# Improving Bacteria Controller Efficiency

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

**Abstract**

We present a novel approach that would enable the placement of dynamic sensor platforms in the most optimal areas for data collection in an environment of any size. Our approach would ensure that more sensors are placed in areas that contain interesting data and less in areas with little or no data. In this paper, we use a bacteria controller to nagivate the environment in the search of interesting data and show that the addition of a flocking algorithm improves the chances of finding data.

**Keywords:** UAV, Bacterium Inspired Algorithm, Environmental Monitoring, Flocking.

## 1. Introduction:

It is often necessary to take measurements of an environmental variable (such as temperature) when studying a particular area. This is because it gives up to date information about the environment under study. Ways of doing this include the installation of static sensors in the environment or going from one location to another in a vehicle to take data from various locations as in biological field studies. However, these systems begin to become inefficient if the conditions in the environment changes rapidly over time from location to location. This would lead to misleading information as data from various locations can not be collected at the same time by using the latter method above.

The use of static sensors also has its drawbacks because of their immobility. The resolution and hence distribution of the sensors can not be adjusted so that areas with no information are avoided whilst areas with more interesting data are covered.

This leads to an efficiency in the use of sensors. In this paper we present the use of mobile sensor network that can configure itself based upon the data being measured so that more sensors are in regions of high concentration density and less in regions of low concentration density. Our approach uses a flocking algorithm to ensure system level control of agents with a bacteria algorithm to ensure approriate distribution of agents based on the density of the data at various locations. We focus on the reasons for using a flocking algorithm with a bacteria algorithm for providing coverage. In section two, we provide reasons for choosing this combination over other existing coverage algorithms. Section three briefly discusses our bacteria controller and flocking approach. Section four discusses our experimental setup, while section five discusses the results of our experiments. We conclude in section six by discussing future work.

## 2. Related Work:

In order to monitor an environmental variable, the intial thought might be to make sure that every single area in the environment is covered. This approach had to led to the development of various single 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 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 vertice 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 variable changes constantly over short periods of time. This leads to data collected being outdated quickly.

To solve the large environment size problem mentioned above, multi agents could be used. This has lead to a development of the spanning algorithm into a multi-spanning algorithm. This was used in [2] to find tree cover with similar tree weights. However, using this approach does address the second question properly. Areas in the environment that do not contain data are still covered and loss of data can take place if the agent is not near or in the same area as changes in data.

The above approaches are deterministic and hence ensure that most or every area in the en-

vironment to be monitored is covered. However, it has been proven that their performance approaches the performance of stotachistic approaches such as bacteria behaviour when the algorithms efficiency are reduced [3][4]. The reduction in efficiency of these algorithms could be as a result of wheel slippage of the robot, robot sensor inaccuracies and so on.

Additionaly, Balch in [3] was able to prove that using random search methods enable the use of less intelligent and expensive robots and hence are less costly. As a result of environmental noise, sensor noise and other inaccuracies in the environment and the platform being used, we believe that the use of a stochastic algorithm in a stochastically changing environment would approach the same level of efficiency of a deterministic algorithm. Such algorithm can deal with changes in the dynamically changing environment. To this end, we propose the use of a bacteria inspired controller. This controller is of a random biased walk nature and can be used to find the source of an environmental variable.

Furthermore, as mentioned previously, by using our approach, we aim to distribute the agents based on the concentration distribution of the environmental variable being monitored. This leads to more efficient and optimal use of a limited number of agents. Researchers have used various multi agent approaches to do this. This includes the use of Voronoi partitions [5] [6] and Virtual Spring Mesh approach[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 benefits of using flocking algorithms with a bacteria algorithm.

## 3. Bacterial Chemotaxis Behaviour

### *3.1. 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 $2.19 \pm 3.43$ s while if swimming down a gradient, the mean length is $1.40 \pm 1.88$s [8]. In other words, the length of the run phase is affected by the concentration of the attractant in the medium.

This was modelled by Berg and Brown as shown in Equations 1, 2 and 3.

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

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

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

where $\tau$ is the mean run time and $\tau_o$ is the mean run time in the absence of concentration gradients, $\alpha$ 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 $P_b$. $\frac{\overline{dP_b}}{dt}$ 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. $\tau_m$ 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 by throwing its flagellum clockwise in the medium. This makes it turn in a random direction $\sigma$. 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 [9].

### *3.2. Our Bacteria Algorithm*

In our simulations, we simulated a tumble behaviour by generating a random direction where the new direction is within the angle $\sigma \varepsilon \{0..., 360\}$. Values in the set $\sigma$ have an equal opportunity of being chosen. This made it possible for our robotic agent to investigate changes in the concentration gradient in any direction $\sigma$ measured from its present position. This would be particularly useful if the air pollutant direction changes suddenly due to wind direction changes. To simulate the changes in duration of the run length, we increased or reduced the rate at which calls to the tumble behaviour subroutine was made. The rate of these calls depend on the changes in the concentration gradient *G*. If climbing up a favorable gradient, we called the tumble behaviour

subroutine less and vice versa if climbing down the gradient. In other words, the tumble frequency or tumble period $\beta$ is a function of the concentration gradient *G*.

In our algorithm, the velocity $\nu$ of the robotic agent is changed depending on the changes in the concentration gradient *G*. If moving down a gradient, the velocity of the robotic agent is reduced, while if moving up a gradient the velocity is increased to aid faster convergence at the optimal point. In the presence of no concentration gradient, the robot is moved at a fast velocity so that we can find particles of the attractant faster. If the present measured concentration is greater than a threshold value, it is assumed that the robotic agent has found the source and it stops there. The above could be modeled as follows:

$$\tau(G) = \beta(G) * v(G) \tag{4}$$

where the tumble period $\beta$, velocity *v* and run length $\tau$ are all functions of the measured concentration gradient *G*. We also assumed that the robotic agent is a single point mass kinematic model. As a result of using a single point mass kinematic model, we can model our algorithm as follows:

$$x(t+1) = x(t) + \nu(t+1)\cos\sigma \tag{5}$$

$$y(t+1) = y(t) + \nu(t+1)\sin\sigma \tag{6}$$

where *x* and *y* are points in the cartesian plane, *v(t + 1)* is the velocity at the next time step. This is dependent on the measured concentration at (*x*, *y*, *t*). *v(t + 1)* is a set of velocities *V* and is defined as:

$$v(t+1) = \begin{cases} V_T \; iff \; C(t) > 14 \\ V_N \; iff \; G = -ve \\ V_P \; iff \; G = +ve \\ V_O \; iff \; G = 0 \end{cases} \tag{7}$$

$$G = C(t+1) - C(t) \tag{8}$$

where *C* is the currently measured concentration; *G* is the difference between the previous measured concentration value and the present concentration value. In our experiments, we chose $V_T = 0$, $V_N = 1$, $V_P = 4$ and $V_O = 3$ and compared it with the result of when $V_T = 0$, $V_N = V_P = V_O = 3$. Our aim was to investigate the effects of having different velocity values at different values of *G* and compare to having the same velocity values regardless the value of *G*. In addition, the velocity *v(t + 1)* was not affected by the previous velocity value there by following the Dahlquist et al model. For the tumble period, a set of time periods as defined below was used:

$$\beta(t+1) = \begin{cases} \beta_T \; iff \; C(t) > 14 \\ \beta_N \; iff \; \Delta G = -ve \\ \beta_P \; iff \; \Delta G = +ve \\ \beta_O \; iff \; \Delta G = 0 \end{cases} \tag{9}$$

where we chose $\beta_T = 0$, $\beta_N = 10$, $\beta_P = 100$ and $\beta_O = 5$. From our work in [11], we know that it is possible to use the bacteria controller defined above to find the source of an environmental variable.

### *3.3. Flocking Controller*

To implement the flocking controller, we used a modified flocking controller as shown in Equation 10. This controller was developed for single point mass kinematic models.

$$x^f = [-k(dist - d)](x^i - x^j) + [h(x^i - x^h)] \tag{10}$$

$$dist = ||x^i - x^j|| \tag{11}$$

where $x^h$ is the position of the agent with the highest reading in the neighbourhood, $dist = \sqrt{(x^i - x^j)^T(x^i - x^j)}, k > 0$, is the magnitude of the repulsion force between agents $x^i$ and

$x^j$. Constant $l$ is the attractant gain for the force between agent $x^i$ and the agent in the neighbourhood with the highest measurement, $x^h$. We then combined the two behaviours and calculate the new position of the agent $x^i$ using the equation 12 below.

$$x^i = f * x^f + b * x^b \tag{12}$$

where constants f and b are gains for each behaviour. In our investigations, we set them both to one.

## 4. Experimental Setup

To test the algorithm, a simulated arena that had a dimension of 600 pixels by 600 pixels was developed. This is shown in Fig. 1. We used kinematic models for the simulated robots as mentioned previously. The simulated robots had dimensions of 10 pixels by 10 pixels and had an array of simulated chemical sensors in the center of the robot. This array of chemical sensors had a dimension of 4 pixels by 4 pixels. It is assumed that each individual chemical sensor making up the chemical sensor array returns 1 or 0 as output. If a chemical sensor detects a pollutant particle in a location it returns a 1 and it returns a 0 otherwise. To measure the concentration of the pollutant in the robot's position, the values of each chemical sensor in the array is added up to get the total measured concentration in that location.

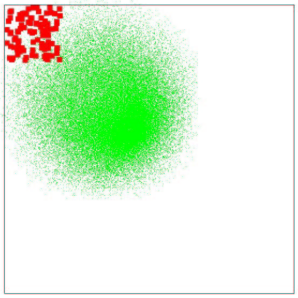

Figure 1: Simulation Setup showing robots and pollutant.

In this experiment, the 2D problem of visually producing a map of the pollutant is investigated before working on the 3D problem as in [7]. Other assumptions taken in our experiment include:

- A simple generated air pollutant is used with randomly generated particles. By generating the particles randomly, there were no clear concentration gradient boundaries.
- The air pollutant is stationary and not moving in the experiments. The effects of wind changes or air convection currents are not investigated but will be done in later studies.
- It is assumed that robots are single point kinematic models based on our previous work in [10] where the control system developed is quite robust. This control system can take co ordinate values from a user and then fly the UAV to the desired position. In this experiment, a large number of robots is needed because of the number needed to visually produce a map of the pollutant. This is possible in real life because of the reducing costs of hardware.
- The robots were placed initially at the edges of the pollutant as in [9] and [12].
- Each robot knows its position in the simulated arena.
- No collision avoidance was used in the simulated experiments. On real robots, proximity sensors such as ultrasonic sensors could be used to detect the presence of other robots and then take action accordingly.
- It is assumed that once the robots find a concentration value greater than a threshold of 14 they have reached the source.
- It is assumed that the chemical sensors used in the experiment were noiseless during the simulation. We plan to introduce noise through the use of a random number generator in subsequent experiments.

## 5. Simulation and Results

In the experiment, a simple air pollutant is simulated at the position 300 pixel by 300 pixels as shown in Fig. 1. A population of 100 robots is randomly placed within an area of 60 pixels by 60 pixels. We developed two test metrics that were used to evaluate our approach. For our first test metric, we aimed to investigate how many robots would find the source within a time frame (Localization Ability). In our second test metric, we investigate the effects that flocking

would have on the amount of time agents spent outside the environmental variable being monitored (Foraging Ability).

For each experiment, the bacteria controller was first ran for various time intervals of 30, 90 and 120 seconds. Then the flocking controller was then used in combination with the bacteria controller and ran for the same time frames mentioned above.

### *5.1. Localization Ability*

As discussed above, we ran both algorithms and obtained the results as shown in Fig. 2

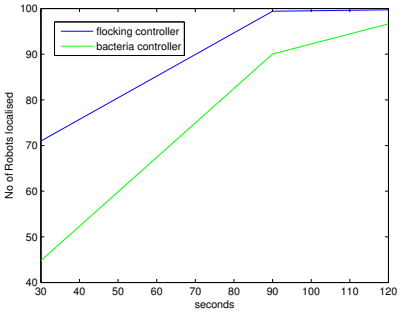


Figure 2: Comparison of Localization Ability when using the Bacteria Controller only and when using it with the flocking controller.

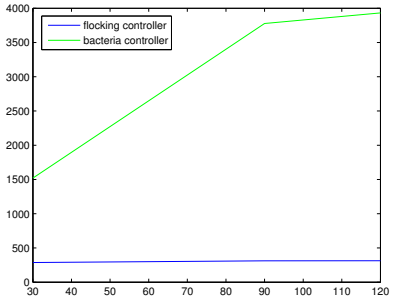


Figure 3: Comparison of Foraging Ability when using the Bacteria Controller only and when using it with the flocking controller.

As can be seen in Fig. 2 the flocking controller when used with the bacteria controller performed better than using only the bacteria controller for short time periods. This property would be useful if a hazardous substance is to be found quickly before more damage is done.

*5.2. Foraging Ability*

Similarly, we ran the bacteria algorithm and compared the results with the results obtained for using the foraging algorithm with the bacteria algorithm and observed the amount of time the agents spent outside the environmental variable. These results were obtained by counting the number of time the chemical sensors on the simulated agents gave zero readings (zero count). The results are shown in Fig. 3.

As can be seen in the results shown in Fig. 3, the agents spent less time outside the environmental variable when using the flocking controller. In addition, it can be seen that the zero counts of the bacteria flock did not increase significantly has the time frame was increased from 90 to 120 seconds.

## 6. Conclusion and Future work:

In this paper, we have shown that it might be possible to use a stocastic algorithm such as the bacteria controller with a flocking controller to provide coverage of an environment. we have also shown that the use of flocking algorithm enable the faster convergence of agents at the source of an environmental variable. This is analogous to natural systems in which group of individual with large numbers are more successfully than groups with smaller numbers or lone agents. We have also shown that the use of flocking algorithms enabled agents to spend more time in areas of environmental variable than in areas lacking the environmental variable. This makes it possible to get more interesting readings from the environment.

In the future, we plan to continue work to investigate how to fuse the flocking behaviour and bacteria behaviour together to provide an effective coverage of an environment. We also aim to use the Berg and Brown model for the bacteria chemotactic behaviour to firstly control a single agent and then use it to collectively control a flock of agents. Furthermore, we plan to use fuzzy logic to fuse the inputs of the flocking algorithm and the bacteria algorithm together to obtain a more smoother behaviour.

## 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 (London) 239:500-504, 1972.
[9] 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.
[10] J. Oyekan and H. Hu, *Towards Autonomous Petrol Behaviours for UAVs*, Proceedings of UK EPSRC Workshop on Human Adaptive Mechatronics, Staffordshire University, Stafford, U.K., 15-16 January 2009.
[11] J. Oyekan and H. Hu, *TToward Bacterial Swarm for Environmental Monitoring*, IEEE ICAL, August 2009.
[12] S.D. Muller, J. Marchetto, S. airaghi, and P. Koumoutsakos, *Optimization Based on bacterial Chemotaxis*, IEEE Transactions on Evolutionary Computation, Vol.6, No.1, February 2002.