ISSN 1598-0170 (Print)
ISSN 2287-1136 (Online)
http://www.jksii.or.kr

유비쿼터스 주차관리 시스템에서 내장 맵 및 센서를 이용한 인프라 독립 네비게이션 시스템☆

Infrastructure-independent Navigation System Using Embedded Map and Built-in Sensors in the Ubiquitous Parking Management

프랭크 엘리호데* 이 재 완**
Frank I. Elijorde Jaewan Lee

요 약

오늘날 사용하고 있는 네비게이션의 신뢰성은 기술적인 발전을 통해 상당히 높아졌다. GPS는 위성기반 위치추적 시스템으로 가장 광범위하게 사용되는 기술이다. 하지만 GPS기반 시스템은 위성이 정확한 뷰를 제공해줄 때에만 위치추적이 정확하다. 본 연구에서는 추적을 위해 내부구조에 의존하지 않는 독자적인 네비게이션을 제안한다. 스마트폰의 내장센서와 내장 맵을 사용하여 정확한 차량 위치추적을 구현한다. 성능평가 결과 정확한 차량의 위치 지원 면에서 우리가 제안한 시스템이 GPS보다 성능이 우수함을 나타내었다

ABSTRACT

Significant advancements in technology enhanced the reliability of navigation systems that are in use today. The GPS is the most widely used technique for satellite-based location estimation. However, systems based on GPS can only be accurate in providing location data when there is a clear view of the satellites. This paper proposes a self-contained navigation system that does not depend on any tracking infrastructure. Using the built-in sensors of a smartphone and a self-contained map, we implemented an accurate car locator. Evaluation results show that our proposed system outperforms GPS in providing accurate car location assistance.

☞ keyword : Navigation System, Self-contained Localization, Car Locator, Built-in Sensors

1. Introduction

For several years, navigation systems have been the subject of numerous experimentations, even long before the discovery of the techniques and technology that would make it more reliable. The advancements in the design of navigation device have greatly improved the efficiency among areas that heavily depend on location-awareness. This is extremely prevalent in applications that rely on GPS such as automotive navigation systems and personal navigation assistants. As a matter of fact, modern navigation approaches rely primarily on positions determined electronically by receivers collecting information from satellites. However, such approach requires navigation systems to be frequently fed with location data.

Mobility has been an important part of carrying out the majority of human activities. This part of human nature has led to countless innovations such as context-aware applications which often rely on information about the user's location. Trend has it that the number of location-based applications has been growing exponentially in the last few years. With the emergence and massive production of smart phones that has location capabilities, many location-aware applications have been developed.

A number of localization technologies have been proposed for cellular phones. Expectedly, the Global Positioning System (GPS) [1] is the most commonly used and is considered the most ubiquitous. Unfortunately, the major

* 정 회 원 : 군산대학교 전자정보공학부 박사과정
frank@kunsan.ac.kr
** 종신회원 : 군산대학교 정보통신공학과 교수
jwlee@kunsan.ac.kr (교신저자)
[2012/05/21 투고 - 2012/05/22 심사 - 2012/08/17 심사완료]
☆ This work (Grants No. 00047659) was supported by Business for Cooperative R & D between Industry, Academy, and Research Institute funded Korea Small and Medium Business Administration in 2011.

problem for the satellite-based method used by GPS is that the performance considerably degrades when the satellite signals are severely blocked. Another drawback of GPS is that it is a high-energy-consumption device that can drain the battery of the phone in just a few hours [2].

With smart phones becoming ubiquitous computing devices, smart phone localization has become an interesting research problem. Recent advancements in smart phone hardware introduced internal sensors which include the accelerometer and compass. The presence of these sensors has inspired the development of many applications. Commonly, these are games and augmented reality applications which exploit these sensors to adapt the visualizations relative to the orientation of the device or how it is moved [3]. With these sensors it is possible to detect movements, angles, and rotations of the phone making it a potential navigation device. With these opportunities in mind, we propose to fuse and utilize the phone's accelerometer and compass for infrastructure-independent navigation.

Our approach uses accelerometer to detect steps and obtain the distance traveled along the path and at the same time maintain the correct orientation towards the destination by matching the direction obtained from the compass. To further aid the user in navigating the vicinity, a self-contained map of the area being traversed is also provided. One important feature of a navigation system is to prevent the user from wandering off the specified route. To achieve this, we provide a mechanism for detecting disorientation which in return prevents accumulative errors in distance displacement. However, it is known that readings from built-in compasses and accelerometers are highly noisy. We compensate for this by careful calibration of sensor data which enabled us to detect whether a user is actually walking or not.

Given this relatively simple setup, we are able to contribute a self-contained scheme that provides infrastructure-independent navigation. As a result, additional infrastructures like GPS, Pseudolites, WiFi access points, and RFID are discarded. Additionally, the accuracy and energy consumption trade-off from the use of GPS and WiFi are also eliminated [4], [5].

2. Related Work

2.1. War Driving and War Walking

War driving is a method of collecting Wi-Fi signal data by driving around has been adopted by Skyhook Wireless [6] and Intel Place Lab [7]. On the other hand, war walking is a similar approach except that data are collected by walkers moving around the area rather than drivers moving on roads. The basic idea is to drive around the area and create a map of existing Wi-Fi access points. The map produced by the process is then made accessible to mobile devices. Therefore, every time a mobile device enters a mapped area, it computes its location by detecting Wi-Fi access points, and locating them in its stored radio map. It was claimed in [8] that war walking is more effective in collecting signals compared with war driving for the reason that sidewalks are closer to buildings, in which Wi-Fi signals originate, than roads. Also, walking is obviously slower than driving, thus collecting more Wi-Fi signal samples from each mapping point. A comparison between war walking and war driving reveals a trade-off between map calibration hours and positional accuracy [9].

Though it might look efficient, a large portion of space remains uncovered. These areas include buildings and paths of university campuses, malls, apartment complexes, parking lots and a lot more. War-driving on these paths is impractical while war walking would be overly time consuming. Generally, war driving is costly and time consuming. When infrastructures change, such as the removal, transfer, or new installation of equipment, new measurements are required to update the mapping database.

2.2. Dead Reckoning

In navigation, dead reckoning is the process of calculating one's current position by using a previously determined position, or fix, and advancing that position based upon known or estimated speeds over elapsed time, and course. Dead reckoning is today implemented in some high-end automotive navigation systems in order to overcome the limitations of GPS/GNSS technology alone. [10].

Dead reckoning also takes advantage of the human

motion. The method involves the process of estimating position by projecting traveled distance and orientation from a known starting point. The common approach is to detect and count steps by means of the acceleration, estimate each step's length and propagate a new position using the measured absolute heading of the compass sensor. Studies conducted in [11] [12] utilized sensor readings which are logged continuously in order to determine the user's current position. This is calculated from the previous position and the sensor data using the dead reckoning algorithm. A study in [13] assumes that sensor readings are somewhat consistent even though they may not be accurate, such as when a user conducts two similar movements the sensor readings should also be similar. However, the drawback of dead reckoning is the accumulation of error which is caused by the noisy sensor readings with the increase of traveled distance. For this reason, as much noise are partially extracted by using a Kalman filter [14] which estimates true values of the observed sensor measurements based on the validated settings.

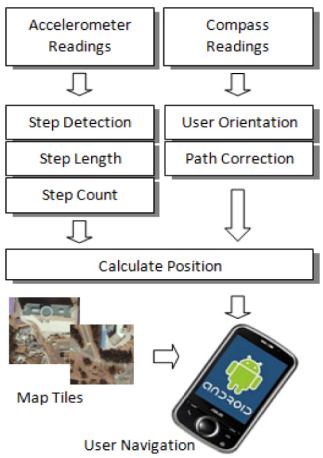
2.3. Triangulation

The triangulation location sensing technique uses the geometric properties of triangles to compute object locations. Triangulation is divisible into the subcategories of lateration, using distance measurements; and angulation, using primarily angle or bearing measurements [15]. Lateration and angulation depend on the distance or angle of a user to a set of beacons to determine the relative position.

To implement a precise indoor localization using lateration, we need an infrastructure of beacons. In [16], they proposed a pedestrian localization system that tracks the accurate position of a pedestrian with a small number of position-unknown beacons. A mobile location estimation and tracking technique for wireless communication systems was proposed in [17]. The location estimation is based on the differences of downlink signal attenuations, which are used to determine circles composed by possible mobile locations. The actual location is then given by the intersection of the circles. In [18], they were able to utilize a combination of the network-based and satellite-based approach in developing a location and tracking system for mobile devices by combining the outcomes from both techniques. Cell tower based localization produces poor localization accuracy while GPS has serious issues regarding signal obstacles and energy depletion. In response to that, we present a scalable approach that is self-contained, thus eliminating the need for additional infrastructure.

3. Self-contained Navigation System

In this section, we describe the approach for the design of our proposed navigation system. We start with capturing the data produced by the sensors and store it for later analysis. Second, we created the map layout of the routes to be used by the user. Third, we present our approach for detecting steps and its relationship to the actual foot trails. Finally, we discuss our approach for trail matching and orientation to assure accurate path traversal. The system overview is shown in Figure 1.



(Figure 1) The system architecture.

3.1. Acquiring Sensor Data

Our first step is to capture the readings of the sensors that we use to track the movements of a person. The movements that we intend to record are the steps of the users as they walk. To do this, the readings of the accelerometer and compass were logged. Once the readings are stored, we can

further analyze the data and lead us to a conclusion about the behavior of the sensors with regards to their accuracy and sensitivity to movements.

3.2. Map Creation and Path Establishment

The map used in our prototype was derived from OpenStreetMap [19], a collaborative mapping site. The maps are created using data from portable GPS devices, aerial photography, other free sources or simply from local knowledge. Both rendered images and the vector dataset are available for download under a Creative Commons Attribution-ShareAlike 2.0 license.

While editing the map directly from the OSM site, the aerial imagery of a geographic location is used as backdrop. Afterwards, we downloaded the map and had it edited offline in order to add more details and enhancements. Using JOSM [20], map data from OpenStreetMap can be accessed in XML format consisting of nodes, ways, and areas with their corresponding coordinates and annotations. More specifically, the nodes in the map are important in establishing the paths for each user-specified point of origin and destination. The path that will guide the user to his point of destination is formed by the interconnection of those nodes.

3.3. Step Detection and Path Displacement

The movement and the direction of the user are tracked by using the actual sensor readings. The natural computation method is to double-integrate acceleration, where the first integration computes speed, and the second computes displacement. Unfortunately, several factors cause fluctuations in acceleration, resulting in erroneous displacements [21].

Our approach is to look at the sensor readings as indicators of the observed motion. To realize this, we took advantage of the sensors available in today's smart phones. In our case, a smart phone equipped with an accelerometer and a compass was used. By analyzing the readings from both sensors, we were able to detect the steps and orientation of the user. The proper tracking and calibration of these readings has a significant impact on the accuracy and reliability of our system.

Upon analysis of the sensor readings from section 3.1, we observed data patterns that can be attributed to certain movements of a user while holding the phone such as walking and turning around. While walking, the phone experiences slight shaking movements which is in turn being detected by the 3 axes of the accelerometer. We infer that a step is detected each time the z axis reading reaches a 0.5 m/s^2difference. To eliminate false step detection and increase the reliability of the procedure, the reading is only processed if the current data pattern is attributed to walking.

Another important routine is the measurement of the entire path that the user will have to walk from its point of origin to the destination. From the known coordinates of the waypoints along the path, the distance between each point can be derived and therefore tells us the total path length. We were able to calculate the distance between two points using the formula [22]:

$$a = \sin^2(\Delta lat/2) + \cos(lat_1).\cos(lat_2).\sin^2(\Delta long/2)$$
$$c=2.atan2(\sqrt{a},\sqrt{(1-a)})$$
$$d=R.c \quad (1)$$
(where R is earth's radius, R = 6,371km)

It is also necessary to determine the bearing or angle of a path in order to direct the user towards the correct heading. Again, using the path formed by two points we derive its bearing by using the formula:

$$\Theta = atan2(\sin(\Delta long).\cos(lat2).$$
$$\cos(lat1).\sin(lat2) - \sin(lat1).\cos(lat2).\cos(\Delta long)) \quad (2)$$

With the path heading and distance at hand, it is now possible to measure the path displacement. As the user moves along, the distance travelled is updated each time a step is detected.

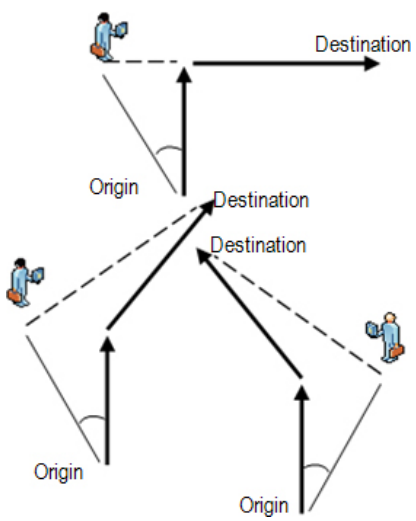
3.4. Trail Matching and Heading Orientation

Together with the phone's accelerometer readings, the compass orientation readings are also recorded. Combining these, we can create a directional trail of the user. The distance traveled as well as the total path length is expressed in meters, while orientation is in degrees with respect to

magnetic north.

Using the method in section 3.3, we are now aware of the heading of each path. This information is used to match the orientation of detected steps against the expected heading of the route. During the acquisition of sensor data, it is observed that even if walking on a straight path the compass reading would always deviate from the specified heading on certain degrees. From that observation, we used a value of 45º to serve as allowable window for the heading of each detected step. Whenever the difference between the step heading and actual path heading exceeds the said value, the user is informed of its disorientation. By keeping track of the heading of each step, the user is prevented from unintentionally wandering off the specified route.

Each time a user commits a disoriented step, it is understood that this could result to accumulative errors. If not handled, this would result to inaccuracy with regards to the user's position and path displacement. To address this, we use a simple approach which allows us to compute the 'penalized' distance. Using the user's current position, the current path, and the next path, we can derive the corrected length of the next path. The process is illustrated in Figure 2.

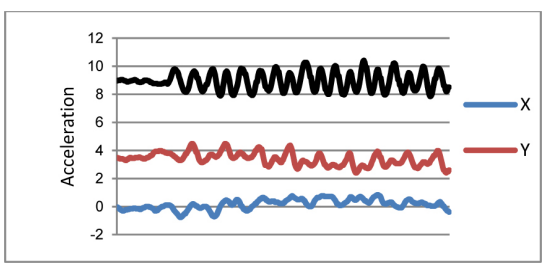


(Figure 2) Three scenarios of path correction process. The arrows represent the real paths, solid lines represent the user's disoriented path, and the dashed lines represent the corrected path.

4. Implementation and Evaluation

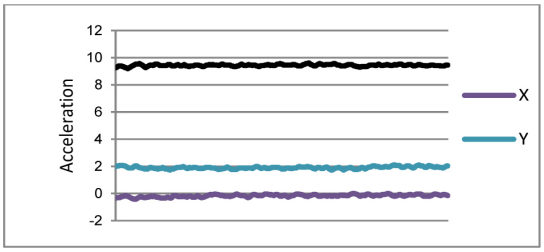
4.1. The Prototype

The device used for the implementation of our prototype is an Android-enabled embedded system module called HBE-SM5-S4210. Aside from the built-in sensors, it is also equipped with full functionalities of an Android phone. Before we started with the actual system development, it is important that we first analyze the behavior of the device sensors. For that purpose, we recorded the sensor readings of the accelerometer and compass for activities like walking as well as standing still. In order to calibrate our measurements we tested the device by walking 10 steps or equivalent to 20 strides. By average each set of steps took 11.4 seconds, thus each stride is approximately 570 milliseconds. We observed that on each step, there is a peak value among the readings which would indicate a variation in the acceleration. To confirm our observation, we plotted our sensor readings and were able to see the actual reading patterns that correspond to the steps indicated by the peak values on the graph shown in Figure 3a and Figure 3b.



(Figure 3a). The readings for the X, Y, and Z axes of the accelerometer have been logged.

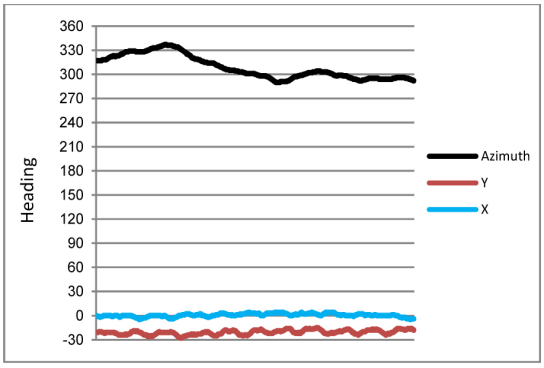
As shown in Figure3a, the less disturbed lines indicate the period where the person has just started walking. As the phone senses a walking movement, the accelerometer readings noticeably show a pattern which indicates variations in the acceleration caused by the shaking motion from a step. We focused on the z axis readings which represents the vertical acceleration of the device with respect to the earth's gravitational pull.



(Figure 3b) Accelerometer readings while the user is standing still.

On the other hand, Figure3b shows the readings of the device while held in place. For the same length of time as Figure3a, we logged the readings for the X, Y, and Z axes of the accelerometer while the person is standing still. As indicated by the lines, there is not much variation in the acceleration values.

Similarly, we also observed some patterns with regards to the actual compass readings. We tested the compass by walking along a straight path and at the same time logging the sensor readings as shown in Figure 4. While walking, the direction of our path with respect to the magnetic north is known. Expectedly, there were variations with the compass readings in each step although the walking motion was kept in the right direction as close as possible. As long as the difference between the step and the path heading does not exceed the threshold value, the step is considered in the right direction.

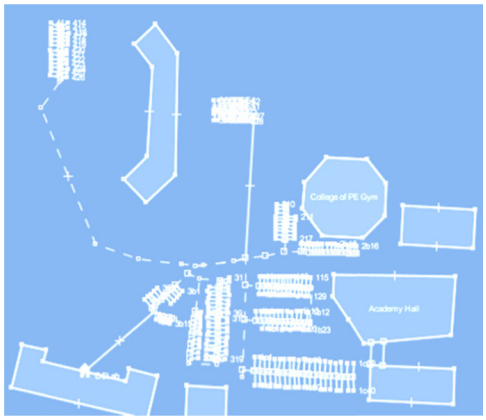


(Figure 4). The azimuth readings represent the angle of the phone with respect to the magnetic north. Although the walking direction is kept as straight as possible, the sensor readings would always deviate from a perceived value.

We then created the map to be used for our prototype. For our test, we intend to use it as a car locator for a wide parking area. We mapped a certain area in our university in which parking areas are located far from each other. From the OpenStreetMap website, we used the aerial imagery of the area as backdrop then we traced it to initially create a draft of the map. We placed roads, paths, buildings and some other markers and labels that will be used as reference points for the navigation. Afterwards, we downloaded the draft and was finalized by adding more details to the map such as parking slots with their corresponding coordinates. Figures 5a-5c show the procedure.

(Figure 5a) The aerial imagery of the area to be mapped as shown from the OpenStreetMap website.

(Figure 5b). The map initially created using the OpenStreetMap online editor.



(Figure 5c) The finalized map of the area created in JOSM, indicating the parking slots.

4.2. Performance Evaluation

The software for our system was implemented using the Android SDK integrated with Eclipse Java IDE. We installed the prototype on an Android-enabled device and tested it with 20 persons as a car locator for a wide parking area. The prototype was used by holding the device face up while the user is walking at a normal pace. From a known point of origin, the users were asked to select a destination point which corresponds to a parking slot. The location of the parking slot chosen by each user is not revealed until he is able to completely cover the path as indicated by the application. For the purpose of testing the accuracy of our approach, the background map tiles were intentionally removed during the testing phase. As a result, the users were only provided with the trail marker and compass pointer to show the direction of the path they are traversing.

In order to ensure that the distance travelled by the user is closely measured, it is important that every step made by the user should be detected. To do this, we did a little experiment in which the users walked while holding the phone. We counted the steps manually while at the same time also being logged by the system. After the walk, we retrieved the log file and the number of steps detected by the application closely matched the actual number of steps we have counted with a minimal difference of 2% by average. This confirms our observation that the spikes in the readings of the accelerometer can be attributed to the walking movements of a person. An excerpt of the result of this procedure is shown in Figure 6.

Test No.	Steps Made	Steps Detected
1	30	31
2	58	61
3	80	82
4	100	102
5	150	155
6	76	75
7	90	93
8	112	113
9	125	124
10	130	129

(Figure 6) The number of steps by the users compared with the number of steps detected by the system.

Aside from detecting user movements, another important capability of a navigation system is to accurately reflect the actual displacement by the user. To evaluate the accuracy of our proposed system in terms of measuring the distance traveled by the user, we took note of the actual distance from the point of origin to the destination as well as the distance travelled by the user to fully cover the given route. As discussed in section 3.3, the step length of the users were derived and is used as the basis for determining the number of steps it would take to reach the destination from a known starting point. For example, if the step length of a person is about 71.5cm, it would take about 140 steps to reach a destination 100 meters away. Thus, we were able to compare the actual distance of the path against the distance travelled by the user as measured by the system. We also compared the calculated number of steps to fully cover the distance against the number of steps detected from the users. We did this test on 20 users, each with a different path to be taken. The figure below shows the result:

As shown in Figure 7a, the distance and number of steps for each path has been pre-determined. By keeping track of the user's movements, the distance travelled as well as the number of steps made by the user were derived. The last two columns show the difference between the actual and measured distance as well as the difference between the calculated and detected number of steps. The highlighted

	Actual Distance (m)	Calcula-ted No. of Steps	Mea-sured Distance (m)	Detec-ted No. of Steps	Distance Diff. (m)	Steps Diff.
User 1	145	203	146.33	205	1.33	1.86
User 2	122.31	171	124.20	174	1.89	2.65
User 3	73.13	102	80.66	113	7.53	10.55
User 4	142.92	200	145.62	204	2.70	3.78
User 5	173.43	243	181.31	254	7.88	11.03
User 6	167.82	235	168.46	236	0.64	0.89
User 7	199.41	279	198.44	278	0.97	1.36
User 8	201.51	282	199.86	280	1.65	2.31
User 9	173.85	244	171.31	240	2.54	3.56
User 10	197.82	277	195.58	274	2.24	3.14
User 11	177.1	248	173.45	243	3.65	5.11
User 12	167.4	235	165.60	232	1.80	2.52
User 13	188.96	265	184.87	259	4.09	5.72
User 14	115.36	162	124.92	175	9.56	13.39
User 15	246.39	345	249.12	349	2.73	3.82
User 16	211.52	296	219.14	307	7.62	10.67
User 17	134.48	188	131.34	184	3.14	4.40
User 18	254.85	357	257.68	361	2.83	3.97
User 19	164.5	230	163.46	229	1.04	1.46
User 20	161	226	160.61	225	0.40	0.55

(Figure 7a) The actual path distance, calculated number of steps, and measured distance compared to the system's readings.

rows indicate a path wherein disorientation occurred, which resulted to an excess distance and a few number of steps. Despite some user disorientation during the test, the system had an average error of only 2% mainly due to its correction mechanism. Since we are able to measure the distance walked by the user, therefore we can also determine his distance from the destination as shown in Figure 7b.

The noisiness of the readings by the built-in sensors in a smart phone has always been a challenge for researchers and developers alike. Same with the accelerometer, the built-in compass we used for keeping track of the user's orientation is also prone to noise. Given a known orientation, walking along and keeping the path as straight as possible will not assure a stable compass reading. The compass reading would always deviate to certain degrees away from the perceived orientation. We solved this by providing a threshold value for the compass reading of each detected step. Through this, the user can walk naturally while keeping him on the right direction. We keep track of the heading of each detected step and compare it against the actual orientation of the path being traversed. The sample readings of two paths are shown in Figure 8a while Figure 8b shows an instance where the user is informed of his disorientation.

(Figure 7b) Based on the calculated step length of the user, the distance to be travelled towards the destination can be determined.

Path 1			Path 2		
Detected Heading	Expected Heading	Heading Diff.	Detected Heading	Expected Heading	Heading Diff.
196.00	185.21	10.79	18.00	286.72	**91.28**
212.00	185.21	26.79	21.00	286.72	**94.28**
215.00	185.21	29.79	296.00	286.72	9.28
217.00	185.21	31.79	296.00	286.72	9.28
216.00	185.21	30.79	298.00	286.72	11.28
216.00	185.21	30.79	297.00	286.72	10.28
216.00	185.21	30.79	299.00	286.72	12.28
218.00	185.21	32.79	298.00	286.72	11.28
218.00	92.51	**125.49**	327.00	2.12	**35.12**
333.00	92.51	**240.49**	356.00	2.12	6.12
35.00	92.51	**57.51**	353.00	2.12	9.12
120.00	92.51	27.49	354.00	2.12	8.12
126.00	92.51	33.49	354.00	2.12	8.12
132.00	92.51	**39.49**	353.00	2.12	9.12
134.00	92.51	**41.49**	353.00	2.12	9.12
139.00	92.51	**46.49**	352.00	2.12	10.12
86.00	92.51	6.51	353.00	2.12	9.12
93.00	92.51	0.49	352.00	2.12	10.12
95.00	92.51	2.49	352.00	2.12	10.12
96.00	92.51	3.49	353.00	2.12	9.12

(Figure 8a) The heading of each step is logged and compared to the expected path orientation. The highlighted rows indicate disoriented steps that have been detected.

(Figure 8b) Whenever a disoriented step is detected, the user is informed and corrected.

Combining the approaches we have utilized, we came up with the most important capability of our proposed system which is to guide the user on its navigation and assuring that the desired destination point is reached. Initially, when the map was created the actual coordinates of waypoints were determined which include the parking slots marked as destination points. In our tests, users were able to reach the specified destination points by following the path directed by the system. To further confirm the accuracy of our proposed system, we also determined the coordinates of the point that the application detected as the reached destination. We determined the distance between the actual coordinates of the destination point and that of the reached location. As shown in Figure 9, there is very little difference between the coordinates of the two locations which makes the extra distance tolerable.

	Actual Coordinates		Endpoint Coordinates		Difference (m)
	Latitude	Longitude	Latitude	Longitude	
Path 1	35.94246	126.68216	35.94246	126.68216	0.09422
Path 2	35.94215	126.68113	35.94215	126.68113	0.2514
Path 3	35.94248	126.68217	35.94248	126.68216	0.1571
Path 4	35.94215	126.68113	35.94215	126.68114	0.6932
Path 5	35.94263	126.68165	35.94261	126.68167	2.752
Path 6	35.94270	126.68239	35.94270	126.68239	0.07836
Path 7	35.94258	126.68274	35.94259	126.68274	0.2542
Path 8	35.94269	126.68276	35.94269	126.68276	0.2198
Path 9	35.94292	126.68251	35.94292	126.68251	0.3784
Path 10	35.94289	126.68293	35.94289	126.68293	0.07339

(Figure 9) The actual coordinates of the marked locations in the map were known. Upon reaching the destination point, the user's location was also logged and was compared against its pre-determined coordinates.

Finally, we compared the user's path on a certain route with the one generated by our system as well as with the path generated by GPS. As shown in Figure 10, the path of the user as guided by the system is very much similar to the actual route than that of the GPS.

(Figure 10) The coordinates of the three paths were plotted to a graph in order to visualize the data. As shown in the figure, the path walked by the user is very much similar to the actual route therefore outperforming GPS.

5. Conclusion and Future Works

In this paper, we presented a self-contained navigation system that is intended to be used on smart phones. With a very minimal setup, we were able to contribute an infrastructure-independent navigation system that does not require additional infrastructures like GPS, Pseudolites, WiFi access points, and RFID. The system was installed on an Android-enabled device and the application was tested by using it as a car locator for a large parking area.

Using the available built-in sensors, namely the accelerometer and the compass, we were able to associate the movements of a user with the reading patterns generated by the sensors. With proper calibration and by taking note of the peak values during movements, steps were detected as well as determined its orientation relative to a given heading which prevents the user from wandering off the path. With a self-contained map and a set of pre-determined waypoints, a path guides the user from a specified point of origin up to the point of destination. To verify the accuracy of the proposed system, we compared the actual coordinates of the specified destination points with the coordinates of the location in which the user was led by the system at the end of the path. As shown by the results, there is a very minimal difference between the two points in terms of their coordinates and is therefore tolerable. Moreover, our work outperformed GPS by providing users with a path which closely resembles the actual route.

Despite the favorable performance demonstrated by our work, there is still room for improvement that will be addressed in the future. As presented, the navigation system was only designed to detect the movements attributed to walking. This means that further improvements can be done by enabling the system to automatically adapt to different movements such as jogging and running, which will add to its robustness.

References

[1] “Global Positioning System”, http://en.wikipedia.org/wiki/Global_Positioning_System

[2] S. Gaonkar, J. Li, R. R. Choudhury, L. Cox, and A. Schmidt, “Micro-Blog: sharing and querying content through mobile phones and social participation,” in MobiSys, 2008, pp. 174 – 186.

[3] Tokusho, Y., and Feiner, S. “Prototyping an Outdoor Mobile Augmented Reality Street View Application.” In International Symposium on Mixed and Augmented Reality (ISMAR) (2009).

[4] I. Constandache et al., “EnLoc: Energy-efficient localization for mobile phones,” in IEEE INFOCOM Mini Conference, 2009.

[5] Youssef, M.; Yosef, M.A.; El-Derini, M.; , ″GAC: Energy-Efficient Hybrid GPS-Accelerometer-Compass GSM Localization,″ GLOBECOM 2010, 2010 IEEE Global Telecommunications Conference , vol., no., pp.1-5, 6-10 Dec. 2010

[6] “Skyhook Wireless”, http://www.skyhookwireless.com

[7] S Y.C. Cheng, Y. Chawathe, A. LaMarca, and J. Krumm, “Accuracy Characterization for Metropolitan-Scale Wi-Fi Localization,” Proc. Third Int’l Conf. Mobile Systems, Applications and Services (MobiSys), June 2005.

[8] M. Kim, J.J. Feilding, and D. Kotz, “Risks of Using

AP Locations Discovered through War Driving," Proc. Fourth Int'l Conf. Pervasive Computing, May 2006.

[9] Tsui, A.W.T.; Wei-Cheng Lin; Wei-Ju Chen; Polly Huang; Hao-Hua Chu; , "Accuracy Performance Analysis between War Driving and War Walking in Metropolitan Wi-Fi Localization," Mobile Computing, IEEE Transactions on , vol.9, no.11, pp.1551-1562, Nov. 2010

[10] "Dead Reckoning", http://en.wikipedia.org/wiki/Dead_reckoning

[11] Z. Sun, X. Mao, W. Tian, X. Zhang, "Activity classification and Dead Reckoning for Pedestrian Navigation with Wearable Sensors," in Measurement Science and Technology, 20(1), 2009.

[12] S. Beauregard and H. Haas, "Pedestrian dead reckoning: A basis for personal positioning," in Proceedings of the 3rd Workshop on Positioning, Navigation and Communication (WPNC06), 2006, pp. 27 – 35.

[13] Nguyen, T.-L.; Zhang, Y.; Griss, M.; , "ProbIN: Probabilistic inertial navigation," Mobile Adhoc and Sensor Systems (MASS), 2010 IEEE 7th International Conference on, vol., no., pp.650-657, 8-12 Nov. 2010

[14] G. Welch and G. Bishop, "An introduction to the Kalman filter," University of North Carolina at Chapel Hill, Chapel, pp. 1 – 16, 1995.

[15] J. Hightower, G. Borriello, "Location sensing techniques," technical report UW-CSE-01-07-01, Computer Science and Engineering, University of Washington, August 8, 2001.

[16] Seungwoo Lee; Byounggeun Kim; Hoon Kim; Rhan Ha; Hojung Cha; , "Inertial Sensor-Based Indoor Pedestrian Localization with Minimum 802.15.4a Configuration," Industrial Informatics, IEEE Transactions on , vol.7, no.3, pp.455-466, Aug. 2011

[17] Ding-Bing Lin; Rong-Terng Juang; Hsin-Piao Lin; , "Mobile location estimation and tracking for GSM systems," Personal, Indoor and Mobile Radio Communications, 2004. PIMRC 2004. 15th IEEE International Symposium on , vol.4, no., pp. 2835-2839

[18] Chao-Lin Chen; Kai-Ten Feng; , "Hybrid location estimation and tracking system for mobile devices," Vehicular Technology Conference, 2005. VTC 2005-Spring. 2005 IEEE 61st, vol.4, no., pp.2648-2652 Vol. 4, 2005

[19] http://www.openstreetmap.org

[20] http://en.wikipedia.org/wiki/OpenStreetMap

[21] Constandache, I.; Choudhury, R.R.; Rhee, I.; , "Towards Mobile Phone Localization without War-Driving," INFOCOM, 2010 Proceedings IEEE , vol., no., pp.1-9, 14-19 March 2010

[22] http://www.movable-type.co.uk/scripts/latlong.html

◐ 저 자 소 개 ◑

프랭크 엘리호데 (Frank I. Elijorde)

2003년 Western Visayas College of Science and Technology, Philippines, BS in Information Technology
2007년 Western Visayas College of Science and Technology, Philippines, MS in Computer Science
2011년~현재 Kunsan National University, South Korea, Graduate Student in Ph. D. Course
관심분야 : Distributed systems, data mining, social networks, ubiquitous sensor networks, RFID.
E-mail : frank@kunsan.ac.kr

이 재 완 (Jaewan Lee)

1984년 중앙대학교 이학사-전자계산학
1987년 중앙대학교 이학석사-전자계산학
1992년 중앙대학교 공학박사-전자계산학
1996년 3월~1998년 1월 한국학술진흥재단 전문위원
1992년~현재 군산대학교 교수
관심분야 : 분산 시스템, 운영체제, 실시간 시스템, 컴퓨터 네트워크, 클라우드 컴퓨팅 등
E-mail: jwlee@kunsan.ac.kr