Towards Cyber-physical Systems in Social Spaces: The Data Reliability Challenge

Shiguang Wang *, Dong Wang §, Lu Su†, Lance Kaplan‡, Tarek Abdelzaher*
*University of Illinois at Urbana-Champaign, Urbana, IL 61801, USA
§University of Notre Dame, Notre Dame, IN 46556, USA
†State University of New York at Buffalo, Buffalo, NY 14260, USA
‡Army Research Labs, Adelphi, MD 20783, USA

***Abstract*—Today's cyber-physical systems (CPS) increasingly operate in *social spaces*. Examples include transportation systems, disaster response systems, and the smart grid, where humans are the drivers, survivors, or users. Much information about the evolving system can be collected from humans in the loop; a practice that is often called *crowd-sensing*. Crowd-sensing has not traditionally been considered a CPS topic, largely due to the difficulty in rigorously assessing its reliability. This paper aims to change that status quo by developing a mathematical approach for quantitatively assessing the probability of correctness of collected observations (about an evolving physical system), when the observations are reported by sources whose reliability is unknown. The paper extends prior literature on state estimation from noisy inputs, that often assumed unreliable sources that fall into one or a small number of categories, each with the same (possibly unknown) background noise distribution. In contrast, in the case of crowd-sensing, not only do we assume that the error distribution is unknown but also that each (human) sensor has its own possibly different error distribution. Given the above assumptions, we rigorously estimate data reliability in crowd-sensing systems, hence enabling their exploitation as state estimators in CPS feedback loops. We first consider applications where state is described by a number of binary variables, then extend the approach trivially to multivalued variables. The approach also extends prior work that addressed the problem in the special case of systems whose state does not change over time. Evaluation results, using both simulation and a real-life case-study, demonstrate the accuracy of the approach.**

I. Introduction

Modern CPS applications increasingly operate in social spaces, where humans play an important part in the overall system. Hence, future applications should increasingly be engineered with an understanding of the humans in the loop. In this paper, we focus on the role of humans as *sensors* in CPS systems; a practice that is commonly known as *crowd-sensing*. Humans act as sensors when they contribute data (either direcltly or via sensors they own) that a CPS application can use. For example, drivers may contribute data on the state of traffic congestion at various locales, and survivors may contribute data on damage in the aftermath of a natural disaster. A challenge in this context is that data sources may be unreliable. In fact, the reliability of individual observers in crowd-sensing applications is generally not known.

A common thread in CPS research focuses on *reliability* of cyber-physical systems. Most research focused on two aspects of CPS reliability; namely, correctness of *temporal behavior* and correctness of *software function*. In order for crowd-sensing to become a viable component in CPS feedback loops, one needs to understand correctness of *collected observations* as well. We call this latter challenge the data reliability challenge, to complement the challenges of timing reliability and software reliability mentioned above.

Consider a CPS application that uses crowd-sensing to collect data about a physical environment. The data reliability challenge refers to designing a state estimator that takes raw unreliable crowd-sensing data as input and outputs reliable estimates of the underlying physical state of the environment, as well as appropriate error bounds. Building optimal state estimators from noisy inputs is an old topic in estimation theory and embedded systems. Much like our work, past research often assumed that sources are unreliable and the noise model is not known. However, in the case of physical sensors, prior research usually assumed that errors of different sensors are drawn from the same distribution (or from a small set of different distributions). In contrast, we assume that each source is unique. Hence, each source has its own error distribution. None of these distributions is known.

In this paper, we also assume that the state of the observed environment changes over time. Hence, when conflicting observations arrive, it is not clear whether one is wrong, or whether the ground truth changed between observations. Had the reliability of different observation sources been known, it would have been easy to statistically fuse them, but since error distributions are both unknown and unique to each source, reconciling conflicts is a bigger problem. This paper is the *first* to offer a rigorous estimation-theoretic approach for state estimation in crowd-sensing applications, where (i) observers have unknown reliability (ii) the error distribution is unique to each observer, and (iii) the observed physical events have time-varying state. It extends prior work by the authors, that solved the problem in the restricted special case when physical state is immutable [35], [31], [32]. Note that, this restriction is not suited for most cyber-physical systems.

One way to accommodate state changes is to cut time into small observation windows and consider only one observation window at a time, during which state can be assumed constant. One can then apply the former static approach [35] independently within each window. Unfortunately, this reduces the number of observations that can be considered within a given window, making it harder to assess their veracity. A

1052-8725/14 $31.00 © 2014 IEEE
DOI 10.1109/RTSS.2014.19

much better approach is to take into account the model of state evolution from one window to the next and reduce trust in observations that are less consistent with that model. Unlike traditional estimation problems, where a model of observation noise is also available, in crowd-sensing, observations can come from different sources whose reliability (i.e., noise model) is not known. Hence, it is hard to tell genuine state changes from incorrect reports. Our contribution lies in taking a model of state evolution into account such that a maximum likelihood estimate can be arrived at, that jointly estimates the reliability of individual observations and the reliability of individual sources, taking only a dynamic model of the underlying observed system as input.

We analytically derive an error bound for the above estimator, by computing the Cramer-Rao lower bound [5] that bounds estimator variance and hence derive confidence intervals. We then evaluate our algorithm through simulations and a real-world crowd-sensing application in which sources report the availability of street parking spots on a university campus. We show that our algorithm outperforms prior state-of-the-art solutions in both event state estimation accuracy and source reliability estimation accuracy.

The rest of the paper is organized as follows. Section II reviews the related work. Our problem is formulated in Section III. Section IV describes how a dynamic system model is converted to an input for our maximum likelihood estimator. We describe our algorithm in Section V and its analysis in Section VI. We extend our algorithm to the general multivalued case in Section VII. Our solution is evaluated in Section VIII. Finally, we discuss and conclude the paper in Section IX and Section X respectively.

II. Related Work

Reliability is a critical requirement for cyber-physical systems [16]. Much prior research focused on temporal and functional reliability of CPS applications. For example, Eidson et al. [7] presented a programming model called PTIDES for the reliable timing control of the cyber-physical systems. Clarke et al. [4] applied formal analysis technique on autonomous transportation control for cars, trains, and aircraft. Faza et al. [9] suggested the use of software fault injection combined with physical failures in identifying integrated cyber-physical failure scenarios for the smart grid. Sha et al. [25] developed a hybrid approach that combines fault-tolerant architectures with formal verification to support the design of safe and robust cyber-physical systems. Different from prior efforts, this paper addresses the *data reliability* challenge that arises in cyber-physical systems operating in social spaces, where significant amounts of data are collected from the "crowd". In this scenario, the "crowd" functions as a noisy sensor of large amounts of physical states. A state estimator is needed to optimally recover reliable data and accurately estimate error bounds.

The work is motivated by human-in-the-loop cyber-physical systems; a challenging and promising class of CPS [23]. Many examples of such systems appear in recent literature, where humans plays important roles in feedback loops, such as operator, load, disturbance, or controlled plant. For example, Lu et al. developed a smart thermostat system to monitor the occupancy and sleep pattern of the residents and turned off the HVAC when not needed [17]. Huang et al. designed a mathematical model to determine the insulin injection by closely monitoring glucose level when it reaches a threshold, a key challenge to design an artificial pancreas [11]. Our work is complementary to the efforts mentioned above in that we investigate the role of humans as *sensors*. Hence, we are interested in addressing the reliability challenge that ensues when data is obtained from unvetted sources, where the reliability of data sources is unknown and the states of observed variables may evolve over time.

Our work is related to the system state estimation problem with unreliable sensors in CPS. Sinopoli et al. designed a discrete Kalman filter to estimate the system state when the sensor reports were intermittent [26]. Ishwar et al. estimated the states of the sink node with noisy sensor nodes in WSNs [13]. Mathematical tools were proposed by Schenato et al. to control and estimate the states of physical systems on top of a lossy network [22]. Masazade et al. proposed a probabilistic transmission scheme to near-optimally estimate the system parameter in WSN with sensing noises [18]. However, the sensor error is either assumed known [18], or generated from a common distribution with known parameters [26], [13], [22]. In contrast, in crowd-sensing applications, not only do we assume that the error distribution is unknown but also that each (human) sensor has its own possibly different error distribution. Therefore, none of the prior work is applicable in crowd-sensing.

The importance of crowd-sensing as a possible data input in cyber-physical systems is attributed to the proliferation of mobile sensors owned by individuals (e.g., smart phones) and the pervasive Internet connectivity. Hence, humans can be sensor carriers [15] (e.g., opportunistic sensing), sensor operators [2], or sensor themselves [32]. Wang et al. proposed data prioritizing schemes to maximize the data coverage in an information space [38] and to maximize the collected data diversity [36], [37] for crowd-sensing applications. An early overview of crowd-sensing applications is described in [1]. Examples of early systems include CenWits [10], CarTel [12], BikeNet [8], and CabSense [24].

Recently, the problem of fact-finding, which refers to ascertaining correctness of data from *sources of unknown reliability*, has drawn significant attention. It has been studied extensively in the data mining and machine learning communities. One of the earliest efforts is Hubs and Authorities [14] that presented a basic fact-finder where the belief in a claim and the truthfulness of a source are jointly computed in a simple iterative fashion. Later on, Yin et al. introduced TruthFinder as an unsupervised fact-finder for trust analysis on a providers-facts network [39]. Pasternack et al. extended the fact-finder framework by incorporating prior knowledge into the analysis and proposed several extended algorithms: Average.Log, Investment, and Pooled Investment [21]. Su et al. proposed supervised learning frameworks to improve the quality of aggregated decision in sensing systems [27], [28], [29]. Additional efforts were spent in order to enhance the

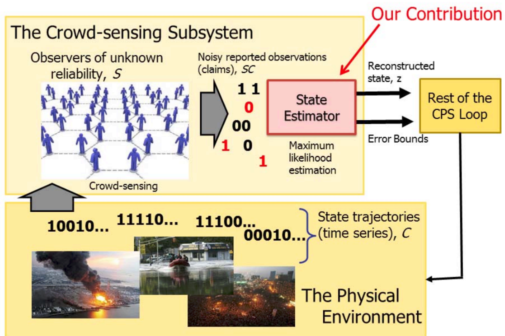


Fig. 1. An example of CPS system with humans in the loop.

basic fact-finding framework by incorporating analysis on properties or dependencies within claims or sources.

The above work is heuristic in nature; it does not offer optimality properties and does not allow computation of error bounds. The latter is an important requirement for a state estimator in CPS applications. Towards an optimal solution, Wang et al., proposed a Maximum Likelihood Estimation (MLE) framework [35], [34] that offers a joint estimation on source reliability and claim correctness based on a set of general simplifying assumptions. In their following work, Wang et al. further extended the framework to handle streaming data [30] and source dependencies [32]. The approach was compared to several of the aforementioned fact-finders and was shown to outperform them in estimation accuracy, while also offering error bounds. However, their work is unsuited for CPS, since it did not consider the evolving event states which is common in CPS applications.

The algorithms proposed in this paper extend the above MLE based framework by being the first to study the state estimation problem (with unknown source reliability) in crowd-sensing applications with time-varying system states. We further derive a Cramer-Rao lower bound for the resulting novel maximum likelihood estimator.

III. Problem Formulation

Consider a CPS application that uses a crowd-sensing subsystem to estimate the state of a physical environment that changes dynamically over time. An example of such a system is shown in Figure 1. It is desired to develop the appropriate state estimator that converts raw noisy crowd-sensing data, from sources of unknown reliability, into state estimates of quantified reliability and error bounds.

We model the physical environment by a set of measured variables, $\mathcal{C}$, whose values constitute the system state we want to estimate. We consider applications where the state of interest varies over time (i.e., the values of these variables change dynamically).

We focus, in this paper, on the *harder* case, where state variables are binary. While it may appear to be restrictive, this particular case is more computationally challenging because state, in this case, does not have "inertia". In continuous systems, such inertia leads to a smooth state evolution that can be leveraged to eliminate outliers, extrapolate trends, and suppress noise. A binary variable, in contrast, can change between the two extremes of its range (0 and 1) at any time. Hence, removing incorrect measurements, predicting correct values, and eliminating noise become harder problems. Indeed we show that solutions to the binary case can easily generalize to the multivalued-state case.

One should also note that exploring systems of binary states is more than just a step towards understanding more general state representations. Binary state is a versatile abstraction. It can indicate, for example, presence or absence of arbitrary conditions, symptoms, features, or alarms in specified locations of the monitored system. More importantly, given the general lack of reliability of human observers in crowd-sensing scenarios, tasking humans with making simple binary observations makes more sense from the perspective of minimizing opportunities for human error. Hence, the authors conjecture that crowd-sensing will likely gravitate to an application space where binary variables are the commonly measured state, assuming that the algorithmic estimation challenge is solved, which is the purpose of this work.

We denote the set of data sources by $\mathcal{S}$. Time is slotted, such that all the reports generated within time-slot k are timestamped with k. Data is available from multiple time slots. We use a 3D matrix SC to summarize the reports, where $SC_{i,j,k} = v$ means that the source i reports (claims) that variable j has value v in the k-th time-slot. SC is called the *source-claim matrix* in this paper.

In a system with time-varying states, we need to account for state transitions. We aim at a general formulation that is able to support a wide range of crowd-sensing applications. Similarly to what's done in multi-target tracking and hypothesis testing, we translate the dynamic or state transition model of a variable into the joint probability of any given sequence of observed values over a finite time horizon. A different probability is computed for each possible sequence. For example, assuming that a variable has three possible values, a, b, c, and that the finite time horizon has two time-slots, we have $3^2 = 9$ possible sequences (or trajectory hypotheses), each has a probability that is computable from the dynamic system model.

In general, suppose that a variable j has q possible values and we consider a window of H time-slots, then we must consider q^H hypotheses on its possible trajectory. The probability of each hypothesis can be computed from the dynamic state transition model. We call these probabilities the *trajectory probability vector* for variable j. Combining the trajectory probability vectors of all variables, we thus have a *trajectory probability matrix* denoted by $\mathcal{P}$. Note that, the trajectory probability matrix represents prior beliefs that can be computed in advance from the system model. It remains to combine those prior beliefs with received claims of different observers who report values of *some* variables in *some* time slots.

It is not hard to see that the size of $\mathcal{P}$ increases exponentially in H, which prevents us from considering a long history. Fortunately, we only need a relatively short history to estimate current system state within reasonable accuracy as shown in the evaluation, Section VIII. As per our evaluation results, considering the past 5 time-slots leads to a reasonably accurate estimation on the current state, and considering more history actually results in very small increments in estimation accuracy. This is because that the older state has less influence on the current state, and thus can be omitted without much loss of estimation accuracy. Therefore, our state transition formulation is both general and computationally feasible in practical settings.

Let us denote the two possible values of each binary state variable in our model by T and F, respectively. (In Section V, we generalize our model to the multivalued case.) Since in crowd-sensing, participants report state at will, and in no systematic fashion, in the binary case, three values become possible in the source-claim matrix SC, namely, T, F, or U, where U represents *unknown*, meaning "lack of reports". The default value of the source-claim matrix SC is U, which means that we do not assume a default system state.

In contrast to much prior work on state estimation from unreliable sources, we assume not only that source error distribution is not known, but also that each source has a different error distribution. In the case of binary signals, one can summarize the error distribution by a single value, t_i, denoting the reliability of source i. It is defined as the probability that *when i claims that variable j has value v at time j, it is indeed of value v at that time*. Hence, the probability of error is $1 - t_i$. (For multivalued signals, the above is only a partial specification of the error probability distribution since it does not mention how the probability of error is split across possible error values.) Let $C_{j,k}$ denote the value of variable j in time-slot k, and $SC_{i,j,k}$ denote the value of variable j that source i reports in time-slot k. The reliability of source i can be formally defined as $t_i = \Pr(C_{j,k} = v | SC_{i,j,k} = v)$. Let's use a short notation $C^v_{j,k}$ for $C_{j,k} = v$, and $SC^v_{i,j,k}$ for $SC_{i,j,k} = v$, then the source reliability is:

$$t_i = \Pr\left(C^v_{j,k} | SC^v_{i,j,k}\right).$$

Let $T_{i,v}$ denote the probability that source i reports that variable j is in state v given that the variable is really in state v at that time. Let $F_{i,v}$ denote the probability that source i reports that variable j is in state $\bar{v}$ given that j is in state v. Formally, $T_{i,v}$ and $F_{i,v}$ are defined as

$$T_{i,v} = \Pr\left(SC^v_{i,j,k} | C^v_{j,k}\right), F_{i,v} = \Pr\left(SC^{\bar{v}}_{i,j,k} | C^v_{j,k}\right).$$

Note that, $T_{i,v} + F_{i,v} \leq 1$, since it is also possible that the source i does not report the value of the variable. Let u denote the "Unknown" value in the source-claim matrix SC, we have:

$$1 - T_{i,v} - F_{i,v} = \Pr\left(SC^u_{i,j,k} | C^v_{j,k}\right).$$

We denote the prior probability that a source i makes a claim by s_i, and denote the prior probability that any variable at any time is in state v by d^v. By the Bayesian theorem, we have:

$$T_{i,v} = t_i \cdot s_i / d^v, F_{i,v} = (1 - t_i) \cdot s_i / d^v. \quad (1)$$

Our problem can be formulated as follows: *Given the source-claim matrix SC for the past H time-slots, and given the trajectory probability matrix $\mathcal{P}$, jointly estimate both the reliability of each source in $\mathcal{S}$, and the current state of each variable in $\mathcal{C}$.*

IV. Computing Trajectory Probabilities

Before describing our solution to the above problem, in this section, we use two examples (with different state transition models) to illustrate how a trajectory probability matrix $\mathcal{P}$ is computed.

A. Independent State Change

We first start with a simple state transition model where the value of each variable is independent from that of the other variables as well as its history values. For simplicity, we consider binary variables. The multivalued case can be generalized trivially. In this system, two parameters are enough to model state: (1) P_t, the probability that a variable is in state T, and (2) P_f, the probability that a variable is in state F.

Given a value sequence of a variable, we can compute the joint probability of all elements of the sequence easily. For example, the joint probability of a value sequence TTF is simply $P_t^2 P_f$. Therefore, if we use the last H time-slots to estimate the current system state, we can define the trajectory probability matrix $\mathcal{P}$ using 2^H joint probabilities; each joint probability is for one possible sequence of length H.

B. Markov Model

We now consider a system whose state transitions follows a Markov model, in which current state (the values of the variables in the system) is determined only by its last state. For simplicity, again, in this example, the variables are binary. The multivalued variables can be easily generalized. In a Markov model with binary variables, two transition probabilities are enough to describe the system dynamics: (1) P_{tf}, the probability that a variable changes its current state from T to F (in the next time-slot), and (2) P_{ft}, the probability that a variable changes its state from F to T. The probability that a variable remains in the T state in the next time-slot (P_{tt}) can be easily computed by $P_{tt} = 1 - P_{tf}$. Similarly, $P_{ff} = 1 - P_{ft}$.

Given a state trajectory, and the probability of its initial T state P^0_t or F state P^0_f (such that $P^0_t + P^0_f = 1$), we can easily compute its probability. For example, if the trajectory is TTF the joint probability of the state sequence is $P^0_t \cdot P_{tt} \cdot P_{tf}$, where P_{tt} and P_{tf} are the transition probabilities. Therefore, if we exploit the last H time-slot to estimate the current system state, in this model, the trajectory probability matrix $\mathcal{P}$ can be computed using the joint probabilities of 2^H state combinations, where each of the joint probabilities can be easily computed as illustrated above.

The above examples are selected for the ease of illustration. For evaluating trajectory probabilities in the presence of more complex system dynamics, please refer to hypothesis testing and target tracking literature.

V. Dynamic State Estimation

In this section we describe our state estimator for crowd-sensing applications. We adopt a maximum-likelihood estimation framework, and restate the problem as one of finding the set of (i) source reliability values, and (ii) trajectories of state variables that jointly maximize the likelihood of our observations (i.e., received claims). This problem is then solved using the Expectation-Maximization framework [6]. We call the resulting algorithm EM-VTC (Expectation-Maximization algorithm for the time-Varying ground Truth case with Conflicting claims).

A. Deriving a Crowd-sensing State Trajectory Estimator

Expectation-Maximization (EM)[6] is a machine learning algorithm to find the maximum likelihood estimates of parameters in a statistical model when the likelihood function contains latent variables. To apply the EM algorithm, we need to define the likelihood function $L(\theta; x, Z)$[1], where θ is the parameter vector to be estimated, x is the vector of the observed data, and Z is the vector of the latent variables. After defining the likelihood function, EM iteratively applies two steps called the E-step and the M-step until they converge to a solution that computes the values of both the parameter vector and the latent vriables. The mathematical formulation of these iterations is given below:

- E-step: Given the current (estimated) parameter vector and the observed data, compute the expectation of the latent variables.

$$Q(\theta|\theta^{(n)}) = E_{Z|x,\theta^{(n)}}[\log L(\theta; x, Z)]. \tag{2}$$

- M-step: Find the parameters that maximize the Q function defined in the E-step, and use these parameters for the next iteration.

$$\theta^{(n+1)} = \arg\max_{\theta} Q(\theta|\theta^{(n)}) \tag{3}$$

We introduce a latent variable $z_{j,k}$ for each state variable j in time-slot k to denote its estimated value in that time-slot. We use vector z_j to denote the estimated time-series of state variable j in the last H time-slots, where H is a parameter of the algorithm as described in Section III. We use $Z_{j,k}$ to denote the random variable corresponding to $z_{j,k}$, and the $Z_{j,k}$'s, $\forall j \in \mathcal{C}$ and $k \in \{1, 2, \cdots, H\}$, constitute the random matrix Z. We define x to be the 3-dimension source-claim matrix SC, where x_j is the matrix of reported observations of variable j from all sources in $\mathcal{S}$ of all of the H time-slots. Note that, the matrix may be sparse (i.e., containing a lot of "U" values) since many sources will not have observed many variables. We define the parameter set θ to be $\{(T_{i,v}, F_{i,v})|\forall i \in \mathcal{S}, v \in \{True, False\}\}$, where $T_{i,v}$ and $F_{i,v}$ is defined in Equation (1).

The likelihood of receiving the claims reported by all sources in a crowd-sensing application becomes as follows:

$$\begin{aligned} L(\theta; x, Z) &= \prod_{j \in \mathcal{C}} p(x_j, Z_j|\theta) = \prod_{j \in \mathcal{C}} \Big\{ \sum_{z_j \in \Lambda^H} p(x_j, z_j|\theta) \cdot \mathbf{1}_{\{Z_j = z_j\}} \Big\} \\ &= \prod_{j \in \mathcal{C}} \Big\{ \sum_{z_j \in \Lambda^H} p(z_j) \mathbf{1}_{\{Z_j = z_j\}} \cdot \prod_{i \in \mathcal{S}} \prod_{k=1}^{H} \alpha_{i,j,k} \Big\} \end{aligned} \tag{4}$$

where $\Lambda = \{T, F\}$, Λ^H denotes the Cartesian product[2] of the set Λ itself for H times, and $\mathbf{1}_{\{x\}}$ is an indicator function whose value is 1 only if x is true otherwise 0. Please note that $p(z_j)$ is the input (prior) trajectory probability vector (the j-th row of the trajectory probability matrix $\mathcal{P}$), which is independent of the parameters θ. Therefore, $p(z_j) = p(z_j|\theta)$. The $\alpha_{i,j,k}$ is defined as follows:

$$\forall v \in \Lambda, \alpha_{i,j,k} = \begin{cases} T_{i,v} & if \quad z_{j,k} = v, SC_{i,j,k} = v \\ F_{i,v} & if \quad z_{j,k} = v, SC_{i,j,k} = \bar{v} \\ 1 - T_{i,v} - F_{i,v} & if \quad z_{j,k} = v, SC_{i,j,k} = u \end{cases} \tag{5}$$

where u denotes the "Unknown" value U in the source-claim matrix SC.

The derivations of the E-step and M-step are in the Appendix. In the next subsection, we present our algorithm in pseudo code.

B. The EM-VTC Algorithm

The pseudo code of our EM-VTC algorithm is shown in Algorithm 1. The inputs of our algorithm are the source-claim matrix SC with H time-slots, where H is a fixed parameter, and the trajectory probability matrix $\mathcal{P}$ that is learned from history data. Both the source reliability and the estimated variable value in the current time-slot are returned by the algorithm. We estimate the source reliability using Equation (1), where s_i^v can be calculated from the source-claim matrix SC, d^v can be computed by $\sum_{j \in \mathcal{C}} \sum_{k=1}^{H} Z_v^c(j, k)$, and $T_{i,v}^c$ and $F_{i,v}^c$ are calculated after the EM iterations are converged.

VI. Accuracy Guarantees

After developing the EM-VTC algorithm, the next natural question is: How accurate is its estimation results? In this section, we answer the above question by first deriving the Cramer-Rao lower bound (CRLB) for the EM-VTC algorithm and then deriving a confidence interval based on the CRLB. In statistics, the CRLB represents a lower bound on the estimation variance of a deterministic parameter [5]. Note that the CRLB derived here is assuming there are enough sources participating in the crowd-sensing application therefore the truth of the variables are known with full accuracy, thus the CRLB is asymptotic. The derivation of the CRLB is in the appendix.

[1]In this paper, we use capital letters for random variables, such as Z, and use small letters for the values of random variables, such as z.

[2]For example, if $A = \{1, 2\}$ and $B = \{a, b\}$, the Cartesian product of A and B is $A \times B = \{(1, a), (1, b), (2, a), (2, b)\}$.

Algorithm 1 EM-VTC: Exepctation-Maximization Algorithm with Time-Varying Variables

Input: The source-claim matrix SC in the latest H time-slots, and the trajectory probability matrix $\mathcal{P}$.
Output: The estimated values of variables in the current time-slot, and the estimated reliability of each source.

Initialize $\theta^{(0)}$ by setting $T_{i,v}$ and $F_{i,v}$ to random values between 0 and 0.5.
$n \leftarrow 0$
repeat
  for Each $j \in \mathcal{C}$, each $k \in \{1, 2, \cdots, H\}$, and each v in $\{T, F\}$ **do**
    Compute $Z_v^{(n)}(j, k)$ based on Equation (11)
  end for
  for Each $i \in \mathcal{S}$ and each v in $\{T, F\}$ **do**
    Compute $T_{i,v}^*$ anf $F_{i,v}^*$ based on Equation (10)
  end for
  $n \leftarrow n + 1$
until θ^* and $\theta^{(n)}$ converge
$Z_v^c(j, k)$ is the converged value of $Z_v^{(n)}(j, k)$, and $T_{i,v}^c$ is the converged value of $T_{i,v}^{(n)}$, $F_{i,v}^c$ is that of $F_{i,v}^{(n)}$, for every $i \in \mathcal{S}, j \in \mathcal{C}, k \in \{1, 2, \cdots, H\}$ and $v \in \{T, F\}$.
for Each $j \in \mathcal{C}$ **do**
  if $Z_T^c(j, 1) > Z_F^c(j, 1)$ **then**
    Variable j is assigned T in the current time-slot
  else
    Variable j is assigned F in the current time-slot
  end if
end for
for Each $i \in \mathcal{S}$ **do**
  Compute source i's reliability t_i by Equation (1)
end for

A. Confidence Interval of Source Reliability

In this subsection, we derive a confidence interval of source reliability based on the obtained (asymptotic) CRLB (in appendix). Maximum likelihood estimators exhibit several nice properties, one of which is asymptotic normality that the MLE estimator is asymptotically distributed in Gaussian as the data size is large enough [3]:

$$(\hat{\theta}_{MLE} - \theta^0) \rightarrow^{\mathcal{D}} N(0, J^{-1}(\hat{\theta}_{MLE})) \tag{6}$$

where J is the Fisher information matrix as defined in Equation (13), θ^0 and $\hat{\theta}_{MLE}$ are the ground truth and MLE of the parameter θ respectively. In other words, as the data size growing up, the difference between the true value and the MLE of the parameters follows normal distribution with mean 0, and covariance matrix given by the CRLB $J^{-1}(\hat{\theta}_{MLE})$.

The variance of estimation error on parameter $T_{i,T}$ is $J^{-1}(\hat{\theta}_{MLE})_{i,i}$. We know that the reliability of source i is $t_i = \frac{d^T}{s_i^T} T_{i,T}$ by Equation 1. Therefore, by the Δ-method [3], we have the variance of reliability estimation error equals to $(\frac{d^T}{s_i^T})^2 J^{-1}(\hat{\theta}_{MLE})_{i,i}$. We denote this variance by V_i. Therefore, the confidence interval to quantify the source reliability t_i is given as follows:

$$(\hat{t}_i^{MLE} - c_p \cdot \sqrt{V_i}, \hat{t}_i^{MLE} + c_p \cdot \sqrt{V_i}) \tag{7}$$

where c_p is the standard score of the confidence level p. For example, for the 95% confidence level, $c_p = 1.96$.

VII. Multivalued Variable Extension

In this section, we extend our EM-VTC algorithm from the binary case to a general multivalued case, where each variable has q (≥ 2) possible values. Although Wang et al. [34] designed an EM algorithm that takes multivalued variables for the static state case, we found that their algorithm is not suitable in the time-varying state case. The main reason is that the time complexity of *each* EM iteration in their algorithm is $O(q^H)$, if the last H time-slots are considered in estimating the current system state. Please note that the EM iterations are the major time-consuming part of an algorithm under the EM framework. Therefore, for a large q, the heavy computational overhead of their algorithm makes it not practically applicable, especially in time sensitive systems. One of our goal is to time-efficiently extend our binary solution for the multivalued case. Specifically, we require a solution in the q-valued case in which the time complexity of each EM iteration grows *no faster than linearly* in q compared with the binary solution.

The pseudo code is shown in Algorithm 2. Our high-level idea is to reduce the multivalued case to a binary case. Suppose that in the multivalued case the value set $\Lambda = \{\lambda_1, \lambda_2, \cdots, \lambda_q\}$. We first construct q new binary variables for each q-nary variable (line 1 to line 3). Next, we construct the source-claim matrix SC_m^b for the binary variables corresponding to each q-nary variable m based on the SC (line 4 to line 12). We then construct the trajectory probability matrix $\mathcal{P}_m^b$ for the constructed binary variables corresponding to each multivalued variable m from the trajectory probability matrix $\mathcal{P}$ that is an input of our algorithm (line 13 to line 20). The combinations of SC_m^b and $\mathcal{P}_m^b$ are denoted as SC^b and $\mathcal{P}^b$ respectively. Next, we apply Algorithm 1 with inputs SC^b and $\mathcal{P}^b$ to get the source reliability and the converged Z_T^c value of each of the binary variables (line 21). Please note that here we do not estimate the state of each binary variable, but only exploit the converged Z_T^c values. Finally, we assign each of the q-nary variables with the value whose corresponding binary variable has the highest Z_T^c value among all the binary variables corresponding to the q-nary variable (line 22 to line 25).

Please note that here the time complexity for constructing the trajectory probability matrix for the binary variables is $O(q^{H+1})$, and this procedure is executed only *once*. With the constructed trajectory probability matrix $\mathcal{P}^b$ and the source-claim matrix SC^b, each EM iteration in Algorithm 1 has the time complexity $O(q \cdot 2^H)$. Therefore, we observe that the computational complexity for the multivalued case increases almost *linearly* in the number of possible values that each variable can take (i.e. q).

VIII. Evaluation

In this section, we evaluate the performance of our algorithm compared with other state-of-the-art solutions and a simple baseline algorithm. We first study the performance in a simulation study, then we evaluate our algorithm in a real-world crowd-sensing application.

A. Simulation Study

1) Methodology: We build a crowd-sensing simulator in Matlab R2013b. In the simulation, 200 binary variables are created whose initial values are assigned randomly. Each

Algorithm 2 EM-VTC for multivalued variables

Input: The source-claim matrix SC in the last H time-slots, the trajectory probability matrix $\mathcal{P}$

Output: The estimation of source reliability and the current variable values

for Each q-nary variable $j \in \mathcal{C}$ **do**
  Construct q binary variables $j_1, j_2, \cdots, j_q$, such that $j_i = T$ if $j = \lambda_i$ and F otherwise.
end for
Allocate the memory for the source-claim matrix SC^b_m for the new binary variables corresponding to each q-nary variable m. Totally, allocate SC^b with size $|\mathcal{S}| \times q \cdot |\mathcal{C}|$, and initiate SC^b with the "Unknown" value U.
for Each $i \in \mathcal{S}, j \in \mathcal{C}, k \in \{1, 2, \cdots, H\}$ **do**
  if $SC_{i,j,k} = \lambda_m$ **then**
    $SC^b_{i,j_m,k} \leftarrow T$
    for Each $m' \in \{1, \cdots, m-1, m+1, \cdots, q\}$ **do**
      $SC^b_{i,j_{m'},k} \leftarrow F$
    end for
  end if
end for
Allocate the memory for the trajectory probability matrix $\mathcal{P}^b_m$ for the new binary variables corresponding to each q-nary variable m. Totally, allocate $\mathcal{P}^b$ with size $2^H \times q$, and initiate $\mathcal{P}^b$ by 0.
for Each q-nary variable $j \in \mathcal{C}$, each value $\lambda_m \in \Lambda$ **do**
  for Each element γ of $\mathcal{P}_j$ **do**
    $\triangleright$ (Comment: γ is some combination of H q-nary values, and $\mathcal{P}_j$ is a column vector.)
    Compute the corresponding combination of H binary values of variable j_m and the index γ^b in $\mathcal{P}^b_{j_m}$.
    $\mathcal{P}^b_{j_m} \leftarrow \mathcal{P}^b_{j_m} + \mathcal{P}_j$.
  end for
end for
Use Algorithm 1 with SC^b and $\mathcal{P}^b$ to get the source reliability t_i for each $i \in \mathcal{S}$ and $Z^c_T(j_m, 1)$ for each $j \in \mathcal{C}, m \in \{1, \cdots, q\}$
for Each $j \in \mathcal{C}$ **do**
  $m \leftarrow \arg\max_{m=1}^{q} Z^c_T(j_m, 1)$
  Variable j is assigned with value λ_m.
end for

variable represents a physical event with state T or F. The initial value of each variable is distributed uniformly at random (i.e., with probability 0.5 the value is assigned to T and 0.5 it is assigned to F). For transition probabilities, we assume a two state Markov model. That is, the value of each variable in one time-slot depends only on its value in the preceding one. There are only two states, T and F. The transition probability from T to T is P_{tt}, and from F to F is P_{ff}. These two parameters are enough to determine the other two transition probabilities in the Markov model. While we could have considered more complex and realistic systems, our goal from considering the two-state Markov model was to help understand the fundamental performance trends of our state estimator as a function of parameters of the system model. Results for more complex models would have been harder to interpret due to the multitude of confounding factors at play.

For the sources, the simulator also choose a reliability t_i for each source i. We set the reliability of each source randomly distributed in $[0.5, 1)$. In the simulation, each source is assigned a probability of making claims, s_i, meaning the probability that a source reports an observation. The higher the s_i is, the more "talkative" the source is.

The default values of the parameters are as follows: the number of sources is 30, the expected source reliability $E(t_i) = 0.6$, the factor $s_i = 0.6$, the number of history time-slots to be considered $H = 5$, the state transition probabilities $P_{tt} = P_{ff} = 0.5$, and the initial ground bias $d^T_j = 0.5$ denoting the probability that a variable is assigned T initially.

We compare our algorithm **EM-VTC** with two state-of-the-art algorithms proposed in [35] and [34]. The algorithm proposed in [35] does not consider state changes. It assumes that the default physical state of each variable is "F", allowing sources to report only "T" values of the observed variables. The algorithm proposed in [34] extends the above algorithm by considering conflicting claims (i.e., both "T" and "F" values). However, it still assumes that system state is immutable. For each of the two algorithm, we further consider two cases: (1) Applying the algorithm with the data in the current time-slot, and (2) Applying the algorithm with the data in the last H time-slots. The algorithm in [35] is denoted by **EM-R1** when it is fed with the data of the current time-slot, and by **EM-Rall** when it is fed with the data of the last H time-slots. The algorithm in [34] is denoted by **EM-C1** when used in the current time-slot, and by **EM-Call** when used in the last H time-slots.

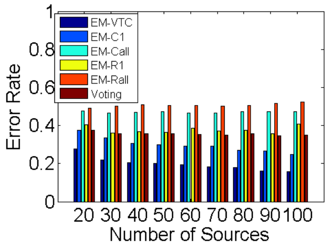
We also compare our algorithm to a simple baseline algorithm **Voting**. Voting estimates the variable to be equal to the majority vote (i.e., most frequently reported value at the time). Each simulation runs 100 times and each result is averaged on the 100 executions.

2) Evaluation Results: In Figure 2, we evaluate the performance of our algorithm as the number of sources varies from 20 to 100. Other parameters are set to the default values. From Figure 2(a), we can observe that the false estimation rate of our algorithm EM-VTC is the smallest among all the algorithms. The EM-Call and EM-Rall algorithms are the worst in estimating the variable values, since they do not consider the fact that the physical state of each variable changes in time. Without considering the time-varying states, it is very unlikely to correctly estimate the source reliability, as shown in Figure 2(b). The reason is that the reports from a 100% reliable source might look "self-conflicting" to EM-Call and EM-Rall in a system with time-varying states, because they assume the system state is immutable. Therefore, the EM-Call and EM-Rall assigns a relatively low reliability to the 100% reliable source.

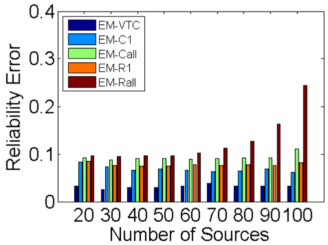
EM-C1 and EM-R1 performs worse than EM-VTC, because they use only the data in the current time-slot, while the EM-VTC algorithm uses all the data in the last H time-slots. With more data, the source reliability can be learned better as shown in Figure 2(b), which in turn results a better estimation of variable values. EM-C1 outperforms than EM-R1, because EM-R1 does not distinguish "Unknown" from "False", thus EM-R1 gives the F state a higher weight (so higher false negative rate).

We can also observe from Figure 2, as the number of sources increases, the estimation of EM-VTC becomes better and better. The reason is that more sources means potentially less "Unknown" values in the source-claim matrix SC. Therefore, estimation becomes more accurate.

Figure 3 shows the performance as the factor s_i (probability of reporting) varies from 0.2 to 0.8. Other parameters are set to the default values. As s_i increases, the estimation error of EM-VTC becomes smaller as shown in Figure 3(a). With less unknown values in the source-claim matrix, the EM algorithm

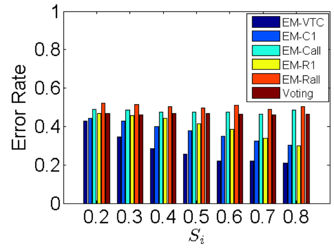


(a) Variable state estimation

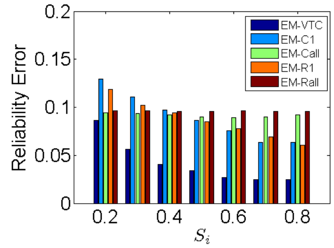


(b) Source reliability estimation

Fig. 2. Performance as the number of sources varies.

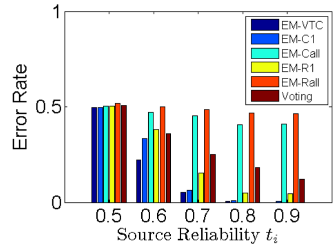


(a) Variable state estimation

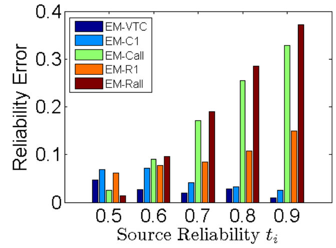


(b) Source reliability estimation

Fig. 3. Performance as the talkative factor s_i of sources varies.



(a) Variable state estimation

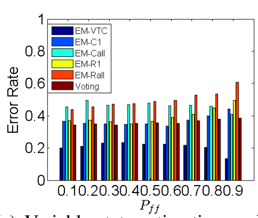


(b) Source reliability estimation

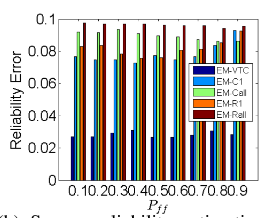
Fig. 4. Performance as the source reliability t_i varies.

can jointly estimate the source reliability (Figure 3(b)) and variable value (Figure 3(a)) more accurately. In Figure 3, we show that EM-VTC outperforms all the baselines in source reliability estimation as well.

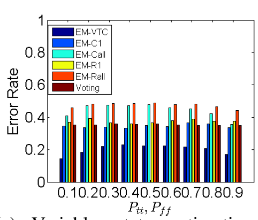
Figure 4 shows the performance as the expected source reliability $E(t_i)$ changes from 0.5 to 0.9. As the expected reliability increases, our algorithm performs better. When the expected source reliability is 0.5, the sources essentially make random reports, offering no information. Therefore, error rate is around 50%. However, when source reliability increases, our EM-VTC algorithm outperforms all the others. When the source reliability is 90%, our algorithm actually estimates the values of variables 100% correctly.



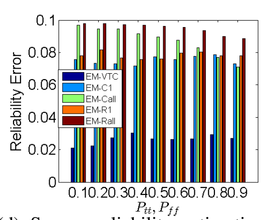
(a) Variable state estimation as P_{ff} varies



(b) Source reliability estimation as P_{ff} varies



(c) Variable state estimation as P_{tt}, P_{ff} vary



(d) Source reliability estimation as P_{tt}, P_{ff} vary

Fig. 5. Performance as the state transit probability varies.

In Figure 5(a) and (b), we evaluate how the *single-sided* system dynamics affect the performance of our algorithm. In this experiment, the probability of staying in one state, P_{tt}, is fixed at 0.5, and the probability of staying in the other P_{ff} varies from 0.1 to 0.9, emulating how "sticky" that state is. Figure 5(a) and (b) shows that our algorithm consistently performs the best no matter what value that P_{ff} takes in terms of both the variable value estimation and the source reliability estimation.

In Figure 5(c) and (d), we set $P_{tt} = P_{ff}$ and vary both from 0.1 to 0.9. The other parameters are set to the default values. Please note that since the initial ground bias d_j^T is set to 0.5, in this simulation the expected number of the T variables and the expected number of the F variables will always be the same, although the switching frequency changes. Again, our algorithm consistently performs the best.

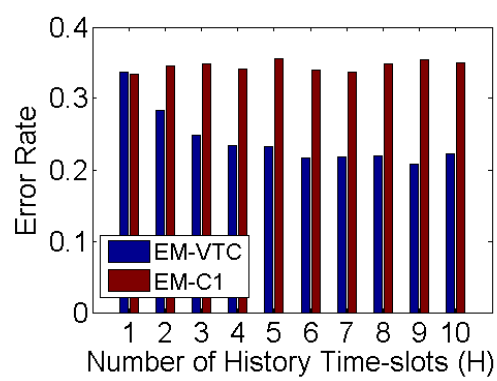


Fig. 6. Variable value estimation performance as the number of history time-slots being considered varies.

In Figure 6, we evaluate the performance of our EM-VTC algorithm when the number of time-slots, H, in the sliding window considered for prediction varies from 1 to 10. Here we compare EM-VTC with EM-C1 that only considers the current time-slot. As shown in Figure 6, when we consider more history, the estimation is more accurate. However, the *marginal increase* in the estimation accuracy decreases.

In Figure 7 we study the performance bounds of our EM-VTC algorithm in terms of the confidence intervals (CI), which aims at validating our derived CRLB. In this experiment, we set the number of sources to 40, and the rest of the parameters to their default values. The area between the red solid lines is a 90% confidence interval of the estimators, and that between the blue dashed lines is a 95% confidence interval. The confidence intervals are computed from the CRLB as derived in Section VI. From CRLB, we can compute the variances of the false positive $F_{i,F}$ ($Var(F_{i,F}^{MLE})$) and false negative $F_{i,T}$ ($Var(F_{i,T}^{MLE})$) respectively. By Equation 6, the 90% confidence interval for the false negative is $F_{i,T}^{MLE} \pm 1.65 Var(F_{i,T}^{MLE})$, and the 95% confidence interval for the false negative is $F_{i,T}^{MLE} \pm 1.96 Var(F_{i,T}^{MLE})$. We can also compute the confidence intervals for the source reliability estimator. Please note that a 90% confidence interval means that no more than 10% of the sample points are outside the confidence interval with high probability, while a 90% confidence interval means that no more than 5% of the sample points are not included in that interval with high probability.

From Figure 7(a), we observe that 3 sample points are out of the 90% CI, which validates our confidence interval since we allow no more than 4 points out of the 90% CI. For the 95% CI, all points are within it, which validates our confidence interval again.

In Figure 7(b), 1 points falls out of the 90% CI and no points falls out of the 95% CI. In Figure 7(c), 3 points fall out of the 90% CI and no points falls out of the 95% CI. Therefore, our confidence intervals are computed correctly, which also validates the derived CRLB.

We also run the above simulation for 100 times, and compute the average percentage of "bad" sample points among the whole sample, where "bad" sample points means the points fall out of the corresponding confidence interval. The results are shown in the Table I, which also validates our confidence intervals and thus the derived CRLB.

TABLE I
PERCENTAGE OF SAMPLE POINTS FALLING OUT OF THE CORRESPONDING CONFIDENCE INTERVAL FOR DIFFERENT ESTIMATORS.

	Reliability	False negative	False positive
90% CI	0.0648	0.0807	0.0865
95% CI	0.0093	0.0138	0.0158

B. A Real-world Case Study

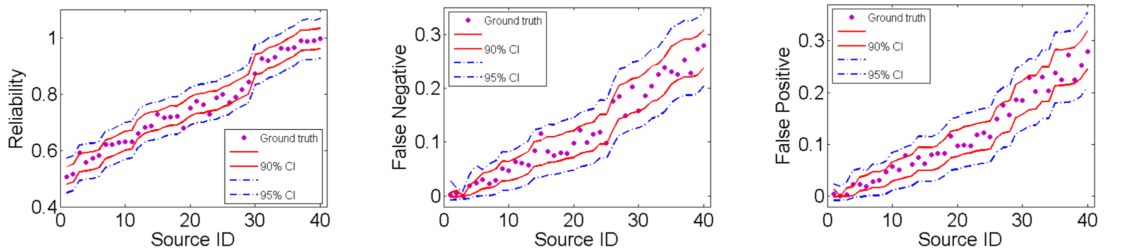
In this subsection, we study the performance of our algorithm through a real-world crowd-sensing application that aims at assisting drivers to find the available street parking spots nearby. Recent work on assisting drivers in street parking relies on pre-deployed infrastructures (e.g. smart meters with communication capability to a remote center server [20]), or relies on special sensors to be embedded on cars (e.g. ultrasonic sensor on the side of a car [19]). We want to test whether a zero-infrastructural overhead crowd-sensing approach can be used in this application with the help of our algorithm.

In this case study, we recorded the availabilities of metered parking spots on two streets near the department of Computer Science at University of Illinois Urbana-Champaign for 8 days (from April 11th, 2014 to April 18th, 2014) using webcams. 30 participants were involved in this case study to report the availabilities of the parking spots on the two streets from $2pm$ to $6pm$ on April 18th, 2014, and 2833 reports were collected. We deliberately picked the period from $2pm$ to $6pm$ is because in this period of time the availability of parking spots around the department of Computer Science has some clear time-varying pattern; students drive to attend the afternoon classes, and leave school for dinner around $5pm$, then some of them drive back to school for self studies after dinner. The time-varying ground truth of the parking availability is shown in Figure 8.

We first define the time-slot, and define the availability (occupied/avaiable) of a parking spot corresponding the slotted time. We partition the time into slots such that each time-slot is 10 minutes. Therefore, during our experiment on April 18th, there are 24 time-slots totally. Each parking spot is assigned with a corresponding variable with values T (representing that it is occupied) and F (representing that it is available). There are totally 15 parking spots in our experiment, therefore 15 variables are defined. In each time-slot, if more than half of the time a parking spot is occupied and the corresponding variable is assigned T, else the parking spot is defined free and the corresponding variable is assigned F.

Next, we define the source-claim matrix SC, and the trajectory probability matrix $\mathcal{P}$ that are the inputs of our algorithm. Each report from the participants is associated with the time-slot during which it is generated. Both the sources and the events (variables) are indexed by integers. Therefore, we can naturally define the source-claim matrix SC to contain all the reports. In our case study, we only consider past two time-slots when estimating the variable values in the current time-slot, which means that the H is set 2 here. The state transition is modeled by the trajectory probability matrix $\mathcal{P}$ where each element is a joint probability of the current value and the last time-slot value of a variable. The joint probabilities are learned from history data between 2pm and 6pm from April 11th to April 17th. Please note that we did not use the data on April 18th to learn the trajectory probability matrix. In this case study, we set all the variable share the same trajectory probability matrix $\mathcal{P}$. Please note that this is not exactly accurate, since different variables might have different dynamics behaviors. However, our evaluation results show that our EM-VTC algorithm still achieves an accurate estimation under this imperfect setting.

We compare with 4 baseline algorithms: (1) **EM-C1** as described in the previous subsection, (2) **EM-R1** that is also described previous, (3) **Voting**, and (4) **History** where we only use the state in the last time-slot to estimate the current state for each variable. The experiment works in a "sliding-window" style: We start from estimating the variable values



(a) CI for source reliability estimation (b) CI of false negatives for variable value estimation (c) CI of false positives for variable value estimation

Fig. 7. Performance bounds of EM-VTC in terms of confidence intervals.

of the second time-slot (2:10pm-2:20pm) using the data (the source reports) in the first time-slot and the second time-slot, then move on to estimate the system state in the third time-slot (2:20pm-2:30pm) using data from the second and the third time-slots, and so on.

The ground truth is shown in Figure 8. From Figure 8, we can observe that the system state (the availability of parking spots) did change overtime. During the dinner time (from 5:00pm to 6:00pm), the system state changes more frequently than that during the working time (before 5:00pm).

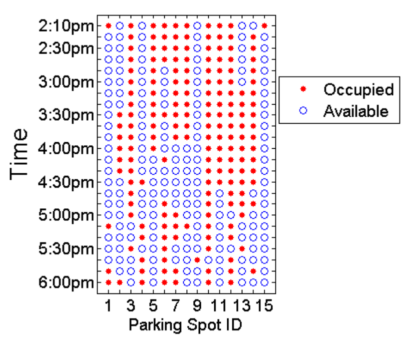


Fig. 8. The ground truth of the availability of parking spots.

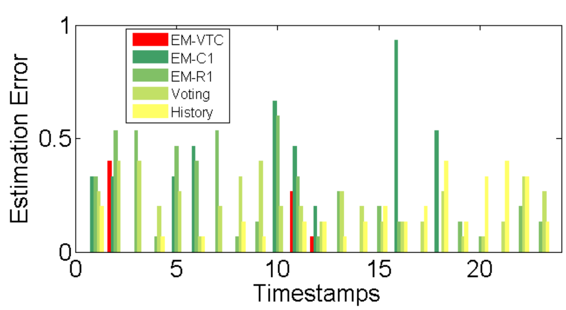


Fig. 9. The performance of variable state estimation.

The estimation results are presented in Figure 9. In the figure, the x-axis represents the time-slots that are indexed using integers, and the y-axis is the estimation error. The red bar shows the estimation error of our EM-VTC algorithm. From Figure 9, we hardly observe the red bars, which shows that our solution in most of the time has *zero* estimation errors. The average errors are summarized in Table II.

TABLE II
ESTIMATION ERROR IN PARKING CASE STUDY

Algorithm	EM-VTC	EM-C1	EM-R1	Voting	History
Est. Error	**0.0319**	0.1855	0.2261	0.2203	0.1420

Please note that the EM-C1 and EM-R1 algorithms perform worse than the History heuristic that using the variable value in the last time-slot to estimate its current value, which is because that our parking data has high correlation between the last time-slot and the current time-slot as shown in Figure 8, and that those two algorithms only use the data in the current time-slot which is not enough to estimate the system state with high accuracy. Our algorithm uses more data and considers the system state trajectory, therefore performs the best.

IX. Discussion and Future Work

In this work, the source reliability is assumed to be static, that is a reliable source consistently reports what he/she has truly observed while an unreliable source reports almost randomly at all time. A better model would assume a time-varying reliability. For example, a source might report the availability of a parking spot more accurately when it is close to that parking spot than when it is far away. Considering the mobility of the data sources, we can no longer assume that the reliability is static. An interesting future work would be how to solve the data reliability problem in CPS when the source reliability is time-varying. In this work, the trajectory probability matrix $\mathcal{P}$ is computed in advance and input into the EM algorithm. It might be better to jointly compute the trajectory probability while addressing the data reliability challenges, since the trajectory probability matrix might also change during the computation, especially when the evolution of the physical environment is modeled by a set of complicated differential equations. We left it as a future work as well.

X. Conclusions

In this paper, we developed a state estimator for crowd-sensing applications, where humans act as data sources reporting observed variables in a dynamic environment. We demonstrated how a model of environmental dynamics can significantly enhance our ability to estimate correct state even though the reliability of observers is not known. We also analytically studied its performance by deriving a Cramer-Rao lower bound for our estimator. Our solution was evaluated using both simulations and a real-world crowd-sensing application. The results show that the solution outperforms prior approaches that do not properly account for differences in source reliability (e.g., voting) or do not properly leverage knowledge of the system model.

ACKNOWLEDGMENTS

Research reported in this paper was sponsored by the Army Research Laboratory and was accomplished under Cooperative Agreement W911NF-09-2-0053, DTRA grant HDTRA1-10-10120, and NSF grants CNS 09-05014 and CNS 10-35736. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either expressed or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes notwithstanding any copyright notation here on.

REFERENCES

[1] T. Abdelzaher, Y. Anokwa, P. Boda, J. A. Burke, D. Estrin, L. Guibas, A. Kansal, S. Madden, and J. Reich. Mobiscopes for human spaces. *Center for Embedded Network Sensing*, 2007.

[2] J. A. Burke, D. Estrin, M. Hansen, A. Parker, N. Ramanathan, S. Reddy, and M. B. Srivastava. Participatory sensing. 2006.

[3] G. Casella and R. L. Berger. *Statistical inference*, volume 70. Duxbury Press Belmont, CA, 1990.

[4] E. M. Clarke, B. Krogh, A. Platzer, and R. Rajkumar. Analysis and veri cation challenges for cyber-physical transportation systems. In *National Workshop for Research on High-confidence Transportation Cyber-Physical Systems: Automotive, Aviation & Rail*, 2008.

[5] H. Cramér. *Mathematical methods of statistics*, volume 9. Princeton university press, 1999.

[6] A. P. Dempster, N. M. Laird, D. B. Rubin, et al. Maximum likelihood from incomplete data via the em algorithm. *Journal of the Royal statistical Society*, 39(1):1–38, 1977.

[7] J. C. Eidson, E. A. Lee, S. Matic, S. A. Seshia, and J. Zou. Distributed real-time software for cyber–physical systems. *Proceedings of the IEEE*, 100(1):45–59, 2012.

[8] S. B. Eisenman, E. Miluzzo, N. D. Lane, R. A. Peterson, G.-S. Ahn, and A. T. Campbell. Bikenet: A mobile sensing system for cyclist experience mapping. *ACM Transactions on Sensor Networks (TOSN)*, 6(1):6, 2009.

[9] A. Faza, S. Sedigh, and B. McMillin. Integrated cyber-physical fault injection for reliability analysis of the smart grid. In *Computer Safety, Reliability, and Security*, pages 277–290. Springer, 2010.

[10] J.-H. Huang, S. Amjad, and S. Mishra. Cenwits: a sensor-based loosely coupled search and rescue system using witnesses. In *SenSys*, 2005.

[11] M. Huang, J. Li, X. Song, and H. Guo. Modeling impulsive injections of insulin: Towards artificial pancreas. *SIAM Journal on Applied Mathematics*, 72(5):1524–1548, 2012.

[12] B. Hull, V. Bychkovsky, Y. Zhang, K. Chen, M. Goraczko, A. Miu, E. Shih, H. Balakrishnan, and S. Madden. Cartel: a distributed mobile sensor computing system. In *SenSys*, 2006.

[13] P. Ishwar, R. Puri, K. Ramchandran, and S. S. Pradhan. On rate-constrained distributed estimation in unreliable sensor networks. *Selected Areas in Communications, IEEE Journal on*, 23(4):765–775, 2005.

[14] J. M. Kleinberg. Authoritative sources in a hyperlinked environment. *Journal of the ACM (JACM)*, 46(5):604–632, 1999.

[15] N. D. Lane, S. B. Eisenman, M. Musolesi, E. Miluzzo, and A. T. Campbell. Urban sensing systems: opportunistic or participatory? In *HotMobile*, 2008.

[16] E. A. Lee. Cyber physical systems: Design challenges. In *ISORC*, 2008.

[17] J. Lu, T. Sookoor, V. Srinivasan, G. Gao, B. Holben, J. Stankovic, E. Field, and K. Whitehouse. The smart thermostat: using occupancy sensors to save energy in homes. In *SenSys*, 2010.

[18] E. Masazade, R. Niu, P. K. Varshney, and M. Keskinoz. A probabilistic transmission scheme for distributed estimation in wireless sensor networks. In *Information Sciences and Systems (CISS), 2010 44th Annual Conference on*, pages 1–6. IEEE, 2010.

[19] S. Mathur, T. Jin, N. Kasturirangan, J. Chandrasekaran, W. Xue, M. Gruteser, and W. Trappe. Parknet: drive-by sensing of road-side parking statistics. In *MobiSys*, 2010.

[20] S. Nawaz, C. Efstratiou, and C. Mascolo. Parksense: a smartphone based sensing system for on-street parking. In *MobiCom*, 2013.

[21] J. Pasternack and D. Roth. Knowing what to believe (when you already know something). In *COLING*, 2010.

[22] L. Schenato, B. Sinopoli, M. Franceschetti, K. Poolla, and S. S. Sastry. Foundations of control and estimation over lossy networks. *Proceedings of the IEEE*, 95(1):163–187, 2007.

[23] G. Schirner, D. Erdogmus, K. Chowdhury, and T. Padir. The future of human-in-the-loop cyber-physical systems. *Computer*, 46(1):36–45, 2013.

[24] Sense Networks. Cab Sense. http://www.cabsense.com.

[25] L. Sha and J. Meseguer. Design of complex cyber physical systems with formalized architectural patterns. In *Software-Intensive Systems and New Computing Paradigms*, pages 92–100. Springer, 2008.

[26] B. Sinopoli, L. Schenato, M. Franceschetti, K. Poolla, M. I. Jordan, and S. S. Sastry. Kalman filtering with intermittent observations. *Automatic Control, IEEE Transactions on*, 49(9):1453–1464, 2004.

[27] L. Su, J. Gao, Y. Yang, T. F. Abdelzaher, B. Ding, and J. Han. Hierarchical aggregate classification with limited supervision for data reduction in wireless sensor networks. In *Proceedings of the 9th ACM Conference on Embedded Networked Sensor Systems*, pages 40–53. ACM, 2011.

[28] L. Su, S. Hu, S. Li, F. Liang, J. Gao, T. F. Abdelzaher, and J. Han. Quality of information based data selection and transmission in wireless sensor networks. In *RTSS*, pages 327–338, 2012.

[29] L. Su, Q. Li, S. Hu, S. Wang, J. Gao, H. Liu, T. Abdelzaher, J. Han, X. Liu, Y. Gao, and L. Kaplan. Generalized decision aggregation in distributed sensing systems. In *Real-Time Systems Symposium (RTSS), 2014 IEEE 35th*. IEEE, 2014.

[30] D. Wang, T. Abdelzaher, L. Kaplan, and C. C. Aggarwal. Recursive fact-finding: A streaming approach to truth estimation in crowdsourcing applications. In *ICDCS*, 2013.

[31] D. Wang, T. Abdelzaher, L. Kaplan, R. Ganti, S. Hu, and H. Liu. Exploitation of physical constraints for reliable social sensing. In *RTSS*, 2013.

[32] D. Wang, T. Amin, S. Li, T. A. L. Kaplan, S. G. C. Pan, H. Liu, C. Aggrawal, R. Ganti, X. Wang, P. Mohapatra, B. Szymanski, and H. Le. Humans as sensors: An estimation theoretic perspective. In *IPSN*, 2014.

[33] D. Wang, L. Kaplan, T. Abdelzaher, and C. C. Aggarwal. On scalability and robustness limitations of real and asymptotic confidence bounds in social sensing. In *SECON*, 2012.

[34] D. Wang, L. Kaplan, and T. F. Abdelzaher. Maximum likelihood analysis of conflicting observations in social sensing. *ACM Transactions on Sensor Networks (TOSN)*, 10(2):30, 2014.

[35] D. Wang, L. Kaplan, H. Le, and T. Abdelzaher. On truth discovery in social sensing: a maximum likelihood estimation approach. In *IPSN*, 2012.

[36] S. Wang, T. Abdelzaher, S. Gajendran, A. Herga, S. Kulkarni, S. Li, H. Liu, C. Suresh, A. Sreenath, W. Dron, et al. Poster abstract: information-maximizing data collection in social sensing using named-data. In *Proceedings of the 13th international symposium on Information processing in sensor networks*, pages 303–304. IEEE Press, 2014.

[37] S. Wang, T. Abdelzaher, S. Gajendran, A. Herga, S. Kulkarni, S. Li, H. Liu, C. Suresh, A. Sreenath, H. Wang, W. Dron, A. Leung, R. Govindan, and J. Hancock. The information funnel: Exploiting named data for information-maximizing data collection. In *Proceedings of the 10th IEEE International Conference on Distributed Computing in Sensor Systems*. IEEE Press, 2014.

[38] S. Wang, S. Hu, S. Li, H. Liu, M. Y. S. Uddin, and T. Abdelzaher. Minerva: Information-centric programming for social sensing. In *Computer Communications and Networks (ICCCN), 2013 22nd International Conference on*, pages 1–9. IEEE, 2013.

[39] X. Yin, J. Han, and P. S. Yu. Truth discovery with multiple conflicting information providers on the web. *Knowledge and Data Engineering, IEEE Transactions on*, 20(6):796–808, 2008.

Appendix

A. Deriving the E-step

We plug-in the likelihood function, given by Equation (4), into Equation (2) to derive the E-step.

$$
\begin{aligned}
Q(\theta|\theta^{(n)}) &= E_{Z|x,\theta^{(n)}}[\log L(\theta;x,Z)] \\
&= \sum_{j\in\mathcal{C}} E_{Z_j|x_j,\theta^{(n)}}\left[\log\Big\{\sum_{z_j\in\Lambda^H} p(z_j)\mathbf{1}_{\{Z_j=z_j\}}\prod_{i\in\mathcal{S}}\prod_{k=1}^{H}\alpha_{i,j,k}\Big\}\right] \\
&= \sum_{j=\in\mathcal{C}}\sum_{z_j\in\Lambda^H} p(z_j|x_j,\theta^{(n)})\left\{\log p(z_j)+\sum_{i\in\mathcal{S}}\sum_{k=1}^{H}\log\alpha_{i,j,k}\right\}
\end{aligned}
\tag{8}
$$

, where

$$
\begin{aligned}
p(z_j|x_j,\theta^{(n)}) &= \frac{p(x_j,z_j|\theta^{(n)})}{p(x_j|\theta^{(n)})} \\
&= \frac{p(x_j|z_j,\theta^{(n)})\cdot p(z_j|\theta^{(n)})}{\sum_{z_j\in\Lambda^H} p(x_j|z_j,\theta^{(n)})\cdot p(z_j|\theta^{(n)})} \\
&= \frac{p(z_j)\prod_{i\in\mathcal{S}}\prod_{k=1}^{H}\alpha_{i,j,k}^{(n)}}{\sum_{z_j\in\Lambda^H} p(z_j)\prod_{i\in\mathcal{S}}\prod_{k=1}^{H}\alpha_{i,j,k}^{(n)}}.
\end{aligned}
\tag{9}
$$

Please note that in the Q function, the parameters are embedded into the $\alpha_{i,j,k}$ that is assigned different values under different (i,j,k) settings as defined in Equation (5). Also note that $p(z_j|x_j,\theta^{(n)})$ is a constant with respect to θ.

B. Deriving the M-step

In the M-step, we set the partial derivatives of the Q function to 0 to get the θ^* that maximizes the value of Q. That is by solving $\frac{\partial}{\partial T_{i,v}}Q = 0$, and $\frac{\partial}{\partial F_{i,v}}Q = 0$, we can get $T^*_{i,v}$ and $F^*_{i,v}$, for each $i\in\mathcal{S}$ and $v\in\{T,F\}$.

$$
\begin{aligned}
T^*_{i,v} &= \frac{\sum_{j\in\mathcal{C}}\sum_{k:SC_{i,j,k}=v} Z_v^{(n)}(j,k)}{\sum_{j\in\mathcal{C}}\sum_{k=1}^{H} Z_v^{(n)}(j,k)} \\
F^*_{i,v} &= \frac{\sum_{j\in\mathcal{C}}\sum_{k:SC_{i,j,k}=\bar{v}} Z_v^{(n)}(j,k)}{\sum_{j\in\mathcal{C}}\sum_{k=1}^{H} Z_v^{(n)}(j,k)}
\end{aligned}
\tag{10}
$$

, where $Z_v^{(n)}(j,k)$ is defined as:

$$
Z_v^{(n)}(j,k) = \sum_{z_j:z_{j,k}=v} p(z_j|x_j,\theta^{(n)}). \tag{11}
$$

The parameter θ^* is used in our EM-VTC algorithm as shown in Algorithm 1.

C. Deriving Cramer-Rao Lower Bound

By definition, CRLB is the inverse of the Fisher information matrix $J(\theta)$, where $J(\theta) = E_X[\nabla_\theta \log p(X|\theta)\nabla_\theta^H \log p(X|\theta)]$. Here matrix X^H is the conjugate transpose of matrix X. The CRLB derived in this subsection is asymptotic by assuming that the values of variables are correctly estimated by the EM algorithm. This assumption is valid when the number of sources is high, as shown in [33]. The loglikehood function under this assumption is $\ell_{em}(\theta;x)$, as defined in Equation (12).

$$
\begin{aligned}
\ell_{em}(\theta;x) = \sum_{j\in\mathcal{C}}\sum_{k=1}^{H}\Bigg\{&\mathbf{1}_{\{z_{j,k}=T\}}\Bigg(\sum_{i\in\mathcal{S}}\Big(\mathbf{1}_{\{x_{i,j,k}=T\}}\log T_{i,T} \\
&+\mathbf{1}_{\{x_{i,j,k}=F\}}\log F_{i,T}+\mathbf{1}_{\{x_{i,j,k}=U\}}\log(1-T_{i,T}-F_{i,T})\Big)\Bigg) \\
&+\mathbf{1}_{\{z_{j,k}=F\}}\Bigg(\sum_{i\in\mathcal{S}}\Big(\mathbf{1}_{\{x_{i,j,k}=F\}}\log T_{i,F}+\mathbf{1}_{\{x_{i,j,k}=T\}}\log F_{i,F} \\
&+\mathbf{1}_{\{x_{i,j,k}=U\}}\log(1-T_{i,F}-F_{i,F})\Big)\Bigg)\Bigg\}.
\end{aligned}
\tag{12}
$$

We compute the Fisher information matrix from its definition, similar as [33]. The parameters in the loglikelihood is order such that the $T_{i,T}$'s come first, then $T_{i,F}$'s, following by $F_{i,T}$'s, and finally $F_{i,F}$'s. Let denote the number of sources by M, so $M = |\mathcal{S}|$. So, we have $4M$ parameters in total, and the size of the Fisher information matrix is $4M\times 4M$. Let's denote the n-th parameter by θ_n. For example, when $n\le M$, $\theta_n = T_{i,T}$ where $i=n$, and when $n\in[M{+}1,2M], \theta_n = T_{i,F}$ where $i = n-M$, so on. We denote the total number of variables by N, i.e., $N = |\mathcal{C}|$. Let $J(\theta)_{m,n}$ denote the element in the m-th row and j-th column of the Fisher information matrix $J(\theta)$. The Fisher information is defined in Equation (13).

$$
\begin{aligned}
J(\theta)_{m,n} &= \begin{cases}
-E[\frac{\partial^2}{\partial\theta_m\partial\theta_n}\ell_{em}] = 0 & m\neq n \\
-E[\frac{\partial^2}{\partial T_{i,T}^2}\ell_{em}] & 0<m=n\le M,\ \text{where } i=n \\
-E[\frac{\partial^2}{\partial T_{i,F}^2}\ell_{em}] & M<m=n\le 2M,\ \text{where } i=n-M \\
-E[\frac{\partial^2}{\partial F_{i,T}^2}\ell_{em}] & 2M<m=n\le 3M,\ \text{where } i=n-2M \\
-E[\frac{\partial^2}{\partial F_{i,F}^2}\ell_{em}] & 3M<m=n\le 4M,\ \text{where } i=n-3M
\end{cases} \\
&= \begin{cases}
0 & m\neq n \\
\frac{d^T\cdot N\cdot H(1-F_{i,T})}{T_{i,T}(1-T_{i,T}-F_{i,T})} & 0<m=n\le M,\ \text{where } i=n \\
\frac{d^F\cdot N\cdot H(1-F_{i,F})}{T_{i,F}(1-T_{i,F}-F_{i,F})} & M<m=n\le 2M,\ \text{where } i=n-M \\
\frac{d^T\cdot N\cdot H(1-T_{i,T})}{F_{i,T}(1-T_{i,T}-F_{i,T})} & 2M<m=n\le 3M,\ \text{where } i=n-2M \\
\frac{d^F\cdot N\cdot H(1-T_{i,F})}{F_{i,F}(1-T_{i,F}-F_{i,F})} & 3M<m=n\le 4M,\ \text{where } i=n-3M
\end{cases}
\end{aligned}
\tag{13}
$$

The Fisher information matrix $J(\theta)$ is actually diagonal. Therefore, the CRLB defined by $J^{-1}(\theta)$ can be computed efficiently by inversing each non-zero element in $J(\theta)$.

Note that the CRLB should be computed by the actual ground truth value of the parameters. However, in real-world applications, due to lack of the ground truth of those parameters, we feed the maximum likelihood estimations of the parameters to approximate the CRLB, i.e. CRLB is $J^{-1}(\hat{\theta}_{MLE})$.