



UNIVERSITATEA DIN BUCUREŞTI
FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ
SPECIALIZAREA INFORMATICĂ

Sistem de asistență în trafic

LUCRARE DE LICENȚĂ

COORDONATOR ȘTIINȚIFIC
CONF. DR. MARIUS POPESCU

ABSOLVENT
ADRIAN ISPAS

BUCUREŞTI, ROMÂNIA

IUNIE 2017

Abstract

Prezenta lucrare de licență propune abordarea unei teme de interes major în ultima perioadă de timp și anume implementarea de sisteme de asistență în trafic pentru mașini.

Prin noțiunea de sistem de asistență în trafic se înțeleg următoarele:

1. Detectarea benzi curente de circulație a mașinii;
2. Detectarea mașinii de pe banda curentă de circulație;
3. Estimarea distanței față de mașina detectată pe banda curentă de circulație;
4. Estimarea vitezei relative a mașinii de pe banda curentă de circulație față de mașina din care se întregistrează video-ul.

Lucrarea prezintă în capitolul de evaluare experimentală rezultate din punct de vedere cantitativ și calitativ. Arătăm că aplicația detectează mașinile și benzile cu o acuratețe foarte mare, în jur de 90%, putând rula în diferite condiții de iluminare.

Din prisma implementării, aplicația a fost dezvoltată în Matlab și rulată pe un sistem dotat cu procesor Intel i7 Quad Core, 2.60 GHz și 8 GB RAM. Biblioteca OpenCV a fost inclusă în Matlab și utilizată în diverse circumstanțe la care au fost adăugate și funcții din biblioteca VLFeat.

Din punct de vedere al vitezei de rulare a întregului proces, a fost obținut un timp mediu de rulare pe frame de 0.05 secunde pe video-uri de 360p, cea ce înseamnă o rulare în timp real a întregii aplicații cu o capacitate de procesare de până la 30 de frame-uri pe secunde.

Abstract

This bachelor's thesis proposes to approach a topic of major interest in the last period of time, namely the implementation of traffic assistance systems for cars.

The notion of a traffic assistance system means the following:

1. Detection of current lane of the car;
2. Detecting the car on the current lane;
3. Estimating distance from the car detected on the current lane;
4. Estimating the relative speed of the car on the current lane to the car from which the video is recorded.

The paper presents quantitative and qualitative results in the experimental evaluation chapter. We show that the app detects cars and lanes with a very high accuracy, around 90%, and can run under different lighting conditions.

From the implementation point of view, the application was developed in Matlab and run on an Intel i7 Quad Core, 2.60 GHz processor and 8 GB RAM. The OpenCV library was included in Matlab and used in various circumstances to which functions from the VLFeat library have been added.

The runtime of the entire process has achieved a mean runtime per frame of 0.05 seconds on 360p videos, which means real-time running of the entire application with a processing capacity of up to 30 frames per second.

Cuprins

Listă de figuri	7
Listă de tabele	9
1 Introducere	11
1.1 Motivație	11
1.2 Obiective propuse	12
1.3 Actualitate	13
1.4 Structura lucrării	14
2 Fundamente teoretice	17
2.1 Mașini cu vector suport	17
2.1.1 Noțiuni generale	17
2.1.2 Mașini cu vector suport multiclassă	18
2.1.3 Mașini cu vector suport pentru regresie	20
2.2 Histograme de gradienți orientați	22
2.2.1 Noțiuni generale	22
2.2.2 Etapele metodei	24
2.3 Metoda glisării ferestrei	25
2.3.1 Noțiuni generale	25
2.4 IPM (Inverse Perspective Mapping)	26
2.4.1 Noțiuni generale	26
3 Dezvoltarea aplicației	29
3.1 Detectare bandă	29
3.2 Detectare mașină	35

3.3 Determinare distanță	37
3.4 Determinare viteză relativă	37
4 Evaluare experimentală	41
4.1 Rezultate benzi circulație	41
4.2 Bază de date benzi circulație	42
4.3 Rezultate detectie mașină	43
4.4 Bază de date mașini	45
Bibliografie	49

Listă de figuri

1.1	Dinamica accidente rutiere	11
1.2	Obiective propuse	12
1.3	Acoperire sistem Tesla Autopilot	13
1.4	Sistem Waymo	14
2.1	Mașini cu vector suport	18
2.2	Mașini cu vector suport pentru regresie	20
2.3	Mașini cu vector suport pentru regresie 2	21
2.4	Mașini cu vector suport pentru regresie 3	22
2.5	Descriptor HOG	23
2.6	Procesul de extragere a histogramelor de gradienți orientați	23
2.7	Metoda glisării ferestrei la nivel de concept	26
2.8	Metoda glisării ferestrei în practică, în patru poziții diferite	26
2.9	Modelul IPM	27
2.10	Imagine IPM	27
2.11	Reprezentare grafică IPM	28
3.1	Zonă interes imagine	30
3.2	Transformare IPM	31
3.3	Transformare IPM în practică	31
3.4	Imagine IPM filtrată	32
3.5	Linii din semnal	33
3.6	Determinare scor linie	34
3.7	Regiune interes detectie mașină	36
3.8	Determinare distanță și viteza	38
4.1	Detectie bandă - exemple reușite	42

4.2 Detectie banda - exemple nereusite	42
4.3 Imagine Caltech Lanes Dataset	43
4.4 Grafic precizie - recall detectie masină	44
4.5 Detectie masini - exemple reusite	44
4.6 Detectie masini - exemple nereusite	44
5.1 Detectie linii curbe	48

Listă de tabele

4.1 Rezultate evaluare detecție bandă	41
---	----

Capitolul 1

Introducere

1.1 Motivație

Anual se produc peste 8 000 de accidente rutiere grave în România, din care rezultă peste 7 000 de răniți grav și peste 2 000 de persoane decedate. Principalul motiv al tuturor acestor incidente, este de departe reprezentat de erorile umane, de la neatenție sau lipsă de experiență până la oboseală.

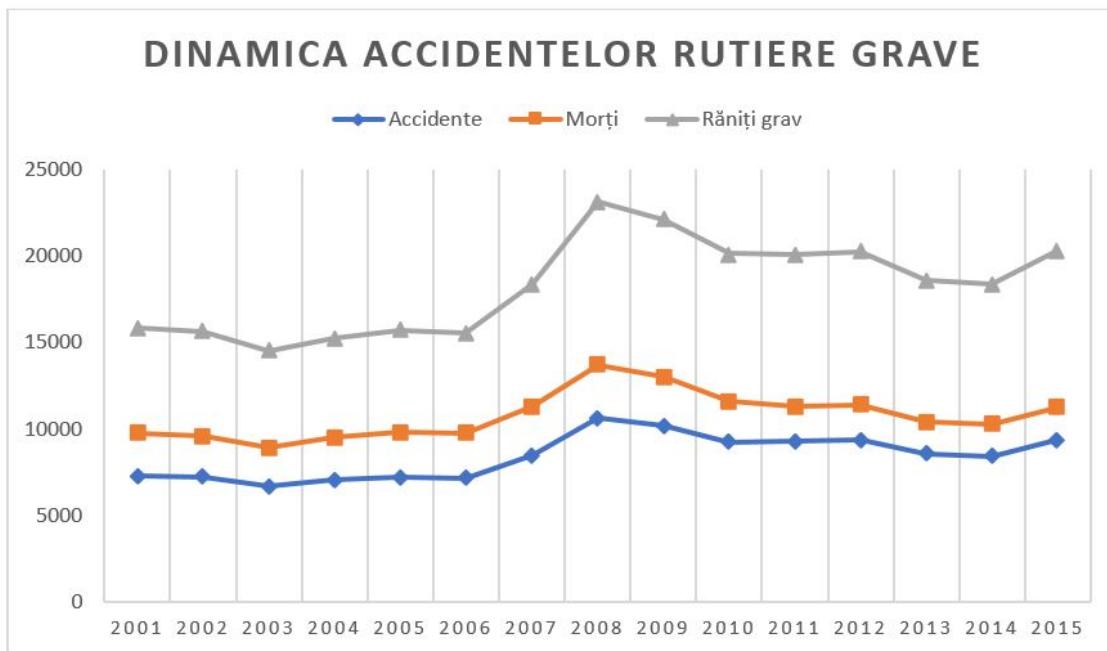


Figura 1.1: Dinamica accidentelor grave produse în România în perioada 2001 - 2015. Statistică preluată din [8].

Însă, o foarte mare parte a erorilor ce cauzează producerea de accidente rutiere pot

fi evitate cu ajutorul sistemelor dotate cu inteligență artificială. Sisteme ce pot fi de la cele mai elementare lucruri, precum recunoașterea benzii de circulație până la un sistem complet care poate anticipa și preveni anumite evenimente iminente pe care o persoană le-ar sesiza și evita mult mai încet din punct de vedere al timpului de reacție.

Conform raportului The Atlantic, accidentele rutiere ar putea fi reduse cu până la 90% în jurul anului 2050 datorită sistemelor de asistență cu care sunt dotate actualele mașini dar și sistemele ce se vor afla în dotarea viitoarelor generații de mașini [17].

Abordarea unei teme de actualitate cu implicații majore în prezent și probabil mult mai intense în viitor, abordarea unei teme ce își propune să vină în ajutorul umanitatii din prisma faptului că are capacitatea de a scădea semnificativ rata accidentelor rutiere produse ca urmare a greșelilor menționate mai sus, au reprezentat principalele motive ale abordării acestei teme.

1.2 Obiective propuse

Prezenta lucrare de licență are drept obiective principale detecția benzii curente de circulație pe care se află mașina, iar pe baza acestei detecții, căutarea eventualei mașini ce se află pe această bandă. Acestor două obiective principale li se alătură alte două obiective secundare și anume determinarea distanței față de eventuala mașină aflată pe banda detectată și determinarea vitezi relative de deplasare a mașinii curente raportată la mașina detectată ce se află în față acesteia pe banda de circulație aferentă.

Figura 1.2 ilustrează un exemplu practic al obiectivelor propuse.



Figura 1.2: Obiective propuse

1.3 Actualitate

În ceea ce privește actualele sisteme de asistență în trafic, cele dezvoltate de Tesla și Waymo (Google) se numără printre cele mai complexe și complete.

Tesla Autopilot

Sistemul oferit de Tesla vine în dotare cu 8 camere ce oferă o vizibilitate de 360 de grade până la o distanță de 250 de metri. La acestea se adaugă alți 12 senzori ultrasonici cu rolul de a completa vizibilitatea oferită de cele 8 camere. Sistemul dispune și de alți senzori cu rolul de a oferi în plus informații esențiale în detectarea obiectelor. Un radar amplasat în partea frontală a mașinii oferă și mai multe informații ceea ce îi oferă capabilitatea de a vedea prin ploi ambidente, ceată, praf, chiar și prin mașina din față. [16]



Figura 1.3: Acoperire sistem Tesla Autopilot. Imagine preluată din [16].

Waymo

Mașinile Waymo sunt dotate cu senzori și sisteme capabile să detecteze zonele pietonale, bicliști, alte autovehicule și multe altele pe distanțe similare cu a două terenuri de fotbal. [20]

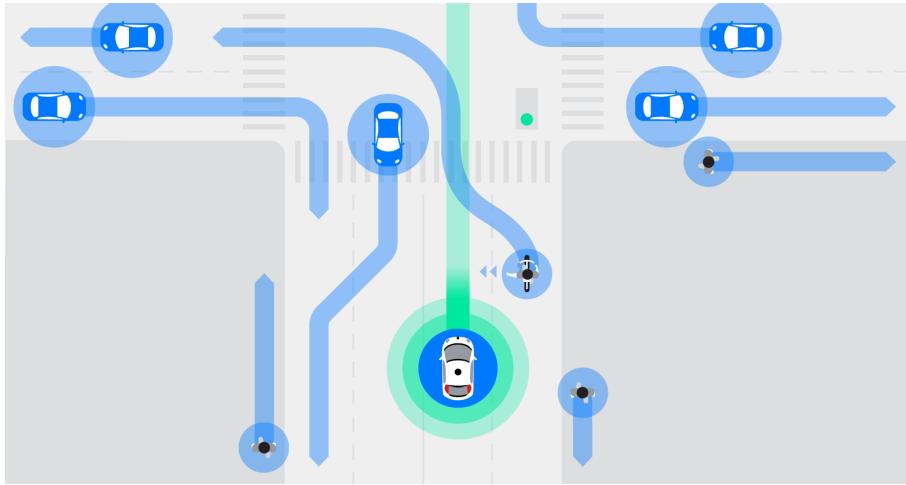


Figura 1.4: Sistem Waymo. Imagine preluată din [20].

1.4 Structura lucrării

Prezentarea lucrării debutează prin detalierea fundamentelor teoretice în capitolul II. Aici menționăm fundamentele teoretice ce stau la baza realizării prezentei lucrări de licență. Capitolul se deschide prin prezentarea noțiunilor despre mașini cu vector suport (SVM), un element esențial în antrenarea detectorului de mașini. Pe parcursul acestui capitol prezentăm noțiunile generale despre mașinile cu vector suport, noțiuni despre mașinile cu vector suport multiclassă dar și despre mașinile cu vector suport pentru regresie.

Continuăm prin a descrie histogramele de gradienți orientați (HOG) folosite în extragerea de caracteristici din imagini. Aceste caracteristici sunt trimise mașinilor cu vectori suport pentru antrenarea dar sunt folosite ulterior și pentru testarea și validarea potențialelor zone ce conțin mașini. Prezentăm pe lângă noțiunile generale și etapele algoritmului de extragere a caracteristicilor prin această metodă.

Metoda glisării ferestrei este și ea abordată în fundamentarea teoretică, această metodă este cea care stă la baza identificării zonelor din imagine pentru care detectorul produce scoruri pozitive în ceea ce privește detectia de mașini.

În finalul acestui capitol prezentatăm noțiunea de IPM, noțiune esențială în detectarea benzii de circulație, pe de o parte, dar și în analiza datelor privind distanța față de mașina aflată pe banda curentă și de viteza relativă a acesteia în maniera abordată de lucrare.

Capitolul III este dedicat procesului propriu-zis de dezvoltare al aplicației. Începe prin prezentarea componentei ce se ocupă de detectarea benzii. Continuăm prin prezentarea

componentei care determină pozițiile eventualelor mașini din imagini și încheiem prin prezentarea metodelor abordate de detectare a distanței față de mașina din față pe banda curentă de circulație împreună cu viteza relativă în raport cu ea.

Spre finalul lucrării, în capitolul IV, sunt prezentate rezultatele propriu-zise dar și bazele de date utilizate atât în cadrul componentei ce se ocupă de detectarea benzii de circulație cât și a componentei ce se ocupă de detectarea de mașini. Pentru fiecare dintre acestea sunt prezentate informații generale despre respectiva bază de date, tipurile de imagini pe care le conține și algoritmii de evaluare utilizați pentru validarea rezultatelor obținute.

În închiderea lucrării sunt trase concluziile finale și sunt prezentate posibile viitoare direcții de dezvoltare ale curentei aplicații.

Capitolul 2

Fundamente teoretice

2.1 Mașini cu vector suport

2.1.1 Noțiuni generale

Mașinile cu vector suport au devenit populare în ultimii ani pentru rezolvarea problemelor de clasificare sau regresie. Acestea realizează o clasificare de tip două clase prin folosirea modelului liniar de forma

$$y(x) = w^T \phi(x) + b \quad (2.1)$$

unde ϕ este o transformare a spațiului caracteristicilor.

Datele de antrenare sunt alcătuite din N vectori de intrare x_1, \dots, x_N fiecare dintre aceștia având asociate etichetele t_1, \dots, t_N din mulțimea $t_n \in \{-1, 1\}$, iar rezultatul din $y(x)$ reprezintă scorul asociat vectorului de intrare x .

De asemenea, în momentul antrenării, setul de date trebuie să fie liniar separabil în spațiul caracteristicilor ceea ce înseamnă că pentru orice alegere a parametrilor w și b dacă funcția $y(x_n) > 0$ atunci eticheta asociată trebuie să fie $t_n = +1$. Situația este similară și pentru cazul în care $y(x_n) < 0$, atunci eticheta asociată trebuie să fie $t_n = -1$. Concluzionând cele două proprietăți menționate anterior putem afirma faptul că următoarea inecuație trebuie să fie mereu validă $t_n y(x_n) > 0$.

Mașinile cu vector suport abordează problema prin conceptul de „margine” ce este definită ca fiind cea mai mică distanță dintre o frontieră de decizie și orice exemplu primit. O ilustrație a acestui fapt se poate vedea în figura 2.1.

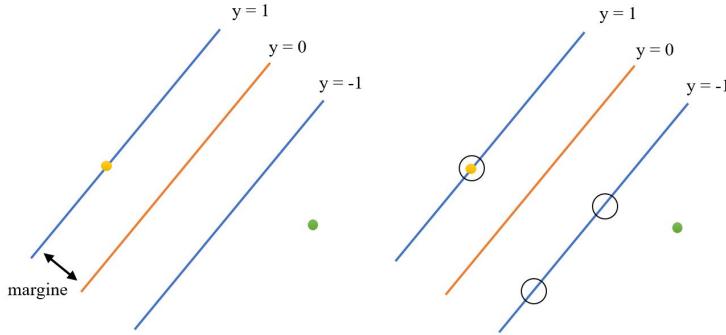


Figura 2.1: Marginea este definită ca distanța perpendiculară dintre limita de decizie și cel mai apropiat punct, după cum se poate observa în figura din stânga. Maximizăm limita marginii precum în figura din dreapta. Locația acestei limite este determinată de mașinile cu vectori suport indicate prin cercuri.

În cazul mașinilor cu vectori suport, limita de decizie este aleasă să fie drept locația unde „marginea” este maximă. Această decizie este motivată de folosirea „teoriei computaționale de învățare” cunoscută drept „teoria statistică de învățare”.

Primul model de distribuție peste vectorii de intrare x pentru orice clasa folosește densitatea Parzen cu un nucleu Gaussian având parametrul comun σ^2 . Utilizarea limitei optime determină cel mai bun hiperplan prin minimizarea probabilității de eroare relative densității modelului învățat.

În cazul limitei $\sigma^2 \rightarrow 0$, hiperplanul optim este cel care are marginea maximă. Intuitiv, deoarece σ^2 este redus, hiperplanul este dominat mai mult de punctele din apropiere decât de cele din depărtare, în limită hiperplanul devenind independent de datele ce nu sunt mașini cu vector suport [8].

2.1.2 Mașini cu vector suport multiclasă

În principiu, mașinile cu vector suport sunt clasificatori de tip două clase însă, în practică întâlnim multe cazuri în care avem nevoie de probleme ce presupun mai mult de două clase. De-a lungul timpului au fost propuse diferite variante de rezolvare a problemei construcției mașini cu vector suport multiclasă.

Cea mai cunoscută abordare (Vapnik, 1998) este reprezentată de construcția a K mașini cu vectori suport separate, unde în fiecare al k -lea model, $y_k(x)$, este antrenat folosind date din clasa C_k pe post de date de antrenare pozitive, iar datele din celelalte $K - 1$ clase rămase pe post de date de antrenare negative. Această tehnică este cunoscută

sub numele de „unul-versus-toți”.

Astfel noile predicții pentru un input x vor fi făcute pe baza următoarei relații

$$y(x) = \max_k y_k(x) \quad (2.2)$$

Evident, în această abordare pot apărea anumite probleme precum cea legată de faptul că acești clasificatori au fost antrenați pe diferite clase, ceea ce înseamnă că nu avem neapărat o garanție a faptului că se vor comporta în practică pe tipuri diferite de intrări similar cu datele de antrenare.

O altă problemă întâlnită în această abordare de unu-versus-toți este legată de cantitatea datelor de antrenare în sensul de pierdere al echilibrului dintre exemplele pozitive și cele negative. Spre exemplu, să presupunem că avem 10 tipuri de clase. Pentru o instanță a unei mașini cu vector suport vom avea 10% dintre toate datele de antrenare disponibile încadrate precum exemple pozitive, iar cealaltă parte, mult mai densă, de 90% vor reprezenta exemple de antrenare negative, fapt ce va crea o discrepanță foarte mare între ele, astfel simetria este pierdută.

Weston and Watkins (1999) definesc o singură funcție obiectiv pentru antrenarea tuturor celor K vectori suport mașină simultan, acest procedeu este bazat pe maximizarea marginilor pentru fiecare clasă. Acest lucru produce un efect de rulare înceată a antrenării deoarece rezolvarea a K probleme separate de optimizare ar însemna că pentru fiecare N punct de date are un cost de rulare per total de $O(KN^2)$, iar o singură problemă de optimizare cu dimensiunea $(K - 1)N$ poate fi rezolvată cu un cost total de $O(K^2N^2)$.

Una dintre cele mai întâlnite metode de rezolvare a acestor probleme este o abordare în care vom antrena $K(K - 1)/2$ vectori suport mașină diferenți pentru toate perechile de clase posibile, iar rezultatul final va fi determinat de aceea mașină cu vector suport care va răspunde cu un număr cât mai mare de voturi (scor) dintre toate cele $K(K - 1)/2$ posibilități. Această este numită deseori drept o antrenare „unu-versus-unu”. Evident, în această situație timpul de rulare va crește odată cu numărul claselor pentru care trebuie antrenate mașinile cu vector suport.

Pentru a rezolva această problemă a timpului de execuție putem organiza perechile de clasificare într-un graf aciclic. Astfel, pentru K clase pe care le avem de analizat vom avea în total $K(K - 1)/2$ vectori suport mașină, iar pentru a clasifica o intrare vom avea nevoie doar de $K - 1$ perechi de clasificatori spre a fi evaluati cu clasificatorii specifici traversării grafului.

Deși clasificarea cu mașini vectori suport multiclassă încă are multe capitoare de îmbunătățit, în practică abordarea unul-versus-toți rămâne cea mai utilizată în ciuda limitărilor de performanță aferente [9, 10, 18, 19].

2.1.3 Mașini cu vector suport pentru regresie

În regresia liniară simplă se minimizează o funcție de eroare regularizată definită de

$$\frac{1}{2} \sum_{n=1}^N \{y_n - t_n\}^2 + \frac{\lambda}{2} \|w\|^2 \quad (2.3)$$

Pentru a obține soluțiile, funcția pătratică de eroare va fi înlocuită de o funcție numită ϵ -intensiv - funcție de eroare (Vapnik, 1995) ce ne returnează eroarea 0 dacă diferența absolută dintre predicția $y(x)$ și asocierea sa t este mai mică decât ϵ unde $\epsilon > 0$. Un exemplu simplu de ϵ -intensiv - funcție de eroare cu un cost liniar asociat cu eroarea din exteriorul regiunii este

$$E_\epsilon(y(x) - t) = \begin{cases} 0, & \text{dacă } |y(x) - t| < \epsilon \\ |y(x) - t| - \epsilon, & \text{altfel} \end{cases} \quad (2.4)$$

exemplu ilustrat în figura 2.2.

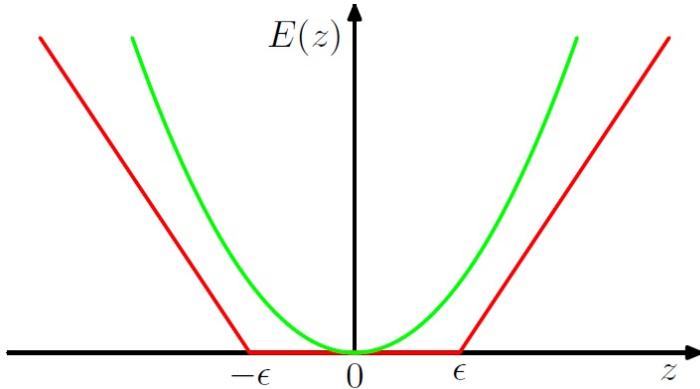


Figura 2.2: Cu roșu se poate observa ϵ -intensiv - funcție de eroare, iar cu verde se poate observa o funcție pătratică de eroare. Imagine preluată din [8].

Prin urmare, vom minimiza o funcție de eroare regularizată dată prin

$$C \sum_{n=1}^N E_\epsilon(y(X_n) - t_n) + \frac{1}{2} \|w\|^2 \quad (2.5)$$

Similar cu cele prezentate anterior putem alege un prag $\xi_n \geq 0$ și $\widehat{\xi}_n \geq 0$, unde $\xi_n > 0$ corespunde punctului pentru care următoare condiție este îndeplinită $t_n > y(x_n) + \epsilon$,

iar $\hat{\xi}_n \geq 0$ corespunde punctului pentru care următoarea condiție este îndeplinită $t_n < y(x_n) - \epsilon$. Acestea este ilustrate în figura 2.3.

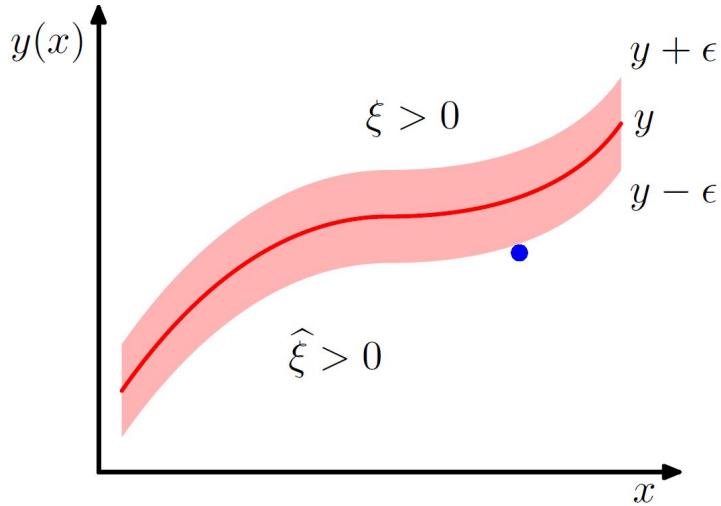


Figura 2.3: Este prezentată o regresie vector suport mașină evidențiind curba de regresie împreună cu zona ϵ – *intensiv*. De asemenea, sunt prezentate variabilele ξ și $\hat{\xi}$. Punctele de deasupra zonei ϵ au $\xi > 0$ și $\hat{\xi} = 0$, iar punctele de sub zona ϵ au $\xi = 0$ și $\hat{\xi} > 0$, în cele din urmă, punctele din zona de margine ϵ au $\xi = 0$ cât și $\hat{\xi} = 0$. Imagine preluată din [8].

Condiția ca un punct să fie în interiorul zonei ϵ este ca $y_n - \epsilon \leq t_n \leq y_n + \epsilon$ unde $y_n = y(x_n)$. Introducând variabilele ξ_n și $\hat{\xi}_n$ permitem punctelor să se afle în afara zonei de interes condiționat de faptul că variabilele introduse sunt nenule iar următoarele condiții sunt indeplinite

$$t_n \leq y(x_n) + \epsilon + \xi_n \quad (2.6)$$

$$t_n \geq y(x_n) - \epsilon - \hat{\xi}_n \quad (2.7)$$

În practică, în detrimentul fixării unui parametru ϵ se preferă fixarea unui parametru v ce are drept rol principal limitarea punctelor din afara zonei ϵ . O rezolvare a unei probleme de regresie având un set de date sinusoidal cu vectori suport mașină este ilustrată în figura 2.4. În această situație parametrii v și C au fost aleși manual, însă în practică aceste valori sunt determinate automat [8].

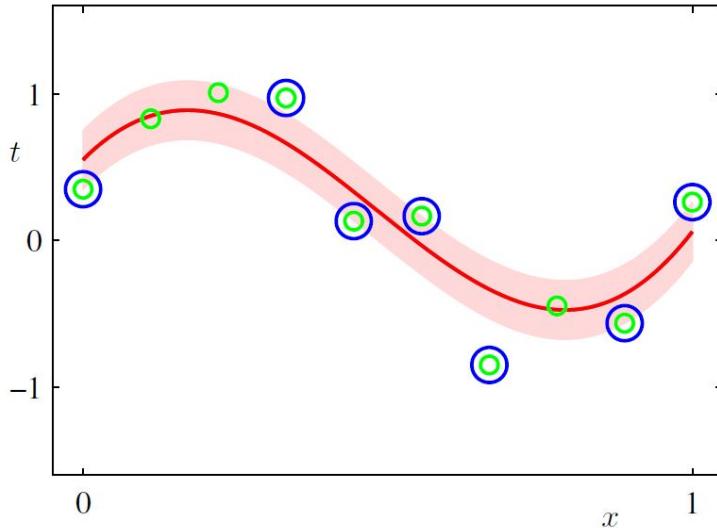


Figura 2.4: Este prezentată un vector suport mașină pentru regresie aplicat pe un set de date sinusoidal și utilizând un nucleu Gaussian. Curba de regresie prezisă este ilustrată prin linia roșie, iar zona $\epsilon - intensiv$ este ilustrată prin zona roșie hașurată. Datele sunt indicate prin cercurile verzi, iar vectorii suport mașină sunt indicați prin cercurile albastre. Imagine preluată din [8].

2.2 Histograme de gradienți orientați

2.2.1 Noțiuni generale

Histograma de gradienți orientați este un descriptor utilizat în vederea artificială și în procesarea imaginilor cu scopul de a detecta obiecte.

Prin această tehnică se numără aparițiile orientării gradientului în anumite porțiuni localizate ale imaginii.

Ideea principală ce stă la baza histogramei de gradienți orientați este reprezentată de faptul că aspectul și forma obiectului dintr-o imagine pot fi descrise prin distribuirea intensității gradienților sau a muchiilor direcțiilor. În figura 2.5 este prezentat un exemplu practic.

Imaginea este împărțită în mici regiuni conectate numite celule, iar pentru pixelii din fiecare celulă se obține o histogramă de gradienți orientați. Descriptorul este rezultatul concatenării acestor histograme. Se poate îmbunătății acuratețea prin normalizarea histogramelor, această normalizare are ca rezultat o invarianță mai bună a schimbărilor intensităților luminii sau apariției umbrelor. În figura 2.6 sunt reprezentate vizual aceste

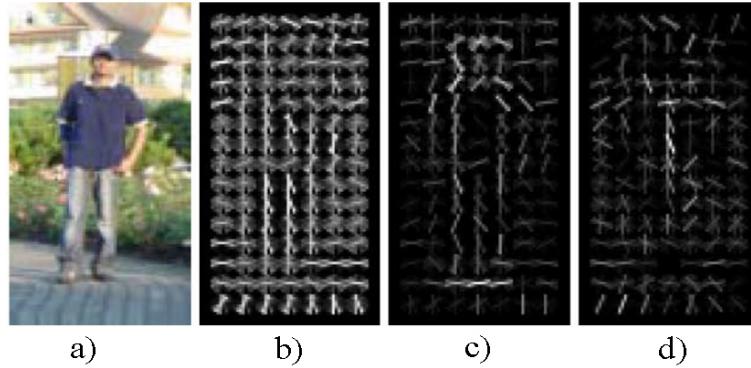


Figura 2.5: (a) imaginea originală; (b) descriptorul histogramei de gradienți orientați asociate imaginii; (c) valorile pozitive din w a descriptorului histogramei de gradienți orientați asociate imaginii; (d) valorile negative din w a descriptorului histogramei de gradienți orientați asociate imaginii. Imagine preluată din [12].

noțiuni.

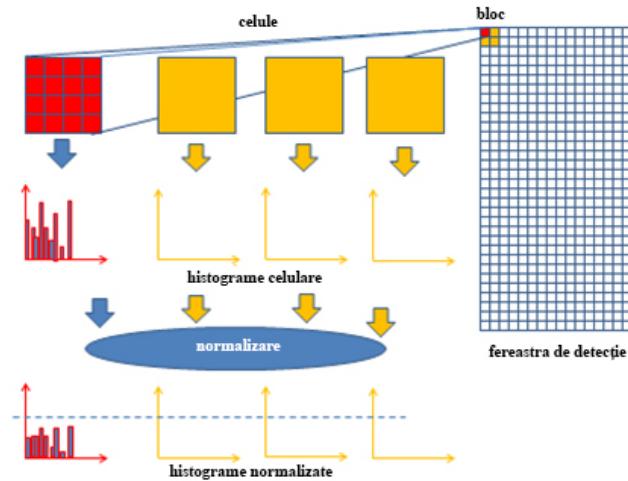


Figura 2.6: Procesul de extragere a histogramelor de gradienți orientați

Histogramele de gradienți orientați prezintă principalul avantaj prin faptul că lucrează la nivel de celulă locală, astfel, în practică, se comportă bine la invarianța geometrică și transformările fotometrice, exceptând, desigur, orientarea obiectului [13].

2.2.2 Etapele metodei

Compunerea gradientului

Principalul pas în majoritatea determinării de caracteristici în preprocesarea de imagini este de a asigura o normalizarea a valorilor asupra culorii. Astfel, primul pas constă în determinarea valorii gradientilor, iar cea mai comună metodă pentru a face acest lucru este să aplicăm un filtru $1 - D$ centrat, atât pe direcția verticală cât și pe direcția orizontală. Pentru acest lucru se pot utiliza următoarele nuclee de filtre

$$[-1, 0, 1] \quad sau \quad [-1, 0, 1]^T \quad (2.8)$$

Orientarea binară

În continuare trebuie create histogramele de celule. Fiecare pixel dintr-o celulă primește un vot ponderat pentru o poziție pe un canal orientat al histogramei. Canalele histogramei sunt distribuite uniform între 0 și 180 de grade sau între 0 și 360 de grade. În practică, pentru a determina contribuția pixelilor se folosește magnitudinea gradientului.

Blocurile descriptor

Datorită diferențelor situației în care iluminarea și contrastul pot varia, este nevoie de normalizarea locală a gradientului, ceea ce presupune gruparea celulelor în blocuri mai mari, iar ulterior concatenarea mai multor blocuri formează descriptorul histogramei de gradienti orientați.

Normalizarea blocului

În ceea ce privește normalizarea blocurilor putem folosi mai multe metode propuse de Dalal și Triggs. În continuare vom presupune că v este vectorul ce nu este normalizat și ce conține histogramele, $\|v\|_k$ reprezintă norma vectorului, pentru $k = 1, 2$ la care adăugăm și o constantă e . Din acestea deducem următoarele forme de normalizare

$$\text{Norma L2 : } f = \frac{v}{\sqrt{\|v\|_2^2 + e^2}} \quad (2.9)$$

$$\text{Norma L1 : } f = \frac{v}{\|v\|_1 + e} \quad (2.10)$$

$$Norma \quad L1 - radical : f = \sqrt{\frac{v}{\|v\|_1 + e}} \quad (2.11)$$

Toate aceste metode de normalizare oferă o îmbunătățire semnificativă făță de datele nenormalizate, iar cea care pare a se comporta puțin mai bine în detrimentul celoralte este $L1 - norm$.

Clasificarea cu suport vector mașină

Pasul final în procesul de detectie folosind histogramele de gradienți orientați este de a antrena clasificatori cu descriptorii obținuți. Clasificarea cu vectori suport mașină poate fi o variantă.

Clasificarea cu rețele neuronale

În cazul în care se dorește o acuratețe de clasificare mai bună se pot folosi rețele neuronale în detrimentul vectorilor suport mașină [12].

2.3 Metoda glisării ferestrei

2.3.1 Noțiuni generale

Tehnica de glisare a ferestrei este una des întâlnită în detectarea de obiecte în imagini. Aceasta presupune deplasarea ferestrei pe toată suprafața imaginii cu un anumit salt de pixel stabilit în prealabil. De asemenea, această tehnică poate fi folosită pentru diferite dimensiuni de detectie. Astfel avem ca alternative, fie să mărim sau să micșoram dimensiunea ferestrei glisante, fie să mărim sau să micșoram dimensiunea imaginii și să rulăm aceeași dimensiune a ferestrei pe diferite dimensiuni ale imaginii [13].

Tehnica prezentată anterior este ilustrată și în figura 2.7 sau în figura 2.8

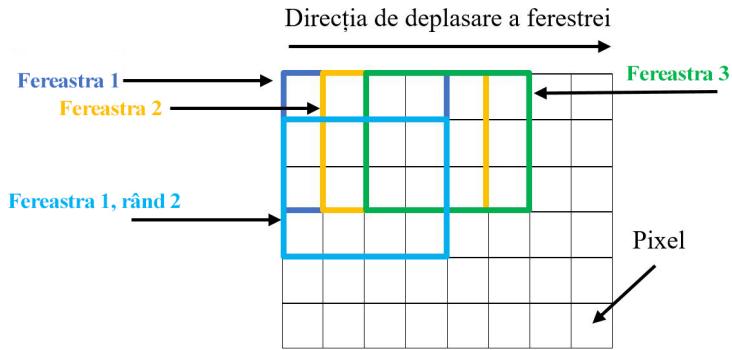


Figura 2.7: Metoda glisării ferestrei la nivel de concept

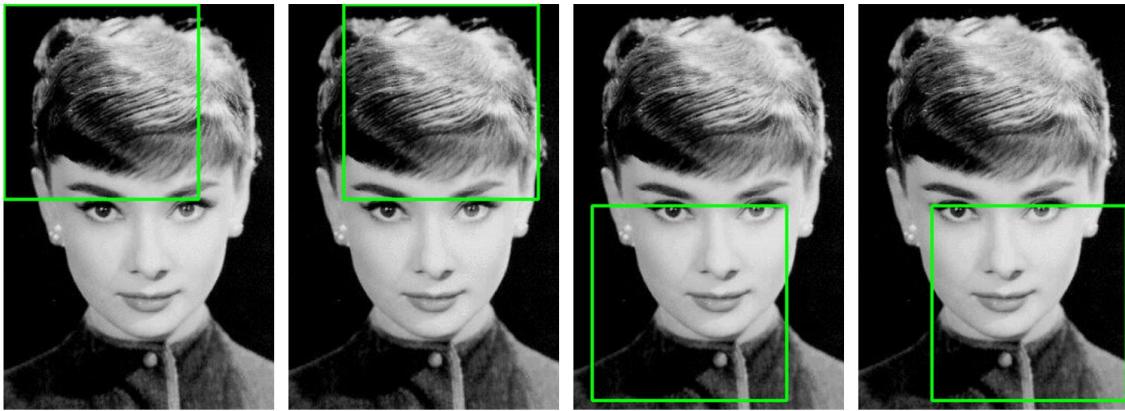


Figura 2.8: Metoda glisării ferestrei în practică, în patru poziții diferite

2.4 IPM (Inverse Perspective Mapping)

2.4.1 Noțiuni generale

IPM-ul reprezintă o tehnică de transformare geometrică care proiectează fiecare pixel a unui obiect din 3D într-o perspectivă 2D. Acest lucru înseamnă repozitionarea fiecărui pixel într-o poziție nouă corespunzătoare noii perspective, astfel construindu-se noua imagine. Din punct de vedere matematic acest lucru poate fi descris printr-o proiecție din planul Euclidian 3D de forma $W = \{(x, y, z)\} \in E^3$ într-un plan 2D de forma $I = \{(u, v)\} \in E^2$. Această transformare mai poartă numele și de „Bird's eye view” datorită privirii de sus pe care o oferă asupra imaginii originale. În figura 2.9 este prezentat vizual acest model.

Pentru aflarea coordonatelor în noul plan putem folosi următoarele formule 2.12 și 2.13 rezultate pe baza modelului IPM.

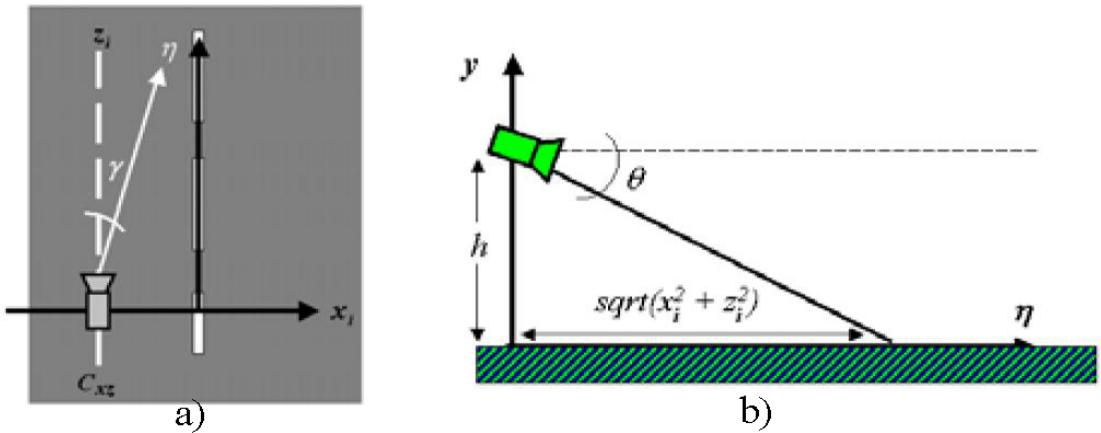


Figura 2.9: a) vizualizarea geometrică „de sus” în planul construit; b) vizualizarea geometrică din lumea reală. Imagine preluată din [1].

$$u(x, 0, z) = \frac{\gamma(x, 0, z) - (Y - \alpha)}{\frac{2\alpha}{n-1}} \quad (2.12)$$

$$v(x, 0, z) = \frac{\theta(x, 0, z) - (\Theta - \alpha)}{\frac{2\alpha}{m-1}} \quad (2.13)$$

În formulele prezentate au fost folosite:

$$\gamma = \tan^{-1}(\frac{z}{x});$$

$$\theta = \tan^{-1}(\frac{h}{\sqrt{x^2+z^2}});$$

Y - reprezintă unghiul dintre proiecția axei optice pe planul real;

Θ - reprezintă unghiul dintre proiecția axei optice și planul orizontal;

α - reprezintă unghiul de înclinare al camerei;

$m \times n$ reprezintă rezoluția imaginii.

În figura 2.10 se poate observa un exemplu practic al IPM-ului.

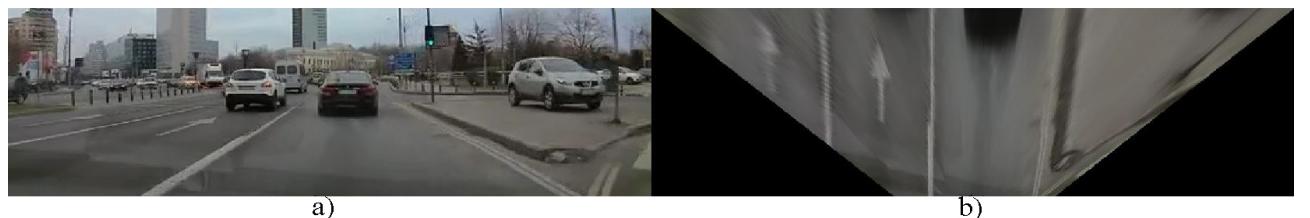


Figura 2.10: Imaginea IPM (b) asociată unei imaginii din plan real (a).

Evident, formulele prezentate anterior prezintă un dezavantaj datorită faptului că procesarea prin intermediul lor se face pixel cu pixel. În elaborarea lucrării practice au

fost folosite metode de obținere a IPM-ului mult mai rapide pentru a îndeplini unul dintre obiectivului lucrării și anume rularea în timp real a aplicației.

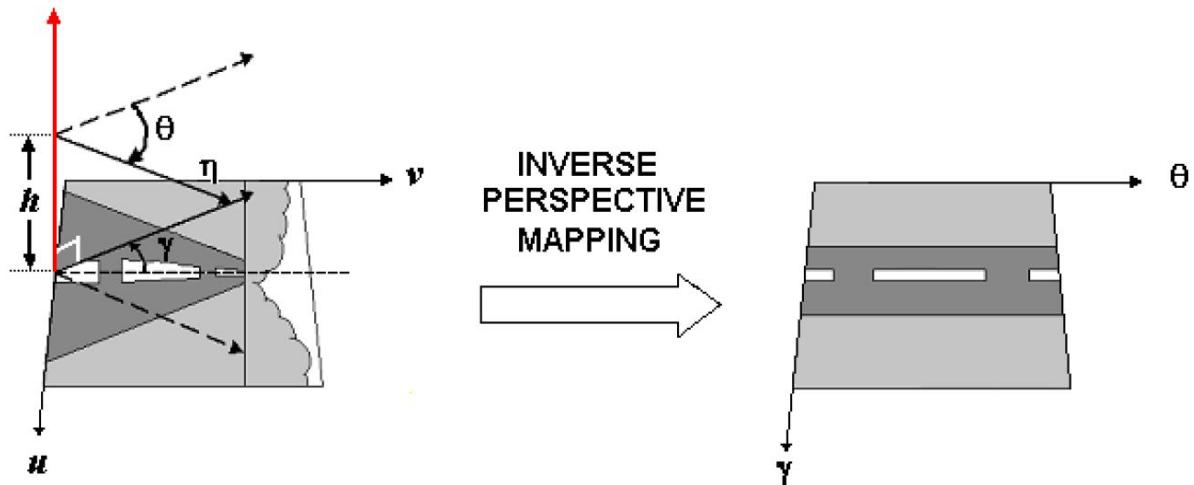


Figura 2.11: Reprezentarea grafică a IPM. Imagine preluată din [1].

Capitolul 3

Dezvoltarea aplicației

3.1 Detectare bandă

Noțiuni introductive

O primă componentă a aplicației de față este ceea cea care se ocupă de detecția benzii curente de circulație. Scopul acesteia este de a identifica marcajele, atât cele continue, cât și cele discontinue, însă fără a face o diferență concretă între ele, iar pe baza acestora indică banda curentă de circulație a mașinii.

Algoritm detectie bandă

Etapa 1: Extragere regiune interes

Această etapă are rolul de a elimina din imagine zonele redundante detectiei [11]. Atât în situația de față, cât și ulterior în cazul detectiei de mașini, ne putem focusa pe o anumită zonă din cadrul frame-ului pentru a detecta banda curentă de circulație și ulterior mașina de pe aceasta. Această zonă este direct determinată de cameră, rezoluția acesteia, amplasarea camerei pe mașină, iar în funcție de toți acești factori se va crea un fișier de configurație specific montajului respectiv cu parametri necesari extragerii zonei de interes.

În figura 3.1 putem observa un exemplu concret.



Figura 3.1: a) Imaginea inițială b) Separația zonei de interes față de cea redundantă c) Imaginea de interes rezultată.

Etapa 2: Obținere imagine IPM

Următoare etapă în procesul de detectie a benzii curente de circulație este cea în care schimba perspectiva de vizualizare a zonei de interes. Prin intermediul unui fișier de configurare se vor da parametri ce vor fi folosiți ulterior în funcția de obținere IPM. Fie A, B, C și D punctele extreme ale imaginii. Redefinim punctele B și C conform figurii 3.2. Cei doi vectori de puncte îi pasăm funcției `cv.getPerspectiveTransform` din biblioteca OpenCV spre a obține matricea M de trecere din perspectiva a), conform figurii 3.2, în perspectiva b). Matricea M este folosită în continuare cu funcția `cv.warpPerspective` pentru a obține imaginea în planul IPM.

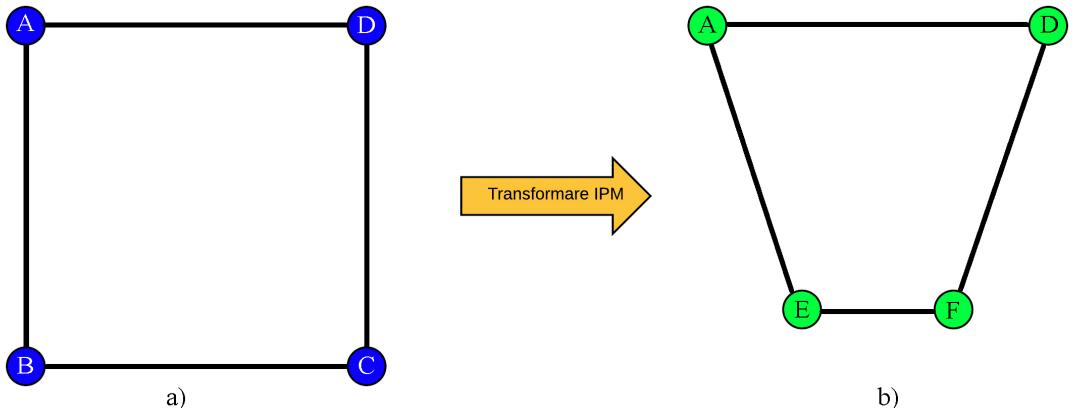


Figura 3.2: a) Cadrul imaginii inițiale b) Cadrul imaginii IPM. Punctele B și C, din cadrul imaginii inițiale, sunt redefinite în punctele E și F.

În figura 3.3 este prezentat un exemplu practic.

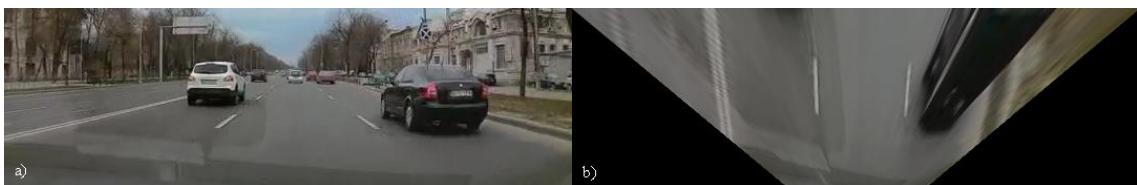


Figura 3.3: a) Imagine zonă interes b) Imaginea IPM asociată zonei de interes.

Etapa 3: Filtrarea imaginii IPM

Filtrarea imaginii IPM are rolul de a păstra în imagine doar zonele cele mai plauzibile de a fi marcaje delimitatorii de bandă și eliminarea conținut nerelevant obiectivului nostru. Noua imagine va fi o imagine binară (alb - negru). Pentru filtrare se folosește filtrul Gaussian 2D orizontal dat de formula

$$d_x \oplus (h \oplus I) = (d_x \oplus h) \oplus I \quad (3.1)$$

$$d_y \oplus (h \oplus I) = (d_y \oplus h) \oplus I \quad (3.2)$$

unde

d_x, d_y - reprezintă filtrele de derivare;

h - reprezintă filtrul Gaussian;

I - reprezintă imaginea.

După aplicarea acestui filtru, matricea rezultată va mai fi filtrată cu un prag, astfel toate valorile din matrice se află sub acel prag vor deveni 0 celelalte păstrându-și intensitatea neschimbată [11].

În figura 3.4 putem vedea o astfel de imagine IPM filtrată.



Figura 3.4: a) Imaginea IPM b) Imaginea IPM filtrată.

Etapa 4: Extragere linii din imaginea filtrată

În continuare ne ocupăm de extragerea potențialelor linii din imaginea filtrată. Pentru aceasta realizăm în primă instanță sumele cumulate pe coloane ale imaginii filtrate [11].

În următorul pas vom aplica un filtru Gaussian, de data aceasta 1D, de forma

$$\frac{\partial I}{\partial x} \quad (3.3)$$

iar pe baza rezultatului generat de acest filtru vom selecta maximele semnalului, în cazul nostru ne vom dori primele 2 maxime.

Etapa 5: Determinare linii finale

Această ultimă etapă este împărțite în alte etape după cum urmează [11].

A. Generează random puncte Această funcție are rolul de a genera un număr prestabilit de puncte din zona celor două maxime găsite la pasul anterior. În aplicația noastră acest număr a fost setat la 10.

B. Potrivire puncte generate Pentru cele n puncte generate, în cazul nostru 10, asignăm o valoare $t_i \in [0, 1]$ pentru fiecare punct $p_i = (u_i, v_i)$. Definim

$$t_i = \frac{\sum_{j=1}^i d(p_j, p_{j-1})}{\sum_{j=1}^n d(p_j, p_{j-1})} \quad \text{pentru } t_i = 1, \dots, n \quad (3.4)$$

d reprezintă distanța euclidiană dată de formula

$$d(p_i, p_j) = \sqrt{(u_i - u_j)^2 + (v_i - v_j)^2} \quad (3.5)$$

În continuare, folosim datele obținute pentru a crea următoarele matrice.

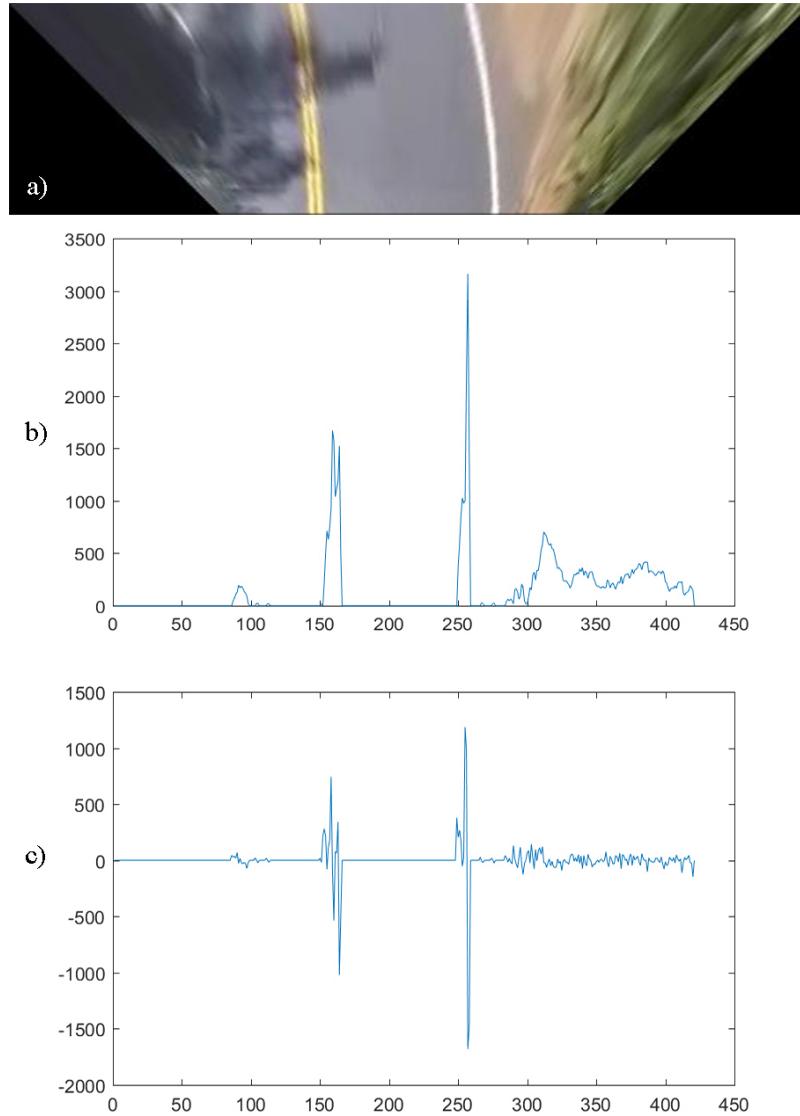


Figura 3.5: a) Imaginea IPM b) Sumele cumulate ale imaginii IPM filtrată pe coloane c) Normalizarea semnalului prin aplicarea filtrului Gaussian.

$$Q = \begin{bmatrix} p_1 \\ \dots \\ p_n \end{bmatrix} \text{ și } T = \begin{bmatrix} t_1^3 & t_1^2 & t_1 & 1 \\ \dots \\ t_n^3 & t_n^2 & t_n & 1 \end{bmatrix}$$

În final obținem punctele după următoarea formulă

$$P = (TM)^+ Q \quad (3.6)$$

unde $(TM)^+$ reprezintă pseudoinversa matricei TM .

C. Determină scorul asociat noilor puncte În final, calculăm pentru fiecare linie generată de punctele procesate un scor ce se bazează pe penalizarea liniilor scurte și/sau

foarte curbate. Scor-ul este obținut pe baza formulei:

$$scor = s(1 + k_1 l' + k_2 \theta') \quad (3.7)$$

unde

s - reprezintă suma pixelilor determinată de linia punctelor;

$l' = (l/v) - 1$ - unde l este lungimea liniei, v este înalțimea imaginii, astfel dacă l' este 0 atunci aveam o linie foarte lungă, iar dacă este -1 aveam o linie foarte scurtă;

$\theta' = (\theta - 1)/2$ - unde $\theta = (\cos\theta_1 + \cos\theta_2)/2$;

k_1 și k_2 - sunt factori de regularizare prestabilități, în cazul nostru $k_1 = k_2 = 0.5$.

În figura 3.6 sunt prezentate vizual componentele necesare determinării scorului.

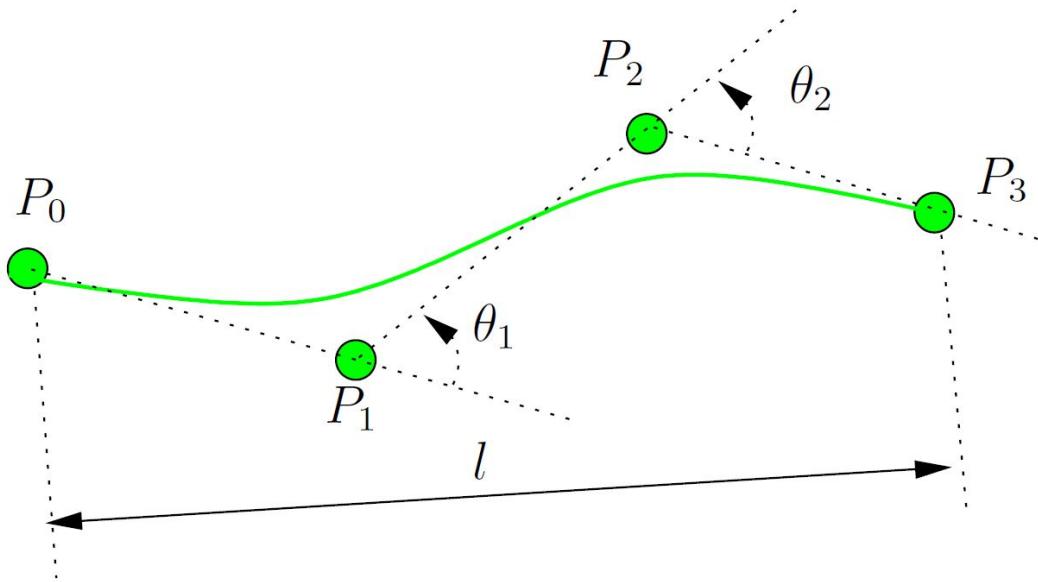


Figura 3.6: Determinare scor linie. Imagine preluată din [11].

D. Determinare cea mai bună linie În această ultimă etapă rulăm de un număr N prestabilit de ori, în cazul nostru 30, pașii 1 - 3 și alegem din aceste N rulări cel mai bun (mare) scor și implicit punctele asociate lui.

Aceste puncte rezultate sunt cele corespunzătoare imaginii IPM. Pentru a le aduce în planul imaginii de interes, imaginea de la care am plecat, avem nevoie de inversa matricei ce a dus punctele în planul IPM, inversă pe care o utilizăm cu funcția din OpenCV `cv.perspectiveTransform`.

În acest punct, toate datele obținute sunt raportate la zona noastră de interes.

3.2 Detectare mașină

Noțiuni introductive

A doua componentă esențială aplicației dezvoltate este cea reprezentată de detectarea de mașini. Aceasta primește de la prima componentă banda de circulație detectată, iar pe baza acesteia încearcă să găsească mașini strict în zona respectivă.

Algoritm detectie mașină

Detectia mașinii, în situația de față, presupune două etape. Prima etapă preliminară este cea de antrenare a clasificatorilor, iar cea de-a doua este detectarea propriu-zisă a mașinii.

Etapa I - Antrenarea clasificatorilor

Extragerea descriptorilor

Primul pas în antrenarea clasificatorilor este reprezentat de extragerea descriptorilor din exemplele pozitive și negative. Acest lucru se va face cu funcția *vl_hog* din biblioteca *VLFeat*.

Antrenarea

Toate exemplele negative vor forma un set de date. Cele patru clase de exemple pozitive vor fi antrenate separat cu setul de exemple negative. Astfel, vom obține patru vectori suport mașină pentru cele patru tipuri de imagini cu mașini. Pentru antrenare se va folosi funcția *vl_svmtrain* din biblioteca *VLFeat*.

Etapa II - Detectarea propriu-zisă

Extragere regiune interes

După cum a fost menționat anterior la detectia benzii curente de circulație vom folosi în continuare aceste informații pentru a elimina zona redundantă căutării mașinii în imagine.

Anticipare poziție mașină

Pentru a mări și mai mult viteza de rulare a programului se poate reduce zona de interes în care este foarte probabil să se afle o mașină cu ajutorul filtrelor de muchii, spre exemplu filtru Sobel pentru muchii orizontale și verticale pe baza următoarelor formule

$$\frac{\partial I(x, y)}{\partial x} \quad \frac{\partial I(x, y)}{\partial y} \quad (3.8)$$

Figura 3.7 ilustrează un exemplu în acest sens.

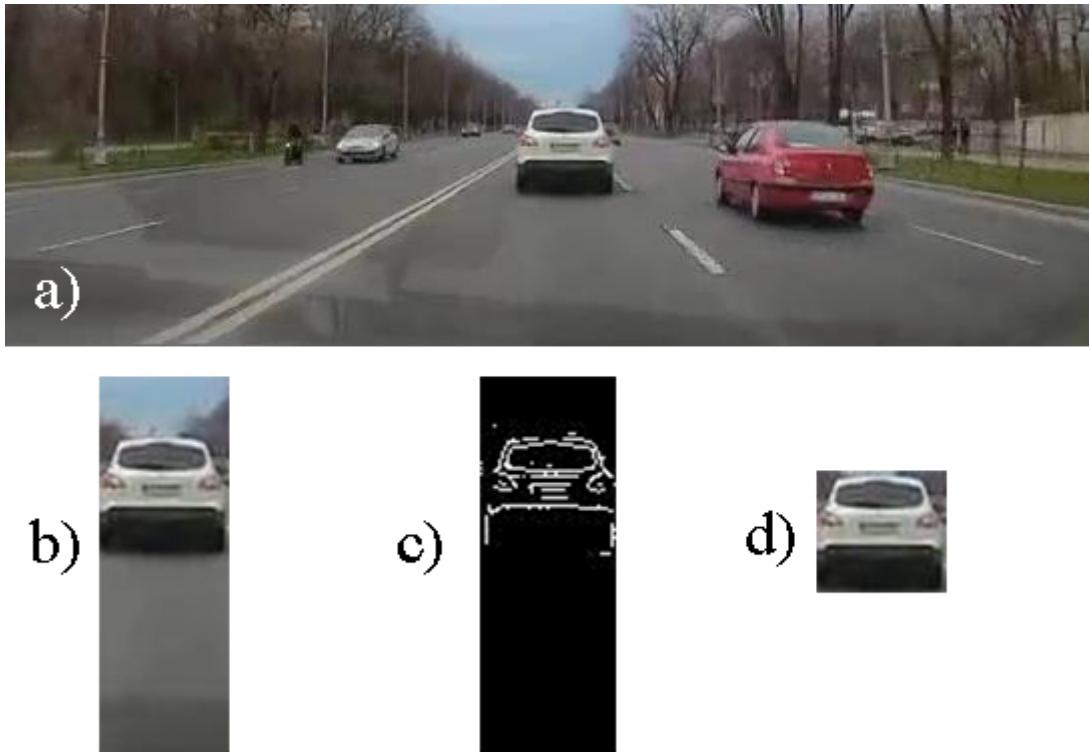


Figura 3.7: a) Imaginea inițială b) Imaginea rezultată pe baza detecției benzii c) Aplicarea filtrelor de muchii d) Extragerea zonei finale de interes.

Rulare detector mașină

Ultima fază din acest proces confirmă sau infirmă existența unei mașini în regiunea extrasă anterior. Pentru aceasta imaginea este mărită cu un factor de scală de 1.2 de 3 ori și micșorată cu un factor de scală 0.9 până când dimensiunea dimensiunea imaginii de analizat ajunge la 31×31 de pixeli, iar pe fiecare dintre aceaste 3 imagini, în cazul primului tip de operație executat asupra imaginii, plus cele n imagini, în cazul celui de-al doilea tip de operație executat asupra imaginii, se aplică metoda glisării ferestrei pentru a extrage caracteristici din imagine cu ajutorul HOG. În final, aceste caracteristici sunt trecute prin cele patru tipuri de mașini cu vectori suport, iar dacă cel puțin unul returnează un scor mai mare sau egal cu pragul stabilit de noi pentru respectiva porțiune, afirmăm că există o mașină pe banda curentă de circulație.

3.3 Determinare distanță

Noțiuni introductive

O primă componentă secundară este cea care determină distanța până la eventuala mașină detectată pe banda curentă de circulație. Figura 3.8 prezintă un exemplu practic.

Algoritm determinare distanță

Această componentă se folosește de detectarea de mașină prezentată anterior. Mai exact, aceasta trece în planul IPM coordonatele centrului detectiei mașinii, iar din acel punct până la baza imaginii care reprezintă în IPM locația mașinii din care este realizat video-ul, se determină cu ajutorul raportării la o configurație 1 metru = n pixeli distanța efectivă în cadrul curent. Configurația este direct determinată de amplasarea camerei și tipul de video produs de aceasta.

Pentru a evita schimbările foarte rapide de distanță, o detectie într-un frame nou este acceptată ca fiind o nouă detectie doar în situația în care diferența euclidiană dintre centrul detectiei curente în planul IPM și centrul detectiei anterioare în același plan depășește un anumit prag stabilit în prealabil.

3.4 Determinare viteză relativă

Noțiuni introductive

O ultimă componentă, în prezenta aplicație, se ocupă de calcularea vitezei relative de deplasare a mașinii din față de pe banda curentă de circulație față de mașina din care este înregistrat video-ul, astfel existând trei situații de răspuns a acestei componente:

- Viteza de deplasare a mașinii din față este mai mare;
- Viteza de deplasare a mașinii din față este similară;
- Viteza de deplasare a mașinii din față este mai mică;

Figura 3.8 prezintă un exemplu practic.

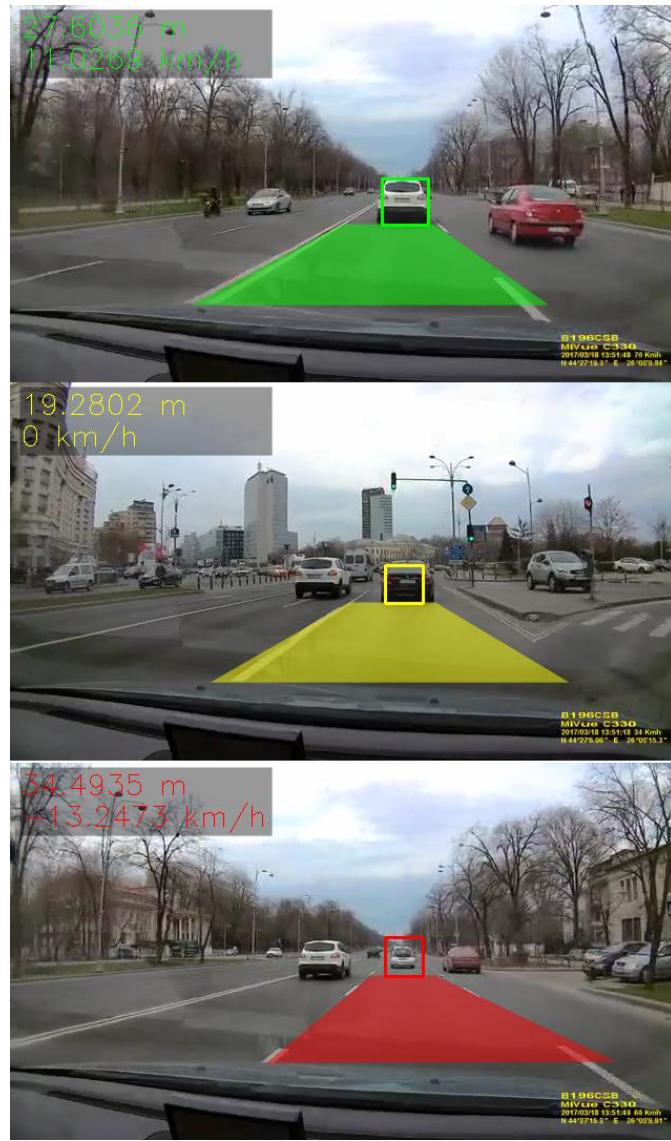


Figura 3.8: a) Distanță 27 m, viteză relativă +11 km/h b) Distanță 19 m, viteză relativă +0 km/h c) Distanță 34 m, viteză relativă -13 km/h .

Algoritm determinare viteză relativă

Precum componenta de detectare a distanței față de mașina din față și această componentă se folosește de detectarea de mașină prezentată anterior. Însă, de data aceasta se folosesc perechi de două frame-uri pentru a determina viteza relativă. Mai exact, aceasta trece în planul IPM coordonatele centrului de detectie a mașinii atât în frame-ul curent cât și în cel anterior, iar diferența dintre aceste două puncte o vom înmulți cu 3.6 pentru a obține kilometri pe oră. Configurația este direct determinată de amplasarea camerei și tipul de video produs de aceasta.

Pentru a evita schimbările foarte rapide de viteză, o detectie într-un frame nou este

acceptată ca fiind o nouă detectie doar în situația în care diferența euclidiană dintre centrul detectiei curente în planul IPM și centrul detectiei anterioare în același plan depășește un anumit prag stabilit în prealabil.

Capitolul 4

Evaluare experimentală

4.1 Rezultate benzi circulație

Aplicația are o precizie în detecție a benzilor de circulație pe video-urile analizate de 92.30% detectii corecte și o rată de detectii false de 7.70%. Rezultatele detaliate pot fi vizualizate în tabelul 4.1.

Video	Total frame-uri	Total linii	Detecții	Detecții corecte	Detecții false
cordova1	250	449	467	91.31%	8.69%
washington1	336	667	662	88.48%	11.52%
washington2	232	448	454	97.10%	2.90%
Total	818	1 564	1 583	92.30%	7.70%

Tabela 4.1: Rezultate evaluare detecție bandă

În continuare pot fi vizualizate diverse exemple reușite, în figura 4.2, iar altele mai puțin reușite, în figura 4.3.



Figura 4.1: Exemplu reușite. Rând 1 - cordova1, 2 - washington1, 3 - washington2.



Figura 4.2: Exemplu nereușite. Rând 1 - cordova1, 2 - washington1, 3 - washington2.

4.2 Bază de date benzi circulație

Informații generale

Partea din aplicație ce are rolul de a detecta banda curentă de circulație a fost dezvoltată plecând de la baza de date Caltech Lanes Dataset de imagini. [4]

Tipuri de imagini

Caltech Lanes Dataset conține 1225 de imagini cu adnotările de rigoare. Baza de date este împărțită în patru tipuri de clipuri și anume: *cordova1* cu un număr total de 250 de imagini, *cordova2* cu un număr total de 406 de imagini, *washington1* cu un număr total

de 337 de imagini și *washington2* cu un număr total de 232 de imagini. Dintre aceste patru seturi de imagini au fost folosite *cordova1*, *washington1* și *washington2*.



Figura 4.3: Exemple imagini din baza de date Caltech Lanes Dataset.

Algoritm de evaluare

Pentru evaluarea rezultatelor a fost folosit soft-ul de evaluare propus de autorul articolului Real time Detection of Lane Markers in Urban Streets [11]. Asupra acestuia au fost aduse modificări în privința citirii etichetelor asociate fiecărui video. Astfel, în cazul lucrării de față, accentul se pune pe banda curentă de circulație. Pentru a extrage doar aceste etichete au fost selectate conform distanței acestora față de centrul imaginii cu o încadrare într-un interval definit în prealabil.

4.3 Rezultate detectie mașină

Precizia medie obținută de aplicație este de 91.30% după cum se poate observa în figura 4.4.

În figura 4.5 și 4.6 sunt prezentate exemple reușite și exemple nereușite.

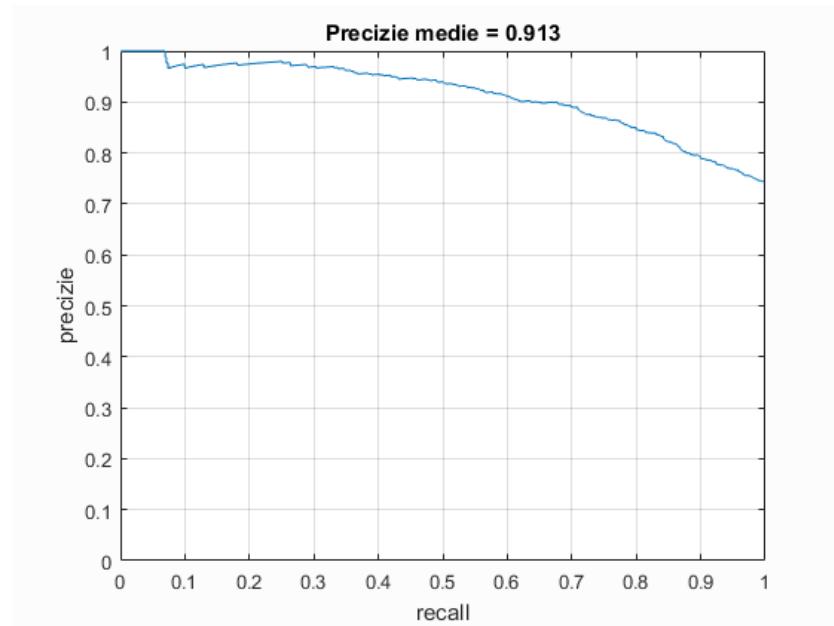


Figura 4.4: Grafic precizie - recall detectie masină

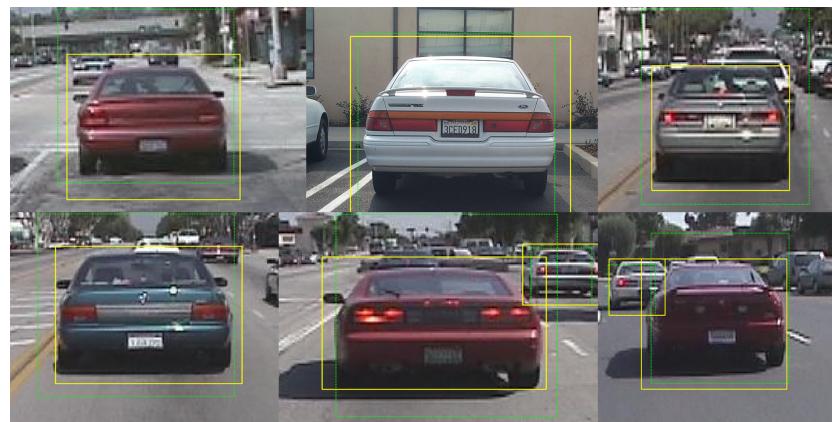


Figura 4.5: Detectie masini - exemple reușite



Figura 4.6: Detectie masini - exemple nereușite

4.4 Bază de date mașini

Informații generale

În cazul componentei ce se ocupă de detectarea mașinii au fost folosite două baze de date. Prima dintre aceasta, cea oferită de Universitatea Politehnică din Madrid și a fost folosită pentru antrenarea clasificatorilor [3].

Pentru testarea a fost folosită o altă bază de date cu imagini, oferită de Universitatea Oxford ce vine în completarea bazei de date oferite de Institutul de Tehnologie din California [2, 5].

Tipuri de imagini

Baza de date oferită de Universitatea Politehnică din Madrid cuprinde două tipuri principale de imagini. Prima categorie conține imagini cu mașini și este împărțită în alte patru subcategori: mașini fotografiate din stânga, mașini fotografiate din dreapta, mașini fotografiate din depărtare și mașini fotografiate din centru. Aceste patru subcategori însumează un număr total de 3 425 de imagini. Aceste imagini provin din diferite condiții, spre exemplu imagini din vreme însorită, înorată, condiții medii, ploaie, iluminat artificial sau iluminare slabă.

A doua parte a acestei baze de date este reprezentată de imaginile de antrenare negative, ce nu conțin mașini. La rândul lor sunt împărțite în cele patru subcategori menționate anterior și însumează un număr total de 3 900 de imagini.

Baza oferită de Universitatea Oxford este formată la rândul ei din alte două baze de date de imagini. Aceste imagini nu au venit însotite de etichete, astfel etichetarea a fost făcută ulterior. Din contopirea celor 2 baze de date a rezultat un set de 987 de imagini cu mașini pentru testare.

Algoritm de evaluare

În ceea ce privește algoritmul de evaluare, fiecare fereastră ce poate conține o mașină oferită de aplicație a fost comparată cu etichetele asociate imaginii respective, iar dacă suprapunere dintre locația indicată de aplicație și locația etichetată se află într-un anumit prag predefinit este acceptată ca detecție adevărată, în caz contrar, ca detecție falsă.

Concluzii

Concluzii generale

Prezenta lucrare a abordat cu succes cele patru componente dezvoltate. Precizia de detecție a fost atât în cazul detectării de bandă de circulație, însă doar în situațiile în care marcajele rutiere tind în general să fie drepte, cât și în cazul detectării mașinii aflate pe respectiva bandă, de peste 90%. Aceste rezultate confirmă abordarea finalizată cu succes a aplicației.

În ceea ce privește determinarea distanței și a vitezei relative putem afirma că este un început acceptabil, însă necesită o dezvoltare mai concretă într-o versiune ulterioară.

Din punct de vedere practic, aplicația poate fi folosită cu rol consultativ. O versiune ulterioară a acesteia cu mențiunile specificate în continuare poate fi chiar conectată la sistemul mașinii.

Direcții viitoare de dezvoltare

Dintre posibilele direcții viitoare de dezvoltare a aplicației curente, putem menționa următoarele.

Diferențiere marcasaj linie

Într-o versiune ulterioară a acestei aplicații se poate implementa un sistem ce are capacitatea de a face diferența dintre liniile discontinue și cele continue, iar în situația în care conducătorul mașinii depășește linia continuă să fie avertizat cu privire la acest aspect.

Determinare viteză de deplasare

O altă posibilă caracteristică adusă prezentei aplicații este reprezentată de determinarea vitezei de deplasare. Astfel pe lângă determinarea vitezei relative ce ne specifică

cu cât mai repede sau cu cât mai încet se deplasază mașina din față, să poate specifica și care este viteza curentă de deplasare a mașinii din care este înregistrat video-ul.

Detectare indicatoare rutiere

De asemenea, detectarea indicatoarelor rutiere este o componentă esențială într-o versiune ulterioară a aplicației curente. Aceasta presupune identificarea tuturor tipurilor de indicatoarelor rutiere și oferirea de avertizamente în funcție de indicatorul rutier detectat și de semnificația acestuia.

Detectarea liniilor curbe

În această variantă preliminară, aplicația are capacitatea de a detecta doar liniile drepte asociate benzii de circulație. Însă, în multe situații se întamplă ca aceste linii să se afle într-o curbă, ceea ce face ca actuala componentă de detectare a benzii de circulație să nu funcționeze la fel de bine. Fapt ce poate fi observat în figura 5.1.

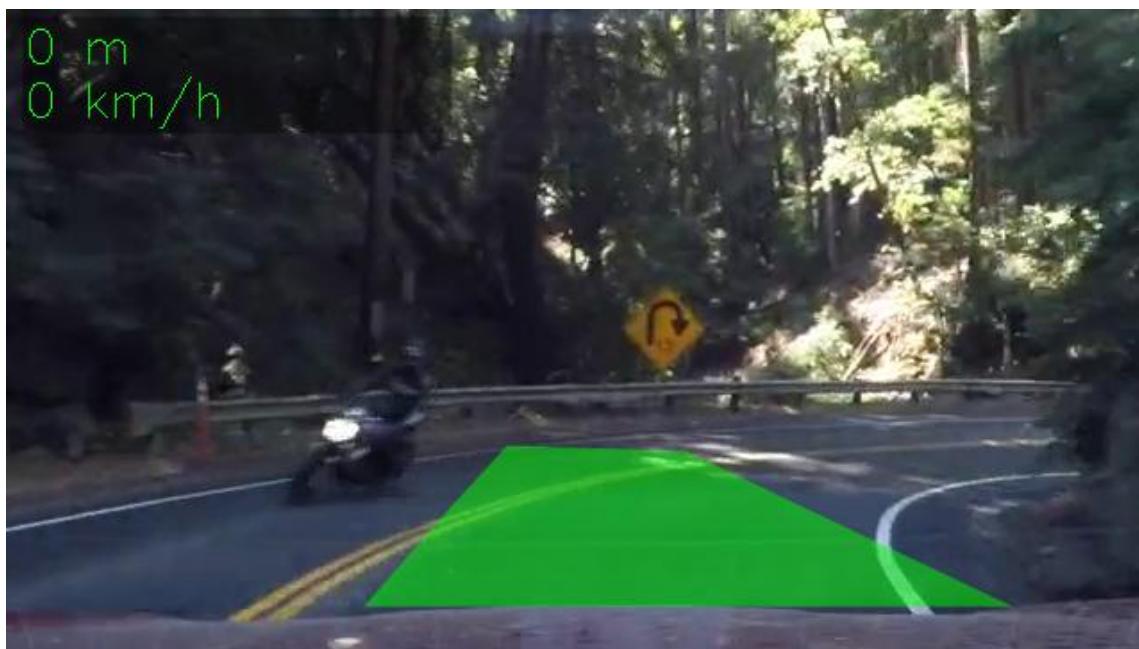


Figura 5.1: Detectie linii curbe

Bibliografie

- [1] Anuar Mikdad Muad, Aini Hussain, Salina Abdul Samad, Mohd. Marzuki Mustaffa, Burhanuddin Yeop Majlis. *Implementation of inverse perspective mapping algorithm for the development of an automatic lane tracking system.* TENCON 2004. 2004 IEEE Region 10 Conference, 207 - 210, 2004.
- [2] Bază de date mașini Universitatea Oxford
<http://www.robots.ox.ac.uk/~vgg/data3.html>
- [3] Bază de date mașini Universitatea Politehnică din Madrid
http://www.gti.ssr.upm.es/data/Vehicle_database.html
- [4] Bază de date benzi de circulație
<http://www.mohamedaly.info/datasets/caltech-lanes>
- [5] Bază de date mașini Institutului de Tehnologie din California
<http://www.vision.caltech.edu/html-files/archive.html>
- [6] Christopher M. Bishop. *Pattern Recognition and Machine Learning.* Springer, New York, 2006.
- [7] Corinna Cortes and Vladimir Vapnik. *Support-Vector Networks.* Machine Learning, 20, 273 - 297:1 - 25, 1995.
- [8] Dinamica accidentelor rutiere
[https://www.politiaromana.ro/ro/structura-politiei-romane/
unitati-centrale/directia-rutiera/statistici](https://www.politiaromana.ro/ro/structura-politiei-romane/unitati-centrale/directia-rutiera/statistici)
- [9] Erin Allwein, Robert Schapire and Yoram Singer. *Reducing multiclass to binary: a unifying approach for margin classifiers.* Journal of Machine Learning Research 1, 113 - 141, 2000.

- [10] Jason Weston and Simon Watkins. *Support Vector Machines for Multi-Class Pattern Recognition*. ESANN, 1 - 6, 1995.
- [11] Mohamed Aly. *Real time Detection of Lane Markers in Urban Streets*. 1 - 6, 2008.
- [12] Navneet Dalal and Bill Triggs. *Histograms of Oriented Gradients for Human Detection*. 1 - 9, 2005.
- [13] Richard Szeliski. *Computer Vision: Algorithms and Applications*. Springer, London, 2010.
- [14] Shane Tuohy, Edward Jones and Martin Glavin *Distance determination for an automobile environment using Inverse Perspective Mapping in OpenCV*. 1 - 6, 2010.
- [15] Sovira Tan, Jason Dale, Andrew Anderson and Alan Johnston. *Inverse perspective mapping and optic flow: A calibration method and a quantitative analysis*. 1 - 13, 2005.
- [16] Tesla Autopilot System
https://www.tesla.com/en_GB/autopilot
- [17] Self-Driving Cars Could Save 300,000 Lives Per Decade in America
[https://www.theatlantic.com/technology/archive/2015/09/
self-driving-cars-could-save-300000-lives-per-decade-in-america/
407956/?utm_source=SFTwitter](https://www.theatlantic.com/technology/archive/2015/09/self-driving-cars-could-save-300000-lives-per-decade-in-america/407956/?utm_source=SFTwitter)
- [18] Thomas Dietterich and Ghulum Bakiri. *Solving Multiclass Learning Problems via Error-Correcting Output Codes*. Journal of Artificial Intelligence Research 2, 263 – 286:1 - 24, 1995.
- [19] Vladimir Vapnik. *Statistical Learning Theory*. Wiley, New York, 1998.
- [20] Waymo System
<https://waymo.com/>