2018 IEEE Conference on Decision and Control (CDC)
Miami Beach, FL, USA, Dec. 17-19, 2018

Geometric Reinforcement Learning Based Path Planning for Mobile Sensor Networks in Advection-Diffusion Field Reconstruction

Jie You and Wencen Wu

Abstract— **We propose a geometric reinforcement learning algorithm for real-time path planning for mobile sensor networks (MSNs) in the problem of reconstructing a spatial-temporal varying field described by the advection-diffusion partial differential equation. A Luenberger state estimator is provided to reconstruct the concentration field, which uses the collected measurements from a MSN along its trajectory. Since the path of the MSN is critical in reconstructing the field, a novel geometric reinforcement learning (GRL) algorithm is developed for the real-time path planning. The basic idea of GRL is to divide the whole area into a series of lattice to employ a specific time-varying reward matrix, which contains the information of the length of path and the mapping error. Thus, the proposed GRL can balance the performance of the field reconstruction and the efficiency of the path. By updating the reward matrix, the real-time path planning problem can be converted to the shortest path problem in a weighted graph, which can be solved efficiently using dynamic programming. The convergence of calculating the reward matrix is theoretically proven. Simulation results serve to demonstrate the effectiveness and feasibility of the proposed GRL for a MSN.**

I. Introduction

Many environmental advection-diffusion processes are often termed distributed parameter systems (DPSs) as their states depend not only on time, but also on spatial dynamics. Approximate mathematical modeling of DPSs often yields partial differential equations (PDEs) [1]–[3]. In environmental monitoring and pollution control, estimation and predication of these advection-diffusion processes find important applications. For example, a life-threatening contaminant source is dropped intentionally or unintentionally into a water reservoir. The release of dangerous materials from the source results in a plume. The real-time mapping of such a plume would allow tracking the source and containing possible adverse effects or at least reducing the impact of the release [3]–[5].

One of the greatest challenges in monitoring advection-diffusion processes is to achieve the state estimation of the processes using mobile sensor networks (MSNs) [1], [6], [7], which are collections of robotic agents with sensing, communication, and locomotion capabilities. Due to their mobility and adaptiveness to the environments, MSNs are ideal for the advection-diffusion field reconstruction mission, which often requires the exploration of relatively large regions. To increase the state estimation performance, there is a need to effectively solve path planning problems to guide agents to move along information-rich paths. There exist several model-based works dealing with the use of mobile agents for the state estimation of diffusion processes [1], [8], [9]. In general, the problem of selecting the number and locations of sensors and actuators for the control and state estimation of such systems is related to the combined control theory and computational approach, which is supported by a sound theory [8], [10], [11]. However, most of these studies have been proven to be effective only in offline schemes, which do not fit in many practical emerging scenarios.

In many realistic scenarios, it is desirable to provide feasible solutions for the path planning in a real-time fashion. In literature [12], [13], the dynamics of mobile agents is incorporated into the dynamics of the spatial-temporal process, which assists in the computation of guidance polices for mobile sensors deployed in spatial domains. However, when solving for the optimal paths, most of these works only consider the mapping error in the cost function [12], [13], which may make the solution easily stuck in local optima and fail to work in cases with multiple sources. Therefore, a reinforcement learning algorithm, which keeps a balance between exploration and exploitation, may provide a more appropriate solution in the long run [7], [14]. The basic idea of reinforcement learning is to obtain the optimal control strategy from the delayed rewards according to the observed state of the environment in a learning discrete map and to make a control strategy to select the actions to achieve the purpose [14]. To the best of our knowledge, the reinforcement learning based path planning for the advection-diffusion field reconstruction is not well studied.

In this paper, we address the problem of real-time path planning for MSNs in the advection-diffusion field reconstruction from the perspective of reinforcement learning. By using the collected measurements from a MSN along the moving trajectory, we first provide a Luenberger observer to achieve the reconstruction of an advection-diffusion field that takes a form of the copy of the advection-diffusion PDE with a stabilizing error term. Since the performance of state estimation heavily depends on the trajectory of the MSN, a novel geometric reinforcement learning (GRL) algorithm based on a time-varying reward matrix is further developed utilizing the length of the path and mapping error from the Luenberger observer. We divide the whole area into a series of lattice and exploit a specific reward matrix, which is simple and efficient for real-time path planning. In GRL, the reward matrix is adaptively updated based on the mapping error and the distance between any two grid points. The convergence of calculating the reward matrix is theoretically

The research work is supported by NSF grant CPS-1446461 and CMMI-1663073. Jie You and Wencen Wu are with the Department of Electrical, Computer, and Systems Engineering, Rensselaer Polytechnic Institute, Troy, NY 12180-3590, USA. youj@rpi.edu, wencen.wu@sjsu.edu

proven. Simulation results validate the effectiveness and feasibility of GRL for a MSN.

The problem is formulated in Section II. Section III presents the process state estimator. Section IV shows the geometric reinforcement learning for path planning using a MSN. Simulation results are presented in Section V. Conclusions and future work follow in Section VI.

II. Problem formulation

In this section, we formulate the problem of geometric reinforcement learning based path planning for mobile sensor networks for advection-diffusion field reconstruction.

A. Environmental models

It is well known that the atmospheric or waterborne pollution transport processes can be described by the following two-dimensional (2D) partial differential equation in a domain Ω:

$$\frac{\partial z(r,t)}{\partial t} = \theta \nabla^2 z(r,t) + v^T \nabla z(r,t),\ r \in \Omega, \tag{1}$$

where $z(r,t)$ is the concentration function, ∇ represents the gradient operator, ∇^2 represents the Laplacian operator, $\theta > 0$ is a constant diffusion coefficient, and v is a vector representing the flow velocity. The meaning of Equation (1) is that there is a net flow of substance from the regions with higher concentration of the substance to the ones with lower concentration. This type of PDEs in Equation (1) is widely used to described physical and engineering phenomena such as heat process, population dynamics, chemical reactors, fluid dynamics, etc., [6], [9], [15]. The parameters θ and v are assumed to be known, which can be identified using mobile sensor networks as described in our previous works [4], [15], [16]. In this work, we will focus on the path planning of MSNs for the advection-diffusion field reconstruction.

In practical applications such as environmental monitoring, the domain Ω is much larger than sensor dimensions so that the boundary can be modeled as a flat surface [9], [15]. Hence, the initial and Dirichlet boundary conditions for Equation (1) are assumed as [9], [15].

$$\begin{aligned} z(r,0) &= z_0(r), \\ z(r,t) &= z_b(r,t),\ r \in \partial\Omega, \end{aligned} \tag{2}$$

where $z_0(r)$ and $z_b(r,t)$ are the arbitrary initial condition and Dirichlet boundary condition, respectively.

B. Sensor dynamics

Consider a formation of N sensing agents forming a mobile sensor network moving in the field. The sensing agents have single-integrator dynamics given by $\dot{r}_i(t) = u_i(t), i = 1,2,...,N$. where $r_i(t) \subseteq \mathbb{R}^2$ is the position, and $u_i(t) \subseteq \mathbb{R}^2$ is the velocity of the ith agent, respectively. As the agent moves in a field, the position $r_i(t)$ is a function of the time t. For simplicity, we drop the variable t in $r_i(t)$ hereafter. We have the following assumption for the sensing agents.

Assumption II.1 *Every sensing agent is equipped with sensors to measure its location r_i and the concentration value $z(r_i,t)$.*

Under Assumption II.1, each agent equipped with sensors is able to provide the concentration measurement $p(r_i,t) = z(r_i,t) + n_i$, where n_i is assumed to be i.i.d Gaussian noise. We can employ a cooperative Kalman filter to reduce the measurement noise n_i, which is described in our previous works [4], [12], [15]. In this work, we focus on the path planning of MSNs. Thus, we assume the noise-free measurements and use $z(r_i,t)$ in equations hereafter.

The problem is formulated as:

1) Under Assumption II.1, develop an estimator that estimates the concentration $z(r,t)\ r \in \Omega$ based on the collected measurements using sensing agents moving in the advection-diffusion field.
2) Utilizing the collected measurements, build an efficient algorithm for real-time path planning for MSNs to improve the state estimation performance.

In the following, we first introduce a Luenberger state estimator to gradually achieve the advection-diffusion field reconstruction. To enable the real-time path planning, we employ a specific reward matrix containing the information of mapping errors and the geometric distance between the position of the formation center and the target location. By using this reward matrix, we design a novel GRL algorithm so that the agents will move along an information-rich path.

III. The process state estimator

In this section, we first introduce the formation control for MSNs, then develop a process state estimator to construct a map of the advection-diffusion processes using the measurements taken by mobile agents over time as input.

A. The view scope and the formation control of MSNs

When multiple coordinated agents move in the field, the agents can only measure the concentration value at finite discrete points and share the information with each other. Then we implement a cubic spline interpolation to fit the measurements so that we obtain the field values in a limited detection area or view scope at each time step. Let us denote the limited view scope as $\Gamma(t)$. The illustration of the view scope $\Gamma(t)$ with eight mobile agents is the grey shaded area shown in Fig. 1, in which the blue circles represent the eight agents at current time step and orange circles represent the eight agents at previous time step. The field value $z(r,t)\ r \in \Gamma(t)$ can be obtained through interpolating the measurements of mobile agents or running a cooperative Kalman filter [15]. Thus, it's reasonable to assume that the estimated field values within the view scope $\Gamma(t)$ are available to us at each time instant t. The field values in the time-dependent view scope $\Gamma(t)$ are then modelled by a spatial Dirac delta function,

$$y(r_s,t) = \int \delta(r - r_s) z(r,t) d\Omega,\ r_s \in \Gamma(t),\ r \in \Omega, \tag{3}$$

where r_s is an arbitrary spatial point in the time-dependent view scope $\Gamma(t)$ and $\delta(.)$ is the impulse function.

In this view scope $\Gamma(t)$, we require the agents to stay relatively close to each other to collect the measurements of the field. It is therefore important to have a formation control

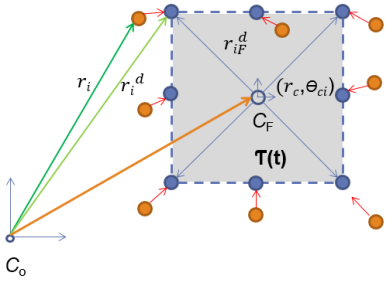


Fig. 1. A formation composed of eight agents with the center r_c.

for the MSN to maintain a desired formation. It should be noted that, $\Gamma(t)$ moves correspondingly, as the MSN moves along a certain trajectory. For simplify, we drop the variable t in $\Gamma(t)$ hereafter.

Motivated by [12], [17]–[19], we apply the following consensus tracking algorithm for each agent,

$$u_i = \dot{r}_i = \dot{r}_i^d - \varphi_i(r_i - r_i^d) - \sum_{j=1}^{N} a_{ij}[(r_i - r_i^d) - (r_j - r_j^d)], \quad (4)$$

where u_i is the control input of the ith robot, a_{ij} is the entry of the adjacency matrix which specifies the interconnection topology of the MSN, φ_i is a positive scalar, and r_i^d represents the desired position of the ith agent. r_i^d is denoted as $r_i^d = r_c + \mathbf{R}_i \cdot r_{iF}^d$, where $r_c = \frac{1}{N}\sum r_i$ is the position of the formation center, which will be designed using the geometric reinforcement learning method in the next section, r_{iF}^d represents the desired deviation of the ith agents relative to the formation center r_c and $\mathbf{R}_i$ is the rotation matrix from body frame to inertia frame. For example, in a 2D setting, $\mathbf{R}_i$ is defined as $\begin{bmatrix} cos(\theta_{ci}) & -sin(\theta_{ci}) \\ sin(\theta_{ci}) & cos(\theta_{ci}) \end{bmatrix}$, where θ_{ci} is the orientation of the body frame with respect to the inertia frame. Fig. 1 illustrates an example of the formation control composed of eight agents, where C_0 represents the inertial frame and C_F represents the body frame with the origin at the formation center r_c with an orientation θ_{ci} relative to C_0.

Remark III.1 *It should be noted that the shape of the formation does not need to be fixed. The optimal formation shapes can be determined by different criteria such as graph connectivity, energy efficiency, etc. [17]–[20]. Moreover, with the changing dynamics of the environment, the optimal formation shape can change over time. Therefore, when the agents switch from one formation to another, a collision avoidance scheme should be employed, which is also referred as the collision-free motion control. Interested readers can refer to [17]–[19] for additional insight.*

B. Luenberger state estimator

By applying the formation control, a MSN can cooperatively provide the measurements of field values in the view scope Γ determined by the desired formation at each time step. In this section, we will show how to use this information to achieve the field reconstruction.

The Luenberger estimator developed in [9] is modified to account for the trajectories of MSNs to estimate the concentration state over the entire spatial domain. The basic idea is employing an estimator in the form of a copy of the system, plus a stabilizing error term. The proposed state estimator takes the form,

$$\frac{\partial \hat{z}(r,t)}{\partial t} = \theta\nabla^2\hat{z}(r,t) + v^T\nabla\hat{z}(r,t) + \gamma\cdot(y(r_s,t) - \hat{z}(r_s,t)), \quad (5)$$

where $r_s \in \Gamma$ and $r \in \Omega$. The hat notation indicates that $\hat{z}(r,t)$ is the estimate of the concentration $z(r,t)$, $\gamma > 0$ is constant, and $\hat{z}(r_s,t)$ provides the state estimator prediction of the concentration value in the view scope Γ. Similar to the definition of $y(r_s,t)$ in Equation (3), $\hat{z}(r_s,t)$ is modelled by a spatial Dirac delta function as follows,

$$\hat{z}(r_s,t) = \int \delta(r - r_s)\hat{z}(r,t)d\Omega, \; r_s \in \Gamma, \; r \in \Omega, \quad (6)$$

where $\delta(.)$ is the impulse function. In fact, $y(r_s,t) - \hat{z}(r_s,t)$ is the stabilizing error in the view scope Γ.

In this paper, the trajectory design of the MSN in the next section will be based on the state estimator error, or mapping error, that is, $e(r,t) = z(r,t) - \hat{z}(r,t), r \in \Omega$. Using Equations (1) and (5), we can obtain the dynamics of $e(r,t)$,

$$\frac{\partial e(r,t)}{\partial t} = \theta\nabla^2 e(r,t) + v^T\nabla e(r,t) - \gamma\cdot(y(r_s,t) - \hat{z}(r_s,t)), \quad (7)$$

where $r_s \in \Gamma$ and $r \in \Omega$. By applying the above error dynamics in Equation (7), the convergence proof of the Luenberger estimator in Equation (5) can be readily obtained by using the Lyapunov function $V = -\langle e(r,t), A_{cl}\cdot e(r,t)\rangle$, where A_{cl} is a closed-loop operator with symmetry and coercivity properties. There are several literatures [9] include this part of proof. We omit the detailed proof here due to space limitation. Please kindly refer to the related references [9], [12] for more details.

IV. Geometric Reinforcement Learning for Path Planning

Once the mapping errors in the view scope are estimated sequentially over time, GRL is designed to determine the path of the the formation center r_c of the MSN.

A. Path planning

The decision in reinforcement learning depends on a set of actions decided by the reward matrix G. The reward matrix G is essential for a reinforcement learning based path planning problem, which can be used to find the optimal path from a given point to the target point with optimal distance and minimum integral mapping error. The real-time path planning problem can be efficiently solved by adaptively updating this specific remard matrix. In order to represent the reward matrix, the computational domain Ω is discretized into a set of $N_X \times N_Y$ grid points, where $\Omega' = [0, N_X] \times [0, N_Y]$. Then, the dimension of G is $N_X \times N_Y$ and the number of grid points in Ω' is denoted as $N_{node} = N_X \times N_Y$.

Suppose that the formation center r_c starts at the start point r_S and ends at the target point r_T. The target point r_T can be considered as a base station, where agents can charge energy. The goal of GRL is to find the optimal sequential positions

$r_c^0, r_c^1, \cdots r_c^\tau$ for the MSN by optimizing the reward matrix G, which can be found as follow,

$$r_c^{t+1} = \arg\min_{r_c^t}(G), t \in [0, \tau], \tag{8}$$

where τ is the terminal time, $r_c^0 = r_S$, and $r_c^\tau = r_T$. In the following, we will show how to construct the reward matrix G based on the optimization criteria for path planning.

B. Optimization criteria for path planning

We first introduce our optimization criteria for path planning. The goal of path planning for MSNs is to minimize both the mapping error and the length of the path. The integral mapping error in the view scope can be defined as,

$$M = \int_\Gamma e^2(r,t)d\Gamma. \tag{9}$$

We denote the geometric distance T, which is the length of the path from the start point r_S to the end point r_T,

$$T = \int_S ds, \tag{10}$$

where S is the path of the formation center r_c. Assuming that the speed of agent is fixed, minimizing the length of the path is then equivalent to minimizing the traveling time along the path. Therefore, the consideration of geometric distance T makes our scheme energy-efficient. Moreover, for a MSN, the geometric distance is a very valuable element for path planning when only partial information of the mapping errors is available. In this case, the mobile agents can use the distance information to guide them to escape from local optimums of mapping error, which increases the robustness of the proposed algorithm. Note that, as the MSN moves along a trajectory, Γ moves correspondingly.

By combining the mapping error in Equation (9) and the length of the path in Equation (10), the optimization objective can be written as

$$S^* = \arg\min_S(T + K \times M), \tag{11}$$

where S^* is the designed optimal path and $K \geq 0$ controls the degree of the influence of the mapping error and the geometric distance. A large K means that the MSN has the tendency of moving towards directions that would reduce the mapping error instead of directions that result in the shorter path towards the target point. K should be selected appropriately to keep a good balance between the accuracy of the field reconstruction and the length of the path.

C. The reward matrix based on the mapping error and geometric distance

In Section IV.B, we illustrate the continuous version of the optimization criteria of GRL (11). Since the field Ω is divided into a series of lattice ($N_X \times N_Y$ grid points), we will introduce the weight between one grid point and its neighbor points, which serves as the discretized representation of our optimization criteria (11). By using this weight, the reward matrix G that is critical in GRL can be effectively generated.

To find the next reward from the current reward, we need know the weight or relationship between one grid point and its neighbor points. Then, the reward matrix can be updated according to the current reward and the weight as follows,

$$G_{r_j} = W_{r_j,r_{n,h}} + G_{r_{n,h}}, j \in 1, 2, \cdots N_{node}, \tag{12}$$

where G_{r_j} is the element of reward matrix G at point r_j and $W_{r_j,r_{n,h}}$ is the weight between grid point r_j and its neighbor point $r_{n,h}$, $h \in \{1, 2, \cdots, 8\}$. To update the reward matrix G efficiently, we assume the further action in each step is limited to eight directions as shown in the Fig. 2. More specifically, the next position of r_j is limited to the eight grid points of the neighborhood $r_{n,h}$, $h \in \{1, 2, \cdots, 8\}$.

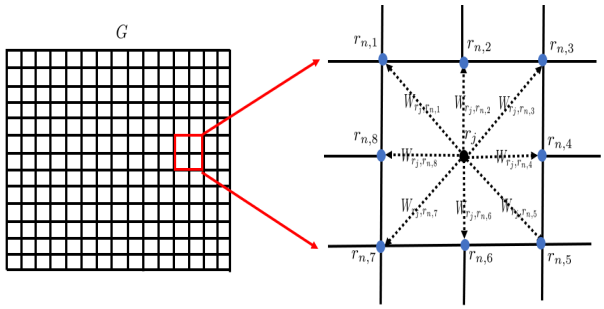


Fig. 2. Illustration of the reward matrix G and the eight directions from a point to its neighbors. The blue circles are the eight neighbor grid points $r_{n,h}$ of r_j.

Considering our optimization criteria in Equation (11), this weight $W_{r_j,r_{n,h}}$ should contain the information of both the geometric distance T and mapping error M. Then, the weight for the path between one point and its neighbor points is computed by the distance and the mapping error measure,

$$W_{r_j,r_{n,h}} = d_{r_j,r_{n,h}} + K \times f_z(e'(r_{n,h},t)), \tag{13}$$

where $d_{r_j,r_{n,h}} = \|r_j - r_{n,h}\|_2$ is the distance between r_j and $r_{n,h}$. It should be pointed out that $d_{r_j,r_{n,h}}$ and $f_z(e'(r_{n,h},t))$ are the discretized representation of T and M in Equation (11), respectively. $e'(r_{n,h},t)$ is the normalized mapping error, which is shown as follow $e'(r_{n,h},t) = \frac{e^2(r_{n,h},t) - e^2_{max}}{e^2_{max} - e^2_{min}}, r_{n,h} \in \Gamma$, where $e(r_{n,h},t)$ is the mapping error at position $r_{n,h}$, which can be obtained from the Luenburger observer in Equation (7), e_{max} and e_{min} are the maximum and minimum mapping errors in the view scope Γ. And $f_z(.)$ is a Z-shaped function shown in Fig 3, which is shown as,

$$f_z(\rho) = \begin{cases} 1 & \text{if } \rho < 0, \\ 1 - 2\rho^2, & \text{if } 0 \leq \rho \leq \frac{1}{2}, \\ 2(\rho - 1)^2, & \text{if } \frac{1}{2} \leq \rho \leq 1, \\ 0, & \text{if } \rho > 1. \end{cases} \tag{14}$$

We use $f_z(.)$ to reverse the influence of normalized mapping error $e'(r_{n,h},t)$. More specifically, the larger the normalized mapping error $e'(r_{n,h},t)$ is, the smaller weight we get. Then, we can readily convert a path planning problem to a shortest path problem in a weighted graph, which can reduce the computational complexity of the graph search algorithm. It should be noted that, different from existing works on reinforcement learning with static reward matrix [7], [14], the

designed reward matrix G based on an advection-diffusion field is time varying. Therefore, as the mobile agents move, the weight matrix W in the scope view will be dynamically updated to track the reward matrix according to (13).

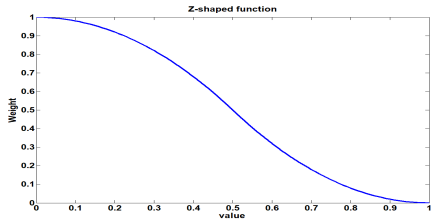


Fig. 3. The illustration of the Z-shaped function $f_z(\cdot)$.

D. The geometric reinforcement learning algorithm

The key idea of GRL is how to calculate the reward matrix G for MSN, which can be used to find the real-time optimal path from a given point r_S to a target point r_T with optimal geometric distance and integral mapping errors.

In this paper, we employ the Bellman Ford algorithm to update $G^t_{r_j}$, which is the element of reward matrix G at point r_j and at time step t. The idea of the Bellman Ford algorithm is based on dynamic programming [7], [14]. The procedure of the Bellman Ford algorithm is described in Algorithm 1.

Algorithm 1 Online Bellman Ford algorithm for real-time path planning

1: Assign $G^0_{r_T}$ at the end point r_T with 0, and the other points with $+\infty$ (a big value).
2: **for** t = 1 to N_{node} **do**
3: **for** Each edge **do**
4: Update G_{r_j} by its neighbors points as: $G^{t+1}_{r_j} = \min\{G^t_{r_j}, W_{r_j,r_{n,h}} + G^t_{r_{n,h}}\}$.
5: **end for**
6: **end for**

We can observe from the above algorithm that the time complexity of this algorithm is $O(8(N_{node})^2) = O((N_{node})^2)$ and the space complexity of this algorithm is $O(N_{node})$. In the following, we show that by running Algorithm 1, we can find the optimal path from a given point r_S to the end point r_T with optimal geometric distance and integral mapping errors.

Proposition IV.1 *Consider the weighted directed graph with weight in Equation* (13)*. Given the start point r_S and the target point r_T, the optimal path from r_S to r_T can be found by running Algorithm 1.*

Proof: In the proposed graph, each point connects with eight neighbor points. Thus, r_S is reachable from r_T no more than N_{node} steps. Next, from Equation (13), we can observe that all of weights $W_{r_i,r_{n,h}}$ are positive. That means there are no negatives cycles in the proposed weight graph. Then, we can prove the proposition by induction : 1) When $r_S = r_T$, $G^t_{r_T}$ equals to 0, since we do not have negative cycles. 2) Supposing the mth point r_m leads to an optimal path with r_T as the target, we have an optimal path containing no more than m edges. Now we just need to prove the $m+1$th point leads to the same target point using our algorithm. From Equation (13), we can find that

$$G^t_{r_{m+1}} = \min\{W_{r_k,r_{m+1}} + G^t_{r_k}\},\ k \le m. \tag{15}$$

As $G^t_{r_k}$ is already calculated for any other points, the optimal path for $m+1$th point can be computed using Equation (15). Thus, the procedure can find the optimal path from the start point to the target point. ■

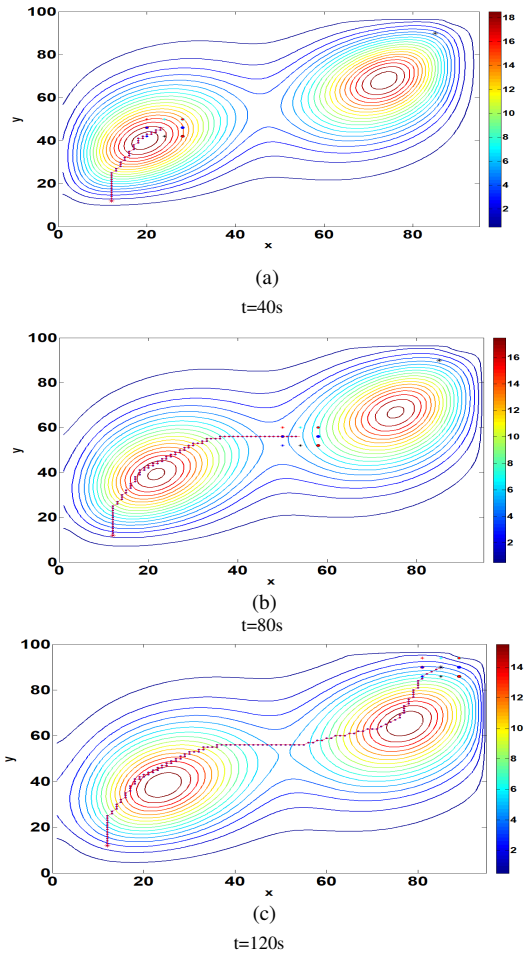
The above procedure is based on the greedy algorithm to find each position of a path. When the mobile agents move, we should update the weight matrix W according to the measured mapping errors in the view scope Γ and re-compute the path in a real time fashion. It also should be noted that after the MSN reaches the target point r_T, the data obtained are then analyzed and used to select the next target point. We can repeat the above path planning until we obtain the satisfactory field reconstruction performance.

V. Simulation Result

To demonstrate the performance of the proposed GRL path planning scheme, we consider a well-known advection-diffusion process in Equation (1) with the diffusion coefficient $\theta = 0.6$ and the flow velocity $v = [-0.4, 0.2]$. The initial condition is illustrated in Fig. 4, in which there are two maximum values at points $(20,40)$ and $(75,70)$. The whole domain is a rectangular area with $0 \le x \le 100$, $0 \le y \le 100$. We implement an implicit ADI finite-difference scheme in MATLAB, with 100-by-100 grid lattices to generate the concentration field. The computational time step of $0.1s$ is chosen for the PDE simulation and the weight K is set to 0.5. In the simulation, we deploy eight sensing agents represented by the colored stars and circles, which are controlled to maintain the desired formation shown in Fig. 4. In Fig. 4, the contour represents the level curves of the advection-diffusion field and the dotted colored line shows the designed trajectory for the formation center of the MSN. To achieve a fair validation, the simulations are performed using three different start points and end points, which are shown in the dotted blue, red, and black lines in Fig. 5. We can observe that the designed trajectories can go through the two local optimums and avoid getting stuck in the local optimums. To illustrate the efficiency of the proposed state estimator, we further calculate the root mean squared error (RMSE) of the state estimation in the whole spatial domain corresponding to the simulation shown in Fig. 4. The RMSE is shown in Fig. 6. As expected, RMSE is gradually decreasing as the sensing agents collect more measurements of the process.

VI. Conclusions and Future Work

This paper proposes a new geometric reinforcement learning method to solve the path planning problem for a MSN to achieve the reconstruction of an advection-diffusion field. By designing a specific reward matrix, the proposed GRL keeps a good balance between the field reconstruction performance and path length. Compared with other methods, the proposed GRL leads to a simple path planning algorithm, which can



(a)
t=40s

(b)
t=80s

(c)
t=120s

Fig. 4. The illustration of the evolution of the field and the real-time designed trajectory with start point $(12, 12)$ and end point $(85, 90)$.

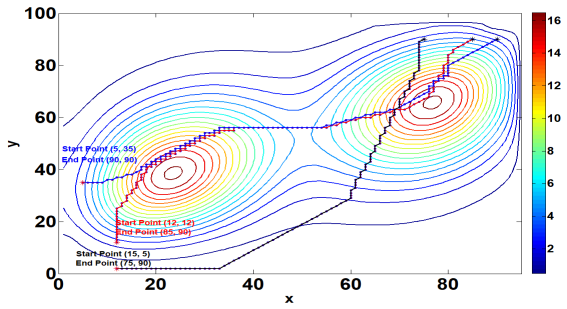


Fig. 5. The illustration of real-time path planning with three different start and end points. The dotted blue, red, and black lines are the resulting trajectories. The dotted blue line starts at (5, 35) and ends at (90, 90). The dotted red line starts at (12, 12) and ends at (85, 90). The dotted black line starts at (15, 5) and ends at (75, 90).

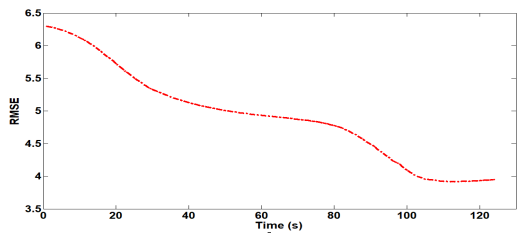


Fig. 6. The RMSE of the state estimation error.

provide a feasible solution within a short time. Theoretical justifications are provided for the reward matrix calculation. Our future work will focus on setting adaptive K for more efficient path planning, extending the algorithm to other types of PDEs, and applying the algorithm to real mobile robot testbed to verify the effectiveness.

REFERENCES

[1] S. Martinez and F. Bullo, "Optimal sensor placement and motion coordination for target tracking," *Automatica*, vol. 42, no. 4, pp. 661–668, 2006.

[2] R. Ghez, *Diffusion Phenomena*. Kluwer Academic/ Plenum Publishers, 2nd edition, 2001.

[3] D. Uciński and M. Patan, "Sensor network design for the estimation of spatially distributed processes," *Int. J. Appl. Math. Comput. Sci.*, vol. 20, no. 3, pp. 459–481, 2010.

[4] J. You, Y. Zhang, M. Li, K. Su, F. Zhang, and W. Wu, "Cooperative parameter identification of advection-diffusion processes using a mobile sensor network," in *American Control Conference*, no. 3230-3236, 2017.

[5] E. Fiorelli, N. E. Leonard, P. Bhatta, and et al., "Multi-AUV control and adaptive sampling in monterey bay," *IEEE Journal of Oceanic Engineering*, vol. 31, no. 4, pp. 935–948, 2006.

[6] L. Z. Guo, S. A. Billings, and H. L. Wei, "Estimation of spatial derivatives and identification of continuous spatio-temporal dynamical systems," *Internal Journal of Control*, vol. 79, no. 9, pp. 1118–1135, 2006.

[7] B. Zhang, Z. Mao, W. Liu, and J. Liu, "Geometric reinforcement learning for path planning of uavs," *J. Intel Robot Syst*, vol. 77, pp. 391–409, 2015.

[8] Z. Tang and U. Ozguner, "Motion planning for multitarget surveillance with mobile sensor agents," *IEEE Transactions on Robotics*, vol. 21, no. 5, pp. 898–908, 2005.

[9] M. A. Demetriou, N. A. Gatsonis, and J. R. Court, "Coupled control-computational fluids approach for the estimation of the concentration form a moving gaseous source in a 2-D domain with a Lyapunov-guided sensing aerial vehicle," *IEEE Transactions on Control Systems Technology*, vol. 22, no. 3, pp. 853–867, 2014.

[10] D. Uciński, *Optimal measurment methods for distributed parameter system identification*. Boca Raton, FL: CRC Press, 2004.

[11] R. Olfati-Saber and J. S. Shamma, "Consensus filters for sensor networks and distributed sensor fusion," in *Proc. of 44th IEEE Conf. on Decision and Control and European Control Conference*, 2005, pp. 6698–6703.

[12] J. You and W. Wu, "Sensing-motion co-planning for reconstructing a spatial distributed field using a mobile sensor network," in *Proc. 56th IEEE Conference on Decision and Control*, 2017, pp. 3113–3118.

[13] M. A. Demetriou, "Guidance of mobile actuator-plus-sensor networks for improved control and estimation of distributed parameter system," *IEEE Transations on Automatic Control*, vol. 55, no. 7, pp. 1570–1584, 2010.

[14] F. Lewis and D. Vrabie, "Reinforcement learning and adaptive dynamic programming for feedback control," *IEEE circuits and systems magazine*, vol. 9, no. 3, pp. 32–50, 2009.

[15] J. You and W. Wu, "Online passive identifier for spatially distributed systems using mobile sensor networks," *IEEE Transactions on Control Systems Technology*, vol. 25, no. 6, pp. 2151–2159, 2017.

[16] J. You, F. Zhang, and W. Wu, "Cooperative filtering for parameter identification of diffusion processes," in *Proc. of IEEE Conference on Decision and Control*, 2016, pp. 4327–4333.

[17] F. Zhang and N. E. Leonard, "Cooperative control and filtering for cooperative exploration," *IEEE Transactions on Automatic Control*, vol. 55, no. 3, pp. 650–663, 2010.

[18] W. Wu and F. Zhang, "Robust cooperative exploration with a switching strategy," *IEEE Transactions on Robotics*, vol. 28, no. 4, pp. 828–839, 2012.

[19] W. Ren and R. W. Beard, *Distributed consensus in multi-vehicle cooperative control, communications and control engineering series*. London: Springer-Verlag, 2008.

[20] P. Yang, R. Freeman, G. Gordon, K. Lynch, S. Srinivasa, and R. Sukthankar, "Decentralized estimation and control of graph connectivity for mobile sensor networks," *Automatica*, vol. 46, pp. 390–396, 2010.