Calculatoare și Tehnologia Informației
Anul universitar 2022-2023

Universitatea Politehnica Timișoara

DISPOZITIV PENTRU STABILIZAREA TACÂMURILOR

Candidat: Lucian-Călin Ghinescu

Coordonator științific: Conf. Dr. Ing. Lucian Prodan

Sesiunea: Iunie 2023

REZUMAT

Lucrarea propune proiectarea, implementarea și testarea unui dispozitiv de stabilizare a tacâmurilor pentru persoanele afectate de boli neurodegenerative precum Parkinson, tremurul esențial dar și cele a căror mobilitate este scăzută.

Datorită numărului în creștere al persoanelor afectate de aceste boli ce reduc mobilitatea mâinilor și care fac ca o activitate obișnuită, precum luarea unei mese, să devină o activitate ce implică un efort considerabil, problematică sau care poate conduce la anxietate, un dispozitiv ce ar putea reduce aceste efecte ar putea înbunătății starea pacienților.

Dispozitivul asigură amortizarea tremuratului prin combinarea datelor obținute de la senzorul IMU (unitatea de măsurare inerțială) pentru a stabili modul de acționare al motoarelor. În cadrul acestui proiect s-a descris modul de calcul pentru obținerea unei precizii satisfăcătoare a unghiurilor de înclinare .

În realizarea dispozitivului s-au folosit tehnologii relativ noi în piața industrială, cum ar fi motoarele brushless în curent continuu, encodere capacitive dar și algoritmi de de tip FOC (Field Oriented Control). Pe parcursul lucrării au fost prezentate diverse tehnici pentru controlul motoarelor BLDC, plecând de la un mod de control simplu cu resurse hardware minime, până la controlul orientat pe câmp prin diverse tehnici cum ar fi modularea sinusoidală și modularea spațial-vectorială.

Acest dispozitiv este ușor de utilizat, ceea ce reprezintă un aspect important pentru că se adresează în special persoanelor vârstnice sau celor ce au dificultăți în mobilitatea mâinii.

Acest dispozitiv a fost testat, iar în urma testelor s-a constatat că funcționează corespunzător.

CUPRINS

1. INTRODUCERE

1.1 Context

Pornind de la citatul:

„ Sănatatea este sufletul ce animă toate bucuriile vieții, ce se estompează și e lipsită de gust fără aceasta.”

Care aparține filosofului Seneca și cu scopul de a ajuta la rezolvarea unei probleme din sistemul medical, care ar putea înbunătății chiar starea de sănătate a oamenilor, am ales să mă documentez despre situația bolnavilor de Parkinson din România.

Boala Parkinson este o tulburare neurodegenerativă și este principala cauză de dizabilitate. Aceasta afectează tot mai mulți oameni în zilele noastre, de exemplu în anul 2016 s-au înregistrat la nivel mondial un număr de 2,4 ori mai mare decât în anul 1990 [1]. O altă boală, ușor diferită, dar cu simptome asemănătoare este ***tremurul esențial*** (spre deosebire de Parkinson, acesta apare la ambele membre, persoana nu suferă de rigiditate, etc.). Acest tremurat nu are o cauză cunoscută, are o distribuție bimodală (la 20 respectiv 60 de ani), este adesa transmis prin arborele genealogic și se estimează că aproximativ 0.9% din populația globului este afectată de el [1].

Conform articolului [2] și a *CNAS* în România anului 2021 erau înregistrați aproximativ 72 000 de pacienți suferinzi de Parkinson.

Majoritatea pacienților suferinzi de aceste maladii întâmpină numeroase dificultăți în activitățile de zi cu zi. Pentru acestea utilizarea tacâmurilor poate reprezenta un lucru problematic și poate conduce la anxietate și chiar depresie.

Pentru a contracara efectele tremuratului și pentru a aduce un plus de mobilitate am ales să dezvolt un dispozitiv ce își propune stabilizarea tacâmurilor și a ustensilelor asemănătoare.

1.2 Descrierea proiectului

După cum am descris anterior, dispozitivul dezvoltat are ca scop reducerea efectului tremuratului mâinii în timpul folosirii tacâmurilor. Având în vedere cele discutate în sub-capitolul anterior acest dispozitiv ar putea reduce anxietatea pacienților.

Datorită progreselor permanente ale tehnologiilor *hardware* și *software*, am reușit să creez un dispozitiv pentru stabilizarea tacâmurilor. În cadrul dispozitivului am folosit motoare de precizie și durabile fără perii ele fiind printre tehnologiile de vîrf din zilele noastre. Dar utilizarea și controlul acestor tipuri de motoare atrage după sine și adăugarea unei complexități crescute, fiind nevoie de utilizarea unor algoritmi de tip *field oriented control* și monitorizarea cu o precizie ridicată a poziției motoarelor prin diferiți senzori de tip encoder. Acești senzori folosiți în cadrul proiectului sunt de tip capacitiv, o tehnologie

relativ nouă în piață, care asigură o precizie sporită motoarelor dar și o rezistență înbunătățită în condiții de praf, vibrații și șocuri.

Dispozitivul combină datele obținute de la accelerometru și giroscop, le filtrează și obține unghiul față de poziția inițială pentru axele Ox și Oy, datele privind accelerația unghiulară și unghiul de rotație prelevate cu ajutorul giroscopului, influențează în cea mai mare măsură rezultatul final. Aceste date sunt integrate și un bias mic în măsurarea lor poate conduce la o deviere severă a unghiului calculat, făcând ca aceste date să fie utile doar în cazul mișcării continue, în cazul staționar fiind mai relevante datele obținute prin accelerometru.

Astfel dispozitivul își propune să stabilizeze cu un grad cât mai mare tacâmurile.

1.3 Structura lucrării

Lucrarea este structurată în 6 capitole: *Introducere, Specificațiile dispozitivului, Arhitectura dispozitivului, Implementarea dispozitivului, Testarea dispozitivului și Concluzii.*

În capitolul *Introducere* s-au prezentat aspectele generale ale domeniului abordat, dar și descrierea lucrării și obiectivele propuse.

În capitolul *Specificațiile dispozitivului* s-au prezentat aspecte teoretice orientate pe necesitatea specificațiilor în dezvoltarea unui produs, analiza domeniului prin prezentarea dispozitivelor asemănătoare dar și specificațiile dispozitivului dezvoltat de mine.

În cadrul capitolului *Arhitectura dispozitivului* s-au prezentat arhitectura generală a dispozitivului .

În capitolul *Implpementarea sistemului* s-au prezentat etapele necesare realizării proiectului din punct de vedere practic. (resurse hardware, software dar și conectarea componentelor).

În cadrul capitolului *Testarea dispozitivului* s-au prezentat testele efectuate asupra dispozitivului.

În capitolul *Concluzii* s-au prezentat rezultatele obținute și posibile direcții de dezvoltare ulterioară.

2. SPECIFICAȚIILE DISPOZITIVULUI

Specificațiile unui sistem reprezintă descrierea a ceea ce ar trebui să facă sistemul, mai exact funcționalitățile și serviciile pe care acesta le oferă și constrângerile asupra funcționării acestuia [3].

Specificațiile unui sistem se pot clasifica în specificații funcționale și specificații nonfuncționale [3]:

- Specificațiile funcționale se referă la funcționalitățile pe care sistemul ar trebui să le ofere și cum ar trebui să reacționeze sistemul la anumite situații. De asemenea, specificațiile funcționale precizează în mod explicit ce nu ar trebui să facă sistemul dezvoltat.
- Specificațiile non-funcționale reprezintă constrângerile asupra serviciilor sau funcționalităților oferite de sistem.

2.1 Analiza domeniului

Pentru a reliefa carcateristicile produsului dezvoltat de mine, „***EasyEat***”, am ales să îl compar cu alte produse asemănătoare: ***Gyenno Spoon*** și ***Liftware Steady***. Acestea sunt în topul preferințelor utilizatorilor privind ustensilele pentru anularea tremuratului, conform site-ului „ Medical News Today - 5 of the best Parkinson’s spoons 2022 “ [4].

Principalele caracteristici pentru Gyenno Spoon (Figura 2.1) [5]:

- Stabilizare 360
- Amortizarea a 85% din tremuratul mâinii
- Detectare a mișcărilor voluntare ale mâinii
- Pornire și oprire automată
- Greutate : 130g
- Baterie reîncărcabilă și înlocuibilă
- Durată utilizare cu o singură încărcare: 180 de minute
- Garanție: 6 luni
- Preț: 200 $

Principalele caracteristici ale Liftware Steady (Figura 2.2) [6]:

- Scăderea cu 70% a tremuratului mâinii
- Detectarea mișcărilor voluntare și a tremuratului
- Greutate: 100g
- Baterie reîncărcabilă
- Durată baterie: 60 de minute
- Garanție: 12 luni
- Preț: 195$

Figura 2.1 : Gyenoo Spoon [5]

Figura 2.2 : Liftware Steady [6]

În tabelul 2.1 am realizat compararea dispozitivului anti-tremurat „ EasyEat ” cu cele două produse din topul oferit de Medical News Today.

Tabel 2. 1 Analiza comparativă între produsul dezvoltat de mine și alte 2 asemănătoare

Caracteristici și funcționalități	Gyenno Spoon	Liftware Steady	EasyEat
Amortizarea tremuratului %	85%	70%	55%
Detectarea mișcărilor voluntare	x	x	-
Baterie reîncărcabilă	x	x	-
Greutate	130g	100g	450g
Preț	200$	195$	-

2.2 Specificațiile dispozitivului

Dispozitivul are ca scop facilitarea procesului de hrănire, pentru persoanele ce experimentează dificultăți în folosirea tacâmurilor. Pentru a realiza un astfel de dispozitiv, el trebuie să aibă următoarele funcționalități:

- Stabilizare pentru axa de tangaj
- Stabilizare pentru axa de ruliu
- Să aibe dimensouni reduse

Dispozitivul ar putea avea și unele constrângeri precum furnizarea unei surse de energie (priză).

3. ARHITECTURA DISPOZITIVULUI

Acest capitol propune schema arhitecturii generale a dispozitivului: prezentarea componentelor și schema interconectării acestora.

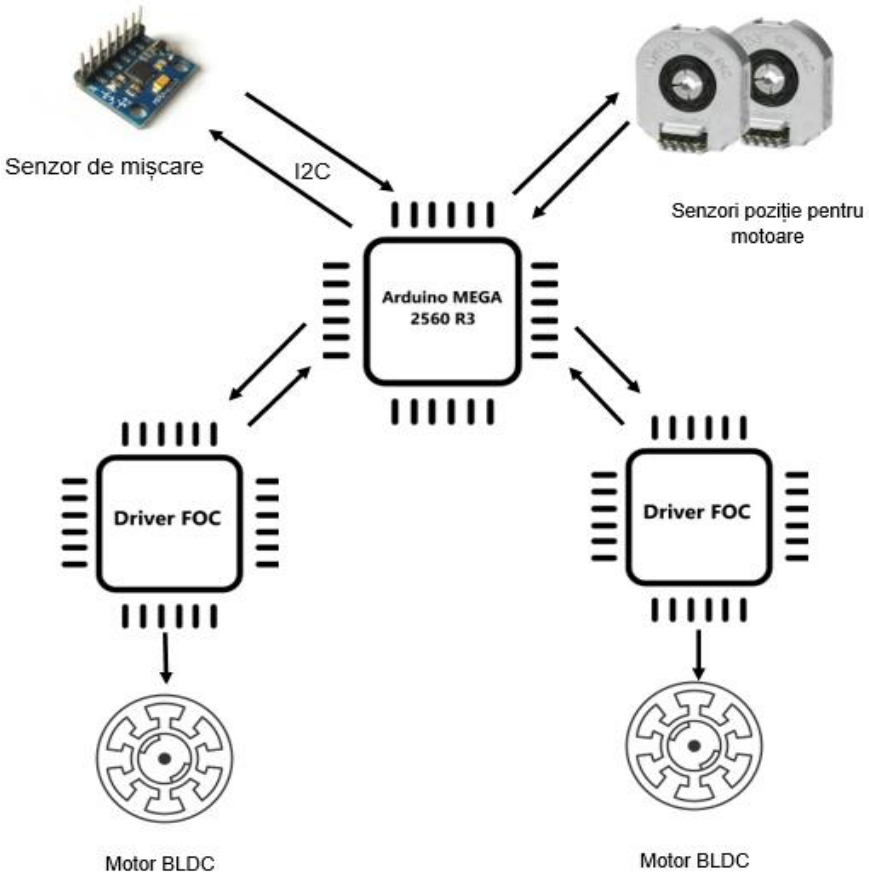


Figura 3.1 Schema generală a dispozitivului

Dispozitivul conține 5 elemente principale: motoarele prin care poziționează tacâmul, driverele motoarelor, microcontroler, senzorul de mișcare și senzorii de poziție pentru motoare.

Senzorul de mișcare IMU furnizează, prin intermediul interfeței seriale I^2C, date legate de viteza unghiulară și accelerația pe cele trei axe microcontrolerului. Acesta din urmă, determinând poziția și orientarea tacâmului.

Senzorii de poziție ai motoarelor, înregistrează date privind poziția și viteza motoarelor și le furnizează microcontrolerului.

Microcontrolerul combină datele primite de la senzori și printr-un algoritm de stabilizare obține comenziile necesare pentru amortizare și le transmite driverelor motoarelor.

Driverele de tip FOC primesc comenziile de la microcontroler și le furnizează prin algoritmul de control de tip FOC (Field Oriented Control) motoarelor de tip brushless.

Procesul de stabilizare a tacâmului este descris, pe larg, în următoarea figură:

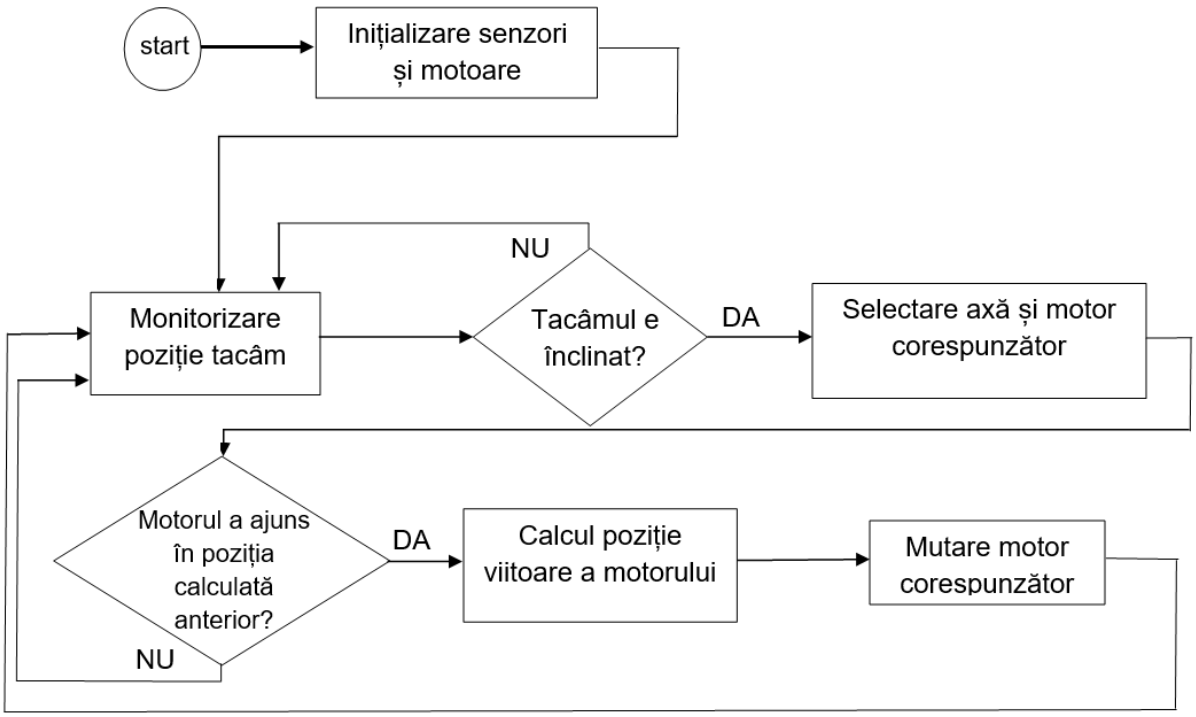


Figura 3.2 Diagrama de evenimente a sistemului de stabilizare

Conform *Figurii 3.2,* procesul de stabilizare începe prin inițializarea motoarelor și a senzorilor pentru obținerea datelor inițiale legate de poziție și înclinare. După acest pas, se monitorizează înclinarea tacâmului și în cazul detectării unei diferențe față de poziția inițială se selectează axa pentru care se va realiza stabilizarea. În continuare se verifică dacă motorul nu este încă în mișcare (nu a ajuns încă în poziția calculată într-un pas anterior) iar în caz afirmativ se calculează poziția necesară pentru a putea aduce tacâmul în poziția inițială și se trimite comanda la motor.

4. IMPLEMENTAREA DISPOZITIVULUI

Acest capitol va descrie etapele realizării proiectului din punct de vedere practic. Se vor prezenta atât resursele hardware cât și cele software ale dispozitivului, dar și modul de interconectare al acestora.

EasyEat este un dispozitiv ce ajută la stabilizarea tacâmurilor, el fiind de dimensiuni medii lucru ce face ca portabilitatea sa să fie limitată.

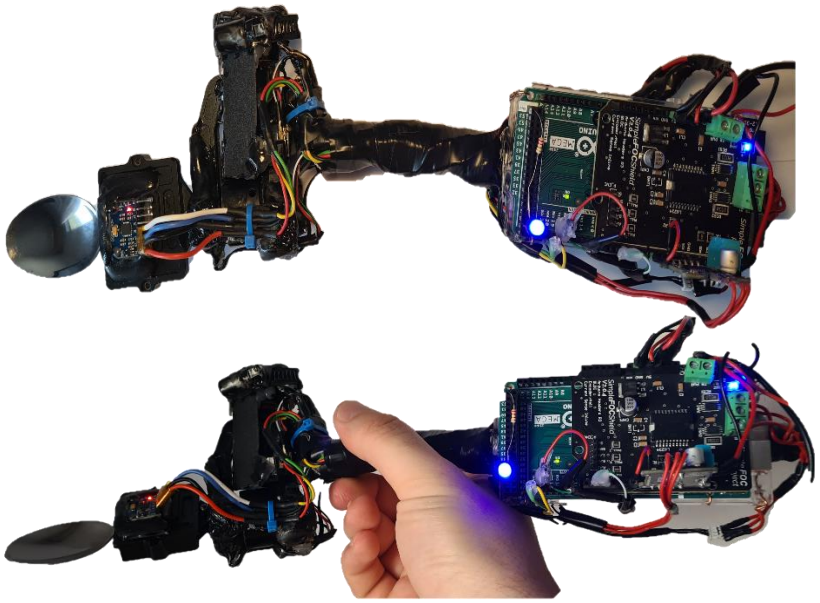


Figura 4.1: Dispozitivul EasyEat

4.1 Resurse hardware

Arduino MEGA Rev3

Pentru a implementa dispozitivul anti-tremurat, am ales să folosesc sistemul cu microcontroler Arduino MEGA Rec3, deoarece este o placa de dezvoltare de dimensiuni mici și accesibilă. El are un preț redus și asigură perfomanța necesară realizării proiectului.

Arduino este o platformă de prototipare electronică open-source înființată în anul 2005. Ea a fost create cu scopul de a fi o platformă accesibilă și ușor de utilizat. Datorită arhitecturii sale modulare și a numărului mare de biblioteci, a devenit populară în comunitatea DIY, permițând crearea de proiecte complexe la un cost mic.

Arduino MEGA R3 (sau REV3) este unul dintre cele mai utilizate modele de microcontrolere oferite de Arduino, el regăsindu-se pe scară largă în proiecte de robotică, IoT și automatizare. El este dispozitivul utilizat în proiectul de față. Arduino MEGA este

bazat pe microcontrolerul pe 8 biți ATMega2560 din familia AVR de la Atmel și rulează la o frecvență de clock de 16MHz. Deține 54 pini de intrare/ieșire (15 din aceștia putând fi utilizați ca ieșiri PWM), 16 intrări analogice, un oscillator de quartz ce rulează la 16 MHz și interfețe USB și I2C. De asemenea, el pune la dispoziție 256 KB de memorie flash (8 KB fiind destinați pentru bootloader), 8 KB de SRAM și 4 KB de EEPROM. Memoria flash este utlizată pentru stocarea programului (codului sursă) încărcat pe placă, SRAM-ul este destinat stocării variabilelor și a altor date temporare generate în timpul execuției și EEPROM-ul este utilizat cu scopul de a păstra date chiar și după ce placa a fost deconectată de la sursă sau a fost resetată. Pe lângă pinii obișnuiți de I/O, placa de dezvoltare mai pune la dispoziție și 6 pini pentru întreruperile de tip hardware.

Figura 4.1.1: Arduino MEGA [7]

MPU6050

Este un circuit integrat ce combină un giroscop cu 3 axe și un accelerometru pe 3 axe formând un MPU (Unitate de Procesare a Mișcării). Este utlizat în diverse aplicații precum drone, senzori de mișcare și roboți.

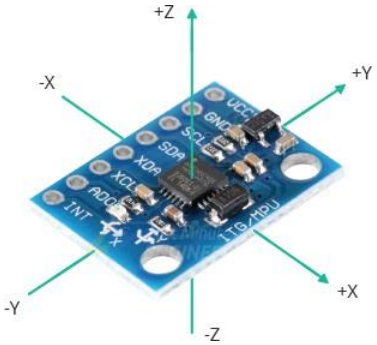


Figura 4.1.2: Senzor MPU6050 [8]

Dispozitivul utilizează tehnologie microelectromecanică (MEMS) pentru a măsura viteza unghiulară și acelerația pe celei trei axe. Giroscopul detectează schimbările în mișcarea de rotație (Roll – rotire în jurul axei Ox, Pitch – rotire în jurul axei Oy și Yaw – rotire în jurul axei Oz) pe când accelerometrul măsoară accelerarea liniară pe orice

direcție. Combinând datele de la acești senzori, el oferă informații despre orientarea și mișcarea dispozitivului.

El este construit pentru a comunica prin interfața I2C, permițându-i conectarea cu un microcontroler. Conține un procesor de semnal digital (DSP) destinat procesării datelor primate de la giroscop și accelerometru și regulatoare și stabilizatoare de tensiune (în mod normal el alimentându-se al 3,3V). Pe lângă senzorii utilizați în detectarea poziției și a orientării, el mai conține și un senzor de temperatură destinat măsurării de temperaturi din intervalul -40 -> 85 °C cu o precizie de ±1°C.

Pentru lucrul cu acest senzor, am utlizat biblioteca MPU6050_light dedicată Arduino.

Motoare BLDC DYS GM2210

Pentru dispozitivul dezvoltat precizia și finețea în mișcare reprezintă un factor important.

Figura 4.1.3: Motor DYS GM2210 [9]

Din cauza acestor factori, am ales să utilizez motoare fără perii (brushless). Acestea reprezintă o variantă modernă de motoare electrice ce oferă o serie de avantaje față de celelate tipuri de motoare convenționale (cu perii de exemplu). Conform [10] printre avantajele motoarelor BLDC se numără:

- Eficiența ridicată : datorată faptului că a fost eliminată frecarea generată de perii și că nu produc scântei electrice ce ar putea afecta motorul
- Durabilitatea ridicată: nu necesită înlocuirea periilor sau curățarea contactelor
- Performanță mai bună la viteze mari
- Control precis al poziției
- Dimensiuni și greutate redusă

Dar totodată, utilizarea acestor motoare necesită și drivere mai performante și mai complexe .

În cadrul proiectului am utilizat două motoare BLDC DYS 2210, specificațiile acestora asigurând funcționarea corectă a dispozitivului. Acestea au 14 poli, o greutate de 44,2 g și cuplu maxim de 3800 g.

Motoarele BLDC sunt tot mai folosite în zilele noastre, piața acestora valorând în zilele nostre, conform [18], aproximativ 15 miliarde de dolari.

Aceste motoare sunt alcăuite din două părți: rotor și stator. În funcție de poziția rotorului ele se diferențiază în *outrunner* (cele folosite în acest dispozitiv) și *inrunner..*

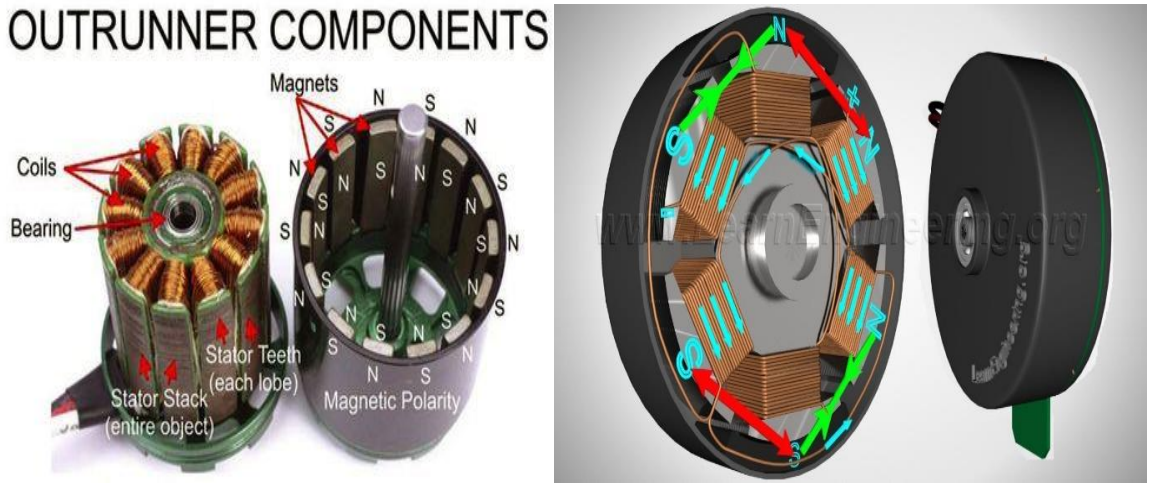


Figura 4.1.4 Componentele unui motor BLDC [18] și principiul de funcționare [19]

Rotorul unui astfel de motor conține magneți permanenți iar statorul conține bobinele bobinele ce vor genera câmpul magnetic necesar rotirii acestor motoare. Rotorul acestui motor încearcă să se alinieze câmpului magnetic creat de bobinele statorului iar când acesta se apropie de poziția potrivită, câmpul magnetic este rotit cu un pas înainte astfel rotorul va „fugi” după câmpul magnetic aflat în cotinuă mișcare.

Pentru a stabili ce bobină și timpul la care se va energiza, un motor BLDC are nevoie de controler electronic, un sensor care să determine poziția rotorului. Cu datele obținute de la sensor controlerul decide ce bobină o să fie energizată .

Controlerul electronic poate avea diferite variante și tehnici de control bazate pe cerințele motorului. O aplicație comună în zilele noastre pentru aceste motoare sunt dronele. În acest caz controlerul electronic este unul de tip *ESC* – electronic speed controller, el fiind capabil să controleze viteza și direcția unui asemenea motor. O altă întrebuințare a acestor motoare sunt gimbal-urile și dispozitivele de control al camerelor în industria video. De obicei, în aceste tipuri de aplicații se folosește un controler special ce implementează o tehnică avansată de control denumită *control vectorial al câmpului magnetic* (Field-Oriented Control – FOC).

SimpleFOC driver

Pentru comanda motoarelor am folosit drivere de tip open-source SimpleFOCShield v2.0.4 și varianta miniaturizată SimpleFOCMINI v1.0. Aceste drivere împreună cu biblioteca Arduino dedicată *SimpleFOC* controlează un motor de tip BLDC după un algoritm de tip FOC (Field Oriented Control). Controlul motorului pe baza acestui algoritm asigurând o precizie ridicată în poziționare, operare foarte lină și un comportament foarte receptiv și dinamic.

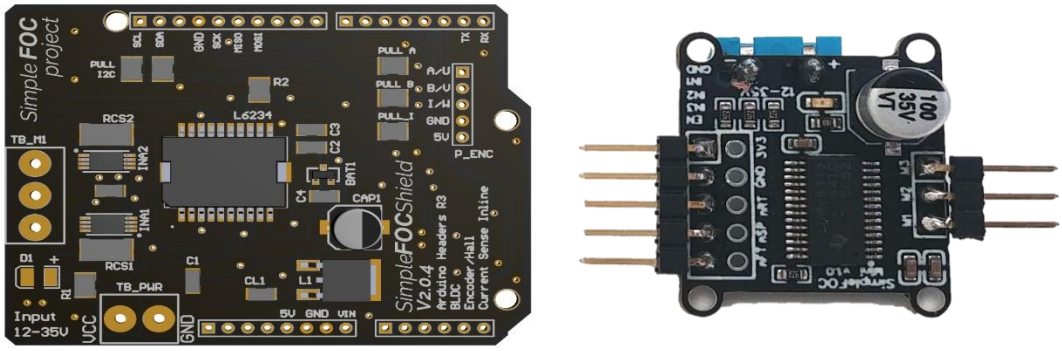


Figura 4.1.5: SimpleFOCShiled și SimpleFOCMini [10]

Driverul SimpleFOCShield include un regulator și un stabilizator de tensiune la 8V lucru ce permite alimentarea plăcii Arduino Mega de la sursa de tensiune de 12 V.

ENCODER CUI AMT 103

Este un encoder incremental modular, cu ajutorul căruia se poate monitoriza mișcarea, poziția sau viteza unui obiect sau a unei părți a unui sistem. Acesta generează semnale electrice (digitale) care indică schimbările de poziție sau mișcare în timp real. Ei sunt utilizați într-o varietate largă de aplicații, cum ar fi roboții industriali, CNC (Computer Numerical Control), imprimante 3D etc. Există mai multe tipuri de encodere: *encoderele optice* ce folosesc un fascicul de lumină și un senzor pentru a converti mișcarea în semnale electrice; *encoderele magnetice* ce utilizează câmpuri magnetice pentru a detecta și a măsura mișcarea. Dar cele două tipuri prezintă o serie de dezavantaje: sensibilitatea la praf, la vibrații și șocuri (cele optice) , rezoluție și acuratețe limitată și sensibilitate la temperatură (cele magnetice).

Figura 4.1.6: Encoder rotativ CUI AMT 103 [13]

Acest tip de encoder folosește o tehnolgie, relativ nouă în piață industrială, codificarea capacitivă (patentată de către *CUI Devices)*. Ea combină avantajele celor două tipuri de encodere descrise anterior, asigurând și un consum de curent mai mic, acest lucru fiind benefic în special în aplicațiile în care energia este asigurată printr-o baterie [13]. Un alt beneficiu alt acestei tehnologii este faptul că rezoluția este programabilă.

Encoderul capacitiv, conform [14], este alcătuit din trei componente principale: rotorul, un transmițător staționar și un receptor staționar. Rotorul este gravat cu după un model sinusoidal și în funcție de viteza de rotire, semnalul de înaltă frecvență emis de transmițător este modulat într-un mod predictibil. Receptorul detectează schimbările în

capacitivitate-reactanță și folosind un algoritm de demodulație, le transpune în incremente de rotație.

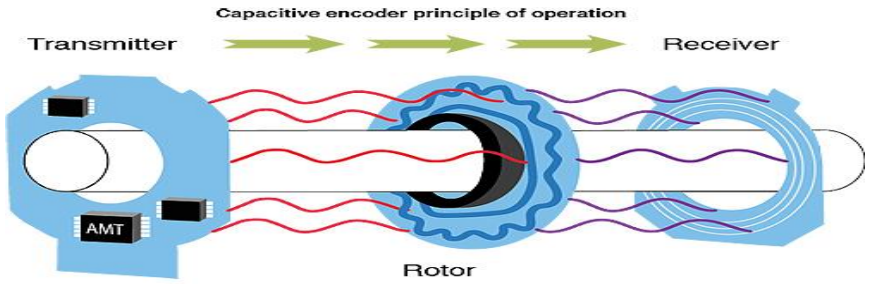


Figura 4..1.7: Mod de funcționare encoder capacitiv [13]

Led

Pentru procesul de inițializare al senzorilor și motoarelor, am folosit un led pentru a marca finalul acestui proces.

4.2 Conectarea componentelor

Pentru implementarea dispozitivului de stabilizare, componenta principală este reprezentată de către microcontrolerul Arduino Mega 2560 Rev3. La acest microcontroler fiind conectate mai multe componente: senzorul IMU, două encodere, două drivere FOC de control pentru cele două motoare dar și un led.

Acest microcontroler are următoarea configurație a pinilor:

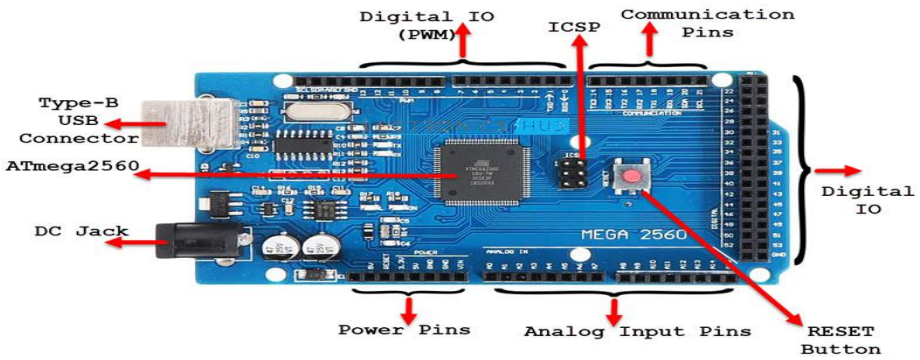


Figura 4.2.1: Configurație pini Arduino Mega [15]

Senzorul *MPU6050* utilizează 4 pini ai microcontrolerului: alimentare 3.3V și GND, și se conectează la unul din cele 2 grupuri destinate comunicației I^2C, în cazul de față pinul 20 SDA și pinul 21 SCL. Pinul SDA asigură transmiterea datelor în mod serial, iar pinul SCL sincronizează comunicația dintre cele două dispozitive.

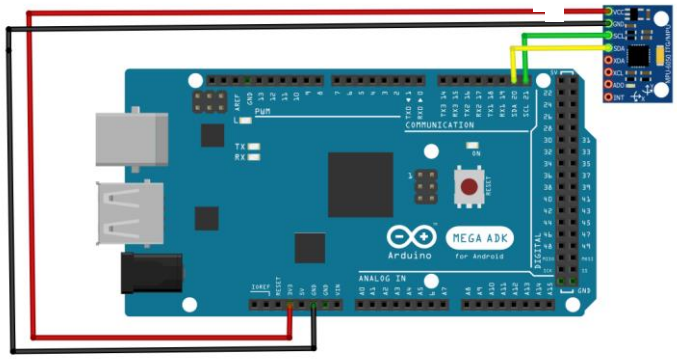


Figura 4.2.2: Schema de conectare a senzorului MPU6050 la Arduino Mega

Encoderele *AMT103-V* utilizează împreună 6 pini: alimentare 5V și GND, encoderul destinat motorului corespunzător axei Oy folosește pinii digitali 2 și 3, iar cel de al doilea se folosește de pini de comunicație 18 și 19. Pini 2, 3, 18 și 19 au o caracteristică necesară lucrului cu aceste encoderele, ei sunt destinați lucrului cu întreruperi externe. Întreruperile externe asigură gestionarea evenimentelor externe în timp real dar și economisirea timpului de procesare, acesta putând executa codul principal fără să mai verifice starea unor evenimente.

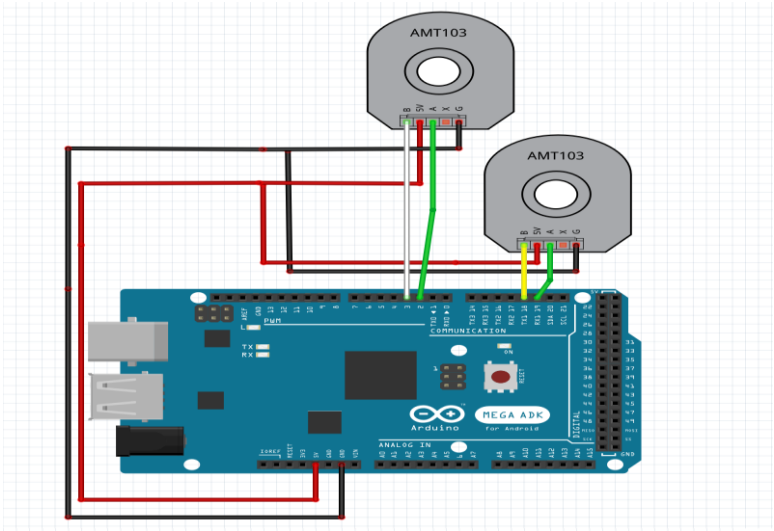


Figura 4.2.3: Schema de conectare a encoderlor AMT la Arduino Mega

Cele două drivere FOC ocupă 12 pini dintre care: 6 pini PWM (pulse width modulation), 4 pini digitali și 2 pini de alimentare. Primul shield, *SimpleFOCShield v2.0.4* ocupă 5 pini digitali PWM: 4, 5, 6, 8 și 9 dar capacitatea de modulare a lățimii semnalului este folosită doar de pinii 5, 6 și 9, pinul 8 fiind utilizat cu rol de enable iar pinul 4 fiind setat pe 0 (LOW). Pe lângă acești pini, acest driver ocupă și pinul de Vin și un pin de masă (prin care se realizează alimentarea microcontrolerului). Cel de al doilea driver, *SimpleFOCMini,* este de o dimensiune mai mică lucru ce se reflectă și în numărul scăzut de pini digitali 5: 3 dintre aceștia fiind utilizați ca PWM (10,11,12) iar pinul 7 având rolul de

enable și pinul 13 fiind setat pe 0. Acestea sunt alimentate de la o sursă de 12V, iar legătura cu motoarele BLDC se face prin intermediul celor 3 ieșiri, L1, L2 și L3 corespunzătoare celor trei faze ale motorului.

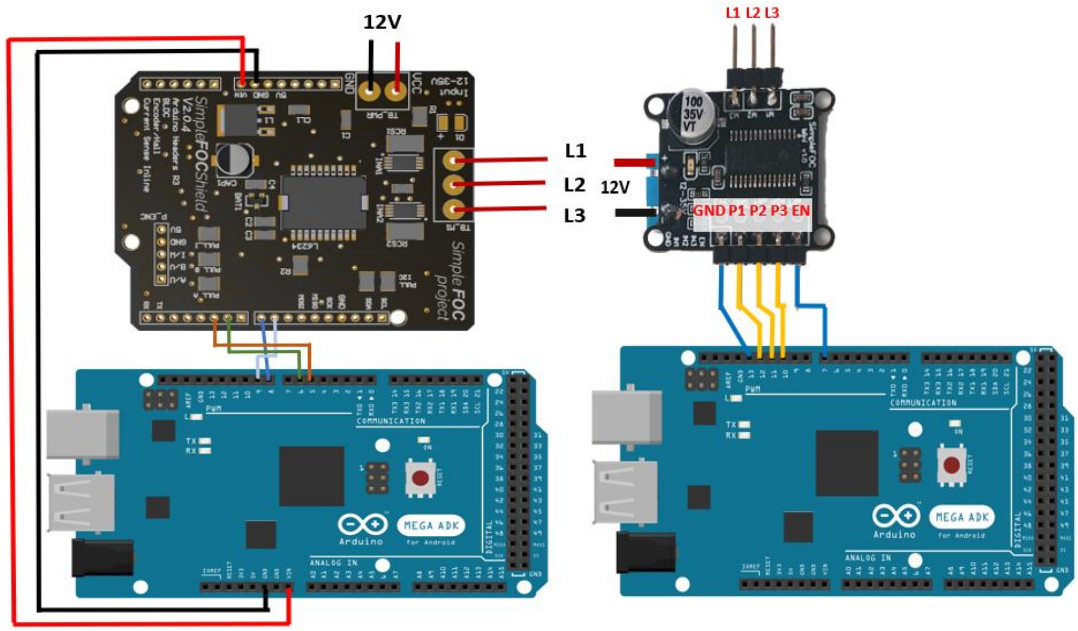


Figura 4.2.4 Schema conectării driverelor FOC și a motoarelor BLDC

4.3 Software embedded

Pentru programarea microcontrolerului Arduino Mega am utilizat un subset al libajului C++, denumit *Arduino programming language.* În cadrul acestui proiect am utlizat diferite biblioteci, ceele mai importante fiind:

- *MPU6050_light* : este o bibliotecă Arduino destinată comunicației cu un senzor de tip MPU6050. Aduce rezultatele măsurătorilor priviind : accelerația, giroscopice și temperatura. Unghiurile inițiale ale dispozitivului sunt calculate din datele brute obținute iar unghiurile de înclinare sunt calculate cu ajutorul unui filtru complementar dintre datele giroscopice și acelerometrice, obținând o acuratețe bună a estimării.
- *SimpleFOC* : etse o bibliotecă open-source pentru prima dată apărută în 2019, ea fiind înbunătățită și actualizată constant. Ea este destinată controlului precis al motoarelor în buclă închisă, utilizând senzori de poziție. O parte din caracteristicile ei sunt: controlul flexibil al motoarelor folosinf diverse metode (controlul PID, controlul cu câmp magnetic virtual), suportul pentru diferite tipuri de motoare (brushless, motoare liniare, pas cu pas,etc.)

Codul a fost editat și compilat utilizând mediul de dezvoltare Arduino IDE 2.0

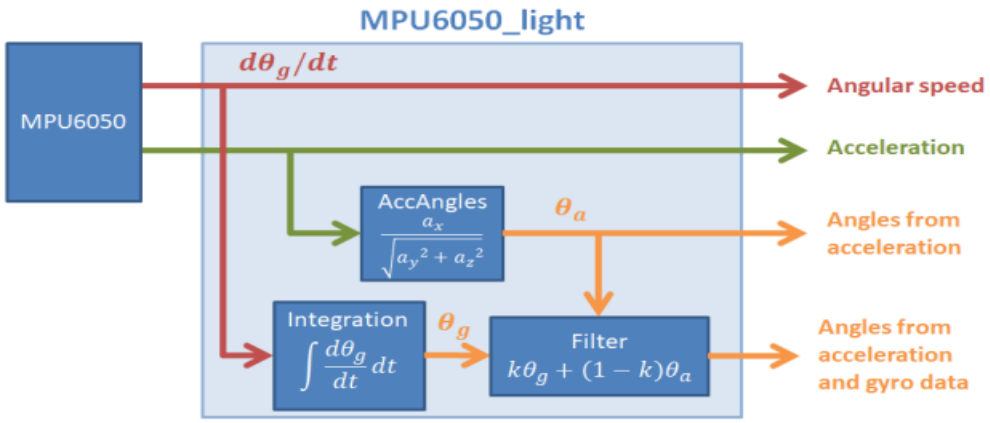
Obținerea datelor de la MPU6050

Conexiunea și inițializarea senzorului IMU este realizată asfel:

```
Wire.begin();
byte status = mpu.begin();
while(status!=0){ } // stop everything if could not connect to MPU6050
Serial.println(F("Calculating offsets, do not move MPU6050"));
delay(1000);
mpu.calcOffsets(true,true); // gyroscope and accelero
Serial.println("Done!\n");
```

Senzorul IMU furnizează trei viteze unghiulare și trei accelerații linerea corespunzătoare axelor sistemului cartezian Ox, Oy și Oz. El este de tip capacitiv, folosind variațiile capacitanței electrice dintre două plane de metal paralele datorate mișcării.

Biblioteca MPU6050_light pune la dispoziție, pe lângă valorile referitoare la accelerație și viteză unghiulară, și două moduri de obținere a unghiurilor: folosind datele accelerometrice sau prin combinarea rezultatelor obținute din datele giroscopice cu cele accelerometrice printr-un filtru complementar.



Figură 4.3.1 Prezentare generală a datelor accesibile prin biblioteca MPU6050_light [16]

Unghiul de înclinare, spre exemplu în raport cu axa Ox, este obținut prin integrarea datelor giroscopice $\dot{\theta}_x$ pe un interval scurt de tip δt *astfel:*

$$\theta_x(t+\delta t) \simeq \theta_x(t) + \dot{\theta}_x \cdot \delta t \quad (1)$$

Unghiurile de înclinare pe Ox și Oy pot fi obținute și din măsurătorile acceleraților conform următoarelor formule:

$$tan(\phi_x) = sgn(a_z)\frac{a_y}{\sqrt{a_x^2+a_z^2}} \quad (2)$$

$$tan(\phi_y) = \frac{a_x}{\sqrt{a_y^2+a_z^2}} \quad (3)$$

Valorile obținute prin intermediul accelerometrului sunt relevante și au o precizie cât mai bună în cazul mișcarilor lente (o frecvență mică) iar cele calculate cu ajutorul giroscopului prezintă precizie în cazul mișcărilor cu o viteză mărită.

Dar cele două tehnici prezintă și o serie de dezavantaje descrise în continuare. Pe de o parte, unghiul obținut din valorile accelerometrice este predispus la erori datorită imperfecțiunilor senzorului sau a unei mici erori de calibrare (eroare de offset) dar și erori cauzate de o accelerație neuniformă (mișcări bruște, vibrații). Pe cealaltă parte unghiul obținut prin măsurarea și integrarea valorilor accelerației unghiulare este susceptibilă la erori datorate fluctuațiilor de temperatură, derivei (viteza unghiulară înregistrată se schimbă în timp, chiar dacă în realitate nu există o mișcare a senzorului).

Aceste date sunt combinate utilizând un filtru complementar obținând cea mai bună precizie de estimare a unghiului de înclinare. Datorită mișcărilor neliniare și a vibrațiilor asociate tremuratului, am ales unghiul final să fie obținut din contopirea rezultatelor, giroscopul având o pondere de 90% din rezultatul final.

După ce am inițializat motoarele, inițializăm senzorul IMU prin stabilirea punctului la care se dorește stabilizarea dispozitivului în felul următor:

```
 int c=50;
while(c > 0){
  c--;
  mpu.update();
  Input_roll = mpu.getAngleX();
  Input_pitch = mpu.getAngleY();
  delay(20);
}
Setpoint_roll = Input_roll;
Setpoint_pitch = Input_pitch;
```

Apelăm de 50 de ori senzorul pentru a obține un unghi precis, datorită integrării accelerației unghiulare și salvăm rezultatele obținute.

În urma configurării și inițializării IMU am obținut conform graficelor de mai jos oscilații pentru măsurarea unghiului roll (± 0.04 grade) și pitch (± 0.03 grade)

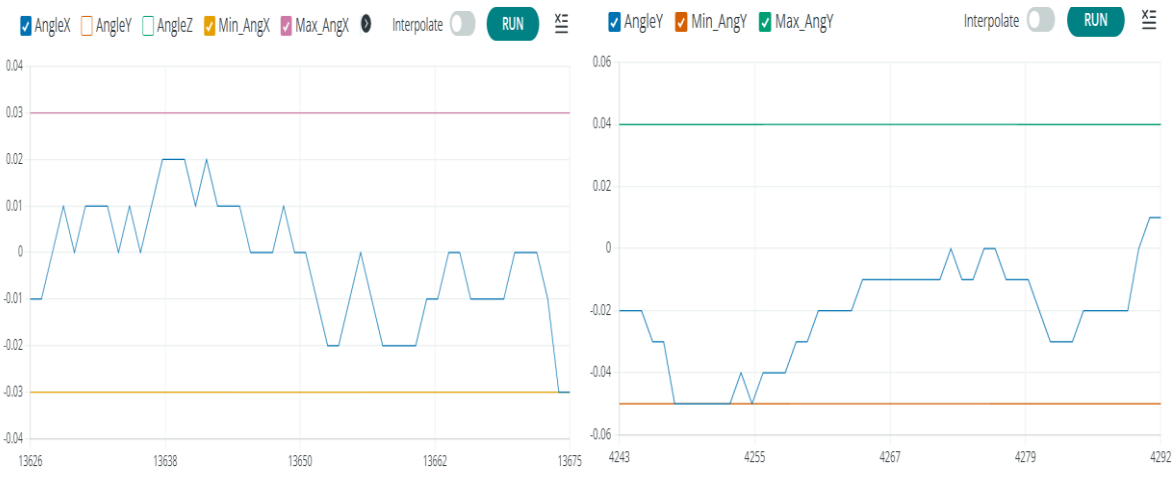


Figura 4.3.2 Oscilații unghi Roll și Pitch

Inițializarea și obținerea datelor necesare de la encodere 1pg

Pentru lucrul cu senzorii de poziție, în cadrul proiectului s-a folosit bilioteca SimpleFOC. Ea definește clasa ce modelează acest tip de encoder și inițializarea celor două encodere este descrisă în porțiunea de cod de mai jos:

```
Encoder encoder = Encoder(18, 19, 2048);
Encoder encoder2 = Encoder(2,3,2048);
```

Primul sensor, *encoder,* este destinat monitorizării primului motor (motorul corespunzător compensării pe axa Ox) iar cel de al doilea motorului ce manevrează brațul (stabilizarea pe axa Oy). Pentru inițializarea unui encoder avem nevoie de 2 pini ce suportă întreruperile hardware externe (în cazul nostrum fiind rămași nealocați doar pinii 2,3,28 și 19) corespunzătoare celor două canale A și B ale senzorului, ele înregistrând apariția și direcția mișcării.

Prin aceste 2 semnale, conform [17], se pot obține 4 stări posibile:

- *Înainte:* semnalul de pe canalul A este recepționat înaintea semnalului de pe canalul B (rotație în sensul acelor de ceasornic). O astfel de secvență este următoarea: A=0, B=0, A=1, B=0, A=1, B=1, A=0, B=1, A=0, B=0.
- *Înapoi*: semnalul de pe canalul B este recepționat înaintea semnalului de pe canalul A (sens contrar acelor de ceasornic).
- *Salt*: ambele semnale sunt active simultan, fapt ce indică o stare fixă sau foarte lentă.
- *Erori*: atunci când secvența semnalelor A și B este incorectă sau există erori în semnale.

Cel de al treilea parametru reprezintă numărul de impulsuri per rotație (PPR). Encoderele pot să lucreze în modul *Quadrature,* mod în care PPR este cvadruplat prin detectarea fiecărei modificări a semnalelor A și B, dar în cazul unei valori PPR mari, modul *Quadrature* devine prea complex pentru capabilitățile unei platforme de dezvoltare de tip Arduino.

Pe langă inițializarea celor două encodere, este nevoie și de definirea a două funcții ce vor implementa rutinile de tratare ale canalelor:

```
encoder.init();
encoder.enableInterrupts(doA, doB);
encoder2.init();
encoder2.enableInterrupts(doA2, doB2);
// interrupt routine intialisation
void doA(){encoder.handleA();}
void doB(){encoder.handleB();}
void doA2(){encoder2.handleA();}
void doB2(){encoder2.handleB();}
```

Biblioteca SimpleFOC mai pune la dispoziție și două funcții pentru lucrul în timp real cu acești senzori: *getVelocity*() – returnează viteza exprimată în radiani/secundă; *getAngle()* – returnează unghiul (radiani) la care se află motorul față de poziția inițială.

Controlul motoarelor utlizând driverul L298N

Ca primă încercare de control al unui motor BLDC și pentru exemplificare am abordat crearea unui control de tip ESC utilizând un driver L298N. Cu ajutorul acestui driver puteam comanda care din cele trei bobine să o alimentăm și astfel să se creeze un câmp magnetic rotativ.

În codul de mai jos este realizată rotirea unui motor BLDC la o viteză de aproximativ 12.5 rad/s în direcția setată.

```
int speed = 7;
void loop(){
  if(direction == 1  ){
      digitalWrite(phase1, HIGH); delay(speed);
      digitalWrite(phase3, LOW); delay(speed);
      digitalWrite(phase2, HIGH); delay(speed);
      digitalWrite(phase1, LOW); delay(speed);
      digitalWrite(phase3, HIGH); delay(speed);
      digitalWrite(phase2, LOW); delay(speed);
  }
  else{
      digitalWrite(phase2, HIGH); delay(speed);
      digitalWrite(phase3, LOW); delay(speed);
      digitalWrite(phase1, HIGH); delay(speed);
      digitalWrite(phase2, LOW); delay(speed);
      digitalWrite(phase3, HIGH); delay(speed);
      digitalWrite(phase1, LOW); delay(speed);
  }
}
```

Codul descris mai sus utilizeaza „*metoda celor 6 pași de modulare*” în care o bobină este încărcată pozitiv cu rolul de a atrage spre ea rotorul iar cea de dinaintea ei este încărcată negativ pentru a „împinge” rotorul spre următoarea bobină, între acestea fiind setată o întârziere pentru a permite rotorului să ajungă în apropierea bobinei.

Folosind aceeași configurație hardware, am realizat și o funcție ce permite mutarea motorului BLDC pas cu pas, 14 pași fiind maximul pe care l-am atins fără utilizarea controlului PWM. Aceasta este descrisă în umătorul paragraf:

```
void rotateMotor(){
  contor=contor+1;
  if(direction == 1){
   if(contor % 2 == 0){
      digitalWrite(phase1, HIGH); delay(speed);
      digitalWrite(phase3, LOW); delay(speed);
      digitalWrite(phase2, HIGH); delay(speed);
    }
    else{
      digitalWrite(phase1, LOW); delay(speed);
      digitalWrite(phase3, HIGH); delay(speed);
      digitalWrite(phase2, LOW); delay(speed);
```

```
    }
  }
  else{
    if(contor%2==0){

       digitalWrite(phase1, LOW); delay(speed);
       digitalWrite(phase2, HIGH); delay(speed);
       digitalWrite(phase3, LOW); delay(speed);
    }
    else{
       digitalWrite(phase1, HIGH); delay(speed);
       digitalWrite(phase2, LOW); delay(speed);
       digitalWrite(phase3, HIGH); delay(speed);
    }
   }
```

Datorită numărului mic de pași, 14, dispozitivul reacționează doar la o înclinare mai mare sau egală de 25,7 grade față de poziția inițială. Acest număr mic de pași este ineficient pentru cerințele dispozitivului, fapt ce a condus la optarea unei metode de control de tip FOC și al unui hardware ce permite implementarea acestui tip de control.

Principiul de funcționare al algorimului FOC

Algoritmul FOC este un algoritm utilizat în controlul motoarelor electrice de curent continuu și cu magnet permanent. Acest algoritm are rolul de a controla câmpul magentic și curentul într-un mod orientat pe câmp, pentru a obține performanțe ridicate de control utilizând în mod eficient resursele motorului.

El este compus din mai mulți pași, iar conform cu [18], cei mai importanți sunt următorii:

- *Măsurători și feedback* : curentul și poziția motorului sunt măsurate utilizând senzori adecvați (precum senzorii de curent și encoderele). Aceste măsurători sunt esențiale pentru a obține informații precise despre starea motorului și pentru a furniza feedback în timp real.
- *Transformarea Clarke și Park* : după ce au fost obținute măsurătorile curentului pentru cele trei faze, rezultatele sunt transformate în coordonatele sistemului statoric prin transformarea Park și Clarke. Transformarea Clarke convertește curentul trifazic în componenta directă I_d și în cvadratură I_q. Transformarea Park rotește aceste componente în funcție de unghiul de poziție al rotorului.
- *Controlul poziției și al vitezei*: concomitent cu controlul curentului, este implementat un controler de poziție și viteză care monitorizează și menține poziția și viteza dorită a motorului. Acest controler utilizează feedback-ul privind poziția și viteza obținute de la senzori și utilizează algoritmi de control, un exemplu fiind controlul PID (Proporțional, Intregral și Derivativ), pentru ajustarea curentului
- *Modulare PWM (pulse witdth modulation)* : după ce sunt calculate și stabilite valorile curentului direct și în cvadratură, se utilizează tehnica modulării PWM pentru a generea semnale de comandă pentru etapele de putere ale motorului.

- *Interație și ajustare continuă*: Algoritmul FOC, funcționează în mod ciclic, actualizând și ajustând constant valorile curentului și controlul motorului. La fiecare iterație, se obțin noi măsurători și se aplică algoritmii de control amintiți mai sus pentru a asigura o funcționare precisă și stabilă a motorului.

Transformarea Clarke și Park

„Transformarea Park convertește un vector dintr-un sistem echilibrat și staționar cu două faze ortogonale într-un cadru de referință ortogonal și rotativ.” [21]

Transformarea Clarke, cunoscută și ca transformarea în coordinate trifazate, are scopul de a converti un set de coordinate tridimensionale într-un alt set de coordonate bidimensionale. Această transformare ia în considerare amplitudinea și fazorul componentelor de fază (*a, b, c*) și calculează două componente (*αβ),* ce reprezintă proiecțiile sistemului trifazat pe un plan orizontal.

Transformarea Clarke este exprimată de următoarele ecuații:

$$I_\alpha = \frac{2}{3}(I_a) - \frac{1}{3}(I_b - I_c) \qquad (4)$$
$$I_\beta = \frac{2}{\sqrt{3}}(I_b - I_c)$$

Iar transformarea inversă, de ecuațiile:

$$V_a = V_\alpha$$
$$V_b = \frac{-V_\alpha + \sqrt{3}V_\beta}{2} \qquad (5)$$
$$V_c = \frac{-V_\alpha - \sqrt{3}V_\beta}{2}$$

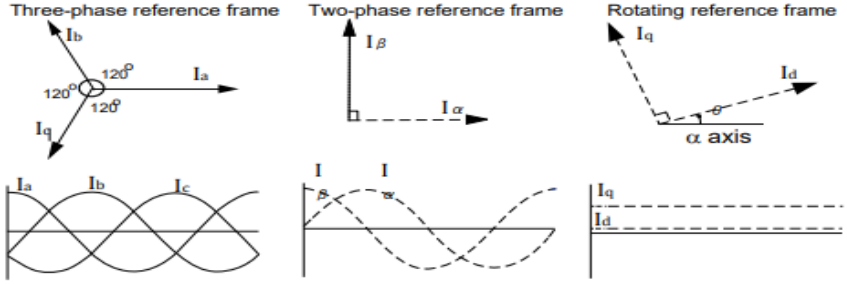
Transformarea Park este aplicată pe rezultatele obținute în urma transformării Clarke și este redată de ecuațiile:

$$I_d = I_\alpha \cdot cos(\theta) + I_\beta \cdot sin(\theta) \qquad (6)$$
$$I_q = I_\beta \cdot cos(\theta) - I_\alpha \cdot sin(\theta)$$

Iar transformarea inversă se folesște de:

$$V_\alpha = V_d \cdot cos(\theta) - V_q \cdot sin(\theta) \qquad (7)$$
$$V_\beta = V_d \cdot cos(\theta) + V_q \cdot sin(\theta)$$

Unde θ reprezintă unghiul de rotație.



Figură 4.3.4 Cadrele de referință în transformarea Park și Clarke [21]

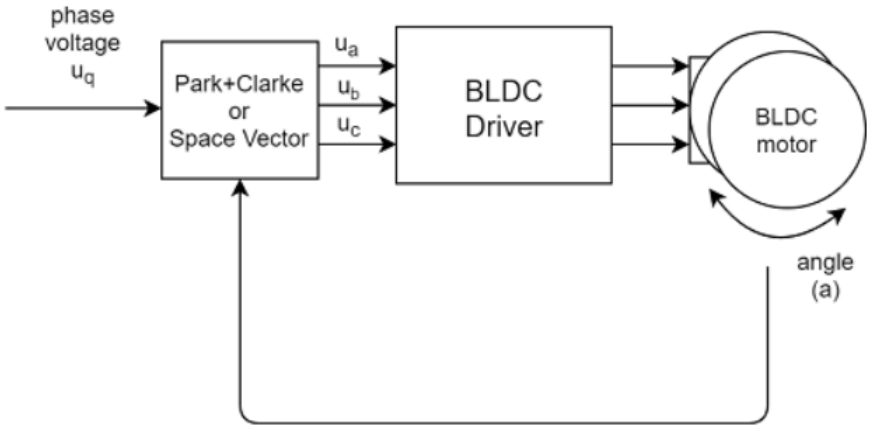
În cadrul SimpleFOC, algoritumul este bazat pe controlul motorului prin ajustarea tensiunii pe fiecare fază. Aceste tensiuni crează câmpul magnetic din statorul motorului, câmp ce se află la exact 90 de grade „în spatele” câmpului magnetic creat de magneții permanenți ai rotorului, creeând astfel efectul de „împingere”.

Necesitatea unghiului de 90° dintre rotor și stator este dată de ecuația forței electrice generate asupra unui conductor de lungime *L [m]*, prin care trece un curent electric de intensitate *I* [A] și care se găsește într-un câmp magnetic *B* [A/m]:

$$F = B \cdot I \cdot L \cdot sin(\alpha) \tag{8}$$

Unde α reprezintă unghiul dintre câmpul magnetic *B* și curentul *I*. Pentru a obține o valoare maximă pentru această forță, trebuie să menținem unghiul α la o valorea cât mai apropiată de 90°.

Biblioteca SimpleFOC pune la dispoziție două metode de modulare pentru obținerea tensiunilor aplicate celor trei faze ale motorului: *Sinusiodal PWM* și *Space Vector PWM.*



Figură 4.3.5 Algoritmul FOC din cadrul SimpleFOC [20]

În cazul *Sinusiodal PWM* se utilizează cele două transformări aminitite mai sus.

În cel de al doilea caz, *Space Vector PWM*, rezultatul pentru cele trei faze este calculat în doi pași. În primul pas se identifică sectorul (unghiul de 360° este împărțit în 6 sectoare egale) și se calculează trei perioade T_0, T_1, T_2. Perioadele 1 și 2 indică cât de mult fazele 1 și 2 ale motorului ar trebuii să fie active, iar perioada 0 indică cât de mult ar trebuii să menținem o tensiune de 0V în motor.

$$\begin{aligned} s &= \left\lfloor \frac{3}{\pi}\theta \right\rfloor + 1 \\ T_1 &= \sqrt{3} sin\left(s\frac{\pi}{3} - \theta\right) \\ T_2 &= \sqrt{3} sin\left(\theta - (s-1)\frac{\pi}{3}\right) \\ T_0 &= 1 - T_1 - T_2 \end{aligned} \tag{9}$$

Pasul al doilea constă în proiecția valorilor perioadelor la ciclurile de lucru adecvate confrom tabelului următor:

Tabel 4.3.1. Valorile tensiunilor de ieșire în funcție de sector [20]

Sector	T_a	T_b	T_c
1	$T_1 + T_2 + T_0/2$	$T2 + T0 / 2$	$T0 / 2$
2	$T1 + T0 / 2$	$T_1 + T_2 + T_0/2$	$T0 / 2$
3	$T0 / 2$	$T_1 + T_2 + T_0/2$	$T2 + T0 / 2$
4	$T0 / 2$	$T1 + T0 / 2$	$T_1 + T_2 + T_0/2$
5	$T2 + T0 / 2$	$T0 / 2$	$T_1 + T_2 + T_0/2$
6	$T_1 + T_2 + T_0/2$	$T0 / 2$	$T1 + T0 / 2$

Cele două tehnici aduc fiecare avantaje și dezavantaje. Modularea sinusoidală este o tehnică relativ simplă pe când Space Vector (SV) implică calcule mai complexe. Iar în termeni de performanță SV oferă o mai bună eficiență energetică.

Analizând graficele de mai jos obținute pentru tensiunea U_q = 0.5 V, observăm: în cazul modulării sinusoidale magnitudinea atinge exact valoarea 1, iar în cazul SV o undă sinusoidală dublă a cărei amplitudine este de aproximativ 1.15 ori mai mică, lucru ce permite setarea unei tensiuni mai mari pentru faza motorului (aprox. $U_q = 0.58$) conduce la o putere cu 15% mai mare utilizînd aceași sursă.

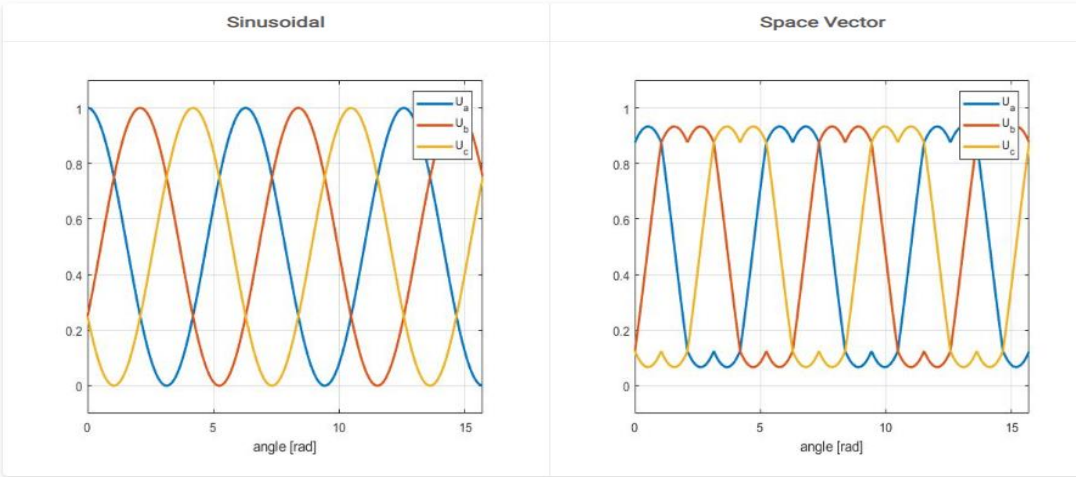


Figura 4.3.6 Semnalele de comandă folosind tehnici diferite de modulare [20]

Astfel, tensiunea maximă ce poate fi livrată pe fazele motorului pentru cazul sinusoidal este $U_q = 0{,}5 \times V_{susă}$, iar în cazul SV $U_q = 0{,}58 \times V_{sursă}$. În cazul depășirii acestor valori semnalul se va satura la tensiunea sursei de alimentare iar mișcarea motorului va deveni neliniară și nu va mai fi lină, fără a aduce o creștere suplimentară a puterii motorului.

Control cu SimpleFOCShield

În cadrul proiectului s-au folosit 2 motoare și 2 drivere, aceste sunt definite astfel:

```
BLDCMotor motor = BLDCMotor(7);
BLDCDriver3PWM driver = BLDCDriver3PWM(9,5,6,8);
BLDCMotor motor2 = BLDCMotor(7);
BLDCDriver3PWM driver2 = BLDCDriver3PWM(10,11,12,7);
```

Motoarele utizate au 14 poli magnetici, iar în definirea unui motor în cadrul SimpleFOC trebuie furnizat numărul de perchi de poli (7 în cazul nostru). După care este definit driverul ce va controla motorul. Unui driver îi sunt necesari 3 pini PWM și un pin utilizat cu rolul de enable.

Pasul următor constă în inițializarea celor două componente:

```
driver.voltage_power_supply = 12;
driver.init();
motor.linkDriver(&driver);
motor.linkSensor(&encoder);
motor.foc_modulation = FOCModulationType::SinePWM;
motor.controller = MotionControlType::angle;
motor.useMonitoring(Serial);
motor.voltage_sensor_align = 8;
motor.voltage_limit = 8;
motor.init();
motor.initFOC();
```

Driverele sunt alimentate de la o sursă de 12V și după setarea acestui parametru se atașează motorului corespunzător, împreună cu senzorul de poziție. După care sunt setate tipul de modulație folosit în cadrul algorimului FOC și modul de control al motorului.

Clasa *BLDCMotor* pune la dispoziție și funcționalitatea de monitorizare a varibilelor în timp real prin intermediul Arduino IDE serial plotter: *target* (specific modului de control – poziția, viteza setată) precum și viteza și unghiul actual al motorului. Acest lucru fiind util în cazul depanării și setării parametrilor motorului, dar după ce au fost realizate modficările, este recomandat să nu se mai utilizeze, deoarece monitorizarea atrage după sine scăderea frecvenței de eșantionare a algoritmului FOC, datorită viezei mici de comunicație prin interfața serială.

În finalul inițializării sunt setate limitele de tensiune pentru controlul motorului.

Comunicarea prin portul serial

Pentru a putea modifica parametrii de funcționare ai algorimului FOC și al motoarelor, SimpleFOC pune la dispoziție clasa *Commander* iar la o instanță a acestui obiect îi pot fi atașate mai multe funcții ce modifică parametrii necesari.

Mai jos este prezentat codul prin care se instanțiază un obiect de tip *Commander* ce comunică utilizând portul serial, și căruia îi sunt atașate două funcții ce modifică parametrii regulatorului PID.

```
Commander command = Commander(Serial);
void doTarget_AP(char* cmd) {
    float ap=0;
    command.scalar(&ap, cmd);
    Serial.println("M Angle P");
    motor.P_angle.P=ap;
}
void doTarget_AI(char* cmd) {
     float ai=0;
     command.scalar(&ai, cmd);
     Serial.println("M Angle I");
     motor.P_angle.I=ai;
 }
```

```
// in setup()
 command.add('p', doTarget_AP, "target ap");
 command.add('i', doTarget_AI, "target ai");
```

Comenzile se înregistrează printr-un caracter, iar la apelarea acelei comenzi se va specifica caracterul corespunzător comenzii și noua valoare a parametrului (de ex: p20 va seta componenta proporțională la valoarea 20).

Cod de testare și calibrare motor

Pentru calibrarea și testatrea configurației motorului și driverului putem determina numărul de poli ai acestuia utilizînd algoritmul descris în continuare:

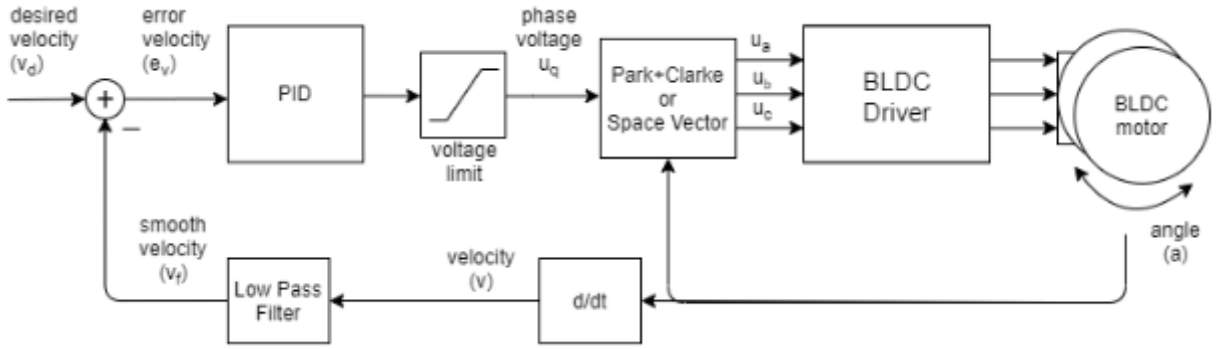
```
motor.controller = MotionControlType::angle_openloop;
motor.move(0);
_delay(1000);
encoder.update();
float angle_begin = encoder.getAngle();
_delay(50);
float motor_angle = 0;
while(motor_angle <= pp_search_angle){
  motor_angle += 0.01f;
  motor.move(motor_angle);
  _delay(1);
}
_delay(1000);
encoder.update();
float angle_end = encoder.getAngle();
_delay(50);

int pp = round((pp_search_angle)/(angle_end-angle_begin));
```

Algoritmul începe prin setarea controlului motorului ca fiind de tip unghi, fără senzor de poziție și mutarea motorului în poziția 0. În continuare se citește unghiul motorului prin intermediul encoderului și se începe mutarea lină a motorului până la un unghi setat (în cazul nostru $2 \cdot \pi$) după care se înregistrează unghiul motorului cu ajutorul encoderului. Rezultatul, numărul de perechi de poli este obținut prin rotunjirea raportului dintre unghiul setat și diferența dintre unghiurile măsurate de encoder.

Controlul unghiului motoarelor

Controlul poziției în timp real este excutat cu ajutorul funcției *.move(target)* ce excută sarcina specifică modului de operare al motorului.

Pentru motoare a fost ales modul de control bazat pe unghiul față de poziția inițială, utilizând senzorul de poziție. Dar pentru a obține un control de tip unghi, mai întâi trebuie să setăm viteza la care dorim să lucreze motorul. Controlul vitezei mișcarii este descris de diagrama următoare:



Figură 4.3.7 Diagrama de control a vitezei motorului [20]

Procesul de reglare al vitezei utilizează un regulator de tip PID și un filtru de tip trece-jos. Regulatorul de tip PID – *P = proporțional, I = integrator și D = derivator,* reprezintă un algoritm de control utlizat în sistemele de reglare pentru a menține o variabilă de ieșire la o vaaloare dorită. Cele trei componente ale acestuia acționează în mod diferit față de eroare înregistrată. Componenta *proporțională* reacționează proporțional cu eroare înregistrată la momentul actual, cu cât eroare e mai mare, cu atât răspunsul este mai mare. Componenta *integratoare*, integreză eroare în timp și produce un semnal de ieșire proporțional cu suma erorilor anterioare dintr-un interval de timp, ea fiind folosită pentru a corecta erorile procesului de reglare atunci când componenta proporțională nu reușete să reducă eroare la zero și pentru eliminarea erorilor de reglare în regimul sațional. Ultima componentă, cea *derivativă*, măsoara rata de schimbare a erorii și produce un semnal de ieșire proporțional cu aceasta. Ea este folosită pentru anticiparea și reducerea oscilațiilor sitemului de reglare.

Acest regulator este descris de formula următoare:

$$u(t) = K_p \cdot e(t) + K_i \cdot \int_0^t e(\tau)d\tau + K_d \cdot \frac{d}{dt}e(t) \qquad (10)$$

Rezumând cele descrise în paragraful anterior, componenta proporțională acționeză la eroarea actuală, cea integrală ia în considerare valorile din trecut ale erorii iar cea derivativă anticipează viitoarele erori. Ele se completeză una pe cealaltă, componenta proporțională face față în cazul unei erori mari, dar când aceasta este mică acuratețea scade, aici intervenind componenta integratoare și ajutând la compensare, dar dacă dorim ca valoarea stabilită să fie atinsă într-un mod cât mai rapid, componenta derivativă ne este

de folos, ea acționând precum o „frână” [22]. Aceste trei componente au diferite ponderi în rezultatul regulatorului. Stabilirea acestor ponderi reprezintă procesul prin care ieșirea se adaptează la cerințele impuse.

Cele trei ponderi infulențează în mod diferit comportamentul regulatorului. O constantă de proporționalitate K_p prea mare poate duce la o depășire excesivă a punctului dorit (overshoot), instabilitate (în loc să ajungă la valoarea dorită, ieșirea poate oscila și crește în mod continuu conducând la un comporatament imprevizibil) și oscilații. Dar și o valoare prea mică afectează în mod negativ performanțele sistemului, o astfel de valoare implicând: eroare de reglare statică (steady-state-error) [23], ieșirea nemaiputând atinge valoarea dorită; răspuns lent; reacție insuficientă.

În mod asemnător și constanta de integrare K_i afecteză regulatorul. Valorea prea mare producând: oscilații și instabilitate, stabilitate deficitară în prezența zgomotelor și a pertubărilor dar și un răspuns lent în cazul schimbărilor rapide ale erorii sau a setpointului. Iar în cazul unei valori prea mici, procesul de reglare fiind încetinit și poate conduce la o eroare persitentă în regimul staționar.

Constanta de derivare K_d adecvată poate anticipa și reduce overshoot-ul și oscilațiile cauzate de variații bruște ale erorii. Cu toate acestea, valoarea prea mare a acesteia conduce la oscilații din cauza zgomotelor.

Procesul de ajustare al acestor constante este cunoscut sub denumirea de *PID parameters tunning* și are scopul de a obține un control optim al sistemului în funcție de caracteristicile și cerințele acestuia . Conform [24] există o multitudine de metode și algoritmi de tunare PID, unele dintre acestea fiind: *metoda încercărilor și erorilor* : metodă ce implică ajustarea manuală a valorilor în funcție de observarea și evaluarea răspunsului sitemului în timp real, valoriile fiind ajustate iterativ până când se obține un răspuns dorit, având în vedere criterii precum timpul de stabilizare, precizia sistemului și overshoot-ul; *metoda Ziegler-Nichols* ce propune o abordare sitematică bazată pe metodele de încercări și erori, *metode bazate pe model* acestea utlizând un model matematic al sistemului pentru a estima și ajusta valorile constantelor. Aceste metode de ajustare sunt un proces iterativ și poate necista ajustări suplimentare pentru a obține un control optim.

În cadrul proiectului, am utilizat *metoda Ziegler-Nichols* ce se baează pe experimentarea și observarea sistemului de control în fața unor perturbații sau comenzi de intrare. Aceasta implică o succesiune de pași:

- Stabilirea celor trei constante la zero.
- Creșterea constantei de proporționalitate K_P în mod gradual până când sistemul intră într-o oscilație continuă (oscilațiile devin persitente și constante) și se denumește în continuare punct critic K_C. Acest lucru marchează faptul că sistemul a devenit instabil și nu se poate stabiliza pe termen lung datorită unei constante de proporționalitae suficient de mari.
- Măsurarea perioadei oscilației: se măsoară perioada de timp a oscilației, ea fiind denumită în continuare *perioadă critică* T_C.
- Calculul constantelor regulatorului PID: acestea se calculeză conform formulelor specifice din tabelul următor în funcție de perioada critică (T_C) și punctul critic (K_C).

Tabel 4.3.2 Formule de calcul metoda Ziegler-Nichols [25]

Tipul de control	K_P	K_I	K_D
P	$0.5 \cdot K_C$	-	-
PI	$0.45 \cdot K_C$	$0.54 / T_c$	
PD	$0.8 \cdot K_C$	-	$0.1 \cdot T_C$
PID	$0.6 \cdot K_C$	$2 / T_C$	$0.125 \cdot T_C$

Această metodă oferă o modalitate simplă și practică de obținere a valorilor inițiale ale parametrilor controlerului PID. Cu toate acestea, valorile sunt utile ca punct de start, ele necesitând ajustări ulterioare pentru obținerea unui control optim și stabil, în funcție de cerințele specifice aplicației.

În cadrul regulatorului PID pentru controlul vitezei motoarelor am utlizat metoda încercărilor și a erorilor, iar în urma acesteia am obținut următorele valori pentru compenente: $K_P = 0.2$, $K_I = 20$ *și* $K_D = 0.005$.

Pe langă valorile componentelor PID, mai putem ajusta precizia și finețea mișcării utilizând și filtrul de tip trece jos și prin numărului de volți pe secundă permiși pentru voltajul aplicat motorului să se schimbe (output_ramp).

```
motor2.PID_velocity.output_ramp = 1000;
motor2.LPF_velocity.Tf = 0.01f;
```

Odată ce a fost configurată viteza de răspuns a motorului, trebuie calibrat și algorimtul de control al unghiului, aceasta calibrare realizându-se precum este definită mai jos:

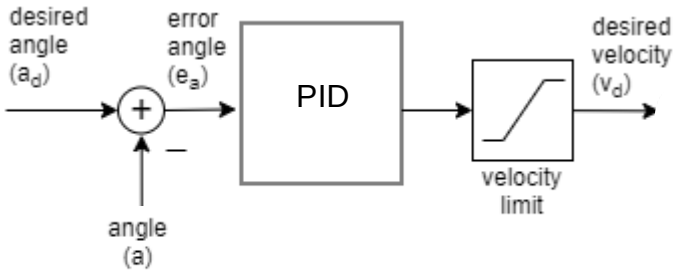


Figura 4.3.8 Diagrama bloc de control al unghiului[20]

Viteza maximă a motoarelor este limitată la valorea de 25 rad/s.

În secțiunea următoare descriu modul de aplicare al tehnicii Ziegler-Nichols pentru controlul PID al unghiului unuia dintre motoare.

În urma aplicării primilor doi pași, pornind de la un pas de 5 pentru constanta de proporționalitate, m-am apropiat de punctul critic, atunci caând a fost atinsă o valoarea 25 sistemul a intrat în oscilație continuă, pentru a afla o valoare apropiată de cea minimă pentru punctul critic am scăzut valoarea constantei cu un pas de 0.5 și valoarea cea mai mică pentru care sistemul oscila continuu a fost cea de $K_C = 20$. Pentru această valoare obținem formele de undă descrise în imaginea următoare:

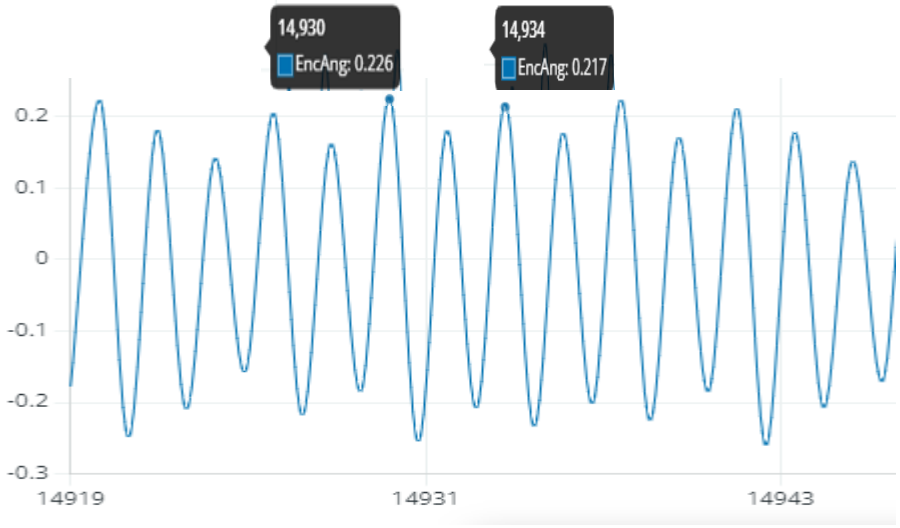


Figura 4.3.9 Oscilații ale poziției motorului pentru punctul critic K_C

Din graficele anterioare, ținând cont că citirea valorilor encoderului se face la o frecvență de 25 ms, perioada oscilațiilor conține 4 citiri succesive, adică $T_C = 100$ *ms.*

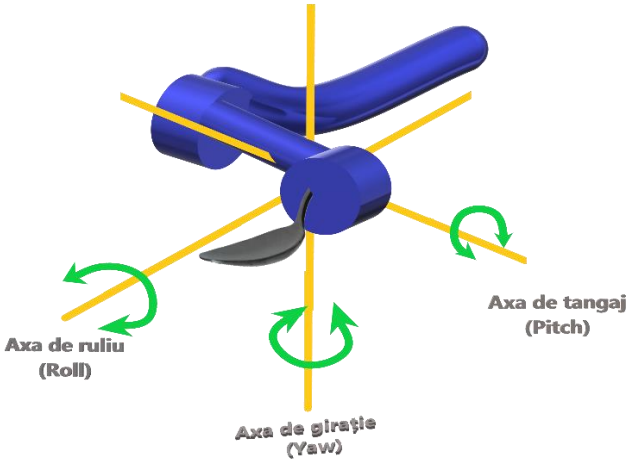
Utilizând formule descrise în anterior obținem următoarele valori pentru componente: $K_P = 12$, $K_I = 20$ *și* $K_D = 0.0125$.

Și pentru modul de control setăm un filtru de tip trece-jos ($T_f = 0.01$) dar și o limită pentru accelerație (prin output_ramp = 1750 rad/s^2).

În mod similiar, s-a procedat și pentru configurarea și celui de al doilea motor.

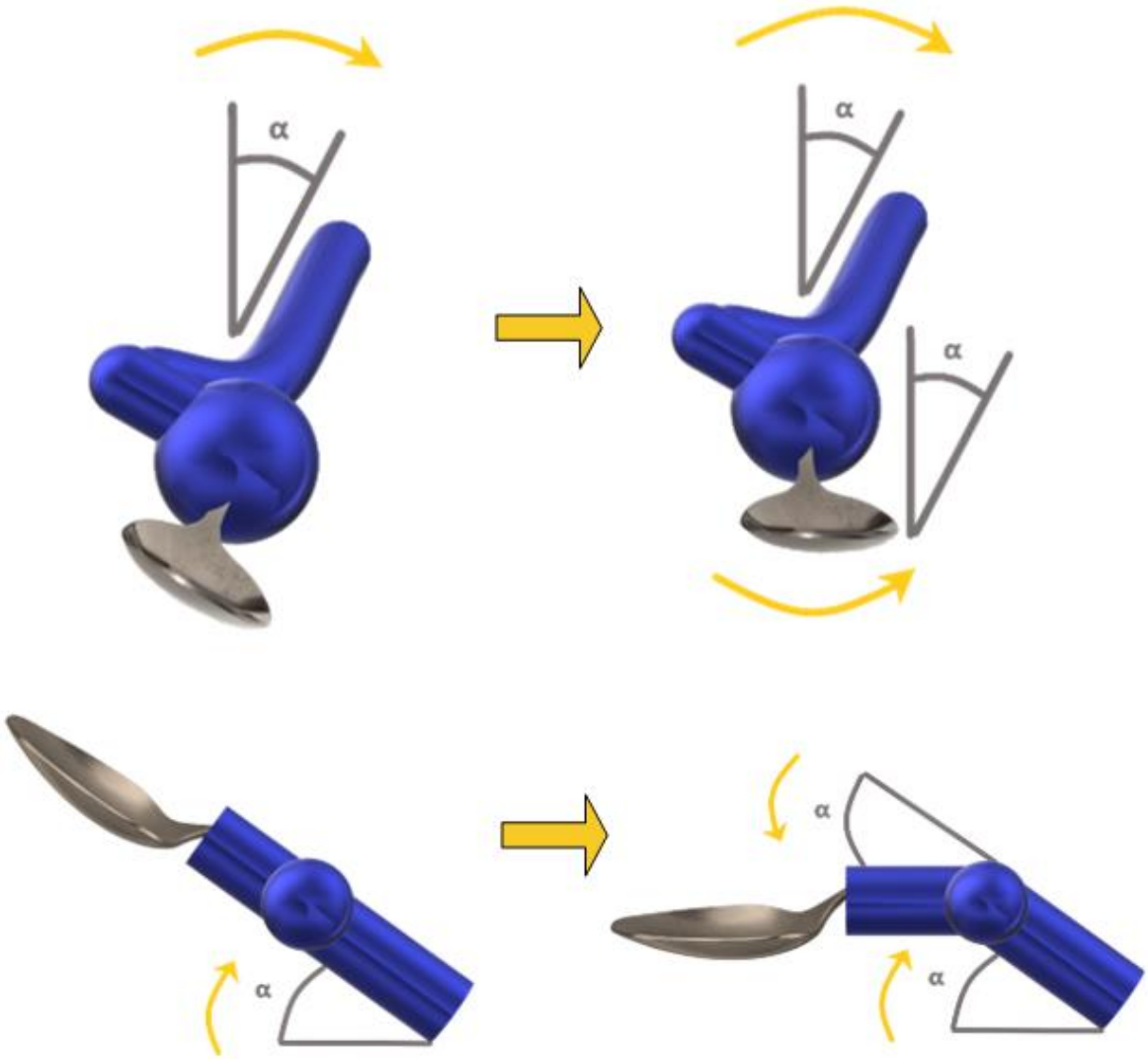
Algoritmul de stabilizare 3 pg

Tacâmul atașat dispozitivului trebuie să își mențină poziția orizontală, în mișcarea pe două axe, acestea fiind axa de tangaj (Pitch) și axa de ruliu (Roll) reprezentate în *Fig.4.3.9*. Două motoare vor fi poziționate ortogonal pentru a asigura sistemului cele două grade de libertate. Cu o astfel de configurație, dispozitivul este destinat menținerii tacâmului într-o poziție orizontală.



Figură 4.3.9 Axele de libertare ale dispozitivului

Algoritmul propune stabilizarea pentru 2 axe în modul reprezentat în figurile de mai jos :



Figură 4.3.10 Poziționare articulați și motoarei dispozitiv

Algoritmul implementat măsoară unghiul de înclinație pentru o axă α și acționează în direcția opusă prin intermediul motoarelor cu un unnghi de valoare egală.

În cadrul algoritmului implementat, după inițializarea, configurarea componentelor și aflarea poziției inițiale, se apelează la o frecvență aproximativ de 1 KHz (frecvența maximă la care poate fi rulat algoritmul SimpleFOC pe plăcile de dezvoltare Arduino) funcțiile vitale pentru controlul motoarelor utilizând algorimtul FOC și pentru citirea valorilor furnizate IMU (necesară pentru integrarea valorilor giroscopice, folosite în calculul unghiurilor de înclinare). În continuare, se verifică la fiecare 15 milisecunde (la o frecvență de aproximativ 66,7 Hz), starea motoarelor. Prin urmăriea stării motoarelor, se verifică în primul pas dacă viteza motorului este mai mică decât o valoare stabilită (*Pitch = 1 rad/s, Roll = 2.5 rad/s)* și dacă diferența dintre unghiul actual al motorului și unghiul necesar

compensării este mai mică decât o valoare stabilită pentru fiecare motor în parte (avem aceste valori *Pitch = 0.025 rad, Roll = 0.035 rad).*

Următorul pas din algoritmul propus, îl reprezintă calculul erorilor pentru cele două axe. Eroare este obținută prin diferența față de valoare la poziția inițială, dar dacă acestă diferență este inclusă într-un interval specific fiecărei axe, ea va fi neglijată pentru a reduce zgomotele și oscilațiile datorate senzorilor. Mai jos este descrisă funcția de calcul al erorii pentru axa de tangaj, menționând că valorile returnate sunt exprimate în grade sexagesimale:

```
double diff_pitch(double x) {
  double dif = 0;
  dif = x - Setpoint_pitch;
  if (dif < 0.5 && dif > -0.5)
    return 0;
  return dif;
}
```

O dată primită valoarea erorii, ea este convertită din grade sexagesimale în radiani (1 rad ≈ 57,2968˚) și se alege pentru compensare axa ce prezintă cea mai mare eroare. În pasul următor se trece la calcularea unghiului de compensare. Acest unghi este limitat la un interval datorită componentelor fizice, mai jos fiind expuse porțiunile ce limitează unghiul:

```
float max_move_roll = 0.7;
float max_move_pitch = 0.5;
float min_move_pitch = -0.4;
target_angle_roll += move_roll;
if (target_angle_roll > max_move_roll) target_angle_roll = max_move_roll;
if (target_angle_roll < -max_move_roll) target_angle_roll = -max_move_roll;
target_angle_pitch += move_pitch;
if (target_angle_pitch > max_move_pitch) target_angle_pitch = max_move_pitch;
if (target_angle_pitch < min_move_pitch) target_angle_pitch = min_move_pitch;
```

Ca ultim pas se trimite comanda cu unghiurile de compensare motoarelor și algoritmul este reluat în mod ciclic până la oprirea alimentării.

Întreg procesul de stabilizare este ilustrat în Figura 4.3.11.

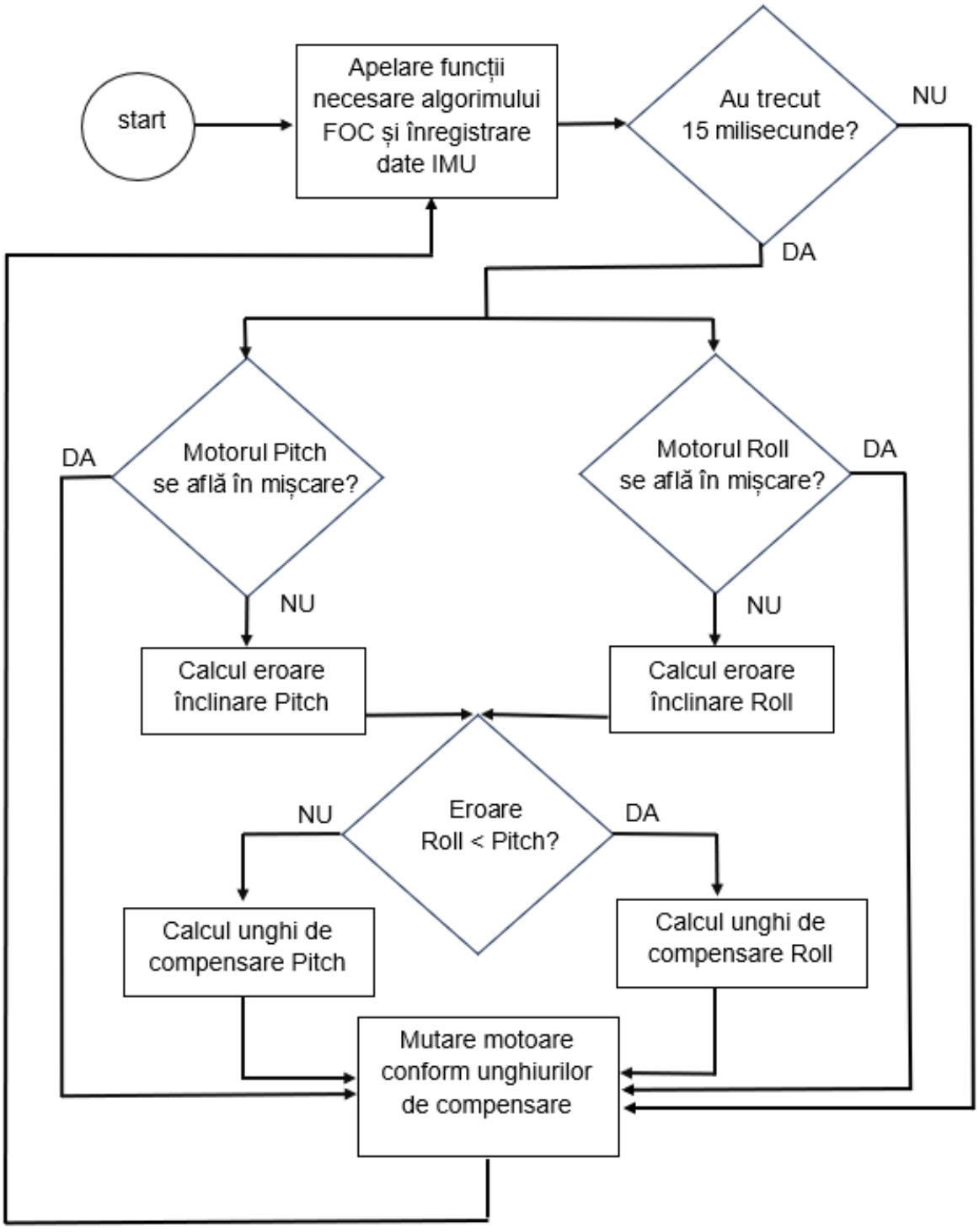


Figura 4.3.11 Procesul de stabilizare pentru cele două axe

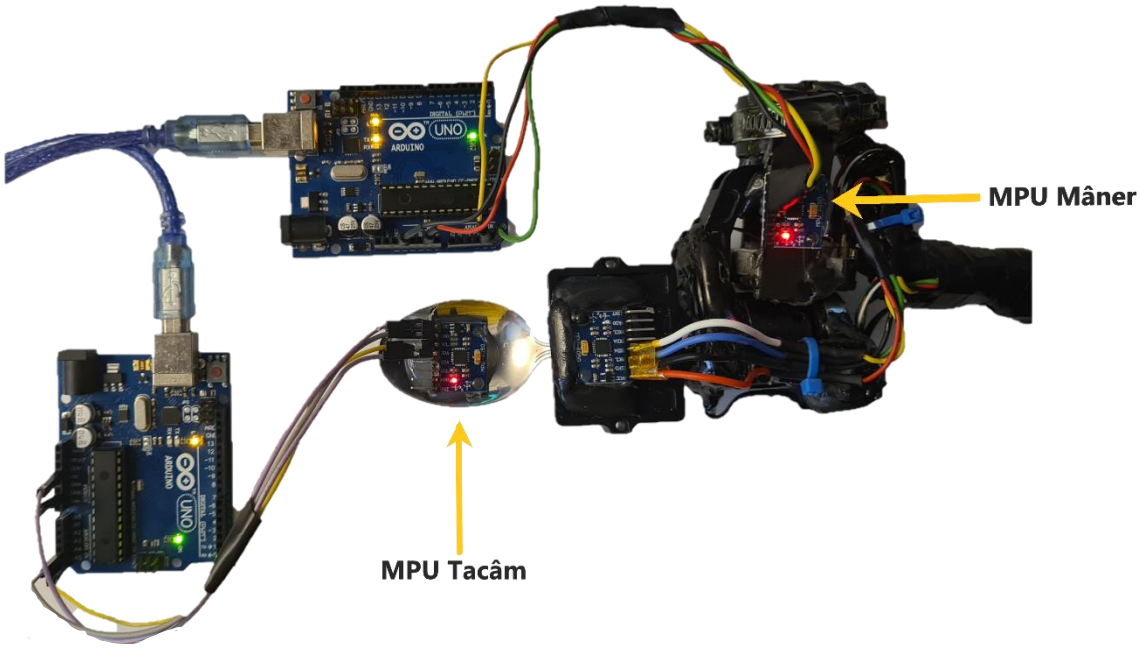
5. TESTAREA DISPOZITIVULUI

Testarea și aflarea procentajului de amortizare

Pentru a efectua testarea dispozitivului am efectuat mai multe teste.

Pentru testarea capacității de amortizare am realizat configurația prezentată în Figura 5.1. Ea constă în plasarea unui senzor MPU6050 pe mânerul dispozitivului pentru a monitoriza tremuratul mâinii. Un al doilea senzor MPU6050 a fost plasat pe suprafața tacâmului. Fiecare senzor este comandat de un microcontroler Arduino UNO și ele furnizează datele prin intermediul portului serial. Monitorizarea și salvarea datelor transmise prin intermediul portului serial s-a realizat cu ajutorul aplicației *CoolTerm [coolterm.en.lo4d.com, 2023].* Aceste date au fost folosite apoi în *Matlab [mathworks.com, 2023]* pentru crearea graficelor de vizualizarea a comportamentului dispozitivului.

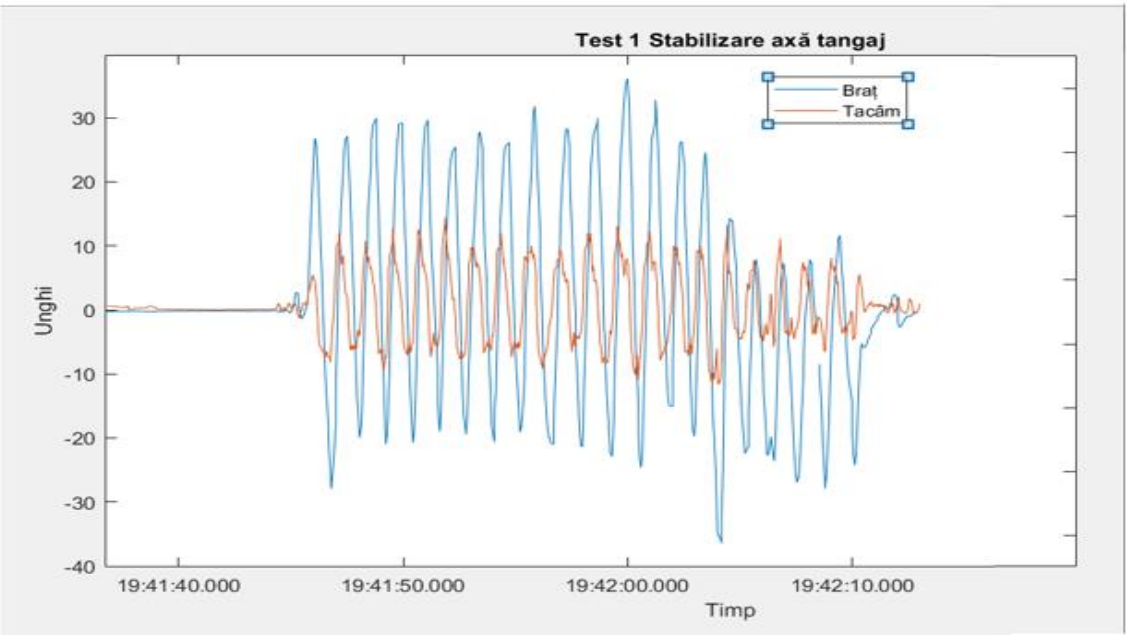
În cadrul primelor teste efectuate, dispozitivul prezenta: o amortizare mică, răspuns lent la înclinări și vibrații nedorite. Acești parametri nesatisfăcători au fost cauzați de proiectarea inițială a regulatoarelor PID ale motoarelor. După aplicare tehnicii descrise în capitolul anterior și căutare prin metoda încercărilor și a erorilor (în jurul valorilor obținute prin prima tehnică), răspunsul dispozitivului s-a înbunătățit iar rezultatele sunt descrise în continuare:



Figură 5.1 Arhitectura instalației de testare a dispozitivului cu două Arduino UNO și doi MPU6050

Utilizând această configurație am efectuat două teste ce au constat în înclinarea dispozitivului, fără sarcină suplimentară, pe cele două axe și înregistrarea măsurătorilor aferente acestora.

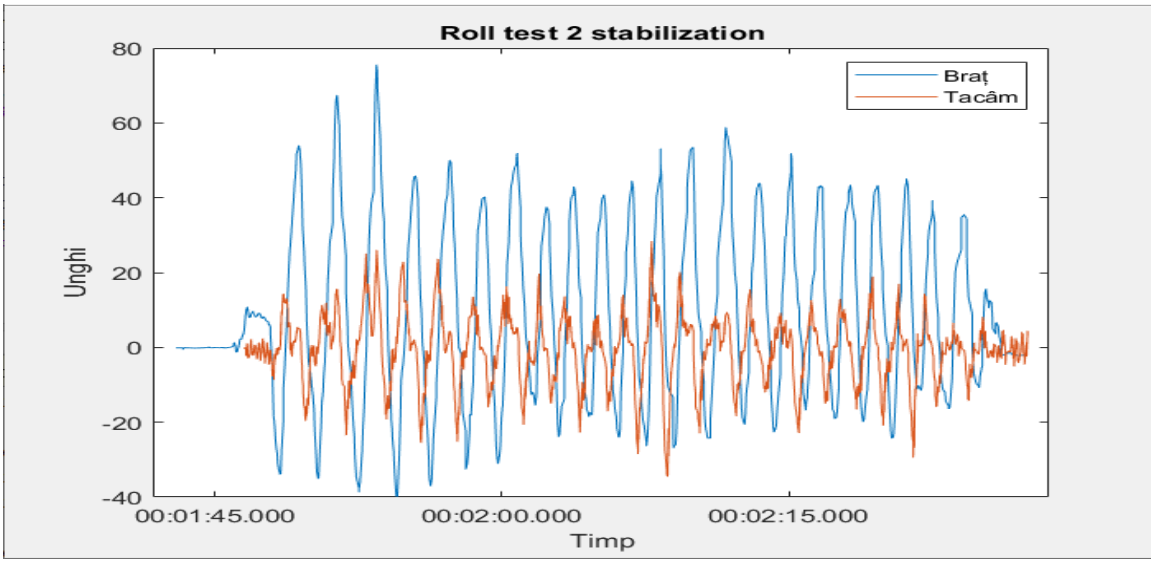
În primul test, am realizat măsurătorile necesare pentru axa de tangaj. În *Figura 5.2* sunt prezentate grafic rezultatele obținute de la cei doi senzori.



Figură 5.2 Rezultate test stabilizare pentru axa de tangaj

Vizualizând graficul anterior, se observă că pentru un tremurat la o frecvență de aproximativ 2 Hz de schimbare a direcției, se obține o amortizare de 54,35%.

În cel de al doilea test, s-a urmărit amortizarea pentru cea de-a doua axă.



Figură 5.3 Rezultate test stabilizare pentru axa de ruliu

Rezultatele obținute conform graficului anterior, prezintă o amortizare medie de aproximativ 55%.

Un ultim test a fost efectuat, în cadrul acestuia s-a plasat o greutate în bolul tacâmului cu rolul de a simula mâncarea și s-au efectuat 20 de înclinări cu o frecvență de aporximativ 2Hz.

În urma acestui experiment am obținut următoarele date:

Tabel 5.1 Rezultate stabilizare axă tangaj

Brat°	Tacam°	Amortizare
26.810	4.943	0.816
-27.950	-8.235	0.705
27.160	12.105	0.554
-19.970	-5.453	0.727
29.930	10.800	0.639
-20.950	-9.458	0.549
29.160	12.735	0.563
-20.740	-7.043	0.660
29.680	12.540	0.577
24.640	14.408	0.415
-19.110	-7.703	0.597
27.810	8.895	0.680
-20.580	-6.405	0.689
25.630	12.098	0.528
-19.110	-7.365	0.615
31.780	10.088	0.683
-20.900	-9.105	0.564
28.230	9.038	0.680
-21.230	-6.525	0.693
29.970	11.955	0.601
-21.830	-7.800	0.643
36.130	13.065	0.638
-24.610	-10.980	0.554
32.770	12.315	0.624
-14.980	-7.868	0.475

Tabel 5.2 Rezultate stabilizare axă ruliu

Brat°	Tacam°	Amortizare
53.981	11.8725	0.780
-34.753	-17.61	0.493
66.183	14.4075	0.782
-37.916	-23.498	0.380
75.590	26.04	0.656
-39.767	-17.21	0.567
44.459	22.8825	0.485
-37.076	-25.485	0.313
49.450	22.4175	0.547
-32.603	-25.2	0.227
39.606	12.375	0.688
-31.108	-15.99	0.486
50.922	15.09	0.704
-20.655	-15.4905	0.250
36.697	19.71	0.463
-23.886	-14.9925	0.372
42.056	12.4725	0.703
-22.740	-18	0.208
52.314	27.285	0.478
57.098	7.6125	0.867
-21.574	-12.045	0.442
42.286	10.3275	0.756
-19.918	-15.6225	0.216
43.976	13.8225	0.686
-16.146	-15.7625	0.024

Efectuând media amortizărilor obținute, dispozitivul a obținut per total o amortizare pentru axa de ruliu de aproximativ 50% iar pentru axa de tangaj o valoare de 62%.

6. CONCLUZII

6.1 Realizări

Pentru realizarea lucrării, au fost consultate multiple surse ale literaturii de specialitate cu scopul de a obține o privire de ansamblu mai largă și a analiza mai multe variante de implementare. Soluția finală a fost realizată în concordanță cu echipamentele accesibile, obiectivele, timpul și costurile implicate.

Etapele parcurse pentru elaborarea lucrării sau fost:

- Investigrarea problemelor existente cu care se confruntă pacienții suferinzi de maladii neuro-degenrative și cei a căror mobilitate este scăzută
- Analiza variantelor de proiectare și implementare
- Dezvoltarea dispozitivului
- Testarea dispozitivului

Pentru realizarea practică s-a ținut cont de raportul cost-performanță, dar a fost luată în calcul și alegerea unor componente cu consum redus de energie și cu dimesniuni și greutăți mici pentru a ajuta la portabilitatea dispozitivului propus.

Făcând o comparație cu obiectivele descrise în subcapitolul 1.2 cu ceea ce s-a realizat în această lucrare, se pot concluziona următoarele:

- Am identificat care sunt cerințele funcționale ale unui sistem menit să vină în ajutorul persoanelor sunt afectate de boli sau au suferit accidente ce reduc mobilitatea membrelor superioare.
- Am implementat un dispozitiv care îndeplinește cerințele funcționale specificate: amortizarea pe axa de tangaj, amortizarea pe axa de ruliu.

6.2 Dezvoltări ulterioare

Chiar dacă a fost atins scopul propus, dispozitivul mai poate fi înbunătățit:

- Înlocuirea microcontrolerului Arduino Mega cu un microcontroler mai performant, ce poate rula la o frecvență superioară algoritmul propus, pentru realizarea unui control mai eficient al motoarelor utilizând tehnici de tip field-oriented-control.
- Adăugarea unei baterii pentru crește portabilitatea dispozitivului
- Conectarea dispozitivului la un server prin Wi-Fi, unde se vor transmite date referitoare la utilizarea dispozitivului și la tremuratul compensat, permițând astfel o soluție IoT ce va face posibilă vizualizarea de la distanță a evoluției în timp tremurului
- Implementarea unui algoritm de tip Machine Learning, care să poată detecta mișcările voluntare ale mâinii

BIBLIOGRAFIE

[1] Song, P., Zhang, Y., Zha, M., Yang, Q., Ye, X., Yi, Q., & Rudan, I. (2021). „The global prevalence of essential tremor, with emphasis on age and sex: A meta-analysis. *Journal of global health"*, *11*, 04028. https://doi.org/10.7189/jogh.11.04028

[2] Rosca, E. C., Tudor, R., Cornea, A., & Simu, M. (2021). Parkinson's Disease in Romania: A Scoping Review. *Brain sciences*, *11*(6), 709. - https://doi.org/10.3390/brainsci11060709

[3] I. Sommerville, „ Software Engineering," England: Addison-Wesley, vol. 9th ed. Harlow, 2010.

[4] „ Medical News Today- 5 of the best Parkinson's spoons " 6 Martie 2023 [Interactiv]. Available: https://www.medicalnewstoday.com/articles/parkinsons-spoons .

[5] „Gyenno Spoon" 6 Martie 2023 [Intercativ] . Available https://www.gyenno.com/spoon-en .

[6] „Liftware Steady Starter Kit " 7 Martie 2023 [Interactiv]. Available: https://store.liftware.com/products/liftware-starter-kit .

[7] „Arduino MEGA Rev3" 13 Martie 2023 [Interactiv]. Available: https://store.arduino.cc/products/arduino-mega-2560-rev3

[8] „MPU6050 Accel-Gyro Arduino Tutorial" 13 Martie 2023 [Interactiv]. Available: https://lastminuteengineers.com/mpu6050-accel-gyro-arduino-tutorial/

[9] „GM2210 Gimbal Motor" 15 Martie 2023 [Interactiv]. Available: http://www.dys.hk/product/GM2210.html

[10] Yedamale, Padmaraja. "Brushless DC (BLDC) motor fundamentals." *Microchip Technology Inc* 20.1 (2003): 3-15.

[11] „SimpleFOC Shiled" 18 Martie 2023 [Interactiv]. Available: https://simplefoc.com/simplefoc_shield_product_v2

[12] „AMT 103-V", 25 Martie 2023 [Interactiv]. Available: https://www.cuidevices.com/product/motion-and-control/rotary-encoders/incremental/modular/amt103-v

[13] „ What are capacitive encoders and where are they suitable?", 26 Martie 2023 [Interactiv]. Available: https://www.motioncontroltips.com/faq-what-are-capacitive-encoders-and-where-are-they-suitable/

[14] „Capacitive, magnetic and optical encoders coparing the technologies", 28 Martie 2023 [Interactiv], Available: https://www.cuidevices.com/blog/capacitive-magnetic- and-optical-encoders-comparing-the-technologies

[15] „Arduino Mega Board Layout", 18 Aprilie 2023 [Interactiv], Available: https://www.electronicshub.org/wp-content/uploads/2021/01/Arduino-Mega-Board-Layout.jpg

[16] Romain JL Fetick , „MPU6050_light library documentation", Ianuarie 2021

[17] „Encoder setup”, 24 Aprilie 2023 [Interactiv], Available: https://docs.simplefoc.com/encoder

[18] Deepak, M & Aruldavid, Ranjeev & Verma, Rajesh & Sathyasekar, K. & Barnawi, Abdulwasa & Bharatiraja, C. & MIHET-POPA, Lucian. (2022). A Review of BLDC Motor: State of Art, Advanced Control Techniques, and Applications.

[19] „Brushless DC motors” 25 Aprilie 2023 [Interactiv], Available: https://www.lesics.com/brushless-dc-motor.html

[20] *SimpleFOC*: A Field Oriented Control (FOC) Library for Controlling Brushless Direct Current (BLDC) and Stepper Motors. A. Skuric, HS. Bank, R. Unger, O. Williams, D. González-Reyes. Journal of Open Source Software, 7(74), 4232, https://doi.org/10.21105/joss.04232

[21] „Park, Inverse Park and Clarke, Inverse Clarke Transformation” Microsemi User Guide, 10 Mai 2023 [Intercativ], Available: https://www.microsemi.com/document-portal/doc_view/132799-park-inverse-park-and-clarke-inverse-clarke-transformations-mss-software-implementation-user-guide

[22] „Understanding derivative in PID Control”, 18 Mai 2023 [Intercativ], Available: https://www.controleng.com/articles/understanding-derivative-in-pid-control/

[23] Rusnak, I. (2000, April). The Generalized PID Controller and its Application to Control of Ultrasonic and Electric Motors. *IFAC Proceedings Volumes*, *33*(4), 119–124. https://doi.org/10.1016/s1474-6670(17)38231-9

[24] O’Dwyer, A. (2000, April). A Summary of PI and PID Controller Tuning Rules for Processes with Time Delay. Part 1: PI Controller Tuning Rules. IFAC Proceedings Volumes, 33(4), 159–164. https://doi.org/10.1016/s1474-6670(17)38237-x

[25] Allu, N., & Toding, A. (2020). Tuning with Ziegler Nichols Method for Design PID Controller At Rotate Speed DC Motor. IOP Conference Series, 846, 012046. https://doi.org/10.1088/1757-899x/846/1/012046

DECLARAŢIE DE AUTENTICITATE A
LUCRĂRII DE FINALIZARE A STUDIILOR*

Subsemnatul _____GHINESCU LUCIAN-CĂLIN_____
_____,
legitimat cu _____CI_____seria ___KS___nr. _____741616_____,
CNP _____5000603114551_____
autorul lucrării_____DISPOZITIV PENTRU STABILIZAREA TACÂMURILOR_____
_____
elaborată în vederea susţinerii examenului de finalizare a studiilor de _____LICENȚĂ_____organizat de către Facultatea _____DE AUTOMATICĂ ȘI CALULATOARE_____din cadrul Universităţii Politehnica Timişoara, sesiunea _____IUNIE 2023_____ a anului universitar _____2022-2023_____, coordonator _CONF. DR. ING. LUCIAN PRODAN_, luând în considerare art. 34 din *Regulamentul privind organizarea și desfășurarea examenelor de licență/diplomă și disertație*, aprobat prin HS nr. 109/14.05.2020 și cunoscând faptul că în cazul constatării ulterioare a unor declaraţii false, voi suporta sancțiunea administrativă prevăzută de art. 146 din Legea nr. 1/2011 – legea educației naționale și anume anularea diplomei de studii, declar pe proprie răspundere, că:

- această lucrare este rezultatul propriei activități intelectuale;
- lucrarea nu conține texte, date sau elemente de grafică din alte lucrări sau din alte surse fără ca acestea să nu fie citate, inclusiv situația în care sursa o reprezintă o altă lucrare/alte lucrări ale subsemnatului;
- sursele bibliografice au fost folosite cu respectarea legislaţiei române şi a convenţiilor internaţionale privind drepturile de autor;
- această lucrare nu a mai fost prezentată în fața unei alte comisii de examen/prezentată public/publicată de licență/diplomă/disertație;
- În elaborarea lucrării nu am utilizat instrumente specifice inteligenței artificiale (IA)[1].

Declar că sunt de acord ca lucrarea să fie verificată prin orice modalitate legală pentru confirmarea originalității, consimțind inclusiv la introducerea conținutului său într-o bază de date în acest scop.

Timişoara,

Data _____15.06.2023_____

Semnătura _____

*Declaraţia se completează de student, se semnează olograf de acesta și se inserează în lucrarea de finalizare a studiilor, la sfârșitul lucrării, ca parte integrantă.
[1] Se va păstra una dintre variante: 1 - s-a utilizat IA și se menționează sursa 2 – nu s-a utilizat IA