EasyEV: Monitoring and Querying System for Electric Vehicle Fleets Using Smart Car Data

Gregor Jossé, Matthias Schubert, Ludwig Zellner

Institute for Informatics, Ludwig-Maximilians-University Munich, Oettingenstr. 67, 80538 Munich, Germany, {josse,schubert,zellner}@dbs.ifi.lmu.de

Abstract. Electric vehicles (EVs) have great potential as a modern mobility concept. Electricity already relies on a broad infracstructure and is available anywhere in developed countries. Furthermore, EVs are emmission-free which makes them the preferable form of individual transportation in urban areas where air pollution is often alarmingly high. However, operating EVs has several drawbacks compared to common combustion engine cars. The range of most EVs is rarely above 150 kilometers, and when running out of energy, recharging an EV usually takes up to several hours. In order to benefit from the advantages of EVs without being afflicted with the disadvantages, it is advisable to rely on the support from smart systems for trip and charge planning. In the project Shared E-Fleet, the shared use of a fleet of electric cars by a heterogeneous group of drivers is examined. In the presented demo, we introduce a spatio-temporal query system which was developed to support drivers and fleet managers alike. For the driver, the system provides assistence to keep in range of charging stations and provides routing alternatives to a specified destination. For the fleet manager, the system incorporates real-time information to identify possible delays or battery drainages and thereby detect deviations from the fleet schedule to allow for early rescheduling.

1 Introduction

With increasing air pollution, limited fossil fuel supply, and growing pressure to politically enforce reduction of CO_2 emissions, it seems, the days of common combustion engine cars are numbered. The number of electric vehicles (EVs) is constantly growing, especially in urban areas where air pollution is an increasing health threat. However, EVs have two major drawbacks which make them significantly less flexible than combustion engine cars. The first limitation is the range of EVs, which varies mostly from 100 to 150 kilometers, excluding some exceptions like the Tesla Roadster which – according to the manufacturer – travels up to 400 kilometers per charge. In addition, the energy consumption and therefore the range of an EV is dependent on the driving style but also on factors like heating, air conditioning, soundsystem, and headlights. The second limitation are the rather long recharging times. In contrast to refueling a car, a full recharge can take several hours.

Despite these drawbacks, EVs have great potential to fulfill the mobility needs of urban populations. This is mainly because the average trip in a city is less than 30 kilometers long and cars are actually parked most of the time. These two aspects can be taken advantage of. Even so, when in a setting where EVs are shared among several people, for instance by the employees of companies residing in the same office building.

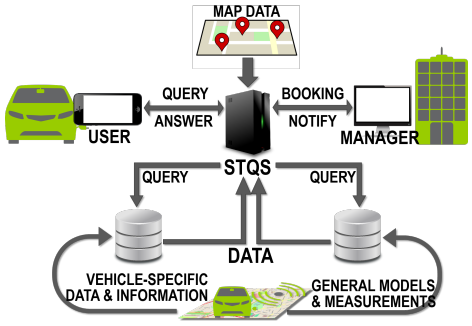


Fig. 1. System architecture of EasyEV. The manager is informed about anomalies concerning his fleet. The user can query the system, his on-board unit provides information to the system.

The business model of a shared fleet of EVs is examined in the project Shared E-Fleet. One goal of this project is to examine how spatio-temporal information systems can counteract the drawbacks and how the use of real-time information can improve the efficiency of a fleet of EVs. For instance, by monitoring the vehicles and keeping track of their remaining range limits, an automated booking system can inform the subsequent user of an expected delay or – even better – assign the user a different vehicle. Conversely, if the booking schedule is kept stable, charging times of vehicles can be optimized, yielding low electricity costs and reliably plannable charging processes.

In this demonstration we present EasyEV, our prototype for supporting drivers and fleet managers alike with spatio-temporal information services. EasyEV is currently employed in pilot projects in three different German cities involving seven BMW i3. Each car is equipped with an on-board transmission unit providing data like position, remaining range, charging status, and active electric devices. By evaluating this data, we are able to detect and predict anomalies, in order to inform the fleet manager about critical situations like expected belated returns or expected range exceedances. In addition to these so-called monitoring features, EasyEV also supports the driver with multiple functionalities, such as providing directions to the nearest charging station, visually informing about reachable destinations, and computing alternative paths.

2 System Architecture

EasyEV runs as a platform-independent system which is connected to the fleet management and to the driver via app as well as to a real-time car database and an aggregated sensor information database. While the driver queries the system directly, the fleet management registers its vehicles and bookings with the system and is informed if anomalies occur. All connections are realized as platform-independent web service interfaces. Thus, the features of EasyEV are easily reused in a system requiring similar functionalities. At the heart of EasyEV lies a spatio-temporal query system (STQS) which answers directly to queries from users (routing features) and indirectly executes all recurrent tasks (monitoring features). For the STQS to answer queries instantly, the road network index has to be kept in memory. Hence, the STQS is modeled as a standalone server

entity, while all communication with the server is handled by the aforementioned web services. The architecture is illustrated in Figure 1.

The STQS relies on OpenStreetMap[1] (OSM) data for all road networks which we model as multicriterion (multiattribute) graphs. In a multicriterion graph, the vertices correspond to crossings, dead ends, etc., and the edges represent directed road segments connecting the vertices. A cost function maps every edge onto its cost vector. Each component of the cost vector corresponds to a particular cost dimension of the respective edge, e.g., distance, travel time, and energy consumption.

For the pilot projects, specific on-board units have been developed which collect data directly from the car computer as well as from additional built-in sensors, most prominently GPS and temperature sensors. The vehicle-specific data is fed into a database (see Figure 1), while the environment-related data is aggregated and used to train time-dependent models reflecting outside influences on city-scale traffic, such as congestion or icy roads. We rely on external service providers feeding this kind of information into a data storage from where the STQS can query current influence factors. In order to be independent of the number of pilot test vehicles used during the demo presentation, we also simulate driving behavior according to the recorded trajectories. This enables us to replay actual historic driving data as well as display real-time movement enriched with simulated trajectories and bookings, ensuring a high frequency of queries and tasks.

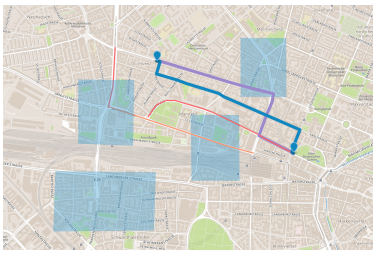
In addition to the STQS, EasyEV offers a GUI to visualize trajectories, traffic-influencing factors, and query results. OSM data, is easily displayed using the open source library Leaflet[1] combined with the map design tool Mapbox[1]. The GUI is browser-based and thereby platform-independent. It is dependent on the STQS-specific interfaces but can run on an entirely different machine. Hence, the architecture of EasyEV follows the model-view-controller principle, which facilitates portability and reusability.

3 Demo Features

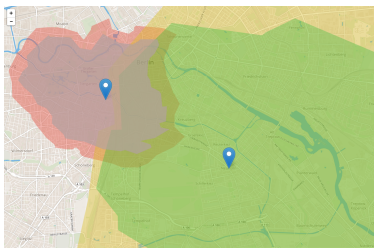
We distinguish between routing features and monitoring features. The former are especially interesting from a user perspective, while the latter are particularly interesting from a fleet manager (operator) perspective. Note that in the pilot projects, monitoring and routing features are separated, as users do not have access to the trajectories of other users, while the monitoring system does not need access to all routing features.

Let us first present the routing features. As an elementary functionality, EasyEV supports shortest path searches given start and target nodes as well as w.r.t. predefined combinations of cost critera. This has proven useful, as some users prefer a single path over a set of alternatives. However, EasyEV supports state-of-the-art algorithms for the computation of two sets of alternative paths, which we present without going into detail. The first set is known as the route or path skyline or the set of pareto optimal paths [3, 1]. More recently, another concept of optimality was introduced in [5], called the linear path skyline. While the conventional path consists of all paths with cost vectors optimal under some monotone cost function, the result set of the linear path skyline consists of all paths with cost vectors optimal under a linear cost function. This is a restriction, limiting the number of results returned to the user and allowing for iterative result generation.

[1] www.openstreetmap.org, www.leafletjs.com, www.mapbox.com



(a) Paths respecting (blue path) and ignoring (purple path) sensor-detected traffic delay (red road segments) and road ice (rectangular blue opaque regions).



(b) Respective range limits of two vehicles, illustrated as isochrone regions. The inner region is a more conservative range estimation than the outer region.

Fig. 2. Visualizations as displayed by the framework.

EasyEV supports efficient methods for the computation of both sets, following [6, 5]. In all the above cases, a result list is displayed which the user can browse.

As mentioned before, EasyEV does not only rely on static edge costs, constituted by the underlying road network, but also incorporates sensor information, as derived from data acquired by mobile (e.g., the vehicles themselves) and stationary sensors (e.g. traffic loop sensors). In our demo we focus on the influence factors road ice and traffic delay, however noting that, given a wider array of data providers, other factors could easily be included. These factors either affect specific roads (traffic delay) or whole areas (road ice). Both variants are displayed on the map. Given this kind of data, EasyEV computes paths avoiding affected roads or regions, depending on the duration of the delay or on the severity of the road icing, as depicted in Figure 2(a).

In addition to sensor data, EasyEV employs static meta-information. Incorporating locations of public charging stations, EasyEV allows for ad hoc queries in order to intercharge when on the road, e.g., during a customer meeting. For the demo, this information is retrieved from providers like Open Charge Map or Plugshare[2]. Relying on probabilistic algorithms, similar to those presented in [2], EasyEV is able to provide paths along a number of charging stations, in order to maximize the probability that one of these stations is currently vacant.

Now, let us turn to the monitoring functionalities which serve the purpose of real-time as well as retrospective fleet analysis. In addition to the recorded trajectories, we also simulate bookings and corresponding trajectories in order to ensure high data density (during the demonstration). Trajectory simulation is done by computing (some pareto-optimal) path and sending positions along this path according to some randomly drawn travel time from a distribution around the speed limit. All trajectories, real-time, historic, and simulated, are retrieved from the vehicle database via web service. Most pilot test vehicles send their data every minute, some vehicles send their data every ten seconds. In order to clean the GPS signals of measurement errors and to be able to draw a valid trajectory between any two data points, we apply a map matching algorithm similar

[2] www.openchargemap.org, www.plugshare.com

to [4]. Hence, we can display the roaming cars either in real-time or for defined past timespans at variable speeds. In both cases, we can display notifications having been or being sent by our monitoring system.

There are four types of notifications distributed to the fleet manager by the STQS. These notifications stem from the use cases covered in the project and will be displayed as pop-ups during the demo. The first one informs the manager of *illegal movement*, meaning that a registered veicle is moving without a valid booking. If a vehicle is not moved throughout the whole period of a booking, the manager receives a *no show* notification, as this might result in different billing. The other two notifications only occur during driving (with a valid booking). First, if a return within the booking period is unlikely, the fleet manager is informed of an expected *delay*. This is the case, if even the fastest path from the vehicle's current location to the booking-specific return station is longer than the remaining booking period. Second, if even the shortest path from the current location to the return station is longer than the currently remaining range of the vehicle, the fleet manager (and the driver) is informed of an expected *malfunction*.

In addition to the geo-positions at the respective timestamps, various other information is collected by the on-board unit and fed into the database. For example, the total number of kilometers driven, whether the passenger seat is occupied, whether wiper or lights are on, the current energy consumption, and the remaining distance. EasyEV displays all this information when hovering over the geo-position of an illustrated vehicle. Of all the data collected, the remaining distance is of particular interest, seeing as range limit exceedance results in vehicle deficiancy and often significant rescheduling efforts. Hence, we incorporate the feature of displaying isochrone diagrams around pilot test EVs, as displayed in Figure 2(b), where the inner isochrones are a more conservative estimation, and the outer isochrones depict the ideally reachable area.

4 Conclusions

In this demo, we present EasyEV, a querying and monitoring system for fleets of EVs, which is being used in a pilot project investigating the potential of smart systems employed in optimization of EV fleets. EasyEV offers features for fleet managers and drivers alike. All computations are handled by a spatio-temporal query system, while the GUI is browser-based and the communication is handled by web services. Hence, EasyEV ensures high functionality and at the same time great flexibility and reusability.

References

1. M. Ehrgott and X. Gandibleux. A survey and annotated bibliography of multiobjective combinatorial optimization. *OR-Spektrum*, 22:425–460, 2000.
2. G. Jossé, K. A. Schmid, and M. Schubert. Probabilistic resource route queries with reappearance. In *EDBT 15*, pages 445–456, 2015.
3. H. P. Kriegel, M. Renz, and M. Schubert. Route skyline queries: A multi-preference path planning approach. In *ICDE 10*, pages 261–272, 2010.
4. P. Newson and J. Krumm. Hidden markov map matching through noise and sparseness. In *ACM SIGSPATIAL GIS 09*, pages 336–343, 2009.
5. M. Shekelyan, G. Jossé, and M. Schubert. Linear path skylines in multicriteria networks. In *ICDE 15*, pages 459–470, 2015.
6. M. Shekelyan, G. Jossé, and M. Schubert. Paretoprep: Efficient lower bounds for path skylines and fast path computation. In *SSTD 15*, 2015.