Multi-Robot Workspace Allocation with Hopfield Networks and Imprecise Localization

Mert Turanli, Hakan Temeltas

Control and Automation Engineering Department, Istanbul Technical University, ITU Ayazaga Campus, Istanbul, 34469, Maslak, Turkey
turanlim@itu.edu.tr, hakan.temeltas@itu.edu.tr

Abstract: With the increased use of robots with different performance characteristics in areas such as search and rescue, patrolling and surveillance, the control of multiple robots with unequal capabilities have gained a lot of interest among robotics researchers. Also, the uncertainty of the sensors utilized on the robots for localization has made the problem of imprecise localization attractive. This paper aims to present the development and implementation of a multi-agent collaboration algorithm under localization uncertainty using Hopfield neural networks, guaranteed power Voronoi diagrams (GPVD or GPD), and coverage control. The agents are considered non-holonomic wheeled mobile robots under the assumption that their locations are not known precisely, but they are known to be in uncertain circles. The workspace is partitioned with a Guaranteed Power Voronoi Diagram (GPVD or GPD) algorithm which takes imprecise localization into account. Also, it is assumed that the actuation capabilities of the robots are different from each other and the agents do not know those performances beforehand. The performance parameters of the robots are learned by using the collaboration algorithm with Hopfield Neural Network (HNN) and then passed to the GPD algorithm. The GPD algorithm together with the HNN provides workspace partitioning for the robots so that the agents with poor actuation performances take smaller regions from the workspace while the agents with strong performances take greater regions. Thus, a collaborative coverage task is achieved which enables the robots to deploy themselves to an optimal configuration minimizing the total coverage cost. The simulation results in MATLAB show the efficiency of the algorithm. The experimental results with the Robot Operating System (ROS) are given. The results obtained are satisfactory since the algorithm has faster convergence and has the capability to assign the regions from the workspace considering the imprecise localization resulting from sensor characteristics. Finally, the algorithm is compared to the base collaboration method, important performance improvements had been observed.

Keywords: Workspace allocation; Coverage control; Guaranteed Power Voronoi Diagrams; Hopfield Neural Network

1 Introduction

The Multi-agent collaboration and coordination problems have been well studied in robotics literature over the last decades. Autonomous agents deploy themselves over the area to maximize sensor coverage. The Voronoi-based coverage control method became quite important due to its application potential in practice, for example, surveillance and security uses, patrolling and environmental monitoring applications, dealing with the deployment and allocation problems at the same time.

The definition of the Voronoi-based coverage control problem is autonomous self-deployment of the agents in the environment so that the optimal coverage configuration can be achieved. The Voronoi-based coverage control accomplishes the dynamic workspace partitioning by making use of Voronoi diagrams. Another definition of the coverage is to observe every point of a given region or to see it with a sensor having a field of view [11].

To summarize the new ideas of this paper with respect to the literature, in the previous method, which is about the adaptation to the performance variations of multiple robots [22], a non-linear estimator was used to enable the robots to learn their own parameters. In this work, a faster and more robust estimator based on HNN is utilized instead. Also, the localization uncertainty coming from sensors such as laser scanners or odometry is taken into consideration by using Guaranteed Power Voronoi Diagrams. In the base work, the localization uncertainty is not taken into consideration where the authors did the workspace partitioning by using the Power Voronoi Diagrams.

The contributions of the work are further explained in the Contributions section.

1.1 Related Studies

Several self-deployment algorithms taking localization uncertainty [21], [14], [19], [28] into account exist in the literature using Guaranteed Voronoi Diagrams [6] and their variants. Also, a Power-Voronoi-based collaboration algorithm is studied by the authors [23]. Pierson, A. and the others investigated inter-robot trust adaptation in response to the relative performances by using multi-robot sensor coverage [24]. A multi-robot cooperative coverage algorithm is proposed in which the robots are spraying a large field by performing task allocation and coordination [10]. Additionally, Hopfield Neural Networks are well studied by the authors for estimating the system parameters online in robotics [1] and control [3], [9] applications.

In the work [20], a multi-agent dynamic coverage method with anisotropic sensing footprints is proposed. Safety and convergence guarantees are taken into consideration. The agents are forced to search an area of interest collaboratively with local and global coverage strategies. Avoidance strategies are also developed.

In a recent paper [27], a framework for distributed coverage control for mobile sensors is discussed. In the multi-agent problem, the locational optimization is considered a special case of optimizing Kullback-Leibler divergence where space is density function space. The distributed coverage control laws are then formulated by minimizing distance functions. For a possible metric for distance functions, L^2 distance is utilized. A mobile sensor network coverage problem is studied in another work [4]. The sensors have different velocity constraints and the effect of the measurement errors is investigated. In the study of the authors [16], the agents accomplish the persistent coverage task in a distributed way. The control law is cooperative, and the environment is structured. It is proven that no collision occurs between agents nor with obstacles.

Also, there are many other related papers especially in the robotics and engineering areas that can be briefly summarized. In the work [15], a genetic algorithm approach is given to a networked high-performance drilling process. The algorithm is used for optimal tuning of linear controllers and the mutation part takes care of doing a search among the different linear controllers. Simulation and experimental results are given. In another paper [12], two open research surgical robot platforms are discussed. The paper gives the aims and related problems in the robotic surgery field. It also briefly introduces the research platforms. In the paper [8], the virtual technologies in engineering practice are discussed. The complex engineering product and related activities are created on the system level computer. The research results in the field of virtual technology are given. The paradigm in the virtual engineering area shifts to the Virtual Engineering Space (VES) approach. A new concept Knowledge Content Background (KCB) is introduced. In another work [26], a method for generating the optimal path for a traveling robot in a partially unknown environment is introduced. The method can handle both static and dynamic obstacles and a comparison with the Particle Swarm Optimization method is presented. In the study [30], a device for parallel robot investigations is given. The size of the elements of the device can be changed so that different robots can be realized. By reconfiguring the device, different characteristics can be tested. The main aim of the device is the construction studies. In the paper [31], the control problems in the surgical robotics area are discussed. For the precision in control methods used, the interaction between the manipulated tissues and the tool should be modeled. In the paper, the design and modeling of telesurgical systems is presented and possible control methods are summarized.

1.2 Contribution

The contribution of the paper is that it uses a coverage collaboration algorithm that calculates the weights of the agents automatically according to their actuation performances [22] under localization uncertainty [21], [14], [6] with HNN estimator [1, 3]. To the best of our knowledge, this is the first work in the

literature that uses an HNN estimator and takes the localization uncertainty into account in a coverage collaboration algorithm at the same time. The net contribution of this work is that it provides faster convergence for the multi-agent systems using Voronoi partitioning-based collaboration approaches under localization uncertainty by using HNNs. This enables the algorithm to compensate different capabilities of the agents such as actuation by assigning the areas from the workspace to the agents according to their performances and to account the imprecise localization.

1.3 Paper Organization

The paper is organized as follows. The first section gives the introduction, related studies and the contribution of the work. In the second section, the formal problem statement, preliminary information about workspace partitioning with Voronoi Diagrams, HNNs and locational optimization will be given. In the next section, the coverage control algorithm with HNNs and adaptation to performance variations will be investigated. In the fourth section, the stability analysis of the control and estimation algorithm will be discussed. After, MATLAB simulation results are given with different case studies. The next section gives the experimental setup and results. In the last section, the conclusions of the work will be explained.

2 Problem Formulation

In this section, preliminary information about Guaranteed Power Voronoi diagrams, HNNs, and locational optimization are presented.

The formal statement of the problem is as follows. Consider a team of n non-holonomic wheeled mobile agents. The aim is to maximize coverage according to locational optimization function by performing collaboration among the agents. The collaboration is achieved by estimating the actuation performances of the agents with HNNs which are not known beforehand. Then, the regions of the agents are assigned automatically by giving the performance parameters to the GPD algorithm. In the resultant configuration, the agents with strong actuators take larger portions from the workspace, while smaller regions are assigned to the robots with weak actuators.

In the N-dimensional space $S \subseteq \mathbb{R}^N$ is defined as a convex region. The positions of the robots are not known precisely, but known to be within an uncertain circle.

2.1 Guaranteed Power Voronoi Diagrams

The Guaranteed Power Voronoi Diagrams (GPD or GPVD) [14] are the types of Voronoi diagrams in which the power distance is used and the generator points are known to be in uncertain regions.

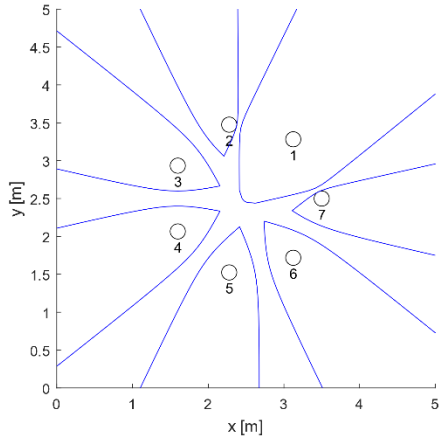

Figure 1
Example Guaranteed Power Diagram (GPVD) with the weight $w_1 = 0.5$

The distance function is defined similarly to the one in the Guaranteed Voronoi Diagrams [6]. The main difference is that the distance function is a power distance in a guaranteed sense. The bisector is a hyperbola when the weights are zero. The definition of the GPV-cell is given below where r_i $\mathrm{r_i}$is the radius of the uncertainty circle and w_i is the weight of the cell:

$$V_i^g = \left\{ p \in S \,|(\|p - p_i\| + r_i)^2 - w_i \leq \left(\|p - p_j\| - r_j\right)^2 - w_j, \right\} \quad (1)$$
$$i \neq j, j = 1,2,\dots n$$

Figure 1 gives an example of GPVD. The region of the first region is greater than the other ones since the weight of the first cell is given as 0.5.

2.2 Locational Optimization

Let us define $S \subseteq R^N$ as a bounded environment, $\phi: R^N \to R^+$ as a density function, and $\ell: R^+ \to R$ as a non-decreasing performance function. Then, the locational optimization function $\mathfrak{H}$ is given as below:

$$h(q, p_i, w_i) = \ell(\|q - p_i\|) - w_i \quad (2)$$

$$\mathfrak{H}(p_1, p_2, \dots p_m) = \sum_{i=1}^{m} \int_{\mathrm{V}_i} h(q, p_i, w_i)\, \phi(q) dq \quad (3)$$

Here, the V_i is defined as Voronoi region i, m gives the number of the site points, w_i represents the weight of the i^{th} cell and the site point of the corresponding Voronoi cell is given as p_i, as it can be viewed in Figure 2.

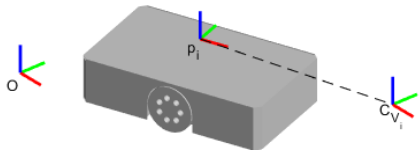

Figure 2
The position of the robot and the centroid location

The mass M_{V_i} and centroid C_{V_i} of a Voronoi region can be defined as given below [13]:

$$\Upsilon_i(h,q) = \int_{V_i} h\, dq \tag{4}$$

$$C_{V_i} = \frac{1}{M_{V_i}} \Upsilon_i(q\ \phi(q), q) \tag{5}$$

$$M_{V_i} = \Upsilon_i(\phi(q), q) \tag{6}$$

The performance function is taken as $\ell(\|q - p_i\|) = \|q - p_i\|^2$. So, the locational optimization function $\mathfrak{H}$ can be written as below:

$$\mathfrak{H}(p_1, p_2, \dots p_m) = \sum_{i=1}^{m} \int_{V_i} h(q, p_i, w_i)\, \phi(q)\, dq \tag{7}$$

The partial derivatives of the function $\mathfrak{H}$ are taken with respect to p_i to find the closed-form solution of the locational optimization function. Then, it can be shown that the locational optimization function given in (7) can be minimized by using the centroid positions given in (5). As a result, the solution is the result where the positions are equal to the centroid locations of the agents.

For the holonomic case $\dot{p}_i = u_i$, the coverage control law is given for i^{th} agent as:

$$\dot{p}_i = u_i \tag{8}$$

$$u_i = K_p(C_{V_i} - p_i) \tag{9}$$

Here, p_i gives the position of the i^{th} agent where K_p is a positive-definite gain matrix.

2.3 Hopfield Neural Networks

The Hopfield Neural Network as presented in [1], is a non-autonomous non-linear dynamical system in continuous time that is able to estimate the parameter vector according to the parameterization of a given dynamical system. Similar to other online parameter estimators, it is able to give the time-evolving estimate of the parameters of the actual dynamical system.

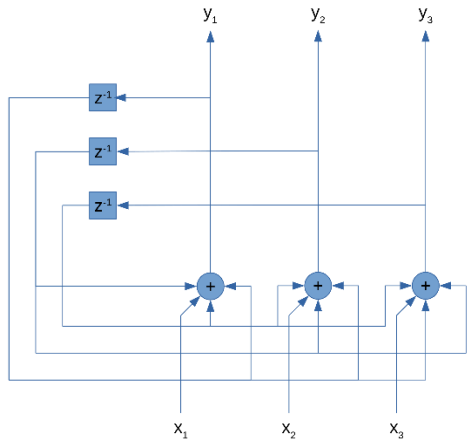

Figure 3
A Hopfield Neural Network with three neurons

Figure 3 gives an example of a Hopfield Network with three neurons. The network is in discrete-time and the previous values of the outputs y_i of the neurons are fed back to the inputs x_i of the neurons.

Since the parameterization of the system is given in the design stage of the HNN, there is no need to train the network. Only the convergence parameters should be adjusted in order to perform a fast and robust estimation. The weights of the network are automatically calculated according to the parametrization of the system. Also, the state vector in the network and the resulting differential equation enables the parameter vector to converge to the real parameter values.

Consider a Hopfield Neural Network of M neurons. Let x_i be the total input to neuron i, s_j be the state or output of the neuron j, W_{ij} be the weight associated with the connection from neuron j to neuron i and I_i be the bias of the neuron i. Then, the state equations of the HNN can be given as:

$$\frac{dx_i}{dt}(t) = -\left(\sum_{j=1}^{M} W_{ij}(t)s_j(t) + I_i(t)\right) \tag{10}$$

$$s_i(t) = \alpha \tanh\left(\frac{x_i(t)}{\beta}\right) \tag{11}$$

The total input-state relation is given as in (11) where $\alpha, \beta > 0$ and $i \in M$.

$$\frac{dx}{dt}(t) = -(W(t)s(t) + I(t)) \tag{12}$$

$$s(t) = \alpha \tanh\left(\frac{x(t)}{\beta}\right) \tag{13}$$

The matrix representation of the HNN is defined in (12) and (13).

In this work, in order to find the performance parameters of the agents, the Hopfield Neural Network is used to perform online parameter estimation.

The choice of the utilization of the HNN as a parameter estimator is made since other approaches like classification with neural networks have accuracy limitations.

3 Coverage Control with Agents with Unequal Actuation Capabilities

In this section, the point-offset control law for non-holonomic robots is given. Then, the parameter estimation with HNN is explained. After, the adaptation to actuation performance variations algorithm is introduced.

The actuation performances are defined as different capabilities of the agents. For example, weak motors and wheel slip can be counted as weak actuation performances, besides powerful motors and favorable terrain are the examples of the strong actuation capabilities. The collaboration algorithm learns the performance variations of the agents by estimating model parameters without prior knowledge and compensates them by giving large regions to the powerful agents and smaller regions to the weak ones.

3.1 The Control Law

The control of the agents is a crucial issue in the coverage collaboration task. After the agent calculates its own centroid location, a control law driving the agent to the position should be executed. In other words, a control law should be selected and designed in order to perform the desired coverage task successfully.

In the literature, there are many works about non-linear controllers for robotics applications. To summarize the recent ones, a stochastic nonlinear model predictive control (MPC) [7], a nonlinear MPC trajectory controller [18], a feedback linearization controller for trajectory tracking for a flexible robot [2], a nonlinear high accuracy feedback linearization control of flexible robots [5] are the related control papers in the literature that are suggested to the reader.

The selected structure of the control law consists of the non-holonomic point-offset control law and the coverage controller given in equation (9). The point-offset controller for the non-holonomic agents consists of a reference point *P* and a distance *l* from the center of the robot [17, 25] to the point *P*. The velocity of the reference point *P* can be transformed to the linear (v) and angular (ω) velocities by using the matrix equation given below where θ is the heading angle of the robot:

$$\begin{pmatrix} v \\ \omega \end{pmatrix} = \begin{pmatrix} cos\theta & sin\theta \\ -\frac{sin\theta}{l} & \frac{cos\theta}{l} \end{pmatrix} \begin{pmatrix} \dot{x} \\ \dot{y} \end{pmatrix} \tag{14}$$

After substituting the velocities from the coverage control law in (9) with the (14), the control law becomes:

$$u_i = K_p \begin{pmatrix} cos\theta_i & sin\theta_i \\ -\frac{sin\theta_i}{l} & \frac{cos\theta_i}{l} \end{pmatrix} (C_{\mathrm{V}_i} - p_i) \tag{15}$$

The K_p is selected so that the control law gives a fast and stable response. Also, the controller is designed in continuous time, but for the discrete-time, a high sampling rate is selected [29] according to the kinematics having in mind that the trajectories calculated by the robots and the dynamics of the robots are slow. The design process of the controller is iterative. First, a high sampling rate is chosen empirically and then the controller gains are adjusted under the physical limits of the robots. The chosen gains are validated by performing a simulation. The process is repeated iteratively until the resulting stable gains are found. The simulation results in the related sections show that the sampling rate is sufficient for the kinematics of the agents under the physical constraints of the robots.

In order for the robot to avoid the collisions, the center *P* should be checked against collisions with radius $\rho = l + r_{robot}$, where r_{robot} is the radius of the robot.

3.2 Parameter Estimation with HNNs

In order to estimate parameters of the system given in (8), the system should be rewritten in linear in parameters (LIP) form:

$$\dot{p}_i = K_p(C_{\mathrm{V}_i} - p_i) \tag{16}$$

$$\dot{p}_i = K_p C_{\mathrm{V}_i} - K_p p_i \tag{17}$$

$$p_i = -{K_p}^{-1}\dot{p}_i + C_{\mathrm{V}_i} \tag{18}$$

$$y = p_i - C_{\mathrm{V}_i} = -{K_p}^{-1}\dot{p}_i \tag{19}$$

$$y = -\begin{pmatrix} 1/K_{p,1} & 0 \\ 0 & 1/K_{p,2} \end{pmatrix} \dot{p}_i \tag{20}$$

$$y = \begin{pmatrix} -\dot{p}_{i,x} & 0 \\ 0 & -\dot{p}_{i,y} \end{pmatrix} \begin{pmatrix} 1/K_{p,1} \\ 1/K_{p,2} \end{pmatrix} \tag{21}$$

Then, the parameter estimation vector is defined as follows:

$$\theta_{est} = \begin{pmatrix} 1/K_{p,1} \\ 1/K_{p,2} \end{pmatrix} \tag{22}$$

The LIP form of the system becomes:

$$A = \begin{pmatrix} -\dot{p}_{i,x} & 0 \\ 0 & -\dot{p}_{i,y} \end{pmatrix} \tag{23}$$

$$y = p_i - C_{V_i} \tag{24}$$

$$y = A\,\theta_{est} \tag{25}$$

The weight matrix W and bias I in equation (12) can be calculated as in the following equations:

$$W = A^T A \tag{26}$$

$$I = -A^T y \tag{27}$$

$$\hat{\theta}_{est} = \begin{pmatrix} s_1(t) \\ s_2(t) \end{pmatrix} \tag{28}$$

$$\widehat{K}_i = \begin{pmatrix} 1/\hat{\theta}_{est,1} & \\ & 1/\hat{\theta}_{est,2} \end{pmatrix} \tag{29}$$

By using (12), (13) together with (23), (24), (26) and (27) the parameters of the closed-loop system (16) can be estimated online. The estimated parameters can be calculated by using the equations (28) and (29).

The equation (12) can be implemented by using numerical integrators.

As explained in Section 2.3, there is no need to train the network because the parametrization of the system dynamics is already given.

For the testing results of the HNN, the reader is referred to the given source code and dataset in the following repository: https://github.com/mertturanli/hnn/

3.3 Estimating the GPVD Weights

The adaptation algorithm in this section is based on the algorithm in work [22]. Assuming that $\widehat{K}_i$ is obtained by using (28) and (29) after the estimation, the values of the parameter vector $\widehat{K}_i$ is transferred to the weight estimator. The output of the weight estimator is passed to the GPD workspace partitioning algorithm.

For the weight w_i of the agent i, the adaptation law is:

$$\delta_i = w_i - f(\widehat{K}_i) \tag{30}$$

$$\dot{w}_i = -k_\omega \sum_{j \in N_i} (\delta_i - \delta_j) \tag{31}$$

where k_ω is a positive coefficient and N_i represents the neighbors of the agent i. The function $f(\widehat{K}_i)$ is related to the desired performance and chosen as $f(\widehat{K}_i) = \|\widehat{K}_i\|$.

Assuming that $\widehat{K}_i$ is obtained for the agent i, the weights are calculated by using the estimation law in (30) and (31). For the implementation, a numerical integrator should be used in order to calculate the estimated weight value.

4 Stability Analysis

To prove the stability of the control and estimation laws, first, a Lyapunov function candidate should be defined as given below for V_3: $\theta_{est} + (-c, c)^M \rightarrow \mathbb{R}$:

$$V = V_1 + V_2 + V_3 \tag{32}$$

$$V_1 = \sum_i \frac{1}{2} \left\| C_{\mathrm{V}_i} - p_i \right\|^2 \tag{33}$$

$$V_2 = \sum_i w_i \tag{34}$$

$$V_3 = \sum_i -\frac{1}{2c} \sum_{j=1}^{M} ln\left(\left(1 + \frac{\tilde{\theta}_{est(i,j)}}{c - \theta_{est(i,j)}} \right)^{c - \theta_{est(i,j)}} \left(1 - \frac{\tilde{\theta}_{est(i,j)}}{c + \theta_{est(i,j)}} \right)^{c + \theta_{est(i,j)}} \right) \tag{35}$$

where $\tilde{\theta}_{est(i,j)} = \theta_{est(i,j)} - \hat{\theta}_{est(i,j)}$ and $\hat{\theta}_{est(i,j)}$ is defined as the output of the j^{th} neuron of the parameter estimator of the i^{th} agent. Also, c is a positive constant [1].

First, if we consider the state vector as $x_i = \begin{pmatrix} C_{\mathrm{V}_i} - p_i & w_i & \tilde{\theta}_{est(i)} \end{pmatrix}^T$, it can be seen that $V(x = 0) = 0$. Also, $V_3(x) > 0$ as given in proof in [1]. So, $V(x) > 0$.

Taking the derivative of the Lyapunov function yields:

$$\dot{V}_1 = \sum_i -\left(C_{\mathrm{V}_i} - p_i \right)^T \dot{p}_i \tag{36}$$

$$\dot{V}_1 = \sum_i -\left(C_{\mathrm{V}_i} - p_i \right)^T K_p \left(C_{\mathrm{V}_i} - p_i \right) \leq 0 \tag{37}$$

The first term is negative semi-definite.

$$\dot{V}_2 = \sum_i -k_\omega \sum_{j \in N_i} \left(\left(w_i - f\left(\widehat{K}_i \right) \right) - \left(w_j - f\left(\widehat{K}_j \right) \right) \right) = 0 \tag{38}$$

Also, the second term is zero. Taking the derivative of the third term becomes:

$$\dot{V}_3 = \sum_i -\frac{1}{c\,\beta} \tilde{\theta}_{est(i)}^T W_i \, \tilde{\theta}_{est(i)} \leq 0 \tag{39}$$

where W_i is the weight matrix for i^{th} agent and clearly positive semi-definite. For $\beta > 0, c > 0$ the third term is negative semi-definite.

The trajectories and estimation errors are bounded since the $\dot{V} \leq 0$. The weights are bounded since the $\sum_i \dot{w}_i$ becomes a stable filter as given in the proof in [22]. To complete the proof, the following theorem is introduced from [1]:

Theorem 1: The equilibrium point $\tilde{\theta}^*_{est(i)} = 0$ is globally asymptotically stable if $I \subset [t_0, \infty)$ and $\bigcap_{t\in I} \ker\big(A(t)\big) = \{0\}$.

From the Theorem 1 and the proof in [1], it can be concluded that for a non-degenerate interval $t \in I$, the equilibrium point $\tilde{\theta}^*_{est(i)} = 0$ is globally uniformly asymptotically stable and the equilibrium point is unique.

From LaSalle's Invariance Principle, the largest invariant set defined by $\dot{V} = 0$ should be found. $\dot{V} = 0$ occurs only when $C_{V_i} = p_i$ and $\tilde{\theta}_{est(i)} = 0$. These equilibrium points are unique, and the case corresponds to the case that the tracking and estimation errors become zero. From the control and estimation laws, it can be concluded that the set is an invariant set. Thus, the system is globally asymptotically stable.

Corollary 1: In the steady-state, the estimation vector converges to its real value and $w_i - f(K_i)$ converges to a common value for all agents [22]:

$$\lim_{t\to\infty}\big(\widehat{K}_i\big) = K_i \tag{40}$$

$$\lim_{t\to\infty}\big(w_i - w_j\big) = f(K_i) - f\big(K_j\big) \tag{41}$$

From the stable filter theory, in the steady-state, the weights converge to their final values, as stated in the referenced paper [22]. Also, from (41) the value of $w_i - f(K_i)$ converges to a common value for all agents.

The stability analysis is done in continuous time since the nonlinear systems are continuous in nature. However, the implementation is carried out in discrete-time and for this purpose, a high sampling rate is selected [29] according to the kinematics of the agents since the trajectories of the robots and the dynamics of the agents are slow and the dynamics are neglected in design. Also, the numerical integrations in the estimators are converted into discrete-time trapezoidal integrations in order to get accurate results in discrete-time.

5 Simulation Results

The simulations are done in MATLAB environment with the map sizes of 5x5 and 10x10 meters. The parameters in the simulation are $k_\omega = 0.2$, $K_p = diag([1 \quad 1])$, $r_{robot} = 0.11$, $l = 0.1$, $\alpha = 3000$ and $\beta = 1$. Also, the uncertainty radius is given as $r_i = 0.1$ meters.

The simulation is performed with 15 agents. In Figure 4 and Figure 5, the position errors and the coverage cost are given, respectively. The position errors of the robots asymptotically converge to zero and the coverage cost is settled to a minimum value after the coverage task is completed.

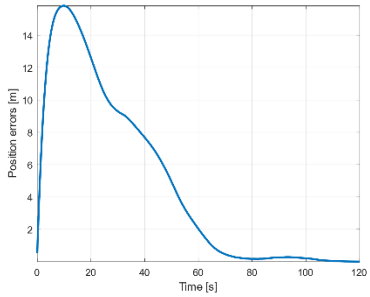

Figure 4
Position errors

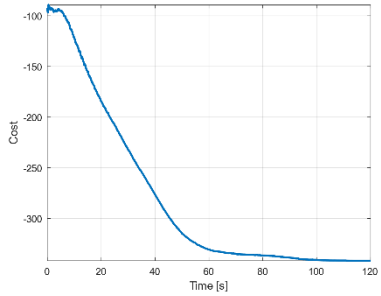

Figure 5
Coverage cost

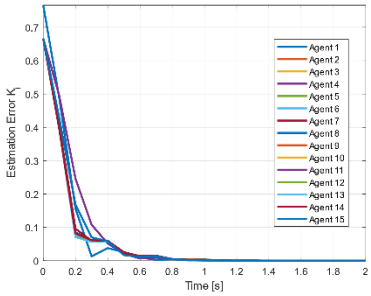

Figure 6
Parameter estimation errors

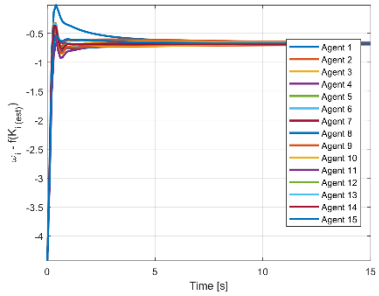

Figure 7
The value of $w_i - f(\widehat{K}_i)$

Figure 6 shows the asymptotic convergence of the parameter estimation errors of the Hopfield Network, obeying Theorem 1.

In Figure 7, the value of $w_i - f(\widehat{K}_i)$ is given which converges to a common point among the agents as given in the Corollary 1.

Figure 8 and Figure 9 represent the weight of agents and the trajectories, respectively. The weight of the first agent has a lower value than the other agents since it has a degraded performance different from the other ones. The trajectories of the agents converge to the optimal coverage positions, as seen in Figure 9.

Table 1 shows the region ratios of the agents at the end of the simulation. The region of the first agent is smaller than the regions of the other agents.

The results show that the HNN estimator outperforms well compared to the base method [22] which is a non-linear adaptive estimator. The HNN provides faster convergence and more robust estimation according to the simulation results, as it can be seen from the parameter estimation errors. As a result, the coverage time can be improved.

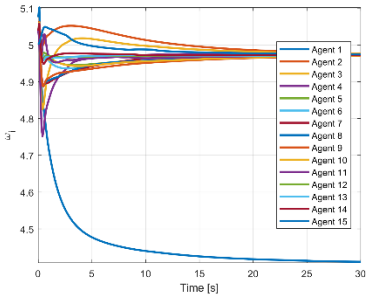

Figure 8
The weights of the agents

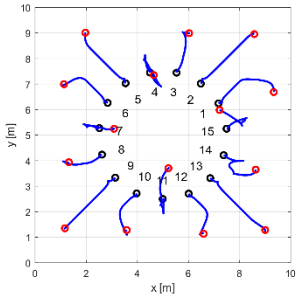

Figure 9
The trajectories of the agents

Table 1
Region ratios of the agents at the end configurations

Agent	Region Ratio	Agent	Region Ratio
1	0.0349	9	0.0592
2	0.0560	10	0.0661
3	0.0537	11	0.0688
4	0.0602	12	0.0614
5	0.0669	13	0.0496
6	0.0572	14	0.0652
7	0.0527	15	0.0629
8	0.0615	-	-

6 Experiments

The experiments were carried out with two Turtlebot 2 agents in 2x2 meter environment. The first agent had a 10% actuation performance degradation. The agents start at initial configurations and perform the collaborative coverage task.

6.1 Experimental Setup

The experiments were carried out with two Turtlebot 2 agents in ITU Robotics Laboratory. The computers on the robots are Acer Aspire E11 with Intel Celeron N2940 processors and 4 GB of memory running Ubuntu 14 and ROS Indigo. The Turtlebot 2 agents are differential drive robots having a maximum velocity of 0.65 m/s and a maximum payload of 5 kg. The weight of the robot is 6.3 kg.

Figure 10
The initial configurations of the robots (ITU Robotics Laboratory)

Figure 10 illustrates the experimental setup in the ITU Robotics Laboratory. The localization information is taken from the wheel odometry of the robots. The ROS driver "kobuki_node" is used for the Turtlebots. The communication between the agents is performed by using publisher/subscriber architecture and ROS topics. The robots are connected to a 450 Mbps Wi-Fi N access point.

The ROS master computer is a laptop PC running Ubuntu and ROS Kinetic. The computers on the robots are connected to the ROS master node over the Wi-Fi network.

6.2 Experimental Results

The experiment starts with a ROS node named as Coverage node running for each agent separately in a decentralized way. Each node estimates its own $\widehat{K}_i$ vector from its own motion by using HNNs by using (12) and (13) and then calculates the weight of the agent by using (30) and (31). The node then passes the weights to the GPD algorithm. After the centroid locations are found by using the GPV-cells and the non-holonomic control law is executed. The obtained velocities are sent to the corresponding topic of the agent.

The coverage node is running on the controller PC of the agent on ROS real-time in a distributed way. The robots communicate with each other and estimate their own performance parameters, estimate the weights and calculate the GPD regions. Then, they execute the control law after finding the centroid positions.

The experiment is repeated three times and the results are given in the table of region ratios.

Figure 11 shows the overall coverage cost of the agents for the first experiment. The cost converges to its local minima as the coverage task completes. Also, in Figure 12, the estimation errors show asymptotical convergence to zero.

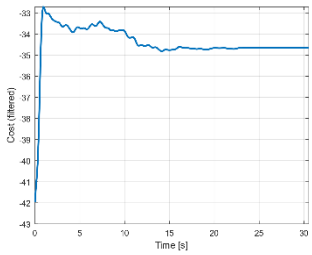

Figure 11
Coverage cost

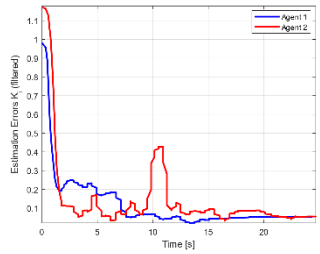

Figure 12
Estimation errors

Figure 13 illustrates the value of $w_i - f(\widehat{K}_i)$. As given in Corollary 1, the value converges to the same value among the agents. Lastly, Figure 14 depicts the weight values calculated by the online estimator. As expected, the weight of the first agent is less than the other agent.

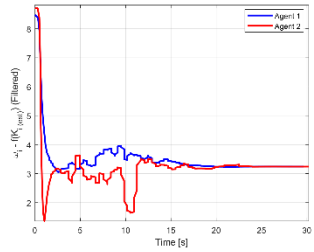

Figure 13
The value of $w_i - f(\widehat{K}_i)$

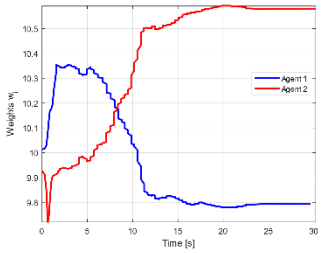

Figure 14
The weights of the agents

In Figure 15, the trajectories of the two agents are given. The black circles denote the initial configurations while the red circles show the final configurations.

At the end configuration, the obtained regions are given in Figure 16 for the first experiment. The region ratio of the first agent is less than the second one. In the first experiment, the α parameter is taken as $\alpha = 6000$.

At the last configuration, the obtained region ratios are given in Table 2 for the second, third, and fourth experiments. Here, in the three experiments, the Hopfield parameter α is taken as $\alpha = 3000$. The region ratio of the first agent is less than the other agent in each experiment. The results taken from the three experiments are similar. The distributed algorithm assigns the weights to the agents according to their performances. The video of the experiment can be viewed at: https://web.itu.edu.tr/turanlim/video/exp-tb2.mp4.

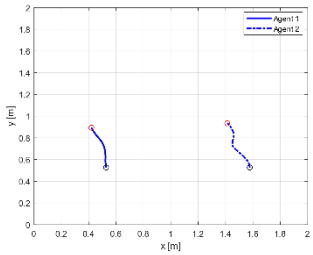

Figure 15
The trajectories of the agents

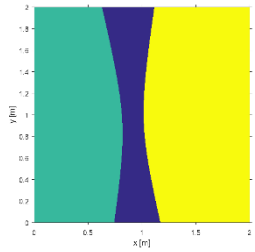

Figure 16
The GPD plot obtained from the first experiment

Table 2
Region ratios of the agents at the end configuration with HNN of three experiments

Agent	Region Ratio Experiment 1	Region Ratio Experiment 2	Region Ratio Experiment 3
1	0.1950	0.1917	0.2120
2	0.6484	0.6534	0.6342

Similar results were obtained in experiments compared to the MATLAB simulations. The agents with greater actuation performances take greater regions than the weaker ones. Also, the estimation errors show that the HNN estimator outperforms well compared to the base method [22] which is a non-linear adaptive estimator.

Conclusion

In the paper, a coverage collaboration algorithm for non-holonomic wheeled mobile agents which learns the actuation performances of the agents by using HNNs and allocates the areas of the workspace to the agents according to their performances under localization uncertainty is introduced. The robots do not know their performances beforehand. By estimating their own performances, they perform the collaborative coverage task by minimizing the locational optimization function. Meanwhile, the GPD algorithm is used to take the positioning uncertainty of the agents into account. Also, the control law drives the robots to their optimal configurations. So, the optimal coverage is accomplished in a decentralized manner. The simulation results in MATLAB show the efficiency of the algorithm. The results are verified with field experiments done with the ROS. The algorithm provides faster convergence and a more robust estimation performance compared to the base method in the literature. Also, it takes the localization uncertainty into account coming from the localization sensors.

References

[1] Alonso, H., Mendonça, T. and Rocha, P. Hopfield neural networks for on-line parameter estimation. *Neural Networks*, 22(4):450-462, 2009

[2] Ansarieshlaghi, F. and Eberhard, P. Trajectory Tracking Control of a Very Flexible Robot Using a Feedback Linearization Controller and a Nonlinear Observer. In *ROMANSY 22--Robot Design, Dynamics and Control*, Springer, pp. 26-33, 2019

[3] Atencia, M. and Joya, G. Hopfield networks: from optimization to adaptive control. In *Neural Networks (IJCNN), 2015 International Joint Conference on*, pp. 1-8, 2015

[4] Dou, L., Song, C., Wang, X., Liu, L. and Feng, G. Coverage Control for Heterogeneous Mobile Sensor Networks Subject to Measurement Errors. *IEEE Trans. Automat. Contr.*, 63(10):3479-3486, 2018

[5] Eberhard, P. and Ansarieshlaghi, F. Nonlinear Position Control of a Very Flexible Parallel Robot Manipulator. In *IFToMM World Congress on Mechanism and Machine Science*, pp. 155-162, 2019

[6] Evans, W. and Sember, J. Guaranteed voronoi diagrams of uncertain sites. In *20th Canadian Conference on Computational Geometry*, pp. 207210, 2008

[7] Gonçalves, L. M. G., Basso, G., Dórea, C. E. T. and Nascimento, T. P. Stochastic Nonlinear Model Predictive Mobile Robot Motion Control. In *2018 Latin American Robotic Symposium, 2018 Brazilian Symposium on Robotics (SBR) and 2018 Workshop on Robotics in Education (WRE)*, pp. 19-25, 2018

[8] Horváth, L. and Rudas, I. J. Information content driven model for virtual engineering space. *Acta Polytechnica Hungarica*, 15(2):7-32, 2018

[9] Hu, Z. and Balakrishnan, S. N. Parameter estimation in nonlinear systems using Hopfield neural networks. *J. Aircr.*, 42(1):41-53, 2005

[10] Janani, A., Alboul, L. and Penders, J. Multi robot cooperative area coverage, case study: spraying. In *Conference Towards Autonomous Robotic Systems*, pp. 165-176, 2016

[11] Leitner, J. and others. Multi-robot formations for area coverage in space applications. 2009

[12] Li, Y., Hannaford, B. and Rosen, J. The Raven Open Surgical Robotic Platforms: A Review and Prospect. *Acta Polytechnica Hungarica*, 16(8), 2019

[13] Luna, J. M., Fierro, R., Abdallah, C. T. and Wood, J. An Adaptive Coverage Control for Deployment of Nonholonomic Mobile Sensor Networks over Time-Varying Sensory Functions. *Asian J. Control*, 15(4):988-1000, 2013

[14] Mahboubi, H., Vaezi, M. and Labeau, F. Mobile sensors deployment subject to measurement error. In *Vehicular Technology Conference (VTC Fall), 2014 IEEE 80th*, pp. 1-6, 2014

[15] MartÃn, D., Caballero, B. and Haber, R. Optimal tuning of a networked linear controller using a multi-objective genetic algorithm. Application to a complex electromechanical process. In *2008 3rd International Conference on Innovative Computing Information and Control*, p. 91, 2008

[16] Mellone, A., Franzini, G., Pollini, L. and Innocenti, M. Persistent Coverage Control for Teams of Heterogeneous Agents. In *2018 IEEE Conference on Decision and Control (CDC)*, pp. 2114-2119, 2018

[17] Michael, N. and Kumar, V. Planning and control of ensembles of robots with non-holonomic constraints. *Int. J. Rob. Res.*, 28(8):962-975, 2009

[18] Nascimento, T. P., Dórea, C. E. T. and Gonçalves, L. M. G. Nonlinear model predictive control for trajectory tracking of nonholonomic mobile robots: A modified approach. *Int. J. Adv. Robot. Syst.*, 15(1), 2018

[19] Nowzari, C. and Cortés, J. Self-triggered coordination of robotic networks for optimal deployment. *Automatica*, 48(6):1077-1087, 2012

[20] Panagou, D., Stipanović, D. M. and Voulgaris, P. G. Distributed dynamic coverage and avoidance control under anisotropic sensing. *IEEE Trans. Control Netw. Syst.*, 4(4):850-862, 2017

[21] Papatheodorou, S., Stergiopoulos, Y. and Tzes, A. Distributed area coverage control with imprecise robot localization. In *Control and Automation (MED), 2016 24th Mediterranean Conference on*, pp. 214-219, 2016

[22] Pierson, A., Figueiredo, L. C., Pimenta, L. C. A. and Schwager, M. Adapting to performance variations in multi-robot coverage. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pp. 415-420, 2015

[23] Pierson, A., Figueiredo, L. C., Pimenta, L. C. A. and Schwager, M. Adapting to sensing and actuation variations in multi-robot coverage. *Int. J. Rob. Res.*, 36(3):337-354, 2017

[24] Pierson, A. and Schwager, M. Adaptive inter-robot trust for robust multi-robot sensor coverage. In *Robotics Research*, Springer, pp. 167-183, 2016

[25] Pierson, A. and Schwager, M. Bio-inspired non-cooperative multi-robot herding. In *Robotics and Automation (ICRA), 2015 IEEE International Conference on*, pp. 1843-1849, 2015

[26] Purcaru, C., Precup, R. E., Iercan, D., Fedorovici, L. O., David, R. C. and Dragan, F. Optimal robot path planning using gravitational search algorithm. *Int. J. Artif. Intell.*, 10(13 S):1-20, 2013

[27] Razak, R. A., Sukumar, S. and Chung, H. Distributed Coverage Control of Mobile Sensors: Generalized Approach using Distance Functions. In *2018 IEEE Conference on Decision and Control (CDC)*, pp. 3323-3328, 2018

[28] Sharifi, F., Zhang, Y. and Aghdam, A. G. A distributed deployment strategy for multi-agent systems subject to health degradation and communication delays. *J. Intell. Robot. Syst.*, 73(1-4):623-633, 2014

[29] Slotine, J.-J.E., Li, W. and others. Applied nonlinear control. Prentice hall Englewood Cliffs, NJ, 1991

[30] Somló, J., Varga, G. D., Zenkl, M. and Mikó, B. The 'Phantom' Delta Robot A New Device for Parallel Robot Investigations. *Acta Polytechnica Hungarica*, 15(4), 2018

[31] Takács, Á., Kovács, L., Rudas, I., Precup, R.-E. and Haidegger, T. Models for force control in telesurgical robot systems. *Acta Polytechnica Hungarica*, 12(8):95-114, 2015