**User Interface for an Electric Vehicle**

The UI is one aspect of EVs or automobiles in general, that often has not been the centre of effort and discussion and used to play a minor role in development. However, in order to fully maximize an EV’s potential and to ensure consumer adoption, the design and functionality of UI is an important aspect. Designers have also picked up on that topic [1]. Due to reports from large user field studies, which show that newly developed interfaces can lead to distraction and confusion along drivers [2,3], its important not to change industry standards completely or reinventing the wheel, but integrating new functionality’s within existing features.

**UI issues and its challenges with in electric vehicles**

Drivers of electric vehicles will face new challenges unknown to them if they have been driving conventional automobiles before. The task of the EV’s UI will be to guide the driver through those challenges while still being a positive and rewarding experience.

The driver of electric vehicles will encounter quite quickly major challenges with electric vehicles today, like the charging procedures. Or even a route that will differ from his conventional driving pattern, because it is more energy efficient or even due to the fact that it is the calculated route by the scheduling algorithm for charging stops.

The task of the UI is therefore also to teach and guide the driver through the process. While still keeping up the driving pleasure and not making that trip a frustration. While an unexperienced EV driver will and probably wants more guidance about recharging duration charging procedures etc. an experienced driver probably needs less guidance and information and will even find to much guidance obstructing.

Furthermore, an electric vehicle behaves quite different to a conventional car due to its characteristics. It has a much more limited driving range, higher acceleration performance, lower top speed and due to the large battery a higher car weight and limited interior space. Therefor those factors need also to be acknowledged when designing an UI to make a driver aware of limitations, and eventually warn him or send notifications to inform him.

Battery technology today only allows for limited range in comparison to conventional cars. Depending on design and size the range is below 250km which usually is sufficient for the most trips ([4] 80% with Mini-E, [8] 95% with a range of 100 miles) and usually is even enough for the daily use [5, 6]. But when it comes to a larger trip today a driver is often faced with a challenge of planning a perfect route, with charging stops, which are also not already blocked at the time of his arrival. That’s where our UI will step in so that the Driver can just enter a location and the backend described in the paper “Smart Charging Schedules for Highway Travel with Electric Vehicles” [\*] will calculate the perfect rout for him, set by his criteria’s. It will than be send and displayed to the driver in real time. So that he than can view his trip and know key facts such as time of arrival, duration of charging stops or even an estimated prize for the trips energy costs. Therefore, the time the driver needs to plan his trip is minimized and he is further aware of his energy consumption and trip costs.

Running out of energy is a sever fear among new drivers of electric vehicles. [5] Which will fade with longer usage [7], but even experience drivers fear running low on energy with unplanned trips such as getting to the hospital or supermarket. Which can be solved with a reliable UI that informs the driver before the trip if the battery level is enough for the trip. Or how long it will take until the battery is sufficiently charged till the trip can be done.

New features like energy regeneration while braking need to be taken into consideration as well. [6] Since the UI can inform a driver about optimizing the braking behaviour or style to increase energy recovery. Which then will also have an affect of range and or performance, therefore such a system needs to be abled to adopt different driving styles and desirably learn different driving behaviours. Which than will conclude in a driver that automatically tries to maximize regenerating energy and extend his range [6].

Long stops with recharging times on trips will be unavoidable. Which also is a chance for the UI to entertain the driver by making use of multimedia capabilities, or assisting him with work features such as internet access.

In summary our UI for electric vehicles faces new challenges, which come with the new features and changes of an EV. But also the chance to make use of new technology like regenerating energy, and making it pleasurable to use for the driver that can be added later in development. Challenges like displaying the charging trip and further information about the charging routine will be handled by our UI.

**Comparison of Navigation Frameworks**

The key factor of every navigation front end, is its navigation framework. Today navigation is not just about reducing the total travel distance anymore. A lot of other factors come into play, for example real time traffic information. Especially EV’s might have new factors that haven’t been considered before like, convenient stopping positions for an optimal charging pattern over a journey. Which as ligned out in Victor del Razo and Hans-Arno Jacobsen paper, will be essential when it comes to traveling with an EV. There are two big competitors out there at the moment which are widely used for navigation purposes. Here maps which used to be part of Nokia and now is an independent company and Google maps. Since global coverage and detailed maps are essential for car navigation, only the two biggest were considered for use in an automobile UI, Here’s map set is widely used in the in-dash navigation segment, were its much more widely used than google maps. “By 2013, four out of five cars globally with fully integrated in-dash navigation systems used Here data. Here supplies map content for Alpine, BMW, Mercedes, Garmin, Hyundai, Pioneer, Volkswagen and Toyota among other car companies and enterprises” [12] Even though with the introduction of android auto many car manufactures now support google maps, they still primarily use here maps in there system and provide google maps just as an addition. Therefore, in this paper the use of both map frameworks were analysed and compared for the use in a car UI. Key factors with different weightings are documentation and customizability, coverage and offline use.

**Comparison by documentation and customizability**

Our study performed both the documentation and customizability comparison, with the documentation being more quantifiable. We found the google maps documentation wider and more detailed. With in depth guides to use embedded as a java script api or for mobile devices. On the other hand the Here maps api is also well documentent, but not as wide and in depth as google maps. The google maps api also has a bigger developer scene than here maps since it’s also more widely used outside of the car theme. But since Here maps also provides a JavaScript api that can be implemented quite quickly and effective. A big downside for google maps is its customizability, it is customizable to some extend but has limitations, especially when it comes to adding custom places or even a whole dataset of those. Especially in our scope of creating a front end for a EV’s scheduling algorithm this is a key factor, were a lot of custom charging stations, which are not existent in googles data base, need to be added. With queue times, free queue slots and so on. In here maps this would not be a problem and can be done with some effort.

|  |  |  |
| --- | --- | --- |
|  | Here Maps | Google Maps |
| Documentation | Yes (But not as rich as google maps) | Yes |
| Customization | Yes (Fully) | Yes (But limited) |
| Coverage | 132 | Coverage unknown but bigger than here |
| Full Offline Mode | Yes | Goes online as connection is detected |

**Comparison by offline Usage**

It is very important to address weather or not offline usage is of much importance for the us of the map api or not. According to google “Roughly 60 percent of the world is without Internet today, and even where online access is available, it can still be spotty.” [9] And we also came to the conclusion, that when it comes to the use with in a car were data coverage is not always guaranteed, an offline mode is necessary. Here maps is much stronger in offline usage and is fully functional with low storage usage in offline mode. Google maps also has a recently introduced offline mode but it uses a lot of storages and is limited to a map area, and some country’s.

**Map Framework selection conclusion**

The two frameworks are quite the same when it comes to data set or the coverage they both cover a similar wide area. Were as they are differing when it comes to customizability, even though both are quite good documented here maps is much more customizable than google api. Which will be of importance when it comes to adding custom charging stations, or even a whole charging station data base with information such as queue times, free queue slots to the system. Google maps offline usage is also quite limited to an area needs to be updated regularly, and is not the intended use of google maps were as here maps is much stronger in this area. For those reasons we decided to use the here maps api as the framework to go with in the EV UI.

**The electric vehicle user interface**

The electric vehicle user interface is implemented by using the JavaScript api provided by here maps [15, 16]. For implementation we made use of an example provided by here for the navigation from A to B. An overview of the user interface can be seen in fig 1.

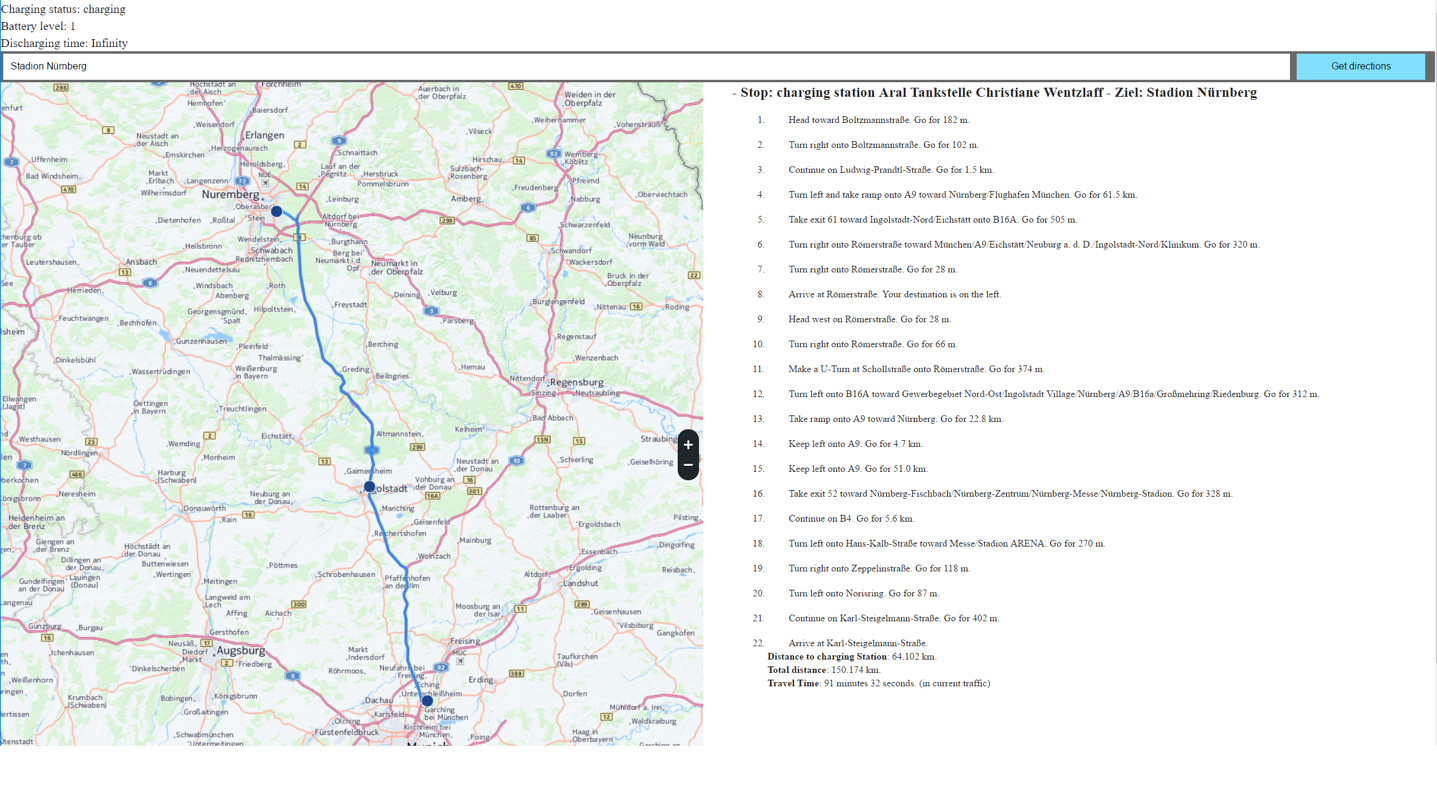


Fig1. User interface

On the left side is the map displayed and the calculated route, in this case from TU Munich to a stadium in Nuremberg, with one charging stop in the middle. At the top is further information about the battery status, for example the charging status and the battery level. On the right side is additional trip information, its only displayed to show the commands which will then be spoken out by text to speech. These should not be displayed in a later version. On the bottom of the left side is further information about the trip like the total travel time and total distance.

**Markup**

As well as for the Simulation Manager Interface the Markup needed to display the user interface is kept minimalistic. At the top are some div elements to display the Battery Status of the device. In this case the computer its running on to show its charging status, the battery level and the time till its fully discharged. If it can not be read it will display “charging state unknown”.

<!DOCTYPE html>  
<html>  
<head ..>  
 <body>  
<div id="charging">(charging state unknown)</div>  
<div id="level">(battery level unknown)</div>  
<div id="dischargingTime">(discharging time unknown)</div>  
<div id="chatbox"></div>  
<button onclick="setButtonPressed();">Try it</button>  
</div>  
  
<div id="map" style="position:absolute; width:49%; height:100%; background:grey" ></div>  
<div id="panel" style="position:absolute; width:49%; left:51%; height:100%; background:inherit" ></div>

Below that are two div elements one to display the map and another one to display the panel. All the rest is in JavaScript.

<head>  
 <meta name="viewport" content="initial-scale=1.0, width=device-width" />  
 <link rel="stylesheet" type="text/css" href="https://js.cit.api.here.com/v3/3.0/mapsjs-ui.css" />  
 <script type="text/javascript" src="https://js.cit.api.here.com/v3/3.0/mapsjs-core.js"></script>  
 <script type="text/javascript" src="https://js.cit.api.here.com/v3/3.0/mapsjs-service.js"></script>  
 <script type="text/javascript" src="https://js.cit.api.here.com/v3/3.0/mapsjs-ui.js"></script>  
 <script type="text/javascript" src="https://js.cit.api.here.com/v3/3.0/mapsjs-mapevents.js"></script>  
 <style type="text/css">  
 .directions li span.arrow {  
 display:inline-block;  
 min-width:28px;  
 min-height:28px;  
 background-position:0px;  
 background-image: url("../img/arrows.png");  
 position:relative;  
 top:8px;  
 }  
 .directions li span.depart {  
 background-position:-28px;  
 }  
 .directions li span.rightUTurn {  
 background-position:-56px;  
 }  
 .directions li span.leftUTurn {  
 background-position:-84px;  
 }  
 .directions li span.rightFork {  
 background-position:-112px;  
 }

For alignment css is used which is embedded in the head of the html file.

**Initialization**

<script type="text/javascript" charset="UTF-8" >  
 window.onload = **function** () {  
 **function** updateBatteryStatus(battery) {  
 document.querySelector('#charging').textContent = battery.charging ? 'charging' : 'not charging';  
 document.querySelector('#level').textContent = battery.level;  
 document.querySelector('#dischargingTime').textContent = battery.dischargingTime / 60;  
 }  
  
 navigator.getBattery().then(**function**(battery) {  
 // Update the battery status initially when the promise resolves ...  
 updateBatteryStatus(battery);  
  
 // .. and for any subsequent updates.  
 battery.onchargingchange = **function** () {  
 updateBatteryStatus(battery);  
 };  
  
 battery.onlevelchange = **function** () {  
 updateBatteryStatus(battery);  
 };  
  
 battery.ondischargingtimechange = **function** () {  
 updateBatteryStatus(battery);  
 };  
 });  
 };

first the battery panel is initiated and set, it will then do automatically subsequent updates.

**Route calculation**

As soon as the back end [] hast resolved the locations for optimal charging stops they can be sent as a list to here maps to get an optimal route from start waypoint0 over potential charging stops waypoint1 to destination waypoint2. The number of waypoints is not directly limited and can vary. Additional parameters such as avoiding areas for example a city centre, or setting the number of alternative routes can be set. For a full documentation have a look at [13]

**function** calculateRouteFromAtoB(platform) {  
 **var** router = platform.getRoutingService(),  
 routeRequestParams = {  
 mode: 'fastest;car',  
 representation: 'display',  
 routeattributes: 'waypoints,summary,shape,legs',  
 maneuverattributes: 'direction,action',  
 waypoint0: '48.2626,11.6679', // TU Munich  
 waypoint1: '48.7752,11.4595', // Aral Petriol Station  
 waypoint2: '49.4268,11.1255' // Stadion Nürnberg  
 };  
  
 router.calculateRoute(  
 routeRequestParams,  
 onSuccess,  
 onError  
 );  
}

When the routing API was successful and gives a response this function will be called. Which then calls the different functions and hands over the route element, to add the calculated route to the map highlighting the waypoints and adding direction commands to a list.

**function** onSuccess(result) {  
  
 **var** route = result.response.route[0];  
 /\*  
 \* The styling of the route response on the map is entirely under the developer's control.  
 \* A representitive styling can be found the full JS + HTML code of this example  
 \* in the functions below:  
 \*/  
 addRouteShapeToMap(route);  
 addManueversToMap(route);  
  
 addWaypointsToPanel(route.waypoint);  
 addManueversToPanel(route);  
 addSummaryToPanel(route);  
 // ... etc.  
}

Route response

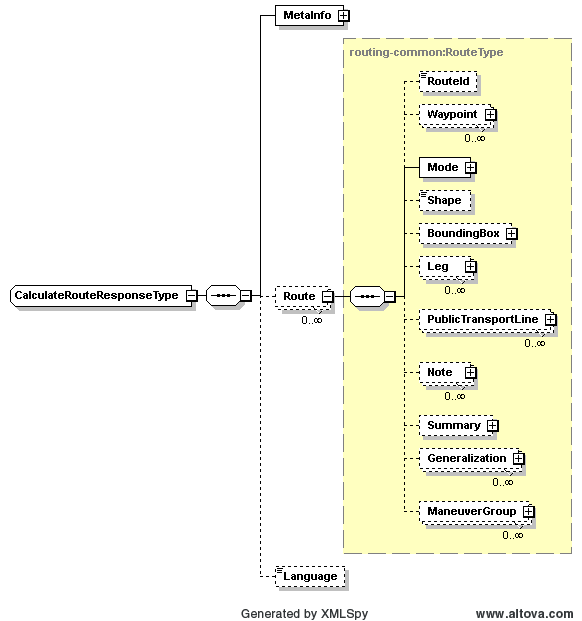


Figure 1. CalculateRouteResponseType [14]

As displayed in figure one the route response contains, besides meta info and information about language for texture information set for the route. The route itself is stored in a route element, which can appear more than once depending on how many routes were calculated.

**Summary Panel**

Here further trip information is added to the panel which could also be extended to for example trip costs, energy used.

**function** addSummaryToPanel(route){  
 **var** summary = route.summary  
 **var** summaryDiv = document.createElement('div'),  
 content = '';  
 **var** sum = [];  
 sum[0] = 0;  
 **for** (**var** i = 0; i < route.leg.length; i += 1) {  
 **for** (**var** j = 0; j < route.leg[i].maneuver.length; j += 1) {  
 sum[i]= sum[i] + route.leg[i].maneuver[j].length  
 }  
 }  
 content += '<b>Distance to charging Station</b>: '+ sum[0]/1000 + ' km. <br/>';  
 content += '<b>Total distance</b>: ' + summary.distance/1000 + ' km. <br/>';  
 content += '<b>Travel Time</b>: ' + summary.travelTime.toMMSS() + ' (in current traffic)';  
  
  
 summaryDiv.style.fontSize = 'small';  
 summaryDiv.style.marginLeft ='5%';  
 summaryDiv.style.marginRight ='5%';  
 summaryDiv.innerHTML = content;  
 routeInstructionsContainer.appendChild(summaryDiv);  
}

Further information about the trip is added to the panel here. For example, direction commands which could be used for text to speech, its displayed at the moment to show the commands but should not be displayed, in a later version since its distracting.

**function** addManueversToPanel(route){  
  
  
  
 **var** nodeOL = document.createElement('ol'),  
 i,  
 j;  
  
 nodeOL.style.fontSize = 'small';  
 nodeOL.style.marginLeft ='5%';  
 nodeOL.style.marginRight ='5%';  
 nodeOL.className = 'directions';  
  
 // Add a marker for each maneuver  
 **for** (i = 0; i < route.leg.length; i += 1) {  
 **for** (j = 0; j < route.leg[i].maneuver.length; j += 1) {  
 // Get the next maneuver.  
 maneuver = route.leg[i].maneuver[j];  
  
 **var** li = document.createElement('li'),  
 spanArrow = document.createElement('span'),  
 spanInstruction = document.createElement('span');  
  
 spanArrow.className = 'arrow ' + maneuver.action;  
 spanInstruction.innerHTML = maneuver.instruction;  
 li.appendChild(spanArrow);  
 li.appendChild(spanInstruction);  
  
 nodeOL.appendChild(li);  
 }  
 }  
  
 routeInstructionsContainer.appendChild(nodeOL);  
}  
  
  
Number.prototype.toMMSS = **function** () {  
 **return** Math.floor(**this** / 60) +' minutes '+ (**this** % 60) + ' seconds.';  
}

this function can open and display an information bubble at a geo position for example information about the queue times at a charging stop provided by the back end. Furthermore, a notification can be raised for example if the energy consumption is to high or the battery is running low.

**Notification Message**

**function** openBubble(position, text){  
 **if**(!bubble){  
 bubble = **new** H.ui.InfoBubble(  
 position,  
 // The FO property holds the province name.  
 {content: text});  
 ui.addBubble(bubble);  
 } **else** {  
 bubble.setPosition(position);  
 bubble.setContent(text);  
 bubble.open();  
 }  
}

Adding the route and the waypoints to the map

here a series of H.map.Marker points from the route is created and added to the map. Furthermore all the manuevers are then added to be displayed on the map.

**function** addRouteShapeToMap(route){  
 **var** strip = **new** H.geo.Strip(),  
 routeShape = route.shape,  
 polyline;  
  
 routeShape.forEach(**function**(point) {  
 **var** parts = point.split(',');  
 strip.pushLatLngAlt(parts[0], parts[1]);  
 });  
  
 polyline = **new** H.map.Polyline(strip, {  
 style: {  
 lineWidth: 4,  
 strokeColor: 'rgba(0, 128, 255, 0.7)'  
 }  
 });  
 // Add the polyline to the map  
 map.addObject(polyline);  
 // And zoom to its bounding rectangle  
 map.setViewBounds(polyline.getBounds(), **true**);  
}

**function** addManueversToMap(route){  
 **var** svgMarkup = '<svg width="18" height="18" ' +  
 'xmlns="http://www.w3.org/2000/svg">' +  
 '<circle cx="8" cy="8" r="8" ' +  
 'fill="#1b468d" stroke="white" stroke-width="1" />' +  
 '</svg>',  
 dotIcon = **new** H.map.Icon(svgMarkup, {anchor: {x:8, y:8}}),  
 group = **new** H.map.Group(),  
 i,  
 j;  
 **var** marker1 = **new** H.map.Marker({  
 lat: 48.2626,  
 lng: 11.6679} ,  
 {icon: dotIcon});  
 group.addObject(marker1);  
  
 **var** marker2 = **new** H.map.Marker({  
 lat: 48.7752,  
 lng: 11.459} ,  
 {icon: dotIcon});  
 group.addObject(marker2);  
  
 **var** marker3 = **new** H.map.Marker({  
 lat: 49.4268,  
 lng: 11.1255} ,  
 {icon: dotIcon});  
 group.addObject(marker3);  
  
 // alert(route.waypoints[1]);  
  
 group.addEventListener('tap', **function** (evt) {  
 map.setCenter(evt.target.getPosition());  
 openBubble(  
 evt.target.getPosition(), evt.target.instruction);  
 }, **false**);  
  
 // Add the maneuvers group to the map  
 map.addObject(group);  
}

[1] 2010. 60 second interview - J. Mays, Ford group vice president of design Automotive Design.(May). 35.

[2] Cocron, P., Buhler, F., Neumann, I., Franke, T., Krems, J.F., Schwalm, M. and Keinath, A. 2011. Methods of evaluating electric vehicles from a user's perspective - The MINI E field trial in Berlin. Intelligent Transport Systems, IET, 5 (2). 127-133.

[3] Kurani, K.S., Axsen, J., Caperello, N., DaviesShawhyde, J. and Stillwater, T. 2009. Learning from Consumers: Plug-In Hybrid Electric Vehicle (PHEV) Demonstration and Consumer Education, Outreach, and Market Research Program. Technical Report. Institute of Transportation Studies, University of California, Davis

[4] Neumann, I., Cocron, P., Franke, T. and Krems, J.F. 2010. Electric vehicles as a solution for green driving in the future? A field study examining the user acceptance of electric vehicles. In European Conference on Human Interface Design for Intelligent Transport Systems (Berlin, Germany, April 29-30, 2010), Humanist Publications, 445-453.

[5] Keinath, A. 2010. Are there differences in the mobility patterns due to BEV? In Abstracts of the 27th International Congress of Applied Psychology

[6] Turrentine, T., Garas, D., Lentz, A. and Woodjack, J. 2011. The UC Davis MINI E Consumer Study. Technical Report. Institute of Transportation Studies University of California.

[7] Hoffman, J. 2010. Does use of battery electric vehicles change attitudes and behavior? In Abstracts of the 27th International Congress of Applied Psychology (Melbourne, Australia, 11-16 July, 2010), 252.

[8] Pearre, N.S., Kempton, W., Guensler, R.L. and Elango, V.V. Electric vehicles: How much range is required for a day's driving? Transportation Research Part C: Emerging Technologies, In Press, Corrected Proof.

[9] https://googleblog.blogspot.de/2015/11/navigate-and-search-real-world-online.html

[10]https://developer.here.com/develop/javascript-api

[11]<https://developers.google.com/maps/documentation/>

[12] Lomas, Natasha (31 January 2013). ["Nokia Chalks Up In-Car Maps Win Against Google, Signs Toyota Motor Europe To Use HERE Local Search"](http://techcrunch.com/2013/01/31/nokia-here-toyota). [Techcrunch](https://en.wikipedia.org/wiki/Techcrunch). Retrieved 10 July 2013.

[13] https://developer.here.com/rest-apis/documentation/routing/topics/resource-calculate-route.html

[14] <https://developer.here.com/rest-apis/documentation/routing/topics/resource-type-calculate-route.html>

[15] <https://developer.here.com/api-explorer/maps-js/v3.0>

[16] https://developer.here.com/api-explorer/maps-js/v3.0/servicesRouting/map-with-route-from-a-to-b