Communication Timing Control and Topology Reconfiguration of a Sink-Free Meshed Sensor Network With Mobile Robots

Junji Takahashi, Takuya Yamaguchi, Kosuke Sekiyama, and Toshio Fukuda, Fellow, IEEE

Abstract—This paper deals with a unified system of fully distributed meshed sensor network and mobile robot cooperation that serves as a sink node. The meshed sensor network in this paper is composed of static wireless nodes, and is capable of fully distributed peer-to-peer (P2P) ad hoc communication with ZigBee-based protocol. A novel communication timing control employing coupled-oscillator dynamics, named phase-diffusion time-division method (PDTD), has been proposed so far, aiming at realization of an ad hoc collision-free wireless communication network. In this paper, we extend the basic PDTD so that it can exhibit flexible topological reconfiguration according to the moving sink node (robot). Unlike conventional sensor network, no static sink node is supposed inside the network; however, a mobile robot will function as a sink node and access the mesh network from an arbitrary position. A large-scale experiment was conducted, and its results show that satisfactory collaboration between the mesh sensor network and the mobile robot is achieved, and the proposed system outperformed the carrier-sense-multiple-access-based sensor system.

Index Terms—Communication timing control, mesh sensor network, mobile robot collaboration, topology reconfiguration.

I. INTRODUCTION

One of the practical applications of multiple mobile robot systems is that they make possible cooperative information gathering from an unknown or partially known environment. One of the research domains, simultaneous localization and map building (SLAM) [1]–[3], demonstrates the classical problems of map building and exploration, but a number of technical challenges have to be solved in order to achieve accurate localization of mobile robots without relying on externally prepared landmarks. Another example is network robotics, which aims at providing daily-life services to human beings by means of a single or a multirobot system, and where the information necessary for task implementation is assumed to be available through the established wireless local area network (WLAN). However, in order to cope with disasters, during which the network infrastructure is unlikely to function normally, more robust and resilient communications and sensor networks are required. A network of this type has to be fully autonomous and should utilize a distributed system that will not need a centralized control. It should also be accessible from an external system.

In an attempt to realize a fully distributed peer-to-peer (P2P) connection, mesh network has been proposed recently [4]. This network is expected to provide a ubiquitous platform for long-term and wide-range surveillance systems that do not require a preestablished static wireless network infrastructure. Instead, the mobile sensor network [5] is composed of a number of mobile sensory nodes or robots, which are mutually linked with peripheral nodes through wireless communication. Due to the factors related to node mobility, new research topics have arisen in terms of optimal node allocation [6]–[8] according to task environments, network topology reconfiguration [9], [10], energy saving issues related to mobile sensor nodes [11], etc. It is clear that the aforementioned research domain consists of vast interrelated technical issues. In this paper, despite the lack of a fully capable mobile robot, we focus on the issue of communication timing control and the adaptive network topology reconfiguration. In wireless communication, the presence of wireless communication is the most fundamental cause of reducing communication efficiency. This becomes more apparent as the number of communication nodes increases. Previous research has shown that the carrier sense multiple access (CSMA) [12]–[14] based communication protocol will suffer from degraded throughput when dealing with more than 20 nodes, even though it supports fully distributed communication. Time-division multiple-access (TDMA) [15]–[18] systems, on the other hand, can avoid the network collision issue, but require a centralized access point to manage communication slot assignment.

In our previous research, we proposed a novel communication timing control method named phase diffusion time division (PDTD) [19], [20], in which a simple oscillator is embedded into each communication node, and each node is allowed to transmit its data signal when its time developing phase is in its assigned phase interval. Signals of this type are exchanged among the peripheral nodes, and local interaction self-organizes an appropriate phase difference pattern that eliminates communication collision. The coupled phase oscillator dynamics method has been implemented for synchronization or timing pattern formation. In the biomimetic research field, several studies have addressed generating biped or quadruped or modular robot locomotion by using coupled neural oscillators [21]–[23], where phase differences between joint angles are determined by.
interactions among connected nonlinear oscillators. These methods are promising because they are asynchronous, decentralized, and suitable for communication timing control for mesh sensor networks. A more detailed review of the method is provided in the following section. Based on the PDTD, we propose a unified collaboration system consisting of a static meshed sensor network and a mobile robot, which will serve as a mobile sink node for the sensor network. In this system, the fixed phase interval for signal transmission of our previous PDTD research will need to be adjusted according to the network topology because the necessary packet size for data transmission will differ depending on the relative position of the node in the network. If a node is located in the edge of the network, it will not need to handle large amounts of data, but if it is located close to the sink node, it will be subject to increased data flow, and thus, should be assigned more communication phase intervals. In this paper, we extend the PDTD concept so that it can execute self-tuning to create the necessary communication phase interval, in accordance with network topological changes. Our proposed systems will not assume the existence of a static sink node, even though conventional sensor networks normally utilize a fixed sink node for data collection purposes. Instead, a mobile robot capable of accessing the meshed sensor network while moving will serve as a sink node, and the network topology will modify itself in order to maintain the communication link to the robot. Additionally, the topological changes to the mesh network will promote the adaptation information needed for the required communication phase interval. A large-scale experiment, described later, will demonstrate the efficiency of this system type.

In Section II, we will describe some of the technical issues related to the sink-free meshed sensor network. As was explained before, we previously developed PDTD, which is a novel communication timing control for wireless sensor networks. In this section, we will review its basic mechanism and propose an extension to the communication timing control. In Section III, our mobile robot platform, including our recently developed multi-hop teleoperation system for distributed autonomous robots, is described. In Section IV, a large-scale communication timing experiment is conducted, during which a throughput comparison is made between conventional CSMA systems and our proposed PDTD-based approach in the case of a fixed sink node. In Section V, in addition to the results of the aforementioned experiment, we validate the sink-free sensor network and its ability to achieve coordination with mobile robots.

II. SINK-FREE MESHED SENSOR NETWORK INTERACTING WITH MOBILE ROBOTS

A. Communication Timing Control for Meshed Sensor Network

We will begin this section with a brief review of the concept and the mechanism of PDTD.

1) Coupled Phase Dynamics: PDTD provides communication timing control based on phase dynamics for collision avoidance in a mesh sensor network (Fig. 1). Node $i$ interacts with virtual nodes and forms appropriate phase difference pattern (Fig. 2). Let $\tilde{\theta}_{ij}$ denote phase value of virtual node $j$ for node $i$.

2) Stochastic Adaptation: When relying on repulsive interaction alone, the phase difference pattern often fails to converge to the desired stationary state. Therefore, a stochastic adaptation term, which is determined by the collision risk, $\xi(S_i)$, is introduced.

Then the governing equation is given by following equations:

$$\frac{d\theta_i}{dt} = \omega_i + \sum_{j \in K_i} k_j R(\Delta \tilde{\theta}_{ij}) + \xi(S_i) \quad (1)$$

$$\Delta \hat{\theta}_{ij} = \tilde{\theta}_{ij} - \theta_i \quad (2)$$

$$\frac{d\Delta \hat{\theta}_{ij}}{dt} = \tilde{\omega}_{ij} \quad (3)$$

where $\omega_i$ and $\tilde{\omega}_{ij}$ denote the angular velocity of node $i$ and virtual node $j$, respectively, $k_j$ is coupled strength value, and $K_i$ is a virtual node set of node $i$. Every node is allowed to transmit data for $\phi_c/\omega_i$ (in seconds) every cycle and $\xi(S_i)$ is a stochastic term, detail of which is explained in Section II-A.2.

Interaction with the neighbor nodes is governed by the phase response function $R(\Delta \hat{\theta}_{ij})$, which is a repulsive function as follows:

$$R(\Delta \theta_{ij}) = \begin{cases} \Delta \theta_{ij} - \phi_c, & \Delta \theta_{ij} \leq \phi_c \\ 0, & \phi_c < \Delta \theta_{ij} < 2\pi - \phi_c \\ \Delta \theta_{ij} - 2\pi + \phi_c, & 2\pi - \phi_c \leq \Delta \theta_{ij} \end{cases} \quad (4)$$

2) Stochastic Adaptation: When relying on repulsive interaction alone, the phase difference pattern often fails to converge to the desired stationary state. Therefore, a stochastic adaptation term, which is determined by the collision risk, $\xi(S_i)$, is introduced.

The phase overlap rate is defined as an evaluation index. Node communication state is defined so that $O_i = 1$ implies that node $i$ is allowed to communicate and $O_i = 0$ implies that node $i$ is prohibited to communicate, and it is given by

$$O_i(\theta_i(t)) = \begin{cases} 1, & 0 \leq \theta_i < \phi_c \\ 0, & \phi_c \leq \theta_i < 2\pi \end{cases} \quad (5)$$
The flag function included to indicate phase overlap of communication timing between node $i$ and virtual nodes is given by

$$x_i(t) = \begin{cases} 1, & O_i(\theta_i) = 1 \text{ and } \sum_{j \in K_i} O_j(\theta_{ij}) > 0 \\ 0, & \text{else} \end{cases}$$

(6)

Here, $x_i = 1$ indicates that there is the phase overlap that would cause a collision. If $\sum_{j \in K_i} x_i(t) \neq 0$, then one collision is counted for one cycle. Let $\gamma$ indicate the occurrence time of phase overlap for past $n$ cycles. The overlap rate $c_i$ is given by

$$c_i(t) = \frac{\gamma}{n}.$$  

(7)

The stress of being exposed to the risk of collision accumulates by the following mechanism:

$$S_i(t) = 2S_i(t-\tau) + s(c_i)$$

(8)

$$s(c_i) = \begin{cases} 0.0, & 0 \leq c_i < 0.2 \\ 0.03, & 0.2 \leq c_i < 0.5 \\ 0.05, & 0.5 \leq c_i < 0.8 \\ 0.1, & 0.8 \leq c_i < 0.9 \\ 0.3, & 0.9 \leq c_i \\ 1, & \text{otherwise} \end{cases}$$

(9)

where $\tau = nT_i$ is a stress accumulating time scale. Random phase jump is implemented every $nT_i$ (in seconds) cycle with probability $S_i$, where, if $S_i > 1$, then $S_i \leftarrow 1$. After random phase jump, then $S_i \leftarrow 0$. The destination of phase jump is decided as follows: assume that node $i$ has $N_i$ virtual nodes, the phase of which is denoted as $\theta_{ij}$. Sorting the phase value $\theta_{ij}$ in ascending order, such as $\theta_{ij}^{(1)} < \cdots < \theta_{ij}^{(k)} < \cdots < \theta_{ij}^{(N_i)}$, the corresponding node to the $k$th phase value is $v_{ij}$. The destination of stochastic jump is depicted in Fig. 3. The list of destination $u_k$ is given by

$$u_k = \frac{v_{ik} + v_{ik+1}}{2}, \quad k = 1, 2, \ldots, N_i - 1.$$  

(10)

The preferential selection probability $u_k$ is decided by

$$p_k = \frac{\exp(\beta(v_{ik+1} - v_{ik}))}{\sum_{l=1}^{N_i-1} \exp(\beta(v_{il+1} - v_{il}))}, \quad l = 1, 2, \ldots, N_i - 1$$

(11)

where $\beta$ is the sensitivity parameter of the selection.

B. Control Scheme of Feasible Communication Assignment: $\phi_c$ Control

1) Algorithm Overview: As shown in Fig. 4, each node is allowed to communicate when the phase interval of node covers the line segment of $0$ rad. PDTD achieves collision-free communication by coordinating the phase relation of each node, so its phase interval $\phi_c$ is distinct and separate from every other node. In the previous version of PDTD, $\phi_c$ was defined as a constant value; however, its interval had to be modified according to the necessary packet size required for data transmission. When the node is placed in the boundary region of the network and communication load is small, its $\phi_c$ can be reduced. Alternatively, the nodes closer to the sink node will have to process larger numbers of data packets, and hence, should be assigned larger $\phi_c$ intervals. This situation depends on the configuration of the network topology and the present communication load. In this section, we propose the self-tuning algorithm of the $\phi_c$ phase interval, which is designed to cope with dynamic adjustment of $\phi_c$ phase interval and is named $\phi_c$ control.

During the process of organizing a collision-free phase difference pattern, each node interacts with the peripheral nodes within communication range, where the phase information of the other nodes is referred. Based on the accumulated packet data acquired from the other nodes, as well as its own observed sensing data, each node estimates its required packet size and its own phase interval $\phi_c$. If $\phi_c$ is too small, the stored data will be lost after the accumulated packet data extends beyond the queue length defined in advance. There are two ways to gain extra $\phi_c$. One is to advance the transmission start phase and the other is to delay the transmission end phase (finish time of transmission). However, a node extending its own $\phi_c$ interval may collide with another node due to interference or interval overlap of $\phi_c$, since the next node in succession might need the full use of its own $\phi_c$ phase interval. Therefore, a procedure capable of negotiating the necessary $\phi_c$ interval must be considered. When node $i$ needs to extend $\phi_c$, it will send a share request (SREQ) signal to node $j$ where its phase interval $\phi_c$ is adjacent to node $i$. If node $j$ has an extra margin in its own $\phi_c$, then node $j$ will issue a share reply (SREP) that contains information on the available margin of $\phi_c$. Otherwise, no reply is returned. The reception of SREP will allow node $i$ to extend its transmitting start phase by the designated phase interval. The parameter set used for $\phi_c$ control is listed in Table I. The superscript of the parameters indicate the cycle number. The stored packet in the node will be updated as follows:

$$q^{n+1}(i) = q^n(i) + p^\text{in}(i) - p^\text{out}(i)$$

(12)

$$\Delta \theta_{ik} = |\Delta \phi_{ik} - \phi_c(k)| - 2 \times \phi_d.$$  

(13)

2) Arrangement of the End Phase: The end phase of node $i$ is arranged by the following procedure (Fig. 5). In this procedure, by using a $\phi_c$ determining criteria described in E3, node $i$ releases the unnecessary communication resource so that other nodes can use it.
TABLE I
DEFINITION OF PARAMETERS FOR \( \phi_c \) CONTROL

<table>
<thead>
<tr>
<th>Parameter</th>
<th>Description</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>( \phi_{\text{in}}(i) )</td>
<td>Phase interval allowed for data transmission</td>
<td>( 0 \sim 2\pi ) [rad]</td>
</tr>
<tr>
<td>( \phi_{\text{d}}(i) )</td>
<td>Required phase interval of node ( i )</td>
<td>( 0 \sim 2\pi ) [rad]</td>
</tr>
<tr>
<td>( \phi_{\text{c}}(i) )</td>
<td>Available phase interval of node ( i )</td>
<td>( 0 \sim 2\pi ) [rad]</td>
</tr>
<tr>
<td>( \phi_{\text{c}}^\prime(1) )</td>
<td>Phase interval for 1 packet transmission</td>
<td>( 0 \sim 2\pi ) [rad]</td>
</tr>
<tr>
<td>( p_{\text{in}}^n(i) )</td>
<td>Number of packet input in ( n_{\text{th}} ) cycle</td>
<td>0 \sim</td>
</tr>
<tr>
<td>( p_{\text{out}}^n(i) )</td>
<td>Number of packet output from data queue</td>
<td>( q_{\text{max}} )</td>
</tr>
<tr>
<td>( q^n(i) )</td>
<td>Number of stored packets</td>
<td>( q_{\text{max}} )</td>
</tr>
<tr>
<td>( \Delta \theta_{ij} )</td>
<td>Phase value of node ( j ) for node ( i )</td>
<td>( 0 \sim 2\pi ) [rad]</td>
</tr>
<tr>
<td>( \phi_{\text{c}}(k) )</td>
<td>Phase margin between node ( i ) and node ( k )</td>
<td>( 0 \sim 2\pi ) [rad]</td>
</tr>
</tbody>
</table>

Fig. 5. Arrangement of the end phase.

\[
E1) \text{Calculate the required phase interval as a sum of inputting and stored packets:}
\[ \hat{\phi}_{\text{in}}(i) \leftarrow \phi_{\text{d}} \times (q^n(i) + p_{\text{in}}^n(i)) \]

\[
E2) \text{Calculate available phase interval from the phase difference of phasic nearest node:}
\[ \hat{\phi}_{\text{c}}(i) \leftarrow \phi_{\text{d}} \times (\triangle \theta_{ij} - 1) \]

\[
E3) \text{Set the transmitting phase interval to the smaller required phase interval and available phase interval:}
\[ \phi_{\text{c}}^\prime(i) \leftarrow \min(\hat{\phi}_{\text{in}}(i), \hat{\phi}_{\text{c}}(i)) \]

3) Arrangement of the Start Phase: When a node cannot obtain sufficient size of \( \phi_{\text{c}} \), it arranges its start phase using the following procedure (Fig. 6).

S1) Node \( i \) sends SREQ to node \( k \), which is nearest of phasic adjacent nodes
\[
\Delta \theta_{ij} \leftarrow \min_{j \in K}(\triangle \theta_{ij})
\]

S2) Node \( k \) sends SREP that contains the information of the margin to node \( k \), if sufficient margin exists the margin:
\[
\Delta \theta_{ik} > 0.
\]

S3) Node \( i \) puts the start phase ahead for the margin.

When a node is unable to obtain sufficient \( \phi_{\text{c}}(i) \) after S3 and when it has not received an SREP, it resends an SREQ until it gets sufficient \( \phi_{\text{c}}(i) \). At S2, when node \( k \) does not have sufficient margin, it cannot send an SREP. In this situation, node \( k \) sends an SREQ to start the phase arrangement process itself for node \( i \), even if it has sufficient \( \phi_{\text{c}}(k) \). And if node \( k \) receives an SREP to advance the start phase, it implies that it has obtained sufficient margin. At this time, node \( k \) sends an SREP to node \( i \). This start phase arrangement process spreads outward to the other nodes repeatedly until node \( i \) receives an SREP providing sufficient \( \phi_{\text{c}}(i) \).

III. MULTIHOP TELEOPERATION CONTROL PLATFORM OF MULTIROBOT SYSTEMS

A. Mobile Robot System

The interaction of experimental mobile robots with a mesh sensor network is shown in Fig. 7. The mobile robot consists of five modules: the decision-making module is a PC (VAIO type-U, Sony Corporation, Japan), the P2P communication module is a ZigBee device, and the mobile module is an Amigobot and Pioneer3AT (MobileRobots, Inc.). The ZigBee device and the robot are connected to the PC through RS232 and USB connections. (The cricket and the pan-tilt-zoom camera have not been used so far.)

B. Sink Controller GUI

The sink controller display is shown in Fig. 8. The robot deployment window shows the position of all robots and the communication links toward the sink node. An operator can monitor the network easily from this window. Furthermore, observation target information such as position should be drawn on this window.
Furthermore, an operator can control any robot from this controller. When an operator mouse clicks on the robot in robot deployment window and then clicks the area that he/she wants to observe, the sink controller generates a control packet and sends it to the selected robot. Intermediary robots will forward this packet to the destination robot. Thus, the operator can control any robot by using the sink controller, even if the target robot (mobile node) is out of the communication range of the sink node.

The sensing data window plots the sensing data that were collected at the target point and transmitted through the multihop network based on the time line. By analyzing this graph, an operator can evaluate the throughput and packet loss rate of a mobile robotic sensor network.

C. Experiment of Multihop Teleoperation

1) Experimental Setup: During the next phase, a multihop teleoperation experiment was conducted using a robot system and the sink controller. The purpose of the experiment is to observe a target point in the experimental field that is selected by an operator. When the target point is out of communication range, robots move to target area while maintaining communication links with the sink node. Detail of the self-deployment algorithm and the routing algorithm are described in [7].

The experimental setup and the display of the sink controller are shown in Fig. 9. The origin and the $x$- and $y$-coordinates are defined as shown in the figure. The target positions are Target1(1, 15) and Target2(2, 6). The sink node is on the origin. There are five mobile robots, which are numbered from 4 to 8. The communication range $R$ and the stable connection range $r$ of each robot are set to $R = 4.0$ m and $R_{st} = 3.0$ m, respectively. Since no localization technique has so far been equipped, each robot determines its position by dead reckoning. Each robot sends a beacon signal that includes position information about itself every 2 s and an s-beacon transmission for building a multihop communication route to the sink every 4 s. In the future, when a robot reaches its target, it will start collecting sense data. However, in the current stage, since the robots are not equipped sensors, the robot virtually generates sine wave data every 0.1 s to simulate actual sensing. It then sends a data packet, which consists of the virtual sensing data and time to the sink node through the constructed multihop communication route. When the sink controller receives data packets, it plots the virtual sensing data on the sensing data window in accordance with the time axis. The experimental plan is shown as follows.

Step 1: Power on the robot to start communication.
Step 2: Assign the position of target 1 on the sink controller.
Step 3: Instruct robots to begin moving.
Step 4: When a robot reaches target 1, it transmits the data packet to the sink.
Step 5: After the sink controller received a number of data packets, assign the position of target 2.
Step 6: Robots start to redeploy to target 2.
Step 7: When a robot reaches the target 2, it transmits data packets.

![Fig. 9. Experimental result of multihop teleoperation.](image)

2) Experimental Results: The experimental process and the display of the sink controller display are shown in Fig. 9. The self-deployment process used to construct the multihop network between the sink and target 1 is confirmed from the first three pictures. In the third picture, robot 7 reaches target 1, and the sink controller displays the present location of the robots, the communication links, and the collected sensing data. When the operator orders the robots to move toward target 2 (in the fourth picture), all the robots start to move toward target 2. This confirms that the control command packet was sent to each robot without packet loss. Therefore, the down-current communication route was built appropriately by the routing algorithm. This demonstrates the validity of the proposed self-deployment
Algorithm. In addition, the up-current communication route was also found to be built appropriately by the routing algorithm.

The arrival rate of the data packet is approximately 50%–80%. Most of the packet loss can be attributed to wireless communication, because no timing control technique has been incorporated in this experiment. When the robots utilize the proposed PDTD method, they will be able to communicate with less packet loss and the communication throughput will increase.

IV. EXPERIMENTAL RESULT 1: $\phi_c$ CONTROL WITH FIXED SINK NODE

A. Experimental Setup

This section first deals with $\phi_c$ control with a fixed sink node. As shown in Fig. 10, the wireless communication node employed in this experiment is UDv4, and the preliminary experimental result indicating the relation between the receive signal strength indication (RSSI) of UDv4 and the internode distance $d$ is shown in Fig. 11, where it can be seen that the successful arrival rate of the transmitted signal will remain over 95% until $d < 5$ m, but sharply decline when $d > 6$ m is reached because the detectable minimum radio wave strength of UDv4 is $-90$ dbm. The configuration of the experimental sensor network is shown in Fig. 12. The specifications of UDv4 are described in Table II. In this experiment, the internode distance $d$ is determined so that a transmitted signal can successfully reach the region of $d$, but will not be able to reach when the distance is over $2d$. From Fig. 11, deployed on a square lattice, the internode distance is $d = 3.5$ m. The sink node that is a data gathering node is connected to the observation PC through the USB extension cable. All the packets arriving at the sink node are forwarded to the PC.

The terminal nodes send a control packet (called beacon) and several data packets allowed in the phase interval $\phi_c$ in one cycle. The beacon packet contains information such as node ID and present phase value. Other nodes use this information for phase arrangement of the PDTD. Since SREQ and SREP used for $\phi_c$ control is included in the beacon, the execution of $\phi_c$ control will not influence the communication resource. Most of the nodes assume the role of intermediate node. They receive the data packets from downstream nodes in their waiting phase and forward the received packets in their transmitting phase. All nodes attempt to send the data packets during transmitting phase $\phi_c$. However, if the stored data packets are too large to be sent in one transmission by the phase interval $\phi_c$, such data will remain in the node until next cycle transmission. Additionally, if the stored packets increase to beyond maximum queue length, they are erased sequence, beginning with the oldest ones. The maximum queue length is set to $q_{\text{max}} = 100$. In all experiments, the control cycle length, which is the time interval for the sending beacon, is $T = 3.0$ s. The traffic cycle length, which is the interval of generating packets, is defined as $T_p = 3 \pm \zeta$ s ($0 \leq \zeta \leq 0.5$), considering a stochastic fluctuation.

The sink controller that we have developed is shown in Fig. 13. It can efficiently display such sundry information as

<table>
<thead>
<tr>
<th>Table II: Specifications of UDv4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Type of wireless communication</strong></td>
</tr>
<tr>
<td><strong>Frequency band</strong></td>
</tr>
<tr>
<td><strong>Channels</strong></td>
</tr>
<tr>
<td><strong>Wireless intensity</strong></td>
</tr>
<tr>
<td><strong>Maximum data rate</strong></td>
</tr>
<tr>
<td><strong>Maximum operation range</strong></td>
</tr>
<tr>
<td><strong>MPU/Clock</strong></td>
</tr>
<tr>
<td><strong>ROM/RAM</strong></td>
</tr>
<tr>
<td><strong>size</strong></td>
</tr>
</tbody>
</table>
Fig. 14. Topology configuration: the numbers located on right upper side of each node indicates the communication load. In this experiment, since the traffic interval is the same as the control interval, the size of communication load is according to the number of downstream nodes. Regarding node A324, its communication load is 5 at topology A, 11 at topology B, and 19 at topology C.

Fig. 15. Queue length and φc of node A324; after the time, which are topology changing timing (t = 1200, 1800 s), although the node increases the queue length primarily, it gets the enough φc according to the communication load and does not store amount of sending queue.

The conditions of each nodes and communication links. Additionally, an operator can change experimental configuration of each node by using the sink controller. When the operator sends a setting packet, it gets transferred to the destination node by means of multihop communication. The sink controller refers the data packets, which are also transferred with multihop communication, to display relevant information. However, the use of multihop communication tends to increase the time delay. Therefore, the information displayed in the sink controller shows the node conditions of 27 s ago at the latest.

The purpose of the φc control is to make PDTD more adaptable to dynamic environment where the communication load will vary. In this experiment, a network topology is shifted at prescribed intervals in order to vary the communication load. The topologies of each interval are shown in Fig. 14. The interval of variation is 600 s, and the topologies sequence is $A \rightarrow B \rightarrow C \rightarrow B$. The total experimental duration is 2400 s.

### B. Experimental Result

First of all, regarding node A324, it should be made sure that φc control is working well. The graph in Fig. 15 shows the size of transmitting phase interval: $\phi_c(A324)$ shows the size of queue length, and $q(A324)$ shows the counts of receiving SREP and the size of expected $\phi_c$ based on topology. The phase disks in Fig. 15 show the phase relation between node A324 and other interacting nodes at the given times.

From the transition of these parameters, it can be confirmed that the $\phi_c$ control is working well. At $t = 600$ s, when the network topology is changing from $A$ to $B$, the size of communication load of node A324 increases. However, since the size of $\phi_c(A324)$ (see the phase disk at time = 709) is too short to send every packet, the queue length increases.

The phase disk at time = 709 indicates that node A324 cannot arrange the transmitting finish phase because of the existence of other nodes. However, the start phase arrangement is activated to instruct node A324 to increase its size $\phi_c(A324)$. Then the size of node A324 becomes sufficient $\phi_c(A324)$ (see the phase disk at time = 715) and it sends its queued stored packets in a burst. A few seconds later, it confirms that the queue size becomes 0.

The counters on the receiving SREP line in the graph indicate that the interchange of SREQ and SREP has been accomplished. When node A324 has reduced its sending queue to 0, it arranges the transmitting finish phase in a way that releases the communication resource for use by other nodes (about time = 800). In a similar fashion, after time $t = 1200$, $1800$ s, which are topology time changes, even though the node primarily increases the queue length, it obtains sufficient $\phi_c$, which is close to the expected $\phi_c$ and does not store amount of sending queue. It was confirmed that $\phi_c$ control is working well in other nodes.

In the graph, in situations when $\phi_c$-controlled PDTD is used, even though throughput declines temporally when the time topology changes ($t = 600, 1200, 1800$ s), it returns to high value levels a few seconds later. Additionally, the phenomenon whereby throughput exceeds its theoretical maximum value is shown as an overshoot. This occurs because the node does not send packets indiscriminately. Instead, when the node does not have sufficient $\phi_c$, it stores surplus packets in the sending queue. Then, when the node obtains sufficient $\phi_c$, it transmits the accumulated packets in a burst that resembles dissipating stress. When this occurs, the throughput can exceed the theoretical...
value. This phenomenon provides strong evidence that a sensor network that is working with $\phi_c$-controlled PDTD can handle communication with only minor packet losses.

It is clearly demonstrated that the $\phi_c$-controlled PDTD achieves better result in comparison with CSMA, which operates under the same conditions. During the experiment, the total packet arrival rate (0–2400 s) of $\phi_c$-controlled PDTD and CSMA are 85.4% and 62.8%, respectively. Optimized $\phi_c$ stipulates that the condign size of $\phi_c$ is always provided according to the communication load, which is based on topology changes. In other words, it could be said that the case of optimized $\phi_c$ demonstrates the performance advantages of normal PDTD because of the way network topology information is managed as global information. On the other hand, the $\phi_c$-controlled PDTD arranges the $\phi_c$ using only local information and interaction, yet performs as well or better than optimized $\phi_c$. Consequently, it is strongly anticipated that $\phi_c$-controlled PDTD will prove very effective, not only in the case when the network topology is unknown or might be changing, but also in situation when communication loads change heterogeneously and spatiotemporally due to increasing local observation demands and/or partial breakdowns of the network.

The packet arrival delay graph is shown in Fig. 17. During a transient period of convergence (from $t = 0$ to $t = 400$), because each node, which cannot obtain sufficient $\phi_c$, stores surplus packets in the sending queue, the packets arrival delay becomes excessive. The delay can be large during the period after the topology changes ($t = 600, 1200, 1800$ s). This is because some nodes store packets due to the decreased size of $\phi_c$. However, the average packet arrival delay for every topology generally reaches a certain level (about 5–8), which does not decline. This is largely due to the disposition of multihop communication. However, the delay is also related to the PDTD. The maximum hop number is $n_{\text{hop}} = 9$ and the control cycle is $T = 3.0$ s in this experiment. This means that the maximum packet arrival delay can become $t_{\text{delay}} = n_{\text{hop}} \times T = 27$ s, even if a packet has not been stored into the queue of any node. This is the only fault of the PDTD method so far detected and is the result of the fact that it gives preference to reduction of packet loss over reductions packet arrival delay. Reduction of packet arrival delay will be the primary goal of future research work.

V. EXPERIMENTAL RESULT 2: SINK-FREE NETWORK INTERACTING WITH ROBOT

A. Experimental Setup

The purpose of the experiment is to verify the network topology reconfiguration according to the movement of the sink node and ensure that the $\phi_c$-controlled PDTD works well on a mesh network where the communication load is changing part of the dynamic network topology. The experimental environment is shown in Fig. 18. The configuration of terminal nodes and its arrangement are the same as those used in the experiment of Section IV. The sink node is installed on a mobile robot in order to change its position. The mobile robot is controlled by another WLAN. The movement track of the sink node is shown in Fig. 19. The sink node stays during 600 s at each point and transits between two points at a time about 40 s. The maximum robot velocity is 1.0 m/s. The total experiment time is 3167 s.
In this experiment, simple hop-based routing is applied in order to reconstruct communication routes. The sink node transmits a beacon once per 3.0 s, as do the terminal nodes. A terminal node immediately determines the location of the sink node after receiving its beacon. The node sets the sending destination (hand on) to the sink node and sets its hop number \( h = 1 \). A node that cannot receive the beacon of the sink node sets its hop number \( h = 100 \) (hand off) and searches for nodes with fewer hops than it has. When a node finds out a node with fewer hops than it has, it sets the sending destination (hand on) to the node and sets its hop number \( h = (\text{destination hops}) + 1 \). If the destination node disappears or changes its hop number to a larger value, a node sets its hop number \( h = 100 \) (hand off) and searches for a node with less hops.

**B. Experimental Result**

The reconstructed topology transition according to the sink node is shown in Fig. 20. It was determined that at time \( t = 615 \), 11 nodes set the sink node as a destination, and the number of interacting nodes (\( <2R \)) could possibly exceed 30. In this situation, the nodes have to rearrange communication timing based on the number of interacting nodes due to the collapse of convergence. Furthermore, they have to control the size of \( \phi_c \) in order to reallocate the communication resource.

The response performance of these processes can be estimated from the communication throughput graph. The throughput graph is shown in Fig. 21. This graph shows that throughput declines scienically while the sink node is in transit because these two processes (routing process and \( \phi_c \) control process) cannot keep up with the sink node movement. However, once the sink node reaches its target point, then throughput returns back to its previous high level within a few seconds. Consequently, it can be said that the proposed \( \phi_c \)-controlled PDTD is robust to network changes because it makes throughput pull round, even though it is sufficient to monitor the sink movement. Furthermore, the \( \phi_c \)-controlled PDTD has sufficient scalability to handle the addition of a new node or the loss of nodes due to accidents, because it can deal with the increasing or decreasing interacting nodes on a flexible basis.

**VI. Conclusion**

We have a proposed PDTD system that provides communication timing control based on phase dynamics for collision avoidance in a mesh sensor network. The developed \( \phi_c \)-controlled PDTD that is a self-tuning algorithm of the \( \phi_c \) phase interval of transmission can cope with dynamic adjustment of \( \phi_c \). While organizing a collision-free phase difference pattern, each node interacts with the peripheral nodes within the communication range. Because of this, this algorithm can operate without global information. This indicates that the \( \phi_c \)-controlled PDTD can leverage limitary communication resource not only in situations when the network topology is unknown or might be changing, but also when the communication load changing heterogeneously and spatiotemporally due to increasing of local fluctuations in radio wave strength. For example, at \( t = 632 \) s, 11 nodes set the sink node as a destination, and the number of interacting nodes (\( <2R \)) could possibly exceed 30.

In this experiment, simple hop-based routing is applied in order to reconstruct communication routes. The sink node transmits a beacon once per 3.0 s, as do the terminal nodes. A terminal node immediately determines the location of the sink node after receiving its beacon. The node sets the sending destination (hand on) to the sink node and sets its hop number \( h = 1 \). A node that cannot receive the beacon of the sink node sets its hop number \( h = 100 \) (hand off) and searches for nodes with fewer hops than it has. When a node finds out a node with fewer hops than it has, it sets the sending destination (hand on) to the node and sets its hop number \( h = (\text{destination hops}) + 1 \). If the destination node disappears or changes its hop number to a larger value, a node sets its hop number \( h = 100 \) (hand off) and searches for a node with less hops.
sensing data and/or situations when the part of network has partially broken down. This ability is extremely useful in a mobile sensor network that consists of mobile nodes or a mobile robot.

The validity of the proposed algorithm was demonstrated by two experiments. During the mesh sensor network experiment, it was clarified that \( \phi_c \)-controlled PDTD far exceeds CSMA in communication throughput. The sink-free mesh sensor network experiment showed that the proposed algorithm has sufficient scalability to handle increasing and decreasing node numbers, and it also has robustness against node accidents. However, one fault detected in the proposed method is that arrival delays have a tendency to increase. Our future works will involve reducing packet arrival delay and installing the \( \phi_c \)-controlled PDTD system on the robot system of a mobile robotic sensor network.

**ACKNOWLEDGMENT**

The authors thank T. Aoyama, M. Ito, M. Hokkoku, M. Kojima, D. Kato, K. Fujita, and M. Rokunuzzaman for their very hard work during the experiments when the temperature sometimes reached 39 °C. The cooperation of members of Fukuda Laboratory is also acknowledged.

**REFERENCES**


Kosuke Sekiyama received the B.E., M.E., and Dr. Eng. degrees from Nagoya University, Nagoya, Japan, in 1992, 1994, and 1997, respectively.

In 1997, he joined Nagoya University as a Research Associate, where he is currently an Associate Professor in the Department of Micro-Nano Systems Engineering. From 1998 to 2001, he was a Lecturer at the Science University of Tokyo, Suwa College. From 2001 to 2006, he was an Associate Professor at the University of Fukui. His current research interests include distributed autonomous systems, in particular, self-organizing systems at various system levels, including swarm intelligence, multiagents, ad hoc sensor networks, self-organizing traffic signal networks, evolutionary graph networks, and distributed manufacturing systems.

Toshio Fukuda (M’83–SM’93–F’95) received the B.S. degree from Waseda University, Tokyo, Japan, in 1971, and the M.S. degree in 1973 and the Ph.D. degree in malfunction diagnosis and application of stable adaptive schemes for a nuclear reactor system in 1977 from the University of Tokyo, Tokyo.

From 1977 to 1982, he was with the National Mechanical Engineering Laboratory, Tsukuba, Japan. From 1982 to 1989, he was with the Science University of Tokyo. Since 1989, he has been with Nagoya University, Nagoya, Japan, where he is currently a Professor in the Department of Micro-Nano Systems Engineering. His current research interests include intelligent robotic systems, cellular robotic systems, mechatronics, and micro-nanorobotics.

Dr. Fukuda was the President of the IEEE Robotics and Automation Society (1998–1999), the Director of IEEE Division X, Systems and Control (2001), and the Editor-in-Chief of the IEEE/ASME TRANSACTIONS ON MECHATRONICS (2000–2002), and the President of the IEEE Nanotechnology Council (2002–2005). He is currently the AdCom President of the IEEE Nanotechnology Council.