Placement of a Team of Surveillance Vehicles Subject to Navigation Failures

Jean-Samuel Marier* Camille-Alain Rabbath† Nicolas Lechevin†

We consider the problem of performing surveillance of a region with vehicles subject to failures that prevent them from navigating through the environment. This work is based on a Voronoi coverage control problem formulation. In this formulation, failed vehicles can cause other vehicles to be trapped in a configuration that provides poor coverage of the region. To avoid this, each vehicle performs self-monitoring to detect failures. Upon detection of a failure, a negotiation mechanism is triggered and vehicles compare variations of a performance measure to decide whether when it is beneficial to help another vehicle. The proposed approach is evaluated experimentally using a fleet of three wheeled mobile robots.

I. Introduction

Unmanned vehicles can be used to perform a variety of tasks such as surveillance. Such tasks are often modeled as the problem of covering the area with the vehicle's sensors. To do this, the space is decomposed into one cell per vehicle. Each vehicle moves to a position that provides the best surveillance for its cell. This approach is often called Voronoi coverage control, or locational optimization. While this approach does not guarantees the vehicles position will be optimal, it can be optimized by knowing only information about nearby agents. This ensures scalability as the number of deployed vehicles grow and is also desirable because the algorithm has no single point of failure.

However, such a local optimization scheme does not perform very well when vehicles can suffer from failures that prevent them from moving to their desired position. In such cases, a single vehicle unable to navigate can prevent many others from reaching their destination, resulting in poor coverage. In this paper, we propose a way to address this problem.

There are many possible causes that can prevent a vehicle from navigating properly. Some examples include the failure of some onboard components as well as being physically trapped by obstacles. We make the assumption that the faulty vehicle is still able to perform surveillance, even though it cannot move. This is typical of actuator failures, as defined by the failure taxonomy of Carlson *et al.*[1] Situations where the vehicle can fail to navigate but still perform surveillance are particularly relevant for ground vehicles. The case where the vehicle is no longer able to perform surveillance is straightforward to handle, as it is sufficient for its peers to ignore it.

Figure 1 shows a simple example of the type of situations we are concerned about. The vehicle near $(0.5, 0.4)$ cannot move. Figure 1(b) shows a configuration which provides better coverage of the region. This configuration was reached by moving the vehicles initially near $(0.2, 0.2)$ and $(0.8, 0.2)$.

The proposed solution is made of three components:

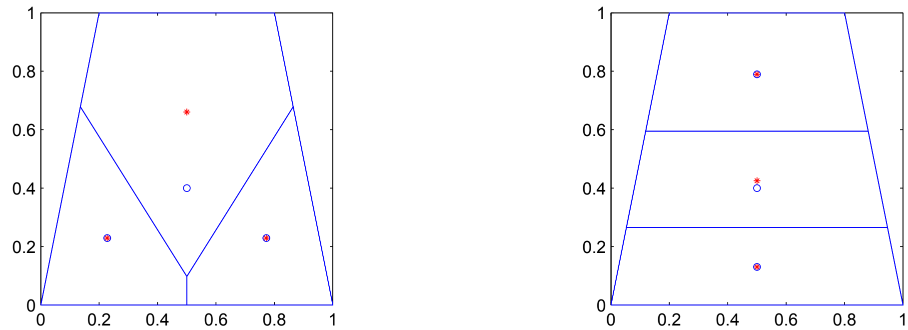
A fault detection component:

A faulty agent must be able to detect its inability to reach the desired location.

A mechanism to perturb the position of the vehicles:

Upon detection of a failure, the faulty agent initiates a negotiation with its peers. The intent of this negotiation is that another agent might accept occupying the location that could not be reached by the faulty agent. This allows escaping a poor local minimum to reach a better configuration, despite at least one vehicle being faulty.

*Numérica Technologies, Québec, QC, Canada.
†Defence Research and Development Canada Valcartier, Québec, QC, Canada.



(a) A vehicle in the center is blocking the other two, preventing them from deploying.

(b) An almost locally optimal configuration where the vehicle in the center has not moved.

Figure 1. Example of a vehicle robot blocking others. Circles represent vehicles locations. Stars represent the centroid of each cell.

A test for improvement:
The negotiation algorithm requires a test to decide whether occupying the offered position is better than another position.

These three components described above are designed to preserve, as much as possible, the dependence on local information only. The resulting algorithm is still decentralized and has no single point of failure.

The remainder of this article is structured as follows. Section II reviews the framework used for optimizing the location of sensors as well as the obstacle avoidance scheme used for robot navigation. The criterion for deciding whether a vehicle should move to some location outside its assigned region is discussed in Section III. The protocol to negotiate this move is presented in Section IV and a self-test for vehicle detection is proposed in Section V. Finally, an experimental evaluation on wheeled mobile robots (WMRs) is carried out in Section VI.

II. Background Information

Locational optimization is a framework to model area coverage tasks.[2] The problem is to find a nearly optimal placement of some resources, such as sensors, within a given area. The objective is stated in terms of the minimization of a cost function which penalizes sensor placement based on the distance between any point of the region of interest and the sensor closest to that point.

Consider a set of vehicles, indexed by elements of $\mathcal{I} = \{1, \ldots, n\}$, a set of cardinality $|\mathcal{I}| = n$. Assume they are located in $\mathcal{Q} \subset \mathbb{R}^2$, a convex subset of the Cartesian plane. The position of the ith vehicle is $\mathbf{p}_i \in \mathcal{Q}$. Let $P = (\mathbf{p}_1, \ldots, \mathbf{p}_n)$ denote the position of all vehicles.

II.A. Voronoi Diagrams

A Voronoi diagram[3] is a decomposition of some space into a set of cells. A Voronoi diagram is parametrized by a set of points, called generators. Each cell corresponds to the set of points closest to its generator. Formally, the cell corresponding to generator $\mathbf{p}_i$ is defined as

$$V_i(P, I) = \begin{cases} \left\{ \mathbf{q} \in \mathcal{Q} \middle| ||\mathbf{p}_i - \mathbf{q}||^2 \leq ||\mathbf{p}_j - \mathbf{q}||^2 \right\}, \forall j \in I \setminus \{i\} & \text{if } i \in I \\ \emptyset & \text{otherwise} \end{cases} \tag{1}$$

where $||\cdot||$ is the Euclidean norm. We have extended the usual definition by explicitly specifying the index set (I) of the generators we are considering. This will be useful in Section III.

The neighborhood relationship between regions can be used to define the Delaunay graph. The Delaunay graph is an undirected graph with vertex set $\mathcal{I}$. There is an edge between i and j if $V_i(P, I)$ and $V_j(P, I)$ share a frontier. If $\partial\mathcal{X}$ denotes the boundary of $\mathcal{X}$, the frontier between i and j is the set $\Delta_{ij} = \partial V_i(P, \mathcal{I}) \cap \partial V_j(P, \mathcal{I})$.

Generators i and j share a frontier if Δ_{ij} has non-zero length. To conveniently express this adjacency relationship, let $N_i(P, I)$ be the set of neighbors of i when its Voronoi cell is $V_i(P, I)$.

II.B. Cost Function

The quality of the coverage task is modeled through a cost function. Minimizing cost results in increased surveillance quality. The cost function is

$$J(P, \mathcal{I}) = \sum_{i \in \mathcal{I}} \int_{V_i(P,\mathcal{I})} \phi(\mathbf{q}) \psi(\mathbf{q}, \mathbf{p}_i) \, d\mathbf{q}. \tag{2}$$

Function $\phi : \mathcal{Q} \to \mathbb{R}_{\geq 0}$ gives the importance of $\mathbf{q}$ to the surveillance task being carried out. A more important location corresponds to a larger value of ϕ. Some authors[4] use a time-varying ϕ to implement target tracking tasks.

On the other hand, $\psi : \mathcal{Q} \times \mathcal{Q} \to \mathbb{R}_{\geq 0}$ measures the sensing quality of location $\mathbf{q}$ performed by a vehicle located at $\mathbf{p}_i$. A common choice is the squared distance:

$$\psi(\mathbf{q}, \mathbf{p}_i) = ||\mathbf{q} - \mathbf{p}_i||^2.$$

With this choice, the Voronoi diagram is the space decomposition of minimum cost, for any given P. The rationale for using the Voronoi diagram when assigning regions to vehicles in a surveillance task is that the vehicle closest to a location should be best at observing that location. This effectively neglects positive interaction between two neighboring vehicles that could arise as a result of sensor fusion. This more general case where interactions are not neglected has been studied. For instance, in the case of Voronoi coverage control[5] and also with probabilistic[6] and information theoretic[7] formulations.

II.C. Minimizing the Cost Function

When the position of the vehicles can also be controlled, the so-called centroidal Voronoi configuration is of particular interest. This configuration corresponds to the case where every vehicle $i \in \mathcal{I}$ is at the centroid of its Voronoi cell. The centroid of a set $\mathcal{X} \subset \mathbb{R}^2$ is defined as

$$C_{\mathcal{X}} = \frac{\int_{\mathcal{X}} \mathbf{x} \phi(\mathbf{x}) \, d\mathbf{x}}{\int_{\mathcal{X}} \phi(\mathbf{x}) \, d\mathbf{x}}.$$

Being in centroidal configuration is a necessary condition for achieving a local minimum of the cost function.[2] This is a desired position for the vehicles and we write this as $\mathbf{p}_i^* = C_{\mathcal{Q}_i}$.

II.D. Navigating to the Desired Position

We use an obstacle avoidance scheme based on navigation functions,[8] which we recall here for completeness. It is important to note that while these obstacles prevent navigation, they do not block visibility. The use of a navigation function is used as a postprocessing step, and it is used to navigate towards $C_{\mathcal{Q}_i}$. Alternatives for coverage and navigation in nonconvex environments exist.[9,10]

A potential function $\eta_i : \mathcal{Q} \to [0, 1]$ is defined for agent i by combining information about obstacles (including other vehicles) and the desired position $\mathbf{p}_i^*$ (i.e. the centroid of its Voronoi cell). The vehicles navigate in the direction of $-\frac{\partial \eta_i}{\partial \mathbf{q}}$. The potential function is of the form

$$\eta_i(\mathbf{q}) = \frac{\gamma_i(\mathbf{q})}{\exp\left(\beta_i(\mathbf{q})^{1/k}\right)},$$

where

$$\gamma_i(\mathbf{q}) = ||\mathbf{p}_i^* - \mathbf{q}||^2$$

and

$$\beta_i(\mathbf{q}) = \prod_{j \in \mathcal{J}} \beta_{i,j}(\mathbf{q}) = \prod_{j \in \mathcal{J}} \left(1 - \lambda \frac{(||\mathbf{q}_i - \mathbf{q}_j||^2 - d^2)^2}{(||\mathbf{q}_i - \mathbf{q}_j||^2 - d^2)^2 + 1}\right)^{\frac{\operatorname{sgn}(||\mathbf{q}_i - \mathbf{q}_j|| - d) + 1}{2}}.$$

In this navigation function, $d \in \mathbb{R}_{>0}$ is a threshold on the range beyond which obstacles are ignored, $k \in \mathbb{R}_{>0}$ is a tunable shape parameter and $\lambda = \frac{1+d^4}{d^4}$. It was shown[8] that with a proper choice of k, following $-\nabla\eta_i$ results in obstacle avoidance and drives robot i to $\mathbf{p}_i^*$. However, an inappropriate choice of k or some particular obstacle configuration might result in the vehicles being trapped and unable to reach their desired position. Both cases are addressed by the proposed approach.

The use of navigation functions has an impact on the provable convergence of the vehicles to a centroidal Voronoi configuration. This is typically proved by establishing an argument that the vehicles get monotonically closer to the centroid of their cell.

It was shown,[8] that k can be chosen such that all critical points of η_i (except $\mathbf{p}_i^*$) can be arbitrarily close from the obstacles and that those critical points (except $\mathbf{p}_i^*$) are saddles. Intuitively, these saddles correspond to points where the vector flow splits to go around an obstacle. The effect of many obstacles' proximity is combined through the definition of β and it is thus possible that the saddle does not lie on the line subtended by $\mathbf{p}_i$ and $\mathbf{p}_i^*$. Because of this, $||\mathbf{p}_i^* - \mathbf{p}_i||$ might be increasing for small time intervals, namely when going around an obstacle. After the obstacle has been avoided, the distance to the centroid will continue to decrease again. More formally, let there be an obstacle of radius r at $\mathbf{c} \in \mathcal{Q}$. Assume for now that this obstacle does not move. The results of Tanner *et al.*[8] imply that, given a suitable choice of k, a properly navigating agent coming closer than $||\mathbf{p}_i^* - \mathbf{c}|| + r$ from $\mathbf{p}_i^*$ ends up closer than $||\mathbf{p}_i^* - \mathbf{c}|| - r$ from $\mathbf{p}_i^*$. Therefore, for properly chosen k and static obstacles, the use of navigation functions should not prevent convergence to a centroidal Voronoi configuration. Since a vehicle cannot reach the centroid if it lies within an obstacle, we project of the centroid on the obstacle boundary in that case.

If the obstacles are moving, the matter is more complicated. We avoid dealing with this issue by assuming that the only obstacles are team vehicles and static obstacles. We further assume that team vehicles that have suffered from navigation failures stop moving. We finally assume the diameter of the vehicles is small when compared to the diameter of a cell, so that in practice agents are usually not near a configuration where they would collide other vehicles navigating normally. These assumptions are probably more stringent than necessary, since having such an obstacle avoidance behavior does not (empirically) seem to prevent the vehicles from deploying to a centroidal Voronoi configuration, when starting from a configuration where all vehicles are close from colliding with each other.

III. Comparing the Cost of Different Positions

We now develop a criterion to compare the cost of having some agent at different positions. We consider the problem from the point of view of vehicle $i \in \mathcal{I}$. Let $\mathcal{J} = \mathcal{I} \setminus \{i\}$ be the set of all other vehicles. We are interested in stating the cost of having i at $\mathbf{p}_i'$ rather than at $\mathbf{p}_i$. Assume vehicles in $\mathcal{J}$ have their position fixed, i.e. $\mathbf{p}_j = \mathbf{p}_j'$ for $j \in \mathcal{J}$. The position of all vehicles after the move of i shall be noted P'.

If agent i is left out, $V_j(P, \mathcal{J})$ is the Voronoi cell corresponding to vehicle j. Note that since $\mathbf{p}_j = \mathbf{p}_j'$, for $j \in \mathcal{J}$, we have $V_j(P, \mathcal{J}) = V_j(P', \mathcal{J})$ for all $j \in \mathcal{I}$. For convenience, let

$$\psi_{\mathcal{I}}(P, \mathbf{q}) = \sum_{i \in \mathcal{I}} 1_{V_i(P,\mathcal{I})}(\mathbf{q}) \psi(\mathbf{q}, \mathbf{p}_i),$$

where $1_{\mathcal{X}}(x)$ is the set indicator function for $\mathcal{X}$. This lets us rewrite the cost as:

$$J(P, \mathcal{I}) = \int_{\mathcal{Q}} \psi_{\mathcal{I}}(P, \mathbf{q}) \phi(\mathbf{q}) \, d\mathbf{q}.$$

An important fact is that for fixed P, $\psi_{\mathcal{I}}(P, \mathbf{q})$ and $\psi_{\mathcal{J}}(P, \mathbf{q})$ differ only for points $\mathbf{q}$ in $V_i(P, \mathcal{I})$. This is the case because points that were in $V_i(P, \mathcal{I})$ must now be in another generator's cell. Furthermore, almost all points of $V_i(P, \mathcal{I})$ are in $\cup_{j \in N_i(P,\mathcal{I})} V_j(P, \mathcal{J})$. This means the restriction of $\psi_{\mathcal{J}}(P, \cdot)$ to $V_i(P, \mathcal{I})$ can be computed using only the position of i and other vehicles in $N_i(P, \mathcal{I})$.

The difference between the cost of having i at $\mathbf{p}_i$ and not having i at all is given by:

$$\left[J(\cdot, \mathcal{I}) - J(\cdot, \mathcal{J})\right](P) = \int_{\mathcal{Q}} \left[\psi_{\mathcal{I}}(P, \mathbf{q}) - \psi_{\mathcal{J}}(P, \mathbf{q})\right] \phi(\mathbf{q}) \, d\mathbf{q} = \int_{V_i(P,\mathcal{I})} \left[\psi(\mathbf{q}, \mathbf{p}_i) - \psi_{\mathcal{J}}(\mathbf{q})\right] \phi(\mathbf{q}) \, d\mathbf{q}.$$

This quantity is always strictly positive, because $\psi(\mathbf{q}, \mathbf{p})$ is a decreasing function of the the distance between $\mathbf{q}$ and $\mathbf{p}$ and because the removal of the i must have resulted in an increased distance for almost all points in $V_i(P,\mathcal{I})$. This is by the definition of a Voronoi cell. This cost difference can also be computed using only the position i and other agents in $N_i(P,\mathcal{I})$, because it is zero outside $V_i(P,\mathcal{I})$ as noted earlier. If we add i back at $\mathbf{p}'_i$, a similar argument holds for $[J(\cdot,\mathcal{J}) - J(\cdot,\mathcal{I})](P')$, except that this quantity is *negative*; i.e. adding a vehicle results in a lower cost. In this case, the cost difference is also zero outside $V_i(P',\mathcal{I})$). If j is such that $\mathbf{p}'_i \in V_j(P,\mathcal{J})$, then $j \in N_i(P',\mathcal{I})$. In some cases however, there exist an agent k such that $k \notin N_j(P,\mathcal{I})$ and $k \in N_i(P',\mathcal{I})$. One can show that if such a k exists, then $k \in \bigcup_{l \in N_j(P,\mathcal{I})} N_l(P,\mathcal{I})$; i.e. it is at most a second neighbor of j. We postpone a discussion about the consequences of this until the end of this section.

The difference in cost between having i at $\mathbf{p}'_i$, rather than at $\mathbf{p}_i$ is therefore

$$[J(\cdot,\mathcal{I}) - J(\cdot,\mathcal{J})](P') - [J(\cdot,\mathcal{I}) - J(\cdot,\mathcal{J})](P) \tag{3}$$

$$= \int_{V_i(P',\mathcal{I})} [\psi(\mathbf{q},\mathbf{p}'_i) - \psi_{\mathcal{J}}(P',\mathbf{q})]\,\phi(\mathbf{q})\,d\mathbf{q} - \int_{V_i(P,\mathcal{I})} [\psi(\mathbf{q},\mathbf{p}_i) - \psi_{\mathcal{J}}(P,\mathbf{q})]\,\phi(\mathbf{q})\,d\mathbf{q}. \tag{4}$$

$$= \left[\int_{V_i(P',\mathcal{I})\cap V_i(P',\mathcal{I})} \psi(\mathbf{q},\mathbf{p}'_i)\phi(\mathbf{q})\,d\mathbf{q} + \int_{V_i(P',\mathcal{I})\setminus V_i(P,\mathcal{I})} [\psi(\mathbf{q},\mathbf{p}'_i) - \psi_{\mathcal{J}}(P,\mathbf{q})]\,\phi(\mathbf{q})\,d\mathbf{q}\right]$$
$$- \left[\int_{V_i(P,\mathcal{I})\cap V_i(P',\mathcal{I})} \psi(\mathbf{q},\mathbf{p}_i)\phi(\mathbf{q})\,d\mathbf{q} + \int_{V_i(P,\mathcal{I})\setminus V_i(P',\mathcal{I})} [\psi(\mathbf{q},\mathbf{p}_i) - \psi_{\mathcal{J}}(P,\mathbf{q})]\,\phi(\mathbf{q})\,d\mathbf{q}\right]. \tag{5}$$

Equation (5) was obtained by using $\mathcal{X} = (\mathcal{X}\cap\mathcal{Y}) \cup (\mathcal{X}\setminus\mathcal{Y})$, for any $\mathcal{Y}$, on the integration domain and noting that $\psi_{\mathcal{J}}(P,\cdot)$ and $\psi_{\mathcal{J}}(P',\cdot)$ cancel out on $V_i(P,\mathcal{I}) \cap V_i(P',\mathcal{I})$. The benefit of this simplification is that the computation of $\psi_{\mathcal{J}}$ on $V_i(P,\mathcal{I}) \cap V_i(P',\mathcal{I})$ requires i to have information about agents that are its neighbors in either $\mathbf{p}_i$ or $\mathbf{p}'_i$. Although one agent in $N_i(P,\mathcal{I})$ has information about the latter set of neighbors, this information might not be readily available to i itself. Without this cancellation, i and j (the vehicle whose cell contains $\mathbf{p}'_i$) would have to exchange that information as well. The first term in (5) can be computed by j whereas the second one can be computed by i. These two vehicles can decide whether i should move from $\mathbf{p}_i$ to $\mathbf{p}'_i$ by exchanging only their corresponding term of eq. (5). The criterion is that $\mathbf{p}'_i$ is a better position for agent i than $\mathbf{p}_i$ if eq. (3) is negative. This criterion accounts for all changes in Voronoi cells that would result from the move of i.

We now revisit the question of knowing the neighbors of i in the space decomposition induced by P'. We mentioned earlier that $N_i(P',\mathcal{I})$ is the set of second neighbors of the agent whose cell contains $\mathbf{p}'_i$. Considering these second neighbors might be inconvenient or unpractical. Suppose k is a second neighbor. If we omit k we are actually using $\psi(\cdot,\mathbf{p}_k)$ rather than $\psi(\cdot,\mathbf{p}'_i)$ on $V_i(P',\mathcal{I}) \cap V_k(P,\mathcal{I})$ in the first line of eq. (5). This results in an increased value for eq. (5), again because ψ is a decreasing function and by definition of a Voronoi cell. Therefore, neglecting those second neighbors of j which are not also first neighbors means i might decrease the cost difference but does not invalidate the use of eq. 5 as a test for improvement. In other words, ignoring second neighbors could result in the rejection of a better position, but not in the acceptance of a worse position.

A similar argument is used to justify the use of $\mathbf{p}'_i = C_{V_j(P,\mathcal{I})}$ in the next section, even though j is not necessarily in a centroidal configuration. (If j can not move, it will not reach the centroid of its cell by itself.) When a vehicle is not in centroidal configuration, eq. 5 has its first line greater than if the vehicle was in a centroidal configuration. Therefore, the use of the test proposed in this section will not cause a move to a worse position.

IV. Selecting the Best Vehicle to Help a Failed Vehicle.

To prevent the vehicles from being stuck in a bad local optima due to failures, we are interested in letting robots head to a point outside their own cell. We now propose and describe a mechanism to enable such a behavior. We call this the *help mechanism.* The criterion of Section III shall be used to determine whether moving to a point outside the cell is an improvement. The mechanism is triggered by the detection of the inability to reach some desired location. This detection mechanism is in turn described in Section V.

Communication is required to implement the help mechanism. We assume agents can send each other structured messages. Throughout, we assume i is the helping agent and j is the agent requesting help. A few different messages can be sent:

- Request$(j,\mathbf{p}^*)$: j requests help because it can not reach $\mathbf{p}^*$. In the current setting, $\mathbf{p}^*$ is the centroid of $V_j(P, \mathcal{J})$. This is a broadcast message.

- RESPONSE($i, j, J_j^i(\mathbf{p}^*)$): i replies to j's help request. The message contains the second line of eq. (5), which we note $J_j^i(\mathbf{p}^*)$.
- OFFER($j, i, J_i^j(\mathbf{p}^*)$): j offers i the opportunity of occupying $\mathbf{p}^*$. $J_i^j(\mathbf{p}^*)$ is the first line of eq. (5).
- ACCEPT(i, j): i accepts j's offer and announce it will move to $\mathbf{p}^*$.
- DECLINE(x, y): x declines y's OFFER or RESPONSE. The former happens if (5) is greater than or equal to zero. The other case arise if some other agent $k \notin \{i, j\}$ accepted an offer that was made to it. In this case, j is the requesting agent, and i is the offering agent. To prevent agents from being stuck waiting for a reply, agent i enters the DECLINEPENDING state and it notifies agents in the WAITINGOFFER state that the task was awarded to another agent.

In accordance with the previous section, the offer shall be accepted if $J_i^j(\mathbf{p}^*) < J_j^i(\mathbf{p}^*)$.

The messages described above are exchanged in a defined order. The behavior of any agent implementing this help protocol are described using a Finite State Machine (FSM). There is an instance of the FSM on each agent, with possibly different states. The transitions in a state machine are triggered by events exogenous or endogenous to the agent. Exogenous events comprise the reception of messages sent by other agents. Endogenous events are either the timing out of a waiting period or the detection of a failure. These wait periods are added to avoid deadlocks should a message be lost.

Figure 2 depicts the help protocol as a state diagram. The resulting communication is pretty straightforward. The transitions out of waiting states (due to timeouts) are not shown for clarity. These can occur in states WAITING REPLY, WAITING RESPONSE (Figure 2(b)) and WAITING OFFER (Figure 2(a)). Finally, when a failure to reach the desired position is detected, the state of an agent switches from that of Figure 2(a) to that of Figure 2(b).

V. Detecting Failures

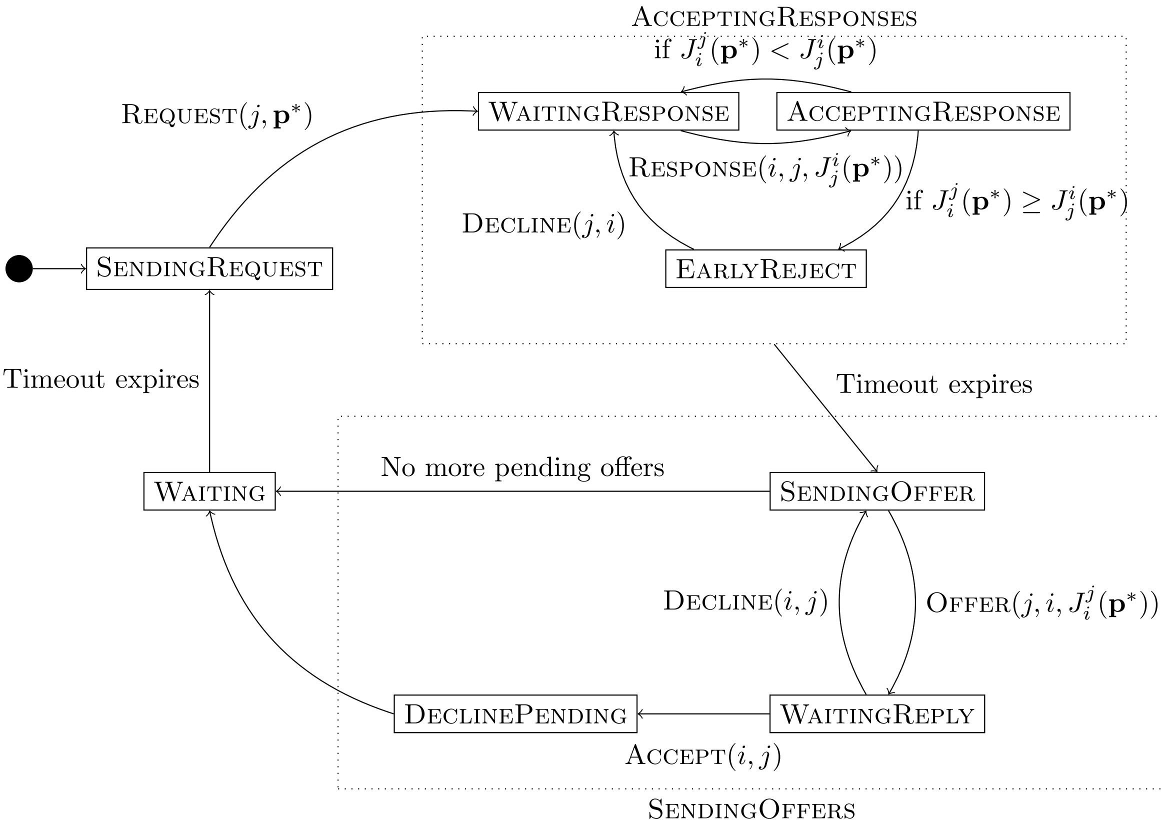
The algorithm of the Section IV requires that an agent can detect that it is unable to reach its desired position. We now propose a way to achieve this. The basic idea is to establish a test on the distance to the desired position. If this distance is greater than some threshold but is no longer decreasing, we shall conclude that the destination is unreachable and we use that to trigger the help protocol described previously. In practice, the situation is a bit more complicated. Measurement noise on the position may cause the observed distance to the desired location to be nondecreasing if one only looks at a pair of measurements. Furthermore, the obstacle avoidance subsystem may cause the vehicle to not travel in a straight line to the desired position. In this case, the distance between the desired and actual positions might increase, even though the vehicle is still functioning properly and heading to its desired position.

In order to check if the vehicle is closing in to its desired location, we sample the remaining distance between it and its desired location at regular time intervals. We use these samples to perform the so-called sign test. This test is performed by every vehicle using its own measured position. We first describe this test and then proceed to explain how it can be used to detect failures in the problem at hand.

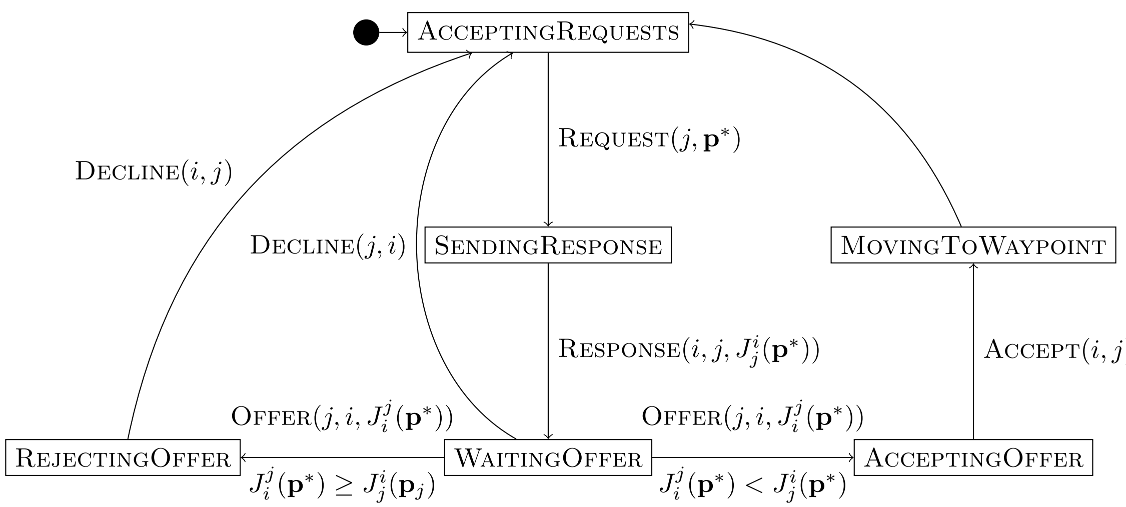
V.A. The Sign Test

The sign test for paired samples [11, Chap. 16] is a nonparametric test used to test the probability that a random variable is larger than another one. The test uses n pairs of samples drawn from two random variables: $(X, Y) = (X_i, Y_i)_{i=1}^n$. We want to test for $p = \Pr(Y < X)$. We introduce the null hypothesis $H_0 : p = 0.5$. The null hypothesis represents the equality of X and Y. The test statistic k is the number of samples pairs for which $Y < X$: $k = |\{i \mid Y_i < X_i\}|$. The test statistic is a realization of a binomial random variable $K \sim B(n, 0.5)$. We consider the alternative hypothesis $H_1 : p > 0.5$ whose p-value is given by $\Pr(k < K)$. Therefore, we reject H_0 in favor of H_1 when the latter's p-value is small. For completeness, the p-value is given by

$$\Pr(k < K) = 1 - \Pr(K \le k) = 1 - \sum_{i=0}^{\lfloor k \rfloor} \binom{n}{i} p^i (1-p)^{n-i} = 1 - I_{1-p}(n-k, k+1),$$



(a) From the point of view of j, the agent being helped.



(b) From the point of view of $i \neq j$, an agent helping j.

Figure 2. Help protocol state diagram

where $I_z(a,b)$ is the regularized incomplete beta function. The only assumption made by the sign test is that $\{Y_i < X_i\}_{i=1}^N$ must be a collection of independent random variables.

V.B. Using the Sign Test

We now establish a link between the sign test and whether the distance between $\mathbf{p}_i$ and $\tilde{\mathbf{p}}_i^*$ is decreasing. To use the sign test, we use a set of $N+1$ samples $D = \{d_i\}_{i=1}^{N+1}$ of the distance to the desired position. Each of those samples is assumed to have been collected at regular time intervals and that they are ordered, i.e. d_i has been collected before d_j if and only if $i < j$. We then pose $X = \{d_i\}_{i=1}^N$ and $Y = \{d_i\}_{i=2}^{N+1}$. This results in $(Y_i < X_i) \equiv (d_{i+1} < d_i)$ for all $i \in \{1, \dots, N\}$. If this inequality is satisfied, the vehicle has come closer within that time interval. The sign test is then applied directly.

The vehicles navigate to their desired position using the navigation function scheme described in Section II. While avoiding an obstacle, the vehicle can follow a curved trajectory. At some points of this trajectory the distance to the desired position might increase, although the vehicle is actually approaching the desired position. We address this to avoid spurious fault detection.

Reusing the notation of Section II, suppose vehicle i is located at $\mathbf{p}_i$ and must travel to $\mathbf{p}_i^*$. The vehicle travels in the direction of $-\nabla\eta_i$. If the vehicle is not in collision with an obstacle at $\mathbf{p}_i$, there is a path S_i from $\mathbf{p}_i$ to some local minimum $\tilde{\mathbf{p}}_i^*$ of η_i. For some obstacle configurations or inappropriate choices of k, it is possible that $\tilde{\mathbf{p}}_i^* \neq \mathbf{p}_i^*$. In all cases, since η_i is known, it is possible to compute both S_i and $\mathbf{p}_i^*$. Rather than using the Euclidean distance to $\mathbf{p}_i^*$, it suffices to use the test on the length of S_i. Each time the length of S_i is used, it is computed assuming the position of all other objects is fixed. Henceforth, the d_i and d_j, $i \neq j$ represent the distance to a (possibly different) centroid, under possibly different obstacle configurations.

The intuitive rationale behind using the sign test is that usually, the path does not change much from one time step to another and that since a properly operating vehicle should move towards desired position, having mostly negative differences should indicate that the vehicle is indeed operating properly.

Arguably, $(d_i < d_{i-1})$ and $(d_{i+1} < d_i)$ are not really independent because $(d_i - d_{i-1})$ and $(d_{i+1} - d_i)$ are not independent given the vehicle's velocity and acceleration at some point in time. However, *we assume independence*. This assumption could be relaxed and information about vehicle motion could be incorporated by using a version of the sign test based on the a dependent binomial distribution[12] (i.e. a binomial distribution constructed using dependent Bernoulli trials). This is out of the scope of the present paper.

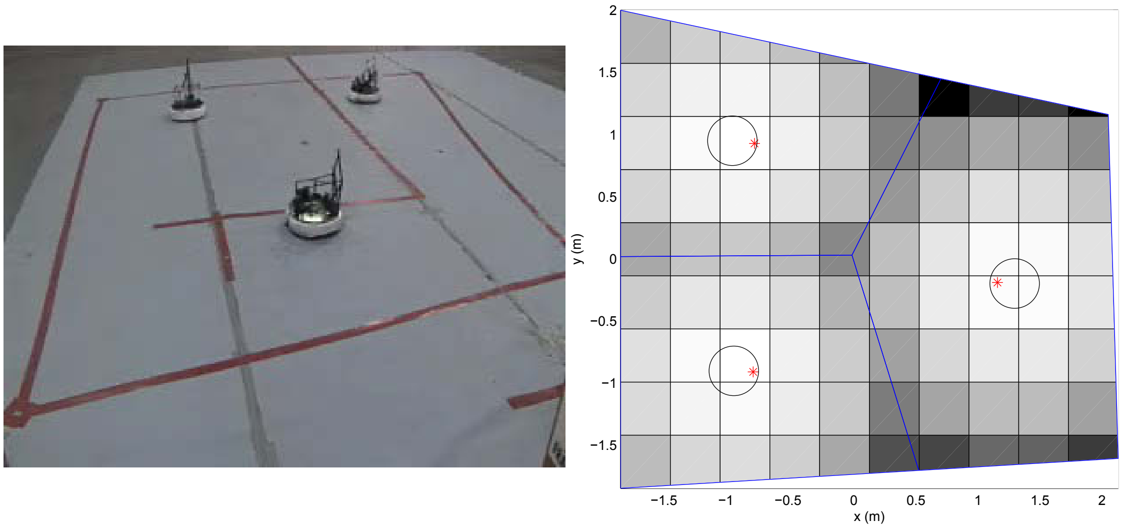
VI. Experimental Evaluation

Experimental evaluation was carried out using three wheeled mobile robots (WMRs),[13] depicted in Figure 3(a). The vehicles' position is obtained through an infrared camera-based positioning system.[14] While the experiments are carried out using WMRs, the proposed approach is independent of vehicle dynamics. In fact, this work is a intermediate stage and the goal is to use a mixture of WMRs and quad-rotor unmanned aerial vehicles (UAVs). The only requirement we have is that a low level controller is provided to follow the trajectory generated by the obstacle avoidance subsystem. In the current experiments, a piecewise linear approximation of the trajectory was used. This trajectory was in turn followed by a feedback-linearization unicycle controller.[15,16]

VI.A. Results

VI.A.1. Self-monitoring

The self-monitoring test of Section V was used to detect whether a vehicle is closing in its desired location. Failures were injected artificially, by enabling or disabling the low level controller (the unicycle). This results in an instantaneous stopping of the vehicle. The fault detection test was not made aware of this switching. The test was using distance to the target location. This distance was computed using position from the camera system. Figure 4 shows the observed relationship between the p-value discussed in Section V and simulated failures. The dashed line represents whether the low level controller was enabled (1) or disabled (0). A low p-value means the vehicle is likely to be failed. In this experiment, the test was performed using 20 samples ($N = 20$). Near time 15 seconds, the vehicle has reached its destination and it stops moving. The test properly detects that the vehicle has stopped moving. This is not a fault and this is why the distance



(a) Experimental setup

(b) Representation

Figure 3. Experimental setup and representation. Robots are represented with circles, centroid of regions with stars. Shading represent surveillance quality: light is good; dark is poor.

to target position is also considered as a criterion to send an help request. Also, notice that choosing a lower threshold for the p-value, results in a later time of detection for both the occurrence of a fault and the detection of a return to a non faulty state.

VI.A.2. Help Protocol

The experiment is set up similarly to the example of Figure 1. The experiment was carried out with three vehicles, whose initial configuration is given by Figure 5(a). Vehicle 1 had its low level controller disabled, much like in the previous experiment. The sign test was used to detect failures. A vehicle would declare itself faulty whenever p-value became lower than 0.05. Upon fault detection, vehicle 1 requested help and vehicles 2 and 3 offered it. Vehicle 3 was elected to help vehicle 1. Figure 5(c) shows the path it used to reach the desired position. This path is not a straight line, because the actual trajectory is determined by the obstacle avoidance subsystems, which tends to favor paths that are far from obstacles. Here, the only obstacles were the other vehicles. The total cost (Figure 5(d)) is lower in the final configuration (Figure 5(b)), as expected. Finally, this configuration is not be best configuration that can be reached with vehicle 1 being blocked, because a configuration similar to that of Figure 1(a) is also possible. However, such a configuration is not reached because the proposed criterion does not indicate that moving vehicle 2 to the desired position of vehicle 1 (shown as a star) would result in a decreased cost.

The state MovingToWaypoint is currently implemented as a waiting period. However, it would be better if it was implemented using a test on the cost of the vehicles. The reason for this is that a vehicle helping another one can accept offers while it is not near the centroid of its own cell. In such a configuration, its cost is not minimal and the acceptance of the help offer is biased, i.e. eq. (5) is lower. Indeed, a vehicle not at the centroid of its cell overestimates the cost of its current cell, which makes it prone to accepting offers which might actually be worse than if it was allowed to reach a centroidal configuration. This can result in an undesirable behavior, namely the oscillation between two configurations which are not local minimums. This results in a higher cost because even when moving from a locally minimal configuration to another, the cost increases as shown in Figure 5(d), between times $t = 3$ and $t = 6$. Exiting that state when the vehicle is near its optimal position is a more sensible thing to do as it would prevent such an oscillatory behavior. An additional benefit is that the timeout period does not have to be tuned manually. The choice of this constant depends on the vehicle dynamics, in that a vehicle that moves more slowly should ignore help requests for a longer time. In the experiments, this timeout was set to 10 seconds.

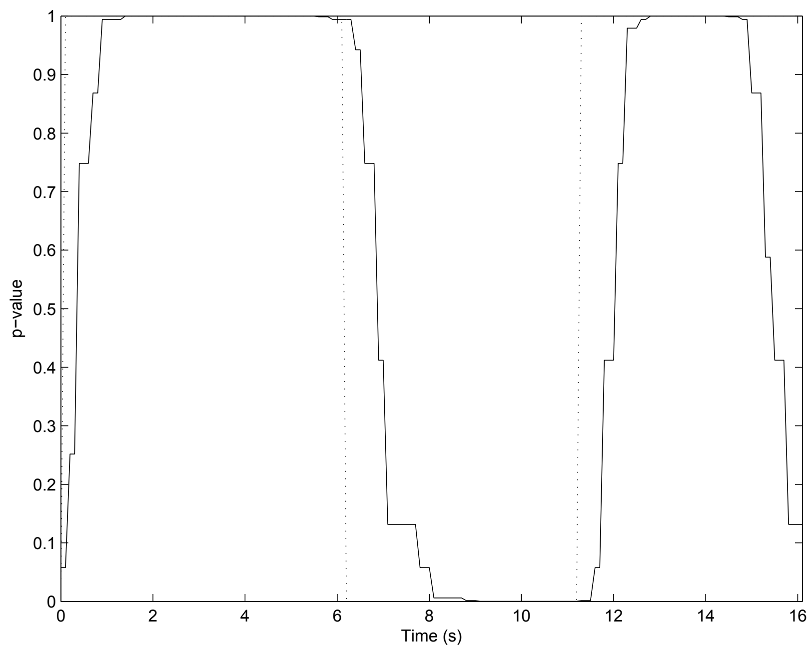


Figure 4. Relationship Between Failures and p-value

VII. Conclusions

This paper presented a mechanism to avoid a limitation of Voronoi coverage control in practice, i.e. when vehicles can suffer from various kinds of failures that prevent them from moving to the desired location. The proposed approach combined a way to compare the quality of various positions, a method to select a replacement vehicle for a failed vehicle and suggested a method to test that a vehicle indeed suffers from a navigation failure. Future work includes the automatic determination of the time at which a vehicle can respond do help requests as well as the performing experiments with a larger group of vehicles comprising both UAVs and WMRs.

References

[1] Carlson, J. and Murphy, R., "How UGVs Physically Fail in the Field," *IEEE Transactions on Robotics*, June 2005, pp. 423–437.

[2] Cortés, J., Martinez, S., Karatas, T., and Bullo, F., "Coverage control for mobile sensing networks," *IEEE Transactions on Robotics and Automation*, Vol. 20, No. 2, 2004, pp. 243–255.

[3] Aurenhammer, F., "Voronoi diagrams—a survey of a fundamental geometric data structure," *ACM Computing Surveys*, Vol. 23, No. 3, 1991, pp. 345–405.

[4] Pimenta, L., Schwager, M., Lindsey, Q., Kumar, V., Rus, D., Mesquita, R., and Pereira, G., "Simultaneous coverage and tracking (SCAT) of moving targets with robot networks," *Algorithmic Foundation of Robotics VIII*, 2009, pp. 85–99.

[5] Schwager, M., *A gradient optimization approach to adaptive multi-robot control*, Ph.D. thesis, Massachusetts Institute of Technology, 2009.

[6] Li, W. and Cassandras, C., "Distributed cooperative coverage control of sensor networks," *IEEE Conference on Decision and Control and European Control Conference*, IEEE, 2006, pp. 2542–2547.

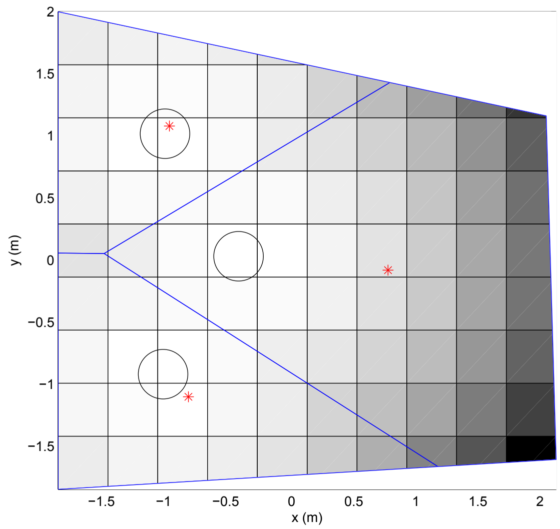
[7] Grocholsky, B., Makarenko, A., and Durrant-Whyte, H., "Information-theoretic coordinated control of multiple sensor platforms," *IEEE International Conference on Robotics and Automation*, Vol. 1, 2003, pp. 1521–1526.

[8] Tanner, H. and Kumar, A., "Towards decentralization of multi-robot navigation functions," *IEEE International Conference on Robotics and Automation*, IEEE, 2006, pp. 4132–4137.

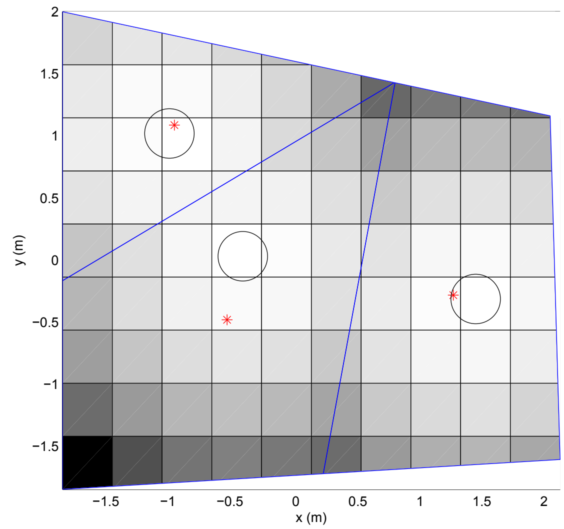
[9] Ganguli, A., Cortés, J., and Bullo, F., "Distributed coverage of nonconvex environments," *Networked Sensing Information and Control*, 2008, pp. 289–305.

[10] Breitenmoser, A., Schwager, M., Metzger, J.-C., Siegwart, R., and Rus, D., "Voronoi coverage of non-convex environments with a group of networked robots," *IEEE International Conference on Robotics and Automation*, May 2010, pp. 4982–4989.

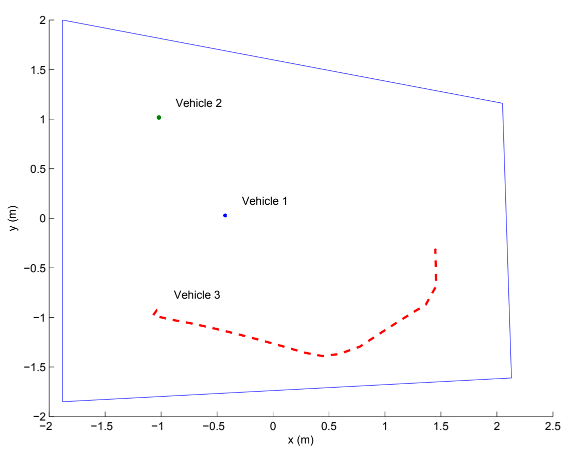
[11] Hines, W. and Montgomery, D., *Probability and statistics in engineering and management science*, Wiley New York, 1980.



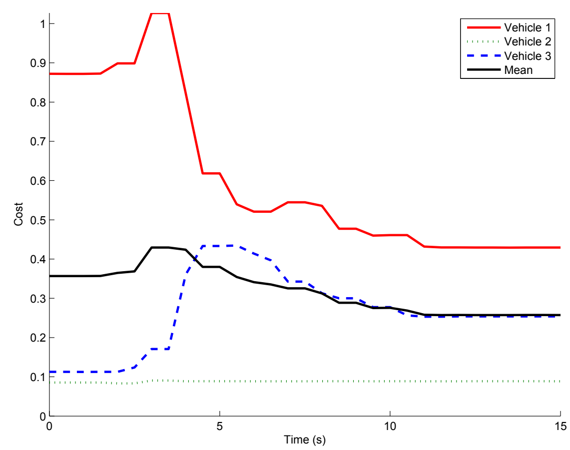
(a) Coverage at $t = 0$



(b) Coverage at $t = 15$



(c) Trajectories



(d) Cost

Figure 5. Behavior of three agents. Agent 1 can not move and asks for help.

[12]Gupta, R. and Tao, H., "A generalized correlated binomial distribution with application in multiple testing problems," *Metrika*, Vol. 71, 2010, pp. 59–77, 10.1007/s00184-008-0202-7.

[13]"Quanser Qbot User Manual, Rev. 1," 2009.

[14]"http://www.naturalpoint.com/optitrack/," .

[15]Léchevin, N. and Rabbath, C., "Sampled-data Control of a Class of Nonlinear Flat Systems with Application to Unicycle Trajectory Tracking," *Journal of Dynamic Systems, Measurement, and Control*, Vol. 128, No. 3, 2006, pp. 722–728.

[16]Oriolo, G., Luca, A. D., and Vendittelli, M., "WMR Control Via Dynamic Feedback Linearization: Design, Implementation, and Experimental Validation," *IEEE Transactions on Control Systems Technology*, Vol. 10, No. 6, 2002, pp. 835–852.