LAPPEENRANNAN-LAHDEN TEKNILLINEN YLIOPISTO
School of Engineering Science
Laskennallisen tekniikan koulutusohjelma
Kandidaatintyö

Teijo Pekkola

Askelmittaus puhelimen kiihtyvyys- ja gyroskooppidatasta

Ohjaaja: Arto Kaarna

TIIVISTELMÄ

Lappeenrannan-Lahden teknillinen yliopisto

School of Engineering Science

Laskennallisen tekniikan koulutusohjelma

Teijo Pekkola

Askelmittaus puhelimen kiihtyvyys- ja gyroskooppidatasta

Kandidaatintyö

2020

27 sivua, 13 kuvaa, 4 taulukkoa

Ohjaaja: Arto Kaarna

Avainsanat: askelmittaus; askelmittari; jaksontunnistus; Android;

Työn tavoitteena oli luoda askelmittarisovellus Android-puhelimelle. Sovelluksen oli tarkoitus esittää käyttäjälle enemmän informaatiota urheilusuorituksesta kuin tavallinen askelmittari, joka usein ilmoittaa ainoastaan kokonaisaskelmäärän.

Työssä tutkittiin erilaisia kohinanpoistomenetelmiä ja jakson tunnistusta puhelimen mittaamasta kiihtyvyys- ja gyroskooppidatasta. Näitä menetelmiä käyttäen luotiin Android Studio -kehitysympäristöllä sovellus, jolla voidaan mitata ja analysoida puhelimen liikkeitä liikuntasuorituksen ajalta. Sovellus toimii kahdessa vaiheessa. Liikuntasuorituksen aikana puhelimen kokemat kiihtyvyydet tallennetaan puhelimen muistiin. Suorituksen jälkeen käyttäjä voi analysoida kerätyn datan, jolloin sovellus etsii datassa esiintyviä askelmalleja, ja laskee näiden esiintymismäärät datasta.

Sovelluksen toteutus onnistui hyvin ja sovellus pääsi lähes aina tarkempiin tuloksiin kuin puhelimessa valmiina ollut sisäinen askelmittari. Keskeisenä erona mittareiden askelmäärissä oli virheen suunta. Tässä työssä toteutetun askelmittarin laskemat askelmäärät olivat yhtä suuria, tai pienempiä kuin todelliset askelmäärät, kun taas puhelimen sisäisen mittarin ilmoittamat askelmäärät olivat todellista suurempia. Sovelluksen käyttöä kuitenkin rajoittaa sovelluksen laskenta-ajan nopea hidastuminen pitkiä mittauksia analysoitaessa.

Sisältö

Symboli- ja lyhenneluettelo

$\vec{a}$	Kiihtyvyys
$\vec{\alpha}$	Kulmakiihtyvyys
t	Aika
n	Pisteiden määrä liukuvan keskiarvon tasoituksessa
σ	Keskihajonta
M	Analyysissa käytettävä malli
T	Testijakso, jota verrataan malliin
d	Kiihtyvyysdimensio
å	Summa
	Absoluuttinen virhe

API	Application programming interface

1 JOHDANTO

1.1 Tausta

Päivittäiseksi askeltavoitteeksi suositellaan 10000 askelta [5]. Tämän ylittävällä askelmäärällä saadaan ehkäistyä painon nousua. Askelmittareiden käyttö ja askelmäärän seuraaminen päivittäin kasvattaa käyttäjän fyysistä aktiivisuutta [4]. Päivittäistä aktiivisuutta voidaan osittain mitata askelmittareilla, jotka mittaavat päivän aikana kävellyt askeleet. Askelmittareita on saatavilla nykypäivänä monissa eri muodoissa, kuten aktiivisuusrannekkeissa, älykelloissa ja älypuhelimissa. Laitteissa valmiina olevat askelmittarit kuitenkin kertovat usein ainoastaan askelmäärän jollain valmistajan ennalta määrittämällä aikavälillä, kuten vuorokauden kokonaisaskelmäärä. Tällaiset askelmittarit eivät anna tarpeeksi informaatiota, jos halutaan tietää enemmän käyttäjän aktiivisuudesta. Liikunnan keston ja askelmäärän lisäksi intensiivisyydellä on myös merkitystä kunnon ylläpidossa. Tästä syystä olisi hyvä tietää esimerkiksi käyttäjän sykkeen muutokset liikuntasuorituksen ajalta. Sykkeen mittaamiseksi tarvitaan kuitenkin siihen erikoistunutta laitteistoa, joten ainoastaan puhelimesta valmiiksi löytyvillä ominaisuuksilla sykkeen mittaaminen ei onnistu. Syke ei kuitenkaan ole ainoa mahdollisuus mitata aktiivisuuden muutoksia. Juostessa ihminen liikkuu eri tavalla kuin kävellessä, jolloin myös mukana oleva puhelin liikkuu eri tavalla. Näitä muutoksia liikkumistavoissa voidaan havaita, sillä moderneissa puhelimissa on useita erilaisia sensoreita[3], jotka mittaavat ympäristöä jatkuvasti. Näihin sensoreihin perustuvat monet puhelimen toiminnot, kuten sovelluksen käyttöliittymän mukautuminen puhelimen asentoon ja puhelimessa mahdollisesti valmiina oleva askelmittari.

Android versio 4.4 toi mukanaan käyttöjärjestelmään sisäänrakennetun askelmittarin, joka käyttää kiihtyvyysanturia askeleiden laskemiseen. Tämä anturi tarjoaa ainoastaan tiedon kokonaisaskelmäärästä [2]. Jos halutaan tarkempaa informaatiota liikuntasuorituksesta, on luotava erillinen askelmittarisovellus.

1.2 Tavoitteet ja rajaukset

Työn tavoitteena on kehittää sovellus, jolla käyttäjä voi liikkuessaan tallentaa puhelimen keräämää dataa kiihtyvyys- ja gyroskooppisensoreilta. Kiihtyvyys-sensori mittaa puhelimen lineaarista kiihtyvyyttä ja gyroskooppi mittaa puhelimen kiertymistä itsensä ympäri. Datasta etsitään askeleita tunnistamalla kohdat, joissa askeleeseen liittyvä jakso esiintyy. Alustava askelmäärä voidaan laskea mittauksen aikana laskemalla kiihtyvyyksien maksimikohtia.

Mittauksen jälkeen voidaan tehdä tarkempaa analyysia liikuntasuorituksesta ja vertailla sitä edellisiin mittauksiin. Tavoitteena jälkianalyysissa on luoda askelprofiili keskimääräisestä askeleesta, jota voidaan verrata tulevien mittausten keskimääräiseen askeleeseen. Näin voidaan verrata askeleen maksimikiihtyvyyttä ja askeltiheyttä.

Sovelluksella on tarkoitus mahdollistaa liikuntasuorituksen vertailu edellisiin suorituksiin, mutta käyttäjän yleistä kuntoa ei analysoida datasta. Tällaisen analyysin suorittamiseen tulisi tehdä yhteistyötä hyvinvointialan ammattilaisten kanssa. Todennäköisesti myöskään pelkästä kiihtyvyysdatasta ei saada tarpeeksi informaatiota, jotta voitaisiin arvioida luotettavasti käyttäjän kuntoa. Käyttäjän profiili tallennetaan puhelimeen paikallisesti. Näin saadaan yksinkertaistettua toteutusta, ja minimoitua mahdollisia tietoturvariskejä.

Käyttöliittymästä tehdään yksinkertainen ja mahdollisimman helppokäyttöinen. Painikkeiden tulee olla tarpeeksi isoja, jotta niihin on helppo osua myös liikkuessa. Mittaus on tarkoitus tehdä puhelimen ollessa taskussa, tai muuten sijoitettuna niin, että se voi vapaasti liikkua käyttäjän liikkeiden mukana. Näin ollen mittauksen on toimittava myös näytön ollessa sammutettuna. Sovellus toteutetaan Android-ympäristöön, joten sovellus ei ole yhteensopiva muiden käyttöjärjestelmien kanssa, kuten iOS ja Windows Phone.

1.3 Rakenne

Luvussa 2 tutustutaan signaalinsuodatukseen ja jaksontunnistukseen askelmittauksessa. Itse projektin toteutukseen keskitytään luvussa 3. Sovellus tarvitsee käyttöjärjestelmältä luvan lukea puhelimen antureista tietoa datan keräämiseksi. Datan käsittelylle ja analyysille tulee luoda funktioita, ja analyysistä saatavat tulokset on voitava esittää käyttäjälle käyttöliittymän kautta.

Sovellustoteutuksen jälkeen luvussa 4 tarkastellaan sovelluksen mittaustuloksia. Askelmääriä verrataan muihin askelmittareihin ja selvitetään, muuttuvatko tulokset puhelimen ollessa eri paikoissa ja asennoissa käytön aikana. Pyritään myös ottamaan selvää, toimiiko askelmittari erilaisten ihmisten käytössä. Lopuksi tarkastellaan työn onnistumista ja mahdollisuuksia sovelluksen jatkokehitykselle.

2 SIGNAALINKÄSITTELY ASKELMITTAUKSESSA

2.1 Signaalin suodatus

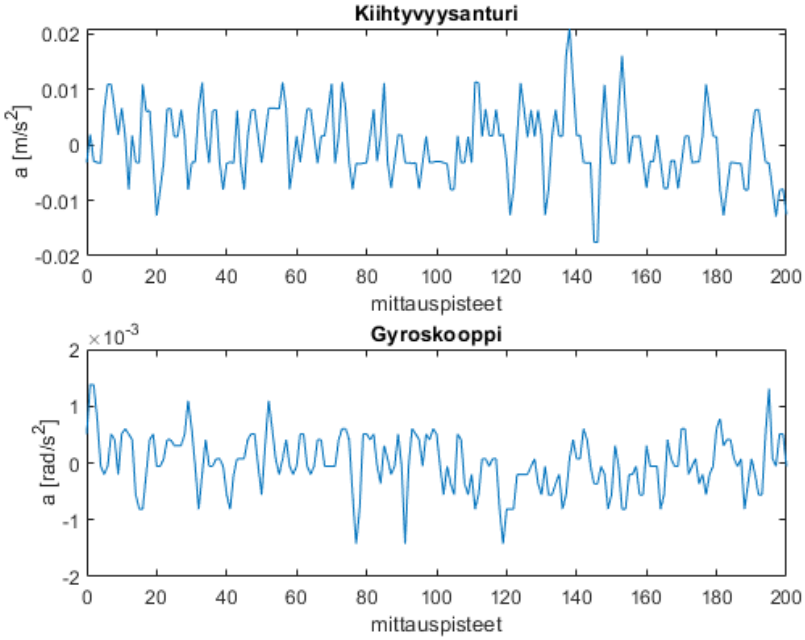
Puhelimen asento voi olla erilainen jokaisen mittauksen välillä. Näin ollen *x*-, *y*-, ja *z*-suunnassa mitatut arvot voivat vaihdella mittauksesta toiseen. Jotta mittauksia voitaisiin vertailla eri mittauskertojen välillä, data tulee muokata sellaiseen muotoon, johon puhelimen asento ei vaikuta. Mittauspisteiden vektorisumma kuvaa puhelimen havaitsemaa kokonaiskiihtyvyyttä kolmiulotteisessa avaruudessa. Tämän vektorisumman pituudesta saadaan laskettua kiihtyvyyden $\vec{a}$ suuruus kaavalla

$$|\vec{a}| = \sqrt{a_x^2 + a_y^2 + a_z^2}, \tag{1}$$

jossa a_i on kiihtyvyys suuntaan i. Tällaiseksi muokattu data ei riipu puhelimen alkuperäisestä orientaatiosta. Sama voidaan tehdä puhelimen gyroskooppidatasta saatavalle kulmakiihtyvyydelle $\vec{\alpha}$. Kävelyn aikana puhelimen havaitsemat kiihtyvyydet johtuvat pääosin puhelimen liikkumisesta taskussa. Kiihtyvyydet käyttäjän hidastaessa tai nopeuttaessa kävelyvauhtia oletetaan huomattavasti pienemmiksi kuin puhelimen kokema kiihtyvyys jalan äkillisessä pysähtymisessä sen osuessa maahan.

Antureilta saaduissa signaaleissa on kohinaa, eli mittauksessa syntyvää häiriötä. Kohina on poistettava datasta ennen askelten tunnistamista, jotta voidaan olla mahdollisimman tarkka tunnistetun askeleen oikeellisuudesta. Antureiden epätarkkuudesta syntyy kohinaa, vaikka puhelin olisi täysin paikallaan. Kuvassa 1 näkyy sovelluksen kehityksessä käytetyn puhelimen antureiden arvoja puhelimen ollessa paikallaan pöydällä. Kiihtyvyyksien suuruus on laskettu käyttäen kaavaa 1. Lineaarisesta kiihtyvyydestä on poistettu painovoiman vaikutus vähentämällä datasta keskiarvo. Kuvasta arvioidaan virheet lineaariselle kiihtyvyydelle $a = 0,01 m/s^2$ ja kulmakiihtyvyydelle $\alpha = 0,001 rad/s^2$. Suurin osa askelmittauksessa syntyvästä kohinasta syntyy todennäköisesti puhelimen liikkumisesta taskussa, varsinkin jalan osuessa askeleen aikana maahan.

Kohinaa voidaan poistaa erilaisilla suodatusmenetelmillä. Puhelimen tärinästä ja antureiden epätarkkuudesta aiheutuva kohina on suuritaajuuksista verrattuna askeleen jaksonaikaan. Suuria taajuuksia voidaan poistaa datasta alipäästösuotimilla. Yksinkertainen esimerkki alipäästösuotimesta on liukuvalla keskiarvolla suoritettu tasoitus. Tasoitus suoritetaan kaavalla



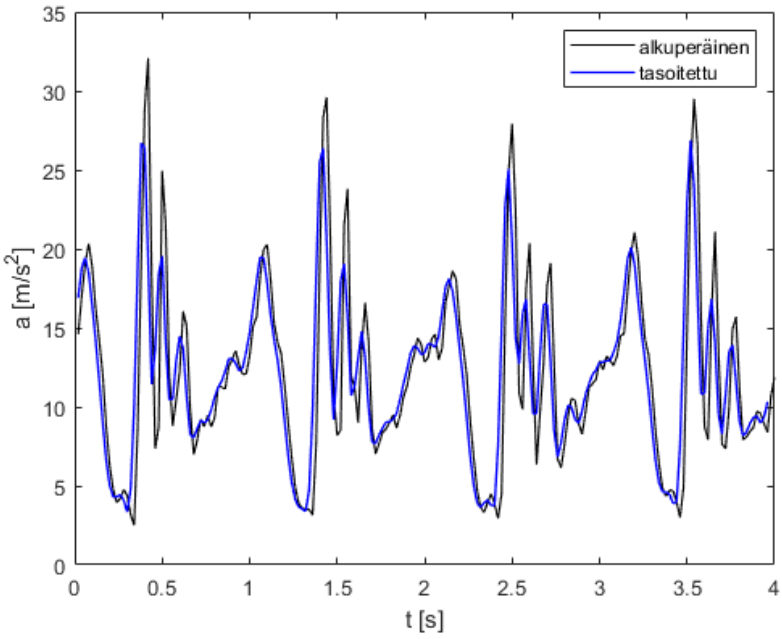
Kuva 1. Puhelimen antureiden epätarkkuus.

$$\overline{p}_m = \frac{1}{n} \sum_{i=0}^{n-1} p_{m-i}, \tag{2}$$

jossa p_m on m:nnes mittauspiste, n on keskiarvoon mukaan otettavien pisteiden määrä ja $\overline{p}_m$ on mittauspisteen tasoitettu arvo. Näin saadaan pienennettyä nopeita vaihteluita datasta, säilyttäen kuitenkin datassa esiintyvät pitkän aikavälin muutokset. Tasoituksessa tarkastellaan pistettä m edeltäviä pisteitä, jotta tasoitus voidaan suorittaa välittömästi uusia pisteitä mitattaessa.

Kuvassa 2 näkyy liukuvalla keskiarvolla suoritettu tasoitus, johon on otettu mukaan kolme pistettä. Tasoitettu kuvaaja muistuttaa suurilta osin alkuperäistä dataa, mutta äkilliset vaihtelut peräkkäisten pisteiden välillä ovat pienentyneet huomattavasti. Tämä muutos voidaan havaita varianssista. Taulukossa 1 on esitettynä alkuperäisen datan, ja kahdella eri n:n arvolla tasoitetun datan keskihajonnat ja keskiarvot. Alkuperäisen datan keskihajonta on $\sigma_{n=1} = 5.9683$. Kolmen pisteen keskiarvolla tasoitetun datan keskihajonta on pienentynyt noin 13% alkuperäisestä, arvoon $\sigma_{n=3} = 5.1688$. Kahdellatoista pisteellä tasoitettaessa keskihajonta on jo huomattavasti pienempi, noin 43% alkuperäisestä. Keskiarvo on pysynyt lähellä alkuperäistä. Data on tallennettu puhelimen kiihtyvyysanturilla. Mittauksen aikana puhelin on ollut housujen oikeanpuoleisessa etutaskussa ja mittaus on toteutettu kävellen.

Tasoitusta saadaan voimakkaammaksi ottamalla keskiarvoon mukaan suurempi määrä



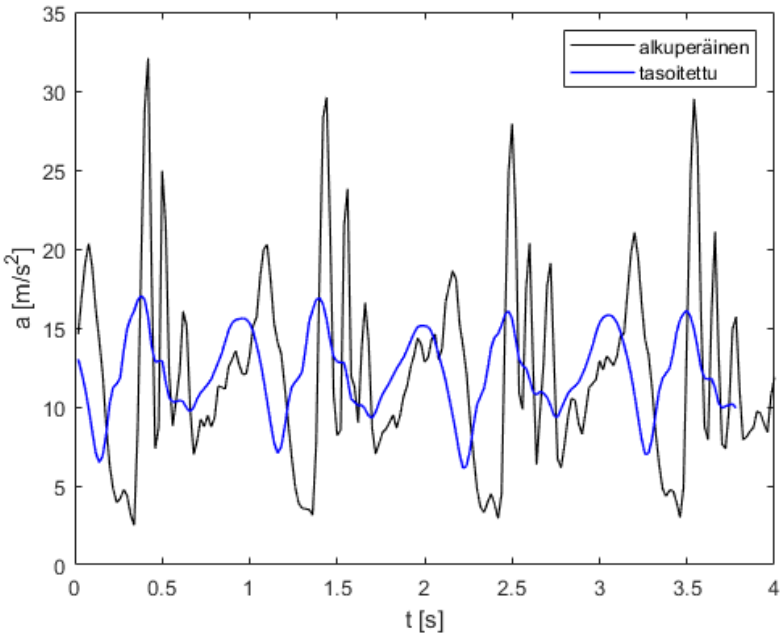
Kuva 2. Signaalin tasoitus kolmen pisteen liukuvalla keskiarvolla.

Taulukko 1. Liukuvan keskiarvon vaikutus dataan.

	Alkup.	n=3	n=12
Keskihajonta, σ	5.9683	5.1688	2.5909
Keskiarvo	12.1784	12.1656	12.1393

pisteitä. Kuitenkin suuret pistemäärät keskiarvon laskemisessa voivat aiheuttaa datan liiallista vääristymistä. Kuvassa 3 keskiarvoon on sisällytetty 12 pistettä. Mittaustuloksena saatu kiihtyvyys käy neljä kertaa arvon $a = 5m/s^2$ alapuolella. Nämä ovat kohtia, joissa puhelimen puoleinen jalka on nostettu ylös. Vastakkaisen jalan liike näkyy minimien välissä, pienempinä vaihteluina. Näin laskemalla voidaan tulkita alkuperäisessä datassa olevan neljä jaksoa. Kun tasoitukseen otetaan kahdentoista edeltävän pisteen keskiarvo, datan arvojen vaihtelu vähenee. Tästä syystä tasoitetusta datasta on vaikea arvioida, esiintyykö siinä neljä vai kahdeksan jaksoa. Myös kiihtyvyyden maksimiamplitudi tasoittuu, jolloin kävelystä on vaikeampi tunnistaa muita ominaisuuksia, kuten askeleen painoa eri mittausten välillä.

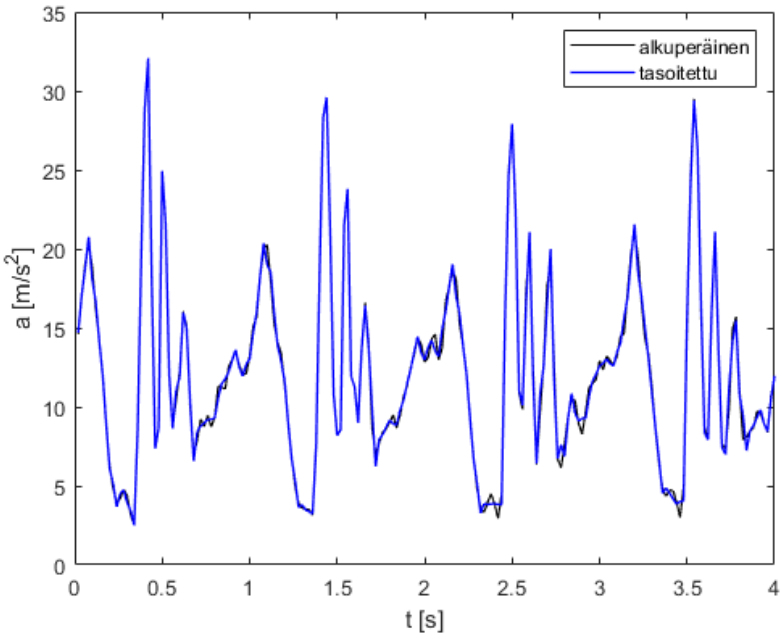
Toinen vaihtoehto korkeiden taajuuksien poistamiseen on diskreettien aallokemuunnosten hyödyntäminen [1]. Yksi yksinkertaisimmista aallokefunktioista on Haarin aalloke, jossa lasketaan kahden peräkkäisen pisteen keskiarvo ja pisteiden erotus. Erotuksesta saadaan selville datassa tapahtuvan muutoksen suuruus. Kun erotuksista poistetaan pienet arvot ja palautetaan data alkuperäiseen muotoon, datasta voidaan poistaa pieniä vaihteluita. Nopeasti



Kuva 3. Signaalin tasoitus 12 pisteen liukuvalla keskiarvolla.

tapahtuvat suuret vaihtelut säilyvät datassa muuttumattomina. Monimutkaisemmissa aallokefunktioissa otetaan huomioon suurempi määrä pisteitä. Kuvassa 4 on Daubechien D4 -aallokkeella suoritettu tasoitus. Tasoitusta varten kohinan sisältävästä datasta kaikki arvot alle 2 on nollattu, jolloin datan tulisi tasoittua. Pieniä muutoksia lukuun ottamatta tasoitettu käyrä näyttää noudattavan alkuperäistä kuvaajaa erittäin tarkasti. Erilaisia tuloksia voitaisiin saada muuttamalla tasoituksessa käytettävää arvoa, tai ottamalla käyttöön jokin muu aallokefunktio. Tämä kuitenkin tuo toteutukseen huomattavasti lisää kompleksisuutta, eikä ole varmaa tulisivatko tulokset olemaan optimaalisia erilaisia kävelytyylejä kohdattaessa.

Aallokkeilla suoritettu tasoitus ei siirrä datan piirteitä, toisin kuin kaavan (2) liukuvalla keskiarvolla tasoittaminen. Tässä projektissa datan piirteiden absoluuttinen paikka ei kuitenkaan ole tärkeässä roolissa, vaan merkitsevää on jaksojen lukumäärä. Alkuperäisessä datassa ajanhetkien 1s, 2s ja 3s läheisyydessä jaksojen yksityiskohdat eroavat toisistaan jonkin verran. Näitä sattumanvaraisia muutoksia tulisi saada poistettua, jotta tasoitetut jaksot muistuttaisivat toisiaan mahdollisimman hyvin. Kuvan 4 aalloketta käyttämällä osa kohinasta näillä alueilla tasoittuu, mutta suuremmat muutokset jäävät voimaan. Kuvan 2 liukuvan keskiarvon tasoituksella jaksoihin jää myös joitain eroavaisuuksia, mutta liukuvan keskiarvon menetelmä näyttäisi tasoittavan dataa johdonmukaisemmin. Liukuvan keskiarvon tasoitus on myös yksinkertaisempi ja kevyempi suorittaa dynaamisesti samalla kun uusia datapisteitä kerätään.



Kuva 4. Signaalin tasoitus Daubechien D4 aallokkeella.

2.2 Jakson tunnistaminen

Kävelyn aikana datasta voidaan etsiä maksimikohtia, joiden perusteella saadaan arvioitua keskimääräiseen askeleeseen kuluvaa aikaa. Myös alustavaa askelmäärän laskemista voidaan suorittaa mittauksen aikana. Maksimikohtia voi kuitenkin esiintyä datassa myös ilman kävelyä, joten tällä laskutavalla voi olla vaikea päästä luotettaviin tuloksiin. Askelprofiilin luomiseksi datasta on löydettävä yhden askeleen jakso, jotta voidaan nähdä, millainen on keskimääräinen askel ja verrata eri mittauskertoja keskenään.

Kävely on yleensä suhteellisen tasaista, ja askelten väli ajallisesti ei muutu merkittävästi. Joskus kuitenkin käyttäjä voi ottaa pidemmän askeleen, tai pysähtyä paikalleen hetkeksi. Tästä syystä mittauksesta saatua aikasarjaa ei voida olettaa stationaariseksi. Stationaarisuuden puuttumisen takia Fourier-analyysia on vaikea toteuttaa datalle. Jakson muodossa voi olla merkittäviä eroja eri mittausten välillä riippuen puhelimen asennosta ja mittaajan liikkumistyylistä, joten ennalta määrättyjä jakson malleja on vaikea luoda. Jakso täytyy siis ensin tunnistaa datasta jokaisen mittauskerran jälkeen, jonka jälkeen löydettyä jaksoa voidaan käyttää kyseisen mittauksen analysoinnissa.

Kuten kuvasta 2 nähtiin, puhelimen ollessa housujen taskussa, vasemman ja oikean jalan askeleet näyttävät erilaisilta. Tästä syystä voi olla helpompaa määrittää askeleet pareittain ja kertoa lopputulos kahdella. Ongelmaksi kahdella kertomisessa syntyy se, että sovelluksen antama askelmäärä on virheellinen käyttäjän kävellessä parittoman määrän askeleita. Toinen

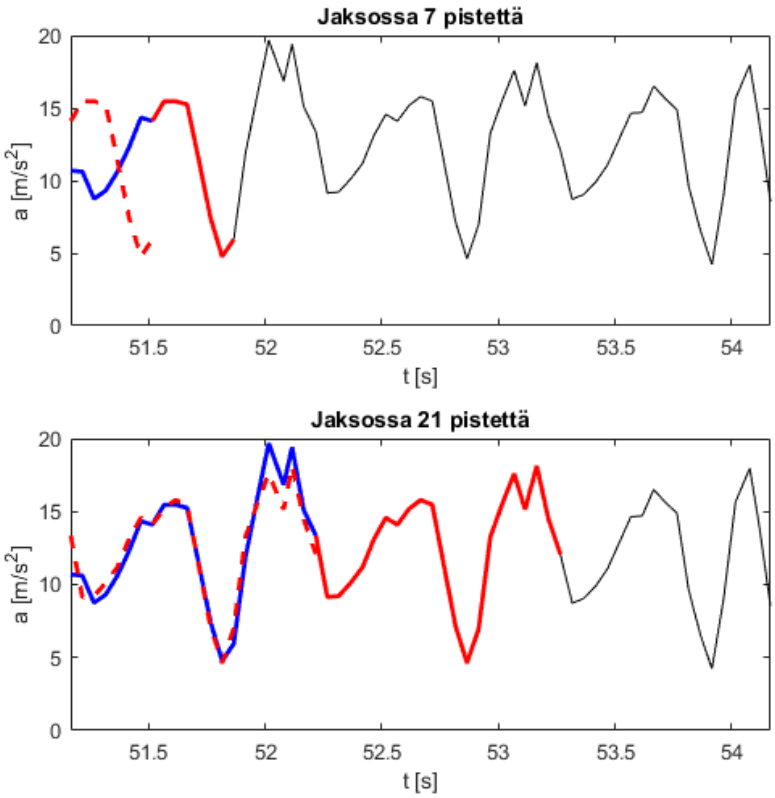
ongelma voi syntyä tilanteessa, jossa tunnistetaan askelpariksi vain yhden askeleen jakso. Tällöin on mahdollista, että askeleet on laskettu alun perin oikein, ja lopuksi askelmäärä on kaksinkertainen todelliseen verrattuna. Tällaisia virheitä voidaan välttää arvioimalla mittauksen aikana keskimääräiseen askeleeseen, tai askelpariin kuluva aika ottamalla aikaa kiihtyvyyden maksimiarvojen välillä.

Askeleeseen kuluva aika voidaan myös arvioida laskemalla. Ihmisen keskimääräinen kävelynopeus on $5km/h$, ja keskimääräinen askelpituus $0,8m$[6]. Näin ollen viiden kilometrin matkalla askeleita kävellään $\frac{5000m}{0,8m} = 6250$. Askeleen keskimääräiseksi ajaksi saadaan tällöin $\frac{3600s}{6250} = 0,576s$. Askeleen pituus ja kävelynopeus vaihtelee käyttäjien välillä, mutta näin laskettuna voidaan arvioida kahden askeleen astumiseen kuluvaksi ajaksi noin yksi sekunti. Tällaisella arviolla voidaan ehkäistä liian lyhyiden, tai pitkien jaksojen etsimistä datasta ja parantaa todennäköisyyttä, että askeleet lasketaan oikein.

Askeljakson löytämiseksi voidaan tutkia kahta peräkkäistä jaksoa datassa. Aloitetaan kahdella lyhyellä jaksonpituudella, jotka ovat peräkkäin. Kuvassa 5 verrataan kahta peräkkäistä jaksoa toisiinsa. Punaisella korostettua jaksoa verrataan siniseen jaksoon. Punainen jakso on piirretty katkoviivalla sinisen päälle, jotta nähdään helpommin, kuinka paljon ne muistuttavat toisiaan. Seitsemää pistettä käytettäessä jaksot eivät muistuta toisiaan, mutta jaksonpituutta kasvattamalla voidaan löytää kaksi peräkkäistä jaksoa, jotka varsinkin suurten kiihtyvyysvaihteluiden kohdalla ovat hyvinkin samanlaiset.

Tällaisessa toteutuksessa on kuitenkin riskinä, että käyttäjä ei kävele analyysin alkupisteessä, jolloin askelta ei koskaan löydetä, tai löydetään virheellinen jakso. Jakson oikeellisuutta voidaan arvioida vertaamalla sen pituutta aiemmin luotuun arvioon keskimääräisen askeleen jaksonajasta. Jos kunnollista jaksoa ei löydetä, voidaan valita uusi aloituspiste ja toistaa analyysi. On myös mahdollista, että kävelijällä on epänormaalin pitkä tauko askeljaksojen välillä. Tämä voidaan ottaa huomioon luomalla tilaa vertailtavien jaksojen väliin. Löydetystä jaksosta tulee lyhyempi, mutta liian pitkän mallin syntyminen on epätodennäköisempää.

Jakson löytymisen jälkeen käydään koko data läpi ja lasketaan askelmallin kanssa yhtenevien jaksojen lukumäärä. Mallia voidaan myös muokata jokaisen mallin kanssa yhtenevän jakson kohdalla, jotta liikkumisen aikana mahdollisesti tapahtuvat pienet muutokset saadaan huomioitua. Ongelmana laskemisessa tällä menetelmällä on suuret äkilliset muutokset askeleessa, kuten esimerkiksi siirtyminen kävelystä juoksuun. Alueet, joista askeleita ei löytynyt voidaan kuitenkin käydä uudestaan läpi. Tällöin saadaan luotua erilaisille liikkumistavoille omat mallinsa. Luotavien mallien määrää on kuitenkin rajoitettava, sillä jos ohjelma esimerkiksi tunnistaa kymmenen erilaista askelmallia, herää kysymys millaiseen liikkeeseen kukin malli liittyy. Jos taas löydetään vain kaksi mallia, mutta datasta on



Kuva 5. Peräkkäisten jaksojen vertaaminen toisiinsa.

analysoitu suurin osa, voidaan yksi malli tulkita juoksuksi, ja toinen kävelyksi. Myös mallin esiintyvyys datassa on hyvä ottaa huomioon. Jos mallia käyttämällä löydetään vain muutama askel, on mahdollista, että tunnistettu malli ei liity kävelyyn. Tämä voi olla jotain muuta liikettä, josta sovellus on löytänyt jaksollisuutta. Askelmallin tunnistusta sovelluksen kannalta, ja löydettyjen mallien vertaamista dataan käsitellään tarkemmin luvussa 3.4.

3 OHJELMISTOPROJEKTIN TOTEUTUS

3.1 Vaatimusmäärittely

Sovellus toteutetaan Android puhelimelle. Sovelluksen testaamisessa käytetään Huawei Honor 8 -puhelinta, jossa on käyttöjärjestelmänä Android versio 7.0. Android-sovelluksia kehitetään Android Studio -ympäristöllä. Android Studio on saatavilla Windows, Linux ja macOS käyttöjärjestelmille. Alunperin Androidin sovelluskehityksessä käytettävä kieli oli Java, mutta vuonna 2017 Google ilmoitti tukevansa Kotlin-ohjelmointikieltä Android kehityksessä[7]. Nykyään Kotlin on suosituin kieli Androidin sovelluskehityksessä, joten tämä projekti toteutetaan käyttäen Kotlinia.

Sovelluksen tulee pystyä tallentamaan puhelimen kiihtyvyysantureilta saatavaa dataa. Antureilta saatava data on kolmiulotteista. Kiihtyvyysanturi kertoo puhelimen havaitseman kiihtyvyyden $\vec{a}$ lineaarisessa liikkeessä. Tähän sisältyy myös maan painovoiman aiheuttama putoamiskiihtyvyys, noin 9.81 m/s^2. Gyroskooppi mittaa puhelimen kulmakiihtyvyyttä $\vec{\alpha}$ pyörimisliikkeessä. Mittauksen yhteydessä kiihtyvyys- ja gyroskooppidatasta poistetaan kohina kaavalla (2), jossa $n = 3$. Tasoituksen jälkeen mittaukset tallennetaan mittausajan kanssa csv-tiedostoon. Tiedostoon tallennetaan taulukko, jossa on viisi saraketta, ja niin monta riviä kuin on mittauspisteitä. Tiedostosta data voidaan myöhemmin lukea analysointia varten. Datan tallentamisen tulee toimia puhelimen ollessa taskussa ja näytön ollessa lukittuna.

Analysoitaessa datasta tulisi löytää niin monta askelmallia, kun on liikkumisen aikana esiintynyt. Kuitenkin erittäin lyhyesti esiintyviä askelmalleja, esimerkiksi kaksi askelta koko datassa, tulisi välttää. Nämä mallit tulisi sisällyttää yleisemmin esiintyviin malleihin, jos ne ovat tarpeeksi samanlaisia. Käyttäjän on helpompi seurata omaa suoritustaan, kun malleja on rajoitettu määrä. Analyysin jälkeen askelmallien ominaisuudet lisätään käyttäjän profiiliin painotetulla keskiarvolla niin, että useammin esiintyneillä malleilla on suurempi painokerroin muihin verrattuna.

Sovellukseen luodaan kolme välilehteä, joista ensimmäisessä on tietoa sovelluksesta ja käyttäjän edellisistä analyyseista. Toisella välilehdellä suoritetaan datan tallennus, ja kolmannella datan analysointi. Analysoinnin seurauksena käyttäjä saa tietää suorituksen kokonaisaskelmäärän, ja lisäksi jokaisen tunnistetun askelmallin askelmäärät. Askelmalleista esitetään kuvaajat, jotta käyttäjä voi vertailla malleja toisiinsa. Askelmalleja ei kategorisoida eri liikkumistyyleihin, kuten kävely tai juoksu. Tämä johtuu siitä, että jokaisella eri käyttäjällä voi olla yksilölliset liikkumistyylit. Käyttäjä voi itse kategorisoida tuloksensa perustuen mal-

lin antamiin tietoihin. Jokaisesta mallista kerrotaan mallilla löydettyjen askelten lukumäärä, mallin keskimääräinen kiihtyvyys, maksimikiihtyvyys ja varianssi.

3.2 Ohjelmiston toteutus Android Studio -ympäristössä

Projektia luodessa valitaan, millaiselle laitteelle sovellus halutaan kehittää. Esimerkiksi puhelimelle ja TV:lle tarjotaan erilaiset määritykset. Puhelinsovelluksen kehitykseen tarjotaan monenlaisia pohjaratkaisuja. Tähän sovellukseen valitaan ”Bottom Navigation Activity”, jossa on sovelluksen alareunassa painikkeet välilehtien välillä siirtymiseen. Seuraavaksi annetaan sovellukselle nimi, ja valitaan millä käyttöjärjestelmän versioilla sovelluksen tulee toimia. Uudemmilla järjestelmäversioilla voidaan käyttää vanhempia sovelluksia, mutta vanhemmissa versioissa ei välttämättä ole tarvittavia ominaisuuksia, jolloin sovellusta ei voida käyttää. Minimiversioksi valitaan API 23: Android 6.0, jonka tulisi toimia yli 80%:lla laitteista. Tämän jälkeen Android Studio luo käyttäjälle projektin.

Android-sovellukset toimivat aktiviteettien kautta (Activity). Aktiviteetit toimivat itsenäisesti ja niillä voidaan erotella sovelluksen eri toimintoja toisistaan. Esimerkiksi ulkoisesta sovelluksesta sähköpostia lähetettäessä voidaan käynnistää vain sähköpostisovelluksen postinkirjoitusaktiviteetti. Näin säästetään käyttäjän aikaa ja puhelimen resursseja, kun ei tarvitse ensin avata sähköpostisovellusta ja tämän jälkeen navigoida sähköpostin lähetykseen. Tässä projektissa on yksi aktiviteetti; *MainActivity*, joka toimii sovelluksen pääohjelmana. Tästä aktiviteetista käynnistetään fragmentteja, joita on kolme. Fragmentit toimivat sovelluksen välilehtinä, joilla voidaan erotella sovelluksen ominaisuudet. Yksi fragmentti toimii ohjeistuksena ja käyttäjän profiilin tarkasteluna. Toisella fragmentilla tallennetaan kävelydata, ja kolmannella analysoidaan tallennettu data. *MainActivity* on käynnissä aina sovellusta käytettäessä, mutta fragmentit luodaan ja tuhotaan sen mukaan, mille välilehdelle navigoidaan.

3.3 Mittausdatan tallennus

Kiihtyvyysdatan tallentamiseksi sovelluksen täytyy pystyä lukemaan sensoreilta saatavaa informaatiota. Sensoreiden käyttöönotto tapahtuu *SensorManager* -oliolla, johon rekisteröidään kiihtyvyys- ja gyroskooppisensorit. Sensoreiden päivitystaajuudeksi asetetaan SENSOR_DELAY_GAME, joka on noin 50 mittausta sekunnissa. Funktio *onSensorChanged* suoritetaan, kun jonkin rekisteröidyn sensorin arvo muuttuu. Funktio saa arvona vain yhden sensorin tapahtuman kerrallaan, joten kiihtyvyys- ja gyroskooppisensoreiden arvot

eroavat toisistaan ajallisesti. Tämä ei kuitenkaan ole ongelma, sillä datasta etsitään jaksoja, eikä tiettyjen tapahtumien täsmällisiä aikoja.

Datan tallentamiseksi luodaan *DataCollector* -olio, joka pitää yllä pientä määrää edeltävistä mittauksista. Olio tasoittaa dataa laskemalla kolmen viimeisimmän mittauksen keskiarvon kaavalla (2). Kiihtyvyysanturin päivittyessä mittauspisteet tallennetaan tasoituksen jälkeen csv-tiedostoon. Yhdelle riville kirjoitetaan lineaarisen kiihtyvyyden aika [ms] alkaen mittauksen alusta, lineaarinen kiihtyvyys [m/s^2] x, y, ja z suunnissa ja viimeisin mitattu kulmakiihtyvyys [rad/s^2] x, y, ja z suunnissa. Tiedostossa on aina yhden mittauskerran data. Seuraavan mittauskerran uusi mittausdata kirjoitetaan entisen päälle, jolloin saadaan rajoitettua sovelluksen tallennustilan käyttöä puhelimesta.

Näytön ollessa lukittuna, tai sovelluksen ollessa taustalla, käyttöjärjestelmä asettaa sovelluksen lepotilaan, jolloin sensoreita ei välttämättä päivitetä tasaisesti. Tästä syystä sovellukseen lisätään *WakeLock* ominaisuus, joka pitää järjestelmän aktiivisena myös näytön ollessa lukittuna. *WakeLock* voi kuluttaa merkittävästi puhelimen akkua, joten se aktivoidaan ainoastaan, kun datan tallennus on käynnissä.

3.4 Mittausdatan analysointi

Sensoreilta on tallennettu kahta erilaista kiihtyvyyttä kolmiulotteisessa avaruudessa. Näin ollen datassa on yhteensä kuusi dimensiota. Analysointivaiheessa askel etsitään vertaamalla keskenään kahta peräkkäistä jaksonaikaa; malliehdotusta ja testijaksoa. Molempien jaksojen jaksonaikaa pidennetään, kunnes löydetään pituus, jolla jaksot ovat riittävän yhtenevät. Vertailu aloitetaan satunnaisesta kohdasta dataa, muttei kuitenkaan liian lopusta, jotta mallin pituutta on varaa kasvattaa. Vertailua varten malliehdotus ja testijakso skaalataan jakamalla kunkin dimension datapisteet euklidisella normillaan. Normi lasketaan kuten kaava (1), mutta summaan lasketaan kaikki tietyn dimension d arvot kaavalla

$$j\vec{a}_d j = \sqrt{\text{å}(\vec{a}_d{}^2)}. \tag{3}$$

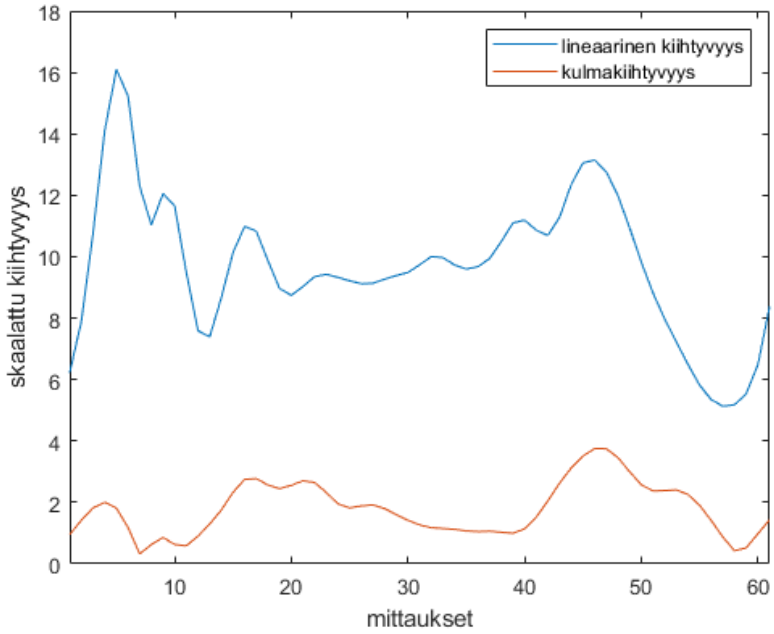
Tämän jälkeen vektorin alkiot jaetaan normilla käyttäen kaavaa

$$a_{\vec{d},i_{skaalattu}} = \frac{\vec{a_{d,i}}}{j\vec{a}_d j}, \tag{4}$$

jossa $\vec{a_{d,i}}$ on kiihtyvyysdimension d i:des alkio.

Näin datapisteet saadaan välille [-1,1]. Skaalaus tehdään jokaiselle dimensiolle erikseen, jotta jaksossa säilyy peräkkäisten mittauspisteiden väliset erot. kuvan 6 askelmalli on

skaalattu dimensioittain, jolloin on saatu kuvan 7 malli. Kuvan selkeyttämiseksi molempien liikkeiden kokonaiskiihtyvyydet on laskettu kaavalla (1). Kuvasta nähdään, että askeleen aikaiset kiihtyvyysvaihtelut erottuvat askeleesta selkeästi ja malli muistuttaa läheisesti alkuperäistä. Jakso voidaan skaalata myös mittauspisteittäin, jolloin skaalataan jokaisen mittauskerran 6-alkioisia vektoreita. Tämä on tehty samalle askelmallille kuvassa 8. Näin skaalaamalla datasta häviää mittauspisteiden väliset erot ja skaalattu malli ei muistuta alkuperäistä mallia.



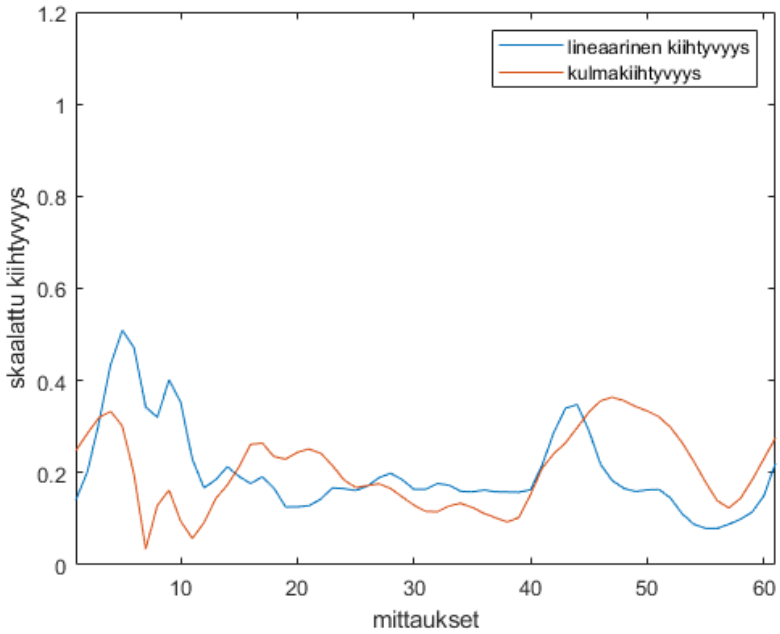
Kuva 6. Askelmalli ennen skaalausta.

Luvussa 2.2 laskettiin keskimääräisen askeleen astumiseen kuluvaksi ajaksi $t_{askel} = 0,576s$. Yhteen askelpariin kuluva aika on siis $2t_{askel} = 1,152s$. Sensoreiden mittausnopeus on 50 mittausta sekunnissa, joten askelpariin kuluva aika vastaa 57,6 sensorin mittausta. Jotkin askeleet voivat olla hitaampia ja toiset nopeampia, joten mallin pituusrajoiksi asetettiin 38 ja 78 mittauspistettä. Askelpariin kuluvan ajan tulee siis osua välille 0,76s - 1,56s.

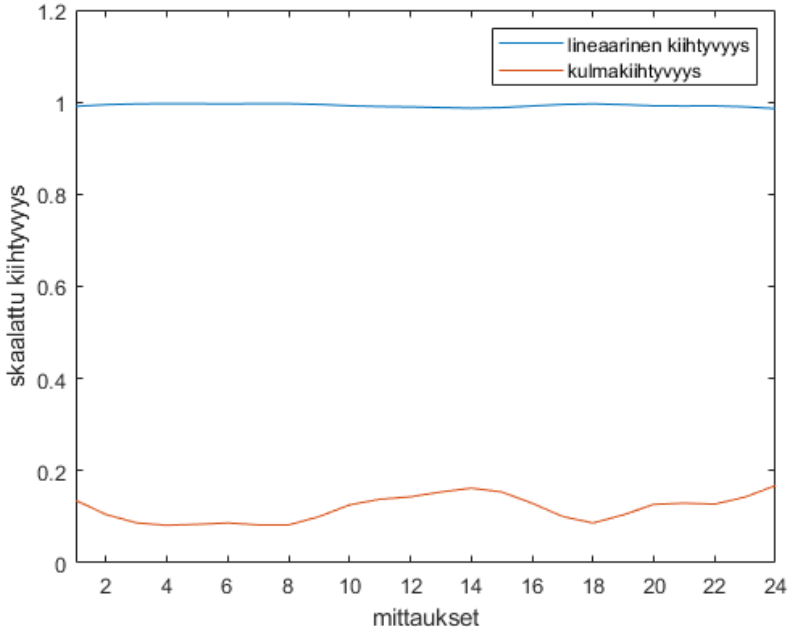
Askeleen tunnistamiseksi lasketaan malliehdotuksen $\vec{M}$ ja testijakson $\vec{T}$ pistetulo dimensioittain kaavalla

$$\vec{M} \ \vec{T} = \sum_{i=1}^{n} (\vec{M}_{d,i}\vec{T}_{d,i}), \tag{5}$$

jossa n on mallin pituus ja d on tarkasteltava dimensio. Datassa on kuusi dimensiota, jolloin saadaan kuusi pistetuloa. Askel tunnistetaan, kun pistetulojen keskiarvo on suurempi kuin 0,7



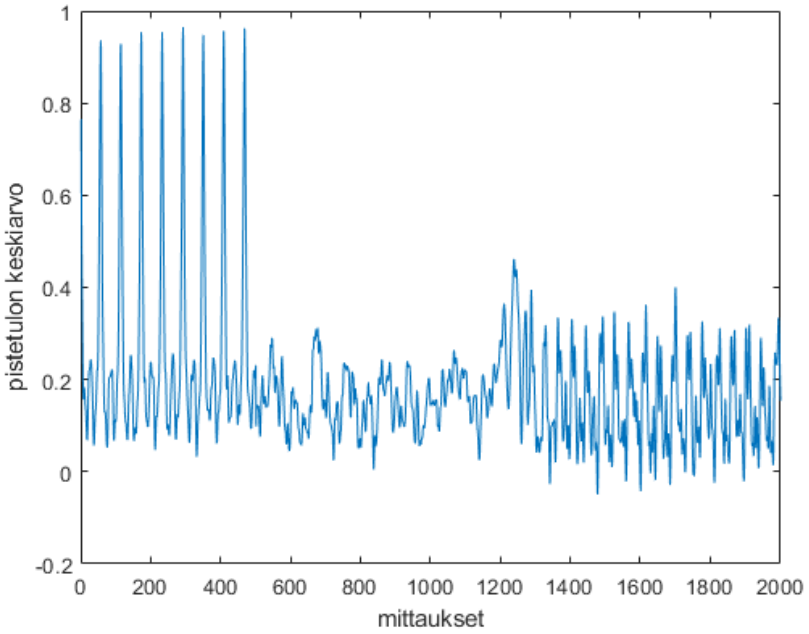
Kuva 7. Dimensioittain skaalattu askelmalli.



Kuva 8. Mittauspisteittäin skaalattu askelmalli.

ja pistetulo on maksimissaan. Arvo 0,7 on saatu testaamalla askeleen tunnistamista useilla eri liikkumistyyleillä. Kuvassa 9 näkyy erilaisten liikkumistapojen vaikutus pistetuloon. Aluksi datassa on ollut mallia vastaavaa liikettä, jolloin pistetulo käy lähellä arvoa 1. Välillä 500-1200 mittaaja on seissyt paikallaan, jolloin pistetulo on lähellä nollaa. Lopuksi mittaaja on hyppinyt paikallaan, jolloin liike on vastannut hieman enemmän mallin liikettä, mutta

nopeasta tahdista ja muuten erilaisesta liikkeestä johtuen pistetulo on pysynyt alle arvon 0,5. Arvolla 0,7 löydetään samaan liikkumistyyliin kuuluvat jaksot, mutta selkeästi erilaiset liikkeet jätetään laskematta. Pistetulon maksimi on löydetty, kun nykyisen ja edellisen pistetulon erotus on negatiivinen. Tällöin askelmalliksi valitaan edelliseen pistetuloon käytettyjen pisteiden joukko.

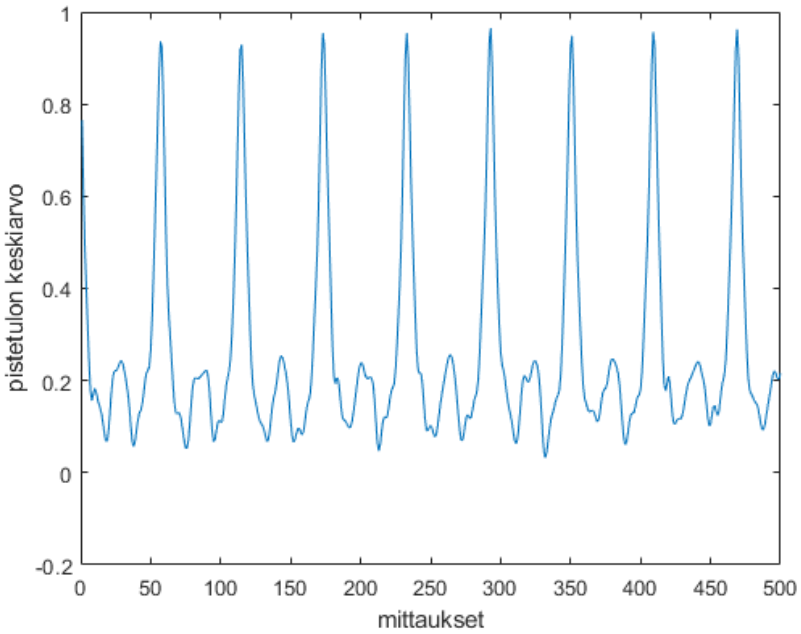


Kuva 9. Pistetulon arvoja kävellessä, paikallaan ollessa ja hyppiessä.

Askeleet lasketaan käymällä data läpi löydetyn mallin kanssa. Kuvassa 10 on esitetty kuvan 9 kävely osuus tarkemmin. Kuvassa näkyy selkeitä huippuja, joiden kohdalla malli on ollut hyvin samanlainen testijakson kanssa. Korkeiden huippujen väleissä on pieniä huippuja, joista näkyy vastakkaisen jalan askeleen korrelointi mallin kanssa. Pieniä huippuja ei kuitenkaan ole järkevää laskea, sillä niissä voin esiintyä useita maksimiarvoja. Esimerkiksi ainoastaan huippuja laskemalla mittauksen 325 kohdalla tulisi vastakkaisen jalan askel laskettua kahdesti. Korkeita huippuja laskemalla saadaan askelparien määrä selvitettyä.

Tässä vaiheessa dimensioittainen skaalaus hidastaa laskentaa, sillä jokainen testijakso täytyy skaalata erikseen. Koko dataa ei voida skaalata kerralla, sillä datassa mahdollisesti esiintyvät yksittäiset suuret piikit voisivat skaalata muun datan lähelle nollaa. Askeleen löydyttyä tämä osa datasta merkataan analysoiduksi, jolloin se on mahdollista ohittaa muiden askelmallien askelmäärää laskettaessa. Myöskään uusia askelmalleja ei hyväksytä analysoiduilta alueilta.

Malleja etsitään, kunnes datasta on analysoitu yli 90%, tai saavutetaan askelmallien maksimimäärä. Maksimimääräksi asetetaan 5, jolloin on mahdollista löytää juoksun ja kävelyn lisäksi muitakin malleja. Asetettaessa mallien maksimimäärä suuremmaksi,



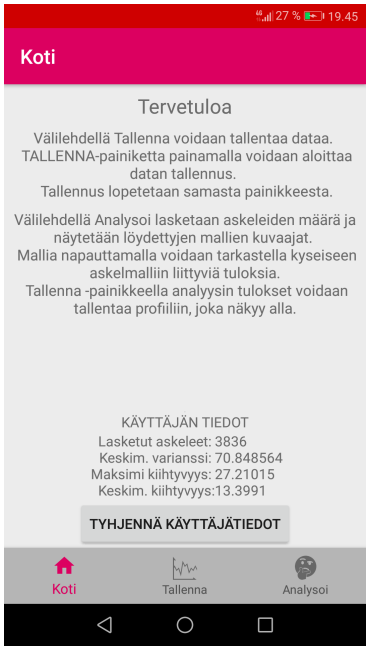
Kuva 10. Pistetulojen keskiarvoja kävelyn aikana.

analysoinnin kesto pitenee huomattavasti. Varsinkin jos koko datassa on löydettävissä alle 90% jaksollista liikkumista, jolloin mallien etsimistä ei lopeteta kesken.

3.5 Käyttöliittymä

Sovellus avautuu Koti-välilehteen, jossa on ohjeita sovelluksen käyttöön. Tällä välilehdellä esitetään myös käyttäjästä tallennetut tiedot. Jokaisesta tallennetusta askelprofiilista tallennetaan kokonaisaskelmäärä, askeleen keskimääräinen varianssi, maksimikiihtyvyys ja keskimääräinen kiihtyvyys. Käyttäjän profiilia muokataan näiden arvojen mukaan painokertoimella $\frac{\text{mittauksessa löydetyt askeleet}}{\text{kaikki askeleet yhteensä}}$.

Kuvassa 12 nähdään sovelluksen välilehti datan tallennukselle ja ilmoitus, joka näytetään dataa tallennettaessa. Välilehdessä saadaan reaaliaikainen esitys molempien sensorien havaitsemista kokonaiskiihtyvyyksistä ja sensoreiden arvot x, y ja z suunnissa. Näin käyttäjä voi tarkistaa sensoreiden toimivuuden ja nähdä kuinka puhelimen liikuttaminen vaikuttaa sensoreiden arvoihin. Kuvaajissa esitetään molempien sensoreiden 50 viimeisintä mittausta. Keskimääräinen askelpari sisältää 57,6 mittauspistettä, joten yksi askelpari mahtuu lähes kokonaan näytölle. Askelluksen vaikutusta sensoreihin voidaan siis seurata jo mittauksen aikana. Puhelimen pitäminen kädessä voi kuitenkin tuoda mittaukseen askeleisiin kuulomatonta liikettä, ja näin heikentää mittaustuloksia. Datan tallennus voidaan käynnistää TALLENNA-

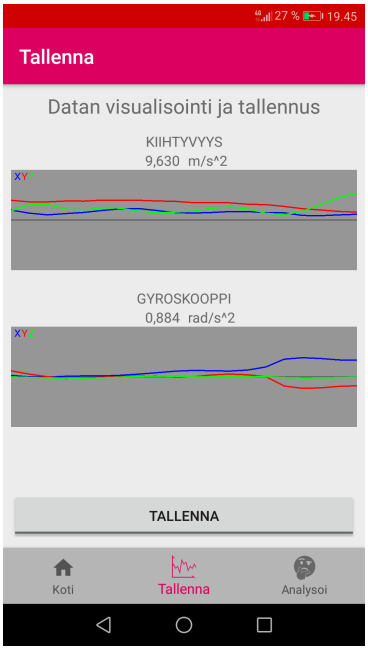


Kuva 11. Koti -välilehti.

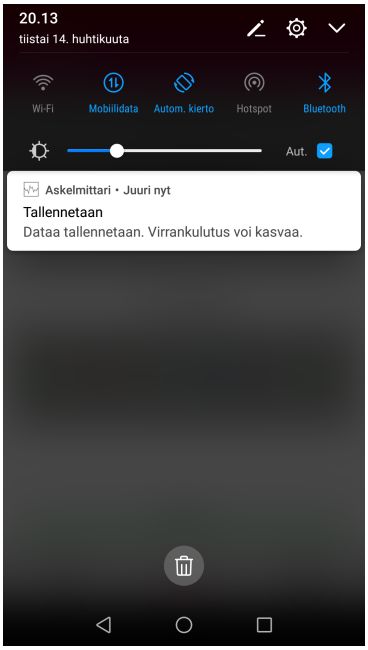
painikkeella. Tallennuksen alkaessa puhelimen ilmoitus ikkunaan lisätään ilmoitus datan tallentamisesta ja varoitus virrankulutuksen kasvamisesta tallennuksen aikana. Näin käyttäjä ei voi unohtaa sovellusta päälle ja kuluttaa puhelimen akkua tarpeettomasti. Ilmoitus poistetaan, kun käyttäjä sulkee sovelluksen tai lopettaa tallentamisen. Tallennuksen loppuessa ilmoitetaan myös käyttäjälle mittaukseen kulunut aika.

Tallennuksen jälkeen voidaan siirtyä Analysoi -välilehteen, jossa voidaan tarkastella viimeisimmän mittauskerran analyysia. Sovellus aloittaa datan analysoinnin taustalla välittömästi välilehden auetessa. Painikkeella ANALYSOI voidaan näyttää löydetyt askelmallit. Jos analyysi ei ole valmistunut painikkeen painamishetkellä, sovellus odottaa analyysin valmistumista. Analyysin valmistuttua sovellus piirtää jokaisen löydetyn askelmallin omaan kuvaajaansa. Kuvaajaan piirretään kiihtyvyys- ja gyroskooppikäyrät kaavalla 1 laskettuna. Kuvaajaa painettaessa sovellus näyttää yksityiskohtaisempia tietoja mallista. Kuvan 13 esimerkissä malleja on löydetty kaksi, joiden kuvaajat ovat esitetty allekkain. Analyysin jälkeen löydetyt mallit voidaan tallentaa käyttäjän profiiliin TALLENNA-painikkeella.

Tallennus välilehti

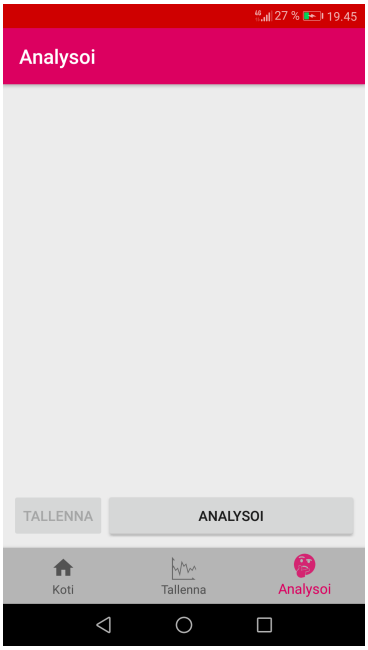


Ilmoitus datan tallentamisesta

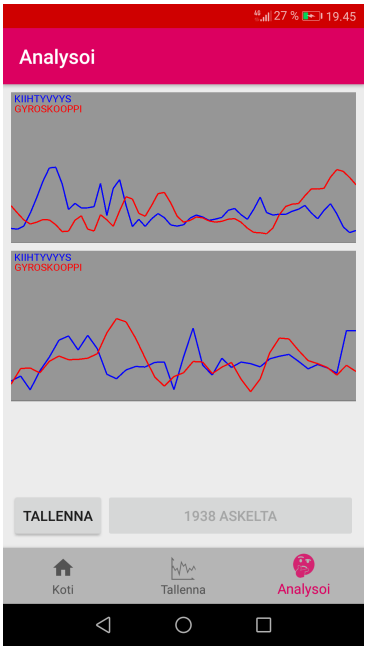


Kuva 12. Tallennus -välilehti.

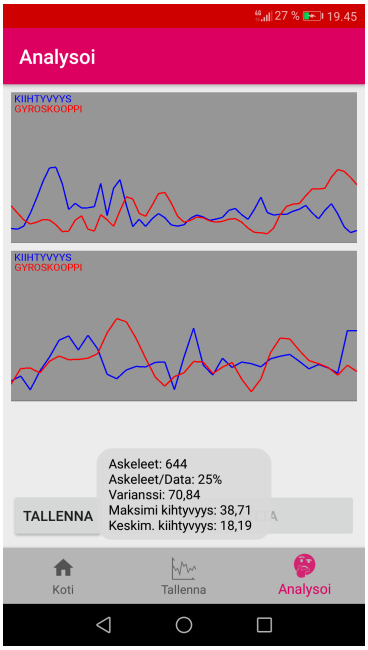
Ennen analysointia



Analysoinnin jälkeen



Mallin tiedot



Kuva 13. Analysoi -välilehti.

4 ANALYSOINNIN TULOKSET

Askelmittarin testaamiseksi käveltiin ja juostiin 200 askeleen jaksoja. Jokaiselta jaksolta kirjattiin ylös puhelimen oman sisäänrakennetun askelmittarin ilmoittamat askeleet, askelmittarisovelluksen löytämät askeleet ja todellinen askelmäärä. Todellinen askelmäärä laskettiin manuaalisesti suorituksen aikana. Jaksot pidettiin lyhyinä, jotta vältyttiin laskuvirheiltä todellisia askelmääriä laskettaessa.

Sovelluksen antamia askelmääriä verrattiin puhelimen sisäiseen askelmittariin. Vertailua varten käveltiin ja juostiin 200 askelta puhelimen ollessa erilaisissa taskuissa. Tulokset kirjattiin ylös taulukoihin 2 ja 3. Molemmissa liikkumatavoissa askelmittari löysi yhden askelmallin mittausta kohti, jolla kaikki mittauksen askeleet laskettiin.

Taulukko 2. Askelmäärät kävely.

	Povitasku	Etutasku	Takatasku
Todellinen	200	200	200
Puhelin	203	208	213
Askelmittari	198	200	200

Taulukko 3. Askelmäärät juoksu.

	Povitasku	Etutasku	Takatasku
Todellinen	200	200	200
Puhelin	198	231	208
Askelmittari	178	200	196

Povitaskussa askelmittari sai huonoimmat tulokset verrattuna muihin taskuihin, kun taas puhelimen sisäinen laskuri oli povitaskussa tarkin. Povitaskussa puhelin pääsi liikkumaan vapaammin takin mukana, jolloin gyroskoopin mittaamissa arvoissa esiintyi mahdollisesti sattumanvaraisuutta. Puhelimen sisäinen askelmittari käyttää ainoastaan lineaarista kiihtyvyyttä askelten mittaamiseen, joka voi vaikuttaa puhelimen tarkempiin arvoihin. Toisaalta molempien mittareiden askelmäärät olivat muita mittauksia alhaisemmat povitaskussa mitatessa. Tämän seurauksena puhelimen sisäinen mittari, joka muissa mittauksissa laskee johdonmukaisesti liikaa askeleita, osuu nyt lähemmäksi oikeaa arvoa. Voidaan siis ajatella molempien mittareiden jättäneen välistä joitakin askeleita povitaskusta mitatessa. Kävelyä mitattaessa sovellus suoriutui lähes täydellisesti, ja povitaskusta mitatessakin vain yhden askelparin jakso jäi havaitsematta. Juostessa etutaskusta mitattiin täydellinen tulos, ja takataskun

askelmäärästä puuttuu kaksi jaksoa. Povitasku suoriutui jälleen heikoiten, vain 178 tunnistetulla askeleella.

Askelmallien erottelukykyä varten yhdistettiin juoksu ja kävely samaan mittaukseen. Ensin käveltiin 80 askelta, tämän jälkeen juostiin toiset 80 askelta, ja lopuksi käveltiin 40 askelta. Tulokset kirjattiin taulukkoon 4. Esityksen selkeyttämiseksi taulukossa on esitettynä ainoastaan mittareiden ilmoittamat kokonaismäärät.

Taulukko 4. Askelmäärät Kävely-juoksu-kävely.

	Povitasku	Etutasku	Takatasku
Todellinen	200	200	200
Puhelin	209	216	217
Askelmittari	194	198	198

Myös yhdistetyillä liikkumistavoilla saatiin hyviä tuloksia etu- ja takataskusta, joissa molemmissa tunnistettiin kaksi askelmallia. Kummassakin mittauksessa löydettiin 120 + 78 askelta, joten oletettavasti juoksun 80 askeleen jaksosta jäi yksi askelpari mittaamatta. Povitaskusta mittaamalla löydettiin kolme askelmallia. Askeleita löydettiin 88+82+24. Mallin, jossa on 82 askelta, maksimikiihtyvyys oli 51,21 m/s^2. Maksimikiihtyvyydet 88 ja 24 askeleen malleilla olivat 15,07 m/s^2 ja 17,43 m/s^2. Tästä voidaan päätellä, että juoksuaskeleita mitattiin 82 ja kävelyä 112. On mahdollista, että juoksun aikana puhelin on liikkunut taskussa eri asentoon, jolloin juoksun jälkeiset kävelyaskeleet eivät ole vastanneet mittauksen alun kävelyä.

Myös juoksusta kävelyyn siirryttäessä voi esiintyä muutama askel, jotka eivät vastaa keskimääräistä liikkumista. Tämä voi myös johtaa uuden askelmallin syntymiseen, tai askelten huomiotta jättämiseen.

Kokonaisuudessaan kuljettiin 1800 askelta, joista puhelin mittasi 1903 ja askelmittari laski 1762. Puhelimen sisäinen askelmittari ilmoitti noin 5,72% enemmän askelia kuin todellisuudessa käveltiin, kun taas askelmittari ilmoitti 2,11% todellista vähemmän. Povitasku oli askelmittarille haastavin paikka, kun taas puhelimen sisäinen mittari vaikutti tunnistavan nämä askeleet parhaiten.

5 KESKUSTELU

Povitaskussa puhelin pääsi liikkumaan vapaammin takin mukana, kun taas etu- ja takataskussa puhelin myötäili tarkemmin kehon liikkeitä. Tästä voidaan päätellä, että parhaimpiin tuloksiin päästään, kun puhelin on mahdollisimman kehonmyötäisessä taskussa. Näin vähennetään puhelimen sattumanvaraista liikkumista ja saadaan tarkemmin informaatiota askeleeseen liittyvistä kiihtyvyyksistä.

Sovellus pääsi suhteellisesti parempiin tuloksiin kuin puhelimen oma askelmittari. Huomattavaa on kuitenkin se, että sovelluksen laskemat askelmäärät olivat aina pienempiä, tai yhtä suuria kuin todelliset askelmäärät. Todellista askelmäärää suurempiin lukemiin pääseminen edellyttäisi askeleen virheellistä tunnistamista. Tämän tapahtuessa on todennäköistä, että sovelluksen ilmoittama askelmäärä on huomattavasti suurempi todelliseen verrattuna. Tämä mahdollistaa käyttäjän kriittisen ajattelun sovellusta käyttäessä, jonka seurauksena käyttäjä voi analysoida viimeisimmän datan uudestaan ja mahdollisesti saada tulokseksi realistisemman askelmäärän.

Sovelluksen analysointivaihe hidastuu huomattavasti askelmäärän kasvaessa yli 300 askeleen. Tämä johtuu suurelta osin datan jatkuvasta skaalaamisesta jokaisen mallin vertailun yhteydessä. Tämä voitaisiin kiertää käyttämällä parempaa skaalausfunktiota. Esimerkiksi jonkinlainen muunneltu sigmoid-funktio voisi olla mahdollinen ratkaisu ongelmaan. Pienillä kiihtyvyysarvoilla skaalaus tapahtuisi lähes lineaarisesti, mutta kiihtyvyyksien kasvaessa oletettua suuremmiksi, funktio saavuttaisi maksimiarvonsa. Esimerkiksi puhelimen osuminen johonkin liikkumisen aikana voi aiheuttaa tällaisia äkillisiä suuria kiihtyvyyksiä.

6 JOHTOPÄÄTÖKSET

Suurimmat haasteet työn aikana kohdattiin ohjelman toteutuksen aikana. Tämä johtuu ohjelmoijan rajoittuneesta kokemuksesta sovellusten luomisessa Android-ympäristöön. Näin ollen esimerkiksi sensoriarvojen piirtämistä kuvaajaan ei ole toteutettu optimaalisesti. Tämä ei kuitenkaan vaikuta askelmittarin askeleentunnistustarkkuuteen, ja sovellus onkin tältä kannalta kilpailukykyinen muiden askelmittareiden kanssa. Suurimpana puutteena sovelluksessa on laskentanopeus, joka heikkenee nopeasti mittauksen pidentyessä ajallisesti.

Jatkokehityksenä sovelluksen laskentanopeutta tulisi saada nostettua. Yksi mahdollisuus olisi testata näytteenottonopeuden vaikutusta laskentatarkkuuteen. Datan skaalaus ja askelten tunnistaminen nopeutuu, jos malli sisältää vähemmän datapisteitä. Myös gyroskooppidatan merkitystä laskentatarkkuuteen voidaan tarkastella, sillä gyroskooppidatan jättäminen pois analyysistä puolittaisi analysoitavien pisteiden määrän. Myös uusia skaalausfunktioita voitaisiin kokeilla. Hyvällä skaalausfunktiolla voitaisiin skaalata koko data yhdellä kertaa, vaikuttamatta askelten tunnistustarkkuuteen negatiivisesti.

Lähteet

[1] C. Sidney Burrus. *Introduction to Wavelets and Wavelet Transforms: A Primer*. Pearson, elokuu 1997. ISBN: 0134896009.

[2] Google Developers. *Android KitKat*. 2019. URL: https://developer.android.com/about/versions/kitkat.html#44-sensors (viitattu 20. 02. 2020).

[3] Google Developers. *Documentation for app developers*. 2019. URL: https://developer.android.com/guide/topics/sensors/sensors_overview.html (viitattu 20. 02. 2020).

[4] Sebely Pal et al. "Using pedometers to increase physical activity in overweight and obese women: a pilot study". *BMC Public Health* vol. 9 no. 1 (elokuu 2009). DOI: 10.1186/1471-2458-9-309.

[5] Mustajoki Pertti. *Liikunta ja painonhallinta.* 2019. URL: http://www.terveyskirjasto.fi/terveyskirjasto/tk.koti?p_artikkeli=dlk01005 (viitattu 28. 02. 2020).

[6] Wellness IN the Rockies. *Steps to distance conversion chart*. URL: https://www.uwyo.edu/wintherockies_edur/win%20steps/coordinator%20info/step%20conversions.pdf (viitattu 01. 04. 2020).

[7] TechCrunch. *Kotlin is now Google's preferred language for Android app development*. 2019. URL: https://techcrunch.com/2019/05/07/kotlin-is-now-googles-preferred-language-for-android-app-development/ (viitattu 20. 02. 2020).