



## DIPLOMATERVEZÉSI FELADAT

**Herőczi Sándor**  
Mechatronikai mérnök hallgató részére

# Mikrovezérlő alapú autonóm fegyverrendszer tervezése és fejlesztése

Napjainkra a technológia fejlődés lehetővé tette a teljesen autonóm hadi eszközök széleskörű megjelenését, elég ha harci drónokra gondolunk. Bár az ilyen autonóm, tehát működésüket tekintve emberi beavatkozástól és felügyelettől mentes fegyverrendszerek használata számos jogi, morális és politikai kérdést vet fel, műszaki szempontból komoly kihívást jelentenek a fejlesztők számára. A diplomamunka célja egy kisméretű, hobbycélokra használható airsoft típusú fegyverrendszer tervezése és megvalósítása.

A hallgató feladatának a következőkre kell kiterjednie:

- Végezzen irodalomkutatást az autonóm fehgyverrendszer területén!
- Készítse el az eszköz koncepciótervét, válassza ki a fontosabb eszközöket!
- Tervezze meg az eszköz egyedi mechanikáját!
- Válassza ki a prototípushoz szükséges elektornikai elemeket, szükséges esetén készítse el az áramköri terveket!
- Valósítsa meg egy mikrokontrolleren az eszköz vezérlőkódját!
- Tesztelje az eszköz működését!
- Dokumentálja a kapott eredményeket!

**Tanszéki konzulens:** Dr. Stumpf Péter Pál, egyetemi docens

**Külső konzulens:**

Budapest, 2024. február 18.

Dr. Charaf Hassan  
egyetemi tanár  
tanszékvezető

---

BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM  
VILLAMOSMÉRNÖKI ÉS INFORMATIKAI KAR  
AUTOMATIZÁLÁSI ÉS ALKALMAZOTT INFORMATIKAI TANSZÉK

---

# Mikrovezérlő alapú autonóm fegyverrendszer tervezése és fejlesztése

Herőczi Sándor  
WH0AMC



## Hallgatói nyilatkozat

Alulírott Herőczi Sándor, szigorló hallgató kijelentem, hogy ezt a diplomatervet meg nem engedett segítség nélkül, saját magam készítettem, csak a megadott forrásokat (szakirodalom, eszközök stb.) használtam fel. minden olyan részt, melyet szó szerint, vagy azonos értelemben, de átfogalmazva más forrásból átvettettem, egyértelműen, a forrás megadásával megjelöltetem.

Hozzájárulok, hogy a jelen munkám alapadatait (szerző(k), cím, angol és magyar nyelvű tartalmi kivonat, készítés éve, konzulens(ek) neve) a BME VIK nyilvánosan hozzáférhető elektronikus formában, a munka teljes szövegét pedig az egyetem belső hálózatán keresztül (vagy hitelesített felhasználók számára) közzétegye. Kijelentem, hogy a benyújtott munka és annak elektronikus verziója megegyezik. Dékáni engedéllyel titkosított diplomatervek esetén a dolgozat szövege csak 3 év eltelte után válik hozzáférhetővé.

Kelt: Budapest, 2024.06.02.

.....  
Herőczi Sándor

## Összefoglaló

A diplomamunkám célja egy autonóm fegyverrendszer fejlesztése és el-készítése, amely funkciójában hasonlít a valós, éles helyzetben alkalma-zott rendszerekhez. Ez annyit takar, hogy a fegyvernek képesnek kell egy bizonyos méretű területet beláttni, ebben felismerni és azonosítani a cél-pontokat, majd tüzelni, vagy engedélyt kérni tüzelésre. Ezentúl szükséges funkciója a manuális vezérlés, amely segítségével az operátor valós időben tudja távolról irányítani az eszközt, egy asztali számítógép segítségével.

Az eszköz alkatrészei 3D nyomtatási technológiával készültek, a kötőele-meket, csapágvakat és elektronikai elemeket kivéve, első feladatom ezek nyomtatáshelyes megtervezése volt. A következő lépés a hardver és elekt-ronikai rendszer tervezése volt, majd a szükséges modulok, szenzorok, mikrovezérlő kiválasztása és megrendelése. Az utolsó alfeladat pedig a szoftver fejlesztése, amely jelentette mind a beágyazott vezérlő szoftvert, és a számítógépen futó felhasználói felületet is.

A projekt megvalósítása során végig sok időt emészttet fel a rendszer tesz-telése, amelyet párhuzamosan kellett végezni a későbbi alfeladatok tervezésével. Kihívást jelentett a megfelelően részletes szakirodalom felkutatá-sa is, ugyanis a valós megoldások paramétereit gyakran nem teszik elér-hetővé civilek számára, illetve a jelenleg folyó fejlesztések is titkosítottak.

Úgy gondolom, a téma interdiszciplináris jellegét tekintve jól illik a me-chatronikai tanulmányaiba, ugyanis egyaránt érinti a műszaki mechanika, a gyártástechnológia, a szoftverkészítés és a jelfeldolgozás területeit.

Diplomamunkám eredményeként megvalósítok egy működő prototípust, amit előadás keretében fogok bemutatni. Végül pedig kitérek az esetleges hibákra amiket elkövettem, a tanulságokra és a továbbfejlesztési lehetősé-gekre.

## Abstract

The aim of my thesis is to develop and build an autonomous weapon system that functions similarly to real-world systems used in live scenarios. This means that the weapon must be able to monitor a designated area, recognize and identify targets, then fire or request permission to fire. Additionally, a necessary feature is manual control, allowing the operator to remotely control the device in real-time using a desktop computer.

The components of the device were created using 3D printing technology, except for the fasteners, bearings, and electronic elements. My first task was to design these components to be suitable for printing. The next step was to design the hardware and electronic system, followed by selecting and ordering the necessary modules, sensors, and microcontroller. The final subtask was the development of the software, which included both the embedded controller software and the user interface running on a computer.

Throughout the project, significant time was consumed by system testing, which had to be carried out simultaneously with the planning of subsequent subtasks. A challenge was also posed by finding sufficiently detailed technical literature, as the parameters of real-world solutions are often not made available to civilians, and current developments are often classified.

I believe that the interdisciplinary nature of this topic fits well with my mechatronics studies, as it encompasses areas such as mechanical engineering, manufacturing technology, software development, and signal processing.

As a result of my thesis, I will produce a working prototype, which I will present during my final defense. Lastly, I will address any mistakes I made, lessons learned, and possibilities for future development.

## Tartalomjegyzék

<b>1. Bevezetés</b>	<b>6</b>
1.1. Motiváció és háttér . . . . .	6
1.2. Kihívások és célkitűzések . . . . .	9
1.3. Korlátozások . . . . .	10
1.4. Erkölcsi nyilatkozat . . . . .	11
<b>2. Irodalomkutatás</b>	<b>12</b>
2.1. Megvalósult rendszerek . . . . .	12
2.2. Tüzelési mechanizmus . . . . .	16
<b>3. Rendszertervezés és követelmények</b>	<b>18</b>
3.1. Rendszeráttekintés . . . . .	18
3.2. Követelmények . . . . .	19
<b>4. Mechanikai tervezés</b>	<b>22</b>
4.1. Kinematika . . . . .	22
4.2. Mechanikai alkatrészek . . . . .	24
4.3. 3D tervezés, modellezés . . . . .	27
4.3.1. Torony . . . . .	28
4.3.2. Keret . . . . .	31
4.3.3. Függőleges tengely elemei . . . . .	31
4.3.4. Fogaskerekek . . . . .	32
4.4. Gyártás és összeszerelés . . . . .	34
4.4.1. Gyártás . . . . .	34
4.4.2. Összeszerelés . . . . .	35
<b>5. Hardvertervezés</b>	<b>37</b>
5.1. Áramköri tervezés . . . . .	37
5.2. Elektronikai alkatrészek . . . . .	39
5.3. Gyártás, forrasztás . . . . .	43
<b>6. Szoftverfejlesztés</b>	<b>44</b>
6.1. Használt Python modulok, eszközök . . . . .	44
6.2. Gépi látás . . . . .	47
6.2.1. Template matching [24] . . . . .	47
6.2.2. Feature Matching [25] . . . . .	47
6.2.3. Target Tracking [26] . . . . .	48
6.2.4. Haar cascade [27] . . . . .	48
6.3. A szoftver működése . . . . .	50

6.3.1. A PC-n futó program működése . . . . .	50
6.3.2. A Raspberry Pi-n futó program működése . . . . .	52
<b>7. Tesztelés</b>	<b>55</b>
7.1. Elektronikai alkatrészek tesztje . . . . .	55
7.2. Képfelismerő algoritmusok tesztje tesztje . . . . .	56
7.2.1. Haar Cascade . . . . .	57
7.3. Éles teszt . . . . .	57
<b>8. Hibák, észrevételek</b>	<b>61</b>
<b>9. Továbbfejlesztési lehetőségek</b>	<b>62</b>
<b>10. Költségek</b>	<b>63</b>
<b>11. Konklúziók</b>	<b>63</b>
<b>12. Köszönetnyilvánítás</b>	<b>63</b>

## 1. Bevezetés

### 1.1. Motiváció és háttér

**E**zt a projektet nem a tanszéki diplomamunkatémák listáján találtam, hanem én kerestem hozzá konzulest. Szeretném ezért először megemlíteni a személyes motivációm, és érdekeltségeimet vele kapcsolatban. Mint a legtöbb reál beállítottságú fiú, engem is fiatal korom óta érdekel a technológia, a járművek, a gépek, vagy a fegyverek. Ezek közös metszéspontja a haditechnológia, ami általában jóval fejlettebb, mint amivel a civil életben találkozhatunk. Ez az érdeklődés megmaradt kamaszkoromra is, amikor a videojátékok által új ingerként értek a *Team Fortress 2*-ben és a *Portal*-ban lévő ún. **Sentry Gun**-ok.



1. ábra. TF2 Sentry Gun [1]



2. ábra. Portal Sentry Gun [2]

Ezek olyan lövegek, amelyek a lábukon állva képesek voltak forogni, és ha az ellenség a látóterükbe jutott, akkor arra tüzet nyitottak. Nyalván játékokról van szó, tehát mindig azt hittem hogy ez csak a jövő technológiája, de nem sokkal később rá kellett jönnöm, hogy nagyon is valódiak. Ezután, mikor már egyetemre jártunk, egy barátommal egyre komolyab-

ban kezdtünk beszélgetni arról, hogy esetleg mi is tudnánk egyet építeni. Végül az egyetemi éveim végére érve el is jött a tökéletes alkalom, hogy megvalósítsam régi álmomat.

A szentimentalitást félretéve, természetesen nem választottam volna ezt a témát, ha nem lenne a tanulmányaim szempontjából is releváns. Úgy gondolom, ez a projekt a mechatronika oktatás sok fontos elemét magában hordozza. A feladat a mechanikai konstrukcióval kezdődik és a CAD modellezéssel, amely a gépészeti oldalát hasznosítja a képzésünknek. Foglakozni kell a hardverrel, amihez elektronikai ismeretek szükségesek. Végül pedig a szoftvert kell lefejleszteni, ami informatikai szempontból érdekes. Ráadásul az egész beágyazott környezetben történik, ami miatt még relevánsabb az *intelligens beágyazott rendszerek* specializációmhoz. Véleményem szerint a projekt nehézsége és kihívása nem az egyes alfeladatokban rejlik, hanem az egész rendszer integrálásában, egymáshoz illesztésében.

Szintén fontosnak tartom megemlíteni, hogy a feladat lehetőséget ad megismerni és használni pár igen modern technológiát. A közelelmúltban elérhetővé (és megfizethetővé) váltak civil felhasználásra a 3D nyomtatók, valamint a fejlődés a *Deep Learning* algoritmusok terén nagyban befolyásolta a gépi látás szakterületét is. Többek között a célom, hogy jobban megismerkedjek ezekkel a módszerekkel, és alkalmazzam őket.

Mint manapság az ipar minden területén, így a fegyveriparban is egyre nagyobb mértékű a digitalizáció és az automatizáció. Ennek ékes példája a távolról irányítható lőállások téryerése, amelyeknek nagy előnye a fegyver és a tüzér egymástól való elkülönítése, aminek több előnye is van. Természetesen a legfontosabb és leginkább szembetűnő, hogy ezzel a módszerrel minimalizálható, vagy akár megszüntethető a saját embereink életének kockázatája. Ezen túl olyan helyen is tudjuk használni ezeket az eszközöket, ahova egy tradicionális géppuskafések telepítése nehézkes lenne, például



3. ábra. Phalanx CIWS rendszer [3]

mostoha természeti körülmények közé, egy torony tetejére, vagy akár egy hadihajó oldalára. Szintén egy nagy előny, hogy ezek az eszközök felszerelhetők több kezeléssegítő alegységgel, például hőkamerával vagy éjjellátóval. Majdnem minden, számottevő hadsereggel rendelkező országnak van saját fejlesztésű távirányított fegyverrendszere.

A következő lépés az automatizálás. Hiszen egyre erősebb hardverekkel rendelkezünk, egyre jobb algoritmusokat tudunk implementálni, és elérültük az a szintet, hogy bizonyos helyzetekben a "gép" jobb munkát tud végezni, mint egy ember. Az első automatikus célzórendszerrel rendelkező légvédelmi gépágyú az amerikai *Phalanx CIWS* [3] az 1970-es években került kifejlesztésre, ezzel megszületett a "Lethal autonomous weapon (LAW)" kifejezés.

A technológiát érhető módon leggyakrabban védelmi célokra használják, sokszor légvédelemre. A gyakorlatban nagy szerepe van Dél-Korea és Izrael védelmében, ahol a rakétámadások minden nap veszélyt jelentenek. Offenzív célokra a gyakorlatban még csak rakéták célzására használnak automatikát, a "terminátor" jellegű gyilkos robotok még csak fejlesztési fázisban vannak.

## 1.2. Kihívások és célkitűzések

A hagyományos, ember által irányított biztonsági rendszerek, illetve manuálisan vezérelt fegyverek hatékonysága korlátozott. Az ember reakcióideje lassú lehet krízishelyzetben, különösen ha nagy sebességű, gyors reagálású, esetleg automatizált a célpont. További gyengesége az embernek, hogy hajlamos hibázni. A fáradtság, stressz, éhség, és rengeteg egyéb tényező kényszerítheti hibára. Ez szükségessé teszi egy olyan automata rendszer kifejlesztését, amely képes azonnal reagálni, felismerni a fenyegetéseket és pontosan célozní.

Az automatikus gépágyúk fejlesztése során számos technikai kihívás merül fel:

**Célpontok felismerése és követése:** A gépágyúnak gyorsan és pontosan kell érzékelnie és nyomon követnie mozgó célpontokat. Ez kihívást jelenthet változó fényviszonyok, mozgássebességek és környezeti zavaró tényezők mellett. A gépágyúnak gyorsan és pontosan kell érzékelnie és nyomon követnie mozgó célpontokat. Ez kihívást jelenthet változó fényviszonyok, mozgássebességek és környezeti zavaró tényezők mellett.

**Pontosság és reakcioidő:** A rendszernek gyorsan kell döntéseket hoznia, ugyanakkor elengedhetetlen a pontos célzás. A mechanikai mozgásvezér-lés, a számítógépes látás és az elektronikai rendszerek szinkronizációja mind kritikus tényező a rendszer hatékonysága szempontjából.

**Környezetérzékenység:** Külső környezeti tényezők (pl. időjárás, akadályok, fényviszonyok) befolyásolhatják a rendszer működését, ezért a szoftvernek és a hardvernek rugalmASNak és robusztusnak kell lennie. A dolgozatom célja egy innovatív megoldás kidolgozása, amely eleget tesz a megszabott követelményeknek és megoldást nyújt a fenti problémákra.

Kiemelt célok:

Egy megbízható célpontfelismerési rendszer fejlesztése, amely változó környezeti feltételek mellett is képes hatékonyan működni.

Egy stabil és erős mechanikai konstrukció tervezése, amely képes követni a szoftver utasításait.

Egy olyan valós idejű szoftvervezérlési rendszer megalkotása, amely gyorsan reagál a célpontok mozgására és változására.

Összegezve tehát:

**Egy olyan automata gépágyú fejlesztése és megépítése, amely magabiztosan képes felismerni egy célpontot, követni azt, célozni, és tüzelni.**

### 1.3. Korlátozások

Természetesen tisztában kell lenni bizonyos korlátozásokkal a projekt elkészítése közben. Két félévem van lefejleszteni az eszközt a nulláról, egyedül vagyok, és nincsen százmilliós bűdzsem, mint az iparban hasonló kutatással foglalkozó csapatoknak. Ebből kifolyólag reáliisan kell látni a helyzetet, és úgy meghatározni a követelményeket, hogy egy egyetemi hallgató számára is elérhető legyen.

A kamerarendszer például, amit a valós megoldásokban használnak, önmagában tízmilliós téTEL szokott lenni. Sajnos az én megvalósításom valószínűleg nem fog működni sötétben, nagy távolságokban, vagy esőben, hiszen a költségvetésbe nem fér bele az éjjellátó, a hőkamera, az optikai zoom vagy több kamerából álló rendszer.

Nem áll rendelkezésemre korlátlanul se CNC gép, se fém 3D nyomtató, így a legtöbb alkatrész műanyagból kell 3D nyomtatnom. Ez befolyásolhatja a szerkezet stabilitását, amit figyelembe kell venni a későbbiekben.

És végül fontos megemlíteni, hogy mivel éles fegyvert alkalmazni minden bizonnal a törvénybe ütközne, ezért valamilyen játékfegyvert kell majd használnom. Ez befolyásolni fogja a gépágyú pontosságát, amit szintén figyelembe kell venni a tervezéskor és értékeléskor.

## 1.4. Erkölcsi nyilatkozat

Az automata fegyverrendszer morális megítélése vitatott téma, ezért szerelemnem kijelenteni, hogy diplomamunkám, melynek címe „*Mikrovezérlő alapú autonóm fegyverrendszer tervezése és fejlesztése*”, kizárolag tanulányi és mérnöki célokat szolgál. A dolgozatom keretében fejlesztett prototípus semmilyen formában nem szándékozik vagy ösztönzi az erősza-kos, káros vagy törvénysértő tevékenységeket. Fejlesztésem célja a tech-nológiai kutatás, a mechatronikai és automatizálási rendszerek megisme-reése, illetve a gépi látás alkalmazásainak tanulmányozása.

Határozottan elhatárolódom bármilyen rosszindulatú felhasználástól, és hangsúlyozom, hogy a projekt eredményei nem használhatók fel ártalmas vagy illegális célokra. A munka során mindenkor tiszteletben tartottam az etikai irányelveket, és felelős mérnöki magatartást tanúsítottam. Az általam fejlesztett rendszer kizárolag oktatási, kutatási és technológiai de-monstráció célját szolgálja.

## 2. Irodalomkutatás

A legjelentősebb katonai hatalmak mindegyike rendelkezik távvezérelt fegyverrendszerrel, leggyakrabban valamilyen távolról irányított gépfegyver formájában. Azonban se az egyes országok nemzetbiztonságának, se a fegyveripari partnereknek nem áll érdekében a szükségesnél több információt kiadni. Ez egy kicsit megnehezítette az irodalomkutatást, de a képek alapján azért sok információt ki lehet nyerni.

### 2.1. Megvalósult rendszerek

**CROWS [4]** Az egyik legnagyobb darabszámban gyártott távirányított fegyverrendszer az amerikai CROWS rendszer, amely a NATO-országokban, köztük Magyarországon is rendszeresített. Ennek értelmében telepíthető sok NATO által használt páncélozott járműre, köztük a Humvee-ra, a Stryker-re, és a Buffalo MRAP-re. Több verziója létezik több kaliberrel, a 4. ábrán egy M240B géppuskával látható.



4. ábra. Az amerikai CROWS rendszer [4]

A szerelék a fegyverrel együtt  $360^{\circ}$ -ban képes elfordulni, és  $-20^{\circ}$ -tól  $+60^{\circ}$ -ig tud billenni. A fegyvercső giroszkóppal stabilizált. A kezelő egy 15 hüvelykes kijelzőn tud célozni a fegyverrel. A rendszer a sima kamera mellett rendelkezik hőkamerával is, így éjszaka is használható. Mind a két kamera el van látva lézeres távolságmérővel, amivel rá lehet állni a célpontra, és a jármű mozgása közben is lehet azt követni. A kamerát és a fegyvert lehet külön is mozgatni, ami azért hasznos, mert anélkül lehet követni a gyanús alakok mozgását, hogy félelmet keltenénk az emberekben.

**Arbalet-DM [5]** A CROWS rendszer orosz megfelelője a hasonló kialakítású Arbalet-DM (5.ábra). Ennek a rendszernek az alapja a 12.7 mm-es KORD nehézgéppuska. Rendelkezik 4 gránátvetővel is, amelyek füstfüggöny felhúzására használható. A kamera és a fegyver elhelyezése, de még a lőszer pozíciója is teljesen hasonló az amerikai párrához.



5. ábra. Az orosz Arbalet-DM rendszer [5]

A rendszer  $360^{\circ}$ -ban képes elfordulni, és  $-20^{\circ}$ -tól  $+70^{\circ}$ -ig tud billenni, tehát egy kicsit nagyobb részt tud lefedni, mint a CROWS. A hatótáv nap-pal 2000 m, éjszaka 1500 m. Ez a rendszer is el van látva hőkamerával és

lézeres távolságmérővel.

**DeFNder [6]** A következő megoldás a belga DeFNder termékcsalád, amelynek két tagja a *Light* és a *Medium*(6. ábra). Értelem szerűen kettő közül az utóbbi az, amelyre nehezebb fegyverzetet lehet telepíteni. A függőleges tengelyen 360°-ban képes elfordulni 90 fok/másodperc sebességgel, a vízszintes tengelyen  $-45^{\circ}$ -tól  $+75^{\circ}$ -ig tud billenni, 60 fok/másodperc sebességgel. Opcionálisan ellátható infravörös- és hőkamerával a rossz látási körülmények esetére, valamint lézeres távolságmérővel a ballisztikai kompenzációhoz.



6. ábra. A belga DeFNder rendszer [6]

Elég sok közös vonást találtam a fentebb említett rendszerekben. Az elrendezésük nagyon hasonló, van egy függőleges forgástengely nagyjából a teljes rendszer súlypontján keresztül, valamint egy vízszintes forgástengely, nagyjából a fegyver csövével egy síkban. Erre valószínűleg azért van szükség, hogy a tüzelés során keletkező erők ne ébresszenek csavarónyomatékot a mozgató mechanizmuson. Az én esetemben nagy erők nem fognak ébredni, de a tervezési elvet érdemes követni.

**Samsung SGR-A1 [7]** Az egyetlen, valós harci helyzetben használt, teljesen autonóm gépágyú a koreai fejlesztésű *Samsung SGR-A1*. A két Korea között húzódó demilitarizált övezet (DMZ) egy 250 km hosszú, 4 km széles sáv, amelyet mind a két oldalon szigorúan ellenőriznek. A határ folyamatos felügyelete rengeteg ember munkájába kerül, ami egy demográfiai válságban lévő ország csökkenő hadseregében egyre értékesebb. Főleg annak tekintetében, hogy csupán járőrzni és figyelni a határt nem feltétlenül igényli egy ember jelenlétéit. Ennek tudatában fejlesztették ki a Samsung és a Korea Egyetem mérnökei az SGR-A1 fegyverrendszert. Hőkamerával és éjjellátóval felszerelve napközben 4 km-ről, éjszaka 2 km-ről képes azonosítani potenciális célpontokat, tehát tulajdonképpen a DMZ teljes szélességében. Képes felismerni az embereket, követni őket, és megkülönböztetni az állatoktól. Hangfelismeréssel képes azonosítani a közeledő személyeket: amennyiben valaki 10 m-nél közelebb kerül és nem azonosítja magát, a rendszer riaszthat, gumilövedéket lőhet, vagy használhatja a K-3 gépfegyverét.



7. ábra. Samsung SGR-A1 [7]

Civil szakértők számára nem egyértelmű, hogy a rendszer lőhet-e emberi engedély nélkül, és a hivatalos koreai álláspont az, hogy nem. Azonban ha már bemérte és ellenségeként azonosította a célpontot, akkor nehéz elképzelni, hogy pont a ravaszt ne tudná meghúzni.

## 2.2. Tüzelési mechanizmus

A korábban említett rendszerek éles fegyverekkel vannak felszerelve, ami az én esetemben sajnos nem megvalósítható. Így valamilyen más megoldást kellett találni, ami legális, és megfelelően be lehet vele mutatni a célfelismerés és célzás működését.

**Paintball [8]** Több, a diplomamunkámhoz hasonló projektet is találtam, amelyek paintball puskákat használnak. Ezek a puskák sűrített levegőt alkalmaznak, hogy egy festékkel töltött golyót lőjenek ki. A torkolati sebességük nagyjából 280 fps, maximális hatékony távolságuk kb 25-30 m. Mivel a lövedék alakja gömb, és a cső sincs huzagolva, ezért nem túl pontos, főleg hosszú távon.



8. ábra. Paintball puska alapú rendszer [9]

További problémák, hogy a legolcsóbb paintball puska is 70000 Ft. fölött van, valamint a lövedék is viszonylag drága, és nem lehet újrahasznosítani. Ezért a fegyverek nagyok, nehezek, és ezért nehéz a beépítésük.

**Nerf [10]** A Nerf fegyverek a legelterjedtebb játékfegyverek. Sűrített levegővel lönek ki egy hosszúkás szivacs lövedéket. A sűrített levegőt egy megfeszített rugó elengedésével érik el, amelyet vagy kézzel, vagy valamilyen átteles villanymotorral húznak fel. A legjobb modellek torkolati sebessége 70 fps körül van, és kb. 15 m a hatótávolságuk. Ezen a távon viszonylag pontosak a lövedék kialakításából adódóan, de jelentős az esés, így a ballisztikai pályát komolyan kell venni.

Ezek a fegyverek modelltől függően 10000 Ft. - 40000 Ft. között mozognak, de a lövedékeket újra lehet használni. Az a probléma itt is fennáll, hogy nagy a fegyver teste, így nehezebb beépíteni.

**Airsoft [11]** Az airsoft fegyverek hasonló módon működnek, mint a Nerf puskák, de egy kisebb, műanyag golyót lönek. Az elektromos airsoft fegyverek torkolati sebessége általában 300 fps és 400 fps között van, ami befolyásol a golyók tömege, a rugó minősége és rengeteg egyéb alkatrész. A hatótávjuk az átlagos fegyvereknek kb. 50 m, de fejlesztésekkel elérheti a 90 m-t is. A lövedék itt is golyó és a cső sincs huzagolva, mint a paintballnál, azonban az airsoft esetében használnak ún. hop-up kamrákat, amelyek perdületet adnak a golyónak.

Az airsoft fegyverek legnagyobb előnye a többi lehetőséggel szemben, hogy van egy kompakt egység, a "gearbox", ami felelős az elsütésért. Ezt ki lehet szedni egy fegyverből, vagy akár külön is meg lehet venni. Ez okkal nagyobb szabadságot ad a beépítéshez, és a végeredmény is sokkal kompaktabb lesz. Az elsütés is csak az áramkör zárását jelenti a gearboxban, ami egyszerűen vezérelhető a mikrokontrollerrel.

## 3. Rendszertervezés és követelmények

### 3.1. Rendszeráttekintés

Az autonóm fegyverrendszer fejlesztése során egy komplex mechatronikai rendszert kellett megvalósítani, amely több különálló modul együttműködését biztosítja. A rendszer fő komponensei közé tartozik a mechanikai szerkezet, az elektronikai hardver, az érzékelők, a vezérlő egység, valamint a szoftveres háttér. Ezek együttesen biztosítják a rendszer autonóm és manuális működését. Az alábbiakban részletes áttekintést adok a rendszer főbb elemeiről és azok feladatáról.

#### Mechanikai konstrukció

A váz gyanánt tulajdonképpen egy kéttengelyes pan-tilt mechanizmust kellett megvalósítanom, ami felelős a fegyver stabilan tartásáért és precíz mozgásáért. A váz építőelemei zömében 3D nyomtatási technológiával készültek, ami lehetővé tette a problémák gyors kiküszöbölését, illetve a nagyfokú szabadságot a tervezés során. Ahol tudtam, kereskedelemből beszerezhető alkatrészeket használtam, például a csapágyakat és a kötőelemeket.

#### Elektronikai rendszer

Az elektronikai rendszer két fő eleme a Raspberry PI, illetve a gearbox volt. Mivel a gearbox igen nagy áramot képes felvenni, külön tápegységgel kellett ellátnom a mikrovezérlőt illetve a golyó kilövéséért felelős eletronikát. A motorok vezérléséért egy kereskedelemben kapható, Raspberryvel kompatibilis shieldet használtam, amely jelentősen megkönnyítette a NEMA-17 típusú léptetőmotorok csatlakoztatását. A lézer diódát és szükséges szenzorokat a Raspberry PI GPIO lábairól vezéreltem, ahogy a gearbox aktiválásáért felelős relét is. A kamera modul egy Raspberry PI Camera V2 volt, amit a mikrovezérlőn található foglalaton keresztül tudtam elérni.

#### Számítógépes vezérlés és szoftveres háttér

A szoftverfejlesztés során két fő területet kellett lefedni: a vezérlőszoftvert, amely a mikrokontrolleren fut, és az asztali alkalmazást, amely a felhasználói felületet és a magas szintű irányítást biztosítja. A beágyazott vezérlő szoftver felelős a léptetőmotorok mozgatásáért, valamint a szenzorokból érkező információ feldolgozásáért. A számítógépen futó szoftver

feladata a felhasználó parancsainak feldolgozása és továbbküldése, valamint a kamera képének megjelenítése valós időben. Mivel a feladat összetett, mind a PC, mind a Raspberry Pi oldaláról történik küldés és fogadás is, ezért több szálon fut egyszerre a szoftver. A két egység közötti kommunikáció során kulcsfontosságú a megfelelő sebesség, ezért a vezetékes kapcsolat mellet döntöttem.

### 3.2. Követelmények

Az autonóm fegyverrendszer fejlesztése során számos követelményt kellett figyelembe venni annak érdekében, hogy a rendszer megbízhatóan, hatékonyan és biztonságosan működjön. Ezek a követelmények a rendszer mechanikai, elektronikai és szoftveres elemeire egyaránt kiterjedtek. A következő szakaszokban részletezem a legfontosabb műszaki és funkcionális követelményeket.

#### Mechanikai követelmények

A mechanikai komponensek tervezése során az alábbi követelményeknek kellett megfelelni:

**Stabilitás és pontosság:** A fegyverrendszer mechanikai szerkezetének stabilnak és tartónak kell lennie annak érdekében, hogy a lövések közben ne mozduljon el, és ne veszítse el a célpontot. Ugyanakkor elegendő pontosságot kell biztosítania a célpont precíz követéséhez.

**Fürgeség:** A rendszernek képesnek kell lennie a fegyver gyors és pontos mozgatására a pan-tilt mechanizmus segítségével. Ennek megfelelően a szervomotoroknak kellően gyorsnak és erősnek kell lenniük ahhoz, hogy valós időben tudják követni a mozgó célpontokat.

**Strapabíró konstrukció:** Habár nagy terhelés nem fogja érni, a gearboxból jöhetnek rezgések, rángások, amik esetleg problémát jelenthetnek egy alulmértezett alkatrész esetében.

#### Elektronikai követelmények

Az elektronikai rendszer megbízható működése érdekében az alábbi követelményeknek kellett eleget tenni:

**Megfelelő teljesítmény:** A szervomotoroknak és szenzoroknak megfelelő tápegységre van szükségük, amely stabil energiaellátást biztosít. A rendszer energiaigényét előzetesen fel kellett mérni, hogy a tápegység terhelés alatt is megfelelően működjön.

**Szenzorok pontossága:** A kamerának és egyéb szenzoroknak elelegendő felbontással és érzékenységgel kell rendelkezniük ahhoz, hogy képesek legyenek a célpontokat megfelelően azonosítani. A valós idejű képfeldolgozás nagy adatsebességet és megbízható szenzorleket igényel.

**Vezérlés:** A mikrokontrollernek kellően gyorsnak kell lennie, hogy a valós idejű adatokat folyamatosan feldolgozza és a vezérlési parancsokat késlekedés nélkül végrehajtsa.

### Szoftveres követelmények

A rendszer működéséhez szükséges szoftverfejlesztés során a következő követelményeknek kellett megfelelni:

**Valós idejű feldolgozás:** A számítógépes látás szoftverének valós időben kell elemeznie a kamerák által közvetített adatokat, felismerve a célpontokat és kiszámítva a mozgás irányát. A célzási és lövési döntések gyors és hatékony adatfeldolgozást igényelnek.

**Biztonságos működés:** A rendszernek rendelkeznie kell olyan szoftveres biztonsági funkciókkal, amelyek megakadályozzák a véletlen tüzelést. Ez magában foglalja a lövési engedélykérés mechanizmust és a manuális vészleállítás lehetőségét.

**Felhasználói interfész:** Az felhasználónak egyszerű és intuitív felhasználói felületet kellett biztosítani, amelyen keresztül könnyedén vezérelheti a rendszert, illetve áttekintheti a célpontok adatait és a kamera képét. A felületnek támogatnia kell a kézi irányítást és a lövési parancsok kiadását.

### Funkcionális követelmények

A rendszer teljes funkcionalitásának biztosítása érdekében a következő kritériumoknak kellett megfelelni:

**Autonóm működés:** A rendszernek képesnek kell lennie arra, hogy teljesen önállóan felismerje és kövesse a célpontokat, valamint meghozza a tüzelési döntéseket az előre meghatározott paraméterek alapján.

**Manuális vezérlés:** Az autonóm működés mellett manuális vezérlési lehetőséget is kellett biztosítani az operátor számára. Ezen keresztül a felhasználó közvetlenül irányíthatja a fegyvert és manuálisan adhat lövési parancsot.

### Biztonsági követelmények

Az autonóm fegyverrendszer használatával kapcsolatban különösen fontos a biztonsági követelmények teljesítése:

**Vészleállítás:** A rendszernek rendelkeznie kell egy vészleállító gombbal, amely azonnal megszakítja a fegyver működését, ha bármilyen hiba vagy vészhelyzet lép fel.

**Engedélyezési mechanizmus:** A tüzelési parancs kiadása előtt a rendszernek engedélyt kell kérnie az operátortól, ezzel minimalizálva a véletlen tüzelés kockázatát.

**Adatbiztonság:** A vezérlő szoftver és az operátor közötti kommunikáció titkosítva kell, hogy legyen, hogy megakadályozza a külső hozzáférést és a rendszer kompromittálását.

### Környezeti követelmények

A rendszernek különféle környezeti feltételek között is megbízhatóan kell működnie:

**Hőmérsékleti tűréshatár:** A rendszernek képesnek kell lennie normál működésre különböző hőmérsékleti körülmények között, amelyek tipikusan a beltéri használat során fordulnak elő.

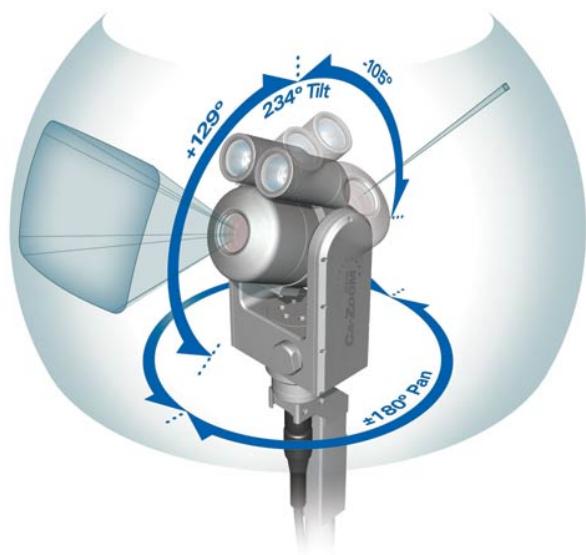
**Nedvesség és porállóság:** A rendszert úgy kell megtervezni, hogy ellenálljon a kisebb por- és nedvességterhelésnek, különösen, ha beltéri használatra is szükség van.

## 4. Mechanikai tervezés

A mechanikai tervezést egy kinematikai modell kidolgozásával kezdtem, majd a meglévő, kereskedelemben kapható alkatrészek méreteihez igazítva megterveztem a 3D CAD modellt. Ezután finomhangoltam a 3D nyomtatási technológiához megfelelően.

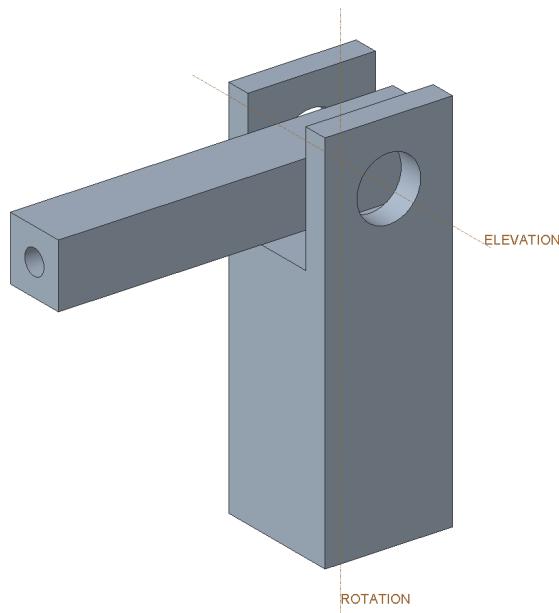
### 4.1. Kinematika

A rendszer kinematikai modellje hasonlatos a biztonsági kamerákhoz, aminek elnevezése Pan-Tilt-Zoom, röviden PTZ. Ennek lényege a 360 fokban elforgatható függőleges tengely, és egy általában korlátoltabb vízszintes tengely. Előnyük, hogy nagy sebességgel tudnak irányt változtatni, és nagy területet képesek beláttni. A zoom aspektus az én esetben nem fontos, hiszen nem a kamera a fő funkció. A 9. ábrán erről látható illusztráció.



9. ábra. PTZ kamera

A 2.1. bekezdésben vizsgált rendszerek is hasonlóképpen mozognak. Ezek közelebbi vizsgálata után elkezdtem kidolgozni a saját koncepciómat. Szemléltetésképpen készítettem a 10. ábrát. Az egyszerűség kedvéért a fegyver csövét egy síkba terveztem a pan és tilt tengelyekkel, ez megkönnyíti a későbbi számításokat.



10. ábra. Egyszerűsített kinematikai ábra

Ki kellett számolnom bizonyos geometriai megkötéseket, amelyek szükségesek a tervezéshez, illetve az alkatrészek kiválasztásához.

A torony szükséges fordulatszámát a következőképpen lehet kiszámolni:

$$rpm_{min} = \frac{v_t}{2 \cdot \pi \cdot r} = \frac{2.778 \text{ m/s}}{2 \cdot \pi \cdot \text{m}} = 0.442 \text{ 1/s} = 26.526 \text{ 1/min} \quad (1)$$

ahol:

$v_t$  A célpont sebessége a fegyvercsőre merőlegesen,

$r$  A távolság a célpont és a rendszer között

Meg lehet állapítani a torony mozgásának felbontását is, tehát hogy hány fokonként lehet állítani a mozgását.

$$\alpha_{min} = \arcsin\left(\frac{a}{2 \cdot r}\right) \cdot 2 = \arcsin\left(\frac{0.3 \text{ m}}{2 \cdot 10 \text{ m}}\right) \cdot 2 = 1.719^\circ \quad (2)$$

ahol:

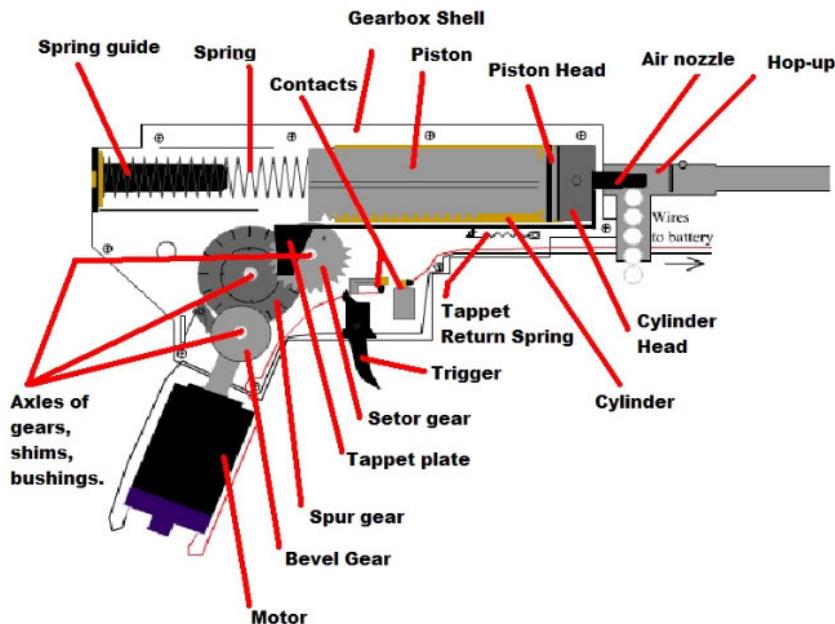
$a$  A célpont mérete

$r$  A távolság a célpont és a rendszer között

## 4.2. Mechanikai alkatrészek

### Elsütő mechanizmus

Az elsütő mechanizmusnak egy *Specna Arms M4*-ból kiszerelt gearbox-ot, illetve annak csövét és hop-up kamráját használtam. A gearbox működése során egy villanymotor több áttételen keresztül hátrahúz egy dugattyút, és azzal együtt a mögötte lévő rugót. Mindeközben a hop-up kamrában betöltött golyó, és megáll a csőben. Ahogy a részlegesen fogazott fogaskerék elengedi a dugattyút, azt a rugó előrelöki, ezáltal a dugattyú-kamrában nagy légyomás keletkezik, ami a fegyver csövén keresztül tud kiegyenlítődni. Folyamatos működés során a motor egymás után húzza fel és engedi el a dugattyút, így amíg van golyó a tárban képes tüzelni. A gearbox belső alkatrészei a 11. ábrán láthatóak.



11. ábra. V2 gearbox alkatrészei [11]

A hop-up kamrán belül még található egy gumi csúszófelület, amivel a kilőtt golyó perdületét lehet állítani, ezáltal pedig a fegyver effektív távolságát növelni.

A gearbox-on belül kellett alakítanom a működésen, hogy az előbb leírt folyamatos működést biztosítani tudjam. Az elsütőbillentyűt, a tűzkapcsolót és minden ehhez tartozó mechanikai elemet kiszereltem, illetve át-

huzaloztam. Erre azért volt szükség, mert különben csak a ravasz meghúzásával lehetett volna tüzelni, ami bonyolít a rendszer megvalósításán. A gearbox belső kialakítása a 12. ábrán látható.



12. ábra. A gearbox belseje[11]

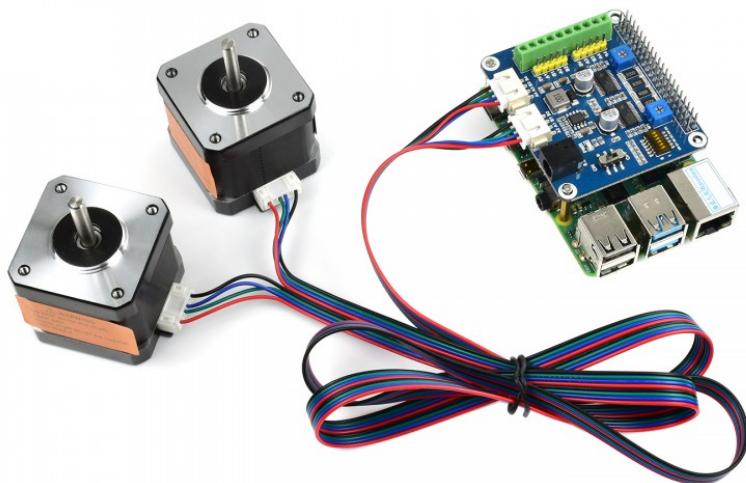
A fegyver gearbox-ot körülvevő alkatrészeiről le tudtam venni méreteket, ez alapján alakítottam ki később a 3D nyomtatott alkatrészeket. Így végéredményben egy magába zárt alkatrészem lett, amin habár mechanikus nem lehet állítani a tüzelés módját, de csupán két vezetékkel csatlakozik az elektronikához.

A gearbox áramfogyasztását illetően találtam méréseket az internet, ahol kifejezetten az én modelemet tesztelték. Az Itteni mérések alapján arra következtettem, hogy 15 A-re méretezni az áramkört megfelelő lehet. [12]

## Motorok

A projekthez kettő **NEMA-17** léptetőmotort használlok[13]. Ezek a léptetőmotorok egy széles körben alkalmazott típusa, amelyet főként precíziós mozgatási feladatokhoz használnak, például CNC gépekben, 3D nyomtatókban, robotikai alkalmazásokban és automatizálási rendszerekben. A NEMA-17 esetében a szám a motor elülső oldalának névleges méretét jelenti, amely 1.7 hüvelyk, azaz körülbelül 42,3 mm. A motorok a használt vezérlővel a 13. ábrán láthatóak.

A léptetőmotorok általánosan használt típusai közé tartoznak a szervomotorok és az aszinkron motorok. A léptetőmotorok azonban a mozgásukat



13. ábra. NEMA-17 léptetőmotorok a használt konfigurációban [17]

apró, egyenlő lépésekre osztják, így lehetővé téve a precíz pozícionálást és sebességszabályozást. A NEMA-17 léptetőmotor tipikusan kétfázisú bipoláris motor, amely négy vezetékes tekercseléssel rendelkezik. minden lépés során a motor egy adott szöggel fordul el, ami az adott motor típusától és felépítésétől függően tipikusan 1.8 fok, így teljes fordulat esetén 200 lépésre van szükség.

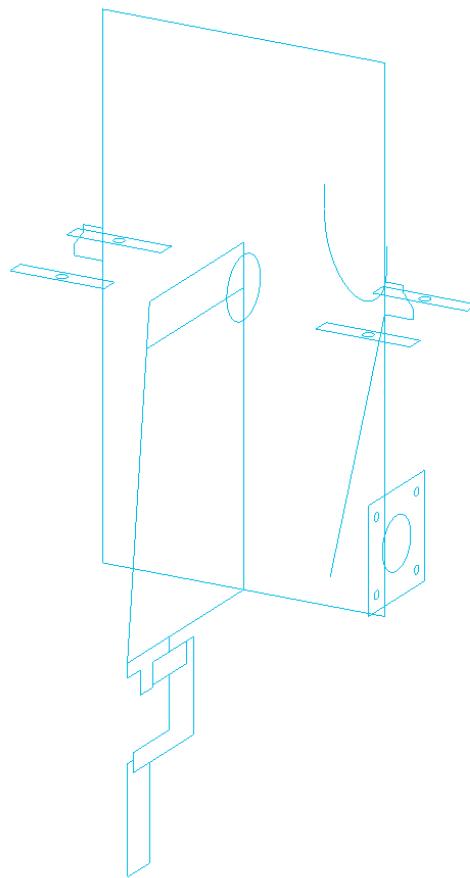
Az én esetemben használt léptetőmotorok lépésszöge 1.8 fok, bár ezt a motorvezérlőn lehet tovább osztani. A tartónyomatéka 0.4 Nm. Ezek a paraméterek 10-es áttételű fogaskerék-kapcsolattal megfelelőnek kell lenniük egy ilyen kis teljesítményű alkalmazásnál.

### 4.3. 3D tervezés, modellezés

A 3D tervezést Top-Down módszerrel végeztem, ez azt jelenti, hogy először az összeállítás szintjéről kezdem a tervezést, és egy úgynévezett skeleton modellbe veszem fel az egyes alkatrészek méreteit. Ezáltal tudom garantálni az egymáshoz való illeszkedést, valamint a változtatások könnyű implementálását.

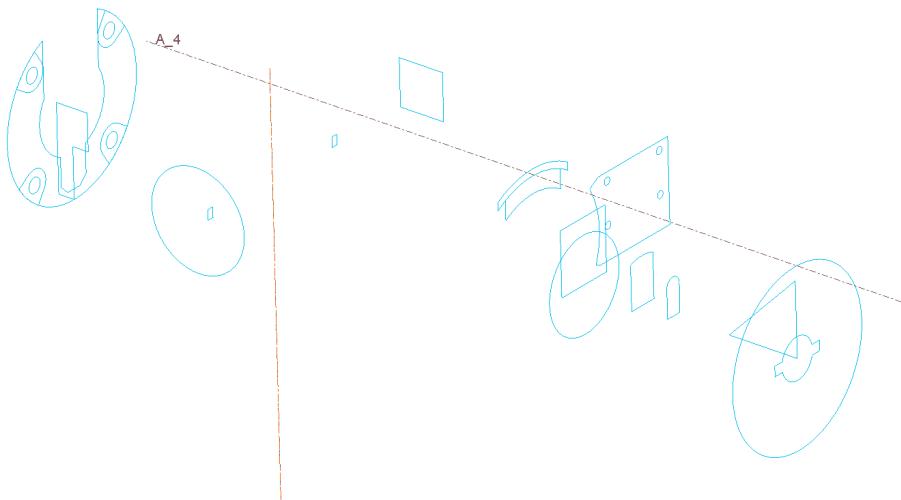
#### Skeleton modell

A modellezést a skeleton modellek megalkotásával kezdtem. A fő összeállításon belül két skeleton modellt hoztam létre, mivel maga a modell két alösszeállítása jól elkülöníthető egymástól.



14. ábra. DT-9999 skeleton modell

Az egyik (14.ábra) alkatrészei rendszerint a függőleges tengely körül szimmetrikusak, a másiké (15.ábra) pedig a fegyver csöve körül.



15. ábra. DT-4999 skeleton modell

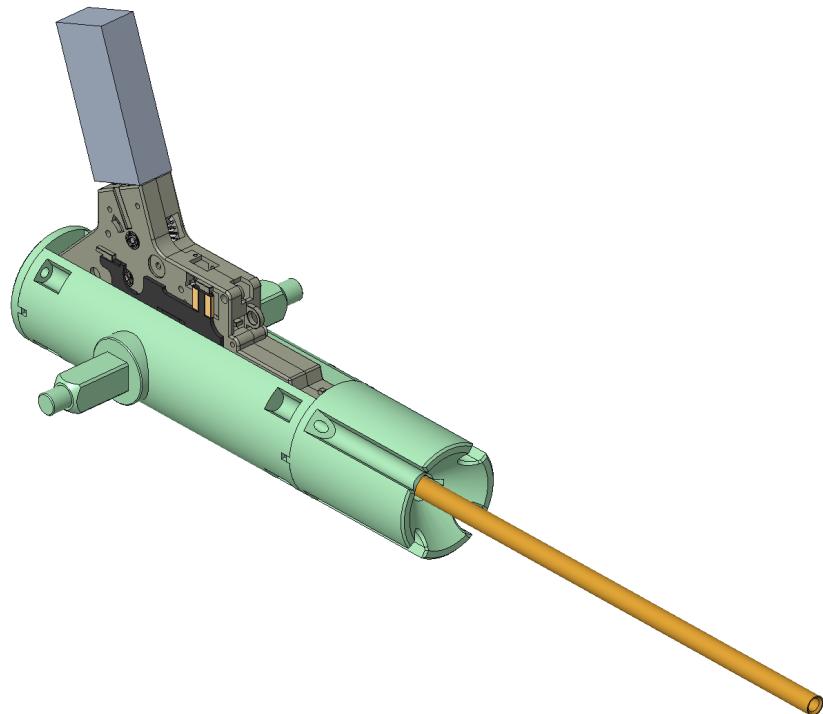
Az ábrákon láthatóak a vázlatok és tengelyek, amelyek alapján kialakítottam az egyes alkatrészeket. Sok méretet először csak hozzávetőlegesen vettet fel, pl. a torony magasságát. A Top-Down módszer előnye, hogy később ezeket könnyedén módosíthatom, és az alkatrészek frissítés után ugyanúgy fognak egymáshoz illeszkedni.

#### 4.3.1. Torony

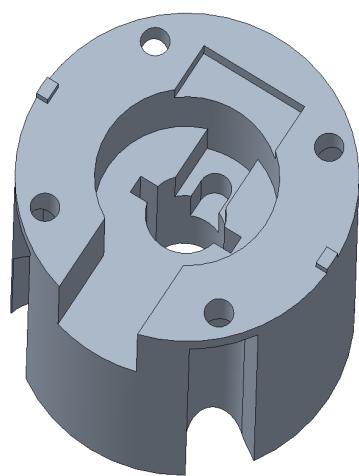
Elsőként a **gearbox tartó elemet** kezdtem el tervezni, mert maga a gearbox volt a tervezés elején az egyetlen elem, amiből tudtam következtetni a szükséges méretekre. Erről kép a 16. ábrán látható.

A gearbox ház 3 alkatrészből áll, egy központi elemből, valamint két fedélből a végein. A kialakítást a teljes airsoft fegyver alapján terveztem, hogy ugyanúgy álljon a gearbox mint az eredeti felhasználása során. A kritikusabb rész ebben az elemben a hop-up kamra körüli kialakítás volt, itt elég komplex volt a geometria, de a 3D nyomtatás lehetővé tette a megvalósítást. Ezt a 17. ábrán lehet látni.

A következő alösszeállítás a **tár** volt, amely a gearbox ház bal oldali tengelyére csatlakozik. Ennek kialakítása a 18. ábrán látható. A 6 mm-es golyók a zöld kupak nyílásán keresztül tölthetők a lila tárba. Alul és felett 1-1 gumigyűrű feszíti előre a zöld dugattyút, ami a tár kúpos végén

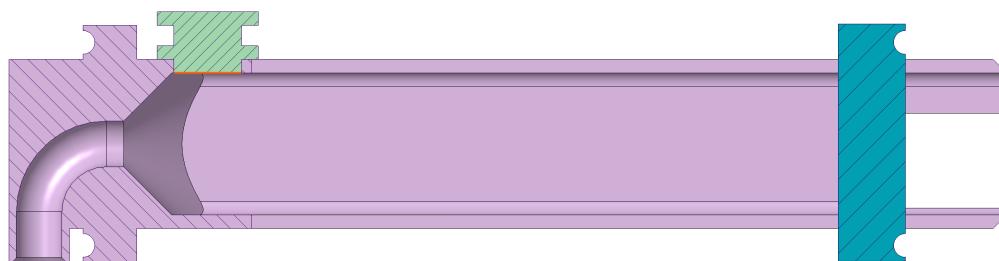


16. ábra. Gearbox tartó ház



17. ábra. Hop-Up kamra körüli elem

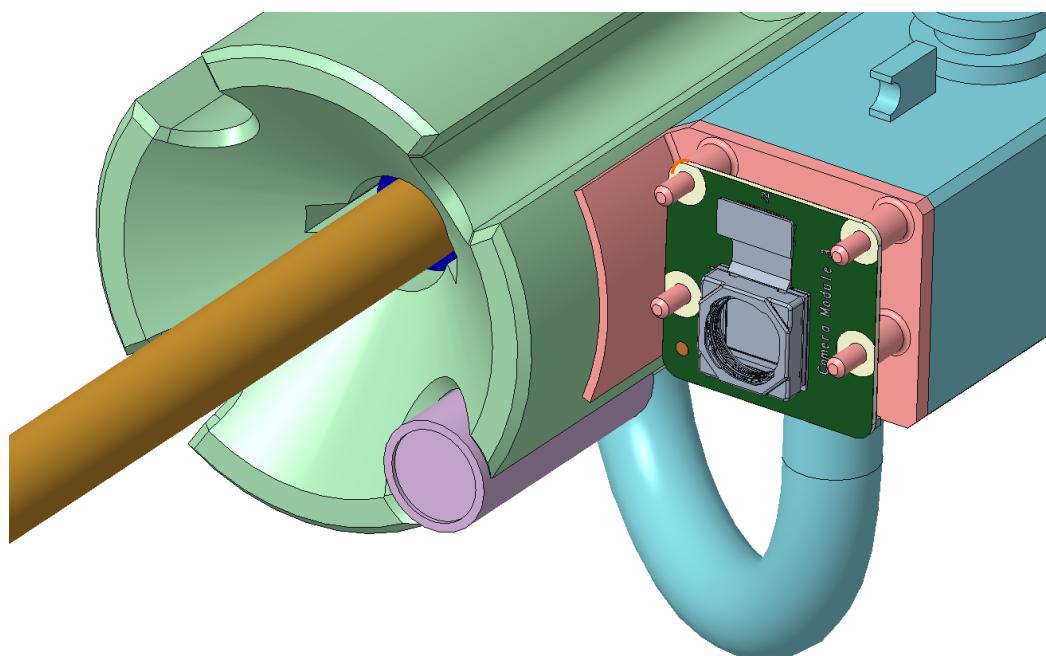
keresztül nyomja ki a golyókat. Így mechanikailag, plusz elektronika nélkül biztosítható a fegyver lövedékkel való ellátása, és szemmel is látható a tár töltöttségi szintje.



18. ábra. Tár

Ez a kialakítás végül nem volt sikeres, ugyanis a golyók nagyon könnyedén elakadtak, gyakorlatilag egyszer sem sikerült lőni vele. Újragondolás után egy hasonló megoldást használtam, ám a golyók egyesével sorakoznak a tárban.

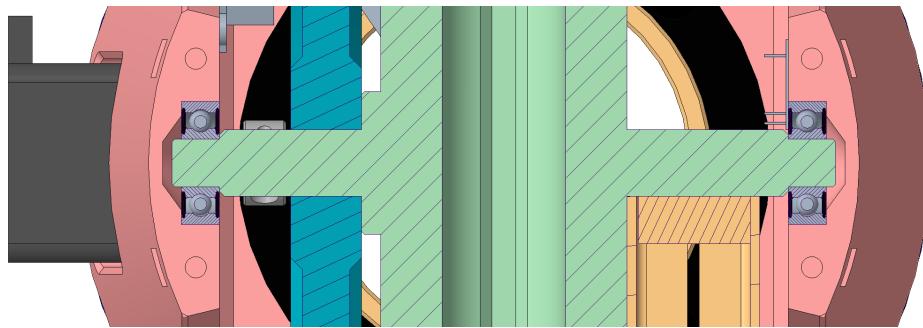
Végül a toronyra kerültek az elektronikai alkatrészek is, ezek a 19. ábrán láthatóak. A lila elem a lézer modul, a piros pedig a kamera konzol. Mind a kettő ragasztással kerül rögzítésre. Fontosnak tartottam, hogy ezek az elemek közvetlenül a fegyverhez legyenek rögzítve, ami a 3D nyomtathatóság miatt egy konzolt szükségesnek éreztem.



19. ábra. Kamera és lézer modul beépítés

### 4.3.2. Keret

Az úgynevezett **keret** volt a konstrukció legösszetettebb eleme, és egyben a legnagyobb térfogatú is. Fő funkciója a torony stabil tartása a csapágyakkal együtt. Ezentúl erre az elemre van rögzítve a vezérlőelektronika nagy része, a végálláskapcsolók és a vízszintes tengelyhez tartozó motor is. Ez az elem az alatta lévő alkatrészhez ragasztással lett rögzítve, hogy a szerelést egyszerűsítse. A csapágyak támasztása X elrendezésű, és az osztott "csapágház" miatt könnyen szerelhetőek. A csapágy elrendezése a



20. ábra. Vízszintes tengely csapágyainak elrendezése

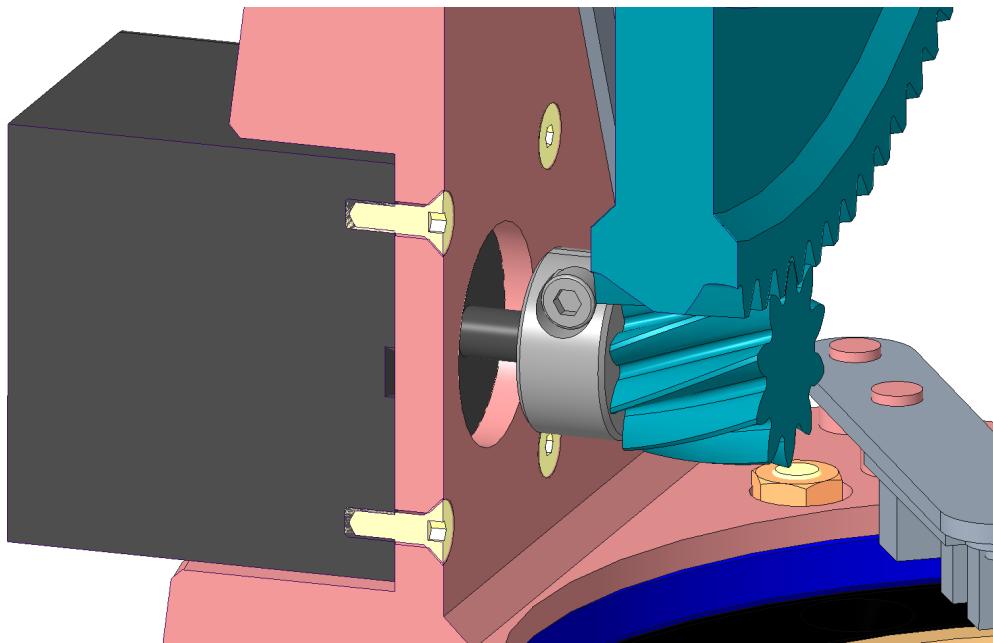
Kétséges rész volt a **motor** rögzítése, ugyanis nagy szükség van a pontos illeszkedésre. A nyomtatás irányára azonban a motor központosítására szánt furat pont merőleges, de szerencsére a nyomtató így is elég nagy pontosságot tudott elérni.

A kereten kaptak helyet a végálláskapcsolók is, amivel a rendszer indítás-kor kalibrálható. A egyik a vízszintes tengely nagy fogaskerekét, a másik pedig a függőleges tengely központi eleméhez viszonyít.

### 4.3.3. Függőleges tengely elemei

Utolsó lépésként megterveztem a függőleges tengely mozgatásáért felelős elemeket. ezek kialakítása a 24. ábrán látható. A narancssárga alakrész a központi elem, amelynek feladata a csapágy támasztása és a motor rögzítése. A motor vezetékei, valamint az elektronikából kilógó többi kábel ennek a közepén fut keresztül. A csapágy egyik fele ehhez az elemhez van rögzítve, a másik pedig az ábrán kékkel ábrázolt alkatrészhez. Erre a kék elemre került rögzítésre a belsőfogazású fogaskerék is.

A sárga alkatrész a talapzat része. Ez a legnagyobb kiterjedésű elem az egész modellben. Hogy elkerüljem a támaszanyag használatát, a csavarok fejeinek felfekvő felületeti egy külön alkatrész ragasztásával oldottam



21. ábra. Vízszintes tengely motor beépítése

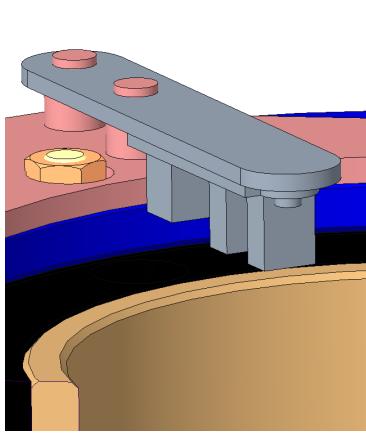
meg. Ez a talp 45 fokos belső felületére illeszkedik, és ragasztással kerül rögzítésre. A 4 oldalán kialakításra kerültek csatlakozó felületek, ahol különböző lábak helyezhetők az eszközre.

#### 4.3.4. Fogaskerekek

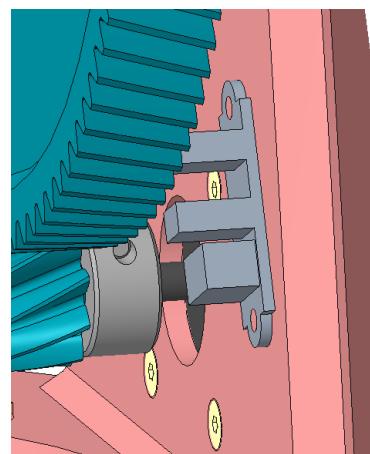
A prototípusban két fogaskerék kapcsolat kapott helyet. A függőleges tengely nagy fogaskereke belső fogazású, ezáltal el tudtam rejteni a motort a váz belséjében és kompaktabb lett a kialakítás, valamint valamivel stabilabb is. A kapcsolat áttétele 10, és ferde fogazású, ezáltal a futás csendesebb és egyenletesebb lett. A kapcsolat illusztrációja a 25. ábrán látható.

A vízszintes tengely nagy fogaskereke a gearbox ház tengelyére lett erősítve, és csak részlegesen lett kinyomtatva. Ennek egyszerű oka, hogy más geometria is blokkolja a mozgást ezekben a tartományokban, illetve a Pan-Tilt kinematika mozgása redundáns lenne, ha teljesen körbe tudna forogni. A kapcsolat a 25. ábrán látható.

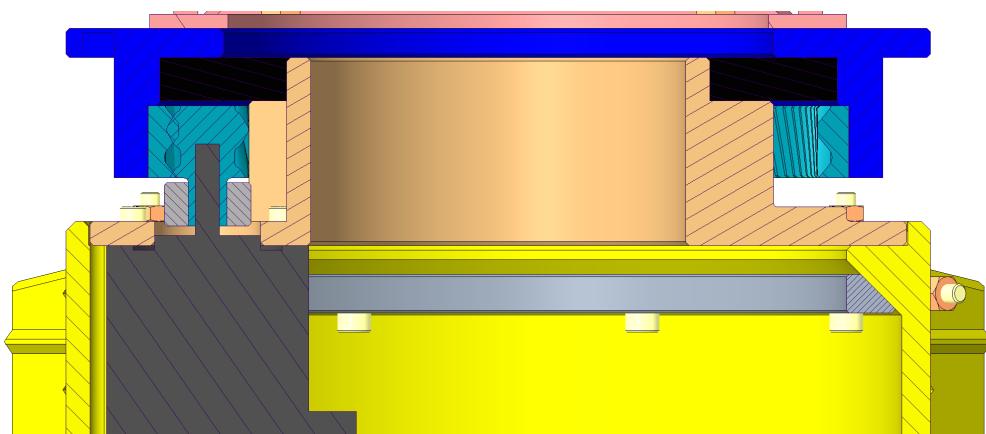
Mind a két fogaskerékpár kisebb közvetlenül a motor tengelyére lett rögzítve, egy acél rögzítőgyűrűvel. Ennek kialakítása a 24. ábra bal oldalán látható.



22. ábra. Pan végálláskapcsoló



23. ábra. Tilt végálláskapcsoló



24. ábra. Függőleges tengely elrendezése



25. ábra. Függőleges tengely fogaskerekek



26. ábra. Vízszintes tengely fogaskerekek

## 4.4. Gyártás és összeszerelés

### 4.4.1. Gyártás

Minden egyedi alkatrészt, amit a projekthez kellett gyártanom, 3D nyomtatással készítettem el. Ehhez egy **Creality Ender 3 Pro** típusú hobbinyomtatót használtam, amely a 27. ábrán látható. Mint a legtöbb ilyen árkategóriában lévő gép, ez is *FDM* technológiát használ. Az *FDM* lényege, hogy a nyomtató egy előre megadott terv alapján vékony műanyagszálat (filamentet) olvaszt meg egy forró extruderfejben (hotend), amelyet pontosan irányítanak a nyomtató XYZ tengelyei mentén. A műanyag szál olvasztott állapotban kerül a nyomtatóágyra, ahol gyorsan megszilárdul, így rétegenként építi fel az adott objektumot.

A használt nyomtató bizonyos tulajdonságai meghatározták a tervezett alkatrészek tulajdonságait:

- **Nyomtatási térfogat:**  $220 \times 220 \times 250$  mm, amely elég nagy volt a legtöbb általam tervezett elem számára.
- **Nyomtatási pontosság:** Az nyomtató rétegvastagsága 0,1–0,4 mm között állítható. Főleg a nyomtatott fogaskereknél volt fontos a megfelelő pontosság, de kielégítő eredmény született.
- **Használható filamentek:** A nyomtató különféle anyagokkal kompatibilis, beleértve a PLA-t, ABS-t, PETG-t és TPU-t. Én a PLA mellett döntöttem, ugyanis nem szükséges a pl. ABS által nyújtott minőség, ellenben a költséghatékonyság igen.

A modelleket **PTC Creo** szoftverben készítettem. Ez egy magas szintű CAD tervező program, amely képes a modelleket exportálni STEP és STL fájlként is. Az STL fájlokat ezután egy slicer programba betöltve tudtam a nyomtató által követhető G-kódra konvertálni. Ezután már csak meg kellett várni, amíg a gép elkészíti az adott alkatrészt, ami a legnagyobb elem esetén kb. 10 óra volt, a teljes prototípus kb. 50 óra volt.



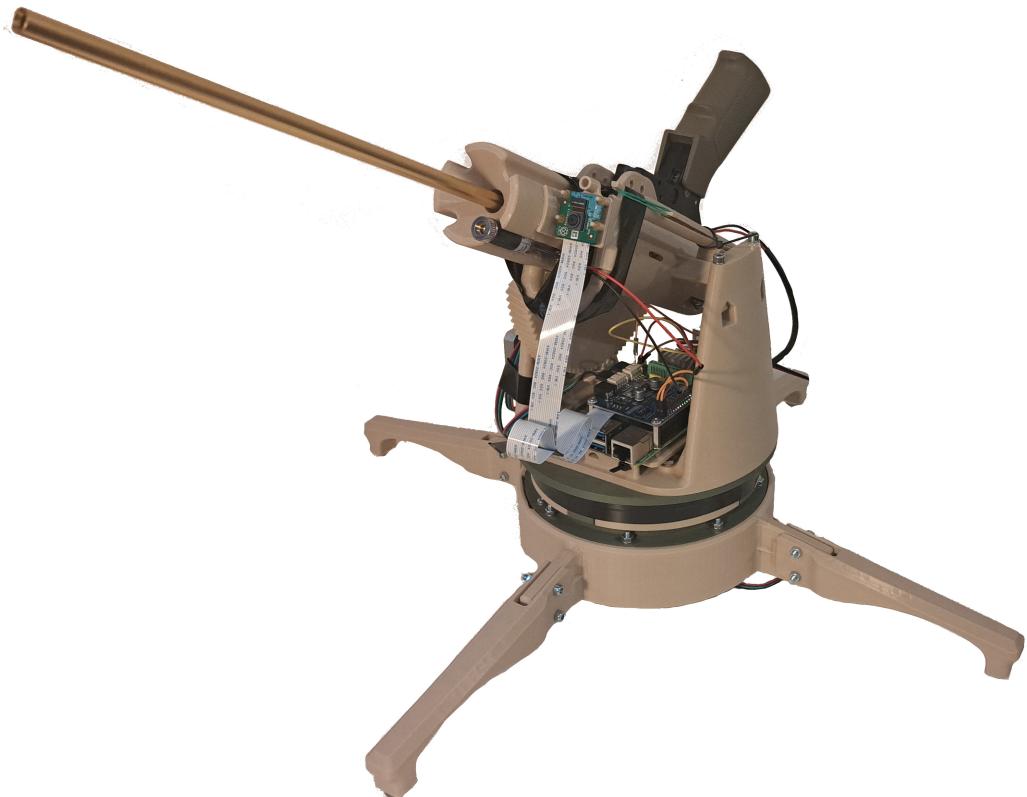
27. ábra. Creality Ender 3 Pro [14]

#### 4.4.2. Összeszerelés

Ebben a fejezetben leírom az összeszerelés menetét. Először az elektronikát tartó konzollal kezdtem.

1. Ráhúztam a motorok tengelyére a kisfogaskerekeket.
2. Rácsavaroztam a távtartókat a Raspberry Pi-ra, a Stepper Motor HAT-ra és a DCDC konverterre, majd minden hármat rácsavaroztam a 3D nyomtatott konzolra. Félretettem az összeállítást.
3. Összeraktam a torony elemeit, a gearboxot, a tartó elemét és a két végén lévő elemeket. Összecsavaroztam a tervezett furatokon, majd ezt az összeállítást is félretettem.
4. Összeállítottam a prototípus 4 lábat és az ezeket tartó hengert. A lábak talpára gumi elemeket ragasztottam, hogy minél stabilabban legyen a rendszer. Félretettem száradni.
5. Ezután összecsavaroztam a nagy csapágyhoz csatlakozó alkatrészeket, illetve a DT-1000 elemhez rögzítettem az egyik motort. A DT-2000 elemhez ragasztottam a hozzá tartozó fogaskereket.
6. A prototípus talpát összecsavaroztam az előző pontban lévő elemmel úgy, hogy a keretet is tartsák a csavarok.
7. A kerethez rögzítettem a megfelelő motort, beragasztottam a relét és a végálláskapcsolókat is. Kábelkötözővel rögzítettem az elektronikai elemek konzolját a kerethez.
8. A korábban összerakott gearbox összeállítás tengelyeire illesztettem az oda tartozó fogaskereket és a csapágyakat. Így ráhelyeztem a keretre, és a csapágházak felső felét hozzácsavaroztam a keretre.
9. Összeillesztettem a kamera konzolt, és felragasztottam a helyére. Ugyanígy tettem a tárral is. Beillesztettem a lézer modult a helyére.
10. Összekötöttem a kábeleket, csatlakozókat.
11. Csatlakoztattam a tápokat és az ethernet-kábelt.

Az elkészült eszközt a 28. ábrán lehet látni.

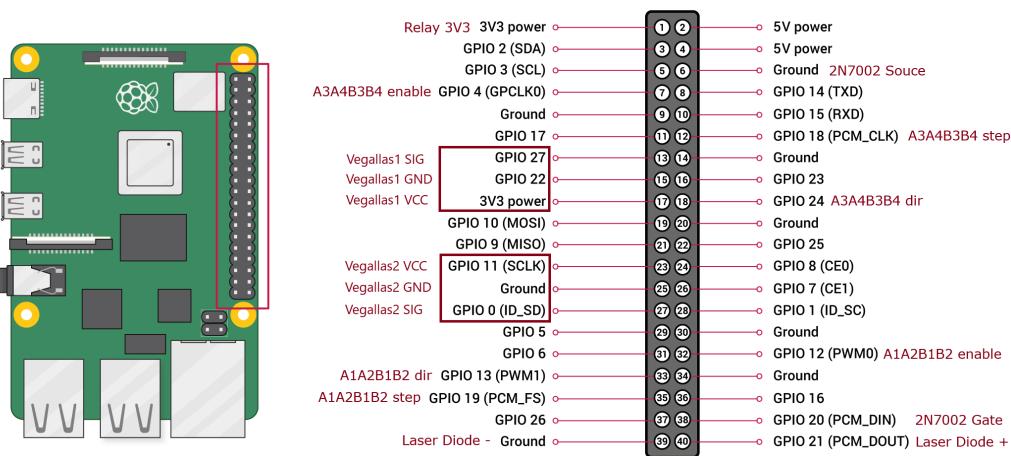


28. ábra. Összeállított prototípus

## 5. Hardvertervezés

### 5.1. Áramköri tervezés

A prototípus áramkörét próbáltam minél egyszerűbben megvalósítani. Alapvetően két részből áll, a Raspberry PI központú vezérlőáramkörből, ami felelős a szenzorokért és a Raspberry PI-ért, illetve egy nagyobb teljesítményű áramkör egy külön tápról, ami a gearbox működtetéséért felelős.

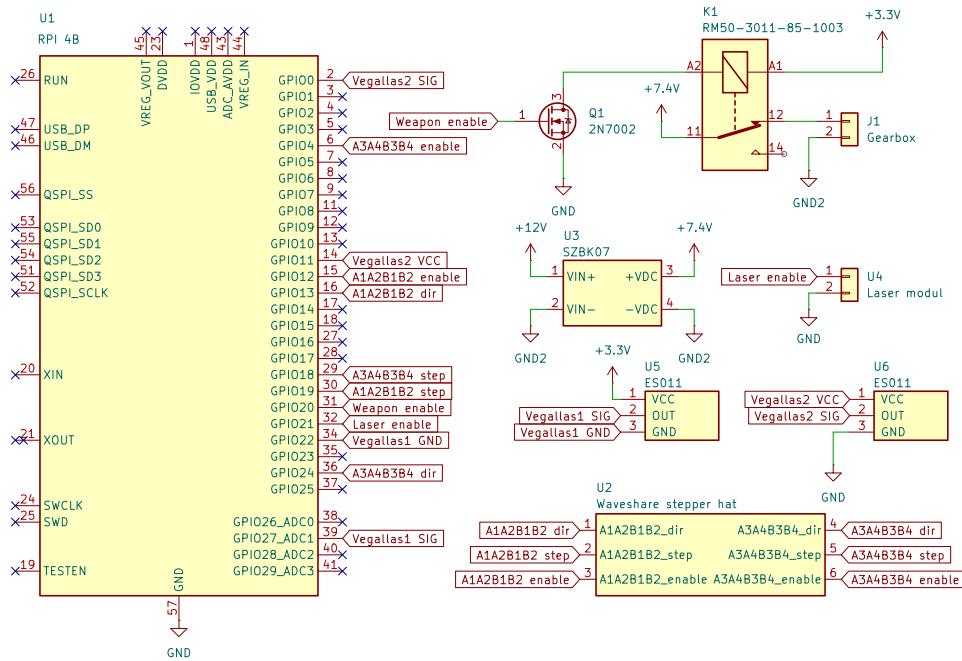


29. ábra. A Raspberry PI lábkiosztása [16]

A prototípus áramköre a 30 ábrán látható. A kereskedelemben kapható stepper motor HAT nagyban leegyszerűsítette a tervezés folyamatát, a segítségével nem kellett megterveznem a motorvezérlők áramkörét, illetve ezeknek az áramellátását. A 29.ábrán látható, motorokra vonatkozó jelölések szerinti GPIO lábakat a stepper motor HAT ugyan lefoglalja, de a kártya tetején lévő tüskesor ugyanúgy elérhető marad, így oda kell figyelni, hogy ezeket a lábakat ne használjam.

Ahogy az ábrán látható, a végálláskapcsolók is a GPIO tüskesorra lett csatlakoztatva. Az egyiknél a földet helyettesíti GPIO22 láb, a másiknál a 3.3 V-t helyettesíti a GPIO11. Azért kellett ezt a megoldást alkalmaznom, mert a kábel csatlakozója egyben van, nem különálló jumper tüskék.

A lézer közvetlenül a GPIO21 lábról működik. Eleinte aggódtam, hogy



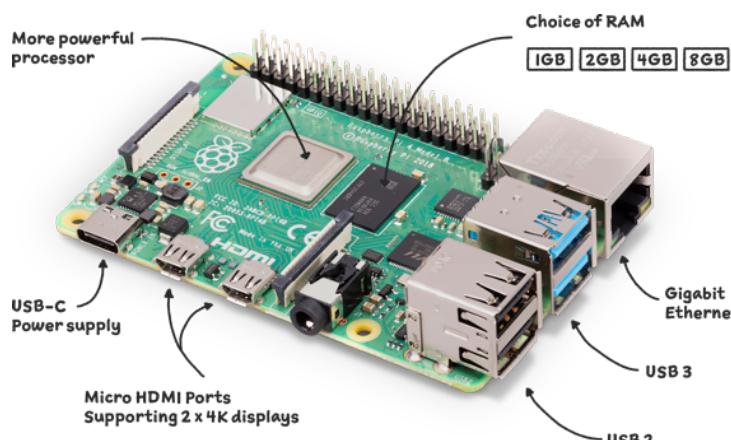
## 5.2. Elektronikai alkatrészek

### Mikrovezérlő

A mikrokontroller kiválasztásánál több opciót vizsgáltam, többek között az Arduino, az *STM-32* és a *Raspberry PI* modellek. A követelmények a következők voltak:

- Legyen könnyen beszerezhető, mind anyagilag, mind elérhetőség szempontjából
- Megfelelő teljesítmény gépi látás algoritmusokhoz
- Elegendő számú GPIO kimenet
- Python programozási nyelve támogatása
- Ethernet port

Azonban ahogy összehasonlítottam a különböző opciókat, körvonalazódott, hogy a *Raspberry PI* lesz a megfelelő megoldás. Az eszköz a 31. ábrán látható. Először is egy gyakorlati szempont szerint, mégpedig hogy egy *Raspberry PI 4B* modell eredetileg is volt a tulajdonomban 4GB rammal. A gyártó hivatalos fórumán több felhasználó szerint ez már elegendő összetettebb gépi látás algoritmusok futtatására is. [15]



31. ábra. Raspberry 4 model B [16]

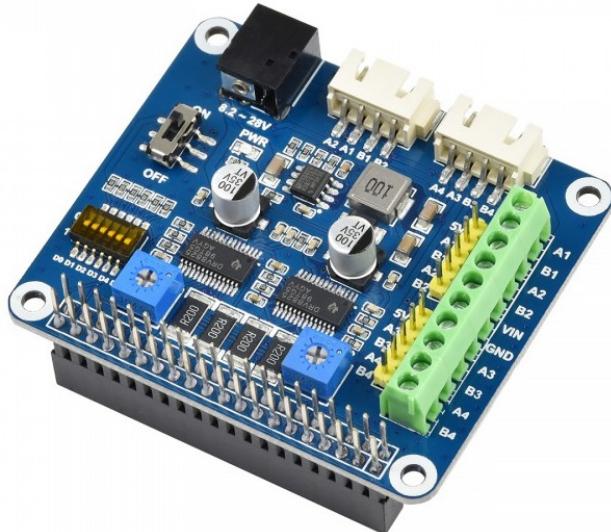
A kamera illesztése szintén különösen fontos a végtermék működése szempontjából, és ezen a területen magasan kiemelkedik a *Raspberry* a többi

mikrovezérlő közül, mivel magán a gyártó a termékcsaládon belül ajánl több jó minőségű, könnyen beszerezhető kamera modult, amik illesztése már kiforrott az összes Raspberry-hez.

Ezentúl a *Raspberry PI* erősebb, mint a fentebb említett mikrokontrolerek. Linux rendszert futtat, és lehet rajta Python nyelven programozni, ami gépi látás, automatizálás projekteknél hatalmas előny. Számos ki- és bemenete van, ráadásul támogatja a *Bluetooth*, *Wi-Fi*, és gigabites *Ethernet* kapcsolatokat is, nem beszélve a rengeteg bővítő "kabátról", amiket lehet kapni hozzá. Ezek a termék továbbfejlesztését nagyban könnyíti, ráadásul kevesebb munkát kell a nyomtatott áramkör tervezésébe tenni. [16]

### Stepper Motor HAT [17]

A NEMA-17 léptetőmotorok vezérlésére több megoldás is létezik, elterjedtek a A4988, TMC5160T és DRV8825 típusú vezérlők. Ezeket azonban vagy külön modulként lehet megvásárolni, vagy pedig SMD alkatrészként, ami mellé mindenkorban kell tervezni kiegészítő áramkört.



32. ábra. Waveshare Stepper HAT [17]

Azonban a Waveshare cég gyárt egy Raspberry Pi-kompatibilis modult, ami számomra rengeteg segítséget nyújtott. Ez egy külön áramkör, amit a Raspberry GPIO tüskesorára lehet illeszteni. Képes két *DRV8825* vezérlővel egyszerre kettő léptetőmotort vezérelni. A kártyán lévő kapcsolókkal

könnyen lehet állítani külön motoronként a microsteppelést is, amennyiben szükség van rá egészen 1/32-ig. A potméterekkel motoronként tudjuk állítani a maximális felvehető áramot, maximum 2.5 A-ig. A kártyán helyet kapott egy standard 5.5 mm-es csatlakozó is, amivel 8.2 V és 28 V közötti feszültséggel lehet táplálni az áramkört. Egy belső regulator chip segítségével a Raspberry PI-t is el lehet látni árammal, így egy tápegységgel tudom működtetni a teljes áramkört. A modulon több lehetőség is van csatlakoztatni a motorokat, akár sorkapoccsal, akár a léptetőmotorok szabvány csatlakozójával. Ezentúl lehetővé teszi a Raspberry PI GPIO lábainak elérését, csupán arra kell odafigyelni, melyikeket használja. Az eszközről a 32. ábrán látható illusztráció. Az eszköz el lett látva különböző biztonsági funkciókkal, pl. túlárám védelemmel, magas hőmérséklet elleni védelemmel, illetve alulfeszültség lockout-tal.

A gyártó szintén elérhetővé tett illesztőszoftvereket több eszközre, több motortípusra, amit a későbbiekben én is tudtam használni.

### Táp

A prototípust két tápegységről működtetem. Az egyik a Raspberry PI, a léptetőmotorok és a szenzorok áramellátásáért, ez egy standard 5.5 mm-es csatlakozóval ellátott laptop töltő. 14.5 V tápfeszültséget és maximum 5 A áramot képes biztosítani, amely bőven elegendő a vezérlőelektronika ellátására.



33. ábra. HP ps-3701-1

A másik táp egy *HP ps-3701-1* 12 V-os redundáns szervertáp 725 W teljesítménnyel.

### Kamera

A rendszer által használt kamera egy *Raspberry Pi Camera Module 2* volt.[18] Ez a modul 8 MP-es Sony IMX219 szenzorral rendelkezik, amivel 3266 x 2450 felbontású állóképet tud készíteni. Képes Full HD videót felvenni 30 fps-el, HD-t 60 fps-el, vagy 640x480 felbontást 90 fps-el. A modul a 34. ábrán látható.



34. ábra. Raspberry Pi Camera Module 2

A kamera befoglaló mérete szerelési furatai és CSI portja megegyezik a többi Raspberry kamerával, így a továbbfejlesztés esetén könnyedén lehet őket cserálni.

### Végálláskapcsoló

A prototípus bekapcsolásánál fontos, hogy beálljon egy kezdőpozícióba, és ahhoz képest tudjuk viszonyítani a mozgását. Ezt én a két tengelyen 1-1 végálláskapcsolóval oldottam meg. Ezek a modulok könnyen szerelhetőek a 3D nyomtatott vázra, és az alapján adnak jelet, valami van-e az optokapu között. A Raspberry GPIO lábaira könnyedén lehet őket bekötni jumper kábelek segítségével. A szenzorok a 35. ábrán láthatóak.



35. ábra. Végálláskapcsoló

### 5.3. Gyártás, forrasztás

A projekthez egyedi nyomtatott áramkört nem kellett gyártatni, azonban egy kis forrasztást igényelt. A gearbox táp folyamatos működéséhez az alább látható módon kellett egy huzallal összeforrasztanom a padokat. Látható a 12V és GND-hez csatlakozó kábelek is, amiket szintén beforraszttam a megfelelő helyre. Ezen a területen van lehetőség a továbbfejlesztésre, például egy 3D nyomtatott elemmel.

Ezentúl a kábeleket kellett forrasztanom, illetve a relé és a hozzá tartozó tranzisztor áramkörét. Ezek az alábbi képen láthatóak. Azokat a kábeleket, ahol nagyobb áram folyhat, és szerettem volna, hogy szerelhetőek legyenek, ott WAGO 221 összekötőkkel oldottam meg a csatlakozást.

## 6. Szoftverfejlesztés

A prototípus szoftverét kettéválasztottam a két működési mód szerint, az egyik a manuális vezérlést valósítja meg, a másik az automatikust. A kettő között lényeges különbségek vannak a prioritásokban. A kézi vezérlésnél a legfontosabb a valós idejű megjelenítés, és a minél gyorsabb reagálás a felhasználói parancsokra. Az automata működésnél a megjelenítés kevésbé fontos, viszont a képfelismerés és ez által a célpont követése kiemelkedő fontosságú.

### 6.1. Használt Python modulok, eszközök

#### OpenCV könyvtár [19]

Az **OpenCV** (Open Source Computer Vision Library) egy nyílt forráskódú könyvtár, amelyet eredetileg az Intel fejlesztett ki, és amelyet széles körben használnak a számítógépes látás (computer vision) és képfeldolgozás területén. Az OpenCV rendkívül sokoldalú eszközökkel biztosít a képekkel és videókkal kapcsolatos különféle feladatok megoldásához, amelyet elsősorban C++-ban fejlesztettek ki, de támogat Python, Java, és MATLAB interfésekkel is. A Python interfész különösen népszerű, mivel egyszerűsíti az OpenCV használatát gépi tanulási és mesterséges intelligencia-alkalmazásokban. A fő funkciói közé tartozik a képfeldolgozás, tárgy- és arcérzékelés, mozgáskövetés, valamint a 3D-s modellezés és képanalízis.

Az OpenCV architektúrája moduláris felépítésű, ami lehetővé teszi a különböző funkciók rugalmas használatát. A modulok között megtalálhatók a képfeldolgozási, objektumfelismerési, gépi tanulási és 3D modellezési könyvtárak. A Pythonban a `cv2` könyvtárként importálható OpenCV API biztosítja a különböző funkciók egyszerű elérését, így akár néhány sor kóddal is gyors eredmény érhető el.

Többféle képformátumot támogat, például JPEG, PNG és BMP, és rendelkezik valós idejű video- és képadatok betöltésére szolgáló funkciókkal is, például a `VideoCapture` objektummal. Ez utóbbi lehetővé teszi, hogy közvetlenül kamerákból vagy videofájlokból dolgozzunk.

Az **OpenCV** használata igen elterjedt az iparban és a kutatásban is, többek között autonóm járművek, biztonsági rendszerek, gyártósori ellenőrző rendszerek, orvosi képalkotás és mobilalkalmazások terén. A könyvtár

rugalmassága és széleskörű támogatása lehetővé teszi, hogy számos programozási és mérnöki területen is alkalmazható legyen, ahol képfeldolgozásra és gépi látásra van szükség.

### Socket modul [20]

Python `socket` modulja alacsony szintű hozzáférést biztosít a hálózati kommunikációhoz, lehetővé téve különböző típusú hálózati kapcsolatok létrehozását, beleértve az általam használt TCP-t és az UDP-t.

A **TCP** kapcsolat-orientált protokoll, amely biztosítja az adatok érkezési sorrendjét és helyességét. A `socket.SOCK_STREAM` típust használjuk, ha TCP kapcsolatot szeretnénk létrehozni. A TCP kapcsolatok esetében a `connect()` függvény a kapcsolat kezdeti lépése, és a kliens-szerver kommunikáció folyamatosan fennmarad.

Az **UDP** kapcsolatnélküli protokoll, amely nem garantálja az adatok sorrendjét és integritását, viszont gyorsabb, mivel nincs szükség kapcsolatra. Az UDP kapcsolathoz a `socket.SOCK_DGRAM` típust használjuk, így a kliens és a szerver bármikor küldhet és fogadhat adatokat. Ideális valós idejű alkalmazásokhoz, ahol nem kritikus minden csomag megérkezése (pl. video streaming).

Mindkét típusnál a `send()`, `sendto()`, `recv()`, és `recvfrom()` függvények használhatók az adatok küldésére és fogadására.

### Multiprocessing modul [21]

A **multiprocessing** modul lehetővé teszi a Python számára, hogy párhuzamosan több folyamatot indítsan el, ami segít a CPU-erőforrások jobb kihasználásában. A modul az operációs rendszer által kezelt különálló folyamatokat hoz létre, így azok külön memóriaterülettel rendelkeznek, és jobban teljesítenek többmagos processzorokon, mint a *threading* modul.

A `Process` osztály segítségével indíthatunk el új folyamatokat, amelyek egymástól függetlenül végrehajtanak egy adott funkciót. A modul támogatja az eseménykezelést, kommunikációs csatornákat (pl. `Queue`, `Pipe`) és szinkronizálást (pl. `Event`, `Lock`), amelyek segítségével a

folyamatok egymással adatokat cserélhetnek és szinkronizálhatják működésüket.

### Pygame modul [22]

A `pygame` egy multimédiás modul, amely elsősorban 2D játékok fejlesztésére alkalmas, de egyéb alkalmazásokkor is lehetőséget ad bemenetek kezeléséhez.

A `pygame.event.get()` függvény segítségével olvashatjuk le az egér- és billentyűzet-eseményeket, így például kattintások, gombnyomások és egérmozgások lekérdezése egyszerű. Az interaktív elemek kirajzolása és mozgatása egyszerű a `pygame.display.update()` és `pygame.draw` függvényekkel, amelyek lehetővé teszik grafikus objektumok megjelenítését a képernyőn. Magában foglalja a számítógépes grafikákat, a hang- és programkönyvtárakat, amiket a Python programozási nyelvre fejlesztettek ki.

### Pickle modul [23]

A `pickle` modul sorosítást és deserializációt tesz lehetővé, azaz a Python-objektumok bináris formátumba alakítását és visszaalakítását.

A `pickle.dumps()` segítségével a Python-objektumokat bináris formátumba alakíthatjuk, amely lehetővé teszi azok egyszerű tárolását vagy átvitelét, például hálózaton keresztül. A `pickle.loads()` a bináris formátumú adatokat visszaalakítja eredeti Python-objektummá. A pickle használata előnyös lehet akkor, ha komplex adatszerkezeteket szeretnénk fájlban tárolni vagy hálózaton keresztül továbbítani.

## 6.2. Gépi látás

A célpontfelismerés módszere volt az automata működés legfontosabb eleme. Elsősorban az OpenCV könyvtár különböző lehetőségeit vizsgáltam, ugyanis ez a legjobban kiforrott modul, amely könnyedén használható python programokban.

### 6.2.1. Template matching [24]

A **Template Matching** (sablonillesztés) egy egyszerű, de hatékony módszer, amelyet statikus képekben keresett minták (sablonok) azonosítására használnak. Ez a módszer összehasonlít egy kisebb, előre meghatározott képrészletet (a sablont) a nagyobb képen található területekkel, és megkeresi a legjobban illeszkedő pozíciót.

**Működés:** Az OpenCV `cv2.matchTemplate()` függvényével kivitelezhető, amely végigfuttatja a sablont a célképen, és minden pixelpozícióján egy hasonlósági pontszámot generál. A legmagasabb értékű pontszám mutatja az illeszkedés helyét.

**Előnyei:** Gyors és könnyen implementálható, különösen ismert méretű és orientációjú objektumok azonosításához.

**Korlátozások:** Gyengén teljesít forgatott, méretarányosan eltérő vagy eltorzult objektumoknál, valamint változó fényviszonyok mellett.

### 6.2.2. Feature Matching [25]

A **Feature Matching** (jellemzőpont-illesztés) egy kifinomultabb módszer a képjellemzők összehasonlítására és egyeztetésére. A módszer a két képben található pontokat (feature-ket) párosítja, ezáltal lehetővé teszi forgatásokkal, elforgatásokkal és méretarányokkal szembeni robusztus egyeztetést.

**Működés:** Először meghatározzuk a képen lévő fontos jellemzőpontokat (például élek, sarkok), amelyeket különböző detektorok (például SIFT, SURF vagy ORB) segítségével találhatunk meg. Ezután az `cv2.BFMatcher()` (Brute Force Matcher) vagy a `cv2.FlannBasedMatcher()` függvénytellyel azonosítjuk a jellemzőpontokat.

**Előnyei:** Robusztus a forgatásokkal, skálázással és elmozdulásokkal szemben, így alkalmas tárgyak nyomon követésére és objektumok felismerésére

különböző nézetekből.

**Korlátozások:** A nagy számítási igényű algoritmusok miatt lassabb lehet, különösen nagy méretű képeken.

### 6.2.3. Target Tracking [26]

A Target Tracking (célkövetés) feladata egy mozgó objektum követése egy videófolyamon vagy valós idejű képforráson keresztül. Az OpenCV több nyomon követési algoritmust is biztosít, amelyek segítenek abban, hogy a kiválasztott objektumot stabilan követni tudjuk, még akkor is, ha változik a pozíciója vagy a környezeti fény.

Algoritmusok:

- Mean Shift: A `cv2.meanShift()` függvénytel elérhető algoritmus egy adott objektum körüli eloszlás csúcsértékeit keresi, és folyamatosan frissíti a helyét.
- CamShift (Continuously Adaptive Mean Shift): A `cv2.CamShift()` algoritmus a Mean Shift továbbfejlesztett változata, amely figyelembe veszi az objektum méretének és alakjának változásait.
- Modern nyomkövetési algoritmusok: Az OpenCV 3.x-es és újabb verziói lehetővé teszik algoritmusok használatát, mint a KLT (Kanade-Lucas-Tomasi) nyomkövető, a CSRT és a MedianFlow, amelyek kifejezetten valós idejű és pontos követést biztosítanak.

**Előnyei:** Alkalmas valós idejű alkalmazásokhoz, mint például a videó megfigyelés vagy az autonóm rendszerek, ahol a cél folyamatosan mozgásban van.

**Korlátozások:** Bizonyos algoritmusok hajlamosak elveszteni a célpontot gyors mozgásoknál vagy hirtelen pozícióváltásoknál.

### 6.2.4. Haar cascade [27]

A Haar Cascade módszer az egyik legelterjedtebb technika az objektum-felismerésre, különösen az arcfelismerésben. A gépi tanuláson alapuló módszer képes egy képben vagy videófolyamon objektumokat megtalálni azáltal, hogy mintákat ismer fel az előképzett adatok alapján.

**Működés:** A Haar-cascade egy többlépcsős osztályozó, amelyet sok pozitív és negatív mintából tanítanak be. A `cv2.CascadeClassifier()` osztály

segítségével előre tanított modelleket, például arcokat, szemeket vagy különböző objektumokat kereshetünk.

**Gépi tanulás:** A módszer előnye, hogy nagyon hatékonyan képes felismerni az objektumokat. A gépi tanulás során a kimeneti modell jellemzőit úgy optimalizálják, hogy gyorsan tudja felismerni a keresett objektumokat még zajos vagy eltérő körülmények között is.

**Előnyei:** A Haar-cascade módszer jól használható olyan alkalmazásokhoz, ahol előre meghatározott objektumokat (például arcokat) kell azonosítani. Nagy előnye, hogy valós időben is képes futni.

**Korlátozások:** Hajlamos a pontatlan felismerésre, ha az objektumok perspektívája, megvilágítása vagy pozíciója jelentősen eltér az előképzett mintáktól. Alternatív megoldásként mélytanulási módszerek (például konvolúciós neurális hálózatok) használhatók az ilyen problémák javítására.

### 6.3. A szoftver működése

A rendszer két alapvető részből áll: egy számítógépből (PC), amely felhasználói interfésként szolgál az operátor számára, és egy Raspberry Pi-ból, amely a fogyverrendszer vezérlését és képátvitelét végzi. A két eszközön futó program között két folyamat kommunikál, két különböző porton: egy UDP protokoll alapú adatátvitel a videó küldéséhez, illetve egy TCP alapú a vezérlőparancsok fogadásához.

#### 6.3.1. A PC-n futó program működése

A PC-s kód két külön folyamatot indít: egy videó-fogadó folyamatot és egy vezérlő-küldő folyamatot, melyek egy multiprocessing-eseményjelző (`stop_event`) segítségével állnak meg szükség esetén.

##### Videó-fogadó folyamat

A videó-fogadó folyamat (`video_receiver`) UDP-n keresztül kapja a képkockákat, amelyek a Raspberry Pi-ról érkeznek csomagokra bontva. A folyamat összegyűjti ezeket a csomagokat, majd a kép adatokat újraegyesíti és megjeleníti a felhasználó számára. Az UDP protokoll nagy sebességet tud elérni, de nem garantált a célba érés. A videó esetében azonban ez nem probléma, ha elveszik egy képkocka attól még értelmezhető marad a videó. A folyamat a következőképpen működik:

- A folyamat egy UDP socketet (`video_socket`) hoz létre a megadott porton.
- Egy ciklusban fogadja a képkockák csomagjait, minden egyes csomag egy fejlécet és a kép adatrészleteit tartalmazza.
- A fejléc alapján a folyamat azonosítja a csomagokhoz tartozó képkockát és azok sorrendjét. A bufferben gyűjtött csomagokat az összes részlet megérkezése után egyesíti.
- Az egyesített képkockát visszafejti (unpickling és JPEG dekódolás), majd a képre helyez egy célkeresztet a `cv2.line()` függvény alkalmazásával.
- A teljes képkockát behelyezi a `frame_queue` sorba, a `.put()` módussal.

### Vezérlés-küldő folyamat

A vezérlés-küldő folyamat ( `control_sender` ) a felhasználó irányítási parancsait továbbítja a Raspberry Pi felé. A folyamat a Pygame könyvtár segítségével észleli az egér- és billentyűeseményeket, majd a megfelelő parancsokat TCP csomag formájában küldi a Raspberry Pi irányába. Ezért felelős a HUD megjelenítéséért, amiről a felhasználó le tudja olvasni a prototípus állapotát.

#### HUD:

A felhasználó számára fontos, hogy minden információt le tudjon egyből olvasni a fegyver működéséről, így ezeket a PyGame ablakon megjelenítettem. Itt látható:

- Használható parancsok és a hozzájuk rendelt gombok
- Milyen módban működik a rendszer
- Biztosítva van-e a rendszer
- Működik-e a lézer

A PyGame ablak háttere pedig az éppen utolsó képkocka, amit a videofogadó folyamat a `frame_queue` -ba tett. A vizuális megjelenítés mellett a folyamat feladata a parancsok küldése, ezek a következők lehetnek:

- Az egér mozgása esetén `MOVE` parancs az X és Y elmozdulási adatokkal, így irányítva a fegyver mechanikai elmozdulását.
- Az bal egérgomb lenyomásakor (`SHOOT_START`, a felengedésekor `SHOOT_STOP`) parancsot adhatunk ki, értelemszerűen a tüzelés megkezdésére és befejezésére.
- A billentyűzet gombjaival a lézert, (`LASERTOGGLE`) a biztosítást (`SAFETY`), illetve a működési módokat lehet állítani (`MANUALMODE` és `AUTOMODE`).
- Amikor a felhasználó az `ESCAPE` gombot megnyomja, a folyamat küld egy leállítási parancsot (`STOP`), majd befejezi működését.

A program a működési módtól függetlenül működik, mindig ugyanazokat az üzeneteket küldi el. A kliens programtól függ, hogyan értelmezi azokat.

### 6.3.2. A Raspberry Pi-n futó program működése

A Raspberry Pi-n több folyamat is fut: Egy videó-küldő, egy vezérlés-fogadó, egy célpontfelismerő, illetve egy motormozgató.

**Célfelismerő folyamat** A `target_detection()` folyamat felelős az arcok felismeréséért, és a helyzetük meghatározásához.

- A `cv2.CascadeClassifier()` függvénytel betölti az előre betanított `haarcascade_frontalface_default.xml` modellt
- A `frame_queue.get()` metódussal kiveszi az utolsó képkockát a sorból
- A `cv2.detectMultiScale()` megtalálja az arcokat a képkockán
- Méret szerint sorba rendezi az arcokat, és kiszámolja a legnagyobb pozíóját
- Betölti a pozíciót a sorba a `pos_queue.put` metódussal

Ez a folyamat csak akkor fut, hogyha az `automode_event()` flag be van állítva, tehát az eszköz automatikusan működik.

**Vezérlés-fogadó folyamat** A `control_listener` folyamat először, a rendszer indításakor elvégzi a fegyver pozíójának beállítását. A motorokat addig forgatja, amíg a végálláskapcsolók nem adnak jelet. Ezután pedig, miután tudjuk a pontos pozíció, visszaforgatja a fegyvert középre, és nullázza a `POSX` és `POSY` globális változókat. Ezután pedig TCP-n keresztül fogadja a PC-ről érkező irányítási parancsokat, és a megfelelő fizikai komponenseket működteti, majd a rendszer leállításakor újra középre állítja a fegyvert. A következő parancsokat tudja fogadni, attól függően, hogy melyik módban működik a prototípus.

- A parancs típusa alapján (`MOVE`, `SHOOT_START`, `SHOOT_STOP`, `LASERTOGGLE`, `STOP`) különböző műveleteket hajt végre.
- `MOVE` parancs esetén az X és Y elmozdulási értékek alapján irányítja a két DRV8825 motorvezérlő áramkört, így mozgatva a fegyvert az elvárt irányba.

- A lövésvezérlés (`SHOOT_START` és `SHOOT_STOP`) esetén aktiválja vagy deaktiválja a fegyver LED-jét.
- A lézerkapcsolás (`LASERTOGGLE`) a lézerdiódát ki- vagy bekapcsolja.
- Amennyiben `STOP` parancs érkezik, a folyamat leállítja a működést.
- `STOP` parancs esetén az `stop_event`-et beállítja, ezzel leállítva az összes folyamatot
- `AUTOMODE` parancs esetén az `automode_event`-et igazra állítja
- `MANUALMODE` parancs esetén az `automode_event`-et hamisra állítja
- `MOVE` parancs esetén az X és Y elmozdulási értékek alapján irányítja a két motorvezérlőt, így mozgatva a fegyvert a megfelelő irányba.
- A (`SHOOT_START` és `SHOOT_STOP`) esetén aktiválja vagy deaktiválja a fegyvert. Aktiválni csak akkor lehet, ha a `safety_on` hamis.
- A (`LASERTOGGLE`) a lézerdiódát ki- vagy bekapcsolja.
- Amennyiben `SAFETY` parancs érkezik, beállítja a `safety_on` flag-et.

A végtelen ciklus ellenőrzi a `stop_event` flag-et, és ha igaz lesz, akkor kilép belőle. Ekkor a `TurnStep()` fügvénnyel addig lép, amíg középen nem lesz a fegyver, valamint a lézert is lekapcsolja.

**Videó-küldő folyamat** A videó-küldő folyamat (`video_sender`) a Raspberry Pi kamerájából érkező képet szerzi meg és osztja meg a PC-vel UDP kapcsolaton keresztül.

- A kamerakép feldolgozása során a kód rögzíti az egyes képkockákat, amelyeket JPEG formátumba kódol és pickle segítségével tömörít.

- Amennyiben az `automode_event` flag igaz, akkor beteszi a legutolsó képkockát a `frame_queue` -ba, és a `pos_queue` legutolsó elemei alapján húz egy téglalapot illeszt a képkockára, ezzel jelezve a talált arcot.
- Az adatok nagy mennyiségét figyelembe véve a képkockákat kisebb csomagokra bontja, amelyekhez megfelelő fejlécek tartoznak.
- A fejlécekben megtalálható a csomag azonosítója, amely segít a PC-n futó videó-fogadó folyamatnak a csomagok megfelelő sorrendbe állításában és összerakásában.
- A csomagokat a megadott IP-címre és portra küldi, és minden új képkockához új azonosítót rendel.

**Motormozgató folyamat** A `motor_motion()` folyamat azért felelős, hogy automata működés során mozgassa a motorokat, a kamera képe alapján. Azért gondoltam, hogy jobb így külön kezelni, mert az automata és kézi vezérlés mozgásának nagyon különböző a logikája. A folyamat működése a következőképpen működik:

- Kiolvassa a legutolsó pozíciót a `pos_queue` -ból, majd a pozícióból és a középpont különbségéből kiszámítja a távolságot.
- Amennyiben a fegyver pozíciója a mozgásterületen belül van, a megfelelő irányba mozdul a távolsággal arányosan.

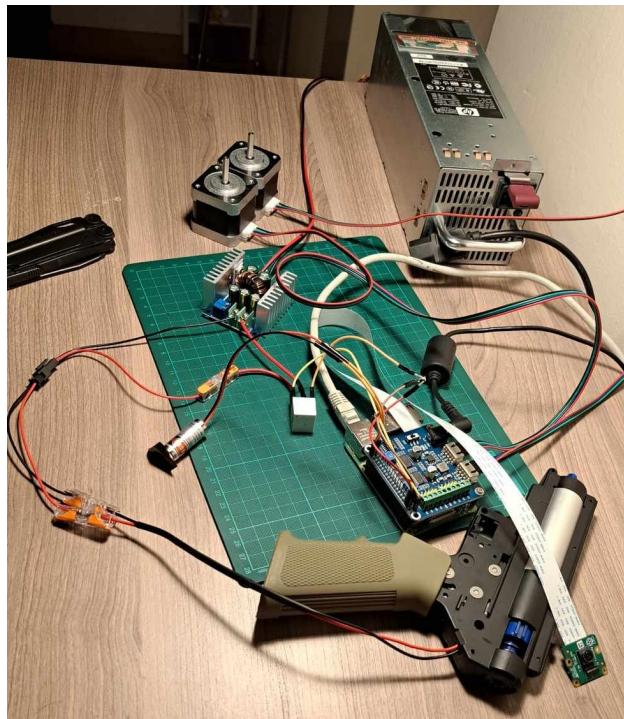
A motor csak akkor mozog, ha egy bizonyos értéknél nagyobb a távolság, így kerültem el, hogy folyamatosan rángratázzon az eszköz. Ez a folyamat is csak akkor fut, hogyha a `automode_event` igaz.

## 7. Tesztelés

A prototípus tesztelése több lépcsőben zajlott: először a hardvert tesztettem, a motorok illesztését, a gearbox működését. Ezután a képfelismerő algoritmusokat hasonlítottam össze, először csak egy statikus kamerával. Mikor elkészültek a 3D nyomtatott alkatrészek, ki tudtam próbálni a kamera célfelismerést és a motorok mozgásának összehangolt működését. Ezzel együtt ki tudtam próbálni, mennyire lett stabil a mechanikai konstrukció, illetve megbízható-e a tüzelés folyamata.

### 7.1. Elektronikai alkatrészek tesztje

Először csak csatlakoztattam a kamerát a Raspberry Pi megfelelő csatlakozójához, a szalagkábelrel, a kamerákat pedig a stepper motor HAT csatlakozóihoz. Ezután a GPIO tüskesorhoz csatlakoztattam a lézer diódát, a relé megfelelő lábait, és a végálláskapcsolókat. A teszt összeállítása a 36. ábrán látható.



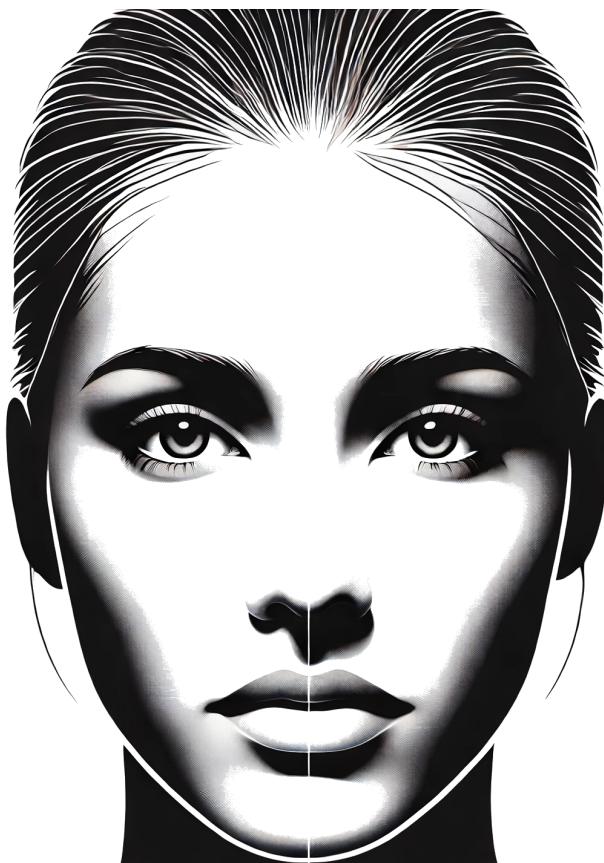
36. ábra. Elektronikai alkatrészek teszt összeállítása

A teszt során azt próbáltam ki először, hogy tudom-e mozgatni a motort, működik-e a kamera, a lézer modul, tudok-e lőni a gearboxszal. minden

sikeress volt, így ezt a tesztet lezártam.

## 7.2. Képfelismerő algoritmusok tesztje tesztje

A teszt összeállítása itt is ugyanaz volt, mint az előző bekezdésben, annyi különbséggel, hogy a kamerát felerősítettem egy állványra, és egy kinyomtatott arcot mozgattam előtte. A célpont a 37. ábrán látható. A minta részletei nagyon kontrasztosak, elütnek egymástól, ezzel könnyítve az algoritmusok munkáját.



37. ábra. Az arcfelismeréshez használt kép

Először két olyan módszert vizsgáltam, ahol a célpont nem általános, hanem egy PNG képet vesz mintának, és ezt keresi a kamera által szolgáltatott képen.

## Template Matching

Erről a módszerről korábban a 6.2.1. bekezdésben írtam. A `cv2.matchTemplate()` függvény létrehozza megkeresi a képen sablont, a `cv2.minMaxLoc()` pedig visszaadja a helyzetét, valamint az eredmény pontosságát. Az eredmény pontosságának megszabtam egy alsó határt, és azt változtattam a teszt során. Azt tapasztaltam, hogy ez a módszer nagyon érzékeny a célpont dőlésére és elfordulására, valamint a távolságára is. Semmiképpen nem optimális a feladat elvégzésére.

## Feature Matching és Target Tracking

A feature matching elviekin jól kezeli a célpont torzulását, ezért ezzel kísérleteztem a későbbiekbén. Az elképzélés az volt, hogy a feature matching megtalálja megbízhatóan a célpontot, majd a target tracking képes lesz gyorsan követni. Amennyiben egyszer megtalálta az algoritmus a célpontot, onnan tényleg pontosan tudta követni, azonban magával a felismeréssel voltak itt is a problémák. Nagyon megbízhatatlan volt, ha az érzékenységet lentebb állítottam, akkor fals pozitívot is mutatott, ha fentebb, akkor pedig nem találta meg a célpontot. A target tracking is csak akkor működött, ha a kamera egy helyen maradt. Ha már a motorok is mozogtak, akkor könnyedén elvesztette a célpontot.

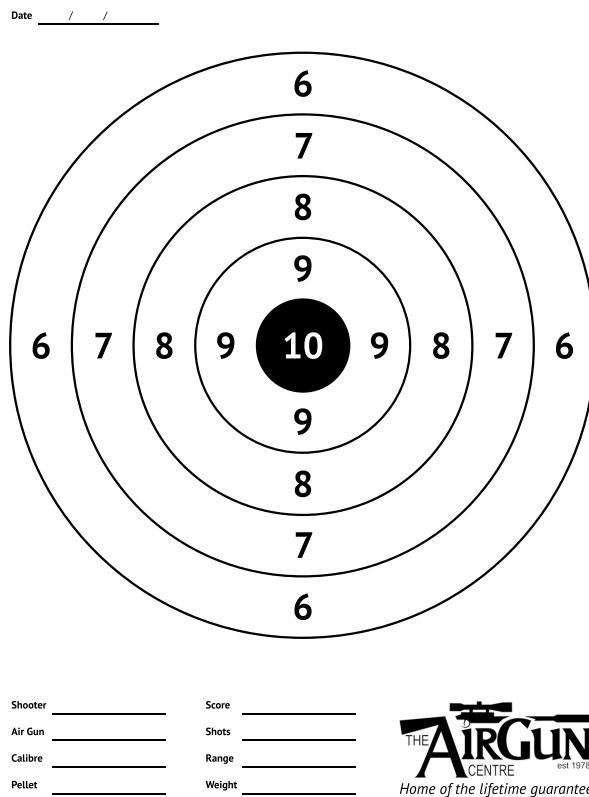
### 7.2.1. Haar Cascade

Az utolsó módszer, amit teszteltem, az a Haar Cascade és tanított neurális hálók használata volt. Ez az előzőekhez képest meglepően megbízható volt, a célpontot felismerte viszonylag távolról is, akár rossz fényviszonyok között is. A célpont kb 15 fokos dőlése esetén szintén felismerte azt, amit én megfelelőnek találtam. Ezzel a módszerrel folytattam a munkát.

## 7.3. Éles teszt

A tesztet egy jól megvilágított tanteremben végeztem. Az eszköz egy stabil padon állt, a célpont pedig 5 méterrel előtte, és kb. fél méterrel fölötte. A célpont a 38. ábrán látható.

A teszt során a manuális működési móddal kezdtem, tulajdonképpen belőttem a fegyvert. Először azt tapasztaltam, hogy a kamera képének a közepe nem ott van, ahova a fegyver csöve mutat. Ezért a célkeresztet addig kellett mozgatnom a képen, amíg oda nem mutatott, ahova a fegyver ténylegesen lőtt. Ezután azt tapasztaltam, hogy nagy pontossággal képes



38. ábra. Az éles teszthez használt lőlap

lóni, igen kis területen belül. A lézer irányzékot azonban nem tudtam pontosan beállítani, 5 méteren nagyjából 10-15 cm-t téved. A következő ábrákon látható a teszt eredménye. 5-ös sorozatokban tüzeltem, minden a cél közepére irányítva a kamera célkeresztjét.

A következő teszt az arcfelismerés, és a követés funkció tesztje volt. A körfüggvények ugyanazok maradtak, és szintén 5 m-re volt a célpont. A teszt során egy kollégá tartotta a kinyomtatott arcképet, és mozgatta, változtatva a sebességet és irányt. Természetesen a gearbox nem volt áram alatt, és a kollégá is viselt védőszemüveget. Azt tapasztaltam, hogy a fegyver körfüggvénybelül a sétáló ember sebességét tudja követni 5 m távolságban, és igen pontosan be tudja célozni.

A tesztkörnyezet a 39. és 40. ábrákon látható.

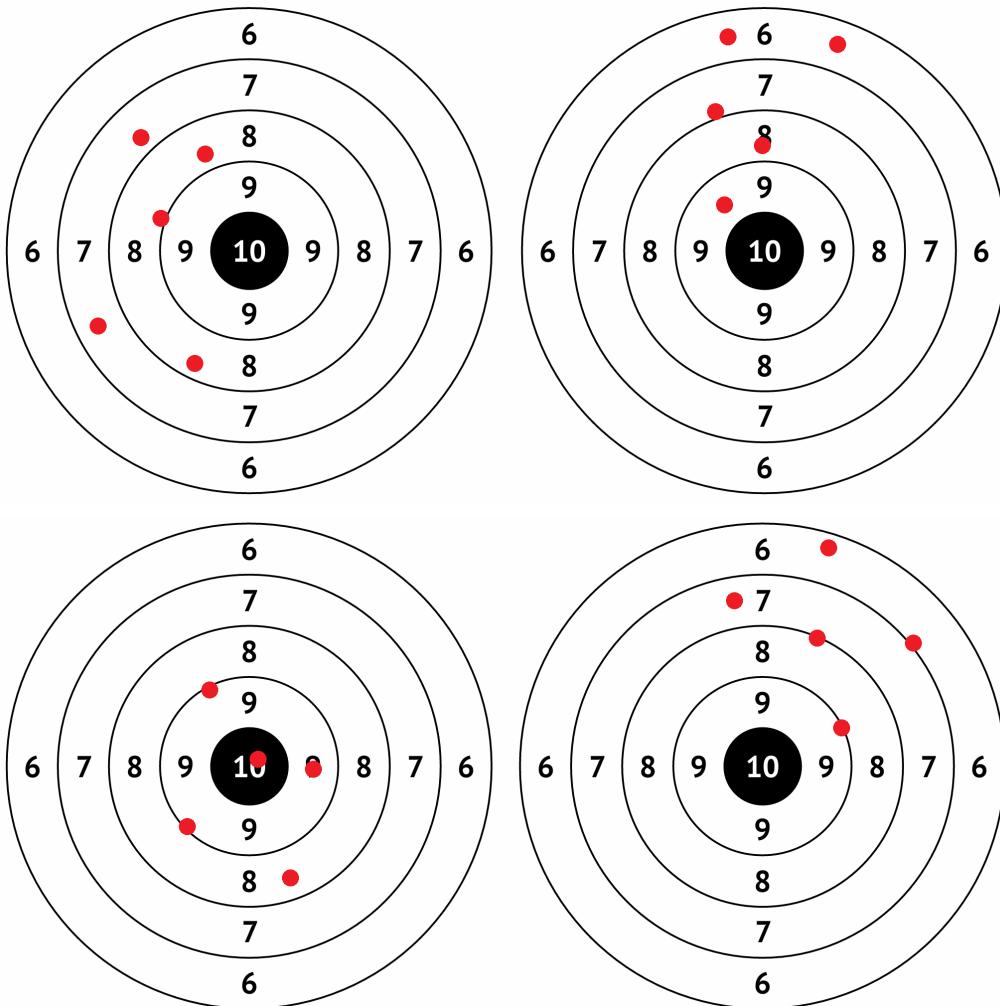


39. ábra. Az éles teszt környezete



40. ábra. Az éles teszt környezete

Összességében elmondható, hogy a beállított célkereszttel, 5 méterről igen pontos eredmények születtek. 4 leadott sorozat során 5-5 golyót lőttem ki, az eredmények az alábbi ábrán láthatóak:



41. ábra. Az éles teszt eredményei

## 8. Hibák, észrevételek

A prototípus fejlesztés során számos helyen vétettem kisebb-nagyobb hibákat, amiket súlyosságuktól függően vagy javítani kellett, vagy valahogy kikerülni. Az első, amit meg szeretném említeni, az a **tár kialakítása** volt. Nem voltam elég körültekintő, és anélkül terveztem meg a tárat, hogy megnéztem volna, az eredeti fegyverekben hogyan működik. Ennek eredménye lett a nagy kapacitású tár, ami aminek elméletben folyamatosan kéne adagolni a golyókat, de a gyakorlatban a szűkítésnél minden esetben elakad. Ezzel a kialakítással egyet sem sikerült tüzelni, így újra kellett tervezni a teljes tárat. Miután jobban megvizsgáltam az eredeti fegyvert, láttam, hogy a golyók egyesével sorakoznak minden esetben, ezzel gátolva az elakadást. Miután hasonlóan megterveztem az új tárat, már megbízhatóan műköött, habár kisebb kapacitással, így ezt a hibát sikerült kijavítani.

A következő, egy szintén mechanikai természetű hiba, a **lézer modul és a kamera rögzítése**. Egyik sem állítható, és ha eredetileg nem egy pontra néznek a fegyver csövével, akkor tulajdonképpen a hiba maradandó lesz. Ez így is lett, de a kamera eltérését még a célkereszt állításával tudtam orvosolni. A lézer modul maradandóan pár fokkal félre áll, ezt sajnos el kellett fogadnom.

A projekt elektronikai részénél is van, amit máshogy csinálnék, ha újra kezdeném. Az első koncepcióterveknél még nem néztem utána, mekkora áramot képes felvenni a gearbox. Úgy gondoltam, hogy egy átlagos 12V-os laptoptöltőről is tud majd működni a rendszer. Így igen meglepődtem, mikor utánanéztem, mekkora teljesítményre van szüksége. Az eredmény így az lett, hogy egy külön tápot kellet adnom csak a gearbox működtetésére, ami igen kényelmetlen. Valószínűleg az lenne helyette egy jó megoldás, ha tervezek egy külön nyomtatott áramkört, amely képes arra, hogy töltön egy akkumulátort a fegyveren. Így a gearbox arról működne, akár csak egy módosítatlan fegyveren, működésen kívül pedig a Raspberry PI tápjáról töltődne.

## 9. Továbbfejlesztési lehetőségek

**Erősebb szerkezet:** Az egyik egyértelmű fejlesztési lehetőség a jobb minőségű alkatrészek használata. Elsősorban a fogaskerekekre gondolok, amelyeket ha nem 3D nyomtatással gyártottam volna, hanem valamilyen nagy pontosságú folyamattal, akkor a szerkezet nem csak csendesebb, de precízebb is lehetne.

**Jobb kamera:** A Raspberry PI Camera 2 megfelelő minőséggel rendelkezik arra, hogy ezt a prototípust használni tudjam vele. Azonban ha távolabbi célpontokat kellene keresni, akkor egy optikai zoom mindenkorban szükséges lenne.

**Vezeték nélküli működés:** A tápellátás és a kommunikáció vezeték nélküli-vé tétele magában hordozna új kihívásokat, például be kellene építeni egy nagyobb teljesítmény akkumulátort, illetve a vezeték nélküli kapcsolatok rendszerint lassabbak, mint a vezetékesek. Ellenben ha az eszköz célját tekintem, a távolsági operálás lenne a természetes fejlesztési irány, ahol a felhasználó akár egy másik épületből is képes lenne irányítani a fegyvert.

**Mozgás:** És ha már vezetékek nélkül képes működni az eszköz, akkor a következő lépés az lenne, hogy mozogni tudjon, például lánctalpak segítségével. Ez is szintén új kihívásokat jelent, de az eszköz többi részéhez képest már nem lenne nagyon bonyolult megvalósítani.

**Radar, távolságérzékelő:** Az eszköz következő verziójába kerülhetne akár radar is. A hasonló eszközök modern felhasználása leginkább légvédelmet jelent, azon belül drónok lelövését. Ezt vizuálisan nehézkes lehet érzékelni, hiszen igen távol is lehetnek, és kicsi a keresztmetszetük. Egy radar beépítése azonban megkönnyítené az észlelést. A távolságérzékelő pedig egyrészt több információt szolgáltat a cépontról, másrészt pedig nagy távolságoknál már számolni kell a lövedék ballistikájával is, tehát hogy esik.

## 10. Költségek

A projekt költségei az alábbi táblázatban láthatók:

Bolt	Termék	Ár
airsoftgun.hu	SA ASSAULT RIFLE C02 Half-tan	49400
amazon.de	Waveshare stepper motor hat	12278
amazon.de	Lazy Susan csapágy	4654
csavardiszkont.hu	csavarok	3700
csapagywebaruhaz.hu	csapágyak	460
conrad.hu	2x szorító	2090
conrad.hu	2m hangszóró kábel	400
conrad.hu	2x loctite ragasztó	1200
3dJake	1 kg filament	8100
conrad.hu	wago	1000
saját tulajdon	raspberry	25000
egyetem	raspberry camera	7500
Hestore (egyetemi költség)	Hestore rendelés	37500
	Szumma	153282

1. táblázat. Költségek

## **11. Konklúziók**

a

## **12. Köszönetnyilvánítás**

a

## Függelék

### Kód részletek

## Hivatkozások

- [1] Leírás a Team Fortress 2 játékban lévő Sentry Gun-ról (Hozzáférés: 2024.11.19.)  
[https://wiki.teamfortress.com/wiki/Sentry\\_Gun](https://wiki.teamfortress.com/wiki/Sentry_Gun),
- [2] Leírás a Portal játékban lévő Sentry Gun-ról (Hozzáférés: 2024.11.19.)  
[https://theportalwiki.com/wiki/Sentry\\_Turret](https://theportalwiki.com/wiki/Sentry_Turret),
- [3] Leírás a Phalanx CIWS rendszerről (Hozzáférés: 2024.11.19.)  
[http://www.navweaps.com/Weapons/WNUS\\_Phalanx.php](http://www.navweaps.com/Weapons/WNUS_Phalanx.php),
- [4] Leírás a CROWS rendszerről (Hozzáférés: 2024.01.22.)  
<https://www.globalsecurity.org/military/systems/ground/m101-crows.htm>,
- [5] Leírás az Arbalet-DM rendszerről (Hozzáférés: 2024.01.22.)  
<http://roe.ru/eng/catalog/land-forces/RWS/arpalet-dm/>,
- [6] Leírás az deFNder Medium rendszerről (Hozzáférés: 2024.01.22.)  
<https://fnamerica.com/products/weapon-systems/defnder-medium/>,
- [7] Leírás a Samsung SGR-A1 rendszerről (Hozzáférés: 2024.02.10.)  
<https://www.globalsecurity.org/military/world/rok/sgr-a1.htm>,
- [8] Információ a paintball fegyverekről (Hozzáférés: 2024.10.22.)  
<https://steeltownpaintball.com/what-you-should-know-about-paintball-buying-upgrades-and-mods/>
- [9] Információ a paintball robotról (Hozzáférés: 2024.10.22.)  
<https://community.robotshop.com/robots/show/robotic-paintball-sentry>,
- [10] Információ a Nerf fegyverekről (Hozzáférés: 2024.10.22.)  
<https://shop.hasbro.com/en-us/nerf>,
- [11] Információ az airsoft fegyverekről (Hozzáférés: 2024.10.22.)  
<https://highspeedbbs.com/all-about-airsoft-guns-types-styles-facts-science/>,
- [12] Teszt az airsoft gearbox áramfogyasztásáról (Hozzáférés: 2024.10.22.)  
[http://www.airsoftlab.eu/docs/experiments/motor\\_current/](http://www.airsoftlab.eu/docs/experiments/motor_current/),
- [13] A NEMA szabvány (Hozzáférés: 2024.10.22.)  
<https://www.cncitalia.net/file/pdf/nemastandard.pdf>,
- [14] Creality Ender 3 Pro (Hozzáférés: 2024.10.22.)  
<https://www.creality.com/products/ender-3-pro-3d-printer>,

- [15] Felhasználói diskurzus a Raspberry PI gépi látás kapacitásáról (Hozzáférés: 2024.10.22.)  
<https://forums.raspberrypi.com/viewtopic.php?t=366431>,
- [16] Raspberry 4B specifikációk (Hozzáférés: 2024.10.22.)  
<https://www.raspberrypi.com/products/raspberry-pi-4-model-b/specifications/>,
- [17] Waveshare stepper motor hat spevifikációk(Hozzáférés: 2024.10.22.)  
[https://www.waveshare.com/wiki/Stepper\\_Motor\\_HAT](https://www.waveshare.com/wiki/Stepper_Motor_HAT),
- [18] Raspberry 2 kamera specifikációk (Hozzáférés: 2024.10.22.)  
<https://www.raspberrypi.com/products/camera-module-v2/>,
- [19] Dokumentáció az OpenCV modulról (Hozzáférés: 2024.10.22.)  
<https://www.raspberrypi.com/products/camera-module-v2/>,
- [20] Dokumentáció a Socket modulról (Hozzáférés: 2024.10.22.)  
<https://docs.python.org/3/library/socket.html>,
- [21] Dokumentáció a Multiprocessing modulról (Hozzáférés: 2024.10.22.)  
<https://docs.python.org/3/library/multiprocessing.html>,
- [22] Dokumentáció a Pygame modulról (Hozzáférés: 2024.10.22.)  
<https://www.pygame.org/docs/>,
- [23] Dokumentáció a Pickle modulról (Hozzáférés: 2024.10.22.)  
<https://docs.python.org/3/library/pickle.html>,
- [24] Dokumentáció a CV2 Template Matching-ről (Hozzáférés: 2024.10.22.)  
[https://docs.opencv.org/4.x/d4/dc6/tutorial\\_py\\_template\\_matching.html](https://docs.opencv.org/4.x/d4/dc6/tutorial_py_template_matching.html),
- [25] Dokumentáció a CV2 Feature Matching-ről (Hozzáférés: 2024.10.22.)  
[https://docs.opencv.org/3.4/dc/dc3/tutorial\\_py\\_matcher.html](https://docs.opencv.org/3.4/dc/dc3/tutorial_py_matcher.html),
- [26] Dokumentáció a CV2 Target Tracking-ről (Hozzáférés: 2024.10.22.)  
[https://docs.opencv.org/4.x/d2/d0a/tutorial\\_introduction\\_to\\_tracker.html](https://docs.opencv.org/4.x/d2/d0a/tutorial_introduction_to_tracker.html),
- [27] Dokumentáció a CV2 Haar Cascade-ról (Hozzáférés: 2024.10.22.)  
[https://docs.opencv.org/3.4/db/d28/tutorial\\_cascade\\_classifier.html](https://docs.opencv.org/3.4/db/d28/tutorial_cascade_classifier.html),