行政院國家科學委員會專題研究計畫 期中進度報告

數位家庭:網路、平台與應用--子計畫四：數位家庭閘道資源管理(2/3)
期中進度報告(完整版)

計 畫 類 別 ： 整合型
計 畫 編 號 ： NSC 95-2219-E-002-017-
執 行 期 間 ： 95 年 08 月 01 日至 96 年 07 月 31 日
執 行 單 位 ： 國立臺灣大學資訊工程學系暨研究所

計 畫 主 持 人 ： 施吉昇

報 告 附 件 ： 國外研究心得報告
出席國際會議研究心得報告及發表論文

處 理 方 式 ： 期中報告不提供公開查詢

中 華 民 國 96 年 05 月 28 日

行政院國家科學委員會補助專題研究計畫 成果報告

計畫名稱:　　數位家庭:網路、平台與應用-
子計畫四：　數位家庭閘道資源管理(2/3)

計畫類別：□ 個別型計畫　　■ 整合型計畫
計畫編號：NSC 94-2219-E-002-017
執行期間：95 年 08 月 01 日至 96 年 07 月 31 日

計畫主持人：施吉昇教授
共同主持人：
計畫參與人員：林靖邦、陳榮彰、陳勇銘

成果報告類型(依經費核定清單規定繳交)：完整報告

本成果報告包括以下應繳交之附件：
□赴國外出差或研習心得報告一份
□赴大陸地區出差或研習心得報告一份
□出席國際學術會議心得報告及發表之論文各一份
□國際合作研究計畫國外研究報告書一份

處理方式：除產學合作研究計畫、提升產業技術及人才培育研究計畫、列管計畫及下列情形者外，得立即公開查詢
□涉及專利或其他智慧財產權，□一年□二年後可公開查詢

執行單位：國立臺灣大學電信工程學研究所

中　華　民　國　96 年　5 月　25 日

中文摘要

關鍵詞：感測網路、時間同步、點對點通訊

在第二年的計畫中， 我們延續第一年對家庭閘道的發展並發展一個在家庭感測網路中所需的時間同步機制。此一機制為明年度計畫中多媒體點對點傳輸的重要技術。時間同步機制對對許多網路協定均相當重要，也有許多相關的技術發展。但是，在家庭的感測網路中有許多低價的感測器無法使用現有的技術。此一發展的機制，無須調整各個感測器本身的即時時鐘或虛擬時鐘。而是利用一個不會影響感測資料的無線電波，播放一個定期或不定期的訊號，作為時間同步的基礎。這個訊號中並不包含任何的時間資料。每一個感測器，接收並記錄無限訊號。資料中心則負責將各組資料調整至同一時間軸中上。

Abstract

In the second year of the project, we develop a data alignment mechanism for sensor networks. The mechanism allows the sensor data streams to be aligned without real-time clocks or virtual clocks [1]. The clock synchronization is essential for the multi-media end-to-end transmission which will be developed in the next year. The developed mechanism makes use of the built-in counters on sensors and external synchronization signals to align data streams. The data server broadcasts out-of-channel synchronization signals with constant or variable intervals. The sensor data streams are aligned when received on the server. Only one way communication is used so as to reduce the communication overhead and clock synchronization overhead on sensors nodes. In addition, the developed mechanism is scalable thanks to one way communication. The synchronization error is bounded by the maximum sampling period of the sensors, and is independent of the number of the nodes in the network. Our analysis also shows the required awake time for sensors to tolerate different clock drifts and signal lost.

目錄 頁次

I. Introduction

Many emerging sensor network applications such as sniper localization[2],building risk monitoring [3], habitat monitoring [4], and movement tracking [5] require the systems to co-relate the data from different sensors in order to discover certain events or locate where an event occurs. The sensors in the network may need to agree on the time. A global clock in the network will help to process and analyze the data, and to predict future system behavior. However, the low computation capacity on the sensors and unpredictable network propagation delay make it difficult to deploy traditional distributed clock synchronization protocols on such sensor networks.

Clock synchronization for distributed systems has been studied for a long time for traditional distributed and embedded systems [6], [7], [8], [9]. Some of them focused on distributed systems, and some of them focused on high computation bandwidth sensor networks. The recent developed sensors impose new challenges for clock synchronization protocols. H. Dai and R. Han [10] propose a protocol called HRTS (Hierarchy Referencing Time Synchronization). This protocol reduces the number of messages of a synchronization to three messages. In addition, they propose a protocol to synchronize nodes in sensor network via multiple hops. Other related clock synchronization protocols are RBS [11], TPSN [12], and FTSP[13]. The related works may use different manners to synchronize the clocks on sensors but all protocols modify the clocks on the sensors. Ukita and Matsuyama [14] developed the dynamic-memory approach to virtually synchronize the clocks on distributed sensors. The approach makes use of the common events on different sensor streams to compute the sample times for different sensors. Then, it predicts the time instances at which the events of interests may occur on different sensors. The sensors are requested to sample at the given time instances, which may be different for all the sensors. The approach introduces great overhead when there is large number of sensors and is difficult to scale up.

In FireFly project [15], an AM transceiver is used to transmit the synchronization signals and each sensor is equipped with a low cost AM receiver to synchronize its real-time clock with the reference real-time clock. In the project, the sensors have to wake up at right time to listen to the sensors in the neighborhood, listen to the server, and transmit the data. In this way, the sensors can avoid data collision to reduce the power consumption of nodes.

Although clock synchronization helps to the sensors and server to obtain correct timing information as discussed above, global clock synchronization leads to heavy overhead in terms of time and may not be necessary. In addition, in several sensor networks such as habitat monitoring [4], the sensors do not have accurate real-time clocks. Last but not the least, it is often consuming too much energy to continuously transmit sampled data to data servers. To reduce energy consumption, the sensor usually transmits sampled data at a rate which is much less than its sampling rate, or when a certain amount of data is collected. Hence, the time instance at which the data server receives the data may not be the time instance at which the data are sampled. In other words, even there is no network propagation delay, using real-time clock on data server to time stamp the received data could lead to incorrect timing information.

In several sensor networks, some of the sensor nodes or data servers can act as intermediate nodes to aggregate or collect the data streams and send the streams to the servers later. It is sufficient that the data streams are aligned on the intermediate nodes and is not necessary to synchronize the clock on the sensors if there are any. In this paper, we propose to align the received sensor data in a passive manner. In our approach, the clocks on sensors if any are not adjusted at all. After the data streams are received on intermediate nodes or data server, the data streams are aligned and time-stamped. The data streams are aligned based on the embedded synchronization information in the streams. For different sensor network applications, we develop different mechanisms. When the network delay is short and energy consumption is not a critical concern, synchronization information is constantly generated. We prove that the synchronization error is bounded by the maximum sampling period of the sensors and is independent of the number of nodes in the network. When the network delay is long or energy consumption is one of the critical concerns, synchronization signals are

generated with variable intervals to tolerate longer propagation delay. The synchronization error is also bounded by the length of synchronization signal.

II. Formal Model

Thus far, we have used the terms used in wireless sensor network, and embedded systems literature and formally define them below. *Sensor*, denoted by O_i where $1 \leq i \leq N$ and N is the number of sensors in an area of interests, is an analogue/digital device which can monitor certain physical events such as images or sound, and transmits the sampled events in digital form to a data server. We assume that there may be no real-time clocks but simple counters are always available on sensors. The sensors may transmit the sampled events via RF radio or wired network such as CAN [16], bus token-ring, and μITRON [17]. Hence, we assume that there is an upper bound for the transmission delay. *Data server* is a device which receives the data streams from sensors, and analyzes the data streams to discover the interested events. The data server may or may not have real-time clocks. When it has real-time clocks, it time-stamps the received data; when there is no real-time clock, the data server marks the data with non-decreasing counters. For the sake of representation, we assume that there is a real-time clock on data server. Note that our approach is still applicable when real-time clocks are not available on data servers. A *sensor group* consists of at most N sensors and one data server. The data server is referred to sensor O_0 in one sensor group. A *sensor network* consists of a set of sensor groups. We assume that, in this paper, data alignment is conducted for one sensor group but our approach can be easily extended for sensor networks.

Each sensor samples at a constant time interval to record the acoustic events, visual events, or the other physical signals. We denote the samples for sensor O_i by $S_{i,1}$, $S_{i,2}$, etc. The *sample period* for sensor O_i, denoted by sp_i, is the minimal time interval between every two consecutive samples for sensor O_i. *Sample time* for sample $S_{i,j}$, denoted by $st_{i,j}$, is the time instant at which sensor O_i conducts the j-th sample. Because there may be no real-time clocks on the sensors, the sample times are not available in practice. We will use the sample times as the reference to evaluate the accuracy for clock synchronization algorithms. When the data server receives a sample $S_{i,j}$, it estimates the time instance at which the sample might occur. The time instance is called *estimated sample time*, denoted by $\hat{st}_{i,j}$ for sample $S_{i,j}$. By determining estimated sample time for every sample, the data server maps estimated sample times for all samples into one time domain, which could be either a real time clock or a logic clock.

After the sensor samples at one period, it may not immediately send the samples to the server. In practice, the sensor will buffer the sample data, waits for a certain amount of time or having enough data to send, and sends to the collected samples to the server. So, it can reduce the communication overhead and energy consumption. *Buffering time* for sample $S_{i,j}$, denoted by $b_{i,j}$, is the time interval between sample time $st_{i,j}$ and the time instant at which the sample data is sent out. Because the sample period is a constant, the difference of the buffering time for every two consecutive samples is the sample period, which is a constant. Hence, assuming that the buffering time of the last sample in a packet is negligible and 0 will not affect our analysis.

When a sensor may be turned on or off to reduce the energy consumption, it is desirable to know the minimal amount of data that a sensor should collect and transmit to the data server. *Least awake time*, denoted by W, is the minimal length of a time interval during which a sensor cannot be turned off so as to collect enough samples requested by the server. To reduce energy consumption, least wake time should be as short as possible.

Transmission delay for sample $S_{i,j}$, denoted by $d_{i,j}$ and $d_{i,j} \geq 0$ for all i,j, consists of the time interval to content for the transmission media and the network propagation delay. *Maximum transmission delay*, denoted by d^t, is the upper bound to the transmission delay for all samples, i.e., $d^t = \max\{d_{i,j} \exists\ i, j\}$. In some network applications, it is more convenient to know the maximum difference between the transmission delays. We call it *maximum clock drift*, denoted by d^c. In other words, $d^c = \max\{d_{i,m} - d_{j,n} \exists\ \ i, j, m, n\}$.

Arrival time for sample $S_{i,j}$, denoted by $A_{i,j}$, is the time instant at which sample $S_{i,j}$ is received on the data

server. Specifically, arrival time $A_{i,j}$ for sample $S_{i,j}$ is the sum of its sample time, buffering time, and transmission delay, i.e., $A_{i,j} = st_{i,j} + b_{i,j} + d_{i,j}$. Because several samples may be sent in one message, the samples sent by the same message have the same arrival time and same transmission delays. Without loss of generality, we assume that buffering time is zero as discussed above. *Un-buffering arrival time* for sample $S_{i,j}$, denoted by $a_{i,j}$, is the time instant at which sample $S_{i,j}$ is received on the data server when the buffering time is assumed to be zero. In other words, $st_{i,j} + d_{i,j} = a_{i,j}$. For the remainder of this paper, arrival times refer to un buffering arrival times unless it is specifically pointed out.

Data stream, denoted by D_i for $1 \leq i \leq N$ where N is the number of sensors in the sensor group, is a sequence of samples of sensor O_i. We define $D_i = (S_{i,1}, S_{i,2}, \ldots, S_{i,m})$. *Reference data stream*, denoted by $D0$, is the sequence of data bits representing the time sequence for the actual events over the time line. Note that reference data stream is a clairvoyant data stream and may not be available. Reference data stream $D0$ will not be used in the developed mechanism but will be used to verify the proposed mechanism. Because it is often too costly for the sensor to transmit one sample data in every transmission, we assume that every sensor transmits the data to the data server after certain number of samples is collected on the sensor. The data server concatenates the data from each sensor to construct sensing data streams.

Event of interested is the physical event which can be sampled by sensors during a time interval. During the time interval, the environment data can vary with time. We assume that for each event, its duration is greater than any sensor sample period. Hence, every event can be sampled by each sensor at least once. When an event can be sampled by a sensor multiple times, we are only interested in the first sample.

Figure1 is an example for the above terms. This example shows a time sequence for six samples on Sensor O_1: $S_{1,j}$ for $1 \leq j \leq 6$. The line on the top is the sample sequence over time line for sensor O_1. Each solid circle represents the time instance at which a sampling occurs. Specifically, the sample time for $S_{1,j}$ is $st_{1,j}$ shown on the bottom line. In this example, four samples $S_{1,j}$ for $1 \leq j \leq 4$ are buffered till time $st_{1,4}$ and sent to the data server. All four samples are received at time $a_{1,4}$ and have the same transmission delay. The solid directed line represents the transmission from sensor O_1 to the data server; the dashed lines represent the virtual transmissions from sensor O_1 to the server. Each sample has different buffering time $b_{1,j}$ for $1 \leq j \leq 4$ and its (un-buffering) arrival time is $a_{1,j}$ for $1 \leq j \leq 4$.

Synchronization error of two samples $S_{i,m}$ and $S_{i,n}$ is the difference between estimated sample times and sample times (i.e., $|(st_{i,m} - st_{j,n}) - (\hat{st}_{i,m} - \hat{st}_{j,n})|$). We will use synchronization error to evaluate the accuracy of the algorithms. The less the synchronization error, the more accurate the algorithm.

The Data Stream Alignment problem is defined as follows.

Definition 2.1 (Data Stream Alignment Problem): We are given a set of data streams: $\{D_1, D_2, \ldots, D_n\}$, maximum transmission delay d^t, and maximum clock drift d^c. The problem is to assign estimated sample time for each interested sample such that for every two samples $S_{i,m}$ and $S_{j,n}$, their synchronization error is bounded.

Examples shown in Figure 2 (a), (b), and (c) illustrate the Data Stream Alignment problem and expected results. Figure2(a) shows the sensor samplings over real-time clock. Suppose an event starts at time 1 and ends at time 7. The circles represent the samples on each sensor. Solid circles represent that the sensor samples the event; empty circles represent no event are sampled by the sensor. The sample periods for sensors O_1, O_2, and O_3, are 2, 3, and 4 units of time. Figure 2 (b) shows the sample times computed by the data server before stream alignment. In this example, the data server may consider that there are two events in the interested area: one starts at time3 and ends at time 7, and the other starts at time9 and ends at time 11. Figure2(c) shows the estimated sample times after stream alignment. The estimated sample times for sample $O_{2,i}$ are aligned with that for sample $O_{1,j}$ and $O_{3,m}$. Consequently, the data sever considers that there is only one event starting at time 2 and ending at time 7. In this example, the synchronization error is one unit of time.

In this paper, we assume that the sensor network only aligns data streams on request. This is because it is not necessary to align the data streams when every signal is sampled. When the data server finds one

interested event on some data stream, it starts to align the data streams to confirm the event. In other words, the data server starts to collect necessary information after the request arrives. Hence, the proposed approach has low space and time complexity to align the data streams.

III. Clock Free Data Streams Alignment

To align the data streams, we propose to use synchronization signals. When a sensor receives the synchronization signals, it uses a different representation to distinguish a synchronization signal from other events. After receiving the data streams from the sensors, the data server identifies the synchronization signals in the data streams and uses the temporal information for synchronization signals to align the data streams. By so doing, the data streams from different sensors are aligned to one reference time-line, i.e., the real-time clock or logic clock on the data server, but there is no need to adjust the real-time clocks on sensors.

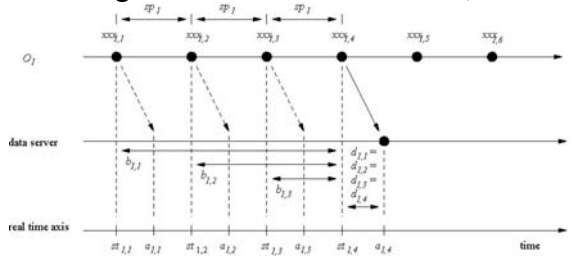

Fig. 1. An example of terminologies

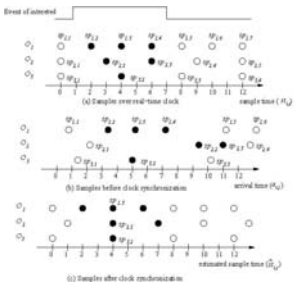

Fig. 2. An Example of Stream Alignment Problem

The proposed approach is similar to the one used in FireFly project [15]. In FireFly project, the sensors have to wake up at right time to listen to the sensors in the neighborhood, listen to the server, and transmit the data. However, in the scenario with which we are concerned, the sensors only transmit the data to the server. There is no need for the sensors to wake up for listening at any specific time. Hence, it is not necessary to adjust the real-time clocks on each of the sensors.

Our observations for Data Stream Alignment problem are the following. First of all, the timing information for the sampled data affects the inference results only when the data are collected and are being co-related by the data server. Second, only the relative timing information of the data streams matters. As shown in Figure 2, the data server can recognize that one interested event is detected by three sensors only when the data server concludes that samples $S_{1,1}$, $S_{2,1}$, and $S_{3,2}$ are sampled at the same time instance.

The correctness of the real-time clocks on sensors or data servers does not affect the result of aligning the data streams. Furthermore, we argue that it introduces unnecessary costs to install additional receivers to achieve clock synchronization. The field sensors are ready to sample certain signals. For the sensor networks interested in acoustic signals, acoustic sensors should be deployed in the network. For the networks interested

in video surveillance, image sensors should be deployed in the network.

In our approach, the synchronization server broadcasts out signals which can be sampled by the sensors but will not affect the sample results. For example, a sensor network in battle field consists of acoustic sensors to detect sniper location. The sensors can sample sound ranging from 100Hz to 20kHz. However, the frequency of gun shot ranges from 1000Hz to 10kHz.We can choose 15kHz frequency signals as the synchronization signals, which is not affected by the background noise and will not affect the sampling results. Then, the server broadcasts the 15kHz frequency sound wave as the synchronization signals, which do not interfere the interested signals. *Synchronization server* is a device which generates synchronization signals for the sensors. The signals are generated in a periodic or aperiodic manner. In a sensor network, there can be more than one synchronization server to generate the signals for different types of sensors to extend the coverage of the networks. However, the signals should be generated with the same schedule. We will discuss how to determine the schedule later. When a sensor samples the synchronization signals, it marks the occurrence of the signals in its data stream and transmits the stream to the data server.

Synchronization interval for sensor O_i, denoted by $c_{i,j}$, is the number of samples between (j- 1)-th and *j*-th synchronization signals for stream D_i. *Synchronization event stream* for sensor O_i, called *stream* for short and denoted by Si for i>=1, is a sequence of synchronization intervals for a data stream D_i. In other words, stream S_i is defined as $S_i = <c_{i,2}, c_{i,3}, \ldots, c_{i,n_i}>$ where n_i is the number of synchronization signals for stream S_i. When the data server receives one data stream, it first counts the number of samples between two synchronization signals in the stream to compute synchronization interval $c_{i,j}$ and constructs synchronization event stream S_i. *HappenTogether* for two samples $S_{i,m}$ and $S_{j,n}$ where i =j represents that one event is sampled by both samples.

In our model, the sensor samples one instance for each sample period. To guarantee that the synchronization signal will be sampled by sensors. The length of synchronization signal is no less than the maximum sampling period, i.e., $\max\{sp_i\}$. This design is common to real-world applications. For instance, an acoustic sensor can sample at 10k Hz or 44k Hz. Hence, as long as the time interval for synchronization signal is longer than 0.1 ms, the acoustic sensor can always sample the synchronization signals.

The clock free data streams alignment approach consists of the following three steps.

Step 1: Determining the schedule for synchronization signals: In this step, the synchronization server determines the schedule to generate the synchronization signals. The signals can be generated with constant or variable intervals. The schedule for synchronization signals affects the tolerable transmission delay, necessarily nodes being awake time, and capability to recover lost signals. For the sake of power saving or sensor control, sensors may be turned on/off by other mechanism. It is not guaranteed that a sensor will receive all synchronization signals. However, in order to align the data streams, the sensor may be requested to be awake for a certain amount of time to receive enough signals. A good schedule for synchronization signals should minimize the least awake time. In addition, the schedule also affects the capability of discovering lost signals. A good synchronization signal schedule should allow the data server to discover a lost synchronization signal in the given streams. We will discuss how to generate the schedule for different types of sensor networks in the following two sections.

Step 2: Determining the temporal order of received data streams: After receiving the data streams, the data server determines the temporal order of the data streams based on the synchronization intervals in every data stream. The schedule of synchronization signals in Step 1 forms a repeating pattern for the sequence of synchronization intervals. Comparing the sequence of the synchronization intervals, the data server determines the temporal order of the data streams.

Step 3: Assigning HappenTogether relations: After determining the temporal order, the data server can determine the HappenTogether relations among synchronization signals. Suppose the latest stream and second latest stream are $S_i = <c_{i,2}, \ldots, c_{i,n_i}>$ and $S_m = <c_{m,2}, \ldots, c_{m,n_m}>$, and the synchronization signals are $s_{i,1}, \ldots, s_{i,n_i}$ for S_i, $s_{m,1}, \ldots, s_{m,n_m}$ for S_m. By some method which we will discuss later, we can find a

synchronization signal $s_{i,j}$ of which the estimated sampled time is equal to that s_{m,n_m}. So we can assign the HappenTogether relations between these two samples. Then, we begin to assign estimated sample time for other samples. Suppose $s_{i,j}$ is recorded in $S_{i,k}$, and $S_{m,s}$ for s_{m,n_m}.We set $t_{i,p} = t_{i,k} + (p - k) \times sp_i$ and $t_{m,q} = t_{m,n_m} + (q - n_m) \times sp_m$. So, we can map all samples for all sensors to a single time axis.

With above three steps, the data server finds out samples with the same synchronization signal, and sets the same estimated sample time for these samples having HappenTogether relation. Because the length of synchronization signal must be longer or equal to the longest sample period in all sensors, the synchronization signal length is set as the maximum sampling period.

One important issue for the approach is to design the schedule for synchronization signals. In the following two sections, we discuss two policies for generating synchronization signals: constant interval and variable interval for synchronization signals, and how to determine the temporal order of the streams.

IV. Constant Interval Synchronization Signals

The simplest policy is to generate the synchronization signals at a constant rate. In other words, the synchronization server generates the signals with a constant interval. *Synchronization period*, denoted by *p*, is the minimal time interval within which the synchronization signal schedule does not repeat. In this section, the synchronization period is equal to the synchronization signal period. A good synchronization schedule should prolong the synchronization period and shorten the least awake time. A long synchronization period allows the system to tolerate greater clock drift error; a short least awake time allows the sensors to consume less energy and the server to complete the data stream alignment earlier. (The server has to wait for all the sensors sampling for least awake time.) We propose two policies for generating constant interval synchronization signal schedule: *simple constant interval* and *constant interval with different signals*. In the following, we discuss how to determine the synchronization period and how to determine the temporal order of data streams. We design two period assignment algorithms: one for the case that maximum transmission delay d^t is known in advance and the other one is for the case that the maximum clock drift d^c is known in advance.

A. Period Assignment for Known Maximum Transmission Delay

When there is an upper bound d^t of all samples transmission delays and the length of one synchronization signal is *l*, we set the synchronization period to $d^t + l$ units of time (i.e., synchronization signals are generated for every $d^t + l$ units of time). While the period is used, we claim that the synchronization error is bounded by the length of synchronization signals, i.e., *l*.

The requirements for assigning the synchronization period is that the data server shall be able to determine the temporal order of the received data streams and assign the estimated sample time for every sample. When the maximum transmission delay is d^t, for any sample $S_{i,m}$ whose arrival time is $a_{i,m}$, we know that the sample time must be in the time interval $(a_{i,m} - d^t - l, a_{i,m})$. The right end of the interval represents the case that the sensor samples the end of the signal and it takes almost no time to transmit the message. The left end of the range represents the case that the sensor samples the beginning of the synchronization signal and it takes the maximum transmission delay to send the message. Because the synchronization period is $d^t + l$, there is one and only one signal in time interval $(a_{i,m} - d^t - l, a_{i,m})$.

Figure 3 shows an example in which the maximum transmission delay is eight units of time and synchronization signal length is two units of time. In the figure, the dark circle and gray circle represent the first and second synchronization signals received by the sensors. According to the known maximum transmission delay and the length of synchronization signals, we know that the first synchronization signal

and second synchronization signal will be received in interval (0, 10) and (10, 20), respectively. For the stream, we choose the sample $S_{i,k}$ which samples the synchronization signal and whose arrival time is the maximal. Suppose the arrival time for sample $S_{i,k}$ is $a_{i,k}$, the time to broadcast the sampled signal must be in the range $(ai,k - d^t - l, a_{i,k})$. Suppose the time instance to broadcast the signal received in the range $(a_{i,k} - d^t - l, a_{i,k})$ is t_b, the estimated sample time $\hat{st}_{i,k}$ is set as t_b. Finally, we assign estimated sample times to other samples by adding/subtracting its sample period sp_i. The CLOCK FREE ALIGNMENT WITH KNOWN MAXIMUM TRANSMISSION DELAY algorithm is shown in Algorithm 1.

Algorithm 1 CLOCK FREE ALIGNMENT WITH KNOWN MAXIMUM TRANSMISSION DELAY

Input: 1. data streams $D_i = (\mathbb{S}_{i,1}, \mathbb{S}_{i,2}, \ldots, \mathbb{S}_{i,n_i})$ for $1 \le i \le N$
2. arrival time $a_{i,m}$ for sample $\mathbb{S}_{i,m}$ for $1 \le i \le N$ and $1 \le m \le n_i$
3. maximum transmission delay d^t
4. synchronization signal length l

Output: estimated sample time $\hat{st}_{i,m}$ for $1 \le i \le N$ and $1 \le m \le k_i$.

1: **for** $i = 1$ to N **do**
2: Select a sample $\mathbb{S}_{i,m}$ in D_i which represents a synchronization signal.
3: Find the synchronization signal broadcasted in interval $(a_{i,m} - d^t - l, a_{i,m})$ on synchronization server and assign the broadcasting time of the signal to estimated sample time $\hat{st}_{i,m}$.
4: **for** $k = 1$ to n_i **do**
5: Assign estimated sample time $\hat{st}_{i,k} = \hat{st}_{i,m} + (k - m) \times sp_i$.

Step 2 of this algorithm takes n_i units of time to find a sample whose synchronization signal is in the worst case. Step 3 takes constant time to find the corresponding synchronization signal. Step 5 is also in constant time. Hence, the complexity is $O(n)$, where n is the total input size.

We claim that at the start of each iteration of the iterative loop, we assign estimated sample times for D_1, ..., D_{i-1}, and $0 \le \hat{st}_{j,m} - st_{j,m} \le l$ for $1 \le j \le i - 1$. Before the first iteration of the loop, there is no streams in D_1, ..., D_{i-1}. Hence, the loop invariant holds. If the loop invariant holds after the i-th iteration, in the i +1-th iteration we will assign estimated sample time for Di+1. Because we assign the time instance at the start of synchronization signal to the estimated sample time, the estimated sample time is no later than the sample time, and difference is bounded by l. The loop invariant holds after the i +1-th iteration. After all iteration, for any two sample $S_{i,m}$ and $S_{j,n}$, $|(st_{i,m} - \hat{st}_{i,m}) - (st_{j,n} - \hat{st}_{j,n})| \le |st_{i,m} - \hat{st}_{i,m}| - |st_{j,n} - \hat{st}_{j,n}| \le l$. In other word, the synchronization error is bounded by l.

We claim that by the algorithm, the synchronization error is at most l units of time. The main idea is that in the above procedure, we assign the estimated sample time to a stream by shifting the sample time of the stream. The displacement of each stream is in the range from 0 to l. Hence, the synchronization error is at most l. The proof for the claim follows.

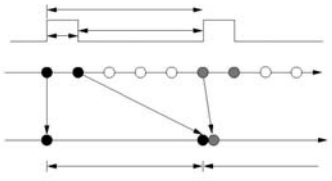

Fig. 3. An example for constant Interval when dt =8 and l =2

Theorem 1: Given a set of data stream, the maximum transmission delay, length of synchronization signal, CLOCKFREE ALIGNMENTWITHKNOWNMAXIMUM TRANSMISSION DELAY algorithm assigns the estimated sample time for each sample in the data streams such that the synchronization error is bounded by l.
Proof: Let S_{i,m_i} be the sample which represents a synchronization signal in stream D_i. Because the time

difference between sample time st_{i,m_i} and the beginning of the synchronization signal is at most l, the difference between the estimated sample time $\hat{st}_{i,m_i}$ and the sample time st_{i,m_i} is at most l (i.e., $\hat{st}_{i,m_i} - st_{i,m_i} \leq l$). Furthermore, we assign estimated sample time $\hat{st}_{i,m} = \hat{st}_{i,m_i} + sp_i \times (m - m_i)$ to sample $S_{i,m}$. Hence, for any two samples, $S_{i,m}$ and $S_{j,n}$, the synchronization error $(st_{i,m} - st_{j,n}) - (\hat{st}_{i,m} - \hat{st}_{j,n}) = |(\hat{st}_{j,n} - st_{j,n}) - (\hat{st}i,m - st_{i,m})| = |\hat{st}_{j,n} - st_{j,n}| - |st_{i,m} - st_{i,m})| \leq l$. We can obtain the result that the synchronization error is at most l.

B. Period Assignment for Known Maximum Clock Drift

In this subsection, we are concerned with the case that the maximum clock drift d^c is known. As a reminder, the maximum clock drift d^c is the maximum difference between any two transmission delays. In other words, $| di,m - dj,n | < d^c$ for all i, j, m, n. When the maximum clock drift is d^c, the synchronization period is set as $2(d^c + l)$.We claim that the synchronization error is also bounded by the length of synchronization signal, i.e., l.

Before we present the algorithm to align the data streams, we define a sequence of interval $Ti \equiv ((2_i - 1)(d^c - l), (2i + 1)(d^c - l))$ for the algorithm. We claim that for two samples of which the arrival time is $a_{i,m}$ and $a_{j,n}$, if $a_{j,n} - a_{i,m}$ is in T_i, and $S_{i,m}$ samples the p-th signal, it implies that the $S_{j,n}$ samples the p + i-th signal.

Corollary 4.1: Given two samples $S_{i,m}$ and $S_{j,n}$ of which the arrival time is $a_{i,m}$ and $a_{j,n}$, $a_{j,n} - a_{i,m}$ is in T_i, and $S_{i,m}$ samples the p-th signal, $S_{j,n}$ samples the p + ith signal.

CLOCK FREE ALIGNMENT WITH MAXIMUM CLOCK DRIFT algorithm determines the estimate sample times for the stream and is listed in Algorithm 2.

Algorithm 2 CLOCK FREE ALIGNMENT WITH MAXIMUM CLOCK DRIFT

Input: 1. data streams $D_i = (\mathbb{S}_{i,1}, \mathbb{S}_{i,2}, \ldots, \mathbb{S}_{i,n_i})$ for $1 \leq i \leq N$
2. arrival time $a_{i,m}$ for $1 \leq i \leq N$ and $1 \leq m \leq n_i$
3. maximum clock drift d^c.
4. synchronization signal length l

Output: estimated sample time $\hat{st}_{i,m}$ for $1 \leq i \leq N$ and $1 \leq m \leq n_i$.

1: Choose sample $\mathbb{S}_{1,m_1}$ in data stream D_1, where $1 \leq m \leq n_1$.
2: Assign estimated sample time $\hat{st}_{1,m_1} = 0$.
3: **for** $m = 1$ to k_1 **do**
4: Assign estimated sample time $\hat{st}_{1,m} = \hat{st}_{1,m_1} + (m - m_1) \times sp_1$
5: **for** $i = 2$ to N **do**
6: Find a sample $\mathbb{S}_{i,m_i}$ recording a signal.
7: Assign estimated sample time $\hat{st}_{i,m_i} = k(2(d+l))$ where $a_{i,m_i} - a_{1,m_1}$ is in T_k.
8: **for** $m = 2$ to n_i **do**
9: Assign estimated sample time $\hat{st}_{i,m} = \hat{st}_{i,m_i} + (m - m_i) \times sp_i$.

Step8 in this algorithm is also in constant time. The other steps are the same as the steps in Algorithm 1. Hence, the total time complexity is also $O(n1 + \ldots + nN)$. The time complexity is $O(n)$.

The proof of correctness is similar to the proof of Algorithm 1. There is a constant α such that for each sample $S_{i,m}$ in this algorithm, the estimated sample time satisfies $\alpha \leq st_{i,m} - \hat{st}_{i,m} \leq 1 + \alpha$. The constant α is the difference between the real time and the logical time. Hence, for each two sample $S_{i,m}$ and Sj,n, $|(st_{i,m} - \hat{st}_{i,m}) - (st_{j,n} - \hat{st}_{j,n})| \leq |st_{i,m} - \hat{st}_{i,m}| - |st_{j,n} - \hat{st}_{j,n}| \leq (1 + \alpha) - (\alpha) = l$. In other word, the synchronization error is bounded by l.

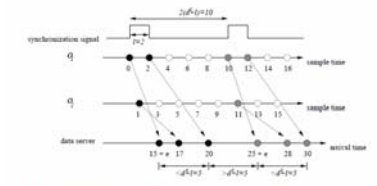

Fig. 4. An example for constant interval when $d^c = 3$ and $l = 2$

Figure 4 shows an example for the case that maximum clock drift is three units of time and the length of synchronization signal length is two units of time. We can find that there are no samples sensing a signal whose arrival time is in the time intervals (20,25], (30,35], etc. If we construct a logical time which sets the arrival time for some sample recording a signal called sig^* be 0. Then, for any sample recording a signal logical arrival time in *Ti*, the signal differs sig^* exactly i signals.

In summary, the synchronization error is bounded by the length of the synchronization signal, i.e., *l*. When the maximum clock delay d^t is known, the least awake time for the sensors is $d^t + 1$ units of time; when the maximum clock drift d^c is known, the least awake time is $2(d^c + l)$ *units of time.*

V. Self-estimation

We summarize our research achievement of this sub-project in this year here:

1. A novel home network synchronization algorithm, the Clock Free Synchronization Algorithm, for use：

   With this protocol, the low-cost sensor in home network could be virtually synchronized without local real-time clocks. In other words, it is not necessary to synchronize the clocks if exist on each sensor. However, the sensed data can be synchronized on one global time-line on the data server. It will help the sensors in digital home to collaborate and collect accurate data.

2. A framework for peer-to-peer multimedia transmission with QoS guarantee is under-development. This framework includes the wireless bandwidth estimation, wireless bandwidth management on local access point, and end-to-end QoS management for peer-to-peer streaming. This framework will allow the multimedia streaming among digital homes become possible.

VI. Summary

In this paper, we are concerned with aligning the data streams for sensor networks in which the low cost sensors may not have real-time clocks. We propose to use the synchronization signals as the reference for aligning data streams. When the clock drifts among the sensors is small and there is no signal lost, one can generate the synchronization signals with constant interval. When the clock drifts among the sensors is large or the sensors may miss synchronization signals, one can generate the synchronization signals with variable intervals. Using this approach, one can avoid the overhead for synchronizing the local real-time clocks in the sensor networks on a global clock. The data server can adjust the time stamps on data streams to obtain correct relative timing information among streams. In the future, we will extend the work for recovering the lost signals. The data server may need to buffer additional counters or additional rules are needed to generate synchronization round. In this way, the data server can insert the lost signals back to the streams.

References

[1] L. Lamport, “Time, clocks, and the ordering of events in a distributed system,” *Communications of theACM*, vol. 21, no. 7, pp. 558 – 565, 1978.
[2] Q. Wang, R. Zheng, A. Tirumala, X. Liu, and L. Sha, “Lightning: A fast and lightweight acoustic localization protocol using low-end wireless micro-sensors,” in *The 25th IEEE International Real-Time Systems Symposium (RTSS’04)*, 2004, pp. 371 – 381.
[3] N.Kurata, B.F. Spencer, Jr., and M. Ruiz-Sandoval, “Application of wireless sensor network mote for building risk monitoring.”
[4] A. Cerpa, J. Elson, D. Estrin, L. Girod, M. Hamilton, and J. Zhao, “Habitat monitoring: Application driver for wireless communications technology,” in *Proceedings of the 2001 ACM SIGCOMM Workshop on Data Communications in Latin America and the Caribbean*, April 2001.
[5] R. Orr and G. Abowd, “The smart floor: A mechanism for natural user identification and tracking,” in *Conference on HumanFactors in Computing Systems, The Hague*, The Hague, Netherlands, April 2000.
[6] D. L. Mills, “Precision synchronization of computer network clocks,” *ACM/IEEETransactions on Networking*, vol. 6, no. 5, pp. 505 – 514, 1998.
[7] C. Liao,M. Martonosi, andD.W. Clark, “Experience with an adaptive globally-synchronizing clock algorithm,” in *ACM SPAA*, June 1999, pp. 106–114.
[8] F. Cristian, “Probabilistic clock synchronization,” *Distributed Computing*, vol. 3, no. 146 – 158, 1989.
[9] A. Ramanathan, K. G. Shin, and R. W. Butler, “Fault-tolerant clock synchronization in distributed systems,” *IEEE Computer*, pp. 33 – 42, October 1990.
[10] H. Dai and R. Han, “Tsync:Alightweight bidirectional time synchronization service for wireless sensor networks,” in *ACM SIGMOBILE Mobile Computing and Communications Review*, 2004.
[11] J. Elson, L. Girod, and D. Estrin, “Fine-grained time synchronization using reference broadcasts,,” in *Proceedings of the 5th Symposium on Operating Systems Design and Implementation*, December 2002, pp. 147 – 163, rBS.
[12] S.Ganeriwal,R.Kumar, andM. Srivastava,“Timing-sync protocol for sensor networks,” in *Proceedings of the 1st International Conference on Embedded Networked Sensor Systems*, November 2003, pp. 138 – 139.
[13] M. Maroti, B. Kusy, G. Simon, and A. Ledeczi, “The flooding time synchronization protocol,” in *Proceedings of the 2nd international conference on Embedded networked sensor systems*, Baltimore, MD, USA, 2004.
[14] N. Ukita and T. Matsuyama, “Real-time multi-target tracking by cooperative distributed active vision agents,” in *AAMAS*, Bologan, Italy, July 2002, pp. 829 – 838.
[15] A. Rowe, R. Mangharam, and R. Rajkumar, “RT-Link: A Time-Synchronized Link Protocol for Energy Constrained Multi-hopWireless Networks,” Department of Electrical and Computer Engineering, Carnegie Mellon University,Tech. Rep., August 2005.
[16] P. Richardson, L. Seih, and P. Haniak, “A real-time control network protocol for embedded systems using controller area network(CAN),” in *IEEE Elestronics and Information Technology Conference*, June 2001.
[17] H. Mori,Y. Mano,H.Takada, andK. Sakamura, “*μ*ITRONbus:a real-time control lan for open network environment,” in *ThirdInternational Workshop on Real-Time Computing Systems Application*, 1996, pp. 227–234.

2006 Fourteenth IEEE International Conference on Networks

施吉昇

國立台灣大學資訊網路與多媒體研究所

一. 參訪經過及會議參加心得

此次參訪係於九十五年 9 月 13 日至 9 月 15 日參加 2006 年在新加坡所舉行的 2006 Fourteenth IEEE International Conference on Networks (ICON 2006)。本屆 ICON 會議係第十四屆年會，議程的安排包含一天的 tutorial 以及兩天的技術會議。

本次會議的參加者共來自於二十六國分佈於五大州：美洲、亞洲、歐洲、大洋洲以及非洲。本屆的會議共有 261 篇論文投稿，大會在經過嚴格的論文審核程序後僅接受 101 篇論文。論文的接受率為 38%。所有接受的論文被分為七個技術組(Technical Session)，在三個同時進行的 track 發表。

此次的會議除了有 101 篇高品質的論文發表外，主辦單位同時安排了一個專題演講，一個討論會以及四個半天的工作會。專題的講者是在數位家庭研究領域中的重要學者：Dr. Sajal K. Das (University of Texas at Arlington)。Dr. Das 發表超過四百篇會議與期刊論文，在二十五本以上的專書中提供專章，以及五個美國的專利。Dr. Das 同時也得過許多的獎項：包含五篇最佳論文獎。同時，他也是許多期刊的編審委員，例如：IEEE Transactions on Mobile Computing, IEEE Transactions on Parallel and Distributed Systems, ACM/Springer Wireless Networks, and Journal on Emergent Parallel and Distributed Systems。在本次會議的專題演講，他以德州大學的數位家庭計畫為題，發表該計畫的執行成果，並探討未來數位家庭網路的發展前景。

本人亦以” Threat-Based Configuration Architecture for Security Gateways”為題發表一篇台灣大學在數位家庭計畫中的網路閘道的網路安全系統。

此次行程除了參加 2006 ICON 外並接受新加坡大學 Dr. Stefan Andrei 的邀請進行一天的訪問。在一天的訪問中，我們就彼此的研究成果交換意見，在計算機系(Department of Computer Science)以筆者的研究成果為題演講，並就新加坡大學鼓勵教授參與國際會議的作法交換意

見，以作為本系鼓勵教授參與國際活動的參考。

整體而言，此次的參訪對筆者有相當大的收穫。首先，透過會議中的論文發表，更清楚的瞭解各國的學者在數位家庭網路方面的研究方向與成果，雙方在研究方向上有相當大的一致性，但是，在研究方法以及成果方面則具有互補性。

二. 攜回資料名稱與內容

筆者攜回 ICON 會議論文集一份以及相關之展示軟體作為後續合作之參考。

2006 Fourteenth IEEE International Conference on Networks

施吉昇

國立台灣大學資訊網路與多媒體研究所

一. 參訪經過及會議參加心得

此次參訪係於九十五年 9 月 13 日至 9 月 15 日參加 2006 年在新加坡所舉行的 2006 Fourteenth IEEE International Conference on Networks (ICON 2006)。本屆 ICON 會議係第十四屆年會，議程的安排包含一天的 tutorial 以及兩天的技術會議。

本次會議的參加者共來自於二十六國分佈於五大州：美洲、亞洲、歐洲、大洋洲以及非洲。本屆的會議共有 261 篇論文投稿，大會在經過嚴格的論文審核程序後僅接受 101 篇論文。論文的接受率為 38%。所有接受的論文被分為七個技術組(Technical Session)，在三個同時進行的 track 發表。

此次的會議除了有 101 篇高品質的論文發表外，主辦單位同時安排了一個專題演講，一個討論會以及四個半天的工作會。專題的講者是在數位家庭研究領域中的重要學者：Dr. Sajal K. Das (University of Texas at Arlington)。Dr. Das 發表超過四百篇會議與期刊論文，在二十五本以上的專書中提供專章，以及五個美國的專利。Dr. Das 同時也得過許多的獎項：包含五篇最佳論文獎。同時，他也是許多期刊的編審委員，例如：IEEE Transactions on Mobile Computing, IEEE Transactions on Parallel and Distributed Systems, ACM/Springer Wireless Networks, and Journal on Emergent Parallel and Distributed Systems。在本次會議的專題演講，他以德州大學的數位家庭計畫為題，發表該計畫的執行成果，並探討未來數位家庭網路的發展前景。

本人亦以” Threat-Based Configuration Architecture for Security Gateways”為題發表一篇台灣大學在數位家庭計畫中的網路閘道的網路安全系統。

此次行程除了參加 2006 ICON 外並接受新加坡大學 Dr. Stefan Andrei 的邀請進行一天的訪問。在一天的訪問中，我們就彼此的研究成果交換意見，在計算機系(Department of Computer Science)以筆者的研究成果為題演講，並就新加坡大學鼓勵教授參與國際會議的作法交換意

見，以作為本系鼓勵教授參與國際活動的參考。

整體而言，此次的參訪對筆者有相當大的收穫。首先，透過會議中的論文發表，更清楚的瞭解各國的學者在數位家庭網路方面的研究方向與成果，雙方在研究方向上有相當大的一致性，但是，在研究方法以及成果方面則具有互補性。

二. 攜回資料名稱與內容

筆者攜回 ICON 會議論文集一份以及相關之展示軟體作為後續合作之參考。