ISSN: 2278 – 7798
International Journal of Science, Engineering and Technology Research (IJSETR)
Volume 1, Issue 6, December 2012

Path Finder And Dictator – GPS Navigation Using TTS Path Algorithm In Smart Phone

Mrs. Vaishali L. Kolhe
Department of Computer Engineering,
DYPCOE, Akurdi, Pune.

*Abstract— **Initially mobile phones were developed only for voice communication but now days the scenario has changed, voice communication is just one aspect of a mobile phone. There are other aspects which are major focus of interest. Two such major factors are web browser and GPS services. Global Positioning System (GPS) navigation is a popular assistant during a trip. By using a GPS navigation system, the travelers can easily and quickly arrive to the destination in an unfamiliar area. However, there is no free and open-source GPS navigation system which integrates many useful applications. Thus, this paper proposes a GPS navigation system on the Android based open source mobile phone, called Path Finder Application. Path Finder not only provides users the GPS navigation function, but also supports quick response in convenient audible form and source to destination positioning. Furthermore, this application is free and open-source software, so service providers or developers can easily extend their own services on this system.***

***Index Terms* — GPS, Android, TTS, Shortest Path Algorithm, Google map.**

I. INTRODUCTION

Smart phones have become ubiquitous as newer less expensive models with greater features sets have been released. Android is an open source operating system, freely available to develop new ideas. Nowadays mobile systems are more than that of communicating tool. The most recent and useful application is regarding the use of browsers and GPS navigation system.

This application uses GPS navigation for finding path from source to destination. While reaching to destination this application provides the proper directions and dictate the way. It helps blind peoples in finding the path. In order to get the proper destination path, the path finder provides actual turns and the distance from the destination location.

II. BACKGROUND

A. GPS Technology

The Global Positioning System (GPS) is a global Navigation satellite system deployed by the US Department of Defense and maintained by the US Air Force. GPS is a space based radio navigation system that provides accurate location and timing services to anyone with a GPS receiver. This service, made available to civilians in 1996 for navigation purposes, is free of charge, can support an unlimited number of users, and functions anywhere in the world. Starting in 2004, the mobile phone industry began successful tests to incorporate GPS receivers into mobile phone devices to support 911 emergency locations. Most of today's smart phones are equipped with fully functional GPS receivers and supporting applications.

Fig 1. Location finding with GPS

B. Android Software

The Apple iPhone has transformed the smart phone's image from a corporate-level personal organizer to a device that could potentially benefit every consumer. Recently, Google released an alternative Operating System (OS) and Application Programming Interface (API) for mobile phones called Android. Android joins iPhone OS and other smart phone platforms including Symbian OS, Blackberry and Windows Mobile. Android is backed by the Open Handset. Alliance (OHA), whose members include Sony, Samsung, Motorola and NVIDIA.

Though these companies are relatively new to the consumer-level smart phone market, they have already shown that by melding multiple technologies together in an open manner some unique applications can result.

Fig 2. Architecture of Android

On a basic level, Android is a distribution of Linux that includes a Java Virtual Machine (JVM), with Java being the preferred programming language for most Android applications. The Android Software Development Kit (SDK)[9] includes a debugger, libraries, a handset emulator, documentation, sample code and tutorials. Android's official integrated development environment is Eclipse using the Android Development Tools (ADT) plug-in. SQLite database support is integrated into the Android platform. The ADT plugin includes an Android emulator that allows for the simulation of GPS and Wi-Fi. The Android emulator is depicted in Fig. 2 displaying the Android desktop.

C. TTS Algorithm

A Text-To-Speech (TTS) synthesizer is a computer-based system that should be able to read *any* text aloud, whether it was directly introduced in the computer by an operator or scanned and submitted to an Optical Character Recognition (OCR) system. There is a fundamental difference between this system and any other talking machine (as a cassette-player for example) in the sense that the automatic production of new sentences is used here.

This definition still needs some refinements. Systems that simply concatenate isolated words or parts of sentences, denoted as *Voice Response Systems*, are only applicable when a limited vocabulary is required (typically a few one hundreds of words), and when the sentences to be pronounced respect a very restricted structure, as is the case for the announcement of arrivals in train stations for instance. In the context of TTS synthesis, it is impossible (and luckily useless) to record and store all the words of the language. It is thus more suitable to define Text-To-Speech as the automatic production of speech, through a grapheme-to-phoneme transcription of the sentences to utter.

- *Speech synthesis*

It is the artificial production of human speech. A computer system used for this purpose is called a speech synthesizer, and can be implemented in software or hardware. A text-to-speech (TTS) system converts normal language text into speech; other systems render symbolic linguistic representations like phonetic transcriptions into speech.

Synthesized speech can be created by concatenating pieces of recorded speech that are stored in a database. Systems differ in the size of the stored speech units; a system that stores phones or diaphones provides the largest output range, but may lack clarity. For specific usage domains, the storage of entire words or sentences allows for high-quality output. Alternatively, a synthesizer can incorporate a model of the vocal tract and other human voice characteristics to create a completely "synthetic" voice output.

The quality of a speech synthesizer is judged by its similarity to the human voice and by its ability to be understood. An intelligible text-to-speech program allows people with visual impairments or reading disabilities to listen to written works on a home computer. Many computer operating systems have included speech synthesizers since the early 1980s.

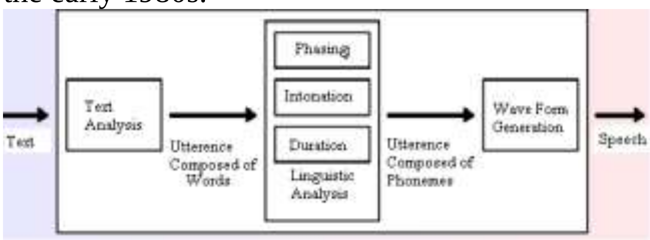

Fig 3. Overview of speech synthesis

A text-to-speech system (or "engine") is composed of two parts: a front-end and a back-end. The front-end has two major tasks. First, it converts raw text containing symbols like numbers and abbreviations into the equivalent of written-out words. This process is often called text normalization, pre-processing or tokenization. The front-end then assigns phonetic transcriptions to each word, and divides and marks the text into prosodic units, like phrases, clauses and sentences. The process of assigning phonetic transcriptions to words is called text-to-phoneme or grapheme-to-phoneme conversion. Phonetic transcriptions and prosody information together make up the symbolic linguistic representation that is output by the front-end. The back-end—often referred to as the synthesizer—then converts the symbolic linguistic representation into sound. In certain systems, this part includes the computation of the target prosody (pitch contour, phoneme durations), which is then imposed on the output speech.

D. Dijkstra's Algorithm

Let the node at which we are starting be called the initial node. Let the distance of node Y be the distance from the initial node to Y. Dijkstra's algorithm will assign some initial distance values and will try to improve them step by step.

1. Assign to every node a tentative distance value: set it to zero for our initial node and to infinity for all other nodes.
2. Mark all nodes except the initial node as unvisited. Set the initial node as current. Create a set of the unvisited nodes called the unvisited set consisting of all the nodes except the initial node.
3. For the current node, consider all of its unvisited neighbors and calculate their tentative distances. For example, if the current node A is marked with a distance of 6, and the edge connecting it with a neighbor B has length 2, then the distance to B (through A) will be 6+2=8. If this distance is less than the previously recorded distance, then overwrite that distance. Even though a neighbor has been examined, it is not marked as visited at this time, and it remains in the unvisited set.
4. When all the neighbors of the current node are considered, mark the current node as visited and remove it from the unvisited set. A visited node will never be checked again; its distance recorded now is final and minimal.
5. The next current node will be the node marked with the lowest (tentative) distance in the unvisited set.
6. If the unvisited set is empty, then stop. The algorithm has finished. Otherwise, set the unvisited node marked with the smallest tentative distance as the next "current node" and go back to step 3.

Evaluation:

An upper bound of the running time of Dijkstra's algorithm on a graph with edges E and vertices V can be expressed as a function of |E| and |V| using Big-O notation. For any implementation of vertex set Q the running time is $O(|E|.dk_Q + |V|.em_Q)$ where dk_Q and em_Q are times needed to perform decrease key and extract minimum operations in set Q, respectively. The simplest implementation of the Dijkstra's algorithm stores vertices of set Q in an ordinary linked list or array, and extract minimum from Q is simply a linear search through all vertices in Q. In this case, the running time is $O(|V|^2 + |E|) = O(|V|^2)$. For sparse graphs, that is, graphs with far fewer than $O(|V|^2)$ edges, Dijkstra's algorithm can be implemented more efficiently by storing the graph in the form of adjacency lists and using a binary heap, pairing heap, or Fibonacci heap as a priority queue to implement extracting minimum efficiently. With a binary heap, the algorithm requires O((|E| + |V|) log|V|) time (which is dominated by O(|E|log|V|), assuming the graph is connected). To avoid O(|V|) look-up in decrease-key step on a vanilla binary heap, it is necessary to maintain a supplementary index mapping each vertex to the heap's index (and keep it up to date as priority queue Q changes), making it take only O(log|V|) time instead. The Fibonacci heap improves this to O(|E| + |V|log|V|). Note that for Directed acyclic graphs, it is possible to find shortest paths from a given starting vertex in linear time, by processing the vertices in a topological order, and calculating the path length for each vertex to be the minimum length obtained via any of its incoming edges.

E. Google Maps

Native map supports to create a range of map based application that leverages the mobility of android devices. Android lets you create activities that include interactive Google maps as part of user interface with full access to maps that can be controlled programmatically and annotated using androids rich graphics libraries. Android provides a number of objects to handle maps in LBS system map view which displays maps.

Fig 4. Google Maps for Mobile

In 2006, Google introduced a Java application called Google Maps for Mobile, intended to run on any Java-based phone or mobile device. Many of the web-based site's features are provided in the application.

On November 28th 2007, Google Maps for Mobile 2.0 was released. It introduced a GPS-like location service that does not require a GPS receiver. The "my location" feature works by utilizing the GPS location of the mobile device, if it is available. This information is supplemented by the software determining the nearest wireless networks and cell sites. The software then looks up the location of the cell site using a database of known wireless networks and cell sites. The Cell-site location method is used by triangulating the different signal strengths from different cell transmitters and then using their location property (retrieved from the online cell site database) to aid My Location in determining the user's current location. Wireless network location method is calculated by discovering the nearby Wi-Fi hotspots and using their location property (retrieved from the online Wi-Fi database, in the same way as the cell site database) to further discover the user's location. The order in which these take precedence is:

- GPS-based services
- *WLAN-, Wi-Fi based services*
- Cell transmitter-based services

The software plots the streets in blue that are available with a yellow icon and a green circle around the estimated range of the cell site based on the transmitter's rated power (among other variables). The estimate is refined using the strength of the cell phone signal to estimate how close to the cell site the mobile device is. As of December 15, 2008, this service was available for the following platforms:

- Android
- iOS (iPhone, iPod Touch, iPad)
- Windows Mobile (NOT Windows Phone 7 as of December 17, 2010)
- Nokia/Symbian (S60 3rd edition only)
- Symbian OS (UIQ v3)
- BlackBerry
- *Phones with Java-Platform (MIDP 2.0 and up), for example the* Sony Ericsson K800i

Fig 5. Map view from Google Map

On November 4, 2009, Google Maps Navigation was released in conjunction with Google Android OS 2.0 Éclair on the Motorola Droid, adding voice commands, traffic reports, and street view support. The initial release was limited to the United States. The service was launched in the UK on 20 April 2010 and in large parts of continental Europe on June 9th 2010 (including Austria, Belgium, Canada, Denmark, France, Germany, Italy, the Netherlands, Portugal, Spain, and Switzerland).

Google Maps for Android

Cell phones are being increasingly used for navigation assistance. Google Maps Navigation for Android 2.0 is free.

Features provided in the application:

- Search in plain English
- Search by voice
- Traffic view
- Search along route
- Satellite view
- Street View
- Car dock mode

III. System Design

Path Finder And Dictator is a 3-tier solution in which the user interface, the business logic and the data management are developed and maintained as independent modules.

User Interface: Top layer of the application, the interface main function is to display information received from other tiers in a user-friendly format.

Business Logic: This middle layer coordinates the application by handling and processing information exchanges between the database and the user interface, and by making logical decisions and performing calculations.

Data Management: Bottom layer of the application, the data management consists of a database server that stores all system's data. It keeps data independent from application servers or business Logic.

This system will be implemented using android operating system. Fig. 6 shows the System Architecture.

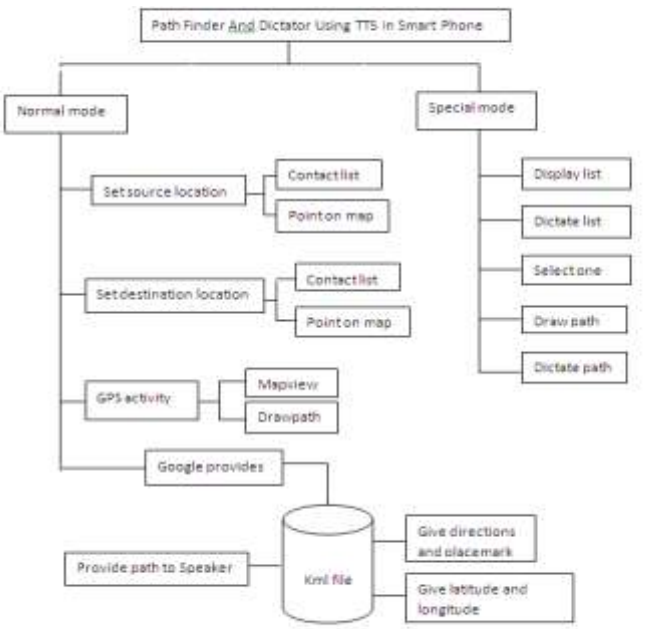

Fig 6. System Architecture of PFD

Path finder provides facility of finding correct destination. The user has to enter source location and destination location to the user interface. Locations are validated by the system. GPS takes the source and destination location and provides latitude and longitude values for that. And it provides a snapshot of map by using Google map. With that map snapshot, user can move with respective direction dictated by the path finder. Kml file will be provided by the Google. Developer has to use this file for getting directions, latitude and longitude. To get the direction, kml file between two tags named placemark can be used. Also It can be used to get latitude and longitude values between two tags named LookAt. These values are stored for further use. This direction is internally converted into audible format by using the Android SDKs.

In order to design the mobile application for smart phones, the android SDKs and GPS navigation with Google Maps returns the particular information that the system requires. The Shortest Path Algorithm that is Dijkstra's algorithm provides shortest path internally using Google maps. The directions with other related information comes to the inbuilt TTS SDK of Android and it converts the text into the speech, which provides user has proper direction that he wants.

IV. System Functionality

Fig. 7 shows the user interface where user can go for Normal mode or Special mode. Special Mode is useful for blind people as some paths are saved and user can use it.

Fig 7. Choice Mode Form

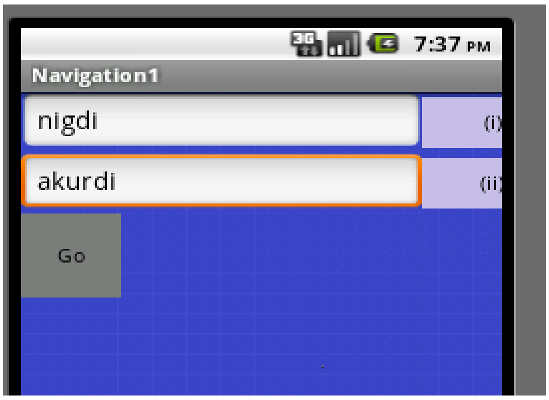

Fig 8. User Form

User has to enter source location in first box and destination location in second box. Fig. 8 shows that the source is set as 'Akurdi' and destination is set as 'Nigdi'. Fig 9. shows that map is displayed with path from source and destination.

Fig 9 Map displaying path from ‘Akurdi’ to ‘Nigdi’

V. Conclusion and Future Scope

This paper proposes a navigation system in smart phones for finding and dictating path in audible format. It provides many services such as searching destination, its distance with proper position, convenient use as it is in audible format.

As android is an open source, this application can be used for further improvements in navigation system. Also this application can be developed by taking source in voice format. This application can be extended in vehicle application. Also electronic digital maps can be used for finding path and dictator machines.

VI. References

[1] Sandeep Kumar, Mohammed Abdul Qadeer, Archana Gupta, “Location Based Services using Android” , IEEE 2009.

[2] Yuan-Cheng Lai, Frannie Han, Yi-Hsuan Yeh, Ching-Neng Lai, Yu-Chin Szu , “A GPS Navigation System with QR Code Decoding and Friend Positioning in Smart Phones”, IEEE 2nd International Conference on Educational Technology and Computer (ICETC) 2010.

[3] J.S.Krishna,A. Shuai,S.Yuan,K.Cai,and Y. Qin, ”A-pragmatic GIS-oriented ontology for location based Services,” 19th Australliann Conference on Software Engineering,2008,pp.562-569.

[4] Y.Liu, Y. Xu, Z. Gui, J. Zheng, and S. Qin, “Feature-based two level structure road network model for navigation,” IEEE International Geoscience and Remote Sensing Symposium,2005.

[5] Romain Prudon & Christophe’s Alessandro, “A selection/concatenation text-to-speech synthesis system: database development, system design, comparative evaluation”, http://www.isca-speech.org/archive [Fourth ICSAITRW on Speech Synthesis(SSW) perthshire,Scotland August 29-Sept 1, 2001].

[6] US Government, Global Positioning System, http://gps.gov/ [last accessed on Mar 1,2009].

[7] 3G, Assisted-GPS Test Calls for 3G WCDMA Networks, http://www.3g.co.uk/PR/November2004/8641. html [last accessed on Mar 1,2009].

[8] John Whipple, William Arensman, Marian Starr Boler, “A Public Safety Application of GPS-Enabled Smartphones and the Android Operating System”, IEEE International Conference on Systems, Man, and Cybernetics San Antonio, TX, USA - October 2009.

[9] Android Developers, http:/ /developer.android.comlindex.html

[10] Y.Liu, Y. Xu, Z. Gui, J. Zheng, and S. Qin, “Feature-based two level structure road network model for navigation,” IEEE International Geoscience and Remote Sensing Symposium, 2005.