

Toward Bacterial Swarm for Environmental Monitoring

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

Abstract- Nature inspires many interesting ideas of solving our day to day problems. In this paper, we take inspiration from the chemotaxis behaviour of the bacterium and create a novel algorithm. Simulated results show the feasibility of using a bacterial inspired algorithm in finding a pollution source and then generating a visual 3D representation of the pollutant. We believe that by providing a visual 3D representation of an air pollutant for example, it could lead to an effective, timely and systematic evacuation of humans in a densely populated urban area.

Index Terms- UAV, Bacterium Inspired Algorithm, Environmental Monitoring, Flocking.

I. INTRODUCTION

Nature provides us with many interesting ideas and ways of solving our day to day problems and how to improve our quality of life. Nature's ways of solving problems are often based on simple but robust and effective rules. Take for example, a flock of starlings. These birds operate on simple rules of keeping distance, avoiding collisions and following a general direction. These simple rules produce an amazing flocking behaviour.

This has led to a range of biological inspired algorithms. Reynolds for example, used the simple rules mentioned above to develop his boids that closely mimic the flocking behaviour of the starlings [1]. Other researchers have developed various forms of flocking algorithms. Some use artificial forces to keep flocking members in formation, while some use fuzzy logic to combine the three simple rules mentioned above to produce a flocking effect in Multi-agent robotic systems [2]- [4].

In this paper, we take inspiration from the bacterium chemotaxis behaviour in order to solve the problem of keeping track, mapping and visually producing a 3D representation of an air pollutant using a swarm of Unmanned Aerial Vehicles (UAVs). This would be particularly useful in situations where the pollutant is hazardous and is invisible to human eyes. This could enable the efficient evacuation of areas in the immediate path of the air pollutant in a city in contrast to evacuating the whole city. In addition, the use of a swarm of UAVs could also aid the better understanding of air pollutants and how they disperse.

Swarm robotic systems or using a large number of simple individual robotic systems have previously been used to track pollutants. In [5], Susca et al. discusses of using a number of robots to produce an estimate of the boundaries of a moving pollutant. However, they made an assumption that the pollutant is moving slowly and that the agents have an estimate of the boundary at the initial time.

Susca et al. also assumed that each agent can estimate the tangent and curvature of the boundary. However, an air pollutant could have a complex nature due to wind speeds and air convection effects [6]. As a result, the estimates of the curvature and tangent of the air pollutant might be difficult to acquire even if there were sensors to do this. A bacterial inspired algorithm does not need such information to detect the boundaries of the air pollutant as can be seen in [7]. In this research, it is assumed that the robotic agents being deployed do not know where the exact location of the pollutant source is, and they are able to produce a visual 3D representation of the air pollutant and its concentration at various places.

The rest of the paper is organised as follows. Section II reviews some previous work related to the use of bacterial chemotaxis behaviours. In Section III, the bacterial chemotaxis model and its application are discussed. Section IV explains the simulation set up, and Section V presents some initial experimental results. Finally, a brief conclusion and future work are presented in Section VI.

II. RELATED WORK

The bacterial chemotaxis behaviour is a simple behaviour but it is very effective in locating food sources. There have been many studies on the chemotaxis behavior of the bacteria both in the robotic field and in the biological field [7], [8]- [12]. The effectiveness of this behaviour in locating food sources has been realized by researchers and converted into bacterium inspired algorithms for various uses.

Muller et al. [9] presents the parameter optimization of the basic bacterial chemotaxis model proposed by Dahlquist et al. [13] using evolutionary concepts. This is to mimic nature in finding solutions to problems. In [9], the problem was finding solutions to multimodal functions.

Passino simulated and studied the behavior of bacteria foraging and swarming for use in distributed optimization and control [8]. The biological concepts of elimination and dispersal, reproduction and death were also used in the study to develop

an optimization algorithm that would escape local minimum in the search landscape.

Dhariwal et al. used the bacterium inspired algorithm to study how it can be used for environmental monitoring [7]. They investigated how to use the biased random walk property of the bacterium inspired algorithm to reduce the amount of time spent on searching for the optimal of a unimodal function. By the use of bias, Dhariwal et al were able to detect multiple sources and dissipative sources. Their algorithms were robust against sensor and actuation noise and also better at boundary coverage than a gradient descent algorithm. However, in their work, the simulated pollutant had uniform concentration bands which are a rare occurrence in an air pollutant in nature. This is because fumes from the pollutant are affected by random changes in the environment such as wind, air convection currents caused by temperature among other factors. Furthermore, in the microscopic world, a bacterium would experience random changes in concentration as a result of Brownian motions.

Furthermore, in the investigation done by Dhariwal et al, the sensor readings were not taken every time step. This might account for the reason why it took a long time to reach convergence in their experiments. If the robot was heading in a wrong direction, it would not know this until the time for the next sensor reading. It would then have to backtrack the distance covered in the wrong direction in the previous time frame. If instantaneous readings were made each time and used, the robot would be capable of making a decision immediately resulting in faster convergence. In our experiments, we followed the approach of taking sensor readings every time step and acting accordingly. We also focused on adjusting the two parameters of velocity and tumble frequency to obtain fast convergence at an optimal location.

III. BACTERIAL CHEMOTAXIS BEHAVIOUR

A. Bacterial Chemotaxis Model

A bacterium finds food sources by climbing up a favourable concentration gradient. Its motion is made up of two phases namely a run phase and a tumble phase [14]. The run phase can be said to be a straight line motion in a particular direction. But, it should be noted that in nature, due to the Brownian motion of the medium in which the bacterium is immersed; this is not really a straight line motion. 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 ± 1.88 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 Jackson [15] as follows:

$$P_t = \frac{\Delta t}{\tau} \quad - (1)$$

Where, P_t is the probability that a tumble will occur within Δt ; τ is the run time and is determined by:

$$\tau = \tau_0 \exp\left(\alpha \frac{dP_b}{dt}\right) \quad - (2)$$

$$\frac{dP_b}{dt} = \tau_m^{-1} \int_{-\infty}^t \frac{dP_b}{dt'} \exp\left(-\frac{(t-t')}{\tau_m}\right) dt' \quad - (3)$$

$$\frac{dP_b}{dt} = \frac{K_D}{(K_D + C)^2} \frac{dC}{dt} \quad - (4)$$

$$P_{tb} = C / (K_D + C) \quad - (5)$$

Where τ_m^{-1} is a time constant dependent on the bacterial system or type of bacterium. τ_0 is the mean run length in the absence of a concentration gradient, 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 . K_D is the dissociation constant of the bacterial chemoreceptor. $\frac{dP_b}{dt}$ is the rate of change of P_b .

While $\frac{dP_b}{dt}$ is the weighted rate of change of P_b [15]. The tumble phase is performed by the bacteria by 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 [9]. Muller et al model this in their experiments as a Gaussian distribution for both the right and left directions. Furthermore, in the Dahlquist et al model, the velocity v is assumed to be constant.

B. Our Algorithm

In our simulations, we simulated tumble behaviour by generating a random direction where the new direction is within the angle $\sigma \in [0, \dots, 360]$. Values in the set σ 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 σ 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 β is a function of the concentration gradient G .

In our algorithm, the velocity v 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 modelled as follows:

$$\tau(G) = \beta(G) * v(G) \quad - (6)$$

where the tumble period β , velocity v and run length τ are all functions of the measured concentration gradient G .

We also assumed that the robotic agent is a single point mass kinematic model. This was based on our previous work in [17]. As a result of using a single point mass kinematic model, we can model our algorithm as follows:

$$x(t+1) = x(t) + v(t+1) \cos \sigma \quad - (7)$$

$$y(t+1) = y(t) + v(t+1) \sin \sigma \quad - (8)$$

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:

$$v(t+1) = \begin{cases} V_T & \text{iff } C(t) > 14 \\ V_N & \text{iff } G = -ve \\ V_P & \text{iff } G = +ve \\ V_G & \text{iff } G = 0 \end{cases} \quad - (9)$$

$$G = C(t+1) - C(t) \quad - (10)$$

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 $V_G=3$ and compared it with the result of when $V_T=0$, $V_N=V_P=V_G=3$. 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:

$$\beta(t+1) = \begin{cases} \beta_T & \text{iff } C(t) > 14 \\ \beta_N & \text{iff } G = -ve \\ \beta_P & \text{iff } G = +ve \\ \beta_G & \text{iff } G = 0 \end{cases} \quad - (11)$$

where we chose $\beta_T=0$, $\beta_N=10$, $\beta_P=100$ and $\beta_G=5$.

A bacterium decision at time t is affected by the concentration of the attractant measured during the previous three seconds. In other words, a bacterium has memory. As a result, a 4-element memory was introduced into our algorithm. It was used to remember the last four positions of the bacterium.

In the event it detects a zero pollutant concentration reading, it goes through the last four positions and looks for the best position it was at previously and then heads in the direction of the position.

It became clear that the introduction of the 4-element memory caused the robot to be more biased by remaining in areas in which the pollutant was present than the areas that had no pollutant. This is analogous to nature in the sense that animals would stay in an environment that has more food than an environment that is less food. This introduced 4-element memory parameter could be used in bacterial inspired algorithms to keep searches for an optimal to certain local regions.

The introduction of this 4-element memory also led to a higher probability of finding the source of the air pollutant. Some of our robots strayed away into areas containing no pollutant. However, the number of robots that strayed away when using our algorithm was not as much as the number of robots that strayed away when using the original bacterium inspired algorithm.

IV. EXPERIMENTAL SETUP

To test the algorithm, a simulated arena that had a dimension of 600 pixels by 600 pixels was developed. 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 centre 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.

In this experiment, the 2D problem of visually producing a map of the pollutant is investigated before working on the 3D problem as in [9]. We also made sure that we ran both algorithms on the same generated air pollutant so has to get a fair comparison of both algorithms. Algorithm 1 shows our modified bacterial algorithm.

Algorithm 1: Our Modified bacterial Algorithm.

```

1: while not stop do
2:   if counter > MEMORYSIZE then
3:     reset counter
4:   end if
5:   if PresentReading > PreviousReading then
6:     Reduce Tumbling Frequency
7:     Set Velocity to four units every time step
8:   else if PresentReading < PreviousReading then
9:     Increase Tumbling Frequency

```

```

10:         Set Velocity to one unit every time step
11: else if PresentReading >ThresholdValue then
12:     Stop
13: else if No Gradient then
14:     Increase Tumbling Frequency
15:     Set Velocity to three units every time step
16: end if
17: PreviousReading = PresentReading
18:     Store co ordinates in Memory at location counter
19:     Store Present Reading in Memory at location counter
20: increment counter
21: end while

```

- A simple generated air pollutant is used with randomly generated particles. By generating the particles randomly, there were no clear concentration gradient boundaries. In addition, 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 [17] 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 [7] and [9].
- 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.

V. 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.

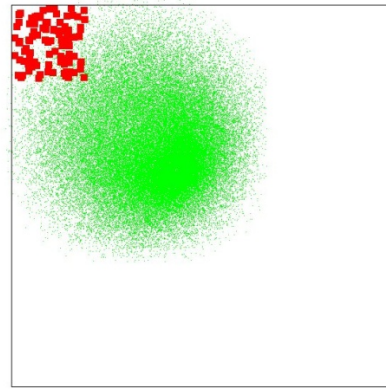


Figure 1. Simulation Setup showing robots and pollutant.

A. Optimizing the number of robots that find the source (Without Memory)

As mentioned in Section III-B, we ran our algorithm using different velocity parameters (algorithm A) and compared it with using the same velocity parameters (algorithm B). We ran each algorithm 40 times for a population of 100 robots to get a good distribution for our results. The results show that in a short time of 30 seconds, more robots were able to find the source using different velocity parameters than when using the same velocity. This is shown in Fig. 2.

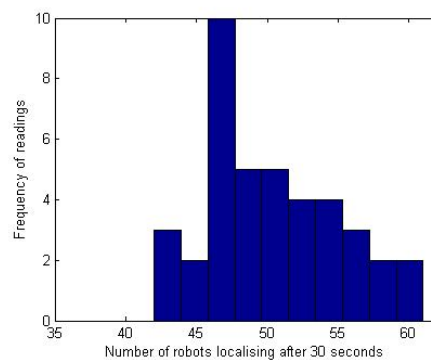
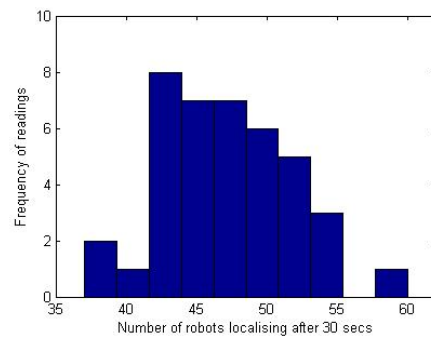


Figure 2 The distribution of Number of robots using Original Algorithm B- Fig. (a) and using our modified Algorithm A-Fig. (b)

As the results were non-parametric, a rank sum statistical test (equivalent to a utest [16]) was carried out to test if there was any significant difference between the algorithms. The result from the statistical test is that the medians of algorithm B and algorithm A were 47 and 49.5 respectively ($U = 1362$, $n_1 = n_2 = 40$, $P < 0.05$ two-tailed). The statistical result showed that the algorithms differed significantly. The results proved that algorithm A had a better performance in finding the pollution source than algorithm B. We also decided to run both algorithms for longer periods of time to investigate if both would achieve the same performance if given adequate time. Our results for 30 seconds, 60 seconds, 90 seconds and 120 seconds are shown in Table I.

From Table I, we discovered that when both algorithms were given adequate time, the performance of algorithm A was still better than the algorithm B. Since we envisage the developed system being deployed to find hazardous substances, we believe that time is of essence when doing this task. As a result, algorithm A would have an advantage over algorithm B.

TABLE I
DISTRIBUTIONS OF ROBOTS AT VARIOUS TIMES

Time (secs)	Algorithm B (Median)	Algorithm A (Median)	U	P-value (two-tailed)
30	47.0	49.5	1362	0.0130
60	78.5	83.0	1062	7.3916 e-008
90	90.0	93.0	1985	8.5047 e-005
120	96.0	98.0	1213	3.7919 e-004

B. Optimizing the time spent by each robot in the pollutant (Using Memory)

As mentioned previously, a 4-element memory is used to remember the previous best positions. To test the effectiveness of introducing this, we paid attention to the amount of times the robots were outside the pollutant. We did this by counting how many times each robot's chemical sensor array measured zero. By adding the amount of occurrences each robot's chemical sensor detected zero, we were able to get a measure of the performance of both algorithms. As algorithm A was better than algorithm B at finding pollution sources, we introduced the memory to algorithm A. We called this algorithm C. Then the results of algorithm B were compared with the results of algorithm C.

Firstly, the algorithms were ran for 30 seconds and it was discovered that the amount of time the robots stayed out of the pollutant is less for algorithm C than for algorithm B as shown in Fig. 3.

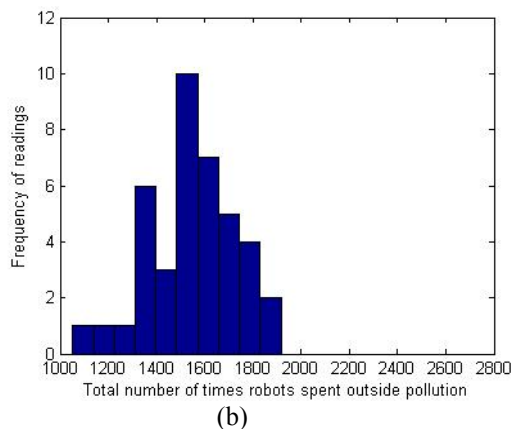
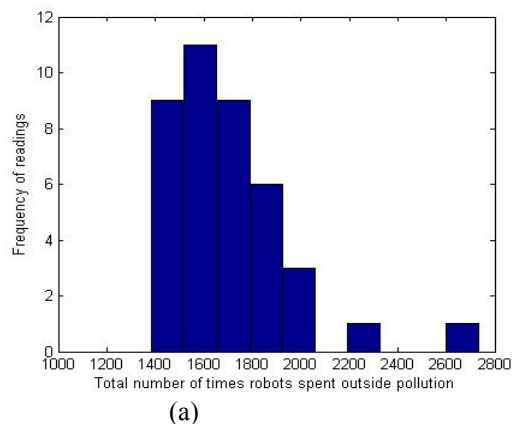


Figure 3. The distribution of Number of times the robots stayed outside the pollutant using Original algorithm B- Fig. (a) and using Our algorithm C- Fig. (b)

A rank sum statistical test was used again to test if there was any significant difference between both algorithms. The result from the statistical test was that the medians of algorithm B and algorithm C were 1651 and 1539.5 respectively ($U = 1910$, $n_1 = n_2 = 40$, $P < 0.05$ two-tailed) with a P-Value = 0.0053 for 30 seconds. The algorithms were also tested for 60, 90 and 120 seconds as shown in Table II. These results proved that algorithm A in combination with a 4-element memory was efficient in keeping a large number of robots in the pollutant area.

TABLE II
QUANTITY OF TIME SPENT OUTSIDE POLLUTANT

Time (secs)	Algorithm B (Median)	Algorithm C (Median)	U	P-value (two-tailed)
30	1651	1539.5	1910	0.0053
60	2734.5	2135.5	2295	8.5556 e-011
90	3597.5	2552.5	2353	1.8077 e-012
120	4000	2385	2385	1.8887 e-013

One of the discovered unique features of our algorithm is that even though it kept majority of the robots in the pollutant some of the robots still left the pollutant in search of other sources. We plan to use this behaviour to find multiple sources in an environment and then subsequently map them also.

VI. CONCLUSION AND FUTURE WORK

This paper presents a novel bacteria inspired algorithm for finding a pollution source under the assumption that the air pollutant is stationary, and no wind effects and heated air convection effects are taken into account. It is clear from experiments that faster convergence is achieved by using a variable dynamic velocity term instead of using a constant velocity term. By using a 4-element memory, it is able to keep more robots in the pollutant than when compared with an algorithm that has no memory element. This feature is desirable especially when tracking a moving air pollutant. The proposed algorithm is capable of ascending the gradient of a simple air pollutant which was simulated by using randomly generated particles leading to an air pollutant with no clear concentration bands. In the future, various pollutant distributions will be taken into consideration during the investigation. It is also important to discover if BCA parameters could be dynamically modified at run time to make it more effective in discovering a dynamic pollution source and concentration distribution. Since the value of the parameters used in this research were chosen after trial and error, it is necessary to use a formal approach to tune these parameters in the next stage. This could involve using some search strategies such as exhaustive searches and evolutionary approaches. Introducing flocking algorithms with BCA would enable fast convergence to the pollution source as a result of co-operative foraging, which makes the visual mapping of the air pollutant easier. Moreover, as nano-technology advances, this algorithm could be used on a flock of nano robots to visually map either temperature, pollution or any other desired parameters. This could be especially useful for the military use of mapping an invisible chemical or biological threat.

ACKNOWLEDGEMENTS

We would like to thank Dr Dongbing Gu for all his contribution on flocking algorithms and systems.

REFERENCES

[1] C. W. Reynolds, *Flocks, herds, and schools: a distributed behavioural model*, Computer Graphics, vol. 21, pp. 25-34, 1987.
 [2] D.Gu and H.Hu, *Using Fuzzy Logic to Design Separation Function in Flocking algorithms*, IEEE Transactions on fuzzy Systems, vol 16, No.4, August 2008.
 [3] M.Sisto and D.Gu, *A Fuzzy Leader-Follower Approach to Formation Control of Multiple Mobile robots*, Department of Computer Science, University of Essex, England.
 [4] Z.Wang and D.Gu, *A Local Sensor Based Leader-Follower Flocking System*, IEEE Int. Conf. on robotics and Automation, Pasadena, CA. May 19-23, 2008.
 [5] S.Susca, S.Martinez and F.Bullo, *Monitoring Environmental Boundaries with a robotic Sensor network*, Centre for Control, Dynamical Systems and

Computation, University of California at Santa Barbara, Santa Barbara, CA 93106.
 [6] C.Barcken, A.Carnemolla, C.Ritter, E.Zielke, *An Analysis of Exhaust Emissions from a Large Ship Docked in Humboldt Bay*, ENGR 416- Transport Phenomena, May 2007.
 [7] 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.
 [8] K.M. Passino, *Biomimicry of bacterial Foraging for distributed Optimization and Control*, IEEE Control Systems Magazine, Vol.22, June 2002.
 [9] 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.
 [10] S.M. Block, J.E. Segal and H.C. Berg, *Adaptation Kinetics in bacterial Chemotaxis*, Journal of Bacteriology, April 1983.
 [11] S.M. Block, J.E. Segal and H.C. Berg, *Impulse Response in bacterial Chemotaxis*, Division of Biology, California Institute of Technology, Pasadena, California 91125.
 [12] M.Wahde, *bacterial Animal Behaviour*, 2007.
 [13] F. W. Dahlquist, R. A. Elwell, and P. S. Lovely, *Studies of bacterial chemotaxis in de ned concentration gradients-A model for chemotaxis toward l-serine*, J. Supramolecular Structure, vol. 4, pp. 329(289)-342(302), 1976.
 [14] H.C Berg and D. A. Brown, *Chemotaxis in Escherichia coli analysed by three-dimensional tracking*. Nature (London) 239:500-504, 1972.
 [15] G. A. Jackson, *Simulating chemosensory responses of marine microorganisms*. Limnol. Oceanog. 32:12531266, 1987.
 [16] J. Segall, S. Block, and H. Berg, *Temporal comparisons in bacterial chemotaxis*, Proc. Nat. Acad. Sci., Vol. 83, pp. 8987- 8991, Dec. 1986.
 [17] 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.
 [18] The MathWorks- statistics Tool box, *Ranksum*, www.Mathworks.com , 11 February 2009