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: 5
11. Critérios de Avaliação: 13
12. Línguas Disponíveis: 13
13. Colaboradores: 14
14. Feedback e Melhoria Contínua: 14

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:

Visão Computacional Baseada em IA para Monitorização de Segurança

---

2. Categoria do Projeto:

IA em Robótica

---

3. Descrição do Projeto:

Este projeto foca-se no desenvolvimento de um sistema simples mas eficaz de monitorização de segurança baseado em IA, utilizando um Raspberry Pi e um módulo de câmara. O sistema analisa continuamente a transmissão da câmara utilizando um modelo de deteção de objetos para identificar a presença de pessoas na cena. Ao contrário dos sistemas básicos de deteção de movimento, esta solução introduz o conceito de uma zona de deteção definida, representada por uma caixa delimitadora virtual correspondente a uma área de interesse real específica.

Uma notificação é acionada apenas quando uma pessoa é detetada dentro desta zona predefinida, reduzindo falsos alertas causados por movimentos irrelevantes fora da área monitorizada. Quando tal evento ocorre, o sistema envia uma mensagem do WhatsApp através da API CallMeBot, simulando um mecanismo de alerta de segurança em tempo real. Este projeto demonstra como as APIs de visão computacional e mensagens podem ser combinadas para criar soluções inteligentes e de monitorização de baixo custo, adequadas para controlo de acessos, áreas restritas ou aplicações de vigilância.

---

4. Palavras-chave:

Raspberry Pi, Deteção de Pessoas, Visão por Computador, Monitorização de Segurança

---

5. Público-alvo:

Tanto Educadores como Alunos

---

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.

- Implemente a deteção de pessoas em tempo real usando um modelo pré-treinado de deteção de objetos.
- Defina e visualize uma zona de deteção virtual correspondente a uma área monitorizada do mundo real.
- Alertas de gatilho apenas quando uma pessoa entra na zona especificada, minimizando falsos positivos.
- Integre as notificações do WhatsApp usando a API CallMeBot.
- Compreenda como as restrições espaciais podem melhorar a fiabilidade dos sistemas de deteção baseados em IA.
- Explore aplicações práticas orientadas para a segurança da visão computacional em dispositivos de borda.

---

7. Cronologia do Projeto:

Com os recursos disponibilizados, este projeto deverá demorar aproximadamente 2-3 horas ou menos a ser realizado de forma calma.

---

8. Recursos Fornecidos:

- Código Python a implementar deteção de pessoas, lógica de interseção de zonas e notificações do WhatsApp
- Vídeo de demonstração a mostrar o 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)
- API CallMeBot para notificações do WhatsApp via pedido HTTP

---

10. Plano de Implementação Passo a Passo:

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. **Preparação de Hardware:** Ligar o Raspberry Pi ao módulo da câmara

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

3. **Configuração da API do CallMeBot:** Siga o guia de configuração do CallMeBot (https://www.callmebot.com/blog/free-api-whatsapp-messages/) e guarde a sua chave API.

4. **Execute o código:** Copie o código fornecido abaixo e guarde-o no seu Raspberry Pi. Antes de executar o código, substitua o valor de "APIKEY" na sua chave API do passo 3 e substitua o valor de "PHONE_NUMBER" pelo número de telefone registado no passo 3 (este valor deve conter o seu código de país).

```
Importado CV2

Tempo de importação

pedidos de importação

de picamera2 importado Picamera2

da importação de ultralytics YOLO



# ===== CONFIGURAÇÃO =====

APIKEY = "1111111" # Substituir pela sua chave API real

PHONE_NUMBER = "+1123456789" # O seu número de telefone real com código de país (exemplo para Portugal: "+351911234567")

# Configuração da zona de deteção (retângulo em pixels)

# Formato: (x1, y1, x2, y2) - canto superior esquerdo para canto inferior direito

DETECTION_ZONE = (300, 100, 550, 300) # Ajusta com base na visão da câmara



# Definições de tempo de recarga

DETECTION_COOLDOWN = 30 # segundos entre notificações
```

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.

```
# ===== 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()


print("> Câmara inicializada")


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

modelo = YOLO("yolo11n.pt")

print("> modelo YOLO carregado")


# ===== A SEGUIR VARIÁVEIS =====

last_notification_time = 0


# ===== FUNÇÕES =====

def send_notification(object_name):

    """Enviar notificação do WhatsApp via CallMeBot"""

    Experimente:

        url                                                                    =
f"https://api.callmebot.com/whatsapp.php?phone={PHONE_NUMBER}&text=A+Person
+was+detected+in+the+zone.&apikey={APIKEY}"

        requests.get(url, timeout=5)

        print(f"✓ Notificação enviada: {object_name}")

    exceto exceção como e:

        print(f"✗ Falhou no envio da notificação: {e}")
```

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.

```
Def check_intersection(box_coords, zone_coords):
    """
    Verifique se a caixa delimitadora intersecta com a zona de deteção
    box_coords: (x1, y1, x2, y2) - caixa delimitadora de objeto detetada
    zone_coords: (x1, y1, x2, y2) - zona de deteção
    """
    bx1, by1, bx2, by2 = box_coords
    zx1, zy1, zx2, zy2 = zone_coords

    # Verifica se as caixas se sobrepõem
    Se bx2 < zx1 ou bx1 > zx2: # Sem sobreposição horizontal
        return False
    se by2 < zy1 ou by1 > zy2: # Sem sobreposição vertical
        return False

    regresso Verdade



Def draw_detection_zone(frame, zone_coords):
    """Desenha a zona de deteção no frame"""
    x1, y1, x2, y2 = zone_coords
    cv2.retângulo(frame, (x1, y1), (x2, y2), (255, 0, 0), 2) # Retângulo azul
    cv2.putText(
        moldura,
        "ZONA DE DETEÇÃO",
        (x1, y1 - 10),
        CV2. FONT_HERSHEY_SIMPLEX,
```

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.

```
        0.5,
        (255, 0, 0),
        2
    )



# ===== PROGRAMA PRINCIPAL =====
print("Sistema de Deteção de Pessoas > Iniciado!")
print(f"> Zona de deteção: {DETECTION_ZONE}")
print(f"> Tempo de recarga da notificação: {DETECTION_COOLDOWN}s")
print("> Pressione 'q' para sair\n")

Experimente:
    embora Verdadeiro:
        # Captar o fotograma da câmara
        frame = picam2.capture_array()

        # Faz inferência YOLO
        resultados = modelo(quadro)

        # Criar moldura anotada
        annotated_frame = frame.copy()
        draw_detection_zone(annotated_frame, DETECTION_ZONE)

        person_in_zone = Falso

        # Deteções de processos
        para resultados em resultados:
```

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.

```
            Para a caixa em Result.Boxes:

                cls_id = int(box.cls[0])

                etiqueta = resultados.nomes[cls_id]


                # Verificar se o objeto detetado é uma pessoa (classe 0 no
conjunto de dados COCO)

                if label.lower() == "pessoa":

                    confiança = box.conf[0] * 100

                    x1, y1, x2, y2 = map(int, box.xyxy[0])


                    # Verifica se a pessoa está na zona de deteção

                    se check_intersection((x1, y1, x2, y2), DETECTION_ZONE):

                        person_in_zone = Verdadeiro


                        # Desenhar a caixa delimitadora a vermelho (pessoa
na zona)

                        cv2.retângulo(annotated_frame, (x1, y1), (x2, y2),
(0, 0, 255), 2)

                        label_text = f"PESSOA NA ZONA {confiança:.1f}%"

                        cv2.putText(

                            annotated_frame,

                            label_text,

                            (x1, y1 - 10),

                            CV2. FONT_HERSHEY_SIMPLEX,

                            0.5,

                            (0, 0, 255),

                            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.

```
                        # Verifica o tempo de recarga antes de enviar a
notificação

                        current_time = tempo.tempo()

                        time_since_last       =       current_time       -
last_notification_time



                        se time_since_last >= DETECTION_COOLDOWN:

                            print(f"> Pessoa detetada na zona! (Confiança:
{confiança:.1f}%)")

                            send_notification ("Pessoa")

                            last_notification_time = current_time

                        Caso mais:

                            restante     =     int(DETECTION_COOLDOWN     -
time_since_last)

                            print(f"> Pessoa na zona (tempo de recarga:
{restantes}s restantes)")



                    Caso mais:

                        # Desenhar a caixa delimitadora a verde (pessoa fora
da zona)

                        cv2.retângulo(annotated_frame, (x1, y1), (x2, y2),
(0, 255, 0), 2)

                        label_text = f"Pessoa {confiança:.1f}%"

                        cv2.putText(

                            annotated_frame,

                            label_text,

                            (x1, y1 - 10),

                            CV2. FONT_HERSHEY_SIMPLEX,

                            0.5,

                            (0, 255, 0),

                            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.

```
                        )


        # Estado do ecrã
        estado = "PESSOA NA ZONA" se person_in_zone caso contrário "Monitorização..."
        cor = (0, 0, 255) se person_in_zone mais (0, 255, 0)
        cv2.putText(
            annotated_frame,
            Estado,
            (10, 30),
            CV2. FONT_HERSHEY_SIMPLEX,
            0.7,
            Cor
            2
        )

        # Mostrar temporizador de recarga se ativo
        se last_notification_time > 0:
            time_since_last = tempo.tempo() - last_notification_time
            Se time_since_last < DETECTION_COOLDOWN:
                restante = int(DETECTION_COOLDOWN - time_since_last)
                cv2.putText(
                    annotated_frame,
                    f"Tempo de recarga: {restantes}s",
                    (10, 60),
                    CV2. FONT_HERSHEY_SIMPLEX,
                    0.6,
                    (255, 255, 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.

```
                    2

                )


        # Quadro de exibição

        cv2.imshow ("Sistema de Deteção de Pessoas", annotated_frame)


        # Para com 'q'

        if cv2.waitKey(1) == ord('q'):

            Pausa


exceto o KeyboardInterrupt:

    print("\n> Interrompido pelo utilizador")


Finalmente:

    print("> A limpar...")

    cv2.destroyAllWindows()

    picam2.stop()

    print ("Programa > terminado")
```

---

11. Critérios de Avaliação:

Explique como será medido o sucesso do projeto. Isto pode incluir feedback qualitativo dos participantes, indicadores quantitativos de desempenho ou resultados específicos do projeto.

- **Funcionalidade do Sistema:** Capacidade de replicar o projeto.
- **Criatividade:** Sugestões de melhorias (novas funcionalidades, novos objetos detetados, etc.). Se for sugerido e tentado (com sucesso ou fracasso), deve ser mais recompensado.

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.

- **Redução de Falsos Alertas:** Uso eficaz da lógica de zonas e temporizadores de recarga para evitar notificações de spam.

---

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.