Distributed and Collision-Free Coverage Control of a Team of Mobile Sensors Using the Convex Uncertain Voronoi Diagram

Jun Chen and Philip Dames

Abstract— **In this paper, we propose a distributed coverage control algorithm for mobile sensing networks that can account for bounded uncertainty in the location of each sensor. Our algorithm is capable of safely driving mobile sensors towards areas of high information distribution while having them maintain coverage of the whole area of interest. To do this, we propose two novel variants of the Voronoi diagram. The first, the convex uncertain Voronoi (CUV) diagram, guarantees full coverage of the search area. The second, collision avoidance regions (CARs), guarantee collision-free motions while avoiding deadlock, enabling sensors to safely and successfully reach their goals. We demonstrate the efficacy of these algorithms via a series of simulations with different numbers of sensors and uncertainties in the sensors' locations. The results show that sensor networks of different scales are able to safely perform optimized distribution corresponding to the information distribution density under different localization uncertainties.**

I. INTRODUCTION

Coverage control is the problem of optimally covering a space using a collection of sensors. In a dynamic setting, this involves reactively adjusting the distribution of sensors over the mission space as new information is collected. This problem has been widely studied by roboticists in robot surveillance, deployment, multi-target search and tracking, *etc*. For example, consider a team of drones tasked with tracking the spread of a forest fire in a mountainous area. Here the team must trade off between remaining in areas with known fires to collect information about the current conditions and maintaining surveillance of the whole area to detect the birth of new fires. While they do this, the robots must simultaneously account for the uncertainty in their positions to properly track the fire and to maintain a safe distance between robots at all times to avoid collisions.

Both centralized and distributed methods have been considered to solve such problems. A number of authors have studied coverage control strategies. Hussein *et al.* [1] proposed a centralized cooperative coverage control strategy with guaranteed collision avoidance to achieve a desired effective coverage level of each point in the search domain. While events happening at each point may be detected with some level of confidence, the probability density of events happening is assumed to known a priori instead of being detected online by sensors.

Distributed algorithms often scale better to large networks and over large geographic regions than centralized approaches, leading to a rising amount of research interest. Other have proposed gradient-based distributed coverage control schemes to maximize the probability of detecting randomly occurring events in a mission space using a team of mobile sensors [2], [3]. However there is no guarantee of sensor collision since sensor dimension were not taken into consideration.

Voronoi-based methods [4] are among the most popular choices to solve distributed coverage control problems in recent years. Lloyd's algorithm iteratively drives each sensor in a convex environment towards the weighted centroid of its local Voronoi cell where the sensor detection probability is optimal [5], [6]. Collision avoidance is guaranteed for point sensors since cells never overlap, and each sensor only moves in its own cell. This can be extended to sensors with finite size using buffered Voronoi cells, which shrink each cell to ensure collision avoidance [7]. By encoding the information distribution, which is a time-varying density function, as the importance weighting function in Lloyd's algorithm, sensors are able to reach their optimized location for detection. One example of an information density function could be the probability density function of target positions the sensors aimed at tracking over the area of interest. Schwager *et al.* [8] extended Lloyd's algorithm and derived a control law enabling sensors to approximate the information density function from measurements while maintaining or seeking a near-optimal sensing configuration. Schwager *et al.* [9] later proposed a controller using an adaptive control architecture for sensors to learn a parameterized model of that measured distribution in the environment. Dames [10] used the probability hypothesis density (PHD) as the weighting function in Lloyd's algorithm to guide a team of sensors towards areas of high target density detected by on board sensors.

All of the above-mentioned coverage control strategies assume that the locations of the mobile sensors are perfectly known. This is a strong assumption which is not true in practice, though occasionally localization error can be ignored to some degree. To account for uncertainty in the positions of points, researchers have recently proposed the uncertain Voronoi diagram (UV diagram), or fuzzy Voronoi diagram, an extended Voronoi partitioning strategy that divides uncertain spatial databases by using a Gaussian distribution to model the uncertainty [11]–[13]. However, none of these works have been applied to the task of collision avoidance or decentralized control. Most recently, two variants of the buffered Voronoi diagram were proposed for multi-agent collision avoidance with localization uncertainty, the buffered uncertainty-aware Voronoi cell (B-UAVC) [14] and the probabilistic buffered Voronoi cell (PBVC) [15]. Neither

J. Chen and P. Dames are with the Department of Mechanical Engineering, Temple University, Philadelphia, PA 19122, USA {jchen,pdames}@temple.edu

This work was supported by NSF Grant IIS-1830419.

of these methods makes any guarantees for coverage during a search task.

In this paper, we introduce a novel coverage control method inspired by these ideas to construct a convex uncertain Voronoi (CUV) diagram over the mission space, which accounts for uncertainty in the locations of sensors and iteratively drives each sensor to the weighted centroid of its CUV cell using Lloyd's algorithm. Simultaneously, we propose a collision avoidance algorithm which guarantees safety and avoids "deadlock", the phenomenon where sensors block each other from moving to their respective goals. Our goal is to propose a distributed, collision-free control strategy that leads sensors to congregate in areas with higher information density while maintaining full coverage of the search space.

II. Problem Formulation

A sensing network with n sensors $S = \{s_1, \dots, s_n\}$ is performing surveillance in an open convex polygonal environment $A \subset \mathbb{R}^2$. Let $Q = \{q_1, \dots, q_n\}$ denote the true sensor locations. The dynamics of each sensor are modeled by the first order equation $\dot{q}_i = u_i$, where u_i is the control input at time t. A time-varying information density function $\phi(x)$ indicates the information content at each point $x \in A$. Note that we remove time t for simplicity of notation in this paper. The information density function could be a probability density function (PDF) of events of interest occurring in a certain area (which can be updated recursively using Bayesian filters [16]) or some other density functions such as the PHD, which can be maintained over time using the distributed PHD filter [10].

A. Lloyd's Algorithm

We let $\|x - q_i\|$ denote the Euclidean distance between a point $x \in A$ and the location of sensor s_i. Let $f(\|x-q_i\|)$ be a monotonically increasing function, which may be used to quantify the cost of sensing due to degradation of a sensor's ability to measure events with increasing distance. As defined in [5], a partition of A is a collection of n polygons $\mathcal{W} = \{W_1, \dots, W_n\} \subset \mathbb{R}^2$ with disjoint interiors and whose union is A.

At time t, the locational optimization functional is defined as follows:

$$\mathcal{H}(Q, \mathcal{W}) = \sum_{i=1}^{n} \int_{\mathcal{W}_i} f\big(\|x - q_i\|\big)\phi(x)dx, \tag{1}$$

where $\mathcal{W}_i$ is dominance region of sensor s_i, *e.g.*, the region that sensor s_i is responsible for. The goal is for the team to minimize the functional (1), both with respect to the partition set $\mathcal{W}$ and the sensor positions Q. Minimizing $\mathcal{H}$ with respect to $\mathcal{W}$ induces a partition on the environment $V_i = \{x \mid i = \arg\min_{k=1,\dots,n} \|x - q_k\|\}$. In other words, V_i is the collection of all points that are the nearest neighbor of s_i. This is the Voronoi partition, as Figure 2a shows, and these V_i are the Voronoi cells, which are convex by construction. Minimizing $\mathcal{H}$ with respect to Q leads each sensor to the weighted centroid of its Voronoi cell [5], that is

$$q_i^* = \frac{\int_{V_i} x\phi(x)\,dx}{\int_{V_i} \phi(x)\,dx}, \tag{2}$$

The dynamic version of Lloyd's algorithm continuously finds the control input

$$u_i = -k_{\text{prop}}(q_i - q_i^*), \tag{3}$$

where $k_{\text{prop}} > 0$ is a positive gain. By following this control law, the sensors asymptotically converge to the weighted centroids of their Voronoi cells. Lloyd's algorithm therefore drives each sensor to the position with maximum detection probability iteratively over the entire mission space.

B. Localization Uncertainty Regions

In this paper, we assume that each sensor s_i only knows its estimated position $\hat{q}_i$ and the associated covariance matrix Σ_i. We assume that this covariance does not change over time for simplicity, but may vary between different sensors. We find the eigendecomposition of Σ_i:

$$\Sigma_i = R\Lambda R^{-1}, \tag{4}$$

where R is the square 2×2 matrix and Λ is the diagonal matrix with eigenvalues λ_1, λ_2 on the diagonal. We define the localization uncertainty region of sensor s_i to be $B_i = B(\hat{q}_i, r_i)$, which is a ball centered at $\hat{q}_i$ with radius

$$r_i = c \max_j \lambda_j \tag{5}$$

where c is a positive constant. The probability of sensor s_i being located within this region is then

$$p(q_i \in B_i) = \int_{B_i} \frac{\exp\left\{-\frac{1}{2}(x - \hat{q}_i)^T \Sigma_i^{-1}(x - \hat{q}_i)\right\}}{2\pi \det(\Sigma_i)^{\frac{1}{2}}} dx, \tag{6}$$

In this paper we use $c = 3$ so that the region covers at least 99.73% (minimum achieved when $\lambda_1 = \lambda_2$) of all possible locations of s_i, though any other level set of the covariance matrix could be used to guarantee a desired level of confidence.

We use a Gaussian distribution in the discussion above as it is a standard choice for modeling localization uncertainty. However, our approach could also be used with other non-Gaussian distributions so long as one can define a bounded, circular region. This is required to construct the CUV diagram, as we will see in Section II-C.

C. Uncertain Voronoi Diagram

Xie *et al.* [11] defined the uncertain Voronoi (UV) diagram and proposed a centralized method to construct the UV diagram over a convex region. In this paper, we define a UV cell in a similar way as follows:

Definition 1 (UV Cells)**.** The UV cell of a sensor s_i is $U_i = \{x \mid p(i = \arg\min_{k=1,\dots,n} \|x - q_k\|) > 0\}$, the collection of points in A such that s_i has a nonzero probability to be the nearest sensor to each point $x \in U_i$.

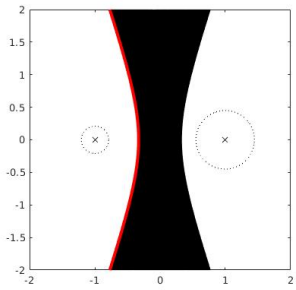


Fig. 1. Figure shows the UV edge $E_{\text{right}}(\text{left})$ of sensor s_{right} with respect to sensor s_{left} (red curve). The X's at $(1, 0)$ and $(-1, 0)$ are the estimated locations of s_{right} and s_{left}, respectively. Dashed circles represent the localization uncertainty regions of the sensors. The black area contains all of the points whose nearest sensor is uncertain.

A UV cell U_i contains all possible Voronoi cells generated from all possible combinations of the positions of sensor s_i and each of its neighbors. Therefore, by assigning each sensor to be responsible for all information in its UV cell, the coverage of the whole environment is guaranteed even with the localization uncertainty of sensors. In other words, no matter where each sensor is actually located within in uncertainty region, the union of all of the UV cells will be equal to the entire environment, $\cup_i U_i = A$.

Let $\text{dist}_{\max}(\hat{q}_i, x)$ and $\text{dist}_{\min}(\hat{q}_i, x)$ denote the maximum and minimum distances between sensor s_i's estimated location $\hat{q}_i$ and x, respectively. Constructing the UV cell for sensor i is equivalent to finding a collection of points in A between s_i and each other sensor $s_j, i \neq j$ that meet the following condition:

$$\text{dist}_{\max}(\hat{q}_i, x) = \text{dist}_{\min}(\hat{q}_j, x). \tag{7}$$

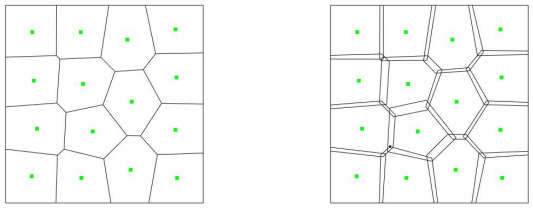
The resulting dividing line from (7) between s_i and s_j, denoted by $E_i(j)$, is called a UV-edge of s_i with respect to s_j. It is proved in [11] that for a circular uncertainty region, $E_i(j)$ is a hyperbola, as Figure 1 shows. Without loss of generality, let the center of the hyperbola be at the midpoint of the line segment connecting $\hat{q}_i$ to $\hat{q}_j$ and that this line segment is parallel with the x axis. The hyperbola is then given by

$$\frac{x^2}{a^2} - \frac{y^2}{b^2} = 1 \tag{8}$$

where

$$a = \frac{r_i + r_j}{2} \qquad c = \frac{\|\hat{q}_i - \hat{q}_j\|}{2} \qquad b = \sqrt{c^2 - a^2} \tag{9}$$

Note that the locations of the sensors are located at $(c, 0)$ and $(-c, 0)$, which correspond to the foci of the hyperbola. Generally when Σ_i is a non-singular non-diagonal matrix, Xie *et al.* defined in their work [11] a "possible region" and used it for centralized UV diagram construction, which inspires our proposed distributed CUV diagram construction algorithm in section III.



(a) Voronoi diagram (b) CUV diagram

Fig. 2. A Voronoi diagram and a CUV diagram with 15 cells. Green markers are estimated sensor locations. Note that CUV cells are a superset of the original Voronoi cells and that CUV cells overlap with one another.

III. Distributed Control with Localization Uncertainty

A. The CUV Diagram and Its Construction

We give the definition of a CUV diagram as follows:

Definition 2 (CUV cell)**.** A convex uncertainty Voronoi (CUV) cell of sensor s_i, denoted by C_i, is the convex hull of its UV cell, U_i.

An example of the CUV diagram is shown in Figure 2b.

Definition 3 (CUV neighbors)**.** The CUV neighbor set for sensor s_i is denoted N_i and contains all sensors $s_j, j \neq i$ such that $E_i(j)$ is on the boundary of the UV cell U_i.

Proposition 1. The CUV neighbors and the Voronoi neighbors of a sensor are identical.

Proof: Since U_i is the union of all possible Voronoi cells of s_i, UV edges $E_i(j)$ and $E_j(i)$ are contours of the union of all possible Voronoi edges between s_i and s_j. Thus, the UV neighbors and the Voronoi neighbors of a sensor are identical. Since the CUV cells are convex hulls of the UV cells, the CUV neighbors and the Voronoi neighbors of a sensor are also identical. □

Algorithm 1 shows how each sensor can construct its CUV cell using only local information, thus enabling a team to compute the CUV diagram in a distributed manner. Specifically, each sensor needs to know the estimated locations and uncertainty region radii of each of its Voronoi neighbors, *i.e.,* the set of agents whose Voronoi cells share an edge with its cell. This is a standard assumption in distributed multi-agent control algorithms [10]. Each sensor s_i computes the UV edges with its own estimated location and these received from neighbors. These UV edges sequentially divide the original mission space and discard the portion not containing s_i after each division. The CUV cell of s_i is then constructed by computing the convex hull of the remaining area. Figure 3 shows a schematic diagram of a Voronoi, UV, and CUV cells for a sensor.

B. Collision Avoidance

1) Collision Avoidance Regions (CARs): By construction, Voronoi cells have disjoint interiors and are convex. Thus, if point sensors have perfect knowledge of their locations

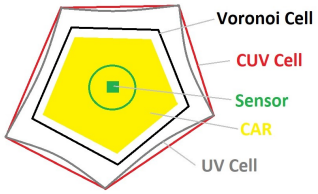


Fig. 3. Figure showing a sensor's estimated location with its localization uncertainty region (green) and its Voronoi cell (black), UV cell (gray), CUV cell (red), CAR (yellow).

Algorithm 1 Distributed Construction of CUV Cells

for Each sensor s_i **do**
  Get estimated location $\hat{q}_i$
  Find the neighbor set N_i
  Initialize $A_i = A$
  for s_j in N_i **do**
    Receive $\hat{q}_j$ and r_j from s_j
    Compute UV edge $E_i(j)$ using (8)
    $A_i \leftarrow \{x \in A_i \mid x, \hat{q}_i \text{ on the same side of } E_i(j)\}$
  end for
  $C_i \leftarrow$ convex hull(A_i)
end for

and never move outside their responding cell, it is naturally guaranteed that they move without collision (*e.g.*, being at the same location). However, this is not the case for CUV-based control with localization uncertainty. In fact, CUV cells always overlap with their neighbors as long as the uncertainty region for any sensor is non-empty. Thus, we want each sensor to perform motion only within a region that ensures no collisions with other sensors, which we call a collision avoidance region (CAR), shown in Figure 3.

Definition 4 (CAR)**.** The collision avoidance region (CAR) for sensor s_i, denoted M_i, is the collection of points inside of its Voronoi cell V_i (constructed using the estimated positions of s_i and each neighbor in N_i) that are at least a distance $r_i + r_{\text{buffer}}$ away from any boundary of V_i, where r_i is the radius of s_i's localization uncertainty region and r_{buffer} is a small buffered distance.

Note that r_{buffer} can be used to account for the size of mobile sensors, a stopping distance for sensors with higher-order dynamics, the maximum distance a sensor can traverse in between location updates, or other safe-related factors. Also, M_i exists if and only if neighbors are initially outside of s_i's localization uncertainty region.

Proposition 2 (CAR safety). Each sensor s_i may go anywhere within its CAR and be guaranteed to avoid collisions with all other sensors.

Proof: From (6) we know that sensor s_i must be inside of its localization uncertainty region, *i.e.,* $\text{dist}(q_i, \hat{q}_i) \leq r_i$. Therefore, Definition 2 guarantees that $q_i \in V_i$. Since V_i has disjoint interiors with its neighbors, it is guaranteed that s_i will not collide with any neighbors. □

Algorithm 2 Deadlock Avoidance

if $\hat{q}_i$ reaches a vertex of M_i **then**
  Move along either of the adjacent edges by r_{buffer}
else if $\hat{q}_i$ reaches an edge E_k of M_i **then**
  Compute distance to the right-hand vertex r_v
  if $r_v < r_{\text{buffer}}$ **then**
    Move along E_k to the right by r_v
  else
    Move along E_k to the right by r_{buffer}
  end if
end if

2) Deadlock Avoidance: Deadlock is the problem that sensor mutually block each other from reaching their goals. While using CUV-based method, this can occur when the goal is located in the the intersection of CUV cells. Zhou *et al.* [7] proved that a deadlock can only happen under the condition that a sensor is at a vertex or on an edge of its safe moving region, the buffered Voronoi cell in their paper or the CAR in our case. They proposed two heuristic solutions that perform well in practice to alleviate deadlock phenomena, the second of which we utilize in our implementation. This basic idea, outlined in Algorithm 2, is to continuously break this deadlock condition.

C. Distributed Coverage Control

As discussed in Section II-A, minimizing the cost functional $\mathcal{H}$ with respect to the sensor dominance regions $\mathcal{W}$ yields Voronoi cells V_i for $i = 1, \ldots, n$ when the sensor locations are known. However, in our setting this final condition is no longer true. Instead, we will utilize the CUV cells C_i as the dominance regions W_i. By construction, the UV cells U_i are the smallest dominance region that ensure that each location A is within at least one sensor dominance region. However, the UV cells are not convex so the weighted centroid may be outside of the cell boundaries. Thus, we choose to use the CUV cells as these are the smallest convex regions containing the UV cells. Additionally, it is more computationally efficient to work with convex polygons rather than regions defined by the intersection of conic sections.

To achieve distributed coverage control, the mobile sensors run Algorithm 3. Each sensor iteratively finds the weighted centroid in its CUV cell and attempts to reach it. If the centroid is outside of its CAR, the sensor goes to the point in its CAR that is closest to the centroid. If a sensor reaches the boundary of its CAR, then it runs Algorithm 2 to avoid deadlock.

IV. Simulations

We conduct simulations using MATLAB to validate our proposed control methods. The environment is an open 100×100 m square mission space with no obstacles. The information distribution function is initially the summation of 20 Gaussian probability density functions (PDFs), each of which has a random mean and a covariance matrix of the

Algorithm 3 Distributed Coverage Control

for each sensor s_i **do**
  Compute Voronoi cell V_i
  Compute CAR M_i using $V_i, r_i, r_{\text{buffer}}$
  Compute CUV cell C_i using Algorithm 1
  Find weighted centroid c of C_i
  Find goal $q_i^* = \arg\min_{x \in M_i} \|x - c\|$
  if q_i^* in the interior of M_i **then**
    Move towards c
  else
    Deal with deadlock using Algorithm 2
  end if
end for

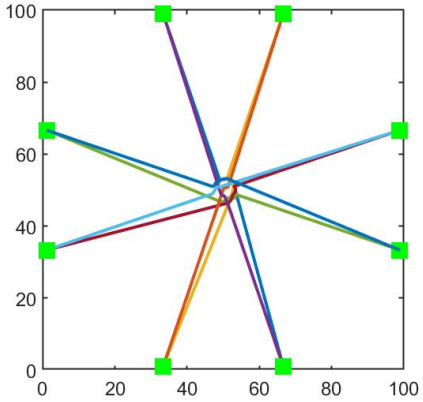


Fig. 4. Trajectories of each sensor in collision avoidance test. The green markers indicate the initial positions of each sensor. Each pair of antipodal sensors has a pair of lines with different colors showing the trajectories of each sensor.

form $\sigma_{\text{env}}^2 I$, where I is an identity matrix and $\sigma = 3\,\text{m}$. The mean of each Gaussian PDF performs a Gaussian random walk with maximum velocity 5 m/s, and the means may move out of the environment and re-enter. The covariance matrices are time invariant.

Sensors are regarded as particles, occupying no space. Sensor motion is holonomic with a maximum velocity of 5 m/s. Sensors localize themselves at the frequency of 10 Hz and the covariance matrix for the location of each sensor is of the form $\Sigma_i = \sigma_i^2 I$, where σ_i is time invariant, though it may be different for each i. The sensors begin each trial uniformly distributed along the edges of the space, ensuring that they begin a safe distance from each other. We assume that each sensor is able to obtain information everywhere in their own CUV cell. While this is a limiting assumption, the goal of this work is to demonstrate the efficacy of the control strategy. Practical concerns, such as sensors with limited field of view and imperfect measurements, will be addressed in future work. Also note the sensors use a sampling-based integration method to calculate the centroids.

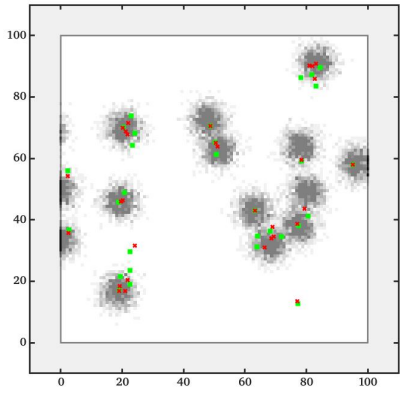


Fig. 5. Distribution of sensors and information after 100 simulated seconds. Green markers show the true positions of 30 sensors and red crosses show their current goals, *i.e.,* the weighted centroid of their CUV cells. The information distribution is shown in grayscale in the background, with darker indicating more information. The sensors were originaly uniformly spaced along the boundaries of the environment.

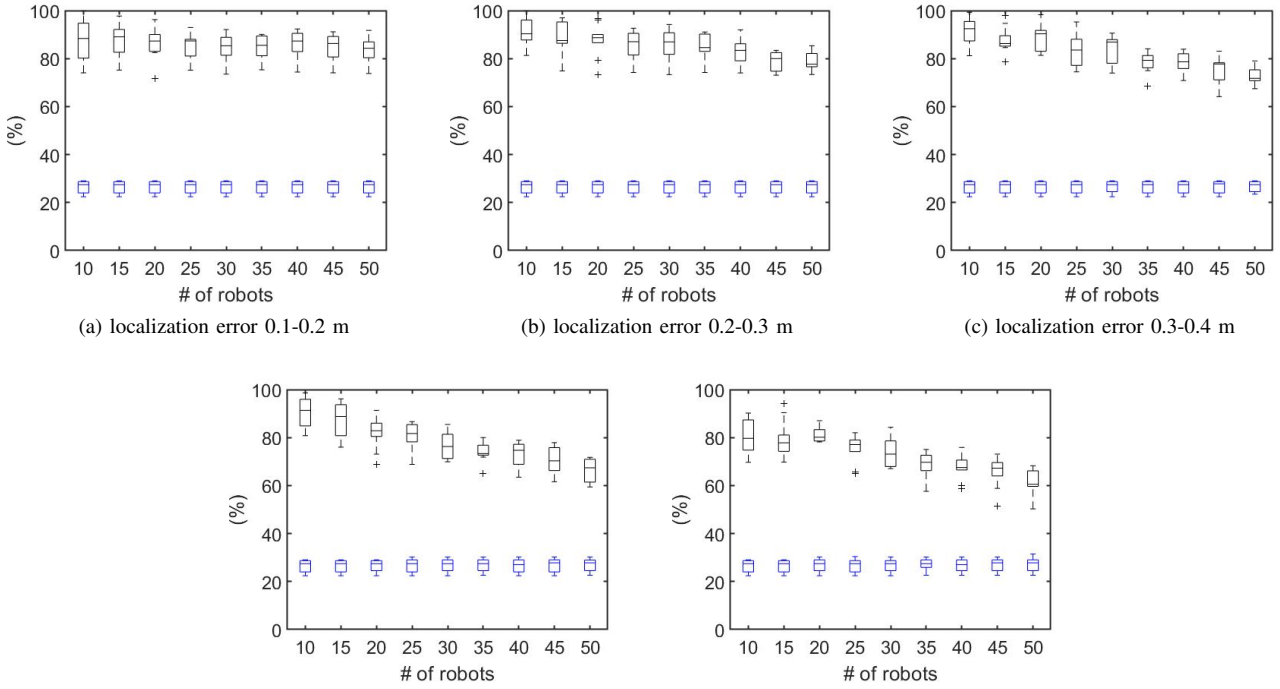
A. Collision Avoidance

Before testing the control algorithm, we first show how sensors avoid collision and deadlock. Eight sensors are evenly distributed at the edges of the mission space at the beginning, formulating four pairs of antipodal sensors, as Figure 4 shows. The goal is for the sensors in each pair to exchange positions. All sensors start moving to their goals simultaneously with the same velocity. Due to this symmetry, all sensors approach the center at the same time, blocking the way of the other sensors. As Figure 4 shows, all sensors were able to successfully avoid collision and eventually reach their goals.

B. Optimized Coverage

1) Single Trial: We first show a single trial using 30 sensors. Each sensor has a localization error σ_i randomly distributed in the range [0.2, 0.3] m (so $r_i \in [0.6, 0.9]\,\text{m}$). All sensors begin uniformly distributed along the boundaries of the environment, and the trial lasts for 100 s. The results are shown in Figure 5. We see that most sensors end up clustered in the areas of high information density, while a few of others stay in low information density area in order to maintain coverage of the entire mission space. Some sensors have not reached their temporary goals since the information density changes over time, resulting in the continuous change of the weighted centroids in their CUV cells.

2) Comparison of Trials: We then conduct a large array of experiments to show the performance of different sensing networks. We define dense-information regions as regions that are within $3\sigma_{\text{env}}$ of the means of the Gaussian PDFs in the information distribution function. To measure the performance of the team, we use two metrics. First, we measure the fraction of the total area that lies within the dense information regions, denoted as the dense-information proportion (DIP). Second, we measure the fraction of the



(a) localization error 0.1-0.2 m (b) localization error 0.2-0.3 m (c) localization error 0.3-0.4 m

(d) localization error 0.4-0.5 m (e) localization error 0.5-0.6 m

Fig. 6. Boxplots showing the OSP (black) and the DIP (blue) percentages for networks with 10-50 sensors and different localization errors σ_i. Each boxplot contains the results from 10 trials.

total number of sensors believed to be within high-density regions, *i.e.,* $\hat{q}_i$ in the high-density region, denoted as the optimized sensor proportion (OSP). The difference between the OSP and the DIP will demonstrate the ability of our control algorithm to guide sensors to areas of high information density. Specifically, we want the OSP to be significantly higher than the DIP, indicating that the sensors are gathering at locations with high information value.

We compare sensing networks of 9 different sizes from 10 sensors to 50 sensors. For each network size we test 5 different uncertainty region sizes, drawing σ from uniform distributions ranging from [0.1, 0.2] m to [0.5, 0.6] m. We run 10 trials for each configuration. We log the data for 300 s and use only the last 200 s to compute the OSP and the DIP for each sensing network since it takes up to 100 s for the OSP to reach steady state. The mean and range of the DIP are nearly identical for all tests, indicating that the total information density over the mission space is relatively stable for all tests. The results show that for all sensing networks, the OSP is at least two times larger than the DIP, meaning that all of the team has optimized the sensor locations.

For each network size, the OSP decreases as the range of σ increases. This is expected, since increasing σ also increases the minimum allowable distance between sensors using the CAR. The result is that fewer sensors are able to gather within high density areas. We also see that for this particular environment, the team size that maximizes OSP is around 20. This makes sense as there are 20 separate components in the information distribution function.

V. Conclusions

We propose a distributed control algorithm for a mobile sensing network that optimizes the sensor locations to improve detections while maintaining coverage of the entire mission space, accounting for uncertainty in the location of each sensor, and guaranteeing safety. This approach uses two novel variants of the Voronoi cell: the convex uncertainty Voronoi (CUV) diagram and the collision avoidance region (CAR). Sensors are able to construct both the CUV and the CAR in a distributed fashion, using only local information about sensors' estimated locations and the associated uncertainty of these estimates. The sensors then recursively drive to the weighted centroid of their CUV cells, using the information distribution function to determine the relative weights of each location in the environment. This enables sensors to move to regions with high information density. The CARs then restrict the motion of each sensor to avoid collision with others and to avoid becoming stuck in any deadlock configuration.

Our simulation results show that the proposed algorithm functions as desired. Furthermore, we explore the effects of changing the size of the network and the scale of the localization error on the performance of the team. We see that increasing localization error results in larger spacing between sensors. Future work will focus on performing hardware tests using teams of mobile robots and incorporating a mechanism to estimate the information distribution function online.

REFERENCES

[1] I. I. Hussein and D. M. Stipanovic, "Effective coverage control for mobile sensor networks with guaranteed collision avoidance," *IEEE Transactions on Control Systems Technology*, vol. 15, no. 4, pp. 642–657, 2007.

[2] W. Li and C. G. Cassandras, "Distributed cooperative coverage control of sensor networks," in *Proceedings of the 44th IEEE Conference on Decision and Control*. IEEE, 2005, pp. 2542–2547.

[3] M. Zhong and C. G. Cassandras, "Distributed coverage control and data collection with mobile sensor networks," *IEEE Transactions on Automatic Control*, vol. 56, no. 10, pp. 2445–2455, 2011.

[4] A. Okabe, B. Boots, K. Sugihara, and S. N. Chiu, *Spatial tessellations: concepts and applications of Voronoi diagrams*. John Wiley & Sons, 2009, vol. 501.

[5] J. Cortes, S. Martinez, T. Karatas, and F. Bullo, "Coverage control for mobile sensing networks," *IEEE Transactions on robotics and Automation*, vol. 20, no. 2, pp. 243–255, 2004.

[6] Q. Du, M. Emelianenko, and L. Ju, "Convergence of the Lloyd algorithm for computing centroidal Voronoi tessellations," *SIAM Journal on Numerical Analysis*, vol. 44, no. 1, pp. 102–119, 2006.

[7] D. Zhou, Z. Wang, S. Bandyopadhyay, and M. Schwager, "Fast, online collision avoidance for dynamic vehicles using buffered Voronoi cells," *IEEE Robotics and Automation Letters*, vol. 2, no. 2, pp. 1047–1054, 2017.

[8] M. Schwager, J. McLurkin, and D. Rus, "Distributed coverage control with sensory feedback for networked robots." in *robotics: science and systems*, 2006, pp. 49–56.

[9] M. Schwager, D. Rus, and J.-J. Slotine, "Decentralized, adaptive coverage control for networked robots," *The International Journal of Robotics Research*, vol. 28, no. 3, pp. 357–375, 2009.

[10] P. M. Dames, "Distributed multi-target search and tracking using the phd filter," *Autonomous Robots*, pp. 1–17.

[11] X. Xie, R. Cheng, M. L. Yiu, L. Sun, and J. Chen, "UV-diagram: a Voronoi diagram for uncertain spatial databases," *The VLDB JournalThe International Journal on Very Large Data Bases*, vol. 22, no. 3, pp. 319–344, 2013.

[12] M. Jooyandeh, A. Mohades, and M. Mirzakhah, "Uncertain Voronoi diagram," *Information processing letters*, vol. 109, no. 13, pp. 709–712, 2009.

[13] W. Evans and J. Sember, "Guaranteed Voronoi diagrams of uncertain sites," in *20th Canadian Conference on Computational Geometry*, 2008, pp. 207–210.

[14] H. Zhu and J. Alonso-Mora, "B-UAVC: buffered uncertainty-aware Voronoi cells for probabilistic multi-robot collision avoidance," in *The 2nd IEEE International Symposium on Multi-robot and Multi-agent Systems*, 2019.

[15] M. Wang and M. Schwager, "Distributed collision avoidance of multiple robots with probabilistic buffered Voronoi cell," in *The 2nd IEEE International Symposium on Multi-robot and Multi-agent Systems*, 2019.

[16] S. Thrun, W. Burgard, and D. Fox, *Probabilistic Robotics*. MIT press, 2005.