UNIVERSITATEA TEHNICĂ
DIN CLUJ-NAPOCA

Tehnici avansate de percepție a mediului folosind Deep Learning și estimatori probabilistici (DEEPSENSE)

Echipa de cercetare: Radu Dănescu, Răzvan Itu, Diana Borza, Andrei Vadan

ETAPA 2 - Proiectarea și implementarea algoritmilor

Cuprins

1. Introducere - rezumatul etapei

Obiectivul principal pentru etapa a doua a fost integrarea modulului de învățare profundă cu modulul de urmărire probabilistică. Alte obiective identificate în cadrul acestei etape sunt: finalizarea proiectării, antrenării, evaluării rețelelor neuronale, dar și detectarea și măsurarea mișcării folosind rețele neuronale.

Accentul a fost pus în principal pe procesul de utilizare a puterii de clasificare și de segmentare a rețelelor neuronale în mecanismul probabilistic de percepție a obstacolelor statice și dinamice din mediul auto. Acest proces a implicat utilizarea rezultatelor rețelelor neuronale pentru calibrarea automată a camerei și pentru generarea modelului de măsură pentru estimatorul probabilistic, dar și experimentele privind înlocuirea algoritmilor componenți ai procesului de estimare cu rezultate ale rețelelor neuronale.

Dezvoltarea algoritmilor, proiectarea și antrenarea rețelelor au mers în tandem cu dezvoltarea aplicației demonstrator, care a integrat toate componentele de calibrare, segmentare și estimare într-un sistem care funcționează aproape în timp real, și care a permis depanarea și evaluarea la fiecare pas.

Rezultatele obținute în cadrul acestei etape au fost publicate în trei lucrări la conferințe internaționale, și au fost folosite și pentru elaborarea unei teze de doctorat.

2. Proiectarea, antrenarea și evaluarea rețelelor neuronale, și reglarea fină a hiper- parametrilor

Rețeaua neuronală convoluțională folosită pentru extragerea zonei de drum este bazată pe U-Net [1] și a fost descrisă în raportul primei etape. Structura rețelei a rămas la fel, având 5 niveluri de codificare, un nivel central și tot 5 niveluri de decodificare. Am testat rețeaua și cu dimensiunea imaginii de intrare de 512 x 512 pixeli, având rezultate mai bune pe setul de validare, dar timpul de procesare a crescut de cel puțin 2 ori. În varianta de 256 x 256 pixeli predicția se face în 8-10ms, în timp ce imaginea de 512 x 512 pixeli necesită ~20ms. Structura rețelei convoluționale neuronale este ilustrată în figura 1.

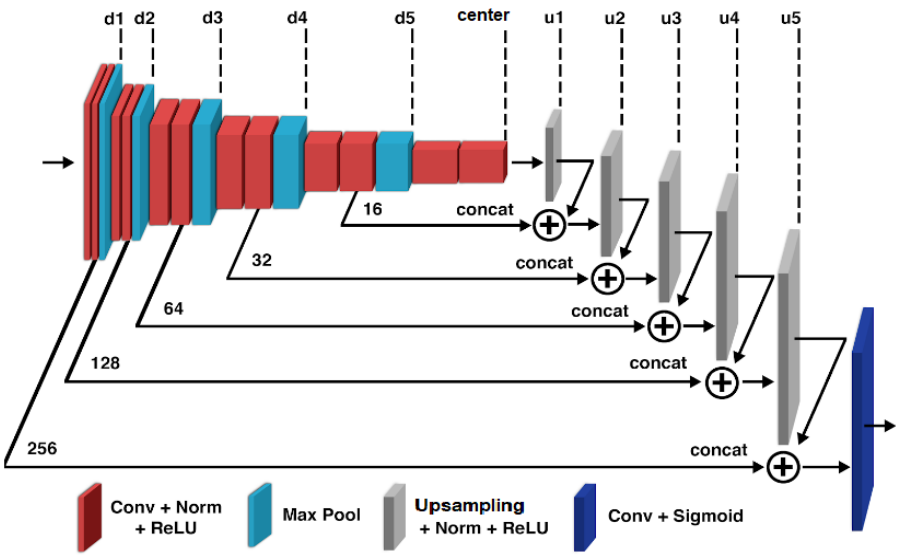

Figura 1. Rețeaua neuronală convoluțională pentru segmentare.

Am testat de asemenea și arhitectura existentă ERFNet [2], folosind software-ul PyTorch. Aceast model de rețea artificială folosește niveluri care au mai puțini parametri, de aceea necesită resurse hardware mai puține. Obiectivul acestui proiect este de a implementa un sistem de percepție direct pe o platformă mobilă, cu resurse hardware limitate, de aceea o abordare folosind ERFNet ar fi justificată. Intrarea și ieșirea rețelei a fost modificată pentru a folosi aceleași baze de date de antrenare (descrise pe larg în raportul etapei 1). Rețeaua necesită un timp de calcul al predicției mai redus decât prima soluție bazată pe U-Net, în detrimentul acurateții. Am antrenat rețeaua folosind aceleași funcții de cost ("binary cross entropy"), dar am monitorizat și coeficientul Dice, iar rezultatul este ilustrat în figura 2.

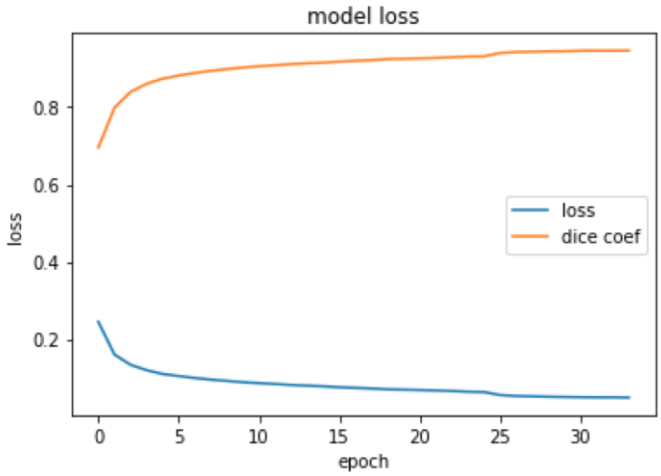

Figura 2. Antrenarea rețelei pentru segmentare folosind funcția de cost "binary cross entropy". Este ilustrat și coeficientului Dice.

În figura 2 se poate observa că antrenarea este oprită automat dacă funcția de cost nu este îmbunătățită după cel puțin 5 epoci consecutive (în acesta caz antrenarea este oprită după 34 epoci). Antrenarea a folosit următoarele baze de date cu imagini din trafic și adnotări: CityScapes [3], Berkeley DeepDrive [4], Mapillary Vistas [5] și KITTI [6].

Segmentarea bazată pe rețele neuronale convoluționale a fost evaluata folosind un set de validare din baza de date CityScapes. Scorul obținut de noi folosind metrica "Intersect over Union" (IoU) este 0.91, iar metoda cea mai performantă la momentul actual [7], de la Google obține un scor de 0.98. Rezultatele comparative sunt ilustrate în tabelul 1.

Model rețea artificială	**Scor IoU (doar clasa de drum)**
DeepLab [7]	0.986
E-Net [8]	0.974
CNN propriu antrenat pe mai multe clase	0.922
CNN propriu antrenat pe o singură clasă (drum)	0.911

Tabel 1. Evaluarea segmentării prin comparație cu alte abordări.

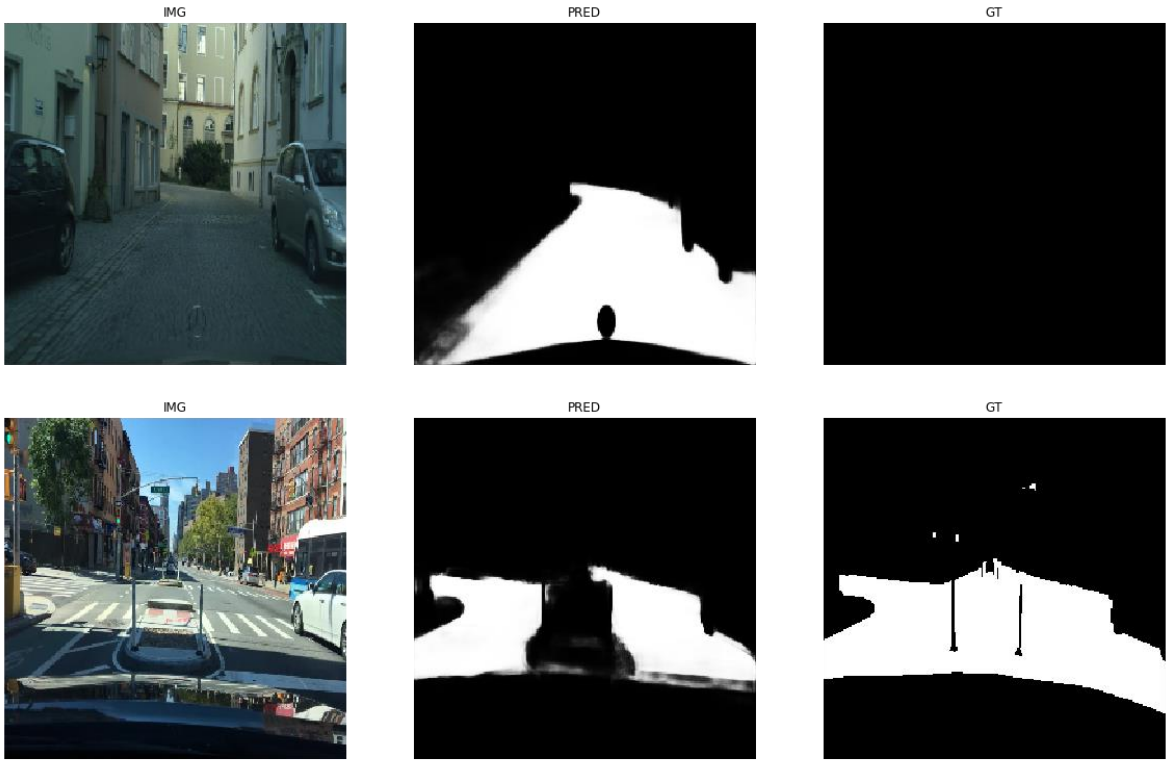

Figura 3. Exemple de adnotări eronate în baza de date CityScapes [3] și Berkeley Deep Drive [4]. Coloana din stânga reprezintă imaginea scenei de drum, în centru sunt predicțiile rețelei, iar în dreapta sunt adnotările greșite.

Evaluarea este realizată folosind o rețea antrenată pentru a oferi predicții asupra unei singure clase (cea de drum). Am observat faptul că la o antrenare pe mai multe clase, scorul obținut crește (0.92 vs. 0.91). Un alt aspect este faptul că metodele de top publicate deja sunt antrenate folosind imagini de dimensiuni mari, iar partea de predicție este post-procesată și rafinată. În cazul nostru, am ales o varianta care să favorizeze o viteză de predicție mare, în timp ce nu efectuam nicio post-procesare a hărții de predicție (am aplicat doar o binarizare cu prag fix).

Totodată, în timp ce am investigat rezultatele slabe ale evaluării, am observat prezența unor imagini adnotate greșit în setul de validare din bazele de date folosite.

Rețeaua antrenată de noi a folosit mai multe baze de date, iar numărul de adnotări greșite este redus proporțional, iar din figura 3 se poate observa faptul că predicția este corectă, deși adnotarea este eronată. Aceste imagini s-ar putea să afecteze unele metrici de evaluare și putem trage concluzia că sistemul propus de noi este suficient de robust pentru a putea fi folosit pentru a extrage zonele de drum într-un sistem de percepție bazat pe o singură cameră.

3. Detectarea si masurarea mișcării folosind rețele neuronale convolutionale

Viteza în spațiul 3D poate fi codificată în imagine sub forma componentelor ei pe axa X și axa Y, iar combinate cu componenta de ocupație (prezența sau absența obstacolelor) se poate genera o imagine pe trei canale (care pot fi scalate în intervalul 0...255, sau 0...1), care poate fi văzută ca o imagine pseudo-color (fig. 4). În concluzie, se poate utiliza o rețea capabilă

de generare de imagini color pentru a genera imaginea de viteze în mod direct, pe baza unui flux de imagini de intrare. Soluția care ar funcționa cel mai bine este cea a utilizării unor rețele neuronale convoluționale recurente, care mențin o stare ascunsă ce va fi folosită în procesul de clasificare, dar din păcate aceste rețele necesită o cantitate foarte mare de date de antrenare, și putere enormă de calcul, astfel că s-a renunțat la această variantă. O altă abordare a fost să luăm perechi de imagini la timpi consecutivi, care să compună o imagine pe mai multe canale (două imagini color produc o imagine pe șase canale, două imagini grayscale o imagine pe două canale), și să folosim o arhitectură de tip encoder-decoder, similară cu cea folosită pentru segmentare, pentru a genera imaginea de viteze. Experimentele pentru această abordare nu au produs încă rezultatele dorite de noi, astfel că s-a decis, deocamdată, folosirea mecanismului probabilistic bazat pe particule pentru extragerea vitezelor, iar utilizarea rețelelor a fost limitată, deocamdată, doar la producerea de informații statice care vor sta la baza modelului de măsură.

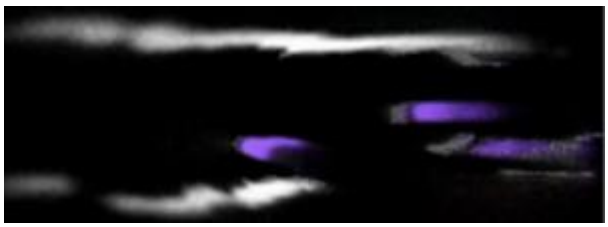

Figura 4. O imagine pseudo-color care codifică viteza sub forma canalelor de culoare.

4. Integrarea modulului de învățare profundă cu modulul de urmarire probabilistica

Rezultatele date de rețelele de tip CNN folosite sunt folosite ca date primare în procesul de calibrare automată a camerei, și în algoritmul de detecție și urmărire a obstacolelor.

4.1. Calibrarea automată a camerei

Calibrarea camerei este necesară pentru a stabili relația dintre spațiul imaginii, unde ne putem baza pe rezultatele date de rețelele neuronale convoluționale, și spațiul lumii 3D, unde vom realiza urmărirea probabilistică a obiectelor reale din traficul auto. Calibrarea poate fi realizată în mod manual sau supervizat, utilizând unelte de calibrare cunoscute, sau poate fi realizată automat, pentru a permite o punere în funcțiune a sistemului cât mai rapidă și care să solicite cât mai puțin utilizatorul.

Pentru calibrare este necesar să punem în corespondență dimensiuni cunoscute din lumea reală cu proiecția lor în spațiul imagine, detectată în mod independent de parametrii de calibrare.

O primă abordare pentru calibrare este de a folosi lățimea medie a unui vehicul, iar modul de calcul a fost prezentat în raportul etapei 1. Principiul folosit este următorul: vehiculele apropiate vor avea o lățime mai mare (în pixeli) în spațiul imaginii decât cele situate la o distanță mai mare. Vehiculele apropiate de linia de orizont vor avea lățimea minimă (ele devin puncte). Totodată aici este și punctul de fugă din imaginile de trafic. Astfel, am creat o hartă de voturi din lățimea unui vehicul și coordonată sa din imagine pe axa rândului. Harta de voturi este folosită pentru a determina relația liniară care există între lățimea unui vehicul și coordonata sa de-a lungul axei orizontale. Am folosit metoda RANSAC [9] pentru a extrage

această dependență. Linia orizontului din imagine (coordonată pe axa orizontală a punctului de fugă) va fi la coordonată în care lățimea unei mașini este 0 pe linia extrasă din RANSAC.

Extragerea dependenței liniare se poate face și prin modificarea transformatei Hough. Acumulatorul Hough va fi de fapt o hartă de voturi similară cu prima abordare: se incrementează valoarea în fiecare punct de coordonate (“lățime vehicul”, “index rând”). Astfel, prin folosirea transformatei Hough este necesară apoi determinarea parametrilor rho și theta ale liniei cu cele mai multe voturi din acumulator. Rezultatul acestei abordări este ilustrat în figura 5.

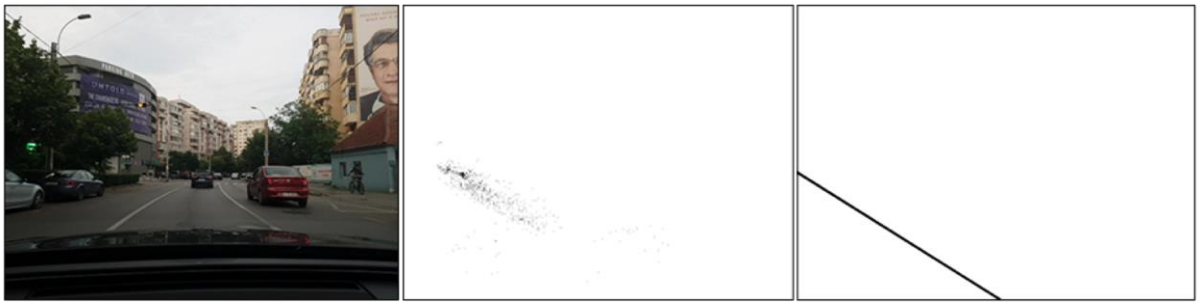

Figura 5. Stânga: imaginea de intrare, centru: acumulatorul Hough, dreapta: linia extrasă din acumulator.

Un alt reper de dimensiune aproximativ cunoscută este lățimea benzilor de circulație, care de asemenea devine mai mică în spațiul imagine pe măsură ce se apropie de punctul de fugă.

Calibrarea înălțimii și a unghiului de înclinare se poate face folosind relația dintre lățimea structurii 3D din lume (lățimea vehiculului sau a benzii) și lățimea proiectată în planul imaginii. În figura 6, relația dintre lățimea benzii de circulație în spațiul 3D al lumii și în spațiul 2D al imaginii poate fi definită folosind ecuația 1.

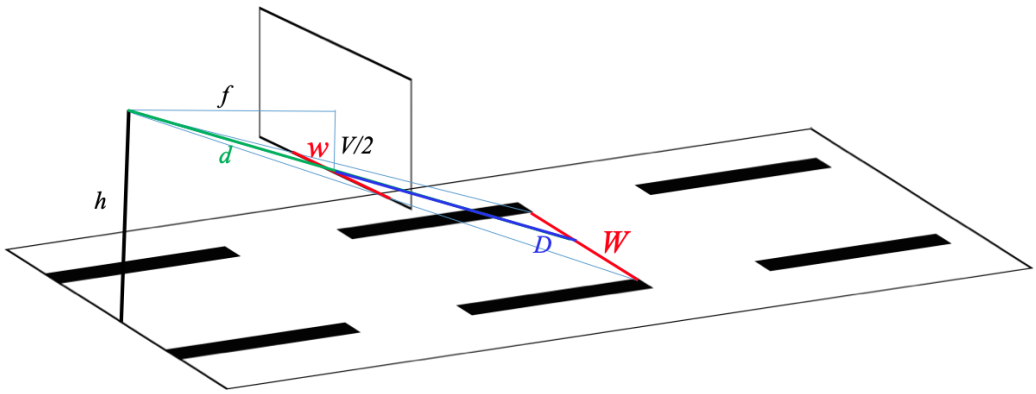

Figura 6. Proiecția în spațiul imaginii a lățimii unei structuri 3D cunoscute, situată în partea de jos a imaginii.

$$w/W = d/D \quad (1)$$

Distanța “d” din ecuația 1 poate fi calculată folosind distanța focală (exprimată în pixeli) și din înălțimea imaginii, presupunând că poziția este în partea de jos a imaginii.

$$d=\mathrm{sqrt}(f^2 + (\text{img height}/2\ ^2) \quad (2)$$

Putem folosi aceste ecuații pentru a determina distanța D dintre structura 3D (lățimea benzii sau a unui vehicul) și centrul optic al camerei. Dacă structura este o bandă de circulație situată pe planul drumului, atunci putem determina și înălțimea camerei față de sol folosind ecuația 2, și după cum este ilustrat în figura 7.

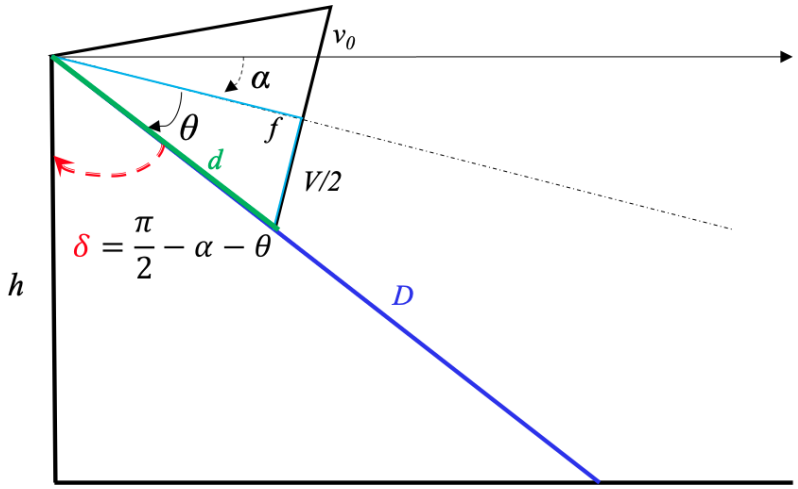

Figura 7. Relația dintre parametri camerei și distanța D.

h= D * cos(delta) (3)

Folosind dependența liniară calculată anterior, avem deja coordonata rând a punctului de fugă “v0” și putem determina unghiul de înclinare al camerei folosind ecuația 4.

tan (pitch) = (img height/2 - v0) / focal (4)
delta = pi/2 - pitch - theta (5)

În ecuațiile 4 și 5, θ reprezintă jumătate din câmpul vizual vertical al camerei. Am publicat rezultatele calibrării folosind lățimea benzilor de circulație în [10], iar prin folosirea lățimii unui vehicul în [11].

Punctul de fugă este punctul din imagine unde se intersectează elementele care în spațiul 3D sunt paralele, precum marcajele rutiere. Acest punct se poate folosi pentru a calcula unghiul de înclinare (“pitch”) și de girație (“yaw”).

Linia pe care se găsește punctul de fugă poate fi determinată prin statistica descrisă anterior, acumulând lățimi de bandă sau de obstacol și poziția lor în imagine. Punctul de fugă poate fi, de asemenea, determinat în mod direct din fiecare imagine în parte, iar combinația celor două metode ne va ajuta să avem o calibrare continuă, care să compenseze mișcările de oscilație ale vehiculului propriu. În lucrarea [10] am propus o metodă pentru a calcula punctul de fugă folosind gradientul și magnitudinea punctelor din imagine. Metodele existente sunt bazate pe extragerea punctelor de trăsături relevante (de exemplu muchii), în timp ce abordarea propusă de noi folosește orice punct din imagine. Totuși, pentru a reduce complexitatea computațională, am definit o zonă de interes de unde punctele din imagine pot vota în funcție de orientarea gradientului și magnitudine. Harta de voturi rezultată este filtrată și maximul local va fi ales ca fiind punctul de fugă. Figura 8 reprezintă harta de voturi și punctul de fugă calculat.

Figura 8. Harta de voturi (stânga) și punctul de fugă rezultat (dreapta).

O altă abordare de a detecta punctul de fugă în imagini este de a folosi o rețea neuronală convoluțională care oferă predicții asupra coordonatelor din imagine ale acestui punct. Am publicat această abordare în [12], dar o îmbunătățire obținută recent este de a folosi două rețele antrenate separat: una pentru predicția asupra coordonatei orizontale (“X”) și alta pentru predicția coordonatei verticale (“Y”). Cele două rețele au aceeași structură cu cea publicată deja: 5 niveluri convoluționale cu număr variabil de filtre, urmate de 3 niveluri complet conectate, ultimul fiind compus din 1 neuron care va reprezenta coordonata punctului de fugă din imagine. Intrarea rețelei este o imagine de dimensiune 200 x 70 pixeli și rețeaua a fost antrenată pentru un total de 200 epoci folosind dimensiunea lotului de 256 și funcția de cost RMSE. Soluția a fost implementată și pe dispozitive mobile Android, iar un exemplu de predicție este ilustrat în figura 9, unde cercul alb reprezintă predicția.

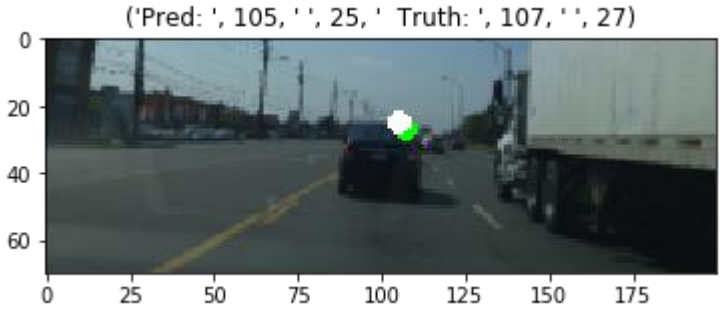

Figura 9. Exemplu de predicție (punctul alb) și valoarea reală (punctul verde).

În sistemul propriu de percepție am folosit și testat ambele implementări, dar am ales pentru integrare varianta algoritmică, bazată pe orientarea și magnitudinea punctelor.

Calibrarea înălțimii camerei față de sol și a unghiului de înclinare a obținut rezultate bune. Am efectuat 3 teste folosind vehicule diferite și camere diferite, dar am comparat și pe baza de date KITTI [6], în care avem informații despre înălțimea camerei față de sol. Rezultatele sunt prezentate în tabelul 2.

Scenariu	Înălțimea camerei față de sol (calibrată)	Înălțimea camerei față de sol (valoarea reala)	Unghiul de înclinare (calibrat)
Test 1	1268	1250	-1.68°
Test 2	1234	1200	-3.8°
Test 3	1468	1480	2.97°
KITTI	1649	1650	-0.85°

Tabel 2. Rezultatele auto-calibrării înălțimii camerei față de sol și calculul unghiului de înclinare (“pitch”).

În testele efectuate de noi, dar și în baza de date KITTI, nu avem informații despre unghiul de înclinare. Astfel, singurul mod de a valida corectitudinea acestui unghi este de a construi matricea de proiecție și de a crea o imagine cu efectul de perspectivă eliminat. Analiza acestei imagini este folosită pentru validare. O imagine în care liniile benzilor de circulație sunt dispuse paralel va însemna că avem un unghi de înclinare corect (figura 10).

Figura 10. Generarea imaginii “bird’s eye view” folosind parametrii calculați din calibrare.

Punctul de fugă calculat cu 2 rețele artificiale este îmbunătățit față de metoda deja publicată (care folosește o singură rețea neuronală). Rezultatele comparative sunt ilustrate în tabelul 3, unde am testat folosind baza de date proprie care este împărțită în două: imagini din scenarii de oraș și de pe autostrada.

Metrică eroare	Oraș (V1)	Oraș (V2)	Autostradă (V1)	Autostradă (V2)
NormDist	0.05072	0.29725	0.02612	0.012905

Tabel 3. Comparație între rețeaua artificială inițială (V1) și metoda îmbunătățită folosind 2 rețele artificiale (V2).

În tabelul 3, metrica de eroare pentru comparație este “NormDist” care este cea folosită și în lucrările deja publicate. Reprezintă de fapt eroarea RMSE dintre punctul de fugă calculat și cel real, iar rezultatul este împărțit la diagonala imaginii de intrare (astfel fiind o metrică invariantă la dimensiunea imaginii de intrare).

4.2. Modelul probabilistic al lumii

Scena de drum este modelată ca o hartă de ocupare, având 120 x 500 celule, în care fiecare celulă va reprezenta o zonă de 20 x 20 cm, ceea ce înseamnă că spațiul complet de procesare este o zonă de 24 x 100 m. Camera este orientată înainte și este poziționată în centrul hărții de ocupare, la coordonată de rând 250 și coordonată de coloană 60. Harta de ocupare bidimensională reprezintă reproiectarea spațiului 3D al lumii (al scenei observate) într-o vedere de sus a scenei (numită de obicei “bird’s eye view”). Spațiul vizibil este doar jumătate din harta de ocupare, iar cealaltă jumătate va fi actualizată de pasul de predicție folosind celulele vizibile din scenă. Prin folosirea acestei abordări, sistemul poate fi ușor extins prin adăugarea mai multor camere și senzori, de exemplu se poate monta o cameră orientată în spate care va actualiza măsurătorile pentru jumătatea inferioară a hărții de ocupare a scenei.

Harta de ocupare va reprezenta obiecte individuale, fiecare compus din particule dinamice având o viteză și o poziție. Migrarea particulelor dintr-o celulă în alta are la bază viteza proprie a particulei, dar și viteza de deplasare a vehiculului și rata de rotație. Algoritmul pentru predicția și actualizarea particulelor este descris pe larg în lucrarea [13] și [14]. Fiecare celulă din hartă va avea o probabilitate de a fi ocupată, determinată din numărul de particule din celulă raportat la numărul maxim de particule dintr-o celulă (Nc = 100, parametru configurabil în funcție de hardware-ul folosit).

4.3. Utilizarea rezultatelor CNN în urmărirea probabilistică

Urmărirea scenei de obiecte 3D se realizează prin estimarea continuă a probabilității de ocupare a celulelor din harta de ocupare, precum și a distribuției de probabilitate a vitezelor celulelor. Pentru realizarea acestui deziderat se va utiliza rezultatul segmentării semantice pe baza CNN pentru a genera modelul probabilistic de măsură. Rezultatul segmentării se proiectează folosind parametrii calibrați ai camerei în vederea de tip “bird eye”, corespunzătoare scenei observate de sus, fiecare pixel corespunzând unei celule din harta de ocupare. Deoarece proiecția este validă doar pentru suprafața drumului, punctele de interes sunt de fapt doar punctele de contact ale obstacolelor cu suprafața drumului, sau mai bine spus punctele de graniță dintre zonele segmentate de CNN ca obstacole și zonele segmentate ca drum.

În lucrările publicate deja am prezentat diferite metode de extragere a punctelor de contact ale obiectelor cu drumul. O abordare este de a proiecta raze cu originea în punctul focal al camerei spre obiecte, de-a lungul cărora se vor căuta tranziții de intensitate. Această abordare a fost publicată în [15] și are la bază idei și principii din lucrarea [16]. În [14] am prezentat o extensie a algoritmului de scanare folosind raze care va calculat 3 distanțe de-a lungul fiecărei raze pentru a îmbunătăți robustețea și precizia. Totuși, majoritatea obstacolelor și în special vehiculele nu vor atinge suprafața de drum complet. Mașinile ating drumul doar cu rotile, iar acest aspect este amplificat în imaginile cu efectul de perspectivă eliminat. Astfel, este nevoie să umplem zonele goale dintre roți pentru a genera o detecție robustă a distanțelor

până la obstacole. Pentru aceasta, am implementat un algoritm de umplere a spațiilor convexe, iar întreg procesul este descris pe larg în lucrarea [17].

Principiul din toate abordările prezentate este de a folosi camera similar cu un scanner bazat pe laser, prin analiza razelor proiectate în imaginile de tip “bird’s eye view”. Aceste imagini procesate sunt folosite pentru a crea o hartă de măsurare pentru modelarea probabilității unei celule de a fi ocupată sau nu. Acest model trebuie să ia în considerare erorile care apar în timpul procesului de măsurare, de aceea am identificat următoarele limitări: erorile de-a lungul razelor longitudinale și limitările determinate de procesul de observație. Pe baza celor două tipuri de erori se va genera, pentru fiecare rază de observație, o probabilitate ideală pentru obstacol de a exista (până la punctul de contact probabilitatea de a exista este aproape de 0%, la punctul de contact aproape 100%, iar după punctul de contact avem zona neobservabilă, unde probabilitatea este de 50%). Probabilitatea ideală va fi supusă unui proces de convoluție cu un nucleu Gaussian unidimensional, cu deviația standard dependentă de modelul de eroare la determinarea distanței din maparea imaginii perspectivă în planul suprafeței drumului.

Aceste erori sunt modelate de ecuația 1, unde h reprezintă înălțimea camerei față de sol, z este distanța până la obiect și σ□ modelează erorile mici (deviațiile) în unghiul de înclinare al camerei, cauzat de suprafețe de drum accidentate sau inegale. Această valoare am setat-o la 0.1 grade, în timp ce valoarea pentru alte surse de erori care nu se pot modela (σ0) este setată la 0.1 metri.

$$\sigma z = h\,(1 + (z/h)^2)\,\sigma\square + \sigma 0 \qquad (6)$$

Procesul de generare a modelului de măsură este redat în figura 11. Mai multe detalii pot fi găsite în [17].

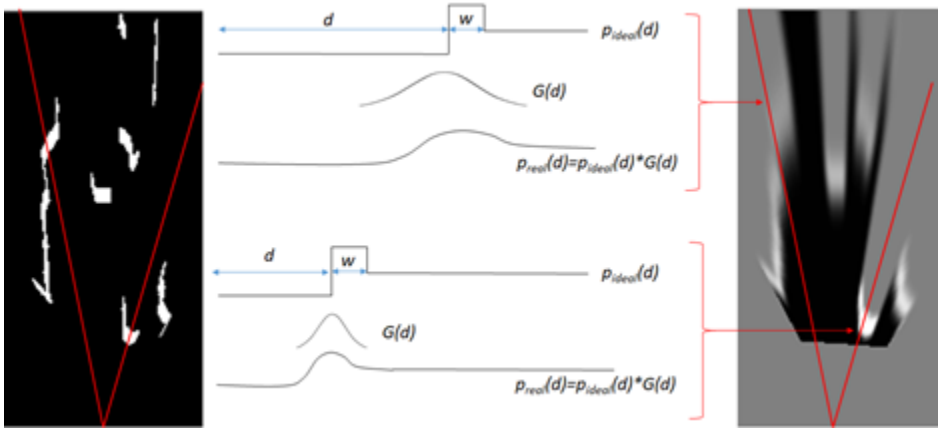

Figura 11. Generarea modelului de măsură pe baza rezultatelor CNN.

După actualizarea modelului lumii pe baza modelului de mișcare a particulelor și a modelului de măsură obținut pe baza segmentării CNN, celulele ocupate din hartă se vor grupa pe baza similarității, extrăgându-se obiectele individuale din scenă.Gruparea celulelor este realizată în funcție de proximitatea celulor, dacă vectorii lor de viteză sunt asemănători (diferența dintre orientarea lor este mai mică de 30°) și dacă celulele au o probabilitate de a fi ocupate de minim 0.5. Presupunând că o celulă conține un singur obstacol, viteza unei celule este calculată ca media vitezei particulelor din celulă. Din celulele grupate am creat cuboide,

iar în final avem o listă de obiecte, fiecare având o poziție, orientare, viteză, lungime, lățime și o înălțime standard de 1.5 metri.

Întreg procesul este descris în figura 12.

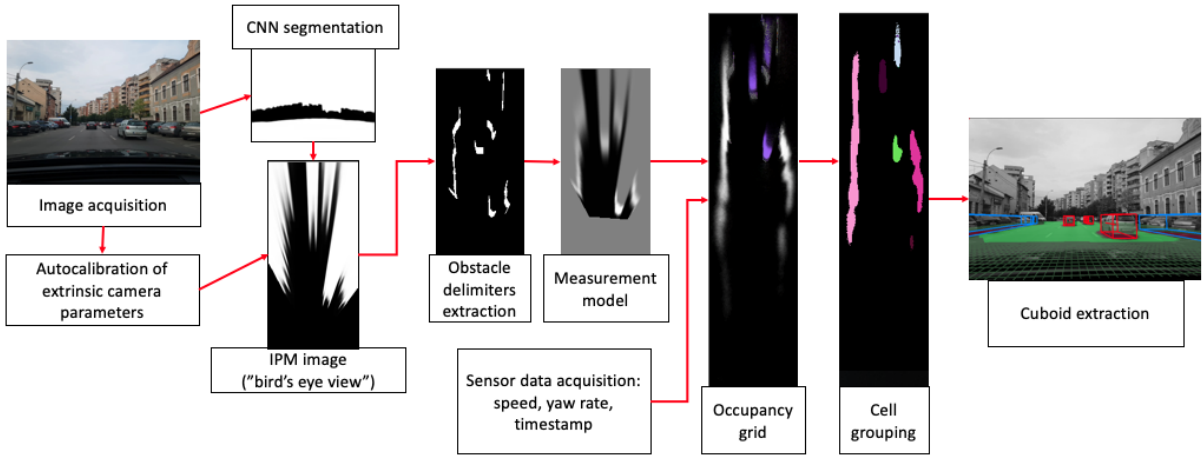

Figura 12. Fluxul de procesare al sistemului monocular de percepție a traficului rutier.

4.4. Evaluarea sistemului de detecție

Pentru evaluare am folosit și baza de date pentru evaluarea algoritmilor de urmărire KITTI [6]. Rezultatele pe secvențele numerotate cu 0005, 0010, 0011 și 0018 sunt ilustrate în tabelul 4. Am calculat eroarea Mean Absolute Error (MSE) dintre locația 3D a obiectului detectat de noi și locația reală din baza de date KITTI. Evaluarea este efectuată doar pentru poziția de pe axa Z, care reprezintă distanța până la obiectul detectat și este cea mai relevantă pentru estimarea adâncimii, în special în sisteme bazate pe o singură cameră. Potrivirea obiectelor este realizată prin calculul "Intersect over Union" aplicat la chenarele de încadrare. Pentru evaluare am filtrat unele obiecte și nu am folosit cele care sunt alungite (sistemul nostru va detecta orice tip de obstacol, uneori și bordurile sau separatoarele dintre benzi, ceea ce va afecta procesul de evaluare). Evaluarea se face pe intervale de distanță diferite, iar distanța maximă de procesare a sistemului nostru este limitată la 50 de metri.

Interval distanță (m)	**KITTI 0005 Mean Absolute Error (m)**	**KITTI 0010 Mean Absolute Error (m)**	**KITTI 0011 Mean Absolute Error (m)**	**KITTI 0018 Mean Absolute Error (m)**
0-10	-	-	0.8	-
10-20	1.30	2.6	3.29	3.91
20-30	3.65	2.74	5.59	5.59
30-40	2.07	6.03	11.72	6.2
40-50	2.08	4.35	19.06	9.32

Tabel 4. Analiza erorii MAE pe secvențe din baza de date KITTI.

Interval distanță (m)	KITTI 0005 Rata de detectie (%)	KITTI 0010 Rata de detectie (%)	KITTI 0011 Rata de detectie (%)	KITTI 0018 Rata de detectie (%)
0-10	-	-	97.56	-
10-20	100	94.73	97.25	79.2
20-30	89.62	62.78	82.95	74.4
30-40	76.95	51.85	40.73	74.71
40-50	60	5.35	31.46	47.05

Tabel 5. Rata de detecție pe diferite intervale pe secvențe din baza de date KITTI.

De asemenea, am folosit pentru evaluare și imagini cu informație 3D obținută prin stereoviziune, obținute din proiecte anterioare ale colectivului. Secvențele constau în imagini achiziționate cu un sistem de stereoviziune precis calibrat. Valorile de “ground truth” la care ne raportăm vor fi considerate cele urmărite folosind un algoritm bazat pe stereoviziune. În tabelul 6 am prezentat analiza acestei evaluări.

Interval distanță (m)	Mean Absolute Error (m)	Rata de detectie (%)
0-10	0.78	88.66
10-20	1.37	96.03
20-30	2.62	92.32
30-40	7.21	80.61
40-50	17.44	57.43

Tabelul 6. MAE și rata de detecție pe secvența proprie unde valorile de referință sunt cele dintr-un algoritm de stereoviziune.

Harta de ocupare și de măsurare acoperă o zonă de drum cu o lățime de 24 metri, ceea ce înseamnă că uneori rata de detecție este mai mică de 100%. Algoritmul de scanare folosit va introduce limitări mai ales în cazul vehiculelor care sunt parțial acoperite. Totodată, rata de detecție în intervalul de măsurare minim (0-10 metri) este afectată de punctele de contact ale obstacolelor care sunt acoperite de capota vehiculul propriu. La distanțe mai mari, eroarea unui sistem monocular este mai mare, ceea ce înseamnă că probabilitatea de ocupare a particulelor dintr-o celulă este mică și de aceea obiectele nu sunt întotdeauna detectate.

5. Îmbunătățirea performanțelor rețelei neuronale folosind informația de la estimatorul probabilistic

În afară de puterea evidentă de clasificare și de segmentare a rețelelor neuronale convoluționale, tendința recentă este de a folosi aceste rețele pentru a rezolva mai multe etape din problematica percepției mediului și chiar a conducerii automate a autovehiculelor. Dacă rețelele sunt suficient de mari, datele de antrenare în cantitate enormă, și puterea de calcul nu este un impediment, Deep Learning poate rezolva orice problemă.

Deoarece nu dispunem de aceste resurse nelimitate, am încercat totuși să experimentăm utilizarea rețelelor neuronale pentru a suplini diferite etape ale algoritmului de detecție a obstacolelor. Au fost abordate două sub-probleme: generarea automată a imaginii de tip “bird-eye” din imaginea perspectivă, și generarea automată a modelului de măsură din imaginea reproiectată. Pentru ambele abordări au fost utilizate mai multe arhitecturi de tip enconder-decoder, iar ca date de antrenare au fost utilizate datele generate de algoritmii analitici proiectați. Rezultatele preliminare sunt foarte promițătoare, după cum se poate observa în figura 13, unde rețeaua este capabilă să estimeze în mod corect zonele cu mare probabilitate de a fi obstacol, zonele libere, și zonele incerte.

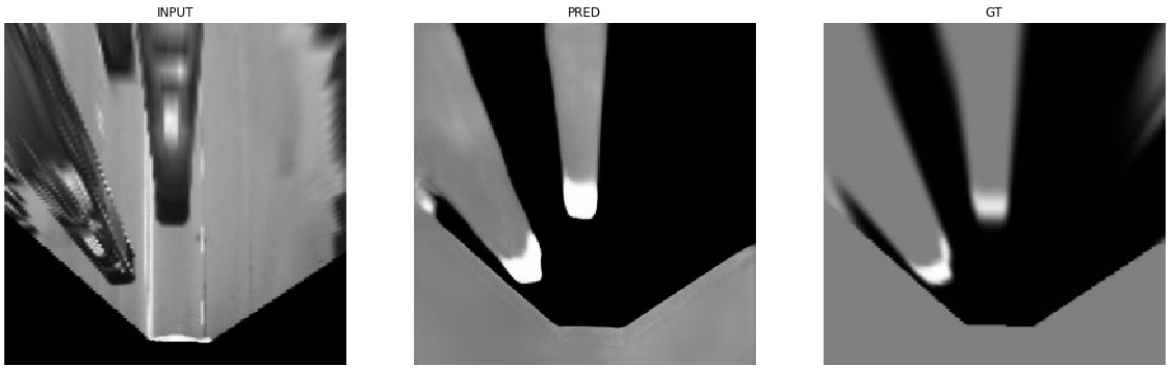

Figura 13. Generarea hărții de probabilitate (măsurare) direct dintr-o rețea artificială. Stânga: imaginea de intrare “bird’s eye view”, centru: predicția rețelei, dreapta: imaginea de adnotare (“ground truth”).

6. Realizarea aplicației demonstrator

În această fază a proiectului au fost dezvoltate două componente independente ale aplicației finale. O componentă este dedicată achiziției datelor, iar a doua componentă este dedicată procesării datelor achiziționate. În faza următoare se va realiza unificarea celor două componente pentru realizarea unui sistem care va percepe obstacolele din trafic în timp real, la bordul unui vehicul de test.

6.1. Aplicația de achiziție a datelor

Sistemul propriu pentru achiziția datelor este scris în limbajul de programare Java și utilizat pe aplicații mobile Android. Achiziția datelor pe dispozitive mobile folosește următoarele componente: camera, accelerometru, giroscop și senzorul de orientare și cel de poziționare prin satelit.

Camera principală este cea folosită pentru achiziția imaginilor, iar restul senzorilor sunt folosiți pentru a stoca sub formă de text informațiile despre poziție, orientare și viteză. Din

procesul de achiziție am reușit să obținem un total de 27 de secvențe de trafic, în condiții de iluminare diferită și în momente diferite ale zilei. În figura 14 sunt ilustrate toate traseele parcurse în împrejurul orașului Cluj-Napoca.

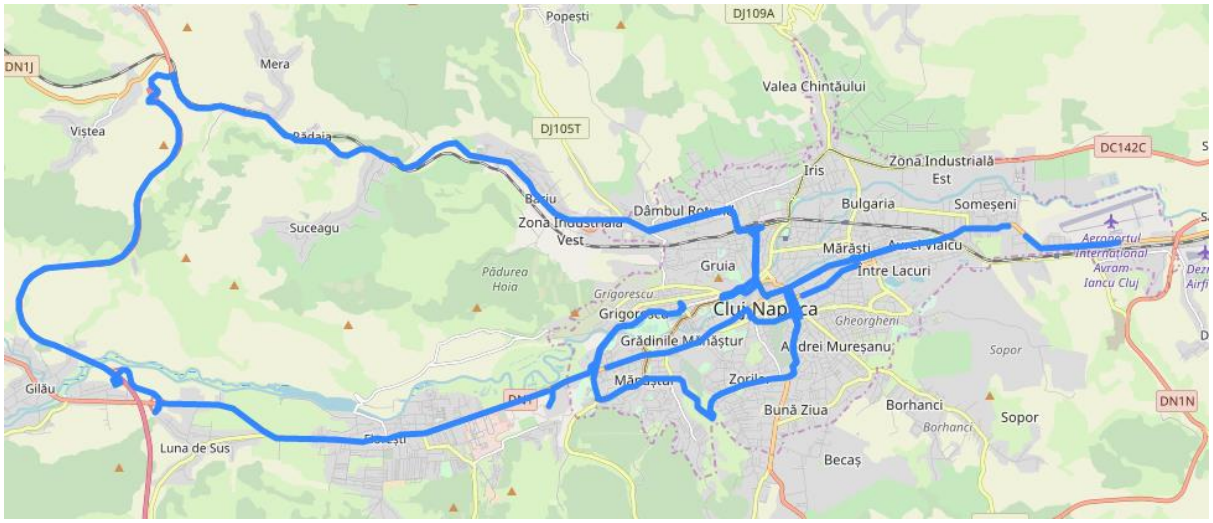

Figura 14. Traseele din GPS pentru călătoriile înregistrate în Cluj-Napoca și împrejurimi.

6.2. Aplicația de procesare a secvențelor achiziționate

Aplicația de procesare este compusă din modulul de clasificare și predicție pe bază de CNN, și implementarea algoritmilor de urmărire probabilistică. Rețelele neuronale sunt dezvoltate în Python, iar algoritmii sunt dezvoltați în C++. O comunicare între cele două procese a fost stabilită prin mecanismul de tip socket, dar această soluție este doar temporară. O altă abordare este exportarea rețelei neuronale din Python în C++, iar această soluție este mult mai eficientă din punct de vedere al vitezei de calcul și al memoriei folosite, dar datorită particularităților bibliotecilor folosite nu s-a reușit exportarea tuturor tipurilor de rețele utilizate.

Cu exportarea rețelelor în C++, întreg fluxul de procesare, incluzând segmentarea semantică, crearea de particule, actualizare și măsurare pentru particule este realizată într-un timp mediu cuprins între 70-80 ms, în funcție de numărul de obstacole din scenă. Cu optimizare și prin reducerea informațiilor de depanare care sunt afișate, acest timp ar putea fi redus suplimentar la sub 40-50 ms. Partea de segmentare semantică este realizată într-un timp mediu de 6 ms în C++ cu modelul exportat din Python.

O captură de ecran cu aplicația de procesare realizată este ilustrată în figura 15.

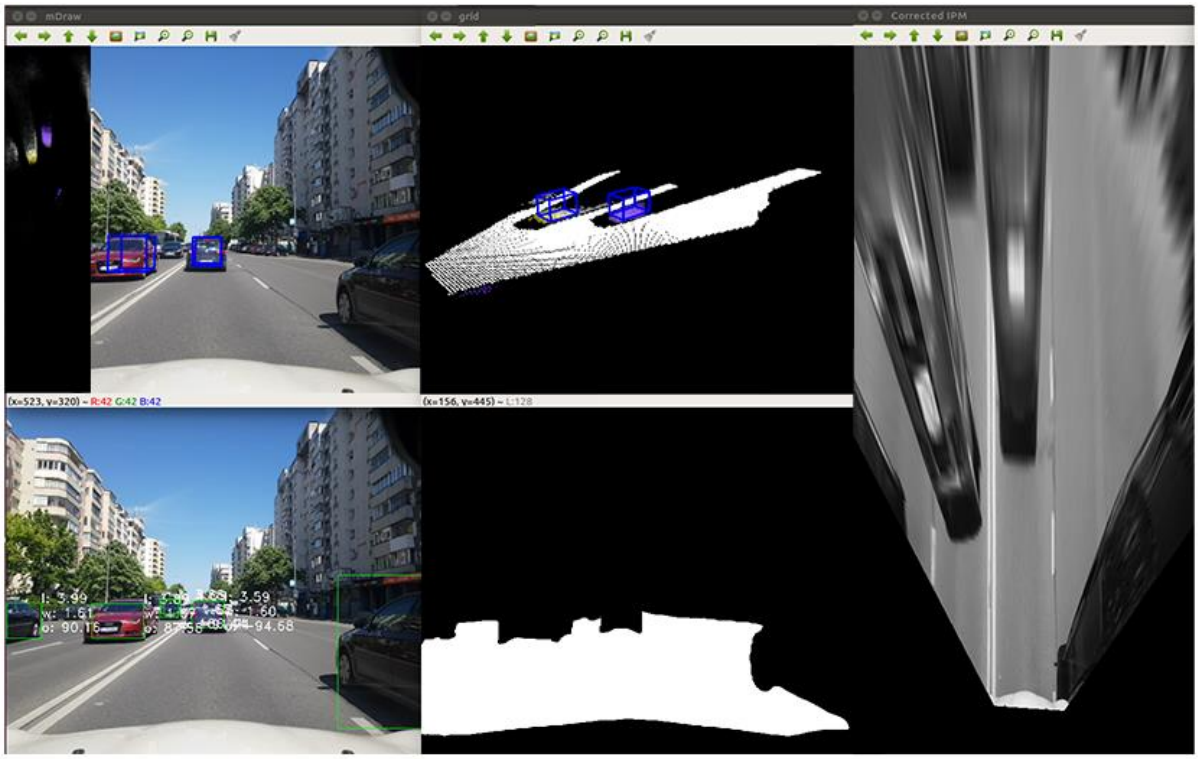

Figura 15. Sistemul de procesare în care sunt afișate cuboidele extrase (colțul din stânga sus și centru sus), chenarele de încadrare a vehiculelor (stânga jos), rezultatul segmentării (centru jos) și imaginea cu efectul de perspectivă eliminat (dreapta).

7. Diseminarea rezultatelor

Rezultatele obținute în cadrul acestei etape au fost publicate în următoarele lucrări, în volume de conferințe internaționale:

1. Radu Danescu, Razvan Itu, "Camera Calibration for CNN Based Generic Obstacle Detection", EPIA Conference on Artificial Intelligence, 2019, pp. 623-636.
2. Radu Danescu, Razvan Itu, Diana Borza, "Dynamic 3D Environment Perception Using Monocular Vision and Semantic Segmentation", 2019 IEEE 15th International Conference on Intelligent Computer Communication and Processing (ICCP 2019).
3. Sorana Capalnean, Florin Oniga, Radu Danescu, “Obstacle Detection Using a Voxel Octree Representation”, 2019 IEEE 15th International Conference on Intelligent Computer Communication and Processing (ICCP 2019).

De asemenea, abordările și rezultatele prezentate în raport au fost folosite și publicate deja într-o lucrare de doctorat cu titlul: “Sistem de percepție monoculară folosind viziune și inteligență artificială”, elaborată de As. Ing. Razvan Itu sub îndrumarea Prof. Dr. Ing. Radu Danescu.

Un articol care va fi trimis la un jurnal cotat ISI, care descrie în detaliu rezultatele obținute în această etapă, este în pregătire și va fi trimis până la sfârșitul acestui an.

8. Concluzii

În această etapă au fost rafinate și puse împreună componentele bazate pe Deep Learning și cele bazate pe modele geometrice și probabilistice, ducând la o metodă completă

și autonomă pentru detecția obiectelor 3D din traficul auto, precum și măsurarea distanțelor și a vitezelor acestora. Au fost abordate mai multe soluții pentru segmentare, calibrare și estimare, și au fost experimentate metode pentru înlocuirea diferiților algoritmi cu procesări bazate pe CNN. Integrarea sistemelor într-o aplicație demonstrator este în stadiu avansat, iar rezultatele au fost publicate în mai multe articole de conferință, urmând ca până la sfârșitul etapei (31.12.2019) să fie finalizat și trimis un articol de jurnal.

Pe baza rezultatelor obținute și descrise în prezentul raport, putem trage concluzia că obiectivele etapei 2 a proiectului DEEPSENSE au fost îndeplinite cu succes.

Bibliografie

[1] - O. Ronneberger, P. Fischer, T. Brox, "U-Net: Convolutional Networks for Biomedical Image Segmentation", Medical Image Computing and Computer-Assisted Intervention (MICCAI), Springer, vol. 9351, pp. 234-241, 2015.

[2] - E. Romera, J. M. Alvarez, L. Miguel Bergasa, R. Arroyo, "ERFNet: Efficient Residual Factorized ConvNet for Real-Time Semantic Segmentation", IEEE Transactions on Intelligent Transportation Systems, vol. 19, no. 1, pp. 263-272, 2018.

[3] - M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, B. Schiele, “The Cityscapes Dataset for Semantic Urban Scene Understanding”, Computer Vision and Pattern Recognition, pp. 3213-3223, 2016.

[4] - F. Yu, W. Xian, Y. Chen, F. Liu, M. Liao, V. Madhavan, T. Darrell, “BDD100K: A Diverse Driving Video Database with Scalable Annotation Tooling”, arXiv: 1805.04687, 2018.

[5] - G. Neuhold, T. Ollmann, S. R. Bulò, P. Kontschieder, "The Mapillary Vistas Dataset for Semantic Understanding of Street Scenes", IEEE International Conference on Computer Vision, pp. 5000-5009, 2017.

[6] - A. Geiger, P. Lenz, R. Urtasun, “Are we ready for Autonomous Driving? The KITTI Vision Benchmark Suite”, Computer Vision and Pattern Recognition, pp. 3354-3361, 2012.

[7] - L.C. Chen, Y. Zhu, G. Papandreou, F. Schroff, H. Adam, “Encoder- Decoder with Atrous Separable Convolution for Semantic Image Segmentation”, arXiv: 1802.02611, 2018.

[8] - A. Paszke, A. Chaurasia, S. Kim, E. Culurciello, "ENet: A Deep Neural Network Architecture for Real-Time Semantic Segmentation", arXiv: 1606.02147, 2016.

[9] - M. Fischler, R. Bolles, "Random Sample Consensus: A Paradigm for Model Fitting with Applications to Image Analysis and Automated Cartography", Communications of the ACM, Volume 24, pp. 381–395, 1981.

[10] - R. Danescu, R. Itu, "Camera Calibration for CNN Based Generic Obstacle Detection", EPIA Conference on Artificial Intelligence, 2019, pp. 623-636.

[11] - R. Itu, R. 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.

[12] - R. Itu, D. Borza, R. Danescu, "Automatic extrinsic camera parameters calibration using Convolutional Neural Networks", 2017 IEEE 13th International Conference on Intelligent Computer Communication and Processing (ICCP 2017), pp. 273-278.

[13] - R. Danescu, F. Oniga, S. Nedevschi, “Modeling and Tracking the Driving Environment with a Particle-Based Occupancy Grid”, IEEE Transactions on Intelligent Transportation Systems, volume 12, no. 4, pp. 1331-1342, 2011.

[14] - R. Danescu, R. Itu, A. Petrovai, "Generic Dynamic Environment Perception Using Smart Mobile Devices", Sensors, vol. 16, no. 10, art. no. 1721, 2016.

[15] - R. Itu, R. Danescu, "An Efficient Obstacle Awareness Application for Android Mobile Devices", International Conference on Intelligent Computer Communication and Processing (ICCP 2014), 2014, pp. 157-163.
[16] - M. Bertozzi and A. Broggi, “GOLD: a Parallel Real-Time Stereo Vision System for Generic Obstacle and Lane Detection”, IEEE Trans. on Image Processing, pp. 62-81, 1998.
[17] - R. Danescu, R. Itu, D. Borza, "Dynamic 3D Environment Perception Using Monocular Vision and Semantic Segmentation", 2019 IEEE 15th International Conference on Intelligent Computer Communication and Processing (ICCP 2019).

28.11.2019,
Director proiect
Prof. Dr. Ing. Radu Danescu