# K+F projekt dokumentáció
# ACSG Kft.

## Időszak:

2023.03.01. - 2023.04.30.

Készítette: Wenesz Dominik

Advanced Cableharness Solution Group

# Tartalomjegyzék

# 1. Mérföldkövek

A hagyományos objektumdetektáláson alapuló módszereket elvetettük, mint opciót a projekt megoldására.

Sikerült kiválasztani a csatlakozóházakhoz megfelelően univerzális, kompakt végberendezés típust.

A Deep learningen alapuló objektumdetektálási módszerekkel sikeresen tudunk detektálni több osztályhoz tartozó objektumokat.

A Deep learning nagy adatigényét kielégítő, ipari környezetbe flexibilisen illeszkedő adatgeneráló szoftvert sikerült kialakítani (GUI még nem teljes).

# 2. Összefoglaló

A dokumentum által tárgyalt időszakban a deep learning alapú objektumdetektálási és pozíciómeghatározási rendszer véglegesítése zajlott (zajlik), egyúttal megkezdtük a valós környezetben, az edge device-on való teszteléseket is.

Egyúttal a GUI és az azt kiszolgáló háttérrendszer fejlesztése is megkezdődött, melyben először a tetszőleges 3D CAD modelből egyszerűen történő tanítóadat generálás, valamint az ezen adatokon történő neurális háló tanítása, elátrolása, illetve az edge device-ra történő hálózaton keresztüli feltöltés valósul meg.

(...)

# 3. Objektum detektáló alrendszer

## 3.1. Rendszer gyártókörnyezetbeli működésének leírása

Mivel az objektumdetektálás egy kellőképpen működő stádiumban tart, ezért célszerű megemlíteni a rendszer tervezett viselkedését, kezelését röviden.

- A kamerarendszerből kapott jelet a feldolgozóegységen futó szoftver (ami a neurális hálót és a megfelelő algoritmusokat tartalmazza) megállíptja, hogy a munkatér mely koordinátáiban és milye pozícióban helyezkednek el csatlakozóházak, illetve a robot számára melyek a felvehetőek.
- A feldolgozóegységből kiadott jel alapján pedig a robot (a robotvezérlőn keresztül) felveszi az objektumot és a kívánt koordinátába és orientációba állítja azt.
- A GUI felületen adható meg a kívánt program (csatlakozóház típus, mennyiség stb.), illetve ide feltöltve adott CAD file-t megtörténik a tanítóadat generálás és az új háló tanítása.
- További funkciók részletezése később történik.

## 3.2. Szöghelyzet meghatározás síkokban

(...)

## 3.3. További fejlesztési irány

### 3.3.1. Adatgenerálás további fejlesztések

(...)

### 3.3.2. Hálóoptimalizálás

(...)

### 3.3.3. Szimulációs implementáció

(...)

#### 3.3.4. Orientáció detektálás további vizsgálata, tesztelése

(...)

# 4. Robot manipulátor

## 4.1. Precíziós tesztek

Tekintve, hogy az objektum detektálás területén a manipulátor számára felvehetőséget, illetőleg a csatlakozó típusát is megfelelő biztonsággal meg tudjuk állapítani, így a valóságban való tesztelés is kezdetét veszi a következő időszakokban a SCARA robottal. Azonban hiába működik megfelelően a detektáló rendszerünk, ha a megállapított koordináta és pozíció a digitális feldolgozórendszeren (robotvezérlő) az aktuátorokkal nem képes megfelelő hibahatáron belül pozícionálni a manipulátoron található végberendezést. Ezen problémaforrás kiküszöbölésére a rendszertesztek előtt szükséges önmagában a robotvezérlő és a manipulátor együttes pontosságát mérésekkel megállapítanunk, jóságát validálnunk.

Megjegyzendő, hogy a munkaterület eltérő pontjaiban ez igen különböző is lehet, jellemzően a szingularitási pont(ok), illetve a szélső helyzetekben és azok környezetében nagyobb pozíciós hibát vét a robot.

Az ilyen jellegű tesztek szükségességét egyben az is indokolja, hogy a modellezésből adódó esetleges halmozódó hibát kivédhessük, továbbá a rendszerről (gyártócella) készülő digitális iker kellően közelítse a valóságot.

Azonban egy-egy ilyen tesztelési folyamat kidolgozása nem egyszerű feladat, hiszen mindenképp objektív, meghízható mérésekre van szükségünk. Továbbá fontos szempont, hogy a mérések lehetséges részeit automatizáljuk, mivel egy manipulátor teljes munkaterületének többszöri pontossági feltérképezése rengeteg humán erőforrást igényel.

Először a megfelelő mérési mód megtalálása, annak validálása, majd automatizálásának lehetőségeinek vizsgálata a kitűzött célunk.
(...)

## 4.2. ROS környezetbeli implementáció

A ROS (Robot operation system) egy nyílt forráskódú middleware, mellyel bármilyen robot szimulálható és egyben irányítható is. Előnyei közé tartozik, hogy több programozási nyelven (pl.: Python, C++) is programozható, minden könyvtár implementálva van mindegyik támogatott nyelvi csomagra. A magasszintű irányítástól egészen a bitszintű kezelésig megvalósíthatók benne a műveletek. A nemrégiben újradolgozott API (ROS 2.x) pedig lehetővé teszi a valós idejű robotvezérlést.

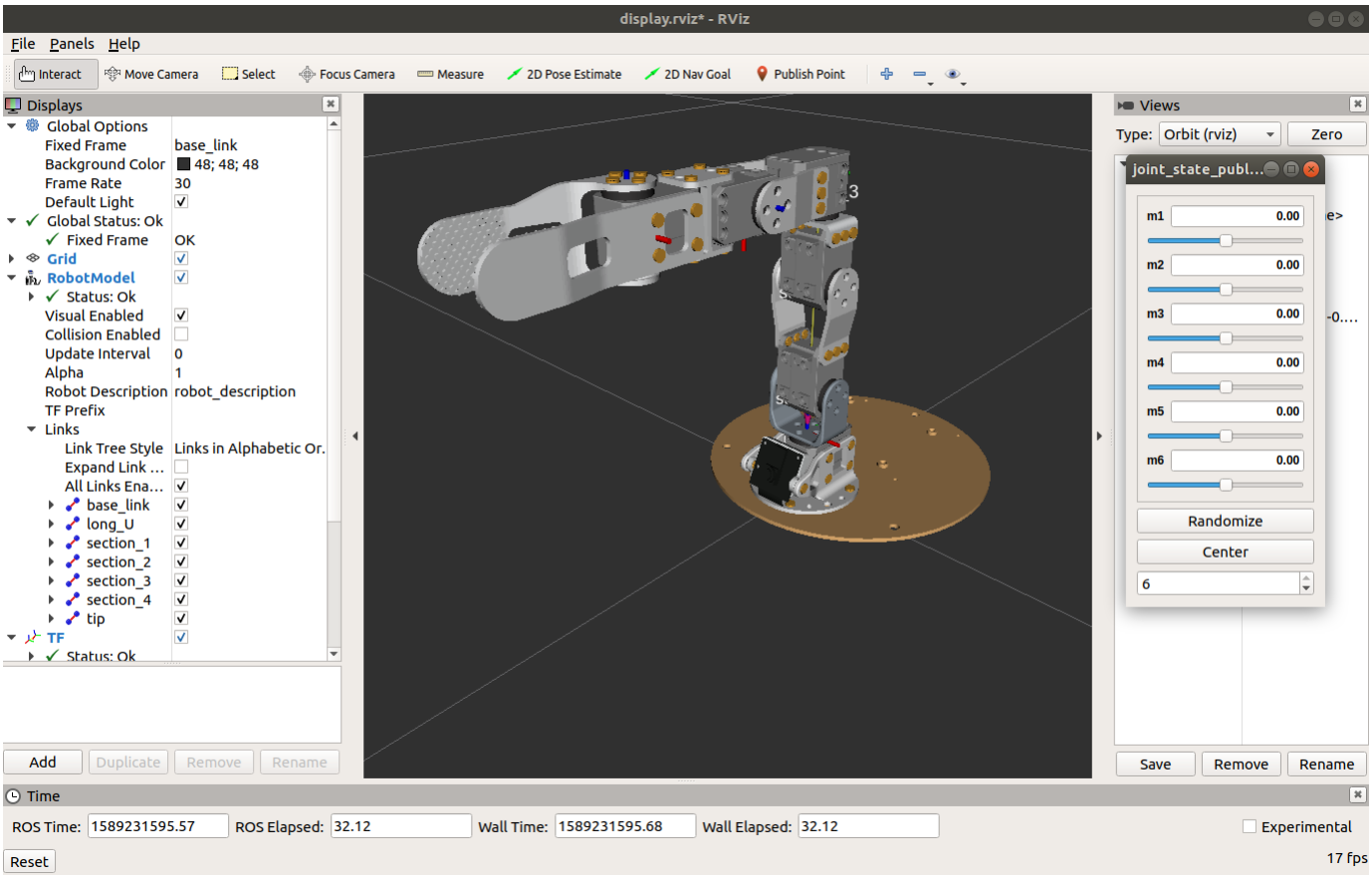
A projekt szempontjából két okból releváns számunka ez a szoftver:

- A rendszerről készülő digitális iker egy részét (közvetlen a robotmanipulátor digitális ikrét) el tudjuk benne készíteni, mely valós/majdnem valós időben képes irányítani, optimalizálni a manipulátor mozgatását. A kamerarendszerrel és további szenzorokkal kombinálva pedig olyan funkciók ellátására is alkalmas, mint például az ütközés elkerülés.
- Másfelől szimulációk futtatására alkalmas környezet biztosít számunkra, melyben a robotvezérlésre szolgáló algoritmusokat, szoftverrészeket a fizikai robot bevonása nélkül tudjuk tesztelni. Az ilyen valós rendszerbeli implementáció előtti virtuális futtatások kiemelkedően fontosak, hiszen egyrészt védjük vele az emberi és tárgyi épséget, továbbá a valós időnél gyorsabban, párhuzamosan, valóságban ritkán előforduló, nehezen reprodukálható eseteket is meg tudjuk így vizsgálni. Példaképp ha valamilyen anomália miatt a programunk hosszú idő után, vagy specifikus input kombinációra nem elvárt működést produkál, akkor a valós tezst előtt tudjuk észlelni és javítani a problémát.

A projekt jelenlegi szakaszában elkezdtük a ROS megismerését, elsajátítását, illetve ezzel párhuzamosan a használt SCARA robot implementációját is elkezdtük.

A precíziós tesztek természetesen a ROS renszerben elkészülő modellre is vonatkoznak majd.
(...)

# 5. Irodalomjegyzék




1. ábra. ROS rendszer mintakép