Haladási napló

1. hét:

Létrehoztam a github repot

Ismerkedtem a ROS-al, csináltam egy ubuntu VS-t, amire telepítettem is elméletileg, bár még nem látom át teljesen

Kicsit utánajártam, hogy milyen frameworköt érdemes használni egyszerűbb VR alkalmazások fejlesztésére, utánaolvastam a Unitynek, illetve Unreal Engine-nek, végül a Unity mellett döntöttem.

Video streamelésére és akár a ROS futtatására is lehet egy RPi-t használnék, az általam talált információk alapján mindkettőre van lehetőség

2. hét

Telepítettem a unity fejlesztői környezetét

VRTK letöltése, ismerkedés a toolkittel

3. hét

Ismerkedés ROSsal, próbaprogramok írása

UR5 robot package telepítése, szimulátor kifagyásának debuggolása (VM reinstallok, nem megfelelő Ubuntu verzió az adott ROS-hoz)

Specifikáció, feladatleírás, ütemterv elkészítése

4. hét

Specifikáció pontosítása

Kerestem, hogy milyen protokollt lenne érdemes használni a vezérlés és visszairányú adatforgalomra

Ezen cikk alapján (<https://www.confluent.io/blog/kafka-fastest-messaging-system/>) a RabbitMQ a leggyorsabb kis adatmennyiség esetén az általa felsoroltak közül, ami feltétel teljesülni fog szerintem jelen esetben. A listában második helyen lévő eszköz a Kafka, aminek késleltetése 5ms, ami jóval nagyobb, mint az 5g-é (ami potenciálisan 1 ms), viszont jóval több adatot tud átvinni.

Mivel azzal már foglalkoztam, illetve a késleltetése elméletileg kisebb, inkább a RabbitMQ-t választottam

Videó továbbításához sima RTMP streamet ki kell próbálni, hogy elég gyors-e, ha nem megfelelő, meg kell nézni az ffmpeg által nyújtott egyéb lehetőségeket

További kutatásom eredményeként megállapítottam, hogy az rtmp nagyon lassú lesz (3-30 s) így érdemesebb lenne a RTC, FTL vagy SRT protokollok egyikét használni inkább.

Igyekeztem az UR5 robot moveit konfigurációját életre kelteni, de még csak a panda robotot sikerült mozgatnom, ami a moveit tutorialban szerepel

Dilemmáztam a python és a cpp között, de végül mivel a döntésem után a cpp-t néhány óra alatt sem sikerült működésre bírnom a moveit-tal, viszont a python szinte azonnal működött és egyébként is bizonytalan voltam, hogy melyik legyen, áttértem a pythonra.

5. hét

Működésre bírtam pythonból az UR5 kart a szimulátorral, meg is mozgattam. Ehhez készítettem egy MoveIt configot az assistanttal.

6. hét

Egyeztettem Ákossal és megismerkedtem a robotkarral, illetve a laborban használt ROS konfigurációval

A megismert config alapján itthon is összeraktam a gazebo szimulátorral is a működést egy új teszt projektben.

Elkezdtem foglalkozni a projekt kommunikációs részével is, kipróbáltam a RabbitMQ-t pythonnal, illetve elkezdtem gondolkodni a kommunikációs csatornákon.

7. hét

A korábban kitalált kommunikációs utak alapján elkezdtem megvalósítani a kommunikációt ROS-on kívülről

sikerült odáig eljutni, hogy kívülről érkező parancs hatására a robot megmozdult a szimulátorban