A Mobile Network for Mobile Sensors

Andrea Simonetto
Dipartimento di Ingegneria Aerospaziale
Politecnico di Milano
via la Masa 34, 20156 Milano, Italy
Email: andrea.simonetto@asp-poli.it

Paul Scerri, Katia Sycara
School of Computer Science
Carnegie Mellon University
Pittsburgh, 15213, PA, USA
Email: {pscerri, katia}@cs.cmu.edu

***Abstract*— When cooperative robots are used for exploring an environment, a key component of their task is to relay data back to a human operator and get input from that operator on exploration priorities. To do this a wireless communication network is required. This paper presents an approach to creating and maintaining an ad hoc communication network for relaying data to and from exploring robots. The network is made up of mobile robots acting as communication hubs. The approach relies on the concept of potential fields, but allows the potential fields to change dynamically depending on the current needs of the team. Specifically, aspects of the potential fields are turned on or off for particular robots, based on required connectivity. The resulting potential fields push the robots towards appropriate locations to maintain the network. Several policies for determining which aspects of the potential field to activate were evaluated, with policies focused on maintaining minimum connectivity performing best. In four distinct simulated environments, the dynamic potential fields approach was shown to effectively maintain a communications network and far out-perform a standard potential field approach.**

Keywords: Ad hoc communication networks, multi robot systems, Potential Fields.

I. Introduction

Robots are increasingly being used to search or explore interesting and risky environments to provide information for human operators. Since, in many of those environments, there will be no wireless network infrastructure to connect robots with operators, such a network has to be dynamically formed in some way. One promising approach is to send additional communication robots to form the network and keep sensor robots and human controllers in contact. For example, if sensor robots are sent into a burning building, it is likely that the fire will disable any building wireless networks and the building's structure disrupt direct wireless communications from the outside, thus additional robots, working as communication hubs, may be the only way to stay in contact. Other domains where such a technology might be useful include mines [1], military [2], space [3] and desert exploration [4].

Dynamically deploying effective networks is difficult for a variety of reasons. First, the communication robots will not have a priori knowledge of where the sensor robots will go, nor of the environment in which they must deploy. Second, to coordinate their deployment they must maintain communication with each other or coordinate without communication. Even if communication between the robots is available, its use has to be minimized, both to make bandwidth available for sensor robots to relay information back to the operator and to allow commands to be relayed to robots. Third, there is typically no clearly defined deployment stage, thus the ad hoc network needs to be maintained for the sensor robots while the communication robots deploy to their positions. Finally, the robots may need to constantly rearrange to adjust to sensor robot movement or failed communication robots, since typically they cannot provide coverage to the whole environment.

A variety of approaches have been developed for this problem. In ad hoc networks, distributed algorithms, which can assure k - connected graphs [5], allow robust robot positioning [6] and provide good coverage [7], have been applied in relatively open environments. However those efforts largely ignore situations in which signals are impeded by obstacles, like walls, or in which only a small dynamically changing part of environment needs coverage. Potential field are lightweight and robust way of positioning robots in a clustered and complex environment [8], often not requiring any communication to coordinate. However, potential fields are best suited for spreading robots out across an environment, not focusing them on dynamically changing areas. Hence, for this application, key extensions to potential fields were required to take advantage of their strengths while meeting problem constraints.

The central idea of this work is to dynamically change the applicable potential fields based in the current overall needs of the team. If the potential fields can be appropriately varied, the robots will robustly move to locations where a connected network can be formed. The key to the dynamic potential field approach is to ensure that each communications robot is influenced by appropriate fields at appropiate times. Specifically, the team must configure itself so that some communication robots move near to the sensor robots, while others position themselves to relay massages to and from the hub. To achieve this, each robot sends out requests for other robots to connect it back to the hub or in the hub's case, sends out requests to be connected to the network. These requests are in the form of *tokens*. When a robot receives a token it either keeps the token, adds a potential field corresponding to the request for support represented by the token, or passes the token on to another robot (which faces the same choice). By controlling the number of tokens each robot sends out, the number of links the team tries to form with the requester can effectively be controlled. The policy by which a robot decides to keep a token, and add the corresponding field, or pass the token on, dictates the effectiveness and the nature of the network. Various policies for accepting or passing on tokens have been evaluated, with the most effective policies, based on connectivity, leading to consistently well connected networks.

If a situation changes quickly or sensor robots can suddenly go out of communication range, e.g., by going around a corner, simply having the communication robots following the sensor robots will not be sufficient to maintain connectivity. In such situations, communications robots need to identify areas where sensor robots might move and proactively act to set up network connectivity in those areas. Experiments show that selecting potential fields to encourage communication robots to fill potential future gaps in the network, substantially improves connectivity over time.

Four different scenarios have been tested and the results show a significant performance improvement over a standard potential field approach, with regard to connectivity and overall efficiency of the network. Moreover, network fault-tolerance and robustness were shown to be good, even with high robot failure rates.

II. Problem Statement

Let $S = \{S_1, \ldots, S_n, H\}$ be a set of moving robots, S_i, called *sensors*, and a hub, H, and let $C = \{C^1, \ldots, C^m\}$ be a set of communication robots, the *comms*. The basic aim is to position C to create a network which connects each S_i to H and dynamically maintain connectivity.

S is assumed to be independent of C but all the robots can move at the same speed. S_i and C^i have a maximum range of communication, d_c. It is assumed that every robot can sense where the others are if they are within their communication range and robots can distinguish between *sensors* and *comms*. This may be done by overhearing messages broadcasted by other robots. Let $\overline{S} \subseteq S$ and $\overline{C} \subseteq C$ be the subsets of *sensors* and *comms* respectively that a robot can sense.

Let x be the position of a generic robot at a given time, while the hub, H is stationary. Let $\mathcal{P}_i(S_k) = \{S_k, C^i, \ldots, C^q, H\}$ be a possible communication path from $S_k \in S$ to H, thus the distance between two consecutive elements, p_i, of $\mathcal{P}_i(S_k)$ is at most d_c, $|x(p_i) - x(p_{i-1})| \leq d_c$. Two paths from the same *sensor* are different if they involve different *comms*, i.e.,

$$\mathcal{P}_i(S_k) \neq \mathcal{P}_j(S_k) \Leftrightarrow \nexists C^i | (C^i \in \mathcal{P}_i(S_k)) \wedge (C^i \in \mathcal{P}_j(S_k))$$

Among all the $\mathcal{P}_i(S_k)$ which have at least a C^i in common, a minimal path can be defined as the one which involves the minimum number of *comms*: $\min\mathcal{P}_i(S_k) = \mathcal{P}_i(S_k)$ if $|\mathcal{P}_i(S_k)|$ is minimum. All different paths from the same *sensor* can be grouped in the local subset of different paths, $P(S_k)$: $P(S_k) = \{\ldots, \mathcal{P}_i(S_k), \ldots\}$ where

$$\forall \mathcal{P}_i(S_k), \mathcal{P}_j(S_k) \,|\, (i \neq j) \wedge (\mathcal{P}_i(S_k), \mathcal{P}_j(S_k) \in P(S_k)) \Rightarrow$$
$$\Rightarrow \mathcal{P}_i(S_k) \neq \mathcal{P}_j(S_k)$$

Let the local connectivity c be $c = |\overline{C}|$ and let the connectivity *Sensor* - H at a given time t, be $\mathcal{K}_i(t) = |P(S_i)|$. Thus, $\mathcal{K}_i(t) = 0$ means there are no communication paths from S_i to H, thus S_i is not connected. The primary goal of C is to avoid this happening. In Figure 1, $\mathcal{K}_1(t) = 1$ and $\mathcal{K}_2(t) = 2$. Let the global connectivity at a given time be $\mathcal{K}(t) = \min_i \mathcal{K}_i(t)$. $\mathcal{K}(t)$ is 1 in Figure 1. A *comm* is *useful* if it is on a minimal path. Let the *used comms* subset be the subset: $\mathcal{U} = \left(\bigcup_i \min\mathcal{P}_i\right) / S$, i.e., the useful *comms*. Define efficiency, $\mathcal{E}$, as the ratio of useful *comms* to total *comms*: $\mathcal{E} = |\mathcal{U}|/|C|$. In Figure 1, $\mathcal{E} = 3/4$, since C^1 is not on a minimal path.

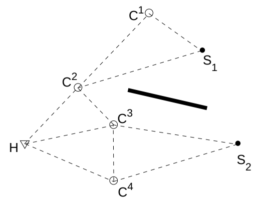


Fig. 1. An example of possible network which connects two *sensors*, S_1 and S_2 and *comms* $C^1, \ldots, C^4$ to the hub H. The dashed line are communication links, the black line is a wall.

Let $< \mathcal{K} >$ and $< \mathcal{E} >$ be the average global connectivity over time and the average efficiency over time respectively.

Finally let v be the *environment change rate*, characterized as the maximum rate a *comm* has to move to prevent the network breakdown. This gives a rough measure of the environment difficult for C.

Thus the overall problem addressed by the work is to maximize:

$$\max\left(\min_{0 \leq t \leq t_{\max}} \mathcal{K}(t)\right)$$

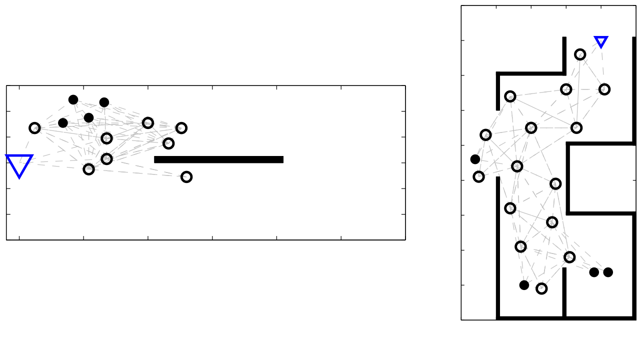
A. Scenarios

While, the space of possible situations for this work is infinite, for experimental purposes, a small set of fixed scenarios was considered. Four different environments were used, see Figure 2, namely:

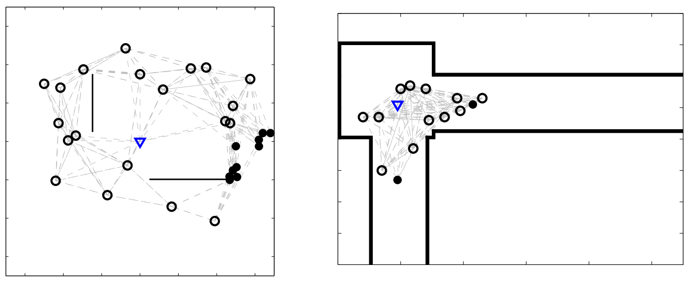
(a) *Wall*. The *sensors* move away from the hub, encountering a wall, the *comms* are initially spread in both sides of the wall. (Figure 2a). v is high because *sensors* move constantly away from the hub.
(b) *Office*. Four *sensors* move from a lift to some offices. (Figure 2b). v is low, because *comms* are initially well spread.
(c) *Open space*. A central hub, *comms* are initially spread in a box, *sensors* enter the box from two different random locations. (Figure 2c). v is at most the same as in (b).
(d) *Corridor*. Two *sensors* move away from the hub, one returns after some time. (Figure 2d). v is very high because, both the *sensors* move away from H and the *comms* are initially in a small box around H.

III. Positioning Algorithm

The basic concept of a potential field is to overlap fields representing different influences on the robot. The robot then simply follows the gradient down the resulting field. The basic



(a) Wall (b) Office



(c) Open space (d) Corridor

Fig. 2. Environments: Black dots are the *sensors*, white dots the *comms*, dashed lines are communication links, big triangles are the hubs.

potential function, $\mathcal{J}_j$, utilizes the Lennard - Jones formulation [9], resulting for each *comm*, in:

$$\mathcal{J}_j(\widehat{S},\widehat{C}) = \alpha \sum_{S_i \in \widehat{S}} -2\left(\frac{d_c f_s}{r_{C^j S_i}}\right)^6 + \left(\frac{d_c f_s}{r_{C^j S_i}}\right)^{12} + \beta \sum_{C^q \in \widehat{C}} -2\left(\frac{d_c f_c}{r_{C^j C^q}}\right)^6 + \left(\frac{d_c f_c}{r_{C^j C^q}}\right)^{12} \quad (1)$$

where $\widehat{S} \subseteq S$ and $\widehat{C} \subseteq C$ are the subsets of *sensors* and *comms* which influence a robot, $r_{C^j S_i}$ and $r_{C^j C^q}$ are the relative distances between C^j and the robots in those subsets. The communication distance d_c and the f_s and f_c coefficients determine the function shape. The coefficients α and β influence whether to move further from *sensors* or *comms*. If $|\widehat{S}| = 1, |\widehat{C}| = 0$, $\mathcal{J}_j(\widehat{S},\widehat{C})$ would have a minimum at a distance $d_c f_s$ from S_i; below that distance $\mathcal{J}_j(\widehat{S},\widehat{C})$ would increase (repulsive part) to prevent robots from being too near one another, while above that distance (attractive part) it would increase to keep robots in the communication range. Once the potential function has been evaluated, the robot moves toward the local minimum.

In the next three section different versions of this approach will be presented. What varies among the versions are $\widehat{S}$ and $\widehat{C}$, i.e., which robots effect the potential field. First the basic potential field algorithm, in which $\widehat{S} = \overline{S}$ and $\widehat{C} = \overline{C}$, i.e., where the potential field is influenced by all robots in sensor range. Second, a version where tokens are passed around the team, with the robot represented by the token being an influence in $\widehat{S}$ and $\widehat{C}$. Third, a proactive token algorithm where $\widehat{S}$ and $\widehat{C}$ are augmented by influences that fill future potential network gaps.

A. Standard algorithm

In the basic algorithm, referred to as *standard*, $\widehat{S} = \overline{S}$ and $\widehat{C} = \overline{C}$, thus every sensed robot influences the potential field shape. This leads to the robots spreading out the environment, since $\mathcal{J}_j(\widehat{S},\widehat{C})$ makes the relative distances among robots almost the same.

The main problem with the *standard* approach is that, when the environment is large, spreading out is not an acceptable solution, since coverage can not be assured. Instead some way needs to be found to focus *comms* on creating paths between S and H.

B. Dynamic Potential Fields

The key is to have the *comms* move to the parts of the environment where *sensors* are, not just anywhere. Since in the standard approach the balance between attractive and repulsive force, i.e., the potential field gradient, determines the spreading pattern, it is reasonable that if *comms* could cooperate they could *turn off* useless repulsive forces, which prevent them moving into critical position, and they could move in better locations. This approach, which dynamically changes the potential field will be called *dynamic*.

The algorithm works as follows: every robot sends a message to N randomly chosen $C^i \in \overline{C}$, in which they request help maintaining the network. The information is packed in a *token*, $\tau = \{x, c, \textit{sensor/comm}\}$. This could be accepted by the other robots or resent. If a *comm* accepts a token it adds the robot represented by that token to $\widehat{S}$ or $\widehat{C}$ and is thus influenced by that robot. If instead it chooses to pass the token on, it is not influenced at all by the robot represented by that token.

By a intelligent choice based on the information on the token, this *dynamic token* potential field approach can overcome the problems of the standard approach.

A simple example can show the token algorithm features and better performance over the standard approach. Let the situation be the one in Figure 3 (left), and let the *comms* be in equilibrium, i.e., at minima in the potential field. If the S_1 moves right and S_2 up, in the standard approach, C^1 tries to follow both sensors breaking the network; moreover C^2 and C^3 repulse each other and they can not help C^1. In the dynamic approach C^1 follows S^1 and informs C^2, which moves to help it maintain the network.

The choice of what tokens to send and, moreover, what to do with them, is made by defining a *policy*. Three different policies have been defined, namely C policy (connectivity), TC policy (threshold connectivity) and RC policy (resend connectivity). In following each policy will be described.

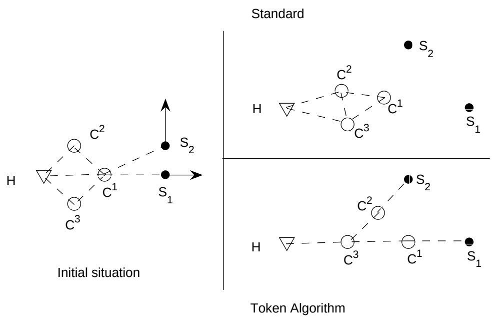


Fig. 3. Standard vs. Token algorithm behavior in an example. S_1 and S_2 are moving in opposite directions causing the network break in the former but not in the latter.

Each *comm* follows the algorithm shown in Algorithm 1, which is executed by each robot at each time step. For every time step, all the *comms* form a token $\tau = \{x, |\overline{C}|, \textit{comm}\}$ (Algorithm 1 line 1), then they keep sending and receiving tokens $i_{\max}$ times and they group them in T. $i_{\max}$ is fixed by the policy, if $i_{\max} = 1$ they simply delete the tokens they do not use. The determination of which tokens are important is made using a policy (Algorithm 1 line 12 - 16), which forms $\widehat{S}$, $\widehat{C}$. If $i_{\max} > 1$, the token τ to be resent. Finally, the robot computes the potential function and moves accordingly (Algorithm 1 line 18 - 19).

Algorithm 1 Token Algorithm

```
1: $\tau = \{x, |\overline{C}|, \textit{comm}\}$
2: $i_{\max} \leftarrow$ POLICY
3: for $i = 0$ to $i_{\max}$ do
4:     for $k = 0$ to $N$ do
5:         SEND($\tau$) $\rightarrow$ RANDOM($C^r \in \overline{C}$)
6:     end for
7:     $T = \{\emptyset\}$
8:     repeat
9:         $T_A$ =GETTOKEN $\leftarrow$ ($C^r \in \overline{C} \vee S^r \in \overline{S}$)
10:        $T = T \cup T_A$
11:    until (No more messages)
12:    if $i_{\max} > 1$ then
13:        ($\widehat{S}$, $\widehat{C}$, $\tau$) $\leftarrow$ POLICY($T$)
14:    else
15:        ($\widehat{S}$, $\widehat{C}$) $\leftarrow$ POLICY($T$)
16:    end if
17: end for
18: $\mathcal{J} = \mathcal{J}(\widehat{S}, \widehat{C})$
19: $x \leftarrow x + (\nabla_x \mathcal{J})\, \mathrm{d}x$
```

1) C policy: Robots with a low local connectivity are in the most critical positions of the network, thus they need more help. For that reason, C policy is based on local connectivity encouraging *comms* to keep the tokens of low connected robots and move toward them. It works as follows: let N_R be the number of received tokens, the *comm* sorts the received N_R tokens so that the first has the lowest c and the last the highest (Algorithm 2 - line 3), then it determines the subsets $\widehat{S}$ and $\widehat{C}$ using $M \leq N_R$ robots, which the first M tokens refer to (Algorithm 2 - lines 5 - 11). Then it deletes the remaining tokens since for C policy $i_{\max} = 1$.

Algorithm 2 C policy

```
1: $i_{\max} = 1$
2: $N_R \leftarrow |T|$
3: SORT($T$): $c(T_1) \leq c(T_{N_R})$
4: $\widehat{S} = \{\emptyset\}$, $\widehat{C} = \{\emptyset\}$
5: for $j = 1$ to $M \leq N_R$ do
6:     if $\textit{comm}(T_j) = 0$ then
7:         $\widehat{S} = \{\widehat{S}, S(T_j)\}$
8:     else
9:         $\widehat{C} = \{\widehat{C}, C(T_j)\}$
10:    end if
11: end for
```

2) TC policy: The policy is the same as C policy, but includes a criterion for determining whether a *comm* is *useful* or not. Since each robot can not know if it is on a minimal path, this criterion is based on the number of received token, N_R. If this is very high it means that the *comm* is very useful, since a lot of robots request its help; if there was only a robot connected to it, it would receive at least N tokens. On the other hand, if few tokens are received, the *comm* is in an useless position.

Therefore, when a comm receives fewer tokens than a specified threshold, it computes a different potential function where the repulsive part is neglected. This is done to allow that *comm* to move closer to other *comms* to reach, eventually, critical location in the network.

The threshold can be different for *sensors* and *comms*, in the sense that the repulsive part of the $\widehat{S}$ subset is neglected if the number of received tokens N_R is less than T_S, whereas the one of the $\widehat{C}$ subset is neglected if the number of tokens is less than T_C. Thus the potential function is

$$\begin{aligned}\widetilde{\mathcal{J}}_j(\widehat{S}, \widehat{C}, N_R) = \alpha \sum_{S_i \in \widehat{S}} -2\left(\frac{d_c f_s}{r_{C^j S_i}}\right)^6 + \\ (\text{tokens} \geq T_S)\left(\frac{d_c f_s}{r_{C^j S_i}}\right)^{12} + \\ \beta \sum_{C^q \in \widehat{C}} -2\left(\frac{d_c f_c}{r_{C^j C^q}}\right)^6 + (\text{tokens} \geq T_C)\left(\frac{d_c f_c}{r_{C^j C^q}}\right)^{12} \qquad (2)\end{aligned}$$

Algorithm 3 TC policy

```
1: C policy(from line 1 to line 11)
2: CHANGE $\mathcal{J}$ to $\widetilde{\mathcal{J}}(\widetilde{S}, \widetilde{C}, N_R)$
```

3) RC policy: The policy is the same of C policy, but $i_{\max} > 1$, thus, after *comm* determined $\widehat{S}$ and $\widehat{C}$ it resends some of the tokens it did not use. It keeps on sending and receiving tokens, until $i = i_{\max}$, when $i = i_{\max}$, it computes the potential field using the last determined $\widehat{S}$ and $\widehat{C}$. This allows to better determine $\widehat{S}$ and $\widehat{C}$, since it spreads the information of where the low local connectivity areas are. In fact, in low local connectivity areas *comms* could be not sufficient to satisfy all the help requests, while in high local connectivity ones, *comms* are less useful. By passing the unused tokens through the network, this can be avoided making less useful *comms* move where it is needed.

The choice of which tokens have to be resent is made by the connectivity value. In particular, each *comm* forms a new token, which carries not only its information, but also Q tokens, from $M+1$ to $M+Q$ (Algorithm 4 line 13 - 15).

Algorithm 4 RC policy

```
 1: $i_{\max}$ = number $> 1$
 2: $N_R \leftarrow |T|$
 3: SORT($T$): $c(T_1) \leq c(T_{N_R})$
 4: $\widehat{S} = \{\emptyset\}$, $\widehat{C} = \{\emptyset\}$
 5: for $j = 1$ to $M \leq N_R$ do
 6:     if comm($T_j$) $= 0$ then
 7:         $\widehat{S} = \{\widehat{S}, S(T_j)\}$
 8:     else
 9:         $\widehat{C} = \{\widehat{C}, C(T_j)\}$
10:     end if
11: end for
12: $\tau = \{x, |\overline{C}|, comm\}$
13: for $j = M+1$ to $Q \leq N_R - M$ do
14:     $\tau = \{\tau, x(T_j), c(T_j), comm/sensor(T_j)\}$
15: end for
```

C. *Proactive Positioning*

In some cases, the above approach can be insufficient to maintain the connectivity, because *sensors* might rapidly go out of range, e.g., when they turn around a corner. In a rapidly changing environment, or when it is important to assure the highest possible local connectivity for the *sensors*, *comms* should *proactively* choose where to move in order to preempt network breaks.

For that reason, a *comm* should follow only one robot at a time, thus from a *sensor* back to the hub there would be a communication chain formed by *comms*. A *relative grade*, r, is defined as follows: $\forall S_i, H \in S : r = 0, \forall C^i \in C : r = k+1$ if C^i follows a robot with $r = k$. Thus each *comm* decides to keep a token, and only one, on the basis of lower r. This means that every *comm* is trying to be useful for the network, since it follows a *comm* which is somehow linked to some *sensor*. Moreover, if there is more than one token with a low value of r it follows the one with the lower c, thus it moves in the most critical positions. This can be called proactive since each *sensor* is followed by a *comm* which has it as a only aim. Thus, this can have better performances in rapidly changing scenario.

The algorithm works as follows. First each robot sends a token $\tau = \{x, c, r\}$ to N_S or N_C sensed *comms*, where N_C is if the robot is a *comm*, N_S otherwise (Algorithm 5 line 1 - 4). Then, after it received tokens by the other, it chooses which of the received tokens has to be kept and form $\widehat{S}$ and $\widehat{C}$ (Algorithm 5 line 10 - 15). It deletes the remaining tokens, it computes the potential function and it moves (Algorithm 5 line 16 - 17).

Algorithm 5 Proactive algorithm

```
 1: $\tau = \{x, c, r\}$
 2: for $k = 0$ to $N_C$ do
 3:     SEND($\tau$) $\rightarrow$ RANDOM($C^r \in \overline{C}$)
 4: end for
 5: $T = \{\emptyset\}$
 6: repeat
 7:     $T_A$ =GETTOKEN $\leftarrow$ ($C^r \in \overline{C} \vee S^r \in \overline{S}$)
 8:     $T = \{T, T_A\}$
 9: until (No more messages)
10: SORT($T$): first, lower $r$, then, lower $c$
11: if comm($T_1$) $= 0$ then
12:     $\widehat{S} = \{S(T_1)\}$, $\widehat{C} = \{\emptyset\}$
13: else
14:     $\widehat{S} = \{\emptyset\}$, $\widehat{C} = \{C(T_1)\}$
15: end if
16: $\mathcal{J} = \mathcal{J}(\widehat{S}, \widehat{C})$
17: $x \leftarrow x + (\nabla_x \mathcal{J})\, \mathrm{d}x$
```

IV. RESULTS

In this section the results for the selected environments will be presented. First, experiments were performed in the *Wall* environment to test the basic functionalities of the algorithms. Second, the *Office* environment is used to test coordination and fault-tolerance. Third, the *Open Space* environment is used to test both behavior with unknown initial conditions and high failure rates. Finally, the *Corridor* environment, which is fast changing, is used to test the Proactive algorithm. The baseline will be the standard approach.

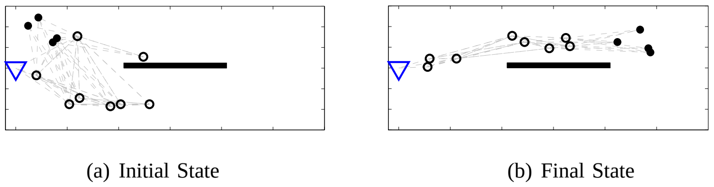
A. Wall

This apparently simple wall scenario tests the basic functionality of the algorithms. If the *comms* do not move north of the wall, connectivity will be lost. Thus, spreading out is an inadequate strategy.

The initial and a final state are shown in Figure 4, notice the *sensors*, black dots, move from west to east.

In Figures 5 and 6 the obtained average results for 200 simulations are presented. The parameters for the experiments were: for the scenario: $|S| = 6$, $d_c = 10$, $v = 0.016$, for the potential function: $f_s = 1/2$, $f_c = 1/2$, $\alpha = 100\,|C|/|S|$, $\beta = 1$; for the policies: $N = 2$, $(M, Q) = ([N_R/2], 0)$ for C and TC policies, $N = 2$, $(M, Q) = (1, 1)$ for RC policy, $T_S = 4$, $T_C = 3$, $i_{\max} = 3$. In the graphs B means Baseline, C the C policy and so on. Bars are the value of average global connectivity of average efficiency. Black bars on the top of the bars are the standard deviations. Lines in the bars are the final value of $\mathcal{K}$.

Both $< \mathcal{K} >$ and $< \mathcal{E} >$ are significantly higher using the token algorithm than in the baseline. Moreover, note that in



(a) Initial State (b) Final State

Fig. 4. Simple wall simulations: black dots are *sensors*, white dots the *comms*, dashed lines communication links, the triangle the hub. The box ticks distance is 5.

the $|C| = 8$, case the baseline can not assure a final global connectivity greater than one, while the other algorithms can. TC policy performs best here, which was expected since it makes *comms* move in the north side of the wall neglecting the repulsive force of *comms* already there. This was less decisive for 14 *comms* because with an increasing number of *comms* resending is more important since it allows a better understanding of critical locations.

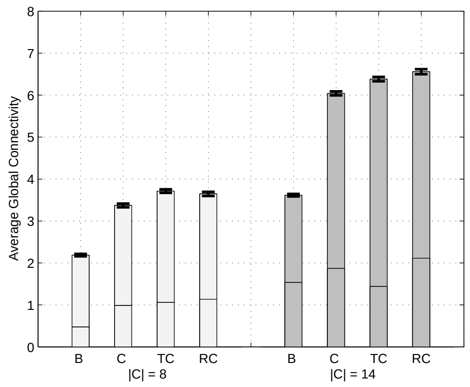


Fig. 5. $< \mathcal{K} >$ for different $|C|$: B means Baseline, C C policy and so on. Black bars on the top of the bars are the average standard deviations, whereas black lines in the bars are the final value of $\mathcal{K}$.

B. Office

The Office scenario spreads the *sensors* out more, requiring more coordination between the *comms*. The initial and a final state are shown in Figure 7, notice that *sensors* start in the upper part of the environment and finish at different locations.

Using this scenario, network reliability and fault-tolerance were tested: to do this, after 60% of time had elapsed, 3 randomly chosen *comms* were disabled.

In Figures 8 and 9 the results for 200 simulation runs are presented. The parameters for the scenario were: $|S| = 5$, $d_c = 12$, $v = 0.005$, for the potential function: $f_s = 1/2$, $f_c = 1/2$, $\alpha = 100\,|C|/|S|$, $\beta = 1$, for the policies: $N = 2$, $(M, Q) = ([N_R/2], 0)$ for C and TC policies, $N = 2$, $(M, Q) = (1, 1)$ for RC policy, $T_S = 2$, $T_C = 1$, $i_{\max} = 3$. $|C| = 15$ is the no-failure case, while $|C| = 12$* means 3 *comms* were disable after 60% of the time.

RC policy performs best in both no-failure and failure cases. This is due to tokens needing to be accepted by robots outside their immediate communication range, necessitating resends.

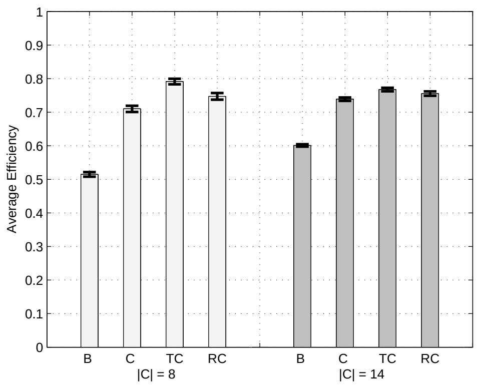
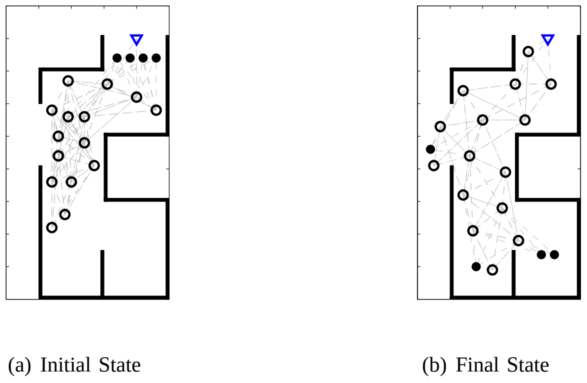


Fig. 6. $< \mathcal{E} >$ for different $|C|$: B means Baseline, C C policy and so on. Black bars on the top of the bars are the average standard deviations.



(a) Initial State (b) Final State

Fig. 7. Office simulations: black dots are *sensors*, white dots the *comms*, dashed lines communication links, the triangle the hub. The box ticks distance is 5.

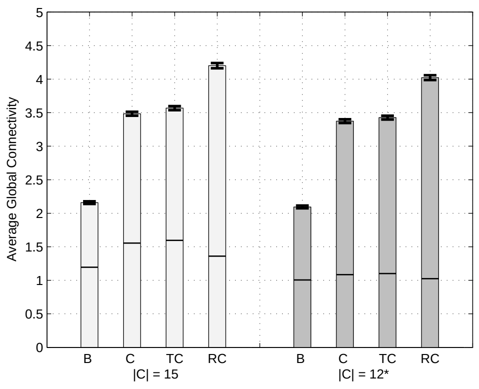


Fig. 8. $< \mathcal{K} >$: B means Baseline, C C policy and so on. Black bars on the top of the bars are the average standard deviations, whereas black lines in the bars are the final value of $\mathcal{K}$. $|C|$ = 15 is the no-failure case, while $|C|$ = 12* means 3 *comms* were disable after 60% of the time.

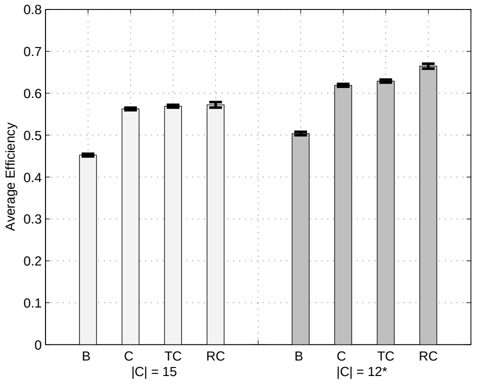
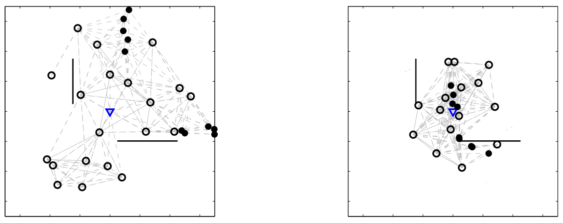


Fig. 9. $< \mathcal{E} >$: B means Baseline, C C policy and so on. Black bars on the top of the bars are the average standard deviations.$|C|$ = 15 is the no-failure case, while $|C|$ = 12* means 3 *comms* were disable after 60% of the time

C. Open space

In the *Open Space* environment the algorithms performance to random initial positioning of *sensors* was tested. The scenario consists of randomly spread *comms* and two group of *sensors* which can enter in the environment from randomly chosen locations and they move to the center where the hub is. (See Figure 10). In this environment, *comms* were disabled at a rate, k_R.



(a) Initial State (b) Final State

Fig. 10. Open air simulations: black dots are *sensors*, white dots the *comms*, communication links are not shown, the triangle is the hub. The box ticks distance is 20.

In Figures 11 and 12 the obtained results for 400 runs are presented. The parameters were: for the scenario: $|S| = 11$, $|C| = 20$ $d_c = 50$, $v = 0.007$, for the potential function: $f_s = 1/2$, $f_c = 1/2$, $\alpha = 100\,|C|/|S|$, $\beta = 1$, for the policies: $N = 2$, $(M, Q) = ([N_R/2], 0)$ for C policy, $(M, Q) = (1, 1)$ for RC policy, $i_{\max} = 10$.

RC policy performs best since, it is crucial to inform *comms* out of the immediate communication range of the *sensors*. In particular RC policy with $N = 2$ is the best because it allows different *comms* follow to different robots, while $N = 10$ tends to make them follow the same robot.

D. Corridor environment and Proactive algorithm

The *Corridor* scenario was used to test the performance of the proactive approach in comparison to the basic algorithms.

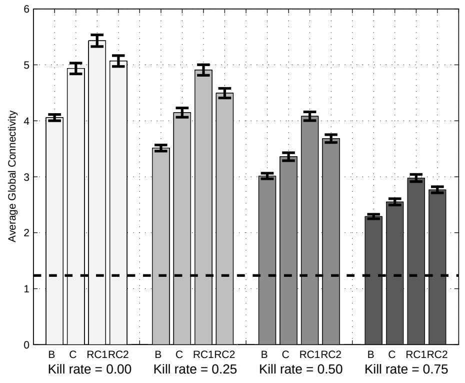


Fig. 11. $< \mathcal{K} >$: B means Baseline, C C policy, RC1 RC policy with $N = 2$, RC2 RC policy with $N = 10$. Black bars on the top of the bars are the average standard deviations. The dashed black line is the initial $\mathcal{K}$.

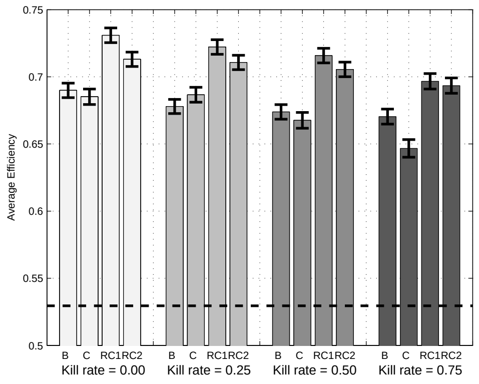


Fig. 12. $< \mathcal{E} >$: B means Baseline, C C policy, RC1 RC policy with $N = 2$, RC2 RC policy with $N = 10$. Black bars on the top of the bars are the average standard deviations. The dashed black line is the initial $\mathcal{E}$.

To ensure this environment is fast changing, $v = 0.022$. The scenario initial and a final states are depicted in Figure 13.

In Figures 14 and 15 the obtained results for 400 simulation runs are presented. The parameters were: for the scenario: $|S| = 3$, $|C| = 12$ $d_c = 20$, $v = 0.022$, for the potential function: $f_s = 1/2$, $f_c = 1/2$, $\alpha = 100\,|C|/|S|$, $\beta = 1$, for the policies: $N = 2$, $(M, Q) = ([N_R/2], 0)$ for C policy, $(M, Q) = (1, 1)$ for RC policy, $i_{\max} = 3$, for Proactive algorithm: $N_S = 5$, $N_C = 10$. In the graphs, connectivity and efficiency behavior during time for different algorithms is depicted. The lines thickness represents the standard deviation.

The RC policy with $N = 10$ and Proactive algorithm lead to best results. The former since it can inform more *comms* by resending tokens; $N = 10$ is better than $N = 2$ because the environment is fast changing and there are few *sensors* to follow, thus it is important that the information of *sensors* location reaches more *comms*. The Proactive algorithm has a smoother behavior with no deep local minima, this was expected since it makes *comms* try to be useful every time. Finally Proactive involves less communications than RC policy with $N = 10$, because tokens do not have to be resent.

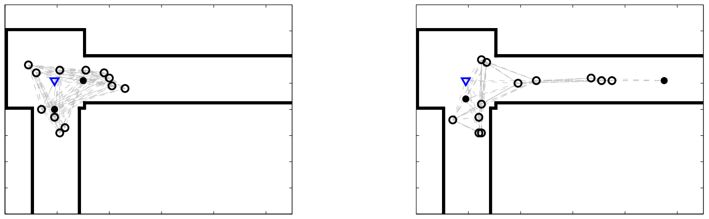


Fig. 13. Corridor simulations: black dots are *sensors*, white dots the *comms*, communication links are not shown, the triangle is the hub. The box ticks distance is 10.

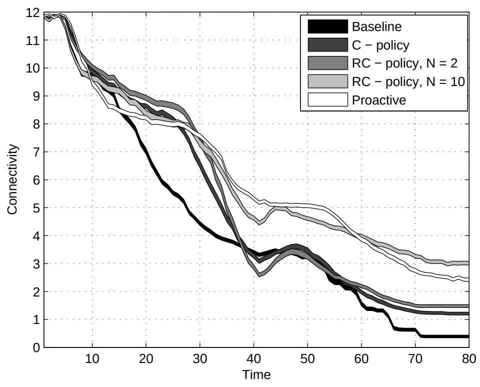


Fig. 14. $\mathcal{K}$ vs. time for different algorithms and policies. The lines thickness represents the standard deviation.

V. Related Work

Standard potential fields have been investigated both for coverage analysis in simple and very complex environments [8]. When they have been applied to build an ad hoc network for communication purposes, this has been done or in simple environments without walls [10], or with the cooperation of sensors [11], or with a priori knowledge of the environment [12]. Ad hoc networks and robot positioning to obtain connected and k - connected graphs have been studied in simple environments without walls [5], [6], [7], and they have currently several applications also in aerospace research [3], [13].

VI. Conclusions and Future Work

This paper presented a novel approach to maintaining an ad hoc communications network for sensor robots exploring an unknown environment. The key to the approach is dynamically and cooperatively changing potential fields that govern robots movement. Different policies for determining how to create the potential field were used and the specifics of the environment determined which policy performed best. However, in all cases, dynamic potential fields out-performed a standard potential field approach.

While these results are promising, significantly more work needs to be done before this approach is ready for real-world

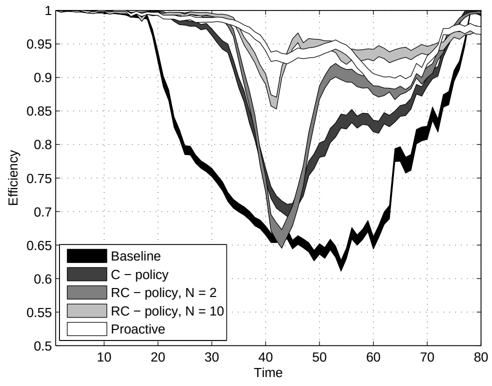


Fig. 15. $\mathcal{E}$ vs. time for different algorithms and policies. The lines thickness represents the standard deviation.

deployment. In the immediate future, we will be extending the approach to larger environments with more complex signal propagation characteristics, e.g., signals may be attenuated or multi-path effects observed. In the medium term, the approach will be implemented on a set of physical robots to better understand the real-world issues that must be addressed.

VII. Acknowledgment

This research was supported in part by AFOSR FA9550-07-1-0039, AFOSR FA9620-01-1-0542, and by L3-Communications grant number 4500257512 to Carnegie Mellon University.

References

[1] E. Martinson and R. C. Arkin, “Learning to role-switch in multi-robot systems,” in *Proceedings of the 2003 IEEE International Conference on Robotics and Automation*, Taipei, Taiwan, September 14 - 19, 2003.

[2] R. Simmons et al., “Coordinated deployment of multiple, heterogeneous robots,” in *Proceedings of the 2000 IEEE/RSJ International Conference on Intelligent Robots and Systems*, Takamatsu, Japan, 2000.

[3] A. Krishnamurthy and R. Preis, “Satellite formation, a mobile sensor network in space,” in *Proceedings of IPDPS’05*, Denver, USA, 2005.

[4] D. Wettergreen; D. Bapna, M. Maimone and G. Thomas, “Developing Nomad for robotic exploration of the Atacama desert,” *Robotics and Autonomous Systems*, vol. 26, no. 2, pp. 127 - 148, 1999.

[5] J. L. Bredin, E. D. Demaine, M. T. Hajiaghayi and D. Rus, “Deploying sensor networks with guaranteed capacity and fault tolerance,” in *MobiHoc’05*, Urbana-Champaign, Illinois, USA, May 25-27, 2005.

[6] C. Savarese J. Rabaey and K. Langendoen, “Robust positioning algorithms for distributed ad-hoc wireless sensor networks,” in *USENIX Technical Annual Conference*, Monterey, CA, USA, June, 2002.

[7] S. Meguerdichian, F. Koushanfar, M. Potkonjak and M. B. Srivastava, “Coverage problems in wireless ad-hoc sensor networks,” in *Proceedings IEEE INFOCOM 2001*, Anchorage, Alaska, USA, April 22-26, 2001.

[8] A. Howard, M. J. Matarić and G. S. Sukhatme, “Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem,” in *Proceedings of DARS’02*, Fukuoka, 2002.

[9] O. Shehory, S. Kraus and O. Yadgar, “Emergent cooperative goal-satisfaction in large-scale automated-agent systems,” *AI*, 1999.

[10] L. E. Parker, “Distributed algorithms for multi-robot observation of multiple moving targets,” *Autonomous Robots*, vol. 12, 2002.

[11] R. Martijn and B. Andreas, “Multi-robot exploration under the constraints of wireless networking,” *Control Engineering Practice*, 2007.

[12] R. Murrieta-Cid, H. H. Gonzglez-Bafiost and B. Tovar, “A reactive motion planner tomaintain visibility of unpredictable targets,” in *Proceedings of the lnter’l Conference on Robotics and Automation*, 2002

[13] K. Chandrashekar M. R. Dekhordi and J. S. Baras, “Providing full connectivity in large ad-hoc networks by dynamic placement of aerial platforms,” in *MILCOM’04*, Monterey, CA, USA, 2004.