Flies open a door to SLAM

Jean Louchet [1,2], Emmanuel Sapin [1]
[1]INRIA-Saclay, équipe APIS, 4 rue Jacques Monod, F-91893 Orsay Cedex France
[2]Artenia, 24 rue Gay-Lussac, F-92320 Châtillon, France
{Emmanuel.Sapin,Jean.Louchet}@inria.fr

Abstract. The "fly algorithm" is a real-time evolution strategy designed for stereovision. Previous work has shown how to process stereo image sequences and use an evolving population of "flies" as a continuously updated representation of the scene for obstacle avoidance in a mobile robot, and the support to collect information about the environment from different sensors. In this paper, we move a step forward and show a way the fly representation may be used by a mobile robot for its own localisation and build a map of its environment (SLAM).

Key words: evolutionary robotics, Fly algorithm, robot vision, Parisian evolution, SLAM, image registration.

1 The Fly Algorithm

1.1 Robotics and Parisian Evolution

Robot vision is known as a computationally expensive process [8]. Contrary to the general belief that artificial evolution is too slow and complex to suit real-time applications, we showed that the Fly algorithm [9, 11], an evolution strategy based on the Parisian evolution paradigm, is an efficient way to exploit asynchronous data delivery and fulfil the real-time constraints found in robot vision [2]. Parisian evolution essentially differs from conventional artificial evolution by its semantics. According to the general scheme of Parisian evolution [3], the problem's solution is represented by the whole population rather than the best individual alone.

A matching asynchronous robot path planner using the fly algorithm's output has been proposed in [1]. In this paper, we open a way to use the flies to enable a robot locate itself and build a map of its environment.

1.2 Flies and stereovision

A fly is defined as a 3-D point with coordinates (x, y, z). If the fly is on the surface of an opaque object, then the corresponding pixels in the two images will normally have highly similar neighbourhoods (figure 1). Conversely, if the fly is not on the surface of an object, their close neighbourhoods will usually be poorly correlated.

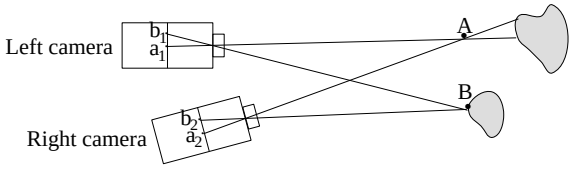

Fig. 1. Pixels b_1 and b_2, projections of fly B, get identical grey levels, while pixels a_1 and a_2, projections of fly A, which receive their illumination from two different physical points on the object's surface, get different grey levels.

The fitness function exploits this property and evaluates the degree of similarity of the pixel neighbourhoods of the projections of the fly, giving higher fitness values to those probably lying on objects surfaces:

$$fitness(indiv) = \frac{G}{\sum_{colours} \sum_{(i,j) \in N} (L(x_L + i, y_L + j) - R(x_R + i, y_R + j))^2}$$

- (x_L, y_L) and (x_R, y_R) are the fly's left and right projection coordinates, which can be calculated readily [7] using the camera calibration parameters,
- L and R the corresponding grey level values in the left and right images,
- N is a small (usually 5x5 or 7x7 pixel) neighbourhood.

The numerator G is an image gradient-based normalizing factor designed to reduce the fitness of the flies projecting into uniform regions.

The population is initialised randomly inside the intersection of the cameras' fields of view (figure 2). The values of z^{-1} are uniformly distributed between zero (or $1 / d_{max}$) and $1 / d_{min}$.

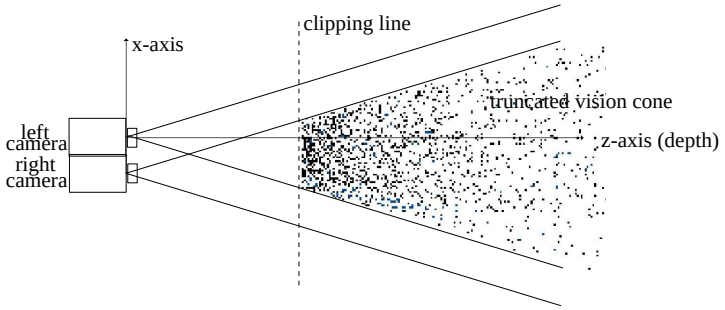

Fig. 2. The fly population is initialised within the intersection of the cameras fields of view.

1.3 The operators

Several versions of the algorithm have been tested: steady state or generational, using elitist ranking or tournament selection [1], with basically equivalent results.

2-D *sharing* reduces the fitness values of flies projecting into crowded areas.

Mutation allows extensive exploration of the search space. It uses an approximation of a Gaussian random noise added to the flies' chromosome parameters (x, y, z).

A *2-parent* and a *3-parent barycentric crossover operator* have been introduced in order to take into account the frequent straight lines or planar surfaces existing in real-world scenes.

An *immigration* operator, similar to the initialisation function, has been introduced in order to better detect new events in the scene.

1.4 Flies as a tool for robot vision

Compared to other state-of-the-art stereovision algorithms, the fly algorithm does not provide the most accurate 3-D representation of the environment, to be fair. However, it is extremely fast and robust. As far as we know, no other stereovision algorithm is able to cope with scene changes while the algorithm is working. Having a progressively improving analysis rather than a blind stage followed by an ultra-accurate

3-D polygonal representation, is highly regarded by the Robotics community. However, most benefits of the fly algorithm would be lost without matching robot controllers.

1.4.1 Processing image sequences while robot is moving

We have developed several methods to process stereo image sequences [10]. The simplest one is the *random approach*, which consists in keeping the same population evolving through frame changes. Thus, only the images (and therefore the parameters used by the fitness function) are modified while the fly population evolves. When motion is slow enough, using the results of the last step speeds up convergence significantly compared to using a totally new random population. This method may be seen as the introduction of a collective memory of space, exploiting the similarity between consecutive scenes. It does not alter the simplicity of the static method and does not require any algorithm change, but it is best suited to situations where the cameras are static while there may be movements in the scene.

The following stage consists of updating the flies' positions at each generation, using the odometric information available about the robot's motion in order to give better initial conditions and allow faster convergence of the fly population. This was the basis of the *proprioceptive fusion* described in [1].

In the next stage, introduced in this paper, we consider the possibility of reversing the flow of information: rather than injecting external information about the robot's motion into the fly algorithm, it is possible to do it the other way and exploit the shape of the fly population in order to extract information about the robot's motion.

1.4.2 Obstacle avoidance and multi-sensor fusion

The first controllers we developed to match the fly algorithm, were aiming at obstacle avoidance. The fastest and most elegant one [1] uses harmonic functions [4], permanently updated by the fly density, in order to generate and update in real time a smooth obstacle avoidance trajectory towards a given target. Dirichlet boundaries are created (1 for high fly densities, 0 for the target position) and the harmonic function is updated. The steering command of the robot is the gradient of the harmonic function.

We have shown it is possible to use several sensors providing independent information sources on the surrounding scene and the robot's position, such as sonars or wheel position coders, and fuse them through the introduction of corresponding additional terms into the fitness function. This sensor fusion technique keeps the main properties of the fly algorithm: asynchronous processing, no low-level image preprocessing or costly image segmentation, fast reaction to new events in the scene.

2 A robot understanding its environment: the SLAM problem

2.1 Classical approaches to SLAM

On a real-world robot, odometric data given by wheel rotation coders give an estimation of the robot's position. In most applications, wheel rotation is under control of the trajectory planner. The actual robot trajectory does not exactly match the target trajectory due to "trajectory noise" caused by factors such as wheel slipping, tyre wear or uneven ground surface. SLAM (Simultaneous Localisation And Mapping) [5, 6] allows the robot to use its vision capabilities to extract odometry information, progressively build a map of its environment and localise itself within this map. One of the main problems is how to deal with noise and errors on the positions of objects.

2.2 Flies and SLAM

2.2.1 Defining a pseudo-distance between fly populations

Due to the evolutionary (and therefore stochastic) nature of the Fly algorithm, two runs on the same stereo image pair will generally not give the same population of flies. In order to achieve localisation based on fly population 3-D patterns, we had to define some sort of "pseudo-distance" between fly populations. We now need a couple of definitions.

Let A, B two fly populations. We say that $A \equiv B$ (A "equivalent" to B) if A and B are the results of two different runs of the Fly algorithm on the *same* image pair.

Let F be the set of 3-D positive isometries (i.e. displacements), I the identical transformation.

We don't need the pseudo-distance to fulfil the usual properties of distances. In fact, the ideal property we would dream of for a pseudo-distance δ is:

$$if\ A \equiv B,\ \forall f \in F - \{I\},\ \delta(A, B) < \delta(A, f(B))$$

In other terms, given two equivalent populations, any displacement to one of them will give a greater pseudo-distance between them.

Let us call d a distance (e.g. the Euclidean distance) in the affine 3-D space.

The Hausdorff distance:

$$\delta(A, B) = \max_{a \in A}\left(\min_{b \in B} d(a, b)\right)$$

is not usable in our application: if applied to two equivalent populations $A \equiv B$ it is extremely sensitive to any single fly missing in either population [12]. What we actually need is a pseudo-distance giving low values for $\delta(A, A)$ but about equally low values for $\delta(A, B)$ if $A \equiv B$. In other terms, $\delta(A, B)$ will have to be low if for each fly $a_i \in A$ there exists a fly $b_i \in B$ close enough to a_i . In order to preserve as much as possible the analytic regularity of the pseudo-distance, we use the harmonic mean:

$$harmmean(X) = \frac{N}{\sum \frac{1}{x_i}}$$

which is not too sensitive to high values. Therefore, with

$$A = \{a_i\}\ i \in [1, n],\ B = \{b_j\}\ j \in [1, p]$$

we define the pseudo-distance between populations A and B:

$$\delta(A, B) = \frac{p}{n} \sum_i \frac{1}{\sum_j \frac{1}{\varepsilon + d(a_i, b_j)}}$$

The parameter $\varepsilon > 0$ avoids divisions by zero. Obviously, δ is not a distance. It lacks at least two properties of distances, as $\delta(A, B) \neq \delta(B, A)$ and $\delta(A, A) \neq 0$. The real point is to know if and how it can be used for self-localisation.

2.2.2 Experimental properties

We tested this technique with stereo images. First, we studied how the pseudo-distance between one population and the same population behaves when one of the

populations is shifted or rotated (figures 6 and 7). Then, we did similar experiments but comparing one reference population to different but equivalent populations (i.e. obtained from the same stereo image pair (figures 8 - 10). In all our experiments, we obtained smooth, convex curves indicating that finding their minimum only requires in practice a basic gradient descent technique. In figure 6, the position of minima shows that fly populations registration based on the detection of pseudo-distance minima will give reasonably small registration errors. A histogram of registration errors, taken on 100 equivalent fly populations, is given in figure 10, showing a registration error standard deviation of 16 millimetres in x, 15 millimetres in y and 2.77 degrees (0.048 radian) angle.

Fig. 3. The “corridor” scene (stereo pair).

Fig. 4. Flies resulting of one particular run of the Fly algorithm, here projected on the left image (printout contrast has been reduced in order to enhance the visibility of flies).

Fig. 5. Projection of the same flies on the right image.

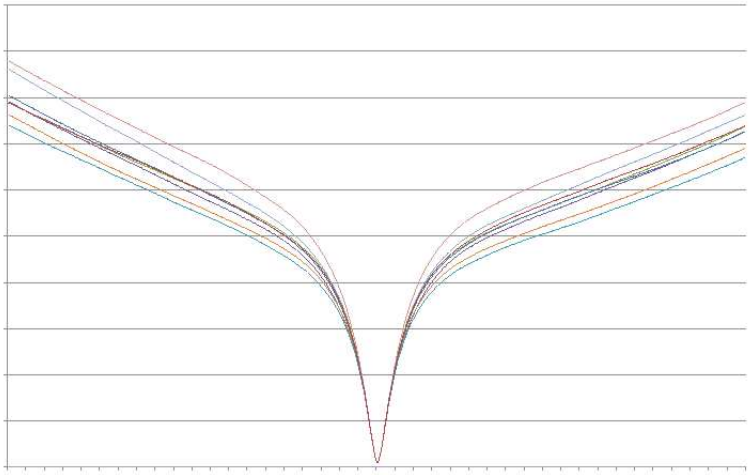

Fig. 6. Pseudo-distances between 10 fly populations and their replicas, in function of translation *x*. Each graduation on the horizontal axis represents 5 millimetres.

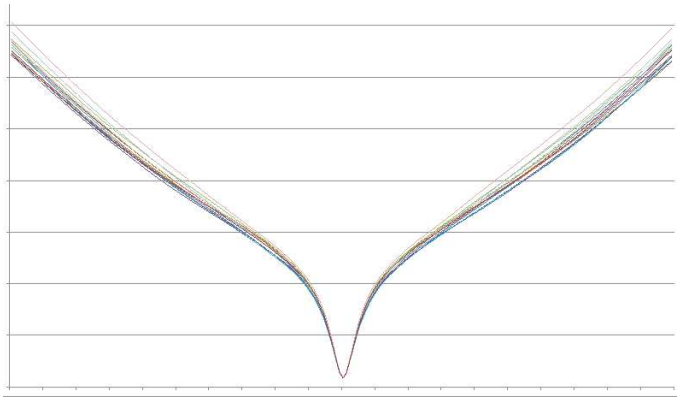

Fig. 7. Pseudo-distance between 10 populations and their replicas, in function of the rotation angle. Each graduation on the horizontal axis represents about 9 degrees.

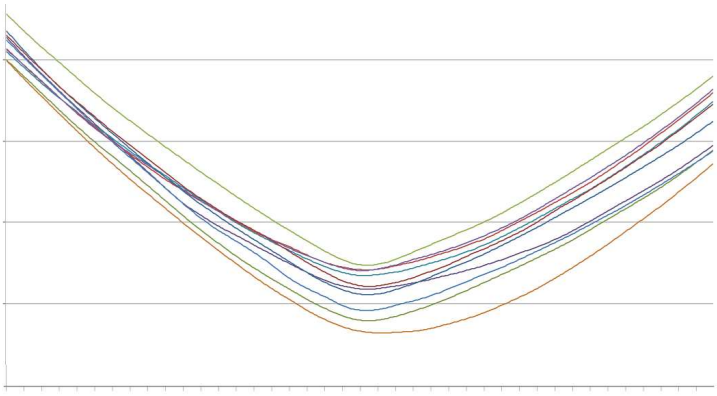

Fig. 8. Pseudo-distances between a reference population and 10 equivalent populations, in function of translation *x*.

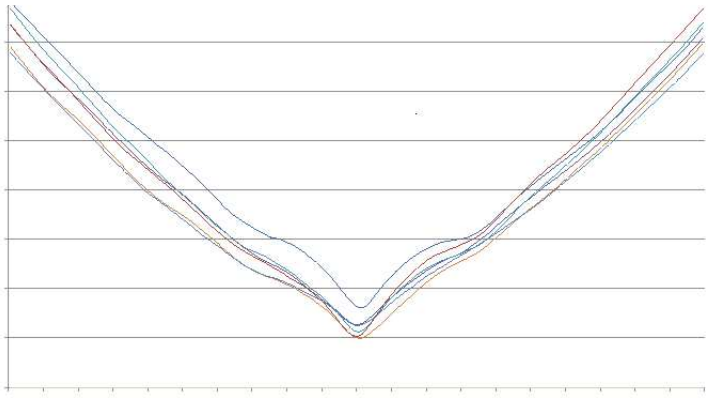

Fig. 9. Pseudo-distances between a reference population and 10 equivalent populations, in function of the rotation angle.

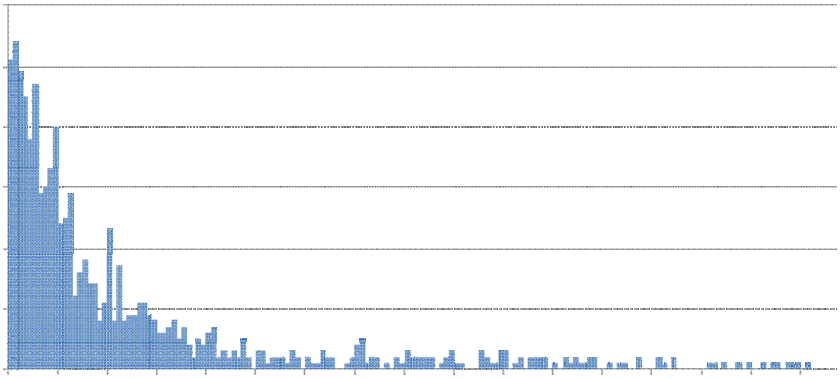

Fig. 10. Histogram of the x - position of the minima of pseudo-distance curves, using 1000 equivalent populations. Each graduation on the horizontal axis represents an error of 5 millimetres. In this series of experiments, standard deviation is 16 millimetres in *x*, 15 millimetres in *z* and 2.77 degrees of angle.

3 Fly populations registration

The next step is now, rather than correlating equivalent fly populations, to examine how the pseudo-distance can be used to register fly populations produced from stereo pairs taken from different points of view, in order to derive the camera's motion. Here, we are using two stereo pairs from the same scene, taken from different points of view. The camera assembly has been moved horizontally on a trolley, only allowing three degrees of freedom (x, z, θ). Figures 11 to 13 show statistical results of fly population registration. All our experiments on different scene types showed the pseudo-distance remains a convex function of (x, z, θ) and its minimum can be found using simple methods. On this example, the standard deviation of camera localisation is 19 *mm* in *x*, 21 *mm* in *z* and 0.85 degrees in angle.

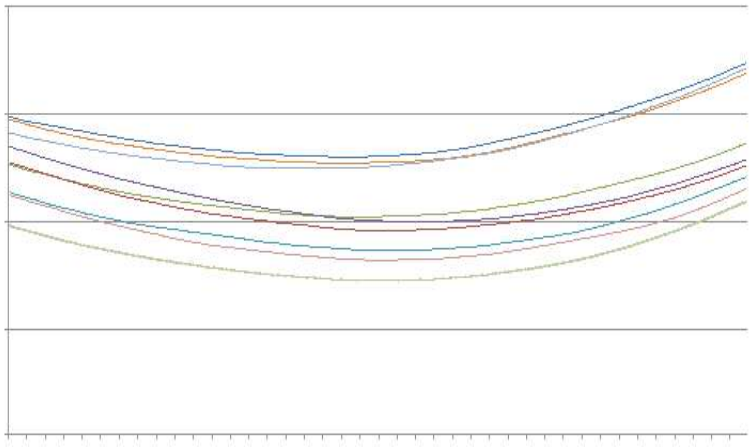

Fig. 11. Pseudo-distances between a population generated from one point of view, and 10 populations generated from another point of view, in function of the translation *x* from the optimal registration position.

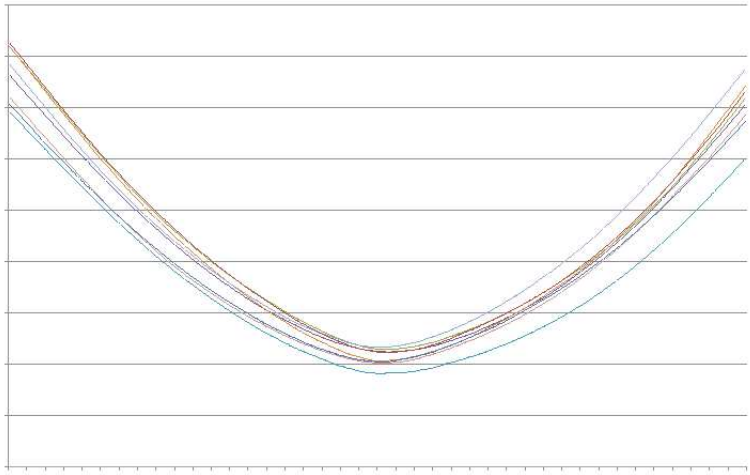

Fig. 12. Pseudo-distances between a population generated from one point of view, and 10 populations generated from another point of view, in function of the rotation angle around the optimal registration position.

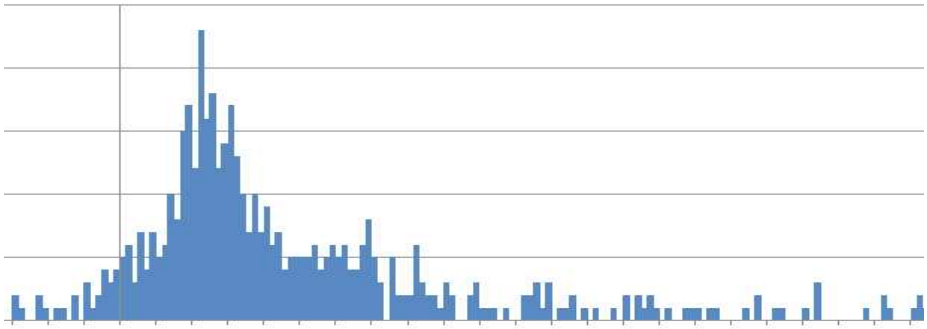

Fig. 13. Histogram of the x - position of the minima of pseudo-distance curves. We took 1000 instances of fly populations generated with the first stereo pair, and for each one we generated an instance of the fly population from the second stereo pair then calculated the position of the optimal (x, z, θ). The maximum corresponds to the value of the actual translation x. Each graduation represents a displacement of 5 millimetres. The histograms with z and θ are similar.

4 Conclusion

The research described above on stereovision-based self-localisation using the Fly algorithm, opens the way to a new, lightweight mobile robot localisation method. Further research will be oriented towards a more complete scheme which would store the fly populations found during the robot's movement and progressively build a fly-based mapping of the robot's environment. Exploiting a great number of continuously updated fly populations should improve geometrical precision. This will hopefully open a new route to fast resolution of the SLAM problem.

References

1. Boumaza, A., J. Louchet, *Mobile robot sensor fusion using flies*, S. Cagnoni et al. (Eds.), Evoworkshops 2003, LNCS 2611, pp. 357-367, Springer, 2003.
2. Boumaza, A., J. Louchet, *Using Real-time Parisian Evolution in Robotics*, EVOIASP2001, Artificial Evolution in Image Analysis and Signal Processing, LNCS 2037, pp. 288-297, Springer, 2001.
3. Collet, P., E. Lutton, F. Raynal, M. Schoenauer, *Individual GP: an Alternative Viewpoint for the Resolution of Complex Problems*, Genetic and Evolutionary Computation Conference GECCO99, W. Banzhaf, J. Daida, A.E. Eiben, M.H. Garzon, V. Honovar, M. Jakiela, R. E. Smith (Eds.) Morgan Kaufmann: San Francisco, CA,1999.
4. Connolly, C. I., J. B. Burns, R. Weiss, *Path planning using Laplace's equation*, Proceedings of the IEEE Conference on Robotics and Automation ICRA'90, pp. 2102-2106, May 1990
5. Davison. A. J., *Real-time Simultaneous Localisation and Mapping with a Single Camera*. In Proceedings of the Ninth IEEE International Conference on Computer Vision, volume 2, pp. 1403-1410, 2003.
6. Dissanayake, M., P. Newman, S. Clark, H.F. Durrant-Whyte & M. Csorba. *A Solution to the Simultaneous Localization and Map Building (SLAM) Problem*. IEEE Transactions on Robotics and Automation, vol. 17, no. 3, pp. 229-241, 2001.
7. Haralick, R. M., *Using Perspective Transformations in Scene Analysis*, Computer Graphics and Image Processing 13, pp. 191-221, 1980.
8. Horn, B. H. , *Robot Vision*, McGraw Hill, 1986.
9. Louchet, J., *From Hough to Darwin : an Individual Evolutionary Strategy applied to Artificial Vision*, Artificial Evolution 99, LNCS 1829, pp. 145-161, Springer, 2000.
10. Louchet, J., M. Guyon, M. -J. Lesot, A. Boumaza, *Dynamic Flies : a new pattern recognition tool applied to stereo sequence processing*, Pattern Recognition Letters, Elsevier Science B.V., 2002, vol. 23, no 1-3, pp. 335-345.
11. Louchet, J., *Using an Individual Evolution Strategy for Stereovision*, Genetic Programming and Evolvable Machines, Vol. 2, No. 2, June 2001, Kluwer Academic Publishers, pp. 101-109.
12. Perez-Garcia, A., Ayala-Ramirez, V., Sanchez-Yanez, R.E., Avina-Cervantes, J.G., *Monte Carlo Evaluation of the Hausdorff Distance for Shape Matching*, Lecture Notes on Artificial Intelligence (Proc. of the CIARP'2006), Nov. 2006. LNCS Vol. 4225 pp. 686-695.