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

Logótipo:

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: 5
10. Plano de Implementação Passo a Passo: 6
11. Critérios de Avaliação: 14
12. Línguas Disponíveis: 14
13. Colaboradores: 14
14. Feedback e Melhoria Contínua: 15

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:

Porta automática para animais de estimação baseada em IA usando visão e deteção de distância

---

2. Categoria do Projeto:

IA em Robótica

---

3. Descrição do Projeto:

Este projeto apresenta o desenvolvimento de um sistema inteligente de portas automáticas concebido especificamente para animais de estimação, utilizando visão computacional e deteção de proximidade. O sistema monitoriza continuamente o ambiente através de uma câmara e aplica um modelo de deteção de objetos para identificar a presença de animais de estimação, nomeadamente gatos e cães. Para evitar ativações acidentais causadas por deteções distantes ou falsos positivos, o reconhecimento visual por si só não é suficiente para ativar o mecanismo.

Uma vez detetado visualmente um animal de estimação, é utilizado um sensor ultrassónico de distância para confirmar que o animal está fisicamente perto da porta. Só quando ambas as condições são cumpridas – deteção correta de animais de estimação e proximidade dentro de um limiar definido – o sistema ativa um servomotor para abrir a porta. A porta permanece aberta durante um período configurável e depois fecha automaticamente. Este projeto demonstra como a perceção por IA e a validação baseada em sensores podem ser combinadas para criar sistemas de automação fiáveis e seguros para aplicações domésticas no mundo real.

---

4. Palavras-chave:

Raspberry Pi, Automação para Animais de Estimação, Portas Inteligentes, Visão por Computador

---

5. Público-alvo:

Tanto Educadores como Alunos

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.

6. Objetivos do Projeto:

- Detetar animais de estimação (gatos e cães) em tempo real usando uma câmara e um modelo de deteção de objetos.
- Combine visão computacional com deteção ultrassónica de distância para validar a proximidade dos animais de estimação.
- Controla um servomotor para abrir e fechar automaticamente um mecanismo de porta.
- Evite a ativação não intencional da porta exigindo múltiplas condições de confirmação.
- Compreenda como a perceção da IA pode ser integrada com atuadores físicos.
- Explore cenários reais de automação para soluções de casas inteligentes e cuidados com animais de estimação.

---

7. Cronologia do Projeto:

Com os recursos fornecidos, este projeto deverá demorar aproximadamente 4-5 horas ou menos a ser realizado de forma calma.

---

8. Recursos Fornecidos:

- Base de código Python contendo deteção de animais de estimação, medição de distância e controlo servo.
- Feedback visual em tempo real que mostra o estado da deteção e o estado da porta
- Vídeo demo do resultado

---

9. Ferramentas e Tecnologias Utilizadas:

- Raspberry Pi 5
- Módulo da Câmara Raspberry Pi 4 v2
- Linguagem de programação Python
- OpenCV para entrada de câmara e processamento de imagem
- Modelo de Deteção de Objetos Pré-treinado (YOLO11-Nano)

Co-funded by the European Union

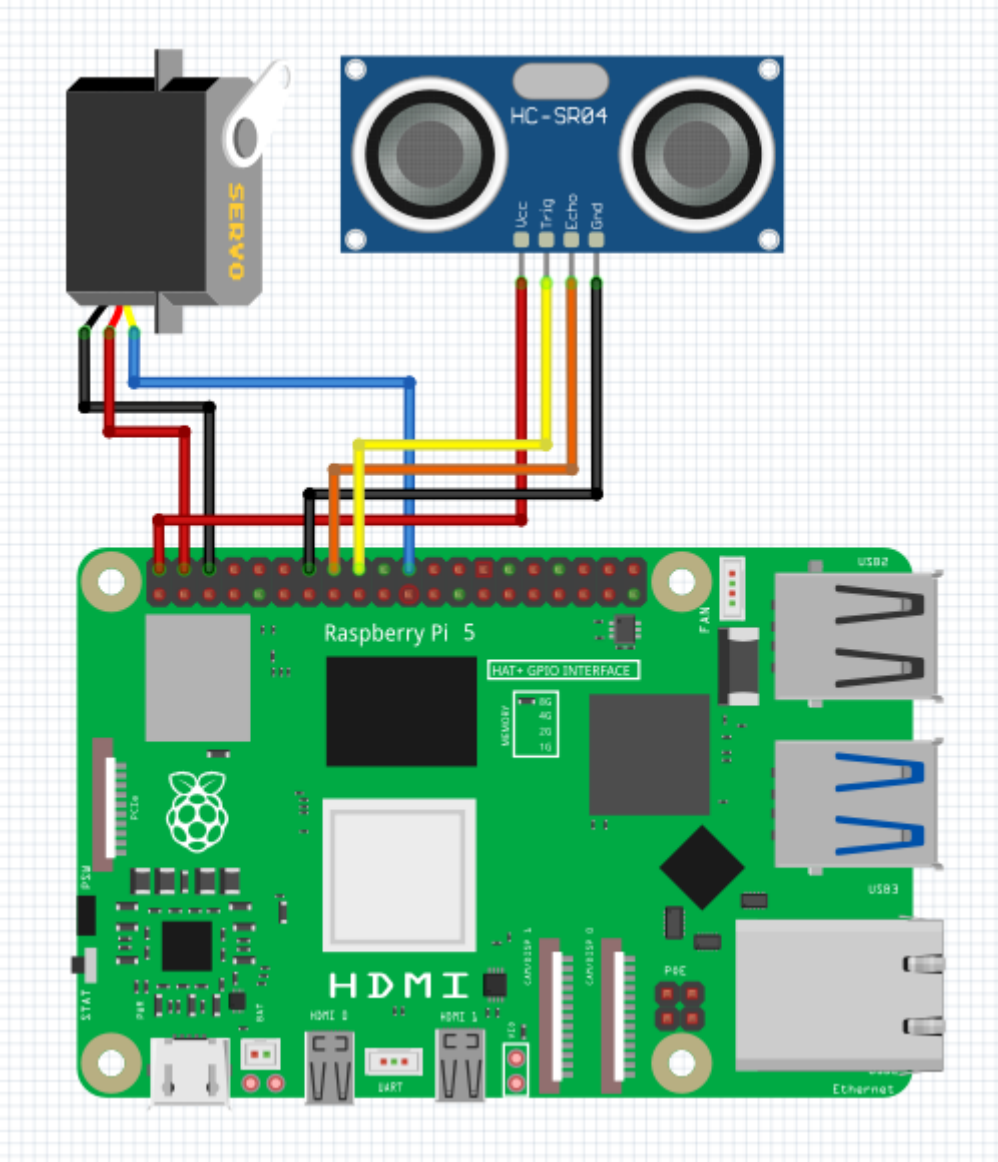
Este projeto foi financiado com o apoio da Comissão Europeia. Esta comunicação reflete apenas as opiniões do autor, e a Comissão não pode ser responsabilizada por qualquer uso que possa ser feito da informação nela contida.

- Sensor de distância ultrassónico
- Servomotor para acionamento de portas
- Bibliotecas GPIO e LGPIO para controlo de hardware

---

10. Plano de Implementação Passo a Passo:

1. **Preparação de Hardware:** Ligue o Raspberry Pi, o módulo da câmara, o servomotor e o sensor ultrassónico de acordo com o diagrama de cablagem.



2. **Configuração da Estrutura:** Crie a estrutura para o resultado (a porta e a parede para ter o sensor/câmara ultrassónico fixo para fins de teste).

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.

3. **Configuração de Software:** Instale o software necessário no Raspberry Pi, incluindo as bibliotecas Raspbian OS e Python, incluindo OpenCV, Picamera2, Ultralytics YOLO e GPIO.

4. **Execute o código:** Copie o código fornecido abaixo e guarde-o no seu Raspberry Pi.

```
Importado CV2

de picamera2 importado Picamera2

da importação de ultralytics YOLO

importar RPi.GPIO como GPIO

Importação da LGPIO

Tempo de importação


# ===== CONFIGURAÇÃO DO PINO =====

# Pinos sensores ultrassónicos

TRIG = 24

ECHO = 23


# Pino servo

SERVO_PIN = 25


# ===== CONFIGURAÇÃO GPIO =====

# Configuração para sensor ultrassónico (usando RPi.GPIO)

GPIO.setmode(GPIO. BCM)

GPIO.setup(TRIG, GPIO. FORA)

GPIO.setup(ECHO, GPIO. DENTRO)


# Configuração para servo (usando lgpio)

h = lgpio.gpiochip_open(0)
```

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.

```
lgpio.gpio_claim_output(h, SERVO_PIN)


# ===== CONFIGURAÇÃO DA CÂMARA =====

picam2 = Picamera2()

picam2.preview_configuration.main.size = (640, 400)

picam2.preview_configuration.main.format = "RGB888"

picam2.preview_configuration.align()

picam2.configure("pré-visualização")

picam2.start()


# ===== MODELO YOLO =====

modelo = YOLO("yolo11n.pt")


# ===== CONFIGURAÇÃO =====

DISTANCE_THRESHOLD = 25# cm

DOOR_OPEN_TIME = 15 # segundos

DOOR_CLOSED_ANGLE = 90

DOOR_OPEN_ANGLE = 0


door_is_open = Falso

door_open_timestamp = Nenhum



def measure_distance():

    """Meça distância usando sensor ultrassónico"""

    Experimente:

        # Garantir que o gatilho está baixo

        GPIO.output(TRIG, Falso)
```

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.

```
        tempo.sono(0.01)


        # Enviar pulso de 10µs para TRIG

        GPIO.output(TRIG, Verdadeiro)

        tempo.sono(0.00001)

        GPIO.output(TRIG, Falso)


        # Mede a duração do pulso no ECHO

        pulse_start = tempo.tempo()

        pulse_end = tempo.tempo()

        timeout = tempo.tempo() + 0,1 # tempo de espera de 100ms


        enquanto GPIO.input(ECHO) == 0 e time.time() < timeout:

            pulse_start = tempo.tempo()


        enquanto GPIO.input(ECHO) == 1 e time.time() < timeout:

            pulse_end = tempo.tempo()


        pulse_duration = pulse_end - pulse_start

        distância = pulse_duration * 34300 / 2 # Velocidade do som 343 m/s


        distância de retorno se a distância < 400 senão 400 # Distância
máxima razoável

    Exceto:

        return 400 # Return max distance em erro



Def set_servo_angle(h, pin, ângulo):
```

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.

```
    """Mover o servo para o ângulo especificado (0-180 graus)"""
    pulse_width = 0,5+ (ângulo / 180,0) * 2,0# milissegundos
    período = 20 # período 20ms = 50Hz

    # Enviar pulsos durante 0,5 segundos
    end_time = tempo.tempo() + 0,5
    while time.time() < end_time:
        lgpio.gpio_write(h, pino, 1)
        tempo.sono(pulse_width / 1000)
        lgpio.gpio_write(h, pin, 0)
        tempo.sono((período - pulse_width) / 1000)



def open_door():
    """Abre a porta rodando servo"""
    door_is_open global, door_open_timestamp
    imprimir ("> PORTA A ABRIR")
    set_servo_angle(h, SERVO_PIN, DOOR_OPEN_ANGLE)
    door_is_open = Verdadeiro
    door_open_timestamp = tempo.tempo()



def close_door():
    """Fecha a porta rodando servo"""
    door_is_open global, door_open_timestamp
    print ("> PORTA A FECHAR")
    set_servo_angle(h, SERVO_PIN, DOOR_CLOSED_ANGLE)
    door_is_open = Falso
```

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.

```
    door_open_timestamp = Nenhum



print("> Sistema de Portas para Animais Iniciado!")

print(f"Limiar de distância: {DISTANCE_THRESHOLD}cm")

print(f"Hora de abertura da porta: {DOOR_OPEN_TIME}s")

print ("Pressione 'q' para sair\n")


# Garantir que a porta comece fechada

close_door()


Experimente:

    embora Verdadeiro:

        # Fotograma de captura

        frame = picam2.capture_array()


        # Faz inferência YOLO

        resultados = modelo(quadro)


        annotated_frame = frame.copy()

        pet_detected = Falso


        # Procura gatos/cães

        para resultados em resultados:

            Para a caixa em Result.Boxes:

                cls_id = int(box.cls[0])

                etiqueta = resultados.nomes[cls_id]
```

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.

```
                if label.lower() == "cat" ou label.lower() == "dog":
                    pet_detected = Verdadeiro
                    confiança = box.conf[0] * 100
                    x1, y1, x2, y2 = map(int, box.xyxy[0])

                    # Desenhar a caixa delimitadora
                    cv2.retângulo(annotated_frame, (x1, y1), (x2, y2), (0,
255, 0), 2)
                    label_text = f"{label} {confiança:.2f}%"
                    cv2.putText(
                        annotated_frame,
                        label_text,
                        (x1, y1 - 10),
                        CV2. FONT_HERSHEY_SIMPLEX,
                        0.5,
                        (0, 255, 0),
                        2
                    )

        # Se o animal for detetado, verifica a distância
        Se pet_detected e não door_is_open:
            distância = measure_distance()
            Imprimir(f"> Animal de estimação detetado! Distância:
{distância:.2f}cm")

            se distância <= DISTANCE_THRESHOLD:
                open_door()

        # Verifica se a porta deve fechar
```

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.

```
        Se door_is_open e door_open_timestamp:
            decorrido = tempo.tempo() - door_open_timestamp
            Se decorrido >= DOOR_OPEN_TIME:
                close_door()
            Caso mais:
                # Contagem decrescente do programa no frame
                restante = int(DOOR_OPEN_TIME - decorrido)
                cv2.putText(
                    annotated_frame,
                    f"Porta fecha-se: {restantes}s",
                    (10, 30),
                    CV2. FONT_HERSHEY_SIMPLEX,
                    0.7,
                    (0, 0, 255),
                    2
                )

        # Mostrar o estado no enquadramento
        estado = "PORTA ABERTA" se door_is_open "PORTA FECHA"
        cor = (0, 255, 0) se door_is_open outros (0, 0, 255)
        cv2.putText(
            annotated_frame,
            Estado,
            (10, 60),
            CV2. FONT_HERSHEY_SIMPLEX,
            0.7,
            Cor
            2
```

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.

```
        )


        # Quadro de exibição
        cv2.imshow ("Sistema de Portas para Animais de Estimação",
annotated_frame)


        # Para com 'q'
        if cv2.waitKey(1) == ord('q'):
            Pausa


exceto o KeyboardInterrupt:
    print("\n> Interrompido pelo utilizador")


Finalmente:
    print("> A limpar...")
    close_door()
    cv2.destroyAllWindows()
    picam2.stop()
    GPIO.cleanup()
    lgpio.gpiochip_close(h)
```

---

11. Critérios de Avaliação:

- **Funcionalidade do Sistema:** Capacidade de replicar o projeto.
- **Criatividade:** Sugestões de melhorias (novas funcionalidades, etc.). Se for sugerido e tentado (com sucesso ou fracasso), deve ser mais recompensado.
- **Otimizações:** Otimizações voluntárias do código com sucesso ou fracasso, especialmente na precisão da deteção de animais de estimação.

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.

12. Línguas Disponíveis:

Inglês

13. Colaboradores:

- **Fátima Pais, Luís Soares** – Designers e Facilitadores de Projetos

14. Feedback e Melhoria Contínua:

Descreva como o feedback será recolhido dos participantes e como esse feedback será usado para melhorar futuros projetos ou iterações deste projeto.

- **Inquérito Pós-Implementação:** Após a implementação do projeto, o feedback será recolhido através de inquéritos direcionados aos participantes envolvidos no processo de testes e observação.
- **Sessões de Reflexão da Equipa:** A equipa do projeto organizará sessões internas de revisão para discutir os desafios enfrentados, o que funcionou bem e as áreas que precisam de aperfeiçoamento.
- **Avaliação Técnica:** Observações relativas à precisão da deteção, estabilidade do sistema e durabilidade do hardware serão documentadas para melhorar versões futuras.
- **Atualizações Iterativas:** Com base no feedback e desempenho, serão implementadas atualizações ao código do software e ao design do sistema para melhorar a fiabilidade e a eficácia.

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.