Roboţii şi Societatea:
Sisteme Cognitive pentru Roboţi Personali şi Vehicule Autonome

Număr contract 72PCCDI **din** 01/03/2018

Etapa I

Raport ştiinţific şi tehnic

Consorțiu
Coordonator proiect: UNIVERSITATEA POLITEHNICA DIN BUCURESTI
P1: INSTITUTUL DE CERCETARI PENTRU INTELIGENTA ARTIFICIALA „MIHAI DRAGANESCU"
P2: INSTITUTUL DE MATEMATICA "SIMION STOILOW" AL ACADEMIEI ROMANE
P3: UNIVERSITATEA TEHNICA DIN CLUJ - NAPOCA
P4: UNIVERSITATEA "DUNAREA DE JOS"

Rezumat

ROBIN este un proiect centrat pe utilizator care proiectează sisteme şi servicii pentru utilizarea roboților într-o societate digitală interconectată și permite companiilor realizarea de produse și servicii complexe, inteligente și performante destinate utilizatorilor și societății în ansamblu. Proiectul se referă la o gamă diversă de roboţi: roboţi asistivi pentru sprijinul persoanelor cu nevoi speciale, roboţi de interacţiune cu clienţii şi roboţi software care pot fi instalaţi pe vehicule în scopul realizării unei conduceri autonome sau semi-autonome. Proiectul combină tehnici și tehnologii avansate de inteligență artificială, interacțiune om-robot, interacțiune cu un mediu pervasiv și prelucrări în Cloud.

Proiectul ROBIN este alcătuit din cinci proiecte componente:

ROBIN-Social: ***Roboți sociali cognitivi pentru o societate digitală a viitorului, centrată pe utilizator*.** Scopul proiectului este realizarea unei soluții integrate și ușor configurabile pentru personalizarea roboților asistivi (pt. persoane cu nevoi speciale) și sociali (pt. clienți), soluție bazată pe tehnici avansate de inteligență artificială care pot oferi roboților un caracter cognitiv și autonom.

ROBIN-Car: ***Sisteme cognitive autoinstruibile pentru vehicule autonome.*** Scopul proiectului constă în dezvoltarea de metode de vedere computaţion- ală care să rezolve o gamă mai largă şi mai sofisticată de sarcini de asistenţă în pilotaj, realizarea unor module inteligente pentru „Hands-off driving" și „Automated driving" și un sistem prototip care va fi testat pe un autovehicul electric semi-autonom.

ROBIN-Context: ***Servicii inteligente, dependente de context, pentru personalizarea roboţilor şi conducere autonomă.*** Scopul proiectului este crearea unei platforme suport pentru definirea / reprezentarea semantică și gestiunea facilă și eficientă a datelor ce devin context în scenarii de asistență robotică personalizată și sisteme ADAS (Advanced Driving Assistance Systems).

ROBIN-Dialog: ***Înțelegerea și sinteza limbajului natural pentru roboți asistivi și interacțiunea cu mediul.*** Scopul proiectului presupune dezvoltarea unei serii de scenarii pentru câteva micro-lumi și tehnologia de prelucrare a limbii române pentru dialoguri situaționale în aceste micro-lumi.

ROBIN-Cloud: ***Platformă informatică distribuită pentru sisteme pervasive, în contextul sistemelor Cloud-Edge***. Scopul proiectului îl constituie crearea unei platforme suport pentru colectarea de date provenind de la senzorii unor sisteme suport pentru roboți (ex. IoT), oferirea de mecanisme de procesare și învățare combinând modele Cloud cu dispozitive aflate aproape de sursa de colectare, oferirea de biblioteci suport pentru procesare inteligentă / semantică a datelor, și în final dezvoltarea de servicii Web centrate spre agenți economici interesați de folosirea datelor stocate la nivelul platformei.

Site proiect http://aimas.cs.pub.ro/robin/

CUPRINS

1. Introducere

ROBIN este un proiect centrat pe utilizator care proiectează sisteme şi servicii pentru utilizarea roboților într-o societate digitală interconectată și permite companiilor realizarea de produse și servicii complexe, inteligente și performante destinate utilizatorilor și societății în ansamblu.

Proiectul se referă la o gamă diversă de roboţi: roboţi asistivi care vin în sprijinul persoanelor cu nevoi speciale, roboţi de interacţiune cu clienţii şi roboţi software care pot fi instalaţi pe vehicule în scopul realizării unei conduceri autonome sau semi-autonome.

Proiectul combină tehnici și tehnologii avansate de inteligență artificială, interacțiune om-robot, interacțiune cu un mediu pervasiv, arhitecturi și prelucrări în Cloud, arhitecturi Cloud-Edge si Cloud-Robotics.

Figura 1.1 prezintă viziunea generală a proiectului complex din punct de vedere al aplicațiilor dezvoltate și al tehnologiilor utilizate. Se pune astfel în evidență interacțiunea cu roboții de tip umanoid sau semiumanoid pentru realizarea roboților asistivi, dar și cu roboții instalați pe autovehicule în scopul conducerii autonome sau semi-autonome, și se reliefează principalele tehnologii utilizate, de exemplu tehnologii de computer vision bazate pe rețele de convoluție, tehnologii de tip Cloud și Cloud-Edge, și tehnologii de integrare a datelor provenite de la sensori.

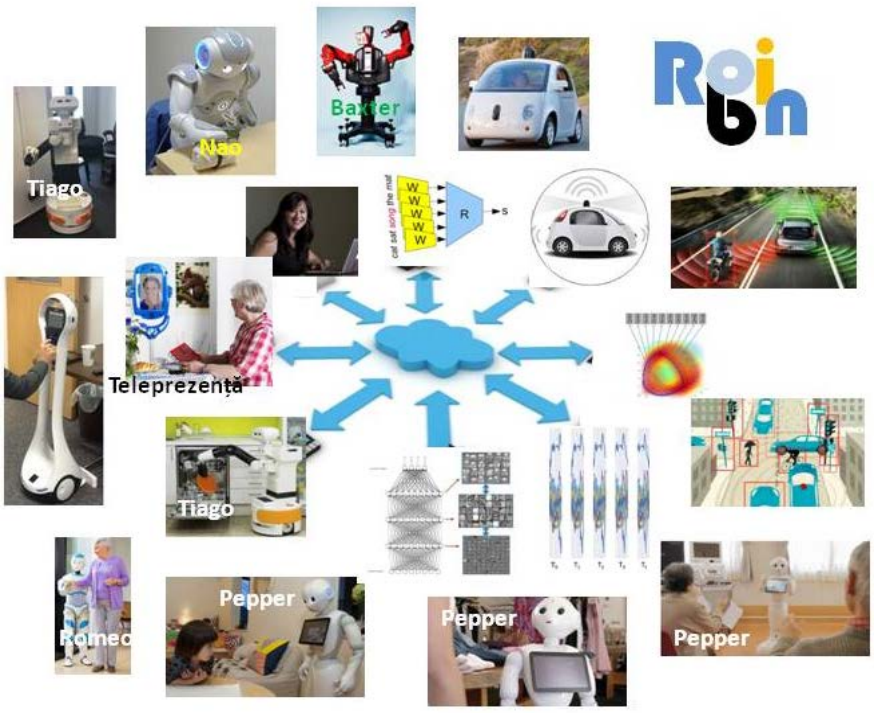


Figura 1.1.Roboții și societatea

Figura 1.2 prezintă conceptul general al proiectului, atât funcțional cât și arhitectural, prin care avem în vedere dezvoltarea de sisteme și servicii științifice și tehnologice performante, construite peste platforme interoperabile, cu capacități de structurare și procesare inteligentă a datelor și oferite comunității științifice și agenților economici.

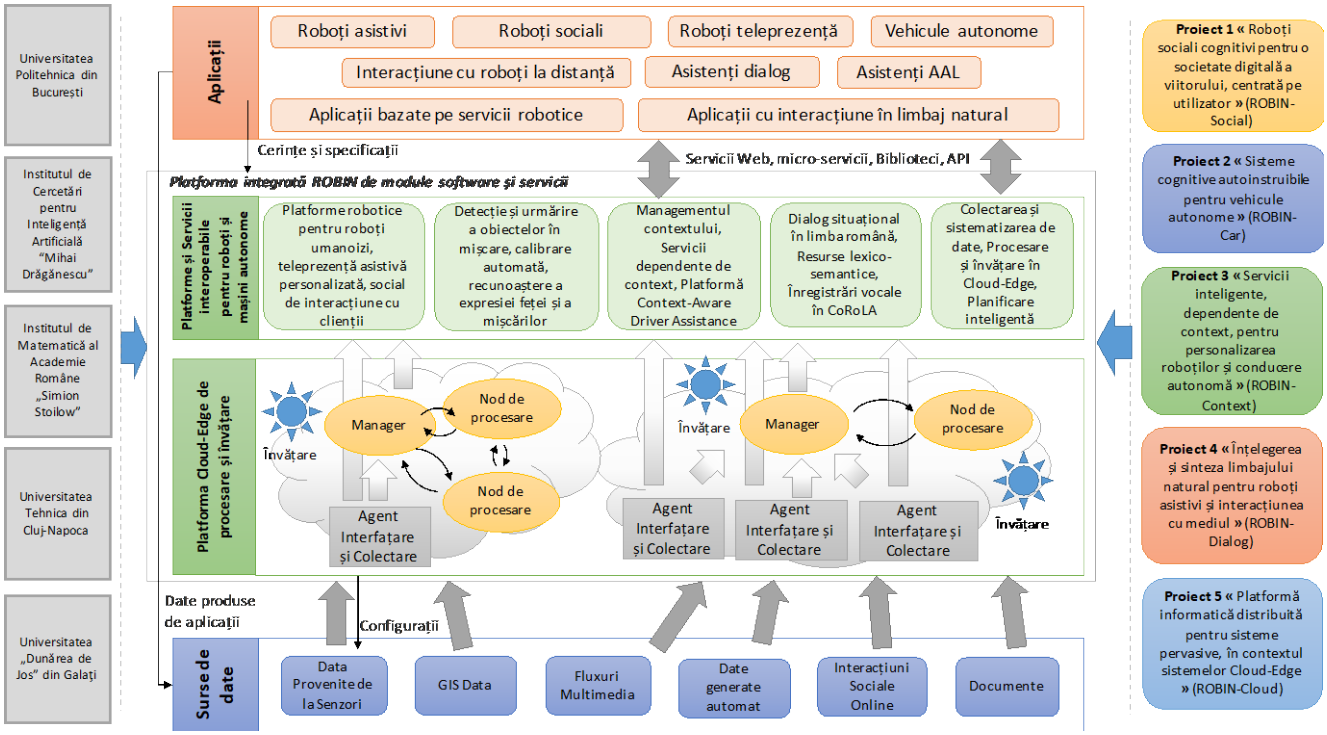


Figura 1.2. Prezentarea conceptului integrat a proiectului ROBIN

Figura 1.2 indică de asemenea cele 5 proiecte componente ale proiectului complex ROBIN, și partenerii din consorțiu, cât și componentele principale ale viitoarei platforme integrate în care se vor regăsi serviciile dezvoltate în proiectele componente.

2. Proiectul component ROBIN-Social

Parteneri ai proiectului ROBIN-Social

UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI (CO)
INSTITUTUL DE CERCETĂRI PENTRU INTELIGENȚĂ ARTIFICIALĂ „MIHAI DRĂGĂNESCU"
INSTITUTUL DE MATEMATICĂ "SIMION STOILOW" AL ACADEMIEI ROMÂNE

2.1 Prezentare generală

În ultimii ani roboții autonomi au început să devină o prezență activă în viața de zi cu zi, în forme diverse. Roboții asistivi [Payr et.al. 2015], care oferă ajutor și suport persoanelor cu nevoi speciale, răspund la o necesitate din ce în ce mai crescută în contextul unei societăți care dorește și promovează o viață mai bună. Domeniul Active Assistive Living[1] îmbină utilizează tehnicile inteligenței ambientale pentru sprijinirea persoanelor în vârstă și a celor cu nevoi speciale. Recent, există o serie de proiecte care au început utilizarea roboților umanoizi sau semiumanoizi în aplicații de AAL.

Roboții sociali [Feil-Seifer&Mataric 2005], capabili să interacționeze cu clienții în magazine sunt din ce în ce mai apreciați de companii, oferind clienților o experiență interesantă și personalizată. Provocările de bază care stau în fața realizării unor astfel de roboți din punct de vedere al software existent sunt legate de: capacitatea de orientare și navigare în medii necunoscute, dinamice și/sau populate intens, capacitatea de identificare a persoanei cu care interacționează (vedere artificială,

[1] http://www.aal-europe.eu/

identificarea vorbitorului, etc.) în condițiile existenței unui grup de persoane și a zgomotului ambiental, interacțiunea în limbaj natural cu utilizatorii și persoanele cu nevoi speciale.

Scopul proiectului ROBIN-Social este realizarea unei soluții integrate și ușor configurabile pentru personalizarea roboților asistivi (pt. persoane cu nevoi speciale) și sociali (pt. clienți), soluție bazată pe tehnici avansate de inteligență artificială care pot oferi roboților un caracter cognitiv și autonom. Prin utilizarea unor algoritmi robuști de învățare automată, vedere computerizată și prelucrare a limbajului natural, inclusiv cel vorbit, cât și prin combinarea algoritmilor de pe robot (de ex. SLAM, recunoaștere persoană) cu date provenite de la dispozitive IoT, dar și prin utilizarea unor soluții de tip Cloud-Robotics[2], proiectul are ca obiectiv depășirea limitărilor existente în roboții comerciali (umanoizi sau semi-umanoizi) și personalizarea acestora pentru aplicații specifice.

Până în prezent nu există o astfel de soluție integrată care să poată fi configurată atât pentru roboți asistivi cât și sociali. În plus, toate soluțiile existente au limitări din punct de vedere al funcționalităților și a capacității de portare și configurare.

În acest context, ***obiectivele proiectului ROBIN-Social*** sunt:

- Realizarea unei platforme de dezvoltare a aplicațiilor robotice pentru diferite modele de roboți umanoizi sau de tip teleprezență, de ex. asistarea persoanelor în vârstă sau cu afecțiuni la domiciliu, asistența clienților, platformă care să permită integrarea dispozitivelor IoT pentru creșterea acurateței de operare și utilizarea resurselor din Cloud pentru calcule intensive.
- Dezvoltarea unor algoritmi performanți de SLAM, 3D-mapping și fuziune a datelor (data fusion) cu datele de la senzori care să permită autonomie pe termen lung într-un mediu dinamic (oameni și obiecte în mișcare).
- Dezvoltarea unor algoritmi de planificare a mișcărilor robotului, bazați pe tehnici de inteligență artificială, considerând candidați cum ar fi rețele neurale adânci sau învățarea prin recompensă.
- Utilizarea și adaptarea algoritmilor de vedere computațională pentru detecția și recunoașterea persoanelor.
- Utilizarea și adaptarea sistemului de dialog în limba română din ROBIN-Dialog pentru interacțiunea cu roboții în limbaj natural
- Realizarea unui produs de tip teleprezență asistivă autonomă, cu un hardware comercial
- Realizarea unui produs de tip robot asistiv pentru persoane cu nevoi speciale, cu un robot comercial
- Realizarea unui produs de tip robot social de interacțiune cu clienții într-un supermarket, cu un robot comercial.

Activitățile Etapei I a proiectului au fost următoarele

Activitatea: ***Act 1.1 - Definirea specificațiilor funcționale și arhitecturale ale platformei robotice*** Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: L1-Specificațiile funcționale și arhitecturale ale platformei robotice.

Obiectivul a fost integral îndeplinit, vezi secțiunea 2.2 din prezentul raport și raportul in extenso.

Activitatea: ***Act 1.2 - Definire use-case-uri roboți asistivi și sociali*** Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: L2 - Descriere use-case-uri

Obiectivul a fost integral îndeplinit, vezi secțiunile 2.3 și 2.4 din prezentul raport și raportul in extenso.

Activitatea: ***Act 1.3 - Proiectarea si implementarea algoritmilor pentru SLAM, 2D/3D-mapping și data fusion, detecția și recunoașterea persoanelor și a activităților*** Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: L3 - Descriere algoritmi și a performanțelor atinse

[2] http://roboearth.ethz.ch/cloud_robotics/index.html

Indicatori de realizare: 2 module software funcționale (SLAM+2D/3D-mapping+miscare robot, detectia si recunoașterea persoanelor)

Obiectivul a fost integral îndeplinit, vezi secțiunile 2.5, 2.6 și 2.7 din prezentul raport și raportul in extenso.

Activitatea: ***Act 1.4 - Diseminare*** Categorie activitate: D1 - Activități suport - Diseminarea pe scară largă prin comunicarea și publicarea națională sau internațională a rezultatelor Indicatori de realizare: 1 lucrare științifică

Obiectivul a fost integral îndeplinit.

O lucrare acceptată, cu modificări, *Spatio-Temporal Aspects of Action Recognition using 3D Skeletal Joints* la revista Sensors (IF 2,475), Manuscript ID: sensors-387078, Status: Accepted with Major Revision

A fost acceptat capitolul de carte *An Integrated System for Improved Assistive Living of Elderly People*, în volumul Recent Advances in Intelligent Assistive Technologies: Paradigms and Applications, care va fi publicat in Springer in 2019.

S-a publicat lucrarea științifică *Remotely Operated Robot with Live Camera Feed*, International Conference on Cyber Systems on the fields of Aerospace, Robotics Manufacturing Systems Mechanical Engineering, Neurorehabilitation, Power Energy and Technology of Materials, Noiembrie 2-4 2018, București, România

În cadrul primei etape a proiectului ROBIN-Social am pornit de la cercetările noastre anterioare pentru detecția și recunoașterea de persoane, implementate pe un robot Pepper[3], și am utilizat și rezultatele unei soluții AAL ce integrează comanda la distanță a dispozitivelor domotice și monitorizarea unor parametrii de sănătate a pacientului, dezvoltată în cadrul proiectului CAMI[4]. Provocările de bază ale actualei etape au fost extinderea, îmbunătățirea și evaluarea unui modul de detecție și recunoaștere de persoane, îmbunătățirea capacității de localizare a robotului prin adăugarea de componente externe în scopul eliminării limitărilor de localizare implicită a robotului, localizarea fiind esențială în interacțiunea cu oamenii, proiectarea modulelor de comportament ale robotului și implementarea lor pe robot în cadrul a 2 scenarii de test, proiectarea modului de planificare care permite definirea ciclului de viață al robotului.

2.2 Descrierea platformei AMIRO

2.2.1 Arhitectura generală

Sistemul este organizat sub forma unei arhitecturi modulare, cu componente independente conectate prin intermediul unității de planificare. Modularitatea sistemului oferă atât posibilitatea de a rula individual fiecare modul, cat și posibilitatea de a le înlocui cu ușurință. Fiecare componenta trebuie sa implementeze metode specifice corespunzătoare sarcinilor pe care trebuie sa le execute. Figura 2.1 jos prezintă arhitectura și conexiunea intre componentele principale ale sistemului: navigare, vedere computațională, vorbire și planificare.

Modulul de navigare se ocupa sarcinile de mișcare a robotului prin mediu. Componenta de vedere computațională este responsabila cu interpretarea vizuala a mediului. Componenta de vorbire interpretează limbaj natural și reproduce text în vorbire. Modulul de planificare preia comenzi care vin de la utilizator prin mai multe interfețe grafice, și le atribuie celorlalte module. Comportamentul robotului este o combinație intre aceste sarcini de baza.

[3] https://www.softbankrobotics.com/emea/en/robots/pepper

[4] http://www.camiproject.eu/

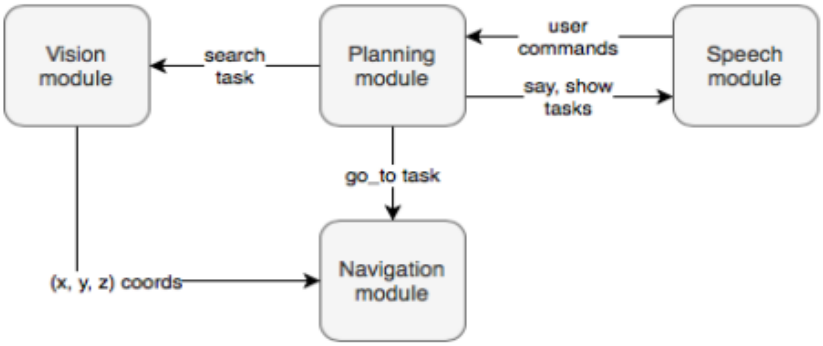


Figura 2.1. Platforma AMIRO

2.2.2 Modulele componente

Modulele componente principale ale arhitecturii sunt: navigare, vedere computationala, vorbire, planificare si interacțiune cu mediul.

Navigarea robotului este esențiala in orice scenariu care implica un robot autonom. Intr-un scenariu in care robotul funcționează ca un asistent care oferă ajutor persoanelor cu care interacționează, comportamentul acestuia trebuie sa fie suficient de precis incat interacțiunea sa cu utilizatorul sa fie naturala si sa aibă un timp de răspuns bun in cazul unei situații neprevăzute. De aceea o harta a mediului este necesara pentru o planificare optima a caii acestuia. In plus fata de asta, robotul trebuie sa tina cont de schimbările care se pot produce in mediu, si sa folosească o metoda dinamica de reprezentare a schimbărilor. Principalul scop al modulului in cadrul proiectului este sa facă o reprezentare a mediului in care se afla robotul, sa plaseze robotul in cadrul harții, si tine minte pe harta unde se afla obiectele detectate,

Din punct de vedere fizic, modulul se bazează pe un senzor LiDAR auxiliar, care poate fi montat pe orice robot. Acest lucru a fost necesar din cauza limitărilor in ceea ce privește distanta si precizia senzorilor roboților, in cazul nostru a robotului Pepper. Achiziția de date se face folosind un RaspberryPi 3 care executa un nod de ROS si care trimite datele mai departe către mașina pe care este pus întreg sistemul. Mașina executa algoritmi de SLAM pentru a localiza robotul si pentru a construi harta.

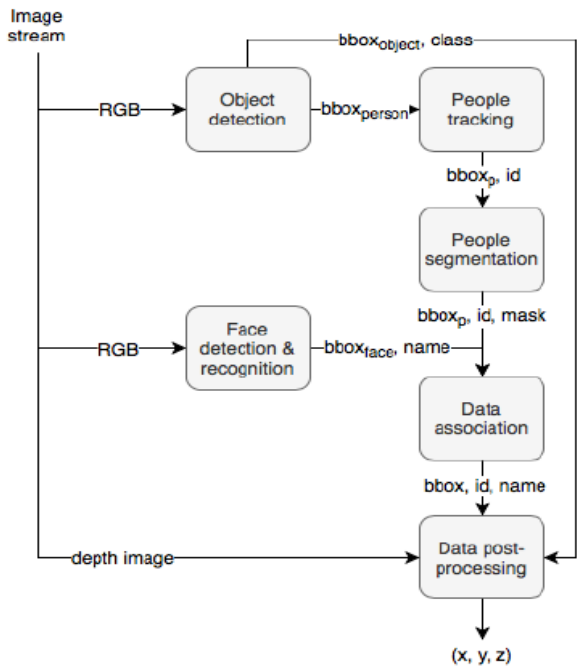


Figura 2.2 Modulul de vedere computațională

Modulul de vedere computațională este organizat ca o secvența de tehnici pentru detecția si identificarea oamenilor, obiectelor si codurilor QR. Principalul scop al modulului este sa identifice corect obiectele in imagini si sa estimeze poziția 3D a acestora relativ la poziția robotului, pentru a le putea reprezenta mai departe pe harta. Submodulele si modul in care acestea interacționează poate fi văzut in figura 2.2.

Procesarea imaginilor începe cu detecția de obiecte si recunoașterea de fete, informație care este apoi comprimata intr-un singur set de date. Detecția de obiecte se face folosind rețeaua YOLO, in timp ce recunoașterea de fete se face folosind rețeaua FaceNet. Daca printre detecții exista si detecții de persoane, atunci pentru acele detecții in particular se executa 2 pași auxiliari: segmentare si urmărire. In pasul de segmentare se separa in zona de detecție a imaginii pixelii persoanei de pixelii mediului, in timp ce in pasul de urmărire se asociază un număr de identificare detecției. Astfel ca, daca aceeași persoana a fost detectata in imaginea anterioara, atunci ea va primi numărul de identificare asociat anterior, altfel va primi unul nou.

Componenta de vorbire este importantă in dezvoltarea unui robot social. Modulul suporta momentan 2 limbi de vorbire, engleza si romana. Fluxul audio este preluat de la robot, trecut prin modulul de recunoaștere vocala ce îl transforma in text. Textul este apoi procesat de un modul de înțelegere a textului ce îl clasifica in ceea ce se numește „intent" apoi sunt extrase entitățile importante. Modulul de înțelegere a textului permite flexibilitate in exprimarea unei comenzi. Un exemplu pentru utilizarea modulelor este: „Pepper, ridica jaluzelele din lateral te rog", unde clasificarea este „actuare", iar entitățile esențiale sunt „ridica" pentru a înțelege acțiunea și „jaluzelele din lateral" pentru a înțelege subiectul acțiunii. In acest fel robotul va putea efectua aceasta acțiune si daca utilizatorul exprima comanda intr-un mod diferit, precum „jaluzelele sus". Pentru realizarea celor doua module s-au folosit API-uri externe precum Google Speech Recognition[5] si Wit.ai[6]. Problema solutiei actuale este limitarea numarului de interogari ce pot fi facute catre cele doua servicii, precum si faptul ca in solutia actuala robotul trebuie sa aiba o conexiune la internet. De asemenea intarzierile cauzate de latenta retelei fac interactiunea cu robotul usor sacadata, motiv pentru care este necesara o solutie ce poate rula local pentru realizarea recunoasterii vocale.

Modulul de planificare este responsabil de toata logica de interactiune a robotului. Acesta se bazează pe existenta unui set de comenzi de baza dintre care amintim:

- Vorbire: robotul va reda audio un text
- Muta la: robotul se va muta la o coordonata sau la o poziție predefinita a unei persoane sau obiect
- Căuta: robotul va roti capul si va folosi modulul de vedere computațională pentru detecția si recunoașterea persoanelor
- Afișează: robotul va afișa pe tableta o pagina HTML sau o imagine
- Asculta: robotul va asculta fluxul audio pentru a recunoaște text si a putea interacționa cu utilizatorul
- Actueaza: robotul va putea interacționa cu actuatori precum jaluzele si lumini

Aceste comenzi sunt combinate de modulul de planificare pentru a realiza comportamente. Un comportament este un flux de lucru mai complex al robotului. Un exemplu de comportament ce face robotul sa caute utilizatorul in mediu poate fi văzut in figura 2.3.

Comportamentul începe printr-o confirmare a robotului, apoi robotul merge la ultima poziția cunoscuta a țintei si începe acțiunea de a căuta. Daca ținta nu este găsita, robotul merge la poziția obișnuita a țintei si începe din nou o căutare. Oricând ținta este identificata, robotul va spune ca a fost găsita si comportamentul se termina cu succes. Oricând o eroare este întâmpinata sau ținta nu este găsita in niciuna din cele doua poziții, robotul va anunța si va opri comportamentul. Orice comportament poate fi întrerupt si poate fi reluat mai târziu in cazul in care o acțiune cu o prioritate mai mare intervine in sistem. Planificarea realizată este numai una preliminară pentru a evalua mai bine comportamentul și interacțiunea în medii reale. În etapele următoare vom dezvolta un nou modul de planificare, dinamic și care va încorpora și tehnici de învățare automată.

[5] https://cloud.google.com/speech-to-text/
[6] https://www.wit.ai/

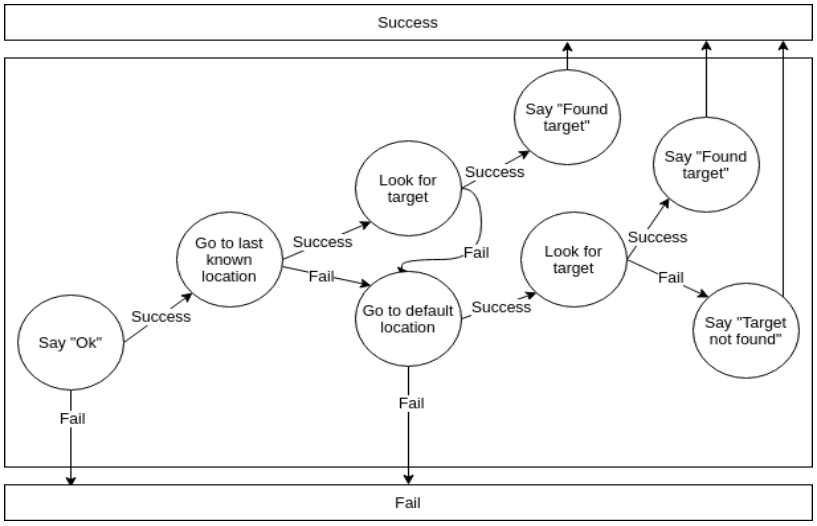


Figura 2.3 Plan comportamental al robotului

Modulul de interacțiune preia comenzi de actuare si realizează achiziția de date de la mediul înconjurător si sisteme de management a stării de sănătate a pacienților. Interacțiunea cu medul înconjurător se face prin expunerea unor noduri ROS prin care orice robot se poate conecta si poate trimite comenzi si cere informații. De la sistemul de management al sanatatii, robotul poate prelua informații legate de notificări pe care un utilizator le are precum si elemente de sănătate precum puls, tensiune, greutate, număr de pași realizați si numărul de ore dormite. Acestea sunt afișate pe tableta robotului in momentul in care utilizatorul cere acest lucru. Doua exemple de actuatori existenți in sistem sunt jaluzele inteligente si lumini inteligente. Robotul poate primi comenzi vocale de ridicare, coborâre a jaluzelelor, ridicare si coborâre a nivelului de luminozitate, oprire si pornire a luminilor si de schimbare a culorii becului inteligent. De asemenea robotul poate prelua date legate de temperatura, presiune atmosferica si umiditate ale mediului.

2.3 Robotul de dezvoltare și test

Robotul pe care îl folosim pentru dezvoltarea proiectului este Pepper, existent la partenerul UPB înainte de începerea proiectului. Pepper este un robot comercializat de compania SoftBank Robotics[7], si este folosit în special pentru interacțiuni cu oamenii, având un aspect umanoid plăcut. Robotul Pepper este bazat pe sistemul de operare NAOqi, un sistem proprietar al companiei. Deși robotul Pepper este dotat cu sistemul de operare NAOqi, s-a instalat sistemul de operare ROS peste acest sistem și dezvoltările s-au făcut și se vor face în ROS (Robot Operating System)[8]. Sistemul ROS este un sistem de operare open-source care funcționează pe multe modele de roboți, atât umanoizi cât și roboți cu capacități de manipulare avansată, și beneficiază de o comunitate de dezvoltatori și utilizatori puternică. Cadrul ROS este un efort colaborativ, în care diferite laboratoare de robotică publica rezultate pe care le-au obținut în activitatea de cercetare. Astfel, laboratoarele care au experiență în vedere artificială pot completa activitatea celor care au experiență în navigarea și controlul robotului. NAOqi se bucură de asemenea de o comunitate de dezvoltatori, aceasta este mai puțin populară decât cea a ROS. Mai mult, aplicațiile NAOqi pot rula numai pe roboți produși de Aldebaran și Softbank: Nao, Pepper și Romeo. Dezvoltările noastre făcute sus sistemul ROS vor fi astfel portabile pe o gamă largă de roboți.

Dimensiunile robotului Pepper sunt 1210 mm înălțime, cu o bază de 480 x 425 mm. Folosindu-și mâinile poate ajunge la 1335 mm înălțime, lățime de 1196.9 mm și o lungime 648,2 mm. Placa de bază a lui Pepper conține un procesor Intel Atom E3845, 4GB de memorie RAM și o placă grafică integrată de tip Intel HD Graphics. Pentru stocarea de date pe termen lung sunt folosite o memorie flash eMMC de 8GB și un card SDHC de 16 GB. Robotul permite conectivitate prin Ethernet și Wifi.

[7] https://www.softbankrobotics.com/emea/en/index

[8] http://www.ros.org/

Dispune, de asemenea, de o tableta cu o rezoluție de 1280x800 pixeli ce poate rula aplicații independente pe sistemul de operare Android. Pepper are o multitudine de senzori ce îi permit să interacționeze cu mediul: o unitate inerțială, LIDARe, sonare (pentru estimarea distanței la obstacole din mediu) și senzori de infra-roșu.

Figura 2.4 Robotul Pepper

Pentru interacțiunea cu utilizatorii, Pepper dispune de senzori tactili (pe cap și pe mâini), precum și de 3 camere montate pe cap: 2 camere 2D și una 3D (cu senzor de adâncime). Pepper poate percepe și produce sunete prin patru microfoane și două difuzoare. Din punct de vedere al mișcărilor, și implicit motoare, Pepper încearcă să semene cu cât mai mult posibil cu oamenii, permițând mișcări ale capului, mâinilor și șoldurilor.

2.4 Scenarii de evaluare și testare

In vederea realizării setului de comportamente complexe ale roboților, s-au luat in vedere doua scenarii de utilizare distincte ale robotului.

Primul scenariu este legat de asistenta persoanelor cu nevoi speciale de către un robot asistiv. Robot va fi conectat la o platforma de management al sanatatii de unde poate extrage informații relevante cu privire la starea de sănătate si a problemelor medicale ale utilizatorului. Intrucat multe din activitățile zilnice ale persoanelor cu nevoi speciale sunt esențiale pentru imbunatatirea stării de sănătate, robotul trebuie sa se asigure ca utilizatorul le îndeplinește cu succes. De asemenea robotul trebuie sa poată sa afișeze si sa notifice utilizatorul cu privire la alerte legate de indici de sănătate precum puls sau tensiune arteriala mărita. Astfel primul scenariu se conturează astfel:

„Mihai are pulsul mărit. O alerta cu privire la acest lucru este transmisa de la sistemul de management al sanatatii la robot. Robotul preia alerta si începe comportamentul de căutare a lui Mihai. Acesta se muta către ultima poziție unde l-a întâlnit pe Mihai si se uita daca il poate identifica. Mihai însa nu mai este acolo. Robotul merge acum către camera lui Mihai unde intra si începe din nou căutarea. Acesta îl recunoaște pe Mihai stand pe pat si îl informează cu privire la faptul ca are pulsul mărit si ca ar trebui sa stea liniștit o perioada. Mihai mulțumește robotului, iar robotul răspunde.”

In cel de-al doilea scenariu, robotul este plasat in locul in care se desfășoară un eveniment. Odată început evenimentul, robotul așteaptă în zona de intrare a participanților. În momentul în care în proximitatea lui apar persoane neidentificate, acesta îi întâmpina și le transmite vocal informațiile generale legate de eveniment, apoi robotul va putea oferi informații suplimentare utilizatorilor la cerere în următoarele doua forme.

În momentul în care un participant îi cere robotului sa îi arate unul din exponatele predefinite în etapa pregătitoare, robotul îi cere utilizatorului sa îl urmeze și îl conduce în zona în care a fost configurată locația, începând căutarea identificatorului pentru prezentarea obiectului. După găsirea acestuia, Pepper prezintă informațiile referitoare la acesta, mulțumește pentru atenție și revine în poziția inițială. În cea de-a doua formă, utilizatorul poate cere robotului să îl urmeze pentru a îi putea cere informații. Robotul identifică și memorează utilizatorul și începe să îl urmeze. În momentul în care robotul a ajuns în locația dorită, utilizatorul îl va informa. Acesta începe căutarea exponatul cel mai apropiat și oferă informațiile pre-configurate. Acest comportament este de asemenea folosit în cazul în care utilizatorul îi cere robotului să facă o fotografie. Utilizatorul poate cere robotului să îl urmeze la locul în care dorește să realizeze fotografia, robotul realizează fotografia și o prezintă utilizatorului prin intermediul tabletei incorporate până când acesta este mulțumit de rezultat.

Dacă în oricare dintre comportamente descrise mai sus, utilizatorul cere asistență de la o persoană responsabilă de eveniment, robotul va notifica personalul responsabil. Daca nici o persoană nu răspunde la notificare, robotul va căuta și recunoaște persoana dorită pe baza memorării anterioare în etapa de pregătitoare. Robotul caută o perioadă predefinită și dacă o găsește, o va ruga să îl urmeze. În momentul în care vor ajunge la participantul la eveniment, Pepper va introduce persoana responsabilă

de eveniment. După terminarea scenariilor descrise, robotul va reveni la o poziția din locația inițială, revenind în etapa de întâmpinare a participanților. Pe tot parcursul etapei de întâmpinare, robotul va memora persoanele ce au intrat la eveniment și au fost deja întimpinate pentru a nu repeta introducerea.

2.5 Modulul de navigare

Planificarea acțiunilor de navigare online a mediilor de interior necesita construirea unei hărți a mediului înconjurător. Robotul trebuie sa își cunoască întotdeauna poziția pe harta, sa poată estima poziția persoanelor și a obiectelor pe harta și sa poată găsi cea mai scurta cale către o poziție oarecare. In vederea realizării **modulului de localizare și mapare** (SLAM) s-a folosit Hector SLAM[9]. Acesta creează o harta 2D sub forma unei grile de ocupanta, i.e. o discretizare rectangulara a mediului, unde fiecare căsuța poate avea doua stări, anume: liber sau ocupat.

Avantajul principal al acestei abordări este dat de faptul ca Hector SLAM necesita resurse computaționale mai resurse si prezintă deviații mai mici în comparație cu ale abordări precum GMapping sau Karto SLAM. In schimb, dezavantajele sunt date de faptul ca harta rezultata nu poate fi ușor extinsa in 3D si de asemenea, nu poate fi îmbunătățita prin revizitarea locațiilor deja cartografiate. In etapa următoare, modulul de cartografiere va fi realizat utilizând framework-ului RTAB pentru cartografierea 3D a mediului înconjurător folosind senzori RGB-D care poate susține procesul de localizare și mapare simultana prin îmbunătățire continua hărții. Totodată, modulul îmbunătățit va urmări cartografierea bazata pe aspectul exterior astfel încât reprezentarea mediului sa poată fi augmentata semantic prin utilizarea unor algoritmi pentru detecția directa obiectelor și a persoanelor.

2.6 Modulul de identificarea persoanelor

Modulul de identificare de persoane preia date de pe un nod de ROS care transmite un flux de imagini RGB și de adâncime, la o rezoluție de 640x480, respectiv 320x240 pixeli și la o viteza de 20 de cadre pe secunda. Identificarea persoanelor presupune o secvența de tehnici de vedere computațională, cu rezultate precise obținute în timp real, sistemul reușind sa proceseze aproximativ 3 cadre pe secunda. Procesarea în cadrul acestei componente este făcuta pe o mașină separata, cu putere computațională mare.

Rețeaua YOLO este folosita pentru detecție de obiecte, implicit persoane, pentru ca se potrivește bine cerințelor sistemului, fiind capabilă să obțină rezultate rapide cu precizie mare in mai multe medii și condiții. Recunoașterea fețelor este folosită pentru a identifica persoana cu care robotul interacționează. Pentru recunoașterea fețelor se folosește rețeaua FaceNet, care oferă rezultate precise într-o varietate de condiții de luminozitate și orientări ale feței. Rezultatul rețelei include poziția feței în imagine și numele persoanei detectate, cu o probabilitate asociată.

O evaluare a retelelor considerand capabilitatile robotului este in figura 2.5. S-au creat 2 seturi de date proprii, unul cu iluminare artificiala normala si unul fara, in laboratorul in care robotul actioneaza in mod regulat. Fiecare set de date contine 300 de imagini pentru 3 persoane, cate 100 pentru fiecare. Imaginile pentru o persoana sunt pentru diferite distante ale persoanei fata de camera, de la 1 la 10 metri. Pentru reteaua FaceNet recunoasteri valide sunt considerate doar cele care au scorul de incredere peste 0.5.

Întrucât componenta de identificare de persoane poate genera rezultate incorecte în anumite condiții speciale, cum ar fi ocluzii, rotiri ale fetelor sau iluminare necorespunzătoare, aceasta poate fi secondata de un modul de recunoaștere de voce. Ideea utilizării unui modul de recunoaștere de voce este adăugarea unor informații suplimentare pentru persoanele detectate în imagini RGB. Vom investiga această direcție în etapa următoare.

[9] http://wiki.ros.org/hector_slam

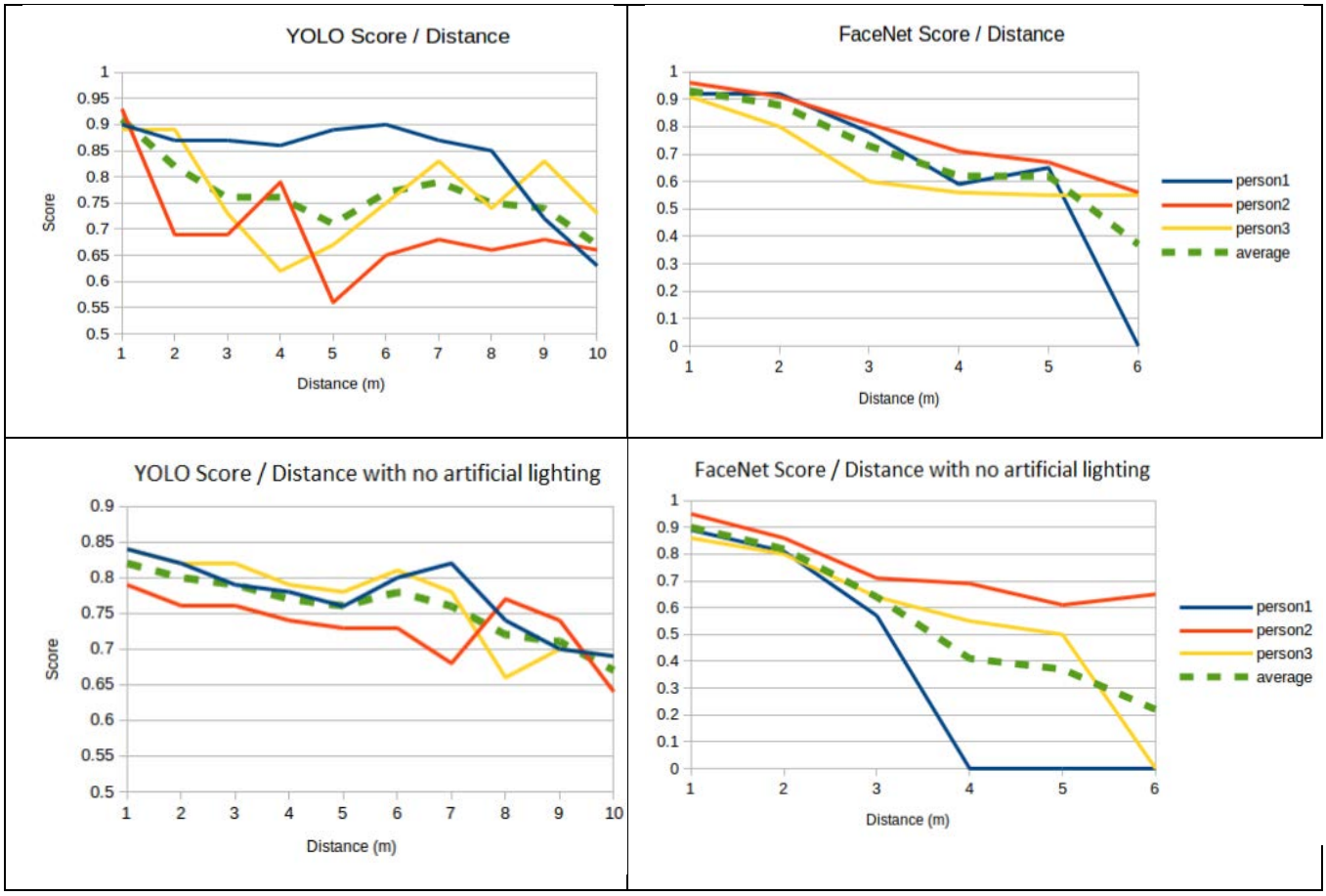


Figura 2.5 Evaluarea modulului de recunoaștere de persoane

2.7 Cercetări preliminare pentru recunoașterea activităților

Pentru recunoașterea activităților, am plecat de la conceptul de reprezentare pe bază de schelet prezentat de Johanson in 1973, care a demonstrat că un număr mic de poziții comune pot reprezenta în mod eficient comportamente umane. Proiecțiile bazate pe schelet 3D oferă performanțe promițătoare în aplicațiile din lumea reală, deoarece reprezentările bazate pe schelet 3D pot să modeleze relația articulațiilor umane și să codifice configurația întregului corp. Am pornit de asemenea de la experimentele noastre recente care utilizează utilizează o cameră Kinect cu un RGB-D sensor capabil să ofere date multimodale și rețele neurale de tip LSTM și convoluție temporală (TCN – Temporal Convolutional Network) cât și de la cercetările prezentate în [Shahroudy et.al. 2016], [Zhu et.al. 2016], [Bai et.al. 2018]. Am implementat o nouă arhitectură de rețea plecând de la [Zhang et.al. 2018] și am obținut rezultate apropiate de state-of-the art. Am explorat 2 variante de arhitectură nouă, una bazată pe intrări multidimesionale, în care toate jointurile 3D sunt transmise într-un unic vector, care sunt apoi transmise către straturile convoluționale 2D. În variant a 2-a, jointurile sunt aranjate într-un layout 2D ca o matrice de 5 x 5 x3, care este apoi transmisă către straturi convoluționale 3D, conform figurii 2.6 (Pentru fiecare strat TCN, numărul de canale produse de convoluție este indicat în figură).

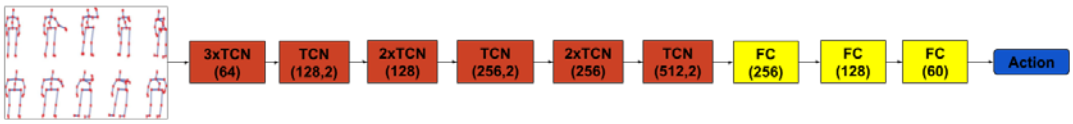


Figura 2.6. TCNs – Arhitectura rețelei pentru model TCN

Pentru antrenare și experimente, am utilizat setul de date NTU RGB + D[10]. Setul de date NTU RGB + D conține în total 60 de clase de acțiuni, care sunt împărțite în trei grupe majore: 40 acțiuni zilnice (băut, mănâncă, citit etc.), 9 acțiuni de sănătate (strănut, împiedicare, cădere etc.) și 11 acțiuni reciproce (punching, kicking, îmbrățișare, etc.). În prezent, acesta este cel mai mare set de date de recunoaștere a acțiunii bazate pe adâncime, furnizând coordonate 3D a 25 articulații colectate de Kinect v2. Acest set de date conține peste 56 000 de secvențe și 4 milioane de cadre, capturate în diferite condiții de fond.

În etapa următoare dorim să colectăm date din laborator cu robotul Pepper, să construim propriul nostrum set de date, și să combinăm recunoșterea activităților pe bază de schelet cu recunoașterea activităților din secvențe video.

2.8 Continuarea cercetărilor

În etapa următoare ne vom orienta spre implementarea unei metode de planificare dinamica cu componente de învățare, a unor module comportamentale flexibile si configurabile, vom încerca creșterea în continuare a performanțelor recunoașterii de persoane, inclusive pe bază de voce, includerea a mai multor scenarii de test, cât și recunoașterea activităților utilizatorului din secvențe video prin metode originale. Vom urmări și integrarea componentelor de prelucrare a limbajului natural dezvoltate în ROBIN-Dialog.

Aceste cercetări și implementări vor fi utilizate pentru implementarea soluției de asistență robotică cognitivă și de urgență pentru utilizatori cu nevoi special, integrarea soluțiilor de asistență robotică cu interfețe în limbaj natural pentru interacțiunea cu roboții și comanda senzorilor ambientali, și implementarea soluției de interacțiune socială.

Raportul in extenso asociat etapei I a proiectului ROBIN-Social se găsește la adresa http://aimas.cs.pub.ro/robin/rezultate/

3. Proiectul component ROBIN-Car

Parteneri ai proiectului ROBIN-Car

INSTITUTUL DE MATEMATICĂ "SIMION STOILOW" AL ACADEMIEI ROMANE (CO)
UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI
INSTITUTUL DE CERCETĂRI PENTRU INTELIGENȚĂ ARTIFICIALĂ „MIHAI DRĂGĂNESCU"
UNIVERSITATEA "DUNAREA DE JOS" DIN GALAȚI

3.1 Prezentare generală

Lumea în care vom vedea vehicule autonome pe străzi nu este departe și multe companii internaționale, atât comerciale cât și high-tech investesc sute de milioane pentru a aduce această tehnologi în viața reală. Există 4 stagii cheie, prevăzute de analiști, ale evoluției tehnologiei vehiculelor autonome: Advanced Assisted Driving Systems (2015-2018), Hand-off self-driving (2019-2021), Automated driving (2022-2025), Fully autonomous cars" (2025-). Deși suntem în etapa AADS și ne pregătim pentru etapa următoare, atât industria cât și comunitatea științifică se confruntă cu provocări majore în realizarea vehiculelor autonome sau semi-autonome. O direcție importanta de cercetare in viitorul apropiat este dezvoltarea de sisteme pilot sau asistent de pilot inteligente robuste, cu funcționare în timp real si costuri de producție, calcul si memorie cât mai scăzute.

[10] https://github.com/shahroudy/NTURGB-D

Scopul proiectului ROBIN-Car constă în dezvoltarea de metode de vedere computaţională care să rezolve o gamă mai largă şi mai sofisticată de sarcini de asistenţă în pilotaj, realizarea unor module inteligente pentru „Hands-off driving” și „Automated driving” și un sistem prototip care va fi testat pe un autovehicul electric semi-autonom pus la dispoziția consorțiului de compania PRIME Motors Industry, pe durata derulării proiectului. Sistemul va fi capabil să observe, recunoasca si monitorizeze scena, drumul, obiectele si persoanele din mediul exterior precum si expresia șoferului, oferindu-i informatiile necesare intr-un mod cat mai non-invaziv (inclusiv interacține vocală prin comenzi simple), capacitate crescuta de pilotaj si de luarea deciziilor.

Obiectivele proiectului ROBIN-Car sunt:

- Realizarea unui modul de înțelegere semantică a obiectelor din mediul înconjurător autovehiculului, pentru detecția și urmărirea obiectelor aflate în mișcare (alte maşini din trafic, pietoni), recunoașterea obiectelor fixe (obstacole, benzi, semne de circulaţie) prin fuziunea datelor de la camere 2D dar și 2D și 3D laser, bazat pe algoritmi noi de computer vision.
- Realizarea unui modul de gestionare eficientă a geometriei scenei 3D, în vederea estimării 3D a zonelor cu gropi, denivelari, etc.
- Realizarea unui modul de recunoaștere a expresiei feței șoferului și a direcției de privire, pentru atenționare și asistare în timpul condusului, in vederea identificării gradului de oboseală sau a direcţiei privirii.
- Realizarea unui modul de atenţionare a conducătorului autovehiculului pe baza unor servicii sensibile la context oferite de subproiectul P3 – ROBIN-Context
- Preluarea de comenzi vocale în limbaj natural și translatarea informațiilor vizuale in limbaj natural (în limba română) pentru realizarea interacțiunii între conducătorul auto și mașină
- Accelerarea algoritmilor dezvoltaţi bazaţi pe reţele convoluţionale, pe sisteme reconfigurabile (FPGA-uri).
- Realizarea unui sistem prototip care să integreze facilitățile descrise anterior, instalarea acestuia pe un vehicul electric semi-autonom și testare intensivă.

Activitățile Etapei I a proiectului au fost următoarele.

Activitatea: ***Act 1.5 - Definirea specificaţiilor funcționale şi arhitecturale pentru modulele ROBIN-Car***, proiectarea modulelor de vedere artificiala oferite de proiect Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: L4 - specificațiile funcționale și arhitecturale si a modulelor de vedere artificiala

Obiectivul a fost integral îndeplinit, vezi secțiunea 3.2 din prezentul raport și raportul in extenso.

Activitatea: ***Act 1.6 - Construcţia seturilor de date necesare antrenării modulelor ce vor fi dezvoltate în sistem*** Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: Set date colectate pentru navigare autonomă

Obiectivul a fost integral îndeplinit, vezi secțiunea 3.4 din prezentul raport și raportul in extenso.

Activitatea: ***Act 1.7 - Proiectarea, implementarea şi testarea locală a funcţionalităţilor modulului de gestionare eficientă a geometriei scenei 3D*** Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: L5 - modulul de gestionare eficientă a geometriei scenei 3D 1 modul software gestionare geometrie scena 3D funcțional

Obiectivul a fost integral îndeplinit, vezi secțiunea 3.3 din prezentul raport și raportul in extenso.

Activitatea: ***Act 1.8 - Diseminare*** Categorie activitate: D1 - Activități suport - Diseminarea pe scară largă prin comunicarea și publicarea națională sau internațională a rezultatelor Indicatori de realizare: 1 lucrare științifică

Obiectivul a fost integral îndeplinit.

A fost publicată lucrarea *Learning a Robust Society of Tracking Parts using Co-occurrence Constraints*, VOT2018 Challenge, European Conference on Computer Vision (ECCV) 2018, 8-14 sept - conferință de categoria A

A trimisă spre evaluare lucrarea *Efficient two-stage approach for speed bumper segmentation and classification in fish-eye images*, International Conference on Computer Vision and Pattern Recognition (CVPR), 16-21 iunie 2019 – conferință categoria A+

A fost acceptată o lucrare, cu modificări, *Unsupervised learning for foreground object segmentation*, în revista International Journal of Computer Vision (IJCV). (IF 11.56), Status: Accepted with Major Revisions

A fost publicată lucrarea *Intelligent Autonomous Driving*, 22nd International Conference on Sytem Theory, Control and Computing, Octombrie 10-12 2018, Sinaia, România, 978-1-5386-4444-7/18, 2018 IEEE

3.2 Cerințe funcționale și arhitecturale ale modulelor ROBIN-Car

Există două abordări majore în dezvoltarea unui pilot automat. Prima abordare este aceea bazată exclusiv pe imagini din fluxuri video, în care scena înconjurătoare este înțeleasă, interpretată semantic și apoi pilotul automat poate lua decizii în consecință. Aceasta este o direcție nouă de dezvoltare, care nu este utilizată în sistemele prototip de conducere autonomă din considerente de securitate, abordarea nefiind pe departe matură ci numai la nivel de cercetare. Cea de a 2-a abordare, utilizată în sistemele de conducere autonomă prototip existente în cadrul marilor companii auto folosește, pe lângă metode de vedere computațională, și date provenite de la senzori, cum ar fi GPS, date inerțiale (accelerometru, magnetometru, giroscop), LiDAR. În cadrul acestei etape a proiectului am investigat ambele abordări.

În cadrul primei abordări, s-a dezvoltat o primă variantă de pilot automat vizual. Pilot automat vizual și asistent în conducere este unul din cele mai ambițioase obiective ale proiectului este acela de a dezvolta un sistem complex capabil de a învăța să conducă singur. Un astfel de sistem include module și algoritmi care pot să ofere indicații de conducere deduse doar din informații vizuale, să realizeze geolocalizarea chiar și în absența GPS-ului, să aibă o înțelegere completă a semanticii și a structurii 3D a scenei și în cele din urmă să învețe cum să piloteze un vehicul fără intervenție umană, într-un mod corespunzător, inteligent și sigur.

Abordarea urmată este bazată pe un ansamblu de rețele adânci: sistemul automat propus folosește mai multe rețele neurale adânci pentru a prezice comenzile de schimbare a direcției pentru navigarea de la sursă la destinație pe o hartă (figura 3.1).

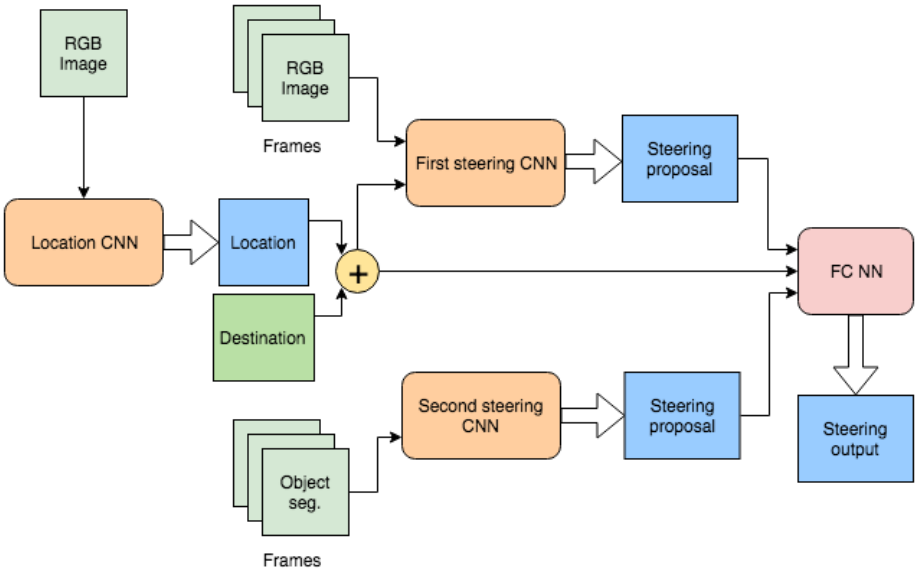


Figura 3.1: Arhitectura pentru schimbarea direcției pentru primul nostru pilot cu recunoaștere vizuală folosind un ansambu cu rețele neurale adânci

Rețeaua convoluțională de localizare învață să prezică localizarea într-o manieră supervizată dintr-o imagine RGB. Localizarea este descrisă prin valorile etichetelor: nod anterior, nod următor, nod de după cel următor și distanța până la următorul nod.

Prima rețea convoluțională de conducere preia la intrare frama curentă concatenată cu alte două frame din trecut, valorile de la ieșirea rețelei de localizare și destinația. Aceasta prezice apoi valori discrete pentru accelerație și unghi de virare. Intrarea pentru a doua rețea convoluțională de conducere este constituită din frame concatenate cu segmentările obiectelor relevante din trafic, cum ar fi drumul, mașinile și pietonii. Spre deosebire de prima rețea convoluțională de conducere, cea de-a doua nu cunoaște localizarea și destinația. Scopul ei este de a prezice comenzi de volan pentru a menține vehiculul pe drum și pentru a evita coliziunile. Segmentarea framelor este obținută cu un model bazat pe arhitectura U-Net pe un set de date de conducere care va furnzia adnotarea pentru segmentări. Ultima rețea, complet conectată, face un balans între predicțiile celor două rețele convoluționale în funcție de localizare și destinație, având ca ieșire decizia pentru comanda de direcție.

Rezultate preliminare: am realizat un prim set de experimente antrenând modelele și apoi testându-le cu date video, pentru care am obținut rezultate încurajatoare. Mai jos sunt prezentate cele mai reprezentative valori ale metricilor de evaluare pentru experimentele inițiale.

Rezultate inițiale pe setul de date

	accelerație			schimbarea direcției		
acuratețe	74%			94%		
precizie	63%	83%	66%	84%	96%	90%
recall	71%	77%	70%	71%	98%	79%
	frânare	deloc	accelerare	stânga	deloc	dreapta

După cum se poate observa acuratețea pentru schimbarea direcției este cu 20% mai mare decât cea pentru controlul accelerației. Acest lucru se datorează faptului că au fost adnotate diferențe de viteză foarte subtile (0.6 km/h într-o secundă) iar majoritatea erorilor sunt făcute în aceste regiuni limită. Per total, considerpm rezultatele ca fiind bune având în vedere că este primul model pe care l-am încercat pentru această problemă. De asemenea, sunt multiple situații în setul de test unde etichetele cu valorile trecute ca fiind adevărate sunt afectate de zgomot GPS (pe care nu l-am putut elimina în pasul de filtrare) în timp ce valorile prezise de rețea sunt defapt cele corecte.

Concluzii: în timp ce rezultatele raportate mai sus sunt încurajatoare deși doar preliminare, ne așteptăm ca performanțele să se îmbunătățească semnificativ într-un termen scurt, odată ce vom combina mai multe module de procesare împreună. Vom obține astfel o interacțiune coerentă și sinergistă între diverse task-uri de recunoaștere vizuală cum ar fi segmentare semantică, estimare 3D a structurii scenei, detecție de diverse tipuri de obiecte cunoscute și descoperirea de regiuni similare obiectelor dar necunoscute. În afară de scală, varietatea și calitatea datelor de antrenare colectate, considerăm că unul din cei mai importanți factori care vor permite conducerea automată la un cost redus este abilitate de a procesa informațiile în domeniul spațiu-timp, renunțând la paradigma procesării imagine cu imagine.

Pentru realizarea unui pilot automat vizual avem astfel nevoie de următoarele componente.

Înțelegere semantică completă a scenei: studiem și dezvoltăm metode pentru segmentarea semantică completă a scenei în diverse clase vizuale care sunt relevante pentru mașini care se conduc singure. Aceasta include segmentarea drumului, a marginilor drumului și a benzilor, a diverselor obiecte prezente pe carosabil sau pe trotuar, cum ar fi oameni, animale și vehicule, dar și a diferite structuri fixe precum clădiri, garduri sau copaci. Acest task este foarte complex întrucât necesită definirea corectă a universului contextual bazat pe aceste clase și pe posibilele interacțiuni între ele. Modelele pe care le dezvoltăm sunt bazate pe rețele neurale adânci de segmentare semantică care sunt de

actualitate în cercetarea din domeniu, dar și pe idei inovatoare în care ne dorim să realizăm reprezentarea scenei cu rețele adânci bazate pe grafuri. Cercetarea noastră trece de la procesarea standard a unei singure imagini la interpretarea unei scene atât spațial cât și temporal folosind modele de rețele adânci relaționale, eficiente și inovatoare, care sunt special concepute pentru procesarea video. Ca finalitate, ne dorim soluții de mare precizie și cu o procesare rapidă.

Învățare nesupervizată a structurii unei scene 3D: înțelegerea automată a structurii tridimensionale a lumii este vitală pentru asigurarea siguranței conducerii automate. Problema estimării 3D poate pleca de la estimarea precisă a localizării fiecărui punct din scenă, până la estimarea calitativă de nivel înalt a suprafețelor, adâncimii și a altor propietăți 3D ale obiectelor și structurilor din scenă. În proiectul ROBIN-Car dezvoltăm metode bazate pe rețele convoluționale adânci și rețele neurale recurente de grafuri pentru a învăța într-o manieră nesupervizată (sau cu foarte puține date de antrenare supervizată) structurile 3D ale scenei din video sau alte fluxuri de date 4D (spațiul 3D + timp).

Set de date de mari dimensiuni pentru pilotare automată bazată pe context: există câteva seturi de date disponibile public care pot ajuta în dezvoltarea și antrenarea modelelor de bază pentru conducere automată, însă nici unul dintre acestea nu conține date înregistrate în contextul coducerii prin București sau prin România. Devine din ce în ce mai evident că mare parte din succesul metodelor actuale bazate pe învățare cu rețele adânci este atribuit setului de date de antrenare depinzând mai ales de timpul, locul, structura scenei, a activităților și actorilor implicați în colectarea acestuia. Un model care este antrenat și testat pe un anumit set de date cel mai probabil va da greș sau va avea performanțe scăzute pe un altul. Abia recent cercetătorii au început să ia în considerare efectele bias-ului din seturile de date folosite în învățarea automată și vederea computațională iar concluziile sunt clare: nu ne putem aștepta la capacitatea de a generaliza dincolo de contextul și domeniul setului de date de antrenare întrucât este de așteptat ca datele de test să facă parte din aceeași distrubiție ca și cele de antrenare.

Din motivele menționate anterior, deducem necesitatea colectării propriul set de date de mari dimensiuni dacă dorim să creăm un pilot automat. În plus, un astfel de set de date va ajuta mult în crearea și perfecționarea de metode și algoritmi proprii, special dezvoltați pentru problemele pe care dorim să le rezolvăm în cadrul proiectului, dar și pentru a adresa limitările actuale ale metodelor din literatura de specialitate sau aplicate în industrie.

3.3 Modulele de gestionare eficientă a geometriei scenei 3D

3.3.1 Recunoașterea și segmentarea semantică a benzilor de circulație

Detecția din imagine a zonei prin care se poate conduce este o problemă interesantă pentru conducerea automată, dar pentru a putea fi aplicată în lumea reală este nevoie să se estimeze și cât de „liberă” este această zonă. Pentru a rezolva această parte de problemă am introdus un nou set de date despre care vom discuta mai pe larg în părțile ulterioare ale acestui document. Pe scurt, am înregistrat un set de date în timp ce am condus prin diverse zone ale Bucureștiului, asigurând astfel suficientă diversitate și variație a datelor.

Pentru problema specifică a recunoașterii și segmentării benzilor am clasificat în funcție de 4 clase: „liberă”, „ocupată”, „bară-la-bară” și „bandă din intersecție”. Clasa „liberă” indică faptul că zona prin care se poate conduce, de pe banda respectivă, nu conține nicio mașină sau mașina din față este foarte departe. Categoria „ocupată” semnifică existența unei mașini pe banda respectivă, dar care nu este însă foarte aproape. Clasa „bară-la-bară” este folosită când în zona prin care se poate conduce există o mașină la o distanță foarte mică. A fost utilizată și clasa „bandă din intersecție” întrucât este dificil de clasificat zonele prin care se poate conduce din intersecții folosindu-se doar primele 3 clase.

În figura 3.2 sunt prezentate câteva exemple de benzi segmentate în imagine. Pentru această problemă am început de la un set de 3 video de lungime relativ mare. Două au fost utilizate pentru antrenare și unul pentru validare. Setul de date conține 3286 imagini etichetate pentru antrenare și 1022 imagini etichetate pentru validare.

Figura 3.2 Exemple de zone care permit conducerea. Regiunile roz reprezintă „benzi-în-intersecție”, cele roșii sunt benzi „bară-la-bară”, regiunile galbene sunt benzi „ocupate” iar cele verzi sunt benzi „libere”

Pentru problema noastră am conceput un nou model plecând de la paradigma U-net care este restrânsă ca dimensiuni și eficientă, respectiv arhitectura UniNet pe care în [Marcu et.al. 2018] am aplicat-o cu succes pe mai multe probleme, cum ar fi segmentarea și localizarea în absența GPS-ului.

3.3.2 Urmărirea vizuală robustă a obiectelor pe termen-lung

Urmărirea obiectelor este o problemă de bază în domeniul vederii computaționale care este în continuă cercetare în ultimele decenii. În contextul mașinilor care se conduc singure ea reprezintă o capabilitate esențială căci furnizează informații despre cum se mișcă și se comportă obiectele din jurul vehiculului, făcând posibilă urmărirea acestor obiecte în timp și spațiu. Deși problema este abordată de ceva vreme, ea este încă departe de a fi complet rezolvată și înțeleasă.

Adresăm această problemă propunând o arhitectură de rețea neurală adâncă care funcționează ca o societate de părți [Burceanu&Leordeanu, 2018]. În Figura 3.3 prezentăm în perspectivă propunerea sistemului cu două căi care învață o urmărire robustă folosind constrângeri de co-apariție.

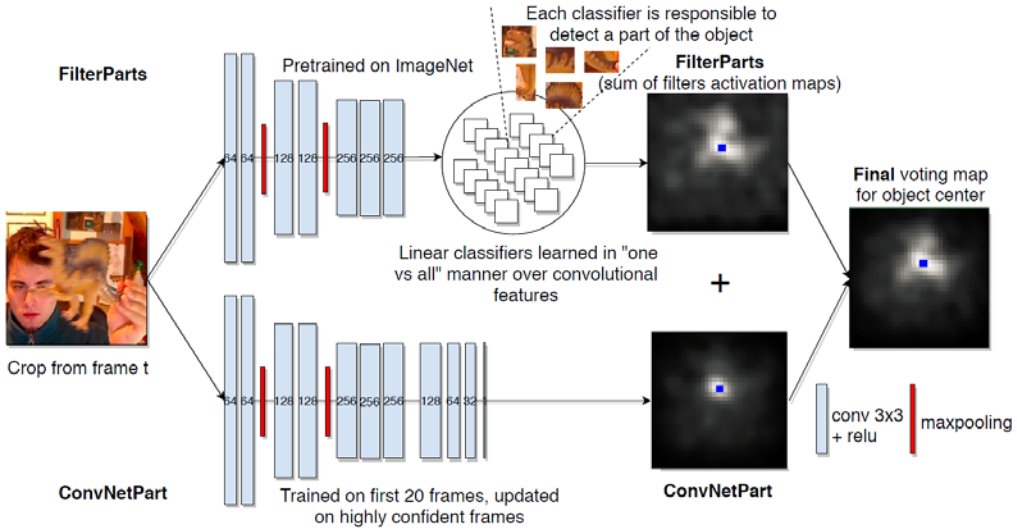


Figura 3.3 Propunerea de urmărire a obiectelor funcționează ca o societate de părți. Combină voturile pentru hărțile centrale din toate părțile celor două căi principale, FilterParts și ConvNetPart. Cele două căi sunt antrenate diferit. Clasificatorii FilterParts sunt antrenați individual dar se adaptează ca grup. Calea ConvNetPart este antrenată de la un capăt la altul folosind backpropagation peste ieșirile nesupervizate ale tracker-ului de la frame anterioare cu confidență mare.

3.3.3 Detecția automată a limitatoarelor de viteză

Înțelegerea automată a scenelor bazat pe informațiile colectate de diverși senzori precum radar, LIDAR și camere video reprezintă un element cheie pentru sistemele inteligentă de asistență în conducere. Deși există cercetări și implementări curente formidabile pentru sistemele de detecție de obiecte și segmentare semantică din imagini în contextul mașinilor autonome, detecția automată a limitatoarelor de viteză nu a fost încă abordată în literatură, din ce cunoaștem noi, deși prezintă un real potențial de a îmbunătăți calitatea șofatului și eficiența la frânare.

Problema detecției de limitatoare de viteză este o problemă de recunoaștere este una foarte specifică și care prezintă dificultăți chiar și pentu un om. Detecția lor pe drumuri și capacitatea de a discerne care dintre ele sunt reale și care sunt false reprezintă o sarcină grea dar importantă pentru a atinge niveluri înalte de calitate și siguranță pentru conducerea inteligentă.

Am adresat simultan două aspecte legate de recunoașterea limitatoarelor de viteză și anume segmentarea lor în imagini și clasificarea lor în reale sau false [Ryu, 2019],. Din cercetarea efectuată a reieșit faptul că a doua sarcină, cea de clasificare, este foarte dificil de realizat – chiar și pentru oameni – și devine o problemă și mai grea când se dorește un cost computațional scăzut. În soluția prezentată am introdus un model de rețea adâncă bazat pe o abordare în două etape. În prima etapă, o rețea convoluțională cu rol de mecanism specializat de atenție vizuală realizează segmentarea limitatorului de viteză dându-se o imagine. În cea de-a doua etapă o altă rețea clasifică ieșirea primei ca fiind reală sau falsă bazat pe intrarea RGB originală cât și pe ieșirile etapei precedente. Cele două etape sunt întâi antrenate separat apoi împreună, de la un capăt la altul, pentru reglaje fine, pe un nou set de date introdus în această cercetare. În figura 3.4 este descris sistemul propus. Rezultate experimentale, prezentate în [Ryu, 2019], arată că metoda noastră surclasează chiar și performanța umană cu o diferență semnificativă.

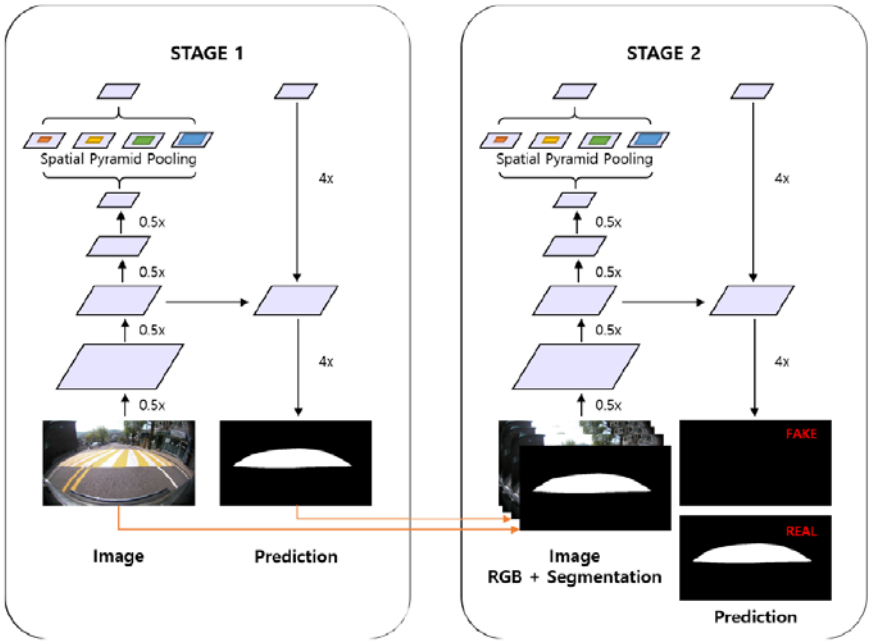


Figura 3.4 Privire de ansamblu asupra sistemului de recunoaștere a limitatoarelor de viteză, formulată ca o problemă de segmentare semantică pe două niveluri de procesare. În prima etapă detectăm regiunile din imagine care conțin limitatoare de viteză, indiferent de tipul lor. Cel de-al doilea nivel discerne între limitatoarele false și cele adevărate. Este de notat faptul că sistemul poate fi adaptat pentru aproape orice altă sarcină de segmentare fină și clasificare în contexul conducerii automate.

3.3.4 Detecția obiectelor din prim plan și a regiunilor relevante ale atenției vizuale

În contextul conducerii automate, învățarea nesupervizată poate conferi autovehiculelor inteligente capacitatea de a descopri și învăța despre obiecte necunoscute lor care pot apărea neașteptat în scenă. Cum lumea se află într-o continuă schimbare, capacitatea de a învăța despre noi obiecte în mod

nesupervizat este crucială pentru a putea acționa în siguranță și inteligent și pentru adaptarea la medii dinamice și cu diversitate mare. Am adresat problema învățării nesupervizate în contextul detectări principalelor obiecte din prim plan în imagini [Croitoru et.al., 2019]. Obiectele din prim plan sunt cele de interes și cele pe care atenția noastră vizuală se concentrează în primul rând când este luată în calcul zona acoperită de câmpul de vedere.

Abordarea noastră este să antrenăm un rețea adâncă student care să prezică ieșirea unei căi profesor care detectează nesupervizat obiecte din date video sau din colecții mari de imagini. O perspectivă asupra metodei este prezentată în figura 3.5.

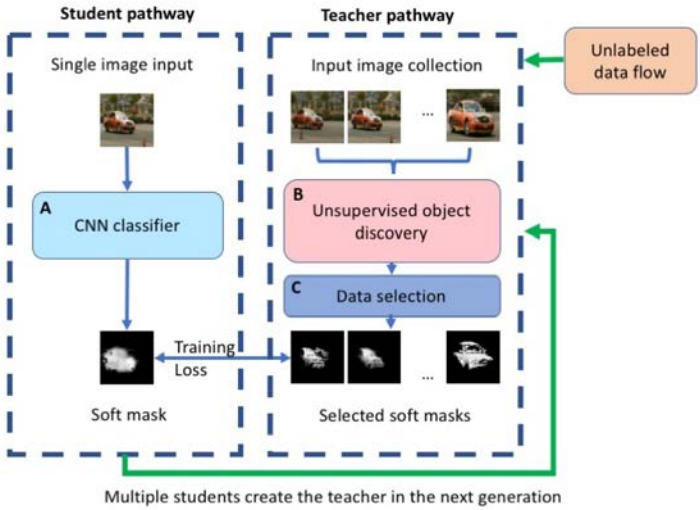


Figura 3.5 Sistemul dual, student-profesor, propusă pentru învățarea nesupervizată în vederea detecției de obiecte în imagini. Este compus din 2 căi: pe ramura profesor un modul care descoperă obiecte în date video și colecții mari de imagini (modulul B) caută obiectele din prim plan.

3.4 Construcţia seturilor de date

3.4.1 Colectarea datelor pentru pilot automat vizual

Pentru abordarea bazată pe pilot automat vizual, am utilizat un smartphone cu sistem de operare Android, care va fi folosit atât pentru colectarea de date de antrenare cât și la testare. Am dezvoltat propria aplicație Android care înregistrează atât video cât și stream-ul GPS într-o manieră sincronizată. Pentru a realiza acest lucru am folosit direct API-ul camerei hardware a smartphone-ului și API-ul de localizare în timp real furnizat de Google prin care am colectat de asemenea și valorile de viteză și orientare. Am utilizat fluxul de date GPS colectat la frecvență mare simultan cu fluxul video. Din diferențele între poziții succesive se pot calcula valorile accelerației și schimbările de unghi, acestea fiind folosite apoi pentru adnotarea automată a setului de date de antrenare.

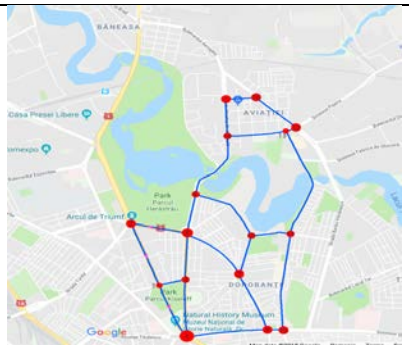


Figura 3.6 Harta selectată pentru crearea setului de date

Harta de navigare: harta selectată pentru colectarea primei părți a setului de date pe care vom testa primul prototip acoperă zona din București descrisă în figura 3.6. În figură se poate observa graful direcționat în care nodurile reprezintă intersecții iar muchiile sunt drumuri între aceste intersecții. Graful rezultate are 16 noduri și 41 de muchii, acoperind în total o lungime de 30 km.

3.4.2 Colectarea datelor din campusul UPB

Așa cum am prezentat în secțiunea 3.2, am investigat și direcția în care conducerea autonomă este realizată atât pe baza informațiilor vizuale cât și a celor provenite de la sensori. De asemenea, conform

desccrierii propunerii de proiect, dorim să realizăm o aplicație de conducere autonomă prin campusul UPB utilzând un autoturism electric Logan pus la dispoziția echipei proiectului de o companie privată (Prime Motors).

Pentru a asigura o integrare mai strânsă între autovehicul și pilotul automat cu intrările și ieșirile sale, și mai ales pentru a pune accentul și mai mult pe siguranță și precizie în control, este necesară colectarea și analiza unui set de date care să cuprindă informații colectate direct de la mașină (accelerație, unghi de volan, frână, etc.) și informații video din mai multe surse calibrate, toate acestea calibrate și sincronizate cu informații precise de localizare. Există deja anumite resurse disponibile, sub formă de seturi de date cu acces deschis, colectate de pe automobile pe care au fost montați diverși senzori. De exemplu, cercetătorii de la Berkeley au dezvoltat o aplicație pe mobil prin care se pot colecta date video, cât și GPS, IMU (accelerometru) și magnetometru (deviația fata de nordul magnetic) [Xu et.al. 2017]. Astfel, în setul de date fiecare imagine este corelată cu unghiul de volan și viteza curentă a mașinii.

Cu toate acestea, pentru a realiza adaptarea algoritmului la condiții similare cu cele în care va fi testat este necesară colectarea unui set de date local, în campusul UPB. Pentru colectare, echipa de proiect a decis implementarea următoarelor tipuri de aplicații:

- aplicație pe smartphone, care permite transmiterea în timp real către un server a datelor de GPS, accelerometru, giroscop (orientare în spațiu 3D) și magnetometru[11]
- un script care colectează simultan cadre RGB de la 3 camere web HD, montate pe vehicul12
- un script care colectează în timp real date de pe magistrala CAN a mașinii, înregistrând îndeosebi viteza curentă a mașinii, precum și unghiul de volan[13]

Camerele au fost montate pe plafonul mașinii, una situată pe axa centrală, iar două puse pe laterale, în colțurile plafonului. Scopul acestei amplasări a fost acela de a putea cuprinde cat mai multa informație în imagini, atunci când mașina se apropie și se află într-o intersecție. Se dorește ca măcar într-una din camere să poată fi cuprins un reper (e.g. marginea bordurii din apropiere în cazul unui viraj la dreapta, bordura de pe sensul opus).

Repere precum bordura sau marcajele de pe șosea sunt necesare, întrucât prin teste proprii s-a observat faptul că metodele de vedere computațională pentru segmentare și identificare a benzilor de circulație se folosesc de astfel de repere în predicția lor, astfel că asigurarea unui grad mai mare de vizibilitate este obligatoriu. În plus, o rază vizuală marită asigură o mai bună acoperire a obiectelor din jurul mașinii (e.g. pietoni la o trecere, mașini parcate) astfel încât detecția acestor tipuri de obstacole să fie înlesnită.

În vederea prelucrării ulterioare a datelor se realizează proceduri de calibrare intrinsecă și extrinsecă a camerelor video. Pentru calibrarea intrinsecă a fost utilizată o procedură clasică bazată pe prelevarea de imagini folosind camera a unei planșe dreptunghiulare cu un caroiaj alb-negru de dimensiune 8 x 6 pătrate, fiecare cu latura de 10 cm. Imaginile surprind planșa așezată în diverse poziții și la diferite distanțe de cameră. Algoritmul de calibrare folosește aceste imagini, precum și informațiile legate de dimensiunea caroiajului pentru a calcula coeficienții de distorsiune și matricea intrinsecă a camerei.

Odată calibrate camerele, s-au efectuat 3 sesiuni de colectare a datelor. Fiecare sesiune a avut 40 de minute durata (limitare data de bateria laptopului pe care se face colectarea), pentru un total de 2 ore de condus prin campusul UPB.

Din cele 3 sesiuni, doua au fost efectuate pe timp de zi, iar una pe timp de seara. Per sesiune de înregistrare, datele video depășesc 100 GB.

11 https://github.com/nemodrive/PhoneSensorsCollectionApp

12 https://github.com/nemodrive/car_data_collection

13 https://github.com/nemodrive/can_decode

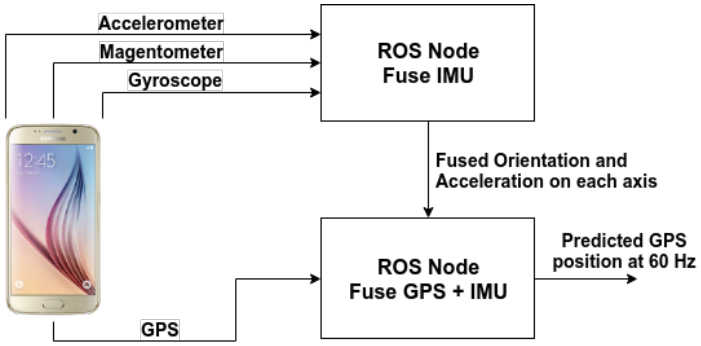


Figura 3.7: Pipeline de fuziune a datelor pentru imbunatatirea localizarii, combinand date GPS și inerțiale

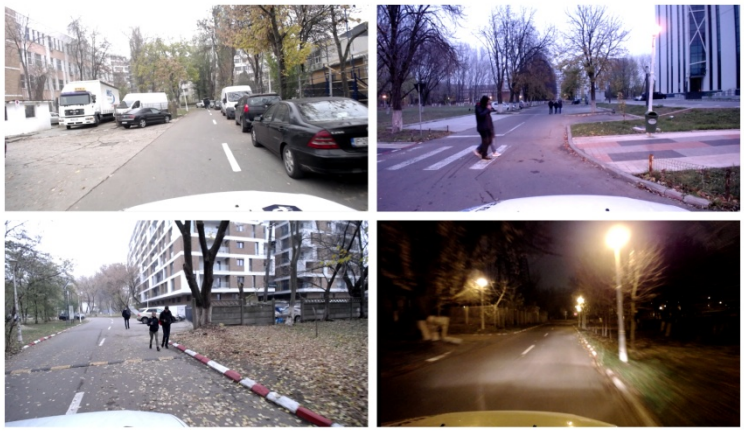


Figura 3.8 Exemple de cadre colectate în datele din UPB. A se remarca condițiile diferite de lumina și tipurile de participanți la trafic des intâlnite: pietoni și mașini parcate

Figura 3.8 arată exemple de cadre dintre cele colectate în campusul UPB. Se poate observa că setul de date acoperă condiții diferite de lumină (de la zi, la înserare și apus de soare). În plus, cadrele surpind tipul de obstacole tipic întâlnit în campus și care influențează în mod puternic politica de conducere și sistemele de detecție și segmentare descrise anterior: pietoni care merg pe marginea carosabilului, mașini parcate pe întreg sensul de mers și care impun depășirea acestora, limitatoare de viteză.

3.5 Continuarea cercetărilor

În perioada imediat următoare vom proceda spre testarea tuturor soluțiilor propuse anterior pe setul de date colectat până acum din campusul UPB. Se vor testa metodele de segmentare ale imaginilor, pentru a determina acuratețea și recall-ul în identificarea vehiculelor, pietonilor, drumului, gardurilor, limitatoarelor de viteză și a altor obiecte de interes.

O altă sarcină care se va afla în testare folosind datele colectate este cea de localizare. Poziționarea corectă a autovehiculului este unul din obiectivele de bază în conducerea automată, în proiectul Robin acest task fiind esențial și necesar pentru a putea fi îndeplinite alte sarcini mai complexe. De asemenea, pentru asigurarea siguranței în operare, task-ul de urmărire al obiectelor din scena 3D în care se află autovehiculul este de asemenea tratat cu mare interes în cadrul proiectelor. Datele colectate în campus conțin suficiente instanțe de pietoni și vehicule care participă la trafic. Prin urmare, vom testa soluțiile propuse pe aceste date.

În paralel cu realizarea testelor, procesul de colectare de date va fi continuat. Mai mult decât atât, odată realizat deployment-ul final, noi date vor fi culese considerând faptul că vor fi instalați noi senzori. Toate aceste teste sunt necesare pentru a confirma ipoteza adaptării soluțiilor din domeniul conducerii automate la condițiile de implementare. Mai precis, vom observa dacă acestea funcționează în aceeași parametri și cu aceleași rezultate și în condițiile date de campusul UPB.

Raportul in extenso asociat etapei I a proiectului ROBIN-Car se găsește la adresa

http://aimas.cs.pub.ro/robin/rezultate/

4. Proiectul component ROBIN-Context

Parteneri ai proiectului ROBIN-Context

UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI (CO)
UNIVERSITATEA TEHNICĂ DIN CLUJ – NAPOCA

4.1 Prezentare generală

Serviciile dependente de context și calculul contextual (context-aware computing) oferă suport atât aplicațiilor cât și utilizatorilor oferind informații oportune, la momentul potrivit și în forma potrivită. Recent, calculul contextual și serviciile dependente de context au început să își facă locul atât în aplicațiile robotice de asistență personală, cât și în aplicațiile de Advanced Driving Assistance Systems (ADAS).

Un sistem de Context-Aware Driving Assistance leagă șoferii de mediul în care aceștia conduc, atât din perspectiva relației om – mașina proprie, cât și a situației din trafic. Solutiile ADAS existente se concentrează fiecare asupra unui singur aspect al condusului.

Scopul unui sistem ADAS sensibil la context este să ofere o înțelegere de ansamblu (la nivel semantic) cât mai bună a condițiilor curente de trafic (e.g. un șofer din spate vrea sa depășească în condiții de ploaie ușoară și a unui drum în curbă largă). Un sistem robotic de asistență personală dependent de context utilizează și raționează pe baza diferiților senzori montați atât pe robot cât și în incinte pentru a oferi servicii robotice personalizate, performante și cu robustețe mai mare decât ar putea fi oferite de robotul stand-alone. În scenarii de asistență robotică pentru persoane cu nevoi speciale precum și în cele de servicii de asistență în spații publice, roboții utilizați trebuie să fie la curent mereu cu situația în care se află ei înșiși, mediul în care operează și, mai ales, utilizatorii pe care îi deservesc.

În serviciile dependente de context, o problemă esențială este aceea de a determina ce fel de informație devine context pentru o aplicație și în ce mod ajunge ea să fie folosită în combinație cu altă informație.

Scopul proiectului ROBIN-Context este crearea unei platforme suport pentru definirea/reprezentarea semantică și gestiunea facilă și eficientă a datelor ce devin context în scenarii de asistență robotică personalizată și sisteme ADAS. Platforma va defini un flux bine stabilit al datelor de context (de la achiziția lor, la diseminare/consum) și va oferi biblioteci suport pentru inferarea situațiilor de nivel semantic înalt prin combinarea de tehnici bazate pe cunoștințe și tehnici bazate pe date.

Obiectivele proiectului ROBIN-Context sunt:

- Definirea unui format de reprezentare a datelor care să fie expresiv și sa poată fi utilizat cu ușurință în inferențe de tip semantic, în sisteme de procesare de evenimente, precum și în modele de inferență pe bază de machine learning.
- Proiectarea și realizarea platformei suport pentru implementarea fluxului de achiziție – inferență – diseminare necesar în procesarea și gestiunea datelor de context pentru aplicații centralizate (de exemplu ADAS) și servicii web
- Dezvoltarea unei componente de procesare semantică a datelor pentru a defini constrângerile de detecție a unor situații
- Dezvoltarea unei componente pentru efectuarea inferențelor sub formă de procesare rapidă a evenimentelor
- Dezvoltarea unei componente de inferență a contextului folosind biblioteci de algoritmi pe bază de modele grafice probabilistice
- Proiectarea și dezvoltarea unei componente ce asigură capabilitatea de explicare a procesului de inferență contextuală.
- Proiectarea, realizarea și testarea de instrumente și servicii destinate agenților economici

Activitățile Etapei I a proiectului au fost următoarele.

Activitatea: ***Act 1.9 - Analiza modelelor de inferență probabilistică utilizate pentru detecția contextului*** Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: L6 - modele probabilistice de inferență a contextului

Obiectivul a fost integral îndeplinit, vezi secțiunea 4.4 din prezentul raport și raportul in extenso.

Activitatea: ***Act 1.10 - Definirea specificațiilor funcționale și arhitecturale pentru componentele din ROBIN-Context*** Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: L7- specificații funcționale și arhitecturale

Obiectivul a fost integral îndeplinit, vezi secțiunile 4.2 și 4.3 din prezentul raport și raportul in extenso.

Activitatea: ***Act 1.11 - Diseminare*** Categorie activitate: D1 - Activități suport - Diseminarea pe scară largă prin comunicarea și publicarea națională sau internațională a rezultatelor Indicatori de realizare: 1 lucare științifică

Obiectivul a fost îndeplinit.

Suntem în curs de a definitiva lucrarea *Context reasoning for detecting daily activities* pentru numărul special al revistei SENSORS (IF 2,475), respectiv *A special issue of Sensors (ISSN 1424-8220). This special issue belongs to the section "Intelligent Sensors". Deadline for manuscript submissions: 31 January 2019.* Am primit o invitație de a trimite o lucrare (care va fi recenzată evident) pentru acest număr special.

4.2 Cerințe funcționale ale platformei ROBIN-Context

Platforma ROBIN-Context trebuie sa dezvolte un sistem de suport de procesare a informatiei contextuale pentru doua tipuri de aplicatii: ADAS si robotica asistiva. La nivel funcțional, cerințele specifice celor doua domenii au puncte de intersecție, dar diferă semnificativ in modul de deployment (implementare). Ca atare, platforma ROBIN Context trebuie sa fie capabila sa răspundă următoarelor categorii de provocări.

Flexibilitate a reprezentării. Tipul de informație care trebuie modelat este divers. Mai mult decât atât, in vederea tratării corespunzătoare a evenimentelor, sunt necesare meta-informații (adnotări) care sa caracterizeze o anumita afirmație contextuala (e.g. utilizator prezent in bucătărie) din punct de vedere al certitudinii (i.e. valoare numerica in intervalul 0..1), al validității temporale (e.g. afirmație valida in urmele 12 minute), proveniența (e.g. pe baza cărui/căror senzori a fost dedusa informația).

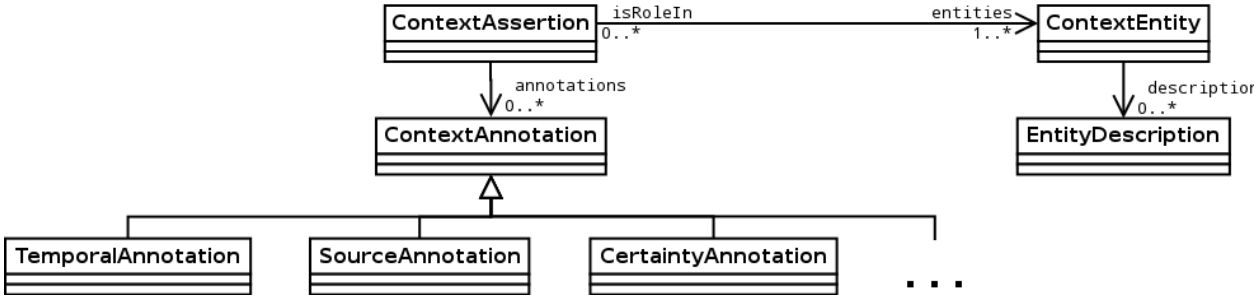


Figura 4.1. Meta-modelul de reprezentare folosit in platforma CONSERT si utilizat pentru ROBIN-Context

Figura 4.1 prezintă meta-modelul [Sorici et al., 2015a] din platforma CONSERT, propusa spre folosire in ROBIN-Context. Proprietati statice sau foarte lent schimbătoare, existente intre entitati (e.g. o camera se afla langa alta camera, un utilizator are o anumita vârsta) sunt reprezentate ca EntityDescriptions. Aserțiunile contextuale sunt in plus caracterizate de adnotări (ContextAnnotation) care surprind tipul de meta-informații discutate anterior. O astfel de reprezentare permite un grad mare

de flexibilitate pentru ca dezvoltatorii sunt liberi sa-si modeleze informația sub forma necesara in aplicație (îndeosebi cu privire la relațiile intre oricâte entitati).

Varietate a metodelor de inferența. In funcție de tipul de senzori folosiți pentru a colecta informații despre situațiile întâlnite atât in contextul șofatului, cat si cel al roboticii asistive, modul in care se infera informația de nivel înalt (e.g. activitatea desfășurata de utilizator pe baza activării unor senzori de mișcare si de deschidere/închidere a ușilor). Unele tipuri de informație pot fi deduse pe baza cunoștințelor de bun simt, sau de domeniu, date de dezvoltatori (e.g. un șofer poate fi determinat ca fiind gata de plecare, daca toate ușile sunt închise si senzorul de presiune de pe scaun este activ). Altele (e.g. a face curat in bucătărie) necesita însa o observare mai îndelungata a modului in care o persoana isi curata bucătăria (e.g. va deschide o ușa către debaraua unde tine o găleata cu un mop, va da drumul la apa intr-un timp relativ apropiat pentru a umple găleata, va intra in bucătărie si va avea un mod particular de a activa senzorii de mișcare - in funcție de modul in care persoana este obișnuita sa isi spele bucătăria). O astfel de conjuncție de condiții este greu de exprimat de la caz la caz, intr-un format bazat pe procesare de evenimente, fie ele si relaționate temporal. Mai eficace este un proces prin care secvența de activări de senzori ulterioara deschiderii debaralei si a robinetului de apa (ca evenimente declanșatoare a activității de curatare) sunt invatate printr-un proces bazat pe date (i.e. de invatare automata - eng. machine learning).

Ca atare, o provocare adusa platformei ROBIN Context este aceea de a găsi un flux de execuție care sa poată îmbina cele doua moduri de inferența, sub influenta unor configurări care sa dicteze cum trebuie dedus fiecare tip de informație.

Adapatare a fluxului informațional la tipul de aplicație. Modul in care sunt consumate informațiile contextuale diferă mult intre cele doua scenarii de aplicabilitate ale platformei ROBIN Context în cadrul proiectului ROBIN. In cazul ADAS, consumul datelor se face de către un singur client (șoferul sau sistemul ADAS in sine). In schimb, intr-o aplicație de robotica asistiva, robotul are mult mai multe surse de date disponibile (inclusiv sisteme cloud externe - e.g. un sistem de gestiune a stării de sănătate pentru o persoana in vârsta). In plus, scenariile propuse in domeniul roboticii asistive nu vizează o singura persoana si un singur spațiu fizic. In cazul scenariului de ghid la muzeu sau intr-o clădire de birouri, robotul va schimba săli si, prin urmare, si conexiunea cu senzorii care se afla intr-un anumit mediu. Se disting astfel doua moduri necesare de deployment al platformei:

- Unul centralizat in care procesarea evenimentelor se face pe o singura unitate de calcul (e.g. cea instalata pe mașina) si care deservește cel mai probabil un singur client.
- Unul descentralizat, unde datele sunt distribuite (cel puțin spațial), iar procesarea se întâmpla in mod independent pe mai multe unitati de calcul (e.g. o mașina de gestionare a senzorilor din sala A si una din sala B)

Afara de modul de propagare a informației de la senzori la consumator (e.g. la robot care isi schimba poziția), pentru modul descentralizat de deployement se disting nevoie suplimentare:

- Cum anume sunt găsite nodurile ce rulează platforma ROBIN-Context, care gestionează o sala anume, atunci când robotul trece dintr-o sala in alta? Cu alte cuvinte, cum se asigura mobilitatea consumatorului. Cum se realizează decuplarea de la o instanța a ROBIN-Context si cuplarea la o alta? Care este factorul declanșator al mobilității?
- Cum si in ce condiții se permite accesul dintr-o instanța a ROBIN Context la o alta instanța (e.g. robotul dorește sa afle statusul unor senzori de mișcare din alta sala, pentru a putea determina daca exista persoane acolo, atunci când primește sarcina de a căuta o anumita persoana intr-o clădire de birouri).

O cerința de ordin tehnic, este modul in care platforma ROBIN-Context se va integra cu framework-uri existente pentru dezvoltarea de aplicații in robotica si in sisteme ADAS. In literatura de specialitate si pe piața de dezvoltare exista deja o platforma - Robotic Operating System (ROS[14]) - care este folosit atat pentru programarea robotilor (fiind conceputa pentru aceasta comunitate), cat si pentru

[14] http://www.ros.org/

prototiparea sistemelor de ADAS (e.g. Blackberry QNX15). Framework-ul are deja capabilitati extinse de definire si transmitere de mesaje, in masura de a fi procesate de roboti si masini inteligente. Ca atare, se adauga cerinta de nivel tehnic, ca platforma ROBIN-Context sa se poata interfata cu sistemul ROS dezvoltarea de aplicatii.

4.3 Arhitectura platformei de procesare a evenimentelor

Platforma ROBIN-Context pleacă de la dezvoltările făcute in direcția motorului CONSERT [Trăscău et al., 2018]. Motorul de inferența CONSERT este o componenta de raționament asupra informației contextuale, a cărei funcționalitate este similara celei unui motor de recunoaștere de evenimente complexe (semantic complex event processing - SCEP). Acesta are însa elemente distinctive inovatoare. Arhitectura elementelor si modul lor de întrebuințare sunt prezentate in continuare.

4.3.1 Functionalitățile motorului de inferență CONSERT

Motorul de inferența CONSERT se bazează pe reprezentarea informației contextuale propusa in [Sorici et al, 2015]. Pentru metamodelul CONSERT exista atât o implementare sub forma de ontologie, cat si una obiectuala (clase in limbajul de programare JAVA), cele doua fiind transformabile una intr-alta. Ciclul de operare al motorului CONSERT este descrisa in detaliu in [Sorici et al., 2015a]. In raportul de fata se va prezenta arhitectura de ansamblu a motorului si fluxul de operații executat. Sunt detaliate in mod particular următoarele aspecte:

- Inserarea si tratarea evenimentelor in funcție de tipul lor de achiziție (informație statica, de la senzori, direct din partea unei aplicații finale sau inferata de motor)
- Aplicarea inferențelor pe baza unui sistem de reguli implementat cu ajutorul framework-ului DROOLS
- Prelucrarea explicita a evenimentelor ce admit adnotări de durata temporala. In particular, motorul CONSERT implementează un mecanism de extensie a validității temporale a unei situații contextuale pe baza evenimentelor atomice.
- Aplicarea de operatori particulari pentru a deduce in mod automat noi valori pentru adnotările care sunt combinate sau extinse in timpul inferențelor (e.g. grad de încredere, timestamp)

4.3.2 Framework-ul DROOLS ca tehnologie la baza motorului CONSERT

Drools[16] este un sistem de management pentru reguli de business si include un motor de reguli de business. Suita de aplicații și platforme asociate cu Drools sunt printre cele mai folosite sisteme de business, unde este nevoie fie de fluxuri operaționale sau de sisteme expert. Pe lângă motorul de reguli implementat cu algoritmi de inferență eficienți, Drools permite funcționarea în două moduri:

- Cloud - în acest mod, regulile sunt evaluate plecând de la o bază de cunoștințe în care faptele conținute reprezintă „starea” sistemului la un anumit moment în timp
- Streaming - în acest mod, faptele se pot introduce treptat, ele având și o caracteristică temporală ce poate fi utilizată de reguli (folosind, de exemplu, principii de logică temporală)

Atunci când Drools funcționează în modul streaming el poate fi folosit pentru a sta la baza unui motor de recunoaștere de evenimente complexe. Acesta funcționează evaluând nu fapte, ci evenimente, care pot fi punctuale (apar la un moment de timp și au o durată de viață instantanee) sau bazate pe intervale (sunt definite prin momentele de activare și cel de dezactivare). Această distincție este folositoare și în cazul nostru, în special pentru că permite implementarea cu ușurință a unei părți din meta-informațiile pe care platforma ROBIN-Context își propune să le folosească.

Motorul CONSERT peste DROOLS

[15] http://blackberry.qnx.com/en/products/adas/index
[16] https://www.drools.org/

Deși framework-ul Drools este robust, este necesar de luat în calcul faptul că specificațiile CONSERT impun o serie de modificări și extinderi ale acestuia pentru a putea fi integrat cu succes în proiect. Mecanismele cu care Drools trebuie extins pot fi identificate analizând în primul rând ciclul de inferență care trebuie implementat în CONSERT (cf. Figura 4.2).

Înainte de a fi preluat de modulul de inferență și pasat rețelei de noduri a algoritmului *PHREAK* (mai specific, nodurilor care filtrează tipurile de obiecte - *Object Type Nodes*), evenimentul trece printr-o *etapă de validare și verificare a adnotărilor* (verificarea *Continuity Check* din Figura 4.2).

Tot în această etapă sunt verificate și validitatea valorilor evenimentului (verificare Constraint Check in Figura 4.2). Ulterior verificării dacă evenimentul trebuie adăugat unei extinderi (sau nu) și validarea valorilor pentru adnotări, middleware-ul CONSERT trebuie să verifice constrângeri de consecvență specifice evenimentului. După ce evenimentele sunt validate dar înainte să fie inserate, este necesară realizarea etapei de inferență ontologică. În cadrul platformei Robin-Context, evenimentele vor putea fi definite și folosind o ontologie. Aceasta poate fi folosită pentru a exprima evenimentele ca entități de diverse tipuri și cu diverse proprietăți, între care există relații de subsumare (sau alte tipuri). Folosind aceste relații, utilizarea unei ontologii permite stabilirea gradului de granularitate al cunoștințelor dorit.

Odată parcurși pașii de mai sus (validarea valorilor, verificarea constrângerilor și realizarea inferențelor ontologice) evenimentul este în cele din urmă introdus în motorul de reguli de inferență. Acesta va genera acțiunile necesare sau va produce noi cunoștințe bazându-se pe nodurile terminale ale rețelei de inferență. Este important de notat faptul că orice eveniment inferat este reintrodus în sistem, completând astfel ciclul de inferență. El va fi tratat asemenea unui eveniment nou introdus și va fi trecut prin aceleași etape descrise mai sus.

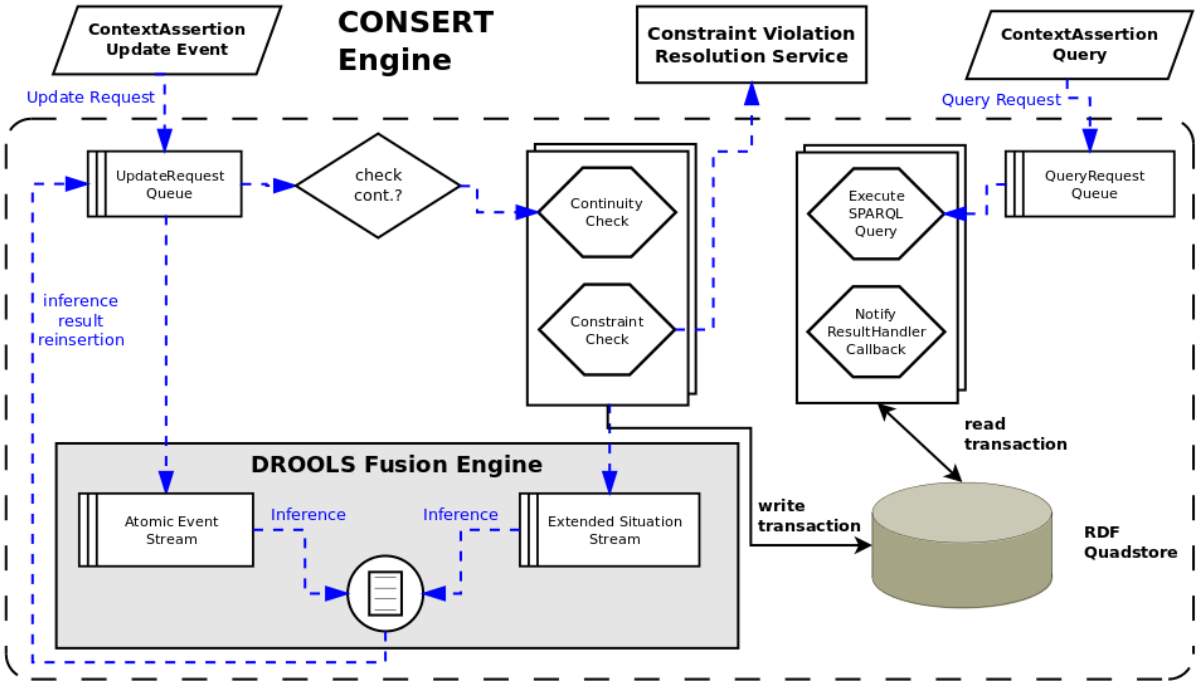


Figura 4.2. Arhitectura motorului CONSERT si fluxul de informații din cadrul ciclului de procesare

4.3.3 Procesare bazată pe cunoștințe

Sub implementarea actuala, motorul CONSERT este capabil de a face procesări de evenimente folosind reguli definite a-priori, ce surprind condițiile valorice si de ordonare in timp ale evenimentelor. Avantajul unei astfel de abordări este ca pot fi surprinse cu ușurința cunoștințe de bun simt, aplicabile unei aplicații sau unui utilizator, definite de către un expert. In Listarea 1, este

exprimata, de exemplu, o regula ce definește daca o persoana ramane intr-o camera (in cazul dat, in baie) in funcție de activarea unui senzor de mișcare plasat in acea camera.

```
rule "Remain in the Bathroom"
when
        $loc : PersonLocation(p: person, room : loc == "Bathroom", locAnn : annotations)
        not( exists PersonLocation(person == p, loc != room,
                this annOverlappedBy[0s, 5s] $loc
                    || $loc annIncludes this))
        not( exists Motion(status == "ON", this annHappensAfter[0s, 5s] $loc))
then
        long ts = eventTracker.getCurrentTime();
        DefaultAnnotationData ann = new DefaultAnnotationData(ts);
        PersonLocation sameLoc = new PersonLocation("Bathroom", ann);
        eventTracker.insertAtomicEvent(sameLoc);
end
```

Listarea 1. Regula de inferenta contextuala care se activeaza atunci cand o persoana se afla in baie si ramane acolo.

Cu toate acestea, o situație mai complexa (e.g. faptul ca o persoana face curat in bucătărie sau faptul ca sta si gateste) pot fi mai anevoios de exprimat, in special in momentul in care senzorii aflați la dispoziție oferă putina informație (e.g. doar senzori de mișcare aflați in camere si senzori de închidere/deschidere al ușilor). In cel din urma caz, secvența in sine a activării senzorilor de mișcare (nu doar in ce camera sunt, si timpii intre activări) pot fi definitorii pentru activitatea desfășurata (e.g. a face curat prin bucătărie este o activitate ce implica mișcare mai deasa, spre deosebire de cea de a găti). Pentru a putea distinge intre astfel de șabloane de activare, se propune extensia modului de operare pe baza de cunoștințe cu unul pe baza de analiza probabilistica a evenimentelor.

4.4 Utilizarea modelelor probabilistice pentru detectarea activităților

S-a explicat mai înainte faptul că, pentru anumite tipuri de activitati, detecția se poate face mai eficace printr-o procedura statistica de invatare a secvenței de evenimente de senzori. In literatura de specialitate, aceasta problema cunoaște multiple abordări. Tipul de abordare pentru detecția activităților prin metode de invatare diferă semnificativ in funcție de tipul de date disponibile ca intrare, astfel ca devine utila descrierea exacta a tipului de intrări pe care platforma ROBIN-Context isi propune sa le exploateze. O distincție se face intre *intrări de tip secvența video* si *cele de tip senzori personali sau de mediu.*

In ROBIN-Context se dorește exploatarea celei de-a doua opțiuni, in mod special a senzorilor de mediu. Motivul principal este ca acestea constituie un mijloc mult mai puțin intruziv in ceea ce privește setul de dispozitive instalate in casa unui utilizator / intr-un birou inteligent. Senzori de mediu precum cei de mișcare[17,] cei de detecție a unei uși închise/deschise[18,] prize inteligente (care detectează când un aparat este sub tensiune) sau contoare inteligente de gaz/apa pot fi instalați fara cerințe speciale de spațiu, spre deosebire de camere video care au nevoie de vizibilitate suplimentara, trebuie instalate in fiecare camera (spre a fi utile in mai multe tipuri de activitati) si care prezintă probleme de incalcare a vieții private. Compromisul in alegerea senzorilor de mediu este acela de a avea date mai puțin informative (intrucat dintr-o imagine se poate deduce mai mult context, decât din activări razlete de senzori).

Cu toate acestea, tipul acesta de senzori este suficient pentru a detecta tipuri de activitati de o granularitate mai redusa precum: a dormi, a lucra, a pregăti mâncare, a se uita la televizor, a face curat in casa, a se toaleta - utile in scenariile roboticii asistive. Pentru detecția activităților de nivel înalt pe baza activării in timp senzorilor de mediu au fost propuse mai multe metode de invatare.

[17] Exemplu de senzor de miscare: https://www.fibaro.com/en/products/motion-sensor/
[18] Exemplu de senzor de detectie a usii deschise: https://www.fibaro.com/en/products/door-window-sensor/

[Cook, 2010] prezintă o recenzie a trei tipuri de algoritmi probabilistici clasici utilizați pentru clasificarea secvențelor: Naive Bayes, Modele Markov Ascunse si Conditional Random Fields. Lucrarea analizează acuratețea algoritmilor pe seturi diferite de date colectate in cadrul proiectului CASAS[19]. Seturile diferă in funcție de dimensiunea mediului analizat, tipul activităților desfășurate, numărul de senzori instalați si numărul de persoane care desfasoara activitățile. Data fiind varietatea, rezultatele au arata ca nici unul din algoritmi nu domina in mod clar un altul. Mai mult, cele mai bune rezultate s-au obținut atunci când au fost folosite metode de invatare suplimentare de tip ansamblu [Zhang & Zhang, 2008] sau de invatare semi-supervizata. Un aspect interesant observat de [Cook, 2010] este ca invatarea pe mai multe seturi de date (aduse la un numitor comun) va creste acuratețea de detecție a activităților pentru seturile de date mici care aveau o acuratețe scăzuta in mod de sine statator si va scădea acuratețea pe cele mari. Procentul creșterii este însa mai mare decât de scădere, demonstrându-se astfel capabilitatea algoritmilor de a generaliza mai bine, dându-se setupuri de senzori diverse (i.e. activitățile desfășurate au un grad rezonabil de similaritate in execuție raportat la diversitatea persoanelor care le executa).

Daca in [Cook, 2010] atributele algoritmilor probabilistici sunt insasi activările de senzori, in [Liu et al, 2016] accentul cade pe relația intre evenimente ca atribute primare pentru algoritmi clasici de clasificare precum Naive Bayes, k-Nearest Neighbours sau Support Vector Machines (SVM). Relația intre evenimente se construiește sub forma matriciala intre 2 sau 3 evenimente si folosește relațiile intre intervale definite de logica Allen[20]. Pe baza unui set de date, se calculeaza intai tipurile de secvențe care se regăsesc frecvent in setul de date (printr-o abordare de Frequent Temporal Pattern Mining). Din acestea, pentru fiecare activitiate, se creează un vector de atribute pe seama tipurilor de secvențe care apar in activitate, fiecare ponderata in funcție de frecventa apariției in activitate. Acești vectori sunt apoi dați ca intrare algoritmilor de clasificare menționați.

Intr-o abordare complet diferita, activitățile sunt descrise la nivel semantic sub forma unor reguli ce leagă evenimentele intre ele spre a analiza condițiile necesare deducerii unei activitati. Diferența fata de o abordare pe baza de cunoștințe este aceea ca regulile acestea sunt ponderate cu un grad de probabilitate al veridicității, iar inferența se face pe baza Markov Logic Networs [Richardson & Domingos, 2006]. O astfel de abordare este folosita, de exemplu, in [Riboni et al., 2016], care isi evaluează abordarea pe un setup din proiectul CASAS.

Provocarea in proiectul ROBIN-Context vine in a găsi o metoda prin care aceste abordări sa poată fi cuplate cu o metoda bazata pe cunoștințe, care sa acționeze in sensul informării suplimentare / ghidării algoritmilor de inferența probabilistica.

4.5 Continuarea cercetărilor

In etapa următoare, eforturile de cercetare si dezvoltare se vor concentra pe următoarele:

- Implementarea nodurilor ROS si a protocoalelor de comunicare pe baza de topicuri, având ca scop interfațarea cu frameworkul ROBIN-Context
- Implementarea unei serii de evaluări a algoritmilor probabilistici de detecție a activităților pe seturi de date care au fost prelucrate deja in mod bazat pe cunoștințe. Efectuarea unor comparatii la nivel de acuratețe intre cele doua abordări.
- Elaborarea unei metodologii de colectare continua a datelor de mediu, necesare antrenării algoritmilor probabilistici intr-un deployment real (i.e. nu pornind de la date culese anterior)
- Proiectarea si dezvoltarea la nivel de flux de informație a metodei de combinare a inferențelor pe baza de cunoștința cu cele probabilistice selectate in urma evaluării.

Raportul in extenso asociat etapei I a proiectului ROBIN-Context se găsește la adresa http://aimas.cs.pub.ro/robin/rezultate/

[19] http://casas.wsu.edu
[20] https://www.ics.uci.edu/~alspaugh/cls/shr/allen.html

5. Proiectul component ROBIN-Dialog

Parteneri ai proiectului ROBIN-Dialog

INSTITUTUL DE CERCETĂRI PENTRU INTELIGENȚĂ ARTIFICIALĂ „MIHAI DRĂGĂNESCU" (CO)
UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI
UNIVERSITATEA TEHNICĂ DIN CLUJ – NAPOCA

5.1 Prezentare generală

Pe măsură ce tehnologiile inteligente și adaptive au devenit din ce în ce mai integrate în viața personală, este de așteptat ca roboții asistivi să devină adevărați parteneri și companioni ai utilizatorilor umani pe care îi servesc. Totuși, roboții de uz general nu sunt încă gata, dar observăm tehnologii emergente în viitorul apropiat care să ajute oamenii. Una din funcționalitățile fundamentale pentru acceptarea unui robot asistiv este capacitatea sa comunicațională. Domotica modernă presupune, de asemenea, interacțiunea naturală prin comenzi în limbaj natural.

Pentru limba română, interacțiunea prin voce este încă o mare provocare, cu câteva experimente încurajatoare, dar limitate. Contextul comunicațional de interes pentru acest proiect este dialogul situațional, care presupune raportarea la realitatea imediată. Una din cele mai eficiente metodologii de proiectare a componentei de comunicare situațională în limbaj natural se bazează pe scenarii pentru micro-lumi. Marele avantaj al acestei abordări constă în posibilitățile de a anticipa intențiile partenerului uman, conținutul cel mai probabil al unei cereri sau comenzi, precum și de a formula solicitări inteligente de clarificare în condițiile insuficienței cunoștințelor pentru prelucrarea mesajului transmis.

Obiectivul proiectului ROBIN-Dialog presupune dezvoltarea unei serii de scenarii pentru câteva micro-lumi și tehnologia de prelucrare a limbii române pentru dialoguri situaționale în aceste micro-lumi. Această tehnologie va fi validată pe scenariile și micro-lumile cercetate, dar va fi dezvoltată în așa fel încât să poată fi aplicată cu ușurință și pe alte scenarii și/sau micro-lumi. Caracterul de generalitate va fi asigurat de metodele de învățare automată de tip ”deep learning” și de specificarea resurselor (e.g. baze de cunoștințe) în limbaje standard (e.g. XML/RDF) care vor asigura funcționarea sistemului în orice micro-lume și/sau scenariu, atâta timp cât datele de antrenare și resursele specifice vor fi disponibile pentru acestea.

1. Proiectarea acestor scenarii și a sistemului de dialog situațional în limba română presupune următoarele activități:
   a. Construirea unui lexicon de cuvinte și expresii reprezentative pentru micro-lumea țintă. Exemple de micro-lumi sunt: i. o casa inteligentă; ii. Un robot acționând într-un mediu/spațiu specificat
   b. Extensia automată utilizând metode semantice moderne („continuous vector spaces”) a lexiconului creat manual la pasul 1a
   c. Crearea universului de discurs pentru micro-lumea/scenariul selectat. Acest pas implică identificarea relațiilor semantice care se stabilesc între cuvinte și care astfel devin predicate care vor fi validate (adevărat/fals) în contextul dialogului.
2. Intrările resursei lexico-semantice creată în pasul 1b vor fi transcrise fonetic și aliniate cu semnalul vocal corespunzător în cazul în care aceste înregistrări există în CoRoLa.
3. Sistemele de antrenare ASR și TTS vor fi alimentate cu rezultatele pasului 2. Sistemele ASR si TTS vor fi testate si validate.
4. Implementarea sistemului de dialog cooperant pentru micro-lumile selectate.

Activitățile Etapei I a proiectului au fost următoarele.

Activitatea: ***Act 1.12 - Definirea specificațiilor funcționale și arhitecturale ale modulelor software și micro-lumilor*** Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: Specificații funcționale ale sistemului si micro-lumilor

Obiectivul a fost integral îndeplinit, s-au definit micro-lumile țintă și structura de principiu a sistemului de dialog în microlumile respective

Activitatea: ***Act 1.13 - Construirea lexiconului/lexicoanelor de cuvinte și a expresiilor reprezentative pentru micro-lumea/lumile țintă*** Categorie activitate:A1 - Cercetare fundamentală Indicatori de realizare: Lexicon in format electronic - descrierea lexiconului

Obiectivul a fost integral îndeplinit, a fost construit lexiconul micro-lumilor țintă. Pe baza scenariilor dezvoltate în micro-lumile vizate de proiect, a fost dezvoltat un lexicon inițial, conținând 587 de forme (corespunzând unor 318 perechi lemă/parte de vorbire). Formatul lexiconului în această etapă a proiectului este de tip tabular (cu coloane separate prin tab-uri), în care pe prima coloană se regăsește forma ocurență a cuvântului, pe a doua se regăsește lema (forma de dicționar) iar pe a treia observăm descrierea (eticheta) morfo-sintactică a cuvântului (verb, modul indicativ, timpul prezent, persoana a 3-a, numărul plural). Setul inițial de leme/forme a trecut prin mai multe etape de extindere, pentru a crea un lexicon cât mai comprehensiv, care să ajute la adaptarea sistemului de dialog la diverse variante de exprimare a aceluiași conținut semantic. Într-o primă etapă, pornind de la acest set inițial de leme/forme, s-au extras din corpusul COROLA [Mititelu et al., 2018] reprezentări vectoriale învățate automat, cunoscute și ca "word embeddings" [Paiș and Tufiș, 2018]. Acestea sunt reprezentări vectoriale dense în spații vectoriale de mici dimensiuni, care pot fi folosite în aplicații de prelucrare a limbajului natural.

Activitatea: ***Act 1.14 - Extensia automată a lexiconului/lexicoanelor creat(e) manual***. Validarea și corectarea lexiconului/lexicoanelor extins(e) Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: L10 - Descriere lexicoane de cuvinte extinse automat, lexicon extins

Obiectivul a fost integral îndeplinit, lexiconul micro-lumilor a fost extins automat, dar etapele intermediare ale extensiei au fost validate manual. În contextul proiectului nostru, vectorii semantici au fost folosiți pentru a identifica cuvinte similare (ca încărcătură semantică) cu cuvintele din setul inițial de leme. Vectorii de similaritate au fost generați atât pentru cuvinte în forma lor ocurență cât și în forma dicționar și au fost validați manual pentru a îndepărta eventualele cuvinte care se îndepărtează foarte mult de universul micro-lumilor cu care lucrăm. Toate cuvintele rezultate din această primă etapă de extindere, au trecut printr-un proces de identificare în RoWordnet (vezi referință) a hiperonimelor (unitate lexicală cu semnificaţie mai generală şi mai extinsă faţă de cea de la care se pornește) și sinonimelor asociate. Cuvintele rezultate au fost validate manual și introduse (eliminând duplicații) la rândul lor în lexicon. Ultima etapă este cea de asociere a informației de lemă și etichetă morfologică pentru cuvintele colectate și de extindere a lexiconului cu toate cuvintele din familiile paradigmatice ale fiecărei leme deja existente, făcând apel la lexiconul dezvoltat la ICIA, tbl.wordform.ro (o resursă cu peste 1.150.000 de intrări, de forma *<formă ocurentă>tab<lemă>tab<etichetă morfo-sintactică>).* Această ultimă etapă nu necesită validare manuală, deoarece tbl.wordform.ro este o resursă care a trecut prin etape de validare pe măsură ce a fost îmbogățită. Într-o etapă ulterioară a proiectului (etapa II, livrabilul 10), lexiconul va fi îmbogățit cu informația de transcriere fonetică, silabație și poziționarea accentului.

Activitatea: ***Act 1.15 - Diseminare*** Categorie activitate: D1 - Activități suport - Diseminarea pe scară largă prin comunicarea și publicarea națională sau internațională a rezultatelor Indicatori de realizare: 1 lucrare științifica

Obiectivul a fost integral indeplinit, fiind prezentată la **Conferința internațională LREC 2018** (conferință ISI) de la Miyazaki, Japonia, lucrarea Dan Tufiș, Dan Cristea A Bird's-eye View of Language Processing Projects at Romanian Academy, Miyazaki, Japan, LREC 2018, pp. 2445-2451, ISBN 979-10-95546-00-9.

La **Conferința internațională *"Linguistic Resources and Tools for Processing of the Romanian Language"***, *a fost prezentată lucrarea* Nenciu, B., Ruseti, S., & Dascalu, M. (2018). Extracting Actions from Romanian Instructions for IoT Devices. In V. Pais, D. Gifu,

D. Trandabat, D. Cristea & D. Tufis (Eds.), 13th Int. Conf. on Linguistic Resources and Tools for Processing Romanian Language (ConsILR 2018) (pp. 168–176). Iasi, Romania.

Toate obiectivele asumate de proiectul component ROBIN-Dialog pentru Etapa I au fost indeplinite complet.

Structura ofertei de servicii de cercetare şi tehnologice Institutul de Cercetări pentru Inteligență Artificială (ICIA/RACAI), va oferi pe platforma ERRIS serviciile de cercetare şi tehnologice enumerate în tabelul următor (https://erris.gov.ro/RACAI-ICIA):

Servicii:

Interogare corpus de referință al limbii române corola.racai.ro; TTL http://ws.racai.ro/ttlws.wsdl, Modular Language Processing for Lightweight Applications (MPLA) cu prelucrări pentru mai mult de 40 de limbi http://slp.racai.ro/index.php/mlplanew/ Sistemul ROBIN-dialog de prelucrare a dialogurilor în micro-lumile țintă; Lexicon extins pentru aplicații de prelucrare a vorbirii; Sistem de detecție de cuvinte cheie în conversații înregistrate http://heimdall.racai.ro/ Romanian Spoken Language Processing; http://rsp.racai.ro Romanian Anonymous Speech Corpus http://rasp.racai.ro.

5.2 Sisteme de dialog bazate pe scenarii

Sistemele de dialog în limbaj natural sunt programe de calculator care conversează cu utilizatorul într-o limbă dată (de ex. în limba română) cu scopul de a furniza informații. De exemplu, există sisteme de dialog care ajută utilizatorul să înțeleagă cum se rezolvă o problemă de fizică [Rus et al., 2013], să cumpere un produs [Yan et al., 2017] sau chiar să asiste doctorul la punerea unui diagnostic [Liu et al., 2018], prin conversația directă cu bolnavul.

Aceste sisteme de dialog sunt foarte specializate astfel încât ele funcționează numai în scenarii predefinite[21]. Într-un astfel de scenariu, din punctul de vedere al programului, dialogul se află într-o stare bine definită după fiecare replică primită de la utilizator: aceeași dacă programul nu a înțeles ce a spus utilizatorul sau o stare nouă dacă sistemul de dialog a analizat corect spusele utilizatorului și a deslușit calea de urmat în graful care formalizează structura dialogului. În momentul în care sistemul de dialog ajunge la un nod din care nu mai poate avansa (nu mai există arce care să iasă din nod), dialogul s-a încheiat cu succes iar sistemul de dialog poate executa acțiunile asociate cu această stare finală (de exemplu să afișeze diagonosticul sau produsul recomandat).

O micro-lume reprezintă o specificație a mediului și a scenariului în care se desfășoară dialogul. Pentru sistemele de dialog exemplificate mai sus, mediul în care se desfășoară dialogul este calculatorul iar acesta nu execută nicio acțiune în afară de a furniza informațiile căutate de utilizator. Însă, în cazul în care sistemul de dialog se atașează unui robot care poate executa acțiuni (se poate deplasa, poate ridica obiecte, poate identifica persoane, etc.), atunci când programul de dialog a atins o stare finală care are asociate acțiuni specifice, robotul le poate executa, *în mediul pe care îl cunoaște*.

Să luăm de exemplu micro-lumea „căutării persoanei X în sala de clasă” de către un robot care se poate mișca prin clasă și, cu ajutorul camerei sale video, poate identifica o persoană. Această micro-lume trebuie să precizeze/ofere:

- Ce persoane se află în clasă
- În ce bancă stă fiecare persoană
- Coordonatele (x, y) ale fiecărei bănci astfel încât robotul să poată ști unde să se ducă
- O imagine (sau mai multe) în format electronic pentru fiecare persoană astfel încât programul de recunoaștere facială să poată fi antrenat.
- O structură (arborescentă) a dialogului de urmat:
    - Formula de salut/start dialog: „Salut Pepper!”, „Bună Pepper!”, etc.

[21] Încă nu există un sistem de dialog universal care să poată conversa cu utilizatorul uman despre un subiect arbitrar, fără niciun scenariu de dialog. Acesta este deziderатul Inteligenței Articiale, conform unuia dintre teoreticienii celebri ai domeniului, Alan Turing (https://en.wikipedia.org/wiki/Turing_test).

- Cererea principală: „F a venit?” (numai numele de familie F e precizat)
  - Robot (în cazul în care sunt mai multe persoane cu același nume de familie F): „Vă referiți la X sau Y?”
    - Utilizator: „La X.”
      - Stare finală: robotul se poate deplasa la banca în care stă F X, îl/o poate recunoaște și se poate întoarce pentru a confirma prezența.
  - Stare finală (în cazul în care există o singură persoană cu numele de familie F): robotul se poate deplasa la banca în care stă F X, îl/o poate recunoaște și se poate întoarce pentru a confirma prezența.
- Cererea principală: „F X a venit?” (numele complet e precizat)
  - Stare finală (în cazul în care există o singură persoană cu numele de familie F): robotul se poate deplasa la banca în care stă F X, îl/o poate recunoaște și se poate întoarce pentru a confirma prezența.

O micro-lume poate exista în realitate (ca în exemplul de mai sus) sau poate fi simulată pe calculator, caz în care devine virtuală (vezi, de exemplu, jocurile 3D interactive în care personajul controlat de utilizator interacționează verbal cu personaje controlate de calculator). Oricare ar fi natura unei micro-lumi, inginerul de cunoștințe trebuie să se asigure că *a enumerat toate modurile în care se poate interacționa cu micro-lumea*. De aici rezultă că, cu creșterea complexității micro-lumii, această enumerare devine din ce în ce mai dificilă, până în momentul în care devine imposibilă pentru „micro-lumea” care este egală cu realitatea, cel puțin cu mijloacele de descriere care există acum.

5.3 Exemplificare a interacțiunior simulate în micro-lumile alese

Folosirea robotului Pepper ca un asistent la vizite in centrul de cercetare PRECIS:

- Robotul intalneste oameni pe hol care:
 * se prezinta pentru ca robotul sa realizeze asocierea intre fata si nume:
Robot: Salut! / Bună! Eu sunt Pepper. / Mă cheamă Pepper. Tu cine ești? / Pe tine cum te cheamă?
Utilizator: [Salut!] / [Bună!] / [Bună ziua!] / [Bună dimineața!] / [Neața]. (Eu) sunt .... / Mă cheamă .... /
 ... #Își spune doar numele.#

* il roaga sa ii conduca pana la o anumita zona (spre exemplu, laboratorul 308).
Utilizator: Arată-mi drumul spre laboratorul ...!
(Îmi arăți) Unde este /e laboratorul ...?
Cum ajung în laboratorul ...?
Am treabă în laboratorul .... Poți să-mi spui unde e?
Robot: Urmează-mă! / Vino după mine! (robotul ar trebui sa se asigure ca persoana il urmeaza):

* il roaga sa ii transmita un mesaj unei anumite persoane din laborator (prin urmare robotul trebuie sa stie cine e acea persoana, sa o caute pe hol, sa o identifice si sa o notifice)
Utilizator: Spune-i ... / (lui ...) (să ...) / (că ...) [Robotul ar trebui să reproducă mesajul.]
Robot: ..., X spune / (a spus) / zice / (a zis) (să ...) / (că ...). [Atenție la variabila X aici: e numele utilizatorului care i-a dat robotului o sarcină.]
#mini-dialog
utilizator X: Caut-o / caută-l pe Y și roag-o / roagă-l să vină până la mine. Dacă este [la biroul ei / lui] / [așezat/așezată pe scaun], nu o / îl deranja, dar întoarce-te și anunță-mă.
robot: Bună, Y! X te roagă să mergi până la el/ea.
utilizator Y: Bună, Pepper! (Spune-i lui X că) sunt foarte ocupat/ocupată. Merg mai târziu/ Bună, Pepper! (Spune-i lui X că) vin imediat.

* il roaga sa caute daca exista un anumit tip de obiect intr-o anumita zona (spre exemplu, cauta cana verde pe catedra din laboratorul 306)
Utilizator:
(Îl/o/îi/le) Vezi ... pe (vre)undeva? Unde este / sunt ...? Nu (îl / o / le / îi) găsesc .... Poți să mă ajuți? Poți să mă ajuți să găsesc / caut ...? Este ... aici / [în acest laborator] / [în laboratorul acesta / ăsta]?
Robot:
... este / e / [se află] (pe ... #la cazul acuzativ#) (sub ... #la cazul acuzativ#) (lângă ... #la cazul acuzativ#) (în ... #la cazul acuzativ#) (deasupra ... #la cazul genitiv#) (în fața ... #la cazul genitiv#) (în spatele ... #la cazul genitiv#) (dedesubtul ... #la cazul genitiv#) (între ... #la cazul acuzativ# și ... #la cazul acuzativ#).
Am realizat și testat de asemenea următoarele scenarii (**prezentate în raportul in extenso**):

- Pepper pentru asistența unei persoane în varstă
- Pepper ca asistent de vânzări în magazinul de electronice, raionul laptopuri
- Pepper ca partener de dialog ”promoțional (small-talk)”

5.4 Alegerea instrumentelor pentru dezvoltare

Am investigat performanțele oferite de mai multe instrumente existente pentru antrenarea și testarea unui sistem de recunoaștere automată a vorbirii. În literatura de specialitate, cele mai folosite instrumente sunt:

- CMUSphinx - conține mai multe unelte open-source care pot fi folosite pentru dezvoltarea aplicațiilor în domeniul recunoașterii vocale. Acesta este dezvoltat în limbajul C și implementează o detecție continuă, independentă de vorbitor, folosind modele Markov cu stări ascunse pentru modelarea acustică și modele statistice de tip n-gram pentru modelarea lingvistică. Acesta este folosit în principal pentru sisteme cu resurse limitate, existând și o variantă destinată sistemelor embedded, numită Pocketsphinx.
- Microsoft Azure Speech to Text API - un API dezvoltat de către Microsoft. Folosirea sistemului este contra-cost: se plătește la nivel de oră. Sistemul nu are suport pentru limba română, dar permite crearea de modele acustice / lingvistice proprii. Este în principal folosit de către companii.
- Kaldi - similar cu CMUSphinx, conține unelte open-source care pot fi folosite pentru dezvoltarea aplicațiilor în domeniul recunoașterii vocale. Este dezvoltat în limbajul C++ și implementează inclusiv rețele neurale profunde. Poate fi antrenat fie dependent, fie independent de vorbitor, și poate fi folosit atât online (datele sunt trimise și prelucrare pe parcursul înregistrării) cât și offline (datele se trimit și prelucrează doar la finalul înregistrării).

Dintre acestea trei, a fost ales Kaldi întrucât a stat la baza proiectelor cu cele mai bune rezultate raportate in literatura de specialitate. Pentru antrenarea unui model cu ajutorul Kaldi sunt necesare mai multe etape de preprocesare. Pentru antrenarea propriu-zisă sunt necesare următoarele fișiere:

- Fișiere audio, precum și textul aferent lor
- Modelul de limbă
- Dicționarul fonetic (cuvintele ce pot fi decodate, urmate de transcrierea lor fonetică)
- Descrierea fonemelor

Pentru crearea modelului de limbă, utilitarul SRILM a fost ales. Acesta primește drept intrare o serie de propoziții, și returnează un model lingvistic bazat pe distribuția statistică a cuvintelor și grupurilor de cuvinte.

O parte din timp a fost alocată pentru înțelegerea și familiarizarea cu Kaldi. În vederea familiarizării cu Kaldi, au fost rulate exemplele oferite pe o bază de date standard, numită TEL-DIUM. Baza conține înregistrări audio în limba engleză a conferințelor TED (un total de 120 ore), transcrierile lor aferente, dicționarul fonetic (cu aproximativ 160.000 cuvinte) și modelul de limbă.

5.5 Pregătirea pentru antrenarea unui model acustic specific limbii române

În vederea antrenării unui model acustic cu ajutorul utilitarului Kaldi, a fost necesară crearea principalelor fișiere menționate anterior. Corpusurile strânse în cadrul centrului au fost prelucrate pentru a satisface cerințele utilitarului Kaldi. Astfel, următoarele operații au avut loc:

- Agregarea tuturor fișierelor audio într-o singură bază de date omogenă
- Crearea unui fișier conținând transcrierea audio pentru fiecare fișier audio din baza de date
- Transcrierea fonetică a fiecărui cuvânt ce apare în fișierele audio
- Crearea unui model statistic de limbă folosind modele tri-gram și propozițiile fișierelor audio

Etapa de agregare a fișierelor audio a presupus trecerea semnalului audio printr-un filtru de detecție a vorbirii (VAD - voice activity detection) pentru a elimina zonele de liniște de la începutul / finalul semnalului. De asemenea, a avut loc și o validare a fișierelor: au fost păstrate doar fișierele mono-canal și cu frecvență de eșantionare 16Kz (caracteristici necesare pentru antrenarea cu Kaldi). La final, s-au obținut aproximativ 46.000 de cuvinte în dicționarul fonetic necesar antrenării. Durata totală a fișierelor audio este de aproximativ 105 ore.

5.6 Continuarea cercetărilor

In etapa următoare, eforturile de cercetare si dezvoltare se vor concentra pe următoarele:

- Transcrierea fonetică a cuvintelor din lexiconul validat în prima etapă
- Alinierea cu semnalul vocal corespunzător
- Crearea înregistrărilor vocale pentru cuvinte pentru care nu există înregistrări în corpusul CoRoLA
- Alimentarea sistemelor de antrenare ASR și TTS și detalierea modelelor specifice ASR și TTS
- Implementarea prototipului generic de sistem de dialog cooperant și configurarea pentru un robot asistiv
- Validarea a unui număr suplimentar de scenarii în conjuncție cu scenarii de navigare li identificae de persoane cu robotul Pepper

Raportul in extenso asociat etapei I a proiectului ROBIN-Dialog se găsește la adresa http://aimas.cs.pub.ro/robin/rezultate/

6. Proiectul component ROBIN-Cloud

Parteneri ai proiectului ROBIN-Cloud

UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI (CO)
UNIVERSITATEA TEHNICĂ DIN CLUJ – NAPOCA
UNIVERSITATEA "DUNAREA DE JOS" DIN GALAȚI

6.1 Prezentare generală

Proiectul ROBIN-Cloud urmărește crearea de module software şi servicii pentru utilizarea roboților într-o societate digitală interconectată, care să permită companiilor realizarea de produse și servicii complexe, inteligente și performante destinate acestor utilizatori cât și societății în ansamblu. Astfel de sisteme de roboți au la bază senzori, din care sunt preluate date și analizate, respectiv sunt extrase informații de nivel mai înalt, pe baza cărora se construiesc sisteme de reguli. Aceste sisteme de

prelucrare și procesare a datelor (de cele mai multe ori de mari dimensiuni, sau generate în fluxuri de viteză crescută - ex., o cameră montată pe mașina autonomă captează mai multe cadre pe secundă) necesită capacități de procesare și stocare mărite, motiv pentru care multe companii preferă folosirea, acolo unde e posibil, unui mediu Cloud suport. Datele sunt captate astfel de senzori, transferate prin diverse tehnologii de comunicații (cu fir sau fără fir) către mediul Cloud, unde se realizează procesarea și extragerea de informații, ce sunt ulterior transmise înapoi către mecanismele de control decizional inteligent.

Din ce în ce mai mult se vorbește de Cloud Robotics iar prelucrările intensive, inclusiv algoritmi puternici de învățare, necesare sarcinilor complexe ale roboților pot fi executate în Cloud. Sistemele de roboți au nevoie de garanții în privința timpului de răspuns a rezultatului, caz în care timpul pentru transferul datelor în Cloud începe să conteze. Acesta este unul dintre argumentele din spatele modelului Edge Computing, unde mecanisme de procesare și analiză sunt aduse cât mai aproape de datele colectate. În cadrul proiectului, acesta înseamnă realizarea de extensii Cloud către marginea lui, așa încât aproape de senzori putem imagina poziționarea unor plăci de procesare sau dispozitive set-top-box, ce preiau din procesare (procesarea acelor date critice), respectiv extensia către Cloud (procesarea de date mai puțin critice, predicții pe date de lungă durată).

Obiectivul proiectului ROBIN-Cloud îl constituie crearea unei platforme suport pentru colectarea de date provenind de la senzorii unor sisteme suport pentru roboți (ex. IoT), oferirea de mecanisme de procesare și învățare combinând modele Cloud cu dispozitive aflate aproape de sursa de colectare, oferirea de biblioteci suport pentru procesare inteligentă / semantică a datelor, și în final dezvoltarea de servicii Web centrate spre agenții economici interesați de folosirea datelor stocate la nivelul platformei. *Obiective specifice sunt*:

- Proiectarea și realizarea unei platforme suport pentru colectarea și sistematizarea de date captate de la nivelul unor sisteme IoT și roboți, peste arhitectura IoT-A ARM;
- Proiectarea și realizarea unor mecanisme de procesare și învățare distribuită pe baza datelor de la senzori, în tehnologie hibridă Cloud / Edge Computing (componentele Complex Event Processing și Context Broker);
- Dezvoltarea unei biblioteci de algoritmi de procesare semantică a datelor pe platforme Cloud;
- Proiectarea și realizarea de tehnici, metode și algoritmi pentru căutarea și extragerea cunoștințelor din Depozitul de Date la nivelul platformei ROBIN-Cloud;
- Dezvoltarea în Cloud a unor componente de suport pentru Machine Learning (algoritmi deep-learning, deep-reinfocement learning, support vector machine, etc.) specifice pentru prelucrarea limbajului natural, mașini autonome, dar și pentru colaborarea roboților în mediul social.
- Planificare inteligentă și adaptivă prin crearea unui set de algoritmi pentru: procesarea în Cloud, cuplarea cu SLAM, planificarea rutelor mașinilor autonome, analiza limbajului natural, etc.
- Proiectarea, realizarea și testarea de instrumente și servicii web destinate agenților economici.

. ***Activitățile Etapei I a proiectului*** au fost următoarele

Activitatea: ***Act 1.16 - Analiza modelelor de interconectare a sistemelor eterogene Cloud-Edge si a modelelor de colectare și procesare a datelor în timp real cu constrângeri de calitate*** Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: L11 - Studiu privind modele de interconectare a sistemelor Cloud-Edge

Obiectivul a fost integral îndeplinit, vezi secțiunea 6.2 din prezentul raport și raportul in extenso.

Activitatea: ***Act 1.17 - Definirea specificațiilor funcționale și arhitecturale pentru componentele ROBIN-Cloud si a tipurilor de date ce vor fi colectate și procesate în cadrul platformei*** Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: L12 - Analiza cerințelor utilizator, formatul datelor, specificații de protocol și metode specifice de procesare

Obiectivul a fost integral îndeplinit, vezi secțiunea 6.3 din prezentul raport și raportul in extenso.

Activitatea: ***Act 1.18 - Implementarea platformei suport pentru colectarea și sistematizarea de date captate de la nivelul unor sisteme IoT și roboți, peste arhitectura IoT-A ARM*** Categorie activitate: A1 - Cercetare fundamentală Indicatori de realizare: Sistem software (platforma) Serviciu nou pentru colectare, transformare și stocare a datelor în depozitul de date central, introdus in ERRIS

Obiectivul a fost integral îndeplinit, vezi secțiunea 6.4 din prezentul raport și raportul in extenso.

Categorie activitate: D1 - Activități suport - Diseminarea pe scară largă prin comunicarea și publicarea națională sau internațională a rezultatelor Indicatori de realizare: 1 lucrare științifică

Obiectivul a fost integral îndeplinit. Lista articolelor cu Acjnowledgement ROBIN

Articole de jurnale cotate ISI

1. Mocanu, Bogdan, Florin Pop, Alexandra Mihaita, Ciprian Dobre, and Aniello Castiglione. "Data fusion technique in SPIDER Peer-to-Peer networks in smart cities for security enhance-ments." Information Sciences (2018). **Publicat (IF: 4,305, SRI: 2,107)**;
2. Cioara, Tudor, Ionut Anghel, Ioan Salomie, Marcel Antal, Claudia Pop, Massimo Bertoncini, Diego Arnone, and Florin Pop. "Exploiting data centres energy flexibility in smart cities: Business scenarios." Information Sciences 476 (2019): 392-412. **Publicat (IF: 4,305, SRI: 2,107)**;
3. Varga, Mihai, Alina Petrescu-Nita, and Florin Pop. "Deadline scheduling algorithm for sustainable computing in Hadoop environment." Computers and Security 76 (2018): 354-366. **Publicat (IF: 2,650, SRI: 1,157)**;
4. Esposito, Christian, Aniello Castiglione, Francesco Palmieri, Massimo Ficco, Ciprian Dobre, George V. Iordache, and Florin Pop. "Event-based sensor data exchange and fusion in the Internet of Things environments." Journal of Parallel and Distributed Computing 118 (2018): 328-343. **Publicat (IF: 1,815, SRI: 1,198)**;
5. Popa, Dan, Florin Pop, Cristina Serbanescu, and Aniello Castiglione. "Deep learning model for home automation and energy reduction in a smart home environment platform." Neural Computing and Applications (2019): 1-21. **Acceptat (IF: 4,213, SRI: 1,032)**;
6. Adrian Pauna, Ion Bica, Florin Pop, and Aniello Castiglione, "On the rewards of Self Adaptive IoT Honeypots.", Annals of Telecommunications ISSN 0003-4347. **Acceptat (IF: 1,168, SRI: 0,375)**;
7. Roxana Gabriela Stan, Catalin Negru, Florin Pop, "CloudWave: Content Gathering Network with Flying CloudsFuture Generation Computer Systems.", ISSN: 0167-739X, 2019. **În evaluare (IF: 4,639, SRI: 2,113)**;
8. Raluca Oncioiu; Florin POP, "Energy-efficient Virtual Machine Replication in Fault Toler-ant Datacenters Simulation Modelling Practice and Theory.", ISSN: 1569-190X, 2019. **În evaluare (IF: 2,092, SRI: 0,927)**.

Articole în volume de conferințe internaționale

1. Dorinel, Filip Ion, Ghita Bogdan, Pop Florin, Iordache George-Valentin, Negru Catalin, and Dobre Ciprian. "EdgeMQ: Towards a Message Queuing Processing System for Cloud-Edge Computing:(Use Cases on Water and Forest Monitoring)." In 2018 17th International Symposium on Parallel and Distributed Computing (ISPDC), pp. 46-52. IEEE, 2018.
2. Rogojanu, Tudor, Mihai Ghita, Valeriu Stanciu, Radu-Ioan Ciobanu, Radu-Corneliu Marin, Florin Pop, and Ciprian Dobre. "NETIoT: A Versatile IoT Platform Integrating Sensors and Applications." In 2018 Global Internet of Things Summit (GIoTS), pp. 1-6. IEEE, 2018.
3. Dragos Comaneci and Ciprian Dobre, "Securing Networks using SDN and Machine Learning.", 21st IEEE International Conference on Computational Science and Engineering, Bucharest Romania, 29-31 October 2018, 43-49

6.2 Sisteme eterogene Coud-Edge

Ideea din spatele paradigmei Cloud-Edge Computing este de a muta datele mai aproape de utilizator și, în același timp, să stocheze numai date importante. Cu noduri suficient de puternice pe niveluri intermediare, este posibilă procesarea datelor la fiecare nivel și înlăturarea tuturor datelor inutile. S-a măsurat impactul unei abordări simple a paradigmei Cloud-Edge cu 4 niveluri în contextul orașelor inteligente și a numărului mare de senzori. Dat fiind faptul că senzorii generează date mici la o rată foarte ridicată, au reușit să reducă cantitatea de date stocate de 20 de ori folosind ideile din spatele lui Cloud-Edge. Toate avantajele sunt din punctul de vedere în care timpul de răspuns. Dispozitivele finale pot necesita date care pot fi difuzate direct de primul layer de dispozitive de margine și dacă nodul nu are datele necesare, întreabă următorul nod pe calea ierarhică la care este conectat. Datele trimise de dispozitivele terminale pot fi citite la fiecare nivel intermediar de noduri, până la nodurile

Cloud, alături de o actualizare locală pe datele curente existente. Această abordare poate fi îmbunătățită și prin distribuirea datelor către dispozitivele utilizatorilor finali, obținându-se o distribuție a datelor cu granulație fină care conduce la un rol suplimentar al primului nivel de dispozitive de margine. Acum, aceste noduri pot direcționa cererile de date către alte dispozitive ale utilizatorilor, transferând încărcarea de la marginea la dispozitivele utilizatorilor capabili de astfel de operațiuni, cum ar fi smartphone-urile, tabletele și ceasurile.

Cloud Robotics este un domeniu de cercetare care este de mare interes în ziua de azi atât pentru mediile academice, cât și pentru cele industriale. Pe măsură ce creșterea IoT devine exponențială, tehnologiile noi și puternice sunt integrate în Cloud pentru a sprijini un număr diferit de roboți și dispozitive din nivelul Edge.

6.2.1 Managementul și prelucrarea datelor în sistemele Cloud-Edge

Preocuparea actuală în sistemele Cloud-Edge este depășirea datelor, deoarece numărul de dispozitive IoT crește exponențial și toate au o mulțime de date brute, însumând un număr imens de date neordonate. Vorbim despre o arhitectură care separă Edge de Fog și de Cloud. Stratul Edge constă în preluarea datelor brute de la senzori sau mecanismul pre-învățat. Apoi, nivelul Fog este de fapt podul dintre Edge și Cloud, deci cu un flux de date care vin la el, acesta poate preprocesa, filtra și modifica datele de-a lungul drumului. Nivelul Cloud este punctul fix de copiere și stocare din arhitectură. Aceste comunicații nu știu cum să schimbe informații una cu cealaltă, astfel că au nevoie de un Message Broker, care poate gestiona mesajele provenind din orice flux care merge la orice nivel. Primul lucru pe care Message Broker îl face este să separe comunicarea dintre nivelul Edge și nivelul Fog, astfel încât serviciile numite sunt perfect separabile. Ei primesc și gestionează fluxuri de mesaje la perioade predefinite de timp. Pot fi de zece secunde, trei ore, două săptămâni etc. A doua abilitate a Message Broker este de a separa mesajele în colecții importante de date și de a le direcționa de la nivelul Fog spre Cloud. Al treilea lucru, care este de remarcaT este faptul că Message Broker permite ca mesajele să circule în interiorul unui nivel specific, permițând astfel migrarea eterogenă a datelor în interiorul nivelului Edge, precum și migrarea semnificativă a capsulelor de date în nivelul Fog.

Heterogeneous data management. Când vorbim despre date eterogene, vorbim despre un număr nedefinit de surse, despre date care nu sunt bine structurate, despre date care nu sunt structurate deloc, despre date cu mai multe dimensiuni brute și, de obicei, despre cantități uriașe de date. Spre deosebire de datele structurate și omogene, trebuie să luăm în considerare modul de organizare a unei stocări mari a datelor, gestionarea acesteia, latența manipulării datelor offline sau în situații în timp real și, desigur, comunicarea.

Edge. Potrivit companiei de cercetare IDC, "Edge computing este o rețea de plase de micro date care procesează sau stochează date critice la nivel local și împinge toate datele primite către un centru de date sau un depozit de stocare a cloud-ului, într-o amprentă mai mică de 100 de metri pătrați".

Nivelul Edge, constă în roboți care colectează date utilizând senzori. Datele sunt trimise către un gateway și sunt procesate. Roboții pot colecta date în mai multe formate sau tipuri. Unele dintre cele mai utilizate tipuri sunt datele senzorilor, datele GIS, multimedia precum imagini, audio sau video, datele generate de mașini, cum ar fi jurnalele, interacțiunile sociale online și documentele. Este posibil ca roboții să coopereze cu alți roboți pentru a rezolva sarcini mai exigente sau pentru a descărca sarcinile altor roboți.

Cloud. Sistemele Cloud au mai multe componente, cum ar fi un șir de procesare a mesajelor, o componentă de microservicii scalabilă și componentele superioare de stocare și procesare. Componenta de prelucrare a șirului de mesaje poate fi un broker care este responsabil pentru gestionarea șirurilor multiple pentru diferite aplicații și tipuri de date.

În ultimii ani, numeroase grupuri de cercetare au publicat articole în care au propus diferite arhitecturi pentru Cloud și Cloud Robotics. De cele mai multe ori, un cloud ad-hoc este construit dintr-o infrastructură Cloud și o rețea de dispozitive. O astfel de combinație ne oferă o mare flexibilitate în alegerile pe care le facem atunci când proiectăm un Cloud pentru aplicații specifice. Există mulți

factori atunci când alegem un model de calcul flexibil, iar unii sunt condițiile de rețea, cum ar fi lățimea de bandă, cerințele aplicațiilor cum ar fi consumul redus de energie, disponibilitatea mare a resurselor în sistemele distribuite.

6.2.2 Microservicii și containere

În ultimii ani, VMs și containerele au făcut subiectul discuțiilor pentru multe lucrări. Reamintim principala diferență dintre aceste două concepte și explicăm de ce acest lucru este încă un subiect nou pentru domeniul IoT. Un sistem server fizic cu tehnologie de virtualizare care rulează în mașinile virtuale rulează de fapt un sistem de operare complet precum și toate aplicațiile corespunzătoare. Este necesar un hypervisor pentru a orchestra VMs și pentru a asigura o izolare între ele. Ca o abordare diferită a aplicațiilor containerizate cu Docker (o soluție bazată pe LXC) necesită doar un singur sistem de operare, sistemul de tip server, deoarece containerul distribuie kernel-ul gazdei în modul read-only și are anumite puncte de montare specifice pentru accesul la scriere. Datorită acestei diferențe, recipientele sunt mult mai ușoare și folosesc resurse mult mai puține decât mașinile virtuale care le fac mai fiabile pentru sistemele încorporate cu cerințe stricte de resurse în care fiecare proces este optimizat pentru a asigura o utilizare adecvată a energiei și pentru a prelungi durata de viață și fiabilitatea dispozitivelor detașate (ex., smart dust).

Multe lucrări propun arhitecturi bazate pe containere care încearcă să rezolve complet nevoile IoT. Sistemul CoESMS10 "beneficiază de avantajele platformei de calcul IoT și Edge pentru pre-lucrarea datelor în timp real, beneficiază de avantajele populației CoaaS performanța și scalabili-tatea dispozitivului de calcul pentru Fog ". Această arhitectură propusă are un sistem pe cinci nivele, bazat pe consumatori, furnizori și, cu un program CoaaS (Container as a Service) pentru planificarea sarcinilor eficiente din punct de vedere energetic la Edge. În timpul tranziției în containerizare, IoT se mută la microservicii din cauza livrării rapide, a scalării și a izolației oferite de containere. Un exemplu tipic al unui astfel de sistem poate avea microservicii în Cloud folosind tehnologii precum Kuber-netes Docker și Jenkins. Un caz special este containerizarea Docker peste RaspberryPI datorită specificațiilor sale hardware care permit rularea a sute de containere într-un singur sistem.

6.2.3 Platforme de procesare

Fiecare platformă Cloud trebuie să aibă o componentă de procesare care cuprinde mai multe cadre de procesare distribuite, specializate, utilizate pentru dezvoltarea algoritmilor. Credem că un Cloud ar trebui să aibă platforme puternice de procesare pentru sarcinile clasice MapReduce și algoritmi de învățare mecanică. O implementare cu Apache Hadoop poate finaliza cu succes sarcinile MapReduce. Pentru a avea un management mai bun al resurselor, îl putem implementa împreună cu HDFS și YARN de la Hadoop. Implementarea Apache Spark poate gestiona analizele de date și acoperă prelucrarea de învățare mecanică. Descriem aceste platforme pentru o mai bună înțelegere a contextului Cloud Computing.

Platforma cea mai utilizată pentru sarcinile MapReduce este Apache Hadoop cu sursă deschisă. Are mai multe niveluri grupate într-un teanc, în care fiecare nivel are toleranță la erori. Apache Hadoop este alcătuit din HDFS11 și Hadoop YARN12 grupate într-o platformă agregată. Apache Spark oferă o interfață convenabilă de programare integrată în limbaj în Scala. Spark API poate folosi o strategie înceată deoarece calculează RDD-urile pentru prima dată când sunt folosite în acțiuni. Un fapt important este că Apache Sparks păstrează toate RDD-urile în memorie, dar poate folosi și discul dacă este necesar.

Apache Storm este un sistem de calcul distribuit în timp real, gratuit și cu sursă deschisă, utilizat pentru prelucrarea datelor scalabile, rapide și mari, într-un cluster. Storm reprezintă o alegere bună pentru multe proiecte în care sunt necesare date care nu sunt obligatorii în timp real datorită atributelor sale: foarte ușor de construit implementat și utilizat; procesarea de mare viteză ajungând la milioane de octeți pe secundă pe nod; prin urmărirea tuturor lucrătorilor, Storm este foarte tolerant la erori; fiabilitate; scalabilitate într-un grup de mașini de până la zeci de mii de noduri.

Un alt cadru open source este Apache Flink pentru procesarea distribuită, de înaltă performanță, întotdeauna disponibilă și precisă a fluxului de date. Flink oferă o execuție care este foarte flexibilă și fiabilă. Sunt aplicate garanții stricte de o singură prelucrare pentru consistență. Toleranța la erori se realizează prin punct de control sau reevaluare parțială, deoarece sursele de date sunt presupuse a fi persistente și repliabile.

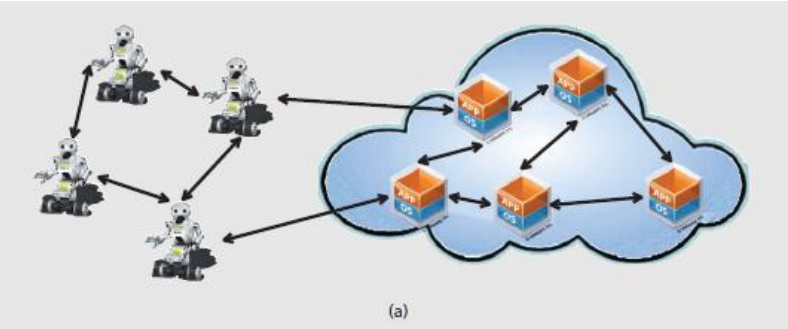
Mulțimea de cadre Apache este completată de Apache Mesos o platformă care servește pentru partajarea resurselor în Centrele de date. Mesos face abstracții asupra procesorului, a memoriei și a spațiului de stocare, permițând dezvoltatorilor să implementeze sisteme distribuite cu toleranță la defecte și elastice. Sistemele pot fi ușor configurate, desfășurate și întreținute. Folosește principii similare cu kernel-ul Linux, dar este un cadru cross platform disponibil pe Linux, OSX și Windows care au suport de bibliotecă pentru C ++, Java și Python.

6.2.4 Aplicații robotice

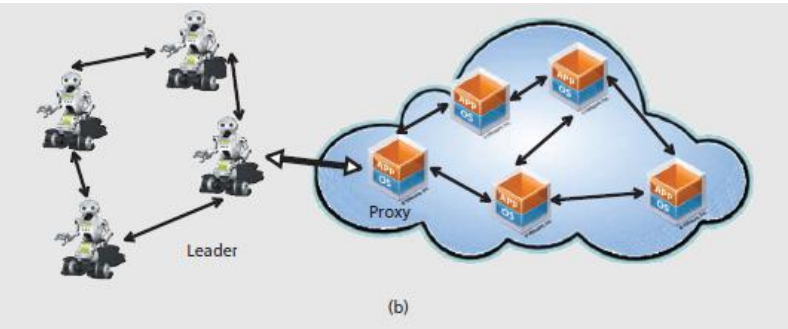
Roboții beneficiază de Cloud Robotics și, uneori, roboții tradiționali în rețea sunt înlocuiți cu un nou tip de rețea bazată pe Cloud. Unele avantaje față de roboții tradiționali în rețea sunt:

- work offload - capacitatea de a descărca sarcinile intensive mari către Cloud;
- big data - accesul la datele mari și la platformele de procesare;
- unlimited knowledge - flexibilitate pentru a dezvolta noi abilități bazate pe cunoștințele stocate în Cloud.

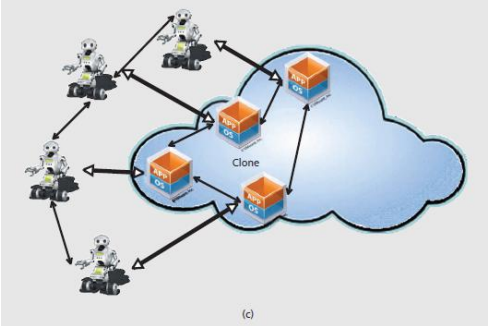
Descărcarea aplicației în Cloud înseamnă că roboții pot păstra acum doar datele necesare și pot lucra pe dispozitivele locale (de exemplu, colectarea datelor de la senzori și prelucrarea în timp real pentru a face un răspuns atunci când interacționează cu mediul). O altă consecință este că roboții sunt mai ușori, mai puțin costisitori, iar menținerea software-ului și hardware-ului este mai ușoară și mai ieftină, ceea ce înseamnă că dezvoltatorii pot face independent upgrade-uri la roboți și Cloud. Deoarece roboții pot accesa acum informații și cunoștințe utilizând Cloud-ul, ei pot să evolueze și să facă sarcini noi sau complexe cu Big Data fără să se ocupe de menținerea acestor date. Cloud-ul oferă capacitatea unui robot de a învăța noi abilități de la alți roboți sau dispozitive. Cloud-ul funcționează ca o bibliotecă de competențe și poate fi un instrument puternic pentru a gestiona comunicarea și colaborarea dintre roboți.



(a) Peer-Based architecture (a) Fiecare robot sau dispozitiv care este conectat la Cloud este considerat o unitate de calcul conectată într-o plasă de calcul complet distribuită.



(b) Proxy-Based architecture (b) O unitate este aleasă ca lider de grup; numai liderul poate comunica cu un proxy VM care este găzduit în infrastructura Cloud.



(c) Proxy-Based architecture (c) Fiecare robot are o clonă corespunzătoare la nivel de sistem în Cloud, care permite posibilitatea executării unei sarcini pe robot sau pe clona lui; atât rețeaua de roboți, cât și rețeaua de clone sunt rețele peer-to-peer care îmbunătățesc conectivitatea.

Figura 6.1. Arhitecturi flexibile pentru Cloud Robotics [Hu et. Al. 2012]

Având un astfel de mediu pentru roboți, toate dispozitivele IoT pot să împărtășească informații și să învețe noi abilități. Roboții pot găsi în Cloud o bază de date sau o bibliotecă de abilități care să corespundă cerințelor de activitate diferite și complexității mediului. Un proiect care lucrează la o implementare similară este proiectul RoboEarth [Waibel et.al. 2011]. Unii dintre cercetătorii de la acest proiect au fondat Rapyuta Robotics care este o sursă deschisă Platform-as-a-Service (PaaS), un cadru utilizat în aplicațiile de robotică [Hunziker et.al. 2013].

În zilele noastre, există o foarte mare diversitate de tipuri de roboți, fiecare servind mai multor scopuri. Datorită dezvoltării tehnologiei roboții pot colabora între ei sau pot solicita informații suplimentare din bazele de date existente în mediile Cloud [Aazam et.al. 2016]. Astfel de sisteme de roboți au la bază senzori, prin intermediul cărora sunt preluate date, ca apoi să fie analizate, re-spectiv sunt extrase informații de nivel înalt pe baza cărora se construiesc sisteme de reguli [Kehoe et.al. 2015]. Aceste sisteme de prelucrare și procesare a datelor (de obicei de mari dimensiuni, sau generate în fluxuri de viteză crescută - exemplu o cameră montată pe o mașină autonomă care captează mai multe cadre pe secundă) necesită capacități de procesare și stocare mărite, motiv pentru care multe companii preferă folosirea, acolo unde este posibil, a unui mediu Cloud suport. Datele achiziționate prin senzori sunt transferate prin diverse tehnologii de comunicații (cu fir sau fără fir) către Cloud, unde se realizează procesarea și extragerea de informații, ce sunt transmise către mecanismele de control inteligent [Ermacora et.al. 2016].

Pentru a putea valida conceptul propus, echipa de cercetare a proiectat o serie de dispozitive/sisteme IoT și roboți pentru a putea genera diferite modele de date primare. Acestea vor fi colectate, stocate și procesate în cadrul arhitecturii Cloud-Edge propuse.

Pentru o **prezentare completă** a analizei modelelor de interconectare a sistemelor eterogene Cloud-Edge si a modelelor de colectare și procesare a datelor în timp real cu constrângeri de calitate, **a se vedea raportul in extenso al proiectului ROBIN-Cloud** de pe site-ul proiectului.

6.3 Specificații funcționale și arhitecturale pentru componentele ROBIN-Cloud

6.3.1 Roboții: o perspectivă generală

Definim R ca set de roboți care modelează N agenți heterogeni cu caracteristici diferite, cum ar fi senzori, putere de procesare, entități periferice care pot achiziționa date, unde N = |R|. Un robot r are următoarea descriere oficială:

$$i = (id, S_i, C_i), \qquad (6.1)$$

unde:

- *id* este un număr care identifică în mod unic un robot;
- S_i este setul de senzori cu care robotul $i \in R$ este echipat;
- $C_i = (C_{i1}, \ldots, C_{iN})$ este *resource capacity vector* a robotului $i \in R$, where C_{ir} este suma resurselor *r* disponibile pe robot cu $C_{ir} > 0$.

6.3.2 Data Capsule: o reprezentare generică și flexibilă a datelor nestructurate în Cloud-Edge

Gestionarea fluxului de date este una dintre cele mai importante caracteristici ale arhitecturii pro-puse. În această secțiune vom descrie un format structurat pentru reprezentarea și stocarea datelor, numit Data Capsule. Definim acest format în modul de obținere a unei specificații reprezentative generice pentru o unitate de date într-o bază de date înseriată în timp cu date nestructurate. Definim un set de data capsules după cum urmează:

$$D = \{D_i \mid D_i = (t_i, c_i, m_i, d_i)\} \tag{6.2}$$

unde:

- t_i este time-stamp pentru actuala data capsule;
- c_i este o mulțime comandată de șiruri de caractere (subcontexte) care definește contextul data capsule;
- m_i esteun set de (*key*, *value*) tuple de cheie și valori comparabile care definesc meta-date ale data capsule;
- d_i este conținutul de date pe care dorim să îl stocm în interiorul data capsule.

Am definit o data capsule astfel încât să permită obținerea de data capsules pe baza potrivirii contextului, a intervalelor de timp și a meta-datelor. Să definim un operator care selectează data capsules într-un context. Considerăm că operatorul va fi definit astfel încât să țină cont de un subcontext special care se potrivește cu orice subcontext existent la un anumit nivel de ierarhie. Permiteți operatorului:

$$\begin{aligned} SearchContext(D_{seT}\, c_{match}) = \{D_i \in D_{set} \mid \forall s_j \in c_i \\ \text{and } \forall s_k \in c_{match} \text{ st. } (j = k) \text{ and } (s_j = s_k \text{ or } s_k = *)\} \end{aligned} \tag{6.3}$$

unde: Dset este setul de data capsules pe care îl căutăm; cmatch este un context care ar trebui să se potrivească și poate conține unele wild-card (∗) subcontexte; ci este contextul Di; ∗ este un context special care se potrivește oricărui subcontext.

Definim un operator care extrage anumite certain data capsules prin meta-date:

$$\begin{aligned} SearchM\,eta(D_{seT}\, m_{match}) = \{D_i \in D_{set} \mid \forall (key_j, value_j) \in m_i \\ \exists\ (key_k, value_k) \in m_{match} \text{ st. } key_j = key_k \\ \text{and } value_j = value_k\} \end{aligned} \tag{6.4}$$

unde Dset este setul de data capsules pe care noi îl căutăm; mmatch este un set de valori meta care ar trebui să fie potrvite; mi este meta-data lui Di. Deoarece vrem să descriem o serie de date temporale, este necesar un operator care selectează toate data capsules pentru un context într-un interval de timp specificat.

6.3.3 Platformă scalabilă bazată pe microservicii pentru procesarea datelor în sistemele Cloud-Edge.

Prezentăm arhitectura de bază a proiectului ROBIN-Cloud în Figura 6.2. Platforma noastră integrează o gamă largă de dispozitive care pot utiliza ROBIN-Cloud ca platformă pentru stocarea și prelucrarea

Cloud. Aceste dispozitive se numesc surse de date și pot produce date în diverse formate cum ar fi: datele senzorilor, datele de sistem de informații geografice (date GIS), fișierele multimedia (audio, imagini, video de la dispozitivele încorporate)), interacțiuni sociale online (roboți sociali) sau documente.

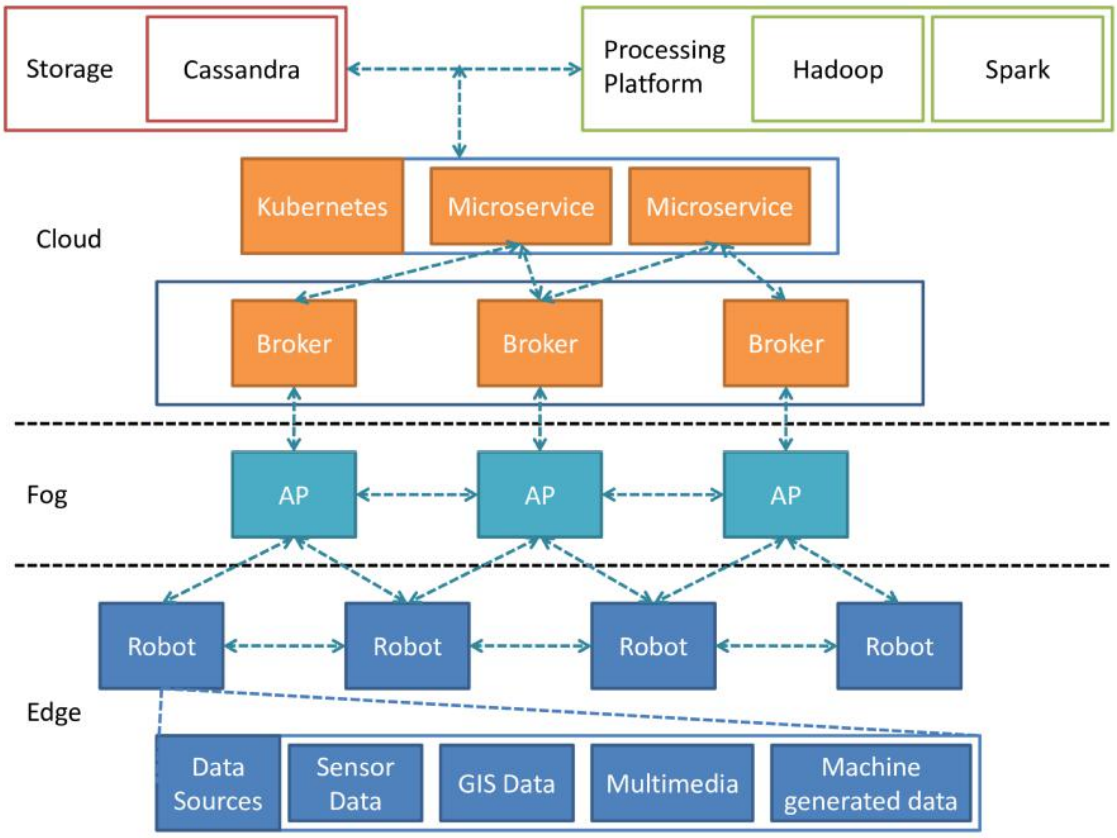


Figura 6.2. Arhitectura Robin-Cloud

Cloud-ul principal este ROBIN-Cloud care reunește datele din sursele de date, le procesează și face o Data Capsule (așa cum a fost prezentat: formatul de stocare internă a datelor în proiectul ROBIN-Cloud, abstractizarea datelor având 4 componente: locație, timestamp, metadate și date) pentru fiecare interogare primită. Data Capsule este stocată într-o bază de date care ar trebui să fie eficientă pentru interogările în serii de timp. Data Capsule este stocată, neschimbată, pentru totdeauna. O interogare funcționează numai cu copii ale Data Capsules din baza de date.

Nivelul Edge este format din roboți care colectează date din mediul înconjurător și îl for-matează în data capsules care sunt trimise către puncte de acces bazate pe context. Datele colectate de către roboți pot fi unele dintre următoarele: datele senzorilor, datele GIS multimedia, datele generate de mașină, interacțiunile sociale online și documentele. În acest model, este posibil ca roboții să coopereze între ei pentru a rezolva sarcini mai complexe sau pentru a descărca sarcinile altor roboți. Nivelul Edge constă în roboți (dispozitive) care colectează, analizează și preprocesează date din mediul înconjurător și îl formatează în Data Capsules care sunt trimise la APs (puncte de acces) pe baza contextului lor.

Cooperarea cu roboți este considerată posibilă, astfel încât ei să poată vorbi unul cu celălalt în scopul rezolvării unor sarcini mai complexe și, de asemenea, să înlăture anumite locuri de muncă ale altor entități, dispozitive sau roboți. Acesta este modelul descris în această subsecțiune. În acest fel, unele dintre dispozitivele situate la marginea rețelei pot comunica între ele, dar acest lucru nu este întotdeauna adevărat. Unele puncte de acces pot fi atât de îndepărtate unul de celălalT fie sarcina poate fi atât de greu de procesat încât singura soluție disponibilă este de a trimite datele în sus în Fog spre Cloud. Fiind un sistem distribuiT acest lucru permite împărțirea datelor în bucăți și reconstruirea lor înapoi spre roboți.

Nivelul Fog constă din agenți care sunt puncte de acces pentru roboți. Punctele de acces sunt responsabile pentru a primi date de la roboți și pentru a le transmite Cloud-ului. După ce datele sau sarcina au fost tratate în Cloud, punctele de acces trebuie să trimită răspunsul robotului care a trimis cererea inițială. Dacă este necesar, punctele de acces pot rula procesarea la nivel local, deoarece sunt configurate cu o implementare Kubernetes și pot rula microservici simple. În mod alternativ, punctele de acces pot transmite datele către alte puncte de acces deoarece acestea sunt conectate într-o rețea peer-to-peer. Dispozitivele de calcul pentru Fog sunt situate între punctele finale și Cloud-ul tradițional, astfel resursele și serviciile sunt disponibile și sunt mai aproape de utilizatorii finali, iar întârzierile induse de implementarea serviciilor pot fi reduse.

Nivelul Cloud constă din mai multe componente: o componentă de șir de procesare a mesajelor care stochează date și sarcini primite de la punctele de acces o componentă microser-vicii auto-scalabilă care gestionează interacțiunea dintre componenta de procesare a mesajelor de șir și componentele superioare de stocare și procesare. Datele primite de la roboți vor fi stocate în componenta de stocare care este o bază de date distribuită și vor fi gestionate de platforma de procesare care va analiza datele folosind alți algoritmi.

Propunem o soluție SaaS deoarece ar corespunde nevoilor noastre, având un mediu deja inițializaT astfel încât să putem implementa orice număr de roboți. Componenta message queue processing va consta dintr-un broker care va fi responsabil pentru gestionarea șirurilor multiple pentru diferite aplicații și tipuri de date. RabbitMQ este un message broker, open source, care acceptă mesageria asincronă cu mai multe protocoale, așteptarea mesajelor, confirmarea livrării, rutarea flexibilă la șiruri și tipul de schimb multiplu. Componenta auto-scalable microservices va consta într-o implementare Kubernetes care va gestiona microservicii care vor funcționa în containere. Kubernetes este un sistem open-source pentru automatizarea implementării, scalării și gestionării aplicațiilor containerizate. Acesta gru-pează containerele care alcătuiesc o aplicație în unități logice pentru o gestionare și descoperire ușoară numite păstăi și fiecare păstaie va rula mai multe microservicii ca containere. Microser-viciile pot avea diferite roluri care pot varia de la interacțiunea cu șirurile de mesaje, descoperirea serviciilor sau lansarea de locuri de muncă pe platforma de procesare.

Componenta storage va consta în Cassandra, un sistem distribuit de gestionare a bazelor de date NoSQL conceput pentru a gestiona cantități mari de date pe mai multe mașini, oferind în același timp scalabilitate și disponibilitate ridicată. Se bazează pe o arhitectură descentralizată fără niciun punct central de eșec și asigură toleranța la erori prin replicarea datelor la mai multe noduri. Fiecare intrare din baza de date va respecta următorul format de date:

DataFormat = (id, robotid, time, location, contexT data) (3.6)

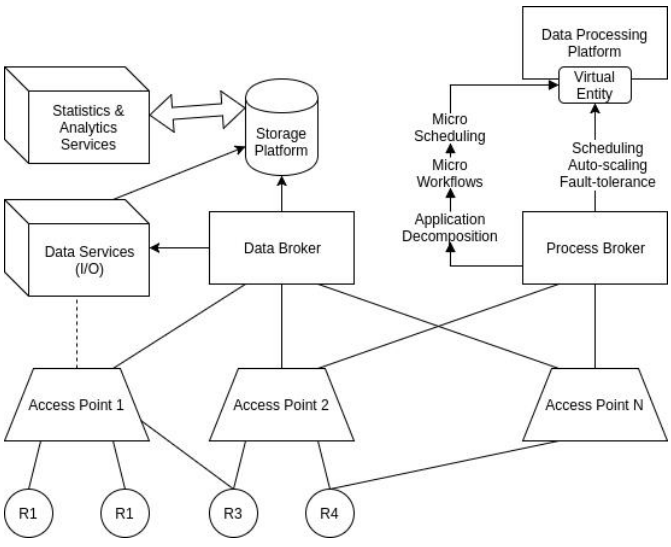


Figura 6.3 Perspectiva asupra arhitecturii ROBIN-Cloud

Platforma de procesare va consta din mai multe cadre de procesare distribuite, special-izate, care vor avea implementați algoritmi multipli. Pentru procesarea clasică MapReduce și pentru un sistem de fișiere distribuit poate fi implementat Apache Hadoop. Pentru algoritmi specializați în învățarea mecanică și analiza datelor, Apache Spark poate fi implementat și poate fi utilizat împreună cu HDFS și YARN de la Hadoop. Pentru a avea o mai bună gestionare a resurselor.

Apache Mesos este un nucleu de sistem distribuit care abstractizează resursele de calcul cum ar fi CPU, memorie, stocare de la mașini pentru a asigura partajarea resurselor între mai multe platforme de calcul. Acesta asigură o bună scalabilitate, disponibilitate ridicată și toleranță la erori. Apache Mesos vine cu o serie de mulțimi de cadre și aplicații care pot profita de capacitățile sale de partajare a resurselor și, prin utilizarea lui Mesos este posibilă multiplexarea mai multor cadre de procesare distribuite și obținerea unei scalabilități mai mari.

Perspectiva generală a conceptului ROBIN-Cloud este prezentată în figura 6.3. Având în vedere specificația principală, am evidențiat componentele Data Broker și Process Broker. Mai mult fluxul de procesare este inclus.

6.3.4 Colectarea și precizia datelor

Dat fiind faptul că datele brute de astăzi pot fi compuse din zgomot date generate de mașini, GIS (Geographic information system) date (latitudine, longitudine), date multimedia (sunete relevante, imagini), datele senzorilor (gradul de poluare, valorile date de un accelerometru), definim trei tipuri de colectare a datelor: Local data; Edge data; Cloud data.

Pe baza puterii de procesare a robotului și a cantităților sale disponibile de resurse, precum și a utilității datelor de citit păstrăm datele pe care le folosim pentru procesele locale (de exemplu, datele utilizate pentru deciziile în mișcare ale robotului) pe dispozitiv și trimitem altceva, fie ea inutilă sau nu, la o poartă care ar distribui datele la nodurile Edge. Dar pentru a decide dacă datele sunt relevante sau nu, implică utilizarea unui DPS (Data Processing System) cu funcția predominantă de preprocesare a datelor colectate în sistemele bazate în timp real.

Data preprocessing este împărțită în mai multe etape, după cum urmează: Curățarea datelor; Integrarea datelor; Transformarea datelor; Micșorarea datelor.

Data cleaning reprezintă procesul de eliminare a zgomotului și a erorilor greșite sau a datelor care se defectează de la intrarea noastră. Datele colectate de un robot pot fi zgomotoase, incon-sistente sau incomplete. Datele zgomotoase reprezintă orice colecție care conține valori similare celor vizate sau cele așteptate sau date eronate. Datele contradictorii sunt orice colecție care nu respectă un set specific de reguli, stabilite în prealabil, sau date care conțin diferențe în modul în care sunt etichetate. Datele incomplete sunt orice colecție care nu are chei, valori, atribute de interes.

Noise este de obicei înțeleasă ca instabilitatea datelor. Unele dintre cauzele sale pot fi: zgomot aleatoriu, care este inevitabil într-un sistem non-perfect; valori care nu aparțin setului de date; erori matematice, umane sau mecanice.

Unele dintre modalitățile de eliminare a zgomotului sunt:

- machine learning: "După cum sugerează și numele, regresia liniară rezolvă o problemă de regresie. Cu alte cuvinte, scopul este de a construi un sistem care să ia un vector $x \in Rn$ ca intrare și să prezică valoarea unui scalar $y \in R$ ca ieșire." [Goodfellow et. al. 2016]; pentru cazul nostru, avem o mulțime de exemple cu date zgomotoase și, mai ales nezgomotoase, și dorim să potrivim o linie între toate aceste puncte și o considerăm linia de valoare vizată;
- "O tehnică de grupare spectrală pentru date zgomotoase. Ideea noastră principală a fost de a descompune graficul de similitudini în doi factori latenți: corupțiile rare și datele curate. Am aflat în comun încorporarea spectrală, precum și datele corupte. Am propus trei soluții algoritmice diferite folosind Laplaciani diferiți. Experimentele noastre au arătat că încorporarea învățată accentuează în mod clar structura de grupare și că metoda noastră depășește clustering-ul spectral și concurenții de ultimă generație [Bojchevski et.al. 2017]";

- binning: având în vedere apropierea fiecărui exemplu de zgomot divizăm valorile în buckets (bins); de exemplu, if data = 3.4 ∧ (3 ≤ x < 4 → x ∈ buckets[3]) → data ∈ buckets [Bellavista& Zanni 2017]

Data integration reprezintă procesul de concatenare a probelor care se potrivesc împreună, reducând astfel numărul de eșantioane utilizate și, de asemenea, utilizând date mai specifice.

Data transformation este modul în care păstrăm datele noastre într-un prag necesar, făcându-l scalabil.

Data reduction înseamnă comprimarea datelor într-un mod ușor de citit simplificat.

Toți pașii prezentați mai sus au ca obiectiv principal analizarea și ordonarea datelor brute spre o calitate superioară, astfel încât să poată fi utilizate în fazele de instruire a Machine learning sau KDD (Knowledge Discovery in Databases). Mesajele de zgomoT inconsistente sau necorespun-zătoare (de exemplu: human: yes hasTail: yes }) pot fi eliminate în etapa de preprocesare, făcând astfel colectarea de date și minarea mai eficiente.

6.4 My Cloudy Time Machine

6.4.1 Cloud Robotics pentru ROBIN

Folosind proiectul ROBIN-Cloud oferim o platformă unificată pentru Cloud Robotics care inte-grează toate tipurile de dispozitive conectate la orice aplicație care poate fi accesată prin Cloud.

Specificațiile aplicației pot diferi de la caz la caz, însă principalele caracteristici, cum ar fi fiabili-tatea, scalabilitatea, performanțele ridicate, consumul redus de energie, rețelele cu latență redusă, ar trebui să fie similare pentru toate aplicațiile care utilizează biblioteca API expusă de Cloud. Ca proiect inițial dorim să facem o dovadă a conceptului de testare a ROBIN-Cloud și să arătăm cum să integrăm o aplicație în această arhitectură.

My Cloudy Time Machine este o aplicație pentru un robot asistiv conectat la ROBIN-Cloud. Specificațiile aplicației sunt relativ complexe și ar trebui să conțină detalii hardware și software referitoare la construirea sau planificarea robotului. Acesta oferă o mai bună înțelegere a sistemului local și informații despre software-ul care preprocesează datele la nivel local și îl împinge în Cloud unde agenții procesează datele într-un format final și le stochează.

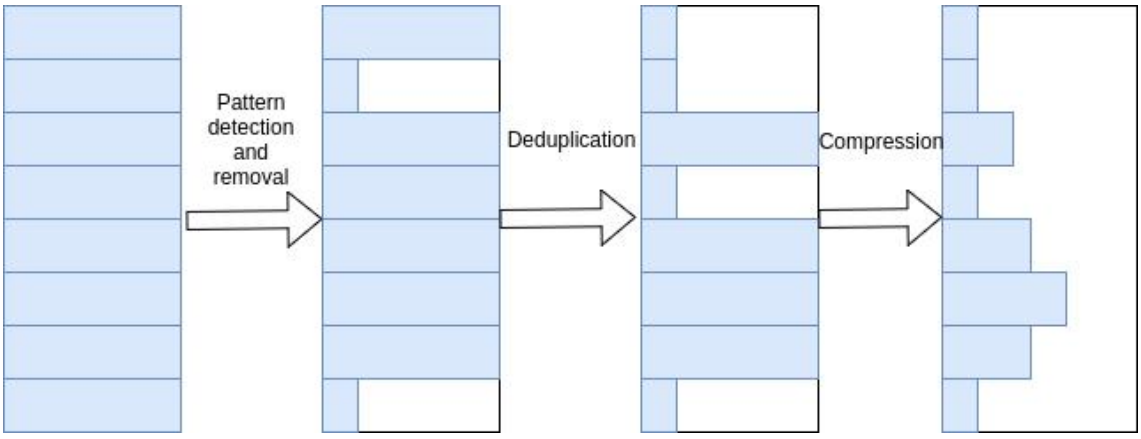


Figure 6.4. Data reduction.

Această aplicație permite utilizatorilor să facă instantanee instantanee ale mediului fizic ori de câte ori se află în raza robotului și să inspecteze înregistrările într-un moment ulterior de timp, ceea ce dă iluzia călătoriei în timp. Am ales să denumim aplicația My Cloudy Time Machine din cauza funcționalității sale principale.

Hardware-ul constă într-o mașină care are diferiți senzori cum ar fi senzorul de temperatură, senzorul ultrasonic, GPS-ul, un număr mare de dispozitive periferice utilizate pentru a face achiziția de date,

cum ar fi microfoanele sau alte dispozitive folosite pentru a interacționa cu mediul, cum ar fi difuzoarele și motoarele. Hardware-ul este folosit pentru a reproduce cât mai bine o copie a mediului și pentru a ajuta robotul să-și îmbunătățească cunoștințele.

Utilizăm robotul pentru a realiza o hartă fină de interior pentru o podea dintr-o anumită locație și pentru a înregistra evenimente în timp real. Robotul are un mod implicit de explorare în care învață mediul și ascultă evenimente audio. El interacționează cu oamenii prin dispozitivele sale, cum ar fi microfoane, difuzoare și senzori ultrasonici. Oamenii pot da comenzi audio ca "Remember this!". Când robotul recunoaște această comandă, acesta pornește Record mode unde utilizează toate dispozitivele sale pentru a realiza instantaneu un snapshot al mediului pentru o perioadă predefinită (de exemplu, 1 minut). Înregistrarea este salvată ca o Data Capsule care va fi împinsă în Cloud pentru procesare și pentru stocare permanentă.

Am ales să facem Data Capsules persistente și invariabile, ceea ce înseamnă că, odată ce stocăm o capsulă în Cloud, nu o putem schimba. Orice interogare sau operație de actualizare poate citi sau copia Data Capsule, fără a permite eventuale modificări. Această abordare este utilizată pentru a menține istoricul Data Capsules și pentru a face interogări de serie de timp pentru o perioadă de timp sau pentru o locație utilizând un API disponibil pentru toate sistemele conectate la ROBIN-Cloud.

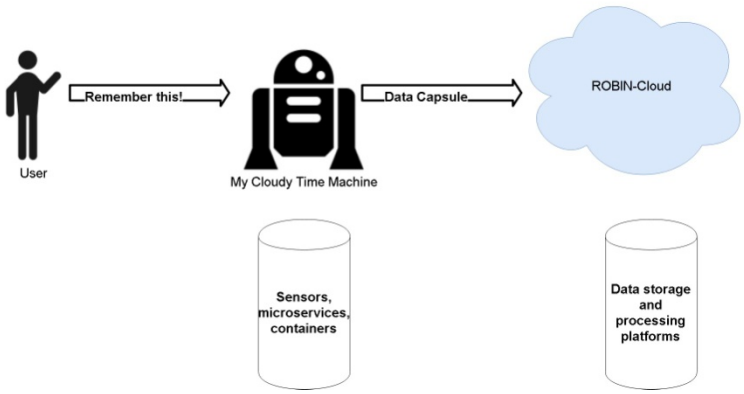


Figura 6.5. My Cloudy Time Machine – flux general de lucru.

În ceea ce privește experiența utilizatorului oferim o interfață grafică în care utilizatorul își poate vedea istoricul de Data Capsules sau poate căuta public sau shared3 Date Capsule ca o listă. Fiecare intrare poate fi extinsă și utilizatorul poate vedea toate detaliile despre o anumită capsulă, având o modalitate ușoară de a exporta datele dorite într-un format local sau de a le distribui prin e-mail, furnizorilor de servicii de stocare în Cloud sau de social media.

Această platformă poate fi reunită în cele din urmă cu alte platforme pentru a extinde funcționalitatea și utilizarea. Este o dovadă a conceptului pentru proiectul ROBIN-Cloud în acest momenT dar poate continua ca un proiect separaT deoarece are deja caracteristici integrate, cum ar fi procesarea imaginilor sau video, folosind agenți care aplică algoritmi de învățare mecanică în Cloud.

6.4.2 Arhitectură auto-scalabilă pentru platforma propusă

Arhitectura noastră auto-scalabilă este capabilă să colecteze date dintr-un număr nelimitat de dispozitive, să o proceseze ca date de capsulă și să le împingă în Cloud. În Figura 3.12 prezentăm o arhitectură pe mai multe niveluri, care oferă o scalabilitate ridicată, o transmitere sigură a datelor și o performanță ridicată datorită designului său. Această arhitectură are 4 niveluri care sunt interconectate după cum urmează:

- **Hardware Layer** - nivelul scăzut în care se găsesc dispozitive care pot produce date (senzori) sau poate primi comenzi (motoare); funcția principală a acestui nivel este colectarea datelor și interacțiunea directă cu mediul

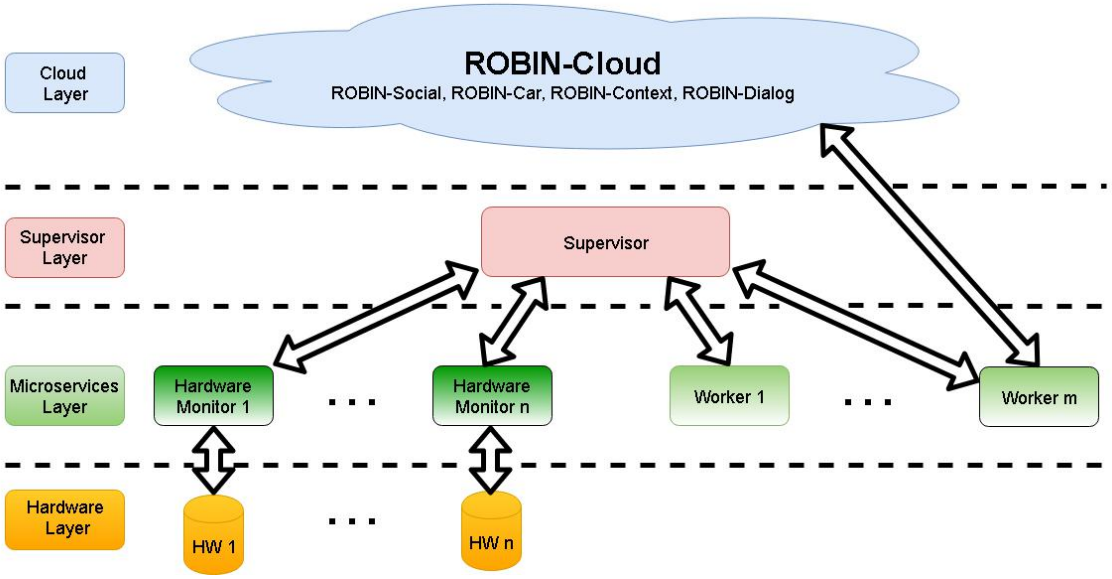


Figura 6.6. Arhitectura My Cloudy Time Machine.

- **Microservices Layer** – nivelul din partea superioară a hardware-ului în care o entitate este un microservice care poate comunica cu hardware-ul, Cloud-ul sau Supervisor-ul;
- **Supervisor Layer** - acest nivel are o singură entitate; Supervisor este un alt microservice care este responsabil pentru gestionarea microserviciilor;
- **Cloud Layer** - acest nivel nu este pe dispozitivul local și oferă resurse sau cunoștințe externe nelimitate pentru robot.

6.4.3 Componente

Dispozitivele din Hardware Layer se numesc Hardware Components (HW) și pot fi orice tip de echipament hardware de la senzori simpli (de exemplu distanță, culoare, poluare, GPS) până la dispozitive complexe, cum ar fi webcamuri, microfoane, module Bluetooth și WiFi. Ele pot avea mai multe forme, diferite dimensiuni, cerințe specifice de consum de energie sau protocoale de comunicare diferite. Aceste specificații ar trebui să fie integrate în sistemele încorporate cât mai eficient posibil și cu scalabilitate ridicată și modificări minime ale driverelor și ale software-ului responsabil cu comunicarea cu hardware-ul. Majoritatea dispozitivelor au un API pentru biblioteci, care permite software-ului de nivel înalt să interacționeze cu ușurință cu hardware-ul într-un mod eficient și sigur.

Entitățile sau microserviciile din nivelul Microservices sunt responsabile de interacțiunea din-tre hardware și nivelurile ridicate. Am împărțit aceste microservicii în două categorii. Prima categorie are microservicii care interacționează direct cu componentele hardware și sunt numite mi-croservicii Hardware Monitor (HM) deoarece configurează și gestionează hardware-ul, colectează și preprocesează date brute de la dispozitive, traduc comenzi înalte în mașină de nivel inferior - instrucțiuni dependente care permit interacțiunea cu mediul. De obicei, aceste microservicii sunt legate de I/O deoarece își petrec cea mai mare parte a timpului comunicând cu hardware-ul.

Celălalt grup de microservicii oferă o putere de procesare ridicată, care poate face aproape orice, în timp ce resursele fizice sunt disponibile. Acestea sunt numite microservicii Worker (W) deoarece munca grea făcută pe sistemul local se află în aceste microservicii care nu interacționează direct cu hardware-ul, petrec majoritatea timpului cu procesarea datelor și acesta este motivul pentru care sunt legate de CPU. Lucrătorii procesează date de la hardware făcând eșantionare, loturi sau grupări în Data Capsules care sunt trimise în Cloud pentru stocare permanentă și prelucrare ulterioară complexă. Roboții pot primi date de la Cloud, care sunt procesate și de către lucrătorii care iau decizii bazate pe situația locală. Unii dintre lucrători pot avea sarcini administrative, cum ar fi logarea, monitorizarea resurselor, alertele de reproducere în cazul apariției unor defecțiuni, curățarea depozitului local.

Serviciul special sau microservice care răspunde de întregul sistem8 este Supervisor (S). Stratul Supervizor este cel mai înalt nivel care se află pe sistemul local al robotului. Princi-palele funcții ale

acestui nivel sunt crearea, gestionarea, uciderea și repornirea microserviciilor. La pornire, sistemul va fi lansat în modul supervizor, unde există doar supervizorul care va crea alte microservicii, cum ar fi Monitoarele hardware necesare pentru a verifica starea inițială a hardware-ului. După verificarea inițială mai multe monitoare hardware sunt generate și începe colectarea datelor. În următorul pas Supervizorul creează lucrători care încep prelucrările de date locale și le trimit către Cloud sau extrag date din Cloud.

Stratul superior este Cloud care poate conține mai multe sub-niveluri unde fiecare sub-nivel are o funcție specială, cum ar fi stocarea, procesarea, securitatea și întreținerea. Deoarece ROBIN-Cloud se presupune că reunește datele dintr-un număr mare de dispozitive, alegem un sistem cu brokeri. Brokerul de proces gestionează toate interogările primite și asigură că timpul procesorului este gestionat corespunzător prin coordonarea platformei de procesare, în timp ce brokerul de date oferă un echilibru corect de încărcare pentru serverele de stocare. Platforma de stocare gestionează stocarea, replicarea, toleranța la erori atât pentru hardware, cât și pentru software, făcând datele disponibile peste tot în Cloud pentru procesarea rapidă, securizată și nelimitată a fluxurilor sau seriilor.

Această arhitectură multi-nivel impune ca, componentele dintr-un nivel să poată comunica numai cu alte componente din nivelurile adiacente. Stratul hardware poate interacționa numai cu Microservices Layer, în special numai cu microservice Hardware Monitor, care sunt respons-abile pentru gestionarea hardware, colectarea datelor și traducerea de la comenzi la nivel înalt la instrucțiunile mașinilor (folosind API-uri). Microserviciile și supervisorul comunică prin schim-bul de mesaje. Microserviciile nu pot comunica direct între ele, trebuie să trimită mesajul către supervisor care îl redirecționează către destinație. Folosind această abordare, asigurăm izolarea între componentele din cel de-al doilea nivel, deoarece supervisorul poate începe, opri sau actualiza starea, independent pentru fiecare microserviciu.

6.5 Procesarea fluxurilor de lucru locale / Cloud

My Cloudy Time Machine are două componente principale de procesare: prelucrarea locală și prelucrarea în Cloud. Prima componentă se ocupă de obicei numai de prelucrarea datelor în timp real, deoarece ghidează sistemul robot prin mediul înconjurător și oferă interacțiune instantanee cu utilizatorii și permite luarea deciziilor cât mai rapide posibil, chiar dacă robotul nu se poate conecta la Cloud. Procesarea cloud se referă atât la procesarea datelor în timp real (de exemplu, stocarea în timp real a datelor, preprocesarea Data Capsules), cât și la procesarea datelor offline (de exemplu, post-procesare de Data Capsules interogări, algoritmi de învățare mecanică). Prezentăm ambele fluxuri de lucru în următoarele subsecțiuni.

6.5.1 Procesarea locală

Componenta locală de procesare este responsabilă cu interacțiunea directă cu mediul, colectarea datelor și preprocesarea datelor. Prezentăm acest flux de lucru în figura 6.7.

Punctul de intrare pentru acest flux de lucru este utilizatorul deoarece poate influența com-portamentul robotului. Pentru simplitate, considerăm că sistemul poate avea doar trei moduri: Stop, Exploration and Record. După pornire, My Cloudy Time Machine se află în Explo-ration mode în timp ce sistemul achiziționează date utilizând senzorii și dispozitivele periferice. Comenzile utilizatorilor sunt primite dar sunt acceptate și executate numai în modul Explorare (modul de înregistrare blochează).

Comenzile sau evenimentele pot declanșa sistemul să ia o decizie. Senzorii declanșați produc acțiuni care sunt transmise la textbf Environment Interaction Engine. Această entitate este responsabilă de acțiunile în timp real ca răspuns la parametrii de mediu.

Evenimente audio sunt trimise la Audio Microservice. Această entitate colaborează cu Machine Learning Engine (ML Engine) pentru recunoașterea mesajelor vocale și audio. Mo-torul ML aplică algoritmi de reducere a zgomotului deoarece sunetul înregistrat poate conține zgomot de la motoare sau alte surse de zgomot. Pentru o precizie mai mare, acest motor combină algoritmi și metode de

învățare în mai multe mașini și inteligență artificială. Atunci când sistemul recunoaște ceva (de exemplu, comanda "Remember this!") face o cerere și o trimite Supervizorului.

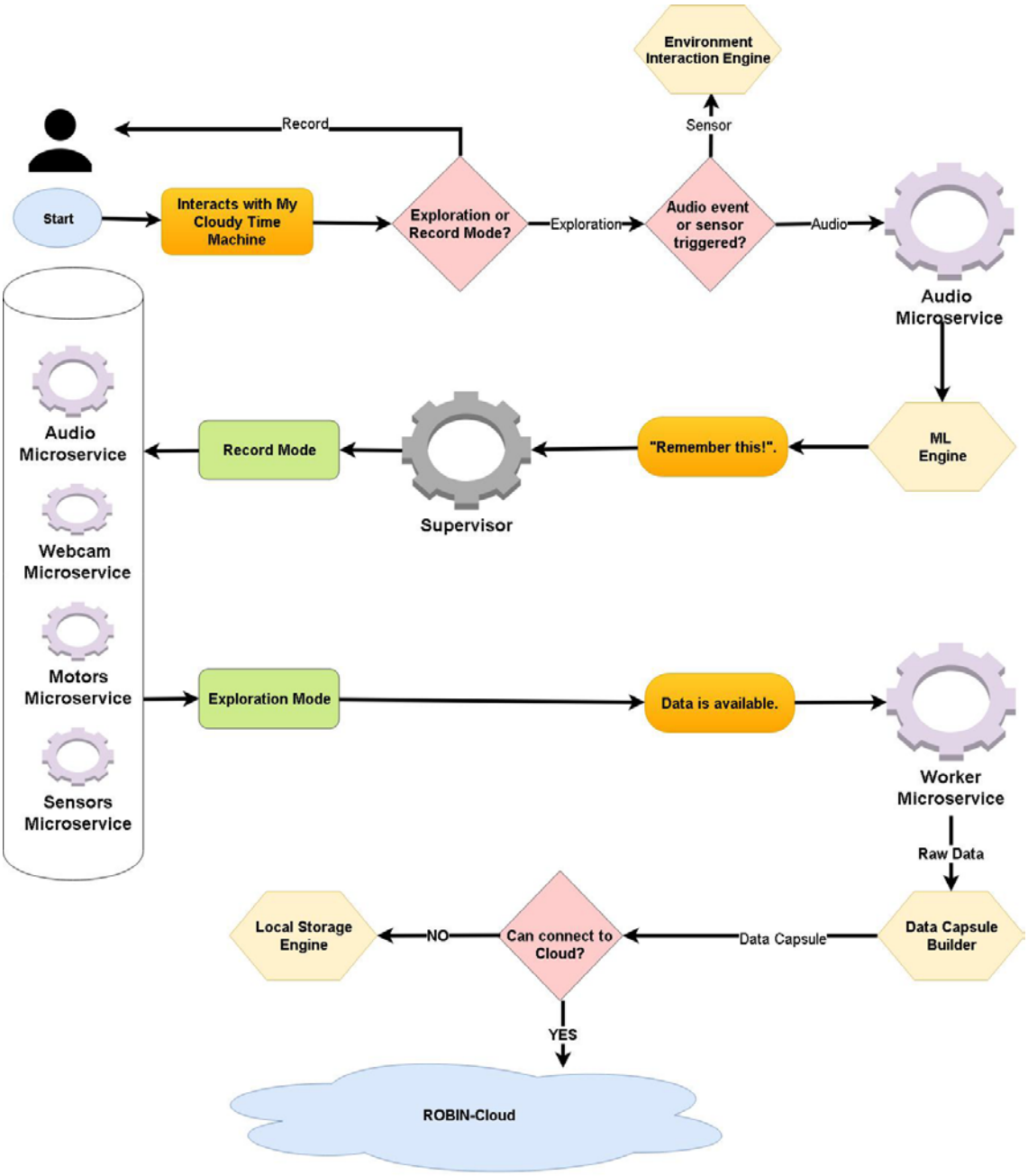


Figura 6.7. Fluxul de lucru local

The Supervisor primește o notificare și pregătește sistemul pentru înregistrare. După veri-ficarea și pregătirea sistemului, Supervisorul trimite o comandă de înregistrare către componentele necesare, cum ar fi Audio Microservice, Webcam Microservice, Motors Microservice, Sensors Microservice. Fiecare serviciu primește id-ul de înregistrare, calea spre spațiul de sto-care local unde poate stoca datele înregistrate temporar și poate primi parametri de înregistrare legați de serviciu (de exemplu, numărul de grade pentru servomotor).

Înregistrarea este de obicei terminată prin expirarea cuantică, dar poate fi făcută și de către utilizator în mod explicit. Modul Explorare este restabilit după terminarea înregistrării. Supervi-zorul verifică datele noi și trimite cererea de procesare pentru o nouă capsulă de date la Worker Microservice. Restul sistemului este configuraT care poate primi și procesa o altă comandă de înregistrare. Lucrătorul lucrează într-un mod asincron, ceea ce face sistemul mai fiabil și mai receptiv.

Lucrătorul începe să proceseze datele și construiește o Data Capsule care este stocată inițial în sistemul local. Data Capsules sunt împinse în Cloud dacă există o conexiune disponibilă. Cloud poate

fi imposibil de atins pentru o perioadă lungă de timp. Având în vedere acest fapT sistemul este pregătit cu o capacitate de stocare locală mare, care poate să dețină Data Capsules până când Cloud-ul este accesibil din nou.

6.5.2 Prelucrarea Cloud

După crearea a cel puțin o Data Capsule, sistemul începe să împingă capsule în Cloud unde datele sunt stocate și procesate. Afișăm fluxul de lucru Cloud din figura 6.8. Există două puncte de intrare sau două acțiuni care ar putea accesa aplicația Cloud din My Cloudy Time Machine. Prima acțiune este împingerea unei noi Data Capsule în Cloud de către sistemul robotului, iar cea de-a doua este declanșată de utilizator când utilizează platforma noastră pentru a prelua informații. Calea urmată de o nouă Data Capsule începe cu Worker Microservice de la serviciul local care a făcut cererea către Cloud. Cererile sunt agregate de Access Points (AP) și trimise la ROBIN-Cloud. După ce o solicitare intră în sistem, este procesată de Brokers: Data Broker și Process Broker. Aceștia sunt responsabili pentru echilibrarea încărcării atât a acțiunilor de stocare, cât și a acțiunilor de procesare care sunt declanșate de toate tipurile de evenimente viitoare.

O cale merge spre Platforma de stocare. O Data Capsule există de la Data Broker și poate merge direct la Platforma de stocare unde este stocată într-un format nemodificat în una sau mai multe baze de date. Reținem că strategiile de replicare sunt utilizate pentru recuperarea rapidă a informațiilor. Dacă este necesară o prelucrare suplimentară pentru o anumită capsulă de date de intrare, aceasta este transmisă serviciului de date, care transformă capsula într-o versiune finală care este apoi trecută la platforma de stocare.

Cea de-a doua cale pentru o Data Capsule este prin intermediul Process Broker. O capsulă este stocată în timp ce vine și o copie este transmisă componentelor de procesare pentru agregarea în timp real. Aceasta permite platformei de procesare să proceseze capsule și să actualizeze o stare în timp real pentru anumite componente, cum ar fi agenții care implementează algoritmi de învățare automata.

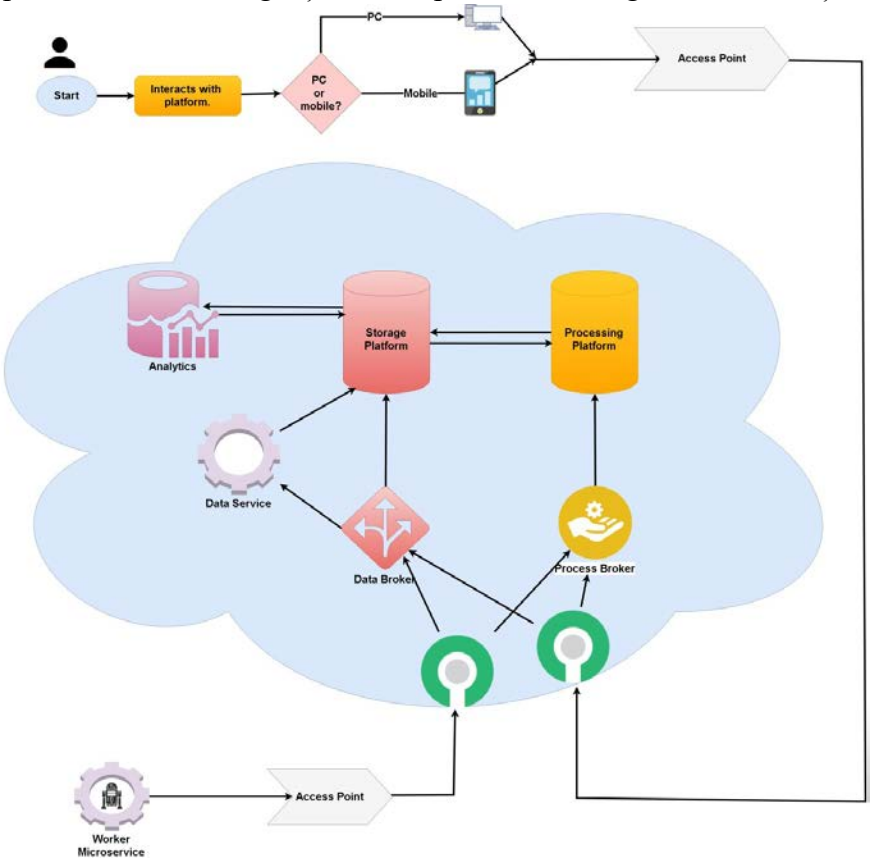


Figura 6.8. Fluxul de prelucrare in Cloud

Platforma de stocare și platforma de procesare schimbă mesajele oricând. De exemplu, atunci când utilizatorul trimite o solicitare de la platforma GUI pentru recuperarea unor Data Capsule, Platforma de procesare trimite o interogare către Platforma de stocare și așteaptă capsulele nece-sare. Platforma de stocare execută interogarea și trimite datele înapoi la Platforma de procesare. Se pot efectua procesări suplimentare (de exemplu, datele de sortare sau de filtrare, algoritmi / aplicații de învățare mecanică), după care rezultatele sunt trimise înapoi utilizatorului.

6.6 Continuarea cercetărilor

In etapa următoare, eforturile de cercetare si dezvoltare se vor concentra pe următoarele:

- Dezvoltarea componentei de Complex Event Processing
- Dezvoltarea componentei Context Broker
- Testarea extinsă a plaformei robotice in Cloud și îmbunătățirea performanțelor
- Proiectarea și implementarea algoritmilor de planificare inteligentă și adaptivă în timp real pentru sisteme de învățare
- Proiectarea și implementarea algoritmilor pentru procesarea în Cloud, cuplarea cu SLAM, planificarea rutelor mașinilor autonome

Raportul in extenso asociat etapei I a proiectului ROBIN-Cloud se găsește la adresa http://aimas.cs.pub.ro/robin/rezultate/

7. Diseminare

În cadrul proiectului ROBIN s-au realizat următoarele acțiuni de diseminare.

- Site-ul Web al proiectului (actualizat permanent) cu secțiuni pentru proiectele componente și o secțiune separată pentru promovare servicii și rezultate

http://aimas.cs.pub.ro/robin (sau la http://robin.cs.pub.ro)

- Publicarea de lucrări științifice se vedea secțiunile fiecărui proiect component

Toate lucrările publicate au în secțiunea de mulțumiri (Acknowledgement) menționat proiectul ROBIN.

- Organizarea de ateliere de lucru pe proiecte componente, la Universitatea Politehnica din București, cu ocazia evenimentului de deschidere al proiectului ROBIN

- Organizarea unei manifestări științifice de prestigiu sub egida proiectului ROBIN
    - **HiPerGRID**. *Special Session on High Performance Grid Middleware, in conjunction with ICCP 2018, to be held on September 6 – 8, 2018 in Cluj-Napoca, Romania*, sub egida proiectului ROBIN

- *This edition of the HiPerGRID special session is organized in the framework of the research complex project ROBIN (PN-III-P1-1.2-PCCDI-2017-0734).*

https://hipergrid.hpc.pub.ro/

- Diseminare în mediul online
  - Realizarea unei pagini de Facebook a proiectului.

https://www.facebook.com/proiectulrobin/

  - Realizarea paginii de Twitter a proiectului

https://twitter.com/ProiectulRobin

  - Publicarea pe YouTube a unor filmulețe cu implementări și demonstrații realizate

ROBIN-Social: https://www.youtube.com/watch?v=bBFZnbMhxyg
ROBIN-Car: https://www.youtube.com/watch?v=J4U24itHHkc

- Întâlniri cu reprezentanți ai companiilor CITST, Prime Motors, Terrasigna, Beia Consult International și All Business Management pentru discutarea potențialului de valorificare a rezultatelor de cercetare obținute în proiect.

8. Angajarea de noi cercetători

În cadrul propunerii de proiect s-a indicat crearea a 11 posturi pentru noi cercetători. Pentru aceste posturi s-au organizat concursuri de angajare, cu respectarea legislației în vigoare și diseminarea anunțului de concurs.
Până în prezent, au fost angajați următorii noi cercetători.
La UNIVERSITATEA POLITEHNICA DIN BUCUREȘTI (CO) au fost ocupate toate cele 5 posturi promise în propunerea de proiect, astfel:

- Iancu David Traian, asistent de cercetare științifică, doctorand cu frecvență
- Hărțan Liviu Adrian, asistent de cercetare științifică, doctorand cu frecvență
- Neagu Laurențiu Marian, asistent de cercetare științifică, doctorand cu frecvență

Au câștigat concursul organizat la începutul lui decembrie și sunt în curs de angajare:

- Gavril Alexandru, asistent de cercetare științifică, doctorand cu frecvență
- Awada Alex Imad, asistent de cercetare științifică, doctorand cu frecvență

La INSTITUTUL DE CERCETARI PENTRU INTELIGENTA ARTIFICIALA „MIHAI DRAGANESCU" au fost ocupate cele 2 posturi promise în propunerea de proiect, astfel:

- Cioroiu George, asistent de cercetare științifică
- Badea Valentin Gabriel, asistent de cercetare științifică

La UNIVERSITATEA TEHNICĂ DIN CLUJ – NAPOCA a fost ocupat postul promis în propunerea de proiect, astfel:

- Corcheș Cosmina Emilia, asistent de cercetare științifică, doctorand cu frecvență

La INSTITUTUL DE MATEMATICA "SIMION STOILOW" AL ACADEMIEI ROMANE a fost ocupat unul din cele două posturi promise, de către

- Paraicu Iulia, asistent de cercetare științifică

iar pentru cel de al 2-lea concursul este în derulare
La UNIVERSITATEA "DUNAREA DE JOS" DIN GALAȚI, cu 1 post de nou cercetător, concursul este în derulare.

9. Concluzii

Prezentul raport reprezintă descrierea activităților de cerectare și a rezultatelor obținute de consorțiul proiectului în cadrul primei etape a proiectului ROBIN. Pentru fiecare proiect component s-au pus în evidență obiectivele, activitățile etapei actuale și modul în care acestea au fost duse la îndeplinire. De asemenea, în secșiunile aferente proiectelor componente s-au pus în evidență direcții de cerectare și dezvoltare din etapa viitoare. Este de remarcat faptul că realizările proiectului au cosntat atât în contribuții teoretice (studii ale dezvoltărilor în domeniu, noi arhitecturi, noi algoritmi sau algoritmi îmbunătățiți) iar fiecare propunere de arhitectură sau algoritm a fost validată print-o implementare, chiar și preliminară, așa cum se poate observa din evaluarea performanțelor atinse, din filmulețele demnstrative de pe youtube, sau din lucrările științifice publicate în conferințe și reviste de prestigiu în lumea științifică.

Este de asemenea de subliniat faptul că cele 5 proiecte componente reprezintă o viziune unitară a conceptului de roboți autonomi, inteligenți, fie că aceștia sunt utilizați pentrua sistenâă socială sau interacțiune cu clienții, fie că sunt utilizați pentru pilotaj automat. Proiectele componente ROBIN-Context și ROBIN-Cloud, pe lângă contribuțiile teoretice intrinsece, au rolul de a aduce și crea tehnologii moderne, inovatoare care să susțină și să eficientizeze realizarea roboților mai sus menționați.

Bibliografie

[Seifer&Mataric, 2005] David Feil-Seifer and Maja J Mataric. 2005. Defining socially assistive robotics. In Rehabilitation Robotics, 2005. ICORR 2005. 9th International Conference on. IEEE, 465–468.

[Payr et.al., 2015] Sabine Payr, Franz Werner, and Katharina Werner. 2015. Potential of robotics for ambient assisted living. Vienna: FFG benefit (2015).

[Trăscău et al., 2018] Trăscău, M., Sorici, A., & Florea, A. (2018, June). Detecting Activities of Daily Living Using the CONSERT Engine. In *International Symposium on Ambient Intelligence* (pp. 94-102). Springer, Cham.

[Sorici et al., 2015a] Sorici, A., Picard, G., Boissier, O., Zimmermann, A., & Florea, A. (2015). CONSERT: Applying semantic web technologies to context modeling in ambient intelligence. Computers & Electrical Engineering, 44, 280-306.

[Cook, 2010] Cook D. J. (2010). Learning Setting-Generalized Activity Models for Smart Spaces. IEEE intelligent systems, 2010(99), 1.

[Zhang & Zhang, 2008] Zhang C-X, Zhang J-S. (2008) A local boosting algorithm for solving classification problems. Computational Statistics and Data Analysis; 52(4):1928–1941.

[Liu et al., 2016] Liu, Y., Nie, L., Liu, L., & Rosenblum, D. S. (2016). From action to activity: sensor-based activity recognition. *Neurocomputing*, *181*, 108-115.

[Höppner, 2001] F. Höppner, Discovery of temporal patterns, In: Principles of Data Mining and Knowledge Discovery, Springer, Berlin, Heidelberg, 2001, pp. 192–203.

[Richardson & Domingos, 2006] Richardson, M., & Domingos, P. (2006). Markov logic networks. *Machine learning*, *62*(1-2), 107-136.

[Riboni et al., 2016] Riboni, D., Sztyler, T., Civitarese, G., & Stuckenschmidt, H. (2016, September). Unsupervised recognition of interleaved activities of daily living through ontological and probabilistic reasoning. In *Proceedings of the 2016 ACM International Joint Conference on Pervasive and Ubiquitous Computing* (pp. 1-12). ACM.

[Bai et.al. 2018] Bai, S.; Kolter, J.Z.; Koltun, V. An empirical evaluation of generic convolutional and recurrent networks for sequence modeling. arXiv preprint arXiv:1803.01271 2018

[Shahroudy et.al. 2016] Shahroudy, A.; Liu, J.; Ng, T.; Wang, G. NTU RGB+D: A Large Scale Dataset for 3D Human Activity Analysis. CoRR 2016, abs/1604.02808

[Zhu et.al. 2016] Zhu, W.; Lan, C.; Xing, J.; Zeng, W.; Li, Y.; Shen, L.; Xie, X. Co-occurrence Feature Learning for Skeleton based Action Recognition using Regularized Deep LSTM Networks. CoRR 2016, abs/1603.07772.

[Zhang et.a;. 2018] Zhang, X.; Xu, C.; Tian, X.; Tao, D. Graph Edge Convolutional Neural Networks for Skeleton Based Action Recognition. CoRR 2018, abs/1805.06184, [1805.06184]

[Marcu et.al. 2018] A. Marcu, D. Costea, E. Slusanschi, and M. Leordeanu. A multistage multi-task neural network for aerial scene interpretation and geolocalization.

CoRR, abs/1804.01322, 2018.

[Burceanu&Leirdeanu,2018] E. Burceanu and M. Leordeanu, Learning a Robust Society of Tracking Parts using Co-occurrence Constraints, VOT2018 Challenge, European Conference on Computer Vision (ECCV) 2018

[Ryu, 2019] H. Ryu, M. Leordeanu, A. Petreanu, B. Jeon. A. Leica, Efficient two-stage approach for speed bumper segmentation and classification in fish-eye images, submitted to the international Conference on Computer Vision and Pattern Recognition (CVPR), 2019.

[Croitoru et.al., 2019] I. Croitoru, V. Bogolin and M. Leordeanu, Unsupervised learning for foreground object segmentation, Status: Accepted with Major Revision at the International Journal of Computer Vision (IJCV).

[Xu et.al. 2017]Xu, H., Gao, Y., Yu, F. and Darrell, T., 2017. End-to-end learning of driving models from large-scale video datasets. IEEE International Conference on Computer Vision and Pattern Recognition (CVPR), 2017

[Hu et.al. 2012] G. Hu, W. P. Tay, and Y. Wen. Cloud robotics: architecture, challenges and applications. IEEE Network, 26(3):21–28, May 2012.

[Waibel et.al. 2011] M. Waibel et al. Roboearth - a world wide web for robots. Robotics and Automation Mag., vol. 18, no. 2, June 2011, pp. 69-82., 2011

[Hunziker et.al. 2013] D. Hunziker, M. Gajamohan, M. Waibel, and R. D'Andrea. Rapyuta: The roboearth cloud engine. pages 438–444, May 2013.

[Aazam et.al. 2016] Aazam, Mohammad, Eui-Nam Huh, Marc St-Hilaire, Chung-Horng Lung, and Ioannis Lambadaris. "Cloud of things: integration of IoT with cloud computing." In Robots and Sensor Clouds, pp. 77-94. Springer, Cham, 2016

[Kehoe et.al. 2015] Ben Kehoe, Sachin Patil, Pieter Abbeel, and Ken Goldberg. A survey of research on cloud robotics and automation. IEEE Trans. Automation Science and Engineering, 12(2):398–409, 2015

[Ermacora et.al. 2016] Gabriele Ermacora, Stefano Rosa, and Antonio Toma. Fly4smartcity: A cloud robotics ser-vice for smart city applications. Journal of Ambient Intelligence and Smart Environments 8(3):347–358, 2016

[Rus et.al. 2013] Vasile Rus, Sidney D'Mello, Xiangen Hu, Arthur C. Graesser. (2013) Recent Advances in Conversational Intelligent Tutoring Systems. AI MAGAZINE, Fall 2013, pp. 42—54.

[Yan et.al. 2017] Zhao Yan, Nan Duan , Peng Chen, Ming Zhou , Jianshe Zhou and Zhoujun Li. (2017). Building Task-Oriented Dialogue Systems for Online Shopping. In Proceedings of the Thirty-First AAAI Conference on Artificial Intelligence (AAAI-17).

[Liu et.al. 2018] Qianlong Liu , Zhongyu Wei, Baolin Peng , Xiangying Dai , Huaixiao Tou , Ting Chen , Xuanjing Huang , Kam-fai Wong. (2018). Task-oriented Dialogue System for Automatic Diagnosis. In Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers), ACL 2018, July 2018, Melbourne, Australia, pp. 201—207.

[Pais&Tufis 2018] Vasile Păiș, Dan Tufiș,. Computing distributed representations of words using the CoRoLa corpus. In Proceedings of the Romanian Academy, series A, pp. 403-410, 2018

[Mititelu et.al. 2018] Verginica Barbu Mititelu, Dan Tufiș, Elena Irimia: The Reference Corpus of the Contemporary Romanian Language (CoRoLa), Miyazaki, Japan, LREC 2018, pp. 1178-1185, ISBN 979-10-95546-00-9

[Pipa& Boros 2016] Sonia Pipa, Tiberiu Boroș: A Recurrent Neural Networks Approach for Keyword Spotting Applied on Romanian Language. In Proceedings of the 12th International Conference "Linguistic Resources and Tools for Processing of the Romanian Language", 27-29 October, 2016, Romania. „Al. I. Cuza" University Publishing House, pp. 111-119, ISSN 1843-911X.

[Tufis&Mititelu 2014] Dan Tufiș, and Verginica Barbu Mititelu. The Lexical Ontology for Romanian. In Nuria Gala, Reinhard Rapp, Gemma Bel-Enguix (eds) Recent Advances in Language Production, Cognition and the Lexicon, Springer, vol. 14, 2014, pp. 491-504

[Bellavista& Zanni 2017] Paolo Bellavista and Alessandro Zanni. Feasibility of fog computing deployment based on docker containerization over raspberrypi. ICDCN, 2017.

[Bojchevski et.al. 2017] Aleksandar Bojchevski, Yves Matkovic, and Stephan Günnemann. Robust spectral clustering for noisy data: Modeling sparse corruptions improves latent embeddings. In Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '17, pages 737–746, New York, NY, USA, 2017. ACM.

[Goodfellow et. al. 2016] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. Deep Learning. MIT Press 2016. http://www.deeplearningbook.org

Director de proiect

Prof. Adina Magda Florea