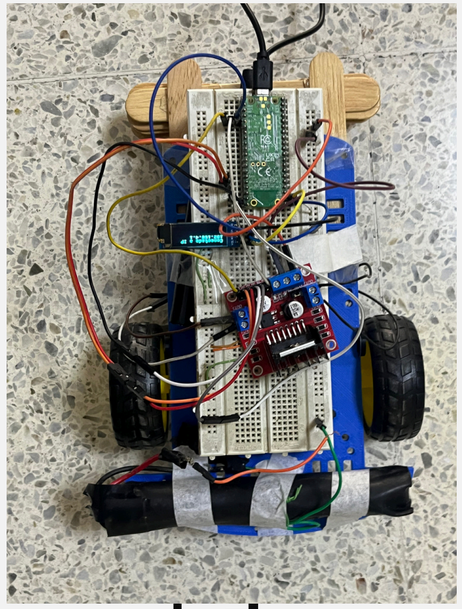
Line-Following Car with Raspberry Pi Pico W and OV7670 Camera

Julián Pacheco
Juan Ramírez
Julian Rodríguez

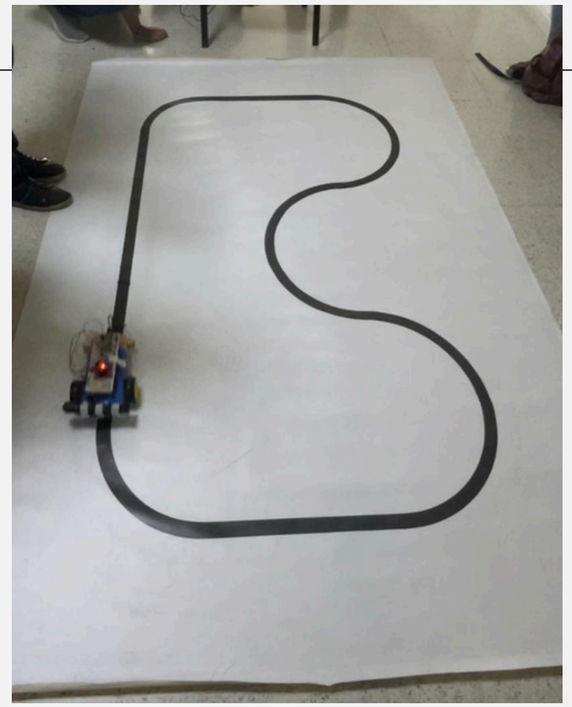


LAYERS
NORMAL
OPACITY 100%
LOCK:
FILL 100%
LAYER 1 COPY
TEXT
LAYER 1
SMART OBJECT

Line-Following Car

Page 1 | Words: 88

Our project is an autonomous car that follows a line on the ground using a Raspberry Pi Pico W and an OV7670 camera. In this presentation, we will explain how we built and programmed it.



Line-Following Car

- Structure: Acrylic base with support brackets.
- Mounted components: Raspberry Pi Pico W, OV7670 camera, battery, H-bridge, and DC motors.
- Fastening: Use of screws, nuts, and brackets.
- Electrical connection: Proper polarization to avoid damage.

Document2 - Macrostuff Board

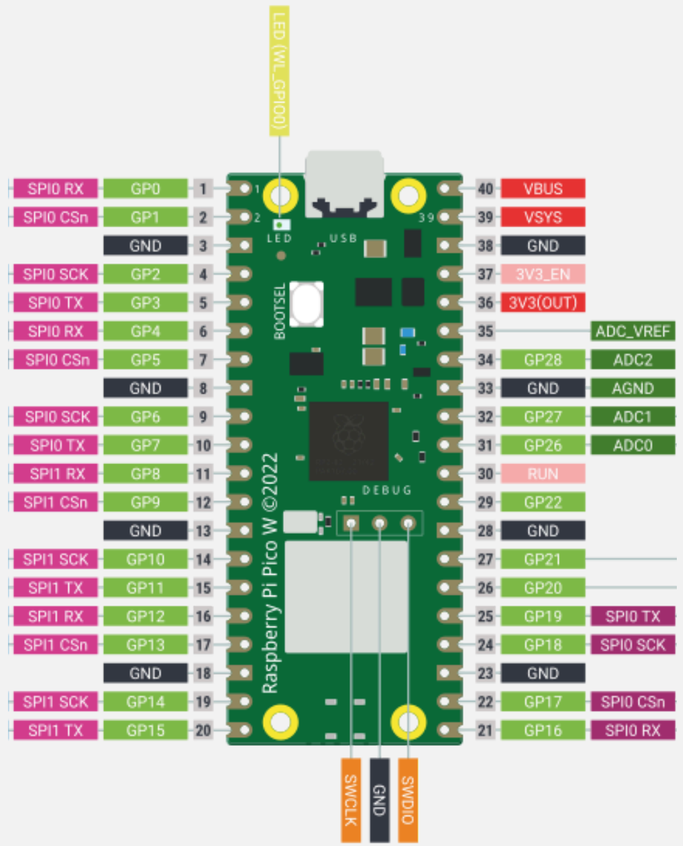
File Edit Format View Help

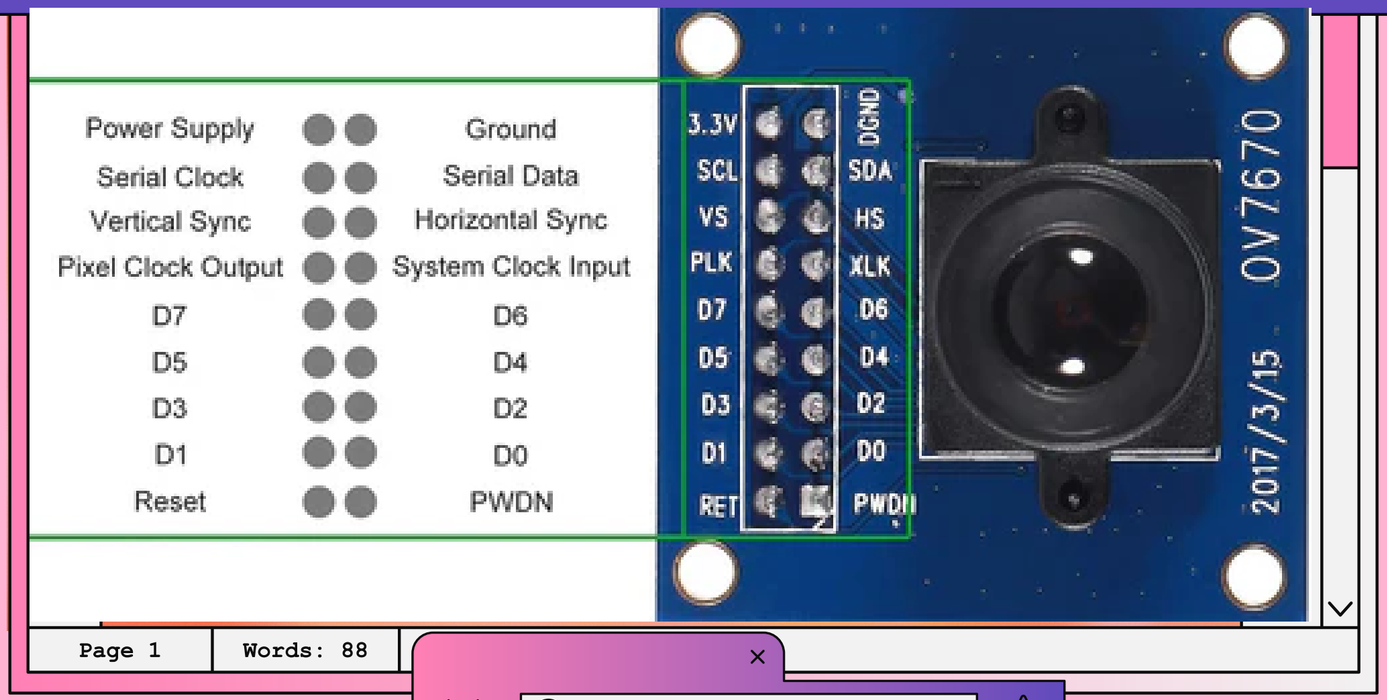
Cool New Font 12

Page 1 Words: 88

Software Configuration

```python
import sys
import time
import pwmio
import digitalio
import busio
import board

vel_ini_derecha=55;
vel_ini_izquierda=55;
#uart = busio.UART(board.GP16, board.GP17, baudrate=9600)
led = digitalio.DigitalInOut(board.LED)
led.direction = digitalio.Direction.OUTPUT
pwm_derecha = pwmio.PWMOut(board.GP18,frequency=100000, duty_cycle=0)
pwm_izquierda = pwmio.PWMOut(board.GP19,frequency=100000, duty_cycle=0)

pwm_derecha.duty_cycle = 0
pwm_izquierda.duty_cycle = 0

pin_motor_derecha=board.GP16
pin_motor_izquierda=board.GP17
pin_ledblanco=board.GP22
# Configurar el pin como salida
motor_d = digitalio.DigitalInOut(pin_motor_derecha)
motor_d.direction = digitalio.Direction.OUTPUT

# Establecer el pin en alto
motor_d.value = True

# Configurar el pin como salida
motor_i = digitalio.DigitalInOut(pin_motor_izquierda)
motor_i.direction = digitalio.Direction.OUTPUT

```





- Camera configuration.
- I2C communication.
- Converting the image to YUV format.

```
from adafruit_ov7670 import (  # pylint: disable=unuse
    OV7670,
    OV7670_SIZE_DIV16,
    OV7670_COLOR_YUV,
    OV7670_TEST_PATTERN_COLOR_BAR_FADE,
)


cam_bus = busio.I2C(board.GP21, board.GP20)

cam = OV7670(
    cam_bus,
    data_pins=[
        board.GP0,
        board.GP1,
        board.GP2,
        board.GP3,
        board.GP4,
        board.GP5,
        board.GP6,
        board.GP7,
    ],
    clock=board.GP9,
    vsync=board.GP13,
    href=board.GP12,
    mclk=board.GP8,
    shutdown=board.GP15,
    reset=board.GP14,

)

cam.size = OV7670_SIZE_DIV16
cam.colorspace = OV7670_COLOR_YUV
cam.flip_y = False
```

- Analyzing the image to detect the black line.
- Converting the image into a matrix of values.
- Calculating the position of the line.

Document2 - Macrostuff Board

File Edit Format View Help

Cool New Font 12 B I U

```
while True:

    cam.capture(buf)

    matrix = []
    rw = []
    for i in range(cam.width):
        intensity = "0" if buf[2 *
        (width * (cam.height-1) + i)] * 10 // 255 < 5 else "-"
        rw.append(intensity)

    matrix.append(rw)

    ulinea=matrix[len(matrix)-1][:]
    print(len(ulinea))
    suma=0
    contador=0
```

Page 1 Words: 88

Image Processing and Motion Control

```
    for i in range(len(ulinea)):
        suma += i if ulinea[i]=="0" else 0
        contador += 1 if ulinea[i]=="0" else 0
    promedio=suma/contador if contador!=0 else 0

    diferencia=(20-promedio)*(100/20) if promedio!=0 else 100
    if promedio != 0 :
        if diferencia<0:
            print(abs(diferencia/100))
            pwm_derecha.duty_cycle = int(((vel_ini_derecha/100)*65535) * (1
            pwm_izquierda.duty_cycle = int((vel_ini_izquierda/100)*65535)
        else:
            pwm_izquierda.duty_cycle = int(((vel_ini_izquierda/100)*65535)
            pwm_derecha.duty_cycle = int((vel_ini_derecha/100)*65535)
    else:
        pwm_derecha.duty_cycle = 0
        pwm_izquierda.duty_cycle = 0



    time.sleep(0.08)

    pwm_derecha.duty_cycle = 1000
    pwm_izquierda.duty_cycle = 1000
```

100% 1

We successfully built and programmed a line-following car that adjusts its direction based on real-time image processing. We learned about image processing, motor control, and sensor communication in CircuitPython.