K+F projekt dokumentáció
ACSG Kft.

Időszak:
2022.11.01. - 2022.12.31.

Készítette: Wenesz Dominik

Advanced Cableharness Solution Group

Tartalomjegyzék

1. Összefoglaló

A dokumentum által specifikált időintervallumban a detektáló rendszerben történtek főbb előrelépések. Először is a későbbi skálázhatóságra való tekintettel hardveres területen változtattunk a feldolgozó egységet és kamerarendszert illetőleg. Ezenfelül a hagyományos képfeldolgozási módszerek után a mélytanuláson alapuló módszerek applikálhatóságát vizsgáltuk.
A másik terület, melyben munkálatok folytak, az a manipulátor megfogójának milyenségének kiválasztása, többféle csatlakozóházon való tesztelése. Ez explicit nem a robotkar pontosságának mérését jelenti, hanem a kereskedelemben fellelhető végberendezések egymással való összehasonlítását (pl.: strapabíróság, felvehető objektum paraméterei, felvehető objektumok számossága, objektumdetektáló rendszerre való illeszthetősége, stb.). A robot pontosságának vizsgálata a következő két hónapos indőintervallumra esedékes.

2. Objektum detektáló alrendszer

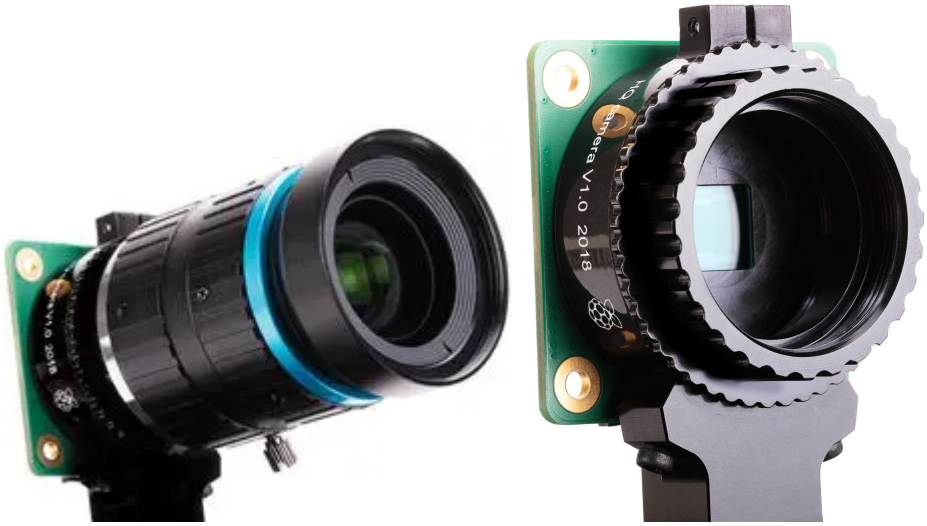
2.1. Hardver változtatások

Az objektum detektálás szempontjából a Jetson Nano és a RaspberryPi V2.1-es kamera elegendőnek tekinthető, azonban azon megfontolásból, hogy a későbbiekben a rendszer bővíthető legyen, esetlegesen párhuzamosan több feladatot is el tudjon látni, például egyszerre több különálló kamerakép alapján több egymástól független párhuzamos program/algoritmus futhasson párhuzamosan valós időben, vagy egy komplex, egész gyártócellára kiterjedő optimalizációs rendszer is futtaható legyen, arra a konklúzióra jutottunk, hogy egy nagyobb erőforrásokkal rendelkező feldolgozó egységet használunk. Így esett a választás a Jetson Xavier NX-re, melynek adatlapja megtalálható mellékelve a dokumentációs mappában.

1. ábra. Jeston Xavier NX Edge System

Ezen modul előnye, hogy több memóriát, nagyobb számítókapacitású CPU-t és nagyobb teljesítményű GPU-t tartalmaz, melyek amint azt későbbiekben láthatjuk a mélytanulásos objektumdetektáló módszerekhez elengedhetetlenek.

A kamera modul választásában szintén változtattunk, mivel a RPi V2.1 képminősége a kisebb objektumoknál már nem bizonyult megfelelőnek, illetve nem bővíthető gyárilag lencserendszerrel, így precíz hangolása nehézkes. Ezen indokoknál fogva választottuk a RaspBerryPi HQ detektort a hozzá tartozó lencserendszerrel (optikával) együtt, melyet könnyedén tudunk precízen hangolni. A szenzor és az optika adatlapja szintén megtalálható a dokumentáció megfelelő almappájában.



2. ábra. Optikával és optika nélkül felszerelt RPi HQ kamera

2.2. Deep Learning

A deep learning (mélytanulás) a gépi tanulás egy területe, melyben neurális hálókat használnak. Rendkívül számításigényes a modellek tanítása, de megfelelően komplex hálóval bármilyen függvény közelíthető (létrehozható), azaz a nemlineáris klasszifikáció is elvégezhető velük, ezért objektumdetektálásra megfelelő architektúrával alkalmasak.

2.3. Objektum detektálás mélytanulás alapon

A neurális hálók alapján történő objektum detektálás célja, hogy a neurális hálóba beérkező inputból (kép) a háló kimenetén olyan struktúrát kapjunk, mellyel leírható az objektumok pozíciója a képen, az alábbi reprezentációk a legnépszerűbbek:

2.3.1. Bounding box

A bounding box detection esetében az egyes osztályokhoz tartozó objektumokat befoglaló téglalalpokat keresünk a neurális hálóval, ez sokszor elegendő és gyorsabb a többi detektálási lehetőségnél.

2.3.2. Semantic segmentation

Ebben a szegmentálási típusban minden egyes pixelhez valamilyen osztályt rendelünk, ez lehet a háttér (BG) is (komplexebb, magasabb rendű mint a bounding box). Önvezetés témakörében népszerű alaprendszert biztosít.

2.3.3. Instance segmentation

Ebben az esetben csak a felismert osztályokhoz tartozó objektumokat reprezentáló pixelek kerülnek detektálásra (outputon csak ezeket látjuk).

2.3.4. Keypoint detection

A keypoint detection esetében egy képen egyrészt az egyes class-okhoz tartozó objektumokat szeertnénk detektálni, illetve egy-egy osztálynak tetszőleges nevezetes pontjait is precízen detektálni szeretnénk. Például egy egyszerű rgb kamera által összetett humán mozgások vizsgálata esetén gyakran alkalmazzák napjainkban. A projekt szempontjából az orientáció és elhelyezkedés könnyedén származtatható ilyen módszerrel.

2.4. Projekt során nem vizsgálandó hálóarchitektúrák

Az olyan jellegű ipari feladatoknál, melyeknél egy adott gyártógép/gyártócella kiszolgálása zajlik valamilyen robottal, kiemelten fontos a precizítás mellett az időhatékonyság, azaz jelen esetben, hogy közel valós időben képesek

legyünk megállapítani a munkatérben tartózkodó csatlakozóházakról, hogy milyen típusúak, továbbá felvehetőeke a manipulátorral. Ennélfogva bár mindenképp megvizsgálásra kerül a későbbiekben, de első közelítésben nem foglalkozunk olyan hálóarchitektúrákkal, melyek nem képesek közel real-time működésre, hiszen ekkor egy olyan nagyságú τ időkésést adnánk a rendszerünknek, mellyel szinte biztosan instabil lenne, azaz olyan módosításokat kellene eszközölnünk az egész rendszeren, melyek kompenzálják ezt a hatást. Példaként megemlíthető, hogy ekkor meg kellene állítanunk a futószalagot, amelyen az új megmunkálandó csatlakozók érkeznek, ez nagyban vissza tudja fogni a termelékenységet, időegységre arányosított kihozatalt.

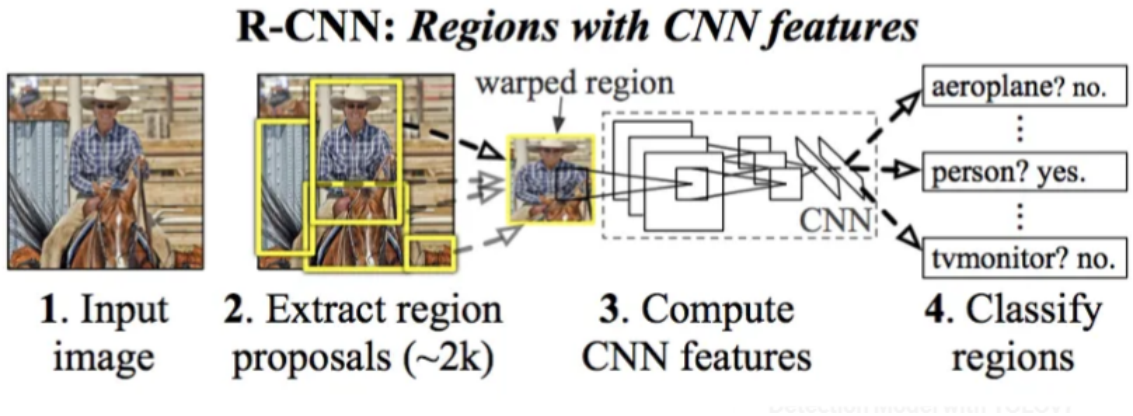
2.4.1. RCNN és Faster-RCNN

AZ RCNN hálóarchitektúrák alapját a konvolúciós layer-ek képzik, melyek a már tárgyalt módon lokális tulajdonságokat, pixelek közti tulajdonságokat képesek kinyerni a képből. A konvolúciós layer-ek előtt azonban az ilyen architektúrákban egy úgynevezett "region extractor" helyezkedik el, melynek feladta, hogy a képet kisebb részekre ossza, mégpedig olyan formán, hogy olyan régiókat kapjunk, melyekben valószínűsíthetően egy-egy objektum helyezkedik vagy éppen egy objektum sem helyezkedik el.

A mélyebb technikai részletektől eltekintve elmondható, hogy több megoldás is létezik a régió kinyerésre, de minden esetben kétlépéses objektumdetektálásról beszélünk, hiszen az objektum klasszifikáció és pixelszintű regresszió csak a háló 2. (konvolúciós) részében valósul meg.
Ennélfogva mint a többi kétlépéses objektum detektáló háló, valós időben nem képesek működni, továbbá az RCNN-ek egy sajátossága, hogy tanításuk kifejezetten nehézkes főként az adatelőkésztés miatt.

A FASTER-RCNN ahogy azt a név is mutatja egy gyorsabb, de még mindig valós időhöz képest lassú hálóarchitektúra.

Ennélfogva a projekt szerves részét nem képzi az RCNN-ek vizsgálata.



3. ábra. RCNN hálóarchitektúra sematikus ábra

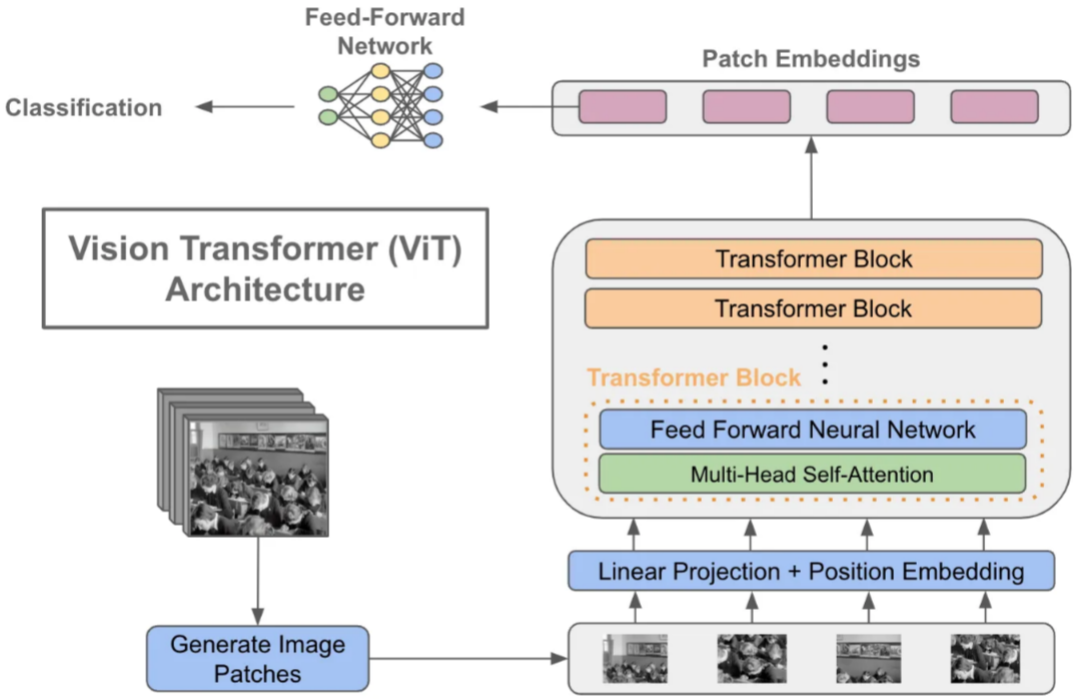
2.4.2. Vision - transzformerek

A deep learning technikákban a transzformerek először a természetesnyelv feldolgozásban kerültek kifejlesztésre és alkalmazásra is, majd a legtöbb alkalmazási területen megjelent, többet között a konvolúciós hálókat is jó ideig maguk mögé utasították a vision-transzformerek. Az utóbbi időben ismét a konvolúciós hálók javára kezd billenni a mérleg. A transzformerek működéséről jelen dokuemntumban nem teszünk említést, csupán addig a szintig, hogy sebességüknél fogva az RCNN-hez hasonlóan nem valós időben operálnak.

2.5. Vizsgált hálóarchitektúrák

2.5.1. Egyszerű konvolúciós háló - U-net

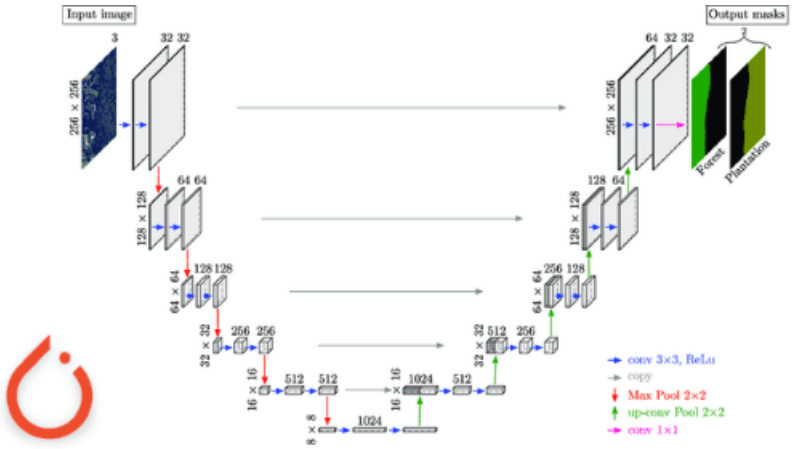
Az deep learning alapúobjektumdetektálás egyik legelső implementációja az úgynevezett U-net volt, mely konvolúciós layer-ekből áll. A bemeneti képet konvolúciós és dropout layereken keresztül egy jóval kisebb dimenziójú úgynevezett feature vektorba transzformáljuk, majd a háló kiemenete felé közeledve inverz transzformációt hajtunk végre a konvolúciós layerekkel, így az eredeti kép dimenziójának megfelelő kimenetet kapunk.



4. ábra. Transzformer hálóarchitektúra sematikus ábrája

A feature vektorba kódoló részt encodernek, az utána következő részt decodernek nevezzük.
Belátható, hogy így megfelelő tanítóadattal megoldható az objektumdetektálás, azonban szintén észlelhető, hogy egyszerűségénél fogva nem feltétlen a legjobb megoldás.

Sebességét tekintve megfelelően kis paraméterszámmal gyors működés érhető el (természetesen a pontosság rovására).



5. ábra. U-net architektúra

2.5.2. SSD

Az SSD (Single-Shot-Detector) egy olyan hálóarchitektúra, mely egy lépésben hajtja végre az objektumdetektálást, így lehetséges vele a közel valós idejű objektumdetektálás.

A háló működésének részletes leírására az objektum detektálás dokuemntációban kerül sor, röviden az alapgondolat,

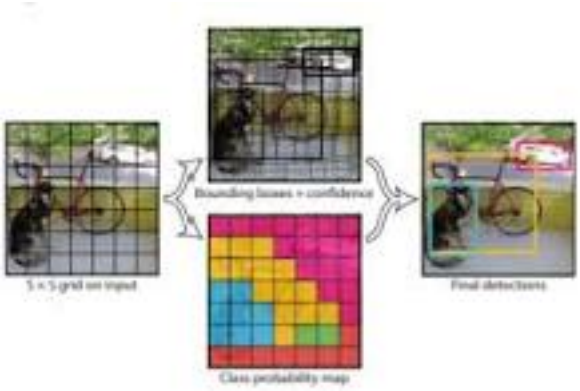
hogy kisebb, egyenlő nagyságú szegmensekre osztjuk a képet, majd ezekre különböző konvolúciós layereket hattatva (és azokat a kimenetre csatolva) klasszifikációt hajtunk végre (illetve bounding box regressziót természetesen). Mindez egy lépésben történik.

A YOLO hálóarchitektúra megjelenéséig (illetve a YOLOv2 előtt) a leggyorsabb és mellette igen precíz hálóarchitektúrának számított, napjainkban a YOLO jobban teljesít.

2.5.3. YOLO

A YOLO (You only look once) napjaink egyik, ha nem a legelterjedtebb hálótípusa ha objektumdetektálásról van szó. A részletesebb elemzés, illetve a későbbiekben a pozíciómeghatározás problémaköre ezen hálóra való illesztése az objektum detektálási dokumentációban szerepel. Röviden működését tekintve hasonló az SSD-hez hiszen szintén egy lépésben történik a detektálás, továbbá szintén kisebb szegmensekre osztja az eredeti képet. A fő különbség a két módszer között, hogy a YOLO esetében a klasszfikáció regresszióra van visszavezetve, hiszen valószínűségi alapon történik a bounding box detektálása (részletesebben lásd obj. detektálás dokumentáció).

A YOLO-ból évente, de az utóbbi időben akár pár havonta újabb verziók érkeznek, melyek egyre jobb és jobb eredményeket adó hálókat eredményeznek.
A projekt során főként ezen hálóarchitektúrára fókuszálunk gyorsasága, precizítása és nagyfokú generalizáló hatása miatt.



6. ábra. YOLO architektúra működése sematikusan

2.6. Konklúzió

Egyelőre mindösszesen irodalomkutatásra és igen minimális méretű adathalmazra alapozva mondhatjuk, de mindenképp valós idejű és jól általánosítható, precíz hálóra van szükségünk a feladat elvégzéséhez, így a különböző YOLO verziók tezstelése, módosítása, kiegészítése egy célravezető útnak mutatkozik.

3. Robot manipulátor

3.1. Végberendezés kiválasztása

Végberendezések tekintetében a lehetőségek feltárását követően arra jutottunk, hogy vákuumos megfogókat fogunk alkalmazni, hiszen ezek megfelelően egyszerűek és mindösszesen három koordinátából determinálható a pozícionálás velük.

3.2. Modellezés

A dokumentum által tárgyalt időszakon belül a használt SCARA robot gépészeti modellezése, jellemző adatainak, melyek a későbbiekben a szimulációknál, digitális ikernél releváns adatként szolgálhatnak, adatlapból és mérésekből történő meghatározása, CAD modell elkészítése, illetőleg a gyártócella hasonló megfontolások alapján történő

modellezése is megtörtént.
(képek...)

7. ábra. CAD modellek

4. Irodalomjegyzék