Increasing Endurance of an Autonomous Robot using an Immune-Inspired Framework

Maizura Mokhtar, *Member, IEEE*, Joe M. Howe

***Abstract*—This paper describes the implementation of an online immune-inspired framework to help increase endurance of an autonomous robot. Endurance is defined as the ability of the robot to exert itself for a long period of time. The immune-inspired framework provides such capability by monitoring the behavior of the robot to ensure continuous and safe behavior. The immune-inspired framework combines innate and adaptive immune inspired algorithms. Innate uses a dendritic cell based innate immune algorithm, and adaptive uses an instance based B-cell approach. Results presented in this paper shows that when the robot is implemented with the immune-inspired framework, health and survivability of a robot is improved, therefore increasing its endurance.**

I. Introduction

ENDURANCE is defined as the ability to exert oneself for a long period of time. For a robotic system, this is the ability to manage its energy onboard and to reschedule its task online. One method of achieving such capability is to incorporate a monitoring controller in parallel with the robot's behavioral based controller. This controller, the immune-inspired framework, will monitor the output responses produced by the behavioural-based controller and ensure that the output produced will not cause significant deviation from its steady state behaviour, if the robot was to endure an increase or change to its functionalities. Any deviation detected causes the immune-inspired framework to reschedule the robot's task (online) based on its current and previous safe conditions in order to ensure sustainable robotic operations. This in turn increases the endurance of the robot.

This paper presents an immune inspired framework that increases the endurance of an autonomous robot. The paper is divided into six sections: Section II describes the objective of the presented immune inspired framework within an autonomous robot. Section III describes the presented immune-inspired framework. Section IV describes the experiments carried out to test the capabilities of the framework and how the framework is instantiated. Section V discusses the results and section VI concludes this paper.

Manuscript received September 15, 2010. This work is supported in part by Military Air and Information, BAE Systems UK.

M. Mokhtar and J. M. Howe are with the Center for Energy and Power Management, University of Central Lancashire, Preston, UK (phone: +44-1779-3235, +44-1779-4220; fax: +44-1779-2926; e-mail: MMokhtar@uclan.ac.uk, JMHowe@uclan.ac.uk).

II. Objective

The main objective of the immune-inspired framework within an autonomous robot is to increase its endurance by maintaining and/or improving the health and survivability of the robot whilst operating in its environment autonomously. This is defined as the ability to perform:

1) Error Detection: to provide the correct indication of erroneous behaviors or behaviors that can cause danger to the robot (for example, performing a high energy demanding task with low energy onboard) (Null hypothesis I presented in section V). This in turn permits:

2) Error Compensation: to help increase longevity and maintain survivability of the robot, whereby compensation to errors allows the robot to survive longer and maintain and/or improve its health whilst performing its functionality. (Null hypothesis II presented in section V).

In the biological immune system, error detection is performed by the innate immune systems. Error compensation is performed by the adaptive immune systems. The innate immune system consists of, among others, dendritic cells that transverse the body detecting and indicating bacteria, viruses or pathogens (invasive bodies) that affect the health of the body. These cells inform the adaptive immune system which consist of, among others, B-cells and T-cells that eliminate the indicated invasive bodies.

The presented framework differs from other immune inspired algorithms because the framework provides a measure of health for the robot and increases its endurance by performing (i) adaptive error detection by correlating new information about the robot (*Adaptive*) [2, 3, 8, 9] with pre-defined information (*Innate*) [1] and (ii) error compensation using the information provided by the *Adaptive* component.

III. Immune-Inspired Framework

The immune-inspired framework, similar to that presented in [4] consists of three parts: (i) *Innate*, (ii) *Adaptive* and (iii) *Compensation.*

1) Innate: Innate is based on the dendritic cell algorithm originally presented in [1]. *Innate* monitors the robot's behavior over time and indicates any deviation from the robot's specified operational boundaries [4]. The robot's operational boundaries are pre-defined using the information produced from the datasheets or operational manuals of the components and modules used in the robot.

If component m is monitored by *Innate*, *Innate* will produce the signal $I_m = \{0,\ldots,1\}$, calculated based on the

progression of data produced by m (i_m) over time $\left(di_m/dt\right)$, as well as its rate of change $\left(d^2i_m/dt^2\right)$. I_m is the normalized collective indication that the weighted sum of the three signals associated with i_m: (i) j_m^1 typically calculated using (1), (ii) j_m^2 using (2), and (iii) j_m^3 using (3) is greater (or lesser, depending on its application) than a threshold value δ_m (4) and is within a certain time window τ_m (5).

$$j_m^1 = i_m(t) \tag{1}$$

$$j_m^2 = di_m/dt \tag{2}$$

$$j_m^3 = \begin{cases} 0 & if \quad (\alpha_m > i_m(t) > \beta_m) \\ 1 & otherwise \end{cases} \tag{3}$$

$$k_m(t) = \begin{cases} 1 & if \quad \left(\dfrac{\sum_a^3 w_a j_m^a}{\sum_a^3 w_a} \right) \geq \delta_m \\ 0 & otherwise \end{cases} \tag{4}$$

$$I_m(t) = \frac{\sum_{b=t-\tau_m}^{t} k_m(b)}{\tau_m + 1} \tag{5}$$

A definite erroneous behavior is indicated if $I_m = 0$. $I_m = 1$ if m is behaving as it should (within the steady state boundaries of the robot).

Information from *Innate* not only provides for the indication of erroneous behavior [1, 5, 7]; but, the values of I_m are also used to provide for the measure of health for the robot. Health of the robot is calculated by *Adaptive*.

2) Adaptive: (Algorithm 1) *Adaptive* is based on the adaptive B-cell immune-inspired algorithm initially presented in [2, 3]. *Adaptive* creates a profile of the robot's behavior at time t (online) or $R(t)$ and provides an additional indication of deviation from steady state behaviour calculated using the information provided by *Innate* and the robot's behavioural profile P [3, 4].

$R(t)$ provides information regarding the n number of attributes monitored by *Adaptive* at time t with $r_y \in R$. r_y is the information produced by attribute y at time t. The robot's profile P contains w_{max} number of detectors with $z_q \in P$. z_q is a detector that describes one steady state condition of the robot. z_q stores n number of attributes for the robot. c_y stores the value for attribute y for one steady state behavior, such that $c_y \in z_q$ and $y = \{1,\ldots,n\}$.

The measure of health for the robot at time t, $H^M(t)$, is calculated using the information from *Innate* (6) and is used to update the value of $H_{z_i}^D$ (7), if the current state of the robot $R(t)$ matches a detector in the profile, $z_i \in P$. R matches z_i if $A_i(t) \geq \varepsilon$. $A_i(t)$ is calculated using (8). $H_{z_i}^D$ is a measure that describe how often the robot performs the behavior recorded by the detector z_i.

$$H^M(t) = f(I_1(t),\ldots,I_m(t)) \tag{6}$$

$$H_{z_i}^D = f\left(H^M(t), H_{z_{i-1}}^D\right) \tag{7}$$

$$A_i(t) = \frac{1}{n}\sum_{i=1}^{n} A_f(r_i, c_i, t) \tag{8}$$

$$A_f(r_i, c_i, t) = 1 - (r_i - c_i) \tag{9}$$

$H_{z_{i-1}}^D$ is the value of $H_{z_i}^D$ at previous match.

If $R(t)$ does not match any of the detectors in P, a new behavior is detected and $R(t)$ is saved as a new detector ($z_i \leftarrow R$, line 25 in Algorithm 1). z_i will replace a detector with the lowest (or highest) value of $H_{z_i}^D$ depending on (7), if the maximum number of detector in P is reached (line 6 – 9 in Algorithm 1).

Adaptive differs from other similar algorithms, for example negative selection algorithm [8, 9] and clonal selection algorithm [2, 3], because these algorithms create their detector pools offline, prior to system deployment (i.e. during their training period) [3, 8, 9]. Furthermore, *Adaptive* not only stores the information regarding a system state (or a behavior) in a detector (z_q), but also how a state is affected by the system's behavior over time (7). This provides for adaptive health measurements and error detection; features which differentiate *Adaptive* from other similar algorithms.

3) Compensation: (Algorithm 2) A deviation of health, D_E (in Algorithm 1), indicates if there is a deviation from the steady state behavior of the robot. If a deviation is detected ($D_E \geq \eta$, line 8 onwards in Algorithm 2), the framework prevents the execution of the behavioral based controller's output. This ensures no transition to an erroneous or bad state is made by the robot. A previous safe state is executed instead. Figure 1 illustrates the compensation mechanism of the immune-inspired framework.

The framework reschedules the robot's behavior online, using the combined information provided by *Innate* and *Adaptive* (error detection), to ensure stable, continuous and sustainable operation; thus, increasing the robot's endurance.

IV. EXPERIMENT

A. The robots and their environment

To test the capabilities of the immune-inspired framework, the framework is implemented in parallel with a behavioral-based controller (as illustrated in Fig. 3) of an EPUCK robot simulated in the Player/Stage robotic simulator (Fig. 2) [6].

Algorithm 1: *Adaptive*, similar to that presented in [4].

Constant: o = no. of attributes monitored by *Innate.*
n = no. of attributes monitored by *Adaptive.*

Input: w_{max} = max. no. of detectors in P.
N = a set of $I_m(t)$ for module m, $m=\{1,\ldots,o\}$.
R = a set of attributes monitored by *Adaptive* with $r_y \in R$, r_y is the value for module y, $y = \{1,\ldots,n\}$.

Output: D_E = significance of error detected, with $D_E = \{0,\ldots,1\}$ and $D_E = 0$ no error is found.

1. **begin**
2. Create w_{max} no of detectors in pool P, with detector $z_q \in P$, $q = \{1,\ldots,w_{max}\}$;
3. $w \leftarrow 0$;
4. $t \leftarrow 0$;
5. **repeat**
6. **if** $w = w_{max}$ **then**
7. Delete z_q with lowest $H^D_{z_q}$ (or highest, depending on (7));
8. $w \leftarrow w_{max} - 1$;
9. **end**;
10. $i \leftarrow 0$;
11. $A_B \leftarrow 0$;
12. $H^M(t) \leftarrow f(I_1(t),\ldots,I_o(t))$;
13. $D_E = 0$;
14. **while** $(i \leq w)$ **do**
15. $A_i(t) \leftarrow f(R,z_i,t)$;
16. **if** $(A_i \geq \varepsilon$ and $A_B = 0)$ **then**
17. $H^D_{z_{i-1}} \leftarrow H^D_{z_i}$;
18. $H^D_{z_i} \leftarrow f\left(H^M(t), H^D_{z_{i-1}}\right)$;
19. $\Delta H(t) \leftarrow H^D_{z_i} - H^D(t-1)$
20. $A_B \leftarrow 1$;
21. **end**;
22. $i \leftarrow i + 1$;
23. **end**;
24. **if** $(A_B \neq 1)$ **then**
25. $z_i \leftarrow R$;
26. $H^D_{z_i} \leftarrow H^M(t)$;
27. $w \leftarrow w + 1$;
28. $\Delta H(t) \leftarrow H^D_{z_i} - H^D(t-1)$
31. **end**;
32. **if** $\left(A_B = 1\right)$ **then**
33. $D_E \leftarrow \left|\Delta H(t)\right|$;
34. **end**;
35. $H^D(t-1) \leftarrow H^D_{z_i}$;
36. t ← t + 1;
37. **until end**;
38. **end**;

Algorithm 2: *Compensation,* the interaction between the robot's behavioral-based controller and the immune-inspired framework.

Input: I_S = set of input sensor values.
I_P = output of the power module.
S_d = robot's speed.
T_r = robot's turn rate.
T_k = robot's state id.

Output: O_P = Output from the behavioural-based controller.
B = Output buffer.

1. **begin**
2. Create buffer for state id, B_{Tk};
3. Create buffer for speed, B_{Sd};
4. Create buffer for turn rate, B_{Tr};
5. **repeat**
6. Determine T_k and calculate O_p (S_d and T_r) based on I_S and I_P;
7. *Adaptive* checks if O_p is correct;
8. **if** $(D_E < \eta)$ **then**
9. O_p calculated is correct;
10. Robot performs O_p;
11. **if** $(D_E \leq \mu)$ **then**
12. O_p is saved to B:
$B_{Tk} \leftarrow T_k$;
$B_{Sd} \leftarrow S_d$;
$B_{Tr} \leftarrow T_r$;
13. **end**
14. **else**
15. *Adaptive* detects erroneous O_p;
16. The framework performs compensation by re-defining O_p:
$T_k \leftarrow B_{Tk}$;
$S_d \leftarrow B_{Sd}$;
$T_r \leftarrow B_{Tr}$;
17. **end**;
18. **until end**;
19. **end**;

The robot's objectives are:

(i) To transfer as many loads as possible from one location in the environment (labeled A in Fig. 2) to another (B), whilst there is sufficient energy onboard to perform this objective.

(ii) Ten robots are simulated in each experiment. The robots must also ensure that all members in the group are capable of performing their first objective. If the available onboard energy is below a certain capacity at time t, $E_C(t) \leq 0.10$, the robot must stop and wait for help (transfer of energy) from another robot until it has sufficient energy to return to the recharging station (area circled in Fig. 2).

If a robot enters B (in Fig. 2), the energy dissipation rate of the robot is increased (likening to a robot traversing an area with a high friction surface or high wind resistance). Because

of this, the robot must avoid B until it has a sufficient amount of energy to go to and return from B.

The immune-inspired framework is responsible for ensuring continuous operations of the robot despite these changes occurring in the environment. As previously stated, error detection is provided by *Innate* and *Adaptive* and error compensation is provided by *Compensation*.

Sixteen experiments are conducted. Each experiment consists of three simulation runs, with each run lasting about 30 minutes of simulated time. The behavior of each robot is updated every 0.1 second of simulated time.

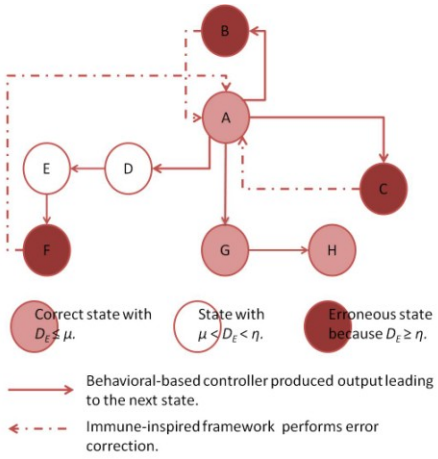

Fig. 1. How the immune-inspired framework effects the state transition of the robot; starting from state A and ending at state H. The framework functions to ensure safe state transition. This helps increase the longevity of the robot in performing its functionalities, thus increasing its endurance. Significance of D_E is indicated in Algorithm 1 and Algorithm 2 and μ and η are in Algorithm 2.

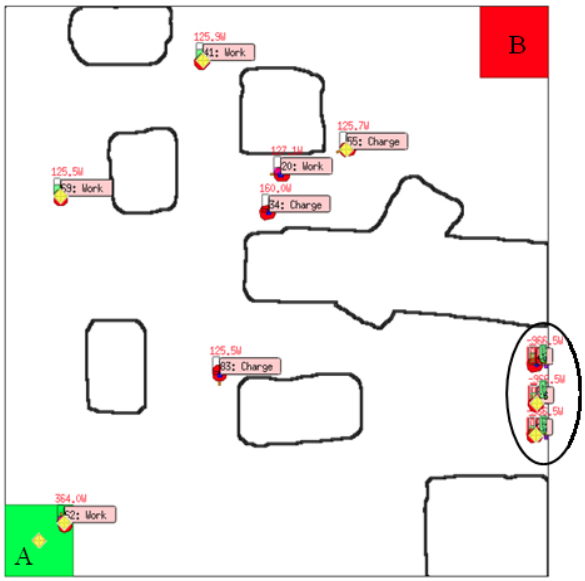

Fig. 2. The ten robots in the simulated environment. Each robot has to transfer as many loads (yellow diamonds) as possible from the area labelled A to the area labelled B within the constraints of its inboard energy. If the onboard energy is low, $E_C < 0.20$, the robot must return to the recharging station (area circled) in order to recharge. If $E_C \leq 0.10$, the robot must stop and wait for help (receive extra energy) from another robot before continuing. When entering B, the robot has to endure (simulated) an increase in its energy dissipation rate. This captures the robot entering an area with a high friction surface or high wind resistance.

A. *Instantiation of the immune-inspired framework*

1) Innate: monitors the behavior of the actuation and the power modules.

(i) *Actuation module (m = s): $i_s(t)$* is the speed of the robot at time t calculated by measuring the rate of change in distance the robot moves in x-axis ($\Delta x(t)$) and y-axis ($\Delta y(t)$). j_S^1 is calculated using (10), j_S^2 using (11) and j_S^3 using (12). $\delta_s = 0.0$ for (4) and $\tau_s = 8$ for (5).

$$j_s^1 = \begin{cases} 1 & if \quad \Delta x(t) = \Delta y(t) = 0 \\ 0 & otherwise \end{cases} \tag{10}$$

$$j_s^2 = 10\left(\sqrt{\left(\frac{\sum_{d=0}^{7}\Delta x(t-d)}{8}\right)^2 + \left(\frac{\sum_{d=0}^{7}\Delta y(t-d)}{8}\right)^2}\right) \tag{11}$$

$$j_s^3 = 100\left(\frac{\sum_{d=1}^{7}\Delta z(d) - \Delta z(d-1)}{7}\right) \tag{12}$$

$$\Delta z(t) = \sqrt{\Delta x(t)^2 + \Delta y(t)^2} \tag{13}$$

(ii) *Power module (m = p): $i_p(t)$* is the total energy dissipated by the robot at time t and j_p^1 is calculated using (14), j_p^2 using (15) and j_p^3 using (16).

$$j_p^1 = \begin{cases} 1 & if \quad E_C(t) > \dfrac{E_S(t)}{i_p(t)} \\ 0 & otherwise \end{cases} \tag{14}$$

$$j_p^2 = \frac{\left|i_p(t) - i_p(t-1)\right|}{E_S(t)} \tag{15}$$

$$j_p^3 = \frac{E_S(t) - i_p(t)}{E_S(t)} \tag{16}$$

$E_s(t)$ is the onboard energy at time t. For (4), $\delta_p = 0.0$ and $\tau_p = 8$ for (5).

2) Adaptive: Attributes presented to *Adaptive*, $R(t)$ and are compared against the detectors in P are:

(i) $r_0 = c_0$ = robot's state ID (state ID is the label alongside each robot in Fig. 2.

(ii) $r_1 = c_1$ = robot's speed, provided by the behavioral based controller.

(iii) $r_2 = c_2$ = robot's turn rate, provided by the behavioral based controller.

$H^M(t)$ and $H_{z_i}^D$ for this experiment is calculated using (17) and (18).

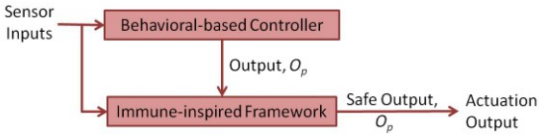

Fig. 3. How the immune-inspired framework integrates with the robot's behavioral based controller.

$$H^M(t) = \frac{I_s(t) + I_p(t)}{2} \tag{17}$$

$$H^D_{z_i} = \begin{cases} H^D_{z_i} H^M(t) + (-0.01) H^M(t) & \text{if match} \\ H^M(t) & \text{otherwise} \end{cases} \tag{18}$$

The higher the frequency of match for a detector, the lower the value for $H^D_{z_i}$. Equation (18) allows the robot to store a new behavior in a detector in P (lines 24 - 31 in Algorithm 1).

If $w = w_{max,}$ z_q with the highest value of $H^D_{z_q}$ is deleted from P, to make room for the detection of new behaviors. ε = 1.

3) Compensation: Each experiment is simulated with sixteen combinations of μ and η values (Algorithm 2). The combinations are listed in Tables I and II. Compensation, via suitable combination of μ and η values, helps increase endurance by forcing the robot to perform previous "safe" behavior (lines 11 – 13 in Algorithm 2) when a transition to "bad" behavior is detected ($D_E \geq \eta$); as illustrated in Fig. 1.

V. Results

A. Null hypothesis I: Innate will not provide useful information for the immune-inspired framework.

Figures 4 and 5 provides snapshots of the last recorded values of *Innate*, I_s (Fig. 4) and I_p (Fig. 5), recorded when the robot is in a particular coordinate in the environment (as indicated by x- and y-axis values). Coordinate (0,0) is the centre of the environment (Fig. 2).

In Fig. 4, coordinates where $I_s < 0.5$ correspond to where the robot has to reduce its speed in order to navigate around obstacles (black lines in Fig. 2); and unnecessary reduction of speed can be considered as an erroneous behavior (I_s = 0.0). In Fig. 5, $I_p \approx 0.50$ is when the robot detects the unnecessary increase in its energy dissipation rate (top right corner of Fig. 5 that corresponds to the increase in energy use when at B in Fig. 2) as well as when the robot is low on energy and requires recharging.

Results presented in Fig. 4 and 5 can thus reject the presented null hypothesis. Results of similar implementation of *Innate* are described in [5] and [7].

B. Null hypothesis: The implementation of the immune-inspired framework cannot increase the endurance of an autonomous robot.

There are no conventional methods suitable for describing how healthy a robot is. The following measures are proposed and used:

(i) The median loads taken from *A* to *B* in Fig. 2 during the experiment, over the ten robots simulated.

(ii) The averaged health $H^M(t)$ over the duration of the experiment, median over the ten robots simulated.

Results in Tables I and II indicate that there is an increase in the number of loads transferred by the robots if a suitable combination of μ and η is used. The results thus reject the null hypothesis presented.

The results also indicate the importance of finding the best combinations of μ and η. If η is too large, no deviation from steady state behavior is detected (see Fig. 6). If η is too small, the robot will continuously resort to what it considered as its previous steady state behavior and the robot is not allowed to operate beyond the boundaries of its steady state behavior. This prevents the robot from trying new behavior that may help increase its endurance. If μ is too large, erroneous transitions will be considered as safe.

Table I indicates that the best μ and η combinations are:

(i) μ = *0.00* and η = *0.50* because of the larger number of loads transferred between A and B (Fig. 2).

(ii) μ = *0.05* and η = *0.50* because of the higher median averaged value of $H^M(t)$.

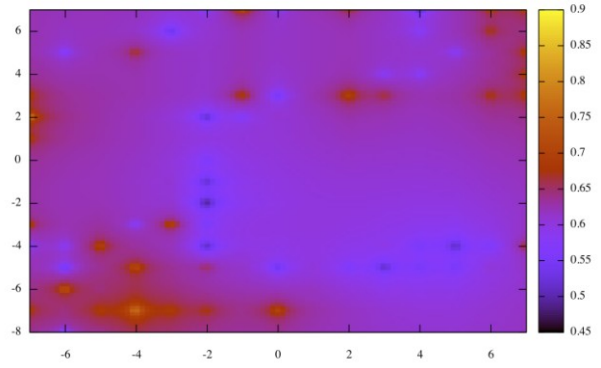

Fig. 4. The last snapshot of the values of I_s for a robot when it is in a particular coordinate in the environment. X- and y-axis indicate the x and y coordinate of the environment presented in Fig. 2. Coordinate (0,0) is located at the centre on the environment of Fig. 2.

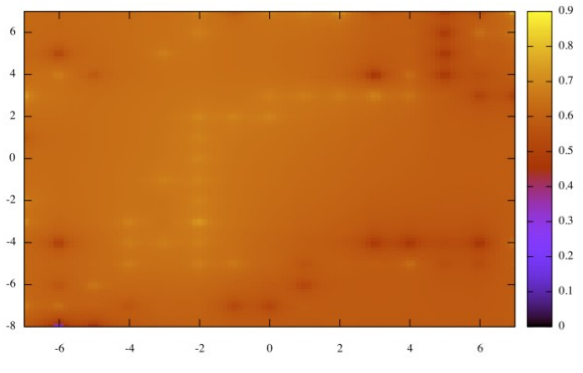

Fig. 5. The last snapshot of the values of I_p for a robot when it is in a particular coordinate in the environment. X- and y-axis indicate the x and y coordinate of the environment presented in Fig. 2. Coordinate (0,0) is located at the centre on the environment of Fig. 2.

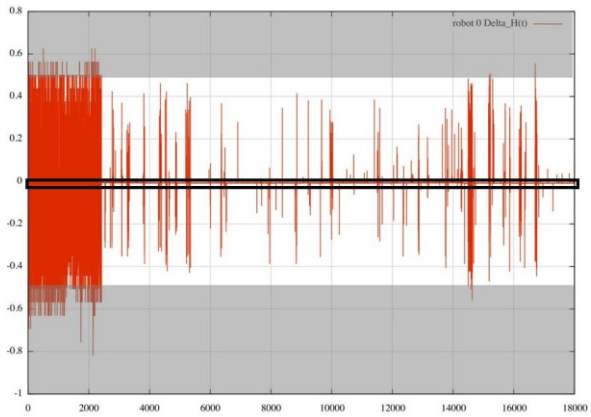

Fig. 6. The values for $\Delta H(t)$ (y-axis) for each simulation time step t (x-axis). Suitable values of μ and η (Algorithm 2) is highly dependent on the values $\Delta H(t)$. μ must be small and within the suitable health margin of the detector, $|\Delta H(t)| < 0.10$ (boxed). Large value of μ causes a deviation from steady state behavior (indicated by the spikes in the figure) as safe. If η is too large (area shaded grey), no deviation from steady state behavior can be detected. If η is too small, robot is not allowed to explore new boundaries beyond its steady state behavior that may help increase its endurance.

VI. Conclusions

With the correct combination of μ and η, the immune inspired framework can help increase the endurance of the robot whilst performing its functionalities in a dynamic environment. The immune inspired framework provides such capabilities by continuously monitoring the behavior of the robot and indicating when erroneous behaviors or deviation from its steady state behaviors are detected. This allows for necessary and safe compensation to be made, based on the current safe conditions of the robot.

References

[1] J. Greensmith, *The Dendritic Cell Algorithm*, PhD thesis, University Of Nottingham, 2007.

[2] R. deLemos, J. Timmis, S. Forrest, and M. Ayara, Immune-inspired adaptable error detection for automated teller machines," In *IEEE Trans. On Systems, Man, And Cybernetics-PartC: Applications and Reviews-Part*, vol. 37 pp. 873–886, IEEE Press 2007.

[3] R. Canham, A. Jackson, and A. Tyrrell, Robot error detection using an artificial immune system. In *Proc. of 2003 IEEE NASA/DoD Conference on Evolvable Hardware*, pp. 199–207, IEEE Press 2003.

[4] J. Timmis, A. Tyrrell, M. Mokhtar, A. Ismail, N. Owens, and R. Bi, "An artificial immune system for robot organisms". In *Symbiotic Multi-Robot Organisms: Reliability, Adaptability, Evolution*, P. Levi and S. Kernbach, Eds, pp. 268-288. Berlin: Springer-Verlag, 2010.

[5] M. Mokhtar, R. Bi, J. Timmis, and A. M. Tyrrell. "A modified dendritic cell algorithm for on-line error detection in robotic systems". In *Proc. of IEEE Congress on Evolutionary Computation (CEC) 2009,* pp. 2055–2062. IEEE Press, 2009.

[6] PlayerStage Robot Simulator. http://playerstage.sourceforge.net/

[7] R. Humza, O. Scholz, M. Mokhtar, J. Timmis, and A. M. Tyrrell, "Towards Energy Homeostasis in an Autonomous Self-Reconfigurable Modular Robotic Organism". In *Proc. of ADAPTIVE 2009*, IEEE Computer Society, 2009.

[8] D. Dasgupta and S. Forrest, "Novelty detection in time series data using ideas from immunology," in *Proc. Int. Conf. Intelligent Systems*, pp. 87–92, 1996.

[9] S. A. Hofmeyr, S. Forrest, and A. Somayaji, Intrusion detection using sequences of system calls. *Journal of Computer Security*, vol. 6, pp151–180, 1998.

TABLE I
THE MEDIAN NO. OF LOADS TRANSFERRED BY THE ROBOTS AND THE MEDIAN OF THEIR AVERAGE HEALTH VALUES OF THE 3 SIMULATION RUNS

	No of Loads	Average $H^M(t)$
Without[1]	**6**	**0.6469**
$\mu = 0.00$; $\eta = 0.05$	9	0.6694
$\mu = 0.00$; $\eta = 0.10$	7	0.6434
$\mu = 0.00$; $\eta = 0.25$	10	0.6469
$\mu = 0.00$; $\eta = 0.50$	**12**	**0.6619**
$\mu = 0.05$; $\eta = 0.10$	8	0.6840
$\mu = 0.05$; $\eta = 0.25$	10	0.6036
$\mu = 0.05$; $\eta = 0.50$	**12**	**0.6977**
$\mu = 0.10$; $\eta = 0.25$	9	0.6469
$\mu = 0.25$; $\eta = 0.50$	11	0. 6152

[1]Outputs from the *Adaptive* (Algorithm 1) are ignored.

TABLE II
THE MEDIAN (M) NO. OF LOADS TRANSFERRED BY THE 10 ROBOTS AND THE MEDIAN (M) AVERAGE HEALTH FOR THE 10 ROBOTS.

	Run No. 1				Run No. 2				Run No. 3			
	No of Loads		$H^M(t)$		No of Loads		$H^M(t)$		No of Loads		$H^M(t)$	
	M	Std. Dev.	M	Std. Dev.	M	Std. Dev.	M	Std. Dev.	M	Std. Dev.	M	Std. Dev.
Without[1]	**6**	**5.4212**	**0.6469**	**0.0128**	**6**	**5.1001**	**0.6454**	**0.0127**	**8**	**4.1042**	**0.6848**	**0.0131**
$\mu = 0.00$; $\eta = 0.05$	10	5.0783	0.6694	0.0805	6	4.8408	0.6477	0.0130	9	4.0675	0.6831	0.0046
$\mu = 0.00$; $\eta = 0.10$	11	3.4785	0.6018	0.0089	7	6.5862	0.7056	0.0223	6	5.1001	0.6434	0.0125
$\mu = 0.00$; $\eta = 0.25$	6	5.1001	0.6423	0.0125	6	5.1001	0.6469	0.0128	6	5.1001	0.6469	0.0128
$\mu = 0.00$; $\eta = 0.50$	**14**	**6.1968**	**0.6624**	**0.0261**	**14**	**5.0728**	**0.5934**	**0.0408**	**9**	**3.1711**	**0.6619**	**0.0118**
$\mu = 0.05$; $\eta = 0.10$	8	4.5228	0.6840	0.0131	9	7.0553	0.7300	0.0115	6	5.1001	0.6469	0.0128
$\mu = 0.05$; $\eta = 0.25$	9	5.0166	0.6717	0.0052	10	6.4083	0.5952	0.0088	11	3.7253	0.6036	0.0089
$\mu = 0.05$; $\eta = 0.50$	**13**	**6.3149**	**0.6977**	**0.0114**	**12**	**5.8500**	**0.7096**	**0.0164**	**10**	**5.0783**	**0.6743**	**0.0058**
$\mu = 0.10$; $\eta = 0.25$	6	5.1001	0.6469	0.0128	9	4.7656	0.6178	0.0053	9	5.8013	0.7021	0.0320
$\mu = 0.10$; $\eta = 0.50$	6	5.1001	0.6460	0.0128	9	6.3736	0.6715	0.0124	6	5.1001	0.6443	0.0126
$\mu = 0.25$; $\eta = 0.50$	11	4.9677	0.6687	0.0073	13	3.3813	0.6021	0.0039	10	5.4365	0.6152	0.0059

[1]Outputs from *Adaptive* (Algorithm 1) are ignored.
[2]Bold face is used to indicate the comparable median no. of loads transferred by the robots and their median average health, with (from best combination of μ and η) and without the monitoring capabilities of the immune-inspired framework.