Cognitively-inspired homeostatic architecture can balance conflicting needs in robots

James Stovold · Simon O’Keefe · Jon Timmis

Received: date / Accepted: date

Abstract Autonomous robots require the ability to balance conflicting needs, such as whether to charge a battery rather than complete a task. Nature has evolved a mechanism for achieving this in the form of homeostasis. This paper presents CogSis, a cognition-inspired architecture for artificial homeostasis. CogSis provides a robot with the ability to balance conflicting needs so that it can maintain its internal state, while still completing its tasks. Through the use of an associative memory neural network, a robot running CogSis is able to learn about its environment rapidly by making associations between sensors.

Results show that a Pi-Swarm robot running CogSis can balance charging its battery with completing a task, and can balance conflicting needs, such as charging its battery without overheating. The lab setup consists of a charging station and high-temperature region, demarcated with coloured lamps. The robot associates the colour of a lamp with the effect it has on the robot’s internal environment (for example, charging the battery). The robot can then seek out that colour again when it runs low on charge. This work is the first control architecture that takes inspiration directly from distributed cognition. The result is an architecture that is able to learn and apply environmental knowledge rapidly, implementing homeostatic behaviour and balancing conflicting decisions.

Keywords Robotics · Homeostasis · Autonomous Systems · Cognition · Swarm Intelligence · Associative Memory · Artificial Neural Networks
1 Homeostasis

Mechanisms that maintain the biochemical environment of the body have been studied at great length since Claude Bernard (1878) observed that the internal environment (or "milieu interne") was essential to survival (Bernard, 1878; Cannon, 1929).

Since then, many have argued that cognition is a highly-developed form of homeostasis (Lewontin, 1957; Ashby, 1960; Godfrey-Smith, 1998), and that cognition is closely linked with autonomy (Varela, 1981; Froese et al, 2007; Bodden, 2008; Vernon, 2015). Given the links between cognition, homeostasis and autonomy, we can take inspiration from cognition to guide our development of either autonomous or, in this case, homeostatic systems.

From a robotics standpoint, the ability to maintain a robot’s internal environment (battery charge, internal temperature etc.) while still completing its objectives is a key challenge. Artificial homeostasis has the potential to provide this ability, paving the way for autonomous systems that are able to effectively balance conflicting decisions without external intervention.

The mechanisms for homeostasis within the human body are driven by interactions between the nervous, endocrine, and immune systems (Widmaier et al, 2006). Previous attempts to develop artificial homeostasis have made use of artificial counterparts for each of these systems (in the form of artificial neural networks, artificial endocrine networks, and artificial immune systems). Neal and Timmis (2003); Vargas et al (2005) take this approach, focussing predominantly on neuro-endocrine networks to produce a robot controller that can change its high-level behaviour based on artificial hormones in its internal environment. The work by Stradner et al (2009); Schmickl et al (2011) follows a similar approach, developing a robotic system that makes use of hormone secretion and diffusion as a mechanism for signalling within a robot. They adapt the behaviour using evolutionary methods, and apply it to the problem of object avoidance.

The work presented in this paper takes a different approach. Rather than focussing on the biological basis for homeostasis, this paper views homeostasis as a cognitive process. When the body is working to maintain its internal environment, it responds to changes in external stimuli as detected through low-level signals. It must then decide which high-level action to take in order to maintain that internal environment. From this perspective, homeostasis could be considered similar to the action selection problem or a decision-making problem (Tyrrell, 1993). In other words, how does the body decide which action to take next in order to maintain its internal environment?

In this paper, we present CogSis, a robot control architecture based on Cohen’s (2000) definition of cognition in the immune system. By linking memory, learning, and decision-making, a robot is able to rapidly adapt to—and sustain itself in—previously-unseen environments.
We show that the CogSis architecture can:

– Enable a robot to learn from previous experiences and use them to influence future behaviour
– Provide homeostatic behaviour to a robot
– Balance two conflicting requirements to provide homeostatic behaviour to a robot

Section 2 describes the CogSis architecture in detail, including a novel collective decision-making algorithm that does not require distributed sensing or leaders within the swarm. This provides the ability for a swarm of robots (or an individual robot simulating the swarm) to make multiple successive decisions without human intervention.

Section 3 presents experimental results showing that a robot running the CogSis architecture is able to balance conflicting decisions. To demonstrate this, the robot is placed in an arena with a charging station within a high-temperature region. In order for the robot to re-charge, it has to venture into the high-temperature region, and balance overheating with running out of charge.

2 CogSis: The cognitively-inspired homeostatic architecture

The similarities between cognition, autonomy and homeostasis have been discussed at great length, most notably in (Varela [1981] Froese et al. [2007] Boden [2008] Vernon [2015]). Cognition and autonomy have many different definitions. The large number of definitions (in excess of 20 separate types and definitions (Schillo and Fischer [2004] Vernon [2015]) gives rise to the problem of how to develop ‘autonomy’ or ‘cognition’ in a robotic system. While it might be possible to construct a system that meets the requirements of one definition, there are a large number of other definitions that will contradict it. The work in this paper has been based on one specific definition, which has evidence in distributed biological systems (Cohen [1992a,b, 2000] Mitchell 2005). Cohen (2000) proposed three properties of cognitive systems:

“Cognitive systems, I propose, differ strategically from other systems in the way they combine three properties:

1. They can exercise options; decisions.
2. They contain within them images of their environments; internal images.
3. They use experience to build and update their internal structures and images; self-organization.” (p. 64, emphasis original)

The CogSis (COGnitively-inspired HomeostaSIS) architecture, shown in fig. 1 takes Cohen’s (2000) definition of cognition and uses it as the basis for homeostasis in a robot. A robot running the CogSis architecture is able to perform basic cognitive tasks—making decisions, learning about the environment, and applying previous experience to influence future decisions. These cognitive tasks enable the robot to exhibit homeostatic behaviour and adapt
to new environments rapidly, in contrast to previous approaches (Vargas et al., 2005; Stradner et al., 2009).

Fig. 1: High-level architecture of the CogSis system. The internal sensors (battery charge, temperature) drive the behaviour of the cognitive decision making (CDM) simulation. As the agents within the CDM react to the internal sensors, counting the proportion of agents in each attractor source within the CDM can be used to drive the binary signals (black arrow) to the CMM input. The CMM (trained to recall which colour lamp is mounted above charging stations and high-temperature areas) then recalls which colour to search for according to the input that is triggered by the CDM. The output from the CMM is then fed into the light-seeking circuits that perform basic chromotaxis to seek out or flee from the appropriate colour in the environment.

The CogSis architecture consists of three main components: the CDM (Cognitive Decision Making) component, a CMM (Correlation Matrix Memory), and some basic light-seeking circuitry. The CDM runs a novel decision-making algorithm (details in section 2.1), and provides action selection based on the robot’s internal sensor values.

Through the CMM, the robot is able to learn about its environment by associating changes in the robot’s external sensor values with changes in its internal sensor values. Once trained, the CMM can then recall the appropriate high-level behaviour for the robot, based on the decisions made by the CDM. Details of CMMs and their use in CogSis are provided in section 2.2.

The rest of this section provides details of each of these components, and how they contribute to the working of the CogSis architecture.
2.1 Decision making

The CDM is a major part of the CogSis architecture, providing action selection based on internal sensor values. The CDM is used to decide which of the low-level signals should affect the high-level behaviour of the robot. It works by simulating an abstract virtual environment formed of attractors and a swarm of agents. Each attractor has an associated coefficient that is linked to an internal sensor. As the sensor value varies, the ‘strength’ of the attraction also varies.

The swarm of virtual agents are collected together into a flock that makes decisions following Cohen’s (2000) definition of decision-making:

“A decision emerges... from a match between an environmental case and an internal motive. Decisions are associations.” (p. 69)

The swarm of agents respond to changes in the virtual environment, linking the behaviour of this virtual swarm to changes in the real environment outside the robot. The movement of the swarm of agents between attractors, therefore, provides the action selection required for the high-level behaviour of the robot by reacting to the low-level sensor values. In this paper, the attractors are connected to the charge level and temperature sensors on the robot.

The layout of this virtual environment could be varied in a number of different ways, but—other than varying the strength of the attraction—we keep the environment static. For this paper, we use an environment of 140x140 patches and centre the attractors at (58, 34) and (58, 110), with a spread of 15 patches. The swarm of agents is initialised at (15, 72). Through the use of emergent identities (Stovold et al, 2014), multiple swarms could be supported in the same environment but this is outside the scope of this paper.

We define the following terms:

– ‘context’ is the state of the environment as perceived by an agent (Cohen’s ‘environmental case’)
– ‘motive’ is the likelihood of picking a certain actions, should an appropriate context arise

In an empty environment, therefore, the actions of the swarm can only be a result of the inherent predisposition of certain actions within the swarm, and so are representative of the motive of the swarm. The internal motive of the swarm is provided by each agent projecting ‘virtual goals’ ahead of them. The details of this algorithm are provided in section 2.1.1.

By combining the internal motive of the swarm with the context that the swarm is in, basic decision-making can be observed. As the swarm traverses the virtual environment, it reacts to contextual cues—provided by attractor sources—and decides whether to follow its internal motive, or to follow the environmental context. This approach to decision-making allows for multiple, successive decisions to be made without intervention from any external influence. This is key to its use in the CogSis architecture, where the component is used to repeatedly make decisions based on continually-changing sensor values.
The CDM differs from previous collective decision-making algorithms as it does not rely on ‘distributed sensing’. Previous approaches (Schmickl and Hamann, 2011; Garnier et al, 2005) rely on the distributed nature of the swarm to sample the environment, then aggregate at a certain point in space to make a decision. Once a decision has been made, however, the agents are all collected together at a single point in the environment. Without a way to recognise when a decision has been made, robots have no way to make subsequent decisions.

Most existing collective decision-making algorithms instead rely on an external influence to redistribute them across the environment (typically a human operator). By collecting the agents into a flock, the CDM algorithm avoids this problem and—unlike Yu et al. (2010)—the approach of the CDM does not require ‘leaders’ within the swarm to guide it to a decision.

The modular nature of the CogSis architecture means that the CDM can be replaced by a different component to provide different behaviour to the robot. In this paper, we compare the CDM component with a control case to test the efficacy of the CDM to make successive decisions. The control case in the presented experiments uses a simple threshold component in place of the CDM. The threshold component provides output signals based on sensor readings, for example, the charge in the battery dropping below a certain fixed threshold.

2.1.1 CDM algorithm

As mentioned above, the CDM simulates a virtual environment that contains attractors and a swarm of agents collected into a flock. This section describes the algorithm that the CDM agents follow.

We defined context and motive in the previous section. Cohen’s (2000) definition of decision-making in a cognitive system is reduced to a function of context and motive:

\[
\text{Decision} = f(\text{motive}, \text{context})
\]

The agents in the swarm move according to three forces: the flock force, the motive force, and the context force. These three forces—described in detail below—combine to provide a final directional force that the agents follow. The flock force is provided by Olfati-Saber’s (2006) algorithm, and works to keep the flock in a lattice formation. The motive force is provided by ‘virtual goals’ projected by each agent ahead of the flock. The context force provides environmental cues to the agents, resulting from attractors in the environment.

The overall algorithm, combining each of the three forces listed above can be described by a series of vector-adjustments, \( u_i \), for an agent \( i \):

\[
u_i = f_i^g + f_i^\gamma
\]

where \( f_i^g \) acts to form a lattice structure among the neighbours in a flock (flock force), and \( f_i^\gamma \) provides ‘navigational feedback’ towards a goal (combining motive and context forces). This definition of \( u_i \) varies from Olfati-Saber’s (2006).
original definition through the removal of the directional term $f_i^d$, relying instead on the navigational feedback provided through $f_i^g$.

**Flock force** The flock force, $f_i^g$, provides the cohesive force that pushes the agents into a lattice formation. It is defined by Olfati-Saber (2006) as:

$$f_i^g = -\nabla q_i V(q)$$ (2)

where $V(q)$ is an attractive/repulsive force based on the distance to the nearest neighbouring agents, and $q_i$ is the two-dimensional position of agent $i$. $f_i^g$ provides the force required to keep an agent $i$ a set distance from its neighbouring agents. The full details of the flocking algorithm are provided in Olfati-Saber (2006). By restricting the agents to only local interactions, the algorithm is applicable to both simulated and real (robotic) platforms. In order to provide $f_i^g$ to the swarm without global information, each agent projects a ‘virtual goal’ ahead of itself, which collectively provides the navigational feedback required to prevent the flock from disintegrating (Olfati-Saber 2006). This virtual goal is the key component of the ‘motive force’.

**Motive force** The motive force is provided through a ‘virtual goal’ in the environment. This is calculated by each agent projecting forward by a predetermined distance, $d$, and using that position in the environment as its goal for a set period of time, $G_{update}$. These parameters, $d$ and $G_{update}$, are set at the start of each simulation run, and do not vary throughout the run or between agents. Between them, these parameters provide a weighting towards motive or context for the agents (i.e. a weighting between the motive force and the context force).

Specifically, an agent $i$ calculates its virtual goal, $G_i$, by taking the average heading of its neighbours, $N_i$, within a pre-defined radius, $r$. Let the average heading of the neighbours of agent $i$ be $\phi_i$, defined as:

$$\phi_i = \frac{1}{|N_i|} \sum_{j \in N_i} \theta_j$$ (3)

then projecting forwards by the predefined distance, $d$, gives the virtual goal for an agent $i$ as the coordinate pair $G_i = (x_{G_i}, y_{G_i})$:

$$x_{G_i} = x_i + d \cdot \cos(\phi_i)$$ (4)
$$y_{G_i} = y_i + d \cdot \sin(\phi_i)$$ (5)

The goal is recalculated periodically so that the motive reflects the current state of the agent, including influences from the environment. This update period is parameterised as the virtual goal update interval ($G_{update}$). As the interval is increased, the agent is weighted further towards the motive than the context (and vice-versa), as information from the context will influence the position of the virtual goal less often.
Context force The environment is empty other than any attractor sources included. An attractor, \( j \), exerts a force on all agents in the environment. The force at any point can be described by the Gaussian probability density function, with mean \( \mu_j \), standard deviation \( \sigma_j \), and distance to the centre of the attractor from agent \( i \), \((q_i - \mu_j)\).

\[
f_{i}^{\text{context}}(\mu_j, \sigma_j, q_i) = \frac{1}{\sqrt{2\sigma_j^2 \pi}} e^{-\frac{(q_i - \mu_j)^2}{2\sigma_j^2}}
\]  

(6)

The agent calculates \( f^\gamma_i \) based on the coordinates of \( G_i \) and the gradient formed by the attractors in the environment:

\[
f^\gamma_i = f^{\text{motive}}_i(q_i, p_i, qG_i, pG_i) + f^{\text{context}}_i(\mu_j, \sigma_j, q_i)
\]  

(7)

Each agent multiplies the attractor gradient by a global parameter \( \text{ctx\_mult} \) as it senses it, in order to provide a weighting between \( f^{\text{motive}}_i \) and \( f^{\text{context}}_i \). See table 1 for a summary of each term used in the definition of \( f^\gamma_i \).

| Variable | Description               |
|----------|---------------------------|
| \( q_i \) | Two-dimensional position of agent \( i \) |
| \( p_i \) | Velocity of agent \( i \) |
| \( G_i \) | Position of virtual goal for agent \( i \) |
| \( \mu_j \) | Centre-point of attractor \( j \) |
| \( \sigma_j \) | Spread of attractor \( j \) |

Table 1: Descriptions of the variables used in equation (7)

| Parameter | Description               | Value | Units |
|-----------|---------------------------|-------|-------|
| \( \text{ctx\_mult} \) | Multiplier for attractor gradient | 185   | N/A   |
| \( d \)  | Distance from agent to virtual goal | 30    | patches |
| \( G\_update \) | Update period for virtual goal | 25    | timesteps |
| \( r \)  | Radius used to calculate neighbourhood | 8     | patches |

Table 2: Typical parameter values and descriptions for the CDM simulation. ‘Patches’ and ‘timesteps’ are generic terms for coordinate space in the simulation and discretised simulated time, respectively.

2.2 Correlation Matrix Memories

Associative memory is a mechanism for associating stimuli with responses. In biological systems, the memory results from temporal associations that emerge between two sets of interacting neurons ([Palm 1980][1981]). As the stimulus is
presented to the first set of neurons, the response of those neurons is sent to the second set of neurons, which subsequently respond. This second response is similar every time, indicating that the same response will occur if given the same stimulus.

The CMM (Correlation Matrix Memory) \cite{Kohonen1972} is a matrix-based representation of this neural network. The matrix represents the binary weights from a fully-connected, two-layer artificial neural network (one input layer, one output layer; see fig. 2b). As such, the network in fig. 2b would be represented by the CMM, $\mathcal{M}$, with $k$ input–output pairs $I$ and $O$ in fig. 2a.

\[
\begin{pmatrix}
I \\
O
\end{pmatrix}
\begin{pmatrix}
\mathcal{M}
\end{pmatrix}
\begin{pmatrix}
a \\
b \\
c
\end{pmatrix}
\begin{pmatrix}
ad \\
bd \\
ac \\
be \\
cd \\
be \\
ef \\
cf
\end{pmatrix}
\begin{pmatrix}
d \\
e \\
f
\end{pmatrix}
\]

(a) Basic CMM architecture (left), where $\mathcal{M}$ represents the matrix of binary weights, $I$ and $O$ represent the input–output pair corresponding to the neurons $a,b,c$ and $d,e,f$ in (b), respectively.

(b) CMM associative memory neural network. The binary weights between the two layers are represented by the matrix $\mathcal{M}$ in (a).

![Fig. 2](image)

Before training, the initial matrix $\mathcal{M}$ is filled with zeros (as there are no associations stored in the network). As the $k$ binary-valued input–output pairs are presented to the network, associations are built up in the matrix, $\mathcal{M}$. These associations are stored as 1s in the matrix, corresponding to coincident 1s in both input and output vectors. For example, in fig. 2a if $a$ and $c$ were both 1 then $ac$ would be set to 1 after training.
2.2.1 CMM in CogSis

In the CogSis architecture, we use a CMM to associate changes in the environment with changes in internal motive, following on from Cohen’s (2000) definition of cognitive decision-making:

“[A] decision emerges...from a match between an environmental case and an internal motive. Decisions are associations.” (p. 69)

He continues:

“Decision-making is positive action; instead of passively receiving what the environment imposes, the cognitive system exerts its will...in choosing among alternatives” (p. 69)

The associations made between the environmental case (referred to throughout this paper as ‘context’) and internal motive is made explicitly through the CMM, but also implicitly in the CDM algorithm described above.

The output from the CDM component (i.e. the robot motive) is passed as an input to the CMM. In the CogSis architecture the CMM stores an ‘internal image’—or imprint—of the environment for the robot to use. As such, the CMM offers the robot the capacity to make associations between changes in internal and external sensor values that occur at the same time. For example, if we mark a charging station with a blue light, the robot can sense the increase in blue light at the same time as an increase in battery charge. By associating these two signals, the CMM offers the ability to recall which colour to search for in order to find a charging station.

Fig. 3 shows how the CMM is constructed for two internal sensors (charge, $c$, and temperature, $t$) and for the three colour components from the external light sensor (red, green, and blue). When one of the sensors passes a threshold, the corresponding binary value switches from 0 to 1. If this occurs on one of the internal sensors at the same time as one of the external sensors, then the corresponding value in the association matrix is set to 1 as well, storing this association between the two sensors. Re-presenting one of the internal sensor inputs (e.g. ‘charge’), by setting it to 1 again will recall the association. This will cause the closest stored associations to be recalled, and the appropriate values set in the output vector. This allows the system to test which colours have been associated with which inputs.

3 Results

By comparing with a control case, we demonstrate that the CogSis architecture can...

RQ1: ...enable a robot to learn from previous experiences and use them to influence future behaviour
RQ2: ...provide homeostatic behaviour to a robot
RQ3: ...balance two conflicting requirements to provide homeostatic behaviour to a robot
Fig. 3: Basic CMM architecture (left), where $c$ and $t$ correspond to the internal sensors for charge and temperature, and $r$, $g$, and $b$ correspond to the red, green, and blue colour components from the light sensor, respectively. The vectors $(c t)$ and $(r g b)$ are provided as stimuli to the association matrix. Any points in the association matrix that has a 1 on both input vectors is set to 1, storing the association between the two. An example association is shown on the right, as would occur after the robot finds a charging station under a blue light.

This control case uses a replacement for the CDM that reacts to its internal state directly with a fixed threshold.

3.1 A robot running the CogSis architecture can learn from previous experiences

The CogSis architecture enables the robot to adapt to its environment through the use of a CMM. The CMM associates large changes in internal sensors with large changes in external sensors. For example, if the robot discovers a charging station under a green lamp, the CMM will associate green light with charging. While, in principle, the CogSis architecture is able to adapt online, this functionality is disabled in all experiments presented in this paper. This ensures that any variation between test and control cases are only due to swapping out the CDM. This section aims to demonstrate that a robot running the CogSis architecture can learn from its experience in an environment, and put it into practice in a simulated environment ahead of being used in real robot experiments.

The simulation is shown in fig. [4]. This simulation is constructed in NetLogo [Wilensky, 1999] and has the entire CogSis architecture implemented. The signals provided by the CDM to the rest of the architecture, however, are provided using switches on the simulator interface. The simulation consists of a single, zero-mass agent that roams around a simple environment of different light gradients. In place of the CDM, we provide a mechanism for signalling that the agent has a certain need. This ‘need’ is usually provided by the decision-making ability of the CDM. By signalling that the agent has a certain need, the CogSis architecture provides a mechanism for the agent to find the corresponding part of the environment. For example, ‘needs charge’
recalls the appropriate colour of light from the CMM and searches for that colour in order to recharge\footnote{we don’t provide the literal term ‘needs charge’ to the CogSis architecture, the system instead provides a signal that indicates it is running low on charge.}

**Simulated training** The agent roams around the simulated environment during its training phase, learning about the environment. Once this has been completed the system is ready to be tested. In this case, if the training is successful, the agent should head towards the green light when the ‘needs charge’ motive is switched on.

Fig. 4 shows a trace of the agent moving around the environment. Once the ‘needs charge’ switch is enabled (green arrow), the agent heads towards the green light source. Once this motive is disabled (blue arrow), the agent leaves the light source.

This shows that, in principle, the architecture has the capacity to learn about its environment.

Fig. 4: Screenshot of the proof-of-principle simulation, along with trace of the post-training agent searching for the green light source. The white arrows are the trace of the agent position, the green and blue arrows signify when the ‘needs charge’ motive switch is enabled and disabled.

The post-training behaviour of the simulation shows that the CogSis architecture is able to search for a specific part of the environment, based on...
high-level inputs such as a ‘needs charge’ signal. The architecture translates this signal back to a colour, based on previous experience of the environment. In other words, the architecture recalls ‘green’ as it learned that its batteries were charged when under a green light, which it subsequently seeks out. will be provided by the CDM algorithm when implemented into a robot.

Having shown that the architecture behaves as expected in a simulated environment, the next step is to implement this architecture on a robotic platform, and test whether this behaviour is consistent.

*Robot-based training* The CogSis architecture is implemented onto a robot (details of the robot platform and experimental setup are provided in appendix A). As before, the robot is allowed to roam around the environment during its training phase, learning about the environment. The environment (shown in fig. 5) consists of two coloured lamps (one red, one blue), and the CogSis is set up to simulate charging its batteries under the blue light, and to simulate an increase in temperature under the red light.

As the robot roams, the CMM associates high values from the internal sensors with high values from the external sensors. The threshold for external (colour) sensor values is 600 acu; for charging stations, 500 acu; for high-temperature areas, 300 acu (see appendix A for details of the colour unit ‘acu’). Once the robot has roamed across both areas of the environment with lamps in, the robot is taken from the arena and the contents of the CMM downloaded.

Due to the small size of the CMM used in this paper (3x2 matrix), the training phase can be tested with relative ease. In the following matrices, the following key can be used to determine whether the training has been successful, \(r, g, b\) correspond to the red, green, and blue external sensors, and \(c, t\) correspond to the charge and temperature internal sensors:

\[
\begin{pmatrix}
  c_r & c_g & c_b \\
  t_r & t_g & t_b \\
\end{pmatrix}
\]

After exposing the robot to an environment with one blue lamp over a charging station, the CMM is trained correctly as:

\[
\begin{pmatrix}
  0 & 0 & 1 \\
  0 & 0 & 0 \\
\end{pmatrix}
\]

After exposing the robot to an environment with both a red lamp over a high-temperature area and a blue lamp over a charging station, the CMM is trained correctly as:

\[
\begin{pmatrix}
  0 & 0 & 1 \\
  1 & 0 & 0 \\
\end{pmatrix}
\]
3.2 CogSis architecture provides homeostatic behaviour to a robot

This section considers how a robot running the CogSis architecture is able to alter its high-level behaviour according to its low-level needs. As the internal state of the robot varies, the CDM signals to the rest of the CogSis architecture what it needs in order to keep the state within its limits.

The CDM works to keep the internal sensors (charge and temperature) within certain limits. If the temperature gets too high or the charge too low then the robot will fail. The sensors are simulated so that the internal behaviour of the robot can be measured (if the battery actually ran out of charge, the data about internal behaviour might be lost or corrupted). The values used for different variables in our experimental setup are given in table 3.

|                       | Temperature | Charge |
|-----------------------|-------------|--------|
| Start value           | 10.0        | 5.0    |
| +ve Delta             | 1.8         | 0.8    |
| -ve Delta             | 0.5         | 0.1    |
| Fail Point            | 55.0        | 0.1    |
| Limits                | 10.0, 60.0  | 0.01, 15.0 |
| Control threshold     | N/A         | 2.5    |

Table 3: Parameter values used in the basic homeostasis experiment. The values have been picked to result in a challenging, but possible scenario.

This section looks at a basic scenario: asking whether the robot is able to keep itself charged while attending to another task. This other task is to warm itself under a lamp. By arranging the experimental arena (depicted in fig. 5) so that the charging station (a blue lamp) is far away from the warming lamp (a red lamp), the robot needs to leave the warming lamp in order to recharge, and vice-versa. This should result in the robot switching from sitting under one lamp to sitting under the other repeatedly and indefinitely. The parameters are chosen such that it is not able to warm itself sufficiently to complete its task before needing to charge again (thus cooling the robot back down again).

**Null Hypothesis (H₀):** A robot running the CogSis architecture will survive no longer with the CDM component than with the threshold component.

The design of the architecture described in section 2 was such that the CDM could be replaced by another component that provides different signals, based on the values of the internal sensors. The threshold component (described in section 2) is used in this way, and has a simple threshold mechanism. The output switches from 0 to 1 once the charge value drops below the control threshold of 2.5 (see table 3, respectively for temperature).

The threshold value for the control system was calculated from the amount of time it takes for the robot to get to the light source it is searching for. This was determined empirically as 45 seconds (the upper quartile of the data
Cognitively-inspired homeostatic architecture

Fig. 5: Photo of arena with charging and warm areas superimposed, arena size: 224x108cm

represented by fig. 6. Using this value for traversing the environment, the threshold can be calculated as 2.0. In order to take into account the variability of working in a noisy environment, the threshold was increased slightly to 2.5, allowing 54 seconds in the worst case for the robot to find its way between light sources. This also helps to reduce the chance of a false positive result by making it easier for the control setup to survive for longer.

Fig. 6: Boxplot showing how long it takes to cross the arena between light sources, measured empirically.

Fig. 7 shows the path of one run (a) with the CDM in place. The graph in (b) shows a subset of the internal sensor data provided to the CDM component for the duration of the experimental run. The robot was allowed to continue
running until it reached 29 minutes, at which point it would be stopped. The traces show that the robot is able to perform a task while preventing itself from running low on charge.

Table 4 shows the length of time that the robot survived when set up with the two components described above (up to the maximum of 29 minutes). This data is presented in a table format, rather than a boxplot due to the relatively few replicates in this experiment. The high variance in the CDM data is unexpected and can be explained by the robot getting stuck in a corner early on in the run and failing as a result. These ‘failed’ runs are still included to ensure the results are not biased by their removal. Even with only 6 replicates, it is clear that the CDM code can survive for longer than the control code. This is confirmed through the Mann–Whitney–Wilcoxon test and Vargha–Delaney A-test, giving $p < 0.05$ and $A < 0.16$ respectively. From this, $H_0$ can be rejected at the 95% confidence level.

| Replicate | Test (mins) | Control (mins) |
|-----------|------------|----------------|
| 1         | 3.5        | 29             |
| 2         | 4          | 2.5            |
| 3         | 2          | 29             |
| 4         | 2.5        | 6              |
| 5         | 10         | 29             |
| 6         | 3          | 29             |

Table 4: Details of how long the robot survived in the environment when using the CDM (test) component and the threshold (control) component. Mann–Whitney–Wilcoxon test and Vargha–Delaney A-test give $p < 0.05$ and $A < 0.16$ respectively, rejecting $H_0$ with 95% confidence.

While performing a relatively simple task, the results presented in this section show that the CogSis architecture is a viable option for providing simple artificial homeostasis to a robot. This provides the evidence required to answer the research question RQ2. The results here also suggest that the robot is able to make use of its previous experience in the environment, and apply it to survive in the same environment. This provides further evidence towards answering research question RQ1. The next section discusses how well the architecture handles more complicated scenarios that involve conflicting needs.

### 3.3 Conflicting decisions

While the work above shows how the CogSis architecture could be used to control a robot in a simple environment (two distinct light sources), with a simple task (survive as long as possible, while performing a basic task),

---

2 29 minutes comes from the maximum amount of time the camera could record for
(a) Trace of robot path between 5.5 and 10.5 mins superimposed on the environment. The blue and red circles indicate points where the robot stopped to charge or warm itself respectively. The green and white circles indicate the start and end points. The red and blue solid lines indicate that the robot is searching for that colour. The white dotted line is when the robot wanders (not actively searching for anything).

(b) Graphs showing internal sensor values (top) and CDM outputs (bottom) for the robot between 5.5 and 10.5 mins. The red and blue line show the internal values for temperature and charge as presented to the CDM component. The red and blue blocks of colour in the bottom graph show the output of the CDM component (red indicating to the robot to search for red light, and blue indicating to search for blue light).

Fig. 7: A short (5 min) excerpt from one of the 29-minute replicates. The trace (a) shows how the robot moves between the red and blue areas in the environment as it needs to charge. The graphs in (b) show that as the internal sensor values for charge get too low, the robot actively searches for the charging station, and once it is charged sufficiently, it returns to warming itself under the red lamp.
this section addresses research question RQ3: can the CogSis-controlled robot survive when faced with conflicting decisions about what to do?

In order to achieve this, the experimental setup is altered such that the charging station and the higher temperature area are provided by the same light source (see fig. 8). The behaviour of the CDM is inverted with respect to temperature, so that the robot now tries to avoid getting warm. Therefore, in order to charge its battery, the robot will have to endure warmer temperatures. Once the battery runs out, the robot will fail, and once the temperature reaches a set maximum, the robot will fail. This experiment once again makes use of the same architecture between test and control cases, except for the CDM component that will be replaced by the threshold component for the control case.

Null Hypothesis ($H_0$): A robot running the CogSis architecture will survive no longer with the CDM component than with the threshold component in the presence of conflicting needs.

The experimental setup is as shown in fig. 8. This is the same arena as above, except for the removal of the blue light from the right-hand side. In this more difficult scenario, the robot is expected to fail more often than it did in the previous setup, as before it was possible for the robot to sit and charge indefinitely without failing, whereas here this would result in a failure. After preliminary testing, it was evident that the parameter values used previously (given in table 3) provided a scenario that was too difficult for the robot to complete with either CDM or threshold-based architecture (all survival times were under 2.5 minutes). As such, the parameter values are adjusted to those shown in table 5, to make the robot less likely to overheat straight away, but balancing this by making it slower to charge.

Fig. 8: Photo showing the new arena for conflicting decisions experiment, arena size: 224x108cm
Table 5: The parameters used in the ‘conflicting decisions’ experiment. These values have been picked to result in a challenging, but possible scenario.

|                  | Temperature | Charge |
|------------------|-------------|--------|
| Start value      | 10.0        | 5.0    |
| +ve Delta        | 1.3         | 0.3    |
| -ve Delta        | 0.8         | 0.1    |
| Fail Point       | 55.0        | 0.1    |
| Limits           | 10.0, 60.0  | 0.01, 15.0 |
| Control threshold| 25.0        | 2.5    |

Fig. 9 shows a trace of the sensor values as provided to the CDM component. The graphs show how the robot balances the need to charge its battery while avoiding getting too warm. The robot with CDM component was able to balance the conflicting decisions well, failing only once in 13 replicates, whereas the control case failed 6 times out of 13. These results are presented in the boxplots in fig. 10.
(a) Trace of robot path between 14 and 19 mins superimposed on the environment. The blue and red circles indicate points where the robot stopped to charge or cool itself respectively. The green and white circles indicate the start and end points. The blue and red solid lines indicate that the robot is searching for, or fleeing from, the red light, respectively. The white dotted line is when the robot wanders (not actively searching for anything).

(b) Graphs showing internal sensor values (top) and CDM outputs (bottom) for the robot between 14 and 19 mins. The red and blue lines show the internal values for temperature and charge, as presented to the CDM component. The red and blue blocks of colour in the bottom graph show the output of the CDM component (red indicating to the robot to flee red light, and blue indicating to search for red light).

Fig. 9: A short (5 min) excerpt from one of the 29-minute replicates. The trace (a) shows how the robot moves in and out of the red area in the environment as it needs to charge and cool down. The graphs in (b) show that as the internal sensor values for charge get too low, the robot actively searches for the charging station, and vice-versa for when the temperature gets too high. Note that, contrary to the behaviour described in fig. 7, when the temperature gets too high at around 17 mins, the CDM does not actively push the robot away from the red light, as it is already out of the area by chance. This has the effect of starting to reduce the temperature, reducing the likelihood that the CDM component would push the robot to seek out a cooler area.
The boxplots in fig. 10 show the amount of time that the two setups survived (up to the maximum of 29 minutes). There are 13 replicates for each. Results from the control code are presented on the left, and from the CDM (test) code on the right.

Fig. 10: Boxplots showing the distribution of survival times from the conflicting decisions experiment. The two distributions are significantly different, with $p = 0.03$ and $A = 0.30$.

While these results were expected to be less clear-cut than the previous experiment, analysis gives similar results. The Mann–Whitney–Wilcoxon test gives $p = 0.03$, and the Vargha–Delaney effect-magnitude test gives $A = 0.30$. From this, $H_0$ can be rejected with 95% confidence, resulting in the conclusion that the CDM component is better at balancing conflicting decisions than the threshold component, and providing sufficient evidence to answer research question RQ3. As with the previous experiment, the results here suggest that the robot is still able to make use of its previous experience in the environment, and apply it to survive in the same environment. This provides even further evidence towards answering research question RQ1.

4 Conclusion

This paper has presented a new approach to developing robotic homeostasis. By taking inspiration from cognition, rather than cellular-level biology, a simple homeostatic architecture has been developed for a robot which is able to balance conflicting needs. Evidence towards answering three research questions is presented. These research questions are summarised below.
RQ1 (Can the CogSis architecture enable a robot learn to from previous experiences and use them to influence future behaviour?) Section 3.1 presents the testing of the CMM training phase. This process consists of the robot roaming around an environment and associating simultaneous changes in the internal and external sensors. These associated changes represent the previous experiences that the robot has learnt. The behaviour of the robot in sections 3.2 and 3.3 show that the robot is able to successfully recall these experiences and use them to influence its high-level behaviour.

RQ2 (Can the CogSis architecture provide homeostatic behaviour to a robot?) Section 3.2 presents results showing that the robot is able to alter its high-level behaviour according to the value of two internal sensors (temperature and charge). The behaviour of the robot is compared between two versions of the CogSis architecture: one with the CDM component, one with a simple threshold function (see section 2). The CDM component consistently outperforms the threshold function at providing homeostatic behaviour to a robot, as shown by the survival task in fig. 7.

RQ3 (Can the CogSis architecture balance two conflicting needs to provide homeostatic behaviour to a robot?) Section 3.3 presents results showing that the robot is able to balance two conflicting needs. The robot was required to withstand a high-temperature region in order to gain access to a charging station. The behaviour of the robot is again compared between two versions of the CogSis architecture as with RQ2, above. The CDM component once again consistently outperforms the threshold function at providing homeostatic behaviour to a robot, while balancing two conflicting needs.

While other approaches to homeostasis have directly mimicked the natural systems of the body (Neal and Timmis 2003, 2005; Vargas et al. 2005; Stradner et al. 2009; Schmickl et al. 2011), the CogSis architecture has been developed by taking inspiration from cognition (Cohen 2000; Mitchell 2005). Combining the CDM (cognitive decision making) with a CMM (Correlation Matrix Memory) associative memory provides the CogSis (COGnitively-inspired HomeostaSIS) architecture with the ability to alter its high-level behaviour according to the low-level state of the robot, while based on previous experiences.

The CMM is trained by coincident ‘spikes’ in the internal and external sensors (e.g. when the ‘charge’ internal sensor rises at the same time as the ‘blue’ external sensor—this relates to a charging station under a blue light in the real-world environment). This provides adaptivity to the robot, as it is able to learn about new parts of an environment and adapt its behaviour accordingly (for example, if the robot were to find another charging station, it would store the association in the CMM and recall it as before).

The robot is able to make decisions about its current internal state through the CDM and transfer this to high-level behavioural changes, showing that the CogSis architecture provides the capacity for homeostasis to the robot. Furthermore, the use of the CMM to recall previous experiences allows the robot
to proactively search for, or flee from, specific areas in the environment. This proactivity allows for behaviour that is more realistic for real-world applications.

The modular nature of the CogSis architecture, in that the CDM can be easily replaced by a different decision-making mechanism, makes the CogSis architecture more flexible for different applications. It could be possible to run multiple decision-making algorithms as an ensemble, with the combination of their outputs being passed to the CMM.

There is also potential for online learning about an environment, where the CMM would be training and recalling simultaneously. Furthermore, the sharing of experience between multiple robots working in the same area could be possible by sharing the contents of CMMs between robots.

In summary, cognition is a viable source of inspiration for homeostasis in robots. The resulting architecture shows that there is significant potential from implementing cognitive behaviour into robots, giving the ability to balance conflicting needs more effectively than a conventional threshold mechanism.

A Experimental setup

The arena is 224x108cm with 15cm high boundaries (see the background of fig. 7a). It has coloured LED lamps that can be moved so they point at different parts of the arena.

The robot platform is the Pi-Swarm [Hilder et al. 2014]. This platform provides basic IR distance sensing and wheeled movement, along with colour LEDs on the top edge of the robot, and a simple speaker built-in. The robot is controlled using an ARM mbed LPC1768 chip [mbed 2016] which runs custom C++ code. It has very limited flash and RAM storage, along with limited computational capacity. In order to provide information to the robot about the environment, the Pi-Swarm platform has a single RGB colour sensor (TCS3472) mounted on top.

LED lamps, mounted on tripods, provide a gradient of colour for the robot, using Perspex colour filters over the front in order to separate the different gradients. After initial analysis the best colours for this laboratory setup (in terms of other lights in the surrounding area and the contrast in the sensor) is a combination of the orange and yellow-green filters (referred to as ‘red’ throughout the paper for simplicity), or the blue filter.

For the purposes of this work, the robot makes use of two internal sensors: battery charge and temperature. These sensors are simulated, exploiting the fact that they are always beneath a lamp. As such, by setting the threshold for ‘detecting’ a charging station higher than the threshold for detecting the corresponding colour, the charging station is placed at the centre of the lamp’s gradient. The colour sensor returns values that are scaled according to the integration time and gain of the sensor (see [Industries 2012] for full details). The integration time and gain used for this experimental setup are 14.4ms and 60x, respectively. The colour sensor returns values based on the irradiance of its sensor, which is dependent on a wide range of environmental factors. The values used are proportional to lux, but it makes more sense to report values in terms of the relative responsivity of the sensor. For this reason, we define the ‘arbitrary colour unit’ (acu) as the value returned from the TCS3472 given an integration time of 14.4ms and gain of 60.

B Acknowledgements

The work in this article was performed during JHS’ PhD [Stovold 2016], funded by an EPSRC Doctoral Training Grant. A copy of JHS’ thesis is available through the White Rose eTheses Online portal.
References

Ashby W (1960) Design for a brain: The origin of adaptive behaviour. Chapman and Hall, London

Bernard C (1878) Les phénomènes de la vie. Bailliere, Paris 879:1

Boden MA (2008) Autonomy: What is it? BioSystems 91(2):305–308

Cannon WB (1929) Organization for physiological homeostasis. Physiological Reviews IX(3):399–431

Cohen IR (1992a) The cognitive paradigm and the immunological homunculus. Immunology Today 13(12):490–494

Cohen IR (1992b) The cognitive principle challenges clonal selection. Immunology Today 13(11):441–444

Cohen IR (2000) Tending Adam’s Garden: evolving the cognitive immune self. Academic Press

Froese T, Virgo N, Izquierdo E (2007) Autonomy: A review and a reappraisal. In: Almeida e Costa F, Rocha LM, Costa E, Harvey I, Coutinho A (eds) Advances in Artificial Life: 9th European Conference, ECAL 2007, Lisbon, Portugal, September 10-14, 2007. Proceedings, pp 455–464

Garnier S, Jost C, Jeanson R, Asadpour M, Caprari G, Theraulaz G (2005) Collective decision-making by a group of cockroach-like robots. In: Swarm Intelligence Symposium, 2005. SIS 2005. Proceedings 2005 IEEE, pp 233–240

Godfrey-Smith P (1998) Complexity and the Function of Mind in Nature. Cambridge Studies in Philosophy and Biology, Cambridge University Press

Hilder J, Naylor R, Rizihs A, Franks D, Timmis J (2014) The Pi Swarm: A low-cost platform for swarm robotics research and education. In: Mistry M, Leonardis A, Witkowski M, Melhuish C (eds) Advances in Autonomous Robotics Systems, Lecture Notes in Computer Science, vol 8717, Springer International Publishing, pp 151–162

Industries A (2012) Color Light-To-Digital Converter with IR Filter. ams

Kohonen T (1972) Correlation matrix memories. Computers, IEEE Transactions on C-21(4):353–359

Lewontin RC (1957) The adaptations of populations to varying environments. Cold Spring Harbor Symposia on Quantitative Biology 22:395–408

mbed (2016) mbed LPC1768. Online: https://developer.mbed.org/platforms/mbed-LPC1768/

Mitchell M (2005) Self-awareness and control in decentralized systems. Metacognition in Computation pp 80–85

Neal M, Timmis J (2003) Timidity: A useful emotional mechanism for robot control? Informatica 27(2):197–204

Neal M, Timmis J (2005) Once more unto the breach...towards artificial homeostasis? In: De Castro LN, Von Zuben FJ (eds) Recent Developments in Biologically Inspired Computing, Idea Group Pub.

Olfati-Saber R (2006) Flocking for multi-agent dynamic systems: algorithms and theory. Automatic Control, IEEE Transactions on 51(3):401–420

Palm G (1980) On associative memory. Biological Cybernetics 36(1):19–31

Palm G (1981) Towards a theory of cell assemblies. Biological Cybernetics 39(3):181–194

Schillo M, Fischer K (2004) A taxonomy of autonomy in multiagent organisation. In: Nickles M, Rovatsos M, Weiss G (eds) Agents and Computational Autonomy: Potential, Risks, and Solutions, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 68–82

Schmickl T, Hamann H (2011) BEECLUST: A swarm algorithm derived from honeybees. Bio-inspired Computing and Communication Networks CRC Press (March 2011)

Schmickl T, Hamann H, Crailsheim K (2011) Modelling a hormone-inspired controller for individual- and multi-modular robotic systems. Mathematical and Computer Modelling of Dynamical Systems 17(3):221–242

Stovold J (2016) Distributed cognition as the basis for adaptation and homeostasis in robots. PhD thesis, University of York

Stovold J, O’Keefe S, Timmis J (2014) Preserving swarm identity over time. In: Sayama H, Rieffel J, Risi S, Doursat R, Lipson H (eds) Proceedings of the 14th international
conference on the synthesis and simulation of living systems (ALIFE ’14), pp 728–735
Stradner J, Hamann H, Schmickl T, Crailsheim K (2009) Analysis and implementation of an artificial homeostatic hormone system: A first case study in robotic hardware. In: 2009 IEEE/RSJ International Conference on Intelligent Robots and Systems, pp 595–600
Tyrrell T (1993) Computational mechanisms for action selection. PhD thesis, University of Edinburgh
Varela FJ (1981) Autonomy and autopoiesis. In: Self-organizing systems: An interdisciplinary approach, Campus Verlag, pp 14–24
Vargas P, Moioli R, Castro LN, Timmis J, Neal M, Zuben FJ (2005) Artificial homeostatic system: A novel approach. In: Capcarrère MS, Freitas AA, Bentley FJ, Johnson CG, Timmis J (eds) Advances in Artificial Life: 8th European Conference, ECAL 2005, Canterbury, UK, September 5-9, 2005 Proceedings, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 754–764
Vernon D (2015) Artificial Cognitive Systems. MIT Press
Widmaier EP, Raff H, Strang KT (2006) Vander’s human physiology, vol 5. McGraw-Hill New York, NY
Wilensky U (1999) NetLogo. http://ccl.northwestern.edu/netlogo/ Center for Connected Learning and Computer-Based Modelling, Northwestern University, Evanston, IL.
Yu CH, Werfel J, Nagpal R (2010) Collective decision-making in multi-agent systems by implicit leadership. In: Proceedings of the 9th International Conference on Autonomous Agents and Multiagent Systems: Volume 3, International Foundation for Autonomous Agents and Multiagent Systems, AAMAS ’10, pp 1189–1196