Collaborative State Estimation and Actuator Scheduling for Cyber-Physical Systems under Random Multiple Events

Lei Mo, Angeliki Kritikakou, Xianghui Cao

Collaborative State Estimation and Actuator Scheduling for Cyber-Physical Systems under Random Multiple Events

Lei Mo[1], Angeliki Kritikakou[1], and Xianghui Cao[2]

[1] Univ Rennes, INRIA, IRISA, CNRS, 35042 Rennes, France,
e-mail: lei.mo@inria.fr, angeliki.kritikakou@irisa.fr
[2] School of Automation, Southeast University, 210096 Nanjing, China,
e-mail: xhcao@seu.edu.cn

Abstract. The design of fast and effective coordination among sensors and actuators in Cyber-Physical Systems (CPS) is a fundamental, but challenging issue, especially when the system model is a priori unknown and multiple random events can simultaneously occur. We propose a novel collaborative state estimation and actuator scheduling algorithm with two phases. In the first phase, we propose a Gaussian Mixture Model (GMM)-based method using the random event physical field distribution to estimate the locations and the states of events. In the second phase, based on the number of identified events and the number of available actuators, we study two actuator scheduling scenarios and formulate them as Integer Linear Programming (ILP) problems with the objective to minimize the actuation delay. We validate and demonstrate the performance of the proposed scheme through both simulations and physical experiments for a home temperature control application.

Keywords: Cyber-physical systems, Gaussian mixture model, event estimation, actuator scheduling

1 Introduction

Cyber-Physical Systems (CPS) bridge the cyber world to the physical world through a network of sensors and actuators. Sensors measure the environment, while actuators control the environment based on sensors' information. Wireless sensing and control facilitate the design of mobile systems, enabling closed-loop control of mobile devices, such as automated guided vehicles, mobile robots, and unmanned aerial vehicles. CPS are becoming a promising technology for a wide range of application domains, such as smart building [3], intelligent transportation [7], power grid [6], and industrial control [4].

The sensors are low-cost devices and they are usually largely deployed in the environment. The sensors usually have limited capabilities in terms of power, communication and computation. Each sensor has a fixed sensing range and its position is usually static. The actuators, based on the state estimation deriving from the sensors' measurements, apply corresponding actions in order to control

the environment. Therefore, the actuators have higher capabilities than the sensors. The measured data of the sensors provide only partial information of the physical world, especially when the Region Of Interest (ROI) is large. Therefore, it is unreliable to make actuator decisions based only on a small number of sensors' measurements. In addition, the measurements can be correlated and each measurement may reflect the overlapped effect of multiple events, making difficult to distinguish and localize the different events in the ROI.

Most of the existing works on state estimation and actuator scheduling problems usually assume that the system model is fixed and given in advance, i.e. the position of the events is a priori known [13]. However, when the events occur randomly, the system model cannot be known. It is time-varying and it is determined by the characteristics of events, e.g., the distances among the events and the distances among the sensors and the events. An example of such an unknown system model is the detection and extinction of fires in a given area. There is no priori knowledge when and where the fires will occur, i.e. fires are random occurring events. In case a fire breaks out somewhere, the actuators should be scheduled to the relevant area to handle this event as soon as possible. For the single-event case under unknown system model, state estimation can be performed by applying the Maximum Likelihood Estimation (MLE) method [10]. However, this method is hard to be extended to the multi-event case, since multiple events can be seen as a linear combination of single-events and the weighs of the linear combination cannot be obtained from the MLE. For the multi-event case, the physical field caused by the events forms a surface. Hence, the data fitting methods [11] can be used to estimate the surface function. However, these methods usually require to determine in advance the function order limiting the applicability of these methods.

In this paper, we focus on systems with unknown model where multiple random events can occur simultaneously. We address the challenges of *1) How to identify, localize and evaluate the occurred events, and 2) How to schedule appropriate actuators to perform fast and effective actions against these events.* Both the event processing delay and the actuation delay should be low, otherwise an event may grow to an urgent level before a certain action takes place. We propose an novel state estimation and actuator scheduling scheme. Our main contributions are:

1. A Gaussian Mixture Model (GMM)-based method to identify, localize and estimate the states of possible multiple events. The proposed method is based on the characteristics of physical field of random events, e.g., the physical filed of a random event, such as fire, follows Gaussian distribution [10]. Then, a virtual sampling method is proposed to improve the estimation accuracy.
2. We consider two possible scenarios based on the number of actuators and the number of identified events and we formulate them as Integer Linear Programming (ILP) problems. The first scenario considers that the number of actuators is no less than the number of events. Hence, the actuator, which resides nearest to the events, is scheduled to act. The second scenario considers the case when the number of actuators is less than the number of events.

Thus, the available actuators should be scheduled to handle the events with high priorities. We propose an approach to predict the change of priority level based on historical date and we propose a predicted-priority-based method to schedule the actuators.
3. We evaluate the performance of our approach by both simulations and experiments based on a physical testbed.

The remainder of this paper is organized as follows. The proposed event estimation and actuator scheduling scheme is described in Section 2 and Section 3, respectively. Simulations and experiments are conducted in Section 4. Finally, Section 5 concludes this paper.

2 Event Estimation Scheme

We consider a CPS for environmental monitoring and control of a spatial area called Region Of Interest (ROI). We are interested in utilizing l sensors $S_1, \ldots, S_l$ and m actuators $A_1, \ldots, A_m$ to monitor and control the state of n Points Of Interest (POIs) $\boldsymbol{t} = [t_1, \ldots, t_n]'$, e.g., temperature and illumination, within a given threshold $\boldsymbol{t}^* = [t_1^*, \ldots, t_n^*]'$. We consider that the location of the POIs is not fixed and assume that an event e_i occurs at POI_i, if $t_i \geq t_i^*$. Since the information exchange among the sensors and the actuators is carried out by the discrete wireless packets [8], we have the following measurement model:

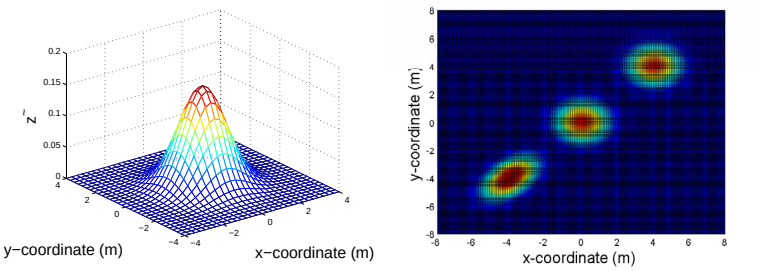
$$\boldsymbol{z}(k) = \boldsymbol{C}\boldsymbol{t}(k) + \boldsymbol{\nu}(k), \tag{1}$$

where $\boldsymbol{z}(k) = [z_1(k), \ldots, z_l(k)]'$, $z_i(k)$ is the measurement of the sensor S_i at the k^{th} step, $\boldsymbol{C}$ and $\boldsymbol{\nu}$ are the measurement matrix and the noise (Gaussian, white and zero-mean) vector with appropriate dimensions, respectively.

To deal with the events in the ROI, two key issues should be addressed: 1) How to estimate the state and the location of the POIs based on the sensor measurement $\boldsymbol{z}(k)$, and 2) How to schedule the actuators to handle the occurring events. (1) shows that the measurement matrix $\boldsymbol{C}$ plays an important role on the estimation of the system states $\boldsymbol{t}(k)$. Usually, the matrix $\boldsymbol{C}$ is determined by the network structure, such as the sensing range of the sensors and the distances between the sensors and the POIs. Most of the existing works are based on the assumption that the matrix $\boldsymbol{C}$ is fixed and given in advance [13]. In contrast to existing approaches, we focus on random events, where the matrix $\boldsymbol{C}$ is unknown.

2.1 Gaussian-based Physical Field Distribution

We focus on monitor and control of physical variables, whose physical field can be described by Gaussian models, such as temperature and illumination [5, 10, 12]. Denote (x_i, y_i) as the location of the sensor S_i, and $\tilde{z}_i$ as the normalized value of sensor S_i's measurement z_i, expressed as $\tilde{z}_i = \frac{z_i - z_{\min}}{z_{\max} - z_{\min}}$, where $z_{\min} = \min_i\{z_i\}$ and $z_{\max} = \max_i\{z_i\}$. When $l \to \infty$, we obtain a complete Gaussian model, as shown in Fig. 1(a) for a single POI.



(a) Signal-POI physical field. (b) Shape of physical field with $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ varying.

Fig. 1. Single-POI case.

An intuitive way to estimate the state and the location of the POI is to compare the sensors' measurements $\{z_1, \ldots, z_l\}$ and select the maximum value $z_j = \max_i\{z_i\}$ as the state of the POI and the location of sensor s_j (x_j, y_j) as the location of the POI. However, as the number of sensors is limited, we have to use the limited data set $\{x_i, y_i, z_i\}$ to estimate the state and the location of the POI. With two-dimensional random variables $\boldsymbol{\chi}$, the Gaussian model of Fig. 1(a) is formulated by

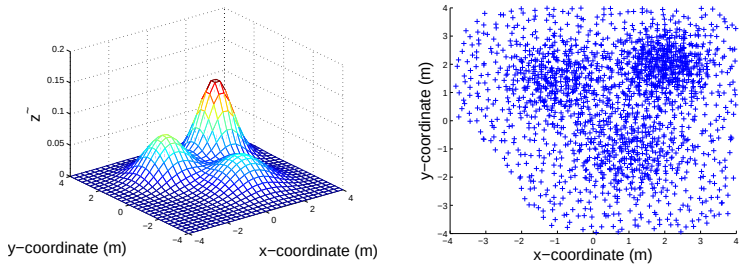
$$\mathcal{N}(\boldsymbol{\chi}|\boldsymbol{\mu}, \boldsymbol{\sigma}) = \frac{1}{2\pi \left(\det \boldsymbol{\sigma}\right)^{1/2}} \exp\left\{\frac{(\boldsymbol{\chi} - \boldsymbol{\mu})' \, \boldsymbol{\sigma}^{-1} \, (\boldsymbol{\chi} - \boldsymbol{\mu})}{-2}\right\}, \tag{2}$$

where

$$\boldsymbol{\mu} = \begin{bmatrix} \mu_1 \\ \mu_2 \end{bmatrix}, \boldsymbol{\sigma} = \begin{bmatrix} \sigma_1^2 & \rho\sigma_1\sigma_2 \\ \rho\sigma_1\sigma_2 & \sigma_1^2 \end{bmatrix}, \det \boldsymbol{\sigma} = \sigma_1^2\sigma_2^2(1-\rho^2).$$

Substituting the sensor S_i's location (x_i, y_i) into (2), we obtain the normalized sensor S_i's measurement $\tilde{z}_i = \mathcal{N}((x_j, y_j)|\boldsymbol{\mu}, \boldsymbol{\sigma})$.

For a fix Gaussian model, its mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\sigma}$ are constant. The location and the shape of Gaussian model changes with the values of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$. As shown in Fig. 1(b), the Gaussian models of the POIs in $(0, 0)$, $(4, 4)$, $(-4, -4)$ are with the parameters $\{\mu_1 = 0, \mu_2 = 0, \rho = 0, \sigma_1 = 1, \sigma_2 = 1\}$, $\{\mu_1 = 4, \mu_2 = 4, \rho = 0, \sigma_1 = 1, \sigma_2 = 1\}$ and $\{\mu_1 = -4, \mu_2 = -4, \rho = 0.5, \sigma_1 = 1, \sigma_2 = 1\}$, respectively. Hence, the mean $\boldsymbol{\mu}$ and the covariance $\boldsymbol{\sigma}$ determines the location and the shape of Gaussian model, respectively. Since l sensors are deployed in the ROI, we obtain a set of sensor data $\{x_i, y_i, z_i\}$ $(1 \leq i \leq l)$. Therefore, if the Gaussian parameters $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ are estimated from the data set $\{x_i, y_i, z_i\}$, we can use $\boldsymbol{\mu}$ and $\mathcal{N}(\boldsymbol{\mu}|\boldsymbol{\mu}, \boldsymbol{\sigma})$ to describe the location and the state of the POI, respectively. For the single-POI case, MLE is an efficient method to estimate the values of $\boldsymbol{\mu}$ and $\boldsymbol{\sigma}$ [10]. For the multi-POI case, as shown in Fig. 2(a), a linear combination of several Gaussians can characterize the data set. However, the MLE is no longer suitable for this case since the weights of the linear combination cannot be obtained by MLE. The next section presents our approach to estimate the values of $\{\boldsymbol{\mu}_1, \boldsymbol{\sigma}_1, \ldots, \boldsymbol{\mu}_n, \boldsymbol{\sigma}_n\}$ through the sensor data $\{x_1, y_1, z_1, \ldots, x_l, y_l, z_l\}$ for the multi-POI case.



(a) Multi-POI physical field. (b) Multi-POI sensor data mapping.

Fig. 2. Multi-POI case.

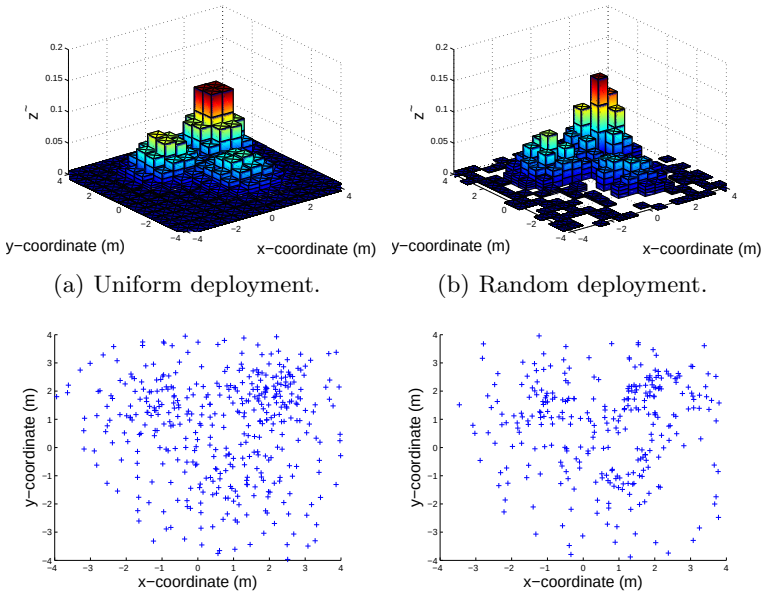
2.2 Virtual Sample Method

In order to estimate the values of mean $\boldsymbol{\mu}$ and covariance $\boldsymbol{\sigma}$, we need to plot enough two-dimensional samples $\{s_i^x, s_i^y\}$, which reside within the ellipses that represent the distribution [2]. However, from the sensor measurements, we only obtain a set of three-dimensional data $\{x_i, y_i, z_i\}$. Therefore, we propose a virtual sample method to map the available data $\{x_i, y_i, z_i\}$ to the desired two-dimensional samples $\{s_i^x, s_i^y\}$:

1. The ROI is divided into several grids, such that each grid g_i has the same size and contains at most one sensor S_i.
2. Under a given constant $\mathcal{K}$, $M_i = \mathcal{K}\tilde{z}_i$ virtual samples $\{s_{i,j}^x, s_{i,j}^y\}$ $(1 \leq j \leq M_i)$ are uniformly deployed in the grid g_i.

Denote $\boldsymbol{s} = \{s_1^x, s_1^y, \ldots, s_N^x, s_N^y\}$ as the total virtual samples, where $N = \sum_{i=1}^{l} M_i$ is the total number of the virtual samples. The virtual samples of the sensor data in Fig. 2(a) are shown in Fig. 2(b). From this figure, we observe that compared with the original data $\{x_i, y_i, z_i\}$, the virtual samples $\{s_i^x, s_i^y\}$ are much more suitable for clustering, processing, and analyzing, due to the adjustable parameter $\mathcal{K}$.

During the state estimation, the estimation accuracy is influenced by the number of the sensors. The more sensors are deployed in the ROI, the better is the estimation accuracy. Moreover, the sensors can be either uniformly or randomly deployed. Plotting the sensor data $\{x_i, y_i, z_i\}$, we obtain a non-smooth or incomplete mixed-Gaussian model, as shown in Fig. 3(a) and Fig. 3(b). Comparing Fig. 3(a) and Fig. 3(b) with Fig. 2(a), we observe that in the uniform deployment case, there exists a large gap between the real and the estimated system states; while in the random deployment case, some information is missing since there are some grids that are not covered by the sensors. Using the virtual sample method, we derive a set of spare and incomplete virtual samples, as shown in Fig. 3(c) and Fig. 3(d). Finally, through the GMM, we obtain $\{\hat{\varepsilon}_i, \hat{\boldsymbol{\mu}}_i, \hat{\boldsymbol{\sigma}}_i\}$ $(1 \leq i \leq n)$, where $\hat{\varepsilon}_i$, $\hat{\boldsymbol{\mu}}_i$ and $\hat{\boldsymbol{\sigma}}_i$ are the estimations of the weight, the mean and the covariance of the i^{th} Gaussian model, respectively. Due to page limitations, the interested reader can find the details of GMM in [2].



(a) Uniform deployment. (b) Random deployment.

(c) Virtual samples: uniform deployment. (d) Virtual samples: random deployment.

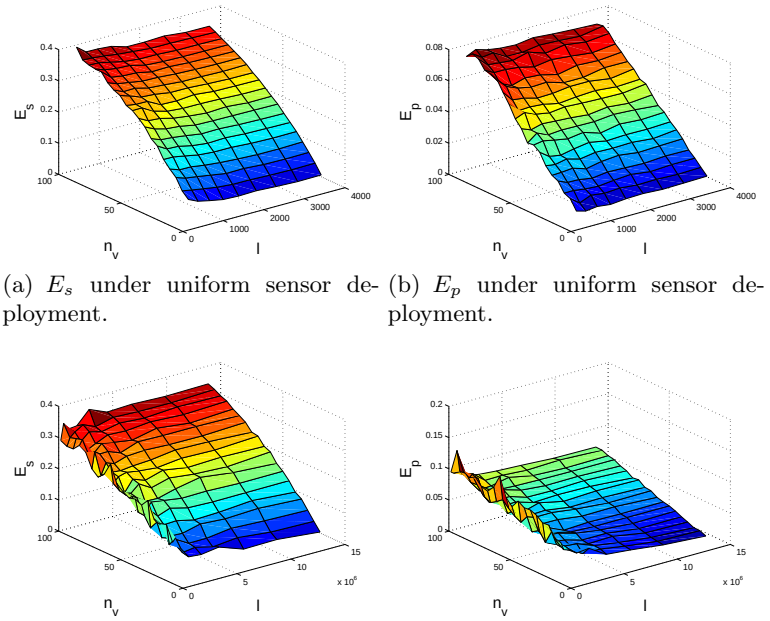
Fig. 3. State estimation with a limited number of sensors.

Fig. 4 shows the average estimation errors of 1) the states of the POIs, i.e., $E_s = \frac{1}{n}\sum_{i=1}^{n}|t_i - \hat{t}_i|$, and 2) the positions of the POIs, i.e., $E_p = \frac{1}{n}\sum_{i=1}^{n}\|\boldsymbol{\mu}_i - \hat{\boldsymbol{\mu}}_i\|_2^2$, with the number of the sensors l and the number of the virtual samples N varying. $\hat{t}_i$ is an estimation of the system state t_i, $M_i = \lceil \mathcal{K}\tilde{z}_i \rceil = \lceil \frac{10000}{n_v}\tilde{z}_i \rceil$, $l \in [1000, 4000]$, and $n_v \in [10, 100]$. From Fig. 4, we observe that in both uniform and random sensor deployment: 1) if n_v is fixed, $E_s \propto \frac{1}{l}$, $E_p \propto \frac{1}{l}$, and 2) if l is fixed, $E_s \propto n_v$, $E_p \propto n_v$. This is because the virtual samples can be used to describe the Gaussian distribution. The more the virtual samples are, the better is the description of the Gaussian model. This characteristic implies that when the number of the sensors is small, we can increase the number of the virtual samples so as to improve the estimation accuracy.

At each step k, using the GMM, we obtain the estimations of the states of n POIs, i.e., $\{\hat{t}_1(k), \ldots, \hat{t}_n(k)\}$. In this paper, we introduce a function $\mathcal{F}$ to evaluate the priorities of the events, expressed by

$$p_i(k) = \begin{cases} \mathcal{F}(\hat{t}_i(k)), & \hat{t}_i(k) < t_i^*, \\ 1, & \hat{t}_i(k) \geq t_i^*, \end{cases} \tag{3}$$

where $p_i(k)$ is the probability of the event e_i at the k^{th} step, the function $\mathcal{F}$ is determined by the applications. It is given in advance and it can be calculated based on simulations or experiments. For instance, in forest fire detection and extinction, if the temperature of POI_1 exceeds $100°C$, we assume that a fire breaks out at the POI_1, i.e., $p_1(k) = 1$; if the temperature of POI_1 is below than



(a) E_s under uniform sensor deployment.

(b) E_p under uniform sensor deployment.

(c) E_s under random sensor deployment.

(d) E_p under random sensor deployment.

Fig. 4. State estimation accuracy of virtual sample method with l and N_v varying.

$100°C$, e.g., $80°C$, we assume that $p_1(k) = 0.8$. Thus, we have $\mathcal{F}(\hat{t}_1(k)) = \frac{\hat{t}_1(k)}{100}$ and $t_1^* = 100$. Based on different application requirements, we define a priority threshold p_i^* for the event e_i and schedule an actuator to handle it, if $p_i(k) \geq p_i^*$.

3 Actuator Scheduling Scheme

We denote κ as the number of the events that need to be handled at the k^{th} step. If the number of the actuators is higher than the number of the events, i.e., $m \geq \kappa$, the actuator scheduling problem is defined as how to move the actuators toward the events as soon as possible. In order to formulate the actuator scheduling problem, we introduce a $m \times \kappa$ binary matrix $\boldsymbol{Q} = [q_{ij}]$. If $q_{ij} = 1$, the actuator A_i is scheduled to handle the event e_j, otherwise, $q_{ij} = 0$. Therefore, the actuator scheduling problem is described by

$$\min_{\boldsymbol{Q}} \ J_1 = \sum_{i=1}^{m} \sum_{j=1}^{\kappa} d_{ij}(k) q_{ij} \tag{4}$$

$$\text{s.t.} \begin{cases} \sum_{i=1}^{m} q_{ij} = 1, \ 1 \leq j \leq \kappa, \\ \sum_{j=1}^{\kappa} q_{ij} \leq 1, \ 1 \leq i \leq m. \end{cases}$$

where $d_{ij}(k)$ is the Euclidean distance between the actuator A_i and the event e_j at the k^{th} step, and the constraints imply that each event requires one actuator to handle it and one actuator moves towards at most one event.

On the other hand, if the number of actuators is smaller than the number of the events, i.e., $m < \kappa$, the available actuators cannot handle all the events at the same time. As it is not possible to deal with all the events, we need to schedule the actuators based on the priorities of the events. Let's initially consider the simple case, where only one actuator A_i is scheduled to handle r events $\{e_1, \ldots, e_r\}$. Although the priorities of the events $\{p_1(k), \ldots, p_r(k)\}$ can be derived by (3), the decision of the actuator scheduling at the k^{th} step cannot be based on them. This is due to the fact that when the actuator A_i arrives at the location of the desired event, e.g., e_j, the values of event priorities $\{p_1, \ldots, p_r\}$ have already changed. Therefore, the decision of the actuator scheduling should be made based on the priorities of the events at the future steps. Our approach uses the Regression Algorithm (RA) [2] to predict the future priorities.

To illustrate the future priority prediction, we consider the estimation of $p_j(k)$. Suppose that $p_j(k)$ is not oscillating, and, thus, the target variable $p_j(k)$ can be given by a deterministic function ϕ_j with additive zero mean Gaussian noise ω_j, i.e., $p_j(k) = \alpha_j k^{\beta_j} + \gamma_j + \omega_j(\rho_j) = \phi_j(k) + \omega_j(\rho_j)$, where $\{\alpha_j, \beta_j, \gamma_j\}$ are the fitting parameters, and ρ_j is the inverse variance of Gaussian noise. Thus, we have $\mathcal{N}(p_j(k)|\phi_j(k), \rho_j^{-1}) = \frac{1}{\sqrt{2\pi\rho_j^{-1}}} \exp\left\{\frac{(p_j(k)-\phi_j(k))^2}{-2\rho_j^{-1}}\right\}$. Denote ϑ as the window size of the estimation, and assume that the historical data $\{p_j(k-\vartheta), \ldots, p_j(k)\}$ are independent and identically distributed (i.i.d.). We obtain a likelihood function $\mathcal{L}(\boldsymbol{p}_j) = \prod_{i=k-\vartheta}^{k} \mathcal{N}(p_j(i)|\phi_j(i), \beta_j^{-1})$. Taking the derivative of $\ln \mathcal{L}(\boldsymbol{p}_j)$ with respect to $\{\alpha_j, \beta_j, \gamma_j, \rho_j\}$ and make them to 0, we derive the estimations of these parameters, i.e., $(\hat{\alpha}_j, \hat{\beta}_j, \hat{\gamma}_j, \hat{\rho}_j)$. Therefore, the priority $p_j(k)$ at the future step k' $(k' > k)$ is estimated by $\hat{p}_j(k') = \hat{\alpha}_j(k')^{\hat{\beta}_j} + \hat{\gamma}_j + \omega_j(\hat{\rho}_j)$.

Denote v_i and Δ as the moving speed of actuator A_i and the system sampling period, respectively. The actuator scheduling process is summarized as follows:

1. At the k^{th} step, based on the distance $d_{ij}(k)$ between the actuator A_i and the event e_j, we obtain the corresponding moving time $\frac{d_{ij}(k)}{v_i}$. Then, we estimate the future priorities of the events $\{\hat{p}_j(k + \frac{d_{ij}(k)}{v_i \Delta k})\}$, and schedule the actuator A_i to handle the event with the highest priority, e.g., event e_d, where $\hat{p}_d(k + \frac{d_{id}(k)}{v_i \Delta k}) = \max_j\{\hat{p}_j(k + \frac{d_{ij}(k)}{v_i \Delta k})\}$.
2. Step 1 is repeated to schedule actuator A_i to handle the residual $r-1$ events.

Denote $p_{ij}(k)$ as the future priority of event e_j at the $(k + \frac{d_{ij}(k)}{v_i \Delta k})^{th}$ step, which is estimated at the k^{th} step. Therefore, the actuator scheduling problem is formulated as follows:

$$\max_{\boldsymbol{Q}} \; J_2 = \sum\nolimits_{i=1}^{m} \sum\nolimits_{j=1}^{\kappa} p_{ij}(k) q_{ij} \tag{5}$$
$$\text{s.t.} \begin{cases} \sum_{i=1}^{m} q_{ij} \leq 1, \; 1 \leq j \leq \kappa, \\ \sum_{j=1}^{\kappa} q_{ij} = 1, \; 1 \leq i \leq m. \end{cases}$$

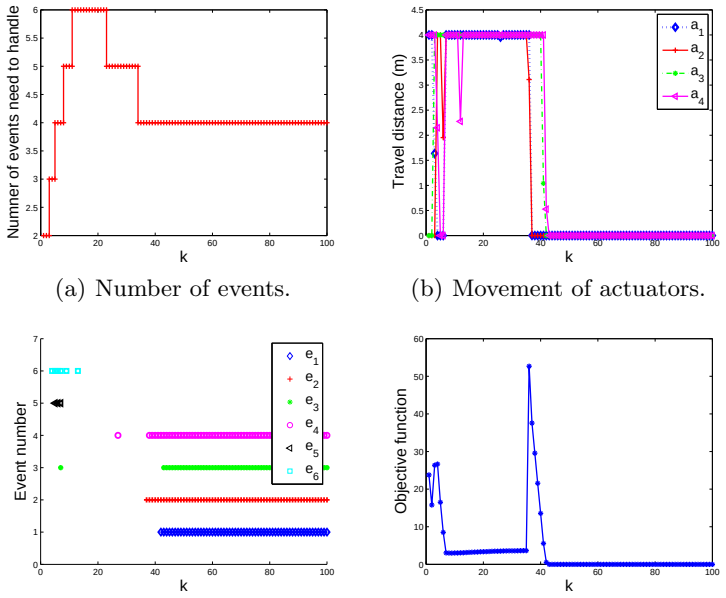
Note that the problems (4) and (5) are ILP, which can be solved by existing ILP algorithms [9]. As the proposed scheduling methods are event-driven, i.e., when a new event occur, the scheduling decision, which is given by the solution of problem (4) or (5), should be updated.

4 Performance evaluation

We randomly deploy 4 actuators to 6 POIs in a 50 m $\times$ 50 m ROI, set $p_i^* = 0.5$, and assume that the priorities $\{p_1(k), p_2(k), p_3(k), p_4(k)\}$ are monotonic increasing, while the priorities $\{p_5(k), p_6(k)\}$ are monotonic decreasing. The dynamic change of the number of occurring events in time steps is shown in Fig. 5(a). From this figure, we observe that the scheduling decisions are divided into 3 periods based on the step k: $[1, 6]$, $[7, 35]$ and $[36, 100]$. Due to the number of occurred events, in the 1^{st} and the 3^{rd} periods we schedule the actuators based on the solution of problem (4). In the 2^{nd} period, we schedule the actuators based on the solution of problem (5). Moreover, in the 2^{nd} period, the actuators need to keep moving, as shown in Fig. 5(b). This is because the number of the actuators is smaller than the number of the events. When the actuator A_i finishes the assigned task in the current round, it will be scheduled again to handle another event in the next round. Fig. 5(c) shows at which step the actuators arrive at the desired events. From this figure, we observe that each event is handled by at least one actuator. Fig. 5(d) shows the corresponding objective function, which is a combination of J_1 and J_2. In the 1^{st} and the 3^{rd} periods, the aim is to minimize J_1 by scheduling the actuator A_i to the nearest event. While in the 2^{nd} period, the aim is to maximize J_2 by handling the events with the higher priorities.

In order to evaluate the performance of proposed scheme, we build a testbed which consists of three main components: 1) base station, 2) LEGO Mindstorm NXT wheeled robot (Fig. 6(a)), and 3) OptiTrack system [1] (Fig. 6(b)). The OptiTrack system includes 6 cameras which are used to track the mobile targets in a 300 cm $\times$ 300 cm ROI. The base station is responsible for processing the data received from OptiTrack system, making the scheduling decision (Matlab), and sending the comments to the robot though a bluetooth connection.

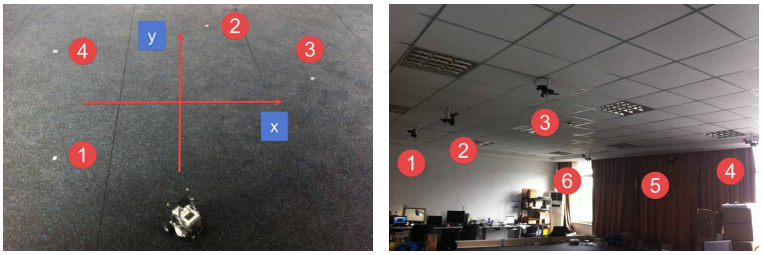
The locations of POI_1, POI_2, POI_3 and POI_4 are set to $(-70, -50)$, $(30, 120)$, $(100, 25)$ and $(-100, 70)$, respectively. The initial location of actuator A_1 is set to $(0, -100)$, as shown in Fig. 6(a). In the base station, we simulate the dynamic changes of the priorities of the events $\{p_1(k), p_2(k), p_3(k), p_4(k)\}$ and assume that $p_1(k)$ is monotonic decreasing, while $\{p_2(k), p_3(k), p_4(k)\}$ are monotonic increasing. If the actuator A_1 arrives at POI_i, $p_i(k) = 0$ immediately. But after that, $p_i(k)$ increases gradually except $p_1(k)$, as it is monotonic decreasing. The base station is able to receive the information with respect to the robot position from the OptiTrack system. This information is recorded in a txt file for off-line analysis. The real robot trajectory and the changes in the probability of the events are shown in Fig. 7(a) and Fig. 7(b), respectively.



(a) Number of events. (b) Movement of actuators.

(c) Steps of actuators reach events (d) Dynamic objective function.

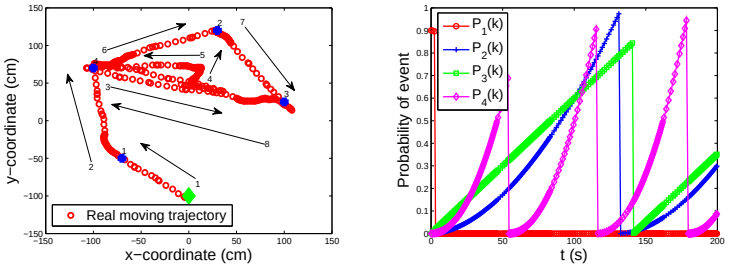
Fig. 5. Actuator scheduling.



(a) LEGO Mindstorm NXT robot. (b) OptiTrack system.

Fig. 6. Overview of testbed.

The experiment runs between $0 \sim 200$ s, $p_i^* = 0.5$, and the system sample period $\Delta = 0.1$ s. Table 1 shows the scheduling sequence, where L_i, L_s, L_d, T_s, T_e are the initial position, the desired position, the final position, the start moving time, and the stop moving time, respectively, and $*$ represents the idle operation. Based on Table 1, the whole scheduling sequence is divided into 8 periods. The moving details are analyzed as follows: at the beginning, $p_1 = 0.9$, the actuator A_1 spends 3 s to move towards the event e_1 and to handle it. Then, $p_1 = 0$. At the same time, p_2, p_3 and p_4 are increasing gradually during this period, and p_4 has the fastest growth rate. When $p_4 > 0.5$ at $t = 46$ s, the actuator A_1 moves towards the event e_4 and arrives there at $t = 54$ s. Then, $p_4 = 0$. When $t = 83$ s, we have $p_3 > 0.5$. The actuator A_1 starts moving towards



(a) Robot moving trajectory (b) Dynamic probability change under actuator influence

Fig. 7. Experimental results.

the event e_3. During the movement, at $t = 91$ s, we have $p_2 > 0.5$. However, the actuator A_1 hasn't reached the position of e_3 yet, and, thus, the events e_2 and e_3 occur simultaneously. Therefore, we schedule actuator A_1 based on p_{12} and p_{13}. p_{ij} is the priority of event e_j at step $k + \frac{d_{ij}}{v_1 \Delta}$, which is estimated at current step k. Since p_2 grows faster, i.e., $p_{12} > p_{13}$, the actuator A_1 changes its direction and moves towards the event e_2. When $t = 100$ s, we have $p_4 > 0.5$, and, thus, the events e_2, e_3 and e_4 need to be handled simultaneously. In a similar way, by comparing p_{12}, p_{13} and p_{14}, the actuator A_1 goes back to the event e_4 and arrives at e_4 at $t = 116$ s. After that, the actuator A_1 moves towards the second highest probability event e_2, and it arrives there at $t = 131$ s. Since the events e_2 and e_4 have been handled, the remaining event is e_3. Therefore, the actuator A_1 moves towards the event e_3 immediately and arrives at e_3 at $t = 141$ s. The actuator A_1 stays at e_3 until $p_4 > 0.5$ at $t = 162$ s, and, then, the actuator A_1 moves towards the event e_4 and it arrives there at $t = 179$ s. Since the events e_1, e_2 and e_3 are smaller than 0.5 during the period $t = 179 \sim 200$ s, the actuator A_1 stays at e_4 until the experiment ends.

	L_i	L_s	L_d	T_s	T_e		L_i	L_s	L_d	T_s	T_e		L_i	L_s	L_d	T_s	T_e		L_i	L_s	L_d	T_s	T_e
R_1	0	e_1	e_1	0	3	R_2	e_1	e_4	e_4	46	54	R_3	e_4	e_3	$*$	83	$*$	R_4	e_3	e_2	$*$	$*$	$*$
R_5	e_2	e_4	$*$	$*$	116	R_6	e_4	e_2	e_2	116	131	R_7	e_2	e_3	e_3	131	141	R_8	e_3	e_4	e_4	162	179

Table 1. Actuator moving sequence

5 Conclusion

This paper deals with the design of CPS for environmental monitoring applications, where sensing and control are the two main tasks of such system. The challenge during the CPS design is how to apply an effective state estimation and actuator scheduling among the nodes. First, according to the characteristic of physical field distribution, we propose a GMM-based method to estimate the locations and the states of the events. Then, based on the number of available actuators and the number of events need to be handled, we propose two actuator

scheduling schemes. Based on the relative distances between the events and the actuators, the former one schedules the nearest actuators to the event areas so as to fulfill the actuator relocation as soon as possible. The latter one predicts the priorities of the events through the regression algorithm, and follows priority-based sequence to handle these events with high priorities. Finally, simulations and experiments are conducted to illustrate the effectiveness of the proposed methods.

Acknowledgment

This research is funded by INRIA post-doctoral research fellowship program, and is partly sponsored by the National Natural Science foundation of China (Grant No. 61403340 and 61573103).

References

1. http://www.naturalpoint.com/optitrack/
2. Bishop, C.M., et al.: Pattern recognition and machine learning. Springer (2006)
3. Cao, X., Chen, J., Xiao, Y., Sun, Y.: Building-environment control with wireless sensor and actuator networks: centralized versus distributed. IEEE Trans. Ind. Electron. **57**(11), 3596–3605 (2010)
4. Chen, J., Cao, X., Cheng, P., Xiao, Y., Sun, Y.: Distributed collaborative control for industrial automation with wireless sensor and actuator networks. IEEE Trans. Ind. Electron. **57**(12), 4219–4230 (2010)
5. Krause, A., Singh, A., Guestrin, C.: Near-optimal sensor placements in gaussian processes: theory, efficient algorithms and empirical studies. J. Mach. Learn. Res. **9**, 235–284 (2008)
6. Li, H., Lai, L., Poor, H.V.: Multicast routing for decentralized control of cyber physical systems with an application in smart grid. IEEE J. Sel. Areas Commun. **30**(6), 1097–1107 (2012)
7. Li, X., Qiao, C., Yu, X., Wagh, A., Sudhaakar, R., Addepalli, S.: Toward effective service scheduling for human drivers in vehicular cyber-physical systems. IEEE Trans. Parallel Distrib. Syst. **23**(9), 1775–1789 (2012)
8. Mo, L., Cao, X., Song, Y., Kritikakou, A.: Distributed node coordination for real-time energy-constrained control in wireless sensor and actuator networks. IEEE Internet Things J. pp. 1–12 (DOI:101109/JIOT20182839030 2018)
9. Mo, L., Kritikakou, A., Sentieys, O.: Controllable QoS for imprecise computation tasks on DVFS multicores with time and energy constraints. IEEE J. Emerg. Sel. Topics Circuits Syst. pp. 1–14 (DOI:101109/JETCAS20182852005 2018)
10. Ota, K., Dong, M., Cheng, Z., Wang, J., Li, X., Shen, X.S.: ORACLE: mobility control in wireless sensor and actor networks. Comput. Commun **35**(9), 1029–1037 (2012)
11. Weiss, V., Andor, L., Renner, G., Varady, T.: Advanced surface fitting techniques. Comput. Aided Geom. D. **19**(1), 19–42 (2002)
12. Wen, Y.J., Agogino, A.M.: Control of wireless-networked lighting in open-plan offices. Lighting Res. Technol. **43**(2), 235–248 (2011)
13. Zhang, X.M., Han, Q.L., Yu, X.: Survey on recent advances in networked control systems. IEEE Trans. Ind. Informat. **12**(5), 1740–1752 (2016)