Towards traffic-aware routing using GPS vehicle trajectories

Carola Wenk
University of Texas at San Antonio
carola@cs.utsa.edu

Collaboration with:

- Dieter Pfoser, Computer Technology Institute, Athens, Greece
- Peter Wagner, German Aerospace Center, Berlin, Germany

Outline

1. Problem Description

   Enable in-car navigation systems to find the best routes using current traffic situation
2. Travel Times and GPS curves
3. Map-Matching
   1. Incremental map-matching
   2. Global map-matching: Fréchet distance
   3. Global map-matching: Weak Fréchet distance
4. Routing System Setup and Future Work

In-Car Navigation Systems

- Navigation systems perform the **routing task**:

  Find a shortest route from A to B

- What does “shortest” mean?
  - Shortest length? No.
  - Shortest travel time !

Model for Routing Task

- Model the street network as a graph:
  - Vertices: Intersections of roads
  - Edge: A road segment between two intersections

Model for Routing Task

- Routing Task: Find shortest path in the graph from ***A*** to ***B***

How to Compute Shortest Paths?

- Dijkstra's shortest path algorithm:
  - Given ***A***, ***B***, and a graph with non-negative edge weights
  - Among all paths from ***A*** to ***B*** in the graph, compute such a path whose total weight (= sum of edge weights) is minimized
- What are our edge weights?
  - Travel times
  - The travel time on a (directed) edge from ***c*** to ***d*** is the time it takes to travel from ***c*** to ***d***

Outline

1. Problem Description

   Enable in-car navigation systems to find the best routes using current traffic situation

2. Travel Times and GPS curves
3. Map-Matching
   1. Incremental map-matching
   2. Global map-matching: Fréchet distance
   3. Global map-matching: Weak Fréchet distance
4. Routing System Setup and Future Work

Travel Time of an Edge

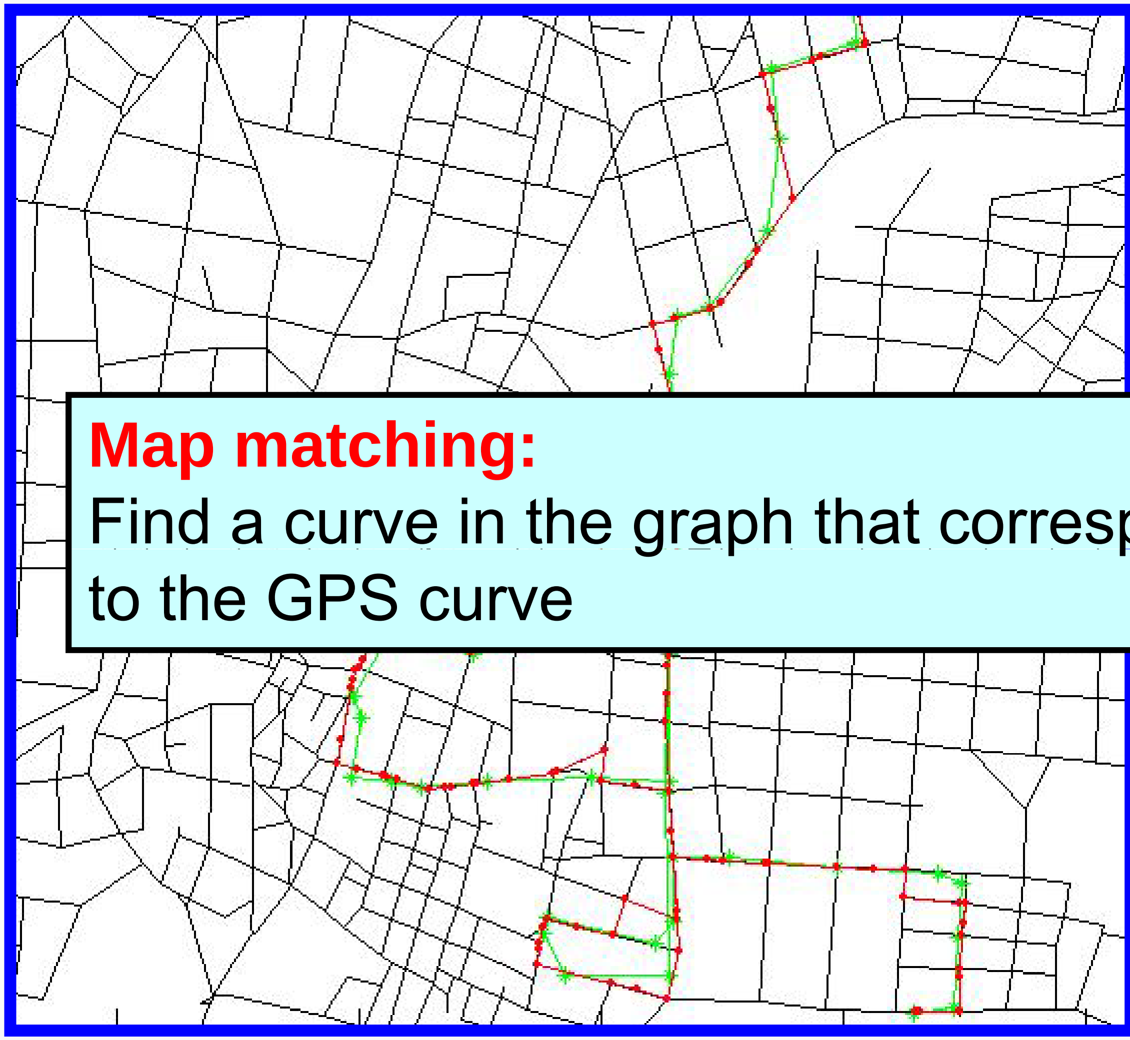
- How do we know the travel time on an edge / road segment?
  - Usually, navigation systems derive travel times from speed limits.
  - A very smart system might take a small number of congestion points into account
  - Usually variation of travel times during rush hour is not taken into account
- New approach: Maintain a database of current travel times
  - Use GPS trajectory data from vehicle fleets (delivery trucks, taxis, etc.)
- Challenge: How do we build this database?

GPS Floating Car Data

- Floating car data (FCD)

  A sequence (trajectory) of data points, each consisting of:
  - Basic vehicle telemetry, e.g., speed, direction, ABS use
  - The *position of the vehicle* (→ tracking data) obtained by GPS tracking
  - A time stamp
- Traffic assessment
  - Data from one vehicle as a sample to assess the overall traffic condition – cork swimming in the river
  - **Large amounts of tracking data** (e.g., taxis, public transport, utility vehicles, private vehicles)
    → Accurate picture of the traffic condition
  - **Tracking data needs to be related to the road network**
    → **Map matching**
  - **Time stamps** from FCD yield **travel times** for road segments

GPS Vehicle Tracking Data



Map matching:
Find a curve in the graph that corresponds to the GPS curve

Roadmap of Athens, Greece — GPS curve — Corresponding path in the roadmap

Problems:

1) Measurement error: GPS points do not exactly lie on the roadmap

...npling error: ...S curve is a by-...duct, and usually ...pled every 30s

⇒ The GPS curve does not lie on the roadmap

Outline

1. Problem Description

   Enable in-car navigation systems to find the best routes using current traffic situation

2. Travel Times and GPS curves
3. Map-Matching
   1. Incremental map-matching
   2. Global map-matching: Fréchet distance
   3. Global map-matching: Weak Fréchet distance
4. Routing System Setup and Future Work

Available Map-Matching Algorithms

- Incremental map-matching
  - Follow greedy strategy of incrementally extending solution from an already matched edge, e.g., [BPSW05]
  - No quality guarantee
  - Classical approach
- Global map-matching
  - Find among all possible trajectories in the road network the one that is most similar to the vehicle trajectory
  - **Distance measure assesses similarity** = quality guarantee
  - Fréchet distance (strong, weak) [BPSW05, WSP06]

BPSW05: ``On Map-Matching Vehicle Tracking Data'' (S. Brakatsoulas, D. Pfoser, R. Salas, and C. Wenk), *Proc. 31st Conference on Very Large Data Bases (VLDB)*: 853-864, 2005, Trondheim, Norway.

WSP06: ``Addressing the Need for Map-Matching Speed: Localizing Global Curve-Matching Algorithms''(C. Wenk and R. Salas and D. Pfoser), *Proc. 18th International Conference on Scientific and Statistical Database Management (SSDBM)*: 379-388, 2006, Vienna, Austria.

Outline

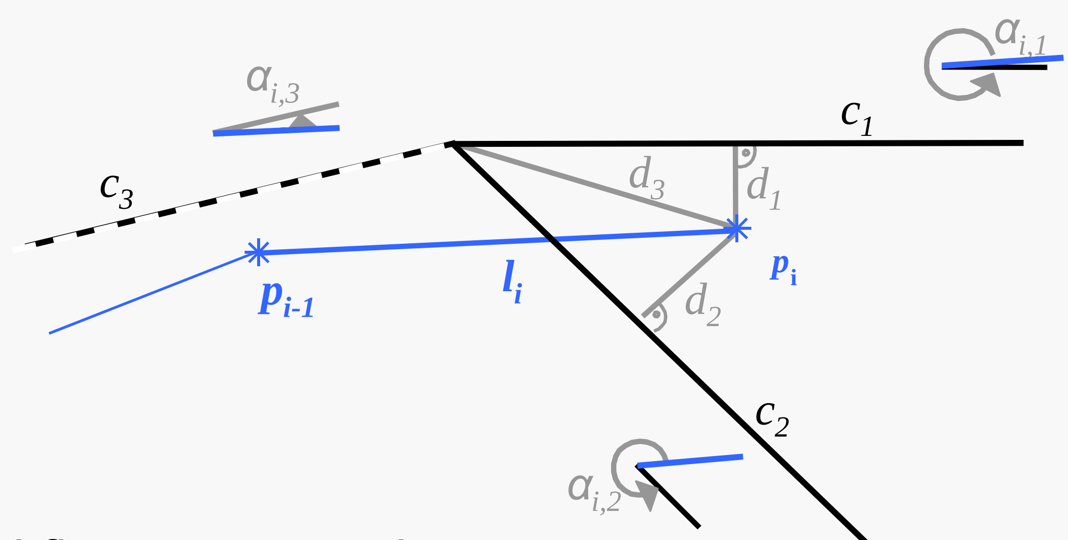
1. Problem Description

   Enable in-car navigation systems to find the best routes using current traffic situation

2. Travel Times and GPS curves
3. Map-Matching
   1. Incremental map-matching
   2. Global map-matching: Fréchet distance
   3. Global map-matching: Weak Fréchet distance
4. Routing System Setup and Future Work

Incremental Map-Matching

- Position-by-position, edge-by-edge strategy for map-matching



  - Initialization: Find first correspondence using a spatial range query. Assume the graph has been preprocessed for spatial range queries.
  - Evaluate for each trajectory edge (or GPS point) all road network graph edges incident to the last vertex.
  - Runtime O(*nd* + *log m*), with *n* being the number of GPS points in the trajectory, m the size of the graph, and *d* the maximum degree of any vertex in the graph. In practice *d* is a constant and O(*nd)* dominates the runtime.

Outline

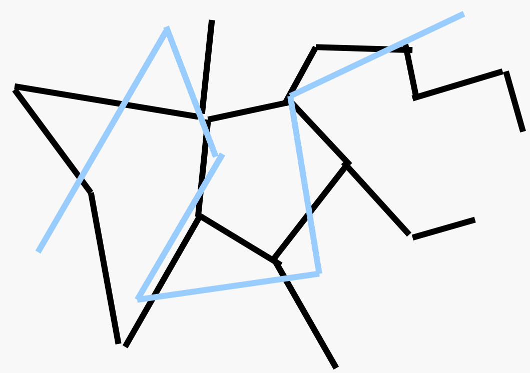
1. Problem Description

   Enable in-car navigation systems to find the best routes using current traffic situation

2. Travel Times and GPS curves
3. Map-Matching
   1. Incremental map-matching
   2. Global map-matching: Fréchet distance
   3. Global map-matching: Weak Fréchet distance
4. Routing System Setup and Future Work

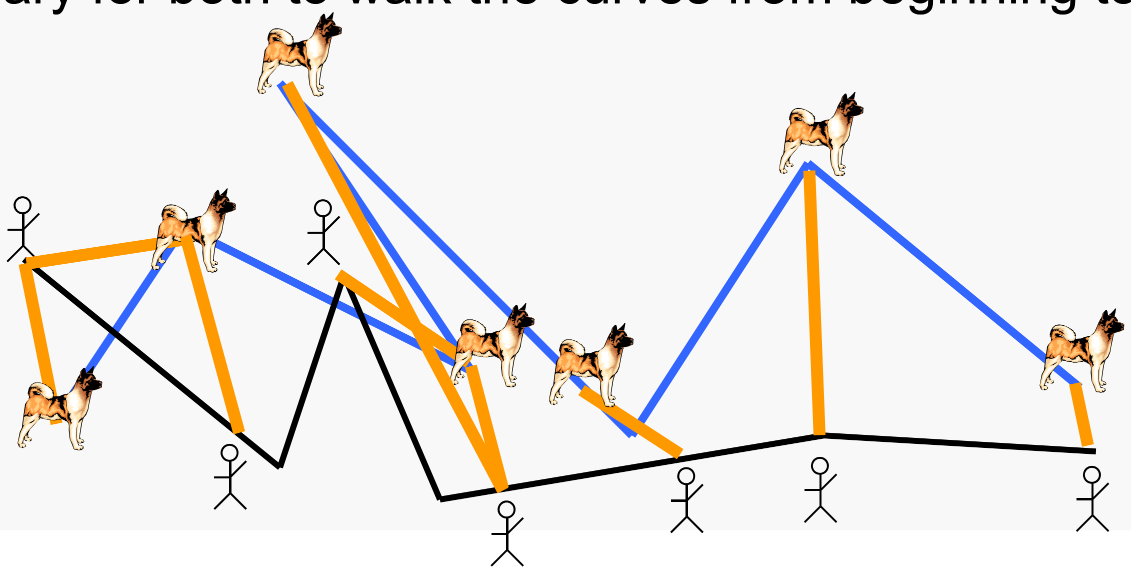
Global Map-Matching

- Find a ***curve in the road network*** that is as ***close as possible to the vehicle trajectory***
- Curves are compared using
  - Fréchet distance and
  - Weak Fréchet distance
- ***Minimize over all possible curves in the road network***



Fréchet Distance

- Dog walking example
  - Person is walking his dog (person on one curve and the dog on other)
  - Allowed to control their speeds but not allowed to go backwards
  - Fréchet distance of the curves: ***minimal leash length*** necessary for both to walk the curves from beginning to end

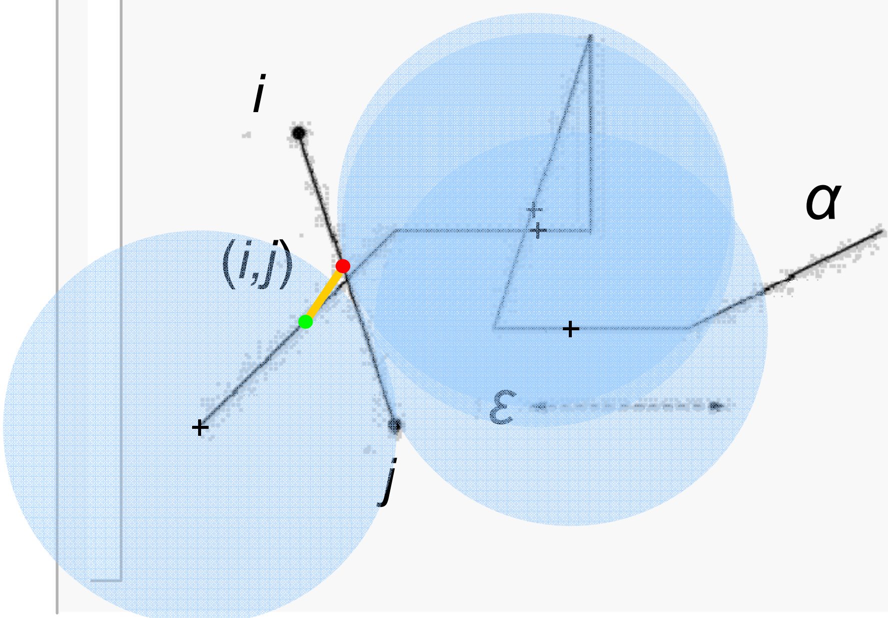


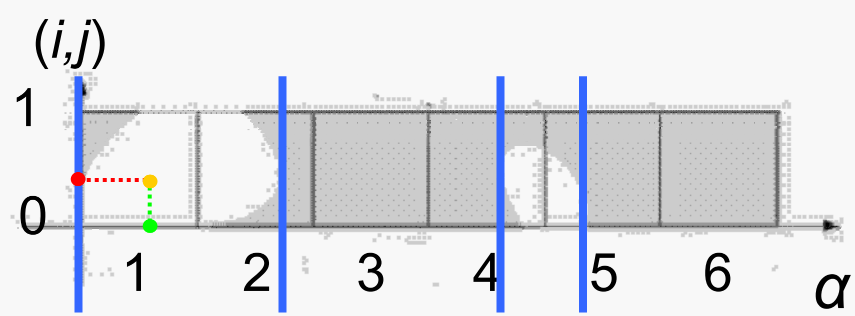
Fréchet Distance

- Fréchet Distance
  - $$\delta_F(f,g) := \inf_{\alpha,\beta:[0,1]\to[0,1]} \max_{t\in[0,1]} \left\| f(\alpha(t)) - g(\beta(t)) \right\|$$
  - where *α* and *β* range over continuous non-decreasing reparametrizations only
- Weak Fréchet Distance
  - $\tilde{\delta}_F(f,g)$
  - drop the non-decreasing requirement for *α* and *β*
  - $\tilde{\delta}_F(f,g) \le \delta_F(f,g)$
- Well-suited for the comparison of trajectories since they take the continuity of the curves into account

Free Space Diagram

- **Decision variant** of the global map-matching problem
  - For a fixed $\varepsilon > 0$ decide whether there exists a path in the road network with distance at most ε to the vehicle trajectory α
- For each (straight-line) edge (i,j) in a graph G let its corresponding Freespace Diagram $FD_{i,j} = FD(\alpha, (i,j))$

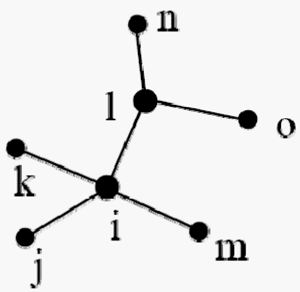


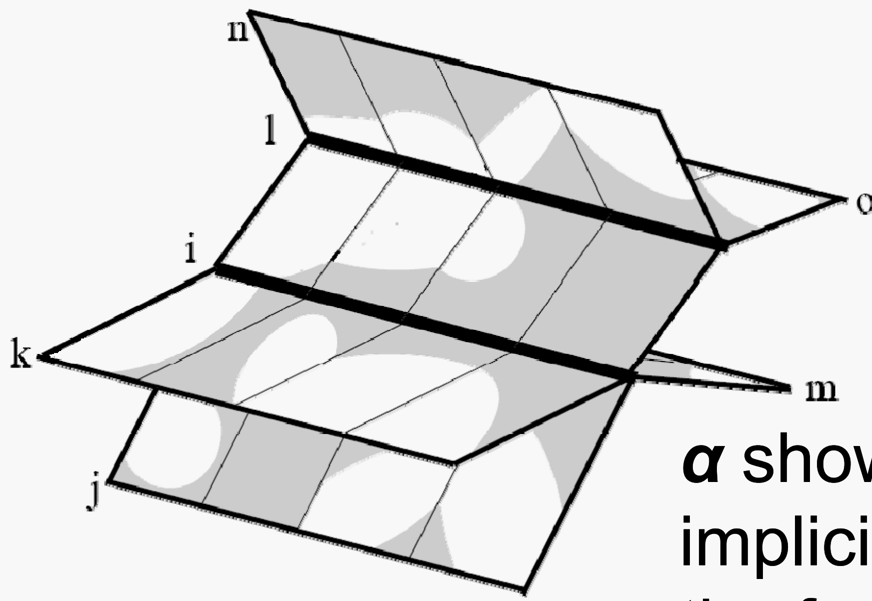


Free Space Surface

- Glue free space diagrams $FD_{i,j}$ together according to adjacency information in the graph *G*
- Free space surface of trajectory *α* and the graph *G*

G

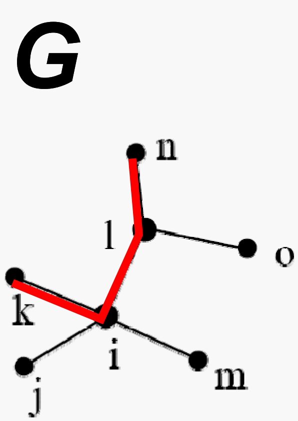


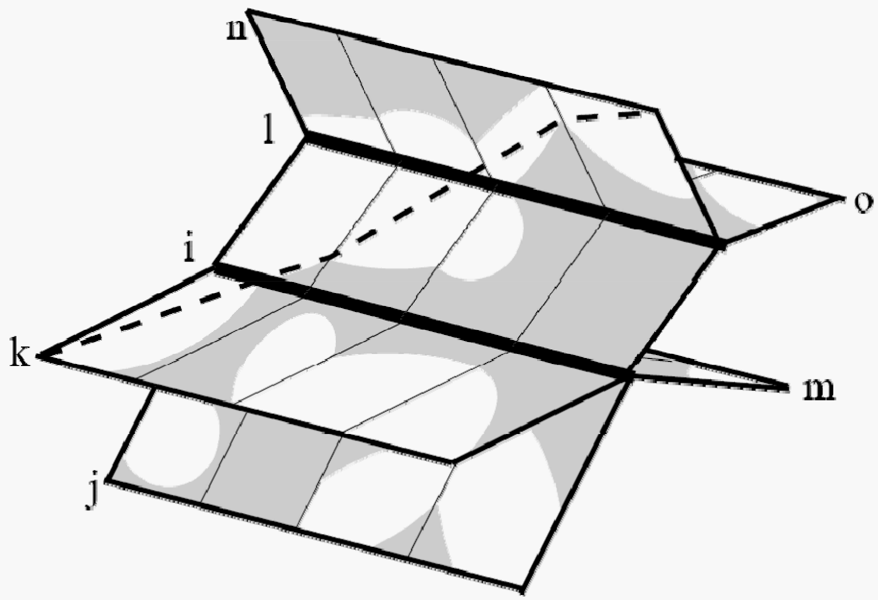


α shown implicitly by the free space surface

Free Space Surface

- TASK: Find **monotone path** in free space surface
  - starting in some lower left corner, and
  - ending in some upper right corner





Map-Matching Using the Fréchet Distance

- Find monotone path in free space surface using dynamic programming
  - O(*mn* log *mn*) time for fixed ε, where *m* is the size of the graph and *n* the size of the trajectory
- *Solve **minimization problem*** for ε using binary search
  - O(*mn* log *mn* log *b*) time, where *b* is the desired bit precision
- Conclusion
  - Finds a curve in the graph together with a given quality guarantee (= Fréchet distance to the GPS curve)
  - The algorithm needs O(*mn*) space which for large graphs will cause a large overhead
  - ⇨ Accurate map-matching results
  - ⇨ Slow algorithm

Outline

1. Problem Description

   Enable in-car navigation systems to find the best routes using current traffic situation

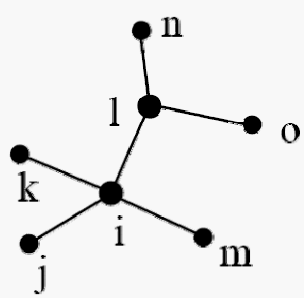
2. Travel Times and GPS curves
3. Map-Matching
   1. Incremental map-matching
   2. Global map-matching: Fréchet distance
   3. Global map-matching: Weak Fréchet distance
4. Routing System Setup and Future Work

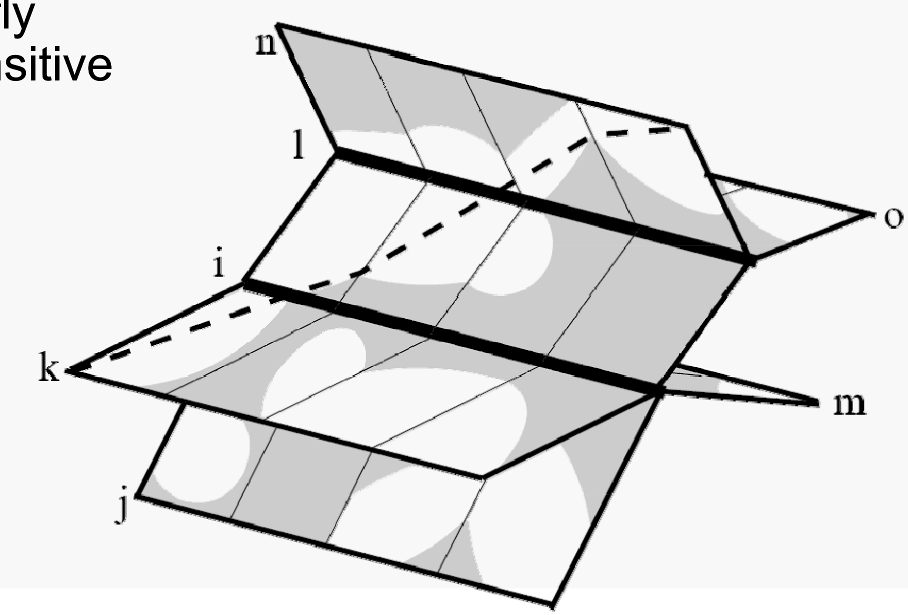
Adaptive Clipping Map-Matching Algorithm

- Uses the weak Fréchet distance
- Output sensitive Map-Matching (algorithmic improvement)
  - Construct and traverse free space graph (which has quadratic complexity) on the fly
  - Improved runtime: O(K log K), where K is the size of the traversed free space which in our experiments is much smaller than *mn*
- Error-Aware Map-Matching (metadata information)
  - Use known GPS error sources to define those trajectories in the roadmap that could have led to the observed vehicle trajectory
- **Adaptive Clipping algorithm**:
  - Solves this error-aware map-matching task
  - Corresponds to pruning/clipping Fréchet-based algorithms

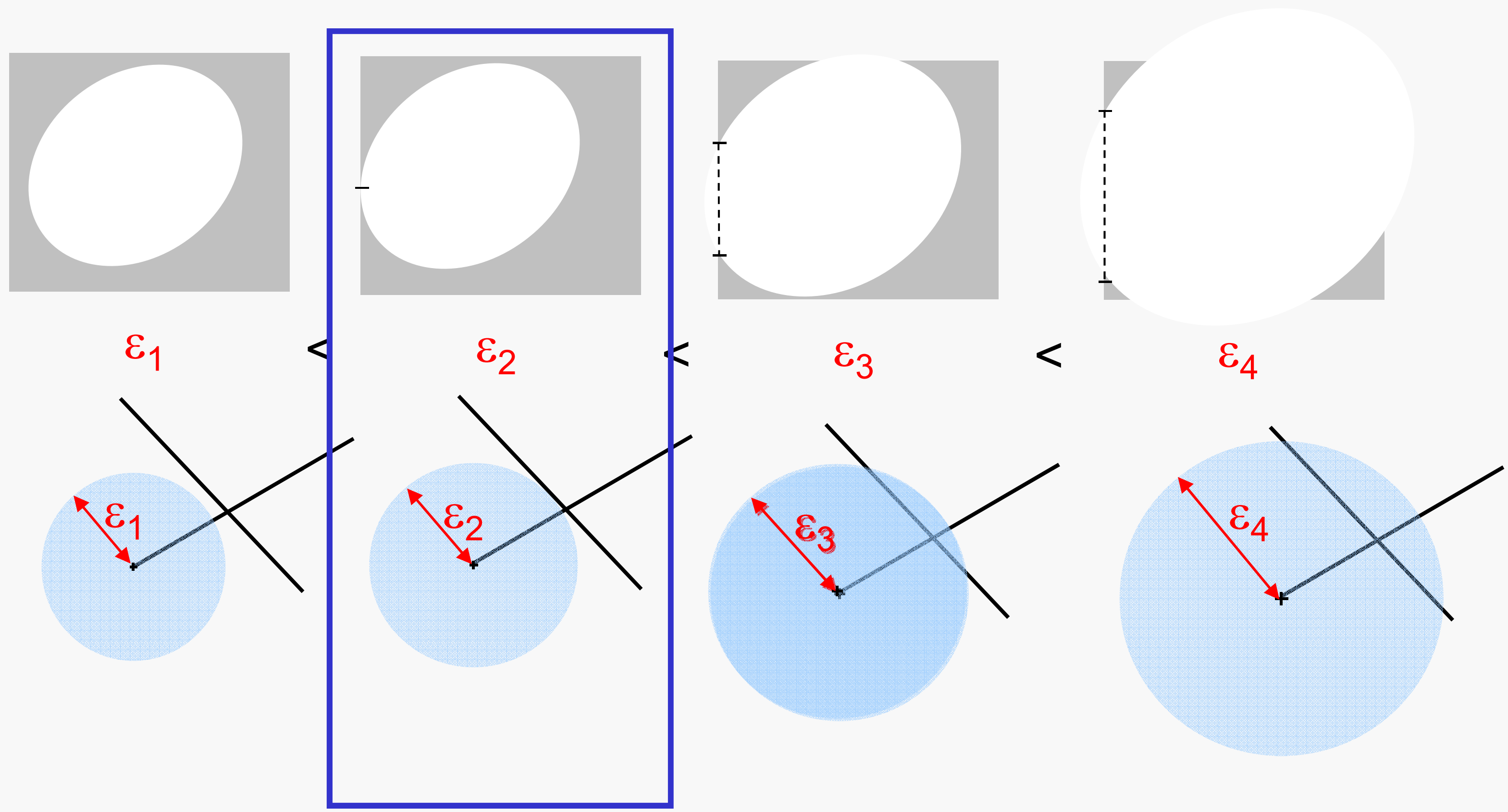
Output-Sensitive Map Matching

- The **weak** Fréchet distance requires finding **any** (possibly non-monotone) path in the free space surface
- Express the problem as finding a shortest path in the **Free Space Graph**
  - Allows running Dijkstra's shortest path algorithm, which also finds ε (no binary search needed)
  - Allows constructing only the **traversed** portion of the free space surface / graph on the fly
    → (Pseudo) output-sensitive





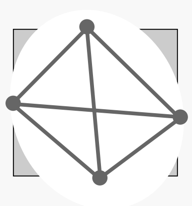
Free Space Cell

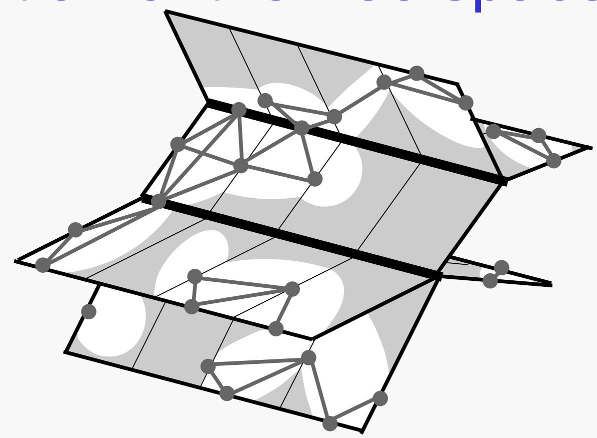


Store ε_2 as the **weight** of the vertical free space boundary

Free Space Graph

- Encodes connectivity information of the free space





- For each cell boundary store optimal ε as its weight
- Weak Fréchet distance = maximum of all ε's along an optimal path in the free space graph

⇨ Need to find a **shortest path (with minimum total ε)**

⇨ Run Dijkstra's algorithm on the free space graph

⇨ Free space graph can be constructed **on the fly**

Algorithm

- Running Time
  - O(K log K), with K being the size of the **traversed** free space graph
  - K = O(mn) in the worst case for traversing all the graph
  - BUT, stops when shortest path to end vertex is found

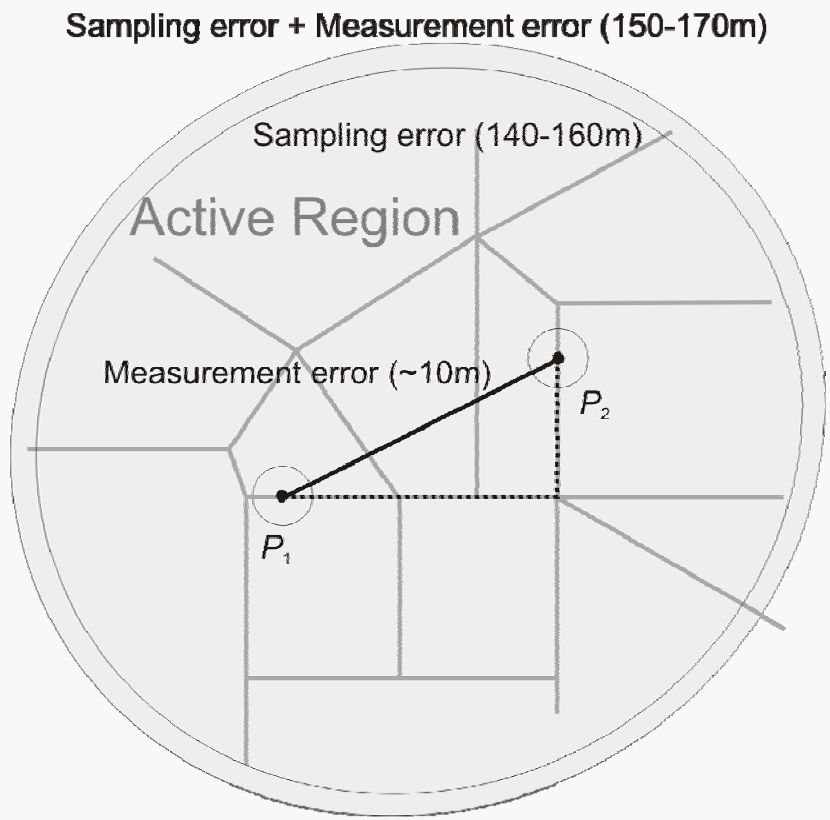
⇨ Need to find a **shortest path (with minimum total ε)**

⇨ Run Dijkstra's algorithm on the free space graph

⇨ Free space graph can be constructed **on the fly**

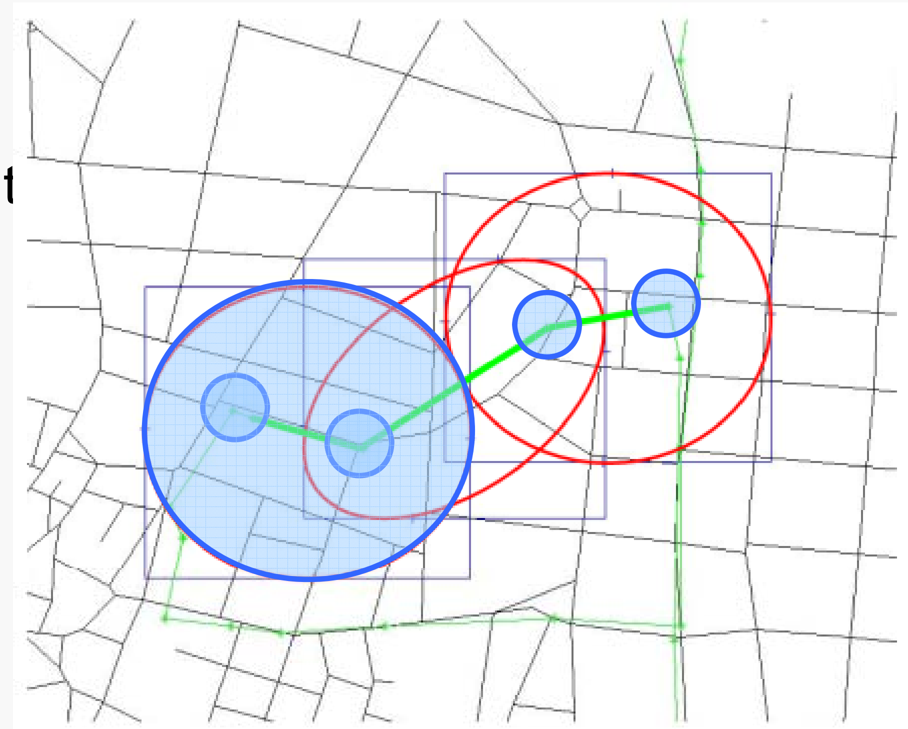
Error-Aware Map Matching

- Considering data acquisition errors
  - measurement error
  - sampling error
- **Active Regions**
  - areas delimiting possible positions
- Sequence of all active regions → **error-aware representation** of the vehicle trajectory



Error-Aware Map Matching

- Find curve that fulfills the following properties
  - Starts at the origin
  - Stops at the destination
  - Intersects measurement error disks around all position samples
  - Is within active regions



Adaptive Clipping Algorithm

- Incremental algorithm

  For **each active region** of an edge

  - Run output-sensitive weak-Frechet/Dijkstra algorithm
  - **Stitch Dijkstra graphs together** at active regions of vertices
  - Construct overall result by tracing back stitched-together Dijkstra graphs

⇨ Solves error-aware map-matching task

⇨ The clipping immensely reduces K. In practice K is almost constant.

⇨ The algorithm is very fast, almost O(n log n) runtime

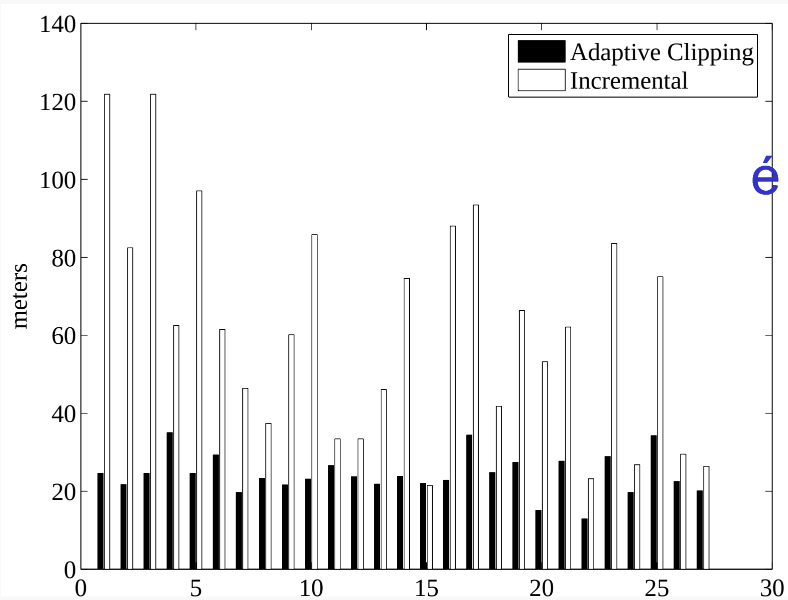
Empirical Evaluation

- GPS vehicle tracking data
  - 27 trajectories
    (~16k GPS points)
  - sampling rate 30 seconds
  - max speed 80km/h
- Road network data
  - vector map of Athens, Greece
    (40 x 40km)
- Evaluating matching quality
  - results from ***incremental*** vs. ***global*** method
  - Averaged Fréchet distance (uses sampled integral / sum instead of maximum)

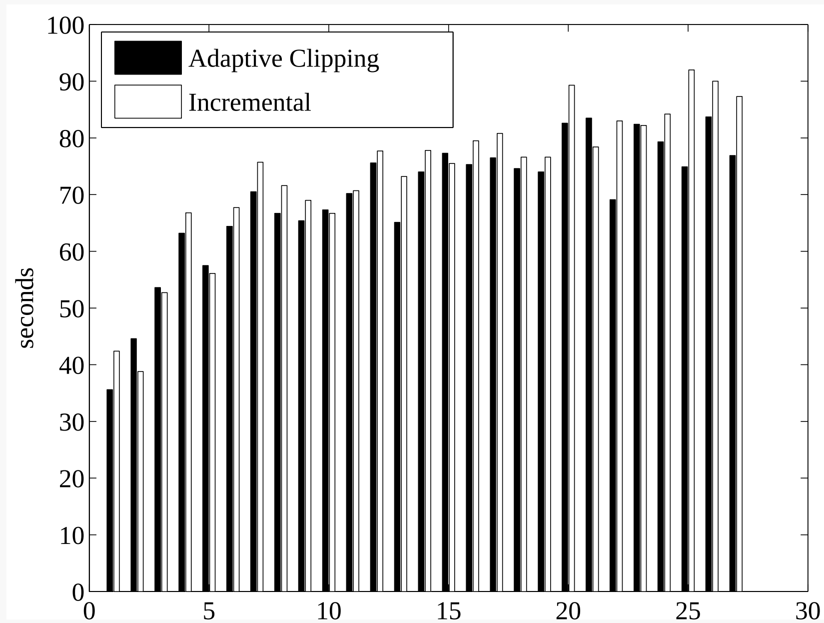
Empirical Evaluation

- **Matching quality**

(Use average Fréchet distance as a quality measure)



- **Matching speed**
  - Adaptive Clipping runs as fast as the Incremental Algorithm



⇨ Adaptive Clipping is **fast** and yields **good matching quality**

Outline

1. Problem Description

   Enable in-car navigation systems to find the best routes using current traffic situation

2. Travel Times and GPS curves
3. Map-Matching
   1. Incremental map-matching
   2. Global map-matching: Fréchet distance
   3. Global map-matching: Weak Fréchet distance
4. Routing System Setup and Future Work

Routing System Setup (work in progress)

- Maintain a database of current travel times
  - Update database with **current feed of GPS data**
  - Test system in Athens gets new data every 5 minutes
  - Data provided by vehicle fleet systems which already have an uplink for GPS data transfer between the vehicle and a central server
- Communication with in-car navigation system
  - Through GPRS (cell phone) – not yet implemented

central server

in-car client

Client Access to Travel Time Database (work in progress) [KW07]

- Local computation model:
  - Client computes routes itself
  - Updated travel times (in neighborhood of initial route) are transferred when they differ too much from original travel time
- Central computation model:
  - Central server computes routes

KW07: ``Dynamic Routing'' (N. Kalinowski and C. Wenk), *Technical Report CS-TR-2007-005, Department of Computer Science,* University of Texas at San Antonio, 2007.

Real-World Applications

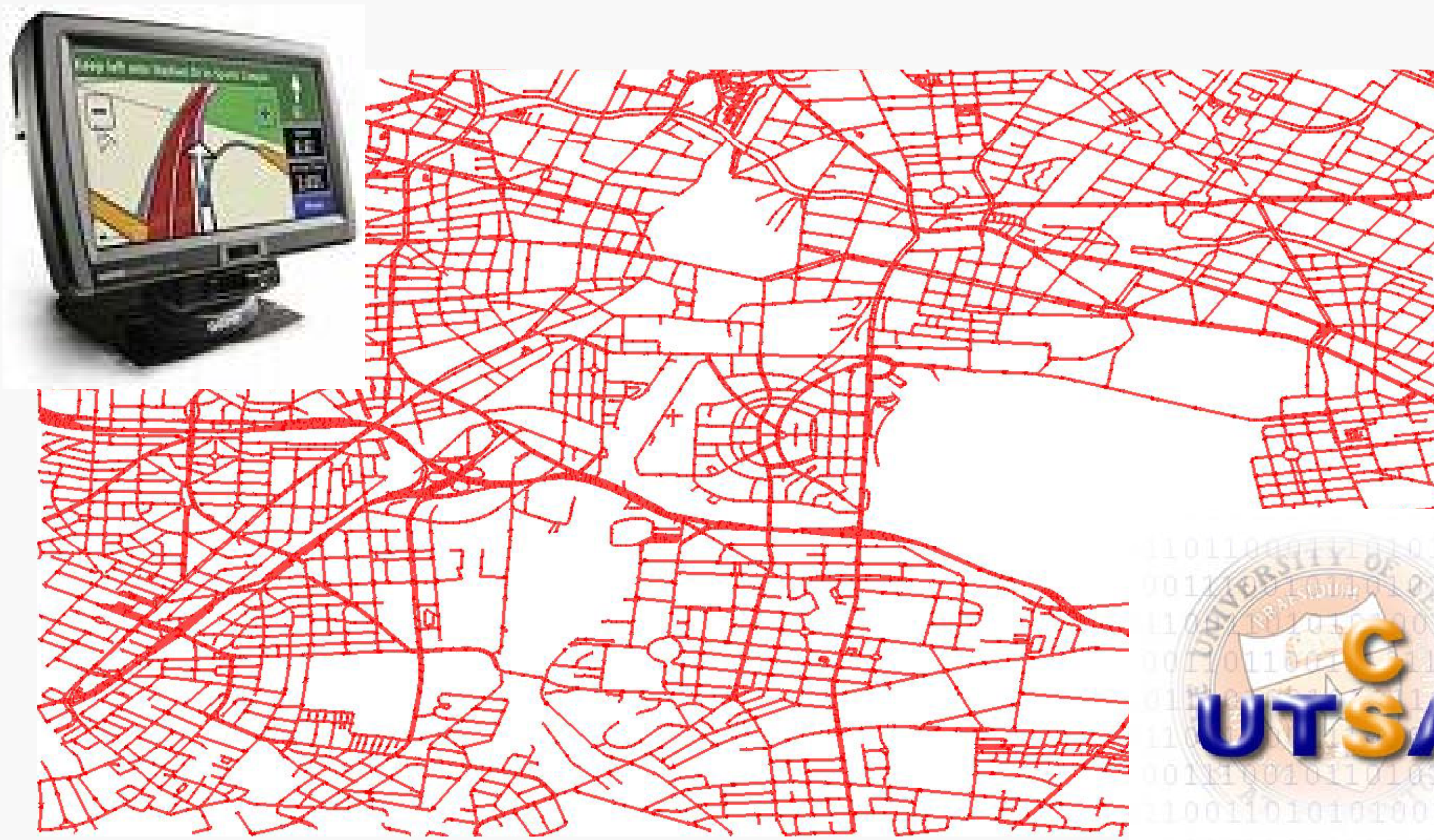
- This is an extremely hot research area
- Almost every car manufacturer and GPS receiver manufacturer is building a *similar* prototype system
  - Different models for GPS data collection and client-server setup / communication
  - GPS manufacturers download “old” GPS data offline from GPS receivers
    $\rightarrow$ Not current travel times

Other Types of Tracking Data

- GPS drawbacks
  - GPS receiver needs line of sight to satellites.
    → Does not work inside buildings, and has problems in cities with very tall buildings
- Other types of tracking data
  - Other positioning technology (wireless networks, GSM)
    → Measurement error is much higher
    → Pilot project in Athens to collect GSM signal strength data
  - Type of moving objects (planes, people)

Thank You

Thank you for your attention



Quality of Matching Result

- Comparing Fréchet distance of original and matched trajectory
- Fréchet distances strongly affected by outliers, since it takes the ***maximum*** over a set of distances.

$$\delta_F(f,g) := \inf_{\alpha,\beta:[0,1]\to[0,1]} \max_{t\in[0,1]} \left\| f(\alpha(t)) - g(\beta(t)) \right\|$$

- Average Fréchet distance – replace the maximum with a path integral over the reparametrization curve (α(t),β(t)):

$$\delta_F(f,g) := \inf_{\alpha,\beta:[0,1]\to[0,1]} \int_{(\alpha,\beta)} \left\| f(\alpha(t)) - g(\beta(t)) \right\|$$

  - Remark: Dividing by the arclength of the reparametrization curve yields a normalization, and hence an „average“ of all distances.