Department of Electrical Engineering and Information Technology
Distributed Multimodal Information Processing Group
Prof. Dr. Matthias Kranz

Development of a Navigation Solution for the Android based Driver Assistance System DriveAssist

Entwicklung einer Navigationslösung für das Android-basierte Fahrerassistenzsystem DriveAssist

Katharina Erhardt

Diploma Thesis

Author:	Katharina Erhardt
Address:	[redacted]
Matriculation Number:	[redacted]
Professor:	Prof. Dr. Matthias Kranz
Advisor:	Dipl.-Ing. Stefan Diewald
Begin:	25.06.2012
End:	25.12.2012

Department of Electrical Engineering and Information Technology
Distributed Multimodal Information Processing Group
Prof. Dr. Matthias Kranz

Declaration

I declare under penalty of perjury that I wrote this Diploma Thesis entitled

Development of a Navigation Solution for the Android based Driver Assistance System DriveAssist

Entwicklung einer Navigationslösung für das Android-basierte Fahrerassistenzsystem DriveAssist

by myself and that I used no other than the specified sources and tools.

Munich, December 20, 2012

Katharina Erhardt

Katharina Erhardt
[redacted]

Abstract

In recent years, Intelligent Transportation Systems (ITS) based on information exchange between traffic participants have been a major research topic of global interest. Various projects have contributed solutions to drive forward the technological means which are necessary to enable communication among vehicles, Vehicle-to-Vehicle (V2V) communication, or vehicles and the environment, Vehicle-to-Infrastructure (V2I) communication. In the course of this process, many different applications have been developed with the goal of increasing road safety and traffic efficiency.

In this thesis, further development of the V2X-based driver assistance system *DriveAssist* is subsequently elaborated on. Up to the starting point of this work, *DriveAssist* already provided, among other things, the functionality of receiving and presenting V2X information on an Android smartphone. In this thesis, the application is enhanced with navigational capabilities such as destination entry, route calculation and route guidance, which transforms the system from a purely informative system to an active navigation system.

The decision to reuse code from already existing open source projects and in this way to promote a fast further development of *DriveAssist* was made at an early stage. After the analysis of freely available software which support OpenStreetMap, the most suitable project *OsmAnd* was selected. As a result, *DriveAssist* and *OsmAnd* were merged together into *DriveAssist 2.0* - an application which differs from its previous version not only in its appearance. It enables the most important navigational functionalities, such as online/offline address search, online routing (via different providers), a prototypic implementation of the A* algorithm for offline routing as well as a guidance engine which provides verbalized and visualized navigation instructions. Furthermore, it also offers the option to download maps or Text-To-Speech (TTS) language packages and a search for nearby Points of Interest (POI) or public transport. Besides, there exists an active developers community around *OsmAnd* and thus, it can be assumed that further improvements will be added in the future, from which also *DriveAssist 2.0* will benefit.

The software implementation was validated in a concluding evaluation. Hereby, various features were tested to obtain objective evidence that the navigational functionalities of *DriveAssist 2.0* are reliable. The offline search of addresses, the functional capacity of the route calculation as well as the route guidance were evaluated in comparison to other established navigation applications on the market.

Finally, *DriveAssist 2.0* enables a future extension of the routing algorithm which may take additional information from V2X and Central Traffic Services (CTS) into account. Moreover, the application can also be used for evaluating whether *DriveAssist 2.0* is able to increase the awareness for potential dangers on the road or not.

Kurzfassung

In jüngster Zeit stellen intelligente Verkehrssysteme (ITS), welche sich Informationsaustausch zwischen verschiedenen Verkehrsteilnehmern zunutze machen, ein weltweit wichtiges Forschungsgebiet dar. Diverse Forschungsprojekte haben dazu beigetragen, die technischen Möglichkeiten, die für die Kommunikation zwischen Fahrzeugen, die sogenannte Fahrzeug-zu-Fahrzeug (V2V) Kommunikation, oder zwischen Fahrzeugen und der Umgebung, Fahrzeug-zu-Infrastruktur (V2I) Kommunikation, voranzutreiben. Im Zuge dessen wurden zahlreiche Anwendungen entwickelt, mit dem Ziel, Verkehrssicherheit und -effizienz zu erhöhen.
In dieser Diplomarbeit wird die Weiterentwicklung des V2X-basierten Fahrerassistenzsystems *DriveAssist* sukzessive erarbeitet. Vor Beginn dieser Arbeit konnte die Applikation *DriveAssist* unter anderem bereits dazu genutzt werden, V2X-Informationen über ein Android Smartphone zu empfangen und darauf anzuzeigen. Im Rahmen dieser Arbeit wird das bestehende System um wichtige Navigationsfunktionen wie Zieleingabe, Routenberechnung und Routenanweisungen erweitert und wandelt es hierdurch von einem rein informativen System in ein aktives Navigationssystem um.
Zu einem frühen Zeitpunkt wurde die Entscheidung getroffen, auf einem Open-Source-Projekt aufbauend, existierende Navigationsfunktionen für Android wiederzuverwenden, um entsprechende Entwicklungsziele für *DriveAssist* möglichst rasch zu erreichen. Hierfür wurde durch eine Analyse des Marktes frei verfügbarer Software, welche OpenStreetMap unterstützen, schließlich das Projekt *OsmAnd* ausgewählt. Durch das Zusammenführen der beiden Projekte *DriveAssist* und *OsmAnd* entstand *DriveAssist 2.0*, welches sich nicht nur in seiner Erscheinung von seiner Vorgängerversion unterscheidet. Vielmehr bietet es die wichtigsten Navigationsfunktionen wie online und offline Adresssuche, online Routenberechnung (über verschiedene Provider), eine Prototyp-Implementierung des A*-Algorithmus zur offline Routenberechnung, ebenso wie eine Engine zur Bereitstellung visueller und verbalisierter Navigationsanweisungen. Darüber hinaus bietet das System die Möglichkeit, Karten oder Text-To-Speech (TTS) Sprachpakete herunterzuladen und nach nahegelegenen Points of Interest (POI) oder öffentlichen Verkehrsmitteln zu suchen. Außerdem existiert um *OsmAnd* eine aktive Entwickler-Community wodurch davon ausgegangen werden kann, dass zukünftige Verbesserungen enstehen werden und auch *DriveAssist 2.0* hiervon profitieren wird.
Die Softwareimplementierung wurde abschließend in einer Evaluierung überprüft. Dabei wurde getestet, wie verlässlich die verschiedenen Navigationsfunktionen von *DriveAssist 2.0* sind. Hierbei wurden in einem Vergleich mit im Markt etablierten Navigationssystemen die Funktionen offline Adresssuche, Routenberechnung und Routenführung bewertet.
Zu guter Letzt ermöglicht *DriveAssist 2.0* eine zukünftige Erweiterung des Routing-Algorithmus, welcher zusätzliche Informationen aus V2X und zentralen Verkehrsdiensten (CTS) verarbeiten könnte. Schließlich kann die Anwendung dazu genutzt werden, die Wirkung auf das Gefahrenbewußtsein des Fahrers mit Hilfe von *DriveAssist 2.0* zu testen.

Contents

Chapter 1.

Introduction

1.1. Motivation

Over the course of only a few decades, the field of information and communication technology fundamentally changed people's lives. Many industries, from education to communication, from medicine to economics profited from the rapid development of this technology. Today, universities are able to deliver online courses, e-mail, peer-to-peer services, broadband video and instant messaging allow communications to take place worldwide. Computer-aided diagnostic processing has already become an important part of clinical routine and the ongoing process of economic globalization that allows companies to partner up with organizations from all over the world have become indispensible [1, p. 1].
Now, the vision of Intelligent Transportation Systems (ITS) providing improvements in transportation system performance, including reduced risks in transportation and congestion as well as increased safety and comfort driving experience, becomes reality: more and more components of transportation systems - vehicles, traffic lights, roads, etc. - are equipped with microchips and sensors and thus become intelligent [1, p. 1] [2, p. 11].
For ITS, the Vehicle-to-X (V2X) communication is a highly promising technology whose aims are of very deverse nature. It is aimed at increasing road safety by reducing accidents or at least, in case of non-avoidable accidents, reducing the impact. Another important goal is to increase traffic efficiency, e.g. with traffic congestion control, and in this way to reduce transport time and fuel consumption. V2X communication is further supposed to make communication and information services available which may provide comfort and business applications to driver and passengers[1] [3, p. 3105].
This thesis is dedicated to the further development of the V2X-capable Advanced Driver Assistance System (ADAS) *DriveAssist* which runs on the Android platform. It allows the visualization of traffic information originated from Central Traffic Services (CTS) and V2X communication services. Furthermore the application is able to display warning messages for certain traffic incidents.

[1]Benefits of V2X communication, `http://www.car-to-car.org/index.php?id=9`, accessed November 16, 2012

In this manner, the driver gets a very good overview of the actual traffic situation [4].
What is missing is a navigational solution for *DriveAssist*. Before presenting the detailed solution for this problem, a short introduction into the subject is given in the following.

1.2. Related Work

Research in the area of integrating mobile devices - tablets and smartphones - into the automotive environment has made considerable progress over the last few years.
The benefits provided by such an approach are manifold and encouraging. According to Diewald et al. [5] mobile devices are predestined for the use in the automotive industry: for the customer as well as for the manufacturer. The former benefits from an easy to learn Natural User Interface (NUI) and from additional advantages provided generally by a modern personal assistant. The access to a huge range of information, such as the address book or the events in the calender, for example, integrates contextual information in the automotive environment. The manufacturer benefits from the ability to finally have a way to bridge the lifecycle gap between automotive and consumer devices.
Besides, it will no longer be necessary to split the guidance of the task of getting to a destination in different states like first driving with the car (usage of in-vehicle navigation), second using public transport (usage of a smartphone application for getting the optimum transport connections) and finally, walking the rest of the way (usage of a smartphone application for getting the route on foot). Instead, the mobile device could create an overall solution and guide the user conveniently from the front door to the desirable final destination.
By now, there are already existing approaches in the field of integration of the features offered by the smartphone into the car. Both, Diewald et al. [5] and Hess et al. [6], agree in that fact, but nevertheless, challenges still remain, including for example the choice of the appropriate integration architecture.
The pioneers in this field are already able to provide solutions for the specific problem of combining the automotive industry and the mobile phone sector in such a way that portable devices can contribute to increase the comfort or the road safety of the driver.
A prototype accident and notification system implemented as an open-source Android application called *WreckWatch* aims at automatically detecting traffic accidents using accelerometers and acoustic data. In case of an accident, it immediately notifies a central emergency dispatch server and provides the required information about the accident (e.g. GPS coordinates or even if available photographs from witnesses) [7].
With the same goal in mind, but with a different implementation, Zaldivar et al. [8] created another application for the Android platform. Their application monitors a vehicle through wireless OBD-II interface and in this way is able to trigger an automated warning procedure. In case an accident is detected, the critical data is read from the vehicle's bus, the information about the

accident is sent via e-mail and SMS and thereafter an emergency call is started. Remarkably, the whole process takes less than three seconds.

Other projects, such as the *Driving Coach*, are dedicated to making eco-driving a reality. This smartphone application presented by Araujo et al. [9] evaluates the driving fuel efficiency and determines useful suggestions and hints to be shown to the driver in real-time with the aim of improving the energy efficiency of the vehicle. A study by Tulusan et al. [10] (realised with the help of a product of DriveGain Ltd., UK,) an eco-driving application was tested with regard to the effectiveness of changing driving behaviour by feedback technologies. This application provides context related eco-driving feedback during driving. As a result, an improvement in the overall fuel efficiency by 3.23% could be achieved.

The idea of CARMA (Car Mobile Assistant) combines all above mentioned ideas. Flach et al. [11] present a system that provides high-level abstractions for, among other things, sensing automotive parameters. In this way developers can easily write smartphone applications to achieve fuel efficiency, responsiveness, or safety goals. *TuneWizard* calculates a route while taking into account environmental characteristics, like the type of terrain (flat, low, hills etc.), the traffic light density (no lights, few lights, and many lights) or the maximum speed limit on the route. The *ValetMode* application responsible shall ensure that the driver uses only the two lowest gears and enforces an RPM limit of 2000. *DriverCustomizer* is a reflective application which monitors the past driving performance and tries to prevent aggressive driving [11].

The examples above show the high potential of mobile devices in the automotive domain. By combining this with V2X communication which enables cooperation between vehicles, it is possible to increase the effectiveness of safety and traffic efficiency applications to a large extent [3].

In this context, Diewald et al. [12] introduces, how mobile devices with V2X communication can be integrated in the automotive environment.

The so-called *CODAR* (Cooperative Object Detection And Ranging) is a concrete example of such a specific system. It provides not only individual but above all cooperative awareness by visualizing other road users with their position, speed and heading [13].

Moreover, Chen et al. [14] propose an algorithm how to consider information of surrounding vehicles and road conditions with the aim to reduce the fuel consumption of the controlled vehicle. As yet, the approach is theoretical, but it is certain that future research will focus more on the practical implementation.

Similarly, Wedel et al. [15] present a new algorithm that can be used to optimize routes circumnavigating congested roads. For this purpose, each vehicle transmits its average speed to vehicles in the neighbourhood. The information about current possible speeds in the neighbouring roads can be used to recalculate the best route to take for a certain vehicle. Simulations on this approach were done and it was found that navigation systems using V2X technology for calculating a more intelligent route can improve the traffic efficiency of future transport systems.

Another very interesting paper presented by Weiß [3] gives on overview about the development of

V2X communication in Europe and, especially, in Germany.

1.3. Thesis Scope

The major goal of this thesis is the development of a navigation solution for the already existing Android-based driver assistance system *DriveAssist*. The extended version *DriveAssist 2.0* has to provide in addition to the previously offered functionalities including basic navigation features. In particular, the following thesis objectives were defined:

- → The new application has to provide offline searching and routing.
- → The calculated route should be displayed on top of the map.
- → The navigation and routing guidance instructions have to be displayed to the user.
- → A Text-To-Speech (TTS) system has to verbalize the navigation instructions.
- → The existing User Interface (UI) should be enhanced with the new interaction possibilities.

1.4. Thesis Structure

An outline of the topics presented in this thesis which is divided into seven chapters is illustrated in Figure 1.1.

Chapter 1 serves as an introduction to the reader in terms of motivation for enhancement of the Android-based driver assistance system *DriveAssist*. Furthermore, the related work part, the thesis scope definition and an overall overview of this thesis can be found in this chapter.

Chapter 2 outlines the basic components of modern navigation systems. It gives a first introduction to themes, such as the database, positioning systems, the map matching, the destination entry, the route calculation, the route guidance and the UI of such systems.

Chapter 3 addresses opportunities for digital map enhancement of ADAS. It provides two different kinds of classification of such systems and outlines some examples, including the Intelligent Speed Advisory (ISA), the Curve Speed Warning System (CSWS) and the adaptive Human Machine Interface (HMI).

Chapter 4 is dedicated to Android navigation and routing software which support the open source project OpenStreetMap. With the help of the requirements listed in this chapter freely available software was reviewed. In particular, the documentation of the analysis of three selected projects (*Gosmore*, *Navit*, *Osmand*) can be found in this chapter. Finally, it presents the most suitable open source project for a successful enhancement of *DriveAssist*.

Chapter 5 presents a description of the implementation environment and the new available features of *DriveAssist 2.0* from the user's point of view.

Chapter 6 describes the evaluation of *DriveAssist 2.0*. Therefore, it gives an overview over the comparison of the application with *Google Maps* and *Route 66 Maps + Navigation* in regard to the interaction complexity for a route search as well as the functional capacity of the route calculation and the guidance module.

Chapter 7 summarizes the most important results of the thesis and provides an outlook of further developments.

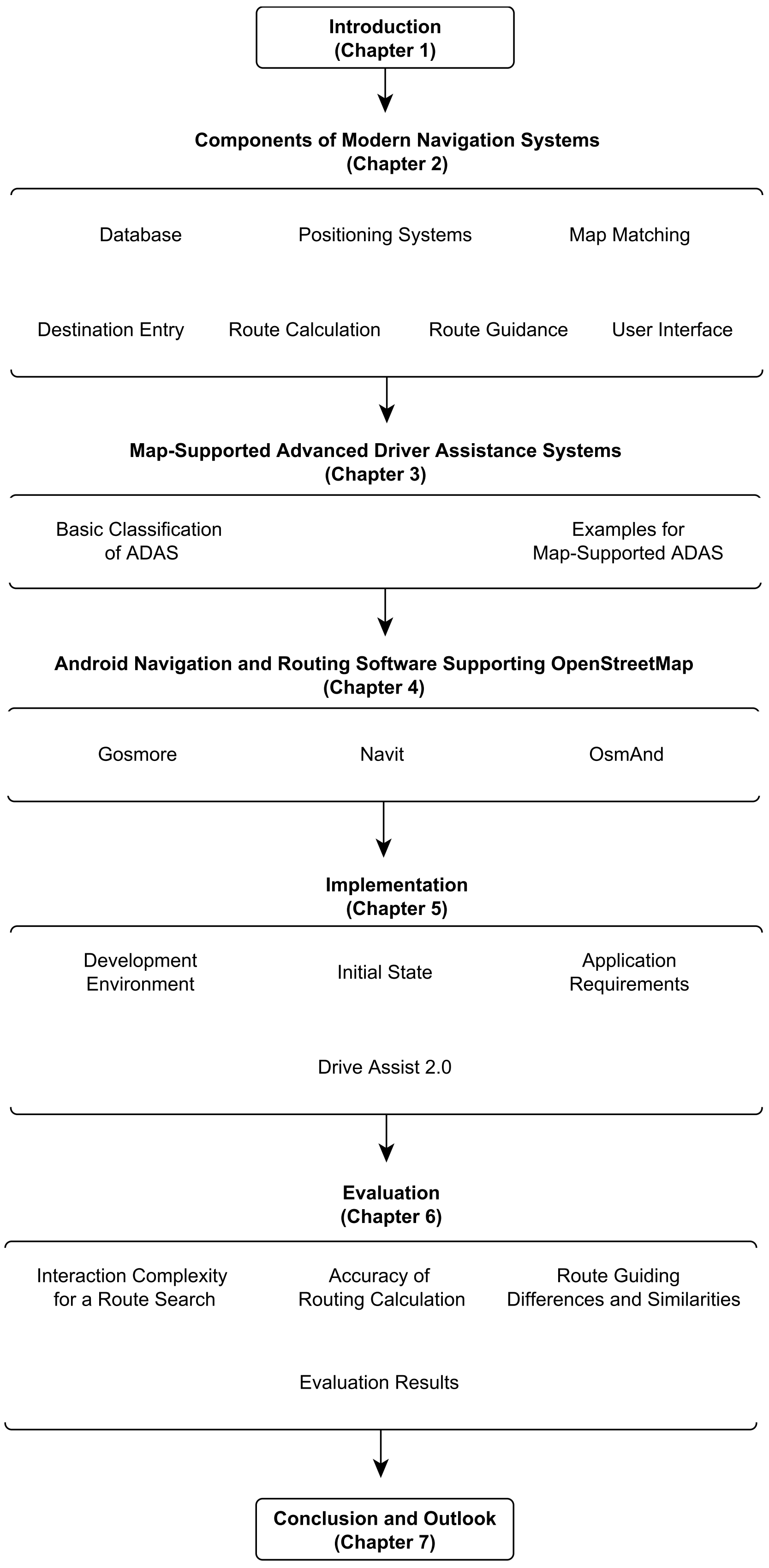


Figure 1.1.: Thesis Structure Overview

Chapter 2.

Components of Modern Navigation Systems

In the past years navigation systems have gained wide acceptance [16, p. 192]. At the end of the year 2011, about 340 million navigation systems were in use worldwide, including about 60 million in-dash navigation devices. Above all, however, the Portable Navigation Devices (PND) which are represented by 150 million (about 44% of 340 millions) and navigation-enabled mobile phones (130 million, about 32% of 340 millions) have led to the rapid spread of this technology[1]. Although navigation systems can be divided into multiple categories and can differ a lot from one another in regard of quality or appearance, the aim of this chapter is not to take a closer look at the differences but at what all systems have in common [17, p. 341ff.] [16, p. 192ff.].

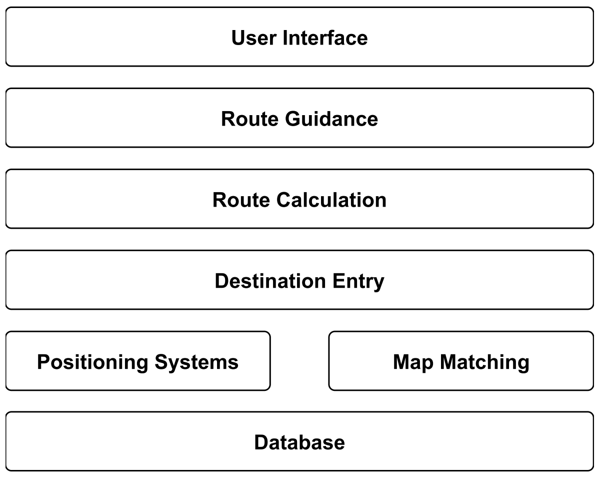


Figure 2.1.: Basic components of modern navigation systems: a reliable database, positioning systems, a map matching module, the destination entry, the route calculation, the route guidance and the UI.

Figure 2.1 shows the main components of modern navigation systems. These include, among other things, the following:
A reliable database, whose central role is shown in Section 2.1. Positioning systems which provide

[1]LBS Research Series: Mobile Navigation Services and Devices, `http://www.berginsight.com/ReportPDF/ProductSheet/bi-mns5-ps.pdf`, accessed October 22, 2012

accurate determination of the current user location (see Section 2.2 for more details). A map matching module which is responsible for finding a given position as a longitude/latitude pair as precisely as possible on a map (see Section 2.3). A destination entry (Section 2.4) which is absolutely essential for executing route calculation (on this, see Section 2.5). A route guidance which gives recommendations for the direction of travel (Section 2.6). And, of course, the UI due to the fact that the market acceptance of navigation systems depends not only on the quality of the map data, the speed or the accuracy of the search algorithm, but also on the quality of interaction with the user (see Section 2.7).

2.1. Database

To create a solid basis for an efficient modern navigation system, a database is needed which in contrast to a paper map does not only contain spatial information (e.g., names and courses of roads), but also very essential aspatial information, such as turn-by-turn instructions and various geospatial query-processing capabilities (e.g., finding the nearest neighbour to a given location, finding the shortest route between two points) [18, p. 45f.] [17, p. 344].
The representation of a geographic area in such a way that it can be managed and processed digitally is only possible with the so-called *digital map* [19, p. 183]. There are principally two different ways to handle this kind of information:
One possibility is to make use of the *raster model*. Thereby, cells which are also called grid cells, raster cells, picture elements or pixels are used for the graphical representation. A row-and-column grid is used to localize the pixels, each with a specific value to represent the shading of the cell (see Figure 2.2(a)). Usually, raster maps tend to be very large and thus require a lot of memory [20, p. 32] [19, p. 183] [21, p. 183f.].

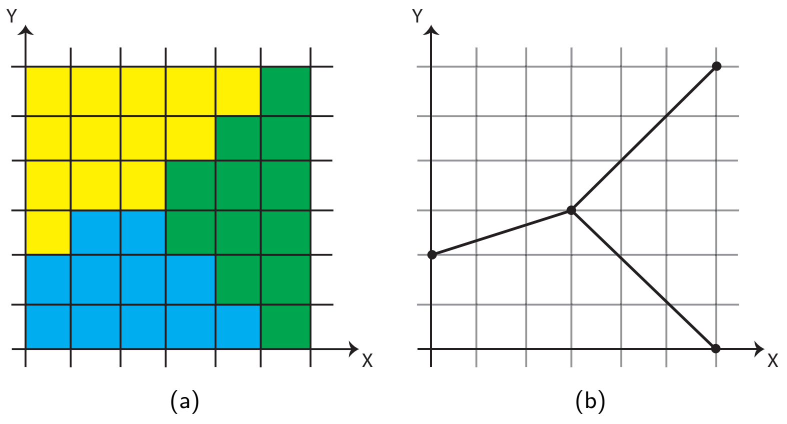


Figure 2.2.: Two different ways of graphical representation: (a) shows the raster model, (b) shows the vector model.

Another possibility is to use the *vector model* (see Figure 2.2(b)). In this case, the information is expressed in coordinates. Points, lines, splines and polygons are used to create a digital database.

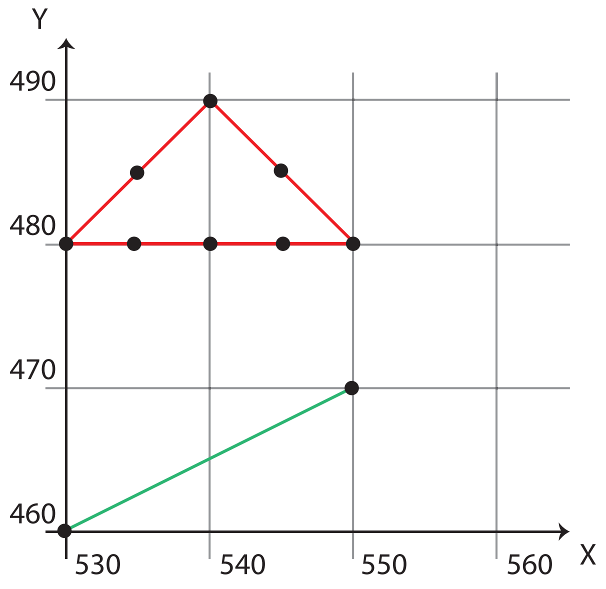


Figure 2.3.: Geometrical data (triangle and a line) which is used to explain the vector model representation.

The following example illustrates how the vector model can be used. To represent Figure 2.3 with the vector-based model the following data is required [21, p. 183ff.] [19, p. 183ff.]:

1. A table of digitized x- and y-coordinates of points is needed (Table 2.1 refers to Figure 2.3).
2. To provide topology or other non graphic information, a file which defines a codification is necessary. In this example, the topology information is stored in four different parameters: (a, b, c, d), see Figure 2.4.
   The value of the parameter *a* can be either *L* or *D*. In the case of *L*, a straight line segment between the points which are listed under the identification number *c* and *d* should be drawn. In the case of *D*, a chain of straight line segments between all points *c* and *d* should be drawn. The parameter *b* defines the colour which has to be used for drawing (number 1 represents red colour, number 2 represents green colour). *c* and *d* are identification numbers of the Table in 2.1.
3. A file with the topology information has to be created. For the Figure 2.3 it would look like this: D,1,1,8,L,2,9,10,0,0,0,0. The last four zeros define the end of the file.

Using vector-based maps offers many advantages: It is easily possible - only with the help of simple analytical handling and processing of coordinate geometry - to produce various useful quantities, such as distances, angles, analytical shading of areas, etc. Besides, vector symbols allow a high number of transformations: they can be easily moved or copied, scaled, rotated or projected. In addition, vector-based maps require less memory space than raster-based maps and their plotting takes place with higher precision and higher speed [19, p. 199f.].

Table 2.1.: Table of coordinate points which refers to the geometrical representation in Figure 2.3

Point Identification Number	x-Coordinate	y-Coordinate
1	540	490
2	535	485
3	530	480
4	535	480
5	540	480
6	545	480
7	550	480
8	545	485
9	550	470
10	530	460

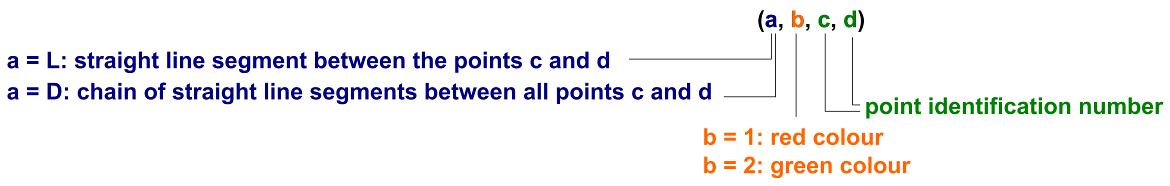

(a, b, c, d)
a = L: straight line segment between the points c and d
a = D: chain of straight line segments between all points c and d
point identification number
b = 1: red colour
b = 2: green colour

Figure 2.4.: Four parameters to define topology or non graphic information for Figure 2.3.

2.2. Positioning Systems

The existence of a positioning system is absolutely essential for the determination of the location of an object on the surface of the earth and for this reason one of the key requirements for a successful navigation. In general, such systems can be classified in five positioning categories [22, p. 177ff.]:

1. **Satellite Positioning**, which relies on satellites of Global Navigation Satellite System (GNSS).
2. **Cellular Positioning**, which is achieved by exploiting information from the terrestrial cellular network used in mobile communication systems.
3. **Wireless Local/Personal Area Network Positioning**, which make either use of communication systems whose main purpose still remain data transmission (e.g.,Wi-Fi, Bluetooth,

Ultra-Wideband (UWB)) or which use dedicated positioning solutions which require specific hardware for this purpose (e.g., Radio Frequency Identification (RFID), infrared and ultrasound technology).

4. **Ad-hoc Positioning**, which is used to determine the relative localization with the help of other reference nodes in an ad-hoc network.
5. **Hybrid Positioning**, which offers more accuracy and reliability on the determined position by combining positioning systems mentioned above.

In the following, the different types of positioning systems are explained in more detailed.

2.2.1. Satellite Positioning

GNSS are those positioning systems that are based on analysing radio signals transmitted by orbiting satellites. For the localization of every place on earth at least four satellites are required. By broadcasting their own positions and clock times, they provide enough information for a GNSS receiver to localize itself. First, the receiver calculates the time of flight, and from that, the distance to the satellites. Second, with the help of the calculated distances and the positions of the satellites the receiver is able to determine its own position, including longitude, latitude and altitude. In practice, GNSS receivers usually evaluate signals from more than four transmitters in order to minimize the transmission and estimation errors [23, p. 161] [22, p. 178f.].
The most prominent representatives of such systems are the American Transit, the Global Positioning System (GPS) (officially the Navigational Satellite Timing and Ranging (NAVSTAR) GPS), the Russian Globalnaya Navigationnaya Sputnikovaya Sistema (GLONASS), the European GALILEO and the Chinese Compass [23, p. 161] [22, p. 178f.].

2.2.2. Cellular Positioning

Even if GPS-enabled cell phones have become a commodity. The field of nonsatellitebased positioning is still relevant. Referring to this, generally two different approaches exist: mobile-assisted GPS-free positioning and mobile- or network-based GPS-free positioning. In the first case - which has not yet been widely introduced -, the mobile calculates its position using signals received from Base Transceiver Stationss (BTSs) and in the second - quite common - case the position of a mobile is determined on a server in the network.
Mobile network operators, for example, use a simple network-based GPS-free positioning method based on the cell coverage. Every BTS has its own cell ID, so an approximate positioning according to the connection of a mobile phone to one of these BTSs can be done by simply evaluating the cell identification.
For a better mobile's position estimation a more advanced network-based GPS-free positioning

method can be used. In such a case, the mobile phone is connected to a multiantenna BTS and the location is then assumed to be the center of gravity of the sector that the mobile belongs to. More complex positioning methods evaluate the Time of Arrival (TOA) and the Time Distance of Arrival (TDOA) of radio signals [22, p. 185ff.].

2.2.3. Wireless Local/Personal Area Network Positioning

Compared with cellular or satellite-based positioning systems, the distances between communication nodes when using Wireless Local/Personal Area Network Positioning are small. Numerous applications which use this positioning method can be found in all branches of industry (home entertainment, consumer electronics, the automotive industry, public transportation, heterogeneous cellular networks). Representatives of this positioning systems: UWB, Bluetooth, WLAN, RFID, infrared and ultrasound technologies [22, p. 200].

2.2.4. Ad-hoc Positioning

Ad-hoc positioning provides the determination of locations based on other known geographic locations. This is interesting as with this positioning method it is, for example, possible to find locations without the aid of GPS. This method would also enable localization in isotropic large networks - without the use of large routing tables.
In this case, the location is determined in a relative coordinate system, with the help of the knowledge of the positions of other participants. The measure of TOA or TDOA and the knowledge of other locations relative to the searched one, provide to collect information about the network topology and based on this information the searched positions can be estimated. As an example of GPS-free positioning in mobile ad-hoc networks, Capkun et al. [24] present some research in this field [25] [22, p. 207f.].

2.2.5. Hybrid Positioning

The coexistence of different wireless access technologies allows the creation of heterogeneous positioning systems. This can be used to facilitate increased positioning accuracy and, furthermore, to provide better coverage for Location Based Service (LBS).
There are two different kinds of heterogeneous positioning. The first one is aimed at introducing a homogeneous interface between indoor and outdoor positioning systems and thus to extend conventional indoor positioning systems. Figueiras et al. [22, p. 208ff.] provide more information on this topic. The second one is aimed at extending conventional outdoor positioning systems and to reduce the position acquisition times. Assisted GPS is an example for such systems. Herby,

cellular positioning is combined with GNSS and thus, the performance with respect to the startup time, sensitivity and power consumption can be improved [22, p. 208ff.].

2.3. Map Matching

The aim of map matching is to find a position given as a longitude/latitude pair with the most probable location with regard to a map. The problems that arise here are of a very complex nature because the given location might deviate from the true position by tens to hundreds of meters, and in addition, digital maps contain inaccuracies [26, p. 741ff.].
In general, map-matching techniques can be broadly classified as geometric, probabilistic, and fuzzy. Schiller et al. [18, p. 60] provide more details on this topic. In the context of this work the classification of the three basic geometric techniques of map matching are introduced [26, p. 741ff.]:

1. **Point-to-Point Matching**, with the goal to find the closest node n_i to the measured position *p* (see Figure 2.5(a)).
2. **Point-to-Curve Matching**, the objective is to find the closest curve from the measured point *p* (see Figure 2.5(b)).
3. **Curve-to-Curve Matching**, uses not only the current point p_n, but an entire polyline of historical positions p_0,p_1, ... ,p_n, the so-called trace of the vehicle, simultaneously to find the most probable line segment (see Figure 2.5(c)).

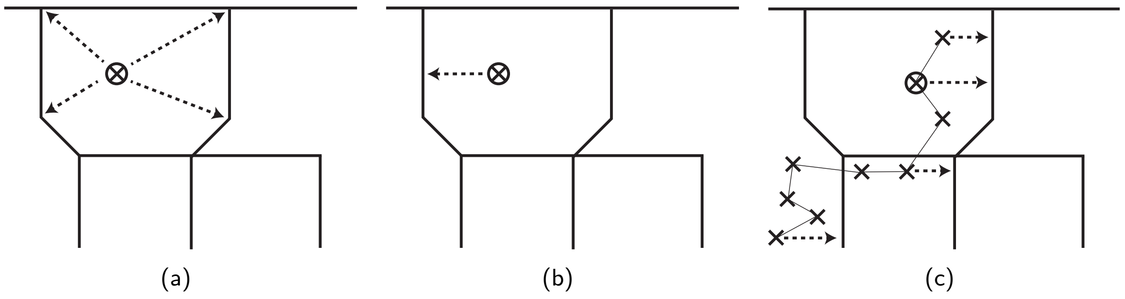


Figure 2.5.: Three basic geometric techniques of map matching: (a) refers to Point-to-Point Matching, (b) refers to Point-to-Curve Matching, (c) refers to Curve-to-Curve Matching.

2.4. Destination Entry

There are means of possibilities for entering the destination. The driver can search a location by an address, by a latitude/longitude pair or clarify the intended goal with the search for **POIs!** (**POIs!**),

such as restaurants, gas stations or shopping centres. In the context of POI-search the so-called proximity search is very helpful [27, p. 604].
The search for an address is usually done by entering first a country, than a city, than the street and the house number - in particular in Europe this order of entering the information for a searched location has prevailed (see also Figure 2.6). By expedient data thinning after every entry the list of possible selections becomes more limited. For example, after choosing a town the user see only the streets which belong to this specific area [27, p. 604].
As the destination entry can be a very time-consuming process there is a popular, helpful feature called the Automatic Spelling Function (ASF) which is responsible for greying not possible combination of letters and thus precluding the selection of not desirable locations [27, p. 604].
The term geocoding is used for determining a location which is described textual, such as a street address, into a location on a map [26, p. 744].

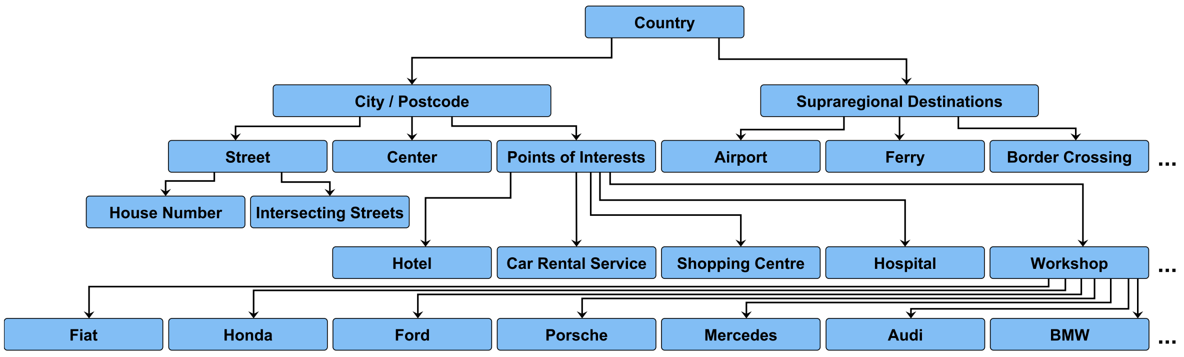


Figure 2.6.: An example for a possible tree structure of a destination entry [27, p. 604].

2.5. Route Calculation

Based on the determined position and the selected destination the route calculation process can be started. As the topological information can be represented in a graph, the task of calculating a route can be formulated as an optimization problem [17, p. 348].
In this manner, the aim of the route calculation is generalized to a cost minimization task in an abstract cost model. Additional constraints include distance, travel time, average speed, number of turning maneuvers or traffic lights, dynamic traffic information etc. [17, p. 348].
Over the last decades, it became clear that the most suitable among a wide range of different algorithms are the Dijkstra and the A* algorithm that are designed to find the shortest path in a weighted graph [17, p. 348]. In general, the most commonly used routing algorithm in modern car navigation systems is an approximation algorithm based on A* [26, p. 744].
However, the major challenge of modern navigation systems is still the adaptation to real-life conditions. This means that traffic conditions, and of course traffic rules should be included in

the calculation. Besides, the problem of route calculation in real-time is computationally still very challenging due to the large size of maps and due to the constraints on the available computation time of the algorithm. The future trend of personalization of planned routes by an increasing adaptation of the used cost functions to the preferences of the individual driver, will not make the situation better. So, to speed up the calculation, meanwhile some shortcut heuristics are applied, but unfortunately this approach leads sometimes to nonoptimal solutions and is not always satisfying enough [26, p. 745].
In general, the cost of a path P with edges $e_0, ..., e_n$ is defined as

$$w(P) = \sum_{i=0}^{n} w_e(e_i) + \sum_{i=0}^{n-1} w_r(e_i, e_{i+1}), \tag{2.1}$$

where w_e is the edge cost and w_r is the cost associated with turning from e_i to e_{i+1}. To sum up, a navigation system is always looking for the most cost-efficient route. In the interests of clarity, this chapter will introduce a possible classification of algorithms used for route calculation. For a detailed explanation of every individual approach, the reader is referred to the original source.

2.5.1. Heuristics for Route Planning

If the assumption is made that the aim is to find the shortest route, the modified A* algorithm can use an h-value, based on the Euclidean distance[2] from the geographical starting location *L(n)* (associated with the node *n*) to the location of the destination *L(d)*, which correlates with the node *d*. Hence, the heuristic function h(n) = d(L(n),L(d)) is a lower bound, since the shortest way to the goal must be at least as long as the air distance [26, p. 745f.].
For calculating the fastest route, generally the travel time estimation is based on road classes, that means that every part of the road network corresponds to a certain road class. The classification according to road classes can be used to indicate the importance of a road (e.g. the higher the road class number the less important the road) and besides, an average speed is associated with each road class. With the help of such a classification the cost of an edge is determined as its length, divided by this speed. In this case, the h-value is the Euclidean distance divided by the overall maximum speed v_{max}. Another option modern navigation systems provide, combines the search for the fastest and the shortest route. Such a calculation is only possible with a parameter value τ which determines the relative weights of a linear combination of the two possibilities. In this case the heuristic estimate h_τ for a node *n* is defined as [26, p. 746]:

$$\begin{aligned} h_\tau(n) &= \tau * \frac{1}{v_{max}} * \|L(n) - L(d)\|_2 + (1-\tau) * \|L(n) - L(d)\|_2 \\ &= \|L(n) - L(d)\|_2 * (\frac{\tau}{v_{max}} + (1-\tau)) \end{aligned} \tag{2.2}$$

[2]The Euclidean distance between Cartesian coordinates $p_1 = (x_1, y_1)$ and $p_2 = (x_2, y_2)$ is defined as $d(p_1, p_2) = \|p_1 - p_2\|_2 = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$, where $\|*\|_2$ is called $L_2 - norm$

2.5.2. Time-Dependent Routing

A considerable challenge calculating the optimal route is also the fact that a lot of road network properties change regularly over time. For example, closed roads, congestion during rush hours, traffic lights timing, timed speed and many more additionally conditions [26, p. 746f.].
To take this time-dependent factors into account, the basic model explained in the Section 2.5.1 has been extended, so that the weight functions w_e and w_r accept an additional argument, representing the time, such that the costs become time-dependent. In [26, p. 746f.] the following formalism is presented [26, p. 746f.]:
A function $t_e(e,t)$ is defined to denote the time needed to pass an edge, another one $t_r(e_1,e_2,t)$ represents the time needed for turning between edges at time t. For the case of the shortest route claims: $t_e = w_e$ and $t_r = w_r$. This definition is also well suited, for example, for modeling phases of stoplights. The cost of a path $P = (e_0, ..., e_n)$ at departure time t_0 is calculated as:

$$w(P,t_o) = w_e(e_0,t_o) + \sum_{i=1}^{n} w_e(e_i, t_i + t_r(e_{i-1}, e_i, t_i)) + \sum_{i=1}^{n} w_r(e_{i-1}, e_i, t_i), \tag{2.3}$$

where $t_{i+1} = t_i + t_r(e_{i-1}, e_i, t_i) + t_e(e_i, t_i + t_r(e_{i-1}, e_i, t_i))$ denotes the arrival time at the head node of edge e_{i+1}. Further details can be found in [26].

2.5.3. Stochastic Time-Dependent Routing

The routing calculation can become even more complex. The calculations mentioned in 2.5.1 and 2.5.2 overlook the fact that the exact progress a particular driver will make at a certain time is unknown [26, p. 747ff.].
Another approach that needs to be discussed in this context is the modelling of the uncertainty of the prediction. In general the cost of an edge *e* is a random variable described by a probability density function *f(e,t)*. However, in such a formalism the complexity of calculating the stochastic cost of a route is an enormous effort and this model is not suitable for practical route planning algorithms at all [26, p. 747ff.].
The solution to this issue: The assumption is made that costs and travel times of adjacent edges are independent and turn restrictions are not included in the stochastic modelling at all. To get more detail on this approach, the reader is referred to Edelkamp et al. [26] who provide a mathematical description of this problem.

2.6. Route Guidance

Route guidance refers to providing directions to the driver. Hereby, the aim is to assist the driver in such a way that one can achieve the desirable destination problem-free. The route guidance system has to perform the following tasks [17, p. 352f.]:
First, the generation of information where the driver must intervene. In general, the instructions for a particular route are static and independent from the current position or the velocity. Hence, they are typically provided before beginning the journey and saved in a hash table.
Second, during the journey, an iterative search over all entries of the hash table is executed to ensure that all saved navigation instructions are forwarded to the driver to make sure the driver can respond appropriately. These instructions can be presented visually, acoustically, or as a combination of both.
K. Reif [17, p. 352f.] introduces four different phases of route guidance (see also Figure 2.7). In the Plain Text Announcement (PTA) - phase the user is usually only directed to keep the car on the actual road because it is too early to inform the driver about next manoeuvres. In case of main roads it can be very helpful and even necessary to inform the user about leaving the road as early as possible - because, for example, changing the lane is necessary. The instructions forwarded at this particular moment rank among the so-called Pre-Information Announcement (Pre-IA). In the phase of Information Announcement (IA) the user is usually informed about the remaining distance to the next manoeuvre. A very short time before the upcoming manoeuvre - in the phase of Activation Announcement (AA) - the driver gets the instructions to perform the manoeuvre.

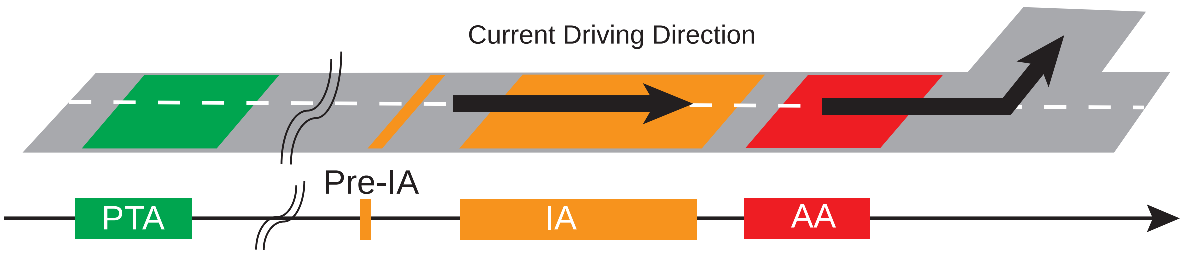


Figure 2.7.: K. Reif introduces four different phases of route guidance [17, p. 353].

2.7. User Interface

User Interface (UI) which enable interaction with a device while driving have to meet high standards. Especially as their potential to distract the driver and in this way to trigger an accident is great [28] [29, p. 72ff.].
Tasks like taking a phone call or looking up an address for the navigation system have, however, become part of everyday driver situations. In particular the navigation system hereby represents

a speciality because its usage is usually combined with the undertaking of an unfamiliar journey, and hence, requires even more attention to the traffic than usually. For that reason, it must be guaranteed that the driver is focused on the road and not on the operation of the navigation system at all [28] [29, p. 72ff.].
Therefore, when designing the UI for a navigation system, the most important questions to be asked are [29, p. 73]:

1. What kind of information is relevant for the driver and must be provided by the navigation system?
2. How should this information be presented to the driver?
3. How should the interaction between the driver and the navigation system be performed?

To answer the first question, one has to become aware of the fact that a navigation system can generally provide two basically different information types. On the one hand, something which refers to the real environment (road signs, junctions, landmarks, etc.) is presented, and, on the other hand, the driver is continuously informed about indirectly referring to or pointing at aspects of the environment (directions, distance to turn, etc.). In this context, researchers found that navigation instructions which include distinctive features of the environment ("turn right at the church") offer advantages over conventional distance instructions, such as "turn right in 300 metres" [29, p. 73].
Whoever investigates the second question must sooner or later conclude: when presenting navigation and related information, the main focus should be on not distracting the driver. This is why a lot of research in this field is dedicated to the impact of system modality (voice and/or visual) on driving/navigating performance. The most important success factor in this connection is that navigation instructions should be, above all, auditory. Even though the information should also be presented visually because the traffic situation is sometimes confusing and the voice instructions alone are not sufficient. For more detailed information on this topic, the reader is directed to J. Lumsden [29, p. 73].
The last question about driver's interaction with the navigation system cannot be answered quite that easily. There is a high tendency for designers to utilise the familiar desktop computing paradigms. This means that the very well know hardware devices (e.g. touchscreens, buttons, etc.) and the associated software approaches (e.g. use of menus, lists, scrolling etc.) are used. But there are also alternatives to such UI: speech as a largely non-visual/manual input method. Nevertheless, it should be noted that research has shown that the potential for cognitive distraction with speech interfaces is high. It was also shown that it is extremely critical to place, for example, a handwriting touchpad in the car. For more information on that, the reader is referred to Kamp et al. [30] and Burnett et al. [31]. Finally, it can be concluded that the interaction when the vehicle is in motion is a rich area for future research [29, p. 73f.].

Chapter 3.

Map-Supported Advanced Driver Assistance Systems

By now, there is a big variety of driver assistance systems which are aimed at supporting the driver in all kinds of traffic situations. For such systems, the objective is always to achieve a relaxed, stress- and accident-free driving. To give a first introduction to ADAS and to place map-supported ADAS within a larger context, Section 3.1 presents two different kinds of classification of ADAS. Use cases of such systems are outlined in Section 3.2.

3.1. Basic Classification of ADAS

In general, ADAS can be divided into the following main types:

1. **Driver information systems**, which are aimed only at informing the driver and which do not intervene directly in driving. This includes, among other things, basic information gauges, such as the speedometer, the odometer, the tachometer, the oil pressure gauge, the fuel level gauge as well as a navigation system's map display [32, p. 534ff.] [16, p. 107ff.].

2. **Driver communication systems**, which increase the road safety by transmitting warnings to the driver, for example in case of accidents and traffic jams. Among these are Wireless Local Area Network (WLAN)-supported technologies as well as the Vehicle-to-Vehicle (V2V) and Vehicle-to-Infrastructure (V2I) communcation systems [33, p. 1f.] [16, p. 107ff.].

3. **Predictive driver assistance and safety systems**, which aim at providing an allround-visibility to the driver by analysing the vehicle environment with, for example, ultrasound, short and long-range radar systems, mono and stereo camera systems, infrared and heat sensitive cameras. The goal of such systems is to identify dangerous situations and to warn the driver or even to intervene in driving [16, p. 107ff.].

4. **Vehicle stabilization systems**, whose main objective is to intervene in throttling, braking and/or steering in case of a critical situation. Typical examples are the Anti-Lock Brake

System (ABS), the Electronic Stability Program (ESP) or the Traction Control System (TCS) [16, p. 107ff.].

One can clearly recognize that ADAS are based on analysing the vehicle's environment. Regardless of whether the information comes from monitoring the vehicle (for example, as in the case of driver information systems) or from scanning the environment - the strong support is provided by using the additional data. Hence, a digital map itself can be seen as an easily accessible source of useful information and this is why map-supported ADAS have a large potential for road safety enhancement. Without any problems, the information, for example, about an upcoming type of road and the speed limitation can be provided to the driver. By extending the driver's horizon ahead of the vehicle, map-supported ADAS allow creating preventive systems and in this way increase the road safety to a large extent [34].
In this context an alternative classification of ADAS - which is more suitable for this thesis - is presented. The distinction is now made between:

1. **Non-map ADAS**, whose performance can not be improved by using digital map data. The information provided from digital maps is useless to this kind of driver assistance systems (e.g., Parking Distance Control (PDC)) [35].
2. **Map-enhanced ADAS**, whose functionality and reliability can be improved by using digital map data. It should be noted, however, that map-enhanced ADAS work also without the additional information gained out of digital maps. Adaptive Cruise Control (ACC), Intelligent Speed Advisory (ISA), Adaptive Frontlighting System (AFS) or Lane Keeping Assistant (LKA) are just a few examples [35].
3. **Map-enabled ADAS**, whose implementation is not possible without digital maps. The Curve Speed Warning System (CSWS) and the Dynamic Pass Predictor (DPP) are good examples for this kind of assistance systems [36, p. 66ff.] [35].

Durekovic et al. introduce the term of map-supported ADAS which refers to map-enabled ADAS and map-enhanced ADAS (see Figure 3.1) [35]. In keeping with this definition, the term is also used in this thesis. It should be noted at this point that the development of map-supported ADAS is a real challenge. In this context, attention must be drawn to the limited accuracy of digital maps. All ADAS depend on exact and reliable map databases. In truth, by now, some data (e.g., the road curvature information) is not available at all or not precisely stored in the maps [34].

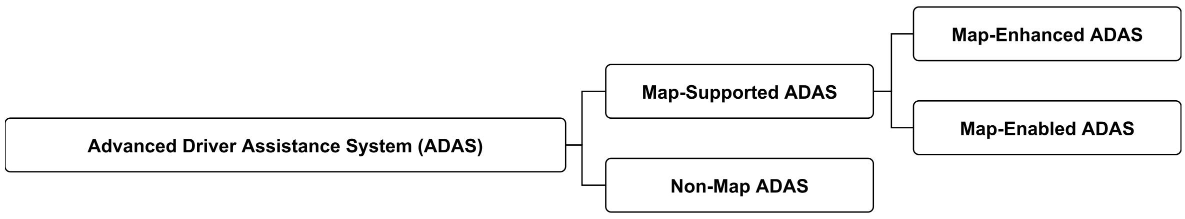


Figure 3.1.: Durekovic et al. introduce the term of map-supported ADAS which refers to map-enabled ADAS and map-enhanced ADAS [35].

3.2. Examples for Map-Supported ADAS

Table 3.1 illustrates some map-supported ADAS[1]. Furthermore, it provides a short description of the individual ADAS and a note how the enhancement with the help of a digital map can be achieved.

3.2.1. Intelligent Speed Advisory (ISA)

Intelligent Speed Advisory (ISA) refers to ADAS which support drivers to adhere to the speed limit. A prerequisite for ISA is that the vehicle's location and heading are known. Besides, the database must provide the actual speed limits. This information is then used to compare the vehicle's speed with the known speed limit of the location. ISA can work as a warning system and just inform the driver about overspeeding (advisory, supportive ISA) or even automatically limit the maximum speed (intervening ISA) [37, p. 407] [38, p. 4].
This technology has been well researched internationally, for example in Denmark [39], the Netherlands [40], France [41], Australia [42], Belgium [43], Malaysia [44], the UK[2] [45] and many other countries.
The safety effects which in this context have been published are very impressive: 18% reduction in fatal accidents were achieved in the UK with advisory ISA, 37% with intervening ISA. With supportive ISA, up to 50% of traffic deaths could be avoided in EU countries [38, p. 4].

3.2.2. Curve Speed Warning System (CSWS)

A Curve Speed Warning System (CSWS) is designed to prevent drivers from entering a curve at a speed faster than the speed allowed at the upcoming part of the route. Glaser et al., for example, present a possible implementation for such a system. With the help of the knowledge about the geometric characteristics of the upcoming road from a digital map, a safe speed according to the

[1]Examples of map-supported ADAS, http://sampledata.navteq.com/site/global/products_licensing/navteq_products/adas/20_applications/p_adas_applications.jsp, accessed December 13, 2012
[2]External vehicle speed control, http://www.its.leeds.ac.uk/projects/evsc/, accessed October 29, 2012

Table 3.1.: Examples of map-supported ADAS.

ADAS Application	**Basic Function**	**Digital Map Enhancement**
Adaptive Cruise Control (ACC)	Maintain set following distance behind a lead vehicle.	Map data enables more natural behavior.
Adaptive Frontlighting System (AFS)	Swiveling and leveling headlamps.	Map curvature and slope data enables predictive ability.
Blind Spot Detection (BSD)	Identify difficult to see objects near vehicle.	Lane information improves threat discrimination.
Curve Speed Warning System (CSWS)	Alert driver to a sharp curve ahead or safe approach speed.	Road curvature data enables predictive ability.
Drowsy Driver Detection (DDD)	Warn driver when lethargic, intoxicated or otherwise impaired.	Road curvature data enables predictive ability.
Electronic Stability Control (ESC)	Stabilize vehicle during severe understeer or oversteer.	Map data improves intended path determination.
Forward/Side Collision Warning (CW)	Identify and respond to vehicles which pose a threat.	Road trajectory and lane information improve threat discrimination.
Green Driving (GD)	Enhance routing for reduced fuel consumption.	Road slope, traffic sign and signal location enable predictive energy used.
Lane Departure Warning (LDW)	Alert driver when vehicle exceeds the lane.	Road shape data improves performance of imaging system.
Lane Keeping Assistant (LKA)	Maintain vehicle within driving lane.	Road shape data improves performance of imaging system.
Lane Change Assistant (LCA)	Warn of vehicles in proximity during lane change.	Lane information improves threat discrimination.
Overtake Assistant (OA)	Identify locations for safe passing.	Digital map includes dedicated overtaking zone locations.
Powertrain Efficiency (PE)	Reduced fuel consumption and emissions.	Road slope data enables predictive powertrain functions.
Intelligent Speed Advisory (ISA)	Inform driver of speed limit or excessive speed.	Map provides legal and advisory speed limits.
Stoplight and Stop Sign Warning (SSSW)	Alert driver to approaching traffic light or stop sign.	Map provides traffic signal and sign locations.

specific road section is computed. Whenever the driver exceeds this critical speed, a warning is emitted [46]. Nevertheless, improvements could already be made by providing the needed speed limitations as part of the digital map.

3.2.3. Adaptive Human Machine Interface (Adaptive HMI)

One of the currently most popular architecture for human-machine interfaces is the so-called adaptive Human Machine Interface (HMI). Its task is to assist the user in such a way that the most salient information has to be provided in the most appropriate form, and at the most opportune time. To achieve this, the organization and presentation of information have to be optimized [21, p. 536f.]. In this context, the following points should be considered:

1. Highlight always the most relevant information. The decision what piece of information is important depends on the information's relevance according to the desirable goal at the particular moment [21, p. 536f.].
2. Organize the display space in such a way that, if necessary, new constraints and parameters can be added problem-free [21, p. 536f.].
3. Select the right representation for the information, that means according, for example, to the resources: use an appropriate media for communication with the user [21, p. 536f.].
4. Special care should be always taken to display the required information timely [21, p. 536f.].
5. Furthermore, one must consider that such an adaptive HMI should always be a collaborator and not a competitor or troublemaker. The driver should not be distracted from the primary task of driving [47] [21, p. 536f.].

Some aspects of optimizing the adaptation of HMI can also be found in [47], [48] or [49].

Chapter 4.

Android Navigation and Routing Software Supporting OpenStreetMap

Thanks to the British programmer Steve Coast, in August 2004 the OpenStreetMap (OSM) project was born. Motivated by the idea of creating a free map of the world, which allows users to view, edit and use geographical data without breaking any licensing conditions or paying a lot of money, numerous volunteers have joined forces and collaboratively started to map features - such as parks, roads and buildings [50, p. 12f.] [51, p. 4]. The result is remarkable: OSM data become popular in such a fast way that, right now, it is used in a large number of applications[1].
In the context of this work, some popular open source projects which support OpenStreetMap are analyzed for potential use in enhancing *DriveAssist* capabilities. For this reason, this chapter reviews the advantages and disadvantages of the evaluated projects. The most important features, such as routing algorithms, route calculation, guidance etc. are compared (more details can be found in Section 4.1). With the help of the selected criteria three different projects are shortlisted: the open source project Gosmore is introduced in Section 4.2, Section 4.3 describes some components of Navit and OsmAnd is presented in Section 4.4. The reasons for the final decision on which project really suits the best are presented in Section 4.5 together with a conclusive comparison of the analyzed projects.

4.1. Requirements

Before proceeding with the implementation of *DriveAssist 2.0*, the question arose how to upgrade *DriveAssist* with as many navigation and routing functionalities as possible in a short time. The answer is actually easy: Considering the very large amount of open source projects on this field, it seemed reasonable to merge *DriveAssist* with an existing solution. Particularly, when one looks at the already implemented features of such open source applications: almost all important navigation

[1]Android software supporting OpenStreetMap, http://wiki.openstreetmap.org/wiki/Android, accessed November 14, 2012

functionalities are available in various versions.
The final selection of the appropriate project is based on the assessment of the requirements listed in Table 4.1. Thirteen different criteria are used, each is of a different importance.
During the search for the most suitable open source project, the properties associated with "+" in Table 4.1 play the least important role. The requirements with five "+" are the most important. For example, the map provider needs to be the publicly available OpenStreetMap, so this criteria is one of the most important. It is also clear that the wanted project needs to be complete, easily expandable, fully operational and its source code freely available. Otherwise it would be not possible to enhance *DriveAssist* within the given time. This is why the criteria completeness, expandability, operating ability and licence are classified as very important.
A prototypic implementation of the map handling, the destination entry, the route calculation, the route guidance and the voice output are adopted as sufficient at this development level, because the mere fact that such functionalities are provided, facilitates further development.
As the GUI can be easily adapted, a stylish design as well as add-on features are not considered sufficiently significant. The degree of how up to date a project is, is beneficial but not a mandatory criterion.
In the following, three open source projects - Gosmore, Navit, and OsmAnd - are introduced and analyzed in detail. Figure 4.1 presents the main icons for the appropriate Android applications.

Table 4.1.: Evaluation criteria and their importance for the selection of the appropriate open source project.

Evaluation Criteria	**Assessment**
License	+++++
Recency of Development	+++
GUI	+
Destination entry	++
Route calculation	+++
Route guidance	+++
Voice output	+++
Map provider	+++++
Map presentation	++++
Completeness	+++++
Operating ability	+++++
Expandability	+++++
Add-on features	+

4.2. Gosmore

The latest version of Gosmore's source code is freely available[2]. Results of the evaluation can be found in Table 4.2 below. The annotations to the criteria license, recency of development and map provider are based on web sites[3] [4]. All other properties are tested with a precompiled[5] Gosmore's .apk, because the source code could not provide a fully operational application.

Table 4.2.: Assessment of the open source project Gosmore according to the criteria which are explained in more detail in Section 4.1.

Evaluation Criteria	**Assessment**
License	Some files of the project are licensed under the GNU GPL v2, but the major files are placed under the BSD License.
Recency of Development	Gosmore's source code is out of date. The last changes have been realised two years ago.
GUI	Very basic.
Destination entry	Could not be tested. Functionality is unavailable.
Route calculation	Could not be tested. Functionality is unavailable.
Route guidance	Could not be tested. Functionality is unavailable.
Voice output	Could not be tested. Functionality is unavailable.
Map provider	OpenStreetMap
Map presentation	Could not be tested. A network error always occurs. So it is not possible to get access to map data.
Completeness	Some files are corrupted or even missing. With the help of the open source code it is not possible to run Gosmore.
Operating ability	The source code of Gosmore can be compiled. But there are some problems with the .apk. On the one hand, it is only able to run the application on Android 1.6 and lower. On the other hand, it crashes after installation even on Android 1.6 or lower.
Expandability	None.
Add-on features	None.

[2]Gosmore's source code, http://svn.openstreetmap.org/applications/rendering/gosmore/, accessed November 14, 2012

[3]General information about Gosmore, https://github.com/openstreetmap/gosmore, accessed November 14, 2012

[4]General information about Gosmore, http://wiki.openstreetmap.org/wiki/Gosmore, accessed November 14, 2012

[5]Gosmore's .apk, https://play.google.com/store/apps/details?id=org.osmu.gosmore&feature=search_result, accessed November 14, 2012

4.3. Navit

Navit's source code can be accessed through a SVN-Server[6]. Even though the Android project is complete and operational, it has a lot of disadvantages. All functionalities[7] are at a very initial development state. For a more detailed overview, the reader is referred to Table 4.3.

Table 4.3.: Assessment of the open source project Navit according to the criteria which are explained in more detail in Section 4.1.

Evaluation Criteria	**Assessment**
License	GNU GPL v2[8]
Recency of Development	Navit's source code is out of date.
GUI	Very basic.
Destination entry	The serch for a location is possible. The problem: the addresses can not be found.
Route calculation	Could not be tested. Functionality is unavailable.
Route guidance	Could not be tested. Functionality is unavailable.
Voice output	Could not be tested. Functionality is unavailable.
Map provider	OpenStreetMap
Map presentation	Very basic.
Completeness	The compilation is possible.
Operating ability	The application does not crash.
Expandability	The source project can be enhanced, but a lot of work would be required.
Add-on features	None.

[6]Navit's source code, `https://navit.svn.sourceforge.net/svnroot/navit/trunk/navit`, accessed November 14, 2012

[7]General information about Navit, `http://wiki.navit-project.org/index.php/Navit_on_Android`, accessed November 14, 2012

4.4. OsmAnd - Open Street Map Automated Navigation Directions

This project has a high potential to become very popular, because it provides many helpful and reliable routing and navigation functionalities. There is even a commercial version of OsmAnd[9], the so-called OsmAnd+.
More detailed information about this open source project can be found in Table 4.4 below.

Table 4.4.: Assessment of the open source project OsmAnd according to the criteria which are explained in more detail in Section 4.1.

Evaluation Criteria	**Assessment**
License	GNU GPL v3[10]
Recency of Development	OsmAnd is supported by a huge and active community of open source developers.
GUI	The UI is very well done and intuitive.
Destination entry	OsmAnd provides search by address, latitude/longitude coordinates, recently searched or favourite locations as well as a search for nearby **POIs!** and public transports.
Route calculation	In this field, OsmAnd has a lot to offer: one can use online (CloudMade, OpenRouteService, Yet another OpenStreetMap Route Service (YOURS)) or offline (A* algorithm) routing calculation.
Route guidance	Essential functionalities are already implemented. The navigation instructions are not only displayed in a very attractive way, but also verbalized in an easily understandable style.
Voice output	All navigation instructions are verbalized.
Map provider	OpenStreetMap
Map presentation	The map can be dragged, zoomed, rotated, etc.
Completeness	OsmAnd has no missing or corrupted files. After downloading the project, it can be compiled problem-free. Without having to make any changes, the project is fully operational and ready to be further developed.
Operating ability	No special difficulties are known. The application is crash-free.
Expandability	OsmAnd is designed in a very modular and easily extensible way. There is even a possibility to develop plugins, one can follow the example of the provided parking plugin.
Add-on features	There are a lot: Maps and TTS packages can be downloaded, locations can be shared, the map can be presented in a sunrise/sunset or night mode - these are only a few examples.

[9] OsmAnd's source code, `http://osmand.net/`, accessed December 15, 2012

4.5. Assessment Summary

In conclusion, it is very obvious, why OsmAnd is the most promising project in all respects. It is complete and fully operational. It provides all desirable functionalities and even more, including e.g., a download or a plugin manager.
Table 4.5 provides a summary of all evaluation criteria in comparison. The green marked cells signify that the project definitely fulfils the specific criterion. The yellow ones mean that a very prototypic implementation of the desirable functionality is available, but a lot of further work is necessary. The red cells indicate properties which do not satisfy the requirements established in this thesis.

Table 4.5.: Comparison of Gosmore, Navit and OsmAnd according to the criteria which are explained in more detail in Section 4.1.

Feature	**Gosmore**	**Navit**	**OsmAnd**
License	🟩	🟩	🟩
Recency of Development	🟥	🟥	🟩
GUI	🟥	🟨	🟩
Destination entry	🟥	🟨	🟩
Route calculation	🟥	🟨	🟩
Route guidance	🟥	🟨	🟩
Voice output	🟥	🟩	🟩
Map provider	🟩	🟩	🟩
Map presentation	🟥	🟥	🟩
Completeness	🟥	🟥	🟩
Operating ability	🟥	🟥	🟩
Expandability	🟥	🟥	🟩
Add-on features	🟥	🟥	🟩

Figure 4.1.: Icons of the three chosen open source projects for the evaluation. (a) represents Gosmore, (b) represents Navit, and (c) represents OsmAnd.

Chapter 5.

Implementation

This chapter presents the development environment for the Android application *DriveAssist 2.0* (see Section 5.1). With the goal to clearly define the boundaries between the already existing and the desirable features and improvements of the new application, Section 5.2 explains the initial state. The application requirements which were specified at the beginning of the development process are presented in Section 5.3. The final part of this chapter, Section 5.4, gives an overview over the most important new features of the application.

5.1. Development Environment

The driver assistance system *DriveAssist 2.0*, introduced in this thesis, is developed as an Android application. For this purpose, the following software tools were used to create a functioning development environment:

1. **The Java Development Kit (JDK)**[1], which provides a lot of libraries that easily can be used for development of Android applications [52, p. 19].
2. **The Android Software Development Kit (SDK)**[2], which makes everything available that is required for the start of creating, testing, and debugging applications for the Android platform, including several development tools, the Android virtual device manager and sample code [52, p. 14f.].
3. **Eclipse**[3], which serves as an integrated development environment [52, p. 19].
4. **The Android Development Tools (ADT)**[4], which provides a powerful, integrated environment for developing Android applications, including, for example, a simple Anroid project

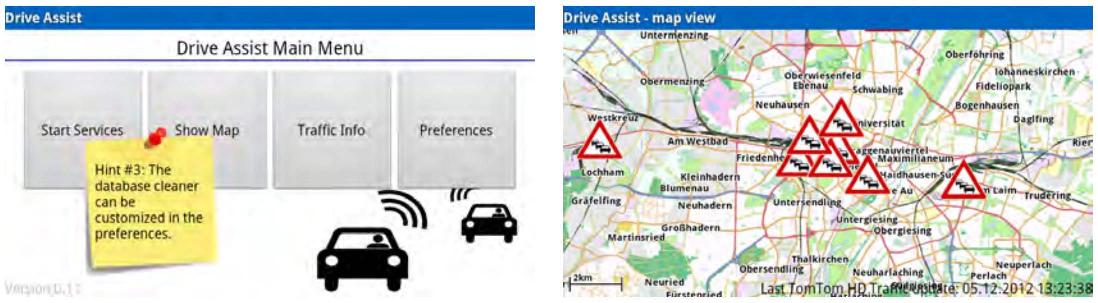
[1]Java SDK, http://www.oracle.com/technetwork/java/javase/downloads/index.html, accessed December 17, 2012
[2]Android SDK, http://developer.android.com/sdk/index.html, accessed December 17, 2012
[3]Eclipse, http://www.eclipse.org/downloads/, accessed December 17, 2012
[4]Notes on installing the ADT Eclipse Plugin, http://developer.android.com/sdk/installing/installing-adt.html, accessed December 17, 2012

creation and several debugging tools.

Besides, it should be noted, that in the context of this work, the application was tested regularly. The testing device *HTC Desire* uses the Android operating system with the version 2.3.7.

5.2. Initial State

The previous version of *DriveAssist 2.0* (see Figure 5.1(a)) was already able to receive (for this feature the device requires WiFi access) and analyse messages originated from V2X communication. Besides, it provided the possibility to get traffic information from CTS.
Both sources could be used to visualize the actual traffic situation on the map view of the application (see Figure 5.1(b)). Furthermore, when the smartphone was used for a different application, the user still could be sure that *DriveAssist* would generate a warning screen if appropriate and inform about changes in the actual traffic situation in surrounding areas.
Another already implemented feature was the location service which could be used to localize the user and display the current position on the map.


Drive Assist
Drive Assist Main Menu
Start Services
Show Map
Traffic Info
Preferences
Hint #3: The database cleaner can be customized in the preferences.
Drive Assist - map view
Last TomTom HD Traffic update: 05.12.2012 13:23:38

(a) (b)

Figure 5.1.: (a) The main menu of *DriveAssist*, (b) the visualization of traffic informations on the top of the map.

5.3. Application Requirements

The further development of the purely informative application *DriveAssist* is aimed at adding navigational functionalities. This means, in detail, that a destination entry, a route calculation and a route guidance module needs to be implemented.
The focus in the implementation of the destination entry is, above all, an offline search for locations. In addition to this possibility, offline routing needs to be provided. In the course of the enhancements, the UI also have to be adapted to the new capabilities.

5.4. Drive Assist 2.0

The enhanced version *DriveAssist 2.0* provides a lot of new features. Figure 5.2 gives an overview of the currently implemented application.

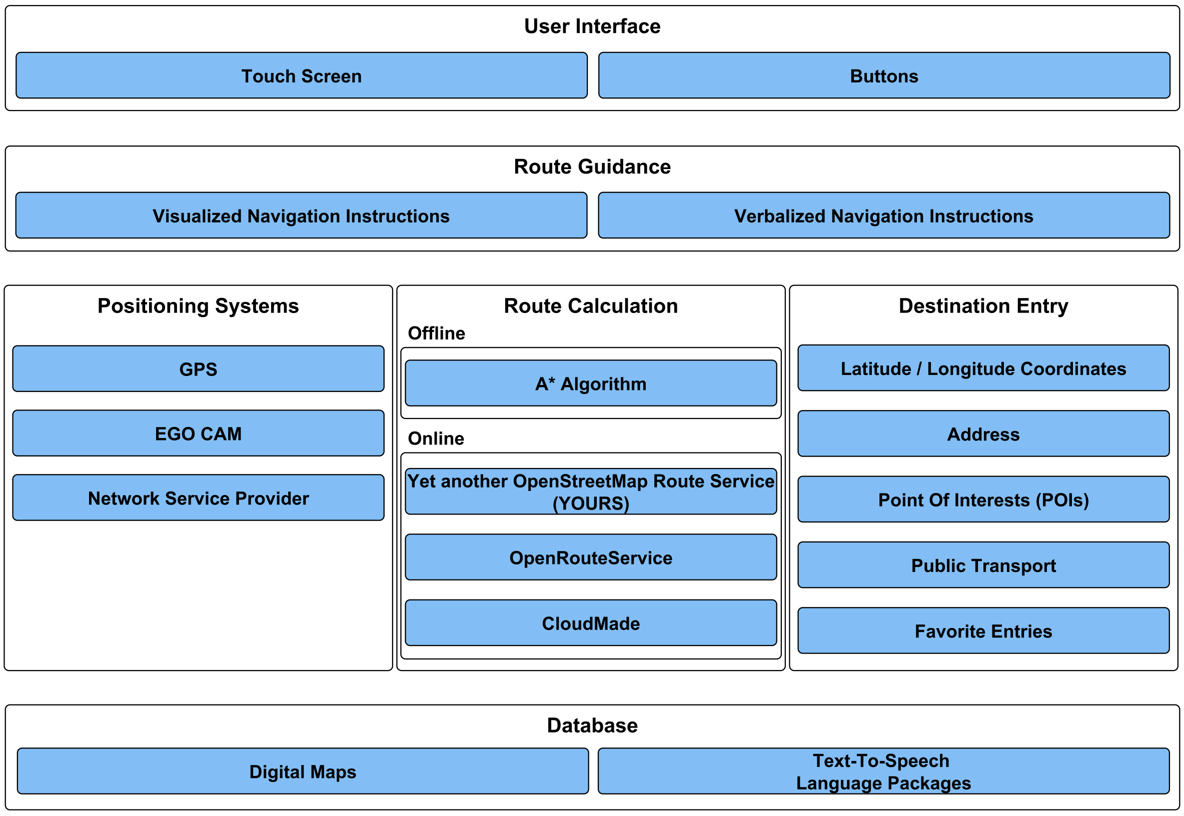
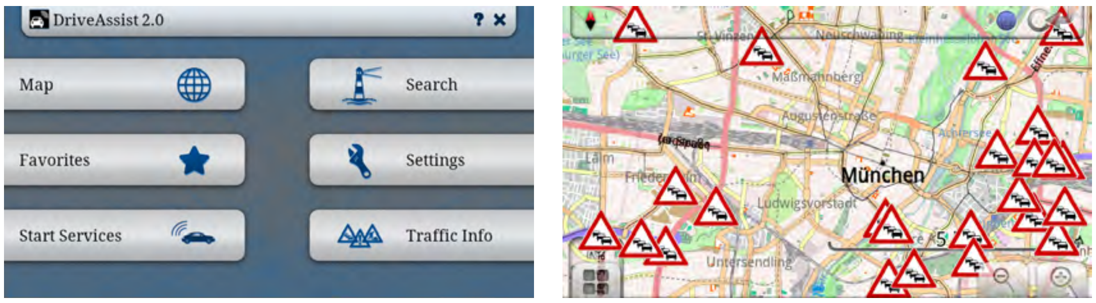


Figure 5.2.: Overview of *DriveAssist 2.0* functionalities.

The positioning systems are used from the previous application *DriveAssist*. The rest of the illustrated modules in Figure 5.2 are adapted from the open source project *OsmAnd*. During the merging process of the two applications, all map overlays originated from *DriveAssist* needed to be modified to ensure that the visualization of the traffic information is still possible (see Figure 5.3(b)).
Figure 5.3(a) shows the main menu of *DriveAssist 2.0* which is extended by two buttons. The *Favorites* and the *Search* button. As the name suggests, the first one provides a search for locations, the second one provides access to a list of favorite locations.

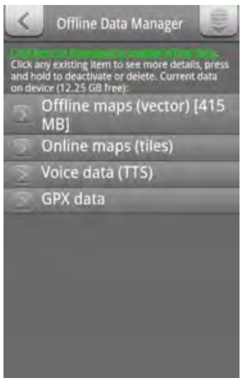


(a) (b)

Figure 5.3.: (a) The main menu of *DriveAssist 2.0*, (b) the visualization of traffic informations on the top of the map.

5.4.1. Database

DriveAssist 2.0 uses offline and online map data. The used offline map data files have the file extension **.obf*[5]. After getting raw OpenStreetMap data from an external source[6] the *OsmAndMapCreator*[7] must be used to generate a **.obf* - file. In general, the generation of *.obf-files is not necessary at all, because all maps are updated continuously and can be directly downloaded with the help of the application (see Figure 5.4). The access to TTS language packages happens also via the application *DriveAssist 2.0* (see Figure 5.4). Another possibility to get access to data is to download the digital maps and TTS packages directly from the website[8].



Click here to download or update offline data.
Click any existing item to see more details, press and hold to deactivate or delete. Current data on device (12.25 GB free):
Offline maps (vector) [415 MB]
Online maps (tiles)
Voice data (TTS)
GPX data

Figure 5.4.: The data manager of *DriveAssist 2.0* which provides access to the download area. With its help digital maps as well as TTS language packages can be downloaded.

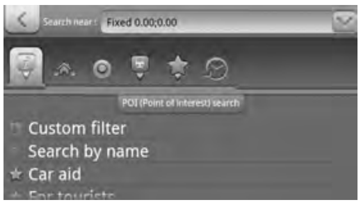
[5]How to prepare data for offline use, `http://code.google.com/p/osmand/wiki/HowToArticles#How_To_Prepare_own_data_to_use_offline`, accessed October 25, 2012
[6]Examples for external sources to get OpenStreetMap data: GEOFABRIK (`http://www.geofabrik.de/`), CloudMade (`http://downloads.cloudmade.com/`), accessed October 25, 2012
[7]OsmAndMapCreator, `http://download.osmand.net/latest-night-build`, accessed October 25, 2012
[8]TTS Language Packages Download, `http://code.google.com/p/osmand/downloads/list`, accessed October 25, 2012

5.4.2. Positioning Systems

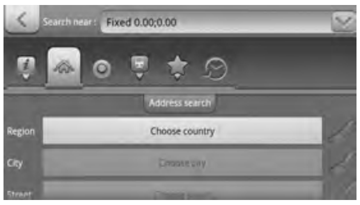
The determination of the user's location can be performed in three different ways: Firstly, the internal GPS receiver of the mobile phone can be used. Secondly, the location can be determined by Wi-Fi and finally the user can choose to determine the position via EGO CAM. It should be noted that the last opportunity is not a conventional positioning system. It is implemented to enable receiving location updates via the so-called EGO Cooperative Awareness Messages (CAM).

5.4.3. Destination Entry

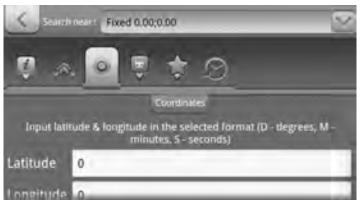
The user of *DriveAssist 2.0* can search a location not only by address (see Figure 5.5(b)), but also by latitude and longitude coordinates (see Figure 5.5(c)). Finding POI (see Figure 5.5(a)) or get public transport information (see Figure 5.5(d)) - nearby to a specific location - is also possible. Besides, the flexible module of favorite search (see Figure 5.5(e)) offers the possibility to save often used locations of the map. Furthermore a history search is provided (see Figure 5.5(f)).



(a)



(b)

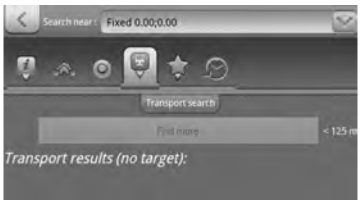


Input latitude & longitude in the selected format (D - degrees, M - minutes, S - seconds)

Latitude 0

Longitude 0

(c)



Transport results (no target):

(d)

(e)

(f)

Figure 5.5.: The user of *DriveAssist 2.0* can search for an address using six different kinds of destination entry: (a) POI search, (b) address search, (c) latitude/longitude search, (d) public tranport search, (e) favorite search, (f) history search.

5.4.4. Route Calculation

The route calculation of *DriveAssist 2.0* can be carried out either in online or in offline mode. When using online routing, services from CloudMade[9], YOURS[10] or OpenRouteService[11] are available. If the user chooses the offline routing, the A* algorithm is used to provide a route.

5.4.5. Route Guidance

Figure 5.6 shows a typical map view while navigation. In the upper left corner of the screen the navigational instructions as well as the remaining distance to the next waypoint on the route are visualized. The destination is always marked with a flag.
While guiding the driver, the map is rotated so that "up" on the map always corresponds to the forward direction of the vehicle. The arrival time and the remaining distance to the final destination are displayd in the upper right corner. Moreover, verbalized navigational instructions assist the driver while moving.

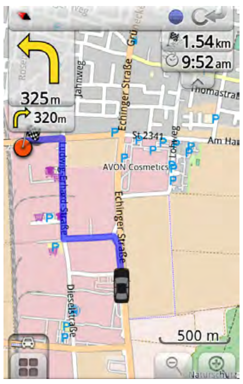
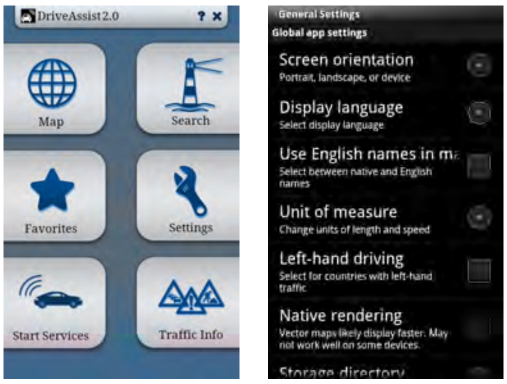


Figure 5.6.: Typical map view while navigation.

[9]CloudMade, http://maps.cloudmade.com/, accessed October 24, 2012
[10]Yet another OpenStreetMap Route Service (YOURS), http://openrouteservice.org/, accessed October 24, 2012
[11]OpenRouteService, http://yournavigation.org/, accessed October 24, 2012

5.4.6. User Interface

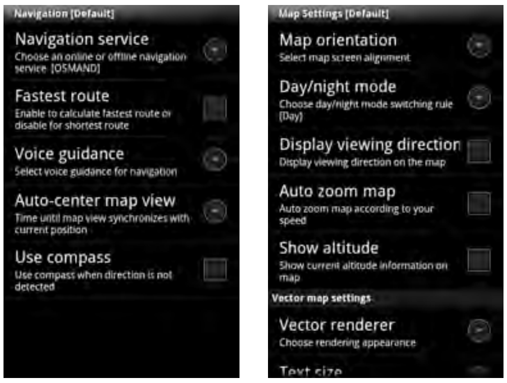
Some examples of the new implemented UI of the application can be found in Figure 5.7 through Figure 5.8.



General Settings
Global app settings
Screen orientation
Portrait, landscape, or device
Display language
Select display language
Use English names in m
Select between native and English names
Unit of measure
Change units of length and speed
Left-hand driving
Select for countries with left-hand traffic
Native rendering
Vector maps likely display faster. May not work well on some devices.
Storage directory

(a) (b)

Figure 5.7.: (a) Main menu of *DriveAssist 2.0*, if used in portrait mode, (b) the presented view if the user selects the *Settings* button.



Navigation [Default]
Navigation service
Choose an online or offline navigation service [OSMAND]
Fastest route
Enable to calculate fastest route or disable for shortest route
Voice guidance
Select voice guidance for navigation
Auto-center map view
Time until map view synchronizes with current position
Use compass
Use compass when direction is not detected

Map Settings [Default]
Map orientation
Select map screen alignment
Day/night mode
Choose day/night mode switching rule [Day]
Display viewing direction
Display viewing direction on the map
Auto zoom map
Auto zoom map according to your speed
Show altitude
Show current altitude information on map
Vector map settings
Vector renderer
Choose rendering appearance
Text size

(a) (b)

Figure 5.8.: (a) Different navigation settings, (b) several provided map settings.

Chapter 6.

Evaluation

This chapter describes how the proposed software and its key functionalities are evaluated in terms of applicability in practical use. The most important aspects of practical street navigation are tested for coherence to the specified requirements described in Section 1.3. To that end, *Google Maps* and *Route 66 Maps + Navigation*, two established Android applications for routing and navigation, serve as reference in a qualitative comparison. The evaluation conducted in this thesis focuses on the three aspects, interaction complexity for route search, accuracy of route calculation and presentation of navigational instructions. Section 6.1 is dedicated to the destination entry. This means, the feature to search addresses offline is tested as well as the whole procedure which is needed to be done in order to start a route calculation. Section 6.2 is dedicated to the reliability of the route calculation itself. It offers verification that a reliable and clear illustration of the calculated route is accessible for the user. Section 6.3 provides documentation of the visual and verbalized navigational instructions and finally, Section 6.4 summarizes the most important results of the evaluation.

6.1. Interaction Complexity for a Route Search

The procedure *Google Maps* and *Route 66 Maps + Navigation* use in designing the destination entry is very similar. Both applications offer in the main menu (to access this menu the user has to press the *Menu* button) of the map view the sub-menu called *Directions* which is used to set the destination. Thereafter, the user needs to start the routing guidance module (see Figure 6.1). The main difference in the destination entry of *DriveAssist 2.0* lies in the fact, that the user first needs to search for a location and set the searched location as the destination. After this is done, the user is taken to the map view environment and can access the sub-menu Directions, in the same way as with the two reference applications, by pressing the *Menu* button (see Figure 6.1). Obviously, the interface architecture used in *Google Maps* and *Route 66 Maps + Navigation* requires one less interaction step from the user. At the same time, it is questionable which of the two interaction concepts is more intuitive to a majority of users. Such can only be revealed in an

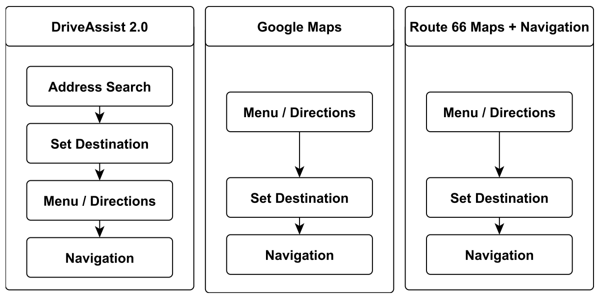


Figure 6.1.: Depicted are the procedural details of the destination entry of three different applications: *DriveAssist 2.0*, *Google Maps* and *Route 66 Maps + Navigation*.

experimental interaction user study which goes beyond the scope of this thesis.

6.2. Accuracy of Routing Calculation

In general, it is quite challenging to quantitatively evaluate route calculation. On the one hand, this is due to the fact that there is no distinct metric for a best route, i.e. the shortest route does not always guarantee the shortest travel time. However, as was described earlier, many sophisticated route estimation algorithms incorporate additional parameters like road conditions to account for this lack of distinction. On the other hand, even if considering for example the route length as a metric, route calculation performance always relies on the accuracy of available map data and producing ground truth data is extremely difficult.
For the goal of this thesis, an evaluation of practical use of *DriveAssist 2.0* as a navigational software suffices. To that end, *Google Maps* as the de-facto standard in publicly available applications was used as a reference system to assess the offline route calculation and presentation which *DriveAssist 2.0* provides. Utilizing three different perimeters with radii of 1km, 5km and 10km and the address *Arcisstraße 21* as the center, which is illustrated in Figure 6.2, thirty different destinations are determined. Using *Google Maps Browser Service*[1], the route length is measured for each route and the routes are assigned to one of the three distance groups of 1km, 5km and 10km, such that for each distance, ten different routes are available for further comparison.
Table 6.1 lists the end points which belong to the routes of 1km distance length, Table 6.2 refers to the 5km distance and Table 6.3 to the 10 km distance respectively. Comparing *DriveAssist 2.0* with *Route 66 Maps + Navigation*, one can clearly recognize that the routes provided by *DriveAssist 2.0* are very close to the *Google Maps* reference. The numerical evaluation of route length differences is visualized in Figures 6.3 through 6.4.

[1]Google Maps browser service, `https://maps.google.de/`, accessed December 10, 2012

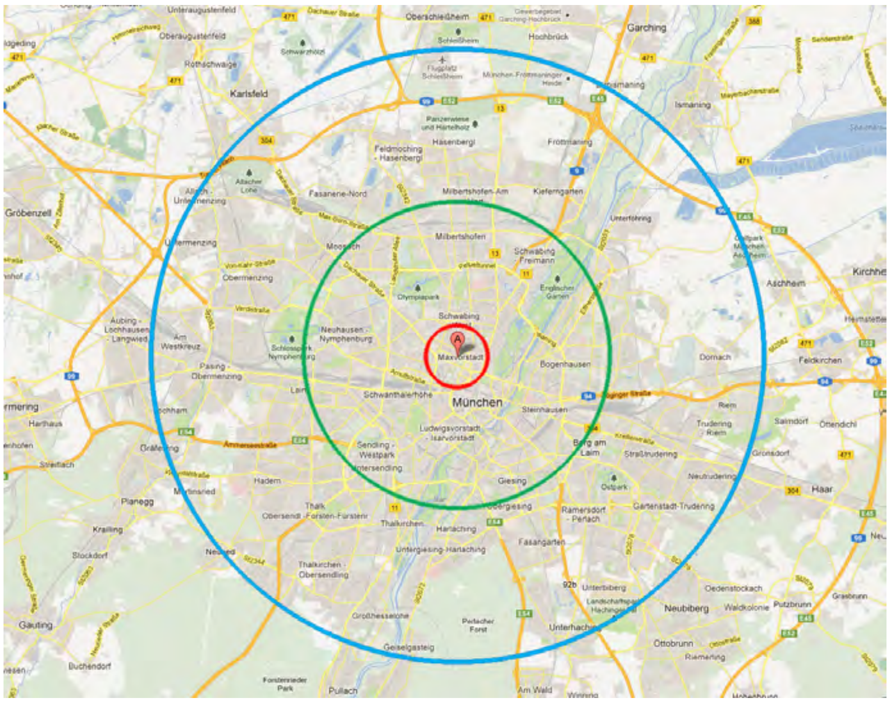


Figure 6.2.: Three perimeters of radii 1km (*red*), 5km (*green*) and 10km (*blue*) centered at the address *Arcisstraße 21 (Munich)* are used to empirically find ten suitable destinations for each route length to assess route calculation.

Table 6.1.: Ten end points which lead to an approximate route length of 1km. Listed are the proposed route lengths provided by the three different applications (*DriveAssist 2.0*, *Google Maps* and *Route 66 Maps + Navigation*). The visualization of the routes is illustrated in Figure A.1 to Figure A.10.

Route ID No.	**Destination**	$l_{\text{DriveAssist 2.0}}$ **[m]**	$l_{\text{Google Maps}}$ **[m]**	$l_{\text{Route 66 Maps + Navigation}}$ **[m]**
1	Görrestraße 22	1490	1000	1000
2	Amalienstraße 33	952	907	1200
3	Türkenstraße 38	855	932	800
4	Oskar-von-Miller-Ring 1	865	963	800
5	Luisenstraße 25	907	955	800
6	Brienner Straße 47	1290	926	800
7	Zentnerstraße 2	1430	1150	1200
8	Max-Joseph-Straße 2	895	983	800
9	Karlstraße 35	940	955	850
10	Sophienstraße 28	1040	1030	850

Table 6.2.: Ten end points which lead to an approximate route length of 5km. Listed are the proposed route lengths provided by the three different applications (*DriveAssist 2.0*, *Google Maps* and *Route 66 Maps + Navigation*). The visualization of the routes is illustrated in Figure A.11 to Figure A.20.

Route ID No.	**Destination**	$l_{\textit{DriveAssist 2.0}}$ **[m]**	$l_{\textit{Google Maps}}$ **[m]**	$l_{\textit{Route 66 Maps + Navigation}}$ **[m]**
1	Moosacher Straße 17	5120	7080	5000
2	Ungererstraße 160	4820	5500	5200
3	Oberföhringer Straße 58	4850	5190	4800
4	Denninger Straße 96	4840	5880	4800
5	Orleansstraße 50	4820	4890	4800
6	Albert-Roßhaupter-Straße 2	5680	4950	4800
7	Landsberger Straße 212	4850	5700	4800
8	Hanauer Straße 46	7110	5100	5200
9	Südliche Auffahrtsallee 72	5620	4980	4900
10	Milbertshofener Straße 54	4250	4840	5200

Table 6.3.: Ten end points which lead to an approximate route length of 10km. Listed are the proposed route lengths provided by the three different applications (*DriveAssist 2.0*, *Google Maps* and *Route 66 Maps + Navigation*). The visualization of the routes is illustrated in Figure A.21 to Figure A.30.

Route ID No.	**Destination**	$l_{\textit{DriveAssist 2.0}}$ **[m]**	$l_{\textit{Google Maps}}$ **[m]**	$l_{\textit{Route 66 Maps + Navigation}}$ **[m]**
1	Verdistraße 142	9900	10380	10100
2	Gräfelfinger Straße 133a	10600	10730	9800
3	Neufeldstraße 20	11700	10800	9900
4	Solalindenstraße 52	11100	10160	9800
5	Cincinnatistraße 17	10500	7980	10000
6	Drygalski-Allee 25	11200	10720	10000
7	Lortzingstraße 26	10900	10740	9800
8	Intersection Flensburger Straße / Apenrader Straße	9640	11310	10000
9	Intersection Feldmochinger Straße / Hammerschmiedstraße	11600	11130	10000
10	Intersection Paul-Wassermann-Straße / Werner-Eckert-Straße	11000	10130	9800

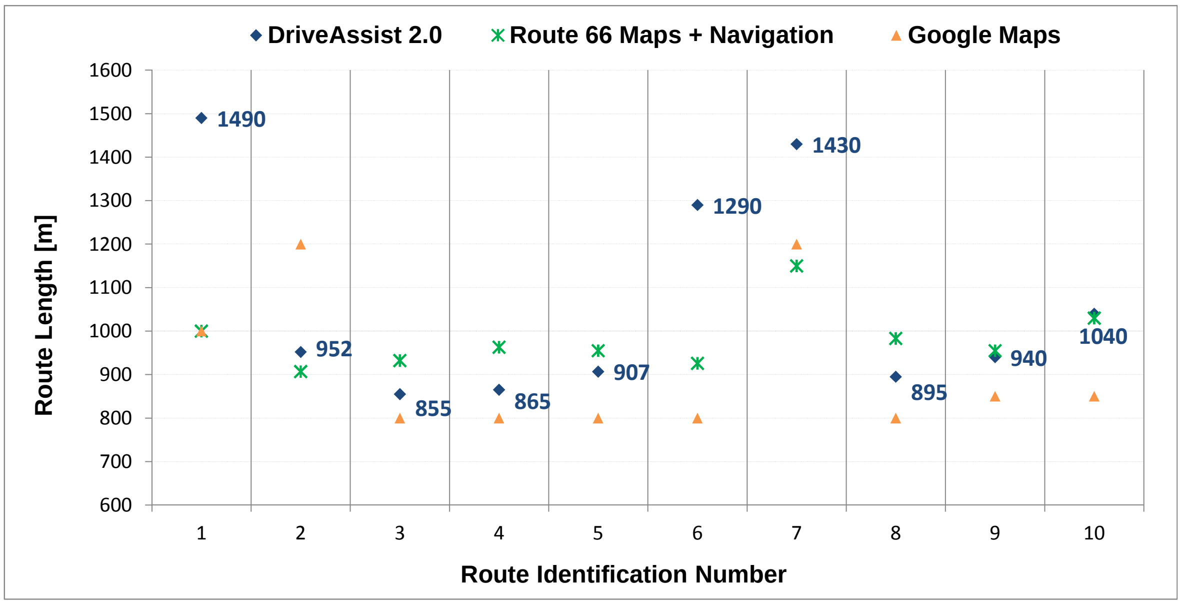


Figure 6.3.: Plot of the calculated route lengths in distance group of 1km. The route identification numbers refer to Table 6.1. Depicted are the proposed route lengths provided by three different applications.

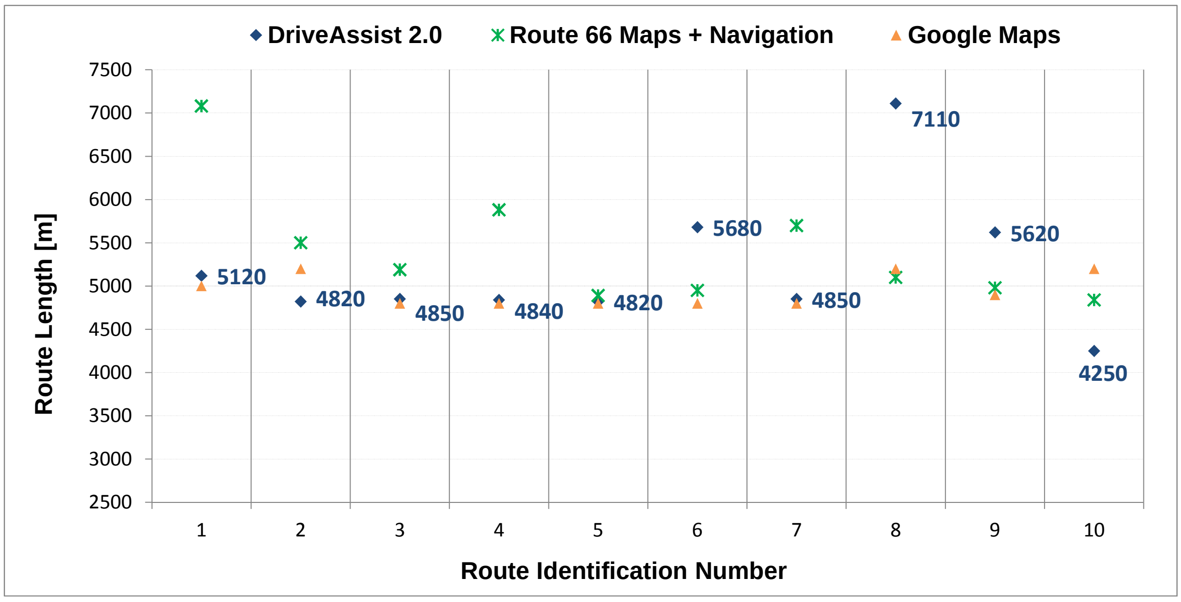


Figure 6.4.: Plot of the calculated route lengths in distance group of 5km. The route identification numbers refer to Table 6.2. Depicted are the proposed route lengths provided by three different applications.

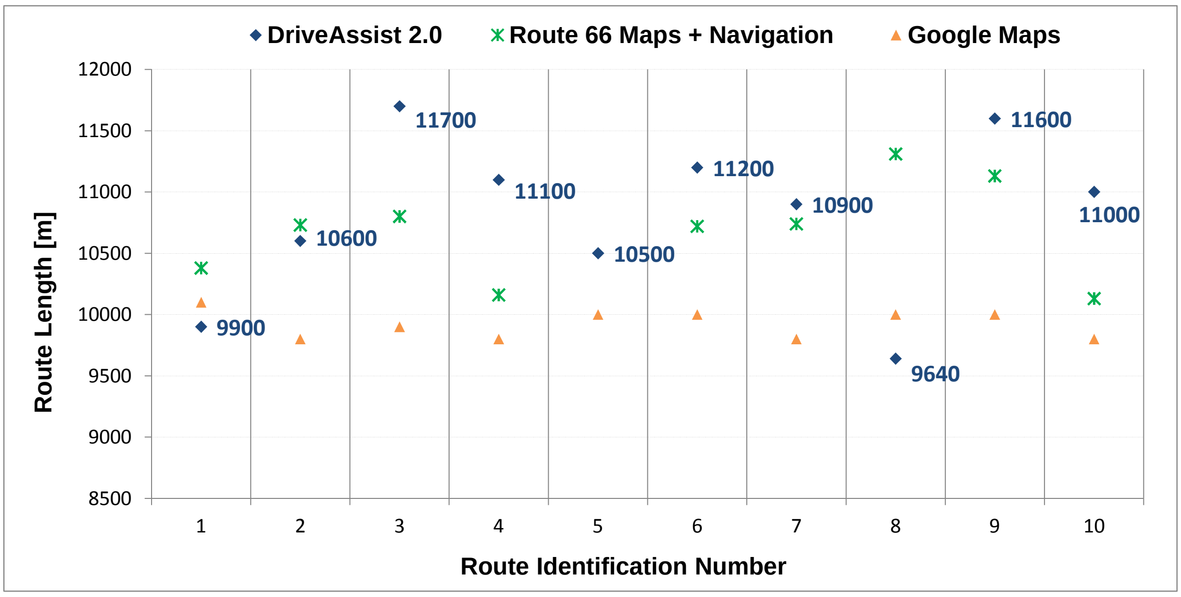


Figure 6.5.: Plot of the calculated route lengths in distance group of 10km. The route identification numbers refer to Table 6.3. Depicted are the proposed route lengths provided by three different applications.

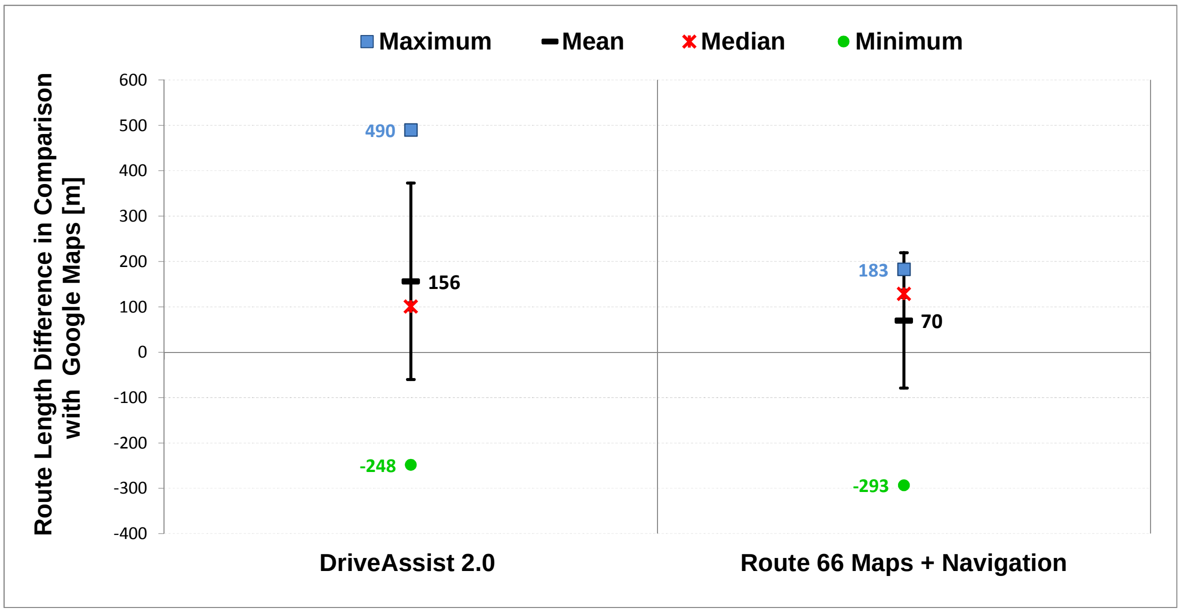


Figure 6.6.: Statistical comparison of the difference in calculated route length between *DriveAssist 2.0* and *Route 66 Maps + Navigation* when using *Google Maps* as reference. Depicted are the maximum, minimum, median, mean and standard deviation of the proposed 1km route length differences (see Table 6.1).

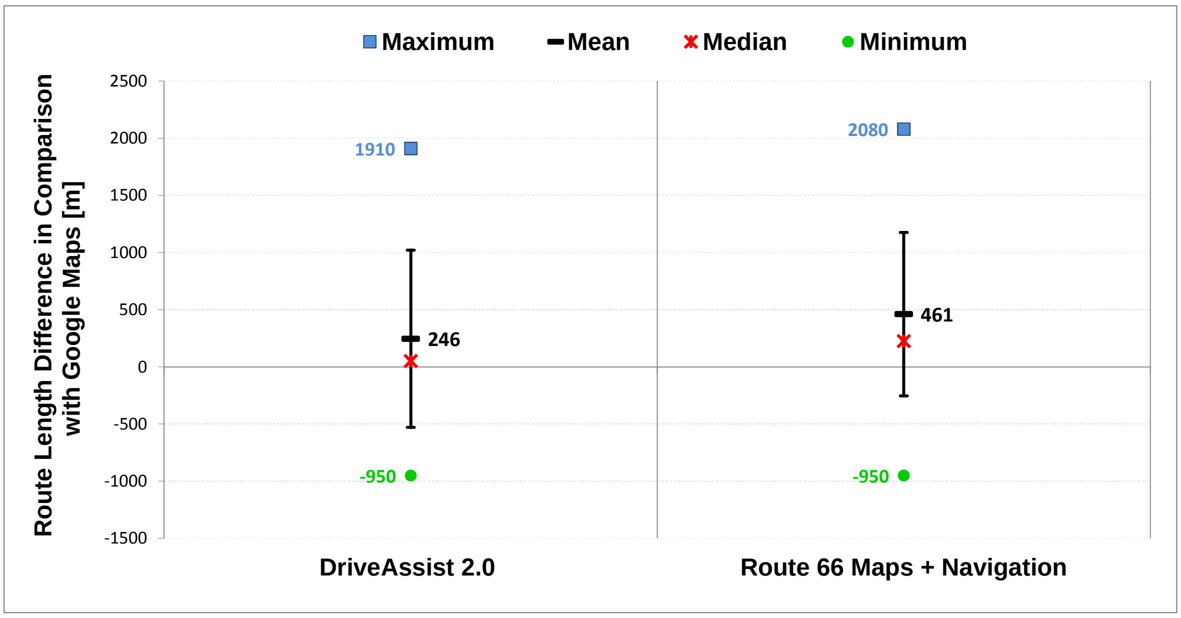


Figure 6.7.: Statistical comparison of the difference in calculated route length between *DriveAssist 2.0* and *Route 66 Maps + Navigation* when using *Google Maps* as reference. Depicted are the maximum, minimum, median, mean and standard deviation of the proposed 5km route length differences (see Table 6.2).

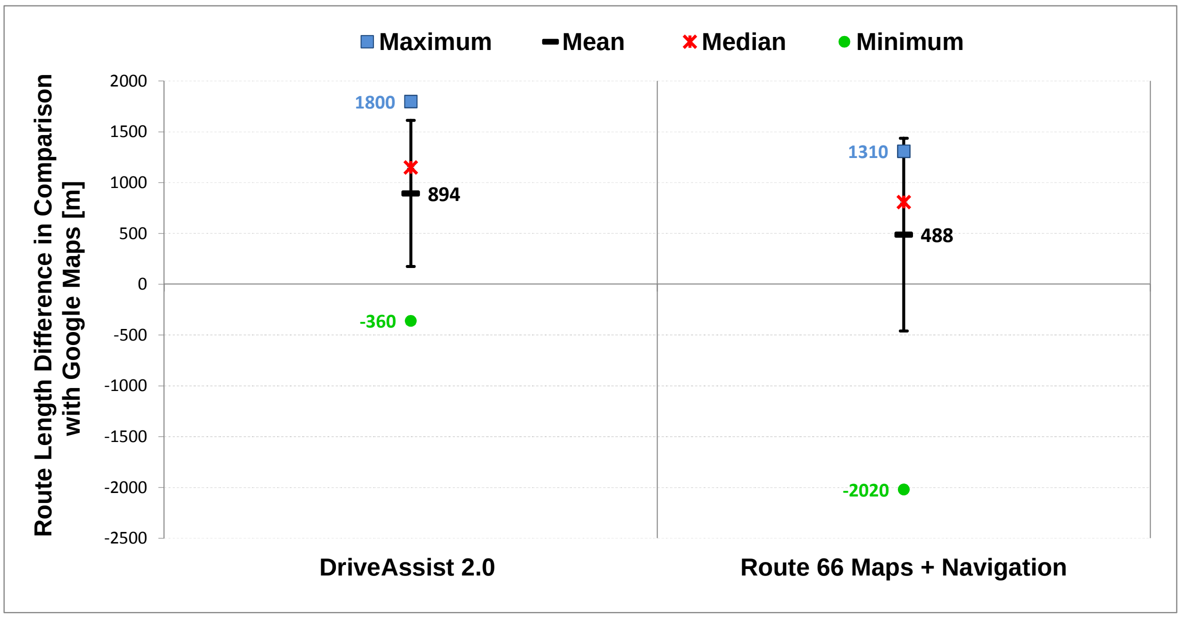


Figure 6.8.: Statistical comparison of the difference in calculated route length between *DriveAssist 2.0* and *Route 66 Maps + Navigation* when using *Google Maps* as reference. Depicted are the maximum, minimum, median, mean and standard deviation of the proposed 10km route length differences (see Table 6.3).

As can be seen in Figure 6.3 through Figure 6.5, *DriveAssist 2.0* does not always offer the same solution as *Google Maps* or *Route 66 Maps + Navigation*. The most probable reason for this is the fact that the digital maps which are used for *DriveAssist 2.0* origin from the license free data of OpenStreeMap and hence, differ from the databases of the other two applications.
However, the deviation of the proposed routes are still within an acceptable range for the scope of this thesis and the user can rely on arriving at the destination without a long delay. Figure 6.6, Figure 6.7 and Figure 6.8 show how the difference of route lengths deviates from the reference statistically. The results demonstrate that the route calculation of *DriveAssist 2.0* are comparable to established products and hence can be employed in practical applications. Furthermore, by looking at Figure 6.9 below, one can see that the goal of displaying the route on top of the map in an attractive and clear way has been successfully implemented.

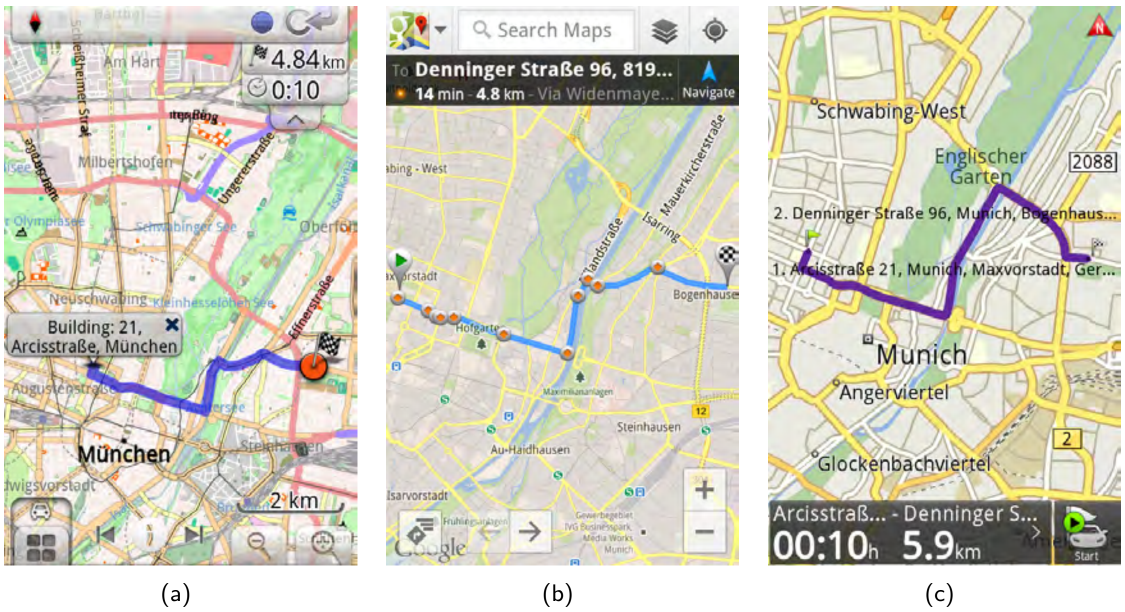
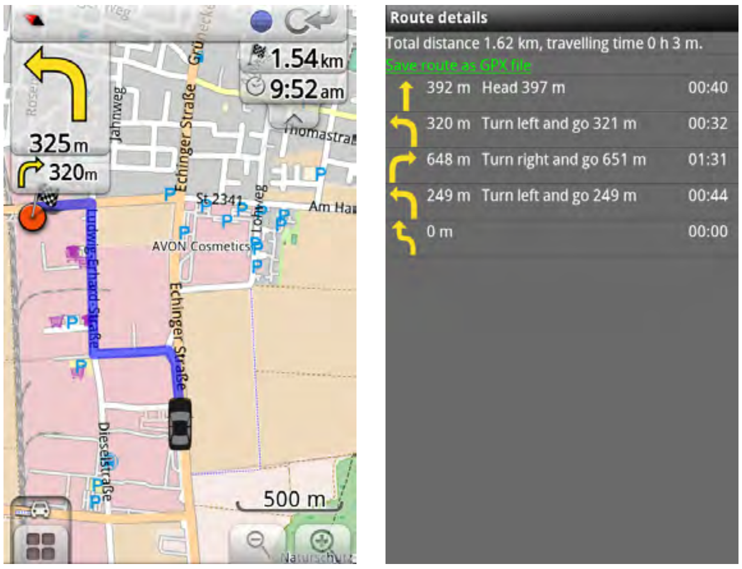

4.84 km
0:10
Building: 21, Arcisstraße, München
München
2 km
Search Maps
To Denninger Straße 96, 819...
14 min - 4.8 km - Via Widenmaye...
Navigate
Schwabing-West
Englischer Garten
2. Denninger Straße 96, Munich, Bogenhaus...
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Munich
Angerviertel
Glockenbachviertel
Arcisstraß... - Denninger S...
00:10h 5.9km
Start
(a)
(b)
(c)

Figure 6.9.: Map view provided by three different applications: (a) *DriveAssist 2.0*, (b) *Google Maps* and (c) *Route 66 Maps + Navigation*. The starting point of the visualized route is the address *Arcissstraße 21, Munich*, the end point refers to the address *Denningerstraße 96, Munich*.

6.3. Route Guiding - Differences and Similarities

As part of this thesis the routing guidance of *DriveAssist 2.0* was tested on random routes which were selected from the the Table 6.1 to Table 6.3. The change of the user's position was always simulated. In case of *DriveAssist 2.0*, the *c2xMessageTester* was used to manipulate the current user location via the provided EGO cooperative awareness messages. The application *GPS Route*

Simulator[2] was used to simulate movement along a route for Google Maps. *Route 66 Maps + Navigation* provides itself with an integrated simulation module for a specific route.
The goal was to identify strengths and weaknesses of the provided navigational instructions of *DriveAssist 2.0* in comparison with the other both applications. In this context, it should be noted that *DriveAssist 2.0* is able to provide visualized as well as verbalized instructions timely and clear. Thereby, the guidance module makes use of conventional distance instructions. In order to establish the number of provided hints, one can also look at the listed route details for the specific route (see Figure 6.10). It can reasonably be concluded that the same instructions are visualized and verbalized.
Google Maps and *Route 66 Maps + Navigation* are similarly structured. They also provide a listed overview so that the user actually knows before the beginning of the journey how many instructions to expect on a route while moving.



Route details
Total distance 1.62 km, travelling time 0 h 3 m.
Save route as GPX file

392 m	Head 397 m	00:40
320 m	Turn left and go 321 m	00:32
648 m	Turn right and go 651 m	01:31
249 m	Turn left and go 249 m	00:44
0 m		00:00

(a) (b)

Figure 6.10.: (a) Map view of *DriveAssist 2.0* which shows the provided visualized navigational instructions in the top left corner of the screen, (b) depicts the corresponding route details which serve as a basis for the verbalized instructions.

6.4. Evaluation Results

As the implementation requires a respective review and validation of principle navigational capabilities, aspects in interaction design, route calculation and result representation were tested for

[2]The application GPS Route Simulator, `https://play.google.com/store/apps/details?id=com.forgottenprojects.mocklocations&hl=de`, accessed December 14, 2012

practical use in comparison with established products on the market. The results clearly show that despite an early development state, the proposed *DriveAssist 2.0* can already serve as a system of practical use. The interaction complexity is based on a different concept than the reference systems but is of similarly low complexity. Further improvements may require a respective interaction user study, which is beyond the scope of this thesis. The quantitative comparison of offline route estimation, using *Google Maps* as a reference, reveals slight deviations, presumably due to differences in the digital map data, but in an acceptable range that compares with the performance of other navigation products.
The performance of *DriveAssist 2.0* in that regard will always be depending on the quality of digital maps provided by a non-licensing community, regardless of the employed algorithm for route calculation. Lastly, the representation of the estimated route and the navigational instructions are represented in a way that coheres to the requirements established earlier and which is very similar to the comparing products.

Chapter 7.

Conclusion and Outlook

7.1. Conclusion

This thesis represents a milestone in further developing the Android-based driver assistant system *DriveAssist*. The goal was to add navigational functionalities to the previously existing features.
In order to meet this objective, different open source projects which support OpenStreetMap were analyzed. In particular, the freely available source code of *Gosmore*, *Navit* and *OsmAnd* were examined in detail. It was discovered that *OsmAnd* was the most suitable project for the enhancement of *DriveAssist*. As a result both projects were merged into *DriveAssist 2.0*.
The extended version provides a lot of new functionalities, including the search for an address using six different kinds of destination entry methods, online and offline routing with verbalized and visualized navigation instructions. The existing UI was replaced by a more user-friendly UI design.
Finally, *Google Maps* and *Route 66 Maps + Navigation* have been used to evaluate *DriveAssist 2.0*. The interaction complexity for a route search was compared as well as the functional capacity of the route calculation and the guidance module. It has been shown that in all three areas the implemented application meets the requirements of a sufficiently reliable mobile navigation system.

7.2. Outlook

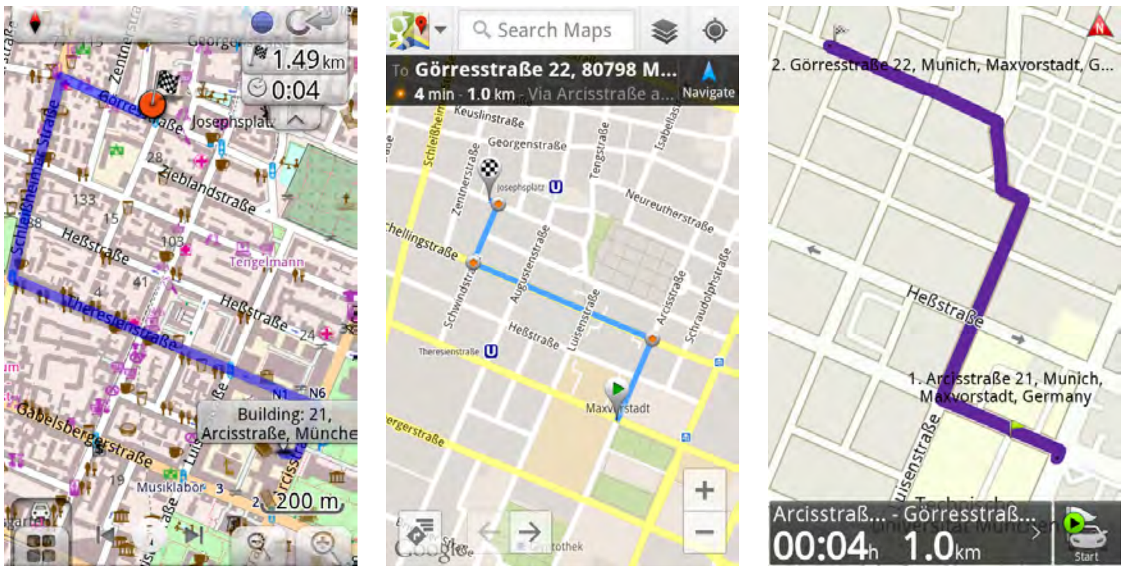
DriveAssist 2.0 is a good example of the next generation ADAS. With the help of the V2X communication module, as well as reliable routing and navigation functionalities the high potential in improving road safety and comfort becomes very clear. The driver is not just navigated to the desired destination, but also warned in case of traffic disruptions.
Despite the many functions available, there is still potential of improvement in terms of the destination entry, the routing calculation, the route guidance and the UI of *DriveAssist 2.0*. The objectives of further development may be the following:

- Destination entry: As already mentioned in Section 6.1 the destination entry of *DriveAssist 2.0* differs from the provided solutions by *Google Maps* and *Route 66 Maps + Navigation*. For this purpose, the next stage could be to adapt the procedural details to the destination entry of these two applications. Such a solution would have the advantage that the user would be familiar with the procedure from its very beginning and would not have to acquire new knowledge.
- Route calculation: The currently implemented routing algorithm A* is very reliable, but it would be of great interest to enhance it in such a way, that the calculation of the route could be based on real-time traffic information. Some useful ideas for this kind of enhancement can be found in Section 2.5.
- Route guidance: In the actual application development lifecycle conventional distance instructions are used to assist the driver. For the future, a step towards including features of the environment into the verbalized instructions could be done. As the research in this area shows that such an approach offers advantages in comparison to the conventional one [29, p. 73]. In this context, Section 2.7 provides more details.
- UI: The user acceptance of an application strongly depends on the UI, for this reason there is always a need for optimization. With the help of a user study, strengths and weaknesses of the actually implemented interaction functionalities can be identified and may be used for further improvements.

Appendix A.

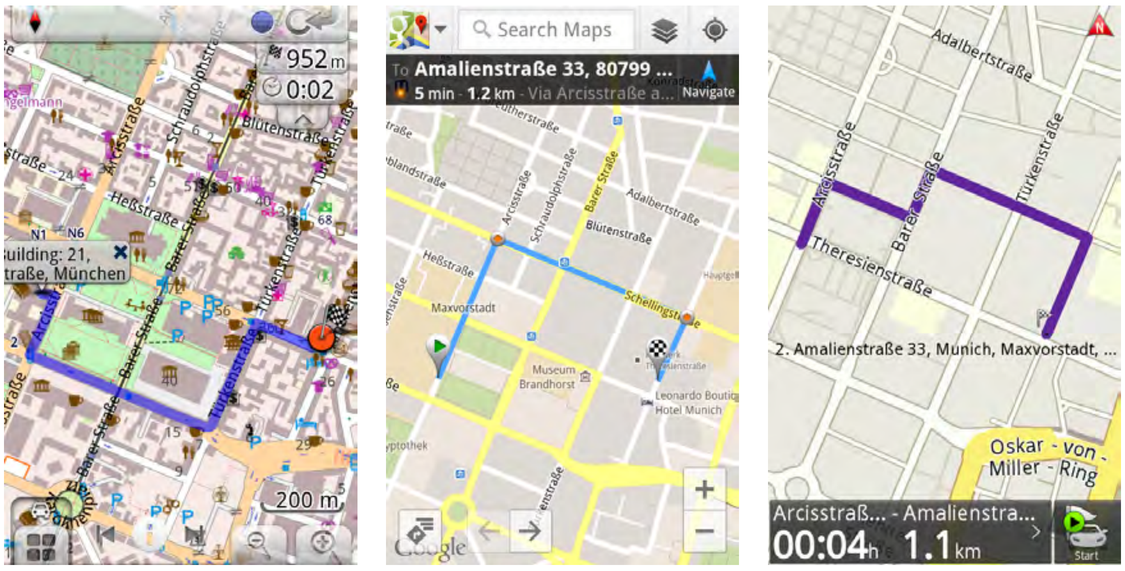
Screenshots

A.0.1. Examples of routes with an approximate route length of 1km


1.49 km
0:04
Building: 21, Arcisstraße, Münche
200 m
Search Maps
To Görresstraße 22, 80798 M...
4 min · 1.0 km · Via Arcisstraße a...
Navigate
2. Görresstraße 22, Munich, Maxvorstadt, G...
1. Arcisstraße 21, Munich, Maxvorstadt, Germany
Arcisstraß... - Görresstraß...
00:04h 1.0km
Start

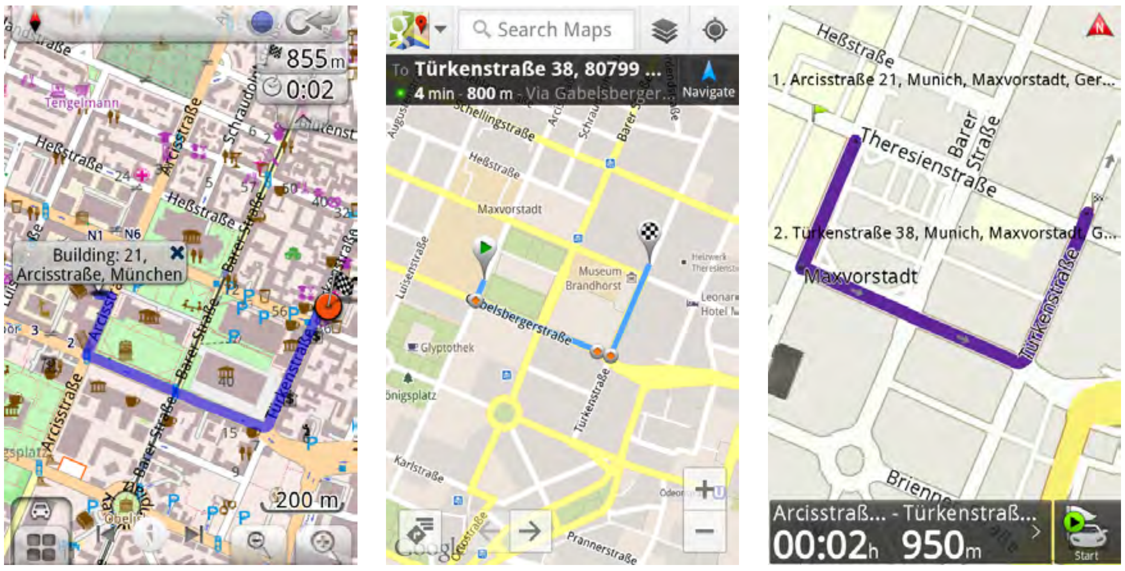
(a) (b) (c)

Figure A.1.: Proposed route from *Arcisstraße 21, Munich* to *Görrestraße 22, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


952 m
0:02
200 m
Search Maps
To Amalienstraße 33, 80799 ...
5 min · 1.2 km · Via Arcisstraße a...
Navigate
2. Amalienstraße 33, Munich, Maxvorstadt, ...
Arcisstraß... - Amalienstra...
00:04 h 1.1 km
Start

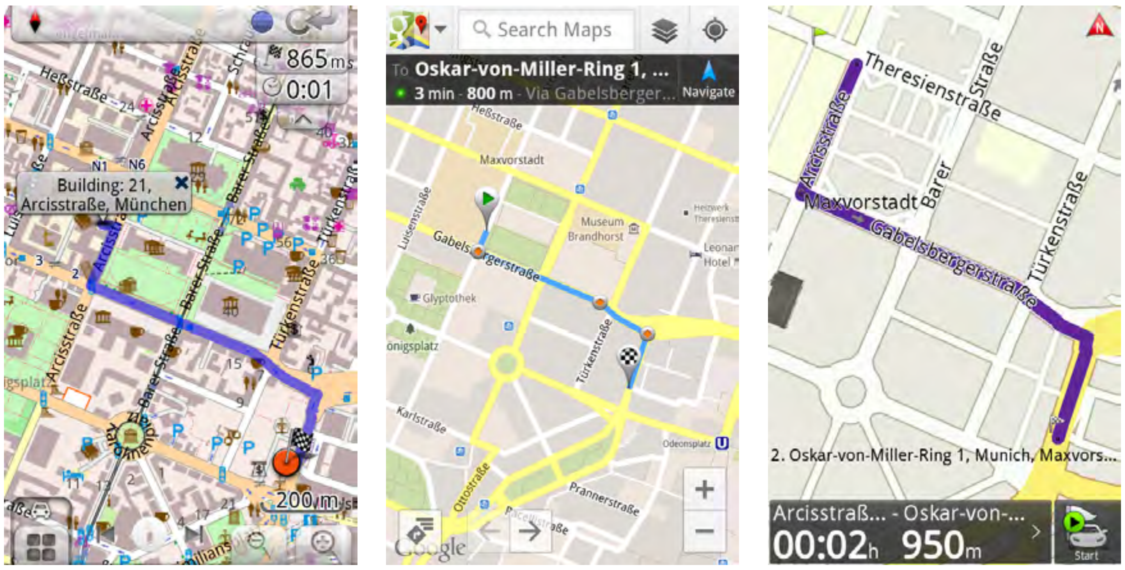
(a) (b) (c)

Figure A.2.: Proposed route from *Arcisstraße 21, Munich* to *Amalienstraße 33, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


855 m
0:02
Building: 21, Arcisstraße, München
200 m
Search Maps
To Türkenstraße 38, 80799 ...
4 min · 800 m · Via Gabelsberger...
Navigate
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
2. Türkenstraße 38, Munich, Maxvorstadt, G...
Arcisstraß... - Türkenstraß...
00:02 h 950 m
Start

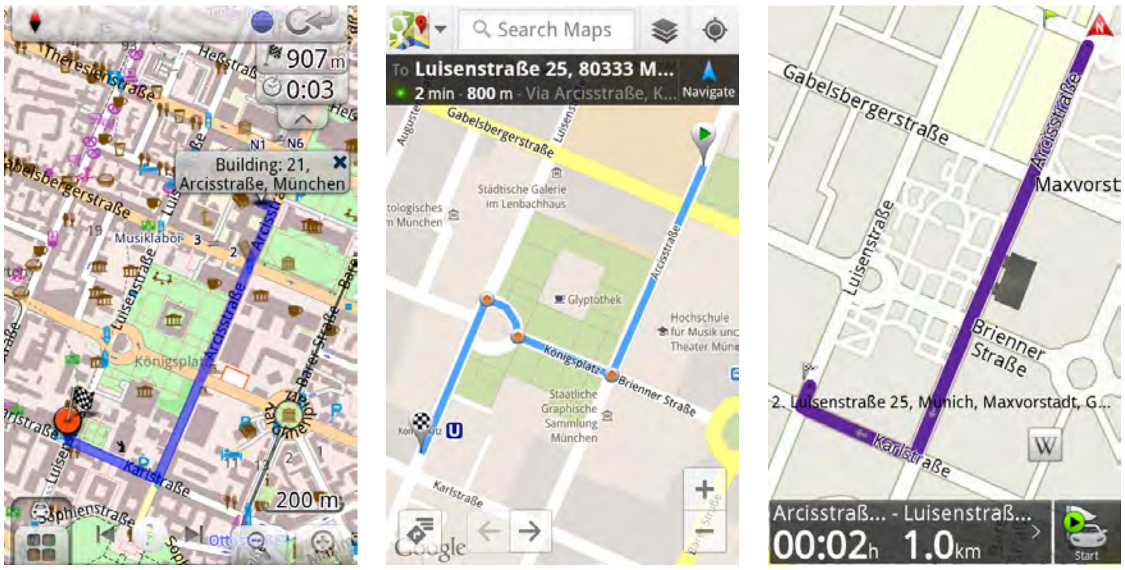
(a) (b) (c)

Figure A.3.: Proposed route from *Arcisstraße 21, Munich* to *Türkenstraße 38, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


865 m
0:01
Building: 21, Arcisstraße, München
200 m
Search Maps
To Oskar-von-Miller-Ring 1, ...
3 min - 800 m - Via Gabelsberger...
Navigate
2. Oskar-von-Miller-Ring 1, Munich, Maxvors...
Arcisstraß... - Oskar-von-...
00:02h 950m
Start

(a) (b) (c)

Figure A.4.: Proposed route from *Arcisstraße 21, Munich* to *Oskar-von-Miller-Ring 1, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


907 m
0:03
Building: 21, Arcisstraße, München
200 m
Search Maps
To Luisenstraße 25, 80333 M...
2 min - 800 m - Via Arcisstraße, K...
Navigate
2. Luisenstraße 25, Munich, Maxvorstadt, G...
Arcisstraß... - Luisenstraß...
00:02h 1.0km
Start

(a) (b) (c)

Figure A.5.: Proposed route from *Arcisstraße 21, Munich* to *Luisenstraße 25, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.

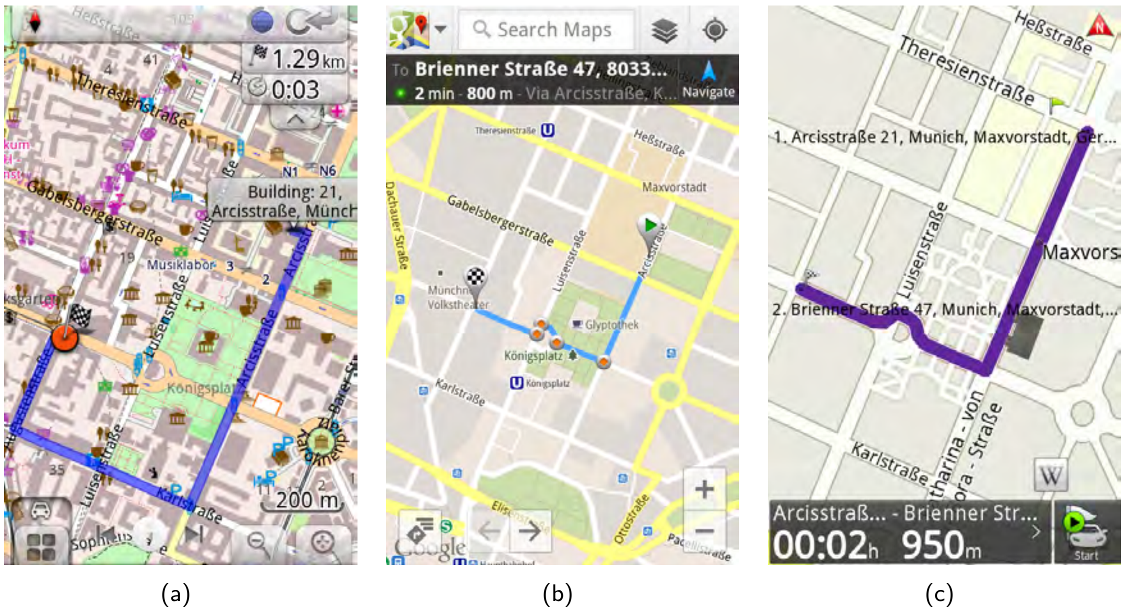

1.29 km
0:03
Building: 21, Arcisstraße, Münch
200 m
Search Maps
To Brienner Straße 47, 8033...
2 min · 800 m · Via Arcisstraße, K...
Navigate
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
2. Brienner Straße 47, Munich, Maxvorstadt,...
Arcisstraß... - Brienner Str...
00:02h 950m
Start
(a)
(b)
(c)

Figure A.6.: Proposed route from *Arcisstraße 21, Munich* to *Brienner Straße 47, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.

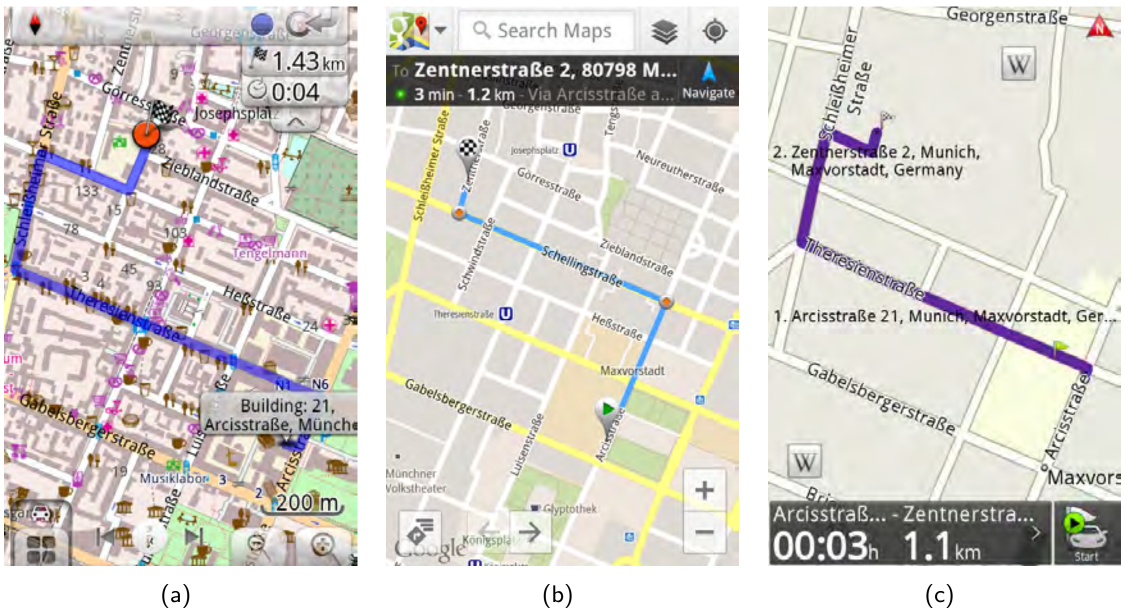
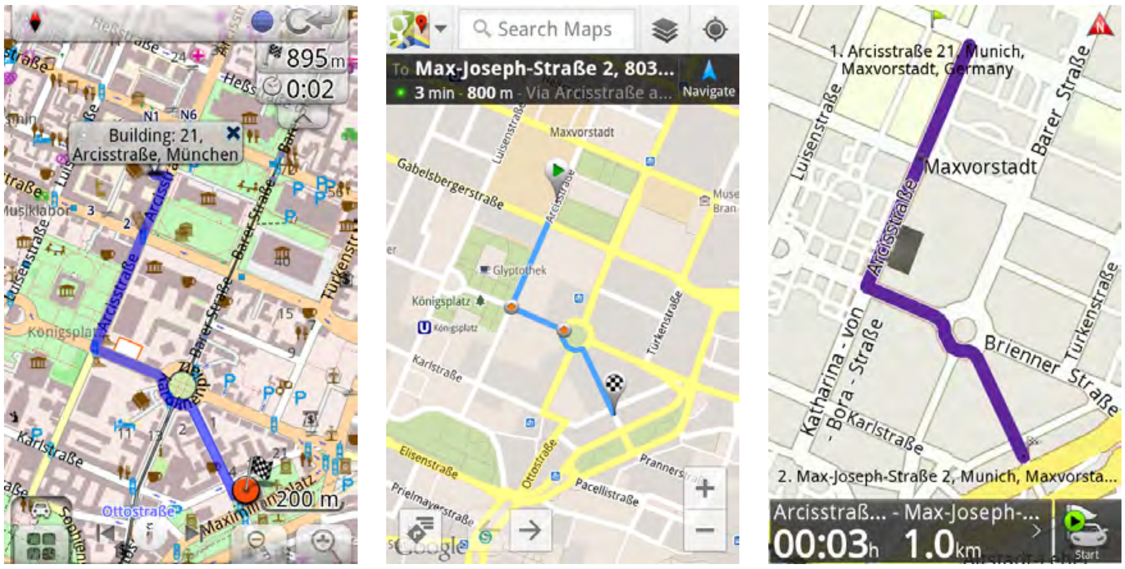
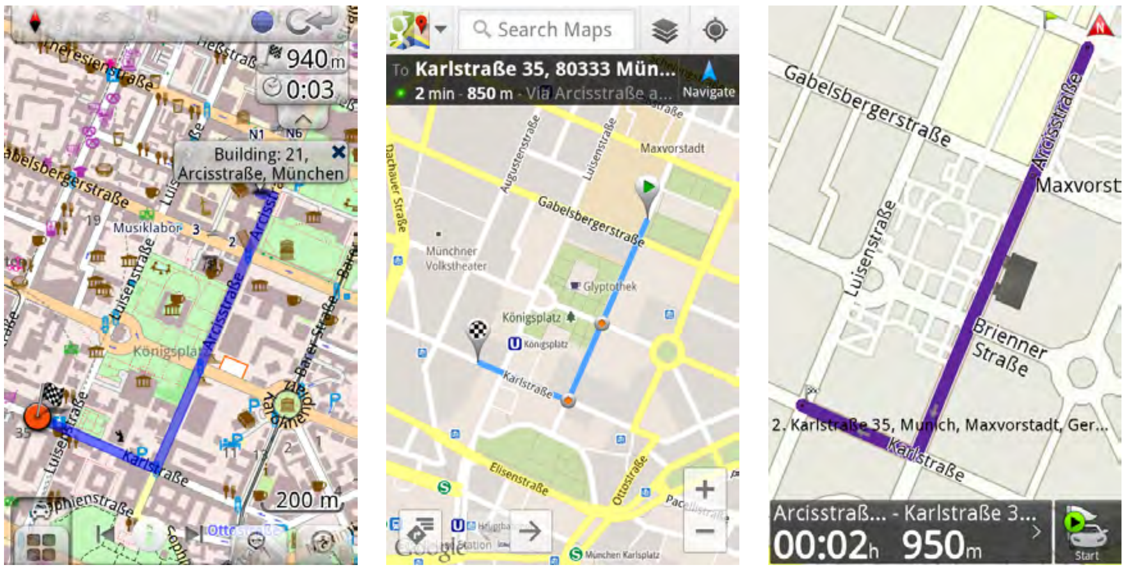

1.43 km
0:04
Building: 21, Arcisstraße, Münche
200 m
Search Maps
To Zentnerstraße 2, 80798 M...
3 min · 1.2 km · Via Arcisstraße a...
Navigate
2. Zentnerstraße 2, Munich, Maxvorstadt, Germany
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Arcisstraß... - Zentnerstra...
00:03h 1.1km
Start
(a)
(b)
(c)

Figure A.7.: Proposed route from *Arcisstraße 21, Munich* to *Zentnerstraße 2, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


895 m
0:02
Building: 21, Arcisstraße, München
200 m
Search Maps
To Max-Joseph-Straße 2, 803...
3 min · 800 m · Via Arcisstraße a...
Navigate
1. Arcisstraße 21, Munich, Maxvorstadt, Germany
2. Max-Joseph-Straße 2, Munich, Maxvorsta...
Arcisstraß... - Max-Joseph-...
00:03h 1.0km
Start

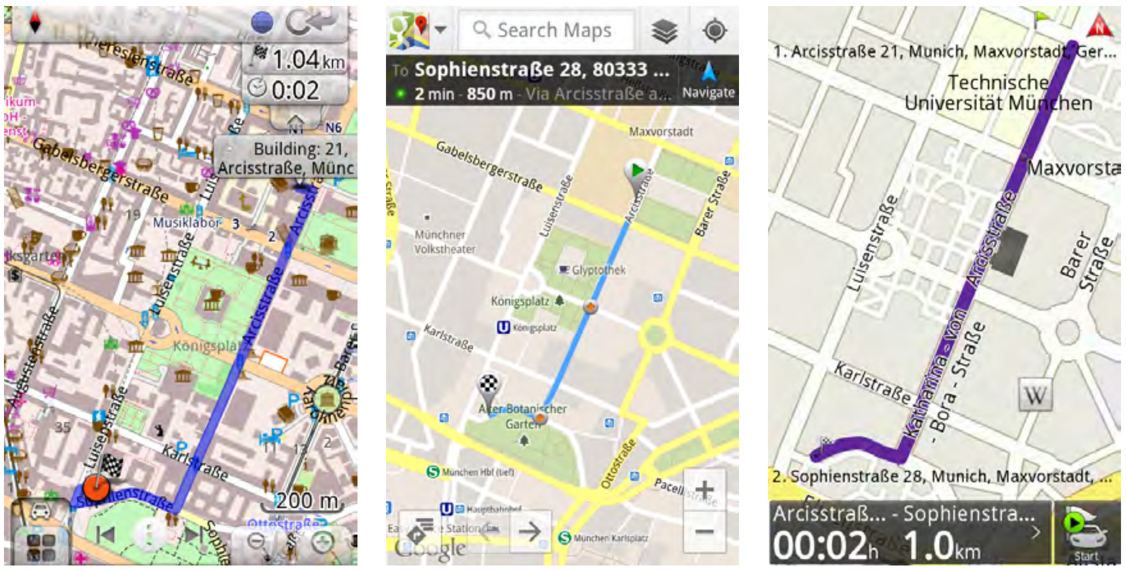
(a) (b) (c)

Figure A.8.: Proposed route from *Arcisstraße 21, Munich* to *Max-Joseph-Straße 2, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


940 m
0:03
Building: 21, Arcisstraße, München
200 m
Search Maps
To Karlstraße 35, 80333 Mün...
2 min · 850 m · Via Arcisstraße a...
Navigate
2. Karlstraße 35, Munich, Maxvorstadt, Ger...
Arcisstraß... - Karlstraße 3...
00:02h 950m
Start

(a) (b) (c)

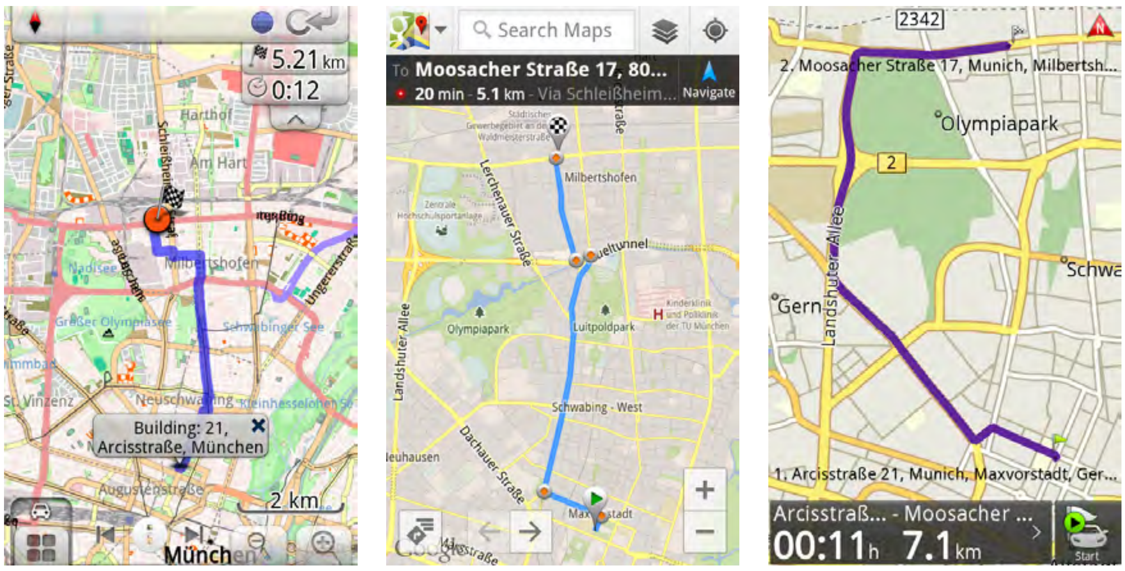
Figure A.9.: Proposed route from *Arcisstraße 21, Munich* to *Karlstraße 35, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


1.04 km
0:02
Building: 21, Arcisstraße, Münc
200 m
Search Maps
To Sophienstraße 28, 80333 ...
2 min - 850 m - Via Arcisstraße a...
Navigate
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
2. Sophienstraße 28, Munich, Maxvorstadt, ...
Arcisstraß... - Sophienstra...
00:02h 1.0km
Start

(a) (b) (c)

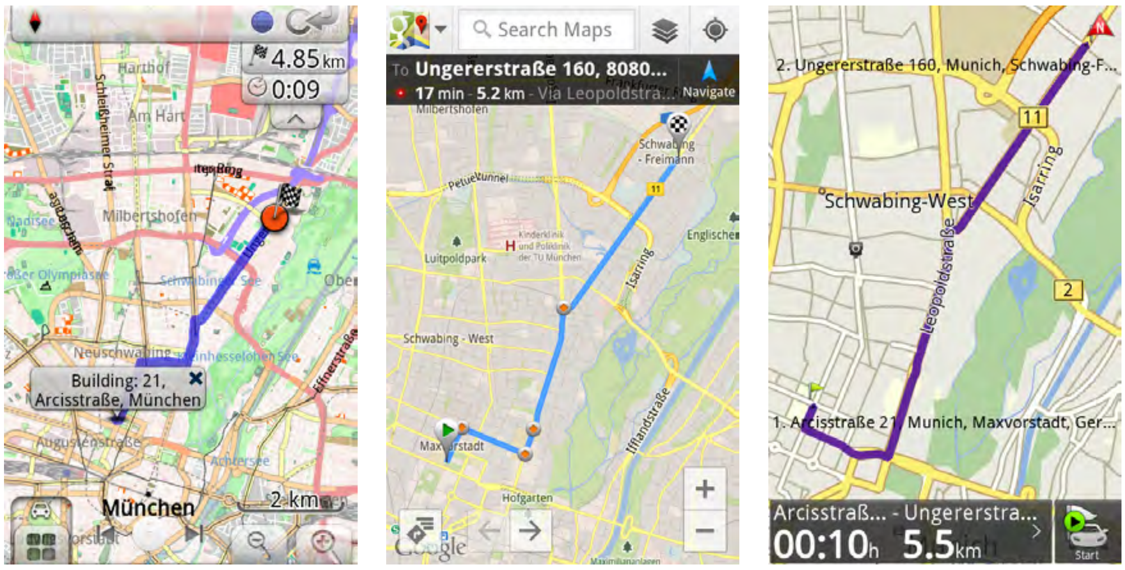
Figure A.10.: Proposed route from *Arcisstraße 21, Munich* to *Sophienstraße 28, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.

A.0.2. Examples of routes with an approximate route length of 5km


5.21 km
0:12
Building: 21, Arcisstraße, München
2 km
München
Search Maps
To Moosacher Straße 17, 80...
20 min · 5.1 km · Via Schleißheim...
Navigate
Milbertshofen
Olympiapark
Luitpoldpark
Schwabing - West
Maxvorstadt
[2342]
2. Moosacher Straße 17, Munich, Milbertsh...
Olympiapark
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Arcisstraß... - Moosacher ...
00:11h 7.1km
Start

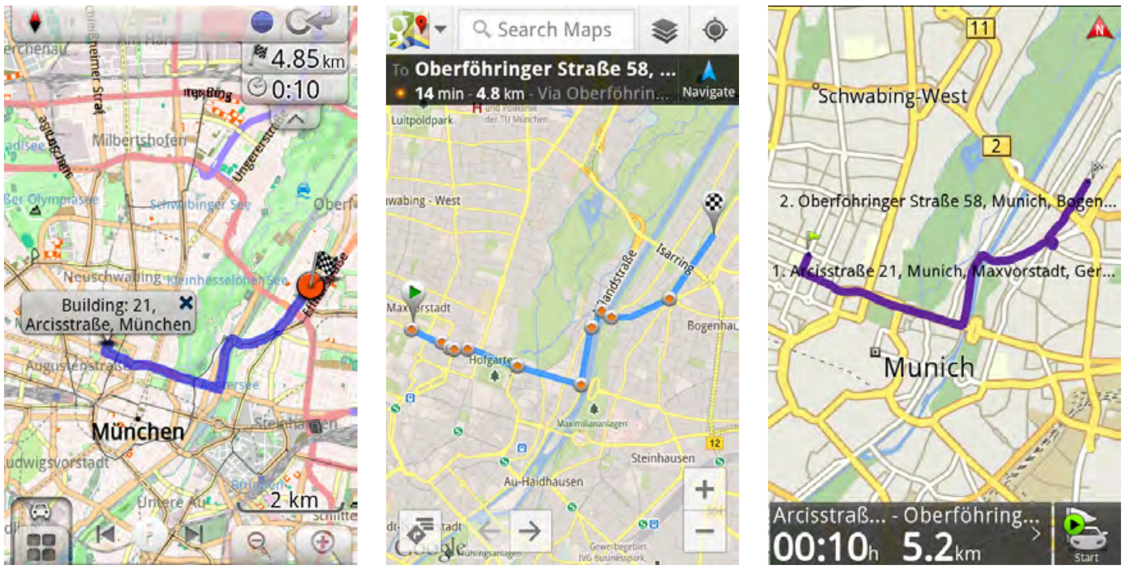
(a) (b) (c)

Figure A.11.: Proposed route from *Arcisstraße 21, Munich* to *Moosacher Straße 17, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


4.85 km
0:09
Building: 21, Arcisstraße, München
2 km
München
Search Maps
To Ungererstraße 160, 8080...
17 min · 5.2 km · Via Leopoldstra...
Navigate
Schwabing - Freimann
Luitpoldpark
Schwabing - West
Maxvorstadt
Hofgarten
2. Ungererstraße 160, Munich, Schwabing-F...
Schwabing-West
Leopoldstraße
Isarring
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Arcisstraß... - Ungererstra...
00:10h 5.5km
Start

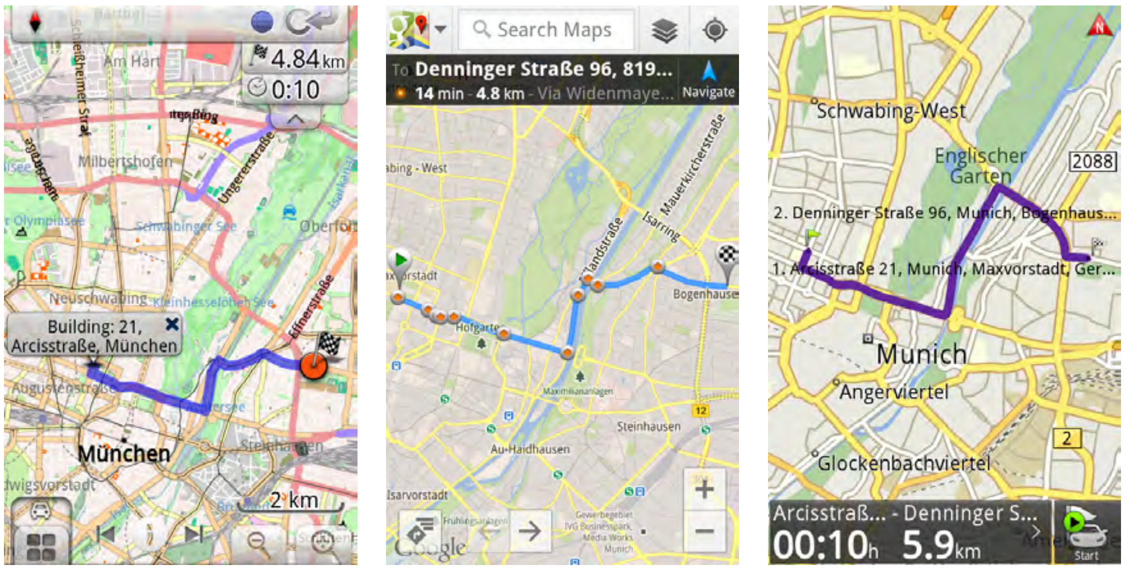
(a) (b) (c)

Figure A.12.: Proposed route from *Arcisstraße 21, Munich* to *Ungererstraße 160, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


4.85 km
0:10
Building: 21, Arcisstraße, München
München
2 km
Search Maps
To Oberföhringer Straße 58, ...
14 min · 4.8 km - Via Oberföhrin...
Navigate
Schwabing-West
2. Oberföhringer Straße 58, Munich, Bogen...
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Munich
Arcisstraß... - Oberföhring...
00:10h 5.2km
Start

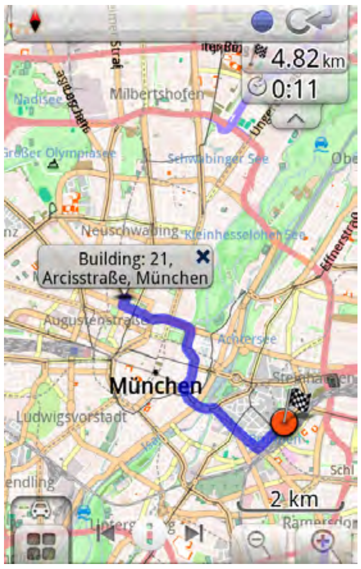
(a) (b) (c)

Figure A.13.: Proposed route from *Arcisstraße 21, Munich* to *Oberföhringer Straße 58, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.

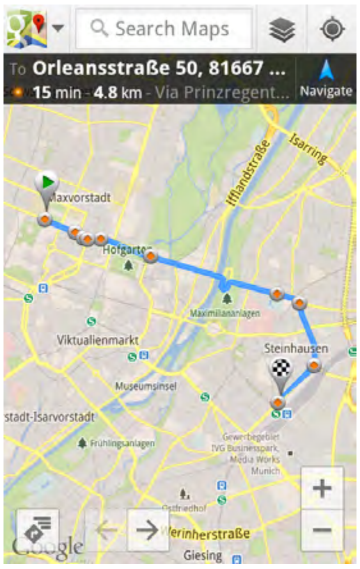

4.84 km
0:10
Building: 21, Arcisstraße, München
München
2 km
Search Maps
To Denninger Straße 96, 819...
14 min · 4.8 km - Via Widenmaye...
Navigate
Schwabing-West
Englischer Garten
2. Denninger Straße 96, Munich, Bogenhaus...
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Munich
Angerviertel
Glockenbachviertel
Arcisstraß... - Denninger S...
00:10h 5.9km
Start

(a) (b) (c)

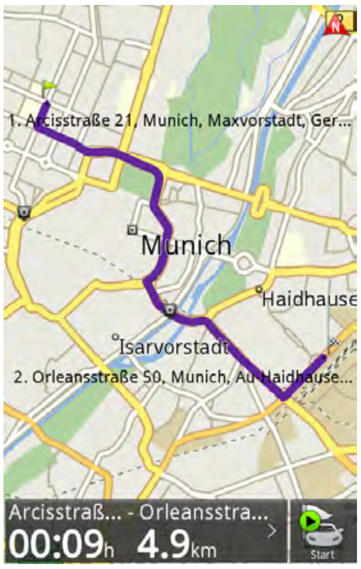
Figure A.14.: Proposed route from *Arcisstraße 21, Munich* to *Denninger Straße 96, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.



(a)

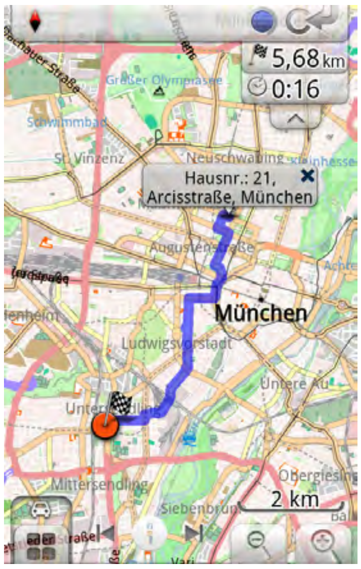


(b)

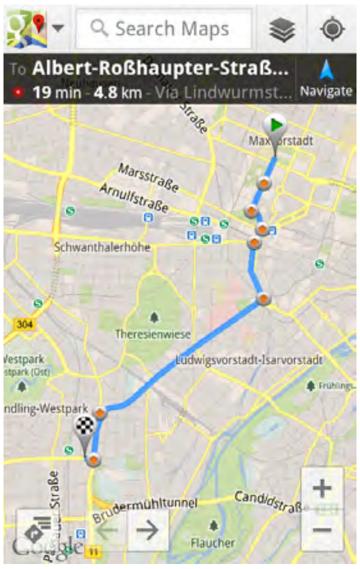

1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Munich
Haidhause
Isarvorstadt
2. Orleansstraße 50, Munich, Au-Haidhause...
Arcisstraß... - Orleansstra...
00:09h 4.9km
Start

(c)

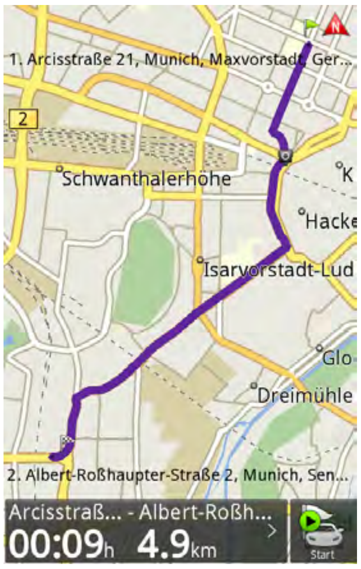
Figure A.15.: Proposed route from *Arcisstraße 21, Munich* to *Orleansstraße 50, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.



(a)

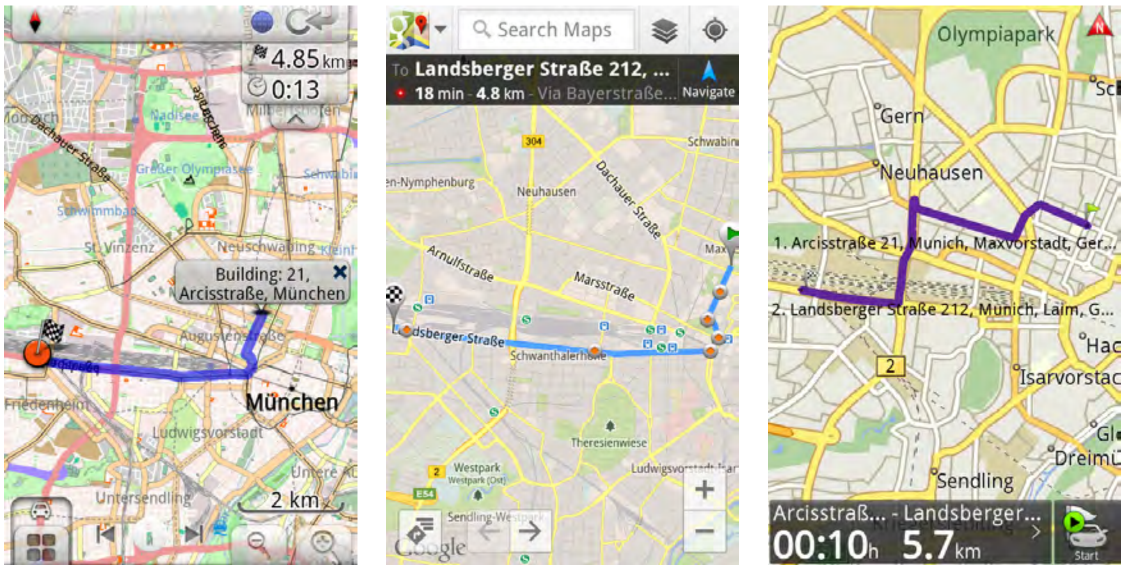

Search Maps
To Albert-Roßhaupter-Straß...
19 min 4.8 km - Via Lindwurmst...
Navigate
Maxvorstadt
Schwanthalerhöhe
Theresienwiese
Ludwigsvorstadt-Isarvorstadt

(b)


1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Schwanthalerhöhe
Isarvorstadt-Lud
Dreimühle
2. Albert-Roßhaupter-Straße 2, Munich, Sen...
Arcisstraß... - Albert-Roßh...
00:09h 4.9km
Start

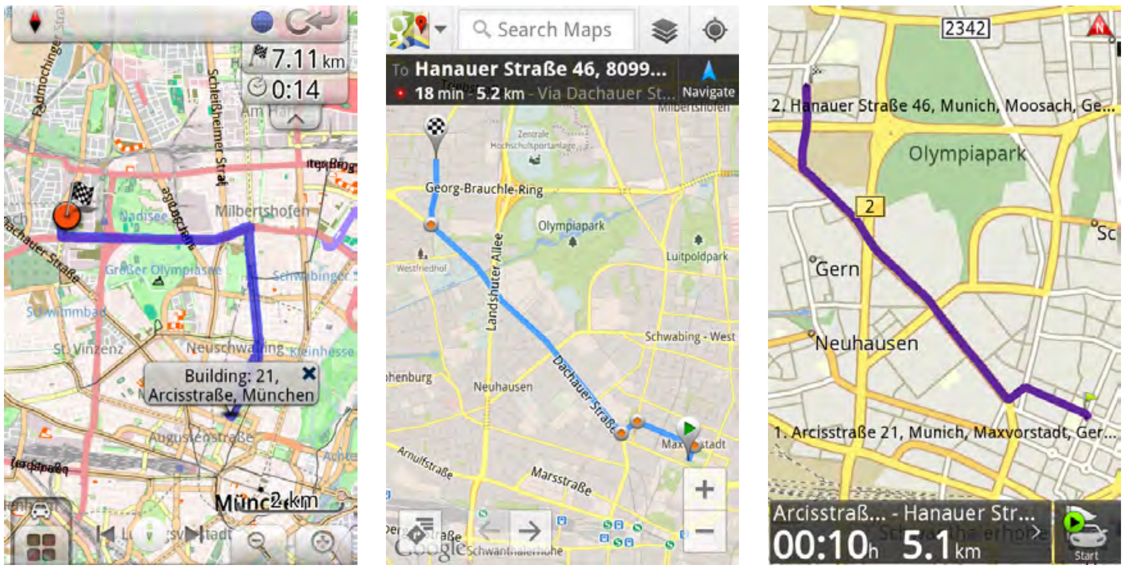
(c)

Figure A.16.: Proposed route from *Arcisstraße 21, Munich* to *Albert-Roßhaupter-Straße 2, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


4.85 km
0:13
Building: 21, Arcisstraße, München
München
2 km
Search Maps
To Landsberger Straße 212, ...
18 min · 4.8 km · Via Bayerstraße...
Navigate
Olympiapark
Gern
Neuhausen
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
2. Landsberger Straße 212, Munich, Laim, G...
Sendling
Arcisstraß... - Landsberger...
00:10h 5.7km
Start

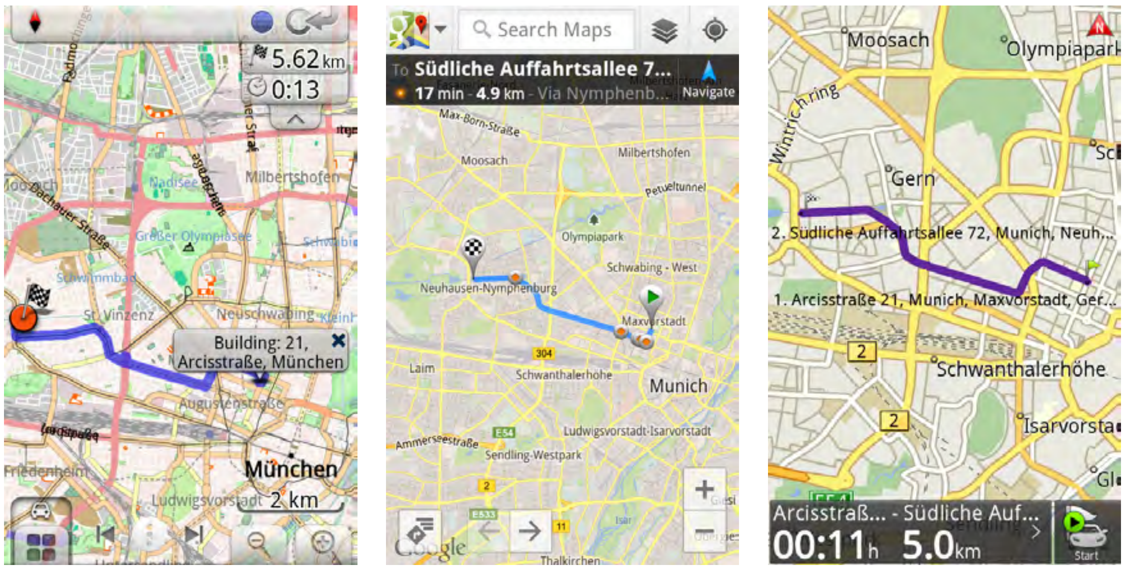
(a) (b) (c)

Figure A.17.: Proposed route from *Arcisstraße 21, Munich* to *Landsberger Straße 212, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


7.11 km
0:14
Building: 21, Arcisstraße, München
2 km
Search Maps
To Hanauer Straße 46, 8099...
18 min · 5.2 km · Via Dachauer St...
Navigate
2342
2. Hanauer Straße 46, Munich, Moosach, Ge...
Olympiapark
Gern
Neuhausen
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Arcisstraß... - Hanauer Str...
00:10h 5.1km
Start

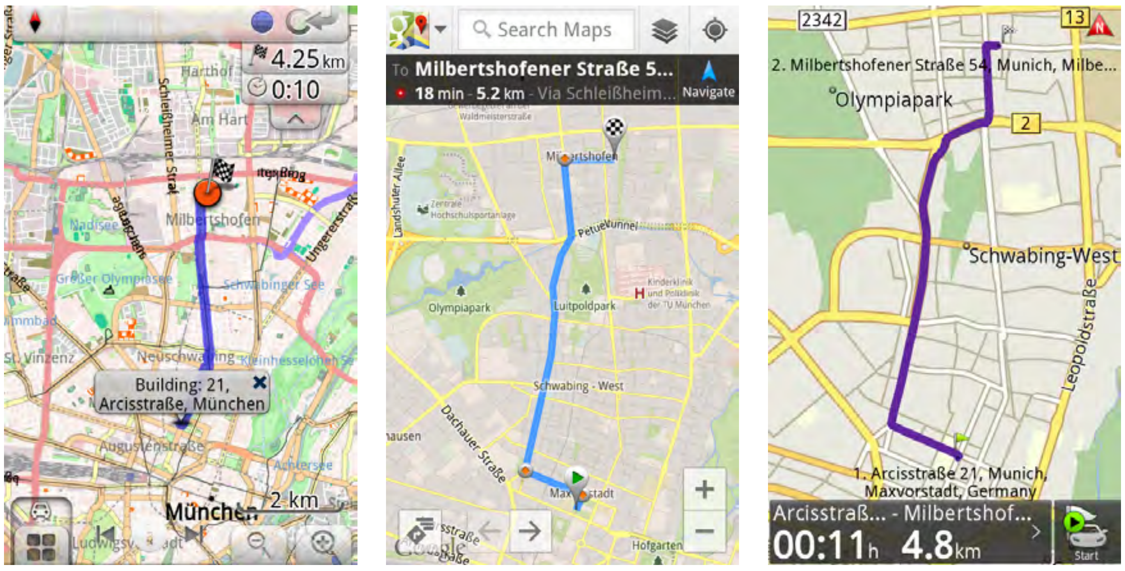
(a) (b) (c)

Figure A.18.: Proposed route from *Arcisstraße 21, Munich* to *Hanauer Straße 46, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


5.62 km
0:13
Building: 21, Arcisstraße, München
München
2 km
Search Maps
To Südliche Auffahrtsallee 7...
17 min · 4.9 km · Via Nymphenb...
Navigate
Munich
Moosach
Olympiapark
2. Südliche Auffahrtsallee 72, Munich, Neuh...
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Schwanthalerhöhe
Arcisstraß... - Südliche Auf...
00:11h 5.0km
Start

(a) (b) (c)

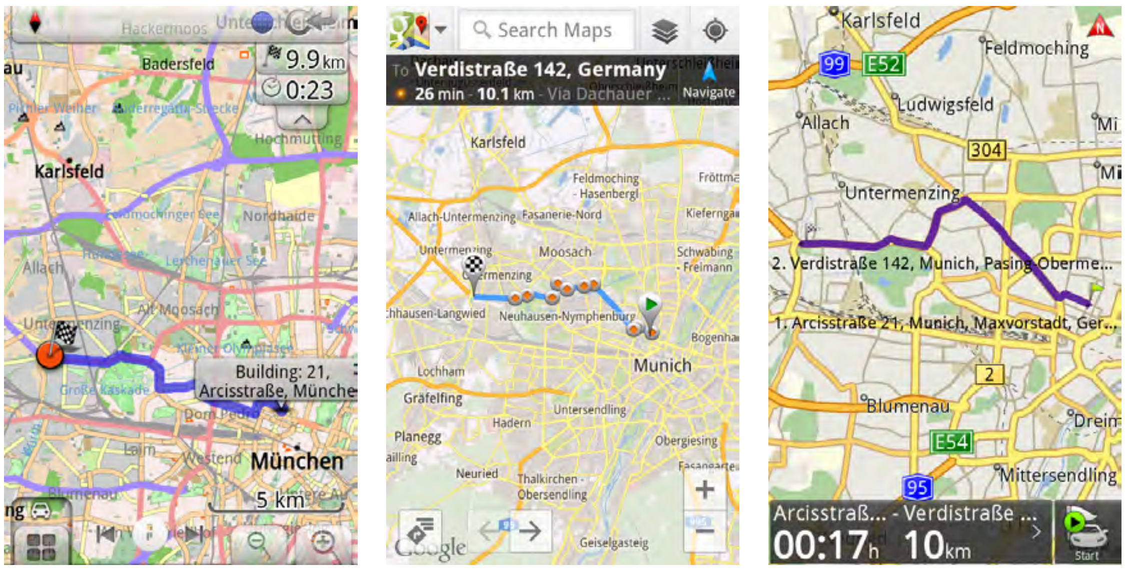
Figure A.19.: Proposed route from *Arcisstraße 21, Munich* to *Südliche Auffahrtsallee 72, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


4.25 km
0:10
Building: 21, Arcisstraße, München
München
2 km
Search Maps
To Milbertshofener Straße 5...
18 min · 5.2 km · Via Schleißheim...
Navigate
Olympiapark
Schwabing - West
2. Milbertshofener Straße 54, Munich, Milbe...
1. Arcisstraße 21, Munich, Maxvorstadt, Germany
Arcisstraß... - Milbertshof...
00:11h 4.8km
Start

(a) (b) (c)

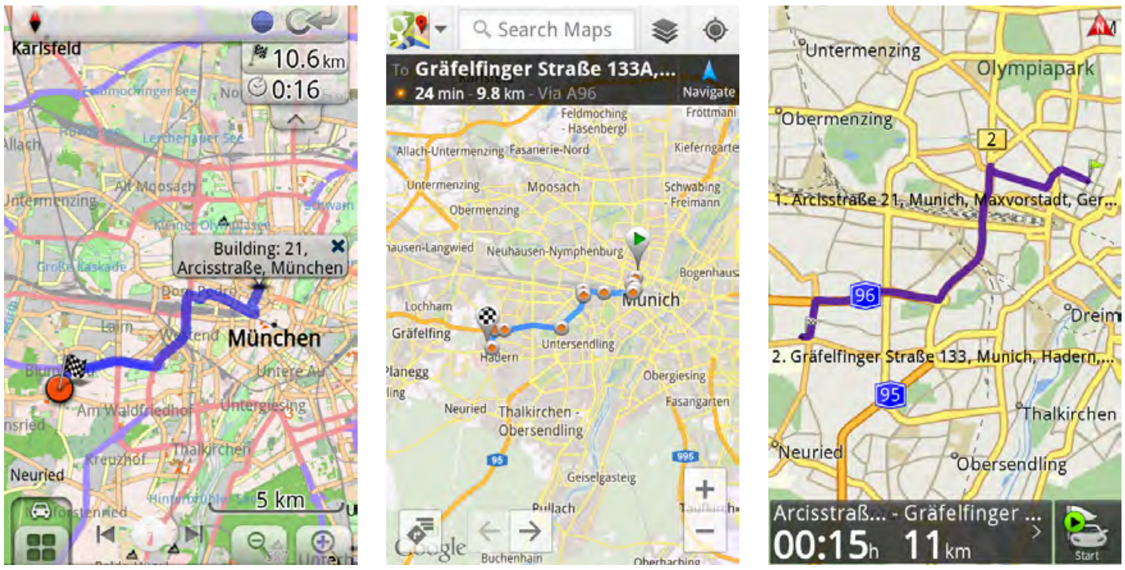
Figure A.20.: Proposed route from *Arcisstraße 21, Munich* to *Milberstshofener Straße 54, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.

A.0.3. Examples of routes with an approximate route length of 10km


9.9 km
0:23
Building: 21, Arcisstraße, Münche
Karlsfeld
München
5 km
Search Maps
To Verdistraße 142, Germany
26 min · 10.1 km · Via Dachauer ...
Navigate
Munich
Karlsfeld
2. Verdistraße 142, Munich, Pasing-Oberme...
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Arcisstraß... - Verdistraße ...
00:17h 10km
Start

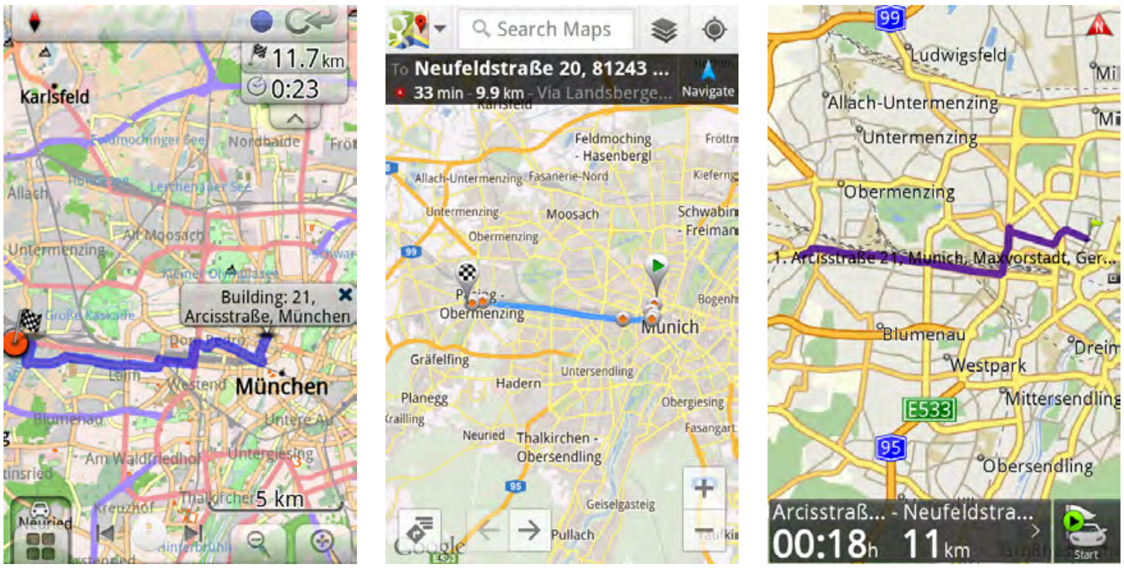
(a) (b) (c)

Figure A.21.: Proposed route from *Arcisstraße 21, Munich* to *Verdistraße 142, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


10.6 km
0:16
Building: 21, Arcisstraße, München
Karlsfeld
München
5 km
Search Maps
To Gräfelfinger Straße 133A,...
24 min · 9.8 km · Via A96
Navigate
Munich
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
2. Gräfelfinger Straße 133, Munich, Hadern,...
Arcisstraß... - Gräfelfinger ...
00:15h 11km
Start

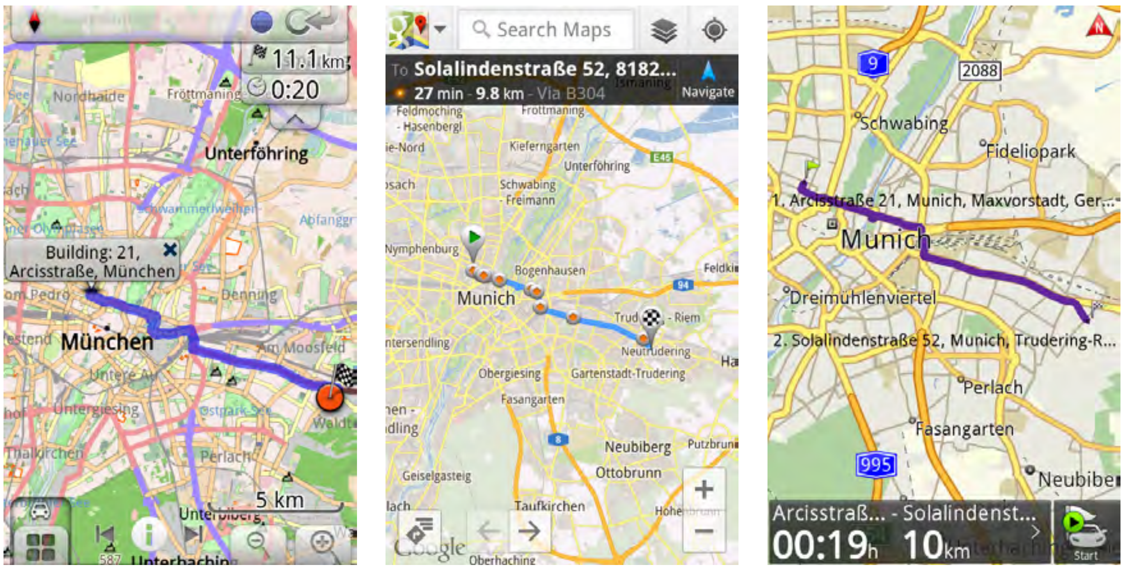
(a) (b) (c)

Figure A.22.: Proposed route from *Arcisstraße 21, Munich* to *Gräfelfinger Straße 133a, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


11.7 km
0:23
Building: 21, Arcisstraße, München
5 km
Search Maps
To Neufeldstraße 20, 81243 ...
33 min · 9.9 km · Via Landsberge...
Navigate
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Arcisstraß... - Neufeldstra...
00:18h 11km
Start

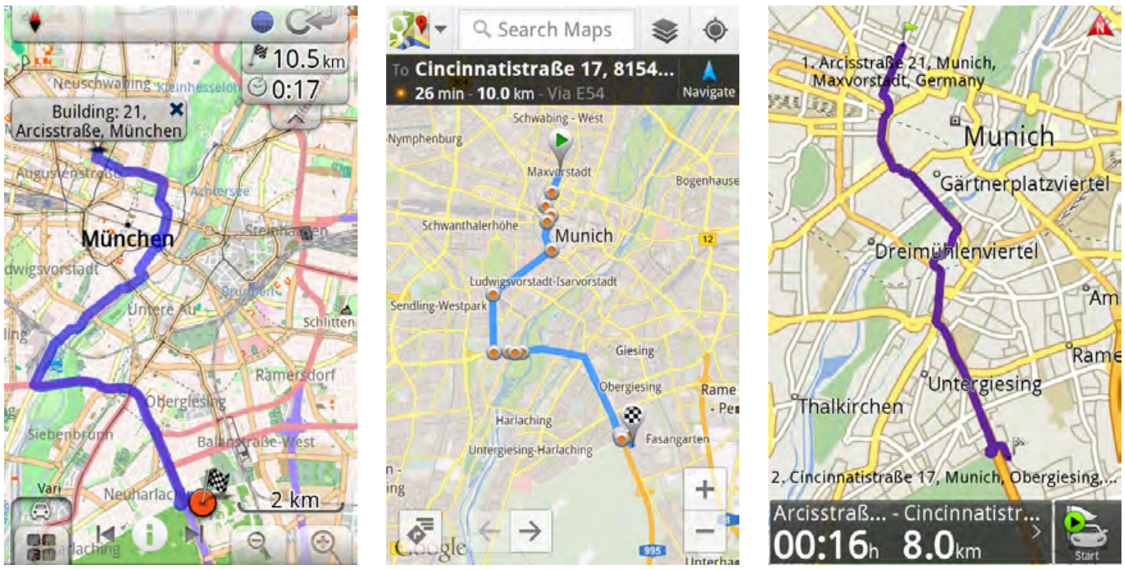
(a) (b) (c)

Figure A.23.: Proposed route from *Arcisstraße 21, Munich* to *Neufeldstraße 20, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


11.1 km
0:20
Building: 21, Arcisstraße, München
5 km
Search Maps
To Solalindenstraße 52, 8182...
27 min · 9.8 km · Via B304
Navigate
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
2. Solalindenstraße 52, Munich, Trudering-R...
Arcisstraß... - Solalindenst...
00:19h 10km
Start

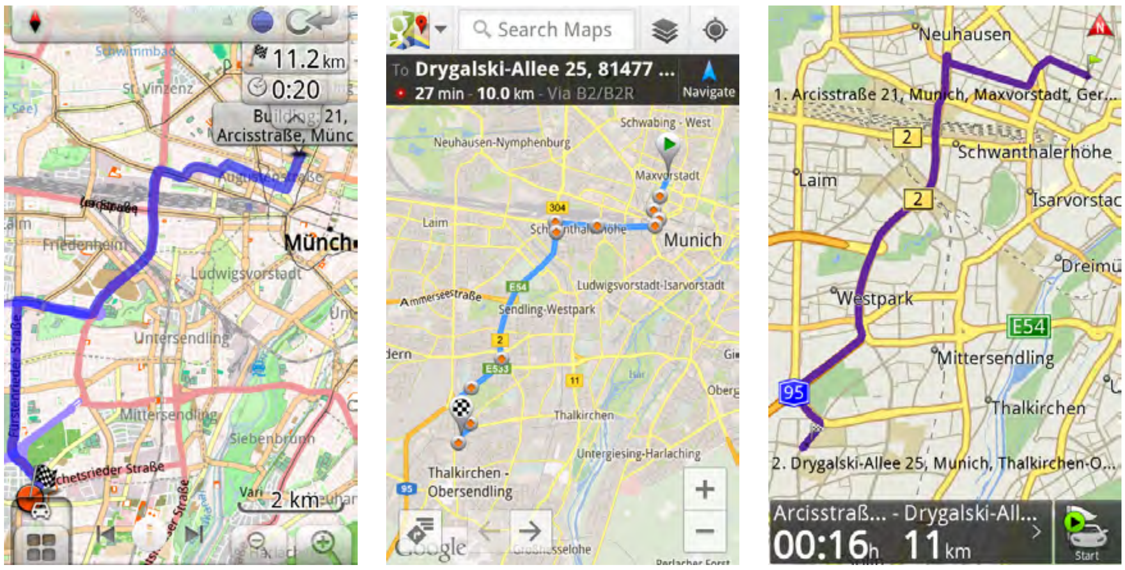
(a) (b) (c)

Figure A.24.: Proposed route from *Arcisstraße 21, Munich* to *Solalindenstraße 52, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


10.5 km
0:17
Building: 21, Arcisstraße, München
München
2 km
Search Maps
To Cincinnatistraße 17, 8154...
26 min · 10.0 km · Via E54
Navigate
Munich
1. Arcisstraße 21, Munich, Maxvorstadt, Germany
Munich
2. Cincinnatistraße 17, Munich, Obergiesing,...
Arcisstraß... - Cincinnatistr...
00:16h 8.0km
Start

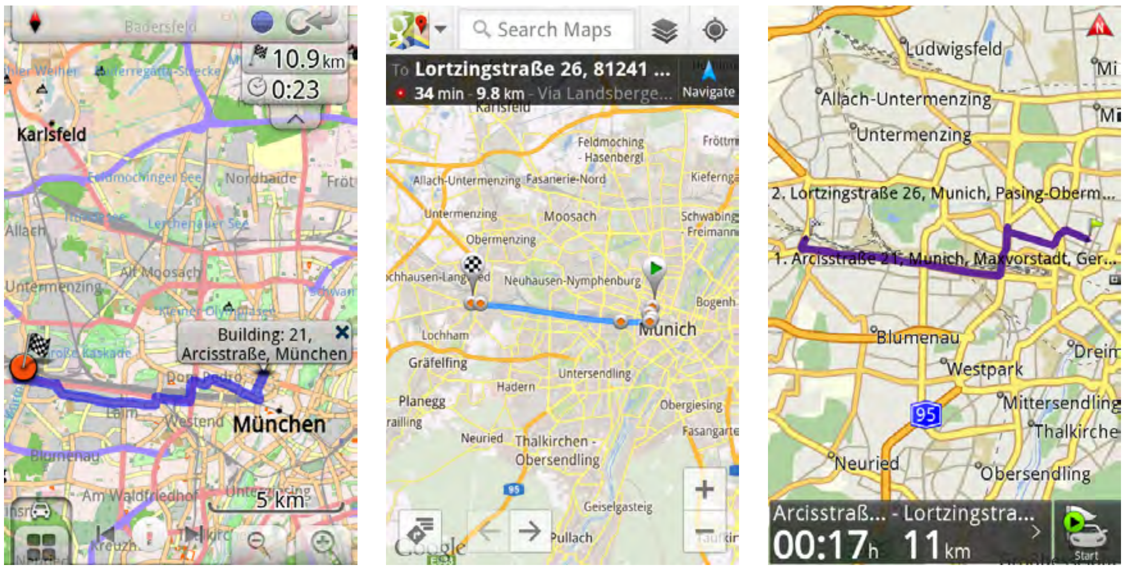
(a) (b) (c)

Figure A.25.: Proposed route from *Arcisstraße 21, Munich* to *Cincinnatistraße 17, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


11.2 km
0:20
Building: 21, Arcisstraße, Münc
2 km
Search Maps
To Drygalski-Allee 25, 81477 ...
27 min · 10.0 km · Via B2/B2R
Navigate
Munich
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
2. Drygalski-Allee 25, Munich, Thalkirchen-O...
Arcisstraß... - Drygalski-All...
00:16h 11km
Start

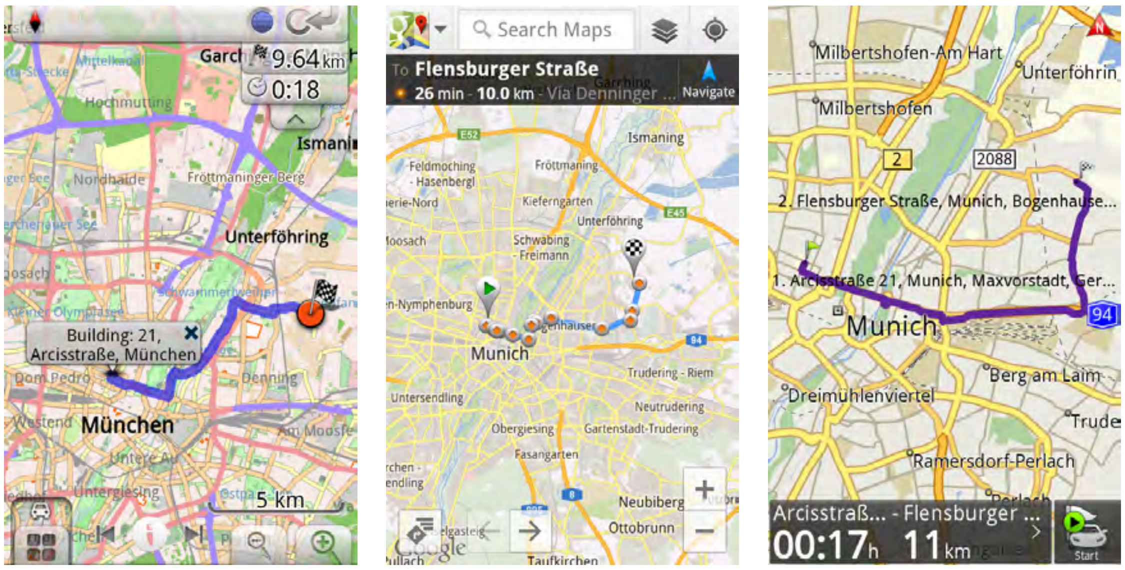
(a) (b) (c)

Figure A.26.: Proposed route from *Arcisstraße 21, Munich* to *Drygalski-Allee 25, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


10.9 km
0:23
Building: 21, Arcisstraße, München
München
5 km
Search Maps
To Lortzingstraße 26, 81241 ...
34 min 9.8 km - Via Landsberge...
Navigate
Munich
2. Lortzingstraße 26, Munich, Pasing-Oberm...
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Arcisstraß... - Lortzingstra...
00:17h 11 km
Start

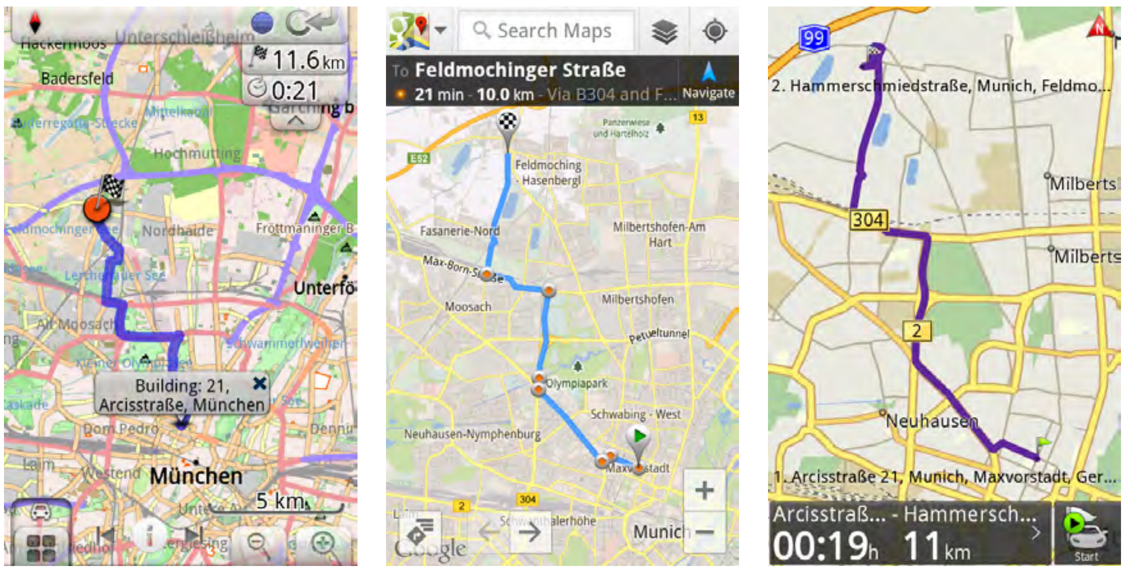
(a) (b) (c)

Figure A.27.: Proposed route from *Arcisstraße 21, Munich* to *Lortzingstraße 26, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


9.64 km
0:18
Building: 21, Arcisstraße, München
München
5 km
Search Maps
To Flensburger Straße
26 min 10.0 km - Via Denninger ...
Navigate
Munich
2. Flensburger Straße, Munich, Bogenhause...
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Munich
Arcisstraß... - Flensburger ...
00:17h 11 km
Start

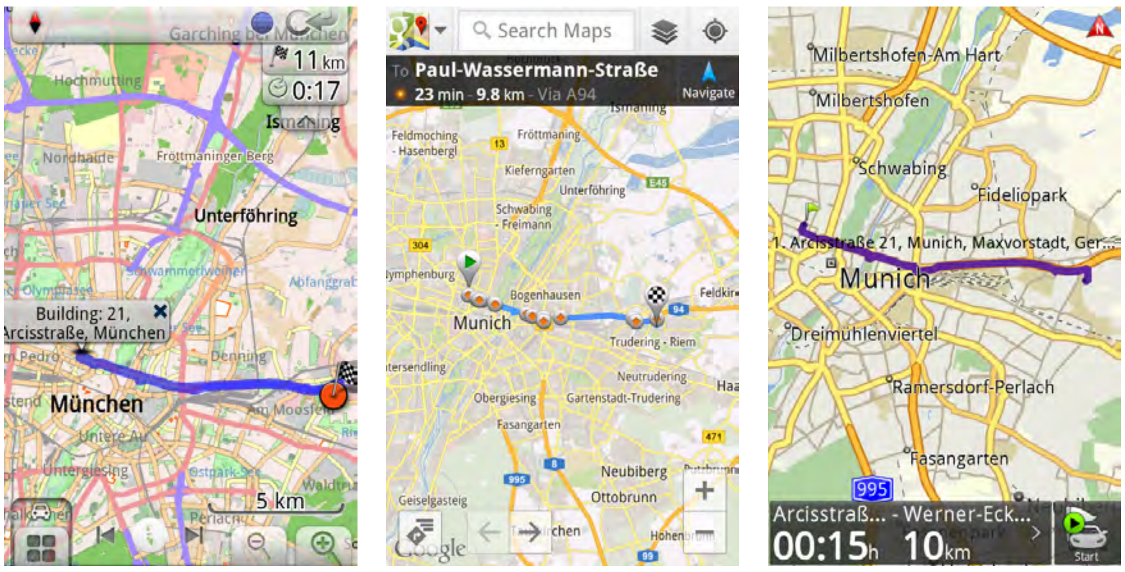
(a) (b) (c)

Figure A.28.: Proposed route from *Arcisstraße 21, Munich* to the intersection *Flensburger Straße / Apenrader Straße, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


11.6 km
0:21
Building: 21, Arcisstraße, München
München
5 km
Search Maps
To Feldmochinger Straße
21 min 10.0 km - Via B304 and F...
Navigate
2. Hammerschmiedstraße, Munich, Feldmo...
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Arcisstraß... - Hammersch...
00:19h 11 km
Start

(a) (b) (c)

Figure A.29.: Proposed route from *Arcisstraße 21, Munich* to the intersection *Feldmochinger Straße / Hammerschmiedstraße, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.


11 km
0:17
Building: 21, Arcisstraße, München
München
5 km
Search Maps
To Paul-Wassermann-Straße
23 min 9.8 km - Via A94
Navigate
1. Arcisstraße 21, Munich, Maxvorstadt, Ger...
Arcisstraß... - Werner-Eck...
00:15h 10 km
Start

(a) (b) (c)

Figure A.30.: Proposed route from *Arcisstraße 21, Munich* to the intersection *Paul-Wassermann-Straße / Werner-Eckert-Straße, Munich* provided by the three different applications: (a) DriveAssist 2.0, (b) Google Maps, (c) Route 66 Maps + Navigation.

List of Figures

List of Tables

List of Acronyms

AA	Activation Announcement
ABS	Anti-Lock Brake System
ACC	Adaptive Cruise Control
ADAS	Advanced Driver Assistance System
ADT	Android Development Tools
AFS	Adaptive Frontlighting System
ASF	Automatic Spelling Function
BSD	Blind Spot Detection
BTS	Base Transceiver Stations
CAM	Cooperative Awareness Messages
CSWS	Curve Speed Warning System
CTS	Central Traffic Services
CW	Collision Warning
DDD	Drowsy Driver Detection
DPP	Dynamic Pass Predictor
ESC	Electronic Stability Control
ESP	Electronic Stability Program
GD	Green Driving
GLONASS	Globalnaya Navigationnaya Sputnikovaya Sistema
GNSS	Global Navigation Satellite System
GPS	Global Positioning System
GUI	Graphical User Interface

HMI	Human Machine Interface
IEEE	Institute of Electrical and Electronics Engineers
IA	Information Announcement
ISA	Intelligent Speed Advisory
ITS	Intelligent Transportation Systems
JDK	Java Development Kit
LBS	Location Based Service
LDW	Lane Departure Warning
LKA	Lane Keeping Assistant
LCA	Lane Change Assistant
NAVSTAR	Navigational Satellite Timing and Ranging
NUI	Natural User Interface
OA	Overtake Assistant
OSM	OpenStreetMap
PDC	Parking Distance Control
PE	Powertrain Efficiency
PND	Portable Navigation Devices
POI	Points of Interest
Pre-IA	Pre-Information Announcement
PTA	Plain Text Announcement
RFID	Radio Frequency Identification
SDK	Software Development Kit
SSSW	Stoplight and Stop Sign Warning
TCS	Traction Control System
TDOA	Time Distance of Arrival
TOA	Time of Arrival
TTS	Text-To-Speech

TUM	Technische Universität München
UI	User Interface
UWB	Ultra-Wideband
VMI	Fachgebiet Verteilte Multimodale Informationsverarbeitung
V2I	Vehicle-to-Infrastructure
V2V	Vehicle-to-Vehicle
V2X	Vehicle-to-X
WLAN	Wireless Local Area Network
YOURS	Yet another OpenStreetMap Route Service

Bibliography

[1] S. Ezell, "Intelligent Transportation Systems," *The Information Technology & Innovation Foundation*, 2010.

[2] B. Williams, *Intelligent Transport Systems Standards*. Artech House Publishers, 2008.

[3] C. Weiß, "V2X Communication in Europe - From Research Projects Towards Standardization and Field Testing of Vehicle Communication Technology," *Computer Networks*, vol. 55, no. 14, pp. 3103–3119, 2011.

[4] S. Diewald, A. Möller, L. Roalter, and M. Kranz, "DriveAssist - A V2X-Based Driver Assistance System for Android," *Mensch & Computer 2012–Workshopband: interaktiv informiert–allgegenwärtig und allumfassend!?*, 2012.

[5] S. Diewald, A. Möller, L. Roalter, and M. Kranz, "Mobile Device Integration and Interaction in the Automotive Domain," in *AutoNUI: Automotive Natural User Interfaces Workshop at the 3rd International Conference on Automotive User Interfaces and Interactive Vehicular Applications (AutomotiveUI 2011)(Nov.–Dec. 2011)*, 2011.

[6] S. Hess, A. Meschtscherjakov, T. Ronneberger, and M. Trapp, "Integrating Mobile Devices into the Car Ecosystem: Tablets and Smartphones as Vital Part of the Car," in *Proceedings of the 3rd International Conference on Automotive User Interfaces and Interactive Vehicular Applications*, pp. 210–211, ACM, 2011.

[7] J. White, C. Thompson, H. Turner, B. Dougherty, and D. Schmidt, "WreckWatch: Automatic Traffic Accident Detection and Notification with Smartphones," *Mobile Networks and Applications*, vol. 16, no. 3, pp. 285–303, 2011.

[8] J. Zaldivar, C. Calafate, J. Cano, and P. Manzoni, "Providing Accident Detection in Vehicular Networks Through OBD-II Devices and Android-Based Smartphones," in *Local Computer Networks (LCN), 2011 IEEE 36th Conference on*, pp. 813–819, IEEE, 2011.

[9] R. Araujo, A. Igreja, R. de Castro, and R. Araujo, "Driving Coach: A Smartphone Application to Evaluate Driving Efficient Patterns," in *Intelligent Vehicles Symposium (IV), 2012 IEEE*, pp. 1005–1010, IEEE, 2012.

[10] J. Tulusan, T. Staake, and E. Fleisch, "Providing Eco-Driving Feedback to Corporate Car

Drivers: What Impact Does a Smartphone Application Have on Their Fuel Efficiency?," in *Proceedings of the 2012 ACM Conference on Ubiquitous Computing*, pp. 212–215, ACM, 2012.

[11] T. Flach, N. Mishra, L. Pedrosa, C. Riesz, and R. Govindan, "CarMA: Towards Personalized Automotive Tuning," in *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, pp. 135–148, ACM, 2011.

[12] S. Diewald, T. Leinmüller, B. Atanassow, L. Breyer, and M. Kranz, "Mobile Device Integration with V2X Communication," in *Proceedings of the 19th World Congress on Intelligent Transport Systems (ITS)*, 2012.

[13] M. Kranz, A. Franz, M. Röckl, A. Lehner, and T. Strang, "CODAR Viewer - a Situation-Aware Driver Assistance System," *Advances in Pervasive Computing Pervasive*, pp. 126–129, 2008.

[14] Y. Chen, D. Zhang, and K. Li, "Enhanced Eco-Driving System Based on V2X Communication," in *Intelligent Transportation Systems (ITSC), 2012 15th International IEEE Conference on*, pp. 200–205, IEEE, 2012.

[15] J. Wedel, B. Schunemann, and I. Radusch, "V2X-Based Traffic Congestion Recognition and Avoidance," in *Pervasive Systems, Algorithms, and Networks (ISPAN), 2009 10th International Symposium on*, pp. 637–641, IEEE, 2009.

[16] K. Reif, *Fahrstabilisierungssysteme und Fahrerassistenzsysteme*. Vieweg+Teubner Verlag, 2010.

[17] K. Reif, *Automobilelektronik: Eine Einführung für Ingenieure*. Vieweg+Teubner Verlag, 2009.

[18] J. Schiller and A. Voisard, *Location-Based Services*. Morgan Kaufmann, 2004.

[19] J. Hatzopoulos, *Topographic Mapping: Covering the Wider Field of Geospatial Information Science & Technology (GIS & T)*. Universal Pub, 2008.

[20] A. Jagoe, *Mobile Location Services: The Definitive Guide*, vol. 1. Prentice Hall PTR, 2003.

[21] P. Zhang, *Advanced Industrial Control Technology*. Elsevier Inc., 2010.

[22] J. Figueiras and S. Frattasi, *Mobile Positioning and Tracking: From Conventional to Cooperative Techniques*. Wiley, 2010.

[23] P. Groves, *Principles of GNSS, Inertial, and Multisensor Integrated Navigation Systems*. 2008.

[24] S. Čapkun, M. Hamdi, and J. Hubaux, "GPS-Free Positioning in Mobile Ad Hoc Networks," *Cluster Computing*, vol. 5, no. 2, pp. 157–167, 2002.

[25] D. Niculescu and B. Nath, "Ad Hoc Positioning System (APS)," in *Global Telecommunications Conference, 2001. GLOBECOM'01. IEEE*, vol. 5, pp. 2926–2931, IEEE, 2001.

[26] S. Edelkamp and S. Schrödl, *Heuristic Search: Theory and Applications*. Morgan Kaufmann, 2011.

[27] H. Winner, S. Hakuli, and G. Wolf, *Handbuch Fahrerassistenzsysteme: Grundlagen, Komponenten und Systeme für aktive Sicherheit und Komfort*. Vieweg+ Teubner Verlag, 2011.

[28] A. Schmidt, A. Dey, A. Kun, and W. Spiessl, "Automotive User Interfaces: Human Computer Interaction in the Car," in *Proceedings of the 28th of the International Conference Extended Abstracts on Human Factors in Computing Systems*, pp. 3177–3180, ACM, 2010.

[29] J. Lumsden, *Human-Computer Interaction and Innovation in Handheld, Mobile and Wearable Technologies*. IGI Global, 2011.

[30] J. Kamp, C. Marin-Lamellet, J. Forzy, and D. Causeur, "HMI Aspects of the Usability of Internet Services with an in-Car Terminal on a Driving Simulator," *IATSS research*, vol. 25, no. 2, pp. 29–39, 2001.

[31] G. Burnett, S. Lomas, B. Mason, J. Porter, and S. Summerskill, "Writing and Driving: An Assessment of Handwriting Recognition as a Means of Alphanumeric Data Entry in a Driving Context," *Advances in Transportation Studies*, 2005.

[32] J. Erjavec, *Automotive Technology: A Systems Approach*. Thomson Delmar Learning, 2005.

[33] M. Emmelmann, B. Bochow, and C. Kellum, *Vehicular Networking: Automotive Applications and Beyond*, vol. 2. Wiley, 2010.

[34] J. Lauffenburger, B. Bradai, A. Herbin, and M. Basset, "Navigation as a Virtual Sensor for Enhanced Lighting Preview Control," in *Intelligent Vehicles Symposium*, pp. 1–6, IEEE, 2007.

[35] S. Durekovic and N. Smith, "Architectures of Map-Supported ADAS," in *Intelligent Vehicles Symposium (IV)*, pp. 207–211, IEEE, 2011.

[36] J. Loewenau, K. Gresser, D. Wisselmann, W. Richter, D. Rabel, and S. Durekovic, "Dynamic Pass Prediction - A New Driver Assistance System for Superior and Safe Overtaking," *Advanced Microsystems for Automotive Applications 2006*, pp. 67–77, 2006.

[37] O. Carsten and F. Tate, "Intelligent Speed Adaptation: Accident Savings and Cost–Benefit Analysis," *Accident Analysis & Prevention*, vol. 37, no. 3, pp. 407–416, 2005.

[38] F. Goodwin, F. Achterberg, and J. Beckmann, "Intelligent Speed Assistance - Myths and Reality: ETSC Position on ISA," *European Transport Safety Council (ETSC), Brussels*, 2006.

[39] H. Lahrmann, J. Madsen, and T. Boroch, "Intelligent Speed Adaptation-Development of a GPS Based ISA-System and Field Trial of the System with 24 Test Drivers," in *8th World Congress on Intelligent Transport Systems*, 2001.

[40] A. van Loon and L. Duynstee, "Intelligent Speed Adaptation (ISA): A Successful Test in the Netherlands," in *Proceedings of the Canadian Multidisciplinary Road Safety Conference XII, Ontario, Canada*, 2001.

[41] R. Driscoll, Y. Page, S. Lassarre, and J. Ehrlich, "LAVIA - An Evaluation of the Potential Safety Benefits of the French Intelligent Speed Adaptation Project," in *Annual Proceedings/Association for the Advancement of Automotive Medicine*, vol. 51, p. 485, Association for the Advancement of Automotive Medicine, 2007.

[42] K. Creef, J. Wall, P. Boland, V. Vecovski, M. Prendergast, J. Stow, R. Fernandes, J. Beck, S. Doecke, and J. Woolley, "Road Safety Benefits of Intelligent Speed Adaptation for Australia," in *Australasian Road Safety Research, Policing and Education Conference, 2011, Perth, Western Australia, Australia*, 2011.

[43] S. Vlassenroot, S. Broekx, J. Mol, L. Panis, T. Brijs, and G. Wets, "Driving with Intelligent Speed Adaptation: Final Results of the Belgian ISA-trial," *Transportation Research Part A: Policy and Practice*, vol. 41, no. 3, pp. 267–279, 2007.

[44] S. Ghadiri, J. Prasetijo, A. Sadullah, M. Hoseinpour, and S. Sahranavard, "Intelligent Speed Adaptation: Preliminary Results of On-Road Study in Penang, Malaysia," *IATSS Research*, 2012.

[45] O. Carsten, F. Tate, and O. Carsten, "Final Report: Integration," *Institute for Transport Studies, University of Leeds*, 2000.

[46] S. Glaser, L. Nouveliere, and B. Lusetti, "Speed Limitation Based on an Advanced Curve Warning System," in *Intelligent Vehicles Symposium, 2007 IEEE*, pp. 686–691, IEEE, 2007.

[47] A. Amditis, A. Polychronopoulos, L. Andreone, and E. Bekiaris, "Communication and Interaction Strategies in Automotive Adaptive Interfaces," *Cognition, Technology & Work*, vol. 8, no. 3, pp. 193–199, 2006.

[48] A. Amditis, L. Andreone, K. Pagle, G. Markkula, E. Deregibus, M. Rue, F. Bellotti, A. Engelsberg, R. Brouwer, B. Peters, *et al.*, "Towards the Automotive HMI of the Future: Overview of the AIDE-Integrated Project Results," *Intelligent Transportation Systems, IEEE Transactions on*, vol. 11, no. 3, pp. 567–578, 2010.

[49] W. Piechulla, C. Mayser, H. Gehrke, and W. König, "Reducing Drivers' Mental Workload by Means of an Adaptive Man-Machine Interface," *Transportation Research Part F: Traffic Psychology and Behaviour*, vol. 6, no. 4, pp. 233–248, 2003.

[50] S. Coast, "How OpenStreetMap is Changing the World," in *Proceedings of the 10th International Conference on Web and Wireless Geographical Information Systems*, W2GIS'11, (Berlin, Heidelberg), pp. 4–4, Springer-Verlag, 2011.

[51] J. Bennett, *OpenStreetMap*. Packt Pub., 2010.

[52] R. Meier, *Professional Android 4 Application Development*. Wrox Professional Guides, John Wiley & Sons, 2012.