Auteursrechterlijke overeenkomst

Opdat de Universiteit Hasselt uw eindverhandeling wereldwijd kan reproduceren, vertalen en distribueren is uw akkoord voor deze overeenkomst noodzakelijk. Gelieve de tijd te nemen om deze overeenkomst door te nemen, de gevraagde informatie in te vullen (en de overeenkomst te ondertekenen en af te geven).

Ik/wij verlenen het wereldwijde auteursrecht voor de ingediende eindverhandeling met

Titel: Verzamelen en minen van tijdsruimtelijke data met oog voor privacy
Richting: 2de masterjaar in de informatica - databases Jaar: 2009

in alle mogelijke mediaformaten, - bestaande en in de toekomst te ontwikkelen - , aan de Universiteit Hasselt.

Niet tegenstaand deze toekenning van het auteursrecht aan de Universiteit Hasselt behoud ik als auteur het recht om de eindverhandeling, - in zijn geheel of gedeeltelijk -, vrij te reproduceren, (her)publiceren of distribueren zonder de toelating te moeten verkrijgen van de Universiteit Hasselt.

Ik bevestig dat de eindverhandeling mijn origineel werk is, en dat ik het recht heb om de rechten te verlenen die in deze overeenkomst worden beschreven. Ik verklaar tevens dat de eindverhandeling, naar mijn weten, het auteursrecht van anderen niet overtreedt.

Ik verklaar tevens dat ik voor het materiaal in de eindverhandeling dat beschermd wordt door het auteursrecht, de nodige toelatingen heb verkregen zodat ik deze ook aan de Universiteit Hasselt kan overdragen en dat dit duidelijk in de tekst en inhoud van de eindverhandeling werd genotificeerd.

Universiteit Hasselt zal mij als auteur(s) van de eindverhandeling identificeren en zal geen wijzigingen aanbrengen aan de eindverhandeling, uitgezonderd deze toegelaten door deze overeenkomst.

Ik ga akkoord,

VANGOIDSENHOVEN, Dries

Datum: 14.12.2009

# *Verzamelen en minen van tijdsruimtelijke data met oog voor privacy*

**Dries Vangoidsenhoven**

promotor :
Prof. dr. Bart KUIJPERS


Eindverhandeling voorgedragen tot het bekomen van de graad master in de informatica databases

# Abstract

In de hedendaagse samenleving wordt het verkeer alsmaar drukker. Om het verkeer te bestuderen wordt er gebruik gemaakt van trajecten van auto's, fietsers, voetgangers, … . Deze trajecten moeten natuurlijk eerst afgelegd en opgeslagen worden. Er bestaan al technieken voor trajecten te construeren uit GPS data, we onderzoeken of het mogelijk is om dit ook te doen gebruik makend van GSM data. Verder willen we deze trajecten modelleren en minen. We onderzoeken verschillende mining technieken die we kunnen toepassen op tijd-ruimtelijke data. Verder bespreken we ook beads, een manier om de onzekerheid over de positie van een bewegend object tussen twee punten te modelleren. We moeten ook rekening houden met de privacy van de personen wier gegevens we gebruiken. Daarom onderzoeken we de technieken om aan de wetten op de privacy te voldoen. We implementeren het bead-algoritme en een geometrisch algoritme en voeren hier experimenten mee uit. Door punten weg te laten uit GPS-data proberen we GSM-data te benaderen. Hieruit kunnen we concluderen dat het niet mogelijk is om louter en alleen door gebruik te maken van hand-over data (GSM-data) trajecten te reconstrueren. We kunnen wel concluderen dat GPS data in een stad overeenkomt met GSM data op autosnelwegen.

# Dankwoord

Bij het maken van deze thesis heb ik van een aantal mensen hulp gehad. Deze verdienen dan ook allemaal een bedankje. Ten eerste zou ik graag mijn promotor Prof. dr. Bart Kuijpers en mijn begeleider Bart Moelans bedanken voor hun hulp bij het tot stand brengen van deze thesis. Dankzij hen heb ik ook mijn hulp mogen verlenen bij het maken van een nieuwe paper! Vervolgens wil ik ook Kristof Gheys bedanken voor zijn uitleg omdat ik toch op zijn werk verder gewerkt heb. Ook wil ik mijn vriendin Johanna bedanken voor de morele steun die ze me gegeven heeft tijdens deze drukke periode en gedurende mijn studentenjaren. Ook mijn vrienden en ouders verdienen een bedankje voor hun steun. Tot slot wil ik nog mijn proffen en medestudenten bedanken voor de vijf jaar die ik doorgebracht heb aan deze universiteit!

# Inhoudsopgave

# Hoofdstuk 1: Mining

[2,7]

## 1.1 Inleiding

De ontwikkeling van verschillende data mining technieken zoals *frequent set* mining, *association rule* mining, *classification*, *prediction* en *clustering* voor tijdsdata en ruimtelijke data is pas vanaf de tweede helft van de jaren 1990 begonnen. Voor tijd-ruimtelijke data is deze studie nog maar pas begonnen. In dit hoofdstuk wordt de nadruk gelegd op het minen van trajecten van bewegende objecten.

In de laatste vijf jaar heeft men pogingen ondernomen om vele technieken voor het ontdekken van kennis in klassieke relationele data zoals clustering, classificatie, association rule mining, … uit te breiden. Het onderzoek hiernaar heeft nog geen theoretisch framework opgeleverd. Daarom is onderzoek in data mining in de context van bewegende objecten extra uitdagend. Wanneer bewegende objecten en trajecten gebruikt worden om verkeer te modelleren, kan er gezocht worden naar manieren om files te detecteren. Ook manieren om files te voorspellen en om relaties tussen twee files te zoeken. Een voorbeeld van een query die moet kunnen worden beantwoord is de volgende:

*Geef alle files in Brussel tussen 7 en 9 uur.*

Een verkeersopstopping wordt gedefinieerd aan de hand van de snelheid en de dichtheid van het verkeer. Hieruit volgt een logische link met *clustering*. Clustering is de classificatie van data in verschillende deelgroepen, clusters genaamd, zodat alle elementen van een cluster een gemeenschappelijke eigenschap delen. Omdat er verscheidene manieren zijn om afstand of mate van gelijkaardigheid te definiëren tussen twee trajecten, zijn verschillende variaties van clustering mogelijk. Typisch voor bewegende objecten is dat ze een snelheid hebben, en clustering kan aangewend worden om gelijkaardig bewegende objecten te vinden. Zo zoekt de volgende query naar drie clusters, een cluster voor de voetgangers, één voor de fietsers en één voor de auto's.

*Zoek drie clusters van objecten met gelijke snelheid (traag, middelmatig en snel).*

Soms wordt verwacht dat fysieke eigenschappen van trajecten van bewegende objecten, zoals lengte, snelheid en versnelling, een rol spelen in de kennis die men wil ontdekken. In veel gevallen is er een relatie tussen twee verkeersopstoppingen. Relaties tussen twee files of verkeersopstoppingen kunnen uitgedrukt worden aan de hand van *association rules*, zoals in het volgende voorbeeld:

*file(Lummen, 7u30) => file(Hasselt, 8u00).*

Dit wil zeggen dat als er om half acht een file is in Lummen, er naar alle waarschijnlijkheid om acht uur een file zal zijn in Hasselt. Meer algemener kan dit patroon als volgt voorgesteld worden:

*file(Lummen, t) => file(Hasselt, t + 30min).*

De support van een itemset is het percentage transacties die deze itemset bevatten. Analoog aan dit voorbeeld kunnen er *frequent* patronen ontdekt worden in data bestaande uit trajecten. Classificatie van trajecten is dan weer moeilijker. Een mogelijke classificatie kan gebeuren aan de hand van de verkeerssituatie, zoals het onderscheiden van normaal verkeer van een verkeersopstopping, of aan

de hand van het doel van de bewegende objecten, zoals woon/werkverkeer, verkeer van mensen die gaan winkelen, mensen die naar een bepaald evenement gaan, enzoverder. In tegenstelling tot *classification* zijn er wel veel mogelijkheden om *sequential* patronen te ontdekken in data van trajecten. Veronderstel dat we gebeurtenissen associëren met trajecten zoals het doorkomen op locaties A, B en C, dan kunnen we uit het patroon

$$A \rightarrow B \rightarrow C$$

afleiden dat A, B en C in die volgorde voorkomen. Uit het patroon

$$A \rightarrow_3 B \rightarrow_7 C$$

kan afgeleid worden dat er tussen A en B een vertraging optreed van 3 minuten, en tussen B en C een vertraging van 7 minuten.

Een andere klasse van tijd-ruimtelijke patronen is die van de tijd-ruimtelijke trends. Een voorbeeld van een trend is bijvoorbeeld:

*De snelheid van objecten neemt toe wanneer ze van Brussel weg bewegen.*

Sommige patronen kunnen gezien worden als een query. Het volgend voorbeeld illustreert dit.

*Vind alle periodieke patronen (voor een gegeven periode).*

Een patroon wordt gedefinieerd als periodiek als het genoeg en door dezelfde objecten herhaald worden in vaste tijdsintervallen. Ook andere gedragspatronen als verkeersopstoppingen en kuddegedrag horen thuis in deze categorie.

Een laatste categorie is die van extrapolatie van trajecten. Een voorbeeld is de volgende query

*Hoeveel trajecten zullen morgen om 9 uur Brussel kruisen?*

In het verdere verloop van dit hoofdstuk wordt er dieper op de hierboven besproken categorieën ingegaan.

## 1.2 Clustering

Wanneer we een grote hoeveelheid data willen analyseren is het handig als deze opgedeeld kunnen worden in groepen die logisch te onderscheiden zijn van elkaar. Een goede opdeling zou zijn dat alle objecten van een groep een eigenschap of meerdere eigenschappen hebben die ze gemeenschappelijk hebben en die objecten van andere groepen niet hebben. *Clustering* zorgt ervoor dat een grote hoeveelheid data opgedeeld wordt in kleinere groepen (clusters) doordat elk object aan een groep toegevoegd wordt.

In de context van bewegende objecten en dus van trajecten die hun beweging beschrijven, houdt clustering in dat er groepen afgezonderd worden met objecten die een gelijkaardig gedrag vertonen. Zoals bij andere vormen van complexe data zijn er twee technieken om clustering toe te passen, namelijk *clustering* van trajecten gebaseerd op afstand en traject-specifieke *clustering*.

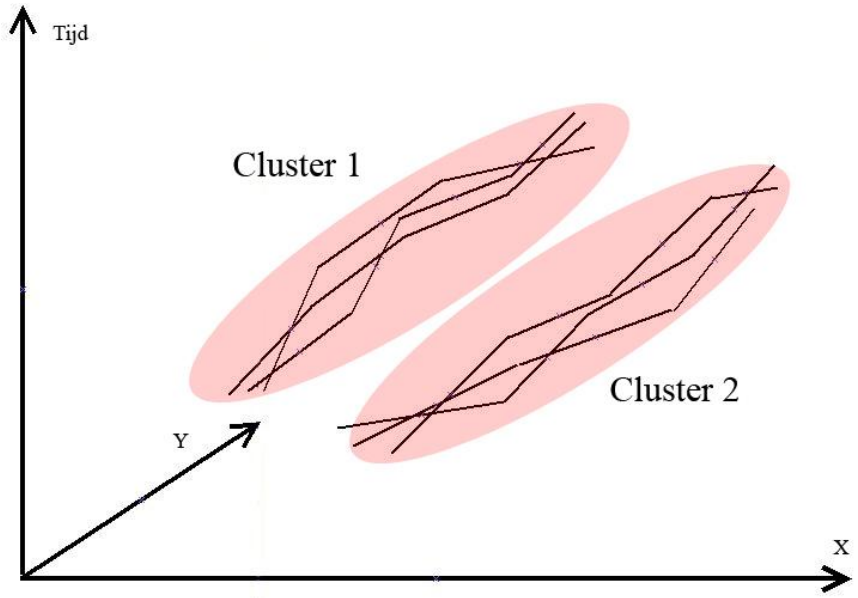
### 1.2.1 Clustering van trajecten gebaseerd op similariteit

Similariteit kan gemeten worden met een afstandsfunctie. Door een afstand te definiëren tussen objecten wordt bepaald welke objecten tot dezelfde cluster behoren en wat voor een cluster we ontdekt hebben. Het gebruikte *clustering* algoritme zorgt er op zijn beurt voor hoe elke cluster samengesteld wordt. Zo zal een centrum-gebaseerd algoritme als k-means *clustering* een verzameling bolvormige en compacte clusters genereren. Een hiërarchisch algoritme zal clusters onderbrengen in een structuur met clusters en sub-clusters. Ten slotte zullen algoritmes gebaseerd op dichtheid zoals PB-scan clusters vormen met een zo groot mogelijke dichtheid.

Om afstand te definiëren beschouwen we paren van objecten gelijk als ze ongeveer hetzelfde tijd-ruimtelijke traject volgen, meer bepaald dat ze op elk ogenblik zich ongeveer op dezelfde plaats bevinden. Elke cluster kan afhankelijk van de context een groep vrienden voorstellen die samen reist, een kudde schapen, een konvooi vrachtwagens, etc. Door objecten met zo een afstand te clusteren moeten we queries kunnen beantwoorden van de volgende vorm:

*Welke objecten van een populatie bewegen tesamen?*

In figuur 1.1 is een populatie opgedeeld in twee clusters die bestaan uit ongeveer gelijkaardige trajecten.




Figuur 1.1 Twee clusters

Zoeken naar bewegende objecten die op hetzelfde ogenblik in dezelfde richting bewegen is soms een te grote beperking om bruikbare informatie te vinden. De beperking van tijd kan verwijderd worden om dit probleem aan te pakken. Zo kunnen we zoeken naar bewegende objecten die dezelfde route volgen, maar niet noodzakelijk op hetzelfde tijdstip. We willen dus queries kunnen beantwoorden als de volgende:

*Zoek groepen van objecten die bewegen langs dezelfde route.*

Een tweede stap in het verwijderen van beperkingen is niet eisen dat de trajecten op dezelfde plaats moeten liggen. Er wordt dus gezocht naar bewegende objecten die gelijkaardige bewegingen maken, zoals in dezelfde richting bewegen of dezelfde rotaties uitvoeren.

### 1.2.2 Traject-specifieke clustering

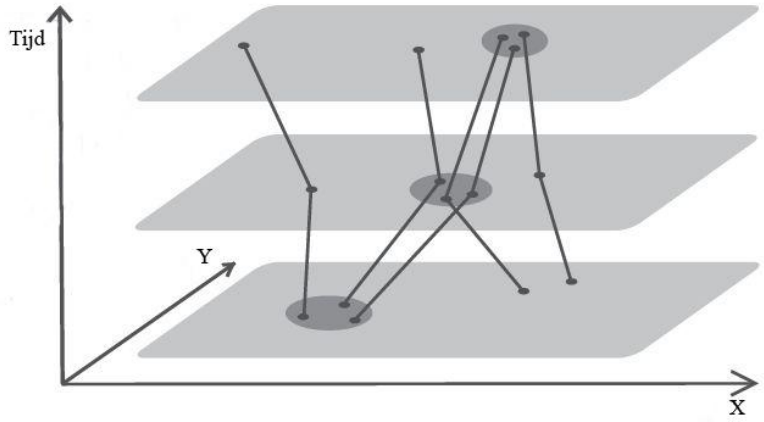
Methodes voor *clustering* die enkel gebaseerd zijn op afstand hebben enkele beperkingen. Zo kunnen sommige clusters niet op een gemakkelijke manier gemodelleerd worden en sommige technieken om de functionering te verbeteren kunnen niet uitgevoerd worden. Daarom werd er een *clustering* methode voorgesteld voor continue trajecten die objecten groepeert die gegenereerd zijn uit eenzelfde kern van een traject door het toevoegen van *Gaussian noise*.

Een andere benadering is gebaseerd op de zoektocht naar deelsegmenten van trajecten die sterk overeenkomen. Trajecten worden voorgesteld door korte lijnstukjes, mogelijk met ontbrekende stukjes. Vervolgens wordt een close time interval voor een groep van trajecten gedefinieerd als het maximale interval zodat alle objecten paarsgewijs dicht bij elkaar liggen. Groepen van trajecten worden geassocieerd met een factor die de hoeveelheid tijd uitdrukt waarin trajecten dichtbij elkaar liggen. Vervolgens wordt er gemined naar alle groepen van trajecten wiens factor boven een bepaalde grens ligt. In deze benadering wordt de grens op het begin gezet om zo een definitie te geven voor trajecten die dicht bij elkaar liggen in de ruimte. Vervolgens probeert de methode clusters van segmenten van trajecten te zoeken met maximale grootte en maximale tijd.

Een gelijkaardige maar vereenvoudigde methode is een uitbreiding op *micro-clustering*. Deze methode groepeert rechtlijnige segmenten van trajecten die in een rechthoek van bepaalde grootte liggen en in een bepaald tijdsinterval. Ook hier wordt ruimtelijke nabijheid gedefinieerd aan de hand van grenzen, namelijk de afmetingen van de rechthoek.

Een andere methode voor hetzelfde probleem houdt in dat trajecten voorgesteld worden door opeenvolgingen van punten zonder informatie over de tijd. Vervolgens wordt er een eenvoudige heuristiek toegepast die elk traject onderverdeeld in een verzameling lijnstukjes. Vervolgens worden alle zulke lijnstukjes gegroepeerd door een clusteringsmethode gebaseerd op dichtheid. Op het einde wordt voor elke cluster een representatief traject geconstrueerd.

*Time slicing* wordt gebruikt in een methode met hetzelfde doel maar bekeken vanuit een ander perspectief. Bewegende objecten worden geassocieerd met een ruimtelijke positie voor een verzameling *time slices*. Hierna gaat men ruimtelijke clusters die gebaseerd zijn op dichtheid proberen te ontdekken. Deze clusters moeten voorkomen in enkele opeenvolgende *time slices*. In figuur 1.2 wordt een voorbeeld van *time slicing* grafisch voorgesteld.




Figuur 1.2 Time slices met bewegende cluster

## 1.3 Locale patronen

In tegenstelling tot globale modellen zoals clustering die de volledige data willen typeren of verklaren, wordt er geprobeerd om slechts in een deel van de data locale patronen te minen. Dit zijn patronen die kleine delen van de data willen typeren, in kleine tijdsintervallen, beperkte groepen van objecten, in bepaalde gebieden, … .

### 1.3.1 Extractie van frequente patronen

Frequente patronen zijn een basisonderdeel van data mining. Een eenvoudige en vaak gebruikte methode om frequente tijd-ruimtelijke patronen te minen gaat als volgt te werk. In een eerste stap worden groepen van kenmerken afgeleid van de data die tijd-ruimtelijke predicaten teruggeven die elk traject beschrijven. In een volgende stap worden algemene mining algoritmes toegepast op de nieuwe voorstelling van de data die frequente sets, *association rules*, of frequente reeksen van kenmerken afleiden. In deze methode wordt er alleen in de pre-processing stap rekening gehouden met de semantiek van de tijd-ruimtelijke data. Vervolgens wordt het niet meer in rekening gebracht in de mining fase. De verscheidenheid aan frequente patronen die we kunnen minen met deze methode is zeer breed, beginnend bij eenvoudige *rules,* zoals volgend voorbeeld

*Lengte(traject) > 50 km => gemiddelde_snelheid(traject) > 60km/u*

tot patronen met betrekking tot complexe handelingen van objecten (bijvoorbeeld verkeersopstoppingen). Deze benadering komt overeen met de tijd-ruimtelijke *association rules* en *evolution rules*. *Association rules* drukken relaties uit tussen eenvoudige ruimtelijke-, niet-ruimtelijke- en tijds-predicaten. *Evolution rules* daarentegen bestaan uit complexere predicaten die de tijd-ruimtelijke evolutie van een object of groep van objecten beschrijven. Het is voor de hand liggend dat de keuze van de attributen die we gebruiken cruciaal is in het mining proces aangezien zij de patronen definiëren die gezocht moeten worden. Een basisverzameling van eigenschappen voor trajecten van bewegende objecten bestaat uit eigenschappen gebaseerd op individuele kenmerken, meer bepaald degene die het gedrag beschrijven van elk object afzonderlijk. Voorbeelden zijn:

- Ruimtelijke en/of tijdstotalen, zoals de lengte van het pad, de minimale/gemiddelde/maximale snelheid, de hoeveelheid tijd gespendeerd op een locatie, …

- Ruimtelijke gebeurtenissen, zoals het bezoeken van enkele vooraf gedefinieerde plaatsen of het bezoeken van plaatsen voor een tweede keer, …
- Tijd-ruimtelijke gebeurtenissen, zoals plotse versnellingen, omkeren in een straat, maar ook sequenties van de vorm

  *Bezoek(x, plein) -> plotse_stop(x) -> keer_om(x)*

  die ruimtelijke gebeurtenissen (plein bezoeken) mengt met eenvoudige acties (stoppen en omkeren). Gebeurtenissen kunnen ook tijd-ruimtelijke predicaten bevatten die toelaten om een tijd-ruimtelijke topologie uit te drukken tussen ruimtelijke gebieden en trajecten met een ruimtelijke onzekerheid (meer bepaald locaties zijn geen punten maar kleine gebieden die de echte positie bevatten). Een voorbeeld van een dergelijk predicaat is soms_zeker_in(*x,A*), wat wil zeggen dat er op minstens één ogenblik zodat object *x* zich in gebied *A* bevindt rekening houdend met onzekerheid. Bijgevolg is het mogelijk om rules te minen van de vorm

  *soms_zeker_in(x,ziekenhuis) => altijd_mogelijk_in(x,centrum)*

  wat wil zeggen mensen die zeker een ziekenhuis bezoeken ten minste één keer gewoonlijk nooit de stad verlaten.

Deze methode kan flexibeler gemaakt worden door de volledige informatie over tijd toe te voegen aan de afgeleide kenmerken. Dit komt overeen met het time-stampen van alle tijd-ruimtelijke gebeurtenissen en het afleiden van dynamische attributen. Deze time-stamps laten toe dat er meer gedetailleerde patronen afgeleid worden die ook de relaties op het vlak van tijd tussen de verschillende gebeurtenissen beschrijven.

### 1.3.2 Occurence retrieval

In tegenstelling tot het afleiden van frequente patronen uit de data kan de gebruiker zoeken naar voorkomens van een bepaald patroon. Dit wordt *occurence retrieval* of de inverse query genoemd. Er kunnen twee soorten queries onderscheiden worden. Elementaire queries handelen over het bewegingsgedrag van enkele objecten. Synoptische queries daarentegen handelen over het bewegingsgedrag van groepen van objecten.

Inverse elementaire queries brengen patronen met zich mee die beantwoord kunnen worden vanuit één traject. Bijvoorbeeld de query

*Zoek alle trajecten die door locatie A gaan tussen tijdstip $t_1$ en $t_2$*

kan verscheidene trajecten opleveren. Maar elk traject afzonderlijk volstaat om te beslissen of voldaan wordt aan het patroon of niet. In bovenstaande query is de plaats expliciet gespecificeerd terwijl de tijdsbeperking overeenstemt met een *range* query. Merk op dat dit patroon geen sequentiële informatie bevat. Om sequentiële informatie toe te voegen aan de query, kan er gevraagd worden dat na locatie *A* ook locatie *B* bezocht moet worden. Zulk een query wordt een tijd-ruimtelijke patroon query genoemd (STP – spatiotemporal pattern query) en wordt gedefinieerd als een opeenvolging van ruimtelijke predicaten met of exacte of relatieve tijdsorde.

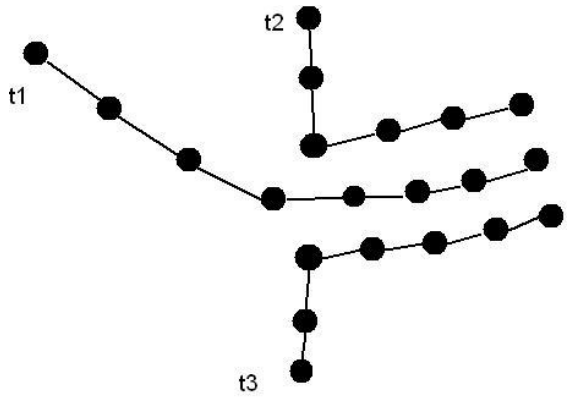
Inverse synoptische queries zoeken objecten die voldoen aan een gespecificeerd collectief gedrag. Deze patronen beogen gelijktijdige bewegingen en interacties tussen objecten te vatten. Ze worden ook wel groep patronen genoemd en kunnen informatie bevatten die afgeleid is van een groep van objecten. Intuïtief wordt een groep gevormd door een aantal objecten die dicht bij elkaar blijven in de ruimte voor een zinvolle periode. Nabijheid wordt gedefinieerd door een maximale afstand tussen elk paar van objecten. Wanneer *k* objecten dicht bij elkaar blijven in een gegeven tijdsperiode, vormen ze een *k*-groep patroon. Los van de algemene definitie van nabijheid van tijd-ruimtelijke objecten kan een groep gedefinieerd worden aan de hand van kenmerken van zijn innerlijke structuur. Voorbeelden van zulke patronen zijn het *flock*-patroon, het *leadership*-patroon, *convergentie*-patroon en het *encounter*-patroon.

### *1.3.2.1 Het flock-patroon*

Een aantal bewegende objecten voldoen aan het *flock*-patroon als ze in dezelfde richting bewegen en ze dichtbij elkaar liggen, meer bepaald in een cirkel met vooraf vastgestelde straal *r*.

Definitie:

> Gegeven $m > 1$ een natuurlijk getal en $r > 0$ een reëel getal, dan is een verzameling van ten minste *m* bewegende punten een *flock* als de bewegende punten zich binnen een cirkel met straal *r* van elkaar bevinden en indien ze in dezelfde richting bewegen.




**Figuur 1.3 Een flock-patroon**

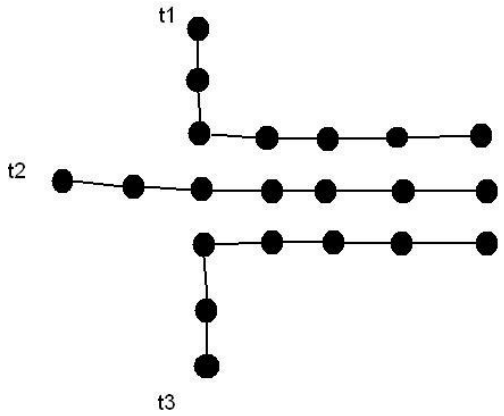
### *1.3.2.2 Het leadership-patroon*

Een volgend patroon dat besproken wordt lijkt heel sterk op het *flock*-patroon. Het enige verschil is dat er nu al één object aan het bewegen was in de richting dat de flock beweegt. Een groep bewegende objecten voldoet aan het *leadership*-patroon als ze in dezelfde richting bewegen, als ze dichtbij elkaar liggen en als één van de objecten, de leader, reeds een aantal stappen in de richting van de groep aan het bewegen was.

Definitie:

> Gegeven $m > 1$ en $\tau > 0$ twee natuurlijke getallen en $r > 0$ een reëel getal, dan is een verzameling van ten minste *m* bewegende punten een *leadership*-patroon als de bewegende objecten zich binnen een cirkel met straal *r* van elkaar bevinden, allemaal in dezelfde

richting bewegen en er één van de bewegende objecten reeds minstens $\tau$ tijdstippen in dezelfde richting aan het bewegen was.



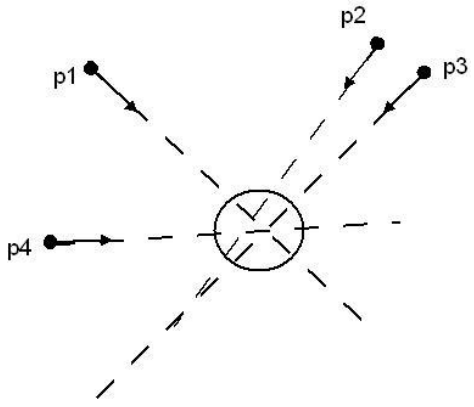

Figuur 1.4 Een leadership-patroon

### 1.3.2.3 Het convergentie-patroon

Het derde patroon is het *convergentie*-patroon. Een groep bewegende objecten voldoet aan het *convergentie*-patroon indien ze zich naar dezelfde locatie begeven, meer bepaald naar een cirkel met straal *r*. Het patroon wordt op elk tijdstip gezocht, er wordt rekening gehouden met de huidige richting waarin het punt zich beweegt. Het wil dus niet zeggen dat de punten die voldoen aan het *convergentie*-patroon ook effectief op die locatie komen.

Definitie:

Gegeven $m > 1$ een natuurlijk getal en $r > 0$ een reëel getal, dan is een *convergentie*-patroon een verzameling van minstens *m* bewegende objecten die door een cirkel met straal *r* gaan bewegen indien ze in dezelfde richting zouden blijven bewegen.




Figuur 1.5 Een convergentie-patroon

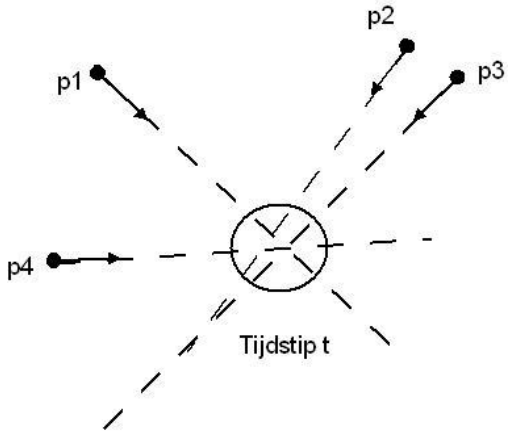
### 1.3.2.4 Het encounter-patroon

Het laatste patroon is het *encounter*-patroon. Dit patroon is een uitbreiding van het *convergentie*-patroon en houdt in dat een aantal bewegende punten convergeren en op hetzelfde tijdstip op dezelfde plaats zullen zijn. Hetzelfde principe als bij het *convergentie*-patroon wordt hierbij in acht

genomen, namelijk dat dit patroon alleen geldt als de bewegende objecten hun richting en snelheid behouden die ze op het moment van het ontdekken van het patroon hebben.

Definitie:

> Gegeven $m > 1$ een natuurlijk getal en $r > 0$ een reëel getal, dan is een groep van minstens $m$ bewegende objecten een *encounter*-patroon als de bewegende objecten zich op hetzelfde tijdstip binnen een cirkel met straal $r$ van elkaar zullen bevinden in de veronderstelling dat ze hun richting en snelheid behouden.




Figuur 1.6 Een Encounter-patroon

## 1.4 Voorspelling

Een betrouwbare voorspelling over de toekomstige positie of bestemming van een object is zeker geen overbodige luxe in de hedendaagse samenleving. Met real-time verkeersmanagement, GPS navigatie, just-in-time logistiek, … bestaan er een aantal toepassingen voor welke dit zeer nuttig zou zijn (zie hoofdstuk 2.4 over recente toepassingen). Het anticiperen van de beweging van objecten of groepen van objecten laat deze systemen toe acties te nemen in het geval van een vertraging, om op het geschikte ogenblik informatie te verstrekken, etc. Tijd-ruimtelijke data geeft een breed perspectief voor voorspellingen, zoals voorspellingen van locaties en trajecten, voorspellingen in verband met dichtheid, bereik en gebeurtenissen en zelfs de classificatie van trajecten.

Een belangrijk aspect is het voorspellen van de route en bestemming die een bewegend object naar alle waarschijnlijkheid zal kiezen. Bijvoorbeeld location-based diensten kunnen meer gesofisticeerde diensten aanbieden wanneer ze weten langs welke plaatsen een gebruiker zal passeren en of de gebruiker op weg is naar zijn werk of naar de supermarkt. Het algemene idee achter het voorspellen van routes en bestemmingen is het feit dat mensen dagelijkse routines volgen. Ze komen slechts op een beperkt aantal plaatsen frequent, zoals hun thuis, hun werk, hun sportclub, enz. Bijgevolg kiezen mensen hun huidige route uit een kleine verzameling van routes.

Voorspelling van dichtheid geeft vele voordelen. De object-dichtheid van een bepaald gebied is gedefinieerd als het aantal objecten binnen dit gebied op een bepaald tijdstip in verhouding met de grootte van dit gebied. Het is van toepassing op alle objecten en varieert over de tijd. Toegepast op het verkeer kent dit een groot voordeel. Een verkeerscontrole systeem dat kan voorspellen waar er

gebieden gaan ontstaan met een te hoge dichtheid kan hier op reageren en ervoor zorgen dat er geen of beperkte files ontstaan.

## 1.5 Semantisch minen

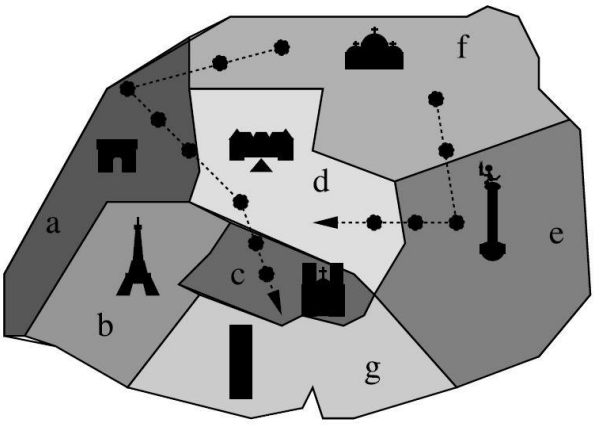
Meestal worden trajecten opgeslagen als discrete punten en wordt er geen semantische informatie eigen aan het traject opgeslagen. Deze semantische informatie is nochtans van fundamenteel belang bij het interpreteren van deze trajecten. Door aan trajecten semantische geografische informatie toe te voegen is het mogelijk om queries, analyse van de trajecten en het minen van de bewegende objecten te vereenvoudigen [22][23].

Trajecten zijn meestal beschikbaar als reeksen van discrete punten van de vorm (*tid, x, y, t*), met *tid* een ID voor elk bewegend object, *x* en *y* ruimtelijke coördinaten en *t* een timestamp. De integratie van semantische geografische informatie in trajectdata is de belangrijkste stap voor analyse van trajectdata in reële toepassingen. Door deze semantische informatie te integreren in trajectdata kunnen er queries beantwoord worden die anders niet beantwoord zouden kunnen worden. Voorbeelden van zulke queries zijn:

- Welke plaatsen worden het meest bezocht door mensen die een conferentie bijwonen in een toeristische stad?
- Welke zijn de belangrijkste sequenties van plaatsen bezocht in de ochtend?
- Op welke plaatsen blijven bewegende objecten een bepaalde hoeveelheid tijd stilstaan?

Deze integratie kan leiden tot de ontdekking van semantische traject patronen die de meeste data mining technieken die trajecten beschouwen als reeksen van punten (*tid, x, y, t*) niet kunnen ontdekken.

Om semantische informatie toe te voegen aan trajecten, worden deze punten geprojecteerd op een verzameling van zones die het gebied dat beschouwd wordt opdeelt. Deze opdeling is gerelateerd aan een specifieke thematische interpretatie van het gebied. Elke zone van het gebied heeft een uniek label. In de onderstaande figuur wordt een gebied voorgesteld dat opgedeeld is in een aantal zones. Elk van deze zones is gelabeld met een symbool (a, b, c, …). Over dit gebied worden de trajecten van bewegende objecten beschreven. Deze zijn op de figuur weergegeven aan de hand van de stippellijnen.




Figuur 1.7 Gebied opgedeeld in zones

Beschouw nu de volgende queries:

- Geef alle objecten die van *a* naar *f* gereisd zijn, meer dan 10 minuten in *f* verbleven hebben en daarna van *f* naar *c* gereisd zijn.
- Geef alle objecten die van *f* naar *d* of *c* gereisd zijn door een andere derde zone van het gebied.
- Geef alle objecten die een gegeven zone verlaten hebben, naar *c* reisden en daarna terugkeerden naar de eerste zone.

Hetgeen al deze voorbeelden gemeen hebben is dat er een opeenvolging van zones gespecificeerd is waartoe een object behoort tijdens het reizen, samen met beperkingen op het gebied van tijd. Deze specificatie worden ook wel *mobiliteitspatronen* geheten.

De trajecten bestaan uit een reeks discrete punten. Voor elk punt wordt de zone berekend waarin dit punt ligt. Het is daarom vrij eenvoudig om het traject van een bewegend object voor te stellen als een discrete reeks van de vorm $l_1\{t_1\}$. $l_2\{t_2\}$… $l_n\{t_n\}$ met $l_1$, $l_2$, … $l_n$ de labels van de zones en $t_1$ … $t_n$ de tijd gespendeerd in elke zone. Bijvoorbeeld in figuur 1.7 wordt het linkertraject, wanneer aangenomen wordt dat er 2 minuten in *f* verbleven werd, 4 minuten in *a*, 3 minuten in *d* en 6 minuten in *c*, voorgesteld door f{2}.a{4}.d{3}.c{6}. Merk op dat telkens een punt uit het oorspronkelijke traject beschouwd wordt, dat oftewel de tijdscomponent van het laatste label wordt verhoogd als het object in dezelfde zone blijft, oftewel een nieuw label toegevoegd wordt aan de reeks.

In wat volgt van deze sectie worden enkele formele definities gegeven van begrippen die hun toepassing hebben in het semantisch minen.

Definitie: Een sample traject is een lijst van tijd-ruimtelijke punten $[(x_0,y_0,t_0),\ldots,(x_N,y_N,t_N)]$, met $x_i, y_i, t_i \in \mathrm{R}$ voor $\mathrm{i} = 0,\ldots,\mathrm{N}$ en $t_0 < t_1 < t_N$.

Definitie: Een kandidaat stop C is een tupel $(R_C, \Delta_C)$, met $R_C$ een veelhoek in $R^2$ en $\Delta_C$ een strikt positief getal. De verzameling $R_C$ wordt de geometrie genoemd van de kandidaat stop en $\Delta_C$ is de minimale tijdsduur van de kandidaat stop.

Definitie: Een applicatie *A* is een eindige verzameling $\{C_1 = (R_{C1}, \Delta_{C1}),\ldots, C_N = (R_{CN}, \Delta_{CN})\}$ van kandidaat stops met niet overlappende veelhoeken $R_{C1},\ldots, R_{CN}$

Als een kandidaat stop een punt is of een polylijn, wordt er een polygonale buffer gegenereerd rond dit object. Zodoende wordt deze kandidaat stop voorgesteld door een veelhoek in de applicatie om ruimtelijke onzekerheid te vermijden.

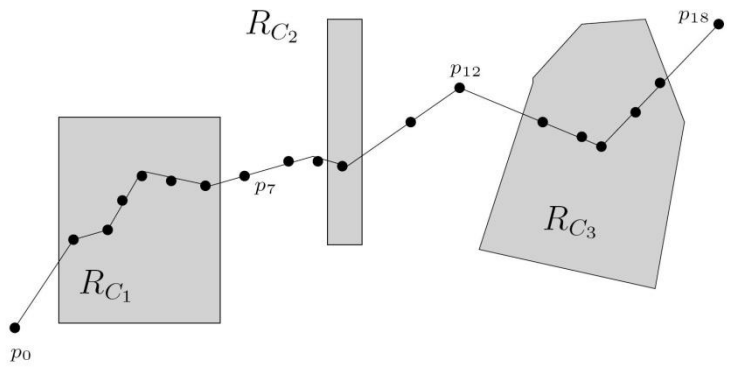
Definitie: Gegeven *T* een traject en gegeven $A = (\{C_1 = (R_{C1}, \Delta_{C1}),\ldots, C_N = (R_{CN}, \Delta_{CN})\})$ een applicatie. Veronderstel $[(x_i,y_i,t_i), (x_{i+1},y_{i+1},t_{i+1}),\ldots, (x_{i+l},y_{i+l},t_{i+l})]$ een subtraject van *T*, met een $(R_{Ck}, \Delta_{Ck})$ in *A* zodat $\forall j \in [i, i+l] : (x_j, y_j) \in R_{Ck}$ en $|t_{i+l} - t_i| \geq \Delta_{Ck}$, en dit subtraject is maximaal (rekening houdend met deze twee voorwaardes), dan wordt een tupel $(R_{Ck}, t_i, t_{i+l})$ gedefinieerd als een stop van *T* in verhouding tot *A*.

Een move van *T* in verhouding tot *A* is één van de volgende gevallen:

- een maximaal opeenvolgend subtraject van *T* tussen twee tijdelijke opeenvolgende stops van *T*;
- een maximaal opeenvolgend subtraject van *T* tussen het eerste punt van *T* en de eerste stop van *T*;
- een maximaal opeenvolgend subtraject van *T* tussen de laatste stop van *T* en het laatste punt van *T*;
- het traject *T* zelf als *T* geen stops heeft.

Wanneer een move begint in een stop, begint deze move in het laatste punt van het subtraject dat deze stop kruist. Analoog, als een move eindigt in een stop, dan eindigt deze move in het eerste punt van het subtraject dat deze stop kruist.

De concepten die hierboven gedefinieerd staan worden geïllustreerd door volgende figuur.




**Figuur 2.8 Applicatie met 3 kandidaat stops**

In figuur 1.8 zijn er drie kandidaat stops met veelhoeken $R_{C1}$, $R_{C2}$, en $R_{C3}$. Stel dat de ruimtelijke projectie van het traject *T* doorlopen wordt van links naar rechts en $t_0,\ldots,t_{18}$ zijn de tijdspunten van *T*. In het begin ligt *T* niet in een kandidaat stop dus wordt er begonnen met een move. Vervolgens gaat het traject *T* $R_{C1}$ binnen op tijdspunt $t_1$. Omdat de tijdsduur van het verblijf binnen $R_{C1}$ groot genoeg is, is ($R_{C1}$, $t_1$, $t_6$) de eerste stop van *T*. Vervolgens gaat *T* $R_{C2}$ binnen, maar voor een tijdsinterval korter als $\Delta_{C2}$, dus dit is geen stop. Daarom is er een move tot *T* $R_{C3}$ binnengaat, omdat dit voldoet aan de eisen voor een stop. Het tupel ($R_{C3}$, $t_{13}$, $t_{17}$) wordt dan de tweede stop. Ten slotte eindigt het traject *T* met een move.

In de definities zijn stops interessante ruimtelijke locaties, ook wel ruimtelijke kenmerken genoemd, gespecificeerd in overeenstemming met de applicatie. Zo zullen verkeerslichten bijvoorbeeld beschouwd worden als stops in een transport management applicatie, maar waarschijnlijk niet in een toeristische applicatie. Ruimtelijke kenmerken worden meestal opgeslagen in verschillende bestanden (bijvoorbeeld shape files) of in verschillende relaties (bijvoorbeeld restaurant, luchthaven) in geografische databases. Daarom is het mogelijk om trajectory sample punten te joinen met belangrijke ruimtelijke kenmerken met als doel stops en moves te vinden.

Een algoritme met als doel stops en moves te vinden is SMoT (Stops and Moves of Trajectories). Het algoritme gaat voor elk punt van een traject T na of het in een veelhoek van een kandidaat stop $R_C$ ligt. Als dit het geval is, gaat het algoritme na of de tijdsduur van het verblijf in de kandidaat stop minimaal gelijk is aan een gegeven drempel $\Delta_C$. Als dit het geval is, wordt de gekruiste kandidaat

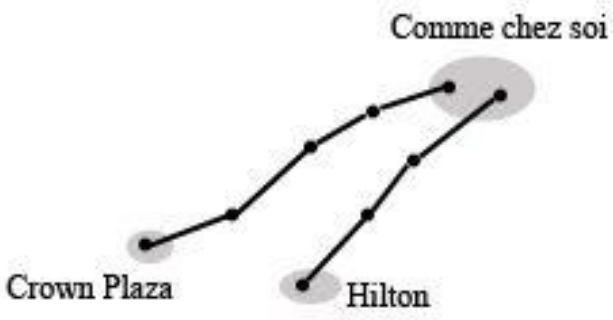
stop beschouwd als een stop en deze stop wordt opgeslagen. Een move wordt opgeslagen tussen de vorige stop en de laatste. Wanneer de laatste stop de eerste stop is, bestaat er geen vorige stop en is deze dus *null*. Wanneer een move wordt toegevoegd aan de verzameling van moves worden de tijd-ruimtelijke kenmerken van deze move ook toegevoegd. In tegenstelling tot stops die liggen in een ruimtelijk kenmerk dat een geometrie heeft, wordt er geen intersectie van de geometrie van de move met ruimtelijke kenmerken uitgevoerd omdat moves niet beschouwd worden als belangrijke onderdelen van een traject. Maar voor sommige applicaties kan het interessant zijn om de ruimtelijke kenmerken te kennen die gekruist worden door een traject. Daarom worden de geometrie en de timestamp van de move bijgehouden voor verdere analyse. De output van SMoT is een semantisch traject dataset, en daarom kunnen er verschillende semantische traject analyses uitgevoerd worden.

De integratie van trajecten met semantische geografische informatie, die de meest belangrijke plaatsen kenmerkt in overeenstemming met de applicatie, reduceert de complexiteit van de query en vergemakkelijkt de traject data analyse. De stops en moves worden slechts één keer berekend in een preprocessing stap. Daardoor wordt het ruimtelijke gebied waarin gezocht wordt en de ruimtelijke joins in de formulering van de query geminimaliseerd in verhouding tot het model met sample punten. Door de decompositie van trajecten in stops en moves wordt er directe toegang verkregen tot de semantische traject informatie.

Tot slot wordt een voorbeeld gegeven dat illustreert hoe extra informatie gemined kan worden door gebruik te maken van semantische informatie. Beschouw het volgende voorbeeld: twee trajecten komen vanuit een verschillende locatie samen op een volgende locatie. Als er niet geweten is wat er zich op die locaties bevindt, worden er geen rules gegenereerd. Maar wanneer er geweten is dat beide trajecten beginnen in een hotel en samenkomen in een restaurant, dan zou bijvoorbeeld de volgende rule afgeleid kunnen worden:

*(Hotel, 11u30) -> (restaurant, 12u00)*

Dit wordt grafisch voorgesteld in figuur 1.9.




Figuur 1.9 Stops en moves

## 1.6 T-patronen

### 1.6.1 Inleiding

Traject patronen of T-patronen [30] zijn beknopte beschrijvingen van frequente gedragingen in termen van zowel ruimte (meer bepaald de bezochte gebieden) als tijd (meer bepaald de duur van bewegingen). Een traject patroon stelt een verzameling van individuele trajecten voor die de eigenschap delen dat ze dezelfde opeenvolging van plaatsen bezoeken met gelijkaardige reistijden. Daarom zijn er twee begrippen van groot belang:

- regions of interest (RoI) in het gegeven gebied
- de typische reistijd van bewegende objecten van plaats tot plaats

In feite is een traject patroon een opeenvolging van gebieden die frequent bezocht blijken te zijn in de volgorde dat ze voorkomen in de sequentie. De overgang tussen twee opeenvolgende plaatsen wordt gekenmerkt door een typische reistijd die gehaald wordt uit de input trajecten. Beschouw bijvoorbeeld volgende twee traject patronen over interessante gebieden in het centrum van een stad:

*Station ->(15 min) Grote Markt -> (2u 15 min) Museum*

*Station ->(10 min) voetgangerstunnel -> (10 min) Universiteit*

Het eerste patroon kan geïnterpreteerd worden als het typische gedrag van toeristen die snel vanuit het station naar een belangrijke plaats gaan om daar ongeveer twee uur te spenderen alvorens naar een dichtbijgelegen museum te gaan. Het tweede patroon daarentegen kan de stroom studenten voorstellen die te voet van het station naar de universiteit gaan. Voor hen is de voetgangerstunnel een verplichte passage. Er moet opgemerkt worden dat een traject patroon geen informatie geeft over de route tussen twee opeenvolgende plaatsen. In plaats daarvan wordt de reistijd gespecificeerd die de reistijd van elk individueel traject, voorgesteld door het traject patroon, benadert. Een tweede belangrijke opmerking is dat de individuele trajecten die gegroepeerd zijn in het traject patroon niet noodzakelijk tegelijkertijd zijn afgelegd. Het enige wat geëist wordt is dat deze trajecten dezelfde opeenvolging van plaatsen bezoeken met gelijkaardige reistijden.

### 1.6.2 Definitie

Een traject van een object is een reeks van met een tijdstip geassocieerde locaties die de sporen voorstellen verzameld door een mobiele infrastructuur, zoals het GSM netwerk, of GPS gegevens opgenomen door draagbare toestellen en doorgezonden naar een centrale server. De plaats, zoals een GSM cel of een lengtegraad/breedtegraad paar, wordt geabstraheerd gebruik makend van Carthesische coördinaten. Dit is formeel weergegeven door volgende definitie:

Definitie: Een tijd-ruimtelijke reeks (ST-reeks) of traject is een reeks van tripels $S = [(x_0,y_0,t_0),...,(x_k,y_k,t_k)]$, met $t_i$ ($i = 0...k$) een tijdstip, $\forall_{0 \leq i < k}\ t_i < t_{i+1}$ en $(x_i,y_i)$ zijn punten in $R^2$.

De belangrijkste stap in het gaan van reeksen naar tijd-ruimtelijke reeksen bestaat uit het vervangen van de discrete elementen die elke reeks vormen met ruimtelijke plaatsen. Daarom focust het minen van tijd-ruimtelijke reeksen zich op de relaties tussen plaatsen, terwijl minen op gewone reeksen zich eerder focust op relaties tussen gegeven gebeurtenissen.

De belangrijkste taak in het minen van reeksen bestaat uit het tellen van het aantal keer dat een patroon voorkomt, meer bepaald de delen van de input die overeenkomen met een mogelijk patroon. Om te controleren of een reeks voldoet aan een patroon, moeten we controleren of de elementen van de reeks, meer bepaald de locaties, overeenkomen. Hiervoor definiëren we een nabijheidsfunctie $N : R^2 \rightarrow P(R^2)$, die aan elk paar (x,y) een verzameling N(x,y) van naburige punten toekent.

Definitie: Gegeven een reeks van ruimtelijke punten $S = [(x_0,y_0),...,(x_k,y_k)]$, een tijd-ruimtelijke reeks $T = [(x'_0,y'_0,t'_0),...,(x'_n,y'_n,t'_n)]$ en een nabijheidsfunctie $N : R^2 \rightarrow P(R^2)$. We zeggen dat S bevat is in T als en slechts als er een reeks getallen bestaat $0 \leq i_0 < ... < i_k \leq n$ zodat $\forall_{0 \leq j \leq k} (x_j,y_j) \in N(x'_{ij},y'_{ij})$.

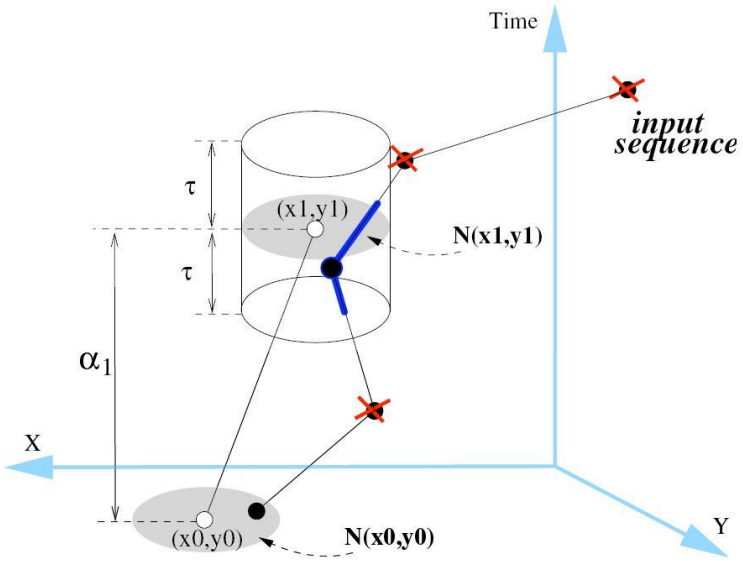
Definitie: Een traject patroon (T-patroon) is een paar (S,A) met $S = [(x_0,y_0),...,(x_k,y_k)]$ een reeks punten in $R^2$, en $A = [a_1,...,a_k] \in R^k_+$ is de (tijds-) annotatie van de reeks. T-patronen zullen ook voorgesteld worden als $(S,A) = (x_0,y_0) \rightarrow a_1 (x_1,y_1) \rightarrow a_2 ... \rightarrow a_k (x_k,y_k)$.

Een T-patroon komt voor wanneer de ruimtelijke posities en de tijden van de overgangen van het patroon overeenkomen met degene van een input reeks.

Definitie: Gegeven een tijd-ruimtelijke reeks T, een tijdsgrens τ, een nabijheidsfunctie $N : R^2 \rightarrow P(R^2)$ en een T-patroon $(S,A) = (x_0,y_0) \rightarrow a_1 (x_1,y_1) \rightarrow a_2 ... \rightarrow a_k (x_k,y_k)$. We zeggen dat (S,A) bevat zit in T als en slechts als er een deelreeks T' van T bestaat met $T' = [(x'_0,y'_0,t'_0),...,(x'_k,y'_k,t'_k)]$ zodat:

- S is bevat in T'
- $\forall_{1 \leq j \leq k} |a_j - a'_j| \leq \tau$ met $a'_j = t'_j - t'_{j-1}$

Intuïtief kan men inzien dat een T-patroon bevat is in een traject als dit traject een instantie bevat die het T-patroon benadert. Er moet opgemerkt worden dat vergelijkingen niet gemaakt worden met absolute tijdstippen, maar dat ze gebaseerd zijn op de overgangstijden tussen twee opeenvolgende locaties van een reeks.




Figuur 1.10

In bovenstaande figuur is te zien hoe te ruimtelijke en tijdsbeperkingen een tijd-ruimtelijke omgeving vormen rond elk punt van het traject. Meer nog, er kan opgemerkt worden dat de omgeving N() van een punt alleen afhangt van de ruimtelijke coördinaten van de punten, en dus zijn omgevingen tijdsonafhankelijk.

### 1.6.3 Regions of interest

T-patronen kunnen op verschillende manieren benaderd worden. Elk van die manieren komt overeen met een verschillende nabijheidsfunctie N(x,y). Het kiezen van een nabijheidsfunctie wil in essentie zeggen dat er een specifieke notie van ruimtelijke gelijksoortigheid moet geïmplementeerd worden die gebruikt zal worden wanneer men wil nagaan of een traject een T-patroon bevat. De nabijheidsfunctie kan gebruikt worden voor het modelleren van Regions of Interest (ROI), die een natuurlijke manier voorstellen om te ruimte op te delen in betekenisvolle gebieden.

Wanneer als input een set R van disjuncte ruimtelijke gebieden gegeven wordt, kan de nabijheidsfunctie als volgt gedefinieerd worden:

$N_R(x,y)$ =

- A als $A \in R$ en $(x,y) \in A$
- leeg anders

De omgeving van een ruimtelijk punt is het hele gebied waarin het ligt, meer bepaald zijn twee punten gelijkaardig als ze in hetzelfde gebied liggen. Alle punten die in geen enkel gebied vallen hebben een lege omgeving, wat wil zeggen dat ze niet gelijkaardig zijn aan een ander punt. Het resultaat is dat deze punten virtueel verwijderd worden uit trajecten en T-patronen.

Het mining probleem heeft in verscheidene contexten een voorafgaande kennis van passende RoI, manueel door domeinexperts toegevoegd of eenvoudig door gezond verstand. Maar in sommige gevallen is deze informatie niet op voorhand aanwezig en dus moet deze informatie op een of andere manier afgeleid worden. Ze kunnen bijvoorbeeld automatisch berekend worden door gebruik te maken van heuristieken. Deze benadering is vrij gelijk aan de vorige, met het verschil dat gebieden automatisch afgeleid worden van echte data in plaats van op voorhand gedefinieerd te zijn. Het onderliggende idee is dat plaatsen die regelmatig bezocht worden waarschijnlijk interessante plaatsen voorstellen, en plaatsen die weinig bezocht worden waarschijnlijk oninteressante plaatsen zijn. Zo zullen toeristen in een bepaalde stad vaak dezelfde plaatsen bezoeken, maar de route die ze nemen zal verschillen.

#### *1.6.3.1 Traject prepocessing*

Wanneer er vanuit gegaan wordt dat een passende reeks RoI gekend zijn, bestaat het toepassen van deze op het T-patroon mining probleem eenvoudig uit het prepocessen van de input reeksen naar de overeenkomstige reeksen van RoI.

Het maken van veronderstellingen over de beweging van objecten uit de geobserveerde punten betekent dat er een model moet voorzien worden voor zulk een beweging, voor dewelke er een brede waaier aan mogelijkheden beschikbaar zijn in de tijd-ruimtelijke literatuur, zoals bijvoorbeeld lineaire regressie, Beziers curves, probabilistische modellen, etc. Een van de meest gebruikte en eenvoudigste modellen is lineaire regressie, dewelke uit gaat van een constante snelheid en een constante richting tussen elk paar van twee opeenvolgende punten. Wanneer de volledige beweging

van objecten gereconstrueerd worden, blijft een object in het algemeen in een gebied A voor een tijdsduur I, in plaats van een enkele instantie t. Daarom is het niet voor de hand liggend welke timestamp geassocieerd moet worden met de gebeurtenis "Gebied A" in de vertaalde reeks. De oplossing voor dit probleem bestaat erin om de timestamp op de volgende manier te kiezen:

- als het traject begint op tijd t uit een punt dat al in gebied A ligt, dan wordt het koppel (A,t) aangemaakt
- in alle andere gevallen wordt het tijdstip genomen van de trajecten wanneer ze een gebied binnengaan, en wordt dit geassocieerd met de naam van het gebied. Een object kan verschillende keren eenzelfde gebied bezoeken, en elke gebeurtenis zal een andere timestamp krijgen.

Meer geavanceerde oplossingen kunnen de tijdstippen beschouwen wanneer een traject een gebied verlaat, of beide door twee verschillende gebeurtenissen te creëren die staan voor het binnenkomen en het verlaten van een gebied.

#### *1.6.3.2 Ontdekken van RoI*

Wanneer RoI niet op voorhand gekend zijn, kunnen enkele heuristieken gebruikt worden die in staat zijn om automatisch RoI te identificeren. Er zijn verschillende methodes mogelijk:

- het selecteren van RoI uit een database van kandidaat-gebieden (bijvoorbeeld een GIS dat alle restaurants, winkels, etc. bevat) door het toepassen van enkele criteria (bijvoorbeeld alle restaurants dicht bij een snelweg)
- het automatisch berekenen van kandidaat-gebieden door het analyseren van trajecten, bijvoorbeeld door het selecteren van alle minimale vierkante gebieden die bezocht werden door ten minste 10% van de objecten
- een combinatie van de vorige twee methodes, bijvoorbeeld door het selecteren van alle kruispunten waar meer als 50% van de trajecten die er passeren van richting veranderen

## 1.7 Conclusie

Het onderzoek naar de mining technieken op tijd-ruimtelijke data en op data van trajecten in het bijzonder is nog zeer jong en er wordt nog volop onderzoek naar verricht. Om deze technieken te kunnen toepassen hebben we natuurlijk data nodig. In het volgende hoofdstuk wordt er dieper ingegaan op data waarop mining technieken kunnen worden toegepast.

# Hoofdstuk 2: Data collectie

## 2.1 Inleiding

Vooraleer er gemined kan worden op trajecten is er data nodig die trajecten voorstelt. Deze data dient verzameld te worden. Momenteel wordt er gebruik gemaakt van GPS-data. Maar deze om deze data te verzamelen dient medewerking gevraagd te worden aan mensen met een GPS-toestel. Het zou handig zijn als we met behulp van GSM-signalen trajecten kunnen opstellen. GSM's zenden immers signalen uit waardoor er veel meer data beschikbaar is.

## 2.2 GPS

[17-19] GPS is een afkorting voor Global Positioning System en is een systeem dat ontworpen is door het Amerikaanse leger. Het systeem maakt gebruik van verschillende satellieten die in een vaste baan rond de aarde cirkelen. Via deze satellieten is het mogelijk om met een GPS-ontvanger, tot op enkele meters nauwkeurig na, de plaats te bepalen op aarde waar de GPS-ontvanger zich bevindt. Alle PND's (Personal Navigation Devices) die tegenwoordig op de markt zijn maken gebruik van GPS. De PND's zenden zelf geen signaal uit, maar zij maken zelf gebruik van de signalen die zij van de satellieten opvangen. Om tot een correcte plaatsbepaling te komen moeten zij van minstens vier satellieten een signaal ontvangen. De Europese tegenhanger van GPS, Galileo, is nog in volle ontwikkeling. Het biedt een aantal voordelen ten opzichte van GPS, waarvan het belangrijkste voordeel is dat men tijdens een eventuele oorlog onafhankelijk is van de Amerikaanse beperkingen. Ook zal met Galileo zenden mogelijk worden in plaats van enkel ontvangen.

Er zijn verschillende nadelen gekoppeld aan het gebruik van GPS. Daarom is het misschien nuttig als er gekeken wordt naar GSM. Zo beschikt in de huidige maatschappij nog lang niet iedereen over een Personal Navigation Device, maar beschikt bijna iedereen over een mobiele telefoon. Een tweede voordeel is dat met GSM zenden ook mogelijk is. Het is dus mogelijk om na te gaan waar een mobiele telefoon zich bevindt, terwijl het niet mogelijk is om na te gaan waar een GPS-ontvanger zich bevindt. Wel moet er opgepast worden dat de wetten op privacy niet geschonden worden, met andere woorden de anonimiteit moet gegarandeerd zijn. Bij GPS weten de gebruikers immers dat hun data gebruikt zal worden en stemmen ze hiermee in, maar bij GSM is dit niet het geval. Omdat dit toch wel zeer belangrijk is wordt hier later uitvoerig op ingegaan.

## 2.3 GSM

[10-16,24]

### 2.3.1 Geschiedenis

In het begin van de jaren 80 kende mobiele telefonie een sterke opgang in Europa. Omdat bijna ieder land een eigen systeem ontwikkelde onafhankelijk van de andere landen, waren al deze systemen niet compatibel met elkaar. Dit zorgde voor een aantal problemen. Zo kon men ondermeer het mobiele netwerk niet uitbreiden over de grenzen heen, wat men toch prioritair vond in een Europa waarin de grenzen een steeds kleinere rol gingen spelen. Hierdoor ontstond er een verlangen naar een uniform globaal mobiel netwerk. Men richtte een onderzoeksgroep op, de Groupe Spécial Mobile (GSM), om onderzoek te voeren naar een dergelijk mobiel netwerk. Dit mobiel netwerk moest voldoen aan een aantal voorwaarden:

- een goede spraakkwaliteit;

- een laag kostenniveau voor apparatuur en diensten;
- de ondersteuning van roaming;
- de mogelijkheid om draagbare terminals te ondersteunen;
- de ondersteuning van nieuwe diensten;
- een efficiënte spectrumtoewijzing;
- compatibiliteit met ISDN;

De onderzoeksgroep boekte vooruitgang en in 1991 werden de eerste telefoondiensten aangeboden.

### 2.3.2 Wat is GSM?

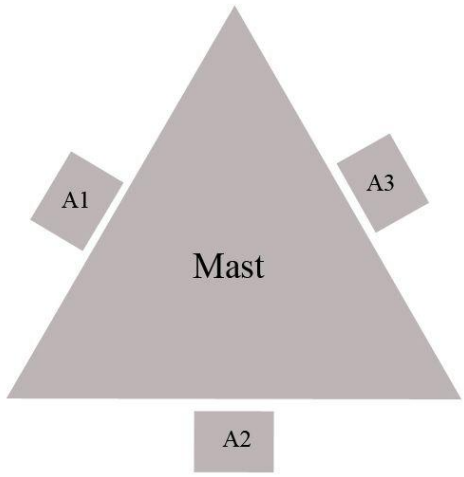
Tegenwoordig heeft GSM een andere betekenis dan de oorspronkelijke. In de huidige samenleving wordt met een GSM vaak een mobiele telefoon bedoeld. Maar eigenlijk is GSM, of Global System for Mobile communications, de belangrijkste standaard voor mobiele telefonie in de wereld. GSM behoort tot de tweede generatie (2G) van mobiele telefonie. Er is ook een derde generatie (3G) van mobiele telefonie. In deze generatie zijn nog meer diensten beschikbaar zoals video-conferencing. GSM verschilt van zijn voorgangers in het feit dat signalen en gesprekken digitaal doorgestuurd worden in plaats van analoog. GSM is zo ontworpen dat er slechts minimale bandbreedte gebruikt wordt om het gesprek door te sturen maar toch genoeg data zodat de stem herkenbaar blijft. Omdat de GSM-standaard wereldwijd gebruikt wordt, biedt het zowel aan de consument als aan de netwerk-operatoren voordelen. Zo kan de consument voordeel halen uit het feit dat hij gebruik kan maken van verschillende netwerken zonder te veranderen van mobiele telefoon, terwijl de netwerk-operatoren kunnen kiezen tussen de verschillende ontwikkelaars van technologie voor GSM.

De grootste concurrent van GSM is cdmaOne. Deze standaard wordt ondermeer in de Verenigde Staten, India, Brazilië, ... gebruikt. Maar GSM is de populairste van de twee; volgens GSMA, de vereniging van GSM operatoren, wordt GSM wereldwijd door meer dan twee miljard mensen gebruikt in meer dan 210 verschillende landen.

GSM heeft ook voor een primeur gezorgd door als eerste een goedkoop alternatief te bieden voor telefoongesprekken, namelijk SMS (Short Message Service). Zoals de naam al doet vermoeden is het via SMS mogelijk om korte tekstberichtjes te sturen of te ontvangen. SMS is zeer populair in Europa en Azië, maar minder populair in Amerika. Een SMS kan maximaal 120 tekens bevatten en enkel tekst. De opvolger van SMS is MMS (Multimedia Messaging Service). Via MMS is het mogelijk om ook afbeeldingen, muziek en zelfs videobestanden door te sturen. Wanneer een SMS/MMS-bericht verzonden wordt, wordt dit verzonden naar de SMS-centrale (SMSC). Hier wordt het bericht opgeslagen en wordt het doorgestuurd naar de recipiënt. Van zodra het bericht aangekomen is, wordt het door de SMS-centrale verwijderd.

GSM maakt gebruik van een netwerk dat onderverdeeld is in cellen. Deze cellen worden gevormd door een GSM-mast of BTS (Base Transceiver Station). Zoals duidelijk is op figuren 2.1 en 2.2 bevatten deze masten meestal drie antennes die samen een gebied van 360 graden bestrijken. De cellen variëren in grootte afhankelijk van de hoogte van de GSM-mast, de versterking van de GSM-mast en van de voortplantingsomstandigheden (vb. veel gebouwen = slechte voortplantingsomstandigheden). De straal van zulke cellen gaat van enkele honderden meters tot enkele tientallen kilometers. GSM ondersteunt tot een maximale afstand van 35 kilometer. Wanneer

een mobiele telefoon ingeschakeld wordt, gaat deze op zoek naar GSM-masten in zijn onmiddellijke omgeving. De mobiele telefoon verkrijgt een lijst van GSM-masten waar aangemeld kan worden. Bij het bepalen bij welke GSM-mast er aangemeld wordt, wordt er rekening gehouden met een aantal factoren. Vooraleerst wordt er nagegaan met welke operator het toegestaan is om te telefoneren of met welke operator er een roaming overeenkomst is. Hiernaast houdt men ook rekening met de signaalsterkte. Wanneer er twee operatoren gevonden worden waarmee het toegestaan is om te bellen, wordt er geopteerd voor degene met de grootste signaalsterkte. Vanaf deze GSM-mast gaat het gesprek naar de BSC (Base Station Controller), een centraal punt waaraan meerdere masten gekoppeld zijn. Vanuit het BSC gaat het GSM-verkeer naar een MSC (Mobile Switching Center). Van hieruit wordt er een verbinding gemaakt met de MSC van de tegenpartij.




Figuur 2.1 Schematische voorstelling van een GSM-mast

Figuur 2.2 Een GSM-mast

GSM-netwerken kunnen in vier verschillende frequenties werken: in de meeste landen wordt gebruik gemaakt van 900 MHz of 1800 MHz band, maar in de Verenigde Staten en nog enkele andere landen waren deze al in gebruik. 900 MHz was al toegewezen aan licentievrije communicatie en 1800 MHz wordt gebruikt door het Amerikaanse leger. Daarom maakt men er gebruik van 850 MHz en 1900 MHz band. Er zijn twee belangrijke verschillen tussen enerzijds 900 MHz en 850 MHz band en 1800 MHz en 1900 MHz band anderzijds. De lagere frequenties (850 en 900 MHz) zorgen voor een sterker signaal: het bereik is groter en de ontvangst in gebouwen is beter. De hoge frequenties (1800 en 1900 MHz) echter hebben een kleiner bereik en nemen sterker af in gebouwen, maar ze zorgen wel voor veel heldere klank.

### 2.3.3 GSM security

GSM is ontworpen met een slechts matig beveiligingsniveau. Mits enige moeite kan het gekraakt worden. Vooraleer een GSM-toestel gebruik kan maken van het GSM-netwerk moet er een authenticatie-procedure afgelegd worden. Tijdens deze procedure controleert het netwerk of het GSM-toestel dat van het netwerk gebruik wil maken wel degelijk het GSM-toestel is dat het beweert te zijn. Als de authenticatie-procedure een positief resultaat oplevert en dus het GSM-toestel echt het toestel is welk het beweert te zijn, krijgt dit toegang tot het GSM-netwerk. Als de authenticatie-procedure een negatief resultaat oplevert en dus het GSM-toestel niet hetgeen is welk het beweert te zijn, krijgt dit geen toegang tot het GSM-netwerk.

De authenticatie-procedure werkt als volgt: het netwerk genereert eerst een “vraag” voor een GSM-toestel. Het antwoord op deze vraag kan het GSM-toestel alleen geven als hij daadwerkelijk het GSM-toestel is dat het beweert te zijn, bijvoorbeeld een sleutelcode. Pas nadat het netwerk dit antwoord goedgekeurd heeft, krijgt het GSM-toestel toegang tot het netwerk. Deze authenticatie-procedure vindt meestal plaats wanneer voor het eerst in het GSM-netwerk wordt gebeld naar iemand anders.

### 2.3.4 Plaatsbepaling

Het GSM-netwerk dient ten alle tijde te weten waar een mobiele telefoon zich bevindt. Om hieraan te voldoen worden er twee databanken bijgehouden: het HLR of Home Location Register en het VLR of Visitor Location Register. Het HLR bevat alle klanten die gebruik mogen maken van het netwerk, in het VLR staat de locatie van de mobiele telefoons die aangemeld zijn bij een gedeeld netwerk. Dit kunnen dus ook mensen zijn die bij een andere operator aangesloten zijn, maar die gebruik mogen maken van het netwerk dankzij een roamingovereenkomst.

Wanneer een mobiele telefoon ingeschakeld wordt, zoekt deze naar signalen uitgezonden door basisstations (BTS). Nu wordt er nagegaan bij welke operator men geregistreerd staat in het HLR of met welke operator er een roaming overeenkomst is. Men kiest voor het basisstation met de grootste signaalsterkte en er wordt aangemeld bij het HLR door gebruik te maken van gegevens die opgeslagen zijn op de SIM-kaart van de mobiele gebruiker. Wanneer het aanmelden gelukt is, worden de gegevens van de mobiele telefoon doorgezonden naar het basisstation en opgeslagen in het VLR. Wanneer er aangemeld wordt bij een operator waarmee er een roaming overeenkomst is, gaat deze de informatie doorzenden naar de eigenlijke operator van de mobiele gebruiker zodat deze operator steeds weet waar de mobiele telefoon zich bevindt. Nu weet het netwerk op ieder ogenblik waar de mobiele telefoon zich bevindt.

### 2.3.5 GSM data

Belangrijk is dat we weten welke GSM data er momenteel opgeslagen wordt. Onderstaande tabel is een voorbeeld van GSM data [20].

| Tijdstip | IMSI | Eerste cel | Tweede cel | Duur gesprek |
|---|---|---|---|---|
| 01/07/06;12:01:55 | 1 | PI003D2 | PI001D1 | 93,00 |
| 01/07/06;12:02:44 | 2 | PI001D3 | PI001D1 | 93,00 |
| 01/07/06;12:06:05 | 3 | PI003D2 | PI001D1 | 19,00 |
| 01/07/06;12:13:49 | 4 | PI001D1 | PI001D1 | 79,00 |
| 01/07/06;12:16:25 | 5 | PI013G2 | PI001D1 | 129,00 |
| 01/07/06;12:18:29 | 6 | PI001D1 | PI001D1 | 14,00 |
| 01/07/06;12:19:55 | 7 | PI001D1 | PI001D1 | 15,00 |
| 01/07/06;12:23:58 | 8 | PI001D1 | PI003D2 | 35,00 |
| 01/07/06;12:30:19 | 9 | PI001D1 | PI001D1 | 6,00 |
| 01/07/06;12:36:06 | 10 | PI003D2 | PI001D1 | 6,00 |

Figuur 2.3 Tabel met GSM data

Vooraleerst wordt het tijdstip van het gesprek opgeslagen. Dit is ook belangrijk voor de facturatie naar de klant toe zodat deze kan controleren of er wel correct gefactureerd werd. Dan wordt het IMSI opgeslagen. Dit staat voor International Mobile Subscriber Identity en is een uniek nummer geassocieerd met een SIM-kaart. Deze wordt naar de operator gezonden zodat deze de gebruiker kan identificeren (Hier gebruiken we fictieve nummers zodat we geen informatie verspreiden die de privacy van de gebruikers zou schenden). Dan worden er twee identifiërs opgeslagen. Deze staan respectievelijk voor de cel van waaruit gebeld wordt en de cel waar de ontvanger van de oproep zich bevindt. Tot slot wordt ook nog de duur van het gesprek opgeslagen zodat de operator weet hoeveel beltijd hij moet factureren.

## 2.4 UMTS

Ondanks de evolutie heeft GSM nog enkele nadelen. Het GSM netwerk is oorspronkelijk ontwikkeld voor spraak te verzenden. Daardoor voldoet het niet meer aan de eisen van de huidige samenleving. Daarom heeft men een nieuwe technologie ontwikkeld dat de opvolger is van GSM, UMTS [27]. UMTS staat voor Universal Mobile Telecommunication System en behoort tot de 3e generatie (3G). De iPhone is een recent ontwikkelde GSM die gebruik maakt van 3G (zie figuur 2.4).


Figuur 2.4 iPhone 3G van Apple

UMTS is zo ontworpen dat het verzenden van data via pakketten verloopt. Alle data die getransporteerd moet worden zoals spraak, beeld, video worden doorgestuurd op dezelfde manier als in het internetprotocol. De data wordt opgedeeld in kleine pakketjes en daarna doorgestuurd naar de ontvanger. De verschillende datapakketjes kunnen elk een andere weg over het netwerk nemen, ze volgen niet dezelfde lijn zoals het geval is bij GSM. Deze techniek is een belangrijke verandering ten opzichte van GSM en biedt een aantal voordelen.

- Degene die het UMTS toestel gebruikt is altijd ingelogd zodat men niet elke keer opnieuw moet inloggen.
- Bij UMTS wordt Global Roaming gegarandeerd. Dit wil zeggen dat een gebruiker wereldwijd bereikbaar is op hetzelfde nummer.
- Data transfers verlopen veel sneller bij UMTS dan bij GSM.
- UMTS is een wereldwijde standaard.
- UMTS wordt gebruikt voor alle mobiele toepassingen.
- UMTS ondersteunt zowel Packet-Switched als Circuit-Switched communicatie.

Een belangrijk voordeel is dat UMTS sneller is dan GSM. UMTS maakt gebruik van radiogolven. Er zijn drie snelheden voorzien die afhankelijk zijn van de locatie waar de gebruiker zich bevindt. De laagste snelheid die gegarandeerd wordt is 144 KBit/s. Deze is van toepassing voor gebruikers die zich bevinden in een bewegend voertuig. De omstandigheden in zulke gevallen zijn dermate ongunstig voor het verzenden van elektromagnetische golven dat men geen hogere snelheid durft te garanderen. Een tweede snelheid die gegarandeerd wordt is 384 KBit/s voor gebruikers die zich bevinden in Micro- en Macrocellen. UMTS gebruikt deze termen voor gebruikers die zich in een stedelijke omgeving bevinden. Men gaat uit van het slechtst mogelijke geval dat de gebruiker rondwandelt in een reflectierijke omgeving zoals muren van huizen. Omdat de signaalsterkte sterk wisselt en de golven gereflecteerd worden, moeten sommige datapakketten regelmatig opnieuw verzonden worden. Hierdoor zakt het rendement van de dataoverdracht. Er kunnen immers geen nieuwe pakketjes verzonden worden zolang er oude pakketjes opnieuw verzonden moeten worden. De maximale overdrachtssnelheid die gegarandeerd wordt is 2 MBit/s voor gebruikers in de Picocel. De Picocel omvat het huis en de omgeving rond het huis van de gebruiker. Omdat er hier geen

reflectie van de elektromagnetische golven kan gebeuren en er geen verschillen optreden in signaalsterkte, gaat het overdrachtssignaal niet verloren. De maximale snelheid van 2 Mbps is alleen haalbaar in optimale omstandigheden. Wanneer de netwerkbelasting toeneemt, zal bijgevolg de snelheid dus dalen.

UMTS biedt een waaier aan mogelijkheden. Beeldtelefonie is mogelijk, films en TV kijken, overal draadloos surfen en e-mailen, ... . Verder is het mogelijk om snel foto's en bestanden te versturen. Een laptop is uitbreidbaar met een UMTS kaart waardoor men in staat is om altijd contact te houden met het bedrijfsnetwerk. UMTS werkt voorlopig naast GSM, maar stilaan zal UMTS de plaats innemen van GSM zodat na verloop van tijd GSM volledig zal verdwijnen.

## 2.5 Recente toepassingen

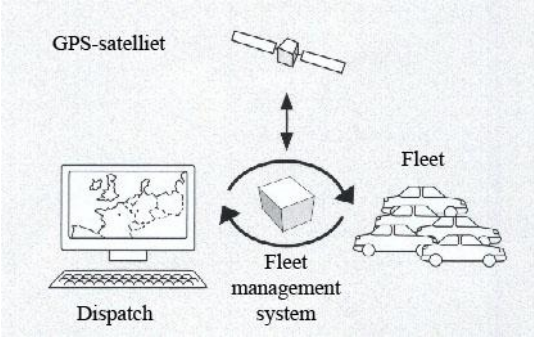
### 2.5.1 Inleiding

Technologie spelt anno 2008 een steeds grotere rol in zowel het optimaliseren van bedrijfsprocessen als het realiseren van beleidsdoelstellingen door overheden. Of het nu gaat over het verhogen van de rendabiliteit van logistieke activiteiten, het beheren van een bedrijfsvoertuigenpark of het reduceren van de CO2 uitstoot van het wegverkeer in België; telematica technologie heeft in een aantal sectoren reeds belangrijke innovaties gerealiseerd en er wordt algemeen aanvaard dat de komende jaren deze spitstechnologie in nog meer domeinen zal gebruikt worden.

In dit deel worden enkele van de laatste ontwikkelingen op het gebied van telematica besproken om de lezer meer inzicht te geven in de huidige stand van zaken binnen deze technologie.

### 2.5.2 Fleet management

Transport speelt een belangrijke rol in de huidige economie. De verschillende bedrijven uit de fleet- en transportsector vormen dan ook een belangrijke afzetmarkt voor telematica.




Figuur 2.5 Schematische voorstelling van een fleet management systeem

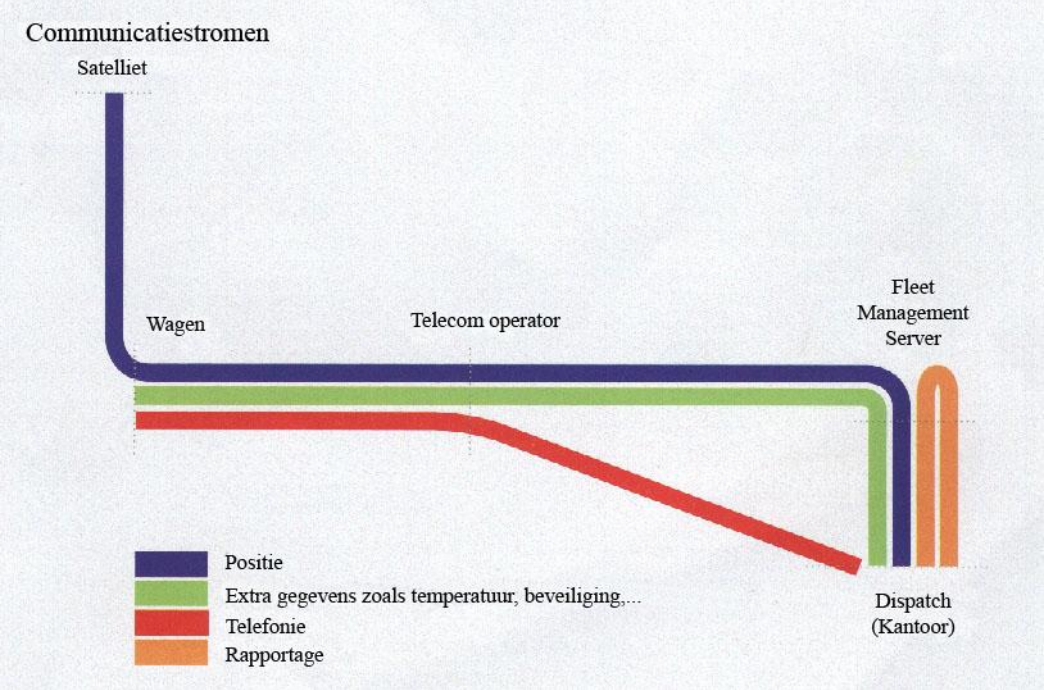
ARC Solutions[1] is een bedrijf dat zich hierop toespitst. Het is een onafhankelijke consultant en heeft als doel het adviseren in het verder uitbouwen en optimaliseren van organisaties door Fleet Management oplossingen en diensten aan te bieden.

Een Fleet Management systeem heeft twee functies. Ten eerste wordt er in real time alle beschikbare informatie over de vloot (de voertuigen van het transportbedrijf) weergegeven. De

[1] http://www.arc-roeselare.be/

positie van elk voertuig, de start- en stoptijden, huidige snelheid, … worden allemaal weergegeven. Ten tweede geeft een Fleet Management systeem de mogelijkheid om op ieder ogenblik met de werknemers te communiceren. Dit kan gebeuren door middel van tekstberichten via het navigatiesysteem of via GSM met handvrije carkit.

Fleet Management heeft vier belangrijke voordelen. Een eerste voordeel is de mogelijkheid tot real-time communicatie. Hierdoor is het mogelijk om de goederen en de werknemers op te volgen in real-time. Zo wordt het mogelijk om de klanten meteen een update te geven over de stand van zaken. Er kan ook eenvoudig en snel belangrijke informatie mee worden gedeeld aan de werknemers waardoor er zeer vroeg gereageerd kan worden op veranderingen in de wensen van de klant of op een wijziging van een verkeerssituatie (vb. een file). Door deze real-time communicatie worden de voertuigen van de vloot ook beveiligd tegen diefstal. Men weet immers op ieder ogenblik waar het voertuig zich bevindt. Alzo is het ook mogelijk om eventuele fraude van het personeel te detecteren (vb. te lange rustperiodes, omwegen voor persoonlijke doeleinden,…) en om bij pech onmiddellijk assistentie te verlenen. Een tweede voordeel is de tijds- en kostenbesparing die een Fleet Management systeem oplevert. Door het bepalen van de meest optimale route voor de chauffeurs kunnen meer klanten bediend worden binnen eenzelfde tijdsspanne en wordt er bespaard op brandstof en onderhoud. Ook de overuren van de werknemers worden verlaagd door een correcte registratie van de werkuren. Een derde voordeel is dat de bedrijfsprocessen geoptimaliseerd worden. Er wordt een overzichtelijke rapportage gegenereerd van de rij- en rusttijden waardoor er geen discussies meer kunnen ontstaan over time sheets en waardoor heel eenvoudig de werklast bijgestuurd kan worden. De loonadministratie kan volledig automatisch verwerkt worden waardoor het papierwerk aanzienlijk verminderd wordt. De werktijden van de werknemers en van de machines kunnen geregistreerd worden waardoor er een optimale en gedetailleerde factuur gemaakt kan worden voor de klanten. Tot slot biedt een Fleet Management systeem ook nog voordelen op het vlak van wettelijke aspecten. Het onderscheid tussen privé en professioneel gebruik van de bedrijfswagens kan makkelijk aangetoond worden. Het is ook mogelijk om temperatuur- en drukregistratie uit te voeren gedurende de volledige rit waardoor aan de nieuwe wetgeving voor traceerbaarheid van de temperatuurkring voldaan wordt.




Figuur 2.6 Schematische voorstelling van communicatiestromen van een fleet management systeem

ARC Solutions biedt vier verschillende technologiën aan die zich elk toespitsen op een specifiek deel van de markt. Een eerste technologie die aangeboden wordt is ontworpen door het Belgisch bedrijf All-Connects[2]. Dit bedrijf is gespecialiseerd in temperatuur en security voor Fleet Management. Zij hebben temperatuurcontrole- en beveiligingssystemen ontwikkeld die in real-time alle noodzakelijke gegevens over de voertuigen doorgeven. Het systeem dat zorgt voor de controle van de temperatuur maakt gebruik van Sensor2Web. Dit systeem volgt de temperatuur van elk voertuig op de voet. Bij de minste afwijking wordt het bedrijf op de hoogte gebracht. Zo weet men steeds dat de goederen vers bij de klant aankomen. Het Sensor2Web-systeem maakt gebruik van gekalibreerde draadloze sensoren die aan de strenge kwaliteitsnormen voldoen. Ze maken positiebepaling en temperatuurscontrole mogelijk via GPRS en SMS of e-mail. Ook wordt, door gebruik te maken van dit systeem, de druk op de administratie verlaagd omwille van de automatische koppeling met de boekhouding.

Een tweede technologie die aangeboden wordt spitst zich toe op Time Management. Deze technologie wordt ontwikkeld door het Belgisch bedrijf GeoDynamics[3]. Omdat het vaak niet zo makkelijk is om de mobiele werknemers stipt op te volgen heeft GeoDynamics hier een oplossing voor ontwikkeld. De werkgever wenst te weten hoeveel uren er effectief gepresteerd zijn door iedere werknemers en waar deze precies geweest is. Om aan deze doelstelling te voldoen wordt ieder voertuig voorzien van een badge-lezer. De werknemers geven aan wanneer ze beginnen of eindigen met hun werk door gebruik te maken van hun persoonlijke badge. Meteen worden deze gegevens doorgestuurd zodat de werkgever altijd een zicht heeft waar iedere werknemer zich bevindt en de start- en stoptijden, de kilometerstand en de exacte locatie van elk voertuig kan bekijken. Deze gegevens kunnen na goedkeuring automatisch doorgestuurd worden naar de boekhoudafdeling zodat ze snel verwerkt kunnen worden. Er wordt ook een privacy button

[2] http://www.all-connects.com/
[3] http://www.geodynamics.be/

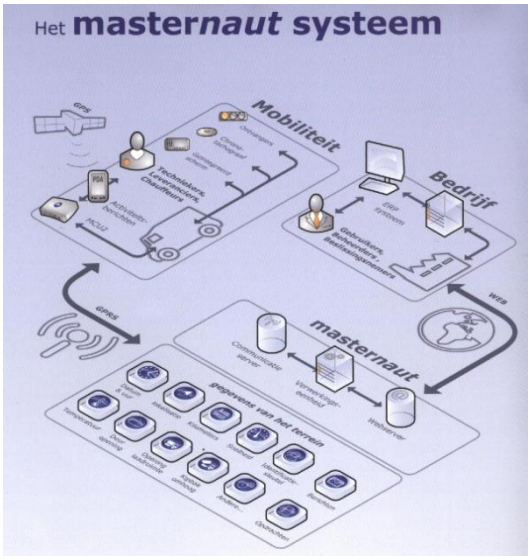
geinstalleerd in elk voertuig zodat de werkgever ook exact weet wanneer een voertuig gebruikt wordt voor privé-doeleinden.

De derde technologie die ARC aanbiedt is een technologie van het bekende bedrijf TomTom[4], genaamd TomTom Work. TomTomWork zorgt voor optimale navigatie, comunicatie en voertuigbeheer. Het systeem tracht om, door een makkelijke communicatie en navigatie in twee richtingen tussen de mensen op de weg en in het kantoor, het beheer over de voertuigen en de chauffeurs te verbeteren. TomTom Work biedt Connected Navigation-oplossingen door gebruik te maken van slimme navigatie- en communicatietools. Elk voertuig moet beschikken over een mobiel navigatiesysteem dat verbonden is met de TomTom WEBFLEET-service voor voertuigbeheer en communicatie. TomTom Work is een snel en efficiënt systeem gericht op tracking, tracing en planning. De werknemer kent de exacte locatie van alle voertuigen en kan ze real-time volgen op gedetailleerde wegenkaarten. Zodra de werkgever een nieuwe opdracht binnenkrijgt, kan hij aan de hand van de locatie van de werknemers en de planning zien welke van de werknemers het meest geschikt is om deze opdracht uit te voeren. Deze opdracht kan dan meteen vanuit het bedrijf samen met de bijhorende adresgegevens verzonden worden naar de werknemer. Deze kan dan met één druk op de knop van zijn navigatietoestel de opdracht aanvaarden en hierna de automatische navigatie-instructies volgen. Hierdoor wordt de kans op fouten (vb. bij het doorgeven van het adres) verkleind. Er worden ook automatisch logboeken bijgehouden met daarin de afgelegede kilometers, de start- en stoptijden, stoplocaties, … . Deze kunnen gebruikt worden voor de administratie bij de berekening van de salarissen van de werknemers.

De vierde technologie die aangeboden wordt, wordt ontwikkeld door Ubidata[5]. Deze technologie is niet specifiek toegepast op een deel van de markt, maar hiermee kan alle richtingen uitgegaan worden. Er kan geopteerd worden voor een minimaal zichtbare tracking oplossing of voor een uitgebreide installatie met een PDA voor communicatie en een ingebouwde printer om meteen te kunnen factureren. Er kan naar keuze van de klant een systeem samengesteld worden dat voldoet aan al diens eisen. Dankzij de module *Gebruikersidentificatie* weet de klant steeds welke medewerker zich op welk systeem heeft aangelogd en de uren waarop deze begonnen en gestopt is met werken. Er kan ook een *Dark Blue Box* ingebouwd worden in de voertuigen zodat op iedere tijdstip van de dag nagegaan kan worden waar het voertuig zich precies bevindt, of er onvoorziene pauzes werden gehouden, of het voertuig veel van de geplande route is afgeweken, enz. In deze *Dark Blue Box* is ook een robuuste PDA geïntegreerd zodat de planners voortdurend op de hoogte gehouden worden van de status van de opdracht. De planners kunnen ook via deze PDA de dagplanning aan de chauffeurs bezorgen. Ook is er een navigatie- en communicatiemodule geïntegreerd in de *Dark Blue Box* zodat de werkgever met de chauffeurs kan bellen of SMS'en.

Een tweede belangrijke speler in de Fleet Management sector is Masternaut[6]. Het bedrijf is gespecialiseerd in Fleet Management en de lokalisatie van personen, voorwerpen en machines. De technologieën van Masternaut vormen een samenhangend geheel waardoor bedrijven hun volledige vloot kunnen opvolgen en beheren via het product van Masternaut.

[4] http://www.tomtom.com/
[5] http://www.ubidata.be/
[6] http://www.masternaut.be/




Figuur 2.7 Het masternaut systeem

Alle gegevens worden via het GSM/GPRS-netwerk van de mobiele telefoonoperatoren naar de Masternautservers verstuurd. Dit gebeurt op een volledig veilige manier zodat er geen gegevens verloren gaan. Door middel van een eenvoudige internetverbinding heeft de klant toegang tot zijn persoonlijke database en krijgt hij een volledig overzicht over zijn vloot. Door gebruik te maken van de ASP-functie (Application Service Provider) kan de onderneming vanaf om het even welke computer verbinding maken. Deze werkwijze laat ook toe om, zonder de service te onderbreken, remote onderhoud en updates uit te voeren. Door de geolokalisatie worden de gegevens op een digitale kaart geplaatst waarop de werkgever via parameters waarschuwingen kan instellen voor ongewone gebeurtenissen zoals overschrijdingen van de rijtijden, te lange pauzes, afwijken van de oorspronkelijke route,… . Deze waarschuwingen worden dan via e-mail en/of SMS verzonden naar de chauffeur. Ondernemingen kunnen zelfs op elk ogenblik rechtstreeks communiceren met hun werknemers op de weg door middel van een PDA. Hiermee kan de werkgever dan de opdracht uitstippelen en vrachtbrieven doorsturen, dringende oproepen, waarschuwingen in verband met de aanwezigheid van de klant, veranderingen in deadlines,… . Er bestaat niet alleen de mogelijkheid om de voertuigen in real-time te volgen, maar wijzigingen aan te brengen in de planning. De gebruiker kan, indien hij dit wenst, dagelijks/wekelijks/maandelijks een rapport aanvragen waarin alle activiteiten tot in detail beschreven staan. Hierin worden met behulp van tabellen en grafieken de gegevens (rijtijden, trajecten, aflevertijden,…) beschreven. Het bedrijf krijgt zo een visueel overzicht en kan zo eenvoudig bijsturen en optimaliseren.

### 2.5.3 Digitale kaarten

De belangrijkste en ook de bekendste ontwikkelaar van digitale kaarten is ongetwijfeld Tele Atlas. Al meer dan twintig jaar ontwikkelen zij digitale kaarten die gebruik worden voor verscheidene doeleinden. Navigatie is ongetwijfeld het belangrijkste. De verkoop van de zogehete PND's (Personal Navigation Devices) is de laatste jaren geëxplodeerd. Meer en meer mensen beschikken tegenwoordig over een PND en dit aantal zal in de komende jaren nog toenemen. Deze toestellen helpen de mensen om de directe omgeving waarin ze zich bevinden snel en efficiënt te interpreteren; of het nu gaat om de route te vinden om van punt A naar punt B te gaan, het dichtsbijzijnde metro station te vinden of om te weten wat een toeristisch gebied allemaal te bieden heeft. Maar opdat deze PND's ook van enige waarde zouden zijn voor de gebruikers, dienen ze de

juiste inhoud te bevatten. Tele Atlas zorgt voor uitgebreide inhoud op elk vlak. Ze ontwikkelen niet alleen zeer gedetailleerde kaarten, maar zorgen ook voor bijkomende diensten en informatie.

Tele Atlas Voice Maps verschillen van gewone kaarten in het feit dat zij de namen van plaatsen opslaan in fonetisch formaat. Zo laten deze Voice Maps in samenwerking met een Text-To-Speech engine toe om deze namen correct te laten uitspreken. Ook kan het systeem extra gedetailleerde instructies geven, vb. Bij het volgende kruispunt, draai rechts richting Brussel. Ook kan de gebruiker zelf de naam van een locatie ingeven via een micro in plaats van deze in te typen.

Een andere zeer recente ontwikkeling is de Tele Atlas 3D Content. Door de toevoeging van 3D Landmarks tot zelfs volledige steden in 3D krijgt de gebruiker een straffere ervaring bij het reizen. De 3D Landmarks tonen lokale landmarks volledig getextured in 3D op het navigatietoestel, en 3D City Maps zijn drie-dimensionale voorstellingen van gebouwen, stadsblokken, ... in steden. De gebruiker heeft niet alleen een mooier beeld van waar hij zich bevindt, maar oriëntatie en navigatie wordt ook gemakkelijker op plaatsen waar men nog nooit geweest is. Voor de navigatietoestellen die niet krachtig genoeg zijn om de 3D gegevens dynamisch te renderen, zijn bitmap afbeeldingen voorzien zodat de gebruiker toch gemakkelijk landmarks en belangrijke gebouwen in steden kan herkennen.

Figuur 2.8 Landmark in 3D

Ook voor de voetgangers heeft Tele Atlas kaarten op de markt gebracht, de zogehete Pedestrian Maps. Voetgangers hebben meer mogelijkheden om een gegeven bestemming te bereiken, openbaar vervoer, wandelpaadjes, doorheen een park, autovrije zones, enz. Ze zijn ook geïnteresseerd in routes die kort en veilig zijn (vb. zebrapaden).

Een laatste recente ontwikkeling van Tele Atlas is Tele Atlas ContentLink. Dit is een unieke web-gebaseerde dienst die een platform aanbiedt voor content-ontwikkelaars (vb. Parking garages, WiFI hotspots, bioscopen, restauranten, ...) om met andere partners van Tele Atlas in contact te komen. Door de krachtige editing en management tools van ContentLink worden content-ontwikkelaars geholpen met het ontwikkelen van nieuwe, gedetailleerde en accurate data die toegevoegd kan worden aan applicaties die beschikbaar zijn voor millioenen gebruikers.

### 2.5.4 Dynamische signalisatie

Variabele verkeersborden zijn een praktisch en doeltreffend middel om weggebruikers te informeren. Dynamische signalisatie kan ingezet worden voor verschillende doeleinden. Ze vinden hun toepassing in filebewaking, informatiecampagnes, omleidingen en veiligheid bij wegenwerken, verkeersinformatie, snelheidsbewaking, publiciteit, enz.

Een bedrijf dat deze technologie aanbiedt is TC Matix. Zij bieden zogenaamde tekstkarren aan. Deze bestaan uit een groot bord dat het verkeer vanop grote afstand via LED's informeert. Zowel tekst als afbeeldingen kunnen door dit bord weergegeven worden. Deze berichten kunnen in meerdere kleuren weergegeven worden. Deze tekstkarren hebben een autonomie tot dertig dagen die mogelijk gemaakt wordt door het gebruik van groene energie in de vorm van zonnepanelen en een windmolentje. Deze tekstkarren kunnen van op afstand aangestuurd worden via GSM/GPRS. Ze bevatten ook een elektronisch anti-diefstalsysteem dat tracking en tracing mogelijk maakt via GPS. Ze kunnen ook uitgerust worden met een radar voor snelheids- en filedetectie.


Figuur 2.9 Dynamische verkeerssignalisatie

### 2.5.5 The Target game

[21] The target is een interactief spel dat gespeeld wordt in de echte wereld. De bedoeling van het spel is dat een misdadiger allerlei digitale voorwerpen in een stad moet stelen. Een groep agenten moet er voor zorgen dat ze de misdadiger onderscheppen vooraleer hij alle voorwerpen gestolen heeft. Zowel de misdadiger als de agenten worden uitgerust met een Nokia 6110 Navigator. Dit is een mobiele telefoon die gebruik maakt van UMTS, de opvolger van GSM. Telkens de misdadiger een voorwerp steelt, krijgen de agenten een melding op hun mobiele telefoon. Zo laat de misdadiger een spoor achter in de stad dat de agenten dienen op te sporen en te achtervolgen. Om de zes minuten krijgen de agenten een update over de positie van de misdadiger doordat ze zogezegd in verbinding staan met politiesatellieten. Maar de misdadiger kan dit signaal onderscheppen en dus krijgt hij ook de posities door waar de agenten zich bevinden. De agenten kunnen op elk ogenblik nagaan hoe ver ze van de misdadiger verwijderd zijn. De positie van de spelers wordt via GPRS doorgestuurd naar de spelserver zodat een vlot spelverloop mogelijk gemaakt wordt.


Figuur 2.10 Nokia 6110 Navigator

# Hoofdstuk 3: Map-matching

[4-7]

## 3.1 Inleiding

Trajecten worden gereconstrueerd op basis van data doorgegeven via navigatietoestellen. De punten die doorgegeven worden zijn niet exact. Er is een kleine fout op de meting. Bij GPS kan deze afwijking oplopen tot maximaal tien meter, terwijl bij Galileo de maximale afwijking één meter bedraagt. Door map-matching toe te passen worden deze punten gekoppeld aan kaarten zodat de afgelegde trajecten daadwerkelijk over wegen lopen. Deze trajecten bestaan uit een reeks punten (coördinaten) opgenomen op verschillende tijdstippen. Door deze punten met elkaar te verbinden wordt een traject gecreëerd. Deze trajecten worden opgeslagen in een database van tupels. Deze tupels zijn van de vorm $(t_i, x_i, y_i)$. Hierbij staat $t_i$ voor een tijdstip en $x_i$ en $y_i$ stellen een punt voor in het vlak. Maar tussen twee opeenvolgende punten is er een onzekerheid over waar het object zich bevindt. Beads zijn een manier om deze onzekerheid grafisch voor te stellen.

Het *space-time prism* model [5] is een model voor het managen van de onzekerheid van de locatie van een bewegend object tussen twee opeenvolgende punten van een traject. In dit model wordt er verondersteld dat er meer achtergrondinformatie beschikbaar is over de weg tussen twee opeenvolgende punten. Zo wordt er vaak gebruik gemaakt van een snelheidslimiet $v_i$ op de route tussen twee punten.

Data gebruikt in trajecten wordt verzameld van bewegende objecten, deze objecten bewegen voort over een netwerk van straten. Dit stratennetwerk, *road network* genaamd, wordt schematisch voorgesteld door een graaf in het vlak. Alle bogen van deze graaf zijn rechte lijnen tussen twee knopen. Alle bogen hebben ook een kost om van de startknoop naar de eindknoop te bewegen. Deze kost is de tijd die er nodig is om van de startknoop naar de eindknoop te bewegen aan de maximale snelheid die er geldt tussen die twee punten. Het kortste pad tussen twee punten in een *road network* wordt de *road network time* genoemd en is dus de weg die het minst tijd in beslag neemt om zich van het beginpunt naar het eindpunt te verplaatsen via het *road network*. Wanneer de snelheidslimieten over het hele *road network* hetzelfde zijn, dan is de *road network time* gelijk aan het kortste pad. Een bekend algoritme om het kortste pad te berekenen is het algoritme van Dijkstra.

De *space-time prism* tussen twee opeenvolgende punten wordt gedefinieerd door de verzameling van *space-time* punten waar het object geweest kan zijn rekening houdend met de geldende snelheidslimiet tussen deze twee punten.

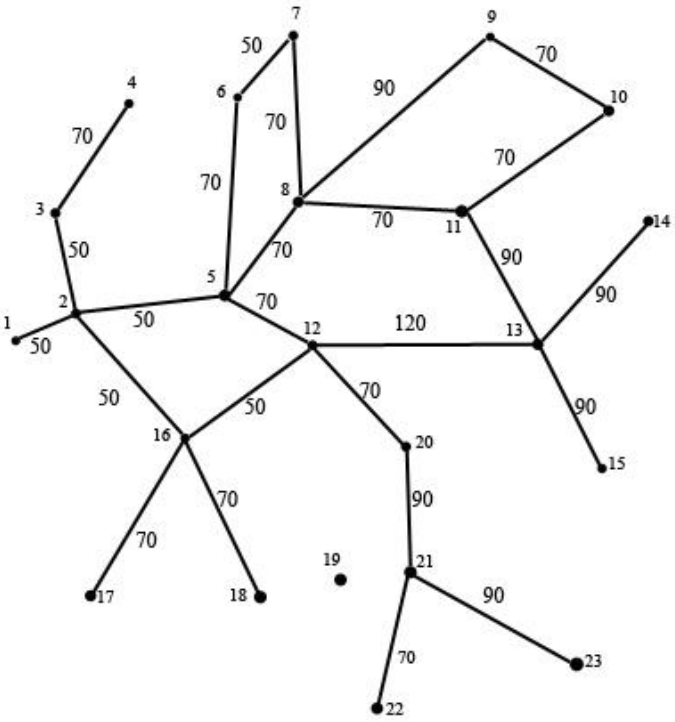
## 3.2 Definities

Kuijpers en Othman beschouwen in [5] bewegende objecten in het tweedimensionale vlak $\mathbb{R}^2$ en beschrijven hun beweging in de $(t,x,y)$-ruimte waar $t$ de tijd in voorstelt. Bewegende objecten, die voorgesteld worden door punten, maken een speciale soort curve, die trajecten genoemd worden.

Stel $I \subseteq \mathbb{R}$ een interval. Een traject $T$ is de graaf van een mapping $\alpha : I \to \mathbb{R}^2 : t \to \alpha(t) = (\alpha_x(t), \alpha_y(t))$, bvb. $T = \{(t, \alpha_x(t), \alpha_y(t)) \in \mathbb{R} \times \mathbb{R}^2 \mid t \in I\}$. We noemen $I$ het tijdsdomein van $T$.

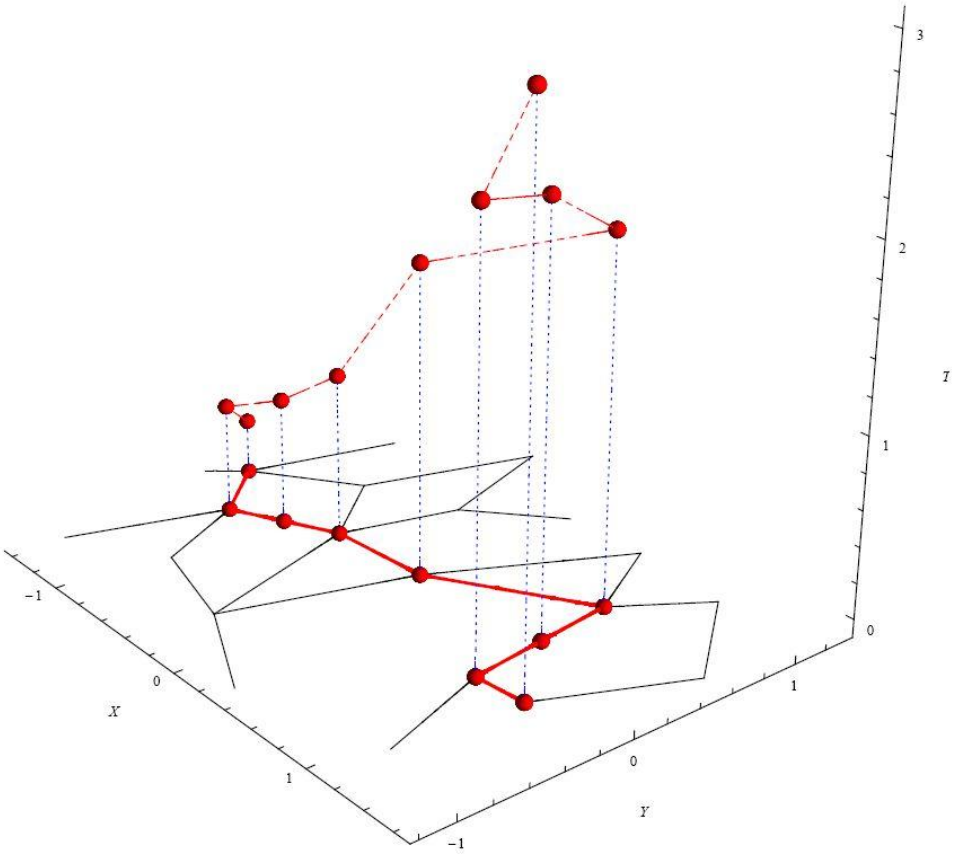
Een sample van een traject is een eindige set $S = \{(t_0, x_0, y_0), (t_1, x_1, y_1) \ldots (t_N, x_N, y_N)\}$ van *time-space* punten. Uit de orde op tijd, $t_0 < t_1 < \ldots < t_N$, wordt de natuurlijke orde van het sample afgeleid.

Een *road network* RN is een gelabelde graaf in het vlak $\mathbb{R}^2$ gegeven door een eindige set van knopen $V = \{(x_i,y_i) \in \mathbb{R}^2 \mid i = 1,\ldots, N\}$ en een set van bogen $E \subseteq V \times V$ gelabeld met een snelheidslimiet en een tijdsspanne. De graaf voldoet aan twee condities. De eerste conditie houdt in dat elke boog een rechte lijn is tussen twee knopen. De tweede conditie houdt in dat als een boog tussen knoop $(x_i,y_i)$ en knoop $(x_j,y_j)$ gelabeld is met de snelheidslimiet $v_{ij} > 0$, de tijdsspanne van deze boog $w_{ij}$ gelijk is aan $[\sqrt{((x_i - x_j)^2 + (y_i - y_j)^2}] / v_{ij}$ , met andere woorden de tijdsspanne is de tijd nodig om van de ene knoop naar de andere te gaan met een snelheid gelijk aan de snelheidslimiet.




Figuur 3.1 Een road network

Een traject in een *road network* RN is een traject wiens ruimtelijke projectie in RN ligt. Formeler, als *T* een traject is gegeven door de functies $\alpha_x$ en $\alpha_y$, dan moet het voldoen aan $(\alpha_x(t), \alpha_y(t)) \in$ RN voor alle *t* in het tijdsdomein van *T* en voor een monster van een traject $S = \{(t_0,x_0,y_0),(t_1,x_1,y_1),\ldots,(t_N,x_N,y_N)\}$ moet $(x_i,y_i) \in$ RN voor alle $i = 0,\ldots,N$.



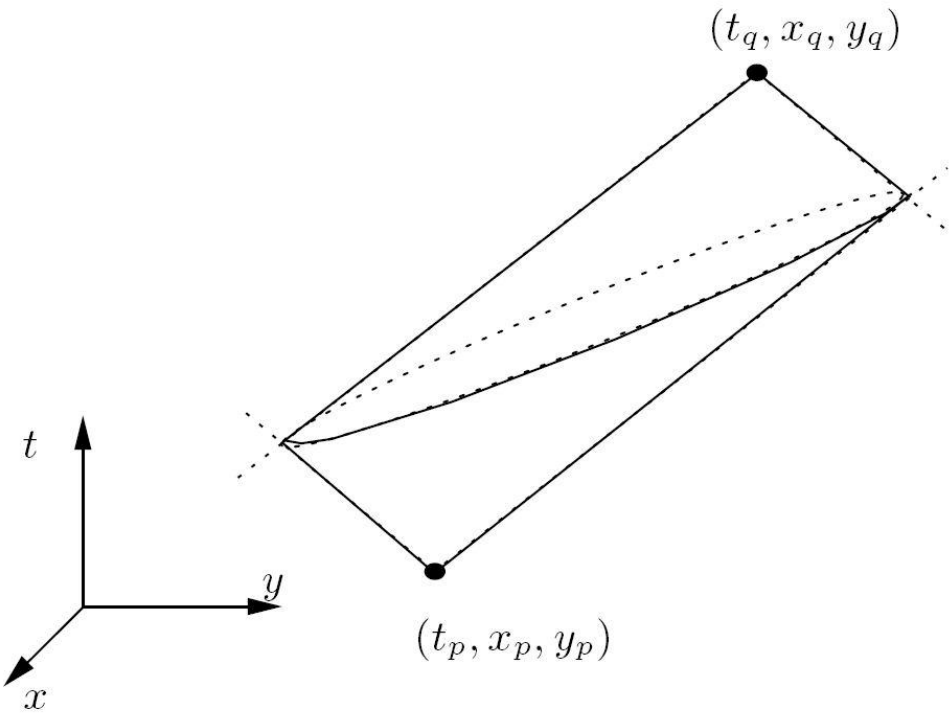

Figuur 3.2 Een traject in tijd en ruimte en de projectie op het route netwerk

In de praktijk weet men vaak meer over een traject dan enkele punten $(t_i,x_i,y_i)$, $i = 0,\ldots,N$. Bijvoorbeeld een door de wet opgelegde snelheidslimiet $v_{max}$ op locatie $(x_i,y_i)$. Als we aannemen dat een snelheidslimiet geldig is tot het volgende punt van het traject, kunnen we deze snelheidslimiet gebruiken om *space-time prisms* te definiëren die de onzekerheid van de locatie van een object tussen twee punten modelleren.

Stel M een metrische ruimte met metriek $d_M$. Stel $p, q \in M$. De *space-time prism* met oorsprong $(t_p, p)$, bestemming $(t_q, q)$ en maximale snelheid $v_{max} \geq 0$ is de set van alle *time-space* punten $(t, u) \in \mathbb{R} \times M$ die voldoen aan de volgende vergelijkingen:

- $d_M(p, u) \leq (t - t_p)v_{max}$
- $d_M(u, q) \leq (t_q - t)v_{max}$
- $t_p \leq t \leq t_q$

We noteren deze deelverzameling van $\mathbb{R} \times M$ als $P^M(t_p, p, t_q, q, v_{max})$.
De continue curve die de punten van een monster van een traject met elkaar verbindt is de *geospatial lifeline* en de opeenvolging van *space-time prisms* die met elkaar verbonden zijn langs opeenvolgende punten van het traject is de *lifeline necklace.*




Figuur 3.3 Een bead tussen twee punten

Er moet een geschikte afstandsfunctie op een *road network* gedefinieerd worden om *space-time prisms* te definiëren op een *road network.* Degene die gebruikt wordt is afgeleid van de kortste pad afstand gebruikt in de graaftheorie.

Veronderstel een *road network* RN, met verzameling knopen V en verzameling gelabelde bogen E. Stel p = ($x_p$,$y_p$) en q = ($x_q$,$y_q$) twee punten op het *road network* RN. De punten p en q moeten niet noodzakelijk knopen van RN zijn. Veronderstel dat p op de boog (($x_{p,0}$,$y_{p,0}$),( $x_{p,1}$,$y_{p,1}$)) ligt en q op de boog (($x_{q,0}$,$y_{q,0}$),($x_{q,1}$,$y_{q,1}$)). We construeren een nieuw *road network* $RN_{pq}$ uit RN met als verzameling knopen $V_{pq}$ = V ∪ {p, q} en als verzameling bogen $E_{pq}$ = E ∪ {(($x_{p,0}$,$y_{p,0}$), ($x_p$,$y_p$)), (($x_p$,$y_p$),($x_{p,1}$,$y_{p,1}$)), (($x_{q,0}$,$y_{q,0}$), ($x_q$,$y_q$)), (($x_q$,$y_q$), ($x_{q,1}$,$y_{q,1}$))}. De bogen waarop p en q lagen zijn dus elk opgesplitst in twee bogen. Deze bogen hebben als label dezelfde snelheidslimiet als de oorspronkelijke bogen en de tijdsspannen worden opnieuw berekend.

Stel RN een *road network* en laat p, q ∈ RN. De *road network time* tussen p en q, genoteerd als $d_{RN}$(p, q), is de kortste pad afstand (volgens het algoritme van Dijkstra) tussen p en q in de graaf ($V_{pq}$, $E_{pq}$) rekening houdend met de tijdsspanne als labels van de bogen.

Merk op dat de *road network time* tussen p en q de snelste tijd is om p te bereieken vanuit q en omgekeerd. De metriek hierboven beschreven neemt twee punten van een *road network* en geeft de kortste tijd terug om van één punt naar het ander te gaan met een snelheid die gelijk is aan de maximale snelheid van elk segment.

Een *space-time prism* op een *road network* is de geometrische locatie in $\mathbb{R}$ x RN ⊂ $\mathbb{R}$ x $\mathbb{R}^2$ van alle punten die een bewegend object mogelijk bezocht kan hebben op weg van oorsprong p naar bestemming q binnen een tijdsspanne van $t_p$ tot $t_q$, rekening houdend met de snelheidslimieten op

de bogen van RN. Gegeven een *road network* RN, punten p, q en u op RN en tijdstippen $t_p$ en $t_q$ voor punten p en q, dan gebruiken we $t_u^-$ om $t_p + d_{RN}(p,u)$ af te korten en $t_u^+$ om $t_q - d_{RN}(u,q)$ af te korten.

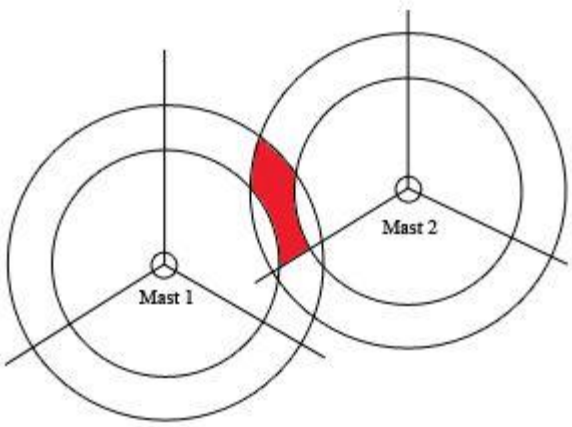
Stel RN een *road network* en stel p, q ∈ RN. De *space-time prism* op het *road network* tussen ($t_p$, p) en ($t_q$,q), rekening houdend met de snelheidslimieten van RN, wordt genoteerd als $P^{RN}(t_p,p,t_q,q)$ en is gedefinieerd als de verzameling van tupels (t,x,y) die voldoen aan volgende voorwaardes:

- (x,y) ∈ RN
- $d_{RN}(p,(x,y)) + d_{RN}((x,y),q) \leq (t_q - t_p)$
- $t^-_{(x,y)} \leq t \leq t^+_{(x,y)}$

De *space-time prism* op het *road network* tussen ($t_p$,p) en ($t_q$,q) rekening houdend met een algemene maximale snelheid $v_{max}$ is de *space-time prism* op RN nadat alle bogen van RN opnieuw gelabeld zijn met $v_{max}$ als snelheidslimiet.

## 3.3 Beads in combinatie met GSM

Wanneer beads gebruikt worden in combinatie met GSM data duikt er een extra probleem op. Een bead wordt geconstrueerd tussen twee punten. Wanneer er gewerkt wordt met GSM data zijn er geen twee punten om een bead tussen te construeren, maar twee gebieden. Dit komt omdat er een onzekerheid is over de locatie van het GSM signaal.




Figuur 3.4 Localisatie mbv GSM masten en signaalsterkte

Op figuur 3.4 is in het rood het gebied aangegeven waaruit het GSM signaal afkomstig is. Er moet dus een bead geconstrueerd worden tussen twee gebieden. Een mogelijkheid is om de verzameling van beads te nemen tussen elke twee punten uit de twee gebieden. Naar technieken om zulke beads te construeren wordt nog volop onderzoek verricht.

## 3.4 Alibi query

[6] De alibi query is een booleaanse query die vraagt of twee bewegende objecten, waarvan de trajecten die bestaan uit *time-space* punten, elkaar mogelijk hebben ontmoet of als ze een alibi

hebben om elkaar niet te ontmoeten. Dit komt neer op het onderzoeken of de *lifeline necklace* van beide trajecten elkaar snijden of niet.

Om de alibi query efficiënt te beantwoorden, volstaat het om een efficiënte methode te gebruiken die beslist of twee *space-time prisms* elkaar snijden. Het is interessant om te weten of twee bewegende objecten elkaar ontmoet kunnen hebben, waar ze elkaar ontmoet kunnen hebben, wanneer en voor hoe lang ze elkaar ontmoet kunnen hebben en waar ze samen gereisd kunnen hebben. Al deze vragen kunnen beantwoord worden door het snijpunt te berekenen als het bestaat.

## 3.5 Mapping op een netwerk van straten

[7]Er zijn twee manieren om de data te mappen op een stratenplan. Een naïeve methode en een methode die gebruik maakt van beads. De data die we moeten mappen bestaat uit een aantal trajecten. Deze trajecten zijn een opeenvolging van discrete punten. Het stratennetwerk waarop we deze data moeten mappen is een kaart in digitale vorm.

### 3.5.1 Naïeve methode

De naïeve methode gaat vrij voor de hand liggend te werk bij het mappen van data op een stratennetwerk. Voor elk punt worden een aantal wegsegmenten geselecteerd die het dichtst bij het punt liggen. Vervolgens worden deze gecombineerd tot een mooi opeenvolgende route van wegsegmenten. Dit algoritme kampt met enkele problemen. Zo kan het zijn dat een punt van het traject even ver ligt van meerdere verschillende wegsegmenten. Dit probleem kan onder andere opgelost worden door de richting van het traject te bekijken en de rijrichting die toegelaten is in de straten. Een ander probleem is dat de trajecten bestaan uit discrete punten. Soms is de afstand tussen twee opeenvolgende punten zo groot dat het niet voor de hand ligt welke route genomen is om van het ene punt naar het andere te gaan.

### 3.5.2 Methode aan de hand van beads

Wanneer trajecten beschreven worden, krijgen we automatisch te maken met een mate van onzekerheid. Deze onzekerheid springt voort uit het feit dat er interpolatie technieken gebruikt worden om de trajecten te construeren of uit het feit dat er meetfouten kunnen optreden bij de sensoren die het traject van het bewegend object opvangen. Deze twee onzekerheden worden respectievelijk interpolatie onzekerheid en onzekerheid door meeting genoemd. Wanneer GPS data gebruikt wordt, is deze zeer klein in vergelijking met de interpolatie onzekerheid. Deze interpolatie onzekerheid kan gemodelleerd worden door gebruik te maken van beads. Zoals eerder al besproken is het niet eenduidig waar een object zich bevindt tussen twee opeenvolgende punten van een traject. Er is een onzekerheid over de locatie van het object. De bekendste manier om een traject te vormen met discrete punten is met behulp van lineaire interpolatie. Maar dit vereist dat een object bij het verplaatsen van het ene punt naar het volgende punt een constante snelheid heeft. Op een stratennetwerk is dit echter niet realistisch. Beads vormen een manier van interpoleren die rekening houden met de snelheidslimieten van de wegsegmenten tussen twee opeenvolgende punten. Ze vormen een volume dat het gebied begrensd waar een object geweest kan zijn tussen twee opeenvolgende punten rekening houdend met de geldende snelheidslimieten. Door tussen elke twee opeenvolgende punten een bead te construeren, houden we een beperkt aantal wegsegmenten over, namelijk de wegsegmenten die zich binnen deze bead bevinden. Wanneer we hier nu een algoritme dat het kortste pad berekent tussen twee punten op los laten krijgen we een route die de originele route sterk benadert. Maar omdat lang niet altijd de kortste route gekozen

wordt, kunnen er optimalisaties uitgevoerd worden zodat de route die het algoritme vindt de originele route beter benadert. In de volgende sectie worden twee algortimes besproken die gebruik maken van optimalisaties, namelijk een *score calculating* algoritme en een *k-shortest path* algoritme.

#### *3.5.2.1 Score calculating algoritme*

Het basisprincipe van dit algoritme is dat er een score toegekend wordt aan de verschillende wegsegmenten. Hoe dichter het wegsegment bij de gegeven punten ligt, hoe beter de score van dit wegsegment. De straat die zich het dichtst bij het gegeven punt bevindt krijgt score *n*, de straat die het tweede dichtst bij het gegeven punt ligt krijgt score *n-1*, en zo wordt er voor elke straat een score berekend totdat de *n-de* dichtstbijzijnde straat score 1 krijgt. Natuurlijk moet niet aan alle mogelijke straten een score toegekend worden. Er moet alleen een score toegekend worden aan de straten die mogelijk gekozen zijn door het bewegend object. Hiervoor kunnen we beads gebruiken. Er wordt dus alleen een score berekend voor de straten die zich binnen de bead bevinden.

#### *3.5.2.2 K-shortest path algoritme*

Het *k-shortest path* probleem is een welgekend probleem in netwerken. In plaats van één kortste pad te zoeken tussen twee punten, worden er *k* kortste paden gezocht. Er bestaan verscheidene algoritmes om *k-shortest paths* te berekenen. Volgens Gheys in [7] is het algoritme van Yen het meest geschikt. Het algoritme van Yen gebruikt het algoritme van Dijkstra voor het berekenen van de *k-shortest paths*. In een eerste fase wordt het kortste pad berekend met het A* algoritme. Vervolgens wordt in een tweede fase voor elk punt in het kortste pad, op het eindpunt na, een tweede kortste pad berekend naar het eindpunt. Dit tweede pad vormt samen met het pad van het startpunt naar het geselecteerde punt (het wortel pad) een pad van het beginpunt naar het eindpunt. Op dit tweede pad worden twee beperkingen geplaatst: het mag geen punten bevatten die in het wortel pad liggen en het mag niet aftakken van het geselecteerde punt naar een wegsegment gebruikt door een eerder gevonden *k-shortest path*. Wanneer zo een pad gevonden is, wordt het wortel pad uitgebreid met dit pad om een pad te bekomen van het beginpunt naar het eindpunt. Vervolgens wordt de score berekend voor dit pad zoals beschreven in sectie 3.5.2.1. Als deze score groter of gelijk is aan de score van het kortste pad uit de eerste fase, dan wordt dit pad toegevoegd aan het resultaat. Tot slot wordt het pad geselecteerd met de beste score.

### 3.5.3 Vergelijking tussen de verschillende methodes

Wanneer we de naïeve methode vergelijken met de methode die gebruik maakt van beads in combinatie met *k-shortest path* komen we tot de volgende conclusies. De naïeve methode geeft een zeer goed resultaat wanneer de data die gebruikt wordt van zeer hoge kwaliteit is. Het is ook een zeer snel algoritme. Maar wanneer de data van mindere kwaliteit is en er bijvoorbeeld gaten in de data voorkomen, dan geeft deze methode een slecht resultaat terug. Het algoritme dat gebruik maakt van beads en *k-shortest path* daarentegen geeft altijd een goed resultaat. Maar het nadeel van dit algoritme is dat het veel trager werkt dan het naïeve algoritme.

# Hoofdstuk 4: Privacy

[2,3]

## 4.1 Inleiding

Bij het verzamelen, bestuderen en analyseren van data speelt privacy een belangrijke rol. De huidige wetgeving is zeer streng omtrent de bescherming van de persoonlijke levenssfeer. Er moet dus streng op toegezien worden dat ten alle tijde de privacy van de mensen wiens data gebruikt wordt gerespecteerd wordt. Maar waar moet er een lijn getrokken worden? Met andere woorden wanneer kan gebruik van data de privacy schenden en wanneer niet? In dit hoofdstuk willen we de lezer overtuigen dat het gebruik van data en in het bijzonder geografische data de privacy kan schenden en zullen we enkele technieken uitleggen die toegepast kunnen worden op data zodat deze gebruikt kan worden zonder dat de privacy geschonden kan worden.

## 4.2 Geschiedenis

De behoefte aan privacy is geen fenomeen dat typisch is aan de moderne maatschappij. In de dierenwereld vinden we zelfs als de nood aan privacy terug. Dieren proberen ook vaak via allerlei technieken een eigen territorium af te bakenen om zo hun privacy te garanderen. Biologen hebben ook aangetoond dat binnen eenzelfde groep mechanismen en technieken bestaan die het individu een bepaalde afstand moeten garanderen ten opzichte van zijn soortgenoten. Zelfs in primitieve samenlevingen bestond de behoefte tot privacy al. Antropologen hebben aangetoond dat hoewel groepen vaak samenleefden in gemeenschappelijke ruimtes, het individu steeds beschikte over momenten waarop hij zich kon terugtrekken of werden er andere technieken gebruikt om afstand te bewaren tot zijn of haar soortgenoten.

Ook in bijna alle godsdiensten vinden we de behoefte aan privacy terug. Mensen zijn er altijd van overtuigd geweest dat ze altijd en overal gezien en gevolgd worden door goden of geesten. Om met deze in contact te komen werd er al van in de oudheid privacy gezocht: als individu op afzondering in een klooster, of geestelijke afzondering door te mediteren.

Een ander bewijs dat de behoefte aan privacy niet iets alleen van deze tijd is, is de tendens vanwege individuen om de privacy van anderen te schenden. Ook de samenleving die individuen controleert en sanctioneert voor asociaal gedrag is hier een bewijs van. Een beetje nieuwsgierigheid ten opzichte van wat iemand anders doet, wordt in alle samenlevingen wel getolereerd en zelfs als positief beschouwd. Maar als deze nieuwsgierigheid te ver gaat wordt deze niet meer getolereerd en zelfs bestraft. Dat dit al sinds lang zo is getuigen de talrijke mythes en sprookjes zoals de doos van Pandora[7] en de vrouwen van Blauwbaard[8] die toch een kijkje nemen in de verboden kamer.

Maar de graad van privacy neemt wel toe naarmate de samenleving zich verder ontwikkeld heeft. Hierdoor krijgen individuen ook een grotere mate van vrijheid en anonimiteit, beide begrippen die onlosmakelijk verbonden zijn met het goede functioneren van een democratie.

## 4.3 Definitie

De term privacy, of met andere woorden een eigen levenssfeer voor het individu, dook pas op in wetten en rechtsboeken rond het eind van de negentiende eeuw. In een beroemde verhandeling

[7] http://nl.wikipedia.org/wiki/Pandora_(mythologie)
[8] http://www.beleven.org/verhaal/blauwbaard

over onrechtmatige daad gepubliceerd in de Verenigde Staten schreef Thomas Cooley voor de eerste keer over "*the right to be left alone*". Ook eind negentiende eeuw werden in verschillende Amerikaanse staten rechterlijke vonnissen geveld waarin het recht op privacy wordt bevestigd. Bij de overgang naar de twintigste eeuw zorgde de opkomst van de massamedia in de vorm van kranten en radio ervoor dat het recht op privacy zeer sterk in de publiciteit kwam, vooral door het feit dat er persoonlijke gegeven gepubliceerd werden van bekende personen.

Mede als gevolg van de slechte ervaringen van het nationaal-socialistisch bewind van Hitler tijdens de Tweede Wereldoorlog werd op 4 november 1950 door de Raad van Europa het Europees Verdrag tot Bescherming van de Rechten van de Mens en de Fundamentele Vrijheden (EVRM) goedgekeurd. Artikel acht hiervan behandelt de privacy en luidt als volgt:

*"1. Eenieder heeft recht op eerbiediging van zijn privé-leven, zijn gezinsleven, zijn huis en zijn briefwisseling.*

*2. Geen inmenging van enig openbaar gezag is toegestaan met betrekking tot de uitoefening van dit recht dan voor zover bij de wet is voorzien en in een democratische samenleving nodig is in het belang van 's lands veiligheid, de openbare veiligheid en het economisch welzijn van het land, de bescherming van de openbare orde en het voorkomen van strafbare feiten, de bescherming van de gezondheid of de goede zeden of voor de bescherming van de rechten en vrijheden van anderen."*

## 4.4 Europese richtlijnen voor het beschermen van data

In 1995 werden door de Europese Unie richtlijnen [28] opgesteld voor het beschermen van data. Het doel van deze richtlijnen was het invoeren van consistente niveaus van bescherming in Europa voor de inwoners. Zo moest de vrije overdracht van persoonlijke data mogelijk worden. Deze richtlijnen zijn van toepassing op het verwerken van persoonlijke informatie in zowel elektronisch formaat als op papier.

In deze richtlijnen worden verscheidene basisprincipes voor Europese burgers gedefinieerd. Deze principes bevatten ondermeer volgende rechten:

- Het recht om te weten van waar data komt
- Het recht om incorrecte data te verbeteren
- Het recht om toestemming te ontzeggen voor het gebruik van data in bepaalde omstandigheden

De lidstaten worden door deze richtlijnen ook verplicht om ervoor te zorgen dat de persoonlijke informatie van Europese burgers op dezelfde wijze beschermd wordt wanneer deze geëxporteerd en verwerkt wordt in landen buiten de Europese Unie. Deze vereiste zorgde voor een verhoging van de druk op landen buiten Europa om ook een strikte en internationaal goedgekeurde privacy wetgeving op te stellen.
In 1997 werden deze richtlijnen uitgebreid met richtlijnen voor privacy in telecommunicatie [29]. Deze nieuwe richtlijnen moesten zorgen voor bescherming bij opkomende technologieën zoals telefoon, digitale televisie, mobiele netwerken en andere telecommunicatiesystemen.
In 2000 werd er in de Europese Commissie een nieuw voorstel gedaan om nieuwe richtlijnen te ontwikkelen in verband met privacy in de elektronische communicatiesector. Er werd voorgesteld om de rechten op privacy van individuen te versterken door de bestaande richtlijnen aan te passen.

Tijdens het opstellen van deze richtlijnen werd ook een regel opgesteld die internet providers en telecommunicatie operatoren verplicht om logs op te slaan van alle telefoongesprekken, e-mails, faxen en informatie over surfgedrag op het internet van de voorbije twee jaar. Dit voor het eventuele gebruik voor het opsporen van terrorisme of andere vormen van criminaliteit. In het begin was er sterke tegenstand tegen dit voorstel omdat dit de privacy van individuen aan grotere risico's blootstelde. Maar na de aanslagen van elf september veranderde het politieke klimaat en werd dit voorstel goedgekeurd. Volgens deze richtlijn mogen Europese lidstaten wetten opstellen die toelaten dat het verkeer en de locatie data van alle communicatie, die gebeurt via mobiele telefonie, SMS, vaste telefonie, faxen, e-mails, chatrooms, het internet of elk ander elektronisch communicatietoestel, bewaard worden.

## 4.5 Privacy in de digitale wereld

Bijna alle ICT-toepassingen laten digitale sporen na. Om te surfen op het internet dient men pakketjes te verzenden die het IP-adres bevatten van de gebruiker. Met behulp van dit IP-adres kan bepaald worden wie deze persoon is, wat zijn interesses en hobby's zijn, naar welke websites hij dagelijks surft, enz. Ook het gebruik van een credit card laat sporen na, welke persoon op welke plaats geld afgehaald heeft, hoeveel geld, ... . Zelfs het gebruik van GSM laat digitale sporen na. De operator dient immers ten alle tijde te weten waar een persoon die aangesloten is bij die bepaalde operator zich bevindt. Hierdoor wordt de geografische positie van deze persoon bijgehouden. Deze data moet opgeslagen worden. Als een operator niet weet waar een klant zich bevindt kan deze niet opgebeld worden, zonder IP-adres kan je niet surfen op het internet, ... . Men zou kunnen aanvoeren dat deze data verwijderd moet worden van zodra deze niet meer nodig is, maar dit kan vaak niet. Om een goede werking van de service te garanderen dient men data op te slaan. Voor facturatie naar de klant toe dient men vaak gedetailleerde gegevens te voorzien (vb. op welk tijdstip getelefoneerd naar wie). Maar ook voor te voldoen aan bepaalde wetten dient men data op te slaan (vb. als de politie wil nagaan vanwaar verdachte X getelefoneerd heeft op dag Y). En met de blijvende daling van de prijzen van harde schijven is de kost voor het opslaan van data ook zeer laag waardoor data vaak een lange periode bijgehouden wordt.

Een voorbeeld van digitale sporen die nagelaten worden en het misbruik hiervan is het AOL-dataschandaal. Op 4 augustus 2006 gaf AOL Research een tekstbestand vrij op één van hun websites. Dit tekstbestand bevatte twintig miljoen zoektermen van meer dan 650 000 gebruikers over een periode van drie maanden. Deze tekst was bedoeld voor onderzoeksdoeleinden. AOL verklaarde dat de data geen informatie bevatte waarmee een persoon geïdentificeerd kon worden. Maar bepaalde zoektermen bevatten wel zulke informatie. Zo waren er bijvoorbeeld bepaalde gebruikers die hun eigen naam hadden ingevoerd om te zien wat voor informatie er over zichzelf te vinden was op het internet. Maar niet alleen hun eigen naam, maar ook hun social security nummer, hun adres etc. In dat tekstbestand werd elke gebruiker geïdentificeerd door een unieke sleutel, zodat men per gebruiker kon opzoeken welke zoektermen deze allemaal ingevoerd had. De New York Times wou testen of het desalniettemin mogelijk was om gebruikers te identificeren. Ze slaagden er in een individu te lokaliseren door ook gebruik te maken van telefoonboeken en andere publieke databanken. Gebruiker met zoeknummer 4417749 werd geïdentificeerd als Thelma Arnold, een 62-jarige weduwe uit de staat Georgia. Hierdoor kwam er een publieke discussie over de ethische gevolgen van het gebruik van deze data. AOL gaf toe dat het fout was en op 7 augustus haalde het

bedrijf het tekstbestand van het internet, maar ondertussen was het al over verschillende websites verspreid en gepubliceerd. Het kan zelfs nog steeds gedownload worden.

Het is dus duidelijk dat het ook mogelijk is om personen te identificeren zonder specifieke data over de naam of woonplaats van die persoon. Bij het gebruik van data gegenereerd door een GSM-netwerk moeten we dus ook zeer goed oppassen. We zullen het gebruik van GSM data uit twee standpunten benaderen. Als eerste is er het standpunt van de analist. De analist gebruikt de data om bepaalde patronen hierin te ontdekken en is vooral geïnteresseerd in groepen van personen. Het tweede standpunt is het standpunt van de terrorist. De terrorist is slechts geïnteresseerd in de data als hij ze kan gebruiken voor criminele doeleinden. Hij is niet zozeer geïnteresseerd in grote groepen van personen, maar vooral in één bepaalde persoon (vb. bekend persoon zoals de president) of in een klein groepje van personen.

Analyse van data levert vaak zeer nuttige kennis op. Diensten worden verbeterd en worden zo efficiënter. Hierdoor gaat de levenswijze van de mensen omhoog. Dus er moet een manier gevonden worden zodat men gebruik kan maken van data zonder dat er een mogelijkheid bestaat dat de persoonlijke privacy van bepaalde personen geschonden wordt. Wanneer we nu specifiek geografische data beschouwen, meer bepaald de geografische coördinaten van plaatsen waar iemand geweest is, kunnen we vaak achterhalen om welke persoon het gaat. Als een bepaald persoon dagelijks van plaats A naar plaats B gaat en omgekeerd, kunnen we met relatief grote zekerheid veronderstellen dat deze persoon op plaats A woont en werkt op plaats B. Als we weten dat op plaats C een groot winkelcentrum is, en een bepaald persoon gaat hier elke zaterdag naar toe, weten we waar en wanneer deze persoon zijn of haar inkopen doet. Hoe meer data we hebben, hoe meer informatie we kunnen afleiden over het leven van die bepaalde persoon en zo bepalen wie deze persoon is.

Toegepast op het voorbeeld van de analist en de terrorist willen we bereiken dat de analist zijn werk kan doen en de data kan analyseren zonder dat de terrorist door gebruik te maken van diezelfde data in staat is om de persoonlijke privacy van iemand te schenden. Een eerste stap die we kunnen uitvoeren is de data anoniem maken. Een terrorist wenst meer te weten te komen over een bepaalde persoon, terwijl de analist tevreden is met anonieme data. In praktijk komt dit neer op het vervangen van namen, telefoonnummers en adressen en het toevoegen van willekeurige pseudoniemen aan de data. Maar zoals we uit het AOL-dataschandaal geleerd hebben is dit niet voldoende. Door combinatie van verschillende andere gegevens kan soms de persoon bepaald worden van wie deze gegevens zijn zonder dat er expliciete persoonlijke informatie tussen staat. Een tweede stap die we kunnen uitvoeren is data niet afzonderlijk beschikbaar stellen maar in de vorm van verzamelingen van data die hetzelfde voorstelt. De analist heeft deze data toch niet nodig en heeft voldoende aan het feit dat een groep van grootte X een bepaald traject aflegt.

Er moet dus een manier gezocht worden om de privacy van de gebruikers van wie data verzameld wordt te beschermen. Een gekende methode is k-anonimity.

## 4.6 K-anonymity

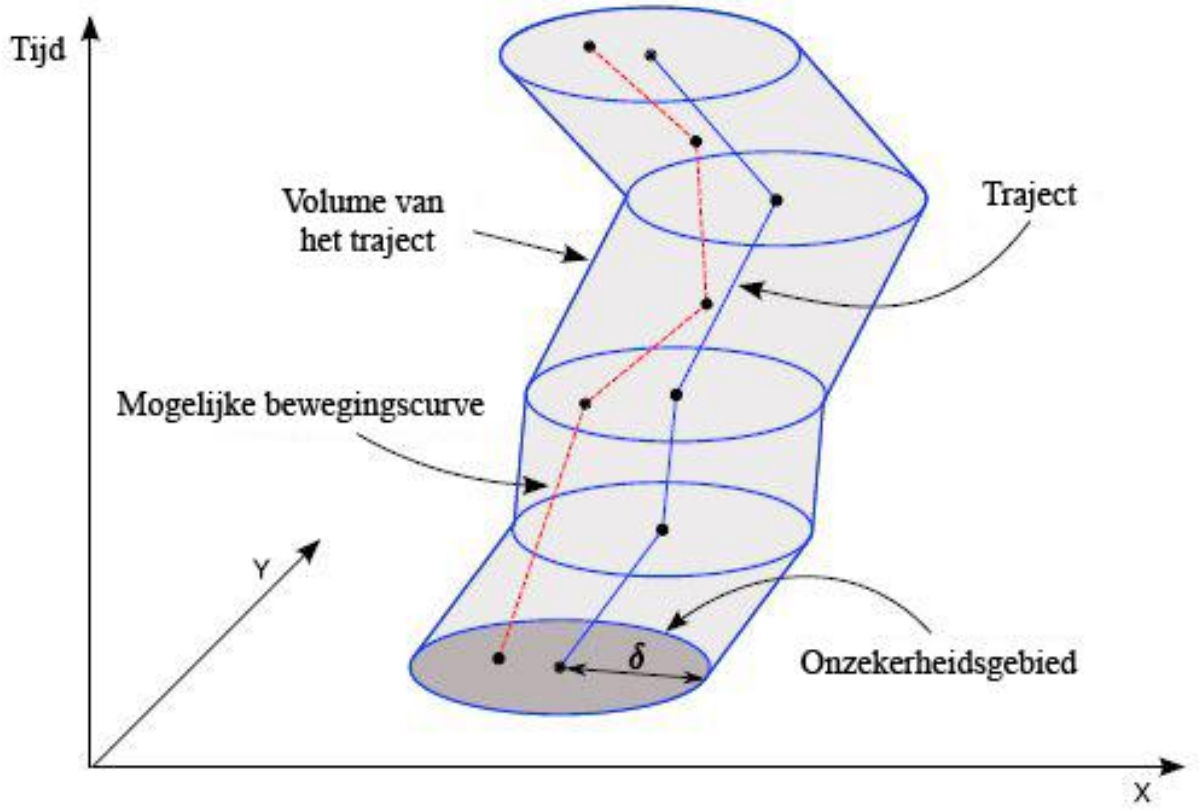
[8] K-anonymity is een manier om ervoor te zorgen dat individuele privacy behouden blijft wanneer data gepubliceerd wordt. Volgens het principe van k-anonymity moet in een verzameling data elk individu niet te onderscheiden zijn van ten minste k-1 andere individu's. Het principe van k-

anonymity moet hier worden toegepast op data die bestaat uit opgenomen trajecten. Omdat deze data onprecies is, zijn deze trajecten geen polyline in de driedimensionale ruimte. In plaats hiervan is het een cilinder waarvan de straal δ de mogelijke afwijking is ten opzichte van de exacte positie. We weten dat het eigenlijke traject zich in deze cilinder bevindt, maar we weten niet exact waar. Andere objecten die in deze cilinder bewegen, leggen een traject af dat niet te onderscheiden is van het huidige traject. Hiervoor wordt het begrip $(k,\delta)$-anonymity gedefinieerd. Omdat de toestellen die positie doorsturen vaak op batterijen werken (GSM, GPS, …), wordt niet continu de positie doorgestuurd. De positie wordt alleen doorgestuurd als op een tijdstip t de eigenlijke positie meer dan een factor voor onzekerheid δ verschilt van de voorspelde positie.

Wanneer we alleen rekening houden met de ruimtelijke informatie voor privacy is anonimiteit vereist op twee verschillende manieren. Als eerste is er *ubiquity* dat er op let dat een object minimaal k verschillende gebieden moet bezoeken. Ten tweede is er *congestion* dat er voor zorgt dat in een gebied ten minste k objecten aanwezig zijn. Hoge *ubiquity* zorgt voor anonimiteit van plaats van elk object en hoge *congestion* zorgt voor anonimiteit van plaats van locale objecten in een bepaald gebied.

### 4.6.1 Definitie: Onzeker Traject

Een traject van een bewegend object is een polyline in drie-dimensionale ruimte voorgesteld door een opeenvolging van spatio-temporele punten: $(x_1, y_1, t_1),(x_2, y_2, z_2)\ldots(x_n, y_n, t_n)(t_1 < t_2 < \ldots < t_n)$. Gedurende het tijdssegment $[t_i, t_{i+1}]$ wordt verondersteld dat het object beweegt met een constante snelheid over een rechte lijn van $(x_i, y_i)$ naar $(x_{i+1}, y_{i+1})$. Gegeven een traject $\tau$ tussen tijdstip $t_1$ en $t_n$, en een onzekerheidsfactor $\delta$, definieert het paar $\langle\tau, \delta\rangle$ een onzeker traject. Voor elk punt $(x, y, t)$ langsheen het traject $\tau$ is diens onzekerheidsgebied de horizontale schijf met straal $\delta$ en met het punt $(x, y, t)$ als centrum. Hierbij is $(x, y)$ de verwachte locatie op tijdstip $t \in [t_1, t_n]$. Het volume van het traject $\langle\tau, \delta\rangle$, genoteerd als $Vol(\tau, \delta)$, is de unie van al zulke schijven voor alle $t \in [t_1, t_n]$. Een mogelijke bewegingscurve van $\tau$ is elke continue functie $f_{PMC}\tau : Time \rightarrow \mathbb{R}^2$ gedefinieerd op het interval $[t_1, t_n]$ zo dat voor elke $t \in [t_1, t_n]$ het spatio-temporele punt $(f_{pmc}(t), t)$ in het onzekerheidsgebied op tijdstip $t$ ligt. Met andere woorden $f_{pmc}(t) \subset Vol(\tau, \delta)$.




Figuur 2.1 Een onzeker traject

Op figuur 4.1 kunnen we duidelijk zien dat twee trajecten niet te onderscheiden zijn als ze in hetzelfde volume liggen. Dit wil zeggen dat ze bijna dezelfde route volgen.

### 4.6.2 Definitie: Co-localisatie

Twee trajecten $\tau_1$, $\tau_2$ gedefinieerd in $[t_1, t_n]$ zijn geco-localiseerd ten opzichte van $\delta$ als en slechts als voor elk punt $(x_1, y_1, t)$ in $\tau_1$ en $(x_2, y_2, t)$ in $\tau_2$ met $t \in [t_1, t_n]$ geldt dat $Dist((x_1, y_1), (x_2, y_2)) \leq \delta$. Hierin is *Dist* de euclidische afstand: $Dist((x_1, y_1), (x_2, y_2)) = \sqrt{((x_1-x_2)^2+(y_1-y_2)^2)}$.

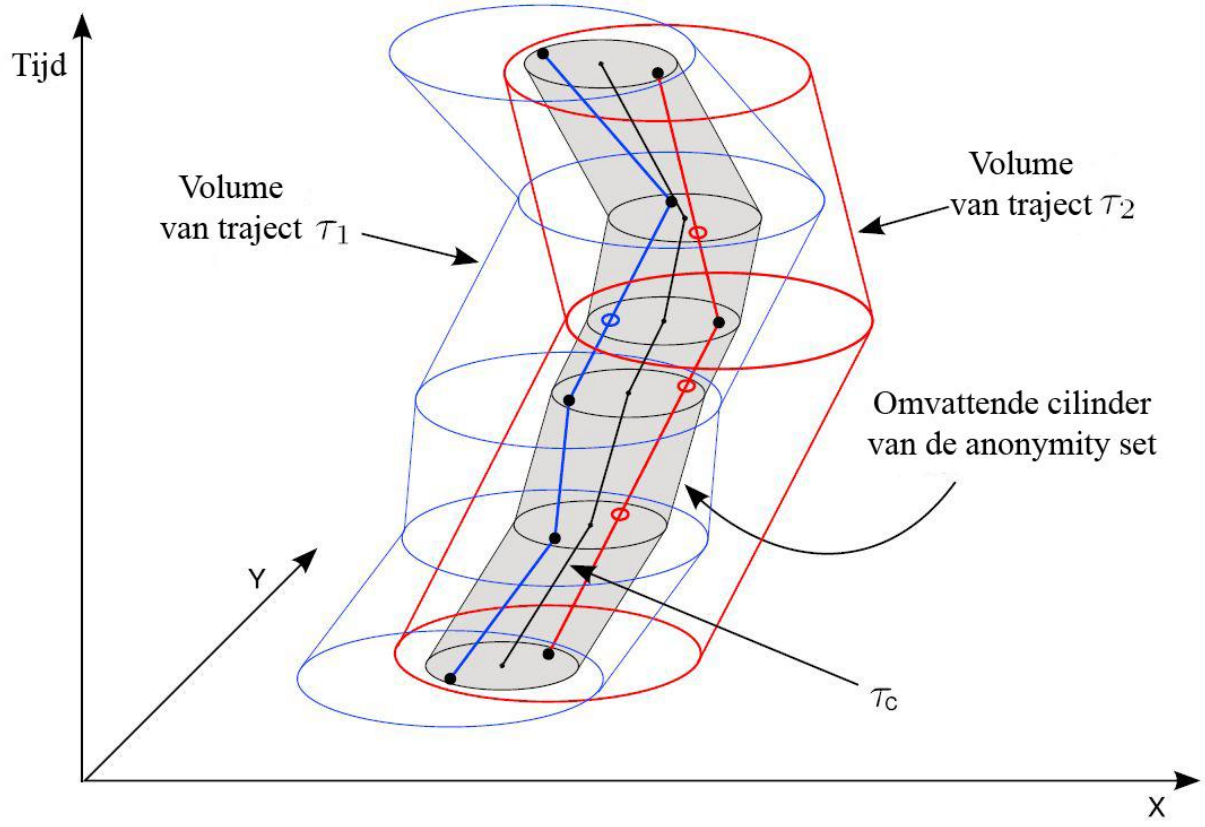
Met andere woorden $Coloc(\tau_1, \tau_2) \Leftrightarrow \tau_1 \subset Vol(\tau_2, \delta) \Leftrightarrow \tau_2 \subset Vol(\tau_1, \delta)$

### 4.6.3 Definitie: Anonymity set van trajecten

Gegeven een bovengrens $\delta$ voor de onzekerheid van positie en een bovengrens k voor de anonimiteit, een set $S$ van trajecten is een $(k, \delta)$-anonymity set als en slechts als $|S| \geq k$ en $\forall\, \tau_1, \tau_2 \in S: Coloc_\delta(\tau_i, \tau_j)$.

Stelling:
Een set $S$ van trajecten met $|S| \geq k$ is een $(k,\delta)$-anonymity set als en slechts als er een traject $\tau_c$ bestaat zodat alle trajecten in $S$ mogelijke bewegingscurven zijn van $\tau_c$ binnen een straal $\delta/2$ van onzekerheid.
Gegeven een $(k,\delta)$-anonymity set $S$, dan wordt het traject $\tau_c$ bekomen door voor elke $t \in [t_1, t_n]$ het punt (x, y) te nemen dat het centrum is van de minimale cirkel die alle punten op tijd $t$ van alle trajecten in $S$ omvat.

Hieruit volgt dat een anonymity set van trajecten omvat kan worden door een cilinder met straal $\delta/2$.




Figuur 4.2 Anonymity set van 2 trajecten

### 4.6.4 Technieken voor anonimiteit van trajecten

Er bestaan verschillende technieken die ervoor zorgen dat anonimiteit van trajecten gegarandeerd wordt. De basistechnieken zijn *generalization* en *suppression*, maar er is ook nog de *condensation* benadering en de *space translation*.

Zoals we hierboven gedefinieerd hebben, moeten twee trajecten die geco-localiseerd moeten worden gedefinieerd zijn over hetzelfde tijdsinterval. Nu is het een de echte wereld heel ongewoon dat twee trajecten beginnen en eindigen op exact hetzelfde tijdstip, maar dit kan opgelost worden door kleine tijdsverschillen toe te laten. Algemener houdt dit in dat er informatie verloren gaat. Dit is nodig om $(k,\delta)$-anonymity te bereiken. Gegeven een dataset van trajecten $D$ en een tijdsinterval $T$, dan is $D_T = \{\tau \in D | \tau \text{ is exact gedefinieerd in } T\}$.

#### *4.6.4.1 Generalization van trajecten*

Het principe van *generalization* houdt in dat attributen die weergegeven worden door echte waardes vervangen worden door minder specifieke doch consistente waardes. Bijvoorbeeld een punt met coördinaat (x,y) wordt vervangen door een gebied dat het punt (x,y) bevat. De niveaus van *generalization* zijn meestal vooraf gedefinieerd, zodat de keuze van welke *generalization* te

gebruiken de kwaliteit en bruikbaarheid van de data sterk beïnvloedt. Een slechte keuze kan resulteren in een slechte kwaliteit van data omdat er meer informatie verloren gaat dan nodig.

#### *4.6.4.2 Suppression van trajecten*

*Suppression* toepassen op een set van trajecten $D_T$ is redelijk eenvoudig. Alle trajecten in $D_T$ die niet tot een anonymity set behoren moeten verwijderd worden. Het grote nadeel van *suppression* is dat de hoeveelheid data en dus de grootte van de database aangepast wordt. Hierdoor gaat er te veel informatie verloren. Maar wanneer we *suppression* uitvoeren in combinatie met andere technieken kan deze wel efficiënt zijn. Door het verwijderen van outliers zal de kwaliteit van de data immers alleen maar toenemen.

#### *4.6.4.3 Condensation*

*Condensation* heeft als doel het behouden van de correlatie van data tussen de attributen onderling. De eerste stap in *condensation* is het opdelen van de originele data in clusters met exact k elementen. In een tweede stap wordt er voor elke cluster een set van k valse elementen gegenereerd die de verdeling en covariantie benadert van de oorspronkelijke cluster. Zo kunnen er geldige data mining modellen gebouwd kunnen worden van de gereconstrueerde data zonder een significant verlies van nauwkeurigheid.

#### *4.6.4.4 Space translation*

Het principe van *space translation* houdt in dat sommige punten van trajecten verplaatst worden van hun oorspronkelijke positie naar een nieuwe positie. Het doel is ($k,\delta$)-anonymity te bekomen door originele en verplaatste trajecten zo gelijk mogelijk te houden.

### 4.6.5 Kritiek op K-anonymity

K-anonymity is niet waterdicht. Beschouw het volgende voorbeeld. Er wordt gebruik gemaakt van k-anonymity bij personen die besmet zijn met het HIV-virus. Stel nu dat een bepaalde persoon tot deze groep behoort. Het probleem met k-anonymity is dat het toelaat die persoon te identificeren als een van de k individuen met HIV. De kern van het probleem is het feit dat de waarde van *k* sterk afhankelijk is van de private attributen. Met attributen zoals ziekte, waar HIV is een mogelijkheid, moet k bijna gelijk zijn aan de grootte van de populatie. Door het invoeren van diversiteit kunnen we dit voorkomen. In dit geval moet een privaat attribuut in een k-anonieme set ten minste *n* verschillende waardes hebben. Zo kan men in het voorbeeld met HIV niet weten of die persoon HIV heeft of een andere ziekte. Maar opnieuw weten we dat deze persoon een van de *k* individuen is die allemaal een bepaalde ziekte hebben, wat opnieuw de privacy schendt. Het is immers niet gewenst dat bijvoorbeeld een verzekeringsmaatschappij kan zien dat een persoon een terminale ziekte heeft.

## 4.7 Besluit

Privacy is een zeer belangrijke factor wanneer we werken met gegevens van personen. Zoals we gezien hebben in het AOL-dataschandaal moet er zeer voorzichtig mee omgegaan worden. K-anonimity biedt een manier om privacy in te leiden in het gebruik van data van personen. In het volgende hoofdstuk zal er dieper ingegaan worden op privacy tijdens het minen van data en de link met uncertainty.

# Hoofdstuk 5: Uncertainty en privacy

## 5.1 Uncertainty bij mining

[25] Een belangrijke vereiste die te weinig aandacht krijgt in het *Knowlegde Discovery and Data mining* proces (KDD) is het behandelen van uncertainty in technieken zoals clustering, association rule extraction en classification. In de grote meerderheid van data mining technieken zijn de categorieën die verkregen worden door clustering scherp afgelijnd, zodat de data ondergebracht wordt in één van een verzameling van categorieën. Er zijn geen voorgedefinieerde klassen en de meeste clustering algoritmes hangen af van veronderstellingen en gokken om deelgroepen te definiëren in een verzameling data. Ze veronderstellen ook dat het aantal clusters dat voortgebracht wordt door een algoritme het beste is voor die specifieke data set waarop het algoritme uitgevoerd wordt. Een gevolg hiervan is dat de uiteindelijke clusters nog geëvalueerd moeten worden. We bespreken kort enkele problemen en hun gevolgen.

### 5.1.1 De clusters overlappen niet

Clusters worden scherp afgelijnd waardoor elk object maximaal tot één cluster kan behoren. Hierdoor gaans soms interessante databasetupels verloren omdat ze tot geen enkele cluster behoren. In de echte wereld waar een waarde in meerdere categorieën kan thuishoren is dit nogal onwaarschijnlijk. Zo kan een man die 1m82 groot is zowel gedefinieerd worden als gemiddeld en als groot. Het begrip groot is immers relatief.

### 5.1.2 Alle waardes worden gelijk behandeld in het classificatie proces

Bij het classificeren worden database waardes in de beschikbare categorieën ondergebracht. Een waarde behoort tot een bepaalde categorie of behoort niet tot deze categorie. Wanneer we mannen onderverdelen in twee groepen, groot en klein, dan zit een man van 1m 82 in dezelfde groep als een man van 1m 99. Toch voldoet de tweede persoon meer aan het criterium ‘groot’ dan de eerste persoon. Deze kennis kan niet opgedaan worden wanneer gebruik gemaakt wordt van huidige classificatie schema’s.

### 5.1.3 Weinig aandacht voor kwaliteit van clusters

De meeste algoritmes voor clustering delen een verzameling data op in een aantal clusters gebaseerd op enkele parameters zoals het gewenste aantal clusters, het minimum aantal objecten in een cluster, de diameter van een cluster, enzoverder. Ze zoeken voor de beste clusters volgens een aantal goed gedefinieerde criteria. Als de juiste waardes niet aan de parameters van deze algoritmes worden toegekend, zullen deze algoritmes een opdeling geven die niet optimaal is voor de gegeven verzameling data. Er is onderzoek verricht naar het probleem van te beslissen hoeveel clusters er gezocht moeten worden alsook naar het evalueren van de gevonden clusters. Maar in de praktijk wordt dit niet gebruikt door de clustering algoritmes.

### 5.1.4 De gevonden regels kunnen kennis verbergen

Een regel is een implicatie van de vorm A->B met A en B groepen van attributen of groepen van categorieën. Alle verzamelingen van waardes die behoren tot categorie A of categorie B dragen evenveel bij tot de sterkte van de regel en elke tupel draagt ook zijn steentje bij tot de regel. Hieruit kan afgeleid worden dat de gevonden regels niet het verschil in sterkte van de associatie weergeven in een tupel basis. Het is duidelijk dat er interessante kennis is in de classificatie van waardes die niet

gevonden wordt gedurende het data mining proces. Dit is zowel te wijten aan het feit dat uncertainty niet in acht genomen is als aan het vaste aantal clusters.

### 5.1.5 Voorgestelde benadering

Omdat de ontdekking en het gebruik van uncertainty en de evaluatie van data mining resultaten belangrijk is, wordt er een benadering voorgesteld die hier rekening mee houdt.

In een eerste stap worden clusters gedefinieerd aan de hand van de initiële categorieën voor een specifieke verzameling data. Hier voor kunnen algemeen gekende algoritmes gebruikt worden.

In de tweede stap wordt een clustering algoritme uitgevoerd met verschillende parameters en wordt aan de hand van enkele criteria de beste gekozen. De beste opdeling van de data zal beter voldoen aan de vooraf gedefinieerde criteria. Dit criterium wordt voorgesteld door een index, de kwaliteitsindex. De definitie van deze kwaliteitsindex is gebaseerd op de twee fundamentele criteria voor kwaliteit van clustering, namelijk compacte clusters en goed gescheiden van elkaar.

Fuzzy clustering algoritmes definiëren clusters en berekenen de graad van lidmaatschap van elke waarde tot de clusters. Maar de meeste clustering algoritmes kennen waardes volledig toe aan één bepaalde cluster. In een derde stap worden er uncertainty kenmerken toegevoegd door geschikte toekenningsfuncties toe te kennen aan de clusters.

In een vierde stap worden de waardes van niet-categorische attributen ($A_i$) van een verzameling data opgedeeld in categorieën volgens een verzameling categorieën L = $\{l_i\}$ en functies. Deze functies werden gedefinieerd in het clusteringsproces. Het resultaat hiervan is een verzameling M = $\{l_i(t_k.A_i)\}$. Elk element van deze verzameling stelt de confidence voor dat de specifieke waarde $t_k.A_i$ behoort tot de verzameling met label $l_i$.

In de vijfde stap wordt de verzameling data getransformeerd in *classification beliefs* en opgeslagen in een *Classification Value Space* (CVS). Dit is een kubus waarvan de cellen de graad van overtuiging bevatten voor de classificatie van de waardes van de attributen.

In de voorlaatste stap maken we gebruik van de informatie opgeslagen in de CVS. Deze informatie kan helpen bij het maken van beslissingen en het evalueren van het toegepaste classificatie model. Dit kan helpen om te beslissen of het initiële clusteringsschema geherdefinieerd moet worden.

In de laatste stap worden de association rules gemined. Hierna kan opnieuw de geldigheid van de relaties vergeleken worden in de verschillende verzamelingen van data.

## 5.2 Privacy bij mining

[26] Er zijn verschillende technieken die data behoeden voor schending van de privacy. Ze kunnen geclassificeerd worden aan de hand van de volgende eigenschappen:

- verdeling van de data
- aanpassing van de data
- gebruikte data mining algoritme
- verbergen van data of van regels
- handhaving van privacy

De eerste eigenschap gaat over de mate waarin de data verdeeld is. Er zijn benaderingen ontwikkeld voor gecentraliseerde data, terwijl andere ontwikkeld zijn voor sterk verdeelde data. Deze laatste groep kan onderverdeeld worden in horizontale verdeeldheid en verticale verdeeldheid van data. Een horizontale verdeeldheid wijst op de gevallen waarin verschillende database records op verschillende plaatsen voorkomen, terwijl verticale verdeeldheid wijst op de gevallen waar alle waardes van verschillede attributen op verschillende plaatsen voorkomen.

De tweede eigenschap handelt over de aanpassing van de data. In het algemeen wordt data aangepast om originele waardes aan te passen die vrijgegeven moeten worden aan het publiek. Het is belangrijk dat de techniek gebruikt om data aan te passen in overeenkomst is met de privacy policy van een organisatie. Manieren van aanpassing van data houden in:

- *perturbation* (storing), die bekomen wordt door de waarde van een attribuut te vervangen door een nieuwe waarde
- *blocking* (blokering), wat overeenkomt met het vervangen van een bestaand attribuut door een '?'
- *aggregation/merging* (samenvoegen), de combinatie van verschillende waardes in een ruwere categorie
- *sampling*, wat inhoudt dat er alleen data wordt vrijgegeven voor een deel van de populatie

De derde eigenschap heeft te maken met het gebruikte data mining algoritme. Dit is niet geweten op voorhand, maar het vergemakkelijkt de analyse en het design van het algoritme om data te verbergen.

De vierde eigenschap handelt over het feit of naakte data of verzamelde data verborgen moet worden. De complexiteit voor het verbergen van verzamelde data in de vorm van regels is natuurlijk groter, en daarom bestaan er hiervoor bijna alleen heuristieken. Door de vermindering van de hoeveelheid publieke informatie worden er zwakkere regels afgeleid waardoor vertrouwelijke waardes niet afgeleid worden.

De laatste eigenschap is de belangrijkste en heeft te maken met de techniek gebruikt voor de selectieve wijziging van de data.

- Heuristiek-gebaseerde technieken: deze passen alleen de geselecteerde waardes aan in plaats van alle waardes
- Cryptografie-gebaseerde technieken: deze technieken zorgen ervoor dat berekeningen veilig zijn als op het eind van de berekening niemand iets meer weet dan diens eigen input en de resultaten van de berekening
- Reconstrucite-gebaseerde technieken: deze technieken reconstrueren de originele verdeling van de data

## 5.3 Uncertainty en privacy bij datacollectie

Ook bij het verzamelen van data moet er rekening gehouden worden met de privacy van de personen van wie we data verzamelen. Door bijvoorbeeld geen mapmatching uit te voeren, wordt er een zekere graad van uncertainty behouden. Hierdoor is de exacte positie ook niet gekend waardoor er minder kans is om de privacy van de personen te schenden.

## 5.4 Uncertainty vs privacy

Het is duidelijk dat er een verband ligt tussen uncertainty en privacy. Wanneer de locatie van een object onzeker is, wordt het ook moeilijker om de privacy van de persoon die dit object voorstelt te schenden. Als de locatie onzeker is, is het immers moeilijker om deze in verband te brengen met andere gegevens om zo de persoon te identificeren. Maar de omgekeerde redenering is niet waar. Als er weinig uncertainty is, wil dit niet a priori zeggen dat de privacy van de individu's daadwerkelijk geschonden wordt. En veel uncertainty houdt ook niet in dat de privacy niet kan geschonden worden. Bijvoorbeeld als er een database van trajecten bestaat van huis-werk verkeer, is door middel van uncertainty het veel moeilijk om te ontdekken wie de persoon is achter één bepaald traject. De exacte woonplaats bijvoorbeeld is niet bekend.

# Hoofdstuk 6: Implementatie

## 6.1 Inleiding

We hebben een applicatie gemaakt die als input GPS-punten verwacht en ervoor zorgt dat deze gematched worden op een stratennetwerk. De output die gegenereerd wordt is een traject dat zo goed mogelijk het oorspronkelijk traject tracht te benaderen. Deze applicatie is geprogrammeerd in de taal Java. De keuze voor Java was voor mij voor de hand liggend. Een Java programma kan op elke computer runnen zonder dat er extra bestanden geïnstalleerd moeten worden. Ook is het relatief eenvoudig om met Java een database aan te spreken. Voor de keuze van DBMS is er geopteerd voor postgreSQL versie 8.3 [35] in combinatie met de postGIS versie 1.3.3 extensie voor de ruimtelijke data [36]. Dit omdat er voor de Gent-data verder kon gewerkt worden op de database van Kristof Gheys. De Gent-data is data die verzameld werd door de politie van Gent. Zij startten bij elke oproep de registratie van hun afgelegde weg door de GPS-punten om de tien meter op te slaan. In het begin doken er wel wat problemen op om hiermee te werken aangezien ik nog nooit eerder met een postgreSQL database gewerkt had, maar dankzij de goede documentatie en de talrijke fora op Internet zijn alle problemen oplost geraakt en werd het werken hiermee vergemakkelijkt. Om deze data te bekijken en enkele testquerries te proberen is er gebruik gemaakt van DBVisualizer versie 6.5 [37].

## 6.2 Hardware en software

### 6.2.1 Hardware

Alle code is geïmplementeerd op een laptop, een Compaq 8510p met volgende specificaties:

- Processor: Core(TM)2 Duo CPU T8100 Intel processor van 2,1 GHz;
- Geheugen: 4 GB DDR RAM;
- Harde schijf: 160Gb @ 7200 RPM.

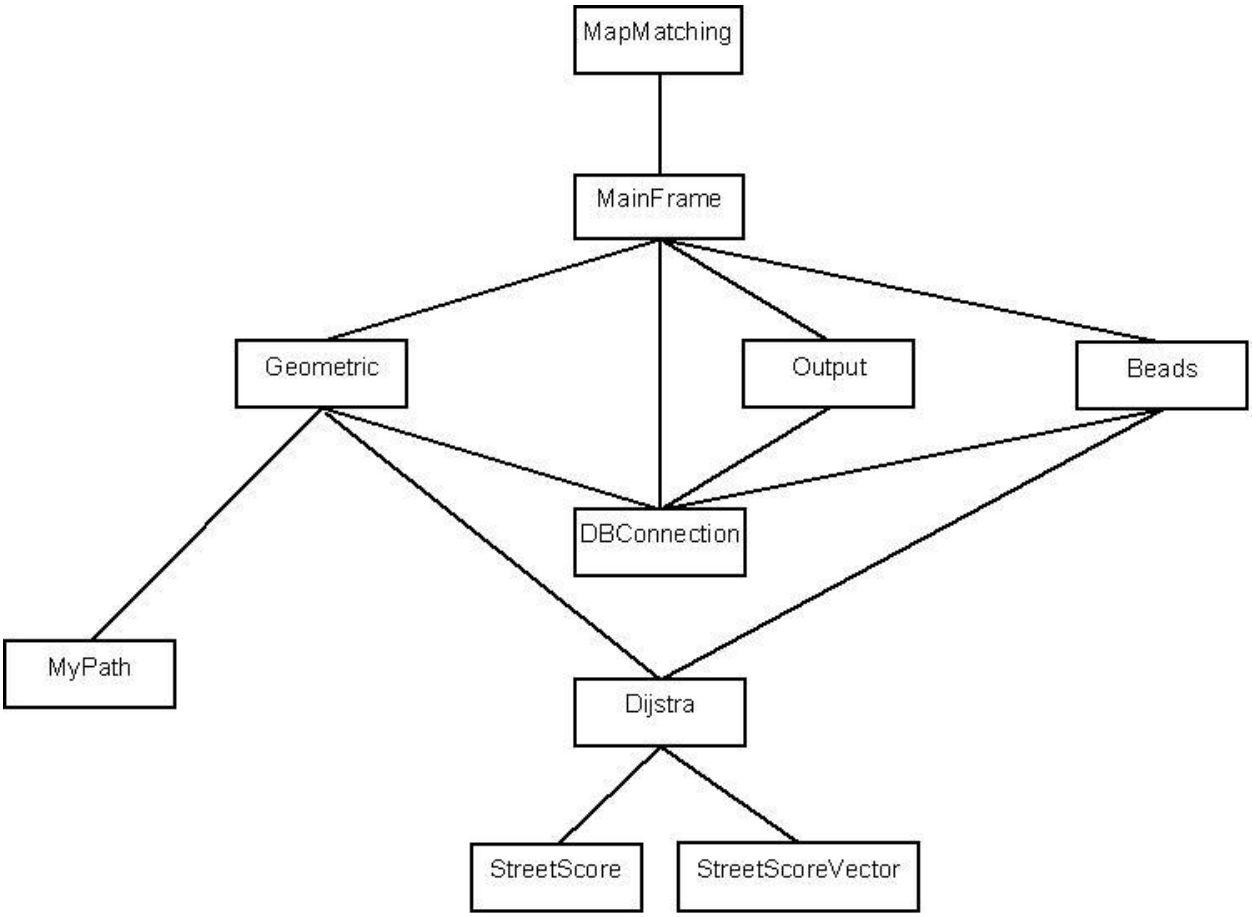
### 6.2.2 Software

Het besturingssysteem van de laptop waarop de applicatie geïmplementeerd werd, is Microsoft Windows XP. De code is geschreven in Eclipse [38], een zeer goed en krachtig programma voor het schrijven en compilen van Java code. Voor het design van de Grafische User Interface (GUI), is er gebruik gemaakt van Netbeans [39]. Dit omdat dit een zeer goede en overzichtelijke GUI designer is en in Eclipse standaard geen GUI designer aanwezig is. Deze GUI is ontworpen in Swing, een extensie van Java speciaal voor het maken van GUI's.

Zoals in de inleiding vermeld is er gebruik gemaakt van postgreSQL met de postGIS extensie voor het opzetten en beheer van een database. De data zelf is opgeslaan in shape-files. Dit is een handig formaat voor het opslaan van geografische data. Daarom is ervoor gekozen om de output te genereren in de vorm van shape-files. De data van het stratennetwerk zijn geproduceerd door Tele Atlas (zie hoofdstuk 1 sectie 1.5.3). Voor het bekijken van de data en de gegenereerde bestanden werd er gebruik gemaakt van TatukGIS [40], een open source applicatie voor het bekijken van shape-files.

## 6.3 Packages

In deze sectie bespreken we kort de implementatie van de applicatie met als doel een eventuele gebruiker meer inzicht te geven in de implementatie van de belangrijkste onderdelen. In figuur 6.1 wordt de samenhang tussen de belangrijkste klasses getoond.




Figuur 6.1: Samenhang tussen de belangrijkste klasses

Hieronder volgt een kort overzicht van de packages. Van elk package worden de klasses besproken tesamen met hun belangrijkste functies en membervariabelen. De gebruikte packages zijn:

- Database;
- GPS;
- GUI;
- Graph;
- Output;
- Main.

### 6.3.1 Package Database

Het package Database bevat alles wat te maken heeft met de onderliggende database. Het bevat slechts één klasse, namelijk DB*Connection.java.*

#### *6.3.1.1 DBConnection.java*

De klasse DB*Connection.java* staat in voor de communicatie tussen de applicatie en de database. De klasse bevat een aantal membervariabelen waarin de naam van de tabellen, van de database, de

gebruikersnaam en het paswoord worden opgeslaan. Ook bevat deze klasse een aantal membervariabelen voor een connectie met een database: een *Statement*, een *PreparedStatement*, een *Connection* en *DataBaseMetaData*. Een belangrijke parameter die ook als membervariabele in deze klasse opgeslaan wordt is de maximale snelheid waarmee een bead berekend wordt (zie hoofdstuk 3.2). Deze zou eventueel vervangen kunnen worden door een dynamische variabele die de snelheidslimiet van het wegennetwerk neemt. Echter, bij het berekenen van de beads weten we niet welke straten er allemaal in de bead liggen, en weten we dus ook niet wat de snelheidslimieten zijn die op deze wegen gelden. Othman schrijft in [44] dat wanneer de GPS punten al op het stratennetwerk liggen, deze snelheid wel dynamisch gemaakt kan worden, maar dit is wel zeer rekenintensief.

De constructor van deze klasse heeft slechts één parameter, namelijk *table*. Als deze de waarde 'GENT' meekrijgt gaan we werken op de Gent-dataset. Wanneer deze de waarde 'MILAAN' meekrijgt werken we op de Milaan-dataset. Een nieuwe dataset kan zo eenvoudig toegevoegd worden. De overige functies in deze klasse zijn allemaal functies die gebruikt worden om gegevens uit de database te halen.

### 6.3.2 Package GPS

De klassen in het package GPS hebben allemaal te maken met het gebruik van GPS data in de applicatie. Ze vormen datastructuren om data uit de database in op te slaan. De volgende klassen zitten in het GPS package:

- GPSProperties.java;
- GPSPoint.java;
- StreetInfo.java;
- StreetScore.java;
- StreetScoreVector.java;
- TimestampStreet.java;
- TimestampStreetTimeComparator.java.

#### *6.3.2.1 GPSProperties.java*

De klasse *GPSProperties.java* wordt gebruikt om de gegevens die nodig zijn om de verschillende trajecten weer te geven in de tabel op te slaan. Deze klasse heeft drie membervariabelen, namelijk een *gid* die overeenkomt met de key in de tabel, een *routeId* waarin de identifier voor het traject opgeslaan wordt en *count* waarin opgeslaan wordt uit hoeveel punten het gegeven traject bestaat.

#### *6.3.2.2 GPSPoint.java*

De klasse *GPSPoint.java* wordt gebruikt om een punt uit een traject op te slaan. Deze klasse heeft twee membervariabelen, namelijk *point* waarin de (x,y) coördinaten worden opgeslaan en *time* waarin het tijdstip *t* wordt opgeslaan waarop het punt geregistreerd werd.

#### *6.3.2.3 StreetInfo.java*

De klasse *StreetInfo.java* wordt gebruikt om een netwerk van straten in op te slaan. De klasse heeft drie membervariabelen. Een eerste membervariabele is *id* waarin de identifier van een straat opgeslaan wordt. Een tweede membervaraibele is *the_geom* die een *MultiLineString* (een verzameling van 1 of meer lijnstukken die samen één lijn vormen) bevat van de straat in kwestie. Een laatste membervariabele tenslotte is *distance* waarin de lengte van de straat opgeslaan wordt.

#### 6.3.2.4 StreetScore.java & StreetScoreVector.java

De klasse *StreetScore.java* wordt gebruikt om een straat op te slaan met zijn bijhorende score gekregen door het algoritme om het traject te reconstrueren (zie sectie 3.5.2.1 Score calculating algortime). Deze klasse heeft bijgevolg twee membervariabelen, *streetId* en *score*.

Een sterk gerelateerde klasse is *StreetScoreVector.java* die een *Vector* bevat van *StreetScore-objecten*. Deze klasse bevat ook een aantal functies die het makkelijk maken om operaties uit te voeren op zulk een *Vector*.

#### 6.3.2.5 TimestampStreet.java & TimestampStreetTimeComparator.java

De laatste twee klassen van deze package worden gebruikt om een straat op te slaan met het bijhorende tijdstip waarop er iemand langsgekomen is. De klasse *TimestampStreet.java* bevat twee membervariabelen, *streetId* waarin de identifier van de straat opgeslaan wordt en *timestamp* waarin het tijdstip opgeslaan wordt.

De klasse *TimestampStreetTimeComparator.java* wordt gebruikt om twee objecten van het type *TimestampStreet* met elkaar te vergelijken. Dit gebeurt op basis van het tijdstip.

### 6.3.3 Package GUI

Het package GUI bevat twee klassen die nodig zijn om de Grafische User Interface te implementeren. De klasse *MainFrame.java* bevat de implementatie van de GUI. De klasse *CustomTableModel.java* bevat een zelf gedefinieerde tabel die in de GUI gebruikt wordt om de trajecten weer te geven waarop een algoritme kan worden losgelaten.

### 6.3.4 Package Graph

De klassen die te maken hebben met grafen kunnen teruggevonden worden in het Graph package. Enerzijds bestaan zij uit algoritmes om kortste paden te vinden, anderzijds uit het opbouwen van grafen en reduceren van opgebouwde grafen. De volgende klassen zijn terug te vinden in dit package en worden verderop besproken:

- Beads.java;
- Dijkstra.java;
- GeometricAlg.java;
- MyPath.java;
- RoadWeightedData.java;
- RoadWeightedDataComparator.java.

#### 6.3.4.1 Beads.java

De klasse *Beads.java* bevat het algoritme om de GPS punten te mappen naar een stratennetwerk met behulp van space time prisms (zie hoofdstuk 3). Deze klasse bevat drie belangrijke membervariabelen. Een eerste membervariabele is een DB*Connection* die de connectie met de database afhandelt. Een tweede membervariabele is *path*. Dit is een *Vector* waarin de straten van het traject wordt opgeslaan die in de beads liggen. Een derde en laatste membervariabele is *beginEndMode*. Hierin wordt de manier opgeslaan voor het selecteren van de beginstraat en eindstraat (zie hoofdstuk 7 sectie 7.1).

De belangrijkste functie van de klasse *Beads.java* is de functie *computeRoute(int routeId, int meter)* die het gereconstrueerde traject teruggeeft in een *Vector*. Deze functie heeft twee parameters,

*routeId* en *meter*. De parameter *routeId* bevat de identiefier van het traject dat we wensen te reconstrueren. De parameter *meter* wordt gebruikt voor het vergelijken van twee trajecten. Dit getal wil zeggen dat het traject gereconstrueerd moet worden met punten om de *n* meter, dus bij *n = 10* gebruikt het algoritme punten die minimaal 10 meter van elkaar verwijderd liggen.

#### 6.3.4.2 Dijsktra.java

De klasse *Dijkstra.java* bevat het Dijkstra algoritme en het *k-shortest path* algoritme (zoals besproken in sectie 3.5.2.2). De functie die aangeroepen moet worden is de functie *AlgorithmMultiDijkstra(roads, start, end)* en geeft het kortste pad terug in een *Vector*. Deze functie heeft drie parameters: *roads* die de straten bevat die binnen de beads liggen, *start* dat de beginstraat bevat en *end* dat de eindstraat bevat.

#### 6.3.4.3 GeometricAlg.java

De klasse *GeometricAlg.java* bevat het geometrisch algoritme voor het reconstrueren van een traject. De twee membervariabelen van deze klasse zijn DB*Connection* en *table.* DB*Connection* handelt de connectie met de database af en in de variabele *table* wordt opgeslagen op welke dataset we aan het werken zijn (Gent of Milaan).

De belangrijkste functie van deze klasse is de functie *mapToRoadNetwork(routeId, path)* en geeft een *Vector* terug met daarin de gereconstrueerde route. De parameter *routeId* bevat de identifier van het traject en *path* bevat de locatie naar waar de bekomen *shapefile* geëxporteerd moet worden.

#### 6.3.4.4 MyPath.java

De klasse *MyPath.java* wordt gebruikt door het Dijkstra-algoritme bij het berekenen van de kortste route. Deze wordt slechts éénmaal gebruikt voor het preprocessen van straten en hun buren. Wanneer er een nieuwe dataset toegevoegd wordt, preprocessen we eerst het wegennetwerk. We zoeken voor elke straat de straten die snijden met deze straat. Dit vergemakkelijkt het zoeken van een route in het wegennetwerk en vermindert de zoektijd naar een route drastisch.

#### 6.3.4.5 RoadWeightedData.java & RoadWeigthedDataComparator.java

De laatste twee klassen van dit package zijn *RoadWeightedData.java* en *RoadWeightedDataComparator.java.* De klasse *RoadWeightedData.java* wordt gebruikt door de *MyPath.java* klasse voor het opslaan van de straten met hun naburige straten en hun scores. De klasse *RoadWeightedDataComparator.java* bevat de implementatie voor het vergelijken van twee *RoadWeightedData* objecten.

### 6.3.5 Package Output

Het package Output bevat twee klassen, *Output.java en StreamGobbler.java*. De klasse *StreamGobbler.java* wordt gebruikt voor het opvangen van output van externe processen zoals het converteren naar *shapefiles*. De klasse *Output.java* bevat de code voor het produceren van verschillende outputs. Volgende formaten van output zijn voorzien en worden allemaal geëxporteerd naar een shape-file:

- Het klassieke *(x,y)*-formaat;
- Het *(x,y,t)*-formaat;
- Id's van de streetnodes;

- Id's van de streetnodes met het tijdstip *t* waarop men er voorbijgekomen is;
- Id's van de streetsegements.

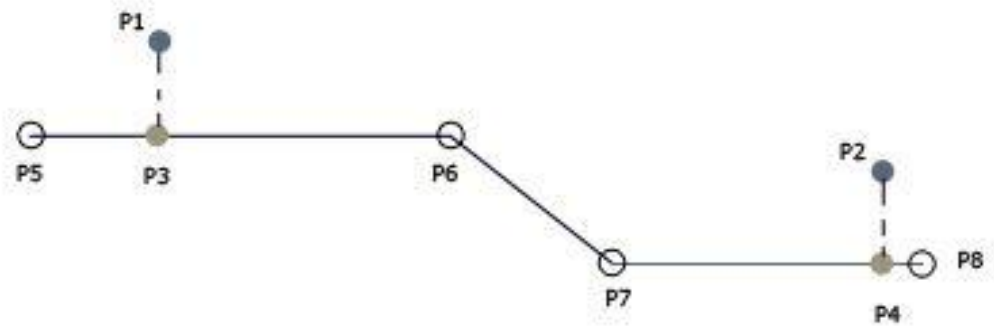
Om te exporteren naar het *(x,y,t)*-formaat projecteren we de GPS punten op de gevonden route. Voor het tijdstip nemen we het tijdstip van het op de route geprojecteerde GPS punt. Voor het berekenen van het tijdstip waarop men voorbijgekomen is in de streetnodes, maken we gebruik van lineaire interpolatie. We interpoleren tussen het punt ervoor en het punt erna om zo een goede benadering te bekomen voor het tijdstip waarop men er voorbijgekomen is. Beschouw figuur 6.2. De GPS-punten zijn P1 en P2. Wanneer we nu het tijdstip willen weten waarop men voorbijgekomen is in punt P6, projecteren we eerst punt P1 en P2 op de route respectievelijk in P3 en P4. Wanneer T1 het tijdstip is waarop voorbijgekomen is in P3, en T2 het tijdstip waarop men voorbijgekomen is in P4, dan berekenen we het tijdstip T6 waarop men voorbijgekomen is in P6 als volgt. We delen de afstand van P3 tot P6 door de afstand van P3 tot P4. Deze factor vermenigvuldigen we met het verschil tussen T4 en T3. Wat we nu bekomen tellen we op bij T3 zodat we T6 bekomen. Concreet, stel T3 = 20 seconden, T4 = 40 seconden, de afstand tussen P3 en P4 gelijk aan 80 meter en de afstand tussen P3 en P6 gelijk aan 20 meter. Dan bekomen we:

$$T6 = \frac{d(P6, P3)}{d(P4, P3)} * (T4 - T3) \; + \; T3$$

$$= \frac{20\ meter}{80\ meter} * (\,40\ sec - 20\ sec) + \; 20\ sec$$

$$= \; 25\ \text{sec}$$

Met *d(x,y)* de afstand tussen het punt *x* en *y*.




Figuur 6.2 Interpolatie van het tijdstip

De constructor bevat vijf parameters en zorgt dat het gewenste formaat geproduceerd wordt:

- *Route*: de gereconstrueerde route;
- *Gpspoints*: de originele GPS-punten waarmee de route gereconstrueerd werd;
- *RouteId*: de identifier van het traject dat gereconstrueerd werd;
- *Type*: het formaat van de output die geproduceerd moet worden;
- *Path*: het path naar waar de output geschreven moet worden;

- *Table*: de tabel waarop gewerkt wordt.

### 6.3.6 Package Main

Het package main bevat slechts één klasse, namelijk *MapMatching.java*. Deze bevat de *main* functie van de applicatie.

## 6.4 GUI

De Grafische User Interface of kortweg GUI is geïmplementeerd in Swing en designed met behulp van Netbeans. De GUI is opgebouwd uit vijf panels:

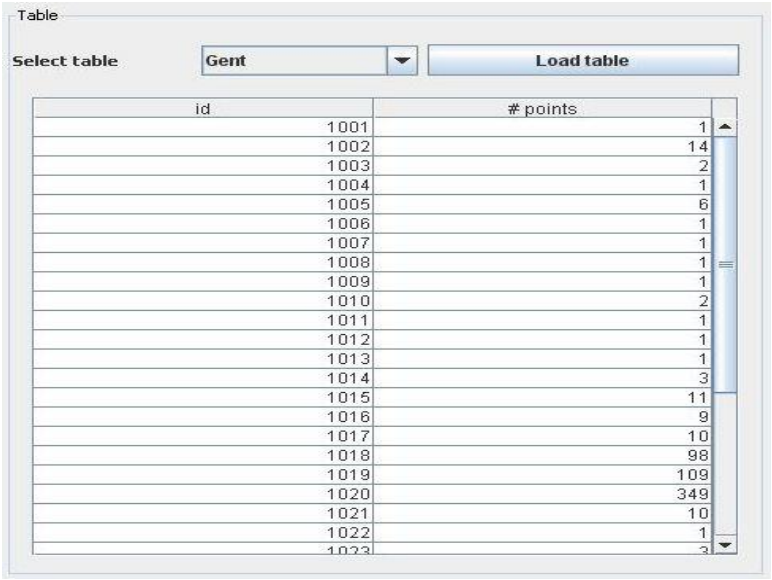
- *Table* panel;
- *MapMatching* panel;
- *Compare* panel;
- *Export points* panel;
- *Output* panel.

Hieronder zullen we kort de verschillende panels overlopen.

### 6.4.1 Het Table panel

Het *Table* panel wordt weergegeven in figuur 6.3. Hier kan de gebruiker kiezen welke tabel hij gebruikt. Voorlopig kan de gebruiker kiezen uit ofwel de Gent dataset of de Milaan dataset. Het is echter mogelijk om relatief eenvoudig een nieuwe dataset toe te voegen. De data van de geselecteerde dataset wordt weergegeven in een tabel. De eerste kolom van de tabel bevat de identifier van het traject, de tweede kolom bevat het aantal GPS punten waaruit het traject bestaat.

Het element dat de gebruiker hier selecteert wordt gebruikt voor het mapmatchen in het *MapMatching* panel.




Figuur 6.3: Het Table panel

### 6.4.2 Het MapMatching panel

Het *MapMatching* panel bevat de opties voor het instellen van het map-matching algoritme en wordt getoond in figuur 6.4. Een eerste optie die de gebruiker krijgt is het kiezen van het te gebruiken algoritme. De gebruiker heeft de keuze uit:

- Geometrisch algoritme;
- Algoritme gebruik makend van beads.

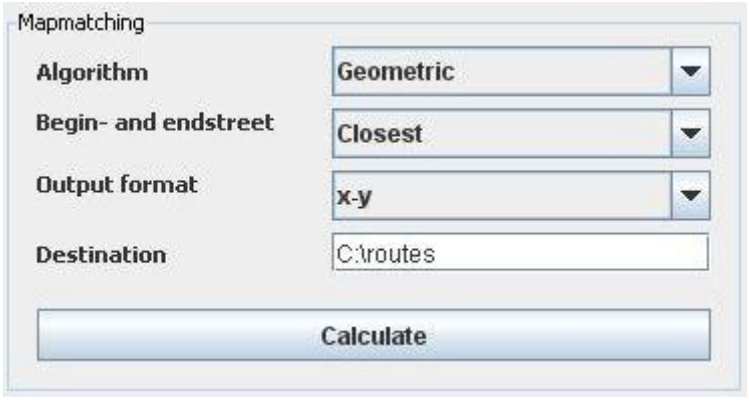
Een tweede optie die de gebruiker kan instellen is de manier waarop de begin- en eindstraat gekozen wordt. Hier heeft de gebruiker de keuze uit de volgende opties:

- *Closest*: de dichtstbijzijnde straat wordt gekozen;
- *Dynamic*: de begin- en eindstraat worden dynamisch gekozen zoals uitgelegd in sectie 7.1.1;
- *Circle*: de begin- en eindstraat worden gekozen op basis van een cirkel rond respectievelijk het begin- en eindpunt (zie sectie 7.1.2 voor meer details).

Een derde optie is het instellen van het output formaat. Hier heeft de gebruiker de keuze uit:

- *x-y:* het klassieke *(x,y)* formaat;
- *x-y-t*: het *(x,y)* formaat met een parameter *t* toegevoegd voor het tijdstip waarop men op de plaats *(x,y)* was;
- *StreetSegment:* de identifiers van de straten van het traject;
- *StreetNode:* de identifiers van de nodes van de straten van het traject;
- *StreetNode with time:* de identifiers van de nodes van de straten van het traject met het tijdstip waarop men hier gepasseerd is.

Een vierde en laatste optie die men kan instellen is de locatie waarnaar de *shapefile* geschreven dient te worden.

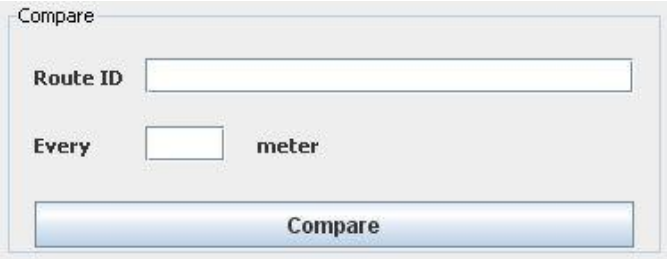



Figuur 6.4: Het MapMatching panel

De gebruiker dient de identifier van het traject dat hij wenst te reconstrueren te selecteren in de table van het *Table* panel.

### 6.4.3 Het Compare panel

Het compare panel, zoals te zien op figuur 6.5, bestaat uit twee velden die de gebruiker kan invullen. Het eerste veld dat de gebruiker dient in te vullen is *Route ID*. Hier moet de gebruiker de identifier van het traject ingeven dat hij wenst te vergelijken. In het tweede veld, *Every X meter*, dient de gebruiker de waarde *X* te specifiëren. Deze waarde wordt gebruikt voor het berekenen van het aantal punten dat de gebruiker wenst te gebruiken voor de map-matching. Zo zal bij een waarde 10 het algoritme om de 10 meter een punt van het traject selecteren en gebruiken voor de map-matching uit te voeren.




Figuur 6.5: Het Compare panel

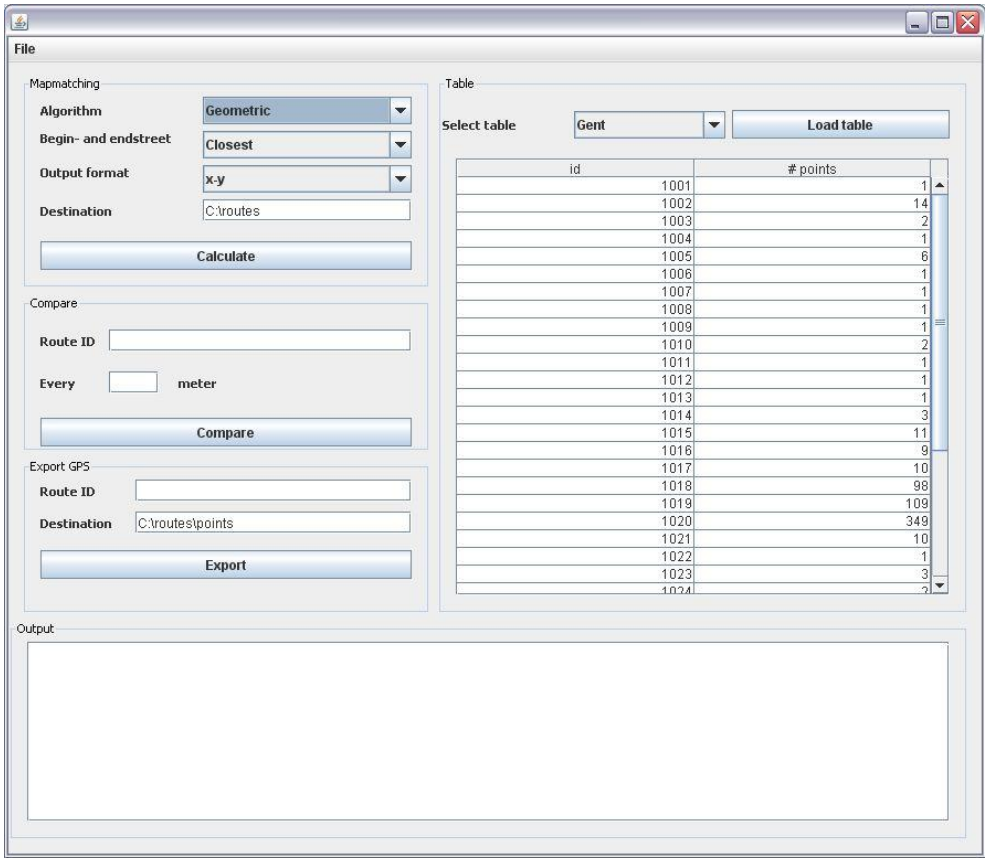
### 6.4.4 Het Export points panel

In figuur 6.6 wordt het *Export points* panel weergegeven. Dit kan gebruikt worden voor het exporteren van de GPS punten van een traject. Ook hier dient de gebruiker twee velden in te vullen. Het eerste veld *Route ID* dient de identifier van het traject te bevatten waarvan de gebruiker de GPS punten wenst te exporteren. In het tweede veld, *Destinaton*, dient de gebruiker de locatie te specifiëren waar hij wil dat de *shapefile* geplaatst wordt.


Figuur 6.6: Export points panel

### 6.4.5 De volledige GUI

Al deze panels vormen samen met het *Output* panel de GUI zoals weergegeven in figuur 6.7. Het *Output* panel wordt door de andere panels gebruikt om de gebruiker feedback te geven.




Figuur 6.7: De GUI

## 6.5 De database

### 6.5.1 De data

Bij het mappen van punten naar een stratennetwerk hebben we twee soorten gegevens nodig:

- Het stratennetwerk;
- De punten van het traject.

Dus als we nieuwe trajecten willen kunnen reconstrueren hebben we zeker het stratennetwerk nodig en de punten van de trajecten. Stratennetwerken worden gedigitaliseerd door bedrijven als Tele Atlas en worden bijvoorbeeld opgeslaan in shape-files. Deze moeten worden geïmporteerd in de database. Ook de trajecten waren in ons geval beschikbaar in shape-files die we geïmporteerd hebben in de database.

We maken gebruik van twee tabellen of views, één met de straten in en de andere met de punten in van de trajecten. De tabel of view met alle straten in moet volgende kolommen bevatten:

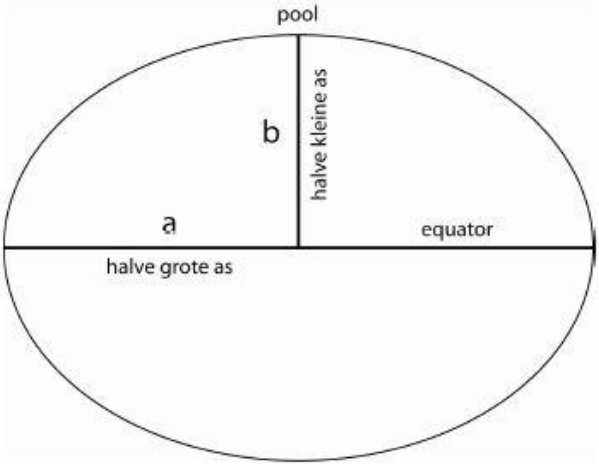
- *id*, die een unieke identifier bevat voor elke straat, type *int8;*
- *meters*, die de lengte van de straat bevat, type *float8;*
- *the_geom*, die de geometrie bevat van de straten, type *geometry.*

De tabel of view met de punten van de trajecten moet de volgende kolommen bevatten:

- *id*, die een unieke identifier bevat voor elk traject (dus meerdere punten hebben hetzelfde id als ze tot hetzelfde traject behoren), type *int8;*
- *tijdstip*, die het tijdstip bevat waarop het punt geregistreerd werd, type *timestamp/time;*
- *the_geom*, die de geometrie bevat van de punten, type *geometry.*

### 6.5.2 Transformatie van coördinaten

De aarde heeft een complexe vorm en er bestaan verschillende methodes om de aarde vereenvoudigd voor te stellen. Voor carthografie en positiebepaling wordt de aarde voorgesteld door een omwentelingsellipsoïde (zie figuur 6.8). Dit is een bol die lichtjes afgeplat is aan de polen. Er bestaan heel wat verschillende ellipsoïden die elk hun eigen schaal en afplatting hebben om telkens voor een deel van de aarde er zo perfect mogelijk bij aan te sluiten.




Figuur 6.8 Een ellipsoïde die de afgeplatte aarde voorstelt

In figuur 6.8 wordt een ellipsoïde voorgesteld. Een veelgebruikte ellipsoïde die beoogt om zo goed mogelijk bij het hele aardoppervlak aan te sluiten is het *World Geodetic System 1984* (WGS84) waarbij *a* gelijk is aan 6378137 meter en *b* gelijk aan 6356752 meter.

Er bestaan twee soorten coördinatensystemen voor het beschrijven van de aarde in de vorm van een omwentelingsellipsoïde:

- Geografische of geodetische coördinaten *(ϕ, λ, h);*
  - Breedte *ϕ*: de hoek tussen de ellipsoïdale normaal en het evenaarsvlak;
  - Lengte *λ*: de hoek met het vlak van de nulmeridiaan;
  - Hoogte *h*: de hoogte boven het oppervlak van de omwentelingsellipsoïde gemeten langs de normaal op dit oppervlak.
- Geocentrische coördinaten *(x, y, z).*

In figuur 6.9 worden deze twee coördinatensystemen grafisch voorgesteld om de lezer een beter inzicht te geven in de betekenis ervan. Geografische coördinaten kunnen naar geocentrische coördinaten getransformeerd worden door gebruik te maken van volgende analytische formules:

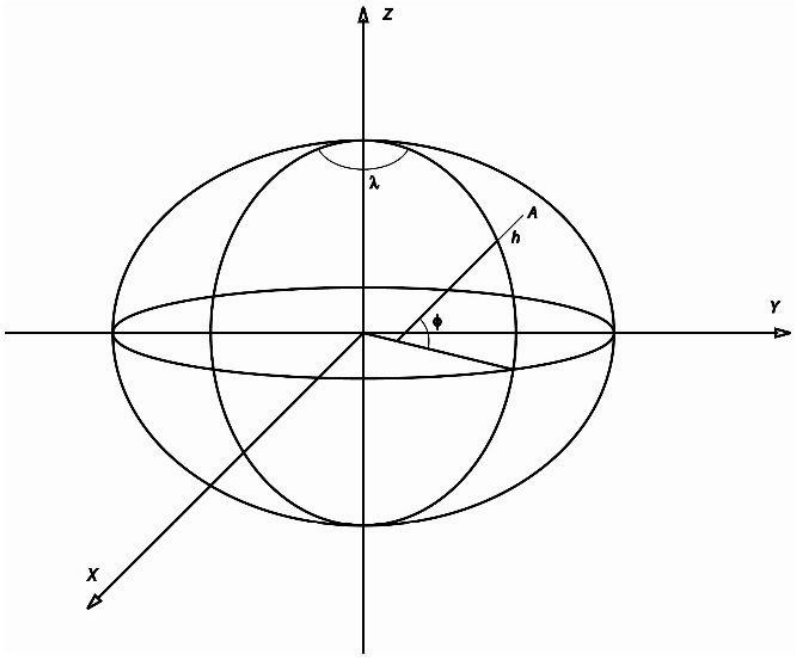
- $X = (N + h)\cos\phi\cos\lambda$
- $Y = (N + h)\cos\phi\sin\lambda$

- $Z = \left[N\left(1 - \frac{a^2-b^2}{a^2}\right) + h\right]\sin\phi$

Met $N = \frac{a}{\sqrt{1-\frac{a^2-b^2}{a^2}\sin^2\phi}}$

Omgekeerd kunnen we $\lambda$ rechtstreeks halen uit *x* en *y*, terwijl $\phi$ en *h* uit *x*, *y* en *z* berekend kunnen worden op iteratieve wijze of door gebruik te maken van benaderende formules.

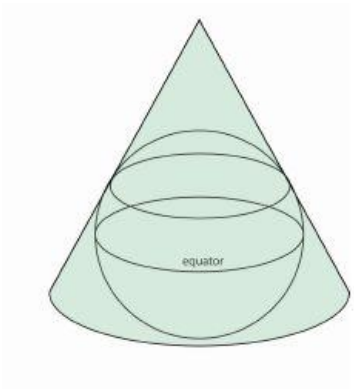
Geografische coördinaten zijn verbonden met een geodetische datum [41]. In deze geodetische datum zit de keuze van de omwentelingsellipsoïde vervat, alsook de vastlegging ervan in een fundamenteel punt en de astronomische oriëntatie. Het is zeer belangrijk dat we weten in welke geodetische datum de coördinaten gegeven zijn van de GPS punten en het stratennetwerk. Identieke waarden zijn verschillende punten in twee geodetische data. Ze kunnen zelfs op enkele honderden meters van elkaar verwijderd zijn. Daarom moeten we altijd de punten die we gebruiken naar hetzelfde systeem transformeren.




Figuur 6.9 De coördinatensystemen

Omdat de omwentelingsellipsoïden voor een bepaald deel van de aarde heel nauwkeurig moeten samenvallen, is het middelpunt van deze omwentelingsellipsoïden niet hetzelfde als het massacentrum van de aarde. Concreet, er bestaat een omwentelingsellipsoïde die beoogt om zo goed mogelijk bij België aan te sluiten. Dit wil zeggen dat op andere plaatsen op het aardoppervlak de omwentelingsellipsoïde niet samenvalt met de aarde. Het middelpunt van deze omwentelingsellipsoïde is dan ook niet gelijk aan het massacentrum van de aarde. Daarentegen in het geval van GPS hangen de satellieten in een baan rond de aarde die bepaald wordt door het massacentrum van de aarde (en de zwaartekracht die hier mee gepaard gaat). Deze baan vormt dus een omwentelingsellipsoïde met als middelpunt het massacentrum van de aarde. Het bijhorende referentiesysteem WGS84 is dan ook geocentrisch.

Om te werken met de GPS coördinaten in het vlak doet er zich nog een probleem voor. Het oppervlak van de omwentelingsellipsoïde herleiden naar het platte vlak kan niet zonder vervormingen gebeuren. Hiervoor worden projecties gebruikt. Voor België kan de Lambert projectie gebruikt worden die gebruik maakt van een kegelprojectie om de vervormingen te minimaliseren. Bij een kegelprojectie wordt er, zoals het woord zegt, geprojecteerd op een kegel zoals voorgesteld in figuur 6.10. De coördinaten die we na de Lambert projectie bekomen staan in het Lambert 72 formaat en de maximale correctie op de afstanden bedraagt 9 centimeter per kilometer, wat dus te verwaarlozen is.




Figuur 6.10 Kegelprojectie

We kunnen op twee manieren coördinaten transformeren van punten op een bol naar het vlak of omgekeerd:

- Analytische methode;
- Numerieke of directe methode.

In de analytische methode wordt de transformatie berekend door gebruik te maken van mathematische formules. Een voorwaarde is echter wel dat we de relatie tussen beide coördinatensystemen volledig moeten kennen. In de numerieke of directe methode berekenen we op basis van een verzameling overeenkomstige punten in beide systemen een benaderende transformatie. Dit wordt vooral gebruikt wanneer de relatie tussen de twee coördinatensystemen niet volledig gekend is. We transformeren coördinaten wanneer we moeten overschakelen van de ene kaartprojectie naar de andere of wanneer we moeten overschakelen van de ene geodetische datum naar de andere.

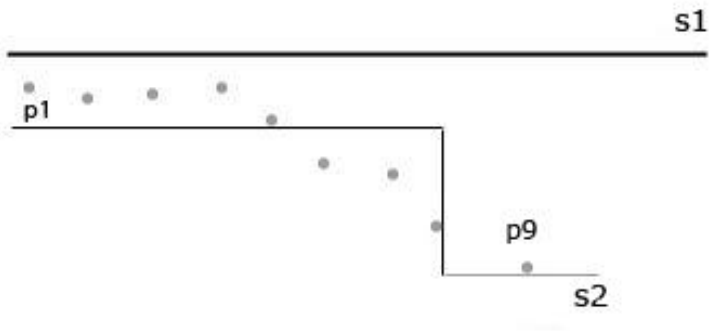
# Hoofdstuk 7: Experimenten

## 7.1 Map-matching GPS data

Om de trajecten te reconstrueren maken we gebruik van map-matching zoals besproken in hoofdstuk 3. Aangezien er een kleine afwijking van maximaal tien meter is op GPS-data, moeten we de punten mappen op een stratennetwerk. Het algoritme dat we hiervoor gebruikt hebben wordt hieronder weergegeven in enkele stappen:

1. Haal de GPS-punten uit de database
2. Bepaal de begin- en eindstraat
3. Bereken voor elke twee opeenvolgende punten hun bead (zoals gedefinieerd in hoofdstuk 3) met $V_{max}$ = 120 km/u en limiteer het stratennetwerk door alleen de straten te beschouwen die in deze beads liggen
4. Zoek de route van de beginstraat naar de eindstraat met de beste score in dit gelimiteerde stratennetwerk

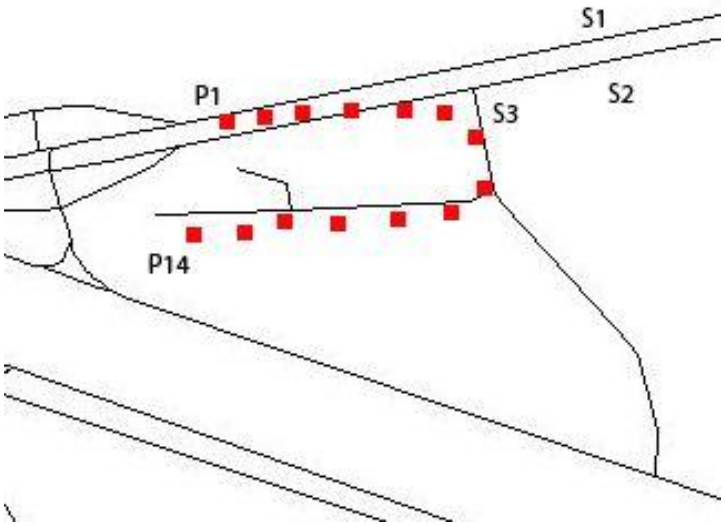
Voor het bepalen van de begin- en eindstraat gebruikte Gheys in [36] de straat die het dichtst bij het eerste punt ligt en de straat die het dichtst bij het laatste punt ligt. Hierdoor werd soms geen route gevonden tussen de begin- en eindstraat.




Figuur 7.1: Probleem met beginstraat

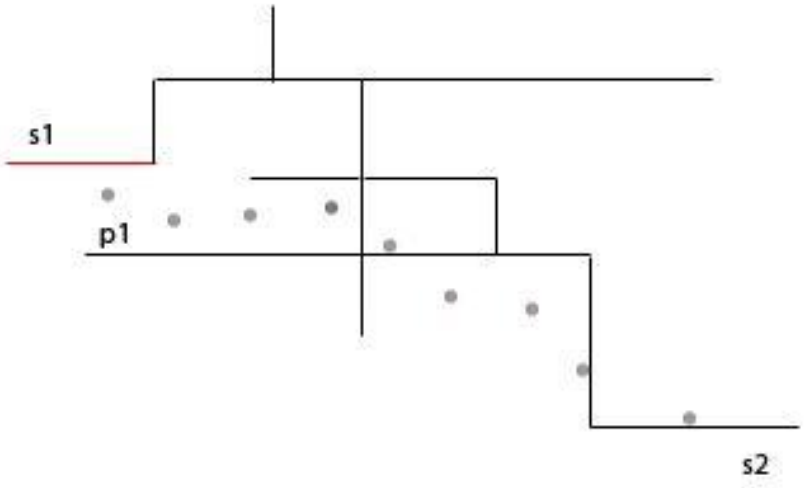
Beschouw figuur 7.1. De GPS punten zijn aangegeven in het lichtgrijs. p1 is het beginpunt van het traject, p9 het eindpunt. Wanneer we nu de dichtstbijzijnde straten selecteren van het begin- en eindpunt, krijgen we als beginstraat s1 en als eindstraat s2. Er is echter geen enkele straat die we kunnen nemen zodat we van straat s1 naar straat s2 kunnen gaan. Dit probleem kan zich voordoen in de buurt van een autostrade of expresweg.

In figuur 7.2 zien we de GPS punten van een traject. We zien dat P1 dichter bij S1 ligt dan bij S2. Er is geen route van P1 naar P14 die vertrekt vanuit S1 omdat S1 niet snijdt met S3. Wanneer we dus de dichtstbijzijnde straat bij P1 zouden nemen als beginstraat, en de dichtstbijzijnde straat bij P14 als eindstraat, zullen we geen route vinden.




Figuur 7.2: Probleem met beginstraat

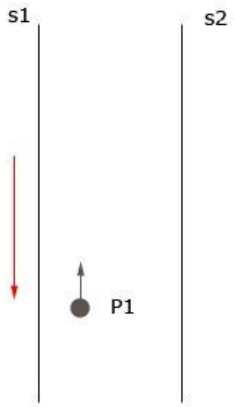
Een gelijkaardig probleem doet zich voor wanneer we kijken naar figuur 7.3. De dichtstbijzijnde straat bij het eerste GPS punt is s1, maar als dit een éénrichtingsstraat is (naar links op de figuur), dan bestaat er geen mogelijke route van s1 naar s2.




Figuur 7.3: Probleem met éénrichtingsstraat

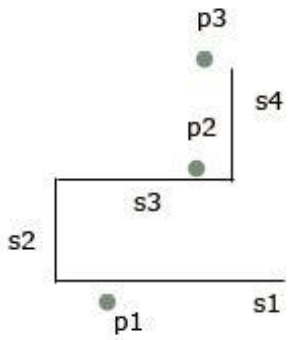
In [31] en [32] wordt dit probleem door Ochieng et al. opgelost door gebruik te maken van de heading informatie. De data die zij gebruiken zijn niet alleen coördinaten, maar hun voertuigen zijn ook uitgerust met een gyroscoop. Deze houdt bij in welke richting het voortuig aan het rijden is. Bij elk punt wordt dus ook de heading informatie opgeslaan.

Beschouw figuur 7.4 waarbij een voertuig in een noordwaartse richting aan het rijden is. Wanneer een éénrichtingsstraat s1 dichtbij dit voertuig (P1) van noord naar zuid loopt, kan men concluderen dat het voertuig zich niet op deze straat bevindt maar op straat s2.




Figuur 7.4: Headingsinformatie

Aangezien onze data echter geen informatie bevat over de heading van de voertuigen, moeten we dit probleem anders aanpakken. We kunnen proberen uit twee opeenvolgende punten de heading informatie te halen. Bijvoorbeeld als punt p1 ten zuiden ligt van p2, dan kunnen we afleiden dat het voertuig naar het noorden aan het rijden is. Beschouw echter figuur 7.5. Het voertuig draait rechtsaf op s1 naar s2. Opnieuw draait het rechtsaf naar s3 en ten slotte draait het linksaf naar s4. Hier ligt punt p2 ten noorden en ten oosten van punt p1, maar toch is de rijrichting van het voertuig westwaarts. Dit is dus geen goede oplossing voor het probleem.




Figuur 7.5: Heading bepalen

Om toch ervoor te zorgen dat er een route gevonden wordt, hebben we twee andere oplossingen bedacht voor dit probleem. Meer concreet hebben we de volgende oplossingen bedacht:

- Dynamisch bepalen van een begin- en eindstraat;
- Start- en eindcirkel die alle mogelijke begin- en eindstraten bevat.

### 7.1.1 Dynamisch bepalen van een begin- en eindstraat

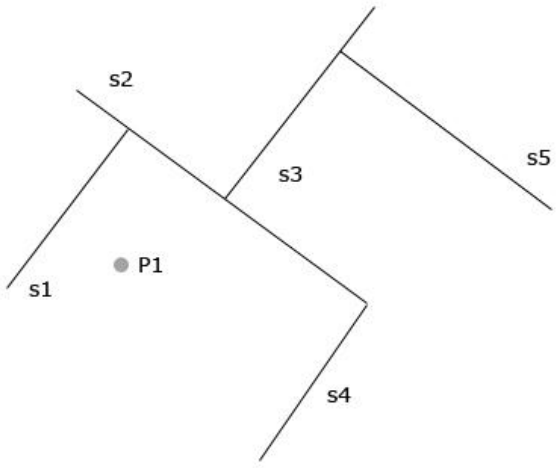
De eerste oplossing die we voorstellen is het dynamisch bepalen van de begin- en eindstraat. We beginnen dus met de straat die het dichtst bij het beginpunt ligt als beginstraat en de straat die het dichtst bij het eindpunt ligt als eindstraat. Zolang we geen route vinden tussen deze twee straten in het wegennetwerk dat omvat wordt door de opeenvolgende beads, gaan we een andere begin- en/of eindstraat kiezen.

In pseudo code geeft dit het volgende:

```
long[] beginstraten, eindstraten;
beginstraten = zoekNDichtsteStraten(beginpunt, aantal);
eindstraten = zoekNDichtsteStraten(eindpunt, aantal);
for(int k = 0; k < (N-1)*2; k++){
    for( int i = 0; i < beginstraten.length; i++){
        for( int j = 0; j < eindstraten.length; j++){
            if(i + j  <= k)
                route = zoekroute(beginstraten[i], eindstraten[j]);
            if(route != null)
                return route;
        }
    }
}
```

Hierbij zijn *beginstraten* en *eindstraten* zo gesorteerd dat de eerste straat de dichtstbijzijnde is en de laatste straat de straat is die het verst verwijderd is van het GPS-punt.

Beschouw figuur 7.6. De dichtstbijzijnde straat bij P1 is s1. Wanneer we geen route vinden beginnend met s1, selecteren we de volgende straat die het dichts bij P1 ligt op s1 na. Dit is s2. Zo gaan we verder tot we een route gevonden hebben of tot we een ingestelde limiet (bijvoorbeeld maximum 5) bereikt hebben.



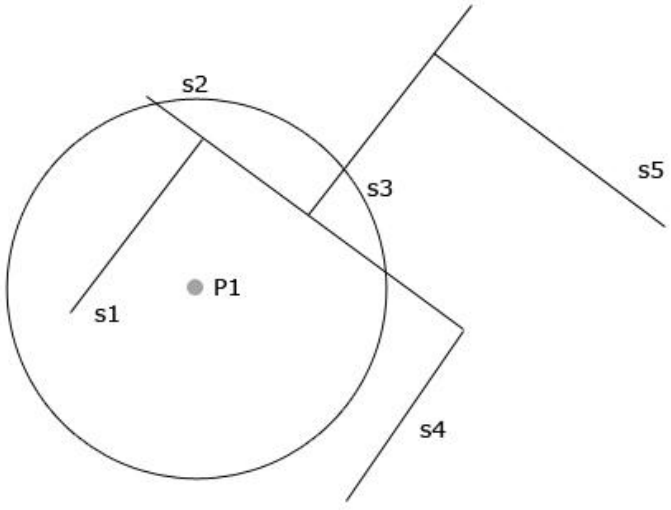

Figuur 7.6: Dynamisch bepalen van begin- en eindstraat

### 7.1.2 Bepalen begin- en eindstraat aan de hand van een cirkel

Een tweede oplossing die we voorstellen maakt gebruik van een start- en eindcirkel. We weten dat een GPS-signaal een maximale afwijking heeft van tien meter. We construeren dus rond het beginpunt en rond het eindpunt een cirkel met straat 10 meter. Alle straten die zich binnen deze cirkel bevinden worden dan beschouwd als start- of eindstraten. Nu zoeken we tussen deze straten een route, waarbij we de route kiezen met de beste score. Een beginstraat die dichter bij het beginpunt ligt krijgt hierbij een hogere score dan een straat die er verder vanaf ligt.

Hieronder geven we het algoritme in pseudo code:

```
long[] beginstratenmetscore, eindstratenmetscore;
beginstraten = zoekStratenInCirkel(beginpunt);
eindstraten = zoekStratenInCirkel(eindpunt);
for( int i = 0; i < beginstraten.length; i++){
      for( int j = 0; j < eindstraten.length; j++){
            route = zoekroute(beginstraten[i], eindstraten[j]);
             score = getScore(route);
            if(route != null)
                  if(score > besteScore){
                        besteScore = score;
                        besteRoute = route;
                  }
       }
}
return besteRoute;
```




Figuur 7.7: Oplossing m.b.v. een cirkel

Zoals we in figuur 7.7 kunnen zien, liggen de straten s1, s2 en s3 binnen de cirkel of intersecteren ze ermee. Er wordt nu een route gezocht vanuit deze straten naar de eindstraten, waarbij een route die gevonden wordt vanuit straat s1 een betere score krijgt dan een route gevonden vanuit straat s2 of s3.

### 7.1.3 Vergelijking van beide algoritmes

Wanneer we beide methodes vergelijken, merken we op dat de dynamische methode gemiddeld sneller is dan de methode met behulp van een cirkel. De dynamische methode stopt immers met zoeken van zodra er een route gevonden is, terwijl de methode met behulp van een cirkel alle mogelijke routes zoekt en die route kiest die de beste score krijgt. Hiermee is ook duidelijk dat

gemiddeld genomen de methode met behulp van een cirkel een beter resultaat oplevert dan de dynamische methode. De dynamische methode stopt immers wanneer een route gevonden is die niet noodzakelijk de beste route is. De methode die gebruik maakt van een cirkel gaat daarentegen wel op zoek naar de beste route. In de experimenten maken we gebruik van het dynamisch selecteren van een beginstraat en eindstraat omdat deze methode sneller is.

### 7.1.4 Map-matching van GPS-data: experimenten

Ik heb geholpen met het realiseren van een nieuwe paper [33] in verband met map-matching (zie Appendix A). Voor deze paper hebben we experimenten uitgevoerd die gebruik maken van twee verschillende datasets:

- Gent-dataset;
- Milaan-dataset.

De data in de Gent-dataset is data opgenomen door politievoertuigen. Telkens wanneer een agent een oproep ontving, zette deze het apparaat aan en begon het registreren van het traject. Gedurende deze registratie werd om de tien meter een locatie opgeslaan in de vorm van een *(x,y,t)* coördinaat, waarbij *(x,y)* de positie aanduidt en *t* het tijdstip. Dit registreren werd stopgezet wanneer de agent op de bestemming aangekomen was. Aangezien het echter steeds om oproepen ging met een lage prioriteit, hielden de agenten zich steeds aan de geldende maximumsnelheden.

De Milaan-dataset bevat in tegenstelling tot de Gent-dataset trajecten die bestaan uit minder punten. Door toepassing van clustering en visual analytics in het GEOPKDD project [42] werden er in deze dataset bevat drie soorten van trajecten ontdekt:

- Trajecten die het centrum van Milaan niet binnengaan;
- Trajecten van mensen die van buiten Milaan komen en tot in het centrum gaan;
- Trajecten alleen in het centrum van Milaan.

De trajecten die het centrum van Milaan niet binnengaan zijn bijvoorbeeld gemaakt door mensen die naar Milaan kwamen met de auto. Maar omdat ze niet in het centrum wonen (het centrum van Milaan is autovrij en alleen toegankelijk voor bewoners, parkeerden ze hun wagen net voor het centrum en gingen ze te voet of met het openbaar vervoer verder. Deze trajecten zijn ook waarschijnlijk gemaakt door mensen die naar hun werk gaan. Bedrijven liggen immers in de rand van de stad eerder dan in het centrum.

Om de kwaliteit van map-matching algoritmes te bespreken wordt er ook in andere publicaties typisch niet vergeleken met andere algoritmes. Dit is een gevolg van het feit dat de code van andere algoritmes niet beschikbaar is, en de data waarop deze uitgevoerd werden al helemaal niet. In het algemeen is er een afwezigheid van benchmarks voor map-matching algoritmes. Zo bekomen Ochieng et al. in [31] en [32] een correctheid van 100%. Zij vergelijken de resultaten van hun algoritme met de route zoals deze afgelegd werd door de testpersoon. Ook Gheys et al. vergelijken de resultaten van hun algoritme niet met andere algoritmes in [36]. Zij geven geen percentage van hoe goed hun algoritme werkt, maar bespreken hun algoritme in functie van de tijd die nodig is om de route te reconstrueren. In [43] melden White et al. ook dat er geen standaard data sets beschikbaar zijn om map-matching algoritmes te evalueren. Daarom maken zij gebruik van vier vooraf vastgelegde routes waarop zij hun algoritmes testen. Zij bekomen dan ook resultaten van

66% tot 86%, afhankelijk van de route waarop het algoritme uitgevoerd werd. Ze bekomen dit percentage door per GPS punt te kijken of het na de map-matching op de afgelegde route ligt.

De data van deze datasets was allemaal beschikbaar in shape-files. Deze werden ingeladen in een database, namelijk postgreSQL DBMS. Voor het werken met GIS data moest de PostGIS extensie geïnstalleerd worden. De testen zijn tenslotte uitgevoerd op een laptop, een Compaq 8510p, Core(TM)2 Duo CPU T8100 Intel processor van 2,1 GHz, 4 GB DDR RAM en een Hard disk van 160 Gb @ 7200 RPM.

Voor de uitgevoerde experimenten op de Gent-dataset zijn 33 trajecten gebruikt die varieerden van 32 tot 433 GPS punten. Deze trajecten samen bevatten 6027 GPS punten. Voor de experimenten op de Milaan-dataset zijn 30 trajecten beschouwd die varieerden van 17 tot 62 GPS punten. Tesamen goed voor 919 GPS punten. De tijd weergegeven in de tabellen beslaat het berekenen van de route zonder het inladen van de data uit de database.

De experimenten werden uitgevoerd met twee doelen:

- Bestuderen van de gevoeligheid van het bead algoritme aan de maximale snelheid waarmee de bead geconstrueerd wordt en hoe dit de uitvoeringstijd en nauwkeurigheid beïnvloedt;
- Het vergelijken van het bead algoritme met een eenvoudig geometrisch algoritme zoals in sectie 3.5.1 besproken wordt.

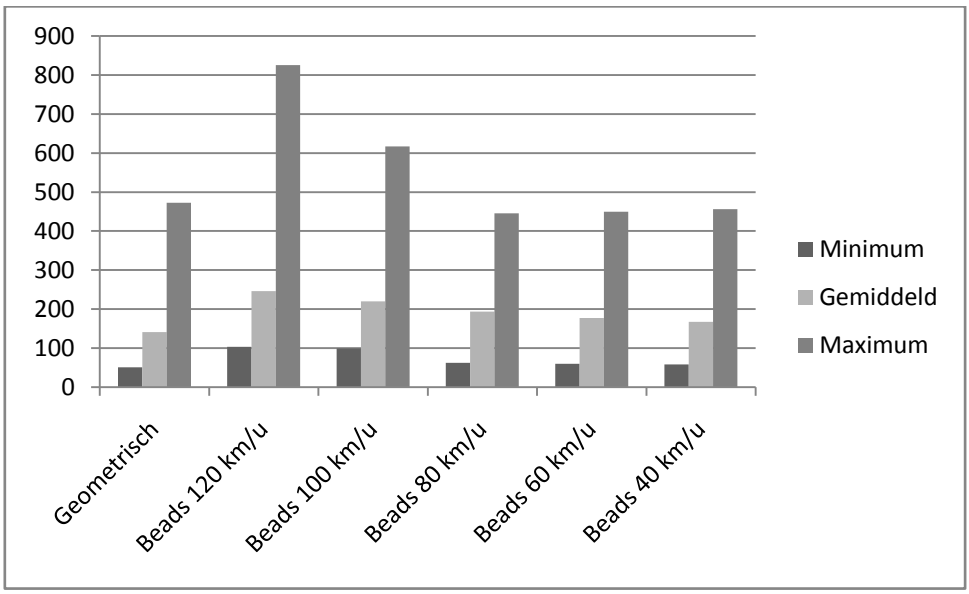
De experimenten werden zowel op de Gent-dataset als op de Milaan-dataset uitgevoerd. De maximale snelheid gebruikt bij het construeren van de beads varieert van 120 km/u tot 40 km/u. De minimale, gemiddelde en maximale waardes worden weergeven in de tabel. Deze waardes geven aan hoe lang het algoritme gelopen heeft (in milliseconden).

Wanneer we tabel 7.8 bekijken, zien we bijna geen verschil tussen de bead methode met maximum snelheid 120 km/u en de bead methode met maximum snelheid 80 km/u. Maar we merken een groot verschil wanneer de maximum snelheid 60 km/u is. We weten dat de data afkomstig is van politievoertuigen wanneer ze een oproep beantwoordden die een lage prioriteit had. Dit komt overeen met de experimenten want we weten dat in een stad er snelheidslimieten van 50 km/u en 70 km/u gelden. We kunnen dus concluderen dat de politievoertuigen zich aan de snelheidslimieten gehouden hebben.

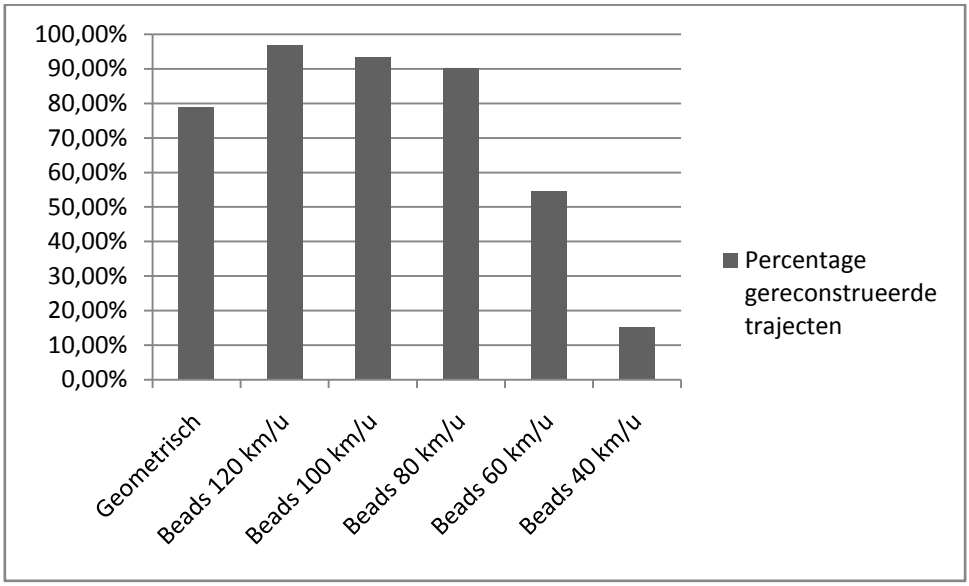
| Algoritme | Minimum | Gemiddeld | Maximum | Gereconstrueerde trajecten |
|---|---|---|---|---|
| Geometrisch | 50,56 | 141,32 | 472,66 | 78,8% |
| Beads: 120 km/u | 103,49 | 246,23 | 825,45 | 96,7% |
| Beads: 100 km/u | 99,42 | 219,93 | 616,80 | 93,4% |
| Beads: 80 km/u | 62,06 | 193,47 | 445,88 | 90,1% |
| Beads: 60 km/u | 60,24 | 177,09 | 449,21 | 54,5% |
| Beads: 40 km/u | 58,38 | 167,23 | 455,82 | 15,1% |

Tabel 7.8: Resulaten van de Gent-dataset

In grafiek 7.9 en grafiek 7.10 worden de resultaten van de experimenten nog eens grafisch voorgesteld. Op grafiek 7.10 merken we duidelijk de kloof tussen de beads met maximale snelheid groter of gelijk aan 80 km/u en beads met maximale snelheid kleiner dan of gelijk aan 60 km/u.




Grafiek 7.9: Running time op trajecten uit de Gent-dataset



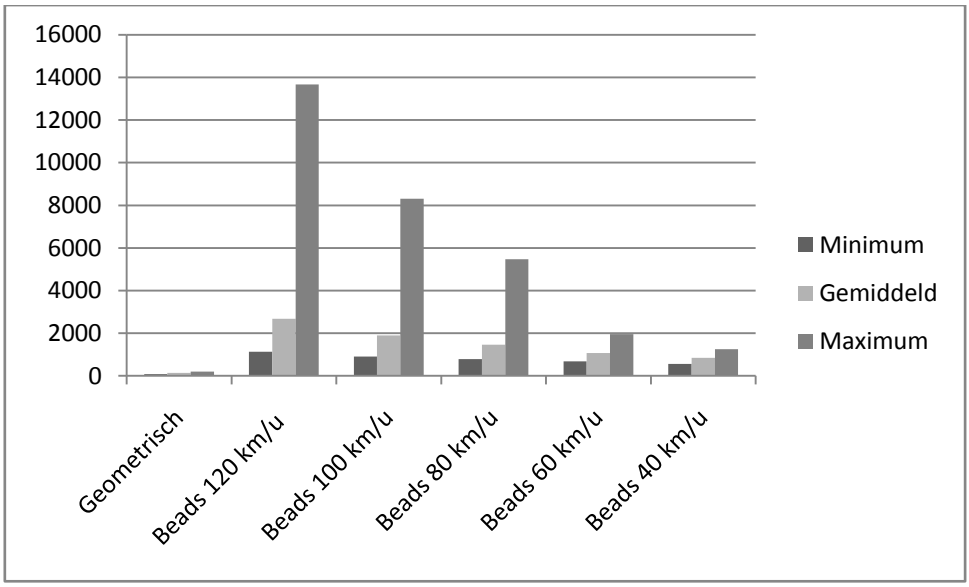

Grafiek 7.10: Percentage gereconstrueerde trajecten uit de Gent-dataset

Wanneer we nu kijken naar tabel 7.11, merken we meteen op dat de waardes veel hoger liggen dan de waardes in tabel 7.8. Dit valt te wijten aan het feit dat de GPS punten van de Milaan-dataset niet zo dicht op elkaar volgen als in de Gent-dataset. Bijgevolg zijn de beads tussen twee opeenvolgende punten veel groter en bevatten dus ook veel meer straten. Het stratennetwerk waar in gezocht wordt naar de beste route is dus veel groter en dus moet er veel meer berekend worden. We merken ook op dat hoe lager de maximum snelheid, hoe minder trajecten gereconstrueerd worden. Dit ligt aan het feit dat twee opeenvolgende punten veel verder van elkaar liggen dan in trajecten van de Gent-dataset.

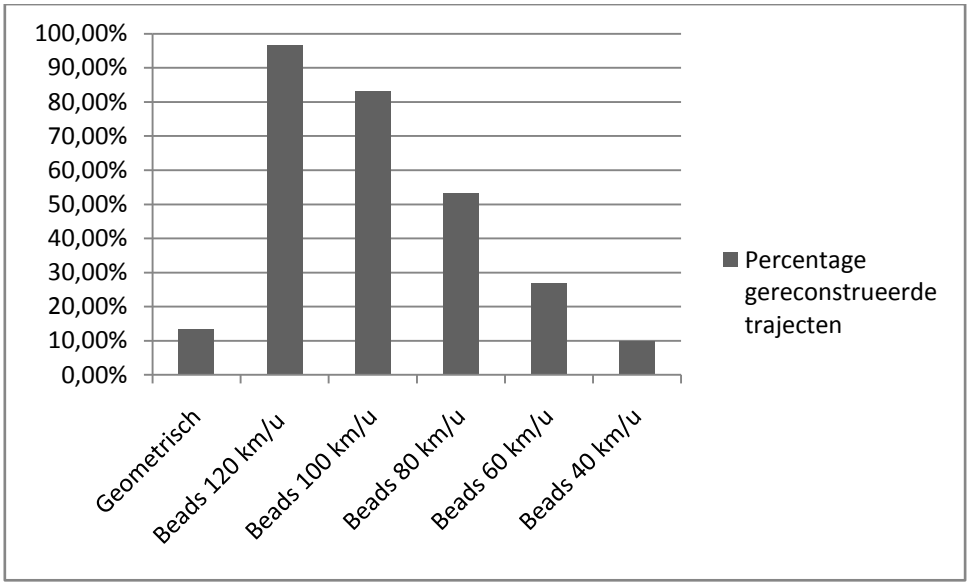
| Algoritme | Minimum | Gemiddeld | Maximum | Gereconstrueerde trajecten |
|---|---|---|---|---|
| Geometrisch | 79,69 | 134,99 | 193,98 | 13,3% |
| Beads: 120 km/u | 1132,82 | 2680,88 | 13665,48 | 96,7% |
| Beads: 100 km/u | 902,73 | 1889,39 | 8303,49 | 83,3% |
| Beads: 80 km/u | 778,41 | 1452,79 | 5475,36 | 53,3% |
| Beads: 60 km/u | 674,68 | 1066,19 | 1957,36 | 26,73% |
| Beads: 40 km/u | 561,39 | 845,75 | 1244,86 | 10,0% |

Tabel 7.11: Resulaten van de Milaan-dataset

Grafieken 7.12 en 7.13 geven een grafische weergave van de resultaten van de experimenten op trajecten uit de Milaan-dataset. In grafiek 7.12 valt vooral het verschil op tussen de geometrische methode en de rest. Dit valt te wijten aan het feit dat de geometrische methode op trajecten met punten die veel verder uit elkaar liggen zeer weinig trajecten kan reconstrueren. Dit kunnen wordt bevestigd door grafiek 7.13.




Grafiek 7.12: Running time op trajecten uit de Milaan-dataset




Grafiek 7.13: Percentage gereconstrueerde trajecten uit de Milaan-dataset

## 7.2 Map-matching van GSM data

In tegenstelling tot GPS toestellen sturen GSM's wel een signaal door. Voor het verkrijgen van GPS data moet er aan de gebruikers gevraagd worden om deze data op te slaan en door te sturen. Dit gebeurt niet automatisch. GPS toestellen sturen geen data door, maar halen alleen data van de verschillende satellieten om de positie te kunnen bepalen. GSM toestellen sturen echter wel data door naar de providers zodat GSM providers veel GSM data (zie hoofdstuk 2) kunnen opslaan. De huidige providers slaan echter niet veel gegevens op omdat deze toch niet gebruikt worden. Er wordt opgslaan wanneer iemand een oproep pleegt of ontvangt, vanuit welke cel, en wanneer hij van cel verandert (hand-over data). Wanneer deze data, die zeer groot is en representatief is, geprojecteerd kan worden op een wegennetwerk, kan deze gebruikt worden om allerlei patronen in te zoeken door er verschillende data mining technieken op toe te passen zoals beschreven in hoofdstuk 1. Maar het verkrijgen van GSM data was moeilijker dan verwacht. Vanaf we het privacy aspect vermeldden, kregen we geen antwoord meer. Dit is niet verwonderlijk aangezien het GEOPKDD project [42], wat toch een groot project is, ook te kampen heeft met dit probleem. Daarom proberen we GPS data zo aan te passen dat deze een indicatie kan geven of map-matching van GSM data mogelijk is.

### 7.2.1 Van GPS naar GSM

We willen vaststellen of het mogelijk is om louter en alleen door gebruik te maken van GSM data trajecten te reconstrueren. Aangezien we geen GSM data voor handen hebben, proberen we GPS data aan te passen om een indicatie te krijgen. We weten dat in een stad zoals Milaan de GSM cellen ongeveer de grootte hebben van een cirkel met straal 300 meter. Een realistische aanname zou dus zijn dat we ongeveer om de 300 meter een GPS punt zouden nemen van een traject. We proberen dan door gebruik te maken van enkel deze punten het traject te reconstrueren. Dit traject gaan we vergelijken met het gereconstrueerde traject dat gebruik maakt van alle punten om zo een beeld te krijgen van deze mapping.

Om een degelijke vergelijking te maken, maken we gebruik van twee technieken. In een eerste techniek vergelijken we het aantal straten van het oorspronkelijke traject dat overeenkomt met het aantal straten van het traject gevonden door gebruik te maken van punten die verder uit elkaar liggen. In de tweede techniek berekenen we de gemiddelde afstand van de GPS punten tot het traject. We zoeken het punt op het traject waar de reiziger was op tijdstip *t*. We berekenen dan de afstand van het GPS punt vastgelegd op tijdstip *t* tot dit geïnterpoleerde punt. Dit doen we voor alle GPS punten. Tot slot nemen we hier het gemiddelde van. Dit vergelijken we dan met de afstand tot het oorspronkelijke traject.

### 7.2.2 Experimenten

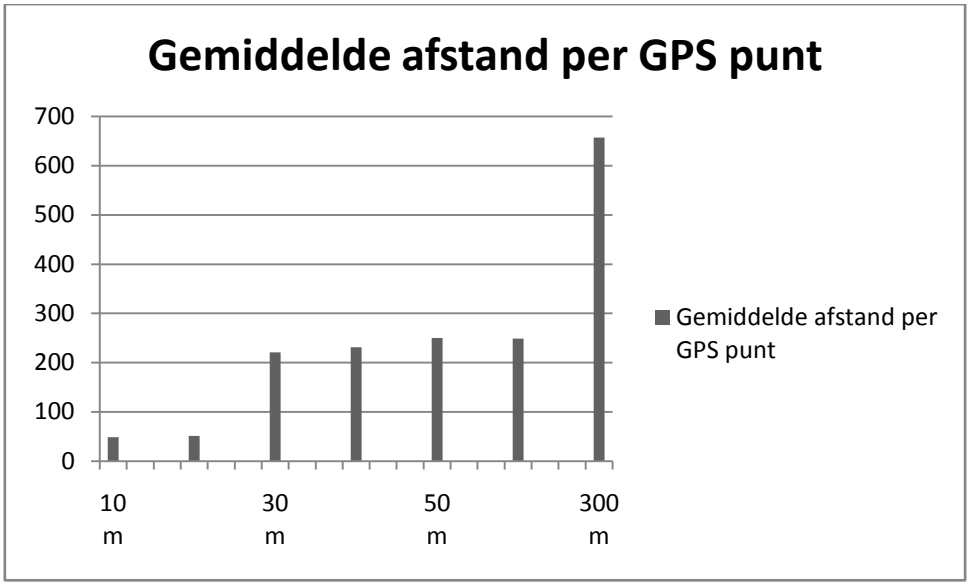
We vertrekken van een traject met punten om de 10 meter en we laten stap voor stap meer punten weg. In een tweede fase nemen we om de 20 meter een punt, dan om de 30 meter, 40 meter, 50 meter. Dan maken we een sprong naar 100 meter en tot slot punten om de 300 meter. Deze testen zijn uitgevoerd op een 15 tal trajecten bestaande uit in totaal 3912 punten. De resultaten worden weergegeven in tabel 7.14.

| Afstand | Gemiddelde afstand per punt (m) | Overeenkomstige straten (%) |
|---|---|---|
| 10 meter | 48,5 | 87,5 |
| 20 meter | 51,2 | 76,9 |
| 30 meter | 221,3 | 71,4 |
| 40 meter | 231,2 | 65,7 |
| 50 meter | 250,3 | 63,6 |
| 100 meter | 249,1 | 54,8 |
| 300 meter | 657,1 | 26,9 |

Tabel 7.14 Vergelijking voor gespreide GPS data

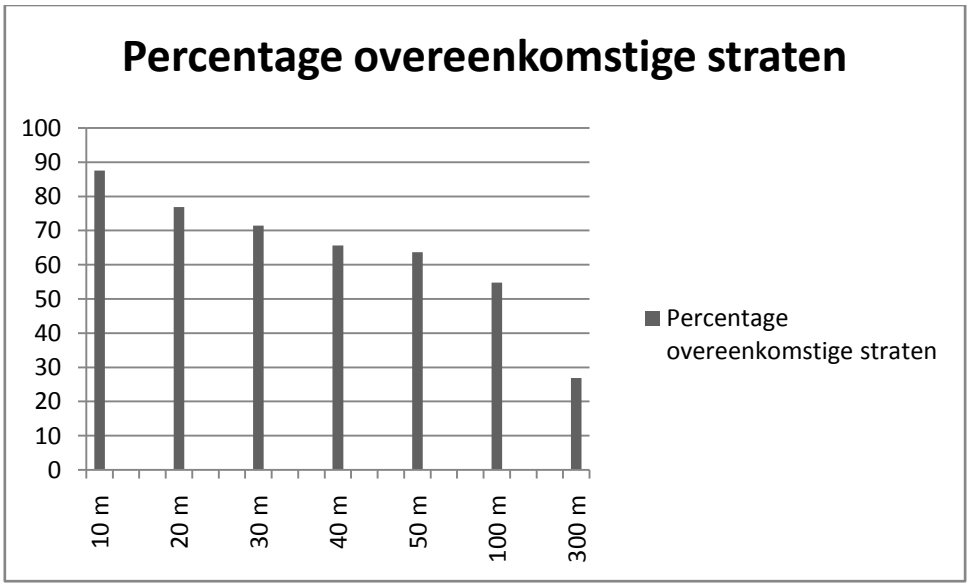
Wat onmiddellijk opvalt in tabel 7.14 is dat de gemiddelde afstand per punt stijgt wanneer de punten verder uit elkaar liggen. Op 300 meter is de gemiddelde afstand zelfs meer als 650 meter. Het percentage overeenkomstige straten daarentegen daalt wanneer de punten verder uit elkaar liggen. Voor punten die op 300 meter van elkaar liggen bedraagt dit percentage slechts ongeveer 27%. Wanneer we de resultaten van dichter bij bekeken, merkten we op dat dit percentage straten dat overeenkwam zich vooral bevond in de nabijheid van de punten.

In grafiek 7.15 en 7.16 worden deze resultaten grafisch weergeven.




Grafiek 7.15 Gemiddelde afstand per GPS punt

In grafiek 7.15 valt de stijgende lijn goed op. Hoe dichter de punten op elkaar volgen, hoe kleiner de gemiddelde afstand per GPS punt en dus hoe verder de punten uiteen liggen, hoe hoger de gemiddelde afstand per GPS punt. Voor GPS punten die op 300 meter van elkaar liggen, is de gemiddelde afstand gelijk aan ongeveer 650 meter. Dit wil zeggen dat het traject gemiddeld een 650 meter van de GPS punten verwijderd ligt, wat toch een relatief grote afstand is.




Grafiek 7.16 Percentage overeenkomstige straten

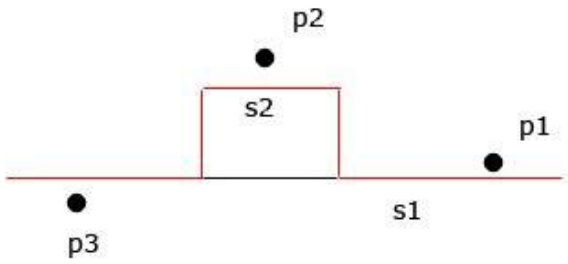
In grafiek 7.16 zien we ook duidelijk een dalende lijn wanneer de punten verder uit elkaar liggen. Voor punten die op 10 meter van elkaar liggen hebben we nog een percentage van 87,5%, terwijl voor punten die op 300 meter van elkaar liggen we nog maar een schamele 27% halen. Deze resultaten bevestigen dus de resultaten uit grafiek 7.15.

Uit deze resultaten kunnen we dus afleiden dat het *niet* mogelijk is om in een stad trajecten te reconstrueren gebruik makend van GSM data. Echter wanneer er bijkomende gegevens beschikbaar

zijn, zoals naburige draadloze netwerken, bluetooth-apparaten, signaalsterkte, ... krijgen we meer punten dan alleen het wisselen van cel en het maken van een oproep. Zo kan GSM data de nauwkeurigheid van GPS data benaderen en kunnen er wel trajecten uit gereconstrueerd worden. Dit wordt onder andere gedaan door een experiment van Nathan Eagle et al. in [34] aan het MIT. Wanneer we weten dat een GSM signaal afkomstig is van iemand die zich op een snelweg bevindt, is er wel een mogelijkheid om deze data te matchen naar een stratennetwerk. *We zouden kunnen concluderen dat GPS data in een stad overeenkomt met GSM data op autosnelwegen.*

Zowel in grafiek 7.15 als in grafiek 7.16 zien we dat het verschil tussen punten om de 30 meter en punten om de 50 meter relatief klein is. Dit viel al op tijdens de experimenten. In sommige experimenten was de reconstructie van trajecten gebruik makend van punten om de 30 meter beduidend beter dan de trajecten gereconstrueerd met punten om de 50 meter. Echter waren er ook een aantal trajecten waarbij trajecten gereconstrueerd uit punten om de 50 meter bijna even goed of soms zelfs iets beter waren. Wanneer we deze trajecten bekeken, zagen we onmiddellijk wat de oorzaak was.

Bekijk figuur 7.17. We zien duidelijk aan de punten dat het rode traject gevolgd werd. Echter wanneer we p2 weglaten zal het gereconstrueerde traject gewoon over s1 lopen. Nu hangt het er dus vanaf welk punt we weglaten. Als we bij punten om de 50 meter p2 niet weglaten, en bij punten om de 30 meter wel, dan zal de reconstructie gebruik makend van punten om de 50 meter een beter resultaat to gevolg hebben.



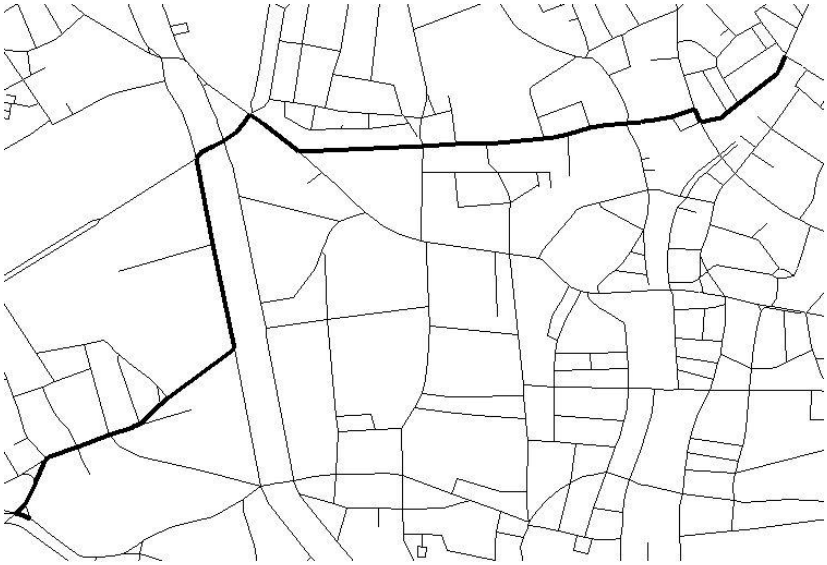

Figuur 7.17 Probleem bij weglaten van punten

Echter wanneer we punten nemen die verder van elkaar liggen, zijn de resultaten veel minder beïnvloedbaar door deze gevallen. De resultaten voor gereconstrueerde trajecten uit punten om de 300 meter waren dan ook altijd beduidend slechter dan experimenten over meer punten.

In de volgende figuren wordt een traject weergegeven dat gereconstrueerd is. In figuur 7.18 werd gebruik gemaakt van alle GPS punten om het traject te reconstrueren. Wanneer we nu de helft van de punten weglaten, en dus een punt nemen om de 20 meter, zien we dat het gereconstrueerde traject in figuur 7.19 al wat verschilt van het oorspronkelijke traject. Wanneer we nu punten nemen om de 50 meter, zien we in figuur 7.20 dat het gevonden traject al grotendeels verschilt van het oorspronkelijke traject. De gevonden route komt hier overeen met de snelste route van het beginpunt naar het eindpunt gebruik makend van wegen van het stratennetwerk.

Figuur 7.18 Traject gereconstrueerd met behulp van alle GPS-punten

Figuur 7.19 Traject gereconstrueerd met GPS-punten om de 20 meter



Figuur 7.20 Traject gereconstrueerd met GPS-punten om de 50 meter

# Bronnen


[1] Jans, W., *Spatial en Spatio-Temporal Data en Data Mining met toepassingen in de verkeerskunde,* Master thesis, UHasselt, 2005.

[2] Giannotti, F. & Pedreschi D., *Mobility, Data Mining and Privacy: Geographic Knowledge Discovery*, 2008.

[3] Mobility, Data Mining & Privacy: Preserving anonymity in geographocally referenced data

[4] Kuijpers B. & Othman, W., *Trajectory Databases: data models, uncertainty and complete query languages*, Lecture Notes in Computer Science, 2006 – Springer.

[5] Kuijpers B. & Othman, W.*, Modelling uncertainty of moving objects on road networks via space-time prisms*, International Journal of Geographical Information Science, 2008.

[6] Othman, W. , Kuijpers, B. & Grimson, R., *A case study of the difficulty of quantifier elimination in constraint databases: the alibi query in moving object databases,* Arxiv preprint arXiv:0712.1996, 2007.

[7] Ghys, K. , *Map matching tracking data,* Master thesis, UHasselt, 2007.

[8] Abul, O., Bonchi, F. & Nanni, M., *Never Walk Alone: Uncertainty for Anonymity in Moving Objects Databases,* IEEE 24th International Conference on Data Engineering, 2008.

[9] Bogorny, V., Kuijpers, B. & Alvares, L. O., *ST-DMQL: a Semantic Trajectory Data Mining Query Language,* International Journal of Geographical Information Science, 2009.

[10] http://ccnga.uwaterloo.ca/~jscouria/GSM/gsmreport.html (laatst bezocht op 11/06/2009).

[11] http://www.gsmfordummies.com/ (laatst bezocht op 11/06/2009).

[12] Varshavsky, A., Chen, MY., de Lara, E., Froehlich, J., *Are GSM phones the solution for localization?*, Proceedings of the Seventh IEEE Workshop on Mobile Computing, 2006.

[13] http://www.gsmworld.com/ (laatst bezocht op 11/06/2009).

[14] Wang, S., Min, J., Yi, BK, *Location Based Services for Mobiles: Technologies and Standards*, IEEE ICC Beijing, 2008.

[15] http://infolab.uvt.nl/~remijn/telematica/scripties98/groep17/script1.html (laatst bezocht op 07/06/2008).

[16] http://www.frequentieland.nl/gsm/index.htm (laatst bezocht op 07/06/2008).

[17] http://www.dmoz.org/Science/Earth_Sciences/Geomatics/Global_Positioning_System/ (laatst bezocht op 11/06/2009).

[18] Bowditch, N., *The American Practical Navigator, an epitome of navigation*, National Imagery and Mapping Agency, p. 163-172, 2002.

[19] http://www.galileoju.com/page.cfm?voce=m&idvoce=301&plugIn=1 (laatst bezocht op 11/06/2009).

[20] Wageningen experiment.

[21] http://www.lamosca.be/thetarget.htm (laatst bezocht op 07/06/2008).

[22] Mouza, C. Du & Rigaux, P., *Mobility Patterns,* GeoInformatica, 2005.

[23] Alvares, L. O., Bogorny, V., Kuijpers, B., Macedo, J. A. F. de, Moelans, B. & Vaisman, A., *A Model for Enriching Trajectories with Semantic Geographical Information,* symposium on Advances in geographic information systems, 2007.

[24] http://www.techpluto.com/what-is-3g/ (laatst bezocht op 11/06/2009).

[25] Halkidi, M. *Quality assessment and Uncertainty Handling in Data Mining Process,* 2003.

[26] Verykios, V., Bertino, E., Fovino, I.N., Provenza, L.P., *State-of-the-art in Privacy Preserving Data Mining,* ACM Sigmod Record, 2004.

[27] http://www.microsoft.com/netherlands/ondernemers/techniek_trends/umts.aspx (laatst bezocht op 11/07/2008).

[28] http://www.cdt.org/privacy/eudirective/EU_Directive.html (laatst bezocht op 13/07/2008).

[29] http://www.dataprotection.ie/viewdoc.asp?m=&fn=/documents/legal/6aiii.htm (laatst bezocht op 13/07/2008).

[30] Giannotti, F., Nanni, M., Pedreschi, D., Pinelli, F., *Trajectory Pattern Mining*, conference on Knowledge discovery and data mining, 2007.

[31] Ochieng, W.Y., Quddus M., Noland R.B., *Map-matching in complex urban road networks*, Brazilian Journal of Cartography (Revista Brasileira de Cartografia) Nº 55/02, 2003.

[32] Ochieng W.Y., Quddus M., Zhao L., Noland R.B., *A general map matching algorithm for transport telematics applications*, GPS Solutions, Vol. 7(3):157–167, 2003.

[33] Ghys K., Kuijpers B., Moelans B., Othman W., Vaisman A., Vangoidsenhoven D., *Map Matching and Uncertainty: an Algorithm and Real-World* Experiments, Technical report, 2009.

[34] MIT Media Lab: Reality Mining, http://reality.media.mit.edu/ (laatst bezocht op 04/06/2009).

[35] PostgreSQL, http://www.postgresql.org/ (laatst bezocht op 04/06/2009).

[36] PostGIS, http://postgis.refractions.net/ (laatst bezocht op 04/06/2009).

[37] Minq Software: DBVisualizer, http://www.minq.se/products/dbvis/ (laatst bezocht op 04/06/2009).

[38] Eclipse, http://www.eclipse.org/ (laatst bezocht op 04/06/2009).

[39] Netbeans, http://www.netbeans.org/ (laatst bezocht op 04/06/2009).

[40] TatukGIS, http://www.tatukgis.com (laatst bezocht op 04/06/2009).

[41] Beeckman, J-P., *Wat men moet weten om zonder zorgen te navigeren met GPS*, Nationaal Geografisch Instituut – www.ngi.be (laatst bezocht op 03/06/2009).

[42] Geographic Privacy-aware Knowledge Discovery and Delivery (GeoPKDD), project gesponserd door de Europese Commissie onder het FP6-IST-FET programma December 2005 - Maart 2009, http://www.geopkdd.eu/ (laatst bezocht op 01/06/2009).

[43] White C.E., Bernstein D., Kornhauser A.L., *Some map matching algorithms for personal navigation assistants*, Transportation Research Part C, 2000, Elsevier.

[44] Othman, W., *Uncertainty management in trajectory databases*, PhD. Proefwerk, 2009.

# Appendix A

# Paper

# Map Matching and Uncertainty: an Algorithm and Real-World Experiments

Kristof Ghys, Bart Kuijpers, Bart Moelans, Walied Othman, Alejandro Vaisman, and Dries van Goidsenhoven

Hasselt University & Transnational University of Limburg, Belgium
{kristof.ghys,bart.kuijpers,bart.moelans, walied.othman, alejandro.vaisman,dries.vangoidsenhoven}@uhasselt.be

**Abstract.** A common problem in moving object databases (MOD) is the reconstruction of a trajectory from a trajectory sample (i.e., a finite sequence of time-space points). A typical solution to this problem, is linear interpolation, which assumes that objects move at constant minimal speed. A more realistic model is based on the notion of *uncertainty*. This model uses *beads* (also called time-space prisms) between two consecutive time-space points $a$ and $b$ in a trajectory sample, to estimate the positions where the object could have been, when it moved between $a$ and $b$. Nowadays, GPS-based navigation systems are becoming increasingly popular. More than often, object positions obtained using these location-aware devices fall outside a road or street network. Thus, matching the users position to a location on the digital map is required. This problem is denoted *map matching*. Many algorithms have already been proposed to solve the map matching problem, although none of them considers the uncertainty issue. In this paper we study the relation between map matching and uncertainty, and propose an algorithm that combines weighted $k$ shortest paths with space-time prisms. We apply this algorithm to two real-world case studies, consisting in trajectory samples of very different kinds: in the first case, observations were taken at small regular intervals; in the second case, they were taken at larger and irregular intervals. In addition, we compare these results against a classic and simpler geometric algorithm, run over the same two datasets. We show that accounting for uncertainty leads to obtaining more positive matchings, that largely compensate the longer running times which, however, remain within reasonable limits.

## 1 Introduction

One of the most popular location-aware devices nowadays are GPSs. Even though most people use a GPS as a navigational tool, it can also be used for storing the position of a moving object (e.g., car, pedestrians) for data analysis. We can, for instance, analyze the routes taken by a person and then study why she chose one road instead of another. The main disadvantage of storing the GPS coordinates is that they are not exact, i.e., they do not always match the road. Therefore, every GPS device accounts for these errors by mapping the exact street driven

on instead of just modeling the GPS coordinates received from a satellite. Many algorithms were devised for this task, as we discuss in Section 2. The problem of matching GPS positions to a road network is called *map matching.*

Research on spatial databases, which started in the 1980s from work in GIS (Geographic Information Systems), was extended in the second half of the 1990s to deal with spatio-temporal data. One particular line of research in this field, started by Wolfson, focused on the so-called *Moving Object Databases* (MODs) [4,17], a field in which several data models and query languages have been proposed to deal with moving objects whose position is recorded at, not always regular, moments in time. Some of these models are geared towards handling uncertainty, that may come from various sources (errors in measurements, or interpolation, for instance), and also ad-hoc query formalisms have been proposed [13]. For a detailed discussion of models and techniques for MODs, we refer the reader to [4].

One particular model for the management of the uncertainty of the moving object's position in between sample points is provided by the *bead* model. In this model, it is assumed that besides the time-stamped locations of the object also some background knowledge, in particular a (e.g., physically or law imposed) speed limitation $\mathsf{v}_i$ at location $(\mathsf{x}_i, \mathsf{y}_i)$ is known. The bead[1] between two consecutive sample points is defined as the collection of space-time points where the moving objects may have passed, given the speed limitation. The chain of space-time prisms connecting consecutive trajectory sample points is called a *lifeline necklace* [2](see Figure 1 for an illustration). Whereas space-time prisms were already conceptually known in the time geography of Hägerstrand in the 1970s [5], they were introduced in the area of GIS by Pfoser [11] and later studied by Egenhofer and Hornsby [2,7], and Miller [9].

In this paper we present an algorithm that uses a combination of weighted $k$ shortest path algorithms and space-time prisms, to solve the map matching problem, and apply this algorithm to two different real-world case studies, corresponding to the movement of cars in two European cities.

### 1.1 Problem statement and case study

We now present the first real-world case study, which we use throughout the paper. An emergency service in an European city[2] found out that new or transferred employees did not know the city very well, and it took them longer than expected to arrive at the place of intervention. Even though the company could solve this problem purchasing a standard route planner, the shortest/fastest route computed by these commercial route planners would not be the best solution, because of several factors, like: (a) They do not take into account the time of the observations, e.g. at five o'clock there is always a traffic jam at the city station, so cars must avoid this area around that time, if possible; (b) they do not take into account certain locations, like schools; (c) They do not take into

[1] In this paper we use the more intuitive term *space-time prism* to refer to beads
[2] Due to privacy reasons we cannot disclose further information about the data

account extra information (like school routes, or tram lines). Thus, they decided to design a tool to solve the problem described above. As a first step of this work, there was the need to perform data analysis over a set of routes followed by cars during their interventions. The officers where asked to start recording with a GPS device (measures were recorded every ten meters) from the moment they got a call from the headquarters to the moment when they arrived at the intervention site. In addition, they were requested to fill out a survey, with questions, for example, about the reason for taking a particular route.

In the scenario above, a typical problem that arises is that about ninety-five percent of the points fall outside the road actually taken. Thus, there is a need to map points to the road network. This problem is denoted *map matching.* Formally, the problem of map matching is defined as follows.

**Definition 1 (cf. [16]).** An object is moving along a finite system (or set) of streets, $\overline{N}$. A location-aware device such as GPS provides an estimate for the vehicles location at a finite number of points in time, denoted by $\{0, 1, \ldots, t\}$. The vehicles actual location at time $t$ is denoted by $\overline{P}^t$ and the estimate is denoted $P^t$. *Map matching* is the process of determining the street in $\overline{N}$ that contains $P^t$. That is, to determine the street that the vehicle is on at time $t$. □

There are many algorithms to map match trajectories to road networks. According to their usage, they can be classified in (a) Online map matching algorithms (the general case), and (b) Off-line map matching algorithms. The former are aimed at responding quickly such that user can be provided with the position information. The latter are mainly used to analyze visited routes. In these algorithms, the main concern is not the response time, but accuracy. This is where our proposal positions.

### 1.2 Contribution and paper organization

The main contribution of this paper is a novel map matching algorithm that uses a combination of techniques for handling uncertainty in trajectory databases (more precisely, *space-time prisms*), and weighted $k$-shortest paths algorithms. We first introduce many usual problems present in GPS data; then, we discuss how our proposal addresses these problems. We also report experimental results over two real-world cases: first, over the running example described in Section 1.1, consisting in very precise information, given the small interval between measurements. Then, we apply the same methodology to a second real-world case study, corresponding to trajectories of cars in the city of Milan, recorded during a week. Here, GPS coordinates are recorded at irregular and less frequent intervals, i.e., data are more imprecise. Also, in the first case, moving objects are all of similar characteristics, while in the second case, we have cars, trucks, buses, among the kinds of vehicles recorded. Not only we compare the results over these two cases, but we also compared our algorithm against other simpler one, of a geometric kind (see Section 2). We show that while the geometric algorithm performs rather well in the case of precise data, when data become

imprecise, like in the case of the Milan example, most of the trajectories cannot be reconstructed, while our algorithm achieves a rate of above 90% of success in trajectory reconstruction.

In Section 2 we review related work on map matching. Section 3 presents the theoretical basis of our work, mainly studying how uncertainty in road networks is addressed using the notion of space-time prisms. In Section 4 we discuss different techniques and technical problems of map matching, and introduce our algorithm. In Section 5 we report experimental results, concluding in Section 6.

## 2 Related work

In this paper we work with *trajectory samples*, which are well-known in MODs, namely finite sequences of time-space points. A trajectory sample database contains a finite number of labeled trajectory samples. There are various ways to reconstruct trajectories from trajectory samples, of which linear interpolation is the most popular in the literature [4]. However, linear interpolation relies on the (rather unrealistic) assumption that between sample points, a moving object moves at constant minimal speed. It is more realistic to assume that moving objects have some physically determined speed bounds. Given such upper bounds, *an uncertainty model* has been proposed which constructs *space-time prisms* between two consecutive time-space points in a trajectory sample(see Section 3 for a formal definition of space-time prism). Basic properties of this model were discussed a few years ago by Egenhofer et al. [2] and Pfoser et al. [11], but space-time prisms were already known in the time-geography of Hägerstrand in the 1970s [5]. In short, a *space-time prism* is the intersection of two cones in the time-space space, such that all possible trajectories of the moving object between the two consecutive space-time points, given the speed bound, are located within the space-time prism. A query of particular interest, studied mainly by Egenhofer and Hornsby [2,7], who give approximate solutions to the problem, is the boolean *alibi query*, which query asks whether two moving objects, given by samples of space-time points and speed limitations, can have physically met [8].

### 2.1 Map-matching algorithms

The technique of mapping GPS coordinates to the road network itself is called map matching. Here we describe relevant work in this field.

Map matching algorithms can be classified into three categories: (a) Geometric: algorithms in this class use geometric information of the original road network, and do not consider topological information; (b) Topological: algorithms that make use of the geometry of the network as well as the connectivity and contiguity of the links; (c) Probabilistic: these algorithms define an elliptical or rectangular confidence region. The error region is superimposed on the road network to identify a road segment. If the error region contains more than one street, probabilistic algorithms perform a weighted search on the candidate

streets. Some authors proposed fuzzy logic approaches as a fourth category, although they can fall into some of the above three classes.

Taylor [14] presents a geometric algorithm, called Road Reduction Filter (RRF), for real-time map-matching. The system tracks a vehicle on all possible roads within an error region and instantly computes probable invalid roads. Another example of a geometric algorithm can be found in [16].

The best known topological algorithm was introduced by Greenfeld [3]. In his work, the author reviews several approaches to map matching, and proposes a weighted topological algorithm, based on assessing the similarity between the characteristics of the street network and the positioning pattern of the user. A weighted score is computed and the match is determined selecting the highest score or the most likely candidate for a correct match. He claims that the procedure performed very well in his tests, although acknowledging that further research is needed to confirm this.

Quddus *et al.* [12] also performs a thorough study of existing techniques, and proposes an algorithm that addresses many issues ignored in other proposals, by taking into account the heading information of the moving object at $\overline{P}^t$. The authors use a similar weighting strategy than [3], and also consider the speed of the moving object. Basically, they use three kinds of weights: weighting for vehicle heading and bearing of the street, weighting for proximity of a point to a street, and weighting for the position of the point relative to the street. The authors also address the problem of outlier points.

In probabilistic algorithms, the *heading information* is the most important factor for deciding the shape of the error region (i.e., an ellipse or a rectangle). If no heading information exists, circles or squares are defined. It can be shown that a point received from a GPS device has a probability between 95% and 99%, of falling into the error region. The main idea is that if there is only one street in the error region, the point matches to that street. If there is more than one street, weighting scores are computed according to some criteria (like perpendicular distance, or connectivity). An example of a probabilistic algorithm is [10]. The algorithm we present in this paper can be considered a variant of a probabilistic algorithm, where error regions are replaced with the projection of space-time prisms over the plane. In addition, we use topological techniques.

Brakatsoulas *et al.* [1] discuss three algorithms for map-matching. Data consist of samples taken every 30 seconds which means that the car, moving at a maximum speed of 50km/h, could have traveled a distance of 417 meters before a next point is recorded (recall that our data are taken every 10 meters). They present an incremental map-matching algorithm which sequentially matches the next possible road segment and tries to obtain a global-optimal solution by using a look-ahead technique, and matching the trajectory with the Fréchet distance and the weak Fréchet distance.

It is worth noting that none of the works discussed above presents conclusive evidence of performance, not a comparison against other methods based on real-world experimentation. In the best case, particular cases are presented. On the

contrary, we do not only compare two datasets of different characteristics, but also two methods, a simple geometric one, and our proposal.

## 3 A model for moving object data with uncertainty

In this paper, we consider moving objects in the two-dimensional $(x, y)$-space $\mathbf{R}^2$ and describe their movement in the $(t, x, y)$-space $\mathbf{R} \times \mathbf{R}^2$, where $t$ is time (we denote the set of the real numbers by $\mathbf{R}$). In this section, we define trajectories, trajectory samples, space-time prisms and trajectory (sample) databases. Although it is more traditional to speak about moving object databases, we use the term trajectory databases to emphasize that we manage the trajectories produced by moving objects.

### 3.1 Trajectories and trajectory samples

Moving objects, which we assume to be points, produce a special kind of curves, which are parameterized by time and which we call *trajectories*.

**Definition 2.** A *trajectory* $T$ is the graph of a mapping $I \subseteq \mathbf{R} \to \mathbf{R}^2 : t \mapsto \alpha(t) = (\alpha_x(t), \alpha_y(t))$, i.e.,

$$T = \{(t, \alpha_x(t), \alpha_y(t)) \in \mathbf{R} \times \mathbf{R}^2 \mid t \in I\},$$

where $I$ is the *time domain* of $T$. □

In practice, trajectories are only known at discrete moments in time. This partial knowledge of trajectories is formalized in the following definition. To stress that some $t, x, y$-values (or other values) are constants, we will use sans serif characters.

**Definition 3.** A *trajectory sample* is a finite set of time-space points $\{(\mathsf{t}_0, \mathsf{x}_0, \mathsf{y}_0), (\mathsf{t}_1, \mathsf{x}_1, \mathsf{y}_1), ..., (\mathsf{t}_N, \mathsf{x}_N, \mathsf{y}_N)\}$, on which the order on time, $\mathsf{t}_0 < \mathsf{t}_1 < \cdots < \mathsf{t}_N$, induces a natural order. □

A trajectory $T$, which contains a trajectory sample $\{(\mathsf{t}_0, \mathsf{x}_0, \mathsf{y}_0), (\mathsf{t}_1, \mathsf{x}_1, \mathsf{y}_1), ..., (\mathsf{t}_N, \mathsf{x}_N, \mathsf{y}_N)\}$, i.e., $(\mathsf{t}_i, \alpha_x(\mathsf{t}_i), \alpha_y(\mathsf{t}_i)) = (\mathsf{t}_i, \mathsf{x}_i, \mathsf{y}_i)$ for $i = 0, ..., N$, is called a *geospatial lifeline* for this trajectory sample [2]. A common example of a lifeline is the reconstruction of a trajectory from a trajectory samples by linear interpolation [4], where the unique trajectory, that contains the sample and that is obtained by assuming that the trajectory is run through at constant lowest speed between any two consecutive sample points, is constructed.

**Definition 4.** Given a sample $S = \langle (t_0, x_0, y_0), (t_1, x_1, y_1), \ldots, (t_N, x_N, y_N) \rangle$, the trajectory $LIT(S) := \bigcup_{i=0}^{N-1} \{(t, \frac{(t_{i+1}-t)x_i + (t-t_i)x_{i+1}}{t_{i+1}-t_i}, \frac{(t_{i+1}-t)y_i + (t-t_i)y_{i+1}}{t_{i+1}-t_i}) \mid t_i \leq t \leq t_{i+1})\}$ is called the *linear-interpolation trajectory* of $S$. □

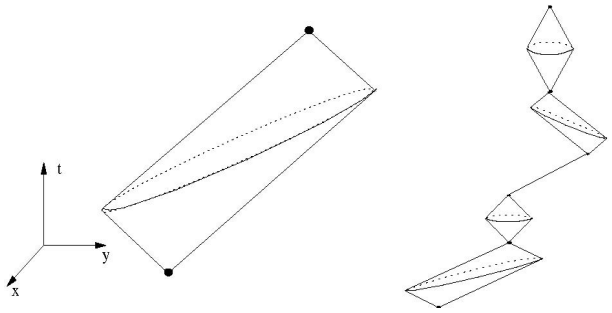



**Fig. 1.** A space-time prism and a lifeline necklace.

### 3.2 Modeling uncertainty with space-time prisms

Often, in practical applications, more is known about trajectories than merely some sample points $(\mathsf{t}_i, \mathsf{x}_i, \mathsf{y}_i)$. For instance, background knowledge like a physically or law imposed speed limitation $\mathsf{v}_i$ at location $(\mathsf{x}_i, \mathsf{y}_i)$ might be available. Such a speed limit might even depend on $\mathsf{t}_i$. The speed limits that hold between two consecutive sample points can be used to model the uncertainty of a moving object's location between sample points. For modeling uncertainty, Pfoser et al. [11], and later Egenhofer et al. [2,7], introduced the notion of *beads* (i.e., space-time prisms) in the moving object database literature. Before, Wolfson used *cylinders* to model uncertainty [4,17]. However, cylinders give less precision (by a factor of 3, compared to space-time prisms). Let $S$ be a sample $\langle(t_0, x_0, y_0), (t_1, x_1, y_1), ..., (t_N, x_N, y_N)\rangle$. Basically, the cylinder approach to managing uncertainty, depends on an uncertainty threshold value $\varepsilon > 0$ and gives a buffer of radius $\varepsilon$ around $LIT(S)$. In the space-time prism approach, for each pair $(t_i, x_i, y_i), (t_{i+1}, x_{i+1}, y_{i+1})$ in the sample $S$, their related space-time prism does not depend on a uncertainty threshold value $\varepsilon > 0$, but rather on a maximal velocity value $v_{\max}$ of the moving object.

We now formalize the concepts above. We know that at a time $t$, $\mathsf{t}_i \leq t \leq \mathsf{t}_{i+1}$, the object's distance to $(\mathsf{x}_i, \mathsf{y}_i)$ is at most $\mathsf{v}_i(t - \mathsf{t}_i)$ and its distance to $(\mathsf{x}_{i+1}, \mathsf{y}_{i+1})$ is at most $\mathsf{v}_i(\mathsf{t}_{i+1} - t)$. The spatial location of the object is therefore somewhere in the intersection of the disc with center $(\mathsf{x}_i, \mathsf{y}_i)$ and radius $\mathsf{v}_i(t - \mathsf{t}_i)$ and the disc with center $(\mathsf{x}_{i+1}, \mathsf{y}_{i+1})$ and radius $\mathsf{v}_i(\mathsf{t}_{i+1} - t)$. The geometric location of these points is referred to as a *space-time prism*, and defined as follows, for general points $p = (t_p, x_p, y_p)$ and $q = (t_q, x_q, y_q)$ and speed limit $v_{\max}$ .

**Definition 5.** The *space-time prism* with origin $p = (t_p, x_p, y_p)$, destination $q = (t_q, x_q, y_q)$, with $t_p \leq t_q$, and maximal speed $v_{\max} \geq 0$ is the set of all points $(t, x, y) \in \mathbf{R} \times \mathbf{R}^2$ that satisfy the following constraint formula.

$$\Psi_{\mathsf{B}}(t, x, y, t_p, x_p, y_p, t_q, x_q, y_q, v_{\max}) := (x - x_p)^2 + (y - y_p)^2 \leq (t - t_p)^2 v_{\max}^2 \\ \wedge\ (x - x_q)^2 + (y - y_q)^2 \leq (t_q - t)^2 v_{\max}^2\ \wedge\ t_p \leq t \leq t_q.$$

We denote this set $\mathsf{B}(p, q, v_{\max})$ or $\mathsf{B}(t_p, x_p, y_p, t_q, x_q, y_q, v_{\max})$. □

In the formula $\Psi_{\mathsf{B}}(t, x, y, t_p, x_p, y_p, t_q, x_q, y_q, v_{\max})$, we consider $t_p, x_p, y_p, t_q$, $x_q, y_q, v_{\max}$ to be parameters, whereas $t, x, y$ are considered variables defining the subset of $\mathbf{R} \times \mathbf{R}^2$.

Figure 1 illustrates the notion of a space-time prism in time-space. Whereas a continuous curve connecting the sample points of a trajectory sample was called a geospatial lifeline, *a chain of space-time prisms* connecting succeeding trajectory sample points is called a *lifeline necklace* [2]. More formally, for a sample $S = \langle (t_0, x_0, y_0), (t_1, x_1, y_1), ..., (t_N, x_N, y_N) \rangle$ the set $\bigcup_{i=0}^{N-1} \mathsf{B}(t_i, x_i, y_i, t_{i+1}, x_{i+1}, y_{i+1}, v_{\max})$ is called the *space-time prism chain* of $S$ [2].

## 4 Mapping trajectory data to a road network

In this section we describe a novel map-matching algorithm that uses space-time prisms, for handling problems of real-world data, like traffic jams, or gaps between measures, produced by some interference in the satellite signal, for instance, tunnels. We first discuss these problems in some detail.

### 4.1 Problems with real-world data

Consider Figure 2, where the GPS points start moving towards the road in the top of the figure (i.e., the incorrect one), but they appear to recover the right path afterwards. This is a typical situation in real data, due to uncertainty in the measurement of the object's position. A second problem that influences the precision of map matching algorithms, are large gaps in the GPS points. Gaps (see Figures 3 and 4) may appear due to several reasons: (a) A faulty GPS signal caused by the bad reception of the coordinates (this is quite unlikely to occur, however). (b) An interruption of the communication between the GPS satellite and the device, due to a densely forested area, a tunnel, or high buildings (if the GPS signals are blocked because of tall buildings, we speak of *urban canyons*). For example, Figure 3 shows a gap of 140 meters (indicated by an ellipse), while Figure 4 shows a gap due to the existence of a tunnel. Yet another problem is depicted in Figure 5, where two possible roads may be correct. This problem can easily be solved by examining the timestamp of each GPS point, and looking, for instance, at the direction of both road segments (i.e., if they are one-way or not).

### 4.2 Using space-time prisms for map matching

In Section 3 we discussed linear interpolation as a method for reconstructing trajectory samples. However, linear interpolation relies on the assumption that between two consecutive points, the object moves with a *constant* speed. A more realistic assumption would be that the object moves within a bounded speed limit, which leads to the *space-time prisms* uncertainty model (Section 3). For example, in our case study, given the time between two consecutive recorded points, and a maximal speed, a car could have been in many possible locations,

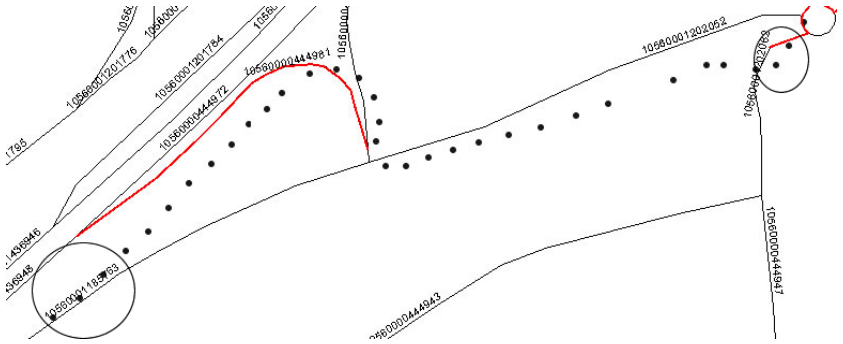


**Fig. 2.** Ambiguous GPS signals

given by the projection of the space-time prisms over the plane. Even though, in fact, this projection would be an ellipse, for simplicity we compute a bounding box given by two points $R(X_1, Y_1)$ and $U(X_2, Y_2)$ (see Figure 6). $R$ is computed as follows: $X_1$ is the farthest point on the X-axis that can be reached moving away from $b$ driving at maximum speed $v_{max}$. Analogously, $Y_1$ is the is the farthest point on the Y-axis that can be reached moving away from $b$ driving at maximum speed $v_{max}$. $U$ is computed as follows: $X_2$ is the farthest point on the X-axis that can be reached moving away from $a$ driving at maximum speed $v_{max}$. Analogously, $Y_2$ is the is the farthest point on the Y-axis that can be reached moving away from $a$ driving at maximum speed $v_{max}$.

Figure 7 shows the space-time prism projection computed for two points $A$ and $B$, that are 13m away from each other. The space-time prism is extremely large in this case, because the traveling distance from $A$ to $B$ is 35 seconds (possibly due to a traffic light stop). The space-time prism computes the region where the car *could have been when it traveled 35 seconds at a maximum speed of 120 km/h,* which results in a prism containing many streets. On the other hand, Figure 8 shows the space-time prism for two points $A$ and $B$, which are 11 meters from each other, with a travel time of one second. Here, the prism includes only two road segments.

### 4.3 A simple geometric algorithm

A simple approach to map matching consists in using a geometric algorithm that computes the road segments closest to the GPS coordinates (measuring the perpendicular distance from the point to the road), and matches a point to a road segment. In this way, a trajectory sample that lies within the road is obtained. In spite of its simplicity, which makes it very (computationally) efficient, this method has some drawbacks, given the characteristics of real-world data discussed above, which, as we see later in our experiments (Section 5), sometimes prevent obtaining a matched trajectory (for instance, if there are large gaps in the data). Thus, we need a more involved algorithm to overcome these problems.

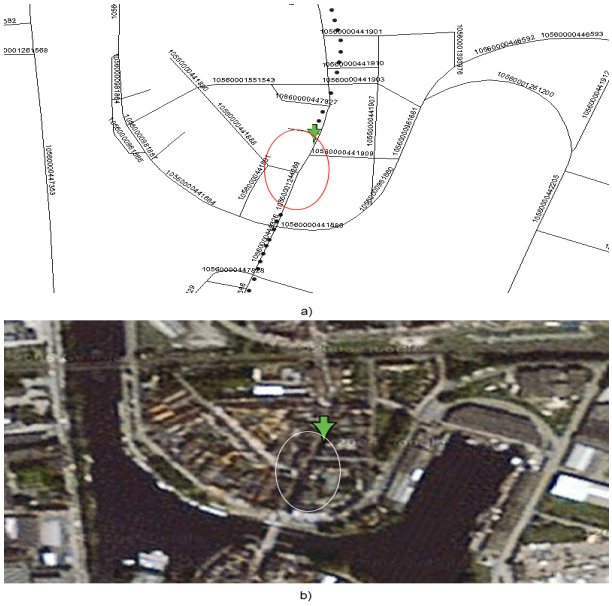



**Fig. 3.** a) Large gaps in trajectory samples. b) A map from the same area.

### 4.4 An algorithm based on weighted shortest path computation

The $k$-shortest path problem is a well known problem in networks. Note that, in a road network, we may have $k$ paths between two nodes, with $k > 1$. We adapt Yen's algorithm [18] to rank the shortest paths between a pair of nodes using a scoring algorithm discussed in Section 4.5 to give weight to the edges in a road network. This algorithm searches the shortest paths in a 'pseudo'-tree containing $k$ shortest loopless paths. The very shortest one is obtained in the first place, and the other shortest paths are always explored based on shorter ones (see Example 1 below). The algorithm described in [18] first computes the shortest path between two points using the A* algorithm [6]. Then, it takes every node in the shortest path, except the end node, and calculates a second shortest path (called a *spur* path) from each selected node to the end node. For each such node, the path from the start node to the current node is the root path. Two restrictions are placed on the spur path: (1) It must not pass through any node on the root path (the paths are loopless); and (2) It must not branch from the current node on any edge used by a previously found $k$-shortest path. If a new spur path is found it is appended to the root path for that node, to form a complete path from start to end. In our adaptation of the algorithm, a score is calculated for each shortest path (as described in Section 4.5); if this score is greater or equal to that of the shortest path calculated at the beginning, it is added to the result list. We return the path with the highest score. The complexity of the algorithm is $O(n^3)$. Calculating the spur paths from each

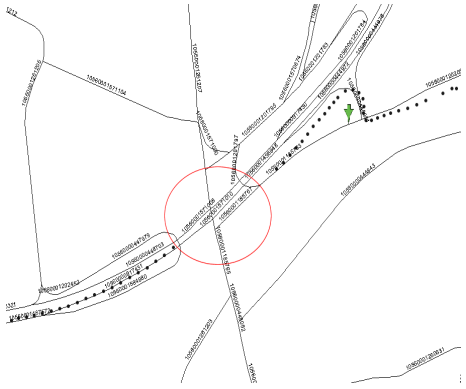


**Fig. 4.** A tunnel producing a gap: a map (a), and an ellipse indicating the gap (b)

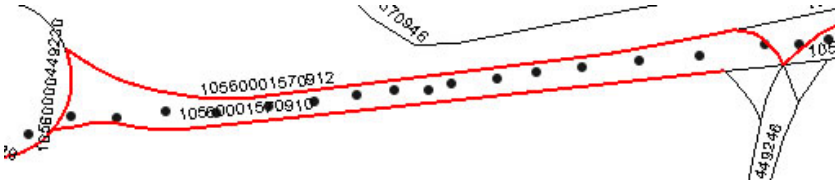



**Fig. 5.** More then one possible road segment

node is $O(n)$ and using a shortest-path algorithm (Dijkstra, [15]) $O(|V|^2 + |E|)$ with $|V|$ number of vertices (intersections) and $|E|$ the number of streets (edges).

*Example 1.* Consider the road network of Figure 9. Assume that all arrows are equally weighted. The problem consists in finding the shortest from $A$ to $D$. It is clear that the shortest path is $A, B, C, D$ so we include this path in the result path. Now we look for other paths starting from the shortest one. We start with root path $A$, and look for a path from $A$ to $D$ that is not already in the result list. The only possible path not including $B$ is $A, E, F, G, H, D$. We add this path to the result list. Now we start with the nodes $A, B$ (the new root path) and find $A, B, F, G, H, D$. These are all possible path, and the algorithm ends. □

There are several special cases which are difficult to handle just using a shortest path algorithm. For example, Figure 10 depicts a moving object that followed the road indicated in thin lines (three points are shown). Algorithms based in shortest paths would likely chose the thick line road. Suppose for instance that weights 6, 5 and 6 are given to streets 1, 2, and 3, respectively. Since Dijkstra's algorithm [15] works backwards, it decides between lines 2 and 3, and the latter has a weight larger than the former one.

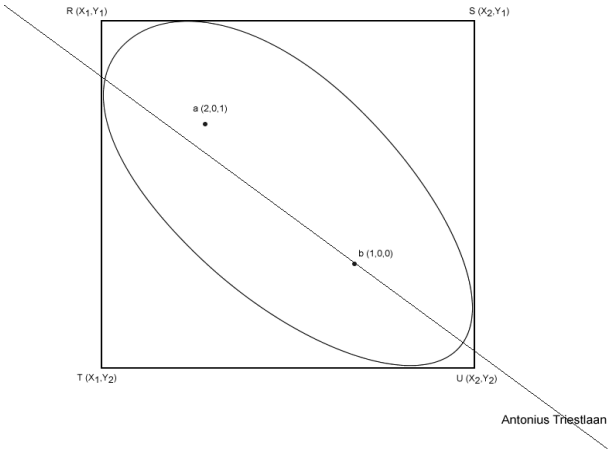



**Fig. 6.** Projection of a space-time prism over the plane, for two points $a$ and $b$.

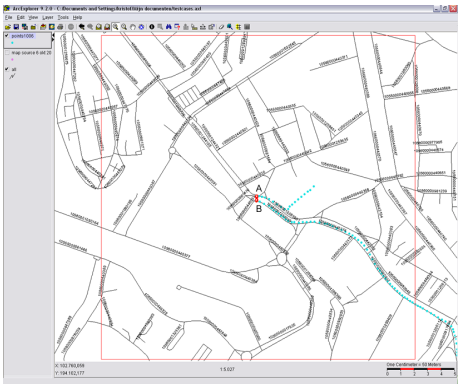



**Fig. 7.** Bounding box for two consecutive points with a time gap of 35 seconds

### 4.5 Map matching algorithms using space-time prisms

Now we show how we can improve the algorithm above, and avoid, for instance, the triangle problem, using the notion of *space-time prisms* introduced in Section 3. In addition, using space-time prisms allows to use a dataset containing outliers, since the weight they receive is meaningless, as we explain below. This algorithm computes the road segments closest to the GPS points, as follows. For each two points we compute a bounding box as explained in Section 4.2. Then, we give weights to the streets, in a way such that the street closest to a given point gets weight $n$, the second closest street weight $n-1$, continuing until the closest $n$ street, which receives weight 1. Notice then that *the road segments to be included are computing using space-time prisms,* avoiding manually checking all the roads (and including roads unlikely to be followed).

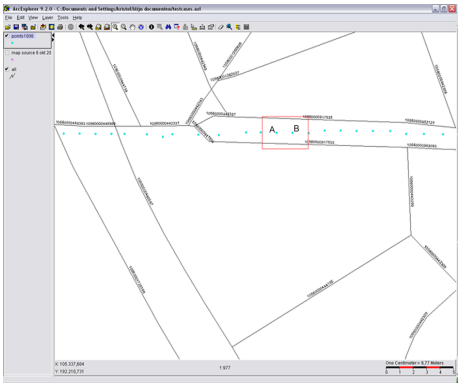



**Fig. 8.** Bounding box for two consecutive points with a time gap of 1 second

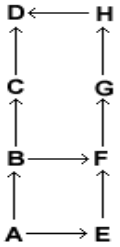



**Fig. 9.** Road network for Example 1.

*Example 2.* An example of the algorithm can be found in Table 1 and Figure 11. We now describe the scoring algorithm, assuming that the space-time prisms have been computed, i.e., that we know $n$ for each point (we do not give details of this computation in this example). Starting from point A, we assign a weight to each street according to the closeness to this point. Therefore, street with id=1 received a score of 3, and streets with id's 3 and 4 received 2 and 1, respectively. Thus, A will likely be matched to street with id=1. We continue in this way until all points have been analyzed and weights assigned. Table 1 shows the outcome of the algorithm. We can see that the final scores follow the actual route taken by the cars. □

In summary, the map matching algorithm proceeds as follows:

1. First, it bounds the network by calculating, for each pair of consecutive points, which roads they could have driven on (as described in Section 4.2).
2. Then, for each GPS point, it computes the closest road, as described in Section 4.5, and assigns scores to each road segment. A score for a segment $s$ is computed adding up the weighs of all the segments that match $s$.

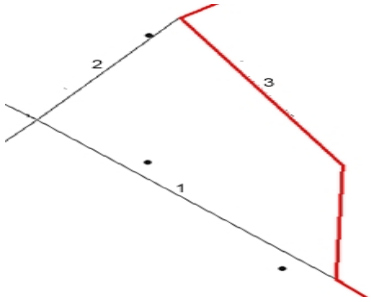



**Fig. 10.** The triangle problem.

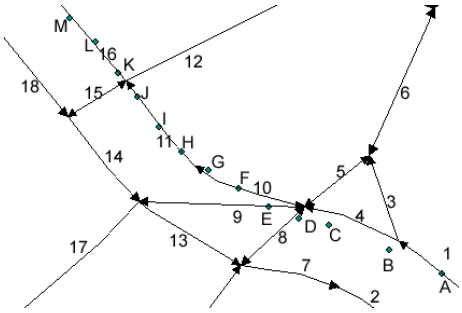



**Fig. 11.** A → M are GPS coordinates. 1 → 18 are streetIds

3. Finally, it computes, within this limited network, the $k$-shortest paths, taking the shortest path with the highest score computed in step 2.

Sometimes, by simply adopting as starting and ending points the street segments closest to the first and last GPS measurements, respectively, we may not find a correct match. This is illustrated in Figure 4.5. Here, the road segment closest to $p_1$ would be R1 (although the trajectory clearly followed R2). The ending point will be on R2, eventually preventing finding a route for this trajectory. To solve this problem, and taking into account that the maximum GPS error is about 10 meters, the algorithm looks at all the possible starting segments (instead of only one) in a circle with a radius of 5 meters with center on the first GPS point, and selects all road segments that intersect with this circle. The same procedure is followed for the end segment. Then the algorithm looks for possible routes between all the possible start segments and end segments within this boundaries.

## 5 Implementation and experimental evaluation

In this section we report the experimental results obtained running the algorithm over two datasets we describe below. The map matching algorithms were coded

**Table 1.** Scoring algorithm: example.

| Id | Initial | A | B | C | D | E | F | G | H | I | J | K | L | M |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| 1 | 0 | 3 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 3 | 0 | 2 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 | 3 |
| 4 | 0 | 1 | 4 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 | 7 |
| 5 | 0 | 0 | 0 | 2 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 6 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 7 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 8 | 0 | 0 | 0 | 1 | 4 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 | 5 |
| 9 | 0 | 0 | 0 | 0 | 2 | 5 | 7 | 8 | 9 | 9 | 9 | 9 | 9 | 9 |
| 10 | 0 | 0 | 0 | 0 | 1 | 3 | 6 | 9 | 11 | 13 | 13 | 13 | 13 | 13 |
| 11 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 3 | 6 | 9 | 12 | 12 | 12 | 12 |
| 12 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 3 | 4 | 5 | 5 |
| 13 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 14 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 15 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 3 | 5 | 5 |
| 16 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 3 | 6 | 9 |
| 17 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |
| 18 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 |

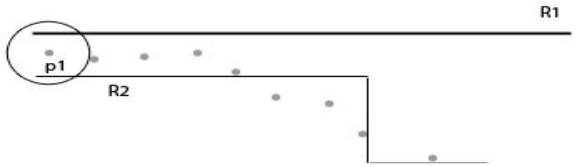



**Fig. 12.** An ambiguous trajectory start

in Java. For storing data, we used the postgreSQL DBMS[3], and the PostGIS[4] extension for handling spatial data. A function *calculatePath(Integer routeId)* computes the path of the GPS points with id *routeId*. The algorithm then combines all steps. First, it retrieves all GPS points from the database with id *routeId*. Then, it limits the road network by computing the streets within each space-time prism, and computes the scores associated with each street. Finally, the $k$-shortest path are computed, returning the path with the highest score. Road segments are stored in a hash table, for fast retrieval.

### 5.1 Experiments

Tests were run on a Compaq 8510p machine, Core(TM)2 Duo CPU T8100 Intel processor, at 2,1 GHz, 4 GB DDR RAM, and a 160 Gb Hard disk size.

[3] http://www.postgresql.org
[4] http://postgis.refractions.net

*Datasets.* We worked with two datasets. The first one corresponds to our emergency service case study presented throughout the paper. The emergency service started recording when they got a call for an intervention, and they stopped recording when they arrived at the intervention site. Every route is a sequence of GPS coordinates (GPS points), stored in $(x, y, t)$ form, where $(x, y)$ represents the location and $t$ is the time where the object was at this location. We studied a total of 33 trajectories, varying from 32 to 433 GPS points, resulting in a total of 6027 GPS points. It is worth noting that a great portion of the available data had to be discarded, mainly for two reasons: (a) they contained less than 10 GPS points; (b) they contain more than 3400 GPS points, indicating that, most likely, the users did not turn off their device as requested. We can see then, that this dataset contains very precise measurements,of homogeneous kind (i.e., all cars are of the same type). On the contrary, our second dataset corresponds to heterogeneous traffic in the Italian city of Milan, recorded during a week. Here, GPS coordinates are recorded at irregular intervals, i.e., data are more imprecise. We considered 30 trajectories varying from 17 to 62 GPS points, a total of 919 GPS points. For both datasets, the route network is represented in a shapefile.

### 5.2 Results

We ran several experiments, aimed at: (a) studying the sensitivity of the space-time prisms method to the selection of the maximal speed (i.e., the size of the space-time prisms). We wanted to find out how this affects precision and execution time, on two datasets of different characteristics; (b) comparing the space-time prisms method against a simple geographic method, which, likely, runs faster, at the expense of obtaining less reconstructed trajectories. Tables 2 and 3 display the results. We report the maximum, minimum, and average execution times (we exclude the time for loading the data), for the two datasets, using the geometric algorithm, and the new uncertainty-based algorithm (indicated as STP in the first column, standing for space-time prism). For the latter, we used the maximum speed as a parameter, ranging from 40km/h to 120km/h.

Table 2 shows that for the space-time prism method with maximum speed 120 km/h and the one with 80 km/h, the results are very similar. However, the difference is significant when the maximum speed is 60 km/h. Note that the speed limit in the city ranges from 50 km/h and 70 km/h. The results suggest then, that the trajectories were not recorded during an urgent intervention that requires driving faster than these limits. Then, since execution times are better for a maximum speed of 80 km/h (it considers less roads), it would be better, in this case, to use this speed for analysis.

For the Milan data (Table 3), results were different, due to the fact that the GPS data was not as dense as in the first case. In consequence, the space-time prisms between two points are larger, and include more roads, which the algorithm has to process. The Milan GPS data consists of three types of trajectories. The first type are trajectories that do not enter the center of the city (e.g. people that park their car and use the public transport in the center). The second type are trajectories that come from outside the city and enter the center. The last

type are the trajectories recorded within the center of the city itself. When we look at the results we notice that when the speed limit decreases, the number of trajectories that can be reconstructed also decreases. This is because the GPS points are further away from each other, compared with our first dataset.

**Table 2.** Results for the emergency service data

| Algorithm | Min. (msec) | Avg.(msec) | Max.(msec) | Reconstructed trajectories(% ) |
|---|---|---|---|---|
| Geometric | 50,56 | 141,32 | 472,66 | 78,8% |
| STP 120km/h | 103,49 | 246,23 | 825,45 | 96,7% |
| STP 100km/h | 99,42 | 219,93 | 616,80 | 93,4% |
| STP 80km/h | 62,06 | 193,47 | 445,88 | 90,1% |
| STP 60km/h | 60,24 | 177,09 | 449,21 | 54,5% |
| STP 40km/h | 58,38 | 167,23 | 455,82 | 15,1% |

**Table 3.** Results for the Milan data

| Algorithm | Min. (msec) | Avg.(msec) | Max.(msec) | Reconstructed trajectories(% ) |
|---|---|---|---|---|
| Geometric | 79,69 | 134,99 | 193,98 | 13,3% |
| STP 120km/h | 1132,82 | 2680,88 | 13665,48 | 96,7% |
| STP 100km/h | 902,73 | 1889,39 | 8303,49 | 83,3% |
| STP 80km/h | 778,41 | 1452,79 | 5475,36 | 53,3% |
| STP 60km/h | 674,68 | 1066,19 | 1957,36 | 26,73% |
| STP 40km/h | 561,39 | 845,75 | 1244,86 | 10,0% |

## 6 Conclusion and Future Work

We presented a novel map matching algorithm that, unlike previous efforts, accounts for the uncertainty of the observations recorded by GPS devices, using the notion of space-time prisms. Experimental results on two real-world datasets, very different from each other, showed that, particularly for unprecise observations, taken at relatively large intervals, our algorithm allows reconstructing most of the trajectories. We also showed that, in such cases, geometric algorithms that do not consider uncertainty, are unable to reconstruct most of the trajectories.

## References

1. S. Brakatsoulas, D. Pfoser, R. Salas, and C. Wenk. On map matching vehicle tracking data. In *VLDB '05: Proceedings of the 31st international conference on Very large data bases*, pages 853–864. VLDB Endowment, 2005.

2. M. Egenhofer. Approximation of geospatial lifelines. In *SpadaGIS, Workshop on Spatial Data and Geographic Information Systsems*, 2003. Electr. proceedings, 4p.
3. J. Greenfeld. Matching GPS observations to locations on a digital map. *Proceedings of the 81th Annual Meeting of the Transportation Board*, 2002.
4. R. Güting and M. Schneider. *Moving Object Databases.* Morgan Kaufmann, 2005.
5. T. Hägerstrand What about People in Regional Science? *Papers of the Regional Science Association* vol.24,pp.7-21,1970.
6. P. Hart, N. J. Nilsson and B. Raphael. A Formal Basis for the Heuristic Determination of Minimum Cost Paths *IEEE Transactions on Systems Science and Cybernetics*, vol. 4 (2),100–107, 1972.
7. K. Hornsby and M. Egenhofer. Modeling moving objects over multiple granularities. *Annals of Mathematics and Artificial Intelligence*, 36(1–2):177–194, 2002.
8. B. Kuijpers, W. Othman and R. Grimson. A case study of the difficulty of quantifier elimination in constraint databases: the alibi query in moving object databases. *CoRR*, abs/0712.1996, 2007.
9. H. Miller. A measurement theory for time geography. *Geographical Analysis 37,* 1, 17–45, 2005..
10. W. Y. Ochieng, M. Quddus and R. B. Noland. Map-Matching In Complex Urban Road Networks. *Brazilian Journal of Cartography*, 55(2), 1–18, 2003.
11. D. Pfoser and C. S. Jensen. Capturing the uncertainty of moving-object representations. In *Advances in Spatial Databases (SSD'99)*, volume 1651 of *Lecture Notes in Computer Science*, pages 111–132, 1999.
12. M . A. Quddus, W. Y. Ochieng, Z. Lin and R. .B. Noland. A general map matching algorithm for transport telematics applications. *GPS Solutions*, 73, 157–167, 2003.
13. J. Su, H. Xu and O. Ibarra. Moving objects: Logical relationships and queries. In *Advances in Spatial and Temporal Databases (SSTD'01)*, volume 2121 of *Lecture Notes in Computer Science*, pages 3–19. Springer, 2001.
14. G. Taylor. Road reduction filtering for gps-gis navigation. *Transactions in GIS*, 5:193–207(15), June 2001.
15. E. Weisstein. Wolfram Mathworld. http://mathworld.wolfram.com/DijkstrasAlgorithm.html.
16. C. E. White, D. Bernstein and A. L. Kornhauser. Some map matching algorithms for personal navigation assistants. *Transportation Research C*, Vol. 8, 91–108, 2000.
17. O. Wolfson. Moving objects information management: The database challenge. In *Proceedings of the 5th Intl. Workshop NGITS*, pages 75–89. Springer, 2002.
18. J. Y. Yen. Finding the lengths of all shortest paths in N-Node Nonnegative-Distance Complete Networks Using $\frac{1}{2}N^3$ Additions and $N^3$ Comparisons. *Journal of the ACM (JACM)*, 19(3), 423 – 424,1972.