Enhancing Vehicle Navigation using Shared Monitoring and Estimation

Rudi Ball
Department of Computing
Imperial College
London, United Kingdom
Email: r.ball@imperial.ac.uk

Naranker Dulay
Department of Computing
Imperial College
London, United Kingdom
Email: n.dulay@imperial.ac.uk

***Abstract*—Presently, travel time estimations and directions do not take into account contextual information about a route, such as, the effects of traffic, population density, weather, accidents or road conditions that may influence travel. As such, estimations and routes provided may be sub-optimal in their ability to route a vehicle to a destination. By monitoring its own and shared travel time delays, a vehicle may more precisely estimate travel times and thereby route itself more effectively across a road network. This paper presents a decentralised vehicular service for travel time estimation. Vehicles monitor and collect mobility data and share this mobility data using broadcast based Vehicle-to-X (V2X) messaging. Using a three phased protocol we investigate issues pertaining to the storage, growth and relevance of stored travel time data. The approach is simulated and results obtained demonstrate the feasibility of such a system while highlighting trade-offs.**

I. Introduction

Which route takes the shortest time from my present position to my destination? Within the transportation domain this is an old and challenging problem. Thus far estimation has been used to predict route performance given a set of known data. Idealy, to estimate the travel time between two geographic positions, estimation algorithms should consider a variety of data, however much contextual data such as travel delay data is difficult to collect and as such cannot be used. Within this paper we refer to *travel time estimation* as the process of estimating the elapsed time required to travel from a start position (A) to an end position (B) using a road network [1]. Navigation computers often refer to their own estimation to a destination as the *estimated time of arrival* (ETA). There are several methods to solving the problem of travel time estimation in the context of motor vehicles on road networks. One possible solution would be to monitor every road section of a road network at every time instance. In reality, this may be infeasible given infrastructure deployment and maintenace costs. Each road section would require its own vehicle counters and sensors to measure vehicle movements and consequently each sensor would need to be maintained.

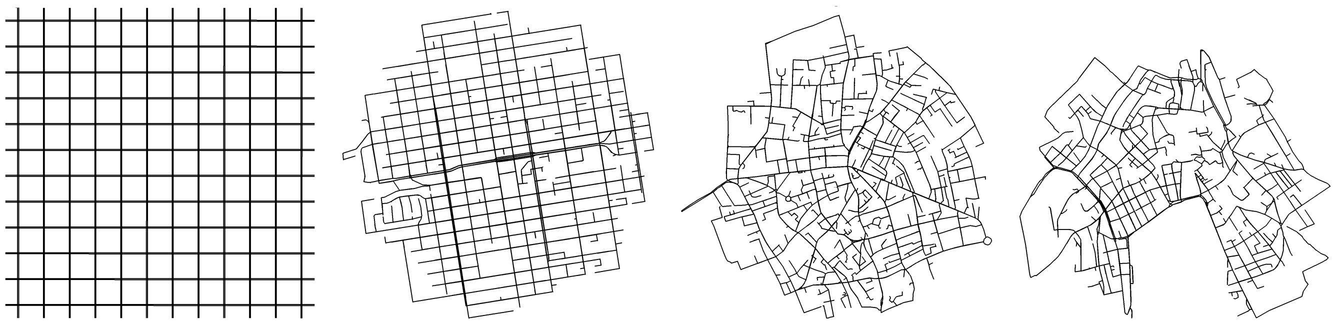
This paper presents an alternative decentralised approach where vehicles automatically monitor, collect and share experienced travel times to model the state of the road network over time. In contrast to other approaches [2], [3] the service is ad-hoc. The approach considers an opposing extreme as it does not use any central repository for the storage or processing of travel time data in its architecture. Rather, vehicles monitor and collect their own travel times and share these travel times with neighbouring vehicles where able to do so. Vehicles communicate and disseminate data using ad-hoc broadcast messages provisioned by the Vehicle-to-X (V2X) wireless standard (IEEE 802.11p)[4]. Using the Geographic Urban Simulator (GUS) we develop a service as it would exist once deployed. The service is written as a real protocol, but simulated using the GUS. Simulations present us with results and visualisations concerning the probable performance and feasibility of the service.

II. Scenario, Aims and Assumptions

The scenario considered is defined as using a *static scenario* [5]. Beginning her or his journey, a driver of a vehicle queries their on-board navigational computer for directions to reach a destination. Using a database, the driver is presented with a set of driving directions and a route specifying how a vehicle can reach its destination. Included with these directions is an ETA and a displacement estimation. We assume that the route provided represents the estimated shortest time between two positions using a shortest path algorithm [6], [5] and vehicle speed data (road speed limits and vehicle performance data).

For a given journey ETA represents a predicted future time instant when a vehicle is expected to reach its destination, while the *Actual Time of Arrival* (ATA) represents the time recorded by a vehicle reaching its destination. Hence, the performance of estimation is calculated as the error between the initial ETA and ATA. A combination of route and directions is used by the driver to reach her or his intended destination. We assume that drivers follow these directions without modifying their route. The protocol developed addresses the following questions:

1) What estimations can be made about unknown travel times?
2) What is the communication overhead of such a service? (Section VI-D)
3) How long does it take for a vehicle to find all travel times *relevant* to itself? (Section VI-C)
4) What estimations can be made from known travel times? (Section VI-C)



(a) Grids (e.g. 12x12). (b) San Francisco. (c) London. (d) Zurich [7].

Fig. 1. Road networks (2.5 x 2.5km).

5) Can we produce travel time maps comparable to those provided by related works [3]? (Section VI-B)
6) Can we produce travel time graphs usable for dynamic routing scenarios [5]? (Section VI-B)

We assume that vehicle mobility is constrained to each vehicle's specified route. Vehicles have the capability of finding their own position, measuring the time to travel between sub-positions and communicating wirelessly with neighbouring vehicles using ad-hoc WiFi (IEEE802.11p WAVE standard [4]).

III. Mobility Traces

While a number of mobility datasets have recorded road use within cities, their characteristics and type (for example bus, taxi and car networks) makes them inappropriate for use within our application scenario. For the purposes of experimentation we used two source datasets, namely: (a) *generated bespoke traces* and (b) traces from the *ETH Realistic Vehicular Traces dataset* [8]. Bespoke traces were generated using routing directions provided by Google Maps[1] and Open Street Maps[2]. Given a chosen start and end position, directions were queried, just as they would be by a driver. Resulting mobility patterns are seen to tend to use main roads rather than arterial roads. Mobility patterns are thus more consistent with mobility patterns observed in reality, where a sizeable proportion of drivers use their GPS navigation systems to route themselves through a city. The prevalence of traffic along specific road sections is subsequently seen in results produced. The sum patterns of mobility traces for separate sample areas can be seen in Figure 1. The secondary, ETH trace dataset has been used by a selection of previous works [9], [10]. ETH traces [8] represent realistic traces generated by the MMTS model. The MMTS model has been used to model the behaviour of the inhabitants of Switzerland using statistical census data. The traces represent 24 hours of vehicular mobility for a 250 x 250 kilometer coverage of Switzerland. Traces were enhanced from their raw form to form a high resolution dataset.

[1]http://maps.google.com

[2]http://www.osm.org

IV. Collect-Merge-Share

To map the state of the road network, we develop the Collect-Merge-Share (CMS) protocol that first fragments available ATA *trace histories* and fits these fragmented travel times atop of a road network map. The specifics of the travel time estimation aims require that we use bespoke components to transport and manage data (Figure 2). Services are written and executed as protocol loops and do not influence any actuators on board the vehicle (for example braking or accellerating). Travel times are transported using a modified Payload and managed using a MapStore datastore component. Additional analysis and application layers access the MapStore to visualise and graph data. Each *Message* holds a single *Payload*, each Payload multiple travel time tuples. Travel Time tuples are expressed in the form: $[A, B, ATA, TS]$.

Here A and B waypoints represent the start and end positions of a road section (a travel time fragment of vehicle mobility). ATA represents the experienced and associated fragment travel time, TS the associated sample time time-stamp. Tuples are stored and merged within the MapStore. Data is added either by vehicle sampling (monitoring) or a vehicle receiving messages from neighbouring vehicles (sharing). Within the architecture, a protocol can, on MapStore selection, construct new Messages for periodic broadcast. Layered with MapStore is the final application used to construct and visualise travel time graphs.

Protocol execution is divided into three phases following the framework as a guide: (a) collection, (b) merging and (c) sharing (Algorithm 1). Our approach uses a epidemic opportunistic networking approach [11] and WAVE Short Messages (WSM). The Collect-Merge-Share (CMS) protocol is provided a large set of parameters including a set of received input messages (MI), a clock (t), a reference to the core Mobility object (R) and a *MapStore* data structure (S) for the storage and management of travel time tuple data. During the collection phase (lines 1 to 6), a vehicle determines whether it has begun driving a new road section. To do this, the subroutine R.`newSection()` matches the present position of a vehicle to the internal map and consults any previous positions stored within the mobility object (R). R maintains both records

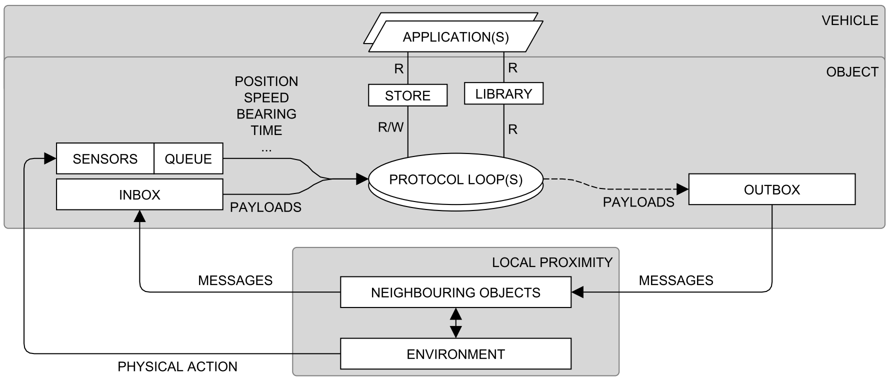


Fig. 2. Travel time estimation architecture.

Algorithm 1: Collect-Merge-Share

```
   Input: A set of received messages $MI$, a mobility
          resource $R$, a MapStore data-store $S$, a clock $t$
          and a broadcast period $bp$
   Output: A set of broadcast messages $MO$
1  begin
       // if beginning a new road section
2      if $R$.newSection() then
           // create and store travel time
           tuple
3          $A \leftarrow R$.getWaypoint(previous)
4          $B \leftarrow R$.getWaypoint(present)
5          $S$.merge([$A$,$B$, t.stopwatch, t.now])
6          t.stopwatch $\leftarrow 0$
       // merge each tuple held within the
       payload
7      for $m \in MI$ do
8          for tuple $\in m$.payload do
9              $S$.merge(tuple)
       // periodically construct new
       broadcast messages
10     if isPeriod($t$,$bp$) then
11         tuples $\leftarrow S$.selectTuples()
12         payloads $\leftarrow$ tuples.split
13         for $p \in$ payloads do
14             nm $\leftarrow$ newMessage($R$)
15             nm.payload $\leftarrow$ p
16             $MO$.append($nm$)
17     return $MO$
```

of mobility history and plans. Depending on the road topology, road sections tend to begin and end at road intersections. The vehicle calculates the distance between it and the route waypoints provided. If the vehicle has changed its road section, we construct a new travel time tuple using the previous-before-last and last route positions (A and B), the present elapsed-time stopwatch value, time-stamp and vehicle identifier. The tuple is *merged* with S and the stopwatch is reset.

As tuples are entered into the S they are compared to filter out cyclical data (redundant tuples) using a 'seen list' and insert road sections into the MapStore. We achieve this by using hashing tuples. Travel times are associated with the particular road section. The merging phase (lines 7 to 9) is concerned with the addition of tuples from messages. A secondary hashmap is used to index and match road sections (for example [A,B]) to entries in the MapStore. Each road section in the MapStore contains a linked list of (multiple) travel time samples. In those cases where the tuple already exists but does not match the hash value, a new tuple is added to the MapStore.

As a single commuting vehicle is unlikely to spend all its time driving all roads, the sharing phase allows vehicles to collect travel times about other parts of the road network using ad-hoc message passing via broadcasting (lines 10 to 14). Sharing, itself, consists of three operations: (a) tuple selection, (b) message construction and (c) message broadcast. A selection strategy is used to determine which tuples to share with neighbouring vehicles for the next subsequent cycle, with selection strategies split into *geographic* and *attribute-based* strategies. A geographic-based strategy would select tuples according to their spatial characteristics. For example, MapStore may be queried to select stored tuples which exist within a certain distance of the present position of a vehicle. In contrast, attributed selection could be made to query tuples which are recent or road sections which have at least N-many samples. For the purposes of implementation, we use a simple geographic selection of the road network, thereby providing neighbouring vehicles with travel time tuples concerning the immediate travel space.

V. Estimation and Mapping

When the CMS protocol is operating, we can assume that the estimation service residing on each vehicle is provided with a road map, the local MapStore and common libraries which contain methods to estimate travel times. As the MapStore contains road sections and associated travel time lists, the estimation of a route's estimated travel time is a combination of both *known* and *unknown* travel times. When provided a route to estimate, this is broken into a set of mobility fragments. These fragments are matched against sampled travel times (stored tuples within MapStore). If one or more sampled travel times can be associated with a fragment entry - the pair is said to be *known*. If more than a single sampled travel time fragment exists for a road section, we interpret the travel time as an arithmetic mean, the sum of sampled travel times for a particular road section divided by the number of samples. Included in the calculations are minimum and maximum travel times. Where no sampled travel times exist for a road section, we denote the road section as being *unknown*. In unknown cases, we revert to a predictive calculation - estimates are calculated as the optimal travel time to travel the particular road section (with knowledge of the speed limits).

VI. Evaluation

To test the feasibility of the CMS protocol, the protocol was developed and simulated using the Geographic Urban Simulator (GUS) framework. We highlight a subset of scenarios which include (a) a synthesised *grid*, (b) two synthesised mobility datasets constructed using OSM maps (for the cities of London and San Francisco) and (c) a subset of mobilities from the ETH trace dataset [7]. The intended service result is the production of travel time maps and graphs which represent the 'present' state of the road network. Maps can be used by various off the shelf routing algorithms to re-route a vehicle as travel time tuples are shared [6], [12]. We do not consider the outcomes of re-routing during travel and the consequence of feedback issues arising from re-routing at runtime.

A. Simulation Parameters

Simulation in our framework requires the provision of *mobility trace patterns*, *simulator parameters*, *protocol parameters* and a *prototype protocol* (decentralised service). The mobility patterns used vehicle mobilities within a 2.5 by 2.5 kilometer geographic region (Figure 1)[3]. Synthesised mobility patterns were computed using Open Street Map (OSM) and Google Maps data. Vehicle routes used either simple or extended versions of Dijkstra's algorithm for single-source shortest path traces [6], [5]. The mean Estimated Time of Arrival expected for vehicles travelling at an average speed of 8 meters per second (28.8 kilometers per hour) was specific to the mobility pattern used. Table I considers the statistical properties of grid and city mobilities used.

Protocol parameter settings were homogeneous to all vehicles within the road network. The protocol processing cycle was repeated every 100 milliseconds with broadcast windows specified for 12.5, 25 and 50 second time-outs. WAVE has a theoretical maximum communication range of 1000 meters. Previous performance work by Kai-Yun et al. [13] suggest that the efficient communication range for WSM exists within the 100 to 300 meter range, where a 3 Mbit/s throughput yielded a maximum packet loss of 9% [14], [15]. We calculated a maximum broadcast or upper limit of 30 Payloads per second for travel time data. Message loss percentages were applied to complete messages (i.e. we drop complete WSMs instead of packets, but follow previous works).

[3]We chose to use this confined region and limited area for simulation as these limits had been specified in related works.

Region	Min	Max	Mean	STD	Mean Travel Distance
Grid	25s	742s	267s	151s	2.1km
London	17s	575s	199s	100s	1.5km
San Francisco	33s	545s	166s	85s	1.38km
Zurich	18s	509s	190s	97s	1.52km

TABLE I
Input mobility patterns: each regional dataset used contained over 1000 vehicle movements. Values represent the 'ideal' mobility travel times.

Parameter	Value	Unit
Communication Range (CR)	200	meters
Maximum Broadcast Interval	12.5, 25, 50	seconds
Maximum Speed	8	meters per second
Received Message Loss (RML)	0 - 50%	messages

Simulation parameters specified a constant population of vehicles within the road network at any time. Vehicles were set a maximum speed of 8 meters per second (28.8 kilometers per hour following United States Transportation guidelines [16]). Travel times stored with a vehicle's MapStore were provided "undefined" storage capacity to reflect the likely storage capacity that future vehicles might have (in the order of several hundred gigabytes). Given a total tuple size of 44 bytes, a vehicle can potentially broadcast 11700 tuples per second.

B. Maps for Travel Time Estimates

One objective of the service is to map travel time estimates. As each MapStore is unique (due to previous contacts, events and dependencies) each vehicle map represents its own view of the state of the road network at a particular time. In comparison to a centralised service, all data is uniform between vehicles, as all vehicles use a single data source. The maps are an approximation of the true state of the total fragments available. Figure 3 visualises modelled traffic overlays.

The maps represent the known state of the road network for the previous 15 minutes. Specifically the maps shown represent the number of fragments known for a particular road section (i.e. the number of tuples and values associated with a particular road section index). Heatmaps are an accessible means by which drivers and city administrators may interpret the state of the road network [17]. The hotter a region of the road network, the more a vehicle knows about that road section. The same maps might be interpreted by city administrators to route or reorganise traffic flows using road rules, with heatmaps highlighting regions of road which form traffic bottlenecks.

C. Route Discovery

Individual vehicle stores grew as successive averaged logistic curves, of the simple form: $M(t) = 1/1 + e^{-t}$. The

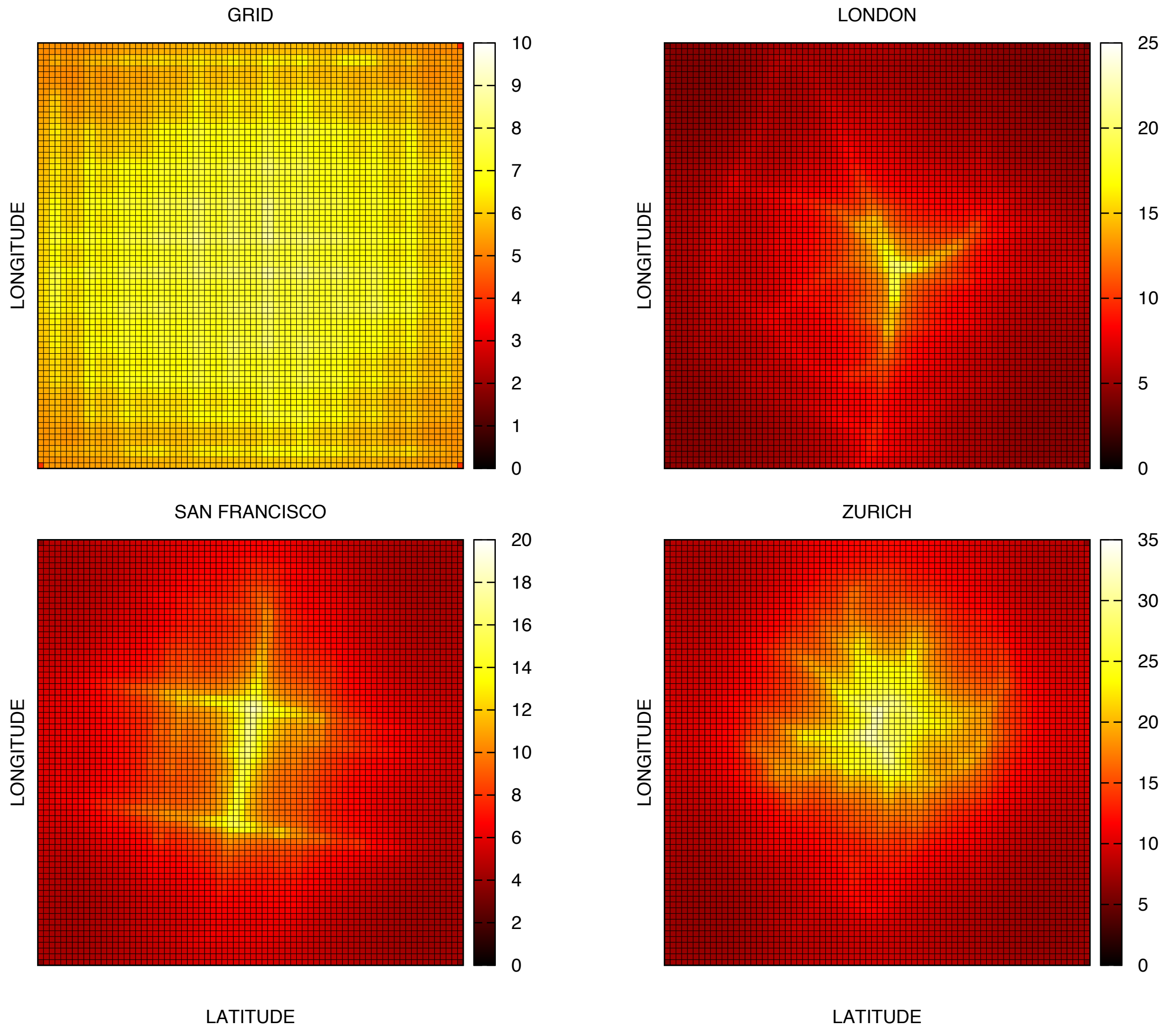


Fig. 3. Grid heatmaps showing congestion as modelled from MapStore data.

tuple size ($M(t)$) is functionally associated with time (t). The logistic curve allows us to predict the performance of the MapStore by monitoring the derivative growth of both index and unique tuple values. Given the sample road networks it took less than a mean of 837 seconds for vehicles in each road network to discover more than 80% of each road network.

Following from known and unknown travel estimation, we measure the distributions of times required to estimate travel times using only ATAs after 1800 seconds of service operation. In other words, how long does it take for us to make estimations using only known travel times? The Route Discovery Time (RDT) represents the elapsed time taken to completely estimate a route from *known* sampled travel times held within the MapStore. RDT is highly dependent on the road network, a vehicle's route, the number of contacts occurring between a vehicle and its neighbours and the speed of a vehicle. Road network topology influenced RDT as vehicles were bottlenecked to follow specific routes. The grid structure presents the most divergent road network and as such RDT performance was worst in the network. The mean RDT requiring 216.4 seconds with a standard deviation (SD) of 121.8s. The cities of London and Zurich had similar RDT performance. Mean RDT requiring 139.8 seconds (SD = 92.7 seconds) in London and 189.4 seconds (SD = 96.4 seconds) in Zurich. San Francisco performed best with a mean RDT of 78.4 seconds (SD = 71.4s).

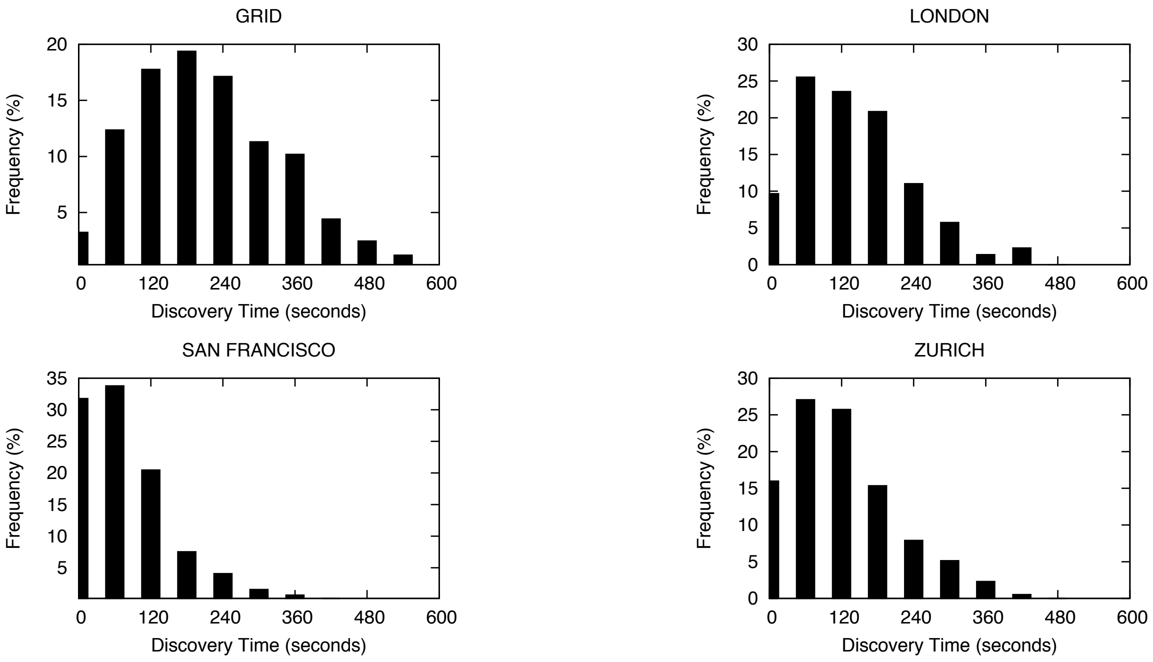


Fig. 4. Route Discovery Time (RDT) distributions for set of vehicles sampled after 1800 seconds of simulation (POP=250, CR=200m, BP=12.5s and RML=0%).

New vehicles entering a road network were required to sample the road network. Hence the time taken to perform route discovery was longer, because fewer samples existed in the early operation of the service. As MapStore grows to its maximum achievable level, the subsequent RDT is reduced. Hence, in an unexplored road network RDT is high, while in an explored network RDT is short (less than the maximum previous RDT), however data may be stale. This short RDT is shown by the resultant frequency of RDT expressed as a percentage distribution of discovery times (Figure 4). For example, for the city of London, 9% of travel time estimations could be based on ATA data, as vehicles entering the road network are instantly provided at time 0. Another 25% of vehicles found estimations relevant to them within 60 seconds. In comparison, more than 60% of vehicles within the San Francisco region discovered relevant ATAs relevant to their journeys within 60 seconds. For each mobility, each differing road network, the outcome was similar. As time progressed the normal distribution of RDT shifted from right to left (i.e. closer to shorter discovery times).

D. Message Counts

Table II shows the typical mean broadcast message counts per minute for increasing vehicle populations versus increasing broadcast period (BP) for the various city scenarios. Vehicle populations (POP) increased between 250 and 1000 vehicles per 2.5 x 2.5 kilometer area. Notably, fewer messages are broadcast than the ideal, as some vehicle mobilities are shorter than others, where mobility patterns are influenced by the city road network.

Table III shows the ratio of received messages to broadcasts for varying vehicle populations (POP) and varying BP. The value represents the redundancy or number of copies made, per broadcast. For example, for the city of London, given a 12.5 second BP, payload data was copied a mean of 14 times for each broadcast made given a population of 250 vehicles. Varying the BP did not significantly effect the receive-broadcast ratio (R:B). However, for each city road network a BP of 25 seconds was seen to raise the R:B factor slightly. For example varying BP for Zurich yielded a 4.4% increase in retrieval for a vehicular population of 250 vehicles. Notably we see the reverse effect in larger vehicle populations. Improving dissemination reduces the staleness of travel time data.

Map	BP	Number of Vehicles (POP)			
		250	500	750	1000
Grid	12.5s	1173	2320	3540	4732
	25s	477	1121	1702	2282
	50s	260	500	842	1174
London	12.5s	1159	2378	3567	4782
	25s	554	1150	1761	2350
	50s	276	557	865	1158
San Francisco	12.5s	1176	2351	3245	4670
	25s	582	1166	1751	2372
	50s	286	568	852	1179
Zurich	12.5s	1152	2378	3565	4754
	25s	567	1152	1757	2383
	50s	253	567	881	1170

TABLE II
MEAN BROADCAST MESSAGE COUNTS (PER MINUTE) FOR INCREASING VEHICLE POPULATIONS (POP) VERSUS INCREASING BROADCAST PERIOD (BP).

Map	BP	Number of Vehicles (POP)			
		250	500	750	1000
Grid	12.5s	4.69	9.45	14.32	19.51
	25s	4.79	9.41	14.13	19.40
	50s	4.79	9.63	14.23	19.21
London	12.5s	14.38	28.61	42.45	62.32
	25s	13.61	28.5	42.61	61.73
	50s	13.29	27.94	44.12	56.55
San Francisco	12.5s	14.1	27.05	42.04	54.50
	25s	14.3	28.47	41.47	56.30
	50s	13.83	28.13	42.99	55.83
Zurich	12.5s	15.11	28.73	46.34	61.84
	25s	15.8	29.36	45.56	61.21
	50s	15.12	30.51	46.87	61.62

TABLE III
MEAN RATIOS OF RECEIVED MESSAGES TO BROADCASTS (R:B), GIVEN TABLE II, FOR INCREASING VEHICLE POPULATIONS (POP) VERSUS INCREASING BROADCAST PERIOD (BP) FOR VARYING ROAD MAPS. EACH VALUE REPRESENTS THE MEAN REDUNDANCY FOR EACH MESSAGE BROADCAST.

E. Message Failure and Redundancy

To determine the effect of dropped messages on the service, we simulated received link failures as received message losses (RML) of between 0% and 50% of messages retrieved (R). Experiments considered the effect of message loss on the growth of the sizes of MapStore index (mI) and unique tuple counts (mU), namely ($|\text{mI}| \leq |\text{mU}|$). MapStore index is affected only slightly by increased RML. However we see that the number of unique travel times stored inside MapStore declines significantly beyond at 25% RML. A smaller mU considers that while MapStore may map a regional road map, the richness of data is limited. For example, for the city of Zurich, mU declines from 2254.69 fragments for 0% message loss to 1748.58 fragments for 50% loss.

VII. RELATED WORK

Services like Google Maps [18] handle many thousands of queries each day. Many vehicles are often provided the same route where alternative routes may be shorter. Navigational computers typically consider very few routes when providing directions. Hence, there is the opportunity to improve navigation such that navigation computers might help improve the flow of vehicles along road networks. The service discussed in this paper draws on work from mobility trace collection, trace analysis, intelligent transport systems (ITS), opportunistic networking, simulation and navigational guidance systems. Notably, the Geographic Urban Simulator (GUS) allows us to develop protocols and services as they would be deployed on a real device, for example an Android device. A number of previous works have successfully collected and analysed trace data about city transportation systems, notably Adhoc City [**?**] (usage of static and dynamic mobile devices), Google Latitude [**?**], Mobile Millenium [3] and urban-based works from the MIT Senseable City Lab [17][**?**]. WikiCity [17] propose a community of citizens for the collection and sharing of data using mobile phones. Waze [2] is a public crowdsourcing traffic repository built on reported traffic data. The Mobile Millenium project is thus far one of the most successful automated approaches in monitoring vehicle mobilities through a city [3], in contrast we use a purely decentralised method of collecting and sharing data. The data is shared with higher layered centralised authorities for service provision. The MIT Syn(c)ity [19] (and Affective Intelligent Driving Agent) proposals seek to provide a centralised computer guide to a driver, recommending routing throughout a city based on driver statistics, preferences, their social network, the state of the vehicle and known traffic conditions.

VIII. CONCLUSIONS

In this paper we investigated the feasibility and evaluated base performance of an ad-hoc and decentralised travel time estimation service which sought to use experienced travel time data to model road network state, thereby reducing error in travel time estimations. Mobility histories are fragmented and shared with nearby vehicles to approximate a travel time estimation service which would normally be constructed using centralised approaches. Results show that such a decentralised service can approximate the operations of centralised service. A disadvantage of decentralisation is data availability and staleness.

A large number opportunities exist in future work. The inclusion of a hybridised approach, mixing beneficial features such as static data repositories and the usage of mobile phone networks (where available) is likely to reduce the problem of data staleness. Further methods of isolating significant changes in the road network state may exist and there is room for performance improvement in the optimisation of data collection and analysis techniques. Moreover, travel time data is just one example of city data which can be mapped. Various other stakeholders within the city may be interested in other sensory data, such as weather, population density, pollution and road surface information. Such an approach may also have application in scenarios of limited connectivity, for instance the monitoring and provision of travel time data on underground train systems. We have not considered the security, privacy and incentive requirements and implications of the system.

ACKNOWLEDGMENT

This research was undertaken as part of the Adaptable Environments for Distributed Ubiquitous Systems (AEDUS2) project. UK EPSRC research grant EP/E025188/1. We thank the Policy group for their feedback.

REFERENCES

[1] R. Krishnan, "Travel time estimation and forecasting on urban roads," Ph.D. dissertation, Imperial College London, 2008. [Online]. Available: http://www.cts.cv.ic.ac.uk/documents/theses/KrishnanPhD.pdf

[2] Waze, "Real-time maps and traffic information based on the wisdom of the crowd," http://www.waze.com, 2011, [Online; accessed October-2011].

[3] D. B. Work, S. Blandin, O.-P. Tossavainen, B. Piccoli, and A. M. Bayen, "A traffic model for velocity data assimilation," *Applied Mathematics Research eXpress*, vol. 2010, no. 1, pp. 1–35, 2010. [Online]. Available: http://amrx.oxfordjournals.org/content/2010/1/1.abstract

[4] J. Burguillo-Rial, E. Costa-Montenegro, F. Gil-Castineira, and P. Rodriguez-Hernandez, "Performance analysis of ieee 802.11p in urban environments using a multi-agent model," in *Personal, Indoor and Mobile Radio Communications, 2008. PIMRC 2008. IEEE 19th International Symposium on*, sept. 2008, pp. 1 –6.

[5] D. Delling, P. Sanders, D. Schultes, and D. Wagner, "Engineering route planning algorithms," in *Algorithmics of Large and Complex Networks*. Springer, 2009.

[6] E. W. Dijkstra, "A note on two problems in connexion with graphs," *Numerische Mathematik*, vol. 1, no. 1, pp. 269–271, December 1959. [Online]. Available: http://dx.doi.org/10.1007/BF01386390

[7] V. Naumov, R. Baumann, and T. Gross, "An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces," in *MobiHoc '06: Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*. New York, NY, USA: ACM, 2006, pp. 108–119.

[8] ——, "An evaluation of inter-vehicle ad hoc networks based on realistic vehicular traces," in *Proceedings of the 7th ACM international symposium on Mobile ad hoc networking and computing*, ser. MobiHoc '06. New York, NY, USA: ACM, 2006, pp. 108–119. [Online]. Available: http://doi.acm.org/10.1145/1132905.1132918

[9] T. Das, P. Mohan, V. N. Padmanabhan, R. Ramjee, and A. Sharma, "Prism: platform for remote sensing using smartphones," in *Proceedings of the 8th international conference on Mobile systems, applications, and services*, ser. MobiSys '10. New York, NY, USA: ACM, 2010, pp. 63–76. [Online]. Available: http://doi.acm.org/10.1145/1814433.1814442

[10] Q. Yang, A. Lim, S. Li, J. Fang, and P. Agrawal, "Acar: Adaptive connectivity aware routing for vehicular ad hoc networks in city scenarios," *Mob. Netw. Appl.*, vol. 15, pp. 36–60, February 2010. [Online]. Available: http://dx.doi.org/10.1007/s11036-009-0169-2

[11] A. Vahdat and D. Becker, "Epidemic routing for partially connected ad hoc networks," 2000. [Online]. Available: http://citeseerx.ist.psu.edu/viewdoc/summary?doi=10.1.1.34.6151

[12] P. Sanders, D. Schultes, and C. Vetter, "Mobile route planning," in *Proceedings of the 16th Annual European symposium on Algorithms*, ser. ESA '08. Berlin, Heidelberg: Springer-Verlag, 2008, pp. 732–743. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-87744-8_61

[13] K.-Y. Ho, P.-C. Kang, C.-H. Hsu, and C.-H. Lin, "Implementation of wave/dsrc devices for vehicular communications," in *Computer Communication Control and Automation (3CA), 2010 International Symposium on*, vol. 2, may 2010, pp. 522 –525.

[14] IEEE, "IEEE Standard for Wireless Access in Vehicular Environments (WAVE) - Networking Services," *IEEE Std 1609.3-2010 (Revision of IEEE Std 1609.3-2007)*, pp. 1 –144, 30 2010.

[15] S. Eichler, "Performance evaluation of the ieee 802.11p wave communication standard," in *Vehicular Technology Conference, 2007. VTC-2007 Fall. 2007 IEEE 66th*, oct 2007, pp. 2199 –2203.

[16] Massachusetts Department of Transportation, "Design build procurement guide," http://www.mhd.state.ma.us/downloads/designGuide/CH_6_a.pdf, Jan. 2006.

[17] F. Calabrese, C. Ratti, and K. Kloeckl, "Wikicity: Real-time location-sensitive tools for the city," *Handbook of Research on Urban Informatics: The practice and Promise of the Real-Time City*, 2008.

[18] Google, "Google maps," http://maps.google.com, Oct. 2011.

[19] M. Martino, K. Kloeckl, G. Di Lorenzo, J. Dunnam, E. Kang, and C. Ratti, "syn(c)ity: Vizualising the potential of a predictive in-car recommendation system," http://senseable.mit.edu/papers/pdf/2010_Martino_et_al_Syn(c)ity_Internet_of_things.pdf, Dec. 2011.