

LAPORAN TUGAS PROYEK
SISTEM PENGATURAN TERTANAM GASAL 2022/2023
“Shooting Point Camera Stabilizer pada Gimbal 2-Axis”



Oleh :

Hakhi Gya Yektianto	07111940000022
Faiz Adhima Fahrudin	07111940000077

Departemen Teknik Elektro
Fakultas Teknologi Elektro dan Informatika Cerdas
Institut Teknologi Sepuluh Nopember Surabaya
2022

BAB 1 : PENDAHULUAN

1.1. Latar Belakang

Dewasa ini, kamera merupakan suatu komponen/peralatan elektrik yang sangat menjamur di kalangan masyarakat, terutama masyarakat Indonesia. Berbagai teknologi terapan yang berbasis pada kamera juga menunjang berbagai aktivitas manusia, seperti halnya fotografi, videografi, bahkan hingga industri per-film-an. Perkembangan paham dan ilmu fotografi maupun videografi juga menjadi salah satu tuntutan tersendiri bagi proses pengembangan teknologi pengambilan gambar maupun video. Perkembangan yang cepat inilah menuntut agar kualitas kamera yang digunakan selalu prima. Kualitas kamera sendiri secara umum bergantung kepada komponen kamera itu sendiri dan juga komponen pendukung. Perangkat pendukung yang sering digunakan untuk mengoptimalkan fungsi kamera adalah gimbal. Gimbal merupakan alat pengendali gerakan kamera yang dapat menggerakkan kamera pada tiga sumbu yaitu pada sumbu x, y, dan z. Gimbal sendiri menjadi salah satu komponen penting yang dapat memengaruhi kinerja maupun hasil akhir yang diberikan oleh sebuah kamera. Penggunaan gimbal diharapkan dapat meningkatkan kinerja dari suatu *device* pengambil gambar, sehingga data yang dihasilkan baik dan memiliki tingkat *noise* yang minimum.

1.2. Rumusan Masalah

Beberapa rumusan masalah yang dapat diformulakan adalah sebagai berikut :

1. Bagaimana merancang suatu sistem penyeimbang kamera dengan gimbal 2-axis untuk mendapatkan hasil tangkapan yang baik ?
2. Bagaimana merancang suatu integrasi sistem yang dapat memudahkan pengguna untuk mengakses maupun melakukan perubahan terhadap gimbal?

1.3. Tujuan

Adapun beberapa tujuan dari tugas proyek ini adalah sebagai berikut :

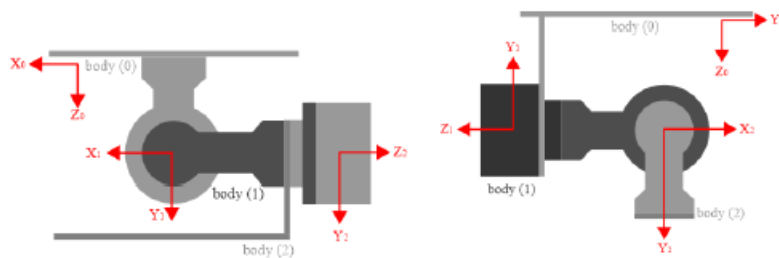
1. Merancang suatu sistem penyeimbang kamera dengan gimbal 2-axis untuk mengoptimalkan fungsi pengambilan gambar pada kamera.
2. Merancang suatu bentuk pengintegrasian agar memudahkan pengguna untuk mengakses maupun melakukan perubahan terhadap gimbal?

BAB 2 : DASAR TEORI

2.1 Camera Gimbal 2-Axis

Camera gimbal adalah alat penstabil kamera, yang berfungsi untuk menghasilkan gambar atau video tanpa terpengaruh oleh kemiringan atau guncangan yang terjadi ketika pengambilan gambar atau video berlangsung. Memanfaatkan kerja gimbal yang dapat bergerak bebas pada porosnya mengakibatkan camera gimbal dapat menyesuaikan orientasi pergerakan ketika terjadi perubahan, sehingga kamera yang ditempatkan pada camera gimbal akan tetap stabil pada tempatnya.

Gimbal 2-sumbu terdiri dari dua buah *joints* yang dapat berputar pada sudut θ_1 dan θ_2 sepanjang sumbu *pitch* dan sumbu *roll*. *Gimbal 2-sumbu* yang digunakan terdiri dari tiga *body frame* (*body* (0), *body* (1), dan *body* (2)). Pada Gambar 2 ditunjukkan hubungan antar *body frame*. *Body* (0), *body* (1), dan *body* (2) dalam bentuk $X_0Y_0Z_0$, $X_1Y_1Z_1$, dan $X_2Y_2Z_2$.



Gambar 2. 1 Camera Gimbal 2-Axis

2.2 ESP-8266

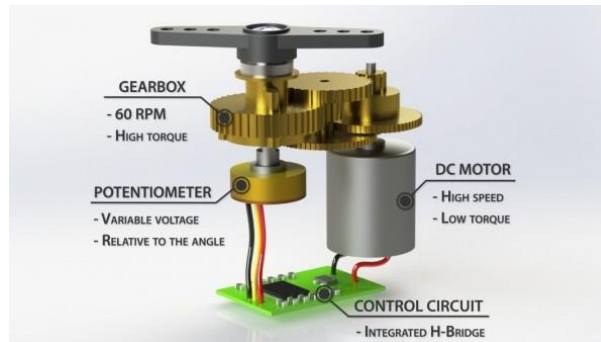
ESP-8266 adalah mikrokontroler yang terintegrasi dengan modul Wi-Fi dan Bluetooth onboard. Proyek berbasis IoT umumnya menggunakan Mikrokontroler ESP32 karena memiliki modul wi-Fi yang sudah terintegrasi onboard sehingga tidak . Selain itu ESP32 juga sudah terintegrasi dengan *built-in antenna switches*, *RF balun*, *power amplifier*, *low-noise receive amplifier*, *filters*, dan *power management modules*. sehingga sangat mendukung untuk pengaplikasian *Internet of Things*. ESP32 mempunyai memori RAM sebesar 320 KB dan ROM sebesar 448 KB. ESP32 memiliki *peripheral Interface* antara lain 34 pin GPIO (*General Purpose Input/Output*), 18 pin ADC (*Analog Digital Converter*), 2 pin DAC (*Digital Analog Converter*), 16 pin PWM (*Pulse Width Modulation*), 10 pin *Capacitive Sensing*, 2 jalur antarmuka UART, pin *interface* I2C, I2S, SPI, dll. Setiap pinout ESP 32 dapat menerima atau memberi tegangan hingga sebesar 3,3V (Espressif Systems, 2021).

2.3 Sensor IMU MPU-6050

Sensor MPU-6050 merupakan sensor IMU 6 derajat kebebasan yang terdiri dari accelerometer 3 derajat kebebasan, gyroscope 3 derajat kebebasan dan sebuah prosesor gerakan digital (DMP). Modul ini menggunakan protokol I2C untuk berkomunikasi dengan mikrokontroler. Gyroscope MPU-6000 dapat diprogram untuk bekerja pada skala 250, 500, 1000 dan 2000 °/detik, untuk accelerometer dapat diprogram pada skala 2G, 4G, 8G, dan 16G. Data dari sensor konversi DMP dengan 16 bit ADC(Vamiko et al., 2013).

2.4 Motor Servo

Motor servo adalah sebuah motor dengan sistem closed loop yang digunakan untuk mengendalikan kecepatan, akselerasi dan posisi akhir dari sebuah motor listrik dengan keakuratan yang tinggi. Umumnya, motor servo banyak digunakan sebagai aktuator yang membutuhkan posisi putaran motor yang presisi. Motor servo terdiri dari tiga bagian utama, yaitu: motor, sistem kontrol dan potensiometer yang terhubung dengan satu set internal gearbox. Motor servo memiliki putaran yang lambat, namun memiliki torsi yang kuat karena adanya internal gear.



Gambar 2. 2 Motor Servo

Motor servo dikendalikan dengan sinyal PWM dari potensiometer. Potentiometer berfungsi sebagai sensor yang memberikan sinyal umpan balik (feedback) untuk megkoreksi posisi target. Sedangkan sudut dari sumbu motor servo diatur berdasarkan lebar pulsa yang dikirim melalui kaki sinyal dari kabel motor. Potentiometer ini terdiri dari tiga kabel dengan 2 kabel untuk power dan 1 kabel untuk kabel sinyal. (Hilal & Manan, 2015)

BAB 3 : PENGUJIAN SISTEM

3.1 Kalibrasi awal

Pada sensor diperlukan kalibrasi nilai awal offset untuk agar pembacaan sensor akurat sebelum dimasukkan ke dalam algoritma perhitungan. Kalibrasi awal menggunakan library dari “MPU6050_6Axis_MotionApps20”. MPU 6050 Berikut merupakan kode untuk kalibrasi.

```
if (devStatus == 0) {
    // Calibration Time: generate offsets and calibrate our MPU6050
    mpu.CalibrateAccel(7); //Kalibrasi Pertama kali
    mpu.CalibrateGyro(7);
    mpu.PrintActiveOffsets();

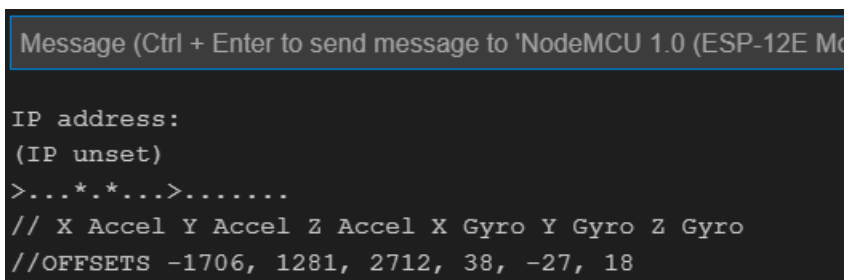
    mpu.setDMPEnabled(true);

    attachInterrupt(digitalPinToInterrupt(INTERRUPT_PIN), dmpDataReady, RISING);
    mpuIntStatus = mpu.getIntStatus();

    dmpReady = true;

    packetSize = mpu.dmpGetFIFOPacketSize();
}
else{
    Serial.print(F("DMP Initialization failed (code "));
    Serial.print(devStatus);
    Serial.println(F(")"));
}
}
```

nilai hasil kalibrasi



```
Message (Ctrl + Enter to send message to 'NodeMCU 1.0 (ESP-12E Module)')
IP address:
(IP unset)
>...*.~>.....
// X Accel Y Accel Z Accel X Gyro Y Gyro Z Gyro
//OFFSETS -1706, 1281, 2712, 38, -27, 18
```

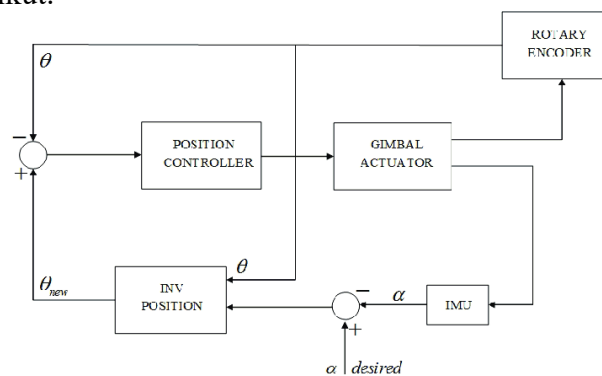
Setelah memperoleh nilai hasil kalibrasi nilai tersebut dimasukkan ke dalam penentuan offset

```
pinMode(INTERRUPT_PIN, INPUT);
mpu.initialize();
devStatus = mpu.dmpInitialize();
pinMode(INTERRUPT_PIN, INPUT);
```

```
// nilai offset hasil kalibrasi
mpu.setXGyroOffset(45);
mpu.setYGyroOffset(-27);
mpu.setZGyroOffset(18);
mpu.setXAccelOffset(-1690);
mpu.setYAccelOffset(1157);
mpu.setZAccelOffset(2742);
```

3.2 Algoritma kontrol feedback

Algoritma yang digunakan untuk kontrol servo pada projek ini adalah close loop sederhana seperti pada gambar berikut.



Sensor gyro (IMU) hanya dapat membaca nilai percepatan perubahan sudut dan percepatan dari gerak translasi dengan rentang -2G sampai +2G .yang man Adapun untuk algoritma perhitungan sudut yaw, pitch, roll menggunakan library “MPU6050_6Axis_MotionApps20”. Berikut merupakan pengaturan output yaw, pitch, roll

```
#include "I2Cdev.h"
#include "Servo.h"
#include "MPU6050_6Axis_MotionApps20.h"
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
    #include "Wire.h"
#endif

// MPU control/status vars
bool dmpReady = false; // set true if DMP init was successful
uint8_t mpuIntStatus; // holds actual interrupt status byte from MPU
uint8_t devStatus; // return status after each device operation (0 = success,
    !0 = error)
uint16_t packetSize; // expected DMP packet size (default is 42 bytes)
uint16_t fifoCount; // count of all bytes currently in FIFO
uint8_t fifoBuffer[64]; // FIFO storage buffer

// orientation/motion vars
Quaternion q; // [w, x, y, z] quaternion container
VectorFloat gravity; // [x, y, z] gravity vector
```

```

float ypr[3];           // [yaw, pitch, roll]   yaw/pitch/roll container and gravi
ty vector

// =====
// ===                INTERRUPT DETECTION ROUTINE                ===
// =====

volatile bool mpuInterrupt = false;    // indicates whether MPU interrupt pin has
gone high
void ICACHE_RAM_ATTR dmpDataReady() {
    mpuInterrupt = true;
}

```

Setelah dapat memperoleh nilai hasil pembacaan nilai sudut roll, pitch, yaw. Nilai tersebut perlu difilter dengan menggunakan low pass filter. Setelah memperoleh nilai sudut tersebut nantinya akan digunakan sebagai koreksi nilai untuk menggerakkan servo

```

void loop() {
    if (mpu.dmpGetCurrentFIFOPacket(fifoBuffer)) { // Get the Latest packet
        // display Euler angles in degrees
        mpu.dmpGetQuaternion(&q, fifoBuffer);
        mpu.dmpGetGravity(&gravity, &q);
        mpu.dmpGetYawPitchRoll(ypr, &q, &gravity);

        if(statusKalibrasi == 0){ //proses kalibrasi
            dataYaw = ypr[0] * 180/M_PI;
            dataPitch= ypr[1] * 180/M_PI;
            dataRoll= ypr[2] * 180/M_PI;
            tSekarang = millis();
            dT = tSekarang - tSebelum;
            if(dT < WAKTU_CEK){
                yawSekarang=ypr[0];
                pitchSekarang=ypr[1];
                rollSekarang=ypr[2];
            }
            else{
                dYaw = abs(yawSekarang - yawSebelum);
                yawSebelum = yawSekarang;
                dRoll = abs(rollSekarang - rollSebelum);
                rollSebelum = rollSekarang;
                dPitch = abs(pitchSekarang - pitchSebelum);
                pitchSebelum = pitchSekarang;
                if(dYaw<ERROR_MINIMAL || dRoll<ERROR_MINIMAL || dPitch<ERROR_MINIMAL ){
                    counterKalibrasi++ ;
                }
                else counterKalibrasi = 0;
                if (counterKalibrasi>6){
                    statusKalibrasi = 1;
                    offsetSudutYaw = yawSekarang;

```

```

        offsetSudutPitch = pitchSekarang;
        offsetSudutRoll = rollSekarang;
    }
    tSebelum = tSekarang;
}
}
else{
    dataYaw = (ypr[0]-offsetSudutYaw) * 180/M_PI;
    dataPitch = (ypr[1]-offsetSudutPitch) * 180/M_PI;
    dataRoll = (ypr[2]-offsetSudutRoll) * 180/M_PI;

    servo1.write(dataRoll);
    servo2.write(dataPitch);
}
// kirim data setiap 5 detik
if ((millis()-last_millis)>5000){
    last_millis=millis();
    String protokol = "/update.php?sudutroll="+String(dataRoll)+
"&sudutpitch="+String(dataPitch);
    Serial.println( "Send Data To Server: Data Roll = "+String(dataRoll)+
" Data Pitch = "+String(dataPitch));
}
}
}
}

```

3.3 Pembuatan IoT

3.3.1 Pembuatan Protokol komunikasi

Inisiasi program koneksi internet dan koneksi host

```

// bagian IoT
#include <ESP8266WiFi.h>
const char* host = "gimbal.galium.ga";
const int httpPort = 80;
const char* ssid = "apoci";
const char* password = "apoci123";
long last_millis_wifi;
String wifi="OFF";
long last_millis;
WiFiClient client;

```

Pembuatan protokol komunikasi server dengan alat

```

// kirim data setiap 5 detik
if ((millis()-last_millis)>5000){
    last_millis=millis();
    String protokol = "/update.php?sudutroll="+String(dataRoll)+
"&sudutpitch="+String(dataPitch);
    Serial.println( "Send Data To Server: Data Roll = "+String(dataRoll)+
" Data Pitch = "+String(dataPitch));
}
}

```


Protokol komunikasi kirim nilai pembacaan dan mengambil nilai set point dari website

```
String send(String url) {
    if (!client.connect(host, httpPort)) {
        return "Error Request Confirm";
    }
    client.print(String("GET ") + url + " HTTP/1.1\r\n" +
        "Host: " + host + "\r\n" +
        "User-Agent: BuildFailureDetectorESP8266\r\n" +
        "Connection: close\r\n\r\n");

    String body = "";
    long last_millis_x=millis();
    long next_millis_x=millis();
    while (client.connected()) {
        if ((millis()-last_millis_x)>10000){
            goto next3;
        }
        next_millis_x=millis();
        while (client.available()) {
            if ((millis()-next_millis_x)>10000){
                goto next3;
            }
            char c = client.read();
            body += String(c);
        }
    }
    next3:
    client.stop();
    int a = body.indexOf("~");
    int b = body.indexOf("!");
    int c = body.indexOf("@");

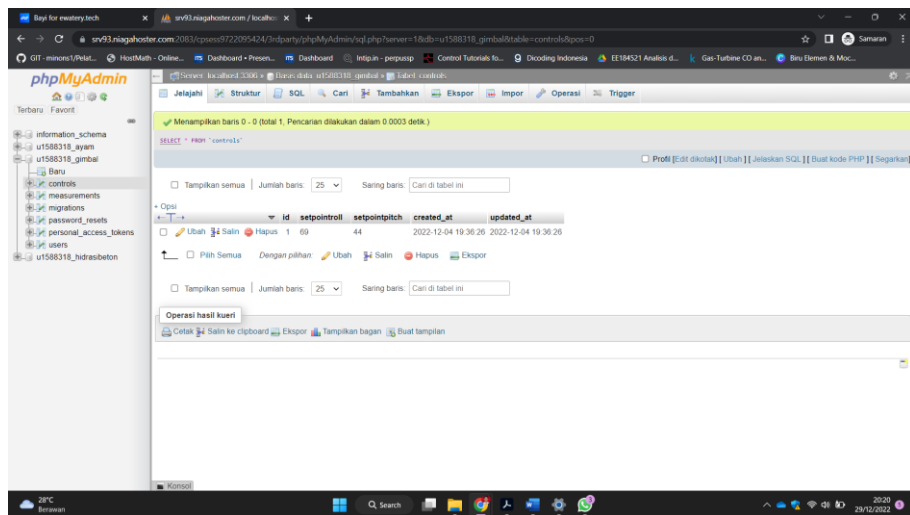
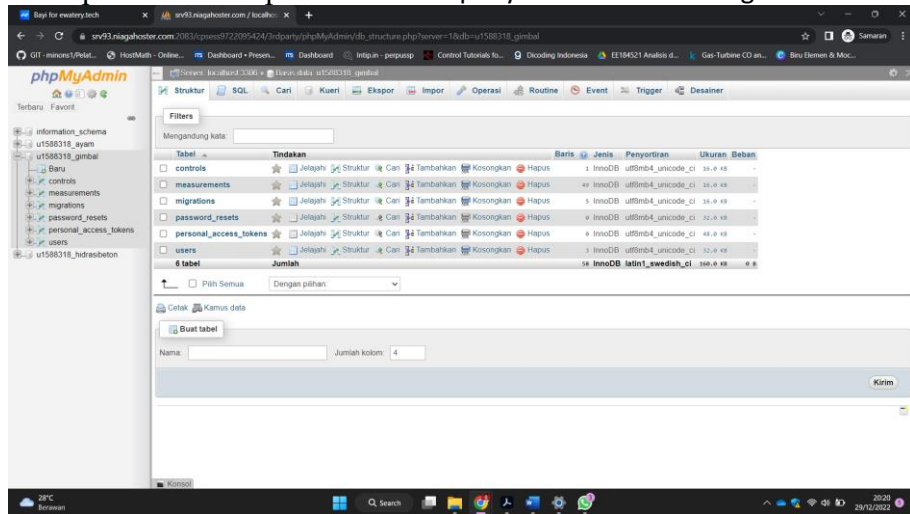
    String data_dataRoll = body.substring(a+1,b);
    String data_dataPitch = body.substring(b+1,c);

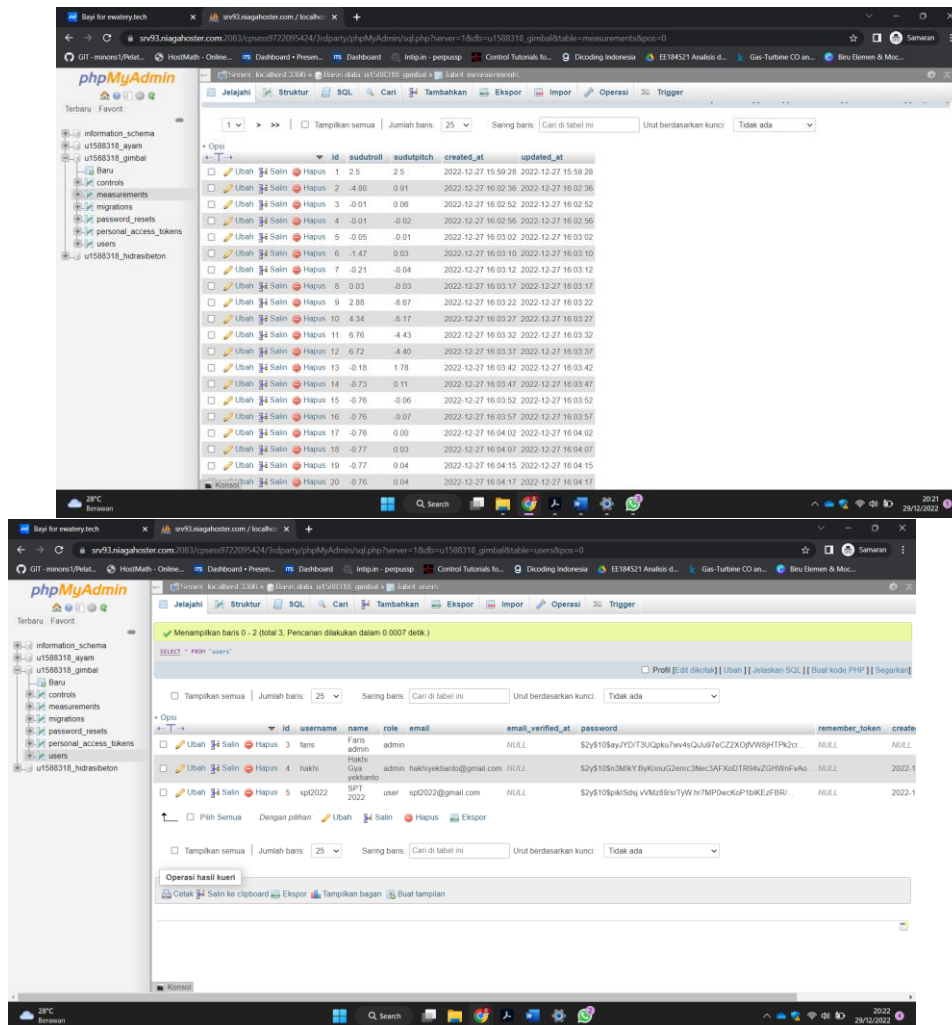
    if (data_dataRoll!=""&&data_dataPitch!=""){
        //Serial.println(body);
        Serial.println("SP="+data_dataRoll+" "+data_dataPitch);
        dataRoll = data_dataRoll.toInt();
        dataPitch = data_dataPitch.toInt();
        body="OK";
    }else{
        body = "ERROR";
    }

    return body;
}
```

3.1.2 Pembuatan *Database MySQL*

MySQL adalah open source Relational database management system yang sering digunakan untuk menyimpan data secara online (Open Source). MySQL digunakan dalam beberapa kasus, seperti pada automated system yang digunakan untuk mendapatkan data dan di input secara live. MySQL dapat disesuaikan dengan besar skala produk dan dapat melakukan penyesuaian sesuai dengan besar dari produk.





3.1.3 Hasil pengujian IoT

Hasil pengiriman data pada serial monitor

```
// X Accel Y Accel Z Accel X Gyro Y Gyro Z Gyro
//OFFSETS -1710, 1321, 2610, 37, -15, 17
Send Data To Server: Data Roll = -0.04 Data Pitch = -0.39
Send Data To Server: Data Roll = -0.11 Data Pitch = -0.51
Send Data To Server: Data Roll = 0.01 Data Pitch = -0.02
Send Data To Server: Data Roll = -0.06 Data Pitch = 0.40
Send Data To Server: Data Roll = -0.02 Data Pitch = 0.45
```

Hasil tampilan pengiriman data pada website

The screenshot displays a web application interface for 'gimbal.galung.ga/measurement'. The main content area shows a table titled 'Data Pengukuran' (Measurement Data). The table contains 10 rows of data, each representing a measurement point. The columns are 'No' (Number), 'Sudut Pitch' (Pitch Angle), 'Sudut Roll' (Roll Angle), and 'waktu pengukuran' (Measurement Time). The data is as follows:

No	Sudut Pitch	Sudut Roll	waktu pengukuran
1	-0.01°	-0.07°	2022-12-29 21:04:44
2	-0.02°	-0.07°	2022-12-29 21:04:49
3	0.00°	-0.06°	2022-12-29 21:04:54
4	5.90°	13.68°	2022-12-29 21:04:59
5	2.32°	-6.54°	2022-12-29 21:05:04
6	9.45°	2.46°	2022-12-29 21:05:09
7	4.14°	6.29°	2022-12-29 21:05:14
8	3.12°	2.00°	2022-12-29 21:05:19
9	1.52°	-33.12°	2022-12-29 21:05:24
10	-4.38°	27.93°	2022-12-29 21:05:29

The browser's address bar shows the URL 'gimbal.galung.ga/measurement'. The browser's taskbar at the bottom shows the date and time as 10:30 on 30/10/2022.

Lampiran

- Akses webiste
Link : <https://gimbal.galium.ga/>
User: spt2022
Pass: spt2022
- Code mikrokontroller

```
#include "I2Cdev.h"
#include "Servo.h"
#include "MPU6050_6Axis_MotionApps20.h"
#if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
    #include "Wire.h"
#endif

// bagian IoT
#include <ESP8266WiFi.h>
const char* host = "gimbal.galium.ga";
const int httpPort = 80;
const char* ssid = "apoci";
const char* password = "apoci123";
long last_millis_wifi;
String wifi="OFF";
long last_millis;
WiFiClient client;

//bagian kontrol
MPU6050 mpu;

Servo servo1; //servo bagian bawah (roll)
Servo servo2; //servo bagian atas (pitch)

//keperluan kalibrasi
#define ERROR_MINIMAL 0.001
#define WAKTU_CEK 1000 //dalam milisekon

long int tSekarang;
long int tSebelum;
long int dT;
double yawSekarang;
double yawSebelum;
double dYaw;
double pitchSekarang;
double pitchSebelum;
double dPitch;
double rollSekarang;
double rollSebelum;
double dRoll;
double bufferDataKalibrasi[10];
```

```

char counterKalibrasi;

double dataYaw;
double dataPitch;
double dataRoll;
char dataYawKirim[4];
char statusKalibrasi = 0;
double offsetSudutYaw;
double offsetSudutPitch;
double offsetSudutRoll;

#define INTERRUPT_PIN D8

// MPU control/status vars
bool dmpReady = false; // set true if DMP init was successful
uint8_t mpuIntStatus; // holds actual interrupt status byte from MPU
uint8_t devStatus; // return status after each device operation (0 = success, !=0 = error)
uint16_t packetSize; // expected DMP packet size (default is 42 bytes)
uint16_t fifoCount; // count of all bytes currently in FIFO
uint8_t fifoBuffer[64]; // FIFO storage buffer

// orientation/motion vars
Quaternion q; // [w, x, y, z] quaternion container
VectorFloat gravity; // [x, y, z] gravity vector
float ypr[3]; // [yaw, pitch, roll] yaw/pitch/roll
and gravity vector

// =====
// == INTERRUPT DETECTION ROUTINE ==
// =====

volatile bool mpuInterrupt = false;
void ICACHE_RAM_ATTR dmpDataReady() {
    mpuInterrupt = true;
}

// =====
// == INITIAL SETUP ==
// =====

void setup() {

    // join I2C bus (I2Cdev library doesn't do this automatically)
    #if I2CDEV_IMPLEMENTATION == I2CDEV_ARDUINO_WIRE
        Wire.begin();
        Wire.setClock(400000);
    #elif I2CDEV_IMPLEMENTATION == I2CDEV_BUILTIN_FASTWIRE

```

```

    Fastwire::setup(400, true);
#endif

Serial.begin(115200);
Serial.print("void setup initialized");

servo1.attach(D6);
servo2.attach(D5);

servo1.write(0);          //ATUR POSISI AWAL SERVO
servo2.write(0);

delay(1000);

// konek wifi
WiFi.mode(WIFI_STA);
WiFi.begin(ssid, password);
Serial.print("Connecting.....");
Serial.print(ssid);

Serial.print("SUCCESS!");
Serial.println("");
Serial.println("WiFi connected");
Serial.println("IP address: ");
Serial.println(WiFi.localIP());
mpu.initialize();
pinMode(INTERRUPT_PIN, INPUT);
devStatus = mpu.dmpInitialize();
pinMode(INTERRUPT_PIN, INPUT);

// nilai offset hasil kalibrasi
mpu.setXGyroOffset(45);
mpu.setYGyroOffset(-27);
mpu.setZGyroOffset(18);
mpu.setXAccelOffset(-1690);
mpu.setYAccelOffset(1157);
mpu.setZAccelOffset(2742);

if (devStatus == 0) {
    // Calibration Time: generate offsets and calibrate our MPU6050
    mpu.CalibrateAccel(7); //Kalibrasi Pertama kali
    mpu.CalibrateGyro(7);
    mpu.PrintActiveOffsets();

    mpu.setDMPEnabled(true);

    attachInterrupt(digitalPinToInterrupt(INTERRUPT_PIN), dmpDataReady, RISING);
}

```

```

    mpuIntStatus = mpu.getIntStatus();

    dmpReady = true;

    packetSize = mpu.dmpGetFIFOPageSize();
}
else{
    Serial.print(F("DMP Initialization failed (code "));
    Serial.print(devStatus);
    Serial.println(F(""));
}
}

// untuk kalibrasi awal
// void loop(){

// }

// =====
// ==                               MAIN PROGRAM LOOP                               ==
// =====

void loop() {
    if (mpu.dmpGetCurrentFIFOPacket(fifoBuffer)) { // Get the Latest packet
        // display Euler angles in degrees
        mpu.dmpGetQuaternion(&q, fifoBuffer);
        mpu.dmpGetGravity(&gravity, &q);
        mpu.dmpGetYawPitchRoll(ypr, &q, &gravity);

        if(statusKalibrasi == 0){ //proses kalibrasi
            dataYaw = ypr[0] * 180/M_PI;
            dataPitch= ypr[1] * 180/M_PI;
            dataRoll= ypr[2] * 180/M_PI;
            tSekarang = millis();
            dT = tSekarang - tSebelum;
            if(dT < WAKTU_CEK){
                yawSekarang=ypr[0];
                pitchSekarang=ypr[1];
                rollSekarang=ypr[2];
            }
            else{
                dYaw = abs(yawSekarang - yawSebelum);
                yawSebelum = yawSekarang;
                dRoll = abs(rollSekarang - rollSebelum);
                rollSebelum = rollSekarang;
                dPitch = abs(pitchSekarang - pitchSebelum);
                pitchSebelum = pitchSekarang;
            }
        }
    }
}

```



```

        if(dYaw<ERROR_MINIMAL || dRoll<ERROR_MINIMAL || dPitch<ERROR_MINIMAL )
    {
        counterKalibrasi++ ;
    }
    else counterKalibrasi = 0;
    if (counterKalibrasi>6){
        statusKalibrasi = 1;
        offsetSudutYaw = yawSekarang;
        offsetSudutPitch = pitchSekarang;
        offsetSudutRoll = rollSekarang;
    }
    tSebelum = tSekarang;
}
}
else{
    dataYaw = (ypr[0]-offsetSudutYaw) * 180/M_PI;
    dataPitch = (ypr[1]-offsetSudutPitch) * 180/M_PI;
    dataRoll = (ypr[2]-offsetSudutRoll) * 180/M_PI;

    // int servo1Value = map(dataRoll, -90, 90, 0, 180);
    // int servo2Value = map(dataPitch, -90, 90, 180, 0);

    // Serial.print("\t");
    // Serial.print(servo1Value);
    // Serial.print("\t");
    // Serial.print(servo2Value);
    // Serial.print("\t");
    // Serial.print(dataYaw);
    // Serial.print("\t");
    // Serial.print(dataPitch);
    // Serial.print("\t");
    // Serial.println(dataRoll);

    servo1.write(dataRoll);
    servo2.write(dataPitch);
}
// kirim data setiap 5 detik
if ((millis()-last_millis)>5000){
    last_millis=millis();
    String protokol = "/update.php?sudutroll="+String(dataRoll)+
"&sudutpitch="+String(dataPitch);
    Serial.println( "Send Data To Server: Data Roll = "+String(dataRoll)+
" Data Pitch = "+String(dataPitch));
}
}
}
}

```

```

String send(String url) {
    if (!client.connect(host, httpPort)) {
        return "Error Request Confirm";
    }
    client.print(String("GET ") + url + " HTTP/1.1\r\n" +
        "Host: " + host + "\r\n" +
        "User-Agent: BuildFailureDetectorESP8266\r\n" +
        "Connection: close\r\n\r\n");

    String body = "";
    long last_millis_x=millis();
    long next_millis_x=millis();
    while (client.connected()) {
        if ((millis()-last_millis_x)>10000){
            goto next3;
        }
        next_millis_x=millis();
        while (client.available()) {
            if ((millis()-next_millis_x)>10000){
                goto next3;
            }
            char c = client.read();
            body += String(c);
        }
    }
    next3:
    client.stop();
    int a = body.indexOf("~");
    int b = body.indexOf("!");
    int c = body.indexOf("@");

    String data_dataRoll = body.substring(a+1,b);
    String data_dataPitch = body.substring(b+1,c);

    if (data_dataRoll!=""&&data_dataPitch!=""){
        //Serial.println(body);
        Serial.println("SP="+data_dataRoll+" "+data_dataPitch);
        dataRoll = data_dataRoll.toInt();
        dataPitch = data_dataPitch.toInt();
        body="OK";
    }else{
        body ="ERROR";
    }

    return body;
}

```