

TECHBOOST

Oinride Oy

-teknologiademoraportti

Yhteistyössä Oinride Oy



Raportin kirjoittaja

Niko Hutri

Mukana projektissa

Niko Hutri, Varis Markkanen, Immi Renko, Oskari Ala-Kulju, Joonas Viljanen, Topi Volanen, Jaakko Ojaluoma, Jarkko Puhakka, Nilofar Mirzai, Hai To, Musse Musse, Sedat nlsik, Rrustem Dragaj, Endrit Feka ja Hamsa Abdi

Oinride OY

Pelimoottorit mobiilirobotiikan simuloinnissa vaikeassa ympäristössä

Metropolia AMK
Joulukuu 2025
TECHBOOST Loppuraportti

Sisällysluettelo

Lyhenteet ja käsitteet.....	4
Johdanto	5
Projektin taustaa.....	6
Simulaation state of the art	6
Gazebo	7
ABB Robot Studio	7
Siemens NX design Simulation	7
Solid Works.....	7
Blender.....	7
Renkaiden state of the art.....	8
Simulaatio alustan valinta	9
Unreal Engine 5 (UE5).....	9
Unity	9
Robotin .fbx malli Unityyn.....	9
Artikulaatio	10
Pistepilvet.....	14
LiDAR vs Stereokamerat.....	16
Unitree L2.....	17
MeshLab.....	18
Blender	20
Projektissa syntynyt workflow	21
Renkaat.....	22
3d-Tulostus.....	24
Koneistettu valumuotti.....	24
Testaus.....	26
Projektin Yhteenveto	30
Liitteet	31
Viittaukset.....	31
Simulaatio.....	31
Release v5.0 beta · Nikk1bo/Unity-Robot-Simulation.....	31

Lyhenteet ja käsitteet

Rocker-Bogie

NASA:n Rovereilleen kehittämä "jousitus" järjestelmä, jonka tärkein tehtävä on pitää mahdollisimman moni kuumönkijän renkaista aina kosketuksissa ajopinnan kanssa.

LiDAR (Light Detection and Ranging)

Valon nopeuteen perustuva hyvin tarkka etäisyyden mittaus teknologia.

Pistepilvi (Point Cloud)

Kokoelma pisteitä, jotka muodostavat koordinaatiston. tämä voi olla 2- tai 3-ulotteinen malli, joka voi sisältää muutakin dataa kuten värejä, normaaleja tai aikaleimoja.

Mesh

Pistepilvestä tai muusta pintageometriasta johdettu malli, jolla on määriteltynä kulmien lisäksi myös tasot pisteiden väliin sekä niille on tiedossa suunnat (sisä- ja ulkopinta), jolloin syntyy kappaleita, jollaisia voi olla olemassa oikeassa maailmassa eli ns. manifold geometrian omaavia kappaleita.

Avoin lähdekoodi (Open Source)

Ohjelmistokehitys periaate, jossa suuri osa kehittämisestä tapahtuu yhteisön tai yksityisten ihmisten toimesta. Kaikki koodi on julkista ja jokainen asiasta kiinnostunut voi halutessaan lisätä koodiin tai kopioida sen itselleen ja kehittää sitä omaan suuntaansa.

ROS2 (Robot Operating System)

Nimestään huolimatta se on enemmänkin rajapinta, jonka kautta robotteja ja niihin liittyviä sensoreita ja moottoreita voidaan ohjata ja tulkita ns. Ros-nodejen kautta.

CAD (Computer-Aided Desing)

Tietokoneavusteinen mallintaminen tähän on useita ilmaisia ja maksullisia ohjelmia. Ohjelma, jolla on mahdollista nopeasti luoda ja tehdä malleja laitteista ja lisätä niille ominaisuuksia kuten materiaaleja ja toleransseja valmistamista varten.

Prefab

Unityn nimitys valmiiksi ladatulle esineelle, joita on tarkoitus tehdä nopeasti useampi tarpeen vaatiessa esim. tuoli, auto, puu, vihollinen jne...

Digital Twin (Digitaalinen kaksonen)

Simulaation muoto missä laitteelle tai prosessille tehdään digitaalinen kaksonen, jossa tapahtuu 1:1 mallinen kaikki toiminta ja liike, joka tapahtuu oikeassa maailmassa. Puhtaimmassa muodossa ohjaus tapahtuu molempiin suuntiin.

SLAM (Simultaneous localisation and mapping)

Samanaikainen lokalisaatio ja kartoitus, tekniikka jolla pystytään rakentamaan karttaa ja paikantamaan itsensä samanaikaisesti.

CNC (Computer Numeric Control)

Tapa tuottaa tietokoneen avulla koneen liikeratoja äärettömästi toistettavassa muodossa, jos halutaan saavuttaa erittäin korkea tarkkuus tai luotettavasti valmistaa usea identtinen kappale.

Reinforcement Learning (vahvistava oppiminen)

Tekoälyn koulutus tapa, jossa tekoälymalli oppii virheistään sisäisten palkitsemismetodien kautta, kuten ihminen.

TRL (Technology Readiness Level)

NASA:n kehittämä asteikko, jolla voidaan arvioida projektin valmius tuotannon tai käytön osalta, asteikko menee 1–9, jossa 9 on käytössä oleva laite ja 1 on vain idea tarpeesta.

Johdanto

Oinride Oy kehittää kestäviä autonomisia mobiilirobotteja ja älykkäitä ohjelmistojärjestelmiä, jotka suorittavat riskialttiita kaivostoimintatehtäviä, kuten tarkastuksia, kartoitusta ja ympäristön seurantaa. Yhdistämme tekoälyä ja robotiikkaa parantaaksemme turvallisuutta ja toiminnan tehokkuutta ääriolosuhteissa. (Oinride Oy, 2025)

Kaivostoimintaan tarkoitetun mobiilirobotin on kyettävä liikkumaan epätasaisessa maastossa, jossa näkyvyys voi olla heikkoa tietyillä alueilla. Robotti on niin sanottu Rocker-bogie Mars Rover, eli NASA:n kehittämään teknologiaan pohjautuva robusti mobiili laite, joka on valmis kulkemaan, vaikka Marssin pinnalla. (Metropolia TECHBOOST, 2025)

Projekti toteutettiin Metropolian Robo Garagella osana TECHBOOST-hanketta hyödyntäen Scrum metodiikkaa kahden vuoden aikana. Työn tekemiseen osallistui tämän aikana noin 20 ihmistä joista 14 opiskelijointa, Projektissa toteutettiin yksi opinnäytetyö sekä useita työharjoitteluita ja kurssitoita.

Projektin alussa määriteltiin, että haluamme tehdä simulaation, jolla pystymme tekemään Digital Twin malleja. Visio simulaatiosta, jota lähdin toteuttamaan, oli että voimme skannata LiDAR teknologialla Metropolian Myyrmäen kampuksen ympäristön ja sen jälkeen simulaatiossa voimme ajaa robotilla kierroksen kampuksen ympäri.

Kun tämä on saavutettu voimme jatkaa siitä eteenpäin, vaikka generatiivisella AI-koulutuksella, jolloin simulaatioympäristössä voisi kouluttaa robotteja kulkemaan vaikeissa ja tuntemattomissa ympäristöissä. Lähtökohdina halusin robottiagnostisen simulaation, jolla voisi tehdä sekä simulaatioita että tekoälykouluttamista.

Viimeinen ominaisuus, jonka halusimme lisätä, on myös Ympäristön skannaaminen livenä ja dynaamisen ympäristön kartoittaminen ajon aikana. Tämä osoittautui aikataulun puolesta haasteelliselta ja Unityn kyky tehdä pistepilvistä meshejä suoraan skannatusta datasta ei ole kauhean suoraviivaista. Onnistuimme kuitenkin tuomaan ROS:in kautta simuloituja pisteitä Unityyn ja laittamaan niiden paikoille valmiiksi luotuja objekteja, kuten neliöitä sekä kivi-prefabejä.

Projektin taustaa

Keväällä 2023 Oinride Oy osallistui Metropolian Big Flash-hankkeeseen, jolloin Metropolia auttoi Yrityksen silloisen robotti prototyypin kehittämässä. Metropolia sai tämän myötä myös itselleen uuden Mobiilirobotiikka alustan, joka soveltuu ulkokäyttöön, joka täytti puutteen Metropolian Robo Garagen silloisessa kalustossa.



Kuva 1. Autojoen kenttätesti Big Flashin aikana

Big Flashin aikana tehtyä tutkimustyötä hyödyntäen ja Oinriden kanssa tehtyjen palaverien perusteella projektille valittiin suunnaksi simulaatioympäristön kehitys sekä uudenmallisten renkaiden kehittäminen robotille. Metropolialla oli jo intiimi tuntemus Oinriden AutoJoen ominaisuuksista ja sen jatkokehittäminen hyödyntäisi molempia osapuolia.

Simulaatioiden tekeminen oli luonnollinen jatko hankkeelle, jossa oli kehitetty laite, jolla ei vielä ole mitään autonomisia ominaisuuksia. Laitteelle haluttiin kehittää autonominen ajaminen ja tehtävien suorittaminen ja avata mahdollisuus jatkokehittämistä varten teknologioihin, joita liitetään mobiilirobotiikkaan.

Renkaiden kehittäminen tuli asiakkaan toiveena ja tämä oli aikaisemmin Big Flashissa tunnistettu haaste robotille, pienet ja huonosti ympäristöön soveltuvat renkaat aiheuttivat haasteita navigoinnissa robotille tarkoitetuissa hankalissa ympäristöissä kuten kaivoksissa tai louhoksissa.

Simulaation state of the art

Pohjaselvitystä varten keräsimme tiimin omia mielipiteitä ja katsoimme millaisia projekteja, olisi mahdollista tehdä erilaisilla olemassa olevilla simulaatiotyökaluilla. Rajasin Simulaation osalta tutkimuksen sellaisiin ohjelmiin, joita olin itse käyttänyt aikaisemmin ja joihin Metropolialla oli jo olemassa oleva lisenssi.

Gazebo

Ros natiivi paketti, jota pystyy käyttämään myös ilman ROS:ia, mutta koska robottimme toimii ROS:issa Gazebon käyttö simulaatioissa ja visualisoinnissa olisi passeli valinta. Gazebo on myös avoimen lähdekoodin ohjelma, jolloin sen käyttö tutkimustyöhön olisi helppoa ja järkevää. (IsaacSaito, 2025)

Aikaisempi kokemus Gazebosta oli kuitenkin tiimin joukossa pientä, joten sen opetteluun kuluva aika tulisi ottaa huomioon suunnittelussa.

ABB Robot Studio

ABB Robot studio on ABB konsernin kehittämä simulaatio ympäristö, jolla voi tehdä hyviä toimivia malleja esimerkiksi tehdaslinjastoja ja sen liitännäisiä toimintoja kuten prosesseja ja materiaalin kuljettamista paikasta toiseen.

Robot studion avulla tehdyn mallin voi siirtää suoraan mallissa käytettyihin laitteisiin ohjelmisto koodina (ABB, 2025), tämä toimii useimpien kaupallisten laitteiden kanssa ja helpottaa ja nopeuttaa huomattavasti isompien kokonaisuuksien ymmärrystä ja mahdollisten pullonkaulojen löytymistä.

ABB:n simulaatiossa on paljon hyvää, mutta sen suunnittelufilosofia ei oikein mene yhteen meidän tarpeidemme kanssa, sen hyödyt nousevat paremmin esille, jos kehittäisimme simulaatiota tehdaslinjastolle tai jollekin ennalta määrätyle prosessille.

Robot studio perustuu maksulliseen lisenssiin ja se on suljettu ympäristö, jolloin kaikki parametrit eivät välttämättä ole muokattavissa ja kaikki data, jota haluaisimme kerätä ei välttämättä onnistu.

Siemens NX design Simulation

Tarjolla olevista simulaatioympäristöistä minulle kaikkein vierain. Kuten ABB:n simulaatio, se on suunniteltu teollisuuden tarpeita ajatellen. Sen pääpaino on rakenteellisissa simulaatioissa ja lämmön siirtymisessä paikasta toiseen (Siemens, 2025)

NX simulaatioilla on mahdollista tehdä liikkuvia laitteita mm laittamalla liukuhihnoja ja kuljettimia renkaiksi laitteille, mutta se on enemmänkin ympäristön rajoitteiden kiertämistä luovalla tavalla, jotta toivottu lopputulos saadaan aikaiseksi. NX:n iso etu on mahdollisuus lisätä erilaisia olemassa olevia ja itse määritettyjä sensoreita.

NX on maksullisen lisenssin takana oleva tuote ja suljettu ympäristö, joten se kärsii samoista haitoista kuin ABB Robot Studio.

Solid Works

Solid Works on ohjelmista itselle kaikkein tutuin ja tiesin jo valmiiksi, että vaikka sillä pystyykin mallintamaan minimutkaisia mekaanisia laitteita, sen käyttö kokonaisen robotin simulaation tekemiseen ei tulisi onnistumaan

Solid Works Simulaatio ympäristönä olisi hyvin samankaltainen kuin Siemens NX, joten se jakaa paljon hyötyjä ja haittoja aiemmin mainitun kanssa.

Halusin silti lisätä sen tähän State of the Art basemarkkiin koska sitä tullaan käyttämään tässä projektissa yksittäisten laitteiden, kuten simulaatiossa käytettävän robotin AutoJoen CAD mallintamiseen.

Blender

Blender on ilmainen yhteisökehitetty ohjelma GNU General Public Licensen alla, jolla pystyy tekemään useita asioita kuten muokkaamaan 3D- Malleja, animoimaan, mallintamaan ja renderöimään monimutkaisiakin asioita. (Blender Foundation, 2025)

Blenderillä voi tehdä monimutkaisiakin mallinnuksia, mutta siitä puuttuu kaikenlainen valmis fysiikan ja materiaalien mallinnus simulaatiotamme varten ja sellaisen tekeminen alusta alkaen itse olisi turhan raskasta meidän resursseillamme ja aikataulullamme.

Blenderiä käytettiin kuitenkin projektissa valmiiden meshien siistimiseen, jolloin saatiin poistettua automaattisen mesh generaation sekä käsin tehdyn meshien yhdistelyn tuomia virheitä.

Renkaiden state of the art

Renkaiden kanssa tilanne eroaa hiukan sillä tähän ei oikein ole muuta tapaa kuin katsoa millaisia muita rengasratkaisuita on kilpailijoilla ja millaisia renkaita haluamme itse alkaa kehittämään.

Renkaiden tutkimisessa paljastui, ettei kauhean monessa vastaavassa laitteessa ole kovinkaan rouheakuvioista rengasta ja useimmilla laitteilla on vain 4 rengasta tai telaketjut, toisin kuin AutoJoen 6 renkainen rakenne.

Tämä ohjasi valintaamme sen osalta, että tiesimme että pienen valmistuserän valmistustavat olisivat todennäköisesti parhaita ja mahdollisuuksien mukaan olisi parasta käyttää hyväksi jo tuotannossa olevia pienen koneiden ja työkalujen renkaita kuten kottikärryt, mönkijät yms.

Renkaiden keskiönä on ns. HoverBoardin renkaiden keskiöt, nämä ovat erilaisten mobiilirobotiharrastelijoiden suosimia napamoottoreita, joten netistä löytyi useita esimerkkejä näihin moottoreihin sovitetuista renkaista.



Kuva 2. Hoverboard lauta

Simulaatio alustan valinta

Projektin alussa tehtiin lyhyt katsaus silloisiin simulaatio alustoihin. Tarjolla olevat simulaatioalustat tarjosivat hyviä työkaluja, mutta yksikään niistä ei tarjonnut kokonaisvaltaisesti kaikkea mitä halusin lisätä simulaatioon. Halu tehdä myös tutkimustyötä Digital Twin-teknologioita ja mahdollisuutta tehdä uniikkeja ja yksityiskohtaisia simulaatioita ympäristöistä riippumatta tarkoittaisi, että simulaation pitäisi taipua moneen erilaiseen. Projektin alkusuunnittelussa todettiin, että mikäli simulaatio tehdään, valittavaksi jää joko olemassa olevia alustoja kuten ABB Robot studio, Gazebo tai Siemens NX. Vastapainona näille olisi tehdä oma simulaatio alusta alkaen.

Tutkituissa simulaatioympäristöissä oli omat etunsa, jokainen toi jonkin ominaisuuden, mutta suurimmassa osassa valmiita simulaattoreita pääpaino oli lisensoinnissa ja valmiissa laitekirjastoissa, jossa suurin osa markkinoilla olevista laitteista on tarjolla. Koska robottimme on itse rakennettu ja sen toiminta on hyvinkin tuttua kinematiikan ja sensoriensa osalta, koimme että simulaation kehittäminen itse olisi paras tapa pitää langat käsissä. Tällä tavalla myös mahdolliset tulevat lisensointi kustannukset pysyisivät kohtuullisina.

Tähän näin mahdollisuuden pelimoottoreissa. Peleissä on jo vuosia haettu koko ajan realistisempaa fysiikkaa, oikeanlaista ja tuntuista painovoimaa, valojen heijastusta pinnoilla ja voimien vaikutusta ympäristöihin. Pelit nykyään ovat jo lähes toimivia simulaatioita, joten jäljelle jäi valinta jonkin avoimen pelikehitysalustan valinta. Meidän tapauksessamme valinta tehtiin Unityn ja Unreal Engine 5:n välillä.

Unreal Engine 5 (UE5)

Unreal Engine 5 on Epic games:in tuottama ilmainen pelien kehitys alusta, joka yrittää erottaa itsensä kilpailijoista tarjoamalla erittäin realistisia grafiikoita ja valaistus työkaluja. Vaikkakin hyviä asioita kun suunnittelee simulaatiota, graafinen esitys ei ollut korkealla listalla tärkeistä asioista. UE5 tarjoaa myös työkalut mm elokuvien, simulaatioiden ja animaatioiden tekemiseen. (Epic Games, 2025)

UE5 isoin huono puoli on sen tuntemattomuus projektin tekijöille, jolloin iso osa projektin tekijöistä joutuisi käyttämään paljon resursseja uuden oppimiseen jo vaikean ja uuden projekti toteutuksen lisäksi.

Unity

Unity on Unreal Enginen isoin kilpailija pelikehitysalustojen saralla Unity yrittää erottaa itsensä lisäämällä kuvioon XR-toiminnallisuuden ja Open Source-kehitysympäristön, joka on tuottanut mm. suoran rajapinnan ROS2:een. Open Source ominaisuus on erittäin haluttavaa, kun tehdään tutkimustyötä, jolloin datan käsittely ja sen kerääminen on helpompaa. (Unity Technologies, 2025)

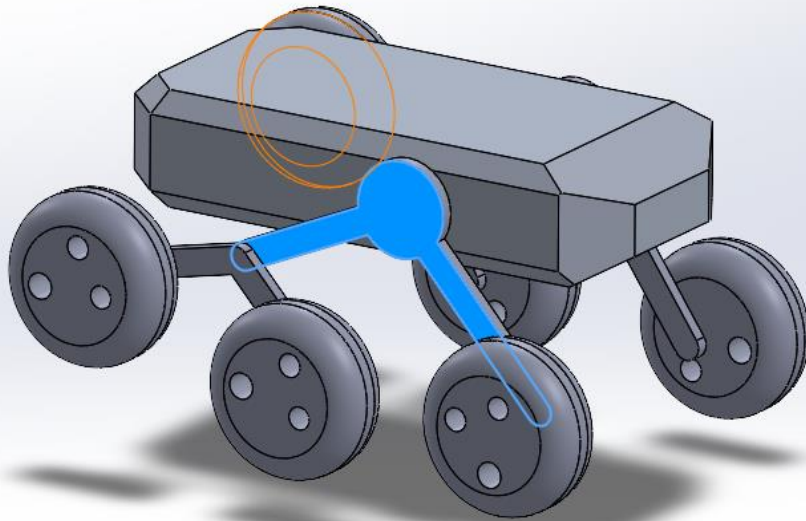
Koska olin jo aikaisemmin tehnyt muutamia omia projekteja Unityssä ja Metropolialta löytyy pelikehityskoulutusohjelma ja useita ihmisiä, jotka hallitsevat ja opettavat kyseisen ohjelman käyttöä katselimme Unreal Engineä lähinnä sillä silmällä, että tekeekö se jotain ehdottomasti paremmin kuin Unity.

jo kehitteillä olevan ja hyvin dokumentoidun ROS2 rajapinnan olemassaolo varmisti sen, että Unity valitaan projektiin simulaatioympäristöksi, sillä robottimme toimii ROS2:n avulla.

Robotin .fbx malli Unityyn

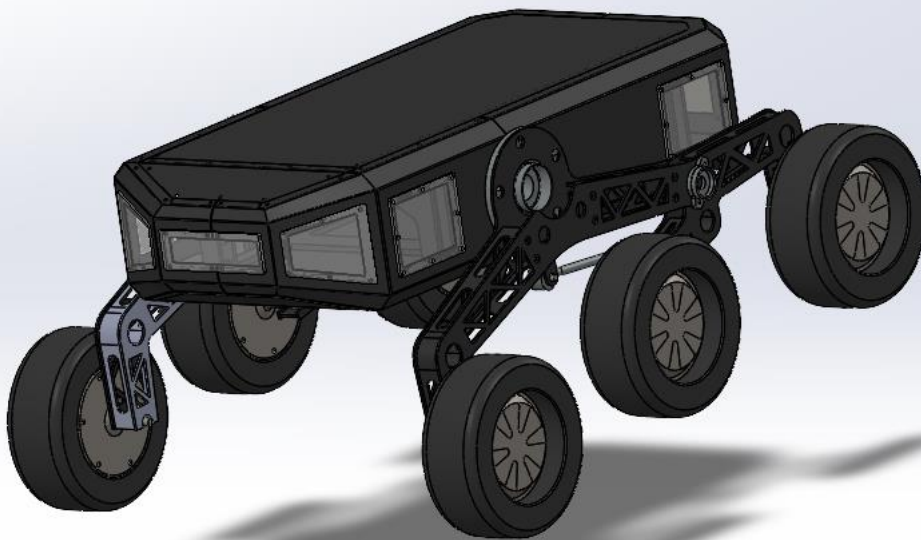
Jotta robottimme saadaan Unityyn sen muoto pitää mallintaa tarpeeksi tarkasti ja sen jälkeen nivelet ja pyörivät osat pitää asemoida malliin oikeille paikoilleen ja niiden rajoitteet ja suhteet toisiinsa pitää merkitä Unitylle, jotta malli simulaatiossa toimii samalla tavalla kuin sen kaksonen oikeassa maailmassa.

AutoJoesta tehtiin ensin hyvin karkea malli Solid Worksilla, joka saatiin muokattua oikeaan tiedostomuotoon käyttäen Blenderin Import/Export toimintoa, jolloin tehty malli saatiin Unityyn.



Kuva 3. Pelkistetty malli AutoJoe robotista, mitat Metropolian robotista

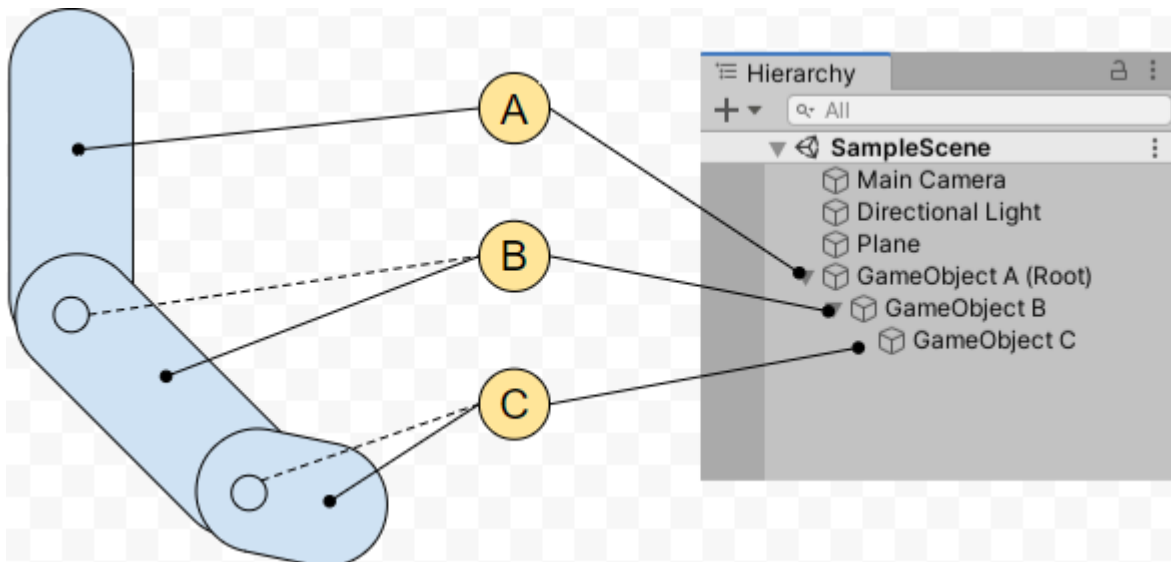
Myöhempää käyttöä varten valmistettiin myös hyvin tarkka malli Metropolian Metrover-robotista, joka on jatkokehitetty Big Flashin aikana tehdyn työn pohjalta.



Kuva 4. Yksityiskohtainen malli Metropolian Robotista

Artikulaatio

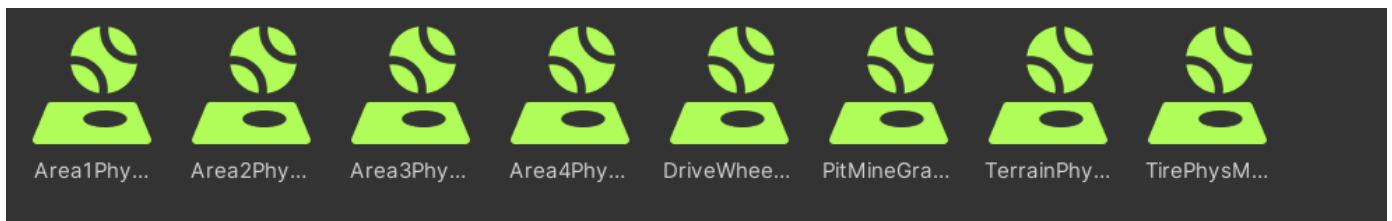
Unityllä on laaja [dokumentaatio](#), jonka hyödyntäminen oli projektin onnistumiselle erittäin tärkeää. Artikulaatiolla varmistetaan, että simuloitu robotti kykenee kaikkiin samoihin liikkeisiin, kun sen oikeassa maailmassa oleva kaksonen.



Kuva 5. Unityn tutoriaali nivel artikulaatiosta

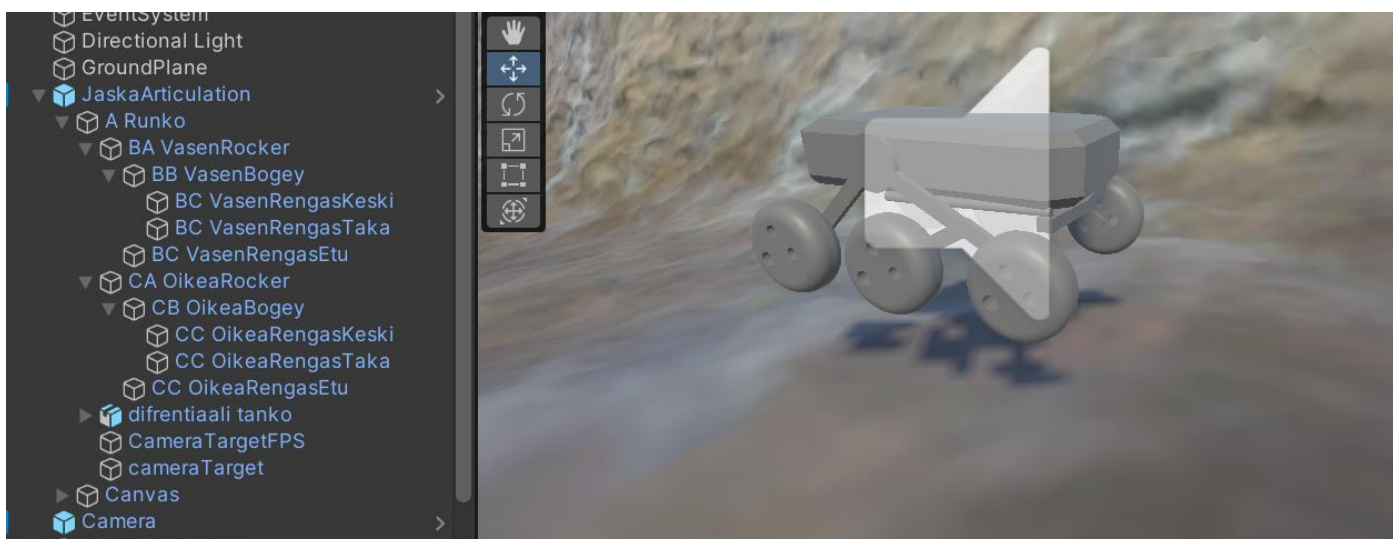
Artikulaatiolla saadaan myös rajattua hyvin se, ettei laite kykene asioihin, jotka eivät ole oikeassa maailmassa mahdollisia. Esim. Metallilevyjen siirtyminen toistensa läpi, nivelien liikutus kulmissa, joihin ne eivät taivu jne...

Unity tukee materiaaliominaisuuksia, joten teimme useita materiaaleja eri tarkoituksiin. mm. Kallio, hiekka, rengaskumi jne.



Kuva 6. Simulaation materiaali-tiedostot

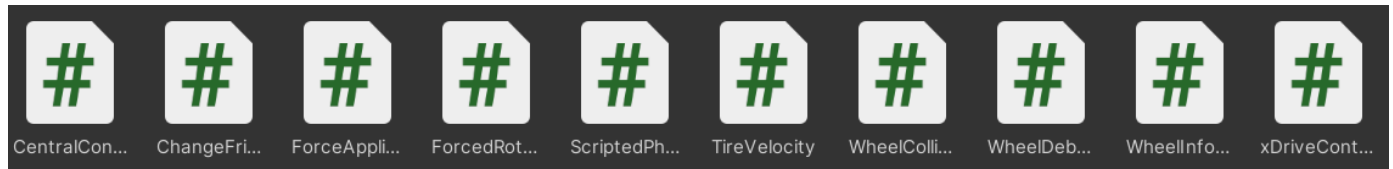
AutoJoe on ihmiselle yksinkertainen ymmärtää konseptina, mutta sen Rocker-Bogie rengas järjestelmä tekee renkaiden simuloinnista haasteellista. Autojoella on kuusi rengasta ja kummassakin jalassa on kaksi toisistaan kiinni olevaa itsenäisesti pyörivää niveltä, jotka hakevat maaston pintaan mahdollisimman hyvän kosketuksen.



Kuva 7. Robotin artikulaatorakenne kokonaisuudessaan

Kuten kuvasta näkee, artikulaatio menee jopa 4 kansiota syvälle, jotta saadaan nivelet toimimaan kuten ne oikeassa elämässä laskeutuvat, niihin on myös asetettu rajoitetut mekaaniset rajat, jotta nivelet eivät heilu enempää kuin laitteen rakenne antaa myöten.

Unityyn teimme myös useita scriptejä, joilla saamme parannettua simulaation vastaavuutta oikean maailman kanssa. Simulaatio ottaa myös huomioon rakenteelliset lujuudet, eli jos laitetta kurittaa liian kovaa, se hajoaa simulaatiossa. Halusin, että mikäli laitteen ajaa liian korkealta kielekkeeltä tai niin kovaa estettä päin, ettei oikean maailman laite siitä selviäisi, niin ei pidä simulaationkaan jatkaa siitä matkaa.



Kuva 8. Simulaation scriptit

Unity on silti pohjimmiltaan pelimoottori ja palvelee niitä tarkoituksia, joten sen käytössä simulaattorina on otettava muutamia asioita huomioon. Robotille pitää määrittää massat, massakeskipisteet, materiaalit ja muut ominaisuudet oikean maailman perusteella ja se vaatii jonkin verran kokeilua ja varmistamista, että kaikki sujuu kuten oikeassa maailmassa.

Oikean robotin tiedot saadaan tuotua Unityyn käyttäen Universal Robot Description Format (URDF) dataa. Näin oikean robotin kokema asento, nopeus, kiihtyvyys yms. kerätty data saadaan tuotua suoraan simulaatiolle. Simulaatiomme käyttää Unityn valmiita ROSTCP-kirjastoja ja kerää mm. LiDAR dataa robotilta ja pystyy tuomaan sen suoraan Unityyn. (Unity Technologies, 2020)



Ros Laser Data (Mono Script)

Assembly Information

Filename Assembly-CSharp.dll

```
using System.Collections;
using RosMessageTypes.Geometry;
using RosMessageTypes.Nav;
using RosMessageTypes.Sensor;
using Unity.Robotics.ROSTCPConnector;
using Unity.Robotics.ROSTCPConnector.ROSGeometry;
using UnityEditor;
using UnityEngine;
using UnityEngine.VFX;
```

```
public class RosLaserData : MonoBehaviour
{
    public ROSConnection ros;
    [SerializeField] bool instantiationEnabled = true;
    private LaserScanMsg previousScanMsg = null;
    public GameObject objectToInstantiate;
    private GameObject points;
    [SerializeField] GameObject vfxPointCloud;

    void Start()
    {
        ros = ROSConnection.GetOrCreateInstance();
        ros.Subscribe<LaserScanMsg>("scan", DrawPoints);

        // Empty GameObject for points
        points = new GameObject("Points");
    }

    void DrawPoints(LaserScanMsg scanMsg)
    {
        previousScanMsg ??= scanMsg;
```

Asset Labels

Kuva 9. Scripti, jolla LiDAR-data saadaan luettua Unityyn

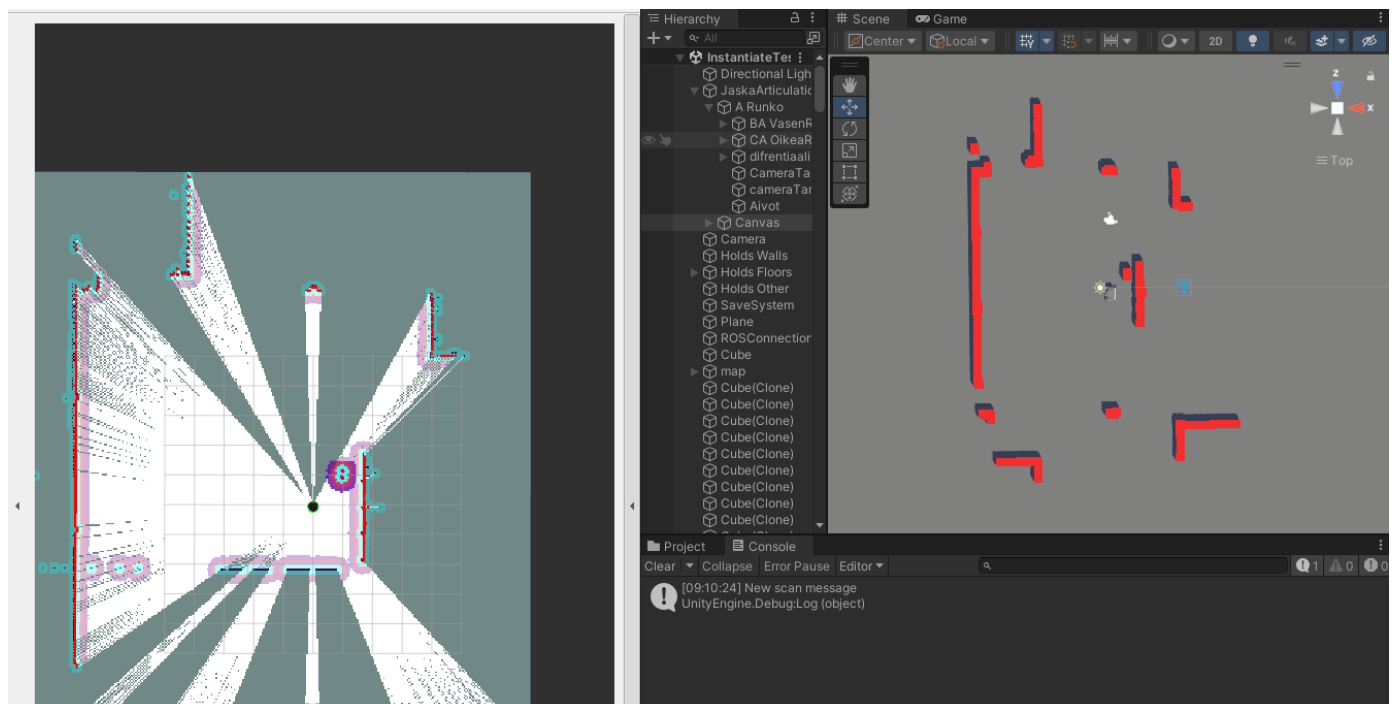
Pistepilvet

Simulaatioympäristön rakentamista varten päädyimme skannaamaan koko Metropolian Myyrmäen kampuksen. Tämän avulla pystyisimme valmiissa simulaatioympäristössä ajamaan AutoJoella kampuksen ympäri ja tarvittaessa simuloimaan sen Digital Twin-mallina.

Halusimme saada aikaiseksi jotain SLAM:in kaltaista. Käyttämällä Unityn ROSTCP kirjastoja tämä data saadaan kerättyä. Kokeilumme keinotekoisen datan kanssa tuottivat lupaavia tuloksia ja saimme tehtyä hyviä demoja käyttäen Clearpath Turtlebot4 robottia, joka käyttää 2D-LiDAR:ia itsensä ja ympäristön paikantamiseen

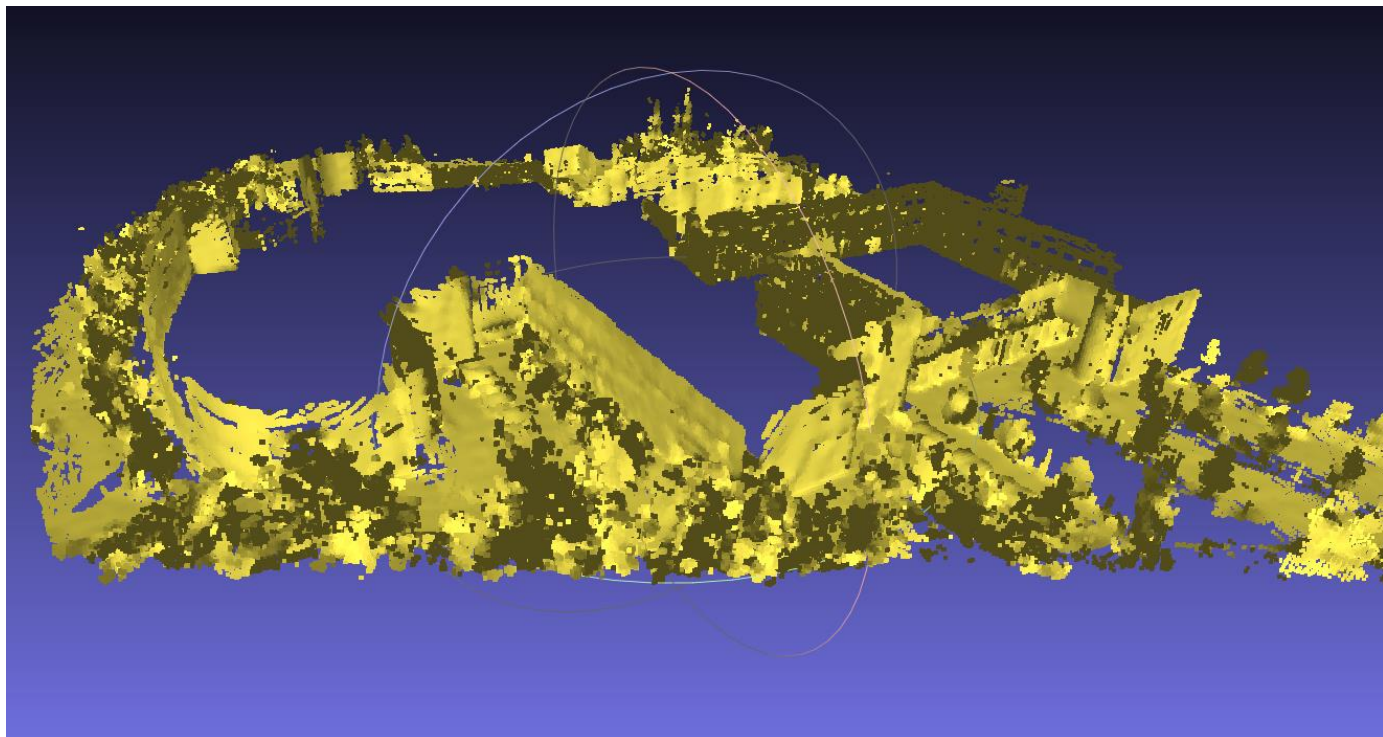


Kuva 10. Turtlebot 4



Kuva 11. Simuloitua lidar dataa ROS:issa ja Unityssä

Alkuun kuitenkin meille riittää, että pistepilvi kerätään, sen pohjalta rakennetaan mesh, joka saadaan vietyä Unityyn, jossa simulaation kaikki osat tulevat yhteen. Tämän pohjalta voimme alkaa kehittämään SLAM:in kaltaista systeemiä, jossa robotti rakentaisi ympäristöä samalla kun se liikkuu siinä. Autonomisen liikkumisen kouluttamista varten staattisen ympäristön mallinnus on tarpeeksi tässä vaiheessa.



Kuva 12. Pistepilvi, jossa pisteille on määritelty suunnat, eli normaalit

Lopullinen pistepilvi Myyrmäen kampukselta kerättiin MiR100-mobiilirobotin avulla. Vaikka MiR100 on omia sensoreita, niitä ei käytetty datan keräämiseen vaan Mir100 toimi ainoastaan kauko-ohjattavana alustana, sillä kun dataa kerättiin AutoJoe ei ollut liikuntakunnossa. Mir100 päälle kiinnitettiin Stereolabs Zedbox tietokone, joka toimii Metropolian AutoJoen aivoina. Sen sisällä ROS2 ympäristössä käyttäen Unitree L2 LiDAR:ia kartoitettiin yllä olevan kuvan pistepilvi.

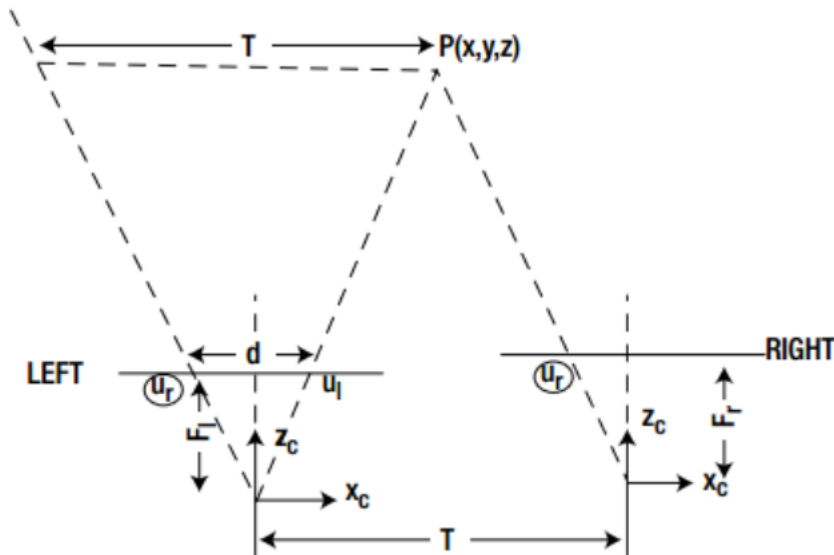


Kuva 13. MiR100- mobiilirobotti

LiDAR vs Stereokamerat

Pistepilvien keräämiseen on tällä hetkellä 2 yleistä tapaa, joko stereokameroilla, jolloin käyttäen parallaksi laskentaa ja trigonometriaa voidaan päätellä esineiden etäisyys kamerasta melko tarkastikin. Tarkkuus perustuu muutamaen ennalta tiedettyyn etäisyyteen, jolloin trigonometria ratkaisee meille loput.

CHAPTER 10 ■ 3D GEOMETRY AND STEREO VISION



Kuva 14. Geometria josta kahaden kameran kuvista voi laskea etäisyyden molempiin

Kahdesta eri kuvasta voidaan mitata kuvapisteen ero, jota kuvassa merkitään d kirjaimella, se saadaan seuraavasti $d = u_l - u_r$, jossa u_l ja u_r ovat pisteen P koordinaatit vasemmassa ja oikeassa osakuvassa (kuvassa oikea kameran kuva on siirretty vasemman kameran kuvan vasemmalle puolelle havainnollistamaan kuvaa). Kun huomioidaan trigonometrian säännöt yhtenevistä kolmiosta, niin voidaan

laskea pisteen P Z-koordinaatti seuraavasti. (Vallinen, 2017)

$$\frac{d}{T} = \frac{f}{Z} \text{ eli } Z = T * \frac{f}{d}$$

Stereo kameroilla tehtävä ympäristön skannaus yleistyy koko ajan ja sitä varten tehdään jatkuvasti uusia algoritmeja ja metodeja, jolla tarkkuus ja nopeus paranee. Tämän rinnalle on nousemassa ns. Gaussian Splat-metodi, jossa rakennetaan 3D-malleja kuvista tai videosta, jolloin tarve kalliille stereokameroille tai LiDAR:eille poistuu, mutta näin tehdyt mallit eivät sisällä tarkempia syvyystietoja, joten niillä ei voi vielä tehdä meshejä luotettavasti, mutta soveltuvat hyvin esim. 3d-ympäristöjen luomiseen videoihin tai tarkoituksiin joissa pinnan tarkka geometria ei ole välttämätöntä tietää.

Meille parempi tapa on käyttää LiDAR tai Ultraääni teknologiaa ja luottaa äänen tai valon lähes vakionomaiseen nopeuteen ilmakehässä. Ultraääni kärsii äänen hajoamisesta, kun se matkustaa pidempiä matkoja ja se tekee mittauksista epätarkkoja. LiDAR:illa voidaan mitata isompia etäisyyksiä hyvinkin tarkasti käyttäen infrapuna valoa.

Unitree L2

Tätä Projektia varten mietimme monia erilaisia LiDAR:eita ja lopulta päätös oli Velodyne Puck:in ja Unitree L2:n välillä. Molemmilla on hyvä käytettävyys ROS ympäristön kanssa, L2 vei lopulta voiton halvemmalla hinnalla ja avoimella lähdekoodilla ja huomattavasti laajemmalla skannaus kentällä.



Kuva 15. Unitree L2 LiDAR

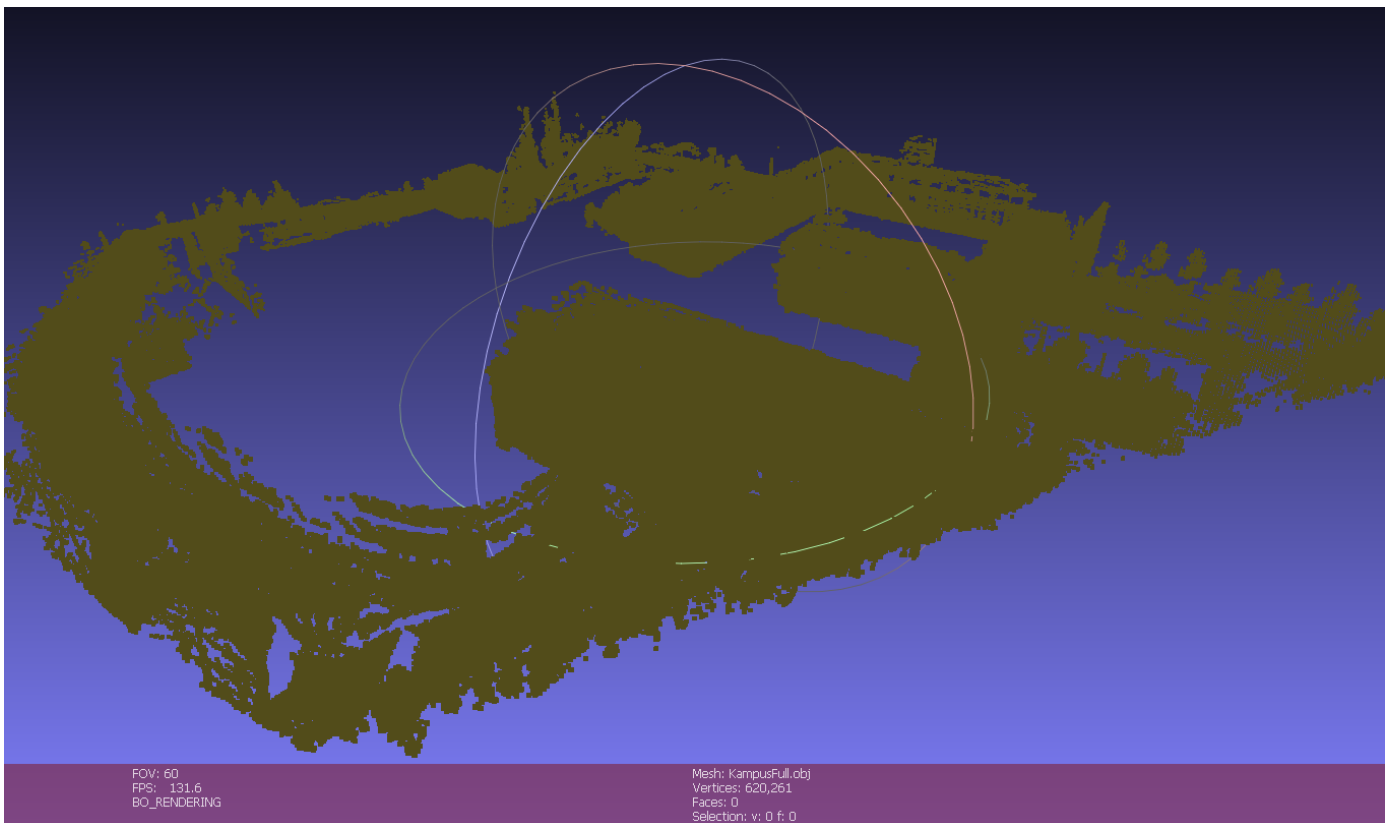


Kuva 16. Velodyne Puck LiDAR

MeshLab

MeshLab on avoimen lähdekoodin projekti, jolla pistepilvistä voi tehdä meshejä helposti. MeshLabissa voi myös käsitellä meshejä, korjata niiden geometriaa ja poistaa skannauksista syntyneitä virheitä ja artefakteja. Se tukee myös suuria meshejä, joten se toimii oikein hyvin meidän tarkoituksiimme. (Visual Computing Lab, 2025)

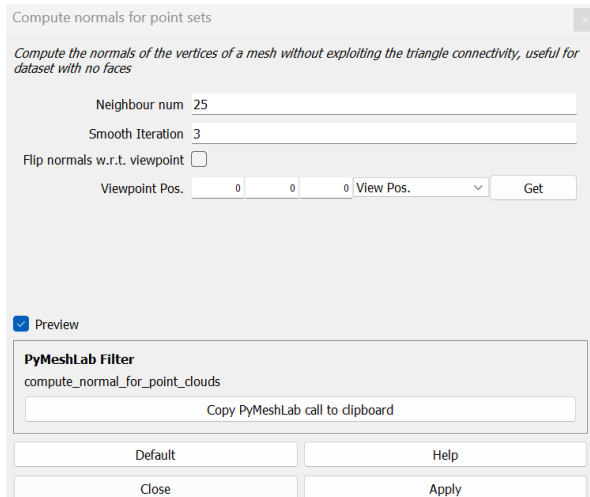
MeshLabin avulla kerätystä pistepilvestä saatiin määritettyä Normaalit eli suunnat pisteille. Tämä vaihe on tärkeä Meshien automatisoinnin luomiselle, sillä algoritmi ei muuten tiedä mihin suuntaan pinnat osoittavat ja tämä aiheuttaa suuria topologia ongelmia myöhemmin.



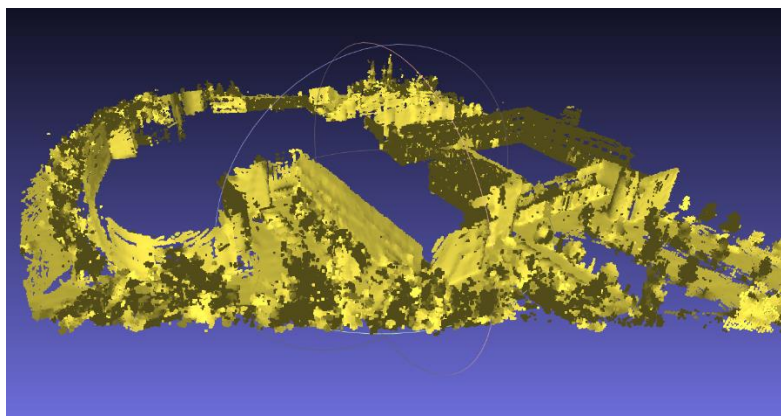
Kuva 17. Pistepilvi Metropolian Myyrmäen kampukselta

Alussa data on vain pisteitä avaruudessa, mutta algoritmisesti on mahdollista tämän mallin jokaiselle yli 620 tuhannelle pisteelle määrittää oletettu suunta sen ympärillä olevien pisteiden perusteella. Tällöin Mesh topologian määrittäminen on mahdollista, kun ymmärretään missä pisteen mittaaja on ollut suhteessa mitattavaan pisteeseen.

Meidän mallillemme määritin normaalit käyttäen jokaiselle pisteelle 25 lähintä naapuria viitteenä ja jokainen piste silloittaa 3 kertaa oman ympäristönsä, jolloin mielestäni tuli hyväksyttävä koheesio kerätyn datan ja oikean kampuksen välille.



Kuva 18. MeshLab parametrit normaalien automaattiseen laskemiseen

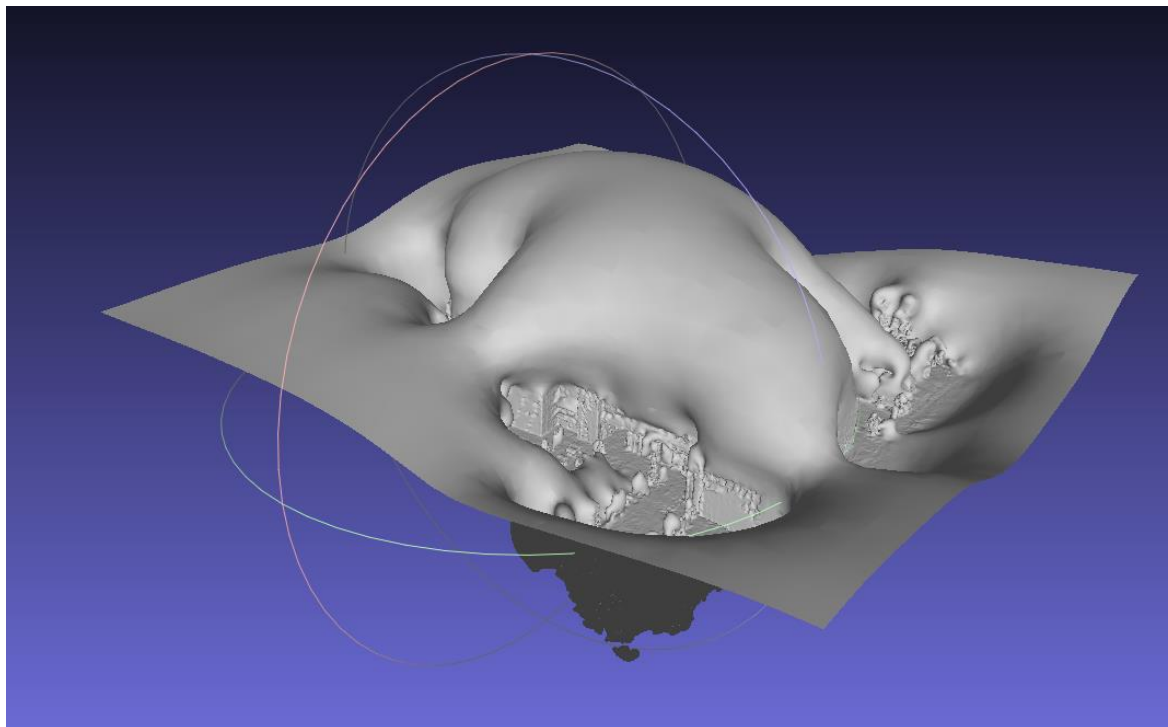


Kuva 19. Pistepilvi, jossa normaalit laskettu

Tämän jälkeen mallin on valmis seuraavaan vaiheeseen, jossa normaalien välille voidaan luoda pintoja ja reunoja, tähän MeshLab tarjoaa 3 helppoa työkalua, joista voi valita mieleisensä.

Käytin aluksi Ball pivoting metodia, jossa jokaiselle pisteelle määritetään naapurit laajentamalla sitä ja sen perusteella tehdään pinnat. Tämän metodi luo siistejä ja tiukkoja meshejä, sen huono puoli on, että meshien luominen on hyvin laskennallisesti työläs prosessi ja MeshLabin kirjasto tähän metodiin tukee vain yhden CPU ytimen käyttöä, jolloin yhden parametrin kokeilu saattoi kestää yli 30 min.

siirryin myöhemmin käyttämään Screened Poisson metodia, jota voisi kuvailla ”kelmun” vetämiseksi pisteiden päälle ja sen sulattamiseksi pisteiden ympärille, tämä metodi osoittautui hyvin nopeaksi, sillä se tuki rajatonta CPU:n käyttöä, jolloin pystyin tekemään uusia parametrikokeiluja jopa minuutin välein.



Kuva 20. Mesh Screened Poisson metodin jälkeen

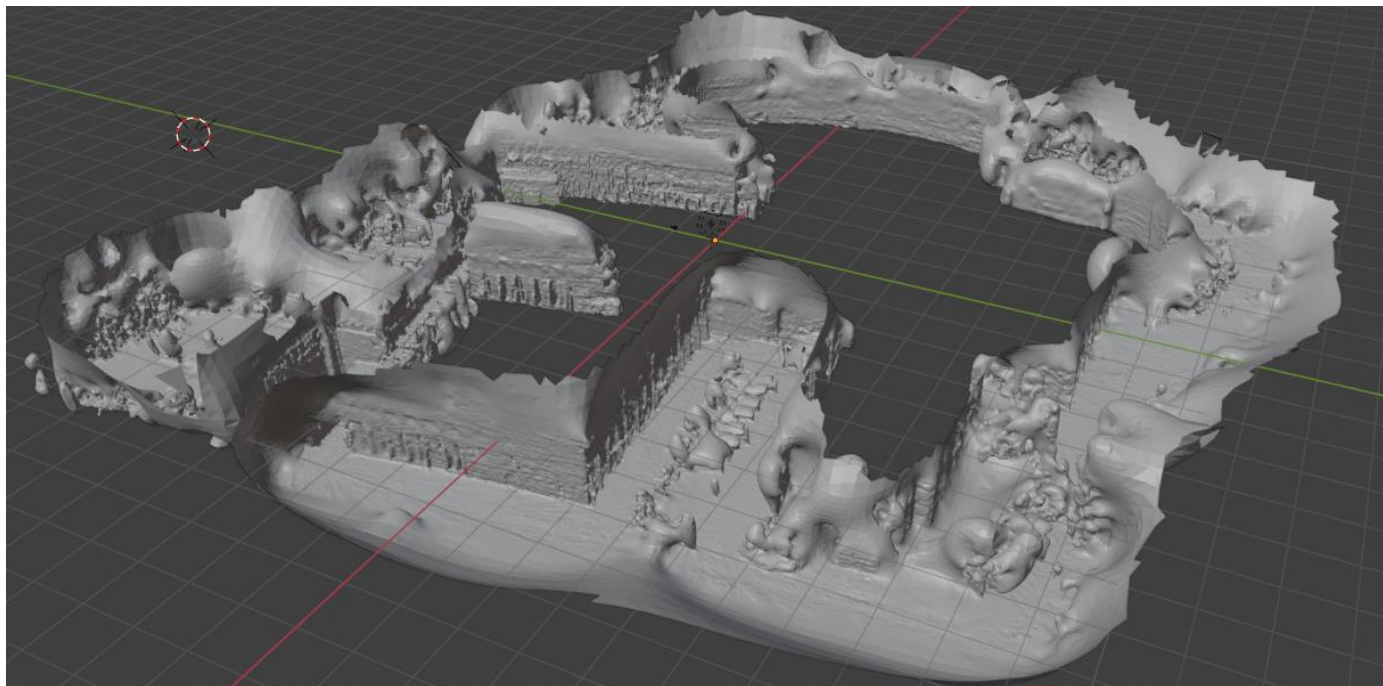
Screened Poisson metodi pyrkii tekemään "vesitiiviitä" malleja, tämä on tärkeää mm. 3D-tulostamisessa, jossa kappaleen täytyy olla kokonainen ja omata manifold geometria. Tämän haittapuolena on "kelmun" poistaminen lopullisesta meshistä

Mesh olisi tässä vaiheessa jo ajettavassa kunnossa, mikäli hyväksyy kelmun olemassaolon. Halusin tehdä vielä käsin siistimistä mallille ja poistaa olemattomia kattoja, joita meshin luonti toi

Blender

Nyt kun on olemassa Mesh, joka vastaa lähes 1:1 Myyrmäen kampusta, halusin siistiä ympäristöä niin, että projektin alussa suunnittelemani kierros kampuksen ympäri simuloitussa ympäristössä olisi mahdollinen.

Käyttäen Blenderin mallinnus työkaluja poistin käsin isoimmat Screened Poisson metodin jättämät artefaktit, jotta jäljelle jäisi mahdollisimman paljon vain meidän mittaamaamme dataa ympäristöstä, luonnollisesti, koska pistepilvemme oli verrattain pieni skannatun alueen kokoon nähden ja siellä oli muutamia katve alueita. Meillä oli vain 620 000 pistettä korttelin kokoisella alueella, jolloin paljon dataa jäi algoritmien pääteltäväksi.



Kuva 21. Mesh sen jälkeen kun osa Screened Poisson artefakteja on poistettu

Nyt kartta on valmis siirrettäväksi Unityyn ja koska käytin tähän vaiheeseen Blenderiä, voin tässä vaiheessa export toiminnolla muuttaa kartan .fbx muotoon, jolloin tiedän, että Unity pystyy ottamaan sen vastaan.

Tiesimme jo valmiiksi, että tarvittava tarkkuus olisi n.10 cm luokkaa, sillä se on korkein este, jonka yli AutoJoe voi meidän nykyisillä renkaillamme kiivetä. Tähän tarkkuuteen ei meidän mittauksellamme päästy.

Projektissa syntynyt workflow

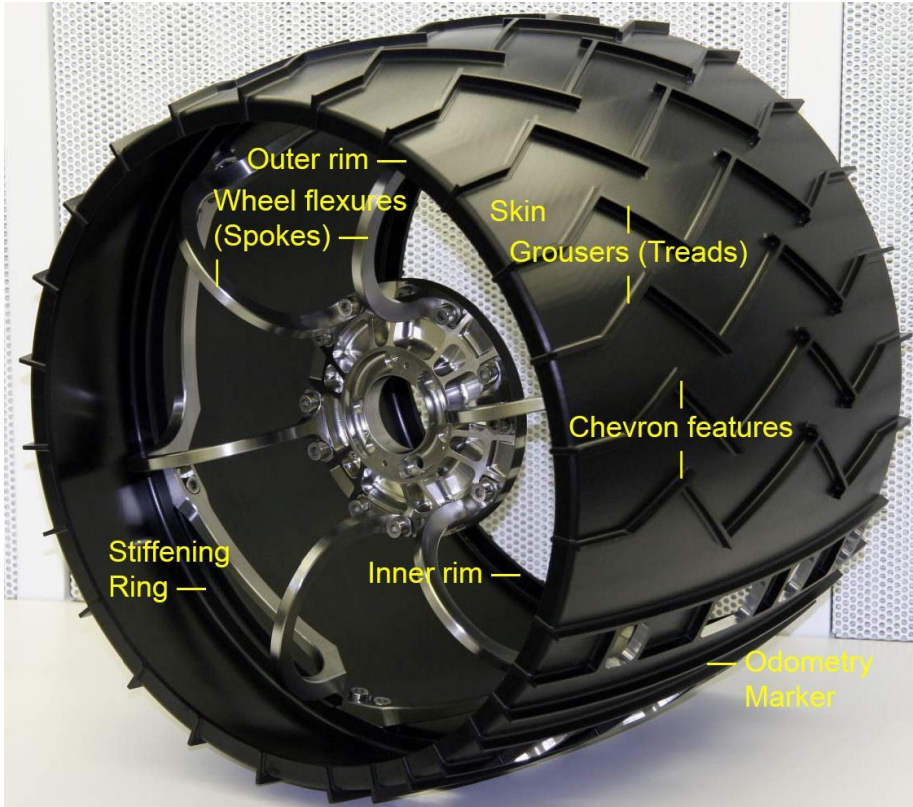
Projektin workflow tällä hetkellä ei ole niin automaattinen, kun olisin toivonut, siinä on monta välivaihetta ja kohtia, jossa on paljon ihmisen käsin tehtävää siloittelua, mutta tämän pohjalta voimme alkaa kehittämään parempaa metodia.

Aluksi pistepilvi kerätään haluamallaan metodilla, sen jälkeen pistepilvi käsitellään MeshLabilla meshiksi ja halutessaan sitä voi parannella käsin haluamallaan 3D-ohjelmalla. Näin tehty Mesh voidaan siirtää Unityyn ja käyttää siellä niihin tarkoituksiin, kuin haluamme.

Renkaat

Renkaita varten valmistimme useita 3D-malleja sekä tulostimme useita prototyyppejä. Päädyimme lopulta valmistamaan renkaat kaatovalamalla ne polyuretaanista.

Renkaissa halusimme korostaa futuristista ilmettä ja tähän tarkoitukseen päätimme käyttää hunajakennopaista sivuprofiilia ja rengaskuviota haettiin mm maastopyörien renkaista ja muista off-road ajoneuvoista ja luonnollisesti myös Mars Curiosity Roverista, josta koko robotin rakenne on peräisin.



Kuva 22. Mars Curiosity Roverin Alumiini-renkaat

Curiosity Roverin renkaat on valmistettu koneistamalla alumiinista ja niissä on otettu huomioon huoltovaikeudet, kun Rover kulkee Marssin pinnalla.

Ostetut kumista tehdyt ilmarenkaan olivat myös hetken tutkinnan alla, mutta robotin jalkojen rakenne aiheutti ongelmia, kun renkailla ei ollut yhtään tilaa laajentua sivuttaissuunnassa.



Kuva 23. Ilma-renkaan sovitus Metropolian roveriin

Harkitsimme myös hetken ruiskuvalumuotista tehtyjä renkaita, mutta ne osoittautuivat vaikeasti valmistettaviksi sekä verrattain erittäin kalliiksi ottaen huomioon meidän hyvin pienen valmistuserämme. Muottien valmistus olisi silti järkevää koska halusimme, että renkaita voisi valmistaa meidän tarpeisiimme jatkossa niin monta kuin haluamme.



Kuva 24. CAD-malli renkaan viimeisestä versiosta

3d-Tulostus

Alkuun käytimme 3D-tulostusta ensimmäisenä mahdollisena tapana valmistaa renkaan, nopeasti tajusimme, että kampuksella olevista tulostimista ei saa helposti tulostettua sopivan kokoista rengasta.

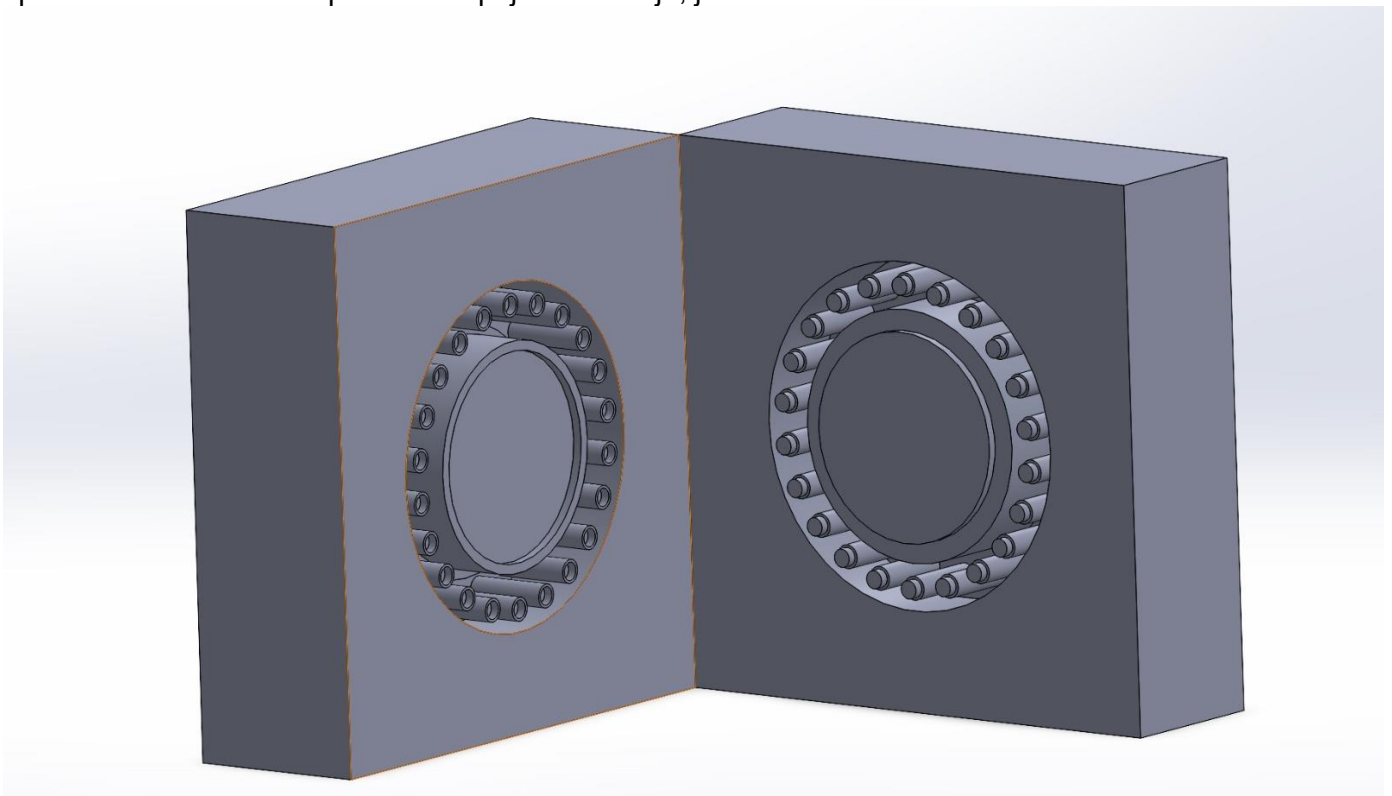


Kuva 25. 50% kokoisia testitulosteita, oikean mallin löytämiseksi.

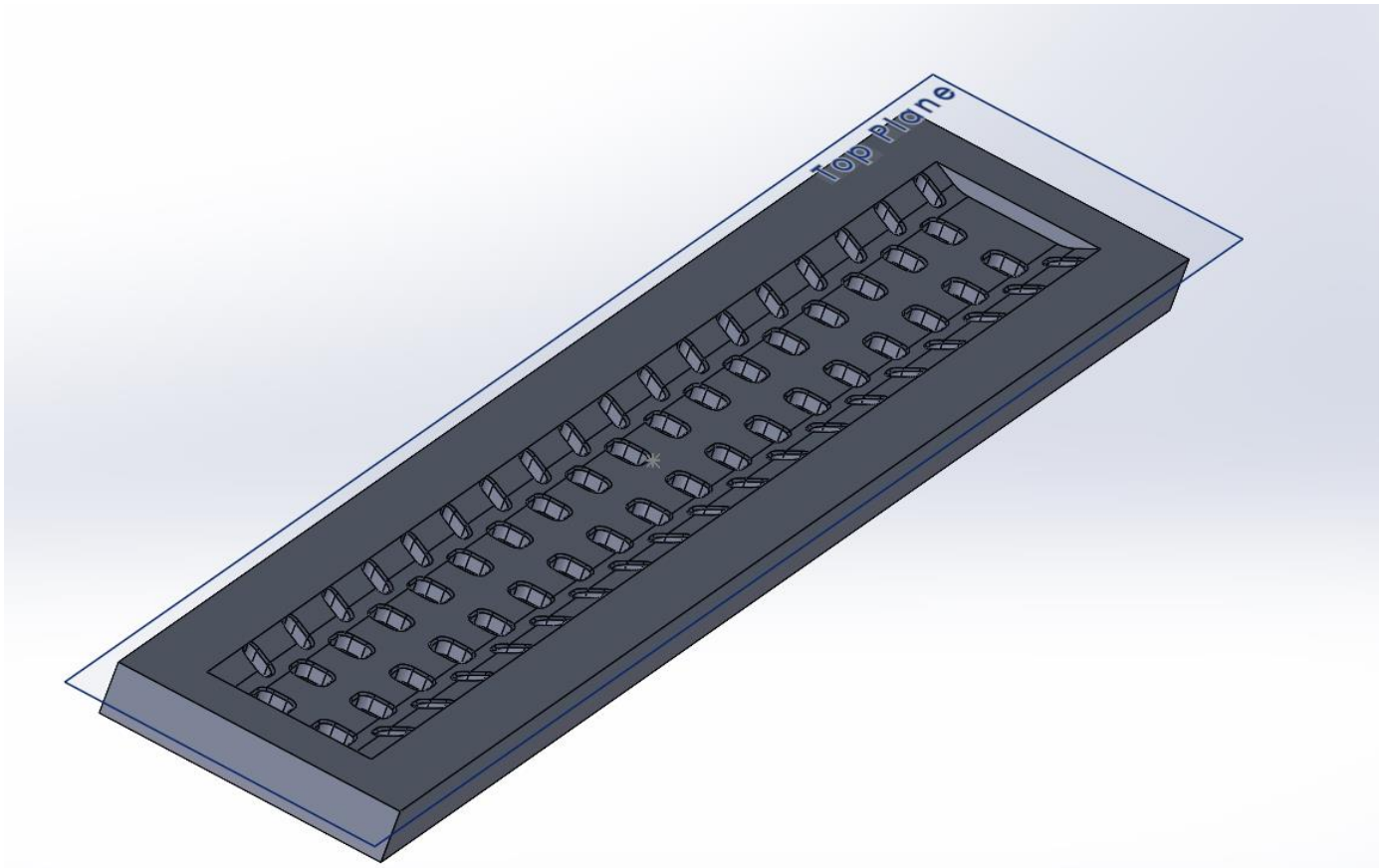
Koneistettu valumuotti

Kun olimme päätyneet valumuottiin valmistustapana, tajusimme että jotta muotti pysyy helposti valmistettavana ja helposti kastattavana, tulee rengas valmistaa 2 tai useammasta osasta, jotta sen monimutkainen muoto saadaan irtoamaan muotista.

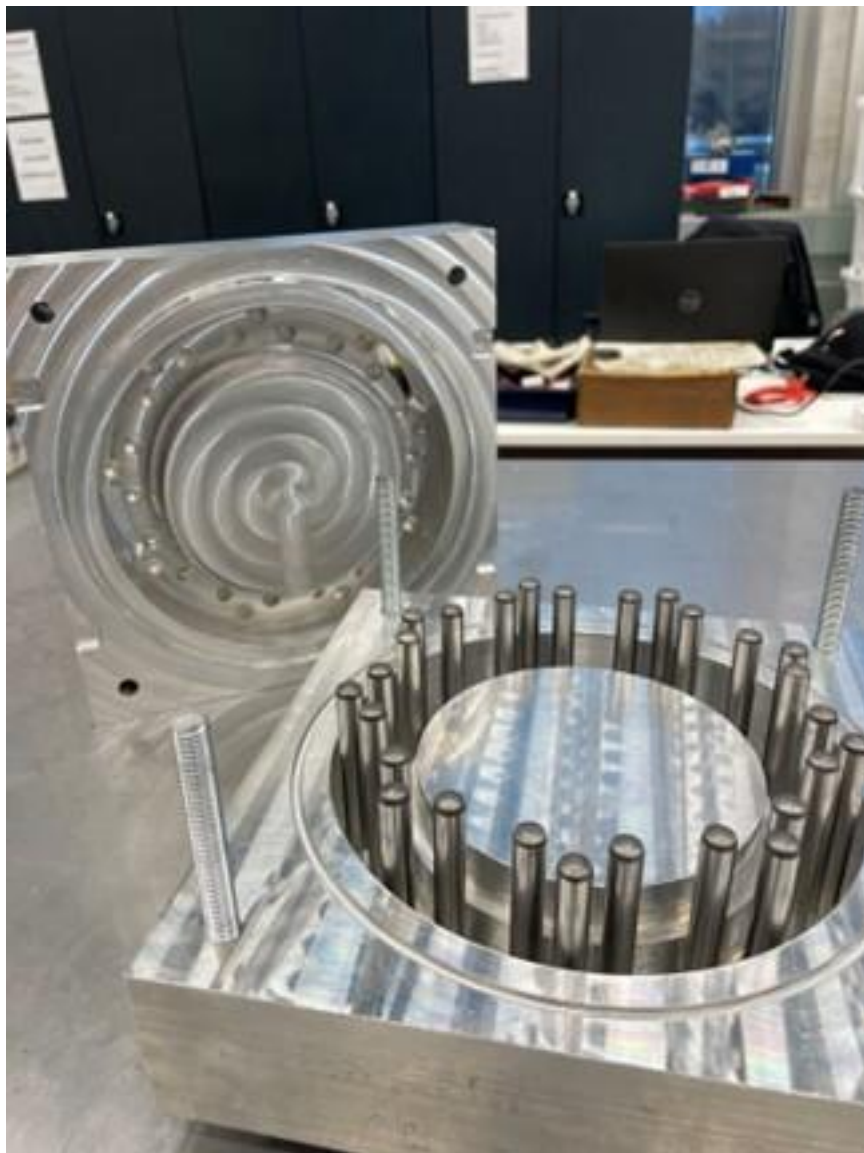
Lopullinen muotti päätettiin valmistaa alumiinista Metropolian 5-akselisella HAAS CNC-koneella, muotin spekseihin haettiin Metropolian konepajalta vinkkejä, jotta sen valmistus onnistuisi.



Kuva 26. Ensimmäinen prototyyppi valurenkaan muotista



Kuva 27. Valurenkaan ulkokuvion muotti



Kuva 28. Valurenkaan sisäosan alumiini-muotti

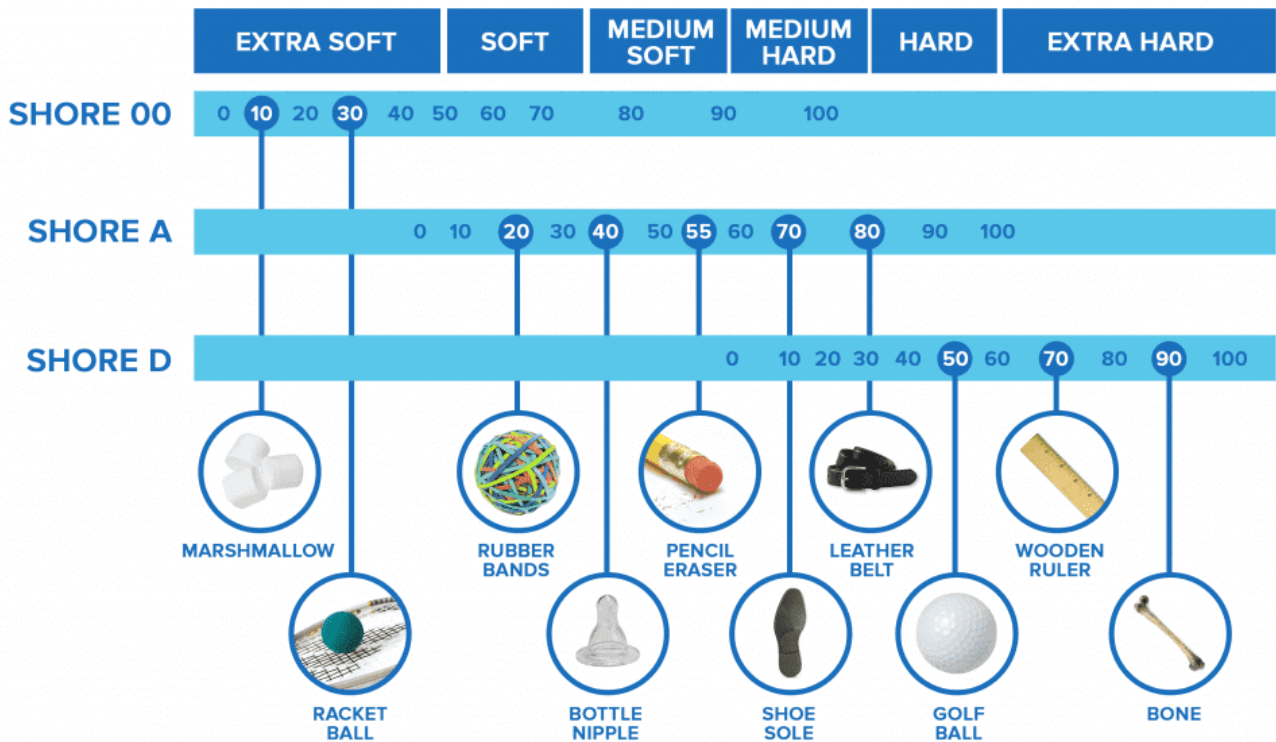
Testaus

Renkaiden testaus jäi tämän projektin puitteissa hyvin vähälle, sillä suurimman osan projektin ajalla meidän AutoJoe robottimme oli poissa käytöstä, koska sille kehitettiin uutta runkoa ja uusia jalkoja, joissa olisi kääntyvät etu- ja takarenkaat.

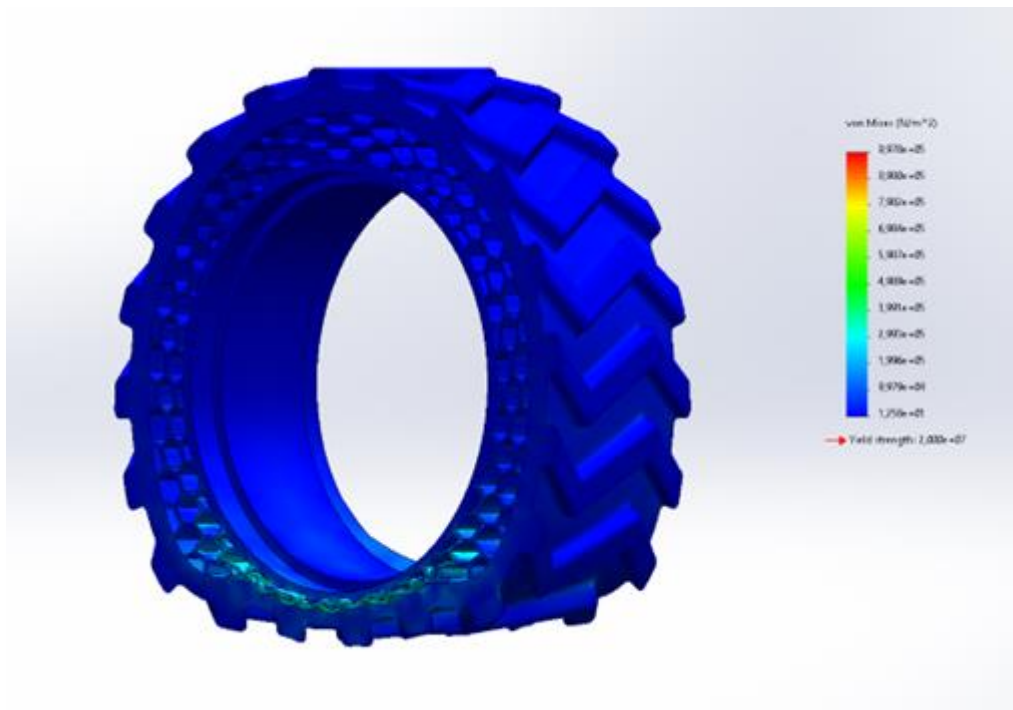
Testauksen sijasta teimme paljon lujuus- ja kestävyyslaskentoja käyttäen Solid Worksin simulointityökaluja. Simuloinnissa pääsimme hyvään käsitykseen siitä mitä ominaisuuksia hyödyntämällä voimme valmistaa paremmat renkaat kuin meillä on tällä hetkellä.

Stressianalyysi paljasti 35 kg massalla, että renkaat painuvat kasaan noin 0.5mm laitteen oman painon alla. Näissä materiaalina oli Shore kovuusasteikon 85A TPU materiaali, josta ensimmäiset prototyypit oli tarkoitus valmistaa.

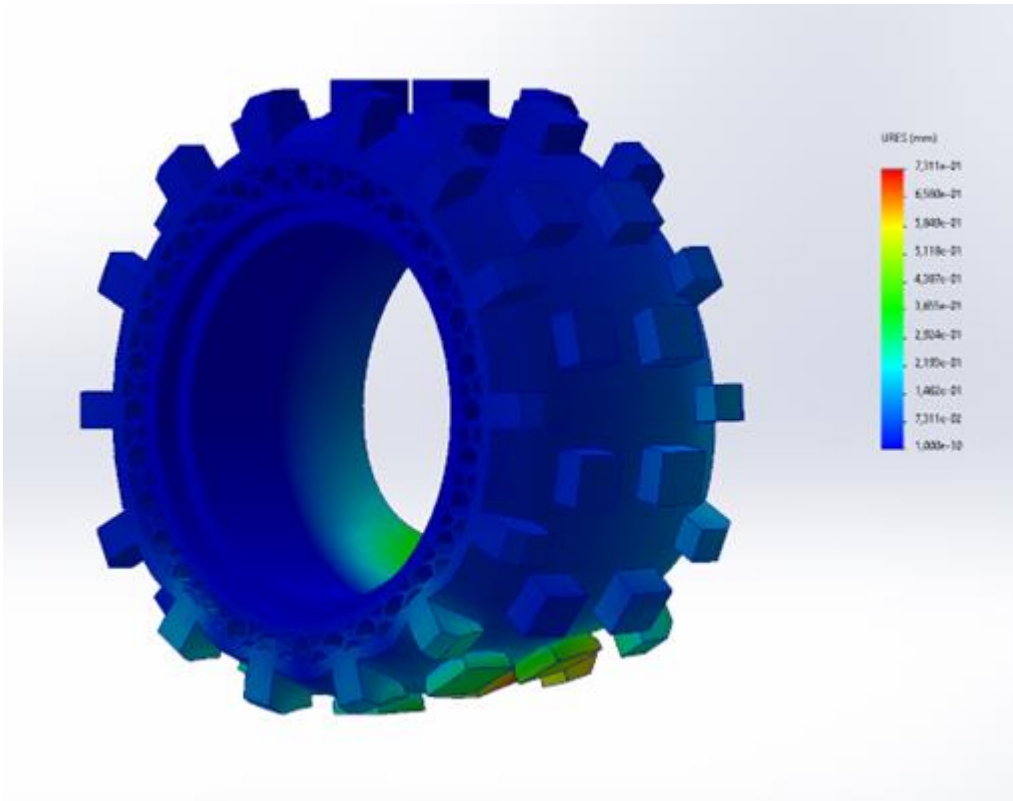
Shore Hardness Scale



Kuva 29. Shore kovuus-taulukko



Kuva 30. Hunajakkenno renkaan stressianalyysi



Kuva 31. Nappulakuviosen hunajakkeno renkaan stressi analyysi

Yhden testirenkaan onnistuimme tulostamaan, mutta sen testausta ei valitettavasti keretty tekemään, johtuen Metropolian robotin keskeneräisestä rungon kehittämisestä.



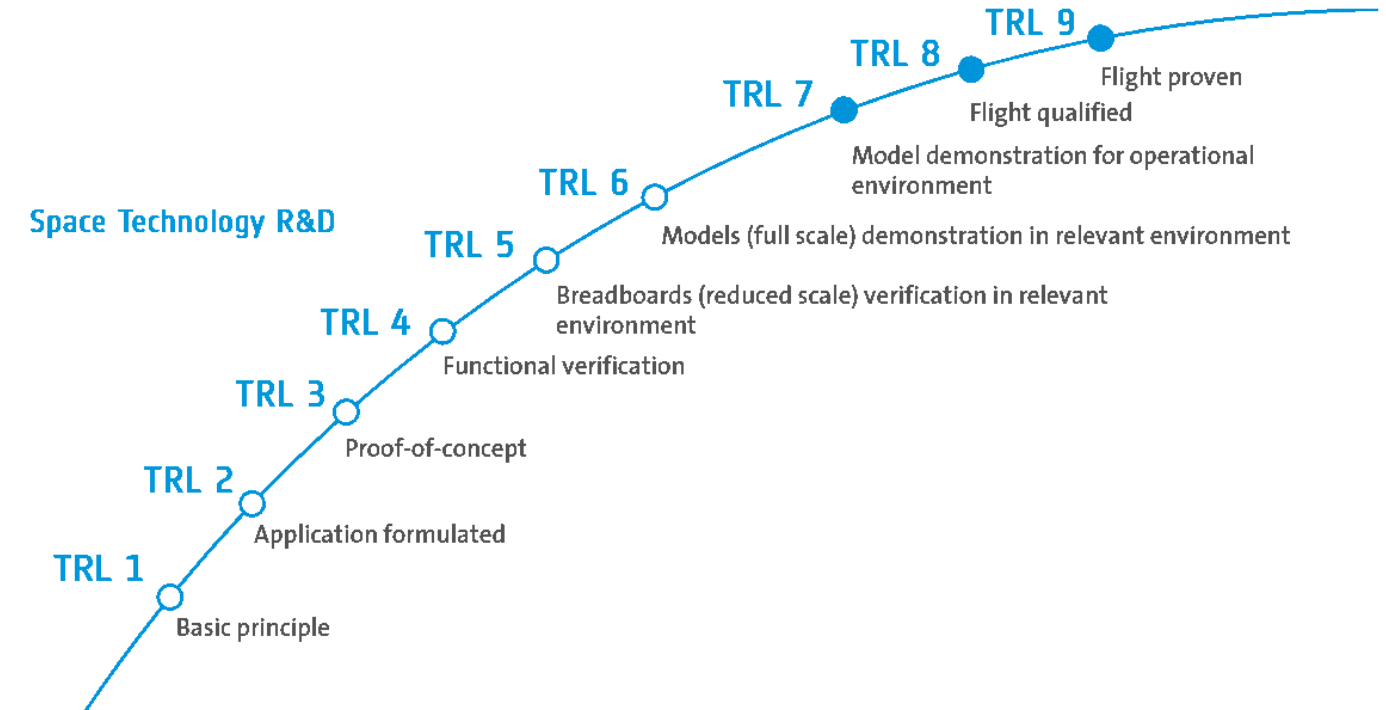
Kuva 32. 3D-tulostetun renkaan kasaanpainumisen testausta pöydällä

Kuvan rengas on tulostettu Bambu labs Carbon X1 tulostimella materiaalista Filaflex Fluor 82A.
https://recreus.com/gb/filaments/9-684-filaflex-82a.html#/1-colour-black/2-diameter-175_mm/3-weight-500_gr

Lisään liitteeksi tiedoston missä on tämän renkaan tulostusasetukset viimeisimmällä rengasversiolla. Renkaasta valmistettiin myös aiemmin näytetty valumuotti, jolla valmistettiin 1 kappale PU-valu renkaita sisäosa Shore-kovuudella A80 ja ulkokehä kovuusluokalla A40. Tämän valmistuksen tilasimme omalla muotillamme Ravelast. Heillä on vuosien kokemus valutuotteista ja halusimme, että ensimmäinen prototyyppi onnistuu varmasti. (Ravelast Polymers, 2025)

Projektin Yhteenveto

Projektissa hankkeen aikana tutkittiin monia asioita robottisimulaation ympäriltä ja saatiin aikaiseksi oikein kehityskelpoinen simulaation alku, joka on enne kaikkea robottiagnostinen ja avoimeen lähdekoodiin perustuva ohjelma.



Kuva 33. Nasan kehittämä TRL-malli

Simulaation osalta projekti eteni TRL tasolta 1 aina tasolle 6 asti.

Renkaiden osa projektista eteni TRL tasolta 3 tasolle 7.

Simulaatiossa on otettu huomioon tulevaisuuden kehityssuuntia mm. Ros integraatio, jolla kykenisi tulevaisuudessa tekemään sensori-integraatiota. Unity ympäristö mahdollistaa mm. reinforcement learning tyyppisen tekoäly pohjaisen autonomia ohjauksen.

Uskon että tämä projekti tulee pohjustamaan hyvin sekä Metropolian että Oinride Oy:n autonomisen ohjauksen, sekä tekoälypohjaisen autonomisen koulutuksen kehitystä.

Seuraava mahdollinen kehityssuunta olisi joko autonomisen liikkumisen koulutus tunnetussa simuloidussa ympäristössä tai datansiirron kehittäminen, jotta saadaan automatisoitua data-putki sille että robotin keräämä sensori-data saadaan älykkäästi visualisoitua Unity simulaatiossa.

Liitteet

Filaflex 82A printtiasetukset.3mf

Viittaukset

- ABB. (3. 12 2025). *ROBOTSTUDIO suite*. Noudettu osoitteesta <https://www.abb.com/global/en/areas/robotics/products/software/robotstudio-suite>
- Blender Foundation. (8. 12 2025). *About Blender*. Noudettu osoitteesta <https://www.blender.org/about/>
- Epic Games. (3. 12 2025). Noudettu osoitteesta Unreal Engine: <https://www.unrealengine.com/en-US>
- IsaacSaito, W. g.-1.-2. (3. 12 2025). *Gazebo*. Noudettu osoitteesta <https://wiki.ros.org/gazebo>
- Metropolia TECHBOOST. (1. 12 2025). *Pelimoottorit mobiilirobotiikan simuloinnissa vaikeassa ympäristössä – Case Oinride Oy*. Noudettu osoitteesta TECHBOOST: <https://www.techboost.fi/projekti/oinride/>
- Oinride Oy. (1. 12 2025). Noudettu osoitteesta oinride.com: <https://oinride.com/fi>
- Ravelast Polymers. (24. 10 2025). *Ravelast*. Noudettu osoitteesta <https://www.ravelast.com/palvelut/kulutussuojaus/valu-ja-muottituotteet.html>
- Siemens. (3. 12 2025). *NX Design Simulation*. Noudettu osoitteesta https://www.plm.automation.siemens.com/en_us/Images/7936_tcm1023-4361.pdf
- Unity Technologies. (19. 11 2020). *Robotics simulation in Unity is as easy as 1, 2, 3*. Noudettu osoitteesta <https://unity.com/blog/engine-platform/robotics-simulation-is-easy-as-1-2-3>
- Unity Technologies. (3. 12 2025). Noudettu osoitteesta Unity: <https://unity.com/>
- Vallinen, J. (2017). *ETÄISYYDEN MITTAAMINEN STEREOKAMERAN JA OPENCVN AVULLA*. Noudettu osoitteesta https://www.theseus.fi/bitstream/handle/10024/123698/Vallinen_Juha.pdf
- Visual Computing Lab. (9. 9 2025). *MeshLab*. Noudettu osoitteesta <https://www.meshlab.net/>

Simulaatio

Simulaatio sellaisenaan kun sen projektin lopussa oli löytyä täältä

[Release v5.0 beta · Nikk1bo/Unity-Robot-Simulation](#)