

We are IntechOpen, the world's leading publisher of Open Access books Built by scientists, for scientists

6,900

Open access books available

186,000

International authors and editors

200M

Downloads

Our authors are among the

154

Countries delivered to

TOP 1%

most cited scientists

12.2%

Contributors from top 500 universities



WEB OF SCIENCE™

Selection of our books indexed in the Book Citation Index
in Web of Science™ Core Collection (BKCI)

Interested in publishing with us?
Contact book.department@intechopen.com

Numbers displayed above are based on latest data collected.
For more information visit www.intechopen.com



Object Recognition and Tracking Using the Particle Estimator

Edgardo Comas and Adrián Stácul

Abstract

In this chapter we describe the particle estimators and its effectiveness for tracking objects in video sequences. The particles estimators are specifically advantageous in transition state models and measurements, especially when these are non-linear and not Gaussian. Once the target object to follow has been identified (in position and size) its main characteristics are obtained using algorithms such as FAST, SURF, BRIEF or ORB. As the particle estimator is a recursive Bayesian estimator, where observations update the probability of validating a hypothesis, that is, they use all the available information to reduce the amount of uncertainty present in an inference or decision problem. Therefore, the main characteristics of the object to follow are those that will determine the probability of validating the hypothesis in the particle estimator. Finally, as an example, the application of a particle estimator is described in a real case of tracking an object in a sequence of infrared images.

Keywords: recognition, tracking, estimator, image analysis, image processing

1. Introduction

The first step in tracking object in an image sequence is to identify the reference object to be tracked; this will allow determining its attributes to carry out its identification by means of some of the main characteristics of the image of the object, such as the characteristics points. It should be noted that if is known: the initial position of the object to be tracked in the camera coordinates and the mathematical model of the camera, it is possible, in addition to tracking the object, to estimate its coordinates and moving in this reference system. One of the techniques applied for object tracking to which we will particularly refer is the particle estimator. This technique is a special type of Monte Carlo sequential method, one of its main advantages being its applicability to any model of transition of states and observations, especially when these are non-linear and non-Gaussian [1]. Although the Kalman estimator, which is applied to systems where the system evolution and measurement models are linear, with Gaussian noise, and with known mean and variance; has extensions to nonlinear models by applying techniques to achieve linearity, the Gaussian additive noise constraint cannot be overcome [1, 2].

Therefore, the particle estimators do not have the restrictive hypothesis of the Kalman estimator, so they can be applied to non-linear models with non-Gaussian and multimodal noise, where the reliable numerical estimate is a function of an adequate number of samples [3].

This estimator distributes N particles over the image, and the observations made on each one of them update their probability of validating a hypothesis, that is, they use all the available information to reduce the uncertainty present in an inference or decision problem [2]. In some cases where the images are not clear or noisy, particularly those acquired through infrared cameras, it is necessary to make an improvement before applying the estimator; generally this improvement is based on a reduction in the incidence of the background image, for the special case of infrared images subtracting the value from the mean intensity and modifying its histogram to increase the contrast result a good choice.

Faced with rapid and unpredictable movements of the referent or the camera, the resampling process considers a scattering value based on the number of valid particles, so that the area covered during tracking is dynamically modified. Basically, the particle estimator provides us with a framework, in which it is possible to insert different algorithms for the recognition of the reference image in each particle; some of them are SURF, BRIEF, ORB, etc. These algorithms have the ability to generate a set of invariant features against some image variations, such as: scaling, rotation, illumination and with robustness against occlusion conditions.

2. Particle estimator tracking

To define the state estimation problem let us consider the system model, composed of the state evolution and observation models described by the following equations:

$$X_k = f(X_{k-1}, v_{k-1}), \quad (1)$$

$$Z_k = h(X_k, n_k). \quad (2)$$

Where $X \in R^n$ contains all the state variables that will be dynamically estimated, f is the non-linear function of the state variables, $v \in R^n$ represents the state noise system, $Z \in R^n$ are all observations related with the state variables by (Eq. (2)), $n \in R^n$ is the measurement noise, and h is known as an observation model. Remembering that $P(a|b)$ is the conditional probability of a if b , then the evolution and observation models given by (Eqs. (1) and (2)) are based on the following hypotheses referring to the following sequences [2, 4]:

- a. $X_k, k = 1, 2, \dots$ is a Markov process

$$P(X_k | X_0, X_1, \dots, X_{k-1}) = P(X_k | X_{k-1}), \quad (3)$$

- b. $Z_k, k = 1, 2, \dots$ is a Markov process regarding the historical data of X such that

$$P(Z_k | X_0, X_1, \dots, X_k) = P(Z_k | X_k), \quad (4)$$

- c. and the sequence of past observations only depends on its history, that is

$$P(X_k | X_{k-1}, Z_{k-1}) = P(X_k | X_{k-1}). \quad (5)$$

Considering that the system and observation noises $v_i \wedge v_j$ and $n_i \wedge n_j$ are mutually independent of $i \wedge j$ and also the initial state for all $i \neq j$; and on the other hand

we know that $P(X_0|Z_0) = P(X_0)$, then, the two-step Bayesian estimator, prediction and update allows us to obtain the probability density $P(X_i|Z_i) = P(X_i)$ [3, 5].

The particle estimator represents the posterior probability density of a random set of samples with their probabilities of validation with the hypothesis, so it is possible to estimate the most likely particle from this set. When we make the number of particles approaches to infinity this process approaches to the a posteriori likelihood function, and the solution approaches an optimal Bayesian estimator [5].

To go into the details of the particle estimator for object tracking, we will rely on the importance sampling method, taking a set of samples from the state space, which characterizes the a posteriori probability density function $p(X_{0:k}|Z_{1:k})$ for the state:

$$X_{0:k}^i = \{X_i, i = 0, \dots, k\}, \quad (6)$$

while that the corresponding observations are $Z_{0:k} = \{Z_i, i = 0, \dots, k\}$, then, the a posteriori density in t_k can be approximated by:

$$p(X_{0:k}|Z_{1:k}) \approx \sum_{i=1}^N W_k^i \delta(X_{0:k} - X_{0:k}^i), \quad (7)$$

where $\delta(\cdot)$ is Dirac's delta function, N the total number of particles and $\{W_k^i\}_{i=1}^N$ are the assigned weighting.

Considering the hypotheses corresponding to the expressions (Eqs. (1) and (2)) the density a posteriori (Eq. (7)) can be written as [6, 7]:

$$p(X_k|Z_{1:k}) \approx \sum_{i=1}^N W_k^i \delta(X_k - X_k^i), \quad (8)$$

and the evaluation of the weights within the importance sampling principle assumes that there is an evaluable probability density function $p_{(X)}$ such that:

$$W_k^i \propto p(X_k|Z_{1:k}) \quad (9)$$

and W_k^i are normalized according to (Eqs. (10) and (11)).

$$\sum_{i=1}^N W_k^i = 1, \therefore \quad (10)$$

$$W_k^i = \frac{w_k(X_{i=1:n}^i)}{\sum_{j=1}^N w_k(X_{j=1:n}^j)} \quad (11)$$

This algorithm has a common problem known as the degeneracy phenomenon, which manifests itself after a few states, where all but a few particles (usually one) have negligible weight [3, 6]. This can be solved resampling the particles, however this creates another problem, which is the increasing information uncertainty arising in the random sampling process [8]. However, this problem can be detected by means of what is known as the effective sample size N_{eff} , which can be estimated by means of the (Eq. (12)).

$$N_{eff} = \frac{1}{\sum_{i=1}^N (W_k^i)^2} \quad (12)$$

When all particles have the same weight, i.e. $W_k^i = \frac{1}{N}$, for $i = 1, \dots, N$, then the effectiveness is maximum and equal to $N_{eff} = N$, but in the case where all but one particle has zero weight, the effectiveness is minimum and equal to $N_{eff} = 1$ [7].

While the correct choice of the probability density function $p_{(X)}$ to evaluate the particles weights (Eq. (9)), minimizes the problem of the degeneracy phenomenon; but to solve it, a resampling has to be incorporated into the algorithm; this incorporation is known as the Sequential Importance Resampling (SIR). This technique is applied in the case where the effective sample size N_{eff} falls below a threshold value N_T , its effect is to remove particles with small weights and replicate those with greater weights.

2.1 Particle estimator algorithm

In the following, we describe the six steps of the particle estimator algorithm applied to video object tracking:

a. Design:

- An observation function $H_{(k)}$, used for the evaluation of the similarity probability for each particle with the referent object.
- Determine whether image pre-processing is required.
- The number of particles N .
- The threshold number of particles N_{eff} , to determine which type of resampling strategy to use.
- The noise distribution function $\chi_{(k)}$, applied for resampling.

b. Initialization:

- Identification on the image the object to be tracked in its position and size; this operation is performed by the user and defining the reference particle.
- Determination of its main characteristics by means of the observation function $H_{(k)}$; generating the information for the identification of the reference particle.
- Generation of a set of N particles in random position over the whole image, if there is a priori information of its location; this is used for the positioning of the particles centered on it, and over this a random distribution.
- The set of particles are initialized with the normalized weights, with the same values $W_k = \frac{1}{N}$.

c. Update:

- For each particle, their normalized weights are calculated, based on the state probability of similarity to the reference.

- From the particles set, extract a sub set with the particles most likely to match the reference particle.
- With the subset of particles most likely to match the reference particle, the most probable location and size of the object is determined a priori.

d. Resampling:

- The effective number of particles N_{eff} is evaluated, and if it is lower than the threshold value N_T , the lowest weight particles are discarded.
- From this new subset of particles most likely to match the reference particle, the new set of N particles for the next state is created.
- The state of this new set of particles is modified by introducing the additive noise $\chi_{(k)}$, that brings variability to the system.

e. Completion:

- You are returned to the Update stage c), as long as the data sequence is not finished.

2.2 Observation function

Observation functions are those that allow me to extract the main characteristics of an image. In the particle estimator they are used to obtain the main characteristics of the reference image and those of the particles. These sets of main characteristics allow determine the probability of similarity between the reference and each particle. Some of the main algorithms are described below.

2.2.1 Features from accelerated segment test (FAST) algorithm

The FAST algorithm is basically searching over the whole image the points where the changes in intensity in all directions are significantly (corner detection method) [9]. The principal advantage of this algorithm is its high speed performance, and is very suitable for real time applications in computer vision processing. Exist several technics to find a characteristic point in an image; two of this is described below.

In the first one, and as parameters of the algorithm, a threshold value T and a radius r are defined for the evaluation. On the pixel p to evaluate and which has an intensity I_p , a Bresenham circle of radius r is considered (see **Figure 1**).

This circle of radius r defines a set of N points, if in this circle there is a set of n pixels whose intensity is greater than $(I_p + T)$ or less than $(I_p - T)$, then the pixel p can be considered as a characteristic point. The values taken by the authors after the experimental results are: $n \geq 0.75N$ to consider a pixel p as a characteristic point, and the value of the radius $r = 3$, which defines $N = 16$ and $n = 12$.

In a first step, the intensity I_p of pixel p is compared with the intensity of the pixels 1, 5, 9, and 13, if 3 of these 4 pixels meet the threshold criteria, then it is checked if there are at least 12 pixels that meet with this criteria to consider it as a characteristic pixel.

This procedure must be repeated for all pixels in the image and its drawbacks are: for values of $n < 12$ a large number of characteristics points are generated, and

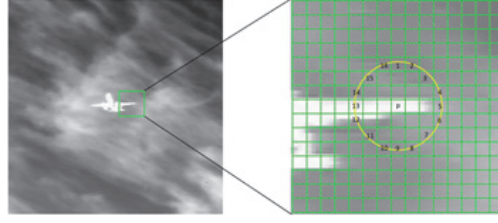


Figure 1.
A Bresenham circle of radius r .

having to evaluate all points of the circle slows down the algorithm. To improve the speed of the algorithm a proposal of the authors is to give it a machine learning approach [10].

In the second one, known as Fast Radial Blob Detector (FRBD), the technique consists of applying the Gaussian Laplacian filter to an image $I_{(x,y)}$, the Laplacian operator, as well as detecting the edges very well also detects the noise very well [11]. Therefore a Gaussian filter must be previously applied to the image to reduce its noise level; a Gaussian kernel of width σ to convolve with the image is represented by the (Eq. (13)) to suppress the noise before using Laplace for edge detection (Eq. (14)), and finally (Eq. (15)) represent the kernel of the Gaussian Laplacian filter to convolve with the image (Eq. (16)).

$$G_{(x,y,\sigma)} = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\left(\frac{x^2+y^2}{2\sigma^2}\right)} \quad (13)$$

$$L_{(x,y)} = \nabla^2 I_{(x,y)} = \frac{d^2 I_{(x,y)}}{dx^2} + \frac{d^2 I_{(x,y)}}{dy^2} \quad (14)$$

$$LoG_{(x,y)} = \Delta G_{(x,y,\sigma)} = \frac{d^2 G_{(x,y,\sigma)}}{dx^2} + \frac{d^2 G_{(x,y,\sigma)}}{dy^2} \quad (15)$$

$$LoG_{(x,y,\sigma)} = \Delta G_{(x,y,\sigma)} * I_{(x,y)} \quad (16)$$

This kernel $\Delta G_{(x,y,\sigma)}$ showed in **Figure 2**, is a feature detector because it finds regions where the image gradients are changing quickly for example blobs, corners and edges.

This FRBD algorithm goes one step further by using a second-order finite-difference approximation on the filtered image. An approximation to the LoG but which can be computed more rapidly is the Difference of Gaussian (DoG) operator, this approximation using a second-order finite differencing which estimates how the filtered image changes at a given pixel. A circle of radius r is constructed with center in a pixel p , and sampled pixels in eight discrete directions are evaluated, refer to the central pixel (**Figure 3**).

Using the sample points $P_{[0 \dots 8]}$ compute the average pixel difference around pixel $P_{[0]}$ as,

$$F(x, y, r) = \text{abs} \sum_{i=1}^8 (P_{[0]} - P_{[i]}) \quad (17)$$

The average pixel difference (Eq. (17)) is identical to convolving the original image with the kernel of the **Figure 2**.

The features are extracted maximizing the rate of change of $F(x, y, r)$ respect to r , calculate as first order differencing,

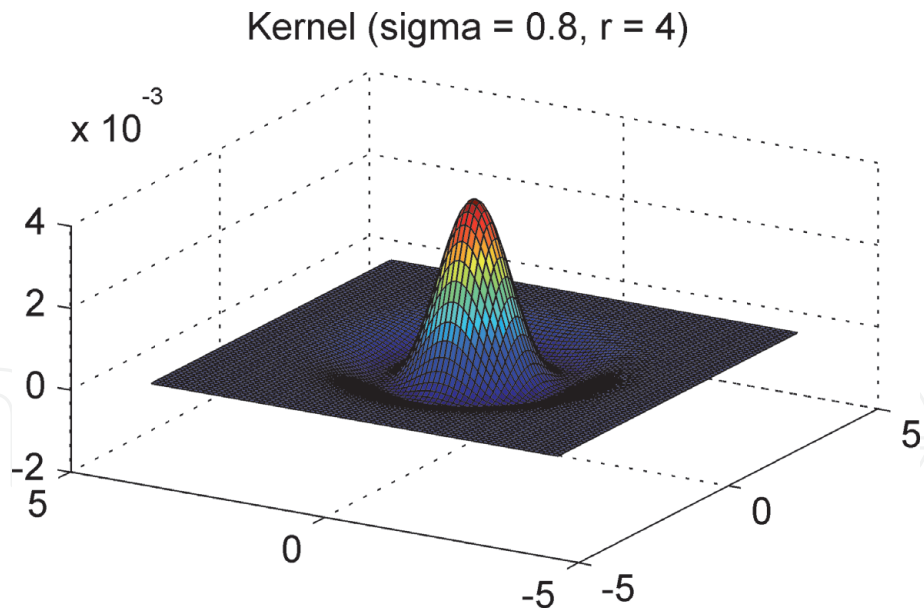


Figure 2.
Kernel of the Gaussian Laplacian filter.

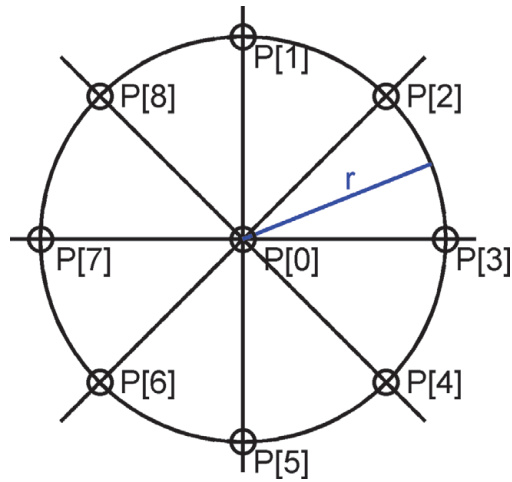


Figure 3.
Eight discrete directions of the sampled pixels.

$$R(x, y, r) = F(x, y, r) - F(x, y, r - 1) \tag{18}$$

if $R(x, y, r)$ is multiplied by the minimum pixel difference,

$$F_{\min}(x, y, r) = \min_i (P_{[0]} - P_{[i]}) \tag{19}$$

the pixels that no exhibit changes in intensity in all directions are suppressing.

2.2.2 Speeded-up robust features (SURF) algorithm

The particularity of this algorithm is its ability to determine the characteristics points in an image, which are invariant to changes in: scale, rotations and translations, and partially to illumination changes. This algorithm is an optimization of the Scale Invariant Feature Transform (SIFT) algorithm [12], being its execution speed much higher than the latter [13]. On the other hand, the SURF algorithm produces less information that the SIFT for each characteristic point of the image,

although the produced information by the SURF algorithm is more than enough for most applications, including the present one.

The use of integral images in this algorithm makes it very fast to represent at different scales of the original image its differential features. Integral images accelerate the computation at different scales the application of Haar wavelets, and together with the application of the Hessian differential operator allows the determination of key points and their robust descriptor features [2, 11].

Given an image I , and a point $X = (x, y)$ in this image, the Hessian matrix $H_{(X,\sigma)}$ in $X = (x, y)$ to the scale σ is defined as:

$$H_{(X,\sigma)} = \begin{bmatrix} L_{x,x,(X,\sigma)} & L_{y,x,(X,\sigma)} \\ L_{x,y,(X,\sigma)} & L_{y,y,(X,\sigma)} \end{bmatrix}, \quad (20)$$

where $L_{x,x,(X,\sigma)}$, $L_{x,y,(X,\sigma)}$, $L_{y,x,(X,\sigma)}$ and $L_{y,y,(X,\sigma)}$ represent the convolution product of the second derivative of the Gaussian $\frac{\partial^2}{\partial X^2} g_{(X,\sigma)}$ with the Image I in the point $X = (x, y)$ [12], see (Eq. (22)).

$$L_{x,x,(X,\sigma)} = \frac{\partial^2}{\partial X^2} * [G_{(x,y,\sigma)} * I_{(x,y)}] \quad (21)$$

$$L_{x,x,(X,\sigma)} = \left[\frac{\partial^2}{\partial X^2} * G_{(x,y,\sigma)} \right] * I_{(x,y)} \quad (22)$$

$$\wedge G_{(x,y,\sigma)} = \frac{1}{2\pi\sigma^2} e^{-\frac{x^2+y^2}{2\sigma^2}} \quad (23)$$

The determinant of the Hessian matrix allows the calculation of the scale of the point, defined as follows:

$$|H_{(X,\sigma)}| = D_{x,x}D_{y,y} - (\omega D_{x,y})^2, \quad (24)$$

where $D_{x,x}$, $D_{y,y}$, and $D_{x,y} = D_{y,x}$ are the approximations of the partial derivatives, and ω is the balance factor of the determinant [2], obtained from (Eq. (25)) where $|\cdot|_F$ is the Frobenius norm [3], see (Eq. (26)).

$$w = \frac{|L_{x,y,(X,\sigma)}|_F |D_{y,y}|_F}{|L_{y,y,(X,\sigma)}|_F |D_{x,y}|_F} \quad (25)$$

$$|A|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n (a_{i,j})^2} \quad (26)$$

Applying the Haar-Wavelet filters in a circular area of radius 6σ provides us a set of outputs in both directions (dx and dy respectively), and the mean value of those responses as a dominant direction within the sliding area of $\pi/3$ [12].

Finally the feature descriptors for a certain scale and for each characteristic point are obtained. To do this a rectangular area of $20\sigma \times 20\sigma$ centered on the point is constructed in the dominant orientation. This is divided into four sub-regions of 4×4 , and for each sub-region the Haar-Wavelet is applied obtaining the horizontal dx and vertical dy responses. A characteristic vector V is formed, and then for each point a total of 64 SURF descriptors are generated [10, 11],

$$V = \left(\sum d_x, \sum d_y, \sum |d_x|, \sum |d_y| \right). \quad (27)$$

2.2.3 Binary robust independent elementary features (BRIEF) algorithm

In terms of execution time, the SURF algorithm performs better than SIFT, but this is not sufficient for current applications for real-time processing of video streams in navigation, augmented reality, etc. To satisfy these applications simpler concepts are applied in algorithms for obtaining fast detectors and descriptors, such as: FAST [14], FASTER [15], CenSurE [16], and SUSurE [17] are some examples of them.

In particular the BRIEF descriptor, like SURF, uses the integral image and applies to a set of very simple binary tests which are adequate to the use of the Hamming distance (distance between two code words, is the number of bit positions in which they differ). This distance is much simpler and faster to evaluate than the Euclidean distance. It is also demonstrated in a practical way that a 32-dimensional BRIEF descriptor achieves similar results to a 64-dimensional SURF descriptor.

This algorithm obtains the descriptors of the characteristics points of the image, and for this a defining a neighboring area centered on this points, this area is known as patch p , which is square and a few pixels high and wide $L \times L$ (see **Figure 4**).

Since, the BRIEF algorithm handles pixel intensity levels this makes it very sensitive to noise, therefore it is necessary to pre-smooth the patch to reduce the sensitivity and increase the accuracy of the descriptors. Is for that, after create a patch centered on the feature point a smooth Gaussian filter is applied to the patch (Eq. (28)),

$$f_{(x,y)} = \frac{1}{2\pi\sigma^2} e^{\left(-\frac{x^2+y^2}{2\sigma^2}\right)}. \quad (28)$$

BRIEF converts the patches into a binary vector representative of it, this descriptor containing only 1 and 0, so each descriptor of a characteristic point is a string of 128–512 bits. After applied the smoothing to the patch p by (Eq. (16)) the patch is converted to binary feature vector as responses of binary test τ , which is define by (Eq. (29)),

$$\tau_{(p;x,y)} = \begin{cases} 1 : p_{(x)} < p_{(y)} \\ 0 : p_{(x)} \geq p_{(y)} \end{cases} \quad (29)$$

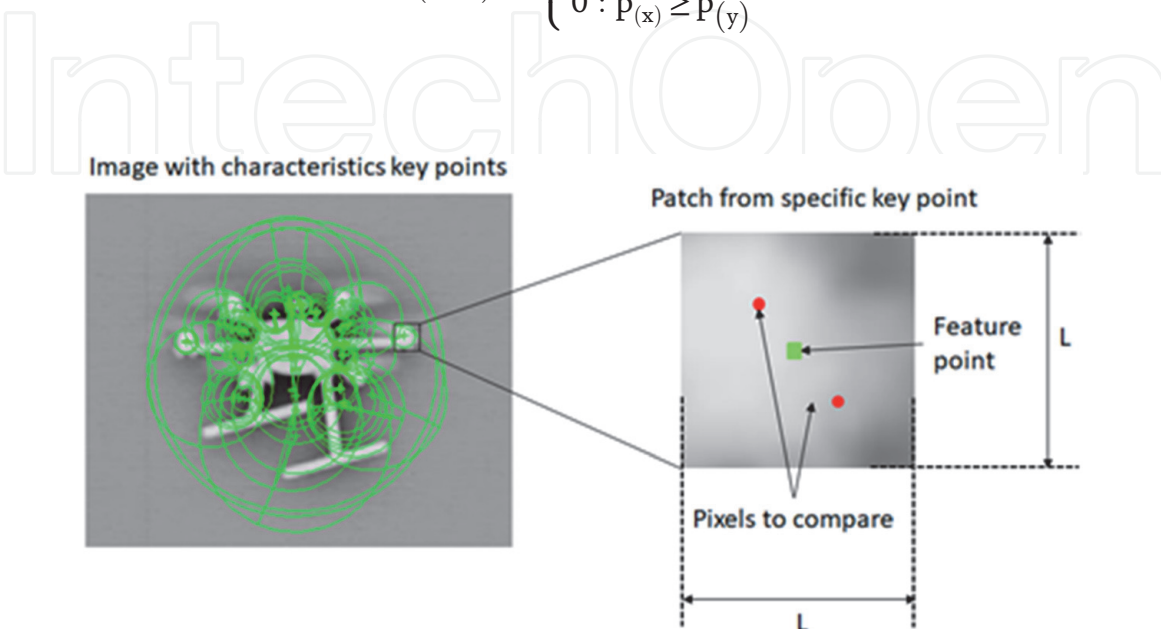


Figure 4.
 Patch in an image over a specific key point.

where $p_{(x)}$ is the intensity value of the pixel at point x ; then a set of $n(x, y)$ with n equal to 128, 256 or 512, and the location pairs (see **Figure 4**) must be defined as a set of binary tests uniquely. The pixel $p_{(x,y)}$ is located inside the patch, and it is called random pair, for creating a binary feature vector of number n is necessary select the random pairs; the most useful functions to this selection are the following five, showed in (Eq. (30)).

$$\left. \begin{array}{ll} \text{I.} & (X, Y) \sim \text{Uniform} \left(-\frac{L}{2}, +\frac{L}{2} \right) \\ \text{II.} & (X, Y) \sim \text{Gaussian} \left(0, +\frac{1}{25}L^2 \right) \\ \text{III.} & \begin{array}{l} X \sim \text{Gaussian} \left(0, +\frac{1}{25}L^2 \right) \\ Y \sim \text{Gaussian} \left(x_i, \frac{1}{100}L^2 \right) \end{array} \\ \text{IV.} & \text{The } (x_i, y_i) \text{ are randomly sampled} \\ \text{V.} & x_i = (0, 0)^T \text{ and } y_i \text{ is takes all} \\ & \text{possible values on a coarse polar grid} \end{array} \right\} \quad (30)$$

The advantages characteristics of the BRIEF descriptor are: high-speed processing, little memory usage and strong to illumination and blur change, and disadvantages are: weak to the rotation of the viewpoint, and the change in the position of a light source.

The (Eq. (31)) describes the BRIEF descriptor, but in its application must be in consideration that its matching performance falls sharply with a few degrees in plane rotation.

$$f_{n(p)} = \sum_{i=1}^n 2^{i-1} \cdot \tau(p; x_i, y_i) \quad (31)$$

2.2.4 Oriented FAST and rotate BRIEF (ORB) algorithm

The main characteristics of the ORB algorithm compared to its equivalents SIFT and SURF are that the ORB performs the feature detection operations as well as these but with a superior performance; having as an additional advantage that its use is free. It is based on the widely proven FAST and BRIEF algorithms, very good performance and low computational cost.

As the FAST algorithm does not have information about features of orientation and multi-scale, these must be implemented. For the multi-scale response, a scale pyramid is generated, where each level of the pyramid contains a version of the original image but at a lower resolution (**Figure 5**).

Applying to each level (scale) the FAST algorithm (see 2.1.1) we obtain the characteristics points at each level (vertices); as this algorithm also responds to edges, these must be discarded. For this we use the Harris algorithm with a low threshold [18], in order to obtain a large number of characteristic points, which are ordered according to the Harris measure to obtain the desired number of main points.

From there, the orientation is done by means of the intensity centroid [19], which assumes that the intensity of a corner is different from that of its center, and the vector formed between both points is used to calculate its orientation. Rosin defines the moments of a patch by means of (Eq. (32)) as,

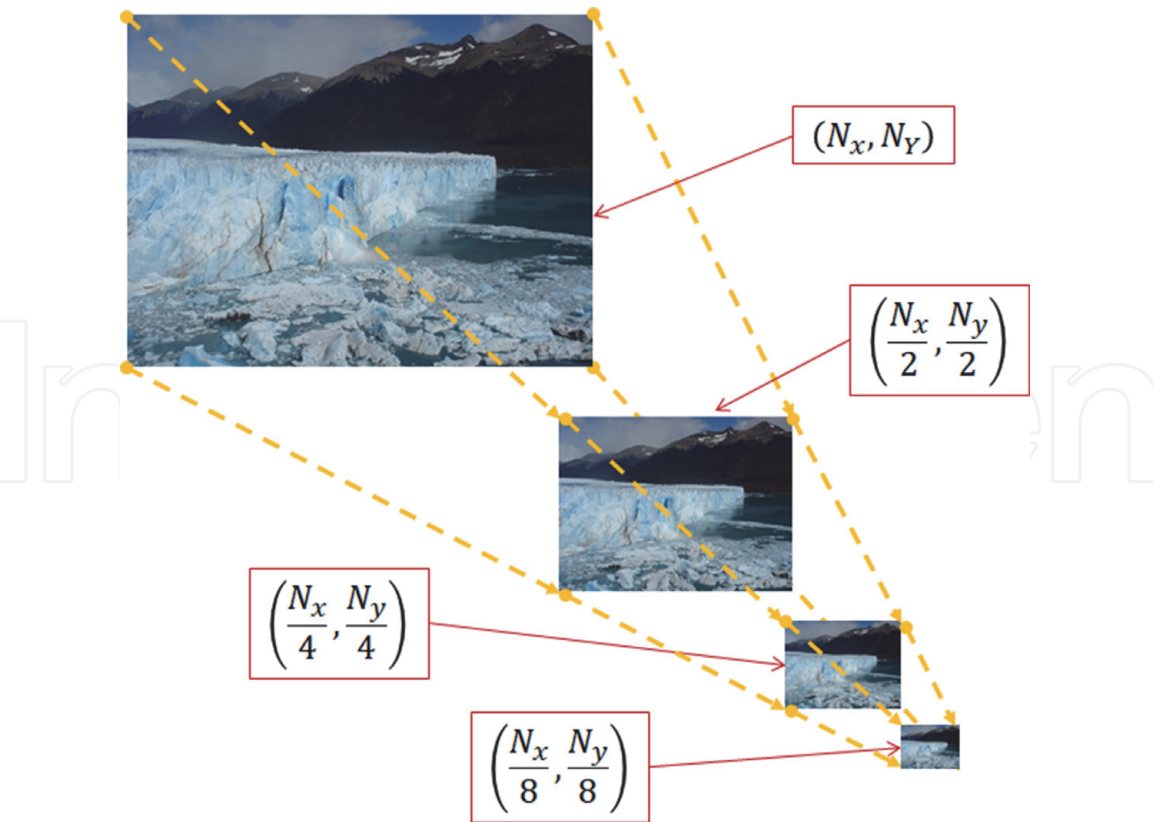


Figure 5.
Multiscale scheme for an image.

$$m_{p,q} = \sum_{x,y} x^p \cdot y^q \cdot I_{(x,y)} \tag{32}$$

and with these moments we can find the centroid by (Eq. (33)),

$$C = \left(\frac{m_{1,0}}{m_{0,0}}, \frac{m_{0,1}}{m_{0,0}} \right). \tag{33}$$

While the orientation of the patch can be calculated by means of the vector \overrightarrow{OC} with origin at the center C of the patch and a point O on its periphery, and is obtained by means of (Eq. (34)),

$$\theta = \text{atan2}(m_{0,1}, m_{1,0}). \tag{34}$$

To improve the invariance of this measurement we must ensure that the calculation of the moments is performed with x and y included in a circular region of radius r contained within the patch.

The table of features for each characteristics point is determined by means of the steered BRIEF method [20], according to the orientation θ of the patch at that point. For each characteristics point (x_i, y_i) we define a $2 \times n$ matrix S (Eq. (35)), and a rotation matrix R_θ as function of the orientation θ of the patch, obtaining an oriented version of the matrix S_θ by (Eq. (36)).

$$S = \begin{bmatrix} x_1 & \dots & x_n \\ y_1 & \dots & y_n \end{bmatrix} \tag{35}$$

$$S_\theta = R_\theta S \tag{36}$$

Obtaining the operator steered BRIEF as,

$$g_n(p, \theta) = f_n(p) | (x_i, y_i) \in S_\theta, \quad (37)$$

where $f_n(p)$ is the binary test operator obtained from (Eq. (31)).

To construct the BRIEF pattern search table, the angle θ is discretizing in increments of $2\pi/30$ (12 degrees). All previous tests are ordered by their distance from a mean of 0.5 generating a vector T . Then the first test of T is taken and added to the table of results R , from there the next test of the vector T is taken and compared with all the tests results in R , if its absolute correlation is less than a threshold it is discarded, otherwise it is added to the table of results R , until a total of 256 tests are obtained.

3. Processes in the particle estimator algorithm

Basically this method deploys in the image a series of random particles, possible states of the process in this case the target position and its size; while their weights represent their posteriori probability of the density functions as an estimated from the observations. One of the particularities of the particle estimator is the number of configuration parameters it has, and to optimize its performance, in terms of estimation quality and processing time, they must be properly chosen. Some of these parameters refer to the behavior of the estimator itself and others to the behavior of the SURF algorithm (used here for infrared images), as described below.

For the behavior of the particle estimator:

- Maximum number of particles,
- Probability threshold to consider a particle valid,
- Adaptability in the standard deviation as a function of the number of valid particles for the vector of states in the resampling.

For the behavior of the SURF algorithm:

- Maximum number of characteristics points to be considered,
- Number of scale level, controls the number of filters used per octave,
- The threshold for the determination of the number of blobs to detect,
- Number of octaves, controls the filters and subsample of the image data; larger number of octaves will result in finding larger size blobs.

Then, and for the case of this application (tracking objects in video images) we can describe the processes implemented following the guidelines of the algorithm described in 2.1.

3.1 Initialization

Mainly in this step, the reference particle to follow is selected, and all the parameters described in the previous paragraph are initialized with the design values.

3.2 Particle description

The particle was described by means of a rectangle; and to characterize it we define a state vector for each particle (Eq. (38)).

$$X = \begin{bmatrix} x & y & w & h & \dot{x} & \dot{y} & \dot{w} & \dot{h} \end{bmatrix}^T = [p \ v]^T \quad (38)$$

This state vector contains the positions (x, y) and for its size, lengths of width and height (w, h) , while for its velocities (\dot{x}, \dot{y}) and (\dot{w}, \dot{h}) respectively.

The velocity of the particle is calculated by means of the difference between $(p_{k+1} - p_k)$ divided by the time video sampling, at the end of the cycle.

When a new frame is acquired, the particles are propagated following state evolution model in correspondence with (Eq. (1)):

$$P_k = AP_{k-1} + Av_{k-1}, \quad (39)$$

$$A = \begin{bmatrix} I_{4 \times 4} & \Delta t \cdot I_{4 \times 4} \\ 0_{4 \times 4} & I_{4 \times 4} \end{bmatrix}, \quad (40)$$

where $I_{4 \times 4}$ is the 4x4 identity matrix, $0_{4 \times 4}$ is the 4x4 null matrix and Δt is the frame sample time. After the particle propagation the bounds check is applied to verify that all the particles are within the image.

3.3 Particle probabilities estimation

The determination of the similarity between each particle and the reference depends on the type of image, either color or black and white. The images in this text are in black and white from images acquired from infrared sensors, but we will pause here to consider a color image.

3.3.1 Color images

In the case of the color image to determine the similarity with the reference the distance between histograms is used, in the example of the **Figure 6** the Bhattacharyya distance [21] for histograms (Eq. (41)) was used, instead of Euclidean distance, because a better response was obtained.

$$B_{dst} = \frac{\sum (hIr - \overline{hIr}) \cdot (hI - \overline{hI})}{\sqrt{\left(\sum (hIr - \overline{hIr})^2 \right) \cdot \left(\sum (hI - \overline{hI})^2 \right)}} \quad (41)$$

Where B_{dst} is the Bhattacharyya distance, hIr and hI the histogram to be compared, and \overline{hIr} and \overline{hI} its mean values. The probability of one is obtained when the sum of the three distances between the red, green and blue histograms, is equals to zero.

Figure 7 shows the sequence of images for the tracking of a person, in which the particles can be seen in three colors, green for the best estimation, blue for the most probable particles, and red for the least probable ones. At 3.399 seconds the action of the increase dispersion parameter in the resampling procedure is observed, due a low number of valid particles; this is maintained only in two frames, returned to normal values quickly.

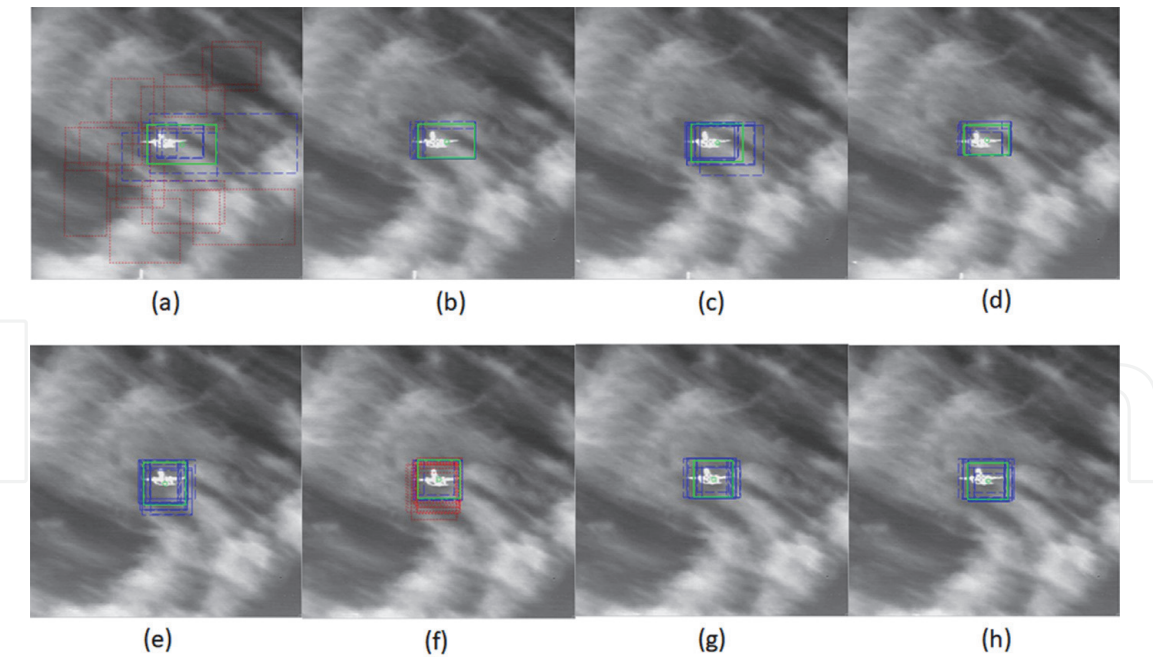


Figure 6.
Airplane sequence of the particle estimator, where [a .. h] correspond to sequences between [4s .. 4.32s].

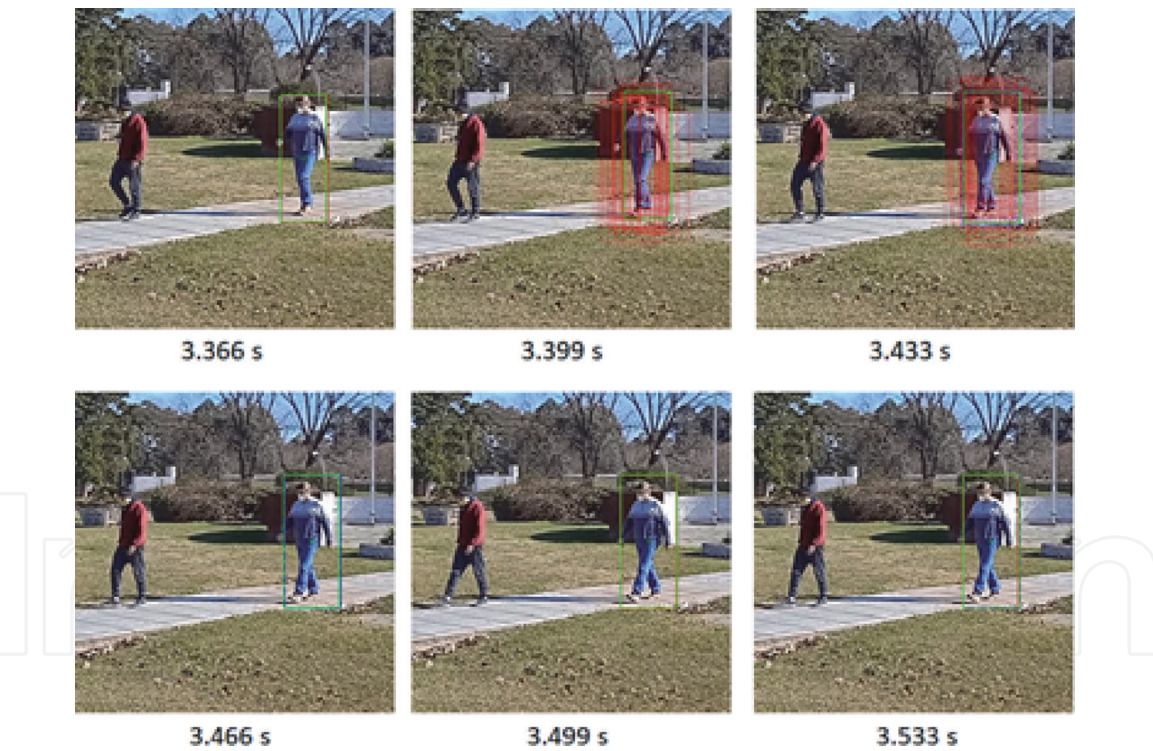


Figure 7.
Person tracking, sequence of the particle estimator.

3.3.2 Black and white images

In the case of the black and white images, for determining the similarity between each particle with the reference, the SURF algorithm was used. Before applying the SURF algorithm, the background effect is reduced, generally this improvement is based on subtracting the value from the mean intensity and modifying its histogram to increase the contrast result a good choice for infrared images. In this case and for this type of image the procedure used to enhance the image of the object was: first

subtract the input image I_P by a proportional factor α to its mean value I , second multiply the result by a proportional factor β to the relationship between the standard deviation of the image (as measurement of the contrast) divided by the maximum value of the luminosity (Eq. (42)).

$$I_P = \left[I - \alpha \sum_{i=1, j=1}^{N, M} I_{(i,j)} \right] \beta \frac{\sigma(I)}{\max(I)}, \quad (42)$$

where, N and M are the number of pixels X and Y respectively, $\sigma(I)$ and $\max(I)$ the standard deviation and the maximum value of image respectively. The adaptive factors α and β are obtained experimentally and the best results was obtained with values of $\alpha = 0.84314$ and $\beta = 25$.

For determining the similarity of a particle with the reference particle, a pairing of the characteristic points is performed. After this pairing, the numbers of pairs of points whose metric is less than a threshold metric are counted; and the relationship between the numbers of pairs of points with respect to the total number of characteristic points of the reference particle gives us its similarity probability.

3.4 Position and size of the object to be tracked

The position and size of the object to be tracked is determined by means of the most significant particles, i.e. those particles whose probability exceeds the selected probability threshold.

3.5 Re-sampling

From the most probable particle obtained in the previous step, a new particle set is distributed according to the dispersion of each variable, always limiting them to the size of the screen. But the dispersion of the variables is adapted according to the number of valid particles. This is implemented as a strategy of re-sampling, where the search zone is extended as function of the number of valid particles decreases.

4. Experimental results

Three experimental tests were performed with the previously described algorithm to tracking an object: a) an airplane; b) a six rotors "UAV" and c) a heliport. For the three experiments an infrared camera Tau 640, in the 8 to 14 micron band was used, in a) and b) experiments the camera was mounted on a positioning system, while in c) the camera was mounted on a six-rotor "UAV", taking a zenithal image.

In case b), where the camera was mounted on a positioning system, this had a tracking system attached to it, so the values of the X and Y coordinates in pixels of the target could be obtained. Comparing them with those obtained from the particle estimator the error of the latter was determined.

The images obtained from the video frames were recorded sequentially according to the video acquisition frequency, and before applying the particle estimation algorithm, they were pre-processed by a background suppression and contrast enhancement procedure (Eq. (42)).

In all cases the same observation function, the SURF routine, was used to determine the likelihood of similarity between the video image and the reference image. The dispersion factors applied to the state vector in the resampling function

vary inversely proportional to the number of valid particles, so that if the number of valid particles decreases, the search area increases.

4.1 Airplane tracking

The first experiment was the tracking of airplane, the estimator sequence can be observed in **Figure 6**, frame (a) correspond to the first sate of the estimator, distributed the particles over an area of the maximum probability of locate the target.

After the correct detection the particle estimator remains locked throughout the entire flight. As can be observed in frame (f) a lot of particle with probability under the threshold probability appears, but nevertheless remains lock.

4.2 Six rotors tracking

This experiment has the particularity of that the UAV to avoid detection flies hidden behind the trees, after which it starts a free flight. And in both cases the particle estimator is able to detect when only one part of its image visible. From the moment of the first detection the particle estimator remains locked throughout the entire flight. The **Figure 8** shows part of the frames sequence of the flight, from left to right and from top to bottom the “UAV” first flying behind the trees, while the particle estimator begins distributing the particles in the largest possible area of the screen, after the third frame, and with the “UAV” still largely behind the trees, the particle estimator still able to detect it and “hook” it as soon as it partially appears.

From the fourth frame the estimator remains locked during the whole flight, especially when the occlusion free flight is carried out, and the tracking is always right.

4.3 Heliport tracking

This experiment was a flight over a heliport located in a park area of the Institute, and the shots were taken from zenithal videos because an artificial vision navigation system was tested to aid the landing. The characteristics of these images

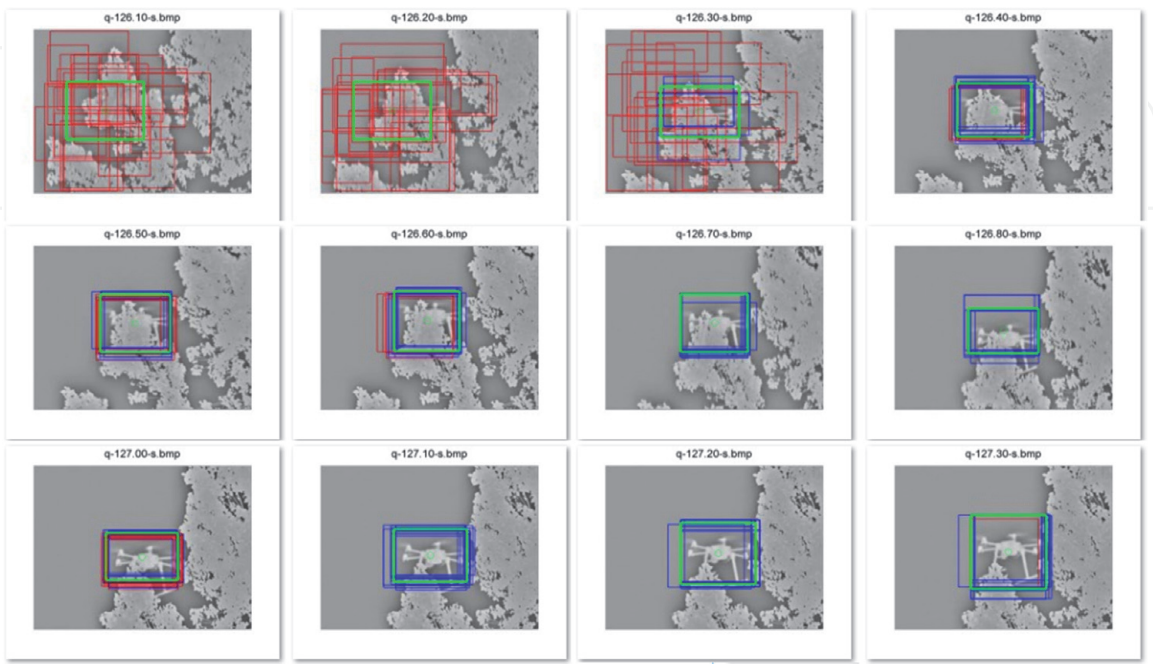


Figure 8.
Six rotors “UAV” sequence of the particle estimator.

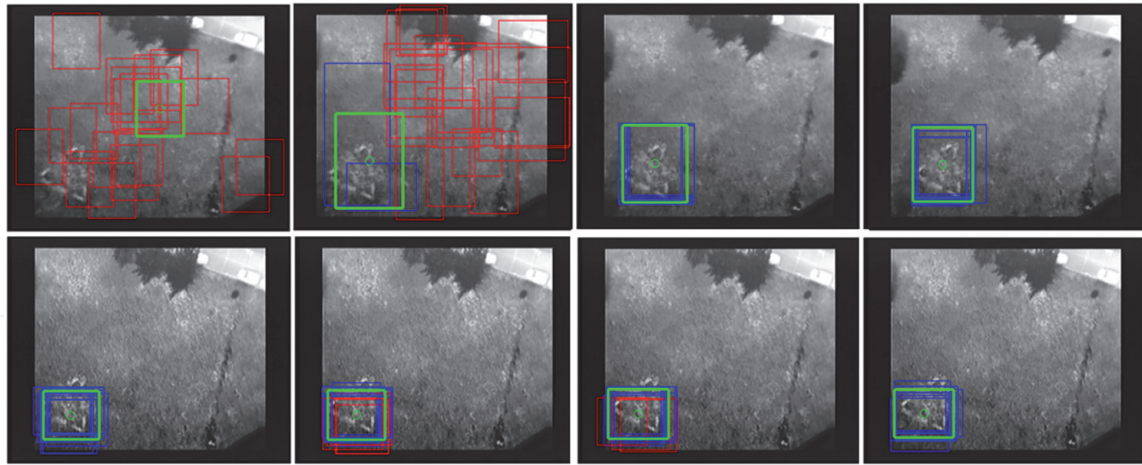


Figure 9.
Helicopter tracking sequence of the particle estimator.

are that they are interfered by infrared radiation from the ground, and the helicopter is not properly marked.

The **Figure 9** shows two sequences of the particle estimator execution, the upper correspond to the startup, and the lower in the middle of the experiment. From left to right and from top to bottom, the first frame corresponds to the initialization of the estimator, the particles are distributed over the whole image and the most probable particle in the center of the screen. In the second frame after 0.04 seconds (frame rate) only two particles have probability higher than the threshold, and it can be seen how the most probable particle reached the target. After that the estimator is locked, and remains in sequence of tracking.

5. Conclusions

The parameters that define the behavior of the particle estimator were described in paragraph 3. Although an analysis of the estimator's response can be made based on each of these parameters; here we will perform only one, for the number of particles of the estimator.

For this purpose, we rely on the experiment corresponding to the tracking of the six-rotor UAV, because it was simultaneously tracked by a positioning system. Therefore, taking as true the observation obtained from the positioning system, it was possible to determine the relative error corresponding to the particle estimator.

To observe the dynamic response and the internal state of the particle estimator, the relative error (upper graph) and the number of valid particles (lower graph) as a function of time were plotted on **Figure 10**.

This figure show the behavior of the estimator, as a function of the number of particles and in different colors, for values of 30 (red), 50 (magenta), 70 (blue) and 90 (green), the time between marks is the acquisition video time, which is 0.050 seconds.

As can be seen always at least a number of 6 frames are needed to begin reach the lock state, and the number of particles parameter does not affect this number to get this state. But when the number of the particles of the estimator it increase, decreases its relative error, increase the computer time process, and presents greater stability to maintain the lock.

Another parameter that improved the particle estimator response is the dispersion multiplying factor of the state variables. This multiplying factor is increased when the number of valid particles decrease, covering a wider area of search in the

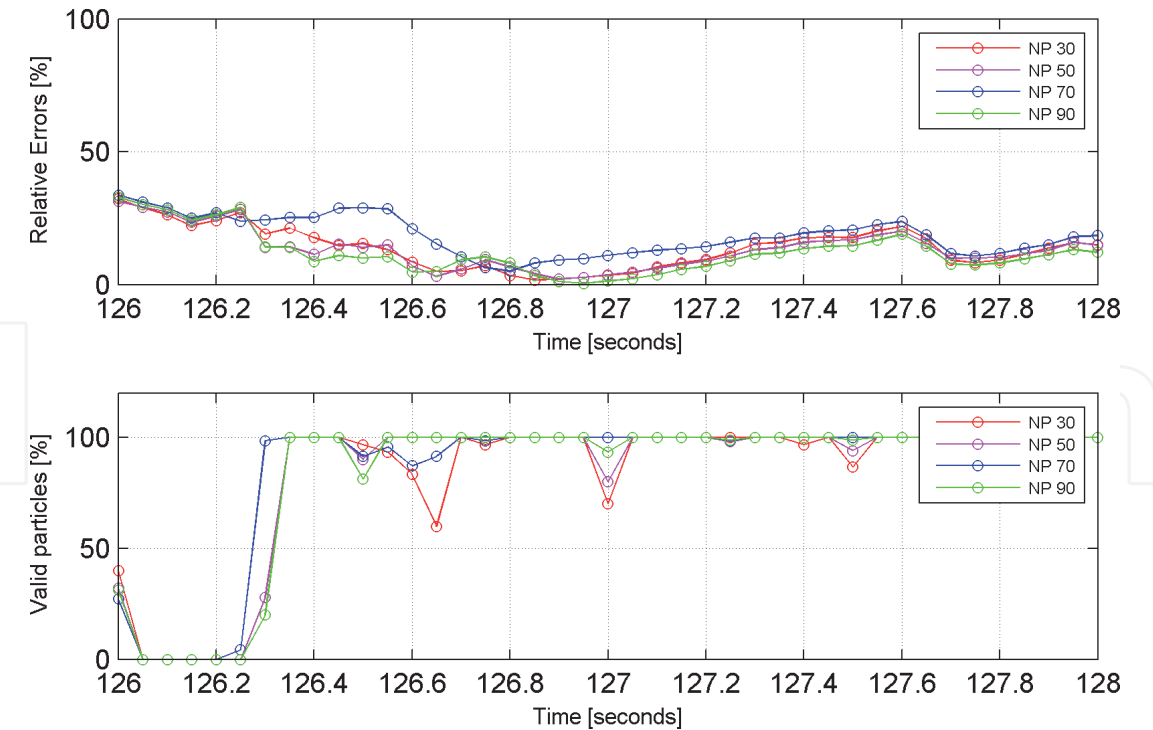


Figure 10.
Dynamic response of the particle estimator.

image. With this search strategy the estimator maintains its stability in the response to the lock with the objective, in a few frames.

After several execution of the algorithm in the experiment b); and varying different initialization parameters of the estimator, the values that produce an acceptable behavior are: number of particles 70 or higher, number of characteristic points of the SURF algorithm 70 or higher, threshold value of probability of similarity 90% or higher.

With the considerations in the initialization parameters of the particle estimator mentioned above, the state lock could be maintained with no more than two frames unlocked. And another observed feature is that the estimator could track the target even when the object is mostly occluded on its surface, for example when the UAV is behind trees (see **Figure 8**).

As a proposed improvement to enhancement to achieve better detection of the target when it is mainly occluded, or when the image is heavily contaminated with noise; it is to reformulate the main strategy. This strategy consists of decomposing the reference image into $N \times N$ sub-images, in sequence and with their corresponding identification. And for each sub-image a particle estimator is applied, having $N \times N$ particle estimators, each one looking for a part of the reference image. From there and starting with the first particle, we look for the particles with more probability, and that are located in the correct sequence; the average of these particles will give us the more likely position of the target to be follow. In case that no particles are found in the correct positions we can obtain the most probability position of the object to be followed as the position of the particle with the largest probability, or as the integration of the information provided by the $N \times N$ estimators.

Acknowledgements

Special thanks to the Post Graduate staff of the National Technological University – Buenos Aires Regional Faculty. And the staff of the Laboratories of the

Institute of Scientific and Technical Research for Defense: Thermal Imaging Laboratory for providing the images of the experience, and to the Digital Techniques Laboratory for carrying out the flight with the unmanned aerial vehicle.

Conflict of interest

The authors declare no conflict of interest.

Author details


Edgardo Comas^{1,2*} and Adrián Stácul^{1,2}

1 Institute of Scientific and Technical Research for Defense, Buenos Aires, Argentina

2 National Technological University – Buenos Aires Regional Faculty, Buenos Aires, Argentina

*Address all correspondence to: ecomas@frba.utn.edu.ar

IntechOpen

© 2021 The Author(s). Licensee IntechOpen. This chapter is distributed under the terms of the Creative Commons Attribution License (<http://creativecommons.org/licenses/by/3.0>), which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited. 

References

- [1] Maybeck, Peter, Stochastic Models, Estimation and Control, vol. 3. Academic Press, Inc, 1982.
- [2] Frank L. Lewis, LIHUA XIE, and Dan Popa, Optimal and Robust Estimation With an Introduction to Stochastic Control Theory, Second. 2008.
- [3] David P. Landau, Kurt Binder, A Guide to Monte Carlo Simulations in Statistical Physics, Third Edition. Cambridge University Press, 2009.
- [4] Jari Kaipio, Erkki Somersalo, Statistical and Computational Inverse Problems, Vol. 160. Springer Science, 2004.
- [5] Arnaud Doucet and Adam M. Johansen, 'A tutorial on particle filtering and smoothing', Handb. Nonlinear Filter., vol. 12, Mar. 2012.
- [6] Arnaud Doucet, Simon Godsill and Christophe Andrieu, 'On sequential Monte Carlo sampling methods for Bayesian filtering', Stat. Comput., Jul. 2000, DOI:10.1023/A:1008935410038.
- [7] Liu Jun and Chen Rong, 'Sequential Monte Carlo Methods for Dynamic Systems', J. Am. Stat. Assoc., Apr. 1998, DOI:10.1080/01621459.1998.10473765.
- [8] Fredrik Gustafsson, 'Particle filter theory and practice with positioning applications', IEEE Aerosp. Electron. Syst. Mag., vol. 25, no. 7, Jul. 2010, DOI: 10.1109/MAES.2010.5546308.
- [9] Miroslav Trajković, Mark Hedley, 'Fast corner detection', Image Vis. Comput.
- [10] Edward Rosten and Tom Drummond, 'Machine Learning for High-Speed Corner Detection'. DOI: 10.1007/11744023_34.
- [11] Charles Bibby and Ian Reid, 'Fast Feature Detection with a Graphics Processing Unit Implementation'. University of Oxford, 2006.
- [12] Herbert Bay, Tinne Tuytelaars, and Luc Van Gool, 'speeded up robust features', Comput. Vis. Image Underst., vol. 110, no. 3, Jun. 2008, DOI:10.1016/j.cviu.2007.09.014.
- [13] H. Kandil and A. Atwan, 'A comparative study between SIFT-particle and SURF-particle', Int. J. Signal Process. Pattern Recognit., Vol. 5, no. No 3, Sep. 2012.
- [14] Edward Rosten and Tom Drummond, 'Fusing points and lines for high performance tracking', in IEEE International Conference on Computer Vision, Oct. 2005, vol. Volume 2.
- [15] Edward Rosten, Reid Porter, and Tom Drummond, 'A Machine Learning Approach to Corner Detection', IEEE Trans Pattern Anal. Mach. Intell., 2010.
- [16] Motilal Agrawal, Kurt Konolige, and Morten Rufus Blas, 'CenSurE: Center Surround Extremas for Realtime Feature Detection and Matching', Oct. 2008. DOI:10.1007/978-3-540-88693-8_8.
- [17] Mosalam Ebrahimi and Walterio Mayol-Cuevas, 'Speeded Up Surround Extrema Feature Detector and Descriptor for Realtime Applications', Jun. 2009.
- [18] C. Harris and M. Stephens, 'A Combined Corner and Edge Detector', Presented at the Alvey Vision Conference, 1988. DOI:10.5244/C.2.23.
- [19] Paul L. Rosin, 'measuring corner properties', Comput. Vis. Image Underst., 1999, DOI:10.1006/cviu.1998.0719.
- [20] Vincent Lepetit, Pascal Fua and Christoph Strecha, 'Binary Robust Independent Elementary Features',

CVLab, EPFL, Lausanne, Switzerland,
May 2014. DOI:10.1007/978-3-
642-15561-1_56.

[21] Briñez de León, Juan C Retrepo
Martinez, Alejandro López Giraldo,
Francisco E., 'Métricas de Similitud
Aplicadas para Análisis de Imágenes de
Fotoelasticidad', Dyna, 2013 [Online].
Available: [https://www.redalyc.org/
articulo.oa?id=49627363006](https://www.redalyc.org/articulo.oa?id=49627363006)