Technical Report

Data Gathering Approach in Dense Sensor Networks

Maryam Vahabi

Eduardo Tovar

HURRAY-TR-121003

Version:

Date: 10-12-2012

Data Gathering Approach in Dense Sensor Networks

Maryam Vahabi, Eduardo Tovar

IPP-HURRAY!

Polytechnic Institute of Porto (ISEP-IPP)

Rua Dr. António Bernardino de Almeida, 431

4200-072 Porto

Portugal

Tel.: +351.22.8340509, Fax: +351.22.8340509

E-mail:

http://www.hurray.isep.ipp.pt

Abstract

Sensor/actuator networks promised to extend automated monitoring and control into industrial processes. Avionic system is one of the prominent technologies that can highly gain from dense sensor/actuator deployments. An aircraft with smart sensing skin would fulfill the vision of affordability and environmental friendliness properties by reducing the fuel consumption. Achieving these properties is possible by providing an approximate representation of the air flow across the body of the aircraft and suppressing the detected aerodynamic drags. To the best of our knowledge, getting an accurate representation of the physical entity is one of the most significant challenges that still exists with dense sensor/actuator network. This paper offers an efficient way to acquire sensor readings from very large sensor/actuator network that are located in a small area (dense network). It presents LIA algorithm, a Linear Interpolation Algorithm that provides two important contributions. First, it demonstrates the effectiveness of employing a transformation matrix to mimic the environmental behavior. Second, it renders a smart solution for updating the previously defined matrix through a procedure called learning phase. Simulation results reveal that the average relative error in LIA algorithm can be reduced by as much as 60% by exploiting transformation matrix.

Data Gathering Approach in Dense Sensor Networks

Maryam Vahabi, Eduardo Tovar
CISTER/ISEP, Polytechnic Institute of Porto
Porto, Portugal
{mmvi,emt}@isep.ipp.pt

Abstract:

Sensor/actuator networks promised to extend automated monitoring and control into industrial processes. Avionic system is one of the prominent technologies that can highly gain from dense sensor/actuator deployments. An aircraft with smart sensing skin would fulfill the vision of affordability and environmental friendliness properties by reducing the fuel consumption. Achieving these properties is possible by providing an approximate representation of the air flow across the body of the aircraft and suppressing the detected aerodynamic drags. To the best of our knowledge, getting an accurate representation of the physical entity is one of the most significant challenges that still exists with dense sensor/actuator network. This paper offers an efficient way to acquire sensor readings from very large sensor/actuator network that are located in a small area (dense network). It presents LIA algorithm, a Linear Interpolation Algorithm that provides two important contributions. First, it demonstrates the effectiveness of employing a transformation matrix to mimic the environmental behavior. Second, it renders a smart solution for updating the previously defined matrix through a procedure called learning phase. Simulation results reveal that the average relative error in LIA algorithm can be reduced by as much as 60% by exploiting transformation matrix.

1. Introduction

Automation of monitoring and control is essential in today's industrial processes. With the fast paced growth in electronics technology, communications and networking, the cost of a sensor node is decreased towards zero. Subsequently, it is economically feasible to deploy dense networks of sensor nodes for monitoring and controlling physical parameters. Since the beginning of the 20th century, a big revolution in information technology transformation has appeared which is still in progress. This revolution is coming from massively deploying networked embedded computing device allowing instrumenting the physical world with pervasive sensor-rich embedded computation [1]. Accordingly, the term Cyber Physical System (CPS) has come to describe the research and technological efforts that will ultimately enable the interlinking of the real-world physical objects and cyberspace [2].

Very dense networks offer a better resolution of the physical world and therefore a better capability of detecting the occurrence of an event. Structural health monitoring (SHM) of physical infrastructures (aircrafts, bridges, etc.) is a well-known example of CPS applications [3]. One effective usage of high-spatial resolution sensor/actuator networks can be seen in avionics; specifically for efficient fuel consumption. Reducing fuel consumption is equally important for environmental effects (CO emission) as well as cost efficiency (business and economics). The potential for a 50% reduction in fuel consumption in the next 15 years can be attained using a combination of

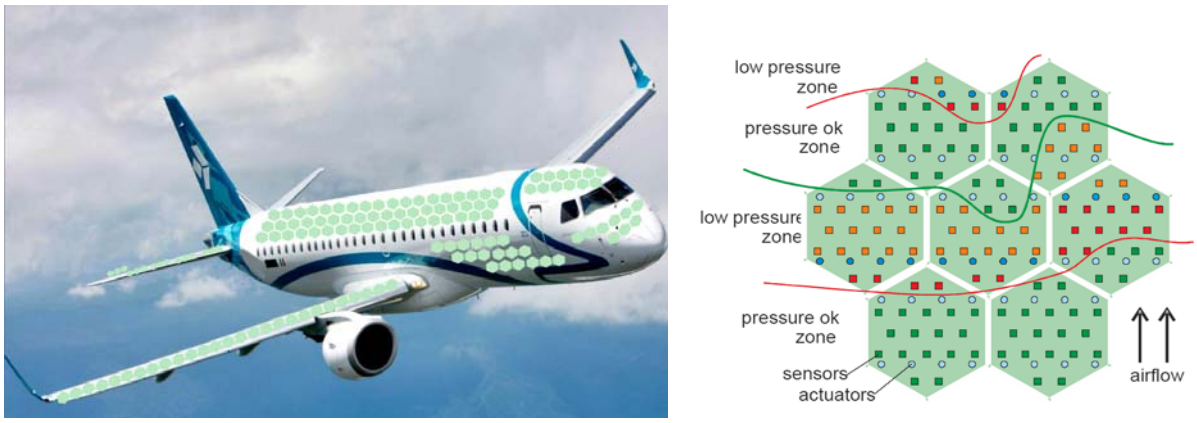

Figure 1. Active air flow control by deploying dense sensor/actuator across aircraft wings and fuselage.

aerodynamic, engine, and structural improvements [6]. Active air flow control is proposed as a promising solution to reduce fuel consumption and emissions [4] by reducing the aerodynamic drags that is one of the most significant factors in increasing aircraft fuel burn. For a typical long range aircraft at cruise condition, skin friction drag with the rate of more than one half (53%) and lift-induced drag with the rate of less than one third (21%) of total drag are the two important sources of aerodynamic drags [4, 5]. Therefore, a remarkable increase in fuel efficiency can be expected with successful skin friction drag reduction via active flow control of smart skin systems—see Figure 1. As illustrated in Figure 1, implementing the air flow control will require a reliable and highly fault tolerant network of thousands of sensor/actuator nodes embedded in the aircraft wings and fuselage which undertakes the following two essential tasks [7]:

T1. Gathering the real-time status of the physical entity (in this case pressure) and detecting the possible failures via data analysis.

T2. Triggering control messages to activate the corresponding actuators located in the vicinity of the failure.

First task is more challenging due to the scale of the system and also because of the difficulty of interconnectivity and timely data processing. In this paper we will address the problem of real-time data acquisition from dense sensor/actuator network.

To aggregate sensor readings in most common sensor networks, nodes are either arranged in different clusters [17-19] or construct a converge-cast tree [20,21] in order to get benefit from parallel data transmission to speedup data acquisition procedure. However, clustering and tree-based convergecasting are not wise solutions for high-spatial resolution sensing and actuating systems because in this kind of systems large number of sensor nodes are deployed within a single broadcast domain—where no concurrent transmission is allowed. Therefore, the time complexity of such data gathering methods is at least in the order of O(m) where m is the number of nodes in a single broadcast domain (SBD).

To degrade the time complexity issue, quantity aggregation methods [8] are proposed in recent researches to make it possible to gather some specific quantities such as MIN or MAX value from a single broadcast domain in a time order of O(log(m)). This recent approach is based on dominance protocols [9], which are used in both CAN bus [10] and WiDom [11]. Beside scalability, having a technique which provides an approximate representation of all sensor readings seems to be more useful in some applications. This representation would enable applications to compute any desired quantity without imposing any further communication costs. To fulfill this need, an algorithm called Basic Interpolation Algorithm (BIA), was proposed [12] to provide an interpolation of the sensor readings in the network. This algorithm is efficient and fast. It is efficient, because it uses the data of all nodes and is fast; because it offers low time complexity. Some improvements on the BIA algorithm were presented in [13, 14] to increase the accuracy and to reduce the average error of the algorithm in dynamic environments. In [15] authors improved the algorithm by embedding the dynamics of the physical phenomenon. However, we identify the limitations in [15] and offer further improvements; especially when dealing with a physical phenomenon with more complicated dynamic behavior.

This paper can be viewed as a continuation of the previously proposed algorithm where the dynamics of the physical phenomenon is embedding on the BIA interpolation algorithm. As our first contribution, we introduce a method to foster the previously proposed technique through implementation of a transformation matrix. Secondly, by considering more realistic changes of the physical entity, we devise a solution to update the transformation matrix by performing a very rapid learning phase.

The remainder of this paper is structured as follow. In Section 2 we first present briefly the required information in the area including the underlying Medium Access Control (MAC) protocol. The problem at stake and a general overview of proposed solution are addressed in Section 3 followed by extensive description of the proposed algorithm in the next section. Detailed performance evaluation and simulation results are presented in Section 5. Finally, in Section 6 we present our conclusions and future works.

2. Background

This section presents a brief summary on basic principles of quantity aggregation in dense sensor network. As stated earlier, in this class of aggregation approach, communication and computation are tightly coupled. In other words, the distributed aggregation algorithm gets benefit from the efficient exploitation of MAC protocol. Therefore, we first explain concisely the underlying MAC protocol and then we describe the technique that is used to estimate the overall image (approximate representation) of the physical entity under investigation.

2.1. Underlying MAC protocol

Dominance/binary countdown protocol [9] is an important family of MAC protocols in the field of quantity aggregation for the following reasoning (i) it offers good properties in terms of providing timeliness support and (ii) it allows simultaneous non-destructive transmission of information in the same broadcast domain. The main feature of this category of MAC protocol is that there is a contention resolution (tournament) phase in which messages

compete for the channel according to their assigned priorities. During tournament phase nodes with pending messages send their priorities bit-by-bit starting with the most significant bit. The medium is designed such that nodes can hear a recessive bit (logical 1), only if no other node sends a dominant bit (logical 0). The node with recessive bit after receiving a dominant bit refrain from the arbitration and it leads to have only one winner at the end of the tournament which is granted the medium to send its data — consider the fact that all priorities are unique.

Thanks to the underlying dominance MAC, various quantity aggregations can be computed easily and executed with low time-complexity if we exploit node´s sensor readings instead of an arbitrary priority number. For instance, consider the case where we are interested in finding the MIN value from a SBD sensor network. By inserting each sensor reading in its priority field, the winner of the tournament phase will simply give us the MIN value of the field in a time order of performing only one tournament execution. Furthermore, by complementing the sensor readings and using them in the priority field, the maximum amount of sensor readings can be found out in the same way. This is the idea which used in the quantity aggregation method of [8] to find the MIN or MAX values among nodes in a SBD. The next subsection reveals the importance of the two mentioned technique for constructing the approximate representation of the physical entity.

2.2. Basic interpolation algorithm

To perform interpolation over an area of dense sensor network, a wisely selected subset of sensor nodes which are so called *control points*, should broadcast their values. The values of control points are used to construct the interpolated graph of the physical entity. In order to find the most effective subset of control points, the technique described in previous subsection for finding the MAX value is used. Basic interpolation algorithm (BIA) [12] is devised such that the node with the maximum interpolation error has the highest priority to sends its value. For a node n_i, interpolation error, e_i, is defined as the absolute difference of its sensor value, v_i, and its interpolated value, $f(x_i, y_i)$:

$$e_i = |s_i - f(x_i, y_i)| \tag{1}$$

where x_i and y_i are space coordinates and the function $f(x_i, y_i)$, approximates sensor reading throughout the area of interest. The next step is to provide an interpolation function which is represented by a set of control points S, and each control point $p_k \in S$ has three attributes of x_k , y_k and v_k. These three attributes demonstrate that the value of interpolated graph on the location (x_k , y_k) is v_k . In each iteration, the node with maximum interpolation error is found and is added to the set of control points, S. The interpolation function $f(x, y)$ — also called weighted-average interpolation (WAI) function — is mathematically defined as follows:

$$f(x,y) = \begin{cases} 0 & if\ S = \emptyset \\ v_i & if\ \exists p_k \in S : x_i = x \wedge y_i = y \\ \dfrac{\sum_{k \in S} v_k \cdot w_k(x,y)}{\sum_{k \in S} w_k(x,y)} & otherwise \end{cases} \tag{2}$$

where $w_k(x,y)$ is a weight which is reversely proportional to the distance between the k^{th} control point and the point in the location (x,y) and is given by:

Algorithm 1. BIA algorithm

1: $S \leftarrow \emptyset$
2: **for** $i \leftarrow$ 1to k **do**
3: calculated the interpolation value according to Equation (2)
4: calculate e_i
5: select the sensor node n_i with maximum e_i
6: add the node n_i to the set of control points S
7: **end for**

$$w_k(x, y) = \frac{1}{(x_k - x)^2 + (y_k - y)^2} \quad (3)$$

Interpolation algorithm produces a signal of physical quantity and improves itself iteratively. Initially the interpolation is zero on each location (this is represented by setting S to the empty set). Then, each sensor node evaluates the interpolation with respect to its location and compares it with its sensor reading. The error is calculated by Equation (1) and the node with the maximum error is granted the medium for transmitting its location and sensor reading. At the end this information is added to the set S. In BIA, this loop is repeated k times (where the value of k is selected by the designer). Algorithm 1 shows the pseudo code of basic algorithm. It must be mentioned that this procedure is executing by all nodes so that all of them have almost same approximation of the interpolation graph.

3. Problem statement and general overview

We next summarize the specific sensing needs for more delicate applications like avionics that needs an accurate and fast representation of the physical world. Then we discuss the problem of such sensing system followed by a description of our general approach.

3.1. Sensing needs and target problem

Various physical quantities can be monitored by sensor networks; pressure, temperature and humidity (weather forecasting and agriculture), acceleration/vibration (structural health monitoring and biomedical sensor networks), gas density (petrochemical companies and refineries), light intensity, flow (avian technology and chemical industries) and so on. We refer to the distribution function of a physical quantity across the monitoring field as *signal*. Based on the nature of monitored parameter and its environment, the signal might be smooth or rough and may even change fast or slow. Temperature and humidity of the weather are two examples of slow changing parameters and their distribution function across a field can be categorized as a smooth signal. Those may not change considerably in hours. On the contrary, some other physical quantities such as airflow on the wings and fuselage of an airplane are very fast changing physical quantities.

In most applications, it is desired to have a fresh approximate image of the physical quantity within every certain time interval which is called *sampling interval* (T_S). Using Interpolation techniques, we try to provide a fast and rather accurate image for monitoring and controlling the systems of interest.

The previous interpolation algorithm, BIA, assumes that the physical quantity remains unchanged while the interpolation algorithm is executing. In practice, however, this is not the case in some applications. A wise solution to tackle this problem— as addressed in differential interpolation algorithm (DIA) [15] — is to embed the physical change pattern into the interpolation algorithm. The pseudo-code of this technique is shown in Algorithm 2. DIA approach helps to provide more accurate representation of the physical signal, since the change pattern keeps updating the previous selected control point continuously.

However, DIA is still unable to provide a system-dependent interpolation algorithm since the proper change pattern cannot be automatically detected by the system and needs to be assigned by the application designer. Consider the case when we need to run the algorithm for a long monitoring time and in a meanwhile the physical change pattern alters. It is obvious that the pre-defined change pattern is not valid anymore and it can cause a significant diverge between the interpolation represented by the algorithm and the physical quantity being measured. This is the main problem with DIA approach since no possible solution is offered to update the change pattern after the first time setting.

3.2. General overview of proposed solution

A more educated solution is to have a *learning phase* at the beginning of the algorithm. Through executing this phase, a matrix that models the changing pattern of the physical signal is computed; we call this matrix *transformation matrix* (T_{matrix}). In fact, the physical world may experience unexpected changes while running the algorithm. Finding transformation matrix in runtime fosters the algorithm to react quickly to sudden changes.

Performing the learning phase would though impose some costs. To alleviate the costs, instead of computing the T_{matrix} repeatedly at the beginning of each sampling interval, we introduce another time interval by which the algorithm performs a self assessment; it is known as *assessment interval* (T_A). Then according to the obtained results from the assessment part, the algorithm decides whether to re-compute a new T_{matrix} or keep the old one for

Algorithm 2. DIA algorithm

Table 1. Notation used in LIA.

T_S	Sampling interval
T_A	Assessment interval
E_t	Acceptable error threshold
k	Number of control points
T_{matrix}	Transformation matrix
e_i	Difference between computed interpolation graph and real value
D_{int}	Duration of performing one complete round of interpolation
Slot	Duration of adding one iteration of interpolation to add a control point to the set *S*
Trnmnt	Duration of performing a tournament
Tx	Time to transmit a packet
Comp	Time to compute the interpolation

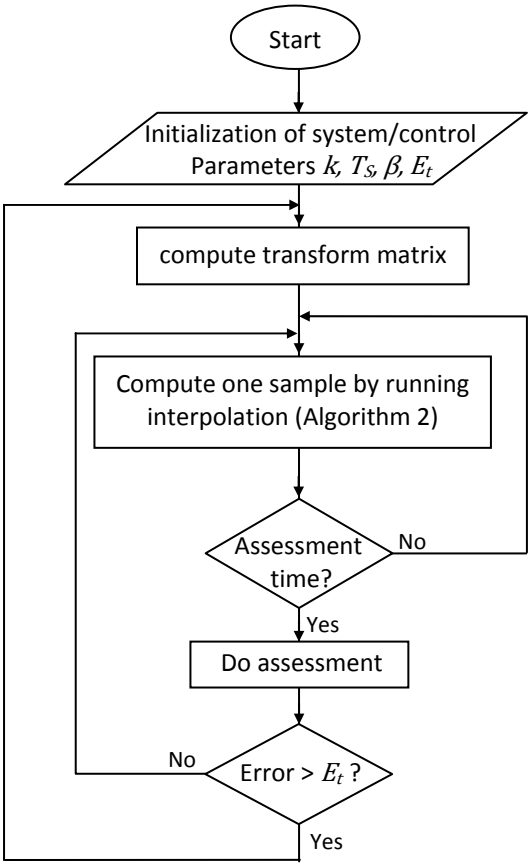

Figure 2. Flow chart of linear interpolation algorithm.

computing the next samples. Figure 2 illustrates the flow chart of our new approach which is called linear interpolation algorithm (LIA). All related notations which used in the LIA algorithm is presented in Table 1. Note that in the execution of Algorithm 2 the change pattern is achieved according to the computed T_{matrix} in the previous step. The next chapter provides more detailed information on the new LIA algorithm.

4. Linear interpolation algorithm

The main purpose of LIA is to offer a system design that shifts most of complexity away from the application designer toward the underlying system. This leads to design a system-dependent interpolation algorithm which copes better with rapid changes of physical signal. To do so, the following steps need to be accomplished successively: (i) computing T_{matrix}, (ii) performing the interpolation algorithm with the knowledge provided by the T_{matrix} and (iii) updating and refining the T_{matrix} after a defined period of time. The second step is already discussed in [15]. Before explaining the remaining two steps, we briefly explain how to use linear model to perform interpolation.

As it is shown in the Algorithm 2 (line 8) the change pattern is needed to update the three attributes of each control point in the set *S* at the beginning of a new *slot*—see Table 1. This update is formulated as follows:

$$\begin{cases} \text{xnew}_i \leftarrow T_{1,1} \times x_i + T_{1,2} \times y_i + T_{1,3} \times v_i + T_{1,4} \\ \text{ynew}_i \leftarrow T_{2,1} \times x_i + T_{2,2} \times y_i + T_{2,3} \times v_i + T_{2,4} \\ \text{vnew}_i \leftarrow T_{3,1} \times x_i + T_{3,2} \times y_i + T_{3,3} \times v_i + T_{3,4} \end{cases} \quad (4)$$

where (x_i, y_i, v_i) are the three attributes of the control point i in slot t and ($xnew_i$, $ynew_i$, $vnew_i$) are the updated attribute of control point i in slot $t+1$. The $T_{j,k}$ is known as the parameter or element of T_{matrix}. By defining the elements of T_{matrix} it would be possible to find any linear change pattern on the physical signal such as scaling, translating and rotating. The following subsection describes the procedure of finding T_{matrix}.

4.1. Defining the parameters of T_{matrix}

One approach for defining the parameters of the linear model is to observe the trend of signal change and then solve a system of equations based on this observation. We assume that there is a monotonous change in a physical quantity which implies that the overall shape of the signal remains unchanged during the observation time. It can be claimed that if the interpolation algorithm executes in different points in time, the same points in the physical signal will be selected as control points. For example, if the signal translates in a specific direction for amount of α, new control point will be the previous control point which its location is translated in the same direction for amount of α. This fact is the basic building block for our proposal. If we can track the change of one special point on the physical signal in each slot we will be able to find all the elements of T_{matrix}.

To this end, first we need to run the interpolation algorithm five times with the duration of one slot. Hence, we are going to have five sets of control points with only one member. Then, to specify the change pattern, we should resolve the relation between these five control points. The obtained model of changing pattern will be then applied in the interpolation algorithm for updating the value and coordinates of control points. The model parameters, $T_{j,k}$s, in Equation (4) are defined by solving the following equations:

$$\begin{bmatrix} x_1 & y_1 & v_1 & 1 \\ x_2 & y_2 & v_2 & 1 \\ x_3 & y_3 & v_3 & 1 \\ x_4 & y_4 & v_4 & 1 \end{bmatrix} \times \begin{bmatrix} T_{1,1} \\ T_{1,2} \\ T_{1,3} \\ T_{1,4} \end{bmatrix} = \begin{bmatrix} x_2 \\ x_3 \\ x_4 \\ x_5 \end{bmatrix} \quad (5)$$

$$\begin{bmatrix} x_1 & y_1 & v_1 & 1 \\ x_2 & y_2 & v_2 & 1 \\ x_3 & y_3 & v_3 & 1 \\ x_4 & y_4 & v_4 & 1 \end{bmatrix} \times \begin{bmatrix} T_{2,1} \\ T_{2,2} \\ T_{2,3} \\ T_{2,4} \end{bmatrix} = \begin{bmatrix} y_2 \\ y_3 \\ y_4 \\ y_5 \end{bmatrix} \quad (6)$$

$$\begin{bmatrix} x_1 & y_1 & v_1 & 1 \\ x_2 & y_2 & v_2 & 1 \\ x_3 & y_3 & v_3 & 1 \\ x_4 & y_4 & v_4 & 1 \end{bmatrix} \times \begin{bmatrix} T_{3,1} \\ T_{3,2} \\ T_{3,3} \\ T_{3,4} \end{bmatrix} = \begin{bmatrix} v_2 \\ v_3 \\ v_4 \\ v_5 \end{bmatrix} \quad (7)$$

Three transform matrixes for control points' locations x_i, y_i and their values v_i will be achieved by solving the equations above. Each node should perform this computation individually. This computation imposes more complexity to our algorithm. It is noteworthy however, since this additional computation promotes the new algorithm

to be more generic compared to DIA algorithm. In § 5.1 we show that applying the linear model in the interpolation procedure results in smaller average error compared to the previous algorithms.

4.2. Runtime control and feedback

Another important issue after computing the elements of T_{matrix} is the ability to update the values of T_{matrix} so that the algorithm could be able to recover after occurrence of a sudden change in the existing change model. Applying a control feedback from the sensor field seems to be helpful. To do this, we employ a self assessment test that take advantage of time division multiple access (TDMA) MAC design [16] to collect the feedback messages. If the outcome of the test is satisfactory then the system uses the current T_{matrix} for taking future samples, otherwise it requests for performing another round of learning phase—see Algorithm 3. A specified number (P) of sensor nodes broadcast their observed error level within their defined TDMA time slot. We refer to this set of sensor nodes as *observers*. The selection of these observers is based on their location in the monitoring field. The observers should be located in an area which is called *observing window*. It is obvious that the observing window is a subset of the monitoring field. We introduce two different strategies for selecting the observer nodes: (i) static positioning and (ii) dynamic positioning.

Static positioning. In this approach the observing window covers the whole monitoring field. First, the nodes which are located at the vertices of the monitoring field are selected. The last node is selected from the center of the field at the intersection point of the diagonals. The idea is to get a general perspective of the error level across the monitoring area.

Dynamic positioning. The purpose of developing this method is to confine the location of observers to the area close to the peak of physical signal. The location of observer can be achieved by taking the coordinates of the first control point (x_1, y_1) and then choosing four sensor nodes that are located on $(x_1 \pm R , y_1 \pm R)$ where R is the radius of observing window. The reason of choosing the first control point (x_1, y_1), follows from the logic behind the BIA algorithm. As discussed in § 2.2, in BIA the node with maximum interpolation error, e_i, is selected as the first

Algorithm 3. Self assessment procedure that runs on node n_i

Input: number of observers P, threshold level E_t
Output: Re-compute control command

```
1: need_to_recompute ← 0
2: n_i constructs the set of P number of observer and computes the respected time slot
3: for j ← 1 to P do
4:     if my.id = observer_j.id then
5:         calculate e_j and broadcast a packet
6:     else
7:         wait_receive_packet() // blocking code
8:         retrieve error value from the packet and save it
9:     end if
10: end for
11: take average of all received e_i and store in the Ave_e_i
12: if Ave_e_i > E_t then
13:     need_to_recompute ← 1
14: end if
15: return need_to_recompute
```

control point. Since at the beginning of the algorithm the interpolation graph has been set to zero, the first selected point is supposed to be the peak of the physical signal.

The feedback control imposes some overhead on the system by applying extra packet transmission. To trade off between the high accuracy and low overhead, we have defined another parameter known as *assessment interval* (A_i). This interval represents the period by which the self assessment test is activated. An appropriate choice for A_i is given by:

$$T_A = \beta \times T_S \tag{8}$$

where $\beta \in \{N \geq 1\}$ is the *assessment factor* that can be set by the application designer according to the required accuracy level. Intuitively, the Smaller the assessment frequency, the more accurate representation of the signal will be achieved.

5. Evaluation

The LIA algorithm embeds a transformation matrix which has a huge impact on the approximate representation of physical signal. In this line, we investigate the efficiency and accuracy of transformation matrix under different linear signal changes and various network densities. Then we show the performance of LIA algorithm in long-term observation of a physical entity.

5.1. The efficiency of transformation matrix

As stated earlier in § 4.1, transformation matrix is built in the learning phase and stands as an acting pattern of LIA algorithm. The procedure of computing T_{matrix} requires five slots and then immediately afterward the LIA algorithm starts executing using this matrix. The approximate representation of the physical signal — let us call it a *sample*— can be obtained by executing the interpolation algorithm for different number of control points. Note that only one control point can be selected during each interpolation round. We have investigated three major types of linear signal changes, i.e. scaling, incrementing and translation. Figure 3 compares the BIA and LIA algorithm for different number of control points under different signal change models. We did not include DIA algorithm since it cannot support all the changing models we considered. One can intuitively understand that DIA and LIA algorithms can provide same results for scaling and incremental change models since they are embedding the physical change pattern to the interpolation mechanism. However, DIA algorithm is not able to provide a proper change pattern for the case that the physical signal is translating. Hence it acts similar to the BIA algorithm and leads to an incorrect result.

Simulation results in Figure 3 show a remarkable improvement in average relative error for all types of signal changes under the LIA algorithm. In these cases, BIA presents poor results as the relative error keeps rising by increasing number of control points. It is important to mention that entering more control points requires longer time for running the algorithm which consequently results in facing higher change level in the physical signal. In BIA algorithm, the value of selected control points in the set *S* remains unchanged. This leads to major deviation of the

result from the correct values. In BIA algorithm, the trend of average error in scaling and incrementing scenarios increases steadily by adding more control points, but there is a period of slight reduction in case of translating signal change. The reason for this reduction is due to the movement of the physical signal out of monitoring field region. In fact, this situation does not happen in the real-world scenario as the physical entity is not limited to a specific space.

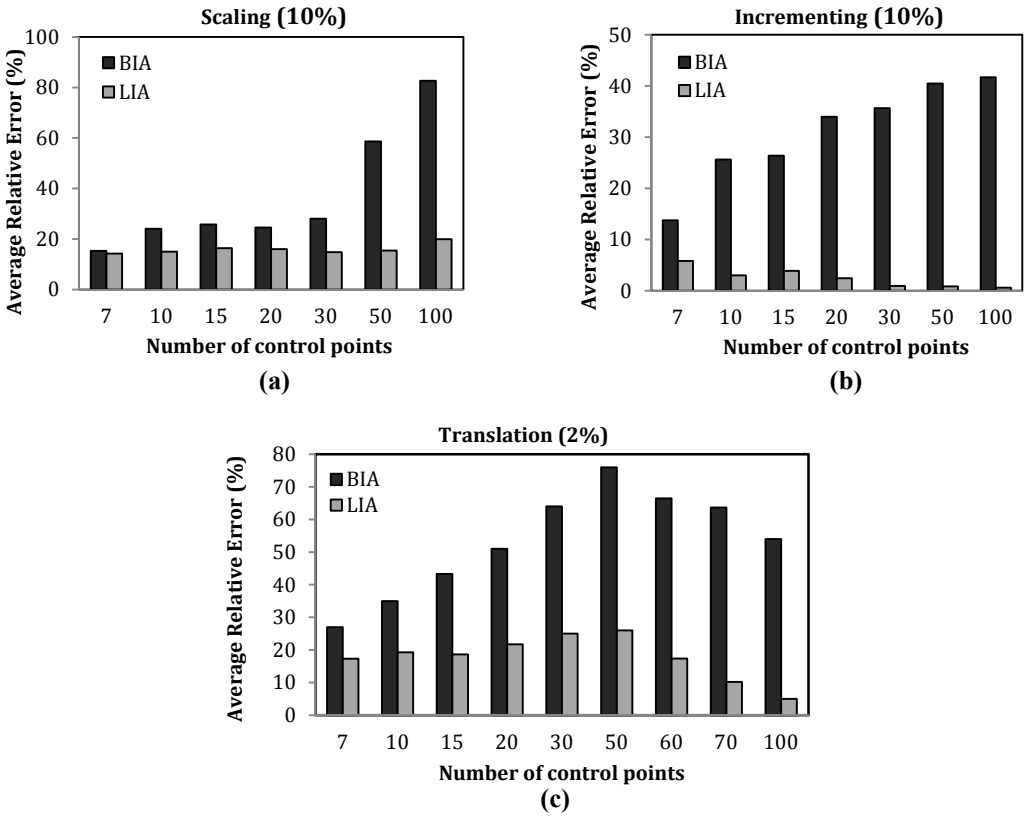

Figure 3. Average relative error for three scenarios of (a) scaling, (b) incrementing and (c) translation.

Generally, studying the intrinsic characteristic of each type of aforementioned signal change, we observe that incrementing causes smoother change compared to other change models since it simply adds a value to the physical signal. Scaling has stronger effect as it multiplies a value to the physical signal. Thus by having a same rate of change for incrementing and scaling change models, scaling shows more rough change of physical signal than incrementing. This fact reveals the reason why LIA is performing much better under incrementing signal change than scaling scenario. In fact LIA is using WAI function ―see Equation 2― which is suitable for smooth signal according to [12]. Applying scaling change model turns the signal to a non-smooth shape which decreases the efficiency of LIA algorithm. Comparing Figure 3(a) and 3(b) we notice that LIA provides a stable average relative error with the upper bound of 20% running under scaling signal change model while it provides constantly decreasing error when facing incrementing change model.

LIA offers an approximate representation of physical signal with the maximum of 30% error rate when dealing with translation change model―see Figure 3(c). Although translation does not change the smoothness of the signal,

but it affects all three attributes of the control points i.e. (x_i, y_i) coordinates and v_i value. The change model of translation has stronger impact on the physical signal compared to scaling and incrementing as all the three attributes of control points are changing in each slot whereas, in scaling and incrementing the change occurs only in the value of control points. As stated before there is a decreasing error trend starting from interpolation round 50 which is due to signal movement toward out of the monitoring field.

A small error in computing the parameters of T_{matrix} leads to a very poor result which is more severe as we increase the interpolation round. This computing error is partly related to the network density. Recall that for defining T_{matrix} we need to track the point with highest interpolation error e_i for five iterations and then compute the parameter of T_{matrix} according to the attributes of the selected points. A correct set of data is achieved by tracking same point of the physical signal as time progresses. In low density network, it is highly probable that the place where the tracking point locates does not cover by a sensor node. Thus another point of the signal with highest e_i which is located within the coverage area of a sensor will be selected. Misplacing correct tracking point of the physical signal yields to an inaccurate output.

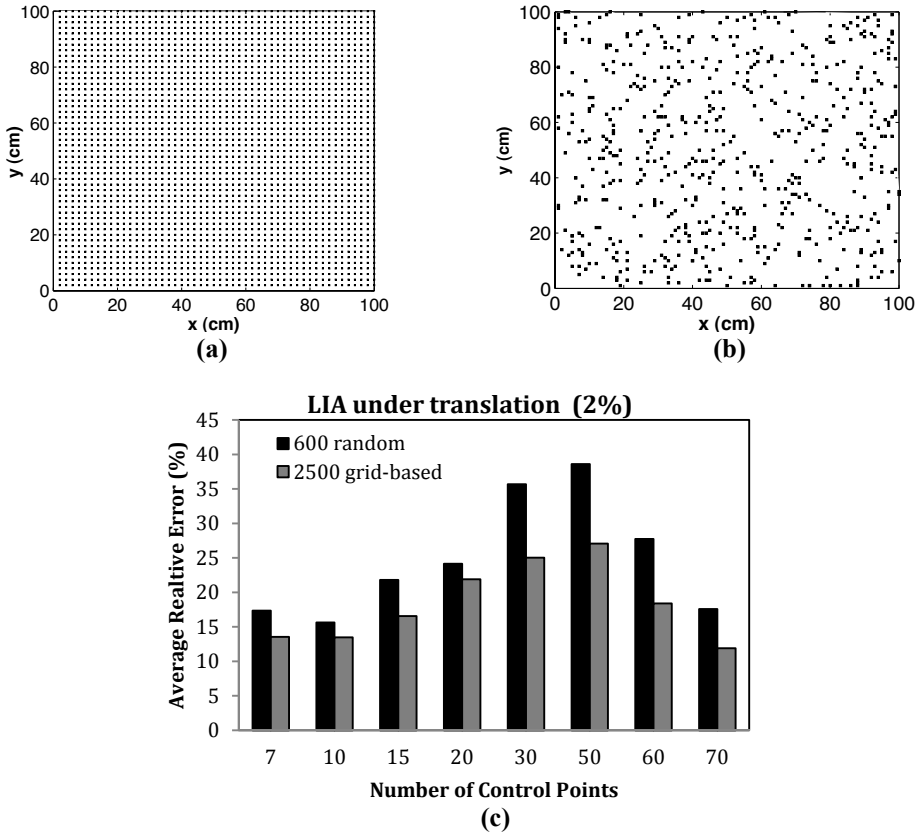

Figure 4. Effect of network density on LIA performance (a) 600 nodes randomly distributed (b) 2500 nodes with grid-based deployment (c) average relative error of LIA under translation (2%).

Figure 4 shows the impact of network density on the performance of LIA algorithm. We have run a scenario for a network with dense and grid-based node deployment —see Figure 4(a), and repeated the same scenario for a

Table 2. Scenario description.

Sampling interval, T_S	40
Assessment factor, β	6
Number of control points, k	7, 10, 15, 20
Initial change model	Incrementing with rate of 5%
Secondary change model	Decrementing with rate of 5%
Time slot of change occurrence	170
Acceptable error threshold, E_t	5%

network with lower density where the nodes are deployed randomly across the observation area —see Figure 4(b). In both scenarios we consider of having translation change model with the rate of 2% toward the northeast direction. The result shows that LIA outperforms much better when running on top of a dense network. The difference between these two implementations comes from the incorrect computation of T_{matrix}. Due to error accumulation, this difference varies up to 10% for bigger number of control points. The correct values of T_{matrix} by considering 2% rate of change have to be $T_1 = [1, 0, 0, 2]$, $T_2 = [0, 1, 0, 2]$ and $T_3 = [0, 0, 1, 0]$ — see Equation 4 — that is exactly what we obtained by running LIA on top of dense network. In lower density network, translation parameters, $T_{1,4}$ and $T_{2,4}$, differ in the range of 1.67 to 2.67 that leads to higher average error for longer execution of the algorithm.

5.2. LIA with feedback control

In many real-world applications, such as the one considered in our paper, taking continuous samples is an essential requirement. In this subsection, we investigate the performance of LIA in long-term simulation when it involves observing the physical signal within very short time interval. To do so, the application designer should set the desired sample interval, $T_{s,}$ with respect to the time criticality feature of the corresponding application. Additionally, for updating the physical signal change pattern it is needed to set the assessment factor, β, which shows the frequency of assessment interval, T_A. In order to evaluate LIA we consider four scenarios with the assumptions shown in Table 2. Each scenario has been performed with specific number of control points for building one sample of physical signal. In all scenarios it is assumed that the physical signal is initially incrementing with the rate of 5% in each time slot and at time slot 170 the change pattern alters from incrementing to decrementing with the same rate of change. Considering that the length of each time slot is about 10*ms* we set the sample interval to 400*ms*. We chose this value in order to have at most 50% active duty cycle in the worst case when each sample needs 20 time slots to be prepared. Five sensor nodes as observers assess the accuracy of T_{matrix} every six-sample period, i.e. β=6. We have also used static positioning technique for covering the whole monitoring field.

The algorithm starts by executing the learning phase, and immediately it builds one sample of physical signal — sample 0. Afterwards it proceeds building the first sample and keep doing so until the sixth one. At the end of building the sixth sample, observers send their observed error values within their assigned TDMA slot. Assuming that each TDMA-slot takes 8*ms*, we need to assign 4 time slots for receiving the observed error. Figure 5 illustrates the average error of the samples taken by LIA algorithm for four different scenarios. In all scenarios there is an increasing trend in the average error for those samples taken after time slot 170. The reason for this increase is due to the variation that happens in the signal change model at time slot 170. Note that for those samples taken after time slot 170, the average error is higher for the scenarios that use more control points. Because in these scenarios,

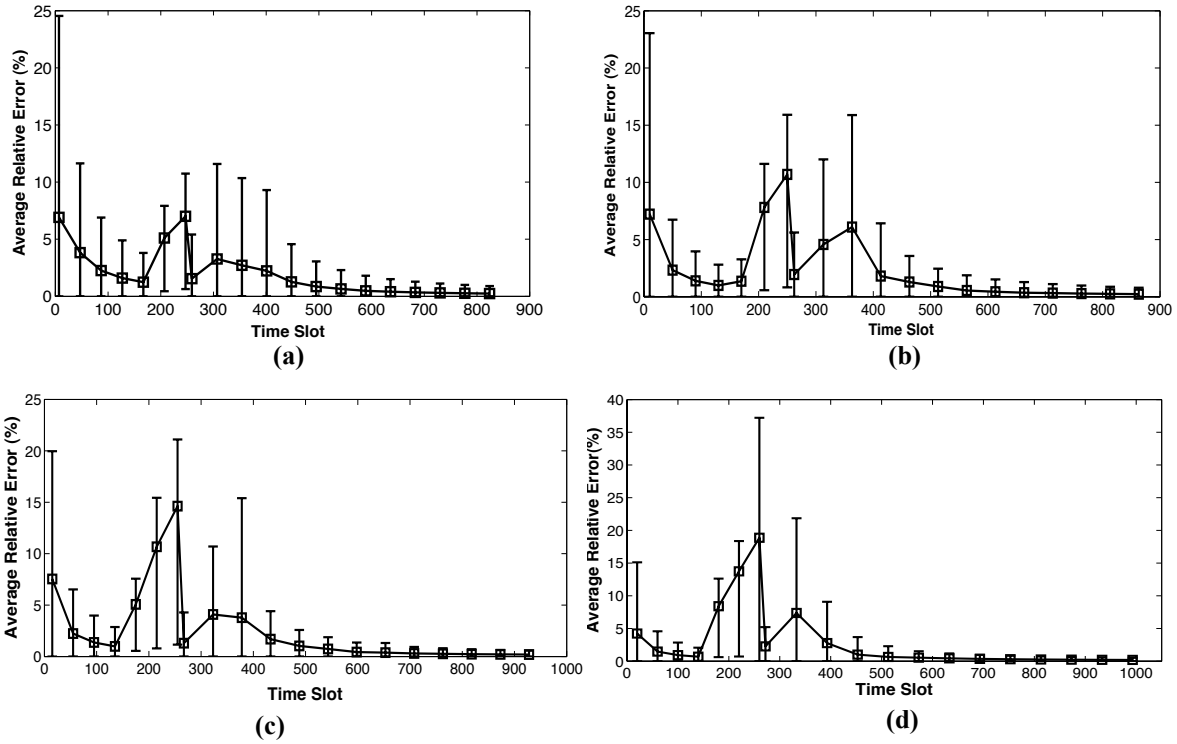

Figure 5. Long-term simulation of LIA with: (a) 7 control points (b) 10 control points (c) 15 control points (d) 20 control points.

the initiatory control points are updating their attributes with an inaccurate T_{matrix} and hence diverging more from their corresponding real values.

Table 3 lists the observed average relative error and the real average relative error for the same time slot. As it is shown the observers can provide a fair assessment of the algorithm and relay proper command for re-computing the T_{matrix}. The acceptable error threshold for re-executing the learning phase is set to 5%. In all scenarios, the first sample taken after assessment has small average error that allows the algorithm to proceed using the latest calculated T_{matrix} for the next samples. As it is shown in Figure 5, there is a slight increase for the first two samples that has taken after first assessment. This situation can be described by considering the decreasing movement of physical signal toward zero. At some time slots the difference between the interpolated and original signal is comparable with the original signal which results in higher average relative error. The average error observed in the second round of assessment is below the acceptable threshold level, hence LIA proceed to use the current T_{matrix}. The slope of improvement is much higher for those scenarios with higher number of control points.

Table 3. Observed and Real average relative error (%).

scenario	1st assessed avg. error at t_1	Real avg. error at t_1	2nd assessed avg. error at t_2	Real avg. error at t_2
1st scenario	9.929	7.007	0.28	0.89
2nd scenario	12.08	10.69	0.25	0.87
3rd scenario	14.62	13.64	0.15	0.187
4th scenario	19.14	18.86	0.102	0.185

6. Conclusion

This paper addresses the design and evaluation of a linear approach of data gathering which is used in dense wireless sensor network application. Avionic is an example of this kind of application that gains from dense deployment of sensors. LIA provides a solution to find an approximate representation of the pressure across the body of an aircraft. Furthermore, it achieves high accuracy and responsiveness by employing very fast learning phase which helps in finding the change pattern of the physical signal. This phase can be re-executed according to the results obtained from the assessment procedure. Simulation results show clearly that by using LIA algorithm, the average relative error can be reduced extensively by as much as 60% compared to the preliminary interpolation algorithms.

Future works. Our work is at an initial phase and should be future analyzed in three particular directions. First, we need to find a strategy to automatically select the optimal number of control points in building a sample. Second, the challenges of having non-linear change in the signal change pattern should be investigated. Third, a more realistic scenario including the previous aspects should be moved from single to multiple broadcast domains.

7. References:

[1] D. Estrin, D. Culler, K. Pister, and G. Sukhatme; "Connecting the physical world with pervasive networks," IEEE Pervasive Computing, pp. 59-69, 2002.

[2] J. A. Stankovic, I. Lee, A. Mok, and R. Rajkumar; "Opportunities and obligations for physical computing systems," IEEE Computer, 38(11), pp. 23-31, 2005.

[3] J. Caffrey, R. Govindan, E. Johnson, B. Krishnamachari, S. Masri, G. Sukhatme, K. Chintalapudi, K. Dantu, S. Rangwala, A. Sridharan, N. Xu, and M. Zuniga; "Networked Sensing for Structural Health Monitoring," In: Proceedings of the 4th International Workshop on Structural Control, Columbia University, NY, June 2004.

[4] J. Reneaux, "Overview on Drag Reduction Technologies for Civil Transport Aircraft," European Congress on Computational Methods in Applied Sciences and Engineering (ECCOMAS04), Jyvskyl, July 2004.

[5] T. Washburn, "Airframe Drag/Weight Reduction Technologies", Green Aviation Summit - Fuel Burn Reduction, NASA Ames Research Centre, September 2010.

[6] S. G. Anders, W. L. Sellers, and A. E. Washburn; "Active Flow Control Activities at NASA Langley", 2nd AIAA Flow Control Conference, June 28 – July 1, 2004.

[7] K. Bür, P. Omiyi, Y. Yang, "Wireless sensor and actuator networks: enabling the nervous system of the active aircraft", IEEE Communications Magazine, Vol. 48, No. 7, pp. 118-125, 2010.

[8] N. Pereira, R. Gomes, B. Andersson, and E. Tovar. "Efficient aggregate computations in large-scale dense WSN," In 15th IEEE Real-Time and Embedded Technology and Applications Symposium, pp. 317-326, San Francisco, California, USA, 2009.

[9] A.K. Mok and S. A. Ward, "Distributed Broadcast Channel Access," Computer Networks, Vol. 3, November 1979.

[10] Bosch GmbH, Stuttgart, Germany. CAN Specification, ver. 2.0, 1991.

[11] B. Andersson, N. Pereira, and E. Tovar, "Widom: A dominance protocol for wireless medium access," IEEE Transactions on Industrial Informatics, vol. 3(2), pp. 120-130, May 2007.

[12] B. Andersson, N. Pereira, W. Elmenreich, E. Tovar, F. Pacheco, and N. Cruz,"A scalable and efficient approach to obtain measurements in CAN-based control systems," In IEEE Trans. Industrial Informatics, Vol 4(2), pp. 80-91, 2008.

[13] E. Tovar, B. Andersson, N. Pereira, M. Alves, S. Prabh and F. Pacheco, "Highly Scalable Aggregate Computations in Cyber-Physical Systems: Physical Environment Meets Communication Protocols," Proceedings of the 7th International Workshop on Real-Time Networks, Prague, Czech Republic, July 1, 2008.

[14] B. Andersson, N. Pereira, E. Tovar, R. Gomes, "Using a prioritized medium access control protocol for incrementally obtaining an interpolation of sensor readings," Seventh Workshop on Intelligent solutions in Embedded Systems, pp. 29 – 36, Ancona, June 2009.

[15] A. Ehyaei, E. Tovar, N. Pereira and B. Andersson, "Scalable Data Acquisition for Densely Instrumented Cyber-Physical Systems", Proceedings of the ACM/IEEE Second International Conference on Cyber-Physical Systems, pp. 174-183, 2011.

[16] R. Nelson and L. Kleinrock, "Spatial TDMA: A collision free multihop packet channel access protocol," IEEETransactions on Communications, vol. 33, no. 9, pp. 934-944, September 1985.

[17] W. Heinzelman, A. Chandrakasan, and H. Balakrishnan, "An Application-Specific Protocol Architecture for Wireless Microsensor Networks," IEEE Trans. Wireless Comm., vol. 1, pp. 660-670,October 2002.

[18] S. Lindsey and C. Raghavendra, "PEGASIS: Power-Efficient Gathering in Sensor Information Systems," Proc. IEEE Aerospace Conf., vol. 3, pp. 1125-1130, March 2002.

[19] B.J. Culpepper, L. Dung, and M. Moh, "Design and Analysis of Hybrid Indirect Transmissions (HIT) for Data Gathering in Wireless Micro Sensor Networks," ACM SIGMOBILE Mobile Computing and Comm. Rev., vol. 8, pp. 61-83, January 2004.

[20]M. Ding, X. Cheng, and G. Xue, "Aggregation Tree Construction in Sensor Networks," Proc. 58th IEEE Vehicular Technology Conf.,vol. 4, pp. 2168-2172, October 2003.

[21] W. Zhang and G. Cao, "DCTC: Dynamic Convoy Tree-Based Collaboration for Target Tracking in Sensor Networks," IEEE Trans. Wireless Comm., vol. 3, pp. 1689-1701, September 2004.