AI & SUSTAINABILITY IN VET EDUCATION

Nome do(s) autor(es): Sucessos Criativos, Lda

Título do WP: WP3 – IA-SAVES@ProjectHub

Referência de Atividade: 3.2 – Modelo de projeto para IA-SAVES@ProjectHub

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.

Índice

1. Título do Projeto: 4
2. Categoria do Projeto: 4
3. Descrição do Projeto: 4
4. Palavras-chave: 4
5. Público-alvo: 4
6. Objetivos do Projeto: 4
7. Cronologia do Projeto: 5
8. Recursos Fornecidos: 5
9. Ferramentas e Tecnologias Utilizadas: 6
10. Plano de Implementação Passo a Passo: 7
Passo 1: Prepare o Raspberry Pi para Interação de Hardware 7
Passo 2: Ligue o servomotor 7
Passo 4: Executar o Programa 10
11. Critérios de Avaliação: 11
12. Línguas Disponíveis: 12
13. Colaboradores: 12
17. Feedback e Melhoria Contínua: 12

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.

1. Título do Projeto:

Primeiros Passos com o Kit AI-VET: Controlar um Servomotor com Raspberry Pi 5

---

2. Categoria do Projeto:

Fundamentos da Microeletrónica

---

3. Descrição do Projeto:

Este projeto apresenta aos participantes o controlo do movimento através de servomotores utilizando o Raspberry Pi 5. Baseando-se na configuração anterior e no projeto de piscar LED, esta atividade demonstra como usar o sistema GPIO para enviar sinais PWM (Modulação de Largura de Pulso) para rodar um servo em ângulos específicos. Ensina aos participantes como o código pode controlar o movimento e serve de base para futuros projetos de automação ou braços robóticos.

---

4. Palavras-chave:

Raspberry Pi, Python, Ambiente Virtual, GPIO, Servomotor, PWM, Geany

---

5. Público-alvo:

Educadores, aprendizes e criadores, com ou sem experiência prévia, que estão a iniciar o seu percurso na microeletrónica e computação física.

---

6. Objetivos do Projeto:

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.

- Revise e confirme que o ambiente Raspberry Pi e as bibliotecas GPIO estão prontos.
- Constrói um circuito simples para ligar um servomotor.
- Escreve e executa um script em Python para rodar o servo para ângulos específicos.
- Compreenda como os sinais PWM controlam o movimento do servo.

---

7. Cronologia do Projeto:

A duração estimada é de aproximadamente **2,5 a 3 horas**, incluindo a montagem do ambiente e a montagem física.

---

8. Recursos Fornecidos:

- AI-VET Kit (Raspberry Pi 5, LED, Breadboard, Jumpers, Micro Servo)

MATERIAL	
Raspberry Pi 5	
Fonte de alimentação USB Tipo-C de 27W para Raspberry Pi 5	

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.

Cabos Flat Jumper F/F	
Breadboard	
FIO EM FILA ÚNICA PARA PLACAS DE ENSAIO	
SG90 1,6Kg 180º Microservo (IM4)	

---

9. Ferramentas e Tecnologias Utilizadas:

- Raspberry Pi OS
- Python 3 (em ambiente virtual)
- `Biblioteca LGPIO`
- Geany IDE
- Terminal (bash)

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.

10. Plano de Implementação Passo a Passo:

Passo 1: Prepare o Raspberry Pi para Interação de Hardware

Este projeto baseia-se diretamente nas tarefas concluídas em "Começar com Raspberry Pi 5: Configuração, Configuração e Hello World em Python." Se está a começar por aqui, ou precisa de uma atualização, deve primeiro passar pelos passos essenciais de configuração para preparar o seu Raspberry Pi para controlar componentes físicos como um servomotor.

Neste passo, irá:

- Instale o Raspberry Pi OS usando a ferramenta Raspberry Pi Imager.
- Configure definições básicas do sistema, como Wi-Fi, layout do teclado e fuso horário.
- Ative interfaces chave incluindo SPI, I2C, 1-Wire e VNC usando o menu de configuração do Raspberry Pi.
- Identifique o endereço IP do Raspberry Pi usando o ifconfig e ligue-se remotamente via TigerVNC Viewer.
- Realizar uma atualização completa do sistema
- Instalar a biblioteca lgpio

Passo 2: Ligue o servomotor

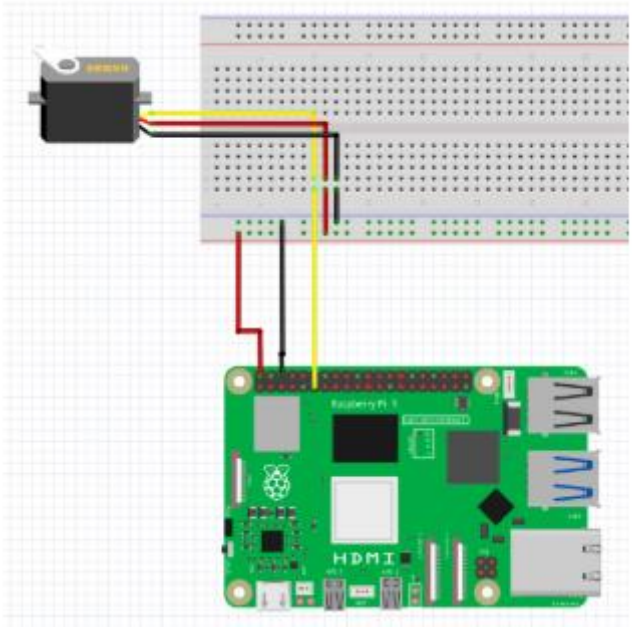
Neste passo, irá construir um circuito simples para controlar um servomotor usando o Raspberry Pi 5. O servo tem três fios: energia, terra e sinal.

Para ligar o servo:

- Liga o fio castanho ou preto (GND) ao pino físico 6 do Raspberry Pi.
- Ligue o fio vermelho (5V) ao pino físico 2 ou ao pino 4.
- Ligue o fio laranja ou amarelo (sinal) ao GPIO 18 (pino físico 12).

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.



Passo 3: Escreva o programa Python para controlar o servo

Agora que o servo está ligado, vais escrever um script em Python para o rodar para múltiplas posições usando a biblioteca lgpio. Esta biblioteca permite um controlo preciso de PWM (Modulação de Largura de Pulso), necessário para operar servomotores.

Abra o Geany e introduza o seguinte código:

```
Importação da LGPIO
Tempo de importação
# Pino GPIO para o servo (ex., GPIO 17)
SERVO_PIN = 17
# Inicializar o chip GPIO (o chip GPIO padrão é normalmente "/dev/gpiochip0")
h = lgpio.gpiochip_open(0)
# Reivindicar o pino GPIO para PWM (controlo servo)
lgpio.gpio_claim_output(h, SERVO_PIN)
# Função para gerar PWM para servomotor
Def set_servo_pulsewidth(h, pin, pulse_width):
    """Envia um sinal alto para o pino servo para a largura do pulso (em
milissegundos)."""
 lgpio.gpio_write(h, pino 1) # Definir pino ALTO (início do pulso)
```

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.

```
 time.sleep(pulse_width / 1000) # Aguarde pela largura do pulso, duração (a
largura do pulso é em milissegundos)
    lgpio.gpio_write(h, pino, 0)            # Definir pino BAIXO (fim do
pulso)
# Função para mover o servo para um ângulo especificado
def move_servo(ângulo):
    """
    Converter o ângulo (entre 0 e 180 graus) para a largura de pulso
correspondente
    (entre 1ms e 2ms) e move o servo para esse ângulo.
    """
    pulse_width = 1 + (ângulo / 180,0) # Mapear ângulo para largura de pulso
em milissegundos (1ms a 2ms)
    set_servo_pulsewidth(h, SERVO_PIN, pulse_width)  # Gerar sinal PWM para a
largura de pulso calculada
# Laço principal para controlar o servo
Experimente:
    embora Verdadeiro:
        # Move o servo para a posição do meio (90 graus)
        print ("A passar para a posição do meio")
        move_servo(90)                      # Avançar para 90 graus
        time.sleep(0.5)                     # Pequeno atraso
        # Mover o servo para a posição mínima (0 graus)
        print ("A mover para a posição mínima")
        move_servo(0)                       # Avançar para 0 graus
        time.sleep(1)                       # Atraso maior
        # Move o servo para a posição central novamente (90 graus)
        print ("A passar para a posição do meio")
        move_servo(90)                      # Volta a 90 graus
        time.sleep(0.5)                     # Pequeno atraso
        # Mover servo para a posição máxima (180 graus)
        print ("A mover para a posição máxima")
        move_servo(180)                     # Avançar para 180 graus
        time.sleep(1)                       # Atraso maior
exceto o KeyboardInterrupt:
    # Tratar do caso quando o utilizador interrompe o programa (Ctrl + C)
    print ("Controlo servo de paragem")
    tempo.sono(1)
    lgpio.gpio_write(h, SERVO_PIN, 0)      # Desliga o SERVO_PIN antes de sair
    lgpio.gpiochip_close(h)
```

Decomposição do Código:

Configuração e Importações:
O script começa por importar as bibliotecas necessárias: `lgpio` para controlar o GPIO e `tempo` para gerir atrasos.

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.

Definição e Inicialização de Pinos: Configura **o GPIO 17** como pino de saída para o sinal servo. O chip GPIO é aberto e o pino é reclamado para saída para que possamos enviar sinais precisos de ligar/desligar.

Função PWM: Uma função chamada `set_servo_pulsewidth` envia manualmente um sinal alto (pulso) para o pino durante um curto período — o comprimento deste pulso indica ao servo para que ângulo deve rodar. Para servos, isto normalmente varia entre **1 ms (0°)** e **2 ms (180°).**

Mapeamento de Ângulos: Outra função, `move_servo`, converte ângulos como 0°, 90° e 180° na largura de pulso correta. Utiliza uma fórmula simples para calcular a duração com base no ângulo, e depois chama a função PWM.

Circuito Principal: Dentro do laço principal, o servo move-se entre diferentes ângulos (90 → 0 → 90 → 180) com pequenas pausas entre eles. Consegues ver claramente o movimento a cada passo.

Gestão de Interrupções: Se pressionar **Ctrl + C** para parar o programa, desliga a ligação GPIO com segurança e desliga o servo, evitando tremores ou movimentos bruscos.

Passo 4: Executar o Programa

Depois de o teu código estar escrito e guardado (por exemplo, servo_manual_pwm.py), podes executá-lo usando o Geany ou o terminal. O servo começa a alternar entre as posições definidas.

Para executar o programa:

- Clica no botão Run em Geany (ou carrega em F5).

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.

- Alternativamente, no terminal:

  python servo_manual_pwm.py

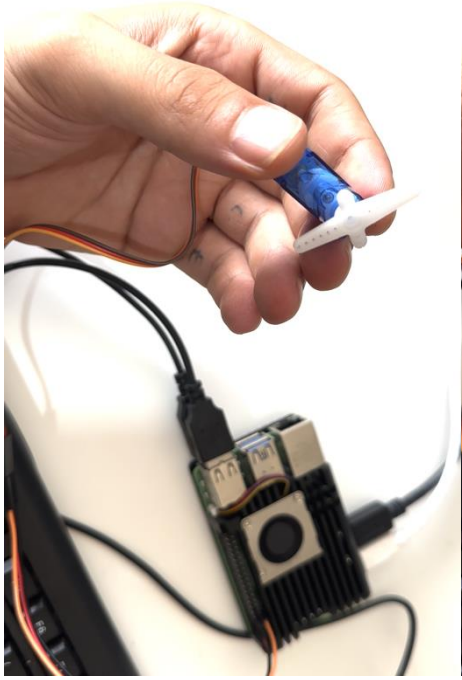


Figura 1 - Servo na posição central.

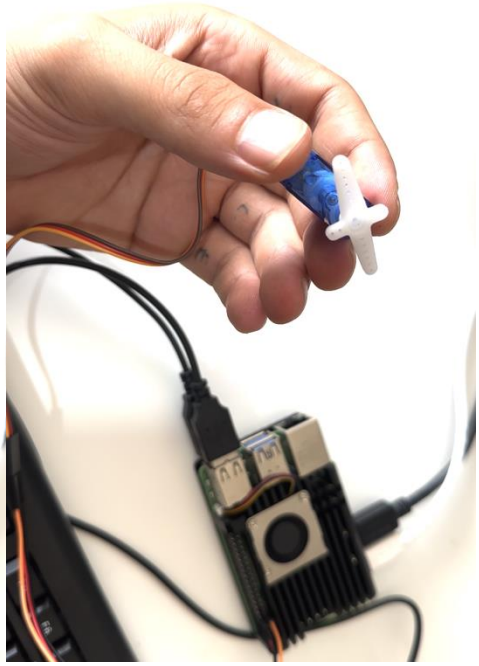


Figura 2 - Servo na posição máxima.

11. Critérios de Avaliação:

- **Conclusão do Projeto:** O servo move-se entre ângulos definidos conforme esperado.
- **Precisão do Circuito:** A cablagem fornece corretamente energia e sinal.
- **Execução de Código:** O script corre sem erro usando `lgpio`.
- **Compreender a PWM:** O participante compreende a relação entre largura do pulso e movimento.

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.

- **Feedback dos participantes:** Reflexão sobre a clareza com que o controlo do movimento foi alcançado.

---

12. Línguas Disponíveis:

Inglês, **Português** (traduções em progresso)

---

13. Colaboradores:

Fátima Pais, Luís Soares – Designers e Facilitadores de Projetos

---

17. Feedback e Melhoria Contínua:

- **Inquéritos**: No final deste marco, alunos e educadores irão preencher inquéritos de feedback para avaliar a eficácia do projeto.
- **Revisões de Projetos**: Após a conclusão do projeto, será realizada uma reunião final de revisão para discutir potenciais melhorias para futuros projetos baseados em IA.
- **Análise de Dados**: Os dados de feedback serão analisados para identificar pontos fortes e áreas a melhorar, informando projetos futuros.

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.

Co-funded by the European Union

Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.