

# Implementation of a Drosophila-inspired orientation model on the Eye-Ris platform

L. Alba<sup>†</sup>, P. Arena<sup>\*</sup>, S. De Fiore<sup>\*</sup>, L. Patanè<sup>\*</sup> R. Strauss<sup>‡</sup> and G. Vagliasindi<sup>\*</sup>

<sup>\*</sup>DIEES - Dipartimento di Ingegneria Elettrica, Eletttronica e dei Sistemi  
 Facoltà di Ingegneria - Università degli Studi di Catania  
 I-95125 Catania, Italy

Email: parena@diees.unict.it

<sup>†</sup>University of Mainz, Inst. f. Zool. III - Neurobiology  
 D-55099 Mainz, Germany

<sup>‡</sup>AnaFocus - Innovaciones Microelectrnicas S.L.  
 Av. Isaac Newton s/n. Pabelln de Italia 7th Floor. PT Isla de la Cartuja  
 E-41092 Sevilla, Spain

**Abstract**—A behavioral model, recently derived from experiments on fruit-flies, was implemented, with successful comparative experiments on orientation control in real robots. This model has been firstly implemented in a standard CNN structure, using an algorithm based on classical, space-invariant templates. Subsequently, the Eye-Ris platform was utilised for the implementation of the whole strategy, at the aim to constitute a stand alone smart sensor for orientation control in bio-inspired robotic platforms. The Eye-Ris v1.2 is a visual system, made by Anafocus, that employs a fully-parallel mixed-signal array sensor-processor chip. Some experiments are reported using a commercial roving platform, the Pioneer P3-AT, showing the reliability of the proposed implementation and usefulness in higher level perceptual tasks.

## I. INTRODUCTION

Experimental observation [2] had shown that walking fruit flies are attracted by near-by objects whose distance is estimated by the parallax motion of their images on the retina. The process in flies is not only selective to image motion created by the self-movement of the fly but also sensitive to object motion and to the pattern contrast of objects. Moreover, objects that are most attractive in the fronto-lateral eye-field, act as repulsive in the rear visual field. In a first step of processing, visual motion is extracted by means of elementary motion detectors (EMDs) of the correlation type [4]. This type of motion evaluation seems to be implemented in the visual system of flies, likely in invertebrates in general and also in vertebrates. Behavioral experiments on flying flies had shown that Drosophila prefers front-to-back motion over back-to-front. Parallax motion of the images of stationary objects is produced only during locomotion of the fly; close objects move with higher speed and larger amplitude over the retina than objects that are further away. Visual motion for distance estimation is most likely perceived by the well-studied array of correlation-type elementary movement detectors (EMDs) of the fly visual system [4], [5], [6]. Within known limits and characteristics these detectors increase their output with increasing speed of visual motion. If object motion interferes with distance estimation, then flies should

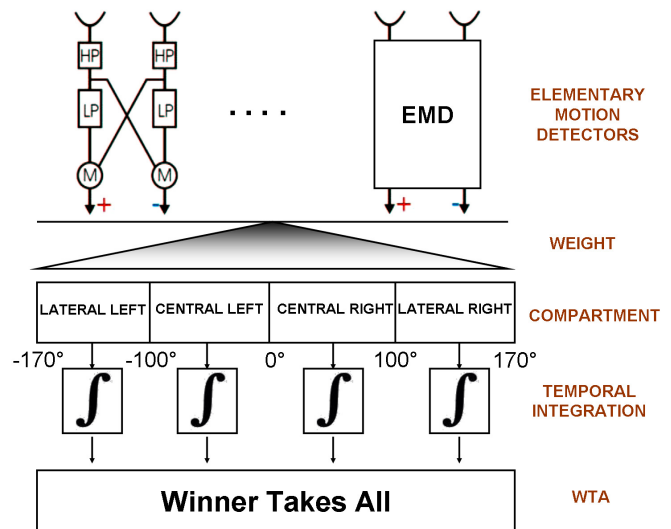


Fig. 1. Minimal model of the orientation behavior towards visual objects (HP: high pass filter; LP: low pass filter; M: multiplication stage).

prefer an oscillating object over a stationary counterpart at the same distance and a fast oscillating object over a slowly oscillating object. In flies, the correlation-type motion detector connects exclusively immediately neighboring photoreceptors. A schematic representation of a minimal model describing orientation behavior in fruit-flies was recently introduced by some of the authors and is reported in Fig 1 [1]. Visual motion information is extracted from a horizontal ring of ommatidia. Each Drosophila ommatidium has an acceptance angle of  $4.6^\circ$ . Correlation-type visual motion detectors connect exclusively immediately adjacent ommatidia. Each elementary motion detector is composed of two mirror-symmetrically oriented motion detectors. The visual input of the first detector is filtered by a high-pass filter and multiplied with the high-pass filtered signal of the second photoreceptor, which will be delayed by means of a low pass filter (EMD in Fig. 1). Motion output is integrated space-wise and time-wise in four

compartments (azimuth angles, both sides frontal  $0^\circ$  to  $100^\circ$  sideways and  $100^\circ$  sideways to  $170^\circ$  in the rear).

Behavioral experiments had shown, also, that motion in the fronto-lateral visual field triggers a reaction twice as fast than motion in the lateral to rear visual field. More weight is then given to frontal motion detector outputs than to rear detector outputs. If one of the four compartments reaches a threshold, it will determine the behavior of the robot for a while. If a frontal compartment wins the competition, the robot will turn towards the object, whereas if a lateral compartment wins the robot will turn away from the object.

## II. THE CNN IMPLEMENTATION

The algorithm described in Section 1 has been implemented on a Cellular Nonlinear Network structure. In order to emulate the acceptance angle of  $4.6^\circ$  of the drosophila ommatidium, a circle of 78 points equally spaced is extracted from the input pictures and translated in a row of a  $78 \times 78$  image. In this way it is possible, exploiting the CNN parallel computation capabilities, to process in a full image all the information coming from the ommatidia. All the operations described in Fig. 1 have been implemented as CNN templates. However, some modifications to the algorithm were required to allow a full CNN implementation, obtaining an equivalent results in terms of the robot performance. The first step is the high pass filter calculation:

$$HP(t) = I(t) - I(t-1) \quad (1)$$

that is the subtraction between the image acquired during the actual step  $I(t)$  and the image acquired the previous step  $I(t-1)$ . This operation was performed applying the template in (2), using  $I(t)$  as initial state and  $I(t-1)$  as input image and performing only one iteration.

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & -1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad I = 0 \quad (2)$$

In Fig. 1, LP signifies a temporal low-pass filter with a time constant  $\tau = 1.25$ . The low-pass filtering leads to a temporal delay of the receptor signals and it is defined according to:

$$LP(t) = (1 - 1/\tau)LP(t-1) + (1/\tau)HP(t) \quad (3)$$

This operation was performed using the template in (4). In this case the initial state is  $HP(t)$  while the input is  $LP(t-1)$ . The output of the operation is  $LP(t)$ .

$$A = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1/\tau & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & (1-1/\tau) & 0 \\ 0 & 0 & 0 \end{bmatrix}, \quad I = 0 \quad (4)$$

The subsequent step of the EMD implementation, i.e. the calculation of the cross product  $\phi$  between HP and LP (Eq. 5), cannot be implemented on CNN since it requires a convolution between images that cannot be implemented using a spatio-invariant template:

$$\phi(t) = HP_i * LP_{i+1} - HP_{i+1} * LP_i \quad (5)$$

for  $i$  from 1 to 78.

To overcome the infeasibility of a convolution between images on CNN, it was chosen to perform an equivalent calculation of the  $\Phi$  converting the output of the previous steps in binary images. In this way it is still possible to perform parallel processing on all the ommatidia, at the expenses of the introduction of other filtering parameters (i.e. the threshold values), to be chosen suitably. To calculate  $\Phi$ , the multiplication has been substituted by a *AND* operation while the subtraction and absolute value with a *XOR*. Since the two multiplications should be performed between *HP* and *LP* of contiguous ommatidia, the *LP* (*HP*) was shifted to the right before performing the first (second) multiplication.

The next steps are the spatial integration in the various compartments and the temporal integration among the various iterations of the algorithm. To perform the spatial integration, the output of the previous step ( $\Phi$ ) is *ANDed* with four different masks which identify the different compartments. The black pixels output of each *AND* operation represent the points where a transition (i.e. a moving object) was identified inside the compartment. The total number of black pixels in each compartment represents, then, the spatial integration inside the compartment. To emphasize the output of the frontal part of the retina, the outputs of the central compartments are multiplied by two. The outputs of the weighted spatial integration are converted into four grayscale images which are then used among the various iterations of the algorithm to perform the temporal integration. At the end of each iteration, a threshold operation is performed on the outputs of the temporal integration. According to the sector where the fixed threshold is overcome an action is performed. In particular if the two outermost compartments exceed the threshold first, the corresponding action is to turn away from the winning compartment, while if the innermost compartments win, the corresponding action is to turn towards the local motion.

Figure 2 depicts a flow diagram of the whole algorithm as it was implemented on CNN.

The result of one iteration, together with some intermediate results, are reported in Figure 3. Each image represents a step in the algorithm, while each iteration is represented as a row in the images. Figure 3(a) depicts the input. Each row is a  $1 \times 78$  vector representing the ommatidia of the drosophila. Figure 3(b) is the output of the high pass filter (eq. 1) while figure 3(c) is the output of the *LP* calculation. Figure 3(d) and 3(e) are the binary version of *HP* and *LP* obtained applying the same threshold to both of them. Figure 3(f) and 3(g) are, respectively, the first and second operand of equation 5, while figure 3(h) is the output. Figure 3(i) is the output of the algorithm. The 1 by 78 vector is divided in 4 regions representing the four compartments of the retina. The value of the pixels in each compartment is the same and is the output of the spatial integration, i.e the sum of the pixels in the corresponding row of figure 3(h). After each iteration

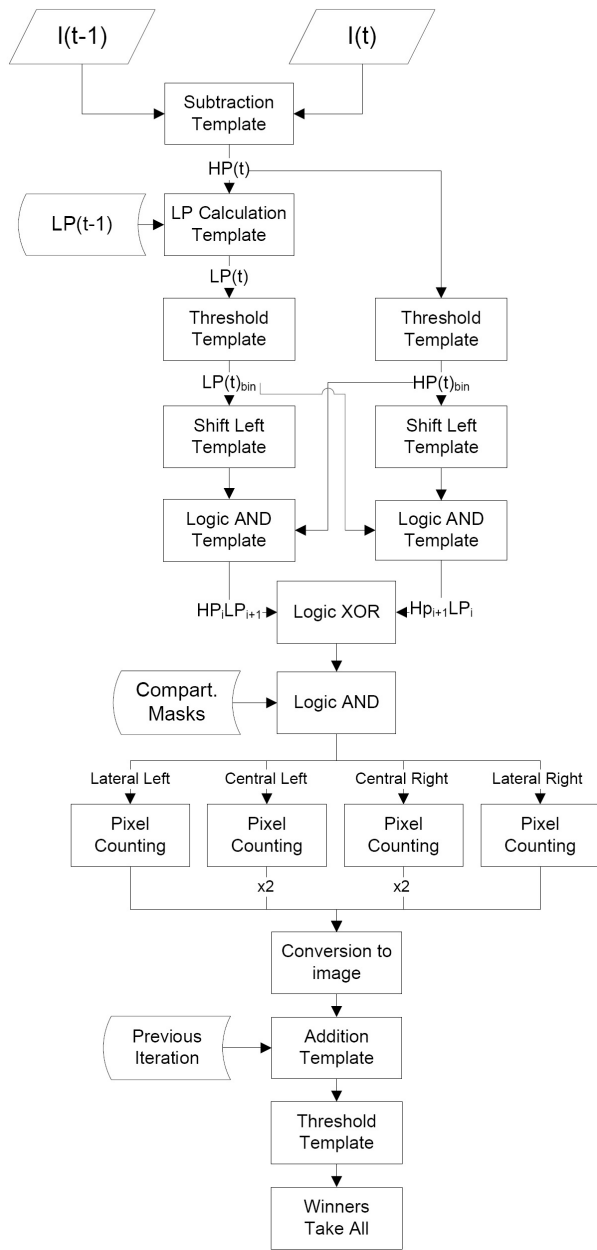


Fig. 2. Flow diagram of the whole algorithm as implemented on CNN

the new output of the spatial integration is added up to the previous value until one of the sectors overcomes a predefined threshold. In that case, according to the winning sector, an action is selected and all the compartments are reset to the initial value.

### III. THE EYE-RIS 1.2 PLATFORM

An hardware implementation of CNN is represented by the Eye-Ris platform [9], a visual system, made by AnaFocus [7], that uses a fully-parallel mixed-signal array sensor-processor chip. The Eye-RIS system implements, indeed, a bio-inspired architecture represented by the retina-like front-end which

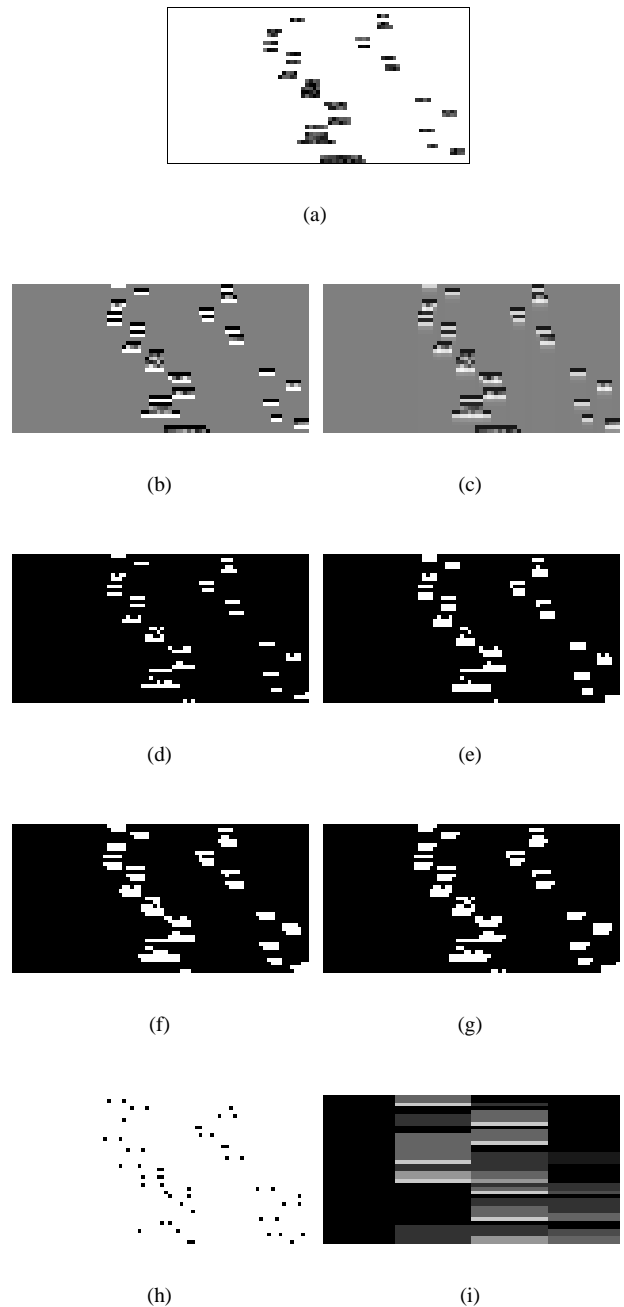


Fig. 3. Some steps of the EMD implemented on CNN. Each row in the images represent an iteration of the algorithm. (a) Input image. (b) Output of the HP filter. (c) Output of the LP filter. (d) HP thresholded binary version. (e) LP thresholded binary version. (f) First operand of equation 5. (g) Second operand of equation 5. (h) Output of equation 5, i.e.  $\Phi$ . (i) Output of the algorithm, after the spatial and temporal integration.

combines signal acquisition and embedded processing on the same physical structure. This is the Q-Eye chip, an evolution of the previously adopted Analogic Cellular Engines(ACE) [10], the family of stand-alone chips developed in the last decade and capable of performing analogue and logic operations on the same architecture. The Q-Eye was devised to overcome the main drawbacks of ACE chips, such as lack of robustness

and large power consumption.

Eye-RIS is a multiprocessor system since it employs two different processors: the Anafocus' Q-Eye Focal Plane Processor and the Altera's Nios II Digital Soft Core processors. The AnaFocus Q-Eye Focal Plane Processor (FPP) acts as an Image Coprocessor: it acquires and processes images, extracting the relevant information from the scene being analyzed, usually with no intervention of the Nios II processor. Its basic analog processing operations among pixels are linear convolutions with programmable masks. Size of the acquired and processed image is the Q-CIF (Quarter Common Intermediate Format) standard  $176 \times 144$ .

Altera NIOS II digital processor is a FPGA-synthesizable digital microprocessor (32-bit RISC  $\mu$ P at 70 MHz- realized on a FPGA). It controls the execution flow and processes the information provided by the FPP. Generally, this information is not an image, but image features processed by Q-Eye. Thus, no image transfer are usually needed in Eye-RIS, increasing in this way the frequency of operation. The platform is programmed through the Eye-RIS Application and Development Kit (ADK), an Eclipse-based software development environment. The Eye-RIS ADK is integrated into the ALTERA Nios II Integrated Development Environment (Nios II IDE). In order to program the Q-Eye, FPP code, a specific programming language, was developed. The Nios II is programmed, instead, using standard C/C++ programming language. In addition, the Eye-RIS ADK includes two different function libraries to ease developing applications. The FPP Image Processing Library which provides functions to implement some basic image processing operators such as arithmetic, logic and morphologic operations, spatio-temporal filters, thresholding, etc. The Eye-RIS Basic Library which is composed by several C/C++ functions to execute and debug FPP code and to display images. All of these features allow Eye-RIS Vision System to process images at ultra-high speed but still with very low power consumption.

#### IV. THE IMPLEMENTATION ON THE EYE-RIS

The algorithm described in the previous sections was also implemented on the Eye-RIS platform. The integrated CMOS sensor was exploited to acquire the image through a fish-eye lens positioned vertically upward. The acquired image provides a 360 grayscale view of the environment. The outputs of the EMDs are calculated according to a formula described in 5. As already stated, according to the model, the acquired image should be divided to emulate the acceptance angle of the drosophila ommatidium. However, in order to exploit the capability of the Eye-RIS visual processing system, when possible the processing was performed on the full image obtaining a parallel processing of the ommatidia.

A first implementation was performed considering the original algorithm reported in the block scheme of Fig. 1.

Fig. 4 shows the implementation of the EMDs structure on the Q-Eye chip using the images acquired from the fish-eye. The subsequent step of the EMD implementation, i.e. the calculation of the cross product  $\Phi$  between HP and LP (Eq. 5),

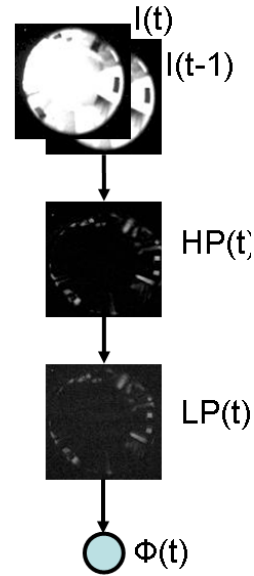


Fig. 4. Implementation of the Elementary Motion Detector (EMD)

as well as the remaining part of the model were implemented on the Nios II soft-core processor. At this stage, to emulate the acceptance angle of  $4.6^\circ$  of the drosophila ommatidium, a circle of 78 points equally spaced was extracted from the pictures.

The execution time of the whole algorithm, from the acquisition of the frame to the output of the robot action, is 100 ms. The part of the algorithm related to the EMD calculation is performed in 29 ms: 17 ms are required to perform all the operations on the Q-Eye, the remaining 12 ms is the time requested by Nios II to perform the calculation of the cross product between HP and LP (Fig. 4). The rest of the algorithm is still performed on the Nios II and represents the most time consuming part since it includes the integration operations. Compared to the previous implementation on the PC onboard of the robot which took about 350 ms, we obtained a relevant reduction of time per iteration.

To further exploit the parallel computational capabilities of the Q-eye chip, another implementation was proposed considering the simplified algorithm reported in Fig. 2 where the most problematic function, the cross product, was substituted with logical operations considering the input data not with gray-scale but with binary images. This modified version produces results qualitatively equivalent to the standard model but needs a fine tuning of the thresholds used during the image conversion.

The time needed to acquire the image from the panoramic lens and to extract the relevant information corresponding to a ring across the horizon is about 17 ms. This includes also the creation of a new image composed by only 78 pixels that will be used in all the successive steps as input data. The macro block of operations that in Fig.2 is reported from the beginning of the scheme to the pixel counting operations needs about 4 ms. Finally the pixel counting operation transfers the

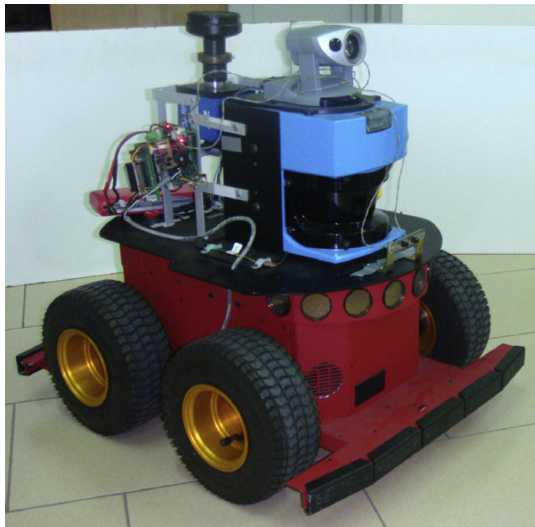


Fig. 5. The roving robot P3AT

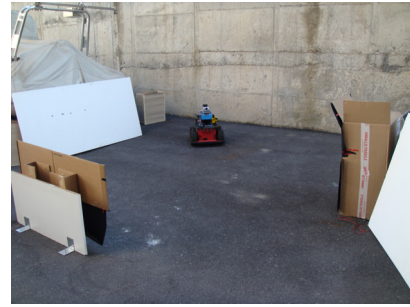
information about the compartment activities on the NIOS II where the rest of the algorithm is performed in about 1 ms. This implementation guarantees the highest performance in terms of time for iteration that is about 22 ms and the robot behaviour is equivalent to the previous implementation, so it represents the best trade-off.

The improvement obtained is important to generate more reliable command actions increasing the image acquisition rate having, in this way, a smoother control of the robot.

## V. EXPERIMENTAL SETUP AND RESULTS

### A. The robotic platform

The robot used for the experimental set-up is a standard platform, the Pioneer P3-AT robot built by MobileRobots inc [8]. It is a classic four wheeled rover controlled through a differential drive system, using encoders with inertial correction to compensate for skid steering. The robot is equipped with an embedded computer, wireless Ethernet-based communication, a laser scanner, a compass sensor, a gyroscope and a pan-tilt actuated color camera. Moreover it is equipped with eight forward sonars that sense obstacles from 15 cm to 5 m and five bumpers for collision detection. A picture of the robot is reported in Fig. 5. The robot can be controlled through a library suite (i.e. the ARIA library) and a 2D virtual simulation environment, named MobileSim, can be used instead of the real robot in a transparent way using the same control library. As shown in Fig. 5, we have customized the standard configuration including a hearing circuit, a CNN-based camera with panoramic lens and a gray-scale color sensor placed on the bottom of the robot, used as a low level target sensor, to detect black spots on the ground. These additional sensors are managed from the onboard computer using a microcontroller-based bridge.



(a)



(b)

Fig. 6. The robot in the arena from two different line of sight (a) and (b)

### B. Experimental results

The robot was placed into an arena created in an external environment (Fig. 6). Two target objects were placed in the arena and highlighted with a black cover to be more visible to the robot (Fig. 6(b)). To avoid the influence of the rest of the environment, a series of white panel were placed to cover black spots. The robot was, then, left free to move according to the output of the algorithm. Several experiments were conducted whose results were pretty similar. In Fig. 7, the trajectory followed by the robot is reconstructed exploiting the output of the gyroscope and encoders embedded in the robot. The lines represent the walls of the environment, previously reconstructed using the robot laser scanner. The rectangular boxes represent the targets. The small squared box is the starting point of the trajectory followed by the robot. As it is possible to observe in Fig. 7, the robot initially moves forward since the two targets are in the frontal compartments of the retina. Subsequently, since the target on the left is nearer than the other one, it causes a larger parallax effect on the EMDs and so becomes more attractive. When the robot reaches the target, the sonar distance sensors intervene to avoid the collision. In this case the robot is programmed to perform a  $90^\circ$  rotation in the best direction to avoid the obstacle. In this specific case, the robot turns right and moves forward as a consequence of a double action: on one side the repulsive action resulting from the presence of the first target on the

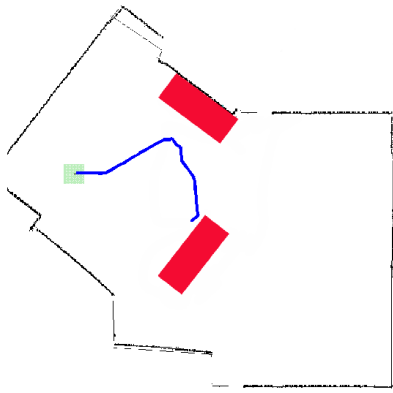


Fig. 7. A reconstruction of the robot trajectory during an experiment

rear compartments, on the other the attractive action due to the presence of the second target in the frontal compartments. Finally, when the repulsive action becomes less important and the attractive one is preponderant the robot reaches the second target.

## VI. CONCLUSION

In this paper a full CNN implementation of a minimal model for the orientation system in the drosophila is introduced. The first part of the paper describes the template based algorithm on a standard CNN, while the second part discusses the implementation on the Eye-ris system, exploiting the capabilities of both the focal plane processor of the visual Q-Eye chip, and the soft core processor embedded into the fpga contained into the Eye-ris system, at the aim to reach the best results. Experimental results of the navigation control of a roving robot are also reported showing the result of the drosophila-inspired orientation system implemented on the Eye-ris platform.

## ACKNOWLEDGMENT

The paper was partially supported by the EU funded project SPARK II “*Spatial-temporal patterns for action-oriented perception in roving robots: an insect brain computational model*”.

## REFERENCES

- [1] M. Mronz and R. Strauss, “Visual motion integration controls attractiveness of objects in walking flies and a mobile robot”, *IEEE / RSJ International Conference on Intelligent Robots and Systems*, Nice. pp. 3559-3564. (ISBN 978-1-4244-2058-2), 2008.
- [2] K. Neuser, T. Triphan, M. Mronz, B. Poeck and R. Strauss, “Analysis of a spatial orientation memory in Drosophila”, *Nature*, vol. 453, pp. 1244-1247, 2008.
- [3] A. Borst, “Detecting visual motion: theory and models”. *Rev Oculomot Res*, vol. 4, pp.3-27, 1993.
- [4] A. Borst, M. Egelhaaf, “Principles of visual motion detection”, *Trends Neurosci*, vol. 12, pp. 297-306, 1989.
- [5] F. Ilda, “Biologically inspired visual odometer for navigation of a flying robot”, *Robotics and Autonomus System*, vol. 44, pp. 204-208, 2003.
- [6] TR. Neumann, HH. Bulthoff, “Insect inspired control of Translatory flight”, *Advances in Artificial Life - Proceedings of the 6th European Conference*, Springer-Verlag, Berlin, pp. 627-636, 2001.
- [7] ANAFOCUS home page [Online]. Available: <http://www.anafocus.com>
- [8] MobileRobots home page [Online]. Available: <http://www.mobilerobots.com>

- [9] A. Rodriguez-Vazquez et al, “The Eye-RIS CMOS Vision System”, in *Analog Circuit Design*, Springer, pp. 15-32, 2007.
- [10] G. Liñan, R. Domiguez-Castro, S. Espejo, and A. Rodriguez-Vazquez, “ACE16K: A programmable focal plane vision processor with 128×128 resolution”, in *Proc. of ECTD 01*, Espoo, Finland, Aug. 2831, 2001.