Co-funded by the European Union

PROJETOS DE IA

Escola técnica Pirot

KA220-VET - Parcerias de cooperação na educação e formação profissional

Título do Projeto: Ferramentas de IA para escolas de VET

Data do Documento: março de 2026

Este material foi compilado e preparado para fins do projeto Erasmus por:
Escola técnica Pirot (autor: Bojan Ćirić, coautores: Boban Blagojević, Aleksandar Madić)
ICEP (autor: Ladislav Mariš, coautor: Adelaida Fanfarova)
Agrupamento de Escolas Tomas Cabreira (autora: Sandra Nobre, coautoras: Rui Dias, Gilherme Mota, Carla Lima)
Traduzido por: Bojana Stojanović

INFORMAÇÃO DE MORADA

Takovska 22, Pirot, Sérvia
Web: https://book.tsp.edu.rs

Índice

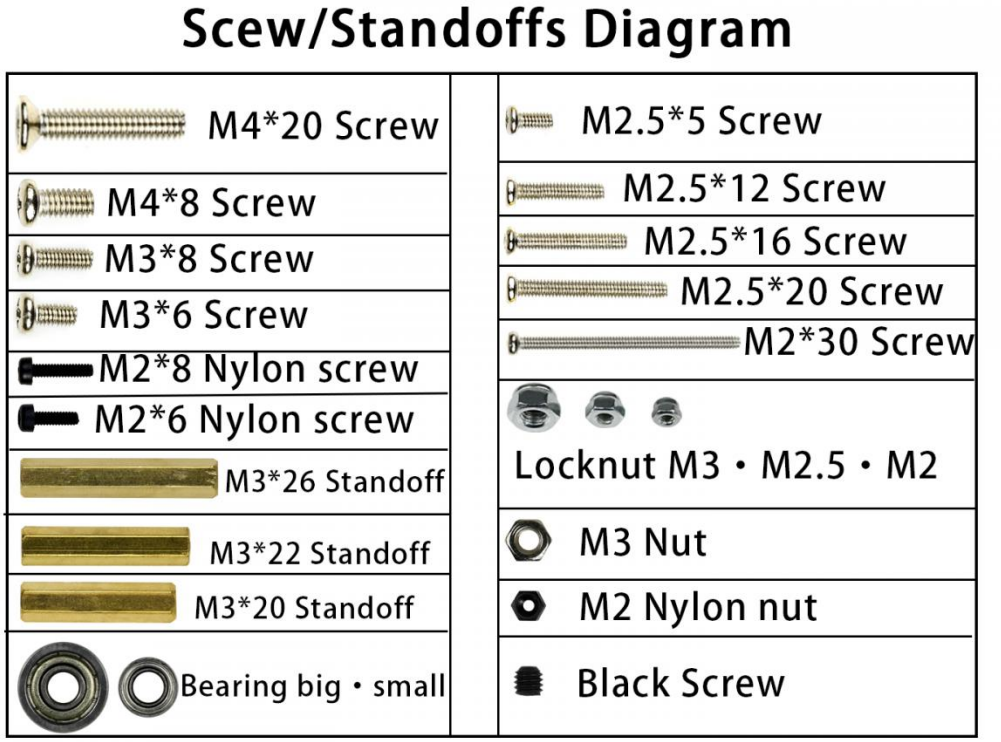
I. PILOTO AUTOMÁTICO DE IA PIRACER

1. PILOTO AUTOMÁTICO DE APRENDIZAGEM PROFUNDA PI-RACER

1.1 Manual de Montagem do PiRacer

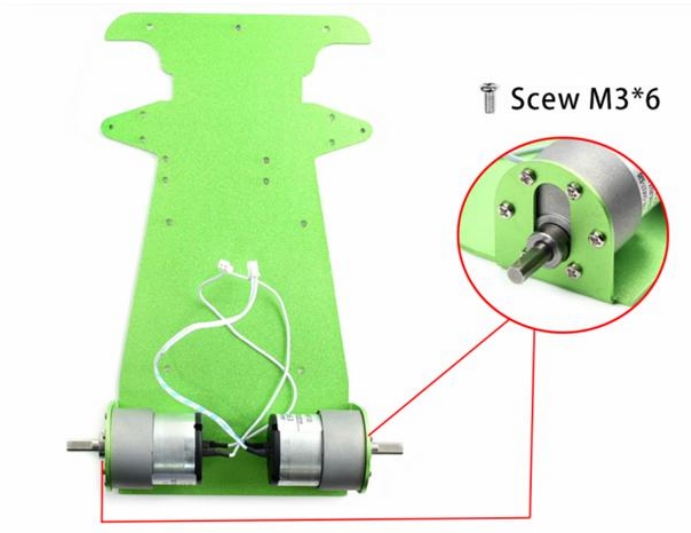
Diagrama de parafusos/separadores

Diagrama para referência. Note que os parafusos que vêm com a roda servo não estão listados aqui.



1. Fixar os motores a um chassis metálico

Nota: Não use o M3*8. É mais comprido e pode danificar o motor.



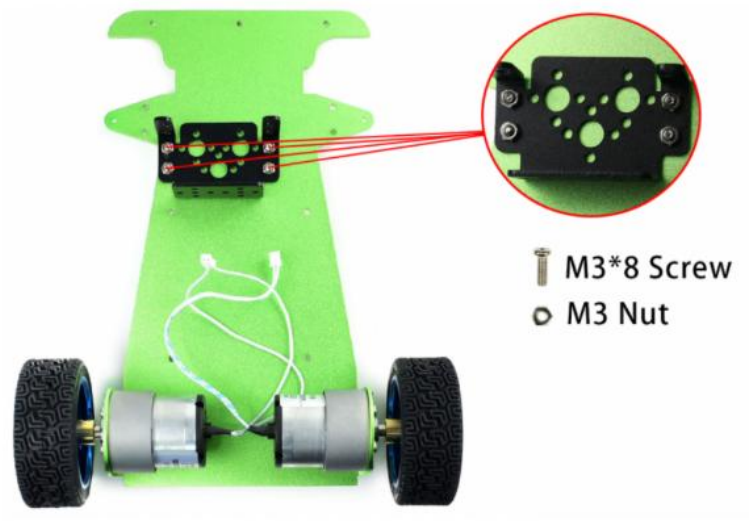
2. Adicionar engates às rodas

Primeiro, insira o parafuso preto no acoplador. Depois adiciona o engate à roda. Pode ser necessário pressionar o engate contra a roda. Fixa o acoplador à roda usando um parafuso M4*8.

3. Montar as rodas

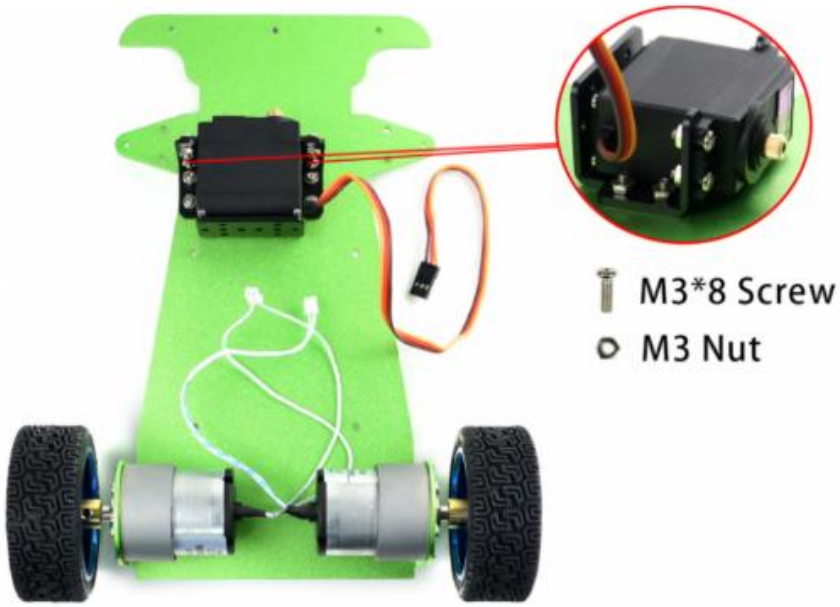
Aperta o parafuso preto para fixar o acoplador ao lado plano do eixo

4. Montar o suporte do servo num chassis metálico



5. Fixar o servo no suporte e usar os parafusos e porcas

Certifica-te de que o servo está bem colocado. O eixo exterior deve estar no centro.

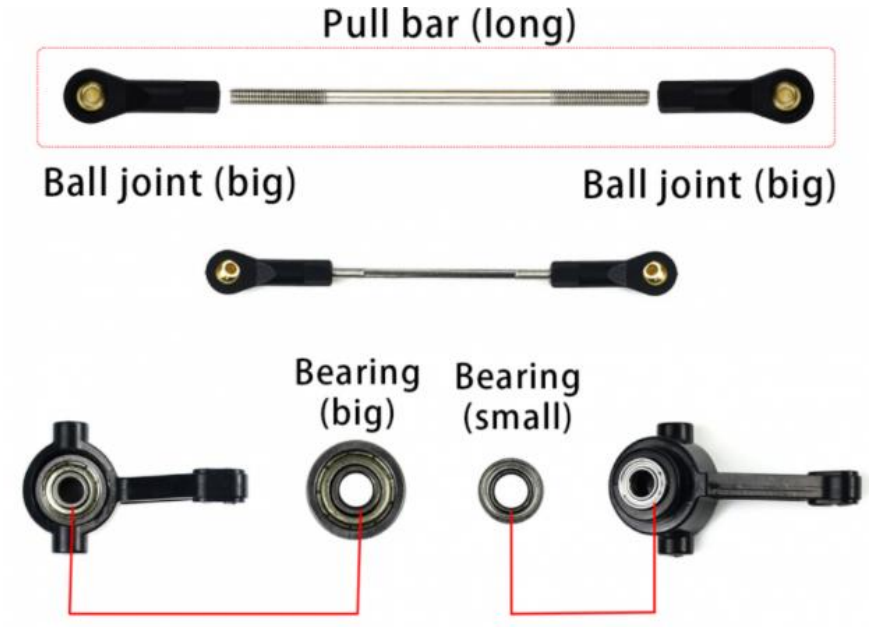


6. Montar a barra de puxar servo

A barra de tração é combinada usando duas juntas esféricas, uma plana e outra esférica, e a barra curta. As duas articulações esféricas devem ser perpendiculares uma à outra. Fixe a roda servo (buzina) ao suporte da roda servo usando os parafusos que vieram com a roda servo. Depois fixa a junta esférica plana ao suporte da roda servo. Note que o sulco da roda servo está voltado para o exterior.

7. Montar a barra de tração da roda dianteira e os nós dos dedos da direção

A barra de tração da roda dianteira é combinada usando duas juntas esféricas e a barra longa. Depois coloca os rolamentos nos nós dos dedos da direção. Cada articulação precisa de um apoio pequeno e outro grande.



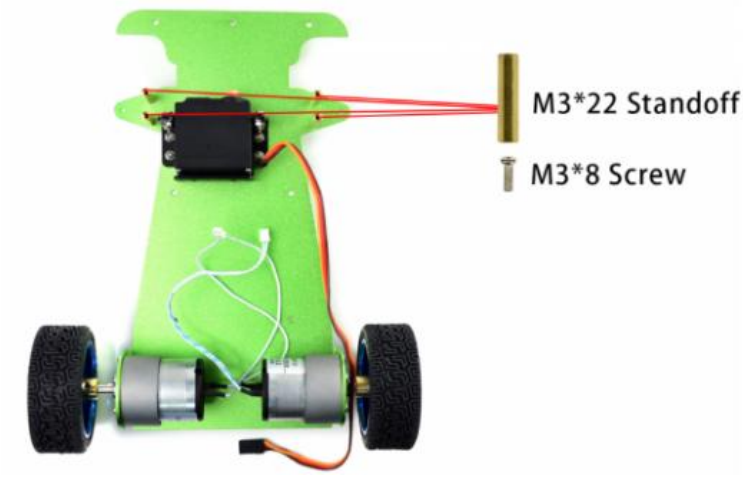
8. Montar o servo pull bard, o pull bard da roda dianteira e os nós dos dedos de direção

Prenda a barra de puxador servo no topo, depois a barra de puxar da roda dianteira e, finalmente, os nós dos dedos. O rolamento maior deve estar voltado para o interior.

9. Fixar as rodas no suporte de direção

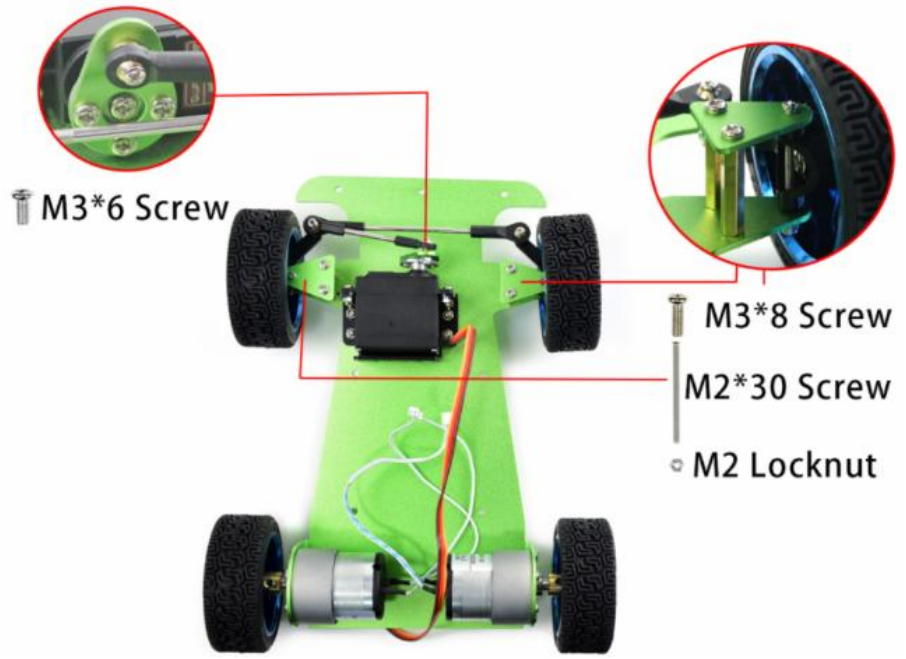
A porca deve estar do lado exterior da roda, em frente à articulação. Certifica-te de que a roda não está nem demasiado apertada nem demasiado solta. Teste a roda para garantir que se move livremente.

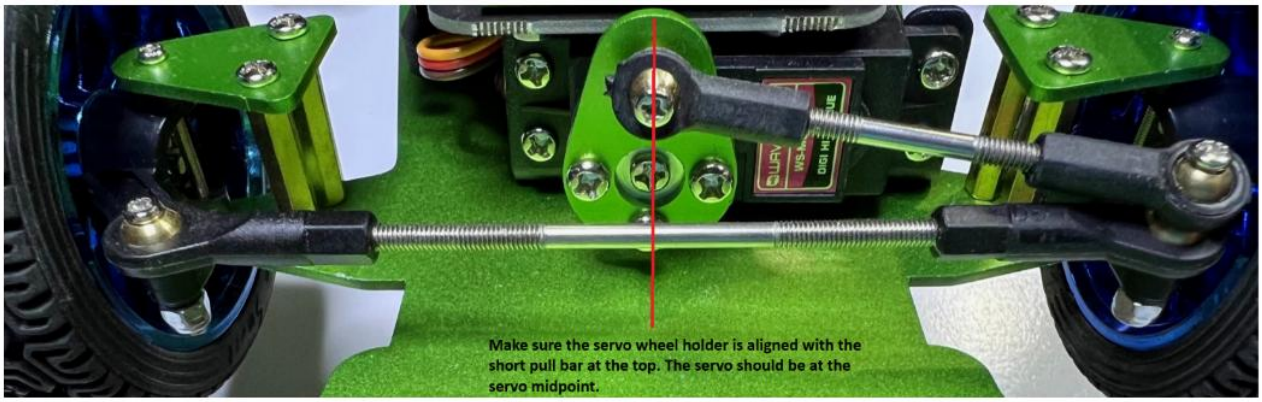
10. Adicionar os entrelaçamentos M3 para as rodas dianteiras



11. Montar a combinação das rodas dianteiras

Coloca a roda servo na servo, fixa-a com o parafuso M3. Fixa as rodas com parafusos M2, porca de bloqueio e a tábua triangular. As rodas dianteiras devem ser diretas. Ajusta a barra de tração longa se necessário.

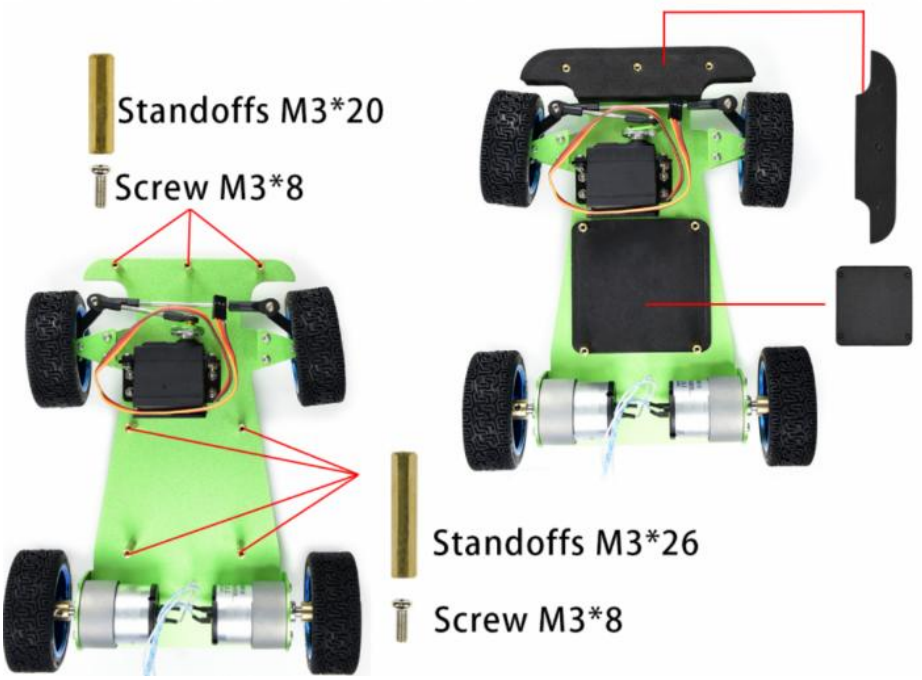




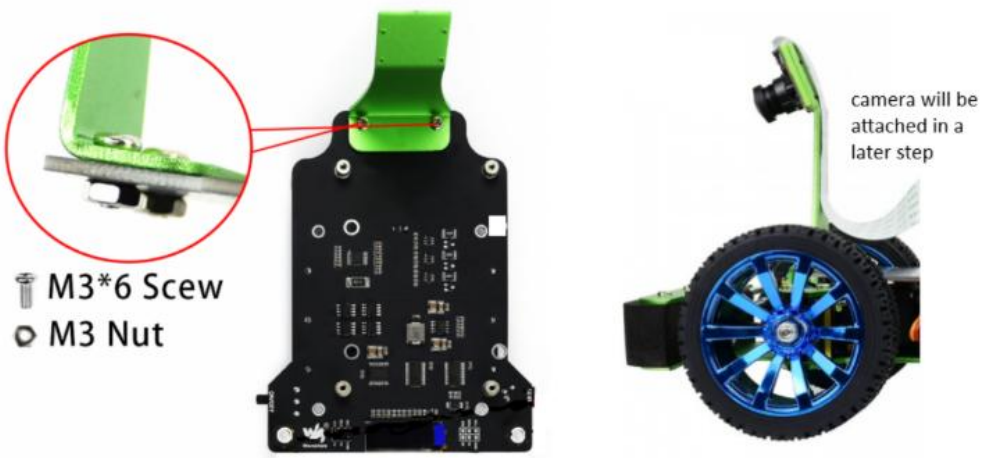
As rodas dianteiras devem ser diretas. Ajusta a barra de tração longa se necessário.

12. Adicionar os suportes para a placa de expansão do PiRacer e o para-choques

Insira as pastilhas de feltro EVA para o para-choques e a placa de expansão PiRacer.

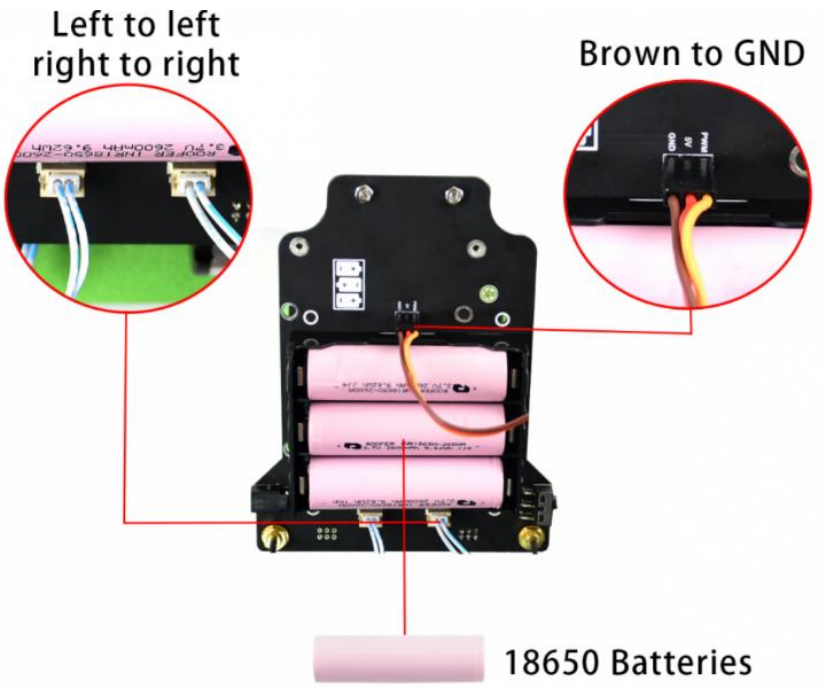


13. Fixar o suporte da câmara na placa de expansão do PiRacer.



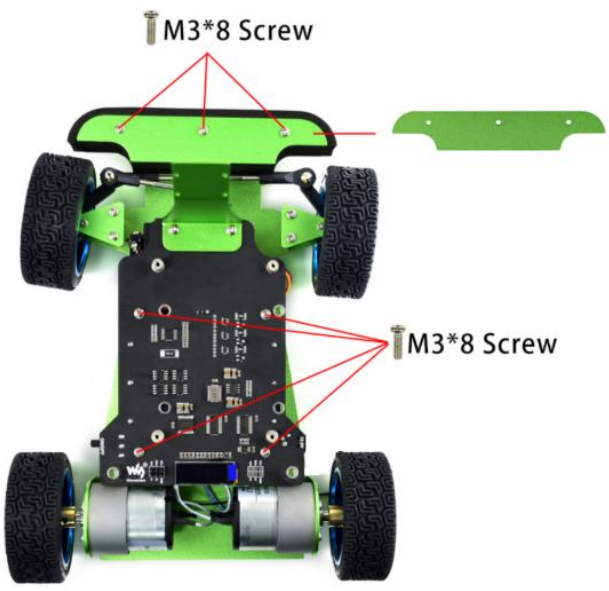
14. Inserir as pilhas na direção correta

Liga os fios dos motores e do servo à placa de expansão PiRacer.



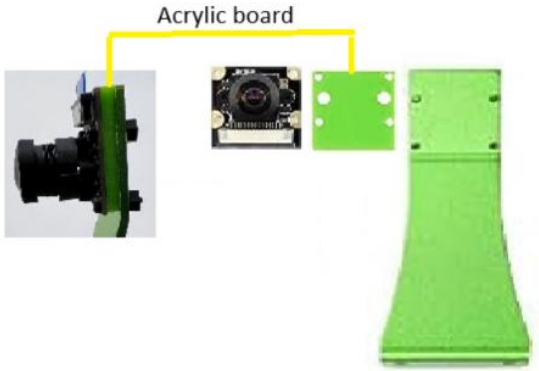
15. Fixa a placa de expansão e o para-choques metálico

Ajusta a posição dos fios do motor e servo e liga a placa de expansão PiRacer ao chassis metálico e fixa o para-choques metálico.

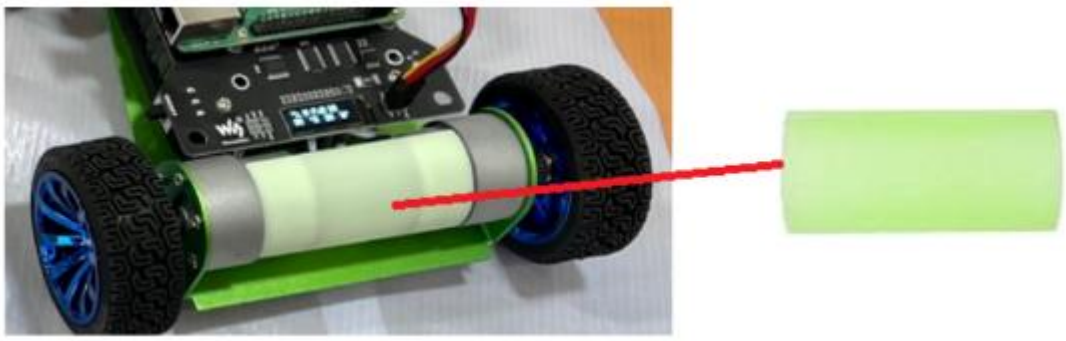


16. Montar a câmara no suporte usando parafusos de nylon

Nota: A placa acrílica deve estar entre a câmara e o suporte metálico para evitar curtos-circuitos.



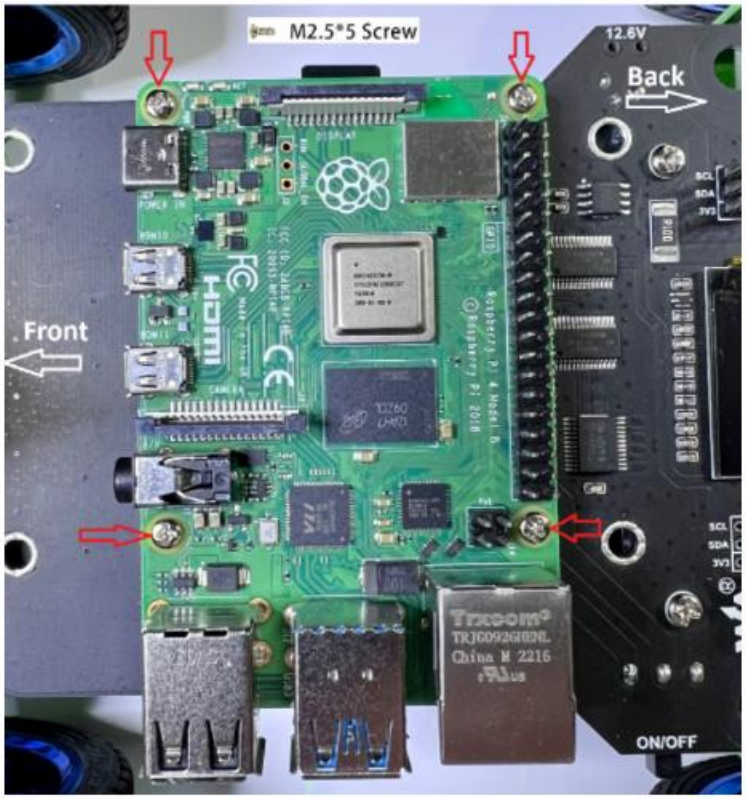
17. Acoplar a caixa do motor impressa em 3D nos motores DC



18. Desliga o Raspberry Pi e desliga o carregador da placa de expansão do PiRacer antes de prosseguir.

19. Ligar o Raspberry Pi à placa de expansão do PiRacer

Os pinos GPIO devem estar na traseira do carro.

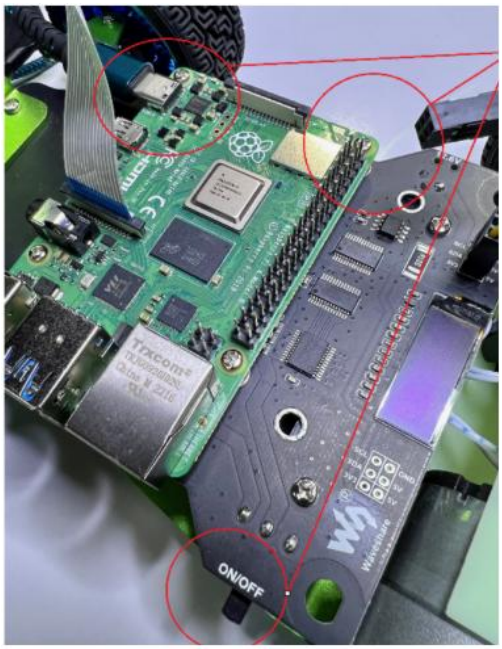


21. Ligue o cabo flat da câmara à entrada da câmara Raspberry Pi

Lado azul em direção às portas USB do Pi.

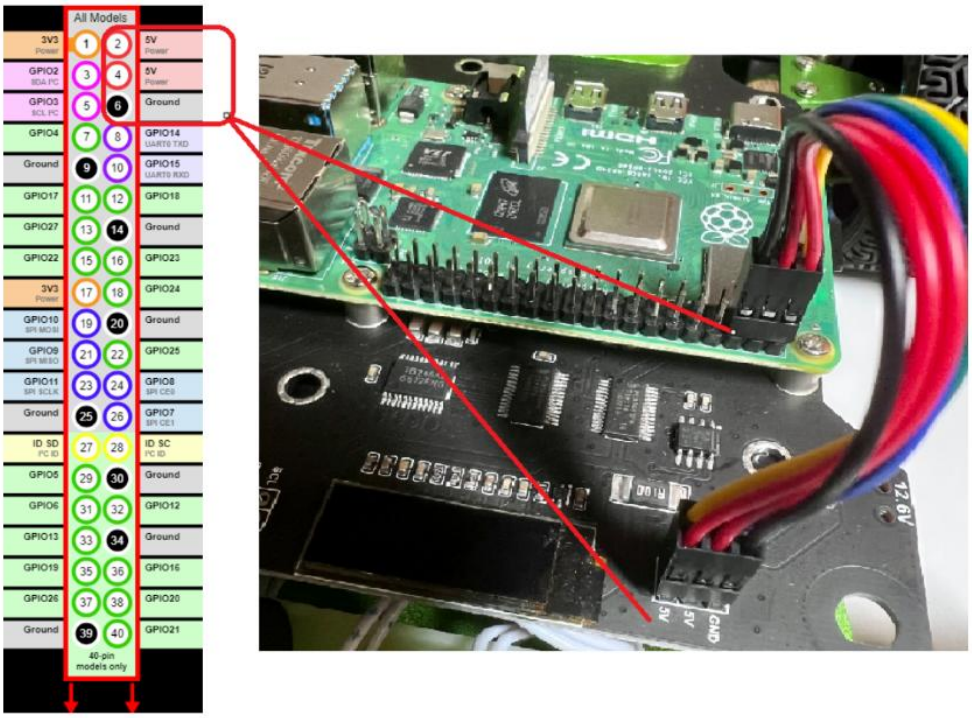
****AVISO** Certifique-se de que o Raspberry Pi não está alimentado ao ligar os fios de 6 pinos**

Também não ligues o Raspberry Pi através do USB-C (alimentação externa) enquanto estiver alimentado pela placa de expansão PiRacer.



22. Ligue o Raspberry Pi à placa de expansão PiRacer usando os fios de 6 pinos

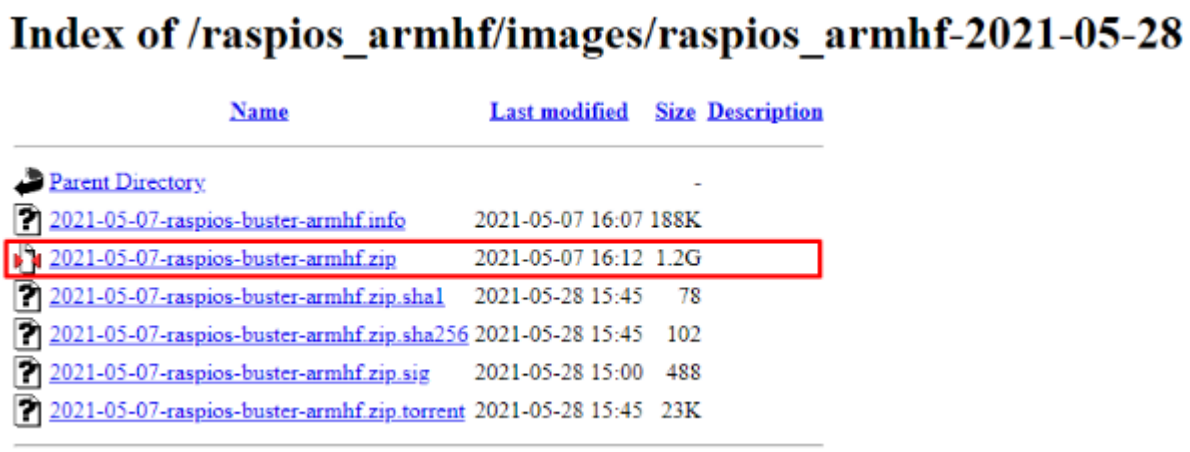
Compare o vermelho|vermelho|preto (5V|5V|terra) no Pi GPIO e o preto|vermelho|vermelho|vermelho (gnd|5V|5V) na placa de expansão do PiRacer.



1.2 Raspian Legacy (Buster) Desktop

Flash OS - Raspian Legacy (Buster) Desktop

Navegue até ao repositório oficial de downloads do Buster OS. **Todas as versões anteriores do Raspberry Pi OS podem ser encontradas e descarregadas aqui** , e o link oficial de download do Raspberry Pi 'Buster' OS diretamente anterior **está aqui**. Veja abaixo como isso se apresenta, descarregue o ficheiro de imagem clicando no ficheiro destacado.

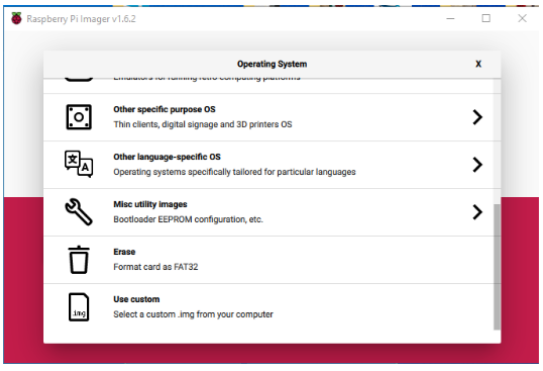


Index of /raspios_armhf/images/raspios_armhf-2021-05-28

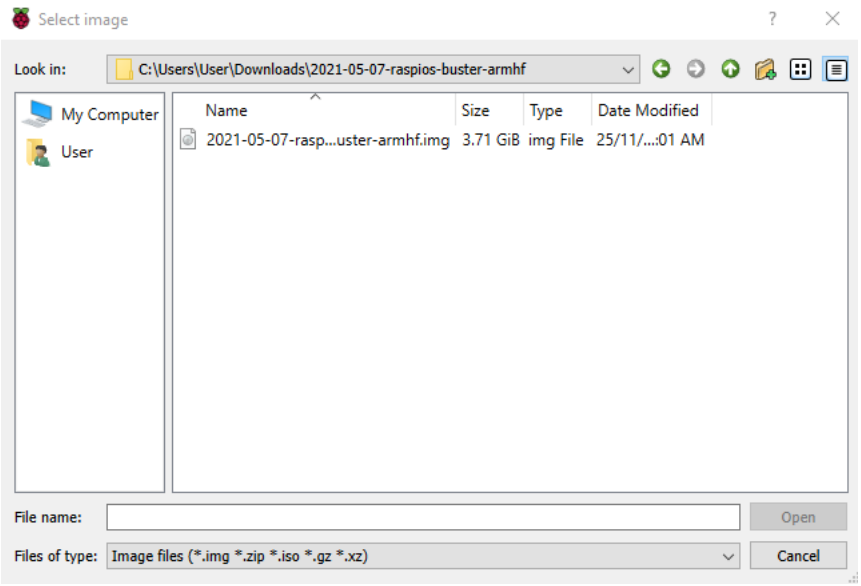
Name	Last modified	Size	Description
Parent Directory		-	
2021-05-07-raspios-buster-armhf.info	2021-05-07 16:07	188K	
2021-05-07-raspios-buster-armhf.zip	2021-05-07 16:12	1.2G	
2021-05-07-raspios-buster-armhf.zip.sha1	2021-05-28 15:45	78	
2021-05-07-raspios-buster-armhf.zip.sha256	2021-05-28 15:45	102	
2021-05-07-raspios-buster-armhf.zip.sig	2021-05-28 15:00	488	
2021-05-07-raspios-buster-armhf.zip.torrent	2021-05-28 15:45	23K	

Descarregue e instale o Raspberry Pi Imager a partir de https://www.raspberrypi.com/software/

Agora, vamos abrir o Oficial Raspberry Pi Imager, veja-o na imagem abaixo. Vale a pena referir aqui - se acertares **| CTRL+SHIFT+X |** no seu teclado, enquanto estiver no Programa Raspberry Pi Imager, abrirá o Menu Oculto Avançado. Este Menu Oculto permite-lhe pré-configurar o seu Raspberry Pi com SSH, credenciais WIFI e definições de localização.



Depois disso, abre um explorador de ficheiros. Navega até ao Ficheiro de Imagem do Disco extraído e clica nele. Veja isto na imagem abaixo. Depois abre a imagem do ficheiro e isso vai armar o Raspberry Pi Imager com o sistema operativo 'Buster' do Raspberry Pi. Encontra-o, seleciona-o e abre-o.



Insira o cartão Micro-SD que pretende instalar no seu computador. Use um **adaptador USB para Micro-SD** se necessário. Depois clica no **| ESCOLHER ARMAZENAMENTO |** e seleciona a Micro-SD inserida. Tenha em mente que qualquer dado que estivesse no seu cartão Micro-SD será apagado ou apagado permanentemente quando for gravado. O Official Raspberry Pi Imager vai agora parecer a imagem abaixo.

Com tudo resolvido (o sistema operativo correto carregado e o Armazenamento correto selecionado), pode agora clicar no **| WRITE |** botão para iniciar o processo de flash. Veja este processo de flashing na imagem abaixo.

Quando a flash estiver concluída, ejetará virtualmente o cartão Micro-SD do computador. Assim, podes simplesmente tirar fisicamente a tua Micro-SD e inseri-la num computador Raspberry Pi de placa única. Depois configura o teu Raspberry Pi normalmente como um computador de secretária. Assim que arrancar, será recebido pelo antigo fundo familiar, veja a imagem abaixo, e terá conseguido flashar o sistema operativo 'Buster' para o seu Raspberry Pi. Agora está livre para explorar o seu terreno digital bem conhecido.

Siga as instruções do Assistente Pi para definir a localização, teclado, wifi e obter atualizações

Menu / Preferências / Configuração do Raspberry Pi / Interfaces

- Ativar Câmara e I2C
- opcional: ativar VNC para acesso remoto
- Clica em OK e reinicia

Software Adicional

Java 3.8.3 (executar um de cada vez)

```
sudo apt install build-essential libncurses5-dev libgdbm-dev libnss3-dev libssl-dev
libreadline-dev libffi-dev  -y
WGET https://www.Python.org/ftp/python/3.8.3/Python-3.8.3.tgz
tar -zxvf Python-3.8.3.tgz
cd Python-3.8.3
sudo ./configure --enable-optimizations
sudo make -j 4
Sudo Make AltInstall
Sudo Python3.8 -M PIP Install BOTO3
Sudo Python3.8 -M Instalação de PIP tqdm
```

Ferramentas adicionais

```
sudo apt install chromium-browser  -y
sudo apt-get install zip descompactar -y
```

AWS CLI

Accou utilizador padrão no Raspberry Pi

```
sudo apt install awscli -y
Instalação do Sudo Pip3 -- Atualizar AWSCLI
Sudo Pip3 Install Boto3

Utilizador - Pi
pw - framboesa
```

1.3 Ramo WaveShare para o Software DonkeyCar

Dependências de Instalação

```
sudo apt-get install build-essential python3 python3-dev python3-pip
python3-virtualenv python3-numpy python3-picamera python3-pandas
 python3-rpi.GPIO i2c-tools avahi-utils joystick libopenjp2-7-dev
libtiff5-dev gfortran libatlas-base-dev libopenblas-dev
libhdf5-serial-dev git ntp -y
```

Opcional?

```
Sudo apt-get install libilmbase-dev libopenexr-dev libgstreamer1.0-dev
libjasper-dev libwebp-dev libatlas-base-dev libavcodec-dev libavformat-dev
libswscale-dev libqtgui4 libqt4-test -y
```

Configurar o Ambiente Virtual

```
python3 -m virtualenv -p python3 env --system-site-packages
Echo "fonte ~/env/bin/ativar" >> ~/.bashrc
fonte ~/.bashrc
```

Instalar Código Python DonkeyCar - Ramo WaveShare para DonkeyCar Software 3.1.0

```
Projetos MKDIR
Projetos de CD
Git clone https://github.com/waveshare/donkeycar

CD Donkeycar
Mestre do Git Checkout
pip install -e .[pi]

pip install tensorflow==1.13.1
```

Não é necessário instalar pip numpy --upgrade

```
 PIP install protobuf==3.20.*
```

Versão do tensorflow de teste - deve mostrar 1.13.1

```
python -c "importar tensorflow; impressão(tensorflow.__version__)"
```

NOTA: Não podia correr modelo no carro até usar os dois seguintes

```
Instalação de pip https://github.com/lhelontra/tensorflow-on-
arm/releases/download/v2.2.0/
 tensorflow-2.2.0-cp37-none-linux_armv7l  .whl

pip install tensorflow==1.13.1
```

Edit camera.py para adicionar self.camera.hflip = Verdadeiro

```
Sudo Nano /Home/Pi/Projects/Donkeycar/Donkeycar/Peças/Câmara.py
```

```
class PiCamera(BaseCamera):
    def __init__(self, image_w=160, image_h=120, image_d=3, framerate=20):
        from picamera.array import PiRGBArray
        from picamera import PiCamera

        resolution = (image_w, image_h)
        # initialize the camera and stream
        self.camera = PiCamera() #PiCamera gets resolution (height, width)
        self.camera.vflip = True
        self.camera.hflip = True
        self.camera.resolution = resolution
        self.camera.framerate = framerate
        self.rawCapture = PiRGBArray(self.camera, size=resolution)
        self.stream = self.camera.capture_continuous(self.rawCapture,
            format="rgb", use_video_port=True)

        # initialize the frame and the variable used to indicate
        # if the thread should be stopped
```

Instalação Opcional do OpenCV

```
sudo apt install python3-opencv  -y
```

Teste com:

Python -C "Import CV2"

1.4 Começar com DonkeyCar

Abra uma janela de terminal, insira o seguinte comando

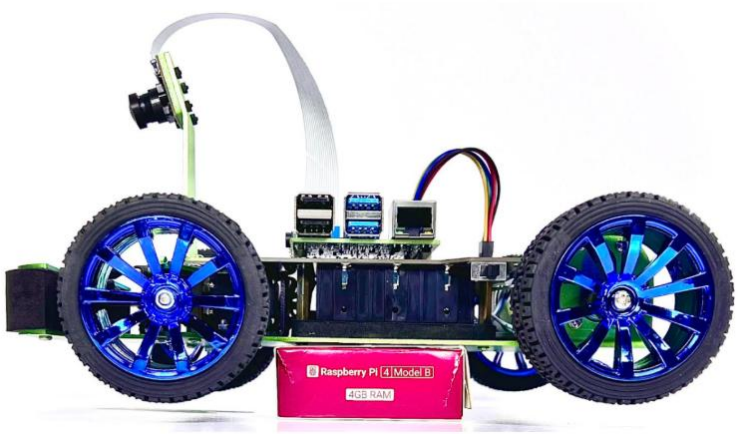
Criar a App DonkeyCar

Isto vai criar uma pasta chamada mycar com todo o código python necessário para conduzir o carro.

Calibrar a direção dianteira

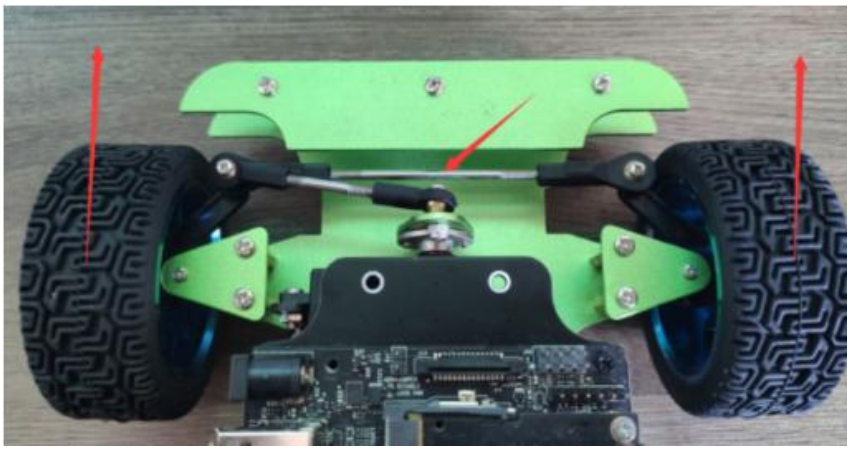
Certifique-se de que o seu carro está fora do chão para evitar situações de descontrolo.

Usa uma caixa pequena como a do Raspberry Pi que veio.

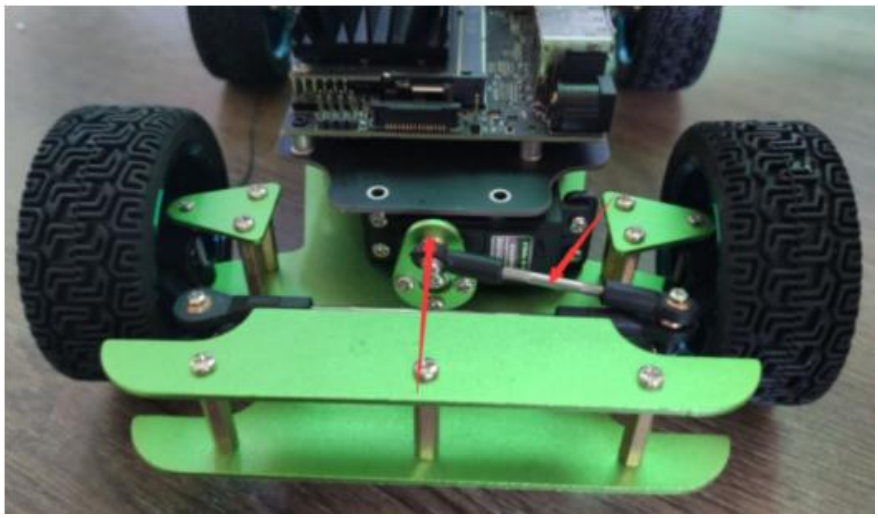


Para garantir que o DonkeyCar consegue conduzir em linha reta e fazer as curvas necessárias na pista, é necessário calibrar o hardware e o software do carro.

As rodas dianteiras devem ser diretas. Ajusta a barra de tração longa se necessário



O suporte da roda servo deve estar alinhado com a barra de tração curta no topo. Ajusta a barra de tração curta se necessário.



Calibrar o software do servo

Encontre o PWM de direção para a esquerda, central e direita para o servo deste carro.

O centro deve estar a meio caminho entre a esquerda e a direita. Exemplo:

Esquerda 200
Direita 560
O centro será 380

Numa janela terminal, introduza o seguinte

```
cd ~/mycar
calibração burro --canal 0 --bus=1
```

Experimenta valores 300, 400, 500 e vê como a direção muda. Calcula o turno máximo para esquerda e direita.

Depois de tiveres os números, edita o config.py e atualiza os valores que encontraste.

O acelerador deve ser ajustado usando os seus números para a direção. O acelerador já está corretamente ajustado.

Nano config.py

```
#STEERING
STEERING_CHANNEL = 0            #channel on the 9685 pwm board 0-15
STEERING_LEFT_PWM = 200         #pwm value for full left steering
STEERING_RIGHT_PWM = 560        #pwm value for full right steering

#THROTTLE
THROTTLE_CHANNEL = 0            #channel on the 9685 pwm board 0-15
THROTTLE_FORWARD_PWM = 4095     #pwm value for max forward throttle
THROTTLE_STOPPED_PWM = 0        #pwm value for no movement
THROTTLE_REVERSE_PWM = -4095    #pwm value for max reverse throttle
```

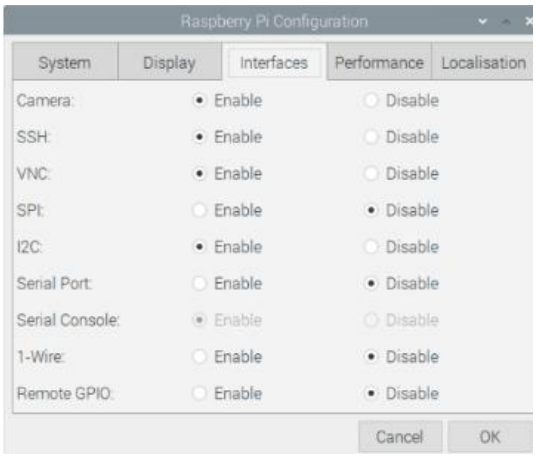
Instalar o Serviço de Ecrã OLED

```
CD ~
Git clone https://github.com/waveshare/pi-display
Ecrã CD Pi
sudo ./install.sh
CD ~
```

2. ACESSO REMOTO AO RASPBERRY PI USANDO REALVNC

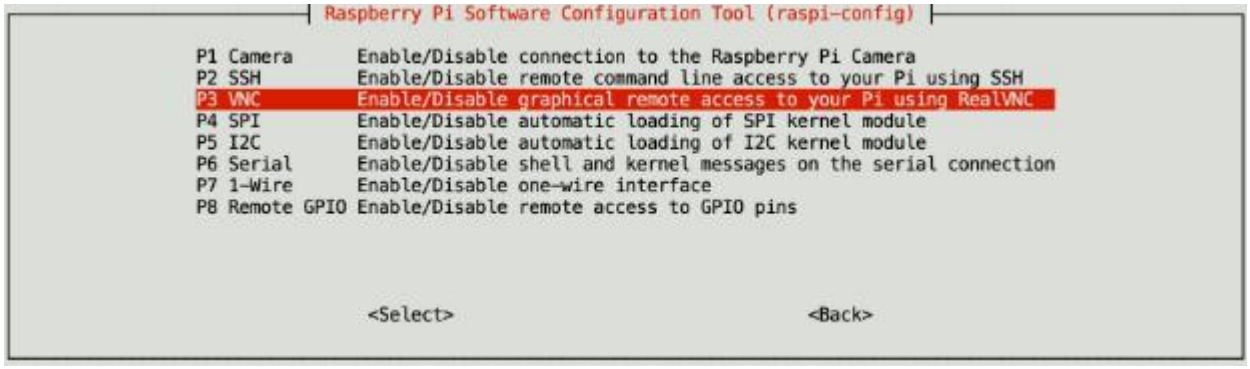
2.1 Ativar o VNC no Raspian OS com a ou b:

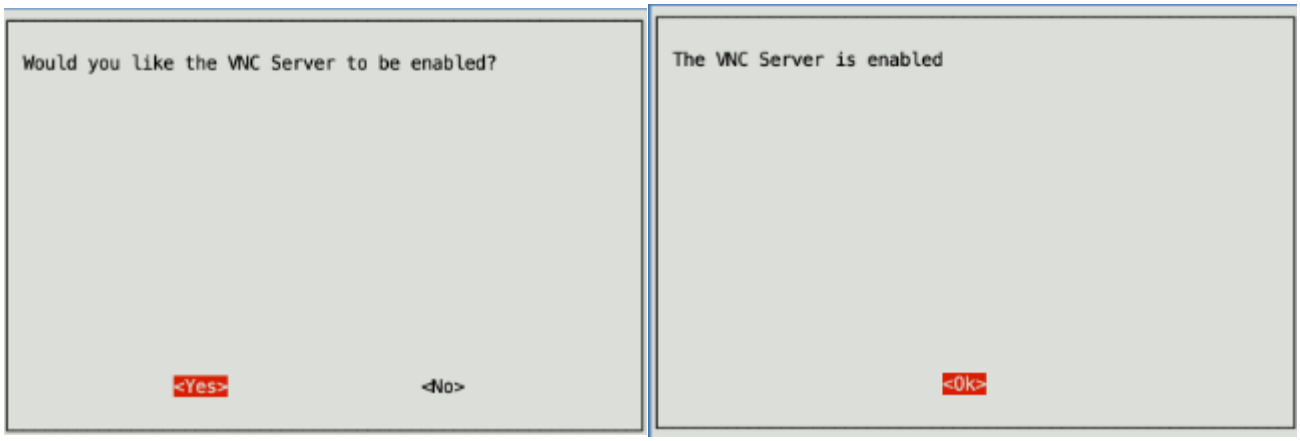
a) Ativar VNC nas Preferências - Configuração do Raspberry PI – Interfaces



b) Outro método para ativar o VNC é usar o Terminal, introduz o comando

```
sudo raspi-config
```

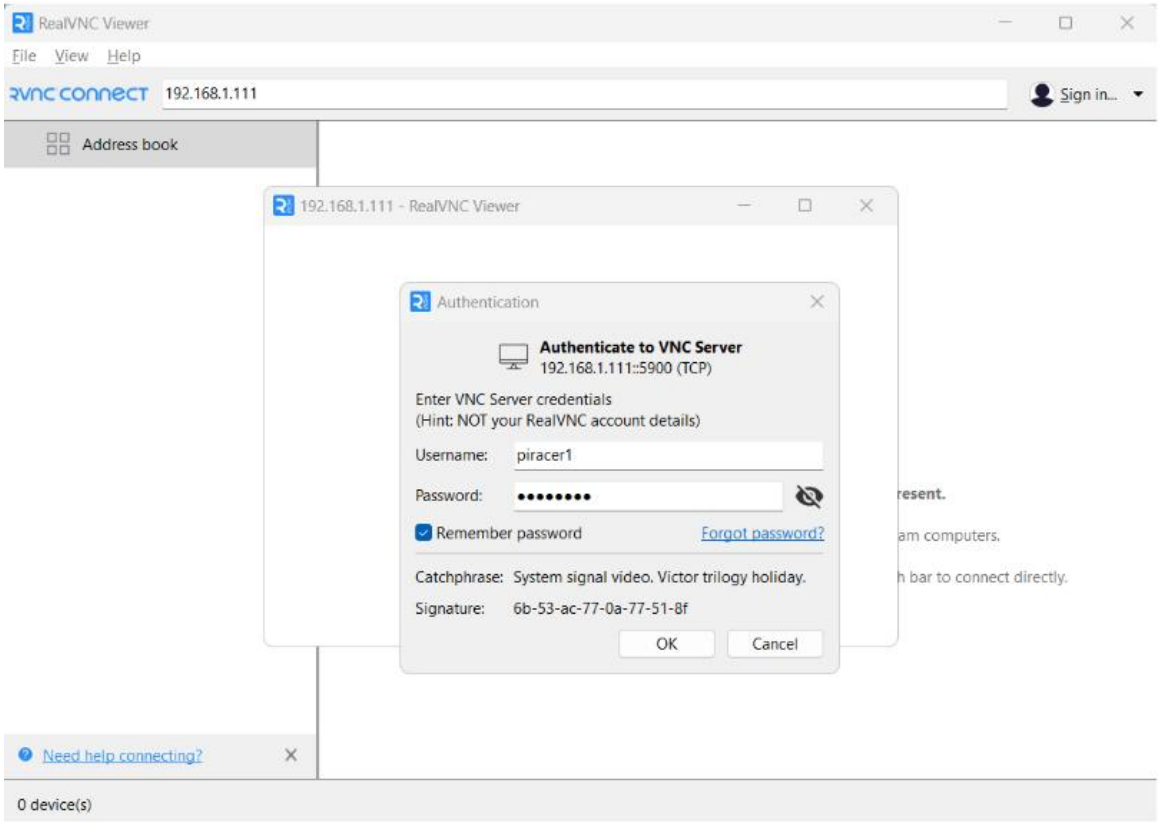




2.2 Instalar um Visualizador VNC

Vai precisar de instalar um Visualizador VNC no seu computador, para poder ligar-se ao seu Raspberry Pi. Existem vários visualizadores disponíveis, mas o mais fácil de configurar é **o Real VNC Viewer**. Pode descarregar instaladores para Windows e Mac aqui: https://www.realvnc.com/en/connect/download/viewer/

Abra o Real VNC Viewer e introduza o endereço IP do Piracer (leia-o no ecrã do carro)



3. COMEÇAR A CONDUZIR E RECOLHER DADOS

Liga o carro

```
cd ~/mycar
Python gerir. Campanha de PY
```

Este script inicia o ciclo de condução no seu carro, que inclui uma parte que serve como servidor web para controlar o seu carro. Agora pode controlar o seu carro a partir de um navegador web na URL: <hostname.local>:8887 do seu carro.

Abra um navegador e ligue-se ao DonkeyCar Monitor em **localhost:8887**

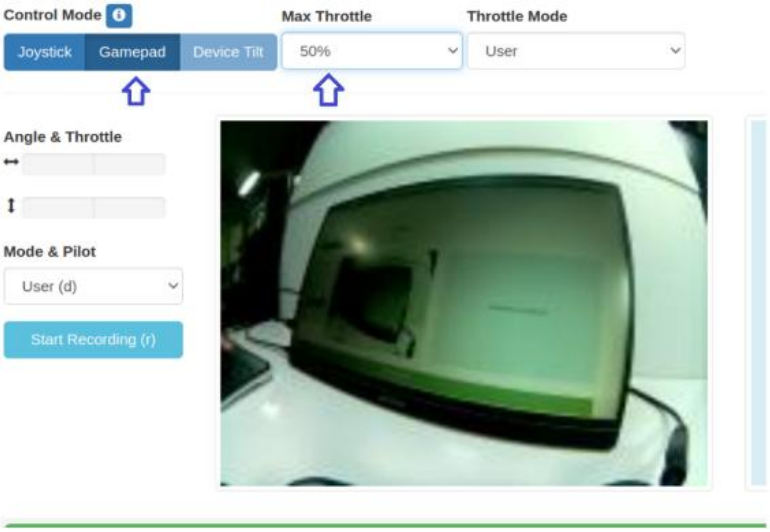
NOTA: Quando o carro é ligado, cria-se uma pasta em /mycar/data chamada tub_#_date para armazenar os dados da sessão. Os dados são recolhidos quando o acelerador é acionado. Para reunir um conjunto de dados limpo, a melhor prática é parar o "disco de manage.py python" usando **Contole + C** e reiniciar para começar o treino com um novo /mycar/data/tub.

Existem 2 opções de disco

1. Usar o comando (recomendado)

Depois de ligar o carro, abra um navegador web a partir do ambiente de trabalho do Raspberry Pi enquanto o carro está ligado ao monitor.

Seleciona Gamepad e define o acelerador para 50% para começares a praticar. Testa as curvas e a velocidade enquanto o carro está na caixa Pi.

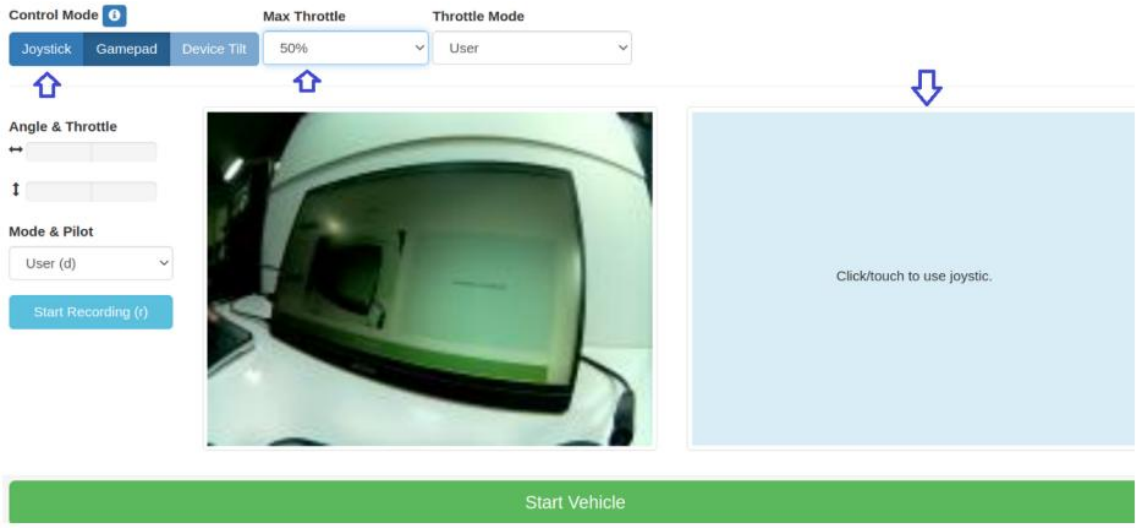


Quando estiver pronto para conduzir na pista, desligue o monitor, coloque o carro na pista e comece a conduzir. Começa devagar até te sentires confortável a manusear o carro.

2. Usar o joystick do navegador web

Vai precisar de usar um tablet ou telemóvel ligado ao mesmo wifi. Depois de ligar o carro, use um tablet ou telemóvel para se ligar ao servidor web do carro usando o IP:8887 do carro. O carro deverá mostrar a sua IP se o Serviço de Visualização OLED estivesse ativado numa etapa anterior.

Selecione Joystick (a área de controlo do joystick está marcada **como click/touch para usar joystic**) Use a área do joystick para conduzir o carro com o dedo.



Quando terminar de conduzir, pare a "drive de manage.py python" usando **Contole + C**.

4. DADOS DE COMBOIO - CRIAR MODELO DE INFERÊNCIA

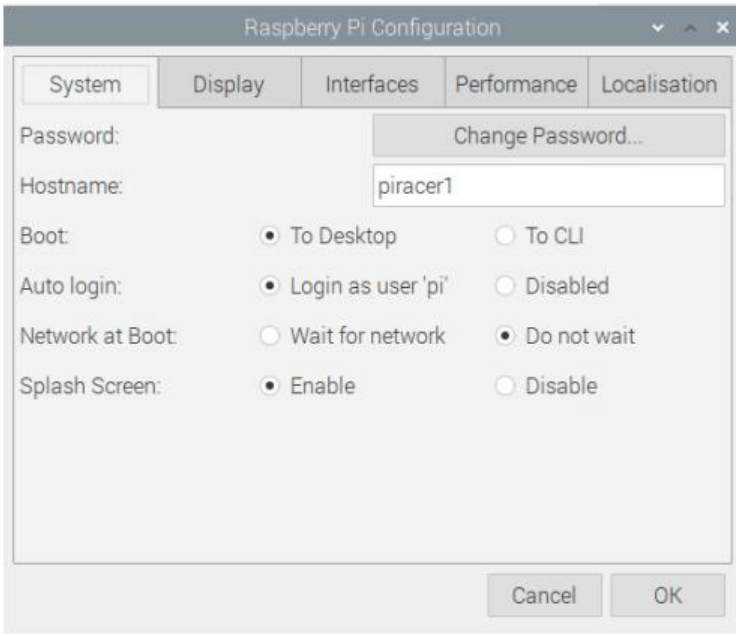
A localização dos dados é /home/pi/mycar/data. Todas as pastas de banho neste diretório serão usadas para criar o seu modelo. Remove tudo o que não quiseres incluído. Pode treinar dados com Raspberry Pi ou com máquina virtual AWS.

1. Treinar com o Raspberry Pi

Dados do Comboio:
Numa nova sessão de terminal no teu PC anfitrião, usa o rsync para copiar a pasta dos carros do Raspberry Pi

```
rsync -rv --progress --partial <hostname>@<your_pi_ip_address>:~/mycar/data/
~/mycar/data/
etc:
rsync -rv --progress --partial piracer1@192.168.1.111:~/mycar/data/ ~/mycar/data/
```

Pode encontrar o nome de host na configuração do Raspberry Pi:



Modelo de comboio:
No mesmo terminal pode agora executar o script de treino na última banheira passando o caminho para essa banheira como argumento. Podes opcionalmente passar máscaras de caminho, como ./data/* ou ./data/tub_?_17-08-28 para recolher vários recipientes. Por exemplo:

```
Python ~/MyCar/Gerir.Py Train --Tub <Tub Pastas nomes Vírgula Separados> --Model
./models/MyPilot.H5
etc:
Python ~/MyCar/Gerir.comboio PY --TUB ./Data/tub_1_24-04-06 --Modelo
./Models/MyPilot.H5
```

Vai custar muito tempo treinar um modelo, por favor tenha paciência. Depois do treino, podes obter um modelo, precisas de copiar o módulo para o teu Raspberry Pi e testá-lo.

```
rsync -rv --show-progress --partial ~/mycar/models/
<hostname>@<your_ip_address>:~/mycar/models/
etc:
rsync -rv --show-progress --partial ~/mycar/models/
piracer1@192.168.1.111:~/mycar/models/
```

5. CARREGAR O MODELO E CONDUZIR DE FORMA AUTÓNOMA

Ligue o seu carro com o modelo

```
cd ~/mycar
Python gerir.PY Drive --modelo ~/mycar/models/mypilot.H5
```

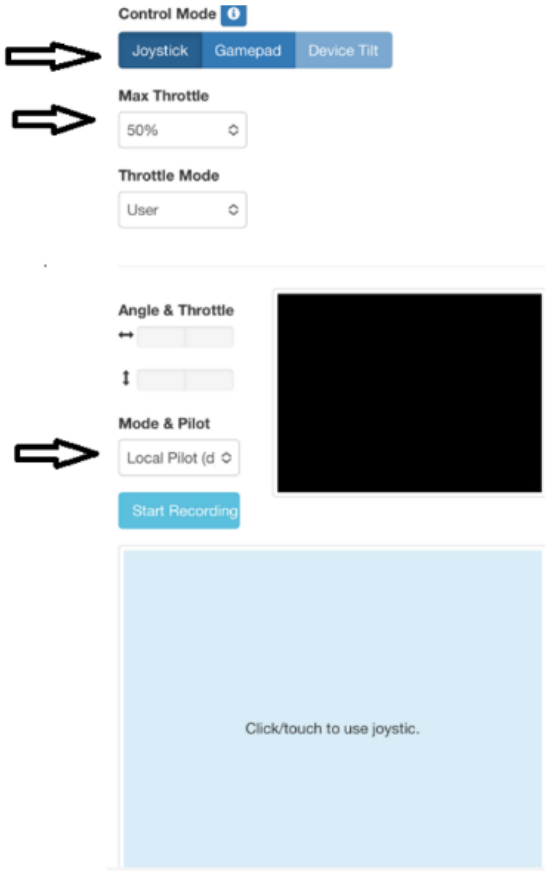
Para condução autónoma ou direção, use o joystick do navegador web noutro dispositivo.

Vai precisar de usar um tablet ou telemóvel ligado ao mesmo wifi. Depois de ligar o carro, use um tablet ou telemóvel para se ligar ao servidor web do carro usando o IP:8887 do carro. O carro deverá mostrar a sua IP se o Serviço de Visualização OLED estivesse ativado numa etapa anterior.

Condução Autónoma

NOTA: Assim que **o Local Pilot** for selecionado, o carro começará a conduzir de forma autónoma

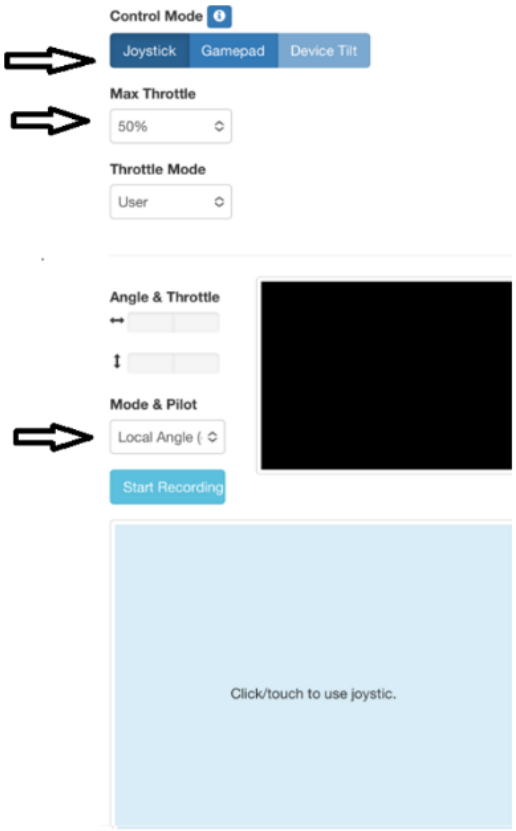
- **Modo de Controlo:** Selecionar **Joystick** (a área de controlo do joystick está marcada como **click/touch para usar joystic**)
- **Acelerador Máximo:** Define o acelerador máximo para cerca de 50% para evitar que o carro vá demasiado rápido
- **Modo & Piloto:** Selecionar Piloto Local (Piloto Local é autónomo)



Direção Autónoma

- **Modo de Controlo:** Selecionar **Joystick** (a área de controlo do joystick está marcada como **click/touch para usar joystic**)
- **Acelerador Máximo:** Define o acelerador máximo para cerca de 50% para evitar que o carro vá demasiado rápido

- **Modo & Piloto:** Selecione o ângulo local (o ângulo local é a direção do carro enquanto você dá o acelerador)



Modelo autónomo mais rápido?

Use o seu modelo de condução autónoma funcional para recolher novos dados a uma velocidade mais rápida. Usa o Local Angle para o carro virar enquanto tu aceleras mais rápido.

II. IA COM AR/VR

6. INTRODUÇÃO

O Meta Quest 3 marca uma mudança decisiva da Realidade Virtual (VR) pura para a Realidade Mista (MR) de alta fidelidade. Ao aproveitar a passagem de cor de alta resolução e um chipset Snapdragon XR2 Gen 2 significativamente mais potente, permite que o conteúdo digital coexista perfeitamente com o mundo físico. Com as suas ópticas mais finas tipo "pancake" e o ecrã 4K+ Infinite, é atualmente a potência mais acessível tanto para consumidores como para desenvolvedores.

6.1 Introdução ao Meta Quest 3 e à Realidade Mista

O Meta Quest 3 utiliza tecnologia de alta resolução Color Passthrough. Ao contrário da realidade virtual tradicional, onde o mundo é inteiramente virtual, **a Realidade Mista (MR)** utiliza câmaras integradas para projetar o ambiente real e depois sobrepõe **elementos WebGL**. O seu código utiliza o modo de AR imersivo, que é o núcleo desta experiência espacial.

Arquitetura do Sistema (The Web Stack)

A aplicação funciona como um "sanduíche" de camadas:

- **Camada 1 (Hardware)**: sensores Quest 3, projetores de profundidade e câmaras RGB.
- **Camada 2 (Navegador)**: Meta Quest Browser (baseado em Chromium com suporte WebXR).
- **Camada 3 (API):** API de Dispositivo WebXR, que atua como a ponte entre hardware e código.
- **Camada 4 (Lógica):** A sua lógica JavaScript usando Three.js para renderização e TensorFlow.js para visão.

7. WEBXR: A ALTERNATIVA BASEADA EM NAVEGADOR

O WebXR permite aos programadores criar experiências imersivas que correm diretamente no Meta Quest Browser. É construído com base em tecnologias web padrão, tornando a "VR na web" tão fácil de aceder quanto um site.

Por que escolher o WebXR?

- Acesso Sem Atritos: Os utilizadores não precisam de descarregar ficheiros grandes da Meta Store; simplesmente clicam num URL e carregam em "Enter VR."
- Frameworks: Utiliza poderosas bibliotecas JavaScript como Three.js, A-Frame (baseado em HTML) ou Babylon.js.
- Multiplataforma: Uma única aplicação WebXR pode frequentemente correr num Quest 3, num telemóvel Android (modo AR) ou num PC de secretária.

7.1 Análise da Biblioteca

Three.js (O Motor de Renderização)

Three.js trata do trabalho pesado do WebGL. Gere a Cena, a Câmara e o Renderizador. No teu código, o renderizador está definido como alfa: true, o que é vital para a RA, pois permite que as partes "vazias" do navegador se tornem uma janela para o mundo real.

TensorFlow.js & COCO-SSD

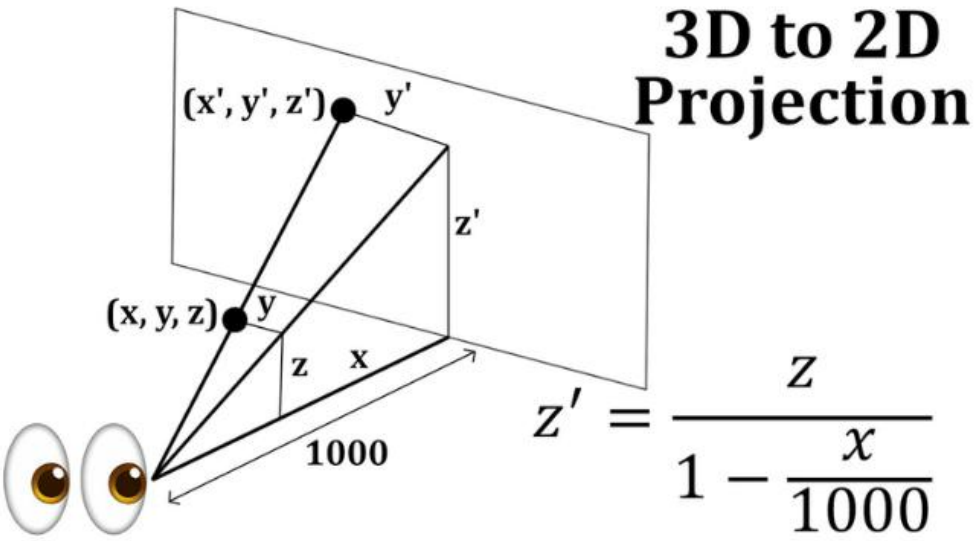
COCO-SSD (Common Objects in Context - Single Shot MultiBox Detector) é um modelo treinado para reconhecer 80 classes de objetos. A beleza do TF.js está no seu **backend WebGL**, que garante que os cálculos de IA são feitos na GPU do Quest em vez da CPU, evitando que o dispositivo sobreaqueça.

7.2 Projeção Matemática (2D para 3D)

Esta é a parte mais técnica do guião. O modelo de IA devolve uma bbox (caixa delimitadora) em pixels (por exemplo, x=100, y=200).

A função detectionTo3D realiza a desprojeção:

- **Normalização**: Converte as coordenadas do ecrã para um intervalo de -1 para $+1$.
- **Cálculo do FOV**: Tem em conta o campo de visão da câmara.
- **Raio Vetorial**: Cria um vetor direcional a partir da posição da cabeça do utilizador em direção ao objeto.
- **Colocação**: Coloca a etiqueta 3D nesse vetor a uma distância definida (DIST).



7.3 Implementação do Teste de Acerto

O teste de acerto permite à aplicação "disparar" um raio invisível (raycast) para o espaço real para detetar interseções com pisos ou mesas. Quando o ***hitTestSource*** devolve um resultado, o cursor (anel verde) é encaixado nessa pose (posição e orientação).

7.4 Pipeline de IA: Deteção e Rotulagem

No código, a deteção não é feita a cada frame (o que faria o headset atrasar), mas sim a cada $2000ms$ (DETECT_INTERVAL_MS).

- **Label** Sprite: Como não Three.js não pode renderizar fontes HTML padrão diretamente numa cena 3D, usamos uma CanvasTexture. "Desenhamos" o texto numa tela invisível de HTML 2D e aplicamo-lo como textura a um Sprite 3D que está sempre virado para o utilizador (outdoors).

7.5 A Necessidade do HTTPS

WebXR e o acesso à câmara (getUserMedia) são classificados como "Funcionalidades Poderosas" pelos fornecedores de navegadores. Só funcionam num **Contexto Seguro** .

- **Exceção Localhost**: Pode testar no seu PC usando http://localhost, mas assim que tenta aceder a esse servidor a partir do seu headset Quest 3, o navegador bloqueia as funcionalidades XR porque detecta um IP de rede inseguro (por exemplo, http://192.168.1.10).
- **A Solução**: Deve usar um **Serviço de Tunelamento** ou **Alojamento Seguro**.

7.6 Meta Quest Link & Modo Desenvolvedor

Para executar e depurar o seu código de forma eficiente, o seu Quest 3 deve ser reconhecido como um dispositivo de programação.

Ativação Passo a Passo:

- **Conta do Desenvolvedor**: Registe-se no **dashboard.oculus.com**. Terá de criar uma "Organização" (pode ser qualquer nome).
- Aplicação Móvel: Abra a aplicação Meta Quest no seu telemóvel, vá ao **Menu > Dispositivos > Definições do Headset > Modo Desenvolvedor** e ative-a.
- **O Link**: Ligue o seu Quest 3 ao seu PC usando um cabo USB-C 3.0 de alta qualidade ou via Air Link (Wi-Fi 6 de alta velocidade).

7.7 A Fundação Three.js (O Boilerplate)

Antes de entrar em AR, temos de inicializar um ambiente 3D padrão. No entanto, para Quest 3, dois cenários são inegociáveis:

- **Alfa & Antialias**:

```javascript
const renderer = new THREE.WebGLRenderer({ antialias: true, alpha: true });
```

- **Ativação XR**:

```
enderer.xr.enabled = true;
```

Isto diz-Three.js para ouvir os dados de head-tracking do Quest ($6DOF$) e aplicá-los automaticamente ao objeto 'câmara'.

7.8 Gerir a Sessão AR (O Ciclo de Vida)

O botão "Enter AR" ativa **o navigator.xr.requestSession**. É aqui que definimos quais os "superpoderes" que a nossa aplicação precisa do hardware do Quest 3.

Funcionalidades Obrigatórias e Opcionais:

- **local-andar**: Isto diz ao Quest para definir a coordenada $Y=0$ no nível físico do piso real.
- **hit-test**: Permite a capacidade de lançar raios contra geometria do mundo real.
- **deteção de planos**: Solicita os dados "semânticos" (sabendo qual a malha que é uma 'tabela' ou uma 'parede').

```JavaScript
const session = await navigator.xr.requestSession('immersive-ar', {
    requiredFeatures: ['local-floor'],
    optionalFeatures: ['hit-test', 'plane-detection']
});
```

8. DEMONSTRAÇÃO DE DETEÇÃO DE OBJETOS AR + IA

8.1 Visão Geral do Projeto

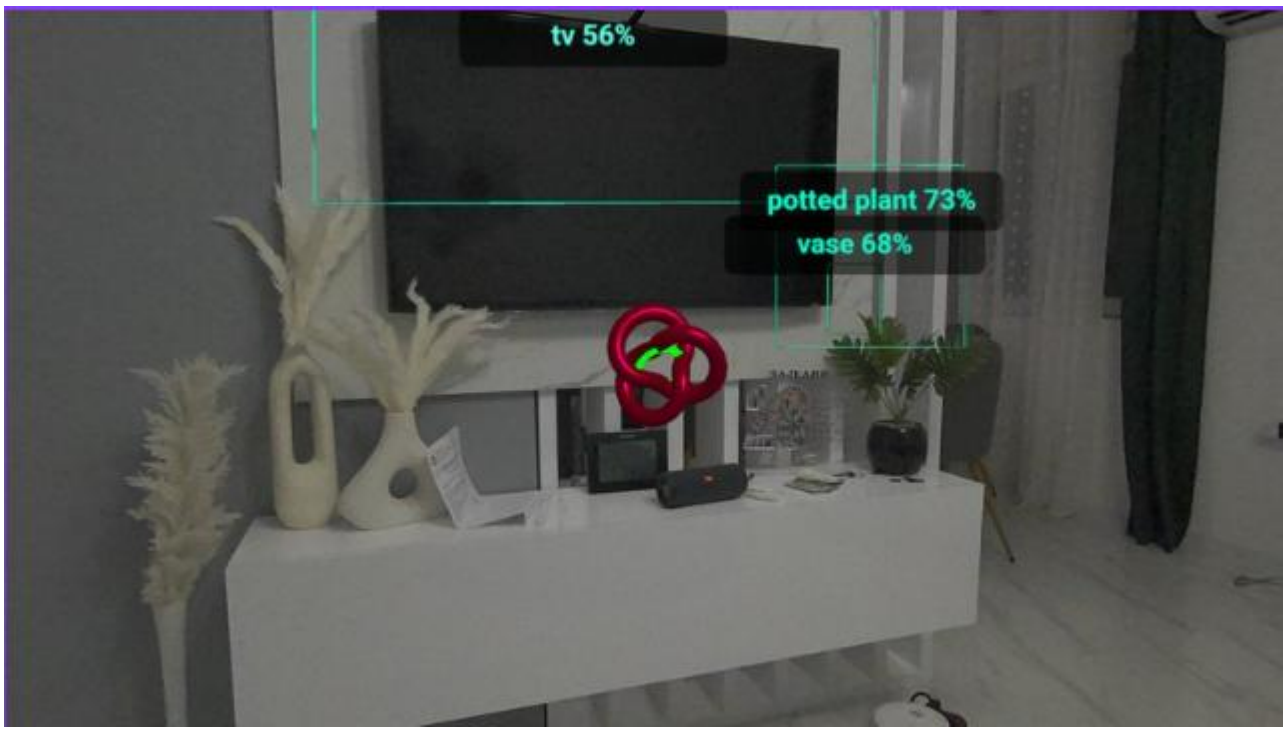
Esta é uma aplicação de Realidade Aumentada WebXR para o Meta Quest 3 que combina:

- **AR passthrough** — a sala real é visível através das câmaras dos auscultadores
- **Interação hit-test** — um cursor virtual que se encaixa em superfícies do mundo real
- **Deteção de objetos por IA** — TensorFlow.js reconhece objetos no feed da câmara e exibe etiquetas no espaço 3D
- **Compreensão de Cena** — A deteção de planos WebXR visualiza superfícies detetadas (chão, paredes, mesa)

URL em direto: http://92.113.18.92/

Em fila indiana:

index.html



8.2 Arquitetura do Projeto

index.html (tudo-em-um)

│

├── HTML → canvas + botão UI + elemento de vídeo oculto

├── CSS → fundo transparente, interface sobreposta

└── JavaScript

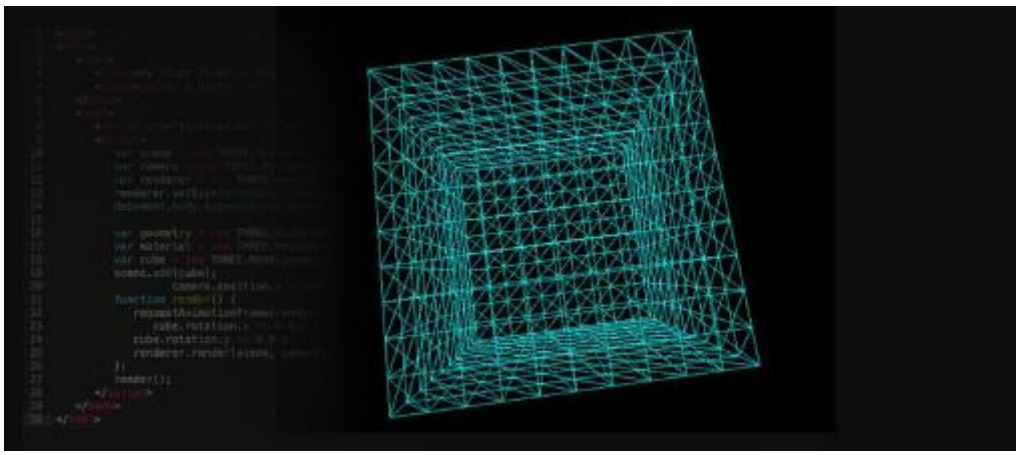
    ├── Three.js → motor de renderização 3D (cena, câmara, materiais)

    ├── WebXR API → sessão AR, hit-test, deteção de plano

    └── TF.js → Inferência de IA (modelo coco-ssd)

Porquê Three.js pura (sem A-Frame)?

Durante o desenvolvimento, descobriu-se que o A-Frame tem o seu próprio loop interno de renderização que entra em conflito com uma sessão WebXR **explícita de ar imersiva**. O A-Frame inicia uma sessão **de immersive VR** (opaca, sem passthrough) em vez de **immersive-ar**. Mudar para Three.js puro deu-nos controlo direto tanto sobre o tipo de sessão como sobre o loop de renderização.



8.3 Stack de Tecnologia

Tecnologia	Versão	Função
Three.js	0.157.0	Renderização 3D, geometria, materiais
WebXR Device API	Nativo do navegador	Sessão AR, teste de acerto, deteção de avião
TensorFlow.js	4.15.0	Motor de inferência ML no navegador
COCO-SSD	2.2.3	Modelo de deteção de objetos pré-treinado
getUserMedia API	Nativo do navegador	Stream de câmara para análise TF.js

8.4 Fluxo de Aplicação

Carregamento da página

→ modelo TF.js (coco-ssd) carrega de forma assíncrona (~6MB)

→ Verificar: navigator.xr.isSessionSupported('immersive-ar')

→ botão "Enter AR" torna-se ativo

O utilizador clica em "Enter AR"

→ Pedido: navigator.xr.requestSession('immersive-ar', {...})

→ Quest ativa o Passthrough (câmara visível através de auscultadores)

→ Simultaneamente: getUserMedia() → stream de câmara para TF.js

→ renderer.setAnimationLoop() inicia (ciclo de renderização XR)

Cada frame (~72fps no Quest 3)

- scene.background = null (garante transparência)
- Teste de acerto → atualiza a posição do cursor
- A deteção de planos → atualiza a visualização da superfície
- A cada 2 segundos → runDetection() → inferência de IA
- renderer.render(cena, câmara)

8.5 Explicação detalhada do código

1. HTML Estrutura (linhas 1–78)

```
HTML
<video id="tfvideo" playsinline muted autoplay></video>
```

Um elemento oculto <vídeo> que recebe o ***fluxo getUserMedia***(). TensorFlow.js usa-o como entrada para inferência — nunca é apresentado ao utilizador, apenas analisado.

```
html
<div id="ui"> ... <button id="ar-button"> </div>
```

Uma camada de interface sobreposta a flutuar acima da **Three.js** tela. ***Eventos de ponteiro: nenhum*** no contentor mas ***eventos de ponteiro: todos*** apenas no botão — para que a sobreposição não bloqueie interações 3D.

2. Three.js Inicialização (linhas 84–109)

```javascript
const renderer = new THREE.WebGLRenderer({ antialias: true, alpha: true });
renderer.xr.enabled = true;
```

alpha: true cria uma tela WebGL com um canal alfa transparente — isto é um pré-requisito para passthrough. ***xr.enabled = true*** ativa **a integração integrada com WebXR da Three.js.**

```javascript
const camera = new THREE.PerspectiveCamera(70, aspect, 0.01, 20);
scene.add(camera);
```

A câmara é **adicionada à cena**. Isto é importante porque os objetos ligados à câmara (entidades filhas) precisam de estar na hierarquia da cena.

```javascript
const cursorGeo = new THREE.RingGeometry(0.04, 0.06, 32);
cursorGeo.rotateX(-Math.PI / 2);
```

A geometria do anel é rodada -90° no eixo X, ficando horizontalmente plana nas superfícies detetadas. Sem esta rotação, ficaria na vertical.

3. makeLabel() — Texto Sprite (linhas 113–138)

```javascript
function makeLabel(text, color, bgColor) {
  const canvas = document.createElement('canvas'); // 512x128 px
  // Draws rounded rect + text
  const texture = new THREE.CanvasTexture(canvas);
  const sprite  = new THREE.Sprite(material);
  sprite.scale.set(0.6, 0.15, 1); // 0.6m wide in 3D space
  return sprite;
}
```

TRÊS. Sprite é um objeto que está sempre virado para a câmara (**outdoors)** — ideal para rótulos no espaço 3D. É desenhado numa tela HTML, convertido numa ***CanvasTexture*** e aplicado a um ***SpriteMaterial***.

depthTest: false garante que a etiqueta é sempre visível mesmo que esteja geometricamente "atrás" de outro objeto 3D.

4. detecçãoTo3D() — Projeção 2D → 3D (linhas 140–163)

Este é o núcleo matemático da integração IA-AR.

```javascript
// Normalize bbox center to [-0.5, 0.5]
const nx = (bbox.x + bbox.width/2)  / videoWidth  - 0.5;
const ny = (bbox.y + bbox.height/2) / videoHeight - 0.5;
// Apply camera FOV (70° horizontal)
const fovH = 70 * Math.PI / 180;
const dir  = new THREE.Vector3(
  Math.tan(nx * fovH),
  Math.tan(-ny * fovV), // flip Y (image top-down, WebGL bottom-up)
  -1                    // forward in camera space (Three.js uses -Z)
).normalize();
// Rotate into world space using the current XR camera orientation
dir.applyQuaternion(camera.quaternion);
// World position = camera position + direction × distance
return camera.position.clone().addScaledVector(dir, placeDist);
```

*Exemplo: Se a IA detetar uma cadeira no terço esquerdo do vídeo, **nx ≈ -0,17**. Isto converte-se num ângulo negativo (à esquerda do eixo do olhar). A etiqueta é colocada 1,8m à frente da câmara nessa direção.*

5. makeBox3D() e bbox2dSize3D() — Caixas Delimitadoras 3D (linhas 165–182)

```javascript
function makeBox3D(w, h, colorHex) {
  const edges = new THREE.EdgesGeometry(new THREE.BoxGeometry(w, h, 0.01));
  return new THREE.LineSegments(edges, material);
}
```

EdgesGeometry + LineSegments *renderiza apenas as arestas de uma caixa — o efeito de uma borda fina em wireframe sem superfície preenchida.*

```javascript
function bbox2dSize3D(bboxW, bboxH, videoW, videoH, dist) {
  const fovH   = 70 * Math.PI / 180;
  const totalW = 2 * dist * Math.tan(fovH / 2); // total visible width at given dist
  return {
    w: (bboxW / videoW) * totalW,  // bbox fraction → meters
    h: (bboxH / videoH) * totalH
  };
}
```

A fórmula ***totalW*** *é a projeção de perspetiva padrão:* ***à distância d****, a largura visível depende do FOV. A partir disto, deduzimos quantos metros correspondem aos píxeis da caixa delimitadora.*

6. runDetection() — Pipeline de Inferência de IA (linhas 205–247)

```javascript
async function runDetection() {
  const predictions = await cocoModel.detect(tfVideo);
  // Clear old labels and boxes
  activeLabels.forEach(({ sprite, box }) => { scene.remove(sprite); scene.remove(box); });
  activeLabels = [];
  predictions.forEach(pred => {
    if (pred.score < 0.5) return; // Confidence threshold: 50%
    // ...create sprite + box...
    activeLabels.push({ sprite, box, expireAt: Date.now() + 4000 });
  });
}
```

cocoModel.detect(tfVideo) *aceita diretamente um elemento <vídeo> — TF.js lê internamente dados de píxeis do quadro de vídeo atual.*

Formato Pred.bbox *:* ***[X, Y, Largura, Altura]*** *em pixels.*

Cada deteção cria um ***par JS (sprite, caixa)*** *que dura 4 segundos.*

7. Sessão WebXR — Detalhes Chave (linhas 274–285)

```javascript
const session = await navigator.xr.requestSession('immersive-ar', {
  requiredFeatures: ['local-floor'],
  optionalFeatures: ['hit-test', 'plane-detection']
});
renderer.xr.setReferenceSpaceType('local-floor');
await renderer.xr.setSession(session);
```

Porquê **a realidade aumentada imersiva** e não **a realidade virtual imersiva**?

- **immersive-VR** = fundo preto opaco (experiência de capacete VR)
- **immersive-ar** = câmara passthrough + conteúdo virtual sobreposto

o espaço de referência do piso local fixa o sistema de coordenadas ao piso da sala — **y=0** está ao nível do piso.

O teste de acerto e a **deteção de aviões** são opcionais porque não são suportados em todos os dispositivos e versões do navegador — declará-los como opcionais impede que a sessão falhe se não estiverem disponíveis

8. Loop de Renderização (linhas 335–395)

```javascript
renderer.setAnimationLoop(function(time, frame) {
  scene.background = null;              // Ensures transparency every frame
  renderer.setClearColor(0x000000, 0); // Alpha = 0 (fully transparent)
  // ...hit-test, plane detection, AI...
  renderer.render(scene, camera);
});
```

Porque **setAnimationLoop** e não **requestAnimationFrame**?

O loop de renderização Three.js XR deve estar integrado com o callback de frames WebXR. **renderer.setAnimationLoop()** liga-se automaticamente à sessão XR quando **renderer.xr.enabled = true**. A alternativa (chamar manualmente **session.requestAnimationFrame()**) requer chamar **o renderer.render()** manualmente, mas corre o risco de perder as matrizes corretas de visualização XR que **Three.js** se aplicam internamente.

scene.background = null deve ser chamado **em cada frame** porque Three.js pode redefinir este valor internamente durante certas operações.

9. Deteção de Planos (linhas 355–378)

```javascript
session.detectedPlanes.forEach(plane => {
  if (!detectedPlanes.has(plane)) {
    // New surface detected — create a mesh
    const label = plane.semanticLabel; // 'floor', 'wall', 'table'...
    const mesh = new THREE.Mesh(PlaneGeometry, transparentMaterial);
    scene.add(mesh);
    detectedPlanes.set(plane, mesh); // store reference
  }
  // Every frame, update the surface pose
  const pose = frame.getPose(plane.planeSpace, xrRefSpace);
  mesh.position.copy(pose.transform.position);
  mesh.quaternion.copy(pose.transform.orientation);
});
```

plane.planeSpace é um XRSpace que rastreia a superfície física. **frame.getPose()** devolve a sua pose no espaço de referência escolhido.

O **mapa** DetectedPlanes **(Map<XRPlane, TRÊS. Mesh>)** impede a criação de malhas duplicadas para a mesma superfície entre frames.

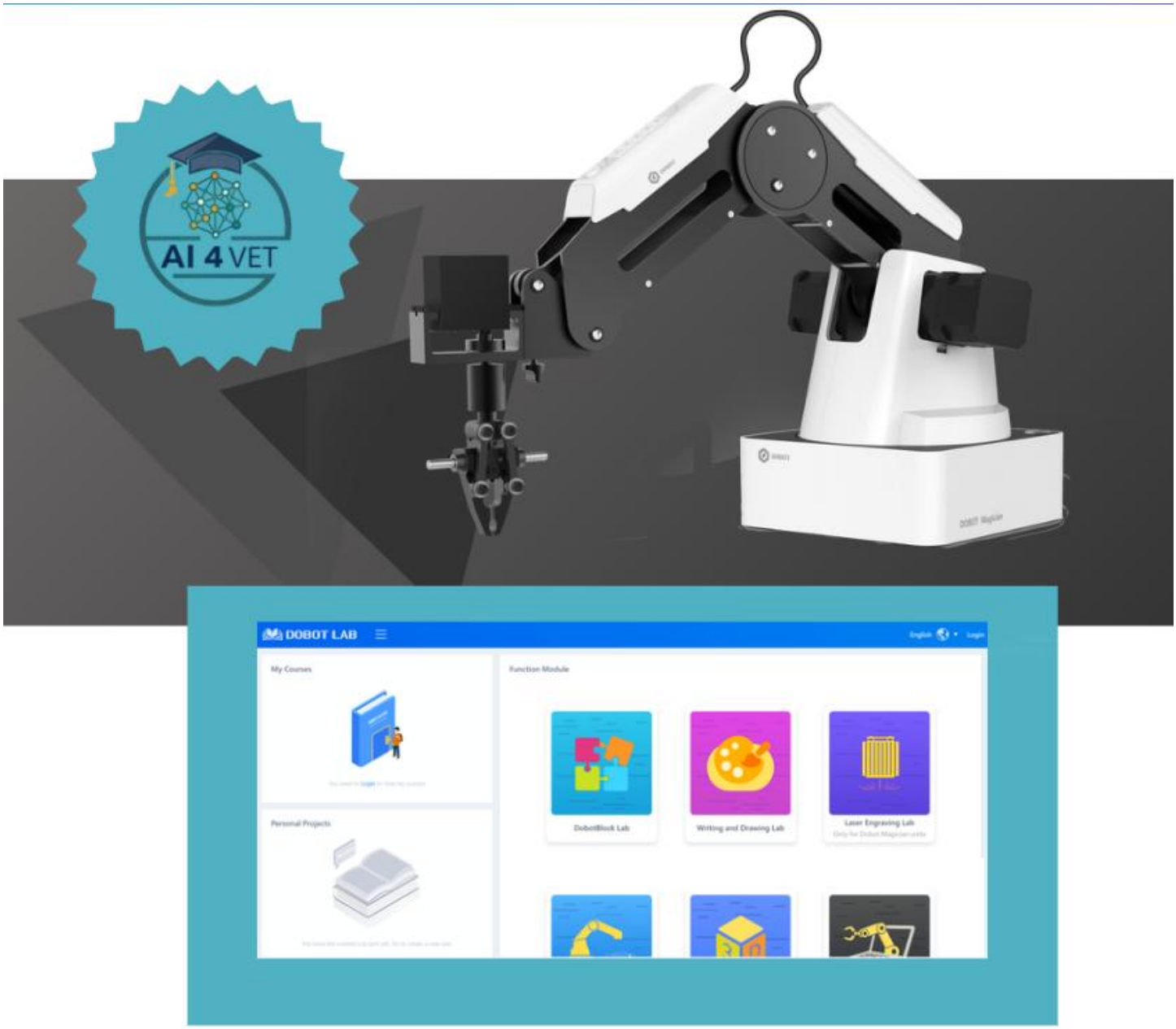
8.6 Limitações Conhecidas

Limitação	Razão
As etiquetas de IA não estão alinhadas perfeitamente com o objeto	As câmaras passthrough do Quest e o getUserMedia entregam diferentes streams com diferentes FOV e calibração ótica
A deteção de planos requer que a Configuração da Sala de Missões seja concluída	O headset deve ter escaneado a sala anteriormente
O getUserMedia pode ser recusado em alguns dispositivos	Depende das permissões do navegador
A inferência de IA não é em tempo real (executa-se a cada 2 segundos)	O Coco-SSD é um modelo relativamente pesado; Alternativas mais leves (SSD MobileNet) dariam maior débito

8.7 Código-fonte

https://github.com/bciric1/ARVR

http://vr.aiforvet.eu/



III. IA COM Dobot

9. INSTRUÇÃO DE INSTALAÇÃO DO DRIVER

Ao ligar o Dobot pela primeira vez, ligue o Dobot ao PC através da porta USB. Depois, ligue o Dobot, e o sistema reconhece automaticamente o hardware correspondente, procura o driver correto e instala-o. No entanto, se falhar a instalação, pode instalá-lo manualmente novamente. O fluxograma da instalação é o seguinte:

9.1 Descarregue o pacote de drivers CH340 e instale-o

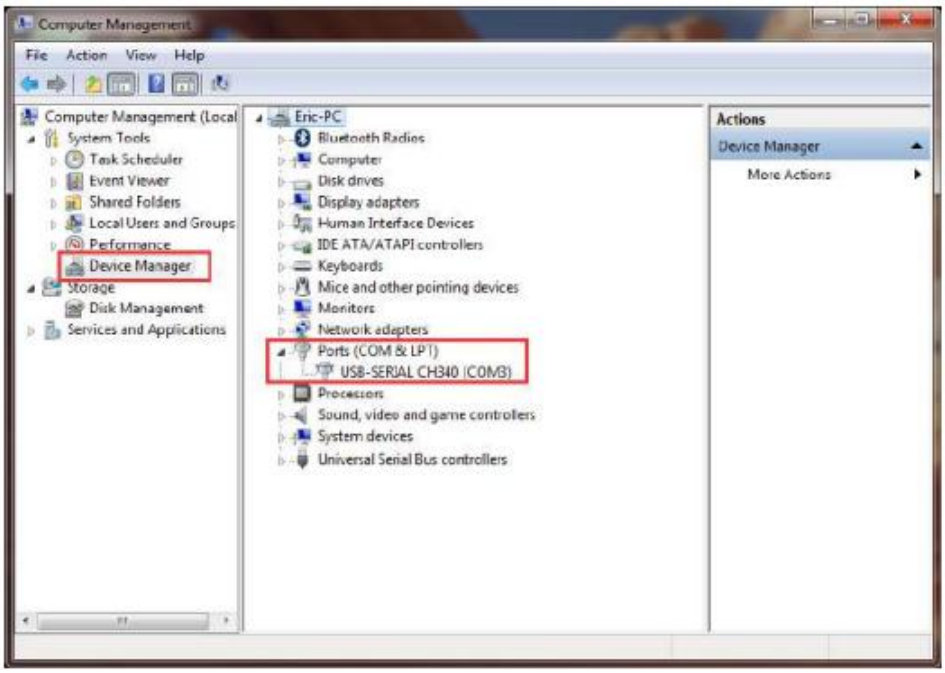
Existem duas versões do driver baseadas em Windows/Linux, por isso por favor escolha a versão correspondente ao seu sistema operativo para descarregar. O driver pode ser localizado no endereço de download:

http://www.dobot.cc/downloadcenter.html?sub_cat=70#sub-download

Depois de descarregar, descompacte e instale o driver

9.2 Verifique se o equipamento funciona corretamente no gestor de dispositivos

Abre o gestor de dispositivos e, se conseguires encontrar a porta COM correspondente "USBSERIAL CH340", então mostra que o driver está instalado com sucesso. A instalação correta é mostrada abaixo:



10. INSTRUÇÕES DE FUNCIONAMENTO DO DOBOTSTUDIO

O software utilizado pelo Dobot Magician é o DobotStudio, e pode descarregar a versão mais recente no nosso site oficial: http://www.dobot.cc/downloadcenter.html?sub_cat=70#sub-download Depois de o ficheiro ser descarregado com sucesso, descompacte e faça duplo clique DobotStudio.exe.

Selecione a porta serial Dobot correspondente, no canto superior esquerdo do DobotStudio e clique em "Conectar". Após uma ligação bem-sucedida, será mostrado o "Desconectar". Quando o Dobot estiver ligado, os parâmetros de coordenadas serão atualizados no lado direito da interface

Existem oito módulos na interface principal do software:

- Ensino e Reprodução: Um sistema para ensinar o Dobot a mover-se. Permite ao Dobot realizar movimentos gravados por controlo manual.
- Escrita e Desenho: Controla o Dobot para escrever, desenhar ou gravar a laser.
- DobotBlockly： Ensina programação básica através de uma interface de puzzle. Intuitiva e fácil de entender.
- Script: Editar a linguagem de script para controlar o Dobot.
- LeapMotion: Controla o Dobot por gesto.
- Rato: Controla o Dobot com o rato.
- LaserEngraving: Grava imagens, formas e palavras através de bitmap com o Dobot.
- Adicione Mais: Adicione ainda mais funções para o Dobot!

10.1 Modo linear

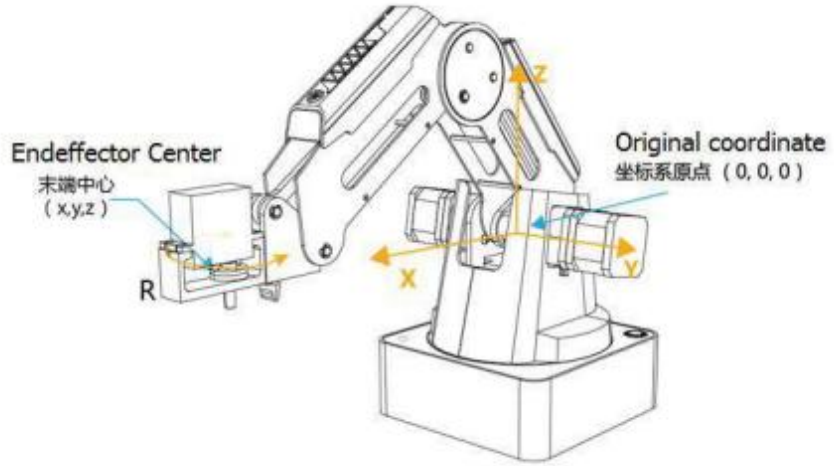
Com base no sistema de coordenadas dos eixos do corpo X, Y, Z, com a origem no centro de três motores. X, Y, Z é a coordenada do centro da plataforma final, e a direção de X é perpendicular à base para a frente, Y é perpendicular à base para a esquerda, e Z é vertical para cima. R indica a rotação da junta servo em relação ao referencial coordenado (no sentido anti-horário é a direção positiva).

(1) Clicar em X+, X- e o Dobot irá mover-se ao longo de X na direção negativa ou positiva;

(2) Clicar em Y+, Y- e o Dobot irá mover-se ao longo de Y na direção negativa ou positiva;

(3) Clicar em Z+, Z- e Dobot irá mover-se ao longo de Z na direção negativa ou positiva;

(4) Clicar em R+, R- e Dobot irá mover-se ao longo de R na direção negativa ou positiva.



10.2 Modo de corrida

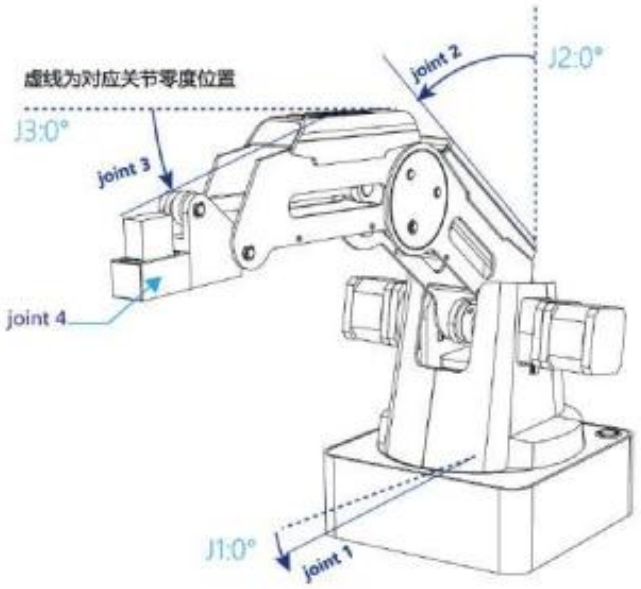
Este movimento é direcionado para um único eixo. Mantenha o botão pressionado e o eixo correspondente move-se de forma independente. Quando o eixo está no máximo, a articulação deixa de se mover. Cada eixo tem o sentido contrário ao dos ponteiros do relógio como direção positiva. Articulação 1、2、3、4 referem-se respetivamente à base, braço traseiro, antebraço e servo.

(1) Clique na Junta1+、Articulação1- e controlar o motor base Dobot para rodar na direção negativa ou positiva;

(2) Clique na Articulação2+、Articulação2- e controla o motor do braço traseiro para rodar na direção negativa ou positiva;

(3) Clicar na Articulação3+、Articulação3- e controlar o motor do antebraço para rodar na direção negativa ou positiva;

(4) Clique na Junta4+、Articulação4- e controle do servo para rodar na direção negativa ou positiva; Entre estes, o alcance de rotação do Joint4 é de ±150°

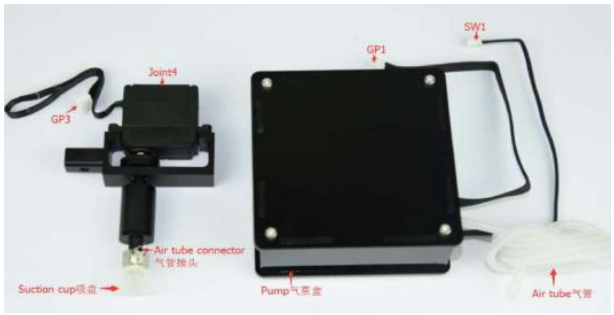


11. ENSINO E REPRODUÇÃO

Aqui vamos aprender a puxar ou agarrar objetos simples usando a função de Ensino e Reprodução. Como precisamos de usar o kit da bomba de ar para a ventosa e o kit de agarre, vamos introduzir estes dois kits juntos.

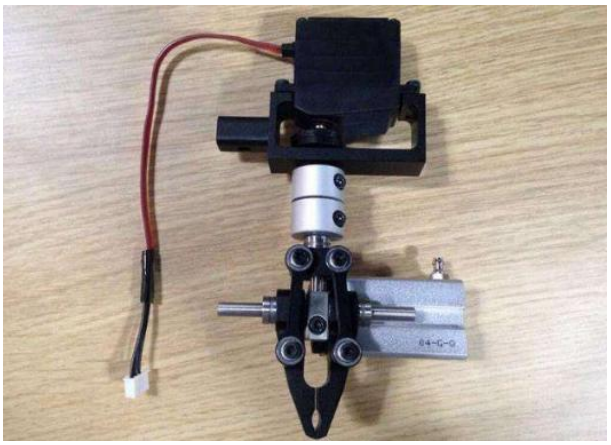
Kit de bomba de ar 11.1

A instalação padrão do Dobot Magician é a ventosa. A caixa da bomba e o kit de ventosa estão mostrados abaixo:



11.2 Kit de Agarra Pneumática

Os acessórios pneumáticos são mostrados na imagem seguinte:

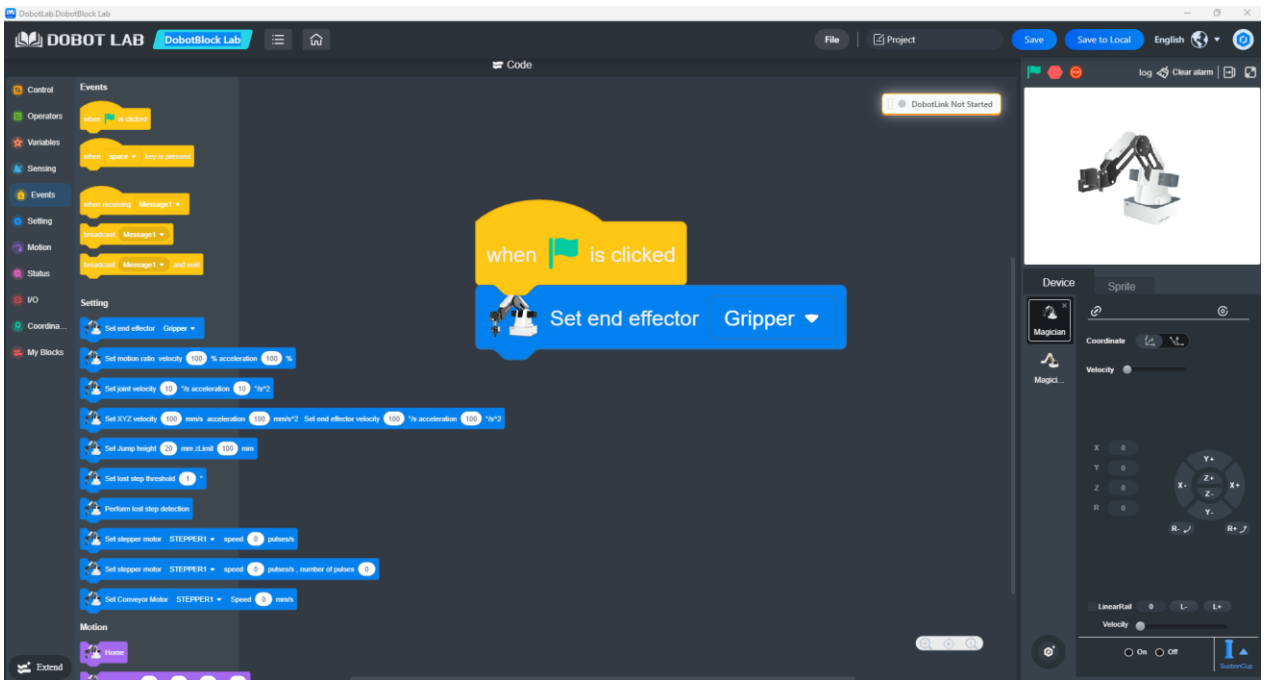


12. PROJECTO: SISTEMA AUTOMATIZADO DE CLASSIFICAÇÃO E TRIAGEM DE RESÍDUOS USANDO DOBOT MAGICIAN

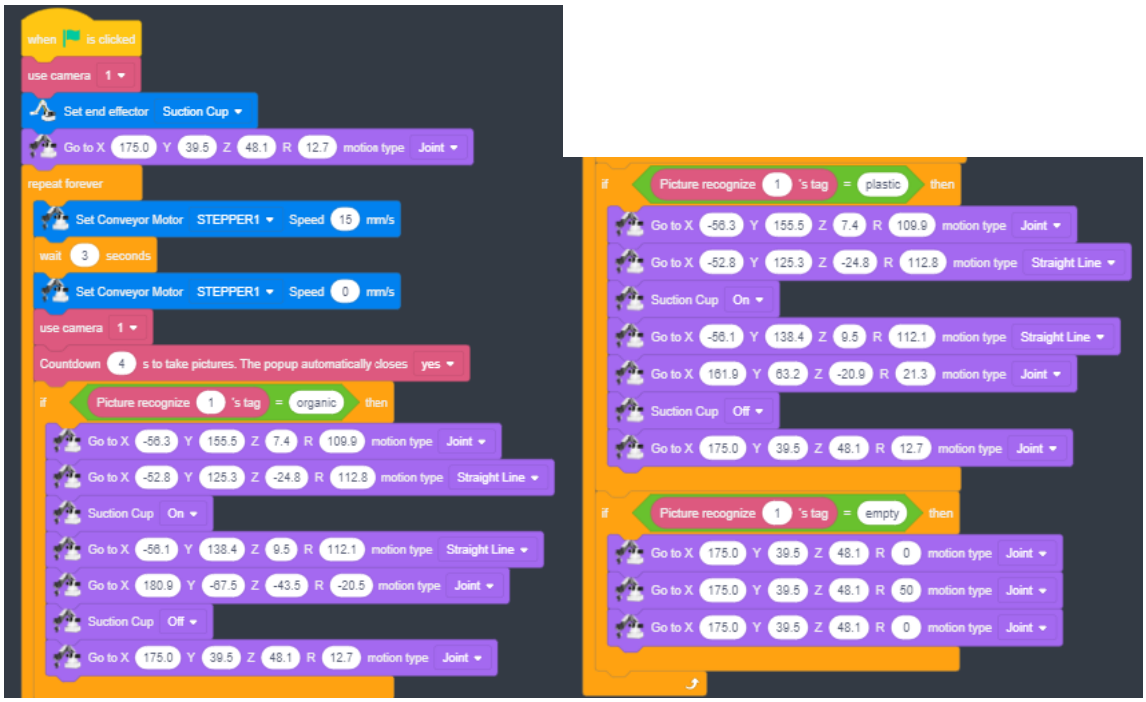
Dobot Blockly é uma plataforma de programação baseada no Google Blockly. Neste processo, os utilizadores podem programar através do formato puzzle, que é simples e fácil de entender. Além disso, os utilizadores podem usar a API integrada do Dobot a qualquer momento.

12.1 Interface Blocada

Abra o DobotStudio e clique em DobotBlock Lab:



Código de bloco 12.2



12.3 Inicialização e Definição do Problema

Contexto e Motivação

Na indústria e ecologia modernas, a triagem manual de resíduos é um processo lento, ineficiente e frequentemente perigoso. O objetivo deste projeto é criar um sistema autónomo que utilize **Visão Computacional** e **manipulação robótica** para reconhecer e separar fisicamente diferentes materiais (neste caso, resíduos orgânicos e plástico).

Enunciado do Problema

O principal desafio reside na sincronização de três subsistemas independentes:

1. **Sistema de Transporte:** Uma correia transportadora que traz objetos para o espaço de trabalho.
2. **Sistema de deteção:** Uma câmara que deve identificar o tipo de item em tempo real.
3. **Sistema de Atuação:** Um braço robótico que deve executar com precisão a recolha e colocação com base no feedback do sensor.

12.4 Arquitetura da Solução

A solução baseia-se na plataforma **Dobot Magician** e num ambiente de programação baseado em blocos (Blockly) que integra módulos de IA para reconhecimento de imagens.

Componentes de Hardware

- **Dobot Braço de Mago:** Um robô de alta precisão de 4 eixos.
- **Ventosa:** Usada como efetivo final pela sua versatilidade em lidar com várias formas.
- **Câmara USB:** Montada acima do cinto para uma vista estável de cima.
- **Correia Transportadora:** Controlada por um motor de passo ligado ao Dobot.

Lógica de Software

O algoritmo segue um **modelo iterativo de circuito fechado**: Iniciar Correia -> Parar -> Capturar -> Analisar -> Ordenar -> Regressar ao Inizar.

12.5 Análise detalhada de blocos de código

O programa está dividido em quatro segmentos principais:

Configuração (Configuração)

- **usar câmara 1**: Declara a entrada de hardware para dados visuais.
- **Definir o efetor final [Ventosa]**: Mapeia suavemente o controlo da bomba de ar para a saída do robô.
- **Ir para X:175, Y:39.5, Z:48.1**: Esta é a "Posição de Segurança." O robô recua para evitar obstruir a câmara e garante um caminho ótimo para qualquer ponto da correia.

Logística (Controlo de Transportadoras)

Dentro do bloco de repetição para sempre:

- **Ajuste o Motor Transportador [STEPPER1] Velocidade 15 mm/s**: Ativa a correia a uma velocidade controlada para evitar que objetos leves deslizem.
- **espera 3 segundos**: Uma constante de tempo crítica que define a distância entre os itens.
- **Velocidade 0**: Para a correia para que o robô possa executar uma "palheta estática" precisa.

Inspeção por IA (Reconhecimento de Imagem)

- **Contagem decrescente 4 s**: Fornece tempo de estabilização para a câmara focar e para o pop-up da IA processar a imagem.
- **Se a Imagem reconhecer a etiqueta 1 = [orgânico/plástico]**: Chama a API de Deep Learning para comparar o frame atual com um conjunto de dados treinado e devolve uma tag de classe.

Cinemática e Manipulação

Para cada ramo (orgânico/plástico), o sistema utiliza dois tipos de movimento:

1. **Movimento Articular (PTP):** Move todas as articulações simultaneamente para o percurso ponto a ponto mais rápido.

2. **Linha Reta (Linear):** Move o braço estritamente verticalmente no eixo Z para garantir que a ventosa faz contacto perfeito sem derrubar o objeto.

12.6 Algoritmo Operacional (Passo a Passo)

- **START:** Inicializar instrumentos e definir o efetor final.
- **TRANSPORTE:** Mover o cinto durante 3 segundos e depois parar.
- **SENTIDO:** Capturar imagem e identificar a etiqueta através de IA.
- **DECISÃO:**
- **Se for orgânico:** Vai buscar coordenadas, liga a sucção, vai para a caixa orgânica, **desliga** a sucção.
- **Se for de plástico:** Vai buscar coordenadas, liga a sucção, vai para a caixa de plástico, desliga a sucção.
- **Se vazio:** Realize um movimento de "sacusão" de sinalização (rodando o eixo R).
- **REINICIAR:** Regressar às coordenadas iniciais de segurança.
- **LOOP:** Repete o ciclo indefinidamente.

12.7 Especificações Técnicas de Coordenadas

Com base no script, as seguintes coordenadas foram calibradas:

Point	X (mm)	Y (mm)	Z (mm)	R (°)	Descrição
Casa	175.0	39.5	48.1	12.7	Posição em marcha lenta/espera
Abordagem	-56.3	155.5	7.4	109.9	Posição acima do item
Agarra	-52.8	125.3	-24.8	112.8	Ponto de contacto (Rebaixado)
Contentor Orgânico	180.9	-67.5	-43.5	-20.5	Eliminação para produtos orgânicos
Contentor de plástico	161.9	63.2	-20.9	21.3	Eliminação de plásticos

12.8 Resolução de Problemas de Implementação

- **Agarrar Impreciso:** Certifique-se de que o objeto está centrado. Se variar, usa guias na correia ou integra feedback dinâmico X/Y da câmara de IA.
- **Erros de Classificação:** O reconhecimento da IA é sensível à luz. Use uma cor de cinto de alto contraste e iluminação LED externa consistente.
- **Segurança:** Certifique-se sempre de que o espaço de trabalho está livre de obstáculos antes de iniciar o ciclo de repetição indefinida.

IV. IA COM rPI 5

13. INTRODUÇÃO

O Raspberry Pi é um pequeno computador do tamanho de um baralho de cartas e capaz de correr um sistema operativo Linux completo enquanto consome apenas uma energia modesta. Inclui portas USB para ligar teclado e rato, juntamente com uma variedade de outros periféricos, um adaptador ethernet e ligações a monitores HDMI. O Raspberry Pi foi originalmente construído para a educação, mas encontrou utilizações frutíferas para amadores, automação doméstica, aplicações industriais e como uma tecnologia apropriada para uso em escolas na maioria do mundo. É fabricado para cumprir as diretivas RoHS (Restrição de Substâncias Perigosas) e depende de um único cartão microSD para o seu armazenamento. Corre uma variante do Linux chamada Raspberry Pi OS e suporta uma grande variedade de software open source, incluindo uma infinidade de programas educativos. Além disso, pode ser adquirido a um preço modesto. Este guia foi escrito principalmente a pensar em estudantes de engenharia e ciência da computação, mas será do interesse de outros que tenham grande interesse em aprender mais sobre programação e computação técnica.

13.1 Configuração Inicial do Raspberry Pi

Insira um cartão SD com o Raspberry Pi OS no Raspberry Pi e aplique energia. Quando arrancares o sistema operativo Raspberry Pi normal pela primeira vez, estará a correr um ambiente gráfico de ambiente de trabalho com uma interface orientada a menus amigável. No primeiro arranque, aparece uma caixa de diálogo a guiá-lo numa configuração inicial. Siga as instruções para configurar o teclado e o nome de utilizador. Fornece um nome de utilizador e uma palavra-passe conforme solicitado. Configura as definições do WiFi e seleciona a opção "Atualizar Software" (nota que isto pode demorar muito tempo quando configurares o Raspberry Pi).

13.2 Começar com a linha de comandos

Existe também uma-versão do sistema operativo baseada em linha de comandos chamada Raspberry Pi OS Lite. Esta versão do sistema operativo consome menos energia do que o sistema operativo Raspberry Pi normal com um ambiente de ambiente de trabalho e pode ser usada em modelos mais antigos do Raspberry Pi com menos RAM. Ao configurar o Raspberry Pi com o sistema operativo Lite, ser-se-á solicitado a configurar o teclado e fornecer um nome de utilizador e palavra-passe. A versão Lite do sistema operativo é muito adequada para usar o Raspberry Pi como servidor, como sistema embutido ou numa aplicação IoT (Internet das Coisas). No entanto, para uso regular de ambiente de trabalho, a versão normal do Raspberry Pi OS é a melhor. Ao usar o Desktop OS, a linha de comandos ainda pode ser acedida através da aplicação Terminal. Alternativamente, pode também ser acedido através de uma consola virtual pressionando CTRL+ALT+F1, o que entra num terminal de ecrã inteiro. Se aparecer um aviso de login, pode iniciar sessão usando o nome de utilizador e a palavra-passe que configurou durante a configuração. Pode-se regressar ao ambiente de trabalho a partir de uma consola virtual pressionando CTRL+ALT+F7. Consolas virtuais adicionais podem ser acedidas independentemente usando CTRL+ALT, juntamente com as teclas F2 a F6.

13.3 A Concha

Assim que entrares na linha de comandos, vais estar a correr num shell Linux. Em termos simples, um shell é um interpretador de comandos que fornece um conjunto rico de comandos que podem ser usados para executar programas e interagir com o sistema operativo. O Raspberry Pi OS usa o Bash (Bourne Again Shell) por defeito, um shell baseado num shell mais antigo chamado Bourne Shell. O BASH é popular entre utilizadores de Linux e apresenta completamento automático na linha de comandos usando a tecla tabulador, podendo ser usado para criar programas chamados scripts shell. Existem vários tipos de shells Linux que podem ser usados. Como indicado, o shell Linux padrão é o shell Bash, mas outros shells também estão disponíveis. Cada concha tem as suas próprias características e opções. Por exemplo, para mudar o shell por defeito de Bash para o shell Z (zsh), escreva o seguinte

```
sudo apt install zsh -y
chsh -s /bin/zsh
```

Depois de emitir estes comandos, faça logout e volte a entrar e agora deve estar a correr com zsh. Várias opções de configuração podem ser definidas dentro de um ficheiro chamado .zshrc localizado na sua pasta principal.

13.4 Comandos shell

Alguns dos comandos disponíveis no shell são resumidos abaixo:

Comando	
Diretório de CD	Altera o diretório de trabalho atual para diretório
PWD	Mostra o nome do diretório de trabalho atual
Diretório mkdir	Crie um novo diretório chamado diretório
Diretório RMDIR	Remove o diretório chamado diretório
ls	Mostrar uma lista de ficheiros no diretório atual
CP F1 F2	Copiar um ficheiro da fonte f1 para o destino f2
Nome de ficheiro RM	Remover um nome de ficheiro
MV F1 F2	Mover um ficheiro de f1 para f2
Anfitrião FTP	Transferir ficheiros para e a partir do host

Estes comandos representam apenas uma parte dos comandos de utilizador disponíveis num shell Linux. Um manual online referido como páginas man (manual) oferece ajuda nos muitos comandos e programas que podem ser chamados a partir do shell. A sintaxe para invocar a utilidade homem é a seguinte:

```
Homem - Nome de Comando
```

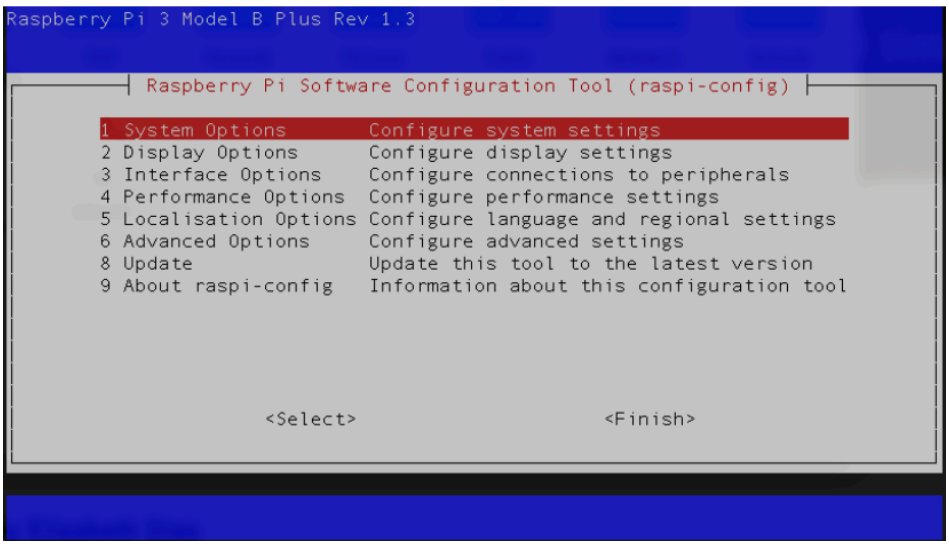
A informação relativa ao nome do comando especificado será então exibida no ecrã.

13.5 Configuração do Raspberry Pi OS

O sistema operativo Raspberry Pi vem com uma ferramenta chamada raspi-config que pode ser usada para configurar uma grande variedade de definições e serviços. Para executar esta utilidade, escreva:

```
sudo raspi-config
```

Isto irá lançar a ferramenta de configuração do Raspberry Pi dentro do terminal com um menu principal como o mostrado abaixo:



A ferramenta de configuração do Raspberry Pi pode ser usada para ativar a interface da câmara, bem como comunicações seriais, I2C e SPI. Inclui também uma opção para arrancar diretamente na linha de comandos em vez do ambiente de trabalho.

13.6 Ligação remota ao Raspberry Pi

É possível correr o Raspberry Pi sem headless, sem ecrã, teclado ou rato. Isto acontece frequentemente ao usar o Raspberry Pi numa aplicação embutida ou numa configuração de IoT (Internet das Coisas). As subsecções seguintes descrevem como se ligar remotamente ao seu Raspberry Pi usando uma das seguintes opções:

- usando um cabo serial USBtoTTL
- usando SSH através de uma ligação Ethernet ou WiFi
- Serviço Raspberry Pi Connect

Alguns modelos do Raspberry Pi podem ser ligados através de um cabo USB. Isto funciona ativando o Modo Gadget USB, que permite que uma porta USB se apresente como uma variedade de tipos diferentes de dispositivos. Sabe-se que isto funciona com o Raspberry Pi Zero e não com a maioria dos outros modelos. As duas abordagens descritas abaixo devem funcionar com todos os modelos do Raspberry Pi.

14. INTRODUÇÃO ÀS LINGUAGENS DE PROGRAMAÇÃO

Os repositórios do Raspberry Pi incluem suporte para COBOL, bem como um conjunto rico de linguagens de programação mais modernas como Python, C, C++ e Java. Suporta linguagens de grama profissional mais específicas e legadas. Geralmente, os paradigmas para linguagens de programação dividem-se em três categorias gerais:

• Linguagens de programação procedural

• linguagens de programação orientadas a objetos

• linguagens de programação funcional

14.1 A Linguagem de Programação Python

Python foi inventado no início dos anos 1990 por Guido van Rossum. Python pode ser escrito usando um paradigma procedural ou orientado a objetos. É um projeto open source amplamente disponível, utiliza sintaxe simples, inclui bibliotecas ricas e oferece uma variedade de ferramentas de programação. O código-fonte em Python é executado numa máquina virtual que traduz o código para o código específico executado pelo processador. Como o Python é interpretado por uma máquina virtual, é independente da plataforma.

O Raspberry Pi deve ter o Python instalado por defeito e pode executar programas em Python diretamente da linha de comandos. Para entrar num programa, primeiro precisa de um editor de texto simples. Se estiver a usar um ambiente de trabalho, existe um ambiente de desenvolvimento integrado (IDE) amigável, gráfico e adequado para iniciantes chamado Thonny. Para instalar o Thonny, escreva:

```
Sudo Apt Install Thonny
```

Para utilizadores mais avançados num ambiente gráfico, o programa vscode fornece um excelente editor para programação em várias linguagens diferentes, incluindo Python. Como descrito anteriormente, o vscode também pode ser executado para editar ficheiros remotamente. Se estiver a usar a linha de comandos, pode

usar qualquer um dos editores de linha de comandos descritos nas secções anteriores, incluindo vi, emacs ou nano. Por exemplo, para editar um ficheiro fonte em Python chamado hello.py, escreva o seguinte:

```
Nano : Olá.py
```

De seguida, introduza o seguinte código no ficheiro fonte:

```
name = input('Qual é o teu nome? ')
print ('Olá', nome, bem-vindo ao Raspberry Pi!')
imprimir ('Adeus')
```

De seguida, guarda e sai do nano e executa o ficheiro escrevendo:

```
python3 Olá.py
```

O programa deverá funcionar como esperado.

14.2 Compilação e Execução de um Programa C/C++

O Linux dispõe de várias ferramentas para suportar o desenvolvimento de software tanto em C como em C++. Na verdade, o próprio sistema operativo Linux está escrito em C. A linguagem de programação C é uma linguagem procedural, e o C++ baseia-se em C para fornecer suporte para programação orientada a objetos. Os compiladores que vamos usar sob Linux são o GNU C Compiler (gcc) e o GNU C++ Compiler (g++). Para garantir que as ferramentas do compilador GNU C/C++ estão instaladas, escreva:

```
Sudo apt install gcc g++ gdb build-essential
```

Por exemplo, para introduzir um programa C simples chamado hello.c usando o nano editor, escreva o seguinte:

```
Nano : Olá.c
```

Usando o editor, introduza o seguinte código no ficheiro fonte:

```
/* Um programa Raspberry Pi C */
#include <stdio.h>
int main(void)
{
    printf("Olá mundo.\n");
    printf("Compilado e executado num Raspberry Pi.\n");
    retorno 0;
}
```

Guarde e saia do editor. Para compilar o seu código-fonte, escreva o seguinte no prompt numa janela de terminal. Por exemplo, para compilar o programa hello.c acima, pode inserir:

```
gcc -Parede -olá, olá.c
```

O compilador gcc tem inúmeras outras opções de linha de comandos que pode usar. Para mais informações, consulte as páginas man. Note que o compilador C++ pode ser invocado usando g++ em vez de gcc. Para

executar o programa compilado no diretório de trabalho atual, não se esqueça de especificar um ./ à frente do nome do programa para especificar o caminho como diretório atual. Por exemplo, para executar o programa hello.c depois de o compilar como ini, escreva:

```
.\olá
```

Se não for fornecido um nome de ficheiro de saída ao compilador, o nome de ficheiro de saída predefinido será a.out.

14.3 Compilação e Execução de Programa Java

Também é possível desenvolver e executar programas Java usando Linux no Raspberry Pi. Existem dois pacotes diferentes que podem ser instalados: um fornece o Java Runtime Environment (JRE) e outro fornece o Java Development Kit (JDK). O JRE apenas permite executar programas em Java, mas o JDK permite compilar e executar programas em Java. Para instalar o kit de desenvolvimento OpenJDK Java, escreva o seguinte:

```
sudo apt install default-jdk
```

Depois de instalado, podes compilar um programa em Java. Por exemplo, entre em cena o seguinte programa simples em Java chamado hello.java usando um editor de texto simples:

```
/* Um programa Java */
classe Main {
 public static void main(String args[]) {
 Sistema.Fora.println ("Olá mundo.\n");
    }
}
```

Compila o programa escrevendo o seguinte no prompt:

```
Javac Olá.Java
```

onde myprogram.java é o nome de um ficheiro fonte Java. Para executar um programa em Java, escreva o seguinte no prompt: **java Main** , onde Main é o nome da classe onde o programa começa.

15. EXPLORANDO A INTELIGÊNCIA ARTIFICIAL

15.1 Suporte de Hardware e Software para IA

Modelos recentes do processador Raspberry Pi incluem múltiplos núcleos ARM. Apesar das suas respeitáveis capacidades de hardware, o Raspberry Pi pode ter dificuldades com as exigências computacionais de aplicações de inteligência artificial (IA) mais exigentes. Para aplicações mais exigentes, existe uma placa adicional Raspberry Pi — referida como HAT (Hardware Attached on Top) — que fornece uma NPU (Unidade de

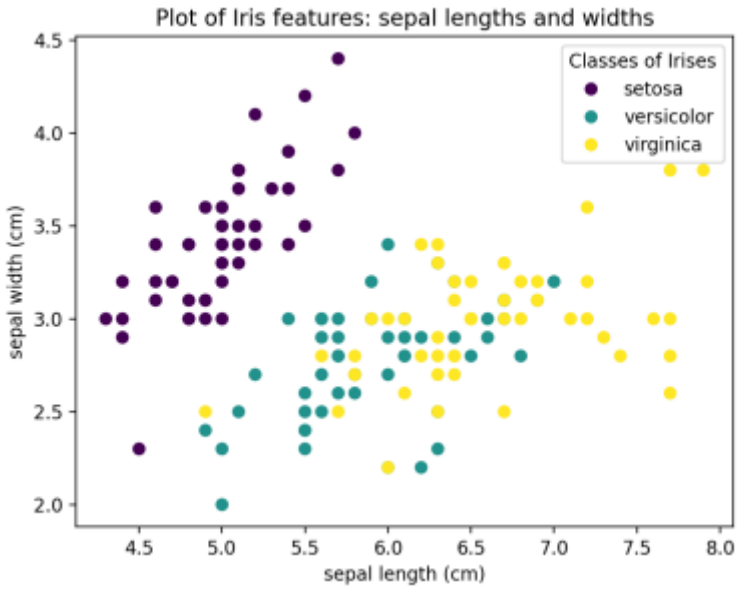
Processamento Neural) para acelerar os cálculos de aprendizagem automática. O Raspberry Pi AI HAT+ oferece um processador de IA de alto desempenho e eficiente em termos energéticos para o Raspberry Pi 5. Para além do suporte de hardware, existe uma grande variedade de poderosas bibliotecas de aprendizagem automática que podem ser usadas com o Raspberry Pi, incluindo scikitlearn e LiteRT. Python inclui várias bibliotecas de suporte poderosas, como Matplotlib e plotly para plotamento, NumPy para fornecer operações rápidas em arrays, e Pandas para análise e manipulação de dados. As secções seguintes fornecem exemplos de algumas destas bibliotecas em ação.

15.2 SciKit Learn

SciKit Learn é uma biblioteca de aprendizagem automática open source que funciona com Python. O SciKit Learn oferece muitas funcionalidades diferentes, incluindo algoritmos de regressão e agrupamento, Análise de Componentes Principais (PCA), Análise Discriminante Linear (LDA) e Máquinas de Vetores de Suporte (SVMs). Todas as formas de aprendizagem automática requerem um conjunto de dados utilizado para treino. O SciKit inclui uma seleção de conjuntos de dados de brinquedos que podem ser usados para experimentar as bibliotecas de aprendizagem automática. As secções seguintes demonstram PCA e SVMs usando o clássico conjunto de dados de flores de íris, que faz parte da coleção de conjuntos de dados de brinquedos. Este conjunto de dados de íris foi originalmente compilado pelo biólogo Ronald Fisher num artigo conhecido de 1936. Este conjunto de dados compreende amostras de três espécies da flor da íris (Iris setosa, Iris virginica e Iris versicolor) e medições do comprimento e da largura tanto dos sépalas1 como das pétalas. A biblioteca SciKit Learn inclui este conjunto de dados de íris para fins de teste e demonstração. O código Python seguinte utiliza o Matplotlib para representar a largura da sépala em função do comprimento da sépala para cada uma das três classes de íris do conjunto de dados.

```
Dos  conjuntos de dados de importação sklearn
importar matplotlib.pyplot como plt
# Carregar o conjunto de dados clássico da íris
íris = conjuntos de dados.load_iris()
# Plotar comprimentos de sépalas vs. larguras para íris no conjunto de dados
fig, ax = plt.Subenredos()
dispersar = machado.Scatter (íris.data[:, 0], íris.data[:, 1], c=íris.alvo)
Axe.set_title("Gráfico das características da íris: comprimentos e larguras de
sépalas")
Axe.set(xlabel=íris.feature_names[0], ylabel=íris.feature_names[1])
fig = machado.Lenda (dispersão.legend_elements()[0], íris.target_names, loc="best
", título="Classes de Íris")
Plt.Show()
```

Executar este código resulta no seguinte gráfico do conjunto de dados de íris:



15.3 Classificação de Imagens SVM

O exemplo seguinte é inspirado no exemplo do SciKit sobre o reconhecimento de dígitos manuscritos e utiliza o conjunto de dados de dígitos manuscritos do repositório de aprendizagem automática da UC Irvine. Este conjunto de dados tem 1797 amostras de imagem compostas por um array de 8x8 píxeis com 10 classes, onde cada classe se refere a um dígito (09). Um curto programa em Python para carregar o conjunto de dados de dígitos manuscritos e exibi-los é mostrado abaixo:

```
importar matplotlib.pyplot como plt
A partir  de  dados de importação SkLearn, métricas, SVM
Do sklearn.model_selection importação train_test_split

# carregar conjunto de dados de dígitos escritos à mão
dígitos = conjuntos de dados.load_digits()

# mostrar uma amostra de dez dígitos manuscritos no conjunto de dados
fig, eixos = plt.Subenredos (nrows=2, ncols=5)
Para Ax, digito em Zip(Axes.achatar (), dígitos.Imagens):
    Axe.set_axis_off()
 Axe.imshow(digit,cmap='gray')
fig.tight_layout()
Plt.Show()
```

O gráfico que mostra dez dígitos manuscritos no conjunto de dados é mostrado abaixo.

Podemos dividir o conjunto de dados de dígitos manuscritos em dois conjuntos: um conjunto de treino e um conjunto para testes. O SciKit tem uma função para pegar no conjunto de dados de dígitos maiores e dividi-lo ao meio em conjuntos de treino e teste da seguinte forma:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0,5)
```

onde X é um array das características e y é um vetor de etiquetas que classificam os dados. Assim, podemos treinar um SVM usando metade do conjunto de dados para treino e depois usar a metade restante para testes. A precisão da SVM pode ser determinada comparando os dígitos previstos pela SVM com as etiquetas reais dos dígitos manuscritos e a taxa de reconhecimento pode ser reportada. A taxa de reconhecimento é o número total de imagens de dígitos corretamente identificadas dividido pelo número total de imagens de teste. O código seguinte define um classificador SVM com base em metade do conjunto de dados e depois determina a taxa de reconhecimento usando a outra metade do conjunto de dados como imagens de teste.

```
importar matplotlib.pyplot como plt
A partir  de  dados de importação SkLearn, métricas, SVM
Do sklearn.model_selection importação train_test_split

# Lê dígitos e remodela os dígitos como um vetor de imagens
dígitos = conjuntos de dados.load_digits()
n_samples = len(dígitos).imagens)
dados = dígitos.imagens.reformulação((n_samples, -1))

# Criar um classificador de máquina de vetores de suporte
SVM = SVM.SVC()
# Dividir o conjunto de dados: metade para treino e metade para testes
X_train, X_test, y_train, y_test = train_test_split(
dados, dígitos.alvo, test_size=0,5, baralhar=Falso)

# Treina com dígitos manuscritos no conjunto de treino
SVM.Ajuste(X_train, y_train)
# Prever o valor do dígito usando o conjunto de teste
previsto = SVM.Prever(X_test)

# calcular a taxa de reconhecimento
Jogos = 0
para x no intervalo (len(y_test)):
se y_test[x] == previsto[x]:
Jogos += 1
recognition_rate = (matches/len(y_test)) * 100;
print(f'Taxa de reconhecimento: {recognition_rate:.2f}%')
```

A saída deste código mostra o seguinte:

Taxa de reconhecimento: 96,11%

Isto indica uma taxa de reconhecimento respeitável usando uma SVM para classificação de dígitos manuscritos. Para mais detalhes, podemos traçar uma matriz de confusão para visualizar a precisão de um algoritmo de aprendizagem automática. A matriz de confusão está organizada em linhas e colunas: cada linha representa cada uma das classes de um conjunto de dados e cada coluna representa as previsões feitas. Idealmente, as classes reais e as previsões alinham-se perfeitamente de modo que a diagonal da matriz de confusão seja preenchida com 100 e com 0s em todo o resto. O mártix diagonal corresponde às taxas de reconhecimento verdadeiras, enquanto todas as outras células representam a ocorrência de classificações falsas.

16. EXEMPLOS

16.1 Construa o Seu Próprio Sistema de Segurança "Detetor Parental"

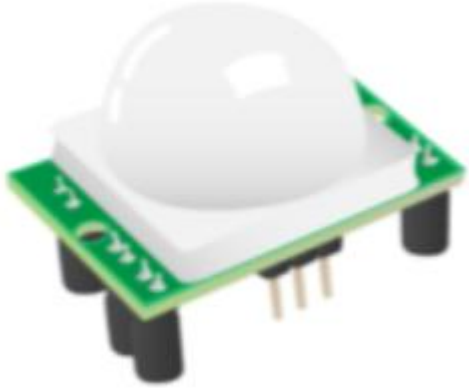
Visão Geral do Projeto: Alguma vez se perguntou quem entra sorrateiramente no seu quarto quando não está por perto? Neste projeto, irá construir um sistema de segurança inteligente — frequentemente chamado de "Detetor de Pais" — para descobrir exatamente quem esteve no seu quarto.

Conseguirás isto combinando um Raspberry Pi, um sensor de movimento e uma câmara para ativar e gravar automaticamente imagens de vídeo de quaisquer intrusos.

Passo 1: Reúne o Teu Hardware

Para construir esta câmara de segurança automatizada, será necessário três componentes principais de hardware:

- Um Raspberry Pi: Este atua como o cérebro do seu projeto, processando os sinais e executando o código.
- Um Sensor de Movimento Passivo por Infravermelhos (PIR): Este componente atua como os "olhos" do seu sistema, detetando alterações no calor infravermelho para detetar quando uma pessoa se aproxima.
- Um Módulo de Câmara Raspberry Pi: Este é o dispositivo que capta as provas de vídeo do intruso.



Passo 2: Compreender e ajustar o sensor PIR

Antes de ligar qualquer coisa com fios, precisa de configurar as definições físicas do próprio sensor PIR. Se olhar para a parte de trás do módulo PIR, verá dois componentes laranja com pequenos soquetes em forma de cruz que encaixam perfeitamente numa chave de fendas Philips.

Estes mostradores laranja chamam-se potenciómetros e permitem-lhe ajustar manualmente o comportamento do sensor:

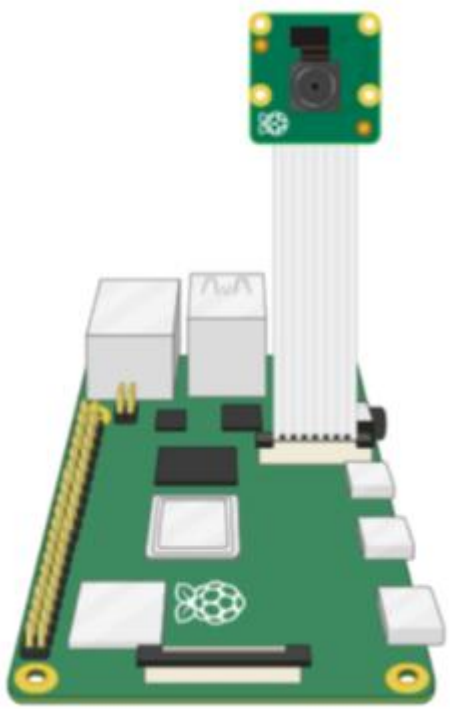
- Sensibilidade: Determina quanto movimento é necessário para ativar o sensor.
- Tempo (Atraso): Determina quanto tempo o sensor permanece "acionado" após detetar movimento antes de reiniciar.

Configuração Inicial: Para que este projeto funcione melhor, use a sua chave de fendas para definir o potenciômetro de sensibilidade ao máximo absoluto e o potenciômetro de tempo ao mínimo absoluto. Podes sempre ajustar e variar estas definições mais tarde se achares que o sensor está demasiado sensível ou fica ligado por muito tempo.

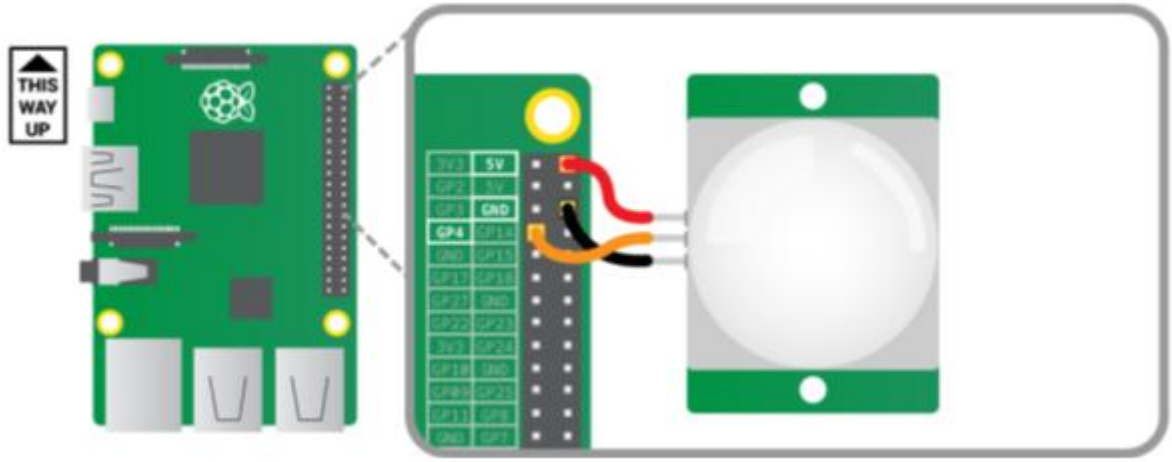
Passo 3: Ligue os componentes em segurança

Agora é altura de ligar o seu hardware ao Raspberry Pi. Regra Crucial: Certifique-se sempre de que o seu Raspberry Pi está completamente desligado e desligado antes de ligar o hardware.

1. Ligue a Câmara: Insira cuidadosamente o cabo de fita do Módulo de Câmara na porta dedicada da câmara do Raspberry Pi.



2. Ligue o Sensor PIR: O sensor PIR precisa de enviar os seus dados de movimento para o Raspberry Pi. Vais ligar o pino de saída de dados do sensor PIR diretamente ao pino rotulado GPIO 4 na tua placa Raspberry Pi. (Também terá de ligar os pinos de alimentação (VCC) e terra (GND) aos respetivos pinos de 5V e GND do Pi).



Passo 4: Escrever o Software de Segurança (Python)

Quando o seu hardware estiver ligado, ligue o seu Raspberry Pi. Abra o ambiente de programação chamado Thonny, crie um novo ficheiro e guarde-o imediatamente como parent_detector.py.

Para fazer o nosso hardware funcionar, precisamos de importar bibliotecas específicas. Vamos usar o MotionSensor da biblioteca gpiozero para gerir o sensor PIR e a classe Camera da biblioteca picamzero para controlar o Módulo Camera.

A Lógica do Código:

1. Monitorização Contínua: Usamos um ciclo while True:, que é um ciclo infinito que mantém o programa a verificar o movimento para sempre.

2. Espera por Ação: O programa usa pir.wait_for_motion() para pausar o código até que o sensor detete movimento. Assim que o movimento é detetado, imprime uma mensagem no ecrã.

3. Gravação: A câmara começa a captar vídeo usando a função cam.start_recording().

4. Parar: Não queremos gravar uma sala vazia para sempre. O código usa pir.wait_for_no_motion() para esperar que o intruso saia, e depois para o ficheiro de vídeo em segurança usando cam.stop_recording().

Resolver um Bug Crítico (O Problema de Sobrescrever): Se simplesmente dissermos à câmara para guardar o ficheiro como intruder.mp4, cada vez que uma nova pessoa entra na sala, o vídeo antigo será completamente sobrescrito e eliminado. Para garantir que mantemos um registo em vídeo de todos (pais ou irmãos irritantes), precisamos de um nome de ficheiro dinâmico. Podemos usar a biblioteca de tempo do Python para descobrir automaticamente a data e hora atuais e injetá-las no nome do ficheiro do vídeo.

Comece com este código e as suas ferramentas de IA favoritas para testar este código:

Escreve isto no teu ficheiro de parent_detector.py:

```
1. a partir da importação gpiozero do MotionSensor
```

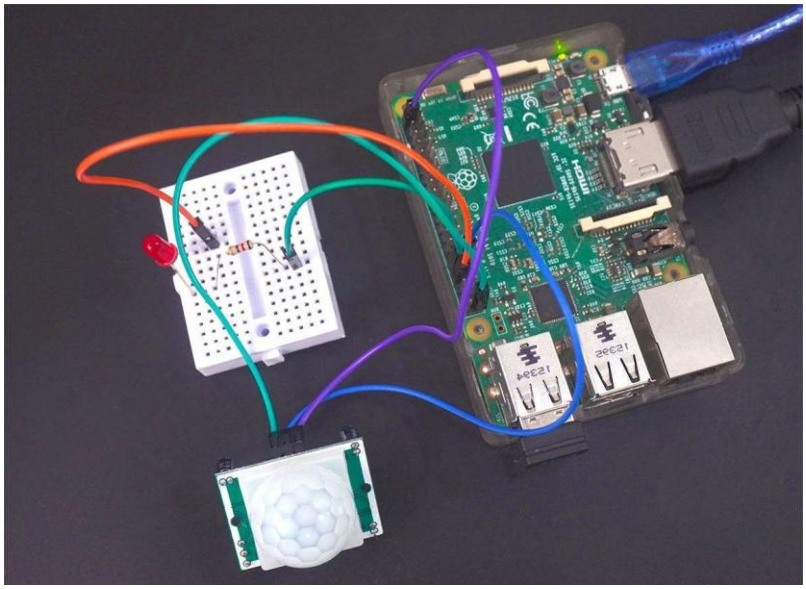
```
da câmara importada picamzero
Tempo de importação

# 1. Inicializar Hardware
# Configurar o sensor PIR na GPIO 4
pir = MotionSensor(4)

# Criar um objeto Câmara para controlar o módulo
cam = Câmara()

imprimir ("Sistema de Segurança Armado. À espera de intrusos...")

# 2. Circuito Principal de Segurança
embora Verdadeiro:
    # O programa pausa aqui até ser detetado movimento
    pir.wait_for_motion()
    print("AVISO: Movimento detetado!")

    # 3. Gerar um Nome de Ficheiro Único
    # Isto cria uma cadeia como "20231025-143000" (AnoMêsDia-HoraMinutoSegundo)
    carimbo temporal = hora.strftime("%Y%m%d-%H%M%S")
    nome do ficheiro = f"intruder_{timestamp}.mp4"

    # Começar a gravar provas em vídeo no novo ficheiro
print(f"A gravar vídeo: {nome do ficheiro}")
cam.start_recording(nome do ficheiro)

    # 4. Esperar que a sala volte a ficar vazia
    pir.wait_for_no_motion()
imprimir ("Movimento parado.")

    # Parar a gravação para finalizar e guardar o ficheiro de vídeo em segurança
    cam.stop_recording()
    print("Sistema a regressar ao modo de espera.")
```

Agora que o seu código está totalmente escrito e o seu hardware está ligado de forma segura, é altura de testar se o detetor parental funciona realmente no mundo real! Primeiro, clique no botão Executar dentro do seu ambiente de programação Thonny para iniciar o script. Quando o programa estiver a funcionar e à espera, acene deliberadamente com a mão diretamente à frente do detetor de movimento PIR para simular um intruso a entrar na sala. Olhe imediatamente para o ecrã do seu computador; deve ver as palavras "Movimento detetado!" impressas na área da consola, confirmando que o sensor conseguiu ativar a câmara para iniciar a gravação.

Depois de acionar o alarme, é necessário testar o mecanismo de desligamento. Saia do campo de visão do sensor, mantenha-se completamente imóvel ou cubra suavemente a cúpula branca do sensor com a mão. Espere alguns segundos até que a consola lhe indique que o movimento parou e que o sistema está a regressar ao modo de espera. Isto indica que o programa finalizou e fechou o ficheiro de vídeo em segurança. Finalmente, abre o gestor de ficheiros do teu Raspberry Pi e navega até à mesma pasta onde guardaste o teu *script de parent_detector.py* . Agora deve ver um ficheiro de vídeo .mp4 novo à sua espera, com a data e hora únicas no

nome, tal como programámos. Clique duas vezes neste ficheiro recém-criado para ver as provas gravadas, verificar a qualidade do vídeo e garantir que o ângulo da câmara capta perfeitamente a porta!

Fotos adicionais:



Fonte: https://www.electronicwings.com/raspberry-pi/pir-motion-sensor-interfacing-with-raspberry-pi

Fonte: https://thepihut.com/products/pir-camera-case-for-raspberry-pi-4-3

16.2 Reconhecimento de Estimativa de Postura YOLO

Neste guia, vamos configurar algumas com o OpenCV e a família de modelos de estimativa de pose YOLO no Raspberry Pi 5. Vamos analisar alguns dos diferentes modelos YOLO disponíveis, bem como otimizá-los para obter FPS mais suaves, e também como usar os dados de keypointpoints gerados pelo modelo para implementar a estimativa de pose no seu próximo projeto. Este tem sido um dos guias mais divertidos que fizemos nos últimos tempos, por isso vamos a isso!

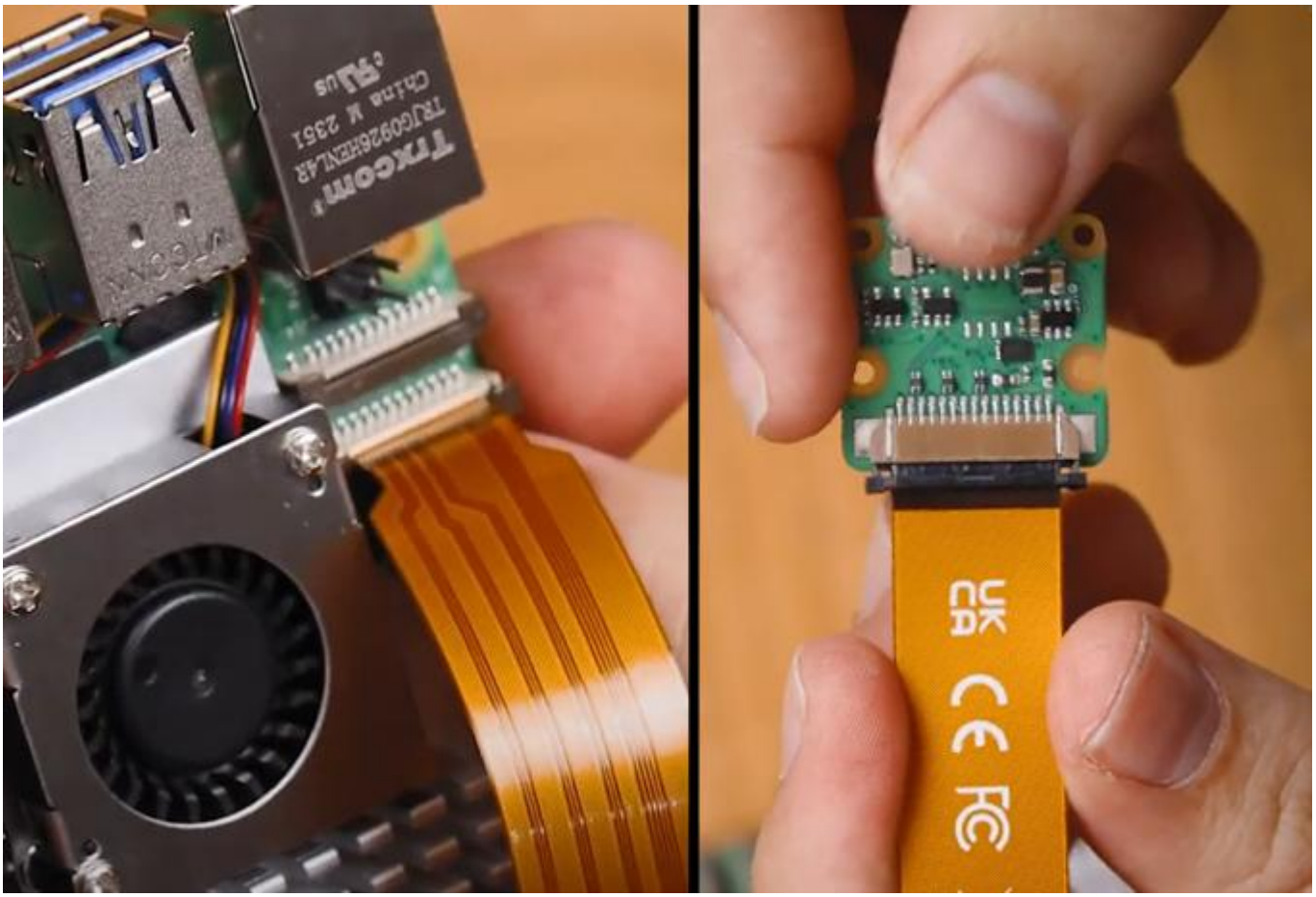
Para acompanhar este guia, vai precisar de:

- Raspberry Pi 5 - Um modelo de 4GB ou 8GB funciona aqui. Embora tecnicamente isto pudesse ser feito num Pi 4, é muito mais lento do que o Pi 5 e não seria uma experiência agradável, e por essas razões, não testámos num Pi 4
- Pi Camera - Estamos a usar o Módulo de Câmara V3
- Cabo Adaptador - O Pi 5 vem com um cabo de câmara CSI de tamanho diferente e a sua câmara pode vir com o mais antigo e mais grosso, por isso vale a pena confirmar. O Módulo da Câmara V3 VAI precisar de um
- Solução de Arrefecimento - Estamos a usar o cooler ativo (a visão por computador vai realmente levar o teu Pi ao limite)
- Fonte de Alimentação
- Cartão Micro SD - Pelo menos 16GB de tamanho
- Monitor e Cabo Micro-HDMI para HDMI
- Rato e Teclado

Montagem de Hardware

Em termos de montagem de hardware, aqui é bastante leve. Liga o lado mais grosso do cabo à câmara e o lado mais fino ao Pi 5. Estes conectores têm uma aba – levanta-os e depois insere o cabo na ranhura. Quando estiver bem alinhado, empurra a aba para baixo para prender o cabo no lugar.

Fica atento, pois estes conectores só funcionam numa orientação e podem ser frágeis, por isso evita dobrá-los muito (um pouco é aceitável).



Instalação do Pi OS

Antes de mais, precisamos de instalar o Pi OS no cartão micro SD. Usando o **Raspberry Pi Imager**, selecione o Raspberry PI 5 como Dispositivo, o Raspberry Pi OS (64 bits) como sistema operativo e o seu cartão microSD como dispositivo de armazenamento.

O mesmo procedimento do PiRacer.

Configuração de um Ambiente Virtual e Instalação de Bibliotecas

Com a introdução do Bookworm OS em 2023, somos agora obrigados a usar Ambientes Virtuais (ou venv), pois são um espaço isolado no Pi onde podemos experimentar sem o risco de prejudicar o resto do nosso Pi OS ou projetos. Temos todos os comandos e instruções necessários neste guia.

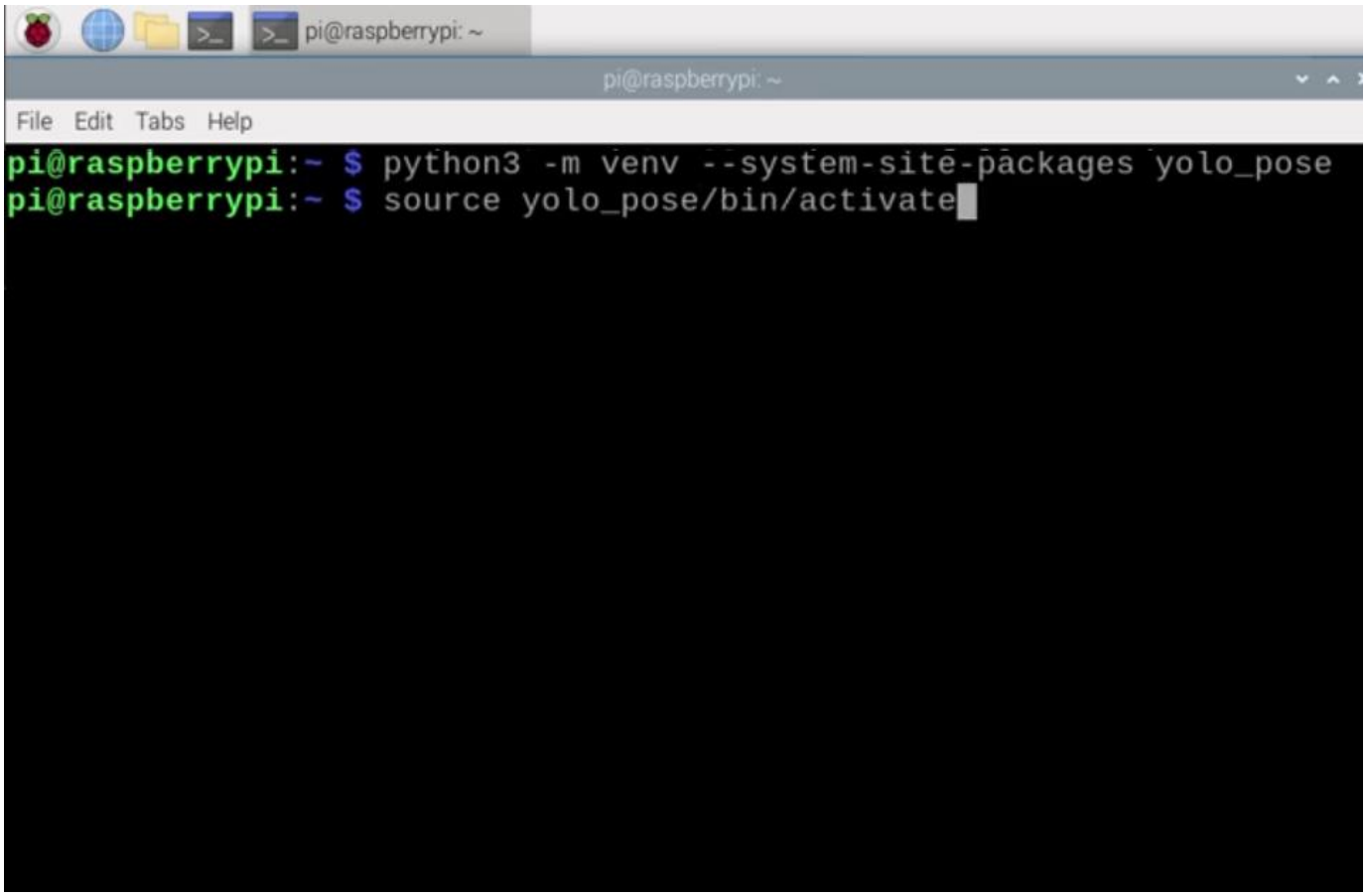
Para criar um ambiente virtual, abra uma nova janela de terminal e digite:

python3 -m venv --system-site-packages yolo_pose

Depois de criar o venv, podemos entrar nele escrevendo:

fonte yolo_pose/bin/ativar

Depois disso, verá o nome no ambiente virtual à esquerda do texto verde – isto significa que estamos a trabalhar corretamente dentro dele. Se alguma vez precisares de voltar a entrar neste ambiente (por exemplo, se fechares a janela do terminal vais sair do ambiente), basta escreveres novamente o comando de origem acima.



Agora que trabalhamos num ambiente virtual, podemos começar a instalar os pacotes necessários. Primeiro, certifique-se de que o PIP (o gestor de pacotes Python) está atualizado, introduzindo as seguintes três linhas:

Atualização Sudo Apt

sudo apt install python3-pip -y

pip install -U pip

Depois instala o Pacote Ultralytics com:

Instalar Ultralytics[exportar]

O pessoal simpático da Ultralytics tem sido um dos principais desenvolvedores e mantenedores dos modelos YOLO mais recentes. Este pacote deles vai fazer grande parte do trabalho pesado e vai instalar o OpenCV, bem como toda a infraestrutura necessária para correrm os YOLO.

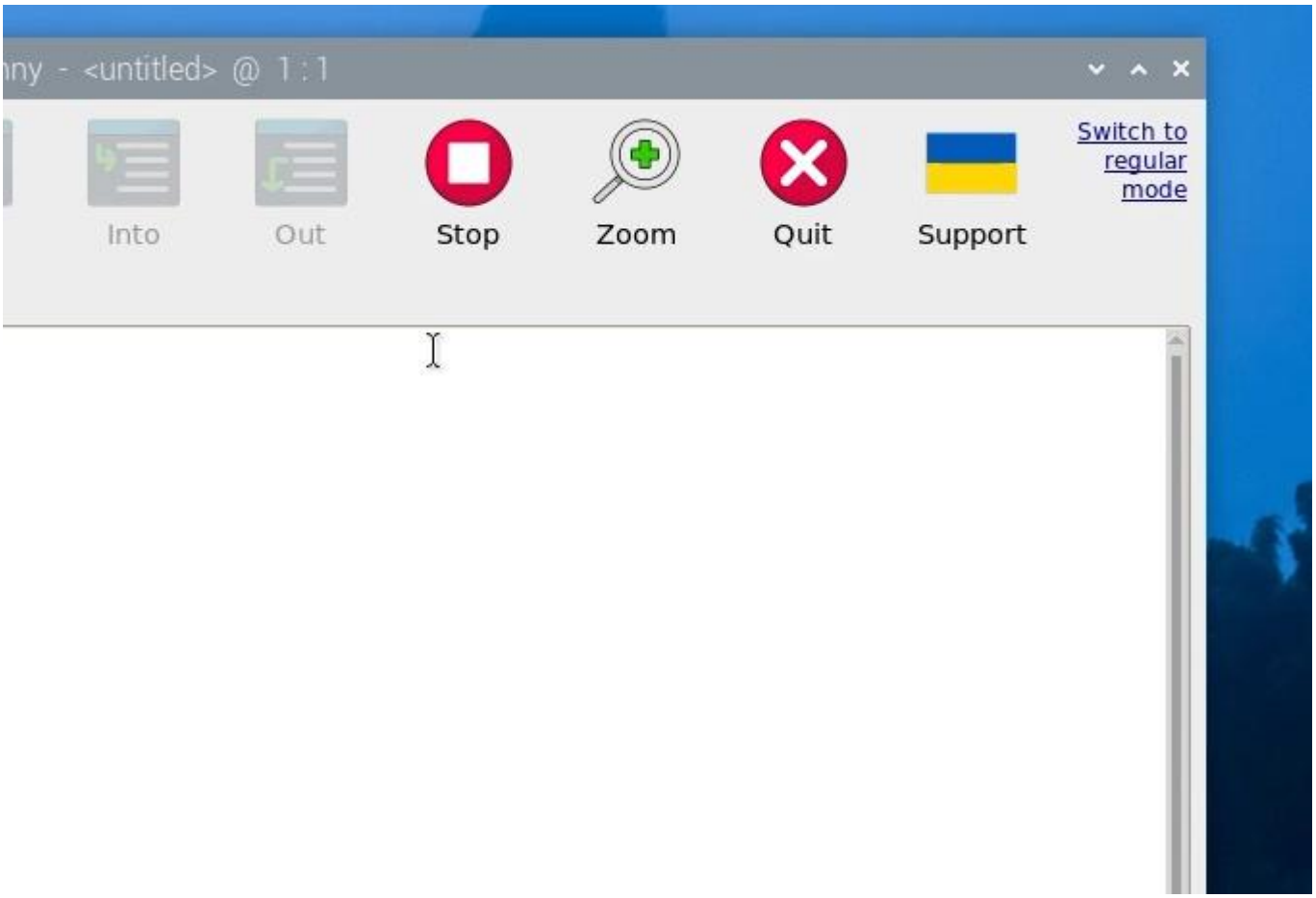
Este processo também instala uma grande quantidade de outros pacotes e, como resultado, é propenso a falhas. Se a tua instalação falhar (vai mostrar uma parede inteira de texto vermelho), basta digitar novamente a linha de instalação do Ultralytics e deverá recomeçar. Em casos raros, pode ser necessário repetir a linha de instalação algumas vezes.

Depois de terminar a instalação, reinicie o Raspberry Pi. Se quiseres ser um utilizador avançado, podes fazê-lo escrevendo no shell:

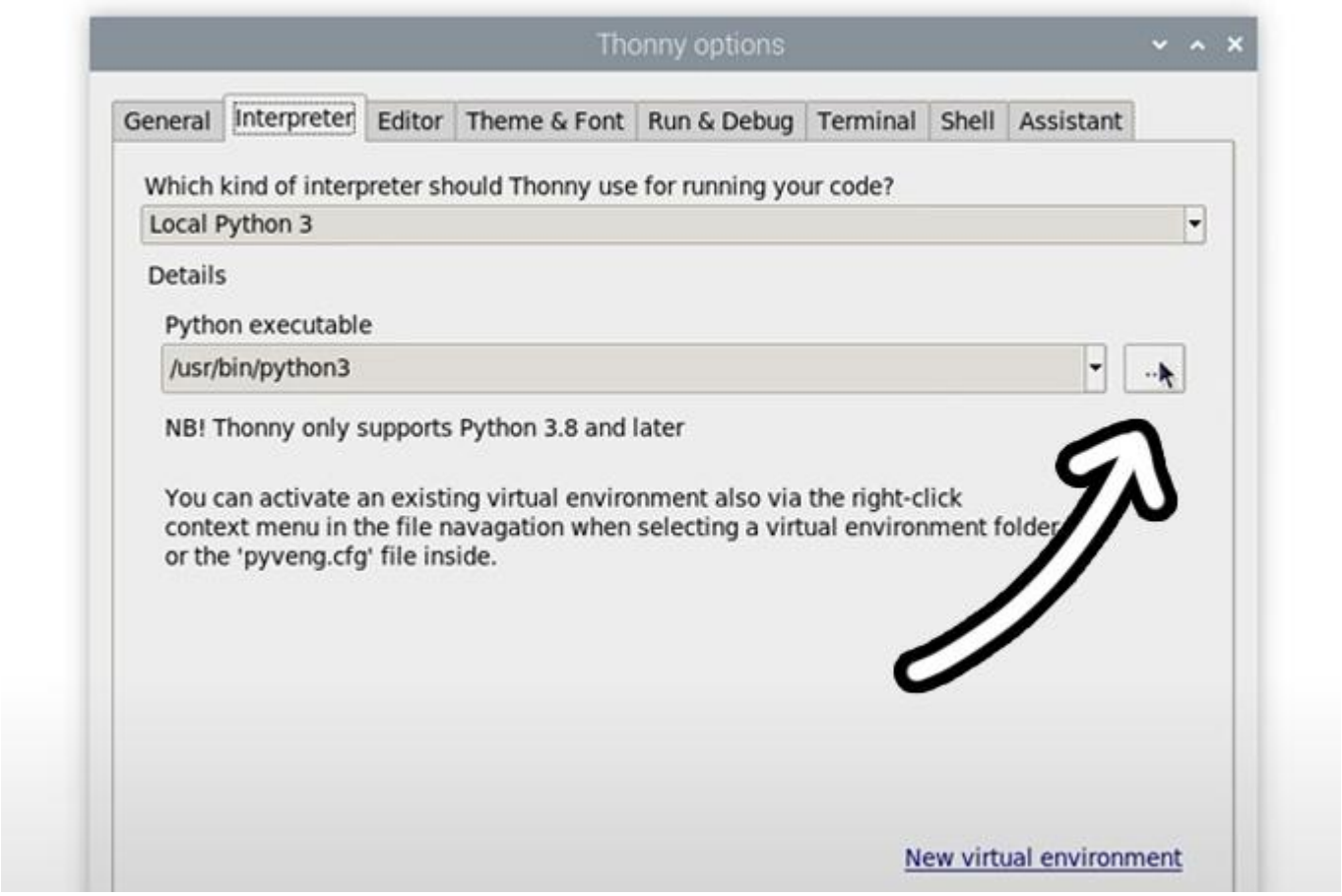
Reinício

Temos mais uma coisa a fazer, que é preparar o Thonny para usar o ambiente virtual que acabámos de criar. O Thonny é o programa do qual vamos correr todo o nosso código e precisamos de o fazer funcionar com o mesmo VENV para que tenha acesso às bibliotecas que instalámos.

Da primeira vez que abrir o Thonny pode estar no modo simplificado, e verá um "mudar para modo normal" no canto superior direito. Se isto estiver presente, clica e reinicia o Thonny fechando-o.



Agora entre no menu de opções do interpretador selecionando Executar > Configurar Interpretador. Na opção executável em Python, há um botão com 3 pontos. Seleciona-o e navega até ao executável Python no ambiente virtual que acabámos de criar.



Isto estará localizado em home/pi/yolo_pose/bin e, neste ficheiro, terá de selecionar o ficheiro chamado "python3". Carrega em ok e agora vais trabalhar neste venv.

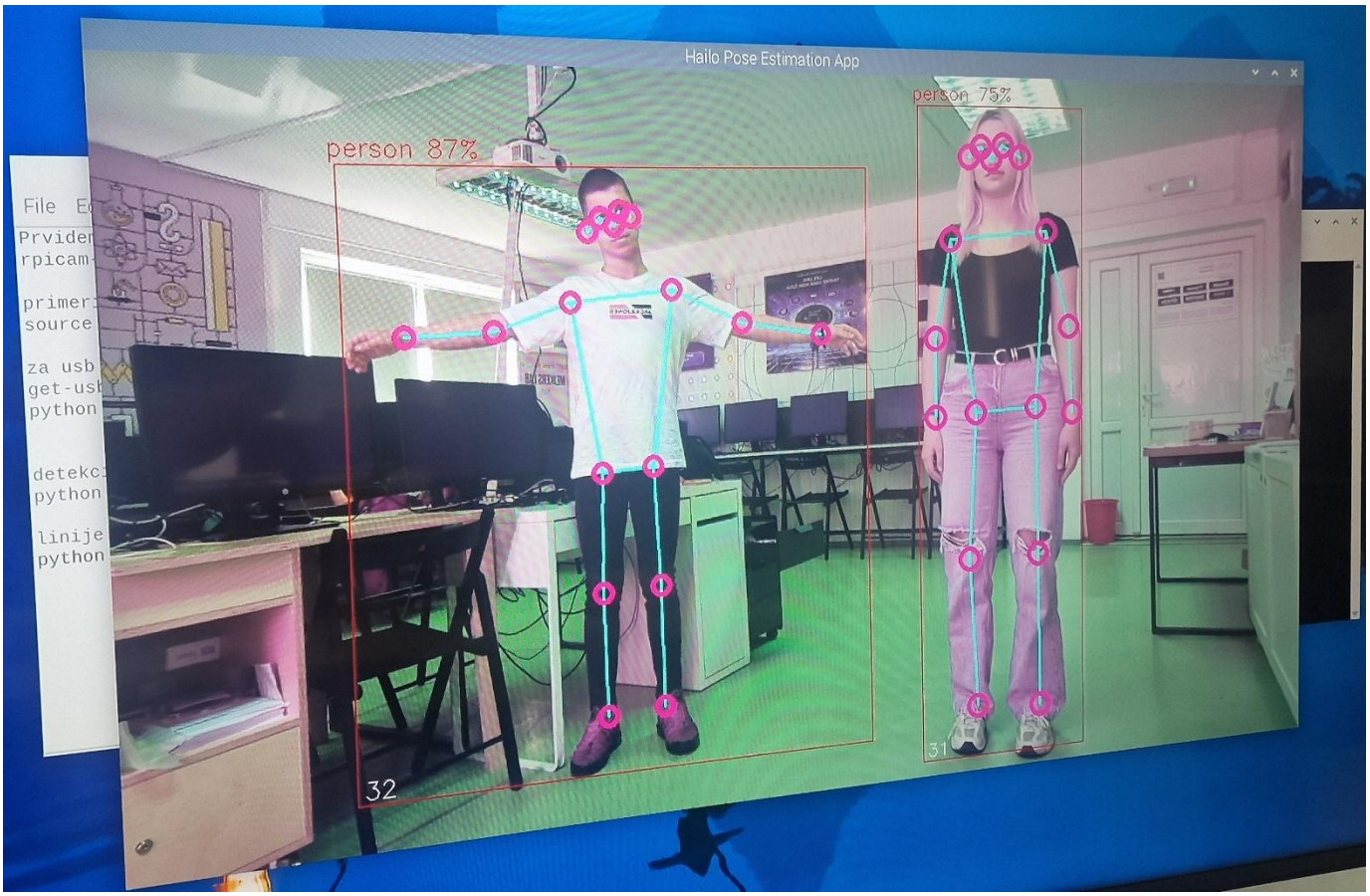
Sempre que abrir o Thonny, ele funciona automaticamente a partir deste ambiente. Podes mudar o ambiente onde estás a trabalhar selecionando-o no menu suspenso sob o executável Python no mesmo menu de opções do interpretador. Se quiser sair do ambiente virtual, selecione a opção bin/python3.

Estimativa de Postura Corrente

Agora que temos as nossas bibliotecas instaladas e o Thonny está a trabalhar no ambiente virtual, podemos executar o nosso script de estimativa de pose. Extrai a **pasta zip do projeto** (descarrega do nosso servidor) para um local conveniente, como o ambiente de trabalho. Lá dentro encontrará o primeiro guião que vamos usar*, "pose demo.py".* Abre-o, Thonny, e carrega no grande botão verde de corrida. Na primeira vez que executares isto, pode instalar algumas coisas extra necessárias (todas automaticamente), e após alguns segundos deves ver uma janela de pré-visualização com a estimativa da tua pose a correr.

Algumas coisas deviam estar a acontecer aqui. Primeiro, o YOLO tentará detetar humanos e, se reconhecer um, desenhará uma caixa à sua volta com a classificação de confiança no topo. O importante é que ele colocará pontos onde considera que estão alguns pontos essenciais do seu corpo (estes são chamados pontos-chave), e irá traçar linhas entre esses pontos para estimar a pose e a orientação da pessoa. No canto superior direito estará também os FPS a que isto está a correr (que vamos melhorar daqui a pouco).

E é isso! Com estes poucos passos, já temos a nossa estimativa da pose de corrida do Pi!

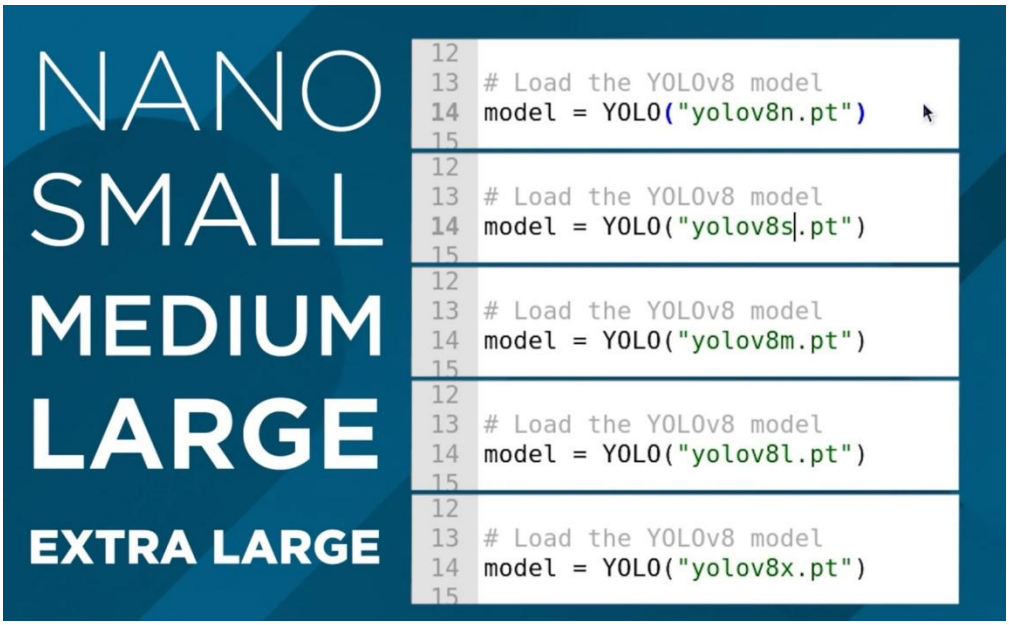


Mudanças nos Modelos YOLO

Até agora temos corrido o YOLO11, e uma das vantagens deste pacote Ultralytics é que podemos simplesmente trocar uma única linha no código para mudar completamente o modelo. Podemos usar isto para correr um modelo YOLO11 mais avançado, ou até um modelo mais antigo. Tudo o que precisas de mudar é esta frase aqui na configuração:

```
# Carregue o nosso modelo YOLO11
modelo = YOLO("yolo11n-pose.pt")
```

Esta linha está atualmente a usar o modelo nano, que é o modelo mais pequeno, menos potente, mas mais rápido do YOLO11, e podemos alterar esta linha para usar um dos diferentes tamanhos em que este modelo existe, alterando a única letra após "11", como mostrado à direita. Se mudares esta linha para outro tamanho de modelo e a executares, o script descarrega automaticamente o novo modelo (que pode estar na casa das centenas de Mb para os modelos maiores).



A diferença entre estes modelos é um compromisso entre desempenho na estimativa de pose e FPS. Quanto maior o modelo, melhor será a estimar as partes do corpo que podem não ser vistas pela câmara, bem como ângulos mais complexos e frames com mais pessoas, no entanto, pode esperar que só seja processado 1 frame a cada 10 segundos! Vamos aumentar este valor no próximo passo.

O nanomodelo, por outro lado, é o que corre mais rápido, conseguindo cerca de 1,5 FPS sem otimização, mas não tem o poder de processamento dos modelos maiores. Para a estimação da pose, podes usar o nanomodelo na maior parte do tempo, pois normalmente é suficientemente bom para as tuas necessidades, mas se precisares de algo um pouco mais potente, continua a aumentar o tamanho do modelo para se adequar às tuas necessidades.

Nesta linha, também podemos alterar a versão do YOLO a correr. Podes voltar a um modelo mais antigo se quiseres, ou podes usar um modelo mais recente. Este guia acabará por ficar desatualizado e, se a Ultralytics lançar o YOLO13, deverá simplesmente conseguir alterar a linha para a seguinte para começar a usar a versão mais recente do YOLO:

```
# Carregue o nosso modelo YOLO11
modelo = YOLO("yolo13n-pose.pt")
```

Aumento da Velocidade de Processamento

Há duas coisas que podemos fazer para aumentar o FPS no Pi e a forma mais eficaz é converter o modelo para um formato chamado NCNN. Este é um formato de modelo mais otimizado para funcionar em processadores baseados em ARM como os Raspberry Pi. Abra o script chamado "ncnn conversion.py" e encontrará o seguinte:

```
da importação de ultralytics  YOLO

# Carregar um modelo YOLO11n PyTorch
modelo = YOLO("yolo11n-pose.pt")

# Exportar o modelo para o formato NCNN
modelo.export(format="ncnn", imgsz=640) # cria 'yolov11n-pose_ncnn_model'
```

Para usar este script, especifique primeiro o modelo que pretende converter. Isto usa as mesmas convenções de nomenclatura que discutimos na secção anterior. Depois, o formato do modelo "ncnn" é especificado como formato de saída, assim como a resolução. Por agora, mantém isto no padrão de 640. Na primeira vez que executares este script, ele vai descarregar mais algumas coisas adicionais de que precisa, mas deverá demorar apenas alguns segundos a executar a conversão propriamente dita.

Quando isso estiver concluído, na pasta onde os scripts estão vais encontrar uma nova pasta chamada algo como "yolo11n-pose_ncnn_model". Copie o nome deste ficheiro e volte ao nosso script de demonstração anterior.

Agora tens de dizer ao script para usar este modelo que criámos, mudando a linha do modelo para o nome da pasta que acabou de criar. Deveria parecer algo assim:

```
# Carregue o nosso modelo YOLO11
modelo = YOLO("yolo11n-pose_ncnn_model")
```