Universitatea "Dunărea de Jos" din Galați

Contribuții la elaborarea unor soluții încorporate (embedded) de conducere în timp real a sistemelor robotice şi vehiculelor autonome

Teză de doctorat

Conducător ştiințific
Prof. Dr. Ing. Adrian Filipescu

Doctorand
Ing. Ioan Şuşnea

Galați, 2009

Rezumat

Principala teză a aceste cercetări este că o rețea distribuită de microcontrollere încorporate poate oferi soluții fezabile pentru cele mai importante probleme legate de conducerea roboților autonomi. Unele elemente ale acestui sistem de control distribuit sunt localizate la nivelul vehiculului autonom, altele sunt dispersate într-un "mediu inteligent". Această abordare poate conduce la reduceri semnificative de cost pentru sistemele robotice şi poate contribui la dezvoltarea unor "asistenti robotici personali" pentru îngrijirea persoanelor în vârstă sau cu diverse dizabilități.

Totuşi, microcontrollerele uzuale au limitări severe în ce priveşte puterea de calcul şi capacitatea de stocare a datelor, iar task-urile asociate cu conducerea roboților autonomi sunt deosebit de complexe.

Din acest motiv, au fost explorate următoarele directii de cercetare:

- În primul rând, a fost propusă o structură de control distribuit multi-procesor. În acest fel, task-urile de navigație sunt împărțite în sub-task-uri mai simple, care sunt asignate unor microcontrollere distincte.
- În al doilea rând, au fost propuşi algoritmi noi, compatibili cu limitările microcontrollerelor uzuale.

Soluțiile propuse au un pronunțat caracter interdisciplinar, îmbinând elemente de electronică, inginerie software si ingineria sistemelor.

Întreaga cercetare a fost subordonată ideii de implementare a *unui asistent robotic personal pentru persoane în vârstă sau cu dizabilități*, dar soluțiile propuse sunt aplicabile şi pentru realizarea altor tipuri de roboți de serviciu.

Sunt propuse contribuții la rezolvarea următoarelor probleme ale conducerii roboților autonomi:

Localizarea.

Problema localizării roboților mobili este esențială în orice algoritm de navigație şi de aceea se propune o nouă metodă de localizare, bazată pe balize active ultrason. Această metodă permite determinarea simultană a distanței şi orientării robotului față de balize şi reduce semnificativ volumul de calcule necesar pentru localizare.

Programarea roboților

Se propune un limbaj de programare simplu, inspirat de standardul IEC61131-3, destinat programării PLC-urilor, limbaj pe care l-am denumit RDPL (Robot Decision Programming Language).

În această abordare, diverse evenimente, asociate cu senzori distribuiți în mediu sunt tratate ca intrări digitale obişnuite, iar comenzile de mişcare a vehiculului către ținte predefinite sunt tratate ca ieşiri digitale din sistem. Limbajul rezultat este suficient de simplu pentru a putea fi folosit de utilizatori care au cunoştințe elementare de programare.

Urmărirea unei traiectorii

Se propune un algoritm pentru urmărirea unei traiectorii bazat pe conceptul de "feromoni virtuali". Aceştia sunt stocați in memoria unui "server de feromoni" care menține o hartă a mediului, în care este încorporată informația despre rutele posibile, în mod similar cu mecanismul prin care unele insecte sociale isi coordonează activitățile. Algoritmul propus permite controlul roboților individuali, dar şi a formațiunilor de roboți.

Ocolirea obstacolelor

Este prezentat un algoritm reactiv de ocolire a obstacolelor în timp real, pe care l-am denumit, "algoritmul ricoşeului". Acesta functioneaza cu senzori low-cost si poate fi implementat pe majoritatea microcontrollerelor uzuale.

Folosirea tehnologiei RFID pentru conducerea roboților mobili

Sunt prezentate aplicații ale tehnologiei RFID (Radio Frecquency IDentification) pentru localizarea roboților mobili si pentru urmărirea unei traiectorii.

Abstract

The main thesis of this work is that a distributed network of embedded microcontrollers is able to provide feasible solutions to the most important problems of autonomous robot control. Some elements of the distributed control system proposed are located on the mobile robot, others are deployed in a *"smart environment"* . This can lead to significant cost reduction of the robotic systems, and could allow the development of affordable personal robotic assistants, designed to support the elderly and disabled.

However, usual microcontrollers have severe limitations in what concerns computing power and storage capacity, and the tasks associated with robot control are proven to be very complex.

Therefore, two design strategies are explored:

- First, a multi-processor control structure is proposed. Thus, the complex task of robot control is split into several simpler sub-tasks, each assigned to a different microcontroller.
- Moreover, some new, simpler, navigation algorithms, compatible with the limited computational power of the usual microcontrollers are presented.

Given the multidisciplinarity of the field of robotics, some of the topics covered in this dissertation are related to Electronics, and software engineering, but the main focus remains on control engineering.

The entire research was subordinated to the idea of implementing a "Personal Robotic Assistant" for the elderly or disabled, but the solutions proposed are equally aplicable to the implementation of other service robots.

The following research fields are addressed:

Robot localization in indoor environments.

The localization problem is crucial for any robot control algorithm, and therefore a new localization method, based on active ultrasonic beacons is proposed. This method allows simultaneous measurement of the distance and orientation of the autonomous vehicle with respect to the active beacon.

Robot programming.

A simple programming language, inspired by the standard IEC61131-3, intended for PLCs is proposed. In this approach, various events, detected by remote sensors, are treated as regular logical inputs in the system, while predefined navigation goals are interpreted as logical outputs.

Path following.

The algorithm for path following proposed is based on the concept of “virtual pheromones”. These are stored by a “virtual pheromone server”, which updates a map of the enviromnent, embedding pheromone trail information. The control algorithm proposed allows controlling individual robots, as well as robot swarms in a bio-mimetic approach.

Obstacle avoidance.

A simple, reactive algorithm, called “the bubble rebound algorithm” is proposed. It requires low-cost sensors, and can be implemented on most available microcontrollers.

The use of the RFID technology for robot control.

Several applications of the RFID technology for robot localization and path following are also presented.

Cuprins

Lista figurilor

Fig. 7.11 Ilustrarea procesului de ocolire a unui obstacol

Fig. 7.12 Navigație pe un coridor cu obstacole multiple

Fig. 7.13 Navigație cu definirea unor ținte intermediare

Lista tabelelor

Lista abrevierilor

ACO - Ant Colony Optimization

AGV – Autonomous Guided Vehicle

A.M. - Amplitude Modulation

ASK – Amplitude Shift Key

DTMF - Dual Tone Multi Frequency

FLC – Fuzzy Logic Controller

ICC – Instantaneous Center of Curvature

ITD – Interaural Time Difference

PLC – Programmable Logic Controller

PRA – Personal Robotic Assistant

RDPL – Robot Decision Programming Language

RFID – Radio Frequency Identification System

VFH - Vector Field Histogram

VPF - Virtual Potential Field

XML – Extensible Markup Language

1

Introducere

1.1 Formularea problemei

Într-un studiu intitulat *"A robot in every home"* [1], Bill Gates, fondatorul Microsoft Corporation, observă similitudini între dinamismul evoluțiilor înregistrate în domeniul roboticii şi rapiditatea cu care a evoluat industria computerelor începând din anii '70 si conchide:

"Because the new machines will be so specialized and ubiquitous - and look so little like the two-legged automatons of science fiction - we probably will not even call them robots. But as these devices become affordable to consumers, they could have just a profound impact on the way we work, communicate, learn and entertain ourselves as the PC has had over the past 30 years."

Având în vedere, pe de o parte, impactul extraordinar pe care l-ar putea avea roboții personali, iar pe de alta parte multitudinea de realizări absolut remarcabile deja raportate in acest domeniu (vezi, de exemplu, [2], [3], [4]) se pune întrebarea de ce, totuşi, nu avem încă "a robot in every home"?

Printre răspunsurile posibile, sugerăm:

- Prețul roboților este încă extrem de ridicat.

- Programarea roboților este încă greoaie, inaccesibilă end-user-ilor;
- Există o paradigmă antropomorfică ([5]) in abordarea problemelor roboticii, care ne determină să atribuim acestora o serie de caracteristici umane, deocamdată dificil sau imposibil de realizat.

Nu dispunem de date statistice privind evoluția prețurilor sistemelor robotice, dar, urmărind istoricul realizărilor demne de atenție în acest domeniu, se poate observa cu usurință că numărul si importanța acestora a crescut semnificativ începând din anii ’90, odată cu răspandirea microcontrollerelor şi a structurilor incorporate (embedded). Acestea au o serie de avantaje care le recomandă pentru aplicații în domeniul roboticii:

- Cost redus;
- Consum redus de energie, ceea ce permite funcționarea îndelungată în regim de alimentare de la baterii de acumulatoare;
- Facilități de comunicație care permit realizarea cu usurintă a unor structuri multiprocesor, ceea ce măreste flexibilitatea de proiectare si reducerea timpului necesar pentru scrierea aplicațiilor software.

Pe de altă parte, există şi câteva dezavantaje evidente ale structurilor încorporate, cum ar fi:

- Putere de calcul redusă față de computerele “clasice”;
- Capacitate redusă de stocare a datelor.

Având în vedere complexitatea task-urilor asociate cu percepția, navigația si decizia în sistemele robotice cât si limitările enumerate mai sus, rezultă, la nivel hardware, necesitatea adoptării unor arhitecturi multiprocesor, iar la nivel software selectarea sau elaborarea unor algoritmi compatibili cu limitările de memorie si putere de calcul specifice sistemelor încorporate.

Elementele subliniate mai sus justifică formularea principalei teze a acestei tezei, după cum urmează:

Un sistem ditribuit de structuri incorporate (embedded) poate oferi soluții fezabile pentru principalele probleme legate de conducerea in timp real a roboților autonomi.

Această formulare justifică titlul tezei:

“Contributii la elaborarea unor solutii bazate pe sisteme încorporate (embedded) pentru conducerea in timp real a sistemelor robotice şi a vehiculelor autonome.”

Formularea de mai sus rămâne extrem de generală şi sunt necesare o serie de delimitări care să asigure coerența cercetărilor.

Din multitudinea de aplicații posibile ale roboticii, am ales să subordonăm cercetările, ideii de implementare a conceptului de “asistent robotic personal” (Personal Robotic Assistant - PRA) destinat persoanelor in varsta, sau cu diverse dizabilități senzorio-motorii.

Această opțiune este justificată de datele statistice publicate de GAO (U.S. General Accounting Office) [6] si U.S. Census [7], care arată că generația baby-boom a Americii este acum la varsta pensionării, ceea ce va mări in viitorul apropiat procentul persoanelor cu vârste peste 65 de ani din totalul populației. Situația este asemănătoare şi în celelalte state industrializate.

În România, conform datelor statistice oficiale, publicate de Ministerul Muncii, Familiei si Protectiei Sociale ([8]) si de Agenția Națională pentru Handicapați ([9]) existau la 31.12.2007 un număr de 643458 handicapați, reprezentând 2.99% din totalul populației de 21537563 persoane. Totodată, la 30.09.2008 existau un număr de 5525957 pensionari (25.6% din total populație), dintre care 909188 pentru

invaliditate, ceea ce conduce la un total de 1552646 handicapați si invalizi (7.2% din total populație). Cifrele de mai sus reflectă o situație gravă, iar tendința de îmbătrânire a populației tinde să mărească numărul persoanelor care au nevoie de supraveghere permanentă si de ingrijire medicală minimală.

Cum vârsta înaintată este în mod inerent asociată cu reducerea acuității senzoriale şi cu o serie de dificultăți locomotorii, fenomenul de îmbătrânire a populației, manifestat, între altele, prin creşterea numărului de persoane care necesită supraveghere sau asistentă medicală minimală permanentă, aduce presiuni importante asupra sistemului medical şi asupra economiei în general ([10]).

În aceste condiții, dezvoltarea rapidă a unor sisteme robotice capabile să compenseze parțial limitările senzorio-motorii ale persoanelor în vârstă, ar putea fi o soluție a acestor probleme.

1.2 Obiectivele cercetării

Obiectivul general al cercetării descrise aici poate fi formulat in felul următor:

Demonstrarea fezabilității ideii de a implementa un sistem de conducere a vehiculelor autonome, destinate suplinirii deficiențelor senzorio-motorii ale persoanelor în vârstă, sau cu diverse dizabilități, bazat pe o structură multi-procesor, realizată cu microcontrollere încorporate (embedded).

Rezultă următoarele obiective specifice:

1. *Delimitarea conceptului de PRA în contextul vehiculelor autonome si stabilirea structurii hardware a sistemului de conducere a acestuia;*
2. *Elaborarea unei metode de localizare cu performanțe superioare sistemelor bazate pe dead reckoning;*
3. *Elaborarea si implementarea unui algoritm de urmărire a unei traiectorii compatibil cu microcontrollerele uzuale;*

4. *Elaborarea si implementarea unui algoritm de ocolire a obstacolelor compatibil cu senzorii low-cost disponibili (ultrason, infrared);*
5. *Elaborarea unei metode de selectare a țintelor de navigatie pornind de la informațiile furnizate de senzorii din mediu si de subsistemul de navigatie.*

1.3 Sumarul contribuțiilor aduse de cercetarea descrisă în prezenta tezã

Pentru realizarea obiectivelor enumerate mai sus, au fost abordate principalele probleme asociate cu conducerea vehiculelor autonome: localizarea, decizia, urmărirea unei traiectorii şi ocolirea obstacolelor.

În problema localizării roboților mobili, se propune o soluție care permite determinarea simultană a distanței şi a orientării robotului în raport cu o baliză ultrason special proeictatã.

Pentru a permite formalizarea descrierii interacțiunii între robot şi mediu, a fost elaborat un limbaj de programare, pe care l-am denumit RDPL (Robot Decision Programming Language), care permite, între altele, selectarea țintelor de navigație în funcție de starea unor senzori şi actuatoare distribuite în mediu.

Pentru problema urmăririi unei traiectorii se propune o soluție bio-mimetică, bazată pe conceptul de “feromoni virtuali”, încorporati intr-o hartă a mediului, stocată de un “server de feromoni”, aflat in comunicatie cu roboții mobili. Soluția propusă permite atât conducerea roboților individuali, cât si a formațiunilor de roboți si are multiple avantaje.

În ce priveşte problema ocolirii obstacolelor, se propune un nou algoritm pur reactiv, pe care l-am denumit “bubble rebound algortihm” (algoritmul ricoşeului), suficient de simplu pentru a fi implementat pe microcontrollere si compatibil cu majoritatea senzorilor uzuali pentru detectarea obstacolelor (ultrason, infrared, laser).

Totodată, sunt explorate posibilitațile de folosire a tehnologiei RFID pentru conducerea roboților mobili şi se propun soluții de localizare si urmărire a unei traiectorii, bazate pe această tehnologie.

1.4 Structura prezentei teze

În afara prezentei introduceri, lucrarea de față este structurată după cum urmează:

- Capitolul 2 formulează o serie de delimitări conceptuale pornind de la stadiul actual al cunoştințelor in domeniul conducerii roboților mobili. Se propune o definiție a "asistentului robotic personal" si o structură a sistemului distribuit de conducere a acestuia.
- Capitolul 3 tratează problematica localizării roboților mobili. Se propune un sistem de localizare bazat pe balize ultrason, care permite atât identificarea balizelor, cât şi determinarea distanței şi a azimutului balizei față de robot. Soluția propusă face obiectul unei cereri de brevet, cu denumirea "*Sistem de localizare pentru roboti mobili si vehicule autonome*", cerere inregistrata la OSIM sub numarul A/01021, din 04-12-2009.
- Capitolul 4 tratează problema deciziei în sistemele de conducere a roboților autonomi. Se propune un nou limbaj de programare, denumit RDPL (Robot Decision Programming Language), derivat din limbajele de programare pentru PLC-uri, descrise de standardul IEC61131-3.
- Capitolul 5 tratează problema urmăririi unei traiectorii. Se propune o soluție de urmarire a traiectoriei bazata pe conceptul de "feromoni virtuali". Spre deosebire de soluțiile cunoscute, care implementează foeromonii artificiali ca urme detectabile cu ajutorul senzorilor, lăsate de agenți in mediu, în abordarea propusă, feromonii virtuali sunt *engrame* create de agenți nu in

mediu, ci într-o *hartã a mediului*, stocatã în memoria unui server de feromoni. Agenții dispersați in mediu se aflã în comunicație permanentã cu severul de feromoni şi folosesc harta dinamicã a mediului, menținutã de server, ca o memorie comunã. Soluția propusã permite definirea şi urmãrirea traectoriei atât pentru roboți individuali, cât şi pentru formațiuni de roboți.

- Capitolul 6 tratează problema ocolirii obstacolelor si prpoune un algoritm reactiv nou, pe care l-am denumit "algoritmul bubble-rebound" (algoritmul ricoşeului).
- Capitolul 7 prezintă rezultatele experimentelor de implementare a soluțiilor propuse pe microcontrollere încorporate low-cost/low power.
- Capitolul 8 este rezervat concluziilor si sugerează posibile directii pentru cercetări viitoare.
- Referințele sunt listate in ordinea citării in textul prezentării. Deoarece nu toate publicațiile proprii sunt citate in mod formal, a fost introdus un paragraf special, conținând lista completã a publicațiilor autorului.
- In Anexe este prezentat un CV sumar al autorului, precum şi listing-ul principalelor aplicații software folosite pentru implementarea algoritmilor descrişi.

Structura tipică a fiecărui capitol este următoarea:

- Formularea problemei;
- Prezentarea soluțiilor cunoscute;
- Contribuții proprii la rezolvarea problemei;
- Concluzii.

Având în vedere numărul foarte mare de soluții cunoscute din literatură pentru fiecare din problemele abordate, sunt trecute in revistă doar acele soluții care au directă legatură cu contribuțiile propuse.

Cercetările descrise în prezenta tezã s-au desfăşurat in Laboratorul de Robotică al Facultății de Ştiința Calculatoarelor din Universitatea Dunărea de Jos, din Galați, sub îndrumarea Prof. Dr. Ing. Adrian Filipescu şi sunt parțial sustinute din grantul ID_641, finanțat de Ministerul Educatiei şi Cercetării.

Robotii mobili folosiți pentru experimente sunt Pioneer3-DX si PeopleBot, produşi de MobileRobots Inc. ([11]), aflați în dotarea laboratorului.

2

Delimitări conceptuale pornind de la stadiul actual al cunoştinţelor in domeniu

2.1 Caracteristicile unui Asistent Robotic Personal (PRA)

Primele referiri la un "nursing robot" datează din 1985, când Borenstein şi Koren ([12]) au propus o platformă mobilă, dotată cu un braț robotic, concepută să suplinească unele deficiențe de mobilitate ale persoanelor cu handicap locomotor.

De atunci, un mare număr de cercetări ([3], [13], [14], ... [36]) au propus diverse soluții fie pentru dezvoltarea unor scaune cu rotile inteligente, fie pentru dezvoltarea unor dispozitive denumite in mod explicit "asistenți robotici".

Principalele funcții/capabilități dezirabile pentru un asistent robotic personal (PRA), destinat persoanelor in vârstă sau cu diverse dizabilități, asa cum sunt reflectate în literatura ştiintifică, sunt următoarele:

- Capacitatea de navigație independentă, cu ocolirea obstacolelor detectate pe parcurs;
- Funcția de protezare locomotorie (walking aid). Este de dorit ca PRA să poată fi folosit ca walker, sau ca scaun cu rotile;
- Funcția de protezare senzorială. PRA trebuie să fie dotat, sau să poata comunica cu senzori capabili să suplinească unele deficiențe senzoriale ale persoanei asistate;
- Funcția de protezare cognitivă (reminder);

- Funcția de ajutor la manipularea unor obiecte (manipulation assistance);
- Managementul teleprezenței prin controlul asupra echipamentului de telecomunicație;
- Monitorizarea unor parametri medicali şi apel automat la urgentă;
- Controlul aparaturii electrocasnice;
- Recunoaşterea unor comenzi vocale.

2.2 Abordarea propusa

Analizând cerințele de mai sus, se impun câteva observații:

- Implementarea tuturor cerințelor enumerate folosind echipamentul amplasat la bordul robotului mobil este dificilă şi contribuie la creşterea semnificativă a prețului sistemului.
- Unele cerințe, de exemplu cea privind asistența la manipularea unor obiecte, care implică includerea in sistem a unui braț robotic, au utilitate practică limitată, dar pot multiplica prețul sistemului de câteva ori.
- Alte cerințe (ex. recunosterea unor comenzi vocale si folosirea acestora pentru controlul aparaturii electrocasnice, monitorizarea unor aparate medicale, sau protezarea cognitiva prin folosirea unui reminder software) au gasit soluții in urma unor cercetări independente de robotică ([37]).

Au rezultat următoarele:

2.2.1 Ipoteze simplificatoare

1. Dispozitivele denumite generic PRA opereaza de cele mai multe ori în interior, în locuințe, spitale, sau centre de reabilitare. Mediul în care evoluează este cunoscut si este, (sau poate fi) prevăzut cu echipament de calcul si telecomunicații, aparatură medicală, etc.

2. Principala funcție a unui PRA rămâne cea de suport locomotor. Acesta trebuie să fie capabil fie să transporte efectiv persoana asistată catre diverse puncte țintă, fie să-l ghideze/sprijine în deplasarea către acele puncte.

Cu aceste ipoteze, se poate formula următoarea definiție:

Definiția 1

Un asistent robotic personal (PRA) este un vehicul capabil să se deplaseze autonom şi să interacționeze în mod programat cu o serie de senzori si actuatoare distribuite într-un mediu inteligent, în scopul compensării unor deficiențe locomotorii sau senzoriale ale persoanei asistate.

2.2.2 Structura unui PRA

Structura propusă pentru un PRA derivă din structura generală de conducere a unui robot autonom, prezentată în figura 2.1.

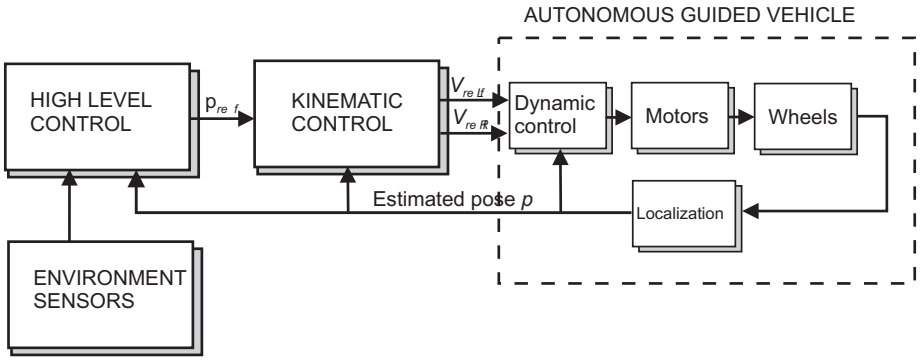


Fig. 2.1 Structura generală tipică de conducere a unui vehicul autonom

Se observă prezența a trei bucle de control ierarhizate. La nivelul inferior se află bucla de control dinamic, implementată de electronica încorporată în vehiculul autonom. Schema bloc a acesteia este prezentată în figura 2.2.

În cazul particular al vehiculelor autonome folosite pe parcursul experimentelor, (roboții mobili Pioneer3-DX şi PeopleBot de la Mobile Robots [11]), controlul dinamic este în întregime realizat de electronica internă. Singurul parametru legat de

dinamica sistemului, care este accesibil pentru nivelele superioare de conducere este acceleratia liniara *a*, $a \in [0, a_{\max}]$, comună pentru ambele roți motoare.

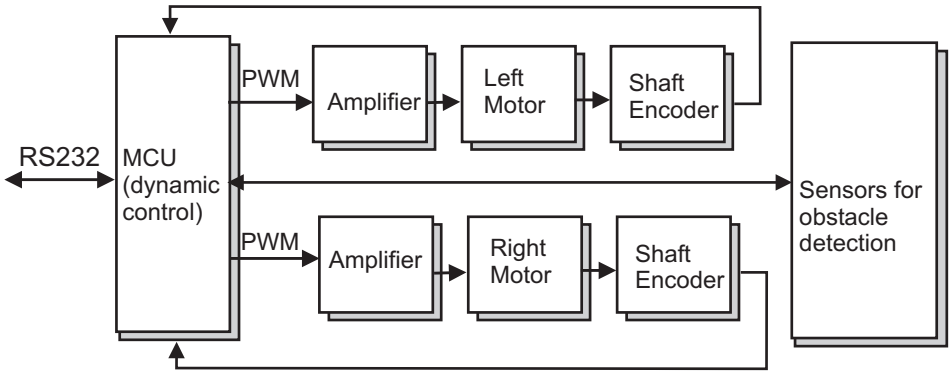


Fig. 2.2 Schema bloc a echipamentului încorporat in AGV

Bucla intermediară de conducere (conducerea cinematică) are ca obiectiv generarea referințelor de viteză pentru cele douã roți motoare v_{refL}, v_{refR} pornind de la o *poziția țintă* $p_{ref} = (x_{ref}, y_{ref}, \theta_{ref})$, sau de la o *traiectorie țintă*, specificată de nivelul superior de conducere, si de la o estimare a poziției curente $p = (x, y, \theta)$, furnizată de *subsistemul de localizare al AGV*. Tot la nivelul acestei bucle se implementează şi algoritmii reactivi de evitare a obstacolelor.

Bucla superioară de conducere are ca obiectiv planificarea şi/sau selecția țintelor de navigație în funcție de informația furnizată de senzorii din mediu şi de o reprezentare internă a mediului.

Pe baza structurii tipice de conducere descrise, rezultă structura propusă pentru PRA, prezentată în figura 2.3.

Fiecare din blocurile funcționale ale sistemului din figura 2.3 este implementat cu un microcontroller distinct.

La bordul vehiculului autonom sunt menținute doar echipamentele strict necesare pentru navigație.

Interfețele cu senzorii şi actuatoarele distribuite în mediu, precum şi echipamentul de telecomunicație şi o parte din task-urile de prelucrare a datelor sunt asignate unor echipamente aflate la sol, cu care robotul se află în comunicație wireless.

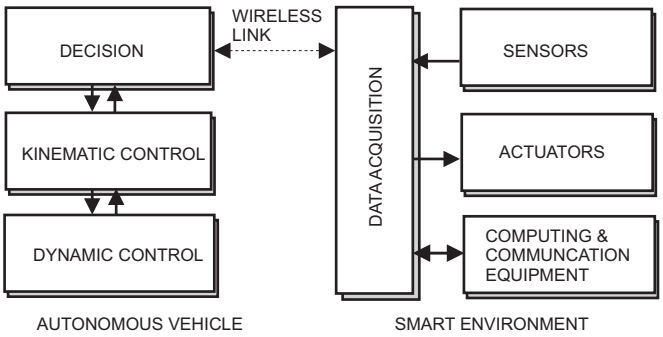


Fig. 2.3 Structura unui PRA, implementat conform definiției 1

Detalii privind aceasta abordare a conceptului de asistent robotic personal au fost publicate în [38].

2.3 Task-uri asociate cu navigația roboților autonomi

In 1991, Leonard si Durrant-Whyte ([39]) au sintetizat problemele legate de navigația roboților autonomi prin următoarele trei întrebări:

- Where am I?
- Where should I go?
- How should I get there, and not get hurt?

Figura 2.4 ilustrează modul în care aceste întrebări se suprapun peste conceptul clasic de *sistem de reglare cu reacție*.

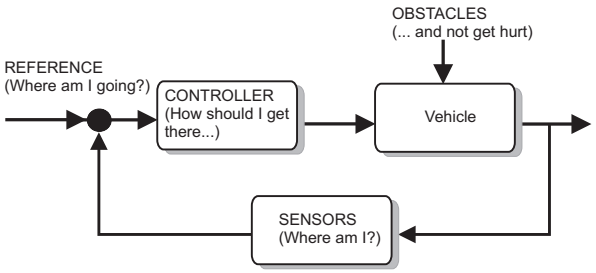


Fig. 2.4 Controlul navigației roboților autonomi din perspectiva teoriei sistemelor automate

In capitolele următoare se vor analiza pe rând task-urile asociate cu navigația roboților autonomi si modalitățile de implementare a acestora cu ajutorul unor microcontrollere încorporate.

2.4 Modelul cinematic al vehiculului cu doua roti motoare si tracțiune diferențială

Problema modelării cinematice a diverselor vehicule a fost extensiv tratată in literatură ([40],[41],[42],[43],[44]).

In cazul particular al roboților mobili cu două roți motoare si cu tracțiune diferențiala, cu notatiile din figura 2.5,

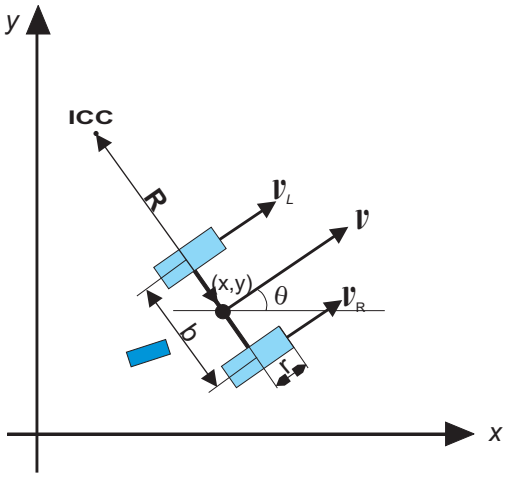


Fig. 2.5 Variabilele cinematice ale vehiculului cu două roți motoare si o roată directoare

unde *ICC* este centrul instantaneu de curbură a traiectoriei vehiculului, *R* este raza instantanee de curbură, *b* este distanța între planele roților motoare, iar ω este viteza unghiulară a vehiculului în raport cu punctul ICC. Presupunând că mişcarea roților este de rostogolire pură, fără alunecare, pe o suprafață plană, sunt valabile relațiile:

$$\omega(t) = \frac{v_R(t)}{R + b/2} = \frac{v_L(t)}{R - b/2} = \frac{v_R(t) - v_L(t)}{b} \qquad (2.1)$$

$$R = \frac{b}{2}\left(\frac{v_L(t) + v_R(t)}{v_L(t) - v_R(t)}\right) \qquad (2.2)$$

$$v(t) = \omega(t)R = \frac{v_R(t) + v_L(t)}{2} \qquad (2.3)$$

Iar ecuțiile cinematice ale vehiculului sunt:

$$\begin{aligned}\frac{dx(t)}{dt} &= v(t)\cos\theta(t)\\ \frac{dy(t)}{dt} &= v(t)\sin\theta(t)\\ \frac{d\theta(t)}{dt} &= \omega(t)\end{aligned} \tag{2.4}$$

ceea ce se poate scrie compact:

$$\frac{d}{dt}\begin{bmatrix}x\\ y\\ \theta\end{bmatrix} = \begin{bmatrix}\cos\theta & 0\\ \sin\theta & 0\\ 0 & 1\end{bmatrix}\begin{bmatrix}v\\ \omega\end{bmatrix} = \begin{bmatrix}\frac{1}{2}(v_R + v_L)\cos\theta\\ \frac{1}{2}(v_R + v_L)\sin\theta\\ (v_R - v_L)/b\end{bmatrix} \tag{2.5}$$

$$\frac{d}{dt}\begin{bmatrix}x\\ y\\ \theta\end{bmatrix} = \begin{bmatrix}\frac{1}{2}\cos\theta & \frac{1}{2}\cos\theta\\ \frac{1}{2}\sin\theta & \frac{1}{2}\sin\theta\\ -1/b & 1/b\end{bmatrix}\begin{bmatrix}v_R\\ v_L\end{bmatrix} \tag{2.6}$$

Ecuatia (2.6) reprezintă modelul cinematic al AGV si a fost folosită pentru simularea MATLAB a mişcării vehiculului.

3

Where Am I? Problema localizării roboților autonomi

Definiția 2

*Termenul **localizare** desemnează procesul de estimare a poziției si orientării unui vehicul autonom, în raport cu o reprezentare (hartă) a mediului.*

Conform *definitiei 2*, rezultă că task-urile asociate cu localizarea roboților autonomi se pot clasifica în:

- Task-uri asociate cu estimarea poziției robotului ("pose estimation");
- Task-uri asociate cu construcția hărtii mediului ("map acquisition", sau "map building") pornind de la informațiile furnizate de senzori.

Având în vedere că cercetările prezentate în această tezã se limitează la studii privind implementarea unui asistent robotic personal (PRA), conform *definitiei 1*, vom presupune ca harta mediului în care acesta evoluează este a-priori cunoscută şi, în consecință, nu vom aborda problemele de construcție a hărtii.

3.1 Clasificarea sistemelor de localizare

Din perspectiva modului de estimare a poziției, sistemele de localizare folosite in cazul roboților autonomi au fost clasificate în:

- *Sisteme de localizare relativă.* Acestea se folosesc in situațiile în care este disponibilă o estimare a poziției initiale a vehiculului, față de care se calculează un increment bazat pe informația furnizată de senzori. In această categorie se încadrează odometria şi sistemele de localizare inerțială.

- *Sistemele de localizare absolută.* În această categorie intră sistemele de localizare bazate pe:
    - Balize active (active beacons),
    - Recunoaşterea unor repere geografice (landmarks) naturale,
    - Recunoaşterea unor repere geografice artificiale,
    - Compararea informației de la senzori cu un model (hartă) geometric sau topologic al mediului (model matching localization).

O prezentare cuprinzătoare a senzorilor si metodelor de localizare cunoscute este disponibilă în ([45],[46]).

3.2 Sisteme de localizare relativă

Acestea sunt denumite sisteme de tip “dead reckoning”.

Pentru un vehicul cu modelul cinematic descris de (2.6), dacă se cunoaşte poziția la momentul t_k, $p_k = [x(t_k), y(t_k), \theta(t_k)]^T$, atunci poziția la un moment de timp oarecare $t>t_k$, se poate exprima:

$$\begin{aligned} x(t) &= x(t_k) + \int_{t_k}^{t} \frac{v_L(\tau) + v_R(\tau)}{2} \cos\theta(\tau) d\tau \\ y(t) &= y(t_k) + \int_{t_k}^{t} \frac{v_L(\tau) + v_R(\tau)}{2} \sin\theta(\tau) d\tau \\ \theta(t) &= \theta(t_k) + \int_{t_k}^{t} \frac{v_L(\tau) - v_R(\tau)}{b} d\tau \end{aligned} \qquad (3.1)$$

In cazul particular al odometriei, presupunând că la fiecare din roțile motoare se cuplează encodere cu rezoluția C_e (impulsuri per rotație) prin reductoare cu factorul de transmisie η, se defineste factorul de conversie c_m, care converteşte numărul de impulsuri in spațiu de translație:

$$c_m = \frac{\pi D_n}{\eta C_e} \qquad (3.2)$$

unde D_n este diametrul nominal al roților motoare.

Notand ΔN_{Lk} si ΔN_{Rk} variația (incrementul) numărului de impulsuri inregistrate de encodere, între momentele t_{k-1} si t_k, se poate calcula distanța parcursă de fiecare roată motoare în acest interval de timp:

$$\begin{aligned} \Delta S_{Lk} &= c_m \Delta N_{Lk} \\ \Delta S_{Rk} &= c_m \Delta N_{Rk} \end{aligned} \tag{3.3}$$

Atunci deplasarea liniară incrementală a centrului de masă al vehiculului se exprimă prin relația:

$$\Delta S_k = \frac{\Delta S_{Lk} + \Delta S_{Rk}}{2} \tag{3.4}$$

iar variația orientării este:

$$\Delta \theta_k = \frac{\Delta S_{Lk} - \Delta S_{Rk}}{b} \tag{3.5}$$

Cu aceste notatii, ecuațiile (3.1) devin:

$$\begin{aligned} x_k &= x_{k-1} + \Delta S_k \cos\theta_k \\ y_k &= y_{k-1} + \Delta S_k \sin\theta_k \\ \theta_k &= \theta_{k-1} + \Delta\theta_k \end{aligned} \tag{3.6}$$

Datorita simplității si costurilor reduse, sistemele de localizare bazate pe odometrie sunt prezente în aproape toate vehiculele autonome cunoscute. Principalul dezavantaj derivă din faptul că sunt vulnerabile la erori cumulative de măsurare.

Au fost descrise ([45]) erori odometrice *sistematice* provocate de:

- Diferențe între diametrele roților motoare;
- Diferențe între diametrul nominal si diametrul real al roților;
- Diferente între distanta nominala dintre planele roților şi distanța reală între acestea;
- Alinierea imperfectă a roților;
- Rezoluția limitată a encoderelor;

- Timpul de esantionare a encoderelor limitat ;

Erorile odometrice *nesistematice* provin de la:

- Traversarea unor suprafețe cu denivelări;
- Alunecări datorate traversarii unor suprafețe alunecoase, interacțiunii cu obstacole externe, unor forțe interne generate la nivelul roților castor, acceleratiilor prea mari, derapajului la curbe, sau contactului nepunctiform între roată şi sol.

Erorile odometrice sistematice sunt, în principiu, predictibile şi au fost descrişi ([47], [48]) algoritmi de estimare a gradului de incertitudine a masurătorilor odometrice. Conform acestei abordări, poziția reală a vehiculului se află, cu o probabilitate specificată, intr-o vecinătate a poziției estimate, delimitată de o *"elipsa de incertitudine"*, ca in figura 3.1.

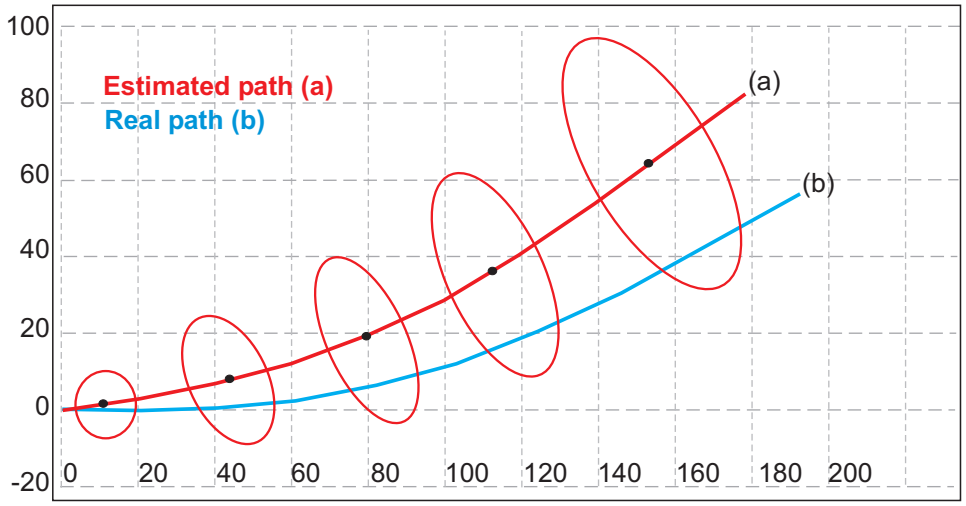


Fig. 3.1 Ilustrarea conceptului de elipsă de incertitudine

In figura 3.2 este prezentata traiectoria simulata a unui robot in conditiile unei diferente de doar 1% între diametrele nominale ale roților motoare. După parcurgerea a doar 10m in linie dreaptă, abaterea laterală înregistrată a fost de 1200mm, iar abaterea unghiulară a fost de 30 grade.

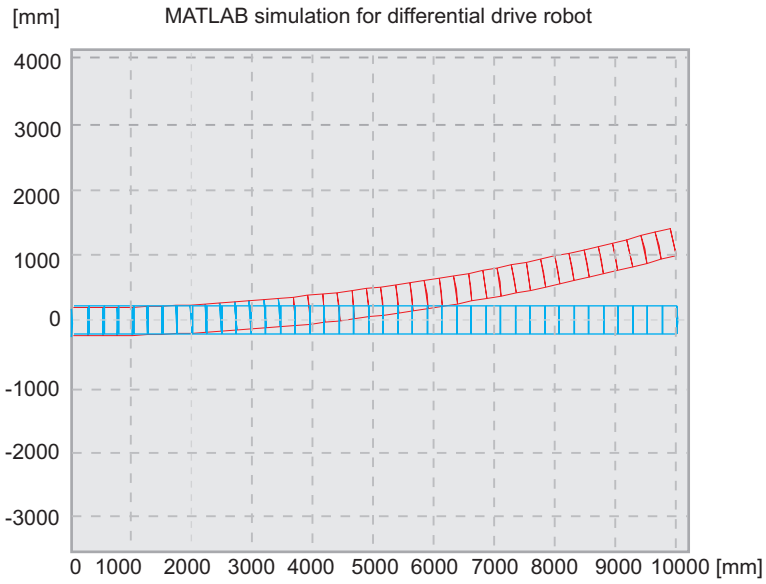


Fig. 3.2 Traiectoria simulată in cazul unei diferențe de 1% între diametrele roților

Au fost prezentate mai multe metode de estimare si reducere a erorilor odometrice sistematice (ex. [49], [50]), dar având în vedere că acestea sunt întotdeauna însoțite de erori nesistematice imprevizibile, utilitatea acestor metode este limitată.

Concluzii privind sistemele de localizare de tip dead-reckoning

Un robot care se bazeaza exclusiv pe dead reckoning va ajunge rapid să-şi piardă localizarea. Una din cele mai răspândite soluții constă in folosirea odometriei în combinație cu un sistem de localizare absolută, bazat pe estimarea poziției vehiculului autonom prin raportarea la un set cunoscut de referințe externe. Periodic, robotul executa secvențe de localizare absolută prin care isi actualizează coordonatele curente, iar între două actualizări se bazează pe odometrie pentru navigație.

Având în vedere că algoritmii de localizare absolută necesită, de regulă, un volum mai mare de calcul, problema frecvenței cu care se execută secvențele de actualizare este netrivială si rămâne deschisă problema determinării situațiilor în care măsurătorile odometrice au devenit inutilizabile şi se impune o secvența de localizare absolută.

3.3 Sisteme de localizare absolută

Sistemele de localizare absolută se bazează pe estimarea poziției robotului fața de un număr de repere externe, a caror poziție este cunoscută şi care pot fi identificate în mod univoc de senzorii aflați la bordul robotului.

Când localizarea se efectuază prin măsurarea unor distanțe, metoda se numeşte *trilaterație*, iar în cazul când se măsoară unghiurile, metoda se numeste *triangulație*.

Reperele folosite pentru localizare pot fi elemente ale mediului (de exemplu colțul unei încaperi, o uşa, etc.), sau pot fi "balize", special amplasate în acest scop, care emit semnale de identificare, uşor de interpretat de sistemul de senzori.

3.3.1 Metode de localizare bazate măsurarea distanțelor

Figura 3.3 ilustreaza principiul trilaterației. Daca poziția punctelor A, B şi C este cunoscută, şi sistemul de senzori ai robotului este capabil să măsoare distanțele d1, d2, d3, se poate calcula poziția punctului R, aflat la intersecția cercurilor cu centre în A, B şi C, şi cu razele d1, d2, d3.

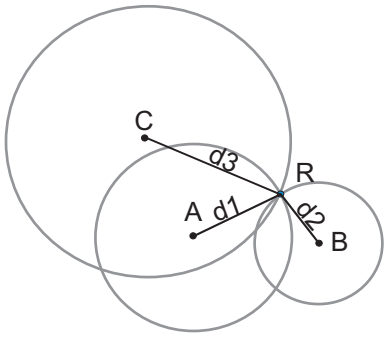


Fig. 3.3 Ilustrarea principiului trilaterației

Daca punctele A, B, C sunt coliniare, rezultă nedeterminarea ilustrată în figura 3.4, în care există două puncte R şi R' care satisfac condiția de apartenență simultană la cele trei cercuri.

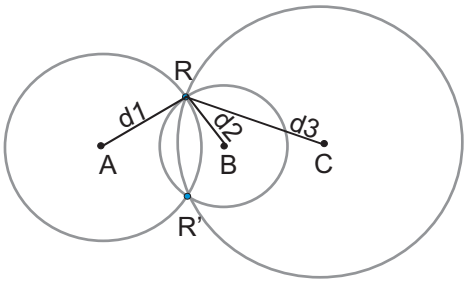


Fig. 3.4 Nedeterminare in cazul trilateratiei cu repere coliniare

Au fost descrise ([45], [51], [52], [53], [54], [55]) diverse metode pentru măsurarea efectivă a distanțelor între robot şi balize. Între acestea se numără:

- *Măsurarea timpului de propagare* a unui semnal specific între robotul mobil şi balizele folosite ca repere. Aceasta presupune fie că balizele sunt "active" şi că există un mecanism de sincronizare între emițatorii şi receptorii semnalului de referința (ca în GPS [56]), fie că sistemul de localizare al robotului este activ iar balizele reflectă sau retransmit semnalul de referință (ca în radar [57]). În cazul când se foloseste un semnal de referință luminos (o raza laser) există limitări la măsurarea distantelor mici, care conduc la timpi de propagare de ordinul picosecundelor, dificil de măsurat. Dacă semnalul de referința este sonor, distanța maximă care poate fi masurată este limitată la circa 20m, iar precizia măsurătorilor este mult mai mică decât la sistemele bazate pe laser.
- *Măsurarea diferenței de fază* între semnalul de referință emis de sistemul activ de localizare al robotului şi o fractiune a acestuia reflectată de țintă. Şi în acest caz există limitări în ce priveste distanța maximă măsurabilă, la circa 20m.
- O abordare interesantă este descrisă în [58] şi [59] şi se bazează pe *măsurarea intensităților câmpului electromagnetic* generat de o serie de puncte de acces Wireless Ethernet. Atenuarea undelor electromagnetice generate de acestea este proporțională cu distanța şi poate servi ca mijloc pentru estimarea distanței între

baliza radio şi receptorul mobil. Deocamdată nu există însa dispoziţive comerciale de localizare bazate pe acest principiu.

3.3.2 Metode de localizare absolută bazate pe măsurarea unghiurilor

Metodele de localizare derivate din triangulaţie sunt prezentate extensiv în [45], [60], [61], [62], [63], [64] iar informaţii despre senzorii folosibili pentru măsurarea orientării sunt disponibile în [65], [66].

Formularea problemei.

Fiind date trei repere în plan A, B, C, a căror poziţie este cunoscută, şi un robot R, capabil să măsoare valorile $\alpha_1, \alpha_2, \alpha_3$ ale unghiurilor formate de semidreptele RA, RB, RC faţa de orientarea curentă a robotului (vezi figura 3.5) se pune problema determinarii poziţiei robotului (x_R, y_R, θ_R) în coordonate carteziene.

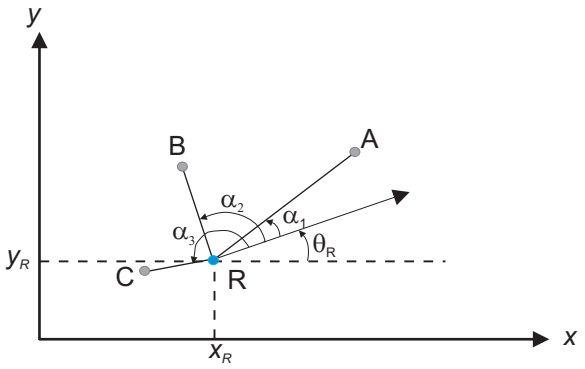


Fig. 3.5 Notaţii folosite pentru formularea problemei triangulaţiei

Cohen şi Loss în [64] descriu principalele metode de localizare prin triangulaţie şi compara performanţele acestora.

Metoda căutării iterative a orientării

Se bazează pe observaţia că, în situaţia în care se iau în considerare doar două repere, daca nu se cunoaste orientarea robotului θ_R, poziţia robotului (x_R, y_R) este nedeterminată, el putandu-se situa oriunde pe cercul determinat de punctele A, B, R (vezi figura 3.6). Invers, dacă orientarea θ_R luată în calcul este cea corectă,

rezultatele obtinute pentru (x_R, y_R) folosind oricare din perechile de puncte (A,B), (B,C) şi (C,A) sunt identice.

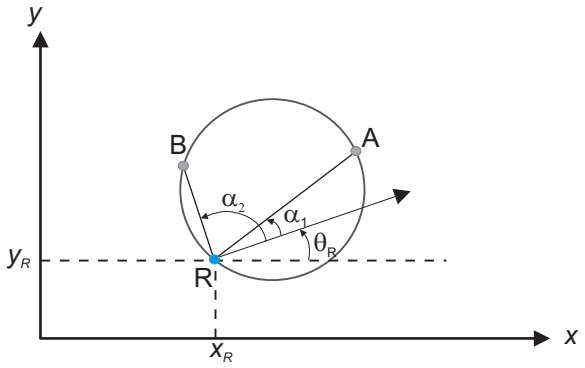


Fig. 3.6 Triangulație cu două repere

Această observație conduce la ideea de a parcurge iterativ toate orientările θ_{Ri} posibile, calculând de fiecare dată coordonatele (x_{Ri}, y_{Ri}) pentru toate perechile de repere (A,B), (B,C) şi (C,A). Se alege ca soluție, valoarea θ_R care conduce la un perimetru minim al triunghiului format de punctele (x_{Ri}, y_{Ri}) calculate pentru toate perechile de repere (A,B), (B,C) şi (C,A).

Metoda triangulației geometrice

Cu notatiile din figura 3.7:

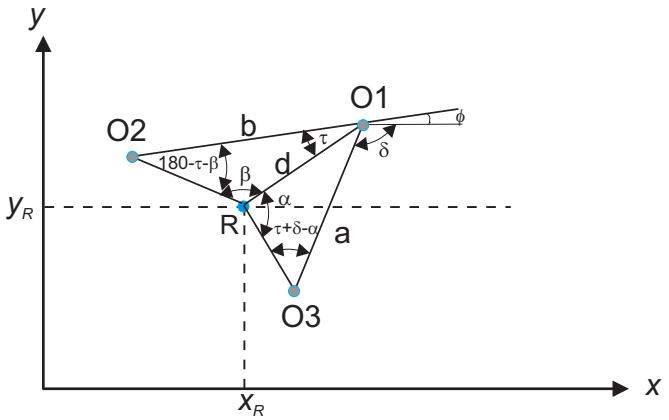


Fig. 3.7 Notații folosite pentru ilustrarea metodei triangulației geometrice

Se cunosc coordonatele reperelor O1, O2, O3 *(x1,y1), (x2,y2), (x3,y3)* şi orientările acestora relativ la robotul R, $\angle O_1, \angle O_2, \angle O_3$.

Ipoteze:

- Numerotarea reperelor se face în sens anti-orar;
- Unghiurile $\alpha = 360^o - \angle O_3 + \angle O_1$ si $\beta = \angle O_2 - \angle O_1$ trebuie să fie mai mici decât 180°. Dacă această condiție nu este indeplinită, se rotesc identificatoarele reperelor cu o poziție în sens anti-orar, până când condiția este îndeplinită.

Ținând sema de notațiile din fig. 3.7, în care, ϕ este unghiul între axa Ox şi dreapta determinată de punctele O1, O2, σ unghiul între axa Ox şi dreapta O1-O3, plus ϕ, şi $\gamma = \delta - \alpha$;

Cu notația:

$$p = \frac{a \sin \beta}{b \sin \alpha} \tag{3.7}$$

În [67]se demonstrează că:

$$\begin{aligned} \tau &= \arctan \frac{\sin \beta - p \sin \gamma}{p \cos \gamma - \cos \beta} \\ d &= \frac{b \sin(\tau + \beta)}{\sin \beta} \end{aligned} \tag{3.8}$$

$$\begin{aligned} x_R &= x_1 + d \cos(\phi + \tau) \\ y_R &= y_1 - d \sin(\phi + \tau) \\ \theta_R &= \phi + \tau - \angle O_1 \end{aligned} \tag{3.9}$$

Metoda de triangulație Newton-Raphson

Este o metodă iterativă de aproximare a poziției unui robot mobil, care porneşte de la ipoteza că este disponibilă o estimare inițială a poziției robotului, suficient de apropiată de poziția reală, obținută, de exemplu, prin dead reckoning. Dacă aproximarea initiala este “departe” de soluție, metoda Newton-Raphson nu dă rezultate.

Considerăm că această metodă se încadrează mai degrabă la metode de corectie a erorilor odometrice, decât ca metodă de localizare distinctă.

Metoda de triangulație cu calculul distantelor între robot şi balize

Cu notatiile din figura 3.8:

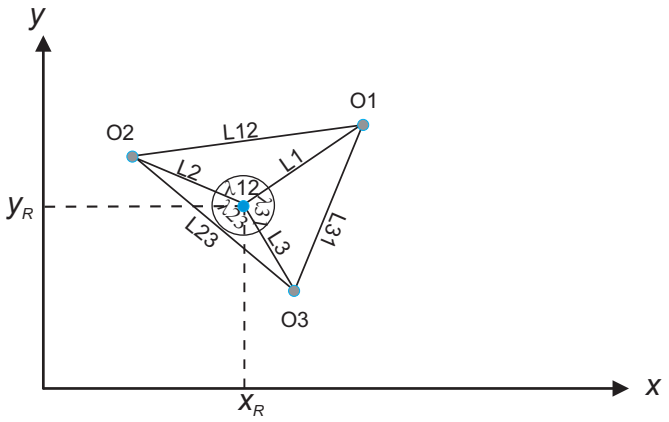


Fig. 3.8 Notații pentru ilustrarea metodei de triangulație cu calculul distanțelor

Distantele între balize se calculează ([67]):

$$\begin{cases} \lambda_{12} = \lambda_2 - \lambda_1 \\ \lambda_{23} = \lambda_3 - \lambda_2 \\ \lambda_{31} = \lambda_1 - \lambda_3 \end{cases} \tag{3.10}$$

$$\begin{cases} L^2_{12} = L^2_1 + L^2_2 - 2L_1L_2\cos\lambda_{12} \\ L^2_{23} = L^2_2 + L^2_3 - 2L_2L_3\cos\lambda_{23} \\ L^2_{31} = L^2_3 + L^2_1 - 2L_3L_1\cos\lambda_{31} \end{cases} \tag{3.11}$$

Iar coordonatele (x_R, y_R) se obtin rezolvand sistemul liniar:

$$\begin{bmatrix} 2(x_2 - x_1) & 2(y_2 - y_1) \\ 2(x_3 - x_1) & 2(y_3 - y_1) \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} L^2_1 - L^2_2 - x^2_1 + x^2_2 - y^2_1 + y^2_2 \\ L^2_1 - L^2_3 - x^2_1 + x^2_3 - y^2_1 + y^2_3 \end{bmatrix} \tag{3.12}$$

Cunoscând x_R, y_R, se poate calcula θ_R:

$$\theta_R = \arctan\left(\frac{y_1 - y_R}{x_1 - x_R}\right) - \lambda_1 \tag{3.13}$$

Alte metode de triangulație

Metoda geometrică bazata pe intersecția a două cercuri

Se bazează pe faptul că punctul în care se află robotul, împreună cu două din reperele folosite determină un cerc. Cu notațiile din figura 3.7, tripletele (R, O1, O2) şi (R, O1, O3) determină două cercuri care se intersectează în punctele R şi O1.

Coordonatele x_R, y_R se obțin rezolvând sistemul de ecuații care rezultă din condiția ca punctul R să aparțină ambelor cercuri, iar θ_R se poate obține cu o relație trigonometrică similară cu (3.13).

Metoda de triangulație cu mai mult de trei repere (Position Estimator)

Această metodă a fost descrisă de Betke şi Gurvits în [60] şi se bazează pe reprezentarea coordonatelor celor *n* repere ca numere complexe. Rezultă un sistem liniar, din care se determină coordonatele robotului.

Borenstein în [45] identifică următoarele avantaje ale acestei metode:

- Erorile mari, provenind de la identificarea gresită a unor repere, sau de la măsurarea gresită a unor unghiuri pot fi identificate şi eliminate;
- Durata de execuție variaza liniar cu numărul de repere folosite;
- Permite obținerea unor estimări corecte ale poziției robotului chiar în prezența zgomotului.

O analiza metodelor de localizare prin triangulație este prezentata extensiv în [64] şi [67]. Printre dezavantajele metodelor prezentate se numără:

- Practic toate metodele prezentate necesitã un volum mare de calcul, ceea ce le face dificil, sau chiar imposibil de implementat pe microcontrollere low cost/low power;
- In anumite configurații ale reperelor, produc soluții multiple, sau conduc la nedeterminări;

3.3.3 Alte metode de localizare

Metode de localizare care folosesc tehnologia RFID

Dezvoltată inițial ca o alternativă la codul de bare pentru identificarea obiectelor, tehnologia RFID (Radio Frequency Idenfification, [71],[72]) s-a dezvoltat accelerat în ultimii ani, fiind propuse zeci de aplicații, de la urmărirea bagajelor în aeroporturi, până la paşapoarte biometrice.

In esența, un “tag” RFID este o memorie nevolatilă care poate fi accesată fără contact, prin intermediul undelor electromagnetice. Principalul avantaj al acestora este că majoritatea sunt “pasive”, în sensul că nu necesită o sursă proprie de alimentare. Alimentarea chipului de memorie se face doar pe durata transferului de date, captând o fracțiune din energia câmpului electromagnetic emis de un cititor (reader).

Transferul datelor se face cu ajutorul modulației de sarcina (load modulation), (vezi figura 3.9) conectând o sarcină variabilă la circuitul acordat al antenei receptorului. Variația sarcinii conduce la modularea în amplitudine (ASK – Amplitude Shift Key) a semnalului emis de reader. Cum între reader şi tag există un cuplaj inductiv strâns, variația amplitudinii purtătoarei este detectabilă la nivelul reader-ului, care poate reconstitui datele transmise de tag.

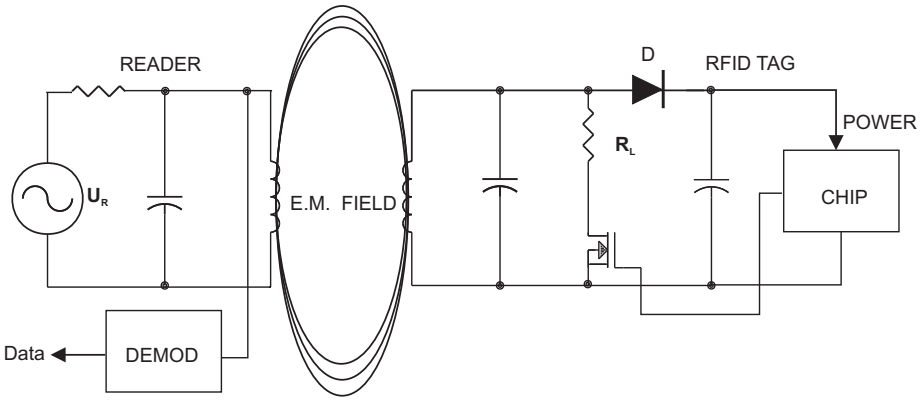


Fig. 3.9 Principiul de funcționare al unui canal RFID

In tabelul 3.1 sunt prezentate frecvențele de operare uzuale şi standardele care definesc caracteristicile diverselor sisteme RFID.

Tabelul 3.1 Frecvențe şi standarde RFID

<table>
<tr><td rowspan="5">Standard</td><td colspan="5">Frecvențe de operare</td></tr>
<tr><td>LF
125-134KHz</td><td>HF
13.56MHz</td><td>HF
433MHz</td><td>UHF
866-930MHz</td><td>UHF
2.4GHz</td></tr>
<tr><td>ISO-11784</td><td>ISO-14443</td><td rowspan="3">ISO-18000-7</td><td>ISO-18000-6a</td><td rowspan="2">ISO-18000-4</td></tr>
<tr><td>ISO-18000-2</td><td>ISO-15693</td><td>ISO-18000-6b</td></tr>
<tr><td>ISO18000-2B</td><td>ISO-18000-3</td><td>ISO-18000-6c</td><td>ISO-24730-2</td></tr>
</table>

In functie de distanța maxima între cititor şi tag la care se pot transmite date, tag-urile RFID se clafifică în:

- Proximity tags (distanța de operare între 0 şi 100mm)
- Vicinity tags (distanța de operare între 100 şi 1000mm)
- Long range tags (distanța de operare 1-15m). Acestea sunt, de obicei active, în sensul ca dispun de o baterie proprie pentru alimentare.

Din punct de vedere al tipului de memorie internă folosită, tag-urile RFID pot fi read-only, situație în care contin o memorie ROM cu un cod unic de identificare, sau read-write, caz în care memoria internă este EEPROM sau FRAM.

Capacitatea memoriei interne variază, pentru tag-urile uzuale, între cativa octeți şi 128 kiloocteți.

Timpul de lectură pentru tag-urile uzuale este de ordinul milisecundelor.

Majoritatea sistemelor RFID actuale folosesc un mecanism de detectare a coliziunilor la transferul datelor între cititor şi tag. În consecință, în situațiile în care mai multe tag-uri se afla în zona de recunoaştere a unui reader (vezi figura 3.10) *acestea vor fi citite pe rand, sau nu va fi citit nici unul.*

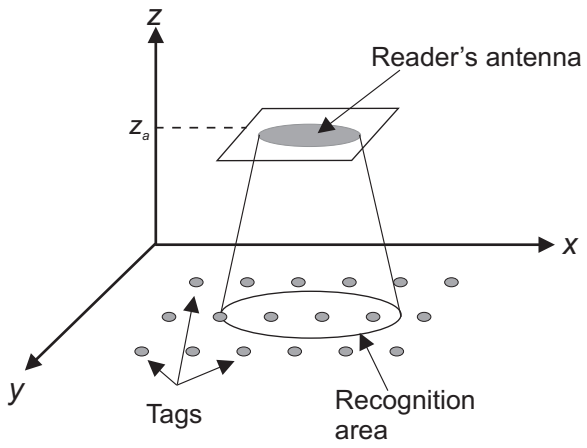


Fig. 3.10 Modelul zonei de recunoaştere intr-un sistem RFID

In ultimii ani (dupa 2004) au fost prezentate câteva experimente care vizează folosirea tehnologiei RFID pentru localizarea robotuilor mobili. Soluțiile propuse diferă în funcție de distanța de citire a tag-urilor şi de tipul de tag-uri folosite.

Hahnel et. al. în [73] propun un sistem de localizare bazat pe vicinity tags read-only, în care se foloseşte o abordare probabilistică pentru corecția poziției raportată de sistemul odometric folosind informația obtinută de la un set redus de tag-uri RFID montate pe pereții incintei.

Kulyukin et. al în [74] propun un sistem asemănător, bazat pe metoda câmpului de potential, pentru implementarea unui sistem de ghidare a persoanelor cu deficiente de vedere. Şi aici, se folosesc vicinity tags, read-only, montate pe pereți.

O abordare asemănătoare este prezentată în [75].

Ulterior, Kulyukin revine în [76] şi propune un sistem cu tag-uri de proximitate read-only, montate pe podea.

Park şi Hashimoto propun în [77] o metodă de localizare bazată exclusiv pe RFID. Tag-urile sunt montate în podea, în asa fel încât un singur tag se afla la un moment dat în zona de recunoastere a cititorului. Se masoară timpul de citire al tag-urilor şi această informație este folosită pentru calcularea orientării.

În [78] şi [79] se propun soluții bazate pe considerații legate de propagarea undelor electromagnetice pentru determinarea orientării robotului în raport cu tag-urile citite.

În toate abordările menționate, poziția tag-urilor este a-priori cunoscută şi se încearcă determinarea poziției robotului în raport cu aceste repere. În [80], configurația inițială a tag-urilor este necunoscută, dar este învățată de un robot echipat cu sisteme laser de localizare. Ulterior informația achiziționată prin învățare despre poziția tag-urilor în mediu este transferată unor roboți mai simpli.

În [81] şi [82] se propun soluții care folosesc tag-uri read-write, încorporate în podea, în care se memorează coordonatele poziției tag-ului. Pentru estimarea poziției unghiulare şi pentru aproximarea deplasărilor între nodurile rețelei de tag-uri se foloseşte sistemul odometric.

În sfârşit, soluția propusă în [83] vizează îmbunătățirea rezoluției de localizare a automobilelor prin GPS, distribuind tag-uri RFID pe şosele.

3.4 Concluzii privind metodele de localizare prezentate

Borenstein în [45], în urma unei analize extensive a sistemelor de localizare cunoscute, formulează următoarea concluzie:

“Perhaps the most important result from surveying the vast body of literature on mobile robot positioning is that to date there is no truly elegant soluțion for the problem.”

In general, se poate observa ca soluțiile bazate exclusiv pe un singur tip de senzor conduc fie la algoritmi care implică volume foarte mari de calcule, fie la nedeterminări inacceptabile. Sunt de preferat soluțiile care fuzionează informațiile de la mai multe tipuri de senzori.

3.5 Contribuții la rezolvarea problemei localizarii roboților mobili

3.5.1 O metodă de identificare a balizelor ultrason

În toate metodele de localizare bazate pe trilaterație şi triangulație descrise în paragrafele anterioare se presupune că robotul cunoaşte a-priori poziția reperelor fixe folosite şi că dispune de un mijloc de a identifica reperul asupra căruia se fac măsurători la un moment dat.

Chiar în cazul folosirii unor senzori laser de înaltă performanță ([84]), care pot măsură distante de până la 200m cu o rezoluție de 1mm, problema identificării obiectului față de care se măsoară distanța rămâne deschisă.

Balizele ultrason au avantajul simplității şi al costului redus şi, din acest motiv, au fost printre primele soluții folosite în practică ([39]). Pentru identificarea acestora a fost propus un sistem de activare selectivă ([85]) în care, o stație-bază activează pe rând balizele, emițând simultan un semnal radio, care conține un cod de identificare al balizei. Robotul recepționează semnalul radio şi porneşte un timer cu ajutorul căruia măsoară timpul de propagare t_p a undei sonore între baliza fixă şi receptorul aflat la bordul robotului (vezi figura 3.11), ceea ce permite calcularea simplă a distanței între baliza şi receptor.

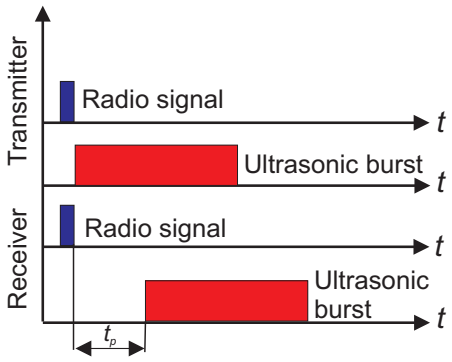


Fig. 3.11 Activarea selectivă a balizelor cu identificator radio

Un sistem bazat pe acest principiu, realizat de I.S. Robotics, citat de Borenstein în [45], a realizat localizarea unui robot mobil (Genghis) într-un perimetru de 9,1x9,1m cu o precizie de 0,5" (12,7mm).

Dezavantajele acestei metode:

1. Foloseşte un canal radio între robot şi echipamentul de la sol, canal care, în practică, este necesar pentru alte scopuri.
2. Metoda este aplicabilă doar în sistemele de localizare bazate pe măsurarea distantelor. Metodele de localizare care implica măsurarea orientării robotului fața de o sursa sonoră nu beneficiaza de acest mecanism de identificare a balizelor (de exemplu, metodele bazate pe măsurarea timpului de intarziere interaurala – ITD, intr-un sistem biomimetic cu două microfoane captoare, cum este cel descris în [68]).

Soluția propusă mai jos elimina ambele dezavantaje enumerate, codificând informația de identificare a balizelor printr-un semnal DTMF (Dual Tone Multi Frequency), modulat în amplitudine peste purtatoarea de frecvența ultrasonoră.

Sistemul de codare DTMF a fost propus în 1960 ([86]) pentru transmisia unor volume reduse de informație digitala pe infrastructura de telefonie analogică, foarte afectată de zgomot. Ulterior au fost descrise numeroase aplicații posibile ([87], [88]).

Un semnal DTMF este o suma de două semnale sinusoidale:

$$y = A\sin\omega_1 t + A\sin\omega_2 t \qquad (3.14)$$

iar informația digitală codata de acest semnal este determinată de perechea de valori (ω_1, ω_2).

Frecvențele corespunzătoare pulsațiilor (ω_1, ω_2) au fost standardizate în plaja [697Hz-1633Hz]. În prezent, sunt disponibile circuite integrate monolitice, care implementeaza funcțiile de encoder/decoder DTMF (ex. Mitel MT8870/MT8880).

Soluția propusă de identificare a balizelor este prezentata schematic în figurile 3.12 (structura balizelor) şi 3.13 (structura receptorului).

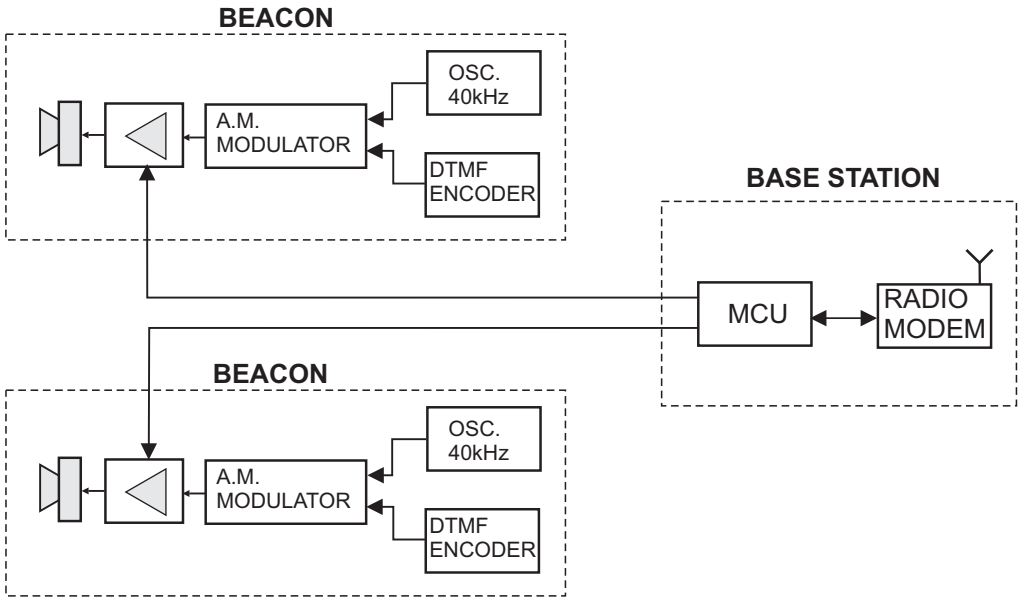


Fig. 3.12 Structura balizelor ultrason cu identificare DTMF

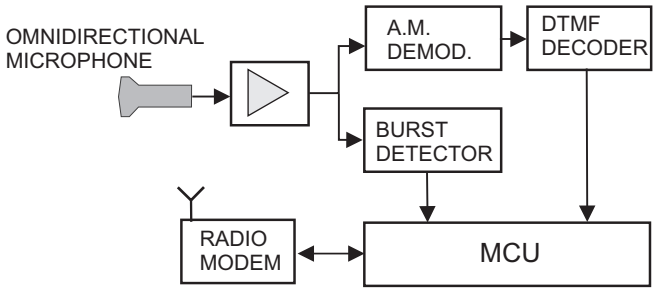


Fig. 3.13 Structura receptorului pentru identificarea balizelor

La nivelul stației-bază, un microcontroller monitorizează, prin intermediul unui radio-modem, comunicația între robot şi echipamentul de control de la sol. De fiecare dată când se emite un pachet de date catre robot, se comandă succesiv fiecare baliză, pentru emisia unui burst de 40KHz, cu durata de 100ms, modulat A.M. cu semnalul DTMF corespunzator codului de identificare al balizei.

La receptor, semnalul audio este captat de un microfon omnidirecțional, amplificat şi prelucrat de circuitele adiționale, care extrag codul DTMF al balizei şi generează o cerere de întrerupere la microcontroller, marcând momentul precis al detectarii burst-

ului ultrason. Microcontrollerul măsoară timpul scurs între momentul recepției pachetului de date şi momentul detectării burst-ului ultrasonic de 40KHz. Nu este necesară modificarea protocolului de comunicatie cu echipamentul de la sol pentru că identificatorul balizei este extras din semnalul DTMF. Semnalul de STROBE generat de decodorul DTMF pentru transferul codului de identificare reprezinta o validare suplimentară a măsurării, care permite rejectia burst-urilor afectate de zgomot.

Metoda de identificare a balizelor ultrason descrisă mai sus poate fi aplicată şi în sistemele de localizare bazate pe triangulație. O soluție simplă şi elegantă de determinare a poziției unghiulare relative a robotului față de o sursă sonoră este prezentată în [68]. Aceasta se bazează pe folosirea a două microfoane omnidirectionale pentru captarea semnalului sonor, în mod similar cu perceptia binaurală la mamifere. Determinarea directiei sursei sonore (azimutul) se face măsurând întârzierea interaurală a semnalului sonor (ITD – Interaural Time Difference).

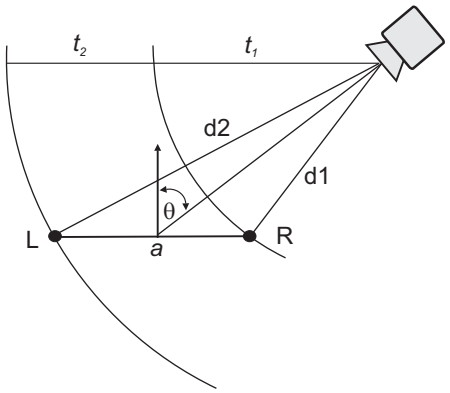


Fig. 3.14 Determinarea azimutului sursei de semnal pe baza măsurării ITD

Cu notațiile din figura 3.14, unde L şi R sunt doi senzori (Left si Right), cunoscând timpul de propagare interaurală Δt, se poate calcula azimutul sursei sonore fața de orientarea curentă a robotului ([68]):

$$\theta = \arcsin(\frac{c\Delta t}{a}) \qquad (3.15)$$

unde *c* este viteza sunetului, iar *a* este distanța între cele două microfoane captoare.

ITD se poate măsura cu un circuit relativ simplu, cum este cel prezentat în figura 3.15.

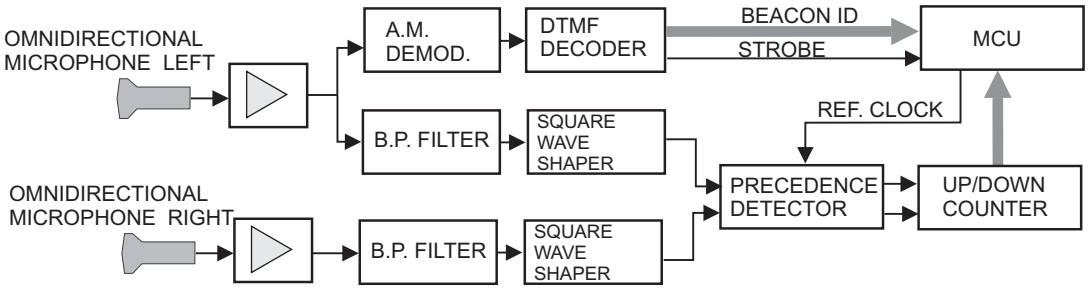


Fig. 3.15 Exemplu de circuit pentru măsurarea Interaural Time Difference ITD

Fața de metoda descrisă în [68], de calcul a ITD prin cross-corelarea software a semnalelor captate de microfoane, circuitul din figura 3.15 are avantajul ca degrevează microcontrollerul de o parte din calculele necesare localizării si, în acelaşi timp, oferă o modalitate de identificare a balizelor.

Evident, structurile hardware descrise în figurile 3.13 şi 3.15 pot fi combinate în asa fel încât să determine simultan şi azimutul şi distanța la care se afla baliza. Dacă orientarea robotului ϕ se determina cu un dispoziţiv dedicat (girocompas, sau busolă electronică [45]), coordonatele (x_R,y_R) ale robotului se pot determina exterm de simplu folosind o singura baliza ca referința.

Cu notațiile din figura 3.16, cunoscând coordonatele balizei (x_A,y_A) şi masurand orientarea robotului ϕ, distanța până la baliza d, şi azimutul balizei fața de robot θ, rezultă coordonatele robotului:

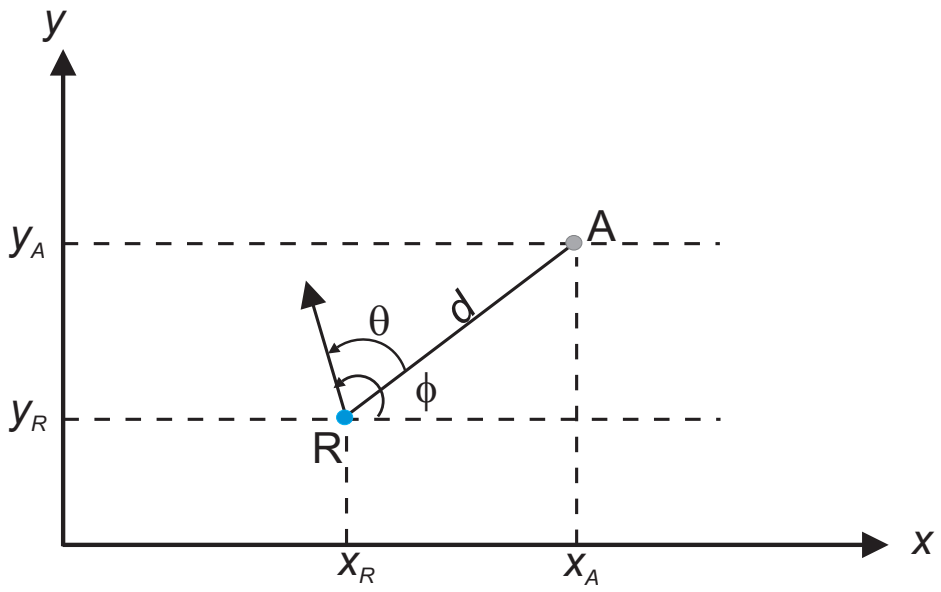


Fig. 3.16 Localizare în raport cu o singură baliză

$$x_R = x_A - d\cos(\phi - \theta)$$
$$y_R = y_A - d\sin(\phi - \theta) \qquad (3.16)$$

Dezambiguizarea față-spate la determinarea azimutului balizei se poate face folosind informația furnizata de sistemul odometric.

3.5.2 Metoda de localizare bazată pe tehnologia RFID

Metoda de localizare propusă, deriva din metodele propuse în [81] şi [82], cu următoarele precizări:

a. Tag-urile folosite sunt de proximitate, read-write, amplasate în podea şi fiecare tag stochează coordonatele absolute ale punctului unde se află amplasat.
b. Tag-urile sunt amplasate intr-un grid în aşa fel încât cel mult un tag sa se afle la un moment dat în zona de recunoştere a cititorului (vezi figura 3.17).
c. Poziția unghiulara a vehiculului autonom se măsoară cu o busola electronica.
d. Vehiculul autonom dispune de un sistem odometric de localizare.
e. Citirea tag-urilor se face cu ajutorul a două cititoare montate lateral. Această opțiune reduce probabilitatea situațiilor în care nici un tag nu este vizibil la un moment dat.

f. De fiecare dată când se citeste un tag, coordonatele absolute ale acestuia şi valoarea poziției unghiulare a robotului, raportată de busola electronică sunt folosite pentru reinițializarea sistemului odometric. În acest mod se evita acumularea erorilor sistematice specifice odometriei.

g. În intervalele dintre două citiri ale tag-urilor, se foloseşte sistemul odometric pentru estimarea poziției.

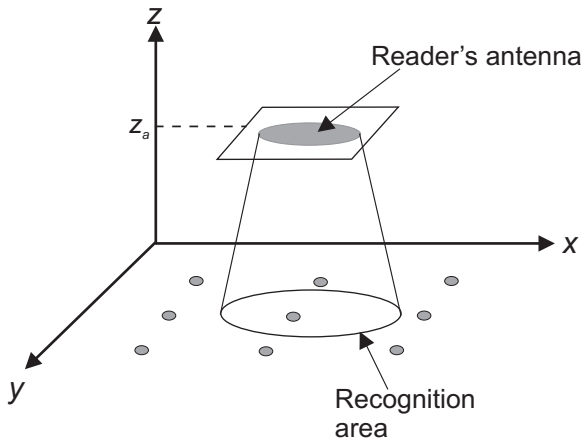


Fig. 3.17 Amplasarea tag-urilor RFID pentru localizarea roboților mobili

Metoda descrisă mai sus are câteva avantaje notabile:

- Degrevează complet microcontrollerul de navigatie de sarcina computațională asociată cu localizarea;
- Este o soluție ieftină şi fiabilă;
- Erorile de localizare sunt constante;
- Tag-urile RFID distribuite în mediu pot stoca informații suplimentare pe lângă coordonatele punctelor în care se află, de exemplu informații care pot folosi la definirea unor trasee, sau la coordonarea activității unor formațiuni de roboti (vezi Capitolul 5).

3.6 Concluzii privind localizarea PRA

- Problema localizării roboților mobili se simplifică drastic dacă admitem ipoteza că mediul în care aceştia evoluează este manipulat în sensul amplasării unor repere artificiale, special proiectate.
- Fuzionarea informațiilor furnizate de senzori de mai multe tipuri contribuie la reducerea volumului de calcule asociate cu problema localizării.

4

Where Should I Go? Selecția țintelor de navigație

Ming Xie propune în [89] o definiție generală a unui sistem automat ca fiind *"acel sistem, în care interactiunea între un subsistem de decizie (decision making) şi un subsistem efector (action taking), se desfaşoară fără intervenția unui operator uman".*

Din această perspectivă, procesul de elaborare a deciziilor ocupă un loc central în funcționarea oricărui sistem automat (vezi figura 4.1).

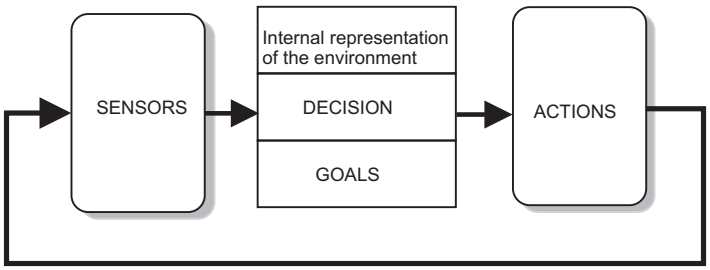


Fig. 4.1 Reprezentarea unui sistem automat ca sistem de decizie şi actiune

În cazul roboților, stabilirea obiectivelor, regulile de instanțiere a reprezentărilor interne despre mediu, precum şi condițiile de selectare a acțiunilor se definesc prin program.

Primele încercări de creare a unor limbaje de programare dedicate pentru programarea roboților datează din 1973, când a fost propus WAVE ([90]), în contextul apariției manipulatoarelor robotice de uz industrial.

Ulterior, producătorii de roboți industriali au creat limbaje de programare specifice pentru aceştia, inspirate de limbajele de programare pentru computere. De exemplu, limbajul RAPID, folosit de roboții produşi de ABB ([91]) este derivat din limbajul

Pascal, iar MBA4, sau MELFA BASIC IV, ([92]), folosit de roboții MELFA de la Mitsubishi, este o extensie a limbajului Basic.

Preocupările pentru uniformizarea programării roboților industriali sunt reduse. Există doar unele încercări de realizare a unor translatoare pentru traducerea automată a programelor dintr-un limbaj în altul ([93]).

Programarea roboților autonomi comportă o serie de dificultăți ([94]) care derivă, în primul rând, din caracterul dinamic şi uneori imprevizibil al mediului în care aceştia evoluează, din faptul că se foloseşte o mare diversitate de senzori, cât şi din complexitatea task-urilor pe care le au de executat.

În cazul particular al roboților de serviciu, în care se încadrează şi asistenții robotici personali, apare o cerință suplimentară dificil de realizat şi anume aceea că programarea roboților trebuie să fie accesibilă unor utilizatori cu cunoştințe tehnice elementare.

4.1 Abordări cunoscute în domeniul programării roboților autonomi

Cea mai răspândita abordare a programarii roboților autonomi se bazează pe crearea unor *extensii ale unor limbaje de programare cunoscute* (Pascal, LISP, ALGOL, BASIC, C++, etc.).

In această categorie se încadrează, de exemplu, ARIA – Advanced Robot Interface Application ([11]), RIPL - Robot Independent Programming Language ([95]), RCL -The Robot Command Language ([96]), FDTL - Fuzzy Decision Tree Language ([97]), GOLOG – alGOL în LOGic ([98], [99]) şi TDL - A Task Description Language for Robot Control ([100]).

Tot în aceasta categorie se pot încadra soluțiile de programare a roboților bazate pe XML – Extensible Markup Language ([101]), de exemplu RoboML ([102]). Soluțiile bazate pe XML sunt deosebit de interesante în contextul unei posibile interacțiuni

om-robot mediate de Internet, având în vedere ca toate browserele actuale suportă XML şi numeroase alte aplicații software sunt echipate cu plug-in-uri XML.

Limbaje de programare vizuala

În această abordare, primitivele limbajului sunt reprezentate prin pictograme grafice, în principiu usor de selectat dintr-o bibliotecă. În practică însă, biblioteca de simboluri poate atinge dimensiuni foarte mari, ceea ce îngreunează mult definirea task-urilor.

De exemplu, Lingraphica ([103]) se bazează pe o bibliotecă cu peste 2000 de pictograme. Limbajul grafic PILOT ([104]) foloseşte un număr redus de primitive, dar parametrizarea e mult mai complexă şi mai dificilă pentru utilizatorii neexperimentati.

Tehnici de programare a roboților folosind limbajul natural

O soluție remarcabilă este propusa de Drews şi Fromm în [105] sub denumirea FSTN – Finite State Transducer Network. Aceasta se bazează pe un pachet software comercial de recunoastere a vorbirii (Dragon Naturally Speaking [106]). Propozițiile recunoscute astfel sunt interpretate din punct de vedere semantic şi sintactic şi apoi convertite în comenzi directe catre robot. Sistemul recunoaste un vocabular de 500 de cuvinte şi este capabil sa formuleze intrebări atunci când nu înțelege comenzile vocale primite.

Lauria et al. în [107] şi [108] propun o metodă, denumita IBL – Instructions Based Learning, bazată pe un set redus de primitive apelabile prin comenzi vocale simple, de genul “go_to <landmark>”, “turn_left”, “follow_known_route_to <landmark>”, etc. Robotul este capabil să învete secvențe de astfel de comenzi şi sa asimileze în acest mod noi cunostinte despre mediu, de exemplu sa învete rute noi.

4.2 Contribuții la rezolvarea problemei programării roboților autonomi

4.2.1 Formularea problemei

Considerând un sistem robotic cu structura definită în figura 1.1, aflat în interacțiune cu un set de senzori şi actuatoare distribuite intr-un mediu inteligent, în scopul implementarii conceptului de asistent robotic personal (PRA) pentru persoane în vârstă sau cu diverse dizabilități, şi presupunand ca nivelele inferioare de control (vezi figura 2.2) realizează conducerea cinematică şi dinamică a vehiculului autonom, se pune problema dezvoltarii unui limbaj de programare pentru modulul de decizie, care sa permită realizarea următoarelor obiective:

a) Crearea şi actualizarea unei reprezentări interne asupra mediului, prin interogarea senzorilor;
b) Definirea unui set de ț*inte* (goals) de navigatie;
c) Definirea unui set de *acțiuni* posibile, care constau fie în comanda actuatoarelor din mediu, fie în deplasarea vehiculului către țintele de navigație definite;
d) Definirea condițiilor în care se *decide* execuția diverselor actiuni.

Având în vedere structura de PRA propusă, bazată pe structuri de microcontrollere încorporate (embedded), se impun următoarele cerințe suplimentare:

- Programele generate în acest mod trebuie executate în timp real, rămânând compatibile cu resursele limitate de memorie şi putere de calcul ale microcontrollerelor uzuale;
- Limbajul de programare trebuie sa fie cât mai simplu, încât sa poată fi folosit de utilizatori cu cunoştințe elementare de programare.

4.2.2 Descrierea generală a soluției propuse

In general, interacțiunea robotului cu mediul poate fi descrisa prin propoziții de tipul următor:

IF((condition$_1$)&&(condition$_2$)&&...&&(condition$_N$)) THEN MOVE_TO GOAL$_k$

sau:

IF((condition$_1$)&&(condition$_2$)&&...&&(condition$_N$)) THEN Activate_output$_k$

unde *(condition$_j$)* este o funcție logică de intrările şi ieşirile sistemului.

Limbajul de programare propus, pe care l-am denumit RDPL (Robot Decision Programming Language), se înscrie în categoria limbajelor de programare a roboților autonomi, create ca extensii ale unor limbaje de programare cunoscute, fiind derivat din limbajul descris de standardul IEC61131-3 ([109]), destinat programării PLC-urilor.

RDPL este un limbaj compilat, bazat pe un set redus de primitive, cu sintaxa generală următoare:

FNAME *dest, op1, op2, op3, op4* (4.1)

unde FNAME este numele simbolic al funcției, *dest* este destinația rezultatului operației indicate de funcție, iar *op1, .., op4* sunt operanzii implicați în evaluarea funcției. De exemplu, funcția:

AND O1, *I1, I2, Y5, Y25*

are ca efect activarea ieşirii O1, daca intrările I1 şi I2 sunt active, iar variabilele Y5 şi Y25 au valoarea logică TRUE.

Programul sursă este un fisier text, care conține o secvența de propoziții cu structura definită de (4.1). Prin compilare, se obține un fisier binar, care conține codul executabil, ce trebuie transferat intr-o memorie nevolatilă situată la nivelul microcontrollerului de decizie.

Execuția programului se face conform schemei logice prezentate în figura 4.2.

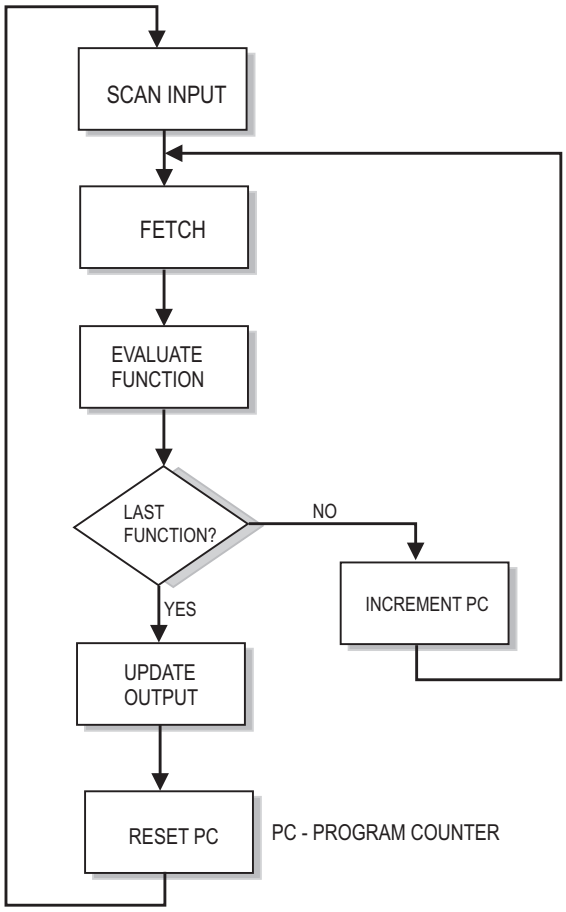


Fig. 4.2 Schema logica de execuție a unui program RDPL

Fiecare ciclu de execuție a programului începe cu o secvență de actualizare a unui set de variabile interne asociate cu intrările în sistem şi se încheie cu actualizarea ieşirilor. Între aceste două momente, se evaluează pe rând toate funcțiile, în ordinea în care au fost definite în program.

Față de un PLC convențional, sistemul descris are următoarele particularități:

- Intrările şi ieşirile sunt distribuite, fiind asociate cu senzorii şi actuatoarele amplasate (preponderent) în mediu.
- Un set limitat de intrări este asociat cu un număr comenzi vocale, de genul STOP, MOVE, LEFT, RIGHT. În aceasta abordare, recunoaşterea unei comenzi vocale, se traduce intr-o tranziție a intrării digitale asociate cu comanda vocală respectivă.

- O categorie specială de ieşiri din sistem o reprezintă comenzile de mişcare a vehiculului autonom. Acestea se generează automat şi se transmit subsistemului de conducere cinematică la detectarea unui front selectat al unor variabile logice. Coordonatele punctului țintă sunt transmise subsistemului de control cinematic odată cu comanda de mişcare.

4.2.3 Detalii privind implementarea RDPL

4.2.3.1 Crearea şi menținerea unei reprezentări interne asupra mediului

Acest obiectiv se realizează definind un set de variabile asociate în mod univoc cu intrările şi ieşirile din sistem. Un task de background interoghează periodic blocul "Data Acquisition" (fig. 1.1) dupa un protocol compatibil cu modulele de achizitie de date folosite (MODBUS în experimentele noastre) şi actualizează aceste variabile.

Pentru a permite codificarea fronturilor şi nivelelor semnalelor digitale, se alocă variabile de tip byte pentru intrările digitale. La fiecare actualizare, valoarea curentă a intrării este shiftată în bitul cel mai puțin semnificativ al octetului asociat, ca în figura 4.3. Acelasi mecanism de actualizare se aplică pentru toate variabilele de tip binar din sistem.

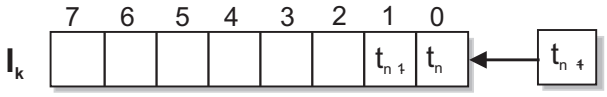


Fig. 4.3 Mecanismul de actualizare a intrărilor digitale

In acest mod, biții cei mai puțin semnificativi (bitii b0 şi b1) ai octetului asociat cu o variabilă binară indică faptul cã respectiva variabilă a avut sau nu tranziție de la ultima esantionare, precum şi tipul acestei tranziții – front crescător sau căzător.

Comenzile vocale sunt tratate ca intrări digitale în sistem. Recunoaşterea propriu-zisă a comenzilor se face la nivelul unui computer, amplasat în mediu, care rulează o

aplicație comercială de recunoaştere a vorbirii, de exemplu Dragon Naturally Speaking ([106]).

La recunoaşterea unei comenzi vocale, computerul emite pe un port serial un string de identificare, asociat în mod univoc cu comanda respectivă. Acesta este recepționat de un modul de conversie de protocol, care raportează datele în format adecvat catre microcontrollerul de decizie (vezi figura 4.4).

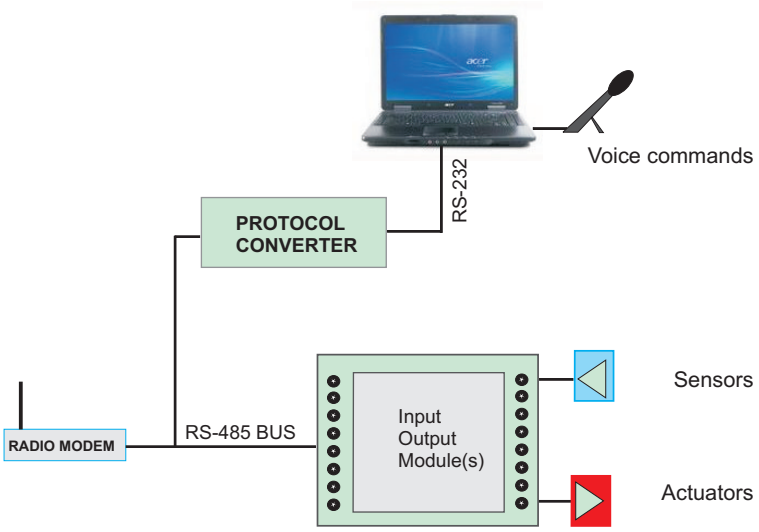


Fig. 4.4 Structura modulului “Data Acquisition” incluând comenzile vocale

Sistemul descris are avantajul că permite tratarea comenzilor vocale în mod absolut similar cu oricare intrare digitală, asigurând maximum de flexibilitate şi simplificând considerabil interpretarea comenzilor vocale. Dezavantajul este că numărul maxim al comenzilor vocale este limitat.

4.2.3.2 Definirea țintelor de navigație

Coordonatele țintelor de navigație sunt furnizate ca parametru în comanda de mişcare. Sintaxa generală a comenzilor de mişcare este definita de funcția MOVE:

```
MOVE        TRIGGER, GOAL_X,GOAL_Y, GOAL_HEADING, EDGE
```

unde `TRIGGER` este denumirea simbolică a variabilei care generează comanda de mişcare, *(`GOAL_X, GOAL_Y, GOAL_HEADING`)* sunt coordonatele țintei, iar `EDGE` este

frontul semnalului TRIGGER care declansează efectiv generarea comenzii de mişcare catre vehiculul autonom. Coordonatele țintei sunt date intr-un sistem de referință cartezian şi sunt exprimate în milimetri şi grade.

Subsistemul de navigație este responsabil de selecția unei rute (predefinite) şi de conducerea vehiculului la ținta specificată, cu ocolirea eventualelor obstacole. Țintele specificate trebuie sa fie posibil de atins (reachable) şi sa apartina unei rute predefinite.

4.2.3.3 Definirea acțiunilor

Pe lângă comenzile de mişcare a vehiculului autonom, definirea acțiunilor pe care robotul le poate exercita asupra mediului se face prin selecția actuatoarelor şi asignarea ieşirilor din sistem care le comandă. Din perspectiva programării, execuția unei acțiuni se limitează la activarea ieşirii corespunzătoare. Optional, terminarea unei acțiuni poate fi semnalată de un senzor tip “limitator de cursă” conectat la una din intrările digitale, care este testată de program.

4.2.3.4 Programarea deciziilor

Resursele sistemului

Din punct de vedere al programării, intrările şi ieşirile sistemului sunt interpretate ca “resurse hardware”. Acestea sunt asociate cu un set de variabile având denumirile şi tipul specificate sintetic în tabelul 4.1.

Pe lângă aceste resurse fizice, s-au introdus o serie de resurse “logice”, predefinite, implementate prin software. Lista acestora este prezentată în tabelul 4.2.

Tabelul 4.1. Variabile asociate cu resursele fizice ale sistemului

Nume	Descriere	Tip	Starea initiala
I0-I15	Intrări digitale	boolean	0
NI0-NI15	Starea complementară a intrărilor digitale	boolean	1
O0-O15	Ieşiri digitale	boolean	0
A0-A7	Intrări analogice	intregi de tip octet - în intervalul 0-255	nedefinită

Resursele logice pot fi grupate în următoarele categorii:

- Variabile asociate cu resursele fizice (intrări/ieşiri)
- Variabile de memorie de uz general
- Timere
- Variabile auxiliare
- Constante predefinite

Sistemul scanează permanent starea intrărilor fizice şi actualizează valorile variabilelor asociate (I0-I15, A0-A7). Starea variabilelor logice asociate cu ieşirile O0-O15 este determinată de *programul* executat de modulul de decizie, iar ieşirile fizice sunt controlate corespunzător cu starea variabilelor asociate.

Nota: Pentru comoditate, au fost definite două seturi de variabile asociate cu intrările digitale: I0-I15 – care reflectă starea reală a intrărilor şi NI0-NI15 – care conțin starea complementară a acestora.

Variabile de memorie de uz general

RDPL dispune de un număr de 104 variabile de memorie de uz general, denumite M0-M7 şi Y0-Y95. Toate sunt, din punct de vedere hardware, locații de memorie RAM, de 8 biti.

Atât M0-M7, cat şi Y0-Y95 pot stoca atat valorile unor variabile de tip întreg, cât şi valorile unor variabile de tip boolean. Totuşi, pentru ca programul sa fie mai uşor de

urmărit şi de depanat, se recomandă folosirea locațiilor M0-M7 pentru variabile de tip întreg, iar Y0-Y95 pentru variabile de tip boolean.

Tabelul 4.2. Resurse logice ale RDPL

Nume	Descriere	Tip	Starea initiala
M0-M7	Variabile de memorie de uz general	întreg de 8 biti cu valori în intervalul[0-255]	0
Y0-Y95	Variabile de memorie de uz general	boolean sau întreg	0
YTMR0-YTMR31	Timere predefinite	boolean	0
STATUS	Variabilă auxiliară de tip întreg	întreg de 8 biti cu valori în intervalul [0-255]	0
S0-S15	Variabile de stare asociate cu STATUS	boolean	toate au valoarea 0 cu exceptia uneia care e 1
0L	Nivel logic zero	boolean. Constantă predefinită	0
1L	Nivel logic unu	boolean. Constantă predefinită	1
RISE	Indică tranziție din 0 în 1 a unei variabile booleene	constantă predefinită	1
FALL	Indică tranziție din 1 în 0 a unei variabile booleene	constantă predefinită	2
UP	Indică numărator de tip “up counter”	constantă predefinită	1
DOWN	Indică numărator de tip “down counter”	constantă predefinită	0

Timere

RDPL dispune de un număr de 32 de timere predefinite, care sunt “vizibile” pentru progam sub forma unor variabile booleene, denumite YTMR0-YTMR31. Acestea pot funcționa fie ca monostabil, fie ca multivibrator astabil (oscilator).

In ambele situații, durata impulsurilor generate de timere se calculează cu formula: *T=K*Q* unde Q este cuanta de timp a timerului, iar K este o constantă definită prin program. Constanta K este un număr întreg de 8 biti, care poate lua valori în

intervalul [1-255]. Rezultă că un timer dat poate genera constante de timp în intervalul [Q-255*Q]. Caracteristicile timerelor sunt prezentate sinoptic în tabelele 4.3 şi 4.4.

Tabelul 4.3. Caracteristicile timerelor

Timer#	Ieşirea	Cuanta de timp Q	Tmin	Tmax
0-7	YTMR0-YTMR7	10 ms	10 ms	2550 ms
8-15	YTMR8-YTMR15	100 ms	100 ms	25.5 s
16-23	YTMR16-YTMR23	1 s	1 s	255 s
24-31	YTMR24-YTMR31	1 min	1 min	255 min

*Nota: în cazul funcționarii timerelor ca multivibrator (oscilator), atât impulsul cât şi pauza dintre impulsuri au durate egale T=K*Q. Rezultă că perioada ceasului generat de un astfel de timer este 2*T. Ținând seama că durata minimă a unui impuls este de 10ms, rezultă că perioada minimă este de 20ms, ceea ce corespunde la o frecvență maximă de 50Hz a ceasului generat.*

Variabile auxiliare, predefinite

Pentru uşurarea implementării prin program a unor automate finite, s-a definit un set de variabile auxiliare, special destinate acestui scop.

Variabila STATUS este de tip întreg de 8 biti, similară cu M0-M7. Particularitatea acestei variabile este aceea ca semioctetul sau inferior este automat decodificat şi se setează variabilele S0-S15, corespunzător valorii STATUS.

De exemplu, dacă STATUS=0, atunci S0=1 iar restul variabilelor S1-S15=0. Similar, dacă STATUS=7, atunci S7=1, iar restul varibilelor S, S0-S6 şi S8-S15 au valoarea 0, etc.

Constante predefinite

Pentru facilitarea scrierii fără erori şi a depanării programelor, s-au definit următoarele constante:

- 0L – ZERO LOGIC. Indicarea acestei constante ca operand al unei funcții logice are un efect similar conectarii intrării unui circuit logic la nivel logic zero (GND).
- 1L – UNU LOGIC. Indicarea acestei constante ca operand al unei funcții logice are un efect similar cu conectarea intrării unui circuit logic la unu logic (VCC).
- RISE – indică o tranziție din 0 în 1 a unei variabile booleene (front crescător).
- FALL – indică o tranziție din 1 în 0 a unei variabile booleene (front căzător).

Descrierea principalelor funcții

Funcția AND (ŞI LOGIC între operanzi)

Sintaxa generală:

```
AND        DEST,OP1,OP2,OP3,OP4
```

Destinația şi cei 4 operanzi ai funcției, OP1-OP4, pot fi orice variabile de tip boolean definite în sistem, fie ca e vorba de intrările digitale ale modulului I0-I7, sau de ieşirile altor funcții, notate Y0-Y94, sau ieşirile timerelor, YMTR0-YTMR31.

Ordinea în care se specifică operanzii nu are nici o importanță. De exemplu, următoarele expresii sunt perfect echivalente:

```
AND        Y1,I0,Y12,1L,1L
AND        Y1,Y12,1L,I0,1L
```

Funcția NAND (ŞI NEGAT LOGIC între operanzi)

Sintaxa: `NAND DEST,OP1,OP2,OP3,OP4`

Destinatia şi operanzii funcției NAND se supun condițiilor enumerate în cazul funcției AND.

Funcția OR (SAU LOGIC între operanzi)

Sintaxa: `OR DEST,OP1,OP2,OP3,OP4`

DEST – Destinația rezultatului operației indicate de funcție.

OP1…OP4 sunt cei patru operanzi asupra cărora se execută operația logică definită de funcție.

Nota: în cazul funcției OR, dacă exista intrări nefolosite, acestea trebuie conectate la 0L.

Funcția NOR (SAU NEGAT LOGIC între operanzi)

Sintaxa: `NOR DEST,OP1,OP2,OP3,OP4`

DEST – Destinația rezultatului operației indicate de funcție.

OP1…OP4 sunt cei patru operanzi asupra cărora se execută operația logică definită de funcție.

Funcția XOR. (SAU EXCLUSIV LOGIC între operanzi)

Sintaxa: `XOR DEST,OP1,OP2,OP3,OP4`

DEST – Destinația rezultatului operației indicate de funcție.

OP1-OP4 sunt cei patru operanzi asupra cărora se execută operația logica definita de funcție.

Funcția FF (SET-RESET Flip Flop)

Sintaxa: `FF DEST,S,R,PRIORITY`

Intrările SET şi RESET sunt active în 1 logic.

DEST, S,R pot fi oricare din resursele sistemului.

PRIORITY – este o constantă, care poate lua valorile 0 sau 1 pentru a indica prioritatea relativă a intrărilor de SET şi RESET.

PRIORITY=0 – Intrarea RESET este prioritară

PRIORITY=1 – Intrarea SET este prioritară

Exemple:

```
FF   Y12,I0,I1,0
FF   Y12,I1,I0,1
```

Evident, în acest caz, ordinea în care se specifica intrările de SET şi RESET este importantă pentru interpretarea funcției. Cele două funcții descrise în exemplul de mai sus NU sunt echivalente.

Funcția DCOMP (COMPARATOR DIGITAL)

Sintaxa: *DCOMP DEST, OP1, OP2, M*

Spre deosebire de funcțiile prezentate anterior, care aveau drept operanzi variabile booleene, funcția DCOMP compară bit cu bit doi operanzi de tip întreg de 8 biti.

Ieşirea funcției DCOMP este activă (are valoarea 1 logic) dacă OP1*M=OP2*M.

M este o constantă de 8 biti, denumită "mască", iar operația indicata de simbolul '*' este bitwise AND. În acest mod, sunt supuşi comparației şi afectează ieşirea funcției DCOMP doar acei biți ai operanzilor OP şi K situați în pozițiile în care M are un bit 1.

Funcția ACOMP (COMPARATOR ANALOGIC)

Sintaxa: *ACOMP DEST,OP1,OP2*

DEST – Destinația rezultatului operației indicate de funcție

OP1, OP2 – pot fi orice variabile de tip întreg. Destinația este de tip boolean.

ACOMP emulează funcționarea unui comparator analogic, în care OP1 este asociat cu intrarea neinversoare iar OP2 este asociat cu intrarea inversoare.

Funcția ACOMP a fost introdusă pentru a facilita operarea cu intrările analogice ale modulului, dar aplicațiile acestei funcții nu se limitează la aceasta.

Funcția STC (Store on Condition)

Funcția STC – Store on Condition (memorare conditionata) – a fost introdusă pentru a putea asigna unei variabile o valoare stocată de o alta, la momente de timp definitie de un operand specificat.

Sintaxa: *STC DEST,SRC,COND,EDGE*

DEST - destinația transferului

SRC – sursa transferului (valoarea transferată)

COND – este o variabilă booleană, care desemnează, atunci când este 1 logic, condiția în care are loc transferul.

EDGE – frontul semnalului de intrare care determină operația

Exemplu:

```
STC        STATUS,M0,Y0,RISE
```

Expresia de mai sus descrie o operație prin care variabila auxiliară STATUS capată valoarea stocată de M0 în momentele când Y0 inregistrează o tranziție din 0 în 1.

Funcția STIC (Store Immediate value on Condition)

Funcția STIC este similară cu STC, cu diferența că operandul sursă este o constantă, adresată imediat.

Sintaxa: *STIC DEST,K,COND,EDGE*

Funcția COUNTER. Definirea număratoarelor

Sistemul permite definirea unor număratoare de 8 biti, reversibile (care pot număra atât în sens crescator, cât şi descrescator), sensibile la frontul crescător sau descrescător al intrării de numărare.

Sintaxa: *COUNTER DEST,DIR,CLOCK,EDGE*

DEST – este destinația funcției – indică o variabilă de memorie de tip întreg de 8 biti, care poate lua valori în intervalul [0-255].

DIR – indică direcția de numărare. Poate fi orice variabilă de tip boolean definită în sistem, sau una din constantele predefinite UP sau DOWN.

Când DIR=0, sensul de numărare este invers (down counter)

Când DIR=1, sensul de numărare este direct (up counter)

CLOCK – este o variabilă de tip boolean, care poate fi oricare din resursele booleene ale sistemului.

EDGE – este o constantă care indică frontul activ al ceasului, pe care se face numărarea.

EDGE=1 – numărare pe front crescător al CLOCK

EDGE=2 – numărare pe front căzător al CLOCK

Pentru usurința citirii programelor, s-au definit două constante RISE şi FALL, care pot fi folosite pentru indicarea frontului crescător, respectiv descrescător de numărare în definiția număratoarelor.

Exemple:

```
COUNTER   M0,UP,I0,RISE
```

Funcția descrisa mai sus număra crescător şi stochează în variabila M0 fronturile crescatoare înregistrate la intrarea I0.

```
COUNTER   M0,I0,I1,FALL
```

Număratorul definit de expresia de mai sus numără direct atunci când intrarea I0=1, şi invers când I0=0, fronturile căzătoare ale intrării I1 şi stochează rezultatul în variabila M0.

Observație.

Număratoarele astfel definite nu-si pot depăşi capacitatea nici la numărare directa, nici la numărare inversă (nu există overflow)

In momentul în care se atinge limita de numărare (valoarea $FF la numărare directa, sau $00 la numărare inversa), număratorul îşi păstrează valoarea iar următoarele fronturi ale intrării CLOCK sunt ignorate.

Funcția TIMER. Definirea timerelor

Sintaxa: *TIMER TMRNO,TYPE,INPUT,EDGE,K*

TMRNO – Timer Number – este numărul de identificare al timerului reprezentat printr-un întreg în intervalul [0-31]. Numărul de ordine al timerului este asociat în mod unic cu cuanta de timp în modul descris în tabelul nr. 4.4.

TYPE – descrie tipul de timer. Este o constantă care poate lua valorile următoare:

TYPE=MONO TIMER-ul e configurat ca monostabil

TYPE=OSC TIMER-ul funcționează ca multivibrator astabil

Tabelul 4.4 Constante de timp asociate cu timerele predefinite

TMRNO	Destinația funcției	Cuanta de timp Q
0-7	YTMR0-YTMR7	10 ms
8-15	YTMR8-YTMR15	100 ms
16-23	YTMR16-YTMR23	1 s
24-31	YTMR24-YTMR31	1 min

INPUT – poate fi orice variabilă booleană definită în sistem. Tranzițiile sau starea acestei variabile determină funcționarea timerului intr-un mod definit de constanta EDGE.

EDGE – indică frontul activ al semnalului INPUT care declansează timerul. Este o constantă cu valori posibile în intervalul [0-3]. Interpretarea valorii EDGE e diferită în funcție de modul de operare definit de TYPE.

Semnificația valorilor constantei EDGE în funcție de modul de funcționare a timerului ca monostabil sau astabil este prezentată în tabelul 4.5.

Constanta K, împreună cu cuanta de timp predefinita a fiecarui timer, Q determină constanta de timp a timerului T_y=K*Q.

Tabelul 4.5. Semnificația parametrilor TYPE şi EDGE în definitia funcției TIMER

TYPE	EDGE	Funcționare ca
TYPE=0 Monostabil	0	Monostabil redeclansabil pe palier 0
	1	Monostabil declansat pe front crescător
	2	Monostabil declansat pe front urcator
	3	Monostabil redeclansabil pe palier 1
TYPE=1 Multivibrator astabil	0	Oscilator validat când INPUT=0
	1	combinație invalidă
	2	combinație invalidă
	3	Oscilator validat când INPUT=1

În cazul funcționarii ca monostabil, T_y este durata cât ieşirea YTMRx stă în starea HIGH dupa detectarea unei tranziții active a variabilei INPUT. În cazul funcționării ca astabil, Ty defineşte durata impulsurilor şi a pauzei între impulsuri, astfel încât perioada ceasului generat de astabil este Tosc=2*Ty.

Funcția MOVE (Comanda de mişcare)

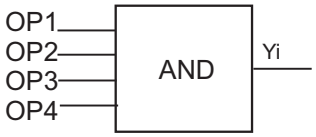
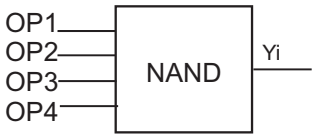
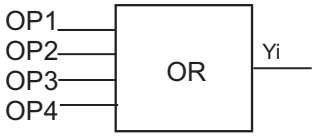
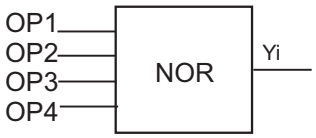
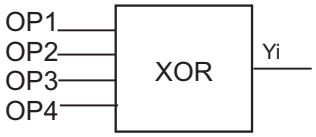
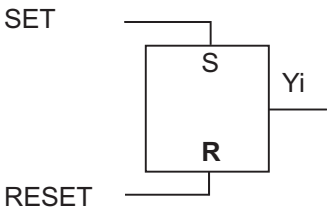
Sintaxa: *MOVE TRIGGER,GOAL_X,GOAL_Y,GOAL_HEADING,EDGE*

Defineşte o țintă de navigație prin coordonatele ei `GOAL_X`, `GOAL_Y`, şi `GOAL_HEADING`.

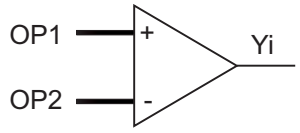
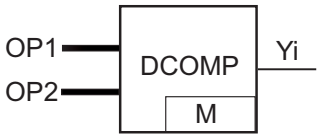
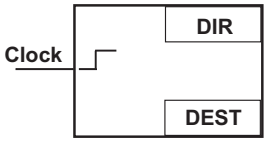
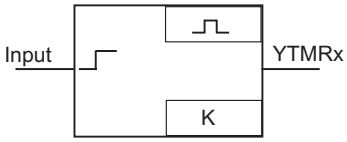
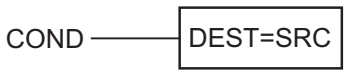
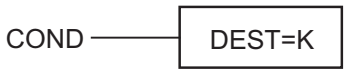
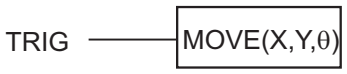
Parametrul `TRIGGER` poate fi oricare din resursele sistemului şi indică semnalul care declanşează transmisia comenzii de mişcare catre robot. `EDGE` este frontul activ al semnalului `TRIGGER`.

Tabelul 4.6 conține lista tuturor funcțiilor admise de limbajul RDPL.

Tabelul nr. 4.6 Lista funcțiilor insotită de simbolurile grafice folosite

Simbol grafic	Sintaxa
	AND DEST,OP1,OP2,OP3,OP4
	NAND DEST,OP1,OP2,OP3,OP4
	OR DEST,OP1,OP2,OP3,OP4
	NOR DEST,OP1,OP2,OP3,OP4
	XOR DEST,OP1,OP2,OP3,OP4
	FF DEST,SET,RESET,PRIORITY PRIORITY=0 RESET PRIORITAR PRIORITY=1 SET PRIORITAR

Tabelul nr. 4.6 Continuare

Simbol grafic	Sintaxa
	ACOMP DEST,OP1,OP2
	DCOMP DEST,OP1,OP2,MASK MASK este o constantă adresată imediat
	COUNTER DEST,DIR,CLOCK,EDGE EDGE este o constantă adresată imediat Valori predefinite RISE, FALL Valori predefinite pentru DIR – UP, DOWN
	TIMER TMR#,TYPE,INPUT,EDGE,K TMR# este o constantă adresată imediat valori predefinite pentru TYPE – MONO,OSC EDGE este o constantă adresată imediat
	STC DEST,SRC,COND,EDGE Transfer conditionat între variabile
	STIC DEST,K,COND,EDGE Initializare conditionata a unei variabile K este o constantă adresată imediat
	MOVE TRIGGER,X,Y,HEADING,EDGE

Exemplu de secvența de program

Presupunem ca intrarea I0 este conectată la butonul soneriei de la intrarea reşedinței persoanei asistate (cu deficiențe senzoriale) iar I1, I2, I3 sunt conectati la detectoarele de mişcare din principalele incinte.

Dacă în interval de 15 secunde de la activarea soneriei, nu se inregistrează mişcare în incintele supravegheate, se cere ca robotul sa se deplaseze către uşa de

la intrare si, eventual, sa acționeze o ieşire conectată la un zavor electromagnetic, pentru deschiderea uşii. Circuitul logic care realizează aceasta funcție este prezentat în figura 4.5.

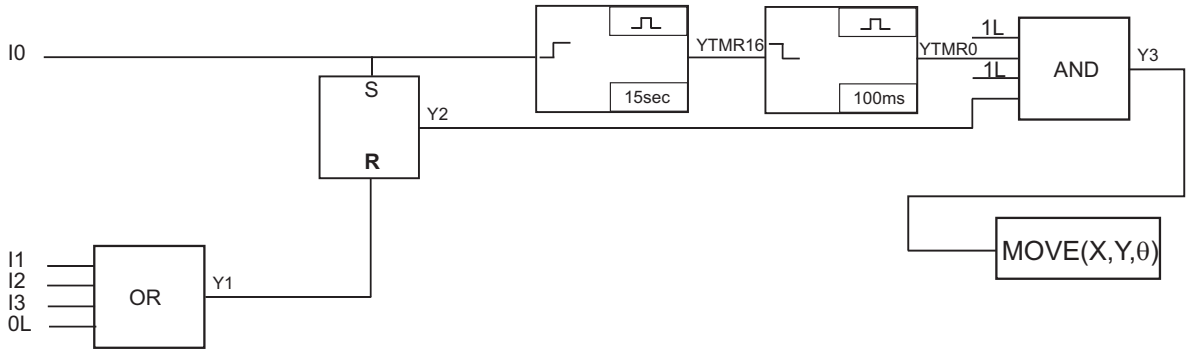


Fig. 4.5 Exemplu de program RDPL

Programul echivalent este următorul:

```
OR          Y1,I1,I1,I3,0L
FF          Y2,I0,Y1,0
TIMER       16,MONO,I0,RISE,15
TIMER       0,MONO,YTMR16,FALL,10
AND         Y3,YTMR0,Y2,1L,1L
MOVE        Y3,5000,3500,45,RISE
```

Detalii privind implementarea limbajului RDPL pe un microcontroller de uz general, precum şi despre structura compilatorului sunt prezentate în Anexa A.

4.3 Concluzii

Principalul avantaj al RDPL este simplitatea, care decurge din structura ierarhizată a sistemului de conducere propus pentru PRA. RDPL raspunde strict cerințelor de decizie în situații predictibile, descrise de anumite configurații ale semnalelor furnizate de senzori, fără a încerca sa includă elemente de control cinematic, sau de localizare prin recunoaşterea unor repere (landmarks).

In consecinta, RDPL poate fi implementat în sisteme de decizie realizate cu microcontrollere uzuale, iar programarea poate fi facută de utilizatori cu cunoştințe tehnice elementare.

Principalele dezavantaje ale RDPL sunt următoarele:

- RDPL nu acoperă situațiile în care informația furnizată de senzori este incertă, din cauza unor perturbații sau a erorilor de comunicație;
- In stadiul actual, RDPL nu include mecanisme de tratare a situațiilor de eroare, sau avarie. De exemplu, în cazul când o secvența de deplasare catre o țintă specificata esuează, din cauza intâlnirii unui obstacol pe care subsistemul de navigație nu il poate ocoli, singura soluție de finalizare a secvenței rămâne ghidarea vehiculului prin comenzi vocale.

In ciuda dezavantajelor enumerate, RDPL rezolvă satisfăcător majoritatea problemelor asociate cu decizia intr-un sistem robotic care evoluează intr-un mediu cunoscut şi manipulat prin amplasarea de senzori şi actuatoare.

Apreciem ca este o soluție demnă de luat în considerare pentru implementarea unor variante comerciale de asistenți robotici personali.

5

How Should I Get There? Problema controlului mişcării roboților mobili catre țintă

5.1 Formularea problemei

Considerând un vehicul autonom, modelat ca un solid rigid sau articulat, cu cinematică şi geometrie cunoscute, situat intr-un punct $p = [x, y, \theta]^T$, se pune problema conducerii vehiculului (guidance) intr-un punct țintă $p_d = [x_d, y_d, \theta_d]^T$, cu ocolirea eventualelor obstacole.

Astfel formulată, problema generală se poate descompune ([110]) în următoarele subprobleme distincte:

5.1.1 Planificarea rutei (path planning)

In acest caz, se pune problema determinarii unei secvențe continue de stări (poziții) intermediare ale robotului între starea initială *p* şi starea finală p_d, cu ocolirea obstacolelor. Problema planificării rutei a fost descrisă extensiv în ([111]) şi comportă în general un volum foarte mare de calcule, incompatibil cu resursele limitate ale microcontrollerelor uzuale. Din acest motiv, abordarea propusă în prezenta tezã pentru implementarea unor asistenti robotici personali, va presupune cã rutele sunt predefinite de un operator uman, iar problema obstacolelor *neaşteptate*, va fi tratată dintr-o perspectiva pur reactivă în capitolul 6.

5.1.2 Stabilizarea "punct la punct" (point to point stabilization, sau posture stabilization) presupune conducerea robotului catre punctul țintă, fără a se face vreo restrictie privind ruta de urmat. Aceasta este echivalent cu găsirea unei legi de control:

$$\begin{bmatrix} v \\ \omega \end{bmatrix} = u((p - p_d), t) \text{ în asa fel încât } (p - p_d,) \to 0 \text{, pentru } \forall p(0) \quad (5.1)$$

Urmărirea unei traiectorii geometrice definite într-o parametrizare independenta de timp (path following).

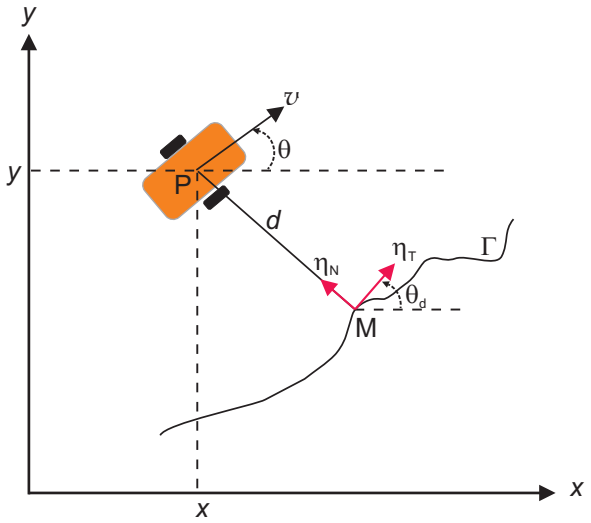


Fig. 5.1 Formularea problemei de conducere pentru path following

Cu notațiile din figura 5.1,unde M este proiecția ortogonală a poziției curente P a robotului pe curba Γ, *d* este distanța între P şi M, *s* este distanța curbilinie de-a lungul curbei Γ între un punct initial oarecare şi M, $\theta_d(s)$ este unghiul între axa Ox şi tangenta în M la curba Γ, iar $\tilde{\theta} = \theta - \theta_d$ este eroarea unghiulara, problema urmaririi traiectoriei este echivalentă cu definirea unei legi de reglaj $u = k(s, d, \tilde{\theta}, v(t))$, în aşa fel încât

$$\lim_{t \to \infty}(d(t)) = 0$$
$$\lim_{t \to \infty}(\tilde{\theta}(t)) = 0 \quad (5.2)$$

5.1.3 *Urmărirea unei traiectorii geometrice definite intr-o parametrizare dependentă de timp (trajectory tracking).*

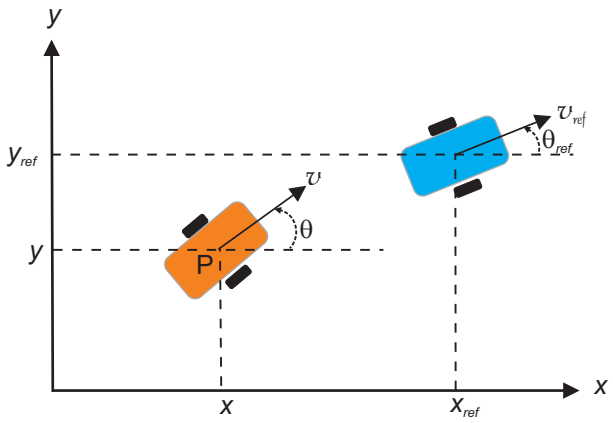


Fig. 5.2 Formularea problemei de conducere pentru trajectory tracking

Cu notațiile din figura 5.2,se considera un vehicul-referință virtual, având vitezele $v_{ref}(t), \omega_{ref}(t)$, mărginite,cu derivatele mărginite si

$$\lim_{t\to\infty}(v_{ref}(t)) \neq 0$$
$$\lim_{t\to\infty}(\omega_{ref}(t)) \neq 0 \qquad (5.3)$$

si notând $e = [x - x_{ref}, y - y_{ref}, \theta - \theta_{ref}]^T$ se pune problema găsirii unei legi de reglaj *u=f(e)*, în aşa fel încât:

$$\lim_{t\to\infty}(e(t)) = 0 \qquad (5.4)$$

Din perspectiva implementării conceptului de asistent robotic personal, prezintă interes practic doar problema urmăririi unei traiectorii definită intr-o parametrizare independentă de timp (path following).

5.2 Soluții cunoscute de conducere a roboților pentru path following

5.2.1 *Modalităti de reprezentare a traiectoriei țintă*

Modalitățile de reprezentare a traiectoriei țintă sunt clasificate ([112], [113]) în “explicite” şi “implicite”. O traiectorie explicită este o secvență de puncte, definite prin

coordonatele lor în raport cu un sistem de referință, optional unite între ele prin segmente de dreaptă sau prin segmente de curbă definită analitic (vezi figura 5.3).

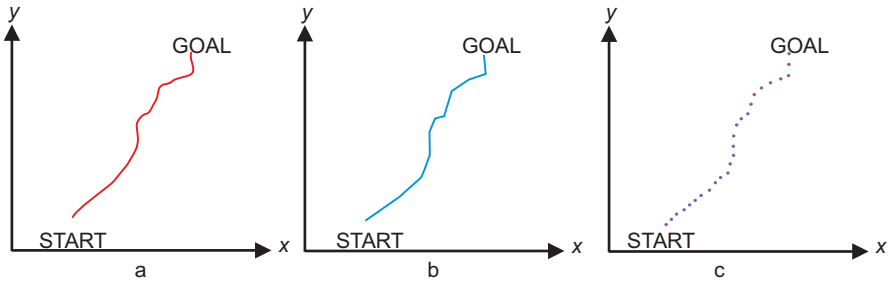


Fig. 5.3 Diverse modalităti de reprezentare explicita a traiectoriei țintă

În această abordare, estimarea erorii de poziționare față de traiectoria țintă presupune în mod obligatoriu existența unui sistem de localizare propriu vehiculului autonom.

O modalitate indirectă de a defini o traiectorie explicită este metoda denumită “follow the past” ([114]), în care robotul are capacitatea de a memora traseul pe care circulă atunci când este (tele)condus de un operator uman şi, ulterior, poate reface drumul în mod autonom.

În cazul reprezentării implicite a traiectoriei țintă, robotul extrage din mediu, cu ajutorul unui set adecvat de senzori, informații despre traiectoria pe care trebuie să o parcurgă, de exemplu urmărind marginile unui drum, radiația electromagnetică emisă de un cablu subteran, sau citind informația înregistrată în tag-uri RFID distribuite în mediu. În această situație nu este absolut necesar ca robotul să dispună de un sistem propriu de localizare, estimarea erorii de poziționare în raport cu traiectoria țintă fiind posibilă doar prin prelucrarea directă a informației furnizate de senzori. Un exemplu tipic de astfel de abordare este algoritmul denumit “follow the wall”, în care robotul este programat să se deplaseze menținând distanța constantă față de un perete de referință al unui coridor.

Urmărirea unei traiectorii definite implicit are avantajul simplitatii, dar are două dezavantaje majore:

- Depinde în mod esential de senzori, ceea ce face ca eventualele măsurători afectate de zgomot sa se traducă imediat în oscilații ale sistemului de poziționare al vehiculului.
- Non-holonomicitatea vehiculului limitează mişcarile posibile pornind de la o anumita poziție, astfel încât este posibil ca rutele definite implicit să conțină secțiuni care nu pot fi urmărite cu acuratețe.

5.2.2 Algoritmi geometrici de urmărire a traiectoriei (path following)

Denumirea de agloritmi geometrici a fost sugerata de Morales et. al. în [113] pentru a descrie o serie de algoritmi de urmărire a unei traiectorii, bazați pe exprimarea erorii de poziționare prin analiza geometrică a poziției vehiculului față de traiectoria țintă.

In aceasta categorie se incadrează algoritmii denumiti “follow the carrot” ([115]), “pure pursuit” ([113],[116]), “vector pursuit” ([117]) şi “stanley” ([118]).

5.2.2.1 Algoritmul “Follow the Carrot”

Principiul algoritmului “follow the carrot” este ilustrat în figura 5.4.

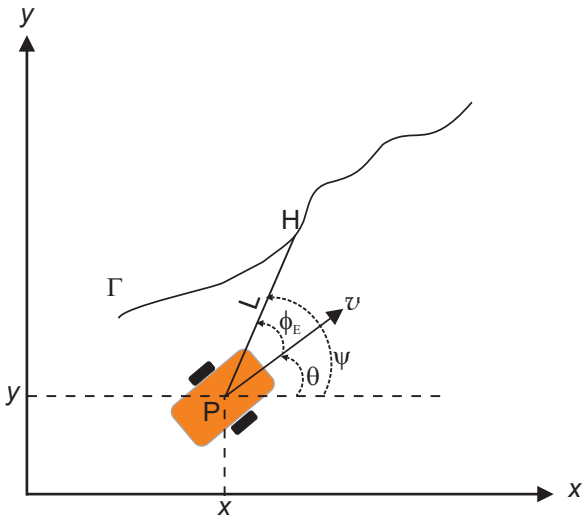


Fig. 5.4 Ilustrarea principiului algoritmului “follow the carrot”

Se alege un punct H, denumit “look-ahead point”, pe traiectoria țintă, aflat la o distanța L, constantă, față de vehicul (“look-ahead distance”). Se admite ca expresie a erorii de poziționare, abaterea unghiulară

$$\phi_E = \psi - \theta \qquad (5.5)$$

unde ψ este unghiul facut de direcția PH cu axa Ox (“look-ahead angle”), iar θ este orientarea curentă a vehiculului şi se controlează direcția vehiculului în aşa fel încât:

$$\Delta\theta = k_p \phi_E \qquad (5.6)$$

Relația (5.6) exprimă faptul ca “follow the carrot” implementează un regulator proporțional pentru orientarea vehiculului si, în consecinta, are toate dezavantajele şi limitările regulatoarelor proporționale.

5.2.2.2 Algoritmul Pure Pursuit

Termenul “pure pursuit” (urmărire simplă) a fost propus în literatura tehnică militară ([119]) în contextul urmăririi țintei de catre rachete si, ulterior, a fost preluat de Amidi ([116]) care a descris algoritmul de urmărire a unei traiectorii ca un proces similar urmăririi unui punct aflat în mişcare pe traiectorie, la o distanța *L* inaintea vehiculului. (“look-ahead distance”). Spre deosebire de algoritmul “follow the carrot”, în pure pursuit se urmăreste determinarea unui arc de cerc, tangent la orientarea curentă a vehiculului, care intersectează traiectoria țintă intr-un punct aflat la distanța L de vehicul.

Considerând un sistem de referința solidar cu vehiculul, ca în figura 5.5 şi definind curbura traiectoriei γ ca fiind inversul distanței *r* între poziția curentă a vehiculului şi centrul instantaneu de rotatie, sau ca variatia elementara a orientării vehiculului raportată la distanța parcursă pe traiectorie:

$$\gamma = \frac{1}{r} = \frac{d\phi}{ds} \qquad (5.7)$$

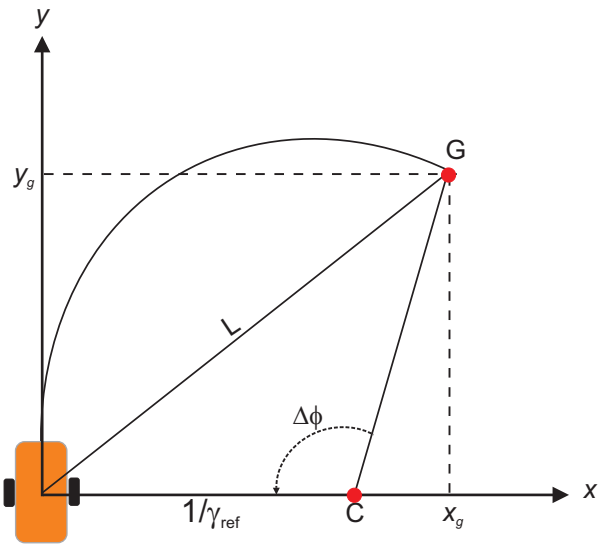


Fig. 5.5 Ilustrarea principiului algoritmului "pure pursuit" pentru path following

Coordonatele puntului țintă G se pot exprima în acest sistem de coordonate prin:

$$x_g = \frac{\cos(\Delta\phi) - 1}{\gamma_{ref}}$$
$$y_g = \frac{\sin(\Delta\phi)}{\gamma_{ref}} \qquad (5.8)$$

$$L^2 = x^2{}_g + y^2{}_g \qquad (5.9)$$

$$L^2 = \frac{2(1 - \cos(\Delta\phi))}{\gamma^2{}_{ref}} \qquad (5.10)$$

$$L^2 = -\frac{2x_g}{\gamma_{ref}} \qquad (5.11)$$

$$\gamma_{ref} = -\frac{2x_g}{L^2} \qquad (5.12)$$

Semnul curburii γ este pozițiv dacă vehiculul executa o rotație în sens invers acelor de ceasornic şi negativ când mişcarea se execută în sens orar.

In expresia (5.12), x_g poate fi interpretat ca eroarea de poziționare iar $-\frac{2}{L^2}$ este un factor de proporționalitate.

În concluzie, strategia de reglaj în cazul algoritmului “pure pursuit” este să schimbe la momente discrete de timp kT_0, valoarea referinței curburii γ_{ref} in aşa fel încât mişcarea sa se desfasoare pe un arc de cerc tangent la axa vehiculului, care intersectează traiectoria țintă la distanța L de punctul curent. Procesul este iterativ, iar rezultatul este prezentat în figura 5.6.

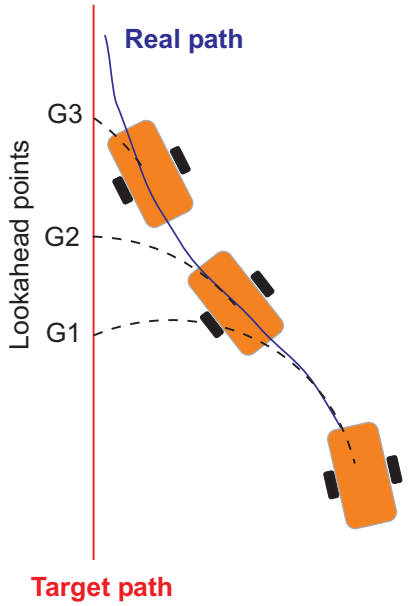


Fig. 5.6 Pure pursuit la urmărirea unei traiectorii liniare

În cazul particular al unui vehicul cu tracțiune diferențialã, unde vitezele v_L şi v_R se pot controla independent (vezi figura 5.7), sunt valabile relațiile:

$$v = \frac{v_L + v_R}{2} \tag{5.13}$$

$$\begin{aligned} v_L T &= 2\pi R_1 \\ v_R T &= 2\pi R_2 \end{aligned} \tag{5.14}$$

$$R_1 - R_2 = b \tag{5.15}$$

$$(v_L - v_R)T = 2\pi(R_1 - R_2) = 2\pi b \tag{5.16}$$

Punând condiția ca viteza rezultantă v, sa fie constantă, vitezele roților motoare v_L, v_R pot fi controlate în aşa fel încât:

$$v_L = v + \Delta v$$
$$v_R = v - \Delta v \qquad (5.17)$$

$$v_L - v_R = 2\Delta v \qquad (5.18)$$

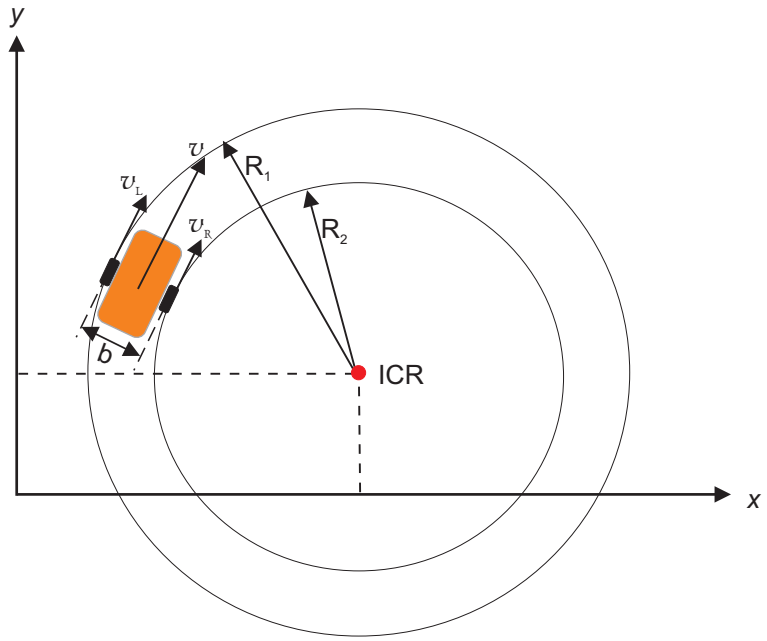


Fig. 5.7 Curbura traiectoriei unui vehicul cu tracțiune diferențială în funcție de vitezele roților motoare

Din (5.16) şi (5.18) rezultă:

$$\Delta v T = \pi b \qquad (5.19)$$

Dar:

$$T = \frac{2\pi R}{v} \qquad (5.20)$$

rezultă:

$$\gamma = \frac{1}{R} = \frac{2\Delta v}{vb} \qquad (5.21)$$

Din (5.12) şi (5.21) rezultă:

$$\Delta v_{ref} = -\frac{vb}{L^2} x_g \qquad (5.22)$$

Relațiile de mai sus sunt valabile intr-un sistem de coordonate solidar cu vehiculul. Deoarece, în general, sistemele de localizare şi metodele explicite de definire a traiectoriei țintă, furnizează coordonatele curente ale robotului şi coordonatele

punctelor țintă intr-un sistem de coordonate global *(x_w,y_w),* este necesara o translatie între cele două sisteme de referința. Se demonstrează ([120]) ca:

$$\begin{aligned} x_g &= (x_{gw} - x_{rw})\cos\theta_w + (y_{gw} - y_{rw})\sin\theta_w \\ y_g &= -(x_{gw} - x_{rw})\sin\theta_w + (y_{gw} - y_{rw})\cos\theta_w \end{aligned} \qquad (5.23)$$

unde $[x_{rw}, y_{rw}, \theta_w]^T$ este poziția robotului în sistemul de coordonate global, iar (x_{gw}, y_{gw}) sunt coordonatele puntului țintă G în sistemul *(x_w,y_w).*

Datorită simplității, algoritmul "pure pursuit" este destul de răspândit. Problema stabilității a fost tratată în ([121]), iar în ([122]) se propune o îmbunătățire a algoritmului prin adăugarea unei componente de reglaj integral. În ([123]) se propune completarea algoritmului cu un supervizor în logică fuzzy.

5.2.3 Conducerea fuzzy a roboților mobili pentru urmărirea unei traiectorii

Conceptul de Fuzzy Logic Controller (FLC) a gasit numeroase aplicații în conducerea roboților autonomi, incepand chiar cu lucrarea de pionierat a lui M. Sugeno şi M. Nishida ([124]). În prezent există sute de lucrări semnificative pe aceasta temă. Un review al principalelor directii de aplicare a logicii fuzzy în domeniul navigației roboților autonomi, poate fi gasit în ([125]).

Detalii privind implementarea FLC sunt disponibile în ([126], [127], [87]). O aplicație tipica este prezentată în ([128]).

Principalele avantaje ale controllerelor fuzzy, care le recomandă pentru aplicații în conducerea vehiculelor autonome, sunt:

- Formatul de enunțare a regulilor fuzzy permite descrierea simplă şi eficientă a unei game largi de task-uri, fără a necesita modele matematice complexe.
- Datorită caracterului calitativ al descrierilor fuzzy, bazele de cunostinte sunt uşor portabile de pe o platformă pe alta.

- Datorită caracterului interpolativ al controlului fuzzy rezultă o mişcare lină, continuă a vehiculului controlat, chiar în situații în care informația furnizată de senzori este afectată de erori sau fluctuatii.

Implementarea unui FLC este simplu de realizat, dacă se cunoaste eroarea de reglaj *e(t).* Derivata erorii *e'(t)* la momentul $t_k = t_{k-1} + \Delta t$ se poate exprima ca $e'(t_k) = e(t_k) - e(t_{k-1})$, dacă $\Delta t \to 0$. e(t) şi e'(t) sunt intrări pentru FLC (fig.5.8).

Fig. 5.8 Schema bloc a unui FLC

Definind pentru funcțiile e(t) şi e'(t) domeniile fuzzy N (Negativ), Z (zero) şi P (Pozitiv), cu funcțiile de apartenenta μ_N, μ_Z, μ_P, respectiv μ'_N, μ'_Z, μ'_P, $\mu : X \to [0,1]$, iar pentru ieşirea *u(t)* domeniile fuzzy L (Low), M(Medium) şi H (High) de tip singleton, comportamentul controllerului se poate descrie printr-un set de reguli lingvistice de tipul următor:

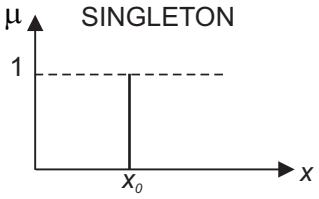
```
IF((e is N) AND (e' is Z)) THEN (u must be M)
```
(5.24)

Valoarea de adevăr a propozițiilor `(e is N)` şi `(e' is Z)` din antecedentul propoziției (5.24) în logica fuzzy, se determină evaluând funcțiile de apartenență $\mu_N(e(t))$ şi $\mu'_Z(e'(t))$. Funcțiile de apartenență μ_N, μ_Z, μ_P, şi μ'_N, μ'_Z, μ'_P, pot fi definite, de exemplu, ca în figura 5.9.

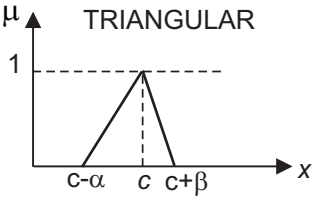
Valoarea de adevăr a întregii propoziții (5.24) este:

$$z = \min(\mu_N(e), \mu'_Z(e')) \tag{5.25}$$

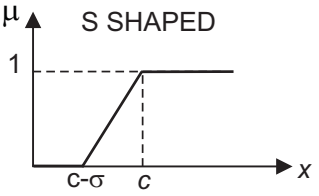
$$\mu(x)=\begin{cases}1 & x=x_0\\ 0 & x\neq x_0\end{cases}$$



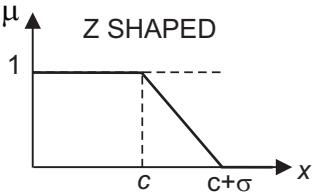
$$\mu(x)=\begin{cases}1-\dfrac{c-x}{\alpha} & x\in[c-\alpha,c]\\ 1-\dfrac{x-c}{\beta} & x\in[c,c+\beta]\\ 0 & x<c-\alpha, x>c+\beta\end{cases}$$



$$\mu(x)=\begin{cases}0 & x<c-\sigma\\ 1-\dfrac{|x-c|}{\sigma} & x\in[c-\sigma,c]\\ 1 & x>c\end{cases}$$



$$\mu(x)=\begin{cases}1 & x<c\\ 1-\dfrac{|x-c|}{\sigma} & x\in[c,c+\sigma]\\ 0 & x>c+\sigma\end{cases}$$



$$\mu(x)=\exp\left(-\frac{1}{2}\left(\frac{x-c}{\sigma}\right)^2\right)$$

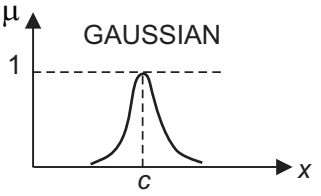


Fig. 5.9 Câteva moduri uzuale de a defini funcțiile de apartenență fuzzy

Întreaga "bază de reguli" care descrie comportamentul sistemului se poate reprezenta tabelar, ca în tabelul 5.1:

Tabelul 5.1 Un exemplu de reprezentare tabelară a bazei de reguli a unui FLC

error dot e'(t)	error e(t)		
	N	**Z**	**P**
N	H	L	L
Z	H	M	M
P	M	M	L

Valoarea "crisp" a ieşirii controllerului fuzzy se calculează cu relația:

$$u = \frac{\sum_{i=1}^{K} z_i S_i}{\sum_{i=1}^{K} z_i} \qquad (5.26)$$

unde z_i sunt valorile de adevăr ale propozițiilor care descriu regulile fuzzy, calculate cu (5.25), K este numărul total de reguli, iar S_i sunt valorile singleton ale ieşirii fuzzy a sistemului.

In ([129]) am descris o metodă simplă de implementare a acestui algoritm fuzzy pentru urmărirea unei traiectorii, definită ca o secvența de segmente de dreaptă sau arcuri de cerc, în cazul unui vehicul autonom cu două roti motoare şi tracțiune diferențială.

Două puncte succesive $(x_i, y_i), (x_{i+1}, y_{i+1})$ de pe traiectorie definesc o dreapta descrisa de ecuația:

$$Ax + By + C = 0 \qquad (5.27)$$

unde:

$$\begin{aligned} A &= y_{i+1} - y_i \\ B &= -x_{i+1} - x_i \\ C &= x_{i+1} y_i - x_i y_{i+1} \end{aligned} \qquad (5.28)$$

Distanța de la punctul curent, (x_c, y_c) la dreapta (5.27) este:

$$d = \frac{| Ax_c + By_c + C |}{\sqrt{A^2 + B^2}} \qquad (5.29)$$

Iar eroarea de poziție față de curba țintă definită de (5.27) este distanța cu semn între punctul curent şi curba țintă:

$$e = \frac{Ax_c + By_c + C}{\sqrt{A^2 + B^2}} \qquad (5.30)$$

iar derivata erorii:

$$e'(t_k) = e(t_k) - e(t_{k-1}) \qquad (5.31)$$

Proiectând controllerul fuzzy în aşa fel încât sa genereze simultan referințe distincte pentru vitezele roților motoare v_R, v_L se pot controla simultan atât translația cât şi orientarea vehiculului.

Regulile lingvistice care descriu funcționarea controllerului fuzzy sunt de forma:

IF((e is N) AND (e' is Z)) THEN (v_L must be L)AND (v_R must be H) (5.32)

Folosind funcții de apartenență liniare simetrice, ca în figura 5.10, se obține un FLC cu doar doi parametri de ajustat, respectiv M şi M', pantele funcțiilor de apartenență pentru *e(t)* şi *e'(t)*.

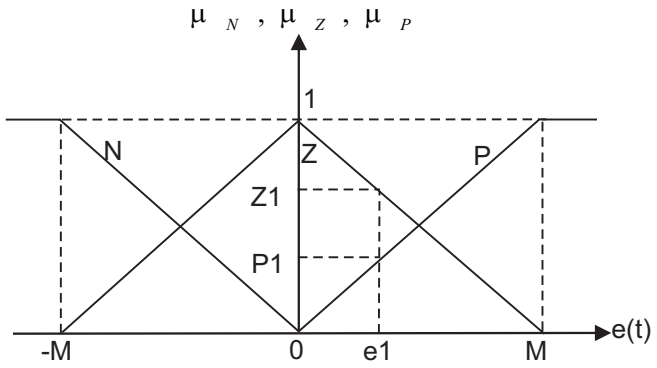


Fig. 5.10 Funcții de apartenență liniare simetrice

Performanțele regulatorului fuzzy descris mai sus se imbunătățesc semnificativ prin impărtirea universului discursului asociat cu e(t) şi e'(t) în cinci subdomenii fuzzy: Zero (Z), Negative Small (NS), Negative Big (NB), Poziține Small (PS) şi Positive Big (PB), ca în figura 5.11.

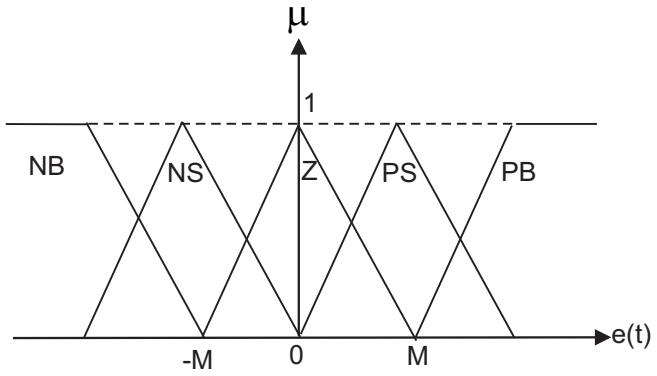


Fig. 5.11 Alta modalitate de definire a subdomeniilor fuzzy pentru e(t)

In concluzie ([130]), controllerele fuzzy asigura performanțe comparabile cu regulatoarele PID, având fața de acestea avantajul robusteții la incertitudinile de model şi acordarea uşoară. Principala critică formulată la adresa FLC e legată de inexistența unor metode riguroase pentru studiul stabilității acestor sisteme.

Exista câteva incercari notabile de a soluționa aceasta problema ([131],[132]), dar dificultățile persistă, pentru că derivă dintr-o contradicție între nevoia de a cunoaşte un model riguros al sistemului pentru studiul stabilității şi caracterul inerent vag, imprecis al aplicațiilor fuzzy.

5.2.4 Conducerea roboților mobili cu ajutorul feromonilor artificiali

5.2.4.1 Conceptul de feromoni artificiali

Feromonii naturali sunt substanțe chimice răspândite în mediu de către unele specii de insecte şi alte organisme, cu scopul de a influența comportamentul şi, uneori, chiar fiziologia, altor membri ai speciei.

Termenul “pheromone” a fost introdus în 1959, de Karlson şi Luescher [133] şi, în acelaşi an, Grasse [134] a explicat mecanismul de coordonare a acțiunilor, în coloniile de termite, printr-un proces de comunicare indirectă, mediat de feromonii distribuiți în mediu, mecanism pe care l-a denumit “stigmergie” (de la gr. stigma - înțepătură, imbold şi ergos - muncă).

A fost nevoie de aproape 30 de ani până când Deneubourg, Aron et. al. ([135], [136], 137]) au observat posibilitatea de a crea agenți artificiali, biomimetici, care comunică şi interacționează prin intermediul unui mecanism similar.

În 1989 Beni şi Wang ([138]) au introdus conceptul de “swarm intelligence”, iar între 1996 şi 1999, Dorigo, Bonabeau et al. au publicat câteva lucrări ([139], [140], [141], [142]) explorând mecanismul de auto-organizare în roiuri (swarms) de furnici şi au numit “ant colony optimization” (ACO) procesul prin care furnicile în căutarea

hranei reuşesc să găsească drumul cel mai scurt între cuib şi sursa de hrană.

In continuare, un mare număr de lucrari ştiințifice propun diverse metode de a crea *feromoni artificiali*.

Evident, cel mai simplu mod de a emula feromonii naturali constă în distribuirea unor substanțe chimice în mediu şi măsurarea concentrațiilor acestora în diverse puncte din spațiul înconjurător, printr-un mecanism similar cu olfacția ([143]).

Aplicații directe în robotică ale acestei abordări pot fi găsite în ([144], [145], [146]).

O abordare diferită propun Payton et. al. în ([147], [148]). Aceştia folosesc transceivere infrared cu rază scurtă de comunicație pentru a transmite mesaje de la un agent la altul, mesaje pe care le denumesc “virtual pheromones”. Termenul “virtual pheromone” poate fi regăsit în ([149]) şi desemnează tot entități de comunicație.

Într-o altă abordare, Kernbach et. al. propun în ([150]) folosirea unor senzori de temperatură pentru detectarea prezenței şi mişcării altor agenți din colonie şi inducerea unor comportamente de tip “swarm intelligence”.

O încercare de sistematizare a cercetărilor legate de aplicarea conceptului de feromoni artificiali în roboticã i se datorează lui Parunak ([151]), care foloseşte termenul de “digital pheromones”.

În sfârşit, în ([152],[153],[154]) sunt descrise câteva experimente în care feromoni digitali, concepuți ca structuri speciale de date, sunt stocați într-o rețea de tag-uri RFID distribuite în mediu.

Conceptul de feromoni artificiali e folosit în literatură într-un context în care se pune problema coordonării comportamentelor unor formațiuni de roboți, printr-un proces de comunicare indirectă, în mod absolut similar cu fenomenul de stigmergie în coloniile de insecte. În ([154]) am sugerat posibilitatea folosirii feromonilor digitali,

pentru definirea unor traiectorii folosite la conducerea unor roboți individuali.

In concluzie, conceptul de feromoni artificiali, desi relativ recent, s-a dovedit deosebit de fertil în privinta formularii unor noi soluții de conducere a roboților mobili individuali şi a formațiunilor de roboti.

5.3 Contributii la conducerea roboților mobili pentru urmărirea unei traiectorii

5.3.1 Conducerea roboților mobili cu ajutorul feromonilor virtuali

5.3.1.1 Modelarea feromonilor naturali

Pentru modelarea feromonilor naturali, trebuie considerate următoarele procese asociate cu aceştia:

- Difuzia – proces care conduce la crearea unor gradienți spațiali ai intensității de feromoni, detectabili de catre senzori.
- Evaporarea – constă în scăderea în timp a intensității sursei de feromoni.
- Agregarea – constă în superpoziția efectelor unor surse multiple de feromoni.

Procesul de difuzie spatială poate fi modelat admițând ca intensitatea feromonilor generați de o sursă de intensitate unitară, percepută la distanța x de sursă este o funcție de distanța x, $p : R^+ \to [0,1]$, de exemplu de forma:

$$p(x) = \begin{cases} p_0\left(1 - \dfrac{x}{\sigma}\right) & 0 < x < \sigma \\ 0 & x \geq \sigma \end{cases} \qquad (5.33)$$

In expresia (5.33), σ are semnificatia distanței maxime la care sursa de feromoni este detectabilă (sensibilitatea receptorului).

Procesul de evaporare poate fi modelat admițând că intensitatea unei surse de feromoni variază în timp, de exemplu liniar:

$$p(t) = p_0\left(1 - \frac{t}{\tau}\right) \qquad (5.34)$$

unde τ are semnificația unei constante de timp de evaporare specifică sursei de

feromoni.

Din (5.33) şi (5.34) rezultă:

$$\overline{p}(x,t) = \begin{cases} p_0 \bar{\imath}\left(1-\frac{x}{\sigma}\right)\left(1-\frac{t}{\tau}\right) & 0 < x < \sigma \\ 0 & x \geq \sigma \end{cases} \qquad (5.35)$$

unde $\bar{\imath}$ este versorul axei de-a lungul căreia se măsoară distanța x.

Tinând seama de (5.35), efectul de agregare a N surse de feromoni, aflate la distantele $d_1, d_2, \ldots d_N$ de un punct oarecare R (vezi figura 5.12) se poate exprima:

$$\overline{P_R} = \sum_{k=1}^{N} p_0 \bar{\imath}_k \left(1-\frac{d_k}{\sigma}\right)\left(1-\frac{t}{\tau}\right) \qquad (5.36)$$

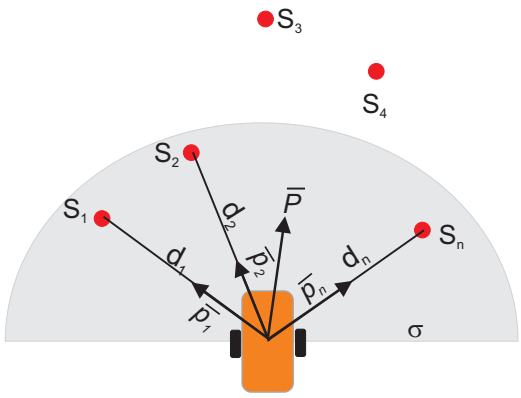


Fig. 5.12 Ilustrarea superpoziției efectelor a N surse de feromoni

Relația (5.36) este modelul feromonilor virtuali.

Pentru aplicații de urmărire a unei traiectorii de către roboti individuali, efectul de evaporare a feromonilor artificiali este neinteresant. Fenomenul de evaporare prezintă interes în situațiile în care se urmăreşte obținerea unor efecte de tip “ant colony optimization” ([139],[140]), sau alte efecte de auto-organizare, în formațiuni de roboți.

Din acest motiv, relația (5.36) se poate simplifica, eliminând dependența de timp:

$$\overline{P_R} = \sum_{k=1}^{N} p_0 \bar{\imath}_k \left(1-\frac{d_k}{\sigma}\right) \qquad (5.37)$$

Se observă similitudinea între modelul feromonilor dat de (5.37) şi conceptul de câmp de potential virtual (Virtual Potential Field) propus de Khatib, în ([155]) pentru rezolvarea problemei planificarii rutei (path planning) cu ocolirea obstacolelor. În aceasta abordare, robotul se comportă ca o particulă încărcată electric, care este atrasă de punctul țintă, fiind în acelasi timp respinsă de obstacole.

Conform modelului propus (5.37) pentru feromonii virtuali, aceştia exercită influențe caracterizate prin intensitate şi direcție asupra sistemului de percepție al robotului, în interiorul unui câmp perceptual, definit de parametrul σ (vezi figura 5.12).

5.3.1.2 Definirea traiectoriei țintă cu ajutorul feromonilor virtuali

Traiectoria țintă se defineşte explicit, creând o hartă de tip grid a mediului şi amplasând surse unitare de feromoni în nodurile retelei, într-o secvență care uneşte un punct de start cu un punct țintă, ca în figura 5.13.

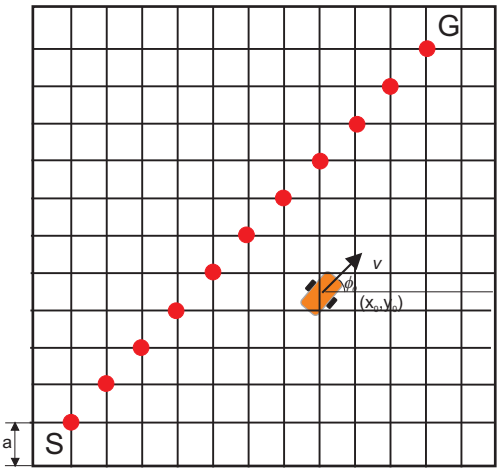


Fig. 5.13 Definirea traiectoriei țintă cu ajutorul feromonilor virtuali

Din rațiuni care țin de limitările de memorie şi viteza de prelucrare specifice sistemelor bazate pe microcontrollere încorporate, *această hartă nu este stocată în memoria microcontrollerului* care implementează task-urile asociate cu navigația vehiculului autonom, ci în memoria unui computer aflat la sol, denumit *"server de feromoni"*.

Periodic, robotii mobili, aflați în comunicație wireless cu serverul de feromoni emit pachete de interogare, care conțin un identificator al robotului şi coordonatele curente ale acestuia (poziția şi orientarea), aşa cum sunt raportate de subsistemul de localizare. Serverul localizează robotul pe harta internă, calculează concentratia de feromoni corespunzătoare poziției şi orientării robotului şi raspunde cu un pachet de date care conține ID-ul robotului care a emis interogarea şi valoarea concentrației de feromoni cerută.

Astfel definită, harta care încorporează informația despre distribuția feromonilor virtuali se comportă ca o *zonă de memorie comună* tuturor roboților aflați în comunicație cu serverul de feromoni.

5.3.1.3 Calculul concentrației de feromoni

Pentru calculul concentrației de feromoni, corespunzătoare unei anumite poziții a robotului, vom face o ipoteză suplimentară şi anume, vom presupune că sensibilitatea la feromoni este directională – ceea ce corespunde realității în sistemele biologice.

In consecință, feromonii vor fi detectabili doar într-o zona semicirculară, de rază σ, pe direcția de deplasare a robotului (vezi figura 5.14). Sursele de feromoni aflate în semiplanul din spatele robotului sunt ignorate la calculul concentrației rezultante.

Cunoscând poziția robotului (x_0, y_0, θ_0), precum şi coordonatele surselor de feromoni (x_i, y_i), se pune problema determinării intensității rezultante a feromonilor, P_R, şi a directiei ϕ_R a sursei echivalente de feromoni.

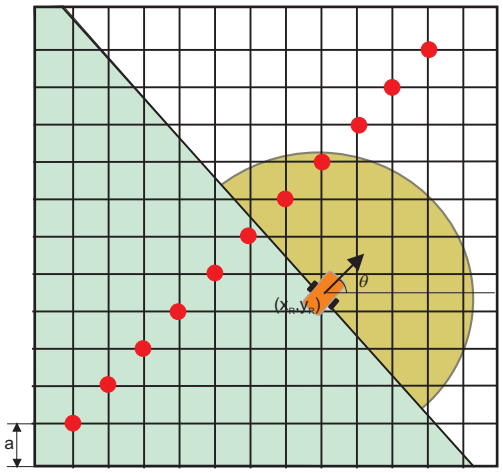


Fig. 5.14 Ilustrarea zonei semicirculare de sensibilitate la feromoni

Pentru fiecare sursă de feromoni, se poate calcula distanța fața de centrul de masă al robotului, intensitatea cu care sunt percepuți feromonii proveniți de la sursa respectivă $\overline{p_i}$ şi orientarea ϕ_i.

$$di = \sqrt{(x_i - x_0)^2 + (y_i - y_0)^2} \tag{5.38}$$

$$p_i = p_0\left(1 - \frac{d_i}{\sigma}\right) \tag{5.39}$$

$$\phi_i = \arcsin\left(\frac{x_i - x_0}{d_i}\right) \tag{5.40}$$

Prin superpoziția efectelor celor N surse de feromoni aflate în zona de sensibilitate se obține:

$$P_R = \sqrt{\left(\left(\sum_{i=1}^{N} p_i \cos\phi_i\right)^2 + \left(\sum_{i=1}^{N} p_i \sin\phi_i\right)^2\right)} \tag{5.41}$$

$$\phi_R = arctg\left(\frac{\left(\sum_{i=1}^{N} p_i \sin\phi_i\right)}{\left(\sum_{i=1}^{N} p_i \cos\phi_i\right)}\right) \tag{5.42}$$

Unde P_R şi ϕ_R sunt respectiv intensitatea şi direcția unei surse echivalente de feromoni, care cumulează efectele celor N surse elementare.

5.3.1.4 Abordări posibile pentru conducerea roboților mobili folosind conceptul de feromoni virtuali

a. Abordarea bio-mimetica

Insectele sociale percep gradienții spațiali ai concentrației de feromoni cu ajutorul a două antene, plasate simetric, de o parte şi de alta a capului. În mod similar, se pot defini două puncte simetrice față de axa robotului, de exemplu în dreptul roților motoare , puncte în care se calculează concentratiile de feromoni conform (5.41).

Coordonatele acestor puncte (L – Left, R – Right) sunt:

$$\begin{aligned} x_R &= x_0 - \frac{b}{2}\sin\theta_0 \\ y_R &= y_0 - \frac{b}{2}\cos\theta_0 \\ x_L &= x_0 + \frac{b}{2}\sin\theta_0 \\ y_L &= y_0 + \frac{b}{2}\cos\theta_0 \end{aligned} \qquad (5.43)$$

unde *b* este distanța între planele roților motoare ale robotului.

Folosind aceste puncte ca referință, cu relațiile (5.38)-(5.41) se determină P_L şi P_R, concentratiile de feromoni la nivelul acestor "antene". Cu aceste valori, presupunand ca $P_R, P_L \in [0,1]$, iar K este o constantă de scalare, se pot determina referințele de viteză pentru roțile motoare:

$$\begin{aligned} v_L &= \begin{cases} K(1-|P_L - P_R|) & \text{for } P_L > P_R \\ K|P_L - P_R| & \text{for } P_L <= P_R \end{cases} \\ v_R &= \begin{cases} K|P_L - P_R| & \text{for } P_L >= P_R \\ K(1-|P_L - P_R|) & \text{for } P_L < P_R \end{cases} \end{aligned} \qquad (5.44)$$

Relațiile (5.44) corespund cu un regulator proporțional, cu performanțe modeste şi fără mare importanță practică, Această abordare are însă meritul de a pune în evidență faptul că întregul proces de inferență prin care se ajunge de la o distribuție spatială de feromoni până la referințele de viteza, este perfect similar cu o retea neuronală (vezi figura 5.15).

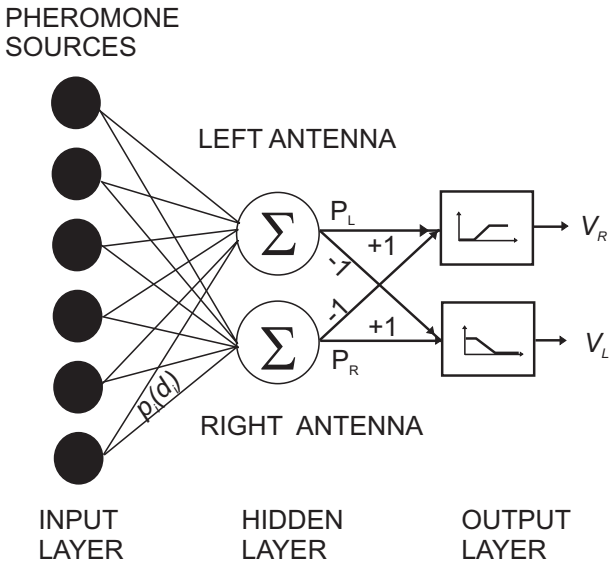


Fig. 5.15 Modelarea neuronala a interacțiunilor mediate de feromoni

In această structură, neuronii de pe layer-ul de intrare sunt sursele discrete de feromoni şi de aceea sinapsele între layer-ul de intrare şi layer-ul intermediar sunt determinate exclusiv de concentrațiile şi de distribuția spațială a surselor de feromoni. Am denumit *exo-sinapse*, acest tip de sinapse mediate de feromoni.

Observatie

Deoarece ponderile sinapselor între layer-ul ascuns şi layer-ul de ieşire sunt fixe, având valorile (+1, -1), iar ponderile exo-sinapselor sunt determinate doar de condiții externe, aceasta rețea nu are nevoie de training!!

Inteligența fiecarui agent dintr-un roi este determinată în întregime de roiul care generează o anumită distribuție de feromoni.

Modelarea neuronală a proceselor în care sunt implicati feromonii deschide o serie de perspective interesante pentru studiul proceselor de auto-organizare în roiuri (swarms) ca procese de învătare şi se sugerează posibilitatea salvării (la nivelul serverului de feromoni, de exemplu) a unor snap-shot-uri ale configurațiilor de feromoni pentru reproducerea (in acelasi roi, sau în roiuri diferite) unor situații de auto-organizare printr-un algoritm de tip follow the past.

Am descris aceasta abordare în ([156]).

b. Abordarea fuzzy

Valorile P_L şi P_R ale intensitatilor feromonilor, detectate cu *două antene* pot fi folosite pentru a exprima eroarea de reglaj:

$$e(t) = P_L(t) - P_R(t) \qquad (5.45)$$

Urmând procedura descrisă în paragraful 5.2.3, se poate implementa cu uşurință un controller fuzzy care sa aducă vehiculul în postura în care P_L=P_R. Aceasta se întâmplă atunci când punctele L şi R (antenele) sunt situate simetric față de traiectoria țintă, ceea ce corespunde situației în care centrul de masă al vehiculului este exact pe traiectoria țintă.

In mod similar, eroarea de reglaj se poate exprima prin relația:

$$e(t) = \phi_R(t) - \theta_0(t) \qquad (5.46)$$

unde $\phi_R(t)$ este valoarea calculată cu (5.42) iar $\theta_0(t)$ este orientarea curentă a vehiculului, raportată de subsistemul de localizare.

Un FLC implementat pornind de la exprimarea erorii cu (5.46) va tinde să mențină orientarea vehiculului tangentă la curba descrisa de un "look-ahead point", definit de rezultanta surselor de feromoni (5.42).

c. Abordarea de tip "pure pusuit"

Relația (5.41) exprima cantitativ intensitatea "campului de feromoni" intr-un punct situat la distantele $d_1, d_2, ..., d_N$ față de cele N surse discrete de feromoni. Pe de altă parte, conform ecuației (5.37) care descrie modelul feromonilor, intensitatea rezultantă P_R a câmpului de feromoni se poate exprima, ca şi cum campul ar fi creat de o singura sursa, cu intensitatea Np_0, aflată la distanța L de punctul considerat.

$$P_R = Np_0\left(1 - \frac{L}{\sigma}\right) \qquad (5.47)$$

Cunoscând P_R din (5.41) şi considerând sursele de intensitate unitară $p_0=1$, se poate determina distanța L:

$$L = \sigma\left(1 - \frac{P_R}{N}\right) \qquad (5.48)$$

Mai departe, se poate aplica direct algoritmul *pure pursuit*, alegând drept look-ahead point, punctul unde se află sursa echivalentă de feromoni, aflat la distanța L de punctul curent, pe o dreaptă cu orientarea dată de (5.42).

5.3.2 Conducerea roboților mobili cu ajutorul informației stocate în tag-uri RFID distribuite în mediu

În paragraful 3.5.2 am propus o metodă de localizare bazată pe o matrice de tag-uri RFID echidistanțe, încorporate în podea. Fiecare tag are pre-înregistrate coordonatele absolute punctului în care se găseşte, astfel încât un cititor aflat la bordul robotului poate determina poziția curentă doar citind conținutul tag-ului.

Marea majoritate a tag-urilor uzuale dispun de o capacitate de memorie de ordinul un kilooctet, mult mai mare decât cei câțiva octeți necesari pentru stocarea coordonatelor punctului curent, dar totuşi insuficientă pentru a stoca în fiecare tag întreaga hartă a mediului.

Această memorie, în principiu, poate fi folosită, fără costuri adiționale, pentru definirea unor traiectorii, sau alte informații de navigație (ex. existanta unor obstacole, bifurcații ale rutei curente, rute ocolitoare opționale etc.). Dificultățile provin din faptul că, la un moment dat, se poate citi un singur tag, aflat în zona de vizibilitatea a reader-ului, de ordinul câtorva centimetri. în aceste condiții, dacă am defini traiectoria țintă cu ajutorul unor trasee de feromoni digitali stocați în tag-uri, ca în figura 5.13, singura informație relativ la ruta pe care un robot o poate extrage citind tag-ul curent este dacă se află pe rută sau nu. Iar în cazul în care nu se află exact pe rută, nu are nici o posibilitate să afle în ce direcție se află ruta şi nici la ce distanță.

Pentru a depăşi aceste limitări, am propus în ([154]) o soluție bazată pe următoarele ipoteze:

- Robotii sunt dotați cu două cititoare, montate simetric fața de direcția de deplasare.
- Traiectoria se defineşte implicit sub forma unor trasee de feromoni digitali cu lățimea de 3-4 celule adiacente (vezi figura 5.16).
- Tag-urile stochează o mărime scalară, reprezentand intensitatea câmpului de feromoni în punctul curent. în figura 5.16, intensitățile feromonilor în diverse puncte ale hărtii sunt reprezentate prin nuanțe de gri. Culoarea alb corespunde lipsei complete a feromonilor în celula respectivă, iar negru corespunde intensității maxime.

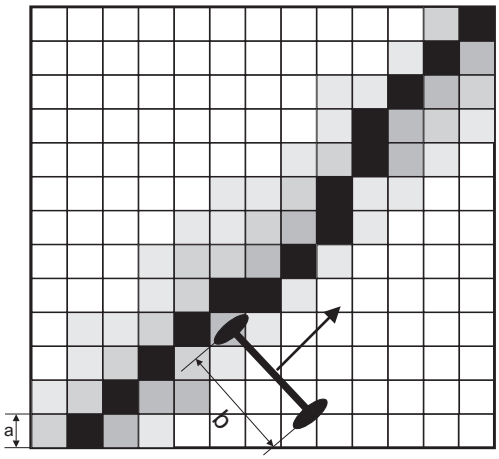


Fig. 5.16 Definirea implicită a unei traiectorii cu ajutorul feromonilor digitali stocati în tag-uri RFID

Cu acest mod de reprezentare a traiectoriei țintă, două cititoare RFID amplasate lateral, pot detecta gradienți ai intensității campului de feromoni şi aceasta informație poate fi exploatată, de exemplu cu un controller fuzzy similar cu cel descris în paragraful 5.3.1.4, pentru urmărirea traiectoriei.

Această abordare are avantajul ca emulează fidel feromonii naturali şi prezintă interes în situațiile în care se doreşte coordonarea unor formatiuni de roboti, care au posibilitatea sa amplifice intensitatea feromonilor pe lângă care trec. Există de

asemenea posibilitatea de a defini mai multe tipuri de feromoni stocati în aceleaşi tag-uri, fiecare codificând comportamente distincte.

Pentru aplicații în care se pune problema conducerii unor roboti individuali, pentru urmărirea cât mai fidelă a unor traiectorii, soluția aceasta are dezavantajul unor oscilații relațiv mari.

Pentru aceste aplicații, este mai avantajoasa o metoda de definire indirectă a traiectoriei țintă, renunțând la conceptul de feromoni digitali.

In acest scop, fiecare tag conține pe lângă coordonatele punctului curent, coordonatele precalculate ale unui look-ahead point (țintă intermediară), care pot fi direct folosite pentru un algoritm “pure pursuit” (vezi figura 5.17).

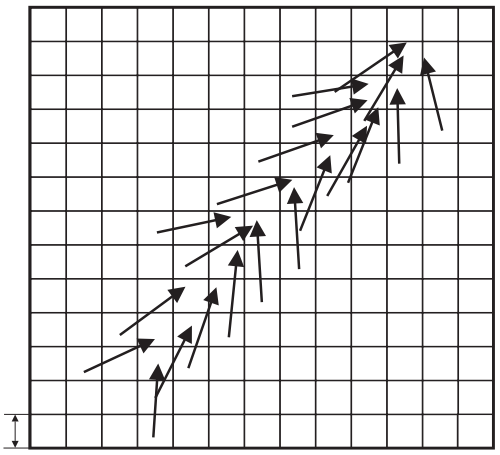


Fig. 5.17 Definirea indirectă a traiectoriei țintă cu ajutorul unor look-ahead points precalculate şi stocate în tag-uri RFID

Aceasta metodă are avantajul că foloseşte un singur reader RFID şi conduce la oscilatii mai mici, ceea ce este important pentru confortul persoanei asistate în cazul în care vehiculul autonom controlat este folosit pentru implementarea unui asistent robotic personal.

5.4 Sumarul contribuțiilor la rezolvarea problemei urmăririi unei traiectorii

Referitor la problema urmăririi unei traiectorii s-au propus două abordări diferite:

- Prima abordare se bazeaza pe conceptul de *feromoni virtuali*, încorporați într-o hartă a mediului stocată în memoria unui *"server de feromoni"*. Agenții autonomi "percep" distribuția spatială a feromonilor prin intermediul serverului, cu care se află în comunicație permanentă. Soluția are, între altele, avantajul că serverul preia o bună parte din calculele necesare pentru navigație. Dezavantajul principal derivă din viziunea centralizată, care reduce robustetea întegului sistem la defecțiuni (defectarea serverului, de exemplu, conduce la blocarea întregului sistem).
- Intr-o a doua abordare se propune folosirea informațiilor stocate într-o retea de tag-uri RFID pentru conducerea robotului la urmărirea unei traiectorii. Soluția aceasta are o mai buna robustețe la defecțiuni, datorita viziunii descentralizate de control, cu pretul unei flexibilități mai scazute în elaborarea soluțiilor de conducere.

Ambele abordări conduc la algoritmi simpli, uşor de implementat pe orice microcontroller de uz general.

6

... and not get hurt. Problema ocolirii obstacolelor

6.1 Formularea problemei

Abilitatea de a ocoli obstacolele existente între poziția curentă a robotului şi punctul țintă este o cerință fundamentală pentru orice aplicație practică care implică vehicule autonome de orice tip.

Soluțiile descrise în literatură pentru această problemă pot fi clasificate în următoarele două categorii:

- metode deliberative;
- metode reactive.

În *abordarea deliberativă*, se presupune că robotul dispune a-priori de o cunoaştere completă a mediului, inclusiv poziția obstacolelor şi ca mediul este static şi se pune problema planificării unei rute sigure şi optimale între punctul asociat cu poziția curentă a robotului şi punctul țintă. Ocolirea obstacolelor este, din această perspectiva, un sub-task al planificării rutei (path planning).

Odată obținută ruta, robotul execută mişcarea către țintă fără a avea nevoie de informații suplimentare despre mediu obținute de la senzori.

În *abordarea reactivă*, robotul se bazează pe senzori pentru a obține o imagine limitată, locală, despre mediu, urmărind să detecteze în timp real eventualele obstacole şi să-şi ajusteze traiectoria pentru ocolirea acestora.

Evident că, în practică, sunt extrem de rare situațiile în care mediul este pe deplin cunoscut şi complet static. In realitate, mediul este de obicei prea complex pentru a fi modelat cu uşurință şi este dinamic - oricând pot apărea obstacole neaşteptate. În plus, metodele deliberative, care implică prelucrarea unui volum mare de date, necesită resurse de calcul mult peste capacitatea unui microcontroller uzual şi, din aceste motive nu vor fi analizate în această tezã.

Pe de altă parte, nici algoritmii pur reactivi nu sunt lipsiți de dezavantaje, cel mai important fiind vulnerabilitatea față de erorile senzorilor.

6.2 Algoritmi cunoscuți pentru ocolirea obstacolelor

6.2.1 Algoritmii de tip "bug"

Cel mai simplu algoritm de evitare a obstacolelor descris în literatură este "the bug algorithm", propus de Lumelsky et. al. în ([157]). Conform acestuia, la întâlnirea unui obstacol, robotul înconjoară complet obstacolul pentru a găsi punctul de pe circumferința acestuia aflat la distanța minimă față de țintă, apoi părăseşte circumferința din acest punct (vezi figură 5.1).

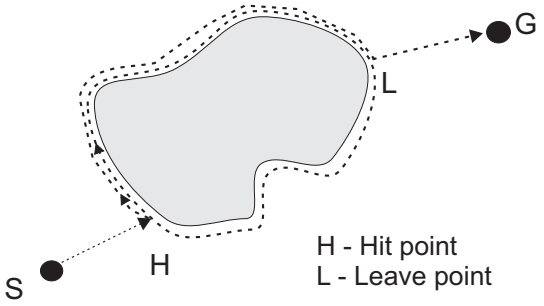


Fig. 6.1 O ilustrare a algoritmului "bug"

Acest algoritm este, în mod evident, foarte ineficient şi, din acest motiv au fost propuse mai multe îmbunătățiri ([158], [159]).

De exemplu, în "bug2", robotul începe să urmărească conturul obstacolului, dar îl părăseşte în momentul când intersectează segmentul care uneşte punctul de start cu punctul țintă (vezi figura 6.2).

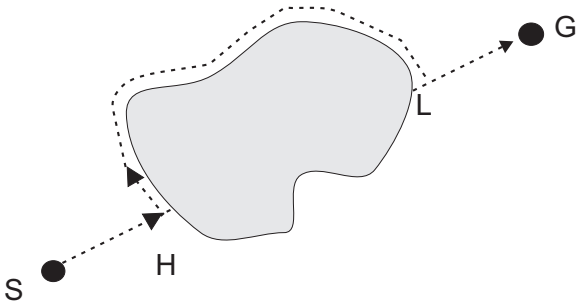


Fig. 6.2 O ilustrare a algoritmului "bug2"

In ciuda avantajului simplității, agloritmii de tip "bug" au câteva dezavantaje importante:

- Nu iau în considerare cinematica robotului, care este importantă în cazul vehiculelor non-holonomice;
- Iau în considerare doar cele mai recente valori raportate de senzori, ceea ce le face vulnerabile la erorile de masura;
- Sunt lente.

6.2.2 Algoritmul campului de potential (Virtual Potential Field Algorithm)

Algoritmii descrişi în continuare sunt deliberativi. Am inclus aici o scurtă descriere a lor, datorită unor particularităţi care prezintă similitudini cu soluţia propusă în paragrafele următoare.

Algoritmul câmpului de potenţial, descris în ([160],[161]) consideră comportamentul robotului similar cu al unei particule aflate sub acţiunea unor forţe virtuale de atracţie către ţintă şi de respingere din partea obstacolelor. Traiectoria roboptului este determinată de rezultanta acestor forţe (vezi figura 6.3).

Deşi deosebit de elegant, algoritmul câmpului de potenţial nu ţine seama de non-holonomicitatea vehiculului şi este dificil de folosit pentru aplicaţii în timp real.

A fost, de asemenea, criticat pentru performanţe scăzute la navigarea pe coridoare înguste şi pentru existenţa unor situaţii în care algoritmul nu găseşte soluţia de ocolire a obstacolului, deşi aceasta, în mod evident, există.

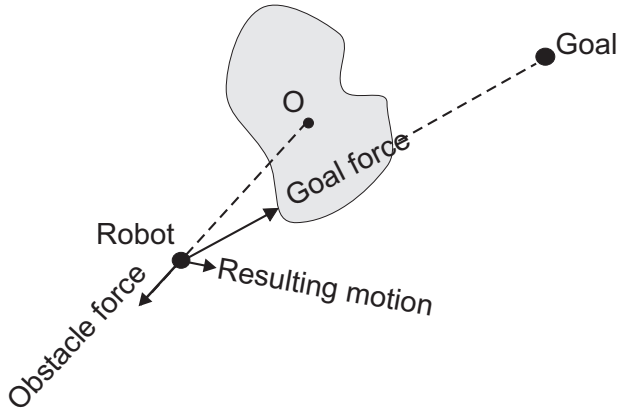


Fig. 6.3 O ilustrare a algoritmului câmpului de potențial

6.2.3 Algoritmul VFH (Vector Field Histogram)

Propus de Borenstein şi Koren în ([162]) şi ulterior îmbunătățit în ([163],[164]), algoritmul VFH, elimină în bună măsură problema erorilor de măsură la senzori, prin crearea unei histograme polare bazate pe mai multe măsurări succesive recente, ca în figura 6.4.

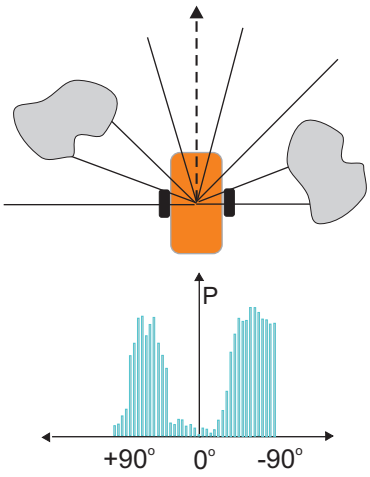


Fig. 6.4 Histograma polară folosita de algoritmul VFH

In figura 6.4, axa x reprezintă unghiurile ascociate cu citirile sonarelor, iar axa y reprezintă probabilitatea P ca sa existe un obstacol în direcția respectivă.

Probabilitățile se calculează în baza unei hărți grid a mediului din jurul robotului şi sunt ajustate prin măsurări repetate.

Histograma polară este folosită pentru a identifica pasajele destul de mari pentru a permite trecerea robotului. Selecția unui anumit pasaj se face prin evaluarea unei funcții cost, definită pentru fiecare pasaj. Aceasta depinde de alinierea orientării robotului pe direcția țintei şi de diferența între orientarea curentă şi direcția rutei ocolitoare. În final, se selectează pasajul cu funcția cost minimă.

Algoritmul VFH ține seama de geometria şi cinematica robotului şi oferă soluție la problema măsurărilor afectate de zgomot, dar nu poate trata erorile sistematice ale sonarelor. O astfel de situație de eroare sistematică este ilustrată în figura 6.5.

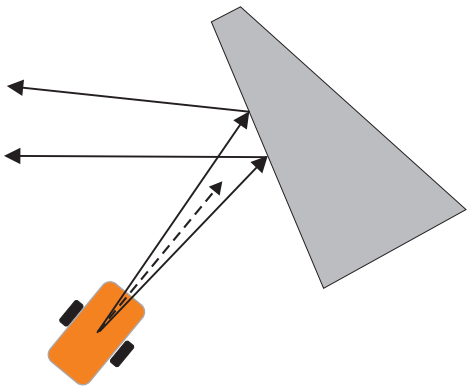


Fig. 6.5 Un exemplu de eroare sistematică la senzorii de tip sonar

VFH este o abordare hibridă combinând elementele deliberative cu cele reactive. Singurul dezavantaj al acestei soluții este acela că implică un volum considerabil de calcule, ceea ce îl face dificil de implementat pe microcontrollere.

6.2.4 Metoda "Bubble Band"

Propusă de Khatib şi Quinlan în ([165]), această metodă defineşte o "bula" (bubble) conținând spațiul maxim disponibil (lipsit de obstacole) în jurul robotului, care poate fi parcurs în orice direcție fără coliziuni.

Forma şi dimensiunile aceste bule sunt determinate de un model simplificat al geometriei robotului şi de informațiile provenite de la senzori (vezi figura 6.6).

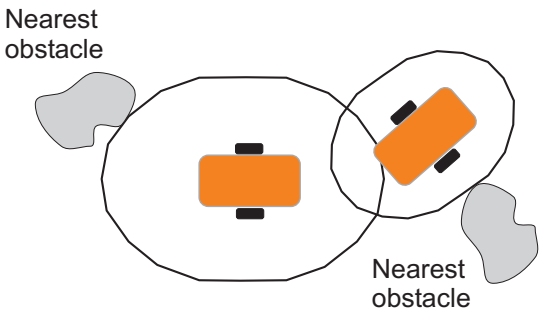


Fig. 6.6 Ilustrarea conceptului de bubble band

Cu acest concept, o bandă de astfel de bule se poate folosi pentru planificarea unei rute între punctul de start şi țintă.

6.2.5 Alte metode de ocolire a obstacolelor

Exista desigur şi alte metode de ocolire a obstacolelor. Totuşi, relațiv puține pot fi implementate pe microcontrollere de uz general, pentru aplicații în timp real.

Printre acestea, soluțiile bazate pe logica fuzzy, cum sunt cele descrise în ([166],[167]) pot fi integrate ca o extensie logică în soluțiile de urmărire a unei traiectorii bazate pe FLC, descrise în paragraful 5.2.3.

6.3 Un nou algoritm de ocolire a obstacolelor. Algoritmul “bubble rebound”.

6.3.1 Detectia obstaclelor

Se considera un vehicul, având un inel de senzori ultrason echidistanti, acoperind un unghi de 180 grade, ca în figura 6.7.

Daca N este numărul se senzori, următoarea secvența de cod defineşte o “bula de senzitivitate” în jurul robotului:

```
unsigned int sonar_readings[N];
unsigned int bubble_boundary[N];
bubble_boundary[i]=K_i*V*delta_t;
int check_for_obstacles(void)
{
for(i=0;i<N;i++)
  {
  if(sonar_readings[i]<=bubble_boundary[i]
                  return(1);
     else return(0);
     }
}
```

Unde, *V* este viteza de translație a robotului, *delta_t* este timpul între două evaluări succesive ale informației de la senzori, iar Ki sunt constante de scalare folosite pentru acord.

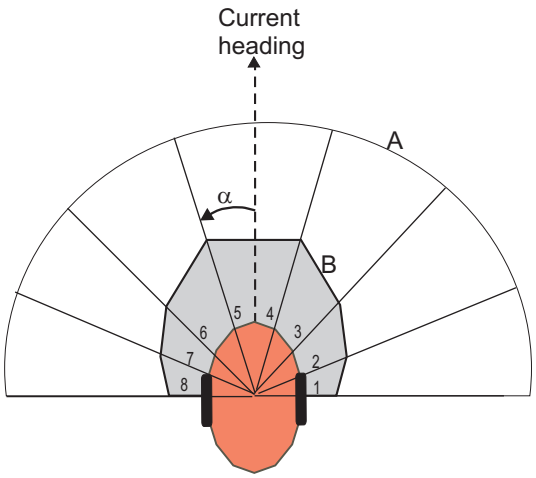


Fig. 6.7 Definirea zonei de sensibilitate la obstacole

Forma şi dimensiunile bulei de senzitivitate (curba B din figura 6.7) depind de forma geometrică a robotului (măsuratorile sonarelor reflectă distanța între circumferința robotului şi obstacole) şi de spațiul parcurs de robot între două citiri succesive ale senzorilor, cu condiția ca acest spațiu să nu depăşească distanța maximă pe care o pot măsura sonarele (curba A din figura 6.7).

Deoarece sonarele sunt distribuite uniform pe circumferință, acoperind un unghi de 180 de grade, valorile citite de acestea se pot reprezenta într-o diagramă polară, ca în figura 6.8.

6.3.2 Descrierea algoritmului de ocolire a obstacolelor

În lipsa obstacolelor, robotul se îndreaptă spre țintă în linie dreaptă. În momentul în care un obstacol este detectat în interiorul bulei de senzitivitate, robotul “ricoşează” într-o direcție caracterizată prin densitate minimă de obstacole şi îşi continuă drumul în linie dreaptă în această direcție până când ținta devine vizibilă (i.e. nu există

obstacole în zonă de vizibilitate, în direcția țintei), sau până când un nou obstacol este detectat, conform schemei logice din figura 6.9

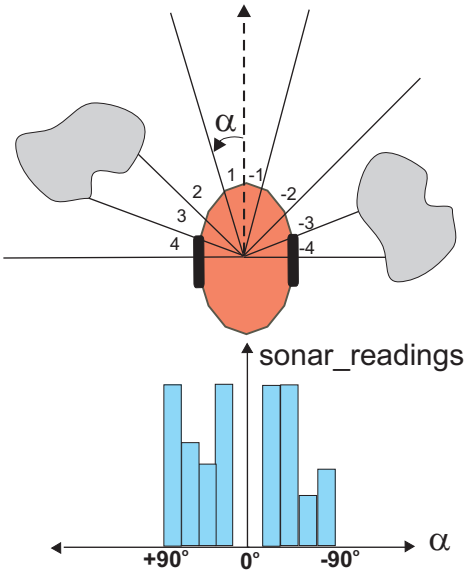


Fig. 6.8 Reprezentarea în diagrama polară a informației de la sonare

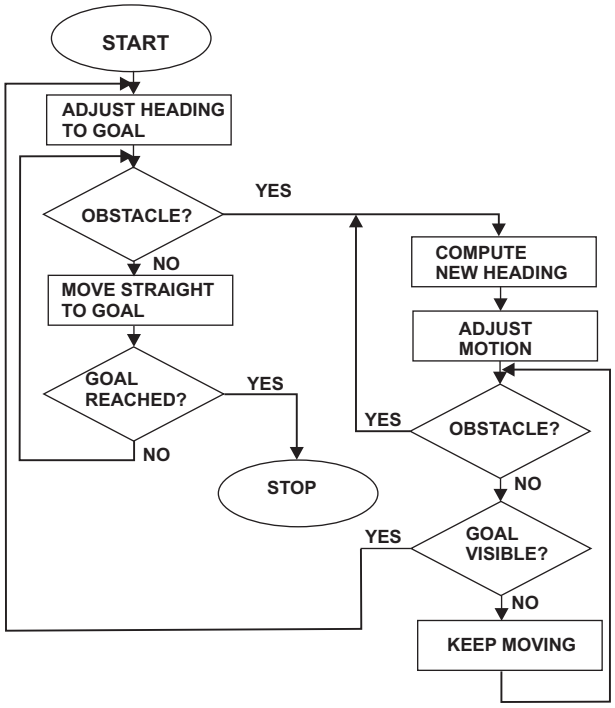


Fig. 6.9 Schema logică a procesului de ocolire a obstacolelor

In figura 6.10 este prezentată traiectoria robotului deviată la întâlnirea unui obstacol şi reluarea mişcarii catre țintă, în momentul când se recapătă vizibilitatea asupra țintei.

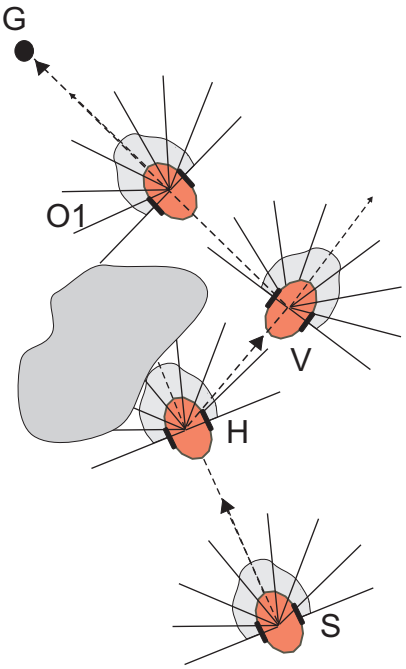


Fig. 6.10 Ilustrarea mecanismului de ocolire a obstacolelor prin ricoşeu

In figura 6.10, H este punctul în care este detectat obstacolul (Hit-point), iar V este punctul în care ținta redevine vizibilă.

6.3.3 *Calculul Unghiului de Ricoşeu*

Daca α_0 este pasul unghiular de distribuție a sonarelor pe circumferinta robotului:

$$\alpha_0 = \frac{\pi}{N} \qquad (6.1)$$

atunci, indexul i al sonarului conține informație despre poziția unghiulară a acestuia:

$$\alpha_i = i\alpha_0$$
$$i \in \left[-\frac{N}{2}, \frac{N}{2}\right] \qquad (6.2)$$

Cu aceste notații, cel mai simplu mod de a determina direcția cu densitate minimă de obstacole este sa calculăm media ponderată a unghiurilor asociate cu sonarele, folosind ca factori de ponderare distanța până la obstacole (6.3). În acest mod, direcțiile cu obstacole apropiate primesc ponderea minimă în sumă şi sunt evitate.

Evident, acest mod de calcul al unghiului de ricoşeu α_R nu are nici o legatură cu ricoşeul mecanic rezultat în urma unei ciocniri elastice.

$$\alpha_R = \frac{\sum_{i=-\frac{N}{2}}^{\frac{N}{2}} \alpha_i D_i}{\sum_{i=-\frac{N}{2}}^{\frac{N}{2}} D_i} \qquad (6.3)$$

In figura 6.11 este ilustrat unghiul de ricoşeu, pentru o configurație simplă a obstacolelor.

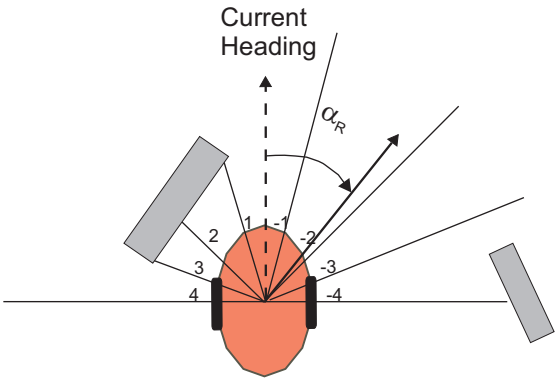


Fig. 6.11 O ilustrare a modului de calcul a unghiului de ricoseu

Rezultate similare se obtin aplicând o metodă inspirată de conceptul de *câmp de potential virtual,* considerând ca robotul este supus unor forte de atracție cu atât mai mari cu cât distanța până la obstacole este mai mare. Totuşi, am preferat metoda descrisă de relația (6.3) pentru că aceasta conduce la calcule mult mai simple şi – pornind de la aceiasi senzori – dă rezultate comparabile.

6.4 Concluzii

Algoritmul descris are următoarele dezavantaje:

- Este departe de a fi optimal;
- Ca majoritatea algoritmilor pur reactivi, necesită un planner de nivel superior pentru a rezolva problema navigației intr-un mediu de tip labirint. De exemplu, în figura 6.12, algoritmul este incapabil să conducă robotul de la *"Start"* la punctul *"Final goal",* dar funcționează perfect dacă se definesc țintele intermediare *Goal1* şi *Goal2*.

- Mişcarea nu este lină.
- Esecul este posibil chiar dacă există o cale catre țintă.
- Mişcarea este inițiată chiar dacă nu există o cale catre țintă.

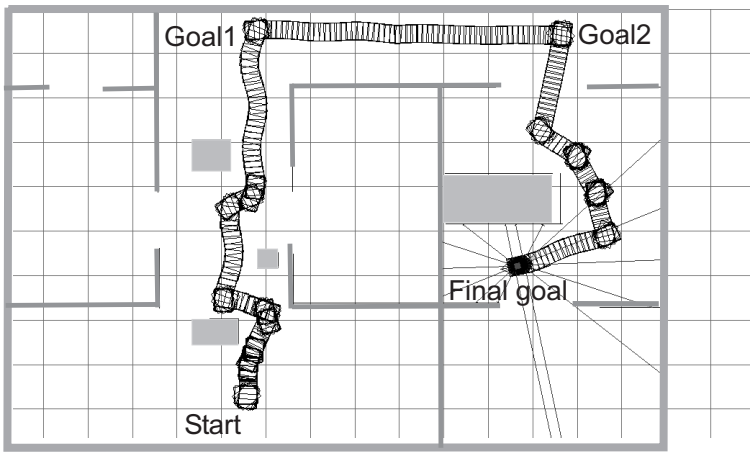


Fig. 6.12 Un exemplu de navigație intr-un mediu tip labirint, cu ținte intermediare

In ciuda dezavantajelor enumerate, exista şi câteva avantaje care îl fac demn de atenție:

- Implementarea algoritmului necesită un volum foarte mic de calcule, fapt care îl recomanda pentru aplicații embedded.
- Funcționează cu senzori ieftini
- Poate fi uşor adaptat pentru altfel de senzori
- Ține cont de geometria robotului
- Este destul de rapid
- Funcționează bine pe coridoare inguste
- Poate detecta şi ocoli orice fel de obstacole, inclusiv unele obstacole aflate în mişcare, cu condiția ca viteza de deplasare a acestora sa fie comparabilă cu viteza robotului.

Principala limitare vine de la performanțele reduse ale senzorilor ultrason, care sunt afectați de erori sistematice (vezi figura 6.5).

Folosirea unor senzori laser, de genul celor produsi de Loke ([84]) elimină acest dezavantaj şi poate conduce la performanțe net mai bune.

Pe ansamblu, rezultatele obținute cu acest algoritm, pe care l-am denumit *algoritmul "bubble rebound*", sunt promițătoare şi merită luat în considerare ca o posibilă soluție de reducere a costurilor de implementare a unor asistenți robotici personali.

7

Exerimente de implementare in timp real

7.1 Delimitări

Experimentele desfăşurate în cadrul programului de cercetare descris în prezenta tezã sunt subordonate obiectivului general de a verifica fezabilitatea ideii de implementare a conceptului de asistent robotic personal, folosind un sistem distribuit de module încorporate (embedded), bazate pe microcontrollere de uz general.

Practic, aceasta înseamnă implementarea şi testarea algoritmilor descrişi pentru fiecare din subproblemele navigației roboților autonomi.

Totuşi, având în vedere resursele de timp şi materiale foarte limitate disponibile, au fost necesare câteva opțiuni de compromis.

Astfel, am ales să nu implementăm soluțiile de localizare prezentate în capitolul 3, apreciind că acestea, deşi sunt importante în contextul realizării unui PRA, sunt doar indirect legate de domeniul ingineriei sistemelor. În consecință, în toate experimentele descrise în acest capitol, ne-am bazat pe sistemul odometric al roboților pentru localizare.

De asemenea, atunci când s-au propus două soluții distincte pentru aceeaşi problemă, am optat pentru implementarea şi testarea unei singure soluții, de regulă

cea care a necesitat resurse materiale minime. Din acest motiv, nu am abordat experimental soluțiile bazate pe tehnologia RFID.

Viteza referință pentru vehiculele autonome controlate a fost fixată la 0.5m/s, valoare comparabilă cu viteza de deplasare a unui om, la mers relaxat în interiorul unei locuințe.

Microcontrollerele folosite în experimente au resurse medii: 128Kb memorie Flash, 4Kb RAM, 4Kb EEPROM, 16Mips. În aceste condiții, am impus ca durata maximă de execuție a unui ciclu de program cu structura generală:

```
while (TRUE)
    {
    PORTB.0=1; //marker pentru inceput de ciclu
    Task_1();
    Task_2();
    .....
    Task_N();
    PORTB.0=0; //marker pentru sfarsit de ciclu
    }
```

să nu depasească 200ms.

7.2 Descrierea dispoziţivului experimental

Roboții folosiți pentru experimente sunt Pioneer-3DX şi PeopleBot, produşi de MobileRobots ([11]), prezentați în figura 7.1.

Fig. 7.1 Roboții folositi pentru experimente

Ambii roboți sunt de acelaşi tip, cu două roți motoare şi tracțiune diferențială şi au acelaşi model cinematic.

MobileRobots oferă şi un excelent simulator software al roboților, denumit MobileSim. Interfața grafică a acestuia este prezentată în figura 7.2.

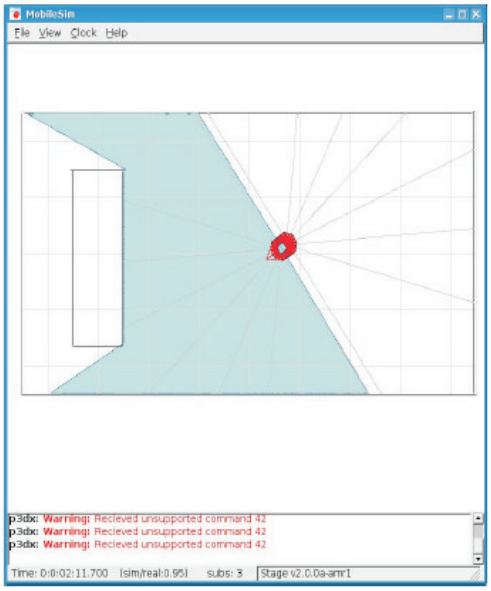


Fig. 7.2 Interfața grafică a simulatorului MobileSim

Electronica de la bordul robotului are structura din figura 2.3, iar comunicația între robot şi nivelele de conducere superioare se face printr-o linie seriala RS232.

Acest fapt sugerează următoarea structura hardware a unui asistent robotic personal implementat conform principiilor descrise în prezenta teză (fig. 7.3):

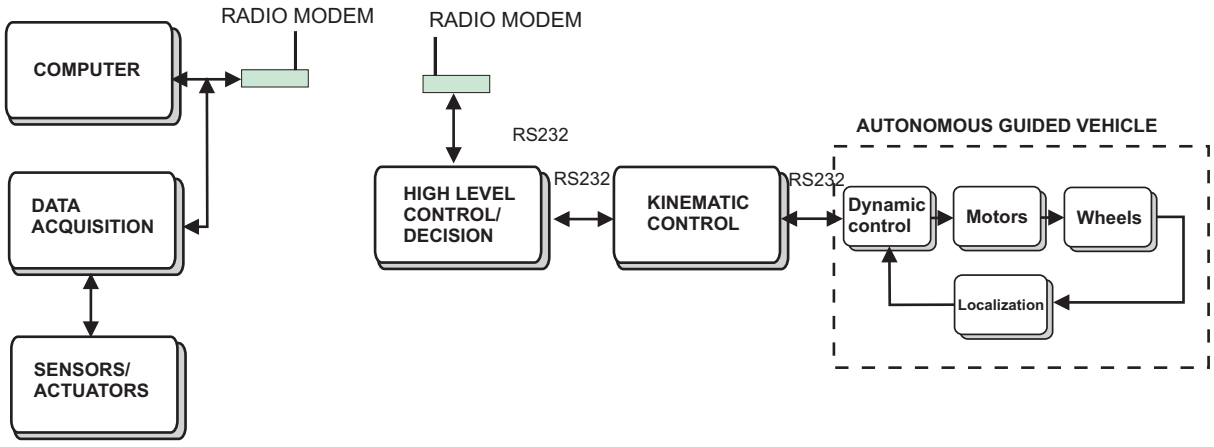


Fig. 7.3 Structura hardware a unui PRA

In această structură, blocurile denumite “Control cinematic” şi “Decizie” se reduc la un microcontroller în configurație single chip, echipat cu două interfețe seriale.

Pe parcursul experimentelor, din ratiuni de economie de timp, structura generală din figura 7.3 a fost simplificată, dupa cum urmează:

Pentru experimentele legate de implementarea şi testarea limbajului de programare RDPL, s-a folosit o structură ca în figura 7.4.

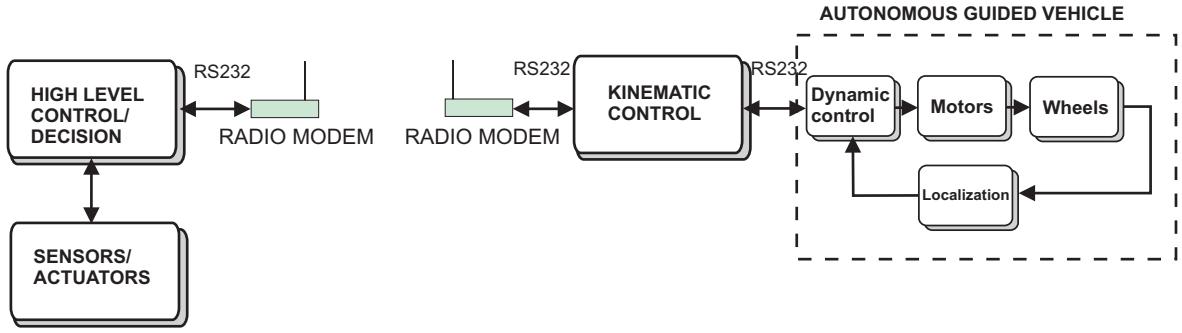


Fig. 7.4 Structura echipamentului folosit pentru testarea RDPL

Simplificarea constă în includerea modulului de achizitii de date în blocul de decizie.

Pentru implementarea şi testarea algoritmilor de urmărire a traiectoriei şi ocolire a obstacolelor s-a folosit o structură ca în figura 7.5.

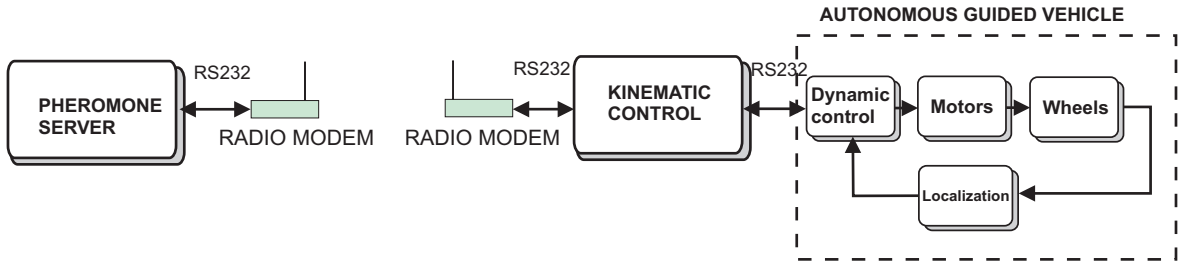


Fig. 7.5 Echipamentul folosit pentru testarea algoritmilor de navigație

In toate situațiile, robotul poate fi direct înlocuit de un computer care rulează simulatorul software MobileSim. Deoarece acesta este proiectat sa comunice cu echipamentul de control printr-o retea Ethernet, iar microcontrollerele folosite dispun doar de interfete seriale RS232, a fost necesara o aplicație software specială, denumită *“Ethernet to Serial Connector”*, produsă de Eltima Software ([168]).

7.3 Evaluarea implementării limbajului RDPL

Un program RPLD se stochează în memoria nevolatilă de tip EEPROM. Pentru microcontrollerul folosit, capacitatea EEPROM disponibilă este de 4Kb. Având în vedere că lungime medie a codului care defineşte o funcție este de 6 octeți, rezultă că dimensiunea maximă admisibilă pentru un program este de 800 de funcții. Această valoare este mult peste dimensiunea necesară, având în vedere că fiecare funcție codifică operații logice destul de complexe.

Din punct de vedere al limitării duratei de execuție a unui ciclu la 200ms, având în vedere că timpul mediu (măsurat) de execuție a unei funcții este de 0.4ms, rezultă că numărul maxim de funcții care pot fi definite într-un program este 500.

Deoarece în practică, numărul de funcții necesar pentru a descrie situațiile uzuale este de ordinul câtorva zeci, considerăm că limitarea dimensiunii programului la 500 de funcții este satisfăcătoare.

7.4 Experimente pentru studiul unui algoritm de urmărire a traiectoriei bazat pe conceptul de feromoni virtuali

La nivelul microcontrollerului pentru conducere cinematică s-a implementat un algoritm de tip fuzzy, descris în paragraful 5.2.3, cu trei subdomenii fuzzy pentru eroare şi derivata erorii. Eroarea de reglaj luată în considerare a fost $e(t) = P_L(t) - P_R(t)$, diferența între intensitatile feromonilor sesizate de două “antene virtuale”, amplasate lateral, simetric fața de axa robotului.

Pentru serverul de feromoni, a fost realizată o aplicație dedicată. Detalii privind implementarea sunt prezentate în anexa B.

In figura 7.7 este prezentată traiectoria robotului, inregistrata cu MobileSim, pentru harta de feromoni din figura 7.6, corespunzător uneri erori initiale de orientare de 45 grade.

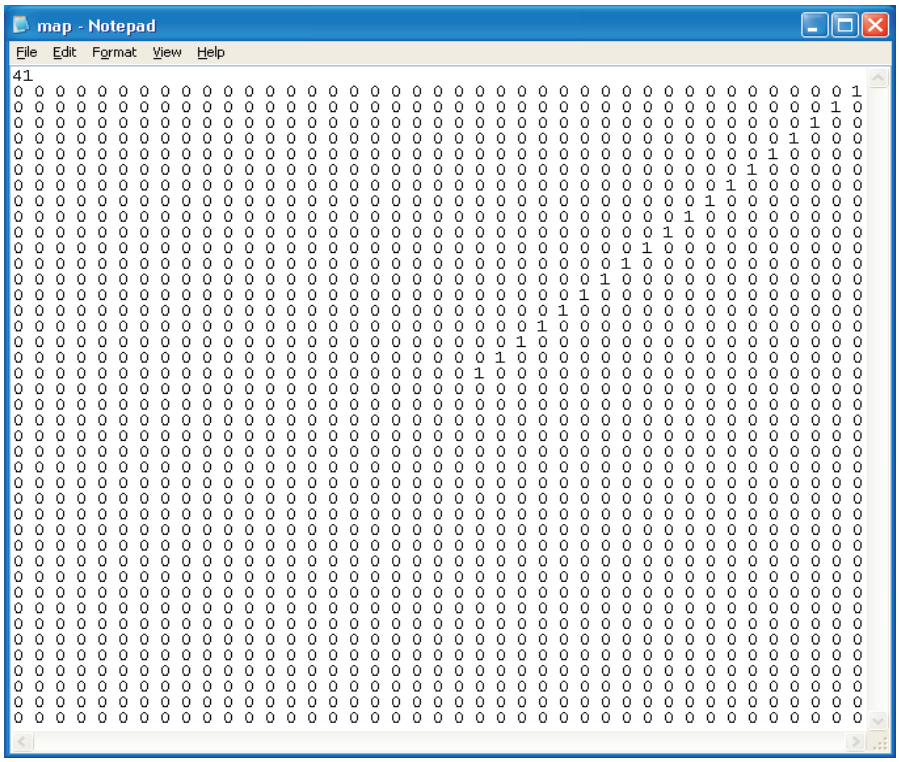


Fig. 7.6 Un exemplu de reprezentare a hărții mediului, încorporând informații despre distribuția de feromoni

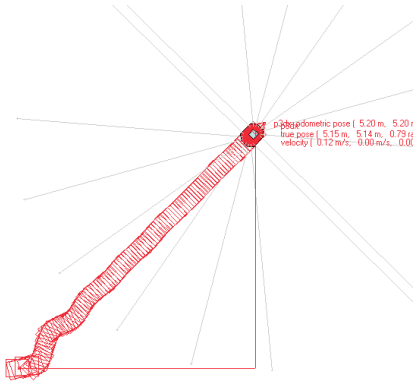


Fig. 7.7 Traiectoria inregistrată corespunzător hărtii din fig. 7.6

Figura 7.8 prezinta o traiectorie simplă, care conține un viraj în unghi drept.

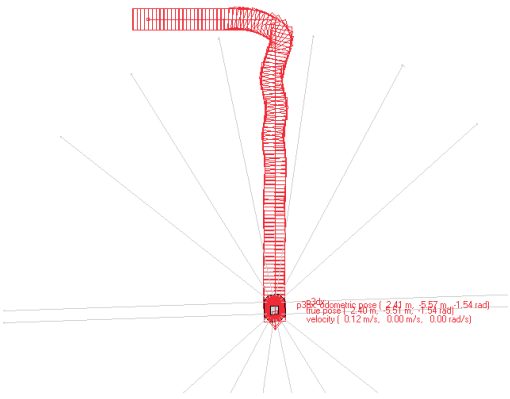


Fig. 7.8 Înregistrarea unei traiectorii care conține un viraj în unghi drept

Iar figurile 7.9 şi 7.10 sunt înregistrari ale traiectoriei robotului, pentru curbe țintă oarecare.

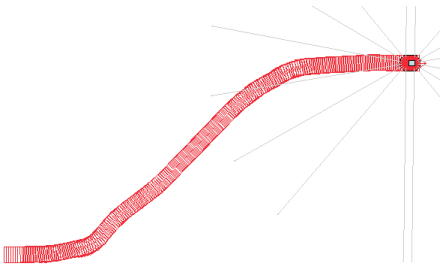


Fig. 7.9 Înregistrarea comportamentului robotului pe o traiectorie curbă în S

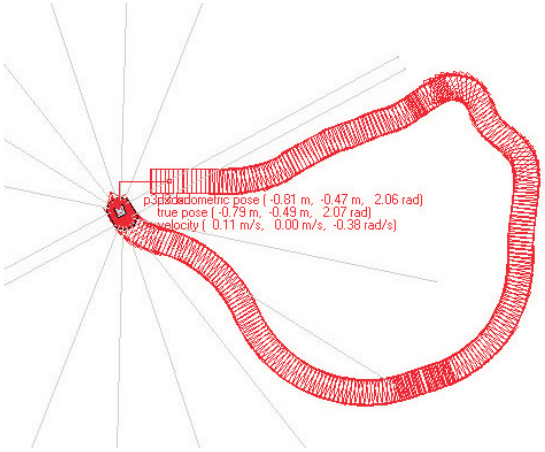


Fig. 7.10 Înregistrarea comportamentului robotului pe o traiectorie oarecare

Înregistrările au fost făcute în următoarele condiții:

- Dimensiunea unei celule a hartii grid, care încorporează distribuția de feromoni $a = 200mm$;
- Sensibilitatea receptorilor de feromoni $\sigma = 600mm$;
- Distanța între receptorii de feromoni dreapta-stanga: $b = 380mm$;
- Viteza de deplasare a vehiculului de-a lungul traiectoriei: $v = 300mm/s$;
- Intervalul de timp între două corecții succesive ale vitezelor referința $\Delta t = 200ms$.
- Viteza de comunicație între robot şi serverul de feromoni 9600 baud.

Experimental s-a constatat o tendință de oscilație a sistemului la creşterea vitezei *v*, sau la mărirea intervalului Δt. Sunt posibile următoarele căi de îmbunătățire a performanțelor sistemului:

- Mărirea sensibilității σ a receptorilor de feromoni. Aceasta corespunde cu mărirea distanței de anticipare (lookahead distance) *L*, pentru un controller de tip "pure pursuit" şi conduce la creşterea stabilității, cu prețul unei tendințe de "tăiere a colțurilor".
- Creşterea numărului de subdomenii fuzzy de la 3 la 5. Aceasta ar conduce la o ieşire a FLC reglabilă mai fin.
- Acelaşi efect se poate obține, micşorând intervalul de timp Δt între două corecții succesive. În acest scop este necesară folosirea unui microcontroller mai rapid (un ARM, de exemplu) şi folosirea unui echipament de comunicație wireless mai performant, care sa reducă timpul de latentă datorat schimbului de informații între robot şi serverul de feromoni.

Cu aceste măsuri, apreciem că se poate realiza cu uşurința un vehicul capabil sa se deplaseze cu 0.5-0.7m/s, fără oscilatii supărătoare, pe o traiectorie definită cu ajutorul conceptului de feromoni virtuali.

7.5 Experimente pentru evaluarea algoritmului “bubble rebound” de ocolire a obstacolelor.

Avantajele şi dezavantajele algoritmului “bubble rebound” au fost descrise în paragraful 6.4. În continuare, în figurile 7.11, 7.12, 7.13 sunt prezentate câteva capturi de imagini realizate cu MobileSim, în diverse configurații de obstacole.

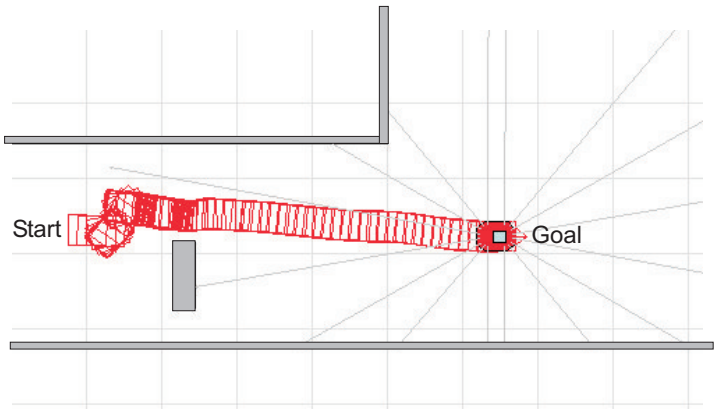


Fig. 7.11 Ilustrarea procesului de ocolire a unui obstacol

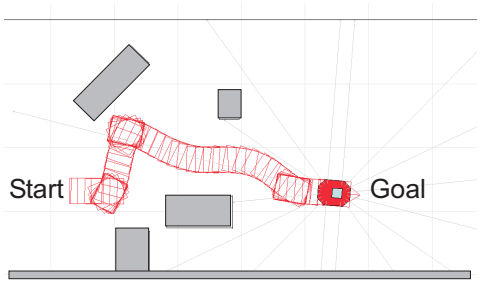


Fig. 7.12 Navigație pe un coridor cu obstacole multiple

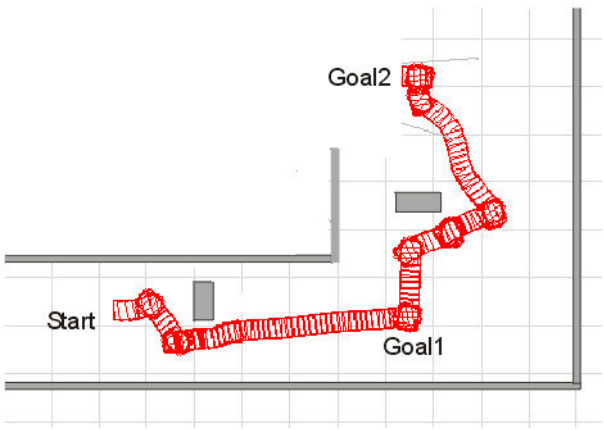


Fig. 7.13 Navigație cu definirea unor ținte intermediare

Testarea algoritmului s-a facut în următoarele condiții:

- S-a implementat un algoritm de conducere fuzzy pentru urmărirea unei traiectorii rectilinii între punctul definit de poziția curenta a robotului şi un punct țintă specificat printr-o comandă transmisă wireless pe linia serială.
- Viteza medie de deplasare a fost fixată la 0.3m/s.
- Pentru detectarea obstacolelor s-au folosit exclusiv senzorii ultrason cu care sunt echipati robotii Pioneer3 şi Peoplebot. Doar cei 8 senzori situați în partea anterioară a robotului au fost luați în considerare.

La aceasta viteză, rata eşecurilor (obstacole nedetectate) a fost practic zero, chiar la teste efectuate pe un coridor aglomerat, pe care circulau oameni. La viteze mai mari (0.5m/s) s-a înregistrat un număr de eşecuri (5-10% din total obstacole), datorate în principal erorilor sistematice de masură ale sonarelor, descrise în figura 6.5.

Aceste date sugerează concluzia că algoritmul bubble rebound poate fi atât de bun pe cât sunt de buni senzorii folosiți pentru detectarea obstacolelor.

8

Concluzii

8.1 Sumarul contribuțiilor cercetării

Această tezã a prezentat rezultatele unor cercetări destinate să exploreze fezabilitatea ideii de implementare a sisteme de conducere a robotilor de serviciu bazate pe microcontrollere low-cost/low-power.

S-a propus o definiție a asistentului robotic personal ca vehicul autonom capabil să transporte persoana asistată (intelligent wheelchair), sau să o asiste la deplasare (intelligent walker), fiind în acelaşi timp capabil să se deplaseze independent într-un "mediu inteligent" (smart environment), interacționând în mod programat cu o serie de senzori şi actuatoare distribuite în mediu.

Au fost abordate, dintr-o perspectivă interdisciplinara, principalele probleme ale navigației roboților autonomi: localizarea, decizia, urmărirea unei traiectorii şi ocolirea obstacolelor.

În domeniul localizării roboților autonomi, a fost descrisă (in paragraful 3.5) o metodă şi un dispozitiv de determinare simultană a distanței şi orientării roboților față de o baliză ultrason special proiectată.

În domeniul programării deciziilor roboților autonomi, a fost creat un limbaj de programare special (cap. 4), pe care l-am denumit RDPL (Robot Decision Programming Language), care rezolvă satisfăcător majoritatea problemelor legate de

formalizarea descrierii interacțiunii între robot şi mediul inteligent, fiind în acelaşi timp suficient de simplu pentru a putea fi folosit de end-useri cu cunoştințe tehnice minimale.

În ce priveşte problema urmăririi unei traiectorii, s-a propus (în paragraful 5.3) o metodă originală de definire şi urmărire a unei traiectorii cu ajutorul conceptului de "feromoni virtuali", stocați în memoria unui "server de feromoni".

A fost elaborat modelul matematic al feromonilor virtuali (paragraful 5.3.1.1) precum si un model neuronal (paragraful 5.3.1.4) al formațiunilor de agenți autonomi (swarms), model care aduce clarificări şi deschide noi direcții de cercetare în studiul swarm intelligence.

A fost, de asemenea, propus (in paragraful 6.3) un algoritm reactiv nou de evitare în timp real a obstacolelor, algoritm pe care l-am denumit "bubble rebound".

În paralel (in paragrafele 3.5.2. si 5.3.2), au fost explorate posibilitățile de folosire şi s-au propus soluții bazate pe tehnologia RFID pentru localizarea şi conducerea roboților autonomi la urmărirea unei traiectorii.

Majoritatea contribuțiilor menționate au fost supuse spre validare atenției comunității ştiințifice, prin prezentarea şi publicarea unui număr considerabil de articole, la conferințe internaționale de prestigiu (listate in paragraful 1.4).

8.2 Direcții de cercetare viitoare

Conceptul de feromoni virtuali a fost explorat, în prezenta tezã, doar din perspectiva conducerii unor agenți individuali, deşi principalele aplicații ale acestor principii sunt mai degrabă legate de conducerea unor formațiuni compuse din zeci, sute, sau chiar mii de agenți. Prezența unui server de feromoni permite memorarea şi ulterior transferul unor comportamente dobândite printr-un proces de optimizare de tip ACO (Ant Colony Optimization). Această analiză depăşeşte, în mod evident,

cadrul prezentei cercetări, dar ideile merită reluate într-un context legat de controlul unor formațiuni de roboți cooperativi.

Merită de asemenea atenție continuarea cercetărilor privind posibilitățile de folosire a tehnologiei RFID pentru conducerea roboților individuali, sau a formațiunilor de roboți.

Sumarul rezultatelor activitatii de cercetare a autorului

Lista lucrarilor publicate

I. Carti in edituri internationale de renume

[1] **Susnea I**. Mitescu M. *Microcontrollers in practice* ISBN: 3540253017, Springer Verlag, New York, Heidelberg, 2005

II. Carti in edituri romanesti recunoscute de CNCSIS

[2] Grigore Vasiliu, **Ioan Susnea** *Sisteme computerizate de masurare* ISBN: 978-973-755-452-9, Editura Matrix Rom, Bucuresti, 2009
[3] **Ioan Susnea**, Griogore Vasiliu *Sisteme distribuite pentru monitorizarea si conducerea proceselor. O introducere practica*, ISBN: 978-973-755-542-5, Editura Matrix Rom, Bucuresti, 2009.
[4] Grigore Vasiliu, **Ioan Susnea** *Mecatronica si microsisteme de actionare*, ISBN: 978-973-755-546-5, Editura Matrix Rom, Bucuresti, 2009

III. Brevete de inventie

[5] John W. Halpern, **Ioan Susnea** *Phone diversifications*, WO/2005/107299, International Application No.: PCT/IB2005/001161, 2005. Detalii in baza de date WIPO, la adresa: www.wipo.int/pctdb/en/wo.jsp?wo=2005107299
[6] **Ioan Susnea**, Grigore Vasiliu *Sistem de localizare pentru roboti mobili si vehicule autonome*, Cerere inregistrata la OSIM sub numarul A/01021, din 04-12-2009

IV. Lucrari publicate in jurnale indexate ISI

[7] **Susnea I.** Vasiliu G. Filipescu A. Radaschin A., *Virtual Pheromones for Real-Time Control of Autonomous Mobile Robots*, in Studies of Informatics and Control, Vol 18, issue 3, 2009, pp: 233-240., ISSN:1220-1760

V. Lucrari publicate in periodice interne recunoscute de CNCSIS

[8] **Susnea I**, Filipescu Adrian, Filipescu Adriana, Coman G., Wheeled Mobile Robot Control Using Virtual Pheromones, Petroleum-Gas University of Ploiesti Bulletin, Tehnical Series, Vol LXI, no.3/2009, ISSN 1224-8495, pp. 117-126.

VI. Lucrari publicate in volumele unor conferinte internationale indexate ISI

[9] **Susnea I**. Vasiliu G, Filipescu A, Coman G., On the Implementation of a Robotic Assistant for the Elderly. A Novel Approach, 7th WSEAS Int. Conf. on Computational, Iintelligence Man-machine Systems and Cybernatics (CIMMACS '08), Cairo, Egipt, December 29-31, 2008, pp.215-220, ISBN 978-960-474-049-9, ISSN: 1790-5117 Proceedings + Book published by WSEAS Press that participate in ISI and all the other important international citation indices: www.worldses.org/indexes, (Proceedings ISI quoted, www.ieeexplore.ieee.org) Published by WSEAS Press
[10] **Susnea I**, Filipescu A, Vasiliu G., Filipescu S., Path Following, Real-time, Embedded Fuzzy Control of a Mobile Platform Wheeled Mobile Robot, IEEE

International Conference on Automation and Logistics, ICAL 2008, 1-3 Sep., Qingdao, China, IEEE ICAL 2008 CD Proceedings, pp.91-96, IEEE Catalog Number: CFP08CAL, ISBN: 978-1-4244-2503-7, Library of Congress: 2008903794 (Proceedings ISI quoted, www.ieeexplore.ieee.org).
[11] **Susnea I**, Vasiliu G, Filipescu A, Real-Time, Embedded Fuzzy Control of the Pioneer3-DX Robot for Path Following, Proceedings of 12th WSEAS International Conference on SYSTEMS, Heraklion, Greece, July 22-24, 2008, pp.334-338, ISBN: 978-960-6766-83-1, ISSN: 1790-2769, Published by WSEAS Press (www.wseas.org, www.worldses.org/indexes).
[12] **Susnea I** . Vasiliu G, Filipescu A, RFID Digital Pheromones for Generating Stigmergic Behaviour to Autonomous Mobile Robots, 4th WSEAS Int. Conf. on Dynamic Systems and Control (Control '08), CORFU ISLAND, GREECE, October 26-28, 2008, pp.20-24, ISBN 978-960-474-014-7, ISSN: 1790-2769 Proceedings + Book published by WSEAS Press that participate in ISI and all the other important international citation indices: www.worldses.org/indexes, (Proceedings ISI quoted, www.ieeexplore.ieee.org) Published by WSEAS Press.
[13] Filipescu A, **Susnea I**, Stancu Al, Stamatescu G, Path following, real-time, embedded fuzzy control of a mobile platform pioneer 3-DX, 8th WSEAS International Conference on Systems Theory and Scientific Computation (ISTASC'08), Rhodes (Rodos) Island, Greece, August 20-22, 2008, pp.334-335, ISBN: 978-960-8764-83-1, ISSN: 1790-3865, Published by WSEAS Press (www.wseas.org, www.worldses.org/indexes)
[14] **Ioan Susnea**, Adrian Filipescu, Adriana Serbencu, A. Radaschin, *Virtual Pheromones to Control Mobile Robots* . A Neural Network Approach, Proceedings of the IEEE International Conference on Automation and Logistics, Shenyang, China August 2009, ISBN: 978-1-4244-4795-4/09, 2009 IEEE, CD-ROM Proceedings IEEE, Catalog #: CFP09CAL ISBN: 978-1-4244-4795-4, August 5 - 7, 2009, Shenyang, China
[15] Adriana Serbencu, Daniela Cristina Cernega, Adrian Emanoil Serbencu, **Ioan Susnea**, *Path Following Problem for PatrolBot Solved with Fuzzy Control* , Proceedings of the IEEE, International Conference on Automation and Logistics, Shenyang, China August 2009, ISBN: 978-1-4244-4795-4/09, 2009 IEEE, Catalog #: CFP09CAL ISBN: 978-1-4244-4795-4, August 5 - 7, 2009, Shenyang, China
[16] **Susnea I**. Filipescu A, Minzu V, Vasiliu G, Virtual Pheromones and Neural Networks based Wheeled Mobile Robot Control, Recent Advances on Systems, Proceedings of thr 13 International Conference on Systems WSEAS CSCC Multiconference, ISBN 978-960-474-097-0, ISSN: 1790-2769 Rodos, Greece , JULY 22-24, 2009, pp 511-516 Mathematics and Computers in Science Engineering, Aseries of Reference Books and Textbook Published by WSEAS Press
[17] **Susnea I**. Filipescu A, Minzu V, Vasiliu G, Virtual Pheromones and Neural Networks based Wheeled Mobile Robot Control, Recent Advances on Systems, Proceedings of thr 13 International Conference on Systems WSEAS CSCC Multiconference, ISBN 978-960-474-097-0, ISSN: 1790-2769 Rodos, Greece , JULY 22-24, 2009, pp 511-516 Mathematics and Computers in Science Engineering, Aseries of Reference Books and Textbook Published by WSEAS Press
[18] **Susnea I**, Filipescu Adrian, Filipescu Adriana, Coman G., Wheeled Mobile Robot Control Using Virtual Pheromones, Petroleum-Gas University of Ploiesti Bulletin, Tehnical Series, Vol LXI, no.3/2009, ISSN 1224-8495, pp. 117-126.
[19] **Susnea I**. Vasiliu G, Filipescu A, Coman G., Radaschin A., Real-Time Control of Autonomous Mobile Robots Using Virtual Pheromones, Proceedings of the 7th Asian

Control Conference, Hong Kong, China, August 27-29, 2009, IEEE Catalog Number CFP09832, ISBN:978-89-956056-9-1, pp.1450-1455.
[20] Filipescu Adrian, **Susnea I**, Filipescu Adriana, Stamatescu G.,Control of Mobile platforms as Robotic Assistants for Elderly, Proceedings of the 7th Asian Control Conference, Hong Kong, China, August 27-29, 2009, IEEE Catalog Number CFP09832, ISBN:978-89-956056-9-1, pp.1456-1461.
[21] Filipescu Adrian, **Susnea I**, Filipescu Silviu, Stamatescu G., Wheeled Mobile Robot Control Using Virtual Pheromones and Neural Networks, Proceedings of 2009 IEEE International Conference on Control and Automation Christchurch, New Zealand, December 9-11, 2009, IEEE Catalog Number CFP09537, ISBN: 978-1-4244-4707-7, pp: 157-162, Library of Congress:2009904841.
[22] Filipescu Adrian, **Susnea I**, Filipescu Adriana, Stamatescu G., Distributed System of Mobile Platform Obstacle Avoidance and Control as Robotic Assistant for Disabled and Elderly, Proceedings of 2009 IEEE International Conference on Control and Automation Christchurch, New Zealand, December 9-11, 2009, IEEE Catalog Number CFP09537, ISBN: 978-1-4244-4707-7, pp: 1886-1891, Library of Congress:2009904841.
[23] **Ioan Susnea**, Viorel Minzu, Grigore Vasiliu Simple, Real-time Obstacle Avoidance Algorithm for Mobile Robots, Proceedings of the 8'th WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics, CIMMACS'09, Tenerife,, Dec. 14-16, 2009, pp:24-30, ISBN: 978-960-474-144-1, ISSN: 1790-5117

VII Alte activitati stiintifice/semne de recunoastere internationala

[24] Review Springer Verlag
[25] Reviewer Bentham Science Publishers Ltd
[26] Reviewer IEEE
[27] Reviewer WSEAS http://www.worldses.org/review/2008reviewers.htm
http://www.worldses.org/review/2009reviewers.htm
[28] Nominalizat pentru includerea in volumul *Who's Who in the world 2010*, publicat de Marquis Whos Who, www.marquiswhoswho.com

Referințe

[1] Gates B., A robot în every home, *Scientific American (2007).*

[2] *http://asimo.honda.com/ Homepage of ASIMO from Honda*

[3] *http://www-robotics.cs.umass.edu/Robots/UBot-5 Homepage of Ubot5 robot at UMass Amherst*

[4] *http://www.bostondynamics.com/ Homepage of Boston Dynamics Inc. creators of "Big Dog" robot*

[5] Duffy B.R. *Anthropomorphism and Robotics* available online*:*

www.csi.ucd.ie/csprism/publications/desire/AISB02-Duffy.pdf

[6] United States General Accounting Office, *Aging Baby Boom Generation Will Increase Demand and Burden on Federal and State Budgets*, *document number GAO-02-544T, released March 2001.*

[7] U.S. Census http://www.census.gov/hhes/www/disability/2006acs.html

[8] www.anph.ro Website of the Romanian National Agency for Handicaped

[9] www.mmuncii.ro Website of the Romanian Ministry of Labor, Family, and Social Protection

[10] Magnus G.*, The Age of Aging: How Demographics are Changing the Global Economy and Our World*, John Wiley & Sons, 2008

[11] *www.mobilerobots.com* Manufacturer of mobile robots for research

[12] Borenstein, J., Koren, Y, *A Mobile Platform For Nursing Robots*. IEEE Transactions on Industrial Electronics, May 1985, pp.158-165.

[13] Tzafestas, S.G. *Research on autonomous robotic wheelchairs în Europe*, IEEE Robotics & Automation Magazine, 2001, vol 8. pp:4-6

[14] Kuno, Y. Shimada, N. Shirai, Y. *Look where you're going,* Robotics & Automation Magazine, IEEE, Vol 10, March 2003, pp: 26-34

[15] Intelligent wheelchair Robotics Laboratory, Nara Institute of Science and Technology http://robotics.aist-nara.ac.jp/~yoshio/research/wchair/index_e.html

[16] DEKA Research and Development Corporation, INDEPENDENCE - IBOT-4000 Mobility System, http://www.ibotnow.com/

[17] Kulyukin V., Kutiyanawala A., LoPresti E., Matthews J., Simpson R., iWalker: *Toward a Rollator-Mounted Wayfinding System for the Elderly*, Proceedings of the 2008 IEEE International Conference on RFID (IEEE RFID). April 16-17, Las Vegas, NV, 2008

[18] Gharpure C., Kulyukin V., *Robot-Assisted Shopping for the Blind*: Issues în Spatial Cognition and Product Selecțion, International Journal of Service Robotics, March 2008

[19] Kwang-Hyun Park, Zeungnam Bien, Ju-Jang Lee, Byung Kook Kim, Jong-Tae Lim, Jin-Oh Kim, Heyoung Lee, Dimitar H. Stefanov, , Dae-Jin Kim, Jin-Woo Jung, Jun-Hyeong Do, Kap-Ho Seo, Chong Hui Kim, Won-Gyu Song and Woo-Jun Lee*, Robotic smart house to assist people with movement disabilities* Journal of Autonomous Robots , Volume 22, Number 2 / February, 2007, pp: 183-198

[20] Tao Lu, Kui Yuan, Haibing Zhu, Huosheng Hu, *An Embedded Control System for Intelligent Wheelchair*, Engineering în Medicine and Biology Society, 2005. IEEE-EMBS 2005. 27th Annual International Conference of the Volume , Issue , 2005 pp:5036 - 5039

[21] Dario P., Laschi C., Guglielmelli E.*, Design and experiments on a personal robotic assistant.* Advanced Robotics 13(2):153–69 (1999)

[22] Dario P. , Guglielmelli E., Laschi C. ,Humanoids and personal robots: Design and experiments*, Journal of Robotic Systems Volume 18 Issue 12, , 2000, pp: 673 – 690*

[23] The MILO project from RoboDynamics Corp. (Mechatronic Intelligent Lovable Organism) http://www.robodynamics.com/press_pr_10212004_MILO.asp

[24] Paulos E., Canny J., Social Tele-Embodiment: Understanding Presence*, Autonomous Robots 11, 87–95, 2001, Kluver Academic Press.*

[25] Sachs N. A., Nulud P.L., Loeb G.E. *, Virtual Visit: improving communication for those who need it most*. Studies în Health Technology of Information, 2003 (Vol. 94, Pages 302-8)

[26] Pineau J., Montemerlo M., Pollack M., Roy N., Thrun S., *Towards robotic assistants în nursing homes: Challenges and results , în* Proc. of the Workshop on Interactive Robotics and Entertainment, 2000

[27] Matthews J., Engberg S., Montemerlo M., Pineau J., Roy N., Rogers J., Thrun S., Handler S., Starrett T., Ting D., and Travis R. *The Nursebot project: Results of preliminary field studies during development of a personal robotic assistant for older adults*.. în Proceedings of the Greater Pittsburgh 14th Annual Nursing Research Conference, Pittsburgh, PA, 2002.

[28] Baltus G., Fox D., Gemperle F., Goetz J., Hirsch T., Magaritis D., Montemerlo M., Pineau J., Roy N., Schulte J., Thrun S. - Towards personal service robots for the elderly *(2000) în Proc. of the Workshop on Interactive Robotics and Entertainment*

[29 Bischoff, R. Graefe, V. HERMES - a versatile personal robotic assistant, Proceedings of the IEEE, Vol. 92, issue 11, pp: 1759- 1779, 2004

[30] Joelle Pineau , Michael Montemerlo, Martha Pollack, Nicholas Roy, Sebastian Thrun*, Towards robotic assistants în nursing homes: Challenges and results*, Robotics and Autonomous Systems, Volume 42, Issues 3-4, 31 March 2003, pages 271-281.

[31] M. Montemerlo, J. Pineau, N. Roy, S. Thrun and V. Varma. ``*Experiences with a Mobile Robotic Guide for the Elderly''*. Proceedings of the International Conference on Artificial Intelligence (AAAI 2002). Edmonton, Jul. 2002

[32] Hans M., Graf B., Schraft R.D., *Robotic Home Assistant Care-O-bot: Past – Present – Future,* Fraunhofer Institute for Manufacturing Engineering and Automation (IPA) în IEEE Int. Workshop on Robot and Human interactive Communication, 2002 (ROMAN2002)

[33] Graf, B., Hägele, M.: *"*Dependable Interaction with an Intelligent Home Care Robot*". în Proceedings of ICRA-Workshop on Technical Challenge for Dependable Robots în Human Environments, 2001, pp. IV-2.*

[34] The HelpMate project http://www.techbriefs.com/component/content/article/2119

[35] Bischoff R. , Graeffe V. , *HERMES: A versatile personal robotic assistant*, Proceedings of the IEEE ISSN 0018-9219, IEEPAD 2004, vol. 92, no 11 pp. 1759-1779

[36] Mazo M., Rodríguez F. J., Lázaro J. L., Ureña J., García J. C., Santiso E. and Revenga P. A., *Electronic control of a wheelchair guided by voice commands,* Control Engineering Practice, Volume 3, Number 5, March 1995 , pp. 665-674

[37] Rudzionis A., Rudzionis V. *Noisy speech detection and endpointing. Voice operated telecom services. Do they have a bright future? Workshop Proceedings*, Ghent, Belgium 11-12 May 2000, pp: 79-82.

[38] **Susnea I .** Vasiliu G, Filipescu A, Coman G., *On the Implementation of a Robotic Assistant for the Elderly. A Novel Approach*, 7th WSEAS Int. Conf. on Computational, Intelligence Man-machine Systems and Cybernatics (CIMMACS '08)**,** Cairo, Egipt, December 29-31, 2008, pp.215-220, ISBN 978-960-474-049-9, ISSN: 1790-5117

[39] Jeonard J.J., Durrant-Whyte H.F. – *Mobile Robot Localization by Tracking Geometric Beacons,* IEEE Tranzactions on Robotics and Automation, vol 7. nr. 3, June 1991

[40] Spong M. W., Hutchinson S., and Vidyasagar M., *Robot Modeling and Control*, John Wiley & Sons, 2007

[41] Craig J., *Introduction to Robotics. Mechanics and Control,* Second Edition, Addison Wesley Longman, 1989

[42] Shabana A. A. *Computational Dynamics, Second Edition,* John Wiley & Sons, 2001

[43] Seiegwarth R. , Nourbakhsh I.R., *Introduction to Autonomous Mobile Robots*, MIT Press, 2004

[44] Noga S., *Kinematics and Dynamics of Some Selected Two-Wheeled Mobile Robots*, Archives of Civil and Mechanical Engineering, Vol. VI, No. 3, 2006

[45] Borenstein J., Everett H. R., and Feng L., *Where Am I. Sensors and Methods for Mobile Robot Positioning*, University of Michigan, 1996.

[46] Jensfelt P. *Approaches to Mobile Robot Localization în Indoor Environments*, PhD Thesis, Royal Institute of Technology, Sweden, 2001, available online

[47] Tonouchi, Y., Tsubouchi, T., and Arimoto, S., 1994, *"Fusion of Dead-reckoning Positions With a Workspace Model for a Mobile Robot by Bayesian Inference."* International Conference on Intelligent Robots and Systems (IROS '94). Munich, Germany, Sept. 12-16, pp: 1347-1354.

[48] Komoriya, K. and Oyama, E., 1994, *Position Estimation of a Mobile Robot Using Optical Fiber Gyroscope (OFG).* International Conference on Intelligent Robots and Systems (IROS '94). Munich, Germany, Sept. 12-16, pp. 143-149.

[49] Borenstein, J. and Feng, L., 1994, "*UMBmark* — A Method for Measuring, Comparing, and Correcting Dead-reckoning Errors în Mobile Robots." *Technical Report, The University of Michigan UM-MEAM*-94-22, Dec.

[50] Borenstein, J., 1994b, I*nternal Correction of Dead-reckoning Errors With the Smart Encoder Trailer.* 1994 International Conference on Intelligent Robots and Systems (IROS '94). Munich, Germany, Sept. 12-16, pp. 127-134.

[51] McKerrow, P. J. *Introduction to Robotics;* Addison-Wesley Publishing Company Inc., 1991.

[52] Everett, H.R. *Sensors for Mobile Robots* A. K. Peters, 1995.

[53] Zhao, Y., *Vehicle Location and Navigation Systems*, Artech House, 1997

[54] Hebert M., *Active and Passive Range Sensing for Robotics*, Proceedings of the IEEE International Conference on Robotics and Automation, 2000, pp: 102-110

[55] Gu J., Meng M., Cook A. , Liu P.: *Sensor Fusion în Mobile Robot: Some Perspectives;* Proceedings of the 4th World Congress on Intelligent Control and Automation, p. 1194-1199, Shanghai, China, June, 2002

[56] El-Rabbani A. *Introduction to GPS, The Global Positioning System*, Artech House, 2002

[57] Skolnik M., *Radar Handboo*", McGraw-Hill Professional 2008

[58] Ladd A.M., Bekris K.E., Rudys A, Marceau G, Kavraki LE, Wallach, DS (2002) *Robotics-based location sensing using wireless Ethernet*. In: Proceedings of the 8th ACMMOBICOM Conference, pp: 227–238, Atlanta, GA, September 2002

[59] Bahl P. and Padmanabhan V.N. *Radar: An in-building user location and tracking system.* Proceedings of the IEEE Infocom, Tel Aviv, Israel, 2 pp:775–784, 2000.

[60] Betke M. and Gurvits K. *Mobile robot localization using landmarks*. în Proceedings of the IEEE International Conference on Robotics and Automation, volume 2, pages 135-142, May 1994.

[61] Esteves J.S. , Carvalho A. *Generalized Geometric Triangulation Algorithm for Mobile Robot Absolute Self-Localization*, Industrial Electronics, 2003. ISIE '03. 2003, Vol. 1 pp: 346-351

[62] Burgard W., Fox D., Thrun S., *Active Mobile Robot Localization* în Proceedings of IJCAI-97, IJCAI, Inc, 1997

[63] Kleinberg J. M., *The Localization Problem for Mobile Robots*, în Proceedings of the 35th IEEE Symposium on Foundations of Computer Science, 1994

[64] Cohen C. and Koss F.V., *A Comprehensive Study of Three Object Triangulation*, Proceedings of the SPIE Conference on Mobile Robots, 1993

[65] www.dinsmoresensors.com Electronic compass sensors

[66] www.nxp.com/acrobat_download/applicationnotes/AN00022_COMPASS.pdf Electronic Compass Sensors

[67] Esteves J.S. *Metodologia de Autolocalizacao Absoluta Em Ambientes Quase Estructuradas*, PhD Thesis, Universidade do Minho, 2005.

[68] Murray J. C., Erwin H. and Wermter S., *Robotic sound-source localisation architecture using cross-correlation and recurrent neural networks*, Neural Networks, Volume 22, Issue 2, pp: 173-189, March 2009

[69] Valian J.M., et al. *Robust Sound Source Localization Using a Microphone Array on a Mobile Robot,* în Proceedings of the 2003 IEEE/RSJ International Conference on Intelligent Robotics and Systems, pages 1228 – 1233.

[70] Hightower J., Borriello G., and Want R. *Spoton: An indoor 3D location sensing technology based on RF signal strength*. UW CSE Technical Report 2000-02-02, University of Washington, Department of computer Science and Engineering, pages 1–15, 2000

[71] Finkenzeller Klaus, *RFID Handbook, Second Edition*, John Willey and Sons, 2003

[72] Rankl W. and Effing W., *Smart-Card Handbook*, John Willey and Sons, 2003

[73] Hahnel, D., Philipose M. et. al., *Mapping and Localization with RFID Technology*, în Robotics and Automation, 2004. Proceedings. ICRA '04. 2004 IEEE International Conference on, 2004 , Vol. 1, pp: 1015- 1020

[74] Kulyukin V., Gharpure C., Nicholson J., Pavithran S., *RFID în Robot-Assisted Indoor Navigation for the Visually Impaired*, Intelligent Robots and Systems, 2004. (IROS 2004). Proceedings. 2004 IEEE/RSJ International Conference on Volume 2, Issue , 28 Sept.-2 Oct. 2004 pp: 1979 -1984 vol.2

[75] Schneegans S., Vorst P., Zell A., *Using RFID Snapshots for Mobile Robot Self-Localization*, Proceedings of the 3rd European Conference on Mobile Robots (ECMR 2007) Freiburg, Germany: ,2007, pp: 241-246.

[76] Kulyukin V., Kutiyanawala A., Jiang M., *Surface-embedded Passive RF Exteroception: Kepler, Greed, and Buffon's Needle*, in Ubiquitous Intelligence and Compuțing, Springer, 2007, pp: 33-42

[77] Park S., Hashimoto S., *Indoor localization for autonomous mobile robot based on passive RFID*, Proceedings of the 2008 IEEE International Conference on Robotics and Biomimetics Bangkok, Thailand, February 21 - 26, 2009

[78] Koutsou A. D., Seco F., Jimenez A. R., Roa J. O., Ealo J. L., Prieto C., Guevara J., *Preliminary Localization Results With An RFID Based Indoor Guiding System*, în IEEE International Symposium on Intelligent Signal Processing Alcala de Henares, Spain, October, 2007.

[79] Zhang Y. , Amin M.G., Kaushik S., *Localization and Tracking of Passive RFID Tags Based on Direction Estimation*, International Journal of Antennas and Propagation , 2007

[80] Senta Y., Kimuro Y., Takarabe S., Hasegawa T., *Self-Localization for Mobile Robot Using RFID Tags without Layout Information*, IEEJ Transactions on Electronics, Information and Systems, Vol. 128, No. 7, 2008

[81] Byoung-Suk Choi, Joon-Woo Lee and Ju-Jang Lee *Localization and Map-building of Mobile Robot Based on RFID Sensor Fusion System*, în IEEE International Conference on Industrial Informatics, Daejeon Korea, INDIN 2008

[82] Lim Hyung Soo, Choi Byoung Suk, Lee Jang Myung, *An Efficient Localization Algorithm for Mobile Robots based on RFID System*, SICE-ICASE International Joint Conference 2006 Bexco, Busan, Korea

[83] Hae Don Chon, Sibum Jun, Heejae Jung, Sang Won An, *Using RFID for Accurate Positioning*, The 2004 International Symposium on GNSS/GPS Sydney, Australia 6–8 December 2004

[84] *www.loke.de* Manufacturer of high precision laser rangers

[85] Soo-Yeong Yi, *Global ultrasonic system with selective activation for autonomous navigation of an indoor mobile robot*, Robotica vol. 26, pp: 277-283, Cambridge University Press, 2007.

[86] Schenker, L, *Pushbutton Calling with a Two-Group Voice-Frequency Code*, The Bell system technical journal 39 (1): 235–255, 1960

[87] **Susnea I**, Mitescu M. *Microcontrollers în practice*, Springer, 2005

[88] Sen M. Kuo, Bob H. Lee, Wenshun Tian, *Real-Time Digital Signal Processing : Implementations and Applications*, (2nd edition), John Wiley & Sons, 2006

[89] Ming X. – *Fundamentals of Robotics – Linking Perception to Action*, World Scientific Publishing, Series în machine Perception and Artificial Intelligence, vol 54, 2003

[90] Paul, R.P. *WAVE: A Model-Based Language for Manipulator Control*, The Industrial Robot, Vol. 4, No. 1, 1977, pp. 10–17.

[91] ABB Flexible Automation. RAPID Reference Manual. S-72168 Västerås, Sweden. Art. No. 3HAC 7783-1. http://rab.ict.pwr.wroc.pl/irb1400/datasys_rev1.pdf

[92] Mitsubishi MELFA Robots. http://www.mitsubishi-automation.com/products/robots_content.html

[93] Freund, E., Lüdemann-Ravit, B, Stern, O. and Koch, T. [2001]. *Creating the Architecture of a Translator Framework for Robot Programming Languages*. Proceedings of the 2001 IEEE International Conference on Robotics & Automation, Seoul, Korea, May 21–26, 2001. pp: 187–192.

[94] Pembeci, I., and Hager, G. *A Comparative Review of Robot Programming Languages*, CIRL technical report, 2001. Available from http://citeseer.nj.nec.com/555282.html

[95] Miller, D.J. and Lennox, R.C. *An Object-Oriented Environment for Robot System Architectures*. Proceedings of the IEEE International Conference on Robotics and Automation, Cincinnati, OH, August 13–16, 1990. Pp. 14–23.

[96] Fraser, R.J.C. and Harris, C.J. *Infrastructure for Real-time Robotic Control: Aspects of Robot Command Language*. IEEE Colloquim on Intelligent Control, 19 February 1991.

[97] Voudouris, C., Chernett, P., Wang, C.J. and Callaghan, V.L. *Hierarchical behavioural control for autonomous vehicles,* în Proceedings of the 2nd IFAC Conference on Intelligent Autonomous Vehicles 95, Espoo, Finland, 1995. pp: 267–272.

[98] Levesque, H.J., Reiter, R., Lesperance, Y. and Scherl, R.B. *GOLOG: A Logic Programming Language for Dynamic Domains,* in Journal of Logic Programming, 31(1–3), 1997, pp: 59–83.

[99] Grosskreutz, H. and Lakemeyer, G. *Towards more realistic logic-based robot controllers în the GOLOG framework*. Korrekturabzug, Künstliche Intelligenz, Heft 4/00, arenDTaP Verlag, Bremen, 2000. pp: 11–15.

[100] Simmons, R. and Apfelbaum, D. *A Task Description Language for Robot Control*. Proceedings of the Conference on Intelligent Robotics and Systems IROS98, Vancouver, October 1998.

[101] North, S. and Hermans, P. *XML Trainer Kit*. IT Press, 2000.

[102] Makatchev, M. and Tso, S.K. *Human-Robot Interface Using Agents Communicating în a XML-Based Markup Language*. Proceedings of the IEEE International Workshop on Robot and Human Interactive Communication ROMAN 2000, Osaka, Japan, September 27–29, 2000. pp: 270–275.

[103] Leifer, L., Van der Loos, M. and Lees, D. *Visual Language Programming for robot command control în unstructured environments*. Proceedings of the Fifth International Conference on Advanced Robotics ICAR-91, Vol 1., June 1991. pp: 31–36.

[104] Fleureau, J.L., Le Rest, E. and Marce, L. *PILOT: A Language for Planning Mission*. Proceedings of the 2nd IFAC Conference on Intelligent Autonomous Vehicles, Espoo, Finland, June 12–14, 1995. pp: 386–391.

[105] Drews, P. and Fromm, P. *A Natural Language Processing Approach for Mobile Service Robot Control*. Proceedings of the 23rd International Conference on Industrial Electronics, Control and Instrumentation (IECON 97), Vol 3, November 1997. pp: 1275–1277.

[106] Speech recognition software http://www.nuance.com/naturallyspeaking/

[107] Lauria, S., Bugmann, G., Kyriacou, T., Bos, J., and Klein E. *Training Personal Robots Using Natural Language Instruction*. Intelligent Systems, IEEE, Volume 16, Issue 5, Sep/Oct 2001. pp: 38–45.

[108] Lauria, S., Bugmann, G., Kyriacou, T., Bos, J. and Klein, E. *Converting Natural Language Route Instructions into Robot Executable Procedures.* Proceedings of the 2002 IEEE International Workshop on Robot and Human Interactive Communication, Berlin, September 25–27, 2002. pp: 223–228.

[109] www.plcopen.org IEC-61131 specifications

[110] Laumond J.P. (Ed.), *Robot Motion Planning and Control*, Springer, 1998

[111] Latombe, J.C. *Robot motion planning*. Kluwer Academic Publishers, 1991.

[112] Leedy B. M., Putney J. S., Bauman C., Cacciola S., Webster J. M. and Reinholtz C. F., *Virginia Tech's twin contenders: a comparative study of reactive and deliberative navigation,* Journal of Field Robotics, vol. 23, no. 9, pp. 709–727, 2006.

[113] Morales J., Martinez J.L., Martinez M.A., and Mandow A. *Pure-Pursuit Reactive Path Tracking for Nonholonomic Mobile Robots with a 2D Laser Scanner*, EURASIP Journal on Advances în Signal Processing Volume 2009, Article ID 935237

[114] Hellström T., Ringdahl O., *Follow the Past: a path-tracking algorithm for autonomous vehicles*, Int. J. Vehicle Autonomous Systems, Vol. 4, Nos. 2-4, 2006

[115] Barton, M.J. *Controller Development and Implementation for Path Planning and Following in an Autonomous Urban Vehicle*. Undergraduate thesis, University of Sydney, 2001

[116] Amidi O., *Integrated mobile robot control*, Tech. Rep. CMURI-TR-90-17, Robotics Institute, Carnegie Mellon University, Pittsburgh, Pa, USA, May 1990.

[117] Wit J.S. *Vector Pursuit Path Tracking for Autonomous Ground Vehicles*, Ph.D. thesis, University of Florida, 2000, available online: www.dtic.mil

[118] Thrun S. et al. *Stanley: The Robot that Won the DARPA Grand Challenge*, Journal of Field Robotics 23(9), 661–692 (2006)

[119] Scharf L. L., Harthill W. P., and Moose P. H., *A comparison of expected flight times for intercept and pure-pursuit missiles*, IEEE Transactions on Aerospace and Electronic Systems, vol. 5, no. 4, pp: 672–673, 1969.

[120] Foley J. D., van Dam A., Feiner S.K., Hughes J.F., *Computer Graphics: Principles and Practice în C* (2nd Edition), Addison-Wesley Professional, 1995

[121] Heredia G., Ollero A., *Stability of Autonomous Vehicle Path Tracking with Pure Delays în the Control Loop*, Advanced Robotics, Vol. 21, No. 1-2, pp. 23-50 (2007)

[122] Urmson C., Ragusa C., Ray D., et al., *A robust approach to high-speed navigation for unrehearsed desert terrain*, Journal of Field Robotics, vol. 23, no. 8, pp: 467–508, 2006

[123] Ollero A., Garcia-Cerezo A. J., and Martinez J. L., *Fuzzy supervisory path tracking of mobile robots,* Control Engineering Practice, vol. 2, no. 2, pp: 313–319, 1994.

[124] Sugeno M., Nishida M., *Fuzzy control of model car*, Fuzzy sets and systems, Vol 16, pp:103-113, 1986

[125] Saffiotti A. *The uses of fuzy logic in autonomous robot navigation*, Soft Compuţing 1(4) pp:180-197, 1997

[126] Reznik L., *Fuzzy Controllers*, Newnes, 1997

[127] Ahmad I. *Fuzzy Logic for Embedded System Applications*, Newnes, 2003, ISBN 0750676051

[128] Lakehal, B.; Amirat, Y.; Pontnau, J. *Fuzzy steering control of a mobile robot*, în IEEE/IAS Conference on Industrial Automation and Control: Emerging Technologies, 1995., pp:383-386

[129] **Susnea I**, Filipescu A, Vasiliu G., Filipescu S., *Path Following, Real-time, Embedded Fuzzy Control of a Mobile Platform Wheeled Mobile Robot*, IEEE International Conference on Automation and Logistics, ICAL 2008, 1-3 Sep., Qingdao, China, IEEE ICAL 2008

[130] Kovacic Z., Bogdan S., *Fuzzy Controller Design Theory and Applications*, Taylor and Francis Group, 2006

[131] Kandel,Y.L. and Zhang,Y.-Q., *Stability analysis of fuzzy control systems*, Fuzzy Sets and Systems, 105, 33–48, 1999.

[132] Bandemer, H. and Hartmann, S., *A fuzzy approach to stability of fuzzy controllers*, Fuzzy Sets and Systems, 96, 161–172, 1998.

[133] Karlson, P., Lüscher, M. *Pheromones: a new term for a class of biologically active substances.* Nature 183, 1959, pp. 55-56

[134] Grassé P.P. *La Reconstruction du nid et les Coordinations Inter-Individuelles chez Bellicositermes Natalensis et Cubitermes sp. La theorie de la Stigmergie: Essai d'interpretation du Comportement des Termites Constructeurs*. Insectes Sociaux, 6:4181, 1959

[135] Deneubourg, J., Aron, S., Goss, S., Pasteels, J. M., and Duerinck, G. *Random behaviour, amplification processes and number of participants : How they contribute to the foraging properties of ants.* Physica 22(D) 1986., pp: 176–186.

[136] Deneubourg, J.-L., Aron, S., Goss, S. and Pasteels, J. M., *The self-organizing exploratory pattern of the Argentine ant*, J. Insect Behav. 32, 1990 pp:159- 168.

[137] R. Beckers, O.E. Holland, and J.-L. Deneubourg. *From local actions to global tasks: Stigmergy and collective robotics* in Articial Life IV. Proc. Fourth International Workshop on the Synthesis and Simulation of Living Systems, pp:181-189, Cambridge, Massachusetts, USA, 1994.

[138] Beni, G., Wang, J. *Swarm Intelligence in Cellular Robotic Systems*, Proceedings of NATO Advanced Workshop on Robots and Biological Systems, Tuscany, Italy, 1989, pp:26–30

[139] Dorigo M et al. *The ant system: optimization by a colony of cooperating agents.* IEEE Trans Syst Man Cybernet 26(1), 1996, pp:29–41

[140] Dorigo M., Stützle T. *Ant Colony Optimization*, MIT Press, 2004

[141] Bonabeau E., Dorigo M., and Theraulaz G., *Swarm Intelligence: From Natural to Artificial Systems*. New York, Oxford University Press, 1999.

[142] Arkin, R.C. and Bekey, G.A. (editors). *Robot Colonies*. Kluwer Academic Publishers, 1997.

[143] Pearce T. C., Schiffman S. S., Nagle H. T., and Gardner J. W., Eds., *Handbook of Machine Olfaction*. Weinheim, Germany: Wiley, 2003.

[144] Purnamadjaja A.H, Russell R. A, *Guiding robots' behaviors using pheromone communication* Journal of Autonomous Robots, Vol. 23 (2), Kluwer Academic Publishers Hingham, MA, USA, 2007, pp: 113 - 130

[145] Genovese, V., Dario, P.,Magni, R., & Odetti, L. *Self organizing behavior and swarm intelligence in a pack of mobile miniature robots în search of pollutants*. în Proceedings of the IEEE/RSJ international conference on intelligent robots and systems, Raleigh, NC, 1992, pp. 1575– 1582

[146] Hayes A.T., Martinoli A., Goodman R.M., *Swarm Robotic Odor Localization*, Robotica, Vol. 21, nr. 4, August 2003, pp: 427 – 441

[147] Payton D., Daily M., Hoff B., Howard M., Lee C., *Pheromone Robotics*, Autonomous Robots Volume 11 , Issue 3 (November 2001) pp: 319 - 324 , 2001

[148] Payton D., Estkowski R., Howard M., *Pheromone Robotics and the Logic of Virtual Pheromones*, în Swarm Robotics, Springer, 2005, pp: 45-57.

[149] Szumel L., Owens J. D., *The Virtual Pheromone Communication Primitive*, în Distributed Computing în Sensor Systems, Vol 4026/2006, pp: 135-149

[150] Kernbach S., Thenius R., Kernbach O., Schmickl T., *Re-embodiment of Honeybee Aggregation Behavior in an Artificial Micro-Robotic System* in Adaptive Behavior, Vol 17(3): 237–259, 2009.

[151] Parunak H. v. D, Brueckner S.A. Sauter J. *Digital Pheromones for Coordination of Unmanned Vehicles*, în Lecture Notes în Computer Science, Vol. 3374, Springer, 2005

[152] Mamei M., Zambonelli F., *Physical deployment of digital pheromones through RFID technology,* Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE

[153] Mamei M., Zambonelli F., *Pervasive Pheromone-Based Interaction with RFID Tags*, in ACM Transactions on Autonomous and Adaptive Systems (TAAS) Volume 2 , Issue 2, June 2007

[154] **Susnea I** . Vasiliu G, Filipescu A*, RFID Digital Pheromones for Generating Stigmergic Behaviour to Autonomous Mobile Robots,* 4th WSEAS Int. Conf. on Dynamic Systems and Control (Control '08), Corfu, Greece, October 26-28, 2008, pp.20-24.

[155] Khatib O., *Real-Time Obstacle Avoidance for Manipulators and Mobile Robots*, The International Journal of Robotics Research, Vol. 5, No. 1, pp:90-98, 1986

[156] **Susnea I.** Vasiliu G. Filipescu A. Radaschin A., *Virtual Pheromones for Real-Time Control of Autonomous Mobile Robots*, in Studies of Informatics and Control, Vol 18, issue 3, 2009, pp: 233-240., ISSN:1220-1760

[157] Lumelsky, V., Skewis, T., *Incorporating Range Sensing în the Robot Navigation Funcție,* IEEE Transactions on Systems, Man, and Cybernetics, 20:1990, pp. 1058–1068..

[158] Lumelsky, V., Stepanov, A., *Path-Planning Strategies for a Point Mobile Automaton Moving Amidst Unknown Obstacles of Arbitrary Shape,* in Autonomous Robot Vehicles. New York, Spinger-Verlag, 1990

[159] Kamon, I., Rivlin, E., Rimon, E., *A New Range-Sensor Based Globally Convergent Navigation Algorithm for Mobile Robots*, în Proceedings of the IEEE International Conference on Robotics and Automation, Minneapolis, April 1996.

[160] Khatib, O., 1985, *Real-Time Obstacle Avoidance for Manipulators and Mobile Robots.* 1985 IEEE International Conference on Robotics and Automation, March 25-28, St. Louis, pp. 500-505.

[161] Koren, Y., Borenstein, J., *High-Speed Obstacle Avoidance for Mobile Robotics,* in Proceedings of the IEEE Symposium on Intelligent Control, Arlington, VA, August 1988, pp. 382-384.

[162] Borenstein, J., Koren, Y., *The Vector Field Histogram – Fast Obstacle Avoidance for Mobile Robots*. IEEE Journal of Robotics and Automation, 7, 278–288, 1991.

[163] Ulrich, I., Borenstein, J., *VFH+: Reliable Obstacle Avoidance for Fast Mobile Robots*, în Proceedings of the International Conference on Robotics and Automation (ICRA'98), Leuven, Belgium, May 1998.

[164] Ulrich, I., Borenstein, J., *VFH*: Local Obstacle Avoidance with Look-Ahead Verification*, în Proceedings of the IEEE International Conference on Robotics and Automation, San Francisco, May 24–28, 2000.

[165] Khatib, O., Quinlan, S., *Elastic Bands: Connecting, Path Planning and Control*, în Proceedings of IEEE International Conference on Robotics and Automation, Atlanta, GA, May 1993

[166] Kim J.H., Park J.B., Yang H. *Implementation of the Avoidance Algorithm for Autonomous Mobile Robots Using Fuzzy Rules* în Fuzzy Systems and Knowledge Discovery, Springer 2006.

[167] Tzafestas S.G. and Zavlangas P. *Industrial and Mobile Robot Collision–Free Motion Planning Using Fuzzy Logic Algorithms*, Industrial-Robotics-Theory-Modelling-Control, ARS/plV, Germany, 2006, pp. 964, 995

[168] *www.eltima.com* Manufacturer of serial to Ethernet connector software

ANEXA A

DETALII PRIVIND IMPLEMENTAREA LIMBAJULUI DE PROGRAMARE RDPL

Implementarea RDPL presupune două aplicații software distincte si anume:

- Compilatorul de macro-instrucțiuni, corespunzător setului de funcții descrise in capitolul 4. Acesta rulează pe un computer personal, primeste ca input un fisier text cu extensia .src, si genereaza un fisier binar, continand codul executabil, care se transferă într-o zonă de memorie nevolatilă a microcontrollerului (EEPROM), precum si un fişier text cu extensia .lst, util pentru depanarea programelor.
- Aplicația pentru decodificarea macro-instrucțiunilor, rulează la nivelul microcontrollerului de decizie, identifică operanzii invocați de fiecare funcție şi execută operațiile specificate.

Ambele aplicații sunt scrise în limbajul ANSI C, pentru maximă portabilitate.

A1. Compilatorul de macro-instrucțiuni

```
FILE *fp1, *fp2, *fp3;

/* fp1 - fisierul sursa, fp2 fisierul de listare, fp3 fisierul binar */

int breakline(void);      /* sparge linia pe campuri */
int descline(void);       /* actualizeaza descriptorul de linie line_desc */
int parse1(void);          /* face analiza campurilor in pass1 */
void getcomment(void);    /* extrage comentariul in filed8 */
void find_field(void);    /* incrementeaza ibptr pana la primul campul
urmator */
void getlabel(void);      /* preia eticheta in field1 */
void getmnemo(void);      /* reia mnemonicul in field2 */

void getop1(void);        /* preia primul operand in field3 */
void getop2(void);        /* preia al doilea operand in field4 */
void getop3(void);        /* preia al treilea operand in field5 */
void getop4(void);        /* preia al patrulea operand in field6 */
void getop5(void);        /* preia al cincilea operand in field7 */

int readline(FILE *fp);   /* citeste linia in ibuf */
int is_space(char ch);
int isdelim(char ch);
int iscomment(char ch);
int isterm(char ch);
```

```
int find_sym(void);
void store_label(void);

int find_mnemo(void);

/* destinatia e unul din operanzi */

int find_op1(void);
int find_op2(void);
int find_op3(void);
int find_op4(void);
int find_op5(void);


int find_imm1(void); /* cauta in tabela de simboluri operandul 1 imediat */
int find_imm2(void); /* cauta in tabela de simboluri operandul 2 imediat */
int find_imm3(void); /* cauta in tabela de simboluri operandul 3 imediat */
int find_imm4(void); /* cauta in tabela de simboluri operandul 4 imediat */
int find_imm5(void); /* cauta in tabela de simboluri operandul 5 imediat */

int find_lval(void); /* cauta valoarea etichetei */

void handle_equ(void);
void handle_label(void);
void handle_mnemo(void);
void handle_operands(void);

void handle_op1(void);
void handle_op2(void);
void handle_op3(void);
void handle_op4(void);
void handle_op5(void);


void store_equ_name(void);
void store_equ_value(void);

int gethex1(void);
int getbin1(void);
int getint1(void);
int getasc1(void);

int gethex2(void);
int getbin2(void);
int getint2(void);
int getasc2(void);

int gethex3(void);
int getbin3(void);
int getint3(void);
int getasc3(void);

int gethex4(void);
int getbin4(void);
int getint4(void);
int getasc4(void);

int gethex5(void);
int getbin5(void);
int getint5(void);
```

```
int getasc5(void);

void list1(void);

void ladr(void);  /* pune adresa bptr in bufferul lbuf */
void lcode(void);
void lf1(void);
void lf2(void);
void lf3(void);
void lf4(void);
void lf5(void);
void lf6(void);
void lf7(void);
void lf8(void);

void wrcode1(void);

void chex2(unsigned char ch);
void chex4(unsigned int pack);

unsigned int pack16(unsigned char c1, unsigned char c2, unsigned char c3);
void unpack(unsigned int pack);
void clear_bintab(void);
void clear_lbuf(void);
void clean_lbuf(void);
void gencode1(void);
void gencode2(void);

void main(int argc, char *argv[])
{

   int i;

   char source_fname[128];
   char list_fname[128];
   char bin_fname[128];

   char ch;
   if((argc < 2)||(argc > 2))
   {
     printf("Syntax: XPLC source_file\n");
     printf("Extension is assumed to be .src\n");
     exit(1);
   }
   if((!strcmp(argv[1],"?")) || (!strcmp(argv[1],"/?")))
   {
     printf("Syntax: XPLC source_file\n");
     printf("Extension is assumed to be .src\n");
     exit(1);
   }

   for(i=0;i<128;i++)
   {
   source_fname[i]=*(argv[1]);
   list_fname[i]=*(argv[1]);
   bin_fname[i]=*(argv[1])++;
   }
   strcat(source_fname,".src");
   strcat(list_fname,".lst");
   strcat(bin_fname,".bin");
   command[0]='D';
```

```
   command[1]='E';
   command[2]='L';
   command[3]=' ';

   strcat(command, bin_fname);
   strcat(command, ">NUL");
   printf("%s\n", source_fname);

   if ((fp1=fopen(source_fname,"r"))==NULL) {
      printf("Cannot open source file.\n");
      exit(1);
   }
   if ((fp2=fopen(list_fname,"w"))==NULL) {
      printf("Cannot open list file.\n");
      exit(1);
   }

   if ((fp3=fopen(bin_fname,"wb"))==NULL) {
      printf("Cannot open binary file.\n");
      exit(1);
   }

   clear_bintab();

     while(pass_no==1)
     {
     if(pass_no!=1) break;
     if(line_no==0)
       {
       fprintf(fp2,"\n%s\n\n\n", "     AKxxx compiler list file.");
       }

     parse1();
     gencode1();
     wrcode1();
     list1();
     if(error_code) errorcount++;
     }

end:
   printf("Lines compiled: %d. Total errors %d\n",line_no, errorcount);
   fprintf(fp2, "Lines compiled: %d. Total errors %d\n",line_no,
errorcount);

   if(errorcount==0)
     {
       BINTAB[bptr]=0x0f;
       fwrite(BINTAB, sizeof BINTAB, 1, fp3);
     }
   fcloseall();
   if(errorcount!=0) system(command);
   exit(0);
}


int breakline(void)
{
   char ch;
   int i;
   ibptr=0;
```

```
    field1[0]='\0';
    field2[0]='\0';
    field3[0]='\0';
    field4[0]='\0';
    field5[0]='\0';
    field6[0]='\0';
    field7[0]='\0';
    field8[0]='\0';

    error_code=0;
    qeol=0;
    qcomment=0;
    qempty=0;

    if(isterm(ibuf[0])) qempty=1;
    find_field();
     if(qcomment) {getcomment(); return(1);}
     if(!isdelim(ibuf[0])&&!qempty) getlabel();
     if(qeol) return(0);
    find_field();
     if(qcomment) {getcomment(); return(1);}
     if(qeol) return(0);
    getmnemo();
    find_field();
     if(qcomment) {getcomment(); return(1);}
     if(qeol) return(0);
    getop1();
    find_field();
     if(qcomment) {getcomment(); return(1);}
     if(qeol) return(0);
    getop2();
    find_field();
     if(qeol) return(0);
     if(qcomment) {getcomment();return(1);}
    getop3();
    find_field();
     if(qcomment) {getcomment(); return(1);}
     if(qeol) return(0);

    getop4();
    find_field();
     if(qcomment) {getcomment(); return(1);}
     if(qeol) return(0);

     getop5();
    find_field();
     if(qcomment) {getcomment(); return(1);}
     if(qeol) return(0);

}
void getcomment(void)
{
     int i,j;
     char ch;
     j=0;
     for(i=ibptr;i<ibptr+SIZE8;i++)
        {
        ch=ibuf[i];
        if(isterm(ch)) { qeol=1; break;}
        field8[j]=ch;
        j++;
```

```
        }
     field8[j]='\0';
     ibptr=i;
}


void getlabel(void)
{
      int i,j;
      char ch;
      j=0;
      for(i=ibptr;i<ibptr+SIZE1;i++)
        {
        ch=ibuf[i];
        if(isterm(ch)) {qeol=1; error_code=1; return;} /* ??? */
        if(is_space(ch)) break;
        if(iscomment(ch)) {qcomment=1; error_code=1; break;}
        field1[j]=ch;
        j++;
        }
      field1[j]='\0';
      ibptr=i;

}

void getmnemo(void)
{
   int i,j;
   char ch;
   j=0;
   for(i=ibptr;i<ibptr+SIZE2;i++)
      {
      ch=ibuf[i];
      if(isdelim(ch)) break;
      if(isterm(ch)) {qeol=1; break;}
      if(iscomment(ch)) {qcomment=1; error_code=1; break;}
      field2[j]=ch;
      j++;
      }
   field2[j]='\0';
   ibptr=i;

}

void getop1(void)
{
   int i,j;
   char ch;
   j=0;
   for(i=ibptr;i<ibptr+SIZE3;i++)
      {
      ch=ibuf[i];
      if(isdelim(ch)) break;
      if(isterm(ch)) {qeol=1; break;}
      if(iscomment(ch)) {qcomment=1; error_code=1; break;}
      field3[j]=ch;
      j++;
      }
   field3[j]='\0';
   ibptr=i;
```

```
}

void getop2(void)
{
   int i,j;
   char ch;
   j=0;
   for(i=ibptr;i<ibptr+SIZE4;i++)
      {
      ch=ibuf[i];
      if(isdelim(ch)) break;
      if(isterm(ch)) {qeol=4; break;}
      if(iscomment(ch)) {qcomment=1; error_code=1; break;}
      field4[j]=ch;
      j++;
      }
   field4[j]='\0';
   ibptr=i;

}

void getop3(void)
{
   int i,j;
   char ch;
   j=0;
   for(i=ibptr;i<ibptr+SIZE5;i++)
      {
      ch=ibuf[i];
      if(isdelim(ch)) break;
      if(isterm(ch)) {qeol=4; break;}
      if(iscomment(ch)) {qcomment=1; error_code=1; error_code=1; break;}
      field5[j]=ch;
      j++;
      }
   field5[j]='\0';
   ibptr=i;

}

void getop4(void)
{
   int i,j;
   char ch;
   j=0;
   for(i=ibptr;i<ibptr+SIZE6;i++)
      {
      ch=ibuf[i];
      if(isdelim(ch)) break;
      if(isterm(ch)) {qeol=4; break;}
      if(iscomment(ch)) {qcomment=1; error_code=1; error_code=1; break;}
      field6[j]=ch;
      j++;
      }
   field6[j]='\0';
   ibptr=i;

}

void getop5(void)
{
```

```
   int i,j;
   char ch;
   j=0;
   for(i=ibptr;i<ibptr+SIZE7;i++)
      {
      ch=ibuf[i];
      if(isdelim(ch)) break;
      if(isterm(ch)) {qeol=4; break;}
      if(iscomment(ch)) {qcomment=1; error_code=1; error_code=1; break;}
      field7[j]=ch;
      j++;
      }
   field7[j]='\0';
   ibptr=i;

}


void find_field(void)
{

   /* cauta primul caracter care nu e delimiter*/

   int i;
   char ch;
   qeol=0;
   qcomment=0;

      for(i=ibptr;i<LINESIZE;i++)
       {
       ch=ibuf[i];
       if(isterm(ch)) {qeol=1; break;}
       if(iscomment(ch)) {qcomment=1; break;}
       if(!isdelim(ch)) break;
       }
      ibptr=i;

}

int readline(FILE *fp)
{
   int i;
   char ch;
   int flag=0;
   int flag2=0;

   for(i=0;i<LINESIZE;i++)
     {
      ch=fgetc(fp1);
      if(feof(fp1)&&(i!=0)) {flag=1; break;}
      if(feof(fp1)&&(i==0)) {flag=2; break;}
      if(isterm(ch)) break;
      if((i>1)&&(ibuf[i-1]=='"')) flag2=1;
      if(flag2==0) ibuf[i]=toupper(ch);
      else ibuf[i]=ch;
     }
   ibuf[i]='\0';
   return(flag);
}

/* cauta in tabelul de mnemonice continutul curent al field2
```

```
Daca il gaseste intoarce 1 si seteaza mtptr la pozitia din tabel
unde a gasit mnemonicul
Daca a gasit menemonicul in tabel, actualizeaza si variabilele
current_opcode, current_nrb, current_nrop
*/

int find_mnemo(void)
{
  int i;
  int flag=0;
  char ch;
  mtptr=0;
  for(i=0;i<MSIZE;i++)
    {
    if(!strcmp(field2,mt[i].mnemo)) {flag=1; break;}
    else flag=0;
    }
    mtptr=i;            /* seteaza pointerul in tabel la locul faptei */
    current_opcode=mt[mtptr].opcode;
    current_nrb=mt[mtptr].nrb;
    current_nrop=mt[mtptr].nrop;
    return(flag);
}

int find_op1(void)
{
  int i;
  int flag=0;
  char ch;
  current_op1=0;
  for(i=0;i<RESSIZE;i++)
    {
    if(!strcmp(field3,res[i].resname)) {flag=1; break;}
    else flag=0;
    }
    current_op1=res[i].resadr;
    return(flag);

}

int find_op2(void)
{
  int i;
  int flag=0;
  char ch;
  current_op2=0;
  for(i=0;i<RESSIZE;i++)
    {
    if(!strcmp(field4,res[i].resname)) {flag=1; break;}
    else flag=0;
    }
    current_op2=res[i].resadr;
    return(flag);

}

int find_op3(void)
{
  int i;
  int flag=0;
  char ch;
```

```
  current_op3=0;
  for(i=0;i<RESSIZE;i++)
    {
    if(!strcmp(field5,res[i].resname)) {flag=1; break;}
    else flag=0;
    }
    current_op3=res[i].resadr;
    return(flag);

}

int find_op4(void)
{
  int i;
  int flag=0;
  char ch;
  current_op4=0;
  for(i=0;i<RESSIZE;i++)
    {
    if(!strcmp(field6,res[i].resname)) {flag=1; break;}
    else flag=0;
    }
    current_op4=res[i].resadr;
    return(flag);

}

int find_op5(void)
{
  int i;
  int flag=0;
  char ch;
  current_op5=0;
  for(i=0;i<RESSIZE;i++)
    {
    if(!strcmp(field7,res[i].resname)) {flag=1; break;}
    else flag=0;
    }
    current_op5=res[i].resadr;
    return(flag);

}

int parse1(void)
{
     int code;
     qcomment=0;
     current_nrb=0;
     current_nrop=0;
     current_opcode=0;
     error_code=0;

     switch(readline(fp1))
     {
     case 0:            /* linie normala */
     breakline();
     code=descline();   /* descrie linia actualizand line_desc */
      switch(code)
      {
         case 0: /* empty line */
         qempty=1;
```

```
            break;

            case 1: /* comment line */
            break;

            case 2: /* EQU line */
            handle_equ();
            break;

            case 3: /* linie normala cu mnemonic valid */
            handle_operands();
            break;

            case 4: /* invalid mnemonic */
            error_code=3;
            break;
          }

        line_no++;

        break;

        case 1:             /* EOF detectat, dar linia de analizat */
        breakline();
        line_no++;
        break;

        case 2:             /* EOF si linie goala. End pass1. */
        qempty=1;
        pass_no=2;
        rewind(fp1);
/*        bptr=0;  */
        break;
        }

}


/*
handle_equ
- verifica faca exista simultan field1, field2 si field3.
- daca da, atunci verifica daca field2 este "EQU""
- daca da, atunci pune numele simbolului in tabel
- calculeaza valoarea simbolului si o pune in tabel
*/

void handle_equ(void)
{
   if((field1[0]=='\0')||(field3[0]=='\0')) {error_code=1; return;}
   store_equ_name();
   store_equ_value();
   stptr++;
}

void store_equ_name(void)
{
   int i=0;
   char ch;
   for(i=0;i<SIZE1;i++)
     {
     ch=field1[i];
```

```
      st[stptr].sname[i]=ch;
      if(ch=='\0') break;
      }

}

void store_equ_value(void)

{
   int i;
   char ch;
   unsigned char ch1;  /* cei doi octeti ai simbolului */
   unsigned char ch2;
   unsigned int sval;

   if(!isdigit(field3[0]))
     {
     if(field3[0]=='$') sval=gethex1();
     if(field3[0]=='%') sval=getbin1();
     if((field3[0]=='\'')||(field3[0]=='"')) sval=getasc1();
     }
   else sval=getint1();
   ch1=sval >> 8;
   ch2=sval & 0xff;
   st[stptr].byte1=ch1;
   st[stptr].byte2=ch2;
   st[stptr].symtype=0;

}
```

A2. Decodificatorul de macro-instructiuni

```
void main(void)
{


init();
initr();
clear_event_buffer();
init_rtc();

// Global enable interrupts
#asm("sei")

while (1)
      {
      timer();
      check_usart0();
      check_usart1();
      #asm("WDR");
      if(rt[offset_QRUN].resource!=0)
      {
      fetch(); /* get opcode */

      switch(opcode)
       {
       case 0x00:
       read_analog_input();
       read_di();
```

```
next();
break;

case 0x01:
break;

case 0x02:
break;

case 0x03:
break;

case 0x04:  /* functia AND */
getopl();
fand();
next();
break;

case 0x05: /* functia NAND */
getopl();
fnand();
next();
break;

case 0x06: /* functia OR */
getopl();
forl();
next();
break;

case 0x07: /* functia NOR */
getopl();
fnor();
next();
break;

case 0x08: /* functia XOR */
getopl();
fxor();
next();
break;

case 0x09: /* functia FF */
getopf();
fff();
next();
break;

case 0x0a: /* functia TIMER */
getopt();
ftimer();
next();
break;

case 0x0b: /* functia COUNTER */
getopc();
fcounter();
next();
break;
```

```
 case 0x0c: /* functia DCOMP */
 getopd();
 fdcomp();
 next();
 break;

 case 0x0d: /* functia ACOMP */
 getopa();
 facomp();
 next();
 break;

 case 0x0e: /* MOVE */
 getopm();
 fmove();
 next();
 break;

 case 0x0f: /* EOC */
 write_do();
 next();
 break;

 case 0x10: /* functia STC */
 getops();
 fstc();
 next();
 break;

 case 0x11: /* functia STIC */
 getopsi();
 fstic();
 next();
 break;

 case 0x12: /* functia OPER */
 getopar1();
 foper();
 next();
 break;

 case 0x13: /* functia MUL */
 getopar2();
 fmul();
 next();
 break;

 case 0x16:
 getope();
 fevent();
 next();
 break;

 default:
 xprg=0;
 break;
 }

} /* end if */
}
```

```
}

// Timer 1 output compare A interrupt service routine
interrupt [TIM1_COMPA] void timer1_compa_isr(void)
{

qtoc1=1;
OCR1=(OCR1AH*256+OCR1AL)+20000;
OCR1AH=(OCR1&0xFF00)>>8;
OCR1AL=OCR1&0xFF;
}


void timer(void)

{
   if(qtoc1==0) return;
   qtoc1=0;
   rtc();
// verificam si decrementam timerele de 10ms
   dectmr10ms();
         if(CNT1 !=0)CNT1--;
         else
         {
           CNT1=KTMR1;
           dectmr100ms();
           if(CNT2 !=0) CNT2--;
           else
            {
            CNT2=KTMR2;
            dectmr1s();
               if(CNT3 !=0) CNT3--;
               else
                {
                CNT3=KTMR3;
                dectmr1m();
                }
            }
         }

}



void dectmr10ms(void)
{
   unsigned char i;
   for(i=0;i<8;i++)
     {
       if(TTAB[i]!=0) TTAB[i]--;
     }
}

void dectmr100ms(void)
{
   unsigned int i;
   for(i=0;i<8;i++)
     {
       if(TTAB[8+i]!=0) TTAB[8+i]--;
     }
}
```

```
void dectmr1s(void)
{
   unsigned int i;
   for(i=0;i<8;i++)
     {
       if(TTAB[16+i]!=0) TTAB[16+i]--;
     }
}

void dectmr1m(void)
{
   unsigned int i;
   for(i=0;i<8;i++)
     {
       if(TTAB[24+i]!=0) TTAB[24+i]--;
     }

}

void rtc(void)
{
  RTC_10ms++;
  if(RTC_10ms>=100)
    {
    RTC_10ms=0;
    RTC_sec++;
    if(RTC_sec>=60)
      {
      RTC_sec=0;
      RTC_min++;
      if(RTC_min>=60)
        {
        RTC_min=0;
        RTC_hour++;
        if(RTC_hour>=24)
          {
          RTC_hour=0;
          RTC_day++;
          if((RTC_day==31)&&((RTC_month%2)==0))
              {
              RTC_day=1;
              RTC_month++;
              }
          if((RTC_day==32)&&((RTC_month%2)==1))
              {
              RTC_day=1;
              RTC_month++;
              }
          if( (RTC_day==30)&&(RTC_month==2)&&((RTC_year%4)==0) )
              {
              RTC_day=1;
              RTC_month++;
              }
          if( (RTC_day==29)&&(RTC_month==2)&&((RTC_year%4)!=0) )
              {
              RTC_day=1;
              RTC_month++;
              }

          if(RTC_month==13)
```

```
              {
              RTC_month=1;
              RTC_year++;
              if(RTC_year==100)
                {
                RTC_year=0;
                }
              }

           }

         }
       }
     }
}

void fetch(void)
{
   opcode=prgbuf[xprg];
}

void next(void)
{
      switch(opcode)
        {
        case 0x00:
        xprg=xprg+1;
        break;
        case 0x01:
        xprg=xprg+1;
        break;
        case 0x02:
        xprg=xprg+1;
        break;
        case 0x03:
        xprg=xprg+1;
        break;
        case 0x04: /* and */
        xprg=xprg+6;
        break;
        case 0x05: /* nand */
          xprg=xprg+6;
        break;
        case 0x06:  /* or */
          xprg=xprg+6;
        break;
        case 0x07:
          xprg=xprg+6;
        break;
        case 0x08:  /* xor */
        xprg=xprg+6;
        break;
        case 0x09: /* FF */
        xprg=xprg+5;
        break;
        case 0x0a: /* timer */
        xprg=xprg+6;
        break;
        case 0x0b: /* counter */
        xprg=xprg+5;
        break;
```

```
        case 0x0c: /* dcomp */
        xprg=xprg+5;
        break;
        case 0x0d: /* acomp */
        xprg=xprg+4;
        break;
        case 0x0e: /* reserved */
        xprg=0;
        break;
        case 0x0f: /* EOC */
        PORTB.7=!(PORTB.7);
        xprg=0;
        break;
        case 0x10: /* STC */
        xprg=xprg+5;
        break;
        case 0x11: /* STIC */
        xprg=xprg+5;
        break;
        case 0x12: /* OPER */
        xprg=xprg+6;
        break;
        case 0x13: /* MUL */
        xprg=xprg+5;
        break;
        case 0x16: /*EVENT */
        xprg=xprg+4;
        break;
        default:
        xprg=0;
        break;
        }
        if(xprg>=PRGSIZE)
         {
          xprg=0;
         }
        dcdstatus();
}

void getopl(void)
{

    dest=prgbuf[xprg+1];
    op1=rt[prgbuf[xprg+2]].resource;
    op2=rt[prgbuf[xprg+3]].resource;
    op3=rt[prgbuf[xprg+4]].resource;
    op4=rt[prgbuf[xprg+5]].resource;

}

/* COUNTER dest,TYPE,CLOCK,EDGE  */
/*              op1   op2   op3  */
/* Atentie TYPE nu este operand imediat!! */

void getopc(void)
{
    dest=prgbuf[xprg+1];
    op1=rt[prgbuf[xprg+2]].resource;
    op2=rt[prgbuf[xprg+3]].resource;
    op3=prgbuf[xprg+4];
}
```

```
/* TIMER    timer_number,TYPE,CLOCK,EDGE,K    */
/*              dest          op1  op2   op3  op4  */

void getopt(void)
{
    dest=(prgbuf[xprg+1])&0x1f; /* dest e offset in TTAB */
    op1=(prgbuf[xprg+2])&0x01;
    op2=(rt[prgbuf[xprg+3]].resource)&0x03;
    op3=(prgbuf[xprg+4])&0x03;
    op4=prgbuf[xprg+5];
}

void getopd(void)
{

    dest=prgbuf[xprg+1];
    op1=rt[prgbuf[xprg+2]].resource;
    op2=rt[prgbuf[xprg+3]].resource;
    op3=prgbuf[xprg+4];

}

/* ACOMP dest,op1,op2 */

void getopa(void)
{

    dest=prgbuf[xprg+1];
    op1=rt[prgbuf[xprg+2]].resource;
    op2=rt[prgbuf[xprg+3]].resource;

}

/* sintaxa pentru functia FF:
      FF    DEST,SET, RESET,PRIORITY
      PRIORITY este operand imediat!!
*/

void getopf(void)
{

    dest=prgbuf[xprg+1];
    op1=rt[prgbuf[xprg+2]].resource;
    op2=rt[prgbuf[xprg+3]].resource;
    op3=prgbuf[xprg+4]; /* operand imediat */

}

/*
      STC   dest, src, condition, edge
      nu are operanzi imediati
*/

void getops(void)
{
    dest=prgbuf[xprg+1];
    op1=rt[prgbuf[xprg+2]].resource; /* src */
    op2=rt[prgbuf[xprg+3]].resource; /* condition */
    op3=(prgbuf[xprg+5])&0x03;       /* edge */
```

```
}


void getopsi(void)
{
    dest=prgbuf[xprg+1];
    op1=prgbuf[xprg+2]; /* operand imediat */
    op2=rt[prgbuf[xprg+3]].resource;
    op3=(prgbuf[xprg+4])&0x03;

}

/* OPER desth,destl,op1,op2,selop */

void getopar1(void)
{

    desth=prgbuf[xprg+1];
    destl=prgbuf[xprg+2];
    op1=rt[prgbuf[xprg+3]].resource;
    op2=rt[prgbuf[xprg+4]].resource;
    op3=rt[prgbuf[xprg+5]].resource;
}

void getopar2(void)
{

    desth=prgbuf[xprg+1];
    destl=prgbuf[xprg+2];
    op1=rt[prgbuf[xprg+3]].resource;
    op2=rt[prgbuf[xprg+4]].resource;
}


void getope(void)
{
    dest=(prgbuf[xprg+1])&0x0f; /* type */
    op1=rt[prgbuf[xprg+2]].resource; /* input */
    op2=(prgbuf[xprg+3])&0x03;       /* edge */
}

void fand(void)
{
    result=rt[dest].resource;
    if((op1&op2&op3&op4) & 0x01)
      {
      result=(result<<1)|0x01;
      }
    else result=result<<1;
    rt[dest].resource=result;
}

void fnand(void)
{
    result=rt[dest].resource;
    if((op1&op2&op3&op4) & 0x01)
      {
      result=(result<<1);
      }
    else result=(result<<1)|0x01;
    rt[dest].resource=result;
```

```
}

void forl(void)
{
   result=rt[dest].resource;
   if((op1|op2|op3|op4) & 0x01)
      {
      result=(result<<1)|0x01;
      }
   else result=result<<1;
   rt[dest].resource=result;

}

void fnor(void)
{
   result=rt[dest].resource;
   if((op1|op2|op3|op4) & 0x01)
      {
      result=(result<<1);
      }
   else result=(result<<1)|0x01;
   rt[dest].resource=result;

}

void fxor(void)
{

   unsigned char temp=0;
   result=rt[dest].resource;

   temp=(op1^op2^op3^op4)&0x01;
   if(temp) result=(result<<1)|0x01;
   else result=result<<1;
   rt[dest].resource=result;
}

/*
  Implementarea functiei FF:
- Daca SET=RESET=0 - starea ramane neschimbata.
- daca SET=RESET=1 atunci destinatia ia valoarea PRIORITY.
- daca SET=1 si RESET=0 atunci destinatia ia valoarea 1
- daca set=0 si RESET=1 atunci destinatia ia valoarea 0
*/

void fff(void)
{
   unsigned char temp=0;
   result=rt[dest].resource;
   op1=op1&0x01; /* set */
   op2=op2&0x01; /* reset */
   op3=op3&0x01; /* priority */
   temp=(op1<<1)|(op2);
   switch(temp)
    {
      case 0:
      break;
      case 1:
      result=result<<1;
```

```
      break;
      case 2:
      result=(result<<1)|0x01;
      break;
      case 3:
      if(op3==0) result=result<<1;
      if(op3==1) result=(result<<1)|0x01;
      break;
    }
    rt[dest].resource=result;

}

void facomp(void)
{

   result=rt[dest].resource;
   if(op1>=op2)
      {
      result=(result<<1)|0x01;
      }
   else
      {
       result=(result<<1);
      }
   rt[dest].resource=result;
}

/*
      Sintaxa DCOMP
      DCOMP dest, op1, op2, mask
      MASK este operand imediat
*/

void fdcomp(void)
{

   result=rt[dest].resource;
   if((op1&op3)==(op2&op3))
      {
      result=(result<<1)|0x01;
      }
   else result=(result<<1);
   rt[dest].resource=result;
}

/*
       Sintaxa pentru STC
       STC dest,op1,condition,edge
*/

void fstc(void)
{
    if((op2&0x03)==op3)
      {
      rt[dest].resource=op1;
      }

}

void fstic(void)
```

```
{
      if( (op2&0x03)==op3 )
      {
      rt[dest].resource=op1;
      }
}


/* OPER desth,destl,op1,op2,op3 */

void foper(void)
{
   int temp;

   if((op3&0x01)==0)  /* adunarea */
     {
     temp=(int)op1+(int)op2;
     rt[desth].resource=(temp&0xff00 >> 8);
     rt[destl].resource=temp&0xff;
     }
   else          /* scaderea */
     {
     temp=op1-op2;
     if(temp<0)
       {
       temp=-temp;
       rt[desth].resource=0x80;
         rt[destl].resource=temp&0xff;
       }
     if(temp>=0)
       {
      rt[desth].resource=0x00;
      rt[destl].resource=temp&0xff;
      }
      }


}
void fmul(void)
{
   unsigned int temp;
   temp=(int)op1*(int)op2;
   rt[desth].resource=temp&0xff00 >> 8;
   rt[destl].resource=temp&0x00ff;
}

/* COUNTER dest,type,clock,edge */
/*           op1  op2   op3 */

void fcounter(void)
{
    unsigned char result;
    unsigned char a,b;

/* verificam intai daca avem front activ la clock */
   a=op2&0x03;
   if((a==0)||(a==0x03)) return; /* nu avem front activ */
   b=op3&0x03; /* izolam ultimii doi biti ai edge si eliminam 0 si 3 */
   if((b==0)||(b==0x03)) return;
   if(a!=b) return;
/* cazul cand avem front activ - facem increment sau decrement in
```

```
functie de TYPE. Dar inainte de asta verificam daca nu apare overflow */
   result=rt[dest].resource;
   if((op1&0x01)==0x01) /* UP counter */
     {
       if(result==0xff) return;
       else result++;
       rt[dest].resource=result;
     }
   else /* down counter */
     {
       if(result==0x00) return;
       else result--;
       rt[dest].resource=result;
     }
}


/* TIMER    timer_number,TYPE,CLOCK,EDGE,K    */
/*              dest         op1  op2   op3  op4  */
/*
   Semnificatia EDGE cand TYPE=0 (monostabil)
   EDGE = 01 monostabil declansabil pe front crescator
   EDGE = 10 monostabil declansat pe fron cazator
   EDGE = 00 monostabil redeclansabil cand intrarea CLOCK este 0
   EDGE = 11 monostabil redeclansabil cand intrarea CLOCK este 1
   Semnificatia EDGE cand TYPE=1 (astabil)
   EDGE=00 - timerul oscileaza (schimba starea) cand CLOCK=0
   EDGE=11 - timerul schimba starea cand CLOCK=1

*/


void ftimer(void)
{
   switch(op1)
     {
     case 0:  /* monostabil */
       switch(op3)
       {
       case 0: /* monostabil redeclansabil pe palier 0 */

/* verificam daca CLOCK-ul (op2) este 0. daca DA e conditie de redeclansare
*/
/* cazul cand avem conditie de redeclansare */

        if( (op2&0x03)==0)
        {
         TTAB[dest]=op4; /* redeclansarea efectiva */
         result=1;
        }
/* cazul cand nu avem conditie de redeclansare */
        else
        {
          if(TTAB[dest]==0) result=0;
          if(TTAB[dest]!=0) result=1;
        }
        break;
      case 1: /* declansare pe front crescator */
      if( (op2&0x03)==0x01) /* avem conditie de declansare */
        {
         TTAB[dest]=op4; /* redeclansarea efectiva */
```

```
          result=1;
         }
         else
         {
           if(TTAB[dest]==0) result=0;
           if(TTAB[dest]!=0) result=1;
         }

       break;

       case 2: /* declansare pe front cazator */
       if( (op2&0x03)==0x02) /* avem conditie de declansare */
         {
          TTAB[dest]=op4; /* redeclansarea efectiva */
          result=1;
         }
         else
         {
           if(TTAB[dest]==0) result=0;
           if(TTAB[dest]!=0) result=1;
         }

       break;

       case 3: /* redeclansare pe palier 1 */
       if( (op2&0x03)==0x03)
         {
          TTAB[dest]=op4; /* redeclansarea efectiva */
          result=1;
         }
/* cazul cand nu avem conditie de redeclansare */
         else
         {
           if(TTAB[dest]==0) result=0;
           if(TTAB[dest]!=0) result=1;
         }

       break;
       }

      break;

/* Cazul cand avem oscilator */

      case 1: /* astabil */
/* verificam intai conditia de oscilatie */
      op3=op3&0x03;

      if( ((op2==0)&&(op3==0)) || ((op2==3)&&(op3==3)) )
      {

        if(TTAB[dest]!=0)
        {
        result=OSCVAL[dest];
        }
        if(TTAB[dest]==0)
        {
        TTAB[dest]=op4;
        OSCVAL[dest]=~OSCVAL[dest];
        result=OSCVAL[dest];
        }
```

```
     break;
     }
     else result=0;
     }
/* modificarea efectiva a valorilor YTMRx */

     if(result==0)
rt[offset_YTMR0+(int)(dest)].resource=rt[offset_YTMR0+(int)(dest)].resource
<<1;
     else
rt[offset_YTMR0+(int)(dest)].resource=(rt[offset_YTMR0+(int)(dest)].resourc
e<<1)|1;

}


void dcdstatus(void)
{
 unsigned char status;
 unsigned int i;
 status=(rt[offset_STATUS].resource)&0x0f;
 for(i=0;i<16;i++)
   {
   if(i==status)
     {
     rt[offset_S0+i].resource=((rt[offset_S0+i].resource)<<1|1);
     }
   else
     {
     rt[offset_S0+i].resource=(rt[offset_S0+i].resource)<<1;
     }
   }
}
```

ANEXA B

DETALII PRIVIND IMPLEMENTAREA ALGORITMULUI DE URMARIRE A UNEI TRAIECTORII CU AJUTORUL CONCEPTULUI DE FEROMONI VIRTUALI

Implementarea unui algoritm de conducere a robotilor mobili la urmarirea unei traiectorii cu ajutorul conceptului de feromoni virtuali implica realizarea urmatoarelor aplicatii software distincte:

- Aplicatia care ruleaza la nivelul serverului de feromoni. Aceasta citeste dintr-un fisier harta mediului, calculeaza concentratiile de feromoni (Left, Right) corespunzatoare pozitiei curente raportate de roboti si implementeaza protocolul de comunicatie.
- Aplicatia de conducere a robotului. Aceasta ruleaza pe microcontrollerul de navigatie si implementeaza urmatoarele functii: comunicatia cu robotul, implementarea algoritmului (fuzzy) de conducere si comunicatia cu serverul de feromoni.
- Editorul de harti – permite crearea si editarea hartilor mediului, intr-un format usor de gestionat de serverul de feromoni si, in acelasi timp, lizibil pentru operatorul uman.

B1. Implementarea Serverului de Feromoni

```
void main(void)
{

  int i;

  textmode(3);
  textbackground(BLUE);
  textcolor(LIGHTGRAY);
  clrscr();
  x=y=1;

  if( (fp=fopen("map.txt", "rb"))==NULL )
```

```
   {
   printf("Cannot open map file.\n");
   exit(1);
   }
read_map();
fclose(fp);

gettextinfo(&screen);
switch(screen.currmode) {
  case 1:
  case 2:
  case 3:
  case 4:
  case 5:
  case 6:  vram=MK_FP( 0xB800, 0); break;
  case 7:  vram=MK_FP( 0xB000, 0); break;
  default: vram=MK_FP( 0xA000, 0); break;
}

set_baud();

window(1,1,80,1);
textbackground(LIGHTGRAY);
textcolor(BLACK);

clrscr();
gotoxy(1,1);
gettextinfo(&screen);
top();
window(1,2,80,25);
textbackground(BLUE);
textcolor(LIGHTGRAY);
gettextinfo(&screen);

hscreen.sizebuf = 4000U;
if(NULL == (hscreen.savebuf = malloc(hscreen.sizebuf))) {
   window(1,1,80,25);
   clrscr();
   printf("Allocate for screen save memory failed.\n");
   exit(8);
}

x=y=1;

gotoxy(x,y);
hook();
init_com();

rbfull=1;

while (1) {
  get_comm_byte();


if(rbfull==1)
  {
  rbfull=0;
  parse_buffer();
  locate();
  SIGMA=680;
  compute_pheromone(SIGMA);
```

```
    make_packet();
    send_packet();

    } /* end if */

.....

int parse_buffer(void)
{
  OPCODE=recbuf[0];
  RID=recbuf[1];
  current_x=pack(recbuf[2],recbuf[3]);
  current_y=pack(recbuf[4],recbuf[5]);
  current_phi=pack(recbuf[6],recbuf[7]);
  current_phi_radians=(current_phi*3.14)/180;
  return(1);
}
void read_map(void)
{
   int pval;
   int i;
   cell_counter=0;
   i=0;

   while(!feof(fp))
   {
   if(read_number()) break;
   pval=atoi(asc_value);
   MAP[i]=pval;
   i++;
   }

}

double cell(int i)
{
   int N, S;
   double a;
   int temp;

   N=sqrt(cell_counter);
   S=cell_counter;
   if (i>S) return(0);

   temp= (-(N/2) + (i%N));
   temp=temp*CELLSIZE;
   cell_x=temp;
   cell_y=((N/2) - i/N)*CELLSIZE;
   cell_value=MAP[i];
   if(cell_value==0) return(0);
   if(cell_value>=0x80) return(0);
   if((A*cell_x+B*cell_y+C) >= 0) a=1;
   else a=0;
   return(a);

}

/*
find_cell - primeste ca input coordonatele curente current_x, current_y
Determina celula din MAP[] in care se gaseste robotul
```

```
Intoarce indexul in matricea MAP[] a celulei curente.
Intarce -1 daca pozitia curenta e in afara matricii
*/

int find_cell(void)
{
  int N, i, j, X,Y;

  N=sqrt(cell_counter);
  X=current_x+(N/2)*CELLSIZE;
  Y=current_y+(N/2)*CELLSIZE;
  i=X/CELLSIZE;
  j=Y/CELLSIZE;
  if((i>N/2)||(j>N/2)) return(-1);
  current_cell=j*N+i;
  return(current_cell);
}


/* functia locate(void) primeste ca input coordonatele curente
ale robotului.
Calculeaza coordonatele antenelor: rax, ray, lax lay
rax=current_x+(b/2)*sin(phi)
lax=current_x-(b/2)sin(phi)
ray=current_y-(b/2)cos(phi)
lay=current_y-(b/2)cos(phi)
Calculeaza si parametrii care descriu dreapta care trece prin antenele
dreapta/stanga.

A=y2-y1
B=x1-x2
C=x2*y1-x1*y2
Totodata actualizeaza valoarea indexului current_cell (asta e indexul
pozitiei curente in MAP[i]

*/

void locate(void)
{

  double RAX,LAX,RAY, LAY;

  RAX=(double)current_x+(aperture/2)*sin(current_phi_radians);
  LAX=(double)current_x-(aperture/2)*sin(current_phi_radians);
  RAY=(double)current_y-(aperture/2)*cos(current_phi_radians);
  LAY=(double)current_y+(aperture/2)*cos(current_phi_radians);
  A=LAY-RAY;
  B=RAX-LAX;
  C=LAX*RAY-RAX*LAY;
  rax=floor(RAX);
  ray=floor(RAY);
  lax=floor(LAX);
  lay=floor(LAY);

}

double distance(int x1, int y1, int x2, int y2)
{
   double d,d1,d2;
   d1=(double)x1-(double)x2;
   d1=pow(d1,2);
```

```
   d2=(double)y1-(double)y2;
   d2=pow(d2,2);
   d=sqrt(d1+d2);
   return(d);

}

int miu(double dist)
{
   int i;
   double u;
   u=cell_value*(1-(dist/SIGMA));
   u=100*u;
   i=floor(u);
   if(i<0) return(0);
   else return(i);
}


void compute_pheromone(int sensitivity)
{
    int i;
    double dist_l, dist_r, k;
    int PiR, PiL;
    double PHI_i;
    double temp;
    double sum, sum1, sum2;

    PR=0;
    PL=0;
    PiR=0;
    PiL=0;

    sum1=0;
    sum2=0;

    for(i=0;i<cell_counter;i++)
      {
      if(MAP[i]==0) continue;
      k=cell(i);
      if( (k>0) && (k<0x80))
      {
      dist_r=distance(cell_x,cell_y,rax,ray);
      if(dist_r<CELLSIZE/100) continue;
      temp=(cell_x-rax)/(dist_r);
      PHI_i=asin(temp);
      PiR=k*miu(dist_r);
      sum1=sum1+PiR*sin(PHI_i);
      sum2=sum2+PiR*cos(PHI_i);
      }

      }
      sum=pow(sum1,2)+pow(sum2,2);
      PR=sqrt(sum);

    sum1=0;
    sum2=0;

    for(i=0;i<cell_counter;i++)
      {
      k=cell(i);
```

```
      if( (k>0) && (k<0x80))
      {
      dist_l=distance(cell_x,cell_y,lax,lay);

      if(dist_l<CELLSIZE/100) continue;
      temp=(cell_x-lax)/(dist_l);
      PHI_i=asin(temp);
      PiL=k*miu(dist_l);
      sum1=sum1+PiL*sin(PHI_i);
      sum2=sum2+PiL*cos(PHI_i);
      }

      }
      sum=pow(sum1,2)+pow(sum2,2);
      PL=sqrt(sum);
      qnil=0;
      if((PL==0)&&(PR==0))
       {
       qnil=1;
       }

}


void interrupt int_svc(void)
{
  disable();
  serr=inp(pad_lsr) & 0x0e;

  if(serr) {
    outp(0x20,0x20);
    rxdata=inp(pad_rbr);
    intstatus=0;
    rbptr=0;
    error_count++;
    enable();
    return;
  }

  rec_data=inp(pad_rbr);

  switch(intstatus)
    {
       case 0: /* asteptam 0x10 */
       if(rec_data==0x10)
       {
       intstatus=1;
       }

       break;

/* intstatus=1 - se verifica daca a venit 0x02. Daca nu, se revine in 0
daca da, se trece in starea 2 */

       case 1:
       if(rec_data==0x02)
       {
       intstatus=2;
       }

       else
```

```
        {
        intstatus=0;
        error_count++;
        rbptr=0;
        }
        break;

/* intstatus=2. Verificam daca a venit 0x10. Daca nu, trecem in starea 0.
Daca a venit, trecem in starea 3 */

        case 2:
        if(rec_data==0x10)
        {
        intstatus=3;
        }
        else
        {
           intstatus=0;
           error_count++;
           rbptr=0;
        }
        break;

/* intstatus=3. Verificam daca a venit 0x10. Daca DA, trecem in 4. */

        case 3:
        if(rec_data==0x10)
        {
        intstatus=4;
        }
        else
        {
           intstatus=0;
           rbptr=0;
           error_count++;
        }
        break;

/* intstatus=4. se preia caracterul salvat anterior in chsav in buffer
si se afiseaza. Apoi se verifica daca s-a primit 0x10. Daca Nu este DLE,
ramanem in starea 4. Daca este DLE, trecem in starea 5, unde verificam daca
e DLE stuffing, sau sfarsit de pachet. */

       case 4:
       recbuf[rbptr++]=rec_data;
       if(rbptr>=RBSIZE)
         {
         rbptr=0;
         intstatus=0;
         rbfull=0;
         count=0;
         }
       if(rec_data==0x10)
           {
           intstatus=5;
           }
        break;

/* intstatus=5. Se verifica daca s-a primit 10 (DLE stuffing) sau
0x03 - incheiere pachet. Daca este DLE, se trece in starea 4.
Daca este 0x03 se trece in starea 0 - s-a terminat de primt pachetul. */
```

```
        case 5:
        if(rec_data==0x10)
           {
           intstatus=4;
           break;
           }
        if(rec_data==0x03)
           {
           intstatus=0;
           recbuf[rbptr++]=0x03;
           count=rbptr-2;
           rbptr=0;
           rbfull=1;
           break;
           }
        intstatus=0;
        rbptr=0;
        break;

    } /* end switch */


  if( ++endq == 2048) endq=0;
  hold_buf[endq] = rec_data;


  outp( 0x20, 0x20); /* Signal End of Interrupt to 8259  */
  enable();
}

void make_packet(void)
{
   xmtbuf[0]=OPCODE|0x80;
   xmtbuf[1]=RID;
   xmtbuf[2]=PL;
   xmtbuf[3]=PR;
   xmtbuf[4]=0;
   xmtbuf[5]=0;
   xcnt=6;

}

void send_header(void)
{
   send_char(0x10);
   send_char(0x02);
   send_char(0x10);
   send_char(0x10);
}

void send_trailer(void)
{
   send_char(0x10);
   send_char(0x03);
}

void send_packet(void)
{
  int i;
  send_header();
  for(i=0;i<xcnt;i++) send_char(xmtbuf[i]);
```

```
  send_trailer();
}
```

B2. Implementarea Algoritmului Fuzzy de Conducere

```
int DOM_E[3];   //Degrees of Membership for e(t)
int DOM_ED[3];  //Degrees of Membership for e'(t)

int antecedent_table[9];

int SGTAB_L[9]={M,M,L,         //Singleton tables
                H,M,M,
                H,L,L};

int SGTAB_R[9]={L,L,M,
                M,M,H,
                L,M,H};

void fuzzy(void)
{
  compute_dom();
  antecedent();
  crisp();
}

/*
aici se actualizeaza valorile DOM_E[3] si DOM_ED[3]
DOM[0] - apartenenta la N
DOM[1] - apartenenta la Z
DOM[2] - apartenenta la P
*/


void compute_dom(void)
{
   double temp;
   if(error<=-ME)
     {
     DOM_E[0]=100;
     DOM_E[1]=0;
     DOM_E[2]=0;
     }
   if((error<0)&&(-ME<error))
     {
     temp=((double)100*(double)abs(error))/(double)ME;
     DOM_E[0]=floor(temp);
     DOM_E[1]=100-DOM_E[0];
     DOM_E[2]=0;
     }
   if((error>=0)&&(error<ME))
     {
     temp=((double)100*(double)error)/(double)ME;
     DOM_E[2]=floor(temp);
     DOM_E[1]=100-DOM_E[2];
     DOM_E[0]=0;
     }
   if(error>=ME)
     {
     DOM_E[0]=0;
     DOM_E[1]=0;
     DOM_E[2]=100;
```

```
      }
    if(error_dot<=-MED)
      {
      DOM_ED[0]=100;
      DOM_ED[1]=0;
      DOM_ED[2]=0;
      }
    if((error_dot<0)&&(-MED<error_dot))
      {
      temp=((double)100*(double)abs(error_dot))/(double)MED;
      DOM_ED[0]=floor(temp);
      DOM_ED[1]=100-DOM_ED[0];
      DOM_ED[2]=0;
      }
    if((error_dot>=0)&&(error_dot<MED))
      {
      temp=((double)100*(double)error_dot)/(double)MED;
      DOM_ED[2]=floor(temp);
      DOM_ED[1]=100-DOM_ED[2];
      DOM_ED[0]=0;
      }
    if(error_dot>=MED)
      {
      DOM_ED[0]=0;
      DOM_ED[1]=0;
      DOM_ED[2]=100;
      }

}

void antecedent(void)
{
  int i, j, k;
  int x;
  k=0;
  for(i=0;i<3;i++)
    {
    for (j=0;j<3;j++)
     {
     x=min(DOM_E[i],DOM_ED[j]);
     antecedent_table[k++]=x;
     }
    }
}

void crisp(void)
{
   int i;
   long int sigma;
   long int sum;
   sigma=0;
   sum=0;

   if(qnil)
     {
     VLEFT=60;
     VL=4;
     VRIGHT=60;
     VR=4;
     return;
     }
```

```
   for(i=0;i<9;i++)
     {
      sigma=sigma+(long)SGTAB_L[i]*(long)antecedent_table[i];
      sum=sum+(long)antecedent_table[i];
     }
   VLEFT=sigma/sum;
   VL=(VLEFT/20)&0xFF;
   sigma=0;
   sum=0;
   for(i=0;i<9;i++)
     {
      sigma=sigma+(long)SGTAB_R[i]*(long)antecedent_table[i];
      sum=sum+(long)antecedent_table[i];
     }
   VRIGHT=sigma/sum;
   VR=(VRIGHT/20)&0xFF;
}



int compute_error(void)
{
   int temp;
   temp=PL-PR;
   return(temp);
}

int compute_error_dot(void)
{
   int temp;

   if(qfirst)
     {
     qfirst=0;
     return(0);
     }
   temp=error-prev_error;
   prev_error=error;
   return(temp);
}
```

B3. Note despre Editorul de Hărți

Din perspectiva serverului de feromoni, o hartă este o matrice cu structura prezentată în figura 7.6, stocată intr-un fişier text. Editorul de hărti gestionează aceste fişiere şi oferă o reprezentare vizuală a traseelor de feromoni, ca în figura B3.

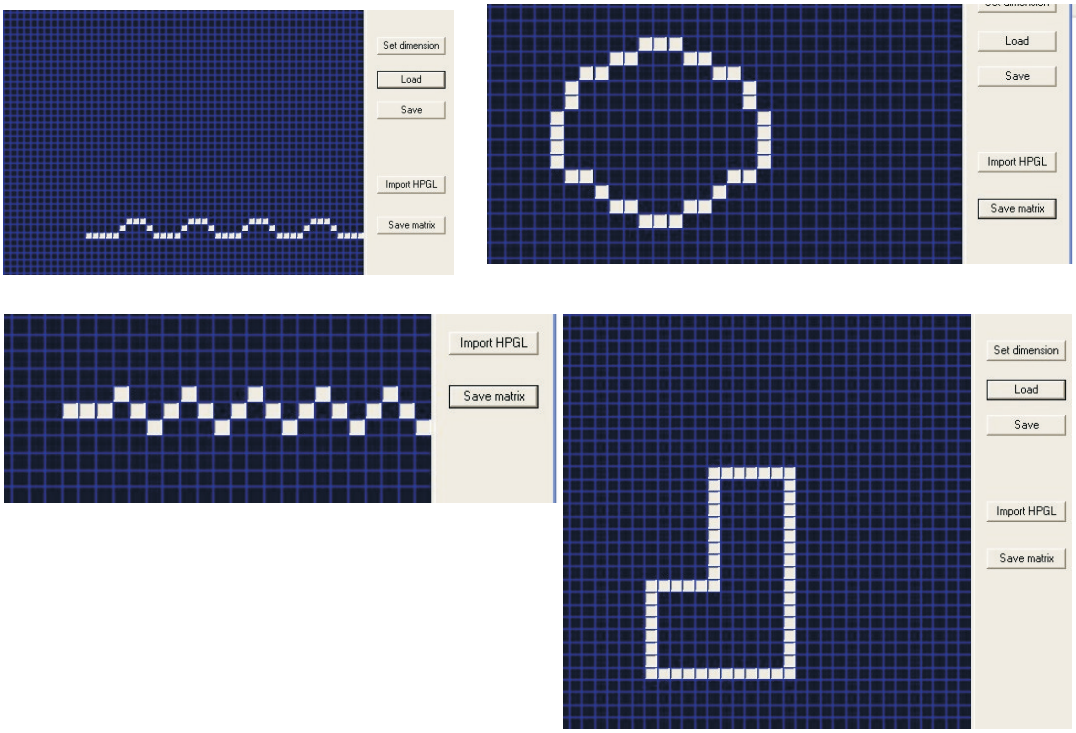


Fig. B3 Exemple de reprezentare a traseelor de feromoni în editorul de hărți

ANEXA C

DETALII PRIVIND IMPLEMENTAREA ALGORITMULUI "BUBBLE REBOUND" PENTRU OCOLIREA OBSTACOLELOR

Aplicatia care implementeaza algoritmul "Bubble Rebound" de ocolire a obstacolelor ruleaza pe microcontrollerul de navigatie. Computerul de la sol a fost folosit la teste doar pentru definirea tintelor de navigatie.

```
void main(void)
{

int tempx, tempy;

init();
init_usart0();
init_usart1();
init_sonar();
VL=0;
VR=0;
current_x=0;
current_y=0;
prev_error=0;
// Global enable interrupts

#asm("sei")

start:

send_SYNC0();
delay_ms(1000);
send_SYNC1();
delay_ms(1000);
send_SYNC2();
delay_ms(1000);
send_control_packet(OPEN,0);
delay_ms(1000);
send_control_packet(ENABLE,1);
delay_ms(1000);

TTAB[0]=200;
robot_state=0;



while (1)
```

```
      {
      timer();

        if(rbfull0)
        {
        rbfull0=0;
        parse0();
        switch(remote_command)
          {
          case 0:
          send_char0('0');
          goto start;
          break;
          case 1:
          report_sonar();
          break;
          case 2:   //F4 - comanda stop
          send_control_packet(RVEL,0);
          send_control_packet(MOVE,0);
          robot_state=0;
          send_control_packet(RVEL,0);
          send_control_packet(MOVE,0);
          break;

          case 254:
          define_target(target_x,target_y);
          robot_state=1;
          send_char0(0x01);
          break;

          case 0xFF:
          send_char0(0xFF);
          break;
          }

        };
    if(rbfull1)
    {
       rbfull1=0;
       parse1();
    }

    if(TTAB[1]!=0) continue;
    TTAB[1]=DELTA_T;
    switch(robot_state)
       {
       case 0: //Asteapta o comanda de miscare si coordonatele tintei

       break;

       case 1: /* calculeaza heading tinta lanseaza comanda rotatie */
       check_stall();
       compute_target_heading();
       send_control_packet(HEAD,target_heading);
       robot_state=2;
       send_char0(0x02);
       break;

/* asteapta terminarea miscarii de rotatie */

       case 2:
```

```
check_stall();
if(TTAB[0]==0)
  {
  send_SYNC0();
  TTAB[0]=HEARTBEAT;
  }
if(check_heading())
  {
  robot_state=3;
  send_char0(0x03);
  }
break;

case 3:
check_stall();
fuzzy();
send_control_packet(VEL2,REALROBOT);
if( ((VR==H)&&(VL=L))||((VL==H)&&(VR=L)) )
{
send_control_packet(RVEL,0);
send_control_packet(MOVE,0);
qobstacle=0;
restore_target();
robot_state=1;
send_char0(0x01);
}
else
{
robot_state=4;
send_char0(0x04);
PORTD.5=1;
}
break;

case 4:
 if(TTAB[0]==0)
 {
 TTAB[0]=HEARTBEAT;
 send_SYNC0();
 }
check_stall();
if(check_for_obstacles())
{
robot_state=5;
send_char0(0x05);
break;
}
compute_distance2target();
if(distance2target<NEAR)
{
      send_control_packet(RVEL,0);
      send_control_packet(MOVE,0);
      robot_state=0;
      send_char0(0);

}
else {
  robot_state=3;
  send_char0(0x03);
  PORTD.5=0;
  }
```

```
        break;

/* obstacol detectat */

        case 5:
        check_stall();
        send_control_packet(RVEL,0);
        send_control_packet(MOVE,0);

        if(qobstacle==0)
        {
        compute_new_heading();
        tempx=current_x+DETOUR*cos(new_heading_radians);
        tempy=current_y+DETOUR*sin(new_heading_radians);
        define_target(tempx,tempy);
        send_control_packet(HEAD,new_heading);
        qobstacle=1;
        robot_state=6;
        send_char0(0x06);
        break;
        }
        else
        {
        compute_new_heading();
        send_control_packet(HEAD,new_heading);
        robot_state=6;
        send_char0(0x06);
        break;
        }
/* asteapta terminarea miscarii de rotatie */
        case 6:
        check_stall();

        if(abs(current_heading-new_heading)>ANGULAR_LIMIT)
         {
         if(TTAB[0]==0)
          {
          send_control_packet(HEAD,new_heading);
          TTAB[0]=HEARTBEAT;
          send_char0(0xff);
          report(sigma1);
          report(sigma2);
          report(temp2);
          send_char0(0xff);
          }

         break;
        }
        check_for_obstacles();
        if(obstacle_ahead)
        {
        robot_state=5;
        send_char0(0x05);
        break;
        }
        else
         {
          robot_state=7;
          send_char0(0x07);
         }
```

```
        break;

        case 7:
        check_stall();
        fuzzy();
        send_control_packet(VEL2,REALROBOT);
        if( ((VR==H)&&(VL=L))||((VL==H)&&(VR=L)) )
        {
        send_control_packet(RVEL,0);
        send_control_packet(MOVE,0);
        qobstacle=0;
        restore_target();
        robot_state=1;
        send_char0(0x01);
        }
        else
        {
        robot_state=8;
        send_char0(0x08);
        PORTD.5=1;
        }
        break;

        case 8:
        if(TTAB[0]==0)
        {
        TTAB[0]=HEARTBEAT;
        send_SYNC0();
        }
        check_stall();
        if(check_for_obstacles())
        {
        robot_state=5;
        send_char0(0x05);
        break;
        }

        compute_distance2target();
        if(distance2target<NEAR) /* am ajuns la tinta intermediara */
        {
              send_control_packet(RVEL,0);
              send_control_packet(MOVE,0);
              qobstacle=0;
              restore_target();
              robot_state=1;
              send_char0(0x01);
              break;
        }

        robot_state=7;
        send_char0(0x07);
        PORTD.5=0;
        break;

        } //end switch

    } // end while

}; // end main

int parse1(void)
```

```
{
unsigned char SIP_type;

unsigned char cnt1;
cnt1=rxbuf1[2];
calc_cks(rxbuf1);
SIP_type=rxbuf1[3];
if((SIP_type!=0x32)&&(SIP_type!=0x33))
  {
  return(0);
  }
if( (cksh==rxbuf1[(int)cnt1+1]) && (cksl==rxbuf1[(int)cnt1+2]))
  {
  update_status();
  update_coord();
  update_sonar();
  return(1);
  }
return(0);
}

void update_status(void)
{
  unsigned char ch;
  if(rxbuf1[3]==0x32) qmove=0;
  if(rxbuf1[3]==0x33) qmove=1;
  ch=rxbuf1[15] | rxbuf1[16];
  ch=ch & 0x01;
  if(ch==0x01)
   {
   qstall=1;
   }
  else
    {
    qstall=0;
    };
  left_vel=bytes2int(rxbuf1[10],rxbuf1[11]);
  right_vel=bytes2int(rxbuf1[12],rxbuf1[13]);

}

void update_coord(void)
{
   float temp;
   current_x=bytes2int(rxbuf1[4],rxbuf1[5]);
   current_y=bytes2int(rxbuf1[6],rxbuf1[7]);
   temp=(float)bytes2int(rxbuf1[8],rxbuf1[9]);
   temp=(temp*(float)180)/((float)2048);
   current_heading=floor(temp);
   if(current_heading<0) current_heading=360+current_heading;
   if(current_heading==360) current_heading=0;

}

void update_sonar(void)
{
    unsigned char cnt1;
    unsigned char sonar_address;
    unsigned char sonar_count;
    int i;
```

```
    cnt1=rxbuf1[2];
    if(cnt1<25) return;
    sonar_count=rxbuf1[22];

    for(i=0;i<sonar_count;i++)
    {
    sonar_address=rxbuf1[23+3*i];
    sonar[sonar_address]=(bytes2int(rxbuf1[24+3*i],rxbuf1[25+3*i]))/10;
    }
    for(i=0;i<8;i++)
      {
      if(sonar[i]>500) sonar[i]=500;
      }

}

int bytes2int(unsigned char ch1, unsigned char ch2)
{
   unsigned int val;
   val=(unsigned int)ch2<<8|(unsigned int)ch1;
   return(val);
}

/* calculeaza si actualizeaza variabila distance2target */

void compute_distance2target(void)
{
double temp;
temp=(long)(target_x-current_x)*(long)(target_x-current_x)+(long)(target_y-
current_y)*(long)(target_y-current_y);
distance2target=floor(sqrt(temp));
if(distance2target<=5) distance2target=5;
}

int check_for_obstacles(void)
{
  int i,r;
  r=0;
  for(i=1;i<7;i++)
    {
    if(sonar[i]<=LIMIT[i]) r=1;
    }
  if((sonar[3]<=LIMIT[3]) || (sonar[4]<=LIMIT[4]))
    {
    obstacle_ahead=1;
    }
  else
    {
    obstacle_ahead=0;
    }
  return(r);
}

void compute_new_heading(void)
{
       swing_angle=check_for_free_area();
       new_heading=current_heading+swing_angle;
       if(new_heading<0) new_heading=360+new_heading;
       if(new_heading>360) new_heading=new_heading-360;
       new_heading_radians=((double)(new_heading)*3.14)/180.0;
}
```

```
int check_for_free_area(void)
{
   int i;

   check_for_obstacles();
   sigma1=0;
   sigma2=0;

   for(i=0;i<8;i++)
     {
     sigma1=sigma1+(SPTAB[i]*sonar[i]);
     sigma2=1+sigma2+sonar[i];
     }
     temp=(double)sigma1/(double)sigma2;
     temp1=(temp*180.0)/8.0;
     i=floor(temp1);
     temp2=floor(temp1);
     if((abs(i)<30)&&(obstacle_ahead))
       {
      if(i>=0) i=i+45;
      if(i<0) i=i-50;
       }

     return(i);
}
```

ANEXA C
CURRICULUM VITAE

1. Nume: SUSNEA
2. Prenume: IOAN
3. Data si locul nasterii: 14 Septembrie 1955, Galati, jud. Galati
4. Cetatenie: Romana
5. Stare civila: Necasatorit
6. Studii:

Institutia	Institutul Politehnic Bucuresti	Facultatea de Electronica si Telecomunicatii
Perioada: de la (luna, anul) pana la (luna, anul)	Sept. 1975	iulie 1980
Grade sau diplome obtinute	Diploma	

7. **Titlul stiintific**: doctorand in ingineria sistemelor (Contribuţii la elaborarea unor soluţii încorporate (embedded) de conducere în timp real a sistemelor robotice şi vehiculelor autonome) Scoala doctorala de pe langa Universitatea Dunarea de Jos, Galati. Conducator stiintific: Prof. Dr. Ing. Adrian Filipescu.

8**. Experienta profesionala**:

Perioada:	din oct.2008 pana in prezent
Locul:	Galati
Institutia:	Universitatea Dunarea de Jos, Galati, Facultatea de Stiinta Calculatoarelor
Functia:	Sef de lucrari
Descriere:	Cursurile de Proiectarea Sistemelor cu Microprocesoare, Comunicatii de Date in Sisteme Distribuite
Perioada:	din 1997 pana in oct. 2008
Locul:	Galati
Institutia:	AWA TRADING SRL
Functia:	Inginer
Descriere:	Proiectare echipament electronic, intregrator de sisteme de automatizare
Perioada:	1990-1991
Institutia:	Secutron Inc. (in prezent, parte a grupului DSC, Canada)
Locul:	Toronto, Ontario, Canada
Functia:	Inginer
Descriere:	Proiectare centrale de alarma anti incendiu
Perioada:	1980-1990
Locul:	Galati
Institutia:	IIRUC Bucuresti, filiala Galati
Functia:	inginer
Descriere:	- Dezvoltare echipamente de diagnostic si testare automata - Trainer pentru personalul implicat in diagnosticarea defectiunilor si depanarea echipamentelor de calcul si automatizari industriale
Perioada:	Anul universitar 1995-1996, 1997-1998, 2007-2008

Locul:	Galati
Institutia:	Universitatea Dunarea de Jos
Functia:	Sef-lucrari asociat
Descriere:	Cursuri de : proiectarea sistemelor cu microprocesoare, comunicatii de date in sisteme distribuite, elemente si structuri de automatizare

9. Locul de munca actual si functia: Universitatea Dunarea de Jos, Galati, Sef de lucrari

10 . Vechime la locul de munca actual: 1 an

11. Brevete de inventii:
- WO2005107299 "Phone diversifications", 2005 in colaborare cu John W.Halpern
- Ioan Susnea, Grigore Vasiliu *Sistem de localizare pentru roboti mobili si vehicule autonome*, Cerere inregistrata la OSIM sub numarul A/01021, din 04-12-2009

12. Lucrari publicate:

I. Carti in edituri internationale de renume

[1] **Susnea I**. Mitescu M. *Microcontrollers in practice* ISBN: 3540253017, Springer Verlag, New York, Heidelberg, 2005

II. Carti in edituri romanesti recunoscute de CNCSIS

[2] Grigore Vasiliu, **Ioan Susnea** *Sisteme computerizate de masurare* ISBN: 978-973-755-452-9, Editura Matrix Rom, Bucuresti, 2009
[3] **Ioan Susnea**, Griogore Vasiliu *Sisteme distribuite pentru monitorizarea si conducerea proceselor. O introducere practica*, ISBN: 978-973-755-542-5, Editura Matrix Rom, Bucuresti, 2009.
[4] Grigore Vasiliu, **Ioan Susnea** *Mecatronica si microsisteme de actionare*, ISBN: 978-973-755-546-5, Editura Matrix Rom, Bucuresti, 2009

III. Lucrari publicate in jurnale indexate ISI

[4] **Susnea I.** Vasiliu G. Filipescu A. Radaschin A., *Virtual Pheromones for Real-Time Control of Autonomous Mobile Robots*, in Studies of Informatics and Control, Vol 18, issue 3, 2009, pp: 233-240, ISSN:1220-1760

IV. Lucrari publicate in periodice interne recunoscute de CNCSIS

[5] **Susnea I**, Filipescu Adrian, Filipescu Adriana, Coman G., Wheeled Mobile Robot Control Using Virtual Pheromones, Petroleum-Gas University of Ploiesti Bulletin, Tehnical Series, Vol LXI, no.3/2009, ISSN 1224-8495, pp. 117-126.

VI. Lucrari publicate in volumele unor conferinte internationale indexate ISI

[6] **Susnea I**. Vasiliu G, Filipescu A, Coman G., On the Implementation of a Robotic Assistant for the Elderly. A Novel Approach, 7th WSEAS Int. Conf. on Computational, Iintelligence Man-machine Systems and Cybernatics (CIMMACS '08), Cairo, Egipt, December 29-31, 2008, pp.215-220, ISBN 978-960-474-049-9, ISSN: 1790-5117 Proceedings + Book published by WSEAS Press that participate in ISI and all the other important international citation indices: www.worldses.org/indexes, (Proceedings ISI quoted, www.ieeexplore.ieee.org) Published by WSEAS Press
[7] **Susnea I**, Filipescu A, Vasiliu G., Filipescu S., Path Following, Real-time, Embedded Fuzzy Control of a Mobile Platform Wheeled Mobile Robot, IEEE International Conference on Automation and Logistics, ICAL 2008, 1-3 Sep., Qingdao, China, IEEE ICAL 2008 CD Proceedings, pp.91-96, IEEE Catalog Number: CFP08CAL, ISBN: 978-1-4244-2503-7, Library of Congress: 2008903794 (Proceedings ISI quoted, www.ieeexplore.ieee.org).
[8] **Susnea I**, Vasiliu G, Filipescu A, Real-Time, Embedded Fuzzy Control of the Pioneer3-DX Robot for Path Following, Proceedings of 12th WSEAS International Conference on SYSTEMS, Heraklion, Greece, July 22-24, 2008, pp.334-338, ISBN: 978-960-6766-83-1, ISSN: 1790-2769, Published by WSEAS Press (www.wseas.org, www.worldses.org/indexes).

[9] **Susnea I** . Vasiliu G, Filipescu A, RFID Digital Pheromones for Generating Stigmergic Behaviour to Autonomous Mobile Robots, 4th WSEAS Int. Conf. on Dynamic Systems and Control (Control '08), CORFU ISLAND, GREECE, October 26-28, 2008, pp.20-24, ISBN 978-960-474-014-7, ISSN: 1790-2769 Proceedings + Book published by WSEAS Press that participate in ISI and all the other important international citation indices: www.worldses.org/indexes, (Proceedings ISI quoted, www.ieeexplore.ieee.org) Published by WSEAS Press.
[10] Filipescu A, **Susnea I**, Stancu Al, Stamatescu G, Path following, real-time, embedded fuzzy control of a mobile platform pioneer 3-DX, 8th WSEAS International Conference on Systems Theory and Scientific Computation (ISTASC'08), Rhodes (Rodos) Island, Greece, August 20-22, 2008, pp.334-335, ISBN: 978-960-8764-83-1, ISSN: 1790-3865, Published by WSEAS Press (www.wseas.org, www.worldses.org/indexes)
[11] **Ioan Susnea**, Adrian Filipescu, Adriana Serbencu, A. Radaschin, *Virtual Pheromones to Control Mobile Robots* . A Neural Network Approach, Proceedings of the IEEE International Conference on Automation and Logistics, Shenyang, China August 2009, ISBN: 978-1-4244-4795-4/09, 2009 IEEE, CD-ROM Proceedings IEEE, Catalog #: CFP09CAL ISBN: 978-1-4244-4795-4, August 5 - 7, 2009, Shenyang, China
[12] Adriana Serbencu, Daniela Cristina Cernega, Adrian Emanoil Serbencu, **Ioan Susnea**, *Path Following Problem for PatrolBot Solved with Fuzzy Control* , Proceedings of the IEEE, International Conference on Automation and Logistics, Shenyang, China August 2009, ISBN: 978-1-4244-4795-4/09, 2009 IEEE, Catalog #: CFP09CAL ISBN: 978-1-4244-4795-4, August 5 - 7, 2009, Shenyang, China
[13] **Susnea I**. Filipescu A, Minzu V, Vasiliu G, Virtual Pheromones and Neural Networks based Wheeled Mobile Robot Control, Recent Advances on Systems, Proceedings of thr 13 International Conference on Systems WSEAS CSCC Multiconference, ISBN 978-960-474-097-0, ISSN: 1790-2769 Rodos, Greece , JULY 22-24, 2009, pp 511-516 Mathematics and Computers in Science Engineering, Aseries of Reference Books and Textbook Published by WSEAS Press
[14] **Susnea I**. Filipescu A, Minzu V, Vasiliu G, Virtual Pheromones and Neural Networks based Wheeled Mobile Robot Control, Recent Advances on Systems, Proceedings of thr 13 International Conference on Systems WSEAS CSCC Multiconference, ISBN 978-960-474-097-0, ISSN: 1790-2769 Rodos, Greece , JULY 22-24, 2009, pp 511-516 Mathematics and Computers in Science Engineering, A series of Reference Books and Textbook Published by WSEAS Press
[15] **Susnea I**, Filipescu Adrian, Filipescu Adriana, Coman G., Wheeled Mobile Robot Control Using Virtual Pheromones, Petroleum-Gas University of Ploiesti Bulletin, Tehnical Series, Vol LXI, no.3/2009, ISSN 1224-8495, pp. 117-126.
[16] **Susnea I**. Vasiliu G, Filipescu A, Coman G., Radaschin A., Real-Time Control of Autonomous Mobile Robots Using Virtual Pheromones, Proceedings of the 7th Asian Control Conference, Hong Kong, China, August 27-29, 2009, IEEE Catalog Number CFP09832, ISBN:978-89-956056-9-1, pp.1450-1455.
[17] Filipescu Adrian, **Susnea I**, Filipescu Adriana, Stamatescu G.,Control of Mobile platforms as Robotic Assistants for Elderly, Proceedings of the 7th Asian Control Conference, Hong Kong, China, August 27-29, 2009, IEEE Catalog Number CFP09832, ISBN:978-89-956056-9-1, pp.1456-1461.
[18] Filipescu Adrian, **Susnea I**, Filipescu Silviu, Stamatescu G., Wheeled Mobile Robot Control Using Virtual Pheromones and Neural Networks, Proceedings of 2009 IEEE International Conference on Control and Automation Christchurch, New Zealand, December 9-11, 2009, IEEE Catalog Number CFP09537, ISBN: 978-1-4244-4707-7, pp: 157-162, Library of Congress:2009904841.
[19] Filipescu Adrian, **Susnea I**, Filipescu Adriana, Stamatescu G., Distributed System of Mobile Platform Obstacle Avoidance and Control as Robotic Assistant for Disabled and Elderly, Proceedings of 2009 IEEE International Conference on Control and Automation Christchurch, New Zealand, December 9-11, 2009, IEEE Catalog Number CFP09537, ISBN: 978-1-4244-4707-7, pp: 1886-1891, Library of Congress:2009904841.
[20] **Ioan Susnea**, Viorel Minzu, Grigore Vasiliu Simple, Real-time Obstacle Avoidance Algorithm for Mobile Robots, Proceedings of the 8'th WSEAS International Conference on Computational Intelligence, Man-Machine Systems and Cybernetics, CIMMACS'09, Tenerife,, Dec. 14-16, 2009, pp:24-30, ISBN: 978-960-474-144-1, ISSN: 1790-5117

VII Alte activitati stiintifice/semne de recunoastere internationala

- Review Springer Verlag
- Reviewer Bentham Science Publishers Ltd
- Reviewer IEEE

- Reviewer WSEAS http://www.worldses.org/review/2008reviewers.htm
http://www.worldses.org/review/2009reviewers.htm
- Nominalizat pentru includerea in volumul *Who's Who in the world 2010*, publicat de Marquis Whos Who, www.marquiswhoswho.com

13. Limbi straine cunoscute: Engleza, Franceza (scris/vorbit)

14. Alte competente: Proiectare hardware si software embedded systems, proiectare cablaje imprimate, programare ASM, C, redactare documentatie tehnica, abilitati de trainer, cunostinte solide in domeniul automatizarilor industriale

15. Alte mentiuni: Proiecte semnificative in ultimii ani:

- 2007-2008 Cercetator in cadrul proiectului ID_641 din programul IDEI "Conducerea adaptiva sliding mode aplicata la manipulatoare robotice si roboti mobili"
- Proiectare si realizare sistem de monitorizare si control distribuit al temperaturii la cuptoarele de tratament termic la Mittal Steel Galati
- Proiectare si realizare remote terminal unit (RTU) cu comunicatie GSM pentru monitorizarea si controlul releelor POLARR (Electrica Galati)
- Proiectare si realizare embedded PLC cu intrari/iesiri distribuite
- Proiectare si realizare GPRS gateway pentru dispozitive cu microcontroller
- Proiectare si realizare positioner cu motor asincron reversibil split-phase pentru valve de tip "butterfly"
- Proiectare si realizare fuzzy controller pentru sincronizarea turatiei la motoarele de curent continuu, la linia de zincare (Laminorul Galati)
- Proiectare si realizare automat programabil dedicat pentru conducerea instalatiei tehnologice de rafinare la fabrica de zahar Carei