UNIVERZA V LJUBLJANI
FAKULTETA ZA MATEMATIKO IN FIZIKO

Matematika – 2. stopnja

Andrej Borštnik

Ocenjevanje kvalitete vožnje iz GPS podatkov

Magistrsko delo

Mentor: doc. dr. Alen Orbanić

Ljubljana, 2017

Kazalo

Program dela

Kvaliteta vožnje voznika/vozila je pomemben podatek tako za voznike, kot za delodajalce, zavarovalnice ipd. Ena izmed glavnih vrednosti tega podatka je ocena tveganja uporabe vozila/voznika, npr. kako se različna vozila obnesejo pod podobnimi pogoji, katerega se torej splača kupiti, po kakšni ceni zavarovati vozilo nekega voznika ipd.

Kandidat naj na osnovi velike podatkovne zbirke GPS sledi identificira primerne značilke. Pri tem naj uporabi dostopne industrijske standarde. Izpelje in utemelji naj metodologijo za ocenjevanje zanesljivosti podatkov in izbranih značilk. Osredotoči naj se tudi na izboljšanje kakovosti GPS sledi s pomočjo algoritmov za projeciranje GPS sledi na cestno omrežje (ang. map matching). Algoritme naj tudi podrobneje predstavi.

S pomočjo razvite metodologije primerja in analizira kakovost voženj različnih tipov voznikov. Pri tem naj uporabi naslednje vire:

- W. J. Stewart, *Probability, Markov chains, queues, and simulation: the mathematical basis of performance modeling*, Princeton University Press, 2009.
- M. A. Quddus, W. Y. Ochieng, R. B. Noland, *Current map-matching algorithms for transport applications: State-of-the art and future research directions*, Transportation research part c: Emerging technologies **15.5**, 2007, str. 312-328.

doc. dr. Alen Orbanić

Ocenjevanje kvalitete vožnje iz GPS podatkov

Povzetek

Skozi delo smo prepoznali pomembne značilke, za katere ocenimo, da vplivajo na varnost, udobnost in kvaliteto vožnje. Na podlagi GPS sledi, ki jih zberemo in obdelamo, izračunamo in ocenimo zanesljivost izračunanih značilk. Da bodo GPS sledi čimbolj natančne, jih z algoritmi za preslikavo na cestna omrežja preslikamo na (digitalno) cestno omrežje. Te algoritme tudi podrobneje opišemo.

Iz izračunanih značilk primerjamo kvaliteto vožnje različnih voznikov. Opazimo očitno razliko med poklicnimi in ostalimi vozniki. Zaključimo, da je v splošnem glavni razlikovalec med vozniki značilka varnosti, voznike pa uspešno razporedimo v tri skupine.

- Tiste, ki vozijo varno, tj. tiste, ki pravočasno predvidijo kako se promet obnaša in primerno reagirajo,
- tiste, ki vozijo nevarno in udobno, tj. tiste, ki vozijo nevarno, vendar ne pospešujejo prekomerno,
- ter tiste, ki vozijo varno in neudobno.

Assessing ride quality from GPS data

Abstract

In this work we identify statistics which estimate the safety, comfort, and quality of a ride. From collected and processed GPS tracks we compute these statistics and estimate their accuracy. To ensure GPS tracks are as precise as possible, we project them onto a digital road network using a map matching algorithm. We also describe how these algorithms work.

From the computed statistics we assess and compare the ride quality of different drivers. We notice a significant difference in quality between professional and other drivers. We conclude the main differentiator between drivers is ride safety and partition the drivers into three groups.

- Those, that drive safely, i.e. those that predict the traffic correctly and react appropriately,
- those, that drive dangerously, but comfortably, i.e. those driving dangerously, that do not accelerate too hard,
- and those that drive safely and uncomfortably.

Math. Subj. Class. (2010): 62E99, 68R01, 60C05, 60E99, 60J05

Ključne besede: kvaliteta vožnje, GPS, projeciranje na cestno omrežje

Keywords: ride quality, GPS, map matching

1. Uvod

Kvaliteta vožnje voznika/vozila je pomemben podatek tako za voznike, kot za delodajalce, zavarovalnice ipd. Ena izmed glavnih vrednosti tega podatka je ocena tveganja uporabe vozila/voznika, npr. kako se različna vozila obnesejo pod podobnimi pogoji, katerega se torej splača kupiti, po kakšni ceni zavarovati vozilo nekega voznika ipd.

Podatek o kvaliteti vožnje ocenimo z značilkami, ki jih izračunamo iz GPS sledi. Da bi bile sledi čim bolj reprezentativne in končni rezultati bolj natančni, jih popravimo z uporabo metod za projeciranje GPS sledi na cestna omrežja, tj. projekcije GPS sledi na digitalni zemljevid, kot je zemljevid OpenStreetMap [19]. Analiza GPS sledi z vidika kvalitete vožnje pomeni predvsem analizo pospeševanja. Natančneje merimo prekomerne pospeške (in zaviranja) ter iz pospeškov ocenjujemo voznikovo predvidevanje prometa.

Opremljeni s tema značilkama lahko na dovolj velikem vzorcu sklepamo o kompetentnosti voznika. Prav tako pa lahko, če fiksiramo voznika (in čim več ostalih parametrov), sklepamo o vplivu vozila na kvaliteto vožnje. V tem delu se bom osredotočil predvsem na ocenjevanje kvalitete vožnje voznika.

Analiza poteka tako, da začnemo z GPS sledjo in, kot je opisano v 3 poglavju, sled čim natančneje preslikamo na zemljevid. V 2 poglavju izboljšane sledi uporabimo za analizo kvalitete vožnje voznikov. Na koncu dela iz izračunanega potegnemo zaključke. Glavni rezultat dela je klasifikator kvalitete vožnje voznikov glede na izbrane značilke. Voznike razporedimo v tri skupine.

- Tiste, ki vozijo varno, tj. tiste, ki pravočasno predvidijo kako se promet obnaša in primerno reagirajo,
- tiste, ki vozijo nevarno in udobno, tj. tiste, ki vozijo nevarno, vendar ne pospešujejo prekomerno,
- ter tiste, ki vozijo varno in neudobno.

2. Ocenjevanje kvalitete vožnje voznikov

Jedro vsake ocene/analize so kakovostni podatki. Želimo si imeti veliko čimbolj natančnih in podrobnih podatkov, saj lahko pomanjkanje podatkov ocenjevanje ciljne značilke oteži oziroma povsem onemogoči. V nadaljevanju bomo s praktičnimi primeri opisali kakšne podatke potrebujemo in poskušali orisati, kaj si večina ljudi predstavlja kot kvalitetno vožnjo. Slednjo bomo opisali z izbranimi značilkami.

Zemlja je podobna sferi, zato se za predstavitev lokacij na njej velikokrat uporablja sferični koordinatni sistem z geografsko širino in višino. Vsaka točka na sferi ima lokacijo (koordinate). *GPS* (ang. Global Positioning System) je sistem satelitov, ki omogoča določanje pozicije GPS merilne naprave na Zemlji. *GPS sled* je zaporedje meritev pozicij (koordinat). Lahko je opremljena s časi merjenja in drugimi podatki. Poglejmo si, kaj lahko izračunamo v izbranih robnih primerih, da bomo ugotovili kaj potrebujemo za izračun značilk.

Kakovostni podatki so podatki, ki vsebujejo vse parametre, ki jih nujno potrebujemo in čim več takih, ki nam pri ocenjevanju pomagajo. Zelo pomembno je tudi, da podatki vsebujejo čimmanj napak, saj lahko sicer (predvsem pri majhnih podatkovjih) pride do velikih napak ocen.

Primer 2.1. Denimo, da imamo na voljo GPS sledi, ki vsebujejo samo zaporedja točk. Ker ne vemo, koliko časa je preteklo med zaporednima točkama, ne moremo

oceniti hitrosti, pospeškov, časa trajanja, povprečne hitrosti, ipd. O kvaliteti vožnje tako ne moremo povedati ničesar. ♢

Če imamo premalo podatkov, ne moremo sklepati o kvaliteti vožnje in voznika.

Primer 2.2. Denimo, da imamo samo eno GPS sled. Ker nimamo podatkov o drugih GPS sledeh, ne vemo kakšni so pričakovani pospeški in pričakovane hitrosti. Iz izračunanih značilk torej ne moremo sklepati o kvaliteti vožnje glede na nek osnovni standard, saj značilk (kvalitete) vožnje nimamo s čem primerjati. ♢

Kvaliteta voznika je povprečje kvalitet voženj, ki jih voznik običajno opravlja.

Primer 2.3. Denimo, da imamo samo eno GPS sled izbrane osebe. Ker imamo veliko različnih podatkov, lahko razberemo, kakšne so povprečne hitrosti in povprečni pospeški, ter znamo razbrati kvaliteto izbrane vožnje iz naših značilk. Kvaliteta ene same vožnje ne predstavlja nujno kvalitete voznika. Lahko da so bile na cesti nepredvidene okoliščine zaradi katerih je dobil slabšo oceno, ali pa je voznik imel slab dan. Opazovana vožnja je bila nekvalitetna, čeprav je voznik lahko kvaliteten. ♢

Kako oceniti kvaliteto posamezne sledi bomo opisali v naslednjem podpoglavju 2.1. Vožnje se lahko razlikujejo po parametrih, kot so prevožena pot, različen promet, vreme, ura, ipd., ki vplivajo na kvaliteto vožnje. Zunanje faktorje je iz GPS sledi zelo težko oceniti in lahko niso neodvisni od ocene; ne samo teoretično, ampak tudi v praksi.

Primer 2.4. Izbrana oseba se vsak dan vozi po isti poti, ob isti uri. Za tako osebo je verjetnost, da je promet povsem naključen napačna predpostavka, saj je promet ob istih urah pogosto podoben. Je pa naključna porazdelitev prometa najboljše, do česar lahko pridemo brez konkretnejših podatkov. Za voznika namreč lahko preverimo, kdaj in kje vozi, vendar brez podatkov ne vemo, kakšen je promet. Ter ali (in kako) promet vpliva na kvaliteto vožnje.

Ko o podatkih ne vemo nič, sta značilki (če vzamemo samo dve), ki jih najbolj opisujeta, značilki povprečja in deviacije. Najbolje je torej v vsakem trenutku predpostaviti povprečen promet (povprečen glede na vse sledi v naših podatkih) z deviacijo vzorca. Natančna porazdelitev ob tej predpostavki ni pomembna, saj je pri vseh enaka. Podobno velja za ostale zunanje faktorje. ♢

Da minimiziramo zgoraj opisane napake je ključno, da zberemo čim več kakovostnih podatkov ter jih ustrezno obdelamo, da lahko iz njih izračunamo uporabne značilke.

Primer 2.5. GPS podatke je potrebno pretvoriti s primerno projekcijo (na zemeljsko oblo), kjer je mogoče preprosto in natančno izračunati razdaljo. V "Googlovih" koordinatah EPSG:4326, ki se uporabljajo za prikaz GPS točk na večini spletnih zemljevidov, računanje razdalje namreč ni enostavno. Več o projekcijah si lahko preberete na spletni strani standarda EPSG [15]. ♢

Šele ko smo podatke zbrali in obdelali, se lahko lotimo ocenjevanja značilk. Seveda pa si moramo značilke najprej izbrati.

2.1. **Izbira značilk.** Da se lahko vprašamo kakšne podatke potrebujemo je potrebno vedeti, kaj želimo oceniti. Mi želimo oceniti "kvaliteto" vožnje, ki je ni lahko strogo formalno definirati.

Vožnja je bolj *udobna*, če ni prevelikih pospeševanj in zaviranj. Zato bom kot mero za (ne)udobnost uporabil značilko, ki prešteje kolikokrat in za koliko je pospeševanje preseglo izbrano mejo 35000 $km/h^2 \approx 2.7\ m/s^2$ oziroma $-40000\ km/h^2$. Meje sem določil glede na razporeditev izmerjenih pospeškov in pospeškov opisanih v produktih namenjen sledenju vozilom, kot so Verizon Telematics [17]. Več o mejah povemo v razdelku 2.4.

Vožnja je bolj *varna*, če je voznik pravočasno predvidel kako se bo promet obnašal in temu primerno reagiral. Na primer, če se je pred nami zgodila prometna nesreča, upočasnimo, da ne bo potrebno kasneje preveč zavirati. Tu (ne)predvidevanje situacije merimo z številom dogodkov, ko vozilo iz pospeševanja takoj preide v stanje zaviranja, oziroma se kar ustavi. Natančneje sta značilki opisani takole

- Udobnost u meri število prekoračitev izbranega pospeška a_{max} in pojemka a_{min}. Za sled $T = \{(\text{lat}_i, \text{long}_i, t_i, v_i, a_i)\}_{i=1}^{n}$., s pospeški intervalov a_i, je udobnost vožnje enaka
$$u = \sum_{i=1}^{n} (\mathbb{1}_{a \leq a_{min}} + \mathbb{1}_{a \geq a_{max}}).$$
- Varnost v meri število prehodov iz stanja pospeševanja P v stanje zaviranja Z, ali ustavitve S. Za sled T z n stanji odsekov s_i, $i \in \{1, 2, \ldots, n\}$, je varnost vožnje enaka
$$v = \sum_{i=1}^{n-1} \mathbb{1}_{s_i \in P \wedge s_{i+1} \in Z \cup S}.$$

Značilki je smiselno normalizirati, saj niso vse sledi enako dolge. Na daljših poteh se lahko zgodi več dogodkov in značilke je potrebno prilagoditi, da bodo imele enak pomen, ne glede na dolžino sledi. Ampak dolžina ni nujno najboljši normalizator. Na primer v mestu se v enem kilometru v povprečju zgodi veliko več dogodkov, kot na avtocesti. Bolj smiselno je primerjati čas, saj se izognemo takim robnim primerom. Naše značilke zato delimo s časom vožnje in tako efektivno merimo značilko na časovno enoto, na primer število prekoračitev maksimalnega udobnega pospeška na uro. Tudi to ni idealno, saj lahko v praksi vedno pride do izjem, vendar brez podatkov več ne znamo izračunati.

Naj bo t čas od začetka do konca GPS sledi, merjen v urah h. Naši glavni značilki sta definirani z

$$U = \frac{1}{t}u$$

in

$$V = \frac{1}{t}v.$$

Negativen dogodek je dogodek, kjer smo povečali eno od glavnih značilk, pozitiven pa dogodek, kjer jih nismo. Ker nimamo podatkov o razmerah na cestah lahko merimo samo očitno negativne dogodke. Pozitivni dogodki bi namreč lahko bili lažno pozitivni. Negativni pa lahko po naključju nastanejo samo zaradi napake v podatkih. Z meritvijo negativnih dogodkov smo naredili le enostransko napako, zato lahko za voznike, ki so izmerjeno nekvalitetni, z neko veliko verjetnostjo trdimo, da so res nekvalitetni. Obratno pa ne moramo sklepati z enako gotovostjo.

Denimo, da je verjetnost, da je bil dogodek pozitiven, enaka p in verjetnost, da je negativen enaka $1 - p$. Tedaj je seštevek vseh ocen u in v porazdeljen z binomsko

porazdelitvijo z verjetnostjo p in številom dogodkov n. Na žalost pa ne vemo koliko dogodkov se je zgodilo in kakšna je verjetnost p.

Predpostavimo, da so značilke $O_1, O_2, \ldots, O_n$ nekega voznika med seboj neodvisne, imajo enako porazdelitev, povprečje μ ter končno varianco σ. Naj bo $S_n = \frac{1}{n}\sum_{i=1}^{n} O_i$. Tedaj po centralnem limitnem izreku [11] značilke v porazdelitvi konvergirajo proti normalni porazdelitvi, ko povečujemo n

$$\sqrt{n}(S_n - \mu) \to N(0, \sigma^2).$$

Značilko za voznika izračunamo kot povprečje enakih značilk vseh njegovih voženj. Daljše vožnje razbijemo na intervale dolge 10 minut. To je dovolj časa, da se v veliki večini primerov kraj vožnje povsem spremeni in lahko trdimo, da sta dela vožnje neodvisna. Če imamo za voznika vsaj 30 voženj (5 ur je zagotovo dovolj), potem za izračun intervala zaupanja (za značilko) uporabimo normalno porazdelitev. Za voznike z manj voženj pa je za izračun intervala zaupanja bolje uporabiti Studentovo porazdelitev. Ker so v našem vzorcu vozniki z manj kot 10 vzorci dosegali zelo čudne rezultate in je bil skupen čas njihovih voženj kratek, smo jih izločili iz analize. Za vse ostale pa je interval zaupanja za vsoto n značilk z izračunanim povprečjem μ in izračunano varianco σ enak

$$\mu \pm z\frac{\sigma}{\sqrt{n}}$$

kjer je z enak $1 - \frac{\alpha}{2}$ kvantilu standardne normalne porazdelitve. Povprečje vzorca značilk $\{Z_i\}_{i=1}^{n}$ izračunamo z $\mu = \frac{1}{n}\sum_{i=1}^{n} Z_i$ in standardno deviacijo vzorca z $\sigma = \frac{1}{n-1}\sum_{i=1}^{n}(Z_i - \mu)$.

Primer 2.6. Naj ima voznik značilke s ocenjenim povprečjem $\mu = 10$, ocenjeno deviacijo $\sigma = 2$ in število vzorcev $n = 30$. Izračunamo, da je 95% interval zaupanja, tj. $\alpha = 0.05$ in $z = 1.96$, enak 10 ± 0.7. Za $\mu = 0.5$, $\sigma = 0.1$ in $n = 30$ pa je interval zaupanja enak 0.5 ± 0.18. ♢

Ker nimamo podatkov o izmerjeni kvaliteti vožnje, se domneve, da so naše izbrane značilke dobre ne da preveriti (lahko bi pa ugotovili, da so slabe). Vidimo lahko le, kako so razporejene in iz slik v nadaljevanju razveremo, da nekateri vozniki res čudno pospešujejo in zavirajo. Prav bi nam prišli kakovostnejši podatki.

2.2. **Pridobivanje podatkov.** Minimalni podatek, ki ga potrebujemo za ocenjevanje kvalitete vožnje je GPS sled s časi. Kaj se zgodi brez časov smo omenili v primeru 2.1. Podatke sem črpal iz večih virov. GPS podatke sem pridobil od

- podjetja Goopti d.o.o. [16] (Goopti), kjer sem dobil dostop do GPS sledi poklicnih voznikov kombijev, in
- arhiva OpenStreetMap (OSM) [19], kjer so shranjene GPS sledi pešcev, kolesarjev, avtomobilov in ostalih vozil.

Zemljevid OSM sem uporabil tudi kot referenčni zemljevid za preslikavo na zemljevid, ki je podrobneje opisana v poglavju 3, saj sem uporabil v knjižnico Open Source Routing Machine (OSRM) [18] vgrajen algoritem za preslikavo na zemljevid. Ta temelji na opisanem algoritmu z markovskimi verigami 3.5. Več o tem, zakaj je preslikava na zemljevid potrebna, je opisano v naslednjem podpoglavju 2.3.

Oboji podatki so vsebovali le GPS sledi s časom. Podatki podjetja Goopti niso javno dostopni, podatki arhiva OSM pa so dostopni na spletni strani [20]. GPS sledi uporabnikov iz celotnega sveta je mogoče z uporabo OSM orodij prenesti v

ZIP datoteki. Mogoče je prenesti tudi sledi iz izbranega geografskega območja. Glede na to, da smo želeli imeti čim več podatkov, se geografsko nisem omejil.

GPS sledi iz arhiva OSM so bile vzorčene na $1s$, podatki podjetja Goopti pa na $10s$. Sledi vzorčene na $1s$ so bile zelo podvržene napakam pri meritvi GPS in so dosegale previsoke vrednosti značilk. Zato (in za bolj enakomerno primerjavo) sem jih ponovno vzorčil na $10s$. V splošnem bi to lahko vplivalo na izračun razdalje (predvsem na križiščih), vendar na zbranih podatkih nisem zaznal velikih sprememb.

2.2.1. *Podatki iz OSM.* Podatki iz arhiva OSM so shranjeni v ogromni ZIP datoteki, ki vsebuje XML zapis sledi z metapodatki. Začetek datoteke, kjer so opisani metapodatki sledi lahko vidimo spodaj.

```
<?xml version="1.0" encoding="UTF-8" ?>
    <gpxFiles version="1.0" generator="OpenStreetMap gpx_dump.py"
    ↪ timestamp="2013-04-09T09:14:45Z">
    <gpxFile id="422" timestamp="2005-08-12T19:19:48Z"
    ↪ points="447173" lat="59.9445080" lon="10.8209930"
    ↪ visibility="public" uid="24" user="Petter Reinholdtsen"
    ↪ filename="public/000/000/000000422.gpx">
        <description>by  in </description>
    </gpxFile>
    <gpxFile id="423" timestamp="2005-08-12T20:55:10Z"
    ↪ points="635073" lat="59.9254420" lon="10.7705530"
    ↪ visibility="public" uid="24" user="Petter Reinholdtsen"
    ↪ filename="public/000/000/000000423.gpx">
        <description>by  in </description>
    </gpxFile>
```

GPS sled znotraj te datoteke pa se začne takole.

```
<gpxFile id="832356" timestamp="2010-10-03T03:46:40.936181Z"
↪ points="5619" lat="45.1031000" lon="34.9655000"
↪ visibility="identifiable" uid="152638" user="gpslib-ru"
↪ filename="identifiable/000/832/000832356.gpx">
    <tags>
        <tag>auto</tag>
             <trkpt lat="45.1142780" lon="4.8901420">
                 <time>2011-09-22T15:11:08Z</time>
             </trkpt>
             <trkpt lat="45.1140670" lon="4.8898850">
                 <time>2011-09-22T15:11:09Z</time>
             </trkpt>
             <trkpt lat="45.1138570" lon="4.8896320">
                 <time>2011-09-22T15:11:10Z</time>
             </trkpt>
             <trkpt lat="45.1136470" lon="4.8893830">
                 <time>2011-09-22T15:11:11Z</time>
            </trkpt>
```

GPS sledi smo zaradi njihove velikosti shranili v skrčenem formatu, ki je primeren za prostorske podatkovne baze. Ena takih prostorskih podatkovnih baz je postgreSQL [23], ki tak format podpira preko vtičnika postGIS [22]. Delo opravi koda, ki je dostopna na [1].

V podatkovno bazo shranimo voznika *gpx_id*, številko vožnje *segment_id*, datum *track_date* in GPS sledi v postGIS formatu *track*. Ko vnašamo Gooptijeve podatke si zapomnimo tudi, da so bili vnešeni iz Gooptijeve podatkovne baze. Ko za sled izračunamo značilke, jih zapišemo v bazo za kasnejšo analizo.

```
CREATE TABLE public.gpx_data
(
    gpx_id integer NOT NULL,
    segment_id integer NOT NULL,
    track_date date,
    track geometry NOT NULL,
    hard_acceleration integer,
    hard_acceleration_severity double precision,
    hard_deceleration double precision,
    hard_deceleration_severity double precision,
    accelerate_stopped integer,
    accelerate_breaking integer,
    distance double precision,
    is_car boolean,
    is_goopti boolean,
    "time" double precision,
    CONSTRAINT gpx_data_pkey PRIMARY KEY (gpx_id, segment_id),
)
```

2.2.2. *Podatki podjetja Goopti.* Anonimizirani podatki so bili razdeljeni na tri dele. V naročilih je bilo opisano, kdaj in kam bo šlo določeno vozilo. V ločeni datoteki je bilo opisano kdo je v določenem trenutku vozil vozilo, na oddaljenem strežniku pa so bile shranjene neprekinjene GPS sledi vozila. To je pomenilo, da sem moral najprej identificirati kdo je vozil vozilo in kdaj ga je vozil, nato pa sem z strežnika prenesel ustrezno GPS sled in jo shranil v datoteko. Kasneje sem tudi te podatke vnesel v zgoraj opisano podatkovno bazo.

2.2.3. *Kakovostnejši podatki.* Za analizo bi bilo zelo koristno, če bi poleg koordinat in časa na vsaki točki imeli zapisano tudi naslednje.

- Hitrost vozila. S tem odstranimo napako, ki zaradi napake GPS pri določanju koordinat (o teh napakah bomo več povedali v poglavju 3) nastane pri računanju hitrosti. Hitrost izračunamo tako, da med dvema zaporednima točkama izmerimo razdaljo in jo delimo s porabljenim časom. Izračunali smo torej povprečno hitrost na intervalu.
- Pospešek vozila. Kot pri hitrosti se izognemo napaki GPS meritve. Izračunan pospešek je povprečen pospešek na intervalu
- G-silo. To je mera, ki pove kakšno težo občuti potnik. Na primer g-sila $2g$ pomeni, da potnik občuti silo, ki je dvakrat večja od sile težnosti. Predvsem jo občutimo pri zavijanju, kjer se pojavijo stranski pospeški. Ker GPS sledi niso dovolj natančne, je g-sila iz GPS sledi praktično neizračunljiva.

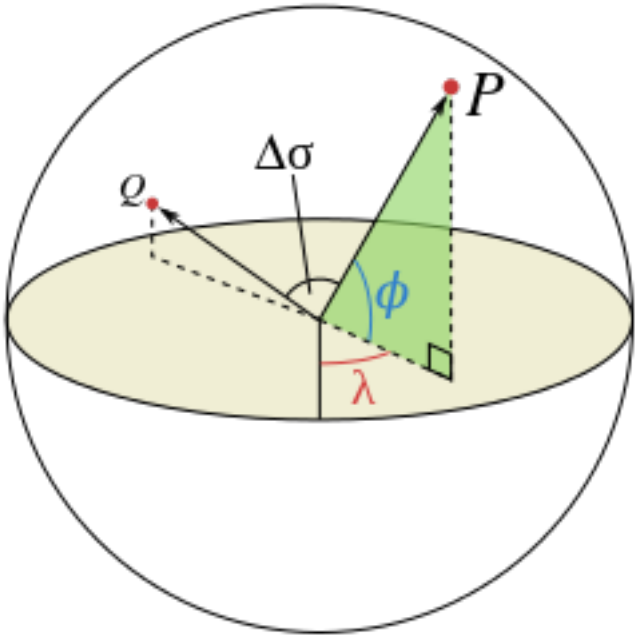
- Koliko je pritisnjeno pedalo za plin, zavoro, ipd., kar je iz GPS sledi praktično nemogoče izračunati.
- Lokacijo glede na ostala vozila.
- Koliko časa je voznik potreboval za odziv na situacijo (npr. zavirajoče vozilo, rdeča luč, prometna nesreča); to se da izračunati iz zgornjih dveh podatkov.

Opremljeni s takimi in podobnimi podatki bi lahko kvaliteto vožnje zelo dobro ocenili. Obstoj podatkov je realističen, saj moderni pametni telefoni že zbirajo podatke o pospeških in g-silah, avtomobili pa podatke o hitrosti vozila ter stanju pedal. Moderni avtomobili imajo tudi sisteme laserjev in kamer, s katerimi zaznavajo vozišče, okolico in ostale udeležence v prometu (ne samo vozila, ampak tudi pešce, kolesarje, stavbe, ipd.).

2.3. **Obdelava podatkov.** Razdaljo med zaporednima točkama izračunamo kot razdaljo na sferi z Zemljinim polmerom. Naj bosta točki predstavljeni s koordinatama zemljepisne širine in dolžine ϕ_1, λ_1 in ϕ_2, λ_2 zapisanimi v radianih. Naj bo r radij Zemlje in razlika zemljepisnih širin $\Delta\phi = |\phi_2 - \phi_1|$. Tedaj je sprememba centralnega kota enaka

$$\Delta\sigma = \arccos\big(\sin\phi_1 \sin\phi_2 + \cos\phi_1 \cos\phi_2 \cos(\Delta\lambda)\big),$$

razdalja pa je enaka $d = r\,\Delta\sigma$ (za $\Delta\sigma$ v radianih). Glede na to, da so naše GPS točke zajete vsakih nekaj sekund, so točke zelo blizu ena drugi (relativno glede na polmer Zemlje) in bo napaka zaradi privzetega (da je zemlja sfera) zanemarljiva.



Slika 1. Vizualizacija kotov za računanje razdalje na sferi. Vir [24].

Hitrost na odseku med dvema točkama izračunamo tako, da razdaljo med dvema zaporednima izmerjenima paroma koordinat d, delimo z časom t, ki je pretekel vmes. S tem smo izračunali povprečno hitrost $v = d/t$ na odseku. Pospešek na odseku med dvema točkama pa izračunamo tako, da razliko med trenutno in prejšnjo hitrostjo

Δv delimo s časom t. S tem smo izračunali povprečen pospešek $a = \Delta v/t$ na odseku. Če bi imeli vse podatke opisane v prejšnjem podpoglavju 2.2.3, bi bila preslikava GPS sledi na zemljevid nepotrebna, saj razdalja ne vpliva (neposredno) na kvaliteto vožnje.

Napaka pri računanju celotne razdalje v povprečju ni velika, zato je izračun celotne razdalje iz GPS sledi dovolj natančna značilka. Če pa moramo iz izračunane razdalje in časa na vsakem odseku GPS sledi oceniti hitrost in pospešek vozila, preslikava na cesto naenkrat postane izjemno pomembna, saj lahko majhne lokalne napake privedejo do nerazumnih pospeškov in hitrosti.

Primer 2.7. Napaka merjenja koordinat z GPS je v povprečju sicer relativno majhna, vendar je lahko na posameznem kratkem odseku zelo velika (več 100% glede na dolžino odseka), kar privede do lažnih pozitivnih in občasno tudi lažno negativnih dogodkov, ki so potem narobe uporabljeni pri izračunu značilke kvalitete vožnje. To bi pomenilo, da so naše značilke zelo nezanesljive. ♢

Opomba 2.8. Lažnih negativnih dogodkov bi bilo manj, saj bi morali vozniki najprej voziti nekvalitetno, nato pa bi napaka merjenja koordinat z GPS nekvalitetnost morala odpraviti. Napaka ima sicer enako verjetnost, da zadevo poslabša. Če voznik ni vozil slabo pa vsaka večja napaka pri merjenju koordinat z GPS privede do lažnega pozitivnega dogodka. Ker so opisane napake GPS signala praktično neodvisne od vožnje in v praksi porazdeljene normalno (glejte poglavje 3), lahko sklepamo, da je lažnih negativnih primerov manj.

Da se izognemo najbolj očitnim napakam GPS meritev, ki se dogajajo v mestih zaradi odbojev od stavb, GPS sledi najprej preslikamo na zemljevid. Več o razlogih in preslikavah samih v poglavju 3. Za preslikavo na zemljevid uporabimo algoritem iz knjižnice OSRM za preslikavo na zemljevid [18].

Preslikava GPS sledi na zemljevid že zelo dobro odpravi napake, ki nastanejo zaradi stranskega premika GPS sledi (pravokotno na cesto). Napak v smeri ceste pa ne odpravi nujno. Ali se jih sploh trudi odpraviti, je odvisno od algoritma za preslikavo, ki ga uporabimo. Naš izbran algoritem se ne trudi odpraviti teh napak, zato moramo biti na njih pozorni.

Nadaljnja obdelava podatkov je potekala med računanjem značilk, kjer smo izločili morebitna prekomerna odstopanja v hitrosti in pospeških, ki so nastala zaradi napake GPS signala.

2.4. **Računanje značilk.** Preden se lotimo računanja značilk moramo povedati, kako bomo računali razdaljo med dvema GPS točkama. Kot smo omenili v primeru 2.5, bomo namesto EPSG:3857 (“Googlove koordinate”) uporabili EPSG:4326, tj. “klasične” koordinate zemljepisne širine ter dolžine. Za izračun razdalje bomo privzeli, da je zemlja krogla in na podlagi tega izračunali razdaljo med točkama. Za algoritme uporabljam (psevdo)kodo, ki sledi sintaksi programskega jezika Python [21].

```
def dist(tocka1, tocka2):
    ## razdalja na sferi z Zemljinim polmerom med podanima točkama

    R = 6371

    ## kote pretvorimo v radiane, za lažje računanje
```

```
    phi1 = radians(tocka1.latitude)
    phi2 = radians(tocka2.latitude)

    ## razlika v zemljepisni širini in širini
    dphi = radians(tocka2.latitude - tocka1.latitude)
    dlambda = radians(tocka2.longitude - tocka1.longitude)

    ## izračun razdalje na sferi. Zaradi numerične (ne)stabilnosti
    ↪ uporabljamo drugo formulo, kot zgoraj v opisu
    harvesine = (sin(dphi / 2) * sin(dphi / 2) +
                 cos(phi1) * cos(phi2) * sin(dlambda / 2) *
                 ↪ sin(dlambda / 2))
    c = 2 * atan2(math.sqrt(harvesine), math.sqrt(1-harvesine))
    return R * c
```

Da bodo enote količin enake skozi cel program, merimo razdaljo v kilometrih km, čas v urah h, hitrost v kilometrih na uro km/h ter pospešek v kilometrih na kvadratno uro km/h^2. Kot smo povedali v podpoglavju 2.1 je cilj, da naši značilki za kvaliteto vožnje merita kdaj vozniki premočno pospešujejo in kdaj slabo predvidijo potek prometa in začnejo nenadno zavirati. Za izračun obeh značilk moramo vedeti kakšen je pospešek vozila v vsakem trenutku. Iz zbranih podatkov lahko izračunamo le povprečen pospešek med zaporednima GPS točkama, idealno pa bi bilo pospešek meriti bolj pogosto. V spodnji funkciji za podano GPS sled izračunamo osnovne značilke, kot so razdalja, hitrost in pospešek.

```
def izracunajStatistike(tocke):
    ## za vsak del sledi izračuna osnovne značilke, kot so razdalja,
    ↪ hitrost in pospešek
    statistike = []
    maxHitrost = 0

    ## za vse naslednje intervale izračunamo hitrost in pospešek
    for i in range(len(tocke)):
        point = tocke[i]
        time = tocke[i].time

        timeDelta = timeDiff(prejsnji cas, cas)
        radzalja = dist(prejsnja tocka, tocka)
        hitrost = radzalja / timeDelta
        maxHitrost = max(maxspeed, speed)
        pospesek = (hitrost - prejsnja hitrost) / timeDelta

        statistike.append({hitrost, cas, pospesek, razdalja, prejsnji
        ↪ cas})
    return statistike, maxHitrost
```

Poleg pospeška, razdalje in časa izračunamo tudi maksimalno hitrost. S pomočjo maksimalne in povprečne hitrosti ter pospeškov lahko z veliko gotovostjo trdimo, da je vozilo, ki ga opazujemo, motorno vozilo. To je pomemben podatek, saj za GPS sledi iz baze OSM ne vemo s kakšnim vozilom so bile narejene.

Na tej točki imamo za izračun pojavitev prekomernih pospeškov, torej značilke udobja U, že dovolj podatkov. Za izračun značilke varnosti V pa bomo sled morali še analizirati. Zaznati želimo dogodke, kjer voznik iz pospeševanja preide v zaviranje oziroma se ustavi. Taki dogodki predstavljajo potencialno nepazljivost voznika, zaradi katere je moral naglo reagirati in bistveno spremeniti smer pospeška v kratkem času.

Definicija 2.9. Časovni interval med dvema zaporednima GPS točkama razporedimo med štiri različne *tipe intervala*. Ti so pospeševanje P, zaviranje Z, ohranjanje hitrosti O in ustavitev S.

Ker nimamo podatka o tem kdaj voznik pritiska na pedala za plin oziroma zavoro, moramo tip intervala zaznati iz spremembe pospeškov, ki smo jih izračunali iz GPS sledi. Pri tem zaradi napak GPS uporabimo toleranco v pospešku; izračunan pospešek na primer praktično ni nikoli 0. Zato izračunanih pospeškov, ki so absolutno manjši od 2500 $km/h^2 \approx 0.69\ km/h/s \approx 0.19\ m/s^2$ nismo obravnavali kot spremembo pospeška. Vozila, ki se premikajo počasneje od 5 km/h razglasimo za stoječa vozila, saj izračunana hitrost ni praktično nikoli enaka 0.

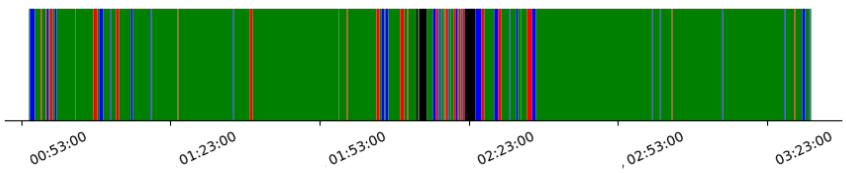
Vrednosti za pospeške smo določili iz industrijskih standardov, kot so [17] in iz opazovanja porazdelitve podatkov. Mejo za prekomerno pospeševanje smo postavili na 35000 $km/h^2 \approx 9.7\ km/h/s \approx 2.7\ m/s^2$ in za prekomerno zaviranje na $-40000\ km/h^2$. Mejo za zaviranje smo absolutno postavili višje, saj vozila ponavadi zavirajo hitreje, kot pospešujejo. Izračunamo tudi kako hudo je voznik prekoračil postavljeno mejo. Podatkovne točke, kjer je bil pospešek nenormalno visok, to je višji od vrednosti 150000 $km/h^2 \approx 41.67\ km/h/s \approx 11.57\ m/s^2$, smo izločili, saj je zelo verjetno prišlo do napake v GPS signalu. Opisano poleg analize prekomernih pospeškov izvedemo v spodnji kodi.

```
def tipizirajSled(statistike):
    ## intervale označimo z tipom; izračunamo značilki U in V
    tipiziranaSled = []

    ## izbrane meje
    pospesekStart = 2500 ## v km / h^2
    pojemekStart = - 2500
    najmanjsaHitrost = 5 ## v km / h

    mocnoZaviranje = - 40000 ## v km / h^2
    mocnoPospesevanje = 35000
    nenormalnoPospesevanje = 150000

    nenormalnih = 0 ## število pospeškov, ki zelo odstopajo
    for interval in statistike:
        if interval['hitrost'] < najmanjsaHitrost:
            interval['tip'] =  'ustavitev'
        elif interval['acc'] > accelerationStart:
            interval['tip'] =  'pospeševanje'
        elif interval['acc'] < decelerationStart:
            interval['tip'] =  'zaviranje'
        else:
            interval['tip'] =  'ohranjanje hitrosti'
```

```
        if abs(interval['pospesek']) > nenormalnoPospesevanje:
            nenormalnih += 1
            continue
        if interval['pospesek'] > hardAcc:
            interval['mocnoPospesevanje'] = 1
        else:
            interval['mocnoPospesevanje'] = 0
        if statPoint['pospesek'] < hardBreak:
            interval['mocnoZaviranje'] = 1
        else:
            interval['mocnoZaviranje'] = 0
        tipiziranaSled.append(interval)
    return tipiziranaSled, nenormalnih
```

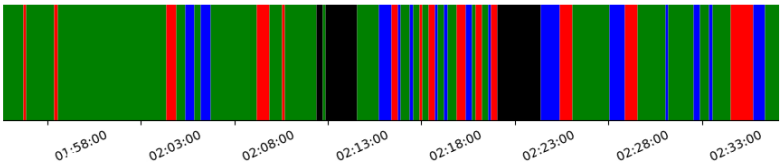
Značilke za pospeške smo že izračunali ter pri tem nismo zaznali veliko prekomerno odstopajočih pospeškov, kar pomeni, da je preslikava na zemljevid odpravila večino večjih napak. Za predvidevanje prometa pa gremo še en korak dlje. Tipe zaporednih intervalov združimo, da lažje vidimo, vizualiziramo in izračunamo kdaj se tip spremeni. Sedaj lahko preštejemo kolikokrat je voznik prešel iz tipa pospeševanja v tip zaviranje oziroma v tip ustavitve. Moramo pa tudi tu vpeljati minimalne tolerance, da zaradi napake GPS sledi ne zajamemo robnih primerov.

Primer 2.10. Najpreprostejši lažno negativen primer pospeševanja bi opazili pri počasnem premikanju skozi mesto ali kolono, saj tam pogosto pride do pospeševanja, ki mu takoj sledi ustavljanje. Temu se izognemo tako, da določimo minimalno razdaljo in čas, ki mora preteči med dvema takima dogodkoma. In res sta potrebna oba, saj je lahko, zaradi napake meritve koordinat z GPS, kljub relativno dolgi razdalji čas kratek (in obratno). Mi smo izbrali minimalno razdaljo 30 metrov in minimalni čas 10 sekund. ♢



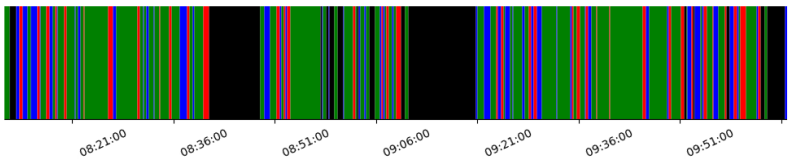
Slika 2. Vizualizacija pospeševanja izbrane GPS sledi (modro), zaviranja (rdeče), ohranjanja hitrosti (zelena) in stanja na mestu (črna).

Če podrobneje pogledamo v del poti na sliki 2, kjer smo zaznali več dogodkov, vidimo dogodke, kjer so vozniki nenadno spremenili smer pospeška (slika 3). Vidimo tudi, da meja hitrosti za ustavljanje ni bila previsoka, saj bi v nasprotnem primeru večkrat za kratek čas videli tip ustavitve.



Slika 3. Vizualizacija pospeševanja, bolj podrobno.

Oglejmo si še en primer vizualizacije vožnje, tokrat v mestu (slika 4). Vidimo, da se tip intervala večkrat spremeni in da se voznik večkrat povsem ustavi. Tu se zgodi več dogodkov, kot na sliki 2, vendar razloga pri izračunu ne moremo zagotovo zaznati.

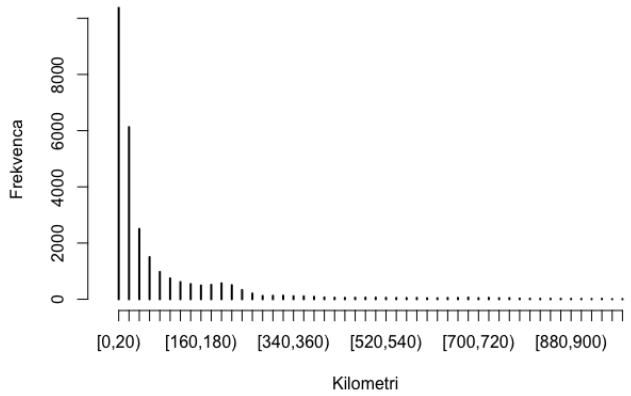


Slika 4. Vizualizacija pospeševanja.

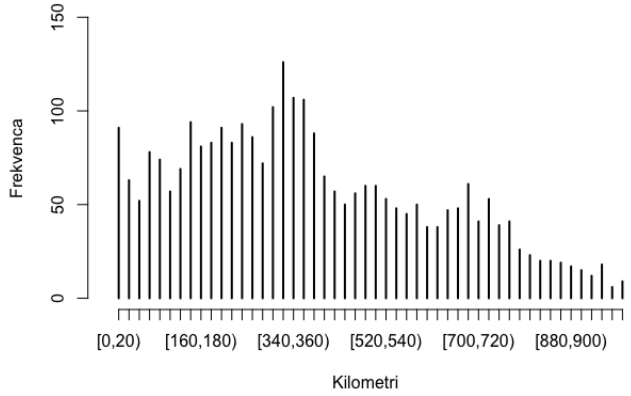
2.5. **Rezultati.** Sedaj, ko smo vse podatke zbrali in iz njih poračunali značilke, si jih lahko natančneje ogledamo, jih analiziramo in vizualiziramo. Imamo podatke tako poklicnih, kot ostalih voznikov. Vozniki teh skupin se razlikujejo v izračunanih značilkah, zato jih je smiselno obravnavati tudi ločeno.

Iz zbirke Goopti podatkov smo zbrali 5200 sledi poklicnih voznikov in iz zbirke podatkov OSM 760000 ostalih sledi. Ker za sledi iz zbirke OSM ne vemo, ali so bile posnete pri vožnji z vozilom, smo tiste, za katere sumimo, da so bile opravljene peš zavrgli. Nekaj več kot polovico sledi smo izločili, ostalo pa nam jih je še 322000, kar ni malo.

Zaradi prostorskih omejitev smo iz arhiva OSM uporabili le šestino vseh sledi. Če smo uporabili sled nekega voznika, potem smo za njega pobrali vse sledi, ki so bile na voljo. Ker več sledi ne koristi neposredno pri natančnosti ocenjevanja kvalitete vožnje enega voznika, bi nam več sledi lahko pomagalo le pri opazovanju trenda in porazdelitve sledi glede na značilke. Za ta namen smo jih zbrali dovolj. Oglejmo si porazdelitve (osnovnih) značilk dolžine in hitrosti (slike 5, 6, 7 in 8).

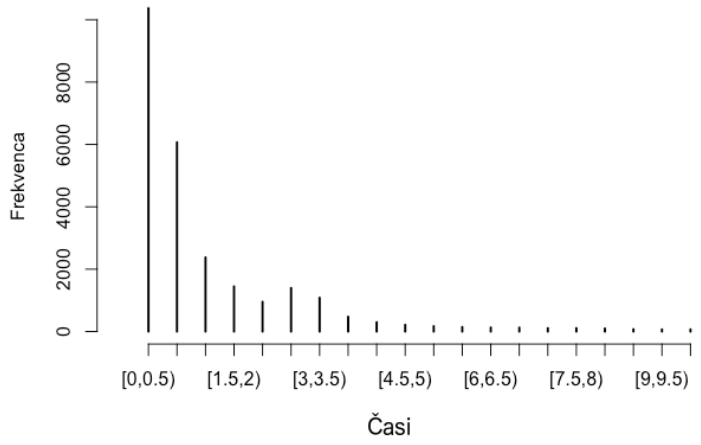


Slika 5. Število vseh sledi, ki po dolžini padejo v izbran interval, prikazan na x osi. Ta je razdeljena na intervale dolžine 20 kilometrov. Opazimo, da je velika večina sledi krajša od 100 kilometrov.

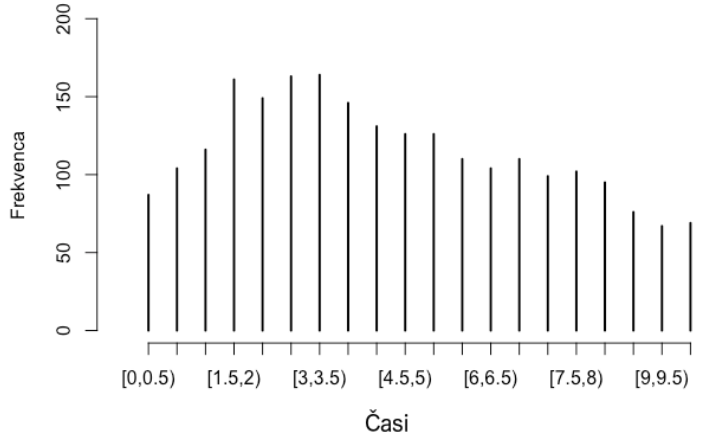


Slika 6. Število sledi poklicnih voznikov, ki po dolžini padejo v izbran interval.

Opazimo, da je povprečna dolžina sledi poklicnih voznikov veliko daljša (glejte sliki 5 in 6), kot povprečna dolžina sledi ostalih voznikov. To je smiselno upoštevati pri razlagi značilk, zato je dobro, da smo jih normalizirali.

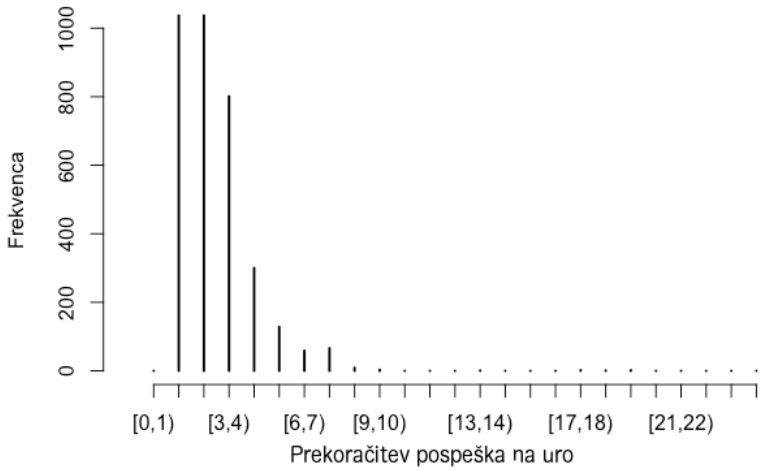


Slika 7. Število sledi, ki po času padejo v izbran interval.

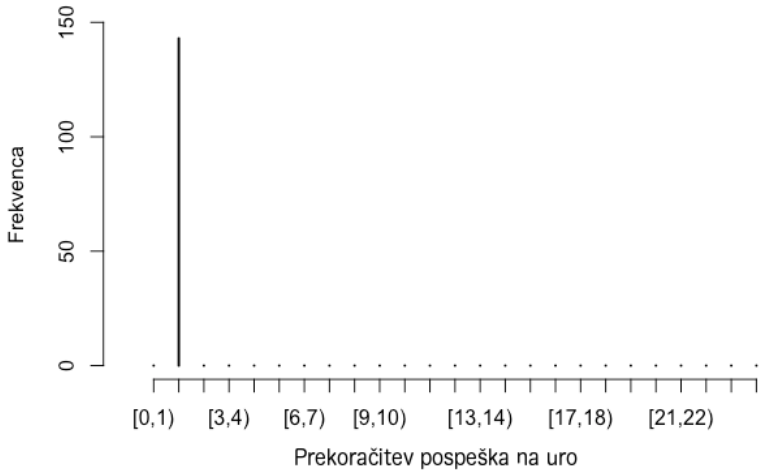


Slika 8. Število sledi poklicnih voznikov, ki po času padejo v izbran interval.

Porazdelitev časov je podobne oblike, kot porazdelitvi razdalj (glejte sliki 7 in 8), kar je smiselno. Je pa porazdelitev časov bolj enakomerna, kot porazdelitev razdalj, kar pomeni, da lahko za nekatere krajše razdalje porabimo veliko časa. To kaže, da obstaja bistvena razlika med normalizacijo s časom in normalizacijo z razdaljo, kot smo predvideli. Oglejmo si kakšne rezultate so dosegali naši vozniki (slike 9, 10, 11 in 12). Ker merimo negativne dogodke, je rezultat boljši, če je značilka bližje 0. Najprej vidimo, kako so se vozniki odrezali pri pospeševanju, torej značilki U (sliki 9 in 10).

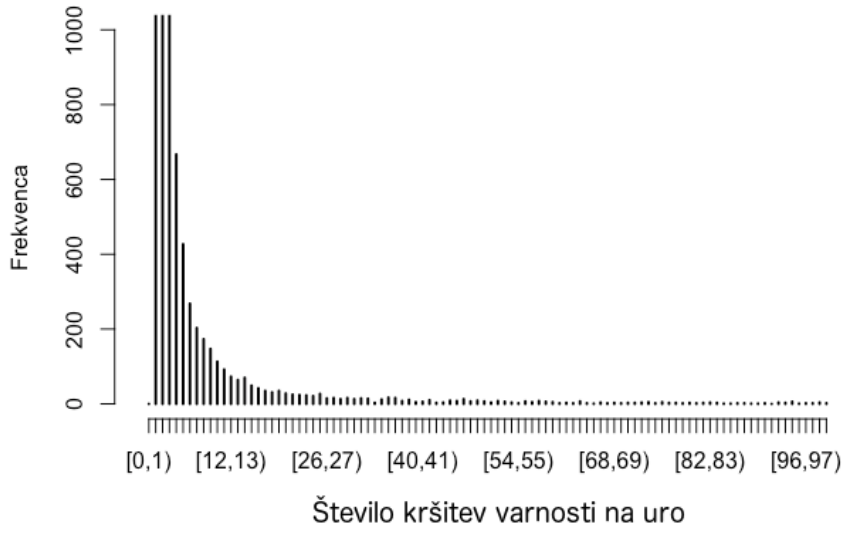


Slika 9. Število sledi, ki po značilki udobja U, padejo v izbran interval.



Slika 10. Število sledi poklicnih voznikov, ki po značilki udobja U, padejo v izbran interval.

In pa kako so se vozniki odrezali pri predvidevanju poteka prometa, torej značilki V (sliki 11 in 12).



Slika 11. Število sledi, ki po značilki varnosti V, padejo v izbran interval.

Slika 12. Število sledi poklicnih voznikov, ki po značilki varnosti V, padejo v izbran interval.

Iz zgornjih grafov lahko razberemo pričakovano - da so poklicni vozniki veliko bolj pozorni in previdni pri vožnji. Oglejmo si razliko še na grafu, kjer bomo z razvrščanjem voznike glede na izračunane značilke razdelili v skupine.

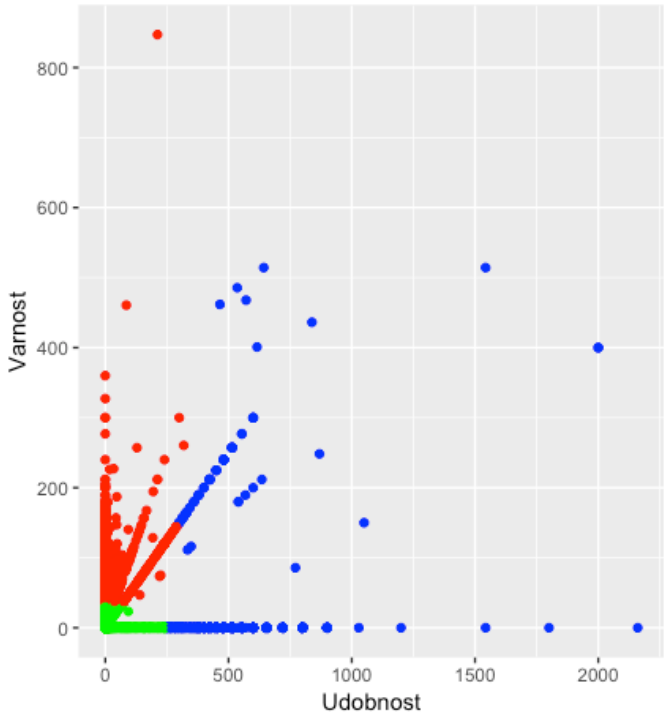
Definicija 2.11. Razvrščanje (ang. clustering) je postopek, pri katerem elemente razporedimo v množice (ang. cluster), tako da je vsak element množice v izbrani meri bližje svoji množici, kot katerekoli drugi množici.

Poznamo veliko različnih algoritmov za razvrščanje. Med najbolj znane spadajo

- hierarhično razvrščanje,
- razvrščanje s sredinskim vektorjem,
- porazdelitveno razvrščanje in
- gostotno razvrščanje.

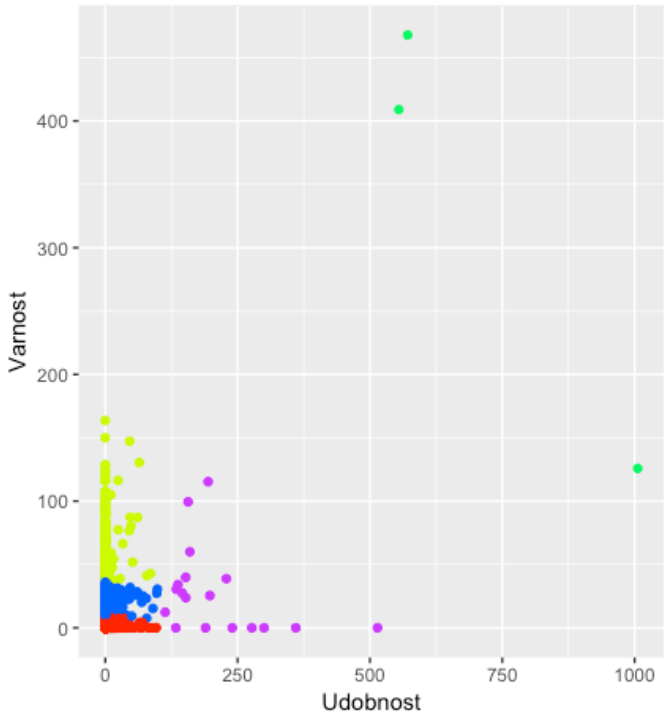
Več o njih lahko preberete v [10]. Mi bomo uporabili razvrščanje s sredinskim vektorjem, z algoritmom $k-$means. Ta za podan k podatke razporedi v k množic. Vsaki množici pripada sredinski vektor, od katerega se v izbrani metriki računa razdalja do vsakega podatkovnega vektorja. Cilj algoritma je minimizirati skupno razdaljo podatkovnih vektorjev do sredinskih točk pripadajočih množic. Ta problem je NP-težak in se za njega uporablja hevristične pristope, ki najdejo (lokalno) optimalno rešitev.

Razvrščali bomo pare varnosti in udobnosti (U, V) in bomo uporabili kar evklidsko razdaljo med točkami. Da bosta udobnost in varnost enakomerno upoštevana pri razvrščanju v množice, bomo izračunane značilke najprej standardizirali. To pomeni, da jim bomo za izračun odšteli povprečje in jih delili z deviacijo. Prikazali pa jih bomo originalnimi vrednostmi. Spomnimo se, da je nižja vrednost značilke boljša.



Slika 13. Razdelitev GPS sledi glede na obe glavni značilki, vse sledi.

Imeti 800 kršitev varnosti na uro in 2000 kršitev udobnosti (glejte sliko 13) je veliko (udobnost upošteva tudi stopnjo kršitve, zato je višja številka primerljiva z številko pri varnosti). Vendar je sledi, ki pretirano odstopajo le majhen delež. Ko pogledamo povprečje značilk za posamezne voznike vidimo že bolj realistične številke (slika 14). Le par voznikov, ki imajo zalo kratke sledi še vedno odstopa. Linearne odvisnosti, ki jih opazimo na sliki 13 verjetno ustrezajo različnim tipov voženj, kot sta vožnja v mestu in vožnja po avtocesti.



Slika 14. Razdelitev voznikov glede na obe glavni značilki, vse sledi.

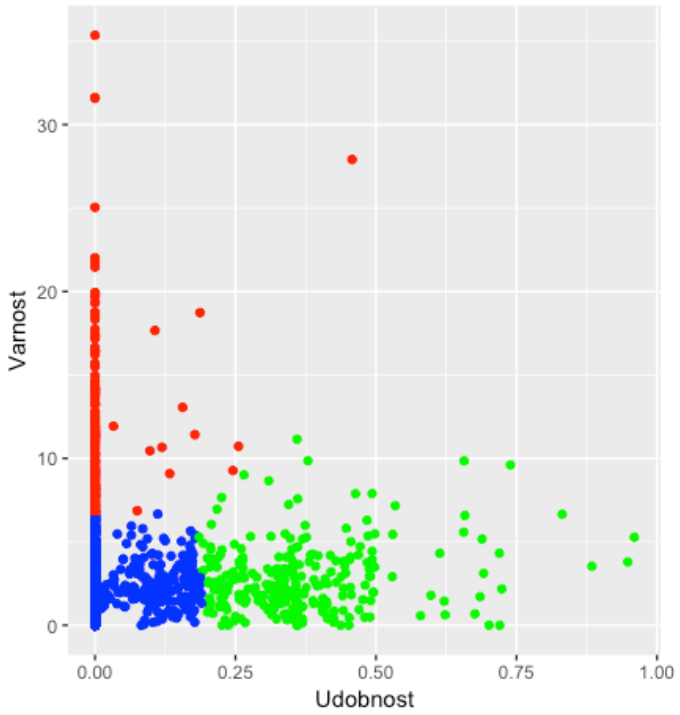
Na sliki 15 z vozniki, ki imajo vsaj 2 uri voženj, vidimo, da dosegajo veliko boljše rezultate, kot preostali vozniki. Povečanje števila ur vožnje rezultatov ne spremeni. Razlika med vozniki z več oziroma manj prevoženimi urami se pojavi predvsem pri udobnosti vožnje, kjer se zaradi kratkega časa vožnje, ki verjetno poteka v mestu, napake voznikov poudarjeno prikažejo v značilki. Da se napake pri večjem času umirijo je pričakovano, saj je interval zaupanja z višjim časom vožnje ozek, kot smo pokazali v 2.1. Razdelitev pa v obeh primerih ostane enake oblike, le z drugimi mejnimi vrednostmi.

Na sliki 15 opazimo, da vozniki redko naredijo obe vrsti napak. To bi lahko bilo zaradi različnih lokacij voženj. Na primer v mestu lahko napravijo več napak pri pospeševanju, na avtocesti pa zaradi nepazljivosti nenadoma začnejo zavirati. Vozniki naredijo veliko manj napak pri udobnosti vožnje, kot pri varnosti. Razlog verjetno leži v tem, da so vsi vozniki relativno izkušeni v primernem pospeševanju, ki preide v podzavest, za varnost pa je potrebno aktivno skrbeti.

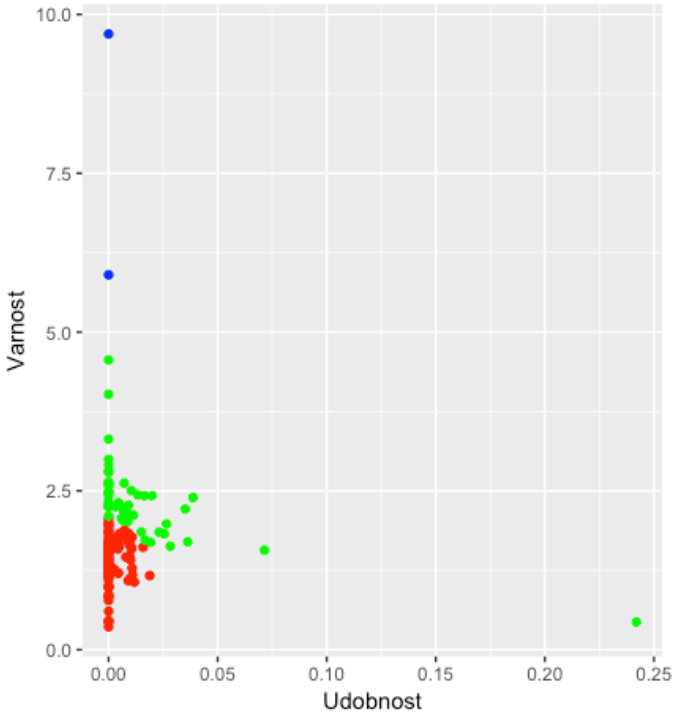
Kot vidimo na sliki 16 imajo poklicni vozniki v primerjavi z ostalimi v povprečju veliko boljše rezultate. Podatke imamo za 26000 voznikov. Nevarno jih vozi kar 3700, neudobno pa 1200. V splošnem voznike razvrstimo v tri skupine (slika 15), pri tem pa upoštevamo le tiste, za katere imamo dovolj sledi.

- Tiste, ki vozijo neudobno (in varno), tj. imajo $U > 0.2$.
- Tiste, ki vozijo nevarno (in udobno), tj. imajo $V > 7$.
- Tiste, ki vozijo varno, tj. imajo $U \leq 0.2$ in $V \leq 7$.

Podrobnejša razdelitev določa predvsem kako varno vozijo vozniki, ne pa kako udobno, iz česar lahko sklepamo, da je udobnost vožnje veliko bolj enakomerno razporejena med vozniki različnih tipov kot varnost in je varnost zato značilka, ki jih najbolj loči.



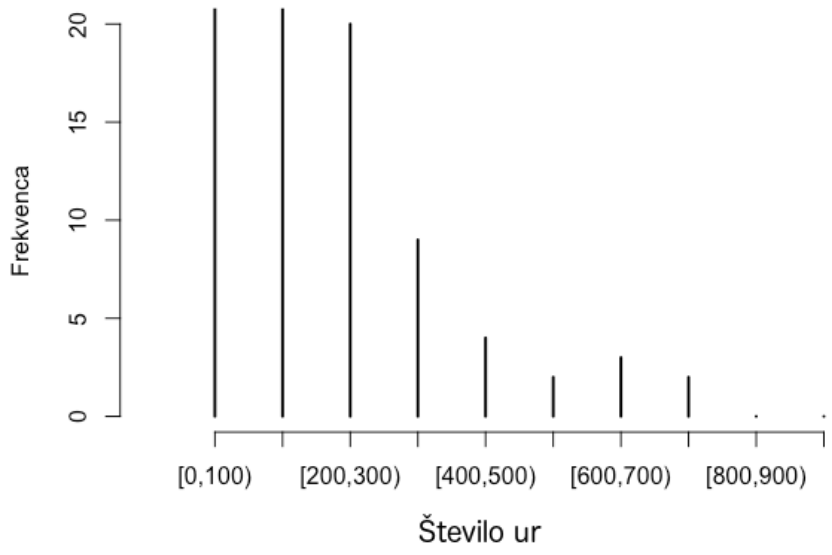
Slika 15. Razdelitev vseh voznikov z vsaj dvema urama vožnje glede na obe glavni značilki.



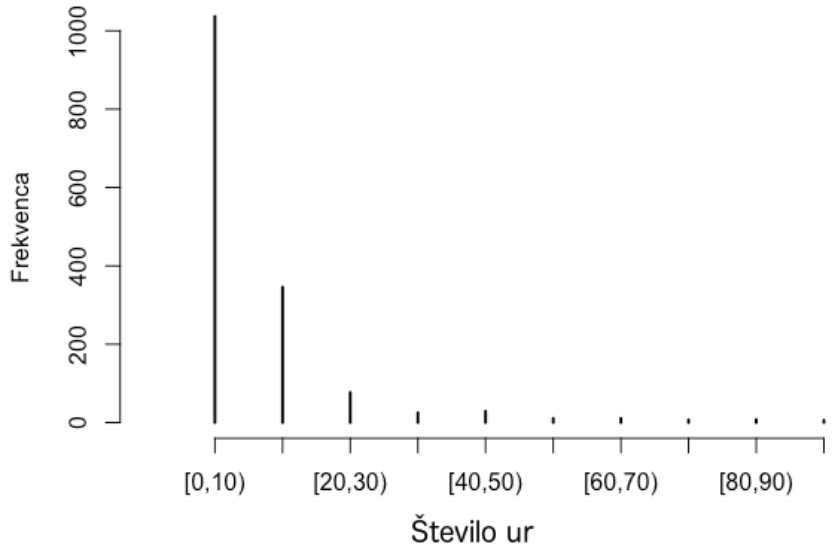
Slika 16. Razdelitev poklicnih voznikov glede na obe glavni značilki.

Sedaj, ko smo analizirali rezultate, pa poglejmo še kako natančni so naši izračuni. Za posameznega voznika imamo največ 800 ur vožnje in najmanj 0.1 ur. Prvih 27

voznikov z največ urami ima več kot 700 ur vožnje. Kako so ure porazdeljene lahko vidimo na slikah 17 in 18.



Slika 17. Število poklicnih voznikov, ki po skupnen času vožnje padejo v izbran interval.



Slika 18. Število voznikov, ki po skupnem času vožnje padejo v izbran interval.

Vse skupaj imamo 26000 voznikov, od tega ima polovica le eno vožnjo, le 9000 več kot 5 voženj in le 5800 več kot 10 voženj. Te vožnje so v povprečju tudi zelo kratke,

zato o voznikih težko natančno sklepamo in jih ne upoštevamo pri definiranju skupin voznikov. Kot smo pokazali v razdelku 2.1, je že 30 voženj oziroma 5 ur vožnje dovolj, da zelo natančno določimo kvaliteto voznika in na podlagi teh kvalitetnih ocen voznike razporedimo v skupine. V skupine lahko razporedimo tudi preostale voznike, za katere izračun ni bil dovolj natančen, saj imajo visoke vrednosti značilk in kljub večji napaki nedvomno spadajo v pripadajoče skupine. Le za kakšnega robnega voznika se lahko zmotimo in ga narobe razvrstimo.

Zgornja ocena velja za vsakega voznika. Ko pa govorimo o oceni vseh voznikov je skupna napaka relativno veliko manjša, saj imamo veliko podatkov. To pomeni, da so naše skupine dobro določene.

3. Projeciranje GPS sledi na cestna omrežja

3.1. **Opis.** Projeciranje na digitalni zemljevid je uporabno za lastnike GPS naprav, saj se sled lahko prikaže na zaslonu in uporabnik lahko spremlja, kje se vozi v živo. Mi pa želimo natančno vedeti, kje je potoval, da lahko izračunamo prevožene kilometre, pospeške ter druge podobne značilke. Za ta namen moramo njegovo GPS sled projecirati na digitalno omrežje cest, ki ga imamo shranjenega v sistemu. Natančneje na zaporedje *cestnih odsekov*, to je delov cest med zaporednima križiščema. Seveda nikoli ne bomo mogli zaporedja določiti s popolnim zaupanjem v rezultat, saj je napaka prisotna tako pri merjenju koordinat z GPS, kot v predstavitvi digitalnega omrežja cest. Med pogoste napake spadajo manjkajoče ceste, parkirišča ter napaka pri projekciji.

Smiselno je razviti algoritem, ki postopno gradi pot, da ga lahko uporabljamo na živih podatkih (ang. on-line), na primer med vožnjo, in ne le za analizo na koncu. Pristop s postopnim grajenjem na živih podatkih ima prednost, da imamo rezultate na voljo že med vožnjo in se lahko vožnjo takoj prilagodimo rezultatom. Ima pa tudi slabost, da lahko pripelje do projekcij na napačne odseke cest.

Primer 3.1. Na podlagi zadnje meritve GPS točke se algoritem odloči, da smo na križišču zavili desno, saj je izmerjena točka padla desno od ceste, po kateri se vozimo. Mi nismo zavili, algoritem pa je naredil napako. Desna cesta je bila najboljši kandidat pri zadnji meritvi in sedaj mora nadaljevati po tej cesti naprej, ker upoštevamo cestna pravila. Problem rečimo tako, da zadnjih nekaj korakov ponovno izračunamo projekcije, ali pa algoritem ponastavimo. ♢

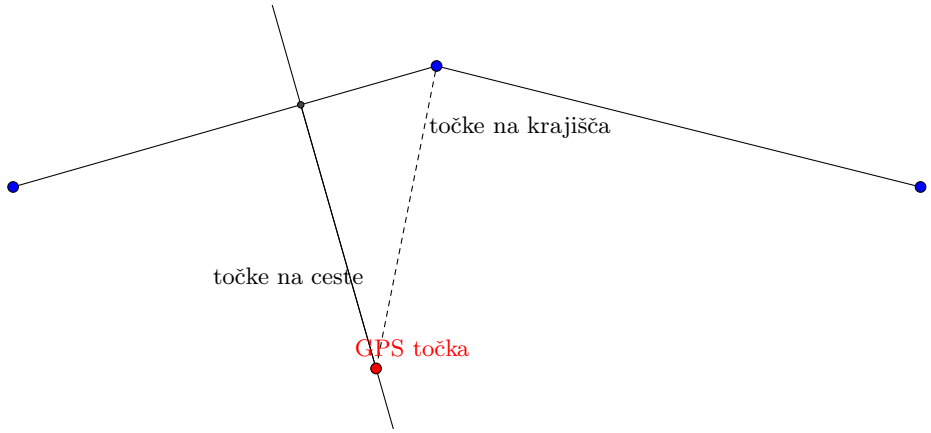
Izziv, ki nastane pri projeciranju GPS sledi na zemljevide, kljub ostalim virom napak, izvira predvsem iz dejstva, da so meritve koordinat z GPS napravami nenatančne. Ta nenatančnost lahko izvira iz večih virov; iz napake meritev na splošno in iz okoljskih motenj, kot so odboj signala od stavbe v urbanem okolju, odboj/odklon signala v ionosferi, napaka v času, ki smo ga dobili od satelita, napaka pri oddajanju lokacije satelita, ipd. Več o napakah meritev koordinat z GPS lahko preberete v [7].

Rešitve problema projeciranja GPS sledi na zemljevid so se pričele razvijati po letu 1990. Takrat so se pojavili prvi digitalni zemljevidi (od leta 1990) in malo kasneje tudi prvi prosti dostop do sistema GPS (1995). Pregled širokega nabora algoritmov si lahko ogledate v [9], nekaj pa jih bomo opisali v nadaljevanju.

3.2. **Osnovni Algoritmi.** Obstaja kar nekaj preprostih rešitev problema (1998 – 2001), ki pa seveda niso tako dobre, kot bolj zahtevni postopki. Pomembnejši predstavniki algoritmov so navedeni v nadaljevanju.

3.2.1. *Preslikava GPS točk na krajišča (ang. point to point matching).* Vsako točko preslikamo na najbližje krajišče cestnega odseka (glejte sliko 19). Ta metoda že v zelo preprostih primerih naredi ogromne napake.

Primer 3.2. Če je dolg cestni odsek označen samo z začetkom in koncem, potem se vse GPS točke preslikajo na dve točki zemljevida. Dve GPS točki, ki sta poljubno blizu, se lahko preslikata na GPS točki, ki sta poljubno daleč. To med drugim pomeni poljubno velike napake in nezveznost preslikave. Pot tudi ni nujno izvedljiva, saj nas algoritem lahko preslika na drugo stran stavbe. ♢



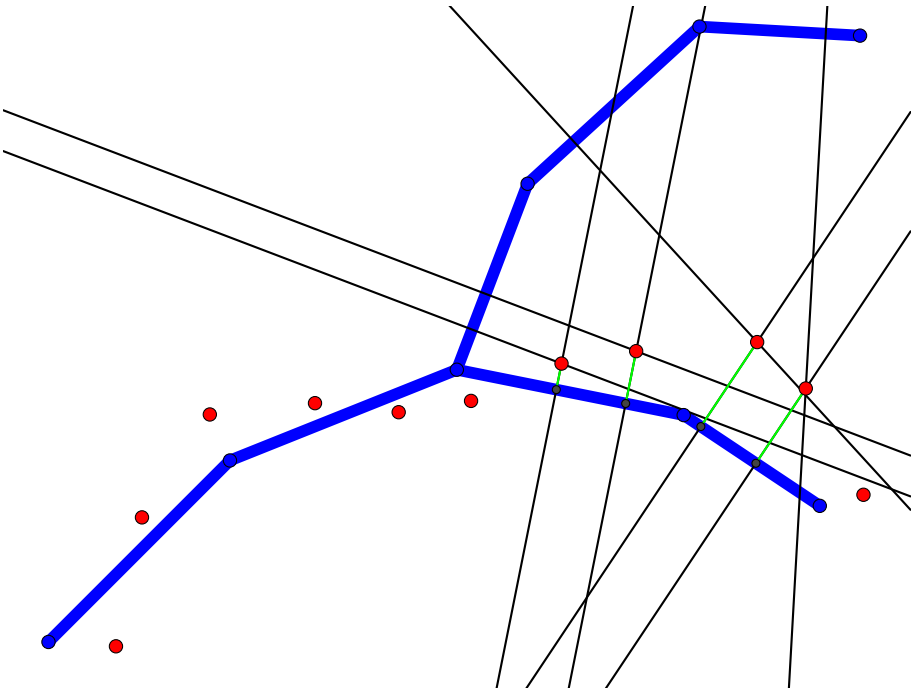
Slika 19. Prikaz preslikave GPS točk na zemljevid z algoritmoma Preslikava točk na krajišča in Preslikava točk na ceste.

3.2.2. *Preslikava GPS točk na ceste (ang. point to curve matching).* Vsako točko preslikamo na najbližjo točko najbližjega cestnega odseka (glejte sliko 19). Ta metoda je že veliko boljša, vendar ima še vedno veliko pomanjkljivost. Kot pri prvi metodi že majhne napake lahko pripeljejo do tega, da se točka pomotoma preslika na napačno stran cestišča, ali pa kar na sosednji cestni odsek. To lahko pripelje do poti, ki jih uporabnik ni mogel prevoziti.

3.2.3. *Preslikava GPS trajektorije na ceste (ang. curve to curve matching). Trajektorija* je krivulja, ki obišče zaporedje točk. Poizkušamo najti cestni odsek, ki je skupno najbližje GPS trajektoriji (glejte sliko 20). Pristop je že bolj podoben modernejšim pristopom, saj (posredno) upošteva tako zgodovino GPS sledi, kot trenutno stanje v okolici. Je pa tudi ta pristop občutljiv na napačne meritve, ki lahko podobno, kot v zgornjih primerih z majhno spremembo vhoda zelo spremenijo smer krivulje.

3.3. **Topološki algoritmi.** Topološki pristop je že uporabljal topologijo omrežja (2000) [14], kjer se upošteva usmeritve vozila, spremembe smeri, oddaljenost od cestnega odseka in nekatere dodatne podatke, da na križišču izračunamo, kam smo zavili. Ideja je, da nam dodatni podatki, predvsem smer, povejo več, kot le lokacija. Dodatne podatke poizkušamo uporabiti tudi pri iskanju najboljšega ujemanja.

Primer 3.3. Če je sprememba smeri manjša od $\pi/8$, smo na isti cesti kot prej. Če je večja, smo verjetno v križišču. Poiščemo vse sosednje cestne odseke (vključimo tudi cesto, po kateri potujemo) in za vsako izmed njih izračunamo katera je najbolj primerna. Za to uporabimo vnaprej definirano (ponavadi že optimizirano) kriterijsko



SLIKA 20. Prikaz preslikave GPS točk (rdeče) na cestno omrežje (modro) z algoritmom Preslikava trajektorij na ceste. Kot vidimo iz pomožnih črt (črno), je spodnja krivulja skupno bližja GPS sledi, kot zgornja, saj bo vse izmerjene GPS točke bližje (zelena) spodnjemu cestnemu odseku.

funkcijo, ki upošteva zgornje parametre (in podobne). Seveda v algoritmu tudi za določanje, ali ostajamo na isti cesti, uporabljamo bolj informativno kriterijsko funkcijo. ♢

Naj omenim, da taka rešitev potrebuje več kot le koordinate točk, torej več kot le najosnovnejši GPS signal. Če za meritve uporabljamo mobilne telefone, imamo vse potrebne senzorje na voljo, sicer pa potrebujemo GPS napravo, ki meri tudi smer. Preden uporabimo ta algoritem, poženemo na GPS sledi razširjen Kalmanov filter [3], ki izloči fizikalno nemogoče hitrosti, pospeške, spremembe smeri ipd. S tem se izognemo najhujšim napakam, ki bi jih algoritem naredil. To je pomembno, saj je problem te metode med drugim tudi, da morebitna napaka vodi k večjim napakam; potujemo namreč le po možnih povezavah na omrežju.

3.4. **Mehka logika.** (ang. fuzzy logic) Kot eden boljših pristopov v praksi je znan pristop z uporabo mehke logike [5], predstavljen leta 2006 [8]. V osnovi deluje s enakimi kazalci podobnosti, kot topološki algoritem. Le način izbire je prilagojen, da uporabi bolj primerno orodje. Deluje tako, da (kot pri topoloških algoritmih) "zdravo pamet" zakodiramo v pravila, ki jih potem apliciramo na naš problem. Pomembna razlika pa je, da dopustimo, da so spremenljivke in rezultati bolj ohlapni. Na primer za hitrost določimo tri skupine: nična, nizka in visoka. Za vsako od teh skupin in vsako izmerjeno hitrost povemo kakšna je verjetnost, da smo v skupini; ne postavimo torej strogih meja med množicami, ampak vsakemu elementu pripišemo še stopnjo vsebovanosti v množici.

Nato pa s pomočjo pravil (in rezultata, ki ga pravila vrnejo) povemo kakšna je verjetnost, da smo na nekem cestnem odseku. Primer pravila: če je hitrost nizka in

smo malo spremenili smer vožnje, potem smo zelo verjetno na isti cesti, kot smo bili pri prejšnji meritvi. Funkcije, ki povejo kakšna je verjetnost pripadanja skupini so lahko poljubne. Večinoma se uporablja t.i. s in z krivulje. To so krivulje določene oblike, na primer kot črka s in z. Natančneje so predstavljene v [8]. Natančno obliko teh krivulj, pa lahko optimiziramo, če imamo kakšno sled, za katero vemo, da je točna (smo na primer uporabili zelo natančne merilne naprave, ali pa smo jo ročno preverili).

Prednost te metode (poleg tega, da je bolj natančna) je preprostost zapisa, spreminjanja in dodajanja pravil. Preprostost dosežemo tako, da so pravila bolj podobna človeškemu jeziku, kot običajna programska koda, ki bi jo bilo za tak problem potrebno napisati.

V algoritmu imamo tri različne modele za prepoznanje dogodkov. Prvi in najpomembnejši algoritem je algoritem za iskanje začetnega cestnega odseka, saj morebitni napačni izbiri skoraj gotovo sledi še več napačnih izbir. Drugi algoritem je namenjen prepoznavanju ali smo še vedno na isti cesti kot prej. Tretji pa, da ugotovimo ali smo prečkali križišče (nismo več na isti cesti) in moramo ugotoviti na kateri od sosednjih cestnih odsekov se nahajamo. Za vsakega od teh primerov ustvarimo pravila, ki sestavljajo algoritem. Prvi algoritem pa je tako poseben, da pravilo uporabimo na več začetnih točkah in cestni odsek izberemo le, če jo večkrat potrdimo, saj se res ne želimo zmotiti.

Ko vsa ta pravila združimo, dobimo že izjemno natančen približek prave poti. Ima pa metoda, v primerjavi z naslednjo metodo, pomanjkljivost, da potrebujemo več, kot le osnovno GPS sled (potrebujemo vsaj še izmerjeno hitrost in smer vožnje). Pomanjkljivost metode je tudi, da je prvotna implementacija algoritma težja, kot pri ostalih metodah.

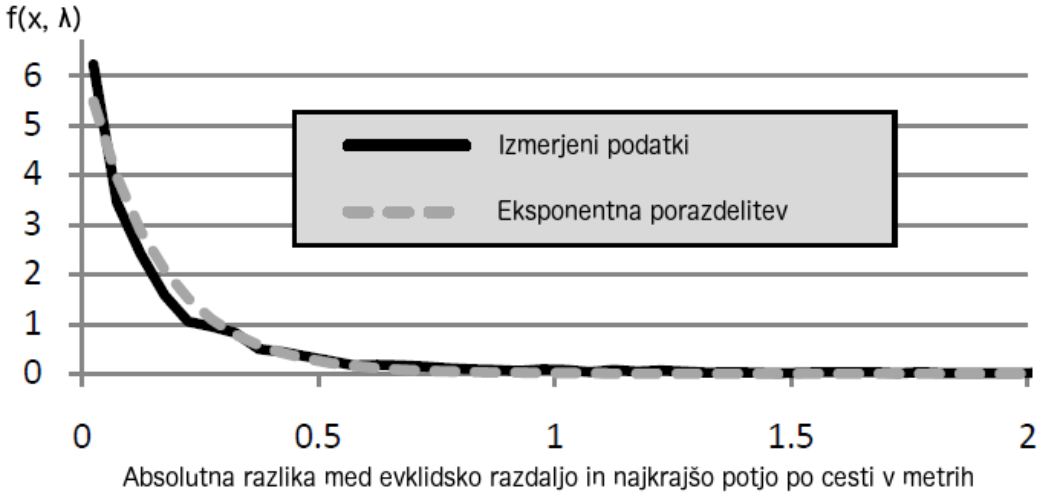
3.5. **Verjetnostni algoritmi - HMM.** (ang. hidden markov models)

To je pristop z uporabo teorije verjetnosti (2004), ki se je izkazal kot zelo uspešen [6]. Ideja algoritma je, da izračunamo zaporedje cestnih odsekov, ki smo jih najverjetneje prevozili. Ko govorimo o razdalji med cestnima odsekoma mislimo razdaljo med točkama, projeciranima na cestni odsek (zadnjo na prejšnjem cestnem odseku in prvo na trenutnem cestnem odseku). Za projeciranje obstaja več različnih pristopov, najpreprostejši pa je preslikava na najbližjo točko cestnega odseka. Enako projekcijo uporabljamo tudi pri računanju razdalje med točko in cestnim odsekom.

Očitno je, da je zaporedje dveh cestnih odsekov manj in manj verjetno, če je razdalja med njima na zemljevidu (torej koliko je najmanjša razdalja med njima, prevoženo po cestah) zelo različna od evklidske razdalje. To bi namreč pomenilo, da smo med meritvama naredili več ovinkov oziroma smo šli daleč okrog. Če GPS točke izmerimo vsakih nekaj sekund, je velika razlika zelo malo verjetna. Iz literature [6] sledi (glejte sliko 21), da je na resničnih podatkih opisana verjetnost razlike razdalj porazdeljena eksponentno (z razliko razdalj kot parametrom). Eksponentna porazdelitev je definirana s parametrom $\lambda > 0$ in z gostoto verjetnosti

$$f(x;\lambda) = \begin{cases} \lambda e^{-\lambda x} & x \geq 0, \\ 0 & x < 0. \end{cases}$$

Parameter λ lahko prilagajamo dejanskim podatkom, ali pa ga izračunamo vnaprej na vzorčni množici podatkov. Ponavadi vzorčno množico zgeneriramo z zajemanjem GPS točk iz digitalnega zemljevida na območju, kjer imamo GPS sledi. Ko smo izračunali oziroma izbrali λ, lahko izračunamo verjetnost vsakega zaporedja



Slika 21. Na GPS sledi izračunane razlike med evklidsko razdaljo in razdaljo po cestah (zelo natančno) izmerjenih GPS točk. Vir [6].

cestnih odsekov s pripadajočimi projekcijami GPS točk. Vemo namreč kakšne so prehodne verjetnosti med sosednjima cestnima odsekoma. Še vedno pa ne vemo na katerem cestnem odseku se nahajamo, ko izmerimo neko GPS točko.

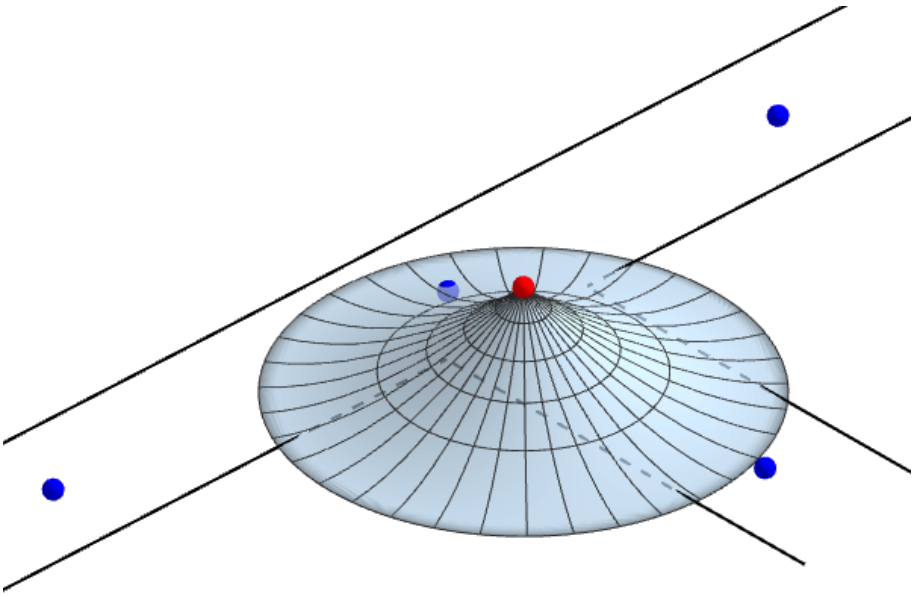
Predpostavimo, da so verjetnosti za izbiro cestnega odseka, ki pripada GPS točki porazdeljene z normalno porazdelitvijo, z razdaljo do odseka za parameter. Porazdelitev teoretično ni nujno normalna, saj GPS podatki zaradi odbojev in podobnih napak nimajo nujno normalno porazdeljene napake. Vendar je napaka v praksi pogosto porazdeljena normalno. Za izračun verjetnosti na katerem cestnem odseku smo sta naslednja stavka ekvivalentna.

- Napaka merjenja z GPS je normalno porazdeljena (z razdaljo kot parametrom) okrog točke, kjer se nahajamo in
- napaka merjenja z GPS je normalno porazdeljena (z razdaljo kot parametrom) okrog izmerjene točke.

To pomeni, da lahko pri zgoraj opisani normalni porazdeljeni napaki uporabimo kar standardno deviacijo napake GPS meritev. Povprečje porazdelitve je lokacija na cestnem odseku, kjer se nahajamo (glejte sliko 22). Več o oceni deviacije si lahko preberete v [6]. Sedaj za vsako točko vemo tudi kakšna je verjetnost, da se nahaja na vsakem cestnem odseku. Ogledali si bomo kako opisane verjetnosti uporabimo za izračun najverjetnejšega zaporedja cestnih odsekov.

Naj bo $T \in \{\mathbb{R}^+, \mathbb{N}_0\}$. Oglejmo si slučajni proces, ki ga tvori zaporedje nekih slučajnih spremenljivk $\{X_t\}_{t\in T}$, katerih zaloga vrednosti leži v S. Izmed vseh takih zaporedij je najenostavnejši primer zaporedje neodvisnih, enakomerno porazdeljenih slučajnih spremenljivk, kjer o prehodu med stanji S ne moremo veliko povedati. Prvi kompleksnejši primer, o katerem lahko več povemo je primer, ko so členi paroma odvisni. Naj bo vsak člen odvisen le od zadnjega člena pred njim, ne pa od prejšnjih.

Opisano lastnost imenujemo *markovska lastnost*, slučajnim procesom, ki ji zadoščajo, pa pravimo *markovski modeli*. Natančneje govorimo o avtonomnem markovskem modelu, saj ne dopuščamo zunanjih interakcij, kot jih na primer v markovskem odločitvenem procesu. Več o markovskih verigah, procesih, modelih, ipd.



Slika 22. Prikaz porazdelitve verjetnosti realne lokacije za izmerjeno GPS točko.

lahko preberete v [12]. Mi se bomo ukvarjali le z markovskimi modeli z diskretnim časom, torej s primeri, ko je $T = \mathbb{N}_0$.

Osnovni markovski model, ki bi ga želeli uporabiti so *markovske verige*. Te so sestavljene iz *stanj* S (cestni odseki našega omrežja, z projeciranimi GPS točkami) in *prehodnih verjetnosti* med stanji P. Prehodna verjetnost $p_{ij} \in P$ med stanjema $s_i \in S$ in $s_j \in S$ je verjetnost, da bo naslednje stanje sistema i, pri tem, da smo zdaj v stanju j. V našem primeru je to verjetnost prehoda iz cestnega odeska s_i na cestni odsek s_j, ki je porazdeljena eksponentno, kot je prikazano na sliki 21. Razdalja se meri od ene projecirane točke do druge.

Opremljeni s podatki o verjetnosti prehoda med cestnimi odseki lahko izračunamo kako verjetno bomo v vsakem stanju, torej na kateri cestni odsek (s pripadajočo GPS točko) bomo najverjetneje zavili. Če bi lahko zgradili tak model, bi bilo idealno, saj bi najverjetnejšo pot relativno preprosto izračunali. Problem je, da stanj modela ne moremo opazovati, tj. iz izmerjene GPS točke ne vemo zagotovo na katerem cestnem odseku smo.

Ker vemo kakšna je verjetnost, da smo v izbranem stanju, če smo izmerili neko GPS točko (verjetnosti porazdeljeno normalno, glejte sliko 22), vemo kakšna je verjetnost, da se nahajamo v vsakem stanju (na kateri cestnem odseku smo). Zato lahko posredno sklepamo tudi o verjetnosti, da bomo prešli v vsako naslednje stanje (torej kam bomo zavili). Tu dovolimo tudi cestne odseke, ki niso sosednji. Vendar je verjetnost, da izberemo njih veliko manjša, kot verjetnost, da izberemo sosednje cestne odseke.

Primer 3.4. Denimo, da imamo škatli A in B. V škatli A imamo 5 črnih krogel in 15 rdečih krogel, v škatli B pa 15 črnih in 5 rdečih krogel. Prijatelj iz nam neznane enakomerno naključno izbrane škatle enakomerno naključno izbere kroglo in nam jo poda. Vprašamo se, kakšna je iz našega vidika verjetnost, da je kroglo potegnil iz škatle A, če je krogla rdeča?

Verjetnost je $15/20 = 0.75$, saj je večina rdečih krogel v škatli A. In ker so bile vse izbire narejene enakomerno naključno je verjetnost odvisna le od števila rdečih krogel v škatli. Enako razmišljanje uporabimo za računanje verjetnosti prevoženega zaporedja cestnih odsekov. Razlika je le, da namesto enakomerne porazdelitve uporabljamo drugo porazdelitev in da iz škatel vzamemo več krogel zapored. ♢

Opisan markovski model imenujemo *prikriti markovski model* (ang. hidden markov model). Uporabili ga bomo za izračun najverjetnejšega zaporedja cestnih odsekov. Pri prikritih modelih se stanj ne da direktno opazovati, je pa viden izhod oziroma meritev (v našem primeru GPS točke). Več o prikritih markovskih modelih lahko preberete v [4]. Naš prikriti markovski model je formalno sestavljen iz

- Stanj, to je cestnih odsekov s pripadajočimi projekcijami GPS točk S,
- prehodnih verjetnosti P med stanji S, tj. opisana eksponentno porazdeljena verjetnost prehoda na sosednji cestni odsek,
- množice možnih opažanj, to je vseh možnih GPS točk O,
- množice izmerjenih GPS točk Y,
- verjetnosti opazovanj (ang. observation likelihoods) B, to je verjetnosti, da smo izmerili točko $y \in Y$, če se nahajamo v stanju $s \in S$, v našem primeru normalno porazdeljeno glede na razdaljo in
- začetne porazdelitve stanj p, tj. kakšna je verjetnost, da smo začeli na vsaki od cestnih odsekov s pripadajočimi projekcijami iz S.

Predpostavka markovskih lastnosti ni preveč omejujoča, saj so tako lokacija, kot smer in hitrost vožnje odvisne le od prejšnje točke, če točke niso zajete preveč pogosto. Ker so naše GPS točke zajete vsakih nekaj sekund, to pomeni, da je imelo vozilo dovolj časa za spremembo smeri in markovska lastnost drži.

Trditev 3.5. *Meritev je odvisna od stanja modela.*

Dokaz. V našem primeru je stanje cestni odsek, po kateri se vozimo in meritev z GPS izmerjeni koordinati. Denimo, da se nahajamo na cestnem odseku c in smo z GPS pri merjenju svojih koordinat, izmerili koordinati k. Če do napak pri merjenju ne bi prišlo, bi izmerili svojo eksaktno pozicijo. Meritev k je v tem primeru odvisna od cestnega odseka, saj mora ležati na njej.

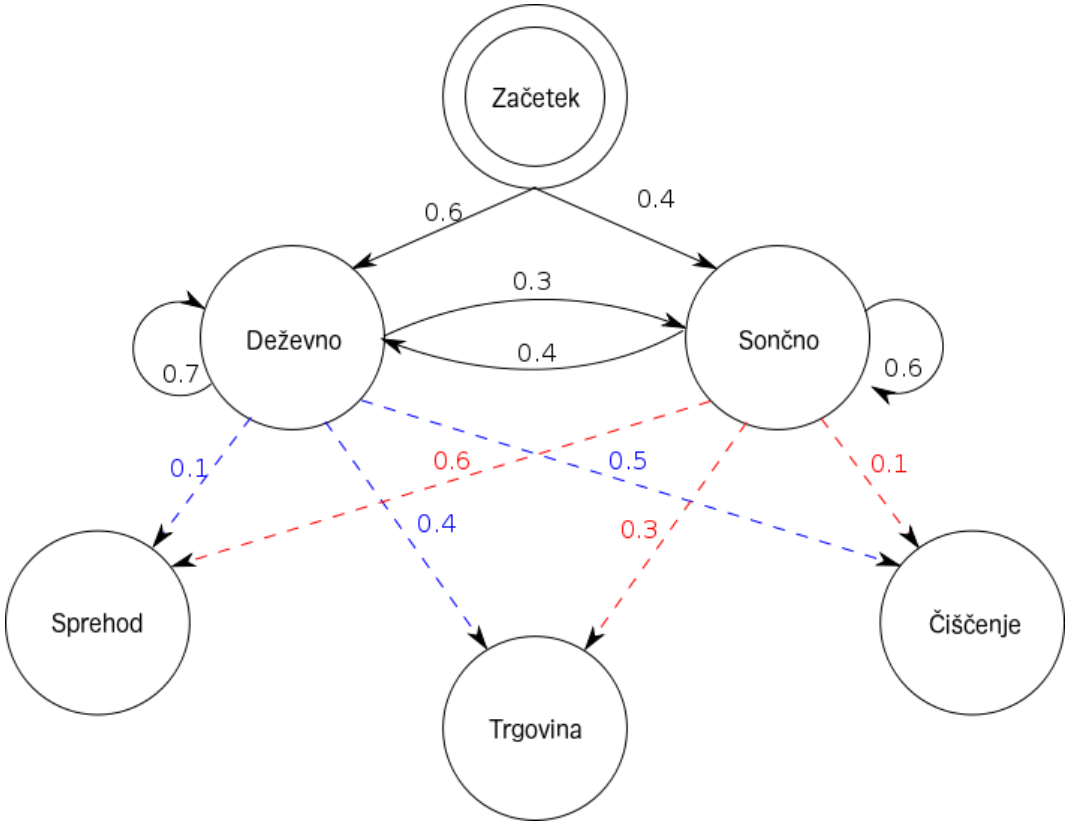
Mi smo predpostavili, da so meritve koordinat GPS normalno porazdeljene okrog resnične lokacije. V tem primeru velja, da je resnična lokacija $E[k]$ odvisna od cestnega odseka c. Sledi, da je meritev k odvisna od cestnega odseka c. Tudi če ne predpostavimo normalne porazdelitve, lahko še vedno sklepamo, da je meritev k vsaj malo odvisna od cestnega odseka c, saj se resnična lokacija, ki jo merimo nahaja na njej. Kako je meritev odvisna stanja pa se izraža kot funkcija napake meritve koordinat z GPS, napake pri odbojih in vseh ostalih napak. □

Da je meritev odvisna od stanja modela je pomembno, saj smo uspešno povezali meritev s cestnim odsekom. Kar pomeni, da lahko na podlagi stanja sklepamo o verjetnosti, da smo opazili meritev. Najverjetnejše zaporedje cestnih odsekov izračunamo z dinamičnim programiranjem, natančneje z uporabo Viterbijevega algoritma [2]. Algoritem deluje tako, da s pomočjo začetne porazdelitve (verjetnosti, da smo v vsakem od stanj) za vsako stanje izračuna najverjetnejšo pot dolžine 1, ki se konča v njem. Nato pa za vsako naslednjo dolžino n in za vsako stanje izračuna najverjetnejšo pot, ki se konča v njem. Pri tem si pomaga s najverjetnejšimi potmi dolžine $n - 1$, ki jih je že izračunal.

Primer 3.6. Za lažjo predstavo in vse prehodne verjetnosti glejte spodnjo sliko. Preprost človek nam vsak dan poroča, kaj je tisti dan delal. Ker je preprost, se na podlagi vremena (je deževno ali sončno) za opazovan dan z določenimi verjetnostmi (glejte sliko 23) odloči ali bo šel na sprehod, v trgovino ali pa bo čistil hišo. Vemo, da če danes dežuje, je verjetnost, da bo deževalo naslednji dan 0.7 in verjetnost, da bo naslednji dan sončno 0.3.

Naša naloga je, da iz zaporedja njegovih aktivnosti napovemo kakšno je bilo vreme v preteklih dneh. Denimo, da nam je sporočil, da se je zadnji dan sprehajal. Verjetnost, da je dan preden nam je prvič poročal sončno je 0.4 in verjetnost, da je deževno 0.6 (začetek). Verjetnost, da je prvi dan deževalo je enaka verjetnosti, da je dan prej deževno in je ostalo deževno. Upoštevati moramo tudi verjetnost, da je javil, da je šel na sprehod, glede na to, če bi deževalo. In podobno za primer, ko je bilo prejšnji dan sonce. Verjetnost je torej enaka $(0.6 * 0.7 + 0.4 * 0.4) * 0.1$.

Ko enako izračunamo za sončno stanje smo izračunali najverjetnejše zaporedje dolžine 1 za vsako od stanj. Postopek rekurzivno ponovimo za vse naslednje dolžine. ♢



Slika 23. Primer prikrite markovske verige. Vir: [25].

V psevdokodi algoritem napišemo takole.

```
def viterbi(O, S, p, Y, P, B):
    n = len(S)
    x = [0] * n
    for i in range(n):
        T1[i, 1] = p[i] * B[i, Y[i]]
        T2[i, 1] = 0
```

```
    for i in range(2, n + 1):
        for j in range(n):
            T1[j, i] = max([T1[k, i-1] * P[k, j] * B[j, Y[i]] for k])
            T2[j, i] = argmax([T1[k, i-1] * P[k, j] * B[j, Y[i]] for k])
    z[T] = argmax([T1[k, n] for k])
    x[T] = s[z[n]]
    for i in range(n, 1, -1):
        z[i-1] = T2[z[i], i]
        x[i-1] = s[z[i-1]]
    return x
```

Trditev 3.7. *Zgoraj opisani Viterbijev algoritem vrača najverjetnejše zaporedje stanj za dano zaporedje opazovanj.*

Dokaz. Naj bo S množica stanj, naj bo s_k trenutno stanje in n število stanj. Naj bo $P = \{P_{ij}\}$ prehodna matrika modela in $B = \{b_{ij}\}$ verjetnosti, da smo v stanju i izmerili opazovanje j. Označimo optimalno zaporedje dolžine i, ki se konča v s_j z $o_i(s_j)$. Enako oznako uporabimo tudi za verjetnost tega zaporedja (zgoraj shranjeno v T_1 in T_2). Ker o preteklih stanjih ne vemo nič, za začetno porazdelitev vzamemo kar enakomerno porazdelitev.

Dokaz bo potekal z indukcijo. Za $i = 1$ vzamemo kar najverjetnejše stanje, kot je opisano v primeru 3.6. Dobili smo najverjetnejšo pot dolžine 1. Ker je vsako od zaporedij dolžine i, ki se konča v stanju s_j stanje optimalno (za zaporedja, ki se končajo v s_j), je optimalno zaporedje dolžine $i+1$, ki se konča v s_k eno od zaporedij $\{o_i(s_j)\}_{j\in[n]}$, ki mu dodamo trenutno stanje s_k. Natančneje je to tisto zaporedje, ki maksimizira verjetnost:

$$o_{i+1}(s_k) = \max_{j\in[n]} o_i(s_j)p_{jk}b_{ko_i}.$$

Ker na koncu vzamemo najverjetnejše od zaporedij iskane dolžine, je izračunano zaporedje cestnih odsekov res najverjetnejše (izmed vseh zaporedij enake dolžine). □

Algoritem s prikritimi markovskimi verigami je posebej uporaben, ker potrebuje le GPS točke in ga je relativno preprosto implementirati. Prav tako se lahko (lokalno) optimalno rešitev gradi sproti, če se temu prilagodi algoritem. Inkrementalni algoritem z uporabo prikritih markovskih verig je opisan v [13]. Res pa je algoritem z prikritimi markovskimi verigami v povprečju malo manj natančen, kot algoritem z mehko logiko.

3.6. **Pregled opisanih metod.** V tabeli so naštete opisane metode z učinkovitostjo in tipom vhodnih podatkov. Podrobnejšo tabelo z večimi metodami in parametri lahko najdete v [9]. DR pomeni, da potrebujemo več kot le GPS podatke, 1 : 2500 je merilo zemljevida, ki je uporabljen, napaka pa je povprečna napaka pri postavitvi GPS točke.

Metoda	Tip podatkov	Napaka	Delež pravilnih c. odsekov
White el al. (2000)	GPS	-	85.8 % ptc
Quddus et al. (2003)	GPS/DR, 1:2500	18.1 m	88.6 % topology
Ochieng et al. (2004)	GPS/DR, 1:2500	9.1 m	98.1 % HMM
Quddus et al. (2006)	GPS/DR, 1:2500	5.5 m	99.2 % fuzzy

4. Zaključek

Skozi delo smo prepoznali pomembne značilke, za katere ocenimo, da vplivajo na varnost, udobnost in kvaliteto vožnje. Na podlagi GPS sledi, ki smo jih zbrali in obdelali, smo jih tudi izračunali in podali oceno za zanesljivost izračunanih značilk. Preden smo izračune lahko natančno opravili, smo morali GPS sledi najprej preslikati na (digitalno) cestno omrežje. Ta proces smo opisali in naredili pregled uspešnih znanih algoritmov, ki ga rešujejo.

Iz značilk smo primerjali kvaliteto vožnje različnih voznikov. Opazili smo očitno razliko med poklicnimi in ostalimi vozniki. Poklicni vozniki so se v povprečju odrezali veliko bolje, kot ostali vozniki. Obstaja pa tudi nezanemarljiv delež ostalih voznikov, ki vozijo enako dobro, kot poklicni. Glavni razlikovalec med vozniki je značilka varnosti, glavni rezultat dela pa klasifikator kvalitete vožnje voznikov, glede na izbrane značilke, prek katerega smo voznike smo uspešno razporedili v tri skupine.

- Tiste, ki vozijo neudobno (in varno), tj. imajo $U > 0.2$,
- tiste, ki vozijo nevarno (in udobno), tj. imajo $V > 7$,
- ter tiste, ki vozijo varno, tj. imajo $U \leq 0.2$ in $V \leq 7$.

Največ je takih, ki vozijo varno in udobno, približno petina jih vozi nevarno in udobno, nekaj pa tudi varno in neudobno.

V nadaljnjih delih bi bilo zanimivo poleg GPS podatkov zbrati še druge relevantne podatke, na primer take, kot so opisani v podpoglavju 2.2.3. Prav tako bi bilo zanimivo imeti podatke, kjer so sledi/vožnje neodvisno od podatkov ocenjene s strani strokovnjakov ali potnikov. Opremljeni z njimi bi se lahko lotili strojnega učenja, kjer bi lahko ocenili, kako pomembne so naše izbrane značilke in mogoče odkrili nove, na katere nismo pomislili. Z modelom bi potem lahko ocenili tudi sledi, za katere še nimamo ocene.

Literatura

[1] A. Borštnik, *Nalaganje GPS sledi v podatkovno bazo*, [ogled 5. 9. 2017], dostopno na `https://github.com/andrejborstnik/gpx2pgsql`.

[2] G. D. Forney, *The viterbi algorithm*, In Proceedings of the IEEE **61.3**, 1973, str. 268-278.

[3] S. J. Julier; J. K. Uhlmann *A new extension of the Kalman filter to nonlinear systems*, Int. symp. aerospace/defense sensing, simul. and controls, 1997, str. 182-193.

[4] D. Jurafsky, *Hidden Markov Models*, [ogled 5. 9. 2017], dostopno na `https://web.stanford.edu/~jurafsky/slp3/9.pdf`.

[5] G. Klir, B. Yuan. *Fuzzy sets and fuzzy logic*, New Jersey: Prentice hall, 1995.

[6] P. Newson, J. Krumm. *Hidden Markov map matching through noise and sparseness*, Proceedings of the 17th ACM SIGSPATIAL international conference on advances in geographic information systems, ACM, 2009, str. 336-343.

[7] B. W. Parkinson, *GPS error analysis*, Global Positioning System: Theory and Applications **1**, 1996, str. 469-483.

[8] M. A. Quddus, R. B. Noland, W. Y. Ochieng, *A high accuracy fuzzy logic based map matching algorithm for road transport*, Journal of Intelligent Transportation Systems **10.3**, 2006, str. 103-115.

[9] M. A. Quddus, W. Y. Ochieng, R. B. Noland, *Current map-matching algorithms for transport applications: State-of-the art and future research directions*, Transportation research part c: Emerging technologies **15.5**, 2007, str. 312-328.

[10] X. Rui, D. Wunsch, *Survey of clustering algorithms*, IEEE Transactions on neural networks **16.3**, 2005, str. 645-678.

[11] P. K. Sen, J. M. Singer, *Large Sample Methods in Statistics: An Introduction with Applications*, CRC press, 1994.

[12] W. J. Stewart, *Probability, Markov chains, queues, and simulation: the mathematical basis of performance modeling*, Princeton University Press, 2009.

[13] P. Szwed, P. Kamil, *An incremental map-matching algorithm based on hidden markov model*, International Conference on Artificial Intelligence and Soft Computing, Springer, Cham, 2014.

[14] C. E. White, D. Bernstein, A. L. Kornhauser, *Some map-matching algorithms for personal navigation assistants*, Transportation Research Part C **8**, 2000, str. 91–108.

[15] *EPSG koordinate*, [ogled 5. 9. 2017], dostopno na `https://epsg.io/`.

[16] *Goopti d.o.o.*, [ogled 5. 9. 2017], dostopno na `https://www.goopti.com`.

[17] *Hard brake & hard acceleration*, [ogled 5. 9. 2017], dostopno na `http://tracknet.accountsupport.com/wp-content/uploads/Verizon/Hard-Brake-Hard-Acceleration.pdf`.

[18] *Open Source Routing Machine*, [ogled 5. 9. 2017], dostopno na `http://project-osrm.org/`.

[19] *OpenStreetMap*, [ogled 5. 9. 2017], dostopno na `http://www.openstreetmap.org/`.

[20] *OSM arhiv GPS sledi*, [ogled 5. 9. 2017], dostopno na `http://planet.openstreetmap.org/gps/`.

[21] *Python*, [ogled 5. 9. 2017], dostopno na `https://www.python.org/`.

[22] *PostGIS*, [ogled 5. 9. 2017], dostopno na `http://postgis.net/`.

[23] *PostgreSQL*, [ogled 5. 9. 2017], dostopno na `https://www.postgresql.org/`.

[24] *Vizualizacija kotov pri računanju razdalje na sferi*, [ogled 5. 9. 2017], dostopno na `https://en.wikipedia.org/wiki/Great-circle_distance#/media/File:Central_angle.svg`.

[25] *Vizualizacija prikritega markovskega modela*, [ogled 5. 9. 2017], dostopno na `https://en.wikipedia.org/wiki/Hidden_Markov_model#/media/File:HMMGraph.svg`.