# K+F projekt dokumentáció
# ACSG Kft.

## Időszak:
2023.01.01. - 2023.02.28.

Készítette: Wenesz Dominik

Advanced Cableharness Solution Group

# Tartalomjegyzék

# 1. Összefoglaló

A dokumentum által tekintett időintervallumon belül megtörtént a robotkar (SCARA) pontosságának tesztelése a munkatérben, melynek eredményeként elmondható, hogy a csatlakozóházak megfelelően precíz manipulálásához megfelelő az alkalmazott robotkar.

A pontosság mérésének tükrében a későbbi digitális iker tervezése is megkezdődött, de a tekintett időszakban mindösszesen a szimulációs keretrendszer beállítása indult el egy Linux operációs rendszerű PC-n.

A mélytanulásos objektumdetektálás tanítóadat generáló szoftvere a jelenlegi teszt alkalmazásokhoz teljesen elkészült, a későbbiekben a rendszerszintű integráció kapcsán lesz csak szükséges további változtatás eszközölése rajta.

Folytattuk a különböző Deep Learningen alapuló objektum detektáló hálóarchitektúra vizsgálatát a csatlakozóházak detektálására.

Az objektum detektáláson túl a pozíciómeghatározás lehetséges megoldásait vizsgáltuk, melynek eredményei lentebb láthatók, illetve az objektum detektálást részletesen tárgyaló dokumentumba felvezetésre kerültek.

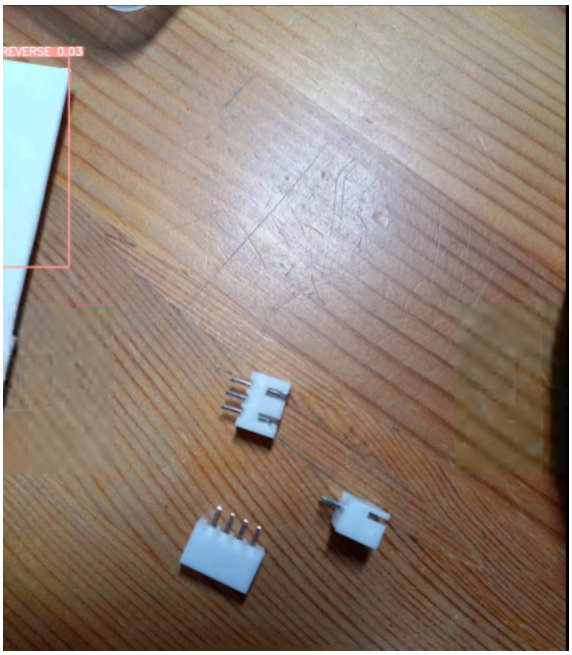
# 2. Objektum detektáló alrendszer

Az előző dokumentumban kifejtésre került, hogy a deep learning egy jó alternatívát jelent a hagyományos képfeldolgozási módszerekre többek közt az objektumdetektálás területén belül is.

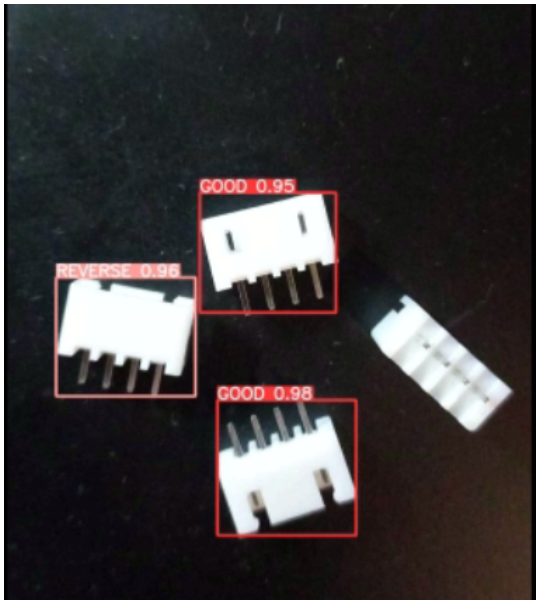
## 2.1. Vizsgált hálóarchitektúrák

Elsősorban a különböző YOLO architektúrákat vizsgáltuk először valós adatokon (kisebb adathalmaz), majd szintetikus képeket, illetve vegyes adathalmazon is.
Jó eredmények születtek, ezek részletezése az összefoglaló dokumentációban tekinthető meg, illetve a tezstelési jegyzőkönyvekben.

Megjegyzésként pár képpel demonstrálva a haladást:




1. ábra. Nem megfelelően generált képek esetén tanítás után




2. ábra. Megfelelően generált képek esetén tanítás után

# 3. Tanítóadat generálás

## 3.1. Probléma ismertetése

A deep learning alkalmazhatósága szinte bármilyen tudományterületen megkérdőjelezhetetlen, struktúrált és struktúsálatlan adatra is egyaránt használható. Jellemzően két robléma merül fel ha a mélytanulásról beszélünk, az egyik a nagyságrendekkel nagyobb tanítóadatot igény, illetve a tanításhoz szükséges számítási kapacitás és idő.

Könnyedén belátható, hogy egy felxibilis objektumdetektáló rendszerhez nagyszámú képre van szükség, adott esetben gyakran, ez ha emberi erőből végezzük időben és kiadásokban is nagy igényekkel rendelkezik.

## 3.2. Megoldási lehetőségek

Egy megoldási lehetőség a fentebb említett adatigényre, ha szintetikusan generáljuk azokat. Természetesen ez már önmagában hibákat szülhet, hiszen a generált képekkel a valóságot lefedni igen nehéz feladat tud lenni. A képgenerálás mellett az adatdúsítás is igen hasznos eszköz ebben a problémakörben, ez csupán annyit jelent, hogy képfeldolgzási technikákkal manipuláljuk a tanítóadatokat (zaj bevezetése, elforgatás stb..) és így sokszor nagyobb adatbázishoz jutunk.

Az adatdúsítás viszonylag egyszerű feladat, bevett módszerek vannak rá, azonban a szintetikus képgenerálás (azaz számítógéppel történő szimuláció gyakorlatilag) már korántsem ilyen triviális.

Egy további tényező mely nehezítést von be az, hogy a képgeneráásnak viszonylag gyorsan is kell történnie (legalább olyan gyorsan, mintha valós képeket készítenénk), azaz egy jó választás lenne valamilyen game-engine használata például a Unity-é, hiszen ezek nagyon gyors renderelésre lettek kifejlesztve. Azonban nagyon fontos tényező a valóság reprezentálása, lefedése és tipikusan a számítógépes játékokat meghajtó motorok csak addig generálják a képeket, amíg kellőképpen valósnak, szépnek látszik az emberi szem számára, ez azonban még nem a valós képhez haosnlító eredményt jelent, több fizikai tényező is elhanyagolásra kerül.

Az általunk használt megoldás egy fizikai renderelésen alapuló nyílt forráskódú szoftver és egyben könyvtár a Blender használatára épül. Tetszőlegesen komplex kép generálható benne, mely azon fizikai paramétereket is figyelembe veszi, amiket a fentebb említett gyorsabb környezetek nem. Emellett a sebessége is elfogadható, GPU-kra optimalizált, így egyszerre több szálon párhuzamosan tud futni a renderelés.

Eze szoftverre építettük még egy nyílt forráskódú segédkönyvtár bevonásával a képgeneráló programunkat, melynek egy CAD file és bemeneti paraméterek (hány képet generáljunk stb..) kell csupán és tetszőleges tanító adatot tudunk létrehozni vele (jellemzően egy képhez 5-30 másodperc szükséges).

Fontos még megemlíteni, hogy a hálókhoz nem elég csupán a nyers képek megléte, azokat címkézni is kell, azaz másik fájlokban megadni, hogy egy-egy adott képen milyen osztályú objektum és hol helyezkedik el, ez humán munkával szintén igen költséges lenne, de a program ezt is megoldja. További részletesebb leírás és eredmények az objektumm detektálás dokumentációban.

3. ábra. Generált adat

### 3.3. Vegyes tanító adat

A szintetikus adatok mellett ha valósakat is tartalmaz az adatbázisunk, az legtöbbször a neurális háló eredményességében javulást eredményez.

### 3.4. Valós adatra illesztés

Egy-egy adott hálót a szintetikus vagy vegyes adatokon tanítás után finomhangolni lehet ha megfelelő minőségű valós képeket tanítjuk tovább jóval kevesebb ideig.

### 3.5. További fejlesztési irányok

Egy másik igen érdekes fejlesztési irány a képeknek neurális hálókkal történő generálása.

## 4. Pozíciómeghatározás

A fentebb említett hálóarchitektúrák önmagukban mindösszesen az objektumdetektálásra képesek, mint azt az előző részekben megállapítottuk, jelen alkalmazási formára a sebesség igényénél fogva elegendő a bounding-box típusú objektu felismerést alkalmaznunk, hiszen elegendő adatot tartalmaz a robotvezérléshes és szignifikánsan gyorsabb, mint a szegmentálással is kiegészített hálóarchitektúra.

Azonban a megfelelő fészekbe való pozícionáláshoz további input paraméterekre van szükségünk, az objektum elfordultsága is nélkülözhetetlen paraméter ha egy kamerával szeretnénk megoldani a feladatot. Azonban itt több lehetőségünk is van a megvalósításra, a dokumentum által tárgyalt időszakban ezen lehetőségeket térképeztük fel és alkalmazhatóságukat az adott problémán elkezdtük megvizsgálni.
Az általunk megfontolandónak talált megközelítések a következők:

- Egy a munkaterület felett elhelyezett kamerától különálló kamera, mely a fészek felett van elhelyezve és hagyományos képfeldolgozási technikával (pl.: szögkülönbség meghatározás) megfelelő rotációs értéket adja a robotvezérlőnek. Természetesen léteznek nem explicit szögmeghatározó algoritmusok is, melyek egy adott hibaminimalizációra, vagy mintakövetésre törekszenek, azaz kvázi regresszív módszerrel közelítik a megfelelő orientációt, mellyel a fészekbe kell kerülnie a csatlakozóháznak.
  Ez a megoldás viszonylag egyszerű, azonban mivel két kamera szükséges hozzá (munkatér nagysága miatt is), illetve nem minden csatlakozó esetén feltétlenül hibamentesen generalizálható, ezért alkalmazhatóságát meg kell vizsgálni részletesebben.

- Az előző esetet egy kamerával is megoldhatjuk ha azt a robotkar és a munkatér, illetve a fészek geometriája lehetővé teszi.

- A hagyományos képfeldolgozási módszereken túl ebben az esetben is használhatunk neurális hálózatokat, azaz deep learingen alapuló módszereket.
  Egyik módszer, hogy már az objektumdetektáló neurális hálózatot bővítjük ki egy orientációt meghatározó résszel. Ekkor annyi szabadságunk van, hogy tetszőleges kondíciókat adhatunk meg, azaz lehetséges a fészekbe való behelyezés több variációval való kiegészítése, illetőleg regressziós módon megadhatunk 1-3 elfordulási komponenst (tengelyek körül), de tekintve, hogy SCARA robottal dolgozunk, így elegendő egy elfordulással operálni ha már megállapítottuk az objektumról, hogy felvehető.
  A rendszer ilyen szintű bővítésével eljutunk oda, hogy implicit meghatározzuk az objektum 3D térben vett 6 szabadságfokú pozícióját, mely a mélytanulás ilyen típusú alkalmazásában egy teljesen külön álló kutatási terület, melyre léteznek megoldási módok, melyeket a későbbiekben tesztelni is fogunk, azonban jelen alkalmazásnál SCARA robotnál egyszerűbb modellel is képesek vagyunk elérni a mefelelő pozícionálást elméleti szinten.

- A pozíció meghatározása nem feltétlen kell, hogy explicit módon történjen, ha egy adott objektum osztályon belül úgynevezett keypoint-okat keresünk, akkor azok elhelyezkedéséből determinálható az objektum pozíciója. Ennek előnye, hogy rengeteg információt tudunk meghatározni egy lépésben, de jellemzően egy ilyen háló betanítása több tanítóadatot igényel és nagyobb osztályszám esetén nem feltétlenül konvergál a valósághoz, továbbá természetesen a végrehajtási idő is jeentősen megnő a háló komplexitásával együtt.

#### 4.0.1. Megvizsgált módszerek

Az adott időintervallumban egyelőre főként csak a lehetséges megoldások elméleti kidolgozása zajlott, illetve a hagyományos képfeldolgozáson alapuló módszerek közül elkezdtük az algoritmusok implementációját is, persze az OpenCV-ben találhatók hasonló algoritmusok, de a feladat specifikussága miatt mindenképp akár ezeket felhasználva is, de saját algoritmust fejlesztenünk a robosztusabb megoldáshoz.

# 5. Robot manipulátor

Miután már kiválasztásra került a technológiai szempontból optimális végberendezés, annak valódi környezetben való tesztelését végeztük, továbbá magának a robotvezérlőnek a beállítása, kalibrálása, a robot felprogramozása és egyszerűbb illetve összetettebb mozgások megvalósítása LUA programnyelven.

Többek között a maximális csuklógyorsulást és csuklósebességet teszteltük. Ezek a robot adatlapjában is meg vannak adva számszerűen, azonban mindig érdemes az ilyen jellegű validálás, főként mivel az adatlapok az ideális, adott modellezési módhoz tartozó értékeket tartalmazzák.
(...)

# 6. Irodalomjegyzék