2010 IEEE International Conference on Robotics and Automation
Anchorage Convention District
May 3-8, 2010, Anchorage, Alaska, USA

A Strategy for Improving Observability with Mobile Robots

Andrew Drenner, Michael Janssen, and Nikolaos Papanikolopoulos
Department of Computer Science and Engineering
University of Minnesota
Minneapolis, MN 55455
Email: {drenner|mjanssen|npapas}@cs.umn.edu

Abstract— **Many surveillance and reconnaissance tasks make use of multi-cameras in order to ensure that a particular mission is accomplished. These networks of cameras are useful as they can reduce the cost of human observers, are continuously observant (unlike humans who may fall asleep on the job), and can be implemented for a fairly low cost. However, in many scenarios, it does not make sense or may not be possible to have a fixed camera installation because the surveillance may only be needed for a short duration or it may take too long to do a proper install and observation is needed now.**

In terms of short terms and immediacy, mobile robots acting as a camera network provide an interesting middle ground. They can be deployed quickly to cover immediate needs, and they can be packed up and moved to another area if needs change. However, as the duration of the mission in which they are used increases, the robotic team will run out of power. This paper addresses some of the issues with keeping a surveillance team active while their batteries drain. Multiple task-reallocation methods are used in conjunction with an analysis of the effects of fixed vs mobile docking stations. Simulations were run requiring the team to provide camera coverage of a group of mobile "pedestrians" moving dynamically through a scene and the results are presented.

I. Introduction

The problem of understanding where to place cameras in a given scene has great practical importance. There are numerous strategies for dealing with where to place the cameras, ranging from the classical art gallery problem [1], to the exterior visibility problem [2], to determining the optimal locations to place cameras for continuous tracking [3]. In each of these approaches, the goal is to determine how many cameras are necessary to cover a given scene effectively, where effectiveness is a function of the task at hand. These algorithms work extremely well when cameras can be installed in a fixed position where secondary constraints such as available power are effectively non-issues.

In [4], the cameras are actually modeled by individual mobile robots and the approach is to adapt their location as the scene changes. The work in this paper takes that work one step further. In addition to assuming that cameras must move over time to respond to dynamic events, the amount of power available to the mobile cameras is constrained. Individual mobile cameras must seek a source of power when they are running low. Coordination between individual members of the team must occur when parts of the team are no longer able to perform their aspect of the mission.

In order to support the team of cameras, a docking station is introduced which can both recharge and transport multiple robots (cameras) simultaneously. Simulated results show the performance difference when this docking station is mobile and fixed, along with the effects of multiple task reassignment strategies which are used when robots (cameras) no longer have enough power to participate in the observation.

The remainder of this paper will be organized as follows. Section II will discuss related work in the areas of camera placement, task allocation across robotic teams, and coordination of robotic teams. Section III and Section IV discuss the approaches for team coordination and camera placement used in this work respectively. Section V provides an overview of the simulation environment and performance metrics. Future work and conclusions are discussed in Section VI.

II. Related Work

This work is most related to recent work which focuses on multi-robot surveillance. In [5], a multi-robot system is simulated with Gazebo in order to surveill a large arbitrary area, detecting any movement and alerting a human guard. The robot movement is random with stops in order to detect movement in the video. This is somewhat similar but simpler than our activity monitoring robot team. However, our system takes into account the activity which is present in an area and attempts to monitor the areas where it is most likely for activity to occur.

This problem could be considered to be a modification of the sensor planning problem [6] with a dynamic feature set. We only seek to find the optimal locations and rotations of the cameras in the system, and have no interest in the computationally complex feature detection optimization. Additionally, the robotic team which is simulated for sensing is very resource-constrained and many of the sensor planning tasks are unable to run on the system.

Static camera placement has been studied extensively, starting with the art gallery problem and extending to more recent placement algorithms for sensor networking and surveillance applications. In [7], a large area under observation is split into sections which are assigned an importance and these sections are given more weight. Thus it is similar to our algorithm but uses a simpler method based on the geometry of the area instead of the actual activity which has been observed. In [8], a placement using heterogeneous sensors and a metric based on observability of subjects is formulated. Again the authors do not take into account the dynamic nature of many situations, but the inclusion of a motion sensor could point to a possible use for simpler robots in the system proposed here.

Optimal allocation of tasks among team members has been shown to be an NP-hard problem [9]. Because of this, much of the current research has been focused on approximations using auctioning methods [10] using various cost functions. Considering the size of the robot team in the experiments, it was possible and feasible to compute all possible robot assignments. Locating the correct cost functions are therefore more important to this research, although if the system is expanded to more robots these auction methods should be considered.

This work utilizes the simulation enhancements to the Player/Stage system allowing marsupial actions [11]. Using the Player/Stage system makes it possible to run many simulations simultaneously as well as calculating the power requirements of the moving robots. The marsupial extension allows for the robots in the simulation to be picked up and replaced and recharged while inside the docking station robots.

III. Deployable Team Formulation

The coordination of the docking station with the deployable robots are based upon the work in [12]. In this work the robots being supported by the docking station are modeled as a finite state machine. This machine has three superstates, “Active”, “Inactive”, and “Maintenance”. In the “Active” superstate, the deployable robots be in one of two states: “Deploy” where the robot is deploying to a specific mission objective or “Mission” where the robot is conducting a mission-specific task. The “Inactive” superstate contains those states where the robot is unable to work on any mission-specific task because it is either “Seeking Home” in search of power, “Abandoned” to the point where it can not make it to the docking station on it’s own power and enters a low power state, or “Dead” where it is completely without power and cannot be recovered. The remainder of states fall into the “Maintenance” superstate, where the robots are “Waiting to Dock”, “Docked”, and “Waiting to Deploy”.

The deployable robots must be serviced by a docking station whose actions are determined as follows. Supportable robots in need of assistance communicate throughout the network that they need support and provide an estimate of their location and remaining energy reserves. Using this information, the docking stations divide the robots in need of service into clusters using the ISODATA [13] algorithm. Once divided into clusters, a specific cluster (S_I) is chosen for the docking station to support. The docking station will continue to support this cluster until all of the members have docked or died.

The docking station can be represented by a vector $\overrightarrow{R_D}$ which has the location of the docking station at a given time. The i^{th} member of S_I is represented by another vector ($\overrightarrow{R_i}$). The docking station must choose where to position itself in the next time step to minimize the cost of recovering all members of S_I. The cost function being optimized is given in Equation (1):

$$f(\overrightarrow{R_D}) = \sum_{i=1}^{n} x e^{\alpha(x-1)}. \qquad (1)$$

In this equation, $x = \frac{1}{v\varepsilon} dist(\overrightarrow{R_D}, \overrightarrow{R_i})$, $v = R_{i_V}$ or the velocity of the i^{th} robot, and $\varepsilon = R_{i_E}$ or time remaining based upon the remaining energy of the i^{th} robot. There are a number of methods which can be used to solve this minimization.

While one docking station is busy with a particular set S_I, other docking stations can repeat this process with the remaining robots that are in need of support to support larger teams in parallel. Previous work, [12], has shown how this coordination algorithm can be easily scalable to larger numbers of robots and docking stations.

It should be noted that in the simulation presented in Section V, the deployable robots have enough power to run for four times the amount of time it takes to charge. Thus, in an ideal situation (i.e., the docking station is not full, there are no difficulties in docking, the docking station was always in the perfect position, etc.) the maximum amount of time that a deployable robot can stay active is 80 % of the simulation runtime.

IV. Camera Placement Formulation

The camera placement formulation derived by Bodor [14] can be utilized with a distributed robotic team. This formulation determines the positions that the robots should take in order to observe the trajectories of targets that are maneuvering in an area of interest.

The basic formulation is as follows:

1) Determine a minimum distance for observation — The key to successful operation of this camera placement algorithm is to maximize the resolution of the images being observed. To ensure that the highest resolution images can be obtained, the path of motion must be observed in its entirety from the closest possible position. We will call this position d_0, and the actual distance between the i^{th} camera and j^{th} path as d_{ij}.
2) Determine the best camera position for the given path — Each path must fall within the frustum generated by the view of the camera projected on the ground plane. This ensures that the camera is able to see the activity along the path in order to attempt to monitor / classify that activity.
3) Minimize the effects of foreshortening — In 2D, there are two sources of foreshortening: the angle between the camera position and the path normal (θ) and the angle between the normal to the image plane and the path center (ϕ). To minimize these effects, the camera should observe the path directly on, thus $\cos(\theta) = 1$ and $\cos(\phi) = 1$. In the 3D case, two more angles are introduced for observation, but will not be utilized in this simulation.

Using the formulation above, an objective function for each path-camera pair can be defined as follows:

$$G_{ij} = \begin{cases} 0 & \text{if } d_{ij}^2 < d_0^2, \\ \frac{d_0^2}{d_{ij}^2} \cos(\theta_{ij}) \cos(\phi_{ij}) & \text{otherwise.} \end{cases} \tag{2}$$

In a single camera case ($i = 1$), the cost function to be optimized becomes:

$$V = \sum_{j=1}^{paths} G_{ij}. \tag{3}$$

In multiple camera systems, the cameras must be placed jointly in order to achieve an optimal solution. The computations to do this are exponential in terms of the number of cameras, and quickly grows infeasible for large systems. As a result, an iterative approach is used to make the complexity of camera placement linear with respect to the number of cameras by "pushing" the next camera into locations that currently have the lowest observability. This is accomplished by using the inverse of the observability of the previous camera values $(1 - G_{kj})$ in Equation (4).

$$V_i = \sum_{j=1}^{paths} \left[G_{ij} \prod_{k=1}^{i-1} (1 - G_{kj}) \right]. \tag{4}$$

The cost of each individual camera can be summed to provide the total cost function to optimize:

$$V = \sum_{i=1}^{cameras} V_i. \tag{5}$$

Using this approach, the ideal positions and poses for each of the cameras (or in this case observing robots) can be calculated.

V. Simulation

In this simulation, a series of simulated people are moving randomly through a scene. A team of robots are tasked with observing these individuals. The algorithm for determining where the robots should be placed for observation is based upon the work in [4], which looks at optimal camera placement using a team of robots and applies the team of robots to scenarios where the distribution of people being observed changes over time, requiring the robotic systems to redistribute themselves. Figure 1 shows a sample environment where such a scenario would take place.

As individuals move throughout the scene, their trajectories are tracked. In reality, this can be done with overhead cameras or spliced together from static camera networks. In this simulation, the tracking of objects was done using the Stage interface. These trajectories are then fed to a camera placement module. The camera placement algorithm uses a model of the environment and the recorded trajectories to determine where the robots which are carrying the camera systems should be located. Figure 2 provides an overview of the simulation.

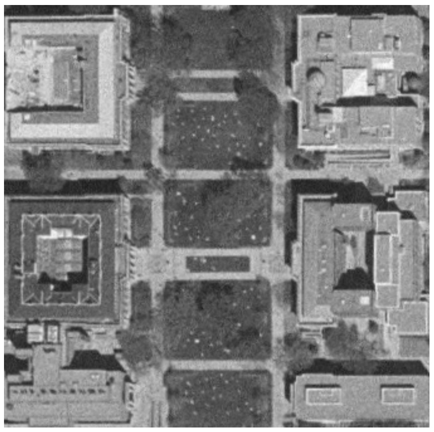

Fig. 1. Sample environment where pedestrians may be monitored as they walk on the sidewalks that surround the buildings.

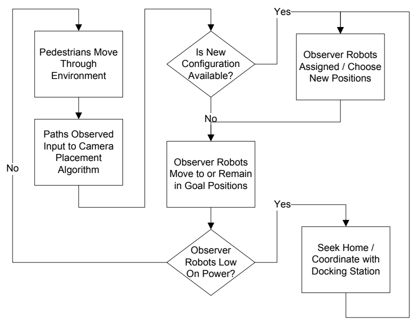
Yes
Pedestrians Move Through Environment
Is New Configuration Available?
Observer Robots Assigned / Choose New Positions
No
Paths Observed Input to Camera Placement Algorithm
No
Observer Robots Move to or Remain in Goal Positions
Yes
Observer Robots Low On Power?
Seek Home / Coordinate with Docking Station

Fig. 2. A schematic overview of the simulation.

Once the observation positions are known, the robots must deploy to these positions. There are a number of methods by which this assignment could take place. Section V-A will discuss the methods implemented further.

As the deployed robots which are monitoring the scene lose power, the mobile docking stations are utilized to coordinate their recovery and redeployment. Additionally, as the trajectories of individuals moving throughout the scene are continuously monitored, the camera placement software periodically re-updates the placement of the distributed camera system. Depending upon the number of cameras deemed necessary by the camera placement software, extra robots can be stored on board the docking stations for recharge.

A. Task Allocation Schema

Throughout the simulation runs, one of the key factors affecting the system's overall performance is expected to be how robots are assigned to take over tasks from one another as individuals lose power or as dynamic changes in the environment require reconfiguration of the distribution of observing robots.

Three strategies have been implemented for addressing the task allocation problem:

1) *Random Allocation* – This schema is not expected to be optimal, but will serve as a baseline. The distributed black board system that is used to manage communications will simply distribute locations to each of the

deployed robots at random. In this case, no analysis will be done with respect to how far the robots will have to travel, how much energy remains on each robot, etc.

2) *Simple Allocation* – The simple allocation strategy will attempt to improve performance of the system over simple random allocation by prioritizing which observing robot will move to which location using simple Euclidean distances between the robot and the observation position. The smaller the distance the observing robot must travel, the higher the priority it will be for the observing robot to go there.
3) *Advanced Allocation* – A third, more advanced, strategy will further attempt to improve this performance by having the observing robots positions assigned based on more than just location. Using this strategy, the observable robots will also weight their priority for where to go by how much power they have available.

B. Simulation Results

A total of 180 runs of this simulation were conducted. These runs were divided up as follows. There are 60 runs of each of the three task swapping methods listed in Section V-A, and each of those 60 runs are split into 30 runs where the docking stations are fixed and 30 runs where the docking stations are mobile.

In each run, there are 20 "pedestrians" which were moving throughout the scene generating paths. Observing these "pedestrians" are a team of 8 deployable robots which are supported by a single docking station. The generation of the optimal camera locations is CPU intensive, and new configurations are available at as soon as they have been computed, albeit at irregular intervals. The irregularity results from the fact that since the "pedestrians" generate the paths as the result of a random walk, at times they may be clustered where a single camera can achieve significant observability. Paths are also aged out of existence in order to reduce the computational burden on the camera placement algorithm. Each of the simulations runs for a total of 80 minutes, so that there is sufficient time to have paths age out multiple times and sufficient time to enable several recharges of the deployable robots to occur.

All of the runs take place in the $CityMap$ environment (Figure 3) which was specially created for Stage for this simulation. In this image, there are a number of black polygons which represent obstacles for pedestrians and observing robots to navigate around. They have been staggered to introduce more variation in the paths that pedestrians must take to cross the scene. As with other environments, this has been scaled to be 50m by 50m to allow expansive deployment by the observing robots.

The metrics of "Mean Time in State (%)" for each of the "Active", "Inactive", and "Maintenance" superstates (as defined in Section III), the number of observing robots which died, and how many recharges a robot receives are all still included in these experiments. However, an additional metric has been created based upon a theoretic observability of each path. When the camera placements are being calculated, each path gains observability (on a scale of [0 1]). Using this metric, an upper bound with the number of available cameras can be calculated and normalized over the number of paths to give a theoretical upper bound as a percentage. Then, as the robots are deploying, observability of the real-time configuration is calculated using the same method.

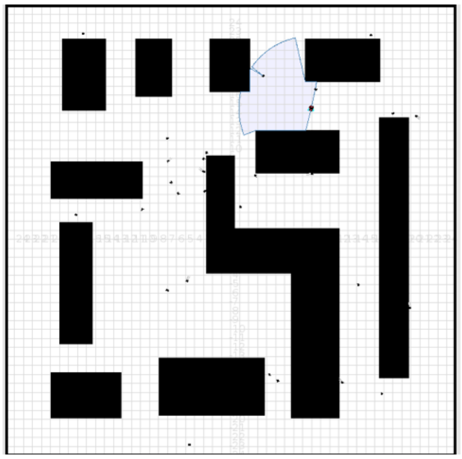
Fig. 3. Stage simulation in the $CityMap$ environment.

Figures 4-9 illustrates how the theoretical and actual observability change over time in each of the six different configurations. The periodic drop that is observable in some of the scenes is the result of the deployed robots having to obtain power and being forced to abandon their observation tasks. In these figures, the upper portion of the figure depicts the observability of the "pedestrians" in two ways. The first, indicated by solid blue lines is the theoretical upper bound of observability (T_o) using 8 cameras. The X's at the start of these lines indicate when the computation for that observation was made. It is important to note that these line segments serve as the base points for where the robots should deploy to and are based on historical data. The second illustration of observability is shown in the form of the dotted green lines. This shows what the observability of the "pedestrians" are based on the present location of the robots using the live path data (L_o). Thus it is possible to have the value of $L_o > T_o$ as T_o is based on aged information.

The lower portion of Figures 4 - 9 indicate the number of robots that are in the "Active" superstate. Periodic changes in this value correspond to robots having to go to the docking station for recharge. It should be noted that the first major drop-off that occurs in both the top and bottom portions of the figure represent that first recharge point. Since all robots are deployed simultaneously, their initial recharge typically occurs at a single time. Previously this has been discussed as one of the reasons that a significant number of deployable robots die as the docking station can become overwhelmed with requests.

The results for these simulations are shown in Tables I and II. Comparing the results between these two tables, it is clear that there is a significant difference between the performance

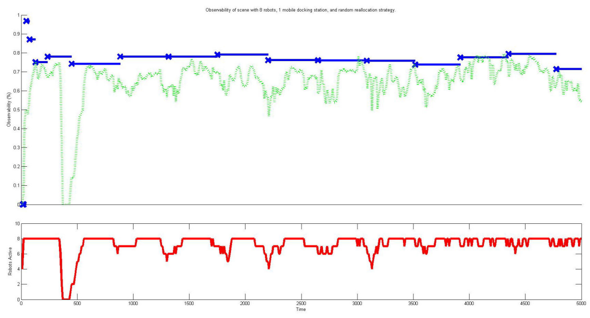

Fig. 4. Theoretical and actual observability calculation for a single simulation run involving 8 observing robots, a single mobile docking station, and random task reassignment.

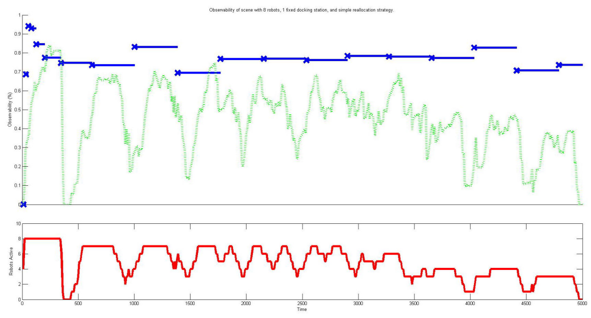

Fig. 7. Theoretical and actual observability calculation for a single simulation run involving 8 observing robots, a single fixed docking station, and simple task reassignment.

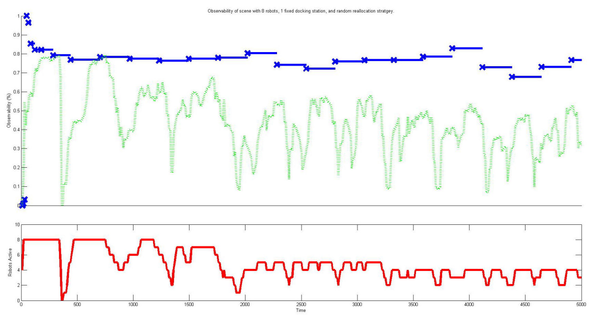

Fig. 5. Theoretical and actual observability calculation for a single simulation run involving 8 observing robots, a single fixed docking station, and random task reassignment.

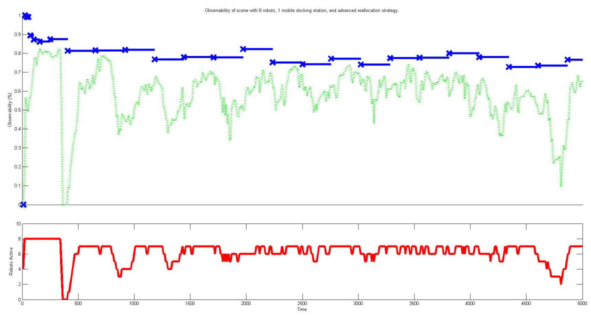

Fig. 8. Theoretical and actual observability calculation for a single simulation run involving 8 observing robots, a single mobile docking station, and advanced task reassignment.

of the system with respect to whether or not there is a moving dock. There were minute differences between the allocation strategies, but ANOVA testing showed that there was no statistically significant difference.

This result was somewhat surprising. However, the lack of a statistically significant result is likely the combination of two factors. First, the rate at which the optimal configuration could be calculated was low, thus the deployable robots would continually swap the same configurations and may not have been able to achieve anything significant benefit from the swaps. Second, the task reallocation was happening with only 8 robots. This was done because of the cost of calculating more optimal camera placement with more paths and more cameras was computationally too expensive. It is believed that in a larger team configuration would have had a more noticeable benefit. In order to test this theory, a faster or alternate implementation of the camera placement algorithm would be required.

VI. Future Work and Summary

This work can be extended in many ways to improve applicabilty, computational requirements and flexibility. Removing the requirement for global tracking of subjects is an interesting enhancement which enables the system to be used in situations where you cannot easily track subjects globally. In these situations, it may be required to have a

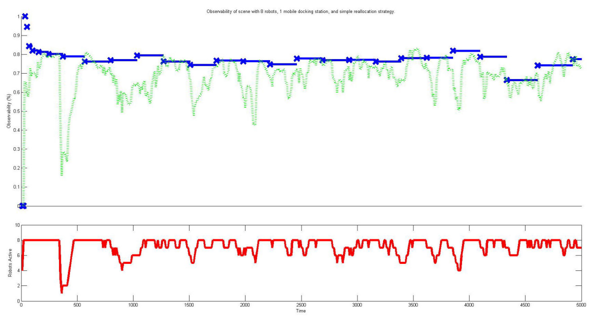

Fig. 6. Theoretical and actual observability calculation for a single simulation run involving 8 observing robots, a single mobile docking station, and simple task reassignment.

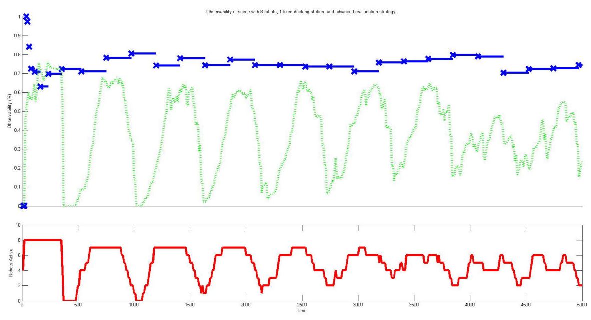

Fig. 9. Theoretical and actual observability calculation for a single simulation run involving 8 observing robots, a single fixed docking station, and advanced task reassignment.

TABLE I

Results of simulation with a fixed docking station.

		Random Allocation	*Simple Allocation*	*Advanced Allocation*
Mean Time in State(%)	*Active*	54.79	55.02	56.40
	Inactive	37.57	37.10	36.14
	Maintenance	7.64	7.88	7.45
# Robots Dead	*Mean*	2.93	2.83	2.77
	Std	0.95	1.69	1.72
Robot Recharges	*Mean*	7.28	7.28	7.46
	Std	1.05	1.78	1.61
% Live Observability	*Mean*	43.87	45.52	47.11
	Std	0.06	0.08	0.10

TABLE II

Results of simulation with a mobile docking station.

		Random Allocation	*Simple Allocation*	*Advanced Allocation*
Mean Time in State(%)	*Active*	73.13	72.97	75.40
	Inactive	15.64	15.99	13.01
	Maintenance	11.23	11.04	11.59
# Robots Dead	*Mean*	0.93	0.87	0.47
	Std	1.13	0.91	0.50
Robot Recharges	*Mean*	10.03	10.03	10.45
	Std	0.94	0.86	0.57
% Live Observability	*Mean*	62.35	62.66	63.62
	Std	0.04	0.04	0.03

general coverage algorithm to bootstrap the system before adapting to the dynamic needs of a situation, and modify the placement algorithm to take into account the information loss.

Three separate strategies were investigated with experiments here, with no statistically significant differences found between them. This is possibly due to the limited team size and the relatively low reconfiguration rate that the optimal placement implementation allowed. The placement formulation currently is very computationally intensive, taking many minutes to complete a placing when there are 500 activity paths to consider. Clustering or dynamic programming could be used to reduce this time, enabling the system to respond more quickly to dynamic situations or take more historical data into account. already, testing with a larger working area or sensing team size may amplify the differences in robot reasssignment strategies.

The docking stations in this formulation were idle when they were not actively involved with recharging one of the sensing team robots. It is worth considering using at least part of this idle time to enhance the sensing task. In this case, it would be essential for the docking stations to either balance the importance of the recharging and sensing tasks, or have the activity monitoring task provide feedback on points which would enhance the observability of the moving subjects but were non-essential for an adequate coverage. The idle time of these docking stations could also be used in some other useful way as well which should be examined.

To illustrate the effectiveness of the coordination strategy, a new simulation was constructed in which the task assigned to the deployed robots was to maximize observability of a scene, rather than to explore randomly. Observability was derived from the approach of Bodor [14], and normalized to indicate the percentage of paths that were completely observed. A series of simulations were then carried out in Player/Stage illustrating how a team of eight robots could observe twenty pedestrians moving through a scene with the support of one docking station. Half of the simulations were run with a fixed docking station and the remainder allowed the docking station to be mobile. The mobile docking station enabled a much higher amount of time in the "Active" superstate as well as a much higher observability rating.

VII. Acknowledgements

This work has been supported in part by the National Science Foundation through grants #IIS-0219863, #CNS-0224363, #CNS-0324864, #CNS-0420836, #IIP-0443945, #IIP-0726109, and #CNS-0708344 and a National Science Foundation Graduate Research Fellowship.

References

[1] J. O'Rourke, *Art Gallery Theorems and Algorithms.* New York: Oxford University Press, 1987.

[2] V. Isler, S. Kannan, K. Daniilidis, and P. Valtr, "VC-dimension of exterior visibility," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 26, no. 5, pp. 667–671, 2004.

[3] Y. Yao, C. H. Chen, B. Abidi, D. Page, A. Koschan, and M. Abidi, "Can you see me now? sensor positioning for automated and persistent surveillance," *Systems, Man, and Cybernetics, Part B: Cybernetics, IEEE Transactions on*, vol. PP, no. 99, p. 1, July 2009.

[4] L. Fiore, G. Somasundaram, A. Drenner, and N. Papanikolopoulos, "Optimal camera placement with adaption to dynamic scenes," in *To Appear in Proceedings of the 2008 IEEE International Conference on Robotics and Automation*, Pasadena, CA, May 2007.

[5] E. Folgado, M. Rincón, J. Álvarez, and J. Mira, "A multi-robot surveillance system simulated in gazebo," *Nature Inspired Problem-Solving Methods in Knowledge Engineering*, pp. 202–211, 2007.

[6] K. Tarabanis, P. Allen, and R. Tsai, "A survey of sensor planning in computer vision," *Robotics and Automation, IEEE Transactions on*, vol. 11, no. 1, pp. 86–104, Feb 1995.

[7] K. Yabuta and H. Kitazawa, "Optimum camera placement considering camera specification for security monitoring," in *Circuits and Systems, 2008. ISCAS 2008. IEEE International Symposium on*, May 2008, pp. 2114–2117.

[8] S. Ram, K. R. Ramakrishnan, P. K. Atrey, V. K. Singh, and M. S. Kankanhalli, "A design methodology for selection and placement of sensors in multimedia surveillance systems," in *VSSN '06: Proceedings of the 4th ACM international workshop on Video surveillance and sensor networks*. New York, NY, USA: ACM, 2006, pp. 121–130.

[9] M. G. Lagoudakis, E. Markakis, D. Kempe, P. Keskinocak, A. Kleywegt, S. Koenig, C. Tovey, A. Meyerson, and S. Jain, "Auction-based multi-robot routing," in *In Robotics: Science and Systems*, 2005, pp. 343–350.

[10] M. Nanjanath and M. Gini, "Dynamic task allocation for robots via auctions," in *Robotics and Automation, 2006. ICRA 2006. Proceedings 2006 IEEE International Conference on*, 2006, pp. 2781–2786. [Online]. Available: http://ieeexplore.ieee.org/xpls/abs_all.jsp?arnumber=1642122

[11] M. Janssen and N. Papanikolopoulos, "Enabling complex behavior by simulating marsupial actions," in *15th Annual Mediterranean Conference on Control and Automation*, Athens, Greece, June 2007.

[12] A. Drenner, M. Janssen, and N. Papanikolopoulos, "Coordinating recharging of large scale robotic teams," in *Proceedings of the 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems*, St. Louis, MO, Oct 2009.

[13] C. W. Therrien, *Decision Estimation and Classification.* New York: Wiley, 1989.

[14] R. Bodor, A. Drenner, P. Schrater, and N. Papanikolopoulos, "Optimal camera placement for automated surveillance tasks," *Journal of Intelligent and Robotic Systems*.