Projekt – končni izdelek

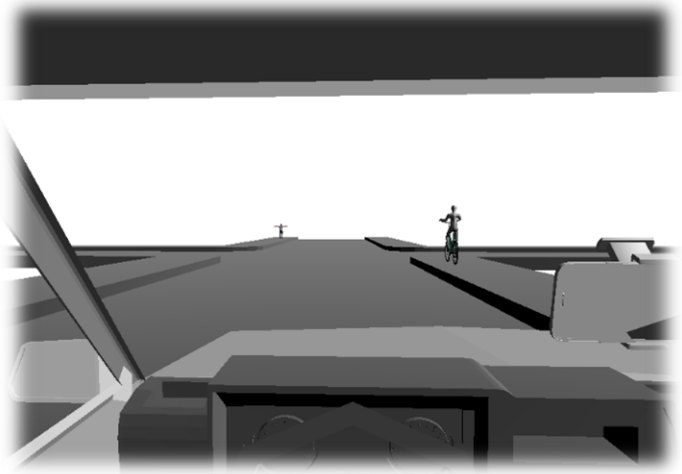
Skupina: PaztePešci

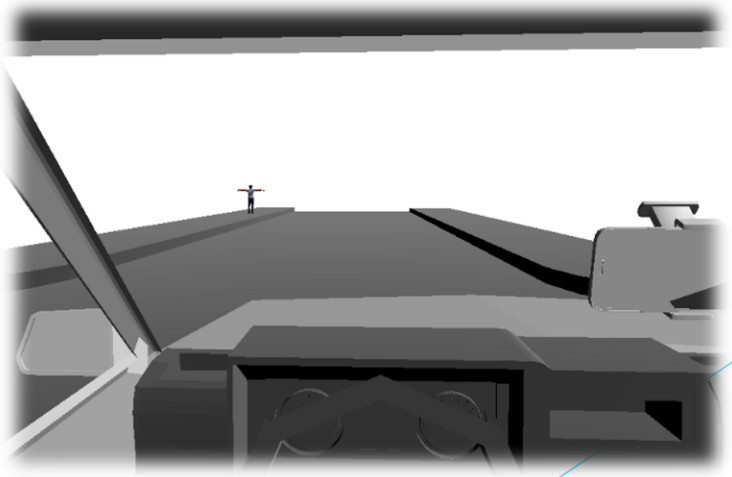
Člani: Niki Vrbnjak, Nay Milak, Denis Vuković

Mentor: Dr David Jesenko

Ideja projekta

- V aplikaciji zajemamo slike animacije in jih shranimo v mapo
- V programu ‚server.py' preberemo dodane slike (neskončna zanka), jih kompresiramo ter pošljemo na MQTT strežnik
- V ‚client.py' se povežemo na MQTT strežnik ter prejemamo slike (neskončna zanka) in jih najprej dekompresiramo, nato spustimo čez YOLOv5 model
- Glede na njegovo uspešnost generiramo sporočilo in ga pošljemo na ESP ploščico
- Tam podatke nadaljnjo obdelujemo in pošljemo na stm32 ploščico ter nazaj na ‚server.py' da prikaže rezultate model v naši aplikaciji





Dodane funkcionalnosti

- Kompresija slik pred pošiljanjem na strežnik
- Dekompresija slik po prejetju s strežnika

```
Slika image3.png poslana.
Slika image1.png poslana.
Slika image2.png poslana.
Slika image3.png poslana.
Slika image1.png poslana.
Slika image2.png poslana.
Slika image3.png poslana.
Slika image1.png poslana.
Slika image2.png poslana.
Slika image3.png poslana.
Slika image1.png poslana.
Slika image2.png poslana.
```

```
Prejeto sporocilo na topic: image
Prejeta slika...
Poslano sporocilo: PAZI PESEC! na topic: check
Prejeto sporocilo na topic: image
Prejeta slika...
Poslano sporocilo: PAZI PESEC! na topic: check
Prejeto sporocilo na topic: image
Prejeta slika
```

Kompresija slik

- Preberemo sliko
- Pretvorimo jo v sivinsko
- Spustimo skozi funkcijo ‚kompresiraj()‘
- Stisnjeno sliko pošljemo na MQTT strežnik pod topic – ‚image‘

```python
def check_and_send(client, path):
    if os.path.exists(path) and os.path.isdir(path):
        files = os.listdir(path)
        if files:
            for file in files:
                if file.endswith((".jpg", ".jpeg", ".png")):
                    img_path = os.path.join(path, file)
                    try:
                        with open(img_path, "rb") as img:
                            img_data = img.read()
                            img_array = np.frombuffer(img_data, dtype=np.uint8)
                            decoded_img = cv2.imdecode(img_array, cv2.IMREAD_COLOR)
                            comp_img = kompresiraj(decoded_img)
                            client.publish(topic, comp_img)
                            print(f"Slika {file} poslana.")
                    except Exception as e:
                        print(f"Napaka pri posiljanju slike {file}: {e}")
```

Dekompresija slik

- Na topic – ‚image‘ prejmemo kompresirano sliko
- Jo dekompresiramo s funkcijo ‚dekompresiraj()‘
- Spustimo čez naučen YOLOv5 model
- Glede na uspešnost modela generiramo sporočilo ter ga pošljemo dalje

```
#when receiving message
def on_message(client, userdata, msg):
    print(f"Prejeto sporocilo na topic: {msg.topic}")
    try:
        img_data = msg.payload
        if len(img_data) == 0:
            print("slika je prazna...")

        decomp_img = dekompresiraj(img_data) # decompress img

        if decomp_img is not None:
            #cv2.imshow("Prejeta slika", decomp_img)     # to show img
            print("Prejeta slika...")
            #cv2.waitKey(1)                             # waiting for image to be rendered
            if detected(decomp_img):
                message = "PAZI PESEC!"
            else:
                message = "NI NEVARNOSTI..."

            if client.publish(send_topic, message):
                print(f"Poslano sporocilo: {message} na topic: {send_topic}")
            else:
                print("Napaka pri posiljanju sporocila...")
        else:
            print("Napaka: Slika je prazna.")
    except Exception as e:
        print(f"Napaka pri dekodiranju slike: {e}")
```