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

4,800
Open access books available

122,000
International authors and editors

135M
Downloads

Our authors are among the

154
Countries delivered to

TOP 1%
most cited scientists

12.2%
Contributors from top 500 universities

CLARIVATE ANALYTICS BOOK CITATION INDEX INDEXED

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

7

Fuzzy Control System to Solve Coupling Between Tracking and Predictive Algorithms

Eduardo Parrilla, Jaime Riera, Juan R. Torregrosa and José L. Hueso
Universidad Politécnica de Valencia
Spain

1. Introduction

In this work, we analyze different topics in the study of object tracking. In Section 2, we study algorithms for tracking objects in a video sequence, based on the selection of landmark points representative of the moving objects in the first frame of the sequence to be analyzed. The movement of these points is estimated using a sparse optical-flow method. Methods of this kind are fast, but they are not very robust. Particularly, they are not able to handle the occlusion of the moving objects in the video. To improve the performance of optical flow-based methods, we propose the use of adaptive filters to predict the expected instantaneous velocities of the objects, using the predicted velocities as indicators of the performance of the tracking algorithm. The efficiency of this strategy to handle occlusion problems are tested with a set of synthetic and real video sequences. In (Parrilla et al., 2008) we develop a similar study using neural networks.

Video tracking deals with the problem of following moving objects across a video sequence (Trucco & Plakas, 2006), and it has many applications as, for example, traffic monitoring and control (Iñigo, 1989), (Erdem et al., 2003), robotic tasks, surveillance, etc. Simple algorithms for video tracking are based on the selection of regions of interest in the first frame of a video sequence, which are associated with moving objects to be followed and a system for estimating the movement of these regions across the sequence. More demanding methods impose constrains on the shape of the tracked objects (Erdem et al., 2003; Shin et al., 2005), or use methods to separate the moving objects from the background of the images (Ji et al, 2006; Wren et al., 1997). Most of these more demanding algorithms deal with partial occlusion of the moving objects, that is, the algorithms are not lost when the target temporarily disappears from the frame and they resume correctly when the target reappears. Generally, this kind of algorithms include an a priori training of the possible shape of the object to be followed and occlusion is handled following the movement of the contour. This is expensive from the computational point of view and makes these algorithms difficult to be implemented in a real-time application.

We have used adaptive filters (Haykin, 1986; Ljung, 1999) to predict the velocities of the object to track. These expected velocities are compared with the ones computed with the optical flow method and are used as indicators of the performance of the method. If the optical flow method fails to compute reasonable values for the velocities, the velocity values predicted by the filter can be used as reliable values for the velocities of the object. A system of this kind

can be useful in different applications, such as economic systems for traffic monitoring and control using images provided by cameras installed in the roads (Iñigo, 1989).
The critical point of the system for solving the occlusion problems is the coupling between optical flow and predictive algorithms. This coupling is governed by different parameters. In Section 3, we propose the use of a fuzzy control system to solve the mentioned coupling.
Section 4 is devoted to stereoscopic video sequences. We analyze simple algorithms for 3D tracking objects in a stereo video sequence, by combining optical flow and stereo vision. This method is not able to handle the occlusion of the moving objects when they disappear due to an obstacle. To improve the performance of this method, we propose the use of adaptive filters or fuzzy control techniques to predict the expected instantaneous velocities of the objects.

2. Handling occlusion in optical flow algorithms for object tracking

2.1 Optical flow

For computing the optical flow of a frame of a video sequence, it is necessary to assume that intensity variations of the image are only caused by displacements of the objects, without considering other causes such as changes of illumination. This hypothesis, proposed initially by Horn and Schunck (Horn & Schunck, 1981), supposes that present intensity structures in the image, at local level, stay constant throughout the time, at least during small temporal intervals. Formally, if $I(\bar{x},t) = I(x(t),y(t),t)$ denotes the continuous space-time intensity function of an image, it has to be fulfilled that

$$I(x(t),y(t),t) \approx I(x(t+\Delta t),y(t+\Delta t),t+\Delta t). \tag{1}$$

By expanding the right-hand side of equation (1) using the Taylor series yields

$$I(t) \approx I(t) + \frac{dI(t)}{dt}\Delta t + O^2(\Delta t) \tag{2}$$

that is,

$$\frac{dI}{dt} = 0, \tag{3}$$

where $O^2(\Delta t)$ are the 2^{nd} and higher order terms, which are assumed to be negligible. Finally, applying the Chain rule, we obtain the optical flow equation

$$\frac{\partial I}{\partial x}\frac{dx}{dt} + \frac{\partial I}{\partial y}\frac{dy}{dt} + \frac{\partial I}{\partial t} = 0 \tag{4}$$

or, in another way

$$I_x u + I_y v + I_t = 0, \tag{5}$$

where I_x, I_y are the components of the spatial gradient ∇I, I_t denotes partial differentiation with respect to time and $\mathbf{v} = (u,v)$ denotes the components of the image velocity field. The distribution of the velocity in each point of the image is known as the optical flow.
Constraint (5) is not sufficient to solve the optical flow equation for a given frame. The problem is ill-posed because we have to compute two components, u and v, and we only have one equation. This phenomenon is known as the aperture problem. It is necessary to consider additional restrictions to the problem to estimate the motion at every image location. There are many techniques for computing the optical flow, which differ one from each other in the different assumptions that are taken into account to determine the solution. In our analysis we use the technique proposed by Lucas and Kanade in 1981 (Lukas & Kanade, 1981).

2.1.1 Lucas and Kanade algorithm

The algorithm proposed by Lucas and Kanade uses a local constant model of velocity, obtaining good results in different applications (Bourel et al., 2001; Parrilla et al., 2005;?). It is based on supposing the values of optical flow in a local neighborhood, R, to be constant, minimizing the following expression

$$\sum_{(x,y)\in R} W^2(x,y)(\nabla I(x,y,t)\mathbf{v} + I_t(x,y,t))^2, \tag{6}$$

where $W(x,y)$ denotes a window function and R is a spatial neighborhood. Equation (6) has the following solution (Barron et al., 1994)

$$A^T W^2 A \cdot \mathbf{v} = A^T W^2 b, \tag{7}$$

where, for n points (x_i, y_i) in R at a moment t,

$$A = \left[\nabla I(x_1,y_1), \cdots, \nabla I(x_n,y_n)\right]^T \tag{8}$$

$$W = diag\left[W(x_1,y_1), \cdots, W(x_n,y_n)\right] \tag{9}$$

$$b = -\left[I_t(x_1,y_1), \cdots, I_t(x_n,y_n)\right]^T \tag{10}$$

We can track a window from frame to frame if the system (7) represents good measurements, and if it can be solved reliably. This means that the 2×2 coefficient matrix $A^T W^2 A$ of the system must be both above the image noise level and well conditioned. In turn, the noise requirement implies that both eigenvalues of $A^T W^2 A$ must be large, while the conditioning requirement means that they cannot differ by several orders of magnitude. In practice, if the two eigenvalues of $A^T W^2 A$ are λ_1 and λ_2, we accept a window if $min(\lambda_1, \lambda_2) > \lambda$, where λ is a predefined threshold.

2.2 Adaptive filters

To obtain an indicator of the performance of the tracking algorithm, each component of the velocity estimated by using optical flow, for the different frames of a sequence, is considered as a time series. We use adaptive filters to predict instantaneous velocities in order to compare them with the velocities obtained by using the optical flow algorithm.
An adaptive filter is a filter which self-adjusts its transfer function according to an optimizing algorithm. Because of the complexity of the optimizing algorithms, most adaptive filters are digital filters that perform digital signal processing and adapt their performance based on the input signal. In our work, we concentrate on the recursive least squares (RLS) filter (Ljung, 1999).

2.2.1 RLS filter

We start from a signal x_n, $n = 0, \ldots, N_T$, and we assume for the signal an AR(p) model

$$x_k = -(a_1)_k\, x_{k-1} - (a_2)_k\, x_{k-2} - \cdots - (a_p)_k\, x_{k-p} + \varepsilon_k\,. \tag{11}$$

Assuming this model for $k = p, \ldots, N$, we obtain the system of equations

$$X_N = C_N a_N + \epsilon_N\,, \tag{12}$$

where

$$X_N = \begin{bmatrix} x_p \\ x_{p+1} \\ \vdots \\ x_N \end{bmatrix}, \; C_N = \begin{bmatrix} x_{p-1} & x_{p-2} & \cdots & x_0 \\ x_p & x_{p-1} & \cdots & x_1 \\ \vdots & & & \vdots \\ x_{N-1} & x_{N-2} & \cdots & x_{N-p} \end{bmatrix},$$

$$a_N = \begin{bmatrix} -(a_1)_N \\ -(a_2)_N \\ \vdots \\ -(a_p)_N \end{bmatrix}, \; \epsilon_N = \begin{bmatrix} \varepsilon_p \\ \varepsilon_{p+1} \\ \vdots \\ \varepsilon_N \end{bmatrix}.$$

The model coefficients, $\left(a_j\right)_N$, are obtained as those coefficients that make minimum the error function

$$\begin{aligned} J_N &= \frac{1}{2}\sum_{k=p}^{N} \lambda^{N-k}\varepsilon_k\varepsilon_k = \\ &= \frac{1}{2}\epsilon_N^T\Lambda_N\epsilon_N = \frac{1}{2}\left(X_N - C_N a_N\right)^T \Lambda_N \left(X_N - C_N a_N\right) . \end{aligned} \tag{13}$$

We have introduced the weights matrix

$$\Lambda_N = \begin{bmatrix} \lambda^{N-p} & 0 & \cdots 0 \\ 0 & \lambda^{N-p-1} & \vdots \\ \vdots & \vdots & \\ 0 & \cdots & 1 \end{bmatrix},$$

where $0 < \lambda \leq 1$ is a constant that controls the importance on the error function J_N of the older samples of the signal, and is called the forgetting factor of the method. The coefficients a_N are the solutions of the equation

$$\frac{\partial J_N}{\partial a_N} = 0 ,$$

that is,

$$a_N = \left[C_N^T\Lambda_N C_N\right]^{-1} C_N^T\Lambda_N X_N . \tag{14}$$

Let us consider now $N+1$ samples of the signal. The system (12) becomes

$$\begin{bmatrix} X_N \\ x_{N+1} \end{bmatrix} = \begin{bmatrix} C_N \\ c_{N+1}^T \end{bmatrix} a_{N+1} + \epsilon_{N+1} , \tag{15}$$

where

$$c_{N+1}^T = \left[x_N, x_{N-1}, \cdots, x_{N-p+1}\right] .$$

a_{N+1} will be the coefficients that minimize the new error function

$$J_{N+1} = \frac{1}{2}\sum_{k=p}^{N+1} \lambda^{N+1-k}\varepsilon_n\varepsilon_N = \frac{1}{2}\epsilon_{N+1}^T\Lambda_{N+1}\epsilon_{N+1} .$$

The solution of this problem is

$$a_{N+1} = \left[C_{N+1}^T \Lambda_{N+1} C_{N+1}\right]^{-1} C_{N+1}^T \Lambda_{N+1} X_{N+1} \,. \tag{16}$$

To obtain the inverse of matrix $C_{N+1}^T \Lambda_{N+1} C_{N+1}$ is an expensive process from the computational point of view, especially when a large number of samples are considered. To overcome this problem, we take into account that

$$\begin{aligned} C_{N+1}^T \Lambda_{N+1} C_{N+1} &= \begin{bmatrix} C_N^T & c_{N+1} \end{bmatrix} \begin{bmatrix} \lambda\Lambda_N & 0 \\ 0 & 1 \end{bmatrix} \begin{bmatrix} C_N \\ c_{N+1}^T \end{bmatrix} = \\ &= C_N^T \lambda \Lambda_N C_N + c_{N+1} c_{N+1}^T \,. \end{aligned}$$

By using the inversion lemma (Ogata, 1987)

$$(A + BD)^{-1} = A^{-1} - A^{-1} B \left(I + DA^{-1}B\right)^{-1} DA^{-1} \,,$$

with

$$A = \lambda C_N^T \Lambda_N C_N \,, \;\; B = c_{N+1} \,, \;\; D = c_{N+1}^T \,,$$

we rewrite

$$\begin{aligned} &\left[C_{N+1}^T \Lambda_{N+1} C_{N+1}\right]^{-1} = \\ &\frac{1}{\lambda}\left(\left[C_N^T \Lambda_N C_N\right]^{-1} - \frac{\left[C_N^T \Lambda_N C_N\right]^{-1} c_{N+1} c_{N+1}^T \left[C_N^T \Lambda_N C_N\right]^{-1}}{\lambda + c_{N+1}^T \left[C_N^T \Lambda_N C_N\right]^{-1} c_{N+1}}\right) \,. \end{aligned} \tag{17}$$

Introducing the notation

$$P_N = \left[C_N^T \Lambda C_N\right]^{-1} \;\; , \;\; K_{N+1} = \frac{P_N c_{N+1}}{\lambda + c_{N+1}^T P_N c_{N+1}} \,,$$

we obtain the recurrence

$$P_{N+1} = \frac{1}{\lambda}\left(P_N - K_{N+1} c_{N+1}^T P_N\right) \,. \tag{18}$$

Therefore, using (16), we can write

$$\begin{aligned} a_{N+1} &= P_{N+1} \lambda C_N^T \Lambda_N X_N + P_{N+1} c_{N+1} x_{N+1} \\ &= P_N C_N^T \Lambda_N X_N - K_{N+1} c_{N+1}^T P_N C_N^T \Lambda_N X_N \\ &\quad + \frac{1}{\lambda} P_N c_{N+1} x_{N+1} - \frac{1}{\lambda} K_{N+1} c_{N+1}^T P_N c_{N+1} x_{N+1} \\ &= a_N - K_{N+1} c_{N+1}^T a_N + \frac{1}{\lambda} P_N c_{N+1} x_{N+1} \\ &\quad - \frac{1}{\lambda} \frac{P_N c_{N+1} c_{N+1}^T P_N c_{N+1} x_{N+1}}{\lambda + c_{N+1}^T P_N c_{N+1}} \\ &= a_N + K_{N+1}\left(x_{N+1} - c_{N+1}^T a_N\right) \,. \end{aligned}$$

To use the recursive least square (RLS) method, we first perform an initialization step, considering N_{in} samples of the signal and computing

$$P_{N_{in}} = \left[C_{N_{in}}^T \Lambda_N C_{N_{in}}\right]^{-1},$$
$$a_{N_{in}} = P_N C_{N_{in}}^T \Lambda_{N_{in}} X_{N_{in}}.$$

After this step, the following recursion is used

$$K_{N+1} = \frac{P_N c_{N+1}}{\lambda + c_{N+1}^T P_N c_{N+1}},$$
$$P_{N+1} = \frac{1}{\lambda}\left(P_N - K_{N+1} c_{N+1}^T P_N\right),$$
$$a_{N+1} = a_N + K_{N+1}\left(x_{N+1} - c_{N+1}^T a_N\right), \quad (19)$$

for $N = N_{in}, \ldots, N_T$, obtaining in this way a new set of coefficients for the AR(p) model each time a new sample of the signal is considered. This kind of autoregressive method where the coefficients change in time is called a Dynamic Autoregressive method of order p, DAR(p), (Young, 1999).
In order to combine adaptive filters with optical flow calculation, we follow the next steps:

1. We calculate velocities for N_{in} frames of each sequence by using Lucas and Kanade algorithm and we use this samples to inicialize the filter coefficients.
2. For the N-th frame, we calculate the velocity v_N in the following way:

   a. We calculate v_N^{of} by using optical flow.

   b. We estimate v_N^{af} by using an adaptive filter.

   c. If $|v_N^{of} - v_N^{af}| < tol_N$, then $v_N = v_N^{of}$. Else $v_N = v_N^{af}$, where tol_N are certain tolerance factors, which depend on the sequence to be analyzed.

2.3 Numerical results

The method explained above has been used to analyze synthetic and real video sequences. The result has been satisfactory, since the algorithm has allowed to track the objects along the whole sequence.

Fig. 1. Moving object

In all the examples, we have used windows of 7×7 pixels to calculate optical flow by using Lucas and Kanade algorithm. The point to track has been selected by indicating to the program the zone where the object that we want to track is. In a final application, we would

select a greater number of points of the image, discarding those that did not have movement. In this way, we would only consider the points that belong to objects in movement.
Parameter values are a critical factor for the correct operation of the proposed algorithms. A very small value of the tolerance will cause the methods to select predicted velocities when there is no obstacle. On the other hand, a large value of the tolerance will make that the methods can not detect any occlusion.
We will analyze the occlusion problem by using the example observed in figure 1. It consists in an object with a horizontal movement and a velocity *vel* pixels per frame.

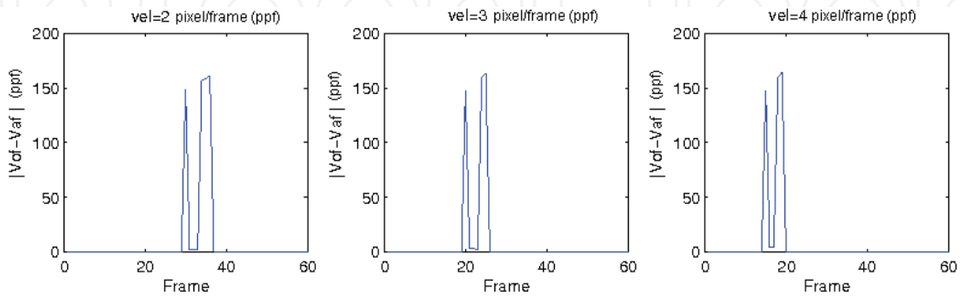


Fig. 2. Difference between optical flow and prediction

We have calculated $|v_x^{of} - v_x^{af}|$ for $vel = 2$, $vel = 3$ and $vel = 4$ pixels per frame (ppf) for the different frames of the sequence. The results can be observed in figure 2. We can see that there are two different peaks for each value of *vel* that corresponds with the input and output frontiers of the obstacle. In these points there are two discontinuities that produce a large error in the calculation of the optical flow. These peaks will always be detected. If we analyze what happens in the region corresponding to the transition of the object through the obstacle, we obtain the figure 3.

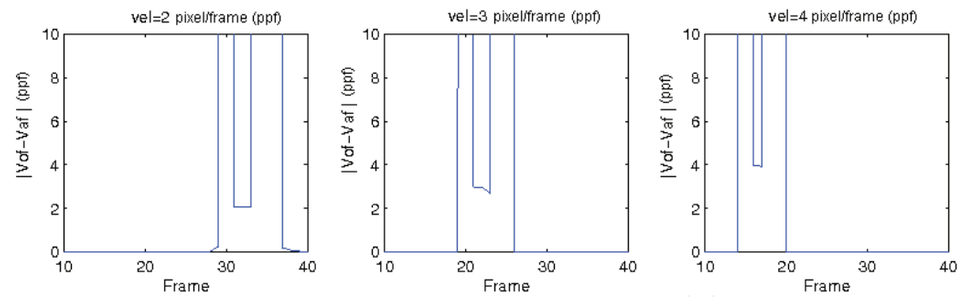


Fig. 3. Difference between optical flow and prediction (zoom)

As we can see, the values of $|v_x^{of} - v_x^{af}|$ are very similar to the velocity values of the object because the optical flow algorithm considers that obstacle is a static object so $v_{x,y}^{of} = 0$ in this region. In this way, we will use a variable value of tolerance in our algorithm for each frame N

$$tol_N = k\,|v_{N-1}| \tag{20}$$

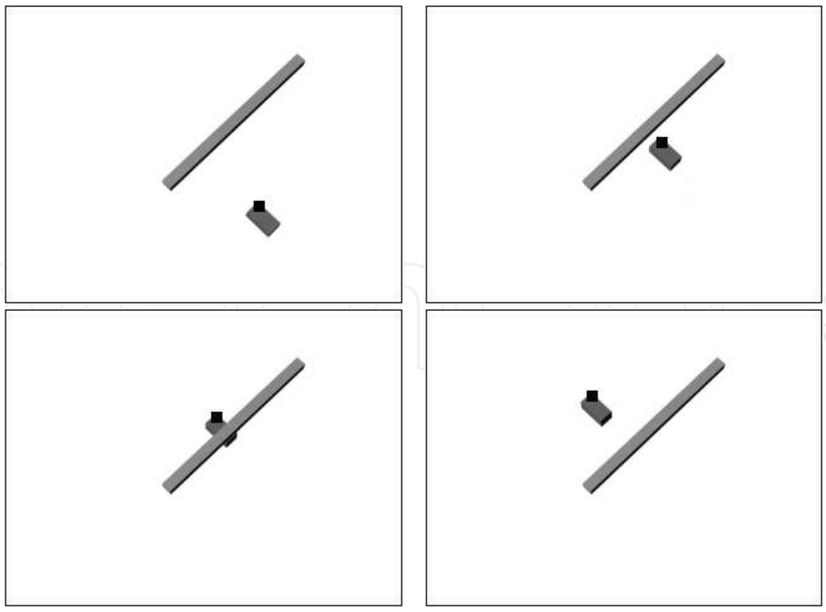


Fig. 4. RLS tracking in a synthetic sequence

In the different sequences that have been analyzed we have used a value of $k = 0.75$.
In figure 4, we can observe differents frames of a synthetic video sequence that consists in an object that follows a curved trajectory passing under an obstacle. Another example is shown in figure 5. We can see an urban route where there is a partial vehicle occlusion because of a street light. In these two examples, occlusions have been handled by using a RLS filter. We have used a value of $N_{in} = 7$ frames, an order filter of 2 and a forgetting factor of $\lambda = 0.99$.
In both sequences, without using the RLS filter, an error occurs when the object arrives at the obstacle. As we can see, by using the RLS filter, objects can be tracked along the whole sequences.
This method has demonstrated to be very efficient to handle occlusion because of its adaptability and tolerance to noisy data.

3. Fuzzy control for obstacle detection in object tracking

As we have said in the former section, tolerance values are a critical factor for the correct operation of the proposed algorithms. A very small value of the tolerance will cause the methods to select predicted velocities when there is no obstacle. On the other hand, a large value of the tolerance will make that the methods can not detect any occlusion. For this reason, the choice of parameter k is very critical. In this section, we propose to use a variable value for parameter k controlled by a fuzzy system, instead of selecting its value a priori.

3.1 Fuzzy controller

A fuzzy control system is a control system based on fuzzy logic (Zadeh, 1965). Fuzzy control provides a formal methodology for representing, manipulating and implementing a human's heuristic knowledge about how to control a system (Passino & Yurkovich, 2000). The fuzzy controller block diagram is given in figure 6, where we can see a fuzzy controller embedded in a closed-loop control system.

Fig. 5. RLS tracking in a real sequence

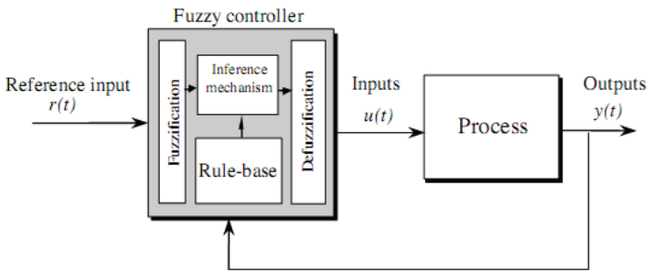


Fig. 6. Fuzzy controller architecture

To design a fuzzy controller, we have to specify the different blocks of the system:

- Input variables LHS
- Output variables RHS
- Input fuzzification method
- Fuzzy rules
- Inference engine parameters
- Defuzzification method

3.1.1 Fuzzification

The function of the fuzzificator is the conversion of the real input x into a fuzzy set. To do this work, it is neccessary to define a set of membership functions. In figure 7, we can see an example of this set.

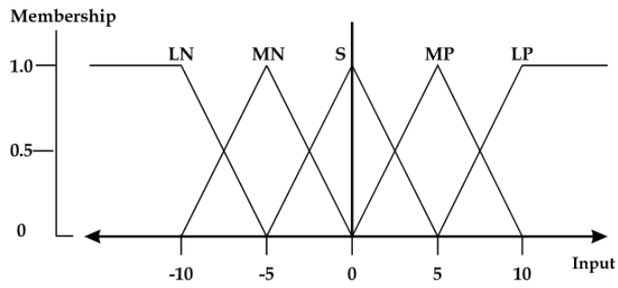


Fig. 7. Membership functions. LN: large negative, MN: medium negative, S: small, MP: medium positive, LP: large positive

The values of each membership function are labeled by $\mu(x)$ and determined by the input value x and the shape of each membership function. In the example of figure 7, if we have an input value $x = 2$, the set of fuzzy variables will be:
$\mu_{LN} = 0$
$\mu_{MN} = 0$
$\mu_{S} = 0.6$
$\mu_{MP} = 0.4$
$\mu_{LP} = 0$

3.1.2 Fuzzy decision blocks

Fuzzy control requires a specification of the rules in terms of linguistic variables, that are conditional statements of type IF-THEN. These rules relate input fuzzy values to output fuzzy value. A set of fuzzy rules R^l is:

$$R^l : \ IF\ x_1\ is\ F_1^l\ and\ ...\ and\ x_n\ is\ F_n^l\ THEN\ y\ is\ G^l,$$

where x_i and y are input and output linguistic variables, and F_i^l and G^l are fuzzy sets.
In this case, we consider a system with multiples inputs and one output. A system with multiple outputs can be separated in a set of simple fuzzy systems.

3.1.3 Inference engine

The inference engine produces a global output $\mu_{out}(y)$, by using the set of rules and considering the membership degree of the input with each membership function. The global output $\mu_{out}(y)$ is a fuzzy set that is calculated from a set of membership functions for the output y.
Continuing the previous example, if we have the two following rules:
If x is small (S), Then y is small (S)
If x is medium positive (MP), Then y is positive (P)
, and we know that
$\mu_{S} = 0.6$
$\mu_{MP} = 0.4$
by using the set of membership functions for the output y observed in figure 8, we will obtain the global output of the figure 9.
Depending of the method of defuzzification, we can consider an only global output $\mu_{out}(y)$ as the union of all the contributions, or a global output $\mu_{out}^k(y)$ for each contribution.

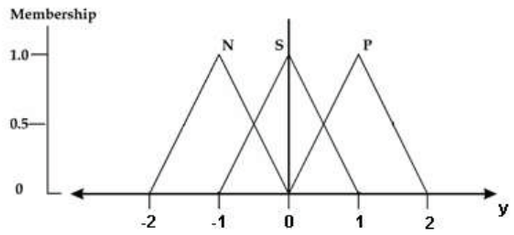


Fig. 8. Membership output functions. N: negative, S: small, P: positive

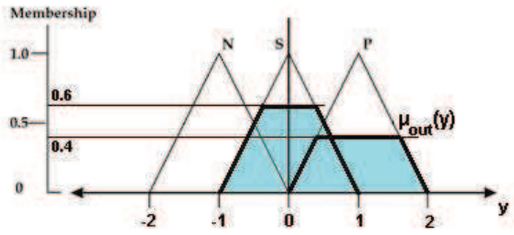


Fig. 9. Global output

3.1.4 Defuzzification

The defuzzification is a process to obtain a real value u^* representative of $\mu_{out}(u)$. There are different methods for defuzzification and method selection is very important for the efficiency and precision of the controller.
The most used defuzzification techniques are:

- COA o Center of Areas
  This technique calculates u^* as the geometric center of the output fuzzy value $\mu_{out}(u)$, where $\mu_{out}(u)$ is the combination of all the contributions. u^* divides the area in two regions with the same area. Some problems of this method are the computational cost, the preference for central values and that the overlapping area is only counted once.
- COS o Center of Sums
  This method operates separately for each contribution. The overlapping areas are considered more than once. It is efficient and it is the most used method.
- MOM o Mean of Maxima
  This technique calculates the mean value of points with maxima membership degree in the global output $\mu_{out}(u)$. It is very fast, but it does not consider the shape of $\mu_{out}(u)$.
- CA o Center Average
  This method operates separately for each contribution and calculates a pondered average of the centers of the contributions.

3.2 Obstacle detection

In our system, we want the value of tol to be small when there is an obstacle and large in any other case. The value of tol can be controlled by the variable k. In this way, we have designed a fuzzy system to control the value of k.

The system has two inputs, $\Delta\varepsilon$ and qV, and one output, k.
The variable $\Delta\varepsilon$ is the increment error (EI). If we call error ε to the absolute value of the difference between the velocity calculated by optical flow and the velocity estimated by the adaptive filter, we have that:

$$\varepsilon_n = \left| v_n^{of} - v_n^{af} \right| , \tag{21}$$

$$\Delta\varepsilon_n = \varepsilon_n - \varepsilon_{n-1} . \tag{22}$$

The variable qV is a binary variable that indicates if we are using the velocities calculated by optical flow in the previous frames ($qV = 1$) or the ones estimated by the adaptive filter ($qV = 0$).
In figure 10, we can see the set of membership functions used to the fuzzification of the input $\Delta\varepsilon$. Fuzzy values for this input can be negative big (NB), medium (M) or positive big (PB). NB values correspond to the final of the peaks that exist in the frontiers of the obstacle as we can see in figure 2, PB values correspond to the beginning of these peaks and M value correspond to the other situations.

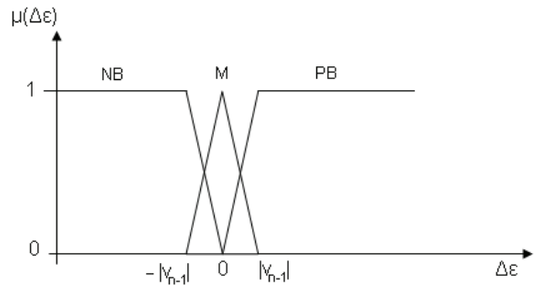


Fig. 10. Membership functions for the input $\Delta\varepsilon$

In this system, the value of variable k must be high (H), to select the velocity calculated by optical flow, when there is no occlusion (EI=M, qV=1) and when the target is in the output of the obstacle (EI=NB, qV=0). In all the other cases, the value of k will be low (L) to select the velocity estimated by the adaptive filter.
In this way, the system fuzzy rules are the following:

- IF EI is NB and qV=0 THEN k is H
- IF EI is M and qV=0 THEN k is L
- IF EI is PB and qV=0 THEN k is L
- IF EI is NB and qV=1 THEN k is L
- IF EI is M and qV=1 THEN k is H
- IF EI is PB and qV=1 THEN k is L

The inference engine calculates the global output $\mu_{out}(k)$ by using these conditions and the set of output membership functions that we can observe in figure 11. In this way, the value of the output variable k can oscillate between 0.5 and 1.
Finally, the defuzzificator calculates from the global output $\mu_{out}(k)$ the real value for the variable k by using the center of sums technique, that provides fast and satisfactory results.

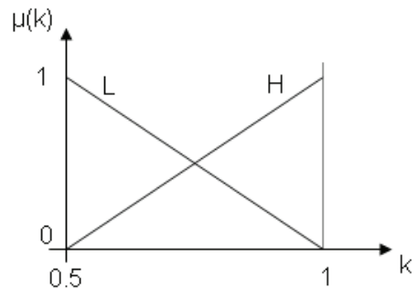


Fig. 11. Membership functions for the output k

3.3 Results

The algorithm exposed has been tested in different video sequences, studying synthetic and real traffic videos. We have used windows of 7×7 pixels for Lucas and Kanade algorithm, and a value of $N_{in} = 7$ frames, an order of 2 and a forgetting factor of $\lambda = 0.99$ for the adaptive filter.

In figure 12, we can observe differents frames of a synthetic video sequence designed with 3D Studio Max ® that consists in an object that follows a curved trajectory passing under an obstacle.

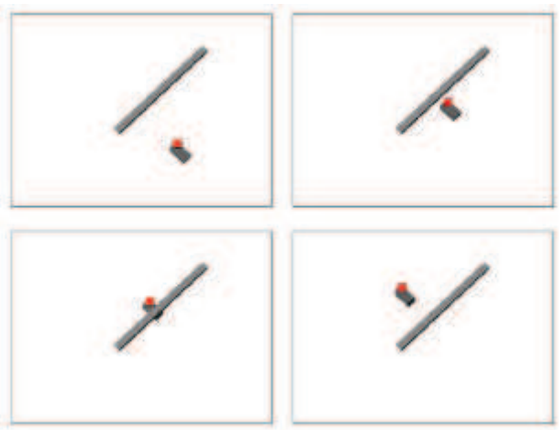


Fig. 12. Fuzzy object tracking in a synthetic sequence

In this example, the occlusion is produced in frames 37-43. As we can observe in figure 13, the value of k decreases in those frames. In this way, the predicted velocity is selected while the target is hidden by the obstacle and the object is correctly tracked along all the sequence.

The same result is obtained analyzing the example shown in figure 14. In this figure, we can see an urban route where there is a partial vehicle occlusion because of a street light.

In this case, the occlusion occurs in frames 21-27 and, as we can see in figure 15, k decreases in this position. On the other hand, in this figure, we can observe that the value of k also decreases in frames 51-54. This change is due to the fact that there is an error in the optical flow calculation. The performance of the system in this point is very interesting since it is able to correct possible errors in the optical flow calculation, besides handling occlusion.

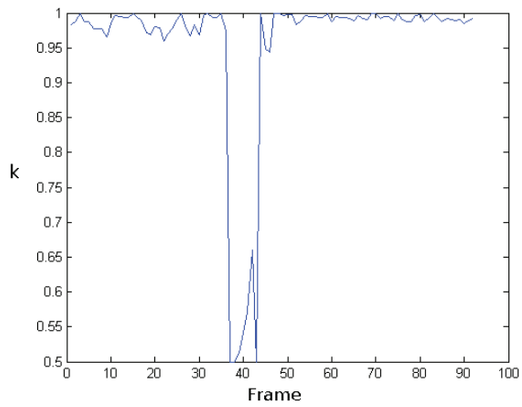


Fig. 13. *k* values in synthetic sequence

Fig. 14. Object tracking in a real sequence

4. Handling occlusion in object tracking in stereoscopic video sequences

Stereo vision is a part of the field of computer vision that uses two or more cameras to obtain two or more images from which distance information can be obtained by using the triangulation principle (Cochran & Medioni, 1992). We can obtain three-dimensional information of static scenes by using this technique. On the other hand, we can use optical flow algorithms for object tracking in two dimensional video sequences along the time (Trucco & Plakas, 2006).

In (Parrilla et al., 2005), we have studied a system that combines stereoscopic vision and optical flow algorithms for object tracking in a three-dimensional space. We select a set of points to be tracked in the first frame of one video of the stereo sequence and, by using optical flow algorithms, we track them in that sequence. For each frame, we calculate the disparity of these points by using the other video and stereo algorithms. In this way, we know the position of each point in a three dimensional space (disparity) and along the time (optical flow). One of the most important problems of this technique is that this method is not able to handle the occlusion of the moving objects.

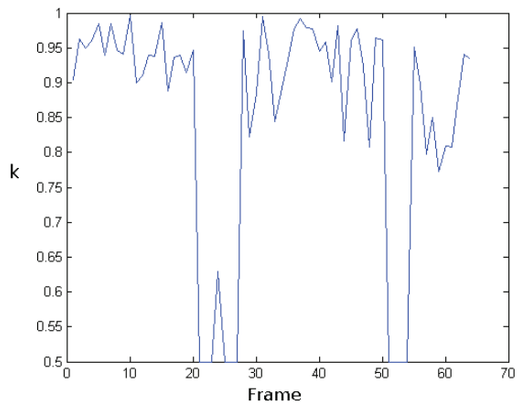


Fig. 15. k values in real sequence

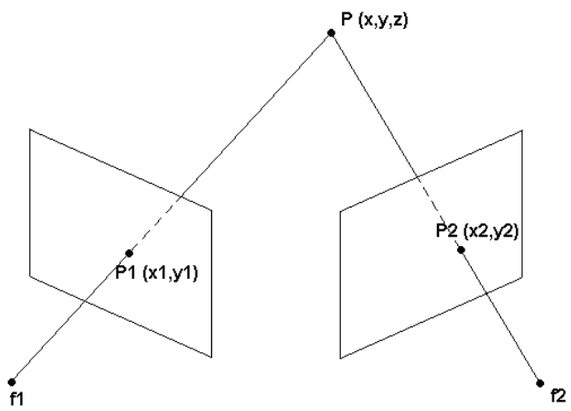


Fig. 16. Triangulation principle

For solving this occlusion problem, we propose the use of adaptive filters (Haykin, 1986; Ljung, 1999) to predict the expected 3D velocities of the objects. In this section, we combine adaptive filters, optical flow algorithms and stereo vision, in order to handle occlusion in object tracking in stereoscopic video sequences.

Stereo vision (Cochran, 1990) consists in, given a point P from which we know their projections P_1 and P_2 in the images of a pair of cameras, calculating its position in the space by using the triangulation principle (see figure 16). If we designate the cartesian coordinates of the projections as (x_1, y_1) and (x_2, y_2) in the respective coordinate systems of the cameras, we can obtain their relationship with the coordinates (x, y, z) of point P.

Without loss of generality, we will assume that the coordinate system of the left camera and the real world is the same and, to simplify the problem, we will consider that the axes of the cameras are parallel and the coordinate system of the right camera is similar but moved a distance B along the axis x. We will also assume that the two cameras are equal with a focal length f. In this way we have the following equations:

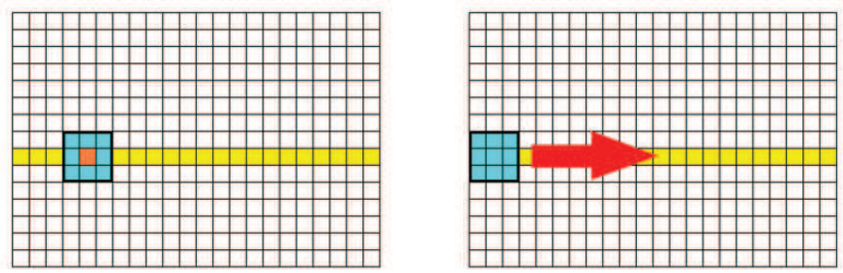


Fig. 17. Correlation search

$$\begin{cases} \dfrac{x_1}{f} = \dfrac{x}{z} \\ \dfrac{x_2}{f} = \dfrac{x-B}{z} \end{cases} \tag{23}$$

By solving the system of equations we obtain the following results:

$$x = \frac{x_1 B}{x_1 - x_2}\,, \tag{24}$$

$$z = \frac{Bf}{x_1 - x_2}\,. \tag{25}$$

By proceeding in a similar way we also can calculate that:

$$y = \frac{y_1 B}{x_1 - x_2}\,. \tag{26}$$

The main problem of stereo vision is to find the correspondences between the points of the different images (disparity), i.e, given the projection P_1 in the left image, to find what point of the right image corresponds to the projection P_2, to situate in the space the point P.
We have programmed a correlation search method that has provided satisfactory results. This technique selects a neighborhood around the point P_1 and calculates the correlation with a window that will displace over the different points of the right image (see figure 17). Finally we select that point where the correlation is maximum.
Because of the geometry of the problem, we only have to displace the window along a line (epipolar line) because projections P_1 and P_2 will be situated in the same horizontal (see figure 18).

4.1 Two-dimensional adaptive filter

To obtain an indicator of the performance of the tracking algorithm, each component of the velocity estimated by using optical flow and stereo vision, for the different frames of the sequences, is considered as a time series. We use adaptive filters to predict instantaneous velocities in order to compare them with the velocities obtained by using optical flow and stereo vision. For the component v_y, we use a simple adaptive filter because this component is the same in the two video sequences. On the other hand, there are two components v_{Lx} and v_{Rx} that correspond to the velocities in the left and right images. In this case, we will adapt the RLS filter to operate with two signals simultaneously.

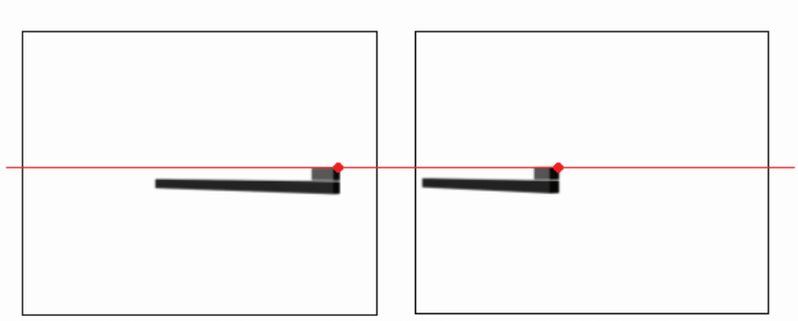


Fig. 18. Epipolar line

We start from two signals x_{Ln} and x_{Rn}, $n = 0, \ldots, N_T$, and we assume for the signals an AR(p) model

$$\begin{aligned} x_{Lk} &= -(a_1)_k\, x_{Lk-1} - (a_2)_k\, x_{Rk-1} - \cdots \\ &\quad \cdots - (a_{2p-1})_k\, x_{Lk-p} - (a_{2p})_k\, x_{Rk-p} + \varepsilon_k\,, \\ x_{Rk} &= -(b_1)_k\, x_{Lk-1} - (b_2)_k\, x_{Rk-1} - \cdots \\ &\quad \cdots - (b_{2p-1})_k\, x_{Lk-p} - (b_{2p})_k\, x_{Rk-p} + \varepsilon'_k\,, \end{aligned} \tag{27}$$

In this case, x_{Lk} and x_{Rk} will depend on the previous samples of both signals. Assuming this model for $k = p, \ldots, N$, we obtain the system of equations

$$\begin{bmatrix} X_{LN} \\ X_{RN} \end{bmatrix} = C_N \begin{bmatrix} a_N \\ b_N \end{bmatrix} + \epsilon_N\,,$$

where

$$X_{LN} = \begin{bmatrix} x_{Lp} \\ x_{Lp+1} \\ \vdots \\ x_{LN} \end{bmatrix}, \quad X_{RN} = \begin{bmatrix} x_{Rp} \\ x_{Rp+1} \\ \vdots \\ x_{RN} \end{bmatrix}, \quad C_N = \begin{bmatrix} x_{Lp-1} & x_{Rp-1} & \cdots & x_{L0} & x_{R0} \\ x_{Lp} & x_{Rp} & \cdots & x_{L1} & x_{R1} \\ \vdots & & & \vdots & \\ x_{LN-1} & x_{RN-1} & \cdots & x_{LN-p} & x_{RN-p} \end{bmatrix},$$

$$a_N = \begin{bmatrix} -(a_1)_N \\ -(a_2)_N \\ \vdots \\ -(a_{2p})_N \end{bmatrix}, \quad b_N = \begin{bmatrix} -(b_1)_N \\ -(b_2)_N \\ \vdots \\ -(b_{2p})_N \end{bmatrix}, \quad \epsilon_N = \begin{bmatrix} \varepsilon_p \\ \vdots \\ \varepsilon_N \\ \varepsilon'_p \\ \vdots \\ \varepsilon'_N \end{bmatrix}.$$

This system of equations can be considered as two separated systems of equations with the same matrix C_N, that contains samples of both signals x_{LN} and x_{RN}

$$\begin{aligned} X_{LN} &= C_N a_N + \epsilon_N\,, \\ X_{RN} &= C_N b_N + \epsilon'_N\,. \end{aligned} \tag{28}$$

If we apply the RLS filter theory to these systems of equations, we will obtain an adaptive filter for two correlated signals. To use this filter, we first perform an initialization step, considering N_{in} samples of the signals and computing

$$\begin{aligned} P_{N_{in}} &= \left[C_{N_{in}}^T \Lambda_N C_{N_{in}}\right]^{-1}, \\ a_{N_{in}} &= P_N C_{N_{in}}^T \Lambda_{N_{in}} X_{LN_{in}}, \\ b_{N_{in}} &= P_N C_{N_{in}}^T \Lambda_{N_{in}} X_{RN_{in}}. \end{aligned}$$

After this step, the following recursion is used

$$\begin{aligned} K_{N+1} &= \frac{P_N c_{N+1}}{\lambda + c_{N+1}^T P_N c_{N+1}}, \\ P_{N+1} &= \frac{1}{\lambda}\left(P_N - K_{N+1} c_{N+1}^T P_N\right), \\ a_{N+1} &= a_N + K_{N+1}\left(x_{LN+1} - c_{N+1}^T a_N\right), \\ b_{N+1} &= b_N + K_{N+1}\left(x_{RN+1} - c_{N+1}^T b_N\right), \end{aligned} \tag{29}$$

where

$$c_{N+1}^T = \left[x_{LN}, x_{RN}, x_{LN-1}, x_{RN-1}, \cdots, x_{LN-p+1}, x_{RN-p+1}\right],$$

for $N = N_{in}, \ldots, N_T$, obtaining in this way a new set of coefficients for the AR(p) model each time a new sample of the signals is considered.
In this way, we can estimate the following sample of the signals

$$\begin{aligned} x_{LN+1} &= c_{N+1}^T a_{N+1}, \\ x_{RN+1} &= c_{N+1}^T b_{N+1}. \end{aligned} \tag{30}$$

4.2 Handling occlusion

In order to predict the velocities of the objects, we follow the next steps:

1. We calculate velocities for N_{in} frames of each sequence by using Lucas and Kanade algorithm and stereo vision, and we use this samples to inicialize the filter coefficients.
2. For the N-th frame, we calculate the velocities v_{LN} and v_{RN} in the following way:
   a. We calculate v_{LN}^{of} and v_{RN}^{of} by using optical flow and stereo vision.
   b. We estimate v_{LN}^{es} and v_{RN}^{es} by using an adaptive filter.
   c. If $|v_{L,RN}^{of} - v_{L,RN}^{es}| < tol_{L,RN}$, then $v_{L,RN} = v_{L,RN}^{of}$. Else $v_{L,RN} = v_{L,RN}^{es}$
3. We use $v_{L,RN}$ and $N_{in} - 1$ previous samples to update the filter coefficients.

Tolerance values are a critical factor for the correct operation of the proposed algorithm. A very small value of the tolerance will cause the method to select predicted velocities when there is no obstacle. On the other hand, a large value of the tolerance will make that the method can not detect any occlusion. In Section 2, we have demonstrated that an optimum value of tolerance is

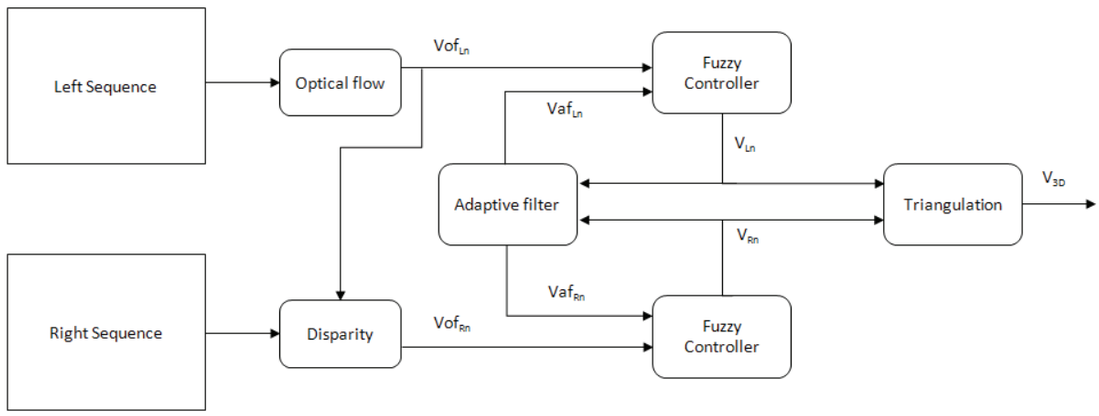


Fig. 19. Tracking system diagram

$$tol_{L,RN} = k\,|v_{L,RN-1}|\,. \tag{31}$$

In the same way as in the case of two-dimensional tracking, the values of k will be controlled by a fuzzy system.

In Figure 19, we can see a diagram of the entire system. An optical flow algorithm is applied to the left sequence in order to perform a 2D tracking of the object. From the obtained results, by calculating the disparity, we track the object in the right sequence. Moreover, the velocity values are predicted by using the adaptive filter. The velocity values calculated using optical flow and the predicted values are the inputs of two identical fuzzy controllers that are responsible for deciding which velocities are selected according to whether there is occlusion. These selected velocities will also be the inputs of the adaptive filter that will update the coefficients in order to make the prediction in the next frame. Once the object has been tracked in both sequences, using the triangulation principle, we can obtain the 3D velocity of the object.

4.3 Numerical results

The method explained above has been used to analyze synthetic and real video sequences. The result has been satisfactory, since the algorithm has allowed to track the objects along the whole sequence.

In all the examples, we have used windows of 7×7 pixels to calculate optical flow and disparity.

In Figure 20, we can observe differents frames of a synthetic video sequence that consists in an object that moves over a rail and passes behind an obstacle. Another example is shown in Figure 21. We can see an urban route where there is a partial vehicle occlusion because of a street light. In these two examples, occlusions have been handled by using an adaptive filter. We have used a value of $N_{in} = 7$ frames, an order filter of 2 and a forgetting factor of $\lambda = 0.99$. As we can observe in the images, objects are successfully tracked along all the frames, there is not any error when they disappear behind the obstacles and their trajectory is predicted correctly.

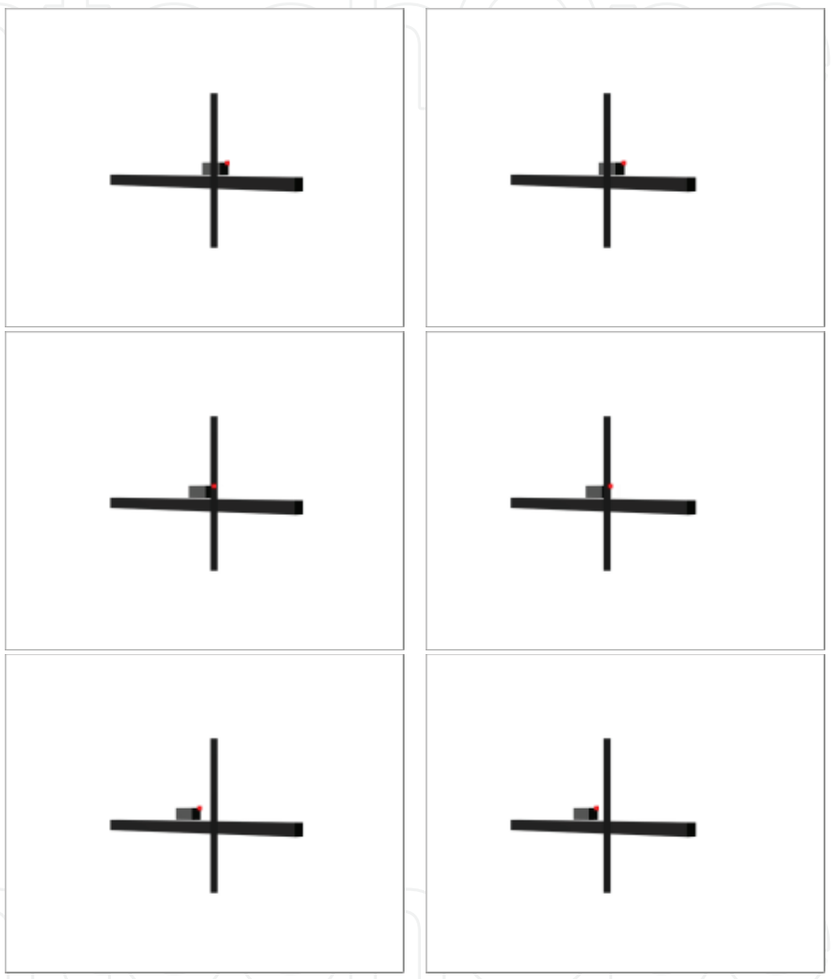


Fig. 20. RLS tracking stereo in a synthetic sequence

Fig. 21. RLS tracking stereo in a real sequence

5. References

Barron, J.L., Fleet, D.J. & Beauchemin, S.S. (1994) Performance of Optical Flow Techniques, *International Journal of Computer Vision*, Vol. 12(1): 43-77.

Bourel, F., Chibelushi, C.C. & Low, A.A. (2001) Recognition of facial expressions in the presence of occlusion, *Proceedings of the Twelfth British Machine Vision Conference*, Vol. 1: 213-222.

Cochran, S.D. (1990) Surface description from binocular stereo, *Dissertation presented in partial fulfillment of the requirements for the degree Doctor of Philosophy*, Vol. 1.

Cochran, S.D. & Medioni, G. (1992) 3-D Surface Description from Binocular Stereo, *IEEE Transactions on Pattern Analysis and Machine Intelligence*, Vol. 14(10): 981-994.

Erdem, Ç.E., Tekalp, A.M. & Sankur, B. (2003) Video Object Tracking with Feedback of Performance Measures, *IEEE Trans. Circuits and Systems for Video Technology*, Vol. 13(4): 310-324.

Haykin, S. (1986) *Adaptive Filter Theory*, Prentice-Hall, Englewood Cliffs NJ.

Horn B.K.P. & Schunck B. G. (1981) Determining optical flow, *Artificial Intelligence*, Vol. 17(1-3): 185-203.

Iñigo, R.M. (1989) Application of machine Vision to Traffic Monitoring and Control, *IEEE Transactions on Vehicular Technology*, Vol. 38(3): 112-122.

Ji, X., Wei, Z. & Feng, Y. (2006) Effective Vehicle Detection Technique for Traffic Surveillance Systems, *J. Vis. Commun. Image R.*, Vol. 17: 647-658.

Ljung, L. (1999) *System Identification. Theory for the User*, Prentice-Hall, Upper Saddle River NJ.

Lukas, B.D. & Kanade, T. (1981) An iterative image registration technique with an application to stereovision, *Proceedings of Imaging Understanding Workshop*, 121-130.

Ogata, K. (1987) *Discrete-Time Control Systems*, Prentice Hall. Upper Saddle River, NJ.

Parrilla, E., Riera, J., Giménez, M., Torregrosa, J.R. & Hueso, J.L. (2005) Vehicle tracking in urban routes by means of Lucas&Kanade algorithm, *Proceedings of the International Conference on Computational and Mathematical Methods in Science and Engineering*, 438-445.

Parrilla, E., Riera, J., Giménez, M., Torregrosa, J.R. & Hueso, J.L. (2005) Cálculo de velocidades mediante un sistema estereoscópico y algoritmos de flujo óptico, *Proceedings of the Congreso de Ecuaciones Diferenciales y Aplicaciones*.

Parrilla, E., Ginestar, D., Hueso, J.L., Riera, J. & Torregrosa, J.R. (2008) Handling occlusion in optical flow algorithms for object tracking, *Computers and Mathematics with Applications*, Vol. 56(3): 733-742.

Parrilla, E., Riera, J., Torregrosa, J.R. & Hueso, J.L. (2009) Handling occlusion in object tracking in stereoscopic video sequences, *Mathematical and Computer Modelling*, Vol. 50(5-6): 823-830.

Passino, K.M. & Yurkovich, S. (2000) Fuzzy control, *International Journal of General Systems*, Vol. 29(2): 341-342.

Rumelhart, D. & McClelland, J. (1986) *Parallel Distributed Processing*. MIT Press, Cambridge, MA.

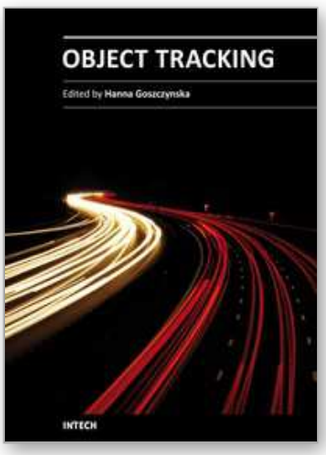
Shin, J., Kim, S., Kang, S., Lee, S., Paik, J., Abidi, B., & Abidi, M. (2005) Optical flow-based real-time object tracking using non-prior training active feature model, *Real-Time Imaging*, Vol. 11: 204-218.

Trucco, E. & Plakas, K. (2006) Video Tracking: A Concise Survey, *IEEE Journal of Oceanic Engineering*, Vol. 31(2): 520-529.

Wren, C.R., Azerbayejani, A., Darrell, T. & Pentland, A.P. (1997) Pfinder: Real-time tracking of the human body, *IEEE Trans. Pattern Anal. Machine Intell*, Vol. 19: 780-785.

Young, P.C. (1999) Nonstationary time series analysis and forecasting, *Progress in Environmental Science*, Vol. 1(1): 3-48.

Zadeh, L.A. (1965) Fuzzy sets, *Inf. Control*, Vol. 8: 338-353.



Object Tracking

Edited by Dr. Hanna Goszczynska

ISBN 978-953-307-360-6
Hard cover, 284 pages
Publisher InTech
Published online 28, February, 2011
Published in print edition February, 2011

Object tracking consists in estimation of trajectory of moving objects in the sequence of images. Automation of the computer object tracking is a difficult task. Dynamics of multiple parameters changes representing features and motion of the objects, and temporary partial or full occlusion of the tracked objects have to be considered. This monograph presents the development of object tracking algorithms, methods and systems. Both, state of the art of object tracking methods and also the new trends in research are described in this book. Fourteen chapters are split into two sections. Section 1 presents new theoretical ideas whereas Section 2 presents real-life applications. Despite the variety of topics contained in this monograph it constitutes a consisted knowledge in the field of computer object tracking. The intention of editor was to follow up the very quick progress in the developing of methods as well as extension of the application.

How to reference

In order to correctly reference this scholarly work, feel free to copy and paste the following:

Eduardo Parrilla, Jaime Riera, Juan R. Torregrosa and José L. Hueso (2011). Fuzzy Control System to Solve Coupling Between Tracking and Predictive Algorithms, Object Tracking, Dr. Hanna Goszczynska (Ed.), ISBN: 978-953-307-360-6, InTech, Available from: http://www.intechopen.com/books/object-tracking/fuzzy-control-system-to-solve-coupling-between-tracking-and-predictive-algorithms

INTECH
open science | open minds

InTech Europe
University Campus STeP Ri
Slavka Krautzeka 83/A
51000 Rijeka, Croatia
Phone: +385 (51) 770 447
Fax: +385 (51) 686 166
www.intechopen.com

InTech China
Unit 405, Office Block, Hotel Equatorial Shanghai
No.65, Yan An Road (West), Shanghai, 200040, China
中国上海市延安西路65号上海国际贵都大饭店办公楼405单元
Phone: +86-21-62489820
Fax: +86-21-62489821

IntechOpen