The LEFT proposal

Andrei-Samuel Blîndu  
ETSETB

Universitat Politecnica de CatalunyaBarcelona, Spain  
andreiblindu2001@gmail.com

***Abstract*—Line following robot colony with obstacle avoidance, going through several stages of planning, comparing and development.**

***Keywords—Line Following, Obstacle Avoidance, Robot, PIC16F877, control algorithm, IR Emitter, Evorobot, mBot, Kephera, Machine Learning***

# Introduction

The LinE Following roboT project, referred throughout this document as the LEFT project, aims to design a robust robot colony, both from software and hardware perspectives, that follows a circuit consisting of a black line drawn on a non-black background. In addition, another task that is addressed is the implementation of a mechanism of avoiding obstacles which may be scattered around the circuit, the ultimate goal of the colony being to complete one lap in the shortest time possible.

The following sections of this proposal act in accordance with different approaches, using several indicated robots for each development stage, which are compared and discussed.

# Mobile Robot Architectures

## Mobile robots: a taxonomy

In order to completely grasp the concept of the mobile robot taxonomy, we have to break down each term before understanding them as a unit.

Robotic systems could be easily understood as the constant interaction of three subsystems: the sensors, the control unit and the actuators. The control unit is the programmable brain of the robot which dictates how the system should interact or respond to the environment. Based on the repercussions of these actions, the robot collects new information about the environment through the sensors, which change parameters in the program of the control unit, and so on this cycle continues. Mobility implies the freedom of movement, so all-together a mobile robot is a system able to easily travel through a given environment.

In Fig.1. a taxonomy of different mobile robots is given. Note that the LEFT project’s focus shifts towards terrestrial wheeled robots designed for educational purposes, although there are also aquatic and airborne mobile robots.

|  |
| --- |
|  |

1. Taxonomy of mobile robots according to environment and mechanism of interaction

## Robot learning architectures: from educational robots to swarms

Educational Robotics is a discipline designed to introduce students to Robotics and Programming interactively, so the philosophy that should guide manufacturing educational robots should prioritize cheap materials, in order to mass distribute them to schools and universities, making them accessible to anyone. In short, and educational robot should comprise of cheap sensors, a microcontroller, DC motors and wheels. Some good examples are the A-Cute Car and Sparki.

A-Cute Car has a DE0-Nano FPGA board for control; a driving board with 3.3 – 12 V DC motors, a AAx4 battery holder, a buzzer, two car lamps, IR receiver and ADC chip to digitalize the input power and line sensors; a 7-sensor board which allow, for example, following a line.

In behavior-based robots, the sensor observation is intercepted by a perceptual schema which extracts the relevant information of the environment for the behavior. This percept is later on used by the motor schema which then leads to an action.

The ability of adapting to an unknown and dynamic environment has sparked the interest for adaptive control robots, which through machine learning can change themselves for the better as a result of their own experience (at the end of one episode). This means that intelligent and adaptive robots can refine existing behaviors, combine them or even learn completely new ones.

# Line Following Robots

## Architecture basics

The basic subsystems provided in section II.A of a robot can act as a general model for line following robots as well. The control unit, which could be for example a PIC16877A microcontroller, dictates based on the sensory input the speed of the gear DC motors, which put the wheels in motion. In other words, as before, the control unit imposes what action the actuators take, implementing usually a robot with differential drive. The sensory unit then measures the brightness of the light reflected from the ground, in arbitrary values measured empirically, through ground reflective sensors mounted on the bottom of the robot. This information is again used by the microcontroller in a continuous cycle.

## Hardware

The principle of operation of the sensory block relies on the reflective sensors mounted on the top front part, represented by a simple IR emitter / detector array. These use infrared light reflection (LED emitter and a phototransistor for detection) in order to determine the color of the terrain above which the robot is located. The sensors are then further connected to microcontroller ports for the purpose of detecting strong terrain contrasts, or to distinguish different tones. If the robot is above a white surface, all of the infrared light is reflected so we read a low voltage at the output of the phototransistor. Different color tones absorb part of the light, so based on the intermediary voltages obtained at the output the robot could distinguish the color of the terrain beneath it, with black resulting in the highest voltage. The information from the whole array is used in order to obtain the empirical output value through fusion, so the robot can tell if he is completely above the line or just partially.

The control unit is divided into the central processing unit CPU, the Memory (registers and program memory) and the Input / Output peripherals (which provide physical access through I/O pins, this being the only contact of the microcontroller with the outside world). Particularly, PIC16F877 is an 8-bit PIC microcontroller that comes in three packages known as PDIP, PLCC, and QFN. The first one is 40-pin while other two come with a 44-pin interface.

It features 256 bytes of EEPROM data memory, 368 bytes of RAM, and program memory of 14K. This microcontroller version incorporates CPU, timers, 10-Bit ADC and other peripherals that are mainly used to develop a connection with external devices. The decent memory endurance around 1,000,000 for EEPROM and 100,000 for program memory, makes this device an ideal choice for many real-time applications, including line following.

## Control algorithms

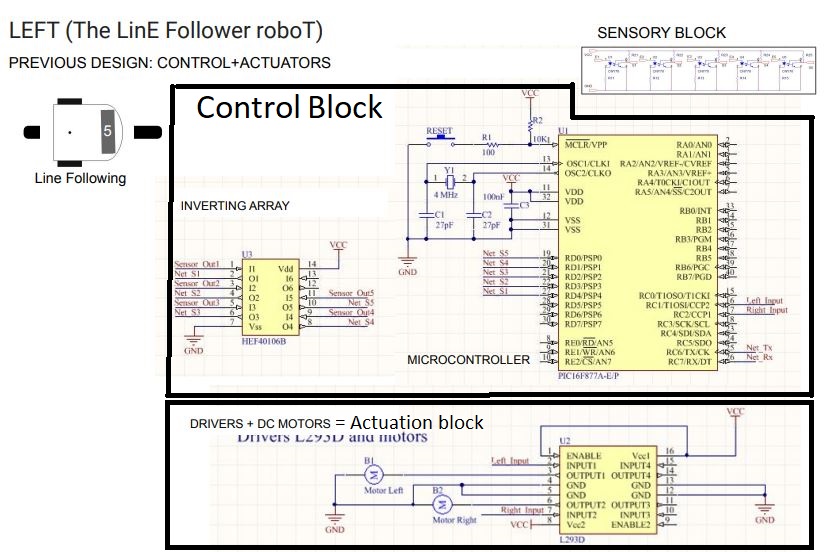
1. Characteristics of LF control algorithms

| **Features** | **Control algorithms for line following** | | | |
| --- | --- | --- | --- | --- |
| ***Pitching*** | ***Rise Time*** | ***Settling Time*** | ***Steady-State Error*** |
| Heuristic w/ two sensors | Increased | Increased | Decreased | Increased |
| Heuristic w/ five sensors | Moderate | Increased | Moderate | Increased |
| P w/one sensor | Increased | Moderate | Moderate | Moderate |
| PD | Moderate | Moderate | Decreased | Moderate |
| PID | Decreased | Decreased | Increased | Eliminated |

# Analysis Of Previous Design

## Functional block circuit decomposition and electronic operation

The infrared light emitted by the diodes in the sensory unit is reflected on the track and converted into voltage by the phototransistor. The sensors are connected to the microcontroller ports which based on the analog information received, dictates the differential drive. So, through the OUTPUT1 and OUTPUT2 ports the microcontroller sets the velocity of the motors which put the wheels in motion. If both wheels are set the same velocity, the robot moves straight forwards. If one wheel is driven faster than the other, the robot turn towards the wheel with the slower velocity. If the wheels are set with the exact same speed but different directions, the robot turns in place.



A problem which may occur in the circuit may be related to the sensory block. The resistor connected to the LED should provide a high current to it, but one just big enough to not destroy it. If the light of the emitter is too weak, the transistor may not be able to pick it up, but if the current is too big, the diode may as well just burn out.

Either case leads to a faulty sensory board, and although there are other sensors to provide backup, in the whole scheme of things it is undesired behavior.

Evidently, other limitations have to be taken into account regarding the maximum velocity of the dc motors, supply voltage and ground ports management and the sensory information manipulation by the control unit.

## Robot behavior

The microcontroller reads the input of each pin of the sensory board and through polling, it sets the value of each sensor accordingly (if the value is different than 0 set S to 1, else set it to 0).

If the middle left sensors are active( S2 and S3), set through PWM signals the left wheel speed to medium and right wheel speed to maximum in order to slowly turn left.

If the middle right sensors are active (S3 and S4), set left speed to maximum and right speed to medium in order to slowly turn right. If one of the side sensors is active on their own, make the robot make a hard turn by setting the corresponding wheel’s speed to low instead of medium. This way the robot will turn in order to keep the line as central as possible.

This approach is only heuristic so obviously it has flaws. The chosen velocities for turning are not the optimal ones for each individual sensory value case, so it is probable that the robot will move in a zigzag and waste time. Also, based on the starting position, or in the unexpected positions the robot can get by trying to follow the line, it can be stuck in an endless oscillatory case, where only the side sensors are activated. Moving too fast, it won’t be able to get the middle sensors activated so the task can become impossible. Also, by going forward at maximum velocity, hard turns can become tricky and the robot may go off track.

## Suggested improvements

To prevent components such as the diode emitters to be burnt out by the large current, we could protect them with NTC thermistors in order to avoid overheating or alternatively with varistors. On the other side of the problem, a current stabilizer approach could be used in order to assure that unexpected current variations, which ultimately could be destructive, will not occur. Ultimately, the resistors could become a big investment, and choosing ones with a low tolerance and thermic variance, backed by Spice simulations, could prove the best solution.

The code could be improved by capping the motor set velocity to their max value each time the program tries to output a value higher than the motor’s ability. An improvement on following accuracy would be going off a velocity smaller than maximum generally, but at the cost of losing valuable time. Also, as far as heuristics go, adding multiple specific cases could prove beneficial but the best approach would be to implement PID adaptive control.

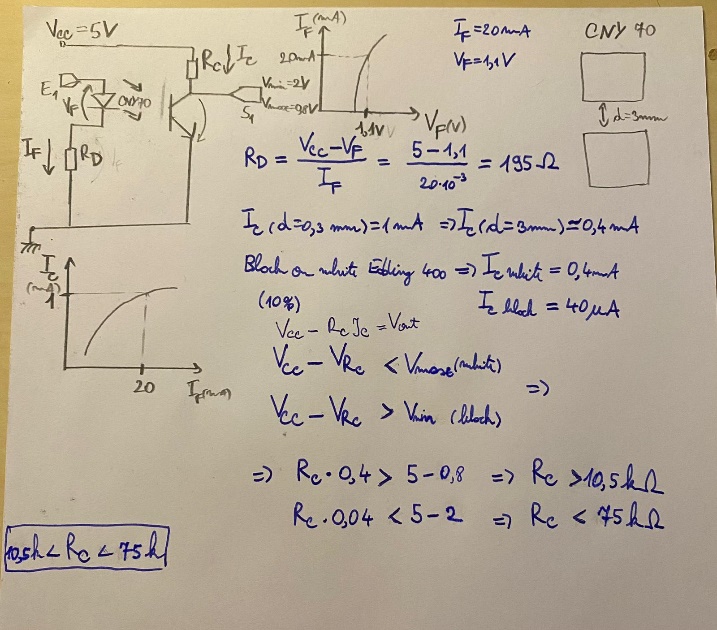
# Basic mBot Design

## Sensory array tuning

The CNY70-based sensor array resistors of mBot can be computed through the following inequalities:

|  |  |
| --- | --- |
| Vcc – VRc < Vmax | (1) |
| Vcc – VRc ≥ Vmin | (2) |

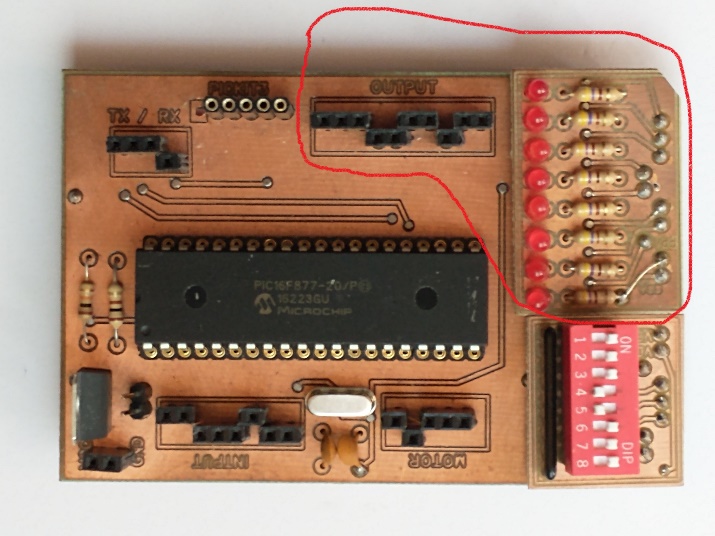
Note that (1) is the case when the phototransistor converts light reflected off a white surface, and that (2) is the case when it converts light reflected off a black surface.

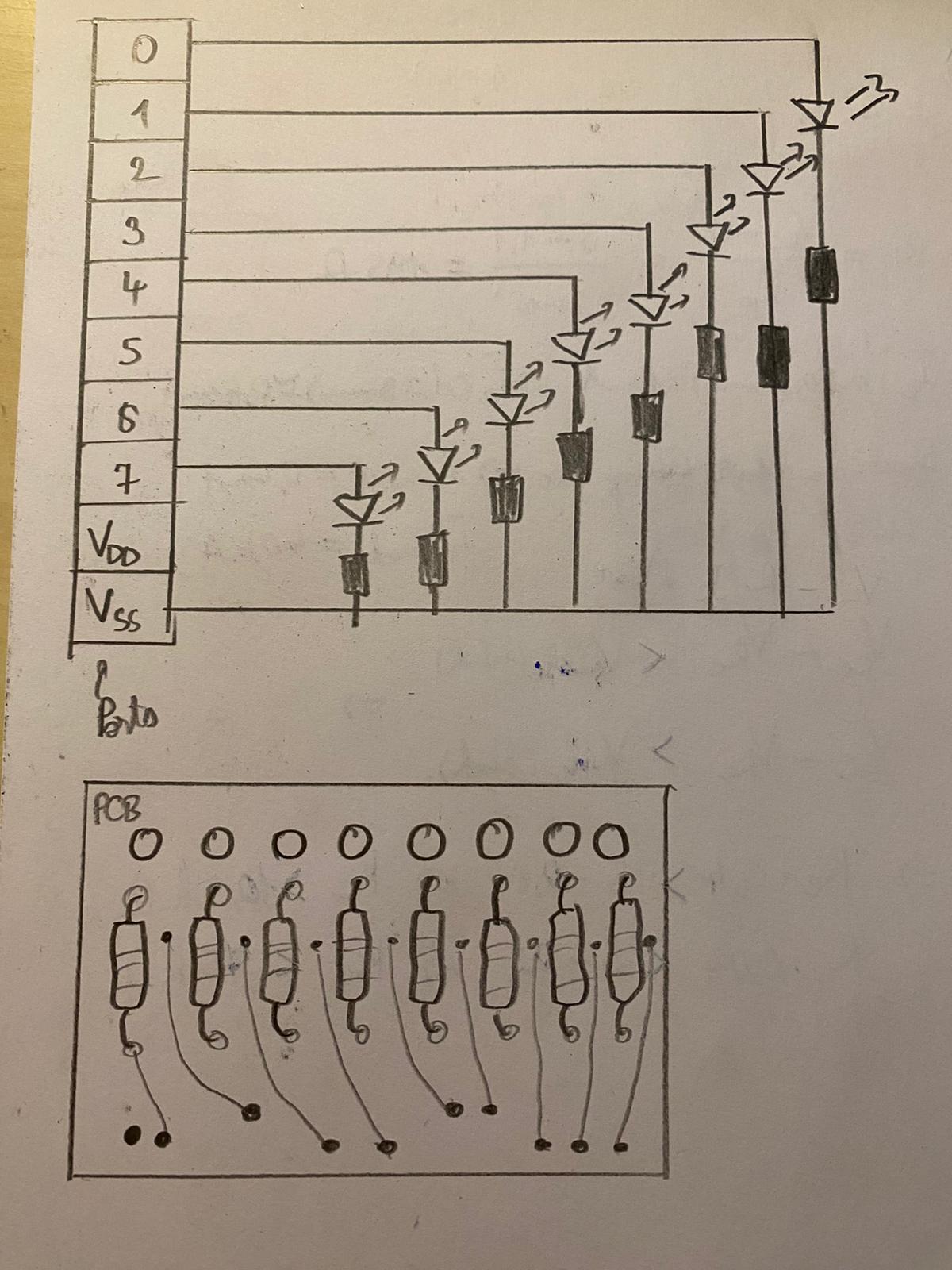


10.5 kohm < Rc < 75 kohm

## Board proposal

The board I have chosen to represent is the Infrared Emitter Output board. It contains 10 total ports, which makes it compatible with the mBot Schema, 8 of them corresponding to the subsystems containing a light emitting diode, which radiates IR light, and a resistor. The other ports remain for the power supply.



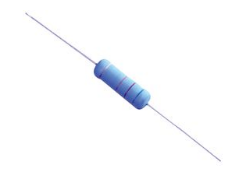


**Bill of materials**

Infrared Emitter, 940 nm, 10 °, T-1 3/4 (5mm), 15 mW/Sr x8

= 8 \* 0.162€ = 1.3€

<https://ro.farnell.com/multicomp/ofl-5102/infrared-emitter-940nm-t-1-3-4/dp/1716710>



Through Hole Resistor, 1 kohm, WMO-S, 5 W, ± 5%, Axial Leaded, 500 V x 8

= 8 \* 0.6€ = 4.8€

<https://ro.farnell.com/tt-electronics-welwyn/wmo5s-1kja05/res-1k-5-5w-axial-metal-oxide/dp/1306249?st=tht%20resistors%201k%205%25>

The prices shown on Farnell are in lei, where 5 lei is roughly 1 euro.

Excluding additional prices of the PCB, connecting terminals and extra parts, the total cost of making the IR Emitter Boars would rise up to **6.1**.

# Obstacle Avoidance with Khepera

## Experimental settling with Evorobot

Evorobot is a software for running evolutionary robotics experiments. It allows to replicate many of the experiments described in [1], (such as obstacle avoidance, T-maze and garbage collecting) and to run your own experiments.

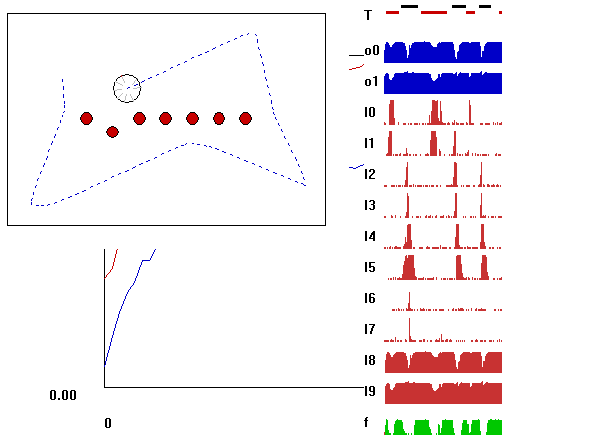
Evorobot allows running evolutionary experiments in simulation or on a real robot, but to run evolutionary experiments on the real robot or to test individuals evolved in simulation on the real robot you need a Khepera robot. Khepera was designed and built at the LPFS (Lausanne) in the nineties and is currently distributed and supported by K-Team S.A. It is a 5.5 cm mobile robot which can be trained to softly navigate in a particular environment.

## Results

[INSERT HERE: define a proper learning environment in Evorobot for the obstacle avoidance task, run a series of training sessions varying different parameters of the learning algorithm, and collect + display properly the obtained results.]

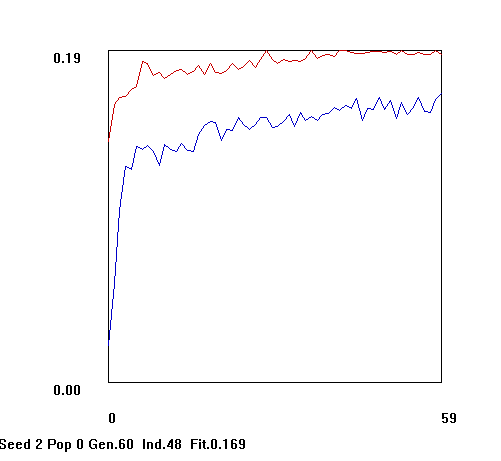
|  |
| --- |
|  |

1. Evolution of the cost functions over training time for the obstacle avoidance task.

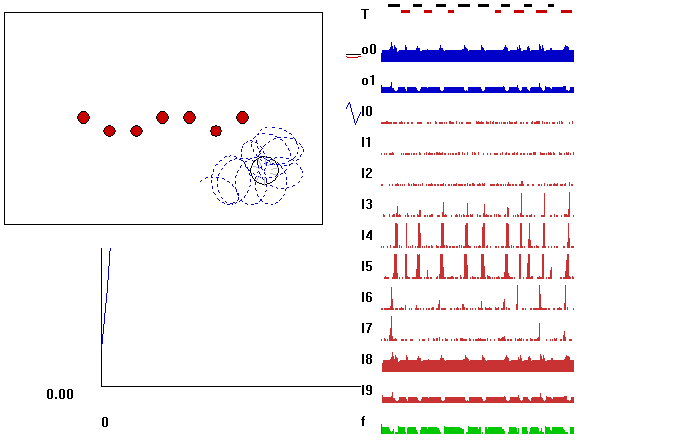


1. Testing the best individuals of the generation

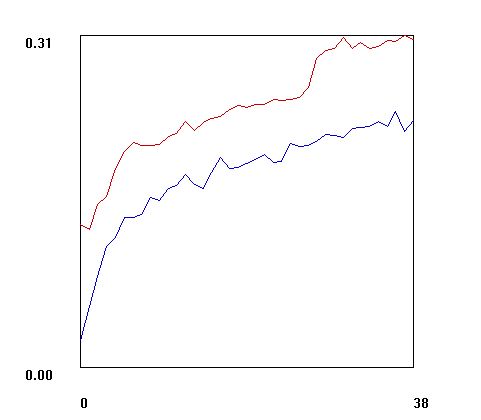
Now varying the environment parameters



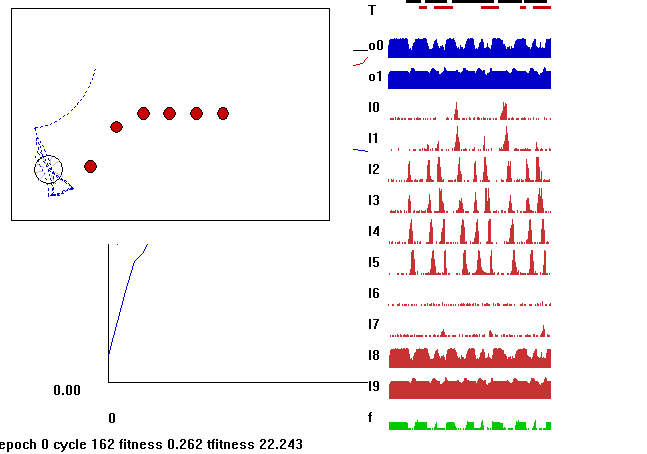
Random individual



Other parameters



Best individual



## Discussion

From all of the graphics of the evolution of the cost functions over training time for the obstacle avoidance task we notice that the longer we let individuals of a generation train, the better they perform. The cost functions are a measure of quantifying how good the individuals are performing, and the continuous variations are justified. By putting into practice machine learning, the Khepera robots learn new behaviors, in this case how to avoid arbitrarily placed obstacles.

We notice as well, by running the Test Best Ind and Test Ind, how significantly a well-trained robot performs than just a random individual.

In the beginning obstacle placement may be completely arbitrary. But, using previous environment parameters, we could train the individuals in steps, and notice which obstacles cause the most problems. Than by saving the trained agents, a way to further improve would be simulating the obstacle avoidance task with those particular obstacles.

# Conclusions

In light of all of the sections presented above, it is underlined the fact that the line following and obstacle avoidance tasks are not valuable just in terms of educational robotics, but going back to the basics represents a good way to evaluate and understand the approach of more complex problems.

Breaking down each stage of development, putting in balance the negatives and positives, while still looking back at previous designs and embracing parallel tasks is challenging, but necessary in engineering.

##### References

1. D. Floreano, S. Nolfi, Evolutionary Robotics, The Biology, Intelligence, and Technology of Self-Organizing Machines. Massachusetts: The MIT Press, 2000.
2. S. Nolfi, “Evorobot 1.1 User Manual”, Technical Report, Institute of Psychology, National Research Council (CNR), Rome, Italy, 2001.