



BUDAPESTI MŰSZAKI ÉS GAZDASÁGTUDOMÁNYI EGYETEM  
GÉPÉSZMÉRNÖKI KAR  
MECHATRONIKA, OPTIKA ÉS GÉPÉSZETI INFORMATIKA TANSZÉK

KREINICKER GÁBOR  
NANOMŰHOLDOK ORIENTÁCIÓ  
MEGHATÁROZÁSA

Témavezető:

*Pál András*

CSFK Csillagászati Intézet, tudományos főmunkatárs

Konzulens:

*Dr. Nagy Balázs Vince*

egyetemi docens, MOGI tanszék

Budapest, 2022.

# Tartalomjegyzék

<b>1. Bevezetés</b>	<b>1</b>
<b>2. Hardver</b>	<b>2</b>
<b>3. Szimulációk</b>	<b>7</b>
3.1. Szoftveres szimulálás . . . . .	7
3.2. Hardveres szimulálás . . . . .	8
<b>4. Képfeldolgozás</b>	<b>12</b>
4.1. Nap-keresés . . . . .	12
4.1.1. Iterációs megoldás . . . . .	12
4.1.2. A szubpixel-eljárás . . . . .	13
4.1.3. Szomszédos pixelek módszere . . . . .	13
4.2. Föld-keresés . . . . .	13
4.2.1. Intervallumfelező módszer . . . . .	14
4.2.2. Keretmódszer . . . . .	15
<b>5. Adatfeldolgozás</b>	<b>17</b>
5.1. Kvaterniógenerálás . . . . .	18
5.2. Bázistranszformáció . . . . .	19
<b>6. Összefoglalás</b>	<b>21</b>
<b>Irodalomjegyzék</b>	<b>22</b>

# 1. fejezet

## Bevezetés

A Csillagászati és Földtudományi Kutatóközpont (CSFK) Konkoly-Thege Miklós Csillagászati Intézetével együttműködve célunk egy infraszenzoros műhold orientáció meghatározó rendszer fejlesztése.

Kisméretű, úgynevezett nanoműholdak[1] esetében jelenleg még számos nyitott kérdést tartogat az orientációnak – azaz a műhold térbeli helyzetének – nemcsak a stabilitása, hanem a meghatározása is. Ugyanis ezek Föld körüli pályára állásuktól kezdve általában irányíthatatlan módon forognak, mely megnehezíti a tudományos mérések végzését vagy akár az adatok Föld-irányú továbbítását.

Ahhoz, hogy stabilizálni tudjuk egy műhold helyzetét, ismernünk kell annak megváltozását is. Ezt tehetnénk magnetométerrel, amely a Föld mágneses teréhez képest határozza meg az orientációt. Azonban ez önmagában nem elegendő, ugyanis ezzel a három szabadsági fokból csak kettőt tudunk meghatározni, amely nem definiálja egyértelműen az orientációt. Egy olyan megoldáson dolgozunk, amely képfeldolgozást alapul véve képes meghatározni a műhold térbeli helyzetét.

## 2. fejezet

### Hardver

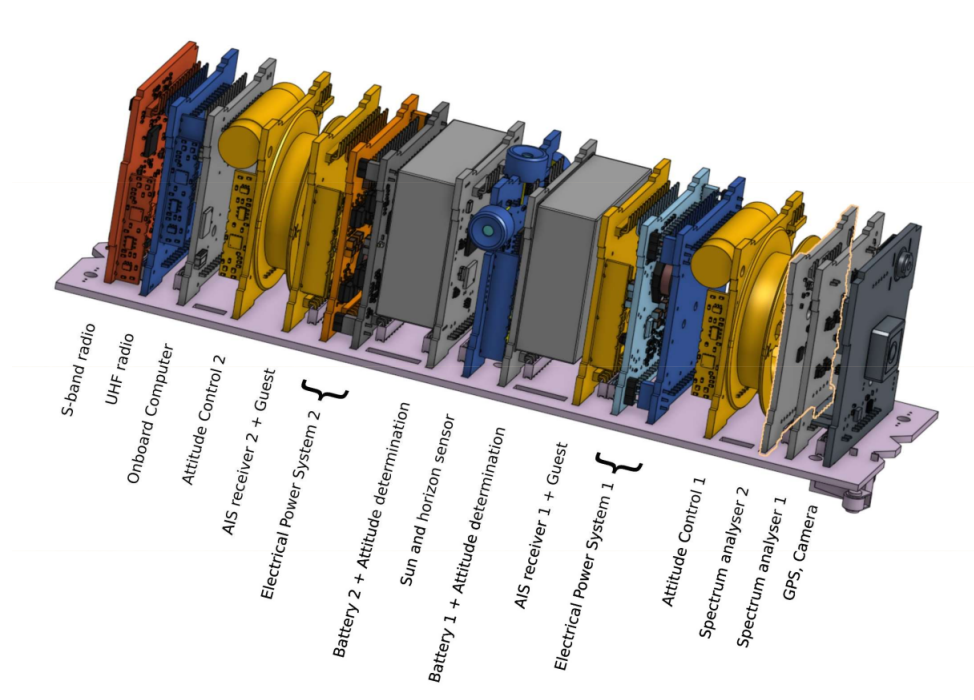
Jelenleg az MRC-100 nevű, BME-s fejlesztésű műholdra készítjük az első, infra szenzoros orientáció-meghatározó rendszerünket. Ez a rendszer lesz felelős azért, hogy orientáció-adatokkal lássa el a műholdat, amely ennek köszönhetően a benne megfelelően elhelyezett tekercsekkel stabilizálni, módosítani tudja térbeli helyzetét.

Az általunk fejlesztett áramkör a műhold közepén fog elhelyezkedni (2.1. ábra). Itt négy darab infra kamera kapott helyet, egymással  $90^\circ$ -os szöget bezárva, párhuzamosan a műhold oldalaival, ezzel maximalizálva a belátott égboltot. Ez az áramkör több részre bontható, melyek közül azokat emelném ki, amelyekkel tüzetesebben foglalkoztam.

Maguk az MLX90641 [3] típusú szenzorok távoli infratartományban érzékelnek (2.2. ábra). Felbontásuk csekély,  $16 \times 12$  pixel, azonban hatalmas a látószögük,  $110 \times 75^\circ$ , így nagy a lefedettségük (lásd: 2.3. és 2.4. ábrák). Ezen tulajdonságainak és kis fogyasztásuknak köszönhetően ideálisnak találtuk őket a kitűzött cél elérésére. Teszteltünk ugyanebbe a családba tartozó MLX90640 szenzort is, amely felbontása  $32 \times 24$  pixel. Ezek is jól működtek, azonban energiaigényük miatt maradtunk a kisebb felbontású érzékelők mellett.

A szenzorok olvasására STM32F072 [4] típusú mikrokontrollert alkalmazunk, mely már bizonyított a GRBA $\alpha$  [5] műholdon, űrbéli alkalmazások terén. Ennek szintén alacsony energiaigénye, valamint nagyobb tranzisztormérete – mely a különböző sugárzások ellen biztosít nagyobb védelmet – ad igazán kedvező lehetőségeket. Ennek azonban az az ára, hogy jóval kisebb erőforrás áll rendelkezésre. A teljes modul prototípusát a 2.5. ábrán láthatjuk.

Így az egyik legnagyobb feladat az optimális szoftver elkészítése, hogy minden a lehető legjobban üzemeljen. Ennek az áramkörnek ugyanis a következő feladatokat kell ellátnia:

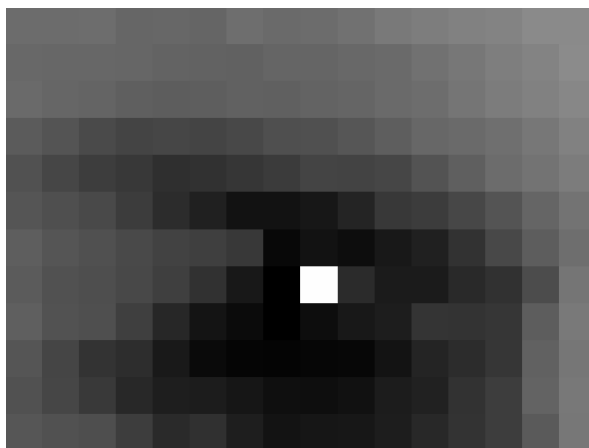


**2.1. ábra.** Az MRC-100 műhold[2] felépítése. A középső, kék színű jelölt áramkör az általunk fejlesztett egység, amelyen megfigyelhetők az  $x, -x, y, -y$  irányú infraszenzorok. A két szélén, sárgával jelölt áramkörök végzik majd az aktív stabilizálást



**2.2. ábra.** Az alkalmazott MLX90641 infraszenzorok[3]. Ezek felbontása  $16 \times 12$  pixel, látószögük  $110^\circ$ . A képen látható TO-39-es tokozásban lesznek szerelve

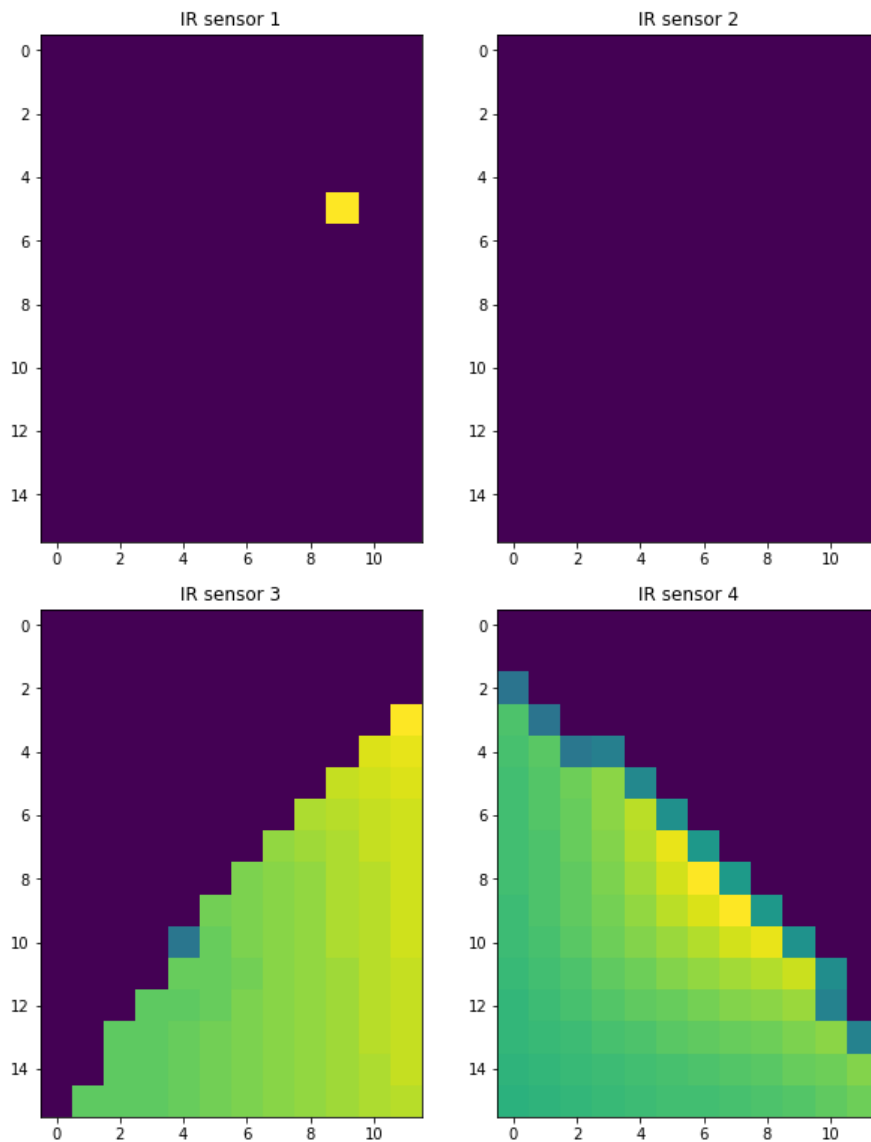
1. Infraszenzorok olvasása. Ezek a kamerák I<sup>2</sup>C protokollt alkalmazva kommunikálnak a mikrovezérlővel. Egy mikrovezérlő egyszerre két szenzort tud maximális sebességgel olvasni, így két mikrovezérlőt alkalmazunk. Közvetlen memóriához-záférést alkalmazva (Direct Memory Access, DMA) sikerült 16 Hz-en tartósan üzemeltetni ezeket, amely biztosan elegendő, figyelembe véve a lehetséges maximális szögsebességeket.



**2.3. ábra.** Egy szenzorral készített szabadtéri kép a Napról. Jól látszik, hogy milyen jól és élesen elkülönül a Nap a környezetétől. Azonban a környező épületek hősugárzása is megjelenik a képen, mely adatfeldolgozásban nehézséget okoz.

2. Beolvasott  $16 \times 12$  pixeles kép feldolgozása, Föld és (amennyiben látható) Nap detektálása. A következő lehetőségek fordulhatnak elő egy szenzor képein:
  - A kamera semmit sem lát. Ekkor semmit nem tudunk felhasználni, a képen csak termikus zaj jelenik meg. Szimulációink szerint olyan lehetőség azonban nem fordulhat elő, hogy egyik szenzor se lásson semmit, ugyanis a pályamagasság kellően alacsony, és a látószög kellően nagy ahhoz, hogy valamelyik kamerán minden esetben látszódjon a Föld.
  - A kamera képen látszik a Föld. Ekkor meg tudjuk határozni a Föld normálisát a detektálószenzor koordinátarendszerében.
  - A kamera képen látszik a Nap. Ekkor meg tudjuk határozni a szenzor-Nap vektort. A szimulációink alapján egy keringési periódus ( $\approx 90$  perc) során az esetek körülbelül 30%-ában nem látható a Nap, ugyanis azt a Föld kitakarja. Ez ellen sajnos nem tudunk mit tenni, azonban megfelelő idő után ezekben az esetekben is jól meg tudjuk becsülni az orientációt korábbi adatok alapján.
  - A kamera képen látszik a Föld és a Nap is. Ekkor mindkét vektor meghatározható.
3. Orientáció meghatározása. Ez az egyik legkritikusabb része az egész rendszernek. A négy szenzor által detektált objektumokból kapott vektorokat kell valamilyen formában összefüggésbe hozni a műhold orientációjával. Egyrészt a szenzorok saját koordinátarendszerében meghatározott vektorokat kell átranzformálni a műhold saját koordináta rendszerébe. Ebből lesz egy térbeli pozíciónk, azonban ez még határozatlan, ugyanis csak a műhold-Föld- és műhold-Nap-vektorok ismertek. Ezért alkalmazzuk a TLE-adatokból[6] nyert Föld-műhold- és Föld-Nap-vektorokat,

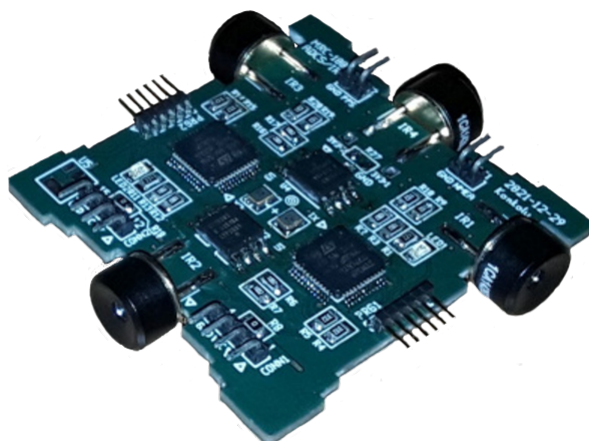
#### Infraszenzorok Unity-ben szimulált képei



**2.4. ábra.** Unity-ben készített szimulációkban generált képek. A szimuláció a valós fizikai méreteknek és kamera paramétereknek megfelel, így a valóságoshoz igen közeli képeket kaptunk. Bal felül a Nap, míg az alsó képeken a Föld figyelhető meg.

amelyből az utóbbi közelíthető a műhold-Nap-vektorral, ugyanis a Föld-műhold távolság elhanyagolhatónak tekinthető a Föld-Nap távolsághoz képest. Ezeknek köszönhetően már határozottá válik a rendszer, és ismertté válik az orientáció.

4. Orientációadatok továbbítása a műhold további rendszerei felé. A feladat ezen részével a dolgozat beadásáig nem foglalkoztam.



**2.5. ábra.** Az általunk fejlesztett egység egyik első prototípusa. Megfigyelhetők rajta az  $x$ ,  $-x$ ,  $y$ ,  $-y$  irányú infraszenzorok.



## 3. fejezet

# Szimulációk

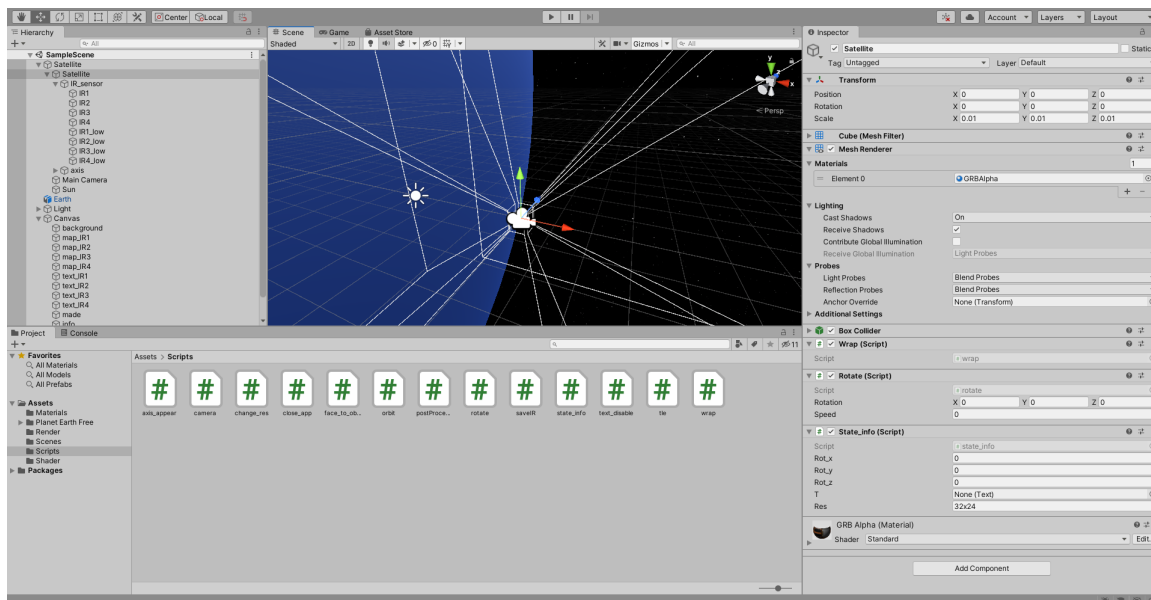
Ahhoz, hogy tesztelni tudjuk a képfelismerő algoritmusainkat, szükségünk van adatokra, azonban valódi, űrben készült képek nem állnak rendelkezésünkre. Mivel ismerjük a műhold pályamagasságát (TLE-adatokból), illetve az infrakamera különböző paramétereit, mint a felbontás és látószög, így meg tudjuk becsülni egy, a műholdon készült kép jellegét. Ennek megfelelően két fő hőforrás van, amelyet a műhold szenzorai egyértelműen érzékelni tudnak: a Föld, mely egy összefüggő, enyhén fényes felületként, illetve a Nap, amely egy nagyon fényes pontként jelenik meg a kapott képen. Ezeknek az ismeretében több módszert is kipróbáltunk egy űrben készült kép imitálására.

### 3.1. Szoftveres szimulálás

Egyik megoldás a szoftveres képgenerálás. Próbáltunk Python nyelven megírt programokkal képeket generálni, azonban különböző mozgásokat ezzel szimulálni nem tudtunk. Ezért a Unity[7], elsősorban videojáték-fejlesztő platform felé fordultunk (3.1. ábra). Ebben lehetőség van tárgyakat elhelyezni, egyszerűen mozgatni és különböző kameraszögeket megjeleníteni. Miután létrehoztuk a Föld és a műhold méretarányos statikus modelljét, elkészítettük a műhold orientációmódosító programkódját.

Ezután elhelyeztük a műholdmodellen a négy kamerát a tervezett elrendezéseknek megfelelően. Beállítottuk ezek látószögét és felbontását, illetve mivel ezen a platformon nem tudunk infrakamerát szimulálni, így monokróm kamerát alkalmaztunk. A Föld illetve a Nap textúráit egyszerű színekre változtattuk úgy, hogy fekete-fehér képen a Nap jóval fényesebbnek tűnjön.

A kamerák által látott képeket fájlba mentjük úgy, hogy a képfrissítés megegyezzen a valódi kamerák képfrissítésével. Ezeket a fájlokat pedig valós időben tudjuk feldolgozni egy Python programmal.



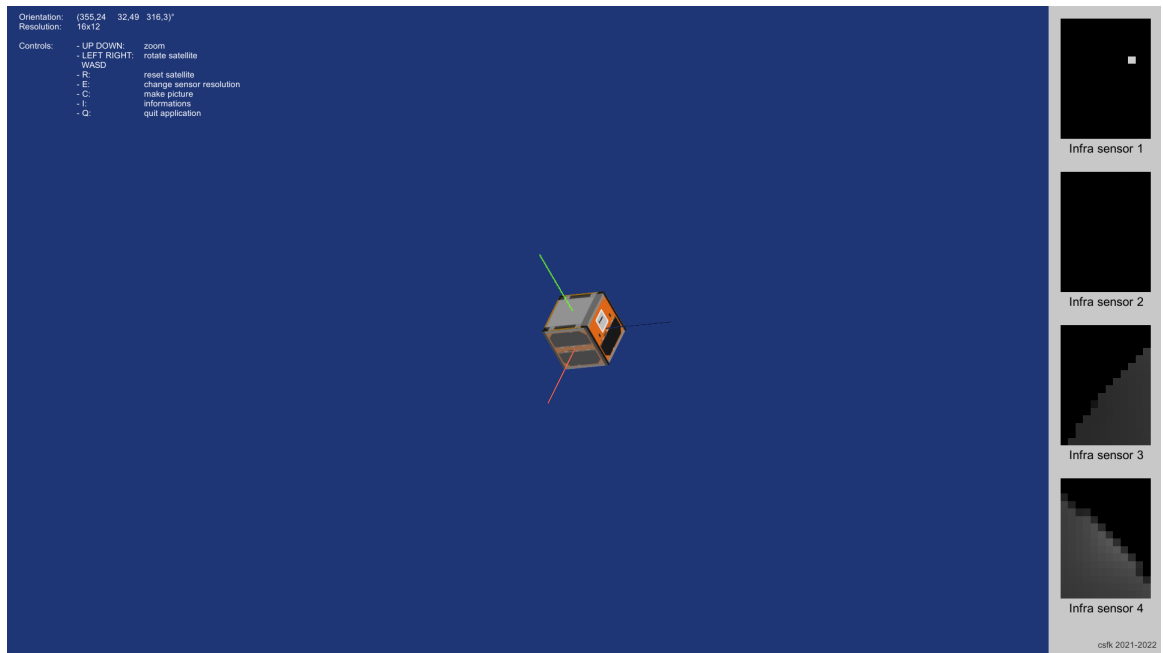
**3.1. ábra.** Az Unity videojáték-fejlesztő platform kezelőfelülete. Ez a környezet az esetünkben nagyszerű lehetőséget nyújt az infrakamerák szimulálásra. Különböző alprogramokkal tovább tudtuk közelíteni a szimulációnkat a valósághoz.

A Nap Földhöz képest vett pozícióját TLE-adatokból ismerjük, így a szimulációban ezt is pontosan meg tudjuk jeleníteni. Ennek a haszna elsősorban a szimuláció pontosságának ellenőrzése más, már a Föld körül keringő műholdak TLE-adatai alapján, illetve a felbocsátás után nyerhetünk képet a műhold kamerái által látottakról (3.2. ábra). Ezeknek megfelelően billentyűkkel tudjuk a műholdat forgatni, illetve a kamerák felbontásán is tudunk változtatni.

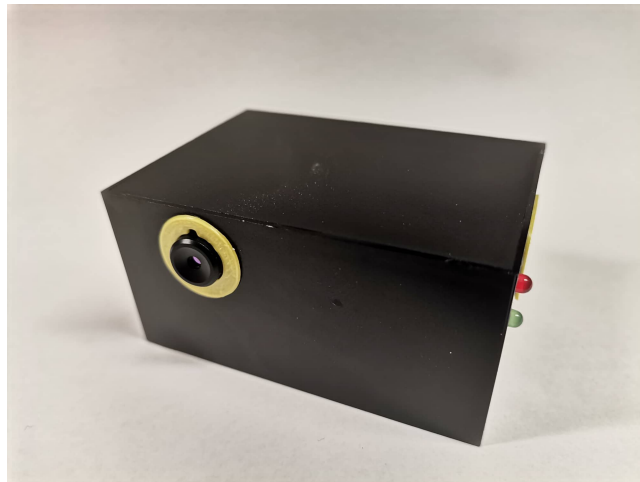
## 3.2. Hardveres szimulálás

Másik megoldás a műholdra elhelyezni tervezett infrakamerákkal megegyező szenzorokkal képeket készíteni a Napról. Eleinte egy szabadban elhelyezett kamerával kezdtünk el képeket gyűjteni (3.3. ábra). Ezeknek a képeknek számos problémája van: a környezet hősugárzása is megjelenik a képen, ami a világűrben nem lesz jelen, így ez megnehezíti a képfeldolgozást, másrészt míg a Föld egyszeri körbefordulása 24 óra, addig a műhold keringési periódusa csak körülbelül 90 perc. Második problémára a megoldás azt jelentette, hogy a kamerát mozgatjuk. Ahhoz, hogy ezt megfelelően tudjuk végrehajtani, szükség volt egy olyan eszközre, ami előre beprogramozott módon tudja forgatni a szenzort.

Ez az eszköz el is készült, egy Arduino Nano [8] vezérlésű, két szabadsági fokú állvány (3.4. ábra). A forgatásért két 28byj típusú léptetőmotor [9] felel. Van lehetőség előre beprogramozott, de akár manuális forgatásokra is egy joystick segítségével. A



**3.2. ábra.** A Unity-ben készült szimuláció futás közben. Valós időben tudjuk változtatni a műhold orientációját, mellyel egyidejűleg láthatjuk a kamerák képeit. Ezen képek lementése automatikusan történik.

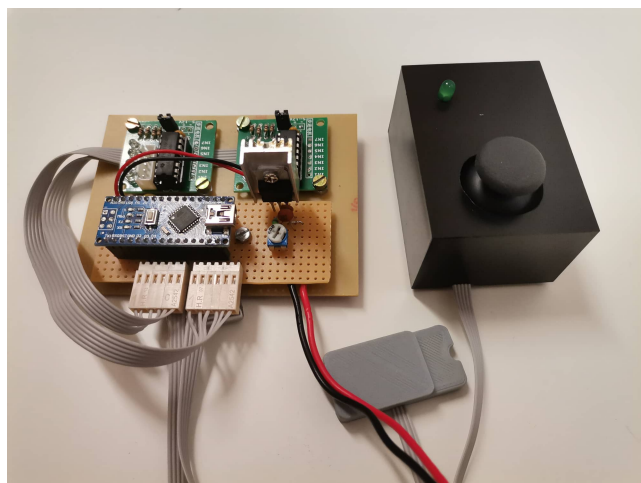


**3.3. ábra.** A szabadtéri képek készítésére összerakott eszköz. Ebben egy STM32F042 mikrovezérlő található, mely nagyon hasonlít a műholdon alkalmazott STM32F072-re.

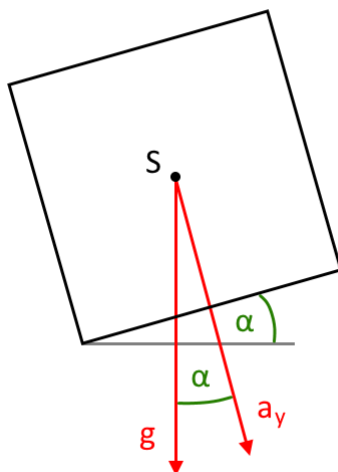
vízszintessel bezárt szöget egy MPU-9250 [10] IMU szenzor gyorsulásadataiból ismerjük: a függőleges irányú gyorsulás és az ismert nehézségi gyorsulás alapján meghatározható:

$$\alpha = \arccos \frac{a_y}{g}, \quad (3.1)$$

ahol  $a_y$  a mért y-irányú gyorsuláskomponens és  $g$  pedig a nehézségi gyorsulás, azaz  $g \approx 9.81 \text{ m s}^{-2}$  (lásd még: 3.5. ábra).

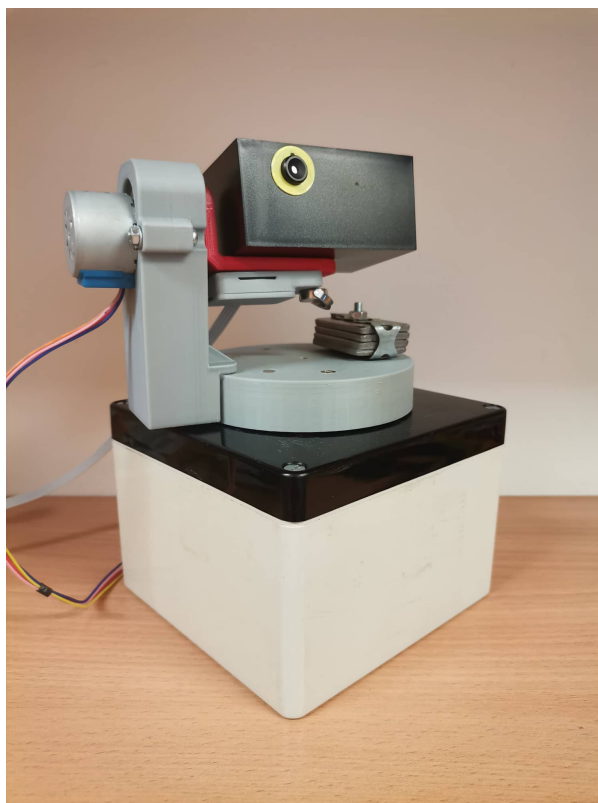


**3.4. ábra.** A forgatórendszer vezérlő áramköre. Itt helyet kapott két 28byj léptetőmotor-vezérlő, egy Arduino Nano és egy LM317-es IC-vel megvalósított feszültségszabályozó áramkör, ugyanis a rendszer 5 V-os, míg az akkumulátor 12 V feszültségű. Alul az IMU szenzor található 3D nyomtatott tokozásban, jobb oldalon pedig a manuális forgatást lehetővé tevő joystick.



**3.5. ábra.** Az emelkedési szög meghatározható a gyorsuláskomponensekből. A forgó tárgyra függőleges irányban állandóan hat a nehézségi gyorsulás, melynek adott komponensét mindig mérni tudjuk, így vissza-számolható az emelkedési szög.

A finom mozgás elérésére megfelelő mechanikát kellett elkészíteni. A függőleges irányú forgatásért egy, a végén és közepén csapágyazott kar felel. A vízszintes eset valamivel bonyolultabb, ugyanis nagyobb, a forgás irányával párhuzamos terhelés éri a rendszert, így görgős csapágyat kellett alkalmazni. Költséghatékonyság céljából azonban megfelelően elhelyezett görgőket alkalmaztunk (3.6. ábra).



**3.6. ábra.** A forgatórendszer és a szenzor összeállítva. A két szabadsági fokú mechanika lehetővé teszi egy műhold bizonyos forgatásának imitálását.

Az első problémára a megoldást az jelentette, hogy elszigeteltük magunkat a környezettől. Levittük a kamerát egy föld alatti, egy ajtót leszámítva nyílászárómentes terembe, amiben ezáltal közel állandó hőmérséklet uralkodott. Azonban így a Nap nem volt látható. Erre volt megoldás egy forrasztópáka felhevített hegye, amely megfelelő távolságból nagyon hasonló képet mutatott, mint a Nap. A mozgás szimulálására pedig ugyanazt a 2 szabadsági fokú eszközt használtuk.

## 4. fejezet

# Képfeldolgozás

A projekt egy sarkalatos pontja a képfeldolgozás. Mivel nincs lehetőségünk a teljes képet leküldeni a Földre, így ezt kis erőforrásokkal kell megtennünk a fedélzeten. Éppen ezért fontos, hogy a lehető legjobb algoritmust használjuk.

A korábban említettek alapján két objektumot kell detektálnunk: Földet és Napot. Ennek megfelelően két részre bontottuk a képfeldolgozást is: a Nap-keresésre és a Föld-keresésre.

### 4.1. Nap-keresés

A Nap-keresés sok szempontból okoz nehézséget. Egyrészt az esetek egyharmadában a műholdhoz képest a Föld mögött található, másrészt pedig előfordulhat, hogy kívül esik a szenzorok által lefedett térrészen, így nem látjuk. Előbbit TLE-adatokból előre meg tudjuk jósolni, utóbbit viszont már csak az orientáció ismeretében. Ha azonban valamely szenzoron megjelenik, úgy egy fényes pontot tapasztalnánk.

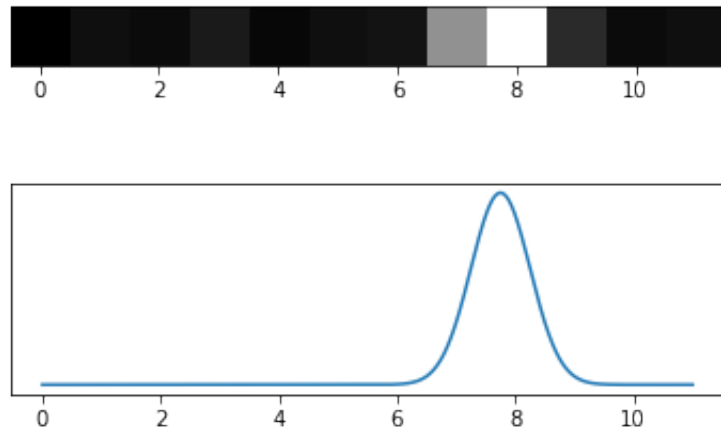
#### 4.1.1. Iterációs megoldás

A legegyszerűbb ám legpontatlanabb megoldás, hogy megvizsgáljuk az összes pixel értékét, és amennyiben valamelyik megugrik egy empirikusan előállított küszöbértéket, úgy azt Napként kezeljük.

A gond akkor kezdődik, amikor a Napból érkező sugárzás nem egy, hanem több pixelen oszlik meg. Ekkor előfordulhat, hogy egyik pixel értéke sem lépi át a küszöbértéket, ha azonban mégis, akkor is adódhat pontatlan eredmény.

### 4.1.2. A szubpixel-eljárás

Alacsony felbontás esetén tudjuk alkalmazni az úgynevezett szubpixel-eljárást[11]. Ennek a lényege, hogy a kép felbonásánál nagyobb pontossággal meg tudjuk határozni a detektálandó objektum képen vett pozícióját. A mi esetünkben ez tovább egyszerűsödik, ugyanis ismerjük a pixel értékét abban az esetben, amikor az összes Napból érkező sugárzás csak ezt érinti. Azaz egy függvény illesztésével nagy pontossággal meghatározható a Nap képbeli helyzete (4.1. ábra).



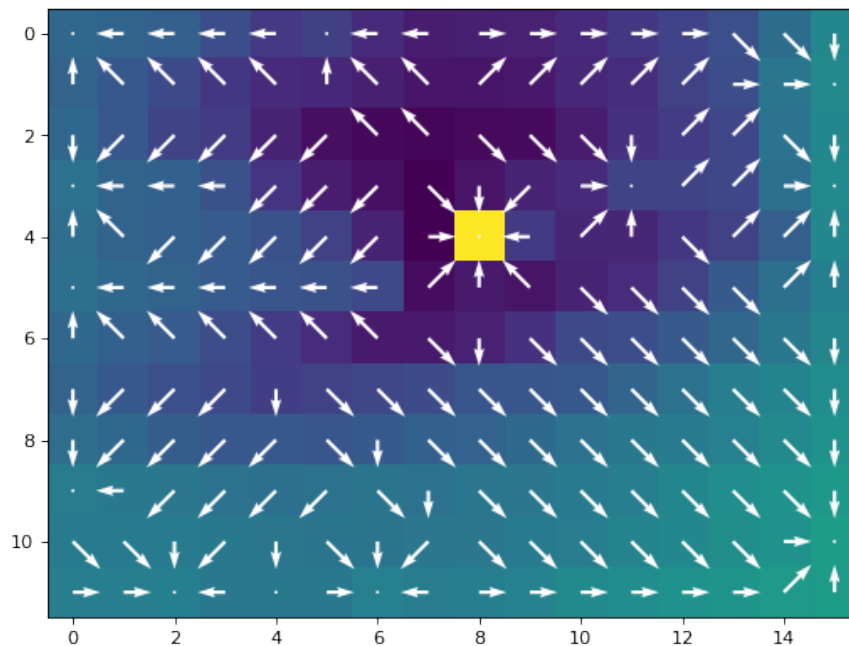
**4.1. ábra.** A szubpixel eljárás alkalmazása a Napon. A felső sáv egy pixel-sor kiragadva a képből, amelyen a Nap megtalálható. Ezzel a módszerrel egy görbét tudunk illeszteni a pixelértékeknek megfelelően, így meg tudjuk becsülni a Nap képen vett elhelyezkedését a szenzor felbontásánál nagyobb pontossággal.

### 4.1.3. Szomszédos pixelek módszere

Kipróbáltunk egy módszert, ahol minden pixelhez hozzárendelünk egy vektort, amely a szomszédos pixelek közül (beleértve a sarkuknál érintkező szomszédokat) a legfényesebbre mutat, ezzel kapva egy, az iteratív módszernél precízebb megoldást. Ezt a módszert a fotózott képeken teszteltük. Azt tapasztaltuk, hogy közvetlen a Nap körüli pixeleket leszámítva nincs rendszer a vektorok irányában (lásd: 4.2. ábra). Ebben az esetben viszont sokkal hatékonyabb ennél a módszernél az iteratív módszer.

## 4.2. Föld-keresés

A Föld-keresés sem egyszerű probléma, azonban a Nappal ellentétben lényeges kiterjedése van a képeken (azaz megjelenése jelentősen nagyobb, mint két pixel), illetve mindig van olyan kamera, amely detektálni tudja. Éppen ezért lehetőségünk van a napkereső



**4.2. ábra.** A szomszédos pixelek módszere. Minden pixelnek megvizsgáljuk a szomszédjait, és a pixelhez hozzárendelünk egy vektort, mely a legnagyobb értékű pixelre mutat, így gyorsítva a Nap-kereső algoritmust.

algoritmusoknál hatékonyabb eljárásokat kipróbálni. Ezidáig két módszert teszteltünk: az intervallumfelező módszert és egy ún. keretmódszert.

#### 4.2.1. Intervallumfelező módszer

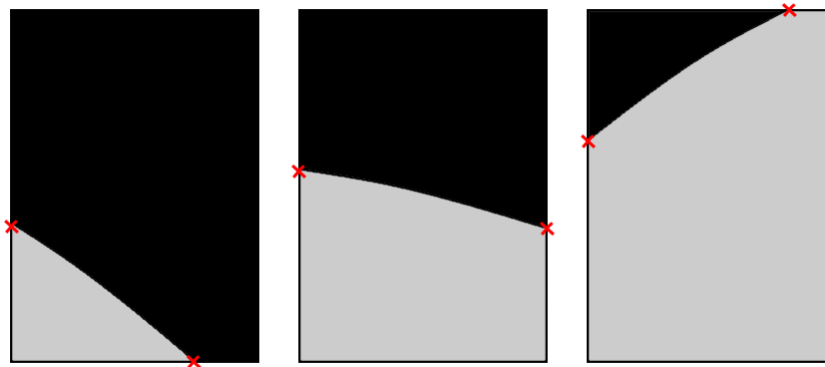
Ez a módszer két részre bontható: egyrészt meg kell határoznunk, hogy a kép mely éleit érinti a Föld, azaz melyek azok az élek, amelyeket a Földfelszín elmetsz (4.3. ábra). Ezek után pedig intervallumfelezéssel közelítjük ezeket a metszéspontokat, majd meghatározzuk az adott szenzorhoz tartozó Föld-normálist.

Megvizsgáljuk, hogy a képen melyek azok a sarkok, ahol a Föld sugárzásának megfelelő küszöbértéket átlépi az adott pixel. Ez alapján öt lehetőség adódik, melyből kettő nem tartalmaz számunkra használható információt: amikor egyik sarok sem lépi át a küszöbértéket, ekkor ugyanis a Föld, méreténél fogva, nem található a képen, illetve amikor mind a négy sarka átlépi ezt az értéket, ekkor pedig az egész képen a Föld található, így nem tudunk normálvektort meghatározni.

A további három lehetőség esetén mindig van kettő él, amelyet elmetsz a Föld felszíne. Ennek a módszernek köszönhetően ismerjük ezeket az éleket, intervallumfelezéssel pedig meghatározható a két metszéspont. A metszéspontok képen vett koordinátáit kivonjuk egymásból, majd az így kapott vektort megfelelő irányba elforgatva  $90^\circ$ -kal



megkapjuk a Föld normálisát. A forgatás irányát az érintett élek számából tudjuk meghatározni.

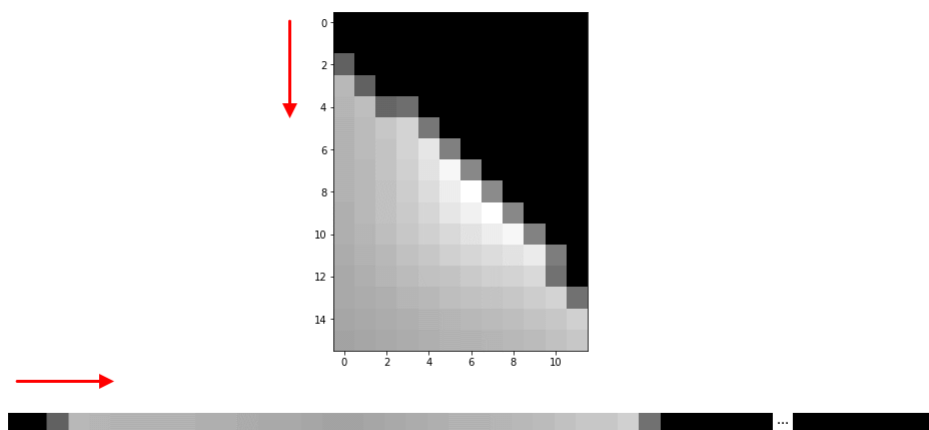


**4.3. ábra.** Sematikus ábra a Föld lehetséges megjelenéseiről a szenzorok képein. Két szélső esetben szomszédos, míg a középső esetben szemközt élek mentén kell keresnünk a Föld felszíne által metszett pontokat.

#### 4.2.2. Keretmódszer

Ez egy lényegesen egyszerűbb módszer a lépésszámot tekintve. Ahogy az elnevezés is sugallja, a kép szélén elhelyezkedő pixeleket használjuk fel. Körbehaladva a képkereten, a pixelek értékeit egy listába rendezzük úgy, hogy ismerjük az érték képen vett pozícióját. Ezután végignézzük a listát, hogy melyek azok a pixelek, amelyek átlépik a küszöbértéket. Az így kialakult szakasz kezdő- és végpontja megadja a metszéspontokat. Innentől pedig a normális meghatározás az előzőekhez hasonlóan meghatározható (4.4. ábra).

A képfeldolgozásban további nehézséget okoz a Nap- és Föld-kereső algoritmusok egyesítése. Mivel a módszerek igen különbözőek lehetnek a végleges rendszerben alkalmazottól, így ez optimális megoldást igényel. A kutatás képfeldolgozás szempontjából még korai fázisban van. A korábban ismertetett képfeldolgozó algoritmusokat, módszereket teszteltük, hogy megtaláljuk a céljainknak leginkább megfelelőt. Az eddig kipróbáltakon felül természetesen még rengeteg módszer létezik, azonban a téma bemutatásáig eddig sikerült eljutnunk.

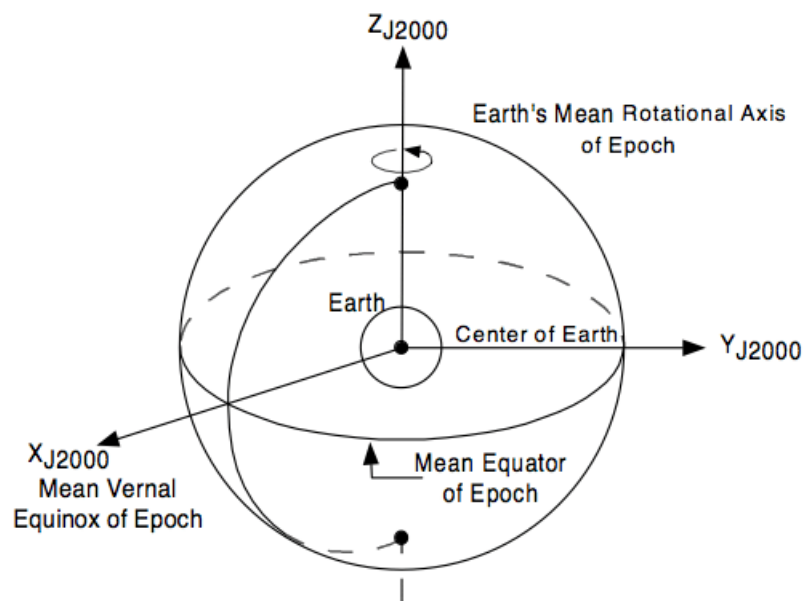


**4.4. ábra.** A keretmódszer. A kép szélein végighaladva megkeressük azt a szakaszt, ahol a Föld megjelenik. Ennek a kezdő- és végpontját összekötve megkapjuk a Föld horizontját.

## 5. fejezet

# Adatfeldolgozás

A műhold TLE-adatait a CelesTrak [12] oldaláról le tudjuk tölteni, ezek alapján sok fontos paraméterhez férünk hozzá. Amelyek az orientációmeghatározás szempontjából a legfontosabbak, azok a Föld-Nap és a Föld-műhold vektorok. Ezek a J2000 koordináta-rendszerben [13] vannak megadva. A J2000 egy Föld-központú, ortogonális koordináta-rendszer, ahol az x-tengely a 2000. január 1. UTC 12:00 időpontban a Föld középpontját és a tavaszpontot összekötő szakasszal, illetve a z-tengely a Föld forgástengelyével esik egybe, az y-tengely pedig a jobbsodrásnak megfelelően adódik (5.1. ábra).



5.1. ábra. A J2000 koordinátarendszer.

## 5.1. Kvaterniógenerálás

A műholdnak is van egy saját koordinátarendszere, melyben meg tudjuk határozni a műhold-Föld és műhold-Nap vektorokat az infraszensorok adatainak feldolgozása után. Ez a rendszer is ortogonális, így ha ismerjük a forgatásmátrixot, amivel áttérhetünk egyik rendszerből a másikba, akkor a műhold orientációját is meg tudjuk határozni. Ahhoz, hogy tesztelni tudjuk az adatfeldolgozó rendszerünket, ezt a forgatásmátrixot véletlenszerűen generáljuk, ezzel szimulálva a képfeldolgozásból kapott adatokat. Ez azonban nem egy triviális probléma, ugyanis a mátrix elemei összetett összefüggéseknek megfelelően adódnak. Megoldást jelenthet a következő kifejezés [14], amelynek köszönhetően az alábbi paramétereket tudjuk generálni:

$$q_r = \sqrt{1 - u_1} \sin(2\pi u_2) \quad (5.1)$$

$$q_i = \sqrt{1 - u_1} \cos(2\pi u_2) \quad (5.2)$$

$$q_j = \sqrt{u_1} \sin(2\pi u_3) \quad (5.3)$$

$$q_k = \sqrt{u_1} \cos(2\pi u_3), \quad (5.4)$$

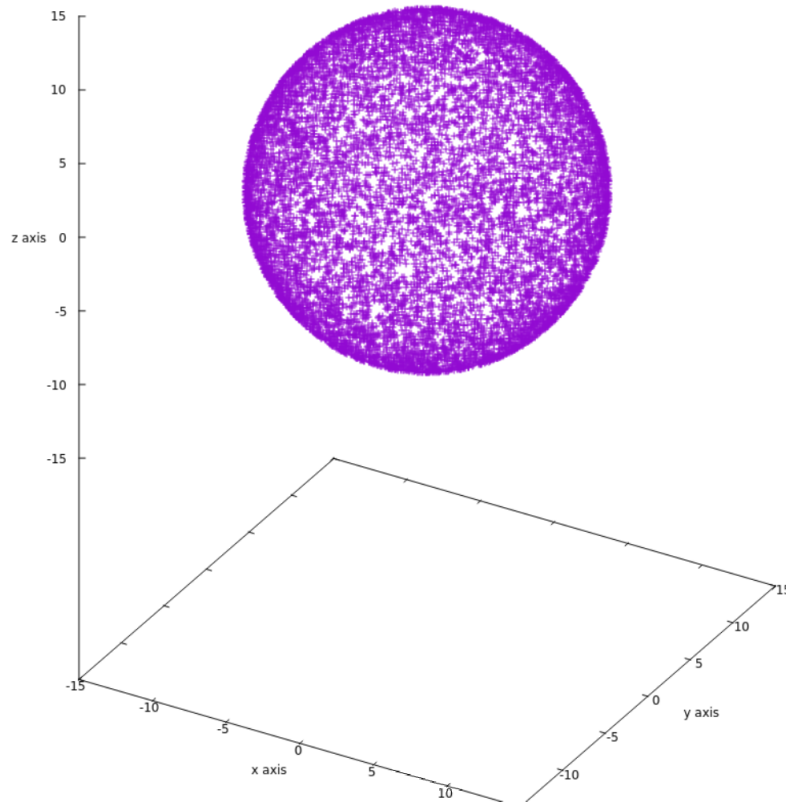
ahol  $u_1, u_2, u_3 \in [0, 1]$ , azaz egység-intervallumból vett egyenletes eloszlású véletlen szám. Ekkor megkapjuk a  $h$  kvaterniót:

$$h = (q_r, q_i, q_j, q_k). \quad (5.5)$$

Ezt pedig fel tudjuk használni  $\mathbf{R}$  forgatásmátrix meghatározásához a következő összefüggés [15] alapján:

$$\mathbf{R} = \begin{bmatrix} 1 - 2(q_j^2 + q_k^2) & 2(q_i q_j - q_k q_r) & 2(q_i q_k + q_j q_r) \\ 2(q_i q_j + q_k q_r) & 1 - 2(q_i^2 + q_k^2) & 2(q_j q_k - q_i q_r) \\ 2(q_i q_k - q_j q_r) & 2(q_j q_k + q_i q_r) & 1 - 2(q_i^2 + q_j^2) \end{bmatrix}. \quad (5.6)$$

Ennek a generálásnak a megfelelő működését és homogén eloszlását tudjuk ellenőrizni a következő módszerrel: veszünk egy háromdimenziós vektort, majd hattatjuk rá a véletlenszerű forgatás mátrixot. Ugyanezzel a vektorral megismételjük ezt a műveletet, minden alkalommal új mátrix generálásával. Akkor mondhatjuk, hogy a random forgatásmátrix-generáló megfelelő, ha a vektorok által meghatározott pontfelhő egy gömbfelület, és a pontok közel állandó távolságra vannak egymástól, mely a mi esetünkben jól láthatóan teljesül (lásd: 5.2. ábra).



**5.2. ábra.** Generált kvaterniók homogenitásának ellenőrzése. Egy vektorra hattatunk egy generált forgatásmátrixot, melyet többször megismétlünk. Ha a generálás megfelelő, és ábrázoljuk a transzformált vektorok végpontjait, úgy egy gömböt kell kapnunk. A képen egy ilyen tesztnek a végeredménye látható, a generálás módszere megfelelőnek bizonyult.

## 5.2. Bázistranszformáció

Tegyük fel, hogy meghatároztuk a műhold-Föld és műhold-Nap vektorokat a műholdon, illetve TLE-adatokból ismerjük a Föld-műhold és Föld-Nap vektorokat. Korábban tárgyaltak alapján a Föld-Nap vektor közelíthető a Föld-műhold vektorral, így a TLE-adatokból származó egységvektorok, a műholdon mért egységvektorok, és a közbezárt szögük is megegyezik.

Így azonban egy forgatásmátrixot kell meghatároznunk, amely ismeretében meg tudjuk adni a műhold orientációját. Ezzel a dolgozat leadásáig nem végeztünk, de az eddig elért részeredményeket a következőkben tárgyalom.

A keresett forgatásmátrix legyen  $\mathbf{M}$ , legyen  $e^{(sat)}$  a műhold-Föld egységvektor a műhold koordinátarendszerében, illetve  $e^{(J2000)}$  ugyanez az egységvektor a Föld J2000-es koordinátarendszerében. Az  $s$  műhold-Nap egységvektorok esetén ugyanez a jelölés a mérvadó. A bázistranszformációt meghatározó összefüggések:

$$e^{(sat)} = \mathbf{M}^T \cdot e^{(J2000)}, \quad (5.7)$$

$$s^{(sat)} = \mathbf{M}^T \cdot s^{(J2000)}. \quad (5.8)$$

Továbbá mivel a vektorkettősök bezárt szöge állandó, így a következő összefüggést is felírhatjuk:

$$e^{(J2000)} \cdot s^{(J2000)} = e^{(sat)} \cdot s^{(sat)}. \quad (5.9)$$

A J2000-es koordinátarendszer előre definiált és ortogonális. Ezért ezt létre kell hoznunk a műhold esetén is. Először kijelölünk egy irányt, ez legyen  $e_x^{(sat)}$ . Ekkor a bázis másik két komponense:

$$e_y^{(sat)} = \frac{s^{(sat)} \times e^{(sat)}}{|s^{(sat)} \times e^{(sat)}|}, \quad (5.10)$$

$$e_z^{(sat)} = \frac{e^{(sat)} \times e_y^{(sat)}}{|e^{(sat)} \times e_y^{(sat)}|} \quad (5.11)$$

Mely alapján a forgatásmátrixok:

$$\mathbf{M}^{(sat)} = \left( e_x^{(sat)}, e_y^{(sat)}, e_z^{(sat)} \right), \quad (5.12)$$

$$\mathbf{M}^{(J2000)} = \left( e_x^{(J2000)}, e_y^{(J2000)}, e_z^{(J2000)} \right) \quad (5.13)$$

Egy másik, hasonló esetekben alkalmazott módszer a Wahba probléma[16]. Ez egy kompakt összefüggést ad, mellyel szintén meghatározható a forgatásmátrix.

## 6. fejezet

# Összefoglalás

Ez egy rendkívül sok komponensű projekt, mely mostanáig is rengeteg bizonytalanságot hordoz magában. Ez az oka annak, hogy jelenleg is fut a kutatás, számos dolog még nincs tesztelve, illetve nem tisztázódott még minden a végleges formájában. Éppen ezért a dolgozat tartalma és megfogalmazása is néhol bizonytalan.

Éppen ezért ennek a dolgozatnak a célja a jelenleg is aktív kutatásunkban folyó tesztelések, ötletek és módszerek ismertetése, egy olyan környezetben, amelynél elengedhetetlen fontosságú az optimalizálás. Rengeteg kiaknázatlan és érdekes probléma vár még megoldásra, melyeket a következő hónapokban szeretnénk véglegesíteni. Az eszköz a jelenlegi tervek szerint az elsősorban műegyetemi fejlesztésű MRC-100 műhold egyik egységeként, tavasszal kerül felbocsátásra. Ennek eredményeit izgatottan várjuk.

# Irodalomjegyzék

- [1] Nanoműhold, 2022. [Letöltve - 2022.11.07.].
- [2] MRC-100, 2022. <http://eik.bme.hu/csurgai/urtech/Sources/Gschwindt-Szabo-nemzetkozi.pdf>, [Letöltve - 2022.11.07.].
- [3] MLX90641 adatlap, 2019. <https://www.melexis.com/en/product/mlx90641/high-operating-temperature-fir-thermal-sensor-array>, [Letöltve - 2022.11.07.].
- [4] STM32F072 adatlap, 2019. <https://www.st.com/en/microcontrollers-microprocessors/stm32f072rb.html>, [Letöltve - 2022.11.07.].
- [5] András Pál, Masanori Ohno, László Mészáros, Norbert Werner, Jakub Ripa, Marcel Frajt, Naoyoshi Hirade, Ján Hudec, Jakub Kapuš, Martin Koleda, et al. Grbalpha: A 1u cubesat mission for validating timing-based gamma-ray burst localization. In *Space Telescopes and Instrumentation 2020: Ultraviolet to Gamma Ray*, volume 11444, pages 825–833. SPIE, 2020.
- [6] Emilian-Ionuț Croitoru and Gheorghe Oancea. Satellite tracking using norad two-line element set format. *Scientific Research and Education in the Air Force-AFASES*, 1:423–431, 2016.
- [7] Unity, 2022. <https://unity.com/>, [Letöltve - 2022.11.07.].
- [8] Arduino Nano, 2022. <https://store.arduino.cc/products/arduino-nano>, [Letöltve - 2022.11.07.].
- [9] 28byj léptető motor, 2022. <https://www.digikey.co.uk/en/datasheets/mikroelektronika/mikroelektronika-step-motor-5v-28byj48-datasheet>, [Letöltve - 2022.11.07.].
- [10] MPU-9250, 2017. <https://invensense.tdk.com/products/motion-tracking/9-axis/mpu-9250/>, [Letöltve - 2022.11.07.].
- [11] Qi Tian and Michael N Huhns. Algorithms for subpixel registration. *Computer Vision, Graphics, and Image Processing*, 35(2):220–233, 1986.
- [12] TLE adatbázis, 2022. <https://celestrak.org/NORAD/elements/>, [Letöltve - 2022.11.07.].
- [13] V. Vittaldev, E. Mooij, and M. C. Naeije. Unified State Model theory and application in Astrodynamics. *Celestial Mechanics and Dynamical Astronomy*, 112(3):253–282, March 2012. DOI: 10.1007/s10569-011-9396-5.
- [14] Random kvaternió generáló algoritmus, 2020. <http://lavalle.pl/planning/node198.html>, [Letöltve - 2022.11.07.].



- [15] Drazen Svehla. Earth orientation quaternion. In *Geometrical Theory of Satellite Orbits and Gravity Field*, pages 355–361. Springer, 2018.
- [16] Grace Wahba. A least squares estimate of satellite attitude. *SIAM Review*, 7(3):409–409, 1965. DOI: 10.1137/1007077.