dem
universidade de aveiro
departamento de engenharia mecânica

Implementação de ESP32 c/ MPU6050 e WIFI

Gonçalo Rosa nº 102988
Gonçalo Aguiar nº 104058

8 de Maio de 2025

Conteúdo

Introdução

Implementação de ESP32 c/ MPU6050 e WIFI Com o avanço das tecnologias embarcadas e da comunicação sem fio, a implementação de sistemas de sensoriamento inteligente em ambientes robóticos tem se tornado cada vez mais acessível e essencial. Este relatório aborda a integração do microcontrolador ESP32, que é um dispositivo de alto desempenho com capacidade de comunicação sem fio via Wi-Fi e Bluetooth, com o sensor MPU6050, que combina um acelerômetro triaxial e um giroscópio triaxial. Juntos, esses componentes formam uma plataforma que pode ser aplicada em ambientes dinâmicos e exigentes, como em robôs industriais.

Embora a implementação não tenha sido fisicamente realizada em um robô Delta, o projeto foi desenvolvido e avaliado em um ambiente simulado utilizando a plataforma Wokwi, que permite simular o comportamento dos componentes de hardware e software. A utilização do ESP32, com seu poder de processamento e conectividade, permite que o sistema seja não apenas eficiente, mas também altamente flexível, sendo possível integrar dados de sensores inerciais em tempo real. O sensor MPU6050, por sua vez, é fundamental para captar as variações de movimento em um robô, fornecendo dados críticos para controle de precisão e estabilidade.

Os robôs do tipo Delta são amplamente utilizados em processos industriais que exigem alta precisão e velocidade, como em sistemas de pick-and-place, que realizam o transporte de objetos de forma automatizada. A implementação de sensores inerciais no end effector, a parte final do braço do robô responsável pela interação com o ambiente, pode otimizar o controle e proporcionar um monitoramento dinâmico, o que aumenta a robustez do sistema e a capacidade de diagnóstico de falhas. Este relatório, portanto, descreve os passos dados para simular a integração desses componentes em uma plataforma de baixo custo, com vistas a uma futura implementação física.

Objetivo Geral e Específicos

O objetivo geral deste projeto é desenvolver e avaliar, em ambiente simulado, uma plataforma de monitoramento inercial sem fio baseada no ESP32 e no MPU6050, com potencial aplicação futura no end effector de um robô Delta.

Objetivos específicos:

- Realizar a aquisição de dados de aceleração e rotação em três eixos via simulação;
- Transmitir os dados via Wi-Fi simulada para uma interface de monitoramento remoto;
- Avaliar o desempenho do código e sua estabilidade em ambiente simulado;
- Documentar o sistema para no futuro implementar no robo físico.

Fundamentação Teórica

A implementação do sistema descrito neste relatório faz uso de dois componentes principais: o microcontrolador ESP32 e o sensor inercial MPU6050. Estes dispositivos são utilizados para criar um sistema de monitoramento dinâmico e controle de movimento, que pode ser integrado ao *end effector* de um robô Delta. A seguir, é apresentada uma análise técnica detalhada desses componentes, com suas características e aplicações em sistemas robóticos.

3.1 MPU6050 - Sensor Inercial de 6 Eixos

O MPU6050 é um sensor MEMS (*Micro-Electro-Mechanical System*) que combina um acelerômetro triaxial e um giroscópio triaxial. Ele permite a medição de acelerações lineares e rotações em torno de três eixos ortogonais, oferecendo um total de 6 graus de liberdade (DoF). Essa combinação de sensores é extremamente útil em aplicações que exigem precisão na medição do movimento e orientação, como em robôs, drones e dispositivos de navegação.

A comunicação do MPU6050 com o microcontrolador é feita através do protocolo I2C, o que facilita a integração em sistemas com recursos limitados de pinos de entrada e saída.

Abaixo está uma tabela com as principais características técnicas do MPU6050:

Parâmetro	**Valor**
Tecnologia de Sensor	MEMS (Micro-Electro-Mechanical System)
Componentes Integrados	Acelerômetro triaxial e Giroscópio triaxial
Faixa de Medição (Acelerômetro)	$\pm 2g$, $\pm 4g$, $\pm 8g$, $\pm 16g$
Faixa de Medição (Giroscópio)	$\pm 250°/s$, $\pm 500°/s$, $\pm 1000°/s$, $\pm 2000°/s$
Precisão	Acelerômetro: 0.001g, Giroscópio: 0.01°/s
Tensão de Operação	2.3V a 3.4V
Protocolo de Comunicação	I2C
Consumo de Energia	3.9mA (em operação normal)

Tabela 3.1: Características técnicas do MPU6050.

O MPU6050 é amplamente utilizado em sistemas de navegação inercial, estabilização de câmeras e em plataformas móveis que exigem sensoriamento de movimento. Sua alta precisão e flexibilidade fazem dele uma escolha ideal para o monitoramento do movimento de um robô Delta, permitindo correções dinâmicas em tempo real durante a operação.

3.2 ESP32 - Microcontrolador com Conectividade Wi-Fi/Bluetooth

O ESP32 é um microcontrolador de 32 bits com arquitetura dual-core desenvolvido pela Espressif Systems. Este microcontrolador se destaca por sua alta capacidade de processamento e recursos de conectividade, como Wi-Fi e Bluetooth, o que o torna ideal para aplicações de Internet das Coisas (IoT) e para sistemas que exigem comunicação sem fio.

O ESP32 oferece uma ampla gama de pinos de entrada e saída, recursos de processamento paralelo e baixa latência, tornando-o altamente eficiente para o controle de sistemas embarcados que exigem a aquisição de dados em tempo real, como no caso de sistemas de sensoriamento inercial.

O ESP32 é amplamente utilizado em sistemas embarcados devido ao seu custo acessível, alto desempenho e versatilidade. Sua conectividade sem fio é um diferencial importante, permitindo que os dados do

sistema, como os provenientes do sensor MPU6050, sejam transmitidos para servidores ou interfaces de monitoramento em tempo real, facilitando o controle remoto e a coleta de dados.

Abaixo está uma tabela com as principais características técnicas do ESP32:

Parâmetro	**Valor**
Arquitetura	32 bits, Dual-core
Frequência de Operação	160 MHz a 240 MHz
Memória RAM	520 KB SRAM
Memória Flash	4 MB (padrão)
Conectividade	Wi-Fi (802.11 b/g/n) e Bluetooth (v4.2, BLE)
Tensão de Operação	3.3V
Consumo de Energia	160 mA (em operação normal)
GPIOs	34 pinos configuráveis para entradas/saídas digitais, analógicas, PWM
Protocolo de Comunicação	I2C, SPI, UART, CAN, etc.

Tabela 3.2: Características técnicas do ESP32.

3.3 End Effector e Robôs Delta

O end effector corresponde a base inferior do robo delta e é responsável pela interação direta com o ambiente. Em robôs industriais, o end effector pode ser uma garra, uma ferramenta de soldagem, uma câmera ou qualquer outro dispositivo de execução. Em robôs Delta, que utilizam uma estrutura de cinemática paralela, o end effector é projetado para ser altamente preciso e rápido, sendo ideal para tarefas como pick-and-place e montagem automatizada.

A implementação de sensores inerciais, como o MPU6050, no end effector de um robô Delta pode aumentar a capacidade do sistema de se ajustar dinamicamente durante a operação. Com os dados fornecidos pelo acelerômetro e giroscópio, é possível realizar correções em tempo real, melhorando a precisão do movimento e a estabilidade do robô. A capacidade de monitorar a aceleração e a rotação do end effector é essencial para detectar falhas de movimentação ou desajustes no sistema de controle.

A adição de sensores inerciais ao end effector de robôs Delta proporciona vantagens significativas, como a melhoria da precisão de movimento e a otimização do controle. A integração do sensor de movimento permite que o robô seja mais adaptável e eficiente, realizando tarefas complexas com maior rapidez e sem erros.

3.4 Desafios e Oportunidades

A utilização de sensores inerciais em sistemas robóticos, embora extremamente útil, apresenta desafios relacionados ao processamento em tempo real e à comunicação de dados. O uso de microcontroladores como o ESP32, que possuem capacidade de processamento paralelo e conectividade sem fio, torna a integração de sistemas como o descrito neste trabalho mais simples e eficiente. No entanto, os dados provenientes de sensores inerciais podem ser afetados por ruídos e imprecisões, o que exige o uso de algoritmos de filtragem para garantir a precisão das medições. Técnicas como o filtro de Kalman ou o filtro complementar são frequentemente aplicadas para melhorar a qualidade dos dados e garantir a estabilidade do sistema.

A implementação de filtros de sinal e o processamento eficiente dos dados adquiridos pelo MPU6050 são fundamentais para garantir que os movimentos do robô Delta sejam realizados de maneira precisa e sem falhas. Esses desafios, no entanto, também representam oportunidades de pesquisa e desenvolvimento, pois a melhoria desses algoritmos pode resultar em sistemas robóticos ainda mais precisos e eficientes.

3.5 Materiais e Métodos

3.5.1 Componentes Utilizados (Simulados)

Componente	Quantidade	Descrição
ESP32 DevKit V1	1	Microcontrolador com Wi-Fi/Bluetooth
Sensor MPU6050	1	Acelerômetro e Giroscópio 3-eixos
Ambiente Wokwi	1	Simulador online de dispositivos embarcados
Arduino IDE	1	Ambiente de desenvolvimento

3.6 Esquemático de Ligação (Simulado)

O sensor foi conectado ao ESP32 via interface I2C, conforme tabela abaixo:

MPU6050	ESP32
VCC	3.3V
GND	GND
SDA	GPIO21
SCL	GPIO22

3.6.1 Desenvolvimento de Software

O software foi desenvolvido em C++ utilizando a IDE do Arduino. Para a implementação, foram empregadas as seguintes bibliotecas: Wire.h, responsável pela comunicação via protocolo I2C; MPU6050.h, destinada à leitura dos sensores inerciais; WiFi.h, que simula a conexão com uma rede sem fio; e HTTPClient.h, utilizada para simular o envio de dados para um servidor remoto. O funcionamento do sistema segue um ciclo contínuo que se inicia com a inicialização do sensor MPU6050 e da rede Wi-Fi. Em seguida, os dados de aceleração e rotação são lidos e impressos no console da simulação. Após esse processo, o sistema aguarda por um curto intervalo e reinicia o ciclo, repetindo essas etapas de forma contínua.

Utilização software wokwi

A seguir foi construir o código responsável por integrar o sensor MPU6050 com um display LCD de 16x2. O objetivo é capturar os dados de aceleração e rotação do sensor MPU6050 e exibí-los em tempo real no LCD.

Este código tem como objetivo ler os dados de aceleração e rotação de um sensor MPU6050, que possui um acelerômetro e um giroscópio, e exibi-los em um display LCD utilizando a comunicação I2C.

```
1   #include <Wire.h>
2   #include <Adafruit_MPU6050.h>
3   #include <Adafruit_Sensor.h>
4   #include <LiquidCrystal_I2C.h>
```

Figura 4.1: Livrarias do wokwi

O código inicia com a inclusão das bibliotecas essenciais para o funcionamento do sistema. A biblioteca `Wire.h` é utilizada para estabelecer a comunicação I2C entre o microcontrolador (como o Arduino ou ESP32) e os dispositivos periféricos, como o sensor MPU6050 e o display LCD. A biblioteca `Adafruit_MPU6050.h` permite a interação direta com o sensor inercial MPU6050, enquanto `Adafruit_Sensor.h` fornece uma interface genérica para a leitura e manipulação dos dados provenientes de sensores diversos. Já a biblioteca `LiquidCrystal_I2C.h` facilita o controlo de displays LCD com interface I2C, simplificando a ligação com o microcontrolador e reduzindo o número de pinos necessários.

Após a inclusão das bibliotecas, são criados dois objetos principais. O primeiro é `mpu`, uma instância da classe `Adafruit_MPU6050`, que representa o sensor inercial e permite aceder diretamente aos seus dados. O segundo é `lcd`, um objeto da classe `LiquidCrystal_I2C`, utilizado para controlar o display LCD. O endereço I2C do display é configurado como `0x27` e o seu tamanho é definido para 16 colunas e 2 linhas, permitindo a exibição de mensagens e dados formatados durante a execução do programa.

```
1   #include <Wire.h>
2   #include <Adafruit_MPU6050.h>
3   #include <Adafruit_Sensor.h>
4   #include <LiquidCrystal_I2C.h>
```

Figura 4.2: Objetos criados

Na função setup(), a comunicação serial é iniciada com a taxa de transmissão de 115200 bauds, permitindo a depuração ou envio de dados para o monitor serial. O LCD é inicializado com as dimensões definidas (16x2) e exibe a mensagem ”Iniciando...”, informando ao usuário que o sistema está sendo configurado.

O código tenta inicializar o sensor MPU6050. Caso o sensor não seja encontrado, é exibida uma mensagem de erro no monitor serial e no LCD, e o programa entra em um loop infinito, interrompendo a execução. Se o sensor for inicializado com sucesso, as configurações do acelerômetro e giroscópio são feitas. A faixa do acelerômetro é configurada para ±2g, a do giroscópio para ±250°/s, e o filtro digital é configurado para uma largura de banda de 21 Hz. Após as configurações, é inserido um pequeno atraso de 100 milissegundos para garantir que tudo esteja corretamente inicializado.

```
void setup() {
  // Inicia a comunicação serial
  Serial.begin(115200);

  // Inicializa o LCD
  lcd.begin(16, 2);
  lcd.print("Iniciando...");

  // Inicializa o MPU6050
  if (!mpu.begin()) {
    Serial.println("Não foi possível encontrar o MPU6050!");
    lcd.clear();
    lcd.print("Erro no MPU6050");
    while (1);
  }

  // Calibra o acelerômetro e giroscópio
  mpu.setAccelerometerRange(MPU6050_RANGE_2_G);
  mpu.setGyroRange(MPU6050_RANGE_250_DEG);
  mpu.setFilterBandwidth(MPU6050_BAND_21_HZ);

  // Atraso para garantir que tudo inicialize corretamente
  delay(100);
}
```

Figura 4.3: Void setup

Na função `loop()`, o código entra num ciclo contínuo de leitura e exibição dos dados provenientes do sensor inercial. As variáveis `a`, `g` e `temp` são instâncias da estrutura `sensors_event_t`, utilizada para armazenar eventos captados por sensores da biblioteca `Adafruit_Sensor`. Essas estruturas são utilizadas para guardar, respetivamente, os dados de aceleração, giroscópio e temperatura.

A função `mpu.getEvent(a, g, temp)` é chamada dentro do ciclo principal e realiza a leitura dos dados atuais do MPU6050, atribuindo automaticamente os valores recolhidos às variáveis correspondentes. Desta forma, o sistema obtém continuamente os valores de aceleração linear, rotação angular e temperatura interna do sensor, que podem depois ser utilizados para exibição em display ou transmissão por rede.

Após exibir os dados, o código aguarda 500 milissegundos com delay(500), permitindo uma atualização das informações no LCD a cada meio segundo.

```
void loop() {
  // Variáveis para armazenar os dados do sensor
  sensors_event_t a, g, temp;

  // Lê os dados do sensor
  mpu.getEvent(&a, &g, &temp);

  // Exibe os dados no LCD
  lcd.clear();
  lcd.setCursor(0, 0);
  lcd.print("Acel. X: ");
  lcd.print(a.acceleration.x, 1);

  lcd.setCursor(0, 1);
  lcd.print("Gyro. X: ");
  lcd.print(g.gyro.x, 1);

  // Espera 500ms para atualizar os dados no LCD
  delay(500);
}
```

Figura 4.4: Void loop

Conclusão

Este projeto demonstrou com sucesso a integração entre o microcontrolador ESP32 e o sensor inercial MPU6050, utilizando um ambiente de simulação virtual disponibilizado pela plataforma Wokwi. Através do desenvolvimento e análise do código em Arduino, foi possível validar a leitura dos dados de aceleração, rotação e temperatura, bem como a sua exibição em tempo real num display LCD via interface I2C.

A simulação apesar de nao concluida permitiria comprovar a estabilidade do sistema e a eficiência das bibliotecas utilizadas, evidenciando que a arquitetura proposta é viável para futuras aplicações físicas em robôs industriais, nomeadamente em robôs Delta, onde a monitorização dinâmica do *end effector* é crucial para garantir precisão e robustez operativa. O uso do ESP32 revelou-se uma escolha acertada devido às suas capacidades de conectividade sem fio e processamento paralelo.

Apesar de os testes não terem sido realizados por completo, os resultados obtidos são encorajadores para a próxima fase do projeto: a implementação em hardware real. Futuramente, pretende-se integrar algoritmos de filtragem, como o filtro de Kalman, para melhorar a qualidade dos dados obtidos e desenvolver uma interface de monitorização em tempo real, aumentando a funcionalidade e o potencial de aplicação do sistema em contextos industriais exigentes.

Bibliografia

[1] InvenSense. *MPU-6050 Datasheet*. Disponível em: https://invensense.tdk.com/products/motion-tracking/6-axis/mpu-6050/. Acesso em: mai. 2025.

[2] Espressif Systems. *ESP32 Series Datasheet*. Disponível em: https://www.espressif.com/en/products/socs/esp32/resources. Acesso em: mai. 2025.

[3] Wokwi. *Wokwi Documentation – Simulating Arduino Projects*. Disponível em: https://docs.wokwi.com/. Acesso em: mai. 2025.

[4] Wokwi. *Getting Started with Wokwi Simulator*. Disponível em: https://blog.wokwi.com/getting-started-with-wokwi/. Acesso em: mai. 2025.