Universitatea “Politehnica” din Bucureşti

Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei

Proiect de diplomă

Dezvoltarea unui sistem stabil de echilibrare și control pentru roboți

prezentat ca cerinţă parţială pentru obţinerea titlului de

Inginer în domeniul *Electronică şi Telecomunicaţii*

programul de studii de licenţă *Reţele şi Software de Telecomunicaţii*

Conducător ştiinţific | Absolvent

Prof.Univ.Dr.Ing Corneliu BURILEANU | *Valentin Gabriel BADEA*

2016

Facultatea de Electronică,
Telecomunicații și Tehnologia
Informației

Anexa 1

Universitatea "Politehnica" din Bucureşti
Facultatea de Electronică, Telecomunicaţii şi Tehnologia Informaţiei
Departamentul Telecomunicaţii

Aprobat Director de Departament
Prof. Dr. Ing. Silviu CIOCHINA

TEMA PROIECTULUI DE DIPLOMĂ
a studentului BADEA GH. VALENTIN GABRIEL - 441D

1. Titlul temei:**Dezvoltarea unui sistem stabil de echilibrare şi control pentru roboţi**

2. Contribuţia practică, originală a studentului va consta în (*în afara* părţii de documentare):

   i. Dezvoltarea unui algoritm de control şi echilibrare pentru un robot cu 2 roţi, care poate fi folosit ca mijloc de transport.
   ii. Asamblarea şi optimizarea pentru obţinerea echilibrului pe verticală a robotului.
   iii. Proiectarea unui circuit imprimat (PCB) bazat pe un microcontroller ARM Infineon, având funcţii de comunicare prin protocol Bluetooth 4.0 Low-Energy, care va controla 2 motoare DC în funcţie de informaţia primită de la senzorii MEMS (giroscop + accelerometru).
   iv. Dezvoltarea unui algoritm software în limbaj C/C++ pentru funcţiile de autoechilibrare iniţială, autoechilibrare la intervenţia externă asupra sistemului şi control Bluetooth al mişcării robotului printr-un dispozitiv mobil compatibil.
   v. Se porneşte de la platforma ARM Infineon şi se doreşte integrarea tuturor sistemelor şi funcţiilor prezentate mai sus pentru a creea robotul.

3. Proiectul se bazează pe cunoştinţe dobîndite în principal la următoarele 3 discipline:
   Arhitectura Microprocesoarelor, Microcontrollere, Proiect 1

4. Proprietatea intelectuală asupra proiectului aparţine UPB şi studentului.

5. Locul de desfăşurare a activităţii: UPB şi studentul.

6. Realizarea practică rămâne în proprietatea studentului.

7. Data eliberării temei: 11.12.2015

CONDUCĂTOR LUCRARE	STUDENT
Prof. Dr. Ing Corneliu Burileanu	Valentin Gabriel Badea

Valentin Badea

Declarație de onestitate academică

Prin prezenta declar că lucrarea cu titlul "*Dezvoltarea unui sistem stabil de echilibrare și control pentru roboți*", prezentată în cadrul Facultății de Electronică, Telecomunicații și Tehnologia Informației a Universității "Politehnica" din București ca cerință parțială pentru obținerea titlului de *Inginer* în domeniul *Electronică și Telecomunicații*, programul de studii *Rețele și Software de Telecomunicații* este scrisă de mine și nu a mai fost prezentată niciodată la o facultate sau instituție de învățămînt superior din țară sau străinătate.

Declar că toate sursele utilizate, inclusiv cele de pe Internet, sunt indicate în lucrare, ca referințe bibliografice. Fragmentele de text din alte surse, reproduse exact, chiar și în traducere proprie din altă limbă, sunt scrise între ghilimele și fac referință la sursă. Reformularea în cuvinte proprii a textelor scrise de către alți autori face referință la sursă. Înțeleg că plagiatul constituie infracțiune și se sancționează conform legilor în vigoare.

Declar că toate rezultatele simulărilor, experimentelor și măsurătorilor pe care le prezint ca fiind făcute de mine, precum și metodele prin care au fost obținute, sunt reale și provin din respectivele simulări, experimente și măsurători. Înțeleg că falsificarea datelor și rezultatelor constituie fraudă și se sancționează conform regulamentelor în vigoare.

București,

Absolvent
Valentin Gabriel BADEA

(semnătura în original)

Cuprins Figuri

Cuprins Tabele

Lista abrevierilor

α – *Accelerația unghiulară a axului (rad/s²)*
a – *Accelerație (m/s)*
B – *Matricea de controlul de intrare*
θ, θ_p – *Unghiul de înclinare al robotului(rad)*
$\dot{\theta}_w$ – *Viteza unghiulară a roții (rad/s)*
$\dot{\theta}_b$ – *Bias (eroarea sistematică)* [1]
Kp – *Constanta proporțională*
Ki – *Constanta integrală*
Kd – *Constanta derivativă*
k_f – *Constanta de frecare (Nms/rad)*
k_m – *Constanta de cuplu (Nm/A)*
k_e – *Constanta de tensiune inversă (Vs/rad)*
K_k – *Câștigul filtrului Kalman*
CL, CR – *Cuplul aplicat de la motor la roți*
τ_m – *Cuplul motorului (Nm)*
τ_e – *Cuplul aplicat (Nm)*
D – *Factorul de umplere (%)*
e(t) – *Funcția de eroare*
F – *Matricea modelului de tranziție al stărilor*
g – *Accelerația gravitațională (N/s²)*
G_p – *Matricea de rotație*
HL, HR, PL, PR – *Forțele de reacție dintre șasiu și roți*
HfL, HfR – *Forțele de frecare ale roților*
H – *Matricea modelului de observație*
I – *Curentul prin armături (A)*
I_w – *Momentul de inerție al roților*
I_p – *Momentul de inerție al șasiului*
IR – *Inerția rotorului (kg/m²)*
l – *Distanța dintre centrul de greutate al robotului și axul roților*
L – *Inductanța rotorului (H)*
M_w – *Masa ambelor roți ale robotului*
M_p – *Masa șasiului robotului*
P – *Matricea de covarianță a erorii*
$P_{k|k-1}$ – *Matricea de covarianță a erorii a priori*
$P_{k-1|k-1}$ – *Matricea de covarianță a erorii precedente*
R – *Rezistența terminalelor (Ohm)*
Q_k – *Matricea de covarianță a zgomotului de proces*
S_k – *Matricea de covarianță a inovației*
V_a – *Tensiunea aplicată (V)*
V_e – *Tensiunea inversă (V)*
ϑ_k – *Zgomotul măsurătorii*
$\hat{x}_{k|k-1}$ – *Starea a priori*
$\hat{x}_{k|k}$ – *Starea a posteriori*
$\hat{x}_{k-1|k-1}$ – *Starea precedentă*
$\tilde{y}_k$ – *Inovația sau reziduu*
ω – *Viteza unghiulară a axului (rad/s)*
u_k – *Intrare de control*
ω_k – *Zgomotul de proces*
x_k – *Starea la momentul k*
ZMP – *Punctul de moment zero*
z_k – *Măsurătoare discretă*

Introducere

Încă din Antichitate, oamenii au visat la obiecte care să le facă viața mai ușoară, să muncească în locul lor și care să se supună întru totul comenzilor. Așa s-a născut dorința de a crea și de a imagina soluții care să îi ajute. Din dorința confortului și nu numai, oamenii au început încet-încet să contureze primii roboți. În secolul al XV-lea, Leonardo da Vinci deschide calea roboticii schițând primul umanoid [2]. Între secolele XVIII-XIX, revoluția industrială dezvoltă nenumărate automatizări care ajută la evoluția omului și a tehnologiilor. Primul robot autonom apare la începutul secolului al XX-lea în Japonia [3]. În anii 1950 apar primele idei de a crea roboți pentru a fi trimiși pe Lună, iar până în 1969 aceștia îl însoțesc pe Neil Armstrong la prima aselenizare din istoria omenirii. După anii 1990 piața roboților crește exponențial, ajungând în prezent la o multitudine de modele, care deservesc diferite scopuri.

Roboții cu 2 roți au apărut din nevoia omului de a reduce cât mai mult componentele și de a optimiza producția, obținând astfel aceleași performanțe ca un robot cu 4 roți. Mai mult, aceștia pot permite manevrabilitate suplimentară, navigare ușoară pe diferite terenuri, schimbarea direcției cu un unghi redus și chiar rularea pe trotuare și peste borduri. Aceste capabilități au atins potențialul de a rezolva o serie de provocări în industrie și societate. De exemplu, un scaun cu rotile motorizat care utilizează această tehnologie ar da operatorului o mai mare manevrabilitate și acces mai ușor la locurile care pot prezenta dificultate crescută pentru oamenii cu afecțiuni motorii. Se pot produce cărucioare folosind această tehnologie care pot permite oamenilor să călătorească pe distanțe scurte, într-o zonă mică sau în fabrici, spre deosebire de utilizarea unor autovehicule, care sunt mai poluante și mai greu de administrat. De asemenea, un robot cu 2 roți poate fi privit ca o platformă care este capabilă să transporte diferite obiecte nu numai omul.

În funcție de dimensiunea și caracteristicile sale electrice/mecanice, robotul poate deservi un larg spectru de aplicații de la cele industriale de stabilizare până la folosirea lui ca mijloc de transport individual. În lucrarea de față, dimensiunile robotului sunt mici raportate la necesitățile comerciale, dar poate fi privit ca un punct de plecare către aplicații mult mai complicate și ca o machetă de observație pentru fenomenele fizice apărute.

Un sistem disponibil comercial, “SEGWAY HT” a fost inventat de Dean Kamen [4], care deține mai mult de 150 de brevete legate de dispozitive medicale, sisteme de control ale climei, precum și un design de elicopter.”SEGWAY HT” este capabil să echilibreze un om care stă pe o platformă în timp ce traversează terenul cu el. Această inovație utilizează, pentru a menține poziția verticală, cinci giroscoape și o colecție de alți senzori de înclinație. Doar trei giroscoape sunt necesare pentru întreg sistemul, senzorii suplimentari au fost incluși ca o măsură de siguranță. Unicitatea acestor produse a atras un interes din partea fanilor de roboți.

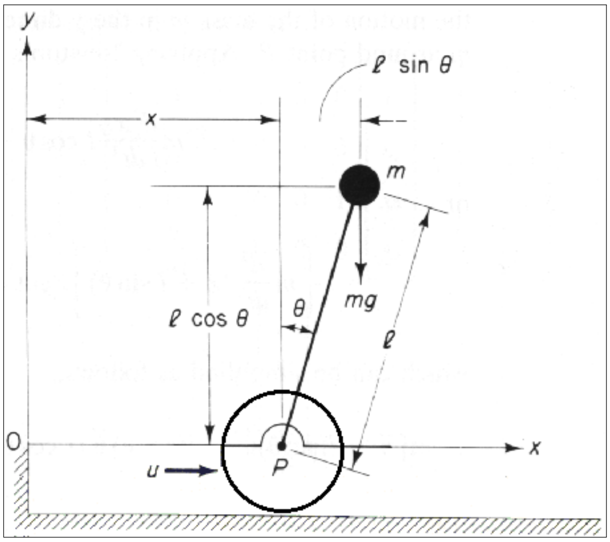


Fig. 0.1 Forțele ce acționează asupra unui pendul inversat cu roți

La baza ideii de echilibru al acestui robot se află conceptul pendulului inversat [5](Fig 0.1). Problema pendulului inversat este frecvent întâlnită în domeniul ingineriei care se ocupă cu controlul sistemelor. Unicitatea și multitudinea de tehnologii derivate din acest sistem instabil a captat atenția multor cercetători și pasionați de tehnologie din întreaga lume. În ultimii ani, cercetătorii au aplicat ideea unui model de pendul inversat mobil în diverse probleme, cum ar fi proiectarea mișcării de mers pe jos pentru roboții umanoizi, scaune cu roți robotizate și sisteme de transport personale. Teoria pendulului inversat care este folosită în acest proiect poate fi descrisă pe scurt astfel:

Un pendul inversat va ieși din poziția de echilibru și va imprima sistemului o forță opusă căderii. Pentru echilibrare este nevoie de o forță egală în modul, pe aceeași direcție, dar de sens contrar forței imprimate de pendul pentru a-l redresa.

Lucrarea "Comportamentul unui pendul inversat pe roți pentru transportarea unui obiect”, [6] arată interacțiunea forțelor între diverse obiecte și robot prin luarea în considerare a efectelor de stabilitate datorate acestor forțe. Această cercetare subliniază posibilitatea transportului prin cooperare între doi roboți similari și între un robot și un om. Creșterea rapidă a populației în vârstă în țări precum Japonia a determinat cercetătorii să dezvolte scaune cu rotile robotizate, ajutând infirmii să se deplaseze. Sistemul de control al unui pendul inversat se aplică atunci când sunt necesare manevre de urcare a bordurilor sau asigurarea unui control mai bun în curbe.

La un nivel mai înalt, T. Sugihara [7] a folosit modelul mișcării de deplasare uman ca pendul inversat în proiectarea unei metode de generare a rutinei de mers pentru un robot umanoid. Aceasta controlează centrul de greutate prin manipulare indirectă a punctului de moment zero (ZMP). Răspunsul în timp real al metodei oferă roboților umanoizi o mobilitate ridicată.

1. Blocurile funcționale ale sistemului

Sistemul este alcătuit din mai multe blocuri funcționale, fiecare având un aport important la forma finală. Puterea de calcul necesară pentru a funcționa la o frecvență foarte ridicată este justificată datorită caracteristicii sistemului de lucru în timp real. Astfel, pentru a putea contracara perturbările din mediu și pentru a stabiliza robotul, este nevoie de un microprocesor care să efectueze calculele necesare și să comande motoarele într-un timp cât mai scurt. S-a ales în acest proiect microcontrolerul Infineon XMC4700.

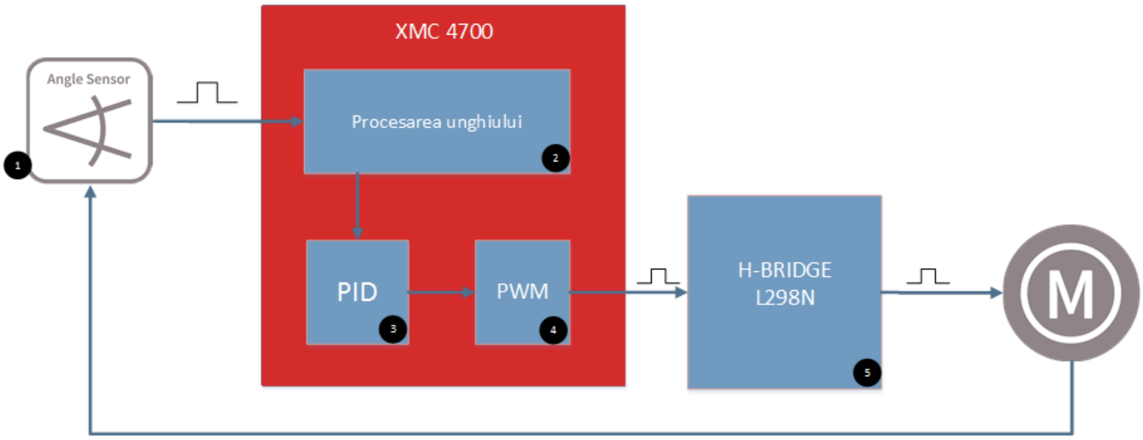


Fig. 1.1 Schema bloc a sistemului

1.1. Infineon XMC4700 Relax Kit

XMC4700 integrează un procesor RISC ARM Cortex M4, 32 bit, funcționând la o frecvență de 144Mhz, are memoria flash de 2048kb, memoria RAM de 352kB și nu are memorie cache [8] . Este un model construit pe baza arhitecturii Harvard. Harta memoriei suportată de acest procesor este de 4GB. Capabil să ruleze programe de o complexitate medie spre ridicată, poate face față cu succes operațiilor din mediul industrial, automatizărilor, aplicațiilor de putere, cât și uzului general. În cazul de față, va fi folosit la o aplicație experimentală cu motoare DC și un senzor. Cea mai complexă operație realizată de acest microcontroler este calculul unghiului robotului în timp real. La acest rezultat, se aplică un filtru Kalman, deoarece datele oferite de senzor au zgomot, fiind necesară o filtrare pentru a le putea folosi eficient.

Limbajul de programare folosit în proiect este C. Dimensiunea codului ajunge la 45kB în modul Debug și scade la 32kB pentru modul Release. Diferența apare datorită informațiilor în plus necesare în modul Debug (breaking points, variabile, regiștrii). Întregul soft rulează o iterație de control în 10 ms, fiind optimizat cât se poate de mult.

Ca mediu de programare a fost folosit DAVEv4 [9]. Acesta prezintă o utilitate crescută, deoarece are module software de bază care fac accesul la funcționalitatea perifericelor microcontrolerului pentru a putea dezvolta aplicații mai complexe într-un mod facil. Prin urmare, deși rigiditatea modului de programare este un dezavantaj, timpul de dezvoltare scade.

1.2. Senzorul MEMS (MPU 9150)

MEMS (eng. Microelectromechanical systems) este o tehnologie care stă la baza structurilor electronice de mărimi reduse de ordinul nanometrilor. În cazul actual, regăsim acest concept în senzorul pentru detectarea unghiului robotului. Principiul de bază al funcționării accelerometrului MEMS este deplasarea unei mase mici (de referință) gravată în suprafața de siliciu a circuitului integrat și suspendată de mici grinzi. În concordanță cu a doua lege a lui Newton ($F = ma$), când o accelerație este aplicată dispozitivului forța care apare dislocă masa de referință. Grinzile acționează ca un arc, iar lichidul (sau aer) prins în interiorul IC acționează ca un amortizor, rezultând într-un sistem fizic de ordin 2. Aceasta este sursa de informație oferită de accelerometru care are o lățime de bandă operațională limitată și răspunsul în frecvență neuniform [10].

Valoarea de ieșire a accelerometrului este un scalar corespunzător cu mărimea vectorului de accelerație. Cea mai comună formă de accelerare și cea la care suntem expuși constant, este accelerația rezultată din atracția gravitațională a Pământului. Aceasta este o valoare comună, de referință, la care se măsoară toate celelalte accelerații **g.**

Un senzor comun de inerție este accelerometrul, un senzor dinamic capabil de o gamă largă de detectare. Accelerometrele disponibile pot măsura accelerația în una, două sau trei axe ortogonale. Ele sunt utilizate în mod obișnuit într-unul din cele trei moduri:

- Pentru măsurarea vitezei și a poziției;
- Ca un senzor de înclinare sau orientare în 2 sau 3 dimensiuni, având ca punct de referință accelerația gravitațională (1 g = 9.8m /s2);
- Ca un senzor pentru vibrații sau șocuri.

Senzorul InvenSense MPU 9150 are încorporat un giroscop și un accelerometru. Acesta a fost proiectat pentru aplicații de putere joasă, fiind ideal pentru telefoane și aparate mobile, având în general un consum foarte mic.

Senzorul comunică cu microcontrolerul prin protocolul I2C. Datele de la accelerometru și giroscop sunt salvate în 12 regiștrii ai senzorului. Pentru a accesa aceste date, microcontrolerul citește în rafală câte 6 regiștrii de 2 ori.

Dorindu-se o eficiență sporită, se folosește pinul INT de la senzor. Acesta va fi HIGH în momentul în care senzorul actualizează regiștrii cu valori noi. Prin această metodă se poate implementa o întrerupere la microcontroler, citind datele senzorului numai când acestea au fost reactualizate, evitându-se astfel operațiile redundante.

Frecvența de eșantionare a accelerometrului este de 1KHz, iar a giroscopului de 8KHz. Ca soluție de mijloc, microcontrolerul va citi datele de la senzor cu o frecvență de 1KHz. Ca alternativă se poate folosi un timer intern pentru a rula funcția de citire a senzorului la un interval prestabilit, dar nu mai mic de 1ms. Protocolul I2C prezintă un avantaj față de versiunea de comunicare analog deoarece are eficiența crescută și erorile care pot apărea sunt nesemnificative. Prezentarea protocolului se poate găsi în capitolul 2.3.

1.3. Determinarea unghiului

Informația citită de la senzorul MEMS prin I2C reprezintă coordonatele pe axa X, Y, Z a accelerometrului și a giroscopului [11]. Având în vedere accelerația gravitațională **g** și accelerația ***a*** pe care accelerometrul o măsoară, putem defini matricea de rotație a robotului în raport cu gravitația Pământului astfel:

$$G_p = \begin{bmatrix} G_{px} \\ G_{py} \\ G_{pz} \end{bmatrix} = R(g - a) \tag{1.1}$$

Pentru a rezolva această ecuație, trebuie să presupunem că accelerometrul are accelerația inițială nulă $\boldsymbol{a} = 0$ și că forța gravitațională a Pământului este perpendiculară pe axa Z, rezultând:

$$G_p = R * g = R \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{1.2}$$

Plecând de la această ecuație, putem scrie matricele de rotație pentru axa transversală, longitudinală şi verticală:

$$R_x(\phi) = \begin{pmatrix} 1 & 0 & 0 \\ 0 & \cos\phi & \sin\phi \\ 0 & -\sin\phi & \cos\phi \end{pmatrix} \tag{1.3}$$

$$R_y(\theta) = \begin{pmatrix} \cos\theta & 0 & -\sin\theta \\ 0 & 1 & 0 \\ \sin\theta & 0 & \cos\theta \end{pmatrix} \tag{1.4}$$

$$R_z(\psi) = \begin{pmatrix} \cos\psi & \sin\psi & 0 \\ -\sin\psi & \cos\psi & 0 \\ 0 & 0 & 1 \end{pmatrix} \tag{1.5}$$

Pentru a determina unghiul avem nevoie de ecuația R_{xyz}, astfel:

$$\boldsymbol{R}_{xyz} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} = \boldsymbol{R}_x(\phi)\boldsymbol{R}_y(\theta)\boldsymbol{R}_z(\psi) \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix} \tag{1.6}$$

$$= \begin{pmatrix} \cos\theta\cos\psi & \cos\theta\sin\psi & -\sin\theta \\ \cos\psi\sin\theta\sin\phi - \cos\phi\sin\psi & \cos\phi\cos\psi + \sin\theta\sin\phi\sin\psi & \cos\theta\cos\phi \\ \cos\phi\cos\psi\sin\theta + \sin\phi\sin\psi & \cos\phi\sin\theta\sin\psi - \cos\psi\sin\phi & \cos\theta\cos\phi \end{pmatrix} \begin{pmatrix} 0 \\ 0 \\ 1 \end{pmatrix}$$

$$= \begin{pmatrix} -\sin\theta \\ \cos\theta\sin\phi \\ \cos\theta\cos\phi \end{pmatrix} \tag{1.7}$$

Putem rescrie această relație raportând axa transversală și cea longitudinală la valoarea normată a accelerometrului:

$$\frac{G_p}{\|G_p\|} = \begin{pmatrix} -\sin\theta \\ \cos\theta\sin\phi \\ \cos\theta\cos\phi \end{pmatrix} \Rightarrow \frac{1}{\sqrt{{G_{px}}^2+{G_{py}}^2+{G_{pz}}^2}} \begin{bmatrix} G_{px} \\ G_{py} \\ G_{pz} \end{bmatrix} = \begin{pmatrix} -\sin\theta \\ \cos\theta\sin\phi \\ \cos\theta\cos\phi \end{pmatrix} \tag{1.8}$$

Dacă rezolvăm ecuația (1.8) pentru axa transversală și cea longitudinală obținem:

$$\tan\phi_{xyz} = \frac{G_{py}}{G_{pz}} \tag{1.9}$$

$$\tan\theta_{xyz} = \frac{-G_{px}}{G_{py}\sin\phi + G_{pz}\cos\phi} = \frac{-G_{px}}{\sqrt{{G_{py}}^2+{G_{pz}}^2}} \tag{1.10}$$

unde G_{px} G_{py} și G_{pz} sunt valorile citite de la accelerometru.

Pentru a determina în final valoarea pe axa transversală, cea care reprezintă unghiul de înclinare al robotului, vom apela la funcțiile din librăria MATH.h (atanf, atan2f).

float roll = atan2f(accY, accZ) * RAD_TO_DEG;

Unghiul calculat va fi parametrul de intrare al algoritmului PID, după ce va trece prin filtrul Kalman.

1.4. Proiectarea filtrului Kalman discretizat

Filtrul Kalman este alcătuit dintr-un set de ecuații matematice pentru algoritmul de calcul optim recursiv care asigură soluția prin metoda celor mai mici pătrate [12]. Acestea încorporează toate informațiile care pot fi furnizate și procesează toate măsurătorile disponibile, indiferent de precizia lor, pentru a estima valoarea curentă a variabilelor de interes. Sunt luate în considerare cunoștințele anterioare ale dinamicii de sistem și de măsurare, descrierea statistică a zgomotelor din sistem, erori de măsurare, imprecizia în dinamica modelului, precum și orice informații disponibile cu privire la starea inițială a variabilei de interes.

Dacă modelul sistemului de interes este liniar și valorile zgomotului nu sunt corelate în timp, atunci filtrul Kalman este optim din orice punct de vedere. Filtrul Kalman nu necesită ca toate datele anterioare să fie păstrate în memorie și reprocesate de fiecare dată când se ia o nouă măsurătoare. Prin acest comportament, filtrul Kalman poate fi implementat în software pe orice microcontroler.

A. Ecuația de stare

Filtrul Kalman poate fi aplicat pentru a estima stările sistemului atunci când acesta este modelat în mod adecvat sub forma unei ecuații diferențiale stocastice liniare.

Ecuația (1.1) reprezintă ecuația de stare a sistemului la momentul k împreună cu zgomot alb ω_k.

$$x_k = \boldsymbol{F} x_{k-1} + \boldsymbol{B} u_k + \omega_k \tag{1.11}$$

Particularizând la sistemul studiat, x_k este matricea stărilor și este egală cu :

$$x_k = \begin{bmatrix} \theta \\ \dot{\theta}_b \end{bmatrix}_k \tag{1.12}$$

unde θ este unghiul de la ieșirea filtrului și $\dot{\theta}_b$ este eroarea sistematică a măsuratorilor de la giroscop și accelerometru. Putem deduce valoarea reală scăzând eroarea din valoarea măsurată.

Matricea F este modelul de tranziție al stărilor care este aplicat stării anterioare x_{k-1}

$$F = \begin{bmatrix} 1 & -\Delta t \\ 0 & 1 \end{bmatrix} \tag{1.13}$$

u_k reprezintă controlul de intrare, în acest caz, este măsurarea giroscopului în grade pe secundă (°/s) la momentul k. Vom rescrie de fapt, ecuația de stare ca:

$$x_k = \boldsymbol{F} x_{k-1} + \boldsymbol{B} \dot{\theta}_k + \omega_k \tag{1.14}$$

unde matricea **B** este modelul controlului de intrare, care este definit ca:

$$B = \begin{bmatrix} \Delta t \\ 0 \end{bmatrix} \tag{1.15}$$

Acest lucru are sens, deoarece se va obține unghiul θ când se multiplică rata $\dot{\theta}_k$ cu $\Delta t = t_k - t_{k-1}$. Pentru că nu se poate calcula eroarea sistematică direct, bazată pe rata $\dot{\theta}_k$, vom stabili valoarea 0 în partea de jos a matricei.

ω_k reprezintă zgomotul de tip gaussian de medie 0 și covarianță Q la momentul k:

$$\omega_k \sim N(0, Q_k) \tag{1.16}$$

Q_k este matricea de covarianță a zgomotului procesului și în acest caz matricea de covarianță a stării estimate a accelerometrului și a erorii acestuia. Se va lua în considerare estimarea erorii sistematice și valoarea accelerometrului ca fiind independente, rezultând că Q_k reprezintă chiar valorile estimate ale varianței accelerometrului și a erorii sistematice (bias).

$$Q_k = \begin{bmatrix} Q_\theta & 0 \\ 0 & Q_{\dot{\theta}_b} \end{bmatrix} \Delta t \qquad (1.17)$$

După cum se poate vedea, matricea de covarianță Q_k depinde de momentul curent k, astfel încât variația accelerometrului Q_θ și variația erorii $Q_{\dot{\theta}_b}$ sunt multiplicate cu variația timpului Δt.
Acest lucru are sens deoarece zgomotul de proces va fi mai mare într-un interval de timp mai lung de la ultimul moment al actualizării de stare.

Măsurătorile sistemului sunt efectuate la intervale de timp discrete, aceste fiind incluse în vectorul de măsurare putând fi modelate prin relația:

$$z_k = \boldsymbol{H} x_k + \vartheta_k \qquad (1.18)$$

După cum se observă, măsurarea z_k este dată de starea actuală x_k înmulțită cu matricea **H** plus ϑ_k zgomotul de măsurare.

Ecuația (1.18) precizează că măsurătorile depind de starea sistemului și sunt legate de matricea de măsurare, cu un adaos de zgomot în măsurători. Matricea **H** face legătura între valoarea reală și cea măsurată. Valoarea reală nu poate fi determinată, deoarece există doar măsurătorile din partea accelerometrului și nu pot fi comparate cu valori din alte sisteme de măsură. Măsurătorile sunt adesea obținute la intervale egale de timp, dar acest lucru nu este obligatoriu. Matricea H este egală cu:

$$H = [1 \quad 0] \qquad (1.19)$$

B. Zgomotul sistemului și al măsurătorilor

Zgomotul de măsurare este gaussian și distribuit de asemenea cu medie 0 și covarianță R:

$$\vartheta_k \sim N(0, R) \qquad (1.20)$$

R este o matrice definită pozitiv, ceea ce înseamnă că toate componentele măsurătorilor sunt corupte cu zgomot și nu există nicio combinație liniară a acestor componente care poate fi lipsită de zgomot. Având în vedere că distribuțiile ω și ϑ sunt presupuse ca fiind gaussiene, acest lucru înseamnă că sunt necorelate între ele. Dar cum R nu este o matrice de măsurare a zgomotului, ea este egală doar cu variația măsurătorii, deoarece covarianța aceleiași variabile este egala cu variația ei. Putem defini R astfel:

$$R = E[\vartheta_k \quad \vartheta_k^T] = var(\vartheta_k) \qquad (1.21)$$

Vom asuma că zgomotul de măsurare este acelaşi și independent de timp:

$$var(\vartheta_k) = var(\vartheta) \qquad (1.22)$$

Notă: Dacă variația zgomotului de măsurare este prea mare, atunci filtrul va avea un timp de răspuns mare și nu va acorda o pondere ridicată noilor măsurători, iar dacă este prea mică este posibil ca valoare măsurată să fie zgomotoasă și ponderea de încredere a valorilor să fie ridicată.

C. Condiții inițiale

Ecuația diferențială de stare (1.1) are prima iterație cu x_0 și pentru orice operațiune specifică a sistemului, starea inițială are impact în calcul. Cu toate acestea, deoarece această valoare nu poate fi cunoscută cu precizie dinainte, va fi modelată ca un vector aleator care este distribuit normal. Astfel, vom descrie x_0 cu media $\hat{x}_0$ și covarianța P_0.

$$E\{x_0\} = \hat{x}_0 \qquad (1.23)$$

$$E\{[x_0 - \hat{x}_0 \quad (x_0 - \hat{x}_0)^T]\} = P_0 \qquad (1.24)$$

P_0 este o matrice simetrică ce oferă valoarea prezisă a diferenței dintre starea reală și starea estimată. Elementele de pe diagonala acestei matrice asigură dispersia fiecărei variabile de stare în raport cu valoarea sa reală.

D. Ecuațiile de filtrare

Filtrul Kalman estimează un proces prin utilizarea unei forme de control cu reacție: filtrul estimează starea procesului la un anumit moment de timp și apoi obține răspunsul în forma măsurărilor (zgomotoase). Astfel, ecuațiile filtrului Kalman se împart în două grupe: ecuații de actualizare în timp și ecuații de actualizare a măsurării. Ecuațiile de actualizare în timp sunt responsabile de determinarea estimaților stării curente și a matricei de covarianță a erorii pentru a obține estimații a priori pentru următorul moment de timp. Ecuațiile de actualizare a măsurării sunt responsabile de reacție – adică de încorporarea unei noi măsurări în estimatul a priori pentru a obține un estimat a posteriori îmbunătățit. Ecuațiile de actualizare în timp pot fi de asemenea gândite ca ecuații predictor, în timp ce ecuațiile de actualizare a măsurării pot fi gândite ca ecuații corector. Într-adevăr algoritmul de estimare final seamănă cu cel al unui algoritm de tip predictor-corector pentru rezolvarea problemelor numerice așa cum este arătat în figura 1.2.

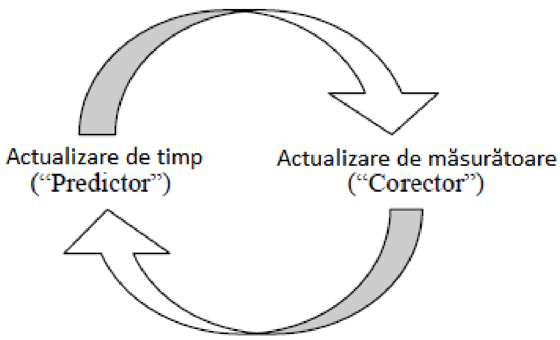


Fig. 1.2 Schema algoritmului predictor-corector

E. Predicția

Primele două ecuații vor încerca să prezică starea curentă și matricea de covarianță a erorii la momentul k. În primul rând, filtrul va încerca să estimeze starea actuală, bazată pe toate stările anterioare și pe măsurarea giroscopului:

$$\hat{x}_{k|k-1} = \boldsymbol{F}_{\hat{x}_{k-1|k-1}} + \boldsymbol{B}\dot{\theta}_k \tag{1.25}$$

Acesta este, de asemenea, motivul pentru care se numește o intrare de control, din moment ce am folosit un parametru de intrare suplimentar pentru a estima starea la momentul actual k, numită starea a priori $\hat{x}_{k|k-1}$. Se va încerca estimarea a priori a matricei de covarianță a erorii $P_{k|k-1}$ bazată pe matricea anterioară a covarianței erorii $P_{k-1|k-1}$, care este definită ca:

$$P_{k|k-1} = \boldsymbol{F}P_{k-1|k-1}\boldsymbol{F}^T + Q_k \tag{1.26}$$

Această matrice este utilizată pentru a estima cât de multă încredere se acordă valorilor curente ale stării estimate. Principiul ecuației de mai sus este, de fapt, destul de ușor de înțeles, așa cum este destul de evident, covarianța erorii va crește, deoarece ultima dată s-a actualizat estimarea stării, prin urmare, am multiplicat matricea de covarianță a erorii cu modelul de tranziție de stare F și transpusa sa $\boldsymbol{F}^T$, iar apoi se adaugă zgomotul procesului curent Q_k la momentul k. Matricea de covarianță a erorii este în cazul actual de formă 2 x 2 :

$$P = \begin{bmatrix} P_{00} & P_{01} \\ P_{10} & P_{11} \end{bmatrix} \tag{1.27}$$

F. Corecția

Diferența $z_k - \boldsymbol{H}\hat{x}_{k|k-1}$ în ecuația (1.28), este numită inovația măsurării sau reziduul. Reziduul reflectă diferența dintre măsurarea actuală z_k şi măsurarea prezisă $x_{k|k-1}$

$$\tilde{y}_k = z_k - \boldsymbol{H}\hat{x}_{k|k-1} \tag{1.28}$$

Se va calcula apoi covarianța reziduului:

$$S_k = HP_{k|k-1}H^T + R \tag{1.29}$$

S_k încearcă să prezică cât de credibilă este măsurătoare bazată pe matricea de covarianță a priori $P_{k|k-1}$ și pe matricea de covarianță a măsurătorilor $\boldsymbol{R}$. Cu cât este mai mare valoarea zgomotului de măsurare este mai mare și valoarea lui **S**, acest lucru înseamnă că nu avem încredere în măsurătorile de intrare atât de mult.

G. Câștigul filtrului

Matricea K_k este numită câştigul sau factorul de amestec, minimizând urma matricei de covarianță a erorii a posteriori.

$$K_k = P_{k|k-1} H^T S_k^{-1} \tag{1.30}$$

Se poate observa că dacă nu există credibilitate a reziduului foarte mare, credibilitatea inovației covarianței **S** va fi ridicată, iar dacă credibilitatea estimatului stării este mare atunci matricei de covarianță a erorii ***P*** va fi mică, prin urmare, câștigul Kalman va fi mic.

Dacă se aprofundează se poate observa că transpusa modelului de observare **H** este folosită pentru a mapa starea de covarianță a erorii matricei **P** în spațiu observat. Vom compara matricea de covarianță a erorii prin înmulțirea cu inversul covarianței de inovare **S**.

Acest lucru are sens deoarece vom folosi modelul de observare **H** pentru a extrage date din covarianța erorii de stare și se compară cu estimarea actuală a covarianței de inovare. În acest moment se poate corecta estimatul a posteriori:

$$\hat{x}_{k|k} = \hat{x}_{k|k-1} + \boldsymbol{K_k} \tilde{y}_k \tag{1.31}$$

Inovația poate fi pozitivă sau negativă. Pe scurt, ecuația poate fi înțeleasă că va corecta estimatul a priori $\hat{x}_{k|k-1}$, care a fost calculat folosind starea anterioară, și măsurarea giroscopului, cu măsurarea - în acest caz - a accelerometrului. Ultimul lucru care se va face este acela de a actualiza matricea de covarianță a erorii a posteriori:

$$P_{k|k} = (I - \boldsymbol{K_k H}) P_{k|k-1} \tag{1.32}$$

unde I este matricea de identitate și este egală cu $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

Filtrul face în principiu o autocorectare a matricei de covarianță a erorii în funcție de cât de mult s-a corectat estimarea. Acest lucru are sens, deoarece s-a corectat starea bazată pe urma matricei de covarianță a erorii a priori $P_{k|k-1}$, dar și covarianța inovării S_k.

Performanța filtrului poate varia foarte mult în cazul în care parametrii nu sunt reglați corect. Prin urmare, această secțiune își propune să ofere o idee despre caracteristicile parametrilor filtrului.

Parametrii reglabili sunt:

1. Matricea inițială de covarianță.
2. Vectorul estimatului de stare.
3. Matricea Q și vectorul ω aleator corespunzător.
4 Matricea R și vectorul ϑ aleator corespunzător.

H. Matricea covarianței erorii inițiale

Variația așteptată a erorii estimatului de stare pentru parametrul corespunzător este reprezentată de elementele diagonale ale matricei. Pentru ca filtrul să funcționeze corect, valorile din matricea de covarianță vor fi definite astfel încât diferența dintre starea inițială și estimatul stării inițiale sunt în intervalul care este admisibil conform matricei de covarianță.

Elementul diagonal al matricei de covarianță inițială trebuie să fie suficient de mare pentru a îndeplini cerința menționată mai sus. În cazul în care estimatul inițial al stării este destul de precis, atunci matricea de covarianță permite doar o mică eroare. Estimatul de stare va avea nevoie de mai mult timp să conveargă cât timp matricea de covarianță devine din ce în ce mai mare.

I. Estimatul inițial de stare

O estimare inițială a sistemului trebuie să fie disponibilă buclei de filtru care urmează să fie executată. Nu există nicio cerință generală a estimării inițiale pentru a fi corectă, cu condiția ca valorile inițiale ale matricei de covarianță să fie suficient de mari pentru ca filtrul să funcționeze corect.

J. Matricea Q

Matricea **Q** reprezintă covarianța vectorului eroare de sistem, ω. Se presupune că elementele ω sunt necorelate. Prin urmare, valoarea fiecărui element **Q** care nu se află pe diagonală este zero. Creșterea matricei Q ar indica fie zgomote puternice din cauza părții mecanice, fie incertitudine crescută în caracterul modelului în sine pentru a descrie adevărata dinamică cu precizie. Acest lucru va mări rata de creștere a elementelor $P(t)$ sau valorile proprii dintre timpii de măsurare și valorile lor la starea de echilibru. Ca rezultat, câștigul de filtrare va crește în general, iar măsurătorile sunt ponderate în mare măsură, acest lucru este rezonabil, dat fiind faptul că matricea **Q** dictează că ar trebui să se acorde mai puțină încredere în ieșirea filtrului.

K. Matricea R

Matricea **R** reprezintă covarianța matricei de eroare de măsurare, ϑ. Această matrice indică cât de mari sunt așteptate erorile de măsurare să fie. O matrice **R** cu valori mari ar indica faptul că măsurătorile sunt supuse unui zgomot mai puternic și așa ar trebui să aibă o pondere mai mică acordată de filtru. Filtrul ajunge în starea de echilibru rapid în cazul în care valorile proprii ale matricei Q sunt mari în comparație cu valorile proprii ale lui R (raportul Q / R este mare). Acest lucru se datorează incertitudinii mari implicate în propagarea de stare în comparație cu precizia măsurătorii, astfel încât noua estimare de stare depinde în mare măsură de noua măsurare și nu este strâns legată de estimările anterioare.

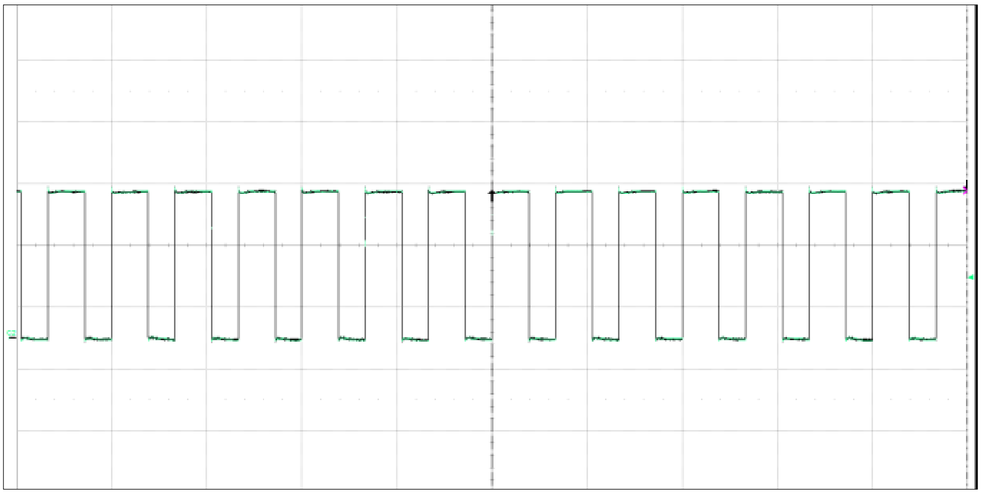
1.5. Modularea PWM

PWM (eng. Pulse Witdth Modulation) este o modulare care folosește lățimea pulsului unui semnal dreptunghiular pentru a transmite informația. Este o tehnică folosită pentru a varia în mod controlat tensiunea dată unui dispozitiv electronic. Această metodă schimbă foarte rapid tensiunea oferită dispozitivului respectiv din ON în OFF și invers. Perioada de timp corespunzătoare valorii ON dintr-un ciclu ON-OFF se numește factor de umplere (*duty cycle*) și reprezintă, în medie, ce tensiune va primi dispozitivul electronic. Așa se pot controla circuitele analogice din domeniul digital. De cele mai multe ori este folosit în controlul aplicațiilor de putere. Este caracterizat de factorul de umplere și frecvență [13].

PWM este folosit în cazul acesta pentru a seta viteza motorului. Eroarea corectată de la ieșirea blocului PID este factorul de umplere din blocul PWM. Rezultă că factorul de umplere variază în funcție de postura robotului, compensând astfel mișcările care îl scot din echilibru. În Figura 1.3 se pot observa semnale PWM cu factor de umplere de 60%. Se poate deduce foarte ușor formula factorului de umplere D:

$$D = \frac{Latimea\ pulsului\ ON}{Perioada\ semnalului * 100} = \frac{t_{ON}}{(t_{ON}+t_{OFF}) * 100} \tag{1.33}$$

Tensiunea medie care ajunge la dispozitiv este dată de relația: $D * V_{cc}$



Modularea folosește variația factorului de umplere a unei forme de undă dreptunghiulară generând la ieșire o tensiune analogică.

Fig. 1.3 Captură osciloscop D ~ 60%

Considerând o formă de undă dreptunghiulară *f(t)* cu o valoare minimă *ymin* și o valoare maximă *ymax* și factorul de umplere *D*, valoarea medie a formei de undă e dată de relația:

$$\tilde{y} = \frac{1}{T}\int_0^T f(t)dt \tag{1.34}$$

cum *f(t)* este o formă de undă dreptunghiulară valoarea sa maximă se atinge pentru $0 < t < D * T$.

1.6. Puntea H

Puntea H (eng. H Bridge) este un circuit electronic ce permite aplicarea unei tensiuni pe o sarcină în orice sens. Aceste circuite sunt adesea folosite în robotică și alte aplicații permițând motoarelor de curent continuu să ruleze înainte și înapoi. Punțile H sunt disponibile ca circuite integrate sau pot fi construite din componente discrete, tranzistoare bipolare sau MOS. Puntea H are numele derivat de la modul obișnuit de desenare a circuitului.

Elementele de comutare (Q1..Q4) sunt, de obicei, tranzistori bipolari sau FET, iar în unele aplicații de înaltă tensiune IGBT. Există, de asemenea, soluții integrate.

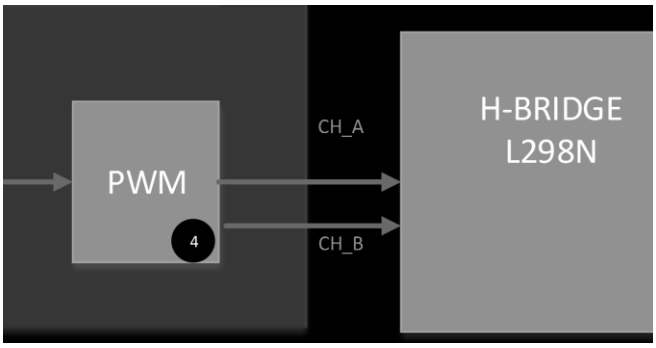


Fig. 1.4 Schema conexiunii PWM - punte H

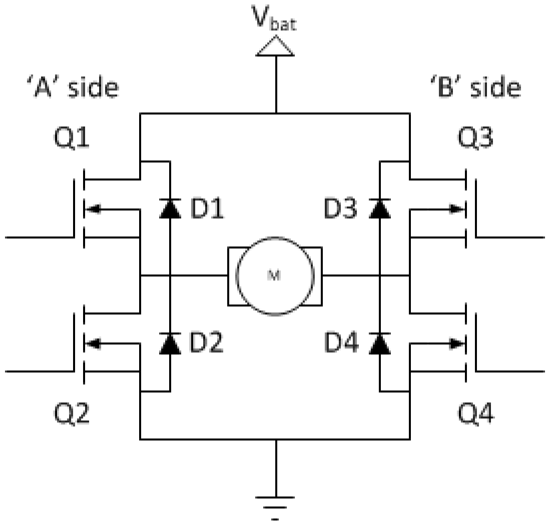


Fig. 1.5 Schema de funcționare generală a punții H

Diodele (D1...D4) se numesc diode de captură și sunt, de obicei, de tip Schottky. Vârful punții este conectat la o sursă de alimentare (baterie, de exemplu), iar partea de jos este la GND. În general, toate cele patru elemente de comutare pot fi pornite și oprite în mod independent, cu toate că există unele restricții evidente. Cu toate că sarcina poate fi, în teorie, orice, de departe cererea cea mai răspândită în cazul punții H este pentru motoare de curent continuu sau stepper.

Pentru a putea fi compatibil cu motoarele DC, în proiectul prezentat chipul punții H trebuie sa aibă un curent de ieșire mai mare de 1.1A și o tensiune de alimentare de minim 12V.

În această lucrare s-a folosit o punte H de tip circuit integrat L298N. Aceasta are o configurație full-bridge și poate comanda cele 2 motoare în mod independent unul de celălalt.

1.7. Motorul brushed DC

Dimensionarea motoarelor DC folosite este făcută cu ajutorul aspectelor expuse anterior; astfel, pentru o greutate de aproximativ 0.6 kg și la o înălțime de 8 cm față de axul de rotație al motoarelor, se pot folosi motoare cu reductor pentru a obține o viteză între 300-500 RPM și un cuplu între 2-4 Kg/cm.

Întregul sistem este pus în mișcare de două motoare brushed DC cu reductor 21:1, viteză 370 RPM. Acestea au și encodere care au un rol important în procesul de optimizare descris în capitolul 4.2.3.

Mai multe detalii despre motoare sunt prezentate în secțiunea care prezintă modelul matematic al motorului în capitolul 3.1.

1.8. Bateria

Este necesară o tensiune de 12V pentru a alimenta sistemul. O asemenea tensiune poate fi obținută dintr-o baterie LiPo cu 3 celule de 3.7V (3C). Timpul de funcționare este ușor de calculat. Motoarele consumă în sarcină maximă 1.1A, iar celelalte componente nu depășesc valoarea de 400mA. Astfel, pentru o autonomie de minimum o oră și jumătate este suficient o baterie cu 2800mAh.

$$autonomie = \frac{capacitatea\ bateriei\ (mAh)}{consumul\ sistemului\ \ (mA)}$$

Bateria are protecție la descărcarea rapidă, deoarece acest incident s-a petrecut și poate avea efecte iremediabile. Senzorul de tensiune al bateriei este format dintr-un *divizor rezistiv* care este interpretat de convertorul ADC al microcontrolerului. Dacă tensiunea bateriei scade sub un prag considerat critic, tot sistemul se oprește și se aprinde un LED de alarmă.

2. Comunicații Seriale. Inter-IC (I2C)

Interfața I2C este un standard răspândit astăzi în foarte multe module electronice. În continuare se va prezenta modul său de funcționare, diferitele caracteristici, avantaje și aplicarea sa practică în proiectul curent.

I2C (eng. Inter-IC) semnifică conectarea circuitelor integrate ca într-o rețea și comunicarea lor pe magistrala astfel creată. Toate circuitele care sunt compatibile au o interfață I2C încorporată, care se poate conecta la unul sau mai multe dispozitive, identice sau nu. Astfel, apar roluri de master/slave în funcție de cine inițiază comunicația și generează semnalul de ceas (SCL).

2.1. Caracteristicile de bază ale interfeței I2C

1. Magistrala I2C este compusă din 2 semnale: o linie serială de date (SDA) și o linie serială de ceas (SCL).

2. Fiecare dispozitiv conectat la magistrală se poate accesa printr-o adresă unică.

3. Relațiile de master/slave sunt întotdeauna prezente.

4. Poate exista o comunicație cu mai multe dispozitive în rolul de master în același timp, deoarece este inclusă opțiunea de evitare a coliziunilor și de arbitrare între mai mulți master când aceștia vor să transmită simultan.

5. Este o conexiune serială, cu preponderență de dimensiune 8 biți, bidirecțională și care poate avea viteze de la 100kbit/s până la 3.4Mbit/s, iar pentru transmisiuni unidirecționale până la 5 Mbit/s.

6. Numărul maxim de dispozitive ce se pot conecta la magistrală este dat de capacitanța magistralei.

2.2. Avantajele interfeței I2C

1. Este foarte ușor de configurat și se poate folosi într-o gama variată de aplicații.
2. Se poate depana foarte ușor și erorile sunt evidente.
3. Are doar 2 pini și este eficient pentru producătorii de IC, deoarece ocupă puțină suprafață pe silicon și nu are un impact mare asupra numărului de pini ca în cazul altor protocoale (SPI).
4. Este o tehnologie larg răspândită putând să interconecteze o varietate de dispozitive: senzori, ecrane LCD, module de memorie etc.

2.3. Protocolul I2C

Pentru a vehicula informația pe magistrala I2C, avem nevoie de cele 2 conexiuni seriale, de date (SDA) și de ceas (SCL). Fiecare dispozitiv este recunoscut printr-o adresă unică și poate fi transmițător sau receptor. Unele dispozitive pot opera doar în modul receptor, astfel un afișaj LCD nu va putea transmite date la fel cum poate face o memorie [14].

Rolurile pe care un dispozitiv le poate avea este fie de master, fie de slave. Masterul inițiază comunicația, generează semnalul de ceas (SCL) și începe transferul de date, fie primește, fie transmite. În acest moment orice alt dispozitiv legat la magistrală devine slave. Dacă există mai mult de un master conectat și doi sau mai mulți master vor să transmită date va apărea situația de arbitrare, când dispozitivele master vor decide care poate să trimită primul, iar celelalte vor trece în modul slave deoarece este posibil ca masterul să vrea să comunice cu alt master și astfel acesta trebuie să fie disponibil.

Pentru a putea transmite informație validă pe magistrală, nivelele HIGH-LOW trebuie să fie stabile. Semnalul SDA poate face trecerea de la HIGH-LOW sau LOW-HIGH doar când semnalul de ceas este LOW.

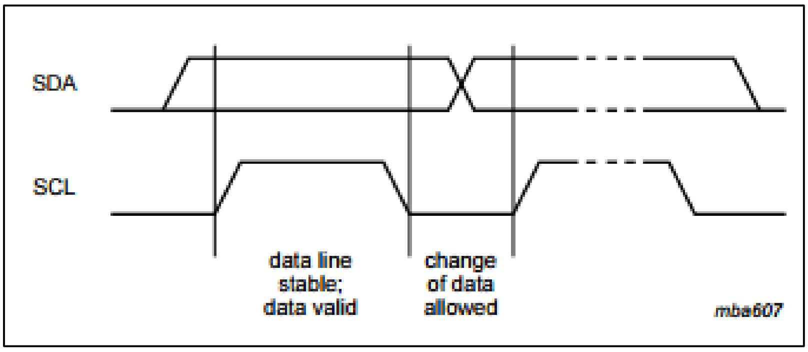


Fig. 2.1 Utilizarea validă a interfeței I2C; Sursa: I2C Manual

Toate transferurile de date încep cu o secvență de START și se termină cu o secvență de STOP.

Secvența de START reprezintă o tranziție HIGH-LOW a semnalului SDA cât timp SCL este HIGH. Secvența de STOP reprezintă o tranziție LOW-HIGH a semnalului SDA cât timp SCL este HIGH.

Secvențele de START și STOP sunt întotdeauna generate de către master. Magistrala este considerată ocupată după o secvență de START și liberă după o secvență de STOP.

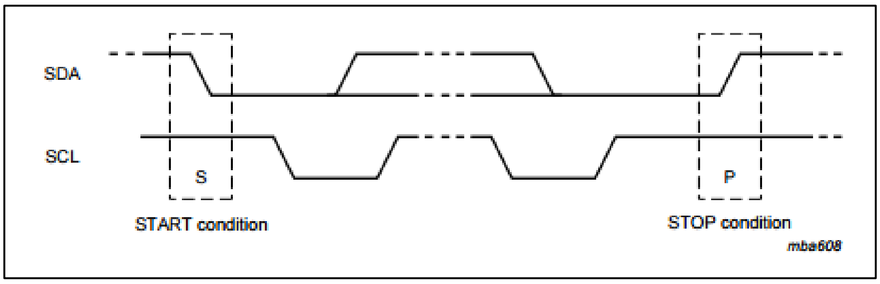


Fig. 2.2 Secvențele de START și STOP; Sursa: I2C Manual

Datele transferate pe magistrală trebuie să aibă lungimea de 8biți și sunt urmați de un semnal de ceas de confirmare (ACK) pe linia SCL. Datele sunt transmise serial, primul bit fiind cel mai semnificativ (MSB).

Pentru ca receptorul să confirme primirea (ACK), masterul eliberează linia SDA pentru ca receptorul să îl fixeze LOW. Dacă SDA rămâne LOW până când pulsul de ceas transmis de master (al 9 puls după cele 8 pentru byte-ul de date transmis) este terminat, înseamnă că transmisia a avut loc cu succes. Dacă semnalul SDA este HIGH cât timp se transmite semnalul de ceas de la master, înseamnă că apare un NACK și transmisiunea nu a fost efectuată cu succes.

Sunt 5 cazuri când poate apărea NACK:

1. Nu există un receptor pe magistrală, deci nu este nimeni să poată confirma comunicația.
2. Receptorul nu este pregătit să primească sau să transmită, deoarece este ocupat cu alte operațiuni.
3. În timpul comunicației, receptorul nu interpretează corect toate datele.
4. În timpul comunicației, receptorul nu poate primi mai mulți biți de informație.
5. În momentul în care un master-receptor transmite terminarea transferului către un slave care a transmis.

2.4. Descrierea pachetului transmis prin protocolul I2C

Pachetul transmis începe cu o secvență de START și adresa slave la care se dorește acces. Adresa slave are 7 biți și este urmată de încă un bit care indică dacă se citește sau se scrie. Dacă acest bit este 0, semnifică scrierea, iar dacă este 1, citirea. La final se transmite secvența de STOP. Dacă se vrea o comunicare prelungită, se poate omite secvența de STOP și se transmite încă o secvență de START (restart). Se poate folosi și adresarea slave pe 10 biți, dar această caracteristică este rar întâlnită. În Anexa 3 se pot observa operațiile de scriere/citire care pot fi efectuate și structura lor.

În proiectul prezentat s-a folosit o interfață I2C Master la viteza de 100 KHz, cu adresare pe 7 biți și o magistrală bidirecțională cu un singur master (XMC4700) și un singur slave (MPU9150). Pentru a putea accesa datele de la senzor, se activează chipul scoțându-l din sleep cu un prim transfer de scriere a registrului de power management. Senzorul este activat și începe să înregistreze valori în regiștrii pentru accelerometru și giroscop. După această inițializare se fac 2 citiri în rafală a câte 6 octeți cu o frecvență de 100 Hz (Anexa 1-2). Astfel, protocolul I2C este folosit pentru ca microcontrolerul să primească direct date reale care nu mai au nevoie de o interpretare sau transformare ulterioară, cum este în cazul senzorilor analogici care au nevoie și de un convertor analog-digital.

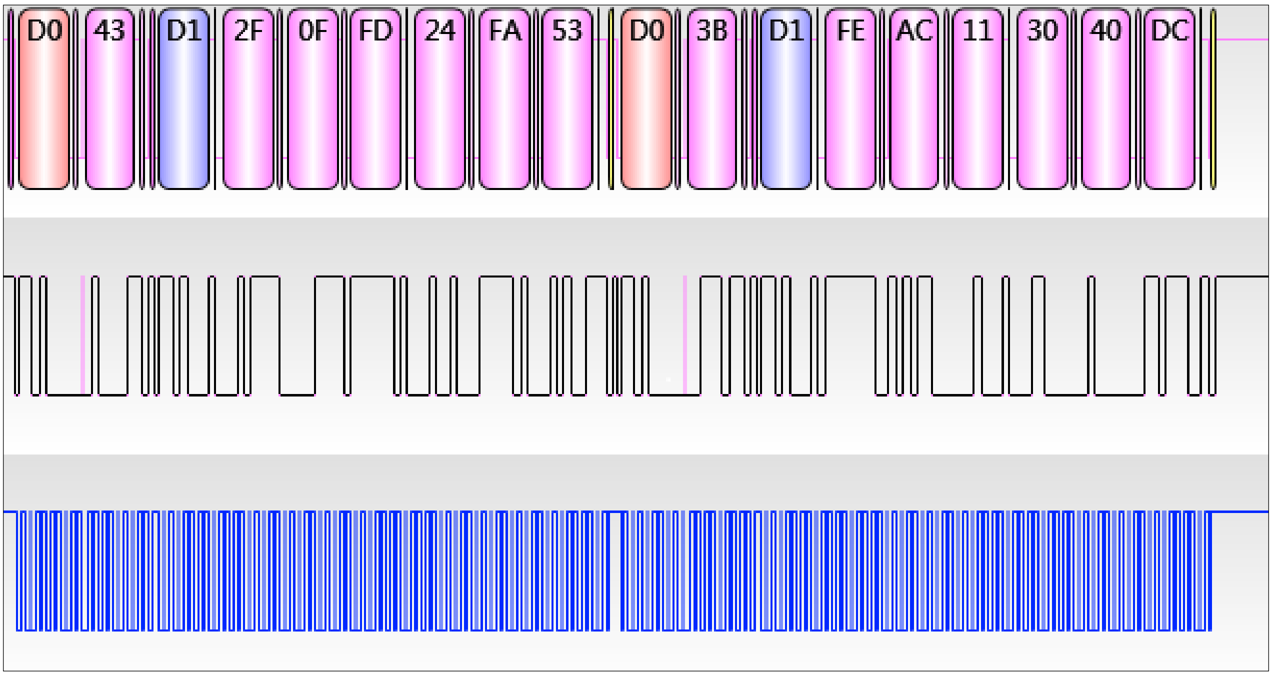


Fig. 2.3 Captură cu analizorul de semnale a bus-ului I2C

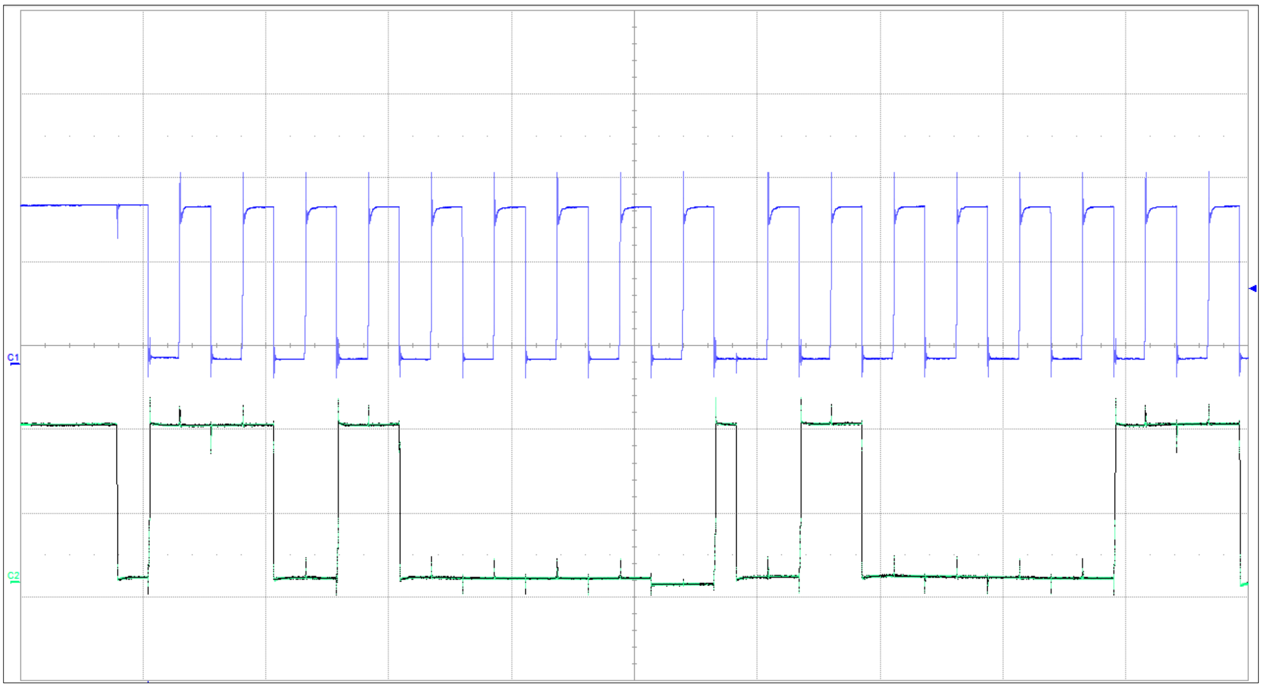


Fig. 2.4 Captură osciloscop a bus-ului I2C

3. Modelarea sistemului

Dinamica robotului trebuie să fie descrisă printr-un model matematic, facilitând dezvoltarea unui sistem de control eficient pentru echilibrare. În acest capitol, sunt prezentate ecuația de mișcare pentru un pendul inversat cu două roți și pentru un motor de curent continuu.

3.1. Modelul liniar al unui motor de curent continuu (DC)

Robotul este alimentat de două motoare de curent continuu Polulu. În această secțiune se va analiza ecuația în spațiul stărilor al motorului. Acest model este apoi utilizat în modelul dinamic de echilibrare oferind o relație între tensiunea de intrare la motoare și cuplul de control necesar.

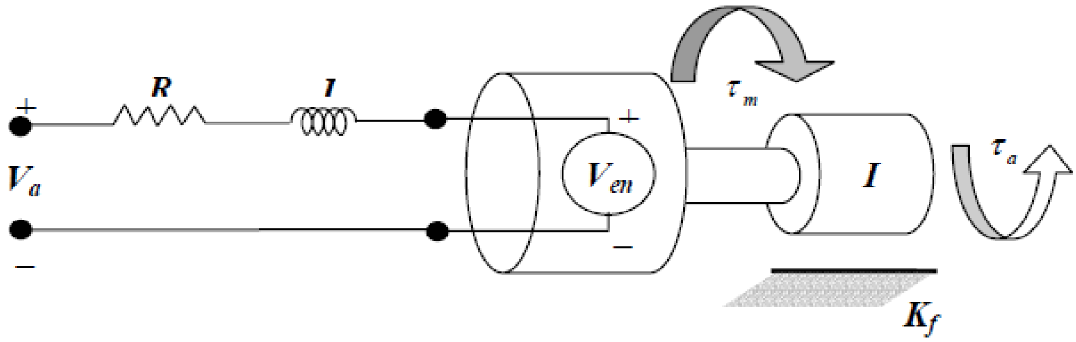


Fig. 3.1 Diagrama unui motor de curent continuu

Figura 3.1 exemplifică un model eficient, liniar pentru un motor de curent continuu. Când o tensiune se aplică la bornele motorului, un curent i este generat între armăturile acestuia. Motorul produce un cuplu τ_m, care este proporțional cu curentul. Această relație poate fi exprimată ca :

$$\tau_m = k_m i \tag{3.1}$$

O pereche rezistor-bobină în serie cu o tensiune, V_{emf}, poate fi folosit pentru a modela circuitul electric al motorului. Tensiune V_{emf} este produsă deoarece bobina motorului se deplasează printr-un câmp magnetic. Tensiunea produsă poate fi aproximată ca o funcție liniară a vitezei arborelui, care poate fi scrisă ca:

$$V_e = k_e \omega \tag{3.2}$$

La acest punct, o ecuație diferențială liniară pentru circuitul electric al motorului de curent continuu poate fi scrisă prin utilizarea legii tensiune Kirchhoff. Legea prevede că suma tuturor tensiunilor din circuit trebuie să fie egală cu zero. Pentru motorul de curent continuu, acest lucru poate fi scris ca:

$$V_a - Ri - L\frac{di}{dt} = 0 \tag{3.3}$$

În derivarea ecuației de mișcare a motorului, frecarea pe axul motorului este aproximată ca o funcție liniară a vitezei arborelui. Legea mișcării lui Newton afirmă că suma tuturor cuplurile produse pe arborele este liniar legată de accelerarea arborelui prin sarcină inerțială a armăturii. Declarația anterioară poate fi scrisă ca:

$$\sum M = \tau_m - k_f\omega - \tau_a = I_R\omega \tag{3.4}$$

Substituind ecuația (3.1) și (3.2) în ecuațiile (3.3) și (3.4) și rearanjând în funcție de derivata în raport cu timpul, sunt evidențiate următoarele două ecuații fundamentale care reglementează mișcarea motorului:

$$\frac{di}{dt} = \frac{R}{L}i + \frac{k_e}{L}\omega + \frac{V_a}{L} \tag{3.5}$$

$$\frac{d\omega}{dt} = \frac{k_m}{I_R}i + \frac{-k_e}{I_R}\omega - \frac{\tau_a}{I_R} \tag{3.6}$$

Ambele ecuații sunt funcții liniare ale curentului și vitezei și acestea includ derivata de ordin 1 în timp. Un model simplificat al motorului de curent continuu este suficient pentru echilibrarea robotului. Din acest motiv, inductanța motorului și frecarea motorului este considerată neglijabilă și este aproximată cu zero. Prin urmare, (3.5) și (3.6) poate fi aproximat cu :

$$i = -\frac{k_e}{R}\omega + \frac{V_a}{R} \tag{3.7}$$

$$\frac{d\omega}{dt} = \frac{k_m}{I_R}i - \frac{\tau_a}{I_R} \tag{3.8}$$

Înlocuind ecuația (3.7) în ecuația (3.8):

$$\frac{d\omega}{dt} = -\frac{k_m k_e}{I_R R}\omega + \frac{V_a}{I_R R} - \frac{\tau_a}{I_R} \tag{3.9}$$

Având în vedere că inductanța motorului este neglijată, curentul prin înfășurare nu este luat în considerare în ecuația de mișcare a motorului. Curentul va ajunge apoi în starea constantă imediat, în comparație cu viteza arborelui, care are nevoie de timp pentru a accelera de la o anumită viteză inițială până la o viteză finală după o modificare a tensiunii de intrare. Dinamica motorului poate fi reprezentată cu un model de spațiu-stare, acesta este un sistem de ecuații diferențiale de ordinul întâi cu funcție de parametri, θ și ω, care reprezintă în mod unic funcționarea acestuia. La intrarea motorului este apoi aplicată tensiunea și cuplul adecvat.

$$\begin{bmatrix}\dot{\theta}\\ \dot{\omega}\end{bmatrix} = \begin{bmatrix}0 & 1\\ 0 & \frac{-k_m k_e}{I_R R}\end{bmatrix}\begin{bmatrix}\theta\\ \omega\end{bmatrix} + \begin{bmatrix}0 & 0\\ \frac{k_m}{I_R R} & \frac{-1}{I_R}\end{bmatrix}\begin{bmatrix}V_a\\ \tau_a\end{bmatrix} \tag{3.10}$$

$$y = \begin{bmatrix}1 & 0\end{bmatrix}\begin{bmatrix}\theta\\ \omega\end{bmatrix} + \begin{bmatrix}0 & 0\end{bmatrix}\begin{bmatrix}V_a\\ \tau_a\end{bmatrix} \tag{3.11}$$

3.2. Modelul robotului

Pendulul inversat cu două roți, are un comportament similar cu un pendul pe o platformă mobilă. Dinamica pendulului și roțile sunt analizate separat la început, dar acest lucru va conduce la două ecuații de mișcare, care descriu complet comportamentul robotului.

Deoarece comportamentul robotului poate fi influențat de tulburări precum cuplul motor, modelul matematic va trebui să ia în considerare astfel de forțe. În primul rând, se obțin ecuațiile de mișcare asociate cu roțile din stânga și din dreapta. Următoarea figură prezintă diagrama de corp liber a ambelor roți. Având în vedere că ecuația pentru roata din stânga și dreapta sunt complet analoage, numai ecuația roții din dreapta este dată.

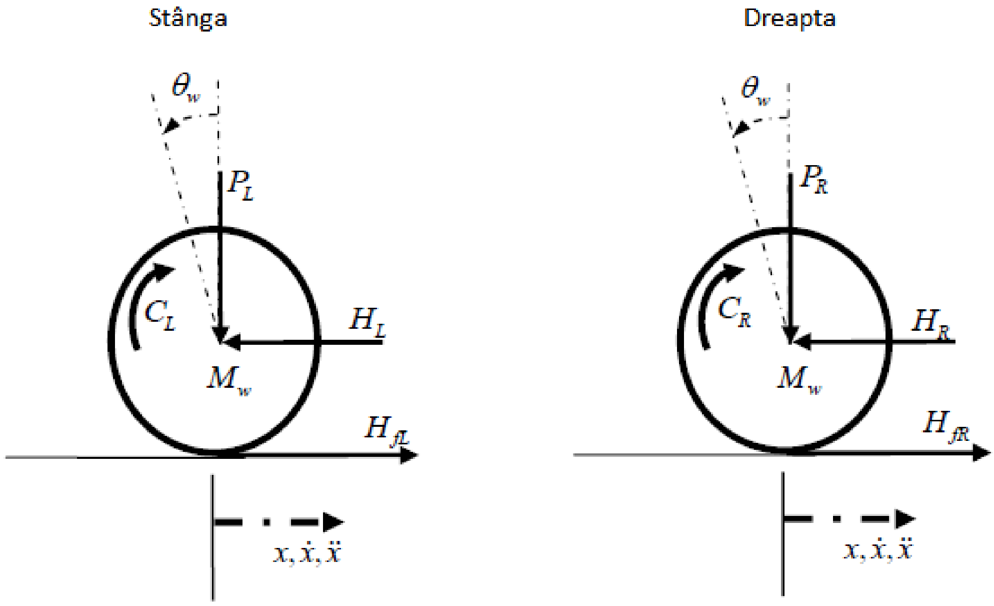


Fig. 3.2 Forțele implicate în modelarea roților

Folosind legea mișcării a lui Newton, suma forțelor pe direcția x este:

$$\sum F_x = Ma$$

$$M_w\ddot{x} = H_{fR} - H_R \tag{3.12}$$

Suma forțelor în jurul centrului roții:

$$\sum M_0 = I\alpha$$

$$I_w\ddot{\theta}_w = C_R - H_{fR}r \tag{3.13}$$

Din dinamica motoarelor de curent continuu, cuplul motorului poate fi exprimat ca:

$$\tau_m = I_R\frac{d\omega}{dt} + \tau_a \tag{3.14}$$

Prin rearanjarea ecuației și substituind parametrii, cuplul de ieșire la roți este:

$$C = I_R \frac{d\omega}{dt} = \frac{-k_m k_e}{R}\dot{\theta}_w + \frac{k_m}{R}V_a \tag{3.15}$$

Prin urmare, ecuația (3.13) devine:

$$I_w \ddot{\theta}_w = \frac{-k_m k_e}{R}\dot{\theta}_w + \frac{k_m}{R}V_a - H_{fR}r \tag{3.16}$$

Rezultă că:

$$H_{fR} = \frac{-k_m k_e}{Rr}\dot{\theta}_w + \frac{k_m}{Rr}V_a - \frac{I_w}{r}\ddot{\theta}_w \tag{3.17}$$

Ecuația (3.15) este substituită în (3.12), pentru a obține ecuația roților.

Pentru roata din stânga:

$$M_w \ddot{x} = \frac{-k_m k_e}{Rr}\dot{\theta}_w + \frac{k_m}{Rr}V_a - \frac{I_w}{r}\ddot{\theta}_w - H_L \tag{3.18}$$

Pentru roata din dreapta:

$$M_w \ddot{x} = \frac{-k_m k_e}{Rr}\dot{\theta}_w + \frac{k_m}{Rr}V_a - \frac{I_w}{r}\ddot{\theta}_w - H_R \tag{3.19}$$

Deoarece mișcarea liniară acționează în centrul roții, rotația unghiulară poate să fie transformată în mișcare liniară prin simpla transformare:

$$\ddot{\theta}_w r = \ddot{x} \Rightarrow \ddot{\theta}_w = \frac{\ddot{x}}{r}$$

$$\dot{\theta}_w r = \ddot{x} \Rightarrow \dot{\theta}_w = \frac{\ddot{x}}{r}$$

Prin transformarea liniară, ecuația (3.18) și (3.19) devine:

Pentru roata din stânga:

$$M_w \ddot{x} = \frac{-k_m k_e}{Rr^2}\dot{x} + \frac{k_m}{Rr}V_a - \frac{I_w}{r^2}\ddot{x} - H_L \tag{3.20}$$

Pentru roata din dreapta:

$$M_w \ddot{x} = \frac{-k_m k_e}{Rr^2}\dot{x} + \frac{k_m}{Rr}V_a - \frac{I_w}{r^2}\ddot{x} - H_R \tag{3.21}$$

Sumând ecuațiile (3.20) și (3.21) :

$$2\left(M_w + \frac{I_w}{r^2}\right)\ddot{x} = \frac{-2k_m k_e}{Rr^2}\dot{x} + \frac{2k_m}{Rr}V_a - (H_L + H_R) \tag{3.22}$$

Șasiul robotului poate fi modelat ca un pendul invers. Figura 3.3 prezintă diagrama de corp liber a șasiului.

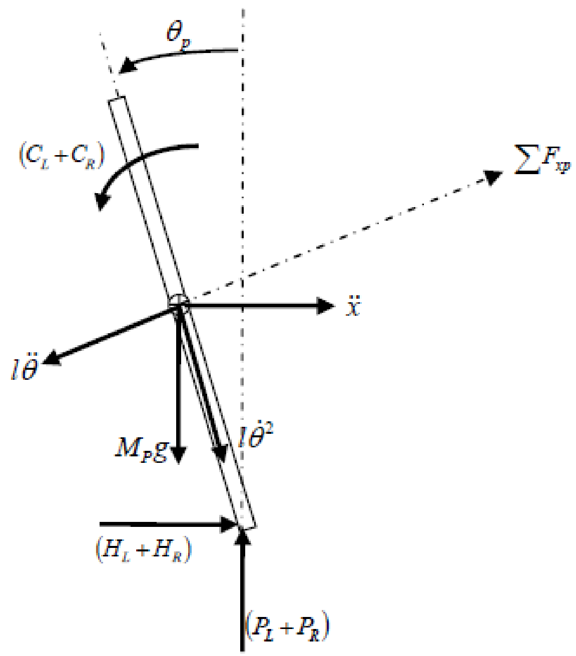


Fig. 3.3 Diagrama de corp liber a șasiului

Din nou, prin utilizarea legii lui Newton, suma forțelor în direcție orizontală:

$$\sum F_x = M_p\ddot{x}$$

$$(H_L + H_R) - M_p l\ddot{\theta}_p \cos\theta_p + M_p l\dot{\theta}_p^2 \sin\theta_p = M_p\ddot{x} \tag{3.23}$$

prin urmare:

$$(H_L + H_R) = M_p\ddot{x} + M_p l\ddot{\theta}_p \cos\theta_p - M_p l\dot{\theta}_p^2 \sin\theta_p \tag{3.24}$$

Suma forțelor perpendiculare pendulului:

$$\sum F_{xp} = M_p \ddot{x} \cos\theta_p$$

$$(H_L + H_R)\cos\theta_p + (P_L + P_R)\sin\theta_p - M_p g \sin\theta_p - M_p l \ddot{\theta}_p = M_p \ddot{x}\cos\theta_p \qquad (3.25)$$

Suma momentelor în jurul centrului de masă al pendulului:

$$\sum M_0 = I\alpha$$

$$-(H_L + H_R) l \cos\theta_p - (P_L + P_R) l \sin\theta_p - (C_L + C_R) = I_p \ddot{\theta}_p \qquad (3.26)$$

Cuplul aplicat pe pendulul de la motor așa cum este definit în ecuația (3.15) și după transformare liniară este:

$$C_L + C_R = \frac{-2k_m k_e}{R}\frac{\dot{x}}{r} + \frac{2k_m}{R} V_a$$

Substituind această relație în ecuația (3.26) rezultă:

$$-(H_L + H_R) l \cos\theta_p - (P_L + P_R) l \sin\theta_p - \left(\frac{-2k_m k_e}{R}\frac{\dot{x}}{r} + \frac{2k_m}{R} V_a\right) = I_p \ddot{\theta}_p$$

prin urmare:

$$-(H_L + H_R) l \cos\theta_p - (P_L + P_R) l \sin\theta_p = \left(\frac{-2k_m k_e}{R}\frac{\dot{x}}{r} + \frac{2k_m}{R} V_a\right) + I_p \ddot{\theta}_p \qquad (3.27)$$

Înmulțind ecuația (3.25) cu – l:

$$\left[-(H_L + H_R) l \cos\theta_p - (P_L + P_R) l \sin\theta_p\right] + M_p g l \sin\theta_p + M_p l^2 \ddot{\theta}_p = -M_p l \ddot{x}\cos\theta_p \qquad (3.28)$$

și se înlocuiește ecuația (3.27) în ecuația (3.28):

$$I_p \ddot{\theta}_p - \frac{2k_m k_e}{R}\frac{\dot{x}}{r} + \frac{2k_m}{R} V_a + M_p g l \sin\theta_p + M_p l^2 \ddot{\theta}_p = -M_p l \ddot{x}\cos\theta_p \qquad (3.29)$$

Pentru a elimina $(H_L + H_R)$ din dinamica motorului, ecuația (3.24) este substituită în ecuația (3.22):

$$2\left(M_w + \frac{I_w}{r^2}\right)\ddot{x} = \frac{-2k_m k_e}{Rr^2}\dot{x} + \frac{2k_m}{Rr} V_a - M_p \ddot{x} - M_p l \ddot{\theta}_p \cos\theta_p + M_p l \dot{\theta}_p^2 \sin\theta_p \qquad (3.30)$$

Rearanjarea (3.29) și (3.30) prezintă ecuațiile neliniare ale mișcării sistemului:

$$(I_p + M_p l^2)\ddot{\theta}_p - \frac{2k_m k_e}{Rr}\dot{x} + \frac{2k_m}{R} V_a + M_p g l \sin\theta_p = -M_p l \ddot{x}\cos\theta_p \qquad (3.31)$$

$$\frac{2k_m}{Rr}V_a = 2\left(M_w + \frac{I_w}{r^2} + \frac{M_p}{2}\right)\ddot{x} + \frac{2k_m k_e}{Rr^2}\dot{x} + M_p l\ddot{\theta}_p \cos\theta_p - M_p l\dot{\theta}_p^2 \sin\theta_p \qquad (3.32)$$

Cele două ecuații de mai sus pot fi liniarizate dacă asumăm că θ = π + φ , unde φ reprezintă un mic unghi față de verticală. Această simplificare a fost folosită pentru a permite un model liniar care urmează să fie obținut. Prin urmare:

$$\cos\theta_p = -1, \qquad \sin\theta_p = -\phi, \quad \left(\frac{d\theta_p}{dt}\right)^2 = 0$$

Ecuația liniarizată de mișcare este:

$$\left(I_p + M_p l^2\right)\ddot{\phi} - \frac{2k_m k_e}{Rr}\dot{x} + \frac{2k_m}{R}V_a - M_p g l\phi = M_p l\ddot{x} \qquad (3.33)$$

$$\frac{2k_m}{Rr}V_a = 2\left(M_w + \frac{I_w}{r^2} + \frac{M_p}{2}\right)\ddot{x} + \frac{2k_m k_e}{Rr^2}\dot{x} - M_p l\ddot{\phi} \qquad (3.34)$$

În scopul de a obține reprezentarea spațiului stărilor a sistemului, ecuațiile (3.33) și (3.34) sunt rearanjate:

$$\ddot{\phi} = \frac{M_p l}{\left(I_p + M_p l^2\right)}\ddot{x} + \frac{2k_m k_e}{Rr\left(I_p + M_p l^2\right)}\dot{x} - \frac{2k_m}{R\left(I_p + M_p l^2\right)}V_a + \frac{M_p g l}{I_p + M_p l^2}\phi \qquad (3.35)$$

$$\ddot{x} = \frac{2k_m}{2Rr\left(M_w + \frac{I_w}{r^2} + \frac{M_p}{2}\right)}V_a - \frac{2k_m k_e}{2Rr^2\left(M_w + \frac{I_w}{r^2} + \frac{M_p}{2}\right)}\dot{x} + \frac{M_p l}{\left(M_w + \frac{I_w}{r^2} + \frac{M_p}{2}\right)}\ddot{\phi} \qquad (3.36)$$

Înlocuind ecuația (3.35) în ecuația (3.34), înlocuind ecuația (3.36) în ecuația (3.33), și după o serie de manipulări a ecuației algebrice, spațiul stărilor pentru sistem este:

$$\begin{bmatrix}\dot{x}\\ \ddot{x}\\ \dot{\phi}\\ \ddot{\phi}\end{bmatrix} = \begin{bmatrix}0 & 1 & 0 & 0\\ 0 & \frac{2k_m k_e\left(M_p l r - I_p - M_p l^2\right)}{Rr^2\alpha} & \frac{M_p^2 g l^2}{\alpha} & 0\\ 0 & 0 & 0 & 1\\ 0 & \frac{2k_m k_e\left(r\beta - M_p l\right)}{Rr^2\alpha} & \frac{M_p g l\beta}{\alpha} & 0\end{bmatrix}\begin{bmatrix}x\\ \dot{x}\\ \phi\\ \dot{\phi}\end{bmatrix} + \begin{bmatrix}0\\ \frac{2k_m\left(I_p + M_p l^2 - M_p l r\right)}{Rr\alpha}\\ 0\\ \frac{2k_m\left(M_p l - r\beta\right)}{Rr\alpha}\end{bmatrix}V_a \qquad (3.37)$$

Unde:

$$\beta = 2\left(M_w + \frac{I_w}{r^2} + \frac{M_p}{2}\right);\ \ \alpha = \left[I_p\beta + 2M_p l^2\left(M_w + \frac{I_w}{r^2}\right)\right] \qquad (3.38)$$

În modelul de mai sus, se presupune că roțile vehiculului vor rămâne întotdeauna în contact cu solul și că roțile nu alunecă. Având în vedere că se cunosc ecuațiile spațiului stărilor putem decide să implementăm o metodă de control. S-a ales pentru aceasta algoritmul PID.

4. Implementarea controlului și optimizarea PID

4.1. Analiza stării inițiale

Deoarece sistemul este în mod inerent instabil, un impuls aplicat în buclă deschisă va determina ca unghiul de înclinare și poziția robotului să crească la infinit. Acest lucru duce la căderea robotului, unghiul de înclinare peste care s-a decis că nu se mai poate reechilibra fiind de 20 de grade pe fiecare parte. Figura 4.1 prezintă simularea pentru un impuls aplicat sistemului liber.

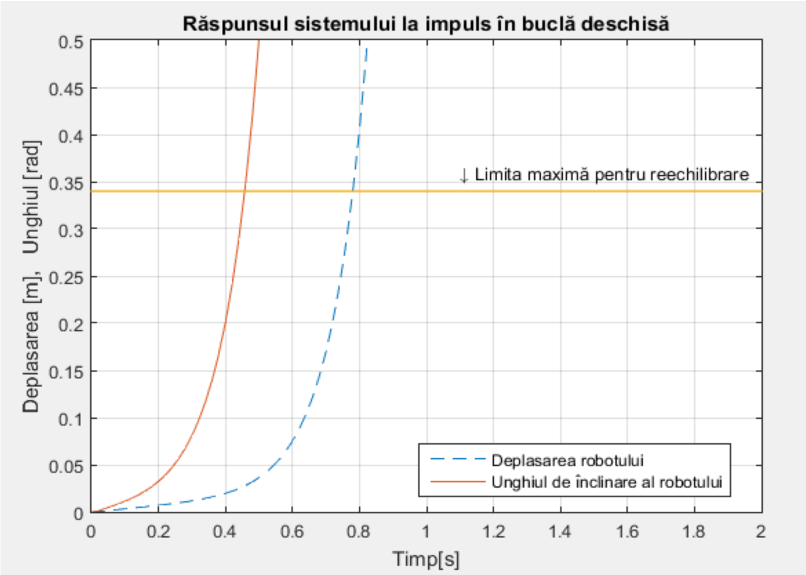


Fig. 4.1 Răspunsul sistemului la impuls în buclă deschisă

Trasarea planului poli-zero-uri al sistemului verifică dacă acesta este instabil. Deoarece există un pol pe planul din partea dreaptă a graficului, instabilitatea se confirmă. În mod ideal, toți polii ar trebui să fie în planul din stânga pentru a fi stabil. Polii sunt prezenți la valorile 0, 9.1605, -9.5141, -1.5739.

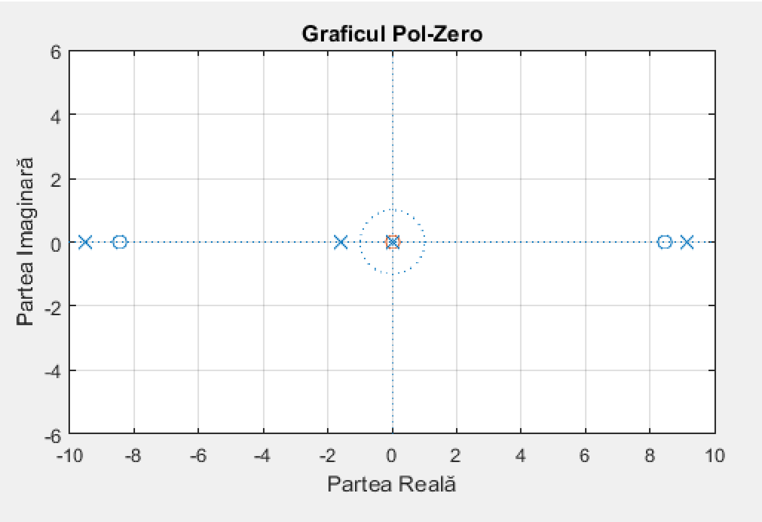


Fig. 4.2 Graficul Pol-Zero

4.2. Controlerul PID

Un **controler proporțional-integral-derivat** (**PID** eng. Proportional-Integral-Derivative) este un mecanism cu buclă de control utilizat în mod obișnuit în sistemele industriale de control. Un Controller PID calculează în mod continuu o valoare de eroare ca diferența dintre o valoare de referință dorită și o variabilă de proces măsurată. Controlerul încearcă să minimizeze eroarea în timp prin ajustarea unei variabile de control, cum ar fi în cazul de față unghiul de înclinare al sistemului, la o nouă valoare determinată printr-o sumă ponderată:

$$u(t) = K_p e(t) + K_i \int_0^t e(\tau)d\tau + K_d \frac{de(t)}{dt} \tag{4.1}$$

unde K_p, K_i, K_d sunt valori pozitive și reprezintă coeficienții proporțional-integral-derivativ.

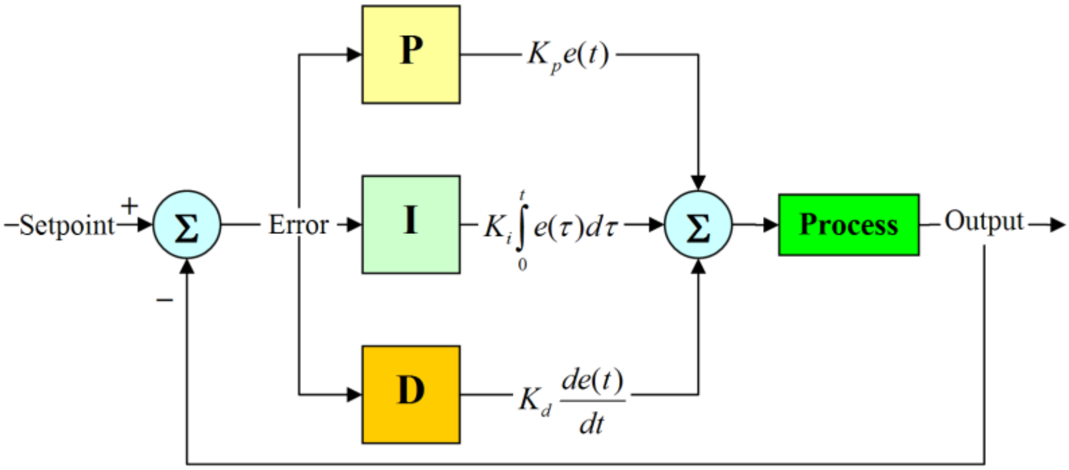


Fig. 4.3 Schema algoritmului PID **[15]**

Fiecare coeficient contribuie la performanță astfel:

- **Proportional**. Acest mod determină viteza de reacție pentru buclă. Cu cât este mai mare câștigul controlerului, cu atât mai repede va răspunde bucla. Cu toate acestea, câștiguri mai mari ale regulatorului duce la depășire și la oscilații ale răspunsului. Pentru fiecare buclă, există o valoare maximă a câștigului controlerului, cunoscut sub numele de câștig final, pentru care bucla este stabilă.
- **Integral**. Contrar credinței populare, modul integral nu produce un răspuns mai rapid. Singura contribuție a modulului integral este de a se asigura că bucla se poate stabiliza doar la valoarea de referință. Ea realizează acest lucru prin ajustarea valorii prejudecată de ieșirea controlerului. Prejudecata trebuie schimbată la o rată în conformitate cu caracteristicile de răspuns ale procesului. Schimbarea prejudecății prea rapid, creează mai multe oscilații și, în cazuri extreme, o buclă instabilă.
- **Derivat**. În unele aplicații, derivata reduce depășirea nivelului dorit și oscilațiile. La rândul său, acest lucru permite să se utilizeze o valoare mai mare pentru câștigul controlerului, deoarece mărește viteza de răspuns.

4.3. Metoda Ziegler-Nichols

Pentru a putea regla algoritmul PID eficient în cazul robotului, se pot aborda mai multe strategii. Cea mai simplă, dar și cea mai costisitoare din punct de vedere al timpului, este metoda try-and-fail [16]. Prin această metodă se incrementează toți coeficienții PID (K_P K_I K_D) până când sistemul devine stabil. Sunt greu de găsit valorile potrivite într-un interval de timp eficient, astfel încât s-a folosit o strategie bazată pe comportamentul modelului.

Metoda aplicată inițial este Ziegler–Nichols, unde în calculul coeficienților PID este nevoie de frecvența de oscilare a robotului (P_u) și de valoarea K_p maximă (K_u) la care acesta intră în oscilație [17].

Având aceste valori ca punct de start pentru parametrii PID, se poate începe optimizarea sistemului, observând comportamentul său și reglând ulterior parametrii.

În punctul acesta există o configurație care poate reprezenta un început în optimizarea echilibrului. Deoarece nu s-au obținut rezultate favorabile s-a decis implementarea unui algoritm genetic de mutare a configurațiilor până se ajunge la o performanță satisfăcătoare. Algoritmul este prezentat în capitolul 4.5.

Ieșirea din blocul PID reprezintă parametrul de intrare al blocului PWM, cel care se ocupă de controlul fizic al motoarelor.

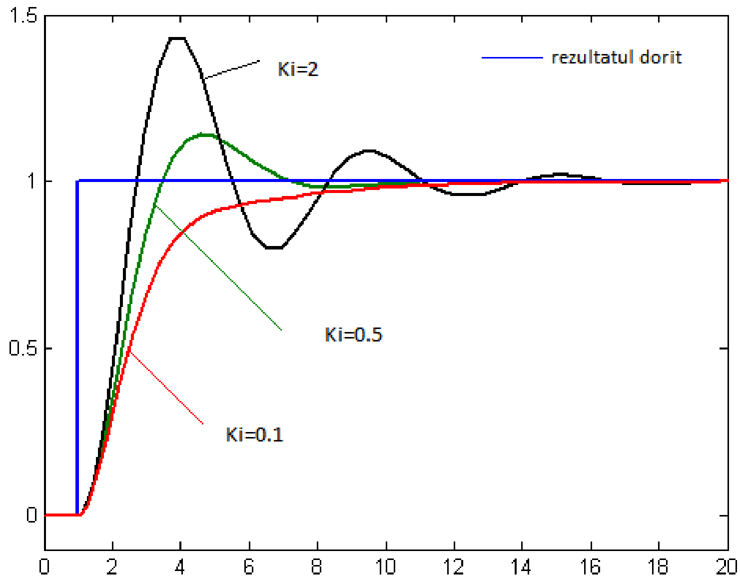


Fig. 4.4 Influența K_i asupra sistemului

Tipul de control	K_P	K_I	K_D
P	0.5 K_u	-	-
PI	0.45 K_u	1.2 K_p / P_u	-
PID	0.6 K_u	2 K_p / P_u	K_p * P_u/8

Tab. 4.1 Coeficienții de tunare Ziegler-Nichols **[17]**

4.4. Reglare PID bazată pe funcția de transfer

Pentru a putea controla acest sistem instabil se va folosi funcția de transfer calculată din ecuațiile de stare. Cu această funcție și cu funcția de transfer a controlerului PID putem simula comportamentul sistemului la diferite valori ale coeficienților PID. Configurația ideală și cea optima nu se pot aplica în realitate deoarece rezultatele obținute cu aceste configurații nu sunt satisfăcătoare. Acest lucru se

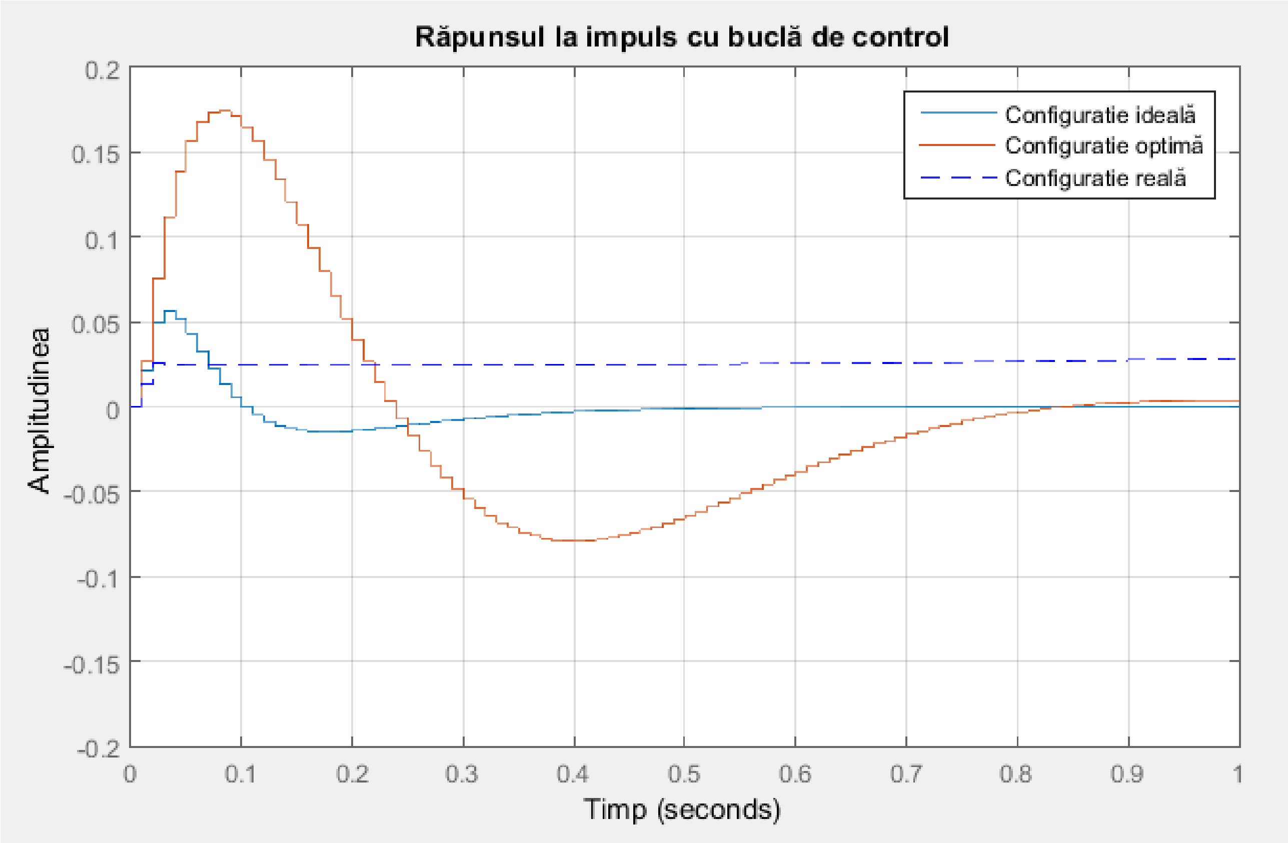


Fig. 4.5 Răspunsul la impuls cu algoritmul PID aplicat

datorează dificultății de modelare a robotului. Formele neregulate ale șasiului, erorile de cuplu ale motoarelor, timpii de răspuns, inerția și indicele de frecare al forței de frecare, toate influențează negativ modelul matematic și îl fac mai puțin credibil în cazul actual. Astfel, singura metodă de îmbunătățire a performanței rămâne metoda evoluției diferențiale [18] care rulează în software-ul de pe microcontroler și lucrează direct cu răspunsul robotului pentru fiecare configurație PID. În acest caz nu mai există limitări date de factorii enumerați mai sus.

Deoarece valorile PWM care sunt folosite de la ieșirea controlerului PID sunt discrete în timp, vom folosi funcția de transfer discretizată cu o frecvență de eșantionare de 100 Hz. Astfel, cel mai bun compromis între overshoot și timpul de stabilizare este reprezentat în figura 4.6.

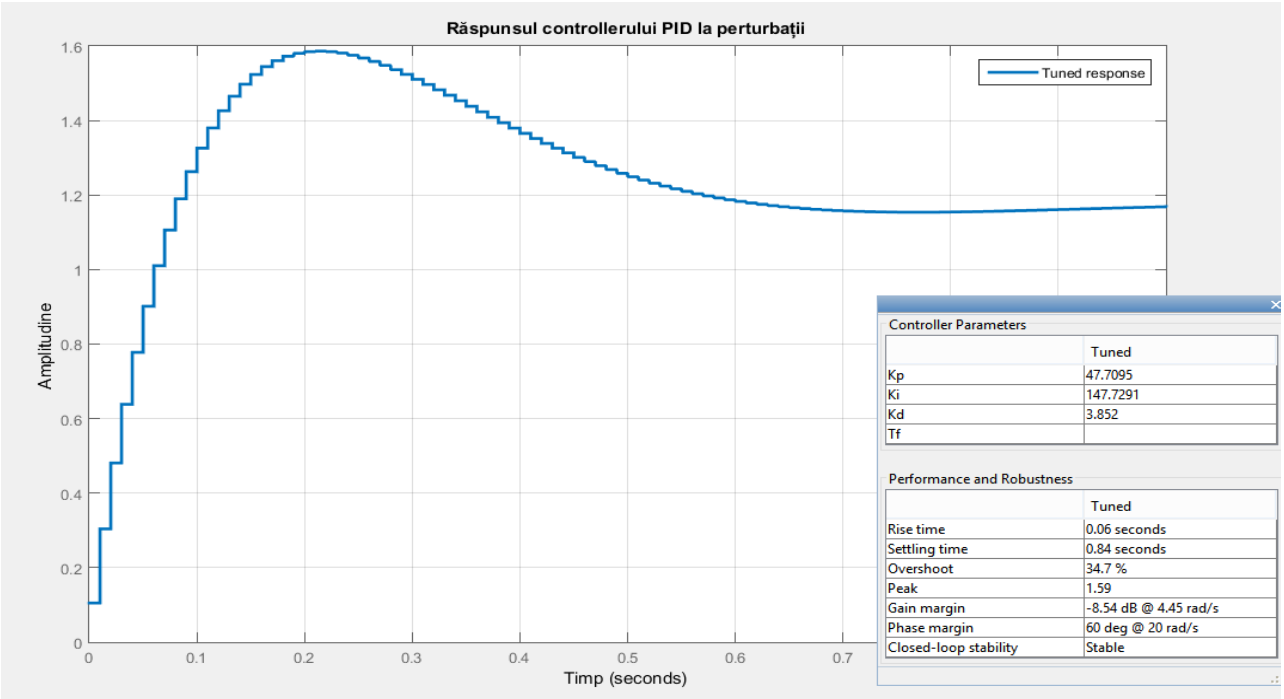


Fig. 4.6 Răspunsul algoritmului PID la perturbații

4.5. Optimizare PID prin metoda evoluției diferențiale

În termeni simpli, optimizarea este încercarea de a maximiza proprietățile unui sistem în timp ce simultan, se minimizează caracteristicile sale nedorite. Care sunt aceste proprietăți și cât de eficient pot fi ele îmbunătățite ține doar de sistemul studiat.

Deoarece schimbarea semnului unei funcții analizate transformă maximele în minime, nu există nicio generalitate pierdută prin limitarea următoarelor discuții numai la minimizarea valorilor funcției. Se va observa că la aplicarea acestei metode este necesară doar căutarea minimului.
Pentru a determina ce soluție de optimizare este potrivită sistemului, se va analiza caracteristicile acestuia.

- **Parametrii**. Sunt variabilele funcției continue, discrete sau ele nu aparțin unui set finit? În plus, sunt toate variabilele de același tip ?
- **Dependența între parametrii**. Sunt parametrii funcției ce urmează a fi optimizată independenți (funcții separabile) sau minimul unuia sau mai mulți parametrii depind de valoarea unuia sau a mai multor parametrii ?
- **Dimensionalitate, D**. Cât de multe variabile definesc funcția ?
- Are funcția doar un minim local (Uni-modal) sau mai multe minime locale(multi-modal)?
- **Dependență de timp**. Minimul local este staționar sau non-staționar (dinamic) ?
- **Zgomot**. Se evaluează un vector. Sunt aceleași rezultate de fiecare dată (fără zgomot) sau fluctuează?
- **Diferențiabilitate**. Este funcția derivabilă în toate punctele de interes ?

Odată evaluate aceste caracteristici se poate lua hotărârea asupra metodei de optimizare. Funcția sistemului studiat are parametrii discreți în timp, cu o dependență între parametrii (performanța depinde de tensiunea de alimentare care descrește în timp, dar și de ceilalți parametrii care au fost optimizați deja), multi-modală, staționară în timp, cu rezultate zgomotoase și derivabilă în toate punctele de interes. Pentru a putea optimiza această funcție se poate apela la algoritmi genetici și strategii evolutive. Metoda evoluției diferențiale are la bază aceste 2 strategii.

Price și Storn au dezvoltat algoritmul DE pentru a fi o funcție de optimizare fiabilă și versatilă care este, de asemenea, ușor de utilizat. Prima publicație scrisă a DE a apărut ca un raport tehnic în 1995 [18]. De atunci, DE a dovedit ce poate în competiții precum Concursul Internațional IEEE pe optimizarea evolutivă (ICEO), în 1996 - 1997 și în lumea reală pe o varietate largă de aplicații. La fel ca aproape toți algoritmii evolutivi, DE este un optimizator de populație care atacă problema punctului de pornire prin puncte inițiale alese în mod aleator. Limitele parametrilor presetați definesc domeniul în care vectorii Np din această populație inițială sunt aleși (Fig. 4.7). Fiecare vector este indexat cu un număr de la 0 la $N_p - 1$. Ca și alte metode bazate pe generarea de populații, DE generează noi puncte care sunt perturbații ale punctelor existente, dar aceste abateri nu sunt nici reflecții, cum ar fi cele din metodele Nelder-Mead [19], nici mostre dintr-o funcție de densitate de probabilitate predefinite, cum ar fi cele din strategiile evolutive [20]. In schimb, DE modifică vectorii cu diferența scalată a doi vectori de populație selectați aleatoriu (Fig. 4.8). Pentru a produce vectorul proces, u_0, DE adaugă diferența vectorului scalat, aleator la un al treilea vector de populație de asemenea selectat în mod aleator (Fig. 4.9). În etapa de selecție, vectorul proces concurează împotriva vectorului populație de același indice, care în acest caz este numărul 0.

Figura 4.10 ilustrează etapa selecției în care vectorul cu valori inferioare ale funcției este marcat ca membru al generației următoare. Cifrele indică faptul că în figurile 4.11 - 4.12 procedura se repetă până când toți vectorii de populație N_p au concurat cu un vector de proces generat aleator. Odată ce ultimul vector proces a fost testat, supraviețuitorii concursurilor N_p vor deveni părinți în perechi pentru următoarea generație în ciclul evolutiv.

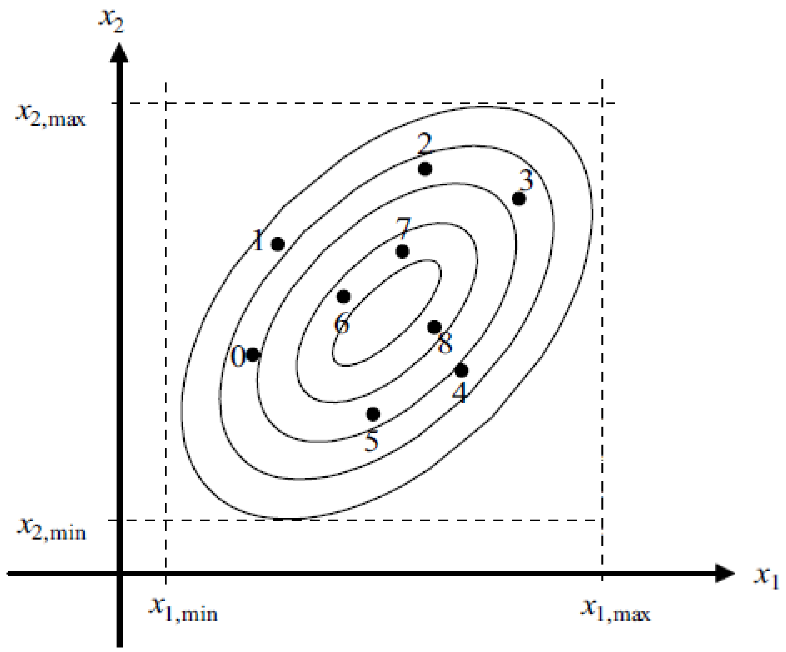


Fig. 4.7 Inițializarea populație **[21]**

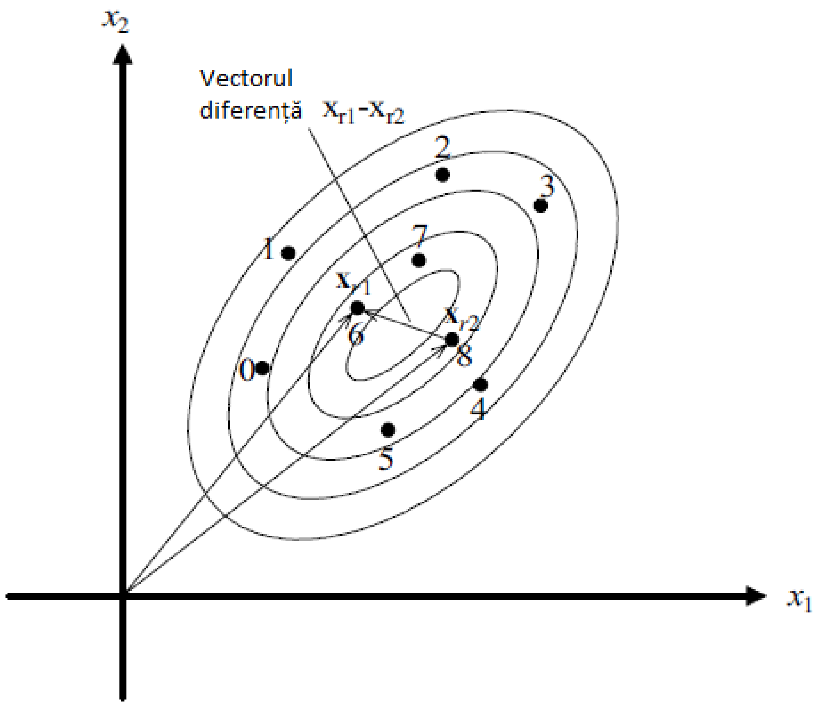


Fig. 4.8 Generarea perturbației x_{r1} - x_{r2} în mod aleator

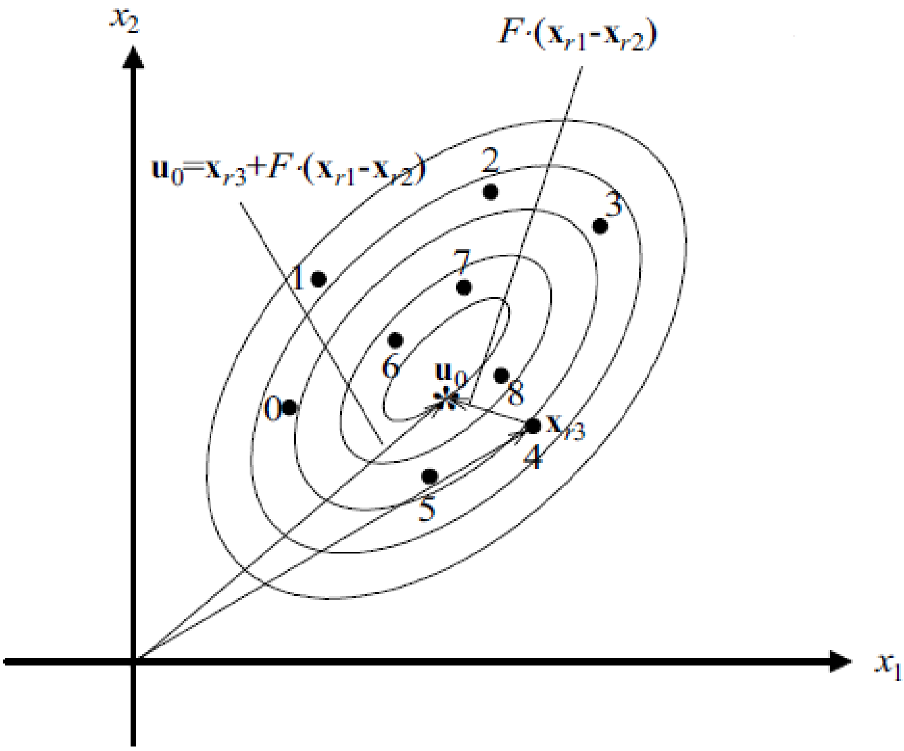


Fig. 4.9 Mutația diferenței vectorilor aleși aleator cu un al treilea, la fel ales aleator

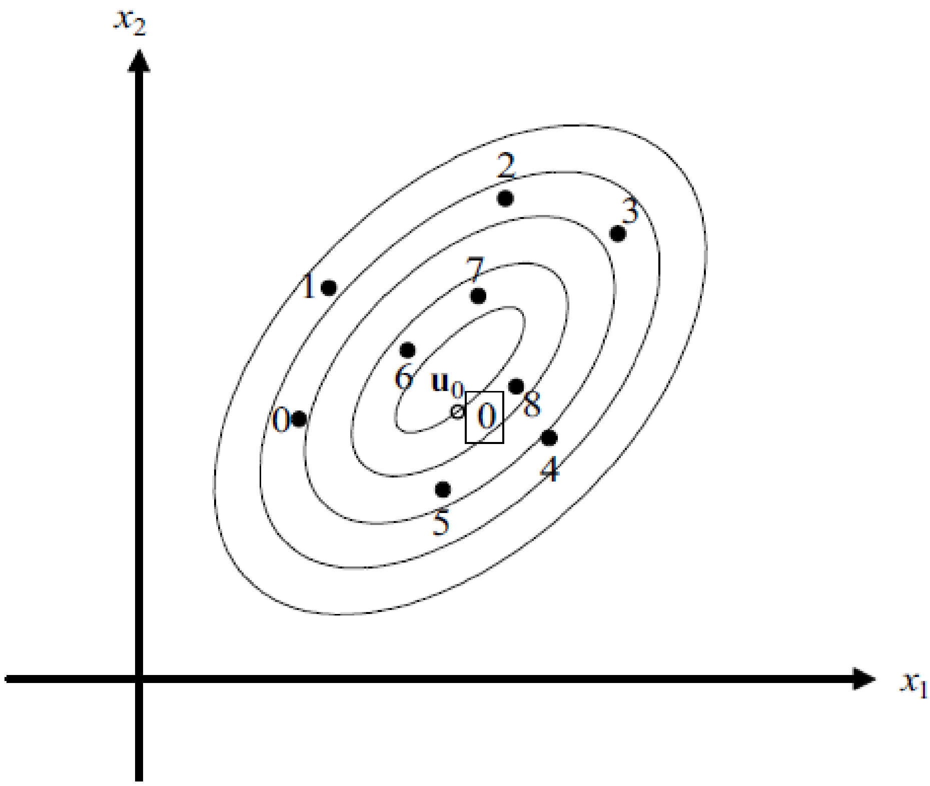


Fig. 4.10 Selecția. Deoarece u_0 este mai performant decât 0 îl înlocuiește în generațiile viitoare

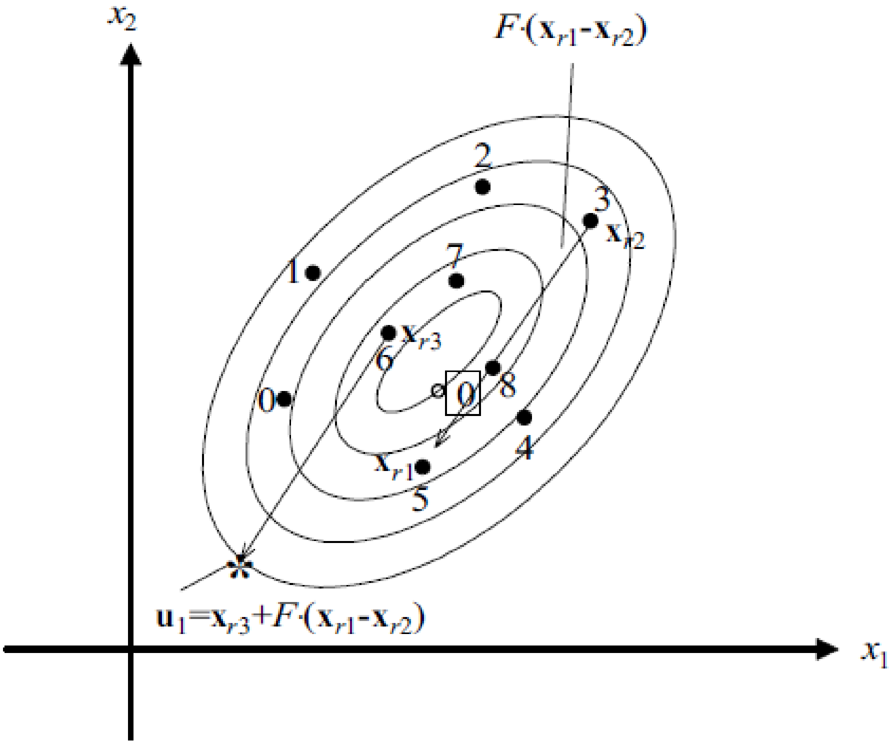


Fig. 4.11 Un nou proces este evaluat

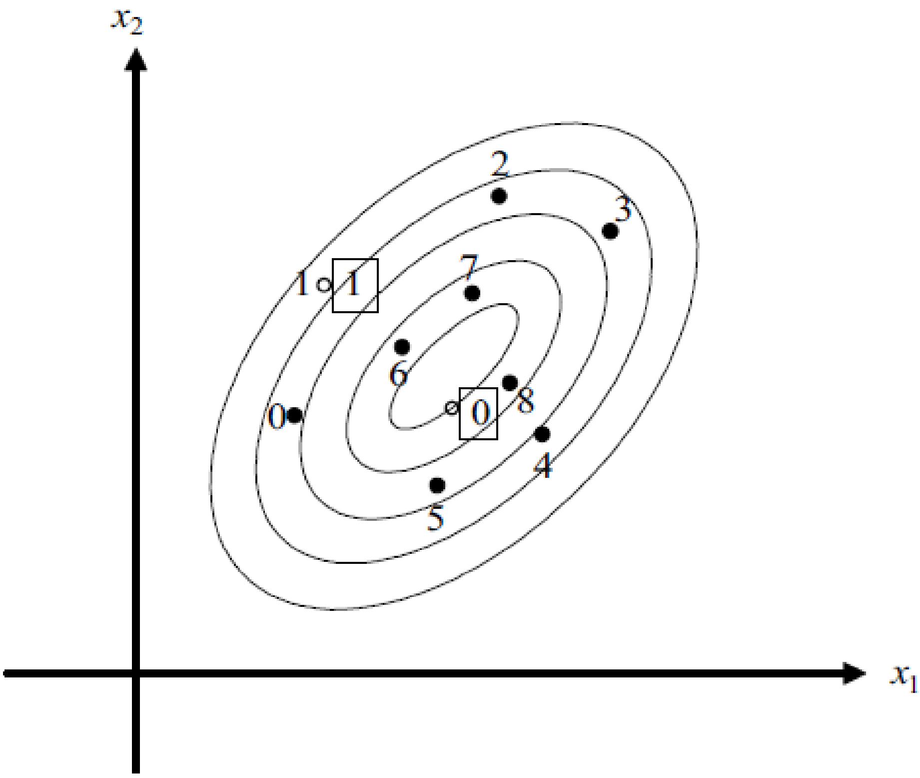


Fig. 4.12 Elementul 1 este mai bun decât elementul cu care concurează, u_1 și nu va fi înlocuit

Particularizând cu sistemul studiat, există 3 variabile care vor fi optimizate, dependente una de cealaltă, având valori discrete.

Algoritmul folosește spațiul tridimensional al variabilelor pentru a putea găsi o tendință de plasare a punctelor performante, pentru a echilibra sistemul și astfel, spațiul acestor soluții este explorat chiar de către robot. Acest lucru se obține plecând de la valorile K_P, K_D, K_I determinate prin metoda try and fail, combinată cu Ziegler-Nichols. Punctul de start al algoritmului necesită o definire a unui spațiu delimitat superior și inferior pentru fiecare parametru ce va fi optimizat din punct de vedere al performanței.

Prima generație va căuta soluții în spațiu delimitat, apoi va încerca să depășească spațiul dacă se consideră că punctul minim global nu va fi găsit în acea zonă. Astfel, algoritmul în funcție de numărul de generații va tinde către punctul de minim. Experimentele au fost făcute cu 30 de generații, fiecare cu cate 10 indivizi. Un test în care se evaluează performanța durează 3 secunde, astfel un întreg set de evaluare și optimizarea durează 20 de minute.

Ecuația care evaluează performanța este de forma:

$$y = var(\theta) + m * avg(encoder) \tag{4.2}$$

unde m este un factor de normalizare, egal cu 0.001 și $avg()$ este media distanței înregistrată de cele două encodere.

În evaluarea performanței are impact varianța unghiului de înclinare, cât și distanța parcursă de robot până la echilibru. Acești parametrii trebuie minimizați simultan pentru ca algoritmul să se dovedească util.

Utilizând acest algoritm de evoluție și evaluând performanța configurației, s-a ajuns la rezultate satisfăcătoare. Acestea sunt analizate pe larg în capitolul 5.

5. Rezultate

Pentru a cuantiza randamentul robotului s-a ales ca indice de performanță **varianța** unghiului de înclinare și **distanța** parcursă până la echilibru. În acest proiect s-a folosit tehnologia Bluetooth pentru a construi canalul de comunicații dintre robot și PC și pentru a putea extrage informații referitoare la performanța acestuia. Se putea folosi și o soluție cu cablu (USB), dar se putea ca firul să influențeze stabilitatea sistemului, astfel încât s-a luat decizia de a se folosi tehnologia Bluetooth.

Cu robotul asamblat și cu software-ul dezvoltate corespunzător, se începe tunarea coeficienților PID prin metoda try and fail. K_D și K_I sunt egalați cu 0 și K_P este incrementat până când acesta începe să oscileze. Apoi, se incrementează K_D până când oscilațiile dispar. Se incrementează din nou K_P și ulterior K_D până când robotul nu mai oscilează puternic. Se repetă această procedură până în momentul în care o incrementare K_P sau K_D nu mai produce schimbări în comportamentul sistemului. Cu aceste valori maxime, se începe calibrarea K_I pentru a încerca să se găsească poziția de echilibru. Robotul nu cade, dar este instabil și oscilează puternic. Deoarece această metodă are nevoie de un timp îndelungat de implementare până la succes, s-a încercat folosirea metodei Ziegler-Nichols în care s-a măsurat empiric timpul de oscilație și valoarea K_P la care începe să oscileze. Folosind tabelul 4.1 s-au determinat valori pentru PID care reușesc să mențină echilibrul un timp mai îndelungat, dar performanța atinsă nu este mulțumitoare.

Existând un comportament pseudo-stabil în care robotul nu mai cade, dar nici nu stă în echilibru, s-a putut trece la faza de optimizare în care scopul principal a fost reducerea oscilațiilor. Algoritmul de optimizare DE [18] se potrivește cu sistemul și a fost implementat pentru a îmbunătăţii performanța. Astfel, s-a reușit în primul rând automatizarea procedurii de testare a configurației PID, reducând substanțial timpul de tunare.

Din acest moment varianța unghiului de înclinare a fost introdusă ca un factor de performanță. Vizualizând grafic acest parametru s-a observat că algoritmul DE nu poate găsi un model de optimizare din cauza zgomotului dat de oscilațiile rapide cu care sistemul se confruntă (Fig 5.1).

S-a constatat că poziția componentelor pe șasiul robotului influențează performanța acestuia. O asemenea componentă este senzorul de înclinare. Inițial, acesta a fost amplasat cât mai sus față de axul roților, dar înregistra târziu căderile și oscilațiile mari. Astfel, senzorul a fost plasat cât mai aproape de roți să aibă un răspuns mai rapid la mișcările bruște de dezechilibru.

O altă problemă de construcție a apărut în cazul bateriei, care este cea mai grea componentă. Aceasta a fost inițial plasată cât mai jos, deoarece s-a considerat minimizarea distanței centrului de greutate față de roți pentru o inerție cât mai mică.

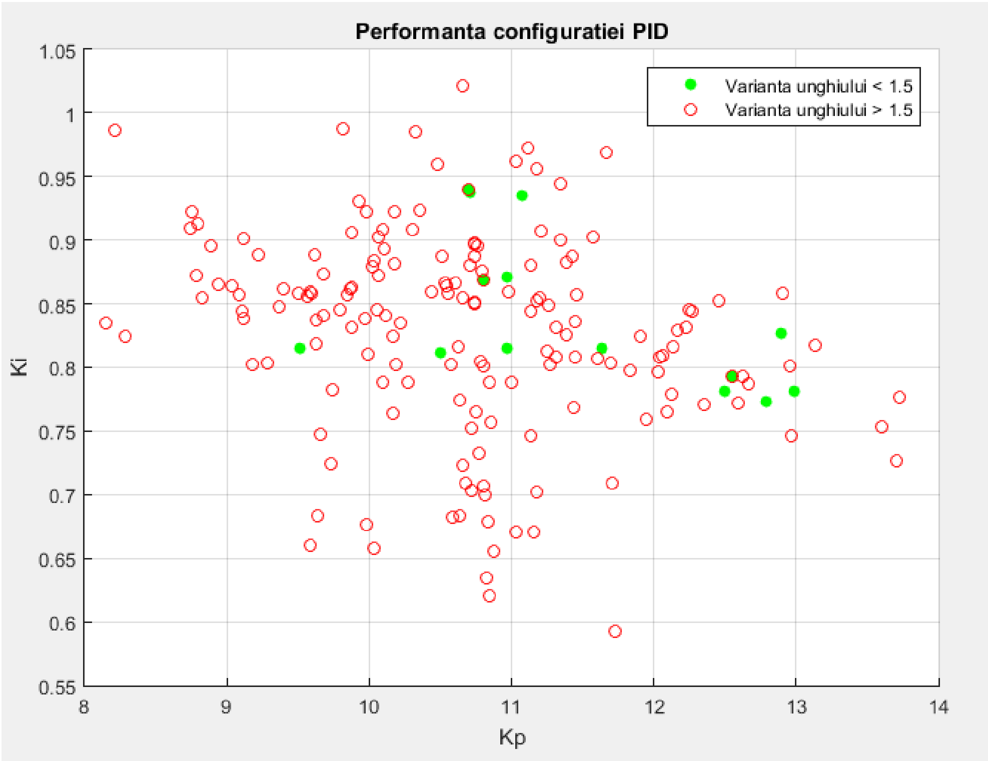


Fig. 5.1 Măsurători zgomotoase din cauza oscilației foarte mari

În figura 5.2 și figura 5.3 se poate observa că varianța unghiului în cercurile pline și verzi este de maxim 5 grade. Acest test a fost făcut în momentul în care bateria era plasată cât mai jos. Conform testelor efectuate cu bateria cât mai jos și cu bateria cât mai sus a reieșit faptul că centrul de greutate poziționat cât mai sus ajută la filtrarea oscilațiile rapide care îl scoteau din echilibru. Astfel, cu o inerție mai mare, sistemului i se oferă un timp de răspuns mai îndelungat pentru reechilibrare. Rezultatele s-au îmbunătățit considerabil și comportamentul robotului s-a stabilizat vizibil (Fig. 5.4). Din acest punct, cu ajustările de șasiu făcute s-au testat cât mai multe puncte din spațiul tridimensional al variabilelor K_P K_D K_I.

Teoretic, algoritmul funcționează corect, caută minimul varianței și încearcă să mărească coeficienții PID. Comportamentul robotului avea momente care implicau mișcare, dar și momente staționare, deci nu era constant. Din acest motiv algoritmul DE nu optimiza în aceleași condiții. Astfel, covarianța unghiului poate fi minimă și în momentul în care robotul se deplasează într-o direcție, cu viteza constantă și un unghi minim de înclinație. Teoretic algoritmul percepe acest comportament acceptabil și îl optimizează până în punctul în care robotul are doar acest comportament, deoarece mișcarea cu unghi constant are varianță minimă față de varianța în momentul de staționaritate în echilibru.

Pentru a evita această situație s-a introdus în formula performanței și media unghiului. Abordarea aceasta reduce numărul de teste fals-pozitive, dar nu în totalitate. Deoarece acest fenomen apare și când media unghiului este mică, nu se elimină problema din cauza proporționalității celor două mărimi, dar diminuează probabilitatea de apariție. Considerând că media unghiului nu este un parametru adițional suficient pentru estimarea performanței s-a luat decizia înlocuirii acesteia cu două encodere pentru cele două motoare. Așa se poate măsura distanța parcursă de robot și se poate încerca minimizarea acesteia cu ajutorul algoritmului DE.

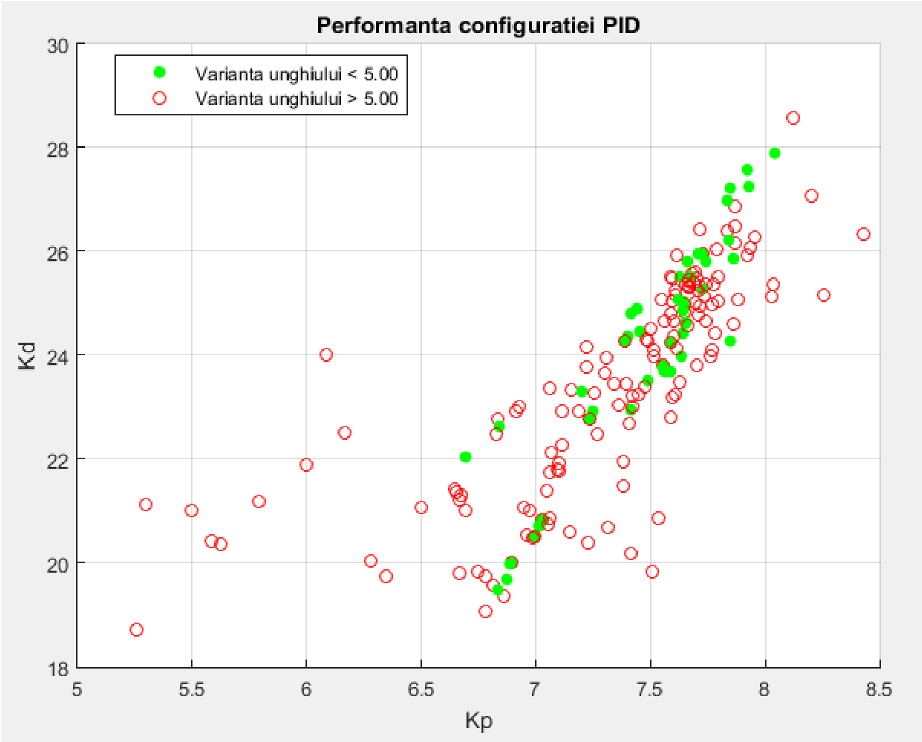


Fig. 5.2 Varianța pentru poziția centrului de greutate ridicat

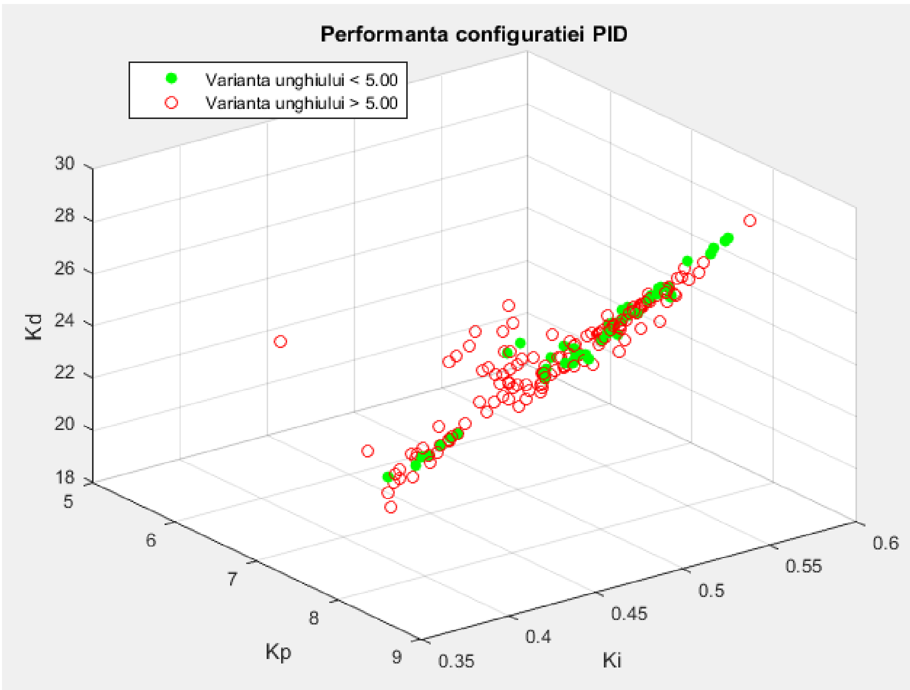


Fig. 5.3 Spațiul 3D cu zgomotul eliminat

Fig. 5.4 Performanța celor 300 de configurații individuale testate

În figura 5.4 se poate observa un test automat cu 300 de puncte în care encoderele își fac simțită prezența. Varianța scade brusc de la 5 grade cum era în testele anterioare la valori subunitare. Se poate observa că algoritmul de optimizare începe să conveargă într-o direcție. Punctele pline verzi indică o performanță acceptabilă, dar punctele pătrate albastre indică o stabilitate crescută. Se observă că în acest test nu sunt foarte multe pătrate. De aceea, se rulează testul din nou schimbând limitele intervalului inițial de optimizare ale algoritmului DE sperând că se va obține o performanță îmbunătățită.

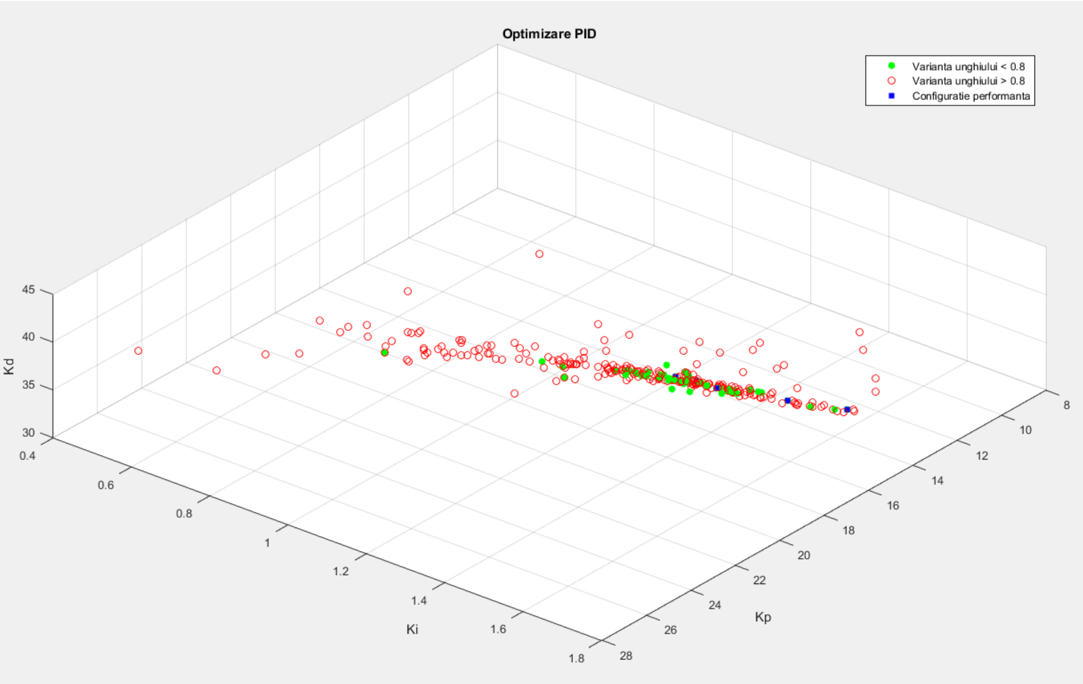


Fig. 5.5 Direcția de convergență a testelor

În figura 5.5 se observă cum, alegând un interval de explorare mai apropiat de starea optimă și un pas de explorare mai mic, algoritmul converge mai repede și direcția soluțiilor este evidentă.

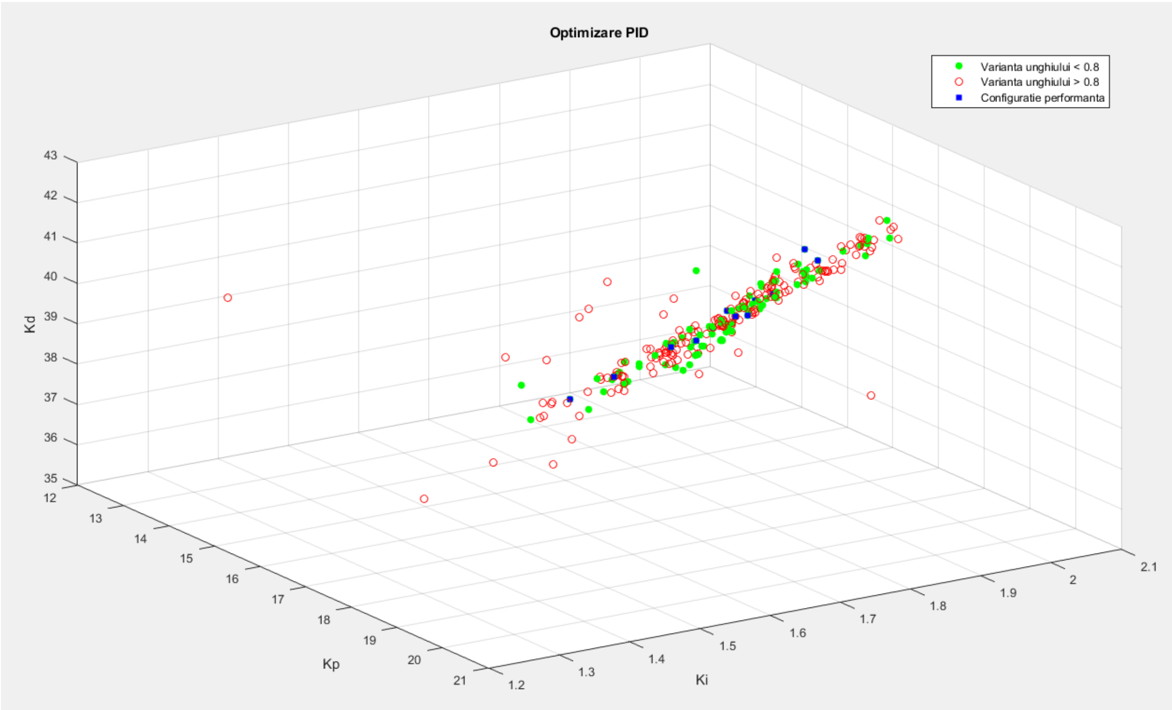


Fig. 5.6 Descoperirea punctelor cu o performanță îmbunătățită în spațiul 3D

Figurile 5.6 și 5.7 arată cum performanța începe sa crească datorită numărului mare de puncte pline verzi. În acest moment pe grafice apare un număr satisfăcător de configurații performante (pătrate albastre). Având această dreaptă a soluțiilor trasată, se poate analiza varianța unghiului acestui test și se poate determina care a fost configurația cu cel mai stabil rezultat. Aceste figuri arată dependența celor 3 variabile, lucru care a fost luat în considerare la început în alegerea algoritmului de optimizare. Se dovedește că acesta s-a potrivit sistemului și că are capacitatea de a explora spațiul soluțiilor, găsind punctul cu cel mai bun randament.

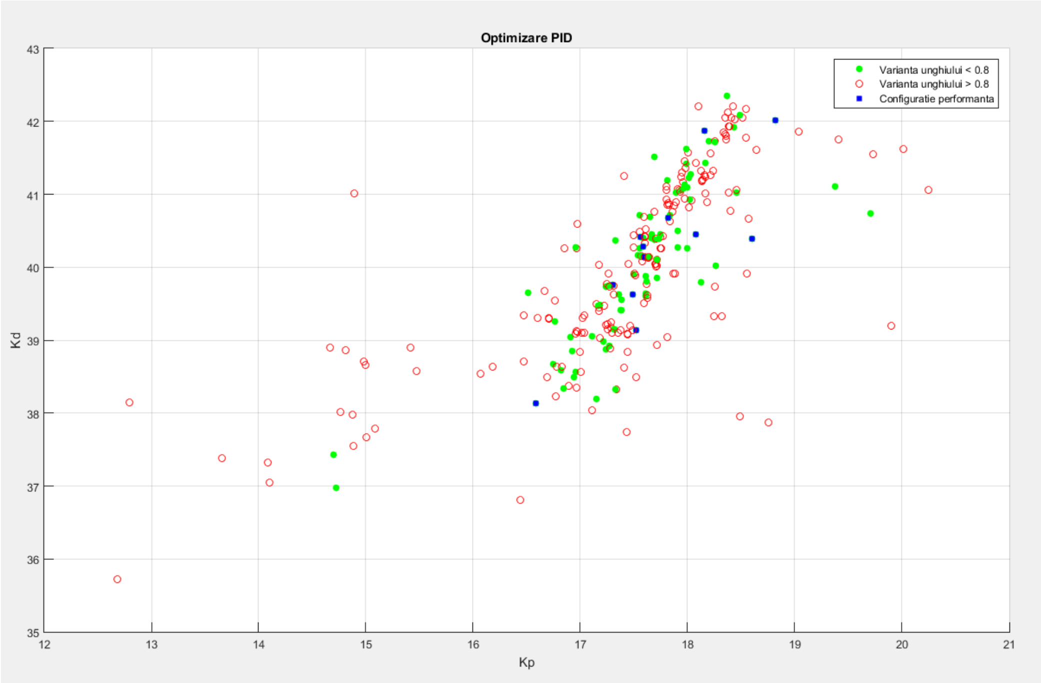


Fig. 5.7 Graficul performanței în funcție de K_D și K_P

În figura 5.7 putem distinge 2 drepte. Prima, cea descrisă de cercurile pline verzi care are o grosime mult mai mare față de cea de-a doua, cea a pătratelor pline albastre. Acest lucru este normal deoarece punctele albastre sunt cazurile particulare în care robotul stă în echilibru și ajunge la acest echilibru parcurgând o distanță minimă. Alegând o configurație de pe linia albastră, analizând varianța din aceste puncte se poate alege o configurație K_P K_D K_I care este suficient de performantă pentru a satisface cerințele sistemului. În tabelul de mai jos sunt prezentate top 10 configurații performante. Se pot observa intervalele $K_P \in [16, 18]$, $K_i \in [1.4, 1.9]$, $K_D \in [38, 40]$.

	Varianța	Distanța	Performanță	KP	KI	KD
1	0.538	294	0.979	16.864	1.624	38.539
2	0.706	185	0.983	16.864	1.684	38.539
3	0.528	321	1.010	17.932	1.474	39.378
4	0.575	294	1.016	17.035	1.642	39.467
5	0.585	290	1.020	16.074	1.832	38.328
6	0.658	246	1.027	16.998	1.657	39.569
7	0.647	255	1.029	15.775	1.954	38.431
8	0.609	284	1.035	18.106	1.497	40.174
9	0.541	341	1.052	17.641	1.622	40.096
10	0.608	310	1.073	16.837	1.632	39.739

Tab. 5.1 Cele mai bune configurații

Privit stric din punctele de vedere al variației unghiului, se poate observa (Fig 5.8) că robotul a căzut doar o singură dată. Dacă se elimină această cădere, adică variația foarte mare a unghiului (Fig 5.9) se observă că media varianței este mai mică de 1.5 grade.

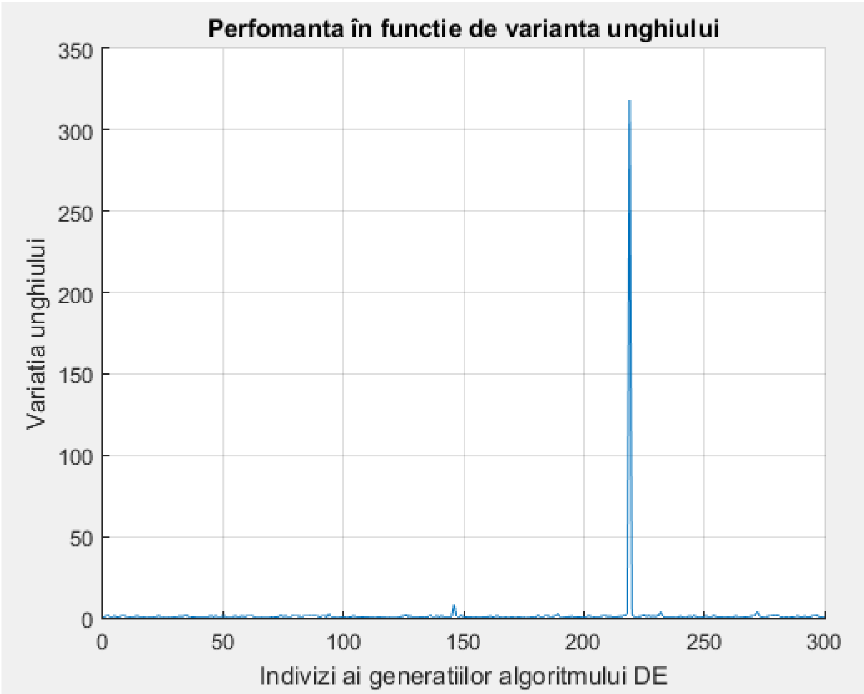


Fig. 5.8 Varianța unghiului

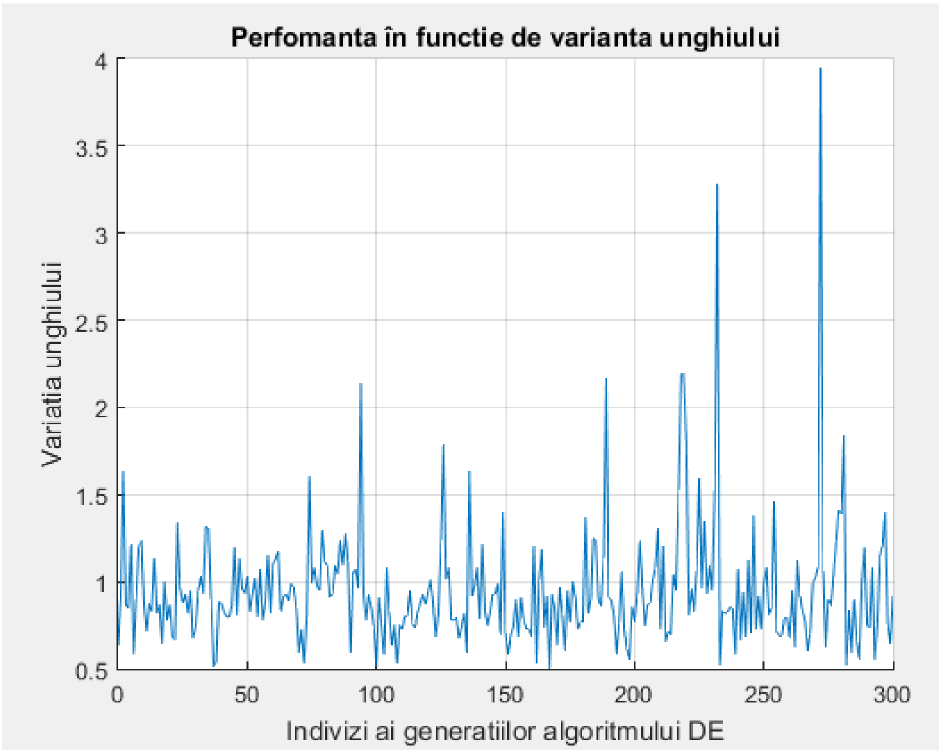


Fig. 5.9 Varianța unghiului, zoom-in

Concluzii

Am dezvoltat pe baza microcontrolerului **ARM M4** un robot funcțional care se auto-echilibrează în timp real. Acesta este independent, toate operațiile fiind desfășurate intern. Am propus o soluție de echilibrare prin estimarea unghiului bazată pe gravitația Pământului. **Senzorul MEMS** determină unghiul de înclinare și îndeplinește cerințele de performanță necesare echilibrării.

Am implementat o interfață de comunicare între microprocesor și senzor prin protocolul **I2C**. Controlul în timp real are nevoie de date de la senzor cu rată fixă, constantă în timp. I2C s-a dovedit a fi soluția potrivită punând la dispoziția sistemului operații de citire în rafală, pachete de dimensiuni mici vehiculate cu viteză suficientă pentru un sistem mecanic, dar și stabilitate pe parcursul transmisiei, zgomotul fiind neglijabil. Am analizat un model matematic al ansamblului care a ajutat la investigarea detaliată asupra robotului. Cu modelul elaborat am simulat stabilitatea acestuia și am propus o metodă de control optim.

Am implementat **algoritmul PID** simulat anterior în formă paralelă pe microcontrolerul ARM. Am iterat mai multe metode de optimizare ai parametrilor de control. Prima și cea mai simplă a fost try and fail. Această metodă este caracterizată de incrementarea parametrilor de control până se ajunge la performanța dorită. Fiind o metodă costisitoare din punct de vedere al timpului, am aplicat **metoda Ziegler-Nichols** care se bazează pe parametrii calculați prin comportamentul robotului. **Algoritmul evoluțiilor diferențiale** s-a dovedit a fi cel mai avantajos, deoarece optimizează parametrii de control în timp prin încercări succesive rulate pe sistem. Am introdus indicatori de performanță ca: varianța și media unghiului alături de distanța parcursă de robot până la echilibru într-o fereastră de timp bine determinată.

În urma optimizării folosite prin metoda propusă în lucrare, performanțele s-au îmbunătățit de la aproape 10 grade la **valori subunitare** ale varianței unghiului, fapt ce determină un comportament stabil. S-a observat că funcționarea poate fi afectată de mai mulți factori. Suprafața de rularea, gradul de descărcare al bateriei, perturbațiile externe și timpul de rulare au fost testate, iar sistemul a funcționat conform.

Complexitatea este mai ridicată în comparație cu un ansamblu pe 4 roți care e stabil intrinsec. E nevoie de senzori care furnizează cu o rată destul de ridicată accelerația pentru a putea prezice unghiul robotului și a aplica un algoritm de stabilizare în timp util, menținând echilibrul. Lipsa de conexiune de depanare și rularea în timp real complică investigarea unor eventuale probleme care sunt greu de reprodus din cauza naturii instabile a sistemului.

Provocarea acestui proiect a reprezentat integrarea tuturor blocurilor funcționale pentru a putea opera în timp real.

Lucrarea s-a bucurat de apreciere la Sesiunea de Comunicări Științifice din cadrul Universității Politehnica București, fiind **premiată** cu locul doi.

În final, se demonstrează că roboții cu două roți pot furniza performanțe similare ca cei pe patru roți.

Bibliografie

[1] V. Croitoru și D. Geleriu, Telecomunicații. Acronime. Termeni. Definiții, vol. I, București: Editura Academiei Române, 2013, p. 305.

[2] M. E. Roshein, Leonardo's Lost Robots, New York: Springer, 2006, p. 69.

[3] S. Narumi, „Japan’s First Robot,” 30 iulie 2012. Disponibil la: http://www.nippon.com/en/views/b00906/.

[4] S. Kemper, Code name Ginger : the story behind segway and Dean Kamen's quest to invent a new world, Boston: Harvard Business School Press., 2003.

[5] Y. Takahashi, N. Ishikawa și T. Hagiwara, „Inverse pendulum controlled two wheel drive system,” 2001.

[6] N. Shiroma, O. Matsumoto, S. Kajita și K. Tani, „Cooperative Behaviour of a Wheeled Inverted Pendulum for Object Transportation,” în *IEEE International Conference on Intelligent Robots and Systems*, Tokyo, 1996.

[7] T. Sugihara, T. Nakamura și I. Hirochika, „Realtime Humanoid Motion Generation through ZMP Manipulation based on Inverted Pendulum Control,” Washington D.C, 2002.

[8] Infineon, „Reference Manual XMC4700,” v1.2, 2016.

[9] Infineon, „DAVE V4 User Manual,” 2016.

[10] M. Elwenspoek și R. Wiegerink, Mechanical Microsensors, New York: Springer, 1993, pp. 132-145.

[11] M. Pedley, Tilt Sensing Using a Three-Axis Accelerometer, Rev. 6 ed., Freescale, 2013.

[12] R. Kalman, „A New Approach to Linear Filtering and Prediction Problems,” *Journal of Basic Engineering,* pp. 35-45, 1960.

[13] V. Croitoru, Comunicații Analog Digitale - Laborator, București: ETTI, 2015.

[14] NXP, „Official I2C Specification,” Rev 6, 2014.

[15] Nick, „Mr. Digital,” 27 Iunie 2011. Disponibil la: https://nicisdigital.wordpress.com/2011/06/27/proportional-integral-derivative-pid-controller/.

[Accesat 10 Martie 2016].

[16] K. Astrom și T. Hagglund, PID Controllers:Theory, Design And Tunning, Washington DC: Instrument Society of America: Research Triangle Park,, 1995.

[17] J. G. Ziegler și N. B. Nichols, Optimum settings for automatic controllers, New York: ASME, 1942.

[18] K. Price și R. Storn, Differential Evolution; A practical Approach to Global Optimization, New Tork: Springer, 1995.

[19] J. Nelder și R. Mead, A simple method for function minimalization, Computer Journal, 1965, p. 303.

[20] H. Mühlenbein și D. Schlierkamp-Vosen, „Evolutionary Computation," în *Predictive models for the breeder genetic algorithm*, 1993, pp. 25-49.

[21] K. Price și R. Storn, Differential Evolution; A practical Approach to Global Optimization, New Tork: Springer, 1995.

[22] K. Lundberg, „web.mit.edu," 21 Ianuarie 2012. Disponibil la: http://web.mit.edu/klund/www/papers/UNP_pendulum.pdf. [Accesat 3 April 2016].

Anexa 1

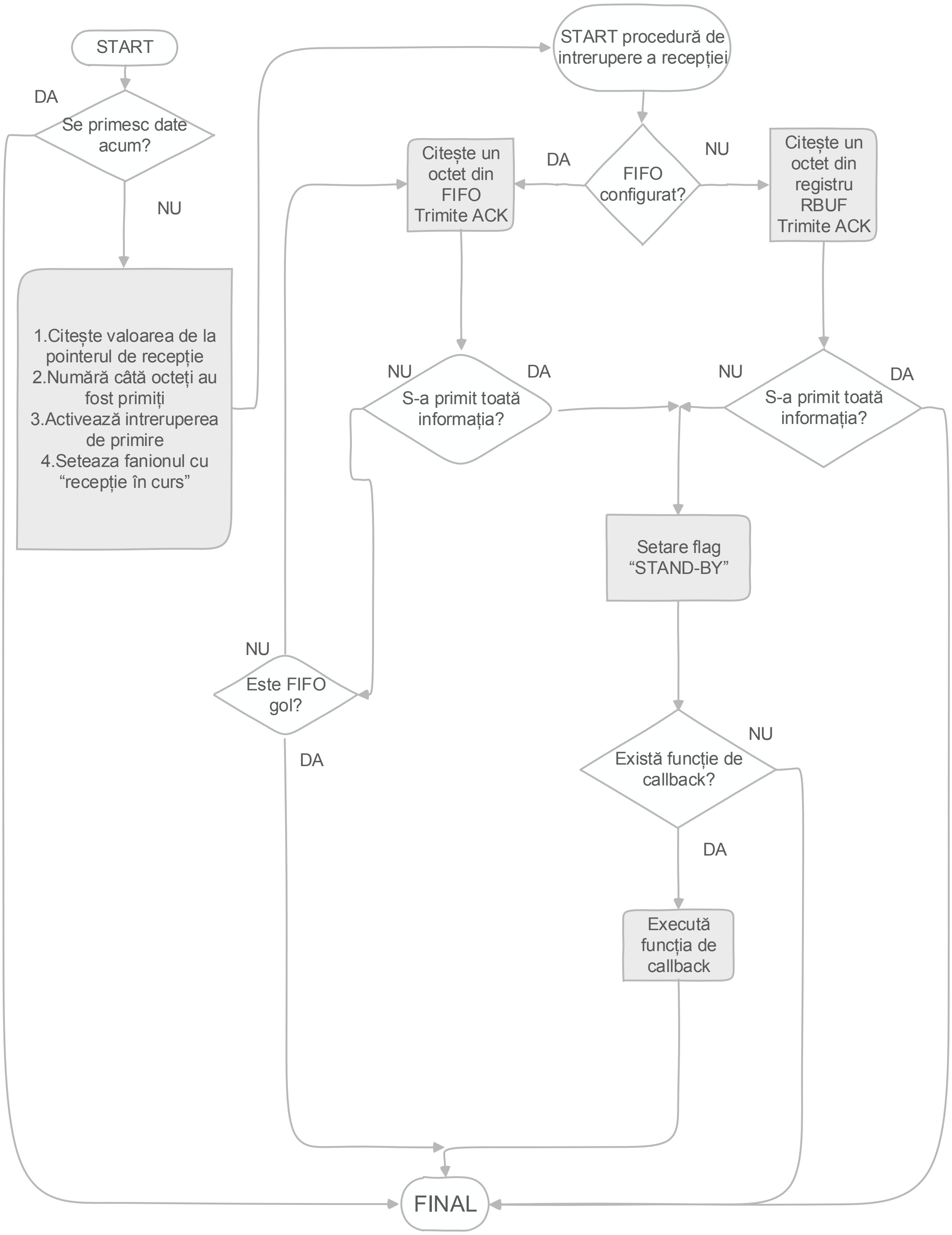


Fig. 1 Diagrama bloc a rutinei de întrerupere pentru I2C RX

Anexa 2

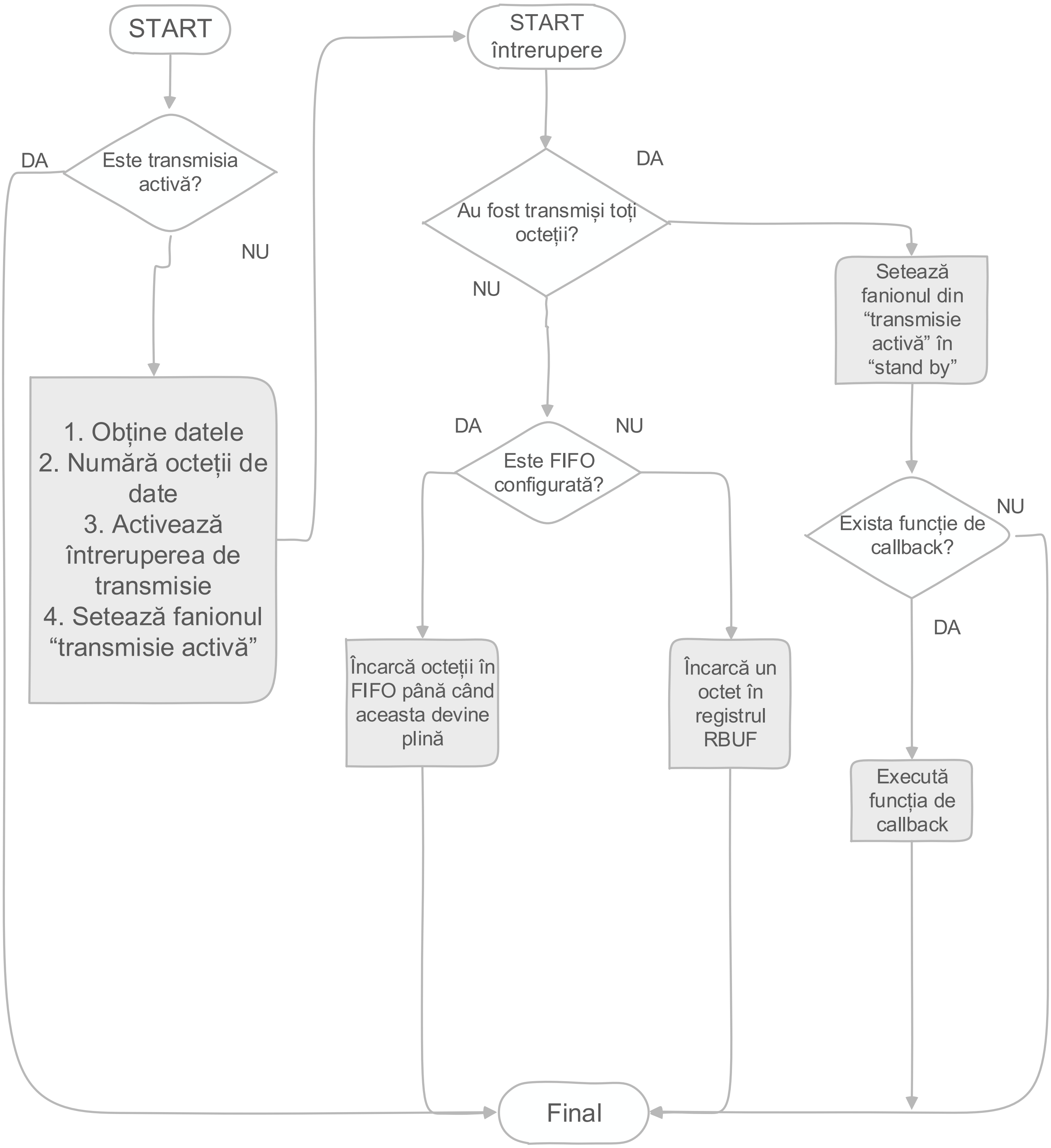


Fig. 2 Diagrama bloc a rutinei de întrerupere pentru I2C TX

Anexa 3

1.Scriere de un octet

Master	START		Adresă Slave +Scriere		Adresă registru		Data		STOP
Slave				ACK		ACK		ACK	

2.Scriere de mai mulți octeți

Master	START		Adresă Slave +Scriere		Adresă registru		Data		Data		STOP
Slave				ACK		ACK		ACK		ACK	

3.Citire de un octet

Master	START		Adresă Slave+ Scriere		Adresă registru		START		Adresă Slave+Citire				NACK		STOP
Slave				ACK		ACK		ACK		ACK		DATA			

4.Citire de mai mulți octeți

Master	START		Adresă Slave +Scriere		Adresă registru		START		Adresă Slave+Citire				ACK		NACK		STOP
Slave				ACK		ACK		ACK		ACK		DATA		DATA			

Fig. 3 Operații folosite în protocolul I2C

Anexa 4

```
//Codul funcțiilor de transfer I2C

const uint8_t Address = 0xD0;
volatile uint8_t tx_completion_0 = 0, rx_completion_0 = 0;
//funcția de READ din MPU--INPUT: //adresa registrului, numărul de octeți //care
trebuie citiți și pointerul //vectorului în care se salvează //octeții primiți
void MPU_Read( uint8_t *mem_address, uint8_t number, uint8_t *receive) {
      I2C_MASTER_Transmit(&I2C_MASTER_0, true, Address, mem_address, 1, false);

       while(tx_completion_0 == 0);
          tx_completion_0 = 0;

      I2C_MASTER_Receive(&I2C_MASTER_0, true, Address, receive, number, true,
true);
       while(rx_completion_0 == 0);
          rx_completion_0 = 0;
}
//funcția de WRITE în MPU--INPUT: //adresa registrului și valoarea ce //trebuie
scrisă în registru
void MPU_Write(uint8_t *mem_address, uint8_t *payload) {

      I2C_MASTER_Transmit(&I2C_MASTER_0, true, Address, mem_address, 1, false);

       while(tx_completion_0 == 0);
          tx_completion_0 = 0;

      I2C_MASTER_Transmit(&I2C_MASTER_0, false, Address, payload, 1, true);

       while(tx_completion_0 == 0);
          tx_completion_0 = 0;
}
```

Apelarea funcțiilor I2C și calcularea unghiului:

```
MPU_Read(&GYROX, 6, received_Gyro);

gyroX = (received_Gyro[0] << 8) + received_Gyro[1];
gyroY = (received_Gyro[2] << 8) + received_Gyro[3];
gyroZ = (received_Gyro[4] << 8) + received_Gyro[5];


MPU_Read(&ACCX, 6, received_Acc);

accX = (received_Acc[0] << 8) + received_Acc[1];
accY = (received_Acc[2] << 8) + received_Acc[3];
accZ = (received_Acc[4] << 8) + received_Acc[5];

float dt = 0.01;
float roll = atan2f(accY, accZ) * RAD_TO_DEG;

float pitch = atanf(-accX / sqrt(accY * accY + accZ * accZ)) * RAD_TO_D
```

Anexa 5

DIAGRAMA GANTT

ID	Activitate	Start	Final	Durarata
1	Documentare pentru partea teoretică	15.02.2016	25.03.2016	6w
2	Documentare pentru partea practică	11.03.2016	31.03.2016	3w
3	Scris la lucrare	01.04.2016	01.07.2016	13w 1d
4	Asamblare si testare initiala a robotului prin try and fail	01.04.2016	21.04.2016	3w
5	Determinarea modelului si diverse simulari MATLAB	06.04.2016	19.04.2016	2w
6	Testare si tunare cu algorimtul Ziegler-Nichlos	19.04.2016	09.05.2016	3w
7	Documentare optimizare algoritm de optimizare DE	02.05.2016	20.05.2016	3w
8	Testare si optimizare prin algoritm DE	25.05.2016	20.06.2016	3w 4d