Journal of Universal Computer Science, vol. 14, no. 14 (2008), 2381-2399
submitted: 30/9/07, accepted: 30/4/08, appeared: 28/7/08 © J.UCS

Implementation of a Prototype Positioning System for LBS on U-campus

Jaegeol Yim
(Dongguk University, Seoul, Republic of Korea
yim@dongguk.ac.kr)

Ilseok Ko
(Dongguk University, Seoul, Republic of Korea
isko@dongguk.ac.kr)

Jaesu Do
(Dongguk University, Seoul, Republic of Korea
dojesu@dongguk.ac.kr)

Jaehun Joo
(Dongguk University, Seoul, Republic of Korea
givej@dongguk.ac.kr)

Seunghwan Jeong
(Dongguk University, Seoul, Republic of Korea
honourj@dongguk.ac.kr)

Abstract: Location-based service is one of the most popular buzzwords in the field of U-cities. Positioning a user is an essential ingredient of a location-based system in a U-city. For outdoor positioning, GPS based practical solutions have been introduced. However, the measurement error of GPS is too big for it to be used for U-campus services, because the size of a campus is smaller than that of a city. We propose the Relative-Interpolation Method to improve the accuracy of outdoor positioning. However, indoor positioning is also necessary for a U-campus because the GPS signal is not available inside buildings. For indoor positioning, various systems including Cricket, Active Badge, and so on have been introduced. These methods require special equipment dedicated to positioning. Our method does not require such equipment because it determines the user's position based on the received signal strength indicators (RSSIs) from access points (AP) which are already installed for WLAN. The algorithm we use for indoor positioning is a kind of fingerprinting method. However, our algorithm builds a *decision tree* instead of a look-up table in the *off-line* phase. Therefore, the proposed method is faster than the existing indoor positioning methods in the *real-time* phase. We integrated our indoor and outdoor positioning methods and implemented a prototype indoor-outdoor positioning system on a laptop. The experimental results are discussed in this paper. In implementing the prototype, we also implemented a C# library function which can be used to read the RSSIs from the APs.

Keywords: ubiquitous-campus, LBS, positioning, fingerprinting method, decision tree
Categories: H.1.2, H.4.2, I.4.3, I.4.4, H.3.3

1 Introduction

A location based service(LBS) provides the users with useful information based on the geographic location they designate or at which they are currently located. Examples of LBSs include directory services, gateway services, location utility services, presentation services, route services, and so on [Marwa, 03] [Krishnamruthy, 02] [Virrantaus, 01] [Koo, 03] [Bravo, 04] [Jose, 01] [Almad, 05] [Wu, 05] [Abdullaev, 07].

Positioning users is an essential technique for developing location-based services. Positioning techniques can be classified into outdoor and indoor ones. Thanks to the Global Positioning System (GPS), outdoor positioning has reached the level of practical use. The disadvantages of GPS include the significant inaccuracy of its positional measurement [Wilson, 96] and the limitation of its availability [Zheng, 05]. In order to overcome these disadvantages, outdoor positioning methods using cellular communication infrastructures [Ygnace, 01] and wireless communication infrastructures [Koo, 03] have been introduced.

Many studies have been performed regarding indoor positioning. Among them, Cricket [Priyanthat, 00] and Active Badge [Want, 92] are pioneers in this field. They are highly accurate, however, they require special equipment dedicated to positioning. RADAR [Bahl, 00] is a radio-frequency based system for indoor positioning which requires a set of base stations.

In this paper, we introduce a positioning system for U-campuses (Ubiquitous campuses) that makes location based services (LBSs) on a campus possible. It is common for a user of an LBS on a campus to drive into the campus, walk into a building and visit an office or a classroom. Therefore, our positioning system should be capable of operating both indoors and outdoors.

Satirapod et al theoretically and experimentally proved that the performance of GPS data processing is largely dependent on the magnitude of the systematic error and the satellite geometry[Satirapod, 03]. It is known that the typical accuracy of commercially available GPS receivers is about 15 meters, because of the systematic errors including ionospheric effects, ephemeris errors, satellite clock errors, and so on. In order to improve the accuracy of GPS, differential GPS is widely used and [Adrados, 02] showed that the mean location error of GPS can be decreased to 5.2 meters by using the differential GPS technique which requires a network of fixed ground based reference stations. Considering the common size of campuses, it may be concluded that the average error of commercial GPS may be too big. However, installing new reference stations is not viable option. Therefore, we propose the Relative-Interpolation Method. It requires two reference points as other interpolation methods do. However, the position of our reference point is the x-y coordinates of the window in which the campus map is being rendered instead of the absolute latitude and longitude.

[Lin, 05] performed a comparison of the accuracy, precision, complexity, robustness, and scalability of the Bayesian method, K-NN, and neural networks. They concluded that K-NN provides the best overall performance for the purpose of indoor positioning. However, the process times of the real-time phases of these methods increase in proportion to the number of candidate points. Since the typical distance between two neighboring candidate points is 1 meter, the number of candidate points of indoor positioning for U-campus is huge. Therefore, we use the decision tree method in order to speed up the real-time phase of indoor positioning. The process time of the real-time phase is important, because the user is continually moving.

We implemented a prototype of our indoor and outdoor positioning system on a laptop computer and performed performance test experiments. The average errors of the prototype system for outdoor positioning and indoor positioning were 4.875 and 2.6 meters, respectively.

2 Related Works

In this paper, we propose both an outdoor and indoor positioning method. GPS is successfully used for outdoor positioning in practice. However, one of the disadvantages of GPS is its inaccuracy. Many attempts have been made to improve the accuracy of GPS. Differential GPS (DGPS) is one of these efforts [Wilson, 96]. Another attempt was made by Feng and Law who used a mobile phone to assist GPS [Feng, 02]. Many researchers have made improvements to GPS positioning by applying Kalman filters [Mao, 04] [Chen, 01].

The idea of DGPS is to make use of reference stations whose exact geographical locations (latitude and longitude) are known. A reference station receives the GPS data, calculates the error associated with it, and broadcasts the error to the mobile terminals. By making use of this error information, a mobile terminal can adjust the location information obtained from its GPS receiver.

The goal of this study is to implement a positioning system for U-campuses. Therefore, the positioning system should be able to find the user's location no matter where he or she is, inside or outside of a building. The outdoor positioning module was able to be implemented easily using a GPS receiver. However, its accuracy needed to be improved, because the size of a campus is much smaller than that of a city. Therefore, we implemented the Relative-Interpolation Method which is inspired by the concept of DGPS.

Cricket and Active Badge are the most well-known indoor positioning methods for LBS. There are many other indoor positioning methods which have recently been described in various journals. The Friis-based positioning method characterized by making use of the relation between the distances and received signal strengths was introduced in [Lassabe, 05].

Our work was directly inspired by RADAR [Bahl, 00]. The conceptors of this system placed three base stations (desktop PCs), BS_1, BS_2, and BS_3 on the floor. Their mobile host was a laptop computer. If the mobile host broadcasts a UDP packet then the base stations can receive the packet and read its signal strength (RSS: received signal strength). The positioning system uses the RSS to construct a *look-up* table during the off-line analysis phase, as well as to infer the location of the user in *real time*. Let X be the vector of the RSS obtained during the real-time phase. RADAR examines the look-up table and finds the K closest entries to X. Then, it returns the average of the K candidate points associated with the K closest entries as the user's current location. Therefore, the strategy adopted by RADAR is called the K nearest neighbors or K-NN.

Many indoor positioning systems which do not require special equipment have also been developed. Most of them are WLAN (wireless LAN) based. A WLAN based positioning system determines the user's position by referring to the RSS of the signals from various access points (APs). The most popular method used to determine the user's position is the fingerprinting method [Kaemarungsi, 05] [Lin, 05] [Madigan, 05] [Youssef, 03]. In implementing the fingerprinting method, any classification technique can be applied such as K-NN [Bahl, 00], Bayesian [Madigan, 05], neural network [Battiti, 02] [Lin, 05], decision trees, and so on [Han, 01].

Let $X = (x_1, x_2, ..., x_n)$ be the vector of collected signal strengths. The Bayesian method predicts that the user's position is CP_i (the i-th candidate point) if $P(CP_i \mid X) > P(CP_j \mid X)\ \mathit{for}\ 1 \le j \le m,\ j \ne i$, where m is the number of candidate points. According to Bayes' theorem, $P(CP_i \mid X) = \frac{P(X \mid CP_i)P(CP_i)}{P(X)}$. As P(X) is constant for all classes, only $P(X \mid CP_i)P(CP_i)$ need be maximized. A positioning system using the Bayesian classification method finds the CP_i that maximizes $P(X \mid CP_i)P(CP_i)$ and returns it as the user's position [Madigan, 05].

The domain of our positioning system is a campus. Consequently, the number of candidate points is huge and we had to speed up the real-time phase of the indoor positioning system. Therefore, we decided to use the decision tree method and our indoor positioning algorithm is similar to the ID3 algorithm [Han, 01].

The whole structure of our indoor positioning module is similar to that of the RADAR indoor positioning system. The major difference is that we use the RSSIs from APs instead of base stations. Since the APs are already installed for WLAN, no extra equipment dedicated to the purpose of positioning is required. Another difference is that we build a *decision tree* instead of a *look-up table*. Constructing a decision tree is more time consuming. However, making a decision based on a decision tree is much faster. Therefore, our method is faster in the *real-time* phase.

3 Design of an Integrated Positioning System

This paper proposes an integrated positioning system which can be used outdoors as well as indoors.

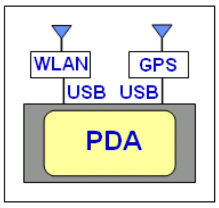


Figure 1: The hardware structure of our indoor-outdoor positioning system

3.1 System Structure

The hardware structure of our system is shown in [Fig. 1]. It is basically a PDA equipped with a WLAN card and a GPS receiver

The proposed indoor-outdoor positioning system runs on a PDA. The system reads the GPS signal when turned on and determines if the current position is outside of a building by referring to the validity field of the GPS data. If the field indicates that the data is valid, then the system decides to use outdoor positioning, otherwise it decides to use indoor positioning. This implies that our strategy sometimes erroneously uses indoor positioning when the user is actually outside of a building. Even when the user is actually outdoors, the GPS data indicates that it is invalid if the line of sight to the satellites suffers from interference. In this case, we actually apply the indoor positioning method outside. We can actually obtain the user's location with the indoor positioning system, because our wireless service covers virtually the entire campus. Our positioning strategy is summarized in [Fig. 2].

```
Indoor-Outdoor Positioning Algorithm{
    1)  Read GPS ;
    2)  If GPS is valid {
            A.  OutdoorPositioning ;
            }
    3)  else {
            A.  Apply indoor positioning ;
            }
}
```

Figure 2: The Indoor-Outdoor Positioning Algorithm

3.2 Outdoor Positioning

We use a GPS receiver for outdoor positioning. At each of the reference points, A and B, we measure its GPS coordinates, longitude and latitude as well as its X and Y coordinates on the window. Let x_{lon} and x_{lat} be the longitude and latitude of the reference point x (A or B), and x_x and x_y be its x and y coordinates, respectively. Let C be the unknown user's current position and C_{lon} and C_{lat} be the GPS data measured at the user's current position. Then, we estimate the user's x and y coordinates on the map, C_x and C_y, respectively, with the following expressions.

$$C_x = \left(\frac{C_{lon} - A_{lon}}{B_{lon} - A_{lon}} \right) (B_x - A_x) + A_x \qquad \textbf{(1)}$$

$$C_y = \left(\frac{C_{lat} - A_{lat}}{B_{lat} - A_{lat}} \right) (B_y - A_y) + A_y \qquad (2)$$

We call our method Relative-Interpolation, because the coordinates of the reference point are not the absolute longitude and latitude, but the x-y coordinates relative to the (0, 0) point of the window in which the map is rendered. We note that the north to south line of the map must coincide with the y axis of the window.

3.3 Indoor Positioning

The proposed indoor positioning method is basically a fingerprinting method similar to RADAR. However, the fingerprint consists of the RSSIs from the APs instead of the RF strengths from the base stations. A more significant difference is that the proposed method builds a decision tree instead of a look-up table during the off-line phase. Therefore, the proposed method takes more time than the existing fingerprinting methods during the off-line phase. However, it is much faster than the other methods in the *real-time* phase. In practice, the *real-time* phase is time critical, whereas the *off-line* phase is not. Therefore, a positioning service must be fast in the *real-time* phase.

In the off-line phase of the proposed method, a decision tree is built with training data. An entry T of the training data set is a 6-tuple, *T=(cp, I1, I2, I3, I4, I5),* where cp is an integer identifying a candidate point and Ii, for 1≤ i≤ 5, is the discretized value of the RSSI of the *i-th* AP. An example of the discretizing policy is $I1 = \{x \mid x > -30\}$, $I2 = \{x \mid -40 < x \le -30\}$, and so on. A training data set consists of a large number of tuples.

Given a set of training data, a decision tree is built with the algorithm Construct_DT shown in [Fig. 3]. Step (4) of the algorithm computes *I* which is the expected information needed to classify a given sample and is given by

$$I(s_1, s_2, ..., s_m) = -\sum_{i=1}^{m} p_i \log_2(p_i) \tag{3}$$

where m is the number of candidate points, *S* is the number of tuples in the training data set (rows of Table in the algorithm), and s_i is the number of rows of *S* in class CP_i, $p_i = {}^{s_i}\!/_{s}$.

Step (5) of the algorithm computes the entropy or expected information based on the partitioning into subsets by CP_i. Let CP_i have v distinct values, $\{a_1, a_2, ..., a_v\}$. CP_i can be used to partition S into v subsets, $\{S_1, S_2, ..., S_v\}$, where S_j contains those samples in S that have value a_j of CP_i. Let S_{ij} be the number of samples of class CP_i in a subset S_j. The entropy E(CP_i) is given by

$$E(CP_i) = \sum_{j=1}^{v} \frac{s_{ij} + ... + s_{mj}}{s} I(s_{ij}, ..., s_{mj}). \tag{4}$$

At step (6), the algorithm computes the information gain G(CP_i) by the following expression, $Gain(CP_i) = I(s_1, s_2, ..., s_m) - E(CP_i)$.

```
Algorithm Construct_DT(int[][] Table, ListType MacList, int Index)
    (1)  If (Number of rows in Table != 1)
         A. If all the CPs of Table are the same, CPi, then Tree[Index] = CPi and return
         B. else if (Number of rows in Table == 0) then Tree[Index] = Empty and return
         C. else if (Number of columns in Table == 1) then Tree[Index] = for each CPi in Table, probability of CPi and return
    (2)  else Tree[Index] = CP and return
    (3)  end if
  // Number of rows != 1 and !A and !B and !C
    (4)  Compute I
    (5)  Compute Entropies for each AP
    (6)  Tree[Index] = MacAddress of the AP with maximum Gain and array P. P[i] is the probability of CPi in Table
    (7)  Construct subMacList
    (8)  loop(i=1; i<=number of Intervals; i++)
         A. generate subTable
         B. subIndex = Index*number of Interval + i
         C. Construct_DT (subTable, subMacList, subIndex)
    (9)  end loop
  end Construct_DT
```

Figure 3: The algorithm used for constructing the decision tree

4 Implementation of RSSI Reader, a library function

WLAN based indoor positioning is more economical than other methods and many studies on this subject have been performed recently. WLAN based indoor positioning cannot be realized without reading the RSSIs from the APs. Therefore, we developed a C# dynamic link library which returns the RSSIs.

4.1 The Process of Communication with 802.11 NIC

The 802.11 NIC (Network Interface Card) is a wireless LAN card which is an OSI layer 1 (physical layer) and 2 (data link layer) device, as it provides physical access to a networking medium and a low-level addressing system through the use of a MAC address. A computer catches the information (SSID, BSSID, RSSI, Network type and so on) broadcasted by the APs through an 802.11 NIC, selects the AP with the strongest RSSI, and communicates with it.

An application program running on the computer communicates with its NIC through the Network Driver Interface Specification (NDIS). The relationship between the NDIS and the other parts of the OSI 7 layers is depicted in [Fig. 4]. The NDIS was jointly developed by Microsoft and 3Com Corporation and is an application programming interface (API) for NICs. In other words, an application program can read the information from a 802.11 NIC by implementing the NDIS IOCTL Interface which can access the NDIS driver. The NDISUIO driver in [Fig. 4] is a public code which comes as a part of Windows-XP and provides an application programming interface for the NDIS driver.

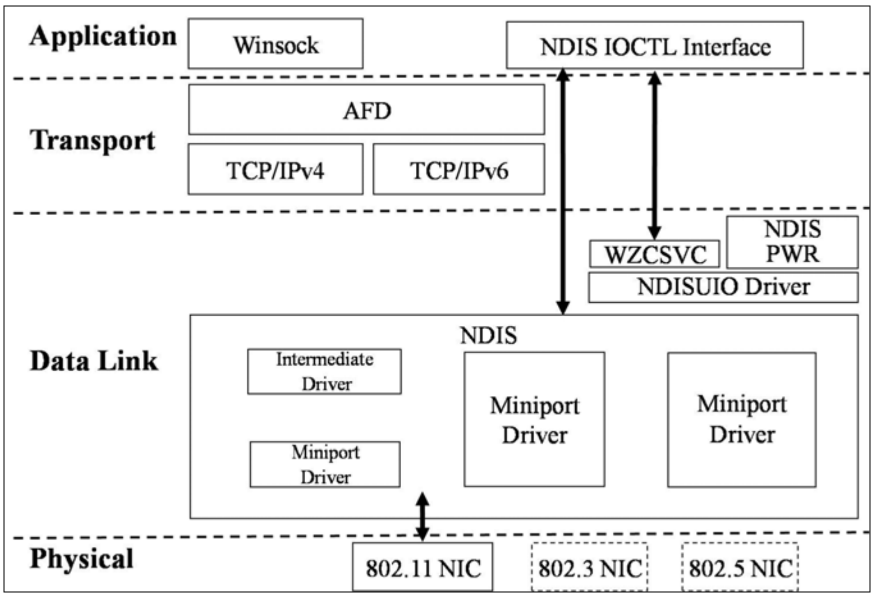


Figure 4: Windows Networking Architecture

Windows XP selects a wireless network as the base wireless utilities embedded in Windows XP using Wireless Zero Configuration. Wireless Zero Config Service (WZCSVC) in [Fig. 4] provides services for initializing the connection service, monitoring the device status, and driver communication with an OID specifying 820.11. Since Windows XP automatically uses WZCSVC, we have to disable Windows XP's Wireless LAN configuration in order for us to use WZCSVC.

An application program can communicate with the 820.11 NIC through the NDISUIO driver using the WZCSVC service, as shown in [Fig. 4]. The process of communication between an application program and wireless LAN card is summarized in [Tab. 1].

4.2 Implementation of an RSSI Reader

The implementation of an RSSI reader in C# with Microsoft Visual .NET 2005 on a Samsung laptop SENS M40 equipped with an Intel(R) PRO/Wireless 2200BG Network Connection will be discussed. We implemented the RSSI reader in WirelessManager Project and named it WirelessManager.dll.

① Disable Windows XP's WZCSVC
② Access NDISUIO driver using CreateFile() function in kernel32.dll. CreateFile() returns the handle.
③ Communicate with the device via DeviceIoControl() function in kernel32.dll
④ Release the device using CloseHandle() when the communication is over and enable WZCSVC.

Table 1: A summary of the process of communication between an application program and wireless LAN card

4.2.1 Preparation for communication with 802.11 NIC in C#

In order for a DLL export function such as kernel32.dll's CreateFile(), DeviceIoControl(), CloseHandle() and FormatMessage() to be invoked in C#, its PInvoke (Platform Invoke) should be declared. [Tab. 2] is an example of PInvoke for DeviceIoControl(). A PInvoke() is a static extern method with the DllImport attribute designated.

```
[DllImport("kernel32", SetLastError = true)]
private static extern bool DeviceIoControl(
void* hDevice, IO_CTL_CODE dwIoCtlCode,
void* lpInBuf, int nInBufSize,
void* lpOutBuf, int nOutBufSize,
out int lpBytesRet, void* lpOverlapped
);
```

Table 2: PInvoke() declaration for DeviceIoControl()

CreateFile() tries to connect to the NDISUIO driver using NDIS_FILE_NAME(@"\\.\\Ndisuio"), and returns Handle if it succeeds. The first parameter of DeviceIoControl() is the Handle returned by CreateFile() and the second parameter, dwIoCtlCode, is the IO Control Code designating the IO operation to be performed. The generation rule for IO Control Code is defined in DDK (Winioctl.h, nuiouser.h) and the code for communication with a network device consists of DeviceType(0x12), Access(0x1|0x2), Function, and Method(0x0) arranged in the following manner:
((DeviceType)<<16) | ((Access)<<14)| ((Function)<<2) | (Method).
[Fig. 5] shows the layout of the IO Control Code.

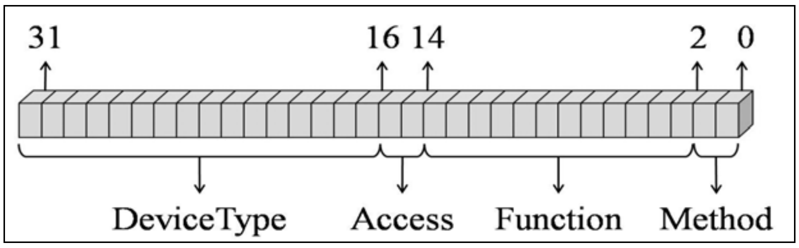


Figure 5: The format of an IO Control Code

The role to be performed by the IO Control Code is determined by the value of the Function field. The values of Function used in the proposed system are listed in [Tab. 3].

IO Control Code	**InBuffer**	**OutBuffer**
Bind Wait (0x204)	not use	not use
Query Binding (0x203)	Use	use
Open Device (0x200)	Use	not use
Set Object ID (0x205)	Use	not use
Query Object ID (0x201)	Use	use

Table 3: IO Control Code and usage of buffer

CloseHandle() should be invoked to release the resources at the end of the communication process. In the case of error, FormatMessage() should be used in order to inform the user of its cause.

4.2.2 DeviceIoControl()

The function to be performed by DeviceIoControl() is determined by the second parameter of the IO Control Code. The usages of InBuffer and OutBuffer are also dependent on the second parameter, as shown in [Tab. 3]. When the value of the second parameter is Bind Wait, DeviceIoControl tries to access the Network Device with the handle generated by CreateFile(). Neither InBuffer nor OutBuffer is used.

If Bind Wait succeeds, Query Binding records BindingIndex, DeviceNameOffset, DeviceNameLength, DeviceDescrOffset, DeviceDescrLength, DeviceName, and DeviceDescription of the wired or wireless LAN card in OutBuffer, as shown in [Fig. 6]. BindingIndex is a sequential number designating the order of recording, DeviceNameOffset is the offset of the DeviceName field, DeviceNameLength is the length of DeviceName, DeviceDescrOffset is the offset of the DeviceDescription field, and DeviceDescrLength is the length of DeviceDescription field.

Therefore, our RSSI reader keeps invoking Query Binding DeviceIoControl() until the DeviceDescription in the buffer reveals that the information recorded in the buffer is about the wireless LAN card. When it is, we save the DeviceName in WLANCardName so that we can use it for opening the device.

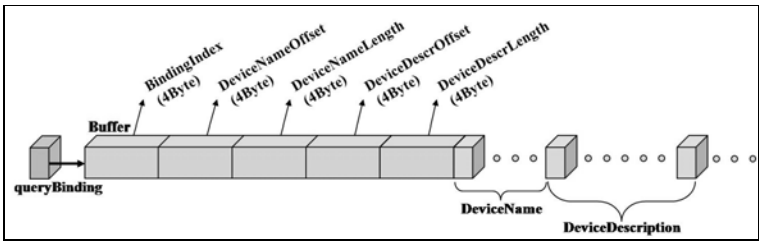


Figure 6: Query Binding's usage of Buffer

Open Device is used for opening the device specified in the InBuffer. Therefore, our RSSI reader stores the WLANCardName in the InBuffer before invoking Open Device DeviceIoControl. Open Device DeviceIoControl does not use OutBuffer.

The purpose of Set Object ID is to deliver a command to the wireless LAN card. If we put Bssid_List_Scan(0x0D01011A), one of the OIDs defined in ntddndis.h of DDK, in the InBuffer and invoke DeviceIoControl then the wireless LAN card collects all the Bssids of the APs around the card. A Bssid is broadcasted by an AP and consists of a Mac Address, SSID, RSSI, Infrastructure mode, and so on. It takes about 6 seconds for the wireless LAN card to collect all of the Bssids.

The purpose of Query Object ID is to take data from the wireless LAN card. If we put Bssid_List(0x0D010217), one of the OIDs defined in ntddndis.h of DDK, in the InBuffer and invoke DeviceIoControl then the Bssid List built by Set Object ID is copied into the OutBuffer, as shown in [Fig. 6]. If we set Oid in [Fig. 7] to Bssid_List(0x0D010217) before the invocation of DeviceIoControl, then Data is set to the number of Bssids recorded in the OutBuffer as a result of DeviceIoControl.

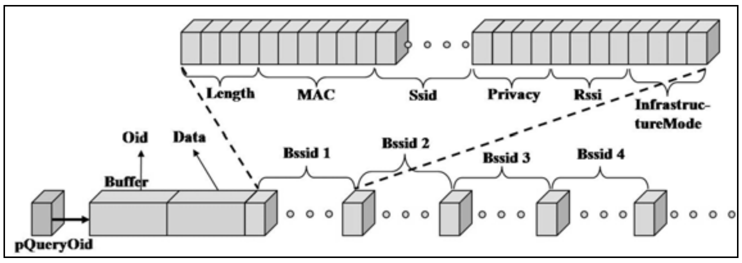
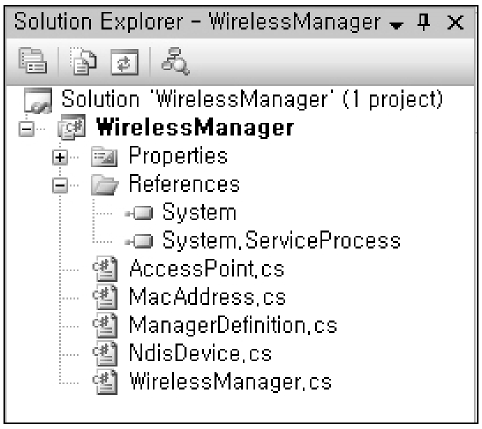


Figure 7: Query Object ID's Usage of the buffer

4.2.3 DeviceIoControl()

WirelessManager in [Fig. 8] is the project generating WirelessManager.dll which is an NDIS IOCTL Interface in [Fig. 4]. The WirelessManager project is written in C# and makes use of the pointer operator (*), the address operator (&), and the member inverse reference operator (->). Therefore, unsafe code should be activated in the project attribute.



```
Solution Explorer - WirelessManager
Solution 'WirelessManager' (1 project)
WirelessManager
    Properties
    References
        System
        System.ServiceProcess
    AccessPoint.cs
    MacAddress.cs
    ManagerDefinition.cs
    NdisDevice.cs
    WirelessManager.cs
```

Figure 8: WirelessManager Project

The roles of the files shown in [Fig. 8] are summarized in [Tab. 4]. Some of them are used for defining data types, while others are used for communicating with or controlling the wireless LAN card. The name space of the project is defined as "Dongguk3CSLab.Wireless".

File Name	Contents
ManagerDefinition.cs	◎ Definitions of IO Control types ◎ Definitions of 802.11 types
AccessPoint.cs	◎ Definitions of AccessPoint types
MacAddress.cs	◎ Definitions of MacAddress structure for AccessPoint
NdisDevice.cs	◎ Definitions of Device types
WirelessManager.cs	◎ Communication with or control of wireless LAN

Table 4: Files of WirelessManager Project

4.3 The result of the implementation

Using the WirelessManager.dll generated by the WirelessManager project, we can read the RSSIs.

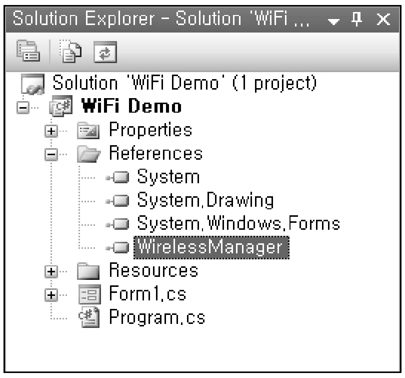


Figure 9: Add WirelessManager into Reference

```
NdisDevice dev;

// If WirelessManager is being used, then// terminate it and reinitialize
if(WirelessManager.CurrentDevice != null)
    WirelessManager.Finish();
WirelessManager.Start();

// Assign the devicename and devicedescription of the WLAN card
into dev
WirelessManager.OpenWirelessDevice();
dev = WirelessManager.CurrentDevice;
```

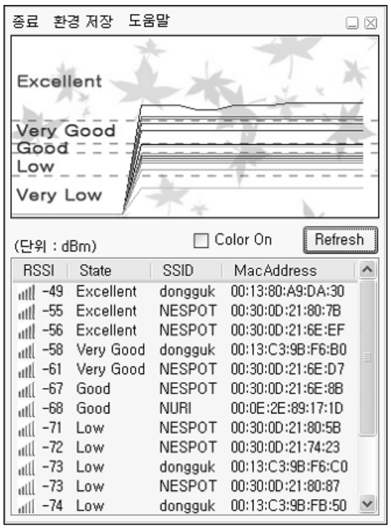
Table 5: Initialization step for WirelessManager.dll

4.3.1 Reading RSSI using WirelessManager.dll

Using WirelessManager.dll, we can implement a C# project which makes use of the RSSIs. In order to do so, we have to add WirelessManager.dll to the References of the project [see Fig. 9] and insert the line, "using Dongguk3CSLab.Wireless", and the initialization step shown in [Tab. 5] into the program. Then, we can collect all of the information broadcasted by the APs near to the computer using ScanAPList() as follows:

```
AccessPoint[] aplist;
aplist = WirelessManager.ScamAPList(dev);
```

We implemented a program which reads the RSSIs and an example GUI of this program is shown in [Fig. 10].



RSSI	State	SSID	MacAddress
-49	Excellent	dongguk	00:13:80:A9:DA:30
-55	Excellent	NESPOT	00:30:0D:21:80:7B
-56	Excellent	NESPOT	00:30:0D:21:6E:EF
-58	Very Good	dongguk	00:13:C3:9B:F6:B0
-61	Very Good	NESPOT	00:30:0D:21:6E:D7
-67	Good	NESPOT	00:30:0D:21:6E:8B
-68	Good	NURI	00:0E:2E:89:17:1D
-71	Low	NESPOT	00:30:0D:21:80:5B
-72	Low	NESPOT	00:30:0D:21:74:23
-73	Low	dongguk	00:13:C3:9B:F6:C0
-73	Low	NESPOT	00:30:0D:21:80:87
-74	Low	dongguk	00:13:C3:9B:FB:50

Figure 10: An example GUI of our program for reading the RSSIs

5 Experiments

We implemented a laptop based prototype of our indoor-outdoor positioning system written in C# on our campus. The operating system of the computer was Microsoft Windows XP. The laptop was equipped with an Intel(R) PRO/Wireless 2200BG Network Connection and a Model X-150 GPS receiver from Jacom Inc. We discuss the experimental results for this positioning system in this section.

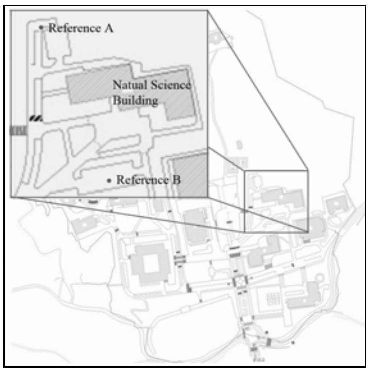


Figure 11 : The test bed used for the outdoor positioning experiments

5.1 Experiments of outdoor positioning

The outdoor positioning experiments were performed outside of the Natural Science Building shown in [Fig. 11]. The measured x-y coordinates and the latitude and longitude of the reference points are shown in [Tab. 6]. We performed experiments in which the x-y coordinates of the current position obtained by clicking the mouse on the window were compared with those obtained from the outdoor positioning program 200 times and the results are summarized in [Tab. 7]. The results indicate that among the 200 experiments, on 11 occasions the error was less than 1 m, on 17 occasions the error was between 1 and 2 m, and so on. The average error was 4.875m.

	Coordinates		**GPS data**	
	X	Y	Latitude	Longitude
A	1842	1140	N 35°51′ 48.7″	E 29°11′ 44.72″
B	2112	1566	N 5°51′ 45.01″	E 129°11′ 47.5″

Table 6: x,y coordinates, latitude and longitude of reference points

error (m)	0 ~ 1	1 ~ 2	2 ~ 4	4 ~ 6	6 ~ 8	8~
Occurrence	11	17	61	51	33	27
Probability	5.5%	8.5%	30.5%	25.5%	16.5%	13.5%
Average Error = 4.875 m						

Table 7: Summary of the results of the outdoor positioning experiments

5.2 Experiments of indoor positioning

We implemented the K-NN, Bayesian and proposed decision tree methods in order to compare their performances. The test bed used for the indoor positioning experiments is the Micro LAB on the 4-th floor of the Natural Science Building on our campus. We performed experiments in which the 1-NN, Bayesian and decision tree methods were applied to training data with N=5, I=6, and M=96, where N is the number of APs, I the number of intervals, and M the number of candidate points, in order to compare their accuracies. The test results are shown in [Fig. 12]. In this figure, the "number of samples" is the same as the number of measurements performed at a candidate point to obtain the training data. An entry of the lookup table of 1-NN is the average of the measurements. When the number of samples is 10, the 1-NN method is much more accurate than the others. However, the difference decreases as the number of samples increases and when the number of samples is 50, the accuracies of the three methods are almost the same.

According to [Lin, 05], changing the number of nearest neighbors, K, from 1 to 2 and 4 has no obvious effect on the accuracy and neural network shows the worst accuracy among the three methods, namely, 1-NN, Bayesian, and neural network methods. Therefore, we compared the decision tree method with the 1-NN and Bayesian methods in this experiment.

The advantage of the decision tree method is the rapidity of the real-time phase process. We measured the real-time phase execution times of the 1-NN, Bayesian, and decision tree methods. The result is shown in [Fig. 13]. The time complexity of the real time phase of K-NN is O(N*M), because for each row of the lookup table K-NN calculates the Euclidean distance. The time complexity of the real time phase of the Bayesian method is O(I*N*M), because it counts the number of samples belonging to the same interval for each AP. In the case of the decision tree, the execution time of the real time phase is not affected by the number of candidate points and it is O(I*N).

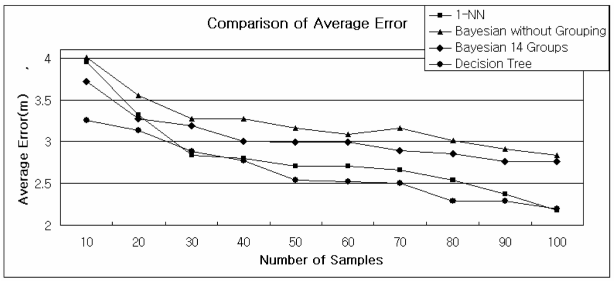


Figure 12: A comparison of the accuracy of the various methods

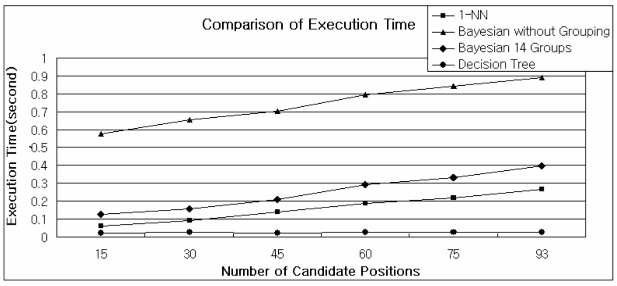


Figure 13: A comparison of the execution times of the various methods

6 Conclusions

Positioning is an essential technique for Location Based Services (LBSs). We proposed an indoor-outdoor positioning system for U-campuses. For outdoor positioning, we proposed the Relative-Interpolation Method. The experiments showed that the average error of the proposed method was 4.875m.

For indoor positioning, we proposed a decision tree method. The algorithm builds a *decision tree* instead of a look-up table in the *off-line* phase. Therefore, the proposed method is faster than the existing indoor positioning methods in the *real time* phase. The experimental results showed that the proposed method is much faster than the existing methods with equivalent accuracy.

Finally, this paper provides a C# dynamic-link library with which the RSSIs can be read from the APs. This library was used to implement the proposed indoor positioning module.

7 Future Work

In the near future, we are planning to develop an LBS system for our U-campus. As the first step of the development, a map viewer will be developed in C#. This map viewer reads a map file created by AutoCAD and draws the map on the graphical user interface of the system. We are planning to enhance the map viewer with various functions such as zoom-in, zoom-out, and so on. This map viewer will be integrated with the positioning module introduced in this paper. Then, we are planning to decorate the system with simple LBSs.

References

[Abdullaev, 07] Abdullaev S., Ko, I.S.: A Study on Successful Business Intelligence Systems in Practice, JCIT:Journal of Convergence Information Technology, Vol.2 No.2, 2007, pp.94-103.

[Adrados, 02] Adrados, C., Girard, I., Gendner, J., Janeau, G.: Global Positioning System (GPS) location accuracy improvement due to Selective Availability removal, Comptes Rendus Biologies, February 2002, vol. 325, Issue 2, pp. 165-170.

[Ahmad, 05] Ahmad, U., Nasir, U., Iqbal, M., Lee, Y., Lee, S., Hwang, I.: On building a reflective middleware service for location-awareness, Proceedings of the 11th IEEE International Conference on Embedded and Real-Time Computing Systems and Applications 2005, 17-19 August 2005, pp. 439 – 442.

[Bahl, 00] Bahl, P., Padmanabhan, V.: RADAR: An in-building RF-based user location and tracking system, INFOCOM 2000, March 2000, pp. 775-784.

[Battiti, 02] Battiti, R., Nhat, L., Villani, A.: Location-aware computing: a neural network model for determining location in wireless LANs, Technical Report DIT-02-00083, Department of Information and Communication Technology, University of Trento, February 2002.

[Bravo, 04] Bravo, A.M., Moreno, J.I., Soto, I.: Advanced positioning and location based services in 4G mobile-IP radio access networks, 15th IEEE International Symposium on Personal, Indoor and Mobile Radio Communications, 2004. PIMRC 2004, 5-8 September 2004, vol. 2, pp. 1085 – 1089.

[Chen, 01] Chen, G., Harigae, M.: Using IMM adaptive estimator in GPS positioning, Proceedings of the 40th SICE Annual Conference, 25-27 July 2001, pp. 78 – 83.

[Lee, 04] Lee, C.H., Chang, Y.S., Park, G.H., Ryu, J.H., Jeong, S.G., Park, S.H., Park, J.W., Lee, H.C., Hong, K.S., Lee, M.H.: Indoor positioning system based on incident angles of infrared emitters, 30th Annual Conference of IEEE Industrial Electronics Society 2004. IECON 2004. Vol. 3, 2-6 November 2004, pp. 2218 – 2222.

[Feng, 02] Feng, S., Law, C.: Assisted GPS and its impact on navigation in intelligent transportation systems, Proceedings of the IEEE 5th International Conference on Intelligent Transportation Systems, 2002, pp. 926 – 931.

[Han, 01] Han, J., Kamber, M.: Data Mining Concepts and Techniques, Morgan Kaufmann Publishers, 2001.

[Jose, 01] Jose, R., Moreira, A., Meneses, F., Coulson, G.: An open architecture for developing mobile location-based applications over the Internet, Proceedings of the Sixth IEEE Symposium on Computers and Communications, 3-5 July 2001, pp.500 – 505.

[Kaemarungsi, 05] Kaemarungsi, K.: Efficient design of indoor positioning systems based on location fingerprinting, Proceedings of International Conference on Wireless Networks, Communications and Mobile Computing 1, 2005, pp. 181-186.

[Koo, 03] Koo, S., Rosenberg, C.: Location-based E-campus Web Services: From Design to Development, Proceedings of the First IEEE International Conference on Pervasive Computing and Communications, 2003 (PerCom 2003), 23-26 March 2003, pp. 207-215.

[Krishnamurthy, 02] Krishnamurthy, N: Using SMS to Deliver Location-based Services, Proceedings of 2002 IEEE International Conference on Personal Wireless Communications (Proceedings of ICPWC'2002), 15-17 December 2002, pp. 177 – 181.

[Lassabe, 05] Lassabe, F., Canalda, P., Chatonnay, P., Spies, F.: A Friis-based calibrated model for WiFi terminals positioning**,** Sixth IEEE International Symposium on a World of Wireless Mobile and Multimedia Networks, 2005. WoWMoM 2005. 13-16 June 2005, pp. 382 – 387.

[Lin, 05] Lin, T., Lin, P.: Performance Comparison of Indoor Positioning Techniques based on Location Fingerprinting in Wireless Networks, Proceedings of the 2005 International Conference on Wireless Networks, Communications and Mobile Computing, Volume 2, 13-16 June 2005 pp. 1569 – 1574.

[Madigan, 05] Madigan, D., Einahrawy, E., Martin, E., Ju, W., Krishnan, P., Krishnakumar, A.: Bayesian indoor positioning systems, Proceedings of the IEEE 24th Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2005), 2005, pp. 1217-1227.

[Mao, 04] Mao, X, Wada, M., Hashimoto, H.: Nonlinear iterative algorithm for GPS positioning with bias model, Proceedings of the 7th International IEEE Conference on Intelligent Transportation Systems, 2004, 3-6 October 2004, pp. 684 – 689.

[Marwa, 03] Marwa M., OpenGIS Location Services(OpenLS): Core Services OGC 03-006r, http://www.opengis.org/

[Priyanthat, 00] Priyanthat, N., Chakraborty, A., Balakrishnan, H.: The Cricket Location-Support System, Proc. of 6th ACM International Conference on Mobile Computing and Networking, Boston, MA, August 2000.

[Satirapod, 03] Satirapod, C., Wang, J., Rizos, C.: Comparing Different Global Positioning System Data Processing Techniques for Modelling Residual Systematic Errors, Journal of Surveying Engineering, Novemver 2003, vol. 129, Issue 4, pp. 129-135.

[Virrantaus, 01] Virrantaus, K., Veijalainen, J., Markkula, J.: Developing GIS-Supported Location-Based Services, Proceedings of the Second International Conference on Web Information Systems Engineering, 2001, Vol. 2, 3-6 December 2001, pp. 66-75.

[Want, 92] Want, R., Hopper, A., Falco, V., Gibbons, J.: The Active Badge Location System, ACM Transactions on Information Systems 10, January 1992. pp. 91-102.

[Wilson, 96] Wilson, T, Barth, J., Pierce, S., Kosro, P., Waldorf, B.: A Lagrangian drifter with inexpensive wide area differential GPS positioning, Proceedings of the MTS/IEEE. Prospects for the 21st Century, 23-26 September 1996, Vol. 2, pp. 851 – 856.

[Wu, 05] Wu, X., Schulzrinne, H.: Location-based services in Internet telephony, Second IEEE Consumer Communications and Networking Conference, 2005, 3-6 January 2005, pp. 331 – 336.

[Ygnace, 01] Ygnace, J., Drane, C.: Cellular telecommunication and transportation convergence: a case study of a research conducted in California and in France on cellular positioning techniques and transportation issues, Proceedings of the IEEE Intelligent Transportation Systems, 2001, 25-29 August 2001, pp.16 – 22.

[Youssef, 03] Youssef, M., Agrawala, A., Shankar, A.: WLAN location determination via clustering and probability distributions, Proceedings of IEEE International Conference on Pervasive Computing and Communications (PerCom), 2003, pp. 143-150.

[Zheng, 05] Zheng, J., Wang, Y., Nihan, N.: Quantitative evaluation of GPS performance under forest canopies, Proceedings of the IEEE Networking, Sensing and Control 2005, 19-22 March 2005, pp. 777 – 782.