Connectivity of Autonomous Agents Using Ad-Hoc Mobile Router Networks

Emi Mathews and Ciby Mathew

Heinz Nixdorf Institute
University of Paderborn
Paderborn, Germany
{emi,ciby}@hni.upb.de
http://wwwhni.uni-paderborn.de/eps

Abstract. Maintaining connectivity among a group of autonomous agents exploring an area is very important, as it promotes cooperation between the agents and also helps message exchanges which are very critical for their mission. Creating an underlying Ad-hoc Mobile Router Network (AMRoNet) using simple robotic routers is an approach that facilitates communication between the agents without restricting their movements. We address the following question in our paper: How to create an AMRoNet with local information and with minimum number of routers? We propose an agent-assisted router deployment algorithm for creating AMRoNet which is a localized, distributed router placement algorithm. The algorithm has a greedy deployment strategy for releasing new routers effectively into the area and a triangular deployment strategy to connect different connected components created by the agents exploring from different base stations. Empirical analysis shows that the agent-assisted router deployment algorithm is one of the best localized approaches to create an AMRoNet.

Keywords: Mobile Routers, Ad-hoc Network, Robotic network, Connectivity, Localized deployment.

1 Introduction

We envision a scenario with several *agents* which are humans or powerful robots moving autonomously on a terrain represented by a plane. These autonomous agents begin their exploration from one or more stationary base camp(s). We are looking for local and distributed strategies for maintaining the *connectivity* of the agents with the base station(s) and the other agents, as it promotes cooperation between the agents and also helps message exchanges which are very critical for their mission. These strategies must not restrict agent movements for the sake of maintaining connectivity.

Scenarios such as urban search and rescue and exploration of an unknown terrain are good examples, where we often have several exploring agents and one or more base station(s). In urban search and rescue scenarios, due to the aftermath

N. Meghanathan et al. (Eds.): CCSIT 2012, Part I, LNICST 84, pp. 225–237, 2012.

of natural or manmade disasters such as earthquakes, tsunamis, hurricanes, wars or explosions, the fixed communication infrastructure that could support communication between the agents are often destroyed. In other scenarios such as exploration of unknown terrains, e.g. subterranea or remote planets, no such infrastructure to support communication exist. A line of sight communication between the agents is not possible in such complex scenarios as the distance between the agents are often very large due to the large area to be explored. The presence of obstacles makes it difficult even at shorter inter-agent distance.

A stable and high bandwidth communication is feasible if we employ a multi-hop ad-hoc networking strategy for the agents. However, in such scenarios the number of agents is often very limited. Hence the agents themselves could not form a connected network always. Moreover, if they try to keep the network connected, it would restrict their movements.

We propose an alternate solution to maintain connectivity of the agents, i.e. deploy cheap router nodes that are mobile and create a network that acts as an infrastructure to support the communication of the agents. Thus we have a two tier network, with the agents and base stations lying at the upper layer and the routers deployed at the lower layer. The lower layer created to facilitate the communication between upper layer members is called Ad-hoc Mobile Router Network (AMRoNet). This network, in addition to supporting upper layer members' communication, provides various services to the agents, such as location information, topological maps and shortest path to base stations, and can also assist the search and rescue operation of the agents. The main advantage of this network is that the routers could relocate and maintain the connectivity in case of failures which are very common in scenarios described above.

In this paper, we address the following question: How to create an AMRoNet with local rules and with minimum number of routers? The remainder of this paper is organized as follows: Section 2 introduces the scenario and notations used in this paper. Related approaches known from the literature are discussed in Section 3. In Section 4 we present our new algorithm for creating AMRoNet with local information and discuss about the optimal solution. Next, in Section 5 we present a simulation based performance evaluation and analysis of the proposed algorithm. Finally, Section 6 summarizes the main results of this work and provides an outlook on possible future research.

2 Preliminaries

We have a two tier network, with the agents and base stations forming the upper layer. The environment where the agents explore is a 2-D area denoted as A and has n base stations. There are N_a agents which are humans or robots capable of performing tasks such as urban search and rescue. As the focus of this paper is mainly on the AMRoNet, we do not specify the requirements of the agents and the base stations, which vary according to the scenario considered. The only assumption we make is that they have a wireless devices to support communication.

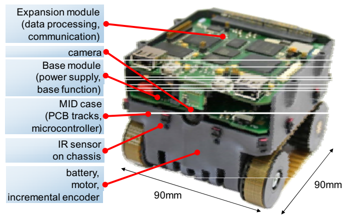


Fig. 1. Bebot mini-robot

The lower layer forming the AMRoNet consists of total N_r routers. The routers denoted by R, are very simple robots compared to the agents with limited sensing capabilities with which they avoid obstacles and perform local navigation. Routers are equipped with wireless transceivers for communication.

We assume the unit disk graph wireless model [2] for communication, where each node (agent, router or base station) can communicate with others located within a circle of radius r_c. We also assume that the communication area of one node πr_c^2 is much less than A. Hence, the agents have to send packets over several routers to reach a particular destination (other agent or base station).

We are interested in maintaining connectivity of the agents with minimum number of routers. Hence our objective is to find a strategy to create AMRoNet that provides *optimal coverage* with respect to the total *communication area*.

Mini-robots such as Bebots [7], shown in figure 1 are suitable candidates for routers. These robots are equipped with a camera with which they can assist agents in search and rescue operations. They have an infrared sensor ring for obstacle avoidance and wifi, zigbee and bluetooth modules for communication.

3 Related Work

Existing approaches to create AMRoNet are mostly based on *mobile routers making a chain*. In [3,10] the authors present different strategies such as Manhattan-Hopper, Hopper, Chase explorer and Go-to-The-Middle for maintaining the connectivity of an explorer with a base station. In [12] depending on whether the knowledge of the agent's trajectory is available or not, the trajectories for the routers are estimated.

The multi-robot spreading algorithms [6, 8, 14], though not meant for maintaining the connectivity of agents, could also be used for AMRoNet robots. In these algorithms, mobile robots spread out based on local rules. If the routers also move out of the base stations pro-actively and spread in the environment, using these algorithms they can form the AMRoNet for agents' communication.

Maintaining the connectivity of a group of robots while rendezvous, flocking, formation control etc. by controlling their motion pattern has been addressed in [4], [13], [11]. However, the agents we consider move independently and cannot be controlled for maintaining the connectivity.

4 Ad-Hoc Mobile Robotic Networks

Existing approaches to create AMRoNet presented in Section 3, maintain connectivity of the agents, if the routers move as fast as the agents. However, this assumption is not valid in our case as the routers used to create AMRoNet are very simple robots and their speed is usually very small compared to the speed of the agents. The *chain based* approach needs routers that can move faster than the agents [3, 10] and [12] needs twice the speed of the agent, to keep the chain connected. Existing chain based approaches cannot support connectivity of multiple exploring agents. Hence they are not useful in our scenario. The proactive spreading using *multi-robot spreading* algorithm also needs router moving as fast as the agents to keep them connected. Using simple routers that are slower than the agents, the multi-robot spreading algorithms based approaches work only if the deployment phase is finished prior to the exploration of the agents. However, in scenarios such as urban search and rescue, such pre-deployment is not feasible.

We propose a new approach called *agent-assisted router deployment* for AMRoNet creation which doesn't need any fast moving routers or pre-deployment phase. In agent-assisted router deployment, the agent carries the routers during the exploration. When they are at the verge of disconnection, they release a new router into the area. Routers move locally to maximize coverage. Such an approach is feasible, as our robots are very small [7] and the agents can carry several robots during their exploration.

4.1 Agent-Assisted Router Deployment Algorithm

Let the N_a agents begin their exploration from n base stations. Each base station has a unique id and one reference node which acts as a base station server for all communication. The base station i, for all $i \leq n$, is denoted as BS_i and its reference node as R_i. We set the *status* of R_i and the agents moving out of BS_i to i. Routers are denoted as R_{ij} and agents as A_{ij}, where i is their status and j indicates their unique id. The agents explore the area based on their own navigational algorithm. Figure 4.1 shows a schematic representation with two base stations and two agents (one agent per base station) exploring an open area.

Initially an agent A_{ij} has wireless links to R_i and other agents A_{ik} for any $k \leq N_a$. As the link between A_{ij} and R_i is initialized, A_{ij} asks R_i about its

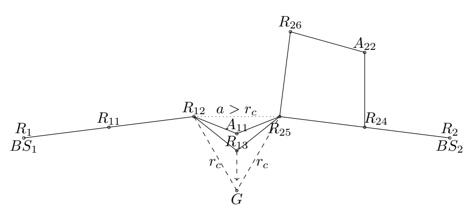


Fig. 2. Schematic representation of agent-assisted router deployment in an open region

position and stores this information. If A_{ij} is about to lose its connection to R_i, it places a new router with its status set to i and position set to A_{ij}'s current position. The new router R_{ik}, for any $k \leq N_r$, is placed very close to the current location of A_{ij} in the direction towards R_i. Agents use the position information of R_i for estimating the direction. This ensures that R_{ik} released is always connected to R_i. R_{ik} becomes the new reference for A_{ij} and for all other agents within R_{ik}'s communication rage. During the navigation, A_{ij} may move inside the range of a router R_{pq} for any $p \leq n$ and $q \leq N_r$ that has already been deployed. In this case R_{pq} becomes A_{ij}'s current reference. A_{ij} asks R_{pq} for its status and repeats the placement steps when it is about to lose its connection to R_{pq}. If an agent has wireless links to many reference robots, any one of them acts as the agent's current reference. The agent releases a new router only when it loses connection to the last reference node in its communication range. We call this placement strategy as *greedy deployment*.

The greedy agent-assisted router deployment builds a graph G with the nodes at the base stations and with routers released during agents' exploration as its vertices. Agents exploring from one base station form a connected component, denoted as CC, of G. However, such CCs created from multiple base stations are not connected. When an agent A_{ij} enters into the range of R_{pq} for $i \neq p$ from the current reference R_{ik}, CC_i and CC_p are temporarily connected. During the navigation, if A_{ij} loses it connection to R_{ik} but still has connection to R_{pq}, A_{ij} does not place another router, as it has R_{pq} as its current reference. In this case, A_{ij} loses connection to its original base station BS_i and CC_i and CC_p get disconnected again.

To solve the disconnection problem, in such situations we adopt another deployment strategy called *triangular deployment*. In triangular deployment, when an agent enters into the range of R_{pq} for $i \neq p$ from its current reference R_{ik}, it releases a new router with its status set to i and it moves to a point that keeps R_{ij} and R_{pq} connected and maximizes the local coverage. The goal point of the new router for maximizing the local coverage can be calculated as follows: If a is the distance between R_{iref} and $R_{j_{ref}}$, the goal point lies at a distance $d = \sqrt{r_c^2 - (\frac{a}{2})^2}$ from the midpoint of the line joining R_{ij} and R_{pq} on the same side of the agent as shown in Figure 4.1. During the *goto goal* behavior, if the new router encounters an obstacle that cannot be avoided in few steps, it stops navigating to the goal location, as the obstacle could be too large to overcome without disconnecting R_{ij} and R_{pq}.

To optimize the number of robots used during the triangular deployment, we propose the following strategy. The agent A_{ij} performs the triangular placement only when it enters into the range of the first R_{pq} with $i \neq p$ and connects CC_i and CC_p. A_{ij} then disables triangular placement to all R_{pk}, for any $k \leq N_r$. If there are multiple agents and multiple base stations we propose two strategies for the triangular deployment. The first one needs global communication and the second one needs only local communication.

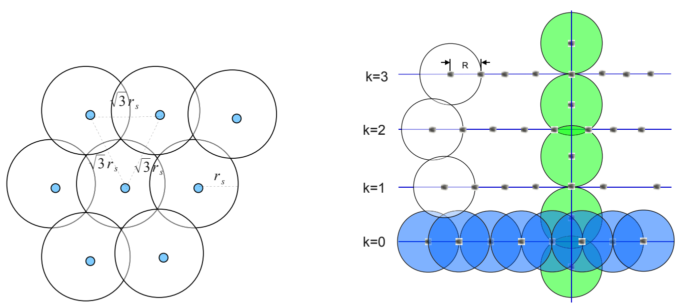
In the first strategy, when an agent A_{ij} entering into the range of R_{pq} with $i \neq p$, checks with R_{pq} if any other agent has already made CC_i and CC_p

connected. If not, it performs a triangular deployment to connect them and sends a message to all other agents and references that are connected to it either directly or by multi-hop networking. All these agents and references update their information about the connected components in G.

In the second localized strategy, the router released during the triangular deployment set the references R_{ij}, R_{pq} and itself as *disabled* for further triangular deployment. When an agent entering into the range of R_{pq} from the current reference R_{ij} with $i \neq p$, it checks if both R_{ij} and R_{pq} has already been disabled from triangular deployment. This ensures that CC_i and CC_p always get connected and prevents redundant deployment at the locations of triangular placements.

4.2 Optimal Deployment

We can find the optimal router location of an AMRoNet from the static optimal placement strategies used in the area coverage problems. The objective of these problems is to place minimum number of nodes in an environment such that, every point is optimally covered. If we look at the *optimal coverage* with respect to the total sensing area, the robots could form a triangular grid as shown in Figure 3(a). When the inter-robot distance $d = \sqrt{3}.r_s$, where r_s is the sensing radius, 100% coverage is attained with minimum number of robots. This approach creates a connected network if $r_c/r_s \geq \sqrt{3}$.



(a) Optimal coverage (b) R-strip tile in 2D

Fig. 3. Coverage and Connectivity

We are interested in maintaining the connectivity of the agents with minimum number of routers. Hence the optimal coverage we refer to is the coverage with respect to the total communication area. A triangular grid with the inter-robot distance $d = \sqrt{3}.r_c$ cannot provide 100% *communication area coverage*, as robots cannot communicate when $d > r_c$. So a coverage and connectivity $(C - C)$ constraint arises and our objective is to maximize the communication area coverage with connectivity.

If we create a triangular grid with reduced inter node distance $d = r_c$, it is not optimal according to the $C - C$ constraint. What is optimal in 1-D, is an r-strip shown at the bottom row of Figure 3(b), where $d = r_c$. In 2-D, the lower

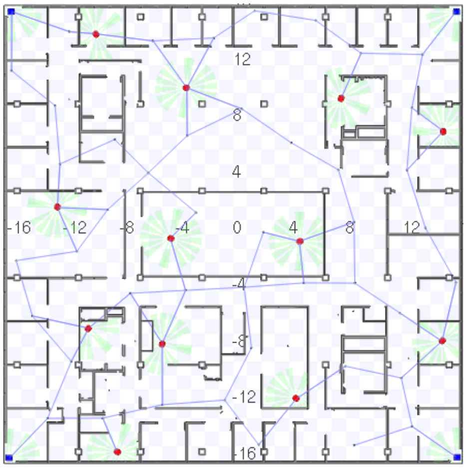


Fig. 4. An example scenario with 12 agents and 4 base stations

bound on node density for optimal $C - C$ is $d_{OPT} \geq \frac{0.522}{r^2}$ [9]. The optimal solution that achieves communication coverage with 1-connectivity in 2-D is the r-strip tile shown in the figure 3(b) [1]. It has a spatial density $d_{STR} = \frac{0.536}{r^2}$ [9]. The r-strip tile in 2-D is created as follows: for every integer k place a strip horizontally such that there is one node positioned at $(0, k(\frac{\sqrt{3}}{2} + 1)r_c)$ for every even k, and one node positioned at $(\frac{r_c}{2}, k(\frac{\sqrt{3}}{2} + 1)r_c)$ for every odd k. Finally place some nodes vertically in the following way. For every odd k, place two nodes at $(0, k(\frac{\sqrt{3}}{2} + 1)r_c \pm \frac{\sqrt{3}}{2})$. The purpose of this vertical strip is to connect the horizontal strips and thus ensure connectivity between all nodes.

A more commonly used regular patterns are hexagonal grid which has $d_{HEX} = \frac{0.77}{r^2}$ and square grid which has $d_{SQR} = \frac{1}{r^2}$ [9]. In triangular grids, the number of nodes in a $D \times D$ square area is $\frac{2D}{\sqrt{3}r} . \frac{D}{r} \approx 1.155 \frac{D^2}{r^2}$ and hence the density $d_{TRI} = \frac{1.155}{r^2}$.

5 Performance Evaluation and Analysis

5.1 Simulation Setup

We evaluate the proposed agent-assisted router deployment algorithm using a simulation based empirical analysis. We use the Player robotic interface and Stage 2D simulator for our experiments [5]. The area considered is a $32m \times 32m$ square area which maps the floor plan of our institute as shown in Figure 4. In our simulations, the agents are modeled as Pioneer2dx robots, routers as Bebot robots and base stations as Amigobot robots. All robots are equipped with WiFi modules for communication. The base station robots are located at the corner of the simulation environment and are immobile. The scenario shown in Figure 4 has 4 base stations and 12 agents (3 per base station). The agents start their exploration from a point very close to the base station robots and are initially connected to them. We have chosen a random exploration strategy for the agents. They detect obstacles using their sonar sensors which have maximum range of 2 m and avoid them using the *obstacle avoidance* behavior implemented. The release of a new router by the agent is implemented by moving a router located outside the simulation environment to its placement point by the simulator. Routers released during the triangular placement use the *goto* behavior to navigate towards the goal points. They avoid collisions using their IR sensors which have maximum range of 14 cm.

5.2 Performance of Agent-Assisted Router Deployment Algorithm

To analyze the performance of the agent-assisted router deployment algorithm, we vary parameters such as r_c and N_a. Figure 5 shows the result of the algorithm, when r_c is varied from 4 to 10 in a square area of size $32m \times 32m$. The graph plot with label ARD shows the average number of routers (including the reference robot in the base station) deployed to cover the entire region, when all agents begin their exploration from one base station. Here, N_a is varied from 1 to 4. For each N_a, the simulation is repeated 5 times and the agents are assigned different start locations. So the graph plot with label ARD given in Figure 5 is the average of 20 simulations with confidence interval at 95%.

To compare the performance of the algorithm, we calculate the number of robots required, by the static placement strategies of the commonly used regular patterns such as r-strip tile, hexagonal grid, square grid and triangular grid. The estimated number of robots required to cover the area can be calculated using the spatial density of the patterns, i.e $d_{STR} = \frac{0.536}{r^2}$, $d_{HEX} = \frac{0.77}{r^2}$, $d_{SQR} = \frac{1}{r^2}$ and $d_{TRI} = \frac{1.155}{r^2}$. Since the area is bounded, the minimum number of robots actually required to cover the entire region is often higher than the estimated values. This is clearly visible in the example figures Figure 6(a) and Figure 6(b), where the estimated number of robots needed for the r-strip $RSTR_{est}$ is 35 and

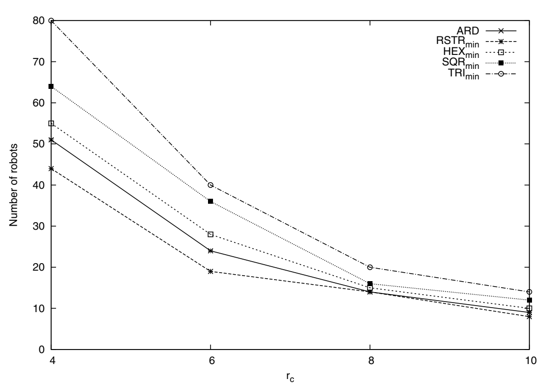
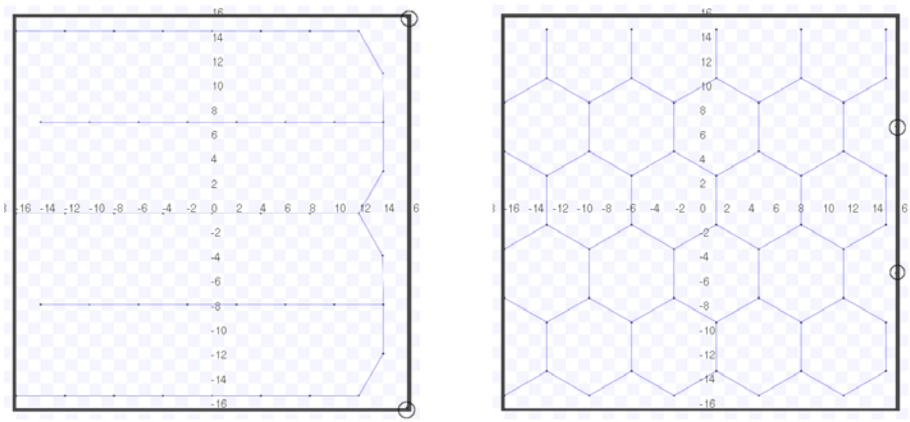


Fig. 5. Comparison of performance of router deployment algorithms

the hexagonal grid HEX_{est} is 50, but the minimum required number for r-strip tile $RSTR_{min}$ is 44 and the hexagonal grid HEX_{min} is 55. The figures also show that there are still uncovered areas, e.g. the location of the robots highlighted with circles. We cannot place additional routers to cover these areas, as they would be placed outside the specified area according to the regular placement pattern.



(a) R-strip tile based topology (b) Hexagonal-grid based topology

Fig. 6. Static placement of regular pattern

Figure 5 also shows the plot of the minimum required values for r-strip tile $RSTR_{min}$, hexagonal grid HEX_{min}, square grid SQR_{min} and triangular grid TRI_{min} in the specified square area, when r_c is varied from 4 to 10. It shows that the proposed algorithm is better than TRI_{min}, SQR_{min}, and HEX_{min} placement strategies. The number of robots needed by the proposed algorithm is close to the $RSTR_{min}$ values which are the actual optimal values.

5.3 Effect of Number of Agents and Base Stations

To analyze the effect of number of agents and base stations on the agent-assisted router deployment algorithm, we now vary number of agents per base station N_{apbs} and the number of base stations n, for a fixed r_c. Figure 7 shows the average number of robots (including the base station robots) needed to cover the square area of size $32m \times 32m$ for $N_{apbs} = 1, 2$ and 3, when n is varied from 1 to 4.

Increasing the number of agents without increasing n do not affect the performance, as the deployments performed by the agents are based on the local rules which are in turn based only on losing or establishing connection with other routers and not with other agents. Hence the number of routers deployed is independent of the number of agents. The data points for a particular n shown in Figure 7 with different N_{apbs} confirm this.

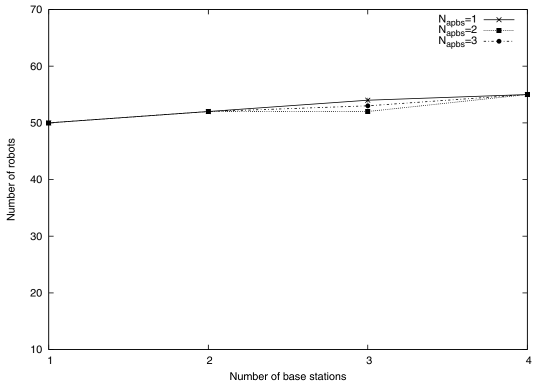


Fig. 7. Effect of number of agents and base stations on the performance

Increasing the number of base stations may result in more triangular deployments. The total area covered by three robots in a triangular deployment is usually lesser than the total area covered by them in an optimal deployment. The largest overlap in a triangular deployment occurs when two references are separated by a distance slightly greater than r_c. However, such deployments do not increase the number of routers considerably. Even the greedy deployment may produce similar less optimal overlapping regions, e.g. when an agent connected to two references move out of the communication radius of both references simultaneously.

Figure 8 shows a scenario where three routers are released during the triangular deployment. Actually at most 2 routers are needed to make the four chains connected. Such redundant deployment increases with the number of base stations. We could add more local rules to make the increase bounded, but this is not actually needed as the agents move independently (in our experiments, they move randomly) and the structures similar to the one shown in Figure 8 occur very rarely. The graph plots for $N_{apbs} = 1$ and $N_{apbs} = 2$ depicted in Figure 7 also show that the total number deployed is more or less the same for different base station counts.

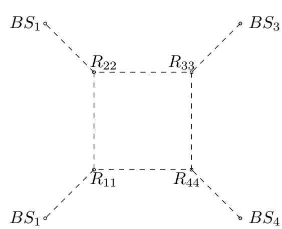


Fig. 8. Redundant router deployment during local triangular deployment

5.4 Localized r-Strip Tile Creation Algorithm

From the evaluation of the proposed agent-assisted router deployment algorithm, we see that it performs better than all other regular pattern based static placements except the r-strip tile in 2-D. Let us now look at a localized agent-assisted r-strip tile creation algorithm that does not restrict agent movements or causes disconnections. A straight forward extension of the agent-assisted router deployment algorithm for r-strip creation is: Agents release routers as per the greedy deployment strategy and the routers move to the goal points that create r-strip tiles locally.

During this localized r-strip creation the following problems arises: The routers released move to their goal point very slowly compared to the agent speed. If the agents use these moving routers as their references, to prevent disconnections they may have to release new routers before their current references reach their goal points. Hence more routers than the static optimal r-strip tiles are needed for this localized solution. Another problem is the presence of obstacles which prevents the routers from reaching the ideal optimal goal point. A third problem occurs when we have multiple base stations. The pattern created from one base station may not be aligned with the other from another base station. This also affects the optimality of the localized r-strip creation algorithm. These problems are not specific to the localized r-strip tile creation algorithm. The localized algorithms for creating regular patterns like hexagon, square or triangular grids also suffers these problems. Another problem that is specific to r-strip tile creation algorithm is: Non-optimal placement of the vertical strip that is needed to connect different horizontal r-strips. In the ideal case, it needs only one router to connect two horizontal strips. However, if the agents move in a adversarial manner, it needs one router per every second router in the horizontal strip.

5.5 Analysis of Agent-Assisted Router Deployment Algorithm

The localized r-strip creation without restricting agent movements or causing disconnection is not an optimal solution due to the problems mentioned above. Hence the actual number of robots needed for localized r-strip creation is much more than the estimated static r-strip tile value. Figure 5 shows that the agent assisted router deployment algorithm's performance is quite close to the actual static r-strip tile $RSTR_{min}$ value. Hence it is one of the best localized approaches to create an AMRoNet.

If we calculate the estimated number of robots needed for the hexagonal grid HEX_{est} in the specified square region for different r_c values, we observe that they are very close to the average number of routers used by the proposed algorithm. Hence we could use the equation $ARD_{est} = \frac{0.77}{r^2} * A$ to get an approximate estimate of the total number of routers needed to cover a given area A. This helps the agents in making an estimate on the numbers routers they need to carry, before beginning their exploration.

In our experiments, where we used the floor plan of our institute, we found that the proposed algorithm performed equally well, irrespective of the obstacles present in the area. The presence of obstacles affects the performance of other localized agent-assisted regular pattern creation algorithms, as they prevent the routers from reaching the optimal goal point. Our approach even works in area where we do not have any prior model or map of the environment. It could be extended to make it work without any location information, where we need just the link quality estimate provided by the WiFi devices. In such cases, the greedy deployment strategy is performed when the link quality drops below a threshold. Routers deployed during the triangular deployment, move in the direction where the link quality tends to be weak, in order to maximize the coverage area.

6 Conclusion

We have presented a new localized and distributed algorithm for creating an ad-hoc mobile router network that facilitates communication between the agents without restricting their movements. The agent-assisted router deployment algorithm has a greedy deployment strategy for releasing new routers effectively into the area and a triangular deployment strategy for connecting different connected components created by the agents exploring from different base stations. Empirical analysis shows that the number of routers deployed by the agent-assisted router deployment algorithm is close to the optimal static r-strip tile values. The performance of the algorithm is not affected by the number of agents or obstacles present in the environment. Increase in the number of base stations did not make any noticeable performance difference either.

We plan to verify the performance of the proposed algorithm in real life scenarios. The performance of the algorithm with link quality estimate needs to be validated with more quantitative results. We conclude that localized algorithms for achieving optimal *communication area coverage* are worth exploring more.

References

1. Bai, X., Kumar, S., Xuan, D., Yun, Z., Lai, T.H.: Deploying wireless sensors to achieve both coverage and connectivity. In: Proceedings of the 7th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2006, pp. 131–142. ACM, New York (2006)
2. Clark, B.N., Colbourn, C.J., Johnson, D.S.: Unit disk graphs. Discrete Mathematics 86(1-3), 165–177 (1990)
3. Dynia, M., Kutylowski, J., auf der Heide, F.M., Schrieb, J.: Local strategies for maintaining a chain of relay stations between an explorer and a base station. In: Proceedings of the Nineteenth Annual ACM Symposium on Parallel Algorithms and Architectures, SPAA 2007, pp. 260–269. ACM, New York (2007)
4. Ganguli, A., Cortes, J., Bullo, F.: Multirobot rendezvous with visibility sensors in nonconvex environments. IEEE Transactions on Robotics 25(2), 340–352 (2009)
5. Gerkey, B., Vaughan, R.T., Howard, A.: The player/stage project: Tools for multi-robot and distributed sensor systems. In: ICAR 2003, Proceedings of the 11th International Conference on Advanced Robotics, New York, NY, USA, pp. 317–323 (June 2003)

6. Heo, N., Varshney, P.: A distributed self spreading algorithm for mobile wireless sensor networks. In: IEEE Wireless Communications and Networking, WCNC 2003, vol. 3, pp. 1597 –1602 (March 2003)
7. Herbrechtsmeier, S., Witkowski, U., Rückert, U.: Bebot: A Modular Mobile Miniature Robot Platform Supporting Hardware Reconfiguration and Multi-Standard Communication. In: Kim, J.-H., Ge, S.S., Vadakkepat, P., Jesse, N., Al Manum, A., Puthusserypady K, S., Rückert, U., Sitte, J., Witkowski, U., Nakatsu, R., Braunl, T., Baltes, J., Anderson, J., Wong, C.-C., Verner, I., Ahlgren, D. (eds.) Progress in Robotics. CCIS, vol. 44, pp. 346–356. Springer, Heidelberg (2009)
8. Howard, A., Mataric, M.J., Sukhatme, G.S.: Mobile sensor network deployment using potential fields: A distributed, scalable solution to the area coverage problem. In: DARS: 6th International Symposium on Distributed Autonomous Robotics Systems, pp. 299–308 (June 2002)
9. Iyengar, R., Kar, K., Banerjee, S.: Low-coordination topologies for redundancy in sensor networks. In: Proceedings of the 6th ACM International Symposium on Mobile Ad Hoc Networking and Computing, MobiHoc 2005, pp. 332–342. ACM, New York (2005)
10. Kutyowski, J., auf der Heide, F.M.: Optimal strategies for maintaining a chain of relays between an explorer and a base camp. Theor. Comput. Sci. 410, 3391–3405 (2009)
11. Lin, Z., Broucke, M., Francis, B.: Local control strategies for groups of mobile autonomous agents. IEEE Transactions on Automatic Control 49(4), 622–629 (2004)
12. Tekdas, O., Wei, Y., Isler, V.: Robotic routers: Algorithms and implementation. Int. J. Rob. Res. 29, 110–126 (2010)
13. Zavlanos, M., Jadbabaie, A., Pappas, G.: Flocking while preserving network connectivity. In: 46th IEEE Conference on Decision and Control, pp. 2919–2924 (December 2007)
14. Zou, Y., Chakrabarty, K.: Sensor deployment and target localization based on virtual forces. In: Twenty-Second Annual Joint Conference of the IEEE Computer and Communications, INFOCOM 2003, March-April 3, vol. 2, pp. 1293–1303. IEEE Societies (2003)