LATVIJAS UNIVERSITĀTE
DATORIKAS FAKULTĀTE

AUTOVADĪŠANAS MANEVRU NOTEIKŠANA IZMANTOJOT VIEDTĀLRUNI

MAĢISTRA DARBS

Autors: **Toms Dembovskis**
Stud. apl. nr. td09013
Darba vadītājs: Dr.dat. Jānis Zuters

RĪGA 2015

ANOTĀCIJA

Automašīnas ir mūsdienu izplatītākais transporta līdzeklis, ar ko ir saistīta gan mūsu ikdiena, gan neskaitāmas uzņēmējdarbības jomas. Diemžēl ir ļoti maz iespēju iegūt informāciju par to, kā automašīnas tiek izmantotas. Viedtālruņi mūsdienās ir ļoti plaši izplatīti un tie ir aprīkoti ar sensoriem, kas var tikt izmantoti šīs problēmas risināšanā.

Šajā darbā tiek izpētīta iespēja izmantot autovadītāja viedtālruņa sensoru iegūto informāciju, lai automatizētā veidā noteiktu vadītāja veiktos manevrus autovadīšanas laikā. Darba ietvaros tiek apzināti līdzīgi pētījumi un risinājumi, kā arī apskatīti un analizēti viedtālruņos sastopamie sensori un pārbaudīti šo sensoru mērījumi autovadīšanas laikā. Darba rezultātā tiek izstrādāts konceptuāls risinājums un arī risinājumam atbilstošs programmatūras prototips, kas spējīgs atpazīt veiktos manevrus.

Atslēgvārdi: Viedtālrunis, iOS, Autovadīšana, manevru noteikšana

ABSTRACT

Cars are a big part of our society. Many areas of our everyday lives, businesses and infrastructure revolve around driving. However the possibilities for getting information about how cars are used, are very limited. Smartphones nowadays are very common and equipped with an array of sensors, that can be used to solve the problem.

This thesis explores the possibility to use the data acquired from the drivers phone to detect and analyze, maneuvers performed by the driver. As part of this thesis similar studies and solutions where examined as well as the sensors that are available in smartphones and the output data that these sensors provide during driving. As a result of the thesis a conceptual solution was designed and software prototype was created according to the solution. The prototype was tested in real driving conditions.

Keywords: Smartphone, iOS, Driving, maneuver detection

AUTOREFERĀTS

Šī darba ietvaros tika izpētīta automātiska autovadīšanas manevru atpazīšana izmantojot viedtālruni. Šīs izpētes ietvaros autors ir:

- Veicis vairāku akadēmisko darbu analīzi un izpēti, kas saistīti ar autovadīšanas analīzi
- Izpētījis šobrīd pieejamo programmatūru, kas veic autovadīšanas analīzi
- Izpētījis viedtālruņos pieejamos sensorus, to darbības principus un pielietojumu.
- Veicis praktisku eksperimentu par autovadīšanas laikā veiktu manevru pazīmēm un attēlošanos viedtālruņa sensoru mērījumos.
- Izstrādājis konceptuālu risinājumu, priekš automatizētas manevru atpazīšanas realizēšanas, kas satur norādes kā realizēt manevru sākuma noteikšanu, manevru atpazīšanas algoritmu, atpazīšanas rezultāta novērtēšanu un manevru glabāšanu.
- Vadoties pēc konceptuālā risinājuma izstrādājis inovatīvu un unikālu mobilo lietojumprogrammu, kas veic automātisku manevru atpazīšanu autovadīšanas laikā, un kuru var pielāgot dažādu manevru atpazīšanai.
- Veicis izstrādātās lietojumprogrammas testēšanu autovadīšanas laikā un pārliecinājies, ka lietotne var veiksmīgi tikt izmantota manevru atpazīšanā.

SATURS

APZĪMĒJUMU SARAKSTS

iOS –Kompānijas Apple izstrādātā mobilā operētājsistēma, kas tiek izmantota šīs kompānijas ražotajās mobilajās ierīcēs.

Android - Google kompānijas izstrādātā mobilā operētājsistēma, kas atbalsta plašu ierīču klāstu.

SDK (Software developer's kit) - Palīgprogrammu (rutīnu) kopa, kas lietojumprogrammu izstrādātājam atvieglo programmu veidošanu, ievērojot konkrētās to darbības vides īpatnības (piemēram, grafisko lietotāja saskarni, operētājsistēmu, datu bāzu pārvaldības sistēmu u.c.).

Sensora troksnis – Viedtālruņa sensoru datos novērojamas mērījumu svārstības, kas nav saistītas ar automašīnas veiktajiem manevriem. Šīs mērījumu svārstības var rasties dažādu cēloņu rezultātā, piemēram, dēļ automašīnas motora vibrācijas, ceļa seguma nelīdzenumiem, pašu sensoru neprecizitātes.

CAN kopne – automašīnās izmantotas kopnes standarts, kura mērķis ir ļaut kontrolieriem un citām ierīcēm sazināties savā starpā neizmantojot saimniekdatoru.

API (Application Program Interface) - Lietojumprogrammu saskarne. Lietojumprocesos izmantojama pilna operētājsistēmas funkciju specifikācija, kā arī šo funkciju izmantošanas procedūru apraksts. Tīkla operētājsistēmās ar saskarni API tiek definēta standarta metode, kas lietojumiem nodrošina visu tīkla iespēju izmantošanu.

DTW (Dynamic time warping) – laikrindu analīzē šis algoritms domāts, lai izmērītu līdzību starp divām no laika atkarīgām vērtību virknēm, kas var variēt gan laikā, gan ātrumā.

Eiklīda attālums - attālums starp diviem punktiem daudzdimensiju telpā, kas atbilst to savienojošā nogriežņa garumam.

JSON (JavaScript Object Notation) –programmēšanas valodai nepiesaistīts datu apmaiņas formāts, kas izmanto cilvēkiem lasāmu tekstu, lai veidotu datu objektus, kas sastāv no atribūtu un vērtību pāriem.

IEVADS

Automašīnas ir mūsdienu izplatītākais transporta līdzeklis, ar ko ir saistītas daudzas uzņēmējdarbības jomas. Lai gan automašīnu vadīšana ir fundamentāla ekonomikas, uzņēmējdarbības un cilvēku ikdienas daļa, pastāv maz iespēju kā automatizētā veidā iegūt informāciju, par to kā automašīnas tiek izmantotas, neskatoties uz to, ka daudzās nozarēs, šī informācija varētu būt ļoti noderīga.

Viedtālruņi ir salīdzinoši nesena tehnoloģiskā inovācija, taču to izplatība ir bijusi ļoti strauja, un tie ir kļuvuši par daudzu cilvēku neatņemamu ikdienas sastāvdaļu. Strauja ir bijusi arī šo ierīču tehnoloģiskā attīstība. Mūsdienu viedtālruņi ir aprīkoti ar dažādiem sensoriem, kas var tikt izmantoti daudz un dažādos inovatīvos veidos.

Viedtālruņu sensoru nodrošinājums un plašā izplatība, padara tos par potenciāli lieliskiem rīkiem, lai analizētu autovadīšanu. Viedtālruņu izmantošanai, lai veiktu autovadītāja darbību analīzi, liela priekšrocība ir tas, ka nav nepieciešama infrastruktūras un tehnoloģiskā nodrošinājuma ieviešana, jo nepieciešamie sensori jau ir sastopami mūsdienu mobilajās ierīcēs. Sensoru nodrošinājums viedtālruņos, dod iespēju analizēt datus, kas varētu tikt izmantoti autovadīšanas analīzē, taču, problēma ir uzticamā un automatizētā veidā no šiem datiem iegūt informāciju, par konkrētām darbībām ko veic autovadītājs. Šī problēma tiks izpētīta šajā darbā.

Šī darba ietveros tiks izpētīts, izplānots un praktiski realizēts risinājums, kas, izmantojot viedtālrunī pieejamos sensoru datus, reālā laikā, būtu spējīgs noteikt dažādus autovadītāja veiktos manevrus. Risinājums konceptuālā veidā tiek plānots kā dažādās nozarēs un pielietojumos izmantojama komponente, kas nav centrēta ap konkrētu atpazīto manevru pielietošanu praktiskā situācijā, bet kuras uzdevums ir no zema līmeņa sensoru datiem iegūt informāciju, par to kādi autovadīšanas manevri ir notikuši. Šāda komponente varētu tikt integrēta un izmantota citās sistēmās un programmatūrā, tādējādi, ļaujot šīm sistēmām abstrahēties no zema līmeņa sensora datu analīzes.

Darba ietvaros vispirms tiek veikta esošās situācijas izpēte un līdzīgu vai saistītu pētījumu un publikāciju analīze. Tiek apskatīta arī jau eksistējoša programmatūra vai risinājumi, kas veic autovadīšanas analīzi. Tālāk seko viedtālruņos sastopamo sensoru analīze un izpēte. Lai noteiktu kādu sensoru dati varētu tikt izmantoti konkrētu manevru noteikšanai, tiek veikta iOS SDK iekļauto ietvaru analīze, kas nodrošina programmatūras integrāciju ar sensoriem, kā arī veikts praktisks eksperiments, kurā ar automašīnu tiek veikti dažādi manevri, un analizēti manevru laikā iegūtie sensoru mērījumi. Šie mērījumi tiek apkopoti un

izdarīti secinājumi, par to kādi mērījumi liecina par kādiem manevriem.

Balstoties uz veikto teorētisko izpēti un eksperimenta rezultātiem, tiek izstrādāts vispārīgs risinājums šajā darbā izvirzītajai problēmai. Risinājums tiek izstrādāts balstoties uz vispārīgām prasībām un ierobežojumiem, kas izvirzīti, lai atrisinātu šajā darbā izvirzīto problēmu. Risinājumā ir izpētīts un definēts, gan kā būtu veicama manevru atpazīšana, gan arī citas komponentes un aspekti, kas ir svarīgi, lai varētu realizēt, automatizētu manevru atpazīšanas sistēmu atbilstoši risinājumā definētajai metodoloģijai, kura būtu pietiekami dinamiska, lai to varētu izmantot dažādos lietojumu gadījumos un darījumprocesos un lai to, kā programmatūras komponenti, varētu integrēt citās sistēmās.

Lai pārbaudītu piedāvāto risinājumu, darba ietvaros ir izstrādāts programmatūras prototips atbilstoši piedāvātajam risinājumam. Programmatūra izstrādāta izmantojot Objective-C programmēšanas valodu priekš viedtālruņiem, kas darbojas ar iOS operāciju sistēmu. Izstrādātās programmatūras darbība ir praktiski pārbaudīta autovadīšanas laikā un iegūtie rezultāti apkopoti un analizēti, lai izdarītu secinājumus par darbā piedāvāto risinājumu.

1. ESOŠIE PĒTĪJUMI UN RISINĀJUMI

Lai gūtu priekšstatu par to, kas jau ir darīts un izpētīts darbā izvirzītās problēmas risinājumam tiek veikta detalizēta esošo pētījumu un risinājumu analīze.

1.1 Esošie pētījumi

Autovadīšanas paradumu un stila noteikšana var tikt izmantota vairākās nozarēs un jomās, tādēļ ir atrodami vairāki akadēmiski pētījumi un eksperimenti, kas analizēti šajā nodaļā.

Pētījumā ar nosaukumu "Safe Driving Using Mobile Phones" veikta līdzīga izpēte kāda plānota šajā darbā [1.]. Šajā pētījuma tika izmantots Android viedtālruņa akselerometra dati. Tika analizēta automašīnas bremzēšana un paātrinājums, braukšanas joslu maiņa, kā arī ceļa kvalitāte. Pētījumā tika pārbaudītas vairākas viedtālruņa novietošanas vietas automašīnā un izvēlēta vislabākais novietojums priekš konkrētā mērījuma, kā arī ierīce tika stingri piestiprināta pie automašīnas virsmas. Analizētie manevri tika veikti gan strauji, gan līgani un pētījumā bija labi uzskatāma akselerometra mērījumu izmaiņa atkarībā no manevra veikšanas straujuma. Pētījumā uzrādītie sensora mērījumu precizitāte un tas cik nepārprotami sensora datos iespējams atrast manevrus, ir pārsteidzoša, un tā iemesls visticamāk ir ļoti stabila un mērījumam veiksmīga ierīces novietojuma izvēle. Šajā darbā tiks pētīts līdzīgu tehnoloģiju pielietojums reālākā dzīves situācijas, kad telefons nav stingri piestiprināts, pie automašīnas korpusa, un ir paredzamas lielākās sensora rādījumu izmaiņas ārpus manevriem.

Derick A. Johnson un Mohan M. Trivedi veiktajā pētījumā "Driving Style Recognition Using a Smartphone as a Sensor Platform"[2.] tika izmantots stingri nostiprināta iPhone 4 ierīce. Pētījumā tika izmantota informācija no žiroskopa, akselerometra un magnetometra. Pētījumā tika savākti dati un izveidotas vairāku manevru bāzlīnijas. Autori pielāgoja DTW algoritmu, kurš domāts divu signālu salīdzināšanai. Salīdzinot sastopamos manevrus ar bāzlīnijas manevriem izmantojot šo algoritmu autoriem veiksmīgi izdevās noteikt dažāda veida manevrus. Autori secināja, ka ar šādu pieeju ir sarežģīti atšķirt apgriešanās manevru no pagrieziena manevra. Sistēma nespēja noteikt līganus pārkārtošanās manevrus, jo šo manevru rezultējošais spēks un rotācija bija pārāk mazs, lai to atšķirtu no sensoru trokšņa.

Citā pētījumā ar nosaukumu "Automatic Accelerometer Reorientation for Driving Event Detection Using Smartphone"[3.] tiek apskatīta pieeja pārorientēt sensoru atgrieztos datus citā koordinātu sistēmā, kas atbilst nevis telefona stāvoklim, bet automašīnas virzienam. Visi iepriekš apskatītie pētījumi tika veidoti balstoties uz to, ka viedtālruņa stāvoklis relatīvi pret

automašīnu ir noteikts un zināms – tas nevar tikt mainīts. Taču šajā pētījumā izmantojot akselerometra, magnetometra un GPS sensora informāciju, autori ir spējīgi aprēķināt telefona stāvokli attiecībā pret zemi un automašīnas virzienu un no šis informācijas autori pārveidoja akselerometra trīsdimensionālo koordinātu sistēmu atbilstoši gravitācijai un automašīnas kustības virzienam. Šāda transformācija ļauj telefonam atrasties iepriekš nedefinētā stāvoklī un analizējot akselerometra datus noteikt dažādus manevrus. Autori savu izveidoto pieeju pārbaudīja mēģinot noteikt bremzēšanas manevru un braukšanas joslu maiņu kreisajā virzienā. Manevri tika noteikti manuāli veidojot grafikus no sensoru rezultātiem, kas tika iegūti ar iOS lietotni. Šī pieeja pārorientēt akselerometra koordinātu sistēmu noteikti ir nepieciešama reālā rīkā, iespējams, ka ir pietiekami šādu koordinātu sistēmas koriģēšanu veikt brauciena sākumā un uzskatīt ka ierīces stāvoklim brauciena laikā nevajadzētu mainīties.

Pētījumā ar nosaukumu “Leveraging Sensor Information from Portable Devices towards Automatic Driving Maneuver Recognition”[4.] autori veica manevru atpazīšanu izmantojot gan automašīnas iekšējo sistēmu datus, gan ar viedtālruni iegūtus datus. Automašīnas iekšējo sistēmu dati tika iegūti tiešā veida no automašīnas iegultajām sistēmām. Mūsdienu automašīnas iekļauj vairākas sistēmas, kas uzrauga un kontrolē lielu daļu no mašīnas funkcijām. Šo sistēmu datus var nolasīt izmantojot CAN-kopni, kas ir sastopama vairumā automašīnu. Izmantojot šo kopni no automašīnas var iegūt datus par šībrīža ātrumu, stūrēšanas leņķi, akselerācijas un bremzēšanas pedāļa stāvokli, kā arī citus datus par vispārīgo automašīnas stāvokli. Pētījuma rezultātā, tika secināts ka izmantojot viedtālruņa sensoru datus, autovadīšanas manevru atpazīšana bija par 15% precīzāka.

Pēc līdzīgu pētījumu apskates un izpētes var secināt, ka autovadīšanas manevru atpazīšanas jomā ir veikti ievērojami pētījumi, un šo pētījumu metodes varētu tikt pielietotas universālāka risinājuma izstrādē. Pētījumos atspoguļojas dažādas manevru atpazīšanas precizitātes, taču nevienā no aplūkotajām publikācijām, netika sasniegts secinājums ka manevru atpazīšana izmantojot viedtālruni nav iespējama. Ne visās publikācijas tika detalizēti aprakstīta manevru atpazīšanas metodoloģija, paņēmieni, un izmantotie ieejas dati, taču pēc publikāciju analīzes var secināt, ka manevru atpazīšanai būtu jābūt iespējamai. Risinājums, kurā izmantoti elementi un secinājumi no šo autoru darbiem, varētu būt pietiekami robusts, lai to izmantotu ne tikai pēc stingri noteiktiem lietošanas nosacījumiem, bet arī ārpus pētnieku eksperimentiem.

1.2 Esošā programmatūra autovadīšanas manevru noteikšanai

Neskatoties uz to, ka šajā jomā ir veikti daudz pētījumi un pārbaudīti vairāki paņēmieni, lai automatizētu autovadīšanas manevru noteikšanu, gatava programmatūra, kas varētu tikt izmantota, lai analizētu tieši vadītāja veiktos manevrus, nebija atrodama. Veicot eksistējošu risinājumu apzināšanu tika meklēti, gan atvērtā koda risinājumi, gan arī maksas risinājumi. Darba autors nespēja atrast nevienu gatavu programmatūru vai pieejamu programmatūras komponenti, kam būtu pieejams un analizējams pirmkods, vai arī detalizēts sistēmas apraksts.

Izpētot Android Play Store un iOS App Store lietotņu veikalus tika atrasti daži risinājumi, kas veic autovadīšanas analīzi, bet neviens no šiem risinājumiem tieši neveic darbā izvirzīto prasību, konkrētu autovadīšanas manevru atpazīšanu. Kompānija StateFarm ir izveidojusi lietotni "Driver Feedback", autovadīšanas laikā analizē to cik strauji lietotājs veic pagriezienus, paātrinājumu un bremzēšanu. Lietotājā vadīšana tiek novērtēta ar rezultātu no 1 līdz 100 šajās 3 kategorijās (skatīt attēlu 1.1.). Diemžēl lietotne ir pieejama tikai ASV tādēļ to nebija iespējams pārbaudīt praksē, bet vadoties pēc apraksta, lietotne neveic konkrētu manevru atpazīšanu, bet tikai akselerometra mērījumu analīzi [5.]. Ļoti līdzīga lietotne ar tādu pašu nosaukumu "Driver Feedback", bet no cita izstrādātāja, pēc apraksta veic praktiski identisku akselerometra datu analīzi un autovadīšanas vērtēšanu, tikai papildus tiek vērtēts braukšanas ātrums [6.].

1.1. att.: **Driver feedback lietotne [5.]**

Plašāk izplatītas, gan priekš Android, gan iOS viedtālruņiem ir lietotnes, kas autovadīšanas laikā izmanto kameru, lai analizētu autovadīšanu. Šāda tipa lietotnes informē autovadītāju par to cik precīzi tiek ievērota braukšanas josla, un vai tiek ievērota pietiekoša distance no priekšā esošajiem transporta līdzekļiem. Šāda tipa lietotnes piedāvā arī iespēju brīdināt autovadītāju par iespējamu sadursmi ar priekšā esošu objektu [7., 8.].

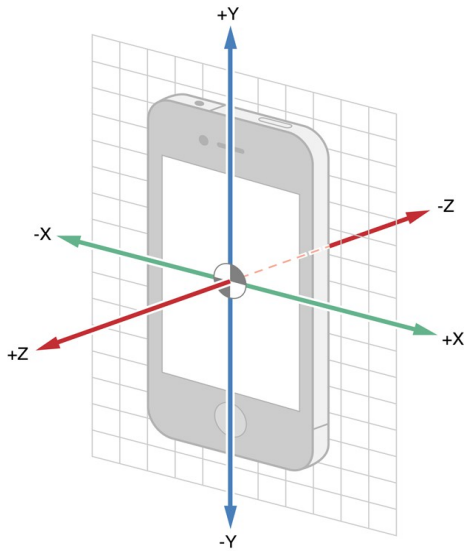
Izpētot un pārskatot gatavās lietotnes un pieejamo programmatūru, darba autoram, neizdevās atrast nevienu realizētu risinājumu, kas veic autovadīšanas manevru atpazīšanu. Tika atrastas vairākas lietojumprogrammas, kas autovadīšanas analīzi veic dažādos veidos, taču neviena aplūkotā lietojumprogramma, nerisina šajā darbā izvirzīto problēmu – autovadīšanas manevru noteikšanu.

2. VIEDTĀLRUŅU SENSORI

Šajā nodaļā tiks apskatīti un izpētīti viedtālruņos sastopamie sensori, kuri varētu tikt izmantoti autovadīšanas manevru noteikšanā. Šie sensori ir – akselerometrs, žiroskops, digitālais kompass un GPS uztvērējs.

2.1 Akselerometrs

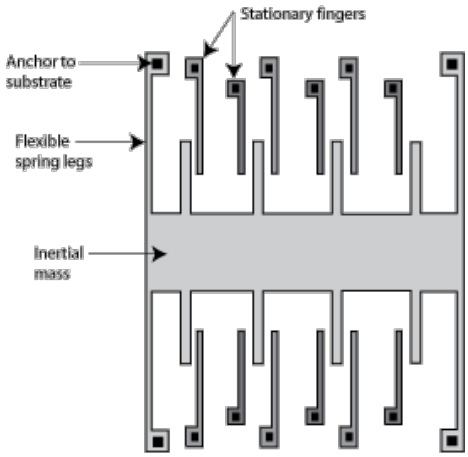
Akselerometrs ir ierīce, kas mēra spēkus kāds iedarbojas uz sensoru konkrētā virzienā. Spēki var būt statiski (zemes gravitācijas spēks) vai dinamiski (spēks kustības rezultātā). Svarīgs šo sensoru raksturlielums ir mērīšanas asu skaits. Sensors mēra spēka lielumu, kāds uz to iedarbojas konkrētā asī, lai efektīvi analizētu sensoru trīsdimensionālā telpā tam ir vienlaicīgi jāveic mērījumi 3 asīs.



2.1. att. **Akselerometra mērījumu asis [9.]**

Visi Apple iPhone modeļi bija aprīkoti 3 asu akselerometru, un mūsdienās šāds sensors ir sastopams vairumā viedtālruņu [10.].

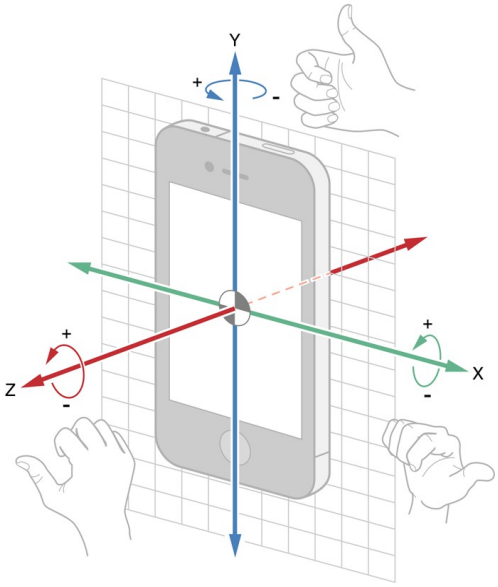
Vairums akselerometru ir mikro-elektromehāniski sensori (MEMS). Vispārīgais darbības princips MEMS akselerometros ir izmērīt neliela svara novirzi no miera stāvokļa. Šie sensori visbiežāk tiek veidoti uz silīcija mikroshēmas, kurā iegravēts neliels svars, kuru notur plāni un elastīgi savienojumi ar kopējo virsmu. Inerces dēļ, kad visa virsma tiek izkustināta, svars arī tiek izkustināts relatīvi pret kopējo virsmu. Svara novirzi no sākuma stāvokļa var izmērīt, un izrēķināt spēku kāds iedarbojies uz ierīci [11.].



2.2. att.: **Akselerometra uzbūve[12.]**

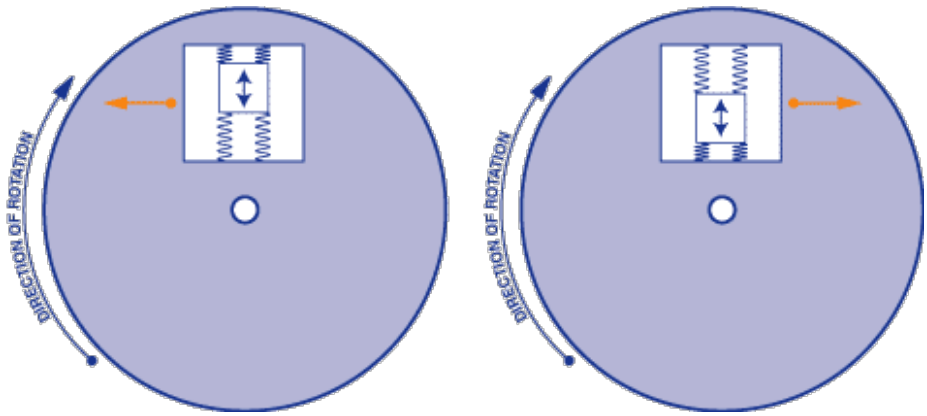
2.2 Žiroskops

Žiroskops ir ierīce, kas mēra vai stabilizē orientāciju, balstoties uz leņķiskā paātrinājuma principiem. Žiroskops pretēji akselerometram ir spējīgs izmērīt nevis spēku, bet gan rotāciju ap kādu no asīm. Viedtālruņos izmantotie žiroskopi arī darbojas ar 3 mērījumu asīm.



2.3. att.: **Žiroskopa mērījumu asis [9.]**

Viedtālruņos tiek izmantoti mikro-elektromehāniski (MEMS) žiroskopi.

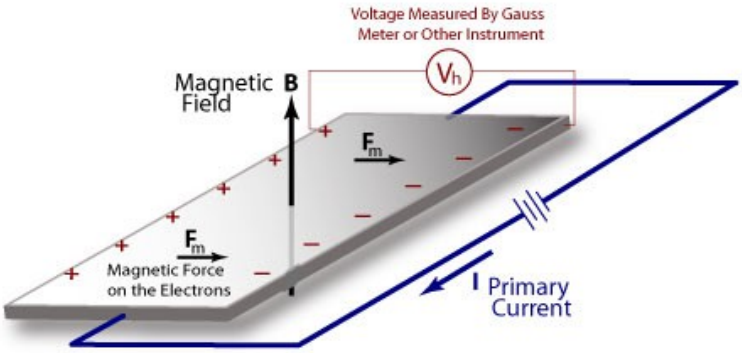


2.4. att.: **Žiroskopa darbības princips [13.]**

Žiroskopa sensors iekš MEMS ir niecīgs – 1 līdz 100 mikrometri. Kad žiroskops tiek rotēts, mazas vibrējošs svars tiek izkustināts un tiek izmainīts tā leņķiskais ātrums. Šī kustība tiek pārveidota elektriskā signālā ko var uztvert un izmērīt [13.].

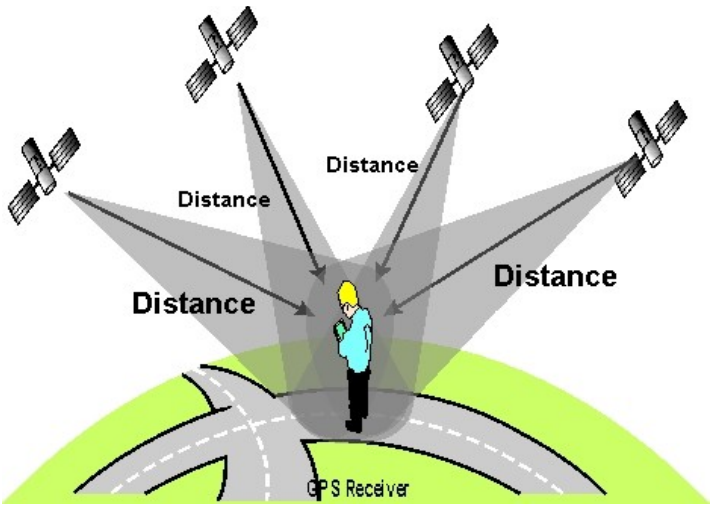
2.3 Digitālais kompass

Digitālais kompass, kas var arī tikt saukts par magnetometru ir sensors, kas ir spējīgs uztvert un izmērīt ārēju magnētisko lauku virzienu un spēku. Šāds digitālais kompass, izmantojot zemes magnētisko lauku, ļauj ar augstu precizitāti noteikt virzienu, kādā ir pavērsta ierīce, taču ja ierīces tuvumā sastopami citi magnētiskie lauki, tas var traucēt sensora darbībai [14.]. Magnetometri var tikt veidoti pēc dažādām metodēm un darboties pēc dažādiem principiem, taču vispopulārākais un mobilajās ierīcēs tipiski sastopamais ir magnetometrs, kas balstīts uz Halla efektu. Šie sensori būvēti pēc principa, ka plānā metāliskā plāksnē, kas novietota perpendikulāri magnētiskajam laukam, var novērot sprieguma izmaiņas. Halla efekta sensori, patērē maz elektroenerģijas, un ir mazi pēc izmēra, tādēļ tie ir piemēroti izmantošanai viedtālruņos [15.].



2.5. att.: **Magnetometra darbības princips[16.]**

2.4 GPS uztvērējs



2.6. att.: **GPS darbības princips [18.]**

Globālās pozicionēšanas uztvērēj saņemot informāciju no vismaz 4 satelītiem dažādās pozīcijās, pēc laika kāds signālam bija nepieciešams, lai sasniegtu uztvērēju, var izrēķināt attālumu no uztvērēja līdz satelītiem. Satelītu raidītie signāli satur informāciju par satelīta atrašanās vietu un laiku, kad signāls sūtīts [17.].

3. MANEVRU IZPĒTE UN NOTEIKŠANA

Pirms tiek izstrādāts risinājums, kas automātiski nosaka autovadīšanas manevrus nepieciešams analizēt sensoru datus, kādi tiek iegūti veicot manevrus automašīnā, kā arī detalizēti izpētīt iespējas un ierobežojumus kādi piemīt izmantotajai ierīcei un platformai.

3.1 Platformas izvēle

Pētījumā netiks apskatīti un izmantoti Android operētājsistēmas viedtālruņi. Šāda izvēlē ir izdarīta, lai izvairītos no ļoti lielās ierīču un operētājsistēmu versiju dažādības Android viedtālruņos. Pētījumā par Android ierīču dažādību 9 mēnešu laikā, 2014 gadā populāra Android lietotne "OpenSignal" tika uzstādīta uz 18798 dažādām Android ierīcēm no dažādiem ražotājiem [19.]. Šīm ierīcēm var būt krasi atšķirīgs sensoru aprīkojums, kā arī pašu sensoru kvalitāte un izejošās informācijas precizitāte un uzticamība. Lai šajā pētījumā izvairītos no nekonsekventiem un neviennozīmīgiem sensoru mērījumiem, pētījums tiks veikts izmantojot iOS operētājsistēmas ierīces. IOS ierīcēs nav jāuztraucas par atšķirīgu operētājsistēmas uzvedību uz dažādām ierīcēm, jo gan ierīces gan operētājsistēmu ražo viena un tā pati kompānija, kā arī, ierīču dažādība ir zema. Analīzes un izpētes fāzē tiks izmantota iPhone 6 ierīce, kas ir šobrīd jaunākais iOS viedtālrunis.

3.2 Sensoru API

Programmatūru plānots izstrādāt izmantojot Objective-C programmēšanas valodu un iOS SDK iekļautos ietvarus un programmatūras saskarni. Šajā nodaļā apskatīsim iOS SDK iekļautos ietvarus, kas nodrošina iepriekš apskatīto sensoru izmantošanu.

3.2.1 Core Motion ietvars

Sākot ar iOS 4 versiju SDK tiek iekļauta Core Motion bibliotēka. Šī bibliotēka ļauj piekļūt ierīces sensoru datiem, gan neapstrādātā, gan apstrādātā veidā. Izmantojot klasi CMMotionManager var pierakstīties uz paziņojumiem par sensoru stāvokli ar noteiktu laika intervālu. Katram sensora datu veidam ir atbilstoša klase, kas sagrupē līdzīgo informāciju[10.]. Klases kuras varētu potenciāli tikt izmantotas šajā pētījumā tiek detalizēti apskatītas:

- CMGyroData klase satur 3 double tipa skaitļus, kas atspoguļo ierīces rotācijas ātrumu ap x,y,z asīm. Šī vērtība ir izteikta radiānos sekundē. Šie dati nav apstrādāti vai filtrēti

un tieši atspoguļo konkrēto žiroskopa stāvokli un tajos var novērot sensora troksni [20.].

- CMAccelerometerData satur 3 double tipa skaitļus, kas atspoguļo ierīces paātrinājumu pa x,y,z asīm. Šī vērtība ir izteikta kā attiecība pret zemes gravitācijas konstanti 9.81 m/s^2. Ja ierīce atrodas nekustīgā stāvokli, visu skaitļu kopējā summa būs 1, kas nozīmē to ka uz akselerometru iedarbojas vienīgi zemes gravitācijas spēks. Ierīcei atrodoties guļus stāvokli gravitācijas spēks iedarbosies tikai uz Z asi. Šie dati nav apstrādāti vai filtrēti un tieši atspoguļo akselerometra stāvokli , tādēļ tajos var novērot sensora troksni [21.].
- CMDeviceMotion apkopo mērījumus no vairākiem sensoriem un, veicot pārveidojumus un aprēķinus, sniedz informāciju par ierīces stāvokli un kustību [10.]. Šī klase satur vairākas vairākus noderīgus mērījumus:
    - rotationRate (rotācijas ātrums) atgriež informāciju par ierīces rotācijas ātrumu, tādā pašā veidā, kā CMGyroData klase, taču šīs vērtības ir precizētas un koriģētas izmantojot CoreMotion algoritmus [20.].
    - attitude (ierīces stāvoklis) atgriež informāciju par ierīces stāvokli trīsdimensionālā telpā attiecībā pret dažādiem atskaites objektiem. Ierīces stāvoklis katrā asī ir izteikts radiānos attiecībā pret sākuma asi. Iespējami vairāki atskaites objektu iestatījumi, piemēram, Z ass vertikāli un X ass uz magnētisko ziemeļpolu, vai Z ass vertikāli un X ass relatīvi pret ierīces sākuma stāvokli, u.c. Tas nozīmē, ka lai iegūtu šo mērījumu, CoreMotion ietvaram jāizmanto gan žiroskops, gan akselerometrs, gan magnetometrs [21.].
    - gravity (gravitācija) atgriež informāciju par gravitācijas paātrinājuma vektoru relatīvi pret ierīces atskaites stāvokli 3 asīs. Šī vērtība ir apstrādāta un filtrēta, tajā ir ļoti minimāls sensoru troksnis. Zemes gravitācijas spēks tiek attēlots šajā mērījumā.
    - userAcceleration (lietotāja paātrinājums) atgriež informāciju par paātrinājumu kādu ārējie spēki rada uz ierīci 3 asīs neņemot vērā zemes gravitāciju. Ja ierīce atrodas nekustīgā stāvoklī, visu atgriezto skaitļu vērtības būs ļoti tuvu 0. Šis mērījums ir koriģēts, lai no tā atņemtu zemes gravitāciju, taču tas nav filtrēts, un tajā var novērot sensora troksni [10.].

3.2.2 Core Location ietvars

Core Location ietvars ļauj piekļūt GPS sensora datiem, kā arī magnetometra datiem. Izmantojot klasi CLLocationManager var pierakstīties uz paziņojumiem par ierīces pārvietošanos ņemot datus no GPS sensora, kā arī par virzienu kādā pavērsta ierīce izmantojot magnetometra datus. Pārvietošanās virziena informācija varētu būt nepieciešama manevru noteikšanā, gadījumos, kad ierīce nav pavērsta tajā pašā virzienā, kurā notiek pārvietošanās. CLHeading klase satur visu informāciju par ierīces pavērsto virzienu konkrētā laika momentā. Magnetometra neapstrādātā sensora informācija izteikta kā ierīces šībrīža novirzījums no x,y, z asīm un tas ir izteikts mikroteslos. Šīs asis atbilst magnētiskajam lauka virzienam, kuru dotajā brīdi uztver ierīce. Šī klase satur arī ierīces šībrīža novirzījumu no magnētiskā ziemeļpola, kā arī ģeogrāfiskā ziemeļpola, kas izteikts grādos – 0 nozīmē ka ierīce pavērsta ziemeļu virzienā, 90 austrumu virzienā, utt.. [22.].

3.3 Manevru analizēšana

Lai autovadīšanas manevru noteikšanu varētu automatizēt, vispirms nepieciešams identificēt, kādus datus nepieciešams analizēt, konkrētiem manevriem, un kā katrs meklētais manevrs tiek modelēts sensoru rādījumos. Lai iegūtu šādus datus, tika veikts praktisks eksperiments.

3.3.1 Eksperimenta apraksts

Analizējot manevrus tika pievērsta uzmanība atšķirībām kādas sastopamas, kad manevrs tiek veikts strauji vai līgani, tādējādi cenšoties diferencēt bīstamu manevru no droša. Eksperimentā tika izmantota automašīna Volvo ar dīzeļa dzinēju. Datu iegūšanai tika izmantota lietotne SensorLog[23.]. Ar šīs lietotnes palīdzību var ierakstīt un saglabāt visu iOS lietotnē pieejamo sensoru atgrieztos datus. Pētījuma aprakstā tiks runāts par konkrētām vērtībām, kas sakrīt ar iepriekš aprakstītajām, no CoreMotion ietvara pieejamajām vērtībām. Eksperimentos tika izmantots mērījumu biežums – 30 mērījumi sekundē. Eksperimenti tika veikti novietojot viedtālruni dažādās vietās automašīnas salonā, pievēršot lielu uzmanību tam, lai ierīce neslīdētu vai nebraukātu autovadīšanas laikā:

- Guļus stāvoklī, automašīnas salona centrālajā daļā esošajā nodalījumā.
- Telefona turētājā, kurš iestiprināts auto magnetolas CD disku atverē. Telefons atradās vertikālā stāvoklī
- Guļus stāvoklī uz automašīnas priekšējā paneļa, uz telefonam paredzēta lipīga

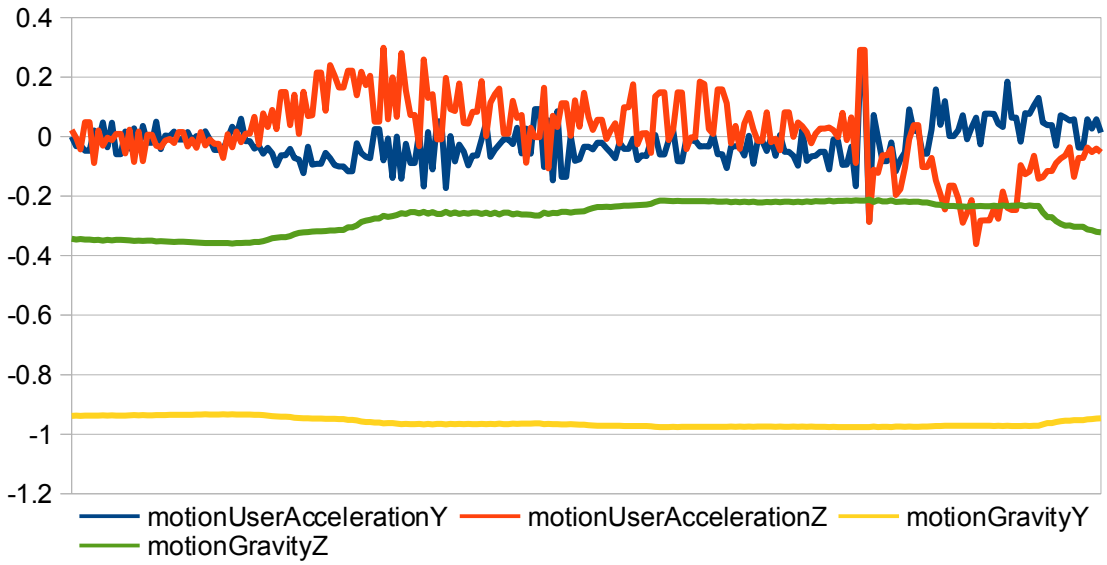
paklājiņa, kurš nodrošina to, ka ierīce braukšanas laikā neslīdēs.

Eksperimentā tika analizēti šādi manevri:

- akselerācija
- bremzēšana
- joslu maiņa
- pagriezienu veikšana

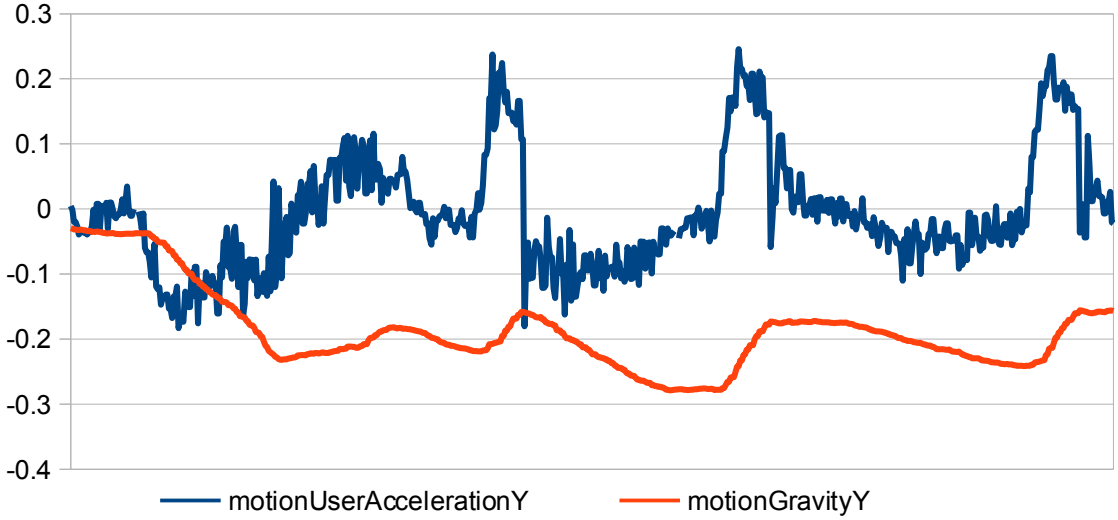
3.3.2 Akselerācija

Automašīnas akselerāciju var noteikt analizējot akselerometra datus asī, kas pavērsta automašīnas kustības virzienā. Attēlā 3.1. atspoguļots automašīnas paātrinājums ierīcei atrodoties vertikālā stāvoklī. Automašīnas kustības virzienam atbilst Z ass negatīvā virzienā, taču ierīce nav pilnībā perpendikulāri zemei, tādēļ jāņem vērā arī Y ass pozitīvais virziens. Automašīnas paātrinājums radīs izmaiņu sensoru rādījumos pretējā virzienā nekā automašīnas kustības virziens Attēlā 3.1. attēloti CMDeviceMotion acceleration un gravity dati Y un Z asīs, kuros var novērot izmaiņas, kas liecina par automašīnās akselerāciju.



3.1. att.: **Akselerācija ierīcei atrodoties vertikālā stāvoklī**

Attēlā 3.2. attēloti CMDeviceMotion acceleration un gravity dati Y asī. Ierīce atradās guļus stāvoklī paralēli ar zemi, tādēļ citu ašu mērījumos netika novērotas vērā ņemamas izmaiņas. Šis mērījums tika veikts ar augstāku mērījumu frekvenci (60hz). Mērījuma laikā ceļa segums bija klāts ar sniegu, tādēļ grafikā iespējams novērot intervālus, kad automašīnas riteņi izslīdēja un paātrinājums samazinājās.

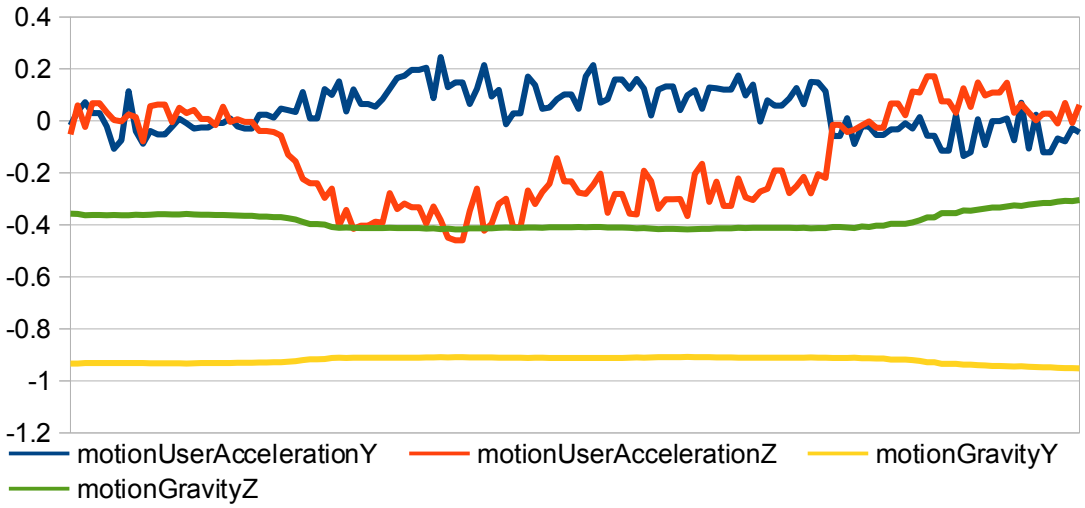


3.2. att.: **Akselerācija ierīcei atrodoties horizontālā stāvoklī**

Pēc šo rezultātu analīzes var izdarīt secinājumu, ka galā izstrādājamajai sistēmai, būs jāspēj noteikt viedtālruņa stāvoklis relatīvi pret kustības virzienu, un atkarībā no tā jāņem vērā vairāku akselerometra ašu mērījum. Akselerācijas manevrā citu sensoru mērījumos netika noteiktas izmaiņas. Pēc abām diagrammām var secināt, ka userAcceleration iekļauj lielu sensora troksni, bet gravity vērtībā, nav novērojamas tik, lielas datu amplitūdas izmaiņas. Attēlā 3.2 mērītā akselerācija bija straujākā nekā 3.1 attēlā, un to var secināt pēc maksimālas mērījumu datu amplitūdas.

3.3.3 Bremzēšana

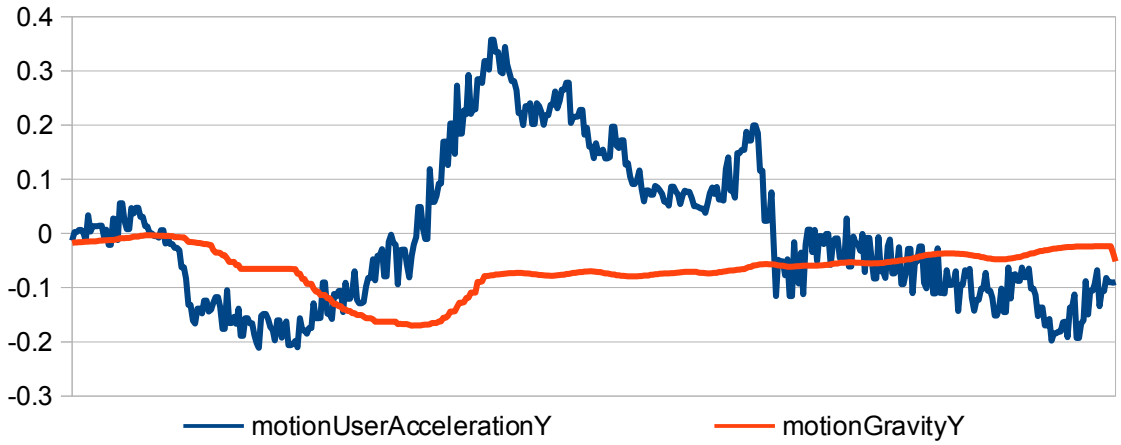
Automašīnas bremzēšanu, līdzīgi kā akselerāciju, var noteikt analizējot akselerometra datus asī, kas pavērsta automašīnas kustības virzienā, taču šajā gadījumā jāmeklē vērtību



3.3. att.: **Bremzēšana ierīcei atrodoties vertikālā stāvoklī**

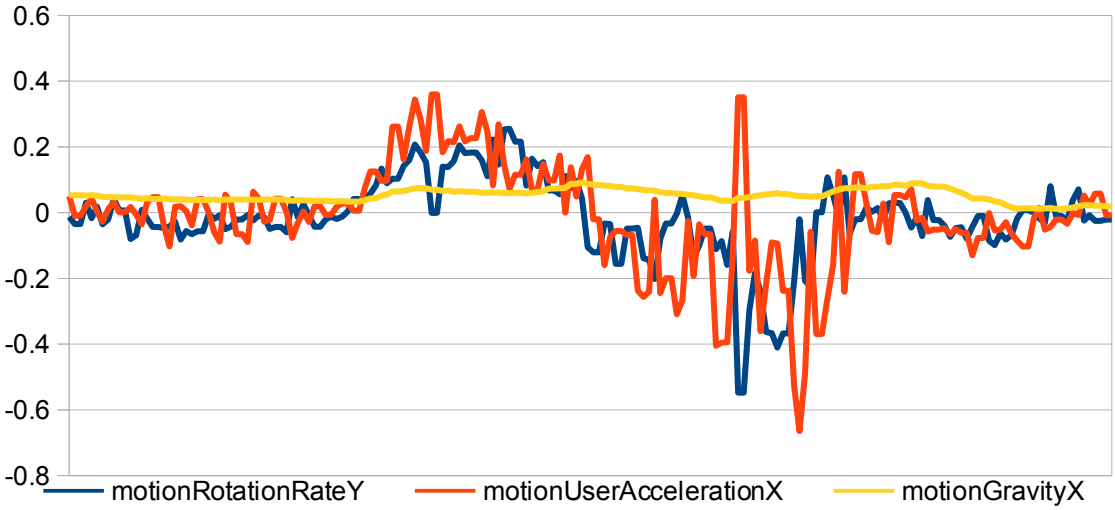
izmaiņa pretējā virzienā.

Attēlā 3.3. atspoguļots automašīnas bremzēšana ierīcei atrodoties vertikālā stāvoklī, taču ierīce nav perpendikulāri pret zemi, tādēļ jāņem vērā arī Y ass vērtības. Automašīnas bremzēšana radīs izmaiņu sensoru rādījumam tādā pašā virzienā kāda ir automašīnas kustība, ko arī var novērot 3.3. attēlā. 3.4. Attēlā modelēta akselerācijas, kas pāriet straujā bremzēšanā. Šajā piemērā ļoti interesanti ir tas, ka gravity vērtība bremzēšanas laikā nenonāk pozitīvā intervālā, kaut arī acceleration vērtība ilgāku laiku ir pozitīva (liecinot par strauju bremzēšanu). Iespējams, ka cēlonis tam ir straujais paātrinājums pirms bremzēšanas. 3.4. Attēlā bremzēšana bija krietni straujāka, ko var novērot pēc acceleration ass straujā kāpuma, par 0.5g. Ārī bremzēšanas datu analīze apstiprina to, ka izstrādājamajai sistēmai jāspēj noteikt automašīnas braukšanas virziens relatīvi pret telefona novietojumu, un jāspēj apvienot vairāku ašu mērījumu rezultāti.



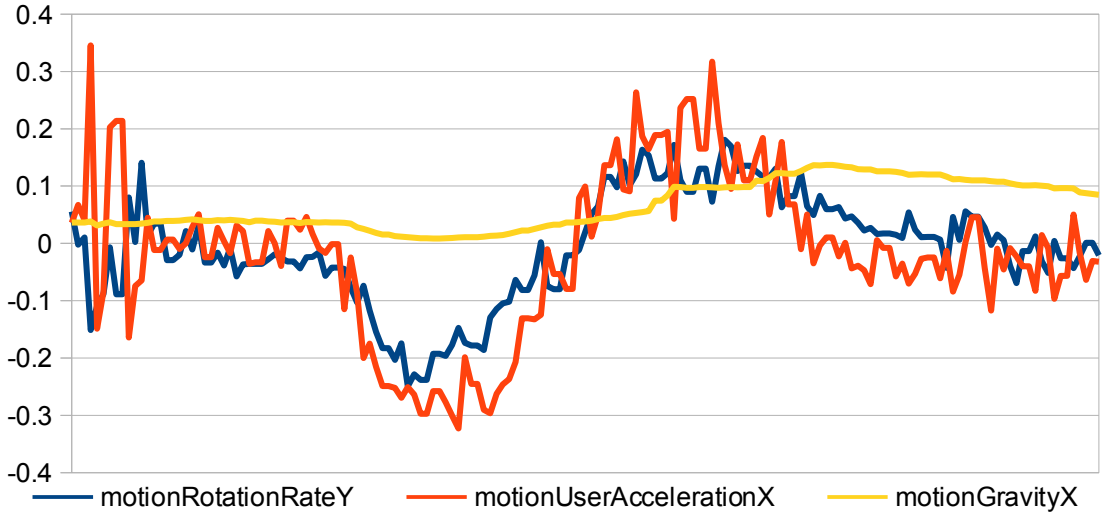
3.4. att.: **Bremzēšana ierīcei atrodoties horizontālā stāvoklī**

3.3.4 Joslu maiņa



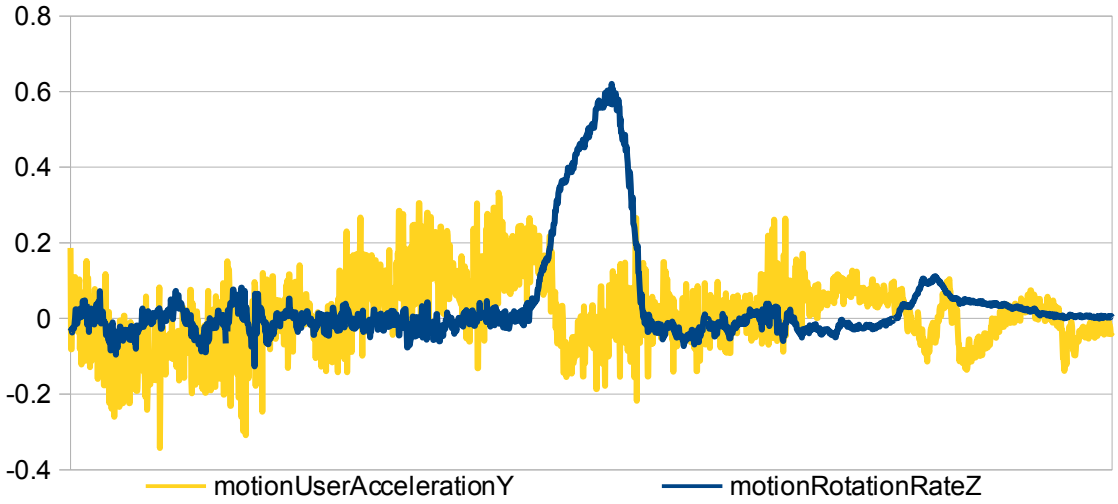
3.5. att.: **Pārkārtošanās uz kreiso joslu ierīcei atrodoties vertikāli**

Automašīnas braukšanas joslu maiņu var noteikt analizējot datus asī, kas ir perpendikulāri asij, kas sakrīt ar mašīnas braukšanas virzienu. Veiktajos eksperimentos viedtālrunis vienmēr bija pavērsts braukšanas virzienā, tādēļ eksperimentā mainot braukšanas joslu, akselerometra mērījumu izmaiņas tika noteiktas tikai X asī. Joslu maiņa ļoti labi atspoguļojās arī žiroskopa mērījumos. Žiroskopa mērījumos izmaiņas būs nosakāmas tajā asī, kas ir perpendikulāri, gan zemei, gan automašīnas kustības virzienam. Attēlos 3.5. un 3.6 šī ir Y ass, jo ierīce atrodas vertikālā stāvoklī un ir pavērsta automašīnas pārvietošanās virzienā. Analizējot iegūtos datus, var novērot ka acceleration un rotationRate vērtības ir laba indikācija straujam pārkārtošanās manevram, taču gravity vērtībās ir ļoti maza izmaiņa. No tā var secināt, ka joslu maiņas manevru ir iespējams noteikt no neapstrādātiem akselerometra un žiroskopa datiem, un gravity vērtība nav piemērota šī manevra identificēšanai.



3.6. att.: **Pārkārtošanās uz labo joslu ierīcei atrodoties vertikāli**

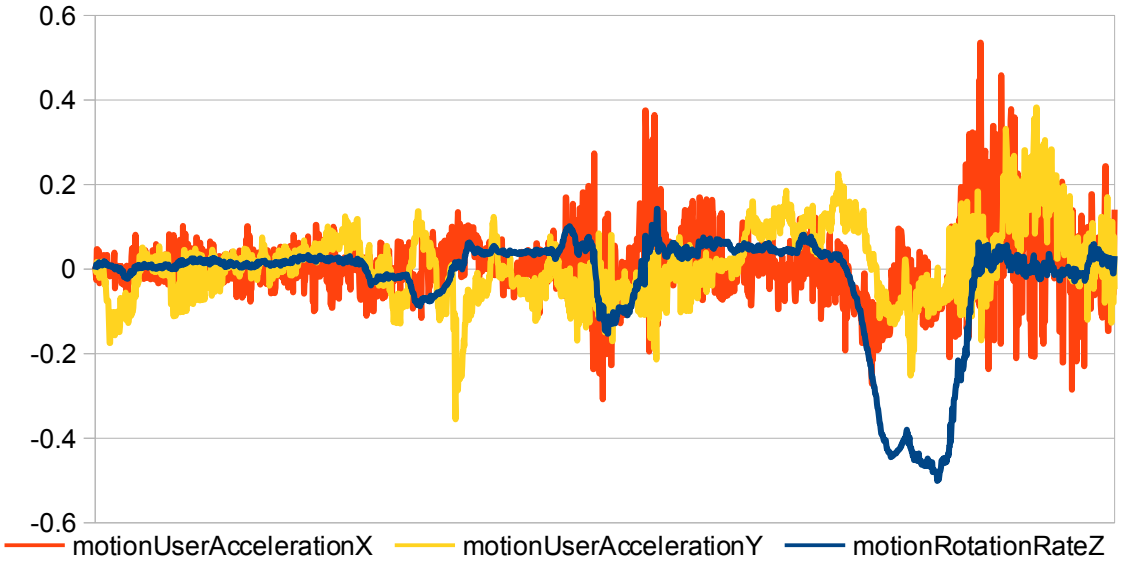
3.3.5 Pagriezienu veikšana



3.7. att.: **Kreisais pagrieziens ierīcei atrodoties horizontāli**

Automašīnas nogriešanās ir manevrs, kurš akselerometra datos uzrādās ļoti minimālu, taču žiroskopa mērījumos ir ļoti uzskatāms. Šis manevrs ir ļoti līdzīgs braukšanas joslu maiņas manevram, taču pastāv nozīmīga atšķirība - pagrieziena manevram ir lielāka žiroskopa mērījumu izmaiņa relatīvi pret sākuma stāvokli, jo automašīnas braukšanas virziena izmaiņa ir lielāka nekā mainot joslas. Neskatoties uz žiroskopa datiem, var analizēt arī citu sensoru datus, lai iegūtu vairāk informācijas par manevru. Attēlā 3.7. atspoguļots kreisais pagrieziens. Izpētot diagrammu, un apvienojot Y ass acceleration datus ar rotationRate datiem, var ievērot, ka līkuma pašā sākumā vadītājs ir bremzējis, un braucot ārā no līkuma palielinājis braukšanas ātrumu. Līdzīgu secinājumu var izdarīt aplūkojot 3.8. attēlu, kurā ir veikts labais pagrieziens.

3.8. attēlā redzamais manevrs, tika veikts samērā strauji un ar lielāku ātrumu nekā iepriekš apskatītais labais pagrieziens. Labajā pagriezienā var novērot arī akselerometra mērījumu svārstības X asī, kas liecina par to ka žiroskopa dati varētu tikt izmantoti kā primārais noteicošais faktors, tam ka ir veikts pagrieziens, un akselerometra dati varētu tikt izmantoti, lai noteiktu manevra straujumu un bīstamību.



3.8. att.: **Labais pagrieziens ierīcei atrodoties horizontāli**

3.3.6 Eksperimenta secinājumi

Pēc eksperimentā iegūto datu analīzes tiek veikti sekojoši secinājumi, kas jāņem vērā manevru atpazīšanas programmatūras izstrādē:

- Visi meklētie manevri ir diferencējami pēc sensoru iegūtajiem datiem.
- Mērījumu diagrammās novērojamas nepārtrauktas mērījumu svārstības ar mazu amplitūdu, kas rodas sensora trokšņa un ārējo apstākļu rezultātā (automašīnas motora vibrācija, ceļa seguma nelīdzenumi, u.c.). Iespējams, ka būs jāveic sensora datu apstrāde, lai novērstu šo parādību.
- Sensoru datos, kas iegūti ar lielāku mērījumu skaitu sekundē, novērojams lielāks sensora troksnis.
- Viedtālruņa novietojums attiecībā pret automašīnās kustības virzienu ietekmē to, kuru sensora ašu mērījumā nosakāms konkrēts manevrs. Tas nozīmē, ka lai izstrādātu programmatūru, kura spējīga noteikt manevrus neatkarīgi no ierīces novietojuma, ir nepieciešama vairāku mērījumu ašu apvienošana viena manevra noteikšanā.

- Visi manevri sensora datos atspoguļojas, kā datu izmaiņas dažādos laika intervālos, un lai izvairītos no mērījumu anomāliju uzskatīšanas par manevru, manevra noteikšana jāveic balstoties gan pēc sensora datu novirzes, gan pēc laika intervāla.

4. PIEDĀVĀTAIS RISINĀJUMS

Apkopojot un analizējot gan veikto teorētisko izpēti, gan arī praktiskā eksperimenta rezultātus, var sākt realizēt šī darba galveno mērķi un izstrādāt konceptuālu risinājumu jeb metodoloģiju, kurā noteiktas vadlīnijas, atrisinātas specifiskas problēmas un analizēti aspekti, kas jāņem vērā, lai varētu izveidot automatizētu risinājumu priekš automašīnas veikto manevru atpazīšanas. Risinājuma tvērumā neietilpst atpazītos manevrus izmantot konkrētā darbībā vai darījumprocesā, bet piedāvāt risinājumu, kas var tikt paplašināts, papildināts un pielāgots dažādiem lietojumu gadījumiem. Programmatūra, kas būtu realizēta atbilstoši risinājumam, tiek plānota kā komponente, kas var tikt integrēta citā sistēmā un izmantota, lai paplašinātu šo sistēmu ar manevru atpazīšanas funkcionalitāti, kas tālāk sistēmā tiek izmantota atbilstoši konkrētajām prasībām un lietojuma gadījumam. Izstrādājot risinājumu tiks noteiktas tā kritiskākās sastāvdaļas jeb komponentes, kā arī aprakstītas darbības un aspekti, kas jāņem vērā realizējot automatizētu manevru atpazīšanu, kas sasniedz darbā izvirzītos mērķus.

4.1 Risinājuma prasības un ierobežojumi

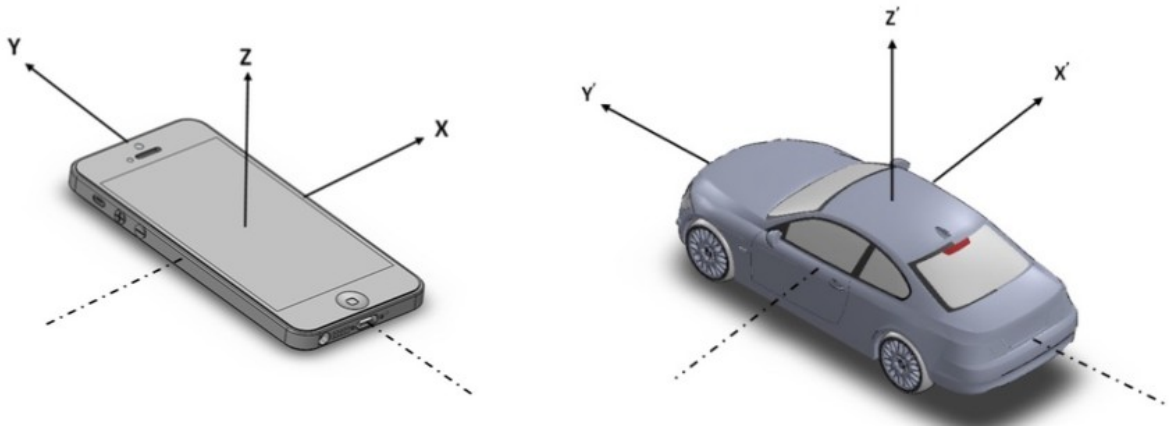
Lai piedāvātais konceptuālais risinājums atbilstu darbā izvirzītajiem mērķiem, tiek definētas vispārīgas prasības, kuras nepieciešams apmierināt. Prasības tiek definētas balstoties uz plānoto komponentes pielietojumu, kā arī ņemot vērā risināmās problēmas specifiku.

1. Manevru atpazīšanai jānotiek reālā laikā, un atpazīšana ir jāveic pēc tekošajiem sensoru mērījumu rezultātiem.
2. Risinājumam jāspēj atpazīt dažādi manevri pēc dažādiem sensoru ieejas datiem. Tādējādi risinājums nav piesaistīts ierobežojumiem kādi manevri var tikt atpazīti, un dod iespēju nepieciešamības gadījumā paplašināt meklējamo manevru kopu.
3. Meklētajiem manevriem jābūt atdalītiem no komponentes. Manevriem jābūt ielasāmiem no ārēja datu avota, piemēram, faila. Tādējādi risinājums nav piesaistīts noteiktai manevru kopai, un to ir iespējams pielāgot jauniem manevriem neveicot izmaiņas.
4. Risinājumam jānodrošina iespēja ierakstīt sensoru izejas datus autovadīšanas laikā, un eksportēt šos datus ārējā datu glabātājā, piemēram, failā. Izejas faili pēc tam var tikt izmantoti kā meklēto manevru bāzlīnijas vai šabloni.
5. Risinājuma realizācijā jāizmanto komponent-bāzēta programmatūras izstrādes pieeja,

lai programmatūru varētu viegli integrēt citās sistēmās, kā arī vajadzības gadījumā modificēt kādu no risinājuma komponentēm, bez nepieciešamības modificēt visu sistēmu.

Lai palielinātu iespēju, ka veiksmīgi tiek sasniegti darbā izvirzītie mērķi, un, ka atbilstoši piedāvātajam risinājumam var tikt izstrādāta strādājoša programmatūra, nepieciešams pieņemt dažus ierobežojumus. Ja risinājumam ir pieņemti racionāli ierobežojumi, tad var koncentrēties uz galvenā mērķa sasniegšanu, kas ir manevru atpazīšana. Pēc galvenā mērķa sasniegšanas, ieguldot papildu darbu, iespējams no šiem ierobežojumiem var definēt jaunas prasības, kas uzlabotu un paplašinātu risinājuma darbību un funkcionalitāti, taču sākuma izstrādes stadijā, tas var pārlieku paplašināt sfēru un traucēt galvenā mērķa sasniegšanai. Risinājumam tiek noteikti sekojoši ierobežojumi:

1. Viedtālrunis automašīnā ir novietots nekustīgā stāvoklī. Lai risinājums spētu veikt autovadīšanas analīzi, ierīcei ir jābūt novietotai tā, lai tā nepārvietotos relatīvi pret automašīnu – ierīce autovadīšanas laikā nedrīkst slīdēt, krist vai kustēties. Jebkāda veida statīvs, turētājs vai neliels sīklietu glabāšanas nodalījums var būt atbilstošs veids kā nodrošināt šī ierobežojuma ievērošanu.
2. Viedtālrunis, kas tiek izmantots manevru atpazīšanai, atrodas noteiktā un nemainīgā stāvoklī relatīvi pret automašīnu, un šāds pats stāvoklis tika izmantots manevru ierakstīšanā. Ar to tiek specificēts, ka, ja manevri tiek ierakstīti stāvoklī, kad viedtālruņa X, Y un Z asis sakrīt ar automašīnas X, Y un Z asīm, tad šāds viedtālruņa stāvoklis tiek saglabāts arī tad, kad tiek veikta manevru atpazīšana (skatīt attēlu 4.1).



4.1. att.: **Viedtālruņa un automašīnas relatīvais novietojums**

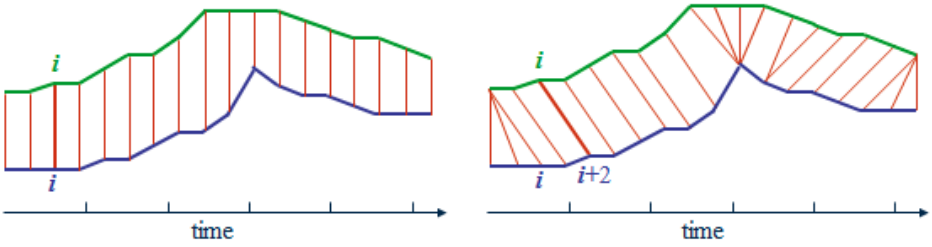
3. Konkrētā laika intervālā var tikt noteikts tikai viens manevrs. Šis ierobežojums ļaus izvairīties no situācijas, kad viens manevrs var tikt vienlaicīgi uzskatīts par vairākiem

savstarpēji līdzīgiem manevriem, piemēram, kreisais pagrieziens varētu vienlaicīgi tikt atpazīts gan kā pagrieziens, gan kā apgriešanās manevrs.

4.2 Manevru atpazīšana

Pati kritiskākā un svarīgākā risinājuma sastāvdaļa ir manevru atpazīšana. Analizējot sensoru datus un apkopojot veiktā eksperimenta rezultātus, var secināt, ka sensoru mērījumos ir novērojamas neparedzētas vērtību svārstības, kas ir sastopamas dažādās amplitūdās un var rasties daudz un dažādu apstākļu rezultātā, piemēram, automašīnas dzinēja vibrācijas, ceļa seguma nelīdzenumi, automašīnas kustības ātrums, u.c. Nozīmīgs aspekts ir arī tas, ka dažādu manevru noteikšanai jāizmanto dažādu veidu sensori, kuriem ir atšķirīgas īpašības un novērojamais troksnis. Ne mazāk nozīmīgi ir arī tas, ka divi vienāda tipa manevri sensoru mērījumos var attēloties gan ar dažādām vērtību izmaiņām, gan arī ar dažādām vērtībām, kas ir atkarīgas no veiktā manevra apstākļiem – pārvietošanās ātrums, manevra izpildes ilgums, mašīnas kustības trajektorijas izmaiņa relatīvi pret sākuma stāvokli, u.c. Ņemot vērā visus šos apstākļus, var secināt ka ir nepieciešams netriviāls veids kā atpazīt manevrus. Manevru atpazīšanā izmantojot tādus triviālus paņēmienus, kā, piemēram, konkrēta skaita minimālo vērtību meklēšana noteiktā intervālā, vai vidējās vērtības noteikšana intervālā, nedos pietiekami labus rezultātus, lai būtu izmantojami šajā risinājumā.

Manevru atpazīšanā nepieciešams izmantot iepriekš sagatavotu manevra bāzlīniju, jeb šablonu. Manevra bāzlīnija ir vienas sensora ass mērījumu rezultātu virkne, kas ir sakārtota augošā secībā pēc laika. Lai atpazītu konkrētu manevru nepieciešams salīdzināt bāzlīniju un ieejas mērījumu virkni, un noteikt to līdzību. Apskatot mērījumu virkni, $N = n_1,n_2,n_3, \ldots n_i$ un bāzlīniju $M = m_1,m_2,m_3, \ldots m_i$. Ja tiks meklēts, piemēram, Eiklīda attālums, starp n_i un m_i iegūtais rezultāts slikti atspoguļos virkņu līdzību, jo tiek salīdzināti tikai punkti, kas atbilst vienai n pozīcijai. Šajā gadījumā netiek apstrādāts gadījums, kad līdzīgās virkņu daļas var atrasties dažādās n pozīcijās. Lai apstrādātu šādus gadījumus nepieciešams algoritms, kas līdzīgus fragmentus mērījumu virknēs var noteikt arī tad, kad tie atrodas dažādos laika momentos, tādēļ tiek apskatīts DTW algoritms [24.].



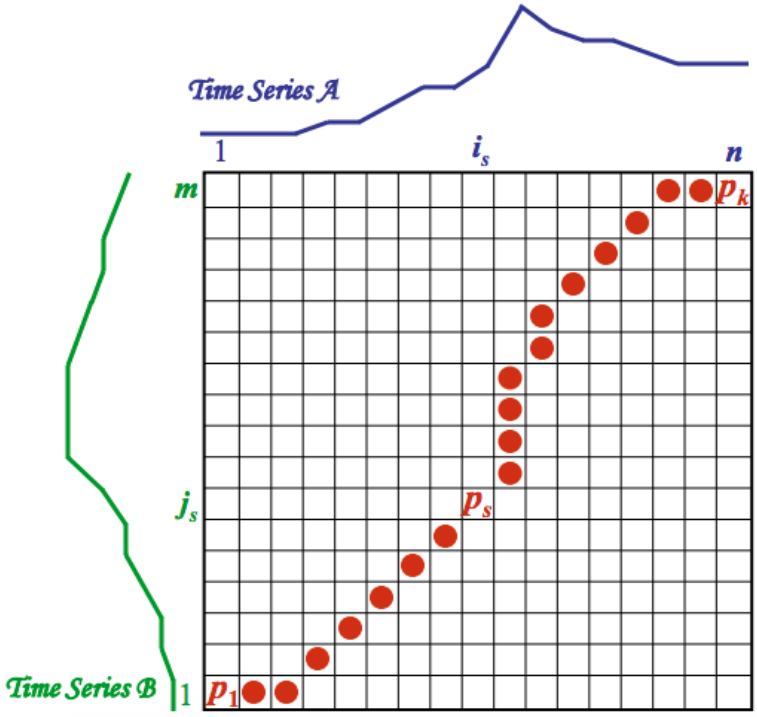
*4.2. att.:***Eiklīda un DTW attāluma ilustrācija [24.]**

4.2.1 DTW algoritms

DTW algoritms savstarpēji salīdzina no laika atkarīgu un secīgu vērtību virknes, kas varētu arī tikt sauktas par signāliem. Šis algoritms sākotnēji tika izstrādāts priekš runas atpazīšanas. Mūsdienās tas tiek izmantots daudzās nozarēs, kā rokraksta un parakstu salīdzināšanā, zīmju valodas atpazīšanā, žestu noteikšanā, datizracē, mūzikas atpazīšanā, u.c. [25.].

DTW algoritms savu popularitāti ir guvis pateicoties tā iespējām izteikt divu signālu līdzību, samazinot signāla trokšņa un līdzīgo fragmentu nobīžu ietekmi uz rezultējošo līdzības mēru. Ar DTW algoritmu var salīdzināt divus signālus, kuriem var būt atšķirīgs garums garums[25.].

Lai ilustrētu kā salīdzināt divas vērtību virknes izpētīsim piemēru virknes A = i_1,i_2,i_3, ... i_n un B = j_1,j_2,j_3, ... j_m. Var konstruējot matricu ar izmēru n*m un katrā matricas laukā ierakstīt attālumu starp i_n un j_m. Lai atrastu abu virkņu labāko sakritību, neņemot vērā nobīdes, nepieciešams atrast ceļu matricā ar vismazāko kopējo vērtību no (1,1) lauka līdz (n,m) laukam (skatīt attēlu 4.3) [26.].



4.3. att.: **Divu virkņu labākās sakritības ceļš [24.]**

DTW algoritms šādu divu virkņu labāko sakritības ceļu $P = p_1,p_2,..p_k$ (skatīt attēlu 4.3) atrod optimālākā veidā, izmantojat dinamiskās programmēšanas pieeju un pieņemot dažus ierobežojumus [24.]:

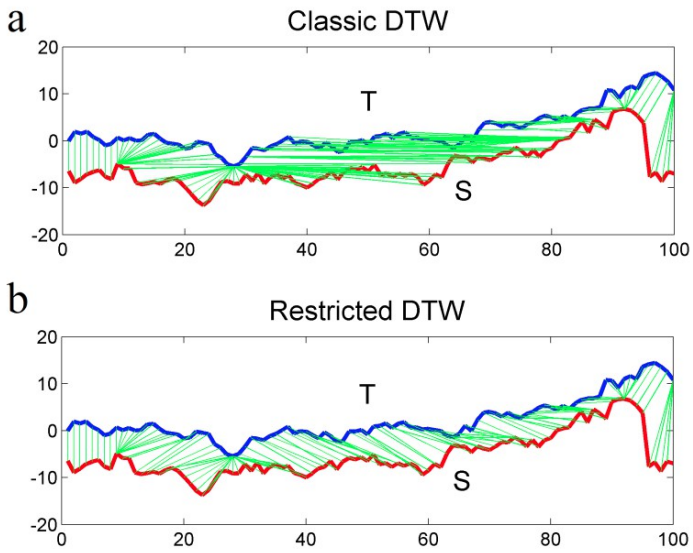
- Robežu nosacījums. $p_1 = (1,1)$ un $p_k = (n, m)$.
- Nepārtrauktība. $i_s - i_{s-1} \leq 1$ un $j_s - j_{s-1} \leq 1$.
- Monotonitāte. $i_{s-1} \leq i_s$ un $j_{s-1} \leq j_s$.

DTW algoritmā šī matrica tiek aizpildīta izmantojot dinamiskās programmēšanas principus, ka kopējā problēma tiek sadalīta mazākās apakš-problēmās. Atrisinot un apvienojot apakš-problēmās tiek iegūts arī kopējais risinājums. Loģika kā algoritms aizpilda matricu ir:

lauks(n,m) = attālums_līdz(n,m) + MIN(lauks(n-1,m), lauks(n-1,m-1), lauks(n, m-1)).

Pēc šādas matricas aizpildīšanas loģikas, pēdējais matricas lauks (n,m) saturēs vērtību, ar kuru izteikts attālums starp abām virknēm [27.].

Šāda metode var izveidot blakus parādību, ka vairāki intervāli vienā virknē var tikt pieskaņoti vienam intervālam otrā virknē. Tipisks veids, kā izvairīties no šīs problēmas ir ierobežot sakritības ceļu un neļaut tam novirzīties no matricas diagonāles – skatīt attēlu 4.4 [27.]. Ierobežojot vērtību algoritma darbību tuvāk diagonālei, tiek arī samazināta algoritma laika sarežģītība.



4.4. att.: **Klasiskais un ierobežotais DTW algoritms [27.]**

Sensoru izejas datu plūsma atbilst visiem ierobežojumiem un prasībām DTW algoritmam, kā arī algoritma tolerance pret trokšņiem un nobīdēm salīdzināmajos signālos, padara šo algoritmu par piemērotu viedu, kā realizēt manevru atpazīšanu piedāvātajā risinājumā. Uzticamāku rezultātu iegūšanai būtu jāizmanto ierobežotais DTW algoritms.

4.2.2 Manevru atpazīšanas izpilde

Risinājumam izvēlētā manevru atpazīšanas metode spēj salīdzināt divas sensora datu vērtību virknes, taču ņemot vērā secinājumus, kas tika izdarīti 3.3 nodaļā aprakstītā praktiskā eksperimenta, manevrs var attēloties vairākās mērījumu asīs, kā arī no viena vai vairākiem sensoriem. Lai efektīvi noteiktu vai ieejas datos ir atrodama līdzība starp meklēto manevru, ir nepieciešams salīdzināt manevram būtiskās mērījumu asis ar tekošajiem ieejas mērījumu datiem, un pārējās mērījumu asis ignorēt. Lai to varētu realizēt, meklētajam manevram, jāpievieno papildus metadati ar informāciju par to, kuras no mērījumu asīm nepieciešams salīdzināt, lai atpazītu konkrēto manevru.

Ar šādu pieeju, kad tiek salīdzinātas tikai būtiskās mērījumu asis, tiek samazināts aprēķinu skaits, kas būtu jāveic, lai novērtētu manevru līdzību, salīdzinot ar nepieciešamo operāciju skaitu, kāds būtu jāveic ja, nosakot manevru, tiktu salīdzinātas visas mērījumu asis. Manevru atpazīšanas precizitāte arī varētu tikt negatīvi ietekmēta, ja tiktu salīdzinātas asis kurās tieši neattēlojās meklētais manevrs, jo salīdzināšana tiktu veikta būtībā ar sensora troksni, kas var ieviest tikai jaunas neprecizitātes.

4.2.3 Manevra atpazīšanas rezultāta novērtēšana

Manevru atpazīšana tiek veikta atbilstoši iepriekš aprakstītajai metodei, kad tiek salīdzinātas individuālas sensoru mērījumu asu vērtības. Šīs salīdzināšanas rezultātā tiek iegūts skaitlis, ar kuru izteikta meklētā un ieejas vērtību virkņu līdzība. Ņemot vērā ka manevru var aprakstīt vairākās mērījumu asis, rodas nepieciešamība izrēķināt katras manevra raksturojošās ass līdzības koeficientu pret ieejas datiem, un apvienot iegūtos rezultātus vienā vērtībā.

Vienkāršākais veids kā iegūt vispārīgu manevra līdzības vērtību ir izrēķināt vidējo vērtību no visām manevra asu līdzības vērtībām, tādējādi tiek iegūta viena skaitliska vērtība, kurā izteikta ieejas datu līdzība pret konkrētu manevru. Šī vērtība var tikt izmantota, lai pieņemtu lēmumu vai konkrētie ieejas dati atbilst dotajam manevram.

Analizējot 3.3 nodaļā veikto eksperimentu, var secināt, ka iespējamo manevru dažādība ir ļoti liela, kā arī ir novērojama liela atšķirība sensoru mērījumos, kad tiek izmantotas neapstrādātas vai apstrādātas vērtības. Vienā gadījumā var novērot lielu sensoru troksni, bet apstrādāto vērtību gadījumā, sensoru troksnis ir ļoti minimāls. Šie aspekti liecina par to, ka ļoti lietderīgi būtu manevra atpazīšanas sliekšņa maksimālo vērtību piesaistīt pie konkrēta manevra, nevis izmantot vienu konstantu vērtību visiem manevriem. Ar šādu pieeju, atkarībā no manevra specifikas būtu iespējams individuāli konfigurēt to, cik liela sakritība ir nepieciešama starp ieejas datiem un konkrētu manevru, lai uzskatītu šo manevru par notikušu. Lai realizētu šo pieeju, meklētajam manevram jāpievieno papildus metadati par to, kāds ir maksimālais līdzības koeficients, līdz kuram ieejas datus var uzskatīt par konkrēto manevru.

4.2.4 Vairāku manevru atpazīšanas rezultātu apstrādāšana

Viens no risinājuma pieņemtajiem ierobežojumiem nosaka, ka vienā laika intervālā var notikt tikai viens manevrs. Taču, netiek izslēgta iespēja, ka vienā, laika intervālā, var tikt atpazīti vairāki manevri. Rodas nepieciešamība definēt mehānismu, kā rīkoties, kad tiek atpazīti vairāki manevri.

Ja manevru atpazīšana tiek veikta vairākiem manevriem, nepieciešams iegūtos atpazīšanas rezultātus savstarpēji salīdzināt. Risinājumam nav noteikts ierobežojums par meklēto manevru garumu, tas nozīmē, ka manevra atpazīšana var noritēt dažādā laikā pēc manevra sākuma brīža. Manevru atpazīšanas rezultātu savstarpējo salīdzināšanu var veikt tikai tad, kad visas šobrīd notiekošās manevru atpazīšanas ir pabeigtas. Vispirms jāveic pārbaude vai katra individuālā manevra atpazīšanas koeficients, ir tāds, lai tiktu apmierināta

šim manevram atbilstošā atpazīšanas sliekšņa maksimālā vērtība. Ja kāda manevra atpazīšanas rezultāts ir lielāks par šī manevra sliekšņa maksimālo vērtību, tad šis manevrs var tikt ignorēts. Tiek iegūti tikai manevri, kas ir derīgi, un lai tos sasvstarpēji objektīvi salīdzinātu, var izrēķināt manevra ticamības koeficientu, dalot iegūtu atpazīšanas rezultātu ar atpazīšanas sliekšņa maksimālo vērtību. Salīdzinot iegūtos ticamības koeficientus, var izvēlēties vienu manevru, kam šis koeficients ir zemākais, un pieņemt, ka šis ir tikko notikušais manevrs.

4.3 Manevru saglabāšana

Pirms risinājums var tikt izmantots manevru atpazīšanai, nepieciešams izveidot šablonus, jeb bāzlīnijas manevriem. Dēļ šī iemesla risinājumam ir jāpiedāvā iespēja, saglabāt sensoru tekošos mērījumu rezultātus un eksportēt tos ārējā datu glabātājā. Vienkāršākajā gadījumā tas var būt fails, kas satur JSON formātā strukturētus mērījumu rezultātus.

Manevru saglabāšanas operācijā ir nepieciešama lietotāja iesaistīšana. Manevru saglabāšanas režīmā ir nepieciešama lietotāja ievade par sekojošiem notikumiem:

- manevra sākums (sākt sensoru datu saglabāšanu)
- manevra beigas (pārtraukt sensora datu saglabāšanu)

Izmantojot risinājumu saglabāšanas režīmā netiek veiktas automātiskas darbības balstoties uz sensoru datiem. Lietotājs pieņem lēmumu un ievada sistēmā, kad sākt un kad beigt manevra ierakstīšanu. Rezultējošie dati, kas atspoguļo manevru pēc eksportēšanas ir pieejami lietotājam. Eksportēti tiek visi atbalstītie sensori un visas sensora mērījumu asis, tādā struktūrā un formātā, kāda ir nepieciešams importēt manevrus, lai tie tiktu izmantoti atpazīšanas režīmā. Dati tiek strukturēti pēc laika momenta, kad tika veikti mērījumi. Visiem mērījumiem ir jābūt sakārtotiem augošā secībā pēc laika momenta, kā arī laika momenta vērtību ir nepieciešams pievienot eksportētajiem datiem. Tādējādi tiks nodrošināts parametrs, pēc kura mērījumi var tikt sakārtoti, ja tie tiek glabāti veidā, kad varētu mainīties to secība.

JSON datu formāts ir izvēlēts kā ieteicamais, jo tas ir cilvēkiem viegli lasāms, kā arī JSON pievieno salīdzinoši maz metadatu informācijai, kas ir jāglabā glabāta, tādējādi rezultējošā faila izmērs netiek krasi palielināts tikai dēļ izmantota datu glabāšanas formāta [28.]. Šis datu formāts ir ļoti populārs un tā izmantošanai ir pieejami gan rīki, kas pēc noklusējuma ir iekļauti iOS SDK[29.], gan trešo personu izstrādātie ietvari, kas atvieglo programmatūrā izmantojamu objektu serializāciju no JSON datiem [30.].

4.4 Manevra sākuma noteikšana

Risinājumā izvēlētā manevru atpazīšanas metode, salīdzina divas vērtību virknes, kas sakārtotas augošā secībā pēc laika un kurām ir ierobežots garums. No ierīces sensoriem mērījumi tiek saņemti nepārtraukti un ar noteiktu intervālu. Šis intervāls ir konfigurējams, un tā maksimāla un minimālā vērtība nav precīzi zināma, un var būt atkarīga no izmantotās ierīces [31.]. Taču tipiski šis mērījumu intervāls var būt robežās no 10 līdz 100 mērījumiem sekundē [32.]. Pie tik biežas datu saņemšanas, kā arī ņemot vērā to ka manevra salīdzināšanas process ir netriviāls, atkrīt iespēja manevru atpazīšanu veikt pie katriem saņemtajiem datiem. Rodas nepieciešamība izveidot mehānismu, ka noteikt vai tekošajos ievaddatos ir potenciāla līdzība ar kādu no meklētajiem manevriem.

Tā kā pārbaude, vai ar tekošajiem ievaddatiem ir nepieciešams veikt manevra atpazīšanu, būs jāveic katru reizi, kad tiks saņemti tekošie sensoru mērījumi, lietderīgi ir izvēlēties pēc iespējas vienkāršāku pārbaudi, kā noteikt intervālu kurā potenciāli būtu sastopams manevrs. Līdzīgā pētījumā [2.] autori bija izvēlējušies tekošās vidējās vērtības rēķināšanu un salīdzināšanu ar iepriekš noteiktām vērtībām, tādējādi atrodot manevra sākumu (brīdī, kad vidējā vērtība pārsniedz noteikto slieksni) un manevra beigu punktu (tad, kad vērtība izmainās zem šī sliekšņa). Šāda pieeja būtu viens no potenciālajiem risinājumiem, taču sagaidāmie trūkumi būtu:

- Šādu mehānismu būtu sarežģīti implementēt
- Ierakstītajiem un eksportētajiem manevriem šī vērtība būtu grūti manuāli nosakāma un pārbaudāma.
- Joslu maiņas manevrus, kas atspoguļojas gan pozitīvās, gan negatīvās mērījumu vērtībās vienā asī, varētu būt problemātiski noteikt, jo vidējā vērtība visā manevrā veidojas tuvu nullei - skatīt attēlu 3.4.

Vidējās vērtības izmantošana sākuma un beigu punktu noteikšanai rada pārāk daudz riskus un nenoteiktības pret nelielu ieguvumu, lai to izmantotu ka manevra robežu noteikšanas mehānismu, tādēļ izvēlēts krietni vienkāršāks veids kā noteikt manevra sākumu un beigu punktu.

Katram meklētajam manevram jānosaka konkrēta minimālā vērtība kādā no mērījumu asīm, un skaits, cik mērījumus pēc kārtas vērtībai ir jābūt virs noteiktā minimālā sliekšņa, lai uzskatītu, ka šobrīd tiek veikts meklētais manevrs. Ar šādu pieeju tiktu fakts, ka manevrs šobrīd tiek veikts, tiktu konstatēts brīdī, kad manevrs jau ir noticis, taču, lai veiktu manevra atpazīšanu nepieciešams zināt tā sākums. Lai risinātu šo problēmu jāpievieno vēl viena

konfigurējamu vērtību – manevra sākuma nobīde. Brīdī kad tiek apmierināts minimālā sliekšņa nosacījums pietiekamā skaitā mērījumu, izmantojot nobīdes vērtību var objektīvi pieņemt, ka ja šobrīd notiek dotais manevrs, tad iegūtajā mērījumu virknes indeksā būtu jābūt dotā manevra sākumam. Beigu indeksu var atrast vienkārši pieskaitot atrastajam sākuma indeksam meklētā manevra mērījumu skaitu. Lai nodrošinātos, ka šobrīd notiekošais manevrs varētu noteikt ilgāk par bāzlīnijas manevru, bāzlīnijas manevru skaitu var reizināt ar nelielu konstanti, kad tiek meklēts beigu indeks. Lai realizētu šādu paņēmienu meklētajiem manevriem jāpievieno papildus meta dati.

4.5 Manevra metadati

Risinājuma izstrādē tika izvirzīta prasība, ka manevriem, kuri tiek atpazīti, ir jābūt atdalītiem no funkcionālās daļas un ielasāmiem no ārēja datu avota. Lai risinājums varētu atbilst pārējām izvirzītajām prasībām, meklētajiem manevriem ir jāpievieno papildus metadati, kas tiek izmantoti dažādās risinājuma komponentēs. Manevru saglabāšanas un eksportēšanas funkcionalitāte nepievieno šos metadatus. Papildu informācija par manevru kalpo, kā konfigurācija konkrētajam manevram, kas ir vitāli nepieciešama, lai risinājums varētu strādāt, kā arī, lai nodrošinātu to, ka risinājums ir pietiekami dinamisks, lai neuzliktu ierobežojumus manevriem, kas var tikt atpazīti. Rediģējot un pielāgojot katra manevra konfigurāciju lietotājs, var krasi ietekmēt manevra atpazīšanas biežumu un precizitāti.

Atpazīstamajiem manevriem nepieciešams pievienot šādu papildu informāciju:

- **nosaukums** – unikāls nosaukums, pēc kura identificēt manevru
- **būtiskās sensoru asis** – jānorāda kāda no iepriekš definētām vērtībām, ar kuru var izteikt visas iespējamās sensoru ašu kombināciju, kuras var tikt izmantotas risinājumā. Pēc šīs vērtības tiks secināts, kuras sensoru mērījumu asis ir jāsalīdzina, lai noteiktu konkrēto manevru
- **manevra sliekšņa minimālā vērtība** – skaitliska mērījumu minimālā vērtība, kas ir jāpārsniedz, lai uzskatītu, ka šobrīd ir sācies konkrētais manevrs
- **manevra sliekšņa garums** – skaitliska vērtība, ar kuru norādīts, cik mērījumiem pēc kārtas ir jāpārsniedz sliekšņa minimālā vērtība lai uzskatītu, ka manevrs ir sācies
- **manevra sliekšņa sensoru ass** – kāda no iepriekš definētām vērtībām, ar kuru izteikta viena no sensora asīm, kura ir jāizmanto, lai konstatētu vai manevrs ir sācies
- **sākuma indeksa novirze** – skaitliska vērtība, ar kuru norādīts, cik indeksus atpakaļ izvēlēties manevra sākuma punktu, kad tiek sasniegts manevra sliekšņa nosacījums

- **manevra atpazīšanas maksimālā vērtība** – skaitliska vērtība, kas ir maksimālā pieļaujamā manevra atpazīšanas rezultātā iegūtā vērtība, lai uzskatītu, ka dotajos sensora datos atspoguļojas konkrētais manevrs.

5. RISINĀJUMA PROTOTIPS

Atbilstoši iepriekš aprakstītajam konceptuālajam risinājumam tika praktiski izstrādāta programmatūra kā prototips, lai pārbaudītu piedāvāto risinājumu un praktiski pielietotu autovadīšanas manevru atpazīšanu. Plānojot programmatūras arhitektūru uzmanība tiek pievērsta tam, lai programmatūra varētu tikt pēc iespējas vienkāršāk pilnveidota un papildināta. Programmatūra izstrādāta priekš iOS operāciju sistēmas izmantojot programmēšanas valodu Objective-C. Pilns programmatūras pirmkods pievienots 7. pielikumā.

5.1 Prototipa arhitektūra

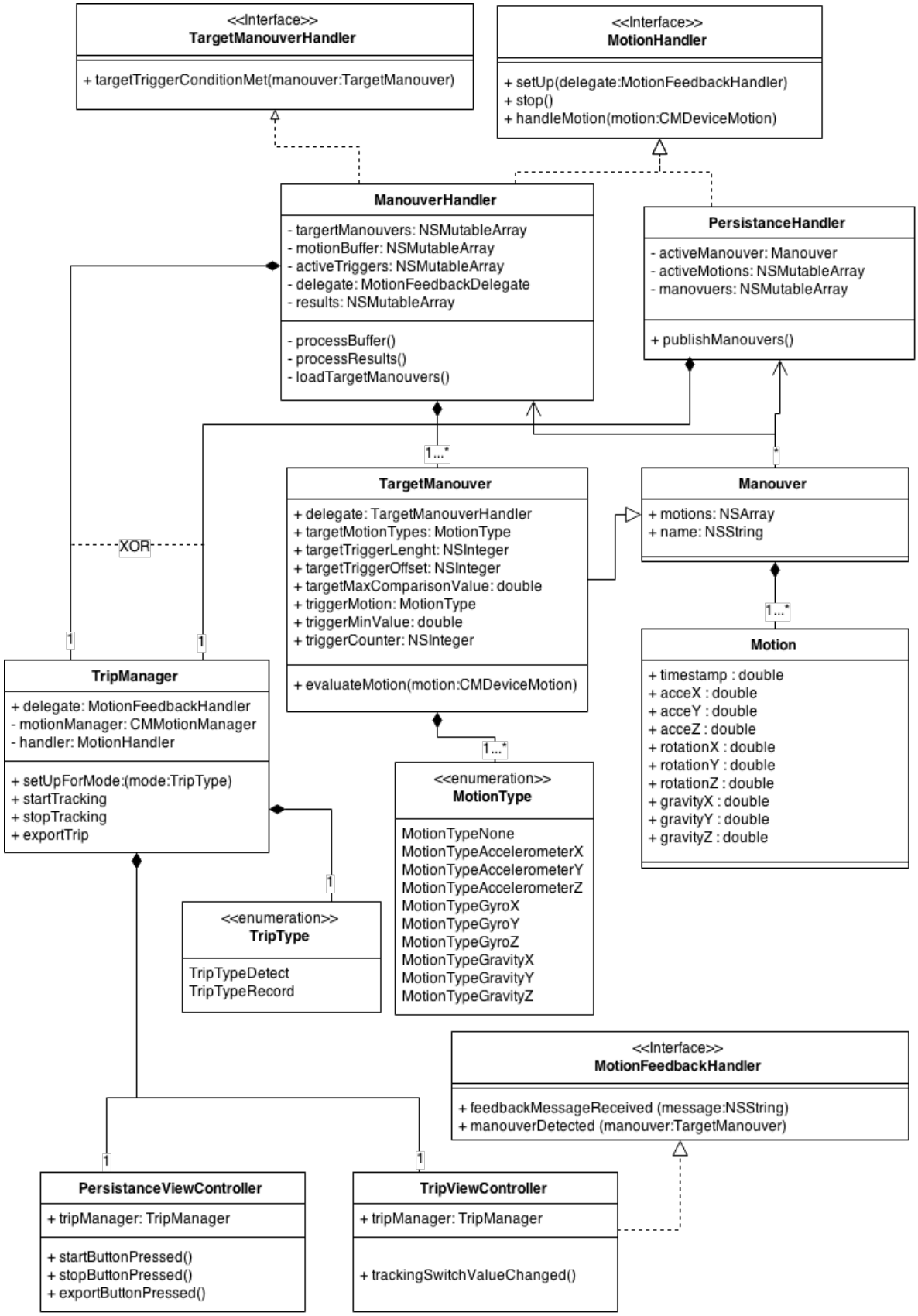
Atbilstoši iepriekš aprakstītajam risinājumam veidotais prototips satur 3 galvenās komponentes, kas ir atbildīgas par dažādu risinājuma aspektu realizēšanu. Komponentes var tikt sekojoši izdalītas:

- Manevru atpazīšanas komponente, kurai atbilst klase ManouverHandler
- Manevru saglabāšanas komponente, kurai atbilst klase PersistanceHandler
- Brauciena pārvaldības komponente, kurai atbilst klase TripHandler

Sistēmas struktūra, izdalītās komponentes, kā arī to metodes, atribūti un savstarpējās saistības, ir attēlotas klašu diagrammā (skatīt attēlu 5.1).

Atbilstoši piedāvātajam risinājumam, meklējamo manevru glabāšana un konfigurēšana ir nodalīta no sistēmas realizācijas. Ierakstītie manevri tiek saglabāti teksta failos JSON datu formātā. Atpazīstamie manevri tāpat tiek ielasīti no failiem, taču atpazīstamajiem manevriem nepieciešams pievienot konfigurācijas informāciju, kas nepieciešama atpazīšanas procesā.

Sistēmā tiek izmantots arī delegācijas pieeja, lai deleģētu dažādu uzdevumu veikšanu, un savstarpēji abstrahētu klases un komponentes.



5.1. att.: **Sistēmas klašu diagramma**

5.2 Prototipa realizācija

Lietotnes realizācijā tiek izmantoti standarta bibliotēkas, kas iekļautas Cocoa Touch ietvarā, kā arī bibliotēkas "Objection" [33.] , kas domāta atkarību injekcijai (dependency injection), un "JSONModel" [34.], kas atvieglo JSON formatētu datu pieskaņošanu objektu klasēm. Detalizētāk tiks apskatītas gan svarīgākās sistēmas komponentes, gan sistēmas saskarne.

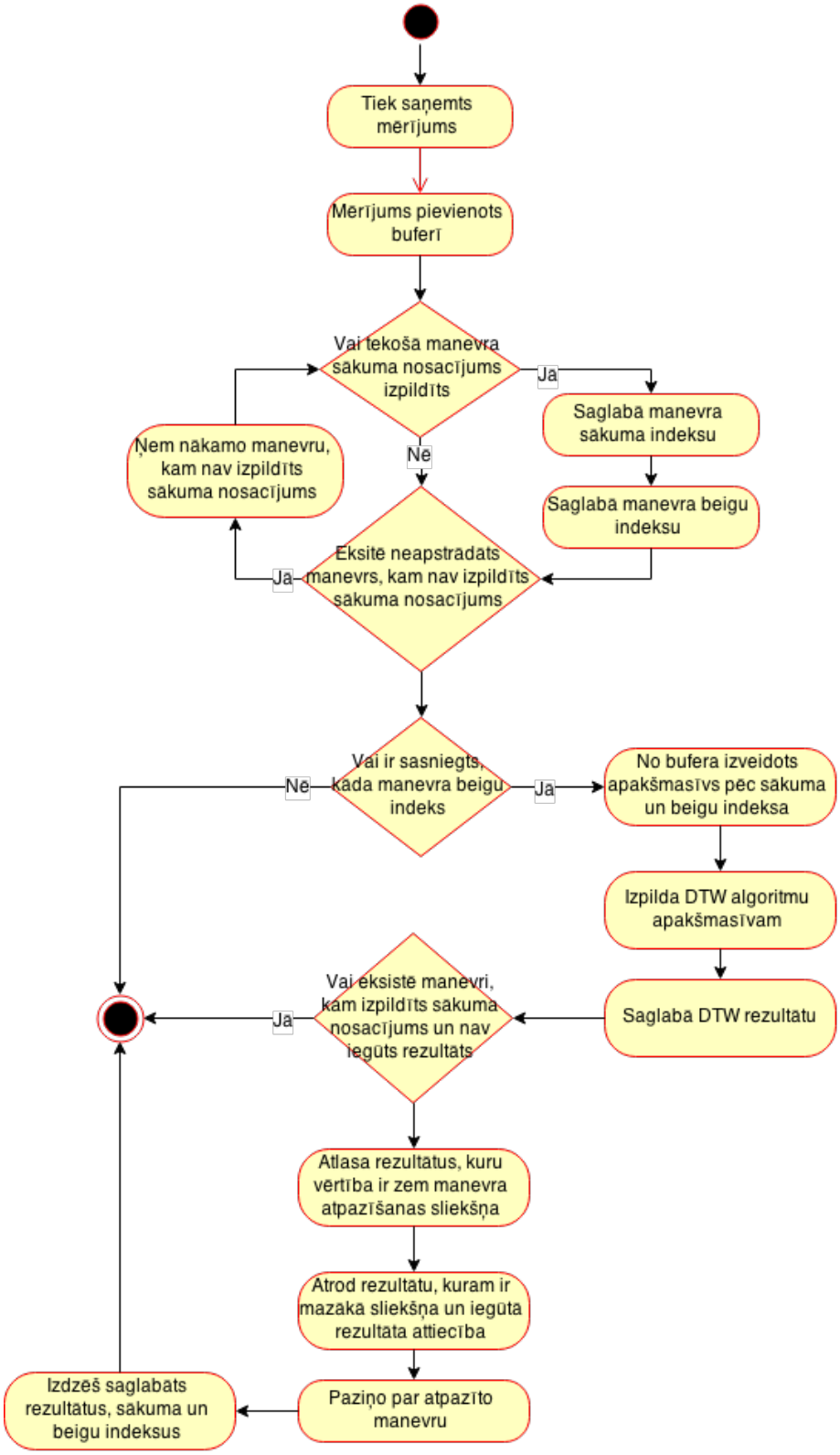
5.2.1 Sistēmas komponentes

TripHandler komponentei tiek padotas darbības, kuras lietotājs izsauc izmantojot lietotnes saskarni, un šī klase ir atbildīga par to, lai tiktu sākta sensoru monitorēšana, kā arī, lai sensoru dati tiktu nodoti piemērotajai komponentei. Šīs komponentes integrācija ar ManouverHandler un PersistanceHandler realizēta izmantojot interfeisu MotionHandler. Pieeja sensoru datiem tiek nodrošināta ar iebūvēto ietvaru CoreMotion un klasi CMMotionManager. Intervāls ar kādu tiek saņemti sensoru dati ir iestatīts 10 reizes sekundē. Šī komponente ļauj abstrahēt manevru saglabāšanas un atpazīšanas komponentes gan no veida kā tiek iegūti sensoru mērījumi, gan no tā veida kā tiek iniciēti šo komponenšu realizētie procesi un darbības.

PersistanceHandler komponente veic sensoru mērījumu saglabāšanu JSON datu formātā un ierakstīšanu failā. Šī komponente apstrādā ievadi par manevra sākšanos, par manevra beigām un saglabāto manevru eksportēšanu. Komponente implementē MotionHandler interfeisu priekš sensoru mērījumu saņemšanas.

ManouverHandler komponente ir nozīmīgākā sistēmas daļa, kurā realizēta galvenā manevru atpazīšanai nepieciešama loģika. Arī šī komponente implementē MotionHandler interfeisu, lai saņemtu tekošos sensoru mērījumus. Atpazīstamie manevri tiek pārvaldīti ar TargetManouver klasi. Klases TargetManouver instances tiek izveidotas no JSON formāta datiem, kas tiek ielasīti no failiem, un tiek paturētas atmiņā visu sistēmas darbības laiku. Šīs komponentes realizācija izmantota darbību deleģēšanas pieeja, lai informētu par manevru atpazīšanas rezultātiem. Komponentei tiek padots delegāts, kurš implementē ManouverFeedbackHandler interfeisu un šī komponente var izmantot interfeisā definētās metodes, un abstrahēties no ar saskarni saistītām darbībām, kas tiek veiktas, lai attēlotu rezultātu lietotājam. Līdzīgā veidā TargetManouver klasē tiek realizēta manevra sākuma pārbaude, un darbības, kuras nepieciešams veikt, kad sākuma nosacījums ir izpildīts tiek deleģētas izmantojot interfeisu TargetManouverHandler. Mērījumu apstrāde, kas tiek veikta

ManouverHandler komponentē, ir detalizēti ilustrēta izmantojot aktivitāšu diagrammu 5.2

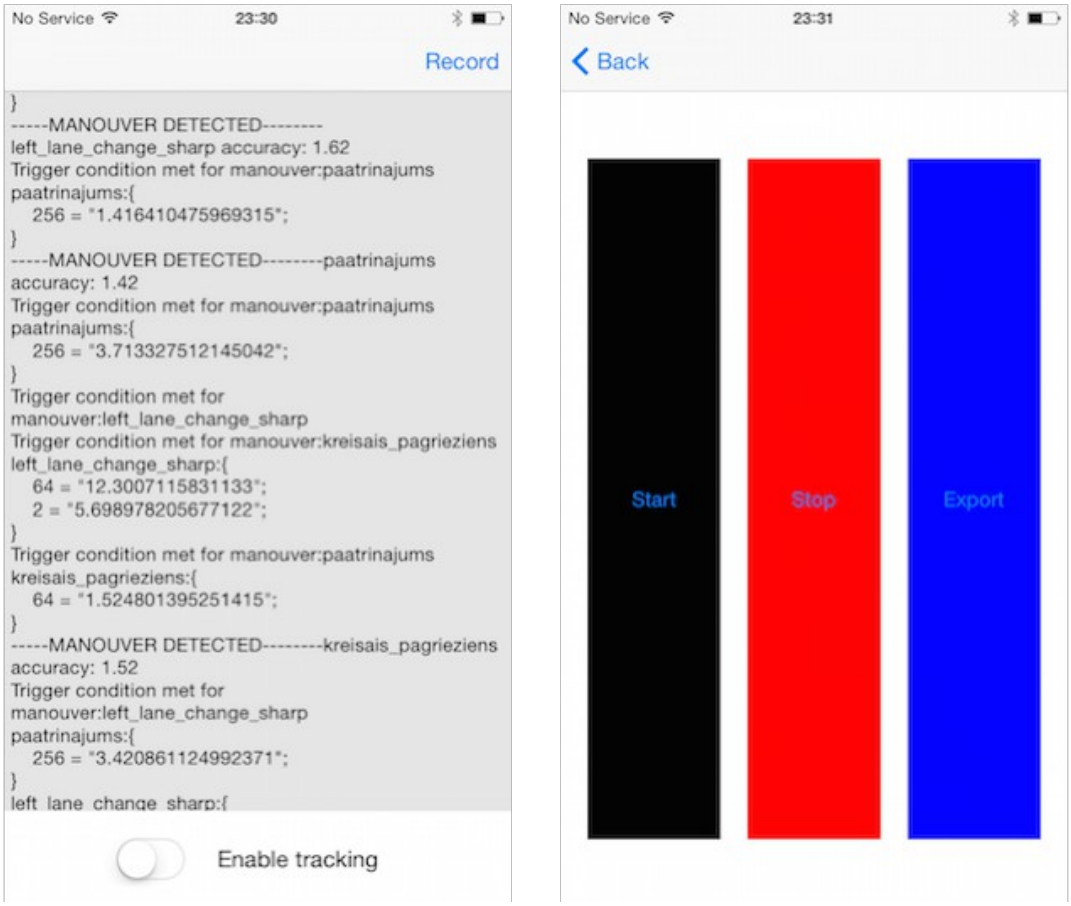


5.2. att.: **Mērījumu apstrādes aktivitāšu diagramma**

attēlā.

5.2.2 Sistēmas saskarne

Tā kā sistēmas izstrādes mērķis ir pārbaudīt piedāvātā risinājuma realizāciju, un praksē notestēt automatizētu manevru atpazīšanu, saskarne tika realizēta pēc iespējas vienkāršākā veidā. Lietotne satur divus skatus – manevru atpazīšanas skatu un manevru ierakstīšanas skatu (skatīt 5.3 attēlu). Saskarne nedod iespēju ierakstīšana un atpazīšana darbības veikt vienlaicīgi. Manevru atpazīšanas skata galvenā komponente ir teksta lauks, kur tiek izvadīta informācija par manevru atpazīšanas gaitu un rezultātiem, lai varētu tikt veikta risinājuma darbības analīze.



5.3. att.: **Risinājuma prototipa saskarne**

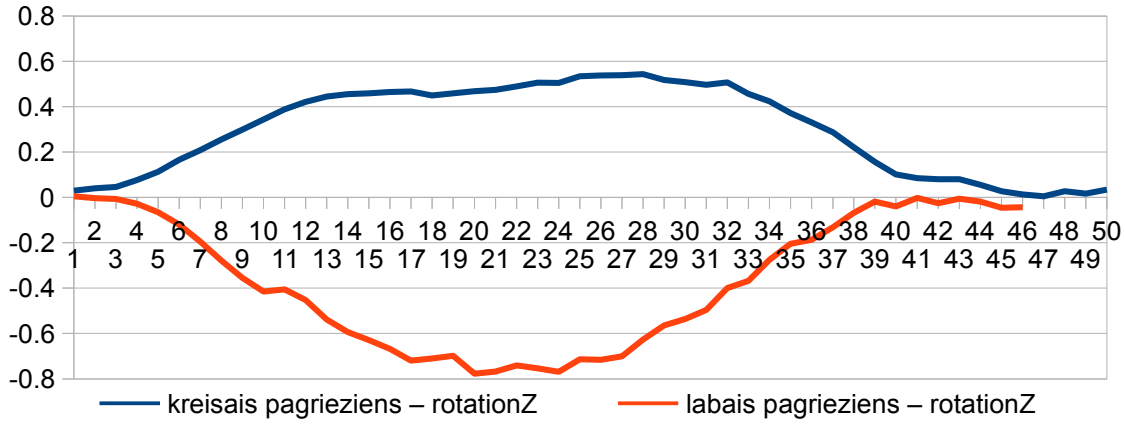
5.3 Prototipa testēšana un uzlabošana

Sistēmas testēšana tika veikta vairākos etapos, ieviešot izmaiņas un uzlabojumus, gan manevru atpazīšanā, gan manevru konfigurācijā. Testēšanas galvenie uzdevumi bija noteikt cik precīzi strādā manevru atpazīšanas funkcionalitāte, kā arī noteikt nepilnības un trūkumus, gan sistēmas implementācijā, gan manevru konfigurāciju, kurus novēršot varētu tikt uzlabota manevru atpazīšanas precizitāte.

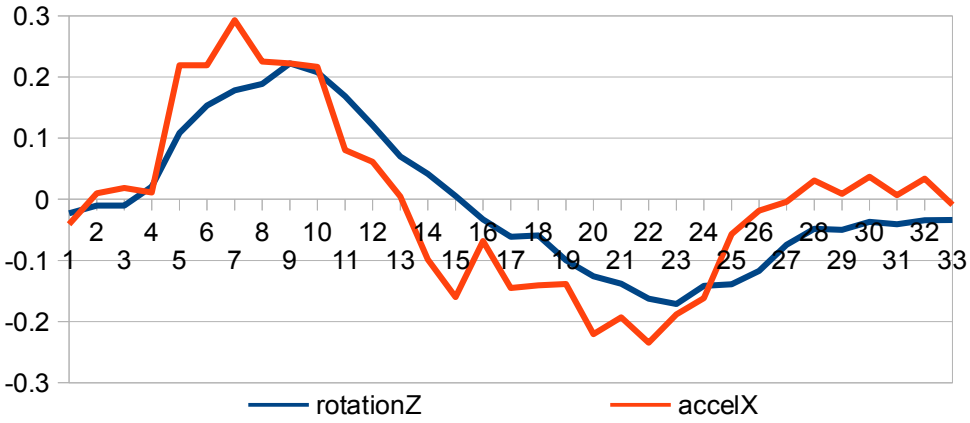
Testēšana tika veikta autovadīšanas laikā. Testēšanā tika izmantots viedtālrunis iPhone 6, kurš bija novietots uz automašīnas priekšējā paneļa guļus stāvoklī. Testu veikšanai tika izmantota Volvo automašīna ar dīzeļa dzinēju. Vajadzētu testu rezultātus pievienot pielikumā.

5.3.1 Pirmais testēšanas etaps

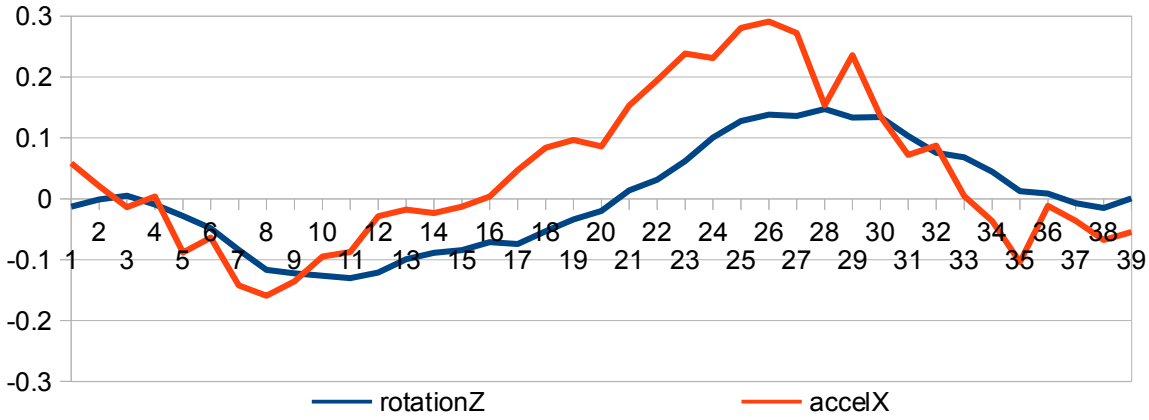
Pirmajā etapā sistēma tika pielāgota 4 manevru atpazīšanai – labais pagrieziens (skatīt attēlu 5.4), kreisais pagrieziens (skatīt attēlu 5.4), straujš pārkārtošanās manevrs uz kreiso braukšanas joslu (skatīt attēlu 5.5), straujš pārkārtošanās manevrs uz labo braukšanas joslu (skatīt attēlu 5.6). Šie manevri tika iegūti izmantojot lietotnes manevru ierakstīšanas funkcionalitāti. Ierakstīti un saglabāti tika vairāki līdzīgi manevri. Tie tika savstarpēji salīdzināti un atrasts tāds manevra attēlojums, kas vismazāk atšķiras no pārējiem un varētu tikt izmantots kā bāzlīnija visiem šāda tipa manevriem. Visu manevru atpazīšanai tika izmantota maksimālā sliekšņa vērtība 6,0.



5.4. att.: **Pagriezienu manevri**



5.5. att.: **Pārkārtošanās pa kreisi manevrs**



5.6. att.: **Pārkārtošanās pa labi manevrs**

Pēc testa brauciena veikšanas tika iegūtie rezultāti ir apkopoti 5.1 tabulā. Testa brauciena maršruts pievienots 1. pielikumā un manevru apkopojums pievienots 2. pielikumā. Ņemot vērā lielo neprecizitāšu skaitu pārkārtošanās manevru atpazīšanā un analizējot precizitāti ar kādu tika atpazīti pareizie pārkārtošanās manevri, tika izdarīts secinājums samazināt pārkārtošanās manevru atpazīšanas slieksni līdz vērtībai 2. Pārkārtošanās manevriem ir zems sākuma punkta atpazīšanas slieksnis, kā arī manevra attēlojuma grafikam ir maza novirze no 0 vērtības, dēļ tā daudzi nelieli pagriezieni un izmaiņas trajektorija tika atpazītas kā pārkārtošanās manevri.

5.1. tabula

Pirmā testēšanas etapa rezultāti

	Kopā atpazīto reižu skaits	Nepareizi atpazīto reižu skaits	Neatpazīto reižu skaits
Kreisais pagrieziens	6	0	0
Labais pagrieziens	5	1	2
Pārkārtošanās pa kreisi	11	6	0
Pārkārtošanās pa labi	8	4	0

5.3.2 Otrais testēšanas etaps

Šajā testēšanas etapā tika pievērsta pastiprināta uzmanība pārkārtošanās manevru testēšanai, lai būtu samazinājies manevru skaits, kas tiek nepareizi uzskatīti par pārkāošanos. Testa brauciena maršruts pievienots 3. pielikumā un manevru apkopojums pievienots 4. pielikumā. Testēšanai tika izmantots cits maršruts nekā pirmajā etapā. Šajā maršrutā iekļauti vairāki lēzeni līkumi, kas iepriekš tiktu nepareizi atpazīti kā pārkārtošanās manevri. Otrā etapa testēšanas rezultāti apkopoti 5.2 tabulā. Pēc rezultātiem var secināt, ka pārkārtošanās manevru atpazīšanas kvalitāte ir uzlabojusies.

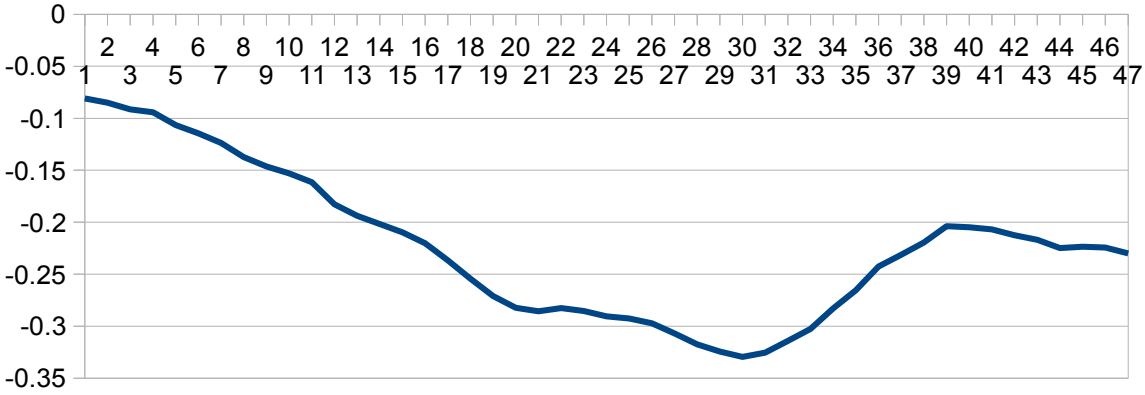
5.2. tabula

Otrā testēšanas etapa rezultāti

	Kopā atpazīto reižu skaits	Nepareizi atpazīto reižu skaits	Neatpazīto reižu skaits
Kreisais pagrieziens	4	0	0
Labais pagrieziens	3	0	1
Pārkārtošanās pa kreisi	1	1	0
Pārkārtošanās pa labi	1	0	0

5.3.3 Trešais testēšanas etaps

Šajā etapā tika papildināta atpazīstamo manevru kopa un pievienots paātrinājuma manevrs. Paātrinājuma manevrā tika saglabāts salīdzinoši straujš paātrinājums no 10 līdz 60 km/h, kurā iekļauta pārnesumu pārslēgšana. Manevra grafiskais attēlojums ilustrēts attēlā 5.7. Manevra atpazīšanas slieksnis tika izvēlēts 2,0. Trešajā testēšanas etapā tika veikti vairāki testa braucieni, un bija novērojamas vairākās neprecizitātes, kas veidojās, dēļ ierobežojuma, ka vienlaicīgi var tikt atpazīts tikai viens manevrs. Vairāki pārkārtošanās un pagrieziena manevri pēc kuriem sekoja paātrinājums, netika atpazīti, jo augstāka precizitāte sakrita tieši paātrinājuma manevram.



5.7. att.: **Paātrinājuma manevrs**

5.3.4 Ceturtais testēšanas etaps

Šajā etapā tika izņemts ierobežojums, ka vienlaicīgi var tikt atpazīts tikai viens manevrs. Iepriekšējos testos netika novērots gadījums viens manevrs ar pieņemamu precizitāti tiek uzskatīts par 2 līdzīgiem manevriem, tādēļ šis ierobežojums tika uzskatīts par lieku. Testējot tika atkārtots pirmajā etapā veiktas maršruts ar nelielām izmaiņām, viena no kurām ir, ka tika pieļauta kļūda un veikts pagrieziens nepareizā ielā, tādēļ nācās apgriezties, kā rezultāta kopumā tika veikti papildus 2 labie pagriezieni, 1 paātrinājums un 1 kreisais pagrieziens. Rezultāti ir apkopoti 5.3 tabulā. Testa brauciena maršruts pievienots 5. pielikumā un manevru apkopojums pievienots 6. pielikumā. Pēc testēšanas rezultātiem var secināt, ka ir uzlabojusies manevru atpazīšanas precizitāte gadījumos, kad pēc kārtas tiek veikti vairāki manevri.

5.3. tabula

Ceturtā testēšanas etapa rezultāti

	Kopā atpazīto reižu skaits	Nepareizi atpazīto reižu skaits	Neatpazīto reižu skaits
Kreisais pagrieziens	9	1	0
Labais pagrieziens	11	0	1
Pārkārtošanās pa kreisi	3	0	0
Pārkārtošanās pa labi	3	0	1
Paātrinājums	7	0	1

5.3.5 Testēšanas secinājumi

Pēc testēšanas veikšanas un rezultātu analizēšanas var izdarīt vairākus secinājumus par izstrādāto prototipu un par risinājumu kopumā. Programmatūras mērķis un galvenais uzdevums tika veiksmīgi izpildīts. Lietotne ir spējīga, ar zemu kļūdu skaitu, atpazīt visus manevrus, kas tika pievienoti.

Testēšanas gaitā tika identificēta nepilnība piedāvātajā risinājumā, ka vienā laika intervālā var notikt tikai viens manevrs. Pēc nepilnības novēršanas tika novēroti uzlabojumi manevru atpazīšanas precizitāte. Pēc testēšanas gaitas un tās laikā veiktajiem uzlabojumiem var secināt, ka realizētajā risinājumā nozīmīgākais aspekts, kas ietekmē manevru atpazīšanas kvalitāti, ir manevra bāzlīnija un manevra konfigurācija, tādējādi dodot lietotājam visus nepieciešamos rīkus, lai pielāgotu izstrādāto sistēmu, konkrētiem manevriem un lietojuma gadījumam.

NOBEIGUMS UN SECINĀJUMI

Šī darba ietvaros tika izpētīta problēma – automatizēta autovadīšanas manevru noteikšana izmantojot viedtālruni. Problēmas risināšanai tika veikta detalizēta esošās situācijas izpēte, kurā tika apskatīti gan akadēmiski pētījumi, gan jau realizēta programmatūra. Esošās situācijas izpētes rezultāta tika nonākts pie secinājuma, ka autora izvirzītajai problēmai šobrīd nav pieejams praktiski realizēts risinājums.

Lai izstrādātu problēmas risinājumu tika veikta viedtālruņos pieejamo sensoru izpēte, kā arī praktisks eksperiments, kura ietvaros ar automašīnu tika veikti dažādi manevru, un analizēti sensora mērījumi. Praktiskā eksperimenta mērķis bija identificēt pazīmes sensoru mērījumos, pēc kurām varētu automatizētā veidā identificēt dažādus manevrus. Eksperimenta mērķis tika veiksmīgi izpildīts, un tika noteikti gan konkrēti sensori, gan šo sensoru mērījumi, pēc kuriem var tikt atpazīti konkrēti manevri.

Balstoties uz veikto teorētisko analīzi un eksperimenta rezultātiem autors izstrādāja savu risinājumu, darbā izvirzītajai problēmai. Risinājumā tika piedāvāts specifisku, ar manevru atpazīšanu saistītu, problēmu risinājumi, manevru atpazīšanas automatizācijā svarīgu nosacījumu specifikācija, kā arī citu nozīmīgu aspektu analīze. Risinājumā piedāvātais veids, kā realizēt manevru atpazīšanu, tika plānots, lai tas būtu vienkārši paplašināms un pielāgojams, dažādiem lietojumu gadījumiem.

Lai pārbaudītu piedāvāto risinājumu, atbilstoši risinājuma principiem un metodoloģijai tika izstrādāta iOS lietotne, kas veic automātisku autovadīšanas manevru atpazīšanu. Izstrādātā lietotne tika realizēta pēc komponent-bāzētas programmatūras izstrādes pieejas, un tā tika veidota kā vispārīga programmatūra, kas var tikt pielāgota dažādu autovadīšanas manevru atpazīšanai. Programmatūra tika praktiski pārbaudīta vairāku manevru atpazīšanā un testēšanas rezultātā tika secināts, ka pēc risinājuma izstrādātā programmatūra veiksmīgi veic dažādu autovadīšanas manevru atpazīšanu, un, ka darbā izvirzītie mērķi un uzdevumi ir veiksmīgi sasniegti. Darba rezultātā ir izveidota novitāte – vispārīgs risinājums un praktiski realizēta programmatūra priekš autovadīšanas manevru automatizētas noteikšanas.

IZMANTOTĀ LITERATŪRA

1. M. Fazeen, B. Gozick, R. Dantu, M. Bhukhiya, M. C. Gonzalez, *"Safe Driving Using Mobile Phones," IEEE Transactions on Intelligent Transportation Systems*, Sept. 2012 – [atsauce 04.01.2015.]. Pieejams: http://humnet.scripts.mit.edu/wordpress2/wp-content/uploads/2010/10/IEEE_safedrivingusingmobilephones.pdf
2. Derick A. Johnson and Mohan M. Trivedi, *"Driving Style Recognition Using a Smartphone as a Sensor Platform" IEEE Transactions on Intelligent Transportation Systems*, Oct. 2011 – [atsauce 04.01.2015.]. Pieejams: http://cvrr.ucsd.edu/publications/2011/Johnson_ITSC2011.pdf
3. N. Promwongsa, P. Chaisatsilp, S. Supakwong C. Saiprasert, T. Pholprasit, P. Prathombutr *"Automatic Accelerometer Reorientation for Driving Event Detection Using Smartphone" Thammasart University* [atsauce 04.01.2015.]. Pieejams: http://www.itsasiapacificforum2014.co.nz/files/3314/0194/4873/Abstract_-_Automatic_Accelerometer_Reorientation_for_Driving_Event_Detection_Using_Smartphone_by_Passakon_Prathombutr.pdf
4. A.Sathyanarayana, S.O.Sadjadi, J.H.L.Hansen *"Leveraging Sensor Information from Portable Devices towards Automatic Driving Maneuver Recognition" International IEEE Conference on Intelligent Transportation Systems*, September 2012 [atsauce 04.01.2015.]. Pieejams: http://www.utdallas.edu/~jxh052100/Publications/CP-ITSC-2012-SensorInfoPortDevicesDrivingRecog-AmarOmidHansen-Alaska-0159-Sept2012.pdf
5. Driver Feedback™ — Android lietotnes pakalpojumā Google Play [tiešsaiste]. [atsauce 01.03.2015]. Pieejams - https://play.google.com/store/apps/details?id=com.statefarm.driverfeedback
6. Driver Feedback on the App Store on iTunes [tiešsaiste]. [atsauce 01.04.2015]. Pieejams - https://itunes.apple.com/lv/app/driver-feedback/id899993049?mt=8
7. Augmented Driving on the App Store on iTunes [tiešsaiste]. [atsauce 01.04.2015]. Pieejams - https://itunes.apple.com/gb/app/augmented-driving/id366841514?mt=8
8. Driver Guard — Android lietotnes pakalpojumā Google Play [tiešsaiste]. [atsauce 01.04.2015]. Pieejams - https://play.google.com/store/apps/details?id=com.badrit.cv.vehicledetect
9. Event Handling Guide for iOS [tiešsaiste]. [atsauce 08.01.2015]. Pieejams – https://developer.apple.com/library/ios/documentation/EventHandling/Conceptual/Eve

ntHandlingiPhoneOS/motion_event_basics/motion_event_basics.html

10. Alasdair Allan “Basic sensors in iOS”. O'Reilly Media, Publ. CA, Sebastopol, July 2011.
11. What Is An Accelerometer? Applications, Operation Principles And Types [tiešsaiste]. [atsauce 08.01.2015]. Pieejams – http://tekedia.com/20102/accelerometer-part-1
12. Introduction to Microelectromechanical Systems (MEMS) [tiešsaiste]. [atsauce 08.01.2015]. Pieejams – http://compliantmechanisms.byu.edu/content/introduction-microelectromechanical-systems-mems
13. How a Gyro Works [tiešsaiste]. [atsauce 09.01.2015]. Pieejams – https://learn.sparkfun.com/tutorials/gyroscope/how-a-gyro-works
14. Magnetometer [tiešsaiste]. [atsauce 09.01.2015]. Pieejams – http://en.wikipedia.org/wiki/Magnetometer
15. “Magnetometer basics for mobile phone applications” B.Y.Cai, Y.Zhao, X.Ding, J.Fennelly, *ELECTRONIC PRODUCTS*, February 2012 [atsauce 17.01.2015]. Pieejams - http://www.memsic.com/userfiles/files/publications/Articles/Electronic_Products_Feb_%202012_Magnetometer.pdf
16. Measuring Magnetic Fields [tiešsaiste]. [atsauce 17.01.2015]. Pieejams – https://www.nde-ed.org/EducationResources/CommunityCollege/MagParticle/Physics/Measuring.htm
17. Global Positioning System [tiešsaiste]. [atsauce 17.01.2015]. Pieejams – http://en.wikipedia.org/wiki/Global_Positioning_System
18. Global Positioning System - GPS [tiešsaiste]. [atsauce 17.01.2015]. Pieejams – http://www.wirelessdictionary.com/wireless_dictionary_GPS_definition.html
19. Android Fragmentation Visualized (August 2014) [tiešsaiste]. [atsauce 17.01.2015]. Pieejams – http://opensignal.com/reports/2014/android-fragmentation
20. Core Motion Framework Reference [tiešsaiste]. [atsauce 23.01.2015]. Pieejams – https://developer.apple.com/library/ios/documentation/CoreMotion/Reference/CoreMotion_Reference/index.html#//apple_ref/doc/uid/TP40009686
21. CMDeviceMotion, Nate Cook, October 28th, 2014 [tiešsaiste]. [atsauce 23.01.2015]. Pieejams – http://nshipster.com/cmdevicemotion/
22. Core Location Framework Reference [tiešsaiste]. [atsauce 23.01.2015]. Pieejams – https://developer.apple.com/library/ios/documentation/CoreLocation/Reference/CoreLocation_Framework/

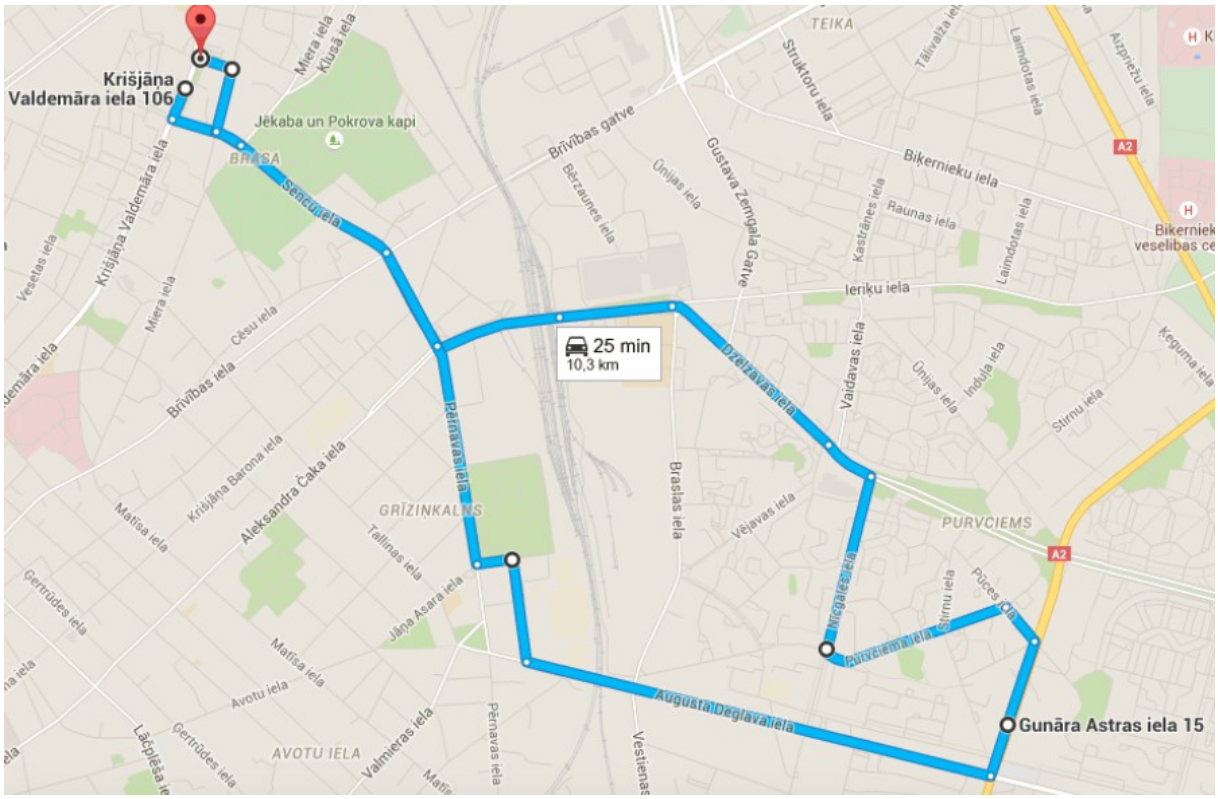
23. SensorLog By Bernd Thomas [tiešsaiste]. [atsauce 23.01.2015]. Pieejams – https://itunes.apple.com/us/app/sensorlog/id388014573?mt=8
24. Elena Tsiporkova, "Dynamic Time Warping Algorithm" [tiešsaiste]. [atsauce 23.01.2015]. Pieejams – http://www.mathcs.emory.edu/~lxiong/cs730_s13/share/slides/searching_sigkdd2012_DTW.pdf
25. Pavel Senin, "Dynamic Time Warping Algorithm Review", Information and Computer Science Department University of Hawaii, Dec 2008 - [atsauce 03.03.2015.]. Pieejams: http://www2.hawaii.edu/~senin/assets/papers/DTW-review2008draft.pdf
26. Mathieu's log, Introduction to Dynamic Time Warping [tiešsaiste]. [atsauce 08.04.2015]. Pieejams – http://www.mblondel.org/journal/2009/08/31/dynamic-time-warping-theory/
27. Similarity Measures and Dimensionality Reduction Techniques for Time Series Data Mining [tiešsaiste]. [atsauce 01.05.2015]. Pieejams – http://www.intechopen.com/books/advances-in-data-mining-knowledge-discovery-and-applications/similarity-measures-and-dimensionality-reduction-techniques-for-time-series-data-mining
28. JSON: The Fat-Free Alternative to XML [tiešsaiste]. [atsauce 03.05.2015]. Pieejams – http://www.json.org/xml.html
29. NSJSONSerialization Class Reference [tiešsaiste]. [atsauce 05.05.2015]. Pieejams – https://developer.apple.com/library/ios/documentation/Foundation/Reference/NSJSONSerialization_Class/index.html
30. Magical Data Modelling Framework for JSON [tiešsaiste]. [atsauce 10.05.2015]. Pieejams – https://github.com/icanzilb/JSONModel/blob/master/README.md
31. deviceMotionUpdateInterval - CMMotionManager Class Reference [tiešsaiste]. [atsauce 11.05.2015]. Pieejams – https://developer.apple.com/library/ios/documentation/CoreMotion/Reference/CMMotionManager_Class/#//apple_ref/occ/instp/CMMotionManager/deviceMotionUpdateInterval
32. Stack Overflow [tiešsaiste]. [atsauce 11.05.2015]. Pieejams – http://stackoverflow.com/questions/11797857/safe-update-interval-for-startdevicemotionupdatestoqueuewithhandler
33. Objection by atomicobject [tiešsaiste]. [atsauce 12.05.2015]. Pieejams –

http://objection-framework.org/

34. JSONModel - Magical Data Modelling Framework for JSON on iOS and OSX! [tiešsaiste]. [atsauce 12.05.2015]. Pieejams – http://www.jsonmodel.com/

PIELIKUMI

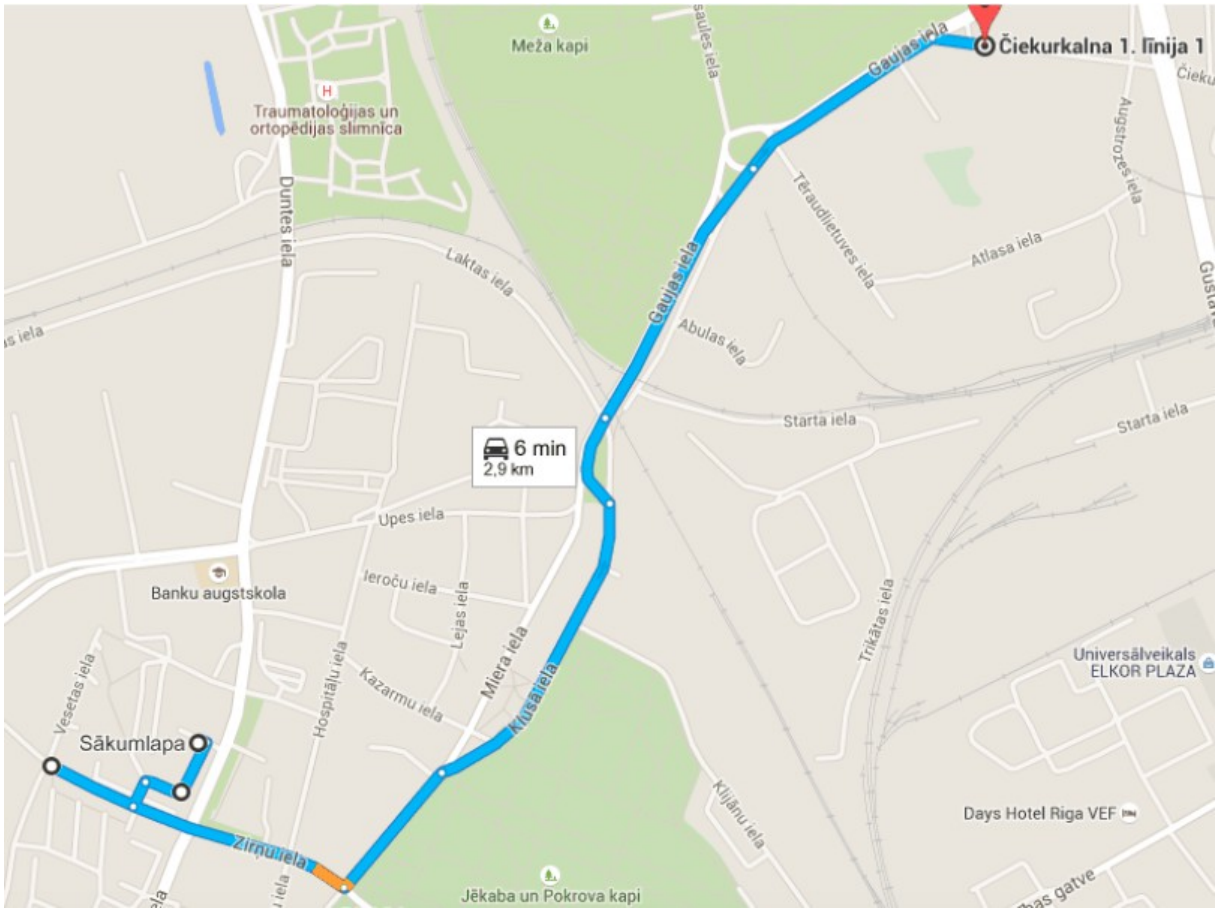
1. pielikums. Pirmā testēšanas etapa maršruts



2. pielikums. Pirmā testēšanas etapa apkopojums

Valid:	kreisais_pagrieziens(5.03)
Invalid:	left_lane_change_sharp(2.39)
Valid:	labais_pagrieziens(2.92)
Valid:	left_lane_change_sharp(1.51)
Valid:	kreisais_pagrieziens(4.60)
Valid:	right_lane_change_sharp(2.37)
Invalid:	right_lane_change_sharp(2.39)
Invalid:	left_lane_change_sharp(2.43)
Valid:	left_lane_change_sharp(1.53)
Valid:	kreisais_pagrieziens(0.82)
Valid:	right_lane_change_sharp(1.71)
Invalid:	right_lane_change_sharp(2.46)
Invalid:	right_lane_change_sharp(4.54)
Invalid:	left_lane_change_sharp(2.37)
Invalid:	left_lane_change_sharp(2.77)
Missed right turn	
Valid:	left_lane_change_sharp(1.82)
Valid:	kreisais_pagrieziens(1.48)
Invalid:	left_lane_change_sharp(2.26)
Invalid:	left_lane_change_sharp(4.31)
Valid:	left_lane_change_sharp(1.49)
Valid:	labais_pagrieziens(3.98)
Valid:	labais_pagrieziens(4.77)
Valid:	kreisais_pagrieziens(4.49)
Valid:	right_lane_change_sharp(2.32)
Valid:	labais_pagrieziens(4.25)
Valid:	right_lane_change_sharp(1.33)
Invalid:	labais_pagrieziens(3.34)
Valid:	left_lane_change_sharp(1.37)
Valid:	right_lane_change_sharp(2.45)
Missed right turn	
Valid:	kreisais_pagrieziens(0.86)
Valid:	labais_pagrieziens(2.51)

3. pielikums. Otrā testēšanas etapa maršruts



4. pielikums. Otrā testēšanas etapa apkopojums

Valid:	labais_pagrieziens(5.89)
Valid:	right_lane_change_sharp(1.63)
Valid:	labais_pagrieziens(2.53)
Valid:	kreisais_pagrieziens(1.56)
Invalid:	left_lane_change_sharp(1.62)
Valid:	labais_pagrieziens(4.65)

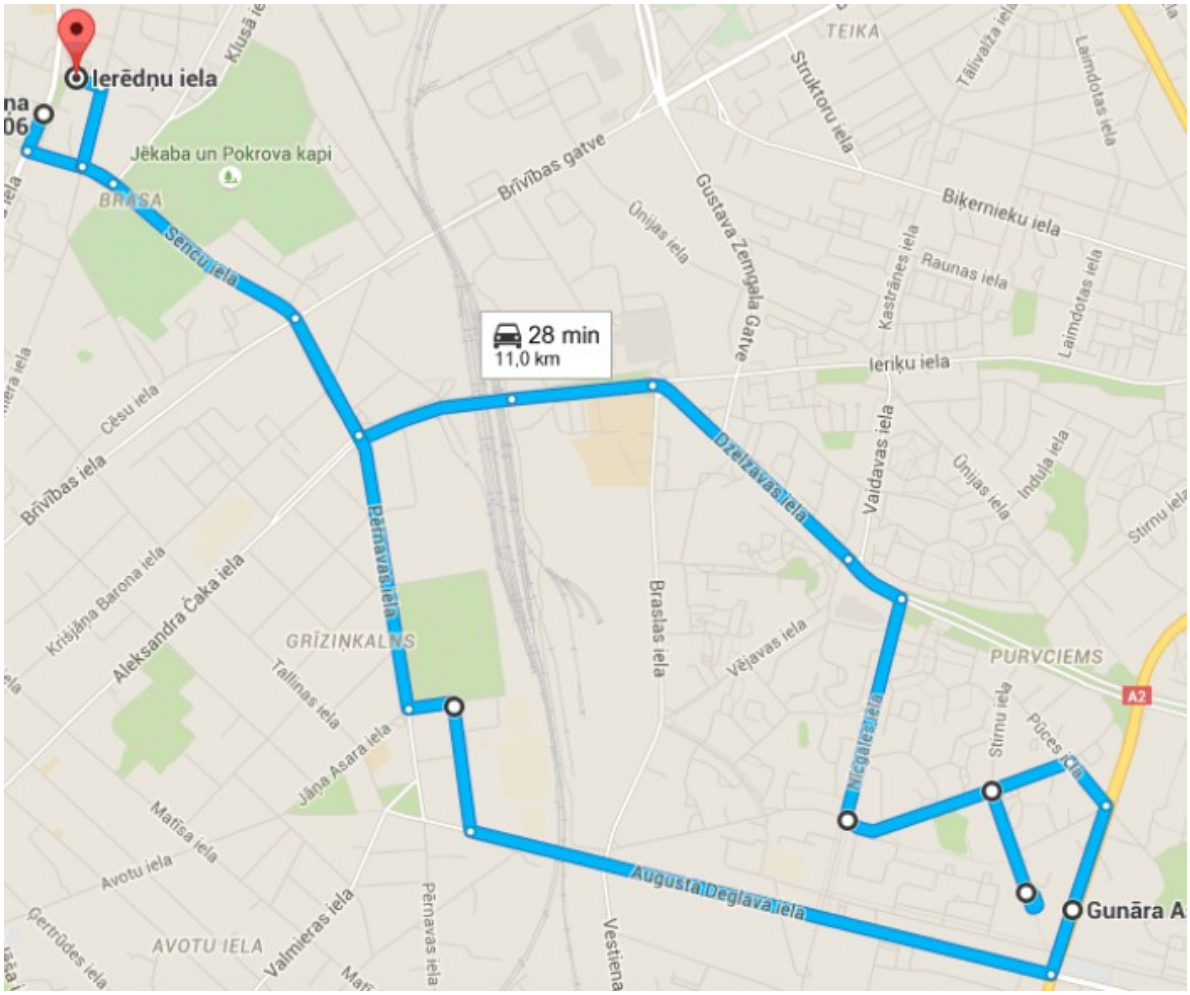
Missed turning around (AS EXPECTED)

Valid:	left_lane_change_sharp(1.22)
Valid:	kreisais_pagrieziens(1.43)
Valid:	kreisais_pagrieziens(4.75)

Missed right turn (Looking at the map this may be as expected)

Valid:	labais_pagrieziens(1.58)
Valid:	kreisais_pagrieziens(1.04)

5. pielikums. Ceturtā testēšanas etapa maršruts



6. pielikums. Ceturtā testēšanas etapa apkopojums

Valid: kreisais_pagrieziens accuracy: 2.61
Valid: paatrinajums accuracy: 1.26
Valid: kreisais_pagrieziens accuracy: 1.64
Valid: left_lane_change_sharp accuracy: 1.62
Valid: paatrinajums accuracy: 1.42
MISSED RIGHT TURN
MISSED ACCELERATION
Valid: kreisais_pagrieziens accuracy: 1.52
WRONG kreisais_pagrieziens accuracy: 5.40
Valid: paatrinajums accuracy: 1.49
Valid: labais_pagrieziens accuracy: 2.47
Valid: kreisais_pagrieziens accuracy: 3.33
Valid: paatrinajums accuracy: 1.12
Valid: labais_pagrieziens accuracy: 3.53
Valid: labais_pagrieziens accuracy: 2.93
Valid: labais_pagrieziens accuracy: 4.98
Valid: labais_pagrieziens accuracy: 6.02
Valid: kreisais_pagrieziens accuracy: 2.21

-----DESTENATION

Valid: labais_pagrieziens accuracy: 5.39
Valid: paatrinajums accuracy: 0.79
Valid: right_lane_change_sharp accuracy: 1.75
Valid: labais_pagrieziens accuracy: 1.91
Valid: paatrinajums accuracy: 1.17
Valid: left_lane_change_sharp accuracy: 1.85
Missed right lane change - but the lane change was very slow
Valid: labais_pagrieziens accuracy: 5.64
Valid: kreisais_pagrieziens accuracy: 1.42
Valid: labais_pagrieziens accuracy: 1.38
Valid: paatrinajums accuracy: 1.05
Valid: right_lane_change_sharp accuracy: 1.66
Valid: labais_pagrieziens accuracy: 1.90
Valid: kreisais_pagrieziens accuracy: 1.92
Valid: left_lane_change_sharp accuracy: 1.68
Valid: right_lane_change_sharp accuracy: 1.52
Valid: kreisais_pagrieziens accuracy: 5.92
Valid: labais_pagrieziens accuracy: 5.79

7. pielikums. Izstrādātās lietotnes pirmkods un veikto testu izvade

Šim darbam pievienots kompaktdisks, kurā iekļauts izstrādātās lietojumprogrammas pirmkods. Kods atrodams mapē ManouverDetector. Pirmkods arī pieejams Git repozitorijā -

https://dembovskis@bitbucket.org/dembovskis/manouverdetector.git

Kompaktdiskā pievienoti arī 4 faili, kuri satur visu sistēmas izvadīto informāciju testu veikšanas laikā. No šīs informācijas tika veidoti iepriekšējos pielikumos atrodamie apkopojumi.

Maģistra darbs: **Autovadīšanas manevru noteikšana izmantojot viedtālruni**

Ar savu parakstu apliecinu, ka pētījums veikts patstāvīgi, izmantoti tikai tajā norādītie informācijas avoti un iesniegtā darba elektroniskā kopija atbilst izdrukai.

Autors: _____
(Autora paraksts)

Ar savu parakstu apliecinu, ka esmu lasījis augstāk minēto maģistra darbu un atzīstu to par **p i e m ē r o t u / n e p i e m ē r o t u** (nevajadzīgo svītrot) aizstāvēšanai Latvijas Universitātes datorzinātņu maģistrantūrā.

Darba vadītājs: _____
(Vadītāja paraksts)

Darbs iesniegts maģistrantūras sekretariātā _____.
(Iesniegšanas datums)

Ar šo es apliecinu, ka darba elektroniskā versija ir augšupielādēta LU informatīvajā sistēmā.

Studiju metodiķe: _____.
(Metodiķes paraksts)

Recenzents: _____
(Akad. amats, zin. grāds, vārds, uzvārds)

Darbs aizstāvēts maģistra gala pārbaudījuma komisijas sēdē

_____ prot. Nr. _____
(Darba aizstāvēšanas datums)

Komisijas sekretārs: _____
(Sekretāra paraksts)