Documentație proiect SI

2024

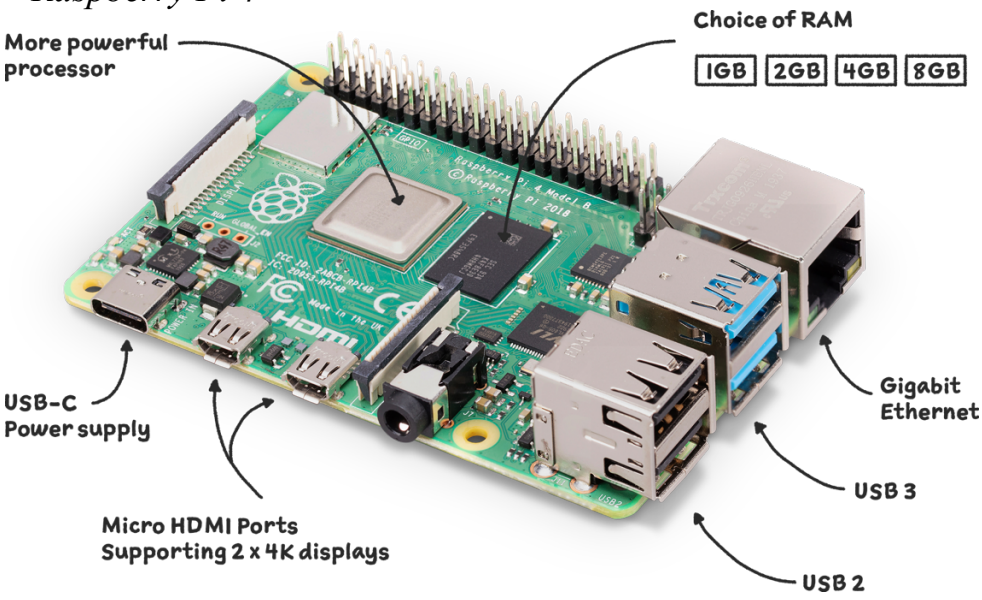
-Sistem automat de procesare de imagini pentru recunoașterea video a semnelor de circulație-

Burduniuc Iulian
Bran Diana-Georgiana
Anița Iulia-Andrada

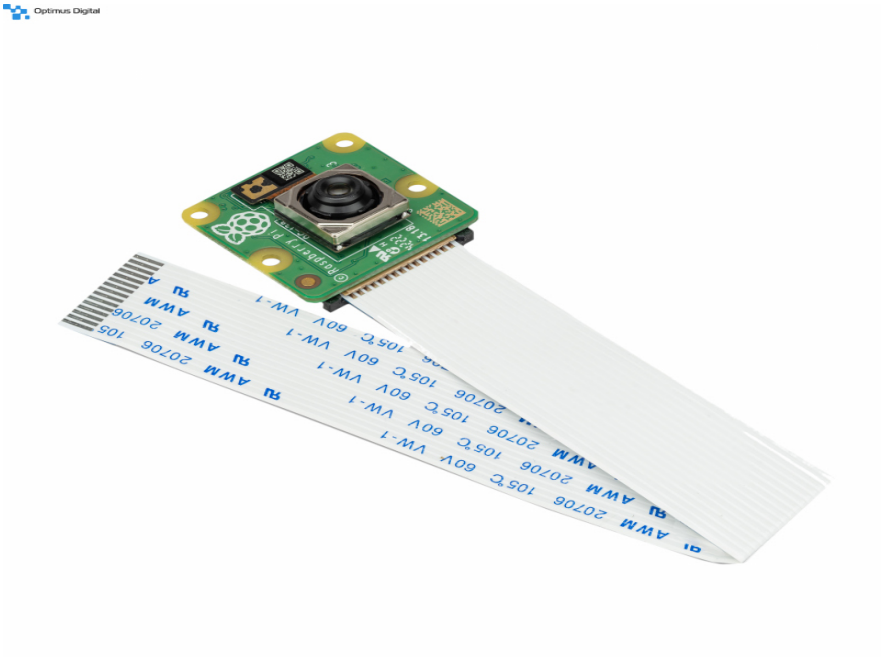
În cadrul acestui proiect, am dezvoltat un sistem automat de procesare a imaginilor utilizând un Raspberry Pi 4 pentru recunoașterea video a semnelor de circulație. Proiectul a inclus mai multe etape esențiale, de la configurarea hardware-ului și instalarea pachetelor software necesare, până la dezvoltarea și implementarea codului pentru procesarea și recunoașterea semnelor de circulație.

1.Componente Utilizate

- *Raspberry Pi 4*



- *Modul Camera 3*

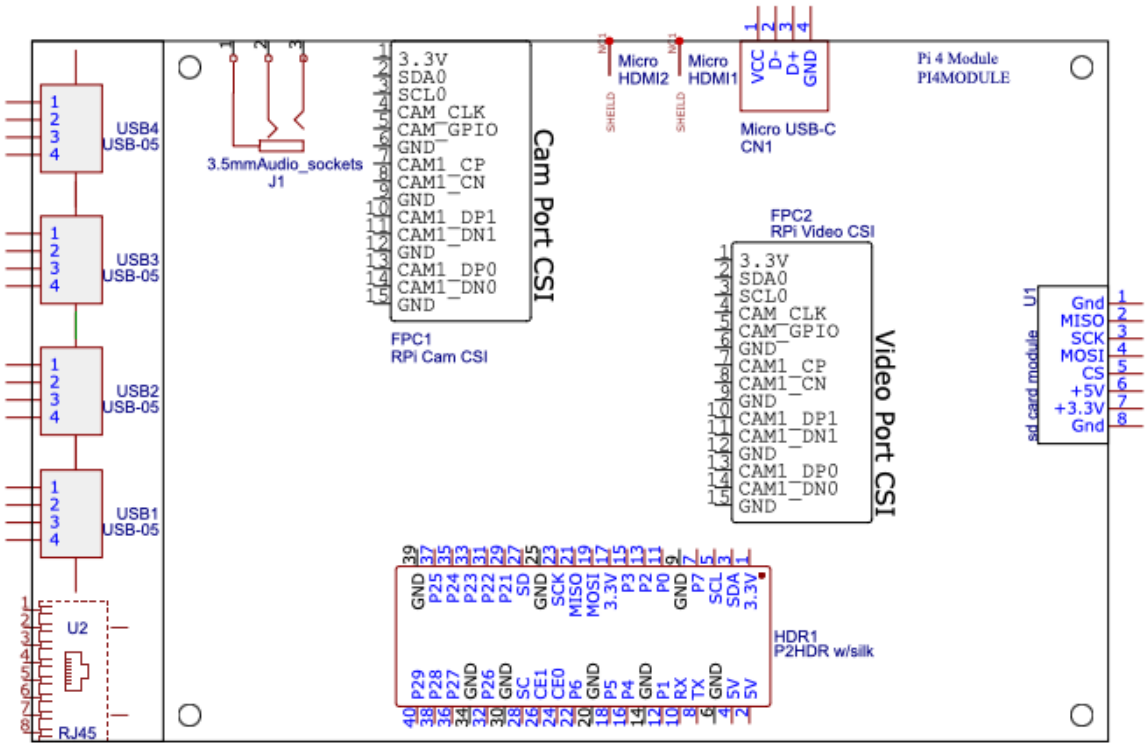


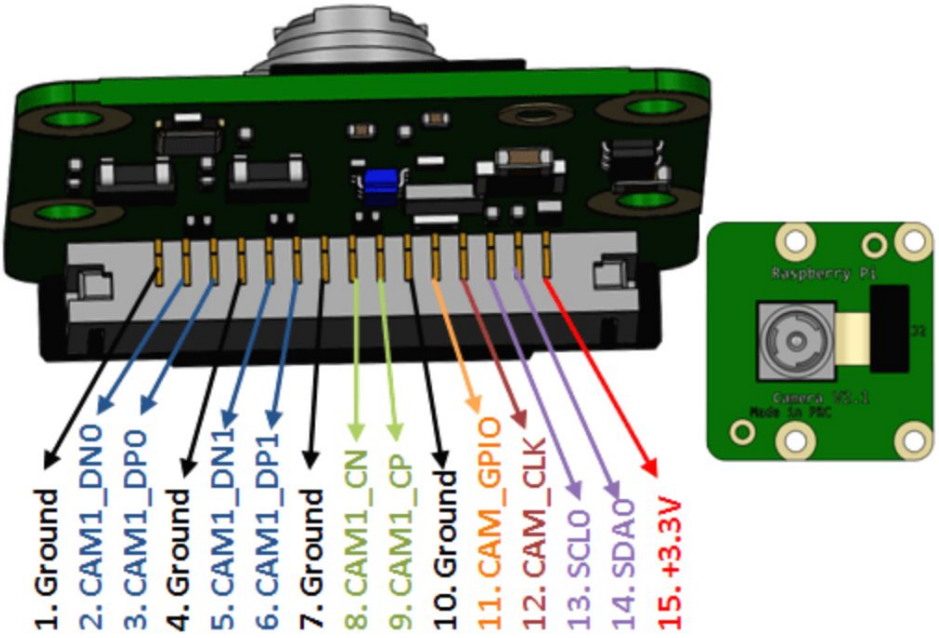
- *Tastatura*
- *Mouse*
- *Monitor*

Componente Utilizate:

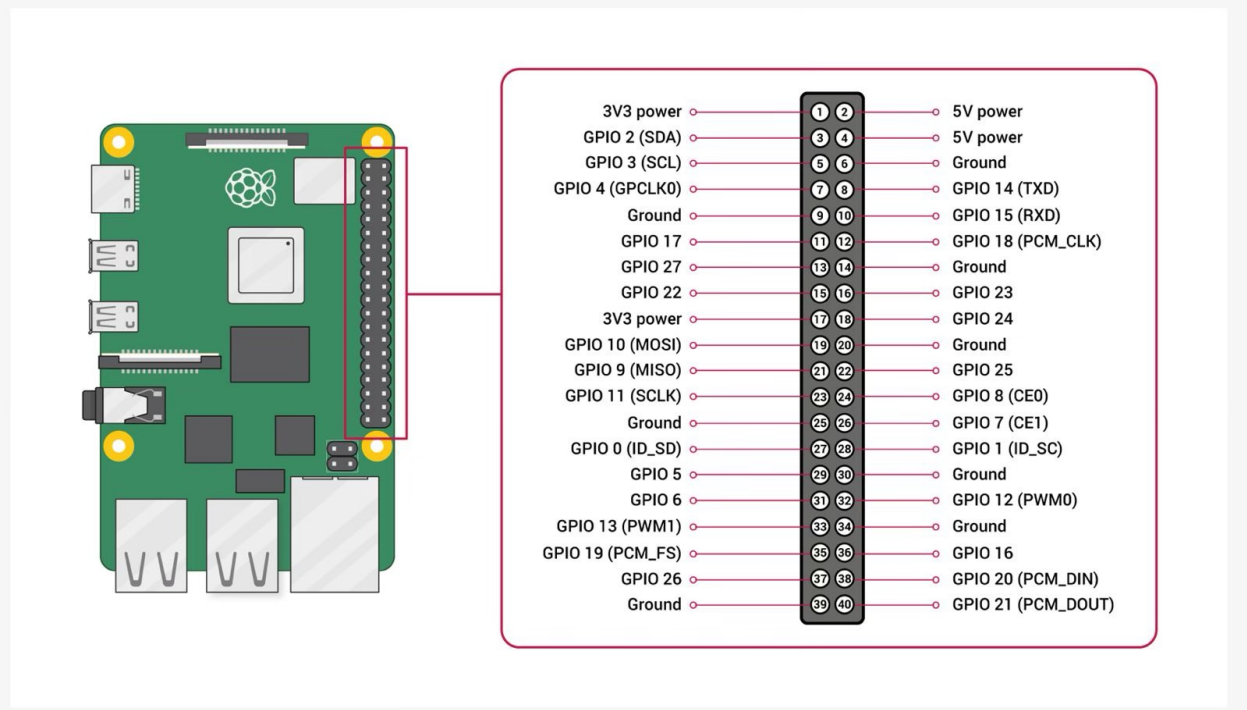
- **Raspberry Pi 4**: Un microcomputer puternic și versatil, ideal pentru proiecte de procesare a imaginilor.
- **Modul Camera 3**: Utilizat pentru capturarea imaginilor necesare procesării.
- **Periferice**: Tastatură, mouse și monitor pentru configurarea inițială și monitorizarea sistemului.

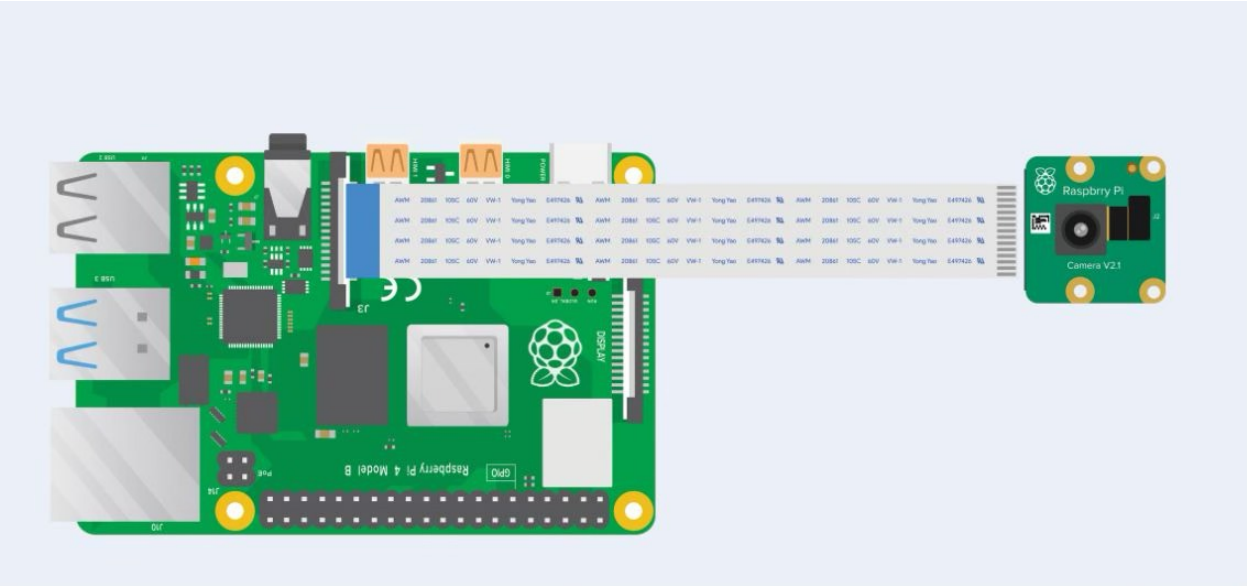
2.Shema Hardware





3.Explicare pin:





4.Configurare:

Pregătirea Raspberry Pi 4:

1. Introduceți cardul microSD în Raspberry Pi 4 și conectați sursa de alimentare.
2. Conectați monitorul, tastatura și mouse-ul pentru configurarea inițială.
3. Porniți Raspberry Pi 4 și urmați pașii de configurare pe ecran.
4. Asigurați-vă că Raspberry Pi 4 este conectat la internet.

Instalarea Pachetelor Software Necesare:

1. **Actualizați sistemul de operare:**

sudo apt update

sudo apt upgrade

2. **Instalați pachetele necesare pentru procesarea imaginii și machine learning:**

sudo apt install python3-opencv python3-pip

pip3 install numpy tensorflow keras

3. **Conectarea Camerei Video:**

1. Conectați camera video la portul dedicat de pe Raspberry Pi 4.
2. Activați camera din meniul de configurare:

sudo raspi-config si selectați Interface Options > Camera > Enable.

4. **Reporniți Raspberry Pi 4 pentru a aplica modificările:**

sudo reboot

5.Dataset

Sursa: Hugging Face.

Link: https://www.kaggle.com/datasets/tuanai/traffic-signs-dataset/data

Descriere:

- 6000 de imagini

- 50 clase

1	Speed limit (15km/h)
2	Speed limit (30km/h)
3	Speed limit (40km/h)
4	Speed limit (50km/h)
5	Speed limit (60km/h)
6	Speed limit (70km/h)
7	speed limit (80km/h)
8	Dont Go straight or left
9	Unknown7
10	Dont Go straight
11	Dont Go Left
12	Dont Go Left or Right
13	Dont Go Right
14	Dont overtake from Left
15	No Uturn
16	No Car
17	No horn

6.Code

```python
import cv2
from libcamera import libcamera, libcamera_core
from libcamera import libcamera_core as core
from libcamera import controls

# Function to convert the captured frame to an OpenCV format
def convert_frame_to_opencv(frame):
    # The frame is in YUV420 format, convert it to BGR for OpenCV
    image = cv2.cvtColor(frame, cv2.COLOR_YUV2BGR_I420)
    return image

# Initialize libcamera
camera_manager = libcamera.CameraManager()
camera = camera_manager.get_camera(0)

# Configure the camera
config = camera.create_still_configuration()
camera.configure(config)

# Start the camera
camera.start()

# Create a request and capture a frame
request = camera.create_request()
stream = config.at(0).stream()
buffer = camera.acquire_buffer(stream)
request.add_buffer(stream, buffer)
camera.queue_request(request)

# Allow the camera to warm up
import time
time.sleep(2)
```

```python
# Capture frames continuously
while True:
    # Capture a frame
    camera.capture(request)
    frame = buffer.map()

    # Convert the frame to OpenCV format
    image = convert_frame_to_opencv(frame)

    # Resize the image to 224x224
    resized_image = cv2.resize(image, (224, 224))

    # Display the resized image
    cv2.imshow("Resized Frame", resized_image)

    # Break the loop if 'q' key is pressed
    if cv2.waitKey(1) & 0xFF == ord('q'):
        break

# Release resources
camera.stop()
cv2.destroyAllWindows()
```

7.Concluzii:

Proiectul a demonstrat eficacitatea utilizării unui Raspberry Pi 4 împreună cu modulele de procesare a imaginii și machine learning pentru recunoașterea semnelor de circulație. Prin integrarea hardware-ului și software-ului adecvat, am reușit să creăm un sistem capabil să identifice semnele de circulație din imagini video în timp real. Acest proiect poate servi ca bază pentru dezvoltări ulterioare în domeniul sistemelor de asistență pentru șoferi și vehicule autonome.