Real-time Mobile Sensor Management Framework for city-scale environmental monitoring

Kun Qian[a], Christian Claudel[a]

Texas, United States

[a]*The University of Texas at Austin, Austin, TX 78712, USA.*

Abstract

Environmental disasters such as flash floods are becoming more and more prevalent and carry an increasing burden to human civilization. They are usually unpredictable, fast in development and extend across large geographical areas. The consequences of such disasters can be reduced through better monitoring, for example using mobile sensing platforms that can give timely and accurate information to first responders and the public. Given the extended scale of the areas to monitor, and the time-varying nature of the phenomenon, we need fast algorithms to quickly determine the best sequence of locations to be monitored. This problem is very challenging: the present informative mobile sensor routing algorithms are either short-sighted or computationally demanding when applied to large scale systems. In this paper, a real-time sensor task scheduling algorithm that suits the features and needs of city-scale environmental monitoring tasks is proposed. The algorithm is run in forward search and makes use of the predictions of an associated distributed parameter system, modeling flash flood propagation. It partly inherits the causal relation expressed by a search tree, which describes all possible sequential decisions. The computationally heavy data assimilation steps in the forward search tree are replaced by functions dependent on the covariance matrix between observation sets. Taking flood tracking in an urban area as a concrete example, numerical experiments in this paper indicate that this scheduling algorithm can achieve better results than myopic planning algorithms and other heuristics based sensor placement algorithms. Furthermore, this paper relies on a deep learning-based data-driven model to track the system states, and experiments suggest that popular estimation techniques have very good performance when applied to precise data-driven models. The data and code can be freely downloaded from [*].

Keywords: Environmental Monitoring, Mobile Sensing, Data Assimilation, Data-driven Model

1. Introduction

Floods are one of the most damaging natural disasters, accounting for 31% of economic losses resulting from all the natural disasters. The ten costliest floods between 1989 and 2014 caused an estimated US $187 billion losses and 13,597 casualties overall [1]. Several flood mitigation strategies exist, including flood channels, early warning systems [2] and temporary flood barriers [3]. Among

[*]Corresponding author. Tel. : (737)484-2035

Email addresses: `kunqian@utexas.edu` (Kun Qian), `christian.claudel@utexas.edu` (Christian Claudel)

[*]https://drive.google.com/drive/folders/1gRz4T2KGFXtlnSugarfUL8r355cXb7Ko?usp=sharing

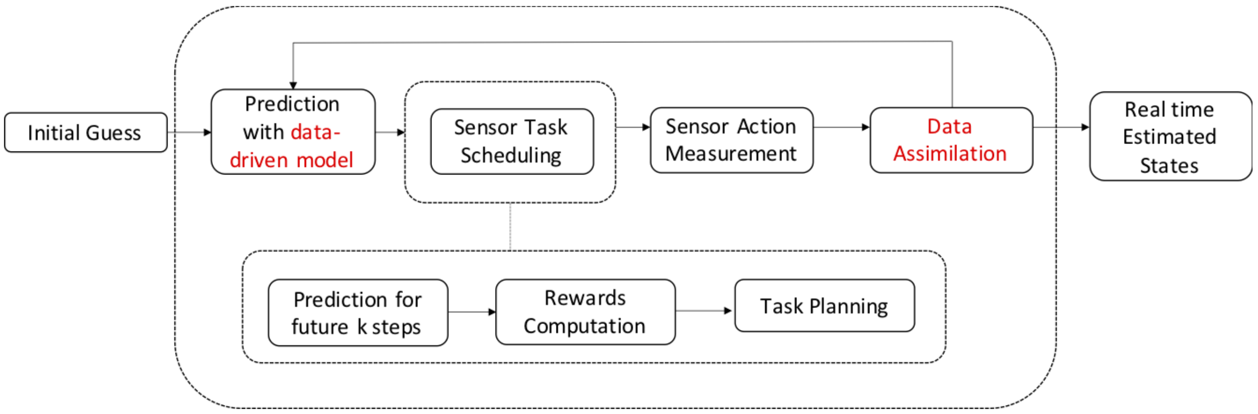

Initial Guess
Prediction with data-driven model
Sensor Task Scheduling
Sensor Action Measurement
Data Assimilation
Real time Estimated States
Prediction for future k steps
Rewards Computation
Task Planning

Figure 1: Overview of sensor placement problem investigated in this article. Starting from an initial guess of the environmental states, the deep learning based data-driven model [6] is applied to predict future states. Using these results, the sensor task scheduling module plans the sensing actions (trajectories and sensing schedule) for mobile sensors. Sensors then follow the planned tasks to take measurements at desired locations. An Ensemble Kalman filter is employed to assimilate measurements. The focus of this article is to introduce an informative and nonmyopic sensor task scheduling algorithm which can be implemented in real-time for such large scale systems.

all these strategies, flood monitoring and prediction is one of the most effective choices in terms of cost to benefit ratio. Such an early warning and emergency management system relies on a model that can provide a real-time prediction of flooding, and a sensor network to provide measurements and correct the model results. This article proposes a real-time algorithm to control robotic flood sensors over large areas. This algorithm is not restricted to flood monitoring but can be generalized to different natural hazards monitoring tasks such as wildfire, oil spills and other problems involving large scale distributed parameter systems. We also perform numerical experiments that demonstrate the effectiveness of the proposed method, which achieves near-optimal performance in minimizing estimation errors and gathering system information.

The workflow of the computational framework developed in this article is presented in Figure 1. Several difficulties exist when attempting to implement this framework in real-time. The first difficulty involves the Partial Differential Equation (PDE) based prediction model, which is computationally challenging. Furthermore, the sensor task scheduling does not scale well to large problems. The sensor task scheduling is a sequential decision making process in which control problems are coupled with estimation. The most popular methods currently available, such as pruning forward searching tree [4, 5], cannot solve the scheduling problem in real-time in large scale problems over some reasonable time horizon.

To enable real-time flood prediction, [6] introduced a physics informed deep learning flood prediction model that can generate city-scale high resolution flood predictions in real-time. The present work builds on this prediction model [6].

Considerable attention has been given to the problem of designing fixed sensor networks for environmental monitoring [7, 8]. Based on the submodular property of the mutual information function, [9] derived a greedy algorithm to select sensor locations for fixed sensor networks. [10] transformed the sensor location selection problem into a convex optimization problem.

In contrast to fixed sensor networks, mobile sensors have many advantages [11, 12, 13]. Since natural hazards usually happen over large spatial scales and tend to be unpredictable, it would be expensive and inefficient to maintain a large number of fixed sensors. However, one of the difficulties associated

with mobile sensors is the problem of sensor placement and scheduling. Many heuristic-based task scheduling algorithms [14, 15] have been developed in the past, such as following Voronoi graphs to cover an area [16] or reformatting the problem into traveling salesman-like problems [12, 17]. Beyond these heuristics, [18] proposes a myopic informative path planning algorithm for the robotics to explore a spatial distribution. [19] extends the work [18] to a non-myopic informative path planning algorithm. However, a limitation of the aforementioned works is that they assume that the events follow a Gaussian Distribution in space. In contrast with state-space models, Gaussian Processes are not accurate enough to model complex, highly nonlinear systems such as flash floods, though the computational requirements are much higher when dealing with state-space models. It has been shown in [5] that for linear state-space models, one can prune branches for the search tree and significantly reduce computational time. [20, 4, 21, 22] followed the idea in [5] to develop practical algorithms for environmental monitoring and searching applications. However, these algorithms are implemented in real-time at the cost of falling back to myopic scheduling strategies, and myopic strategy is not desired in city-scale environmental monitoring which would be shown in Section 4.
Another related field to this article is sensor management for object tracking. In sensor management problems, the objective is usually to maximize the information gain [23, 24] with finite sensing resources. [25, 26, 27] focuses on target monitoring with approximate mutual information. [28] formulates the active sensing problem into a Partially Observed Markov Decision Process(POMDP). [29] introduces a novel data-driven approach via imitation learning. The methods used in object tracking do not apply well to distributed parameters systems, however. Often, the dimensionality of the system to be modeled is usually much larger in distributed parameters systems.
This article proposes an informative and non-myopic mobile sensor task scheduling algorithm that is computationally efficient and can achieve near-optimal performance over simulated experiments. For this, we use predictions based on a deep learning model derived from conventional PDE flood propagation models. The task scheduling algorithm is based on a simplified forward search tree. It reorganizes the previously exponential growing trees and skips the data assimilation steps in the tree. An empirical function depending on the correlation of different observation sets is added to compensate for the skipped data assimilation steps in the planning process. Although the number of action sequences to be checked would still grow exponentially with the prediction horizon, this algorithm saves most of the data assimilation steps which are slowest to compute. Numerical experiments are conducted for the entire state estimation framework with a special focus on the task scheduling algorithm. It is shown that this sensor task scheduling algorithm achieves near-optimal performance.
Section 2 presents in detail the problem that the article intends to solve. In Section 3, each part in the mobile sensor framework is explained with a focus on the real-time mobile sensor task scheduling algorithm. Numerical experiment results are shown in Section 4. Section 5 presents additional insights and perspectives for future work.

2. Problem Description

We follow and extend the problem formulation of active information acquisition problem introduced in [4]. The formulation is adapted to better suit nonlinear high dimensional distributed parameters systems.
Consider mobile sensors whose dynamics are governed by

$$\omega(t+1) \in \mathcal{A}(\omega(t)) \tag{2.1}$$

in which $\omega(t)$ is the states of mobile sensors at time t and $\mathcal{A}(\omega(t))$ denotes the reachable set from $\omega(t)$.
The goal for mobile sensors is to monitor the states of a distributed parameters system. Suppose that the estimated states of the system is denoted as $b_S(t|Z_{1:t-1})$ in which $Z_{1:t-1}$ means the observations from time 1 to $t-1$, the dynamics of the distributed parameters system is written as

$$b_S(t|Z_{1:t_0}, \hat{Z}_{\omega(t_0)}, \dots, \hat{Z}_{\omega(t-1)}) = f(b_S(t-1|Z_{1:t_0}, \hat{Z}_{\omega(t_0+1)}, \dots, \hat{Z}_{\omega(t-1)}), u(t)) \tag{2.2}$$

in which $u(t)$ is an uncontrollable input term to the distributed parameters system and $\hat{Z}$ is used to denote the expected observation value instead of a real observation value.
We suppose that the observation model is linear, such that it can be described as

$$\hat{Z}_{\omega(t)} = H_{\omega(t)}(b_S(t|Z_{1:t_0-1}, \hat{Z}_{\omega(t_0)}, \dots, \hat{Z}_{\omega(t-1)})) \tag{2.3}$$

in which $\hat{Z}$ denotes the observation value extracted from belief states, and $H_{\omega(t)}$ denotes the observation model under the sensor state $\omega(t)$.
The major difference with previous formulation [4] is that previous formulation assumes that the system to be tracked is linear, however, we consider a nonlinear system. The differences arises as linear systems only need to consider the evolve of covariance matrix of estimated system while nonlinear systems need to consider both the mean value and covariance matrix because the dynamics changes with different mean values. Thus, we use a generalized expression for the measurement update.

$$b_S(t|Z_{1:t_0}, \hat{Z}_{\omega(t_0)}, \dots, \hat{Z}_{\omega(t-1)}, \hat{Z}_{\omega(t)}) = M(b_S(t|Z_{1:t_0}, \hat{Z}_{\omega(t_0)}, \dots, \hat{Z}_{\omega(t-1)}), \hat{Z}_{\omega(t)}) \tag{2.4}$$

in which M denotes the function of data assimilation. In this paper, we assume that the distribution of states of the distributed parameters system follows Gaussian distribution and Ensemble Kalman Filter will be applied for the data assimilation step. However, the function M can be any data assimilation methods.
Given all the constraints, we give the objective function.

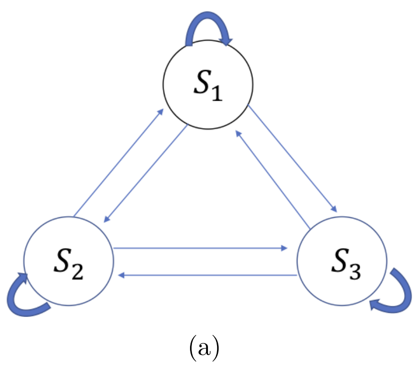
$$\min_{\omega(t_0+1),\dots,\omega(t_0+T)} \frac{1}{T} \sum_{t=t_0}^{T+t_0} \gamma^{t-t_0} \mathbf{R}(\omega(t), b_S(t|Z_{1:t_0-1}, \hat{Z}_{\omega(t_0)}, \dots, \hat{Z}_{\omega(t)})) \tag{2.5}$$

in which γ is a discounting coefficient and $\mathbf{R}$ denotes any reward function which can measure the uncertainty of the system. Note that $\mathbf{R}$ can also be the information gain of each sensing action. However, the min should be changed to max if information gain is used as the metric. The expression of objective function is also more generic than [4]. The detailed information of function $\mathbf{R}$ is discussed in Section 3.

3. Methodology

3.1. Real-time mobile sensor task scheduling algorithm for large scale systems

Without loss of generality, let us consider a simple example to illustrate how the algorithm simplifies a full forward search tree. Assuming that one mobile agent carrying sensors can move among



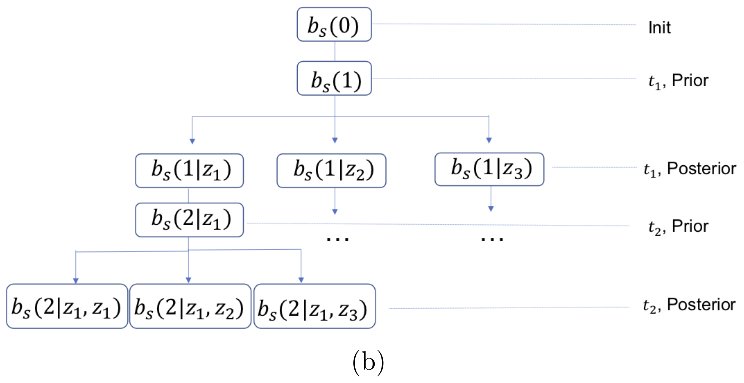


Figure 2: Illustration of a small-scale monitoring setup. (a) shows a simple system with three states $S_{1,2,3}$. It also shows the paths from which mobile sensors can travel between states to observe at each step. (b) constructs the full search tree which describes all possible action sequences of a single mobile sensor monitoring this system.

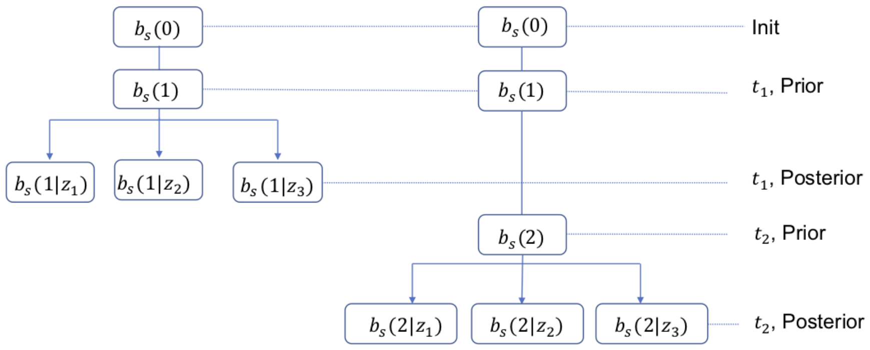


Figure 3: Approximated Search Tree. This approximation omits the data assimilation steps shown in Figure 2(b). The data assimilation steps are only conducted in the last step of each separate tree to compute the rewards value. The function Γ connects rewards obtained at different time steps together to form a new search tree with less computational complexity.

locations and obtain a measurement at each step in subplot (a) of Figure 2, a forward search tree is constructed in the (b) subplot of Figure 2 based on the possible sequences of sensing locations. Two factors contribute to a large number of computations. First, the number of branches in the tree to be searched grows exponentially. To be concrete, to predict T steps ahead, the number of branches to be searched is of the order $\mathcal{O}(D_{obs}^T)$ in which D_{obs} is the dimensionality of the candidate set for observations. Second, each data assimilation step is time-consuming. For instance, the approximated computational complexity of the Ensemble Kalman filter (EnKF) is $\mathcal{O}(ND^2)$ where D is the dimension of the system and N is the number of ensembles. Combining the above two factors, the overall computational complexity of the forward search tree is $\mathcal{O}(D_{obs}^{T+1}(TND^2 + \mathcal{O}_p))$ in which $\mathcal{O}_p$ is the computational complexity of model prediction. The computation cannot be implemented in real-time because D would be a large number for large scale environmental systems. Branch pruning methods [4] can not reduce the computational amount significantly while preserving the non-myopic planning feature when both D and D_{obs} are large.
To significantly reduce the computational complexity, the original search tree shown in Figure 2

Table 1: Computational time comparison between the simplified method and the full search tree

Prediction Horizon	Dimension of D	Dimension of D_{obs}	Full Search Tree Time	Simplified Time
$T = 2$	100	10	0.06s	0.1s
	1000	50	51s	7s
	10000	100	$> 1h$	420s
$T = 3$	100	10	0.7s	1.5s
	1000	50	2600s	28s
	10000	100	> 1 day	2600s
Computational Complexity			$\mathcal{O}(D_{obs}^{T+1}(TND^2))$	$\mathcal{O}(D_{obs}TND^2 + D_{obs}^{T+1}\mathcal{O}_\Gamma)$

is decomposed into T independent prediction trees. The approximation shown in Figure 3 omits the data-assimilation steps except at the last step of each tree. The model prediction power is fully preserved and the data assimilation step is kept in the last step of each decomposed tree for the purpose of calculating rewards. However, this simplification loses the causal relations among a sequence of observation actions. To address this, an empirical function is applied. The correlation between observation sets ω_i and ω_j is denoted as $\Sigma(\omega_i, \omega_j)$. We multiply a coefficient $\Gamma(\omega_t)$ computed from Equation 3.6 with the reward gained from observing ω_t at time step t. This coefficient serves to model the causal relationship between the previous observations and new observation set ω_t. The computational complexity of the simplified algorithm is $\mathcal{O}(D_{obs}TND^2 + NT\mathcal{O}_p + D_{obs}^{T+1}\mathcal{O}_\Gamma)$ in which $\mathcal{O}_\Gamma$ is the computational complexity of computing the function $\Gamma(\omega)$ for a single time. The term $D_{obs}^{T+1}\mathcal{O}_\Gamma$ may dominate under the large T situation. However, in this paper, because $\mathcal{O}_\Gamma$ is much smaller than ND^2 and we will not set T to be too huge, $D_{obs}^{T+1}\mathcal{O}_\Gamma$ is not a computational heavy burden. Also, the computational complexity of model prediction $\mathcal{O}_p$ will not be an obstacle to real-time implementation because of the deep learning prediction model. The comparison of the computational complexity between the full branch search tree and the simplified one is shown in Table 1. We fix some specific values for forward prediction horizon T, system dimension D and observation locations D_{obs} to compare the computational time for full search tree and simplified search method. It can be observed that when the dimension of D is 1000 and D_{obs} is 50, the full search tree based method cannot handle the computational amount in real time any more.
The expression of $\Gamma(\omega)$ is shown in Equation 3.6. We need $f_e(x)$ to be high when x is small. Indeed, the more we gather information on specific states, the less uncertain we are about these states and other states that are strongly correlated. The correlation between states is based on empirical correlation drawn from a large amount of simulations. Detailed discussion about $f_e(x)$ is in Section 4.

$$\Gamma(\omega_{t+1}) = f_e((\Pi_{i=1}^{t}\Sigma(\omega_i, \omega_{t+1}) \exp{(i - t - 1)}) + 0.01 \tag{3.6}$$

The approximation is summarized in Equations 3.7 3.8 3.9 3.10 3.11, and the corresponding algorithm is presented in algorithm 1.

$$\min_{\omega(t_0),\ldots,\omega(t_0+T)} \frac{1}{T}\sum_{t=t_0}^{T+t_0} \gamma^{t-t_0}\Gamma(\omega(t))\mathbf{R}(\omega(t), b_S(t|Z_{1:t_0-1}, \hat{Z}_{\omega(t)})) \tag{3.7}$$

Subject to

$$b_S(t|Z_{1:t_0}) = f(b_S(t-1|Z_{1:t_0}), u(t)) \tag{3.8}$$

$$\hat{Z}_{\omega(t)} = H_{\omega(t)}(b_S(t|Z_{1:t_0-1})) \tag{3.9}$$

$$b_S(t|Z_{1:t_0}, \hat{Z}_{\omega(t)}) = M(b_S(t|Z_{1:t_0}), \hat{Z}_{\omega(t)}) \tag{3.10}$$

$$\text{s.t. } \omega(t+1) \in \mathcal{A}(\omega(t)) \tag{3.11}$$

Algorithm 1 Fast Sensor task Scheduling Algorithm for single mobile sensor

```
 1: Input: $\Omega(t_0), \ldots, \Omega(t_0+k)$; $b_S(t|Z_{1:t_0})$, $k$, $u(t_0+1), \ldots, u(t_0+k)$, $T$;    ▷ $\Omega(t)$ is the set of
    already fixed observation locations at time step $t$.
 2: Init: $t = 1$;
 3: while $t \leq T$ do                                          ▷ Prediction $T$ steps
 4:     $b_S(t+1|Z_{1:t_0}) = f(b_S(t+1|Z_{1:t_0}), u(t+1))$;
 5:     t += 1
 6: set $t = 1$
 7: while $t \leq T$ do                                ▷ Compute Rewards for each future step
 8:     for $\omega_i(t) \in \mathcal{A}(t)$ do      ▷ $\mathcal{A}(t)$ is the set of all possible observation locations at time $t$
 9:         Compute: $\mathbf{R}(\omega_i(t), b_S(t|Z_{1:t_0}, \hat{Z}_{\omega(t)}))$
10:     Normalize: $\forall \omega_i(t) \in \mathcal{A}(t), \mathbf{R}(\omega_i(t), b_S(t|Z_{1:t_0}, \hat{Z}_{\omega(t)})) = \frac{\mathbf{R}(\omega_i(t), b_S(t|Z_{1:t_0}, \hat{Z}_{\omega(t)}))}{\max_{\omega_j(t)\in\mathcal{A}(t)}(\mathbf{R}(\omega_i(t), b_S(t|Z_{1:t_0}, \hat{Z}_{\omega(t)})))}$
11:     $t+ = 1$
12: Organize all the possible sequences of observations $Seq_i$ into set $Seq$
13: Compute the value $\sum_{t=t_0}^{T} \gamma^{t-t_0}\Gamma(\omega(t))\mathbf{R}(\omega(t), b_S(t|Z_{1:t_0-1}, \hat{Z}_{\omega(t)}))$ for each $seq_i$
14: return: $\max Seq_i$
```

3.2. Ensemble Kalman filter(EnKF)

Data assimilation is one of the main components in the frame shown in Figure 1. As mentioned in Section 2, the proposed framework can be deployed along with general data assimilation methods. Meanwhile, since this framework targets at city-scale distributed parameters systems, we apply the Ensemble Kalman Filter (EnKF) for assimilation. EnKF has been shown to be a computationally

Methods	Computation Speed
PDE Solver	1×
Model 1	50000×

Table 2: Computational time improvements. All computations are performed on a Intel i7-7700 CPU @3.60GHz. Since the computational time of the PDE solver depends on the initial conditions and the inputs of the problem, we average the PDE solver performance over a large number of different cases. [6]

efficient and robust data assimilation method [30, 31, 32] for large scale geographical and environmental applications. With the assumption of Gaussian Distribution, EnKF follows the same format as the classical Kalman filter to conduct data assimilation and employs Monte Carlo methods to capture nonlinearities. The ensemble of estimated state b_S is denoted as B_S. The measurement update of EnKF is introduced in Equation 3.12. $\Sigma(B_S(t|Z_{1:t_0}))$ is the covariance matrix of $B_S(t|Z_{1:t_0})$ and R is covariance matrix of measurement noise induced by $\omega(t)$.

$$B_S(t|Z_{1:t_0}, Z_t) = B_S(t|Z_{1:t_0}) + \Sigma(B_S(t|Z_{1:t_0}))H^T_{\omega(t)}K_{\omega(t)}(Z_t - H_{\omega(t)}B_S(t|Z_{1:t_0})) \tag{3.12}$$

in which

$$K_{\omega(t)} = (H_{\omega(t)}\Sigma(B_S(t|Z_{1:t_0}))H^T_{\omega(t)} + R)^{-1}$$

3.3. Data-Driven Dynamics Model

This work depends on the deep learning based data-driven model invented in [6] to predict flood dynamics in real-time. The data-driven model is derived from training a Convolutional Neural Network(CNN) fed with data simulated by a Shallow Water Equation(SWE) solver. Table 2 compares the time taken to compute dynamics by data-driven model and PDE solver. The data-driven model not only boosts the prediction but also keeps the precision. It was validated that the prediction by the data-driven model will not diverge from predictions by the SWE solver when evolving along with time. In Section 4 of this paper, it is validated with experiments that Ensemble Kalman Filter could have good performance in assimilating measurement data with the data-driven prediction model.

3.4. Reward Function

Information theory [33] provides ways to measure the uncertainty of a system. Entropy describes the uncertainty of the belief of a system. The reduction of entropy through measurement can be used as a way to measure information gain [24]. The trace or log of determinant of the covariance matrix is also employed by many works [5, 21, 20, 4] to measure the uncertainty of systems. From the estimation point of view, Cramer Rao bound [34] is the lower bound of error for nonlinear estimators. It is also reasonable to employ it as the rewards function for sensor actions [35, 36]. In many other works about sensor management [23, 37, 38], Kullback Divergence(KL Divergence) [39] or Renyi Divergence are popular choices.

3.5. Real-time solution to multiple sensors case

The dimensionality of observation space is exponential in the number of sensors n if we follow algorithm 1. Real-time implementations are only possible when n is small. [9] introduced a greedy algorithm to select locations for multiple fixed sensors. This greedy algorithm is built upon the submodular property of the mutual information reward function. [20] relies on coordinate descent which is a similar greedy algorithm with [9] to schedule tasks for a number of mobile sensors. To apply the coordinate descent algorithm to multiple mobile sensors, we need to fix an order of sensors to be computed in advance. Algorithm 2 extends algorithm 1 and follows a similar coordinate descent style as introduced in [20].

Algorithm 2 Multiple Mobile Sensor task Scheduling

```
   Input: $b_S(t|Z_{1:t_0})$, $k$, $u(t_0+1)$, …, $u(t_0+k)$, $n_S$;  ▷ $n_S$ is the total number of mobile sensors
   Init: $\Omega(t_0)=\{\}$, …, $\Omega(t_0+k)=\{\}$   ▷ $\Omega(i)$ is the set to save all the locations decided to be visited at time step $i$
3: for $i=1:n_S$ : do
       $(\omega(t_0),\dots,\omega(t_0+k))$ = Fast Sensor task Scheduling Algorithm for single sensor($\Omega(t_0)$, …, $\Omega(t_0+k)$, $b_S(t|Z_{1:t_0})$, $k$, $u(t_0+1)$, …, $u(t_0+k)$)
       $\Omega(t_0)=\{\Omega(t_0),\omega(t_0)\}$, …, $\Omega(t_0+k)=\{\Omega(t_0+k),\omega(t_0+k)\}$
6: Return: $\Omega(t_0)$   ▷ $\Omega(t_0)$ is the set of locations to be observed at next step
```

4. Numerical Experiments

Section 4 analyzes the performance of the real-time mobile sensors task scheduling algorithm with experiments. Since floods are events that occur over large (city-scale) spatial domains, a very large number of sensors operating for a few hours is required to accurately monitor these events, which have not been deployed in a given location to date. There is thus a lack of accurate, a city-scale flood propagation in earlier literature. To adddress this, we use simulated flood data based on [6] to conduct numerical experiments. Austin, Texas is chosen as the studied area shown in Figure 4. Using Austin as an example does not restrict the application domain since the method can be applied to any other city where the flood would follow the Shallow Water Equation [6], with different model parameters. As a city with numerous unmonitored creeks, Austin would benefit from such a mobile flood tracking system. Furthermore, [6] provides a data-driven model to boost the computational speed of the forecast step, which is key to real-time implementation of the present path-planning framework.
Floods caused by heavy rains are simulated for mobile sensors to track. It is assumed that the Partial Differential Equation solver simulates trues states of the environment. The planning frame relies on the deep learning model [6] to predict the development of floods and has no access to any of the true rain inputs or flood states but only noisy values corrupted by Gaussian noise from predictions and sensor measurements. In this work, the rewards are defined using the log-determinant of the covariance matrix. The reward at each step is defined by Equation 4.13.

$$\mathbf{R}(\omega(t), b_S) = \log\det(\Sigma(b_S(t)|\hat{Z}_{\omega(t)})) \tag{4.13}$$

To evaluate the performance of $\Gamma(\omega_t)$ and search for the best hyperparameters, simple experiments are conducted to compare the result by full tree search and results by approximated search tree with different heuristic functions. In order to make the computation tractable for the full search case, each experiment selects only 4 different locations as candidate measurement locations for one mobile sensor. The planning horizon is set as 4. In Figure 5, the ablation study is conducted with 30 different observation candidates sets. The ablation study indicates that with an appropriate

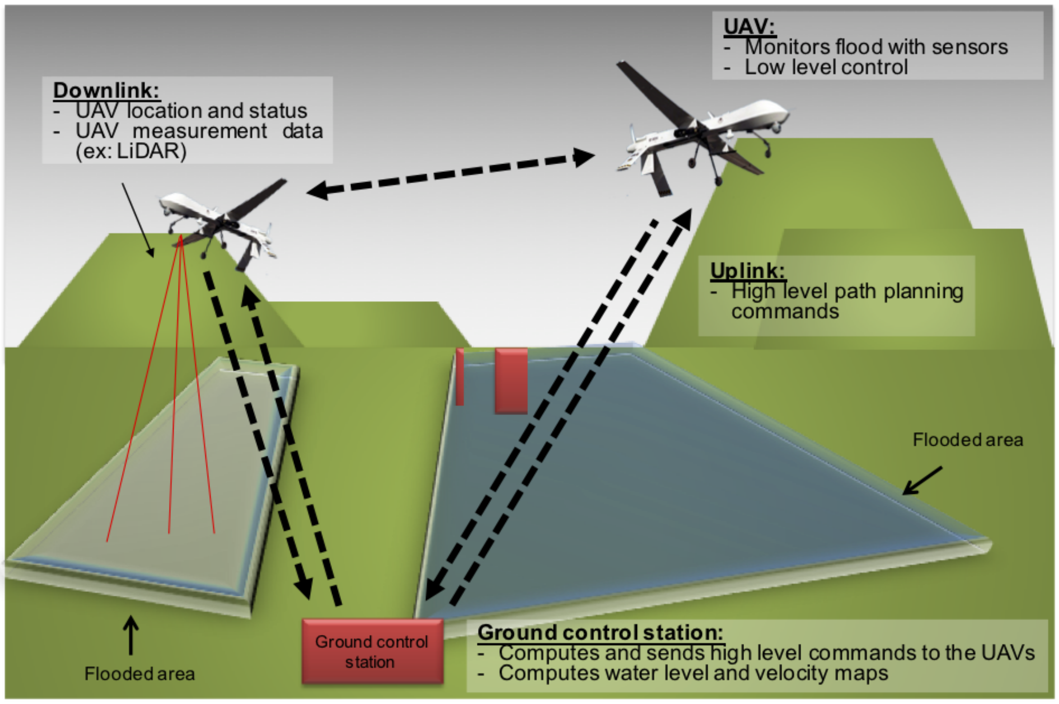

UAV:
- Monitors flood with sensors
- Low level control
Downlink:
- UAV location and status
- UAV measurement data (ex: LiDAR)
Uplink:
- High level path planning commands
Flooded area
Flooded area
Ground control station
Ground control station:
- Computes and sends high level commands to the UAVs
- Computes water level and velocity maps

Figure 4: Illustration of the UAV flood monitoring procedure. UAVs carrying sensors are distributed around the area to be monitored. Ground control stations send commands to UAVs with high level task commands (target locations). This paper focuses on the algorithm to generate high level tasks commands, while an earlier paper focused on low-level UAV control [40].

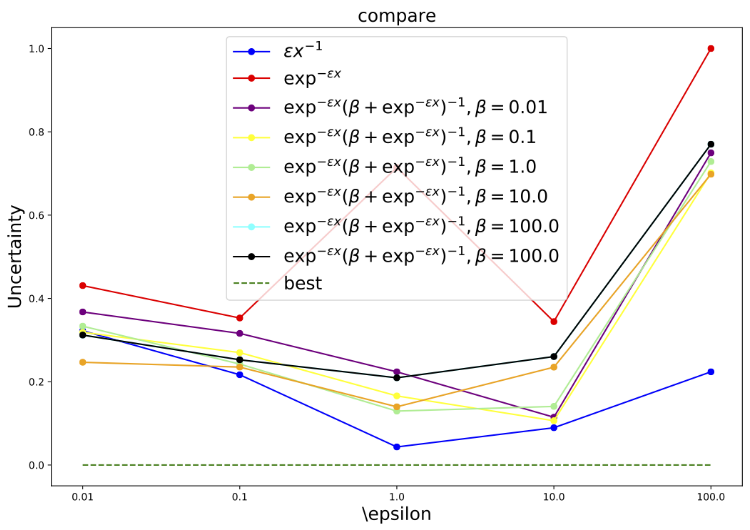


Figure 5: Comparison of different hyper-parameters and empirical function in $\Gamma(\omega)$. The experiments are conducted with 30 scenarios of each a different set of observation candidates is considered. In order to reduce the computation time, the chose a small set of observation candidates.

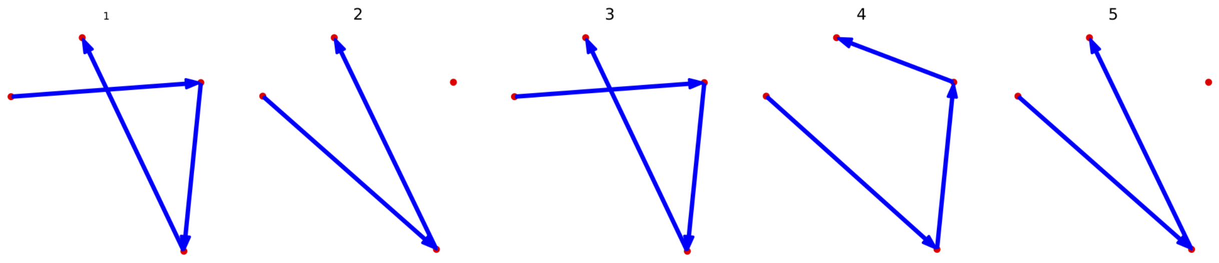


Figure 6: Comparison between result by the full forward search tree and results by approximated search tree. 1 shows the optimal sensor task sequence computed by full forward search tree. 2 shows the observation sequence searched by approximated search tree without adding function $\Gamma(\omega(t))$. 3 shows the result by setting $f_e(x) = 1.0/x$. 4 shows the result by setting $f_e(x) = \exp^{-10.0x}$. 5 shows the result by setting $f_e(x) = \frac{\exp^{-100.0x}}{1.0+\exp^{-100.0x}}$.

function $\Gamma(\omega(t))$, the algorithm can find observation sequences performing almost as well as sequences computed by full branch search tree while requiring considerably less computational time according to Table 1. To qualitatively understand the performance, according to the Figure 6, we can see that without $\Gamma(\omega(t))$, the sensor will be guided to visit the same spot again and again continuously. With an appropriate choice of $\Gamma(\omega(t))$, the mobile sensor can be guided to visit different spots at different steps.

The performance of algorithm 1 is studied with two experiments. The first experiment compares the performance of a single mobile sensor under myopic and non-myopic strategies. The second experiment assumes few mobile sensors working together with a number of fixed sensors installed along the river. We analyze quantitatively the performance of the algorithm 1 with different prediction horizon lengths and show qualitatively the estimation results. Experiments with a large number of mobile sensors are conducted to show that following algorithm 2 could achieve much better performances in managing a large number of mobile sensors than other strategies. To

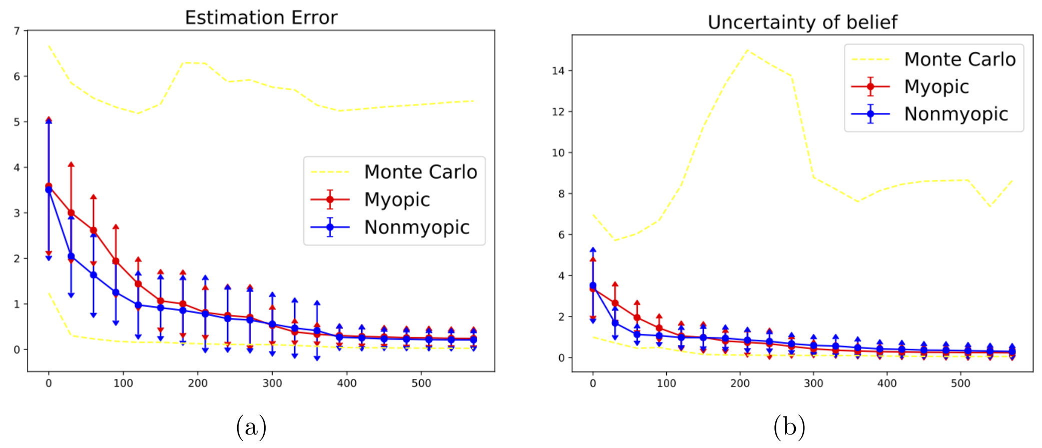


Figure 7: Performance comparison between Nonmyopic and Myopic strategies with single mobile sensor. The results are averaged over 300 cases that come from 30 different simulated time series data and 10 different initial guesses for each. Results are combined together from each experiment after normalization. The vertical lines in the plot indicate the variance of performance through different cases. Monte-Carlo Simulation produces the range of performance for all 300 cases. (a) shows the estimation error. (b) show the uncertainty of belief states.

evaluate the performance, we use 30 different pieces of time series data and generate 10 different initial guess of states for each time-series data. Notice that even with algorithm 1 and algorithm 2, the computation amount is still huge for large scale systems.

4.1. Single Mobile Sensor

In the first experiment, we follow the algorithm 1 to schedule the sensing task for a single mobile sensor. Although single mobile sensor is a particular case of the multi-agent problem, this experiment shows with the simple setting that the proposed algorithm can guide mobile sensors to choose more informative ways to explore and overcome the drawbacks of myopic strategies. We compare the performance of myopic and non-myopic planning strategies. With the non-myopic strategy, the prediction horizon is set to be 3 steps ahead. Monte Carlo simulations are used to show that the informative scheduling algorithm achieves near-optimal performance.

According to Figure 7, we can see that mobile sensors planned using algorithm 1 would reduce the estimation error and the uncertainty of the system in time. Besides, the non-myopic planning strategy helps improve the performance over myopic strategy. This is due to the fact that non-myopic strategies tend to bring the mobile sensor to explore new areas such that information from different areas can be collected. Comparing to the cases simulated by Monte-Carlo methods, both strategies' performance is close to optimal performance after a period.

In Figure 8, we qualitatively compare the estimation of floods states under different sensing strategies. The same initial guess, which is corrupted by noise from true states, is exposed to different task scheduling algorithms. We can see that the estimated flood states are closer to ground truth, in comparison to the prediction-only case. Furthermore, the informative strategy keeps a better track of the evolution of floods than choosing sensing locations randomly.

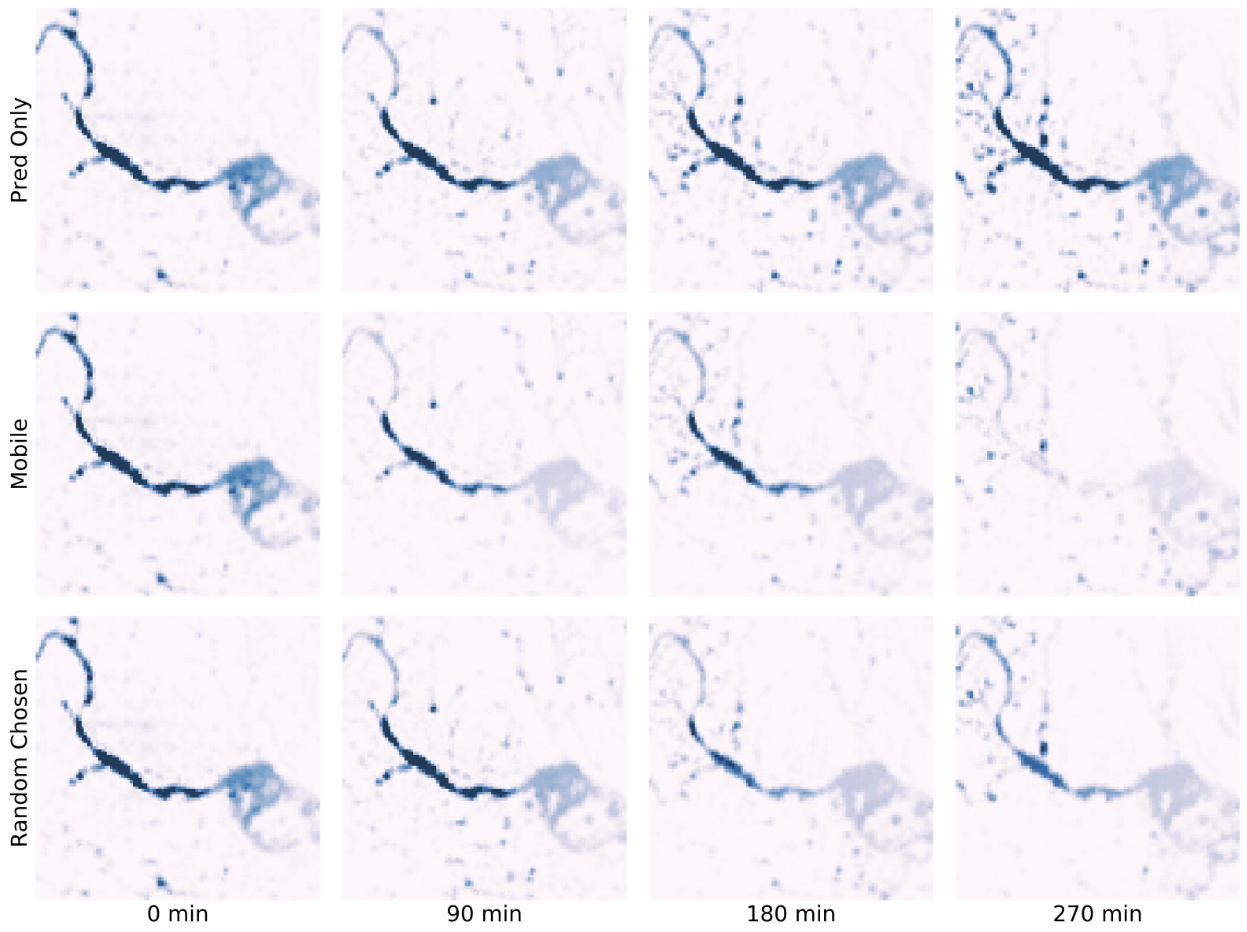


Figure 8: Estimation error by different sensor strategy. The difference in depth of water with real status is represented by a color shade. The first row is obtained by only predicting with the data-driven model. The second row shows results estimated with mobile sensor scheduled by algorithm 1. The third row shows the estimation results by randomly chosen sensors. We assume the same biased guess of initial status for each case.

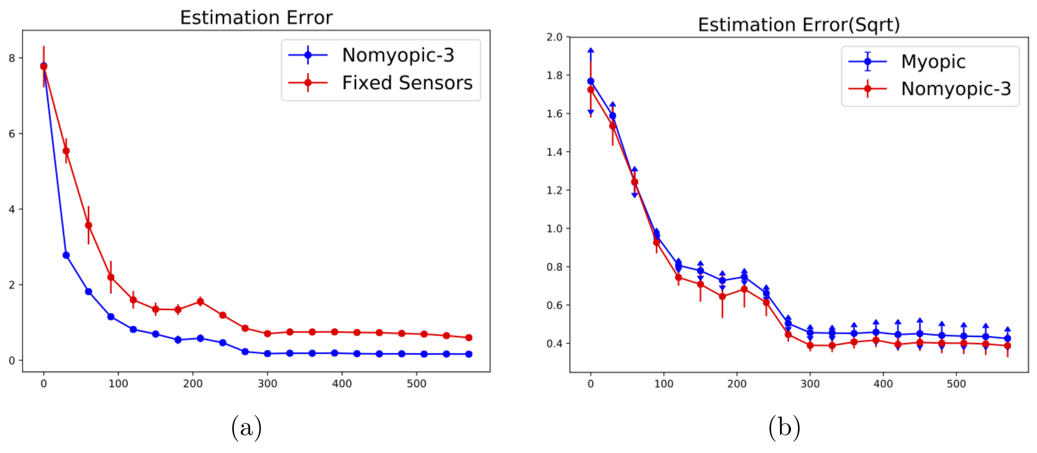


Figure 9: Performance comparison of mobile sensing and fixed sensing. In this figure, we use the same data as in Figure 7. (a) shows the estimation error with fixed sensors only and the result by mobile sensors working together with fixed sensors. (b) shows the comparison between setting different time horizons in algorithm 1 when scheduling mobile sensor tasks. 1 step(myopic), 3 steps(non-myopic) are compared.

4.2. Small number of mobile sensors together with fixed sensors

Rivers are among the most important areas to monitor during floods [41]. First, most precipitation would ultimately flow into the river and cause its level to increase significantly, bringing important information about the flooding situation. Second, permanent rivers typically have water levels and surface velocities that are considerably larger than levels and surface velocities associated with temporary urban water streams during floods, yielding a higher signal to noise ratio for the flood sensors. In the second experiment, we simulate the deployment of a number of fixed sensors along the river, together with several mobile sensors. This experiment shows that algorithm 1 makes advantage of mobile sensors to improve the estimation results beyond the fixed sensor only strategy. Meanwhile, both quantitative and qualitative analyses are given to illustrate how the prediction horizon in algorithm 1 affects the estimation results and planned trajectories.

In Figure 9(a), it can be seen that, qualitatively, adding mobile sensors to an existing fixed sensor network would significantly improve the estimation performance. In order to quantify the added value of mobile sensor data (in addition to existing fixed sensor data), we compare the performance improvement using different number of fixed sensors in Figure 10. We can see that while increasing the number of fixed sensors could improve the estimation performance when the number of sensors is less than 10, while the improvement is almost non-existent when the number is greater than 20. However, we can see from the difference between last two points in Figure 10 that adding one mobile sensors to 24 fixed sensors could continue to improve the estimation performance greatly.

Figure 9(b) compares the performance of algorithm 1 with different prediction horizons. Planning under the non-myopic strategy would reduce the estimation error comparing to planning under myopic strategy.

In order to understand qualitatively the results in Figure 9, we plot the scheduled trajectories of mobile sensors by the algorithm with different predicting horizons. According to Figure 11, we can see that the green track which is planned under the non-myopic strategy is guided around the whole interested domain. However, the blue track which is planned under the myopic strategy is

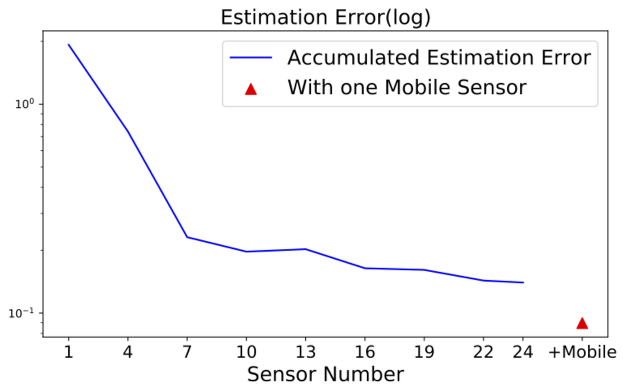


Figure 10: The comparison of estimation errors by different number of sensors. We gradually increase the number of fixed sensors and plot the estimation error changes (X-axis 1 - 24). When increasing fixed sensors cannot reduce the estimation error significantly, we add a mobile sensor to the group of fixed sensors (+Mobile).

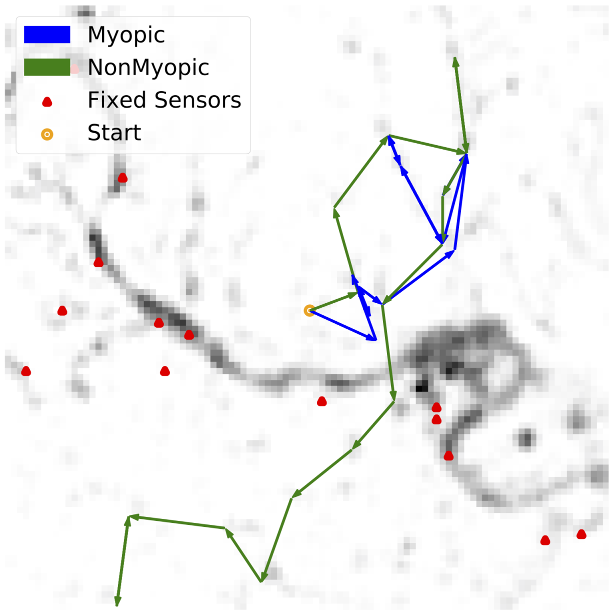


Figure 11: The comparison of mobile sensor tracks when planning under myopic and nonmyopic strategies. Red dots are the locations of fixed sensors, the blue track is induced by the myopic strategy, while the green track is induced by the nonmyopic strategy. In this Figure, all mobile sensors start from the same location, and the initial states and inputs are identical.

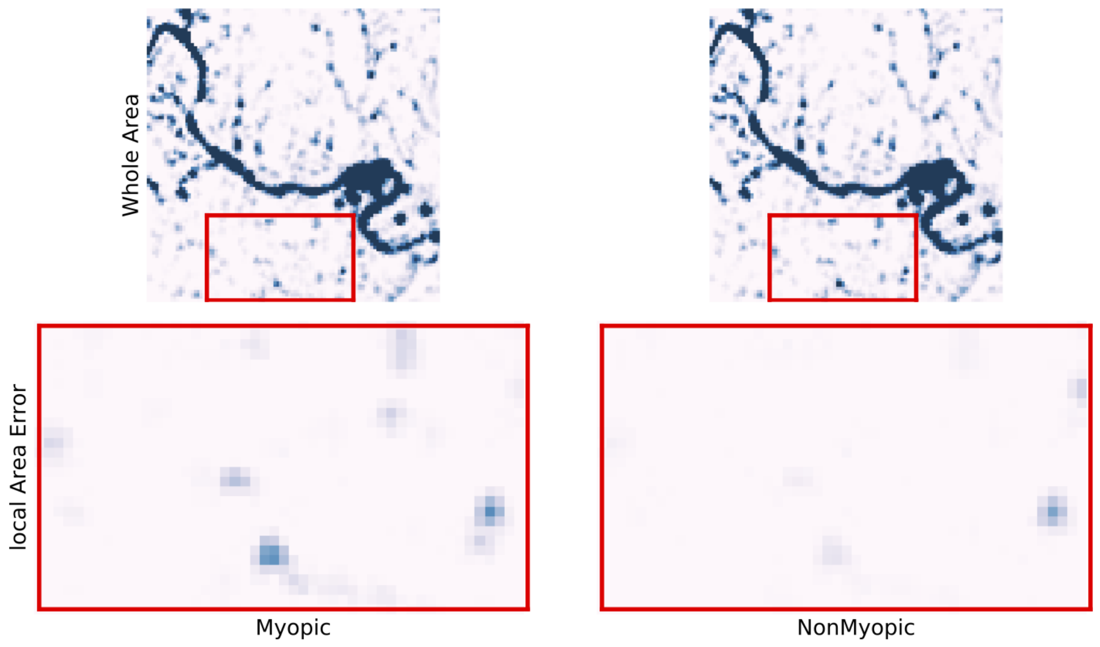
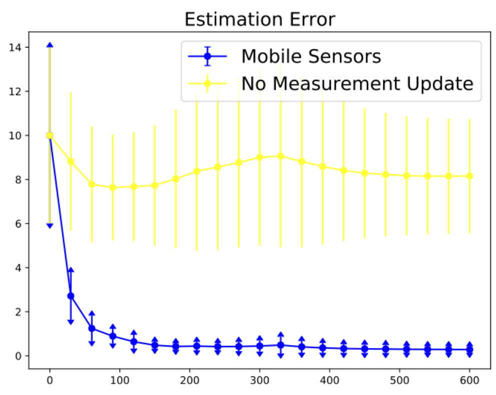


Figure 12: The demonstration of estimation results corresponding to the same experiment case in Figure 11. The first row shows the estimation result of whole domain. The second row shows the estimation error in a local area for each algorithm. The first column shows the estimation result following myopic strategy. The second column shows the estimation result following nonmyopic strategy.

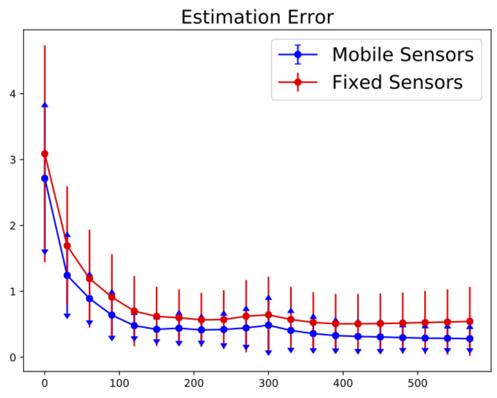
trapped in a local area. This is a reasonable result as myopic strategies can not see other areas taht possibly have higher rewards during the plan. It also indicates that the function $\Gamma(\omega)$ successfully models the causal relation between sequential observation actions and encourages sensors to explore uncorrelated states. The difference in estimation results are presented in Figure 12. The non-myopic strategy gives estimation results closer to ground truth in local areas where the myopic strategy has not brought the mobile sensors to.

4.3. Large number of mobile sensors

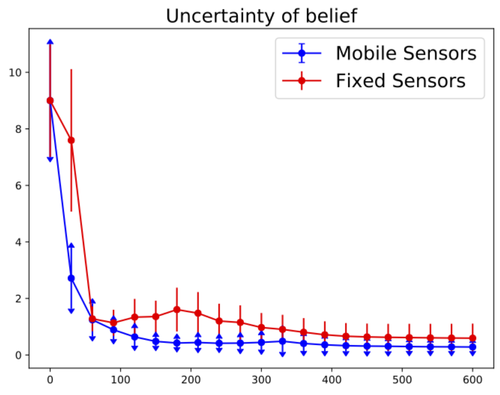
In this experiment, 24 mobile sensors are deployed. We follow the algorithm 2 to schedule tasks for mobile sensors. To compare with the performance of mobile sensors, we experiment with another 30 different sets of fixed sensors. Each set of fixed sensors is also made up of 24 sensors and they are chosen according to simulation data and metrics by [9, 10].
According to the results shown in Figure 13, we can see that in terms of estimation error, both the fixed sensors deployment and mobile sensors strategy could improve a lot over the prediction only case. Mobile sensors do better than fixed sensors in reducing the estimation error and uncertainty of the system. Meanwhile, it is clear that even after a long time, the estimation results by fixed sensors are still not as good as estimation results by mobile sensors. The reason is that mobile sensors are keeping exploring new areas and thus keep reducing the error. Although these sets of fixed sensors are picked to maximize specific metrics, they can not actively exploring the region thus would miss information in some local areas.



(a)



(b)



(c)

Figure 13: Quantitative Performance analysis of estimation results with large number of sensors. Experiments data used here is the same as the data used in Figure 7. (a) shows the comparison of estimation results between algorithm 2 and the results without any measurement updates. (b) shows the comparison of estimation error between fixed sensors and mobile sensor strategies. In order to show the comparison clearly, we plot from the second step. (c) shows the comparison in term of uncertainty of estimation.

5. Conclusion

In this paper, we propose a real-time mobile sensor task scheduling algorithm to arrange mobile sensors to monitor city-scale environmental hazards. The algorithm is based on an approximation of the forward search tree. The approximated search tree keeps the full prediction power of the dynamical model, which is data-driven, based on a Partial Differential Equation. Functions based on the correlation between observation actions are added to the approximated search tree to account for the missed causal relationship between observations due to omitting the data assimilation steps in the search tree. The approximated tree thus decouples the term D_{obs}^{T+1} which grows exponentially with the prediction horizon and the time-consuming data assimilation term. Experiments show that the algorithm is real-time on regular desktop computers for city-scale high dimensional systems, and achieves near-optimal performance in reducing estimation error and uncertainty. The scheduled paths for mobile sensors can explore actively around the entire spatial domain, which overcomes the drawbacks of myopic strategies. The experiments results also indicate that a precise data-driven model could track the development of covariance of the system well such that assimilation could improve the estimation precision significantly. In future work, the authors are interested in two possible research directions. First, the authors plan to explore sensor management strategies in which the environmental dynamics are modeled in a continuous manner, as a distributed parameters system. Modeling the dynamics of the environment continuously could benefit the problem by reducing the dimension of the state, through the choice of a sparse set of functions to represent it. Second, the authors would like to explore searching for optimal or suboptimal solutions of the same sensing problem, by modeling the problem as a Markov Decision Process (MDP). The authors hope that new progress in MDP solvers or reinforcement learning could help find near-optimal results to this large scale estimation/sensor placement problem.

Acknowledgement

The Authors would like to acknowledge NSF(National Science Foundation) CPS No.1739964 and NSF CIS No.1636154 for research Funding.

References

[1] M. Trigg, C. Birch, J. Neal, P. Bates, A. Smith, C. Sampson, D. Yamazaki, Y. Hirabayashi, F. Pappenberger, E. Dutra, et al., The credibility challenge for global fluvial flood risk analysis, Environmental Research Letters 11 (9) (2016) 094014.

[2] V. V. Krzhizhanovskaya, G. Shirshov, N. Melnikova, R. G. Belleman, F. Rusadi, B. Broekhuijsen, B. Gouldby, J. Lhomme, B. Balis, M. Bubak, et al., Flood early warning system: design, implementation and computational modules, Procedia Computer Science 4 (2011) 106–115.

[3] R. Few, Flooding, vulnerability and coping strategies: local responses to a global threat, Progress in Development Studies 3 (1) (2003) 43–58.

[4] N. Atanasov, J. Le Ny, K. Daniilidis, G. J. Pappas, Information acquisition with sensing robots: Algorithms and error bounds, in: 2014 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2014, pp. 6447–6454.

[5] M. P. Vitus, W. Zhang, A. Abate, J. Hu, C. J. Tomlin, On efficient sensor scheduling for linear dynamical systems, Automatica 48 (10) (2012) 2482–2493.

[6] K. Qian, A. Mohamed, C. Claudel, Physics informed data driven model for flood prediction: Application of deep learning in prediction of urban flood development, arXiv preprint arXiv:1908.10312.

[7] B. Kerkez, S. D. Glaser, R. C. Bales, M. W. Meadows, Design and performance of a wireless sensor network for catchment-scale snow and soil moisture measurements, Water Resources Research 48 (9).

[8] J. E. Weimer, B. Sinopoli, B. H. Krogh, A relaxation approach to dynamic sensor selection in large-scale wireless networks, in: 2008 The 28th International Conference on Distributed Computing Systems Workshops, IEEE, 2008, pp. 501–506.

[9] A. Krause, A. Singh, C. Guestrin, Near-optimal sensor placements in gaussian processes: Theory, efficient algorithms and empirical studies, Journal of Machine Learning Research 9 (Feb) (2008) 235–284.

[10] S. Joshi, S. Boyd, Sensor selection via convex optimization, IEEE Transactions on Signal Processing 57 (2) (2008) 451–462.

[11] P. Tokekar, E. Branson, J. Vander Hook, V. Isler, Tracking aquatic invaders: Autonomous robots for monitoring invasive fish, IEEE Robotics & Automation Magazine 20 (3) (2013) 33–41.

[12] P. Tokekar, J. Vander Hook, D. Mulla, V. Isler, Sensor planning for a symbiotic uav and ugv system for precision agriculture, IEEE Transactions on Robotics 32 (6) (2016) 1498–1511.

[13] H.-L. Choi, Adaptive sampling and forecasting with mobile sensor networks, Ph.D. thesis, Massachusetts Institute of Technology (2009).

[14] P. Neumann, S. Asadi, J. H. Schiller, A. J. Lilienthal, M. Bartholmai, An artificial potential field based sampling strategy for a gas-sensitive micro-drone, in: IROS Workshop on Robotics for Environmental Monitoring (WREM), San Francisco, USA, 2011, 2011, pp. 34–38.

[15] B. Zhang, G. S. Sukhatme, Adaptive sampling for estimating a scalar field using a robotic boat and a sensor network, in: Proceedings 2007 IEEE International Conference on Robotics and Automation, IEEE, 2007, pp. 3673–3680.

[16] M. Schwager, M. P. Vitus, S. Powers, D. Rus, C. J. Tomlin, Robust adaptive coverage control for robotic sensor networks, IEEE Transactions on Control of Network Systems 4 (3) (2015) 462–476.

[17] J. Yu, M. Schwager, D. Rus, Correlated orienteering problem and its application to informative path planning for persistent monitoring tasks, in: 2014 IEEE/RSJ International Conference on Intelligent Robots and Systems, IEEE, 2014, pp. 342–349.

[18] A. Singh, A. Krause, C. Guestrin, W. J. Kaiser, M. A. Batalin, Efficient planning of informative paths for multiple robots., in: IJCAI, Vol. 7, 2007, pp. 2204–2211.

[19] A. Singh, A. Krause, W. J. Kaiser, Nonmyopic adaptive informative path planning for multiple robots, in: Twenty-First International Joint Conference on Artificial Intelligence, 2009.

[20] B. Schlotfeldt, D. Thakur, N. Atanasov, V. Kumar, G. J. Pappas, Anytime planning for decentralized multirobot active information gathering, IEEE Robotics and Automation Letters 3 (2) (2018) 1025–1032.

[21] S. T. Jawaid, S. L. Smith, Submodularity and greedy algorithms in sensor scheduling for linear dynamical systems, Automatica 61 (2015) 282–288.

[22] A. B. Asghar, S. T. Jawaid, S. L. Smith, A complete greedy algorithm for infinite-horizon sensor scheduling, Automatica 81 (2017) 335–341.

[23] A. O. Hero, D. Castañón, D. Cochran, K. Kastella, Foundations and applications of sensor management, Springer Science & Business Media, 2007.

[24] A. O. Hero, D. Cochran, Sensor management: Past, present, and future, IEEE Sensors Journal 11 (12) (2011) 3064–3075.

[25] P. Dames, V. Kumar, Autonomous localization of an unknown number of targets without data association using teams of mobile sensors, IEEE Transactions on Automation Science and Engineering 12 (3) (2015) 850–864.

[26] B. Charrow, V. Kumar, N. Michael, Approximate representations for multi-robot control policies that maximize mutual information, Autonomous Robots 37 (4) (2014) 383–400.

[27] G. M. Hoffmann, C. J. Tomlin, Mobile sensor network control using mutual information methods and particle filters, IEEE Transactions on Automatic Control 55 (1) (2009) 32–47.

[28] M. Lauri, R. Ritala, Stochastic control for maximizing mutual information in active sensing, in: Proc. IEEE Int. Conf. Robot. Autom, 2014, pp. 1–6.

[29] S. Choudhury, A. Kapoor, G. Ranade, D. Dey, Learning to gather information via imitation, in: 2017 IEEE International Conference on Robotics and Automation (ICRA), IEEE, 2017, pp. 908–915.

[30] G. Evensen, Sequential data assimilation with a nonlinear quasi-geostrophic model using monte carlo methods to forecast error statistics, Journal of Geophysical Research: Oceans 99 (C5) (1994) 10143–10162.

[31] P. L. Houtekamer, H. L. Mitchell, Data assimilation using an ensemble kalman filter technique, Monthly Weather Review 126 (3) (1998) 796–811.

[32] T. M. Hamill, J. S. Whitaker, C. Snyder, Distance-dependent filtering of background error covariance estimates in an ensemble kalman filter, Monthly Weather Review 129 (11) (2001) 2776–2790.

[33] C. E. Shannon, A mathematical theory of communication, Bell system technical journal 27 (3) (1948) 379–423.

[34] P. Tichavsky, C. H. Muravchik, A. Nehorai, Posterior cramér-rao bounds for discrete-time nonlinear filtering, IEEE Transactions on signal processing 46 (5) (1998) 1386–1396.

[35] X. Shen, S. Liu, P. K. Varshney, Sensor selection for nonlinear systems in large sensor networks, IEEE Transactions on Aerospace and Electronic Systems 50 (4) (2014) 2664–2678.

[36] M. Hernandez, T. Kirubarajan, Y. Bar-Shalom, Multisensor resource deployment using posterior cramér-rao bounds, IEEE Transactions on Aerospace and Electronic Systems 40 (2) (2004) 399–416.

[37] B. Ristic, B.-N. Vo, Sensor control for multi-object state-space estimation using random finite sets, Automatica 46 (11) (2010) 1812–1818.

[38] E. K. Chong, C. M. Kreucher, A. O. Hero, Partially observable markov decision process approximations for adaptive sensing, Discrete Event Dynamic Systems 19 (3) (2009) 377–422.

[39] S. Kullback, R. A. Leibler, On information and sufficiency, The annals of mathematical statistics 22 (1) (1951) 79–86.

[40] O. Eldad, C. Claudel, Real-time computation of safe input sequences for unmanned aerial vehicles, Journal of Guidance, Control, and Dynamics (2016) 2578–2585.

[41] A. Tinka, M. Rafiee, A. M. Bayen, Floating sensor networks for river studies, IEEE Systems Journal 7 (1) (2012) 36–49.