Program di Arduino

```
#define motorkanan_in1 11
#define PWMkanan_ena 10
#define PWMkiri_enb 9
#define motorkiri_in3 8
#define trigger_depan 2
#define echo_depan 3
#define trigger_kanan 6
#define echo_kanan 7
#define trigger_kiri 4
#define echo_kiri 5
int serialdata_raspi = 0;
long jarak_depan;
long jarak_kanan;
long jarak_kiri;
long microsecondsToCentimeters (long microseconds) {
  return microseconds / 29 / 2;
  }

void setup() {
  pinMode(motorkanan_in1, OUTPUT);
  pinMode(motorkiri_in3, OUTPUT);
  pinMode(PWMkiri_enb, OUTPUT);
  pinMode(PWMkanan_ena, OUTPUT);
  pinMode(trigger_depan, OUTPUT);
  pinMode(echo_depan, INPUT);
  pinMode(trigger_kanan, OUTPUT);
  pinMode(echo_kanan, INPUT);
  pinMode(trigger_kiri, OUTPUT);
  pinMode(echo_kiri, INPUT);
  Serial.begin(115200);
  }

void loop() {
  baca_depan();
  delay(80);
  serialdata_raspi=Serial.read();
  Serial.println(serialdata_raspi);
  if (serialdata_raspi <= 0 && jarak_depan <= 31) {
    rem_total();
    }
    else if (serialdata_raspi > 0 && jarak_depan > 29) {
      baca_kamera_1();
      }
      else if (serialdata_raspi <= 0 && jarak_depan > 31) {
```

```
        rem_total();
        }
        else if (serialdata_raspi > 0 && jarak_depan <= 29) {
          baca_kamera_2();
          }
          }

void rem_total() {
  digitalWrite(motorkanan_in1, LOW);
  digitalWrite(motorkiri_in3, LOW);
  analogWrite(PWMkanan_ena, 0);
  analogWrite(PWMkiri_enb, 0);
  }

void belok_kanan() {
  digitalWrite(motorkanan_in1, HIGH);
  digitalWrite(motorkiri_in3, HIGH);
  analogWrite(PWMkanan_ena, 50);
  analogWrite(PWMkiri_enb, 85);
  }

void belok_kiri() {
  digitalWrite(motorkanan_in1, HIGH);
  digitalWrite(motorkiri_in3, HIGH);
  analogWrite(PWMkanan_ena, 85);
  analogWrite(PWMkiri_enb, 50);
  }

void maju() {
  digitalWrite(motorkanan_in1, HIGH);
  digitalWrite(motorkiri_in3, HIGH);
  analogWrite(PWMkanan_ena, 85);
  analogWrite(PWMkiri_enb, 85);
  }

void baca_depan() {
  digitalWrite(trigger_depan, LOW);
  delay(2);
  digitalWrite(trigger_depan, HIGH);
  delay(5);
  digitalWrite(trigger_depan, LOW);
  jarak_depan = pulseIn(echo_depan, HIGH);
  jarak_depan = microsecondsToCentimeters(jarak_depan);
  }
```

```
void baca_kanan() {
  digitalWrite(trigger_kanan, LOW);
  delay(2);
  digitalWrite(trigger_kanan, HIGH);
  delay(5);
  digitalWrite(trigger_kanan, LOW);
  jarak_kanan = pulseIn(echo_kanan, HIGH);
  jarak_kanan = microsecondsToCentimeters(jarak_kanan);
  }

void baca_kiri() {
  digitalWrite(trigger_kiri, LOW);
  delay(2);
  digitalWrite(trigger_kiri, HIGH);
  delay(5);
  digitalWrite(trigger_kiri, LOW);
  jarak_kiri = pulseIn(echo_kiri, HIGH);
  jarak_kiri = microsecondsToCentimeters(jarak_kiri);
  }

void baca_kamera_1() {
  if (serialdata_raspi > 193) {
    belok_kanan();
    }
    else if (serialdata_raspi < 63) {
      belok_kiri();
      }
      else if (serialdata_raspi <= 0) {
        belok_kiri();
        }
        else {
          baca_jarak();
          }
          }

void baca_kamera_2() {
  if (serialdata_raspi > 193) {
    belok_kanan();
    }
    else if (serialdata_raspi < 63) {
      belok_kiri();
      }
      else if (serialdata_raspi <= 0) {
        belok_kiri();
        }
        }
```

```
void baca_jarak() {
  if (jarak_depan > 29) {
    maju();
    delay(300);
    }
    else if (jarak_depan <= 31) {
      rem_total();
      }
      }
```

Program di Python

```
from collections import deque
from picamera.array import PiRGBArray
from picamera import PiCamera
import time
import numpy as np
import argparse
import imutils
import cv2
import serial

ser = serial.Serial('/dev/ttyACM0', 115200)
greenLower = (100, 107, 5)
greenUpper = (138, 255, 255)
camera = PiCamera()
camera.resolution = (256, 240)
camera.framerate = 2
rawCapture = PiRGBArray(camera, size=(256, 240))
time.sleep(0.1)
for frame in camera.capture continuous (rawCapture,format="bgr",use_video_port
= True):

# grab the raw NumPy array representing the image, then initialize the timestamp
and occupied/unoccupied text
Image = frame.array
hsv = cv2.cvtColor(image, cv2.COLOR_BGR2HSV)
mask = cv2.inRange(hsv, greenLower, greenUpper)
mask = cv2.erode(mask, None, iterations=2)
mask = cv2.dilate(mask, None, iterations=2)
cnts = cv2.findContours(mask.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)[-2]
center = None
        if len(cnts) > 0:
        c = max(cnts, key=cv2.contourArea)
        ((x, y), radius) = cv2.minEnclosingCircle(c)
        M = cv2.moments(c)
        center = (int(M["m10"] / M["m00"]), int(M["m01"] / M["m00"]))
        hori = (int(M["m10"] / M["m00"]),)
                ser.write(hori)
                ser.flushOutput()
                #sleep(.1)
        if radius > 10:
        cv2.circle(image, (int(x), int(y)), int(radius), (0, 255, 255), 2)
        cv2.circle(image, center, 5, (0, 0, 255), -1)
        cv2.imshow("Frame", image)
```

```
key = cv2.waitKey(1) & 0xFF
rawCapture.truncate(0)
If key == ord("q"):
break
```