Tallinna Ülikool

Informaatika Instituut

Leap Motion: Uue generatsiooni 3D sensor

Bakalaureusetöö

Autor: Kristen Kivimaa

Juhendaja: Jaagup Kippar

Autor: ..... ,, ... ,, 2015

Juhendaja: ..... ,, ... ,, 2015

Instituudi direktor: ..... ,, ... ,, 2015

Tallinn 2015

Autorideklaratsioon

Deklareerin, et käesolev bakalaureusetöö on minu töö tulemus ja seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on viidatud.

....

(kuupäev) (autor)

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina _____ (sünnikuupäev: _____)

(*autori nimi*)

1. annan Tallinna Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

_____
_____
_____

(*lõputöö pealkiri*)

mille juhendaja on _____,

(*juhendaja nimi*)

säilitamiseks ja üldsusele kättesaadavaks tegemiseks Tallinna Ülikooli Akadeemilise Raamatukogu repositooriumis.

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tallinnas/Haapsalus/Rakveres/Helsingis, _____

(digitaalne) allkiri ja kuupäev

Sisukord

Sissejuhatus

Viimaste aastate jooksul on välja töötatud mitmeid erinevaid optilisi andureid, mis on võimelised tuvastama kolmemõõtmelisi objekte. Paljud neist suudavad aru saada ka objekti liikumisest ning see omakorda loob võimaluse sensorite poolt saadud andmeid kasutada erinevate rakenduste loomiseks. Kõige tuntum 3D sensor on tõenäoliselt Microsoft – i poolt arendatud Kinect. Nagu paljud teised sarnased tooted, jälgib ka Kinect inimese terve keha liikumist ning see annab võimaluse luua rakendusi, kus kasutajal tuleb erinevate toimingute jaoks liigutada enda jäsemeid. 2013ndal aastal tuli müügile aga uudne ning ainulaadne seade, mis kannab nime Leap Motion.

Tegemist on 3D anduriga, mis ei jälgi enam kasutaja terve keha liikumist, vaid keskendub ainult kätele. Seade võimaldab kasutajal käsi liigutades suhelda arvutis töötavate rakendustega ning seetõttu vähendab arvutihiire tähtsust. Kontroller tunneb eraldi ära inimese kümme sõrme ning suudab täpselt ära määrata, kuidas need liiguvad ning mis asendis on. Firma ise nimetab enda andurit uue generatsiooni seadmeks, sest teist sama funktsionaalsusega sensorit müügil ei ole.

Autor valis selle teema, kuna Eestis on Leap Motion üsna tundmatu ning varasemalt pole selle kohta suuremaid uurimustöid koostatud.

Bakalaureusetöö eesmärk on tutvustada Leap Motion seadet, selle ajalugu, tööpõhimõtet ning arendada kaks näidisrakendust, et uurida, kuidas nende loomine käib. Selleks, et proovida erinevaid arendusviise, lõi autor ühe veebirakenduse, kasutades Javascript programmeerimiskeelt, ning ühe töölaua keskkonnas käivitatava rakenduse, kasutades Unity mängumootorit ja C# programmeerimiskeelt. Autor valis need arendusviisid, sest on mõlema programmeerimiskeelega varem kokku puutunud. Veel on eesmärgiks uurida kasutajate kogemust selle seadmega töötamisel, mille tarvis koostas autor ingliskeelse küsitluse ning jagas seda erinevatel Leap Motion teemalistel veebilehtedel ning foorumites.

Bakalaureusetöö koosneb kolmest peatükist, mis omakorda jagunevad alapeatükkideks. Esimeses peatükis tutvustab autor erinevaid 3D sensorite tööpõhimõtteid ning nimetab ära neil töötavad populaarsemad seaded.

Teise peatüki on autor jaganud neljaks alapeatükiks, kus esimene tutvustab Leap Motion seadet, teine alapeatükk tutvustab ajalugu ning arenduslugu, kuidas seade loodi. Kolmas kirjeldab kontrolleri ning tarkvara tööpõhimõtet ja seda, kuidas need kaks omavahel andmeid vahetavad. Neljas alapeatükk tutvustab erinevaid arendusvõimalusi, millega on võimalik Leap Motionile rakendusi arendada.

Kolmandas peatükis kirjeldab autor, kuidas loodi uurimustöö käigus arendatud rakendused ning analüüsib kasutajate kogemusi. Esimeses alapeatükis kirjeldatakse Unity mängumootoriga loodud äpi arenduskäiku ning teises veebipõhise rakenduse arenduskäiku. Kolmandas alapeatükis toob autor välja kasutajate poolt antud hinnangud sellele seadmele.

Samuti tekkis autoril võimalus teha Skype'i kõne San Franciscosse, Leap Motioni peakorterisse. Bakalaureusetöö kirjutamisel kasutas autor ka infot, mis antud intervjuust saadi.

Võõrkeelsete lühendite loetelu

Loetelu on koostatud tuginedes Vallaste sõnastikule (Vallaste, 2015).

API – rakendusliides, programmiliides, API-liides Arvuti operatsioonisüsteemiga või rakendusprogrammiga määratud reeglistik, mille alusel rakendusprogramm kasutab operatsioonisüsteemi või teise rakendusprogrammi teenuseid.

C# - C#-keel (ingl. k. häädlus "sii-sharp", eesti k. häädlus "C-diees") on objektorienteeritud programmikeel firmalt Microsoft, mis üritab ühendada C++ arvutusvõimekust Visual Basic'u programmeerimislihtsusega.

C++ - C++ keel on C-keele laiendatud edasiarendus, kus C-keelele on lisatud objektorienteeritud programmeerimise kontseptsioone

Core – südamik Põhimälu ehk muutmälu. See termin pärineb aegadest, kus põhimäludes kasutati mäluelementidena ferriitsüdamikke. Tänapäeval on sõna "südamik" muutunud arhaismiks.

ESC (escape) - pagu, paomärk ASCII-kood 27. Kui selle saadab kasutaja, siis kasutatakse seda sageli programmi täitmise või andmete sisestamise abortimiseks.

HTML – hüpertekst-märgistuskeel Enimlevinud kodeerimissüsteem (tekstivorming) veebidokumentide loomiseks. HTML koodid ehk märgendid määravad ära selle, kuidas veebileht arvutiekraanil välja näeb.

Java – Java on kõrgkeel, mis on mõeldud spetsiaalselt kasutamiseks Interneti hajuskeskkonnas.

JavaScript – Netscape'i poolt välja töötatud skriptikeel, mis võimaldab veebiautoritel luua interaktiivseid veebisaite.

Objective – C – C-keele esimene objektorienteeritud kommertsversioon. See töötab IBM PC ja Macintosh'i arvutitel ning mitmesugustes UNIX'i tööjaamades.

Python – Interpreteeritav objektorienteeritud programmikeel, mille lõi Guido van Rossum 1990.a. Keele nimetus on pärit Monty Python'i telesarjast "Flying Circus".

Rendering – visualiseerimine Andmete teisendamine kuvamiseks või printimiseks sobivasse vormingusse. Näiteks kiirte jälituse meetodi puhul võetakse kolmemõõtmelise objekti (või objektide kogumi) matemaatiline mudel ja teisendatakse see kahemõõtmeliseks bittrasterkujutiseks, mida saab ekraanil vaadata.

Socket – sokkel IP-protokollil põhinevas võrgus, näiteks Internetis, nimetatakse antud hosti sokliks sidekanali otspunktina toimivat IP-aadressist ja pordinumbrist koosnevat identifikaatorit (Cisco määrang). Sageli loetakse sokli määrangusse kuuluvaks ka kasutatav sideprotokoll (harilikult TCP või UDP).

TCP – Edastsohje protokoll Levinuim võrgu transpordikihi protokoll, mida kasutatakse Etherneti võrkudes ja Internetis.

TLS – transspordikihi turbeprotokoll Avatud protokoll, mis võimaldab klient-server rakendustel omavahel turvaliselt suhelda üle Interneti, olles kaitstud pealtkuulamise või sõnumite rikkumise ja võltsimise eest.

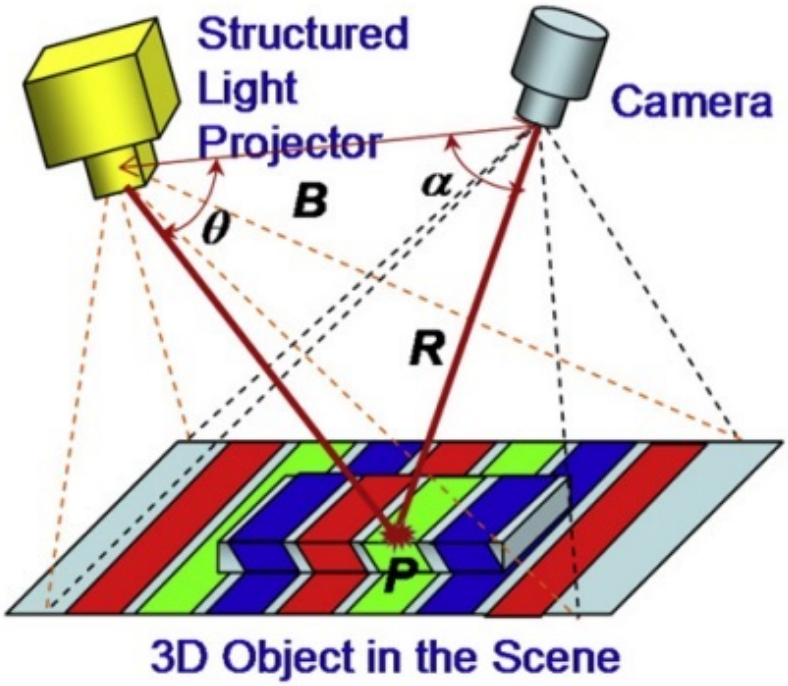
Widget – vidin. Arvutite valdkonnas graafilise kasutajaliidese (GUI) element, mis kuvab informatsiooni või pakub kasutajale võimalust suhelda opsüsteemi või rakendusprogrammiga.

1 Sarnased seadmed ning tööpõhimõtted

Kuna Leap Motion 3D sensor on üsnagi uus leiutis, mille kohta on koostatud väga vähe erinevaid uurimustöid, siis otsustas autor esmalt anda ülevaate erinevatest 3D sensorite tööpõhimõtetest. 3D pildi loomiseks tuleb mõõta skaneeritava pildi sügavust ehk peab olema määratud z – telg. Sensorid mõõdavad z – telge erinevalt ning seetõttu saabki neid tööpõhimõtete järgi jagada järgnevateks mehhanismideks: Struktureeritud valgus (*Structured Light*), Lennuaeg (*Time of Flight*) ja Ruumiline Nägemine (*Stereo-Vision*).

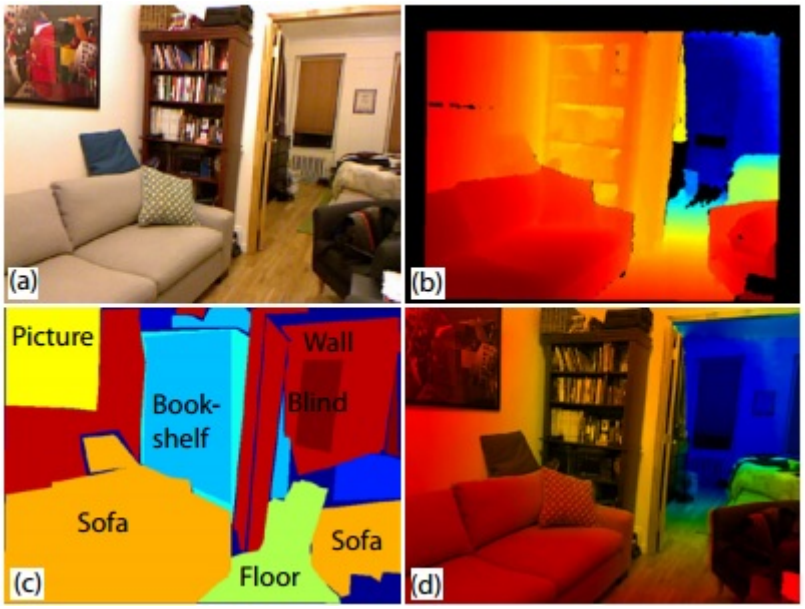
1.1 Struktureeritud Valgus

Üks peamine meetod pinna 3D skaneerimiseks põhineb struktureeritud valguse kasutamisel. See tähendab, ruumi tavalisele 2D pildile lisatakse juurde spetsiaalse valgusprojektoriga saadud, erineva intensiivsusega muster, mis näitab esemete kaugust kaamerast. Nagu näidatud joonisel 1, on lisaks kaamerale, mis kuvab 2D pildi, valik ka spetsiaalne valgusprojektor. Iga piksli intensiivsus struktureeritud valguse mustris on esitatud digitaalsignaaliga {Iij = (i, j), i= 1, 2, …, I, j = 1, 2, …, J}, kus i ja j tähistavad projekteeritud mustri x ja y koordinaate. Seletatud struktureeritud valguse meetodi korral töödeldakse 2D mustreid (Geng, 2010).



Joonis 1. Struktureeritud valguse skaneerimine. (Geng 2010)

Enim levinud seade, mis töötab struktureeritud valguse põhimõttel, on Microsofti poolt välja töötatud Kinect. See seade loob skanneritavast pildist täpse valguskaardi, mis joondatakse ruumiliselt ning ajaliselt aparaadi veebikaameraga (vaata joonis 2) (Fergus ja Silberman, 2012). Kinecti eelised teiste sama tööpõhimõtet kasutavate seadmete ees on selle täpsus, kompaktsus, teisaldatavus ning taskukohane hind. Need omadused muudavad selle seadme kasutatavaks mitmetes nägemisrakendustes, mis on abiks näiteks vaegnägijatele ning robotite navigeerimisel.



Joonis 2. Toa pilt läbi Microsoft Kinecti (Fergus ja Silberman, 2012).

Joonis 2 on kokku pandud neljast erinevast pildist. Pilt a on tavaline veebikaamera poolt pildistatud 2D kujutis. Pildil b on kujutatud toores sügavuskaart[1], mis on saadud Kinectis olevast valgussensorist. Sellel pildil tähistab punane kaamerale lähedal ning sinine kaugemal asuvaid objekte. Pildil c on märgistatud kõik toas äratuntavad objektid. Pildil d on kokku pandud veebikaamera ning struktureeritud valgussensori pildid pärast eeltöötlust, millega täidetakse puuduvad piirkonnad.

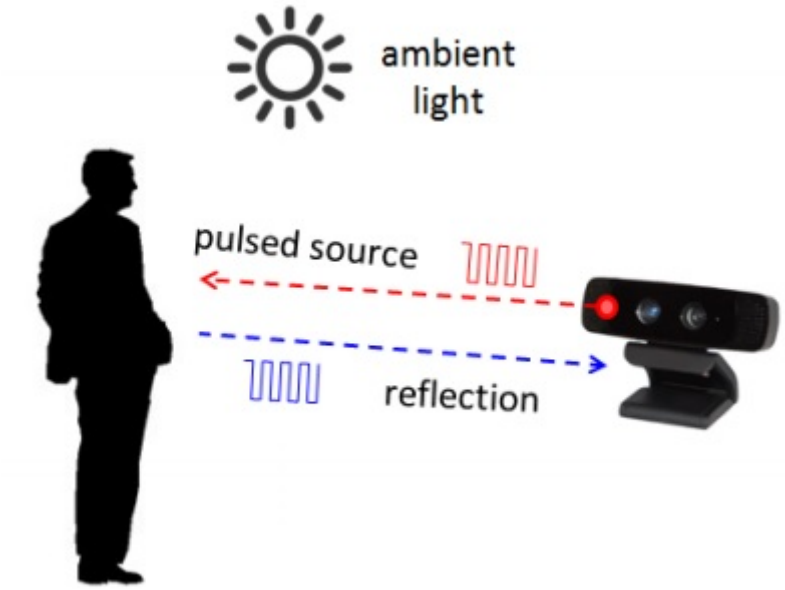
Kuigi Kinect on arendatud põhiliselt töötamiseks koos Microsoft Xbox mängukonsooliga, on sellel ka erinevaid ühendusliideseid, et sellega saaks töötada ka läbi arvuti. Spetsiaalselt arvutite jaoks on Asus välja töötanud seadme nimega Xtion Pro Live. See seade töötab täpselt samal põhimõttel nagu Kinect ning isegi nende välimus on sarnane.

[1] Sügavuskaart on 3D punktide kogumik (igat punkti nimetatakse vokseliks (Li, 2014))

1.2 Lennuaeg

Lennuaja (*Time of flight*, edaspidi TOF) põhimõtteid on mitmeid. Selles peatükis keskendub autor aga ühele kindlale meetodile, milleks on Pidev Modulatsioon (*Continious Modulation*), ehk pilti skaneeritakse pidevalt, mistõttu saadaksegi arvutiekraanile 3D kujutis. Autor tegi sellise valiku, kuna just see meetod on tänapäeval laialdaselt kasutuses. Seda peamiselt riistvara pärast, mis käesoleval ajal on kergesti poodidest kättesaadavad.

TOF kaamera valgustab skaneeritavat pilti moduleeritud valgusallikaga ning vaatleb tagasi peegeldunud valgust. Aeg, mis kulub väljasaadetud ajast kuni peegelduse tagasi jõudmiseni kaamerasse mõõdetakse ning arvutatakse ümber kauguseks. Joonis 3 illustreerib klassikalist Lennuaja põhimõttel töötavat 3d kaamerat (Li, 2014).



Joonis 3. 3D TOF kaamera tööpõhimõte (Li, 2014).

Tavaliselt on valgusallikaks laserkiir või LED valgusele põhinev infrapunakiir, mis mõlemad on inimsilmale nähtamatud. Lasersensori puhul mõõdetakse kaugust objektist, lastes kiir sensorist välja ning mõõtes aega, millal see tagasi peegeldub. Infrapunal töötava seadme puhul mõõdetakse faasinihet väljastatud ning tagasi peegeldunud valguse vahel (Bachmann, Fisseler, Rudak, & Weichert, 2013).

Sensor, mis on võimeline enda poolt välja lastud ning tagasi peegeldunud kiirele reageerima, muudab saadud footonenergia elektriliseks impulsiks. Kuna skaneeritavas keskkonnas on alati

ka taustavalgus, siis peavad seadmed olema võimelised eristama tagasi peegeldunud ning välisteguritest tingitud impulsse, sest informatsioon distantsi kohta on salvestatud peegelduvasse komponenti. Seega vähendab suur taustamüra signaali ja müra suhet (Li, 2014).

TOF anduritega mõõdetakse iga 2D pildil oleva piksli kaugust, mis lisatakse sellele massiivina. Selle tulemusena valmib sügavuskaart. Sügavuskaart visualiseeritakse arvutis kolmemõõtmelisse keskkonda punktide kogumikuna või punktide pilvena. 3D punktid ühendatakse matemaatiliste valemitega ning seeläbi saadaksegi võrgustik, mis loob 3D kujutise. Kui skaneeritav tekstuur on reaalaajas värviline kujutis, tõuseb esile kolmemõõtmeline pilt nagu on näha pildil 4 (Li, 2014).

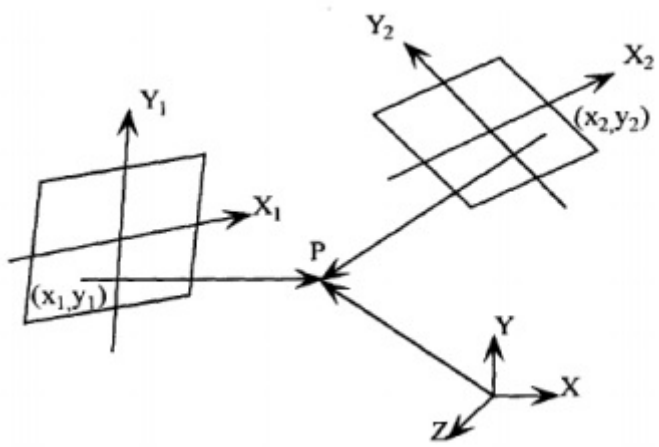
Joonis 4. 3D visualiseeritud inimene

Tänapäeval kasutusel olevatest TOF mehhanismil töötavatest kaameratest on enimlevinud Mesa Imaging'i poolt toodetav Swissranger 4000 ning PMDVisioni poolt toodetav CamCube 3.0. Need mõlemad töötavad infrapunakiirte abil. Laserkiirte tehnoloogiaga töötavatest masinatest tasuks välja tuua Laser Sicki poolt toodetav LMS511.

1.3 Ruumiline nägemine

Kuna ruumiline nägemine sarnaneb inimese bioloogilisele süsteemile, on sellel mõned omadused identsed. Näiteks on inimesel kaks silma, et näha samast keskkonnast pisut erinevat pilti. Ruumilise nägemise mehhanismil on kaks kaamerat paigutatud kindla vahemaaga ning need teevad samaaegselt pilte. Kasutades kaamerate geomeetriat, on võimalik teatud algoritme kasutades luua skaneeritava keskkonna geomeetria. Stseeni z – telje määramisel otsitakse mõlema 2D pildi samasuguseid punkte (Bachmann et al., 2013, Calin 2013).

Kuna need kaks kaamerat sellel seadmel on väga lähestikku, siis salvestavad nad peaaegu samasugust pilti ainult erineva nurga alt. Sel hetkel, kui leitakse kahel pildil samasuguste pikslite paar, peaksid nende projektsiooni kiired mingis punktis lõikuma. Selle tõttu ongi võimalik teha kindlaks selle punkti koordinaadid kolmemõõtmelises ruumis (vaata joonis 5). Kui teha kindlaks skaneeritava objekti kõik pinna punktid, saabki selle kujundi ning positsiooni üheselt määrata (Li, Liu, Miao & Wan, 2010).



Joonis 5. Ruumilise nägemise põhimõte (Li et al., 2010).

Joonis 5 kujutab ruumilist nägemist, kus vaadeldav ruumiline punkt on P. Erinevatest kaameratest saadud piltide järgi on P – l erinevad x ning y telje koordinaadid (X_i, Y_i | i=1, 2). Punkti P kolmemõõtmelised koordinaadid (X, Y, Z) on arvutatavad teades mõlema kaamera asukohta teineteise suhtes ning kasutades absoluutse erinevuse summat (*Sum of Absolute Difference*, edaspidi SAD).

SAD on lihtne ja populaarne algoritm ruumilise nägemise koordinaatide arvutamiseks. Kuna selle töötamiseks pole vaja teisendamisfunktsioone, ei koorma see ka eriti seadme riistvara. Selle põhifunktsioon C_{ad} annab kahelt pildilt mõõdetud sama punkti pikslite väärtuste absoluutse erinevuse:

$$C_{ad}\left(i_{1_{x,y}}, i_{2_{x+d,y}}\right) = |i_{1_{x,y}} - i_{2_{x+d,y}}|$$

kus $i_{1x, y}$ on põhikaamera pildist saadud piksli tugevus x ja y koordinaatidega ning $i_{2x+d, y}$ on teise kaamera salvestatud sama piksli tugevus, kus x – ile on liidetud kaameratevaheline erinevusaste d.

Kõige populaarsemateks ruumilise nägemise mehhanismi kasutavateks kaamerateks on Calin 2013 järgi Point Gray poolt loodud Bumblebee 2 ning Bumblebee XB3 mis suudavad luua 3D pildi. Bumblebee 2 puhul 640 x 480 töötades resolutsiooniga 48 kaadrit sekundis või 1024 x 768 resolutsiooniga töötades 20 kaadrit sekundis. XB3 pakub kõrgemat resolutsiooni 1280 x 960, töötades 15 kaadrit sekundis.

2 Leap Motion

Järgnevas peatükis tutvustab autor 3D sensorit nimega Leap Motion. Selles peatükis tutvustatakse seadet ennast, selle ajalugu, tööpõhimõtet ning sellele rakenduste arendamise võimalusi.

2.1 Leap Motion seade

Leap Motion seade on 3D sensor, mis tunneb ära käe liigutused selle kohal. Seda peetakse arvutihiirele alternatiiviks, sest sellega töötades ei ole vaja midagi puudutada vaid lihtsalt liigutada enda käsi.

Esimene käed-vaba liikumist tuvastav seade oli Nintendo Wii. Sellega tuvastati seadme liikumist ning saadud teavet kastutati videomängude mängimiseks. Selle edasiarenduseks nimetakse Microsoft Kinecti. See oli edusamm, sest kui Wii puhul jälgiti pulti, mida inimene enda käes liigutas, siis Kinect jälgib inimese terve keha liikumist (Jaroslavsky 2013). Neid seadmed arendati välja peamiselt videomängude tarbeks, et mängija tunneks ennast võimalikult mängu sees olevana. Viimaste aastate jooksul on aga välja arendatud uue generatsiooni liikumist tuvastav seade, mis kannab nime Leap Motion. See sarnaneb küll paljuski Microsoft Kinectile, ainult enam ei keskenduta tervele kehale, vaid ainult käte liigutustele.

Leap Motion kontroller (vaata Joonis 5) on väike seade, mis ühendatakse arvutiga usb ühenduse kaudu. Seadme ostes tuleb sellega kaasa seade ning kaks eripikkusega kaablit. Kontroller ise on kolm tolli pikk, üks toll lai ning pool tolli kõrge (Leap Motion 2015). Alumisel poolel on õhuke kummikiht, mis hoiab ära libisemisvõimaluse. Pealmine pool on kaetud klaasiga.

Joonis 5. Leap Motion seade. (Pcmag 2015)

2.2 Ajalugu

Leap Motion ettevõte on loodud kahe lapsepõlve sõbra, David Holzi ning Michael Buckwaldi poolt. Algselt hakkas sellele seadmele tarkvara arendama Holz 2008ndal aastal, kui ta omandas Põhja – Carolina Ülikoolis Chaper Hillis doktorikraadi matemaatika erialal. Ta hakkas antud seadet looma, kuna pettus arvutihiires ning klaviatuuris, kuna ainuüksi neid kasutades võtsid väga paljud toimingud lootusetult palju aega (Richardson 2013).

Holz mõistis, et tänapäeva arvutid on juba niivõrd võimsad ning seetõttu peaksid need inimeste elu ja tegemisi tunduvalt kiirendama. Kui ta aga õppis omal käel 3D modelleerimist tuli talle ette väikene takistus. Nimelt, võttes kätte tükikese savi ning üritades seda voolida, kulus tahetud kujundi saamiseks mõned minutid, aga kui ta üritas sama teha arvutis, võis tal sama kujundi loomiseks minna tunde. See asjaolu tõstiski temas motivatsiooni luua midagi, mis selle olukorra lahendaks (Foster 2013).

Holzil arendas seadmes olevaid kaameraid viis aastat ning selleks ajaks, kui ta kutsus 2010ndal aastal enda meeskonda Buckwaldi, enda ammuse lapsepõlve sõbra, oli ta juba valmis saanud prototüübi. Nad lõid koos idufirma nimega Ocuspec. Kahjuks oli see prototüüp mõõtmetelt väga suur ning selle üles seadmiseks kulus üle tunni – seega polnud see veel valmis potentsiaalsetele investoritele demonstreerimiseks. Meeste õnneks aga avaldas isegi see suuremõõtmeline seade muljet Avid Technology loojale, Bill Warnerile. Tema sõnade järgi oli see prototüüp küll suur ning kohmakas, aga tõeline võlu peitus selles, mida see seade teha suutis. Nimelt võlus teda asjaolu, et see seade suutis jälgida inimese kõigi kümne sõrme liikumist korraga ning seda hämmastavalt kiiresti, peaaegu ilma mingisuguse mahajäämiseta (Richardson 2013, Tsotsis 2013).

Warner investeeris meeste idufirmasse 25 000 dollarit ning tegi asutajatega kaks aastat tihedat koostööd aidates neil Ocuspeci üles ehitada ning arendada seadet, mille saaksid nad ka suuremale üldsusele esitada. Ta inspireeris neid ning kogenenud ettevõtjana õpetas neil raskustest üles saama (Richardson 2013).

Alates sellest ajast märkasid mitmed firmad ning investorid seda idufirmat ning 2011ndal aastal investeeriti ettevõttesse 1,3 miljonit dollarit. Peamisteks annetajateks olid firmad

nimega Andreesen Horowitz, Founders Fund, SOSventures International ning ettevõtjad Brian McClendon ning Bill Warner (Tsotsis 2011).

2012ndal aastal, kui ettevõtte nimi muudeti Leap Motioniks, kuulutati välja A seeria rahastamine, mille käigus saadi 12,75 miljonit dollarit. Peamiseks rahastajaks oli firma nimega Highland Capital Partners (Bowerman 2012). 2013ndal aastal järgnes sellele B seeria rahastamine, kus juba eelnevalt Leap Motionisse investeerinud ettevõtted Highland Capital Partners ning Founders Fund lisasid sinna veel 30 miljonit dollarit. Samuti sõlmiti koostööleping ka Asus arvutitega (Kosner 2013).

Seega oli Leap Motion ettevõte nende paari aastaga kogunud investoritelt üle 40 miljoni dollari. See avas neile mitmed uksed. Kui 2012ndal aastal töötas seal umbes 12 inimest, siis 2013ndaks aastaks oli see arv suurenenud juba 80 – ni (Richardson 2013).

2012nda aasta kevadel teatas ettevõte ametlikult enda esimesest tootest, millele pandi nimi the Leap. Sama aasta oktoobris alustati tarkvaraarendajate programmiga, mille käigus saadeti kaheteistkümnele tuhandele arendajale, kes oli taotlenud programmile juurdepääsu, seadmest prooviversioonid. Selle eesmärk oli lasta soovijatel arendada seadmele programme ning seda testida. Läbi antud programmi tekitati olukord, kus enne seadme ametlikku väljatulekut loodi juba mahukas äppide kogus, mida tavakasutajad kasutama said hakata kohe peale seadme ostmist (Panzarino 2012).

Ametlikult pidi seade välja tulema eeltellimuse esitanutele 13 mai 2013 ning laiemale ringkonnale 19 mai 2013. Olude sunnil lükkus aga ametlik turuletulemise kuupäev edasi sama aasta juuli kuusse, täpsemalt 27 juuli 2013. Edasilükkumise teemal pidas Michael Buckwald ka pressikonverentsi, kus ta seletas kuupäeva muutumise asjaolusid (Etherington 2013).

Põhjuseks oli peamiselt liialt vähe testitud tarkvara. Leap Motion seadmeid oli turulelaskmiseks valmis tehtud juba 600 000, aga Buckwaldi sõnade kohaselt vajas tarkvara veel viimast korrigeerimist. Samuti sooviti veel arendada valmisolevaid äppe, et seadme ostjateks ei oleks ainult tarkvaraarendajad, vaid ka tavakasutajad, kes saaksid reaalselt seda kasutama hakata enda igapäevategemistes (Etherington 2013).

Kui Leap Motion seade ametlikult välja tuli, ei läinud kõik kaugeltki nii nagu oli algselt plaanitud. Esimeseks turuloleku aastaks ennustati, et ära suudetakse müüa 5 miljonit seadet.

Karm reaalsus oli aga see, et esimesel aastal müüdi umbes 500 000 ühikut. Kuna ootused olid suured, siis oli firmas palgatöölisi 120, kellest kehvade müügitulemuste tõttu koondati 10%, kes töötasid peamiselt turundusosakonnas või tegelesid kasutajatetoega. Pärast esimest tagasilööki hakati parandama seadme tarkvara ning töötati selle kallal, et peatselt oleks võimalik välja lasta järgmine versioon, mille nimeks sai V2 (Etheromgton 2014).

2014nda aasta mai kuus tuligi välja Leap Motion seadmele tarkvarauuendus V2. Kuna V1 (tarkvara esimene versioon) oli küll esmakordsel kasutamisel uus ning huvitav, olid sellel ka mõned vead. Nimelt oli sellel kalduvus skaneeritavate käte näppe omavahel segamini ajada ning vahel isegi ei suutnud see päris korrektselt ja täpselt aru saada kus teatud näpp asetseb (Moynihan 2014).

V2 suudab teha kõike seda, mida suutiski V1, ainult juurde on lisatud veel mõned võimalused. Kui V1 suutis ära tunda käe koos selle sõrmede ning liigutustega, siis V2 tunneb ära kohad, kus on näppudel liigesed ja luud. Näiteks on sellega võimalik arendada programm, mis teeb vahet nimetissõrme nukil ning keskmise sõrme nukil. Seega saab sellega luua programme, kus tuleb ekraanil olevat eset kas haarata või näpistada, mis V1 versioonil võimalik ei olnud. Samuti on seade palju täpsem ning loomulikum (Moynihan 2014).

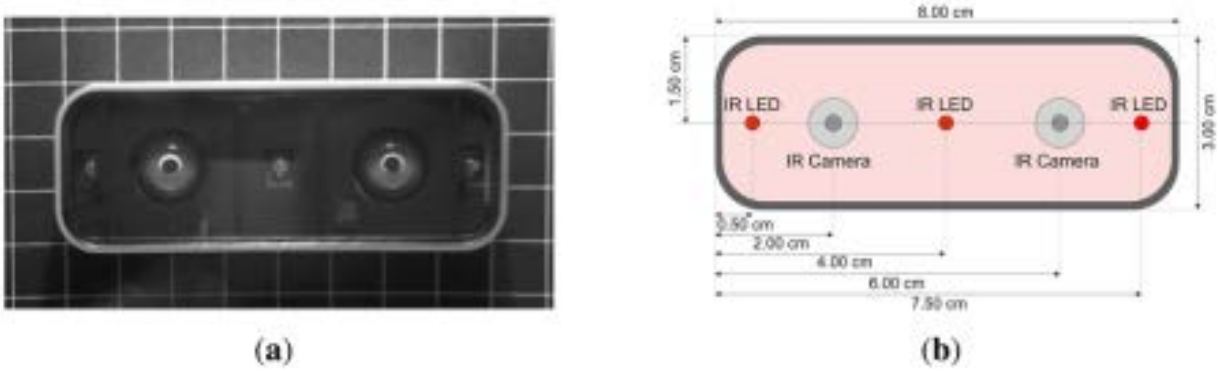
Bakalaureusetöö kirjutamise ajal tegi autor ka Skype'i kõne San Franciscosse, Leap Motioni peakorterisse, ning uuris, millised on nende tulevikuväljavaated. Nagu intervjuust selgus, uut seadet välja anda plaanitud ei ole. Keskendutakse rohkem tarkvara arendamisele. Nende peamine eesmärk on aidata rakenduste arendajaid erinevate abivahenditega, nagu näiteks avatud lähtekoodiga vidinad, näited ning dokumentatsioon, mis muudaksid nende töö paremaks ja lihtsamaks.

2.3 Tööpõhimõte

Leap Motioni täpne viipetundlikkus on võimalik, kuna omavahel on hästi koordineerima saadud kaks vajalikku osa: seade ise ning sellele programmeeritud rakendusliides.

2.3.1 Seadme tööpõhimõte

Leap Motion kontroller koostöös selle rakendusliidesega määrab ära kindlate objektide, nagu näpud, pliiats vms., asukoha kartesiaanlikus[2] ruumis. Edasiantud koordinaadid on omavahel suhtes Leap Motion seadme keskpunktiga, mis asub teise infrapuna anduri juures (vaata Joonis 6). Nagu Joonis 6 näitab on seadmel kaks infrapunakaamerat ning kolm infrapuna andurit, mis tunnevad ära kümne sõrme liikumise ühe sajandiku millimeetri täpsusega. Nii täpse liikumise tuvastamisega on võimalik luua väga täpseid joonistusi arvuti ekraanile. Seega võib Leap Motioni enda tööpõhimõttelt liigitada ruumilisel nägemisel põhinevaks, liikumist jälgivate seadmete hulka (Bachmann, Fisseler, Rudak, & Weichert, 2013).

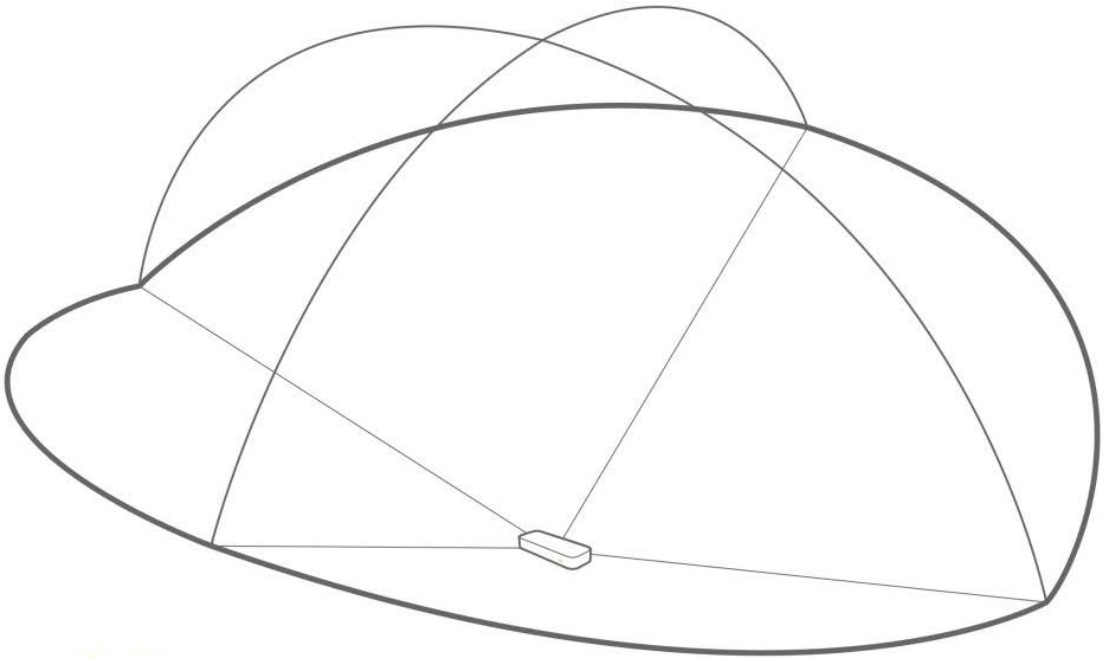


Joonis 6. Leap Motion kontrolleri vaade (a) ilma kaitseklaasita, (b) skemaatiline vaade. (Bachmann, Fisseler, Rudak, & Weichert, 2013)

Seade suudab tuvastada objekte saja viiekümne kraadi raadiuses enda ümber ning saja kahekümne kraadi raadiuses enda kohal (Colgan 2014). Selline suur vaateväli võimaldab kasutajatel liigutada enda käsi 3D keskkonnas. See tähendab, et neil on võimalik Leap Motion äppe kasutades haarata ning liigutada objekte arvutiekraanil justkui oleksid need päriselus.

Seadme laiade skaneerimisnurkade tõttu on sellel suur tuvastusala, mille suuruseks on kaheksa jalga kuubis, moodustades enda ümber tagurpidi püramiidi, mille tipuks on infrapuna kaamerate vaadete ristumiskoht (vaata joonis 7). Leap Motion kontroller suudab tuvastada maksimaalselt 60 sentimeetri kaugusel olevaid objekte. Ulatus on piiratud, kuna seadmes oleva LED pirni valgus on täpselt nii kaugel veel efektiivne, et kaamerad saaksid objekte tuvastada. Muidugi oleks võimalik paigaldada võimsamad LED valgusallikad, aga siinkohal seab piirid ette USB ühenduse kaugu leviv elektrivoolu tugevus (Colgan 2014).

[2] Abstraktne ruum, mida on kirjeldanud prantsuse filosoof René Descartes.



Joonis 7. Leap Motion Seadme tuvastusala. (Colgan 2014)

Skaneerimise käigus loeb USB kontroller sensorist saadud andmed enda kohalikku mällu ning teeb ise vajalikud resolutsiooni kohandused. Seejärel saadetakse andmed USB ühenduse kaudu arvutisse, kus nendega edasi hakkab tegelema Leap Motioni rakendustarkvara (Colgan 2014). Täpseid skeeme ning seletusi, kuidas kontroller õiged punktid ning koordinaadid kätte saab, pole avaldatud.

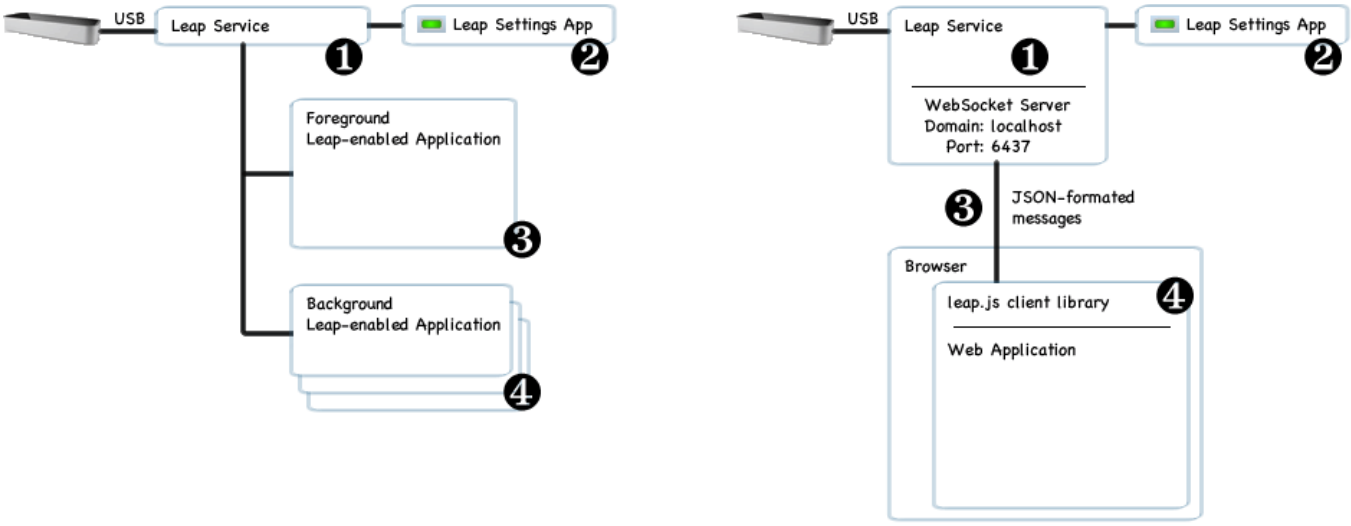
2.3.2 Leap Motion tarkvara

Leap Motion seade ei tee skaneeritavatest objektidest sügavuskaarte, seega, kui seadmest tulnud pildi andmed on arvutisse kantud, hakkab tarkvara tööle kasutades keerulisi matemaatilisi algoritme. Tarkvara, mis hakkab saadud pilti töötlema, kannab nime Leap Motion Service. Algselt üritatakse ära kaotada taustal olnud objektid ning ümbritseva keskkonna valgus. Seejärel analüüsitakse saadud andmeid ning rekonstrueeritakse 3D kujutis sellest, mida seade näeb (Colgan 2014).

Pärast seda vastandatakse liikumist tuvastav Leap Motion Service'i tarkvara osa selle 3D kujutisega, mis eelnevalt on saadud, määrates ära informatsiooni, kus asetsevad sellel kujutisel sõrmed. Liikumist tuvastav algoritm tõlgendab 3D andmed ning nendest järeldab skaneeritud objektide positsiooni. Seejärel kasutatakse filtreerimistehnikaid, et tagada sujuv ning järjepidev andmete kulg. Pärast seda saadab Leap Motion Service saadud tulemused, mis

on kujutatud kaadrite seeriana, milles on kaasas ka liikumiste andmed, transpordi turbeprotokollile (Colgan 2014).

Läbi selle protokolli suhtleb Leap Motion Service nii Leap Motion seadete rakendusega kui ka kohaliku kliendi – ja veebikliendi teekidega kohaliku sokli ühenduse kaudu (vaata joonis 8). Kohalike rakenduste puhul kasutatakse edastusohje protokolli ning veebiteenuste puhul kasutatakse veebisokklit. Klientide teegid kohandavad saadud andmed objekt orienteeritud äpi struktuuri kohaseks, töödeldes kaadrite ajalugu ning varustades neid abifunktsioonide ning klassidega. Sealt alates hakkab tööle juba konkreetse rakenduse loogika, mis võimaldab viipetundlikku interaktiivset kogemust (Colgan 2014).



Joonis 8. Leap Motion - i töö kohalike rakendustega (vasakpoolne pilt) ja veebirakendustega (parempoolne pilt) (Leap Motion Developer Portal).

2.4 Arendamisvõimalused

Programme on võimalik Leap Motionile arendada kuues erinevas keeles. Nendeks on JavaScript, C#, C + +, Java, Python ning Objective – C. Kõikide nende kuue variandi jaoks on olemas täielik dokumentatsioon mis on leitav Leap Motioni kodulehelt.

Leap Motioni arendusteegid on kirjutatud keeles C + +. Leap Motion kasutab ka SWIG – i, mis on avatud lähtekoodiga tööriist, et luua siduvused keeltega C #, Java ja Python. SWIG

tõlgib C + + keeles kirjutatud teegid nendele keeltele mõistetavateks ning seetõttu ei olegi vaja igale keelele erinevaid teeke luua. JavaScripti ja veebirakenduste arendamiseks pakub Leap Motion veebisokliserverit ning kliendipoolset JavaScripti teeki. (Leap Motion Developer).

Rakenduste arendamise jaoks on vajalik paigaldada arvutisse Leap Motioni arendustarkvara. See tarkvara on saadaval nii Windowsi (Windows 7), Macintoshi (OS X 10.7) kui ka Linuxi (Ubuntu 12) poolt pakutavatele operatsioonisüsteemidele. Kõik teegi - , koodi - ning päisefailid, mis on vajalikud, et arendada rakendusi ning pistikprogramme, on kaasatud Leap Motion arendustarkvarasse, välja arvatud leap.js teek, mis on vajalik JavaScriptiga arendamiseks (Leap Motion Developer).

Samuti on olemas erinevaid kompilaatoreid ning integreeritud programmeerimiskeskkondi, millega saab Leap Motion – ile äppe arendada. Nendeks on (Leap Motion Developer):

- Windowsil C + + keelega arendamiseks: Visual Studio 2008, 2010, 2012 ja 2013
- Macintoshil C + + keelega arendamiseks: Xcode3.0+, Clang 3.0+ ja GCC
- Objective – C keelega arendamiseks: Mac OS 10.7+, Xcode 4.2+ ja Clang 3.0+
- C# keelega arendamiseks: .NET Framework versioonid 3.5 ning 4.0
- Mono 2.10
- Unity Pro 5.0 ja Unity Personal 5.0
- Java versioonid 6 ja 7
- Python versioon 2.7.3
- UnrealEngine 4.7

Iga nimetatud kompilaatori või integreeritud programmeerimiskeskkonna jaoks on olemas eraldi teegid, mis aitavad neil suhelda seadmega. Teegid tuleb arendajal Leap Motion Developer keskkonnast leida ning alla laadida.

Kui arendaja on enda loodud äpi valmis saanud, on tal võimalik selle pealt tulu teenida. Selle jaoks on arendatud äppide pood. Sinna on võimalik igal arendajal üles laadida enda loodud rakendus, kus ta saab ise määrata selle hinna, distributsiooni ning millistele operatsioonisüsteemidele on see mõeldud (Developer Portal).

See internetikeskkond on arendatud nii, et äpid on jaotatud kategooriatesse. Igas kategoorias on esireas soovitatud rakendused ning nende vahele on lisatud ka juhuslikult valitud äpid, et igaühe loodud rakendusel oleks võimalik klientidele silma jääda. Kuna igaüks võib selle seadmega luua midagi uut, siis kategooriaid lisatakse juurde, kui selleks vajadus tekib. Suuremad kategooriad on (Developer Portal):

- Arvuti juhtimine
- Loomingulised tööriistad
- Haridus
- Eksperimentaalne
- Mängud
- Muusika ja meelelahutus
- Tootlikus ja utiliidid
- Teadus
- VR Beta

Igaühel, kes omab Leap Motion kontrollerit, on juurdepääs äppide poele, kui nad on allalaadinud Leap Motion tarkvara. See on kõige suurem koht, kus on võimalik rakendusi alla laadida. Kuna enne iga äpi sinna keskkonda lubamist, see kontrollitakse ning antakse heakskiit, on see samuti ka kõige ohutum koht, kust seadmele rakendusi alla laadida (Developer Portal).

Enne rakenduse ülevaatuseks esitamist, peab arendaja läbi lugema ning allkirjastama arendaja levitamislepingu. Samuti tuleb tutvuda ülevaatuse suunitlustega, kus on kirjas kõik, mida

arendaja peaks teadma enne äpi üleslaadimist nagu näiteks millise sisuga rakendused on aktsepteeritavad ning mis võivad saada takistusteks selle avaldamisel. Kui rakendus on keskkonda üles laetud, vaadatakse see üle ning testitakse Leap Motion App Review Teami poolt. Juhul kui see ei läbi teste ning avaldus lükatakse tagasi, saadetakse arendajale detailne testiraport, kus kirjeldatakse, miks äpp ei kiidetud heaks. Seejärel on võimalik vead parandada ning uuesti see testimiseks üles laadida. Kui rakendus vastab nõuetele, laetakse see äppide poodi üles ning arendajal on võimalik selle pealt raha teenima hakata (Developer Portal).

Selle, kui palju rakendus maksma hakkab, valib arendaja ise. On kahte sorti mudeleid, kuidas äppi poes jagada: tasuta või ühekordse tasu eest. Samuti on võimalik ka välja anda proovi-ning täisversioon, et kasutajad saaksid sellega tutvuda ning soovi korral osta. Minimaalne hind, mis on võimalik rakenduse eest küsida on 0.99 dollarit ning maksimaalne hind on 999.99 dollarit. Arendaja saab iga müüdud äpi eest 70 % hinnast ning ülejäänud 30% läheb Leap Motionile (Developer Portal).

3 Uuring Leap Motion seadmest

Bakalaureusetöö osana arendas autor Leap Motion seadmele kaks rakendust. Üks neist on kohalik rakendus, mis arendati keeles C# ning Unity 5 mängumootoriga. Teine on veebipõhine, mis on arendatud kasutades JavaScripti. Veel viis autor läbi uuringu Leap Motion kasutajate seas, et tutvuda, millised on nende kogemused.

3.1 Jenga mängu loomine Unity 5 mängumootoriga

Kuna autor polnud varemat Unityga tegelenud, siis algselt pidi ta sellega end kurssi viima.

Unity on võimas mitmeplatvormiline 3D mootor ning kasutajasõbralik arenduskeskkond. Unity rakendus on 3D keskkond, mis on sobilik erinevate tasemete, menüüde, animatsioonide loomiseks, skriptide kirjutamiseks ning projektide organiseerimiseks. Selle kasutajaliides on hästi organiseeritud ning erinevaid paneele saab kohandada graafiliselt lohistades (Zamojc 2012).

Esialgne õpetus, kuidas Unityga Leap Motionile rakendusi luua, leidis arendaja Leap Motion – i arendajate kodulehelt, aadressil https://developer.leapmotion.com/getting-started/unity.

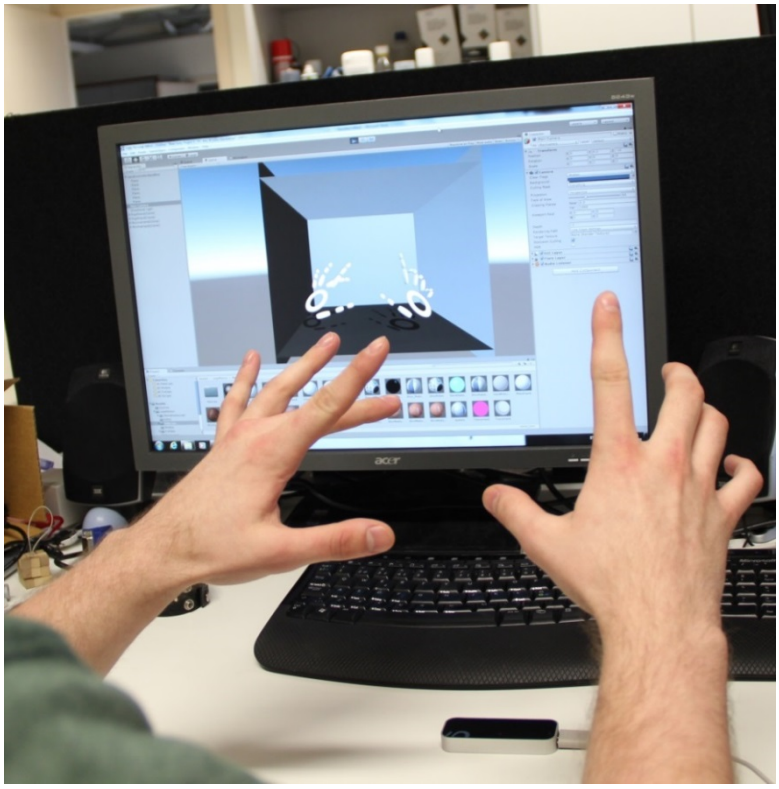
Esmalt tuli laadida alla Unityle loodud südamik varad (*core assets*), kuhu on juba eelnevalt kirja pandud vajalikud erinevad valemid, failid, teegid, mis aitavad Unityl ära tunda Leap Motion seadmelt saabuvad andmed. Selles varade kogumikus on kaasas abipaketid mille ülesanne on lihtsustada arendamist Leap Motion seadmele, Oculus Rift seadmele (mängimiseks mõeldud arvutiga ühendatav pähekäiv seade, mille ülesanne on hägustada virtuaalmaailma ja reaalsuse piire) ning Oculus Rifti ning Leap Motioni kombineeritud variandile.

Esimese asjana tuli luua uus projekt, anda sellele nimi ning valida koht, kuhu projektis valmivad failid salvestatakse. Seejärel avanes Unity keskkond, kuhu olid kaasatud juba eelnevalt nimetatud südamik varad. Kuna autori eesmärk oli arendada lihtne Jenga mäng, siis läks tal vaja Leap Motion seadmele mõeldud varade paketti.

Jenga on mäng, kus mängu alguses laotakse klotsidest torn ning mängu käigus hakatakse sellest üks haaval klotse ära võtma nii, et torn ümber ei kukuks. Täpsemate reeglitega saab ennast kurssi viia aadressil http://www.jenga.com/about.php.

Paketis on juba eelnevalt valmis arendatud näidised, skriptid, stseenid, mida on võimalik enda arenduse aluseks võtta. Samuti on seal olemas erinevad kätemudelid, varjud ning muud kujundamiseks vajalikud failid. Muidugi on võimalik ka kõike nullist alustades ise programmeerima hakata, aga kuna autoril puuduvad nii põhjalikud oskused ja kogemused programmeerimisvaldkonnas, otsustas ta valida valmisdetailidest koosneva aluspõhja enda programmi arendamiseks.

Selleks lohistas ta keskkonda faili nimega HandControlSandBox.prefab. See fail lõi ekraanile kolmemõõtmelise kasti koos virtuaalse Leap Motion seadmega. Kui stseen käivitada ning liigutada käsi arvutiga ühendatud Leap Motion seadme kohal, ilmusid ekraanile virtuaalsed käed, mis käitusid nii, nagu töö autor käsi liigutas (vaata Joonis 9). Kasti seinad olid piirideks, kus sees Leap Motion seade suutis liikumist tuvastada. Kui autor liigutas enda käsi kastist väljas pool, siis seade neid enam ei tuvastanud. Seega oli antud kast suureks abiks mängu loomise käigus, sest nii loodi ette suurused, millest arenduse käigus välja minna ei tohtinud.



Joonis 9 HandControlSandBox.prefab töötamas.

Seejärel hakkas autor lisama keskkonda kolmemõõtmelisi kuubikuid, mille erinevate telgede suurust muutes kujundas ta need selliseks, et need näeksid välja nagu Jenga mängus olevad klotsid. Seejärel tuli igale klotsile määrata ära algne asukoht x, y ja z telje suhtes. Seda tehes pidi autor arvestama algselt paika pandud kasti koordinaatidega, et kuubikud jääksid Leap Motion seadme tuvastavasse piirkonda. Samuti oli vaja lisada neile omadus, mis muutis kuubikud jäikadeks. Selline omadus oli Unity mootoris olemas ning selle rakendamiseks tuli igale objektile lisada komponent *rigidbody*. See andis võimaluse mängu käivitades kätega nende asukohta muuta. Ilma selleta oleksid kuubikud olnud justkui õhus koosnevad objektid. *Rigidbody* komponendis pidi töö autor ära määrama, et objektid alluksid gravitatsioonile. Ilma selleta oleksid objektid küll mängu alguses tekkinud õige Jenga tornina, aga neid mõjutades poleks tulemus olnud enam sama, mis päriselus.

Seejärel lisas autor Jenga torni klotsidele värvid ning kustutas ära arenduse alguses loodud keskkonna seinad, et need ei jääks mängijat klotside eemaldamise juures segama. Seeläbi oli võimalik eemaldatud klotsid Leap Motion seadme tuvastamispiirkonnast välja lükata ning kuna neid pärast tornist eemaldamist enam vaja ei läinud, ei jäänud need ka edasist mängu mõjutama.

Selleks, et lisada mängule funktsionaalsust, lõi autor mängu juurde mängu objekti, kuhu oli võimalik hakata looma erinevaid skripte. Tegevuse alguses sai määrata, kas kirjutatav kood on C# keeles või JavaScript keeles. Selleks, et mõlemad variandid läbi proovida, tegi autor kaks faili. C# jaoks loodi NewBehaviourScript.cs, kuhu pandi kirja koodiread, mis sulgesid mängu vajutades klaviatuuril ESC klahvi ning taaskäivitasid mängu vajutades tühik klahvi (vaata Koodinäide 1). JavaScriptis pandi kirja kood taimeri jaoks (timer.js), mis hakkas alates mängu algusest lugema kolmekümnest sekundist alla poole. Kui taimer jõudis nullini, siis mäng taaskäivitus (vaata Koodinäide 2).

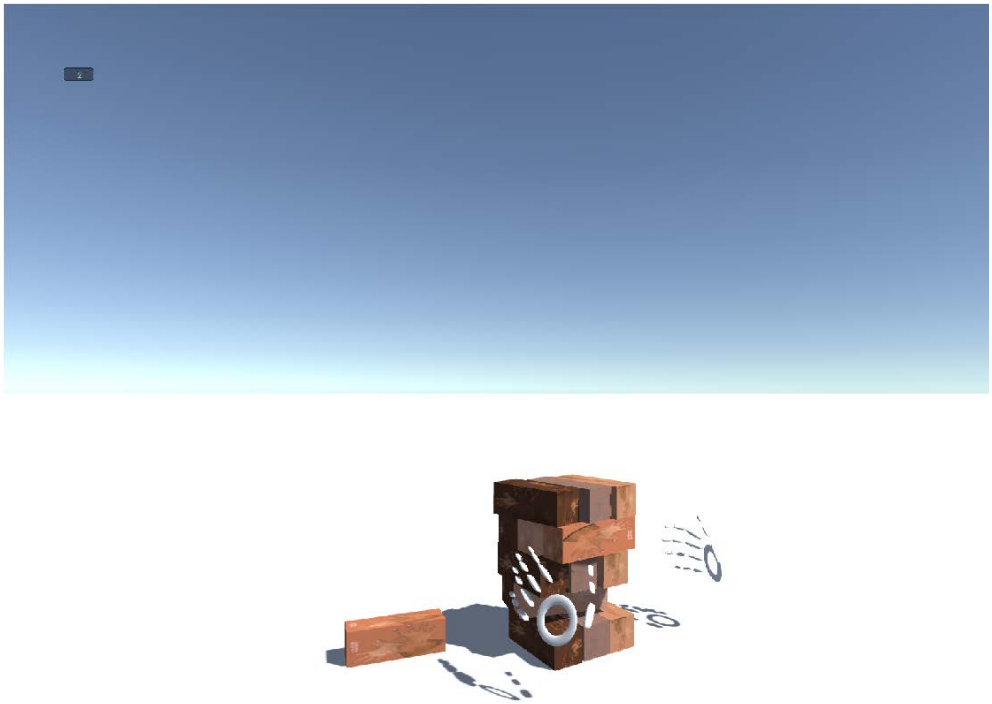
```
void Update(){
        if (Input.GetKey ("escape")) {
                Application.Quit();
        }
        if (Input.GetKeyDown ("space")) {
                Application.LoadLevel (0);
        }
}
```

Koodinäide 1: NewBehaviourScript.cs

```
var timer : float = 30.0;
function Update(){
	timer-= Time.deltaTime;
	if(timer<=0){
		timer=0;
		Application.LoadLevel (0);
	}
}
function OnGUI(){
	GUI.Box(new Rect(100, 100, 50, 20), "" + timer.ToString("0"));
}
```

Koodinäide 2. Timer.js fail.

Kuna mängu objekti luues seoti see kohe ka rakendusega, siis käivitusid need automaatselt ning käsitsi midagi määrama ei pidanud. Sellega oligi autori loodud mäng valmis (vaata joonis 10).



Joonis 60. Autori loodud Jenga mäng Leap Motion seadmele.

Kuna mängu loomisel kaasati projekti automaatselt mitmeid erinevaid funktsioone ning faile, mida autoril vaja ei läinud, siis annab programm mitmeid hoiatusi, aga need ei sega mängul töötamast. Samuti leidis autor, et mängu käivitades antakse veateade, et paketis kaasas olnud

failis nimega LeapRGBUndisorted.shader on real 123 kasutatud valet muutujat. Tegemist on failiga, mis loob mängukeskkonnas varje ning kuna autor antud faili ei redigeerinud ja veateade mängu käivitust ja käitumist ei mõjutanud, siis on ilmselt tegemist tootja poolt välja antud praak failiga.

Unity mootoriga oli äpi loomine autori arvates üpriski lihtne, kuna palju sai ära teha graafilises keskkonnas. Kõige suurem väljakutse oli Unity rakenduse õppimine ning arusaamine kust ning kuidas teatud valikuid leida. Kui programmi käitumisloogika oli juba selge, siis edasi sujus Leap Motion seadmele rakenduse loomine lihtsamalt. Suuresti aitasid kaasa ka südamik varade erinevad valmis objektid, mida autor enne rakenduse loomist läbi katsetas. Pärast nende toimingute tegemist tundus rakenduse enda loomine küllaltki kerge.

Arenduse käigus loodud ning kasutusel olnud faile on võimalik alla laadida aadressilt http://www.cs.tlu.ee/~kristen/leap_motion/unity/Jengaa.

3.2 Veebipõhise rakenduse loomine

Veebipõhise rakenduse loomiseks tutvus autor kõigepealt Leap Motion lehel, aadressil https://developer.leapmotion.com/documentation/javascript/devguide/Sample_Tutorial.html, oleva näidisrakendusega ning seadis selle ise veebikeskkonda üles. Ekraanitõmmis antud näidises on ära näidatud lisas nr 1. See näidis kuvab ekraanile informatsiooni, mille seade tuvastab ning millega saab edasisi arendamisi tegema hakata, kui kasutaja arvutiga ühendatud seadme kohal kätega liigutab. Ära tuntakse eraldi mõlemad käed ning kõik kümme sõrme. Selle rakenduse abil sai autorile selgemaks, milliseid erinevaid andmeid suudab seade tuvastada ning jälgida.

Seejärel leidis autor valmis veebirakenduse koos lähtekoodiga aadressil https://developer.leapmotion.com/getting-started/javascript. Tegemist on rakendusega, kus kuvatakse ekraanile pilt, mis vahetab positsiooni nii nagu seadme kohal kätt liigutatakse. Autor otsustas antud rakenduse järgi luua sarnaselt töötav veebileht.

Kõigepealt lõi autor uue HTML faili, mille päisesse tuli lisada kaks Leap Motioni JavaScript faili (js.leapmotion.com/leap-0.6.4.js ning js.leapmotion.com/leap-0.6.4.js). Sinna on kirja pandud funktsioonid ning muutujad, mille abil saab seade internetipõhiselt töötada. Seejärel tuli luua massiiv, kuhu veebilehe käivitamisel lisatakse vajalikud andmed. Järgmisena oli vaja

luua silmus (*loop*) sellejaoks, et iga millisekundi jooksul, kui seade saadab käte asukohast uue informatsiooni, saab selle järgi muuta kuvatava pildi asukohta (vaata Koodinäide 3).

```
var logoMassiiv = {};
Leap.loop(function(frame) {
        frame.hands.forEach(function(hand, index) {
        var logo = ( logoMassiiv[index] || (logoMassiiv[index] = new TLU()) );
        logo.setTransform(hand.screenPosition(), hand.roll());
        });
}).use('screenPosition', {scale: 0.25});
```

Koodinäide 3. Näidisrakenduse loomine (a).

Kuna programm peab jälgima ainult käe liikumist, siis selle jaoks on klass Hand, millel on omakorda atribuudid screenPosition, mille läbi saab veebileht teada käe koordinaadid, mille seade talle edastab, ning roll, mis määra ära käe kalde nurga. Funktsioon setTransform() paneb ekraanil oleva pildi ette antud väärtuste järgi käituma.

Seejärel lõi autor klassi TLU, kuhu määras ära veebiaadressi, kust kuvatav pilt laetakse ning selle algsed positsioonid. Kõige viimasena lisas ta antud klassi algselt loodud massiivi (vaata Koodinäide 4).

```
var TLU = function() {
        var logo = this;
        var img = document.createElement('img');
        img.src = 'http://www.tlu.ee/UserFiles/Turundus-
        %20ja%20kommunikatsiooniosakond/Logo/TLU-logo-pilt.gif';
        img.style.position = 'absolute';
        img.onload = function () {
                logo.setTransform([window.innerWidth/2,window.innerHeight/2], 0);
                document.body.appendChild(img);
        }
        logo.setTransform = function(position, rotation) {
                img.style.left = position[0] - img.width  / 2 + 'px';
                img.style.top  = position[1] - img.height / 2 + 'px';
                img.style.transform = 'rotate(' + -rotation + 'rad)';
                img.style.webkitTransform = img.style.MozTransform = img.style.msTransform =
                img.style.OTransform = img.style.transform;
        };
};
logoMassiiv[0] = new TLU();
```

Koodinäide 4. Näidisrakenduse loomine (b).

Pärast lehe laadimist kuvati ekraanile Tallinna Ülikooli logo, mille liikumist sai kontrollida arvutiga ühendatud Leap Motion seadme kaudu.

Pärast antud rakenduse käima saamist otsustas autor lisada sellele funktsionaalsust. Selle jaoks leidis ta Leap Motion JavaScripti dokumentatsioonist eelpool nimetatud Hand klassile atribuudi pinchStrength. See atribuut tunneb ära, kui seadme kohal olev käsi puudutab pöidlaga ükskõik millist teist sama käe näppu. Kui käsi on avatud, on selle väärtus 0, mida lähemale pöial näiteks nimetissõrmele liigub, seda suuremaks väärtus kasvab ning kui kaks näppu omavahel kokku puutuvad on selle väärtus 1. Autor otsustas antud atribuuti kasutada ekraanil oleva pildi suurendamiseks ning vähendamiseks.

Selle jaoks lisas autor koodi tsükli, mis tunneb ära, millise näpuga pöial kokku puutub. Idee oli muuta ekraanil olev pilt suuremaks, kui pöial puutub kokku keskmise näpuga ning väiksemaks kui pöial puutub kokku nimetissõrmega.

Kõigepealt loodi tingimuslause, mis tuvastab ära, kas kasutaja pöial on mõnele teisele näpule lähedal, for tsükliga tuvastatakse ära kõik viis sõrme ning tuvastatakse pöidla ning sõrme vaheline kaugus (vaata koodinäide 5).

```
Leap.loop(function(frame) {
   frame.hands.forEach(function(hand, index) {
      if(hand.pinchStrength > 0){
         var napud_koos;
         var closest = 500;
         var zm;
         for(var f = 1; f < 5; f++)    {
             current = hand.fingers[f];
             distance = Leap.vec3.distance(hand.thumb.tipPosition, current.tipPosition);
             if(current != hand.thumb && distance < closest) {
                closest = distance;
                napud_koos = current;
             }
          }
```

Koodinäide 5. Pöidla kauguse tuvastus

Kui näpuks on nimetissõrm, siis omistatakse muutujale zm väärtus 0.9, kui selleks on aga keskmine sõrm, siis 1.1. Muul juhul on zm väärtuseks 1. Zm väärtus lisatakse setTransform atribuudile, et klass TLU saaks seda kasutada (vaata koodinäide 6).

```
        if(napud_koos == hand.indexFinger){
          var logo =( logoMassiiv[index] || (logoMassiiv[index] = new TLU()));
          zm=0.9
          logo.setTransform(hand.screenPosition(), hand.roll(),zm);
        }
        if( napud_koos == hand.middleFinger){
          var logo =( logoMassiiv[index] || (logoMassiiv[index] = new TLU()));
          zm=1.1;
          logo.setTransform(hand.screenPosition(), hand.roll(),zm);
        }
        }else{
          var logo = ( logoMassiiv[index] || (logoMassiiv[index] = new TLU()) );
          zm=1;
          logo.setTransform(hand.screenPosition(), hand.roll(), zm);
        }
    });
}).use('screenPosition', {scale: 0.25});
```

Koodinäide 6. Pildi suurendamiseks ja vähendamiseks kordaja määramine.

Seejärel lisati TLU klassis oleva setTransform atribuudile muutujad pildi kõrguse ning laiuse kohta ning korrutati see zm väärtusega (vaata Koodinäide 7). Terviklik HTML fail on välja toodud lisas nr 2.

```
logo.setTransform = function(positsioon, nurk, zm) {
        wid=img.width;
        ht=img.height;
        img.style.width=(wid*zm)+"px";
        img.style.height=(ht*zm)+"px";
        img.style.left = positsioon[0] - img.width  / 2 + 'px';
        img.style.top  = positsioon[1] - img.height / 2 + 'px';
        img.style.transform = 'rotate(' + -nurk + 'rad)';
        img.style.webkitTransform = img.style.MozTransform = img.style.msTransform =
        img.style.OTransform = img.style.transform;
};
```

Koodinäide 7. JavaScript kood, mis muudab pildi suurust.

Antud koodi käivitades oligi võimalik suurendada pilti, kui pöidlaga puudutada keskmist näppu ning vähendada, kui puudutada nimetissõrme.

Rakenduse käivitamisel võib tekkida olukord, kui seade ajab nimetissõrme ja keskmise sõrme sassi ehk näiteks hakkab pildi vähendama, kui kokku pöial ning keskmine sõrm kokku puutuvad. Sellisel juhul tuleks käsi täielikult avada et pinchStreinght väärtus oleks null ning seejärel uuesti proovida.

Veebirakenduse loomine oli autori arvates küllalti lihtsaks tehtud, sest käe liigutamise jälgimiseks on mitmed atribuudid juba ära määratud ning programmeerija ei pea ise hakkama koordinaatide pealt mingeid lisaarvutusi tegema. Samuti on lihtsaks tehtud näiteks näppude tuvastamine kuna seade tunneb ise ära millise näpuga on tegemist ning need on muutujatena ära määratud. Näiteks kui on vaja jälgida väikese näpu liikumist, siis piisab sellest kui kasutada atribuuti hand.pinky. Samuti on veel dokumentatsioonis mitmeid erinevaid atribuute kätele, näppudele, liigutustele jne. Seetõttu on JavaScriptiga arendamine loodud üpriski lihtsaks.

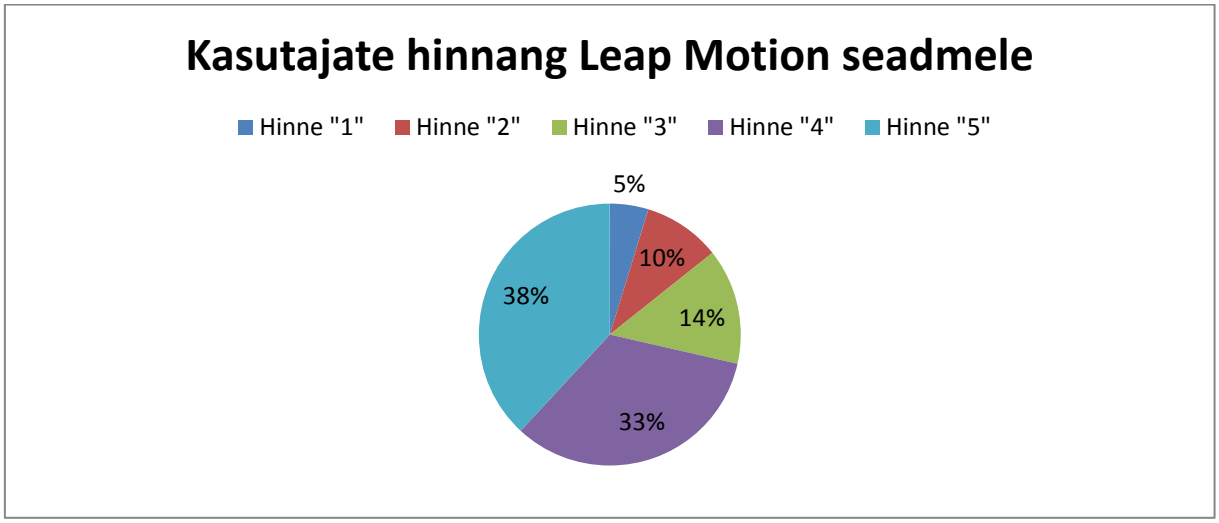
Arenduse käigus loodud veebirakendust on võimalik näha aadressilt http://www.cs.tlu.ee/~kristen/leap_motion/JS/rakendus.html.

3.3 Kasutajate kogemus

Autor koostas küsimustiku, millele palus Leap Motioni kasutajatel vastata. Kuna vastata said ainult need, kes on Leap Motion seadet kasutanud, siis ei oodanud autor väga suurt tagasisidet sellele. Autor jagas küsimustikku erinevates Leap Motioniga seotud foorumites, interneti kommuunides ning samuti lasi ta inimestel proovida enda isiklikku seadet ning hiljem palus neil vastata küsimustikule. Kuna eeldatavad vastajad olid enamjaolt välismaalased, siis otsustas autor küsimustiku koostada inglise keeles. Küsimustik on näidatud lisas nr 3.

Vastanuid oli 21, kellest enamik olid mehed vanuses 18 – 25. 26 % küsitlusele vastanutest on seadet kasutanud ainult mõned korrad, ehk tegemist on inimestega, kes ise ei oma seadet ning on seda ainult proovinud. 19 % vastanutest oli neid, kes on Leap Motionit regulaarselt kasutanud alla kuu ning samapalju on ka neid, kes on seda teinud 1,5 – 2 aastat.

Küsitluses palus autor vastajatel hinnata oma kogemust Leap Motion seadmega viie palli süsteemis, kus 1 tähendas, et kogemus oli halb ning 5, et kogemus oli hea. Kõikide vastajate keskmine hinne antud seadmele oli 3,9. Hinnete kujunemist iseloomustab joonis 11.



Joonis 11. Kasutajate hinnang Leap Motion seadmele

Numbriga 5 hinnanud kasutajad tõid põhjenduseks välja seadme täpsuse, ning selle, et teist taolist sensorit, mis jälgib inimese käsi, turul ei ole. 2 inimest kiitsid seadme kasutamise mugavus 3D modelleerimisel. Hinnetega 1 ja 2 hinnanud kasutajad midagi täpset välja ei toonud, miks seade neile ei meeldinud. Sellest oodati midagi muud, või lihtsalt ei meeldinud seda kasutada. Teisel poolt kõik need kolm vastajat olid seadet kasutanud ainult mõned korrad, seega on nende hinnang põhinenud ainult esmamuljel. Veel toodi välja, et seade töötab hetkel, pärast uuenduskuuri, paremini kui 1,5 aastat tagasi, kui see esmakordselt turule tuli, aga kasutajatele võiks olla rohkem rakendusi saadaval.

Küsimusele, mis vastajale seadme juures meeldib, toodi kõige enam välja sensori väikseid mõõte, mobiilsust ja mitmekülgsust. Paljudele meeldis mõte, et on arendatud seade, mis on kergesti seadistatav ning võimaldab inimestel käte abil arvutikeskkonnas ringi liikuda. Teiselt poolt toodi välja, et sensoril võiks olla rohkem kasutajaid ning rakendusi. Samuti mainiti, et sensori tuvastusala võiks olla suurem.

Kõikidest vastanutest kaheksa on ka sellele seadmele rakendusi arendanud. Enda kogemust kommenteerides toodi välja, et programmeerimine Leap Motion seadme jaoks sarnaneb paljuski arendamisele Microsoft Kinecti jaoks. Kui seadme tööpõhimõte on selge ning arendaja teab, milliseid andmeid on tal võimalik kasutada, siis ei ole arendamine kuidagi

raskendatud. Veel mainiti, et arendamine esimese tarkvaraversiooniga oli palju keerulisem ning seade ei andnud nii täpseid ja stabiilseid andmeid kui seda teeb V2.

Kuna David Holz arendas selle seadme välja, kuna pettus arvutihiires ning klaviatuuris, siis küsis töö autor enda koostatud küsitluses ka kasutajate arvamust, kas Leap Motion võiks olla alternatiiviks neile kahele. Kahekümne ühest vastajast kolmteist arvas, et see sensor seda olla ei saaks. Põhjusteks toodi välja et seadme kasutamine on väsitav ning inimene ei jõuaks kaheksa tundi päevas sellega töötada. Samuti toodi välja hiire mugavust ning mainiti, et Leap Motion töötaks nii – öelda abivahendina väga edukalt. Üks vastaja arvas, et klaviatuuri ta ei asendaks, aga hiirt mõningate toimingute puhul kindlasti.

Küsimusele, mida tuleb Leap Motioni puhul täiustama, tõid paljud vastajad välja, et sellega töötamine võiks olla täpsem ning inimesele lihtsam ja mugavam. Samuti sooviti suuremat tuvastusala, aga mainiti ka ära, et kõik oleneb sellest, mida kasutaja selle seadmega teha soovib. Näiteks üks vastaja, kes tegeleb igapäevaselt 3D modelleerimise, väitis, et tema jaoks töötab see väga hästi ning midagi muuta pole vaja.

Kokkuvõttes võib öelda, et enamus vastanutest on seadme tööga rahul. Kuna tegemist on alles lapsekingades oleva sensoriga, siis paelub see inimesi just enda uudse lähenemise tõttu. Küsitlusest tuli välja, et inimesed, kes on seadet rohkem kasutanud, on sellega ka rahule jäänud. Väiksema kasutamiskogemusega inimeste seas oli nii neid, kes sellest vaimustuses olid ning samas ka neid kellele see üldse ei meeldinud.

Kokkuvõte

Bakalaureusetöö eesmärk oli tutvustada Leap Motion seadet, selle ajalugu, tööpõhimõtet ning arendada kaks näidisrakendust, et uurida, kuidas nende loomine käib. Samuti oli eesmärgiks koguda andmeid kasutajate kogemusest.

Autor arendas kaks rakendust, et uurida kuidas käib veebipõhiste ning töölaua keskkonnas käivitavate äppide loomine. Unity 3D mängumootoriga oli rakenduse loomine küllaltki lihtne. Selleks tuli koos programmiga käivitada ka Leap Motioni poolt loodud südamik varad, mis sisaldas endas mitmeid algfaile, mis lihtsustasid oluliselt arendusprotsessi. Samuti oli palju abi ka Unity keskkonnast, sest seal on võimalik lisada objektidele varasemalt kirja pandud füüsika seaduseid, mis muutsid rakenduse reaalsemaks ning loomulikumaks.

Veebipõhise rakenduse loomisel tuli kõigepealt lisada kaks JavaScript faili, mille abil oli võimalik kasutada erinevaid funktsioone ning atribuute. Need atribuudid lihtsustavad oluliselt arendaja tööd, sest seal on kirja pandud erinevad muutujad ning isegi kindlad liigutused (nagu näiteks autori poolt kasutatud pinchStreingth), mida saab kasutada erinevate käskude korral. Kuna dokumentatsioon on küllaltki mahukas ning arusaadavalt koostatud, siis on erinevate rakenduste loomiseks sellest palju abi.

Kasutajate tagasiside oli küllaltki positiivne. Kuna seade pole veel piisavalt populaarne ning vastata said ainult need, kes reaalselt on seda kontrollerit kasutanud, siis oli vastajate leidmine raskendatud. Autor palus kasutajatel hinnata oma kogemust Leap Motion seadmega viie palli süsteemis, kus 1 tähendas, et kogemus oli halb ning 5 tähendas, et kogemus oli hea. Keskmiseks hindeks sai seade 3,9. Vastustest tuli välja, et enamus suhtus Leap Motion seadmesse positiivselt, kuigi sooviksid, et see töötaks pisut loomulikumalt ning sujuvamalt.

Autorile pakkus töö huvitavat väljakutset. Kuna Leap Motion seade on küllaltki uus ning Eestis suhteliselt tundmatu, siis erinevaid valdkondi, mida saaks edaspidi uurida, on mitmeid. Käesolev bakalaureusetöö keskendus Leap Motionile üldisemalt, tutvustades lühidalt erinevaid seadme omadusi. Edasiarenduseks oleks võimalik testida selle seadme täpsust ning robustsust. Kuna autor tutvustas ainult kahte moodust, kuidas saab sensorile rakendusi arendada, siis võiks seda teha ka teiste võimaluste puhul.

Kasutatud kirjandus

Ambrosch, K. , Kubinger, W. (2010). Accurate hardware-based stereo vision. Computer Vision and Image Understanding, 114(11), 1303 – 1316.

Bachmann, D. , Fisseler, D. , Rudak, B. , Weichert, F. (2013). Analysis of the Accuracy and Robustness of the Leap Motion Controller. Sensors, 13(5), 6380 – 6393. doi:10.3390/s130506380

Bowerman, E. (2012). Leap Motion Announces $12.75 Million Series A Funding Round Led by Highland Capital Partners. Loetud aadressil http://www.marketwired.com/press-release/leap-motion-announces-1275-million-series-a-funding-round-led-highland-capital-partners-1654737.htm.

Calin, D. (2013). Fundamental Guide for Stereo Vision Cameras in Robotics – Tutorials and Resources. Loetud aadressil http://www.intorobotics.com/fundamental-guide-for-stereo-vision-cameras-in-robotics-tutorials-and-resources/.

Colgan, A. (2014). How Does the Leap Motion Controller Work?. Loetud aadressil http://blog.leapmotion.com/hardware-to-software-how-does-the-leap-motion-controller-work.

Developer Portal. (kuupäev puudub). The Leap Motion App Store. Loetud kuupäeval 08. aprill 2015 aadressil https://developer.leapmotion.com/apps/app-store.

Etherington, D.(2013). Leap Motion Controller Ship Date Delayed Until July 22, Due To A Need For A Larger, Longer Beta Test. Loetud aadressil http://techcrunch.com/2013/04/25/leap-motion-controller-ship-date-delayed-until-july-22-due-to-a-need-for-a-larger-longer-beta-test.

Etherington, D. (2014). Leap Motion Lays Off 10% Of Its Workforce After Missing On First Year Sales Estimates. Loetud aadressil http://techcrunch.com/2014/03/20/leap-motion-lays-off-10-of-its-workforce-after-missing-on-first-year-sales-estimates.

Foster, T. (2013). Will These Guys Kill the Computer Interface as We Know It? Loetud aadressil http://www.popsci.com/technology/article/2013-07/will-these-guys-kill-computer-interface-we-know-it

Geng, J.(2010). Structured-light 3D surface imaging: a tutorial. Advances in Optics and Photonics. 3(2), 128 – 160.

Jaroslavsky, R. (2013). Leap Motion's magic bubble gives you PC control with wave of hand. Loetud aadressil http://www.seattletimes.com/business/leap-motionrsquos-magic-bubble-gives-you-pc-control-with-wave-of-hand.

Kosner, A. (2013). Leap Motion Announces $30M Series B Funding And Bundle Deal With ASUS Computers. Loetud aadressil http://www.forbes.com/sites/anthonykosner/2013/01/03/leap-motion-announces-30m-series-b-funding-and-bundle-deal-with-asus-computers/.

Leap Motion. (kuupäev puudub). Technical Specifications. Loetud kuupäeval 02. aprill 2015 aadressil https://www.leapmotion.com/product.

Leap Motion Developer. (kuupäev puudub). SDK Libraries. Loetud kuupäeval 08. aprill 2015 aadressil https://developer.leapmotion.com/documentation/csharp/devguide/Leap_SDK_Overview.html.

Leap Motion Developer Portal (kuupäev puudub). System Architecture. Loetud aadressil https://developer.leapmotion.com/documentation/csharp/devguide/Leap_Architecture.html.

Li, J. , Liu, X., Miao, Z., Wan, Y. (2010). 3D Reconstruction Based on Stereo Vision and Texture Mapping. Loetud aadressil http://www.isprs.org/proceedings/XXXVIII/part3/b/pdf/1_XXXVIII-part3B.pdf.

Li, L.(2014). Time-of-Flight Camera – An Introduction. Loetud aadressil http://www.ti.com/lit/wp/sloa190b/sloa190b.pdf.

Moynihan, T. (2014). Leap Motion Adds More Intrigate Tracking To Its Amazing Controller. Loetud aadressil http://www.wired.com/2014/05/leap-motion-adds-more-intricate-tracking-to-its-amazing-controller.

Panzarino, M. (2012). Leap Motion launches Software Developer Program and starts sending test units of its 3D controller. Loetud aadressil http://thenextweb.com/apple/2012/10/29/leap-motion-launches-software-developer-program-and-starts-sending-test-units-of-its-3d-controller.

Pcmag. (2015). Leap Motion Controller. Loetud aadressil http://www2.pcmag.com/media/images/393268-leap-motion-controller.jpg?thumb=y.

Richardson, N., M. (2013). One Giant Leap for Mankind. Loetud aadressil http://www.inc.com/30under30/nicole-marie-richardson/leap-motion-david-holz-michael-buckwald-2013.html

Silberman, N. ,& Fergus, R. (2012). Indoor Scene Segmentation using a Structured Light Sensor. Loetud aadressil http://www.cs.nyu.edu/~silberman/papers/indoor_seg_struct_light.pdf.

Zamojc, I. (2012). Introduction to Unity 3D. Loetud aadressil http://code.tutsplus.com/tutorials/introduction-to-unity3d--mobile-10752.

Tsotsis, A.(2011). OcuSpec Raises 1.3M From Andreessen And Others To Build An "Affordable Kinect". Loetud aadressil http://techcrunch.com/2011/06/10/ocuspec-raises-1-3m-from-andreessen-and-others-to-build-an-affordable-kinect/.

Vallaste. (2015). Loetud kuupäeval 25. aprill 2015 aadressil http://www.vallaste.ee/.

Summary

Leap Motion: The New Generation 3D Sensor

The objective of this bachelor thesis is to introduce the Leap Motion device, its history, how does it work and develop two applications to explore, how the creation of them works. In order to try out different ways of development, author created one web application, using JavaScript programming language and one native application, using Unity game engine and C# programming language. The author chose these methods of development, because he had experience with those languages. Another aim was to investigate the users experience of working with this device. For that, the author created a survey and shared it on various Leap Motion websites and communities.

Bachelor thesis consists of three chapters, which in turn are divided into subchapters. In the first chapter, the author introduced different 3D sensors and their working methods. In the second chapter, the author introduced the device, its history, how it works and also presented, how developers can develop apps for the controller. In third chapter the author describes how the two application were created and analyze user experience.

Writing this thesis offered interesting challenge for the author. Since Leap Motion device is fairly new and relatively unknown in Estonia, then a variety of areas that could be further explored, are several. This study focused on the Leap Motion more generally and introduced the various features of the device. Further development may contain an analysation of the accuracy and robustness of the Leap Motion controller. As the author presents only two ways in which you can develop applications in the sensor, then this could be done with other options as well.

Lisad

Lisa 1. JavaScript näidisrakendus.

Leap Motion JavaScript Sample

Pause ☑ Pause on gesture

Frame data:

Frame ID: 589143
Timestamp: 42263453011 µs
Hands: 2
Fingers: 10
Tools: 0
Gestures: 1
Translation: (0.3, -0.7, 0.1) mm
Rotation axis: (0.47, 0.06, 0.88)
Rotation angle: 0.00 radians
Scale factor: 0.99

Hand data:

Hand ID: 494
Type: left hand
Direction: (0.18, 0.50, -0.85)
Palm position: (-142.0, 134.8, 46.5) mm
Grab strength: 0
Pinch strength: 0
Confidence: 0.745842
Arm direction: (-0.4, 0.9, -0.0)
Arm center: (-105.0, -19.3, 91.7)
Arm up vector: (-0.6, -0.2, 0.8)
Translation: (0.3, -0.2, -0.0) mm
Rotation axis: (0.8, 0.5, -0.1)
Rotation angle: 0.01 radians
Scale factor: 1.00
Fingers IDs: 4940, 4941, 4942, 4943, 4944

Hand ID: 495
Type: right hand
Direction: (-0.07, 0.50, -0.87)
Palm position: (129.2, 134.0, 31.4) mm
Grab strength: 0
Pinch strength: 0
Confidence: 0.723009
Arm direction: (-0.7, -0.1, -0.7)
Arm center: (237.6, 123.2, 168.1)
Arm up vector: (0.4, 0.7, -0.6)
Translation: (-1.2, -1.2, 0.3) mm
Rotation axis: (-0.1, -0.1, 1.0)
Rotation angle: 0.03 radians
Scale factor: 1.00
Fingers IDs: 4950, 4951, 4952, 4953, 4954

Finger and tool data:

Pointable ID: 4940
Type: Thumb
Belongs to hand with ID: 494
Classified as a finger
Length: 52.1 mm
Width: 20.3 mm
Direction: (0.74, 0.56, -0.37)
Extended?: true
Metacarpal bone
Center: (-137.7, 118.5, 105.6)
Direction: (0.4, 0.8, -0.5)
Up vector: (-0.1, 0.6, 0.8)
Proximal phalanx bone
Center: (-119.2, 133.0, 97.0)
Direction: (0.7, 0.6, -0.3)
Up vector: (-0.1, 0.6, 0.8)
Intermediate phalanx bone
Center: (-88.1, 156.9, 82.1)
Direction: (0.7, 0.6, -0.4)
Up vector: (-0.1, 0.6, 0.8)
Distal phalanx bone
Center: (-66.7, 171.9, 70.2)
Direction: (0.7, 0.5, -0.5)
Up vector: (0.0, 0.7, 0.7)
Tip position: (-62.1, 174.6, 66.6) mm

Pointable ID: 4941
Type: Index finger
Belongs to hand with ID: 494
Classified as a finger
Length: 58.8 mm
Width: 19.3 mm
Direction: (0.21, 0.52, -0.83)
Extended?: true
Metacarpal bone
Center: (-142.4, 146.8, 66.3)
Direction: (0.4, 0.5, -0.8)
Up vector: (-0.8, 0.6, -0.0)
Proximal phalanx bone
Center: (-130.5, 180.7, 22.5)
Direction: (-0.1, 0.7, -0.7)
Up vector: (-0.9, 0.3, 0.4)
Intermediate phalanx bone
Center: (-130.1, 201.1, -3.5)
Direction: (0.2, 0.5, -0.8)
Up vector: (-0.9, 0.5, 0.1)
Distal phalanx bone
Center: (-123.5, 210.3, -20.5)
Direction: (0.5, 0.3, -0.8)
Up vector: (-0.8, 0.6, -0.2)
Tip position: (-122.1, 212.6, -23.8) mm

Pointable ID: 4942
Type: Middle finger
Belongs to hand with ID: 494
Classified as a finger
Length: 67.0 mm
Width: 19.0 mm
Direction: (0.39, 0.24, -0.89)
Extended?: true
Metacarpal bone
Center: (-151.2, 136.0, 56.3)
Direction: (0.3, 0.4, -0.8)
Up vector: (-0.9, 0.4, -0.1)
Proximal phalanx bone
Center: (-140.5, 161.7, 5.3)
Direction: (-0.0, 0.4, -0.9)
Up vector: (-0.9, 0.3, 0.2)
Intermediate phalanx bone
Center: (-135.7, 175.3, -29.2)
Direction: (0.4, 0.2, -0.9)
Up vector: (-0.9, 0.5, -0.2)
Distal phalanx bone
Center: (-123.7, 179.2, -48.6)
Direction: (0.7, 0.0, -0.7)
Up vector: (-0.6, 0.5, -0.6)
Tip position: (-121.8, 180.3, -52.4) mm

Pointable ID: 4943
Type: Ring finger
Belongs to hand with ID: 494
Classified as a finger
Length: 64.4 mm
Width: 18.1 mm
Direction: (0.30, 0.26, -0.92)
Extended?: true
Metacarpal bone
Center: (-158.6, 122.0, 49.6)
Direction: (0.3, 0.3, -0.9)
Up vector: (-0.9, 0.4, -0.1)
Proximal phalanx bone
Center: (-153.3, 141.2, 0.8)
Direction: (-0.1, 0.4, -0.9)
Up vector: (-0.9, 0.2, 0.2)
Intermediate phalanx bone
Center: (-152.0, 153.7, -32.2)
Direction: (0.3, 0.3, -0.9)
Up vector: (-0.9, 0.4, -0.2)
Distal phalanx bone
Center: (-142.0, 158.3, -52.1)
Direction: (0.6, 0.1, -0.8)
Up vector: (-0.7, 0.4, -0.5)
Tip position: (-139.6, 158.8, -56.1) mm

Pointable ID: 4944
Type: Pinky finger
Belongs to hand with ID: 494
Classified as a finger
Length: 50.5 mm
Width: 16.1 mm
Direction: (0.26, -0.02, -0.97)
Extended?: true
Metacarpal bone
Center: (-161.2, 105.2, 45.4)
Direction: (0.2, 0.2, -1.0)
Up vector: (-1.0, 0.2, -0.1)
Proximal phalanx bone
Center: (-159.8, 113.2, 0.2)
Direction: (-0.2, 0.1, -1.0)
Up vector: (-1.0, 0.2, 0.2)
Intermediate phalanx bone
Center: (-160.4, 114.2, -26.6)
Direction: (0.3, -0.0, -1.0)
Up vector: (-1.0, 0.2, -0.3)
Distal phalanx bone
Center: (-152.7, 113.3, -42.9)
Direction: (0.6, -0.1, -0.8)
Up vector: (-0.8, 0.2, -0.6)
Tip position: (-150.1, 112.5, -46.6) mm

Pointable ID: 4950
Type: Thumb
Belongs to hand with ID: 495
Classified as a finger
Length: 52.1 mm
Width: 20.3 mm
Direction: (-0.71, 0.70, -0.14)
Extended?: true
Metacarpal bone
Center: (115.3, 116.2, 88.5)
Direction: (-0.3, 0.7, -0.6)
Up vector: (-0.1, 0.6, 0.8)
Proximal phalanx bone
Center: (97.4, 132.7, 83.1)
Direction: (-0.7, 0.7, -0.2)
Up vector: (0.1, 0.4, 0.9)
Intermediate phalanx bone
Center: (67.5, 161.1, 75.4)
Direction: (-0.7, 0.7, -0.1)
Up vector: (0.2, 0.4, 0.9)
Distal phalanx bone
Center: (47.5, 181.6, 72.7)
Direction: (-0.7, 0.7, -0.0)
Up vector: (0.3, 0.3, 0.9)
Tip position: (43.2, 185.9, 72.0) mm

Pointable ID: 4951
Type: Index finger
Belongs to hand with ID: 495
Classified as a finger
Length: 58.8 mm
Width: 19.3 mm
Direction: (-0.04, 0.57, -0.82)
Extended?: true
Metacarpal bone
Center: (124.7, 145.6, 50.9)
Direction: (-0.3, 0.5, -0.8)
Up vector: (0.8, 0.6, 0.1)
Proximal phalanx bone
Center: (117.1, 178.8, 5.3)
Direction: (0.1, 0.7, -0.7)
Up vector: (0.8, 0.4, 0.5)
Intermediate phalanx bone
Center: (119.4, 200.0, -20.4)
Direction: (-0.0, 0.6, -0.8)
Up vector: (0.8, 0.5, 0.3)
Distal phalanx bone
Center: (117.4, 210.9, -37.7)
Direction: (-0.2, 0.5, -0.9)
Up vector: (0.8, 0.6, 0.2)
Tip position: (114.8, 212.8, -41.7) mm

Pointable ID: 4952
Type: Middle finger
Belongs to hand with ID: 495
Classified as a finger
Length: 67.0 mm
Width: 19.0 mm
Direction: (-0.48, 0.05, -0.87)
Extended?: true
Metacarpal bone
Center: (136.2, 135.9, 42.7)
Direction: (-0.2, 0.4, -0.9)
Up vector: (0.9, 0.5, 0.1)
Proximal phalanx bone
Center: (126.2, 158.1, -10.8)
Direction: (-0.1, 0.3, -0.9)
Up vector: (0.9, 0.5, 0.0)
Intermediate phalanx bone
Center: (116.3, 166.1, -46.0)
Direction: (-0.5, 0.1, -0.9)
Up vector: (0.7, 0.6, -0.4)
Distal phalanx bone
Center: (102.9, 165.4, -65.0)
Direction: (-0.7, -0.2, -0.7)
Up vector: (0.5, 0.6, -0.6)
Tip position: (100.7, 165.6, -68.8) mm

Pointable ID: 4953
Type: Ring finger
Belongs to hand with ID: 495
Classified as a finger
Length: 64.4 mm
Width: 18.1 mm
Direction: (-0.33, 0.09, -0.94)
Extended?: true
Metacarpal bone
Center: (146.1, 122.7, 37.6)
Direction: (-0.1, 0.3, -0.9)
Up vector: (0.9, 0.4, 0.0)
Proximal phalanx bone
Center: (143.3, 138.9, -13.3)
Direction: (0.0, 0.3, -1.0)
Up vector: (0.9, 0.4, 0.1)
Intermediate phalanx bone
Center: (139.5, 146.2, -47.9)
Direction: (-0.3, 0.1, -0.9)
Up vector: (0.8, 0.5, -0.2)
Distal phalanx bone
Center: (129.6, 146.7, -68.5)
Direction: (-0.6, -0.1, -0.8)
Up vector: (0.7, 0.5, -0.5)
Tip position: (127.7, 146.5, -72.7) mm

Pointable ID: 4954
Type: Pinky finger
Belongs to hand with ID: 495
Classified as a finger
Length: 50.5 mm
Width: 16.1 mm
Direction: (-0.07, -0.23, -0.97)
Extended?: true
Metacarpal bone
Center: (151.1, 106.4, 33.9)
Direction: (0.0, 0.2, -1.0)
Up vector: (1.0, 0.3, 0.1)
Proximal phalanx bone
Center: (156.1, 111.1, -11.0)
Direction: (0.3, -0.1, -1.0)
Up vector: (0.9, 0.3, 0.2)
Intermediate phalanx bone
Center: (160.3, 106.5, -37.4)
Direction: (-0.1, -0.2, -1.0)
Up vector: (1.0, 0.2, -0.1)
Distal phalanx bone
Center: (156.7, 101.8, -54.6)
Direction: (-0.3, -0.3, -0.9)
Up vector: (0.9, 0.2, -0.4)
Tip position: (155.6, 100.5, -58.6) mm

Gesture data:

Gesture ID: 1154, type: circle, state: start, hand IDs: 495, pointable IDs: 4950, duration: 0 µs, center: (52.8, 192.5, 66.9) mm, normal: (0.68, -0.41, 0.60), radius: 10.5 mm, progress: 0.80 rotations

Lisa 2. Rakendus.html terviklik fail

```
<!doctype html>
<html lang="en">
<head>
        <meta charset="UTF-8">
        <title>Leap Motion rakendus</title>
        <script src="//js.leapmotion.com/leap-0.6.0.js"></script>
<script src="//js.leapmotion.com/leap-plugins-0.1.6.js"></script>
<script>
var logoMassiiv = {};

Leap.loop(function(frame) {

    frame.hands.forEach(function(hand, index) {
        if(hand.pinchStrength > 0){
        var napud_koos;
    var closest = 500;
        var zm;
    for(var f = 1; f < 5; f++)
    {
        current = hand.fingers[f];
        distance = Leap.vec3.distance(hand.thumb.tipPosition,
current.tipPosition);
        if(current != hand.thumb && distance < closest)
        {
            closest = distance;
            napud_koos = current;
        }
    }
   if(napud_koos == hand.indexFinger){
       var logo =( logoMassiiv[index] || (logoMassiiv[index] = new TLU()));
               zm=0.9
           logo.setTransform(hand.screenPosition(), hand.roll(),zm);
        }
         if( napud_koos == hand.middleFinger){
       var logo =( logoMassiiv[index] || (logoMassiiv[index] = new TLU()));
               zm=1.1;
        logo.setTransform(hand.screenPosition(), hand.roll(),zm);
    }

  }else{
    var logo = ( logoMassiiv[index] || (logoMassiiv[index] = new TLU()) );
        zm=1;
    logo.setTransform(hand.screenPosition(), hand.roll(), zm);
        }
  });

}).use('screenPosition', {scale: 0.25});
```

```
var TLU = function() {
  var logo = this;
  var img = document.createElement('img');
  img.src = 'http://www.tlu.ee/UserFiles/Turundus-
%20ja%20kommunikatsiooniosakond/Logo/TLU-logo-pilt.gif';
  img.style.position = 'absolute';
  img.onload = function () {
    logo.setTransform([window.innerWidth/2,window.innerHeight/2], 0);
    document.body.appendChild(img);
  }
  logo.setTransform = function(positsioon, nurk, zm) {
        wid=img.width;
        ht=img.height;
        img.style.width=(wid*zm)+"px";
        img.style.height=(ht*zm)+"px";
    img.style.left = positsioon[0] - img.width  / 2 + 'px';
    img.style.top  = positsioon[1] - img.height / 2 + 'px';

        img.style.transform = 'rotate(' + -nurk + 'rad)';

    img.style.webkitTransform = img.style.MozTransform =
img.style.msTransform =
    img.style.OTransform = img.style.transform;

  };

};
logoMassiiv[0] = new TLU();

</script>
</head>
<body>

</body>
</html>
```

Lisa 3. Küsitluse ankeet.

Survey about Leap Motion user experience

Hello! My name is Kristen Kivimaa and I'm an Informatics student in Tallinn University. I'm writing my Bachelor's thesis about Leap Motion. I would be greateful if You can answer some questions about your experience with Leap Motion device.

* Kohustuslik

What is Your gender? *

- Male
- Female

Your age.. *

- Under 18
- 18 - 25
- 26 - 30
- 31 - 35
- 35 - 40
- 41 - 50
- Over 50

How long have You been using Leap Motion device? *

- I've just tried it couple of times
- I've used it less than a month
- I've used it for 1 - 6 months
- I've used it for 6 - 12 months
- I've used it for 1 - 1.5 years
- I've used it for 1.5 - 2 years
- I've used it over 2 years

What for do You use The Leap Motion device? *

- For work
- For developing
- For studing
- For entertainment
- For gaming
- Muu:

How many Leap Motion apps have You used? *

- 1 - 5
- 6 - 10
- 11 - 15
- 16 - 20
- Over 20

Where did You hear about Leap Motion? *

- From friends
- From the news
- From reseller
- From the Internet
- Muu:

In scale 1 to 5, rate Your experience with the device? *

	1	2	3	4	5	
Bad						Good

Justify Your previous answer *

Have You tried to develope some apps for the device? *

- Yes
- No

If you answered yes for the previous question, thed describe Your experience.

What do You like/dislike about the device? *

Would you recommend the device for your friends? *

- Yes
- No

Justify Your previous answer. *

Leap Motion creator, David Holz, started to develot the device because he was frustrated by the limitations of the mouse and keyboard. Do You think Leap Motion can replace the mouse and keyboard? *

- Yes
- No

Justify Your previous answer. *

What should be improved about the device? *

If You have something to add, You can write it here.

Saada ära

100%: ongi valmis.

Ärge saatke paroole kunagi Google'i vormide kaudu.