M2O2P

Multi Modal Online & Offline Programming Solution

Multimodaalinen online- ja offline-ohjelmointi ratkaisu

Käyttöohje

Samu Rautiainen

4.4.2022

Contents

Esittely

Tämän käyttöohjeen tarkoituksena on helpottaa M2O2P komponentin asennusta ja käyttöä, sekä tehdä siitä saavutettavampi myös käyttäjille, joilla on vähemmän tietoa samankaltaisista järjestelmistä. Ohjeita seuraamalla käyttäjä saa tietoa asennuksesta, volyymeista, joilla konfiguroidaan Docker-kontit, kuinka web-käyttöliittymää käytetään, sekä tapoja järjestelmän testaamiseksi.

Komponentin vaatimukset:

- Windows PC
- Kaikki tarvittavat M2O2P Docker kuvat ladattuna
- CaptoGlove sensorihanska
- Capto Suite asennettuna (ei pakollinen, mutta suositeltu sensorihanskan toimivuuden puolesta)

Testaamista varten (ei pakollinen):

- Postman -sovellus ladattuna

Tämän käyttöohjeen seuraaminen on helpompaa, jos CaptoGlove on jo etukäteen otettu käyttöön CaptoGloven oman käyttöohjeen mukaan. Käyttöohjeessa selitetään lyhyesti, kuinka CaptoGlove sensorihanska toimii, mutta tietoa itse hanskasta ei tässä käyttöohjeessa esitellä.

LATAUKSET JA ASENNUS

Kloonaa tämä Github repositorio ennen tulevien ohjeiden seuraamista (sama repositorio, jossa tämä käyttöohje sijaitsee). Kaikki tarvittavat tiedostot Docker kuvien lisäksi löytyy tästä kyseisestä repositoriosta.

```
git clone https://github.com/SHOP4CF/M2O2P.git
```

Kun CaptoGlove sensorohanskaa otetaan ensimmäistä kertaa käyttöön, sen firmware täytyy päivittää. CaptoGlove käyttöohjeesta löytyy tästä lisää tietoa. Uusin firmware voidaan ladata CaptoGloven nettisivuilta, ja asentaa Capto Suite -ohjelmalla Windows tietokoneella käyttäen USB-kaapelia.

Firmwaren asentemisen jälkeen CaptoGlove sensorihanska täytyy parittaa Windows tietokoneen kanssa Bluetoothin välityksellä. Kun CaptoGlove on yhdistetty, tarkista sensorihanskan nimi Bluetooth laitteista tietokoneelta. Sensorihanskan nimi voi olla esimerkiksi *CaptoGlove3170.* Tämä tunnus täytyy kirjoittaa CaptoGlove SDK:n konfiguraatiotiedostoon (configuration.txt), joka sijaitsee ”captoglovesdk” kansiossa. Varmista että hanskan numero on oikealla rivillä, riippuen onko kyseessä vasemman vai oikean käden hanska. Testataksesi tämän konfiguraation toimivuutta, suorita CaptoGloveSDK.exe tiedosto ”captoglovesdk” kansiossa. Jos konfiguraatio onnistui, kymmenen sekunnin lähtölaskennan jälkeen komentokehotteessa näkyy viimeisenä viesti ”can’t connect to server”.

On suositeltavaa käyttää Capto Suite -ohjelmaa hanskan kalibroimiseen jos kyseessä on uusi hanska, tai jos sensorihanskan firmware on juuri päivitetty. Kun CapotGlove on kytkettynä tietokoneeseen Bluetoothilla, se voidaan yhdistää CaptoSuiteen kyseisen hanskan (vasen tai oikea) valintaikkunasta. Kun yhteys on vihreä (yhteyttä kuvaava pallo), “Setup” osion alla on ”Fingers” painike, jota painamalla päästään valintaikkunaan, jossa voidaan kalibroida sensorihanskan taittuvat ja painesensorit. Tällä helpotetaan komponentin omaa kalibrointia.

KONFIGURAATIO

Seuraavia vaiheita varten komponentin Docker kuvat tulee olla ladattuna RAMP Docker rekisteristä. Tämän voi myös tehdä automaattisesti käyttämällä docker-compose.yml tiedostoa. Docker kuvat voidaan käynnistää samanaikaisesti käyttäen docker-compose.yml tiedostoa.

Docker-compose -tiedosto

Ennen kuin docker compose tiedostoa voidaan käyttää, siihen tulee tehdä tarvittavat muutokset. Docker kuvien konfigurointia varten käytetään paikallisia volyymeja (jotka kloonattiin aikaisemmin GitHub repositoriosta). Volyymit liitetään kuvaan käyttämällä docker-compose tiedostossa sijaitsevien palvelujen alla olevan volume attribuutin alle. Esimerkkinä integration-service palvelun alla oleva volyymi voisi olla kirjoitettu seuraavasti:

- /c/Users/User/Documents/ros2_ros1_command.yaml:/home/is-workspace/src/bridge/src/ros2_ros1_command.yaml

Polku volyymin sijaintiin ennen ":" merkkiä tulee korjata vastaamaan polkua, josta aikaisemmin kloonatut tiedostot löytyvät. Tämä tulee tehdä kaikkien volyymien kohdalla.

M2O2P komponentti tukee natiivisti ROS2 (Foxy) rajapintaa normaalisti käytettävän FIWARE rajapinnan lisäksi, ja käyttäen integration-service palvelua sama tieto on myös saatavilla ROS1 (Noetic) aiheissa. Jos kyseistä ROS1 rajapintaa ei tarvita, ei ole tarvittavaa myöskään käyttää integration-service palvelua, ja se voidaan poistaa kokonaan docker-compose tiedostosta.

ALustavaa testausta varten on helpompaa jättää ainakin mongo-db ja orion kuvat docker-compose tiedostoon koskematta. Jos Orion Context Broker ja MongoDB käynnistetään toisaalla, tulee konfiguraatio tiedosto configuration.json päivittää. Kyseinen tiedosto esitellään seuraavassa osassa.

Volumes

Docker volyymit ovat tiedostoja, jotka liitetään Docker kuvaan, ja ne antavat mahdollisuuden konfiguroida Docker kuvia ilman että niitä tulisi uudelleen asentaa. Seuraavat alaosat esittelevät nämä volyymit, niiden käyttötarkoituksen ja tarvittavat muutokset jotka niihin tulee tehdä riippuen käyttötavasta.

configuration.json

Tämä tiedosto on tärkein konfiguraatiotiedosto järjestelmän toimivuuden kannalta. Konfiguraatiotiedosto sisältää seuraavat kohdat:

operating_mode

- Jos tämä on 0, myöhemmin esitettävä PostgreSQL on käytössä työtehtävän lisätietojen hakemista varten, jos 1, tarvittavat lisätiedot on annettu Työtehtävä -entiteetin "workParameter" attribuutissa

config_ac, jota käytetään Application Controller konfigurointiin

- *gloves_connected*: yhdistettyjen hanskojen lukumäärä (1-2). Järjestelmä toimii toistaiseksi parhaiten yhdellä hanskalla
- *gesture_list*: Lista käytettävistä käsimerkeistä. Jokainen käsimerkki pitää sisällään kolme kenttää, taite sensorien statukset, painesensorien statukset ja käsimerkin nimi. Viimeisimmässä versiossa painesensorit eivät ole käytössä luotettavuusongelmien vuoksi

- *command_map*: kartoittaa käskyn indeksinumeron käsimerkkiin. Jokainen käsimerkillä tulee löytyä myös *gesture_list* attribuutista
- *postgres*: käytetään PostgreSQL tietokannan konfigurointiin, jos sellaista käytetään Työtehtävien määritelmien tallentamiseen. Jos *self_deploy* attribuutti on 1, Application Controller tekee tarvittavan taulukon työtehtäville sen testaamista varten. Jos kyseistä tietokantaa ei käytetä, tai jos sen käynnistää/täyttää joku muu ohjelma, tämän tulee olla 0. Esimerkki itse tehdystä taulukosta löytyy *1*.
- *query*: Tämä attribuutti kertoo, että mitkä rivit tulisi hakea ja mistä taulukosta PostgreSQL tietokannasta. WHERE attribuutin täytyy vastata saraketta jossa sijaitsee työtehtävien koodit. Näiden tietojen konfigurointiin annetaan mahdollisuus jotta käyttäjä voi itse valita sarakkeiden nimet.

config_bridge, jota käytetään ros2-fiware-bridge:n konfigurointiin

- *self_deploy_device*: Kun tämä on 1, Laite entiteetti luodaan sillan toimesta. Tämä on normaalikäytäntö
- *orion_url*: Jos docker composea käytetään Orion Context Brokerin (OCB) luontiin, käytä tässä <kontin_nimi>:1026 (normaalisti ”orion:1026”). Jos OCB käynnistetään jossain muualle (toisella host tietokoneella), käytä kyseisen host tietokoneen IP osoitetta kontin nimen sijasta
- *device_entity*: Laite entiteetti, jonka silta itse julkaisee. Tämän entiteetin status ja käskyn indeksinumero päivittyy tarvittaessa Application Controllerin toimesta
- *update_status*: json jolla päivitetään työtehtävä entiteetti
- *update_device_state*: json jolla päivitetään laite entiteetin status
- *update_command_id*: json jolla päivitetään laite entiteetin käskyn indeksinumero
- *subscription_entities*: Tämä sisältää kaikki komponentin tilaukset FIWAREen. Ensimmäinen tilaa kaikki työtehtävä entiteetit, ja suodattaa ilmoitukset laite indeksinumeron mukaisesti. Tällä hetkellä komponentti tukee vain työtehtävä entiteettejä, joissa on vain yksi laite, jolle työtehtävä asetetaan. Toinen tilaa statuksen muutokset sen varalta, jos jostain syystä joku toinen applikaatio päivittää työtehtävän statuksen joka oli annettu M2O2P komponentille

1

Table 1. Esimerkki miltä TaskDef taulukko näyttää

task_id	task_process_id	task_name	task_code	command_id
1	2	Reach the tray COMMAND	RTTC	1
2	2	Freedrive COMMAND	FC	2
3	2	Grasp component COMMAND	GCC	3

ESIMERKKI KONFIGROINTITILANNE:

OCB ajetaan toisella host tietokoneella, jonka IP on 192.168.7.21. M2O2P kommunikoi FIWAREn kanssa käyttäen ROS2-FIWARE siltaa, jonka takia kaikki tarvittavat muutokset tulee tehdä sillan konfiguraatio osuudessa. Jokainen URI jossa on ”orion” tulee muuttaa vastaamaan kyseistä IP osoitetta. Myös tilaus entiteetit tulee muuttaa siten, että se URI vastaa sillan host tietokoneen IP osoitetta.

Lisää esimerkkejä lisätään jos/kun sellaisia löydetään testausvaiheessa

ros2_ros1_command.yaml

Tätä tiedostoa käytetään integration-servicen konfigrointiin. YAML-tiedostoa käytetään kahden järjestelmän yhdistämiseen, antamalla reitti näiden kahden järjestelmän välissä. Tähän ei tarvitse tehdä muutoksia. Silta yhdistää seuraavan nimiset aiheet ROS2 (Foxy) ja ROS1 (Noetic) väliohjelmistoissa:

- command_id
- finger_sensor_values_left
- finger_sensor_values_right
- finger_states_left
- finger_states_right

original_limits.txt

Tätä tiedostoa käytetään ilmoittamaan käynnistysvaiheessa taite sensorien raja-arvot kolmelle eri tilalle, joka vastaa siis kahta raja-arvoa, ja yksi raja-arvo painesensoreille. Taite sensorien raja-arvot voidaan säätää Web käyttöliittymällä, joten tähän ei välttämättä tarvitse koskea heti. M2O2P ilmaisessa versiossa tätä ei voi tehdä, joten tällöin vihjeeksi:

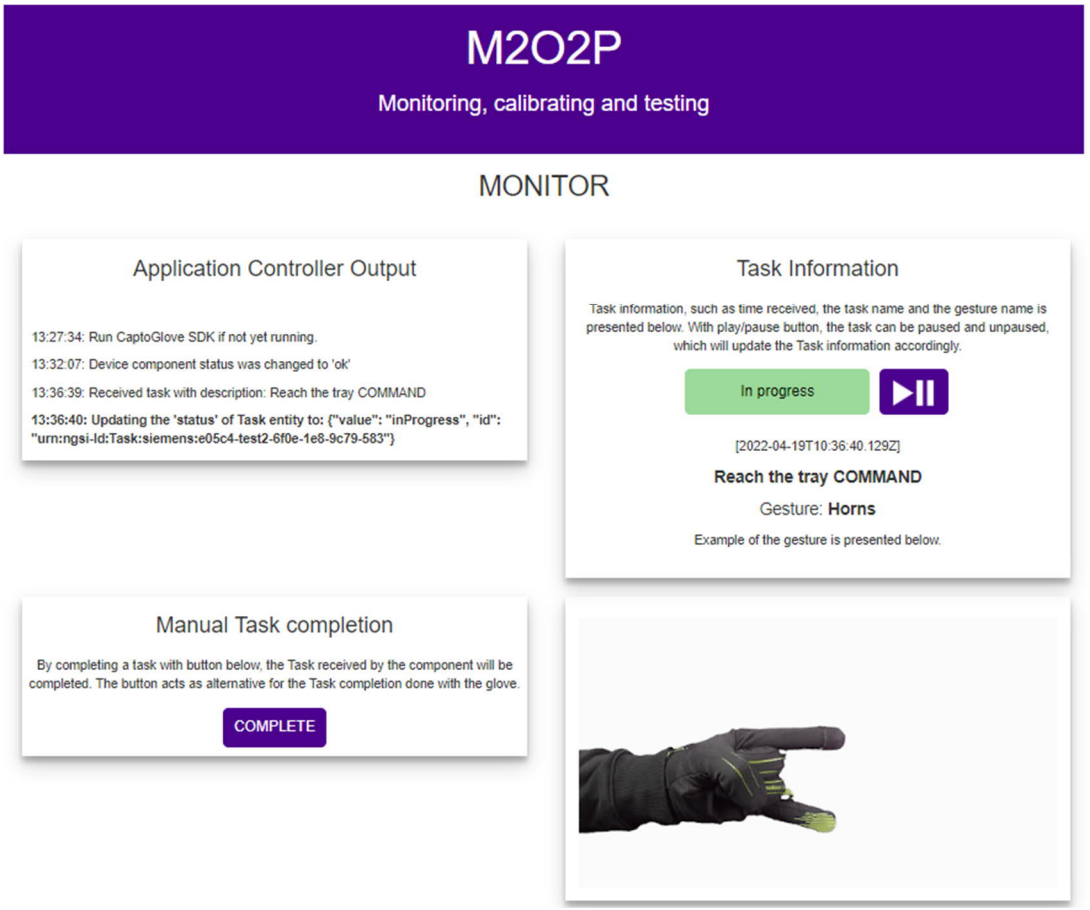
- Suorista kaikki sormet ja tarkista kaikkien sensorien arvot joko CaptoGlove SDK:sta ottamalla näyttökuva, tai jokaiselle sensorille arvo käyttäen Capto Suite -ohjelmaa. Tämän jälkeen ylemmät raja-arvot voidaan säätää esimerkiksi 500 pienemmäksi kuin suoristetun sormen sensoriarvo, ja samoin pienemmät 500 suuremmaksi kuin taitetun sormen sensoriarvo (arvot voivat olla jotain muuta, toiminnallisuus täytyy kokeilla testaamalla). Täten jokaiselle taitesensorille on kolme tilaa, suoristettu, jotain suoristetun ja taitetun välistä, ja taitettu.
- Hanki M2O2P kokonaisversio

KÄYTTÖLIITTYMÄ

Kun Docker-kontit ovat käynnistetty, Web käyttöliittymään pääsee käsiksi host-tietokoneella navigoimalla osoitteeseen *localhost:54400* selaimella, tai vaihtoehtoisesti toiselta koneelta käyttäen *localhost* sijasta kyseisen tietokoneen IP osoitetta (tämä kuitenkin vaatii konfigurointia tietokoneen internet asetuksissa). Seuraavaksi käydään lyhyesti läpi käyttöliittymän eri osat:

Ensimmäiseksi käyttöliittymässä on monitorointi osa. Tämä on ainut osa käyttöliittymästä, joka sisältyy ilmaiseen versioon komponentista. Osa sisältää neljä lohkoa:

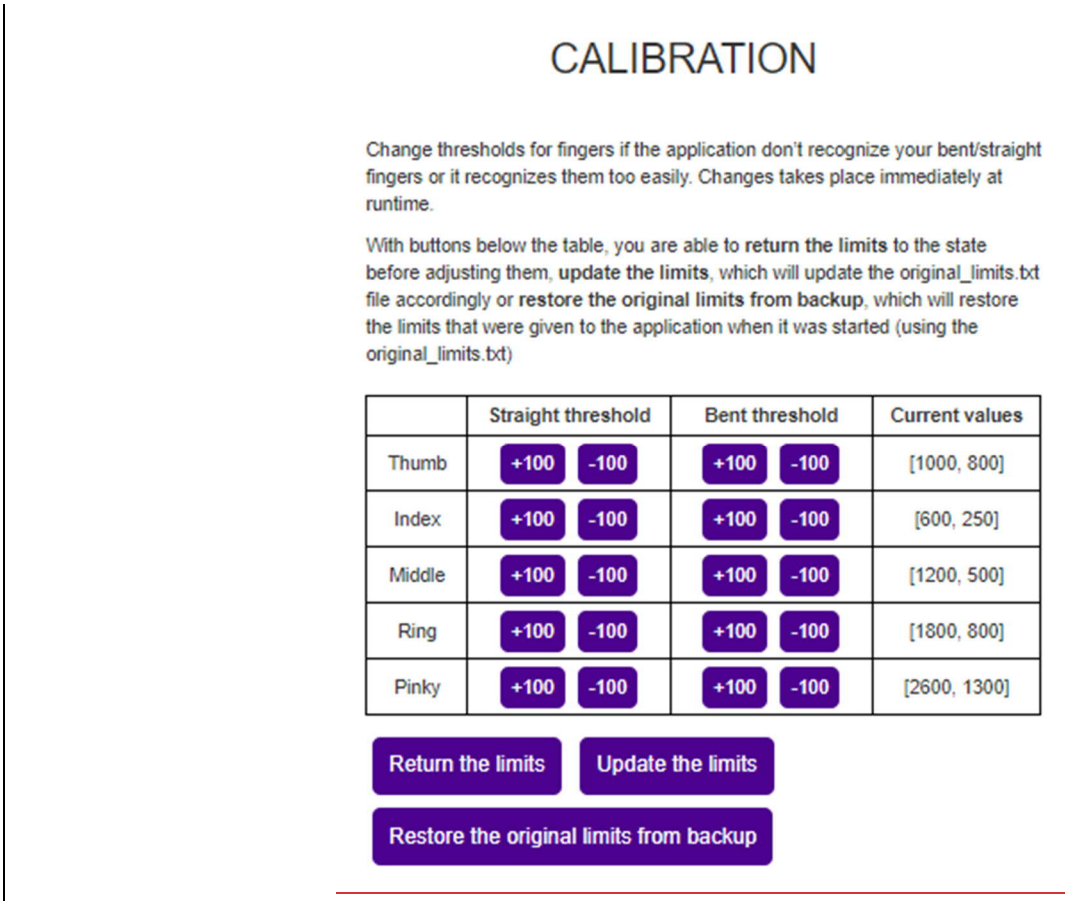
- *Application Controller output*: näyttää AC:n ulostulon, ja näyttää hyödyllistä tietoa ohjelman taustalla pyörivistä prosesseista
- *Task Information*: Kuvassa näkyvä vihreä laatikko saa eri värejä (vihreä/keltainen/harmaa) riippuen työtehtävän tilasta (keskeneräinen/tauolla/suoritettu tai ei annettu). Play-pause painike antaa käyttäjälle mahdollisuuden laittaa työtehtävän tauolle, ja takaisin keskeneräiseksi. Tämän lohkon alla esitetään käsimerkistä GIF-animaatio esimerkkinä käyttäjälle.
- *Manual task completion*: Tätä painiketta painamalla käyttäjä voi suorittaa tehtävän ilman sensorihanskan käyttöä



Seuraavana käyttöliittymässä on KALIBROINTI osio

Tässä osiossa käyttäjä voi muuttaa raja-arvoja käyttämällä osiossa näkyvää taulukkoa. Jokaiselle sormelle voidaan säätää suoran ja taitetun sensoriarvon raja-arvo. Jos suoristetun raja-arvon +100 painiketta painetaan, tämänhetkinen korkeampi raja-arvo kasvaa sadalla. Kyseinen muutos tulee voimaan heti ajoaikana.

Taulukon alapuolella on kolme painiketta, *return the limits* joka palauttaa raja-arvot, jotka olivat olemassa ennen kuin muutoksia tehtiin taulukon avulla, *update the limits* joka tallentaa muutetut raja-arvot original_limits.txt -tiedostoon ja muutokset pysyvät voimassa myös uudelleen käynnistäessä, ja *restore the original limits from backup* palauttaa ne raja-arvot, jotka komponentille annettiin original_limits.txt tiedostossa sovellusta käynnistäessä.

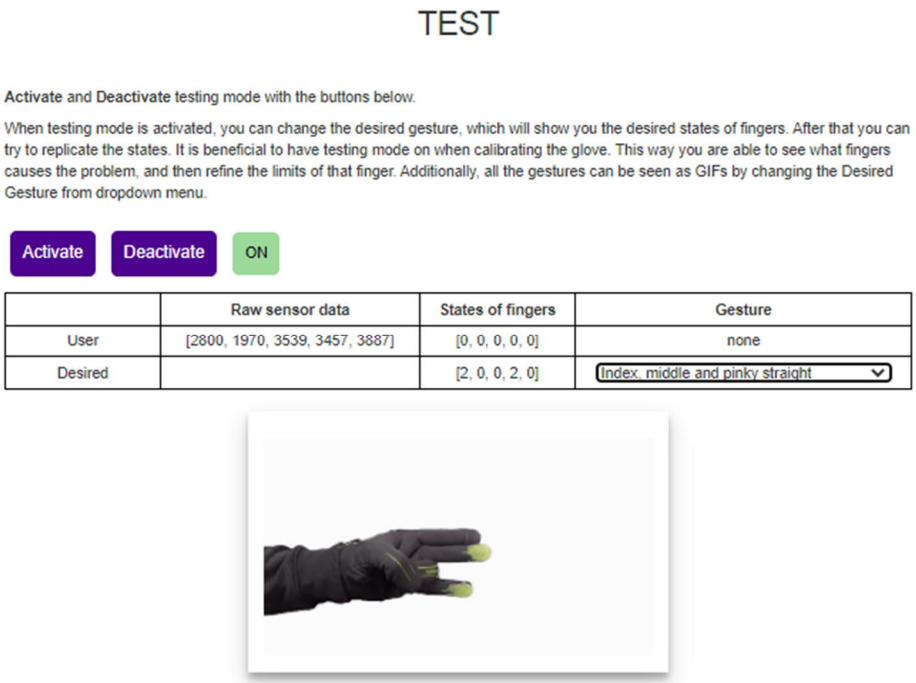


Seuraavana käyttöliittymässä on TESTAUS osio.

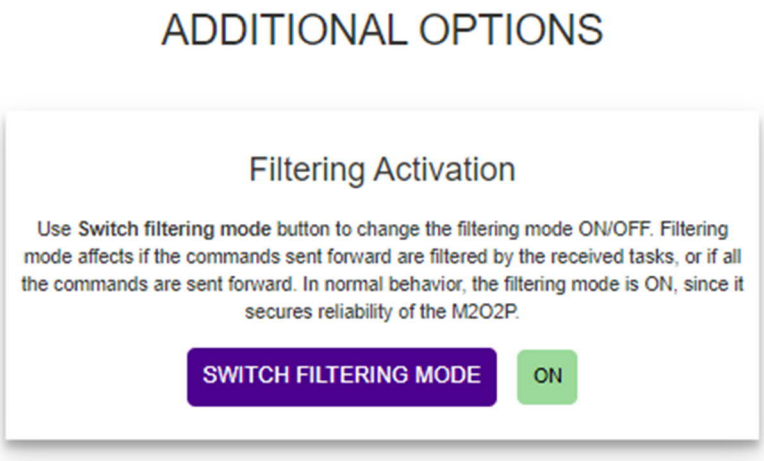
Tässä osiossa sensorihanskaa voidaan testata. Jos *Activate* painiketta painetaan, testaus tila käynnistyy, ja sensorihanskan sensoriarvot ja taitesensorien tilat näkyvät taulukossa. Testaus-tilassa komponentti ei suorita työtehtäviä sensorihanskalla.

Ensimmäisellä rivillä taulukossa on raaka sensoridata ja tilat jokaisen sormen sensorille, ja käsimerkki jos sellainen tunnistetaan kyseisellä hetkellä. “Desired” -rivillä käyttäjä voi valita käsimerkin pudotusvalikosta, jonka hän yrittää toistaa. Kyseisen käsimerkin vaadittavat tilat näkyvät *States of Fingers* sarakkeessa ja GIF-animaatio esitetään taulukon alapuolella, vaikkei testaus-tila olisikaan päällä.

Testaus tilaa voidaan käyttää käsimerkkien harjoittelemiseen, ja yhdessä kalibroinnin kanssa kalibroimaan hanska.



Viimeisenä käyttöliittymässä on LISÄASETUKSET, jossa suodatus tila voidaan laittaa päälle ja pois. Jos suodatustila on pois päältä, käyttäjä voi tehdä minkä vain käskyä vastaavan käsimerkin, pitää sitä 1,5 sekuntia, jolloin käskyn tunnistenumero lähetetään Laite-entiteettiin. Tämän tilan käyttö voi tulla kysymykseen, jos käskyjä halutaan testata ilman että komponentille on asetettu työtehtävää, tai jos käyttäjä haluaa, että kaikki käskyt lähetetään. Näin voisi olla esimerkiksi tilanteessa, jossa käyttäjällä on neljä eri käsimerkki valintaa, jotka kaikki tekevät jonkun käskyn. Normaalissa tilanteessa tämä on kuitenkin päällä, koska suodatus tekee sensorihanskan käytöstä luotettavampaa.



TESTAUS

Nyt kun kaikki järjestelmät ovat asennettuna ja konfiguroitu, komponenttia voidaan testata. Ensimmäiseksi tulee mennä kansioon jossa docker-compose.yml sijaitsee, ja käyttää *docker compose up* käskyä kyseisestä kansiosta komentoriviltä. Kun Docker-kontit ovat käynnissä, suorita CaptoGloveSDK.exe tiedosto. Jos kaikki on mennyt tähän asti oikein, komentokehotteessa tulisi näkyä ulostulo ”<left or right> glove connected” ja CaptoGloveSDK.exe:n ikkuna tulisi näyttää jatkuvaa ulostuloa sensoriarvoista. Nyt voidaan navigoida selaimella osoitteeseen *localhost:54400* jossa tulisi näkyä edellä esitetty käyttöliittymä. Jos nyt käytetään Postman -sovellusta ja Postman collectionia, joka sijaitsee komponentin GitHub sivulla, voidaan käyttää GET pyyntöä, jolla haetaan kaikki Laite-entiteetit, jossa pitäisi olla mukana juuri luotu Laite-entiteetti.

Käyttöliittymän TESTAUS-osiota voidaan käyttää nyt raja-arvojen hiomiseksi ja käsimerkkien testaamiseksi. Jotta annettuja työtehtäviä voidaan testata, voidaan Postman -sovelluksessa luoda uusi työtehtävä komponentille. Työtehtävän tunnistenumero tulee olla aina uusi, joten jos halutaan testata monta kertaa putkeen, tulee tunnistenumeron olla aina uusi. Mallipohjassa ”task_code” on valmiiksi “RTTC”, joka vastaa “Reach the TRAY COMMAND” työtehtävää. Kun työtehtävä luodaan, käyttöliittymään päivittyy työtehtävän tiedot. Komponentti päivittää työtehtävän tilaksi ”inProgress”, joka voidaan testata käyttämällä GET pyyntöä, joka hakee kaikki työtehtävä-entiteetit. Kun käsimerkki tehdään ja sitä on pidetty 1,5 sekuntia, käyttäjä voi huomata *Application Controller Output* lohkossa, että työtehtävä päivitettiin tilaan ”completed”. Tämä voidaan todentaa taas kerran Postmanillä hakemalla kaikki työtehtävä-entiteetit. Laite-entiteetti pitäisi myös olla päivittynyt käskyn tunnistenumeroon.