An Immuno-engineering Approach for Anomaly Detection in Swarm Robotics

HuiKeng Lau[1,3], Iain Bate[1], and Jon Timmis[1,2]

[1] Department of Computer Science, University of York
[2] Department of Electronics, University of York,
Heslington, YO10 5DD, UK
[3] School of Engineering and IT, Universiti Malaysia Sabah, Malaysia

Abstract. In this paper, we present the first stage of our research strategy to develop an immune-inspired solution for detecting anomalies in a foraging swarm robotic system with an immuno-engineering approach. Within immuno-engineering, the initial stage of our research involves the understanding of problem domain, namely anomaly detection, in a foraging swarm robotic system deployed in dynamic environments. We present a systematically derived set of activities for this stage derived with Goal Structuring Notation and results of experiments carried out to establish the time-varying behaviour and how anomalies manifest themselves. Our future work will then be used to select and tailor an appropriate AIS algorithm to provide an effective and efficient means of anomaly detection.

Keywords: Immuno-engineering, swarm robotics, foraging.

1 Introduction

Research in swarm robotics has seen an increase in recent years due to its huge potential in various applications from medical, industrial, civilian and military to deep water and space exploration. A large amount of research funding has been awarded, for example a recent project called the SYMBRION project [10] supported by the European Commission. In a swarm robotic system (SRS), the achievement of a collective task is through the coordination of a group of simple homogeneous swarm that interact among themselves and with the environment [11]. Research into swarm robotics emerges from the inspiration of system-level functioning of social insects (ants, wasps, termites) that demonstrates the characteristics of robustness, flexibility, and scalability [2]. Insect communities are able to survive even after losing large numbers of insects (workers) or when faced with environment changes. In addition, they are very flexible in that they are able to solve foraging, prey retrieval and chain formation problems with the same base self-organised mechanism [2]. The increase or reduction in the number of individuals seem not to have severe impact on the operation of the community [2]. And thus, our intention is to attain these same desirable properties for the system-level operation of a SRS [12]. To distinguish swarm robotics from other terms being used to describes different approaches used in multi-robot systems,

P.S. Andrews et al. (Eds.): ICARIS 2009, LNCS 5666, pp. 136–150, 2009.

Sahin *et al* [12] outlines four distinguishing characteristics of swarm robotics that are considered representative for the purposes of our research. These characteristics, quoted verbatim, are:

- **Coordination of swarm:** Coordination mechanisms should be scalable to various swarm sizes.
- **Homogeneous robot:** Robots should be identical, at least at the level of interactions. Heterogeneous multi-robot systems fall outside of the swarm robotics approach.
- **Simplicity:** Robot should be simple in capabilities in relative to the task (not necessary hardware or software complexity).
- **Local interaction:** Individuals should have local interaction to ensure distributed coordination and scalability.

Swarm robotics is essentially an instance of distributed autonomous systems where a collection of interacting entities work together to accomplish certain goals without centralised control [13]. Each entity interacts with its environment and other entities through local communication but acts with some degree of autonomy [9]. For any system, given a reasonably well understood operational environment it is bound to experience undesirable behaviour or anomalies. These undesirable behaviours can be caused by random errors in hardware components, design errors, or deliberate sabotage [17]. To ensure normal operation of a system, detecting abnormal behaviour or malicious activities is very important. In current swarm robotics research, anomalies are normally handled through redundancy of robots that results in fault tolerance and no detection is carried out [17]. Such an approach assumes that anomalies occur at a rate that will not prevent the completion of a collective task and that individual robots with anomalies do not have an undesirable effect on the behaviour of the overall swarm. In this work we address situations where these assumptions may not hold. In such cases, detection of anomaly may help in deciding whether to continue with current task or switch to another task. Most work in SRS usually considered normal behaviour of the whole swarm. However as shown by [3] and [6], the issue of anomaly detection is an area that is starting to receive more attention from researchers.

There are significant challenges in detecting anomalies in swarm robotics in particular those deployed in dynamic environments. Due to the nature of SRSs in dynamic environments, anomaly detection systems (ADS) for such systems must fulfil the requirements of accuracy, responsiveness, resource usage and robustness [8]. With a lack of support from existing ADS to the above mentioned requirements, we decided to look at biological distributed autonomous systems that operates in analogous environment for inspiration. This is also in part influenced by research into artificial immune systems (AIS) that has showed that the immune system exhibits the properties we wish to endow the ADS in swarm robotics with. In approaching this problem, a systematic and principled approach of developing AIS called immuno-engineering [16] is adopted. One lesson from the thesis of Andrews [1] is that a firm understanding of the problem domain is needed before developing an immune inspired solution, therefore we focus our initial efforts at a detailed understanding of the domain first: our findings are reported in this paper. The novel contribution of this paper is considered to be showing how the initial stage of immuno-engineering in *understanding the problem domain* [4] can be systematically derived through empirical experimental means, i.e., the nature of anomalies, and the time-varying nature of the environments. Specifically our work addresses the problem of *Modelling of Information Processing* described in [16].

Section 2 introduces immuno-engineering and how an instantiation of immuno-engineering is carried out in this research with research plan derived using Goal Structuring Notation (GSN) [7]. Section 3 and section 4 report the set of experiments carried out to establish the time-varying behaviour (TVB) and anomalies in a foraging SRS. Section 5 is the conclusion.

2 Immuno-engineering

The concept of immuno-engineering was proposed as an approach to develop biologically grounded and theoretically understood AIS through an interdisciplinary collaboration to capture the richness offered by immune system as opposed to weak analogy of the immune process they are based [16]. Immuno-engineering is defined in [16] as

> *The abstraction of immuno-ecological and immuno-informatics principles, and their adaptation and application to engineered artefacts (comprising hardware and software), so as to provide these artefacts with properties analogous to those provided to organisms by their natural immune systems.*

Immuno-engineering involves the adoption of conceptual framework [15] in AIS algorithm development and the problem-oriented perspective [4] in developing engineered AIS solutions. Thus an understanding of a problem domain is crucial in determining the type of AIS solution to develop. With a problem-oriented perspective in mind, the first stage of this research involves the investigation on the TVB and anomalies in a foraging SRS and how these behaviour exhibit themselves. In this research, GSN [7] is adopted to derive the set of activities required to answer the research questions. The advantage and motivation is that problem derivation with GSN allows the research problem to be successively broken down into smaller goals or objectives to the level where they are supported by the defined set of activities or solutions. This allows us to systematically derive research questions to be answered until we reach a level of detail that clearly derives a set of experiments, and associated experimental conditions, that need to be performed. Worth mentioning here that GSN can be used for any problem and not only in the context of AIS. Figure 2 shows the set of activities derived with GSN to establish the TVB and anomalies in a foraging SRS.

2.1 Overview of GSN

GSN was originally derived for use in the production of safety cases as part of the certification of systems. During the establishment of the safety argument's

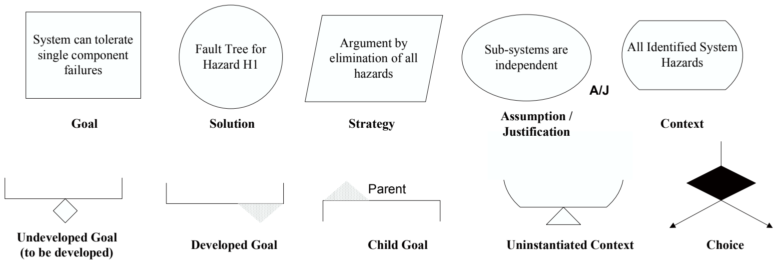

Fig. 1. Principal Elements of the Goal Structuring Notation

claims (often referred to as goals), context, assumptions and justifications are captured which has a number of uses including managing change. The GSN [7] - a graphical argumentation notation - explicitly represents the individual elements of any safety argument (requirements, claims, evidence and context) and, perhaps more significantly, the relationships that exist between these elements (i.e. how individual requirements are supported by specific claims, how claims are supported by evidence and the assumed context that is defined for the argument). The principal symbols of the notation are shown in Figure 1 (with example instances of each concept).

The principal purpose of a goal structure is to show how goals (claims about the system) are successively broken down into sub-goals until a point is reached where claims can be supported by direct reference to available evidence (solutions). The solutions, and their parent claims, map onto the experimental questions to be answered and hence the test cases needed. As part of this decomposition, using the GSN it is also possible to make clear the argument strategies adopted (e.g. adopting a quantitative or qualitative approach), the rationale for the approach (i.e. assumptions, justifications) and the context in which goals are stated (e.g. the system scope or the assumed operational role). In our work these *non-spinal elements* help define the experimental conditions relevant. It is noted in our work not all symbols are used.

2.2 Overview of Research Strategy

The first stage of developing an immune-inspired ADS for a foraging SRS with immuno-engineering involves the identification and understanding of the TVB and anomalies in such a system. Figure 2 shows the set of activities derived for this stage with GSN where Gx refers to the research goal, Cx refers to context and Snx refer to an activity or solution. In order to establish G1, the experimental strategy is to set-up simulation environment (G2), simulating scenarios in a static environment (G3), dynamic environments without anomaly (G4), and dynamic environments with anomaly (G7). G4 and G7 are both further decomposed into defining sufficient test cases (G5) and identifying key features (G6). The static environment serves as the baseline for comparison to observe the deviations when TVB and anomalies are introduced into the system. As part of breaking down the goals is establishing context. For instance, goal G1 has context concerning what TVB is (C1) and overall task which is foraging (C2). The three TVB to establish dynamic environments are:

- **Varying Food Growth Rate** (V_{FGR})**:** The growth of food in the arena changes at different time interval.
- **Varying Food Distribution** (V_{FD})**:** The concentration of food redistributed in the environment is biased towards certain region.
- **Varying Presence of Obstacles** (V_{PO})**:** The presence of obstacles at different time interval affect the time required to reach the food, avoid obstacles, and carrying the food back to the base.

In terms of anomalies, random errors due to hardware malfunction are investigated focusing on robot wheels, food grippers and the communication device, labelled as C3 in Figure 2. We only considered hardware malfunctions in this stage to focus on one type of failure before proceed with more complicated failures such as those influenced by control software and environmental factors. The failures in both robot wheels and grippers have direct and immediate impact on the foraging task. They also span both subtle transient failures and much easier failures - permanent or significant in size. Whilst it is not claimed these are complete, they do represent a reasonable comprehensive coverage. Thus, the scale of failure is divided into the following:

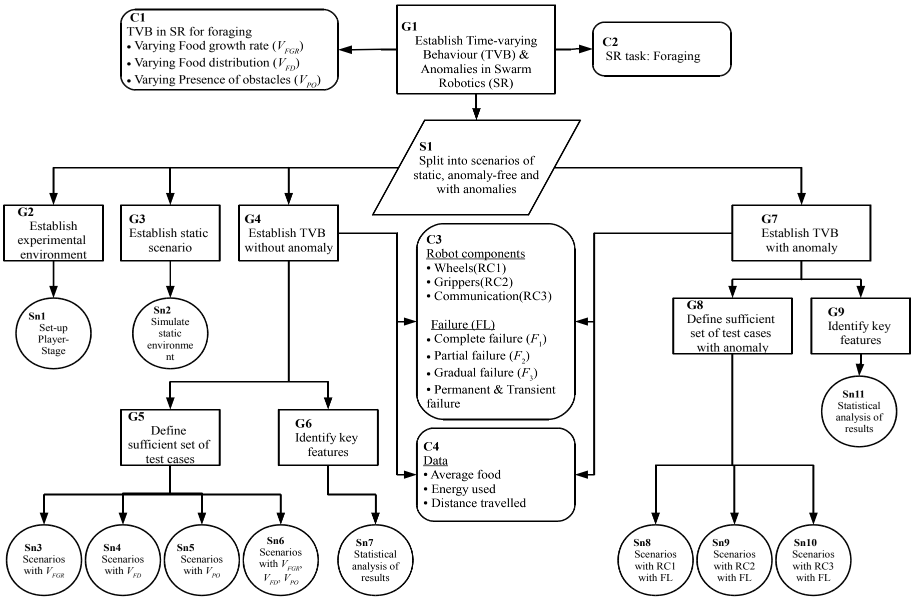

Fig. 2. Derived set of activities with GSN to investigate the TVB and anomalies in a foraging SRS

- **Complete failure (F_1):** Robot component stop responding completely. For instance in robot wheels, this failure may causes the wheels to halt to a complete stop or the robot turning with a fix angle causing it to move in a circle indefinitely.
- **Partial failure (F_2):** Robot component functioning less efficiently. In the case of robot wheels, the moving speed might be set to certain value x m/s.
- **Gradual failure (F_3):** Robot component fail slowly. For example, gradual failure in robot wheels involve gradual reduction in robot speed by y m/s for each simulation update.
- **Transient failure (F_4):** This type of failure involves alternate on and off occurrences of faulty robot component. For transient failure in robot wheels, the wheels might stop responding for z seconds and then start functioning normally before stop responding again. The component fault can be a complete, partial or gradual.

Data identified to be beneficial in detecting anomalies in foraging include the average unit of food collected (μ_{food}), distance travelled and energy used (C4). For all experiments, simulations for the swarm robotic system are implemented on the Player-Stage [14] - a software tool consists of the Player for the definition of robots and the Stage that simulates the population of robots in two-dimensional environment.

3 Time-Varying Behaviour Experiment

This section describes the simulations carried out to simulate the TVB in a foraging SRS without anomaly (G5). For clarification, following parameters are used in all experiments carried out in this paper.

- **Size of arena:** 10 m x 10 m
- **Normal robot moving speed:** 0.15 m/s
- **Initial food in arena:** 100 unit
- **Maximum food in arena:** 200 unit
- **Normal food growth rate (FGR):** 0.1 unit/simulation update
- **Each simulation update:** 200 microseconds

As mentioned in section 2.2, the simulation of foraging swarm robots in a static environment (Sn2) serves as the baseline for comparison to dynamic environments. Figure 3 shows the μ_{food} by each robot in such an environment. It is noted that individual robots behaviour cannot easily be distinguished, however as it is the differences in individual behaviour versus the swarm that is important then this is not considered to be an issue. In this paper, the μ_{food} is calculated over a time window of size four with each time slot of size 250 simulation seconds. These values are chosen from experimental results showed that they are most suitable for the problem of interest. In Figure 3, a difference up to 2 units of food (11.70%) is observed between the robots but the largest difference, which occurs around time 48, only lasts one or two clock cycles. This gives the potential to achieve more reliable detection by observing differences over a period of

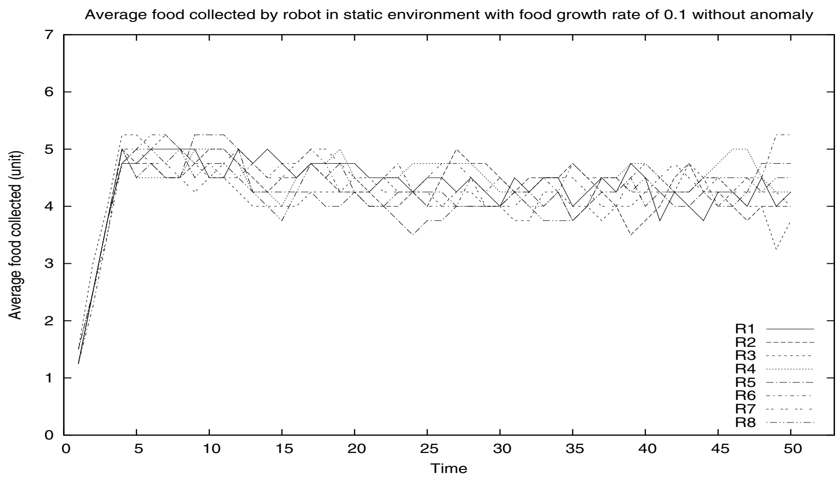

Fig. 3. μ_{food} in a static environment without anomaly

time. However, the overall pattern remained consistently similar throughout the experiment in all robots. This observation confirmed our initial belief that the μ_{food} by all robots is consistent within a local proximity and this information may proved to be useful in helping detecting anomalies.

When the same robots are placed in environments with V_{FGR} (Sn3), V_{FD} (Sn4), V_{PO}(Sn5), or combination of all three behaviour (V_{MIX}) (Sn6), the observed collection of food changes significantly as shown in Figure 4. In this simulation, the FGR (simulated with normal distribution random number generator provided in GNU Scientific Library (GSL)) is varied between 0.1 and 0.025. FGR of 0.1 means that for each simulation update, the probability of adding one unit of food into the arena is 0.1. Lower value of FGR means lower probability of adding new food into the arena. With FGR set to lower value, μ_{food} by each robot decreases as the concentration of food in the arena decreases.

The observations in Figure 4 as well as other simulations with V_{FD}, V_{PO} and V_{MIX} signify the huge influence of these TVB over the collective task. Although the pattern of food collected is very similar among the robots in all scenarios, deviation from the local mean (calculated over all robots in the arena) from 2.82% to 33.29% is observed. Again the larger difference were observed to be short lived. Noise of such magnitude make the development of an ADS with high detection rates and low false alarms a very challenging task. Table 1 is a detailed information regarding the maximum and minimum standard deviation from local mean calculated for all five simulated scenarios. It can be seen that the deviations among robots with scenarios of single TVB (V_{FGR}, V_{FD}, V_{PO}) are very similar. When the environment becomes more dynamic (V_{MIX}), it becomes more unstable and thus higher deviation in food collection by robots is observed.

Table 2 shows the distance travelled and energy used by each robot under both static and dynamic environments. It can be seen that depending on the type

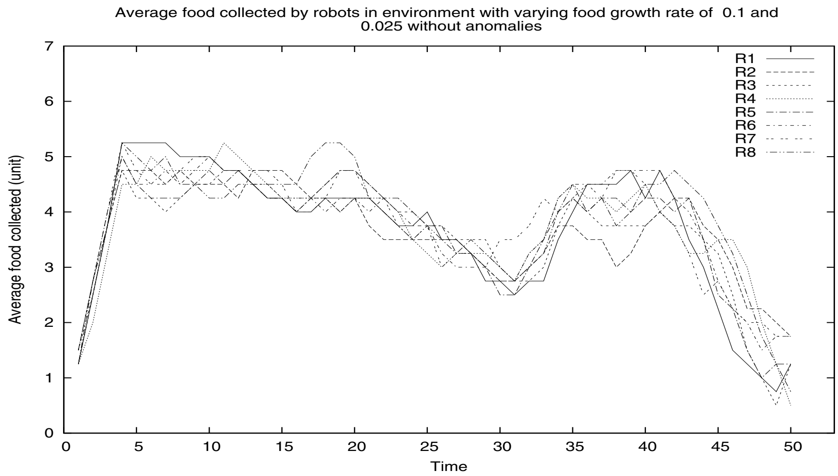

Fig. 4. μ_{food} by robots in a dynamic environment with V_{FGR} without anomaly. In this experiment, FGR is varied between 0.1 and 0.025. At time t=[10,20) and t=[30,40), the FGR is set to 0.1 while at time t=[20,30) and t=[40,50) it is set to 0.025.

Table 1. Minimum and maximum percentage of deviation, σ, between the mean for all eight robots in the arena and their individual values

Scenario	Static	V_{FGR}	V_{FD}	V_{PO}	V_{MIX}
σ_{min} (%)	3.35	4.88	6.22	3.54	6.95
σ_{max} (%)	11.70	25.41	25.57	20.06	33.29

of TVB simulated, robots might move either with greater distance or shorter distance compared to the one in a static environment. Simulations so far have showed that V_{FGR} and V_{FD} resulted in a greater distance travelled by all robots. With V_{PO}, one explanation as to why less distance was travelled by each robot might be that time was wasted to avoid the obstacles instead of wandering the arena to search and collect food. This is similar to the case with V_{MIX}. In terms of energy usage, results in Table 2 indicate that robots are consistently consuming less energy for all four scenarios in a dynamic environment compared to a static environment. This is largely due to the fact that the number of food collected by each robot is less in dynamic environments and carrying of food consumes more energy than wandering around the arena.

The TVB experiment showed the nature of dynamic environments and the effects of the TVB on the foraging task. It can be concluded that dynamic environments pose significant challenges and influence greatly the ability of an ADS in detecting anomalies.

Table 2. Distance travelled and energy usage by robots in a static and dynamic environments. For energy usage, object avoidance and moving around in the arena without food in the grippers cost 1 unit of energy. Moving with food in the grippers consumes 1.5 unit of energy while robot in resting state only consumes 0.1 unit of energy.

Metric	Distance (meter)					Energy (unit)				
Robot	Static	V_{FGR}	V_{FD}	V_{PO}	V_{MIX}	Static	V_{FGR}	V_{FD}	V_{PO}	V_{MIX}
R1	1565.36	1590.38	1610.56	1517.33	1557.97	15131.7	14662.7	14423.8	14923.5	14557.9
R2	1561.66	1589.48	1586.09	1512.09	1548.55	15107.0	14780.8	14295.2	14969.2	14575.2
R3	1569.17	1595.31	1584.71	1521.10	1531.81	15103.6	14818.2	14342.4	15060.9	14369.1
R4	1566.37	1585.18	1600.88	1523.31	1553.73	15209.3	14787.9	14352.0	15005.2	14444.7
R5	1566.62	1592.64	1602.72	1520.98	1507.24	15113.7	14760.6	14426.3	14929.3	14456.6
R6	1570.64	1610.90	1592.42	1528.34	1544.99	15086.8	14791.9	14372.3	15031.7	14441.6
R7	1568.66	1585.28	1589.67	1488.28	1565.98	15167.2	14748.8	14358.2	14997.3	14498.0
R8	1562.83	1597.19	1593.70	1522.35	1540.28	15143.0	14793.4	14329.6	14965.3	14402.6

4 Anomaly Experiment

To establish anomalies in a foraging SRS in dynamic environments (G7), four scales of malfunctions to robot components are investigated (C2). In all simulations conducted for G7, the anomaly (F_1, F_2 and F_3) is injected into robot R5 at time t=25 while F_4 is injected at time t=[15,25) and t=[40,50). Due to space limitation, we only present the results of simulating failure in robot wheels (Sn8) in this paper. Table 3 shows the details about the type of anomaly, parameter and time when the anomalies are injected. These anomalies are simulated for all four dynamic environments.

Table 3. Test cases for TVB with anomaly

Anomaly	Parameter	Time
Permanent failure, F_1	Speed=0.15 m/s, turn angle = left 10°	t=[25,50)
Partial failure, F_2	Speed={0.05, 0.1, 0.11}m/s	t=[25,50)
Gradual failure, F_3	Speed reduction per simulation update ={0.00001, 0.00005, 0.0001}m/s	t=[25,50)
Transient failure, F_4 (with F_1, F_2 or F_3)	Same as F_1, F_2 and F_3	t=[15,25) and t=[40,50)

When F_1 is injected into a robot, the effect is immediate as illustrated in Figure 5 for robot R5. Significant drop in μ_{food} for R5 can be observed and the value reduced to zero at time t=29. This type of failure should be easier to detect and is included as a baseline for comparison with other type of failures. Similar results are expected when such failure is introduced in other scenarios with V_{FD}, V_{PO} and V_{MIX}.

For partial failure (F_2), the severity of partial failures has significant influence on the possibility of detecting the anomaly. If the partial failure is more critical, significant deviations can be observed over a period of time. However

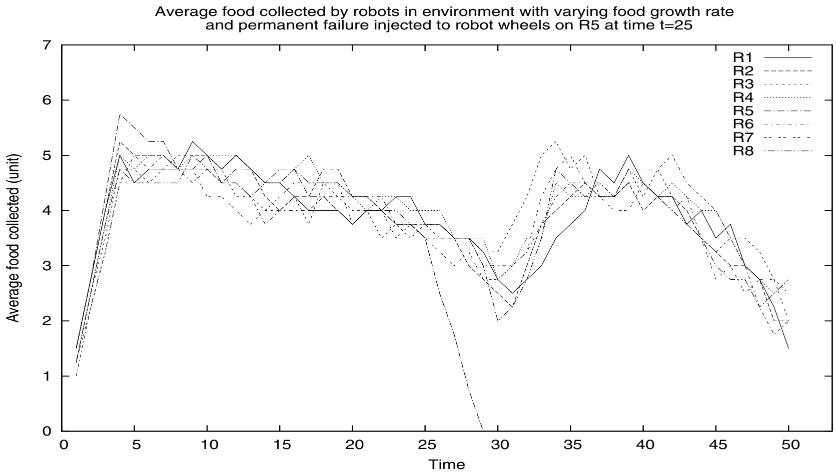

Fig. 5. μ_{food} by robots in a dynamic environment with V_{FGR} and F_1 is injected to robot wheels on R5 at time t=25. F_1 in this scenario is setting the turning angle of R5 to left turn 10° causing the robot to move in a circle.

when the partial failure is more subtle, such as setting the value of F_2 to 0.11 m/s (normal is 0.15 m/s), the differences are not that obvious and can be very difficult to be spotted as illustrated in Figure 6. One might be tempted to further experiment with F_2 value set to be as close as possible to the normal speed such as 0.13 or 0.14 m/s. However, such an action is questionable since such subtle

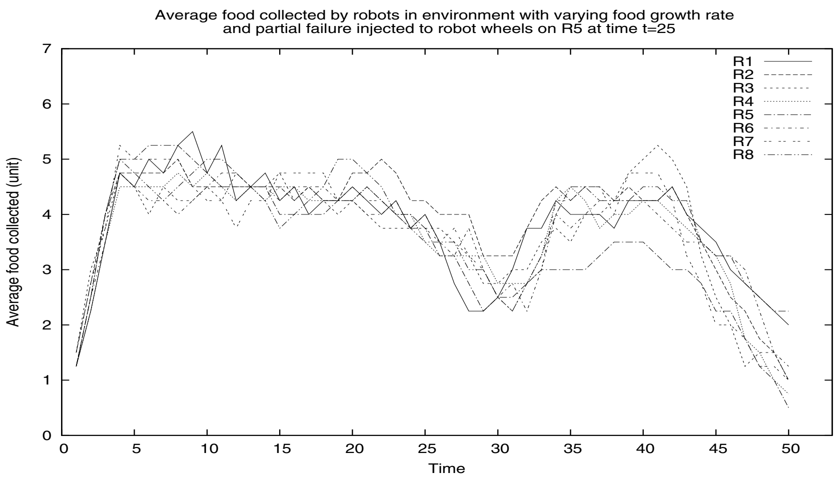

Fig. 6. μ_{food} by robots in a dynamic environment with V_{FGR} and F_3 is injected to robot wheels of R5 by setting the moving speed to 0.11 m/s at time t=25

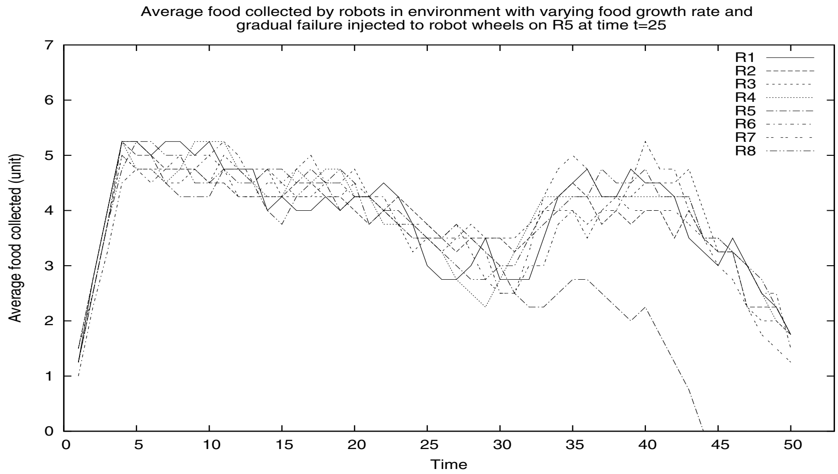

Fig. 7. μ_{food} by robots in a dynamic environment with V_{FGR} and gradual failure F_2 of 0.00001 m/s is injected to robot wheels of R5 at time t=25

difference might not be suitable to be classified as failure especially in real world applications where navigation surfaces are not exactly the same for all robots.

F_3 represents failures in the robot components that occur gradually instead of immediate failure as in F_1 and F_2. Such anomalies have a greater impact on the responsiveness of an ADS since it takes some time before any differences can be detected. Similar to F_2, this type of failure will prove to be very difficult to detect if the gradual failure progress very slowly over a long period of time. Figure 7 shows such an example where significant deviation is only apparent after time $t \geq 32$. Therefore, in detecting such anomalous behaviour over a period of time (long term) is more beneficial instead of looking at values at each time instance. This is also the reason why we chose to use average value instead of exact quantity of food collected for each time period. As discussed in the previous section this would also help remove exceptional, but short-lived, measures within the normal behaviour. Such characteristic is similar to biological immune system that operates on multiple timescales to protect the body.

Transient failure (F_4) is the type of error that occurs periodically instead of permanently as in F_1 to F_3. Such failures can happen to robot swarms in an environment with inconsistent environmental conditions. It maybe a case where a few sections on a long path between the base and the food source are covered with rough and uneven surfaces or there exist electromagnetic interference to the sensors. In such situation, ADS developed needs to be able to learn such behaviour and able to provide faster responses on second encounter. Figure 8 shows an example of F_4 simulated in an environment with V_{FGR}. Two occurrences of anomaly can be seen from the graph. Both of these have slightly different failure patterns even though they are injected with the same F_2 value. This observation

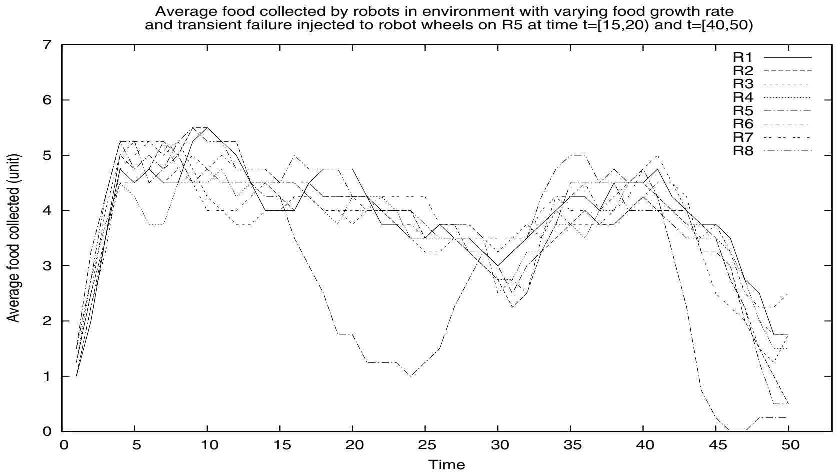

Fig. 8. μ_{food} by robots in a dynamic environment with V_{FGR} and F_4 is injected to robot wheels of R5 by setting moving speed to 0.05 m/s at time t=[15,25) and t=[40,50)

Table 4. Minimum and maximum percentage of deviation, σ (for μ_{food} by robots), from local mean in simulations with faulty robot wheels. Mod range is the range where three quarters of the deviations are located.

Scenarios	σ_{lmin} (%)	σ_{lmax} (%)	mod range (%)	σ_{R5min} (%)	σ_{R5max} (%)	mod range (%)
F_1+V_{MIX}	5.1	35.1	6.0-24.0	25.0	100	85.0-100
F_2+V_{MIX} (speed=0.05 m/s)	5.8	34.9	5.0-20.0	19.7	100	62.0-90.5
F_2+V_{MIX} (speed=0.10 m/s)	4.6	35.8	4.0-16.8	8.7	53.3	10.8-43.2
F_2+V_{MIX} (speed=0.11 m/s)	3.9	36.5	6.4-23.4	3.8	44.4	4.5-31.5
F_3+V_{MIX}(speed-=0.00001 m/s)	7.8	29.0	11.4-24.6	10.0	100	17.2-100
F_3+V_{MIX} (speed-=0.00005 m/s)	3.6	29.4	5.7-21.9	25.8	100	90.0-100
F_3+V_{MIX} (speed-=0.0001 m/s)	4.8	30.0	4.0-17.5	9.7	100	90.9-100
F_4+F_1+V_{MIX}	6.8	33.1	6.0- 25.0	9.1	100	90.9-100
F_4+F_2+V_{MIX} (speed=0.11 m/s)	4.7	32.2	9.8-18.5	5.0	52.7	12.2-35.0
F_4+F_3+V_{MIX} (speed-=0.00001 m/s)	6.8	27.6	10.4-23.6	9.9	61.1	6.1-48.8

demonstrates the unpredictability of failures identified and the need for an ADS that is able to adapt accordingly.

Table 4 summarises the minimum and maximum percentage of deviations from local mean in scenarios identified to pose the greatest challenges in detecting anomalies for faulty robot wheels. In this table *speed* refers to the moving speed of robot injected with a failure. By looking at the minimum and maximum deviations for anomalous and normal robots, it is apparent that in many cases the σ_{R5min} is less than σ_{lmax}. This means that the minimum deviation from local mean for anomalous robot R5 is still within observed normal behaviour. However, since these σ_{R5min} are only obtained from the first few slots after the anomaly was injection such pattern is expected (due to the fact that μ_{food} is calculated over four slots).

To analyse the implicit nature of anomalies, we analysed the deviations observed over the full duration of anomalies and tabulated the range where three quarters of the deviations are located as *mod range*. To interpret the significant of mod range, we take first row of Table 4 as an example. In this case, three quarters of deviations from local mean by normal robots are in the range of 6.0% to 24.0%. However, the range for the anomalous robot R5 is between 85.0% to 100%. Thus, to detect anomalies any deviation greater than 24% from local mean maybe considered as anomalous. Looking at the mod range column for normal robots, it is apparent that the maximum range is only up to 25% from local mean as opposed to 100% for R5. Thus, it can be seen than deviation of more than 25% from local mean can be use as a threshold to differentiate between anomalous and normal instances. Similar analysis were carried out for distance travelled and energy used by normal robots and anomalous robot R5. It was discovered that deviation of more than 1.02% and 4.45% from local mean can be used as threshold for energy used and distance travelled respectively to differentiate anomalies. We also analysed the data using other methods such as Jacobson/Karels algorithm [5] that combines the short-term (based on the standard deviation between moving average and current sample) and long-term (moving average) effects of collected data but found it to be inappropriate for our SRS due to noise and inconsistency of data in dynamic environments. Besides this, analysis was also conducted to compare the local mean and individual robot internal mean (calculated over the four time slot). Again, standard deviation between the two values is apparent and should be useful in detecting anomalies.

The anomaly experiment showed the difficulties in differentiating between normal and anomalous instances in dynamic environments. Our preliminary analysis has discovered that deviations from local mean of more than 25% for μ_{food}, 1.02% for energy usage and 4.45% for total distance travelled can be used as threshold to differentiate between normal and anomalous instances. At the same time, looking using a detector over a short period of time (typically 2-3 clock cycles) can help make differences more significant. Our current work is continuing the experimentation to complete the specification. With the requirements of acceptable accuracy, response within time, lightweight and robust to dynamic environment in mind, our next stage of research involves the identification of suitable immune models that addresses such issues.

5 Conclusion

This paper has introduced immuno-engineering as a principled approach to developing artificial immune systems. Immuno-engineering advocates an initial problem-oriented perspective so as to better understand the problem domain, so that an effective, and importantly, tailored AIS can be developed. Rather than looking first at the immunology to see what can be used to develop an algorithm, the adoption of the immuno-engineering approach forces us to examine carefully the application area first. In this paper we have taken the first steps to understanding the problem of anomaly detection in swarm robotic systems with time varying behaviour. We have made use of a well established technique known as Goal Structured Notation (GSN) to derive the set of information needed for us to establish an understanding the problem domain: from our experience we would advocate the use of the GSN within the context of immuno-engineering. With this information establish, a comprehensive series of experiments were defined to give the raw data needed to analyse the domain and hence create a set of requirements for any anomaly detection system. Our initial analysis of the swarm behaviour data showed the effects of dynamic environments on the collective task, the nature of anomalies and the need for learning and adaptation in detecting anomalies in a foraging SRS.

References

1. Andrews, P.: An investigation of a methodology for the development of artificial immune systems. Ph.D thesis, Department of Computer Science, University of York (2009)
2. Bayindir, L., Sahin, E.: A review of studies in swarm robotics. Turkish Journal of Electrical Engineering and Computer Sciences 15(2) (2007)
3. Christensen, A.: Fault detection in autonomous robots. Ph.D. dissertation (2008)
4. Freitas, A., Timmis, J.: Revisiting the foundations of artificial immune systems for data mining. IEEE Trans. on Evolutionary Computation 11(4), 521–540 (2007)
5. Jacobson, V.: Congestion avoidance and control. ACM Computer Communication Review 18, 314–329 (1988)
6. Jakimovski, B., Maehle, E.: Artificial immune system based robot anomaly detection engine for fault tolerant robots. In: Rong, C., Jaatun, M.G., Sandnes, F.E., Yang, L.T., Ma, J. (eds.) ATC 2008. LNCS, vol. 5060, pp. 177–190. Springer, Heidelberg (2008)
7. Kelly, T.: Arguing Safety - A Systematic Approach to Safety Case Management. Ph.D thesis, Department of Computer Science, University of York (1999)
8. Lau, H.K., Timmis, J., Bate, I.: Anomaly detection inspired by immune network theory: A proposal. In: Proc. of the 2009 IEEE Congress on Evolutionary Computation (IEEE CEC 2009). IEEE, Los Alamitos (2009)
9. Liu, Y., Passino, K.M.: Swarm intelligence: Literature overview. Technical report, Ohio State University (March 2000)
10. SYMBRION REPLICATOR, `http://www.symbrion.eu/`
11. Sahin, E.: Swarm Robotics: From Sources of Inspiration to Domains of Application. In: Şahin, E., Spears, W.M. (eds.) Swarm Robotics 2004. LNCS, vol. 3342, pp. 10–20. Springer, Heidelberg (2005)
12. Sahin, E., Girgin, S., Bayindir, L., Turgut, A.E.: Swarm Robotics. In: Blum, C., Merkle, D. (eds.) Swarm Intelligence: Introduction and Applications. Natural Computing Series, pp. 87–100. Springer, Heidelberg (2008)
13. Segel, L., Cohen, I.: Design Principles for the Immune System and Other Distributed Autonomous Systems. Oxford University Press, USA (2001)
14. Stage-Player, `http://playerstage.sourceforge.net/`
15. Stepney, S., Smith, R., Timmis, J., Tyrrell, A., Neal, M., Hone, A.: Conceptual frameworks for artificial immune systems. Int'l Journal of Unconventional Computing 1(3), 315–338 (2005)

16. Timmis, J., Hart, E., Hone, A., Neal, M., Robins, A., Stepney, S., Tyrrell, A.: Immuno-engineering. In: Proc. Second IFIP Int'l Conf. on Biologically Inspired Collaborative Computing, pp. 3–17. Springer, Heidelberg (2008)
17. Winfield, A., Nembrini, J.: Safety in numbers: Fault tolerance in robot swarms. Int'l Journal on Modelling Identification and Control 1(1), 30–37 (2006)