Tehnici avansate de perceptie a mediului folosind Deep Learning si estimatori probabilistici (DEEPSENSE)

Etapa 1 - Analiza cerinţelor, pregătirea datelor şi a reţelelor neuronale

Echipa de cercetare: Radu Dănescu, Diana Borza, Razvan Itu, Andra Petrovai, Andrei Vadan

Cuprins

1. Introducere - rezumatul etapei

Obiectivul principal al acestei etape a fost proiectarea, antrenarea şi evaluarea mai multor soluții de interpretare a imaginilor din trafic folosind rețele neuronale convoluționale (*convolutional neural networks*). Rețelele neuronale convoluționale au atins performanțe foarte bune, apropiate de cele ale oamenilor într-o mulţime de probleme de clasificare automată, dar necesită un set de antrenare foarte mare. De asemenea, antrenarea lor necesită reglajul fin (*fine tuning*) a mai multor hiper-parametri: rata de învățare (*learning rate*), funcțiile de activare utilizate, *weight decay,* utilizarea regularizării, utilizarea nivelurilor (layer) de *dropout* etc.

Având în vedere aceste aspecte, activităţile principale ale acestei etape au fost:

(1) centralizarea bazelor de date existente pentru interpretarea automată a scenelor de trafic pentru a avea o bază de date suficient de mare pentru antrenarea rețele neuronale convoluționale. Toate bazele de date au fost transformate într-un format unic, bazat pe fişiere cu metadate asociate fiecărei imagini. În plus, folosind dispozitive mobile, am capturat şi adnotat (folosind) formatul definit anterior o nouă bază de date cu imagini din trafic.

(2) antrenarea şi reglajul fin a mai multor rețele neuronale convoluționale care au obținut performanţe ridicate în problemele de clasificare de obiecte şi segmentare semantică a scenei.

(3) intergrarea rezultatelor CNN într-un framework probabilistic pentru urmărirea scenei, pentru estimarea parametrilor de poziţie şi viteză ai obiectelor relevante.

Un alt aspect pe care ne-am focalizat a fost monitorizarea atenției şoferului. Pentru aceasta am dezvoltat un framework generic de analiză a clipirilor şi a direcției de privire. Pe baza acestora se poate determina dacă şoferul este atent la drum (direcția de privire) sau dacă manifestă oboseală (frecvența clipirilor). În plus, metoda propusă are aplicații şi în alte domenii, cum ar fi detecţia înşelăciunii (*deceit detection*) sau în psihologie.

Rezultatele obținute în această etapă au fost publicate în două articole la o conferinţă internaţională, respectiv un jurnal.

2. Colectarea imaginilor de trafic, ingineria şi selecţia datelor, adnotare

Pentru realizarea obiectivelor acestui proiect, vom folosi date din două surse principale: baze de date internaţionale disponibile pentru comunitatea de cercetare, care au avantajul că sunt adnotate cu obiecte detectate, distanţe, şi alte rezultate de referinţă, şi date achiziţionate de noi, unde putem adăuga informaţii suplimentare, sau putem acoperi scenarii diferite.

Bazele de date existente au fost create de diferite instituţii de cercetare şi educaţie, de aceea ele vor avea imagini de dimensiuni diferite şi datele de etichetă stocate în formate diferite.

În Tabelul 2.1. se prezintă o comparaţie din mai multe puncte de vedere (dimensiune, scenarii) a bazelor de date existente pentru analiza automată a imaginilor de trafic.

Tabelul 2.1. Prezentare comparativă a bazelor de date cu imagini de trafic

Baza de date	Număr secvenţe	Număr imagini	Mai multe oraşe	Condiţii meteo diferite	Momente diferite ale zilei	Scenarii diferite
KITTI	22	14999	Nu	Nu	Nu	Da
Cityscapes	50	5000	Da	Nu	Nu	Nu
BDD100K	100000	120000000	Da	Da	Da	Da

a. Baze de date pentru detecţia obstacolelor

KITTI

Baza de date KITTI [1] a fost creată de Karlsruhe Institute of Technology din Germania şi oferă diferite seturi de date cu aplicaţii multiple: baza de date pentru obstacole (vehicule, autobuze, pietoni, etc.), segmentare, urmărire obstacole. Baza de date cu obstacole conţine un total de 7481 imagini de antrenare cu vehicule şi 7518 imagini de test, conţinând un total de peste 80.000 de obiecte etichetate.

Udacity

Baza de date oferită de Udacity [2] pentru detecţia de obstacole conţine imagini din traficul rutier împreună cu fişiere de tip “csv” asociate fiecărei imagini. În fişierele text sunt stocate informaţiile despre locaţia obstacolelor în spaţiul imaginii şi sunt definite coordonatele dreptunghiului care le încadrează: coordonatele punctului din colţul stânga sus: (xmin, ymin) şi colţul din dreapta jos: (xmax, ymax) al dreptughiului, numele imaginii, tipul clasei etichetate. Opţional în fişiere text mai este un câmp numit “occluded” care va avea valoarea 0 dacă obstacolul nu este obstrucţionat şi valoarea 1 în caz contrar.

b. Baze de date pentru segmentare semantică

CityScapes

Baza de date CityScapes [3] cuprinde secvenţe din diferite oraşe din Germania şi Elveţia (ex: Aachen, Koln, Dusseldorf, Zurich). Datele sunt din traficul rutier urban în 5000 de imagini şi

cu 30 de clase de obiecte diferite. Imaginile sunt stocate pe 3 canale, de dimensiune 2048 x 1024 pixeli (un raport de 2:1). În imaginile adnotate culoarea clasei drumului este: (128, 64, 128). Baza de date CityScapes conţine 2975 imagini adnotate cu carosabil.

KITTI

Baza de date KITTI [1] adnotată cu zone de drum conţine doar 289 imagini din scene din trafic în oraşul german Karlsruhe. Drumul este marcat folosind culoarea: (255, 0 255) în imaginile adnotate.

Berkeley Deep Drive

Baza de date Berkeley Deep Drive [4] pentru segmentare conţine peste 100.000 imagini etichetate. Dimensiunea imaginilor este de 1280 x 720 pixeli pe 3 canale color (RGB). Imaginile de etichetă sunt de aceeaşi dimensiune, unde fiecare clasă are asignata câte o culoare diferită. Pentru drum, culoarea este: (128, 64, 128). Datele sunt stocate pe 8 biţi, ceea ce înseamnă că intensitatea pe fiecare canal poate avea valori cuprinse între 0 şi 255. Baza de date conţine un total de 6575 imagini adnotate.

c. Definirea unui format standard pentru bazele de date

Format standard pentru segmentare semantica:

Prin folosirea şi definirea unui format standard pentru toate cele 3 baze de date pentru segmentare, am ajuns la un total de 9839 imagini, din care 7871 de antrenare şi 1968 imagini de test (validare). Aceste imagini sunt transformate în matrice de valori reale (numere în virgulă mobilă) scalate în intervalul (0...1), un format mai potrivit pentru antrenarea şi validarea reţelelor neuronale.

Deoarece bazele de date au un raport diferit între înălţimea şi lăţimea imaginilor sursă, am ales folosirea unei dimensiuni standard. Astfel toate imaginile au fost scalate la dimensiunea 256 x 256 pixeli.

Format standard pentru detecţia obstacolelor:

Pentru a putea procesa toate bazele de date pentru obstacole, am ales definirea unei perechi imagine - fişier text. Toate bazele de date existente pentru detecţie de obstacole au fost convertite în formatul comun definit în această secţiune. În cazul în care bazele de date nu dispun de toate informaţiile senzoriale, au fost scrise valori de 0. De asemenea, în această etapă am implementat şi un sistem propriu de achiziţie, cu acelaşi format de date, folosind dispozitive mobile rulând sistemul de operare Android. Achiziţia imaginilor este realizată prin camera foto a dispozitivului, iar restul de date senzoriale sunt preluate din senzorii disponibili pe dispozitivul mobil. Imaginile sunt salvate în format RGB pe 8 biţi.

Fişierele text au următoarea structură:

- timestamp: UNIX timestamp - timpul achiziţionării imaginii
- accX, accY, accZ: acceleraţia gravitațională în jurul axelor x, y şi z
- gyroX, gyroY, gyroZ: rata de rotaţie a dispozitivului în jurul axelor x, y şi z (radiani/sec)
- magX, magY, magZ: rotaţia din senzorul geo-magnetic în jurul axelor x, y şi z
- yaw: unghiul de rotaţie în jurul axei x (rotaţie laterală a vehiculului de-a lungul axei drumului)
- pitch: unghiul de înclinaţie a dispozitivului mobil
- roll: unghiul de rotaţie în jurul axei y (rotaţia dispozitivului mobil pe axa verticală a drumului)
- yawRate: rata de rotaţie în jurul axei x (aceeaşi cu gyroX)
- speedMs: viteza de deplasare (metri / secundă)
- lat, long: coordonata de latitudine şi longitudine obţinută din senzorul GPS

Cele 3 unghiuri de rotaţie (yaw, pitch şi roll) au fost obţinute din valorile senzorului geo-magnetic şi cel de acceleraţie gravitaţională. Acest format de date poate fi accesat şi procesat “offline” pe sisteme desktop.

3. Proiectarea, antrenarea şi evaluarea reţelelor neuronale, şi reglarea fină a hiperparametrilor

Inteligența artificială reprezintă un domeniu activ de interes şi cercetare, mai ales în ultimii ani datorită popularităţii reţelelor neuronale convolutionale şi apariţiei unor metode de a le antrena folosind hardware accesibil (plăci video grafice). În cadrul acestei etape am proiectat şi antrenat o reţea neuronală convoluţională pentru segmentarea semantică a imaginilor din trafic. În această fază am ales segmentarea zonei de drum (“driveable road area”) din imagini. Deşi în această etapă am procesat şi adus la un format comun şi bazele de date pentru detecţie de obstacole, în prima fază am implementat doar partea de detecţie de carosabil, urmând ca în etapele următoare să implementăm o altă reţea neuronală pentru a detecta obstacolele.

Pentru segmentare am ales o arhitectură de reţea de tipul “encoder-decoder”. Am ales folosirea unei reţele U-NET [5] modificată. Prima parte, cea de codificare, are rolul de a extrage informaţii relevante despre trăsături şi caracteristici ale imaginii de intrare, în timp ce reduce dimensiunea spațială (datorită aplicării operaţiilor de convoluţie cu filtre având dimensiune de nucleu variabilă). Informațiile despre localizare sunt astfel pierdute, dar prin introducerea unor legături directe între nivelurile de codificare şi cele de decodificare se va rezolva problema localizării trăsăturilor. Partea de decodificare, prin folosirea operaţiilor de deconvoluţie, are rolul

de a reconstrui harta de segmentare. În reţelele de tip U-NET, partea de decodificare are acelaşi număr de niveluri ca şi cea de codificare (figura 1).

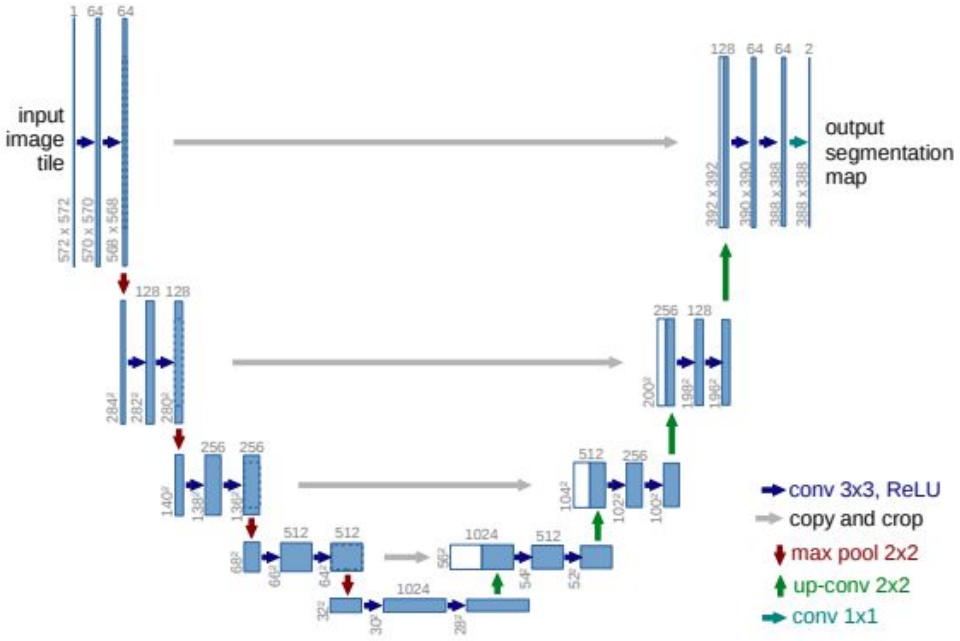

Figura 1. Arhitectura clasica a retelei U-NET. Sursa: https://arxiv.org/pdf/1505.04597.pdf

Reţeaua de tip U-NET folosită în această etapă are ca intrare perechi de imagini de dimensiune 256 x 256 pixeli. Predicţia (ieşirea reţelei), adică imaginea segmentată, va fi de aceeaşi dimensiune. Perechile de imagini de intrare sunt stocate pe disc astfel: imaginea originală (cu scena de trafic) este de tip întreg pe 8 biţi pe 3 canale de culoare, în timp ce imaginea de etichetă (mască) este pe 1 canal pe 8 biţi şi au valoarea intensităţii 255 pentru carosabil, iar 0 pentru restul. Intensităţile din imaginile de etichetă sunt normalizate în intervalul [0, 1] pentru o antrenare mai eficientă. O pereche de imagini de intrare pentru reţea este ilustrată în figura 2.

Figura 2. Date de intrare pentru reţea: stânga imaginea originală, centru: imaginea de etichetă, dreapta: rezultatul aplicării etichetei ca mască pe imaginea originală

În procesul de antrenare am ales minimizarea funcţiei de cost “intersect over union”. Implementarea funcţiei de cost a fost făcută prin utilizarea coeficientul Sorensen-Dice (“dice loss”). Am utilizat o variantă modificată a coeficientului Sorensen, formula de calcul fiind:

$$IoU_{loss} = \frac{2(predicted_{output} \times label)}{predicted_{output} + label}$$

Ecuatia 1. Calculul functiei de cost “IoU” modificata.

Antrenarea unei reţele convoluţionale este facilitată prin utilizarea mai multor date de antrenare. Pentru extinderea bazelor de date existente am ales să aplicăm procesul de augmentare a datelor. Am folosit următoarele tehnici de augmentare: variaţii aleatorii ale intensităţii imaginilor de intrare (în spaţiul de culoare HSV), translatări şi rotaţii aleatorii ale imaginilor, scalarea lor şi oglindirea pe orizontală (“horizontal flipping”). Pentru antrenarea şi predicţia folosind reţele neuronale am folosit limbajul de programare Python şi software-ul TensorFlow [6] şi Keras [7]. Pre-procesarea imaginilor şi generarea bazelor de date modificate, iar apoi generarea imaginilor augmentare am realizat-o folosind software-ul OpenCV [8] pentru Python.

Reţeaua aleasă are următoarea structură: 5 straturi de codificare, 1 strat central şi 5 pentru decodificare. Un strat de codificare are următoarele operaţii:

- convoluţie
- “batch normalization”
- funcţie de activare ReLU
- convoluţie
- “batch normalization”
- activare ReLU
- “max pooling”

Stratul central al reţelei are aceleaşi operaţii ca şi cel de codificare, cu excepţia operaţiei de “max pooling” de la final.

Stratul de decodificare are urmatoarele operaţii:

- deconvoluţie (“up sampling”)
- concatenare (legatura directă cu stratul echivalent din codificare)
- deconvoluţie
- “batch normalization”
- activare ReLU
- deconvoluţie
- “batch normalization”

- activare ReLU
- deconvoluţie
- “batch normalization”
- activare ReLU

Predicţia finală va fi rezultatul aplicării a unei convoluţii cu nucleu 1 x 1 şi a unei activări sigmoide.

Pentru antrenare am folosit o dimensiune a lotului de 16 şi un număr maxim de 50 de epoci. Timpul de antrenare pentru o epocă este de 172 secunde (în medie) pe un sistem desktop echipat cu 2 plăci video Nvidia 1080 Ti. O analiză a antrenării este ilustrată în figura 3, unde se observă minimizarea funcţiei de cost (“loss”) şi convergenţa coeficientului Dice spre valoarea 1 (în metrică IoU utilizată, valoarea 1 reprezintă o suprapunere perfectă a datelor).

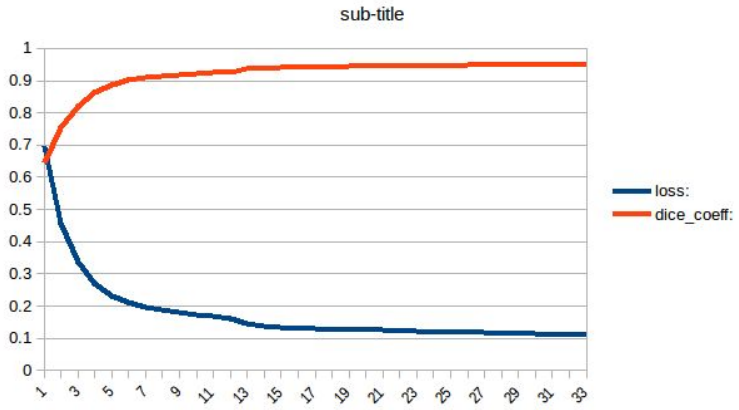

Figura 3. Funcţia de cost (“loss”) scade, iar coeficientul Sorensen-Dice (“dice coeff”) converge spre valoarea 1 după antrenarea timp de 33 epoci.

Un exemplu de rezultat al segmentării folosind reţeaua neuronală este ilustrat în figura 4:

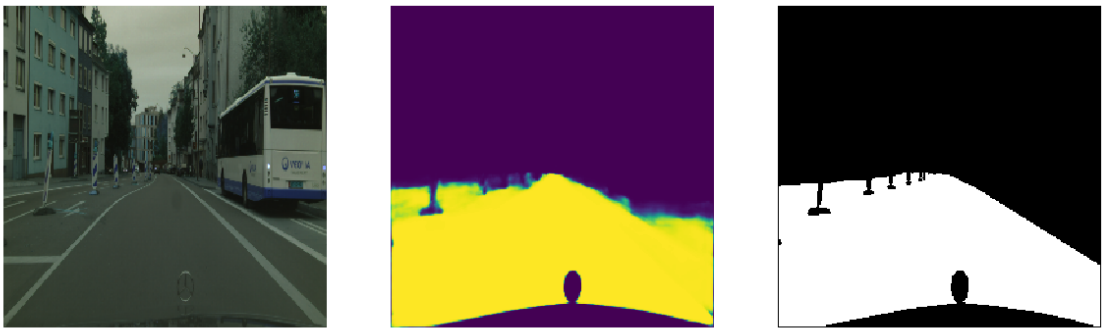

Figura 4. Exemple de predicţie a reţelei. Stânga: imaginea de intrare color, centru: rezultatul segmentării (predicţia), dreapta: imaginea etichetată (“ground truth”).

4. Detecţia şi măsurarea mişcării folosind reţele neuronale convoluţionale

Imaginile segmentate folosind tehnicile descrise în secţiunea anterioară sunt integrate într-un framework probabilistic, bazat pe o hartă dinamică a scenei din jurul autovehiculului.

Din detecţia zonei libere de drum, putem considera că tot ce nu este drum reprezintă un posibil obstacol şi poate fi interpretat şi urmărit în timp. Astfel, această soluţie este utilă pentru a detecta orice tip de obstacol, nefiind limitată doar la vehicule sau pietoni. Pentru implementarea acestei funcţionalităţi am ales folosirea unei hărţi de ocupare bazată pe un filtru de particule. Aceste abordări pot fi folosite pentru estimarea sistemelor neliniare, multi-modale.

Harta de ocupare a scenei bazată pe filtre de particule este o reprezentare dinamică a scenei unde efectul de perspectivă este eliminat. În implementarea din cadrul acestui proiect, obstacolele sunt compuse din particule, fiecare având o poziţie şi un vector de viteză. Harta de ocupare este împărţită în celule de 20 x 20 cm, iar probabilitatea unei celule de a fi ocupată este dată de numărul de particule din acea celulă. Totodată, particulele pot avea un rol dublu: de a forma ipoteze pentru posibile obiecte, sau pot fi considerate elemente de bază a mediului cu proprietatea că particulele se pot deplasa dintr-o celulă în alta (astfel reuşind o reprezentare dinamică a scenei). Partea de predicţie a stării este realizată prin deplasarea particulelor.

Obstacolele din scenă sunt reprezentate de un set de particule, unde fiecare particulă are o poziţie în harta de ocupare, o componentă de viteză dar şi o variabilă pentru “vârsta” particulei din momentul creării ei. Acest parametru de “vârstă” este folosit în procesul de validare şi estimare a vitezei pentru particulă, deoarece sunt luate în considerare numai particulele care “supravieţuiesc” pentru mai multe cadre. Probabilitatea unei celule de a fi ocupată este estimată ca fiind raportul dintre numărul de particule din celula (poziţia lor coincide cu poziţia celulei) şi numărul total de particule permise într-o singură celulă *“Nc”*. Acest parametru este constant şi a fost setat la 200 de particule totale într-o celulă. Alegerea acestei valori reprezintă un compromis între acurateţe şi performanţa sistemului: un număr mare de particule într-o celulă înseamnă că se menţin mai multe ipoteze de viteză simultan, iar sistemul de urmărire va gestiona mai bine obstacolele care deplasează rapid în scenă, dar viteza de execuţie a algoritmului va scădea. Viteza unei celule din harta de ocupare este calculată ca fiind viteza medie a particulelor din acea celulă.

Primul pas al algoritmului de urmărire este cel de predicţie şi este aplicat pe fiecare particulă. Poziţia unei particule este modificată în funcţie de viteza ei proprie şi în funcţie de parametri de mişcare ai ego-vehiculului (viteza şi rata de rotaţie “yaw rate”). Al doilea pas al algoritmului este procesul de actualizare, bazat pe harta binară de ocupare a celulelor creată prin procesarea imaginii segmentate.

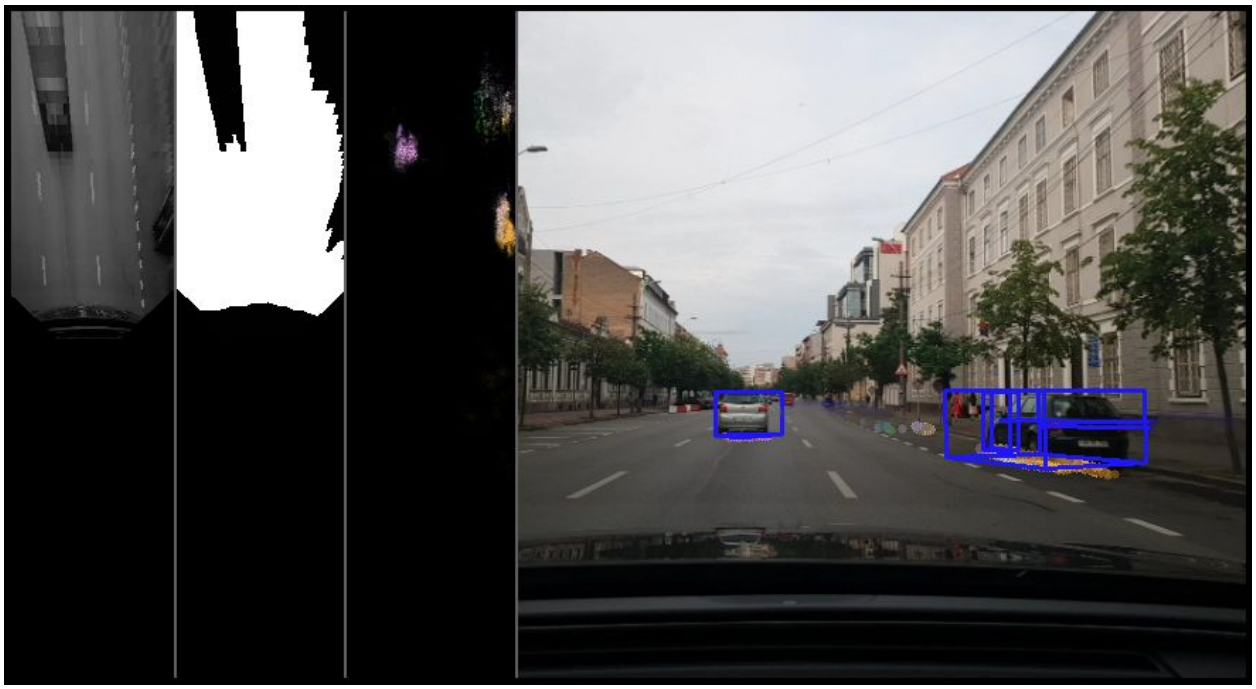

Figura 5. Framework-ul probabilistic de percepţie a scenei.

Figura 5 ilustrează etapele de procesare din cadrul framework-ului probabilistic. În imaginea din stânga se pot observa etapele procesării şi crearea hărţii de ocupare dinamică: prima imagine reprezintă scena cu efectul de perspectivă eliminat (IPM), în a doua este ilustrată imaginea segmentată IPM (obţinută din reţeaua neuronală convoluţională), a 3-a imagine ilustrează particulele şi viteza lor codificată în funcţie de vectorii de viteză (albastru - obiect în mişcare, galben - obiect static). În imaginea din dreapta este prezentat rezultatul procesării şi grupării particulelor şi a celulelor în obstacole.

5. Rezultate suplimentare: Calibrarea automată a parametrilor extrinseci ai camerei

Unul dintre dezideratele acestui proiect este de a oferi un sistem de percepţie a mediului care să poată funcţiona cu resurse senzoriale limitate, inteligenţa artificială suplinind absenţa unor senzori costisitori dar foarte precişi, precum laserscannerele sau radarele. Şoferul va putea monta o cameră sau un dispozitiv mobil în spatele parbrizului, camera fiind orientată spre traficul din exterior, dar pentru a putea utiliza informaţiile de la această cameră trebuie ca trăsăturile din imaginile achiziţionate să poată fi puse în legătură cu trăsăturile 3D ale lumii reale. Acest lucru implică procesul de calibrare, ceea ce un utilizator obişnuit ar prefera să nu fie obligat să facă.

În mod tradiţional, calibrarea se face în medii controlate, în condiţii de laborator, de obicei prin măsurarea manuală a unor obiecte de referinţă. Dacă ne referim la scenariul în care

utilizatorul pur şi simplu montează camera în maşină, dorind apoi să pornească la drum, condiţiile de laborator nu pot fi îndeplinite, şi astfel avem nevoie de calibrare automată.

Pentru calibrarea automată, avem nevoie de repere cu dimensiuni 3D cunoscute, precum lăţimea unei benzi de circulaţie sau dimensiunea unor obstacole cunoscute. Noi am ales varianta a doua, şi ne vom folosi de avantajele reţelelor neuronale pentru a selecta doar obstacolele de tip autovehicul din imagini necalibrate. Soluţia pentru detecţia obstacolelor este bazată pe o CNN pre-antrenată cu exemple formate din perechi imagini-obstacole reprezentate ca un dreptunghi în spaţiul imagine. Am folosit API-ul TensorFlow pentru simplificarea procesului de antrenare, şi am re-antrenat o soluţie existentă de detecţie a obstacolelor din imagini singulare numită MobileNet [10], care este o reţea proiectată în mod explicit pentru a avea un număr redus de parametri de antrenare, şi care necesită putere de calcul redusă pentru predicţie, putând astfel fi utilizată în timp real pe dispozitive mobile. Un exemplu de detecţie automată a acestor obiecte este dat în figura 6.

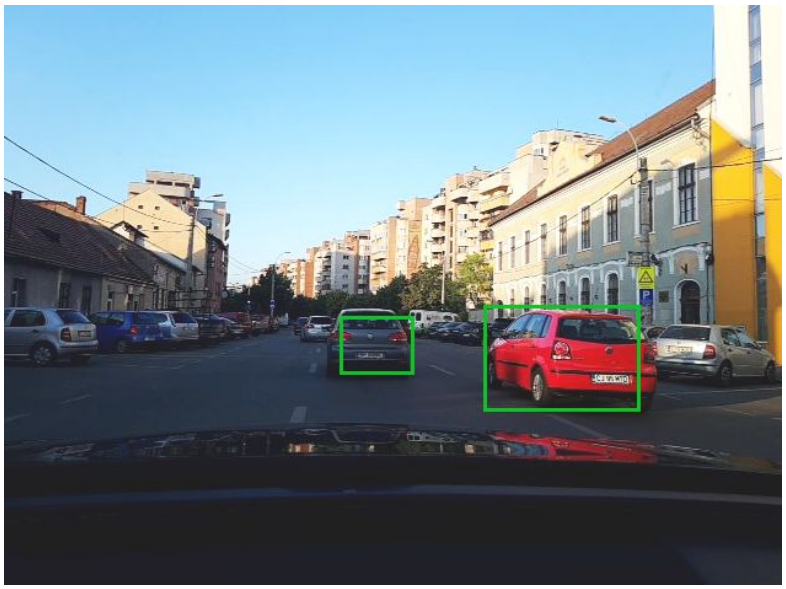

Figura 6. Exemplu de autovehicule detectate automat de MobileNet.

În funcţie de distanţa unui autovehicul faţă de noi, dimensiunea lui aparentă în imagine va fi variabilă. Distanţa în lumea reală va avea o legătură directă cu coordonata rândului din imagine, iar obiectul va avea dimensiunea zero (va dispărea) când coordonata limitei lui inferioare va fi egală cu linia orizontului (linia de fugă, “vanishing line”). De fapt, între coordonata rândului limitei inferioare a obiectului şi dimensiunea lui în imagine va exista o relaţie liniară, parametrii dreptei fiind direct influenţaţi de parametrii extrinseci ai camerei video, unghiul de aplecare (pitch) şi înălţimea camerei faţă de sol.

Datorită faptului că suprafaţa drumului nu e întotdeauna plană, dimensiunea vehiculelor nu este cunoscută cu exactitate, şi detecţia nu este perfectă (vehiculul nu este încadrat perfect), nu vom obţine o dreaptă exactă pentru toate vehiculele din scenă. Colectând toate detecţiile

dintr-o secvenţă de 5 minute, se va obţine un grafic de genul celui prezentat în figura 7. Pentru identificarea dreptei care se potriveşte cel mai bine acestor date, se foloseşte metoda RANSAC.

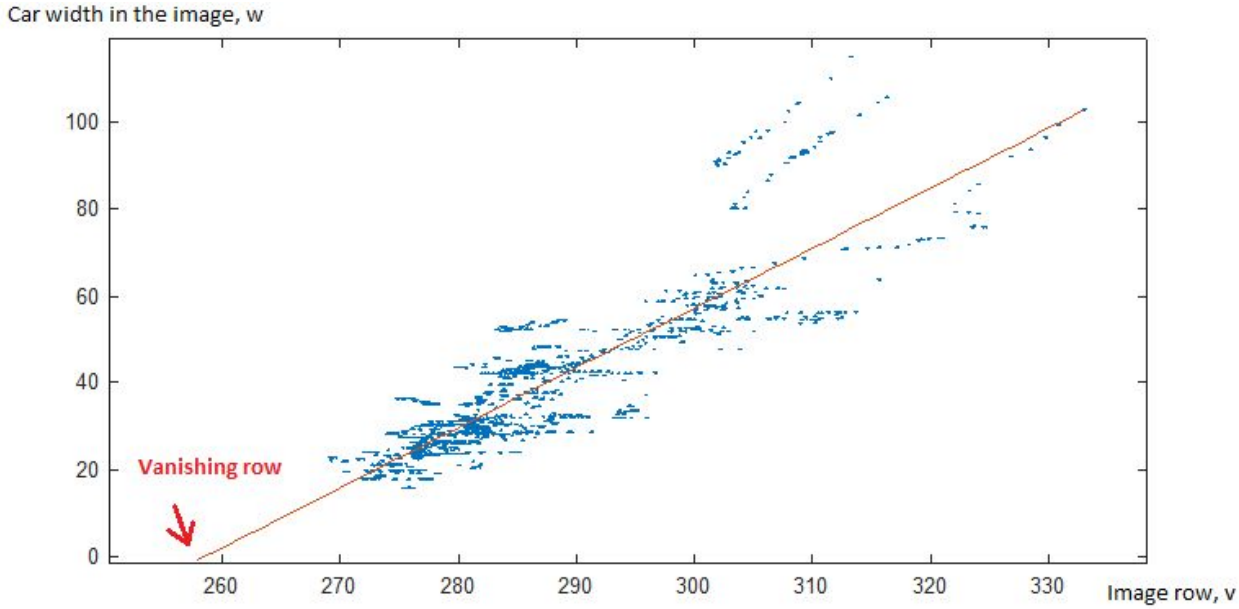

Figura 7. Exemplu de autovehicule detectate automat de MobileNet.

Din parametrii dreptei obţinute vom estima înălţimea şi unghiul de aplecare al camerei folosind o metodă bazată pe filtrul Kalman extins, unde starea estimată este un vector format din înălţimea camerei şi unghiul de pitch, iar măsurătoarea este dată de două puncte de tip (coordonată rând, lăţime) de pe dreapta estimată. Filtrul converge cam dupa 4-5 iteraţii.

Pentru măsurarea unghiului de orientare laterală a camerei faţă de axul longitudinal al maşinii, se folosesc perechi de detecţii obţinute în cadre succesive, presupunând că ele aparţin aceluiaşi obiect. Traiectoria obiectului trebuie să conveargă spre punctul de fugă, care este poziţionat pe linia orizontului, pe care am estimat-o deja. În realitate, fără un algoritm de urmărire complex, nu putem garanta că detecţiile aparţin aceluiaşi obiect, dar vom utiliza din nou o abordare statistică, şi vom selecta mediana ipotezelor acestor puncte de fugă, precum în figura 8.

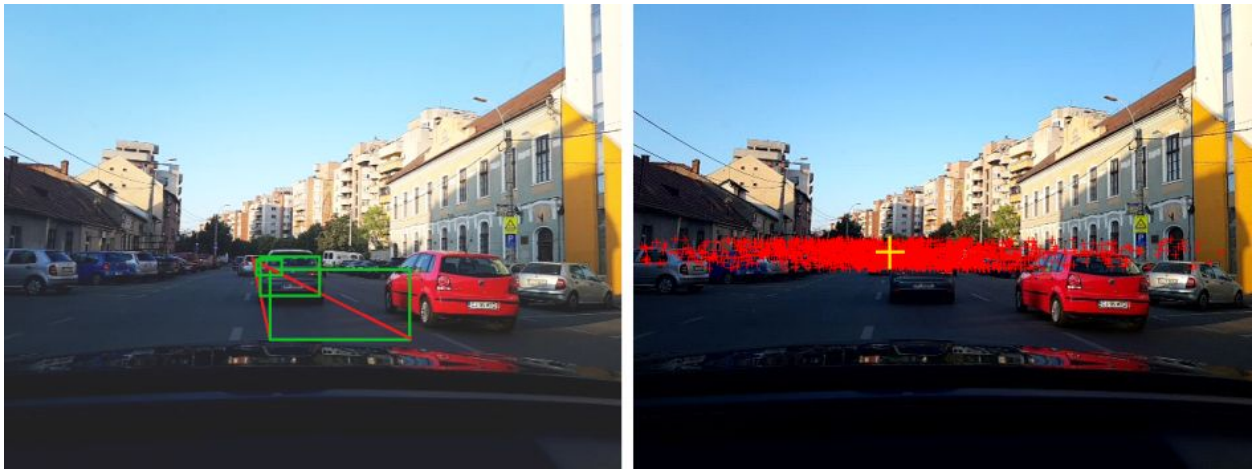

Figura 8. Principiul deteţiei punctului de fugă (stânga), şi selectarea punctului de fugă din multiple ipoteze (dreapta).

Coordonata coloanei punctului de fugă este direct legată de unghiul de orientare laterală a camerei. O pozitie perfect centrală a acestui punct înseamnă o aliniere perfectă a camerei cu axul autovehiculului propriu, iar orice deviaţie de la acest punct este dată de unghiul de orientare ψ, astfel:

$$tan\ \psi = (x0 - w/2)/f\ ,$$

unde x0 este coordonata coloană a punctului de fugă, w este lăţimea imaginii, şi f este distanţa focală a camerei în pixeli.

6. Rezultate suplimentare: Dezvoltarea şi implementarea unui algoritm pentru monitorizarea atenţiei şoferului pe baza clipirilor şi a direcţiei de privire

În cadrul acestei etape am dezvoltat un sistem automat de analiză a mişcărilor oculare care calculează mai multe metrici oculomotoare, cu implicații importante într-o varietate de aplicații: monitorizarea atenției, estimarea nivelului de oboseală, interacțiune om calculator etc. Arhitectura sistemului propus este ilustrată în Figura 9: metoda propusă se bazează pe două module principale: modulul de analiză a clipirilor şi modulul de analiză a direcției de privire.

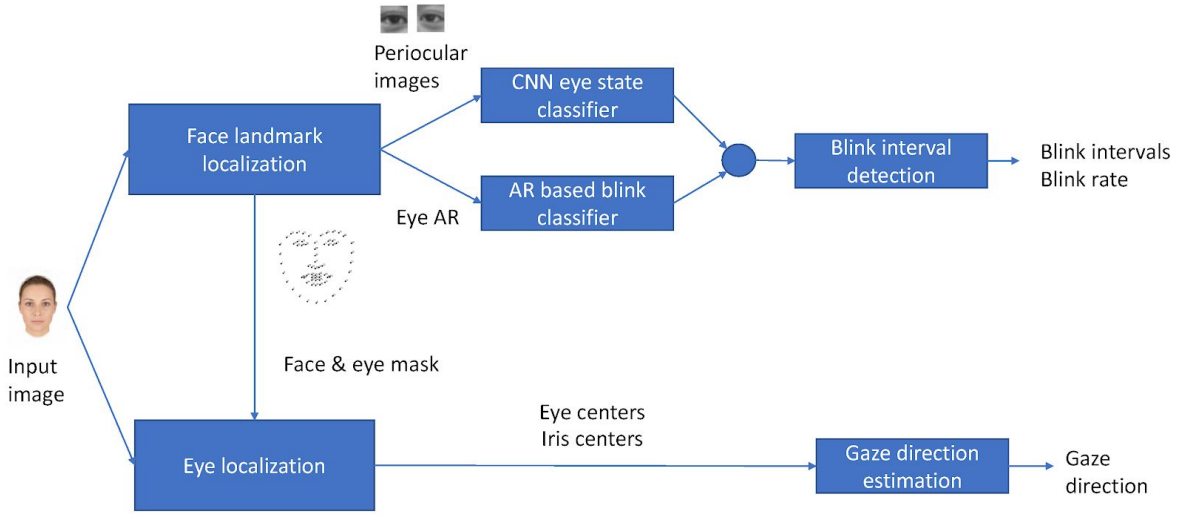

Figura 9. Arhitectura soluției de analiză a clipirilor şi a direcției de privire

La început, se utilizează o bibliotecă software (*dlib* [9]) pentru a detecta fețele umane şi 68 de puncte de interes de pe față, inclusiv colțurile ochilor şi mai multe puncte de pe pleoape. Pe baza acestor puncte se stabilesc regiunile de interes pentru metoda propusă.

Modulul de analiză a clipirilor combină răspunsul a doi clasificatori pentru a determina starea ochilor (închişi sau deschişi) în fiecare cadrul al secvenței video. Primul clasificator se bazează pe punctele de interes detectate de librăria *dlib*, mai precis analizează raportul de aspect (*aspect ratio*) dintre lățimea şi înălțimea conturului ochilor. Cel de-al doilea clasificator este o rețea neuronală convoluțională care analizează regiunea perioculară pentru a determina starea ochiului.

Apoi, utilizând o metodă rapidă de localizare a centrului irisurilor, modulul de detecție a direcției de privire corelează poziția irisurilor cu centrul şi colțurile ochilor pentru a determina punctul în care priveşte subiectul.

Pe baza acestor trăsături extrase automat, se calculează mai multe metrici care pot fi utilizate pentru a monitoriza nivelul de atenție al şoferului: rata clipirilor, direcția de privire, recunoaşterea mişcărilor oculare.

7. Diseminarea rezultatelor parțiale

Rezultatele obținute în această etapă au fost publicate în următoarele jurnale/conferințe internaționale:

a. Diana Borza, Razvan Itu, Radu Danescu, "In the Eye of the Deceiver: Analyzing Eye Movements as a Cue to Deception", Journal of Imaging, Vol. 4, No. 10, 2018, Art. No. 120. [**Indexat ISI**]

b. Razvan Itu, Radu Danescu, "Machine Learning Based Automatic Extrinsic Calibration of an Onboard Monocular Camera for Driving Assistance Applications on Smart Mobile Devices", 2018 International Forum on Advanced Microsystems for Automotive Applications, pp. 16-28.[**Capitol carte Springer**]

8. Concluzii

În cadrul acestei etape am reuşit aducerea la un format comun a bazelor de date pentru detecţia de obstacole şi a bazelor de date pentru segmentarea semantică a scenelor din trafic. Am reuşit, de asemenea şi implementarea şi antrenarea unei reţele neuronale convolutionale pentru a segmenta semantic zona de drum din imaginile din traficul rutier. Acesta abordare a fost implementat utilizând biblioteci software precum TensorFlow şi Keras şi folosind limbajul de programare Python. Rezultatele preliminare obţinute prin analiza imaginilor folosind reţelele neuronale au fost integrate într-un framework probabilistic pentru urmărirea componentelor statice şi dinamice ale scenei, obţinând o estimare preliminară a poziţiei şi mişcării obiectelor din scenă. În mod suplimentar, am realizat, folosind rezultatele CNN, un sistem de calibrare a parametrilor extinseci ai camerei, element valoros pentru realizarea unui sistem de percepţie ce necesită o cât mai mică intervenţie din partea utilizatorului. Totodată, în această etapă am obţinut dezvoltat un framework generic pentru analiza miscărilor oculare în vederea monitorizării atenţiei şoferului. Sistemul propus determină momentele în care subiectul a clipit, precum şi rata de clipire. De asemenea, sistemul include o metodă simplă de estimare a direcţiei de privire pe baza deplasamentului relativ dintre centrele ochilor şi centrele irişilor. Pe baza acestor metrici se poate propune un sistem care atentionează şoferul atunci când ele indică o creştere a oboselii.

Rezultatele preliminare au fost publicate în jurnale şi conferinţe internaţionale.

9. Bibliografie

[1] - Geiger, A, Lenz, P, Urtasun, R, "Are we ready for autonomous driving?", Computer Vision and Pattern Recognition Conference (CVPR), pp. 3354–3361, 2012.

[2] - Udacity Vehicle Dataset, online:
https://github.com/udacity/self-driving-car/tree/master/annotations

[3] - M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, "The Cityscapes Dataset for Semantic Urban Scene Understanding," in Proc. of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[4] - Yu F., et. al, “BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling“, preprint arxiv:1805.04687, 2018, online: https://arxiv.org/abs/1805.04687.

[5] - O. Ronneberger, P. Fischer, T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation", Medical Image Computing and Computer-Assisted Intervention (MICCAI), Springer, LNCS, Vol. 9351: 234-241, 2015, available at arXiv:1505.04597.

[6] - TensorFlow, online: https://www.tensorflow.org/

[7] - Keras, online: https://keras.io/

[8] - OpenCV, online: https://www.opencv.org/

[9] - King, D.E., “Dlib-ml: A machine learning toolkit”, J. Mach. Learn. Res. 2009, 10, 1755 - 1758.

[10] - Howard, A, et al, “Mobilenets: Efficient convolutional neural networks for mobile vision applications”, 2017, arXiv:1704.04861(preprint).