| PAPER |
| --- |

# Polling Schedule Algorithms for Data Aggregation with Sensor Phase Control in In-Vehicle UWB Networks*

**Hajime MIGITA**[†a], *Student Member*, **Yuki NAKAGOSHI**[††b], **Patrick FINNERTY**[†††c], *Nonmembers*,
**Chikara OHTA**[†††d], *Senior Member*, **and** **Makoto OKUHARA**[†e], *Nonmember*

**SUMMARY**    To enhance fuel efficiency and lower manufacturing and maintenance costs, in-vehicle wireless networks can facilitate the weight reduction of vehicle wire harnesses. In this paper, we utilize the Impulse Radio-Ultra Wideband (IR-UWB) of IEEE 802.15.4a/z for in-vehicle wireless networks because of its excellent signal penetration and robustness in multipath environments. Since clear channel assessment is optional in this standard, we employ polling control as a multiple access control to prevent interference within the system. Therein, the preamble overhead is large in IR-UWB of IEEE 802.15.4a/z. Hence, aggregating as much sensor data as possible within each frame is more efficient. In this paper, we assume that reading out data from sensors and sending data to actuators is periodical and that their respective phases can be adjusted. Therefore, this paper proposes an integer linear programming-based scheduling algorithm that minimizes the number of transmitted frames by adjusting the read and write phases. Furthermore, we provide a heuristic algorithm that computes a suboptimal but acceptable solution in a shorter time. Experimental validation shows that the data aggregation of the proposed algorithms is robust against interference.

*key words:*  *UWB, in-vehicle network, wireless network control system, scheduling optimization, heuristic scheduling, IEEE 802.14.5*

## 1.  Introduction

In-vehicle wire harnesses used for data transmission and power delivery are gaining weight due to the increase in electrical components, reducing fuel efficiency and causing manufacturing and maintenance costs. For example, the weight of a harness can reach 10 kg, 20 kg, and in excess of 40 kg in light, standard, and large cars, respectively. In the era of automated driving, we anticipate an increase in the number of electric devices in motor vehicles, further adding to the weight of wire harnesses. Replacing part of a wire harness with a "wireless harness" is expected to relieve these problems [1].

In collaboration with DENSO TEN Limited and the Advanced Telecommunications Research Institute International (ATR), we have been trying to reduce the weight of wire harnesses by substituting them for a combination of Impulse Radio Ultra-Wideband (UWB) and Power Line Communication (PLC) [2]–[4]. IR-UWB is suitable for in-vehicle applications due to its high tolerance to multipath propagation and penetration, and it has already shown high reliability in the in-vehicle environment [5]–[11]. The power lines of current harnesses need to remain to power these electronic devices. Thus, there is an opportunity to use these lines as a medium for PLC [1]. In this paper, we focus on the wireless part and leave the integration with PLC as future work. The power consumption of the devices and their impact on the performance of the motor vehicle remains outside the scope of this study.

In contrast to using wireless harnesses to reduce the weight of in-vehicle networks, adopting Ethernet represents a different approach. This trend aims to provide the high-speed connectivity and ample bandwidth necessary to support advanced driver assistance systems (ADAS). However, most in-vehicle electronic devices do not require such high-speed communications. In practice, a reference-model integrated board [12] engineered for vehicle computers and service-oriented gateways (SoG) supports Ethernet and controller area network (CAN). Therefore, we aim to substitute CAN by UWB.

An in-vehicle ECU collects sensing data from sensors and sends control data to actuators, usually cyclically, via the harness. Since vehicles are controlled in real-time, scheduling is essential to satisfy the strict latency and data loss constraints. Further, a lightweight scheduling algorithm is desired so that even a weak ECU can compute it quickly. Increasing the channel efficiency is also essential because, as explained in Sect. 2.2, the preamble overhead is not negligible. Thus, this paper proposes polling scheduling with data aggregation for the UWB wireless harness. In this context, "aggregation" means aggregating and conveying as much data as possible within a MAC frame.

In-vehicle systems can be regarded as networked control systems. Some wireless schedule algorithms are found in Refs. [14]–[17]. These papers propose scheduling algorithms for Time Division Multiple Access (TDMA) using linear programming and heuristic. These algorithms aim to transmit data as evenly as possible in each subframe to satisfy latency constraints under periodic data generation

conditions. They purposely leave unused space in every subframe so that event-triggered sensors can transmit with minimal latency, and lost data can be retransmitted before their deadline. These scheduling algorithms, however, limit the data transmission cycle for each sensor or actuator to a multiple of the base cycle. For this reason, these scheduling algorithms cannot be adapted to our scenario, which will be explained in Sect. 2. In particular, Ref. [14] primarily targets in-vehicle UWB networks, but it does not consider UWB preamble overhead.

In terms of aggregation, Ref. [18] proposes a frame aggregation method using the aggregated-MAC protocol data unit (A-MPDU) that works with the IEEE 802.15.4 MAC layer to implement a patient monitoring system. In this context, many sensing data generated cyclically in the system must be collected with strict latency constraints and quality-of-service requirements satisfied. This paper shows that analyzing the traffic pattern and aggregating and transmitting frames is more effective than sending frames frequently. Reference [19] shows that aggregation can reduce physical layer overhead and improve communication efficiency in wireless time-sensitive networking. However, it also points out that a trade-off between aggregation and packet error rate (PER) is needed, indicating that aggregation has a side effect when the bit error rate (BER) is large.

Regarding computationally lightweight real-time scheduling algorithms, Early Deadline First (EDF) [20] and Round-robin (RR) [21] are well-known. EDF prioritizes data with the earliest deadline and selects them for transmission. This contributes to satisfying latency constraints, but considering only the current state makes aggregation difficult. Further, its work-conserving feature can cause transmissions to be temporally concentrated, leaving insufficient room for retransmission. As its name suggests, RR is a fair scheduling scheme in which a parent terminal (PT) gives rights to send data to each child terminal (CT) equally in a round-robin fashion. RR, however, cannot take data aggregation into account either.

Considering the above, this paper proposes scheduling algorithms based on integer linear programming (ILP) and heuristic, minimizing the number of transmitted frames to increase channel efficiency by adjusting the phases of the sensor data. This paper is an extended version of [2]–[4]. In Refs. [2], [3], the scheduling algorithm is formulated as an ILP problem to reduce the number of polling slots. However, the freedom in the phase of periodic data generation was not fully exploited and presents opportunities to further reduce the number of response frames. Consequently, in [4], we endeavored to directly minimize the response frames by incorporating the data generation phase as a variable. While effective, this approach led to an increase in the number of variables, causing optimization algorithms computation time to increase. Moreover, periodically recalculating the schedule as sensors are added or removed is necessary. To mitigate this increase in computation time, we fix the polling target in the ILP, as will be elaborated in Sect. 2.3. Furthermore, given the limited computational resources of ECUs,

we developed a heuristic to further reduce the computation time.

The contributions of this paper are as follows:

1. The modified scheduling algorithm is described in detail.
2. A new heuristic scheduling algorithm that reduces the number of transmitted frames is proposed.
3. Experimental results demonstrating the effectiveness of our heuristic scheduling algorithm are shown.
4. Additional experiments with different data emission patterns for both the ILP-based and heuristic-based algorithms are also presented.

The rest of this paper is organized as follows. First, we describe an overview of the in-vehicle UWB network, UWB features, and polling control methods in Sect. 2. In Sect. 3, we formulate the schedule as an Integer Linear Programming (ILP) optimization problem. Then, we propose a heuristic scheduling algorithm to compute within reasonable time a sub-optimal schedule in Sect. 4. In Sect. 5, we present our experimental results and demonstrate the robustness of the schedules against interference. Finally, we conclude in Sect. 6.

## 2. System Model

This section presents an overview of in-vehicle networks, communication requirements, and control schemes.

### 2.1 UWB/PLC Integrated In-Vehicle Network

An overview of the UWB/PLC integrated in-vehicle network we envision is presented in Fig. 1. The network comprises one PT and five CTs. The PT serves as the electronic control unit and reads out data from sensors connected to CTs. When needed, the PT also sends control data to the actuators connected to CTs. In this paper, we only consider the uplink communication of the UWB part. More precisely, we consider that the PT periodically collects sensor data from the CTs via UWB.

Each sensor has a specific generation cycle, depending on its type. The length of sensor data is 6 octets, consisting of 2 octets for the sensor ID and 4 octets for sensor value. In this study, we consider two scenarios with different sensor data cycles: short cycles and long cycles. Tables 1 and 2 show the number of sensors connected to each CT and their
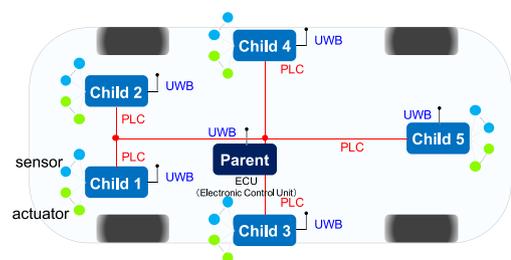


**Fig. 1** UWB/PLC integrated in-vehicle network.

**Table 1**     Number of sensors and generate cycles for the short cycles case.

| CT | Generation Cycle (ms) | | | |
|---|---|---|---|---|
|  | 24 | 56 | 72 | 108 |
| 1 | 16 | 12 | 10 | 2 |
| 2 | 14 | 14 | 10 | 8 |
| 3 | 16 | 14 | 6 | 4 |
| 4 | 14 | 18 | 6 | 6 |
| 5 | 14 | 16 | 10 | 4 |

**Table 2**     Number of sensors and generate cycles for the long cycles case.

| CT | Generation Cycle (ms) | | | |
|---|---|---|---|---|
|  | 24 | 56 | 72 | 108 |
| 1 | 6 | 12 | 18 | 30 |
| 2 | 8 | 16 | 16 | 38 |
| 3 | 6 | 16 | 21 | 26 |
| 4 | 6 | 14 | 18 | 30 |
| 5 | 5 | 16 | 20 | 27 |

**Table 3**     Communication requirements for the in-vehicle UWB/PLC integrated network.

| Data type | Communication system | Load | Allowable latency | Allowable data loss rate | Allowable jitter |
|---|---|---|---|---|---|
| Non-low latency | UWB | 250 kbit/s | $\leq$ 25 ms | $\leq 10^{-3}$ | $\leq$ 10 ms |
|  | PLC | 250 kbit/s | $\leq$ 25 ms | $\leq 10^{-3}$ | $\leq$ 10 ms |
| Low latency | PLC | 250 kbit/s | $\leq$ 15 ms | $\leq 10^{-4}$ | $\leq$ 5 ms |

generation cycles for the short and the long cycles cases respectively. The data transfer load is about 250 kbit/s in both cases.

Table 3 summarizes our performance target. The target data loss rate for UWB networks is $10^{-3}$. The time between when the data is created by the sensor and when it is received by the PT is considered latency. Regarding latency, communications are categorized into *low latency* for safety-critical data related to driving operation and *non-low latency*. Non-low latency corresponds to Class A or B of the Society of Automotive Engineers (SAE) classification. These classes comprise body-equipment data such as wiper operation, communication about windows, mirrors, air conditioning and meter displays [22]. Therefore, the maximum latency requirement is 25 ms or less and the jitter requirement is 10 ms or less. On the other hand, the target schedule computation time is 0.5 s. This corresponds to the approximate time it takes for the in-vehicle ECU to start up.

## 2.2   Preamble Overhead and Aggregation

Figure 2 shows the format of the polling and response frames. The polling frame from the PT to CTs includes the IDs of the sensors to be read and the data for the actuators. The response frame from CTs includes the readout data of the sensors specified in the polling frame. Standard IEEE 802.15.4 UWB frames can carry up to 127-byte Physical Service Data Unit (PDSU). As shown in Fig. 2, the MAC payload is 116 octets or less, and the sensor data in the response frame consists of 2 octets of ID and 4 octets of data; one frame can store the data from 19 sensors.

Figure 3 shows the ratio of PSDU transmission time to PLCP Protocol Data Unit (PPDU) frame transmission
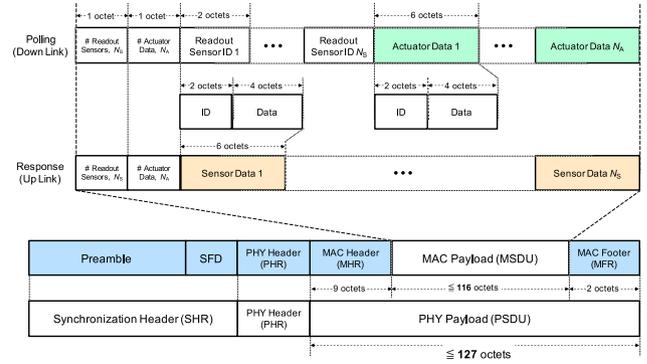


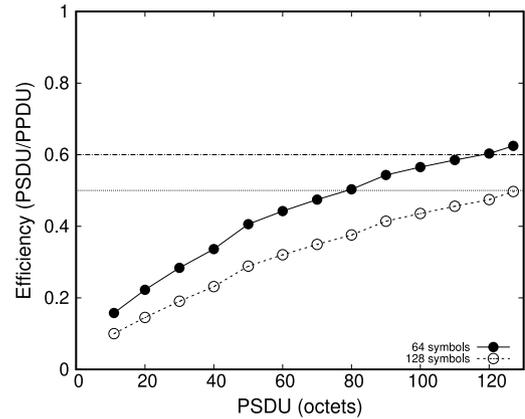**Fig. 2**     MAC SDU format/PPDU format.



**Fig. 3**     Ratio of PSDU transmission time to PPDU frame transmission time.

time for each PSDU size, estimated according to [23]. The communication efficiency in the case of 64 symbols is better than that of 128 symbols, but it remains at most 60%. In the case of 128 symbols, efficiency is further reduced to 50%, to the point that the preamble and payload transmission times are almost the same. Therefore, it is more efficient to use a 64-symbol preamble.

The UWB module we used, Qorvo DW3000, can extend the PSDU length up to 1023 octets using the extended format. This, however, makes the PHY Header (PHR) non-compliant with IEEE 802.15.4 because this format is an option in the IEEE 802.15.8 standard [24]. Therefore, when this long frame mode is chosen, the DW3000 cannot communicate with any device operating under the IEEE 802.15.4 standard frame encoding. Considering the implementation difficulty and the trade-off between frame length and PER, we choose to keep the IEEE 802.15.4 standard and to aggregate as much data as possible into the PSDU of up to 127 octets.

## 2.3   Polling Control and Retransmission

In IEEE 802.15.4, Clear Channel Assessment (CCA) is enabled by periodically inserting a preamble symbol into the data symbol [25]. This is, however, optional and is not implemented in UWB modules from major manufacturers such as NXP Semiconductors or Qorvo. The UWB module Qorvo

DW3000 we used is equipped with a simple pseudo-CCA, but it can detect only the preamble and not the PHR or PSDU. Therefore, we employ a polling control where the PT allows CTs to communicate one at a time. The DW3000 is compliant with 802.15.4 and already implements Reed-Solomon coding as Forward Error Correction (FEC) [24], [25].

When and which sensor data is read is determined through offline scheduling. The schedule is generated slot-by-slot, with a single CT communicating per slot. The generation cycle of each sensor is an integer multiple of the slot time. The slot length is determined by taking into account both the common divisor of the data generation cycle and the time required to adequately receive a response frame following the transmission of a polling frame.

In the short and long cycles, we introduced in Tables 1 and 2, the common divisors of the generation cycles are 1, 2, and 4 ms. Preliminary experiments showed that the time from sending a polling frame to receiving a single response frame is about 0.86 ms; until two response frames are received, it is about 1.35 ms; and until three response frames are received, it is about 2.1 ms. Considering these factors, we opted for a slot length of 4 ms in both of our scenarios. Furthermore, the number of sensor data specified to be read out in one polling frame should be approximately 38 data that can be transmitted in two response frames, leaving room for retransmission.

Polling is conducted sequentially from CT1 to CT5. Furthermore, as indicated in Table 3, the allowable latency is set at 25 ms, within which each CT should be polled at least once. Consequently, in our case, each CT must be polled at least every sixth slot, equating to 24 ms. Frequent polling, however, can lead to a waste of channel capacity. Therefore, we keep the unallocated slot after CT5 to reduce interference with other UWB systems and provide room for retransmission in case of communication failures [3]. This results in a total of 6 slots of 4 ms each that repeat every 24 ms as depicted in Fig. 4.

The polling control flowchart is shown in Fig. 5. The PT keeps the readouts to perform in a priority queue in which they are sorted according to their deadline. At the beginning of a slot, the planned readouts for that slot are added to the queue, prompting the PT to start polling the data from the corresponding CT in the earliest-deadline-first order. If the PT fails to receive a response for some sensor data, the readout request specifying the CT and sensor IDs is re-added to the priority queue, causing the other planned readouts to be performed later in the slot. As a result, failed transmissions are attempted again as soon as possible until they succeed, minimizing the latency even in case of failure. To ensure that latency constraints are met, data that would not satisfy latency requirements are discarded from the transmission queue. The empty slot serves as a buffer in case retransmissions cause the planned schedule to shift beyond the originally planned slots. Forward error correction is not further considered, given that the PSDU is encoded using Reed-Solomon code.
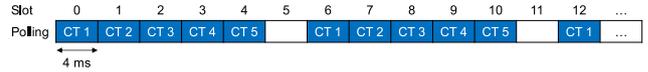


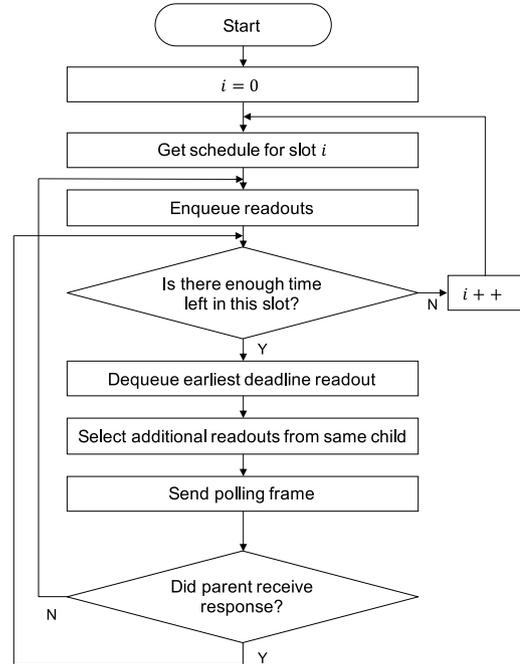**Fig. 4** Pre-allocated slot to each chile device and unallocated slot.



**Fig. 5** Polling control flowchart.

## 3. Optimization of Polling Schedule

This section explains the basic ideas for aggregating readout data into a frame and how to generate an optimal schedule using Integer Linear Programming (ILP).

### 3.1 Adjustment of Data Generation Phase

We explain the idea of phase control to reduce the number of frames transmitted. In this paper, each sensor is read out every corresponding cycle, and the polling order for each CT is also predetermined, as shown in Fig. 4. On the other hand, the generation cycle phase is still variable. Therefore, we adjust it to convey the readout data over fewer frames.

Let us explain the phase control using a simple example. Figures 6 and 7 depict scenarios without and with phase control, respectively. In this instance, we consider three types of periodic data that need to be read out before their respective deadlines. Sensor 1's data is expected to be read out every sixth slot, while Sensors 2 and 3 are read out every 12th slot. Polling is performed every 6th slot. For simplicity, we assume that two pieces of datum can be transmitted in a response frame. In the absence of phase adjustment and given these constraints, six frames are required, with the headers and trailers of frames transmitting only one datum, as shown in Fig. 6. In contrast, in Fig. 7, the generation cycle phase of Sensor 3 is shifted to avoid having to transmit the
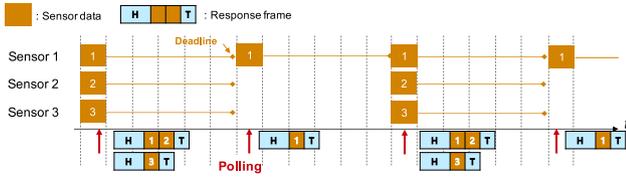
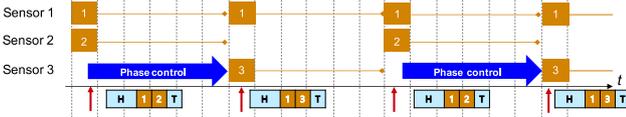**Fig. 6** Polling without control of cyclic generation phase.



**Fig. 7** Polling with control of cyclic generation phase.

data from all 3 sensors in the same slot, reducing the required number of frames to four.

## 3.2 Scheduling Formulation

Based on this idea of phase control, we optimize when and which sensor data is read out using the following ILP formulation.

### 3.2.1 Variable Definitions

The variables used in our formulation are the following:

- $T$: schedule cycle (least common multiple of polling cycle and generation cycles),
- $\delta$: slot duration in milliseconds,
- $P$: polling cycle for each CT in milliseconds,
- $L$: allowable readout latency in milliseconds,
- $M$: the maximum number of data that can be read within a slot, where $M$ is a multiple of $N$,
- $N$: the maximum number of data that can be stored in a single frame,
- $k$: index of the allocated slots for an ST,
- $\mathcal{S}$: set of sensors,
- $\mathcal{J}_s$: set of readout data for sensor $s$,
- $\mathcal{J}$: set of readout data of all sensors,
- $c_s$: data generation cycle of sensor $s$,
- $g_{s,j}$: generation time of data $j$ from sensor $s$ before time shifting,
- $r_{s,j}$: allocated readout time in milliseconds for data $j$ of sensor $s$,
- $\phi_s$: integer variable, indicating the number of phase shift slots for the sensor $s$,
- $l_{s,j}$: latency from the generation time of data $j$ of sensor $s$ to the polling occasion for allocated readout time,
- $d_k$: the number of data transmitted in the $k$th allocated slot,
- $f_k$: the number of frames transmitted in the $k$th slot,

and

- $x_{s,j,k}$: binary variable, 1 if data $j$ of sensor $s$ is readout in the $k$th allocated slot, 0 otherwise.

It is assumed that time starts from 0 ms and slots are numbered in ascending order from zero. Note that the selection and timing of sensor data for readout repeat in cycles, with each cycle $T$ being the least common multiple of the sensor-generated cycles, as follows:

$$T = \mathrm{LCM}(\{P\} \cup \{c_s | s \in \mathcal{S}\}). \tag{1}$$

### 3.2.2 Objective Function and Constraints

We formulate a multi-objective function to reduce the total number of frames and avoid unnecessary polling latency. Recall that each CT is polled every $P/\delta$ slots, i.e., every six slot in the system model explained in Sect. 2. Therefore, the $k$th allocatable slot for CT $n$ is the $\{(P/\delta)k + (n-1)\}$st slot. Then, each CT is allocated $T/P$ polling slots during $T$ ms. For the latency of data occurring towards the end of the schedule cycle, $\{\lfloor (L-\delta)/P \rfloor + 1\}$ slots are added. The additional polling corresponds to the $\{\lfloor (L-\delta)/P \rfloor + 1\}$st allocatable slot from 0 in the early part of the cycle. Based on these, the range of $k$ is from 0 to $\{T/P + \lfloor (L-\delta)/P \rfloor + 1\}$. Taking this into consideration, we employ the following formulation, which is then solved for each CT to generate a schedule:

Minimize

$$\alpha \sum_{k=0}^{T/P-1} f_k + (1-\alpha) \sum_{s \in \mathcal{S}} \sum_{j \in \mathcal{J}_s} l_{s,j}, \tag{2}$$

subject to

$$l_{s,j} \leq L - \delta, \quad \forall j \in \mathcal{J}_s, \forall s \in \mathcal{S}, \tag{3}$$

$$l_{s,j} \geq 0, \quad \forall j \in \mathcal{J}_s, \forall s \in \mathcal{S}, \tag{4}$$

$$l_{s,j} = r_{s,j} - (g_{s,j} + \phi_s \delta), \quad \forall j \in \mathcal{J}_s, \forall s \in \mathcal{S}, \tag{5}$$

$$r_{s,j} = \sum_{k=0}^{T/P + \lfloor (L-\delta)/P \rfloor} k P x_{s,j,k},$$
$$\forall j \in \mathcal{J}_s, \forall s \in \mathcal{S}, \tag{6}$$

$$\phi_s \leq c_s/\delta - 1, \quad \forall j \in \mathcal{J}_s, \forall s \in \mathcal{S}, \tag{7}$$

$$\phi_s \geq 0, \quad \forall j \in \mathcal{J}_s, \forall s \in \mathcal{S}, \tag{8}$$

$$\sum_{k=0}^{T/P + \lfloor (L-\delta)/P \rfloor} x_{s,j,k} = 1, \quad \forall j \in \mathcal{J}_s, \forall s \in \mathcal{S}, \tag{9}$$

$$d_k \leq M, \quad 0 \leq k \leq T/P - 1, \tag{10}$$

$$d_k = \begin{cases} \sum_{s \in \mathcal{S}} \sum_{j \in \mathcal{J}_s} (x_{s,j,k} + x_{s,j,k+T/P}), \\ \qquad\qquad 0 \leq k \leq \lfloor (L-\delta)/P \rfloor, \\ \sum_{s \in \mathcal{S}} \sum_{j \in \mathcal{J}_s} x_{s,j,k}, \\ \qquad\qquad \lfloor (L-\delta)/P \rfloor < k \leq T/P - 1, \end{cases} \tag{11}$$

$$f_k = \min\{1, d_k\} - \sum_{p=1}^{M/N} \min\{pN, d_k\}$$
$$+ \sum_{p=1}^{M/N} \min\{pN+1, d_k\}, \quad 0 \leq k \leq T/P - 1. \tag{12}$$

The weight parameter $\alpha$ is set to a significantly large value to primarily minimize the number of response frames while concurrently reducing readout latency. To achieve this, we choose the value of $\alpha$ to satisfy the following condition:

$$\frac{NL}{1+NL} \ll \alpha. \tag{13}$$

The derivation of the above is explained in Appendix.

The inequalities of Eqs. (3) and (4) are constraints on the time of data transmission allocation. The inequality of Eq. (3) guarantees the transmission of data within the allowable latency. Since the response frame will follow its polling frame in a slot, the latter should be sent $\delta$ before the allowable latency. The inequality of Eq. (4) forces the data transmission to occur after its creation, i.e., the latency should be non-negative. Equation (5) defines the latency, where $g_{s,j} + \phi_s\delta$ represents the generation time after phase shifting. Equation (6) converts the data readout time in the $k$th allocated slot to the time in slots. Equations (7) and (8) limit the range of phase shift to between 0 to the data generation cycle. Equation (9) ensures that all data are allocated for transmission only once. The inequality of Eq. (10) restricts the number of data transmitted in a slot to $M$ or less. Equation (11) represents the number of data transmitted in $k$th allocated slot. Data allocated for transmission after $T$ ms is concurrently transmitted within time slots 0 to $\lfloor L/P \rfloor - 1$. Equation (12) converts the number of readout data to the number of response frames to convey them.

Note that Eq. (12) is not a linear form. Nevertheless, it can be rewritten with Eq. (11) as a linear form of the column vector $(d_k, f_k)^\top$ by introducing the SOS2 (Special Ordered Set of Type 2) [26] auxiliary variables $t_1, \ldots, t_{2(M/N+1)}$ and binary variables $z_1, \ldots, z_{2(M/N)+1}$ as shown below:

$$\begin{pmatrix} d_k \\ f_k \end{pmatrix} = \sum_{p=0}^{M/N} t_{2p+1} \begin{pmatrix} pN \\ p \end{pmatrix}$$
$$+ \sum_{p=0}^{M/N} t_{2(p+1)} \begin{pmatrix} pN+1 \\ p+1 \end{pmatrix}, \tag{14}$$

$$t_i \geq 0, \quad i = 0, 1, \ldots, 2(M/N+1), \tag{15}$$

$$\sum_{i=1}^{2(M/N+1)} t_i = 1, \tag{16}$$

$$\sum_{i=1}^{2(M/N)+1} z_i = 1, \tag{17}$$

$$t_1 \leq z_1, \tag{18}$$

$$t_i \leq z_{i-1} + z_i, \quad i = 2, 3, \ldots, 2(M/N) + 1, \tag{19}$$

$$t_{2(M/N+1)} \leq z_{2(M/N)+1}. \tag{20}$$

## 4. Heuristic Algorithm

The computational complexity of ILP grows exponentially with the number of inputs, i.e., sensors, posing challenges in finding a solution within a reasonable time. In our case, we want the low computing power of a motor vehicle's ECU to be capable of recomputing the schedule as the sensors polled and their cycles may evolve depending on the operation of

---

**Algorithm 1** Heuristic algorithm

**Input:** Set of sensors $\mathcal{S}$
**Output:** Sensor data generation time and transmission schedule
1: Schedule cycle: $T$
2: Time slot length: $\delta$
3: Data generation cycle of sensor $s$: $c_s$
4: Polling cycle: $c_P$
5: Sort the set of sensors $\mathcal{S}$ in order of shortest generation cycle $c_s$
6: **for** $s \in \mathcal{S}$ **do**
7:     **for** $\phi = 0$ to $c_s/\delta - 1$ **do**
8:         **for** $j = 0$ to $T/c_s - 1$ **do**
9:             Calculate $j$th data generation time of sensor $s$: $g_{s,j,\phi}$
                $g_{s,j,\phi} = j\, c_c + \phi\, \delta$
10:             Calculate $j$th data readout time of sensor $s$: $r_{s,j,\phi}$
11:             **if** $g_{s,j,\phi} < T - c_P$ **then**
12:                 $r_{s,j,\phi} = \lceil (g_{s,j,\phi} - \delta)/c_P \rceil c_P$
13:             **else**
14:                 $r_{s,j,\phi} = \lceil (g_{s,j,\phi} - \delta - T)/c_P \rceil c_P$
15:             **end if**
16:         **end for**
17:         Calculate the total number of frames, the maximum number of frames per slot, the maximum room in payload among the response frames, and total latency
18:     **end for**
19:     Select $\phi$ and determine the phase of $s$ (priority is given to above criteria)
20:     Add readout time $r_{s,j,\phi}$ into schedule
21: **end for**

---

the vehicle. Consequently, we introduce a heuristic algorithm designed to swiftly yield an acceptable solution to the schedule optimization problem.

Our heuristic scheduling algorithm is shown in Algorithm 1. The algorithm performs the phase optimization for each CT independently, taking the set of sensors and their corresponding generation cycles as input. The algorithm gradually builds the schedule by adding the sensors to be read out one by one, starting with those exhibiting the shortest generation cycle.

The phase adjustment consists of selecting the optimal slot for each sensor based on the following evaluation criteria from most (top) to least important (bottom):

1. Minimizing the total number of response frames,
2. Minimizing the maximum number of response frames per slot,
3. Minimizing the maximum room in payload among the response frames,
4. Minimizing the total latency.

If a tie among multiple phase candidates occurs, one is selected at random.

In the above, Criterion 1 aims to suppress the total number of response frames, corresponding to the first term in Eq. (2). Criterion 2 intends to leave residual time in a slot for retransmissions. Criterion 3 aims to evenly distribute the room of payload, which we call "payload margin," among response frames, allowing flexibility for sensors whose phases have not yet been adjusted. Finally, Criterion 4 focuses on minimizing latency, corresponding to the second term in Eq. (2).

### 4.1 Applying the Heuristic Algorithm to Simple Examples

We illustrate the heuristic algorithm using a simple example presented in Fig. 8. In this example, we consider a CT equipped with five sensors: Sensor 1 through 5, each having generation cycles of $c_1 = 12$ ms, $c_2 = c_3 = c_4 = 16$ ms, and $c_5 = 24$ ms, respectively. The CT is polled every three slots, which amounts to every 12 ms. Unlike the example from Sect. 3.1, three pieces of data can be conveyed in a response frame.

The algorithm starts by determining the phase for Sensor 1, which has the shortest cycle. Since the generation cycle of Sensor 1 is equal to the polling cycle, shifting the generation phase only increases the latency, so it is scheduled as shown in Fig. 8(a). Next, Sensor 2 is equivalent in the above criteria, even if its phase is shifted over 16 ms, i.e., four slots. In such a case, its phase is randomly chosen; Figure 8(b) shows that a shift of 0 was picked. In the scheduling of Sensor 3, there is a difference in terms of Criterion 3 over its cycle, i.e., four slots. As shown in Fig. 8(c), the maximum payload margin is two, while it is one when the offset of the phase is one. Therefore, the latter is chosen. Sensor 4 is scheduled as shown in Fig. 8(d) for the same reason as Sensor 2. The last sensor has four potential phase offset candidates, as illustrated in Fig. 8(e). In the scenario without any offset from the 0th slot, the total number of response frames is seven, whereas it is six for the other offset options. The former case is eliminated based on Criterion 1. Ultimately, the choice is determined by Criterion 4 in favor of the offset by three slots, as it incurs no latency compared to the one- and two-slot offsets.

For reference, the schedule obtained without the heuristic algorithm is depicted in Fig. 9. With the heuristic algorithm, one less frame is necessary to transmit the data, reducing the total number of frames from seven down to six. Overall, the heuristic algorithm is effective in reducing overhead.

### 4.2 Comparison of ILP and Heuristic Scheduling Results

We compare the schedules generated by the heuristic algorithm and those by the ILP method discussed in Sect. 3. With the condition of the weight $\alpha$ of the objective function (2), the left-hand side of Eq. (13) is 0.997. Therefore, we set $\alpha = 0.999$. The number, $M$, of sensor data specified to be read out in one polling frame is 38, corresponding to two frames. The evaluation metrics include the total number of response frames and the average latency. Note that Latency is defined as the time from when the data is generated until the readout is allocated and polling begins. All calculations were performed on a MacBook Air equipped with an Apple M1 chip and 16 GB of memory, with user CPU time measured using the 'time' command. The ILP was implemented in Python, using the Gurobi Optimizer as the optimization solver. In contrast, the heuristic algorithm, which does not rely on an optimization solver, was implemented in C to
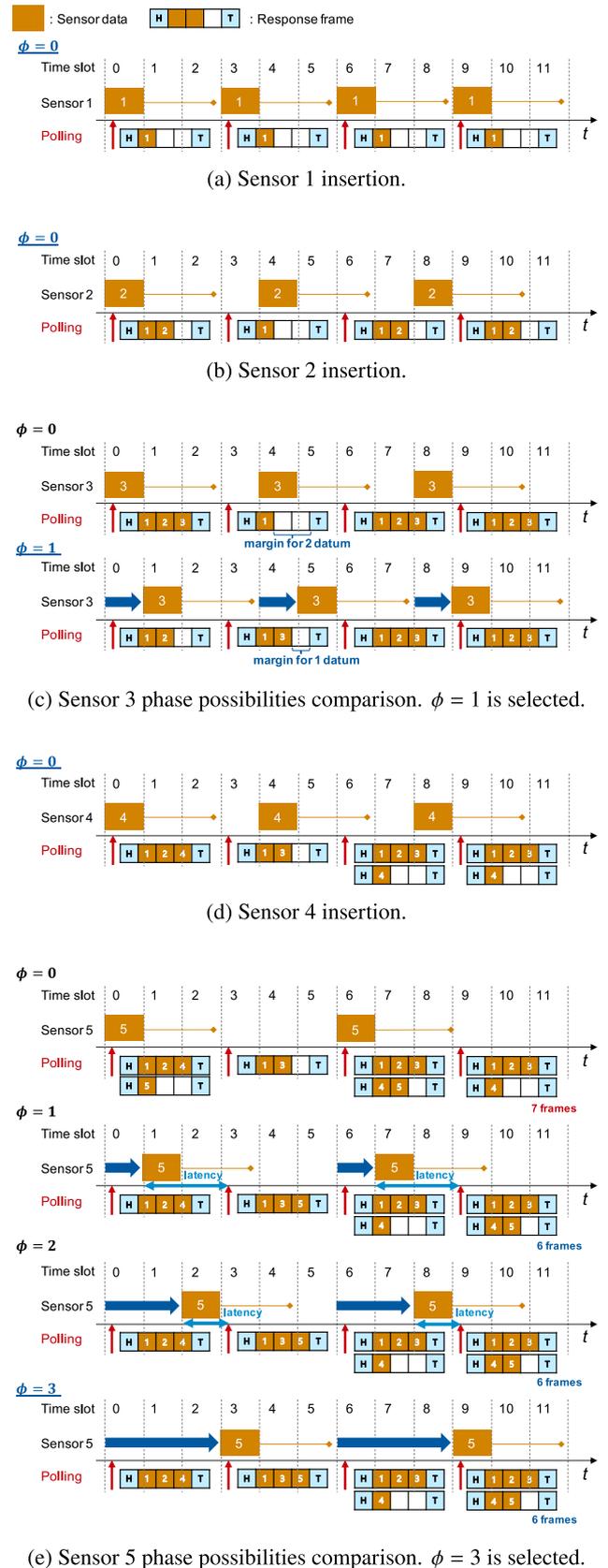


(a) Sensor 1 insertion.

(b) Sensor 2 insertion.

(c) Sensor 3 phase possibilities comparison. $\phi = 1$ is selected.

(d) Sensor 4 insertion.

(e) Sensor 5 phase possibilities comparison. $\phi = 3$ is selected.

**Fig. 8**   Heuristic algorithm execution example.
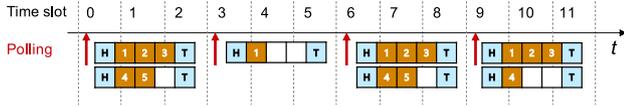
**Fig. 9**   Schedule without phase adjustment by the heuristic algorithm.

**Table 4**   Schedule characteristics for the short cycles case.

| Scheduling | Phase adjustment | Number of response frames | Average latency (ms) | Latency std deviation (ms) |
|---|---|---|---|---|
| RR w/ empty | ILP | 466 | 2.28 | 4.99 |
| RR w/ empty | Heuristic | 538 | 2.42 | 5.20 |
| RR w/ empty | w/o | 543 | 2.29 | 4.99 |
| RR w/o empty | w/o | 554 | 7.87 | 5.73 |
| PO w/o empty | w/o | 546 | 2.93 | 5.14 |

**Table 5**   Schedule characteristics for the long cycles case.

| Scheduling | Phase adjustment | Number of response frames | Average latency (ms) | Latency std deviation (ms) |
|---|---|---|---|---|
| RR w/ empty | ILP | 439 | 3.53 | 5.71 |
| RR w/ empty | Heuristic | 525 | 3.53 | 5.71 |
| RR w/ empty | w/o | 598 | 3.53 | 5.71 |
| RR w/o empty | w/o | 605 | 7.80 | 5.77 |
| PO w/o empty | w/o | 606 | 7.04 | 7.50 |

maximize processing speed.

For comparison, we include three polling methods without phase adjustment: "RR without empty slots," which is the simplest polling without empty slots, and "RR with empty slots," which is described in Sect. 2.3. Note that both ILP and heuristic also keep an empty slot. "Polling-order optimized (PO)" approach optimizes the sequence of polling without incorporating empty slots, utilizing the ILP model described in [2], [3]. This method is designed to minimize polling frequency as much as possible. The two scenarios were used using short and long generation cycles previously introduced in Tables 1 and 2.

The results of the calculations are summarized in Tables 4 and 5. These tables show that the ILP and the heuristic algorithm successfully reduce the number of response frames, indicating effective aggregation of readout data. It is, however, important to note that this reduction in response frames comes at a cost. The average latency for the ILP and the heuristic algorithm is higher than that for no phase adjustment in the RR with empty slots. This is primarily due to readouts being postponed (within latency constraints) for aggregation. The reason for the extremely high latency of the round-robin without empty slots is that the polling cycle is 20 ms, while most sensors have a generation cycle of 24 ms. This forces polling timing to shift gradually and cyclically from the expected generation timing, resulting in a large latency. In contrast, this does not happen thanks to the matching polling cycle of 24 ms in the round-robin schedules with empty slots. In scenarios involving a long cycle, there is a significant disparity in the schedules when comparing cases with and without phase adjustment. Although the data rates for both patterns remain identical, these differences arise due to the increased flexibility in sensor generation timing afforded by the long cycle. Consequently, in "Polling-order optimized," which lacks phase adjustment, there is an ob-

**Table 6**   Schedule computation time.

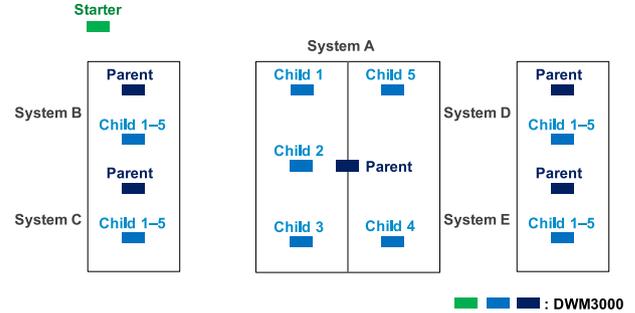| Method | Short cycles | Long cycles |
|---|---|---|
| ILP | 987 s | 1676 s |
| Heuristic | 70 ms | 190 ms |



**Fig. 10**   Physical layout of the UWB modules inside of the EMC tent.

served increase in both the number of frames and latency.

Next, as depicted in Fig. 6, the heuristic algorithm achieves a remarkable reduction in computation time, approximately 99.9% less than the ILP, for both scenarios. The heuristic is below the target value of 0.5 s in both scenarios, while the ILP requires about 15 minutes or more. This suggests that the heuristic algorithm may enable on-board ECUs to implement phase adjustment in response to changing situations.

## 5.   Experimental Validation

To verify the effectiveness of our various methods, we conducted an experiment in which we ran the obtained schedules and measured the data loss ratio and the latency. The experiment was conducted in an Electromagnetic Compatibility (EMC) tent manufactured by Medical Aid Co. LTD, and Qorvo DWM3000 UWB modules were used as the UWB equipment. In this experiment, a set of UWB modules is used to simulate the wireless harness (System A), and four sets of interfering UWB modules (Systems B through E) are placed, as shown in Fig. 10. System A uses six modules to emulate the in-vehicle network, while for Systems B through E, one module serves as the PT, and one module serves the role of the five CTs. The configuration used was channel 9 (center frequency 7987.2 MHz, bandwidth 499.2 MHz), data rate of 6.81 Mbit/s, and a preamble length of 64 symbols (as opposed to the default 128 symbols). This change in the preamble length was made to increase the time available for retransmission in case of reception failure, as explained in Sect. 2.

### 5.1   Experimental Methods

Figure 11 shows our experimental method. Five types of transmission schedules for each system were employed, as shown in Tables 4 and 5. The duration of a single schedule is 1512 ms, corresponding to the schedule cycle $T$, and it repeats three times in an experimental trial. To start the
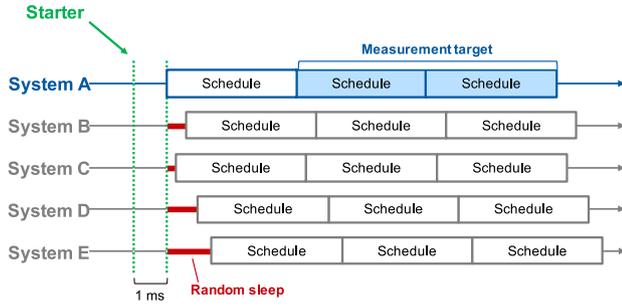
**Fig. 11** Synchronization method between Systems and measurement target.

experiment, the starter UWB module sends a synchronization signal to the PT of Systems A through E. As soon as the PT in System A receives the synchronization signal, it polls the CT according to its schedule. On the other hand, Systems B through E sleep for a random period of time upon receiving the start signal and then communicate with their CT according to the same schedule. The random sleep time is selected in the range 0 through 1511.999 ms using the Mersenne-Twister method. We measured the communication performance of System A's second and third schedules, focusing on aspects such as data loss rate and latency while it is receiving interference from the other systems. To increase the reliability of the experiment, we repeated the experimental trial 100 times. As a result, we obtained data loss rate, average latency, and latency standard deviation.
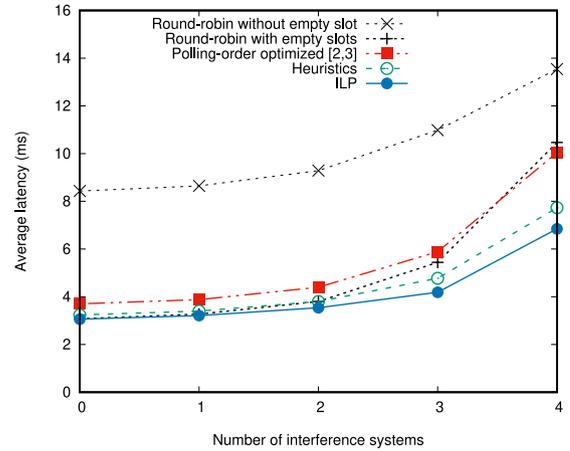
## 5.2 Experimental Results

The data loss rate, average latency, and latency standard deviation for the short and long data cycles are shown in Figs. 12 and 13, respectively. Note that Figs. 12(a) and 13(a) use a logarithmic scale; cases with zero loss rate are therefore not plotted.

As a general trend, the data loss rate is lower when the number of frames is smaller, indicating that aggregation is effective. Since "Round-robin without empty slots" has a larger data loss rate even when there is only one interfering system, we conclude that it is worthwhile to prepare empty slots for retransmission in case of data loss. The schedules obtained through the "ILP" and the "heuristic" algorithm achieve the target data loss rate, $\leq 10^{-3}$, in the presence of up to two interfering systems in both patterns. It can be inferred that the degradation of radio link quality was not a factor in our experiments, as they were conducted within the EMC tent, thus avoiding external interference. In scenarios devoid of interfering systems, data loss was not observed. However, in the presence of one or more interfering systems, data loss occurred due to collisions between the communications of different UWB systems. "PO" demonstrated lower values than "RR without empty slot," yet it showed values nearly identical to those of "RR with empty slots." Moreover, it was confirmed that both the "ILP" and "heuristic" methods yield better outcomes than the existing method for both patterns.
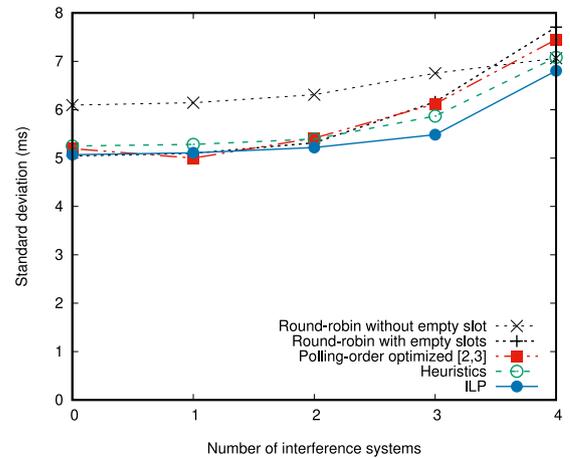
For average latency and its standard deviation, both the
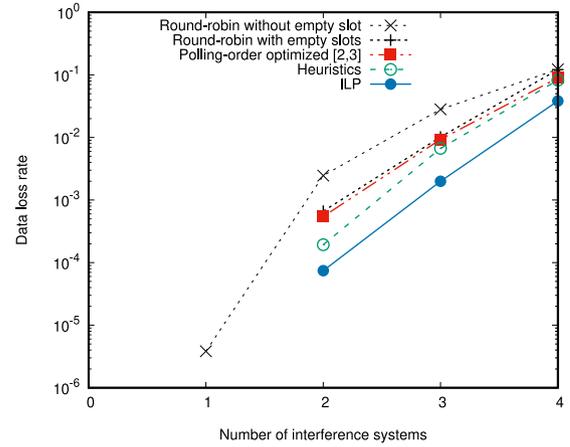


(a) Data loss rate depending on the number of interference systems. Loss rates of 0 are not plotted.
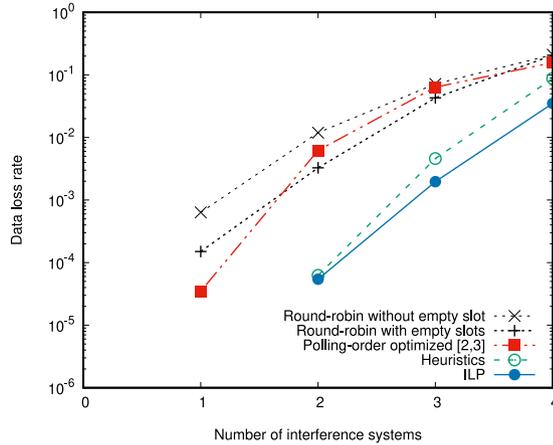


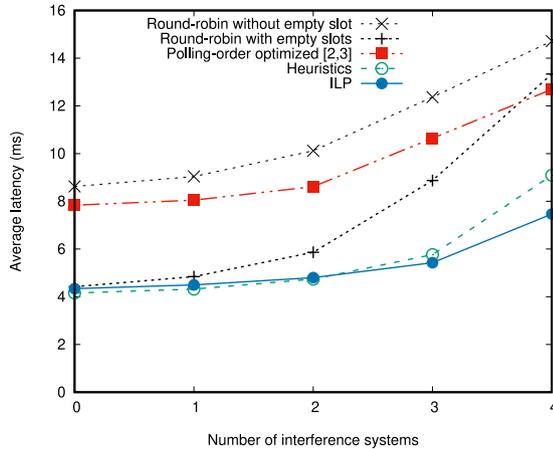(b) Average latency depending on the number of interference systems.



(c) Standard deviation of latency depending on the number of interference systems.

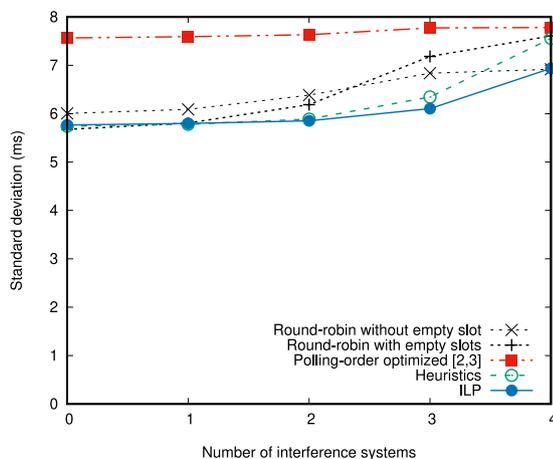**Fig. 12** Results of the short cycle experiments.

ILP and heuristic algorithms exhibit values that are comparable to those without interfering systems. However, in the case of the round-robin without empty slots, data loss

(a) Data loss rate depending on the number of interference systems. Loss rates of 0 are not plotted.



(b) Average latency depending on the number of interference systems.



(c) Standard deviation of latency depending on the number of interference systems.

**Fig. 13**     Results of the long cycle experiments.

increases as interference grows, leading to more frequent retransmissions and higher latency. Conversely, the ILP and heuristic algorithms show a smaller increase in latency

compared to the other methods. Moreover, schedules in the existing methods tended to tolerate a higher latency if it could reduce the number of polling, resulting in more significant latency compared to ILP and heuristics. In particular, the long cycle exhibited a significant concentration of data generation at certain points, resulting in increased latency. Data pollings that would fail to meet their latency requirements are systematically discarded from the polling queue and are therefore treated as transmission failures. Therefore, the observed maximum latency remains within the specified requirements. The standard deviations of the latency representing the jitter were all below the target value of 10 ms.

As a result, the ILP produces the most favorable schedule regarding data loss rate and latency. While not as good as the schedule obtained through the ILP, the heuristic algorithm still provides superior scheduling compared to standard polling methods without any adjustments.

We estimated the average power consumption of the UWB modules in the experiment using the DW3000 Datasheet [27]. Power consumption is mostly influenced by the number of frames transmitted. With the "PO" scheduler, where the most frames were transmitted, the PT consumes 41.07 mW and each CT consume 10.61 mW. These are of the same order of magnitude as the average power consumption of the CAN-FD (CAN Flexible Data Rate) transceiver "TCAN1042" which consumes 58.75 mW [28]. The power consumption of the UWB devices is not a concern.

## 6. Conclusion

This paper introduced a polling control method and scheduling algorithm designed for in-vehicle UWB networks. We developed an ILP formulation to generate schedules with minimum response frames. Also, we proposed a heuristic algorithm capable of producing schedules within a few hundred milliseconds. By comparing the quality of schedules obtained from both methods, we demonstrated that our proposed heuristic approach yields nearly optimal solutions. Additionally, we conducted experimental validation using a UWB module to assess the robustness of our schedules against interference. The ILP optimization and the heuristic schedules showed low data loss rates and minimized readout latency.

In this paper, we have primarily addressed scheduling for sensor readouts. Future directions include developing scheduling schemes incorporating data transmission to actuators and improving system reliability, especially in scenarios with three or more interference systems.

## References

[1] S.C. Ergen and A. Sangiovanni-Vincentelli, "Intravehicular energy-harvesting wireless networks: Reducing costs and emissions," IEEE Veh. Technol. Mag., vol.12, no.4, pp.77–85, Dec. 2017.

[2] C. Ohta, T. Tanaka, H. Migita, S. Yamaguchi, M. Takenaka, P. Finnerty, and T. Kamada, "A study on optimization of polling scheduling for in-vehicle UWB wireless networks," IEICE Commun. Express, vol.11, no.7, pp.429–434, July 2022.

[3] H. Migita, T. Tanaka, S. Yamaguchi, M. Takenaka, P. Finnerty, T. Kamada, and C. Ohta "Optimization of polling-based MAC schedule considering data aggregation for in-vehicle UWB wireless networks," Proc. 2022 IEEE 8th World Forum on Internet of Things (WF-IoT), 6 pages, Oct.–Nov. 2022.

[4] H. Migita, M. Takenaka, P. Finnerty, M. Okuhara, and C. Ohta, "A study on optimization of polling schedule to minimize the number of frames for in-vehicle UWB wireless network," IEICE Commun. Express, vol.12, no.6, pp.333–338, June 2023.

[5] M. Koca, G. Gurbilek, B. Soner, and S. Coleri, "Empirical feasibility analysis for energy harvesting intravehicular wireless sensor networks," IEEE Internet Things J., vol.8, no.1, pp.179–186, Jan. 2021.

[6] U. Demir, C.U. Bas, and S. Coleri Ergen, "Engine compartment UWB channel model for intravehicular wireless sensor networks," IEEE Trans. Veh, Technol., vol.63, no.6, pp.2497–2505, July 2014.

[7] C.U. Bas and S.C. Ergen, "Ultra-wideband channel model for intravehicular wireless sensor networks beneath the chassis: from statistical model to simulations," IEEE Trans. Veh. Technol., vol.62, no.1, pp.14–25, Jan. 2013.

[8] A. Chandra, A. Prokeš, T. Mikulášek, J. Blumenstein, P. Kukolev, T. Zemen, and C.F. Mecklenbräuker, "Frequency-domain in-Vehicle UWB channel modeling," IEEE Trans. Veh. Technol., vol.65, no.6, pp.3929–3940, June 2016.

[9] J. Li and T. Talty, "Channel characterization for ultra-wideband intra-vehicle sensor networks," Proc. IEEE Mil. Commun. Conf. (MILCOM), pp.1–5, 2006.

[10] T. Tsuboi, J. Yamada, N. Yamauchi, M. Nakagawa, and T. Maruyama, "UWB radio propagation inside vehicle environments," Proc. 2007 7th International Conference on ITS Telecommunications, pp.1–5, 2007.

[11] P.C. Richardson, W. Xiang, and W. Stark, "Modeling of ultra-wideband channels within vehicles," IEEE J. Sel. Areas Commun., vol.24, no.4, pp.906–912, April 2006.

[12] NXP Semiconductors, "S32G3 Vehicle Networking Reference Design," accessed on 5 Dec., 2023. https://www.nxp.com/design/designs/s32g3-vehicle-networking-reference-design:S32G-VNP-RDB3

[13] L.A. Perişoară, D.I. Săcăleanu, and C. Dănişor, "Automotive ethernet architecture for the connected dacia logan electric vehicle," Proc. 2022 14th International Conference on Communications (COMM), pp.1–6, 2022.

[14] Y. Sadi and S.C. Ergen, "Optimal power control, rate adaptation, and scheduling for UWB-based intravehicular wireless sensor networks," IEEE Trans. Veh. Technol., vol.62, no.1, pp.219–234, Sept. 2013.

[15] Y. Sadi and S.C. Ergen, "Energy and delay constrained maximum adaptive schedule for wireless networked control systems," IEEE Trans. Wireless Commun., vol.14, no.7, pp.3738–3751, July 2015.

[16] G. Karadag, M.S. Iqbal, and S. Coleri, "Optimal power control, scheduling, and energy harvesting for wireless networked control systems," IEEE Trans. Commun., vol.69, no.3, pp.1789–1801, March 2021.

[17] B. Farayev and S.C. Ergen, "Towards ultra-reliable M2M communication: Scheduling policies in fading channels," Proc. 2016 23rd International Conference on Telecommunications (ICT), pp.1–6, May 2016.

[18] M.S. Akbar, H. Yu, and H. Cang, "IEEE 802.15.4 frame aggregation enhancement to provide high performance in life-critical patient monitoring systems," Sensors, vol.17, no.2, 241, Jan. 2017.

[19] L. Zhang, Y. Gu, R. Wang, K. Yu, Z. Pang, Y. Li, and B. Vucetic "Enabling real-time quality-of-service and fine-grained aggregation for wireless TSN," Sensors, vol.22, no.10, 3901, May 2022.

[20] C.L. Liu and J.W. Layland, "Scheduling algorithms for multiprogramming in a hard-real-time environment," J. ACM (JACM), vol.20, no.1, pp.46–61, Jan. 1973.

[21] J. Nagle, "On packet switches with infinite storage," IEEE Trans. Commun., vol.COM-35, no.4, pp.435–438, April 1987.

[22] C. Lupini, "Vehicle multiplex communication," Vehicle Multiplex Communication: Serial Data Networking Applied to Vehicular Engineering, pp.i–viii, SAE, 2004.

[23] M. Charlier, B. Quoitin, S. Bette, and J. Eliasson, "Support for IEEE 802.15.4 ultra wideband communications in the Contiki operating system," Proc. 2016 Symposium on Communications and Vehicular Technologies, 6 pages, Nov. 2016.

[24] Qorvo Inc., "DW3000 Family User Manual," 2019. https://www.qorvo.com/products/d/da008154

[25] "IEEE Standard for Low-Rate Wireless Networks," IEEE Std 802.15.4-2020 (Revision of IEEE Std 802.15.4-2015), pp.1–800, July 2020.

[26] E.M.L. Beale and J.J.H. Forrest, "Global optimization using special ordered sets," Mathematical Programming, vol.10, no.1, pp.52–69, Dec. 1976.

[27] Qorvo Inc., "DW3000 Datasheet," 2020. https://www.qorvo.com/products/d/da008142

[28] Texas Instruments, "TCAN1042-Q1 Automotive Fault Protected CAN Transceiver with CAN FD datasheet (Rev. D)," 2021. https://www.ti.com/lit/gpn/tcan1042-q1

## Appendix: Derivation of the Inequality for the Weight Parameter $\alpha$

Let us consider the value of $\alpha$ that mainly minimizes the total response frames and also reduces the total latency when minimizing the objective function of Eq. (2). For this sake, the first term should be much larger than the second term of the objective function, which means

$$(1 - \alpha) \sum_{s \in \mathcal{S}} \sum_{j \in \mathcal{J}_s} l_{s,j} \ll \alpha \sum_{k=0}^{T/C} f_k. \qquad (\text{A·1})$$

First, let us consider an ideal situation such that all readout data during $T$ are conveyed via the least number, $F_{\min}$, of the response frames. The value of $F_{\min}$ can be obtained by

$$F_{\min} = \frac{1}{N} \sum_{s \in \mathcal{S}} \frac{C}{c_s}, \qquad (\text{A·2})$$

where recall that $N$ is the maximum number of readout data stored in a response frame, and $C/c_s$ is the number of readout data transmitted during the schedule cycle $T$. The value of $F_{\min}$ can be regarded as the lower bound of the total number of response frames, $\sum_{k=0}^{T/C} f_k$, which appears in the first term of the objective function. Thus, we have

$$F_{\min} \le \sum_{k=0}^{T/C} f_k. \qquad (\text{A·3})$$

Second, let us consider the worst case where all data are read out with the maximum latency $L$. In this case, the total maximum latency $L_{\max}$ is obtained by

$$L_{\max} = L \sum_{s \in \mathcal{S}} \frac{C}{c_s}. \tag{A·4}$$

This can be regarded as the upper bound of the total latency appearing in the second term of the objective function. Thus, we have

$$\sum_{s \in \mathcal{S}} \sum_{j \in \mathcal{J}_s} l_{s,j} \le L_{\max}. \tag{A·5}$$

From Eqs. (A·3) and (A·5), a sufficient condition satisfying the inequality of Eq. (A·1) is given as

$$(1 - \alpha) L_{\max} \ll \alpha F_{\min}. \tag{A·6}$$

Finally, by substituting Eqs. (A·2) and (A·4) to Eq. (A·6) and solving it for $\alpha$, we have the inequality of Eq. (13).

**Chikara Ohta** was born in Osaka, Japan 1967. He received the B.E., M.E., and Ph.D. (Eng.) degrees in communication engineering from Osaka University, Osaka, Japan, in 1990, 1992, and 1995, respectively. From April 1995, he was an assistant professor with the Department of Computer Science, Faculty of Engineering, Gunma University. From October 1996, he was a lecturer at the Department of Information Science and Intelligent Systems, Faculty of Engineering, University of Tokushima, and an associate professor there from March 2001. From November 2002, he was an associate professor of the Department of Computer and Systems Engineering, Faculty of Engineering, Kobe University. From April 2010, he was an associate professor at the Graduate School of System Informatics, Kobe University, and a professor there from January 2015. From April 2016, he was a professor at the Graduate School of Science, Technology and Innovation, Kobe University. Since April 2022, he has been a professor at the Graduate School of System Informatics, Kobe University. From March 2003 to February 2004, he was a visiting scholar at the University of Massachusetts at Amherst, USA. His current research interests include the performance evaluation of communication networks. He is a member of IPSJ, IEEE, and ACM SIGCOMM.

**Hajime Migita** was born in Hyogo, Japan, in 2000. He received the B.E. degree in engineering in 2022, from Kobe University, Hyogo, Japan. He is currently enrolled in the master's program at the Graduate School of Science, Technology and Innovation, Kobe University. He is mainly engaged in research on in-vehicle communications. He is a student member of IEICE.

**Yuki Nakagoshi** is an undergraduate student of the Department of Computer Science and System Engineering at Kobe University. He is mainly engaged in research on in-vehicle communications.

**Makoto Okuhara** was born in Shimane, Japan, in 1977. He received the B.E. in intelligent information engineering from Tottori University, Japan, in 2000. In 2006, he Joined DENSO TEN Co., Ltd, Japan. He serves as Chief Inspector of the Mobility Solution Development Department of the Automotive Electronics Business Headquarters and Project Leader of the Innovation Development Center. He is mainly engaged in in-vehicle ECU development and wireless communication technology research. He is enrolled in the doctor's program at the Graduate School of Science, Technology and Innovation, Kobe University.

**Patrick Finnerty** was born in Cholet, France, in 1995. He received his French Engineering degree in computer science and information technology from INSA Lyon in 2018 and his Ph.D. (Eng.) from Kobe University, Japan, in 2022. Since February 2022, he has been working as an assistant professor at the Graduate School of System Informatics, Kobe University. His research interests include parallel and distributed computing techniques and distributed systems. He is a member of IPSJ, IEEE Computer Society, and ACM.