DAFTAR PUSTAKA

[1] T. Umetani, Y. Kondo, and T. Tokuda, “Rapid development of a mobile robot for the nakanoshima challenge using a robot for intelligent environments,” *J. Robot. Mechatronics*, vol. 32, no. 6, pp. 1211–1218, 2020, doi: 10.20965/jrm.2020.p1211.

[2] R. Siegwart, I. R. Nourbakhsh, and D. Scaramuzza, *Introduction to Autonomous Mobile Robots, second edition*. MIT Press, 2011.

[3] T. J. Lee, C. H. Kim, and D. I. D. Cho, “A Monocular Vision Sensor-Based Efficient SLAM Method for Indoor Service Robots,” *IEEE Trans. Ind. Electron.*, vol. 66, no. 1, pp. 318–328, 2019, doi: 10.1109/TIE.2018.2826471.

[4] M. Fausi, “Analisis Pengaruh Penambahan Jumlah Array Microphone Terhadap Estimasi Direction Of Arrival (DOA) dengan Teknik Pemrosesan Sinyal Fast Fourier Transform Beamforming,” vol. 12, no. 1, pp. 36–39, 2018.

[5] R. I. Alfian, A. Ma’Arif, and S. Sunardi, “Noise reduction in the accelerometer and gyroscope sensor with the Kalman filter algorithm,” *J. Robot. Control*, vol. 2, no. 3, pp. 180–189, 2021, doi: 10.18196/jrc.2375.

[6] A. P. Abseno, “Penerapan Kinematika Untuk Lokalisasi Pada Robot Sepak Bola Beroda,” p. 73, 2019.

[7] A. Madhloom, F. Raheem, and A. Kareem, “A Modified Kalman Filter-Based Mobile Robot Position Measurement using an Accelerometer and Wheels Encoder,” *Eng. Technol. J.*, vol. 40, no. 1, pp. 267–274, 2022, doi: 10.30684/etj.v40i1.2082.

[8] A. Maarif, R. D. Puriyanto, and F. R. T. Hasan, “Robot Keseimbangan dengan Kendali PID dan Kalman Filter,” *It J. Res. Dev.*, vol. 4, no. 2, pp. 117–127, 2020, doi: 10.25299/itjrd.2020.vol4(2).3900.

[9] F. Fahmizal, D. U. Rijalussalam, M. Budiyanto, and A. Mayub, “Trajectory Tracking pada Robot Omni dengan Metode Odometry,” *J. Nas. Tek. Elektro dan Teknol. Inf.*, vol. 8, no. 1, p. 35, 2019, doi: 10.22146/jnteti.v8i1.488.

[10] A. Ma’arif, I. Iswanto, A. A. Nuryono, and R. I. Alfian, “Kalman Filter for Noise Reducer on Sensor Readings,” *Signal Image Process. Lett.*, vol. 1, no. 2, pp. 11–22, 2019, doi:

10.31763/simple.v1i2.2.

[11] I. Hudati, E. S. A. Nugroho, and N. D. Resty, “Implementasi Filter Kalman pada Sensor Jarak Berbasis Ultrasonik,” *J. List. Instrumentasi dan Elektron. Terap.*, vol. 2, no. 2, pp. 20–24, 2021, doi: 10.22146/juliet.v2i2.71147.

[12] M. I. Marzuki, I. N. Farida, and ..., “Implementasi Controller PID (Proportional, Integral, Derivative) pada Robot Sepak Bola Beroda,” *Pros. SEMNAS ...*, pp. 297–302, 2021, [Online]. Available: https://proceeding.unpkediri.ac.id/index.php/inotek/article/view/971%0Ahttps://proceeding.unpkediri.ac.id/index.php/inotek/article/download/971/641

[13] T. O. Hodson, “Root-mean-square error (RMSE) or mean absolute error (MAE): when to use them or not,” *Geosci. Model Dev.*, vol. 15, no. 14, pp. 5481–5487, 2022, doi: 10.5194/gmd-15-5481-2022.

[14] M. Amin and J. Triyanto, “Rancangan Perangkat Lunak Akuisisi Data Modul Detektor Gamma RosRao Berbasis Modbus Over TCP/IP Menggunakan PyQT5,” *Prima*, vol. 17, no. 1, pp. 40–49, 2020.

[15] V. M. Ionescu and F. M. Enescu, “Investigating the performance of MicroPython and C on ESP32 and STM32 microcontrollers,” *2020 IEEE 26th Int. Symp. Des. Technol. Electron. Packag. SIITME 2020 - Conf. Proc.*, pp. 234–237, 2020, doi: 10.1109/SIITME50350.2020.9292199.

[16] M. Babiuch, P. Foltynek, and P. Smutny, “Using the ESP32 microcontroller for data processing,” *Proc. 2019 20th Int. Carpathian Control Conf. ICCC 2019*, pp. 1–6, 2019, doi: 10.1109/CarpathianCC.2019.8765944.

[17] M. Thothadri, “An Analysis on Clock Speeds in Raspberry Pi Pico and Arduino Uno Microcontrollers,” *Am. J. Eng. Technol. Manag.*, vol. 6, no. 3, p. 41, 2021, doi: 10.11648/j.ajetm.20210603.13.

[18] M. Mudarris and S. G. Zain, “Implementasi Sensor Inertial Meansurenment Unit (IMU) untuk Monitoring Perilaku Roket,” *Avitec*, vol. 2, no. 1, pp. 55–64, 2020, doi: 10.28989/avitec.v2i1.610.

[19] Y. R. Sujono, E. S. Budi, and I. Nugrahanto, “Modul Pengaturan Motor Pompa DC Metode PID pada Sistem Kontrol Ketinggiian

Air berbasis Arduino,” vol. 10, pp. 128–136, 2023.

[20] M. M. H. Ma’arif, “Sistem Navigasi Mobile Robot Mekanum Menggunakan Sensor Kompas,” *Elkolind*, vol. 9, no. September, pp. 154–160, 2022.

LAMPIRAN A
Program sistem

Program Arduino Kalman Filter

```
#include "Wire.h"
#include "I2Cdev.h"
#include "MPU6050.h"
MPU6050 mpu
#include <ArduinoJson.h>

#define enc0_interrupt 2
#define enc0_digital 3
#define enc1_interrupt 6
#define enc1_digital 7

volatile long count0 = 0, count1 = 0;
volatile long posisi0, posisi1;
volatile float enc0, enc1;
float enc_dataX, enc_dataY;

// Deklarasi variabel global yaw imu
float yaw = 0.0;
float firstYaw, offsideYaw;
int setAwal = 0;

// Kalman filter variables
const float Q_gyro = 0.01;
const float R_angle = 10;

float yaw_angle = 0.0;
float P_00 = 0.1, P_01 = 0.0, P_10 = 0.0, P_11 = 0.1;
unsigned long prev_time;
bool is_yaw_resetting = false;
```

```
// Gyro calibration offsets
int16_t gyro_offset_x = 0;
int16_t gyro_offset_y = 0;
int16_t gyro_offset_z = 0;

//variabel kirim data
unsigned long delayReadGyro = 0;
unsigned long delaySendData = 0;

void setup() {
  Wire.begin();
  Serial.begin(115200);
  Serial1.setRX(13);
  Serial1.setTX(12);
  Serial1.begin(115200);

  enc_setup();

  mpu.initialize();
  Serial.println(mpu.testConnection() ? "MPU6050 connection
successful" : "MPU6050 connection failed");

  calibrateGyro();
  prev_time = millis();
}

void loop() {

  encoder();
  if (millis() >= delayReadGyro) {

    delayReadGyro = millis() + 7;
    int16_t gx, gy, gz;
    mpu.getRotation(&gx, &gy, &gz);

    gx -= gyro_offset_x;
    gy -= gyro_offset_y;
    gz -= gyro_offset_z;
    float gyro_z = gz / 131.0;

    //  Serial.print("Raw Gyro: ");
    Serial.print(gyro_z);
    Serial.print("\t");


    updateKalmanFilter(yaw_angle, gyro_z);
    yaw = (yaw_angle - offsideYaw) + setAwal;
    float newYaw;
    float mapyaw;
    if (yaw > 180)newYaw = map(yaw, 180, 270, -180, -90);
    else newYaw = yaw;
    if (newYaw < 0)mapyaw = map(newYaw, -180, 0, 180, 360);
    else mapyaw = newYaw;
//    newYaw = int(newYaw);

    float inversmap = map(mapyaw, 0, 360, 360, 0);
     mapyaw = map(inversmap, 360, 0, 0, 360);

    //  Serial.print("Filtered Yaw: ");
    Serial.println(yaw);
    delay (10);
  }

  // Membaca data dari Serial1 jika tersedia
  if (Serial1.available() > 0) {
    String dataFromEsp = Serial1.readString();
    Serial.println(dataFromEsp);
    if (dataFromEsp == "R") {
      reset_encoder();
      resetYaw();
    }
  }

  if (millis() > delaySendData)
  {
    delaySendData = millis() + 10;
    StaticJsonDocument<200> jsonDocument;
    jsonDocument["yaw"] = String(yaw, 3);
    jsonDocument["enc_dataX"] = String(enc_dataX, 3);
    jsonDocument["enc_dataY"] = String(enc_dataY, 3);

    String jsonString;
    serializeJson(jsonDocument, jsonString);

    Serial1.println(jsonString);
//    Serial.println(jsonString);
  }
}

void resetYaw()
{
  offsideYaw = yaw_angle;
}

void calibrateGyro() {
  int16_t gx_sum = 0;
  int16_t gy_sum = 0;
  int16_t gz_sum = 0;
  const int num_readings = 100;

  for (int i = 0; i < num_readings; i++) {
    int16_t gx, gy, gz;
    mpu.getRotation(&gx, &gy, &gz);
    gx_sum += gx;
    gy_sum += gy;
    gz_sum += gz;
  }
  gyro_offset_x = gx_sum / num_readings;
  gyro_offset_y = gy_sum / num_readings;
  gyro_offset_z = gz_sum / num_readings;
}

void updateKalmanFilter(float newAngle, float newRate) {
  unsigned long curr_time = millis();
  float dt = (curr_time - prev_time) / 1000.0;
  prev_time = curr_time;


  // Update covariance matrix
  P_00 += dt * (dt * P_11 - P_01 - P_10) * dt;
  P_01 -= dt * P_11;
  P_10 -= dt * P_11;
  P_11 += Q_gyro * dt;

  float S = P_00 + R_angle;
  float K_0 = P_00 / S;
  float K_1 = P_10 / S;

  // Calculate difference between measured angle and estimated angle
  float y = newAngle - yaw_angle;

  // Update yaw_angle based on the Kalman gain
  yaw_angle += K_0 * y;

  // Map yaw_angle to the range -180 to 180 degrees
//  yaw_angle = mapTo180(yaw_angle);

  // Update covariance matrix based on Kalman gain
  P_00 -= K_0 * P_00;
  P_01 -= K_0 * P_01;
  P_10 -= K_1 * P_00;
  P_11 -= K_1 * P_01;

  // Finally, update yaw_angle using gyro rate
  yaw_angle += dt * newRate;
}
float mapTo180(float angle) {
  if (angle > 180) {
    angle -= 360;
  } else if (angle < -180) {
    angle += 360;
  }
  return angle;
}
```

Program main Micropython

```
from kirim_dataRobot import *
import kirim_dataRobot
import _thread
from time import ticks_ms
from machine import UART
from moving_control import OmniMobileRobot
from calculation_PID import PID
from calculation_motorSpeed import motor_speed


# Inisialisasi objek PID
pid = PID(Kp=6, Ki=0, Kd=2)
setpoint = 0

# Variabel data_robot
dataX_before, dataY_before, dataX_after, dataY_after, yaw, enc_dataX,
enc_dataY, Accel, data1, data2, data3 = 0, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0
pid_output = 0

# Variabel multitask
time_moving_control,time_calc_pid, time_send_basestation = 0,0,0

uart = UART(2, 115200)
strMsg = ' '

def parsed_data(data):
    parsed_data = ujson.loads(data)
    # Mengakses nilai-nilai yang diparsing
    yaw = parsed_data["yaw"]
    enc_dataX = parsed_data["enc_dataX"]
    enc_dataY = parsed_data["enc_dataY"]

    if yaw is not None and enc_dataX is not None and enc_dataY is not
None:
        yaw = float(yaw)
        enc_dataX = float(enc_dataX)
        enc_dataY = float(enc_dataY)

    return yaw, enc_dataX, enc_dataY

moving = OmniMobileRobot(wheel_radius=10)  # Setel wheel_radius
sesuai dengan nilai yang sesuai
x_target = 0
y_target = 500
Kp_pos = 8
Kp_theta = 0  #robot gak perlu berputar arah dulu makanya 0

# Tetap menjalankan program utama
while True:
    data_robot = yaw, enc_dataX, enc_dataY, pid_output
    if uart.any() > 0:
        data = uart.read()
        try:
            data_sensor = parsed_data(data)
            yaw, enc_dataX, enc_dataY = data_sensor
            # print(data_sensor)
        except:
            print("eror serial")
            # pass


    if kirim_dataRobot.ping_received :
        motor_speed(0,1200,pid_output)
#
    else :
        motor_speed(0,0,0)

     # Logika kirim data
    data_robot = (
        round(moving.x, 2),    # Bulatkan dataX_before menjadi 2 angka
dibelakang koma
        round(moving.y, 2),    # Bulatkan dataY_before menjadi 2 angka
dibelakang koma
```

```
        round(yaw, 2),            # Bulatkan yaw menjadi 2 angka dibelakang
koma
        round(enc_dataX, 2),       # Bulatkan enc_dataX menjadi 2 angka
dibelakang koma
        round(enc_dataY, 2),       # Bulatkan enc_dataY menjadi 2 angka
dibelakang koma
        round(pid_output, 2),      # Sudah dibulatkan di atas
        round(dataX_after, 2),     # Bulatkan dataX_after menjadi 2 angka
dibelakang koma
        round(dataY_after, 2)      # Bulatkan dataY_after menjadi 2 angka
dibelakang koma
    )

    if ticks_ms() >= time_send_basestation:
        time_send_basestation = ticks_ms() + 20
        try:
            data_to_basestation(data_robot)
            # print(data_robot)
        except:
            print("eror komunikasi basestation"
```

BIODATA

Nama	:	Muhamad Arif Hidayat
Tempat Tanggal Lahir	:	Pemalang, 11 Maret 2002
Alamat	:	Desa Kuta, RT 41 RW 09, Kec.Belik, Kab.Pemalang, Jawa Tengah, Indonesia, 52356
Email	:	ariif.hiidayat11@gmail.com
Motto	:	Jika *keadaan* kamu buruk, ingat diluar sana banyak yang *keadaannya* **lebih baik** dari kamu.
Hobi	:	Nunggu **One Piece** tamat.

Riwayat Pendidikan:

1.	SDN 04 Kuta	(2008-2014)
2.	SMPN 1 Randuduongkal	(2014-2017)
3.	SMAN 1 Belik	(2017-2020)
4.	Politeknik Negeri Cilacap	(2020-2023)

Penulis telah mengikuti seminar Tugas Akhir pada tanggal 2 Agustus 2023, sebagai salah satu persyaratan untuk memperoleh gelar Ahli Madya (A.Md).