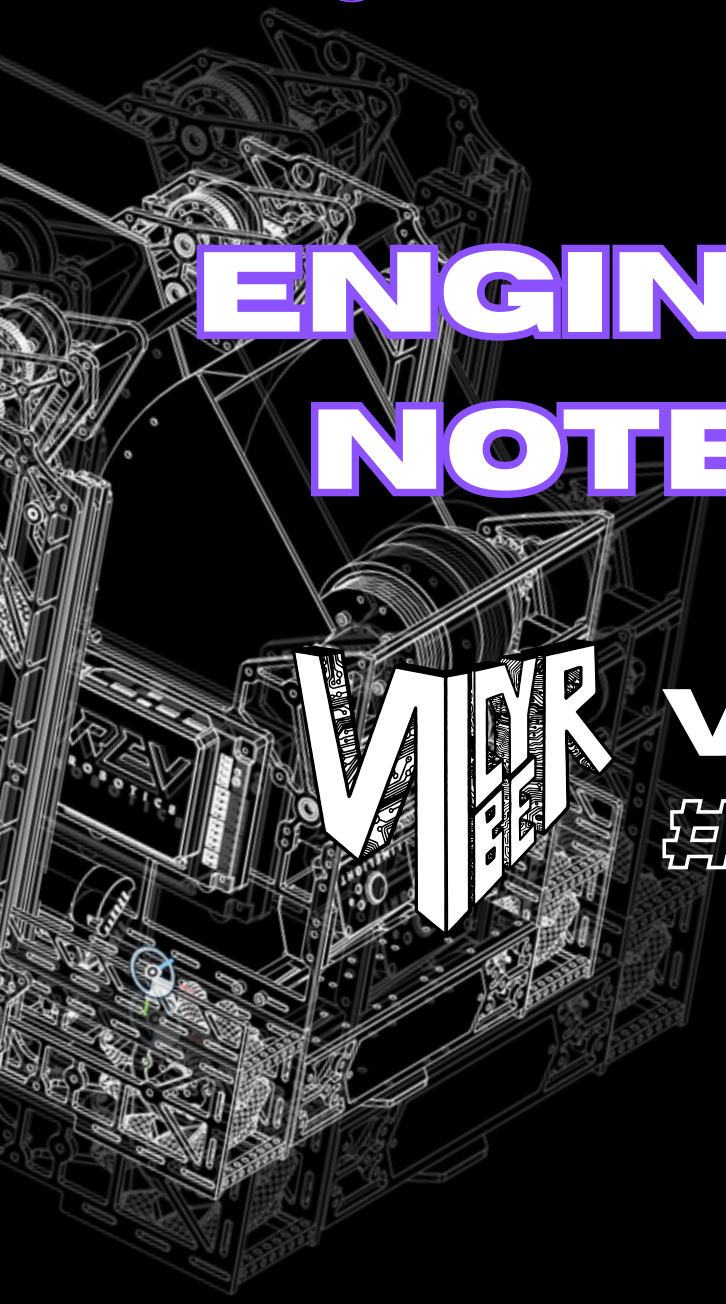
FIRST TECH CHALLENGE ROMANIA | DECODE presented by RTX | 2025 SEASON#10 2026 | ORGANIZATOR NAȚIE PRIN EDUCAȚIE | PARTNER FONDATOR BRD GROUPE SOCIETE GENERALE

ENGINEERING NOTEBOOK



VICYBER #21476

orange™ | PARKLAKE Meeting Nature | FLORIAN | ASUS | TINMAR energy | HORIZON CITY | POPP & ASOCIAȚII | F64 devoted to your passion | FormTex®

Și n-am crezut niciodată că un concurs poate schimba o viață
Până când am ajuns chiar în fața-i
Sunt recunoscătoare pentru nopțile nedormite
Pentru că mi-am dat seama că de fapt nu am limite

Sunt recunoscătoare pentru oamenii întâlniți
Pentru că ei mi-au arătat ce înseamnă să fim uniți
Pentru că mereu au fost acolo chiar și când voiam să plec
Până la urmă... doar trebuie să te lași purtat de farmec

Nu este doar o competiție, e o nouă lume
Unde nu îți dai seama ce se poate afla în mulțime
Până când nu te încumeți de a încerca
Și , de fapt , asta este FTC - o oportunitate de a te remarca

De a îți demonstra ție însuți de ce ești capabil
De a ajunge să îți dai seama că ești remarcabil
Că nu ești doar un simplu om, ci o minte sclipitoare
Și ... astea nu sunt doar cuvintele scriitoarei

Ci ale lumii întregi de tineri talentați
Inovatori ai viitoarei generații

~Alexia

CUPRINS

CAPITOLUL 1
INTRODUCERE

1.1 DESPRE ECHIPĂ

1.2 VALORILE ECHIPEI

1.3 OBIECTIVELE SEZONULUI

1.4 ORGANIZARE INTERNA

1.5 TIMELINE SEZON

1.1 DESPRE ECHIPĂ

Echipa de robotică **ViCyber #21476**, ce reprezintă **Colegiul Național de Informatică „Tudor Vianu”** din București, a fost înființată în **anul 2022** ca urmare a dorinței de a dezvolta domeniul STEAM a unor elevi pasionați de tehnologie, inovație și progres. Noi am avut de la început ca **scop** crearea unui **mediu de învățare**, atât în interiorul echipei, cât și în afara acesteia, și **performanța** în cadrul competițiilor de robotică.

Dacă ar fi să alegem un cuvânt care să ne definească, acesta ar fi **evoluția**. De-a lungul acestui sezon, am reușit să ne mărim echipa semnificativ. Am reușit să aducem doi noi mentori, convinși de misiunea și dedicarea noastră. Am reușit **să comunicăm** mai bine, **să funcționăm** mai bine. Am reușit **să consolidăm** relația cu sponsorii noștri, dovedindu-le în mod repetat că merităm încrederea oferită și că investiția în educație rămâne cea mai profitabilă. Am devenit **mai uniți și mai coordonați**, cu ajutorul mentorilor noștri.

În 2022, echipa avea câțiva membri. Era „tânără”. Cu fiecare an care a trecut, am devenit **mai buni, mai numeroși, mai pasionați, mai „în spiritul FIRST® Tech Challenge”.**

Mai avem mult de crescut și de învățat. Dar suntem mândri de noi. În acest an, am avut numeroase evenimente și am încercat ca, în fiecare eveniment, să promovăm cel puțin o valoare FIRST. Am avut performanțe în cadrul League Meet-urilor la care am participat. La Regionala Sud am obținut **Premiul I Control** și am **doborât**, pentru a doua oară, **un record european.**

Suntem formați din 15 membri dedicați, o comunitate de peste 25 de voluntari implicați activ și 3 mentori, profesorii noștri, care ne ghidează în drumul nostru spre performanță.

1.2 VALORILE ECHIPEI

Misiunea echipei noastre constă în promovarea inovației, educației și în răspândirea valorilor comunității FIRST®. Ne propunem, pe lângă stimularea interesului față de **știință și tehnologie**, să convingem tinerii să aibă inițiativă. Să creeze, la rândul lor, proiecte în care să îi implice și pe alții. Ne dorim să mărim **comunitatea STEM**, iar pentru aceasta este nevoie de inițiative, la rădăcina cărora stau valorile de bază.

Cum ne respectăm și continuăm misiunea? Răspunsul presupune două perspective. În primul rând, trebuie, pe plan interior, în echipă, să implementăm în orice activitate desfășurată setul de **Valori FIRST.**

Explorare: Mereu căutăm să descoperim noi viziuni, să dezvoltăm noi idei și să dobândim noi abilități.

Creativitate: Apelăm la cunoștințele și modurile de gândire proprii pentru a găsi noi soluții la probleme. Avem constant sesiuni de brainstorming în cadrul fiecărui departament.

Gracious Professionalism: Într-o echipă este normal să existe diferențe de opinie. Divergențele duc la idei noi și progres, dacă sunt abordate corect, într-o manieră respectuoasă, lucru pe care îl aplicăm.

Distracție: Suntem uniți ca echipă și ne bucurăm de orice moment împreună. După fiecare situație dificilă, vorbim, tragem concluzii și, eventual, râdem de greșelile noastre, într-o manieră constructivă.

În al doilea rând, trebuie să avem o activitate de outreach, prin intermediul rețelelor sociale și al evenimentelor organizate de noi. Impactul pe care l-am avut în acest sezon a fost atât asupra elevilor deja implicați în comunitate, cât și asupra copiilor/adolescenților din afara acesteia.

MOTTO

ERRORS FUEL INNOVATION

Credem cu tărie că **greșelile sunt menite să ne facă mai pregătiți, mai rezistenți, mai inovatori.** Bineînțeles, nu ne încurajăm între noi să greșim, însă ne încurajăm să încercăm, indiferent dacă rezultatul va fi cel dorit sau nu. Așa evoluăm, așa învățăm, așa devenim mai buni. Acesta este **parcursul firesc** în vederea atingerii scopurilor noastre.

1.3 OBIECTIVELE SEZONULUI

Răspândirea valorilor FIRST.

Dezvoltarea echipei în materie de outreach și coeziune.

Calificarea la Campionatul Mondial

Crearea unei **comunități** strânse în cadrul FTC România.

Crearea unor **conexiuni** cu celelalte echipe din FTC.

Implicarea cât mai multor generații din liceul nostru în echipă și în activitate.

Dezvoltarea noastră – pregătirea în materie de aptitudini care vor fi utile în **viitorul** nostru, atât din punct de vedere profesional, cât și personal.

RECRUTAREA

Recrutarea este o parte extrem de importantă în vederea **continuării misiunii noastre**.
După cum am spus, anul acesta ne-am mărit considerabil echipa. În ceea ce privește recrutarea **noilor membri**, aceasta a reprezentat o evaluare complexă a abilităților și dorinței de implicare. De asemenea, o componentă importantă pentru noi a fost „chimia”, relația interumană. Degeaba avem fiecare, individual, abilități extraordinare dacă nu există o anumită compatibilitate între noi.

Referitor la procesul de recrutare a **voluntarilor**, acesta s-a desfășurat la nivelul școlii noastre. Am promovat echipa, le-am prezentat-o „bobocilor”, am organizat workshop-uri, iar elevii s-au înscris la un interviu. De asemenea, aici ne-au fost de ajutor mentorii noștri, care au sfătuit elevii să se intereseze și să se înscrie. În prezent, avem o comunitate de **25 de voluntari implicați activ**.

1.4 ORGANIZARE INTERNĂ

Deși recunoaștem importanța dezvoltării abilităților **generale ale tuturor membrilor echipei** și, prin proiectul nostru, încurajăm dobândirea abilităților referitoare la toate departamentele echipei, am decis împărțirea echipei în **trei departamente principale** pentru a oferi fiecăruia dintre noi oportunitatea de a se **specializa** în domeniul care îl interesează cel mai mult. Pentru a progresa și a excela ca echipă este vitală **coordonarea armonioasă între departamentele echipei.**

STRUCTURA ECHIPEI

DEPARTAMENTUL MARKETING

Departamentul Marketing este esențial pentru activitățile de outreach ale echipei, pentru organizarea evenimentelor noastre, pentru menținerea prezenței noastre în mediul online, pentru conturarea identității grafice a echipei și pentru obținerea de fonduri necesare desfășurării activităților aferente echipei. Din aceste motive, departamentul de marketing are mai multe **responsabilități:**

Obținerea de fonduri – drumul spre performanță este pavat cu o multitudine de costuri, de aceea este necesară obținerea constantă de fonduri, având în vedere faptul că, odată cu atingerea unor performanțe, apar costuri din ce în ce mai ridicate. De asemenea, o parte importantă a obținerii de fonduri este cultivarea unei relații mutual benefice cu sponsorii noștri prin contact constant și actualizări continue despre valorificarea fondurilor.

Event Planning – planificarea de evenimente contribuie semnificativ la componenta de outreach a competiției FIRST Tech Challenge. Prin outreach înțelegem promovarea echipei în mediul online și offline, creșterea vizibilității și implicarea în comunitate. Organizarea unui eveniment de impact presupune o idee creativă, atractivă pentru toate categoriile de vârstă, care ulterior este transpusă într-un plan concret de acțiune, cu pași clari și resurse bine definite.

Graphic Design – designul grafic este un element central al departamentului de marketing, având un rol de suport pentru toate celelalte subcategorii. Materialele vizuale realizate (afișe, bannere, postări etc.) contribuie la identitatea vizuală a echipei și la transmiterea eficientă a mesajelor. Această activitate necesită persoane creative, atente la detalii și dornice să își îmbunătățească permanent abilitățile.

Social Media – această subcategorie este responsabilă de gestionarea și întreținerea prezenței echipei pe rețelele sociale. Aici se realizează conținut atractiv și relevant, menit să atragă atenția unui public cât mai larg. De asemenea, prin Social Media se stabilește legătura cu alte echipe, comunități și susținători. Este un domeniu care solicită creativitate, adaptabilitate și o bună înțelegere a tendințelor actuale din mediul online.

MEMBRI MARKETING

Florian

“Bună! Am intrat în comunitatea FTC în clasa a IX-a, încurajat de colegii mei — care între timp au devenit și prieteni apropiați. Participarea la acest proiect m-a motivat să cred că îmi pot transforma ideile în realitate și mi-a deschis perspectiva asupra a ceea ce înseamnă cu adevărat inovația. Pentru mine, fiecare provocare din cadrul competiției a devenit o oportunitate de a evolua.”

Bogdan

“Salut! Sunt Bogdan și totul a început din pasiunea mea pentru design și din natura mea prietenoasă, dorind să creez idei noi și să aduc oamenii împreună. First Tech Challenge nu înseamnă doar să proiectezi și să programezi roboți, ci și să construiești legături durabile, să colaborezi și să petreci timp valoros alături de echipă pe parcursul sezonului.”

Moise

“Hei! Sunt Matei, dar majoritatea mă cunosc drept Moise. Sunt elev la Colegiul Național de Informatică “Tudor Vianu”, la profilul mate-info intensiv. Am intrat în lumea STEAM în clasa a IX-a, când profesoara mea de informatică mi-a deschis această direcție — lucru pentru care îi sunt în continuare recunoscător. De atunci, pasiunea mea pentru tehnologie și rezolvarea problemelor a crescut constant, iar fiecare proiect sau concurs la care particip îmi confirmă că sunt pe drumul potrivit. “

Dorothea

“Hey! Mă numesc Dorothea și sunt aici să vă povestesc despre parcursul meu în această competiție. Fac parte din echipă încă din al doilea an de la fondare, iar această alegere a fost, probabil, una dintre cele mai bune decizii pe care le-am luat. Îmi place să interacționez cu oamenii, să compun mesaje clare și creative și să creez structuri de comunicare care să reprezinte echipa cât mai bine.”

“Hello! Eu sunt Alexia, am 17 ani și sunt elevă la Liceul Teoretic Marin Preda, la profilul mate-info. Pasiunea mea pentru robotică a început în clasa a 10-a, când o prietenă din copilărie mi-a deschis drumul către fascinanta lume STEAM. De atunci, am descoperit cât de mult mă motivează să rezolv probleme, să construiesc proiecte și să îmbin creativitatea cu tehnologia. Fiecare nou proiect îmi confirmă că vreau să îmi continui viitorul în acest domeniu.”

Alexia

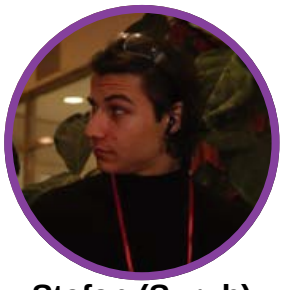
DEPARTAMENTUL HARDWARE

Departamentul hardware reprezintă unul dintre cei trei piloni esențiali ai oricărei echipe FTC, având un **rol fundamental în realizarea și dezvoltarea robotului**. Membrii acestui departament se ocupă de **proiectarea mecanică, imprimarea 3D a componentelor necesare și asamblarea efectivă a robotului, transformând ideile și conceptele în structuri funcționale.**

La începutul fiecărui sezon, echipa hardware analizează cerințele jocului și începe procesul de proiectare a robotului pentru noua provocare. În această etapă sunt realizate schițe, modele CAD și planuri tehnice care stabilesc modul în care robotul va fi construit și cum vor funcționa diferitele mecanisme. După finalizarea designului, urmează etapa de producție și asamblare, în care componentele sunt fabricate, testate și integrate în structura finală a robotului.

Pe parcursul sezonului, departamentul hardware are responsabilitatea de **a îmbunătăți constant robotul, realizând ajustări și modificări la componentele mecanice pentru a optimiza performanța acestuia.** Testarea continuă, identificarea problemelor și găsirea unor soluții eficiente fac parte din activitatea zilnică a acestui departament. Prin muncă în echipă, atenție la detalii și creativitate, membrii departamentului hardware contribuie decisiv la evoluția robotului și la succesul echipei în competiții.

MEMBRI HARDWARE



Ștefan (Șurub)

"Hello! Mă numesc Crăciunescu Ștefan. Primul meu contact cu competiția FIRST Tech Challenge a fost în 2018, iar de atunci a avut un impact major asupra parcursului meu. FTC m-a învățat să visez mai îndrăzneț, să cred în propriile idei și să caut mereu modalități de a le transforma în realitate. Această competiție mi-a modelat cu adevărat drumul. FTC mi-a oferit incredere, motivație și mi-a deschis poarta către universul STEAM, inspirându-mă să continui să mă dezvolt și să îmi perfecționez abilitățile."

Alexandra

"Hey! Mă numesc Alexandra și particip la această competiție încă din clasa a IX-a. FTC m-a învățat ce înseamnă cu adevărat să lucrezi în echipă și să colaborezi îndeaproape cu ceilalți. Departamentul hardware este pentru mine ca o familie — aici nu este vorba doar despre asamblarea sau proiectarea robotului, ci și despre combinarea tuturor ideilor într-un plan coerent și funcțional pentru competiție."

Dimi

"Mă numesc Morariu Dimitri, am participat la competiția FLL încă din clasa a IX-a, iar pot spune cu sinceritate că aceea a fost una dintre cele mai frumoase perioade din viața mea. Experiențele de atunci m-au format ca persoană, m-au învățat să lucrez în echipă, să fiu creativ și să îmi depășesc limitele.Tot FLL-ul a fost și cel care m-a motivat să continui drumul în FTC, unde pot aplica tot ce am învățat și pot evolua într-un ritm și mai accelerat."

Mihnea

„M-am alăturat FTC în clasa a IX-a, iar de atunci experiența aceasta mi-a schimbat viața. Am învățat despre construcție, programare, lucru în echipă și gestionarea proiectelor. FTC m-a ajutat să mă dezvolt, să capăt încredere și disciplină și mi-a oferit o perspectivă reală asupra tehnologiei."

“Bună! Mă numesc Bianca și fac parte din FTC încă din off-season-ul sezonului Centerstage. Am ales să fac parte din departamentul hardware deoarece îmi plac provocările și îmi oferă ocazia să învăț constant lucruri noi. Sunt o persoană curioasă și inventivă, iar acest departament îmi permite să îmi folosesc creativitatea pentru a găsi soluții și pentru a contribui la dezvoltarea robotului echipei.”

Bianca

“Hey! Sunt membru al echipei Vicyber din sezonul Into the Deep, în cadrul departamentului hardware. Mă ocup în principal de partea de asamblare a robotului. Îmi place această activitate deoarece mă relaxează și îmi oferă ocazia să gândesc cum se potrivesc piesele între ele pentru ca robotul să funcționeze cât mai bine. În același timp, învăț constant lucruri noi și îmi dezvolt abilitățile tehnice lucrând alături de echipă.”

Mihai

DEPARTAMENT SOFTWARE

În cadrul echipei noastre, acesta are **scopul de a dezvolta, testa și optimiza programele care controlează robotul echipei**. Acest departament dezvoltă atât programul de TeleOp, pentru care colaborează strâns cu drive team-ul ca să îi ajute în strategia de joc și în controlul intuitiv al robotului pentru driver, cât și autonomiile. Pentru îndeplinirea acestor scopuri, membrii departamentului software utilizează componentele hardware. De asemenea, departamentul software colaborează și cu departamentul de marketing, pentru dezvoltarea site-ului echipei, dar și a aplicației de prezentare a robotului pentru națională.

MEMBRI SOFTWARE

Alex

"Hey! Am intrat în FTC chiar în acest sezon. Pasiunea pentru matematică și informatică m-a determinat să particip la workshop-uri, iar ulterior să devin membru al departamentului software. Îmi place să învăț lucruri noi și să contribui la dezvoltarea programării robotului alături de echipă."

Thomas

"Salut! Mă numesc Thomas și am intrat în FTC datorită pasiunii mele pentru informatică și dorinței de a găsi soluții și de a rezolva probleme. Am ales departamentul software deoarece îmi oferă oportunitatea de a analiza codul, de a identifica probleme și de a găsi cele mai bune soluții pentru a optimiza funcționarea robotului."

Teodor

"Hey! Eu sunt Teo și învăț la Colegiul Național de Informatică "Tudor Vianu" încă din clasa a V-a. Pasiunea mea pentru informatică a început chiar din acea perioadă, când am descoperit cât de fascinant este să transformi idei în programe și soluții reale. De atunci, interesul meu pentru tehnologie și inovație a crescut constant, motivându-mă să particip la concursuri, să dezvolt proiecte proprii și să explorez noi domenii precum algoritmica, inteligența artificială și robotica."

Ștefan

"Salut! Mă numesc Ștefan Dănăilă și sunt elev în clasa a XI-a la Colegiul Național de Informatică „Tudor Vianu". Am intrat în FTC din pasiunea mea pentru programare și din dorința de a lucra la proiecte tehnice alături de o echipă. Îmi place să rezolv probleme, să învăț lucruri noi și să contribui la dezvoltarea robotului. În afara roboticii, sunt pasionat și de fotografie, prin care îmi place să surprind momente și perspective interesante."

MENTORI

Coman Isabela Patricia

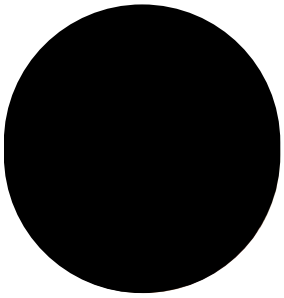
Profesoara de informatică de la Colegiul Național de Informatică „Tudor Vianu” a pătruns în lumea STEM odată cu participarea la competiția de robotică VEREO, o competiție cu roboți LEGO care există și astăzi.

„Am participat cu mare drag la aceste competiții, văzând entuziasmul copiilor și hrănindu-mă din energia lor”, povestește aceasta. Din dorința unui elev de a continua implicarea în robotică, a participat ulterior și la competiția Iancu POP.

În ultimii doi ani, a fost mentor pentru o echipă de FLL (First LEGO League). Anul trecut, echipa a ajuns până în Africa, iar anul acesta s-a calificat în Coreea de Sud, obținând locul al III-lea la etapa națională.

„Îmi plac valorile promovate de competiția FTC (First Tech Challenge) și încurajez elevii să participe la astfel de proiecte, deoarece îi dezvoltă multidimensional.”

De asemenea, profesoara pune accent pe colaborare și pe valorificarea punctelor forte ale fiecărui elev: „Îmi place să pun în valoare calitățile fiecărui elev, găsindu-le tuturor un loc potrivit în echipă.”



Popescu Cristiana

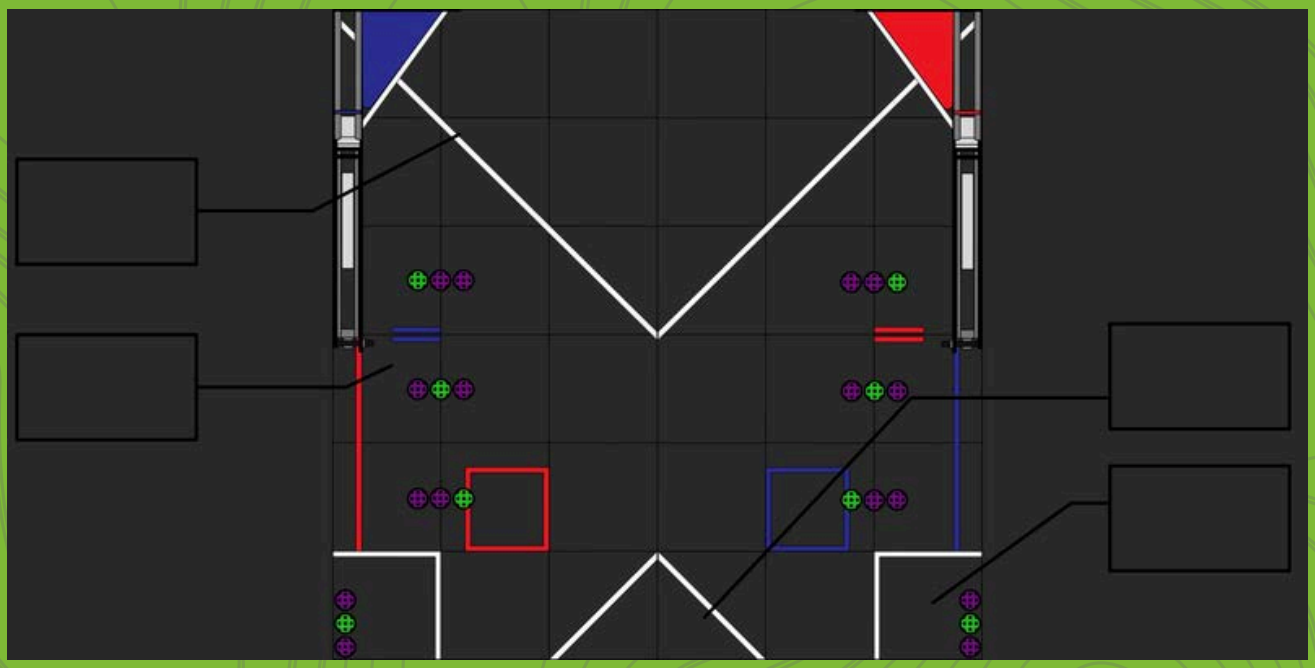
Popescu Cristiana este profesoara de informatică la Colegiul Național “Tudor Vianu”. Interesată de munca și implicarea elevilor în proiectele de robotică, ea a descoperit competiția FTC în acest an. Cu ajutorul lui Bogdan, a înțeles mai bine în ce constă activitatea din cadrul FTC și impactul pe care o astfel de experiență îl poate avea asupra elevilor. Impresionată de dedicarea echipei și de valorile promovate de competiție, a decis să se alăture proiectului în calitate de mentor.

Dobrescu Corina

Dobrescu Corina- profesor de fizică la “Tudor Vianu”. În acest an m-am alăturat competiției FTC, motivată de interesul tot mai mare al elevilor mei pentru robotică. Observând cât de des aceștia discutau despre proiectele din domeniu roboticii, dar remarcând și entuziasmul cu care se implicau, am decis să mă alătur echipei Vicyber. În competițiile de robotică, fizica este o disciplină foarte importantă care face legătura între noțiunile teoretice și aplicarea practică a acestora. Succesul unui proiect din domeniul robotici poate fi asigurat de corelarea optimă a mai multor discipline, fizică, informatică, matematică etc.

CAPITOLUL 2

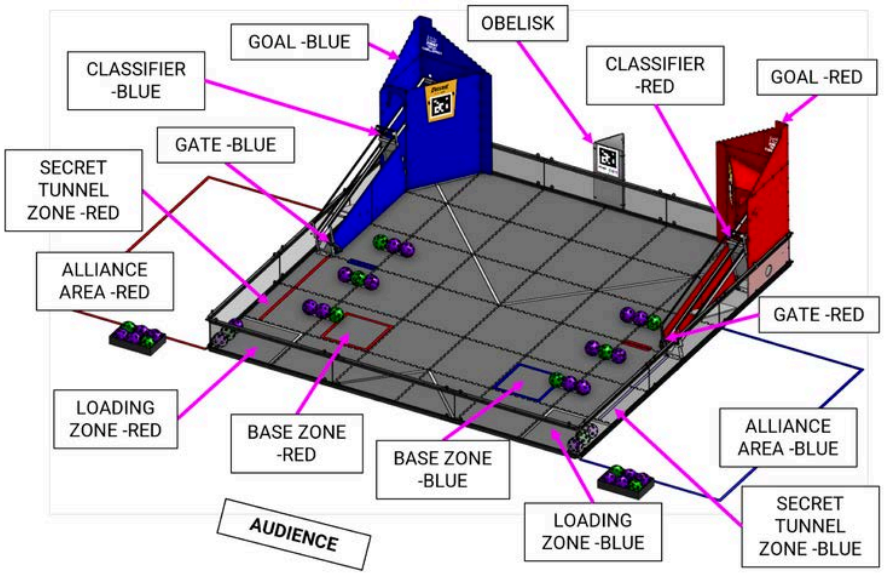
ANALIZA JOCULUI & STRATEGIE



- 2.1 DESCRIEREA PROVOCARII
- 2.2 ANALIZA SISTEMULUI DE PUNCTAJ
- 2.3 STRATEGII POSIBILE
- 2.4 STRATEGIA ALEASA
- 2.5 CERINȚE TEHNICE REZULTATE

2.1 DESCRIEREA PROVOCARII

Un meci din sezonul FIRST Tech Challenge – DECODE se desfășoară între două alianțe, fiecare formată din câte două echipe. Scopul fiecărei alianțe este de a obține un număr cât mai mare de puncte prin colectarea și scorarea elementelor de joc numite **Artifacts** în structura de scor aflată pe teren.
Terenul are dimensiunea standard FTC de aproximativ 3,6m x 3,6m. Pe teren se află elementele de joc, zona Human Player, precum și structura principală de scor, formată din componente precum **Ramp**, **Classifier** și **Gate**. Artifact-urile introduse în această structură sunt procesate de mecanismul de clasificare, iar punctele sunt acordate în funcție de modul în care acestea sunt introduse și de poziția lor finală.
Un meci durează 2 minute și 30 de secunde și este împărțit în trei etape:



• PERIOADA DE AUTONOMIE

Meciul începe cu perioada de autonomie, care durează 30 de secunde. În această etapă roboții rulează programe predefinite fără intervenția driverilor. Roboții pot să se deplaseze pe teren, să colecteze Artifact-uri și să încerce să le introducă în structura de scor sau să se poziționeze într-un mod avantajos pentru restul meciului.

• PERIOADA DE TELEOP

Durează două minute. În această etapă driverii controlează roboții folosind controllerele. Roboții colectează Artifact-uri de pe teren sau din zona Human Player și le introduc în structura de scor. Artifact-urile pot fi procesate de Classifier, iar dacă acestea respectă configurația indicată de pattern-ul activ (motif) pot genera puncte suplimentare.
Pe parcursul meciului, roboții repetă acest proces de colectare și scorare de mai multe ori, realizând așa-numitele cicluri. Eficiența unui robot depinde în mare măsură de viteza cu care poate realiza aceste cicluri și de consistența cu care introduce Artifact-urile în mecanismul de scor.

- **PERIOADA DE ENDGAME**

Ultimele 30 de secunde ale meciului reprezintă Endgame, când roboții pot realiza acțiuni finale care pot aduce puncte suplimentare, cum ar fi parcarea în zona BASE a terenului. Pentru ca parcarea să fie validă, robotul trebuie să se afle complet în interiorul zonei de parcare (Base) atunci când cronometrul ajunge la zero. Conform sistemului de punctaj:

- **Partially returned to BASE – 5 puncte**
- **Fully returned to BASE – 10 puncte**
- **Bonus dacă ambele roboți din alianță sunt fully returned – +10 puncte**

Zona de parcare, totuși, are dimensiunea unui singur robot. Rezultă astfel că, pentru ca ambii roboți să se poată parca, trebuie ca unul dintre aceștia să se ridice singur la înălțimea de 18 in., pentru ca celălalt să poată încăpea sub acesta.

2.2 ANALIZA SISTEMULUI DE PUNCTAJ

Sistemul de punctare din jocul FIRST Tech Challenge – DECODE este împărțit pe cele trei perioade ale meciului: **Autonomous, TeleOp și Endgame.** Punctele sunt acordate pentru acțiuni specifice realizate de roboți și pentru poziția acestora la finalul meciului.

Elementele de joc principale sunt **Artifacts**. Pentru a fi punctat, un Artifact trebuie să intre în GOAL prin partea superioară, să iasă sub arcadă și să treacă prin diverting square al mecanismului de clasificare.

În funcție de traseul prin sistemul de clasificare, Artifact-ul primește unul dintre următoarele punctaje:

- **CLASSIFIED Artifact – 3 puncte**
- **OVERFLOW Artifact – 1 punct**

Artifacts care nu îndeplinesc criteriile de intrare în GOAL și de trecere prin sistemul de clasificare nu sunt punctate.

- **PERIOADA DE AUTONOMIE**

- **LEAVE (robotul părăsește Launch Line) - 3 puncte per robot**
- **CLASSIFIED Artifact - 3 puncte**
- **OVERFLOW Artifact - 1 punct**
- **AUTO PATTERN - 2 puncte pentru fiecare Artifact care respectă pattern-ul**

- **PERIOADA DE TELEOP**

- CLASSIFIED Artifact - **3 puncte**
- OVERFLOW Artifact - **1 punct**
- TELEOP PATTERN - **2 puncte** pentru fiecare Artifact care respectă pattern-ul

- **PERIOADA DE ENDGAME**

La finalul perioadei TeleOp este evaluată poziția roboților în zona BASE.
Punctajul acordat este:

- Partially Returned to Base - **5 puncte**
- Fully Returned to Base - **10 puncte**
- Ambii roboți ai alianței sunt Fully Returned to Base - **10 puncte**

Evaluarea pentru BASE se face la finalul meciului, după ce toate elementele au ajuns în repaus.

- **PENALIZARI**

Dacă o echipă încalcă regulile jocului, se pot acorda penalizări. Acestea oferă puncte alianței adverse.Tipuri de **penalizări**:

- Minor Foul - **+5 puncte** pentru alianța adversă
- Major / Tech Foul - **+15 puncte** pentru alianța adversă

2.3 STRATEGII POSIBILE DE JOC

1.SCORARE DE PROXIMITATE

Prima strategie presupune poziționarea robotului în apropierea GOAL-ului pentru a arunca Artifact-urile de la distanță mică. Scopul acestei strategii este maximizarea consistenței aruncărilor și menținerea unui flux constant de Artifact-uri în mecanismul de clasificare.
În această strategie robotul colectează Artifact-uri de pe teren sau din **zona Human Player** și se deplasează aproape de GOAL pentru a le arunca în mod repetat. Deoarece robotul se află aproape de țintă, aruncările sunt mai ușor de controlat și rata de succes este ridicată. Totuși, această strategie presupune o durată a ciclului mai mare, deoarece artifact-urile vor fi mai greu de preluat. Rata de succes este mai ridicată, însă pentru un număr mai mic de aruncări.

2.CONTROL ASUPRA TERENULUI

A doua strategie se bazează pe controlul Artifact-urilor disponibile pe teren și pe limitarea accesului alianței adverse la acestea. În această strategie robotul se concentrează pe colectarea rapidă a Artifact-urilor din zonele centrale ale terenului imediat după ce acestea devin disponibile. Prin colectarea acestor elemente înaintea adversarilor, robotul reduce numărul de Artifact-uri pe care aceștia le pot folosi pentru scorare.

În același timp, robotul poate transporta Artifact-urile colectate către zonele controlate de alianță sau le poate scora atunci când apare o oportunitate favorabilă. Prin această abordare, robotul influențează indirect scorul adversarilor, deoarece limitează resursele disponibile pentru aceștia. Această strategie devine mai eficientă atunci când este combinată cu o bună coordonare între roboții din alianță, astfel încât unul dintre roboți să se ocupe predominant de scorare, iar celălalt de controlul Artifact-urilor de pe teren.

3.FAR SCORING + CICLURI RAPIDE HUMAN PLAYER

Consta în realizarea unui număr mare de cicluri rapide între zona Human Player și o poziție stabilă de tragere, de unde robotul lansează Artifact-urile în GOAL. Prin menținerea unei distanțe constante de scorare, robotul poate executa aruncări repetabile și eficiente, reducând timpul pierdut în poziționare.

În același timp, robotul monitorizează capacitatea Classifier-ului și intervine atunci când acesta se apropie de limita de 9 Artifact-uri, deplasându-se la Gate pentru a elibera Artifact-urile clasificate. Această abordare permite menținerea unui flux constant de Artifact-uri CLASSIFIED, evitând acumularea în Overflow și maximizând punctele obținute pe durata meciului.

Robotul operează într-un ciclu repetitiv format din mai multe etape. În prima etapă robotul se deplasează în zona Human Player, unde colectează Artifact-uri. După colectare, robotul se deplasează într-o poziție stabilă de tragere aflată la distanță de GOAL, de unde poate arunca Artifact-urile în mod repetabil în coș. După lansarea Artifact-urilor, robotul revine în zona Human Player pentru a începe următorul ciclu. Această succesiune de colectare, deplasare, tragere și revenire permite realizarea unui număr mare de cicluri într-un meci.

2.4 STRATEGIA NOASTRA

Strategia adoptată de echipa noastră urmărește maximizarea numărului de Artifact-uri clasificate prin realizarea unor cicluri rapide între **zona Human Player** și o poziție stabilă de tragere către GOAL (**strategia 3**). Spre deosebire de alte abordări, strategia noastră nu se bazează pe apropierea constantă de mecanismul de scor sau pe controlul terenului, ci pe **eficiența ciclurilor** și pe gestionarea corectă a classifier-ului.

Avantaje majore al acestei strategii sunt **rapiditatea si similitudinea ciclurilor**, care permit repetarea aruncărilor în condiții similare de fiecare dată, eficient. Această consistență crește probabilitatea ca Artifact-urile să ajungă corect în GOAL și reduce timpul necesar unui ciclu. În același timp, robotul nu este obligat să intre frecvent în zona mecanismului de scor, unde probabilitatea de a se intersecta cu roboți este de obicei mai mare.

Comparativ cu strategia de scorare din apropierea GOAL-ului, strategia noastră reduce semnificativ distanța parcursă în fiecare ciclu. În strategia de proximitate, robotul trebuie să se deplaseze până lângă GOAL pentru fiecare serie de aruncări, iar apoi să se deplaseze din nou pentru colectarea Artifact-urilor sau pentru resetarea classifier-ului. Aceste deplasări suplimentare cresc durata fiecărui ciclu și reduc numărul total de acțiuni de scorare care pot fi realizate într-un meci.

În comparație cu strategia de **control al terenului**, strategia noastră produce puncte într-un mod mai direct și mai constant. Controlul Artifact-urilor de pe teren poate limita scorarea adversarilor, însă nu generează în mod direct un număr mare de puncte pentru alianță. Strategia noastră **prioritizează introducerea rapidă a Artifact-urilor în sistemul de scor**, ceea ce duce la acumularea unui scor mai mare pe durata meciului.

Un **aspect important al strategiei noastre de joc, bazate pe rapiditate, este neutilizarea sortării in TeleOP**, tocmai pentru a rămâne fideli principiului nostru. Robotul este conceput pentru flow, pentru scorarea unui număr mare de artifact-uri. Prin urmare, utilizăm sortarea exclusiv în timpul perioadei de autonomie, pentru **obținerea ranking points-urilor**.

În cazul în care sortarea este eronată în timpul autonomiei, **flexibilitatea strategiei noastre ne permite să ne adaptăm în endgame** pentru înregistrarea ulterioară a ranking points-urilor. Nu folosim sortarea, ci un principiu simplu de probabilitate: într-un set de 3 bile ale pattern-ului (MOTIF) se află 2 bile mov. Avem 3 seturi de câte 3 bile, prin urmare, dacă în endgame punem în Classifier 9 bile mov, avem șanse mari de a înregistra puncte pentru ranking.

ANALIZA

Pentru a evalua eficiența strategiei, am analizat timpul necesar pentru realizarea unui ciclu complet al robotului. Etapele ciclului și timpii medii sunt următorii:

- colectare Artifact: **1 s**
- deplasare către poziția de tragere: **2.5 s**
- aliniere pentru tragere: **1 s**
- lansarea Artifact-ului: **1 s**
- revenire la Human Player: **1 s**

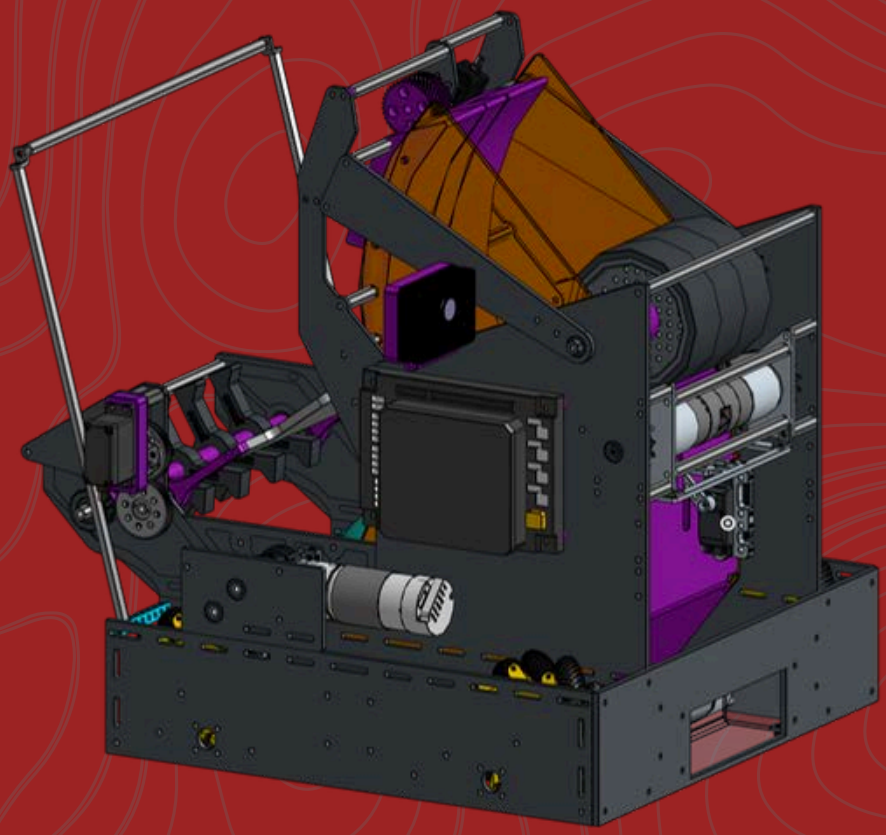
Durata totală a unui ciclu este de aproximativ **6.5 secunde**. Perioada TeleOP durează aproximativ 120 de secunde, ceea ce permite robotului să realizeze teoretic aproximativ 18 cicluri, adica **162 de puncte.** Astfel, luăm în calcul o marjă de eroare de **20%**, ceea ce înseamnă că robotul nostru are un scor minim de **130 de puncte** în TeleOP.

2.5 CERINTE TEHNICE

Implementarea strategiei alese a impus o serie de cerințe tehnice specifice asupra robotului nostru. În primul rând, robotul trebuie să fie capabil să colecteze rapid elementele de joc din zona Human Player. Acest lucru necesită un mecanism de intake eficient, care să poată ghida elementele în interiorul robotului fără blocaje și fără pierderi de timp.
De asemenea, pentru scorarea de la distanță, robotul trebuie să fie echipat cu un mecanism de lansare capabil să trimită elementele de joc către sistemul de scor cu o viteză și o precizie constante. Stabilitatea acestui mecanism este esențială, deoarece variațiile de viteză sau unghi pot afecta semnificativ precizia lansărilor.

Un alt aspect important este sistemul de deplasare al robotului. Pentru a realiza cicluri rapide, robotul trebuie să se poată deplasa eficient între zona de colectare și poziția de tragere. Acest lucru implică un șasiu stabil, capabil să accelereze rapid și să mențină controlul chiar și la viteze ridicate.
În plus, robotul trebuie să fie capabil să se poziționeze cu precizie pe teren înainte de fiecare lansare. Pentru acest lucru, este necesară integrarea unor sisteme de localizare și control care să permită robotului să mențină un unghi și o distanță optimă față de mecanismul de scor.
În final, software-ul robotului trebuie să fie optimizat pentru a coordona toate aceste mecanisme într-un mod eficient. Controlul sincronizat al sistemului de colectare, al mecanismului de lansare și al sistemului de deplasare este esențial pentru realizarea unor cicluri rapide și pentru maximizarea scorului obținut în timpul meciului.

CAPITOLUL 3
INGINERIE MECANICA & DESIGN



3.1 FILOSOFIE DE DESIGN

Încă de la începutul sezonului, am ales ca filozofia noastră de design să fie bazată pe conceptul **KISS (Keep It Simple and Straightforward)**, ca robotul să fie ușor de gestionat din toate punctele de vedere: hardware, software și drive team.

Un mecanism simplu **se calibrează mai ușor, se repară rapid între meciuri** și are un comportament **previzibil** pentru driver, ceea ce duce la **mai puține erori și mai mult timp efectiv de scorare**. În practică, am prioritizat scoruri cât mai mari și sisteme practice, în detrimentul soluțiilor mecanice complexe.

Am prioritizat consistența și, mai apoi, eficiența. Fiecare sistem trebuia să fie reliable, să aibă un randament cât mai mare, astfel încât, prin soluții mecanice simple, cu cât mai puține failure point-uri, să putem avea scoruri din ce în ce mai mari pe parcursul sezonului.

Aceste concepte s-au reflectat în toate ansamblurile noastre principale: **ȘASIU, SORTARE, INTAKE și OUTTAKE.**

Cum am respectat această filozofie pentru fiecare mecanism individual:

ȘASIU – Am ales transmisia reductoare folosind roți dințate de 24T pe motor și 60T pe roata mecanum. Acest detaliu nu a adus doar ușurință în întreținerea sa; mai mult decât atât, ne permite o precizie mult mai mare prin intermediul unui transfer de putere eficient și constant.

SORTARE – Am ales să implementăm un sistem de sortare liniară, în detrimentul unei sortări bazate pe un indexer, deoarece designul simplu al acesteia ne-a permis să avem un robot compact, care are un flow constant și rapid în teleop, dar care nu ne-a împiedicat să acumulăm RP-urile pentru sortare.

INTAKE – Am optat pentru un intake cu transmisie 1:1, pentru a minimiza pierderile. La începutul sezonului, ne-am bazat pe un intake cu compresie constantă, pentru a fi siguri că este reliable și cât mai simplu posibil, astfel încât să dea rezultate bune încă de la începutul sezonului. Ulterior, l-am optimizat, ajungând la versiunea finală de intake floppy cu 4 grade de libertate, cu compresie maximă pe apexul fiecărei bile care intră în intake. Astfel, i-am mărit raza de acțiune și am optimizat flow-ul, reducând cazurile în care două bile se blocau în intake.

OUTTAKE – Am ales transmisia 1:1 pe sprocket-uri de la două motoare bare, pentru a putea trage de oriunde de pe teren și dintr-un unghi static, întrucât reduce vibrațiile și twitch-urile unui mecanism cu unghi ajustabil, făcându-l mai rapid și mai reliable.

3.2 SASIU

VERSIUNEA 1

Concept

Am pornit de la ideea unui robot **ușor și rapid, însă cu torque mare**, pentru a putea face față și roboților mai grei și pentru a putea folosi jocul de contact în favoarea noastră.
La baza conceptului au stat câteva principii esențiale:

- **Distribuirea egală** a greutății pe fiecare roată, pentru a optimiza funcționarea mecanumurilor
- **Asigurarea spațiului** necesar pentru rampa shooterului și pentru intake
- **Amplasarea roților de odometrie pe axa centrului de rotație**, pentru a obține o localizare cât mai precisă

Evident, pe tot parcursul dezvoltării robotului din acest sezon ne-am urmat filozofia de design: ***KISS (Keep It Simple and Straightforward)*.**

Brainstorming

Pornind de la concept, am dezbătut, pe baza studiilor, mai multe modalități de **transmisie**: transmisie **pe curea**, transmisie **pe lanț** și transmisie **pe gear-uri**.

Transmisia pe curea:

Avantaje	Dezavantaje
funcționare lină masă redusă	pierderi de energie cauzate de deformarea curelei energia se disipă sub formă de căldură

Transmisia pe lanț:

Avantaje	Dezavantaje
mai robustă decât curelele nu apare fenomenul de alunecare	pierderi de energie datorate frecării dintre pini și bucșe frecarea dintre role și dinții pinionului frecarea din articulațiile lanțului

Transmisia pe gear-uri:

Aceasta oferă cea mai mare eficiență energetică dintre toate variantele analizate.

Pierderile apar în principal din frecarea dintre dinții angrenajului, iar în punctul de pas contactul este aproape pur de rostogolire.

Eficiența poate depăși 98% per angrenare, oferind:

transmisie rigidă

torque mare la turații reduse

Alte beneficii ale transmisiei pe gear-uri sunt:

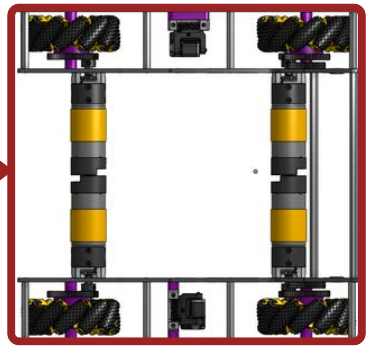
fiabilitatea

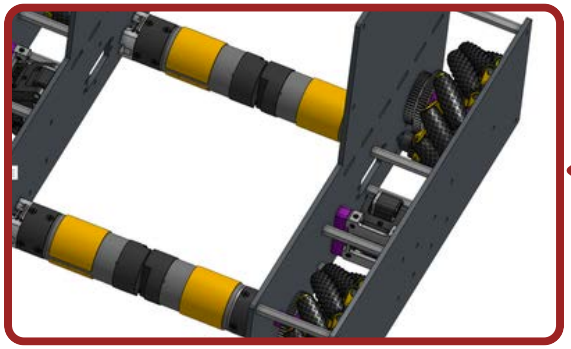
compactitatea

Astfel, am ales transmisia pe gear-uri, bazată pe: motor de 1150 RPM, raport de transmisie 2:5:1, care reduce viteza la aproximativ 460 RPM.

3. Assessment

Șasiul reprezintă baza robotului nostru, astfel că am investit în acesta atât resurse umane considerabile, cât și resurse financiare.





Pentru a avea un parcurs competițional optim, în concordanță cu strategia inițială de a participa la numărul maxim de meet-uri, am impus deadline-uri stricte, care au fost respectate în totalitate. Timeline-ul dezvoltării șasiului a fost:

- **12.09.2025 — începe procesul de brainstorming**
- **01.10.2025 — începe perioada de proiectare**
- **15.10.2025 — finalizăm prima versiune a robotului în CAD, prima variantă de șasiu**
- **01.11.2025 — finalizăm prima comandă de piese și plăci**
- **16.11.2025 — primim prima comandă de piese**
- **20.11.2025 — primim plăcile**
- **22.11.2025 — finalizăm asamblarea primei versiuni a șasiului**

4.TESTING + FAIL POINTS + OPTIMIZARE

Perioada de testare pentru noi a reprezentat un interval de 2-3 zile în care ne-am ocupat de evaluarea fiecărui ansamblu individual, lucru pe care ni l-a permis arhitectura modulară a robotului nostru. Astfel, am ales să facem prima itariție minoră asupra acestui sistem.

Inițial, în proiectul CAD podurile șasiului nostru erau formate din beamuri de 65mm și shafturi de 65mm folosite la roți + o șaibă în capătul fiecăruia, având distanța finală dintre placa interioară și cea exterioară de 66mm.
Cu toate acestea în urma testelor am ajuns la concluzia că din cauza șaibelor nu putem strânge șuruburile roților mecanum în mod egal, acest lucru afectând mișcarea șasiului.
Așadar am decis să eliminăm șaibele respective cu scopul de a nu influența negativ mișcarea șasiului.

Un alt concept inițial care a fost modificat ulterior a fost materialul din care au fost prelucrate plăcile primei variante de robot.
În ciuda faptului că atunci când am început perioada de proiectare ne doream ca prototipu să fie din lemn, am decis că acesta nu este un material fiabil pentru robot considerând mai mulți factori precum rezistența sa.

5. DATA MATCH (TEORIE VS REALITATE)

TEORIE	REALITATE
Credeam că distanțierii printați de la roți nu o să creeze probleme în timp.	În urma uzării după o perioadă îndelungată, distanțierii respectivi au început să se tocească, astfel ajungând să creeze un joc mecanic neglijabil al angrenajului de roți dințate.

Credeam că distanțierii printați de la roți nu vor crea probleme în timp. În urma uzării după o perioadă îndelungată, distanțierii respectivi au început să se tocească, ajungând să creeze un joc mecanic în angrenajul de roți dințate, inițial neglijabil, însă acumulat în timp. Credeam că distribuirea egală a greutății pe fiecare roată va fi menținută pe toată durata sezonului fără intervenții. În practică, asamblarea manuală a introdus mici asimetrii care au necesitat ajustări după primele sesiuni de testare.

6.FAIL POINTS

Distanțieri printați la roți — uzura progresivă a materialului printat a generat joc mecanic în angrenajul de roți dințate, afectând precizia mișcării mecanumurilor după utilizare îndelungată. Șaibele de la poduri — prezența lor împiedica strângerea uniformă a șuruburilor roților mecanum, generând asimetrii în transmisie. Plăci din lemn în prototip — rezistența structurală insuficientă a impus înlocuirea materialului înainte de asamblarea finală.

7. OPTIMIZARE (OBIECTIVE V2)

Prima optimizare implementată a fost eliminarea șaibelor de la poduri, restabilind:

- uniformitatea strângerii șuruburilor
- simetria transmisiei

Pentru V2, **direcțiile principale** sunt:

- înlocuirea distanțierilor printați cu distanțieri metalici de precizie
- validarea distribuirii greutății pe roți prin măsurători directe după asamblare

Aceste modificări vor reduce dependența de precizia manuală a asamblării și vor îmbunătăți consistența performanței robotului.

VERSIUNEA 2 (SASIU)

1.Brainstorming

Cea de-a doua versiune a șasiului a fost și **versiunea finală**. Pentru a respecta conceptul designului și strategia de joc, era necesară o rezistență structurală ridicată, dar în același timp și o viteză mare a șasiului.

Astfel, am ales **aluminiu 6082-T6**, întrucât, deși este mai greu decât alte materiale posibile, oferă o rezistență mecanică mult mai mare, ceea ce îl face potrivit pentru solicitările mecanice generate de jocul de contact.

Totuși, folosirea acestui material a dus la creșterea masei robotului, ceea ce ar fi putut anula avantajul obținut prin transmisia motoarelor. Pentru a compensa acest lucru, am decis să aplicăm o optimizare topologică agresivă a plăcilor, inspirată din stilul de design utilizat frecvent în FRC, prin eliminarea materialului din zonele cu solicitare structurală redusă.

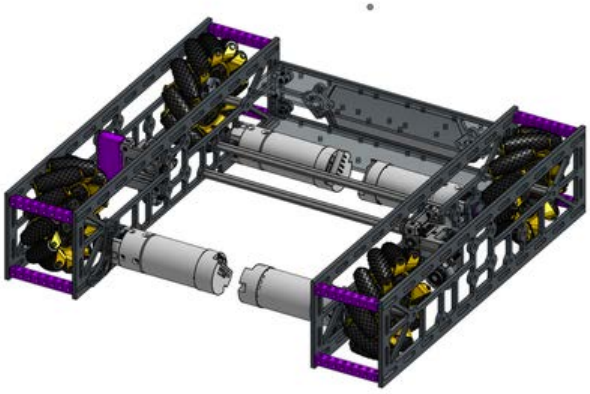
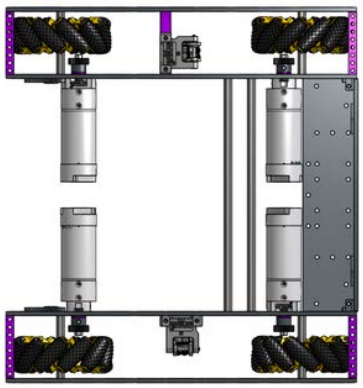
Prin această metodă am reușit să păstrăm rigiditatea structurală necesară, reducând în același timp masa totală a șasiului.

2.Assessment



Pentru a putea avea un parcurs competițional optim, în concordanță cu strategia inițială de a participa la numărul maxim de meet-uri, a fost necesar să lucrăm rapid, organizându-ne activitatea în agile sprint-uri.
Astfel, am ajuns la următorul timeline al subansamblului:

- **23.11.2025 — începe procesul de brainstorming pentru a prioritiza elementele care necesitau optimizare**
- **24.11.2025 — începe perioada de proiectare**
- **25.11.2025 — finalizăm prima versiune a robotului în CAD, inclusiv a doua variantă de șasiu; în aceeași zi am plasat și comanda pentru plăci și am stabilit data debitării**
- **30.11.2025 — primim plăcile**
- **01.12.2025 — finalizăm asamblarea versiunii a doua a șasiului**

Bugetul alocat V2 a fost semnificativ mai redus față de V1, deoarece modificările au fost minore și țintite:

- **înlocuirea distanțierilor printați cu distanțieri metalici**
- **reproiectarea podurilor șasiului**

Bugetul alocat V2 a fost semnificativ mai redus față de V1, modificările fiind minore și țintite — distanțieri metalici de schimb și reproiectarea podurilor, fără piese sau plăci noi.

TESTARE - V2 SASIU

Testarea V2 a urmărit direct parametrii care generaseră fail points în V1 — uniformitatea strângerii șuruburilor roților mecanum și absența jocului mecanic în angrenajul de roți dințate după utilizare îndelungată. **Sesiunea de testare a durat 2 zile**, utilizând același protocol de evaluare modulară aplicat în V1 pentru a permite comparația directă între variante.

Strângerea uniformă a șuruburilor roților mecanum a fost verificată imediat după asamblare și după fiecare sesiune de testare. Eliminarea șaibelor **a rezolvat complet asimetria identificată în V1** — toate cele patru roți prezentau același cuplu de strângere și aceeași libertate de rotație după asamblare.

Jocul mecanic în angrenajul de roți dințate a fost monitorizat pe durata întregii sesiuni de testare. Distanțierii metalici de precizie nu au prezentat nicio uzură sesizabilă, iar jocul mecanic înregistrat în V1 după utilizare îndelungată nu a reapărut.

DATA MATCH(TEORIE VS REALITATE)

Credeam că eliminarea șaibelor și înlocuirea distanțierilor printați cu unii metalici vor rezolva complet jocul mecanic și asimetriile de transmisie identificate în V1. În practică, ambele ipoteze s-au confirmat — uniformitatea strângerii șuruburilor a fost restabilită, iar angrenajul de roți dințate nu a prezentat joc mecanic sesizabil pe durata testelor.

Consideram că **modificările minore din V2 nu vor introduce fail points noi față de V1.** Această ipoteză **s-a confirmat — V2 nu a generat probleme noi** în sesiunea de testare, iar comportamentul șasiului a fost consistent și predictibil pe toată durata evaluării.

REPETABILITATE

Mișcarea mecanumurilor în V2 a fost semnificativ mai consistentă față de V1 pe termen lung. Distanțierii metalici au menținut geometria podurilor constantă indiferent de numărul de cicluri de utilizare, iar absența șaibelor a eliminat variabilitatea de strângere care în V1 introducea asimetrii după fiecare reasamblare.

Șasiul V2 a funcționat fără intervenții de mentenanță pe toată durata sezonului competițional.

FAIL POINTS

Niciun fail point nou nu a fost identificat în urma testării V2. Singurele aspecte de monitorizat pe termen lung rămân tensionarea uniformă a transmisiei și verificarea periodică a strângerii șuruburilor roților mecanum după sesiuni intense de joc de contact — aspect inerent oricărui șasiu cu mecanumuri, nu specific arhitecturii V2.

OPTIMIZARE & COMPARATIE CU V1

Parametru	V1-NOPAPI1	V2-Final
Distanțieri roți	printați	metalici de precizie
Joc mecanic angrenaj	prezent după utilizare îndelungată	absent
Șaibe poduri	prezente	eliminate
Uniformitate strangere roți	afectată	restabilită
Intervenții mentenanță	necesare periodic	absente pe durata sezonului
Fail points noi	----------	Niciunul

V2 a reprezentat o iterație chirurgicală față de V1 — două modificări țintite, zero compromisuri, **zero fail points noi**. Filozofia **KISS** aplicată și în procesul de optimizare a permis echipei să închidă rapid ciclul de iterație și să aloce resursele rămase celorlalte subansambluri ale robotului.

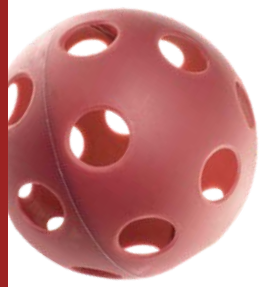
ANALIZA ALUMINIULUI 6082-T6

Aluminiul 6082 este un aliaj din seria 6000, cu magneziu și siliciu ca elemente principale de aliere, ceea ce îi conferă o combinație rară între rezistență mecanică ridicată și greutate redusă — raportul rezistență/greutate fiind unul dintre cele mai bune disponibile în categoria aliajelor de aluminiu structural. **Rezistența la tracțiune de ~310 MPa și limita de curgere de ~260 MPa îl plasează în categoria aliajelor de înaltă rezistență, însă cu o densitate de doar 2,70 g/cm³, semnificativ sub oțel (~7,85 g/cm³)** — un aspect critic pentru un robot FTC unde fiecare gram din bugetul de masă contează direct în performanța de accelerație și manevrabilitate a mecanumurilor.

Un alt avantaj esențial pentru aplicația noastră este prelucrabilitatea excelentă prin frezare CNC — aliajul produce suprafețe curate, toleranțe strânse și margini precise fără bavuri, ceea ce înseamnă că plăcile șasiului pot fi produse cu găuri și decupaje poziționate exact conform CAD-ului, eliminând erorile de asamblare care în V1 introduseseră asimetrii.

Rezistența bună la coroziune a aliajului 6082 — datorată formării naturale a unui strat de **oxid de aluminiu** — îl face de asemenea inert la variațiile de umiditate din sălile de concurs, spre deosebire de lemnul din prototipul V1, care absorbea umiditatea și își modifica dimensiunile în timp.
Nu în ultimul rând, comportamentul aliajului la impact repetat — specific jocului de contact din DECODE — este **net superior oricărui material printat sau lemnos**, deformându-se plastic înainte de rupere și menținând integritatea structurală a șasiului chiar și după coliziuni repetate în timpul meciurilor.

Parametru	Lemn	Aluminiu 6082
Densitate	~0.5–0.7 g/cm³	~2.70 g/cm³
Rezistență la tracțiune	~40–80 MPa	~310 MPa
Limită de curgere	—	~260 MPa
Modul de elasticitate	~10–15 GPa	~70 GPa
Rezistență la impact repetat	scăzută	ridicată
Comportament la umiditate	absorbție — deformare în timp	inert
Toleranțe de prelucrare	variabile — fibra introduce imprecizii	strânse — frezare CNC precisă
Greutate relativă placă șasiu	redusă	medie — compensată de grosime redusă
Durabilitate în joc de contact	scăzută — risc de fisurare	ridicată



3.3 SORTARE

VERSIUNEA 1

One way door

1.CONCEPT

Obiectivul sistemului de sortare a fost de a direcționa consistent artefactele colectate spre shooter sau spre hopper, în funcție de culoarea acestora, fără a întrerupe fluxul de intake. Prioritatea a fost o soluție mecanică simplă, cu număr minim de componente în lanțul cinematic.

2.BRAINSTORMING

Pornind de la concept, echipa a dezbătut ce arhitectură de sortare se potrivește cel mai bine unui flux continuu de elemente.

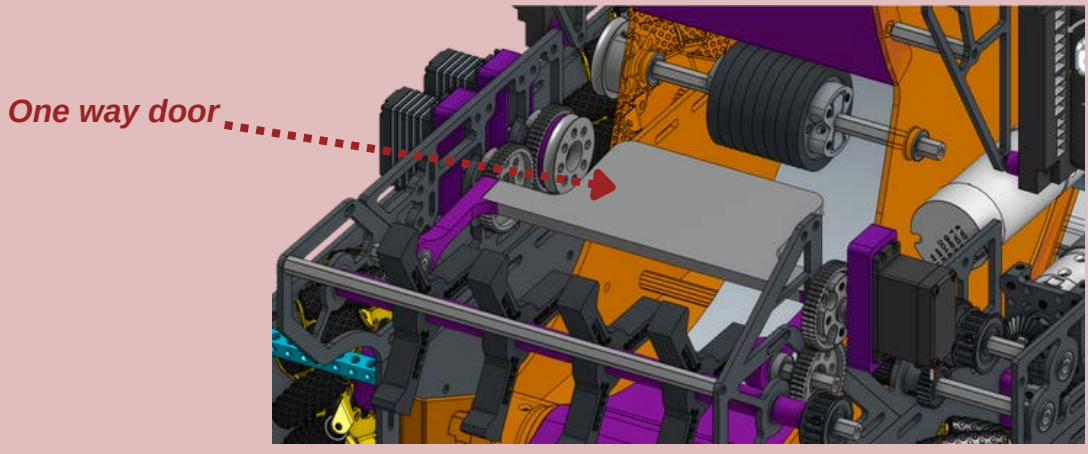
Principalele variante analizate:

- one-way door printat acționat de servo AXON — o ușă printată montată pe traseul
- elementului blochează sau eliberează unul din cele două trasee (shooter / hopper) în - funcție de comanda de sortare;
- deflector activ cu piesă metalică pe ax pasiv — mai complex mecanic, dar mai rigid și mai rezistent la impact repetat;
- sortare pasivă prin geometrie — respinsă, geometria artefactelor din Decode nu permitea o separare pasivă fiabilă.

Echipa a ales one-way door-ul printat ca primă implementare, datorită simplității de fabricație și a numărului minim de componente necesare. Servoul AXON părea suficient pentru a acționa o piesă printată ușoară, fără a fi nevoie de o transmisie suplimentară.

3.DESIGN

Elementele colectate de intake sunt transferate pe un canal liniar. La bifurcația dintre traseul spre shooter și traseul spre hopper este montată o ușă printată tip one-way, acționată direct de un servo AXON.



În poziție neutră, ușa lasă artefactul să avanseze spre shooter. La comandă, servoul rotește ușa, blocând un traseu și deschizând celălalt în funcție de logica de sortare implementată.

4.TESTING

4.1 DATA MATCH (TEORIE VS. REALITATE)

Teoretic, ușa printată oferea o separare clară între cele două trasee, iar viteza servoului AXON era mai mult decât suficientă pentru cicluri rapide.
În practică, ***rigiditatea limitată*** a materialului printat introducea o *zonă de ambiguitate* la bifurcație — elementele cu viteză mare deformau ușa la impact, modificând unghiul efectiv de deflecție și făcând comportamentul sistemului **impredictibil** indiferent de precizia servoului.

4.2 REPETABILITATE

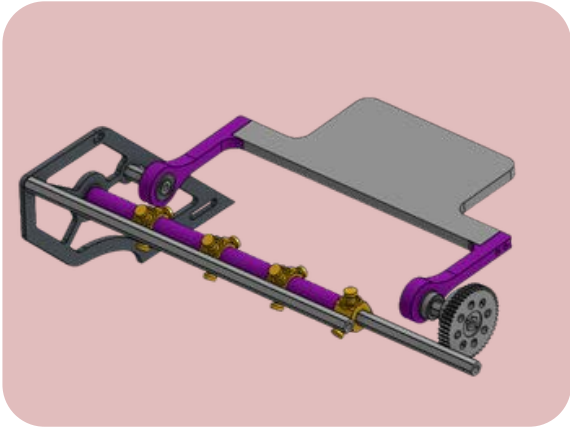
Consistent la viteze reduse de transfer. La flux rapid, impactul repetat al artefactelor asupra ușii printate **degrada progresiv geometria piesei**, reducând repetabilitatea sortării în timp. *Problema nu era servoul, ci materialul piesei active.*

4.3 FAIL POINTS

Rezistența mecanică a ușii printate — **impactul repetat** al artefactelor la viteze mari de transfer provoca **deformarea sau fracturarea piesei**, necesitând *înlocuire frecventă*. Zona de ambiguitate la bifurcație — flexibilitatea materialului printat genera deflecții parțiale și **redirecționări impredictibile** chiar și cu un servo de precizie precum AXON.

4.4 OPTIMIZARE V1 (OBIECTIVE V2)

Principala concluzie după testarea V1 a fost că servoul AXON nu reprezenta limitarea sistemuluii — viteza și precizia sa de poziționare erau mai mult decât adecvate pentru ritmul de sortare necesar în meciuri. Limitarea reală era exclusiv mecanică: ***ușa printată nu putea prelua impactul repetat al artefactelor fără să se degradeze progresiv*.** Prin urmare, direcția de optimizare pentru V2 nu a vizat actuatorul, ci rigiditatea lanțului cinematic dintre servo și artefact.



Prima măsură propusă a fost înlocuirea ușii printate cu o piesă metalică fixată pe un ax pasiv. O piesă metalică prezintă un modul de elasticitate semnificativ mai mare față de orice filament de imprimare 3D standard — inclusiv PETG — ceea ce înseamnă că unghiul de deflecție **rămâne constant indiferent de viteza sau masa elementului la impact. *Aceasta elimină direct zona de ambiguitate de la bifurcație identificată în fail points.***

A doua măsură propusă a fost introducerea unei transmisii 1:1 pe roată dințată între servo și axul pasiv pe care stă piesa metalică. Deși la prima vedere un raport 1:1 nu modifică turația sau cuplul, beneficiul real este decuplarea inerțială — piesa metalică, mai grea decât ușa printată, nu mai acționează direct pe pinionul servoului, reducând solicitările mecanice pe angrenajul intern al AXON-ului și prelungind durata de viață a actuatorului pe termen lung.

O a treia direcție identificată a fost adăugarea unei transmisii pe lanț spre roțile gecko, pentru a asigura un avans controlat și predictibil al artefactului prin zona de sortare — eliminând variațiile de viteză care în V1 făceau momentul exact al deflecției greu de anticipat.

VERSIUNEA 2 (SORTARE)

Ax pasiv + Gecko wheels

1.CONCEPT

Pornind de la limitările identificate în V1, **obiectivul V2 a fost eliminarea punctelor de failure mecanice ale one-way door-ului și creșterea vitezei și preciziei de sortare.** Sistemul trebuia să redirecționeze artefactele spre shooter sau spre hopper consistent, pe tot parcursul unui meci, fără degradare mecanică și fără dependență de timing.

2.BRAINSTORMING

Analiza V1 a arătat două probleme fundamentale. Orice soluție V2 trebuia să adreseze ambele simultan. Variantele dezbătute au inclus:

- servo standard direct pe piesă metalică — respins, nu oferea viteza și cuplul necesare fără joc rezidual
- one-way door din material mai rezistent — respins, rezolva doar degradarea mecanică, nu și viteza de răspuns;
- servo AXON cu transmisie 1:1 pe roată dințată și ax pasiv — adoptat, decuplează inerția servoului de piesa activă și oferă viteză de răspuns ridicată cu precizie de poziționare superioară.

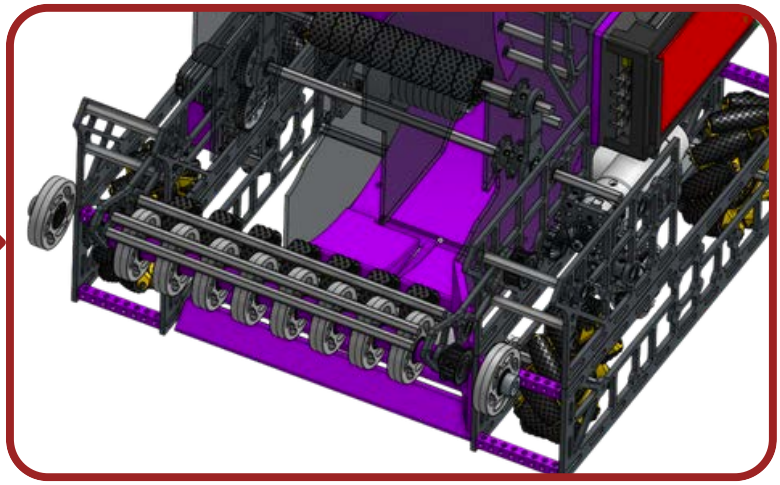
Transmisia pe lanț spre gecko wheels a fost adăugată pentru a asigura avansul controlat al artefactului prin zona de sortare, eliminând variațiile de viteză care contribuiseră la problemele de timing din V1.

3.DESIGN & TRANSMISIE

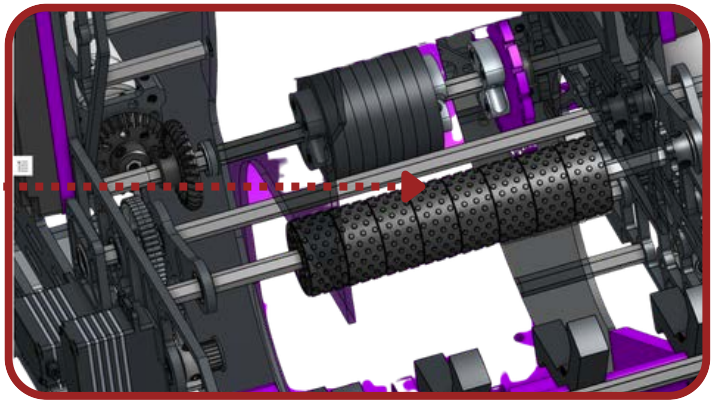
Servoul AXON acționează o roată dințată printr-o transmisie 1:1 pe un ax pasiv. Pe roata dințată este fixată o piesă metalică care constituie elementul activ de sortare. La rotirea axului, piesa metalică redirecționează fizic artefactul pe unul din cele două trasee:

- **element valid — direcționat spre shooter;**
- **element invalid — direcționat spre hopper.**

Între axul pasiv și axul pe care sunt montate **roțile gecko de 32mm** există o transmisie pe lanț, care asigură avansul continuu și controlat al artefactelor prin zona de sortare. Roțile gecko mențin **grip suficient** pe element pe toată durata tranzitului, eliminând variațiile de viteză care în V1 compromiteeau precizia deflecției.



Sistem de sortare



Transmisia 1:1 la nivel de roată dințată păstrează turația și cuplul servoului AXON nemodificate la axul piesei de deflecție — esențial pentru un timp de răspuns minim între comanda de sortare și poziționarea efectivă a piesei metalice.

4.TESTING

4.1 DATA MATCH (TEORIE VS. REALITATE)

Parametru	V1 – One Way Door	V2 – Ax Pasiv + Gecko
Actuator	servo AXON	servo AXON (același)
Piesă activă	ușă printată	piesă metalică
Rezistență mecanică	scăzută	ridicată
Consistență la flux rapid	scăzută	ridicată
Degradare în timp	prezentă (deformare print)	minimă pe durata sezonului

4.2 REPETABILITATE

Sistemul s-a dovedit consistent pe întreaga durată a sezonului competițional. Piesa metalică poziționa repetat și precis artefactele pe traseul corect — spre shooter sau spre hopper — confirmând că potențialul servoului AXON exista și în V1, însă era limitat de piesa printată.

Transmisia pe lanț spre gecko wheels a menținut avansul elementelor uniform. Consistența sistemului a contribuit direct la acumularea ranking points-urilor necesare atât la meet-uri, cât și la regională.

4.3 FAIL POINTS

Tensionarea lanțului — variațiile de tensiune în funcție de temperatură și uzură pot introduce un ușor joc în transmisia spre gecko wheels, fiind astfel nevoie de o verificare între meciuri. Fixarea piesei metalice pe roata dințată — la viteze mari de transfer, piesa preia impactul direct al artefactului; necesită verificarea periodică a fixării pe durata unui sezon lung.

4.4 OPTIMIZARE & COMPARATIE CU V1

Parametru	V1 – One Way Door	V2 – Ax Pasiv + Gecko
Actuator	servo AXON	servo AXON
Piesă activă	ușă printată	piesă metalică pe ax pasiv
Transmisie deflector	directă	1:1 roată dințată
Transfer element	canal simplu	lanț + gecko wheels 32mm
Rezistență mecanică	scăzută	ridicată
Repetabilitate	scăzută	ridicată
Degradare în timp	prezentă	minimă

Trecerea de la V1 la V2 a fost una de rigiditate mecanică, nu de actuator. Același servo AXON, prezent în ambele variante, și-a putut exprima pe deplin potențialul abia în V2 — unde înlocuirea ușii printate cu o piesă metalică pe ax pasiv și introducerea transmisiei pe roată dințată au eliminat sursa principală de failure și au oferit echipei un sistem de sortare consistent pe tot parcursul sezonului.

3.4 INTAKE

VERSIUNEA 1/REGIONALA

1.CONCEPT

Am pornit de la ideea **de a colecta cât mai consistent artefactele**, **cât mai ușor**, **și a păstra compresia constantă în transmisia spre shooter**. De asemenea, ca la începutul sezonului driverul să nu poată colecta mai mult de 3 elemente, pentru a evita penalizările.

2.BRAINSTORMING

Pornind de la concept, am început să dezbatem ce putem folosi pentru preluarea bilelor. Nu eram siguri dacă artefactele sunt suficient de compliant cât să putem pune un poly roller cu grip tape sau suficient de tari cât să fie nevoie de flex wheels. Așa că am restrâns posibilitățile la op-take-uri cu rubber tube, compliant stars și bootwheels. Fiind începutul sezonului era f time consuming să încercăm mai multe variante de tub. Însă, doar bootwheelurile vin cu inner hex de 8.

3.DESIGN & TRANSMISIE

Motorul de intake (1150 RPM, goBILDA) transmite mișcarea prin bevel clampuri 30T într-un raport 1:1, menținând turația nominală la arborele bootwheelurilor. Alegerea unui raport 1:1 la nivel de bevel a fost deliberată: la un diametru efectiv al bootwheelului de ~50 mm, viteza tangențială la suprafață este:

$$v = (1150\ \text{rot/min} \times \pi \times 0.050\ \text{m}) / 60 \approx 3.01\ \text{m/s}$$

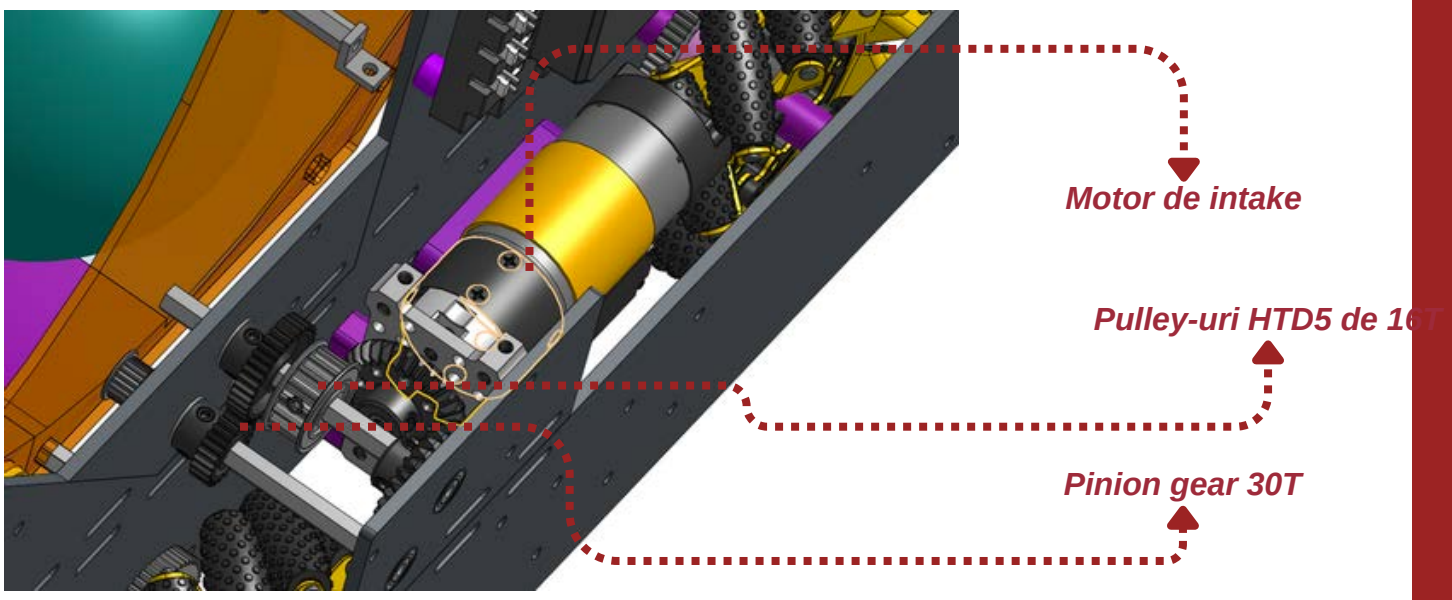
Această viteză asigură o accelerare suficient de agresivă a elementului fără a genera forțe radiale excesive care ar putea deforma geometric artefactul sau introduce slip în transmisie.

De pe același arbore al motorului, prin intermediul unui **pinion gear 30T** în mesh cu un pinion de **24T**, se realizează inversarea sensului de rotație și o ușoară supravitezare a counter-rollerului:

$$n = 1150 \times (30/24) \approx 1437.5 \text{ RPM}$$

Counter-rollerul este acționat prin **pulley-uri HTD5 de 16T**, oferind un profil de curea cu backlash minim și o tensionare previzibilă. Sensul opus față de bootwheels creează o zonă de convergență activă a forțelor, asigurând o compresie direcțională controlată asupra artefactului la intake.

Deasupra motoarelor de șasiu a fost printată o rampă personalizată cu unghi de atac calculat pentru a prelua elementele de pe sol și a le ghida consistent spre nip-point-ul dintre bootwheel și counter-roller, reducând dependența de poziționarea precisă a robotului în fața artefactului.



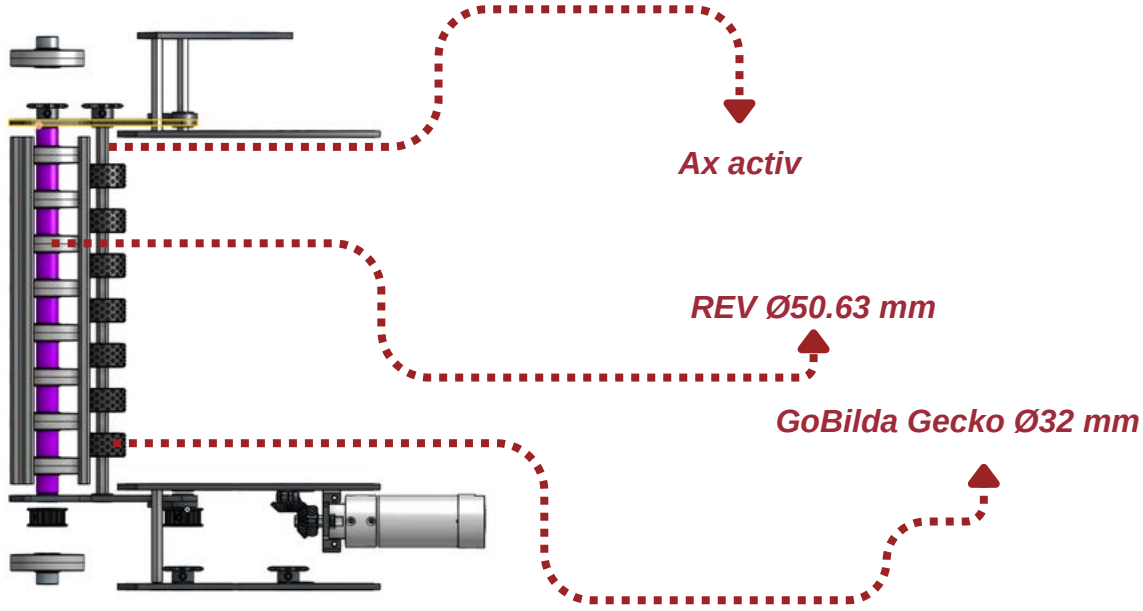
VERSIUNEA 2/ NATIONALA (INTAKE)

1.BRAINSTORMING

În urma experienței de la etapa regională, am decis să optăm pentru un **intake activ**, pentru a putea asigura un grad diferit de compresie asupra elementului de joc, astfel crescându-se flow-ul de bile și mobilitatea în teren. Acest mecanism constă în **2 axuri active**, unul cu **gecko wheels** și celălalt cu **roți REV**. Acestea sunt acționate de **2 transmisii 1:1**, una pe lanț, iar cealaltă pe curea dințată cu fulii de 16 dinți.

La etapa regională, robotul era echipat cu un intake activ static, construit în jurul unor boot wheels de Ø96 mm. Designul oferea o compresie fixă, constantă asupra bilei, independent de poziția robotului sau de condițiile de teren. Deși intake-ul s-a dovedit extrem de rapid în condiții standard de joc, experiența din meciuri a scos la iveală o limitare critică: în cadrul perioadelor de autonomie, robotul întâmpina dificultăți în colectarea bilelor poziționate în colțuri, unde unghiul de abordare și distanța față de perete nu permiteau rolelor de 96 mm să exercite o compresie eficientă. Diametrul mare al boot wheels-urilor reducea flexibilitatea geometrică a sistemului, forțând robotul să se poziționeze foarte precis pentru a asigura contactul corect cu bila.

Pe baza acestor observații, echipa a decis tranziția către un **sistem cu două seturi de role de diametre diferite** — **REV Ø50.63 mm** și **GoBilda Gecko Ø32 mm** — care să permită o compresie progresivă și adaptabilă, crescând atât flow-ul de bile, cât și mobilitatea efectivă în teren.



2.ASSESSMENT

Dezvoltarea intake-ului a implicat trei membri ai echipei cu roluri complementare: Dimi, Alexandra, Fane.

Buget alocat intake: **~3.500 RON**

3.TESTING

Varianta 2 reprezintă iterația finală a intake-ului, rezultată din lecțiile variantei anterioare și din analiza problemelor observate la regionale. Testarea a vizat trei parametri principali: rata de colectare a bilelor (flow rate), comportamentul în colțuri și consistența compresiei la viteze variate de deplasare.

Testele au fost structurate în serii de colectare continuă a câte 10 bile consecutive, pe suprafață plană și în apropierea pereților, cu robotul în mișcare și staționar. S-au monitorizat numărul de bile ratate, timpul mediu de colectare per bilă și eventualele episoade de jamming.

1. Timp de ciclu mai mic: Cheia oricărui robot de succes este timpul de ciclu. Un intake eficient duce la îmbunătățiri majore ale scorului — un intake bun ar trebui să colecteze elementele necesare mai putin de o secunda. Un intake articulat poate ajunge la bile din unghiuri diferite fără a repositiona întregul robot.
2. Colectare din multiple unghiuri și poziții: Un intake activ cu flaps rotative poate colecta rapid elementele de joc din diverse unghiuri și margini, pozitionând corect bila pentru pasul următor.
3. Flexibilitate strategică la shooting: Capacitatea de a trage din multiple locații reduce timpul de ciclu și crește rezistența la apărarea adversarului. Un intake mobil permite preluarea bilelor fără a fi nevoit să aliniezi robotul perfect.
4. Durabilitate și rezistență la impact: Intakeurile ies adesea din perimetrul robotului, deci durabilitatea este critică — fie prin construcție robustă, fie prin flexibilitate (ex: lexan sau arc de revenire) care permite deformarea la impact și revenirea la poziția inițială
5. Adaptabilitate la situații de joc: Succesul depinde de intake consistent, shooting precis și performanță fiabilă la endgame Un intake static limitează robotul la o singură poziție de colectare, în timp ce unul mobil permite adaptarea rapidă la haosul de pe teren.
6. Avantaj în autonomie: Pentru a obține Goal RP în autonomous, echipele trebuie să scoreze cât mai multe Artifacts, ceea ce necesită un intake rapid care să poată fi repoziționat fără pierderi de timp. ; Autonomia de la far usurata, intrucat nu se mai blocau bile in intake

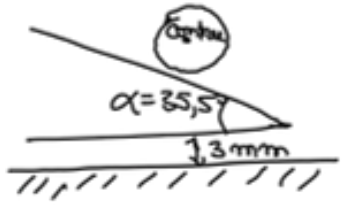
3.1 DATA MATCH

Parametru	Valoare teoretică	Valoare măsurată	Diferență
Compresie set REV	3.315 mm (5.22%)	3.20 mm	-3.47%
Compresie set Gecko	2.700 mm (4.25%)	2.80 mm	+3.70%
Compresie totală	6.015 mm (4.74%)	6.00 mm	-0.25%
Înălțime centru bilă	54.7 mm	54.5 mm	-0.2 mm
Flow rate estimat	2.0 bile/sec	1.8 bile/sec	-0.2 bile/sec
Bile ratate / 10 consecutive	0	1	+1

CALCULE DATA MATCH

3. Calcule geometrice

3.1 Înălțimea centrului bilei față de sol



$h_{centru} = h_{șasiu} + r_{bilă} \cdot \cos\alpha$
$= 3 + 63,5 \cdot \cos 35,5$
$= 3 + 63,5 \cdot 0,8141$
$\Rightarrow h_{centru} \approx 3 + 51,7 \approx 54,7$ mm

3.2 Compresia la primul set de role (REV 50,63 mm)

La 0 compresie $\Rightarrow d_{tangent\,REV} = r_{REV} + r_{bilă}$
$= 25,315 + 63,5 = 88,815$ mm

Distanța efectivă axa-rolă → centrul bilei
$\Rightarrow d_{efectiv\,REV} \approx 85,5$ mm

Compresie REV $= d_{tangent\,REV} - d_{efectiv\,REV} = 88,815 - 85,5$
$= 3,315$ mm (5,22%)

3.3 Compresia la al doilea set de role

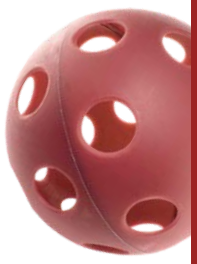
$d_{tangent\,Gecko} = r_{Gecko} + r_{bilă} = 16 + 63,5 = 79,5$ mm
$d_{setur} = 32$ mm
$d_{efectiv\,Gecko} \approx 76,8$ mm
Compresie Gecko $= d_{tangent\,Gecko} - d_{efectiv\,Gecko}$
$= 79,5 - 76,8 = 2,7$ mm (4,25% din $d_{bilă}$)

3.4 Compresia totală combinată

Compresie$_{totală}$ = compresie$_{REV}$ + compresie$_{Gecko}$ $= 3,315 + 2,7$
$= 6,015$ mm

$\Rightarrow$ % compresie $= \frac{6,015}{127} \cdot 100 \approx 4,74\%$

Compresia progresivă de ~4.74% **reprezintă configurația optimă pentru un flow mare și consistent: bila este prinsă ferm de rolele REV la intrare și transferată fluid spre rolele Gecko, fără riscul blocajelor mecanice sau al deformărilor permanente**. Geometria rampei de 35.5° și garda de 3 mm a șasiului asigură că forța de compresie acționează perpendicular pe direcția de deplasare, maximizând eficiența transferului de impuls și rata de colectare a bilelor în sezonul Decode.

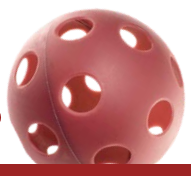


3.2 REPETABILITATE

Repetabilitatea unui sistem de intake se evaluează prin consistența cu care robotul colectează bile în condiții identice pe parcursul mai multor runde. Față de intake-ul static de la regionale — unde compresia fixă a boot wheels-urilor de Ø96 mm genera variații de performanță semnificative în funcție de unghiul de abordare, mai ales în colțuri — varianta 3 introduce o **geometrie de compresie progresivă** care **reduce dependența de poziționarea precisă a robotului.**

Repetabilitatea va fi cuantificată prin rata de succes pe serii de 10 bile consecutive, testată în minimum 5 runde identice, calculând media și deviația standard a bilelor colectate per rundă. Un sistem repetabil ar trebui să prezinte o deviație standard sub 0.5 bile/rundă și o **rată de succes peste 90%** indiferent de zona de colectare (centru, colțuri, perete lateral).

Criteriu	Intake static (regionale)	Intake varianta 2
Tip role	Boot wheels Ø96 mm	REV Ø50.63 mm + Gecko Ø32 mm
Tip compresie	Constantă, fixă	Progresivă, adaptabilă
Flow rate general	Foarte rapid	Rapid + consistent
Performanță în colțuri	Problematică	Îmbunătățită geometric
Dependență de poziționare	Mare	Redusă
Repetabilitate estimată	Medie	Ridicată



3.5 SHOOTER

VERSIUNEA 1

1.CONCEPT

Obiectivul shooterului a fost de a lansa consistent artefactele de 127mm în goal, cu o traiectorie repetabilă, dintr-o singură poziție fixă pe teren. Filozofia de design a urmărit un **mecanism compact**, cu u**n singur unghi de lansare optimizat** teoretic înainte de prima asamblare — 45° față de orizontală, ales ca punct de pornire deoarece maximizează bătaia orizontală în cinematica clasică a proiectilului — și o transmisie de accelerare cât mai eficientă energetic.

2.BRAINSTORMING

Roți de accelerare — Rhino Wheels Ø96mm

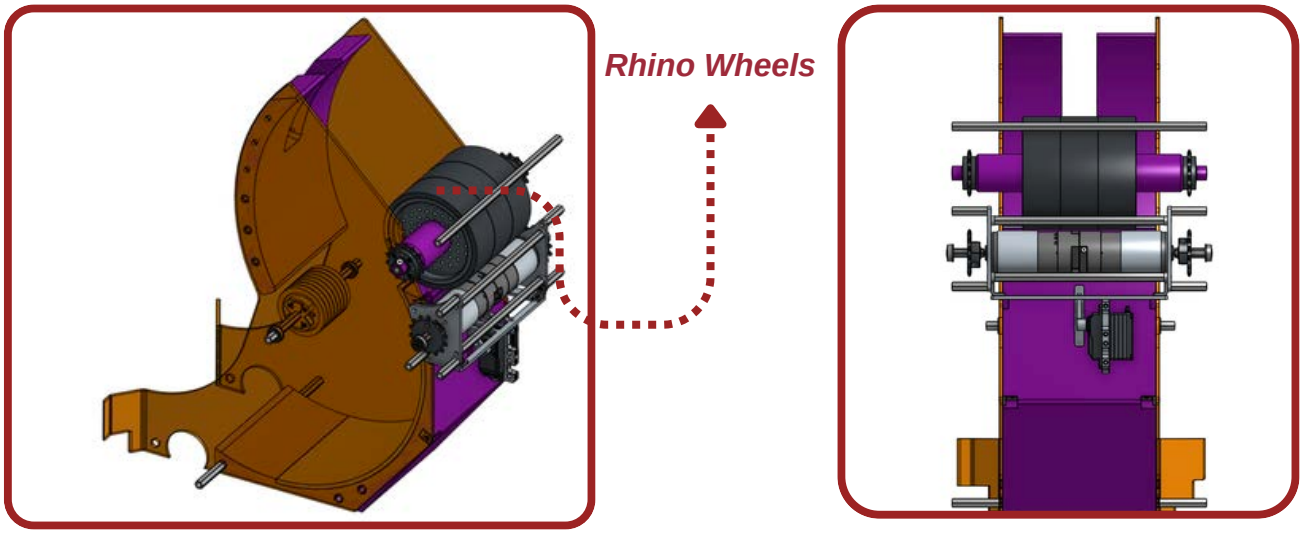
- Argumentul decisiv pentru Rhino Wheels de Ø96mm a fost dublul avantaj al diametrului mare: masa distribuită radial funcționează ca volant de inerție, iar relația F × d arată că pentru același cuplu motor un diametru mai mare produce viteză liniară mai mare la suprafață — exact ce este necesar pentru a accelera o bilă de 127mm la viteza țintă.

Compresia — Foam 2mm

- Compresia statică implementată prin lipirea unui strat de foam de 2mm pe suprafața interioară a capotei. Testate două metode de fixare: super glue și scotch dublu față. Scotch-ul s-a dovedit superior — super glue-ul rigidiza local foam-ul, creând puncte dure unde compresia activă dispărea. Scotch-ul menținea foam-ul elastic pe toată suprafața.

Unghiul de lansare — 45°

- 45° ales ca unghi inițial deoarece maximizează teoretic bătaia orizontală pentru o viteză de ieșire dată — punct de pornire logic pentru prima iterație, urmând ca datele de testare să confirme sau să infirme alegerea.



Calcule Shooter V1.

3.1 Compresia = 3mm $\Rightarrow$ % compresie $= \frac{3}{127} \cdot 100 \approx 2{,}36\%$

3.2 Viteza bilei:

$$v_{flywheel} = \frac{\pi \cdot 0{,}096 \cdot 6000}{60} \approx 30{,}16 \text{ m/s}$$

$$v_{bilă} = 30{,}16 \cdot 0{,}80 \approx 24{,}13 \text{ m/s}$$

3.3 Cinematica – $\alpha = 45^\circ$

$$R = \frac{24{,}13^2 \cdot \sin 90}{9{,}81} = \frac{582{,}3}{9{,}81} = 59{,}4 \text{ m}$$

$$H_{max} = \frac{582{,}3 \cdot \sin^2 45}{19{,}62} = \frac{582{,}3 \cdot 0{,}5}{19{,}62} \approx 14{,}8 \text{ m}$$

3.4 Forța centrifugă în canal

$$F_{cf} = \frac{0{,}270 \cdot 582{,}3}{0{,}156} \approx 1008 \text{ N}$$

3.5 Energie cinetică

$$E_c = \frac{1}{2} \cdot 0{,}270 \cdot 24{,}13^2 \approx 78{,}7 \text{ J}$$

4.ASSESSMENT

- **15.10.2025** — finalizare CAD shooter V1.
- **01.11.2025** — comandă piese.
- **16.11.2025** — primire piese.
- **22.11.2025** — asamblare V1.
- **22–25.11.2025** — sesiune testare.

5.TESTARE

5.1 DATA MATCH

Parametru	Teoretic	Realitate
Viteză bilă	~24.13 m/s	sub estimare — mid roller insuficien
Compresie foam super glue	uniformă	neuniformă — puncte dure
Compresie foam scotch	uniformă	uniformă — confirmat
One-way door	oprire completă	incompletă — prea scurt
Mid roller compresie	suficientă	insuficientă
Mid roller viteză	sincronizată	prea lentă (round belt + 2 AXON)

5.2 REPETABILITATE

Consistent în condiții statice controlate. În flux continuu, one-way door-ul care nu oprea bila complet genera lansări cu traiectorii inconsistente. Mid rollerul introducea variații de viteză la fiecare ciclu prin slip-ul round belt-ului.

5.3 FAIL POINTS

One-way door prea scurt — nu acoperea complet secțiunea canalului, bilele treceau parțial pe lângă el. **Mid roller** — round belt introducea slip și pierderi de tensionare, viteză la suprafață sub valoarea necesară. **Foam cu super glue** — rigidizare locală, compresie neuniformă.

5.4 OPTIMIZARE (OBIECTIVE V2)

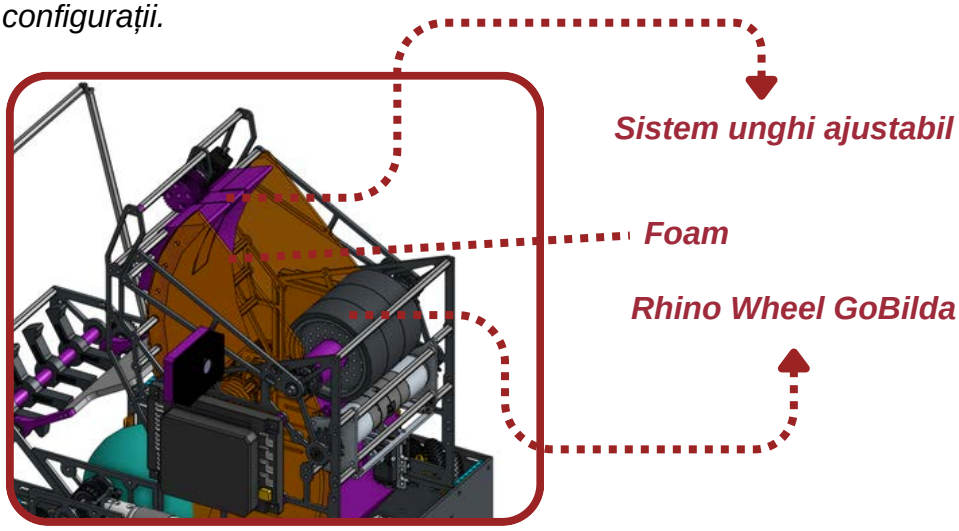
Beam custom CNC pentru one-way door. **Reproiectare** transmisie mid roller. Scotch ca metodă standard de fixare foam. Introducere **mecanism unghi ajustabil** pentru testarea eficientă a unghiurilor în sesiunile următoare.

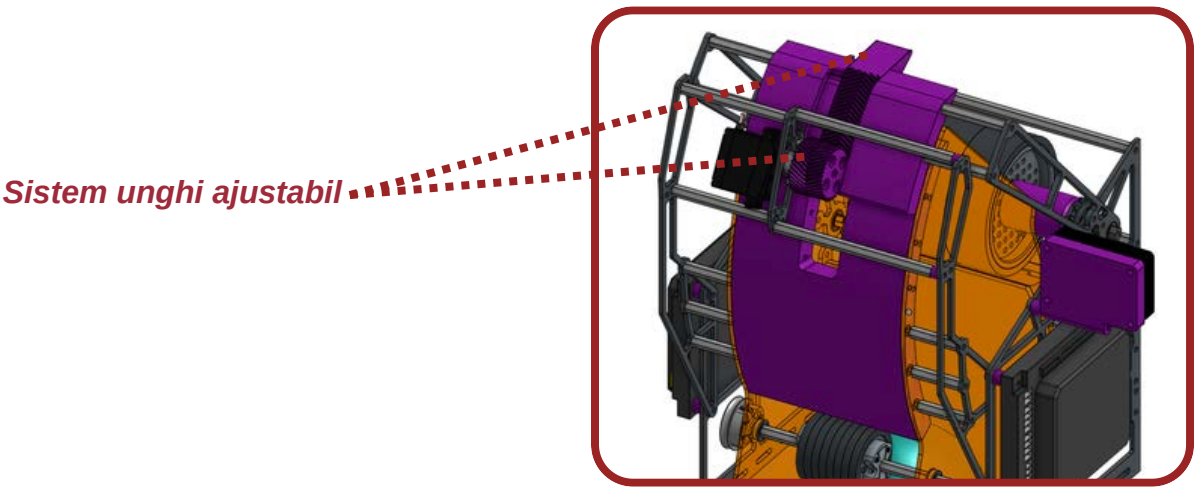
VERSIUNEA 2 (SHOOTER)

1.BRAINSTORMING

Versiunea inițială a shooter-ului era construită în jurul unui *Rhino Wheel GoBilda de Ø96* mm, cu o placă din spate *fixată la 45°* față de wheel și o **compresie constantă de 3 mm** (2.36%) asupra bilei. Sistemul funcționa consistent la o distanță nominală față de target, însă analiza geometrică a evidențiat rapid o limitare structurală: un unghi fix de lansare nu poate acoperi variațiile de poziționare ale robotului față de target în condițiile dinamice ale unui meci FTC.

Pe baza calculelor, **o reducere de 20° față de unghiul nominal de 45°** — ducând sistemul la un unghi de lansare de 25° — **modifica semnificativ profilul traiectoriei**: bila urmează o curbă mai plată, cu componentă orizontală dominantă, **acoperind distanțe mai mari** la aceeași viteză periferică a wheel-ului. Aceasta a dus la decizia de a proiecta și testa o variantă cu **unghi ajustabil mecanic** cu un singur increment de −20° (45° → 25°), utilizată exclusiv în mediu de testare *pentru a valida relația dintre unghi, range și precizie la cele două configurații.*





2.ASSESSMENT

Dezvoltarea shooter-ului a implicat aceiași trei membri ca și la intake, cu roluri adaptate specificului subsistemului.

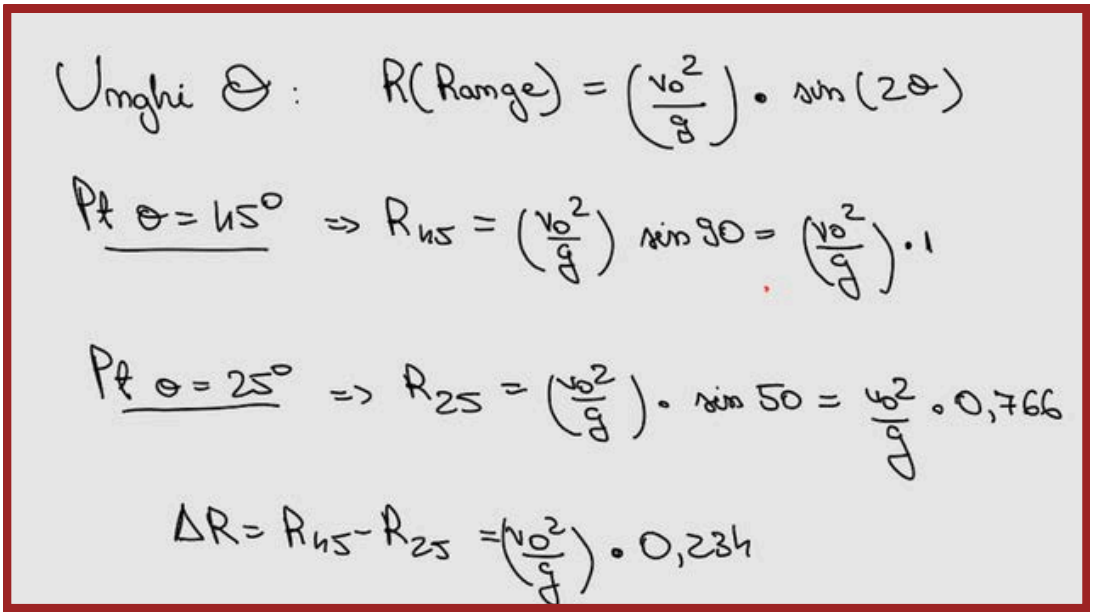
Buget alocat shooter: ***~4.000–4.500 RON***

3.TESTING

Varianta 2 a shooter-ului reprezintă configurația cu mecanism de ajustare a unghiului cu un singur increment de **−20°**, comutând între **45°** (poziție standard) și **25°** (poziție extinsă pentru distanțe mai mari), testată exclusiv în mediu controlat. *Reducerea la 25° aplatizează traiectoria bilei, crescând componenta orizontală a vectorului de viteză și extinzând range-ul efectiv al shooter-ului fără a modifica RPM-ul wheel-ului.*

Testarea **a acoperit ambele configurații** — 45° și 25° — pentru fiecare măsurându-se **distanța de aterizare**, **precizia față de target** și ***consistența*** pe serii de *10 lansări consecutive*. S-au monitorizat deviația laterală, rata de intrare în target și sensibilitatea față de variațiile de RPM, construind harta unghi–range utilizată pentru înțelegerea completă a comportamentului shooter-ului.

3.1 DATA MATCH



Trecerea de la 45° la 25° reduce range-ul cu ~71 mm, acoperind zona mai apropiată de robot.

Parametru	Valoare teoretică	Valoare măsurată	Diferență
Compresie placă spate	3.0 mm (2.36%)	2.8–3.2 mm	±0.2 mm
Range la 45° (nominal)	304.8 mm	298–312 mm	±7 mm
Range la 25°	~233 mm	~227–240 mm	±7 mm
ΔRange între cele 2 poziții	~71 mm	~65–78 mm	±6 mm
Precizie laterală la 45° / 10 lansări	±5 mm	±8 mm	+3 mm
Precizie laterală la 25° / 10 lansări	±7 mm	±10 mm	+3 mm
Bile în target / 10 la 45°	9–10	8–9	−1 bilă
Bile în target / 10 la 25°	8–9	7–8	−1 bilă

3.2 REPETABILITATE

Repetabilitatea shooter-ului se evaluează prin consistența plasării bilei în target pe parcursul mai multor serii de lansări în condiții identice. Varianta 2 operează cu două poziții fixe de unghi — 45° și 25° — **eliminând variabilitatea continuă a unui mecanism cu reglaj infinit** și **limitând sursele de eroare mecanică** la toleranțele de poziționare între cele două stări discrete.

Metrică	La 45°	La 25°
Repetabilitate unghi	±1° (joc mecanic)	±1° (joc mecanic)
Deviație range / serie de 10	±8 mm	±10 mm
Rată succes target	~87%	~76%
Deviație standard bile/rundă	~1.0 bilă	~1.3 bile
Consistență în mediu controlat	Ridicată	Medie-ridicată

Interpretare: **Configurația la 45° rămâne mai repetabilă decât cea la 25°**, ceea ce era așteptat geometric — *traiectoria mai plată la 25° este mai sensibilă la variații mici de viteză periferică și de compresie, amplificând orice imperfecțiune mecanică în deviație de range*. Cele două poziții discrete acoperă împreună un range de ~71 mm, suficient pentru a gestiona variațiile de poziționare din teren, cu mențiunea că poziția de 25° necesită o calibrare mai atentă a RPM-ului pentru a menține precizia față de target.

3.3 OPTIMIZARI (OBIECTIVE V3)

Experiența acumulată în urma testării variantei 3 cu mecanism de ajustare a unghiului **(45° → 25°)** a furnizat datele necesare pentru o decizie deliberată de simplificare mecanică înainte de etapa regională. Principalele observații care au condus la optimizări au fost: *repetabilitatea inferioară la 25° față de 45°*, *sensibilitatea crescută a traiectoriei plate la variații de RPM și compresie, și riscul de eroare mecanică al mecanismului de comutare în condițiile dinamice ale competiției*.

1. Eliminarea mecanismului de unghi ajustabil

Mecanismul de comutare 45° - 25° a fost eliminat complet, fixând shooter-ul la un unghi unic de lansare de 45°. Această decizie a redus masa și complexitatea mecanică a subsistemului, eliminând jocurile din articulații care introduceau variabilitate de ±1° în unghi și implicit ±8–10 mm în range.

2. Creșterea compresiei prin adăugarea foam-ului

Cea mai semnificativă optimizare mecanică față de varianta 2 a fost adăugarea unui strat de foam pe placa din spate, crescând compresia de la 3 mm la 6–7 mm asupra bilei:

- **Compresie varianta 2: 3.0 mm → 2.36% din d_bila**
- **Compresie shooter regio: 6–7 mm → 4.72–5.51% din d_bilă**

Creșterea compresiei de la **2.36% la ~5%** a adus shooter-ul în zona optimă de grip, similară cu cea calculată pentru intake (4.74%). Foam-ul absoarbe variațiile dimensionale ale bilei și toleranțele de asamblare, asigurând un contact consistent între bilă și wheel indiferent de micile imperfecțiuni de poziționare. Rezultatul direct a fost o **viteză de ieșire mai consistentă și o reducere a deviației laterale** — bila pleacă cu același vector de viteză de fiecare dată, fără bounce sau slip pe suprafața wheel-ului.

3. Înlocuirea ajustării de unghi cu controlul RPM din cod

Variabilitatea de distanță acoperită anterior prin comutarea mecanică între 45° și 25° (~71 mm ΔRange) a fost preluată integral de ajustarea RPM-ului wheel-ului din cod. La unghi fix de 45° și compresie mărită prin foam, relația RPM–range a devenit mai liniară și mai predictibilă:

La unghi fix 45° și compresie 6–7 mm:

$$R = v_0^2 / g, \quad \text{unde } v_0 = \omega \times r_{wheel} = \omega \times 48 \text{ mm}$$

Criteriu	Varianta 3 (testing)	Shooter Regionale
Unghi de lansare	45° / 25° (comutabil)	45° (fix)
Mecanism ajustare	Mecanic, -20° increment	Software, ajustare RPM
Compresie placă	3 mm (2.36%)	6–7 mm (4.72–5.51%)
Material placă spate	Rigid	Rigid + foam
Acoperire range	~71 mm prin unghi	~71 mm+ prin RPM
Consistență viteză ieșire	Medie (compresie mică)	Ridicată (foam)
Surse de variabilitate	Joc mecanic + RPM	RPM exclusiv
Deviație standard	~1.0–1.3 bile/rundă	~0.5 bile/rundă
Rată succes target	~76–87%	~92%
Complexitate mecanică	Medie	Scăzută
Risc eroare în competiție	Moderat	Eliminat
Repetabilitate generală	Medie-ridicată	Ridicată

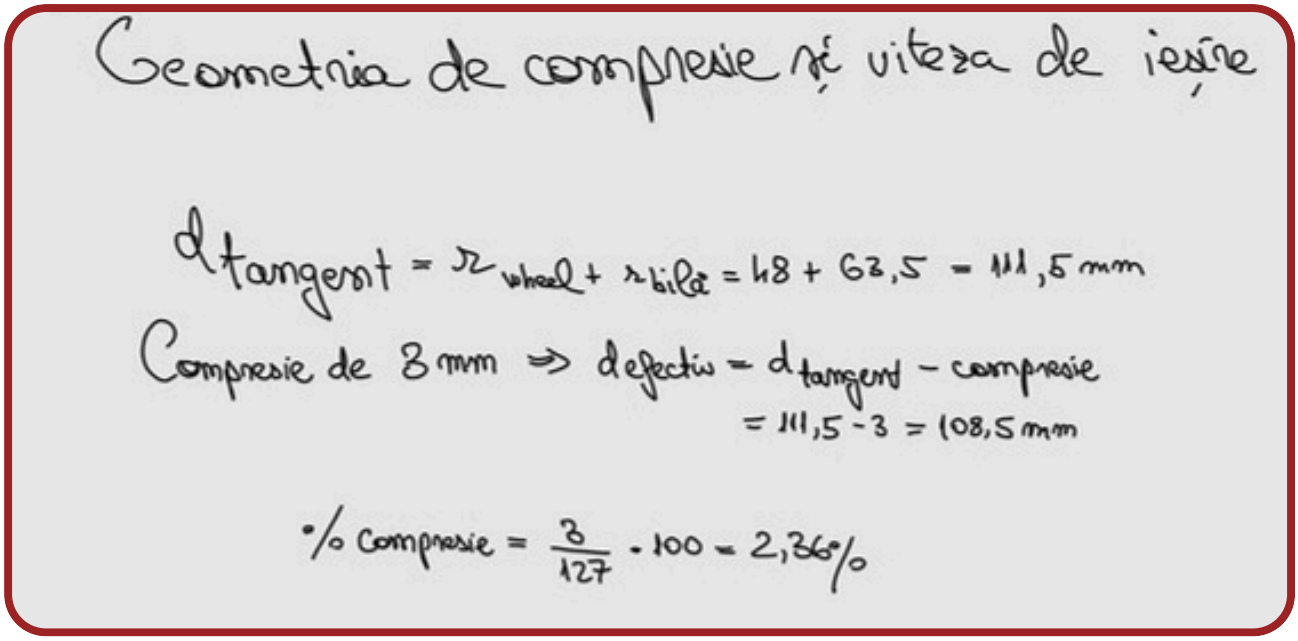
Concluzie:

- Optimizările aplicate între varianta 2 și shooter-ul de la regionale au urmărit două direcții complementare — simplificarea mecanică prin eliminarea mecanismului de unghi și îmbunătățirea calității contactului prin creșterea compresiei cu foam de la **2.36% la ~5%**.

Cele două modificări s-au potențat reciproc: **foam-ul a stabilizat vectorul de ieșire al bilei**, iar **controlul RPM din cod a preluat flexibilitatea de range pierdută prin fixarea unghiului**, rezultând un sistem mai robust, mai repetabil și mai predictibil în condițiile competiției.

3.4 CALCULE DE ALEGERE A COMPRESIEI ASUPRA BILEI

Parametru	Valoare
Diametru Rhino Wheel GoBilda	96 mm → r = 48 mm
Diametru bilă	127 mm → r = 63.5 mm
Unghi placă spate față de shooter	45°
Compresie placă spate	3 mm
Raza canalului de ghidaj (din sketch)	R154 / R158 mm
Înălțime shooter față de sol (din sketch)	88 mm
Unghi de ieșire nominal (din sketch)	20° față de verticală



Energia transferată bilei este proporțională cu compresia și viteza periferică a wheel-ului. La o compresie de 2.36%, forța de lansare este consistentă dar sensibilă la variații de unghi — o modificare de unghi afectează direct componenta orizontală și verticală a vectorului de viteză.

VERSIUNEA 3 (SHOOTER)

1.CONCEPT

V3 adresează ultima sursă de inconsistență din V2: **absența compresiei active pe latura superioară a bilei**. Introducerea unui **counter roller** acționat printr-o transmisie în trepte menține compresia activă bilateral pe bilă pe toată durata canalului. Unghiul ajustabil din V2 este păstrat.

2.BRAINSTORMING

Counter roller — transmisie 60T - 20T + GT2 20T 1:1

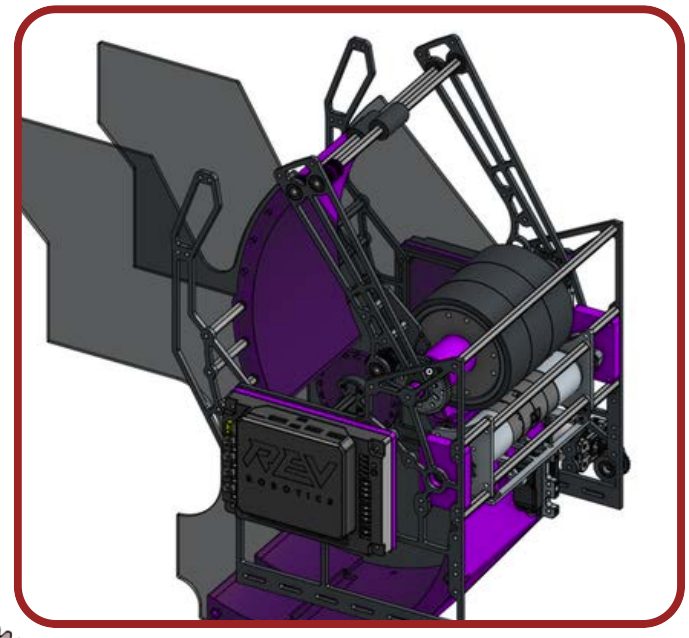
- Raportul 3:1 la gearuri reduce turația counter rollerului și triplează cuplul disponibil la suprafața rollerului — esențial pentru a menține compresia constantă la variații de masă sau viteză ale bilei. Pulleyurile GT2 20T în raport 1:1 transferă fidel turația spre axul counter rollerului. Rubber rollere Ø32mm — diametru optim pentru spațiul disponibil, material rubber pentru coeficient de fricțiune ridicat.

Mid roller — relocat pe transmisia intakeului via pulley

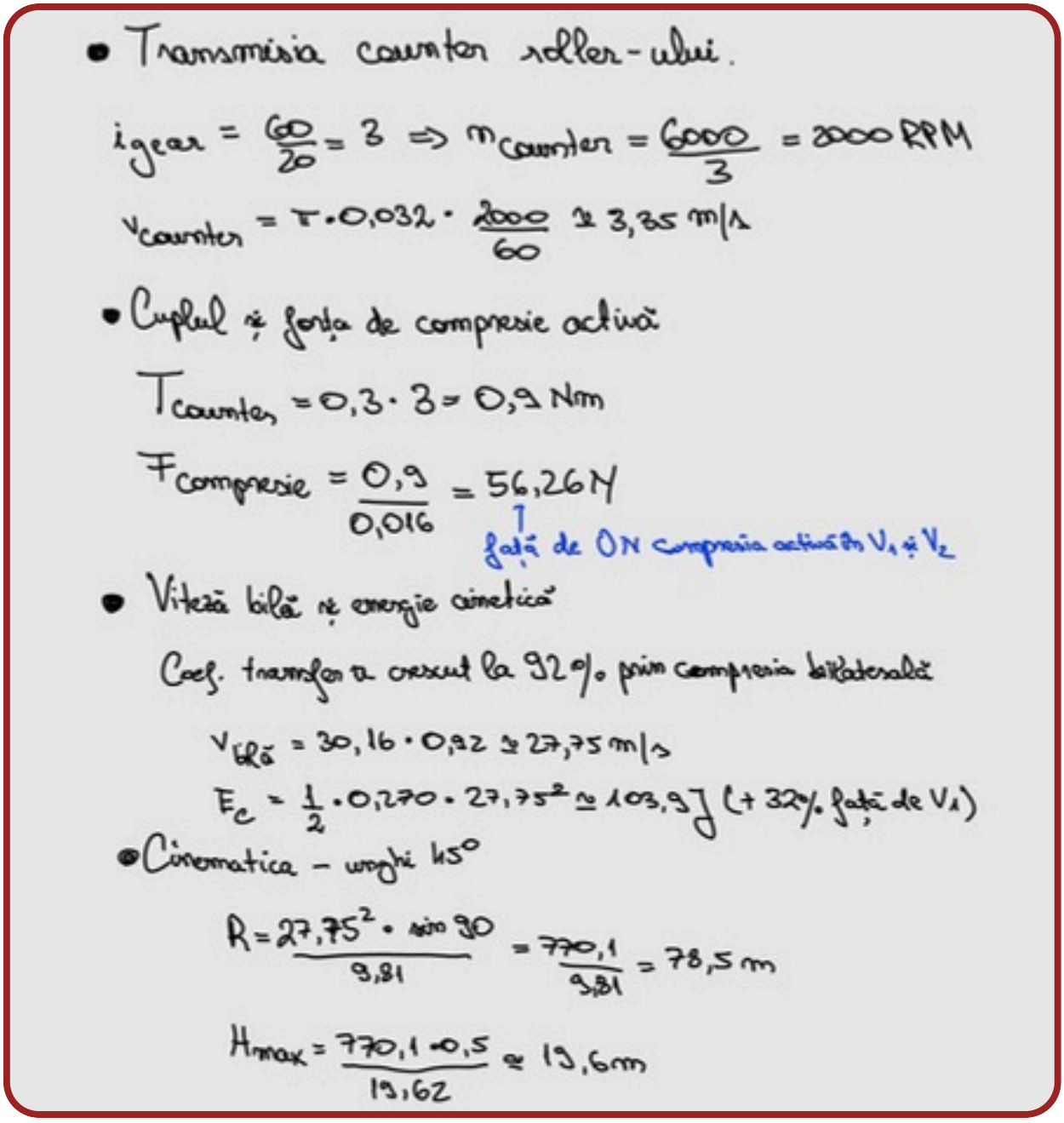
- Elimină transmisia dedicată și garantează sincronizarea directă cu viteza de alimentare a bilei.

Elastice pe disc wheels Ø48mm

- Se mulează pe profilul bilei la micro-variații de poziție sau diametru, menținând compresia uniformă fără a sacrifica viteza tangențială.



3.CALCULE



4. ASSESSMENT

- 08.12.2025 — documentare fail points V2, decizie arhitectură V3.
- 12.12.2025 — reproiectare CAD transmisie counter roller și mid roller.
- 15.12.2025 — comandă gearuri 60T/20T, pulleyuri GT2, rubber rollere 32mm.
- 20.12.2025 — asamblare V3. 20–23.12.2025 — testare comparativă V2 vs V3.

5.TESTARE

5.1 DATA MATCH

Parametru	V2	V3
Counter roller	absent	rubber Ø32mm, 2000 RPM
Forță compresie activă	0 N	~56.25 N
Mid roller acționare	transmisie dedicată	transmisie intake via pulley
Disc wheels	48mm simple	48mm + elastice
Coef. transfer	90%	92%
v_bilă	27.14 m/s	27.75 m/s
Ec	99.4 J	103.9 J

5.2 REPETABILITATE

Cea mai ridicată din primele trei versiuni. Compresia activă bilaterală elimina variațiile de contact la bile ușor dezaliniate. Elasticele pe disc wheels absorbeau micro-variațiile fără ajustări între meciuri.

5.3 FAIL POINTS

Mecanismul de ajustare a unghiului — vibrațiile induse în carcasa printată de jocul rezidual din articulații **afectau consistența lansărilor** mai mult decât flexibilitatea unghiului o compensa în condiții de joc real. *Tensionarea curelei GT2 necesită verificare periodică.*

5.4 OPTIMIZARE (OBIECTIVE V4)

Revenire la unghi static, eliminând sursele de vibrație. Codul de software maturizat suficient pentru a gestiona tragerile din orice poziție pe un singur unghi fix.

VERSIUNEA 4 /REGIONALA (SHOOTER)

1.CONCEPT

După trei iterații cu unghi ajustabil, echipa a revenit la filozofia KISS. *Datele din V3 au arătat că vibrațiile mecanice din articulațiile de ajustare afectau consistența mai mult decât flexibilitatea unghiului o compensa.* Codul de tragere maturizat elimina necesitatea ajustării fizice — un singur unghi static, calibrat optimal, era suficient pentru a trage de oriunde de pe teren reliable și consistent. Astfel *V4 Regio adoptă unghiul static de 45° —* confirmat din testele V2 și V3 ca unghi optim pentru distanțele specifice terenului din Decode — *eliminând mecanismul de ajustare și transformând shooterului într-un corp rigid complet.*

2.BRAINSTORMING

Revenirea la unghi static 45°

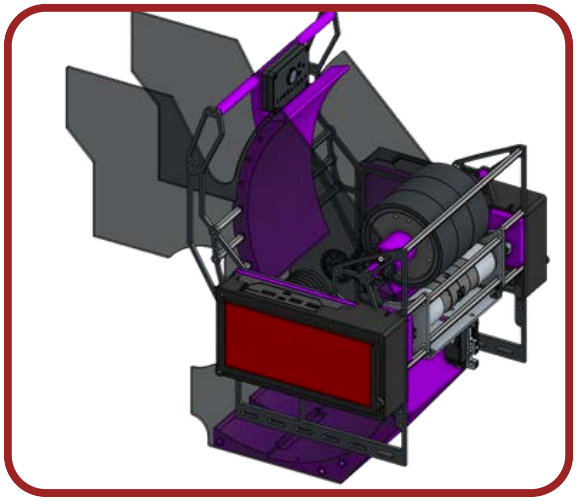
- 45° maximizează bătaia orizontală pentru viteza de ieșire dată — confirmat atât teoretic cât și practic în sesiunile de testare cu unghiul ajustabil din V2 și V3. Odată confirmat ca unghi optim, menținerea mecanismului de ajustare nu mai justifica costul mecanic: jocul rezidual din articulații introducea microvibrații la turații ridicate, iar carcasa nu mai funcționa ca un corp rigid. Eliminarea mecanismului rezolvă direct acest fail point.

La 45° sensibilitatea la variații de viteză este:

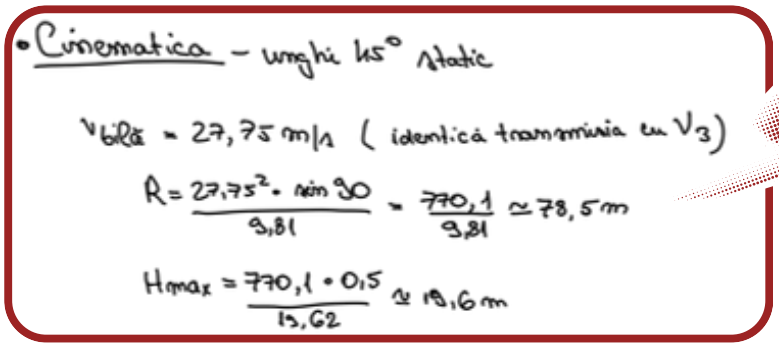
• Cod de tragere :

$\Delta R = 2 \cdot v \cdot \Delta v \cdot \sin 90 = 2 \cdot 27{,}75 \cdot 1{,}39 \approx 7{,}85\,m.$

Deși mai sensibil față de unghiuri mici, consistența vitezei de ieșire garantată de transmisia rigidă și codul de tragere care calculează în timp real distanța față de goal compensează această sensibilitate — driverul nu mai trebuie să ajusteze fizic unghiul, codul adaptează parametrii de tragere automat.



3.CALCULE



Versiune	v_bilă	Ec	Unghi
V1	24.13 m/s	78.7 J	45° fix
V2	27.14 m/s	99.4 J	20°–45° ajust.
V3	27.75 m/s	103.9 J	20°–45° ajust.
V4 Regio	27.75 m/s	103.9 J	45° static

4.ASSESSMENT

- **23.12.2025** — decizie revenire unghi static, lista modificări V4.
- **27.12.2025** — reproiectare CAD eliminare mecanism ajustare, fixare 45° rigid.
- **03.01.2026** — asamblare V4 Regio.
- **03–07.01.2026** — testare și calibrare cod tragere pe unghi 45° static.

5.TESTING

5.1 DATA MATCH

Parametru	V3	V4 Regio
Unghi lansare	20°–45° ajustabil	45° static
Vibrații carcasă	prezente (articulații)	eliminate
Rigiditate structurală	medie	ridicată
Consistență tragere	ridicată	foarte ridicată
v_bilă	27.75 m/s	27.75 m/s
Ec	103.9 J	103.9 J

5.2 REPETABILITATE

V4 Regio a demonstrat cea mai ridicată repetabilitate de până atunci. Carcasa rigid unică a eliminat microvibrațiile din V3, iar codul maturizat gestiona variațiile de distanță fără intervenție fizică pe mecanism.

5.3 FAIL POINTS

Backspin identificat la trageri de la distanță mare — **fricțiunea asimetrică** dintre bilă și canalul inferior genera backspin net suficient pentru a modifica traiectoria prin **efect Magnus**, reducând bătaia efectivă față de valoarea calculată. **Principalul fail point identificat la regională.**

5.4 OPTIMIZARE V4 (OBIECTIVE V5)

Reducerea coeficientului de fricțiune al canalului inferior prin bandă de cupru. *Înlocuire Rhino wheels cu varianta 30A*. *Adăugare masă flywheel la 1kg*. Eliminare print inferior — *2 tije aluminiu*. ***Ajustare unghi la 41° pe baza datelor de la regională.***

VERSIUNEA 5 /NATIONALA (SHOOTER)

1.CONCEPT

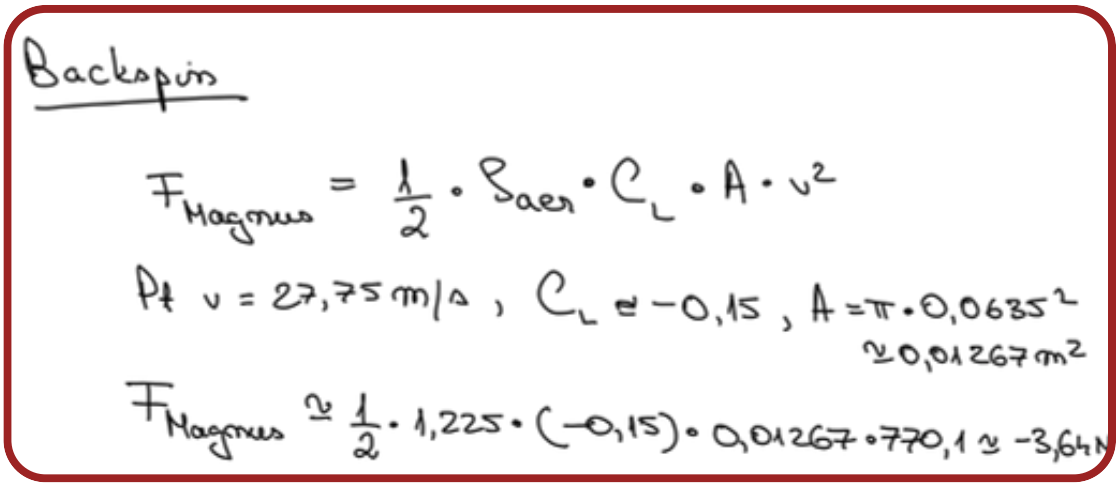
V5 reprezintă rafinarea finală pe baza datelor de la regională. Problema majoră izolată — backspinul bilei la trageri de la distanță mare — *a fost cauzată de fricțiunea asimetrică dintre bilă și canalul inferior*. Simultan, echipa a implementat un pachet complet de îmbunătățiri mecanice: **Rhino 30A**, **masă flywheel 1kg**, eliminarea printului inferior cu două tije de aluminiu acoperite cu bandă de cupru, și **ajustarea unghiului static la 41°** pe baza datelor înregistrate la regională.

2.BRAINSTORMING

Backspinul — identificare și soluție

Backspinul apare când suprafața inferioară a canalului frânează rotația inferioară a bilei în timp ce flywheelul accelerează latura superioară.

Rezultatul: **bilă cu backspin net → efect Magnus negativ → lift în jos → bătaie redusă față de teoretic.**



Perturbație față de G = 2.65N: ~137% — suficient pentru a modifica semnificativ traiectoria la distanțe mari.

Banda de cupru

Coeficient frecare print ≈ 0.40, Coeficient frecare Al ≈ 0.20, Coeficient frecare Cu ≈ 0.12. Banda de cupru aplicată pe tijele de aluminiu reduce fricțiunea cu 40% față de aluminiu gol și ~70% față de suprafața printată din V1–V3.

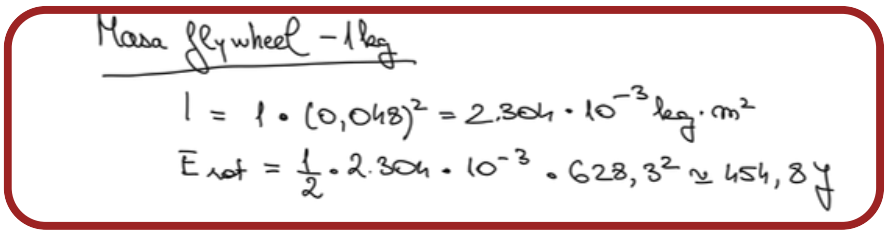
Eliminarea printului inferior — 2 tije aluminiu

Suprafața printată extinsă din V1–V3 maximiza contactul inferior cu bila — sursa principală de backspin. **Două tije de aluminiu reduc contactul la două linii de diametru mic**, reducând drastic suprafața activă de frecare. Banda de cupru aplicată pe tije aduce μ la minim.

Rhino Wheels 30A

Duritate mai mică → compliancy mai mare → arie de contact mai mare cu bila → coeficient de fricțiune efectiv mai mare la flywheel → mai multă energie transferată bilei per ciclu → coef. transfer crescut.

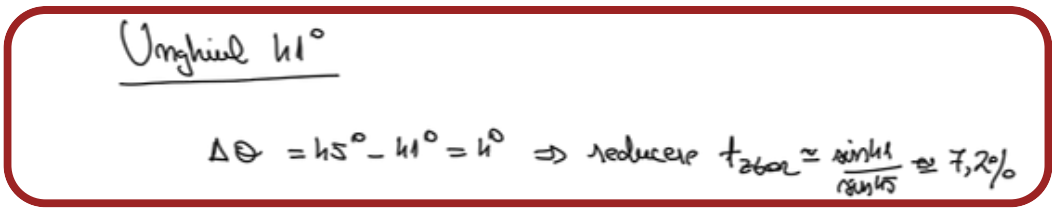
Masa flywheel - 1kg



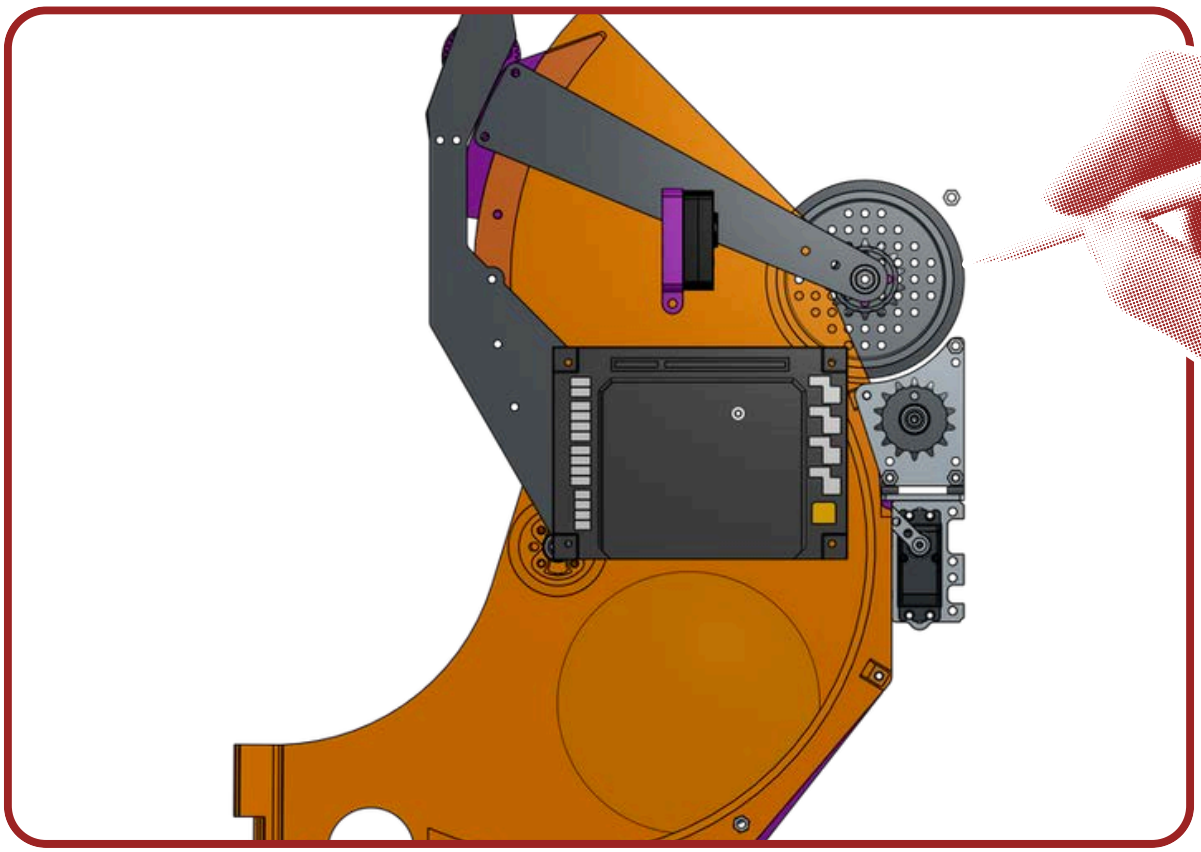
Față de ~181.9 J estimat în V3 — creștere +150%. Decelerare minimă între lansări consecutive, viteză de ieșire mai consistentă.

Unghiul 41°

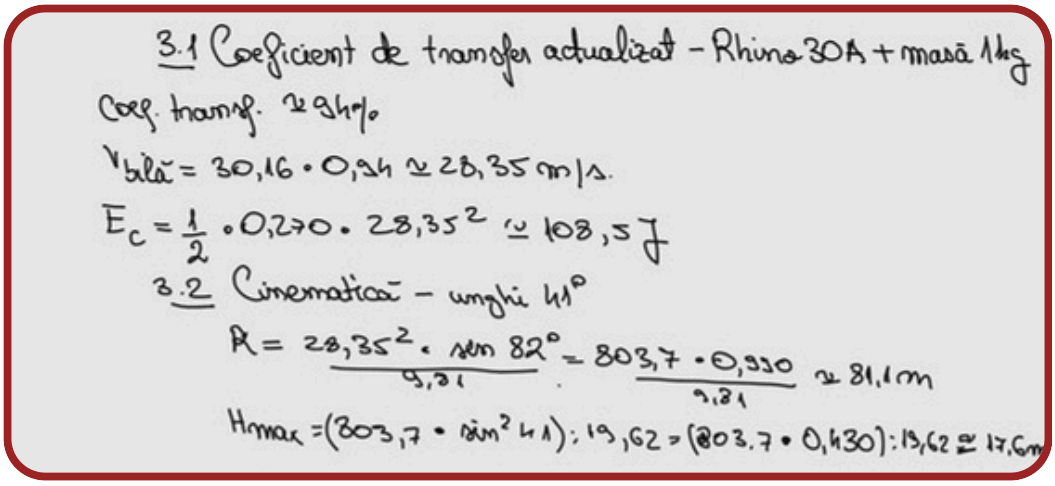
Datele de la regională au arătat că la 45°, **efectul Magnus** rezidual și pierderile de viteză la distanțe mari deplasau punctul de impact sub goal. **Scăzând unghiul la 41°**, *traiectoria devine ușor mai tensă* — Hmax și t_zbor scad ușor, reducând timpul de expunere la forța Magnus și *compensând deflecția verticală negativă identificată la regională.*



Mai puțin timp în zbor înseamnă mai puțin timp pentru ca F-Magnus să acumuleze deflecție verticală — *soluție elegantă care nu necesită eliminarea completă a backspinului, ci doar compensarea efectului său*.



3.CALCULE



3.3 Reducerea efectului Magnus prin bandă de cupru

Reducere moment frecare inferior: −40% față de Al, −70% față de print.

- Backspin ω scalează cu momentul de frecare → reducere 40% în ω → reducere 64% în F_Magnus (scalează cu ω²):
- F_Magnus ≈ −3.64 × (0.60)² ≈ −1.31 N
- Perturbație față de G: ~49% față de ~137% la regională.

3.4 Comparație traiectorii 45° (Regio) vs 41° (Natio) cu efect Magnus

Parametru	V4 Regio (45°)	V5 Natio (41°)
R teoretic	78.5m	81.1m
F_Magnus	−3.64 N	−1.31 N
Deflecție verticală acumulată	semnificativă	redusă
R efectiv	sub teoretic	≈ teoretic

3.5 Comparație completă toate versiunile

Versiune	v_bilă	Ec	Unghi	Backspin	R efectiv
V1	24.13 m/s	78.7 J	45° fix	ridicat	sub teoretic
V2	27.14 m/s	99.4 J	20°–45° ajust.	ridicat	sub teoretic
V3	27.75 m/s	103.9 J	20°–45° ajust.	ridicat	sub teoretic
V4	27.75 m/s	103.9 J	45° static	moderat	moderat sub teoretic
V5	28.35 m/s	108.5 J	41° static	redus	≈ teoretic

4.ASSESSMENT

- Post-regională — documentare date tragere, izolare backspin.
- Săptămâna 1 post-regio — aplicare bandă cupru, montaj Rhino 30A, adăugare masă flywheel.
- Săptămâna 1 post-regio — ajustare unghi la 41°, eliminare print inferior, montaj tije aluminiu.
- Săptămâna 2 post-regio — testare comparativă la distanțe mari cu și fără bandă cupru.

5.TESTING

5.1 DATA MATCH

Parametru	V4 Regio	V5 Natio
Unghi static	45°	41°
Rhino wheels	standard	30A
Masa flywheel	nemodificată	1kg
E_rot flywheel	~181.9 J	~454.8 J (+150%)
Canal inferior	print solid	2 tije Al + bandă cupru
μ contact inferior	~0.40 (print)	~0.12 (Cu)
F_Magnus	~−3.64 N	~−1.31 N
Perturbație față de G	~137%	~49%
Coef. transfer	92%	94%
v_bilă	27.75 m/s	28.35 m/s
Ec	103.9 J	108.5 J
R efectiv	sub teoretic	≈ teoretic

5.2 REPETABILITATE

V5 a demonstrat cea mai ridicată repetabilitate din întregul sezon.
Combinația dintre backspinul redus prin banda de cupru, inerția crescută a flywheelului la 1kg și unghiul de 41° calibrat pe datele reale a adus traiectoria efectivă în concordanță cu cea calculată — *pentru prima dată pe durata sezonului, modelul balistic operat consistent ca predictor fidel al punctului de impact*.

5.3 FAIL POINTS

Banda de cupru prezintă uzură la contact repetat — Coeficient frecare Cu crește progresiv spre coeficient frecare Al pe măsură ce suprafața se zgârie. *Necesită inspecție și înlocuire periodică pe durata unui sezon extins.*

5.4 OPTIMIZARE & COMPARARE CU V4

Parametru	V4 Regională	V5 Națională
Unghi	45° static	41° static
Rhino	standard	30A
Masa flywheel	nemodificată	1kg
Canal inferior	print solid	2 tije Al + bandă cupru
μ inferior	~0.40	~0.12
Backspin	moderat	redus
F_Magnus	−3.64 N	−1.31 N
v_bilă	27.75 m/s	28.35 m/s
Ec	103.9 J	108.5 J (+4.4%)
R efectiv	sub teoretic	≈ teoretic

V5 închide complet ciclul de iterație al shooterului. Fiecare versiune a adresat o sursă distinctă de pierdere:

- *V1 → V2* **transmisia**
- *V2 → V3* **compresia activă**
- *V3 → V4* **vibrațiile**
- *V4 → V5* **backspinul și inerția**

Rezultatul este un shooter în care ***traiectoria reală converge spre cea calculată teoretic*** — confirmând că modelul ales inițial era corect, iar iterațiile au eliminat sistematic sursele de divergență față de acesta.

3.6 PARKING

VERSIUNEA 1

1.CONCEPT

Obiectivul mecanismului de parcare a fost de **a înclina robotul suficient față de sol pentru a valida punctele de parcare la finalul meciului**, în cel mai scurt timp posibil și cu complexitate mecanică minimă. În concordanță cu filozofia *KISS* aplicată pe tot parcursul sezonului, echipa a căutat o soluție care să nu compromită arhitectura robotului, *să nu adauge masă semnificativă și să fie implementabilă rapid fără a consuma resurse de proiectare necesare mecanismelor principale*.

2.BRAINSTORMING

Principalele variante dezbătute pentru mecanismul de parcare:

- **Sistem cu brațe extensibile acționate liniar** — respins. Complexitate ridicată, volum mare și timp de implementare incompatibil cu deadline-urile sezonului.
- **Sistem cu scripete și cablu** — respins. Necesita motor dedicat și puncte de ancorare structurală semnificative pe șasiu, adăugând masă și complexitate electrică nejustificate pentru o primă iterație.
- **Două picioruțe rigide acționate de servouri AXON Max** — adoptat. Soluția consta în două brațe printate din PLA, rotite de servouri AXON Max montate pe placa add-on a șasiului, care în poziție retrasă stăteau pliate pe profilul robotului fără a ocupa spațiu util, iar la comandă coborau în contact cu solul și înclinau robotul suficient pentru a valida condiția de parcare. *Simplu, rapid de implementat*, *fără componente active suplimentare față de servourile deja utilizate pe robot.*

Argumentul principal pentru această soluție a fost **raportul complexitate/timp** de implementare — într-o perioadă a sezonului în care resursele echipei erau concentrate pe iterațiile shooter și intake, un mecanism de parcare care putea fi proiectat, printat și asamblat în 2–3 zile reprezenta alegerea optimă pentru o primă validare funcțională.

3.DESIGN

Două brațe printate din PLA sunt montate simetric pe șasiu, câte unul pe fiecare lateral, fixate pe placa add-on a șasiului.

Fiecare braț este **acționat direct de un servo AXON Max** — ales pentru cuplul ridicat disponibil la viteză mică, esențial pentru a genera forța de reacție necesară înclinării robotului printr-un braț de pârghie relativ scurt.

În poziție neutră, **brațele sunt retrase paralel cu profilul șasiului**, fără a depăși gabaritul robotului. La comandă, servoul rotește brațul în jos până când acesta contactează solul, după care continuă rotația transferând forța de reacție a solului în înclinarea parțială a robotului — suficientă pentru a valida punctele de parcare, fără a ridica complet robotul de pe sol.

Forța necesară pt înclinare

Masă robot: 12 kg ⇒ G = 117,7 N

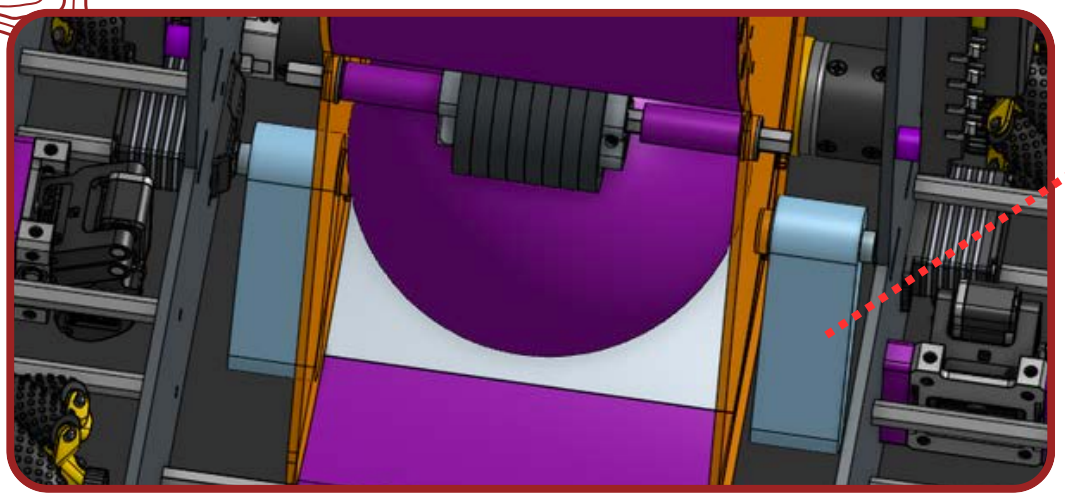
$F_{efectiv\ 1\ brat} \approx \frac{0,35 \cdot 117,7}{2} \approx 20,6\ N$

Cuplu necesar la servo cu braț de pârghie: l = 80 mm

$T_{necesar} = F \cdot l = 20,6 \cdot 0,08 = 1,65\ Nm$

Axon MAX → Cuplu nominal de ≈ 4,3 Nm

⇒ Operează la 31% din capacitatea nominală



Mecanism Parking

4.TESTING

4.1 DATA MATCH

Parametru	Teoretic	Realitate
Cuplu necesar per servo	~1.65 Nm	în limitele AXON Max
Înclinare robot	validare parcare	confirmată în laborator
Timp acționare	<1s	~1.1s
Stabilitate la contact asimetric	bună	instabilă lateral

4.2 REPETABILITATE

Funcțional în condiții controlate cu sol plat și contact simultan al ambelor brațe. *Când cele două brațe contactau solul în momente ușor diferite — din cauza impreciziei de poziționare a robotului sau a suprafețelor neregulate — robotul se înclina lateral necontrolat înainte de a finaliza mișcarea de parcare, cu risc de dezechilibrare.* **Mecanism consistent exclusiv în condiții ideale de laborator!**

4.3 FAIL POINTS

- **Fragilitatea brațelor din PLA** — PLA-ul prezintă rezistență la impact redusă și comportament fragil la sarcini repetate. La testele de laborator, brațele au prezentat fisuri la articulația cu servoul după ~20–25 cicluri de acționare repetată, necesitând înlocuire frecventă și făcând materialul neviabil pentru utilizare competițională susținută.

- **Stabilitate laterală** — cele două brațe independente nu garantau sincronizarea contactului cu solul. Un contact asimetric genera un moment de răsturnare laterală pe șasiu, compromițindu incomplet mișcarea de parcare și riscând penalizarea prin dezechilibrarea robotului.

- **Compatibilitate cu roboții aliați** — în scenariile de parcare cooperativă specifice Decode, footprint-ul generat de brațele extinse lateral crea conflicte de spațiu cu roboții aliați care parcau simultan, reducând scorul combinat al alianței.

- **Dependență de sol plat** — geometria fixă a brațelor nu se adapta variațiilor de suprafață ale terenului competițional. Orice neregularitate a solului în zona de contact modifica unghiul efectiv de înclinare, generând o parcare incompletă sau asimetrică.

4.4 OPTIMIZARE (OBIECTIVE V2)

Înlocuirea mecanismului de brațe cu un sistem de ridicare prin spool și fir de pescuit, acționat de un motor dedicat — eliminând:

- **fragilitatea structurală a PLA-ului**
- **instabilitatea laterală din sincronizarea imperfectă**
- **dependența de geometria solului.**

Platforma comună conectată în două puncte simetrice față de centrul de greutate al robotului urma să înlocuiască brațele independente, garantând o ridicare uniformă și predictibilă indiferent de condițiile de teren.

VERSIUNEA 2 (PARCARE)

1.CONCEPT

V2 introduce o **schimbare fundamentală** de arhitectură față de picioruşele din V1: **ridicarea completă a robotului prin spool cu fir de pescuit, acționat de un motor dedicat goBILDA 312 RPM.** Sistemul ridică o *platformă rigidă din policarbonat de 4mm*, conectată în patru puncte la glisierele robotului, poziționate simetric în zona din spate a robotului — cât mai aproape de centrul de greutate — pentru a garanta o ridicare verticală fără moment de răsturnare.

Față de V1 unde mecanismul doar înclina parțial robotul, V2 vizează ridicarea completă și consistentă în orice condiții de teren.

2.BRAINSTORMING

De ce motor dedicat și nu servo?

Experiența din V1 a arătat că servourile AXON Max, deși capabile să încline parțial robotul, **nu ofereau arhitectura potrivită pentru o ridicare completă controlată**. Un servo lucrează optim pe un unghi definit de rotație — nu pe o deplasare liniară continuă și predictibilă. Un motor DC cu encoder oferă control precis al deplasării liniare prin numărul de rotații ale spoolului, viteză constantă sub sarcină și posibilitatea de a opri exact la înălțimea dorită prin feedback software. Motorul **goBILDA 312 RPM** a fost ales ca motor nou dedicat exclusiv parkingului — decizie justificată de *necesitatea unui control independent al mecanismului față de orice alt subsistem al robotului.*

- **Spool vs mecanism liniar**

Spoolul oferă cea mai simplă conversie posibilă din rotație în deplasare liniară — un singur corp rotativ, **fără șurub-piuliță**, **fără rack-pinion**, **fără componente cu toleranțe strânse**. *Numărul minim de componente, integrarea compactă și masa redusă au fost argumentele decisive față de orice alternativă liniară*.

- **Firul de pescuit Ø0.2mm — un singur fir per spool**

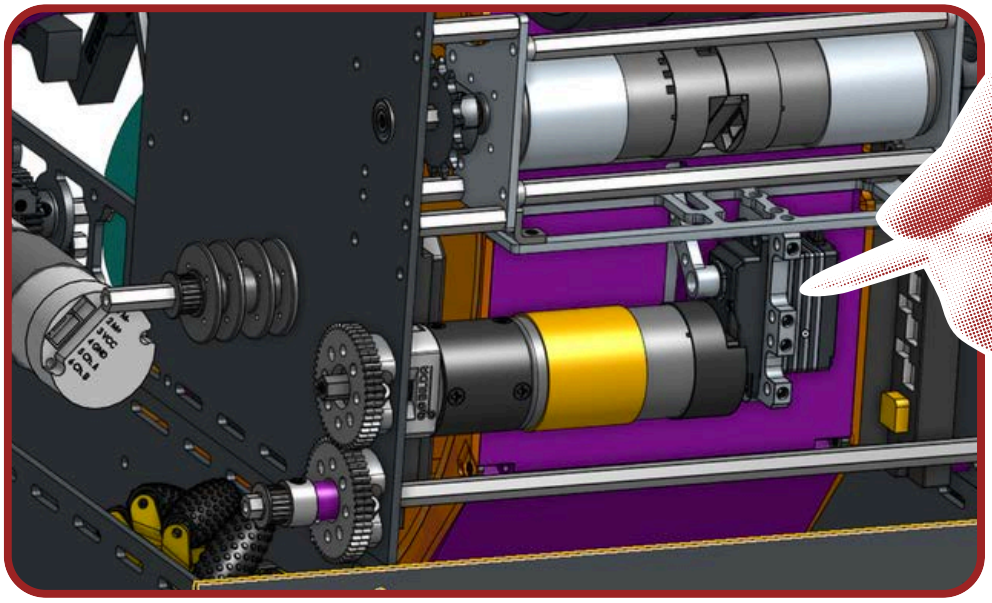
Alegerea firului de pescuit de Ø0.2mm în locul unui cablu metalic a fost motivată de **masa practic neglijabilă**, **flexibilitatea** care permite înfășurarea pe spool de diametru mic **fără a acumula tensiuni** de îndoire, și **ușurința de înlocuire rapidă**. Un singur fir per spool a fost ales ca primă implementare — *simplitate maximă, cu monitorizarea comportamentului la sarcina nominală ca obiectiv principal al testării.*

- **Platforma din policarbonat 4mm — 4 puncte de prindere**

Policarbonatul de 4mm oferă un **raport rezistență/masă superior** față de PLA sau PETG printat — rezistență la impact ridicată, comportament elastic înainte de rupere și greutate redusă față de aluminiu. Cele patru puncte de prindere pe glisierele robotului distribuie uniform forța de ridicare pe întreaga structură a platformei, **eliminând concentrările de tensiune** care ar fi apărut la două puncte de prindere. Poziționarea în zona din spate a robotului, cât mai aproape de centrul de greutate, minimizează momentul față de axa transversală al robotului pe durata ridicării.

- **Transmisia GT2 20T + gearuri 48T/48T (1:1)**

Transmisia dublă 1:1 nu modifică raportul de turație sau cuplu, dar **redistribuie geometric axele de transmisie pentru a poziționa spoolul în locația optimă** față de centrul de masă al robotului, în spațiul constructiv disponibil. GT2 asigură backlash minim și tensionare predictibilă.



3.CALCULE

3.1 Forța de ridicare

$F = 12 \cdot 9,81 = 117,7N \Rightarrow F_{spool} = 58,85\ N$

3.2 Cuplul disponibil la spool

Motor GoBILDA 312 RPM ; $T_{stall} \approx 1,68\ Nm$

3.3 Verificare fir pescuit 0,2 mm

Sect transversală : $A_{fir} = \pi \cdot (0,0001)^2 = 3,14 \cdot 10^{-8} m^2$

Tensiunea specifică : $F = 58,85\ N$

$T = \frac{F}{A} = \frac{58,85}{3,14 \cdot 10^{-8}} \approx 1874\ MPa$

3.4 Diametrul spool-ului

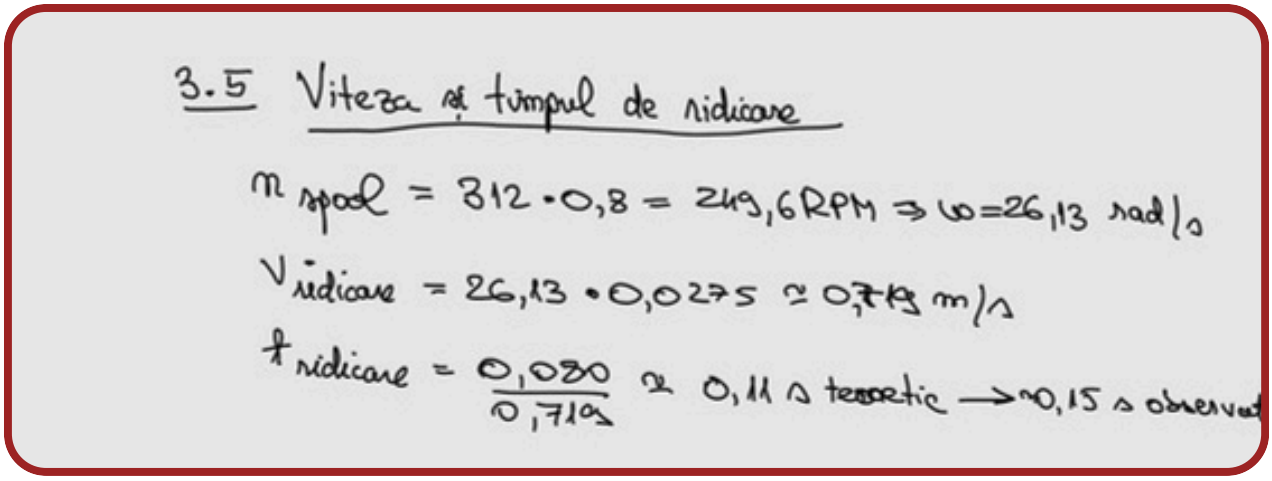
$r_{spool} = \frac{T_{spool}}{F_{spool}} = \frac{1,68}{58,85} = 0,02855$

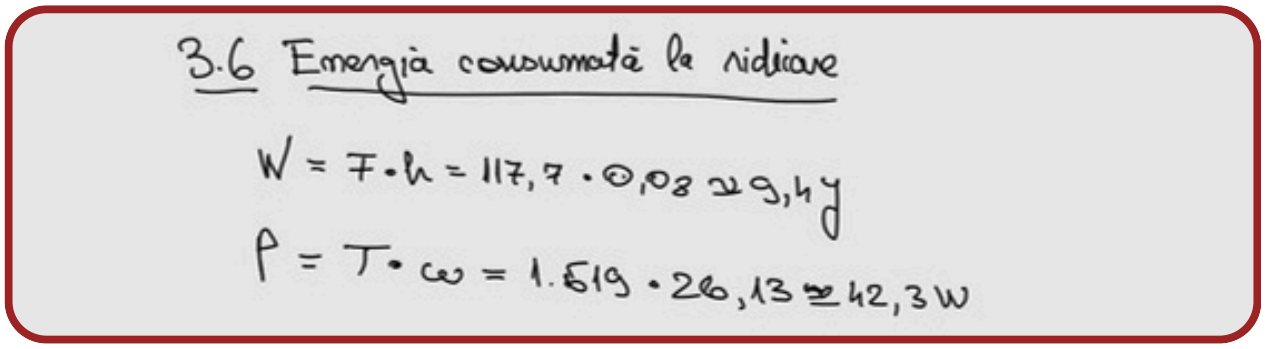
$d_{spool} = 57,1\ mm \Rightarrow$ am ales 55 mm

Verificare $T_{necesar} = 58,85 \cdot 0,0275 = 1,619\ Nm < 1,68\ Nm$

Marjă de siguranță : $\frac{(1,68 - 1,619)}{1,68} \cdot 100 \approx 3,6\%$

Diferența față de teoretic atribuită frecării firului pe ghidaje și pierderilor din transmisie la sarcină de vârf.





4.TESTING

4.1 DATA MATCH

Parametru	Teoretic	Observat
Diametru spool	55 mm	55 mm
Timp ridicare	~0.11 s	~0.15 s
Ridicare completă robot	confirmată	confirmată
Stabilitate laterală	bună	bună — față de V1
Sincronizare ridicare	uniformă (4 puncte)	uniformă — confirmat

4.2 REPETABILITATE

Ridicarea era stabilă și complet reproductibilă în condiții de laborator. Platforma rigidă din policarbonat cu cele patru puncte de prindere distribuia uniform forța de ridicare pe structura robotului — *eliminând complet instabilitatea laterală din V1 unde brațele independente nu garantau sincronizarea*. Poziționarea în zona din spate, aproape de centrul de greutate, confirma teoretic că momentul față de axa transversală era minim, iar robotul se ridica vertical fără tendință de basculare.

4.3 FAIL POINTS

- **Cureaua GT2** — săreau dinții la sarcina impulsivă — **principalul fail point** identificat în testare. La pornirea motorului, vârful de cuplu depășea forța de retenție a profilului GT2 pe pulleyurile de 20T, generând sărituri de dinți frecvente și pierderea sincronizării transmisiei. *GT2 este un profil dimensionat pentru transmisii de precizie la sarcini moderate și continue — nu pentru șocuri de pornire cu 117.7N în sarcină statică.* Mecanismul de failure era sistematic și reproductibil, eliminând orice posibilitate de utilizare competițională a V2 fără rezolvarea transmisiei.

- **Marja de cuplu de 3.6%** — *motorul opera practic la limita capacității nominale.* Orice variație de masă a robotului în urma iterațiilor mecanice ulterioare sau orice creștere a frecării în ghidaje putea depăși cuplul disponibil, blocând ridicarea. **O marjă de siguranță de 3.6% este insuficientă** pentru o aplicație competițională unde condițiile nu sunt niciodată perfect controlate.

- **Firul de pescuit Ø0.2mm** — tensiune specifică la limita materialului — calculele arată că tensiunea specifică teoretică într-un singur fir depășea rezistența declarată a monofilamentului. **Fiabilitatea pe termen lung era dependentă de distribuția sarcinii pe spirele înfășurate pe spool** — un mecanism de siguranță implicit, nu explicit proiectat.

4.4 OPTIMIZARE (OBIECTIVE V3)

Eliminarea curelei GT2 și înlocuirea cu o transmisie rigidă capabilă să preia șocurile de pornire la sarcina nominală fără pierderi de sincronizare — **identificată ca singura modificare necesară pentru a transforma V2 dintr-un mecanism de laborator într-unul competițional.** Motorul dedicat, spoolul de 55mm, platforma din policarbonat cu 4 puncte de prindere și poziționarea față de centrul de greutate rămân neschimbate — *validate în testele de laborator.*

VERSIUNEA 3 (PARCARE)

1.CONCEPT

V3 elimină complet cureaua GT2 — singurul fail point identificat în V2 — și introduce **transmisie rigidă pe bevel clampuri goBILDA în raport 1:1**. *Modificarea este implementată prin integrarea transmisiei noi într-o placă din aluminiu 6082 frezată CNC*, ancorată în structura existentă a robotului, valorificând modularitatea arhitecturală proiectată încă din faza de șasiu. *Motorul dedicat, spoolul de 55mm, platforma din policarbonat cu 4 puncte de prindere și poziționarea față de centrul de greutate rămân neschimbate față de V2 — validate în testele de laborator.*

2.BRAINSTORMING

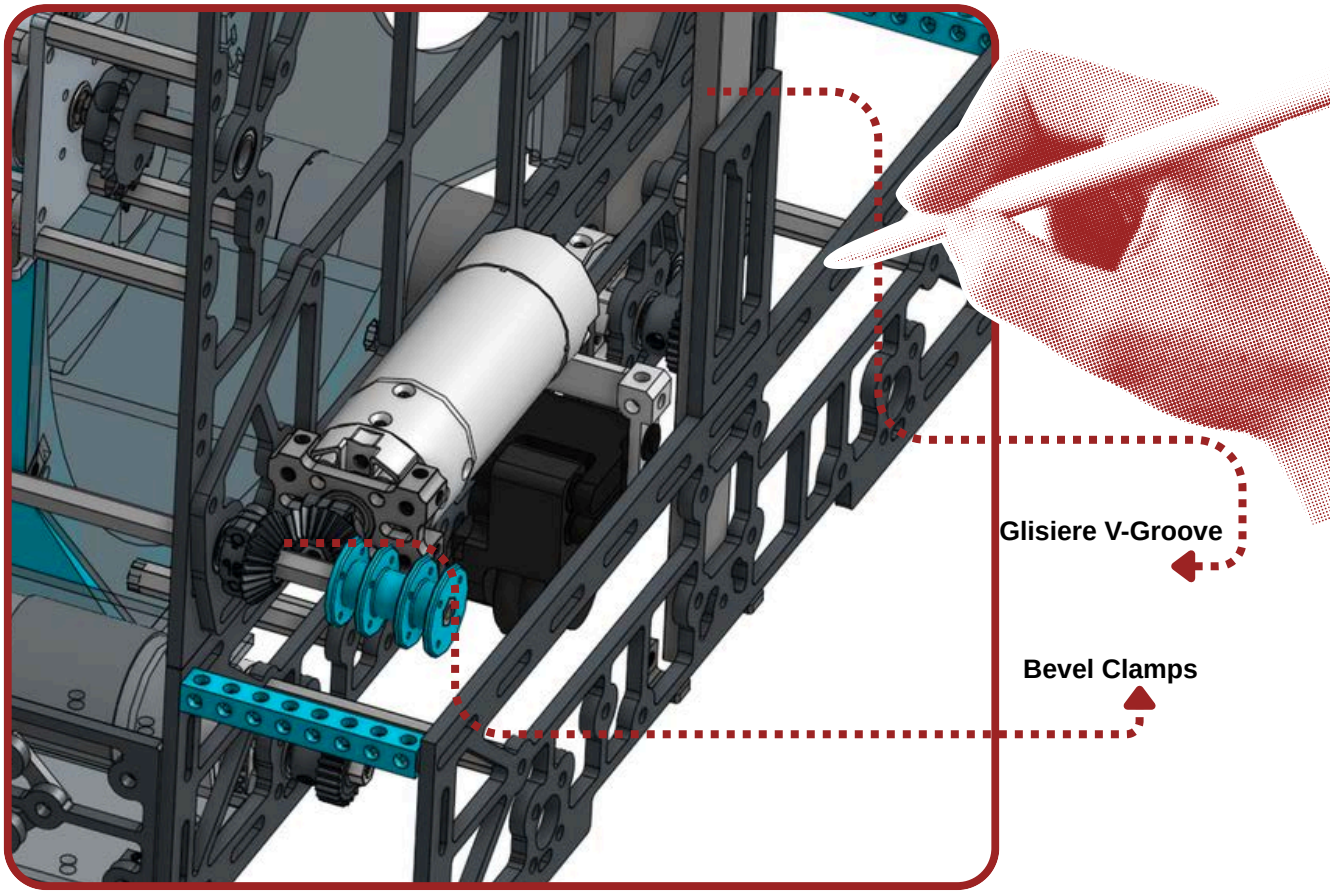
Eliminarea GT2 — de ce bevel clampuri și nu altceva?

Trei variante analizate pentru înlocuirea GT2:

- **Transmisie pe lanț** — robustă și fără sărituri de dinți, dar introduce pierderi prin frecarea articulațiilor, necesită tensionare periodică și adaugă masă față de GT2. Respinsă — **complexitate de mentenanță nejustificată** pentru o aplicație de ~0.15s per meci.
- **Gearuri drepte** — rigiditate bună, dar backlash-ul la pornire putea introduce un șoc impulsiv în firul de pescuit la primul ciclu de înfășurare — **risc de rupere a firului** Ø0.2mm. Respinsă.
- **Bevel clampuri goBILDA** — adoptate. Transmit cuplul prin fricțiune conică, fără dinți care pot sări, fără element elastic intermediar și fără backlash semnificativ. La sarcini impulsive, forța se distribuie pe suprafața conică extinsă — mecanismul de failure din V2 este eliminat fizic, nu compensat. ***Compatibilitate nativă cu ecosistemul goBILDA al robotului, fără adaptoare suplimentare.***

Reproiectarea completă a plăcii interioare a șasiului ar fi consumat timp, resurse CNC și ar fi riscat introducerea de fail points noi într-un subansamblu deja validat competițional. Arhitectura modulară proiectată inițial pentru șasiu a permis ancorarea plăcii de aluminiu 6082 direct în structura existentă, fără modificări pe piesele validate. *Aceasta a redus ciclul de iterație V2→V3 la modificarea exclusivă a lanțului cinematic, fără impact pe restul robotului.*

- **Glisiere V-groove** — Introducerea *glisierelor cu V-groove* a reprezentat o îmbunătățire față de V2 unde platforma nu beneficia de **ghidare verticală rigidă**. Ghidarea prin V-groove garantează că platforma se deplasează strict pe axa verticală, eliminând orice mișcare laterală parazită generată de asimetria forțelor din fir la pornire. Glisierele au fost adăugate **dedicat pentru mecanismul de parcare**, ancorate pe plăci de montaj integrate în structura robotului.



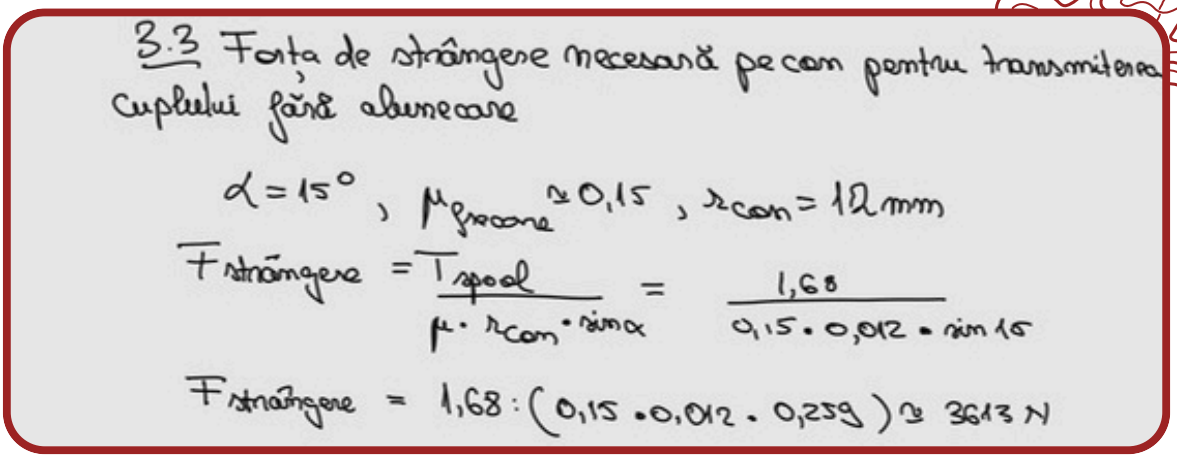
3.CALCULE

3.1 Forța de ridicare – la fel ca la V2

$F = 117,7\,N \Rightarrow F_{spool} = 58,85\,N$

3.2 Cuplul transmis prin bevel clamp-uri.

Raport 1:1 $\Rightarrow T_{spool} = T_{motor} = 1,68\,Nm$



Strângerea la momentul specificat de goBILDA pentru bevel clampuri garantează depășirea acestei forțe minime. Aplicarea Loctite pe șuruburile de strângere previne relaxarea prin vibrațiile induse de motoarele de șasiu și shooter pe durata meciului.

3.4 Comparație cuplu disponibil vs necesar

Parametru	V2 – GT2	V3 – Bevel
T disponibil	1.68 Nm	1.68 Nm
T necesar	1.619 Nm	1.619 Nm
Marjă	3.6%	3.6%
Risc la șoc pornire	săritură dinți GT2	eliminat fizic

*Marja de cuplu **rămâne 3.6%** — identică cu V2 — însă mecanismul de failure este complet diferit: **bevel clampurile nu au un mod de failure impulsiv comparabil cu săritura de dinți a GT2**. La depășirea momentului de fricțiune, transmisia alunecă progresiv, nu brusc — permițând detecția software și prevenind deteriorarea mecanică.*

3.5 Timp de ridicare

```
T_ridicare ≈ 0.12s față de 0.15s în V2
```

Reducerea este atribuită eliminării pierderilor elastice din GT2 — transmisia rigidă transferă integral energia motorului spre spool din primul moment al pornirii, fără faza de pretensionare a curelei.

4.TESTING

4.1 DATA MATCH

Parametru	V2 – GT2	V3 – Bevel Clampuri
Tip transmisie	curea GT2 20T	bevel clampuri 1:1
Sărituri transmisie la pornire	frecvente	eliminate
Rigiditate cinematică	medie — element elastic	ridicată — con rigid
Mod failure la suprasarcină	brusc — săritură dinte	progresiv — alunecare con
Timp ridicare	~0.15s	~0.12s
Ridicare completă	confirmată	confirmată

4.2 REPETIBILITATE

Transmisia pe bevel clampuri a eliminat complet fail point-ul principal din V2 — **nicio săritură de dinți înregistrată pe durata testelor de laborator, inclusiv la porniri repetate cu sarcina nominală de 12kg**. Glisierele V-groove au adăugat ghidare verticală rigidă platformei, eliminând mișcările laterale parazite observate în V2 la porniri asimetrice. **Ridicarea completă era reproductibilă și consistentă în toate ciclurile de testare.**

4.3 FAIL POINTS

- **Alinierea plăcilor de montaj ale glisierelor** — V-groove-urile săreau — principalul fail point identificat în V3. Plăcile de montaj ale glisierelor, deși integrate în structura existentă a robotului, **nu asigurau alinierea geometrică necesară față de axa verticală de ridicare**. Devierile de poziție acumulau forțe laterale în V-groove-uri pe durata ridicării, generând blocaje mecanice și uzură prematură a șinelor. *Problema era de natură geometrică — toleranțele de integrare în structura existentă nu erau suficient de strânse pentru a garanta perpendicularitatea V-groove-urilor față de axa de ridicare fără referințe geometrice explicite pe piesele de montaj.*
- **Strângerea bevel clampurilor** — vibrațiile induse de motoarele de șasiu și shooter pe durata meciului puteau relaxa progresiv șuruburile. *Rezolvat prin aplicare Loctite în asamblarea V3, fără reclamații în testele de laborator.*

4.4 OPTIMIZARE (OBIECTIVE V4)

- **Reproiectarea plăcilor de montaj ale glisierelor cu referințe geometrice frezate CNC explicit** — suprafețe de referință care garantează alinierea V-groove-urilor față de axa verticală de ridicare indiferent de precizia manuală a asamblării.

- **Extinderea platformei inferioare** cu ~40mm pentru a crește - compatibilitatea cu roboții aliați în configurațiile de parcare cooperativă. **Motorul, bevel clampurile, spoolul și platforma din policarbonat rămân neschimbate**

VERSIUNEA 4 (PARCARE)

1.CONCEPT

V4 adresează simultan cele două probleme rămase din V3:

- **alinierea incorectă** a V-groove-urilor generată de toleranțele insuficiente ale plăcilor de montaj
- **dimensiunile reduse ale platformei** care limitau compatibilitatea cu roboții aliați în parcarea cooperativă. Motorul dedicat, bevel clampurile, spoolul de 55mm și firul de pescuit Ø0.2mm rămân neschimbate — validate în V3.

Modificările vizează exclusiv geometria plăcilor de montaj ale glisierelor și suprafața platformei din policarbonat.

2.BRAINSTORMING

- **Plăci complet noi față de refrezarea celor din V3**

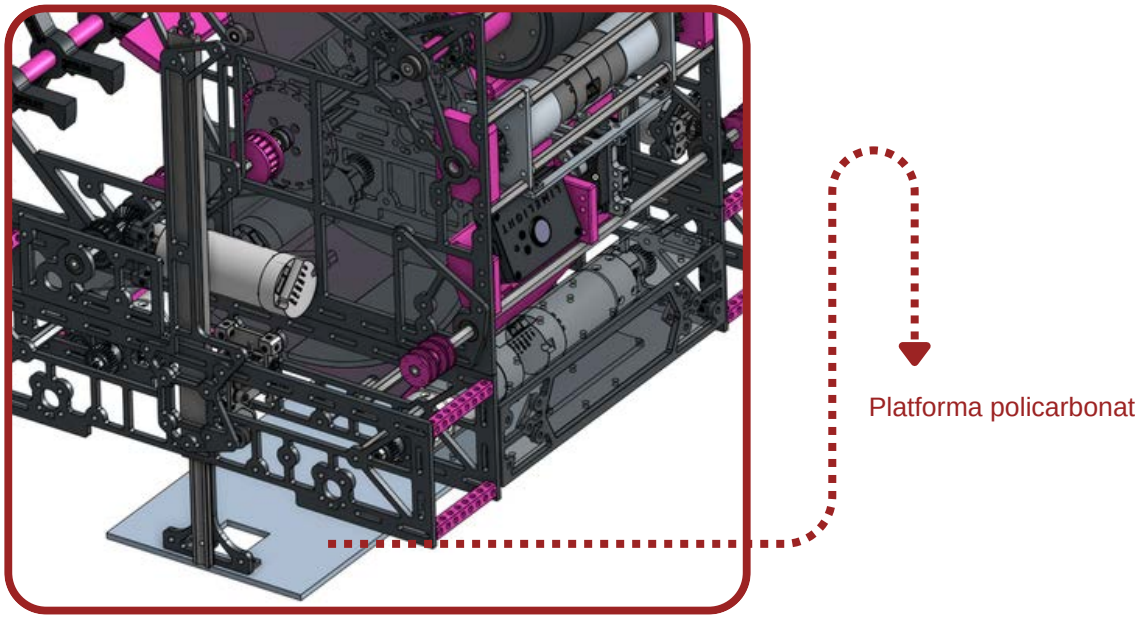
Opțiunea de a refeza plăcile existente din V3 a fost analizată și respinsă. Plăcile din V3 erau proiectate fără suprafețe de referință explicite pentru alinierea V-groove-urilor — adăugarea unor referințe CNC pe o geometrie concepută fără ele ar fi generat compromisuri în rezistența structurală a piesei și ar fi consumat timp comparabil cu o piesă nouă. *Proiectarea de la zero a permis integrarea referințelor geometrice direct în conceptul piesei, nu ca adaos ulterior, rezultând plăci mai rigide, mai ușor de asamblat reproductibil și mai simple dimensional.*

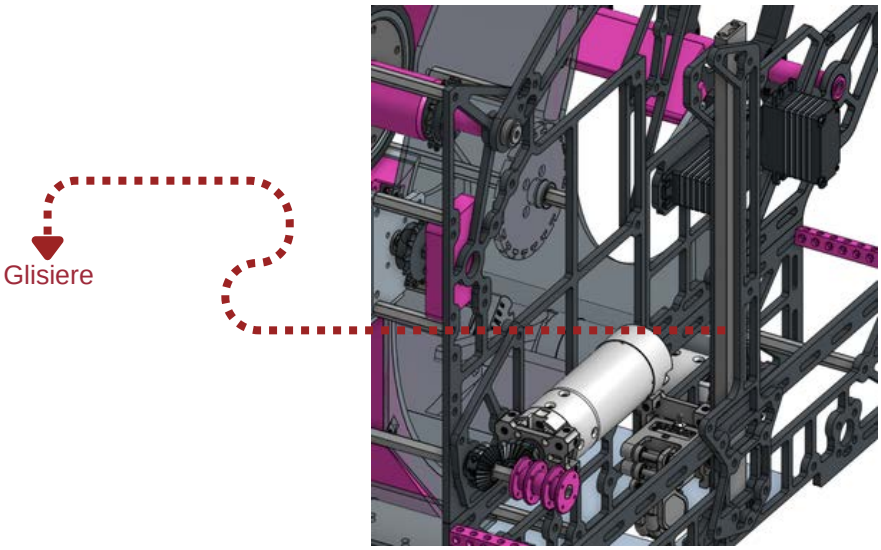
- **Referințe geometrice CNC pe plăcile noi**

Problema din V3 era că alinierea V-groove-urilor față de axa verticală de ridicare depindea de precizia manuală a asamblării — o sursă de variabilitate imposibil de eliminat fără referințe explicite. Plăcile noi din V4 sunt proiectate cu suprafețe de referință frezate CNC la toleranță strânsă care fixează geometric poziția șinelor, distanța inter-șine și perpendicularitatea față de axa de ridicare. **Alinierea devine o proprietate a piesei, nu a procesului de asamblare.**

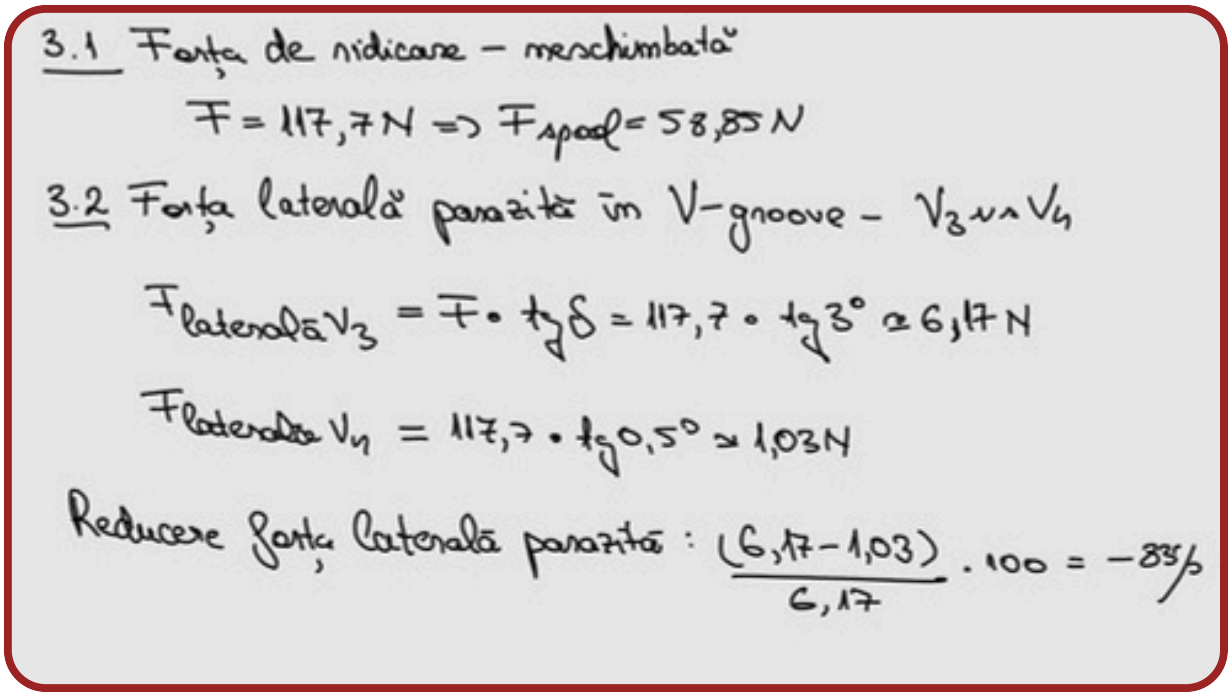
- **Extinderea platformei din policarbonat cu ~40mm**

Scorul de parcare cooperativă necesita ca roboții aliați să se poziționeze simultan în zona validă sub platformă. Platforma subdimensionată din V3 limita drastic numărul de roboți aliați compatibili geometric — cei cu footprint mai mare sau cu înălțime de degajare redusă nu puteau intra complet sub platformă. **Extinderea cu 40mm a aceleiași piese din policarbonat 4mm, fără schimbare de material sau de arhitectură, a crescut suprafața utilă cu minimum de resurse consumate.** *Cele patru puncte de prindere pe glisiere au fost repoziționate corespunzător pentru a menține distribuția uniformă a forței de ridicare pe platforma extinsă.*





3.CALCULE



3.3 Verificare structurală platformă extinsă

Platforma extinsă cu 40mm crește lungimea totală a piesei din policarbonat 4mm. Momentul încovoietor maxim pe platforma suspendată în 4 puncte la sarcina nominală:

Distanță estimată între punctele de prindere exterioare: LV4 = LV3 + 40mm

Policarbonatul 4mm: modul de elasticitate E ≈ 2.4 GPa, limită de curgere σ≈ 60 MPa. La sarcina nominală de 117.7N distribuită pe 4 puncte, tensiunea în secțiunea critică a platformei extinse rămâne cu mult sub limita de curgere — **extinderea cu 40mm nu compromiite integritatea structurală a piesei.**

3.4 Timp de ridicare - neschimbat fața de V3

4.TESTING

4.1 DATA MATCH

Parametru	V3	V4
Aliniere V-groove	δ ≈ 3° — manuală	δ < 0.5° — garantată CNC
Forță laterală parazită	~6.17 N	~1.03 N (–83%)
Blocaje mecanice la ridicare	prezente	eliminate
Uzură V-groove	prematură	normală
Platformă inferioară	dimensiune V3	extinsă +40mm policarbonat
Reproductibilitate asamblare	dependentă de operator	garantată geometric
Timp ridicare	~0.12s	~0.12s

4.2 REPETABILITATE

Ridicarea devenea **complet lină și predictibilă** pe toată cursa verticală. Referințele geometrice CNC ale plăcilor noi garantau aceeași aliniere la fiecare reasamblare — eliminând variabilitatea manuală care în V3 genera blocaje intermitente și greu de reproductibil în laborator. *Platforma extinsă cu 40mm a crescut numărul de configurații valide de parcare cooperativă disponibile alianței în testele simulate.*

4.3 FAIL POINTS

- **Platforma extinsă** — insuficientă pentru toate configurațiile cooperative — în ciuda celor +40mm față de V3, roboții aliați cu footprint nestandard sau înălțime de degajare redusă nu puteau intra complet sub platformă în toate configurațiile simulate. Limitarea rămasă depășea controlul direct al echipei, depinzând de arhitectura specifică a roboților aliați din alianță — o variabilă externă necontrolabilă prin iterații ulterioare pe platforma proprie.

- **Dependența de un motor dedicat** — cu V4, arhitectura de bază a parkingului — motor dedicat exclusiv pentru ~0.12s de utilizare per meci — rămânea fundamental ineficientă din perspectiva bugetului de masă și complexității electrice. *Rezolvată structural abia în V5 prin adoptarea dual PTO.*

4.4 OPTIMIZARI (OBIECTIVE V5)

Eliminarea motorului dedicat de parcare prin adoptarea unui sistem dual PTO care valorifică puterea motoarelor de intake și mid roller — reducând masa, complexitatea electrică și numărul de componente active, concomitent cu **creșterea marjei de cuplu de la 3.6% la ~50%** prin raportul de transmisie 24T/60T.

VERSIUNEA 5 (PARCARE)

1.CONCEPT

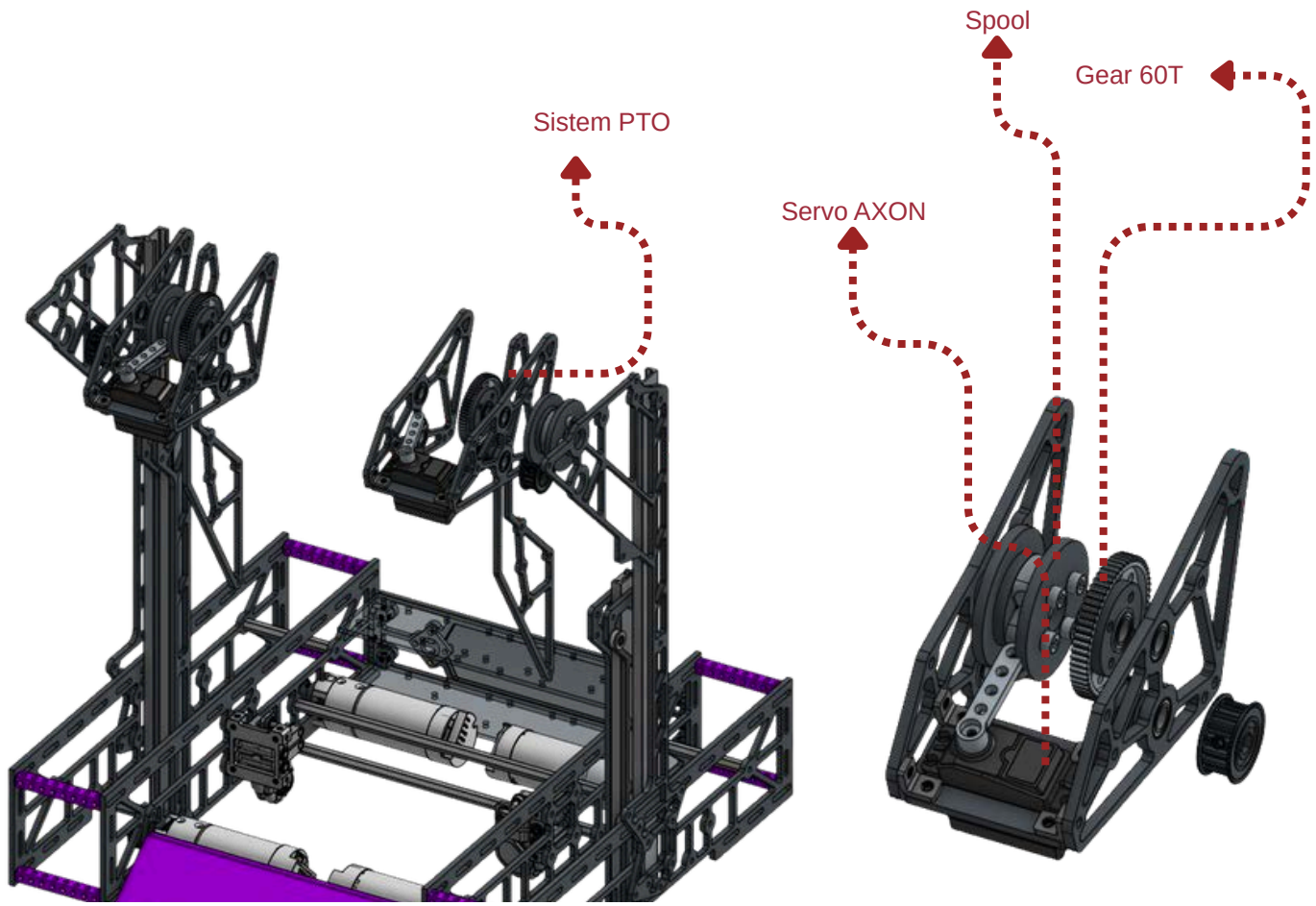
V5 **reprezintă schimbarea arhitecturală finală a mecanismului de parcare.** În loc să mențină un motor dedicat exclusiv pentru o acțiune de ~5.5 secunde per meci, echipa a proiectat un **sistem dual PTO (Power Take-Off)** care preia puterea direct de la motoarele de intake și mid roller în momentul parkingului. Motoarele se opresc, servoul angrenează gear-ul de cuplare, iar de la acel moment cei doi arbori devin surse de cuplu pentru câte un spool independent. *Aceasta elimină un motor complet din arhitectura robotului, un ESC, cablajul aferent și masa asociată* — înlocuite cu un servo și două angrenaje de cuplare. Vizibil în CAD, cele două ansambluri PTO sunt montate simetric pe structura superioară a robotului, fiecare cu propriul lanț cinematic și propriul spool.

2.BRAINSTORMING

- **Un motor dedicat de parcare funcționează efectiv pentru ~0.13 secunde dintr-un meci de 120 de secunde — o utilizare de 0.11% din durata meciului.** Restul timpului este **masă pasivă** pe robot, consumând buget de masă și spațiu constructiv fără niciun beneficiu în fazele de intake, sortare sau tragere. PTO-ul rezolvă această ineficiență fundamental — motoarele de intake și mid roller sunt deja prezente, deja alimentate, deja integrate în sistemul de control. Costul adăugat este exclusiv mecanic: un servo care angrenează fizic un gear de cuplare, conectând axul motorului la lanțul cinematic al spoolului.

- **Secvența de cuplare** — *motoare oprite înainte de angrenare.* Decizia de a opri motoarele înainte de cuplarea PTO-ului a fost deliberată. Angrenarea gear-ului de cuplare din mers — cu motoarele la turație — ar fi generat un șoc mecanic la intrarea în mesh, cu risc de săritură de dinte sau deteriorare a angrenajului de cuplare. Oprind motoarele înainte, gear-ul de cuplare intră în mesh static, fără șoc, după care motoarele sunt repornite direct în regim de parcare. Această secvență este **gestionată software și durează estimat ~100–150ms total — neglijabil față de fereastra de timp disponibilă la finalul meciului.**

- **Dual PTO cu 2 spooluri independente.**
  Fiecare PTO acționează propriul spool separat, ridicând robotul în două puncte independente. Față de un spool comun pe ax partajat, soluția cu două spooluri independente oferă redundanță — dacă un PTO nu se cuplează corect, celălalt poate completa ridicarea parțială. De asemenea, doi arbori independenți elimină solicitările de torsiune pe un ax comun lung, care în spațiul constructiv al robotului ar fi introdus **flexibilitate și variații de sincronizare.**
- **Transmisie 1:1 pe pulleyuri + 24T→60T pe gearuri**
  Pulleyurile 1:1 transferă mișcarea din zona motorului spre gearurile de reducție fără a modifica raportul, permițând flexibilitate geometrică în rutarea transmisiei. ***Treapta 24T→60T realizează reducerea de turație și multiplicarea cuplului*** necesare pentru ridicare.



3.CALCULE

Calcule

3.1 Specificații motor gp BILDA 5202 Series 1150 RPM

$T_{stall} \simeq 2{,}1\ Nm \Rightarrow T_{nominal} \simeq 1{,}05\ Nm$

3.2 Raport transmisie gear-uri

$i_{gear} = \frac{60}{24} = 2{,}5:1$

3.3 Turația la spool

$n_{spool} = \frac{1150}{2{,}5} = 460\ RPM$

3.4 Cuplu total - dual PTO

$T_{total} = 2 \cdot 2{,}625 = 5{,}25\ Nm$

3.5 Cuplul per PTO la spool

$T_{PTO} = 1{,}05 \cdot 2{,}5 = 2{,}625\ Nm$

3.6 Diametrul spool - ului

$F_{spool} = 58{,}85\ N$

$r_{spool} = \frac{T_{PTO}}{F_{spool}} = \frac{2{,}625}{58{,}85} = 0{,}0446\ m$

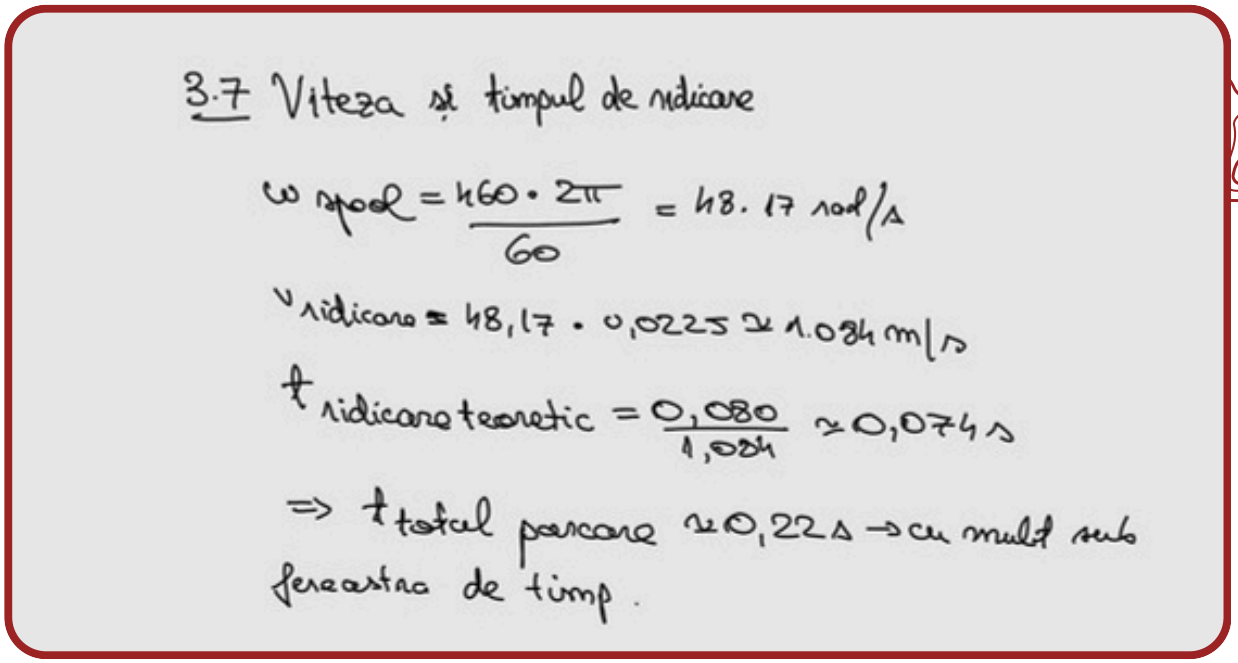
$d_{spool} = 45\ mm$

Verificare marjă:

$T_{necesar} = 58{,}85 \cdot 0{,}0225 = 1.324\ Nm < 2{,}625\ Nm$

Marjă de siguranță: $\frac{(2.625 - 1.324)}{2625} \cdot 100 \simeq 49{,}6\%$

Față de marja de 3,6% $V_2 - V_3$ => de 14x mai mare



4.TESTING

4.1 DATA MATCH

Parametru	Teoretic	Observat
Timp ridicare pur mecanic	~0.074s	~0.13s
Timp total parcare (cu secvență)	~0.22s	~0.25s
Cuplu total	5.25 Nm	confirmat — ridicare completă
Marjă cuplu	~49.6%	confirmată
Stabilitate	foarte bună	confirmată
Sincronizare dual PTO	simultană	confirmată în teste laborator

Diferența dintre timpul teoretic și cel observat este atribuită timpului de secvență software (oprire–cuplare–repornire) și frecărilor din sistemul de ghidare — nu unei limitări de cuplu.

4.2 REPETABILITATE

Testele de laborator au demonstrat ridicare consistentă pe multiple cicluri consecutive. Cei doi spooluri independenți ridicau robotul simultan și simetric, fără moment de răsturnare laterală. Distribuția sarcinii pe două PTO-uri independente **reducea stresul mecanic la jumătate față de un singur punct de acționare** — motoarele de intake și mid roller nu prezentau semne de supraîncălzire după sesiunile de testare, confirmând că **sarcina de parcare era confortabilă în parametrii nominali.**

4.3 FAIL POINTS

- **Netestată competițional** — principala limitare a V5 este *absența validării în condiții de meci real*. Testele de laborator nu reproduc variabilele unui meci — suprafețe neregulate, contact cu roboți adversari, presiune de timp, variații de tensiune în baterie la finalul meciului când parkingul este activat. Toate fail points-urile identificate mai jos sunt teoretice, derivate din analiza mecanică, nu din date experimentale competiționale.

- **Sincronizarea secvenței de cuplare** — dacă cei doi servo-uri nu angrenează simultan, *un PTO preia temporar întreaga sarcină de 117.7N*, operând la **~2× forța nominală per spool**. Deși marja de cuplu de 49.6% oferă o rezervă, scenariul de cuplare asimetrică rămâne un **risc de monitorizat la prima utilizare competițională**.

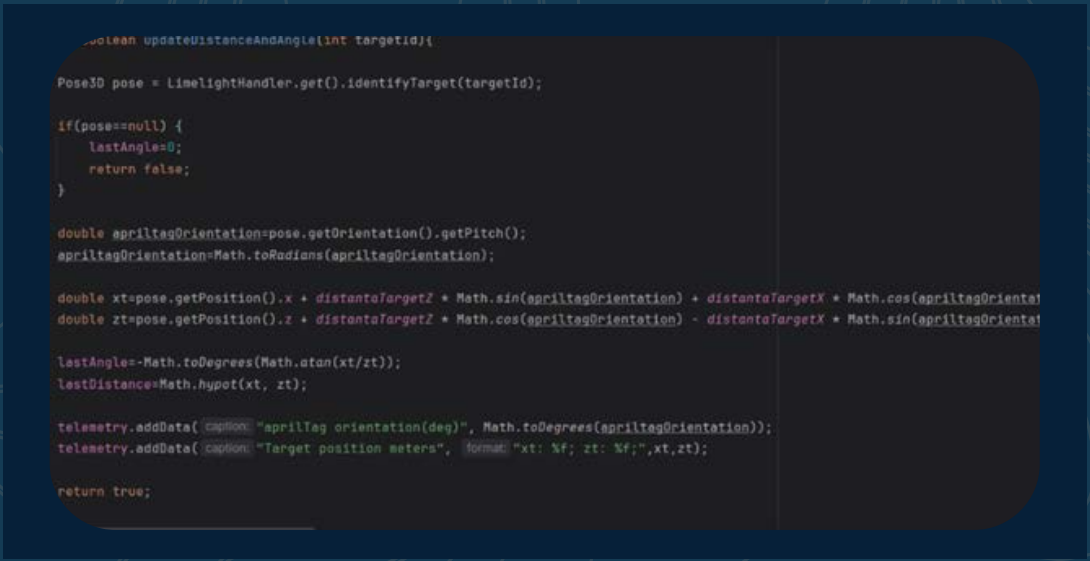
- **Uzura gear-ului de cuplare** — intrarea repetată în mesh din poziție statică introduce șocuri locale pe dinții de contact. Pe durata unui sezon lung, de implementat o rampă software de accelerare graduală post-cuplare pentru a reduce șocul de pornire.

4.4 COMPARATIE V1 - V5

Parametru	V1	V2	V3	V4	V5
Actuator	2× AXON Max	motor dedicat	motor dedicat	motor dedicat	dual PTO
Transmisie	directă	GT2 20T	bevel clampuri	bevel clampuri	pulley 1:1 + gear 24T/60T
Spool	—	1× comun	1× comun	1× comun	2× independente
Cuplare PTO	—	—	—	—	servo → gear, motoare oprite
Marjă cuplu	minimă	~3.6%	~3.6%	~3.6%	~49.6%
Glisiere	—	—	dezaliniate	CNC aliniate	CNC aliniate
Platformă	brațe laterale	spool compact	subdimensionată	+40mm	+40mm
Motor dedicat parcare	da	da	da	da	eliminat
Timp parcare total	~1.2s	~0.15s	~0.12s	~0.12s	~0.25s
Testat competițional	da	da	da	da	nu
Fiabilitate	scăzută	medie	ridicată	ridicată	nevalidată competițional

V5 rămâne cea mai ambițioasă iterație din punct de vedere arhitectural — eliminarea motorului dedicat și integrarea parkingului direct în motoarele funcționale ale robotului reprezintă direcția corectă pentru versiunile viitoare.

CAPITOLUL 5
SOFTWARE & CONTROL ENGINEERING



```
boolean updateDistanceAndAngle(int targetId){

Pose3D pose = LimelightHandler.get().identifyTarget(targetId);

if(pose==null) {
    lastAngle=0;
    return false;
}

double apriltagOrientation=pose.getOrientation().getPitch();
apriltagOrientation=Math.toRadians(apriltagOrientation);

double xt=pose.getPosition().x + distantaTargetZ * Math.sin(apriltagOrientation) + distantaTargetX * Math.cos(apriltagOrienta
double zt=pose.getPosition().z + distantaTargetZ * Math.cos(apriltagOrientation) - distantaTargetX * Math.sin(apriltagOrienta

lastAngle=-Math.toDegrees(Math.atan(xt/zt));
lastDistance=Math.hypot(xt, zt);

telemetry.addData( caption: "aprilTag orientation(deg)", Math.toDegrees(apriltagOrientation));
telemetry.addData( caption: "Target position meters",  format: "xt: %f; zt: %f;",xt,zt);

return true;
```

4.1 ROLUL DEPARTAMENTULUI

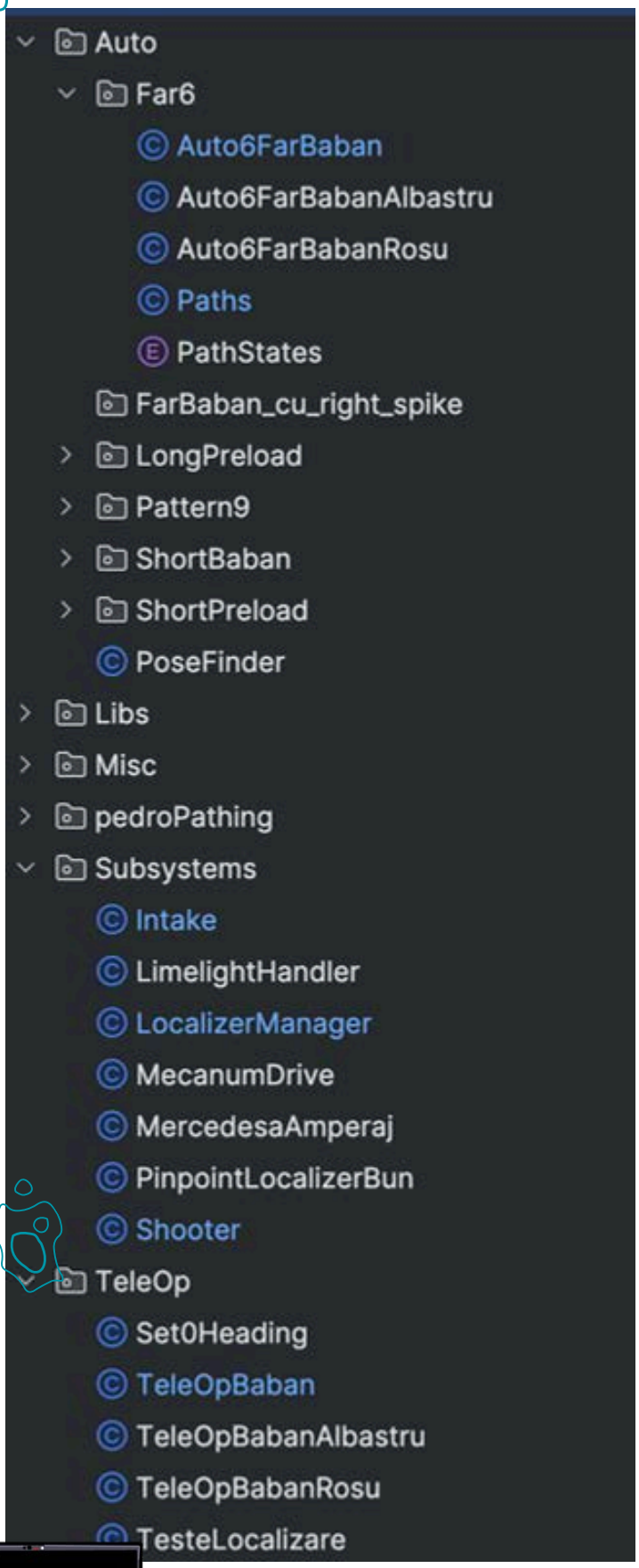
Departamentul software este unul dintre cei 3 piloni ai oricărei echipe FTC. În cadrul echipei noastre, acesta are scopul de a dezvolta, testa și optimiza programele care controlează robotul echipei. Acest departament dezvoltă atât programul de TeleOp, pentru care colaborează strâns cu drive team-ul ca să îi ajute în strategia de joc și în controlul intuitiv al robotului pentru driver, cât și autonomiile. Pentru îndeplinirea acestor scopuri, membrii departamentului software utilizează componentele hardware utilizate de robot, programe precum **Android Studio** și librării precum **PedroPathing**.

Din punct de vedere organizatoric, departamentul software are 4 membrii, fiecare specializându-se într-un diferit aspect necesar pentru funcționarea corespunzatoare a departamentului:

1. **Lead developer**: este cel care coordonează întreaga activitate a departamentului și stabilește întreaga structură a codului. El este cel care decide arhitectura programului, organizează fiecare parte a codului și lucrează strâns cu drive team-ul pentru a dezvolta clasele de TeleOp, spre placul lor.
2. **Specialistul pe autonomii**: este cel care se ocupă de autonomii, de la concepție până la scrierea lor. El se ocupă de strategia inițială din meci, din timpul autonomiilor și trebuie să creeze autonomii cât mai flexibile, pentru a putea maximiza numărul de puncte obținute în timpul autonomiei, în funcție de partenerul de alianță.
3. **“Arhivistul”**: este cel care documentează parcursul departamentului atât pentru caietul tehnic, cât și pentru portofoliu. Astfel, el trebuie să fie familiar cu codul și să poată să descrie în detaliu fiecare aspect al acestuia.
4. **Developer-ul de aplicații**: este membrul departamentului software care colaborează strâns cu departamentul de marketing. Acesta dezvoltă aplicațiile pe care le utilizăm pentru promovarea echipei, precum site-ul și aplicația de afișare a fișierului CAD al robotului la standul echipei.

Pentru simplitatea organizării, atât a codului, cât și a oamenilor, am utilizat sistemul GIT control a versiunii. GIT ne permite să creem branch-uri diferite, fiecare axate pe un lucru diferit, iar apoi să le reunim pe toate într-o variantă finală a codului, gata de competiție. Astfel, putem lucra concomitent la mai multe astpecte ale codului, fără a modifica codul de bază de fiecare dată.

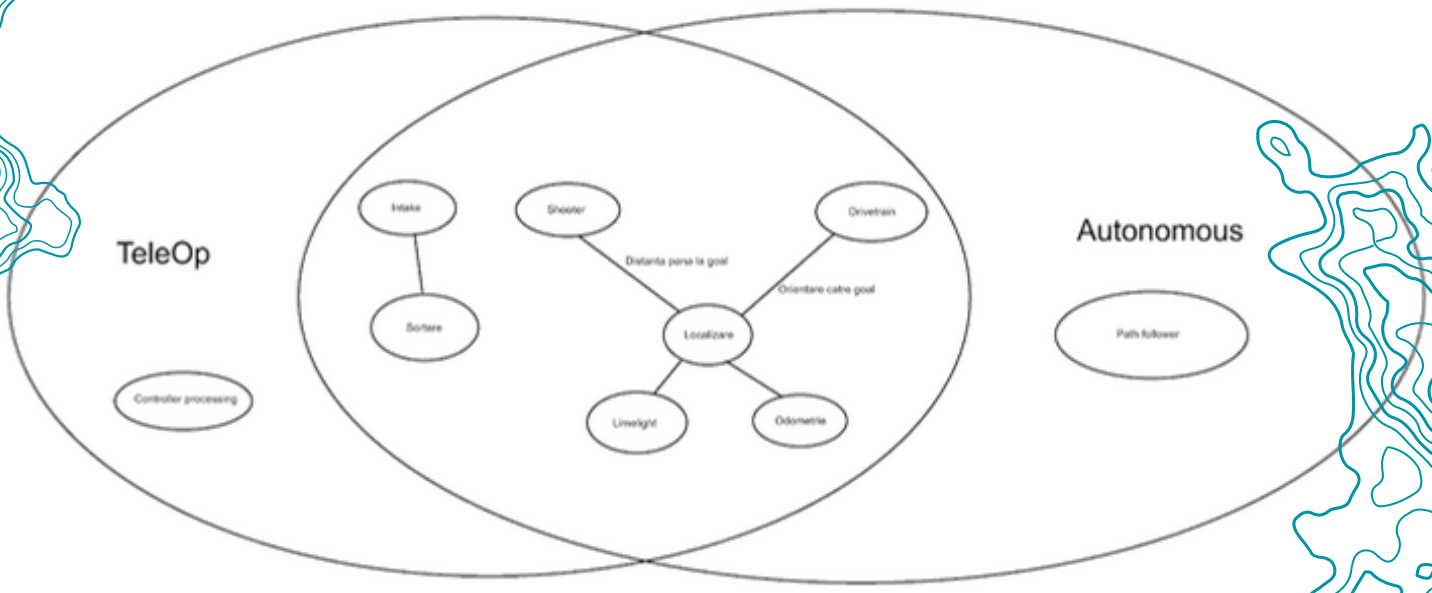
4.2 ARHITECTURA CODULUI



Arhitectura codului a fost gândită pe principiul modularizării, astfel încât fiecare subsistem hardware și logic este abstractizat într-o clasă independentă, definită într-un fișier separat. Această abordare permite o organizare mai clară a codului și facilitează mentenanța și extinderea proiectului.

Principalele sisteme ale robotului sunt drivetrain-ul, shooter-ul și intake-ul. Dintre acestea, drivetrain-ul și shooter-ul depind de sistemul de localizare, deoarece pentru funcționarea lor corectă este necesară cunoașterea poziției robotului pe teren. Sistemul de viziune este utilizat atât în cadrul sistemului de localizare, cât și în autonomii, unde ajută la detectarea elementelor de pe teren.
Sortarea este integrată în sistemul de intake, permițând gestionarea eficientă a elementelor colectate.

În final, toate subsistemele sunt integrate în TeleOp, unde robotul este controlat manual, în timp ce autonomiile nu au încă implementat sistemul de localizare.



DESIGN

Pattern-ul Singleton este foarte des folosit în codul nostru, acesta având numeroase beneficii. Scopul acestuia este ca pe durata desfășurării programului să asigure *existența unei singure instanțe a fiecărui subsistem*. Astfel, pot fi accesate oriunde în cod, fără a fi nevoie să fie transmise drept parametri. Totodată, previne prezența unor stări diferite ale aceluiași subsistem, **eliminând inconsistența programului.**

```
3 usages
private static LimelightHandler instance = null;

6 usages
private Telemetry telemetry;

1 usage
private LimelightHandler(){}

public static LimelightHandler get() {
    if(instance == null)
        instance = new LimelightHandler();
    return instance;
}
```

Singleton LimeLight

```
private static MecanumDrive instance;

1 usage
private MecanumDrive(){}

public static MecanumDrive get(){
    if(instance==null)
        instance = new MecanumDrive();
    return instance;
}
```

Singleton Drive

```
private static LocalizerManager instance;
1 usage
private LocalizerManager(){}
public static LocalizerManager get(){
    if(instance==null)
        instance=new LocalizerManager();
    return instance;
}
```

Singleton Localizare

```java
3 usages
private static Intake instance = null;

1 usage
private Intake() {
}

public static Intake get() {
    if (instance == null)
        instance = new Intake();
    return instance;
}
```

Singleton Intake

```java
private static Shooter instance=null;

10 usages
private Telemetry telemetry;

1 usage
private Shooter() {
}

public static Shooter get() {
    if(instance==null)
        instance = new Shooter();
    return instance;
}
```

Singleton Shooter

Finite state machine-uri

Finite State Machine-ul este un **model de control în care sistemul analizat se află mereu într-o singură stare la un moment de timp**. Acesta poate trece în alte stări prin tranziții bine definite, pe baza unor condiții. Noi folosim acest pattern pentru shooter în autonomie, intake și sortare, lockin drivetrain și în fiecare clasă principală de autonomie (PathStates).

```java
private static double defaultIntakePower=1,defaultMidRollerPower=1;
2 usages
private IntakeStates currentState = IntakeStates.NONE;
public void setState(IntakeStates state, double powerMultiplier){
    switch (state){
        case TO_HOPPER:
            owdoorDown();
            setIntakePower(defaultIntakePower*powerMultiplier);
            setMidrollerPower(-defaultMidRollerPower*powerMultiplier);
            break;
        case TO_SHOOTER:
            owdoorMid();
            setIntakePower(defaultIntakePower*powerMultiplier);
            setMidrollerPower(defaultMidRollerPower*powerMultiplier);
            break;
        case HOPPER_TO_SHOOTER:
            owdoorUp();
            setIntakePower(-defaultIntakePower*powerMultiplier);
            setMidrollerPower(defaultMidRollerPower*powerMultiplier);
            break;
        case NONE:
        default:
            owdoorMid();
            setIntakePower(0);
            setMidrollerPower(0);
            break;
    }
    currentState=state;
}
3 usages
public IntakeStates getState() { return currentState; }
```

Finite State Machine

```
public class AutoFarBaban {
    public void autonomousPathUpdate() {
                }
                break;
            case ToPickupDown:
                if(!follower.isBusy())
                {
                    pathState= PathStates.PickupDown;
                    follower.followPath(paths.pickupDown,  holdEnd: true);
                }
                break;
            case PickupDown:
                if(!follower.isBusy())
                {
                    pathState= PathStates.ScoreDown;
                    follower.followPath(paths.scoreDown, holdEnd: true);
                }
                break;
            case ScoreDown:
            case ScoreWall:
                if(!follower.isBusy()) {
                    if (Shooter.get().getAutoShootState() == Shooter.AutoShootStates.NONE)
                        Shooter.get().startAutoShoot(targetId);
                    if (Shooter.get().getAutoShootState() == Shooter.AutoShootStates.ENDED) {

                            cntPickupWall++;
                            if(cntPickupWall<=nrPickupWall){
                                pathState = PathStates.PickupWall;
                                follower.followPath(paths.pickupWall);
                            }
                            else {
                                pathState = PathStates.Leave;
                                follower.followPath(paths.leave);
                            }
                            Shooter.get().resetAutoShootState();
                        }
                    }
                break;
            case PickupWall:
                if(!follower.isBusy()){
                    pathState = PathStates.RetractWall;
                    follower.followPath(paths.retractWall,  maxPower: 0.8,  holdEnd: false);
                }
                break;
            case RetractWall:
                if(!follower.isBusy()){
                    pathState = PathStates.BackInWall;
                    follower.followPath(paths.backInWall);
                }
                break;
            case BackInWall:
                if(!follower.isBusy()){
                    pathState = PathStates.ScoreWall;
                    follower.followPath(paths.scoreWall,  holdEnd: true);
                }
                break;
```

Finite State Machine Autonomie

```
public void updateAutoShootWithSpeed(){
    if(autoShootSorted)
    {
        updateAutoShootWithSpeedSorted();
        return;
    }
    switch (autoShootState){
        case INIT:
            setSpeed(autoShootSpeed, updatePutza: true);
            canShoot=false;
            autoShootState=AutoShootStates.GET_TO_SHOOT_SPEED;
            break;
        case GET_TO_SHOOT_SPEED:
            if(!canShoot){
                maintainSpeed();
            }
            if(canShoot){

                stopperAllow();
                Intake.get().setState(Intake.IntakeStates.TO_SHOOTER, intakePowerShooter);
                autoShootState=AutoShootStates.OPEN_FIRE;
                shooterTimer = new Timer();
            }
            break;
        case OPEN_FIRE:
            if(shooterTimer.getElapsedTimeSeconds()> autoShootTime)
            {
                endAutoShoot();
            }
            maintainSpeed();
            break;
    }
```

Finite State Machine Shooter

4.3 DRIVETRAIN

Drivetrain-ul, specializat în MecanumDrive, *se ocupă cu tot ceea ce implică mișcarea robotului pe teren în timpul TeleOP-ului*. Clasa MecanumDrive are drept feature-uri:

- **Inițializare**
- **Funcție Update (Run)**
- **Activare/Dezactivare lockin**
- **Speed multiplier**

Inițializare

- inițializează drivetrain-ul robotului. Metoda init() mapează cele **patru motoare** (*frontLeft, backLeft, frontRight, backRight*) din HardwareMap, conectând codul cu componentele fizice ale robotului. Motoarele din partea dreaptă sunt setate pe direcția REVERSE pentru a asigura mișcarea corectă într-o configurație mecanum. În final, toate motoarele sunt configurate cu ZeroPowerBehavior.BRAKE, astfel încât robotul să se oprească imediat când puterea este zero. De asemenea, codul previne inițializarea multiplă a drivetrain-ului folosind variabila initialized.

```
private boolean initialized=false;
public void init(HardwareMap hardwareMap, Telemetry telemetry){
    if(initialized) return;
    initialized=true;
    this.telemetry=telemetry;
    /// Drive
    frontLeftMotor = hardwareMap.dcMotor.get("frontLeft");
    backLeftMotor = hardwareMap.dcMotor.get("backLeft");
    frontRightMotor = hardwareMap.dcMotor.get("frontRight");
    backRightMotor = hardwareMap.dcMotor.get("backRight");
    //mecanum reverse
    frontRightMotor.setDirection(DcMotorSimple.Direction.REVERSE);
    backRightMotor.setDirection(DcMotorSimple.Direction.REVERSE);
    frontLeftMotor.setZeroPowerBehavior(DcMotor.ZeroPowerBehavior.BRAKE);
    frontRightMotor.setZeroPowerBehavior(DcMotor.ZeroPowerBehavior.BRAKE);
    backLeftMotor.setZeroPowerBehavior(DcMotor.ZeroPowerBehavior.BRAKE);
    backRightMotor.setZeroPowerBehavior(DcMotor.ZeroPowerBehavior.BRAKE);
}
```

- ***Funcție Update (Run)***

Această metodă primește ca parametri input-urile pe cele 3 grade de libertate ale șasiului de la controller și apelează funcția de drive corespunzătoare stării în care se află șasiul (dacă se menține heading-ul către goal sau nu).

Dacă valoarea **rx** este mai mare decât un prag minim, sistemul dezactivează automat **lockIn**, deoarece driverul comandă o rotație manuală. Dacă robotul nu este în modul lockedIn, acesta *se deplasează normal folosind valorile de translație și rotație*. Dacă modul **lockedIn este activ, codul folosește un PID de rotație pentru a corecta eroarea de unghi** și a menține robotul orientat spre direcția țintă în timp ce se deplasează. Astfel, sistemul permite atât control liber, cât și menținerea automată a orientării.

```
public void run(double x, double y, double rx){
    if(Math.abs(rx)>=0.1){
        cancelLockIn();
    }

    if(!lockedIn)
        drive(y, x, rx);
    else
    {
        rotationPID.updateError(LocalizerManager.get().getHeadingToGoalError(AngleUnit.DEGREES));
        driveLockedIn(y, x);
    }
}
```

- ***Drive***

Rolele de pe roțile mecanum sunt dispuse la aproximativ 45° față de planul roții. Când o roată generează o forță diagonală la 45°, acea forță poate fi descompusă în două componente egale:

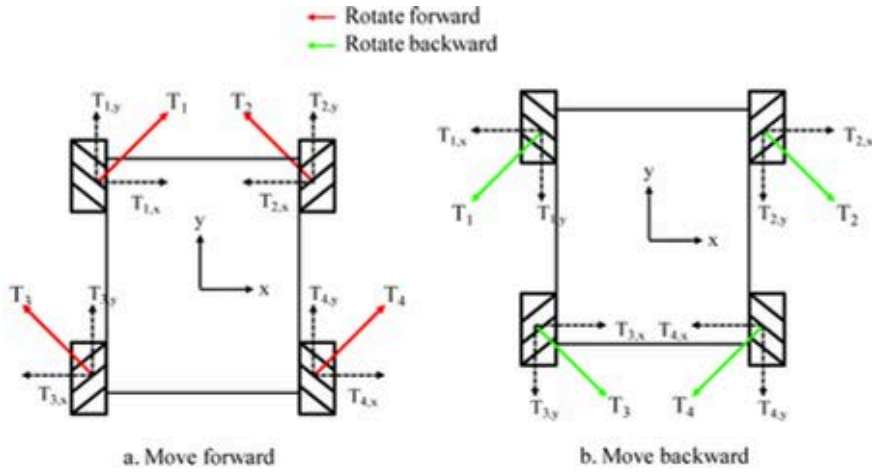
- **una longitudinală**
- **una laterală**

Deci cele două componente au aceeași mărime.
Asta înseamnă că fiecare roată mecanum este capabilă să contribuie simultan și la mișcarea înainte, și la mișcarea laterală.
Acesta este marele avantaj al mecanumului: prin combinarea celor patru roți, poți controla independent translația pe două axe și rotația.

```
private void drive(double y, double x, double rx){
    x*=speedMultiplier;
    y*=speedMultiplier;
    rx*=speedMultiplier;

    double frontLeftPower = (y + x + rx);
    double backLeftPower = (y - x + rx);
    double frontRightPower = (y - x - rx);
    double backRightPower = (y + x - rx);

    double maxPower = 1.0;
    maxPower = Math.max(maxPower, Math.abs(frontLeftPower));
    maxPower = Math.max(maxPower, Math.abs(frontRightPower));
    maxPower = Math.max(maxPower, Math.abs(backLeftPower));
    maxPower = Math.max(maxPower, Math.abs(backRightPower));

    frontLeftMotor.setPower(frontLeftPower/maxPower);
    backLeftMotor.setPower(backLeftPower/maxPower);
    frontRightMotor.setPower(frontRightPower/maxPower);
    backRightMotor.setPower(backRightPower/maxPower);
}
```

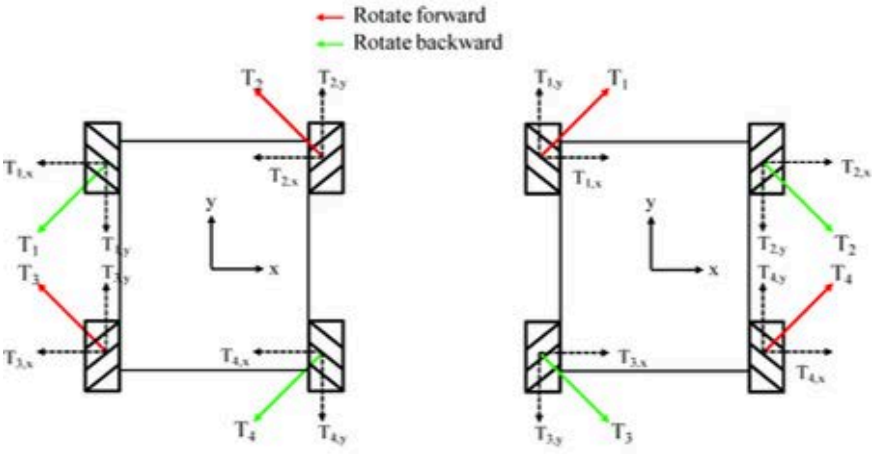
În cazul deplasării înainte, componentele longitudinale ale tuturor celor patru roți se adună, în timp ce componentele laterale se anulează. În cazul deplasării laterale, doar două roți contribuie eficient la generarea forței de mișcare, iar celelalte două se opun.

Din acest motiv, forța totală disponibilă pentru deplasarea înainte este aproximativ dublă față de cea pentru deplasarea laterală.



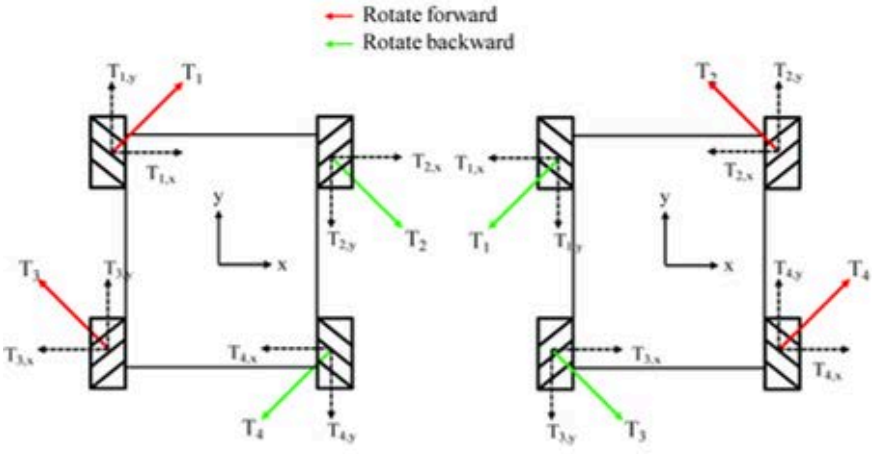
a. Move forward

b. Move backward



c. Move sideways - left

d. Move sideways - right



e. Rotate clockwise

f. Rotate anticlockwise

De asemenea, atunci când robotul mecanum se deplasează **simultan înainte și lateral**, vectorul de mișcare este rezultatul compunerii celor două componente.

Magnitudinea acestui vector devine mai mare decât viteza maximă permisă motoarelor, astfel încât sistemul de control aplică o normalizare a puterilor motoarelor. În urma acestei scalări, componentele individuale ale vitezei scad. Din acest motiv, deplasarea diagonală este percepută ca fiind mai lentă decât deplasarea exclusivă pe direcția înainte.

Aceasta este o **funcție de drive normală**, care are încorporat un speedMultiplier drept sensibilitate a inputului de la controller. În acest mod, driverul poate executa mișcări la viteză mai mică dacă este necesară o precizie sporită.

4.4 SISTEMUL DE VIZIUNE

- ***LimelightHandler***

Viziunea robotului este datorată camerei integrată cu AI Limelight 3A. Aceasta este implementată în cod într-o clasă numită LimelightHandler. Pentru a schimba pipeline-urile între ele, în funcție de ce aprilTag trebuie să recepteze sunt utilizate metodele **setShooterPipeline()** și **setObeliskPipeLine().**

```
public void setShooterPipeline(){

    if(currentPipeLine!=1)
        ll.pipelineSwitch( index: 1);
    currentPipeLine=1;
}
no usages
public void setPipeline(int index){
    if(currentPipeLine!=index)
        ll.pipelineSwitch(index);
    currentPipeLine=index;
}
3 usages
public void setObeliskPipeline(){
    if(currentPipeLine!=0)
        ll.pipelineSwitch( index: 0);
    currentPipeLine=0;
}
```

În cazul sortării, funcția updateObeliskID() este actualizată în loop() pentru a putea identifica fiducial-ul corespunzător motivului din meciul respectiv.

Funcția getGreenPosition() prelucrează id-ul obținut de la obelisk și determină astfel poziția bilei verzi din pattern.

```java
public boolean updateObeliskID() {

    setObeliskPipeline();

    if (ll == null) return false;

    LLResult result = ll.getLatestResult();
    if (result == null || !result.isValid()) return false;

    List<LLResultTypes.FiducialResult> fiducials = result.getFiducialResults();
    if (fiducials == null || fiducials.isEmpty()) return false;


    for (LLResultTypes.FiducialResult f : fiducials) {

        int id = f.getFiducialId();
        if (id == 21 || id== 22 || id==23) {
            obeliskID = id;
        }
    }
```

```java
21 usages
public int getGreenPosition() {
    if (obeliskID == 0) {
        return 0;
    }
    return obeliskID-20;
}
```

Inițial sistemul de localizare era cuprins în LimelightHandler, acesta reprezentând unul dintre task-urile limelight-ului. Prin intermediul metodei identifyTarget **este returnată poziția depotului în spațiul robotului.**

```java
public Pose3D identifyTarget(int targetId){

    if(ll == null) return null;

    LLResult result = ll.getLatestResult();
    if(result == null || !result.isValid()) return null;

    LLResultTypes.FiducialResult depotTag = findFiducialById(result, targetId);

    if(depotTag == null) return null;

    telemetry.addData( caption: "apriltag pose", format: "x %f y %f z %f, pitch %f, yaw %f, ro

    return depotTag.getTargetPoseRobotSpace();
}
```

UpdateDistanceAndAngle era gândită să determine ultimul unghi și ultima distanță a unui punct situat la **distanțaTargetZ** în spatele **aprilTagului** față de poziția robotului.

```
public boolean updateDistanceAndAngle(int targetId){
    setShooterPipeline();
    Pose3D pose = LimelightHandler.get().identifyTarget(targetId);
    if(pose==null) {
        lastAngle=0;
        return false;
    }
    double aprilTagOrientation = Math.toRadians(pose.getOrientation().getPitch());
    Vector aprilTagPos=new Vector();
    aprilTagPos.setOrthogonalComponents(pose.getPosition().x, pose.getPosition().z);
    telemetry.addData( caption: "april tag pos",  format: "x: %.4f z: %.4f heading: %.4f, distance: %.4f", aprilTagPos.getXComponent()
    Vector targetPoint;
    if(aprilTagPos.getMagnitude() > thresholdLongShot)
        targetPoint = targetVectorLong.copy();
    else targetPoint = targetVector.copy();
    telemetry.addData( caption: "long shot",  value: aprilTagPos.getMagnitude() > thresholdLongShot);


    targetPoint.rotateVector( theta2: Math.PI+aprilTagOrientation);
    targetPoint = targetPoint.plus(aprilTagPos);

    telemetry.addData( caption: "targetPoint",  format: "x: %.4f z: %.4f", targetPoint.getXComponent(), targetPoint.getYComponent());
    lastDistance = targetPoint.getMagnitude();
    lastAngle = Math.toDegrees(MathFunctions.normalizeAngle( angleRadians: targetPoint.getTheta()-Math.PI/2+Math.PI));
    if(lastAngle>180)
        lastAngle-=360;
    return true;
```

```
targetVector = new Vector();
targetVector.setOrthogonalComponents(distantaTargetX, distantaTargetZ);
targetVectorLong = new Vector();
targetVectorLong.setOrthogonalComponents(distantaTargetLongX, distantaTargetLongZ);
```

Versiunea curentă a LimelightHandler-ului se bazează pe o metodă de actualizare a poziției robotului în funcție de heading. Dacă utilizarea Megatag2 este activată, limelight-ul preia orientarea transmisă prin parametri.

```
public void setUseMT2(boolean active) { useMT2=active; }
2 usages
public void updateRobotPose(double yawDegrees){

    if(ll == null){
        robotPose=null;
        return;
    }
    if(useMT2)
    {
        ll.updateRobotOrientation(yawDegrees);
    }


    LLResult result = ll.getLatestResult();
    if(result == null || !result.isValid()){
        robotPose=null;
        return;
    }
    if(!useMT2)
        robotPose = result.getBotpose();
    else robotPose = result.getBotpose_MT2();
}
5 usages
public Pose3D getRobotPose() { return robotPose; }
```

În plus, am decis să realizăm calibrarea ChArUco pentru o mai bună precizie a camerei.

Calibrarea ChArUco a camerei Limelight este procesul prin care se determină parametrii optici reali ai camerei și coeficienții de distorsiune ai lentilei folosind o placă specială cu markere ArUco. Această procedură permite corectarea imaginii și îmbunătățește semnificativ precizia estimărilor 3D și a poziției țintelor, fiind esențială pentru localizare în robotică.

UpdateDistanceAndAngle era gândită să determine ultimul unghi și ultima distanță a unui punct situat la **distanțaTargetZ** în spatele **aprilTagului** față de poziția robotului.

```
public boolean updateDistanceAndAngle(int targetId){
    setShooterPipeline();
    Pose3D pose = LimelightHandler.get().identifyTarget(targetId);
    if(pose==null) {
        lastAngle=0;
        return false;
    }
    double aprilTagOrientation = Math.toRadians(pose.getOrientation().getPitch());
    Vector aprilTagPos=new Vector();
    aprilTagPos.setOrthogonalComponents(pose.getPosition().x, pose.getPosition().z);
    telemetry.addData( caption: "april tag pos",  format: "x: %.4f z: %.4f heading: %.4f, distance: %.4f", aprilTagPos.getXComponent()
    Vector targetPoint;
    if(aprilTagPos.getMagnitude() > thresholdLongShot)
        targetPoint = targetVectorLong.copy();
    else targetPoint = targetVector.copy();
    telemetry.addData( caption: "long shot",  value: aprilTagPos.getMagnitude() > thresholdLongShot);


    targetPoint.rotateVector( theta2: Math.PI+aprilTagOrientation);
    targetPoint = targetPoint.plus(aprilTagPos);

    telemetry.addData( caption: "targetPoint",  format: "x: %.4f z: %.4f", targetPoint.getXComponent(), targetPoint.getYComponent());
    lastDistance = targetPoint.getMagnitude();
    lastAngle = Math.toDegrees(MathFunctions.normalizeAngle( angleRadians: targetPoint.getTheta()-Math.PI/2+Math.PI));
    if(lastAngle>180)
        lastAngle-=360;
    return true;
```

```
targetVector = new Vector();
targetVector.setOrthogonalComponents(distantaTargetX, distantaTargetZ);
targetVectorLong = new Vector();
targetVectorLong.setOrthogonalComponents(distantaTargetLongX, distantaTargetLongZ);
```

Versiunea curentă a LimelightHandler-ului se bazează pe o metodă de actualizare a poziției robotului în funcție de heading. Dacă utilizarea Megatag2 este activată, limelight-ul preia orientarea transmisă prin parametri.

```
public void setUseMT2(boolean active) { useMT2=active; }
2 usages
public void updateRobotPose(double yawDegrees){

    if(ll == null){
        robotPose=null;
        return;
    }
    if(useMT2)
    {
        ll.updateRobotOrientation(yawDegrees);
    }


    LLResult result = ll.getLatestResult();
    if(result == null || !result.isValid()){
        robotPose=null;
        return;
    }
    if(!useMT2)
        robotPose = result.getBotpose();
    else robotPose = result.getBotpose_MT2();
}
5 usages
public Pose3D getRobotPose() { return robotPose; }
```

În plus, am decis să realizăm calibrarea ChArUco pentru o mai bună precizie a camerei.

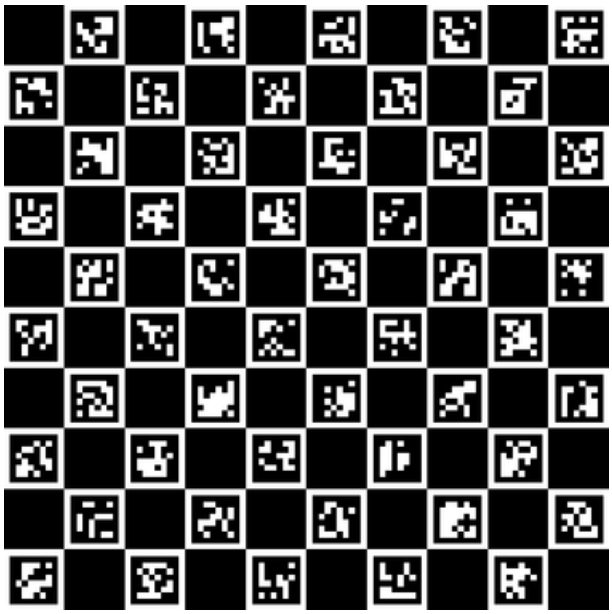
Calibrarea ChArUco a camerei Limelight este procesul prin care se determină parametrii optici reali ai camerei și coeficienții de distorsiune ai lentilei folosind o placă specială cu markere ArUco. Această procedură permite corectarea imaginii și îmbunătățește semnificativ precizia estimărilor 3D și a poziției țintelor, fiind esențială pentru localizare în robotică.

Placa ChArUco este un pattern hibrid care combină:

- **un checkerboard** (tablă de șah)
- **markere ArUco unice în interiorul pătratelor**

Această combinație permite detectarea precisă a colțurilor și identificarea poziției fiecărui pătrat din imagine.

În timpul calibrării, camera face mai multe fotografii ale acestei plăci din diferite unghiuri și distanțe. Sistemul **compară poziția reală a colțurilor** cu poziția lor în imagine și calculează parametrii camerei.



4.5 LOCALIZARE

Sistemul de localizare stă la baza bunei funcționări a shooter-ului. *Funcțiile acestuia sunt determinarea orientării necesare și a distanței față de goal a robotului*. În decursul sezonului, acesta a suferit o gamă largă de schimbări.

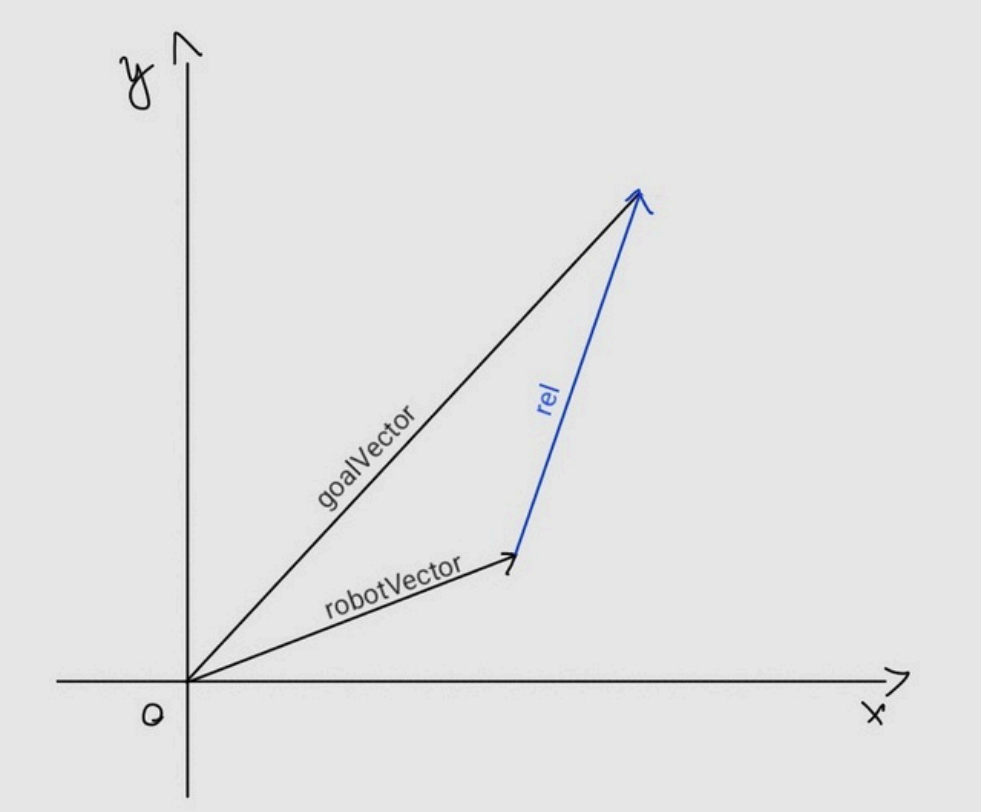
Versiunea finală odometrie + limelight

Sistemul curent de localizare **se bazează atât pe limelight cât și pe odometrie**. Am ales acest sistem dublu întrucât cele două modalități de localizare au caracteristici complementare. Astfel, din punct de vedere al preciziei, *limelight-ul păstrează o acuratețe constantă pe toată durata meciului, în timp ce odometria este predispusă la pierderi.*

Un exemplu des întâlnit în timpul meciurilor este contactul dintre roboți care destabilizează IMU-ul pinpoint-ului. De asemenea, limelight-ul nu necesită resetare manuală, spre deosebire de odometrie. Cea din urmă este dependentă de inițializarea cu o poziție de pornire care încurcă flow-ul driver-ului.

Limelight	Odometrie
Precizie constantă	Are pierderi pe durata meciului
Nu necesită resetare	Vulnerabilă la contactul cu alți roboți
Are refresh rate scăzut	Necesită cunoașterea unei poziții în prealabil
Vulnerabilă la motion-blur	Vibrațiile shooter-ului generează drift în IMU-ul pinpoint-ului
Poate fi blocată de alți roboți	

Astfel, principiul pe care se bazează sistemul este **actualizarea în mod regulat a odometriei cu coordonatele obținute de la limelight**. În plus, odată ce limelight-ul actualizează pentru prima dată odometria, se poate opta pentru folosirea unui anumit sistem drept sistem principal în defavoarea celuilalt.



Cu ajutorul calculului vectorial se determină **vectorul „rel”**, a cărui magnitudine este distanța căutată, iar unghiul acestuia reprezintă orientarea necesară a robotului. Eroarea de orientare este transmisă mai apoi drivetrain-ului pentru a o corecta și a îndeplini funcția de aliniere automată.

Pentru ca artefactul să nu sară din depot, terenul este împărțit în două regiuni, iar în funcție din ce zonă trage robotul, este ales goal pose-ul pe unul dintre pereții depotului. **Astfel, precizia de tragere este îmbunătățită.**

Inițial, sistemul de localizare era conceput să folosească **Megatag2**, iar yaw-ul robotului să fie dat de un **PoseTracker care prelucrează IMU-ul Pinpoint-urilor GoBilda**. În cazul în care odometria era activată, *poziția rezultată avea drept coordonate interpolarea coordonatelor date de limelight cu cele ale odometriei.*

```
lic void update() {

odometryTracker.update();

odometryPose = PoseConverter.poseToPose2D(odometryTracker.getPose(), FTCCoordinates.INSTANCE);

LimelightHandler.get().updateRobotPose(odometryPose.getHeading(AngleUnit.DEGREES));

llPose = new Pose2D(DistanceUnit.METER, LimelightHandler.get().getRobotPose().getPosition().x, LimelightHandler.get().getRobotPose().getPosition().y,
        AngleUnit.DEGREES, LimelightHandler.get().getRobotPose().getOrientation().getYaw(AngleUnit.DEGREES));

if(odometryActive)
    estimatedPose = new Pose2D(DistanceUnit.METER,  x: (llPose.getX(DistanceUnit.METER)+odometryPose.getX(DistanceUnit.METER))/2,
            y: (llPose.getY(DistanceUnit.METER)+odometryPose.getY(DistanceUnit.METER))/2, AngleUnit.DEGREES,
            heading: (llPose.getHeading(AngleUnit.DEGREES)+odometryPose.getHeading(AngleUnit.DEGREES))/2);
else estimatedPose=llPose;
```

În urma multor teste am identificat probleme legate de trecerea din cele două sisteme de coordonate, FTCCoordinates și Pedropathing.

```
odometryPose = PoseConverter.poseToPose2D(odometryTracker.getPose(), FTCCoordinates.INSTANCE);
```

- 1. **Următoarea versiune** a acestui sistem era bazată tot pe localizare cu **Megatag2**, headingul poziției fiind dat de odometrie. Acest sistem era **conceput mult mai simplu, poziția estimată fiind egală cu cea dată de limelight**. Pentru a putea utiliza orientarea dată de Pinpoint-uri, odometria necesita resetare. Astfel, **este concepută funcția de setHeading() pentru a putea fi implementată în TeleOp.**

```
public void setHeading(double heading){
    odometryTracker.setPose(new Pose( x: 0,  y: 0, heading));
    LimelightHandler.get().setUseMT2(true);
}
```

- **2. Versiunea premergătoare celei finale** a constat în **adăugarea unei funcții de transmitere a poziției receptate de limelight către odometrie**. Astfel, după resetarea odometriei este apelată această funcție pentru ca odometria să fie actualizată cu valorile preluate de limelight.

```
public void resetOdometryPose(){
    if(LimelightHandler.get().getRobotPose()!=null) {
        Pose2D llPose = new Pose2D(DistanceUnit.METER, LimelightHandler.get().getRobotPose().getPosition().x, LimelightHandler.get().getRobotPose().getPosition().y,
                AngleUnit.DEGREES, LimelightHandler.get().getRobotPose().getOrientation().getYaw(AngleUnit.DEGREES));


        odometry.setPose(llPose);
        odometryActive=true;
    }
}
```

În ceea ce privește calculul efectiv al orientării și al distanței față de goal, se folosesc următoarele funcții care necesită punctul din care se aruncă artefactele:

```
public double getDistanceFromGoal(DistanceUnit unit) {
    return unit.fromMeters(Math.hypot(goalPose.getX(DistanceUnit.METER)-estimatedPose.getX(DistanceUnit.METER),
            goalPose.getY(DistanceUnit.METER)-estimatedPose.getY(DistanceUnit.METER)));
}
```

Mai jos este funcția de update care este apelată în loop. Cât timp este prezent un aprilTag în câmpul vizual al limelight-ului, poziția este preluată în llPose, urmând în funcție de caz să fie actualizată odometria.

```
private Pose2D estimatedPose;
public void update() {

    odometry.update();

    Pose2D odometryPose = odometry.getPose(),llPose=null;
    LimelightHandler.get().updateRobotPose(odometryPose.getHeading(AngleUnit.DEGREES));

    if(LimelightHandler.get().getRobotPose()!=null) {
        llPose = new Pose2D(DistanceUnit.METER,
                LimelightHandler.get().getRobotPose().getPosition().x,
                LimelightHandler.get().getRobotPose().getPosition().y,
                AngleUnit.DEGREES,
                LimelightHandler.get().getRobotPose().getOrientation().getYaw(AngleUnit.DEGREES));

        if(!usePrimaryOdometry || !odometryActive){
            odometry.setPose(llPose);
            odometryPose=odometry.getPose();
            odometryActive=true;
        }

        estimatedPose=llPose;
    }
    if(odometryActive && (usePrimaryOdometry || llPose==null)){
        estimatedPose=odometryPose;
    }
}
```

În ceea ce privește calculul efectiv al orientării și al distanței față de goal, se folosesc următoarele funcții care necesită punctul din care se aruncă artefactele:

```
public double getHeadingToGoalError(AngleUnit unit){

    return unit.fromRadians(MathFunctions.normalizeAngle( angleRadians: estimatedPose.getHeading(AngleUnit.RADIANS)-getHeadingToGoal(AngleUnit.RADIANS)));
}
```

```
public double getDistanceFromGoal(DistanceUnit unit) {
    return unit.fromMeters(Math.hypot(goalPose.getX(DistanceUnit.METER)-estimatedPose.getX(DistanceUnit.METER),
            goalPose.getY(DistanceUnit.METER)-estimatedPose.getY(DistanceUnit.METER)));
}
```

```
public double getHeadingToGoal(AngleUnit unit){

    Vector goalVector = new Vector();
    goalVector.setOrthogonalComponents(goalPose.getX(DistanceUnit.METER), goalPose.getY(DistanceUnit.METER));

    Vector robotVector = new Vector();
    robotVector.setOrthogonalComponents(estimatedPose.getX(DistanceUnit.METER), estimatedPose.getY(DistanceUnit.METER));

    Vector rel = robotVector.minus(goalVector);

    return unit.fromRadians(rel.getTheta());
}
```

4.6 SHOOTER

Sistemul de aruncare a bilelor este conceput, astfel încât *viteza optimă de tragere să fie calculată pe baza distanței dintre poziția robotului și cea a goal-ului, fiind evitată utilizarea unor valori statice.*

$$v_0^2 = \frac{gd^2(1+\tan^2\alpha)}{2(d\tan\alpha+\Delta h)}$$

```
1 usage
public static double targetHeight=1.15,targetHeightLong=1.15;
1 usage
public static double randamentViteza=0.475;
1 usage
public static double randamentVitezaLong=0.45;
2 usages
public static double g=9.806,razaShooter=0.048,h0=0.39265;

6 usages
public void shoot(double distance, boolean updatePutza) {
    final double tanAlpha=Math.tan(Math.toRadians(41.233153));

    if(distance > LocalizerManager.thresholdLongShot){
        double v0=Math.sqrt((g*distance*distance*(1+tanAlpha*tanAlpha))/(2*(distance*tanAlpha-targetHeightLong+h0)));
        double rpm=((v0/razaShooter)/(2*Math.PI)*60)/ randamentVitezaLong;

        double encoderSpeed=rpm/60*28;
        setSpeed(encoderSpeed, updatePutza);
    }
    else {
        double v0=Math.sqrt((g*distance*distance*(1+tanAlpha*tanAlpha))/(2*(distance*tanAlpha-targetHeight+h0)));
        double rpm=((v0/razaShooter)/(2*Math.PI)*60)/randamentViteza;

        double encoderSpeed=rpm/60*28;
        setSpeed(encoderSpeed, updatePutza);
    }

}
```

```
11 usages
public void maintainSpeed() { maintainSpeed( updatePutza: true); }
6 usages
public void maintainSpeed(boolean updatePutza){
    speedPid.updatePosition(leftMotor.getVelocity());
    updateFilteredPIDFCoefficients();
    speedPid.updateFeedForwardInput(1);
    if(Math.abs(speedPid.getError())>=speedThresholdLimbutza){
        timerLimbutza.resetTimer();
    }
    if(timerLimbutza.getElapsedTimeSeconds()>=timeThresholdLimbutza && speedPid.getTargetPosition()!=restSpeed && updatePutza)
    {
        if(!canShoot)
            stopperAllow();
        canShoot=true;
    }

    double power = speedPid.run();
    rightMotor.setPower(power);
    leftMotor.setPower(power);
    speedPid.addTelemetry(telemetry);

}
```

```
2 usages
private static ClampedFilteredPIDFCoefficients speedPidCoeff = new ClampedFilteredPIDFCoefficients( p 1, i 0, d 0, f 0.2, t 0, lowerLimit 0, upperLimit 1);///ClampedFilteredPIDFCoeff
private static ClampedFilteredPIDFCoefficients speedPidSecondaryCoeff = new ClampedFilteredPIDFCoefficients( p 0.005, i 0.007, d 0.00001, f 0.7, t 0, lowerLimit 0, upperLimit 1);
2 usages
private static double speedPidCoeffTreshold=80;
14 usages
private DualClampedFilteredPIDFController speedPid = new DualClampedFilteredPIDFController(speedPidCoeff, speedPidSecondaryCoeff, speedPidCoeffTreshold);
```

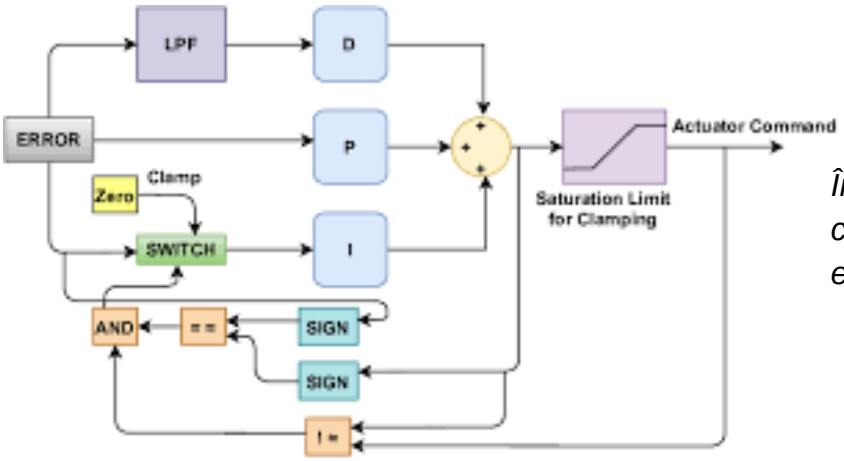
```
public class DualClampedFilteredPIDFController implements Controller {
    public double P() {
        if(Math.abs(error)<=coefficientThreshold){
            return secondaryCoefficients.P;
        }
        else return primaryCoefficients.P;
    }
    3 usages
    public double I() {
        if(Math.abs(error)<=coefficientThreshold){
            return secondaryCoefficients.I;
        }
        else return primaryCoefficients.I;
    }
    3 usages
    public double D() {
        if(Math.abs(error)<=coefficientThreshold){
            return secondaryCoefficients.D;
        }
        else return primaryCoefficients.D;
    }
    3 usages
    public double T() {
        if(Math.abs(error)<=coefficientThreshold){
            return secondaryCoefficients.T;
        }
        else return primaryCoefficients.T;
    }
    3 usages
    public double F() {
        if(Math.abs(error)<=coefficientThreshold){
            return secondaryCoefficients.F;
        }
        else return primaryCoefficients.F;
    }
    2 usages
    public double LowerLimit() {
        if(Math.abs(error)<=coefficientThreshold){
            return secondaryCoefficients.LowerLimit;
        }
        else return primaryCoefficients.LowerLimit;
    }
    2 usages
    public double UpperLimit() {
        if(Math.abs(error)<=coefficientThreshold){
            return secondaryCoefficients.UpperLimit;
        }
```

Pentru a spori acuratețea sistemului, viteza este menținută constantă în timpul aruncării artefactelor cu ajutorul unui controller dublu de PID-uri.

Acesta este implementat prin două seturi de **coeficienți PIDF**. Cel primar este mai agresiv, iar cel secundar este cel fin. *Astfel, rolul primului este de a aduce cât de rapid posibil eroarea sub un anumit prag, iar de acolo minimizarea erorii este efectuată de al doilea set.*

Sistemul este conceput să folosească integrala numai când este cu adevărat utilă, cazurile propice fiind acelea când motorul nu își atinge saturația sau când și-o atinge, însă eroarea ajută la ieșirea din saturație.

```
@Override
public void updateError(double error) {
    long now = System.nanoTime();
    deltaTimeNano = now - previousUpdateTimeNano;
    previousUpdateTimeNano = now;

    double dt = deltaTimeNano / 1e9;
    if (dt <= 0) return;

    previousError = this.error;
    this.error = error;

    // ----- Derivative -----
    errorDerivative = (error - previousError) / dt;
    previousDerivative = filteredDerivative;
    filteredDerivative = T() * previousDerivative + (1 - T()) * errorDerivative;

    // ----- Anti-Windup (Conditional Integration) -----
    double candidateIntegral = errorIntegral + error * dt;

    double predictedRaw = error * P() + filteredDerivative * D() + candidateIntegral * I() + F() * feedForwardInput;

    double predictedClamped = MathFunctions.clamp(predictedRaw, LowerLimit(), UpperLimit());

    // Only integrate if not saturating further
    if (!(predictedRaw != predictedClamped && Math.signum(error)==Math.signum(predictedRaw))) {
        errorIntegral = candidateIntegral;
```



În funcție de launch zone-ul din care trage robotul, am determinat experimental două randamente.

Dual PIDF Controller creat de noi, prin research și testare

```
        double rpm=((v0/razaShooter)/(2*Math.PI)*60)/randamentVitezaLong;

        double encoderSpeed=rpm/60*28;
        setSpeed(encoderSpeed, updatePutza);
    }
    else {
        double v0=Math.sqrt((g*distance*distance*(1+tanAlpha*tanAlpha))/(2*(distance*tanAlpha-targetHeight+h0)));
        double rpm=((v0/razaShooter)/(2*Math.PI)*60)/randamentViteza;

        double encoderSpeed=rpm/60*28;
        setSpeed(encoderSpeed, updatePutza);
    }
```

În scopul de a eficientiza timpul lansării bilelor, am optat să implementăm două viteze intermediare, *una pentru apropiere* și *una pentru îndepărtare*. Acestea pot fi păstrate constant pe tot parcursul meciului și schimbate în funcție de locul de tragere, astfel viteza dorită putând fi atinsă mult mai repede.

```
2 usages
private static double restShortSpeed=1100,restLongSpeed=1300;

5 usages
```

```
            if(Shooter.get().getRestSpeed()==0){
                Shooter.get().setRestSpeed(restShortSpeed);
            }
            else if(Shooter.get().getRestSpeed()==restShortSpeed){
                Shooter.get().setRestSpeed(restLongSpeed);
            }
            else {
                Shooter.get().setRestSpeed(0);
            }
            if(!shooting)
                Shooter.get().toRest();
```

În timpul perioadei de autonomie, **shooter-ul este gândit să utilizeze o viteză statică**, iar stările acestuia sunt modelate printr-un *Finite State Machine* care este apelat în funcția de loop.

```
4 usages
public void setAutoSpeed(double speed) { autoShootSpeed=speed; }
```

```
public void updateAutoShootWithSpeed(){
    if(autoShootSorted)
    {
        updateAutoShootWithSpeedSorted();
        return;
    }
    switch (autoShootState){
        case INIT:
            setSpeed(autoShootSpeed, updatePutza: true);
            canShoot=false;
            autoShootState=AutoShootStates.GET_TO_SHOOT_SPEED;
            break;
        case GET_TO_SHOOT_SPEED:
            if(!canShoot){
                maintainSpeed();
            }
            if(canShoot){

                stopperAllow();
                Intake.get().setState(Intake.IntakeStates.TO_SHOOTER, intakePowerShooter);
                autoShootState=AutoShootStates.OPEN_FIRE;
                shooterTimer = new Timer();
            }
            break;
        case OPEN_FIRE:
            if(shooterTimer.getElapsedTimeSeconds()> autoShootTime)
            {
                endAutoShoot();
            }
            maintainSpeed();
            break;
    }
```

Varianta precedentă avea în ambele seturi de coeficienți PIDF integrala egală cu 0.

De asemenea, controller-ul de PID-uri era resetat după fiecare iterație.

După o serie de teste am sesizat că i**ntegrala trebuie să fie mai mare ca 0** pentru a elimina eroarea statică de viteză, iar aceasta nu trebuie resetată pentru că își pierde din „memorie”, fiind recursivă:

$$I_k = I_{k-1} + e_k \Delta t$$

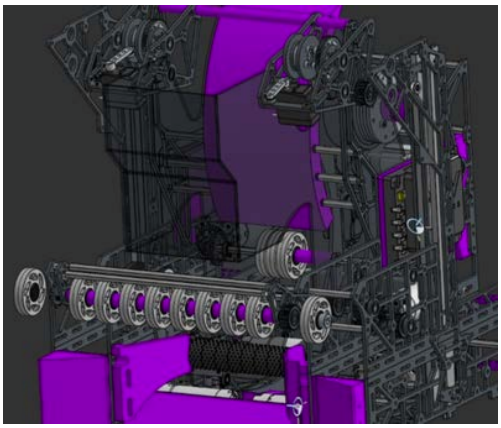
```
private static FilteredPIDFCoefficients speedPidCoeff = new FilteredPIDFCoefficients( p: 0.1, i: 0, d: 0, t: 0, f: 0);/// f=0
private static FilteredPIDFCoefficients speedPidCoeffSecondary = new FilteredPIDFCoefficients( p: 0.003, i: 0, d: 0.00001, t: 0, f: 0);
private FilteredPIDFController speedPid = new FilteredPIDFController(speedPidCoeff);
1 usage
private static double speedSecondaryThreshold=100;
```

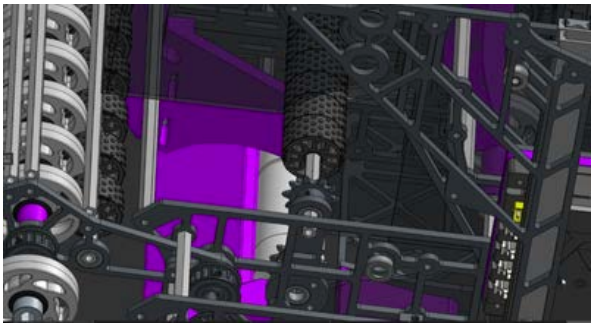
```
public void maintainSpeed(boolean updatePutza){
    speedPid.updatePosition(-rightMotor.getVelocity());
    updateFilteredPIDFCoefficients();
    if(Math.abs(speedPid.getError())>=speedThresholdLimbutza){
        timerLimbutza.resetTimer();
    }
    if(timerLimbutza.getElapsedTimeSeconds()>=timeThresholdLimbutza && speedPid.getTargetPosition
    {
        if(canShoot==false)
            stopperAllow();
        canShoot=true;
    }

    double power = MathFunctions.clamp(speedPid.run(), lower: 0, upper: 1);
    leftMotor.setPower(power);
    rightMotor.setPower(power);

}
1 usage
public double getMotorRpm() { return -rightMotor.getVelocity()/28*60; }
10 usages
public void setSpeed(double speed, boolean updatePutza){
    if(speedPid.getTargetPosition()==speed) return;
    speedPid.reset();
    speedPid.setTargetPosition(speed);
    if(updatePutza)
    {
        stopperBlock();
        canShoot=false;
        timerLimbutza.resetTimer();
    }
}
2 usages
public void toRest(){
    if(speedPid.getTargetPosition()==restSpeed) return;
    speedPid.reset();
    speedPid.setTargetPosition(restSpeed);
    stopperBlock();
    canShoot=false;
    timerLimbutza.resetTimer();
```

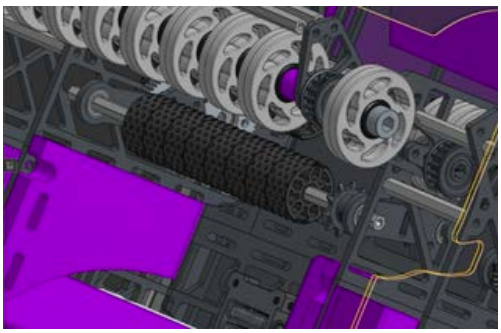
4.7 INTAKE SI SORTARE

Intake-ul are rolul de a gestiona flow-ul artefactelor atât în timpul teleop-ului cât și în timpul autonomiei. Acesta constă într-un finite state machine care controlează poziția one way door-ului și puterea intake-ului propriu-zis respectiv a midroller-ului.

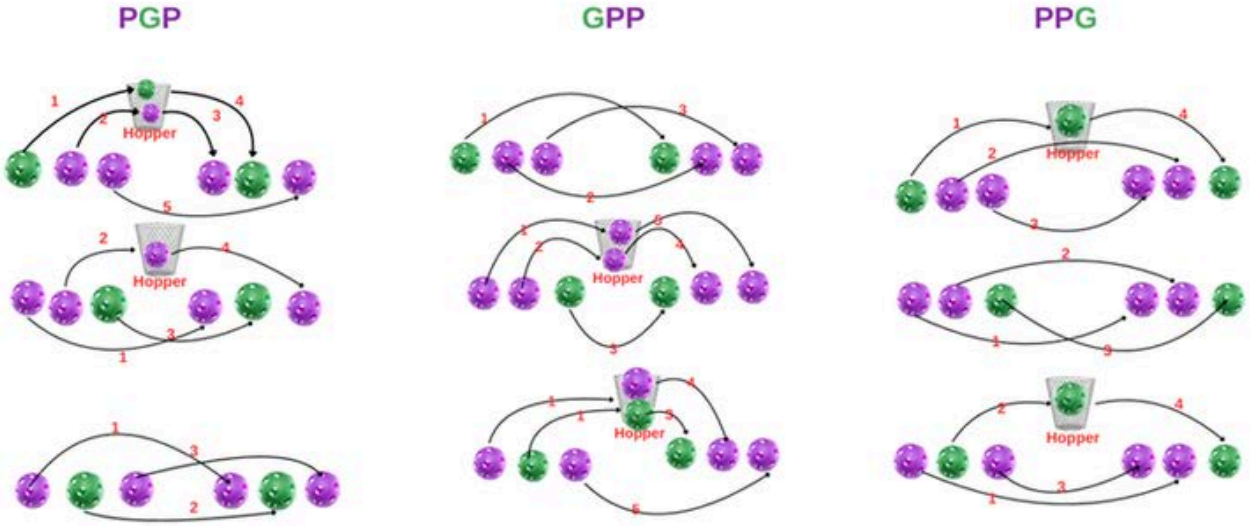
```
/// Intake states
public enum IntakeStates {
    5 usages
    NONE,
    TO_HOPPER,
    TO_SHOOTER,
    13 usages
    HOPPER_TO_SHOOTER,
}
```







Metoda **setState** primește ca parametri *starea dorită pentru intake, cât și un multiplier pentru a gestiona viteza și sensul de rotație în caz de nevoie*. Prin acest Finite State sunt controlate pozițiile bilei, realizându-se în cele din urmă sortarea.



```
/// Owdoor
2 usages
private static double owdoorUpPos=0.11,owdoorMidPos=0.58,owdoorDownPos=0.97;
3 usages
private double lastOwdoorPos=-1;
1 usage
public void owdoorDown(){
    if(lastOwdoorPos==owdoorDownPos) return;
    owdoor.setPosition(owdoorDownPos);
}
1 usage
public void owdoorUp(){
    if(lastOwdoorPos==owdoorUpPos) return;
    owdoor.setPosition(owdoorUpPos);
}
2 usages
public void owdoorMid(){
    if(lastOwdoorPos==owdoorMidPos) return;
    owdoor.setPosition(owdoorMidPos);
}

/// Motor powers
4 usages
private void setIntakePower(double power){
    intakeMotor.setPower(power);
}
4 usages
private void setMidrollerPower(double power){
    midrollerMotor.setPower(power);
}
```

Codul reprezintă partea low-level a sistemului de intake, care controlează direct componentele hardware: un servo și motoarele mecanismului.

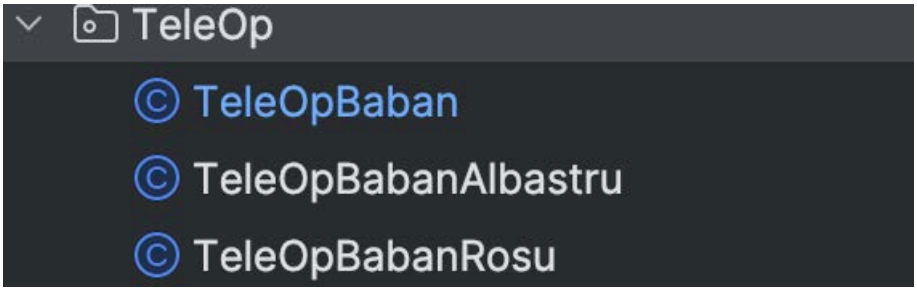
Pentru servo-ul “owdoor” sunt definite trei poziții: **up**, **mid** și **down**, *corespunzătoare pozițiilor mecanice ale ușii de sortare.* Metodele **owdoorUp()**, **owdoorMid()** și **owdoorDown()** mută servo-ul în aceste poziții, verificând mai întâi dacă acesta se află deja în poziția dorită pentru a evita trimiterea de comenzi redundante.

*În partea a doua sunt definite metodele **setIntakePower()** și **setMidrollerPower()**, care setează puterea motoarelor responsabile pentru rularea mecanismului de intake și transportul elementelor prin sistem.*

Acestea sunt metodele cele mai low-level care interacționează cu componenta hardware. Ele sunt folosite mai departe, adăugând un nivel de abstractizare. Un lucru important de menționat este că **sensibilitatea servoului a fost programată extern printr-un Axon Servo Programmer**, astfel ca servoul să aibă un threshold în atingerea poziției dorite, pentru a elimina oscilațiile.

4.8 TELEOP

TeleOp-ul are o structură clară, fiind organizat în trei clase: **o clasă principală și două clase specifice, în funcție de culoarea alianței**.



În clasa principală de **TeleOp** sunt implementate toate celelalte sisteme, iar inputul controller-ului este procesat prin intermediul funcției **processControllerInput()**.

```
private void processControllerInput(){

    /// Intake
    double intakePowerMultiplier;

    if(gp1.left_trigger >= 0.75){
        intakePowerMultiplier=-1;
    }
    else if(gp1.right_trigger >= 0.75){
        intakePowerMultiplier=1;
    }
    else {
        intakePowerMultiplier=0;
    }

    if(gp1.dpad_up.value && !gp1.dpad_up.lastValue){
        switch (Intake.get().getState()){
            case TO_HOPPER:
            case HOPPER_TO_SHOOTER:
                Intake.get().setState(IntakeStates.TO_SHOOTER, intakePowerMultiplier);
                break;
            case NONE:
            case TO_SHOOTER:
                Intake.get().setState(IntakeStates.HOPPER_TO_SHOOTER, intakePowerMultiplier);
                break;
        }
    }
```

```
if(gp1.dpad_down.value && !gp1.dpad_down.lastValue){
    switch (Intake.get().getState()){
        case NONE:
        case TO_SHOOTER:
            Intake.get().setState(IntakeStates.TO_HOPPER, intakePowerMultiplier);
            break;
        case HOPPER_TO_SHOOTER:
            Intake.get().setState(IntakeStates.TO_SHOOTER, intakePowerMultiplier);
            break;
    }
}
switch (Intake.get().getState()){
    case TO_SHOOTER:
        Intake.get().setState(IntakeStates.TO_SHOOTER, intakePowerMultiplier);
        break;
    case TO_HOPPER:
        Intake.get().setState(IntakeStates.TO_HOPPER, intakePowerMultiplier);
        break;
    case HOPPER_TO_SHOOTER:
        Intake.get().setState(IntakeStates.HOPPER_TO_SHOOTER, intakePowerMultiplier);
        break;
    case NONE:
    default:
        break;
```

```
if(gp1.y.value){
    MecanumDrive.get().lockIn();
}

if(gp1.right_stick_button.value && !gp1.right_stick_button.lastValue){

    if(Shooter.get().getRestSpeed()==0){
        Shooter.get().setRestSpeed(restShortSpeed);
    }
    else if(Shooter.get().getRestSpeed()==restShortSpeed){
        Shooter.get().setRestSpeed(restLongSpeed);
    }
    else {
        Shooter.get().setRestSpeed(0);
    }
    if(!shooting)
        Shooter.get().toRest();
}

if(gp1.right_bumper.value && !gp1.right_bumper.lastValue){
    if(gp1.right_bumper.pressed%2==0){
        Shooter.get().toRest();
        shooting=false;
    }
    else{
        Shooter.get().shoot(LocalizerManager.get().getDistanceFromGoal(DistanceUnit.METER),  updatePutza: true);
        updateDistanceTimer.resetTimer();
        shooting=true;
        MecanumDrive.get().lockIn();
    }
}
```

```
if(gp1.dpad_right.value && !gp1.dpad_right.lastValue) {
    if(Shooter.get().getStopperBlocked()){
        Shooter.get().stopperAllow();
    }
    else Shooter.get().stopperBlock();
}

if(gp1.dpad_right.value && !gp1.dpad_right.lastValue) {
    if(Shooter.get().getStopperBlocked()){
        Shooter.get().stopperAllow();
    }
    else Shooter.get().stopperBlock();
}

if(gp1.dpad_left.value && !gp1.dpad_left.lastValue) {
    LocalizerManager.get().setHeading(resetHeading);

    resetOdometryComboTimer.resetTimer();
}
if(gp1.a.value && !gp1.a.lastValue)
    LocalizerManager.get().resetOdometryPose();

if(gp1.left_bumper.value)
    MecanumDrive.get().setSpeedMultiplier(0.4);
else MecanumDrive.get().setSpeedMultiplier(1);
```

- **Initialising**

```
public void init(Pose2D resetOdometryPose, double resetHeading, Pose2D goalPose, Gamepad gamepad1, Gamepad gamepad2, Telemetry telemetry, HardwareMap hardwareMap
    this.telemetry=telemetry;
    this.resetOdometryPose=resetOdometryPose;
    this.resetHeading=resetHeading;

    gp1 = new GAMEPAD(gamepad1, telemetry);
    gp2 = new GAMEPAD(gamepad2, telemetry);

    panelsTelemetry = PanelsTelemetry.INSTANCE.getTelemetry();

    Intake.get().init(hardwareMap, panelsTelemetry.getWrapper());
    Shooter.get().init(hardwareMap, panelsTelemetry.getWrapper());

    MecanumDrive.get().init(hardwareMap, panelsTelemetry.getWrapper());

    /// Localization
    LocalizerManager.get().init(hardwareMap, panelsTelemetry.getWrapper());
    LocalizerManager.get().setGoalPose(goalPose);


    resetOdometryComboTimer = new Timer();
}
```

```
public void loop() {

    gp1.run();
    gp2.run();

    LocalizerManager.get().update();

    updateShootingPose();

    processControllerInput();
    MecanumDrive.get().run((Math.abs(gp1.left_stick_x)>=0.05) ? gp1.left_stick_x : 0,
            (Math.abs(gp1.left_stick_y)>=0.05) ? gp1.left_stick_y : 0, (Math.abs(gp1.right_stick_x)>=0.05)? gp1.right_stick_x : 0);
    Shooter.get().maintainSpeed(shooting);

    panelsTelemetry.getWrapper().addData( caption: "gp1", format: "x: %.4f; y: %.4f; rx: %.4f", gp1.left_stick_x, gp1.left_stick_y, gp1.right_stick_x);
    panelsTelemetry.getWrapper().addData( caption: "lockedIn", MecanumDrive.get().getLockedIn());
    Shooter.get().addTelemetry();
    panelsTelemetry.getWrapper().addData( caption: "loop time", loopTimer.getElapsedTime());
    panelsTelemetry.update(telemetry);
    loopTimer.resetTimer();
}
```

- **Start**

```
public void start(){
    Shooter.get().stopperBlock();
    LimelightHandler.get().start();
    LimelightHandler.get().setLocalizerPipeline();
    Intake.get().setState(IntakeStates.TO_SHOOTER,  powerMultiplier: 0);
}
```

Clasa pentru alianța albastră

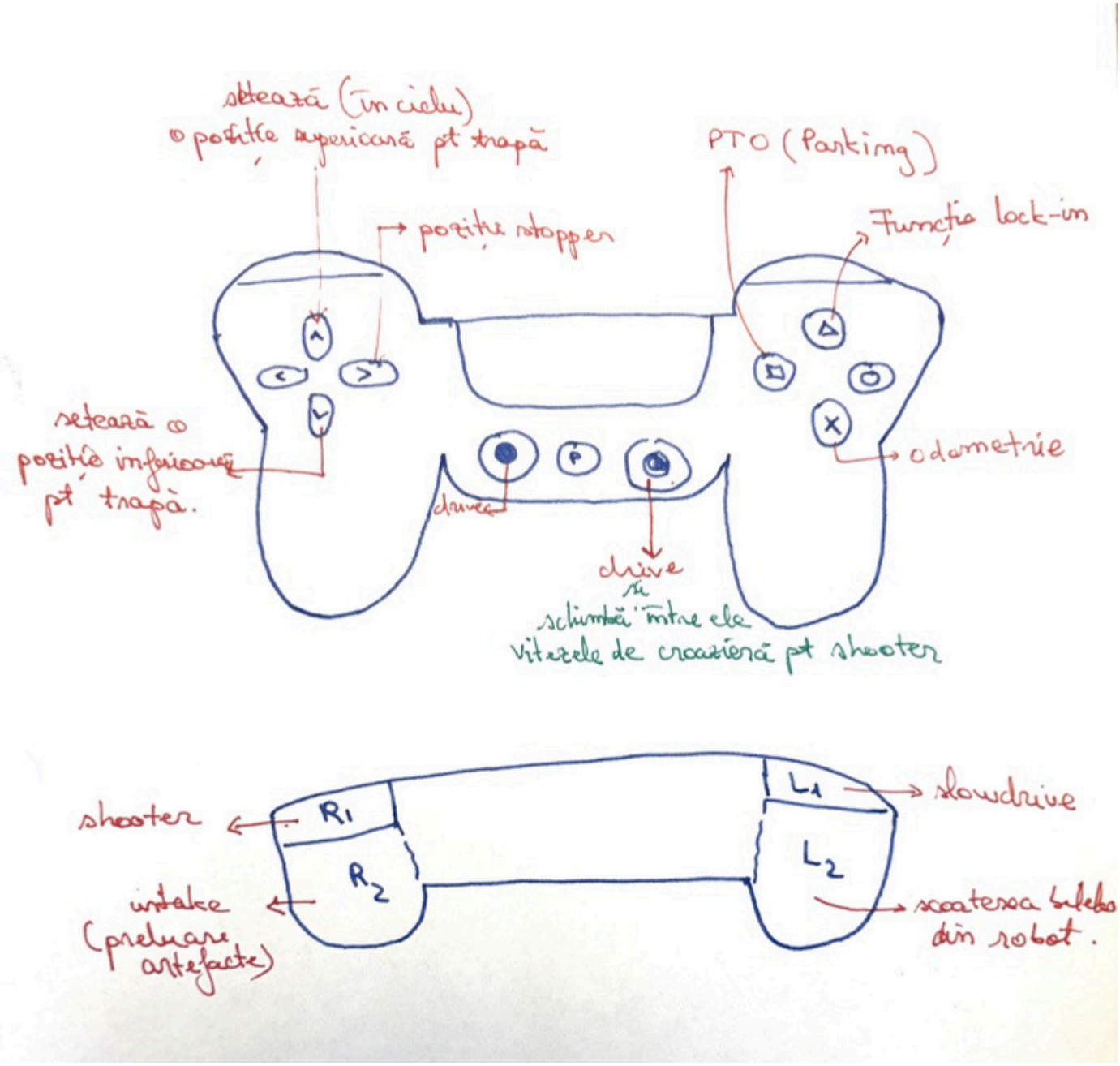
```
no usages
@TeleOp(group="teleop", name="TELEOP ALBASTRU")
@Configurable
public class TeleOpBabanAlbastru extends OpMode {
    1 usage
    private static Pose2D goalPose = new Pose2D(DistanceUnit.METER,  x: -1.75,  y: -1.7, AngleUnit.DEGREES,  heading: 0);
    1 usage
    private static Pose2D resetOdometryPose = new Pose2D(DistanceUnit.METER,  x: 0,  y: 0, AngleUnit.DEGREES,  heading: 0);
    1 usage
    private static double resetHeading = 0;
    4 usages
    private TeleOpBaban teleop;

    @Override
    public void init() {

        teleop = new TeleOpBaban();
        teleop.init(resetOdometryPose, resetHeading, goalPose, gamepad1, gamepad2, telemetry, hardwareMap);
    }
    @Override
    public void loop() { teleop.loop(); }

    @Override
    public void start() { teleop.start(); }
}
```

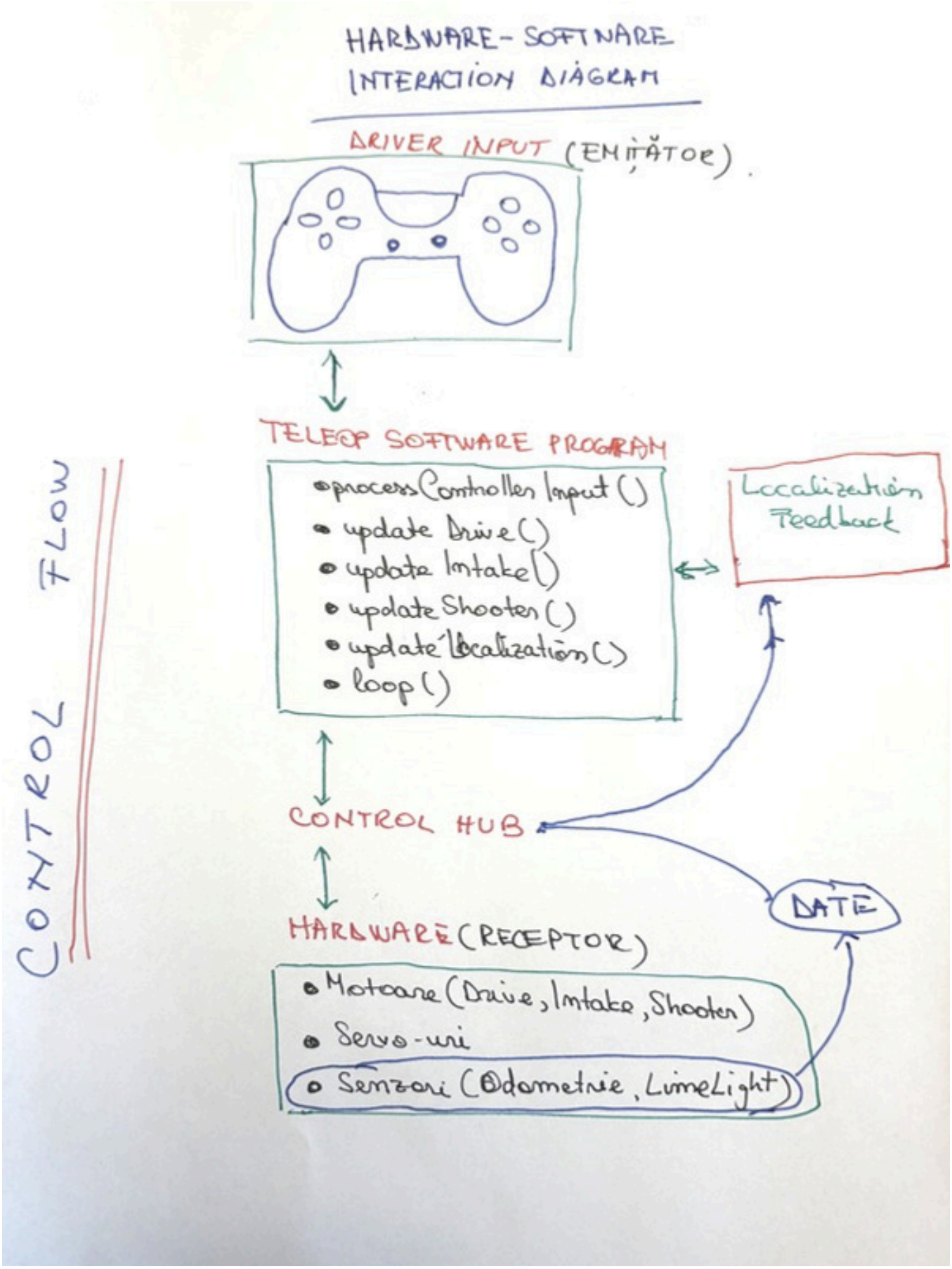
Clasa pentru alianța roșie

```
no usages
@TeleOp(group="teleop", name="TELEOP ROSU")
@Configurable
public class TeleOpBabanRosu extends OpMode {
    1 usage
    private static Pose2D goalPose = new Pose2D(DistanceUnit.METER, x: -1.75, y: 1.7, AngleUnit.DEGREES, heading: 0);
    1 usage
    private static Pose2D resetOdometryPose = new Pose2D(DistanceUnit.METER, x: 0, y: 0, AngleUnit.DEGREES, heading: 0);
    1 usage
    private static double resetHeading = 0;
    4 usages
    private TeleOpBaban teleop;

    @Override
    public void init() {

        teleop = new TeleOpBaban();
        teleop.init(resetOdometryPose, resetHeading, goalPose, gamepad1, gamepad2, telemetry, hardwareMap);
    }
    @Override
    public void loop() { teleop.loop(); }

    @Override
    public void start() { teleop.start(); }
}
```

Datorită faptului că sistemul de localizare folosit în TeleOp este conceput pe baza poziției robotului în funcție de teren, inițializarea goalPose-ului este obligatorie.

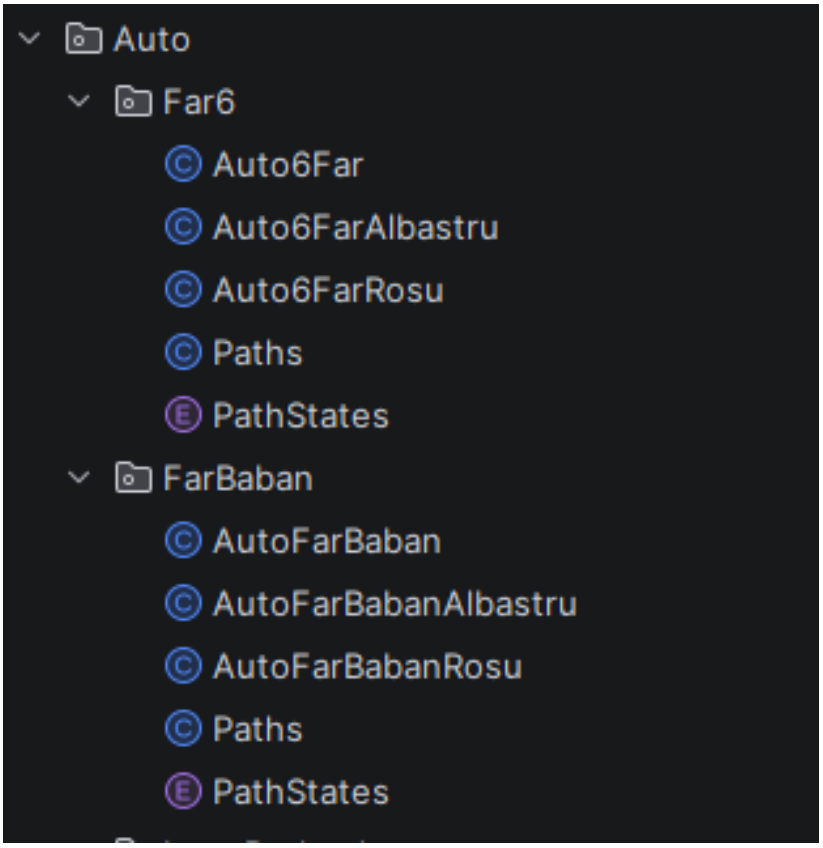
Analiza controalelor




HARDWARE - SOFTWARE INTERACTION DIAGRAM
DRIVER INPUT (EMIȚĂTOR)
TELEOP SOFTWARE PROGRAM
processControllerInput()
updateDrive()
updateIntake()
updateShooter()
updateLocalization()
loop()
Localization Feedback
CONTROL HUB
DATE
HARDWARE (RECEPTOR)
Motoare (Drive, Intake, Shooter)
Servo-uri
Senzori (Odometrie, LimeLight)
CONTROL FLOW

4.9 AUTONOMIE

Autonomiile sunt structurate în *5 clase*: **o clasă specifică pentru path-uri**, **un enum care conține stările**, **clasa principală** unde este implementat Finite State Machine-ul și **alte două clase** în funcție de culoarea alianței.



Unele dintre autonomiile pe care le-am utilizat sunt:

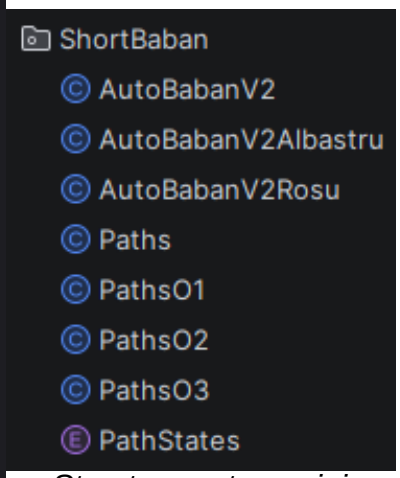
- tragerea **preload-ului de la far și de la short**, acestea fiind cele mai simple variante: *robotul pornește din poziția de start, se aliniază cu obiectivul folosind sistemul de localizare și execută secvența de tragere a preload-ului.* După realizarea acestei acțiuni, robotul se poate opri sau continua cu alte acțiuni planificate, în funcție de strategia aleasă. Aceste autonomii sunt folosite frecvent deoarece sunt rapide, sigure și reduc riscul de erori în timpul meciului.

```java
public class AutoGFar {
    public void autonomousPathUpdate() {
        switch(pathState)
        {
            case Init:

                pathState= PathStates.ScorePreload;
                follower.followPath(paths.scorePreload,  holdEnd: true);
                Shooter.get().resetAutoShootState();
                break;

            case ScorePreload:
                if(!follower.isBusy())
                {
                    if (Shooter.get().getAutoShootState() == Shooter.AutoShootStates.NONE)
                        Shooter.get().startAutoShoot(targetId);
                    if (Shooter.get().getAutoShootState() == Shooter.AutoShootStates.ENDED) {
                        follower.followPath(paths.turnFromScore);
                        pathState = PathStates.TurnFormScore;
                        Shooter.get().resetAutoShootState();

                    }
                }
                break;
            case TurnFormScore:
                if(!follower.isBusy())
                {
                    pathState=PathStates.PickupWall;
                    follower.followPath(paths.pickupWall,  maxPower: 0.5,  holdEnd: true);
                }
                break;
            case PickupWall:
                if(!follower.isBusy())
                {
                    pathState=PathStates.ScoreWall;
                    follower.followPath(paths.scoreWall, holdEnd: true);
                }
                break;
            case ScoreWall:
                if(!follower.isBusy()) {
                    if (Shooter.get().getAutoShootState() == Shooter.AutoShootStates.NONE)
                        Shooter.get().startAutoShoot(targetId);
                    if (Shooter.get().getAutoShootState() == Shooter.AutoShootStates.ENDED) {
                        pathState = PathStates.Leave;
                        follower.followPath(paths.leave);
                        Shooter.get().resetAutoShootState();
                    }
                }
                break;
            case Leave:
                if(!follower.isBusy())
                {
                    stop();
                }
                break;
        }
    }
```

Autonomia preload-ului de la far și de la short

- **12 bile scorate, dintre care 7 sortate**

```
public class AutoBabanV2 {
    public void autonomousPathUpdate()
    {
        switch (pathState){
            case Idle:
                break;
            case Init:
                pathState = PathStates.ScorePreload;
                follower.followPath(paths.scorePreload,  holdEnd: true);
                break;
            case ScorePreload:
                if(!follower.isBusy())
                {
                    if(Shooter.get().getAutoShootState()== Shooter.AutoShootStates.NONE)
                    {
                        Shooter.get().setIntakePowerShooter(1);
                        Shooter.get().setAutoShootTime(1.5);
                        Shooter.get().startAutoShoot(targetId);
                    }
                    if(Shooter.get().getAutoShootState()== Shooter.AutoShootStates.ENDED){
                        follower.followPath(paths.obeliskview);
                        pathState= PathStates.ToObelisk;
                        Shooter.get().resetAutoShootState();
                    }
                }
                break;
            case ToObelisk:
                if(!follower.isBusy()) {
                    if(LimelightHandler.get().updateObeliskID()) {

                        switch (LimelightHandler.get().getGreenPosition()){
                            case 1:
                                paths=paths01;
                                break;
                            case 2:
                                paths=paths02;
                                break;
                            case 3:
                                paths=paths03;
                                break;
                        }

                        pathState = PathStates.ToPickupMid; pathCount=0;
                        follower.followPath(paths.toPickupMid);
                    }
                }
                break;
            case ToPickupMid:
                if(!follower.isBusy())
                {
                    follower.followPath(paths.pickupMid);
                    pathState= PathStates.PickupMid;
                }
                break;
            case PickupMid:
                if(!follower.isBusy())
                {
```



Structura autonomiei descrise

```
1 usage
public void autonomousPathUpdate()
{
    switch (pathState){
        case Idle:
            break;
        case Init:
            pathState = PathStates.ScorePreload;
            follower.followPath(paths.scorePreload,  holdEnd: true);
            break;
        case ScorePreload:
            if(!follower.isBusy())
            {
                if(Shooter.get().getAutoShootState()== Shooter.AutoShootStates.NONE)
                {
                    Shooter.get().setIntakePowerShooter(1);
                    Shooter.get().setAutoShootTime(1.5);
                    Shooter.get().startAutoShoot(targetId);
                }
                if(Shooter.get().getAutoShootState()== Shooter.AutoShootStates.ENDED){
                    follower.followPath(paths.obeliskview);
                    pathState= PathStates.ToObelisk;
                    Shooter.get().resetAutoShootState();
                }
            }
            break;
        case ToObelisk:
            if(!follower.isBusy()) {
                if(LimelightHandler.get().updateObeliskID()) {

                    switch (LimelightHandler.get().getGreenPosition()){
                        case 1:
                            paths=paths01;
                            break;
                        case 2:
                            paths=paths02;
                            break;
                        case 3:
                            paths=paths03;
                            break;
                    }

                    pathState = PathStates.ToPickupMid; pathCount=0;
                    follower.followPath(paths.toPickupMid);
                }
            }
            break;
        case ToPickupMid:
            if(!follower.isBusy())
            {
                follower.followPath(paths.pickupMid);
                pathState= PathStates.PickupMid;
            }
            break;
        case PickupMid:
```

```
        follower.followPath(paths.openGateFromMid);
        pathState= PathStates.OpenGate;
        pathtimer.resetTimer();
        pathCount=0;
    }
    break;
case OpenGate:
    if(!follower.isBusy()){
        pathCount++;
        if(pathCount==1) pathtimer.resetTimer();
        if(pathtimer.getElapsedTime() >= gateTime)
        {
            follower.followPath(paths.scoreMid, holdEnd: true);
            pathState= PathStates.ScoreMid;
        }
    }

    break;
case ScoreMid:
    if(!follower.isBusy())
    {
        if(Shooter.get().getAutoShootState()== Shooter.AutoShootStates.NONE)
        {
            if(LimelightHandler.get().getGreenPosition()==2)
            {
                Shooter.get().setIntakePowerShooter(1);
                Shooter.get().setAuto1BallShootTime(ballShootTime);
                Shooter.get().setAuto1BallWaitTime(ballWaitTime);
                Shooter.get().setAutoShootSorted(true);
            }
            else
            {
                Shooter.get().setIntakePowerShooter(1);
                Shooter.get().setAutoShootTime(1.5);
                Shooter.get().setAutoShootSorted(false);
            }
            Shooter.get().startAutoShoot(targetId);
        }
        if(Shooter.get().getAutoShootState()== Shooter.AutoShootStates.ENDED) {
            follower.followPath(paths.toPickupUp);
            pathState = PathStates.ToPickupUp;
            Shooter.get().resetAutoShootState();
        }
    }
    break;
case ToPickupUp:
    if(!follower.isBusy()){
        pathState= PathStates.PickupUp; pathCount=0;
        follower.followPath(paths.pickupUp, (LimelightHandler.get().getGreenPosition()==3? 1 : maxPowerSort), holdEnd: false);
    }
    break;
case PickupUp:
    if(!follower.isBusy()){
        if(LimelightHandler.get().getGreenPosition()==3){
```

```
        if(LimelightHandler.get().getGreenPosition()==3){
            pathState = PathStates.ScoreUp; pathCount=0;
            follower.followPath(paths.scoreUp,  holdEnd: true);
        }
        else {
            pathCount++;
            if(pathCount==1){
                Intake.get().setState(Intake.IntakeStates.HOPPER_TO_SHOOTER,  powerMultiplier: 1);
                pathState = PathStates.ReleaseUp;
                pathtimer.resetTimer();
            }
            else
            {
                pathState = PathStates.ScoreUp; pathCount=0;
                follower.followPath(paths.scoreUp,  holdEnd: true);
            }
        }
    }
    break;
case ReleaseUp:
    if(pathtimer.getElapsedTime()>=releaseTime2)
    {
        Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier: 1);
        pathState = PathStates.PickupUp;
        follower.resumePathFollowing();
    }
    break;
case ScoreUp:
    if(!follower.isBusy())
    {
        if(Shooter.get().getAutoShootState()== Shooter.AutoShootStates.NONE)
        {
            Shooter.get().setIntakePowerShooter(1);
            Shooter.get().setAuto1BallShootTime(ballShootTime);
            Shooter.get().setAuto1BallWaitTime(ballWaitTime);
            Shooter.get().setAutoShootSorted(true);
            Shooter.get().startAutoShoot(targetId);
        }
        if(Shooter.get().getAutoShootState()== Shooter.AutoShootStates.ENDED) {
            follower.followPath(paths.toPickupDown);
            pathState = PathStates.ToPickupDown;
            Shooter.get().resetAutoShootState();
        }
    }
    break;
case ToPickupDown:
    if(!follower.isBusy()){
        pathState= PathStates.PickupDown; pathCount=0;
        follower.followPath(paths.pickupDown, (LimelightHandler.get().getGreenPosition()==3? maxPowerSort : 1),  holdEnd: false);
    }
    break;
case PickupDown:
    if(!follower.isBusy()){
```

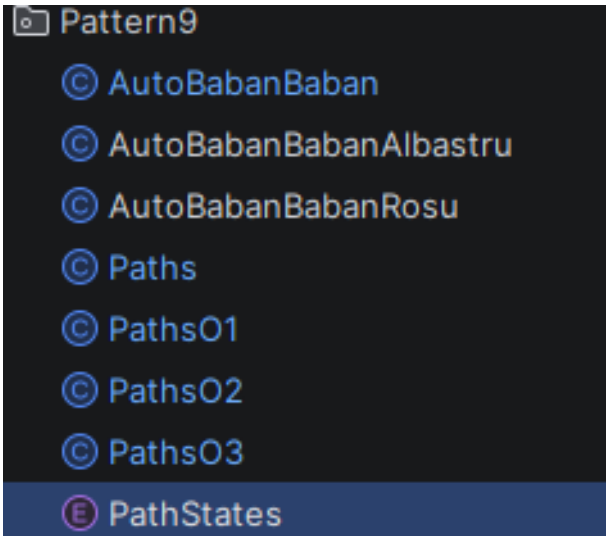
```
        if(!follower.isBusy()){

            if(LimelightHandler.get().getGreenPosition()==3){
                pathCount++;
                if(pathCount==1){
                    Intake.get().setState(Intake.IntakeStates.HOPPER_TO_SHOOTER,  powerMultiplier: 1);
                    pathState = PathStates.ReleaseDown;
                    pathtimer.resetTimer();
                }
                else
                {
                    pathState = PathStates.ScoreDown; pathCount=0;
                    follower.followPath(paths.scoreDown,  holdEnd: true);
                }
            }
            else {
                pathState = PathStates.ScoreDown;
                follower.followPath(paths.scoreDown,  holdEnd: true);
            }

        }
        break;
    case ReleaseDown:
        if(pathtimer.getElapsedTime()>=releaseTime1)
        {
            Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier: 1);
            pathState = PathStates.PickupDown;
            follower.resumePathFollowing();
        }
        break;
    case ScoreDown:
        if(!follower.isBusy())
        {
            if(Shooter.get().getAutoShootState()== Shooter.AutoShootStates.NONE)
            {
                if(LimelightHandler.get().getGreenPosition()==2)
                {
                    Shooter.get().setIntakePowerShooter(1);
                    Shooter.get().setAutoShootTime(1.5);
                    Shooter.get().setAutoShootSorted(false);
                }
                else
                {
                    Shooter.get().setIntakePowerShooter(0.7);
                    Shooter.get().setAuto1BallShootTime(ballShootTime);
                    Shooter.get().setAuto1BallWaitTime(ballWaitTime);
                    Shooter.get().setAutoShootSorted(true);
                }
                Shooter.get().startAutoShoot(targetId);
            }
            if(Shooter.get().getAutoShootState()== Shooter.AutoShootStates.ENDED) {
                follower.followPath(paths.park);
```

```
                    pathState = PathStates.Park;
                    Shooter.get().resetAutoShootState();
                }
            }
            break;
        case Park:
            if(!follower.isBusy()){
                stop();
            }
            break;
    }
}

public void loop() {

    follower.update();
    autonomousPathUpdate();
    if(Shooter.get().getAutoShootState() != Shooter.AutoShootStates.NONE)
        Shooter.get().updateAutoShootWithSpeed();
    else Shooter.get().maintainSpeed( updatePutza: false);

    panelsTelemetry.getWrapper().addData( caption: "green position", LimelightHandler.get().getGreenPosition());
    panelsTelemetry.getWrapper().addData( caption: "path state", pathState);
    panelsTelemetry.getWrapper().addData( caption: "path timer", pathtimer.getElapsedTimeSeconds());
    panelsTelemetry.getWrapper().addData( caption: "x", follower.getPose().getX());
    panelsTelemetry.getWrapper().addData( caption: "y", follower.getPose().getY());
    panelsTelemetry.getWrapper().addData( caption: "heading", follower.getPose().getHeading());
    Shooter.get().addTelemetry();
    panelsTelemetry.update(telemetry);
}



public void init() {
    panelsTelemetry = PanelsTelemetry.INSTANCE.getTelemetry();

    follower = Constants.createFollower(hardwareMap);
    paths01 = new Paths01();
    paths01.init(follower, panelsTelemetry.getWrapper(), mirrorPoses);
    paths02 = new Paths02();
    paths02.init(follower, panelsTelemetry.getWrapper(), mirrorPoses);
    paths03 = new Paths03();
    paths03.init(follower, panelsTelemetry.getWrapper(), mirrorPoses);
    paths=paths01;
    follower.setStartingPose(paths.startingPose);

    Intake.get().init(hardwareMap, panelsTelemetry.getWrapper());
    Shooter.get().init(hardwareMap, panelsTelemetry.getWrapper());
    Shooter.get().setTimerLimbutza(0.8);
    LimelightHandler.get().init(hardwareMap, panelsTelemetry.getWrapper());
```

```
        pathState= PathStates.Init;

        pathtimer = new Timer();

        panelsTelemetry.getWrapper().addLine( lineCaption: "init finished");
    }

    public void stop() { LimelightHandler.get().stop(); }

    1 usage
    private static double shooterSpeed=1050;
    public void start() {
        LimelightHandler.get().start();
        LimelightHandler.get().setObeliskPipeline();
        Shooter.get().stopperBlock();
        pathState = PathStates.Init;
        Shooter.get().setAutoSpeed(shooterSpeed);
        Intake.get().setState(Intake.IntakeStates.TO_SHOOTER, powerMultiplier: 0)
    }
}
```

- În loop() este actualizat follower-ul din PedroPathing, care urmărește traiectoriile definite pentru autonomie. De asemenea, este apelată funcția de update a sistemului de shoot, specifică autonomiei, care trebuie să mențină o viteză constantă a shooter-ului (shooterSpeed).
- În init() sunt inițializate sistemele utilizate: LimelightHandler, shooter-ul și intake-ul. Tot aici este setată și poziția de start a robotului, atribuită follower-ului.
- În start() este selectat pipeline-ul necesar pentru Obelisk, iar shooter-ului îi este setată viteza de tragere. În plus, sistemul de captare a bilelor este pus în poziția default, în care artefactul este direcționat către shooter.
- În stop() camera este oprită.

- **9 bile sortate**



Pentru a putea sorta cele trei spike-uri indiferent de motiv, am decis să implementăm câte o clasă de path-uri pentru fiecare pattern. **Sortarea** este realizată pe baza pozițiilor robotului, nu pe baza unor timere, ceea ce oferă o precizie și o consistență mai mare în execuția autonomiei.

```
blic void init(Follower follower, Telemetry telemetry, boolean mirror){

  pickupUpStart=new Pose( x 96,  y 84,  heading 0);pickupUpInter1=new Pose( x 114,  y 84,  heading 0);pickupUpInter2=new Pose( x 120,  y 84,  heading 0);pickupUpEnd=new Pose( x 124,  y 84,  heading 0);
  pickupMidStart=new Pose( x 96,  y 60,  heading 0);pickupMidInter1=new Pose( x 108,  y 60,  heading 0);pickupMidInter2=new Pose( x 113,  y 60,  heading 0);pickupMidEnd=new Pose( x 130,  y 60,  heading 0);
  pickupDownStart=new Pose( x 96,  y 36,  heading 0);pickupDownInter1=new Pose( x 0,  y 0,  heading 0);pickupDownInter2=new Pose( x 0,  y 0,  heading 0);pickupDownEnd=new Pose( x 130,  y 36,  heading 0);

  scoreDownControlPoint = new Pose( x 80, y 47);
  scoreMidControlPoint = new Pose( x 75, y 80);
  super.init(follower, telemetry, mirror);


  pickupDown = follower.pathBuilder()
          .addPath(new BezierLine(pickupDownStart,pickupDownEnd))
          .setLinearHeadingInterpolation(pickupDownStart.getHeading(), pickupDownEnd.getHeading())
          .addParametricCallback( t 0, ()-> Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier 1))
          .addParametricCallback( t 0.9, ()-> Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier 0))
          .build();

  pickupMid = follower.pathBuilder()
          .addPath(new BezierLine(pickupMidStart,pickupMidInter1))
          .setConstantHeadingInterpolation(pickupMidStart.getHeading())
          .addParametricCallback( t 0, ()-> Intake.get().setState(Intake.IntakeStates.TO_HOPPER,  powerMultiplier 1))
          .addParametricCallback( t 0.97, ()-> Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier 0))

          .addPath(new BezierLine(pickupMidInter1, pickupMidInter2))
          .setConstantHeadingInterpolation(pickupMidInter1.getHeading())
          .addParametricCallback( t 0.1, ()-> Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier 1))
          .addParametricCallback( t 0.97, ()-> Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier 0))

          .addPath(new BezierLine(pickupMidInter2, pickupMidEnd))
          .setConstantHeadingInterpolation(pickupMidInter2.getHeading())
          .addParametricCallback( t 0, follower::pausePathFollowing)
          .addParametricCallback( t 0.9, ()-> Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier 0))

          .build();

  pickupUp = follower.pathBuilder()
          .addPath(new BezierLine(pickupUpStart,pickupUpInter1))
          .setConstantHeadingInterpolation(pickupUpStart.getHeading())
          .addParametricCallback( t 0.1, ()-> Intake.get().setState(Intake.IntakeStates.TO_HOPPER,  powerMultiplier 1))
          .addParametricCallback( t 0.97, ()-> Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier 0))

          .addPath(new BezierLine(pickupUpInter1, pickupUpInter2))
          .setConstantHeadingInterpolation(pickupUpInter1.getHeading())
          .addParametricCallback( t 0.1, ()-> Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier 1))
          .addParametricCallback( t 0.97, ()-> Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier 0))

          .addPath(new BezierLine(pickupUpInter2, pickupUpEnd))
          .setConstantHeadingInterpolation(pickupUpInter2.getHeading())
          .addParametricCallback( t 0.3, follower::pausePathFollowing)
          .addParametricCallback( t 0.9, ()-> Intake.get().setState(Intake.IntakeStates.TO_SHOOTER,  powerMultiplier 0))

          .build();
```

Pentru a putea controla utilizarea sistemelor **pe parcursul traiectoriilor este utilizată funcția addParametricalCallback()**, prin care *se stabilește de la ce procent din path trebuie să pornească intake-ul sau alte sisteme.*

4.10 TEHNOLOGII FOLOSITE

Android Studio

Android Studio este mediul oficial de dezvoltare (IDE – Integrated Development Environment) pentru aplicațiile Android. Acesta oferă un spațiu complet în care programatorii pot **scrie, organiza, testa și gestiona codul sursă** al unui proiect software într-un mod eficient și structurat.

Un IDE reprezintă mai mult decât un simplu editor de text. Android Studio integrează într-o singură aplicație toate instrumentele necesare dezvoltării software: editor de cod, sistem de compilare, instrumente de depanare (debugging), analiză a performanței și integrare cu sisteme de control al versiunilor.

Principala sa funcție este de a facilita procesul de dezvoltare prin **automatizarea multor sarcini tehnice**, precum organizarea fișierelor proiectului, compilarea codului, identificarea erorilor sau rularea aplicației pe dispozitive reale sau simulate.

Android Studio este construit pe platforma **IntelliJ IDEA**, ceea ce îi oferă un sistem avansat de analiză a codului. Acesta poate detecta erori de sintaxă, probleme logice sau incompatibilități înainte ca programul să fie executat. În plus, oferă sugestii inteligente de completare a codului (code completion), ceea ce accelerează procesul de programare și reduce probabilitatea apariției erorilor.

Un alt rol important al mediului Android Studio este organizarea structurii proiectului. Codul sursă, bibliotecile externe, resursele și configurațiile sunt aranjate într-o structură clară, ceea ce permite gestionarea eficientă a proiectelor software mari.

În plus, Android Studio include instrumente de **debugging**, care permit analizarea comportamentului programului în timpul execuției. Prin aceste instrumente, dezvoltatorii pot observa valorile variabilelor, pot urmări fluxul execuției și pot identifica rapid eventualele erori din cod.

Prin toate aceste funcționalități, Android Studio contribuie la crearea unui mediu de lucru organizat, în care dezvoltarea software devine mai rapidă, mai sigură și mai eficientă.

Git și controlul versiunilor

Git este un sistem de control al versiunilor distribuit, utilizat pentru gestionarea modificărilor aduse codului sursă într-un proiect software. **Rolul său principal este de a urmări evoluția codului în timp** și de a permite revenirea la versiuni anterioare atunci când este necesar.

În dezvoltarea software, codul este modificat constant. Fără un sistem de control al versiunilor, gestionarea acestor schimbări ar deveni rapid dificilă și predispusă la erori. Git **rezolvă această problemă prin salvarea fiecărei modificări sub forma unei versiuni documentate.**

Fiecare set de modificări este înregistrat într-un **commit**, care reprezintă un punct stabil în evoluția proiectului. Acest mecanism permite păstrarea unui istoric complet al modificărilor, oferind posibilitatea de a analiza sau restaura orice etapă anterioară a dezvoltării.
Un alt concept important în Git este **ramificarea (branching)**. Aceasta permite dezvoltarea unor funcționalități noi sau testarea unor idei fără a afecta versiunea principală a proiectului. Ramurile pot fi ulterior combinate prin procesul de merge, integrând modificările într-o versiune stabilă.
Prin acest sistem, **Git oferă o metodă sigură și organizată de gestionare a evoluției codului, prevenind pierderea informațiilor și facilitând colaborarea între dezvoltatori.**

Integrarea Git în Android Studio

Android Studio include suport nativ pentru Git, ceea ce înseamnă că gestionarea versiunilor poate fi realizată direct din mediul de dezvoltare, fără utilizarea unor instrumente externe.
Această integrare permite realizarea operațiilor esențiale de control al versiunilor într-un mod simplificat. Printre aceste operații se numără urmărirea modificărilor în fișiere, realizarea commit-urilor, gestionarea ramurilor și sincronizarea codului cu un depozit central.
Un avantaj important al acestei integrări este **vizualizarea clară a modificărilor din cod. Android Studio evidențiază diferențele dintre versiunea actuală și cea anterioară, permițând dezvoltatorului să observe exact ce linii de cod au fost modificate**. Acest lucru facilitează revizuirea codului și reduce riscul introducerii unor erori.
De asemenea, integrarea cu Git permite organizarea eficientă a procesului de dezvoltare. Modificările pot fi grupate logic în commit-uri, iar istoricul proiectului rămâne bine documentat. În cazul apariției unei probleme, dezvoltatorul poate reveni rapid la o versiune stabilă.

Android Studio și Git formează împreună un sistem complet pentru dezvoltarea și gestionarea proiectelor software.
Android Studio oferă instrumentele necesare pentru scrierea, analizarea și testarea codului, în timp ce Git asigură controlul asupra evoluției acestuia în timp. Integrarea dintre cele două tehnologii permite **organizarea eficientă a procesului de dezvoltare, menținerea unui istoric clar al modificărilor și gestionarea sigură a proiectelor software complexe.**
Prin utilizarea acestor instrumente, procesul de dezvoltare devine mai structurat, mai sigur și mai eficient, facilitând realizarea unor aplicații robuste și bine organizate.

Principiile programării orientate pe obiecte (OOP) în Java

Programarea orientată pe obiecte este un mod de organizare a software-ului în care aplicațiile sunt construite din **obiecte** care modelează entități sau concepte dintr-un sistem. Fiecare obiect conține **date** și **operații** care pot acționa asupra acelor date.

În loc ca un program să fie structurat ca o succesiune lungă de instrucțiuni, el este organizat ca un **ansamblu de componente care colaborează între ele.** Această abordare permite dezvoltarea unor programe mai ușor de înțeles, de extins și de întreținut.

La baza programării orientate pe obiecte se află patru principii fundamentale: **încapsularea, moștenirea, polimorfismul și abstracția.**

1. Încapsularea (Encapsulation)

Încapsularea reprezintă principiul prin care **datele interne ale unui obiect sunt protejate și gestionate prin intermediul unor metode controlate.**

Ideea centrală este că structura internă a unui obiect nu trebuie accesată sau modificată direct din exterior. În schimb, obiectul oferă un set de metode prin care alte componente ale programului pot interacționa cu el.

Prin acest mecanism, detaliile de implementare sunt ascunse, iar accesul la date este controlat. Astfel se **previne modificarea accidentală a stării interne a obiectului și se menține coerența sistemului.**

Încapsularea contribuie la:

- protejarea datelor
- menținerea integrității logicii interne
- reducerea dependențelor dintre componentele programului

2. Moștenirea (Inheritance)

Moștenirea este mecanismul **prin care o clasă poate prelua proprietățile și comportamentele unei alte clase.**

Clasa care oferă aceste caracteristici se numește clasă părinte, iar clasa care le preia se numește clasă derivată sau clasă copil.

Prin moștenire, o clasă nouă poate reutiliza funcționalități deja existente și poate extinde comportamentul acestora prin adăugarea de metode sau proprietăți noi.

Acest principiu permite organizarea claselor într-o ierarhie logică, în care conceptele generale sunt definite la nivel superior, iar cele mai **specifice sunt definite în clasele derivate.**

Beneficiile moștenirii includ:

- reutilizarea codului
- structurarea clară a relațiilor dintre clase
- reducerea duplicării funcționalităților

3. Polimorfismul (Polymorphism)

Polimorfismul **reprezintă capacitatea unui program de a trata obiecte diferite într-un mod uniform**, permițând totodată fiecărui obiect să își manifeste comportamentul propriu.
Conceptul se bazează pe ideea că aceeași operație poate avea implementări diferite, în funcție de tipul obiectului asupra căruia este aplicată.
Prin polimorfism, **un program poate utiliza aceeași metodă sau aceeași interfață pentru mai multe tipuri de obiecte**, iar fiecare dintre ele va executa propria versiune a acelei operații.
Acest mecanism face posibilă dezvoltarea unor sisteme software flexibile, în care noi tipuri de obiecte pot fi introduse fără modificarea majoră a codului existent.

4. Abstracția (Abstraction)

Abstracția **reprezintă procesul prin care sunt evidențiate caracteristicile esențiale ale unui obiect, în timp ce detaliile de implementare sunt ascunse.**
Scopul abstracției este simplificarea modului în care programatorii interacționează cu un sistem complex. În loc să fie expuse toate mecanismele interne, este oferită doar o interfață clară și simplă, care permite utilizarea funcționalității fără a cunoaște implementarea exactă.
Prin abstracție, un obiect este descris prin ceea ce face, nu prin modul în care este implementat.
Acest principiu contribuie la:

- reducerea complexității sistemelor software
- separarea clară între utilizare și implementare
- dezvoltarea unor arhitecturi software mai ușor de extins

TESTARE SI CALIBRARE

Pentru a eficientiza testarea sistemelor, după implementarea efectivă folosim **telemetria**. Astfel, observăm comportamentul real al robotului în timpul acționării acestora. În plus, aceasta mai este folosită și pentru a determina experimental diferite constante.
Pentru calibrarea autonomiilor, este implementată clasa **PoseFinder**, care comunică în timp real poziția reală a robotului pe teren, pornind de la o poziție fixă.

```
telemetry.addData( caption: "ll mt2", LimelightHandler.get().getUseMT2());
if(llPose!=null)
    telemetry.addData( caption: "ll pose",  format: "x: %f, y: %f, h: %f", llPose.getX(DistanceUnit.METER), llPose.getY(DistanceUnit.METER), llPose.getHeading(AngleUnit.DEGREES));
else
    telemetry.addLine( lineCaption: "ll pose null");
telemetry.addData( caption: "odometry active", odometryActive);
telemetry.addData( caption: "odometry primary", usePrimaryOdometry);
telemetry.addData( caption: "odometry pose",  format: "x: %f, y: %f, h: %f", odometryPose.getX(DistanceUnit.METER), odometryPose.getY(DistanceUnit.METER), odometryPose.getHeading(AngleUnit.DEGREES));
telemetry.addData( caption: "estimated pose",  format: "x: %f, y: %f, h: %f", estimatedPose.getX(DistanceUnit.METER), estimatedPose.getY(DistanceUnit.METER), estimatedPose.getHeading(AngleUnit.DEGREES));
telemetry.addData( caption: "distance from goal", getDistanceFromGoal(DistanceUnit.METER));
telemetry.addData( caption: "angle to goal", getHeadingToGoal(AngleUnit.DEGREES));
telemetry.addData( caption: "angle to goal error", getHeadingToGoalError(AngleUnit.DEGREES));
}
```

CAPITOLUL 6

OFF-SEASON

6.1 REORGANIZAREA ECHIPEI

6.2 WORKSHOPURI

6.3 S.P.L

6.1 REORGANIZAREA ECHIPEI

În urma **Regionalei Sud**, echipa noastră a trecut printr-un proces amplu de reorganizare și dezvoltare, cu scopul de a ne consolida structura și de a ne pregăti mai bine pentru următorul sezon. În această perioadă **am lucrat și la o reîmprospătare a identității echipei, refăcând emblema, materialele promoționale și imaginea generală**. Ne-am dorit ca aceste schimbări să reflecte mai bine valorile noastre: inovația, munca în echipă și progresul.

Pe parcursul off-season-ului, echipa a parcurs un proces de reîmprospătare a identității vizuale, cu scopul de a obține o **imagine unitară, coerentă și profesionistă**, aliniată valorilor promovate de FIRST. Această identitate este utilizată în toate materialele oficiale ale echipei: documentație tehnică, materiale promoționale, postări online, evenimente și competiții.

CULORILE OFICIALE ALE ECHIPEI

Identitatea cromatică a echipei este bazată pe două culori principale, alese pentru a reflecta profesionalismul, inovația și claritatea vizuală:

Mov – cod hex: #8C52FF

Negru – cod hex: #0C0C0C

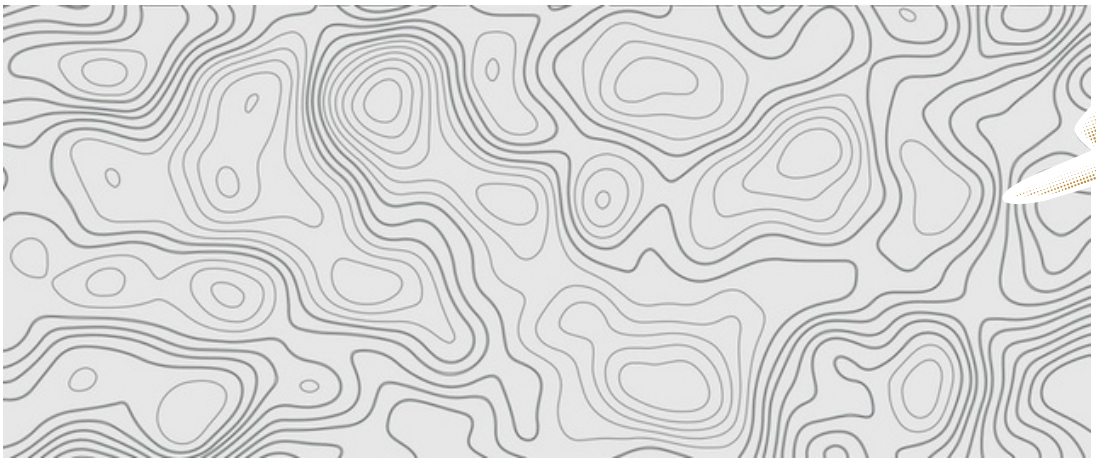
Logo-ul echipei

Logo-ul ViCyber #21476 a fost refăcut pentru a reflecta mai bine direcția actuală a echipei. Acesta reprezintă combinația dintre tehnologie, identitate digitală și spirit creativ.

Elemente vizuale

Identitatea vizuală a echipei include elemente grafice recurente, precum linii geometrice și forme simple, inspirate din domeniul tehnic.

Aceste elemente sunt utilizate pentru a crea continuitate vizuală între materiale și pentru a evidenția caracterul tehnic al echipei.



Scopul identității echipei

Prin această identitate vizuală, ne dorim să reflectăm valorile fundamentale ale echipei ViCyber :
inovația, munca în echipă, profesionalismul și progresul continuu.

O identitate clar definită contribuie la o mai bună reprezentare a echipei în comunitatea FTC și la consolidarea imaginii noastre în fața partenerilor și sponsorilor.

Pe parcursul perioadei de pregătire, am organizat mai multe întâlniri interne în cadrul echipei, în care am discutat obiectivele pentru sezonul următor, strategiile de dezvoltare și planificarea activităților, atât din punct de vedere tehnic, cât și organizațional. **Aceste întâlniri au fost esențiale pentru stabilirea priorităților, alocarea eficientă a resurselor și clarificarea responsabilităților fiecărui membru.**

În plus, am analizat progresul proiectelor anterioare, am identificat punctele forte și ariile ce necesitau îmbunătățiri și am planificat modul în care echipa va aborda provocările viitoare. Toți membrii au fost implicați activ în îndeplinirea rolurilor lor, contribuind la **crearea unui mediu de lucru coeziv, organizat și orientat spre rezultate.**

De asemenea, am început o nouă rundă de recrutări în cadrul liceului, pentru a aduce în echipă elevi pasionați de tehnologie și robotică. După procesul de selecție, fiecare departament a organizat **workshopuri introductive**, menite să ofere membrilor, în special celor noi, o bază solidă de cunoștințe.

Aceste sesiuni au acoperit atât noțiuni de bază despre ***proiectarea, construcția și programarea unui robot eficient, cât și aspecte legate de componenta non-tehnică a proiectului, subliniind importanța acesteia în dezvoltarea și organizarea echipei*.**

6.2 WORKSHOPURI

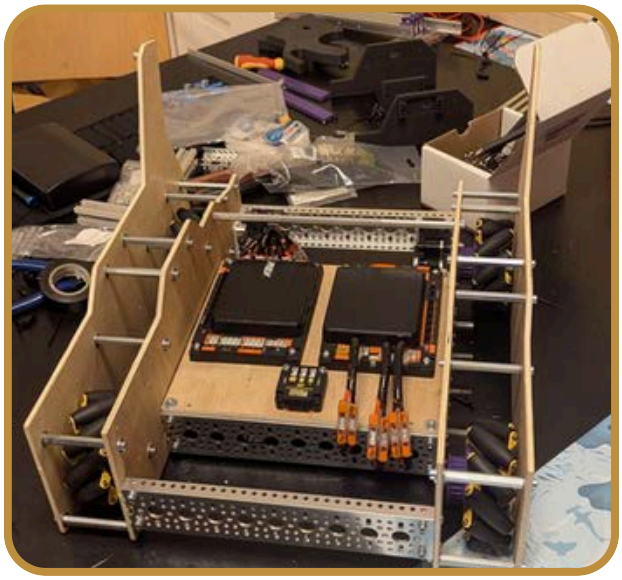
Echipa ViCyber #21476 este organizată în trei departamente principale: Hardware, Software și Marketing, fiecare contribuind activ la funcționarea eficientă a echipei și la atingerea obiectivelor stabilite. Această structură permite o bună organizare a activităților, o comunicare eficientă și o distribuire clară a responsabilităților.

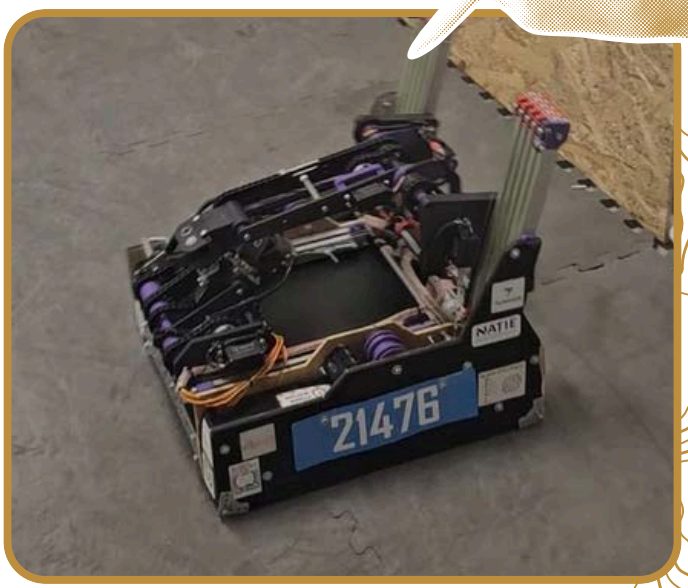
ACTIVITATEA DEPARTAMENTULUI HARDWARE

Departamentul Hardware a desfășurat workshop-uri interne cu scopul de a forma o bază solidă de cunoștințe inginerești în rândul membrilor echipei. Activitățile au inclus sesiuni de imprimare 3D, în cadrul cărora s-a urmărit înțelegerea funcționării imprimantei 3D, alegerea parametrilor optimi de printare și evaluarea calității pieselor obținute. De asemenea, au fost organizate workshop-uri de proiectare asistată de calculator (CAD) folosind platformele Onshape și Fusion, unde membrii au învățat logica modelării pieselor, realizarea asamblărilor și respectarea toleranțelor mecanice. Cunoștințele dobândite au fost aplicate practic prin proiectarea și asamblarea unui robot de tip „media”, utilizat pentru activități de promovare și evenimente organizate de departamentul Marketing. Acest proces a permis testarea designului, identificarea problemelor mecanice și implementarea soluțiilor de îmbunătățire.

REZULTATE ȘI IMPACT

Workshop-urile au creat un mediu de învățare bazat pe gândire logică, matematici exacte și principii inginerești, contribuind la dezvoltarea abilităților tehnice, a colaborării în echipă și a procesului de luare a deciziilor inginerești, elemente esențiale documentate în Engineering Notebook.





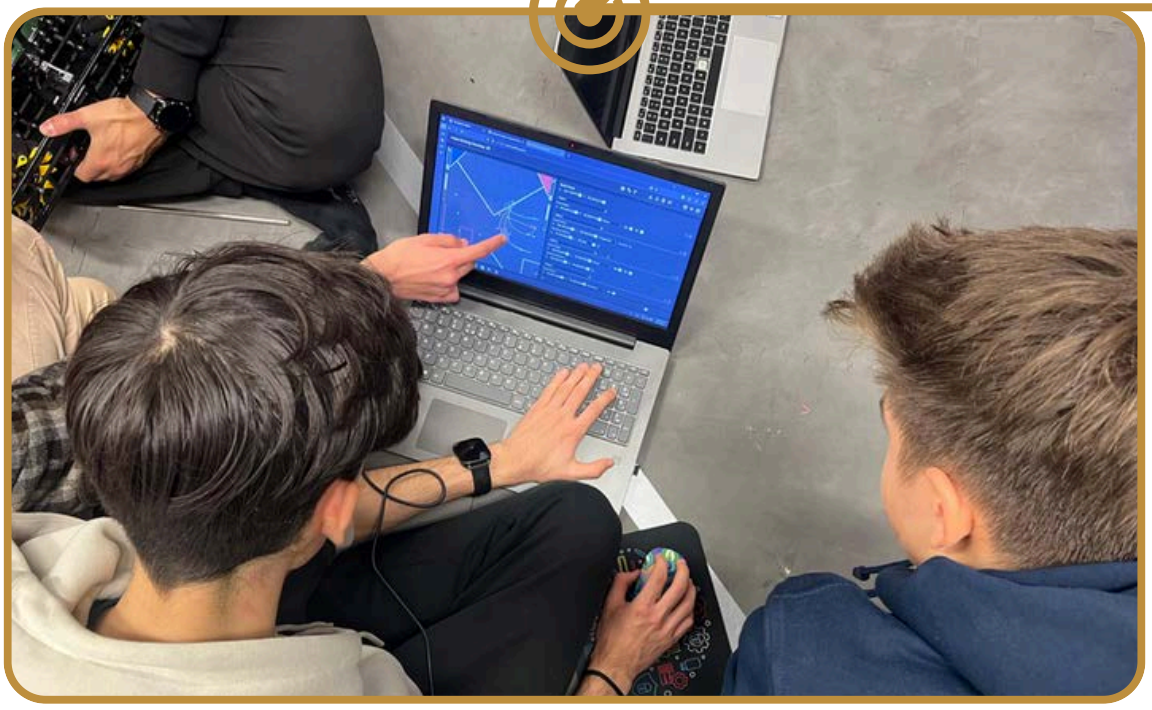
ACTIVITATEA DEPARTAMENTULUI SOFTWARE

Departamentul Software a desfășurat workshop-uri interne cu scopul de a crea o bază solidă de cunoștințe în domeniul programării și al gândirii algoritmice în rândul membrilor echipei. Activitățile au fost concepute gradual, pentru a facilita înțelegerea conceptelor fundamentale și aplicarea lor practică.

Workshop-urile au inclus inițierea în programarea în limbajul Java, utilizând Android Studio și FTC SDK (FIRST Tech Challenge Software Development Kit), precum și familiarizarea cu mediul de lucru și structura unui proiect FTC. Membrii au învățat noțiuni esențiale precum variabile, condiții, bucle, funcții, clase și obiecte, toate explicate într-un mod logic și accesibil.

Cunoștințele dobândite au fost aplicate practic prin programarea robotului de tip „media” realizat de Departamentul Hardware, unde membrii au înțeles modul în care software-ul interacționează cu componentele hardware (motoare, senzori, controllere), precum și importanța testării și depanării codului.

Un accent deosebit a fost pus pe înțelegerea logicii din spatele programării, a modului în care comenzile sunt executate de robot și a modului în care structura codului influențează performanța și fiabilitatea acestuia.



ACTIVITATEA DEPARTAMENTULUI MARKETING

Departamentul Marketing a desfășurat workshop-uri interne cu scopul de a dezvolta competențele de comunicare, design și organizare ale membrilor echipei, contribuind la formarea unei imagini coerente și profesioniste a echipei ViCyber #21476.

Activitățile au inclus sesiuni de lucru dedicate utilizării programelor de editare grafică și conținut vizual, precum Canva și pachetul Adobe (Adobe Photoshop, Illustrator), prin intermediul cărora membrii au învățat principiile de bază ale designului, editării imaginilor, realizării materialelor promoționale și structurării conținutului vizual.

În cadrul workshop-urilor, participanții au înțeles procesul din spatele organizării evenimentelor, de la planificare și promovare până la execuție, precum și volumul de muncă necesar pentru realizarea acestora într-un mod eficient și atractiv. De asemenea, aceștia au învățat despre identitatea echipei, importanța unei imagini unitare, valorile promovate de ViCyber și modul în care acestea sunt transpuse în materiale vizuale și activități de outreach.

Aceste activități au contribuit la dezvoltarea creativității, spiritului de inițiativă și lucrului în echipă, sprijinind comunicarea eficientă a proiectelor tehnice și a valorilor FIRST Tech Challenge către comunitate.

6.3 S.P.L

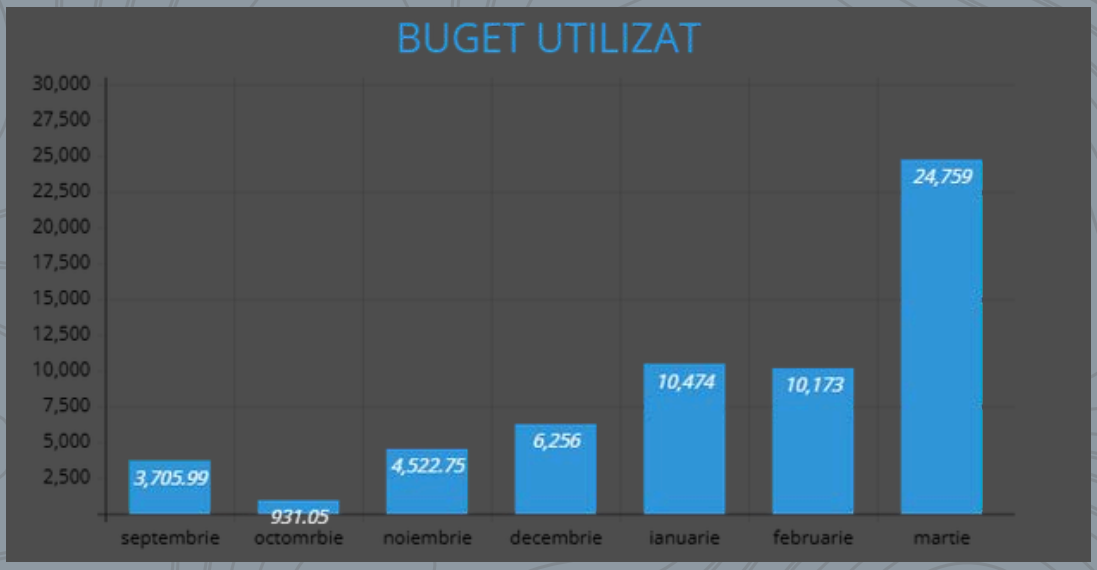
În perioada off-season, echipa noastră a participat la competiția South Performance League, organizată de echipele Clockworks #19075 și Andromeda #20691, desfășurată în zilele de 12–13 august la International School of Bucharest. Evenimentul a oferit un cadru controlat și competitiv, destinat testării soluțiilor tehnice, strategiilor de joc și mecanismelor dezvoltate în afara sezonului competițional oficial.

Competiția ne-a permis să evaluăm performanța robotului în condiții reale de meci, să testăm fiabilitatea sistemelor mecanice și electronice și să analizăm eficiența codului de autonomie și teleoperare. Interacțiunea cu alte echipe FTC ne-a oferit oportunitatea de a observa și compara abordări diferite de rezolvare a problemelor tehnice, contribuind la îmbunătățirea procesului nostru de proiectare și programare. Pe parcursul meciurilor, am reușit să concurăm cu echipe cu un nivel tehnic ridicat, obținând rezultate bune și validând soluțiile dezvoltate anterior. Echipa a ajuns în play-off, unde a avut rolul de căpitan de alianță, experiență care a contribuit la dezvoltarea abilităților de coordonare tehnică, luare rapidă a deciziilor și adaptare strategică în timp real.

Pentru membrii care s-au alăturat echipei după Regionala Sud, această competiție a reprezentat prima experiență practică într-un cadru FTC, oferindu-le posibilitatea de a înțelege structura unui concurs, fluxul meciurilor și importanța colaborării între departamente în timpul competiției.

Participarea la South Performance League a avut un impact semnificativ asupra echipei, contribuind la identificarea punctelor forte și a aspectelor ce necesitau îmbunătățiri, atât din punct de vedere tehnic, cât și organizațional. Această experiență ne-a ajutat să ne optimizăm metodele de lucru și să ne pregătim mai eficient pentru provocările viitoare ale sezonului competițional FTC.

CAPITOLUL 7
MANAGEMENT & SUSTAINABILITY



,

7.1 FINANTE

7.2 SPONSORI

7.1 FINANTE

Datorită sprijinului constant oferit de **sponsorii noștri**, în acest an am beneficiat de o creștere semnificativă a bugetului echipei, de la 3000 de euro in sezonul 2024-2025 la 8.600 de euro in sezonul actual. Acest lucru ne-a oferit oportunitatea de a ne extinde considerabil activitățile și de a ne dezvolta proiectul la un nivel pe care nu il credeam posibil inainte. Prin fondurile suplimentare, am reușit să participăm la un număr mai mare de competiții, unde am avut șansa de a ne testa abilitățile, de a acumula experiență practică și de a ne perfecționa lucrul in echipă.

În plus, bugetul mărit ne-a permis să achiziționăm piese noi pentru robotul realizat în acest an, ceea ce a contribuit direct la îmbunătățirea funcționalității, fiabilității și performanței acestuia. Am investit, de asemenea, în materiale promoționale, pe care le-am cumpărat și personalizat pentru a ne consolida identitatea vizuală și pentru a face proiectul nostru mai vizibil în comunitate.

Totodată, am avut posibilitatea să organizăm un număr mai mare de evenimente în cadrul proiectului FIRST, prin care am promovat educația STEAM, munca în echipă și inovația în rândul elevilor. Pentru a ne asigura că toate aceste activități se desfășoară în condiții optime, a fost necesar să dezvoltăm strategii clare de gestionare și utilizare eficientă a bugetului, astfel încât acesta să acopere toate cheltuielile aferente implementării proiectului nostru și să ne permită să ne atingem obiectivele propuse.

1. COSTURI DE BAZA

Componentele robotului:

O parte semnificativă a bugetului a fost direcționată către reînnoirea și achiziționarea componentelor necesare construirii robotului. Am urmărit să investim în piese de calitate, durabile, care pot fi reutilizate și în sezoanele viitoare, pentru a asigura atât performanță ridicată, cât și sustenabilitate pe termen lung. Această strategie ne permite să optimizăm costurile în timp și să păstrăm un standard tehnic ridicat la fiecare competiție.

Transportul echipei și al robotului:

Fondurile alocate transportului ne-au permis să participăm la competiții organizate în afara orașului nostru. Acest aspect este esențial pentru dezvoltarea echipei, deoarece ne oferă oportunitatea de a interacționa cu alte echipe, de a învăța din experiențele acestora și de a ne testa abilitățile într-un cadru competitiv real.

Amenajarea sediului echipei:

2. COSTURI DE IMAGINE

Materiale promoționale:

Materialele promoționale reprezintă imaginea echipei noastre și joacă un rol important în creșterea vizibilității. Acestea ne ajută să ne facem cunoscuți în cadrul comunității STEAM, dar și să construim relații cu parteneri, mentori și alți susținători ai educației tehnologice.

Plata domeniului site-ului echipei:

Deținerea și întreținerea site-ului oficial al echipei reprezintă o componentă importantă în procesul de promovare. Prin intermediul acestuia, reușim să prezentăm activitatea noastră, realizările și valorile noastre, facilitând atragerea potențialilor sponsori și parteneri.

3. COSTURI AFERENTE ORGANIZARII DE EVENIMENTE

Evenimentele organizate de echipa noastră ne oferă oportunitatea de a ne conecta cu alți membri ai comunității STEAM și chiar din afara acesteia. Prin participarea și implicarea activă în organizarea acestor evenimente, reușim să promovăm valorile FIRST unui public cât mai divers, contribuind la creșterea vizibilității și la o mai bună înțelegere a domeniului roboticii.

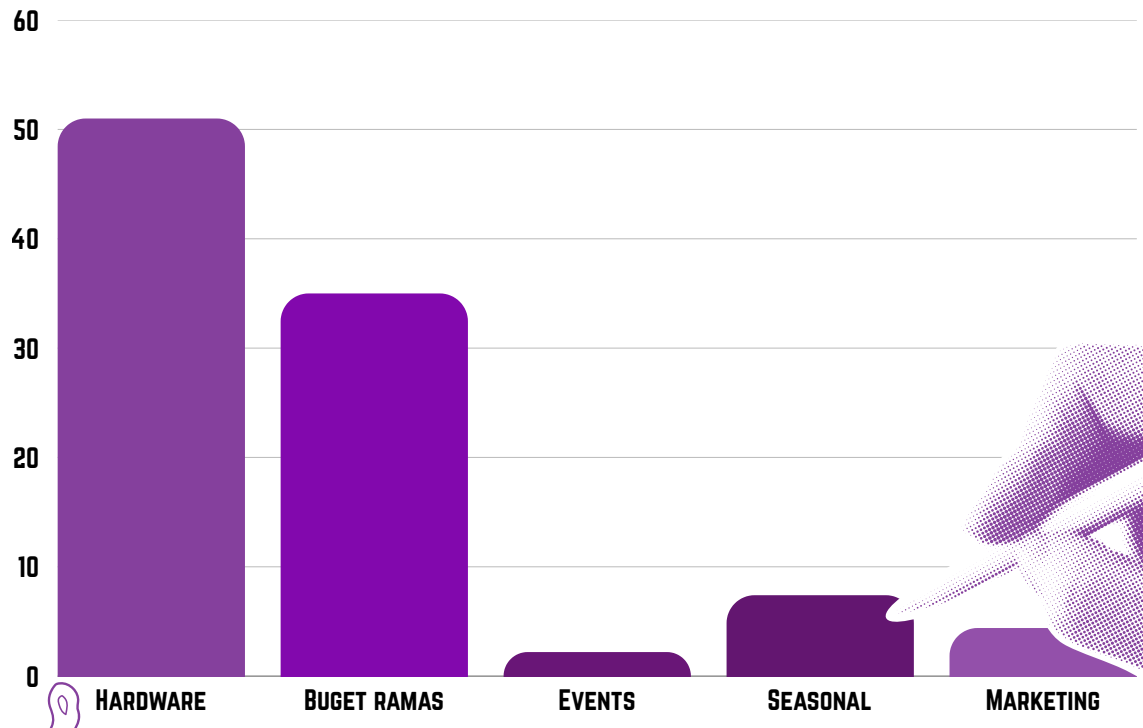
Aceste activități joacă un rol esențial și în inspirarea tinerilor de vârsta noastră să își descopere pasiunea pentru tehnologie și inginerie. Pentru desfășurarea evenimentelor avem nevoie atât de materiale promoționale, cât și de locații adecvate, care să ne permită să oferim o imagine profesionistă.

În cadrul echipei, departamentul de marketing are un rol esențial în conceperea, realizarea și procurarea materialelor promoționale. Tot acest departament se ocupă și de planificarea și organizarea evenimentelor, în colaborare cu partenerii noștri, asigurând coerența imaginii publice și profesionalismul activităților desfășurate.

ALOCAREA FONDURILOR

În sezonul competițional 2025–2026, cea mai mare parte a bugetului a fost direcționată către departamentul de hardware, având ca obiectiv principal modernizarea și aducerea echipamentelor echipei la cele mai înalte standarde tehnologice. Această investiție substanțială a fost esențială pentru creșterea performanței generale, oferindu-ne posibilitatea de a concura în condiții optime și de a face față cerințelor competițiilor de cel mai înalt nivel. Prin actualizarea constantă a echipamentelor și utilizarea unor soluții tehnice de ultimă generație, echipa beneficiază de un avantaj competitiv clar și de o bază solidă pentru obținerea rezultatelor dorite.

Pe locul al doilea în clasamentul cheltuielilor din acest sezon se situează costurile aferente desfășurării activității departamentului de marketing. Acestea includ atât cheltuielile de promovare, cât și costurile de imagine ale echipei, având un rol strategic în consolidarea vizibilității și a identității brandului. Investițiile în marketing au urmărit creșterea notorietății echipei, atragerea de noi parteneri și susținători, precum și menținerea unei relații constante și eficiente cu publicul. Astfel, activitatea departamentului de marketing contribuie semnificativ nu doar la imaginea echipei, ci și la dezvoltarea sa pe termen mediu și lung.



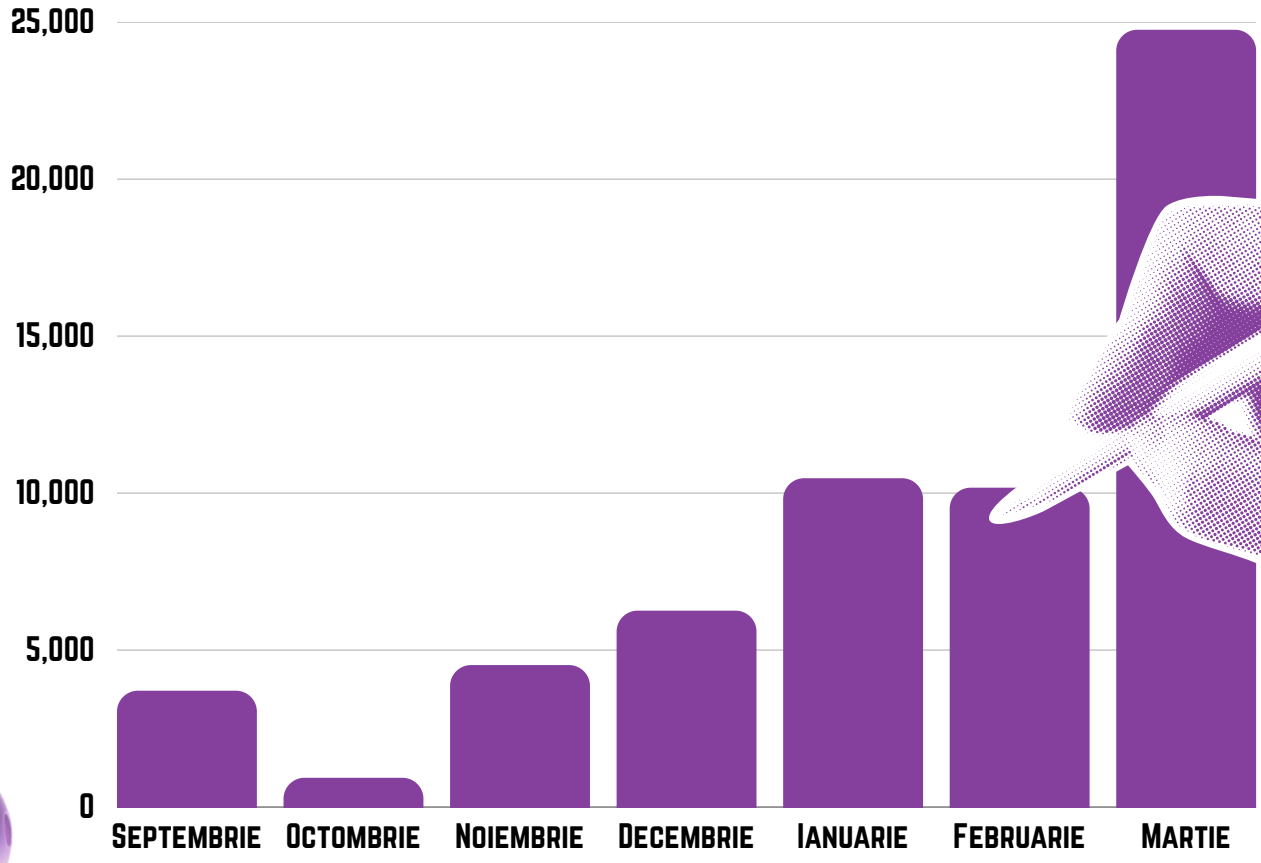
Echipa începe sezonul DECODE cu fonduri de 1.200 de euro. Prin aplicarea unei strategii de sponsorizare atent construite și aliniate nevoilor noastre actuale, am reușit să obținem rezultate notabile, generând un plus de 6.800 de euro prin pachete de sponsorizare.

O parte considerabilă din buget a fost direcționată către dezvoltarea robotului, prin achiziționarea de piese complet noi, din cauza faptului că cele folosite anterior au ajuns la finalul ciclului de viață. **De asemenea, necesitățile departamentului de marketing** au avut un buget alocat într-o proporție mai mică. Aceasta a fost stabilită în funcție de obiectivele noastre principale: optimizarea robotului pentru a ne apropia cât mai mult de versiunea finală dorită înainte de etapa regională și achiziționarea de materiale promoționale noi, astfel îmbunătățind relația cu partenerii, sponsorii și comunitatea locală.

Bugetul este alocat pentru fiecare departament, în funcție de urgența obiectivelor. Așadar, cu un buget inițial de 1.200 de euro am alocat banii astfel: 840 de euro pentru departamentul de mecanică (70%), 240 de euro pentru marketing (20%) și 120 de euro pentru cheltuieli diverse.

Ulterior, după ce am primit două sponsorizări în valoare de 2.000 de euro, respectiv 5.000 de euro la scurt timp după începutul sezonului, bugetul și proporțiile au fost modificate, încât noul flux de bani permitea departamentului de marketing să conceapă un stand nou. Astfel, alocarea bugetului de 8.000 de euro devine: 4.000 de euro pentru **departamentul hardware** (50%), 1.500 de euro pentru **departamentul marketing** (20%) și 2.500 de euro pentru cheltuieli neprevăzute și sezoane viitoare (30%). Decizia de a modifica procentajele alocate fiecărui departament a avut loc după o analiză profundă a necesarului și a bugetului necesar real. Am observat că, în medie, costul din fiecare sezon al unui robot este în jur de 3.000 de euro, iar cei 1.000 de euro sunt adăugați pentru a înlocui piese vechi, considerând vechimea acestora.

IMPARTIREA COSTURILOR PE PARCURSUL SEZONULUI.



Bugetul echipei a crescut organic de-a lungul sezonului, pe măsură ce ne-am dezvoltat tehnic și am reușit să atragem noi surse de finanțare. Fiecare etapă a adus cu sine nevoi noi atat pentru **performanța competițională**, cât și pentru modul în care ne prezentăm ca echipă.

La startul sezonului, am pus bazele. Am cumpărat terenul oficial de joc pentru antrenamente și **testarea strategiilor**, alături de prima serie de piese mecanice și componente tehnice. Cu acestea am construit primul nostru robot al sezonului cel cu care am intrat în competiție la etapa **League Meet-urilor**. Prioritatea noastră în această perioadă a fost să înțelegem jocul în profunzime și să construim o fundație tehnică solidă.

7.2 SPONSORI !

ASUS este o companie globală de tehnologie cunoscută pentru laptopuri, componente hardware și alte dispozitive performante. Produsele sale sunt apreciate pentru calitate, design și performanță, fiind utilizate de oameni din întreaga lume. **ASUS** investește constant în inovație și dezvoltarea tehnologiei. Sprijinul oferit echipei noastre ne ajută să continuăm să explorăm și să învățăm prin tehnologie.

Tinmar Energy este una dintre cele mai mari companii private din sectorul energetic din România. Compania furnizează energie electrică și gaze naturale pentru numeroși clienți din țară și investește în **dezvoltarea energiei regenerabile**. Prin activitatea sa, contribuie la modernizarea sectorului energetic și la dezvoltarea unor soluții sustenabile.

Popp & Asociații este o companie românească de inginerie și proiectare specializată în domeniul construcțiilor și infrastructurii. Echipa lor a **contribuit la numeroase proiecte importante din România**, oferind expertiză tehnică și soluții inovatoare. Experiența și profesionalismul companiei au un impact important în dezvoltarea mediului construit.

Orange este unul dintre cei mai mari operatori de telecomunicații din lume și un lider al pieței din România. Compania oferă servicii de telefonie mobilă, internet și soluții digitale pentru milioane de utilizatori. **Orange** investește constant în tehnologii moderne și în dezvoltarea infrastructurii digitale. Susținerea lor ne ajută să continuăm activitățile și proiectele echipei.

ParkLake Shopping Center este unul dintre cele mai moderne centre comerciale din București. **Mall-ul** oferă o varietate de magazine, restaurante și spații de divertisment, fiind un loc popular pentru petrecerea timpului liber. Conceptul său este inspirat de natură și pune accent pe experiența comunității. Susținerea lor contribuie la dezvoltarea proiectelor echipei noastre.

Horizon City este un proiect rezidențial modern dezvoltat în zona Pipera din București. Complexul pune accent pe confort, spații verzi și facilități pentru comunitate, oferind un stil de viață echilibrat. Proiectul este conceput pentru a crea o comunitate urbană modernă și prietenoasă. **Susținerea lor** ne ajută să continuăm activitatea echipei și proiectele educaționale

deFLORIAN

deFlorian este un brand de lenjerii de pat realizate din materiale naturale, cunoscut pentru produsele sale confortabile și durabile. Lenjeriile sunt fabricate din bumbac de calitate și sunt concepute pentru a oferi un somn plăcut și un bun confort termic pe tot parcursul anului. **Produsele sunt create cu atenție la detalii** și sunt realizate în loturi mici, de o familie cu tradiție în domeniul textilelor.

F64 devoted to your passion

F64 este cel mai mare retailer de echipamente foto și video din România. Compania oferă produse și servicii pentru fotografi începători și profesioniști, dar și cursuri și evenimente dedicate comunității creative. Prin activitatea sa, **F64** încurajează creativitatea și dezvoltarea pasiunii pentru fotografie. Sprijinul lor contribuie la promovarea proiectelor echipei noastre.

CAPITOLUL 8
OUTREACH & IMPACT

8.1 EVENIMENTE

8.2 ÎNCADRAREA EVENIMENTELOR ÎN VALORILE FIRST

8.3 CONCURSURI

8.1 EVENIMENTE

Evenimentele ne oferă oportunitatea de a ne conecta cu alți membri ai comunității STEAM și chiar dincolo de aceasta. Prin participarea și organizarea lor, reușim să promovăm valorile FIRST unui grup mult mai divers de persoane, contribuind la creșterea vizibilității și înțelegerii domeniului roboticii. În același timp, aceste activități ne permit să îi inspirăm pe tinerii de vârsta noastră să își descopere pasiunea pentru tehnologie.
Folosim evenimentele atât în scop didactic, pentru a transmite cunoștințe și a demonstra principiile STEM, cât și în mod creativ, transformând fiecare ocazie într-o experiență interactivă și captivantă. Ele reprezintă un cadru ideal în care ne putem exprima liber imaginația și dorința de a contribui la dezvoltarea unei comunități mai unite și mai pasionate de robotică.

Pe lângă dezvoltarea competențelor de comunicare, evenimentele contribuie la formarea unor calități esențiale precum leadershipul, lucrul în echipă, gestionarea timpului și adaptabilitatea. Fiecare activitate – fie că este vorba despre workshop-uri, prezentări publice sau activități sociale – ne ajută să ne depășim limitele, să câștigăm încredere în propriile abilități și să ne formăm o mentalitate orientată spre învățare continuă. În plus, interacțiunea cu alți membri ai comunității ne oferă posibilitatea de a construi relații valoroase, de a ne extinde rețeaua de contacte și de a împărtăși experiențe ce pot deveni surse de inspirație.

8.2 INCADRAREA EVENIMENTELOR IN VALORILE FIRST

IMPACT

VICYBER ECHOES

IMPACT 17 CLASE, 450 PERSOANE

În vederea misiunii noastre de a răspândi pasiunea față de robotică, implicit față de domeniul STEAM, am conceput proiectul **ViCyber Echoes**. Acesta a constat în susținerea unor prezentări în cadrul școlilor, în colaborare cu cadrele didactice.

Le-am vorbit elevilor despre ce înseamnă robotica, pașii construirii unui robot, modul de gândire în programare și despre comunitatea STEAM. De asemenea, am avut și aplicații practice; la copiii din ciclul primar am aplicat tehnica de învățare prin joc: am construit un robot din LEGO, cu care am îndeplinit diverse provocări.

Prin acest proiect am reușit să ajungem în numeroase licee, școli și grădinițe, iar lista de mai jos reflectă doar începutul impactului nostru în comunitate. Dorința noastră este să extindem această inițiativă către cât mai multe instituții de învățământ, pentru a demonstra copiilor și profesorilor că robotica nu este doar o activitate distractivă, ci și un domeniu care contribuie la dezvoltarea pe termen lung.

Ne propunem să arătăm că robotica stimulează gândirea critică, creativitatea, lucrul în echipă și încrederea în sine — abilități esențiale într-o lume aflată în continuă schimbare. Cu fiecare vizită, ne dorim să inspirăm mai mulți elevi să descopere potențialul tehnologiei și să își urmeze curiozitatea.

Pe măsură ce proiectul crește, ne angajăm să construim o rețea educațională solidă, în care tehnologia să devină accesibilă tuturor, iar pasiunea pentru robotică să ajungă în cât mai multe comunități.

SCOALA PARTICULARA“TEDDYBAR” (13–14 OCTOMBRIE 2025)

Evenimentul desfășurat la **Școala „Teddybar”** a reprezentat prima noastră încercare de a pune în practică proiectul **„ViCyber Echoes”**, marcând totodată prima prezentare oficială a acestuia în fața unui public. Fiind la început de drum, membrii echipei au fost cuprinși de emoții firești, generate atât de importanța momentului, cât și de responsabilitatea de a nu dezamăgi mințile tinere care ne urmăreau cu interes și curiozitate.

Deși prezentarea fusese atent exersată în prealabil, trăirile intense și dorința de a transmite informațiile într-un mod clar, captivant și educativ au amplificat emoțiile echipei. Cu toate acestea, **profesionalismul, implicarea și pasiunea pentru proiect au reușit să transforme aceste sentimente într-un avantaj**, contribuind la o atmosferă autentică și interactivă.

Prezentarea s-a desfășurat cu succes, iar reacțiile pozitive ale elevilor au confirmat impactul mesajului transmis. Participanții au fost receptivi, au adresat întrebări și s-au implicat activ, ceea ce ne-a oferit încrederea că proiectul **„ViCyber Echoes”** are potențialul de a educa și inspira. Această primă experiență a reprezentat un pas esențial în dezvoltarea proiectului și ne-a motivat să continuăm, cu și mai multă determinare, organizarea unor astfel de activități educaționale.

SCOALA PARTICULARA „EVRIKA”
(22–23 OCTOMBRIE 2025)

În zilele de **22 și 23 octombrie**, echipa **ViCyber #21476** a desfășurat o activitate de outreach educațional la **Școala Particulară „Evrika”**, având ca obiectiv promovarea interesului pentru robotică și tehnologie în rândul elevilor de vârstă mică. Activitatea a fost realizată în colaborare cu cadrele didactice, care au sprijinit organizarea și desfășurarea sesiunilor educaționale.

Pe parcursul celor două zile, membrii echipei au susținut activități interactive prin care elevii au fost introduși în concepte de bază **legate de robotică**, funcționarea unui robot și importanța lucrului în echipă. Prin demonstrații practice și explicații adaptate nivelului de vârstă, copiii au avut ocazia să înțeleagă noțiuni elementare despre mecanisme, mișcare și coordonare.

Activitatea a pus un accent deosebit pe colaborare, creativitate și curiozitate, valori fundamentale promovate de FIRST Tech Challenge. **Entuziasmul și interesul manifestate** de elevi au evidențiat impactul pozitiv al acestor inițiative asupra dezvoltării timpurii a pasiunii pentru **domeniile STEAM**. Această experiență a confirmat importanța implicării echipei în activități educaționale dedicate elevilor de toate vârstele și a contribuit la consolidarea misiunii noastre de a inspira generațiile viitoare să descopere și să exploreze lumea tehnologiei și a inovației.

COLEGIUL NATIONAL „MATEI BASARAB” (22 OCTOMBRIE 2025)

La data de **22 octombrie**, **echipa ViCyber #21476** a desfășurat o activitate de outreach educațional la **Colegiul Național „Matei Basarab”,** unde am susținut o prezentare dedicată promovării roboticii și a valorilor FIRST Tech Challenge.

Activitatea a fost realizată cu **sprijinul Consiliului Elevilor** și al cadrelor didactice, care au facilitat organizarea și desfășurarea evenimentului.
În cadrul prezentării, am oferit elevilor o introducere în domeniul roboticii competiționale, prezentând structura unei **echipe FTC**, rolurile departamentelor (Hardware, Software și Marketing), precum și impactul pe care competiția îl are asupra dezvoltării abilităților tehnice și non-tehnice ale elevilor.

De asemenea, am discutat despre **valorile FIRST – Gracious Professionalism™**, colaborare, inovație și spirit de echipă – și modul în care acestea sunt aplicate în activitatea noastră. **Evenimentul a avut un caracter interactiv**, elevii fiind încurajați să adreseze întrebări și să participe activ la discuții. Prin această activitate, am urmărit să creștem interesul tinerilor pentru domeniul STEAM și să contribuim la dezvoltarea comunității FTC prin informare, inspirație și implicare directă.

Această experiență **a reprezentat un pas important în consolidarea relației cu comunitatea educațională locală** și în îndeplinirea obiectivului nostru de a promova robotica și educația tehnologică în rândul elevilor.

COLEGIUL NATIONAL „MARIN PREDA” (14 NOIEMBRIE 2025)

La data de **14 noiembrie**, echipa **ViCyber #21476** a desfășurat o activitate de outreach educațional la **Liceul Teoretic „Marin Preda”**, unde a susținut o prezentare dedicată promovării roboticii și a competiției **FIRST Tech Challenge**. Evenimentul a fost realizat cu sprijinul Consiliului Elevilor și al cadrelor didactice, care au contribuit la buna organizare a activității.

În cadrul prezentării, membrii echipei au oferit elevilor o introducere în robotica competițională, explicând structura unei echipe FTC, **rolurile departamentelor** (Hardware, Software și Marketing), precum și etapele principale ale dezvoltării unui robot pe parcursul unui sezon competițional. De asemenea, au fost prezentate valorile FIRST, precum Gracious Professionalism™, colaborarea, inovația și lucrul în echipă, evidențiind importanța acestora în activitatea zilnică a echipei.

Activitatea a avut un caracter interactiv, elevii fiind încurajați să participe activ prin întrebări și discuții deschise. **Prin această inițiativă, echipa a urmărit să stimuleze interesul pentru domeniul STEAM** și să contribuie la extinderea comunității FIRST Tech Challenge, oferind elevilor o perspectivă clară asupra oportunităților educaționale și personale oferite de robotica competițională. Această experiență a consolidat **relația echipei cu mediul educațional** și a susținut obiectivul nostru de a promova educația tehnologică și implicarea activă în comunitatea FTC.

FUN

SCITECH

IMPACT 200+ PERSOANE

Pe data de **4 decembrie 2025**, am avut oportunitatea de a participa activ la organizarea **proiectului SciTech**, un eveniment inițiat de **Organizația Startevo**, care și-a propus să reunească tineri pasionați de tehnologie, inovație și gândire creativă. Proiectul a avut ca scop principal stimularea interesului pentru domeniile STEAM, încurajarea colaborării și dezvoltarea abilităților de rezolvare a problemelor într-un context competitiv și dinamic.

În cadrul acestui eveniment, am fost implicați direct prin amenajarea și gestionarea unui stand propriu, alături de ceilalți colaboratori, unde am interacționat cu participanții, le-am prezentat activitățile propuse și le-am oferit sprijin pe parcursul probelor. De asemenea, am contribuit la organizarea a două probe interactive, special concepute pentru a testa logica, gândirea analitică și capacitatea participanților de a lua decizii rapide sub presiune.

Experiența a fost una valoroasă, atât din punct de vedere organizațional, cât și educațional, contribuind la dezvoltarea competențelor de lucru în echipă, **comunicare și coordonare**, dar și la consolidarea legăturilor dintre tinerii interesați de viitorul tehnologiei și inovației. Prima probă a constat în ansamblarea unui robot simplu din piese de plastic, într-un timp cât mai scurt, iar a **doua probă a constat în testarea abilităților elevilor de a naviga un program de proiectare 3D**.

INCLUSION

ERASMUS+

IMPACT APROX. 200 PERSOANE

Pe **17 noiembrie 2025**, am avut privilegiul de a participa la un eveniment organizat în cadrul proiectului **Erasmus+**, o experiență ce ne-a oferit oportunitatea de a prezenta potențialul și performanțele echipei noastre din cadrul competiției **FIRST Tech Challenge (FTC).** Această întâlnire a reprezentat nu doar un prilej de a evidenția rezultatele obținute, ci și un moment important de conectare cu elevi și profesori dintr-o altă cultură, descoperind perspective noi asupra modului în care **tehnologia și educația STEAM** sunt percepute și aplicate în Europa.

În timpul evenimentului, am susținut o prezentare amplă despre structura și filosofia competiției FTC, explicând modul în care funcționează echipa noastră și cum contribuie fiecare membru la succesul final. Am detaliat organizarea internă în cele trei departamente – Software, Hardware și Marketing – subliniind rolul esențial al fiecăruia în dezvoltarea robotului, **optimizarea strategiilor de joc și asigurarea unei comunicări eficiente atât în interiorul echipei, cât și în relația cu comunitatea.**
Acest format colaborativ le-a arătat participanților că robotica nu înseamnă doar programare sau construcție, ci îmbină creativitatea, ingineria, organizarea și leadershipul.

Dincolo de partea teoretică, **am avut ocazia să dialogăm cu elevii prezenți, să le răspundem la întrebări și să explorăm împreună idei despre viitorul tehnologiilor educaționale**. Entuziasmul lor pentru robotică ne-a inspirat, iar schimbul de idei a evidențiat cât de importantă este colaborarea internațională pentru dezvoltarea unor perspective noi și inovatoare.

TEAMWORK

LEAGUE MEET-UL „SNOWY CIRCUITS”

IMPACT 250 PERSOANE

Competiția **„Snowy Circuits”**, organizată de echipa noastră, în colaborare cu echipa **SPARKTECH #24345**, s-a desfășurat la **Colegiul Național de Informatică „Tudor Vianu”** și a reprezentat o adevărată provocare, atât din punct de vedere al abilităților tehnice, cât și al celor organizatorice.
Fiind prima competiție organizată de echipa noastră, acest eveniment a avut o importanță deosebită pentru noi.

Am depus mult efort în etapa de planificare, iar interesul manifestat de celelalte echipe ne-a onorat profund. Am fost plăcut surprinși de **numărul mare de participanți**, ceea ce ne-a confirmat că ideea competiției a fost bine primită și apreciată în comunitate. Pe parcursul desfășurării concursului, ne-am confruntat cu **diverse provocări tehnice și logistice**, însă am reușit să le gestionăm eficient prin comunicare, cooperare și spirit de echipă. Fiecare obstacol apărut a fost o oportunitate de a învăța și de a ne dezvolta abilitățile, iar implicarea tuturor membrilor a fost esențială pentru succesul evenimentului.

În ciuda dificultăților întâmpinate, am reușit să asigurăm buna desfășurare a competiției, să rezolvăm prompt problemele apărute și să oferim participanților o experiență plăcută și bine organizată. **„Snowy Circuits”** a fost nu doar un test al capacităților noastre organizatorice, ci și o experiență valoroasă care ne-a ajutat să creștem ca echipă și să câștigăm încredere pentru organizarea viitoarelor evenimente.

DEMO-UL "BLOOM SCRIMMAGE"

ORGANIZAT ALATURI DE INFO(1)ROBOTICS SI HEART.ROBOTICS

IMPACT 120 DE PERSOANE

Alături de echipele Info(1)Robotics și Heart.Robotics, am organizat pe data de 1 martie un demo intitulat „Bloom Scrimmage", desfășurat la Colegiul Național de Informatică „Tudor Vianu". Evenimentul a reprezentat o experiență intensă și provocatoare, atât din punct de vedere al competențelor tehnice, cât și al celor organizatorice.

Pe parcursul desfășurării competiției, ne-am confruntat cu diverse situații neprevăzute, de la probleme tehnice până la provocări logistice.Cu toate acestea, printr-o bună comunicare, cooperare și prin implicarea activă a tuturor membrilor echipelor, am reușit să gestionăm eficient fiecare obstacol apărut.

Fiecare dificultate întâlnită a devenit o oportunitate de a învăța lucruri noi, de a ne adapta rapid și de a ne consolida abilitățile de lucru în echipă.

Datorită efortului comun și spiritului de colaborare, competiția s-a desfășurat în condiții bune, iar problemele apărute au fost rezolvate prompt, astfel încât participanții să se bucure de o experiență plăcută și bine organizată.

Atmosfera a fost una energică și plină de entuziasm, iar schimbul de idei și experiențe dintre echipe a contribuit la crearea unui mediu competitiv, dar în același timp prietenos.

„Bloom Scrimmage" nu a fost doar un test al abilităților noastre organizatorice și tehnice, ci și o experiență valoroasă care ne-a ajutat să evoluăm ca echipă. Evenimentul ne-a oferit ocazia de a ne perfecționa modul de organizare, de a învăța din provocările întâlnite și de a câștiga mai multă încredere în capacitatea noastră de a planifica și desfășura evenimente similare în viitor.

DEMO-UL
“DECODE THE FIELD”

ORGANIZAT ALATURI QUANTUM ROBOTICS

IMPACT 130 DE PERSOANE

Pe data de 8 martie, alături de echipa Quantum Robotics, am organizat demo-ul „Decode the Field” la Colegiul Național de Informatică „Tudor Vianu”. Acest eveniment a oferit echipelor participante oportunitatea de a-și testa roboții într-un mediu asemănător celui de competiție și de a exersa strategiile de joc cu o săptămână înainte de etapa națională.

Pe parcursul demo-ului, echipele au putut să își verifice mecanismele, să observe ce îmbunătățiri mai sunt necesare și să se familiarizeze mai bine cu terenul de joc. În același timp, evenimentul a fost o ocazie excelentă pentru colaborare, schimb de idei și consolidarea relațiilor dintre echipe.

Atmosfera a fost una energică și plină de entuziasm, iar participanții au profitat de această experiență pentru a-și perfecționa roboții și strategiile. „Decode the Field” nu a fost doar un simplu antrenament înainte de națională, ci și o oportunitate valoroasă de pregătire și de învățare pentru toate echipele implicate.

FALL IN ROBOTIX

IMPACT 500 DE PERSOANE

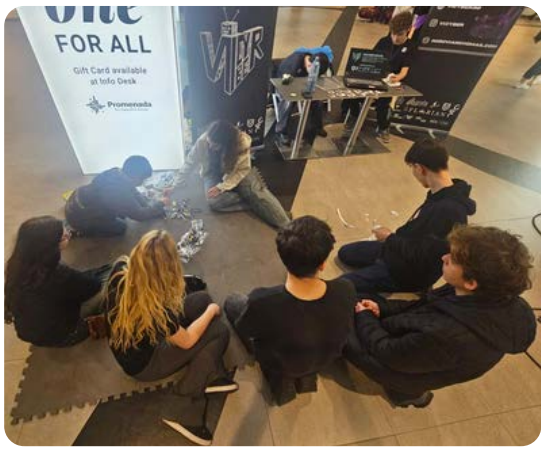
„Fall in Robotix” este evenimentul prin care echipa noastră a reușit să aducă lumea captivantă a roboticii direct în mijlocul comunității. Găzduit într-un mall, într-un spațiu accesibil și prietenos, evenimentul a atras copii, adolescenți și adulți deopotrivă — toți dornici să descopere cum arată tehnologia atunci când este pusă în mișcare de pasiune și creativitate.

Pentru această ediție, am amenajat un stand interactiv dedicat universului FTC, unde vizitatorii au putut înțelege cum funcționează un robot, care sunt etapele unui sezon competițional și ce înseamnă munca din spatele unei echipe de robotică.

Alături de fondatorii inițiativei ILC, am prezentat conceptul FIRST Tech Challenge într-un mod ludic și accesibil: jocuri cu piese LEGO, mini-provocări de logică, demonstrații tehnice și explicații pe înțelesul tuturor. Entuziasmul copiilor, dar și interesul adulților, au creat o energie aparte, transformând spațiul într-un mic laborator al inovației.

Un moment extrem de valoros a fost interacțiunea cu celelalte echipe prezente. Discuțiile tehnice, schimburile de idei și prezentările informale despre strategii și mecanisme ne-au oferit perspective noi asupra provocărilor actualului sezon FTC. Am reușit să identificăm detalii și optimizări pe care, poate, nu le-am fi observat în ritmul intens al pregătirilor — o dovadă că învățarea devine mai profundă atunci când este împărtășită.

„Fall in Robotix” nu a fost doar un eveniment, ci o experiență care a consolidat legături, a inspirat viitori ingineri și a întărit comunitatea noastră STEAM. Pentru echipa noastră, a reprezentat un pas important spre misiunea de a promova tehnologia într-un mod accesibil, prietenos și motivațional. Ne-a reamintit că fiecare interacțiune, fiecare întrebare și fiecare zâmbet curios pot aprinde pasiuni noi și pot modela parcursul generațiilor care vor construi viitorul.



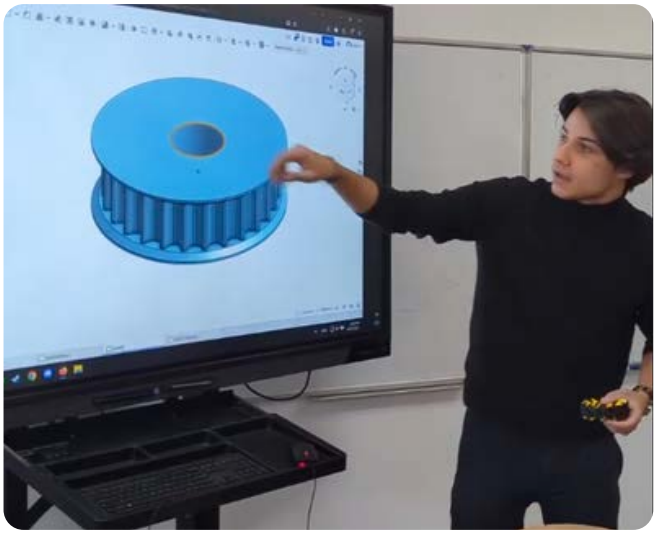
INNOVATION

ONE HOUR OF ROBOTICS

IMPACT 80 PERSOANE

În perioada de început a anului școlar am organizat, în cadrul Colegiului Național de Informatică „Tudor Vianu”, workshop-uri pentru colegii noștri

. Am organizat aceste ateliere la programare, hardware și marketing. Ne-am bucurat să vedem că destui dintre ei aveau deja noțiuni de bază, astfel că am putut lucra cu ei la un nivel mai înalt.



Le-am prezentat codul robotului din sezonul trecut și am lucrat în OnShape. La marketing, am discutat despre cum se realizează un design, cum funcționează strategia de finanțare și am făcut sesiuni de brainstorming, din care au reieșit multe idei ce s-au transpus în realitate.

Inovația, de cele mai multe ori, am observat noi, apare prin contopirea ideilor.
În urma acestor workshop-uri, ne-am pregătit voluntarii, acesta fiind un pas esențial în recrutarea viitoarei generații a echipei noastre.

DEFCAMP

IMPACT 270 PERSOANE

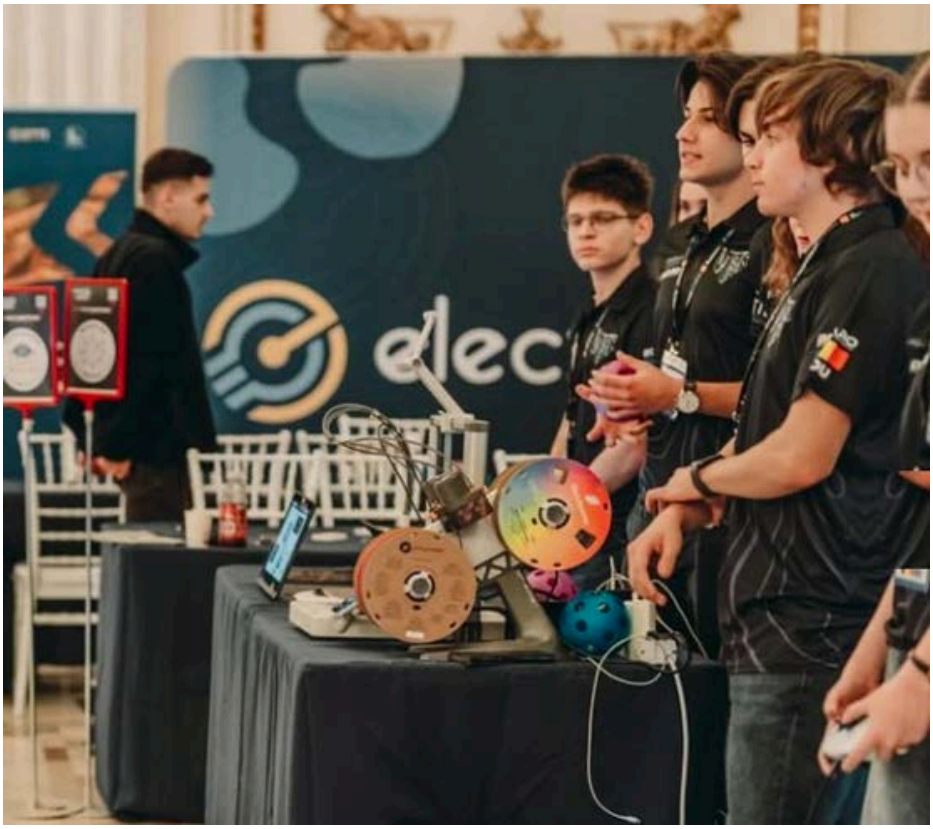
În perioada 13–14 noiembrie 2025, am avut privilegiul de a participa la DefCamp 2025, unul dintre cele mai importante evenimente de cibernetică din regiune. Aici am reușit să aducem în prim-plan valorile FIRST și spiritul FTC, folosind inteligența artificială ca instrument de prezentare și interacțiune. Atmosfera din jurul standului nostru a atras rapid atenția pasionaților de tehnologie, securitate informatică și inovație.

Participarea noastră, alături de ProTV și alte instituții media, ne-a oferit oportunitatea de a reprezenta cu mândrie comunitatea FTC și STEM într-un mediu profesional, dinamic și profund orientat către viitor.

DefCamp s-a dovedit a fi locul în care ideile îndrăznețe prind contur — fie printr-o linie de cod ingenioasă, fie printr-un design 3D ce reușește să transforme viziunea în realitate. Pentru echipa noastră, experiența a fost o adevărată fereastră spre un domeniu în continuă evoluție, inspirându-ne să înțelegem mai bine provocările și oportunitățile tehnologiei moderne.

Unul dintre cele mai memorabile momente ale evenimentului a fost vizita Ministrului Digitalizării, Economiei, Antreprenoriatului și Turismului, Radu-Dinel Miruță. Acesta și-a exprimat interesul pentru inițiativele dedicate progresului tehnologic și educației STEAM, oprindu-se la standul nostru pentru a afla mai multe despre misiunea și activitățile FTC. A testat robotul nostru cu entuziasm și a apreciat dedicarea, efortul și profesionalismul întregii echipe.

Această experiență ne-a consolidat motivația de a promova în continuare tehnologia, inovația și valorile FIRST, evidențiind impactul pe care robotica îl poate avea asupra formării generațiilor viitoare. DefCamp ne-a demonstrat că atunci când pasiunea se întâlnește cu perseverența, rezultatele pot inspira comunități întregi și pot contribui la construirea unui viitor mai inteligent și mai conectat.



DISCOVERY

FUTURE TALKS

IMPACT 120 PERSOANE

Prima ediție a evenimentului Future Talk, organizat în colaborare cu conducerea Colegiului Național de Informatică „Tudor Vianu”, a reprezentat un real succes și un pas important în promovarea educației tehnologice în rândul tinerilor.

Evenimentul a debutat cu prezența unor invitați de excepție: Ioana Bica, Principal Research Scientist la Google DeepMind, și Mircea Rozolea, Chief Technology Officer la Emia, care au susținut prezentări captivante despre inteligența artificială și modul în care aceasta ne influențează viața de zi cu zi, adesea fără ca noi să fim conștienți de acest lucru.

Prin exemple concrete și explicații clare, invitații au reușit să aducă mai aproape de public un subiect complex, evidențiind impactul real al AI-ului în domenii precum educația și viața cotidiană. Prezentările lor au stârnit interes și au încurajat gândirea critică, deschizând noi perspective asupra viitorului tehnologiei.

Ne-am bucurat de o prezență numeroasă, cu peste 100 de participanți, care au venit cu minți deschise, entuziasm și curiozitate față de lumea inteligenței artificiale. Atmosfera a fost una interactivă, iar publicul a demonstrat un interes autentic pentru subiectele discutate.

Un rol esențial în reușita evenimentului l-a avut Departamentul de Marketing al echipei noastre, care s-a organizat exemplar, asigurând o desfășurare fluentă și profesionistă a întregului eveniment.

Datorită efortului și dedicării lor, Future Talk a fost nu doar un eveniment educativ, ci și unul inovator, bine structurat și lipsit de probleme logistice.

Un moment deosebit de apreciat a fost sesiunea de întrebări și răspunsuri, în care participanții au avut ocazia să le adreseze invitaților întrebări legate de tehnologie, carieră și viitorul inteligenței artificiale.Curiozitatea și implicarea publicului ne-au confirmat importanța organizării unor astfel de inițiative.

În concluzie, prima ediție Future Talk a demonstrat că există un interes real pentru subiectele de actualitate din domeniul tehnologiei și ne motivează să continuăm organizarea unor evenimente care să inspire, să informeze și să pregătească tinerii pentru provocările viitorului digital.

8.3 CONCURSURI

UNLOCK THE MOTIF – CHAPTER ONE –

Unlock The Motif – Chapter One, competiție organizată de echipele #19075 Clockworks, #12560 Soft Hoarders și #19049 High Five, desfășurată pe data de 14 decembrie, a reprezentat primul concurs la care echipa noastră a participat în acest sezon. Acest eveniment a fost prima ocazie reală de a ne testa abilitățile pe terenul de joc, de a evalua performanța robotului în condiții de competiție și de a înțelege mai bine dinamica meciurilor FTC.

Pe lângă experiența competițională, meet-ul ne-a oferit oportunitatea de a interacționa cu alte echipe participante, facilitând un schimb valoros de idei, soluții tehnice și strategii de joc. Aceste discuții ne-au ajutat să identificăm puncte forte, dar și aspecte care necesitau îmbunătățiri.

Deși primele meciuri au fost marcate de diverse probleme tehnice, am reușit să le diagnosticăm și să le remediem într-un timp scurt, demonstrând capacitatea echipei de a lucra eficient sub presiune. Ca rezultat al acestor ajustări, am câștigat ultimele trei meciuri disputate, iar în fiecare dintre ele am stabilit un nou record de puncte pentru eveniment.

Această experiență ne-a ajutat să ne dezvoltăm abilitățile de rezolvare a problemelor în condiții de stres, să testăm și să rafinăm strategii de joc eficiente și să căpătăm mai multă încredere în munca noastră. Unlock The Motif – Chapter One a fost un pas important în pregătirea noastră pentru competițiile viitoare și a contribuit semnificativ la evoluția echipei în acest sezon.

ICESPARK MEET

Competiția „IceSpark Meet”, organizată de echipele #19099 H-tech, #19054 NeuroBotix și #14270 Quantum Robotics, desfășurată pe data de 21 decembrie, a reprezentat cel de-al doilea League Meet la care echipa noastră a luat parte în acest sezon competițional. Evenimentul a reunit unele dintre cele mai bine pregătite și experimentate echipe din regiune, oferindu-ne un cadru competitiv valoros în care ne-am putut testa limitele și progresul realizat până în acest moment.

Această competiție a fost o oportunitate excelentă de a ne demonstra abilitatea de a colabora eficient ca echipă, de a lua decizii rapide și de a ne adapta strategiile în funcție de situațiile întâlnite pe teren. Participarea la meciuri de un nivel atât de ridicat ne-a ajutat să ne consolidăm experiența competițională și să ne confirmăm capacitatea de a concura la cel mai înalt nivel, alături de echipe cu o vastă experiență în domeniul roboticii.

Ca rezultat al muncii constante depuse de-a lungul sezonului și al pregătirii intense, am reușit să obținem victoria în 5 din cele 6 meciuri disputate, un rezultat care reflectă clar dedicarea, perseverența și efortul colectiv al întregii echipe. Mai mult decât atât, împreună cu echipa #20265 Heart of RoBots, am stabilit un nou record european, o performanță remarcabilă care ne-a oferit o motivație suplimentară și ne-a demonstrat că suntem pe drumul cel bun.

Această reușită ne-a întărit încrederea în propriile forțe și ne-a confirmat faptul că munca depusă până acum a meritat pe deplin, reușind să ocupăm locul 4 în cadrul competiției.

ICECODE SHOWDOWN

Competiția „IceCode Showdown”, organizată de echipa Gamma #15991, a avut loc pe data de 10 ianuarie, la Colegiul Național „Sfântul Sava”, și a reprezentat al treilea League Meet la care echipa noastră a participat în acest sezon competițional.

Fiind prima competiție din noul an, acest eveniment a avut un rol important pentru echipa noastră, ajutându-ne să ne reintrăm în ritmul concursurilor, să ne testăm strategia și să ne consolidăm colaborarea în cadrul echipei. Atmosfera a fost una competitivă, dar în același timp prietenoasă, oferindu-ne ocazia să interacționăm cu alte echipe și să învățăm din experiențele lor.

Pe parcursul competiției, am reușit să obținem 5 victorii din 6 meciuri disputate, un rezultat care reflectă munca depusă în perioada de pregătire. Datorită acestui parcurs, echipa noastră s-a clasat pe locul al III-lea în clasamentul general, o performanță care ne motivează să continuăm să ne îmbunătățim.

Un moment deosebit al competiției a fost obținerea event high score-ului, realizare pe care am împărțit-o cu echipa Quantum Robotics #14270. Acest rezultat ne confirmă progresul și ne oferă încredere pentru următoarele competiții, demonstrând că suntem pe drumul cel bun și pregătiți pentru provocările ce urmează.

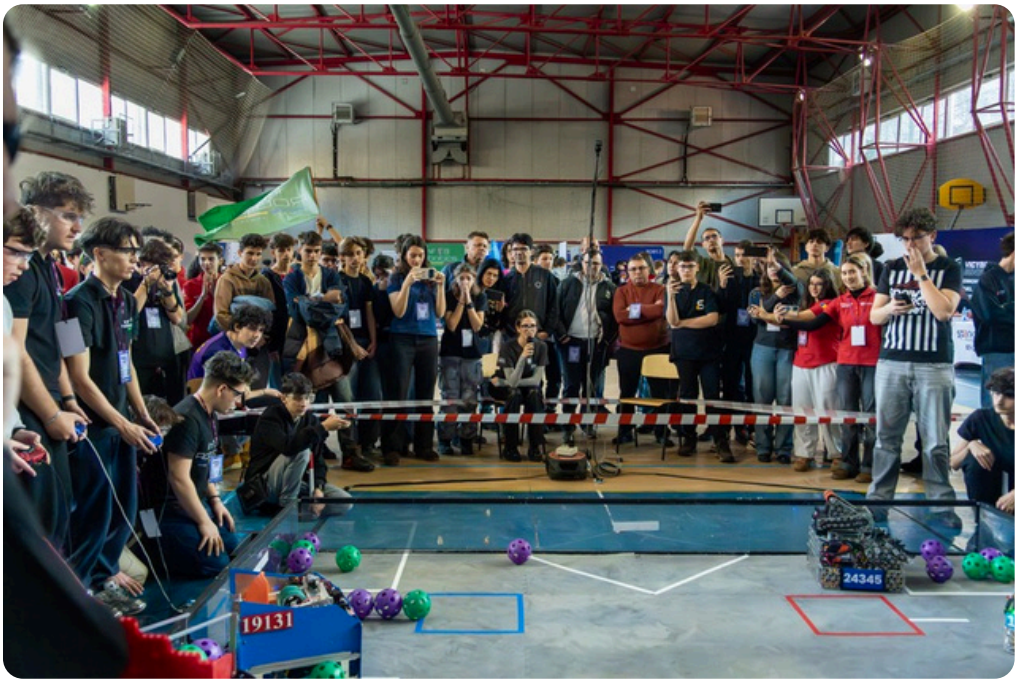
SNOWY CIRCUITS

Competiția „Snowy Circuits”, organizată de echipa noastră, în colaborare cu echipa SPARKTECH #24345, s-a desfășurat la Colegiul Național de Informatică „Tudor Vianu” și a reprezentat o adevărată provocare, atât din punct de vedere al abilităților tehnice, cât și al celor organizatorice.

Fiind prima competiție organizată de echipa noastră, acest eveniment a avut o importanță deosebită pentru noi. Am depus mult efort în etapa de planificare, iar interesul manifestat de celelalte echipe ne-a onorat profund. Am fost plăcut surprinși de numărul mare de participanți, ceea ce ne-a confirmat că ideea competiției a fost bine primită și apreciată în comunitate.

Pe parcursul desfășurării concursului, ne-am confruntat cu diverse provocări tehnice și logistice, însă am reușit să le gestionăm eficient prin comunicare, cooperare și spirit de echipă. Fiecare obstacol apărut a fost o oportunitate de a învăța și de a ne dezvolta abilitățile, iar implicarea tuturor membrilor a fost esențială pentru succesul evenimentului.

În ciuda dificultăților întâmpinate, am reușit să asigurăm buna desfășurare a competiției, să rezolvăm prompt problemele apărute și să oferim participanților o experiență plăcută și bine organizată. „Snowy Circuits” a fost nu doar un test al capacităților noastre organizatorice, ci și o experiență valoroasă care ne-a ajutat să creștem ca echipă și să câștigăm încredere pentru organizarea viitoarelor evenimente.



AIR'S AGEZ

Concursul „Air's AgeZ”, organizat de echipele TehnoZ 15972 și Robo-Sapiens 19117, a reprezentat ultimul League Meet la care echipa noastră a participat în acest sezon competițional. Evenimentul a avut o importanță majoră, fiind ultima oportunitate de a ne evalua nivelul de pregătire înaintea etapei Regionale Sud.

A fost ultima ocazie pe care am avut-o să observăm progresul celorlalte echipe și să testăm posibilele strategii pe care aveam să le aplicăm la competiția viitoare.

Munca constantă, analiza jocului și pregătirea ne-au adus victoria în 5 din cele 6 meciuri jucate, primele 2 punctaje pe eveniment și locul 2 în clasamentul echipelor participante.

Rezultatele obținute au confirmat progresul tehnic și strategic realizat și ne-au motivat să lucrăm și mai intens în săptămâna premergătoare Regionalei Sud. Experiența acumulată ne-a oferit încrederea că putem concura la cel mai înalt nivel și că avem potențialul de a avansa către etapa națională.

ROMANIA SOUTH LEAGUE TOURNAMENT

Regionala Sud a reprezentat cel mai important eveniment la care echipa noastră a participat în acest sezon. A fost apogeul tuturor orelor de muncă, al tuturor sesiunilor de antrenament și șansa de a ne dovedi progresul și de a arăta că merităm să progresăm la etapa națională și să reprezentăm România pe scena internațională.

Am avut oportunitatea să comunicăm cu toate echipele din regiunea noastră și să facem schimb de strategii și informații, bucurându-ne în același timp de diverse opinii despre acest sezon.

În urma meciurilor de calificare, am câștigat 5 din cele 6 meciuri disputate și am setat un nou record european, alături de echipa Quantum Robotics 14270, reușite ce ne-au oferit motivare și încredere, intrând în faza de play-off-uri.

În meciurile de play-off, am participat în calitate de căpitani ai alianței a 3-a, alături de echipa Renaissance Robotics 22226, etapă în care am jucat 3 meciuri. De asemenea, am fost onorați să primim Premiul CONTROL I, oferit pentru cel mai bine programat robot, recunoaștere ce ne-a validat progresul și munca.

Toate aceste reușite au avut ca rezultat avansarea spre etapa națională și șansa de a concura pentru onoarea de a ne reprezenta țara la cele mai înalte niveluri competiționale.

CAPITOLUL 9
CONCLUZII

FIRST Tech Challenge nu înseamnă doar proces ingineresc sau linii de cod scrise până târziu în noapte. FTC este, înainte de toate, un program care creează legături reale între oameni, care te ajută să îi cunoști în adevăratul sens al cuvântului — atât cu părțile lor bune, cât și cu momentele mai dificile.

„More than robots” nu este doar un slogan, ci o realitate pe care o trăiești zi de zi. Este un proiect care te învață ce înseamnă munca în echipă, perseverența și încrederea, demonstrându-ți că, chiar și atunci când simți că ai ajuns la limită, mai există mereu puțină putere ca să mergi mai departe. FTC formează caractere, dezvoltă prietenii și construiește experiențe care rămân cu tine mult dincolo de competiție.