Geographic Information Systems
Winter term 2020/2021

Daniel Muffler[1]
Emilio Silvestri[2]
Junran Yang[3]

Advanced Route Planning of Electric Vehicles based on Charging Stations

Abstract

As electric vehicles become popular, route planning with consideration of charging becomes important for drivers due to the short drive range of electric vehicles. We built a route planning tool for Germany to allow users to participate in route planning, choosing charging stations by users' preference of surrounding amenities. We introduced a heat map to assist the user during route planning to indicate the station distribution and various interactions to ease the planning process.

1 Introduction

Electric Vehicles (EVs) are more and more part of our daily life. Despite their unarguable benefits, electric vehicles do not come without drawbacks. One of the biggest issues that electric cars owners have to face is related to the charging process, mainly concerning *where* and *for how long*. Thanks to government and local's incentives, it is luckily no longer hard to find a place where to charge an electric car, especially in big cities and along main roads. According to data from 2021, Germany, for example, counts already more than 20K charging stations covering the whole country [4]. With this problem (partially) solved, the only open question remains the charging time. A complete charge can last up to 8 hours and, even if fast charge is available, this can become problematic when planning long trips that include a longer refill break.

With our application, we want to address the issue of route planning considering charging halfway, which electric car drivers could encounter. In particular:

1. Finding the shortest route to the desired destination;
2. Showing the availability of charging stations along the route;
3. Interactively exploring the available charging stations in an area, prioritizing the ones with interesting surrounding amenities;
4. Showing the complete route, including charging stops, along with an estimated arrival time.

The code is available at this Git repository along with installation instructions and links to the necessary data.

2 Dataset

This section describes the datasets used, how they were retrieved and processed before being added to our project.
For the purpose of our project, data were provided by three datasets:

- Charging Stations in Germany from ESRI Deutschland;
- Restaurant and food amenities in Germany by OpenStreetMap;
- Rated special restaurants by the Michelin Guide website.

[1]daniel.muffler@uni-konstanz.de
[2]emilio.silvestri@uni-konstanz.de
[3]junran.yang@uni-konstanz.de

[4]`https://de.statista.com/statistik/daten/studie/460234/umfrage/ladestationen-fuer-elektroautos-in-deutschland-monatlich/`

The Charging Stations dataset did not require any preprocessing. The CSV file downloaded by the source provides information about more than 12,000 electric charging stations in Germany as of February 2020. In addition to the location coordinates, it contains information about the operator, the address, commissioning, the connected load in kilowatts, the charging device, the number of charging points, and the connector types with information on the kilowatt output.

The OpenStreetMap dataset contains detailed information about geographic elements of interest (roads, railways, houses, shops, forests, lakes). Our focus was on amenities across Germany. Therefore we extracted a subset of the data to be imported into our system. In particular, for restaurant-related amenities (e.g., bar, pub, Biergarten, fast food), we retrieved the name and the geographic information. For the entries whose geography was a polygon, we only retained the middle point of the polygon to ensure consistency.

In order to provide "special" results, we decided to include information from the Michelin guide. The dataset scraped[5] from the Michelin website reports, along with the geographic location of 1500 top-rated restaurants, a link to the webpage, a picture link, the type of cuisine offered, and, for some restaurants, a 1-5 score.

3 Methodology

This section explains the usage of the data and the necessary preprocessing. Furthermore, visualization techniques used in our system are mentioned as well.

3.1 Data Preprocessing

One of the core points of our system is the integration of data, mainly in a spatial fashion. This operation is performed, in particular, when providing the charging stations that are located in a specific area (Esri database) and subsequently showing the amenities reachable within walking distance, from both the OpenStreetMap data and the Michelin Guide database. We opted for a simple union of elements for the last two databases' amenities, resulting in some possible (however sporadic) duplicates, but it would not affect the final visualization.

3.2 Algorithms

The amenities around a charging station are instead retrieved using the spatial function *ST_DWITHIN* provided by the PostGIS extension. The use of this function within the database query allowed us to quickly select the amenities located at a specific air distance from the station. This can provide an upper bound approximation of the walking distance, keeping the computational time low.

The charging stations for a specific area are instead retrieved by exploiting the isochrones provided by the integrated OpenRouteService, together with the *ST_CONTAINS* spatial query function PostGIS. In particular, the isochrone identifies an area of reachable points within a certain **driving distance** from a starting point. In our case, the starting point would be manually selected on the map by the user. The use of the driving distance allows a better precision than the air distance mentioned before. However, this leads to higher computational complexity since PostGIS does not directly provide this function.

According to the OpenRouteService documentation, the isochrone calculation is based on an adapted version of the algorithm described in "Fast Computation of Isochrones in Road Networks" by Valentin Buchhold using a partitioning fashion.

3.3 Visualizations

Other worth-to-mention methodologies regard the visualizations. In particular, the plotted charging stations within a certain isochrone are grouped using density clustering. This technique allows us to have a neater high-level visualization for more expansive areas (Figure 1) and to show the actual data points when zooming in a smaller portion of the map (Figure 2).

[5]Link to the scraper repository: https://github.com/danmuf/michelin-stars-restaurants-api

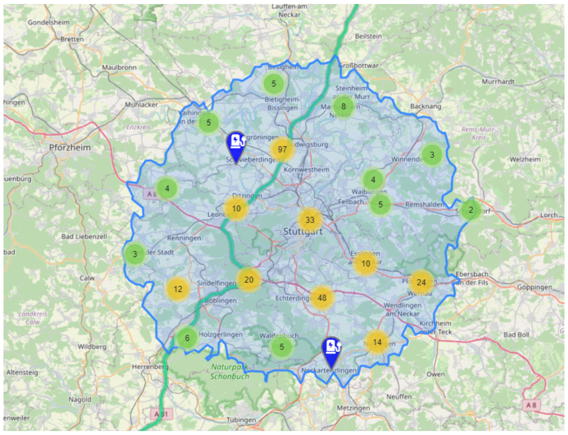


Figure 1: Density Cluster

One last technique provided by our system is the heat map visualization (Figure 3). Completely frontend-coded, it shows the charging station availability over the whole national territory. It is worth mentioning that this feature also offers some interactivity: the user can change two parameters, *radius* and *zoom level* in order to perform a multi aggregation level exploration. The heat map can be kept visible during route planning, giving visual help to the user when choosing travel areas.

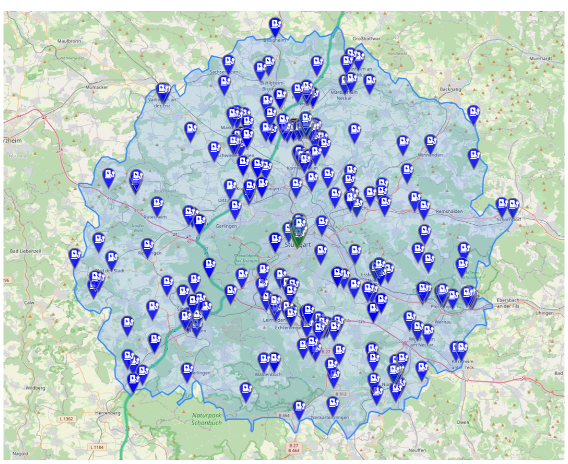


Figure 2: Single Point Charging Stations

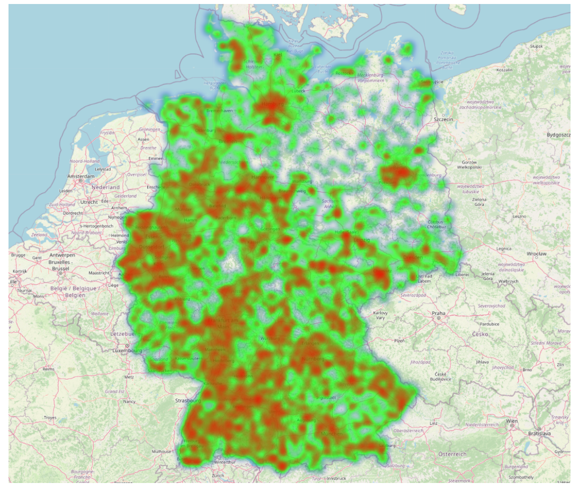


Figure 3: Heat Map

4 System

This section describes and introduces the main components of our application including a short target user description.

4.1 Overall Structure and User

The system architecture mirrors the one proposed during the exercises. A Docker Container holds:

1. A PostgreSQL instance with the PostGIS extension installed

2. Python backend based on the Flask framework

3. Frontend based on Angular

In addition to that, a new component running OpenRouteService has been added to the architecture in order to provide the routing and isochrones calculation locally and avoid the limitations of their remote API [6].

The target user of the application is an EV driver. The main tasks that he or she can perform include:

[6] OpenRouteService API restrictions https://openrouteservice.org/restrictions/

1. Route calculation, including estimated time of arrival and along the route;

2. Show reachable charging stations from a selected location;

3. Explore amenities around the shown charging stations;

4. Complete the route selecting the desired stations;

5. Show global charging station distribution using the heat map.

During the interaction with the application, the user has the possibility to tune some specific parameters.

4.2 Frontend

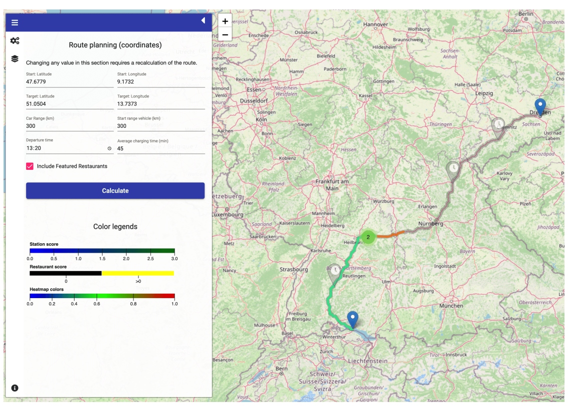


Figure 4: Frontend Interface: Sidebar and Map

The frontend interface is built based on Angular. The interface contains two major components: sidebar and map (Figure 4).

The sidebar provides complete control of the tool's function and customization of the visualization.

In the route section (5(a)) of the sidebar, the user can specify the latitudes and longitudes of departure and destination, which are used to compute the route in between. Below the location input, the maximum range of the electric vehicle and the battery life on departure are located, which are used to calculate the danger segment of the route where the battery might run out of power. Besides, the departure time and average charging time can also be set by the user to show the correct time along in the route.

The settings section (5(b)) offers further customization: (1) Isochrone Max Range specifies the maximum station search range when the user clicks at one location, to avoid unnecessary computation; (2) Amenity Distance limits the restaurant search range of one station and indicates how far the user would like to "walk" while the car is being charged; (3) The amount for fast charge defines the power gained with a fast charge depending on the vehicle model.

The heat map section (5(c)) in the sidebar allows the user to toggle the heat map of the charging stations and to tune its parameters.

4.2.1 Map

The map shows the computed route and assistant visualization.

In a route (Figure 4), the reachable part of the route is denoted in green color, the danger segment where the battery may die is in orange, and the whole route in gray. When a danger segment exists, an isochrone, indicating the reachable area from a specific location, will be shown when the cursor hovers at a location for 0.5 seconds. Along with the hover isochrone, a new route to the hover location would give the user a hint of how the altered route would be if the location is selected.

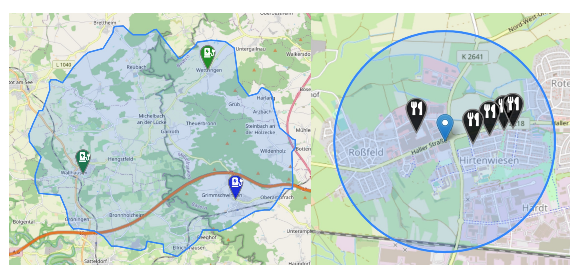
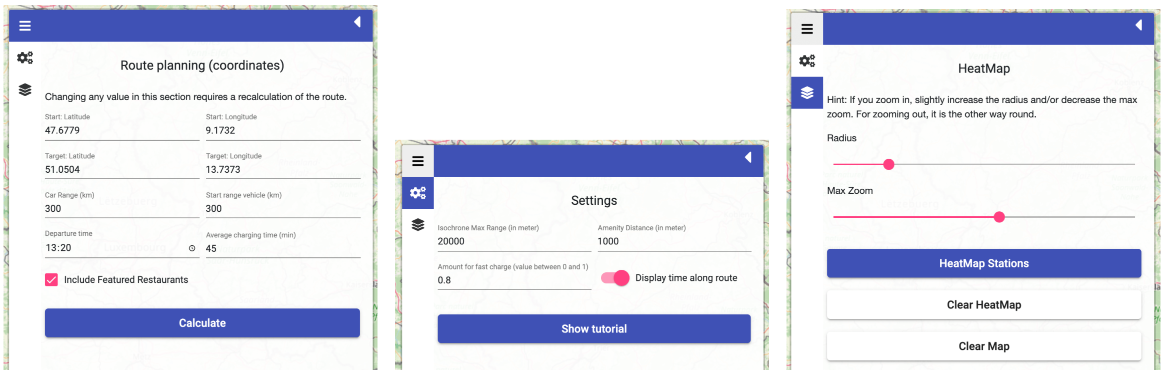


Figure 6: Stations and Restaurants

Once the user clicks at one location, the map will show the stations within the corresponding isochrone.



(a) Route Section (b) Settings Section (c) Heat Map Section

Figure 5: Sidebar's Three Sections

If the user further clicks in the menu of one station, he or she can choose to show the restaurants, and then select this station or return to see all stations. When selecting a station, the user can choose whether a full charge or a fast charge will be performed.

The user can turn on the heat map from the sidebar. The heat map visualizes the distribution of the charging stations to help the user to choose a stop location. Another benefit of using the heat map is that it does not collide with the station visualization, in which stations are presented as pins, but still works as a compatible underlying layer.

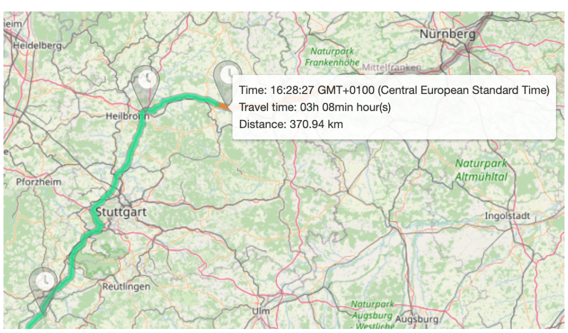


Figure 7: Timing Along the Route

Besides, we introduced visualization of timing along the route (Figure 7). It is helpful for the user to know the time at a specific location, and to decide which amenities are more important in that time of the day (e.g. restaurants for lunch, pubs for dinner and so on).

4.3 Backend

The backend is built on Flask with Python. It takes the responsibility to communicate with OpenRouteService and database for route planning, isochrones computing, station query, and restaurants query.

4.4 Database

We used PostgreSQL with PostGIS extension as our database, which contains restaurant data extracted from OpenStreetMap and Michelin Guide, as well as Charging Stations data.

4.5 OpenRouteService

The self-deployed instance of OpenRouteService provides the major routing service.

The application mainly takes advantage of two APIs provided by the OpenRouteService instance: (1) Route planning for multiple waypoints; (2) Isochrones in terms of distance or time. [7]

[7] https://openrouteservice.org/dev/#/api-docs

5 Use Cases

5.1 User-Participated Routing

The main feature of the tool is to allow the user to participate in route planning by his preference.

When the user needs a route, he or she can input the latitudes and longitudes of the departure and destination, and the route will be computed and showed on the map.

If the destination is not reachable given the battery life on departure and the maximum range of the electric vehicle, the route shows a danger segment denoted in orange color, which gives the user an idea of where the battery may die.

The user can select a point in the map to search for the stations around the map, which are scored by their respective surrounding amenities. The user can select a station immediately or after checking the restaurants near the station, and the station will be added to compute the new route. The user can select another station again, if the destination is still not reachable.

In order to correctly calculate the estimated time and power along the route, the user has to indicate the expected charging time when selecting a station. Two possibilities are given: full charge or fast charge. The battery duration will be increased accordingly.

Thus, the user has complete freedom to participate in the route planning by his preference.

5.2 Charging Station Selection Assistance

To ease the route planning process, we provide two assistant features for the user to select stations.

When the destination is not reachable, given the battery life on departure and the maximum range of the electric vehicle, the user needs to decide where to charge the vehicle.

First, the user can turn on the station heat map, which will give him an idea where the stations could be (Figure 8).

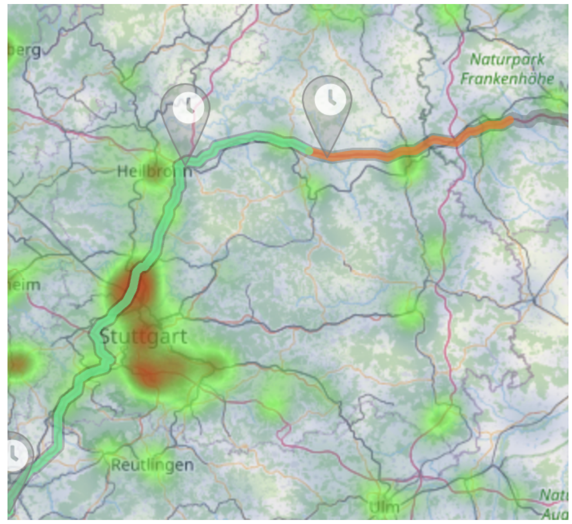


Figure 8: Assistant Heat map Visualization

Besides, the user can hover at a location for 0.5 seconds, then the isochrone and the route to the location will be presented, indicating the reachable range for searching stations and the change of the route (Figure 9).

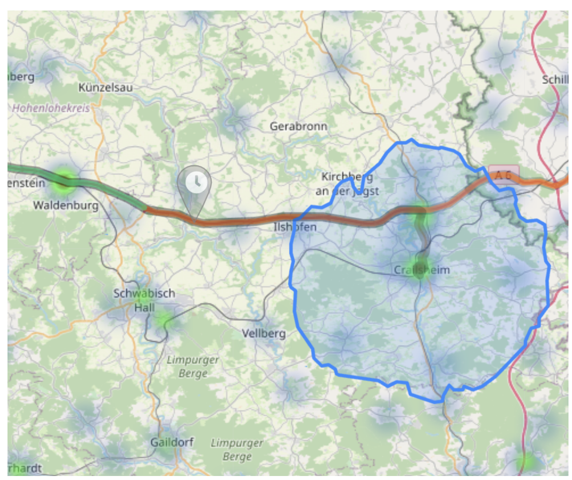


Figure 9: Assistant Hover

Then the user can click the location to search for the station (Figure 10).

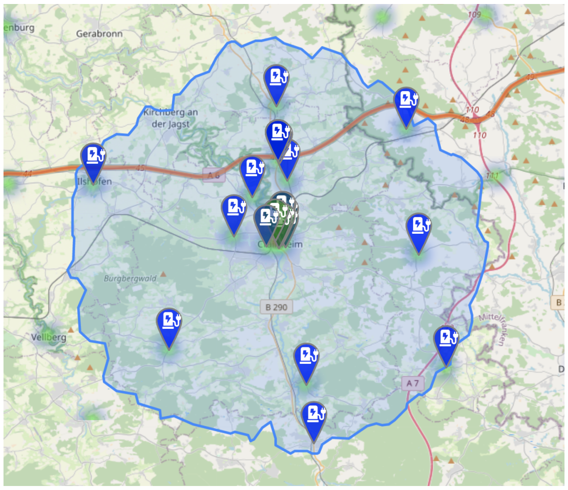


Figure 10: Stations After Clicked

6 Conclusion

In this report, we presented the route planning tool for electric vehicles. The tool highlights the route (orange danger segment) when the destination is not reachable, given the current battery situation, and allows the user to select stations on his own. To ease the process of looking for stations, we adopted the heat map and the hover effect to assist the user. We also integrated amenities data from OpenStreetMap and Michelin restaurants to score the stations and allow the user to explore the surrounding amenities of a station. Besides, we visualized the timing along the route to further help the user understand the route and make a better decision when selecting charging stations.

During the development, we experienced multiple iterations and code refactoring, and managed to maintain a decoupled and extensible project structure. By testing the use cases, we added the user guide and necessary user action alerts to refine the user experience.

For future development, we can extend our data of charging stations and restaurants beyond Germany. Another option is to refine the data and routing considering the socket type and power supply information, which can help the user to avoid the issue when the charging station does not have a usable socket.