DOLPHIN: A Practical Approach for Implementing a Fully Distributed Indoor Ultrasonic Positioning System

Masateru Minami[1], Yasuhiro Fukuju[2], Kazuki Hirasawa[1], Shigeaki Yokoyama[1], Moriyuki Mizumachi[1], Hiroyuki Morikawa[2], and Tomonori Aoyama[2]

[1] Shibaura Institute of Technology, 3-9-14 Shibaura, Minato-ku, Tokyo, Japan
[2] The University of Tokyo, 7-3-1 Hongo, Bunkyo-ku, Tokyo, Japan

Abstract. Obtaining indoor location information is one of the essential technologies for enriching various ubiquitous computing applications. Although many indoor location systems have been proposed until now, wide-area deployments in everyday environments are still extremely rare. To deploy indoor locating systems beyond laboratory use, we believe that the initial configuration cost of the system should be reduced. This paper describes a fully distributed ultrasonic positioning system which enables us to locate various indoor objects with lower initial configuration cost.

1 Introduction

Utilizing real-world information, including both low-level sensor data and high-level context, is expected to enhance the potential of future ubiquitous computing applications. So far, many researchers in ubiquitous computing have made concerted efforts to obtain such information, and have tried to utilize it for developing innovative and useful applications. Since real-world information usually depends highly on the movement of people, geographic locations of people and objects are highly relevant to location-aware applications.

Usually, we can easily obtain location information in outdoor environments by using the global positioning system (GPS). Therefore, researchers on location systems for ubiquitous computing applications have focused mainly on indoor environments. To obtain indoor location information, various types of systems have been developed based on the following systems [1]: infrared [2], floor sensors [3], and RF (radio frequency)[4][5][6]. In comparison with these systems, ultrasonic positioning systems [7][8][9] provide the most accurate location information.

There have been two important research efforts on the ultrasonic positioning system: the Active Bat system[7] and the Cricket system[8]. The Active Bat System is the pioneer work in the development of the ultrasonic positioning system. The Active Bat system consists of Active Bat tags, which transmit an ultrasonic pulse, and ultrasonic receivers mounted on the ceiling. The Active Bat system measures the distance between a tag and a receiver based on the time-of-flight

N. Davies et al. (Eds.): UbiComp 2004, LNCS 3205, pp. 347–365, 2004.

of the ultrasonic pulse, and computes each tag's position by performing multilateration. The Active Bat system also provides direction information, which is useful for implementing many ubiquitous computing applications. However, the Active Bat system employs centralized system architecture, and requires a large number of precisely positioned ultrasonic receivers. Thus, large deployments of the system tend to be prohibitively expensive.

In contrast to the Active Bat system, the Cricket system employs distributed system architecture to avoid the scalability problem. In the Cricket system, ultrasonic transmitters are mounted on the ceiling and a randomized algorithm allows the transmitters to emit ultrasound signals in a distributed manner. Ultrasonic receivers attached on various objects hear these signals and compute their position as well as their orientation. Since the Cricket system has no centralized element and employs a GPS-like positioning model, its privacy protection is advantage. Although both systems are well designed and have the capability of locating indoor objects with high accuracy, there still remains an important problem.

Inherently, a trilateration-based positioning system such as the Active Bat and Cricket systems requires precisely positioned references. In the case of the ultrasonic positioning system, we have to use a lot of references to locate objects in an actual indoor environment since the ultrasonic signal usually can propagate less than five meters and does not penetrate obstacles. For example, in the Active Bat system, ultrasonic receivers are the references, and they are mounted on the ceiling at intervals of around 1.3-1.5 meters. If the room size is a square with 10m sides, we have to use around 70 references and precisely measure their positions. If we apply the system to a larger environment, such as a big conference room or an office building, measuring the positions of all the references becomes more labour-intensive. To avoid this problem, Ward proposed a method for determining the positions of references by using three transmitters [10], but it is not a substantial solution. We believe that reducing the configuration cost is desirable for deploying an indoor location system beyond laboratory use.

On the other hand, there has been interesting work[11] on node localization in wireless sensor networks. In [11], an iterative multilateration technique was proposed to locate huge amounts of sensor nodes by using a small number of precisely positioned references. The idea of iterative multilateration is that a sensor node which can estimate its position using references, becomes a new reference itself. This algorithm proceeds until all sensor nodes are located. Based on this idea, the authors of [11] proposed collaborative multilateration, which can locate a node even if the node cannot receive signals from a sufficient number of references.

The idea of iterative multilateration is applicable to the configuration cost problem, i.e., it can locate many objects using a small number of preconfigured references. In this paper we design and implement a fully distributed indoor ultrasonic positioning system called DOLPHIN (Distributed Object Localization System for Physical-space Internetworking) that locates various indoor objects based on a distributed positioning algorithm similar to iterative multilateration.

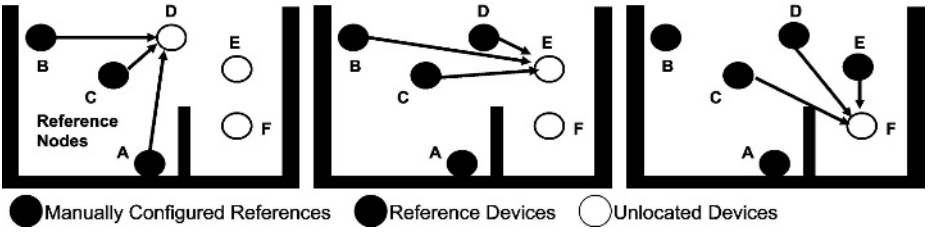


Fig. 1. Iterative Multilateration

The major contribution of [11] is the design of the collaborative multilateration algorithm in a distributed manner and the evaluation of the algorithm via computer simulations; this applies to our interest, which is how to make a workable indoor ultrasonic positioning system which overcomes practical problems, such as the configuration cost, based on the idea of iterative multilateration. Since one important goal of ubiquitous computing is to show the potential of innovative technologies and deploy them beyond laboratory use, we believe that it is important to design and implement a workable system that overcomes the various problems caused in an actual environment. Toward this end, we employed an implementation-based approach to design our indoor ultrasonic positioning system.

This paper is organized as follows. In the next section, we describe our design goals and corresponding approaches in designing the DOLPHIN system. In Section 3, we describe a detailed design of the DOLPHIN system, including the hardware design and positioning algorithm. Then, several techniques for achieving high-accuracy positioning are introduced in Section 4. In Section 5, we evaluate the basic performance of our implemented system in various practical situations. Finally, we summarize our work and discuss our future direction in Section 6.

2 Design Goal

At the start of designing the DOLPHIN system, we listed the following design goals and corresponding solutions:

(1) Easy Configuration
As described in Section 1, one important problem of a conventional triangulation-based indoor positioning system is the configuration cost, because we are forced to measure a huge number of reference positions precisely to achieve high-accuracy positioning. To mitigate this configuration load, we utilize the idea of iterative multilateration.

Figure 1 illustrates the basic idea of the iterative multilateration technique. In the initial state, devices A, B and C have precisely measured positions, and the positions of the other devices are unknown. In the next step, device D, which

can directly receive signals from devices A, B and C, can determine its position (here, we assume that one device can compute its position by receiving three or more signals from the references). However, devices E and F cannot yet receive a sufficient number of signals to determine their position due to such physical obstacles as the wall. Here, if the position of device D is determined and device E can receive a signal from device D, device E can compute its position by using signals from devices B, C and D. If the locations of device D and E are determined, device F can compute its position using devices C, D and E. In this way, all devices can be located.

However, iterative multilateration is just an idea to locate huge numbers of objects by using a small number of references. To develop a practical system, we should consider not only how to apply iterative multilateration to our system, but also how to start the system and recover from failures. We introduce such a positioning algorithm in Section 3.

(2) High-Accuracy Positioning
The second goal is achieving high-accuracy positioning in our system for supporting a wide variety of ubiquitous computing applications. Basically, the trilateration-based ultrasonic positioning system shows good accuracy in an indoor environment. However, since ultrasound is seriously degraded by various obstacles, we must design some error mitigation technique to obtain precise distance measurements. And, moreover, because our system is trying to locate many objects using a small number of references in a distributed manner, structural errors such as the accumulation of positioning errors should also be taken into account. To achieve high-accuracy positioning, in Section 4 we introduce several approaches, including reference selection and no-line-of-sight signal rejection techniques.

Usually, existing indoor ultrasonic positioning systems support not only 3D positioning but also additional functions, such as the capability of detecting the orientation of objects. However, much good work [8][10][12] has already been done for such additional functions. From this point of view, we mainly focus on the above basic goals in this paper.

3 Hardware Design and Positioning Algorithm

3.1 Hardware Design

Figure 2(a) illustrates a block diagram of the DOLPHIN device. To implement iterative multilateration, bi-directional ultrasonic transducers (MA40S, MURATA) are used for both sending and receiving ultrasonic signals. As shown in Figure 2(b), we attached five transducers to the cylindrical module to extend the coverage of the ultrasonic signals in every direction. Each transducer is connected to a signal detector circuit, which separately detects the existence of the ultrasonic pulse and received signal strength level. Figure 2(c) shows the sending and receiving patterns of the transducer module. As shown in this figure, the sending pattern has several null points and the receiving pattern is spherical.

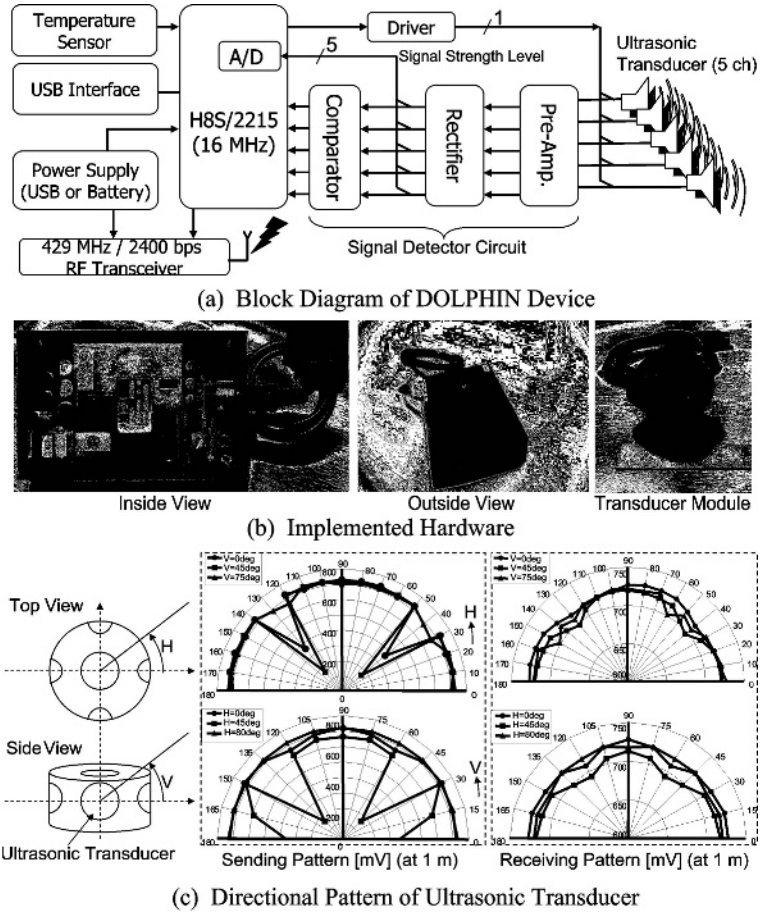


Fig. 2. Hardware Design

This is because each ultrasonic transducer has a spherical directional pattern, and the combination of such patterns yields the null point. Moreover, the placement of the ultrasonic transducers is not determined by considering the signal interference among the outputs of the transducers. As a result, there is a little interference among the transmitted ultrasonic signals. Although it is possible to improve the sending pattern by carefully placing the transducers and attaching horns to the cylindrical module, we assume in this paper that the module has an ideal sending pattern.

The DOLPHIN device also has a 2400-bps, 10-channel, 429-MHz RF transceiver (NHM-10185, Nagano JRC) for time synchronization and the exchange of control messages. Because this RF transceiver includes a powerful error collection function, we selected it for the DOLPHIN system so that we do not need to care about errors in the exchange of control messages. However, the transmission

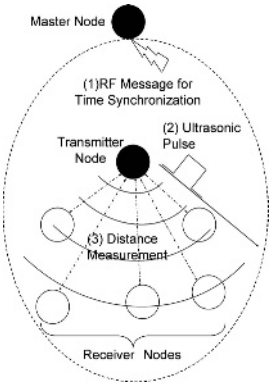
speed of this transceiver is relatively slow compared to recent RF transceivers, and so it limits the positioning speed (1-2 times/sec). This is not a substantial problem in the DOLPHIN system, although we are planning to employ a higher-speed transceiver in the next version.

A one-chip microcontroller (H8S/2215 16 MHz, RENESAS) controls both the ultrasonic transducer and the RF transceiver, and calculates a device's position by performing multilateration. The microcontroller detects the received signal strength information of the ultrasonic signal at each transducer via internal A/D converters, and can obtain the temperature from a one-chip temperature sensor (LM35, National Semiconductor) to estimate the ultrasound propagation speed. This CPU has a USB interface, and we can pull location data out through the USB interface as well as supply power to the device by attaching the device to a USB-enabled device such as a laptop or a PDA.

3.2 Distributed Positioning Algorithm

To organize many DOLPHIN devices as a distributed positioning system, we designed a distributed positioning algorithm that implements the idea of iterative multilateration as well as bootstrapping and failure recovery mechanisms. In the positioning algorithm, the nodes in the system play three different roles in a sequence, as shown in Figure 3: there is one master node, one transmitter node, and the rest are receiver nodes. A node which has determined its position can become a master node or a transmitter node. In every positioning cycle, the master node sends a message via the RF transceiver for time synchronization of all nodes. Here we assume that all nodes in the system can hear the RF signal. When a transmitter node receives the message, it sends an ultrasonic pulse. At the same time, each receiver node starts its internal counter, and stops it as the ultrasonic pulse arrives. The receiver nodes that received the ultrasonic pulse then compute the distance to the transmitter node. If a receiver node measures a sufficient number of distances, it computes its position based on multilateration. By using a member list, three RF messages and two internal timer events, as summarized in Figure 4, the distributed positioning algorithm proceeds as follows.

(1) Bootstrapping Sequence
Figure 5 shows an example bootstrapping sequence of the distributed positioning algorithm. In the initial state of the system, the member list in each node is empty. As described later, the distributed positioning algorithm determines both the master and transmitter node based on the member list. Therefore, each node must collect node IDs in the bootstrapping sequence to start positioning. After turning on a manually positioned reference node, it resets its ADVERTISEMENT_TIMER with an interval of $t_A (t_A = 10 + t_r)$ seconds. Where, t_r is a random initial value in the order of several seconds. In Figure 5, after turning on node A, ADVERTISEMENT_TIMER in the node expires because there is no SYNC_MSG that specifies the node. Then, node A sends ID_NOTIFY_MSG via the RF transceiver. In this example, nothing occurred because there is no node



List	MEMBER_LIST	The MEMBER_LIST contains the following information about neighboring nodes that are capable of becoming reference node: Node ID | Node Priority | Node Position | Distance
RF Messages	ID_NOTIFY_MSG	The node which determined its position transmits this message so that the node can be used as a reference. This message contains the node ID of the node.
	SYNC_MSG	This message is used for time synchronization for distance measurement, and contains the ID of the transmitter node selected by the master node.
	LOC_NOTIFY_MSG	This message is used for advertising the latest location of the transmitter node. This message contains the node ID and 3D position of the transmitter node.
Timers	ADVERTISEMENT_TIMER	This timer is set with t_A (t_A=10sec+random initial value) when the node receives SYNC_MSG for the node. It expires when the node capable of becoming reference node does not receive any SYNC_MSG specifying the node from other nodes within t_A.
	RECOVERY_TIMER	This timer is set with t_R (t_R=15sec+random initial value) when the node holds no-empty node list and receives SYNC MSG. The timer expires if there has been no SYNC_MSG within t_R.

Fig. 3. Positioning Sequence **Fig. 4.** Definition of List, Messages, and Timers

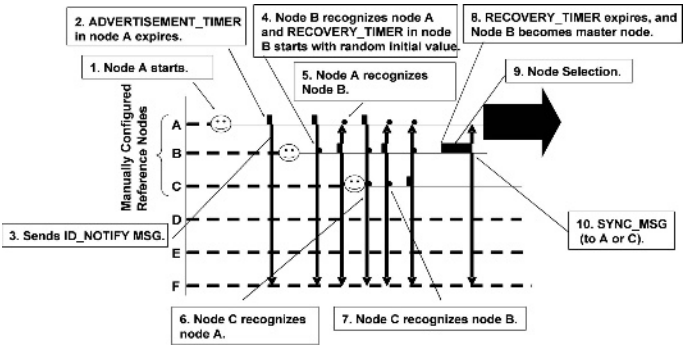


Fig. 5. Example Bootstrapping Sequence

that could hear the message at that time. Therefore, node A simply continues to transmit ID_NOTIFY_MSGs.

After several time periods, node B is turned on next, and starts its ADVERTISEMENT_TIMER. On receiving ID_NOTIFY_MSG from node A, node B recognizes that there is a node which can act as a master node or transmitter node. Then node B stores the node ID in the received message into its member list. At the same time, node B starts its RECOVERY_TIMER with an interval of $t_R(t_R = 15 + t_r)$ seconds. When the ADVERTISEMENT_TIMER in node B expires, node B sends an ID_NOTIFY_MSG, then node A recognizes node B and starts its RECOVERY_TIMER. In the same way, every node including node C recognizes each other. Note that each node sets RECOVERY_TIMER when the node recognizes a neighboring node as summarized in Figure 4. However, there is no master node in the system, and the nodes simply exchange ID_NOTIFY_MSG. Once RECOVERY_TIMER in node B expires, node B becomes a master node. Then node B selects one node as a transmitter node based on the member list, and transmits SYNC_MSG to the selected node. In this way, the bootstrapping

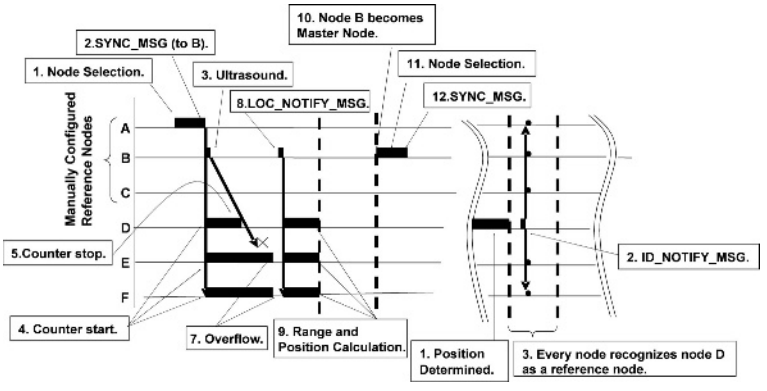


Fig. 6. Example Positioning Sequence

process completes, and the system enters into positioning cycles. Note that our bootstrapping procedure works even if all nodes turned on simultaneously because both ADVERTISEMENT_TIMER and RECOVERY_TIMER are set with random initial values.

(2) Positioning Algorithm

Figure 6 shows an example of the timing sequence in one positioning cycle. In this example, nodes A-C are manually configured references, and nodes D-F are unlocated nodes. Here we assume that nodes A- C have member lists [B, C], [A, C], and [A, B], respectively. Now we consider that node A acts as a master node.

First, node A chooses one node randomly from its node list [B, C]. If node B is chosen, node A transmits SYNC_MSG, including the ID of node B. On receiving the message, node B becomes a transmitter node and generates an ultrasonic pulse. At the same time, nodes D, E, and F become receiver nodes and start their internal counter. If the receiver nodes detect the transmitted ultrasound pulse, they stop their internal counter and calculate the distance from node B.

After several milliseconds (this depends on the time taken by the overflow of the internal counter), node B sends LOC_NOTIFY_MSG to notify the receiver nodes of its position. The receiver nodes that can detect ultrasound store the location of node B and the distance to node B in their member table. After that, all nodes wait and listen for ID_NOTIFY_MSG for a certain period (advertisement period). Any node which can determine the position based on a sufficient number of distances advertises its ID in this phase, and the ID is added to the node list in every node. In the above example, because nodes D, E and F cannot determine their position, no ID_NOTIFY_MSG is sent in this period. The sequence of the above phases is one cycle of the positioning algorithm in the DOLPHIN system. In the next cycle, node B, which acted as a receiver node in the previous cycle, becomes a master node. And, thus, the positioning algorithm proceeds in the same way.

After several cycles of positioning, node D can measure distances from nodes A-C, and position (x_D, y_D, z_D) is obtained by solving the following none-linear simultaneous equations based on a least squares estimation:

$$r_1 = \sqrt{(x_D - x_i)^2 + (y_D - y_i)^2 + (z_D - z_i)^2} \quad (1)$$

where (x_i, y_i, z_i) denotes the position of the i-th reference, and r_i is the measured distance between node D and the reference. At that time, node D sends ID_NOTIFY_MSG in the advertisement period. All of the other nodes that received ID_NOTIFY_MSG from node D add the ID of node D to their member list, and node D is recognized as a candidate for master node or transmitter node.

In our current implementation, a candidate for master or transmitter node sends its average position to achieve high-accuracy positioning. In this way, we can locate all nodes in the DOLPHIN system. Note that we utilize four reference nodes to estimate node position in our current implementation so that we can resolve ambiguity and obtain good estimation, although we considered three reference nodes in this example.

(3) Failure Recovery

In the DOLPHIN system, we have to consider two failures, which are node failure and recognition failure, to continuously execute the abovementioned positioning algorithm. Node failure occurs when a node suddenly stops because of an unpredictable accident, and recognition failure occurs when the ID_NOTIFY_MSG transmitted from a node does not reach the other nodes because of a bad communication channel or message collision.

RECOVERY_TIMER is utilized for recovering from node failure. This timer is set with a random initial value when the nodes receive SYNC_MSG. If any SYNC_MSG is not transmitted for a certain period in the system , the system can be concluded that the positioning algorithm halted because of some node failure. In this case RECOVERY_TIMER in a node expires, and the node becomes a master node. In this way, the positioning algorithm resumes from node failure.

If a node capable of being a master node does not receive SYNC_MSG which specifies the node from other nodes within a certain period, the node conclude that the node is not recognized as a node capable of being a master or transmitter node by neighboring nodes (i.e., recognition failure). In this case, the advertisement timer in the node expires, and the node retransmits ID_NOTIFY_MSG again in the next positioning cycle. Note that, to avoid an ID_NOTIFY_MSG collision, the node sends ID_NOTIFY_MSG at a certain probability similar to the backoff algorithm utilized in 802.11. We have run the above algorithm via 24 DOLPHIN devices for 24 hours, and the algorithm has continued normally.

4 Improving Positioning Accuracy and Scalability

When we utilized the DOLPHIN system in an actual environment, the system suffered from various factors leading to errors. According to our experimentation,

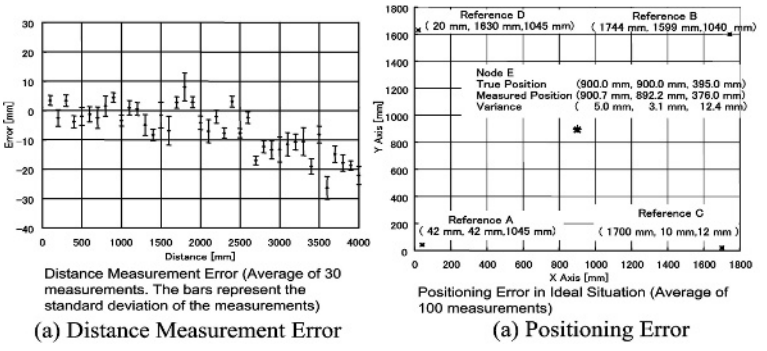


Fig. 7. Positioning Performance in Ideal Situation

there are two major factors leading to errors in the DOLPHIN system: error accumulation and a no-line-of-sight signal. The following sections describe how we have tackled these problems.

4.1 Error Accumulation Problem

Basically, our hardware implementation can measure the distance between two nodes with an accuracy within 1-2 cm, if the distance is shorter than 3 m, as shown in Figure 7(a) (based on this measurement, our system determines that a measured distance shorter than 3 m is valid). This accuracy is achieved by using a distance measurement error correction technique. Usually, ultrasound attenuates according to its travel distance, and this change causes variations of the hardware delay in the ultrasound detection circuit (more precisely, combination of the comparator and the rectifier in Figure 2(a) causes this variation). Since the ultrasonic positioning system assumes that the hardware delay is constant, this change causes a distance measurement error. The distance measurement error correction technique corrects this error by using a previously measured hardware delay. By using precisely placed references, the positioning accuracy of the system is less than 2-3 cm as shown in Figure 7(b).

However, the error accumulation problem seriously degrades this accuracy. As shown in Figure 8(a), the error accumulation is caused by the structural characteristic of the distributed positioning algorithm, in which an unlocated node that can estimate its position using references becomes a new reference. Here we assume that nodes A, B, C and D are precisely positioned reference nodes. Node E determines its position based on nodes A, B, C and D, and node F determines its position based on nodes B, C, D and E. Since the measured distances from node E to nodes A, B, C and D contain measurement errors, the accuracy of the position of node F is degraded in comparison with nodes A, B, C, and D. In the example in Figure 8(a), although node F can utilize precise reference nodes A, B, C, and D, the positioning accuracy of node F is lower than that of node D, since node F utilizes node D. This is because node F accumulates

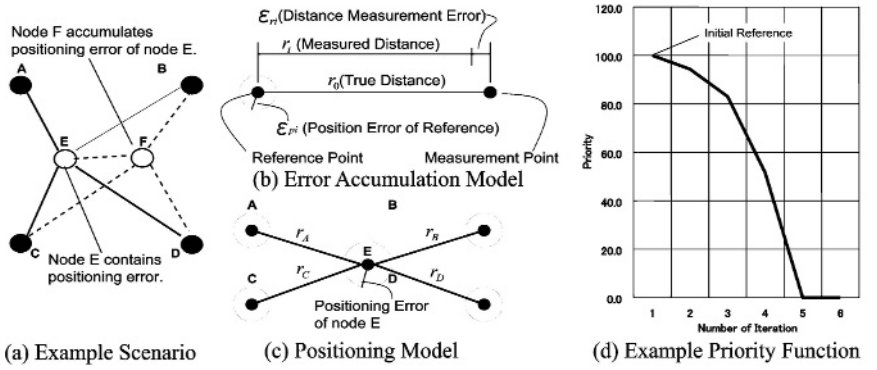


Fig. 8. Error Accumulation Problem

the positioning error of node D. This error accumulation becomes larger as the iterative multilateration proceeds. Moreover, there is a possibility that a sharp rising error accumulation occurs between two nodes. For example, there is the case that node D utilizes node F after node F utilizes node D, and vice versa. In this case the positioning error will be exponentially accumulated between nodes E and F.

To avoid the error accumulation problem, we designed a priority-based node selection algorithm. In this algorithm, we assign priority to all candidates of the reference node system based on the position error of the candidates. Let us consider the error accumulation model shown in Figure 8(b). Here we assume that a measurement point obtains a distance from a reference point which contains the position error ϵ_{pi}. In this case the measured distance r_i is described as:

$$r_i = r_0 + \epsilon_{pi} + \epsilon_{ri} \tag{2}$$

where r_0 is the true distance between the two points, ϵ_{ri} is the error in the measurement process. Here we assume that there is no correlation between ϵ_{pi} and ϵ_{ri} . Then, after a sufficient number of trials, the variance of the measured distance σ^2 becomes:

$$\sigma^2 = \sigma_p^2 + \sigma_r^2 \tag{3}$$

where σ_p^2 is the variance of the position error of the reference, and σ_r^2 is the variance of the measurement error. Now we assume the positioning model depicted in Figure 8(c). According to the concept of PDOP (Position Dilution of Precision) in the global positioning system[13], the positioning error E_p in this model is described as:

$$E_p^2 = PDOP^2 \times \sigma_{UERE}^2 \tag{4}$$

where $PDOP$ is a parameter which is determined by the geometry of the references (see [10] and [13] for more detail about PDOP), and σ_{UERE} is a parameter called the User Equivalent Renege Error (UERE)[13]. Using the error propagation rule, the UERE in the positioning model can be computed as:

$$\sigma_{UERE}^2 = \sqrt{(\sigma_A^2)^2 + (\sigma_B^2)^2 + (\sigma_C^2)^2 + (\sigma_D^2)^2} \tag{5}$$

where σ_A^2, σ_B^2, σ_C^2, and σ_D^2 are variances of the measured distances r_A, r_B, r_C, and r_D, respectively. Assuming $\sigma_A^2 = \sigma_B^2 = \sigma_C^2 = \sigma_D^2 = \sigma^2$, we can obtain the positioning error as follows:

$$E_p = \sqrt{PDOP^2 \times \sqrt{4}(\sigma_p^2 + \sigma_r^2)} \tag{6}$$

However, in the actual positioning, we cannot simply assume $\sigma_A^2 = \sigma_B^2 = \sigma_C^2 = \sigma_D^2 = \sigma^2$, because position error σ_p^2 is usually not the same for four reference nodes in the DOLPHIN system. To handle this problem, we utilize the average mean square error for computing σ_p^2 as shown in the following equation:

$$\sigma_p^2 = \frac{w_A\sigma_A^2 + w_B\sigma_B^2 + w_C\sigma_C^2 + w_D\sigma_D^2}{\sqrt{w_A + w_B + w_C + w_D}} \tag{7}$$
$$(w_A : w_B : w_C : w_D = \frac{1}{\sigma_A^2} : \frac{1}{\sigma_B^2} : \frac{1}{\sigma_C^2} : \frac{1}{\sigma_D^2})$$

Based on equations (6) and (7), we decided on the following formula for assigning priority P to the new reference candidate:

$$\begin{cases} P = 1 - E_p/E_{th} & (E_p < E_{th}) \\ P = 0 & (E_p \geq E_{th}) \end{cases} \tag{8}$$

Here, E_{th} is a design parameter which defines the allowable error in the DOLPHIN system. Figure 8(d) shows an example of the computed priority function, assuming that the position errors of the manually configured references are 0, the distance measurement error is 1 cm, PDOP is constantly 2.0, and the error threshold is 50 cm. As shown in this figure, the priority function decreases according to the iteration times of positioning. Note that, if there are movable nodes in the system, these nodes are forced to use the lowest priority (i.e., zero).

All nodes in the DOLPHIN system that could be reference candidates broadcast their priority by LOC_NOTFY_MSG, and the received priorities are stored in the member list in each node. When a node computes its position, the node utilizes the priorities of the references to determine which reference should be used for the position computation. Needless to say, utilizing references that have a higher priority yields better positioning accuracy. Although the above algorithm considers that only four references are used for the position calculation, there is the case that a node can measure more than four distances from reference nodes with a good priority. In this case, it is possible to utilize a weighted least squares estimation that can compute the position of the node from more than four references. The priority of the references can be used as a weight parameter in the weighted least squares estimation. Though we do not derive the priority function for more than four references, it is easy to extend equations (6) and (7) to such a situation.

Note that the error accumulation problem is inherently unavoidable problem in iterative multilateration. In other words, though our proposed method can

mitigate the error accumulation problem, it cannot completely remove the effect of the error accumulation. This problem becomes serious, if we apply iterative multilateration with minimum number of manually configured references to very large environment. In this case, we have to use sufficient number of manually configured references to obtain acceptable accuracy. Needless to say, there is a tradeoff between accuracy and configuration cost.

4.2 NLOS Reflected Signal Problem

Another error factor in the DOLPHIN system is the no-line-of-sight (NLOS) signal, which is classified into two types: reflected signal and diffracted signal. Basically, our system can correctly locate objects when a receiver node can detect the line-of-sight signal, even if there are reflected or diffracted signals. However, if there are only NLOS signals, such a condition seriously degrades the positioning accuracy of the system. Therefore, we have to detect the NLOS signal and reject it so that the system can provide good positioning accuracy.

According to the physics of acoustic waves, if the size of an obstacle is smaller than the wavelength of the acoustic signal, the diffraction of the acoustic wave is negligible. Since the wavelength of a 40-kHz ultrasonic wave is around 9 mm, we can neglect the effect of the diffracted signal in almost all indoor environments. In addition, generally, reflected signal causes more serious distance measurement erro than diffracted signal. For this reason, we mainly focus on how to reject reflected signals in this section.

Reflected signals arrive at a receiver node in the situation where the direct signal is blocked by an obstacle, and only the reflected signals reach the receiver from the reflector obstacles. Since this reflected signal travels a longer distance than the actual distance between the transmitter and receiver, it seriously degrades the positioning accuracy. To detect and reject the reflected signals, we employed a received-signal-strength-based approach (RSS-based approach). This approach is based on the assumption that the signal strength of a reflected signal is weaker than that of the direct signal because the reflected signal travels a longer distance and is attenuated by the reflector obstacle. That is, we reject the weak signal by comparing the received signal strength to a previously measured reference signal strength.

Figure 9 shows the experimental result of the RSS-based reflected signal rejection technique. As shown in this figure, when the RSSI-based rejection is disabled, the positioning accuracy of the system is seriously degraded with the existence of the reflected signal. On the other hand, the positioning accuracy is dramatically improved when we the reflected signal rejection technique is enabled.

Although this approach is easy to implement, we cannot always reject all reflected signals. As shown in Figure 10, we measured the received signal strength level at various points. The results show that the signal strength level fluctuates largely depending on the geometry of the ultrasonic transmitter and receiver. This is because there are interferences among the reflected signals that form an undulating field of signal strength.

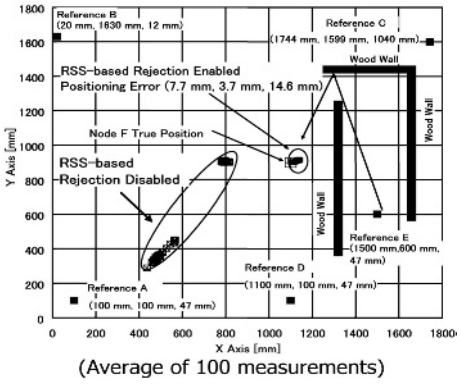


Fig. 9. RSS-based Rejection

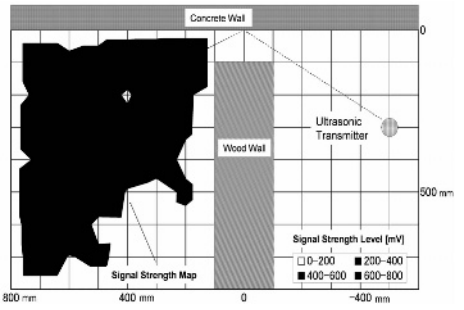


Fig. 10. Example RSS Map

To handle this problem, we designed a geometry-based reflected signal rejection technique. In this technique we assume that a receiver node can detect the direction of the arrived ultrasonic signal. Now we consider the scenario depicted in Figure 11. In this figure, receiver node R measures the distance to transmitter nodes O and P. However, R can hear only the reflected signal transmitted from P. As a result, the signal from P seems to be transmitted from the virtual node V at receiver R. Since we assumed that R can detect the direction of the arrived signal, R can obtain the angle θ_{ORV}. Then, R can compute the distance between node O and V as:

$$d_{OV} = \sqrt{r_O^2 + r_V^2 - 2r_O r_V \cos\theta_{ORV}} \tag{9}$$

On the other hand, as we stated in Section 3.2, since the positions of nodes O and P are notified via LOC_NOTIFY_MSG, node R also can compute the distance between O and P, denoted as d_{OP}. Therefore, we recognize that there is a reflected signal by comparing distances d_{OV} and d_{OP}. If node R can hear one more signal from node Q, then we can specify that the transmitted signal from node P is a reflected signal.

Note that the geometry-based technique does not always work well since it is based on the assumption that there is sufficient number of direct paths from neighboring nodes. Moreover, it is necessary to compare all combinations of nodes for detecting reflected signals. However it is possible to improve the algorithm based on the empirical knowledge that an ultrasonic pulse which traveled shorter distance is more reliable. We have implemented the above algorithm in our device. In our implementation, we detect coarse angle of arrival of received signal based on difference of received signal strength among ultrasonic transducers. We also verified that this algorithm could reject reflected signals in some cases. However, the resolution of the detecting angle was too low to reject all reflected signals. We hope to improve the resolution of the angle estimation in our future work.

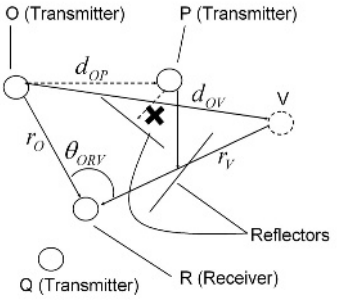


Fig. 11. Geometory-based Approach

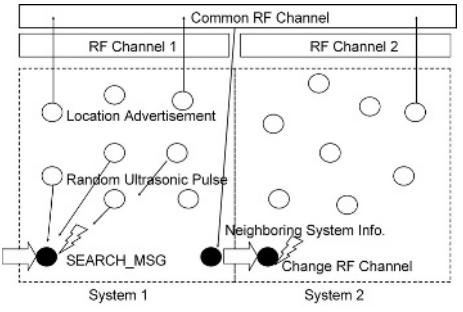


Fig. 12. Scalability Problem

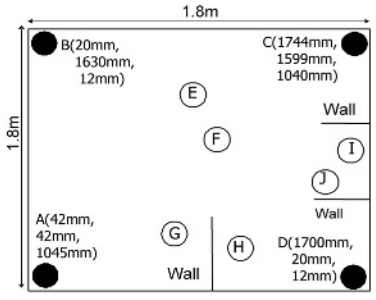
4.3 Scalability

Generally, the limitation of scalability in the ultrasonic positioning system is caused by the slow propagation speed of ultrasound. Usually, an ultrasonic positioning system utilizes the RF channel for time synchronization of the system, then measures the propagation time of an ultrasonic pulse to compute the distance between an ultrasonic transmitter and receiver. The propagation time is about 3 milliseconds to travel 1 m. If signal processing is taken into account, the time for one distance measurement would be more than 10 milliseconds. To avoid signal collision and interference, only one ultrasonic transmitter at a time should be permitted to send an ultrasonic pulse, and in that case the maximum positioning speed would be around 50 times per second. If there are 50 transmitters in the same area, the average interval time to transmit an ultrasonic pulse is degraded to 1 second for each transmitter. Since all devices in the DOLPHIN system send and receive an ultrasonic pulse to run iterative multilateration, this limitation causes deployment scalability.

One effective approach to handle this scalability problem is to divide a big group of nodes into multiple small groups based on some dynamic clustering algorithms that have been proposed in various researches on wireless sensor networks. However, utilizing a complicated and dynamic algorithm is not suitable for developing practical system since such an algorithm causes unexpected instability in the system.

From this point of view, we employ a simple approach in which multiple DOLPHIN systems are running in different RF channels that are schematically based on floor plans. Even if we employ such an approach, we must consider how a newly added node joins a running system and how a movable node changes to another RF channel when it enters a different system.

Figure 12 shows the procedure to add a new node to a running system. When a new node is present, it sends a SEARCH_MSG to every channel randomly. On receiving the message, all DOLPHIN devices in the same channel transmit an ultrasonic pulse with random delay. The new node counts the number of received pulses, and selects the channel in which the node could receive the largest number of pulses. Then the node enters the system by following the



Node	Positioning Error [mm] (95% CDF)	
	Primitive Iterative Multilateration	Node Selection+ NLOS Rejection
E	980	24
F	1090	17
G	380	24
H	550	13
I	530	18
J	970	43

Fig. 13. Experimental Result in Desktop Testbed

bootstrapping algorithm described in Section 3. Each DOLPHIN system shares a common channel to exchange information among systems. This common channel is used for changing RF channels when a movable node enters a new system. As shown in Figure 12, all nodes in all systems are randomly transmitting their location and RF channel via the common channel. The movable node hears this channel and knows what channel is used in the next system. When the node enters the new system, it changes RF channels for that system.

Note that the abovementioned procedure may cause ultrasonic signal collisions at the boundary between two independent systems. However, the time for sending ultrasonic pulses is quite short, and the valid measured distance in our system is limited to 3 m. Therefore, the probability of signal collisions would be negligible. Moreover, we can reject the ultrasonic signal from a neighboring system by using the statistical method and geometry-based reflected signal rejection technique described in the previous section.

5 Performance Evaluation

5.1 Positioning Performance in Ideal Environment

In this experimentation, we placed 4 reference nodes (A-D), 6 unlocated nodes (D-J) and several obstacles in our "desktop testbed" as shown in Figure 13. The size of the testbed is 1.8 meters square, and we placed the 4 reference nodes so that unlocated nodes can obtain good PDOP. When a node is placed inside the boxy area which is constructed from the reference nodes, PDOP becomes around 1.5. In this experimentation, 100 measurements of positioning error are performed with and without techniques for archiving high accuracy positioning. 13 shows the experimental results of positioning error obtained from cumulative distribution function (CDF). Example CDF of node F in this experiment is shown in Figure 14. In our experimentation, we utilized error at 95 % CDF to evaluate accuracy of positioning. Without the techniques, various error factors, such as error accumulation and NLOS signals, seriously degrade positioning accuracy. Estimated location information is no longer acceptable for location-aware

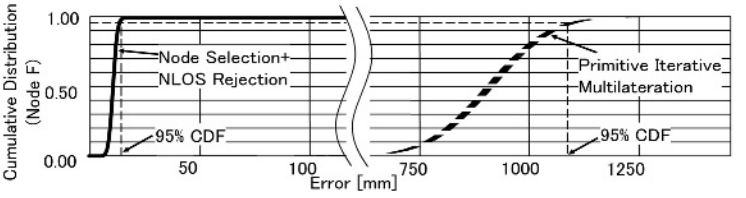


Fig. 14. Example Cumulative Distribution Function (Node F)

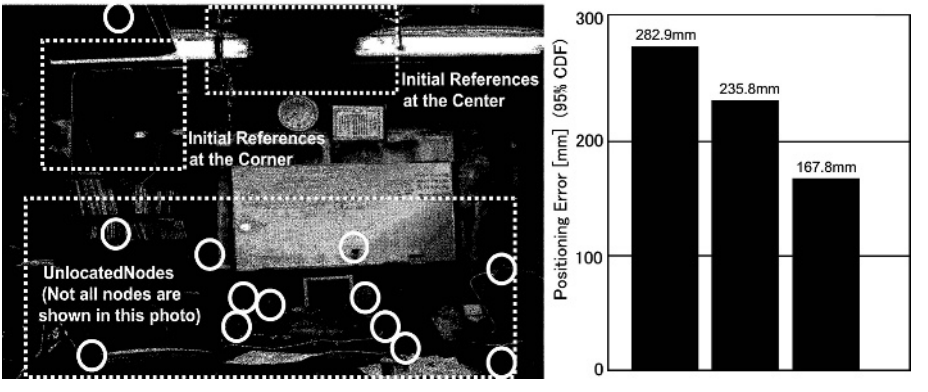


Fig. 15. Experimental Results in Room Environment

applications. On the other hand, positioning accuracy is dramatically improved when we apply all techniques for high accuracy positioning (the worst-case positioning error is less than 5 cm). These result show that designed system can basically provide good positioning accuracy compared to primitive iterative multilateration.

5.2 Effect of Geometry of Initial Reference Nodes

Next, we distributed 24 nodes in a small room (3.6 m x 3.6m x 2.4 m) in our laboratory, as shown in Figure 15. First, we placed 20 unlocated nodes randomly in the room and attached 4 preconfigured references in the corner of the room (Condition (a)). Under this condition, we ran the system with all techniques for high-accuracy positioning. Next, we placed the references at even intervals on the center of the ceiling to improve the PDOP (Condition (b)). The placement of the unlocated nodes was the same as in the above experiment. Finally, we utilize 6 references and 18 unlocated nodes to investigate how the number of references affects positioning accuracy (Condition (c)). In every condition, we performed 100 measurements of positioning error. Figure 15 shows the measurement result. In the case of condition (a), the positioning error was 282.9 mm. The positioning error of the nodes placed around the opposite corner is over 1 m, whereas that of

the others is around 10-20 cm. This is mainly because the PDOP of these nodes is quite poor and the accumulated error seriously increased (actually, the PDOP of these nodes was over 10). In the condition (b), the average positioning error of all the nodes was improved compared to condition (a) since this condition yields good PDOP. In the condition (c), the positioning accuracy was improved greatly as compared to previous two conditions. This result shows that utilizing 6 references enhances the chance to obtain good PDOP for each unlocated nodes and the positioning error is reduced as the result.

The above results mean that the geometry of the initial reference nodes seriously affects the positioning error of the system. Therefore, we should carefully place the initial references so that they provide good PDOP for the unlocated nodes. Finding the best way for determining positions of initial references in actual environment is our future work.

6 Summary and Further Work

We implemented and evaluated a fully distributed indoor ultrasonic positioning system in this paper. We designed a distributed positioning algorithm which implements the idea of iterative multilateration to enable us to reduce the configuration cost of the indoor ultrasonic positioning system, and introduced several techniques for achieving high-accuracy positioning. We showed that our system basically works well through various evaluations in an actual indoor environment.

Despite the success of the basic implementation of our system, there still remain several problems we have to tackle. The most important work will be enabling the system to handle movable objects. Basically, we think that it is possible to track movable objects in our system by using high-speed RF transceiver. However, there will be several technical problems in such dynamic environment. For example, current DOLPHIN node can compute its position only if a sufficient number of distances is obtained. If a node is movable, old measurement results contain distance errors that depend on positioning speed, density of neighboring nodes and speed of the movable node. As the result, positioning accuracy of the movable node will be seriously degraded. Therefore we have to analyze positioning error and develop a viable solution. In designing such a solution, we should consider, for example, resource allocation scheme for movable nodes or integration of other sensors and our system [14].

Another important task will be the stabilization of ultrasonic signal reception including the effective rejection of the NLOS signal. Through various kinds of experimentation, we found that stabilizing propagation of ultrasound is quite difficult but important to improve positioning accuracy of ultrasonic positioning systems. However, our current implementation simply utilizes a rectangular pulse for distance measurement, and is unprotected from various kinds of error sources. Therefore, we belive that a more sophisticated signal design and/or new devices such as broadband ultrasonic transducer [15] is necessary to improve the performance of our system.

Acknowledgements
We would like to thank all reviewers for their constructive comments and invaluable help in shaping this work.

References

1. J. Hightower and G. Borriello, Location Systems for Ubiquitous Computing, IEEE Computer, vol. 34, no. 8, Aug. 2001, pp. 57–66.
2. R. Want, A. Hopper, V. Falcao and J. Gibbons, The Active Badge Location System, Trans. on Information Systems, Vol.10, No. 5, Jan. 1992, pp. 42–47.
3. M. Addlesee, A. Jones, F. Livesey, and F. Samaria, The ORL Active Floor, IEEE Personal Communications, Vol.4, No.5, Oct. 1997, pp. 35–41.
4. P. Bahl and V. Padmanabhan, RADAR: An In-Building RF-Based User Location and Tracking System, Proc. IEEE INFOCOMM 2000, Mar. 2000.
5. J. Hightower, C. Vakili, G. Borriello, and R. Want, Design and Calibration of the SpotON Ad-Hoc Location Sensing System, unpublished, August 2001.
6. BlueSoft Inc. Website, http://www.bluesoft-inc.com/.
7. A. Ward, A. Jones and A. Hopper, A New Location Technique for the Active Office. IEEE Personal Communications, Vol. 4, No. 5, Oct. 1997, pp. 42-47.
8. N. Priyantha, A. Miu, H. Balakrishnan and S. Teller, The Cricket Compass for Context-aware Mobile Applications, Proc. ACM MOBICOM 2001, Jul. 2001.
9. Hexamite Website, http://www.hexamite.com/.
10. A. Ward, Sensor-driven Computing, PhD thesis, University of Cambridge, 1998.
11. A. Savvides, C. Han and M. Srivastava, Dynamic Fine-Grained Localization in Ad-Hoc Networks of Sensors, Proc. ACM MOBICOM 2001, Jul. 2001.
12. R. Harle and A. Hopper, Building World Models by Ray-Tracing, Proc. UBICOMP 2003, Oct. 2003.
13. E. Kaplan Eds., UNDERSTANDING GPS: PRINCIPLES AND APPLICATIONS, Artech House, 1996.
14. A. Smith, H. Balakrishnan, M. Goraczko, and N. Priyantha, Tracking Moving Devices with the Cricket Location System, Proc. MOBISYS 2004, June 2004.
15. M. Hazas and A. Ward, A Novel Broadband Ultrasonic Location System, Proc. UBICOMP 2002, Sept. 2002.