Campus Navigation System with Seamless Real-Time Information

Yueshi Wu, Daniel Raviv, and Mihaela Cardei
Department of Computer and Electrical Engineering and Computer Science
Florida Atlantic University
Boca Raton, FL 33431, USA
(wuy2013, ravivd, mcardei)@fau.edu

Abstract

In this paper we present a system for navigation and real-time information that can be used in university campuses. The system incorporates optimal route selection for both pedestrians and drivers with seamless integration into mapping software. In addition, it provides efficient, real-time information about parking spot availability in the driver’s vicinity. As a case study, the system is implemented as an Android app that can significantly reduce campus visitors confusion and time consumption.

keywords: campus navigation, Android OS, mobile app, Google map.

1 Introduction and Related Work

Recent technological advancements are finding many useful applications in everyday activities. For example, numerous devices and applications assist users navigate to desired locations [6]. A navigation service is widely used in our daily life and indispensable when traveling to a new place. Vehicle navigation systems and smartphones are the two most commonly used navigation devices. Global Positioning System (GPS) is usually utilized to provide users’ current position and to allow path calculation to a specific destination. While many vehicle navigation devices are not portable, ubiquitous smart phones nowadays are suitable for providing navigation for both driving and walking.

University campuses can be large and scattered, making it difficult for new students and visitors to reach specific destinations on campus. For example, at the Florida Atlantic University (FAU) Boca Raton campus, people are sometimes told to go to room 220 in building SU-80. This is a very ambiguous information for a new student. Even when receiving directions, a new student may not easily find the destination because these directions involve knowing the surrounding buildings and landmarks of the campus. In some cases, people need to visit a specific faculty or staff member when they don’t know which building or office they need to go.

The situation is more complicated if the visitor is driving to campus. Campus parking lots are usually divided into different zones where different permissions are required. Even when people are told to go to a specific parking lot, they do not know the exact location of that parking lot. Even more, due to the limited parking space on campus, finding available parking with correct user permission can be time consuming and frustrating.

To address the issues stated above, we developed a Campus Navigation System (CNS). CNS provides seamless navigation around campus for both walking and driving directions. The user has the option to select a building, parking, or a particular person (e.g. a professor) as the destination. If the user selects the option of driving to the closest parking to the destination, he will be directed to the parking lot with available parking spaces, which is nearest to his/her chosen destination.

Android is one of the largest and fastest growing
mobile platforms [1]. As of 2015, it has the largest installed base of all general-purpose operating systems [5]. Android is a Linux kernel based open source platform with versatile development and debugging environment. The fact that Google has released its code as open source triggered a large community of developers to develop applications that extend the functionality of the device using a customized version of Java. In this paper we develop our campus navigation system based on the Android platform.

Most related works applications developed provide campus information in a static manner. FAU Mobile Application [2] has integrated campus map option where the detailed campus map of FAU is displayed as well as the current location of the user, while navigation is not supported. Another similar project is the Interactive Map Project [3] from The University of California at Santa Barbara. Their map allows users to select buildings to zoom into and locate, as well as find a certain room within a building. However, this project does not support navigation.

Mapyst Project [4] from Carnegie Mellon University provides navigation and other information about the campus, for example, the operating hour of food courts. Another application is the University of California, San Diego campus map. This map is accessed through a web browser and allows users to select a source location and a destination. When this information is submitted, the shortest path is outlined on the map, and the distance and expected walking time are shown [8]. These two applications [3][4] do not provide driving directions within the campus. Providing driving directions is important for a campus navigation system since there are usually more restrictions when driving on campus, and most of the visitors use some form of transportation to enter the campus.

2 The System Architecture

As illustrated in Figure 1, CNS consists of two parts: the mobile front end and the server back end. The mobile front end is implemented as a mobile application that runs on users’ smart phone. The mobile frontend takes user request i.e. source and destination, being either a person or building and displays the computed route. The server back end consists of a mapping server where a web-based map editor tool is stored and a parking server where automatic parking lot occupancy tracking system is integrated.

2.1 Workflow

In this section, we describe the steps performed by the CNS when a user requests directions to a building while driving, with the parking option on. The way in which mobile frontend, mapping server and parking server interact with each other and how the CNS works is illustrated in Figure 2. The CNS:

- Prompts the user for information: user type (student, staff, or visitor), campus (e.g. Boca Raton, Davie), source (current location or from the default location list), and destination (building, parking, or person).
- Downloads the campus map XML file from the Mapping Server using HTTP.
- Parses the campus map XML file and builds the campus map graph data structure.
- Downloads parking information for the specified campus.
- Calculates the shortest path from the source to the destination.
• Periodically inquires the parking server to check if the current destination (parking lot) is still available. If not, the CNS sets the destination to the closest available parking and calculates the shortest path.

2.1.1 The Mapping Server

MapEditor is a web-based tool designed to edit and manage campus maps and is stored on the Mapping Server. The campus maps managed by the MapEditor tool are stored on the server as XML files. A detailed description of our MapEditor tool can be found in [6].

We represent a map as a graph where vertices describe locations such as buildings, parking lots, and intermediary segment ends and edges describe walkable and drivable segments between vertices. The UML class diagram is illustrated in Figure 3. All objects except LatLng inherit the CampusMapObject with an attribute id.

The Vertex class has two attributes, name and abbreviation, and is used to represent a location at a basic level. A vertex indicates its location using the LatLng object that stores the latitude and longitude. The WalkSegmentEnd object that inherits the Vertex represents a location that can be reached only by walking, while a RoadSegmentEnd object represents locations that are directly reachable by car. We made the RoadSegmentEnd a subclass of WalkSegmentEnd since locations accessible by car are also accessible by foot on campus.

The Building class models campus buildings and the Parking class holds the information for a parking lot on the campus. Parking has the attribute permit type since our university issues different permit types for faculty and staff, students, and visitors. A Segment object represents an edge in the map graph. The cost attribute is the distance in meters between two vertices it connects.

The direction attribute indicates whether a segment is navigable in one direction, end1 \(\rightarrow\) end2, in reverse, or in both directions. The WalkSegment class inherits from the Segment and the RoadSegment inherits from the WalkSegment. A WalkSegment between two locations can be navigated only by walking while a RoadSegment can be navigated both by walking and driving.

2.1.2 The Parking Server

The parking server in the CNS is used to maintain the occupancy of each parking lot on the campus. The CNS is integrated with our automatic parking lot occupancy tracking system [7]. This system requires the use of a camera to monitor the target parking lot. The parking occupancy monitoring application captures images in real time from the camera.

The algorithm in [7] applies motion tracking to process images. Internally the proposed algorithm performs blob tracking, where the blobs in the images are classified as cars based on predefined criteria. Then, based on the movement of blobs in the video (transitions in space), the proposed algorithm is able to count the number of cars that enter/leave the parking lot, and it keeps track of the number of cars in a given parking lot.

The occupancy information is stored in a database, that is queried by the CNS as needed. The detailed explanation of the automatic parking lot occupancy tracking system can be found in [7]. The data obtained from the tracking system is stored in a MySql database. The Parking table in the database stores the occupancy of targeted parking lots in different campuses. The structure of the Parking table is shown in Figure 4.

2.1.3 The People Directory Server

We designed a Visiting Management System (VMS) for professors and staff members, allowing visits during certain times. The VMS is implemented with JavaScript and MySql database. Professors and staff members have an account in the VMS where they can log in to change the allowed visitation time or turn on/off visitations. The default visiting time and location is set to the office hours and the office of each individual professor.

The default allowed visitation is set to off - the
professors’ location is not accessible from the CNS. Professors can turn on the visitation option and add specific locations and time periods. The allowed visitation time period and location are stored in the MySQL database, which is queried by the CNS as needed. VMS is stored in the People Directory Server and can be accessed from smartphones and PCs using any web browser.

2.1.4 The Mobile Frontend

The mobile frontend of our system is implemented as an Android app. The app gets user input and displays the route in real time. We implemented the app in Java using the Android SDK. As a development platform we use the Eclipse IDE with the ADT plugin for editing the files and managing the project.

Android API comes with many useful packages that allow access to a broad functionality of Android devices. Examples are the classes for the user interface and for accessing the compass and the GPS functionality. The mobile frontend interacts with different servers to provide seamless navigation for users. The detailed steps are outlined as follows:

(1) First, the user interface gets user information including user type, campus, source, and destination. User type includes student, staff, and visitor and it has to be specified accurately in order to get the correct parking direction. The source can be any location chosen from the system’s default list or it can be the user’s current location. If the user does not select his current location as the source, then CNS will display the static path from the chosen source to the destination. The destination can be a building, parking, or person working at FAU (e.g. a professor). CNS currently provides a list of professors from the Department of Computer and Electrical Engineering and Computer Science. If a professor allows visitations, then his/her name will show up in the destination list during his/her allowed time period. This is done by inquiring the database on the People Directory Server. The user also has the option to select walking or driving to the destination. By selecting both driving and guide to parking, the application calculates a path to the nearest parking lot to the destination, that has free parking spots corresponding to the user type.

(2) The app downloads the XML file of the selected campus map from the Mapping Server and parses it to create the map graph data structures. Document Object Model (DOM) is used to perform parsing. We build data structures storing campus map objects and their attributes, such as vertices (e.g. Buildings, Parking lots, etc.) and segments (e.g. road segments, walking segments).

(3) The destination validation step is performed only when the user requests directions to a parking. The parking data is downloaded from the Parking Server using HTTP. We construct a candidate list of parking lots. A parking lot is added to the list if it is compatible with the user type (e.g. if the
user is allowed to park there) and if the number of available spots is greater than 0. The parking lot from the candidate list with the smallest Euclidean distance to the destination will be selected as the destination. The frontend periodically inquires the Parking Server to check the number of spots available. If the number of spots available becomes 0, then the app downloads the updated parking data and performs the same procedure as before to update the destination parking lot.

(4) The app computes the shortest-path between the source and the destination using Dijkstra’s algorithm [9]. The user has the option to choose between driving and walking navigation. Only road segments are considered for the shortest path computation when driving direction is requested. Both road segments and walk segments are used for shortest path computation when walking direction is requested. The weight of a segment is the Euclidean distance, which is one of the attributes (e.g. denoted as the cost) of a driving/walk segment. The Dijkstra’s algorithm stores the vertices to which the shortest-path has not been computed yet in a minimum-priority queue. At each step, one vertex with the minimum weight is removed. In order to reduce the run time complexity, we only keep the next step candidate vertices in the minimum-priority queue. Also our algorithm ends as soon as the shortest-path to the destination has been computed. The shortest path is then displayed on the user interface overlaid on top of the campus Google Map in map or satellite view.

(5) CNS does not provide real time indoor navigation at this stage since the buildings in the university campus are usually not as large and confusing as airports or shopping malls. If a user requests directions to a person, then the application first provides real time navigation to the building. Once the user gets close enough to the destination building, the floor plan of that building automatically pops out. The user is able to view the floors by selecting a floor number from the floor picker. Our floor plan is implemented as GroundOverlay in the Google Map API.

3 CNS Functionality

We demonstrate the functionality of our CNS system in this section. The mobile frontend application is installed on a smart phone Motorola Moto G with android version 4.4.4. We first demonstrate the scenario where a new student driving a car needs to go to the Engineering & Computer Science building.

The current location of the user is indicated with a blue dot. Our system first shows the driving direction to the parking lot 96, which is a student parking with available spaces, located nearest the Engineering & Computer Science building (Figure 5a).

Once the user arrives at the parking lot, he/she can simply switch to the walking mode and the walking path to the destination building is shown (Figure 5b). When the user is approaching the entrance, our application automatically zooms in and shows the floor plan of the destination building. The user is able to choose and view different floors via the floor picker (Figure 5c).
Figure 6 shows walking directions for a user to the Administration building. From the satellite view we can see that the app considers small alleys and paths when calculating the walking path. This provides a nice feature for campus navigation for walking since it computes shorter path compared to other existing general purpose maps which do not consider small alleys or walking paths through buildings.

4 Conclusions and Future Work

In this paper we presented a Campus Navigation System developed for the Android platform. Our application is able to provide seamless driving and walking navigation around typical university campuses and will improve the campus experience of students, staff, professors, and visitors. In the future we plan to enhance CNS with robust indoor navigation and weighted shortest path navigation. We plan to use a more sophisticated cost function for driving directions that besides the Euclidean distance accounts for the number of turns, traffic signs on the path, the speed limit, and so on.

References


