Universitatea “Politehnica” din București

Facultatea de Electronică, Telecomunicații și Tehnologia Informației

***Sistem de monitorizare a traficului pentru identificarea unor potențiali urmăritori***

###### **Proiect de diplomă**

prezentat ca cerință parțială pentru obținerea titlului de

*Inginer* în domeniul *Electronicii, Telecomunicației și Tehnologiei Informației*

programul de studii de licență *Electronică Aplicată*

Conducător(i) științific(i) Absolvent

*ŞL. dr. ing. Valentin-Adrian NIȚĂ Mădălina-Elena COSTEA*

##### Anul 2021

|  |  |  |
| --- | --- | --- |
| |  |  | | --- | --- | | Universitatea “Politehnica” din Bucureşti Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei Program de studiu | **Anexa 1** | |
|  |
| **TEMA PROIECTULUI DE DIPLOMĂ** |
| a studentului **COSTEA V. Mădălina-Elena , 441B-ELA** |
|  |
| **1. Titlul temei:** Sistem de monitorizare a traficului pentru identificarea unor potențiali urmăritori  **2. Descrierea temei și a contribuției personale a studentului (în afara părții de documentare):**  Obiectivul proiectului de diplomă este proiectarea și dezvoltarea unui sistem care verifică existența unui alt automobil care te urmărește în trafic. O cameră monitorizează toate numerele de înmatriculare ale mașinilor detectate în jurul tău, în trafic și te anunță dacă există un vehicul care apare prea des în urma ta. Sistemul se bazează pe urmatoarele componente: - platforma Raspberry Pi - un modul de cameră compatibil cu placa de dezvoltare Raspberry Pi, conectat prin USB -telefon conectat la placă prin wireless  Pe placă se va crea o bază de date, unde se vor inregistra toate numerele de înmatriculare detectate și odată cu aparitia fiecărui număr înregistrat, se va notifica telefonul.  Resursele folosite în dezvoltarea proiectului: Python, MySQL, procesare de imagini (librăria OpenCV)  **3. Discipline necesare pt. proiect:**  TPI (Tehnologii de Programare în Internet), POO (Programare Orientata pe Obiect), BD (Baze de Date), MC (Microcontrollere)  **4. Data înregistrării temei:**2020-12-08 18:25:41 |
| |  |  |  | | --- | --- | --- | | |  |  | | --- | --- | | **Conducător(i) lucrare,** ŞL. dr. ing. Valentin-Adrian NIȚĂ | **Student,** | | | |  |  | | --- | --- | | S.L. Dr. Ing. Răzvan Crăciunescu |  | | | |  |  | | --- | --- | | **Director departament,** | **Decan,** Prof. dr. ing. Mihnea UDREA | | |
| |  |  | | --- | --- | | Cod Validare: **1a89b7afe0** |  | |

Anexa 5

**Declarație de onestitate academică**

Prin prezenta declar că lucrarea cu titlul “*Sistem de monitorizare a traficului pentru identificarea unor potențiali urmăritori*”, prezentată în cadrul Facultății de Electronică, Telecomunicații și Tehnologia Informației a Universității “Politehnica” din București ca cerință parțială pentru obținerea titlului de *Inginer* în domeniul *Electronicii, Telecomunicației și Tehnologiei Informației*, programul de studii *Electronică Aplicată* este scrisă de mine și nu a mai fost prezentată niciodată la o facultate sau instituție de învățămînt superior din țară sau străinătate.

Declar că toate sursele utilizate, inclusiv cele de pe Internet, sunt indicate în lucrare, ca referințe bibliografice. Fragmentele de text din alte surse, reproduse exact, chiar și în traducere proprie din altă limbă, sunt scrise între ghilimele și fac referință la sursă. Reformularea în cuvinte proprii a textelor scrise de către alți autori face referință la sursă. Înțeleg că plagiatul constituie infracțiune și se sancționează conform legilor în vigoare.

Declar că toate rezultatele simulărilor, experimentelor și măsurătorilor pe care le prezint ca fiind făcute de mine, precum și metodele prin care au fost obținute, sunt reale și provin din respectivele simulări, experimente și măsurători. Înțeleg că falsificarea datelor și rezultatelor constituie fraudă și se sancționează conform regulamentelor în vigoare.

București, *iunie 2021*

Absolvent *Mădălina-Elena COSTEA*

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

(semnătura în original)

Cuprins

[Tabel figuri 8](#_Toc74594285)

[Listă Acronime 10](#_Toc74594286)

[*Introducere* 12](#_Toc74594287)

[1. *Capitolul 1 - Funcții implementate*: 13](#_Toc74594288)

[1.1 Email alert 13](#_Toc74594289)

[1.2 Optical character recognition 13](#_Toc74594290)

[1.3 Plate extraction 14](#_Toc74594291)

[1.4. Sqlite database 16](#_Toc74594292)

[2. *Capitolul 2 - Tehnologii folosite*: 16](#_Toc74594293)

[2.1. Raspberry Pi 4 17](#_Toc74594294)

[2.2. Python 18](#_Toc74594295)

[2.3. Sqlite 18](#_Toc74594296)

[2.4. Tesseract 18](#_Toc74594297)

[2.5. Opencv 18](#_Toc74594298)

[2.6. ANPR - Automated Number Plate Recognition 20](#_Toc74594299)

[2.7 Clasificator în cascadă bazați pe trăsături Haar pentru detectarea obiectelor 21](#_Toc74594300)

[2.2. Configurarea mediului de lucru 22](#_Toc74594301)

[3. *Capitolul 4 - Testare si probleme intampinate:* 23](#_Toc74594302)

[3.1. Claritatea imaginilor/video: 23](#_Toc74594303)

[4.2. Probleme software (sisteme de operare folosite) 23](#_Toc74594304)

[4. *Capitolul 5 - Concluzii*: 23](#_Toc74594305)

[Bibliografie 24](#_Toc74594306)

# Tabel figuri

[**Figură 1**. Raspberry Pi Board - Version 4 Model B 18](#_Toc74594246)

[**Figură 2**. NVIDIA Jetson Nano Developer Embedded Development Board A57 Artificial Intelligence AI Development Platform 20](#_Toc74594247)

[**Figură 3.** Exemplu de nr. de înmatriculare cu rezoluție acceptabilă 23](#_Toc74594248)

# Listă Acronime

USB = Universal Serial Bus

SBC = Single Board Computer

GPIO = General-purpose input/output

WiFi = Wireless Fidelity

PIL= Python Imaging Library

MB = Mega Bytes

RAM = Random Access Memory

HDMI = High-Definition Multimedia Interface

ARM = Advanced RISC Machine

OS = Operating System

SD = Secure Digital

PC = Personal Computer

OpenCV = Open Source Computer Vision Library

BSD = Berkeley Software Distribution

GUI = Graphical User Interface

GND = ground

CPU = Central Processing Unit

BGR = Blue Green Red

RGB = Red Green Blue

FPS = Frames Per Second

RNN = Recurrent Neural Network

LSTM = Long Short-Term Memory

YOLO = You Only Look Once

ANPR = Automated Number Plate Recognition

SOC = System On Chip

# *Introducere*

Integrarea masivă a tehnologiilor informatice, sub diferite aspecte ale lumii moderne, a condus la tratarea autovehiculelor ca resurse conceptuale în sistemele informaționale. Întrucât un sistem informațional autonom nu are nicio semnificație fără date, este necesară transpunerea informațiilor despre vehicule între sistemul informațional și realitate. Acest lucru poate fi realizat cu ajutorul echipamentelor inteligente specializate, care permit identificarea acestora prin intermediul plăcuțelor de înmatriculare. Printre aceste echipamentele inteligente, se menționează și sistemul de detectare și recunoaștere a plăcuțelor de înmatriculare ale vehiculelor.

Acest sistem, precum îi sugerează și numele, este utilizat pentru a detecta plăcuțele, urmând să extragă caracterele acestora din imagini sau chiar din video, toate acestea datorându-se modulelor de calcul care utilizează algoritmi de localizare, plăci de segmentare și de recunoaștere a caracterelor.

Proiectul se bazează foarte mult pe utilitatea sa, deoarece putem utiliza caracterele alfanumerice extrase, în scopuri mai aplicative, cum ar fi un vehicul furat sau pentru a afla din ce regiune a țării face parte vehiculul, numele proprietarului, timpul de vizitare al unui loc etc. Această propunere de proiect poate economisi din timpul și din energia ofițerilor care monitorizează vehiculele.

Dezavantajele acestei monitorizări umane sunt:

● Ochiul uman nu poate monitoriza mai multe locuri simultan.

● Verificarea manuală a numerelor de înmatriculare dintr-un sistem, este o sarcină

foarte solicitantă și aproape imposibilă.

Aceste dezavantaje pot fi remediate cu ajutorul acestui proiect deoarece:

● Camerele pot surprinde un cadru mult mai larg decât ochiul uman.

● Este dovedit a fi mult mai eficient din punct de vedere al timpului.

În proiectul nostru, plăcuțele de înmatriculare sunt identificate și transpuse în sistem, iar ulterior sunt procesate pentru a verifica dacă vehiculul este sau nu înmatriculat. Dacă numărul vehiculului este procesat cu succes, atunci va fi stocat în fișierul bazei de date.

# *Capitolul 1 - Funcții implementate*:

## Email alert

Python vine cu modulul smtplib încorporat pentru a facilita trimiterea e-mailurilor, utilizând Simple Mail Transfer Protocol ( SMTP). smtplib folosește protocolul RFC 821 pentru SMTP. Un server local de depanare SMTP poate fi util pentru remedierea oricăror probleme legate de funcționalitatea e-mailului și pentru a vă asigura că nu există erori înainte de a trimite un mail.

Iată patru pași de bază pentru trimiterea e-mailurilor folosind Python:

• Configurarea serverului SMTP și conectarea la un cont personal.

• Crearea mesajului MIMEMultipart și încărcarea lui cu anteturile corespunzătoare pentru câmpurile From, To și Subject.

• Compunerea mesajului.

• Trimiterea mesajului utilizând serverul SMTP.

În cazul în care se decide utilizarea unui cont Gmail pentru a trimite e-mailurile, se recomandă să se configureze un cont creat special pentru testare, folosit în dezvoltarea codului. Acest lucru se datorează faptului că vor trebui ajustate setările de securitate ale contului Gmail pentru a permite accesul din codul Python și pentru că există șansa expunerii accidentale ale datelor de autentificare. De asemenea, inbox-ul contului creat se va umple rapid cu e-mailuri de testare, motiv suficient de bun pentru a crea un nou cont Gmail pentru dezvoltare.

Pentru crearea funcției de alertare a telefonului prin E-mail, am creat un cont de Gmail numit [alerta.urmarire1@gmail.com](mailto:alerta.urmarire1@gmail.com), iar pentru a face posibilă utilizarea acestuia în Python, a fost nevoie de activarea autentificării în 2 pași, deoarece, așa cum am precizat mai sus, este necesară ajutarea setărilor de securitate pentru a folosi adresa în cod.

## Optical character recognition

Python-tesseract este un instrument de recunoaștere optică a caracterelor (OCR) pentru Python. Adică, va recunoaște și „va citi” textul încorporat în imagini. Python-tesseract este un pachet pentru motorul Tesseract-OCR de la Google. Este, de asemenea, util ca script independent pentru tesseract, deoarece poate citi toate tipurile de imagini acceptate de bibliotecile de imagini Pillow și Leptonica, inclusiv jpeg, png, gif, bmp, tiff și altele. În plus, dacă este utilizat ca script, Python-tesseract va afișa textul recunoscut în loc să îl scrie într-un fișier.[1]

**Parametri folosiți în cadrul funcției**

image\_to\_data(image, lang=None, config='', nice=0, output\_type=Output.STRING, timeout=0, pandas\_config=None)

text = pytesseract.image\_to\_string(plate, config=tessdata\_dir\_config, lang="eng")

* **image** Obiect sau Șir de caractere – PIL (Python Imaging Library) Image/matrice NumPy sau calea imaginii ce urmează a fi procesată de Tesseract. În cazul utilizării unui obiect în locul unei imagini, pyteseract va converti în mod implicit imaginea în modul RGB.
* **lang** Șir de caractere – limba Tesseract. Ex. eng
* **config** Șir de caractere - Orice steaguri de configurare personalizate suplimentare care nu sunt disponibile prin funcția pytesseract. Ex.: config='--psm 6'

## 1.3 Plate extraction

Pașii de extragere a caracterelor din numărul de înmatriculare

1. Imaginea inițială



1. Transformarea în tonuri de gri



1. Extragerea contururilor



1. Delimitarea numărului de înmatriculare



1. Extragerea numărului de înmatriculare



## 1.4. Sqlite database

# *Capitolul 2 - Tehnologii folosite*:

Metoda propusă evidențiază cele cinci etape ale implementării: captarea imaginilor, preprocesarea, extragerea numărului de înmatriculare de pe vehicul, segmentarea caracterelor de pe plăcuță și recunoașterea optică a caracterelor. Partea de captare a imaginii este realizată de cameră, în timp ce restul, de un Raspberry Pi (versiunea 4 model B).

Imaginile de intrare sunt de fapt cadre extrase din fluxul video live captat de cameră. Pentru acest proiect, folosim camera SriHome SH004, cu o rezoluție de 1080p, la 30 de cadre pe secundă (fps).

Preprocesarea joacă un rol important în detectarea plăcilor de înmatriculare prin conversia imaginilor de intrare într-un format adecvat pentru alte părți ale algoritmului. Pentru procesarea imaginilor am folosit funcții ale bibliotecii openCV precum cvtcolor-pentru conversia în tonuri de gri, filtrare bilaterală pentru netezirea imaginilor și reducerea zgomotului; păstrând în același timp marginile pentru detectarea marginilor.

Procesul de extragere a plăcuțelor include diferite tehnici de procesare de imagine. Acest proces este împărțit în două părți: detectarea plăcuțelor prin Haar-like Features și prin detectarea muchiilor (Edge Detection).

Segmentarea caracterelor este o parte a procesării imaginilor care are ca scop eliminarea datelor inutile. După segmentare, rămân numai acele caractere care aparțin plăcuței.

Recunoașterea optică a caracterelor este realizată de Tesseract versiunea 4, care implementează un motor de recunoaștere bazat pe memoria Long Short Term Memory (LSTM). LSTM este un fel de Rețea neuronală recurentă (RNN). Caracterele selectate sunt trimise la motorul de recunoaștere optică a caracterelor (OCR), care returnează codul ASCII al numărului de înmatriculare.

## Raspberry Pi 4

Raspberry Pi este un SBC (Single-board computer) de dimensiuni reduse (aproximativ dimensiunea unui card de credit), creat de Raspberry Pi Foundation. Scopul acestui proiect a fost de a crea un produs mic și ieftin, capabil de programare avansată . „Pi” din numele său provine din limbajul de programare în dezvoltare, numit Python.

Deși acest computer este mai lent decât un computer obișnuit sau un laptop, are totuși puterea de a rula câteva sarcini impresionante. În următoarele capitole, se va prezenta multitudinea de sarcini pe care le poate îndeplini această placă. Raspberry Pi are o caracteristică specială: are intrare / ieșire de uz general (General-Purpose Input/Output - GPIO). Aceasta extinde capacitățile mini computerului la un micro-controler foarte puternic. Placa are, de asemenea, un cip WiFi integrat și se poate conecta la internet prin wireless.

În funcție de model, placa poate avea între 258 MB și 8 GB RAM. Raspberry Pi 4 B + utilizat în acest proiect are o intrare USB de tip C, utilizată în principal pentru alimentare și un port HDMI pentru conectarea la un monitor sau TV. De asemenea, are o intrare utilizată pentru conectarea unei game largi de camere. Sbc-ul are un procesor ARM cortex similar cu procesoarele folosite la smartphone-uri. Există mai multe tipuri de plăci Raspberry Pi: Raspberry Pi 1 A, Raspberry Pi 1 B, Raspberry Pi 2B, Raspberry Pi 3B, Raspberry Pi 4B + și așa mai departe.

Sistemul de operare optimizat pentru un Raspberry Pi este Raspbian, care este derivat din Debian. Este o distribuție Linux, ceea ce îl face potrivit pentru programarea și dezvoltarea de software. Acest sistem de operare poate fi copiat pe un card SD și apoi introdus în plăcuță, iar el va rula automat odată conectat la alimentare. Pentru acest proiect, am ales să rulez Windows 10, ca sistem de operare, deoarece a fost lansat foarte recent și am dorit să îi testez perfomanțele.



**Figură 1**. Raspberry Pi Board - Version 4 Model B

## 2.2. Python

## 2.3. Sqlite

## 2.4. Tesseract

## 2.5. Opencv

OpenCV (Open Source Computer Vision Library) este o bibliotecă open-source licențiată BSD care include câteva sute de algoritmi de „computer vision”.

[2]Această bibliotecă este utilizată în aproape toate proiectele care utilizează vizionarea computerizată. Este un instrument foarte puternic, capabil să calculeze algoritmi de procesare a imaginilor în timp real. Această tehnologie reprezintă nucleul proiectelor, deoarece majoritatea funcțiilor legate de „computer vision” sunt implementate cu ajutorul acesteia.

OpenCV are o structură modulară, ceea ce înseamnă că pachetul include mai multe biblioteci partajate sau statice. Sunt disponibile următoarele module:

* Core Functionality (core) - un modul compact care definește structurile de date de bază, inclusiv matricele multi-dimensionale și funcțiile de bază utilizate de toate celelalte module.
* Procesarea imaginilor (imgproc) - un modul de procesare a imaginii care include filtrarea liniară și neliniară a imaginilor, transformările geometrice ale imaginii (redimensionare, deformare de perspectivă, remapare generică pe bază de tabel), conversia spațiului de culoare, histograme și așa mai departe.
* Analiza video (video) - un modul de analiză video care include estimarea mișcării, scăderea fundalului și algoritmi de urmărire a obiectelor.
* Calibrarea camerei și reconstrucția 3D (calib3d) - algoritmi de bază cu geometrie cu vizualizare multiplă, calibrarea camerei single și stereo, estimarea poziției obiectelor, algoritmi de corespondență stereo și elemente de reconstrucție 3D.
* 2D Features Framework (features2d) - detectoare de caracteristici, descriptori și potrivire de descriptori.
* Detectarea obiectelor (objdetect) - detectarea obiectelor și instanțelor claselor predefinite (de exemplu: fețe, ochi, oameni, mașini și așa mai departe).
* High-level GUI (highgui) - o interfață ușor de utilizat pentru capabilități UI simple.
* I / O video (videoio) - o interfață ușor de utilizat pentru video și codecuri video.
* ... alte module de asistență, cum ar fi module de testare FLANN și Google, conexiuni cu Python și altele.



## 2.6. ANPR – Recunoașterea automată a numerelor de înmatriculare

Recunoașterea automată a plăcuțelor de înmatriculare (Automated Number Plate Recognition ANPR) este un domeniu în curs de intensă cercetare în vremurile noastre. Au fost propuse diverse tehnici de ultimă generație pentru a pune totul cap la cap, însă rezultatele nu au fost întotdeauna mulțumitoare. Aceste tehnici includ notații matematice bazate pe vizionare computerizată și antrenări bazate pe Rețele neuronale recurente(RNN) și rețele Long Short-Term Memory (LSTM) pentru a ajuta la o mai bună recunoaștere a caracterelor (OCR) de pe plăcuțele de înmatriculare.

În prezent, cea mai puternică tehnică de deep learning pentru implementarea ANPR este YOLO (You Only Look Once). Algoritmul YOLO este foarte rapid în comparație cu alți algoritmi și, prin urmare, este potrivit pentru utilizarea în recunoașterea automată a numărului de înmatriculare, având o precizie ridicată.

În afară de partea software, utilizarea deep learning a necesitat o platformă hardware cu o performanță ridicată pentru a rula algoritmii și o cameră care poate capta imagini la o

rezoluție mare, cu destul de multe cadre pe secundă. Alte specificații necesare pentru realizarea părții hardware pot fi – procesor multi-core cu memorie ram semnificativă și un procesor suplimentar.

Jetson Nano ar putea fi considerat ca o alternativă mai bună în comparație cu Raspberry Pi în cazul configurării proiectului de recunoaștere a plăcuței de înmatriculare.



**Figură 2**. NVIDIA Jetson Nano Developer Embedded Development Board A57 Artificial Intelligence AI Development Platform

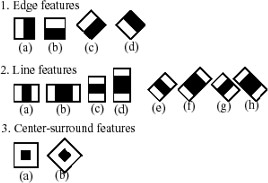
## 2.7 Clasificator în cascadă bazați pe trăsături Haar pentru detectarea obiectelor

Detectorul de obiecte descris mai jos a fost propus inițial de Paul Viola și îmbunătățit de Rainer Lienhart. În primul rând, un clasificator (sau o cascadă de clasificatori care funcționează cu Haar-like Features) este antrenat cu câteva sute de vizualizări ale unui anumit obiect (adică o față sau o mașină), numite exemple pozitive, care sunt scalate la aceeași dimensiune (să zicem, 20x20) și exemple negative - imagini arbitrare de aceeași dimensiune [3].

După ce un clasificator este antrenat, acesta poate fi aplicat unei regiuni de interes (de aceeași dimensiune ca și cea utilizată în timpul antrenamentului), în cadrul unei imagini de intrare. Clasificatorul afișează „1” la ieșire, dacă regiunea recunoaște obiectul (adică față / mașină) și „0” în caz contrar. Pentru a căuta obiectul în întreaga imagine, puteți muta fereastra de căutare și puteți verifica fiecare locație folosind clasificatorul. Acesta este conceput astfel încât să poată fi „redimensionat” cu ușurință pentru a putea găsi obiectele de interes la diferite dimensiuni, ceea ce este mai eficient decât redimensionarea imaginii în sine. Deci, pentru a găsi un obiect de o dimensiune necunoscută în imagine, procedura de scanare trebuie făcută de mai multe ori la diferite scale .

Cuvântul „cascadă” din numele clasificatorului se refera la faptul că clasificatorul rezultat constă din mai multe clasificatoare (etape) mai simple care sunt aplicate ulterior unei regiuni de interes până când, într-o anumită etapă, candidatul este respins sau toate etapele sunt trecute. Cuvântul „boosted” înseamnă că acești clasificatorii, din fiecare etapă a cascadei, sunt ei înșiși complexi și sunt construiți din clasificatori de bază folosind una dintre cele patru tehnici diferite de amplificare. Clasificatorii de bază sunt clasificatori de arbori de decizie cu cel puțin 2 frunze. Funcțiile Haar sunt elementele de intrare pentru clasificatorii de bază și sunt calculate așa cum este descris mai jos. Algoritmul folosește următoarele caracteristici Haar. Figura descrie câteva caracteristici ale algoritmului.

Caracteristica utilizată într-un anumit clasificator este specificată prin forma sa (1a, 2b etc.), poziția în regiunea de interes și scala (această scală nu este aceeași cu scala utilizată în etapa de detectare, deși aceste două scale sunt multiplicate). De exemplu, în cazul celei de-a doua linii (2c), răspunsul este calculat ca diferența dintre suma pixelilor de sub dreptunghi care acoperă întreaga caracteristică (inclusiv cele două dungi albe și dunga neagră din mijloc) și suma pixelilor de sub banda neagră înmulțită cu 3 pentru a compensa diferențele de mărime a zonelor.

*Capitolul 3 - Componente Hardware folosite*:

**Figure 4**. Haar-Like Features

Componentele necesare realizării acestui proiect sunt:

● **Raspberry Pi** – Un microprocessor de tip SoC, utilizat pentru programarea algoritmilor de extragere a numărului de înmatriculare și recunoașterea optică a caracterelor.

● **Camera Module** – O camera folosită la captarea numerelor de înmatriculare, care ulterior vor fi preprocesate.

● **Monitor** – Ecran folosit pentru partajare prin intermediul VNC Viewer.

## Configurarea mediului de lucru

Primul pas pentru configurarea mediului de lucru al proiectului este încărcarea sistemului de operare pe un card micro SD, cu memorie de minim 8GB. Acesta va fi ulterior introdus în Rasberry, iar bootarea se va face automat odată cu conectarea plăcuței la priză. Procesul de încărcare a sistemului de operare pe card, este unul destul de simplu și necesită descărcarea software-ului Raspberry Pi Imager, în care va trebui sa încărcăm OS Image file-ul cu sistemul pentru care optăm. În cazul nostru, am ales Windows 10, deoarece a fost lansat recent pentru versiunea de plăcuță cu care lucrăm (Raspberry Pi 4 Model B 8GB RAM).

Următorul pas este să conectăm Raspberry-ul la un monitor, prin intermediul unui cablu

HDMI-mini HDMI pentru a rula sistemul de operare. De asemenea, am conectat un mouse, o tastatură și camera web în porturile USB ale acesteia. Deoarece această versiune de Windows este încă în stadiu de testare, aceasta nu este prevăzută cu conexiune WiFi, motiv pentru care suntem nevoiți să conectăm și un cablu de Ethernet în portul LAN.

Urmează instalarea VNC Viewer-ului pentru a ne putea conecta remote de pe oricare alt computer si a softurilor folosite pentru rularea codului în Python.

## 

# *Capitolul 4 - Testare si probleme intampinate:*

## Claritatea imaginilor/video:



**Figură 3.** Exemplu de nr. de înmatriculare cu rezoluție acceptabilă

Distanța dintre camera și vehiculul ce urmează a fi detectat, nu ar trebui să depășească distanța ce permite captarea vizibilă a numarului de înmatriculare.

● Poziția camerei, luminozitatea și claritatea of lentilelor camerei sunt principalii factori

de care trebuie să avem grijă pentru obținerea ieșirii dorite.

● Raspberry Pi-ul necesită o tensiune de alimentare de 5.1V printr-un cablu micro USB.

● Raspberry Pi-ul trebuie așezat într-un loc în care performanțele sale nu sunt afectate de

factorii externi ai vremii.

## 3.2. Probleme software (sisteme de operare folosite)

# *Capitolul 5 - Concluzii*:

Scopul acestui proiect este de a arăta că tehnologiile gratuite și open source sunt suficient de bine închegate pentru utilizarea lor în domeniile de calcul științifice. Sistemul funcționează satisfăcător pentru variații mari ale condițiilor de mediu și diferite tipuri de plăcuțe de înmatriculare întâlnite în întreaga lume.

În urma realizării acestui proiect, aș putea propune câteva idei de îmbunătățiri sau de alte multiple întrebuințări.

● Sistemul poate fi utilizat la intrarea în curți și îmbunătățit folosind senzori la

intrare. Pe măsură ce un vehicul intră pe poartă, se va capta imaginea plăcuței sale de înmatriculare. Dacă plăcuța de înmatriculare este cunoscută, atunci poarta se va deschide, altfel va trimite o alertă.

● ANPR poate fi exploatat în continuare pentru identificarea proprietarului

vehiculului, identificarea modelului vehiculului pentru controlul traficului, controlul vitezei

și localizarea acestuia.

● Poate oferi diverse avantaje, cum ar fi îmbunătățirea siguranței traficului,

a securității, ușor de utilizat, accesare imediată a informațiilor - comparativ cu căutarea manuală a detaliilor de înmatriculare a proprietarului vehiculului și rentabilă pentru orice țară.

● Cea mai mare parte a ANPR se concentrează pe prelucrarea unei singure

plăcuțe de înmatriculare a vehiculului, dar în timp real pot exista mai multe plăcuțe în timp ce imaginile sunt capturate.

● Poate fi extins în continuare ca ANPR multilingv pentru a identifica limba

caracterelor automat pe baza datelor de instruire.

# Bibliografie

|  |  |
| --- | --- |
| [1] | „Pytesseract,” [Interactiv]. Available: https://pypi.org/project/pytesseract/. [Accesat 9 Iunie 2021]. |
| [2] | „OpenCV Introduction,” 2021. [Interactiv]. Available: https://docs.opencv.org/master/d1/dfb/intro.html. [Accesat 11 Iunie 2021]. |
| [3] | „OpenCV Cascade Classifier Training,” [Interactiv]. Available: https://docs.opencv.org/2.4/doc/user\_guide/ug\_traincascade.html. [Accesat 11 Iunie 2021]. |

Anexa A – codul de Alerta Email

Anexa B – codul de recunoastere a carcaterelor din imagini

Anexa C – codul de extragere a nr de inmatriculare

Anexa D – codul de implementare a bazei de date si detectare a numerelor