Evolution of Consciousness Systems with Bacteria

Daniela López De Luise[1], Lucas Rancez[1], Marcos Maciel[2], Bernardo De Elía[3], Juan Pablo Menditto[3]
[1]CI[2]S Labs, Buenos Aires, Argentina
[2]CAETI, Buenos Aires, Argentina
[3]Escuela Superior Técnica del Ejército, Buenos Aires, Argentina

Abstract.

Disasters are characterized by extreme situations where human-made structures are partially or completely destroyed. Earthquakes, landslides, cyclones, floods are that type of phenomena. This paper presents an outdoor autonomous mobile robot named MARCOS, which uses consciousness (set of codelets) to derive its world map and avoid obstacles. Disaster zones constitute a dramatic environment where obstacles and dangerous conditions require extra flexibility and good reasoning capabilities upon unexpected situations. The aim of this work is to show statistically how performs traditional consciousness for complex avoidance and compares it with solutions provided by a human following a novel approach called artificial bacterial infection. Results show that bacteria are a good artifact to enhance original robotic consciousness, providing originality, and mainly variety in the strategies considered by the robot. Therefore, it increments the possibility to devise wiser paths that will compete with traditional ones. A unique type of evolutive consciousness emerges from this: not only considering current codelets generating consciousness but also evolving them into diverse paths and even generating new ones. As part of this paper's scope, the basics of such artificial bacterial infection is also presented.

Keywords

Autonomous Mobile Robots, Consciousness, Bacteria, Disaster Zone, Obstacle Avoidance, Obstacles Detection

1. Introduction.

Ever since people involved robots in the human life [1, 2] to do repetitive tasks which were previously executed by operators, more specific responsibilities have been developed. For example, saving lives in disaster zones which are characterized by extreme situations.
In these situations, the structure created by people or those created by nature, are partially or completely destroyed by earthquakes, landslides, cyclones, floods, fires etc. To overcome these obstacles, innumerable jobs were implemented to grasp these new challenges.
Robots are used to assist and cooperate with people in many applications and projects. A list of examples ordered in increasing complexity is listed below:
-Assist people in their daily routines [1, 2]. E.g.: Medication administration, event notification for a doctor's appointment.
-Groceries online shopping according to a list on a smart fridge.

-Autonomous navigation by corridors between plants to replace repetitive agricultural tasks [3]:"The robot starts from an initial point, makes its way between plants until it finishes its goals".
-Environment survey to provide security [4]: "This robot goes through a defined path and detects objects movements to alert about possible threats".
-Proposed in [5] there is an autonomous robot which detects previously set up objects in real time, for instance, a white line on the street goes through a path until it finds a stationary object blocking a pedestrian path. Such examples have no autonomous learning or risky tasks.
-Overcome its own mechanic limitation [6]: "authors created a robot which can climb over a bigger object than the wheel radius. An arm is used to assist itself, useful when there is no alternative path to go forward."

There is much research focused to improve the robots technical skills. Some proposals are accomplished by mathematical and geometric algorithms such as recognizing objects through environment semantic segmentation where these objects can be physically represented. It is possible to create a classification of the objects and then quickly take a decision about the path to take because the robot has a higher level of certainty about the environment [7]. As an alternative, detecting and reconstructing a 3D environment using an ICP algorithm based on images edges characteristic is proposed [8]. On the other hand with a camera assembled on the robot to record the area characteristics and working out the virtual barycenter, it is possible to determine the direction when a robot is moving [9, 10]. Thanks to barycenter update the robot can recalculate its position about this new barycenter.
There are researchers who classify objects as obstacle or non-obstacle depending on its color [11]. Another method of working with environment information is dividing it (based on laser) in inspection and direction through an algorithm. Then the information is then rebuilt from each recorded frame taking into account the shape of the robot to work out the angle area.
As previously stated, when mathematical algorithms are applied over collected information from devices, skills are provided to robots to overcome limitations and to handle situations that take place in risky environments.
These robots need to complete the following tasks to be useful within a dangerous environment: read volumetric information of the area, read ambient temperature and gas concentration, image analysis and processing, detect danger around the area, send information to an operator, make contact with survivors, etc.
To improve performance, some strategies are listed below:

- A thermal sensor is attached to the robot that detects temperature human body in a Radial Basis Function Network trained with Dynamic Decay Adjustment model. When a risky situation for the subject is detected, it identifies and isolates the survivor [12].
- That the robot may be able to detect human body heat with a mounted thermal camera is used to detect human body temperature based on Radial Basis Function Network trained with Dynamic Decay Adjustment model to identify and pinpoint survivor locations.
- Another approach related to networks is to determine master nodes conforming a set of communication cells to improve the operator's communication [13].
- An effective method is the use of a data vector conformed by different data gathered by many different sensors in a single robot such as VR, intelligent detection, telecommunications, virtual simulation, environment dynamic modeling, ultrasonic sensor, laser radar, etc. managed and guided by a human operator . As a case study, robot SALVOR [14] is well suited to protect sensors from harsh conditions and weather (dust, water or high temperatures).
- Some robots are outfitted with wheels and arms [15] not only sensors since they were conceived to assist people and to communicate with ISM (Industrial, Scientific, and Medical) radio band.
- A specific case is a fire-oriented robot [16] that uses different nodes to establish a proper wireless communication with MANET.

Among other complex tasks, robots dynamically recognize the environment, build a world-map in real time and move avoiding obstacles. The last activity is definitely the most difficult. There are many relevant advances in this field, some of them detecting concavities in the obstacles [17], merging sensor information to generate better short/long distance world-maps [18], etc.
The millimeter wave radar allows to collect information even when weather conditions are difficult, and therefore makes it possible to build an updated world-map [19]. Combining different sensor data and algorithms like EKF they produce recommendations [20]. Those EKF and SLAM approaches are used to auto-localize and plan the shortest path to the objective [21].
An additional example is the auto-localization using minimal aggregated sensor error [22], implementing a probabilistic localization based on the metric-topology cartography. The multithreading processing [23] allows relating information from many sources.
There are robots with important equipment and volume, which allows them to operate in very harsh conditions, self-adaptable and self-reconfigurable, all with an advanced processing capability [24, 25].
The robot prototype MARCOS implements the learning concept based on technology consciousness via codelet. Each of them stores environment information and allows the COFRAM interface to issue recommendations and path plans to avoid objects and achieve the goal. The RTA (Robot Task Adviser) takes action for the long-term decisions. These decisions are generated with data collected from the robot sensors, which is processed and subsequently sent to the robot.
The interface has a built-in set of strategies and dynamically-created ones upon consciousness learning. By combining the Bacteria Theory with RTA consciousness process, a better performance for the codelet is expected, making it more flexible and more adaptive model when facing dynamic environments such as disasters zones.
The autonomous mobile Robot MARCOS (Movement of Autonomous Robotics Codelet System) is based on the consciousness model. It uses a kernel named COFRAM (COdelet FRAMework) to implement the concept of learning (set on consciousness technology) to propose recommendations for guiding the robot movements [26] in disaster zones.
It is proposed in this paper to compare the Consciousness Model implemented by MARCOS and a new approach named Bacterial Infection. The test outcome allows determining that it is possible to evolve the present consciousness model through originality, new strategies to be considered by the kernel and a better path planning derived from a more precise in real time world-map building.
The rest of this paper is organized as follows:

- Section 2: Exposition of the working hypothesis, working thesis and Consciousness theory as well as MARCOS robot features.
- Section 3:Details of the experimental results are exhibited
- Section 4: Final considerations and future work are presented.

2. Proposal

Appearance variability and unpredictability, environmental variations and sensing noise or unwanted signals are some examples of the wide variety of issued faced by robotics in disaster zones.
Approaches like the ones proposed in [27] or in [28] can work in a close and predefined environment, as they rely on predefined concepts and structured portions of code, but they are not suited for the highly dynamic situations that a disaster environment presents.
The prototype MARCOS uses COFRAM as its kernel which implements concept learning bases on technology to process the robot's movements.

The architecture is mainly conformed by an RTC (Real Time Controller) for immediate decision and a RTA (Robot Task Adviser) for middle to long-term strategies [12]. It can be seen in Fig.01

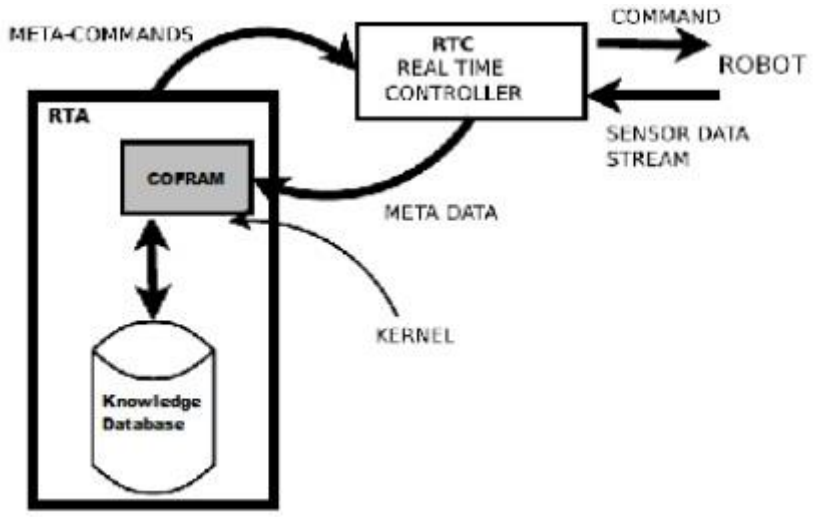


Fig. 01. COFRAM architecture

The next subsection follows with a description of a working hypothesis, working thesis, Consciousness Theory and finally Bacteria Theory adapted to the present context.

2.2.1. Working Hypothesis

Hypothesis 1: The bacteria behavior allows to generate original alternatives to solve movement strategies in the system context based on consciousness.

Hypothesis 2: The bacteria behavior allows to generate robust alternatives to solve movement strategies in the system context based on consciousness.

Hypothesis 3: The movement strategies' generation process conducted by consciousness is appropriate in real time.

Hypothesis 4: The consciousness model can build an acceptable world map in real time without the need of a previously loaded one.

Hypothesis 5: The consciousness model is flexible to self-adapt to environment changes automatically.

Hypothesis 6. The autonomous movement model can consider two dimensions to a reasonable work in general terms.

Hypothesis 7. The robot autonomy must be such that allows independent management for dynamic situations in disaster zones. I.e.: exception of an abrupt level difference in the path.

Hypothesis 8. The robot is able to record the path, through distance and angle in movement to work out its localization at any given time.

2.2.2. Working Thesis

Hypothesis 1: The infection involves adding up additional behavior to codelets or set up a pre-defined function. The behavior is dual as stated below:

a) Given by common actions based on consciousness systems (implemented with codelets)
b) Given by the Bacteria model creativity which alters the a) type actions.

Hypothesis 3: The strategies generation process of movement conducted by consciousness is appropriate in real time. It has been demonstrated in [12].

Hypothesis 4: It is also demonstrated in [12], the model based on consciousness builds a world map in real time y successfully avoids objects.

Hypothesis 5: The consciousness model is flexible to self-adapt to environment changes automatically. In [12] is shown the traces where it is possible to verify that the robot can move between different traces. Additionally, the robot achieves the goal.

Hypothesis 7: The robot autonomy must be such that allows the independent management of dynamic situations in disaster zones like an abrupt level difference in the path. The autonomy conditions [16, 29, 4], ability to avoid (Hypothesis 4) and change autonomously strategy [12].

Hypothesis 8: The robot is able to record the path, through distance and angle in movement to work out its localization at a time x.
It's a standard [12].

2.2.3. Consciousness Theory

Figure 02 shows the consciousness dynamics and Figure 03 the design for its implementation.

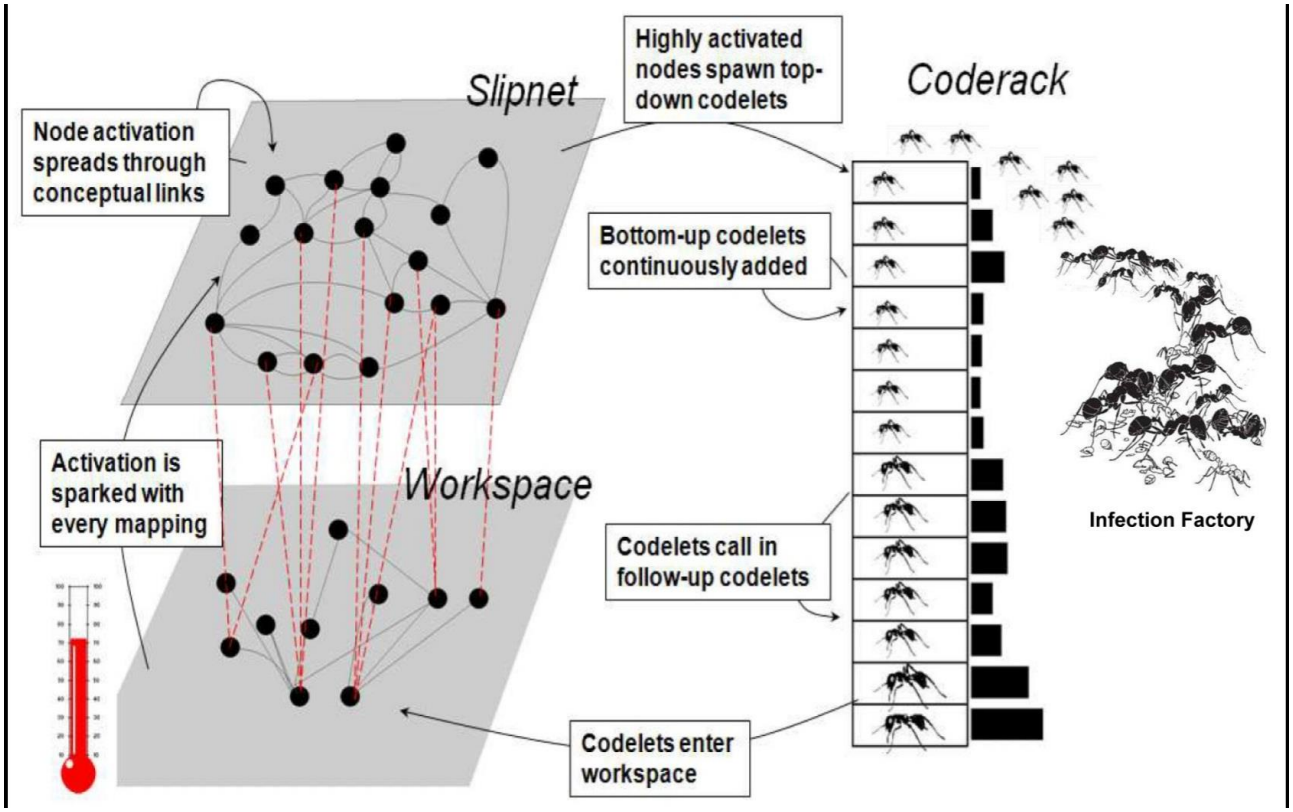


Fig. 02. Consciousness theory.

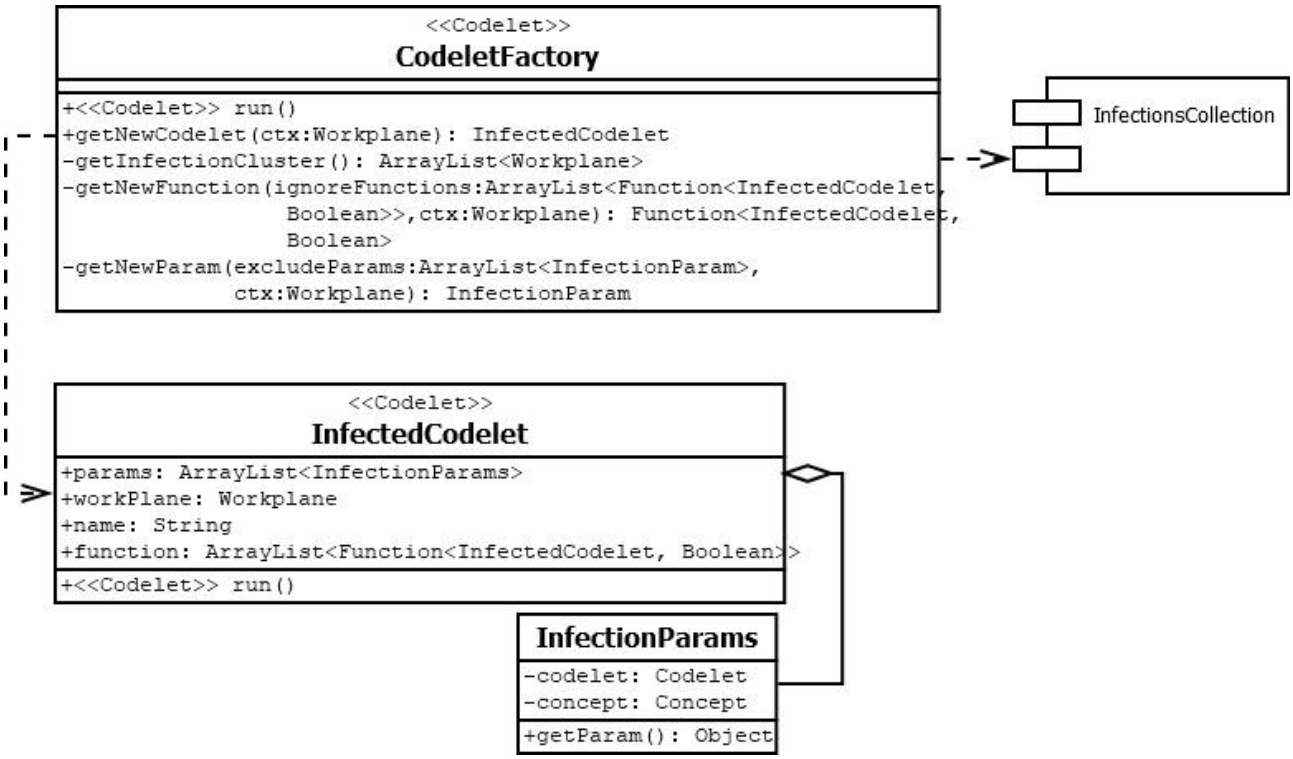


Fig. 03 Code implementation to Consciousness theory

According to that, a set of simple small parts of code (named codelets) are probabilistically fired by a manager named coderack. When active, they enter the workspace to produce certain reinforce or construction of concepts and/or a predefined relationship between them. When a concept (denoted as "node" in fig. 02) is being recalled (activated) many times, it becomes a long-term concept and migrates to the slipnet.

2.2.4. Development of the bacteria theory.

This section introduces the basis of **Consciousness Systems with Bacteria (CSB)** adapted to the robot context guided by consciousness elements (codelets).

Fundamentals of the theory **CSB** are the following five steps:

STEP 0. Components set up

All components in CSB are vectors that evolve as bacteria expands. Vectors must represent the key functional processes in the problem, its parameters and problem constraints:

VS(viral system) =($VIRUS_i$, $ORGANISM_i$, $PROCESS_i$)
$VIRUS_i$ =($STATE_i$, $INPUT_i$, $OUTPUT_i$, $PROCESS_i$)
$ORGANISM_i$ = ($STATE_i$, $PROCESS_i$)

With:
$STATE_i$=($PARAM_1$, $PARAM_2$, …) are virus characteristics, actually codelet parameters
$INPUT_i$ = ($CONCEPT_1$, $PERCEPTS_2$, $CODELET_3$, …) is the organism information that virus compiles in order to keep track of key functional processes. In the conscious context it is the set of active/passive codelets that perform any activity in the system, and any percept (that is, a concept that comes from environment sensing)

$OUTPUT_i$ =($CODELET_1$, $CODELET_2$, …) is the result of virus activity (represented by $PROCESS_i$). Typically, for conscious a new set of codelets with unprecedented activity. The new set is expected to produce a change in the system's behavior. In the current context, it means a different ability to learn concepts.
$PROCESS_i$=(FUN_1, FUN_2, …) virus activity that changes the consciousness functioning, leading to a $STATE_i$
$STATE_0$ = organism state at initial moment, it also declares the current status (named clinical profile) and prospective evolution (also referred to as minimum health required)
$PROCESS_0$ = current conscious model. In biological terminology body-defense, related to the freedom of antigens to protect itself from the infection threat. The application here refers to the capability to change codelets abilities

STEP 1- Vectors initialization
Every vector is assigned a value according to the outlines described in the previous step.

a) Set $INPUT_i$ = ($CONCEPT_1$, $PERCEPTS_2$, $CODELET_3$, …):

$INPUT_i$=[
getHandler(Codelet$_1$), // CODELETs
getHandler(Codelet$_2$),
...
getHandler(Codelet$_M$),
getHandler(Percept_actual$_1$), //PERCEPTs
getHandler(Percept_actual$_2$),
…
getHandler(Percept_actual$_n$),
getHandler(ConceptName_actual$_1$ (Class, argv$_1$...argv$_k$)), // CONCEPTs
getHandler(ConceptName_actual$_2$ (Class, argv$_1$...argv$_k$)),
…
getHandler(ConceptName_actual$_m$ (Class, argv$_1$...argv$_k$)),
]

b) Set $STATE_i$ = ($PARAM_1$, $PARAM_2$,…), that is declare codelets' parameters to change their behavior.

$STATE_i$=[
getHandler(Val_sensor_ultrasound$_1$),
getHandler(Val_sensor_ultrasound$_2$),
getHandler(Val_sensor_RPLIDAR$_1$),
getHandler(Val_sensor_accelerometer$_1$),
getHandler(Val_sensor_accelerometer$_2$),
getHandler(Val_sensor_accelerometer$_3$),
getHandler(Percept_actual$_1$),
getHandler(Percept_actual$_2$),
getHandler(Percept_actual$_3$),
getHandler(Percept_actual$_4$),
…
getHandler(Percept_actual$_n$),
getHandler(ConceptName_actual$_1$ (Class, argv$_1$...argv$_k$)),
getHandler(ConceptName_actual$_2$ (Class, argv$_1$...argv$_k$)),
…
getHandler(ConceptName_actual$_m$ (Class, argv$_1$...argv$_k$)),
]

c) Set $PROCESS_i$=(FUN_1, FUN_2, …)
FUNCIONS ={
getHandler(FUN$_1$), // modify sensors values in percepts

getHandler(FUN$_2$), // modify concept arguments total
getHandler(FUN$_3$), // associate two exists codelets
getHandler(FUN$_4$), // duplicate codelets
getHandler(FUN$_5$), // change using percept y concepts, using the same function
getHandler(FUN$_6$), // concept that gets others concepts and lose any fragment from it-self or build a more simple concept (lose variables)
}

As mentioned previously, in this context the process represents the change of the current codelets set by other codelets (the new ones produced by the infection activity). Several functions for new codelets were considered. Many of them for understanding the shape of eventual obstacles:

FUN(cell) : adjacent? -> pair
FUN(adjacent): continuity? -> collection of adjacent continuous pairs
FUN(continuity): linearity? -> test 4 direct address +x,+y +x,y x,+y +x,-y sabe first and last point
FUN(linearity): curves? -> test lines' adjacent (sharing the first or last point): curves as couples line$_i$, line$_j$
FUN(curves): concave? -> test angle < 180 named CCV
FUN(CCV): concave_continuity? -> common extremes from two or more CCV named concave_curve
FUN (curves):convex? -> test angle > 180 named CVX
FUN(CVX): continuity_convexa? -> common extremes from two or more CVX named convex_curve

It is important to note that considering the out-shape of the obstacle helps to define better strategies for avoiding it.
Some other functions relate to strategies. Since are also responsible for creating optimized displacement from one point to another, this is very important to perform efficient movements while avoiding obstacles:

FUN(mov): bring_near? -> verify if it's near to a distance and performs a movement
FUN(mov1, mov2): adjacent?-> sequence of movement SEQ
FUN(SEQ): movement_simplification -> summary the duplicated SEQ in a SEQ-Max

STEP 2- Colony building

The solution set can be characterized by:
$X=(g\ (INPUT_k),...)$ (1)

In equation 1, $\mathbf{k} \in \mathbf{K}$, and **K** is a natural number, **X** represents the vector of new and potentially more powerful codelets obtained by the application of function **g(x)** that implements certain heuristic based on the current set of FUNCTIONS that conforms PROCESS vector.
The parameter **k** is the index of definite function with parameters $\mathbf{p_1}\ldots\mathbf{p_m}$, and $g(x) <= 0$.
Although g(x) could implement any heuristic, in order to keep simulating bacteria, it represents the most probable behavior of the virus under current temperature conditions. In this context it can be the selectability of certain function (Binomial probability of firing a FUN_j) given the degree of conscious activity (**T** parameter):

$G(x)= Bin(FUN_j) \times T$ (2)

Equation 2 represents the chance of certain set $\{FUN_j\}$ being elected at **T** temperature given the current INPUT (that is a subset of CONCEPTs, PERCEPTs, and CODELETs)

STEP 3- Infection type definition and infection
In biological bacteria the infection produced can be classified as follows:
a) Selective
b) Massive

To extrapolate this concept to consciousness in robots, let us define **Y** as the vector **X** size (the number of elements that can be selected for being part of the solution, in other words, the total number of CONCEPTs, PERCEPTs, and CODELETs that constitute the conscious. Let us also define **A** as the infections (the number of elements that are affected by bacteria processing, in biology the number of infected cells).
With the following conditions:
$A \leq Y$ (3)
$x \in K$
Infection type helps to improve flexibility and provide diversity in OUTPUT when it is massive. Conversely, a selective infection helps stabilize results. A combination of both can implemented using a simple random number as follows:

IF Rnd(1) > 0.5 THEN
select option (a)
else select option(b)

3-A) Selective Infection
A single cell is chosen from the entire colony according to selection criteria. In the current proposal a uniform distribution U(x) has been considered.

3-B) Massive Infection
It affects a subset of cells. Let's say **Y-A** cells are infected. The procedure takes **Y-A** elements from INPUT by chance. Infected elements are replaced: $PROCESS_u$ ($\mathbf{x_h}$)

In any case, the result of the infection is an OUTPUT vector that must be tested by conscious system using its Work Plane to evaluate it goodness.

Then the process continues setting $OUTPUT_i$ =($CODELET_1$, $CODELET_2$, …), the set of new codelets more abstract. Algorithm for this is:

$OUTPUT_i$=[
getHandler($Codelet_1$), *// CODELETs*
getHandler($Codelet_2$),
...
getHandler($Codelet_M$),
]

Here, $CODELET_i$ is a CODELET that changes its parameters to get a more abstract function. It is a kind of factory for CODELETs:

```
getNewCodelet(newName){
    NumberOfParameters= getNumberOfParameters(); // returns a total of parameters
    parameters=newArray[NumberOfParameters];
    new Class.newInstance(newName, parameters);
}
```

Step 4- Virus evolution type definition
At this time, the biological process continues with cell antagonistic response. Virus evolves in one of the following ways: lysogenic or lytic. To emulate this, "affected" cells (elements) are evaluated as follows:

*if Bernoulli (**A**) > 0 THEN*

The response is lysogenic
ELSE
The response is lytic

A) Lysogenic replication

When a virus, penetrate in the host cells and keep in them, with no production of new viral particles

B) Lytic replication

When virus disseminates, it penetrates the host ell and use its replication mechanism to produce new viral particles. It starts only if a specific number of nucleus-capsids have been replicated. The number of that the threshold (limit number of nucleus-capsids replication **LNR**) in a cell **x** is given by:

$$LNR_{cell-x} = LNR^0 \cdot \left(\frac{f(x) - f\left(\tilde{x}\right)}{f\left(\tilde{x}\right)} \right), \quad (4)$$

With:

f(x) = scoring function of the Work Plane (**WP**). Its algorithm is composed by four scoring components that follows

1.-Prioritize forward *straight* line moves when no obstacles:

getScoreNoObstaclesStraightForward(int count, double coeff)

where count is the number of obstacles in a straight line, and coeff is defined by

coeff= wpStrategy.getEvaluationCoefficients().get(0);

2.-Prioritize minimal *distance* until objective:

getScoreWithObstacles(double distance, double coeff)

coeff= wpStrategy.getEvaluationCoefficients().get(1);

3.-Prioritize *rotations* with forward movements:

getScoreTurnForward(double angle, double coeff)

where angle is the minimum angle with no obstacles, and coeff is

coeff= wpStrategy.getEvaluationCoefficients().get(2);

4.-Prioritize rotations with *backward* movements.

getScoreTurnBackward(double angle, double coeff)

angle is the minimum angle with no obstacles.

coeff= wpStrategy.getEvaluationCoefficients().get(3);

To perform the evaluation, every WP is checked to evaluate if exists a strategy associated to it: if it is the case, count in how many **WPs** there is at least one of the codelets listed by **x**. If there is no related strategy then select any **WP** by chance (since at least one of the 0.. count(**WP**) must be selected)

The cell that produces the better result to the problem is x (in terms of f(x)). Here, **x** is the infected cell under consideration and $\mathbf{LNR^0}$ is initial value of **LNR**.

Infection evolves by cycling all the steps making the systems element change progressively. In each iteration there is a number (**NR**) of bacteria replica. The replication number is calculated adding a binomial variable (**z**) to the total value **NR**.

The probability of replicating exactly **z** nucleus-capsids, **P(z)**, the average **E(z)** and the variance **var(z)** are given by equations 5 to 7.

$$P(Z = z) = \binom{LNR}{z} p_r^z (1 - p_r)^{LNR - z}, \quad (5)$$

$$E(Z) = p_r LNR, \quad (6)$$

$$Var(Z) = p_r (1 - p_r) LNR, \quad (7)$$

In this context p_r is the probability after of a replication and **z** is a random binomial variable, that is:
$Z = Bin (LNR, p_r)$ (8)

When the edge of a bacteria is broken and releases virus, each of these released ones have a probability (**p_i**) of infecting others in its neighboring.

The probability of infecting exactly **y** nucleus-capsids (P(Y=y)), the average (E(Y)), and the variance (Var(Y)) are given by equations 9 to 11.

$$P(Y = y) = \binom{|V(x)|}{y} p_i^y (1 - p_i)^{|V(x)| - y}, \quad (9)$$

$$E(Y) = p_i |V(x)|, \quad (10)$$

$$Var(Y) = p_i (1 - p_i) |V(x)|, \quad (11)$$

with **|V(x)|** being the feasible solution for current "environment" or status. In conscious systems represented by a set of concepts produced by the sequence of codelets fired previously. **Y** is a random binomial variable which represents the infected cells by the virus in this environment, that is, **Y** = Bin ($|V(x)|$, p_i).
Each of these infected cells in the clinical picture have a probability of developing antibodies against the infection based on the Bernoulli distribution probability (p_{an} : $A(x) = (p_{an})$).

Thus the total of infected cells population which produces antibodies are characterized by a binomial distribution given by equation 12:
p_{an} : A(population) = Bin (n . p_{an}) (12)

The probability of finding exactly **a** immune cells (P(A= a)), the average (E(A)), and the variance (Var (A)) are given by equations 13 to 15.

$$P(A = a) = \binom{|V(x)|}{a} p_{an}^a (1 - p_{an})^{|V(x)| - a}, \quad (13)$$

$$E(A) = p_{an} |V(x)|, \quad (14)$$

$$Var(A) = p_{an} (1 - p_{an}) |V(x)|, \quad (15)$$

where **a** indicates the immune cell and **x** the infected cell. In the case of lysogenic type replication, it is possible to calculate the maximum interaccion limit (LIT) of a **x** cell with equation 16.

$$LIT_{cell-x} = LIT^{0} \cdot \left(\frac{f(x) - f\left(\tilde{x}\right)}{f\left(\tilde{x}\right)} \right), \quad (16)$$

with **LIT$_0$** being the initial value of **LIT**.

Step 5-Closing Step

The algorithm finishes according to one of the following criteria
A – Collapse or death of the organism (in this case, all the options were tested and there is no more exploation to be done)
B – Virus isolation (in this case the bacteria mechanism is cancelled)

2.3. Hardware considerations

The current proposal for MARCOS's hardware is the following:

1 x Ultrasonic Module - HC-SR04
Ultrasonic ranging module HC - SR04 provides 2cm - 400cm non-contact
Measurement function, the ranging accuracy can reach up to 3mm. The modules
Include ultrasonic transmitters, receiver and control circuit.

• Working Voltage: DC 5 V
• Working Current: 15mA
• Working Frequency: 40Hz
• Max Range: 4m
• Min Range: 2cm
• Measuring Angle: 15 degree
• Trigger Input Signal: 10uS TTL pulse
• Echo Output Signal: Input TTL lever signal and the range in proportion
• Dimension: 45*20*15mm
• Test distance: (high level time×velocity of sound (340M/S) / 2



Fig. 04 Ultrasonic device.

1 x Gyroscope Module MPU6050
• Digital-output X-, Y-, and Z-Axis angular rate sensors (gyroscopes) with a user-programmable full-scale range of ±250, ±500, ±1000, and ±2000°/sec
• External sync signal connected to the FSYNC pin supports image, video and GPS synchronization
• Integrated 16-bit ADCs enable simultaneous sampling of gyros
• Enhanced bias and sensitivity temperature stability reduces the need for user calibration

• Improved low-frequency noise performance
• Digitally-programmable low-pass filter
• Gyroscope operating current: 3.6mA
• Standby current: 5μA
• Factory calibrated sensitivity scale factor
• User self-test

1 x Accelerometer Module s MPU6050
• Digital-output triple-axis accelerometer with a programmable full scale range of ±2g, ±4g, ±8g and ±16g
• Integrated 16-bit ADCs enable simultaneous sampling of accelerometers while requiring no external multiplexer
• Accelerometer normal operating current: 500μA
• Low power accelerometer mode current: 10μA at 1.25Hz, 20μA at 5Hz, 60μA at 20Hz, 110μA at 40Hz
• Orientation detection and signaling
• Tap detection
• User-programmable interrupts
• High-G interrupt
• User self-test
• 2 Encoders



Fig. 05 Accelerometer device.

1 x Servo Module SG90
• Weight: 9 g
• Dimension: 22.2 x 11.8 x 31 mm approx.
• Stall torque: 1.8 kgf·cm
• Operating speed: 0.1 s/60 degree
• Operating voltage: 4.8 V (~5V)
• Dead band width: 10 μs
• Temperature range: 0 ºC – 55 ºC
• Approximately 180 degrees of rotation, 90 in each direction



Fig. 06 Servo device

1 x Arduino UNO Module
• Microcontroller: ATmega328
• Operating Voltage: 5V
• Input Voltage (recommended): 7-12V
• Input Voltage (limits): 6-20V
• Digital I/O Pins: 14 (of which 6 provide PWM output)
• Analog Input Pins: 6
• DC Current per I/O Pin: 40 mA
• DC Current for 3.3V Pin: 50 mA
• Flash Memory: 32 KB (ATmega328) of which 0.5 KB used by bootloader
• SRAM: 2 KB (ATmega328)
• EEPROM: 1 KB (ATmega328)
• Clock Speed: 16 MHz



Fig. 07 Arduino Uno device.

1 x Extension V5 Module
• I2C interface
• Bluetooth module communication interface
• SD card module communication interface
• APC220 wireless RF modules communication interface

• RB URF V1.1 ultrasonic sensors interface
• 12864 LCD serial and parallel interface
• 32 servo controller interfaces
• Dimensions: 57x57x18.5mm
• Weight: 23g



Fig. 08 Extension Shield v5 device.

1 x Bridge H L298N Module
• Operating Voltage: 4V to 35V
• Motor controller L298N, Drives 2 DC motors or 1 stepper motor
• Max current 2A per channel or 4A max
• Free running stop and brake function
• Chip: ST L298N
• Logic power supply:5v
• Max power:25w
• Weight: 35g
• Size:55mm x 60mm x 30mm
• Storage temperature: -25 to +135



Fig. 09 Bridge H L298N device

2 x Batteries
• Generic 3800mh 3.7V li-ion



Fig. 10 Battery

Others
• Generic 4 Motors
• Generic 4 Wheels

2.4. Marcos.COFRAM Handshake

The handshake between MARCOS.COFRAM and the arduino-based robot is required to negotiate the way both sides will perform during the current performance.
The fields' format is defined through a GUI interface that enables an administrator to easily manage the sequence and values expected from both sides.

Figure 11 shows a screenshot of the Handshake administrator.

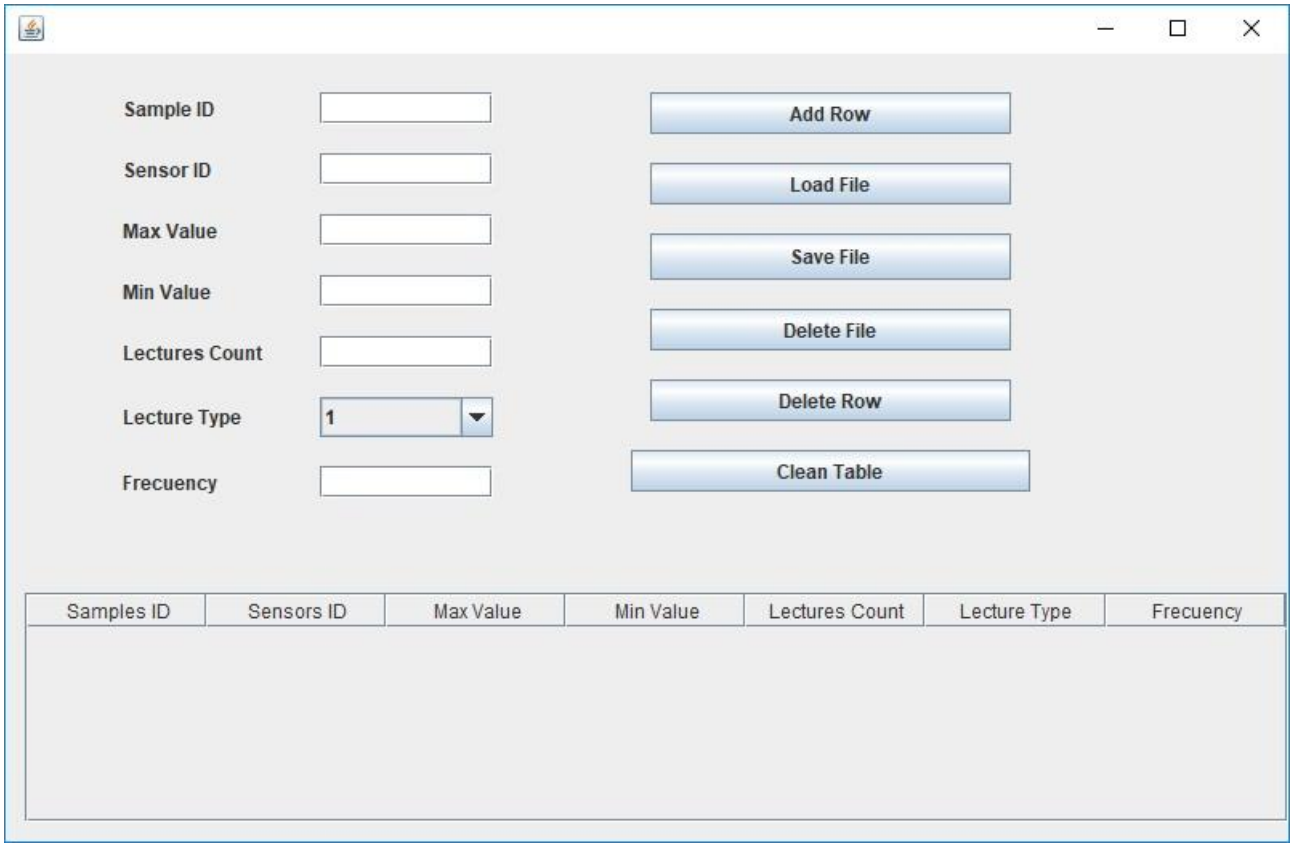


Fig. 11 Handshake administrator interface.

The file built by the application has comma separated values records (CSV) with the following fields:

Iinternal-ID
PIN-controler: with format AAAAA MMMM NNNN

with AAAA being the model of the arduino device
MMM factory or firm
NNNN sequenced number
Number of data in the vector
Type of data

Steps to establish the communication between MARCOS y COFRAN:

1 - Send at the startup – Sensing or searching the devices in the robot.
Figures 12 and 13 shows Handshake interface.

Init handshake message	DQ_MSJ_INIT
for each sensor	
Init id device	DQ_MSJ_ID_DEVICE
id device	(uint)
Init sensor sign	DQ_MSJ_ID_SENSOR
id sensor	(uint)
lectures amount	(uint)
min val	(double)
max val	(double)
check if readed	DQ_MSJ_ACK

Fig. 12 Handshake interface

.
2 - With the devices metadata, the environment information is gathered.

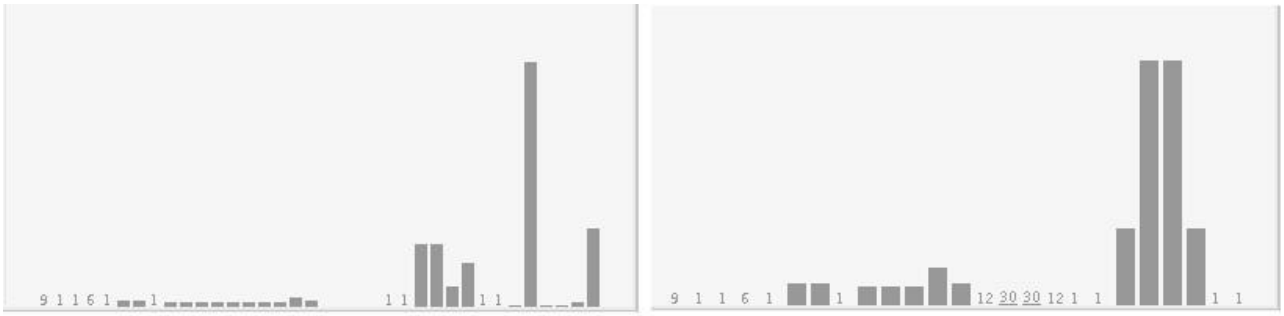


Fig. 13 Handshake interface

3. Testing

Codelets that mimic bacteria behavior were added to the original code-rack prior to perform any tests. This will be here referred to as “Bacteria behavior” whilst the traditional consciousness system that evolves progressively from bottom knowledge will be referred to as “Consciousness behavior”. Both cases use the same consciousness-based system (the COFRAM framework), but in the first case, a set of codelets mimic bacteria infections in order to produce new codelets with additional abilities. It is important to note that with plain consciousness behaving there are no means that this “extra” behavior can be automatically acquired. This paper aims to introduce, exhibit and compare both configurations.
It is important to note that bacteria produce codelets with extra flexibility since its paradigm is quite different: adaptation through evolution and interaction with host. On the other hand, codelet creation and mutation (when exists) in traditional consciousness is the result of mere learning by reinforce (see LIDA [30] for more details). That is a good approach when precision and

consciousness have not severe time restrictions; but it is not always the case, especially with autonomous mobile robots.

The remaining of this section presents three tests performed with bacteria and consciousness behavior. Every time, a summary of main statistical results and explanatory figures are being introduced, as well as a short interpretation.

3.1 Dataset

Three tests with different world-map, produced a total of six datasets (two samples each test, one for bacteria and the other for consciousness). Both datasets have the following variables:

- Timestamp: a sequence number to denote every step in the processing
- clockTic: the ID of the tic when the clock advances. This is a virtual clock that serves to make evident which steps are to be performed during the same tic. Many of them will be implemented in parallel threads.
- Component: signature of the function in Java without parameters
- Activity: task performed by the component
- Detail: extra information for the activity

All the testing was performed to evaluate the resolution ability of the codelet set, so the system kept running until it reaches a strategy and declares it as the winner. There is no feedback or further reinforcement, since it would be more difficult to evaluate which part of the strategy has evolved by the codelets quality and diversity, and which has evolved due to feedback (try and error).
This approach is intended to overcome the codelet diversity in conscious systems for autonomous mobile robots, avoiding long evolution of the conscious until getting into a best ability to react to the environment. This is important since such robots need to react in real time to stimuli, while being flexible enough to avoid obstacles with smart strategies.

3.2 Test 1

The first test has one linear obstacle as can be seen in Fig 14. This constitutes the simplest test of the entire test set.

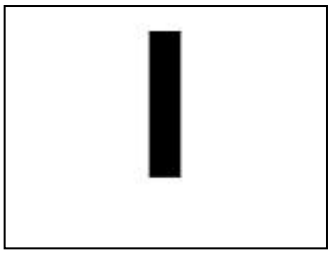


Fig. 14. World-map for test 1

Figure 15 shows the performance with and without bacteria-like behavior.

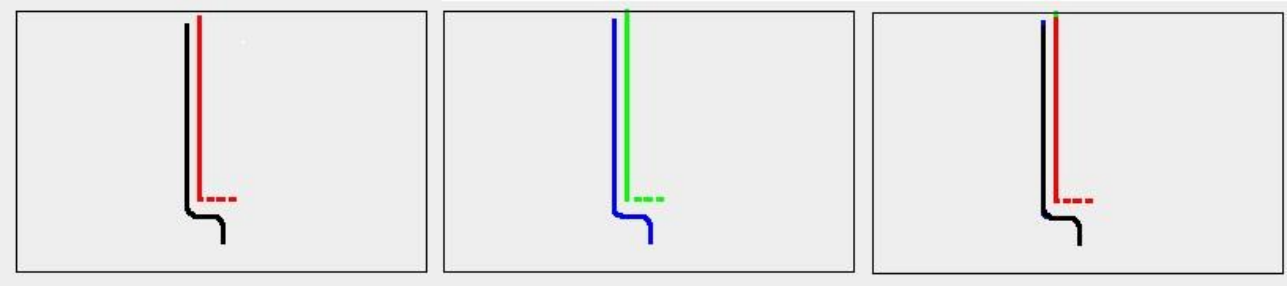


Fig. 15. Test with (left) and without (middle) bacteria and both (right)

The light line shows the current obstacle position (as it was detected during robot displacement). The dark line is the real trace performed.
On the right, there is a compound image with both tests. As can be seen, they are almost the same except for a couple of dots. This is because the codelets are working in both cases using the same world-map and the resolution is quite simple.
22752 records were collected for the test with bacteria behavior, and 11463 records for traditional consciousness behavior. The difference is due to the elevated variety and productivity of bacteria compared to traditional consciousness system.

a) Global behavior performance

The first step is to compare system's performance with and without bacteria. The descriptive stats are detailed in Table 01.

Table 01. Descriptive stats with and without bacteria

variable	Bacteria	Consciousness
#activities performed	27	22
#clock tics	14	12
Most frequent activity	Created	Created
Less frequent	Winner, run	Winner, run
#components activated	36	26
Most frequent component	Percept	Percept

Table t1 is a summary of the detailed descriptive statistics obtained with Infostat (c). As can be seen from the table, there is more "activity codelets" and "components activated" in Bacteria than in pure Consciousness. This is due to the bigger effort in building new strategies required by Bacteria. But from the number of "clock ticks", it is clear that, the system does not implement all the alternates. The reason is simple: the system knows in advance which strategy is better and does not need to test them. The advantage is significant (14 against 251 for consciousness) and derives from the fact that bacteria criteria adds extra powerful abstraction levels to the codelets in the coderack from the beginning, showing a good improvement in the performance.

In a more visual way, figures 16 and 17 show the histogram comparison for bacteria (black) and consciousness (grey) frequencies of component activation during the activity performed. Detailed frequencies are in Appendix A.

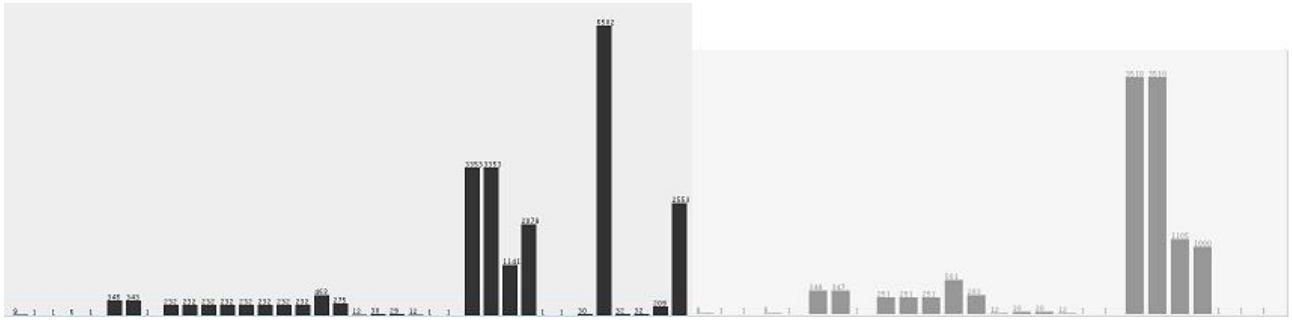


Fig. 16. Components activation for (black) and consciousness (grey)

Components most activated in bacteria behavior are:
-validation of continuity: 6582 (28.9%)
-Percept and PerceptLauncher: 3353 (14.7%)
-ContinuityEdges: 2553 (11.2%)
-Coderack: 2079 (9.1%)
-Sensing: 1141(5%)
The rest of the components remain under the threshold of 469 activations (2%).

Components activated in consciousness behavior are:
-Percept: 3510 (30.6%)
-Percept Launcher: 3510 (30.6%)
-Sensing: 1105 (9.6%)
-Coderack: 1000 (8.7%)
The rest of the components remain under the threshold of 349 (3%)

From the activity percentages, it can be said that the distribution of activities is fewer in bacteria than in consciousness, but in the first case dominates the evaluation of shapes (validation of continuity) while in the second there are more sensing and percept manipulation. That is due to the need to reevaluate the strategy many times because it is not suffice to solve the trajectory determination.
An interesting matter is that Coderack (set of codelets available) is being accessed more frequently in bacteria. That is because there is a need to evaluate alternate codelets not present in the case of pure consciousness.

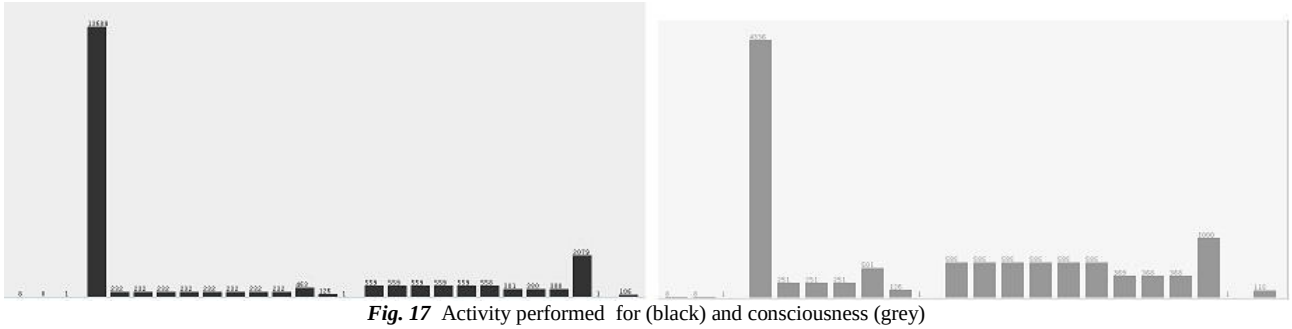


Fig. 17 Activity performed for (black) and consciousness (grey)

Here the efficiency of the bacteria system is made evident since in that case, there is mainly one activity which is “create” with 59.8% of the records in the log (13609). That is because many different codelets intensively being created and destroyed during exploration of strategy options (destruction of the codelets was not logged). In the case of consciousness. A similar scenario takes

place, since the creation activity dominates, but just with 37.8% of the records (4336 instances). This means that most of the activity is spread in many concomitant activities but not finding new strategies. Therefore, exploration of approaches for robots displacement becomes broader and thus more flexible.

b) Strategy for resolution

This section compares the approach for finding a strategy with and without bacteria. In order to do this, the dataset was reduced, taking out all the records whose activity are other than DISTANCE, DISTANCE L, DISTANCE R, and CodeletExecution. This was done to evaluate only activities related to strategy resolutions and no world map construction, codelet creation and other additional tasks. This results in 3220 records for bacteria test, and 2105 records for consciousness.

The dispersion diagram of timestamp (x) and clockTic (y), partitioned by activity (Codelet execution) is in figure 18 for bacteria (left) and for consciousness (right). Figure 19 shows the superposition of both curves where it is clear that the number of strategies built are greater for bacteria than for consciousness.

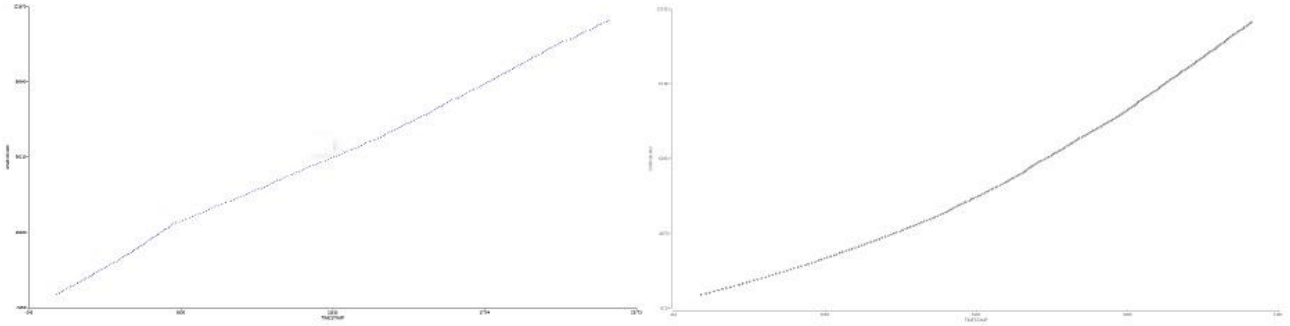


Fig. 18. Dispersion diagram: Timestamp VS clockTick for bacteria (left) and consciousness (right)

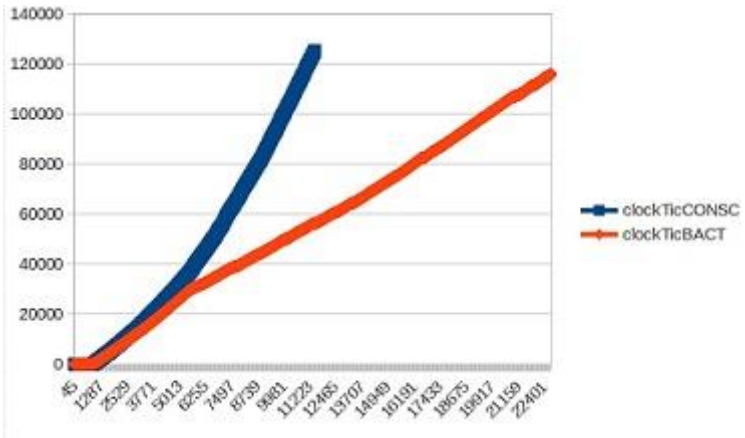


Fig. 19. Comparison of clockTic evolution for bacteria and consciousness

Fig. 18 shows a different behavior in the activity for both approaches: more intensive for bacteria than for consciousness. In Fig. F4.6 clockTicCONSC denotes the evolution of tics when consciousness is being running, while clockTicBACT stands for bacteria. For consciousness, the tics are quicker and the system ends earlier because, as mentioned before, the number of codelets in bacteria are quite higher and the decision takes more time. In the first case, the number of codelets are limited and represent just global approaches (don't have time to learn and enhance strategy).

Figures 20 and 21 show a comparison between bacteria and consciousness frequencies respectively, related to different strategies used in both cases.

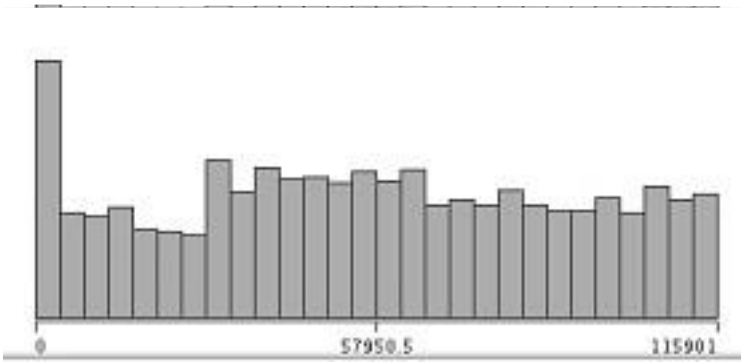


Fig. 20. Comparison number of clockTics for bacteria

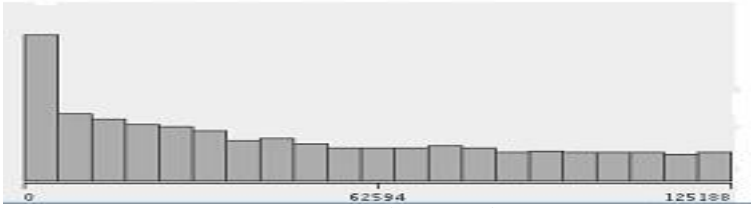


Fig. 21. Comparison number of clockTics for consciousness

In the case of bacteria, there are 232 different strategies created, while 251 strategies are created for consciousness. The first frequency in both cases for bacteria remains for a dummy strategy that is being used while the robot is performing its set-up and initial codelet preparation, before the bacteria starts its activity.
It is evident that the number of different strategies is higher for bacteria than for consciousness. So, it becomes clear that there is an extra flexibility in the solution exploration along with more alternates and possibilities to try. Therefore, the system can explore and use different approaches each time. For consciousness, after an initial intensive search, the system calms down. For bacteria, there is an initial activity level that increases and afterwards it remains rather similar. Concretely, after an initial sensing and codelet startup, there are many other codelets, fired afterwards, that remain looking for improvement of the initial proposal of the strategy.

3.c) Test 2

To evaluate a more complex world-map, a set of obstacles were inserted. Although they do not constitute a continuous obstacle, the distribution is such that it requires an extra evaluation for going through a global concave distribution of the objects (see fig. 22 dot line)

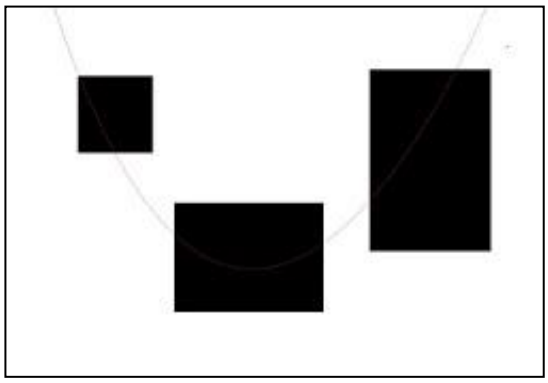


Fig. 22. Three objects for the new world-map, arranged in a concave distribution

As for the previous test, the robot run with and without bacteria. Figure 23 shows the traces (left image for bacteria, right trace just consciousness as it naturally evolves).

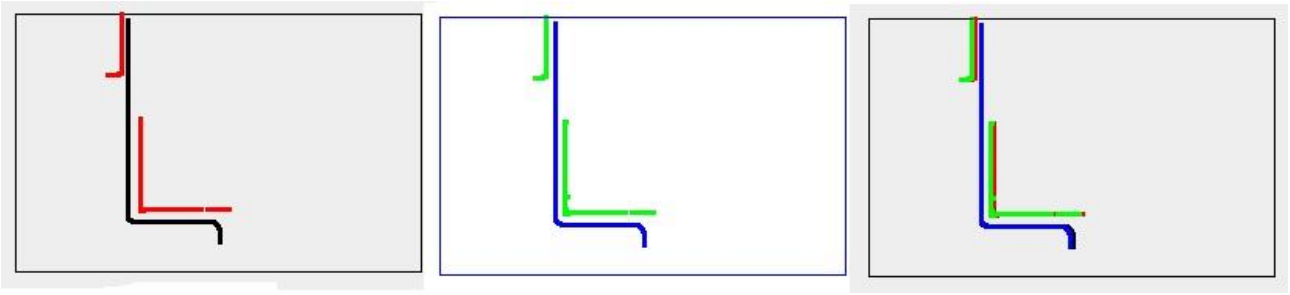


Fig. 23. Trace with bacteria (left) and just consciousness (middle) and both (right)

At this scenario, the traces have more differences. For consciousness, the distance is not equal to both obstacles. There is a trend to go closer to the left obstacle. The strategy for bacteria seems to be better since it keeps in the middle of both. From robotics perspective it's a clear improvement since the distance to obstacle is being kept safer without losing the original global direction and successfully avoiding all the obstacles.
Filtered dataset for activities under study lead to 3965 records for bacteria, and 2372 for consciousness.

a) Global behavior performance

The first step is to compare the system's performance with and without bacteria. The descriptive stats are detailed in Table 2.

Table 2. Descriptive stats with and without bacteria

variable	Bacteria	Consciousness
#activities performed	27	22
#clock tics	15	13
Most frequent activity	Created	Created
Less frequent	Winner, run	Winner, run
#components activated	36	25

Most frequent component	ci2s.marcos.concepts.links.cell.Adjacency	Percept

The key component for bacteria in this test is the Adjacency concept that reflects the search for the coherence between sensed cells. For consciousness, the elaboration was replaced by the permanent requirements to get the same information from the environment using the sensing mechanism (Percept component).
To sum up, bacteria develops a strategy without requiring further perception from the environment. It elaborates abstract concepts (adjacency, linearity, etc) for the set of information gathered from Percepts.

b) Strategy for resolution

This section compares the approach for finding the strategy. To do so, the dataset was reduced, taking out all the records out of those whose activity are other than DISTANCE, DISTANCE L, DISTANCE R, and CodeletExecution. This was done in order to evaluate only the activities related to strategy resolutions and no world map construction, codelet creation and other additional tasks. That leads to a total of 3965 records for bacteria test, and 2372 records for consciousness.

Figure 24 shows dispersion plots for both approaches and 25 compares the tics taken for both approaches during the test.

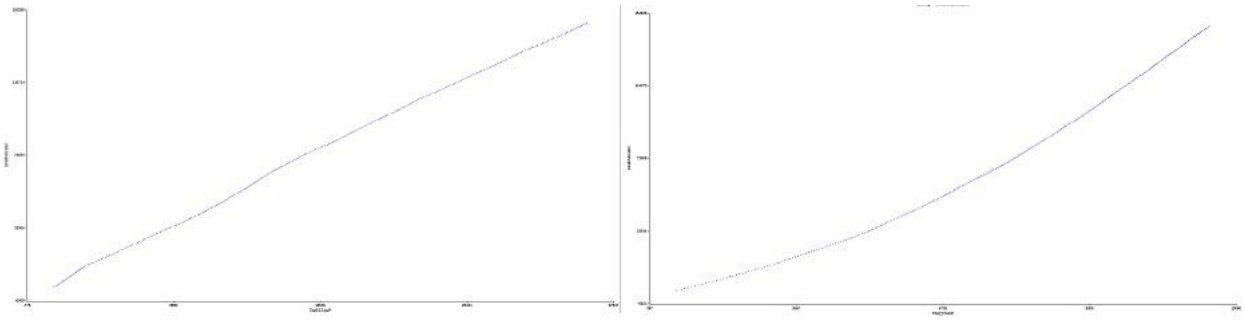


Fig. 24. Comparison of clockTic evolution for bacteria (left) and consciousness (right)

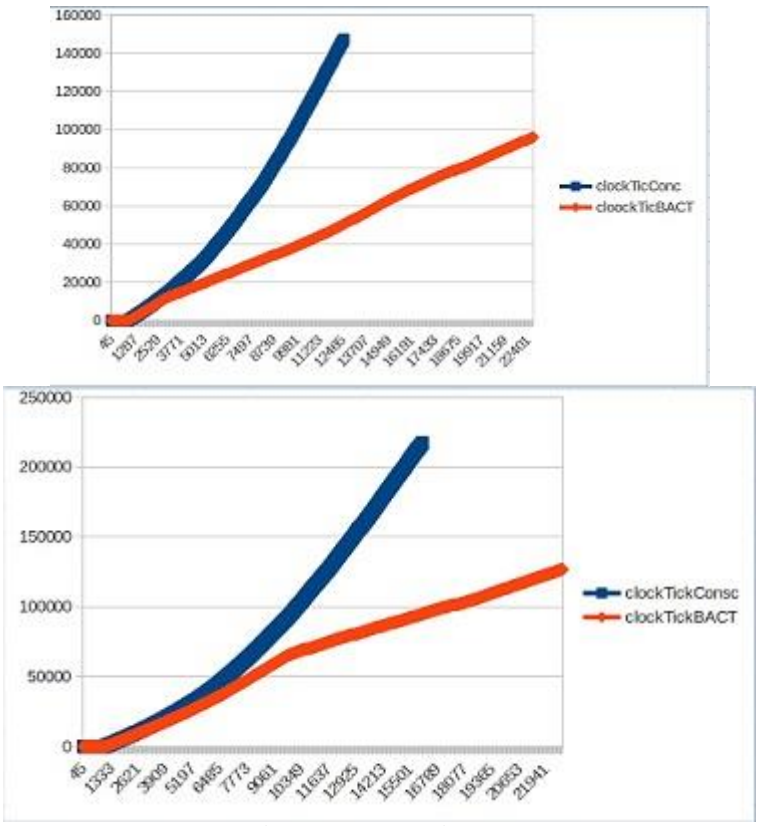


Fig. 25. Comparison of clockTic evolution for bacteria and consciousness

This test keeps the same trends from test 1, even though the world-map is more complex. It is interesting to compare figure 26: the activity is now different for consciousness but remains the same for bacteria.

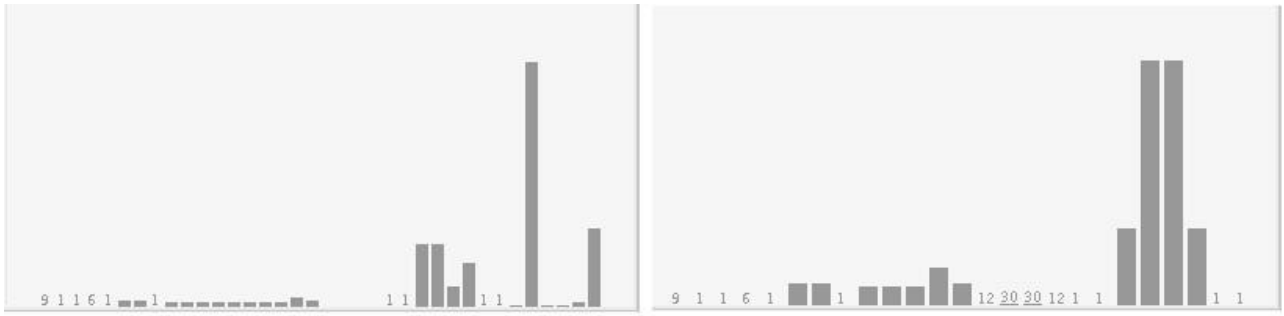


Fig. 26. Comparison of components activated for bacteria (left) and consciousness (right)

This clearly indicates that bacteria behaves similarly even when the task is more complex, denoting the efficiency and simplicity of this method.

3.c) Test 3

This final test aims to evaluate concave and convex shaped obstacles. Figure 27 shows the new world-map.

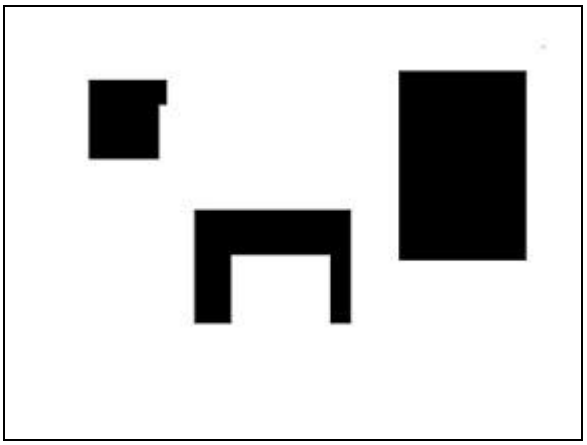


Fig. 27. World-map with concave and convex shaped obstacles

The test results in the traces indicated in figure 28 for bacteria (left), consciousness (middle) and the comparison (right)

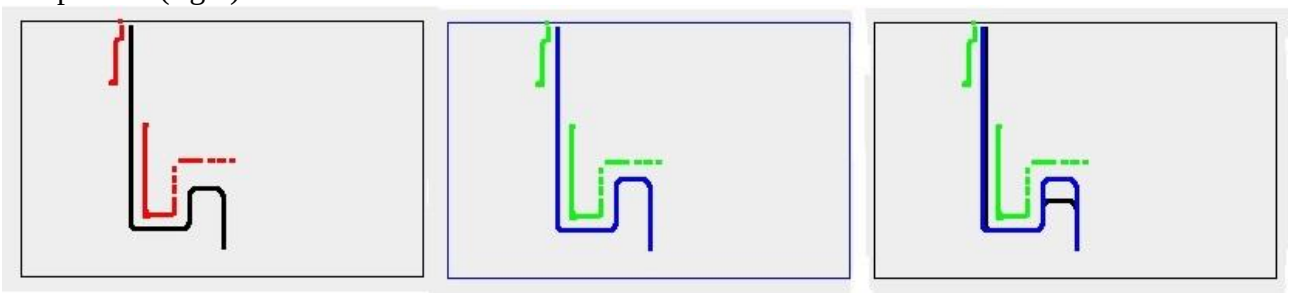


Fig. 28. Test for bacteria (left), consciousness (middle) and the comparison (right)

As can be noted, both strategies differ more than in the previous tests. The security margin increases, the trace's efficiency.

a) Global behavior performance

The first step is to compare system's performance with and without bacteria. The descriptive stats are detailed in Table t3.

Table 3. Descriptive stats with and without bacteria

variable	**Bacteria**	**Consciousness**
#activities performed	27	22
#clock tics	15	13
Most frequent activity	Created	Created
Less frequent	Winner, run	Winner, run
#components	37	25

activated		
Most frequent component	ci2s.marcos.concepts.links.cell.Adjacency	Percept

The table is very similar to table 2, for test 2. Confirming the trend observed for that test.

b) Strategy for resolution

This section compares the approach for finding the strategy. To do so, the dataset was reduced, taking out all the records out of those whose activity are other than DISTANCE, DISTANCE L, DISTANCE R, and CodeletExecution. This is done in order to evaluate only the activities related to strategy resolutions and no world map construction, codelet creation and other additional tasks. That leads to a total of 5550 records for bacteria test, and 3272 records for codelets.
Figure 29 shows dispersion plots for both approaches and 30 compares the tics taken for both approaches during the test.

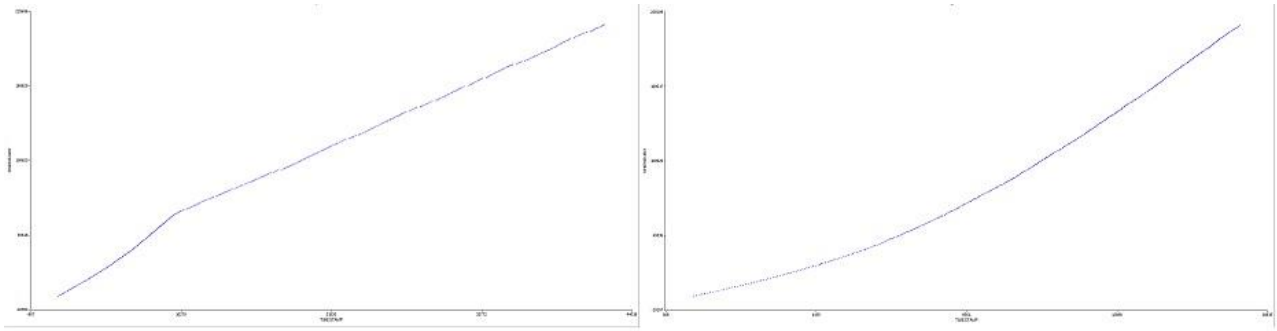


Fig. 29. Comparison of clockTic evolution for bacteria (left) and consciousness (right)

That indicates that bacteria behaves similarly even when the task is more complex.

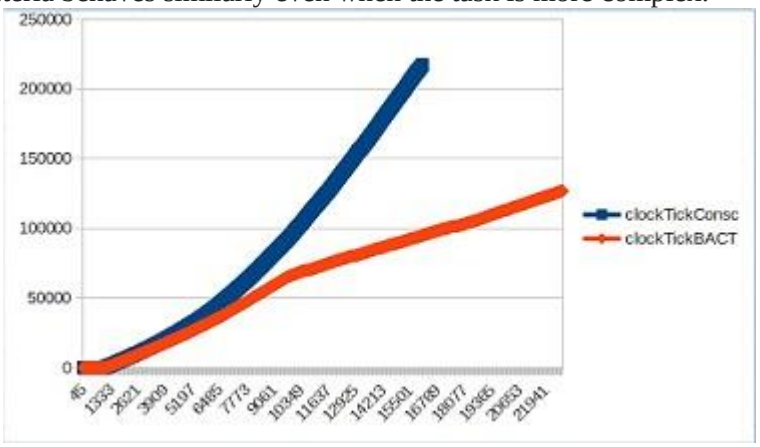


Fig. 30. Comparison of clockTic evolution for bacteria

It keeps the same trends as for tests 1 and 2, even though the world-map is more complex. Comparing figure 30 and figures 24, 25: the activity remains the same for both approaches.

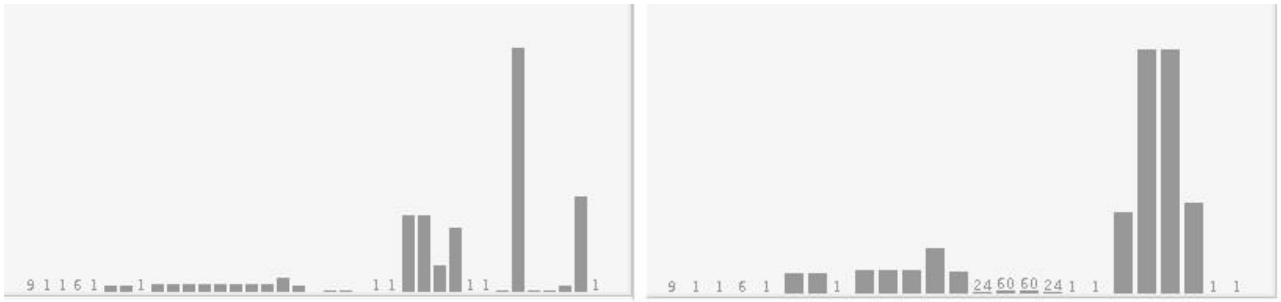


Fig. 31. Comparison of components activated for bacteria (left) and consciousness (right)

4. Conclusions and Future Work

This paper presented the basis of a new approach that intends to add flexibility, efficiency and creativity to the decisions taken by consciousness when applied to autonomous mobile robots. Three tests, in increasingly obstacle-avoidance complexity, were performed in order to compare traditional codelets versus the ones created with bacteria. The focus is on "activity codelets" and "components activated", given that bacteria paradigm provides adaptation through evolution and interaction.

These tests evaluate how bacteria produce more "activity codelets" and "component activated" than plain Consciousness system, adding extra powerful abstraction level to the codelets showing a true improvement in the performance. It is also evident from results that the diversity of strategies is higher for Bacteria (making it clear that there is an extra flexibility in the exploration of the solution). At the same time this gives the possibility of processing complex obstacle distributions more efficiently. As a conclusion, Bacteria-based consciousness robots behaves very well for tasks with different degrees of complexity, broadening the scope at which a robot can be put into.

As future work, there are many important things to work on. Among others to- Implement temporality to the codelets colony in order to achieve the auto-regulation according to the current strategy, research and implement new infections, tune the communication between codelets to imitate the behavior in a bacterial colony and test new strategies to overcome abrupt level difference in the path.

References.

[1] Raj Reddy. Robotics and Intelligent Systems in Support of Society. Carnegie Mellon University. Vol. 21, No. 3. May/June 2006

[2] Daniela López De Luise, Lucas Rancez, Nelson Biedma, Bernardo De Elía, Diego Lazcano Colodrero and Julián Mendez. Odometer Errors for a Robot controlled by Consciousness

[3] F. Penizzotto, E. Slawiñski and V. Mut. Laser Radar Based Autonomous Mobile Robot Guidance System for Olive Groves Navigation. IEEE LATIN AMERICA TRANSACTIONS, VOL. 13, NO. 5, MAY 2015

[4] Alessandro Di Fava, Massimo Satler and Paolo Tripicchio. Visual navigation of mobile robots for autonomous patrolling of indoor and outdoor areas. 2015 23rd Mediterranean Conference on Control and Automation (MED) June 16-19, 2015. Torremolinos, Spain

[5] Aneesh Chand and Shin'ichi Yuta. Navigation strategy and path planning for autonomous road crossing by outdoor mobile robots. The 15th International Conference on Advanced Robotics Tallinn University of Technology Tallinn, Estonia, June 20-23, 2011

[6] Yu-huan Chiu, Naoji Shiroma, Hiroki Igarashi, Noritaka Sato, Masahiko Inami and Fumitoshi Matsuno. FUMA: Environment Information Gathering Wheeled Rescue Robot with One-DOF Arm. Proceedings of the 2005 IEEE International Workshop on Safety, Security and Rescue Robotics Kobe, Japan, June 2005

[7] Ken Shimazaki and Tomoharu Nagao. Semantic Segmentation Considering Location and Co-occurrence in Scene. 2015 IEEE 8th International Workshop on Computational Intelligence and Applications November 6-7, 2015, Hiroshima, Japan
[8] Yan Zhuang, Wei Wang, Hang Chen and Keqiang Zheng . 3D Scene Reconstruction and Motion Planning for an Autonomous Mobile Robot in Complex Outdoor Scenes. Proceedings of the 2010 International Conference on Modelling, Identification and Control, Okayama, Japan, July 17-19,2010
[9] Shohei Iwata, Shota Sahashi and Tadahiro Hasegawa. Verification of visual attitude control for outdoor autonomous mobile robots. 2013, 10th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI).
[10] Hidefumi Kawamura, Shohei Iwata, Shota Sahashi, and Tadahiro Hasegawa. Visual attitude control using a virtual barycenter of a quadrangle that constructed from feature points for outdoor autonomous mobile robots.
[11] Ji Eun Jung, Kil Soo Lee, Hyung Gyu Park, Yung Ho Koh, Jong Il Bae, Man Hyung Lee. Vision-based Obstacle Avoidance System for Autonomous Mobile Robot in Outdoor Environment. International Conference on Control, Automation and Systems 2010 Oct. 27-30, 2010 in KINTEX, Gyeonggi-do, Korea
[12] Ufuk Zeki Sanal, Aydan M. Erkmen, Ismet Erkmen. Logical Sensing and Intelligent Perception Based on Rough Sets and Dynamic Decay Adjustment Learning. Proceedings of the 2002 IEEE International Conference on Robotics 8 Automation Washington, DC May 2002
[13] Ulf Witkowski, Mohamed El-Habbal, Stefan Herbrechtsmeier, Andry Tanoto, Jacques Penders, Lyuba!Alboul and Veysel Gazi. Ad-hoc network communication infrastructure for multirobot systems in disaster scenarios.
[14] Ahmet Denker and Mehmet Can Iseri. Design and Implementation of a semi-autonomos Mobie Search and Rescue Robot: SALVOR
[15] Abhijit Konde-Deshmukh, Jenil Doshi, Jeet Kothari, Manav Kawa and Dr. Vaishali Wadhe. Disaster Rescue Robot. International Journal of Electrical and Electronics Research ISSN 2348-6988 (online) Vol. 4, Issue 1, pp: (120-124), Month: January - March 2016, Available at: www.researchpublish.com
[16] A. Rama Krishna, G. Sowmya Bala, A. S. N. Chakravarthy, B. Bhanu Prakash Sarma and Gokul Sai Alla. Design of a Rescue Robot Assist at Fire Disaster. International Journal of Computer Applications (0975 – 888)
[17] Soumen Bag, Partha Bhowmick, and Gaurav Harit. Detection of Structural Concavities in Character Images—A Writer-Independent Approach.
[18] Ahmed Hussein, Pablo Marín-Plaza, David Martín, Arturo de la Escalera and José María Armingol. Autonomous Off-Road Navigation using Stereo-Vision and Laser-Rangefinder Fusion for Outdoor Obstacles Detection. 2016 IEEE Intelligent Vehicles Symposium (IV) Gothenburg, Sweden, June 19-22, 2016
[19] Shakeel Muhammad, Daniele Nardi, Kazunori Ohno and Satoshi Tadokoro. Environmental Sensing using Millimeter Wave Sensor for Extreme Conditions.
[20] Heesung Chae, Christiand, Sunglok Choi, Wonpil Yu, Jaeil Cho. Autonomous Navigation of Mobile Robot based on DGPS/INS Sensor Fusion by EKF in Semi-outdoor Structured Environment. The 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems October 18-22, 2010, Taipei, Taiwan
[21] Midhat Noor Kiyani and Muhammad Uzair Masood Khan. A Prototype of Search and Rescue Robot
[22] Lars Kuhnert, Benjamin Meier and Klaus-Dieter Kuhnert . Probabilistic approach to self-localization for autonomous mobile outdoor robotics based on hybrid map knowledge. Proceedings

of the 16th International IEEE Annual Conference on Intelligent Transportation Systems (ITSC 2013), The Hague, The Netherlands, October 6-9, 2013
[23] Miaolei Zhou, Wei Gao. Multi-sensor Data Acquisition for an autonomous mobile outdoor robot. 2011 Fourth International Symposium on Computational Intelligence and Design.
[24] Klaus-Dieter Kuhnert. Software Architecture of the Autonomous Mobile Outdoor Robot AMOR
[25] Klaus-Dieter Kuhnert. Concept and Implementation of a Software System on the Autonomous Mobile Outdoor Robot AMOR
[26] Daniela López De Luise, Lucas Rancez, Nelson Biedma and Bernardo De Elía. A conscious model for autonomous robotics: Statistical evaluation
[27] Ravikumar Kondadadi and Stan Franklin. Deliberative Decision Making in "Conscious" Software Agents
[28] E. Ben-Jacob, Learning from Bacteria about Natural Information Processing, Natural Genetic Engineering and Natural Genome Editing: Ann. N.Y. Acad. Sci.
[29] Mohiuddin Ahmed, M. R. Khan, Md. Masum Billah and Soheli Farhana. Walking Hexapod Robot in Disaster Recovery: Developing Algorithm for Terrain Negotiation and Navigation. , New Advanced Technologies, Aleksandar Lazinica (Ed.), ISBN: 978-953-307-067-4, InTech, Available from:
http://www.intechopen.com/books/new-advanced-technologies/walking-hexapod-robot-in-disaster recoverydeveloping-algorithm-for-terrain-negotiation-and-navigation
[30] Franklin, S., Ramamurthy, U., D'Mello, S., McCauley, L., Negatu, A., Silva R., & Datla, V. (2007). LIDA: A computational model of global workspace theory and developmental learning. In AAAI Fall Symposium on AI and Consciousness: Theoretical Foundations and Current Approaches. Arlington, VA: AAAI
[31] New Robot Strategy. Ministry of Economy, Trade and Industry of Japan
[32] Rafael Silveira Xavier, Nizam Omar and Leandro Nunes de Castro. Bacterial Colony: Information Processing and Computational Behavior
[33] B. Baars, A cognitive theory of consciousness.
[34] Bernard J. Baars. In the Theater of Consciousness The Workspace of the Mind. Oxford University Press 1997.
[35] R. Bostelman, T. Hong, R. Madhavan, T. Chang, and H. Scott. Performance Analysis of an Autonomous Mobile Robot Mapping System for Outdoor Environments.
[36] Zhang Xuhui, Dong Runlin and Liu Yongwei, VR-based Remote Control System for Rescue Detection Robot in Coal Mine. 2017, 14th International Conference on Ubiquitous Robots and Ambient Intelligence (URAI)
[37] Quan Qiu and Jianda Han . Function Sector Based Real-Time Autonomous Navigation for Outdoor Mobile Robots Equipped with Laser Scanners. 2009 IEEE International Conference on Control and Automation Christchurch, New Zealand
[38] Soumen Bag, Gaurav Harit. A Medial Axis Based Thinning Strategy for Character Images
[39] Helena Matutea, Carmelo P. Cubillasb and Pablo Garaizarc. Learning to infer the time of our actions and decisions from their Consequences.
[40] Daniel Dennet and Marcel Kinsbourne. Time and the observer: the where and when of consciousness in the brain.
[41] http://www.sc.ehu.es/sbweb/fisica_/cinematica/rectilineo/rectilineo/rectilineo.html
[42] I. Ulrich and I. Nourbakhsh, "Appearance-Based Obstacle Detection with Monocular Color Vision", Proceedings of the AAAI National Conference on Artificial Intelligence, Austin, TX, July/August 2000.
[43] Yan Zhuang, Shengpeng Yang, Xiaotao Li, Wei Wang. 3D-laser-based Visual Odometry for Autonomous Mobile Robot in Outdoor Environments.