Co. L., 2019). Berikut adalah potongan programnya.

```
void moveArm(float x, float y, float z, float r, bool vacuumOn) {

   gPTPCmd.x = x;
   gPTPCmd.y = y;
   gPTPCmd.z = z;
   gPTPCmd.r = r;

   Serial.print("Pindah ke x:");
   Serial.print(gPTPCmd.x);
   Serial.print(" y:");
   Serial.print(gPTPCmd.y);
   Serial.print(" z:");
   Serial.println(gPTPCmd.r);

   SetPTPCmd(&gPTPCmd, true, &gQueuedCmdIndex);
```

Gambar 3.14 Program perintah menggerakkan Dobot Magician bagian 1

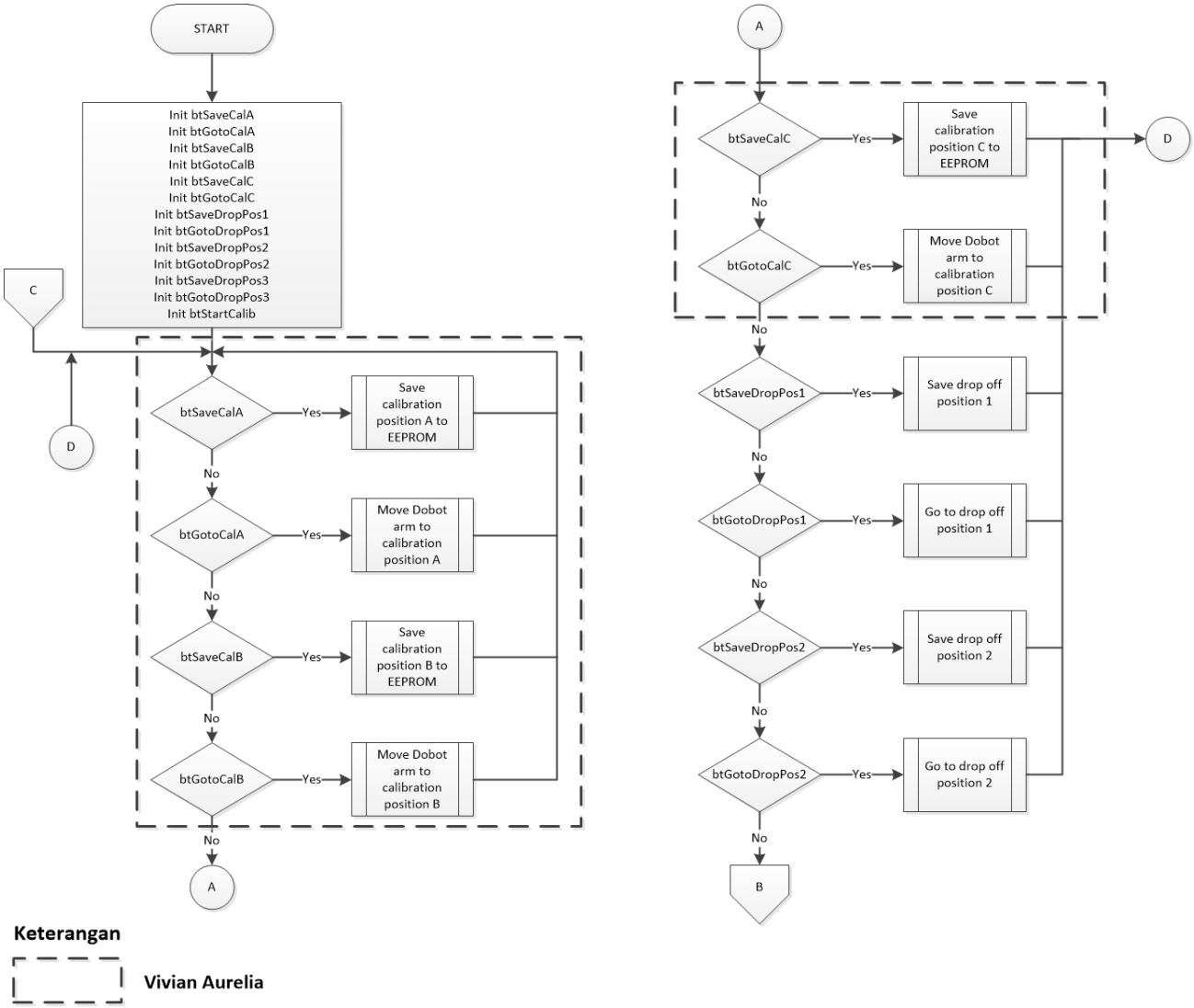
```
    if (endeffectorGripper == 1) {
       if (vacuumOn == false) {
          Serial.println("Buka Gripper");
          SetEndEffectorSuctionCup(false, true, &gQueuedCmdIndex);
          ProtocolProcess();
          delay(500);
          SetEndEffectorGripper(true, true, &gQueuedCmdIndex);
          ProtocolProcess();
          delay(500);
          SetEndEffectorGripper(false, true, &gQueuedCmdIndex);
          delay(500);
       }
    } else {
       if (vacuumOn == false)
         SetEndEffectorSuctionCup(false, true, &gQueuedCmdIndex);
       if (vacuumOn == true)
         SetEndEffectorSuctionCup(true, true, &gQueuedCmdIndex);
    }

     if (vacuumOn == true && vacuumOn != currentVac)
         SetEndEffectorSuctionCup(true, true, &gQueuedCmdIndex);
     if (vacuumOn == false)
         SetEndEffectorSuctionCup(false, true, &gQueuedCmdIndex);

    ProtocolProcess();

    currentVac = vacuumOn;
    currentX = x;
    currentY = y;
    currentZ = z;
    currentR = r;
}
```

Gambar 3.15 Program perintah menggerakkan Dobot Magician bagian 2

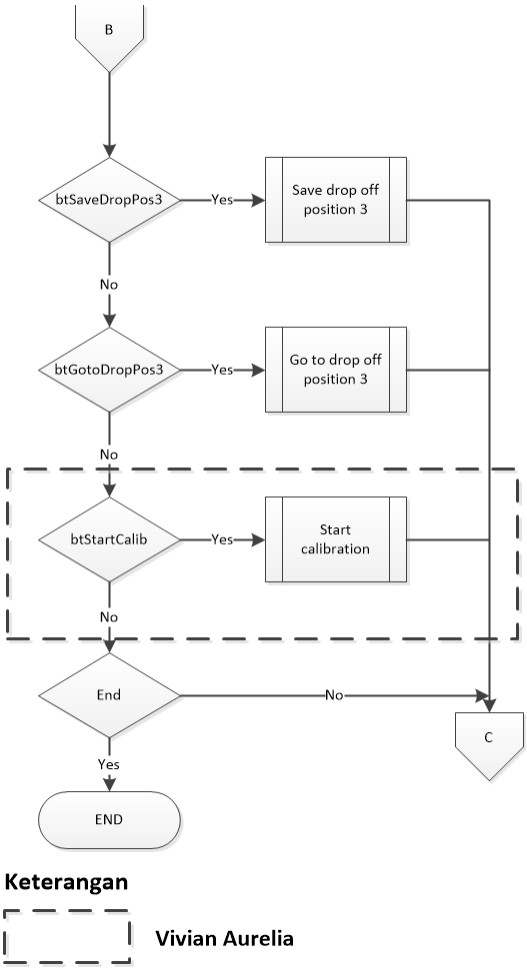


Gambar 3.16 *Flowchart* pengaturan Dobot Magician

Seperti yang terlihat pada *flowchart* yang terdapat pada Gambar 3.16, sistem ini berfungsi untuk menyimpan koordinat Dobot Magician ketika baru dihidupkan, posisi untuk melakukan kalibrasi, dan ketinggian sumbu Z ketika akan melakukan pengambilan objek. Terdapat juga pemilihan *end-effector* Dobot Magician yang digunakan dalam melakukan proses *pick and place.* Pada sistem ini juga, terdapat bagian yang digunakan untuk mengatur kecepatan dari pergerakan Dobot Magician. Penjelasan lebih lanjut mengenai prosedur yang ada akan dijelaskan pada *flowchart* yang ada pada lampiran.



Gambar 3.17 *Flowchart* sistem pengaturan kalibrasi bagian 1



Gambar 3.18 *Flowchart* sistem pengaturan kalibrasi bagian 2

Bagian *flowchart* pada Gambar 3.17 dan Gambar 3.18 merupakan bagian yang diperlukan dalam menyimpan koordinat tiga titik kalibrasi dan koordinat *drop-off* objek ke EEPROM. Terdapat juga bagian yang memerintahkan lengan robot untuk pergi ke tiga titik yang telah disimpan sebelumnya. Selain itu, sistem ini juga berfungsi untuk melakukan proses kalibrasi dengan menggunakan Pixy2 dalam mencari tiga titik pada kertas kalibrasi. Apabila jumlah titik kalibrasi yang ditemukan tidak sama tiga maka kalibrasi gagal dilakukan. Program yang digunakan dapat dilihat di bawah ini.

```
if (CalibDotFound == 3) {
      Serial.print(F("Detected "));
      Serial.println(pixy.ccc.numBlocks);
      for (i = 0; i < pixy.ccc.numBlocks; i++) {
         Serial.print(F("  block "));
         Serial.print(i);
         Serial.print(F(": "));
         pixy.ccc.blocks[i].print();
      }
for (j = 0; j < pixy.ccc.numBlocks; j++)
{
         for (k = 0; k < (pixy.ccc.numBlocks - 1); k++) {
            if ((pixy.ccc.blocks[k].m_x + pixy.ccc.blocks[k].m_y) >
(pixy.ccc.blocks[k + 1].m_x + pixy.ccc.blocks[k + 1].m_y)) {
               tempFloatVarX = pixy.ccc.blocks[k].m_x;
               tempFloatVarY = pixy.ccc.blocks[k].m_y;
               tempIntVarSignature = pixy.ccc.blocks[k].m_signature;
               pixy.ccc.blocks[k].m_x = pixy.ccc.blocks[k + 1].m_x;
               pixy.ccc.blocks[k].m_y = pixy.ccc.blocks[k + 1].m_y;
               pixy.ccc.blocks[k].m_signature = pixy.ccc.blocks[k+1].m_signature;
               pixy.ccc.blocks[k + 1].m_x = tempFloatVarX;
               pixy.ccc.blocks[k + 1].m_y = tempFloatVarY;
               pixy.ccc.blocks[k + 1].m_signature = tempIntVarSignature;
            }
         }
      }
      k = 0;
      for (j = 0; j < pixy.ccc.numBlocks; j++) {
         Serial.print(pixy.ccc.blocks[j].m_signature);
         if (pixy.ccc.blocks[j].m_signature == signatureCalib) {
            if (k == 1) {
               a1X = pixy.ccc.blocks[j].m_x; }
            if (k == 1) {
               a1Y = (207 - pixy.ccc.blocks[j].m_y); }
            if (k == 2) {
               b1X = pixy.ccc.blocks[j].m_x; }
            if (k == 2) {
               b1Y = (207 - pixy.ccc.blocks[j].m_y); }
            if (k == 0) {
               c1X = pixy.ccc.blocks[j].m_x; }
            if (k == 0) {
               c1Y = (207 - pixy.ccc.blocks[j].m_y); }
            k++;
         }
      }
```

Gambar 3.19 Program mencari tiga titik kalibrasi

Program ini juga akan menentukan titik kalibrasi mana yang merupakan titik kalibrasi A, B, dan C dengan cara menjumlahkan koordinat X dan Y tiap titik. Titik kalibrasi C berada pada bagian kiri atas kertas kalibrasi sehingga memiliki nilai yang paling kecil. Titik kalibrasi A berada pada bagian kiri bawah sehingga memiliki nilai yang berada ditengah titik B dan C. Sedangkan, titik kalibrasi B berada pada bagian kanan bawah sehingga memiliki nilai yang paling tinggi.

3.3 Modul Pembelajaran

Pada penelitian ini akan dibuat modul pembelajaran yang bertujuan untuk meningkatkan pemahaman peserta didik dalam penggunaan *vision camera* Pixy2 dan Dobot Magician. Terdapat dua modul yang akan dibuat, modul pertama akan berupa cara menggunakan Pixy2 dengan bantuan aplikasi PixyMon. Pada modul ini, peserta didik akan diberikan langkah-langkah mengenai bagaimana cara mengunduh PixyMon, memilih program yang akan digunakan, bagaimana mengajarkan warna pada Pixy2, melakukan *tuning* pada kamera untuk tiap warna, pencahayaan, dan pengaturan lainnya. Pada bagian akhir modul akan diberikan pertanyaan dan latihan untuk menguji pemahaman peserta didik.

Pada modul kedua peserta didik diberikan langkah-langkah dalam mengontrol Dobot Magician dengan menggunakan Arduino Mega dan kamera Pixy2. Pada modul ini, pengaturan Pixy2 dapat menggunakan pengaturan yang telah dibuat pada modul pertama. Peserta didik akan diberikan gambar rangkaian dan program yang perlu dilakukan. Program yang dibuat akan berupa program sederhana dalam melakukan *pick and place* dengan menggunakan *vision camera* Pixy2 dan Dobot Magician. Pada akhir modul ini juga akan diberikan pertanyaan dan latihan untuk menguji pemahaman peserta didik.