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

Bacteria Controller Implementation on a Physical Platform for Pollution Monitoring

John Oyekan and Huosheng Hu
School of Computer Science and Electronic Engineering,
University of Essex, Wivenhoe Park, Colchester CO3 4SQ, United Kingdom.
Email: jooyek@essex.ac.uk; hhu@essex.ac.uk

***Abstract*—Inspired by the simplicity of how nature solves its problems, we implement a bacteria controller on a physical platform that would enable the localisation and subsequent mapping of environmental pollution. We investigate the effects of each parameter in the controller on the localisation and exploration ability of the platform used. We also present how we can tune the controller for a given environmental condition depending on whether localisation or exploration is of a major priority. Some experimental results are presented to show the feasibility and performance of the proposed bacteria control.**

***Index Terms*—Bacterium Inspired Algorithm, Environmental Monitoring, Flocking.**

I. Introduction

In order to use robotic agents to provide environmental monitoring, two issues have to be solved. The first issue has to do with controlling the agents in the most appropriate way to collect data of environmental pollution being monitored. The second issue has to deal with processing the data and presenting it in the most appropriate way for a human user. In this paper, we focus on the first challenge i.e. developing a controller that would enable us to control a robotic agent towards localising the source of environmental pollution whilst providing environmental coverage.

In order to monitor an environmental pollutant, the initial thought might be to make sure that every single area in the environment is covered. This approach has led to the development of various individual agent deterministic algorithms that ensure that every area in the given environment is visited at least once during the run time of the agent. For example, [1] uses a spanning tree algorithm to provide coverage to an area. This involves the cellular decomposition of an area into cells and then developing a spanning tree to ensure that every area of the cell or vertex in the given decomposed environment is covered.

However, using this approach quickly starts losing appeal when a large environment is to be covered by the single agent and if the environmental pollutant changes constantly over a short period of time. This leads to data collected being outdated quickly. In order to solve this large environment size problem, multiple agents could be used. By using multiple agents, individual agents in the "system" could be programmed to spread out in the environment and hence collect data everywhere at once.

This has led to a development of the spanning algorithm into a multi spanning algorithm [2] to find tree cover with similar tree weights. However, using this approach does not address the second issue properly. Areas in the environment that do not contain data are still covered and loss of data can happen if the agent is not near or in the same area as the environmental pollutant. This approach is deterministic and hence ensures that most or every area in the environment is covered. However, it has been proven that the performance of deterministic approaches get close to that of stochastic approaches when their efficiency is reduced [3][4]. In practical situations, efficiency could be reduced by wheel slippage, sensor inaccuracies and other platform imperfections.

Balch [3] was able to prove that using a random search method enabled the use of less intelligent robots. Since a stochastic algorithm can deal with changes in the dynamically changing environment, we investigate the use of a controller based upon the bacteria random biased walk. By using our controller with a flocking algorithm, we aim to distribute the agents based on the concentration distribution of the environmental pollutant being monitored. This leads to more efficient use of a limited number of agents.

Researchers have used various multi-agent approaches to achieve this, including Voronoi partitions [5] [6] and Virtual Spring Mesh approachs[7]. However, these approaches are either computationally expensive or require a long distance communication between agents. In addition, a prior knowledge of the target is required when using Voronoi partition and this approach can only be used in polygon derived environments. In this paper, we shall focus on the development and implementation of our chosen bacteria controller on a physical robotic platform and investigate its parameters.

The rest of the paper is organised as follows: Section II describes the implementation of a bacterial chemotaxis controller. In Section III, experimental setting and results are presented. Further investigations are presented in Section IV, including the effect of noise on the bacteria controller, the effect of using memory and how to tune the controller. Finally, a brief conclusion and future extensions are discussed

978-1-4244-5040-4/10/$26.00 ©2010 Crown

in Section V.

II. Implementing a Bacterial Chemotaxis Controller

A. Bacterial Chemotaxis Model

A bacterium finds food sources by executing a biased random walk behaviour. Its motion is made up of two phases namely a run phase and a tumble phase. The run phase can be said to be a straight line motion in a particular direction. When swimming up a gradient, the mean run length is 2.19±3.43 s while if swimming down a gradient, the mean length is 1.40 ±1.88s [8][9]. In other words, the length of the run phase is affected by the concentration of the attractant in the medium and was modeled by Berg and Brown as follows:

$$\tau = \tau_o exp(\alpha \overline{\frac{dP_b}{dt}}) \tag{1}$$

$$\overline{\frac{dP_b}{dt}} = \tau_m^{-1} \int_{-\infty}^{t} \frac{dP_b}{dt'} exp(\frac{(t\prime - t)}{\tau_m}) dt\prime, \tag{2}$$

$$\frac{dP_b}{dt} = \frac{k_D}{(k_D + C)^2} \frac{dC}{dt} \tag{3}$$

where τ is the mean run time and τ_o is the mean run time in the absence of concentration gradients, α is a constant of the system based on the chemotaxis sensitivity factor of the bacteria, P_b is the fraction of the receptor bound at concentration *C*. In our work, *C* was the present reading taken by our Robotic agent. K_D is the dissociation constant of the the bacterial chemoreceptor. $\frac{dP_b}{dt}$ is the rate of change of P_b. $\overline{\frac{dP_b}{dt}}$ is the weighted rate of change of P_b. This is used to simulate the exponentially decaying memory of an event on a bacterium system. In our implementation, we used a 4-element memory to simulate the 4 second memory of a bacterium [10][11]. τ_m is the time constant of the bacterial system.

The above equations determine the time between tumbles and hence the length of runs between tumbles. The tumble phase is performed by the bacteria throwing its flagellum clockwise in the medium. This makes it turn in a random direction σ. This random direction is governed according to Dahlquist et al by a probability distribution which makes the probability of turning either right or left azimuthally symmetric about the previous direction [12]. In our implementation, our robotic agent can randomly choose a range of angles in the set $\sigma \epsilon \{0...360\}$ by spinning on its axis.

In [13], their agent had to cover a certain distance based upon a bias value before it took readings again. It then compared the present and previous readings to decide whether to tumble or keeping moving in a straight line. However, by using the controller above, the system is able to react to a dynamic environment during runs. Light sensor readings are taken every time step and used to determine immediately whether to keep moving in that direction or to tumble immediately.

B. The Platform and Environmental setup

In order to investigate the effects of our algorithm on various platforms, we implemented the algorithm on a Lego mindstorm platform. We used Lejos (A java derivative programming language) to program the robot. For the environmental variable or pollutant, we printed a gradient of black color on paper as shown in Fig. 1. The paper was placed so that the simulated pollutant source was at a position of (0,0) in the arena. We used an infra red light sensor to read the values of the color from the paper and then responded accordingly. The values of the reading from the infra red light sensor was between 0 and 65. We decided against using light as a target source as we did not have total control over the light levels entering our robot arena. This made sure that our results were collected in a controlled environment.

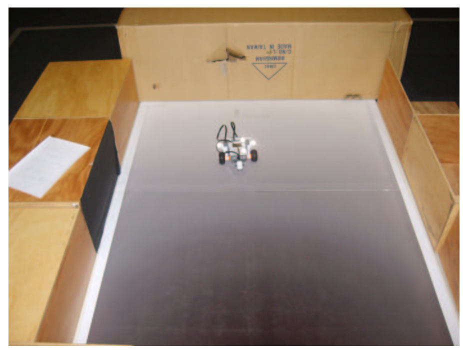


Fig. 1. Lego mindstorm platform in our arena with little background light.

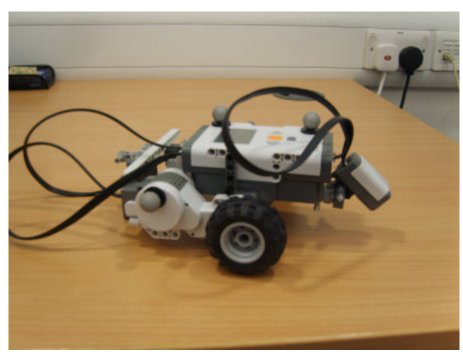


Fig. 2. Lego mindstorm platform with light sensor.

We placed the light sensor at an angle as shown in Fig. 2, which aided the robot decision making capabilities. It enabled our control algorithm take action before it was too late. An overhead motion camera was used to obtain position data of the robot. In addition, the robot was placed in a bounded environment as can be seen in Fig. 1. This made sure that the robot did not wander away from the region of interest during the experiment when investigating the various parameters. The arena for this experiment had a dimension of 1200mm by 1400mm (Width by Length) while the robot had a speed of 18cm per second.

III. Experimental Setting and Results

During our experiments, we placed the robot at a distance of approximately 1200mm from the source. We assumed that we have reached the source when the robot's infra red sensor is within a 50mm by 50mm box at the source. For each parameter change, we took twenty readings to get a good representation of the parameter's effect. To investigate the effects of the parameters, we developed two metrics. The first metric was how quickly the agent was able to localize the source while the second metric was how well environmental coverage was achieved in the environment.

A. Investigating the run length (τ_o) parameter

We casted the mean run time τ parameter into a mean run length parameter and τ_o into a mean run length parameter in the absence of concentrations. In order to investigate the effect of the run length parameter τ_o, we used various values of $5, 10, 15$, and 20. For each parameter, we measured the distance from the source every 500 milliseconds and plotted the average of the results as shown in Fig. 3.

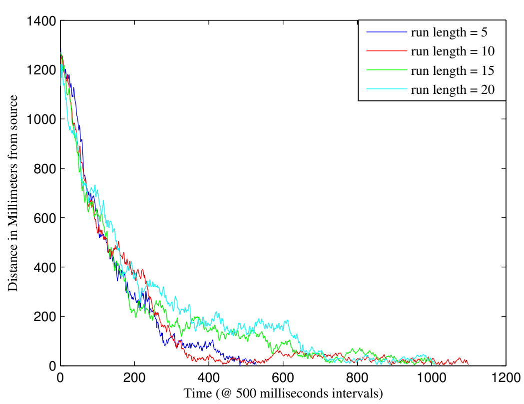


Fig. 3. Graph showing the effect of using different values of run length parameter τ_o. $\alpha = 1000, kd = 2$

From Fig. 3, it can be seen that a smaller value of run length resulted in faster convergence at the source with a faster descent while a larger run length value had the opposite effect. This is seen more clearly in Fig. 4 where we checked for the first algorithm to reach within the 50 by 50 millimeter box at the source.

We also discovered that a smaller value of τ_o results in less environmental coverage but more resolution during searches with little chance of overshooting the source while a larger

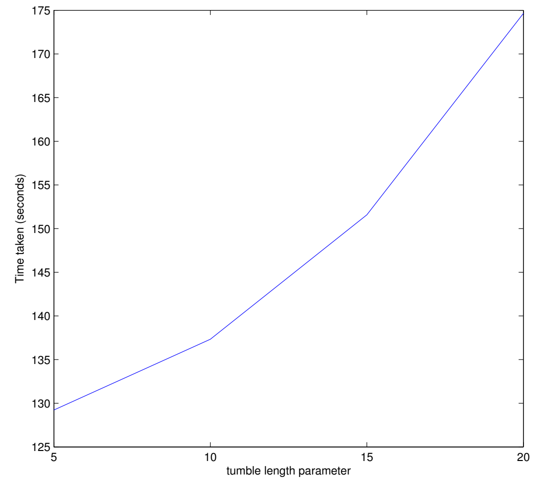


Fig. 4. Showing the time taken for each run length parameter τ_o value.

value of τ_o results in more environmental coverage. This can be seen in Fig. 5. Using $\tau_o = 25$ resulted in a more uniform coverage of the entire arena while using $\tau_o = 5$ resulted in a more directed search towards the source with edge positions covered less.

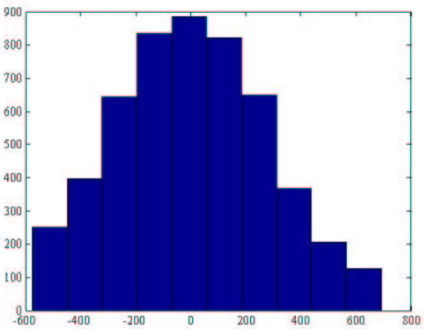
B. Investigating the kd parameter

The kd parameter was investigated by using the values of $2, 10$, and 20. The effects of using the various values are shown in Fig. 6 in which large values of kd resulted in faster descent and convergence. We believe that this parameter is associated with system response.

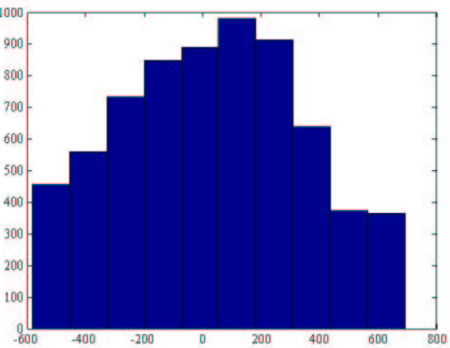
C. Investigating the alpha parameter

In order to carry out investigation into the alpha parameter, we decided to introduce noise into the environment to see if it would have any effect on the performance of the system. This involved putting on the lights in the robot arena. We initially believed that this change in condition would have an effect on our readings because of the nature of the paper. This is because the paper reflected light shining on it which might reduce the effect of the infra red light sensor by introducing noise into the readings.

We used a kd value of 2, a run length value τ_o of 5 and an alpha value of 1000. The results of how quickly it localised at the source is shown in Fig. 8, in which a change in light conditions did not cause the gradient of the localisation curve to change drastically when compared with the gradient of the localisation curve of the same parameters without the light on. This shows that a change in light conditions does not have much effect on our algorithm performance. This robust nature of our algorithm could be because of the nature of the sensor we are using in that it measures infra red emissions



(a)



(b)

Fig. 5. Graphs showing the positions in the arena covered using run length τ_o = 5 Fig. (a) and using run length τ_o = 25 Fig. (b)

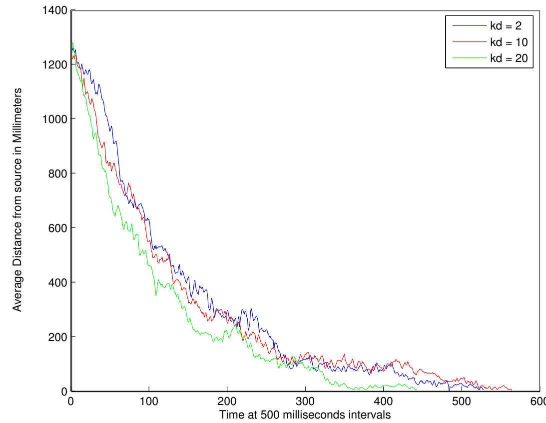
and not light levels. However, from preliminary observations, we believe that the alpha parameter is useful for tuning the controller for various environments to deal with noise. This parameter would be adjusted dynamically in future work to adapt to various environments in order to achieve optimal performance.

IV. Further Investigations

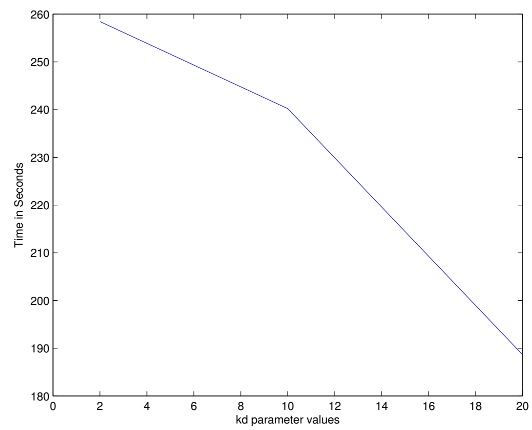
A. Using a light source

In this experiment, we used a light sensor measuring light levels in the environment. However, testing the performance of each parameter was very challenging as the light condition in the arena changed as a result of outside light.

Nevertheless, we were able to discover a very interesting behaviour. We found out that each time the robot backed the light source, the shadow of the robot was casted onto the light sensor path. The robot responded immediately by rotating on its axis to get back into the light. This resulted in a faster system response. This behaviour would be very useful in practical situations when searching for a pollutant in a river for example. If the robot turned around down stream, the robot structure would cause the pollutant to miss the sensor resulting in a low reading and an immediate response.



(a)



(b)

Fig. 6. Showing the rate of descent to the source 6(a) and Showing the time taken for each kd parameter value 6(b).

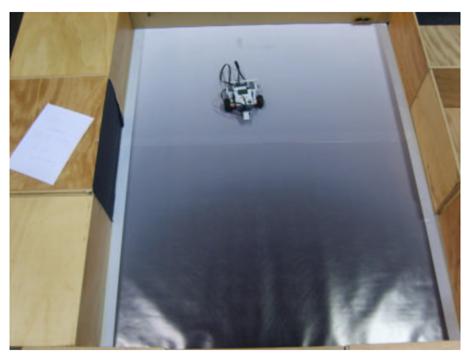


Fig. 7. Showing the arena with lights on. Notice the reflections on the paper

B. Investigating memory

We also investigated the effects of changing the exponential parameter in equation 2. However, we did this in

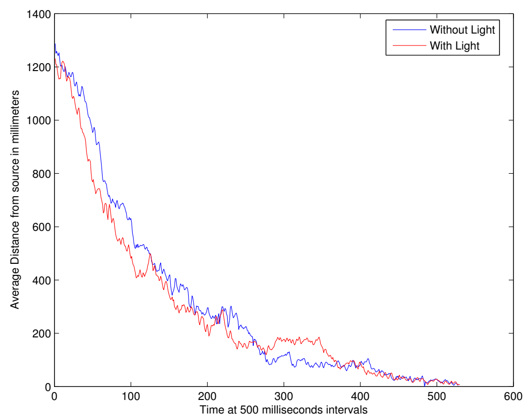


Fig. 8. Showing the system response with light and no light.

simulation. We used a 4 element memory which is similar to the 4 second memory of a bacteria. The exponential function of equation 2 works by weighting the value of the read concentrations by values of 1, 0.367879, 0.135335 and 0.049787 so that present concentration readings are weighted with a value of 1 while readings taken 4 seconds ago are weighted with 0.049787. As a result, this resembles a decaying memory effect. These values are stored in 4 data points respectively in Fig. 9. We investigated the effect of changing the value of data points 2 and 3 of the exponential function in Fig. 9 on a pollutant profile shown in Fig. 11 without changing data points 1 and 4. Our performance matrix in this case was to find out how many agents were able to localise at the source in 30 seconds. Each of the agent had a kinematic model of movement. The results of our experiment is shown in Fig. 10.

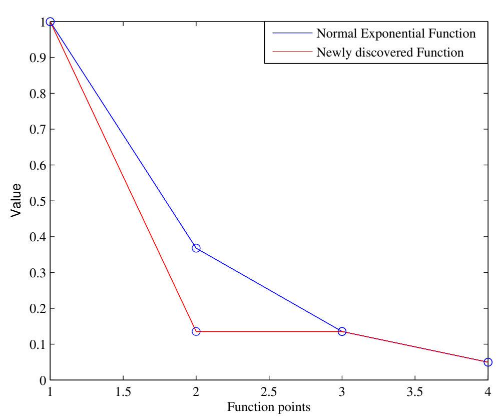


Fig. 9. Showing the normal exponential function and the discovered new function.

From our results, we discovered that using data points

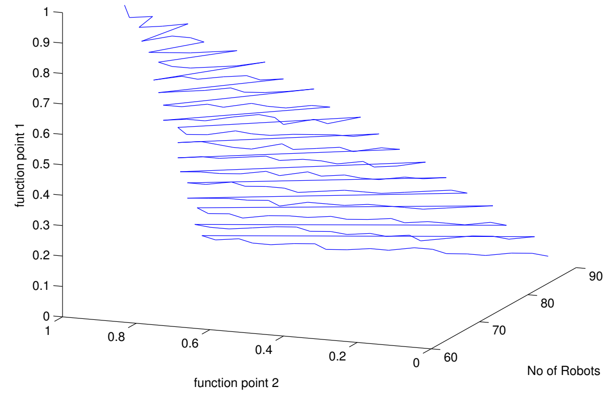


Fig. 10. Showing the no of robots localising with each data point.

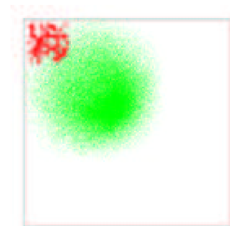


Fig. 11. Showing the pollutant profile.

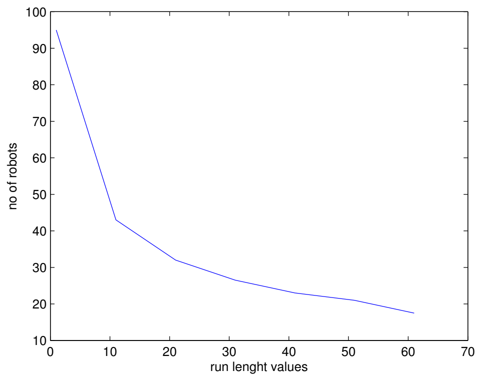
similar to that of an exponential function resulted in a greater number of robots at the source. However, the function having data values of 0.1 and 0.1 for data points 2 and 3 resulted in more agents localising than other points. This is shown in Fig. 9.Nevertheless, other data points might result in more exploration of the given environment. Investigation into this is still on going.

C. *A way of tuning the controller*

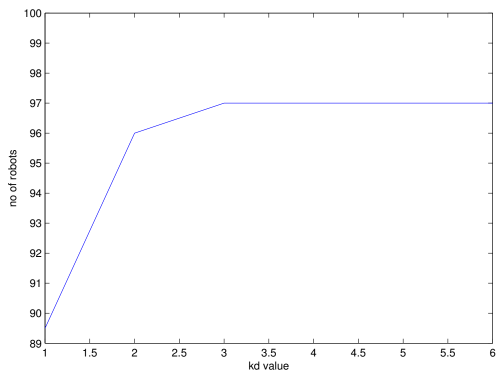
After studying the effects of the above parameters, we can come to a way of tuning the controller to achieve faster localization or exploration. Firstly, for the particular environment, the alpha value should be increased until there is no increase in the performance of the system. This is because based upon our work in [14], there is a saturation point over which there will be no dramatic increase in system performance. Then kd should be increased until there is no increase in system performance. Finally, the run length parameter τ_o can be increase or reduced to either achieve faster convergence or more exploration. This can be seen in Fig. 12.

V. Conclusion and Future Work

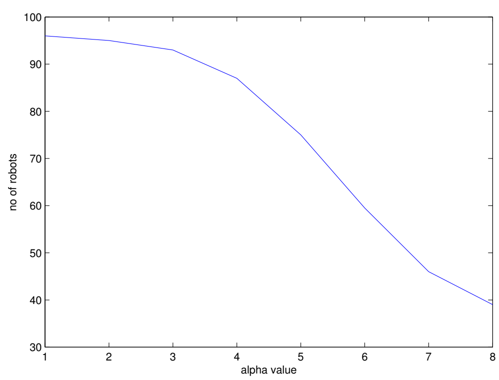
We have shown how to use a physical robot to achieve localization at a source using a bacteria controller. We have also shown how to use the controller to achieve either



(a)



(b)



(c)

Fig. 12. Number of Robots with adjusting: the run length τ_o -Fig. (a); the kd parameter-Fig. (b); the α parameter-Fig. (c)

more exploration or faster localisation. Both effects are contradicting and a tradeoff is needed. In other words, we can not achieve more exploration while performing to faster localisation.

Due to the gradient based method of searching, the bacterial controller might fail in a high turbulent environment. We believe that using this controller with a flocking controller performance might be improved due to the advantage of co operative foraging. We also believe that by investigating the parameters of the bacterial controller closely, we can adapt it to work in a high turbulent environment.

In future, we plan to use neural networks and adaptive learning to configure these parameters as every environment the robot encounters might be different from the last one it encountered. We also plan to use our algorithm for boundary detection of water pollution and subsequent efficient cleaning of the pollutant using multiple agents.

VI. Acknowledgement

This research has been financially supported by European Union FP7 program, ICT-231646, SHOAL.

References

[1] Y. Gabriely and E. Rimon, *Spanning-tree based coverage of continuous areas by a mobile robot*,Annals of Mathematics and Artificial Intelligence, 32:77-98, 2001.

[2] X. Zheng, S. Jain, S. Koenig, D. Kempe, *Multi-Robot Forest Coverage*, pp 3852-3857, IROS , August 2005.

[3] T. Balch, *The case for randomized search*, Workshop on Sensors and Motion, IEEE International Conference on Robotics and Automation, San Francisco, CA , May 2000.

[4] D. Gage, *Randomized search strategies with imperfect sensors*, pp270-279, Proc. SPIE Mobile Robots VIII, Boston, MA, Sep 1993.

[5] J. Cortes, S. Martinez, T. Karatas, F. Bullo , *Coverage Control for Mobile Sensing networks* , IEEE Transactions on Robotics and Automation, vol. 20, no. 2, pp. 243-255, April 2004.

[6] M. Schwager, F. Bullo, D. Skelly and D. Rus, *A Ladybug Exploration Strategy for Distributed Adaptive Coverage Control*, Proceddings of International Conference on Rbotics an Automation, Pasadena, CA, May 2008.

[7] B. Shucker, T. Murphey, and J.K. Bennett, *Convergence Preserving Switching for Topology Dependent Decentralized Systems*, IEEE Transactions on Robotics, Vol. 24, No. 6, December 2008.

[8] H.C. Berg and D.A. Brown, *Chemotaxis in Escherichia coli analysed by Three dimensional tracking*, Nature vol.239, October 1972.

[9] D.A. Brown and H.C. Berg, *Temporal Stimulation of Chemotaxis in Escherichia coli*, Proc. Nat. Acad, Sci. USA Vol. 71, No. 4, pp. 1388-1392, April 1974.

[10] K.M. Passino, *Biomimicry of bacterial Foraging for distributed Optimization and Control*, IEEE Control Systems Magazine, Vol.22, June 2002.

[11] J.O. Oyekan and H. Hu, *Toward Bacterial Swarm for Environmental Monitoring*, IEEE ICAL, pp. 399-404, Aug 2009.

[12] A.Dhariwal, G. S. Sukhatme and A.A. G. Requicha, *bacterium-inspired robots for Environmental Monitoring*, Proceedings of the 2004 IEEE International Conference on robotics and Automation, New Orleans, LA, April 2004.

[13] F. W. Dahlquist, R. A. Elwell, and P. S. Lovely, *Studies of bacterial chemotaxis in defined concentration gradients-A model for chemotaxis toward l-serine*, J. Supramolecular Structure, vol. 4, pp. 329(289)-342(302), 1976.

[14] J.O. Oyekan and H. Hu, *Exploiting Bacteria Swarms for Pollution Mapping*, Proceedings of the 2009 IEEE International Conference on Robotics and Biomimetics, pp. 39-44, December 19 -23, 2009, Guilin, China.