Icon

Description automatically generated

**Tehnička škola Čakovec**

**ELABORAT ZAVRŠNOG RADA**

**SUSTAV ZA BEŽIČNO NAVIGIRANJE ROBOTSKIM VOZILOM**

|  |  |
| --- | --- |
| **Mentor:** | **Učenik:** |
| **Mirko Jambrošić, mag. ing. inf. et comm. techn.** | **Štefan Granatir, 4.RT** |

**Čakovec, svibanj 2023.**

**Tehnička škola Čakovec**

**Prosudbeni odbor za završni rad**

Učenik: **Štefan Granatir**

Razred: **4.RT**

Školska godina: **2022./2023.**

Obrazovno područje: **računalstvo**

Zanimanje: **Tehničar za računalstvo**

Naziv zadatka: **SUSTAV ZA BEŽIČNO NAVIGIRANJE ROBOTSKIM VOZILOM**

Opis zadatka: Učenik će predstaviti programsko rješenje kojim će omogućiti robotskom vozilu da navigira kroz 2d labirint

Učenik će se za konzultacije obratiti svojem mentoru.

Zadatak zadan: Rok predaje pisanog rada: Predviđen datum obrane:

16.05.2023.

Mentor:

Mirko Jambrošić, mag.ing.el.

SADRŽAJ

[UVOD 5](#_Toc135067693)

[RAZRADA 6](#_Toc135067694)

[ODABIR ALGORITMA ZA RJEŠAVANJE LABIRINTA 7](#_Toc135067695)

[OBRADA SLIKE – UNOS SLIKE 8](#_Toc135067696)

[OBRADA SLIKE – TRANSLACIJA 9](#_Toc135067697)

[OBRADA SLIKE – PRONALAŽENJE KLJUČNIH TOČAKA 12](#_Toc135067698)

[KOMUNIKACIJA S VOZILOM – PYTHON STRANA 14](#_Toc135067699)

[KOMUNIKACIJA S VOZILOM – C++ STRANA 15](#_Toc135067700)

[IZRADA VOZILA 16](#_Toc135067701)

[ZAKLJUČAK 17](#_Toc135067702)

[LITERATURA 18](#_Toc135067703)

[PRILOG 19](#_Toc135067704)

[ELEKTRIČNA SHEMA 19](#_Toc135067705)

[POPIS 20](#_Toc135067706)

[POPIS SLIKA 20](#_Toc135067707)

[POPIS TABLICA 20](#_Toc135067708)

[KONZULTACIJSKI LIST IZRADE ZAVRŠNOG RADA 21](#_Toc135067709)

# UVOD

Umjetna inteligencija (UI) predstavlja jedno od najaktualnijih područja razvoja tehnologije u današnjem svijetu. Svojim sposobnostima da obrađuje velike količine podataka, otkriva skrivene uzorke te pruža precizne i pouzdane rezultate. UI je postala središnja tema istraživanja i primjene u brojnim industrijama. Jedna od primjena UI koja je postala sve važnija u posljednjih nekoliko godina je njezina primjena u prijevozu. UI može se primijeniti u različitim aspektima prijevoza, od upravljanja prometom i poboljšanja sigurnosti vozača, do povećanja efikasnosti prijevoza i smanjenja troškova poslovanja.

U ovom radu bit će obrađena tema navigiranja vozila kroz 2d prostor implementacijom UI. Rad se sastoji od dva dijela, programskog i sklopovskog. Programski dio sastoji se od programa za obradu slike napisanog u Pythonu i programa za upravljanje vozilom napisanog u C++-u. Sklopovski dio sastoji se od raznih elektroničkih komponenti koje vozilu daju mogućnost navigiranja u prostoru.

Temu sam odabrao zbog osobne zanimacije prema autonomnim vozilima.

# RAZRADA

Programski dio za obradu slike, rješavanje labirinta i komunikaciju s vozilom pisan je u programskom jeziku Python. Kod izrade programskog rješenja korištene su razne biblioteke otvorenog koda, neke od njih su OpenCV (engl. Open Source Computer Vision), Numpy (engl. Numerical Python), PySerial (engl. Python Serial). Programski dio za upravljanje vozilom pisan je u C++ programskom jeziku i korištene su biblioteke SoftwareSerial i AccelStepper. Fizički dio vozila dizajniran je u Autodesk Fusion 360 programu za 3d modeliranje.

## ODABIR ALGORITMA ZA RJEŠAVANJE LABIRINTA

Za rješavanje labirinta korišten je BFS(engl. Breadth first search) algoritam. To je algoritam koji odlikuju laka implementacija i srednja razina kompleksnosti. Radi tako da se iz unesene prve točke širi na susjedne sve dok ne dođe do krajnje točke. To ga čini efikasnim za labirinte u kojima su početna i krajnja točka relativno blizu jedna drugoj za razliku od DFS(engl. Depth-first search) algoritma koji se rekurzivno širi u određenom smjeru. Njegova je jedina mana je prisutna kad su točke na većim udaljenostima jer se broj provjerenih točaka eksponencijalno povećava što rezultira njegovom malom implementacijom u stvarnom svijetu.

|  |
| --- |
| Slika Usporedba BFS i DFS algoritma |

## OBRADA SLIKE – UNOS SLIKE

Funkcija cv2.imread() dio je biblioteke OpenCV za obradu slike u programskom jeziku Python. Ova funkcija se koristi za čitanje slike iz datoteke na disku i pretvaranje u format koji se može koristiti za daljnju obradu. Sintaksa funkcije je sljedeća:

|  |
| --- |
| cv2.imread(putanja\_do\_slike, flagovi) |

Tablica Primjer upotrebe funkcije cv2.imread()

Ovdje je ‘putanja\_do\_slike’ relativna ili apsolutna putanja do datoteke slike koju želimo unjeti, a ‘flagovi’ su opcionalni argumenti koji se koriste za određivanje načina učitavanja slike. Funkcija se može koristiti u nekoliko formata, uključujući JPEG, PNG, BMP i druge.

Neki od uobičajenih ‘flagova’ koji se koriste uključuju:

* cv2.IMREAD\_COLOR: učitava sliku u boji(3 kanala, BGR)
* cv2.IMREAD\_GRAYSCALE: učitava sliku u crno-bijelom kanalu
* cv2.IMREAD\_UNCHANGED: učitava sliku u boji s alfa kanalom

Nakon što se slika učita, može se dalje obrađivati koristeći druge funkcije iz OpenCV biblioteke. na primjer, funkcija cv2.imshow() se može koristiti za prikazivanje slike na zaslon, dok funkcija cv2.imwrite() sprema određenu sliku na disk.

Važno je napomenuti da funkcija cv2.imread() može vratiti ‘None’ ako putanja do datoteke nije ispravna ili se datoteka ne može učitati. Stoga je važno provjeriti povratnu vrijednost funkcije prije nego što se nastavi s daljnjom obradom slike.

## OBRADA SLIKE – TRANSLACIJA

Funkcija cv2.goodFeaturesToTrack() u biblioteci OpenCV korištena je za pronalaženje rubnih točaka na slici. Ova funkcija koristi se na razne načine, kao što su praćenje objekata u pokretu, izračunavanje optičkog toka ili izrada panoramskih slika.

Korištenjem ove funkcije mogu se dobiti koordinate rubnih točaka. Funkcija radi tako da analizira intenzitete piksela u blizini svake točke na slici i izračunava kutnu razliku između vektora koja pokazuje smjer promjene intenziteta.

|  |
| --- |
| Slika Prikaz rada funkcije cv2.goodFeaturesToTrack() |

Sintaksa funkcije je sljedeća:

|  |
| --- |
| rubovi=cv2.goodFeaturesToTrack(slika, Max\_rubova, razina\_kvalitete, Min\_udaljenost, veličina\_blokova, HarrisDetector, k) |

Tablica Primjer upotrebe funkcije cv2.goodfeaturesToTrack

Ovdje, ‘slika’ predstavlja ulaznu sliku u kojoj se traže rubne točke, ‘Max\_rubova’ je maksimalni broj točaka koje će se pronaći, ‘razina\_kvalitete’ predstavlja prag kvalitete rubnih točaka, ‘Min\_udaljenost’ je minimalna udaljenost između dvije točke, ‘veličina\_blokova’ je veličina bloka za izračunavanje gradijenata, ‘HarrisDetector’ se koristi za odlučivanje hoće li se koristiti Harris detektor ruba ili ne, dok se ‘k’ koristi za izračunavanje odgovarajućih pragova za Harris detektor.

Za izračunavanje matrice transformacije perspektive korištena je funkcija cv2.getPerspectiveTransform(), funkcija kao argument prima dva seta koordinata, od kojih prvi set predstavnja koordinate točaka u ulavnoj slici, a drugi set predstavlja koordinate točaka na izlaznoj slici. Na temelju tih koordinata, funkcija izračunava matricu transformacije perspektive koja se dalje koristiti za transformaciju perspektive slike.

Sintaksa funkcije je sljedeća:

|  |
| --- |
| pts1=np.float32([[RubneTocke[**1**].x,RubneTocke[**1**].y],[RubneTocke[**0**].x,RubneTocke[**0**].y],[RubneTocke[**2**].x,RubneTocke[**2**].y],[RubneTocke[**3**].x,RubneTocke[**3**].y]])  pts2=np.float32([[najmanjaX.x, najmanjaX.y-konstUdaljenost], [najvecaX.x, najvecaX.y-konstUdaljenost], [najmanjaX.x, najmanjaX.y], [najvecaX.x, najvecaX.y]])  matrix=cv2.getPerspectiveTransform(pts1, pts2) |

Tablica primjer korištenja funkcije cv2.getPerspectiveTransform() sa ulaznim i izlaznim setom koordinata

Funkcija cv2.warpPerspective() se koristi za primjenu matrice transformacije perspektive na ulaznu sliku kako bi se dobila transformirana izlazna slika. Ova funkcija prima matricu transformacije perspektive koju smo izračunali korištenjem cv2.getPerspectiveTransform(), kao i dimenzije izlazne slike.

|  |
| --- |
| Slika promjena slike nakon upotrebe funkcije cv2.warpPerspective |

## OBRADA SLIKE – PRONALAŽENJE KLJUČNIH TOČAKA

Funkcija cv2.moments() koristi se za izračunavanje različitih karakteristika kontura slike. Funkcija prima binarnu sliku koja sadrži konture i vraća rječnik s karakteristikama kontura. Jedna od karakteristika koju vraća je središte kontrue, koja se ovdje koristi za automatizirano pronalaženje početne i krajnje točke labirinta.

Da bi se pronašlo središte konture, prvo je potrebno pronaći konture slike pomoću funkcije cv2.findContours(), Nakon pronalaska kontura, moguće je primjeniti funkciju cv2.moments() na svaku konturu kako bi se dobila njezina svojstva. u rječniku karakteristika konture, središte konture predstavljeno je kao vrijednost ‘m10’ i ‘m01’, koje su zatim korištene za izračun koordinata središta konture.

|  |
| --- |
| **for** i **in** contours:  M = cv2.moments(i)  rect=cv2.minAreaRect(i)  area=cv2.contourArea(i)  **print**(area)  **if** M['m00'] != **0**:  cx = int(M['m10']/M['m00'])  cy = int(M['m01']/M['m00'])  **if**(area>**500**)**and**(area<**1200**):  brojac+=**1**  pixelVal = image[cx-**8**,cy-**8**]  **print**(cx,cy)  **print**("boja=",pixelVal )  cv2.circle(image, (cx, cy), **3**, (**0**, **255**, **255**), -**1**)  cv2.putText(image, str(brojac), (cx - **20**, cy - **20**) ,cv2.FONT\_HERSHEY\_SIMPLEX, **0.5**, (**0**, **0**, **0**), **2**) |

Tablica primjer korištenja ffinkcije cv2.moments na konture koje se nalaza na slici

|  |
| --- |
| Slika prikaz rada funkcije cv2.moments() |

## KOMUNIKACIJA S VOZILOM – PYTHON STRANA

PySerial je Python biblioteka koja se koristi za komunikaciju sa serijskim uređajima poput Arduino mikrokontrolera, senzora i drugih perifernih uređaja. PySerial je popularna biblioteka u području ugrađenih sustava, interneta stvari i drugih sličnih područja. Korištenjem PySerial biblioteke moguće je ostvariti bluetooth komunikaciju između računala i vozila putem serijskog porta.

|  |
| --- |
| ser = serial.Serial("COM4", 38400, timeout=1) |

Tablica kreiranje objekta ser tipa Serial koji uspostavanja vezu s mikrokontrolerom

Ova funkcija prihvaća tri argumenta: naziv porta, brzinu prijenosa i opcionalne postavke za serijsku vezu. Nakon otvaranja porta mogu se koristiti funkcije write() i read() za slanje i primanje podataka preko serijske veze.

|  |
| --- |
| ser.write(bytes(str(f'poruka'), 'utf\_8'))*#slanje*  input\_data=ser.readline() *#primanje* |

Tablica primjer slanja i primanja podataka preko serijskog porta na Python strani

## KOMUNIKACIJA S VOZILOM – C++ STRANA

Za bluetooth komunikaciju između računala i vozila koristi se HC-05 modul. HC-05 je Bluetooth modul koji se često koristi s Arduino platformom za bežičnu komunikaciju. Međutim, kako Arduino nema ugrađen softverski serijski port, može biti teško uspostaviti vezu između HC-05 modula i Arduina. U tom slučaju, SoftwareSerial biblioteka se često koristi za stvaranje virtualnog softverskog serijskog porta koji se može koristiti za uspostavljanje veze između HC-05 modula i Arduina. Kod komunikacije između Arduino mikroupravljača i modula koristi se UART(engl. Universal asynchronous receiver-transmitter) protokol, što znači da je potrebno spojiti Rx, Tx, VCC i GND pinove za uspješnu komunikaciju.

|  |
| --- |
| #include <SoftwareSerial.h>  SoftwareSerial serial\_con(Rx\_pin,Tx\_pin); |

Tablica uključivanje bbiblioteke SoftwareSerial i stvaranje njezinog objekta

Budući da UART protokol ne koristi sinkronizacijsku liniju potrebno je definirati baud rate (broj bitova po sekundi).

|  |
| --- |
| serial\_con(9600); |

Tablica pokretanje serijske komunikacije objekta na baud rate-u od 9600

Kod čitanja sa serijskog porta potrebno je dodatno definirati veličinu buffera, kako ne bi došlo do učitavanje prevelike količine podataka u jednom trenutku. Veličina buffera varira od komponente do komponente, u ovom slučaju, na Arduino Nano mikroupravljaću, ona iznosi 64 bajta.

|  |
| --- |
| #define BUFFER\_SIZE 64  byte byte\_count=serial\_con.available();  **if**(byte\_count){  **for**(i=0;i<byte\_count;i++){  inChar=serial\_con.read();  indata[i]=inChar;  }  indata[i]='\n';  } |

Tablica primjer primanja podataka preko serijskog porta na strani mikrokontrolera

## IZRADA VOZILA

Fizički dio vozila izrađen je prema načelima tehničke dokumentacije. Zbog korištenja manje količine ulazno-izlaznih pinova i težnje prema kompaknosti cijelokupne konstrukcije, korišten je mikrokontroler Arduino Nano. On upravlja cijelim procesom i izvršava sve naredbe zadane programom. Komunikaciju s računalom omogućuje HC-05 bluetooth modul koji se na mikrokontroler povezuje pomoću UART protokola. Robot se kreće pomoću “ NEMA11 koračnih motora koje kontroliraju zasebni A4988 upravljački čipovi. Smješteni su na prednju stranu vozila. Kako bi vozilo bilo stablinije, na zadnju stranu smješten je 360 kotač. Kako A4988 zahtijeva minimalno 8V za napajanje koračnog motora, Za napajanje cijelog vozila koristie se baterija koja se sastoji od 3 serijski povezanih 18650 članaka. Baterija daje 12V napona što je dovoljno za dugotrajnu upotrebu vozila. Konstrukcija vozila u potpunosti je dizajnirana u Autodesk Fusion 360 programu za 3d modeliranje i 3d printana PLA filamentom. Dodatnu stabilnost vozilu daju metalne matice i vijci.

# ZAKLJUČAK

Primjena umjetne inteligencije u kombinaciji s OpenCV-om priža snažan alat za obradu slika i analizu vizualnih podataka. Korištenjem OpenCV-a omogućuje se automatsko prepoznavanje objekata, klasifikacija slika, praćenje i detekcija pokreta te mnoge druge funkcionalnosti. Njegova primjena u stvarnom životu obuhvaća mnoga područja, uključujući sigurnost, medicinu, industriju i promet. Integracija istog pruža mogućnost automatizacije, povećanja učinkovitosti i poboljšanja kvalitete analize slika, što ima širok utjecaj na naš svakodnevni život i industrije.

# LITERATURA

ansh2919. (2022, November 6). *Serial Communication between Python and Arduino*. Retrieved from hackster.io: https://www.hackster.io/ansh2919/serial-communication-between-python-and-arduino-e7cce0

Bapat, K. (2018, July 8). *Find the center of a Blob (Centroid) using OpenCV*. Retrieved from LearnOpenCV: https://learnopencv.com/find-center-of-blob-centroid-using-opencv-cpp-python/

openCV. (n.d.). *Shi-Thomasi Corner Detector & Good Features to Track*. Retrieved from OpenCV: https://docs.opencv.org/3.4/d4/d8c/tutorial\_py\_shi\_tomasi.html

team, E. A. (2023). *How to implement a ​breadth-first search in Python*. Retrieved from educative: https://www.educative.io/answers/how-to-implement-a-breadth-first-search-in-python

# PRILOG

## ELEKTRIČNA SHEMA

|  |
| --- |
|  |

# POPIS

## POPIS SLIKA

[Slika 1Usporedba BFS i DFS algoritma 7](#_Toc135066384)

[Slika 2 Prikaz rada funkcije cv2.goodFeaturesToTrack() 9](#_Toc135066385)

[Slika 3promjena slike nakon upotrebe funkcije cv2.warpPerspective 11](#_Toc135066386)

[Slika 4 prikaz rada funkcije cv2.moments() 13](#_Toc135066387)

## POPIS TABLICA

[Tablica 1 Primjer upotrebe funkcije cv2.imread() 8](#_Toc135004865)

[Tablica 2 Primjer upotrebe funkcije cv2.goodfeaturesToTrack 9](#_Toc135004866)

[Tablica 3 primjer korištenja funkcije cv2.getPerspectiveTransform() sa ulaznim i izlaznim setom koordinata 10](#_Toc135004867)

[Tablica 4 primjer korištenja ffinkcije cv2.moments na konture koje se nalaza na slici 12](#_Toc135004868)

[Tablica 5 kreiranje objekta ser tipa Serial koji uspostavanja vezu s mikrokontrolerom 14](#_Toc135004869)

[Tablica 6 primjer slanja i primanja podataka preko serijskog porta na Python strani 14](#_Toc135004870)

[Tablica 7 uključivanje bbiblioteke SoftwareSerial i stvaranje njezinog objekta 15](#_Toc135004871)

[Tablica 8 pokretanje serijske komunikacije objekta na baud rate-u od 9600 15](#_Toc135004872)

[Tablica 9 primjer primanja podataka preko serijskog porta na strani mikrokontrolera 15](#_Toc135004873)

# KONZULTACIJSKI LIST IZRADE ZAVRŠNOG RADA

Ime i prezime učenika: **Štefan Granatir**

Razred: 4.RT

Program-zanimanje: **tehničar za računalstvo**

Mentor: **Mirko Jambrošić, mag.ing.inf. et comm. techn.**

|  |  |  |  |
| --- | --- | --- | --- |
| **Red.**  **br.** | **Datum**  **konzultacije** | **Sadržaj rada** | **Potpis mentora** |
| 1. | 10.1.2023 | Demonstracija programa za rješavanje labirinta , te dane daljnje upute o izradi završnog rada |  |
| 2. | 31.1.2023 | Pregled napredka, te dane daljnje upute o izradi  završnog rada |  |
| 3. | 3.4.2023 | Prikazan fizički prototip vozila. Pregled napredka, te dane daljnje upute o izradi završnog rada |  |
| 4. | 1.5.2023 | Pregledan elaborat i napredak završnog rada |  |
|  |  |  |  |
|  |  |  |  |

Završni rad ocijenjen pozitivno

DA / NE

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

(potpis mentora)