Real-time identification of smoke images by clustering motions on a fractal curve with a temporal embedding method

Philippe Guillemant
Jérôme Vicente
Institut Universitaire des Systèmes
Thermiques Industriels
CNRS UMR 139
Marseilles, France

Abstract. Automatic forest fire detection with CCD cameras requires a landscape image analysis in two stages: first the tracking of local dynamic envelopes of pixels, and second the discrimination between the various natural phenomena that may cause such envelopes. For this second process, we have to deal with restrictive conditions: lack of spatial information, complexity of motions, and real-time constraints on detection. We present here a fast algorithm adapted to the extraction of complex motions in small spatial envelopes. The principle of the method is to extract local motions from cluster analysis of points in a multidimensional temporal embedding space. We detail the four successive steps of this method: temporal embedding of gray-levels, fractal indexing of points, chaining points into a linked list, and motion extraction from point sequences of the linked list. [DOI: 10.1117/1.1355254]

Subject terms: computer vision; pattern recognition; temporal embedding; signal processing; data clustering; fractal space-filling curves.

Paper 990482 received Dec. 6, 1999; revised manuscript received Aug. 18, 2000; accepted for publication Oct. 24, 2000.

1 Introduction

Forest fire detection usually requires observers looking for early smoke sources in the landscape from a high lookout post. As that is a difficult and a workforce-consuming task, automatic apparatus using image-processing algorithms has been developed for artificial vision and detection with CCD cameras. We had to improve one of these systems in order to reduce the number of false alarms due to various dynamic phenomena, such as wind-tossed trees, cloud shadows, reflections, human activity, and so on. The difficulties of processing landscape images are due to their varying nature and to the large number of dynamic events that may appear under various illumination conditions, depending on weather, distance, time of day, masking objects, etc. These events produce dynamic envelopes, which are not always caused by motion and consist of time-varying gray-levels of connected pixels in several image regions. The system given to us was able to extract persistent dynamic envelopes through a fast-optimized image differentiation algorithm. We present here a method to improve this system by estimating the various motions within the given envelopes, in order to discriminate smoke from other phenomena.

Usual image-processing tools for dynamical analysis work well on instantaneous motion detection of individual rigid or deformable moving objects,[1,2] but are ill suited to detect and quantify variegated and fleeting motions into small or dimly contrasted quasistatic envelopes. In the case of smoke, the motions to detect are not sharply contrasted, because they consist essentially of smoke fronts that propagate on an already smoke-filled background. Figure 1 shows an example of a smoke envelope produced by a fire in a sequence of images. As the smoke puffs vary slightly in direction, the contours of the envelope are unstable and fuzzy. Therefore any kind of spatial approach is very uncertain.

The fundamental basis of our approach is the observation that the motions of various patterns like smoke puffs are producing correlated temporal segments of gray-level pixels (Fig. 2), which we call *temporal signatures*. They are due to the propagation of the smoke puff into the pixels of the envelope. The temporal embedding of the gray-level segments allows getting the correlated segments as clusters of points. These clusters are formed by the repetition of temporal segments at various times and positions. Motion extraction is then reduced to the problem of clustering points. We need an algorithm that is fast enough to process envelopes in real-time, as these envelopes may contain thousands of new pixels for each sampling time. To have direct access to the neighbor points of the same cluster and then minimize computing time, we use a fractal indexing technique and an optimized linked-list algorithm.

The four subsections of the next section describe in detail the successive processing steps of our real-time algorithm, optimized for smoke detection. In the last section, we also present an empirical validation of the method through an off-line motion-detection calculation. Finally, representative results obtained with smoke-concurrent phenomena (clouds and wind-tossed trees) are illustrated and discussed.

2 Processing Steps

2.1 *Temporal Embedding*

Dynamic envelopes of pixels can be considered as sets of gray-level signals. Figure 2 shows the temporal correlation

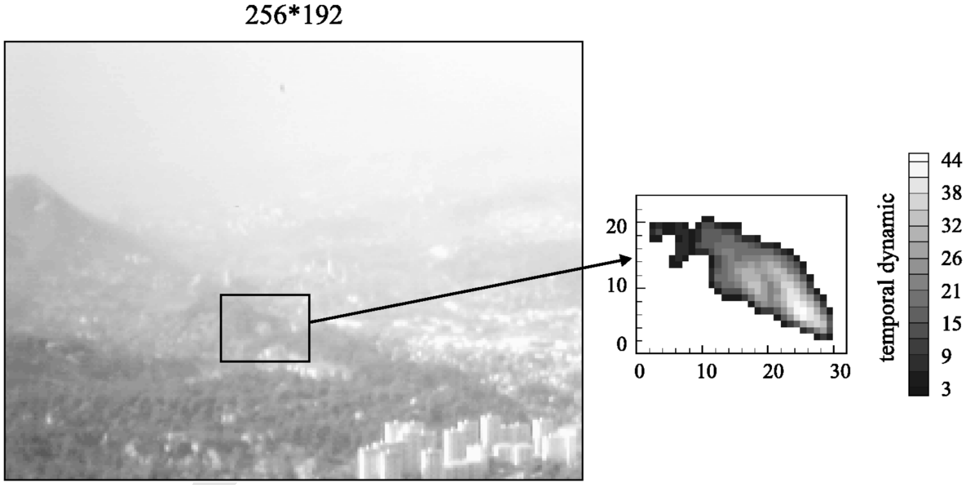


Fig. 1 Smoke source and its gray-level dynamic envelope for a 4-km-distant forest fire.

of four smoke signals incoming from four spatially close pixels, located along the mean trajectory of the smoke puffs. The two straight lines represent different propagation velocities of two successive smoke puffs that cross the same pixels. To follow just one smoke-puff trajectory, we have to track the short-time luminosity variations produced by the passage of a single puff. A temporal signature, for example the bold one plotted in Fig. 2, can represent such a transitory puff of smoke. When temporal segments show spatial redundancy, an appropriate choice of embedding parameters, using the method of time-delay coordinates,[3] may induce correlated data that appear as a cluster of points in the embedding space (Fig. 3). For each pixel p of the envelope we recover its last d luminosity variations in order to obtain in the mathematical embedding space a new point P_k:

$$P_k=(l_p(t-(d-1)\tau),\ \ldots,l_p(t-\tau),l_p(t)), \qquad (1)$$

where $l_p(t)$ is the luminosity of pixel p (coded in 8 bits) at time t.

The choices of the embedding dimension d and of the image sampling period τ are critical, as the puff segment duration depends on the puffs apparent propagation speed, which depends mainly on the distance of the observed smoke. The sampling period is fixed by the detection system at the value $\tau=0.33$ s. This is a sampling rate that is fast enough for close smokes (500 m from a 60-deg CCD objective), whose puffs cross quickly over the pixels. At the opposite extreme we need to detect distant smokes, at least 5 km away. In this case the mean frequency of the signals is reduced, and the optimal segment duration is longer. The embedding-space dimension d determines the duration of the temporal signature. If this duration is too short, far motions are not tracked, and if it is too long, close smoke puffs cannot be estimated. The choice $d=16$ that we made was guided by a balance between two considerations. A higher value would have required an excessive quantity of memory for the system and significantly increased the calculation time. A smaller value would have reduced the detection ability for distant smokes.

After embedding all the temporal segments of a given envelope for successive sampling times, groups of embed-

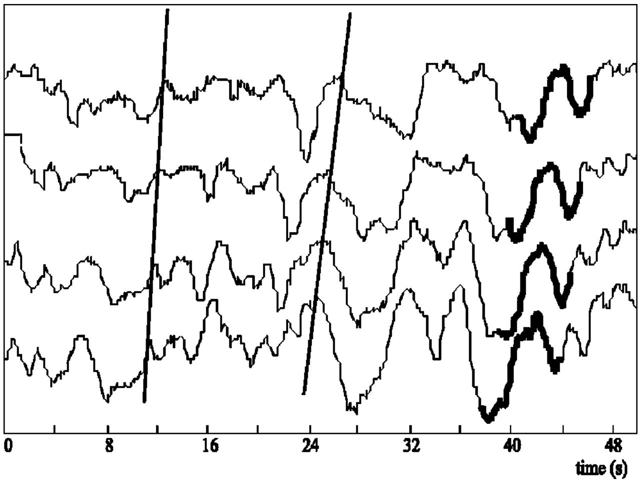


Fig. 2 Temporal signals of four pixels along a smoke-puff trajectory, and their correlated gray-level segments.

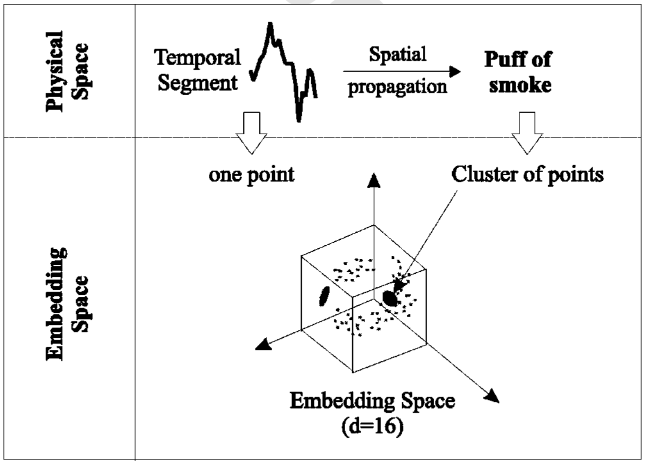
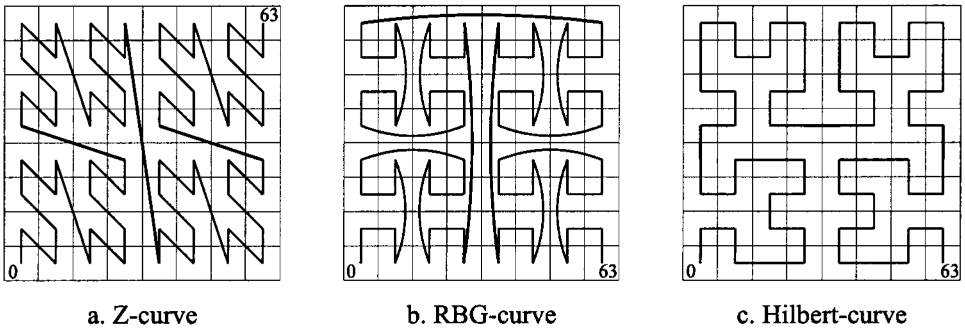


Fig. 3 Cluster formation induced by the spatial propagation of a correlated temporal segment.



a. Z-curve b. RBG-curve c. Hilbert-curve

Fig. 4 Fractal space-filling curves in dimension 2.

ded points are accumulated within numerous clusters. The extraction of these clusters from a large amount of embedded data requires an optimized clustering approach adapted to real-time analysis.

2.2 *Fractal Indexing*

2.2.1 *Clustering with space-filling curve*

Clustering techniques are most often used for data mining, which is the process of extracting useful information from very large data sets. Data mining consists essentially in looking for similitudes in a high-dimensional space where each point constitutes a record. More generally, finding nearest neighbors in *d*-dimensional space is a task encountered in many data-processing problems. Today these techniques are currently applied to the analysis of time-series,[4,5] multimedia,[6] and medical[7] databases.

The problem of multidimensional, in contrast with one-dimensional, data clustering is that no system of total ordering can preserve spatial locality. There was a significant step forward in the optimization of multidimensional databases due to the advent of spatial indexation[8,9] and spatial join algorithms,[10] which provide structured storage of the data and effective multidimensional queries. One-dimensional access may also be established by means of a R+ or ϵ-*k* tree indexation, for example.[9] Such indexation is obviously not general, and a recalculation is necessary for any new configuration of the data.

An interesting approach to the problem of spatial ordering, proposed by Orenstein,[11–13] uses a linear mapping of the multidimensional space. The most desired property of such a mapping is that the localities of objects in multidimensional space are preserved in the linear space. Sophisticated mapping functions have been proposed in the literature in order to design good distance-preserving mappings (Fig. 4). One, based on interleaving the bits from the coordinates, which is called the *Z* curve, was proposed in Ref. 11 [Fig. 4(a)]. An improvement on it was suggested by Faloutsos, using Gray coding on the interleaved bits.[14] A third method, based on the Hilbert curve [Fig. 4(b),(c)], has been proposed in Ref. 11. All distance-preserving space-filling curves are special instances of fractal curves.[15] They represent a continuous path going through each point in the space just once, and they also maintain continuity and proximity between the points. Thus spatially close points will find themselves close in position along the fractal space-filling curve. In general, a space-filling curve starts with a basic path on a *k*-dimensional square grid of size 2. The path visits every point in the grid exactly once without crossing itself. It has two free ends that may be joined with other paths. The basic curve is said to be of order 1; each vertex of the basic curve is then replaced by the curve of previous order, which may be rotated and/or reflected to fit the new curve.

2.2.2 *Z ordering*

Experimental data indexation on a space-filling curve is made possible by the fact that measurements are always discrete and have a limited resolution, encoded as a number *e* of bits. As a result, the embedding space has a limited number of points equal to 2^{de}, where *d* is the embedding dimension. Each point can be encoded as an index that represents its position on a space-filling curve. The value of this index, called the *fractal rank,* is contained between 0 and 2^{de}.

The Hilbert space-filling curve [Fig. 4(c)] achieves a better distance-preserving mapping than the other two space-filling curves,[16] but the *Z* curve yields the fastest index calculation.[13] As computing time was critical for our application, we used the *Z* curve to index our temporal embedded points. The basic *Z* curve for a 2×2 grid is shown in Fig. 5(a). Higher-order *Z* curves are obtained by replacing each vertex of the basic curve with the previous-order curve [Fig. 5(b),(c)]. Orenstein used the term *Z* ordering to refer to the ordering of points on the *Z* curve. He also used the term *Z* value to refer to the *Z* curve fractal rank.

Let *P* be a point in the *d*-dimensional space with its *e*-bits coordinates $x_l \in \{0,\ldots,2^{e-1}\}$, and x_l^j the *j*'th bit of the coordinate x_l. Thus

$$x_l=(x_l^{e-1},\ldots,x_l^1 x_l^0)_2=\sum_{j=0}^{j=e-1} 2^j x_l^j. \tag{2}$$

The *Z* values are computed by interweaving the bits of the binary representation of the coordinates of the point. The *Z* ordering permits quick analytical calculation of the *Z* value, *z(P)*, through the following relation:

$$z(P)=\sum_{j=0}^{e-1}\sum_{l=0}^{d-1} 2^{l+jd} x_l^j. \tag{3}$$

The example in Fig. 6 shows the *Z* value calculation for the point *P* of coordinates (1, 6), in dimension 2 and with $e=3$

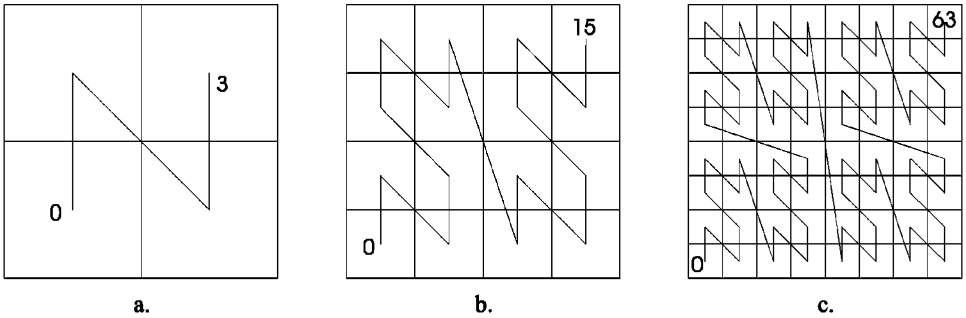


Fig. 5 *Z* curves of order 1, 2, and 3 in dimension 2 for discrete spaces that contain 4, 16, and 64 points, respectively.

bits. For higher-dimension spaces ($d>2$) the *Z* value is obtained in a similar way by interweaving *d* numbers instead of two.

2.3 *Linked-List Chaining*

The objective of the linking algorithm presented here is to give, for each incoming *Z*-ordered point, a direct access to the very few points that neighbor it. We use the distance-preserving properties of the *Z* curve to reduce drastically the number of comparisons that are necessary to extract these neighboring points. Indeed, scanning all the points would be prohibitive. All the embedded points are then linked into a list, with respect to their *Z* value, so that consecutive points in the list are neighboring points in the space. The problem of real-time cluster extraction then boils down to the comparison of a limited number of successive neighboring points along a dynamic list that is constantly refreshed. The chaining procedure we describe here makes it possible to insert every new incoming point into a linked list, between its immediately preceding and following neighbor points.

To connect consecutive points according to their *Z* value, we use two series *a()* and *b()*, which contain, for each point index, the indices of the next and previous points crossed by the *Z* curve. Such a doubly linked list is a data structure in which each element contains a datum and pointers to the next and the previous elements in the list. A pointer to a nonexistent location indicates the end of the list. To access the list we only have to know where its first element is stored. A new element is inserted into the list by redirecting the pointer of the preceding element and letting the new element point to the following one (Fig. 7).

In the case of *K* embedded points P_i, $i\in\{1,\ldots,K\}$, the list pointers are simply the point indices in the sampling order. If P_j is the point following P_i on the *Z* curve, then *a(i)=j*. Let *b()* be the inverse series of *a()*, such that *a(b(i))=i*.

The chaining technique we present now is optimized for real-time applications where fast addition of new points and removing of old points into the linked list is necessary. We use an optimized memory access algorithm that works as follows: when a new point P_i is embedded, we calculate its *Z* value $z(P_i)$. After performing low-order bit truncation of $z(P_i)$, we obtain an *m*-bit *Z* value to identify its hypercube among the 2^m distinct hypercubes. We use a 12-bit addressing table of 4096 hypercube identifiers. Then the truncation reduces the 128-bit *Z* value to a 12-bit address $\mathrm{Hc}_k=(z(P_i)\gg(ed-m))$, where $\gg$ is the bit-shifting operator. This table contains also, for each hypercube Hc_k, data that allow us to retrieve directly all the points of this hypercube. For instance, the table contains the index of its first point and the number of points belonging to the hypercube.

The hypercube identifier of P_i makes it possible to identify a restricted set of chained points into which the new point must be inserted. This insertion is made between the two points whose *Z* values frame $z(P_i)$. We use the index i_j to refer to the *j*'th point in the list. For example, in Fig. 7 Hc_1 is the hypercube identifier of P_i. To find the position

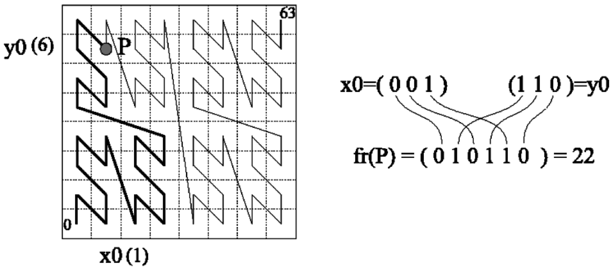


Fig. 6 *Z*-value calculation by interweaving bits.

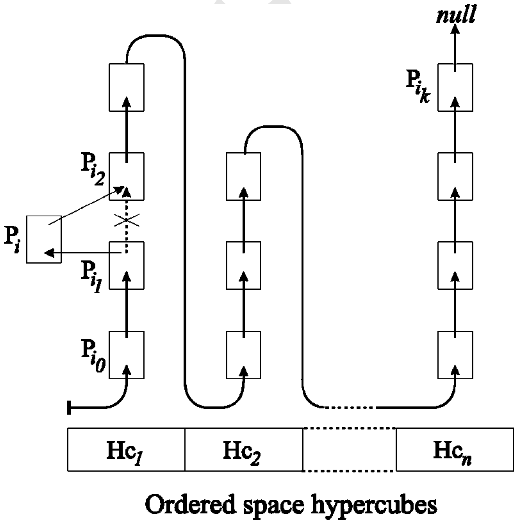


Fig. 7 Insertion of a new point in the linked list.

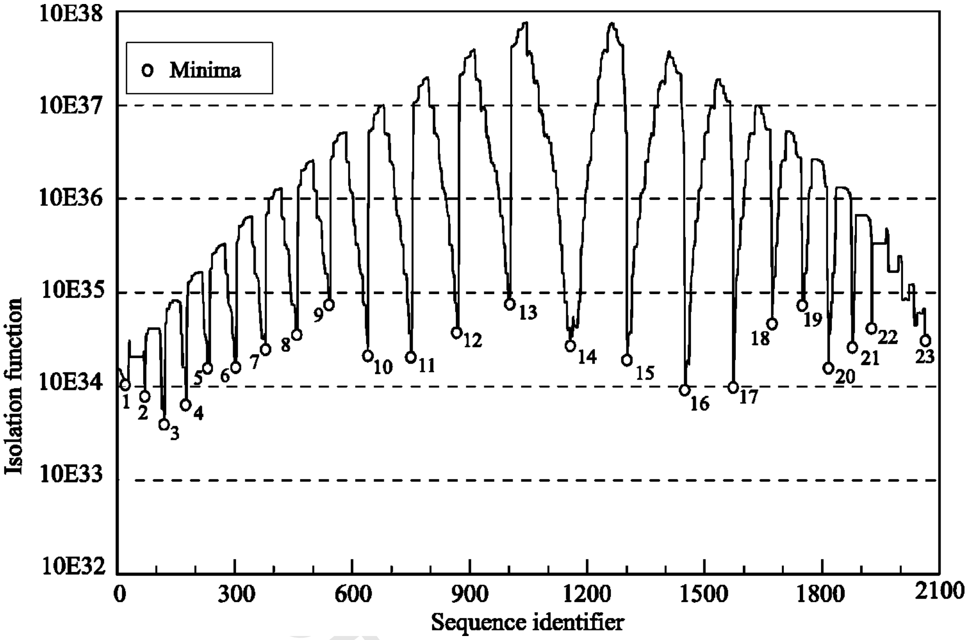


Fig. 8 Isolation function and cluster identification by the local minima of the isolation function.

of P_i in the list, we go from the first point stored in Hc_1 to the next ones, and we compare the rank $z(P_i)$ with the Z value of each point met in Hc_1. Once the position of P_i is found in the list, we update, in the general case:

- the value $a(i_1)=i$ of the preceding point P_{i_1} to point toward the index of the new point P_i
- the value $b(i_2)=i$ of the next point P_{i_2} to point toward the index of the new point P_i
- the values $a(i)=i_2$ and $b(i)=i_1$ of the new point to point toward the points that frame it, P_{i_1} and P_{i_2}.

2.4 *Motion Clustering*

2.4.1 *Off-line analysis*

The interest of the off-line calculation we present here is to demonstrate the distance-preserving effects of the Z-ordering and to provide an empirical validation of the use of the linked list to extract neighboring points. We introduce a mathematical function, the *isolation function*, which is in inverse relation to the density of point clusters along the Z curve. The isolation function $I_\Delta(j)$ is defined so as to quantify the average dispersion of Δ successive points on the linked list. Its value is deduced from the successive neighbors on the fractal space-filling curve by the relation

$$I_\Delta(j)=z(P_{a_\Delta(i_j)})-z(P_{i_j}), \tag{4}$$

where $a_\Delta(i_j)$ is the index of the Δ'th point preceding the j'th point on the linked list. This function provides a simple way to demonstrate the distance-preserving effect by means of tracking the local minimum peaks due to the clusters. If the value of $I_\Delta(j)$ is in inverse proportion to the local density of the j'th chained sequence of points, then a local minimum for one sequence of Δ successive points on the fractal chaining should correspond to a cluster in the space, thus providing a way to identify it.

To verify this hypothesis empirically, we have calculated I_Δ (Fig. 8) from the envelope of a 500-m-distant smoke, near enough to fill a spot of approximately 200 pixels. The maximum extension of the envelope is about 40 pixels in the mean direction of the smoke. The data were processed from a 32-s-duration sequence of 100 images. We used an embedding space of dimension $d=16$, where each axis represents an $e=8$-bit gray-level; thus the Z values varied from 0 to 2^{128}. To represent such a 128-bit number we have used an array of four long integers. We have fixed the parameter $\Delta=40$ to be equal to the maximum extension of the envelope, as it is an estimate of the number of pixels affected by a puff of smoke during its propagation. We use it as a rough estimate of the cluster size in the linked list.

The graph of Fig. 8 shows the isolation function versus point index along the linked list. We notice the presence of 23 main minima. Each minimum corresponds to a sequence of Δ points, which gives evidence of the presence of a cluster. The huge number of clusters shown by this function is partly due to various puffs, but is mainly due to the redundant repetition of clusters associated to the various phases of the temporal signatures. To verify this assumption, we estimated the number of signatures of each cluster identified by a minimum, by averaging the Δ temporal segments of each cluster (Fig. 9). We notice in that example that their temporal segment evolutions differ in three categories, and are generally delayed so as to correspond to the various phases of three different propagation fronts of smoke puffs. We also notice that the various phases of each temporal signature are consecutive along the chain. The first temporal signature is displayed through the minima 1 to 9, the second through 10 to 13, and the third through 15

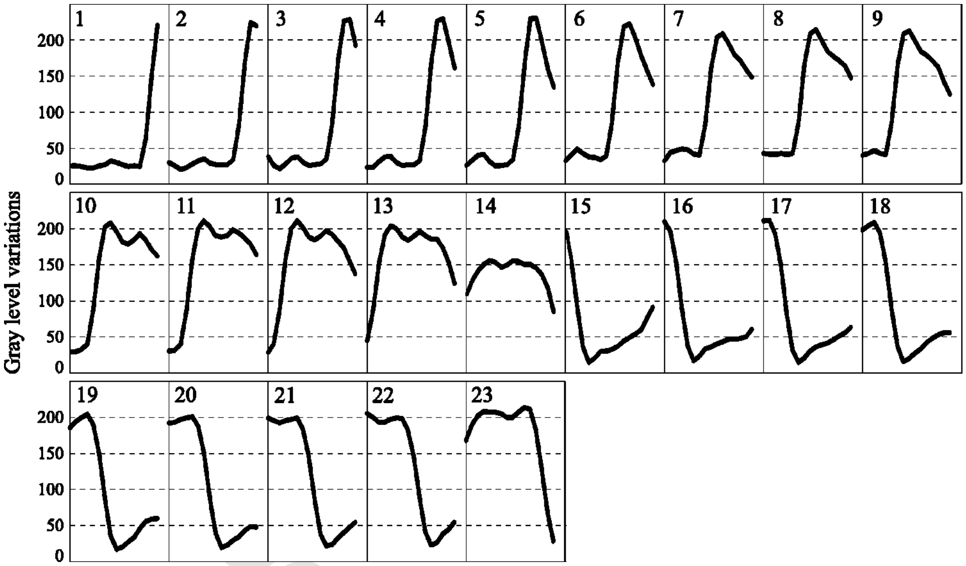


Fig. 9 Average temporal signatures of clusters identified by the local minima of the isolation function.

to 23. This is due to the *Z* ordering, as a delay of one phase corresponds approximately (for high-order bits) to a one-bit shift (multiplication or division by 2) of the *Z* value.

For our off-line process, we suppose that it is possible to identify a smoke puff's spatiotemporal trajectory using the space and time coordinates of the data contained into a sequence of Δ points. But this hypothesis is limited in several respects. First, we do not know the real limits of the cluster of a sequence, so we have to deal with sequences of the linked list that may contain more than one cluster or parts of clusters. Second, we use space-filling curves that do not totally preserve distances through linear mapping, particularly in the case of the *Z* curve mapping, which generates artifacts (Fig. 10).

These limitations explain the jumps encountered in *Z* ordered sequences, which contain successive features that may be rather different. Figure 11 illustrates a cluster of $\Delta=40$ points that is associated with local minimum 5 of the isolation function of Fig. 8. The various temporal signatures associated to the extracted points of this cluster are plotted in Fig. 11 in increasing order of their *Z* values. It is to be noticed that the succession of temporal features is not continuous, as sometimes little clusters of the same continuous signature are inserted into the larger one, for example from minima 10 to 18. Nevertheless, these artifacts may be acceptable if we do not need precise delimitation of clusters, but only extraction of redundant signatures. The temporal segments in Fig. 11 are identical-looking, and so we can extract a specific temporal signature from this sequence of segments. The temporal segments that present the best correlation with the average signature are plotted in bold lines. The other temporal segments, which are correlated but differ slightly from the average signature, are interspersed within the sequence.

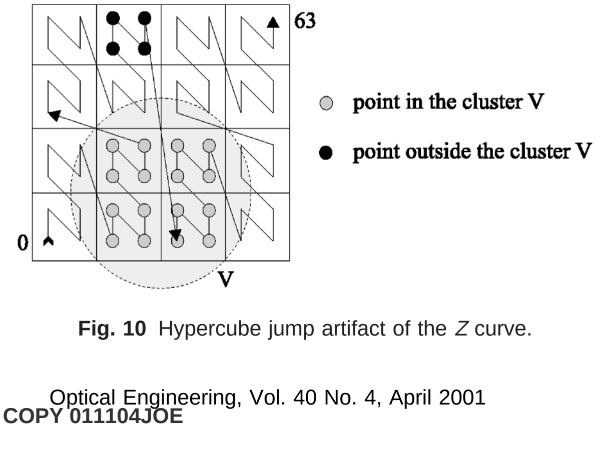


Fig. 10 Hypercube jump artifact of the *Z* curve.

Despite this mixing between various temporal segments in the same sequence, trajectories may be identified with a unique cluster analysis. For recovering the cluster's points of the *j*'th sequence of the linked list $P_{i_j},P_{a_1(i_j)},\ldots,P_{a_\Delta(i_j)}$ we consider their spatiotemporal coordinates:

$$(x_{i_j},t_{i_j}),(x_{a(i_j)},t_{a(i_j)}),\ldots,(x_{a_\Delta(i_j)},t_{a_\Delta(i_j)}) \qquad (5)$$

for the horizontal pixel positions in the picture, and

$$(y_{i_j},t_{i_j}),(y_{a(i_j)},t_{a(i_j)}),\ldots,(y_{a_\Delta(i_j)},t_{a_\Delta(i_j)}) \qquad (6)$$

for the vertical ones. Here (x_{i_j},y_{i_j}) is the pixel position in the image that produces the luminosity variations of P_{i_j}.

The smoke we are studying in this work presents a preponderantly horizontal orientation. Figure 12 shows the horizontal displacement of two puffs of smoke associated to cluster 5. The alignments obtained in Fig. 12 imply a constant propagation speed of the extracted puffs. These two distinct trajectories are associated to the two slightly different temporal segments differentiated in Fig. 11. More generally, all the diagrams *x(t)* and *y(t)* associated to the various clusters present this linear disposition. These results show various motions of the puffs of smoke, tracked at various phases by their temporal signatures. This redundancy of data is useful to compensate the cases where cluster trajectories do not have enough points to identify them or to calculate their parameters precisely.

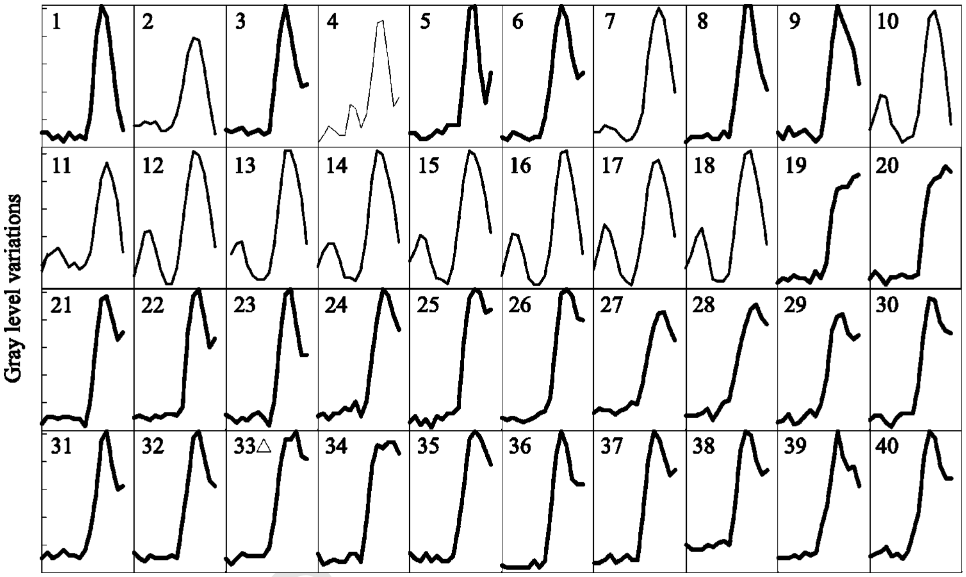


Fig. 11 Temporal segments of the point cluster identified by local minimum 5.

Figure 13 shows results for other smoke puffs and concurrent phenomena such as a cloud and a wind-tossed tree. We notice a very significant difference between the tree results [Fig. 13(c)] and the other ones [Figs. 13(a),13(b)] which give evidence of motion along a trajectory. This result argues in favor of an elementary motion criterion, which is used in our real-time algorithm (see next section). The similar alignments in Figs. 13(a) and 13(b) show that it is not possible to distinguish a smoke and a cloud with the analysis of a single cluster.

For the sole purpose of validation, we used an off-line process to make a cumulative analysis of the set of local minima that allows us to extract the main trajectories inside a spot of smoke. Figure 14 shows that the angles of the motions calculated from the set of local minima are distributed between extremes that correspond to the smoke envelope. The lines' lengths represent the numbers of pixels crossed by the temporal signature extracted. The variability of these angular directions demonstrates the complexity of the puffs' motion inside the spot of smoke which itself has a global motion direction. The movements of the puff of smoke are due to the thermal and turbulent phenomena connected with the wind. This causes the observed large diversity of the detected motions. However, this characteristic is not sufficient for a perfect identification of a smoke source, because the extreme angles vary greatly with the smoke and the wind conditions. Other criteria are necessary to make an efficient smoke detector. In the next section, we present more efficient, real-time calculation criteria, which are actually used by our system.

2.4.2 *Real-time detection*

The objective of our real-time processing algorithm is to provide a diagnosis based on cumulative motion estimation at each sampling time. Calculating the isolation function

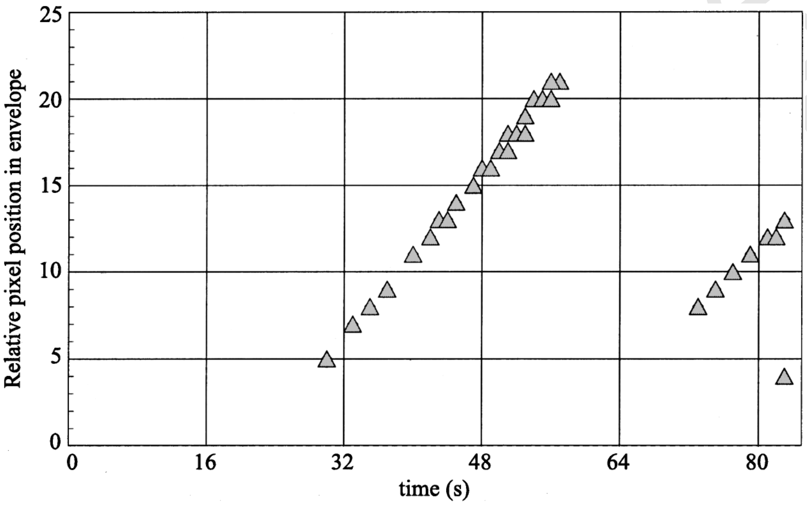


Fig. 12 Horizontal displacement of the temporal segment of Fig. 11.

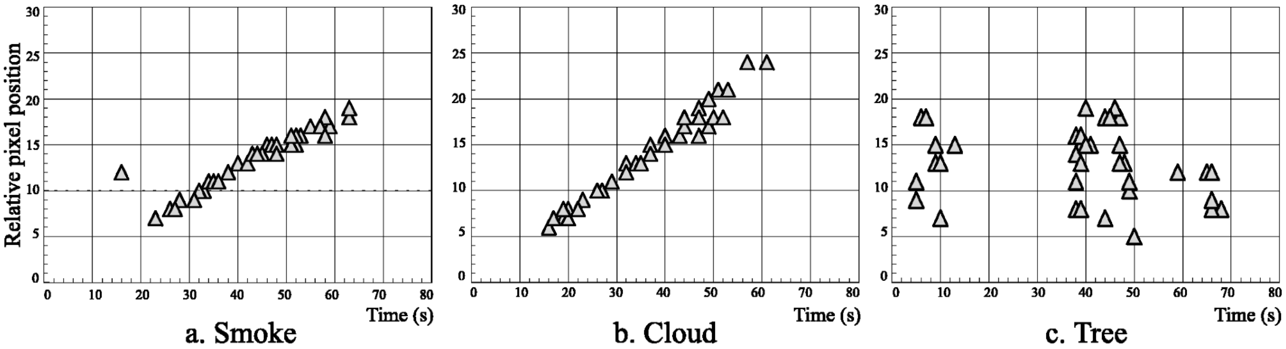


Fig. 13 Motion detection from several dynamical envelopes of natural phenomena.

and extracting its peaks and clusters, as described earlier on, could provide this motion estimation. But this calculation presupposes that enough data have already been embedded. Also, we do not have an efficient method to refresh the isolation function at each sampling time without scanning all the linked list points. We propose here a new algorithm that is more adapted to online processing and does not use the isolation function.

At each sampling time, temporal signatures of the last $d=16$ gray-levels of each pixel are embedded and chained into the linked list of the envelope. Then comparisons are made between each embedded point and the other points in the immediate neighborhood of the list. The number Δ of comparisons is no longer fixed and is limited only by the calculation time available for the process (which is necessarily less than the image sampling period, 0.33 s). As this calculation time depends on the number of pixels that are embedded at each sampling time, Δ varies in inverse proportion to the number of pixels of the envelope. A minimum $\Delta_{\min}$ and maximum $\Delta_{\max}$ are used so as to keep Δ within reasonable bounds.

For each new embedded point, we calculate the histogram of the instantaneous velocities associated to the Δ pairs of points formed with this point and its Δ immediate neighbor points of the list ($\Delta/2$ before this point, $\Delta/2$ after); knowing the space and time coordinates of the neighbor points, the quotients of their differences from those of the new embedded points are calculated and added to a velocity distribution histogram (IVH). Figure 15 shows representative results calculated for the envelopes of a cloud, a smoke, and a wind-tossed tree, the smoke being at a distance of 4 km. We notice no significant results in the case of the tree and in contrast a very well-defined peak for the cloud.

When examining various types of smoke results, we always observe a spreading histogram with a certain variability of the standard deviation, but that always keeps above that of the cloud. Conversely, we consider that there is no motion detected if the value of this instantaneous standard deviation (ISD) is above a certain threshold, this parameter being included in the initialization file of the system. If any motion is detected, the velocity of the maximum is then incremented into a cumulative velocity histogram (CVH). At each sampling time, the cumulative histogram is analyzed to calculate criteria to distinguish smoke from other phenomena.

In the case of smoke and clouds, we observe that the CVH [Figs. 15(a),15(b)] has the same shape as the IVH, but is smoothers and the standard deviation (CSD) is stable. When the CSD is less than a minimum deviation (CMD), we invalidate smoke detection for this sampling time (Fig. 16). The use of the CMD criterion poses a problem only in the rare case when certain smoke envelopes are stretched by a very stable wind, so that their velocity distribution looks like a cloud's typical one. But that cannot last, and the only consequence is that the smoke detection time is increased.

In the case of wind-tossed trees (Fig. 15c) and many other phenomena, the cumulative histogram is most often rather poor in data, because motions are detected in the case of pointlike situations and for precise pixels only. A second criterion is then the minimum average energy (ME)

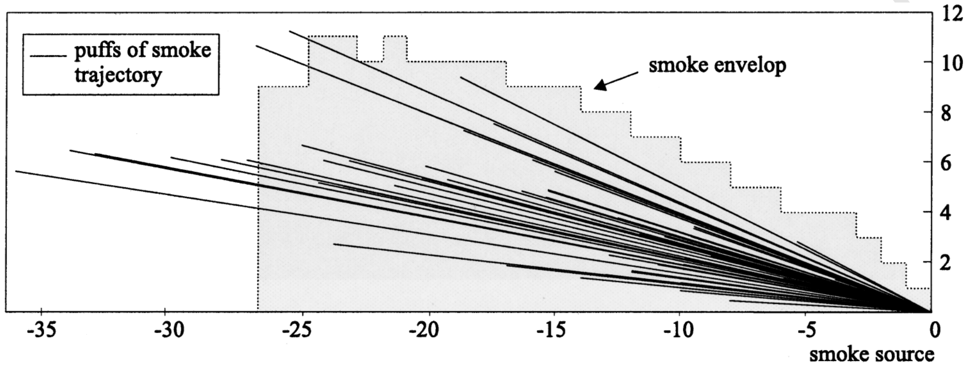
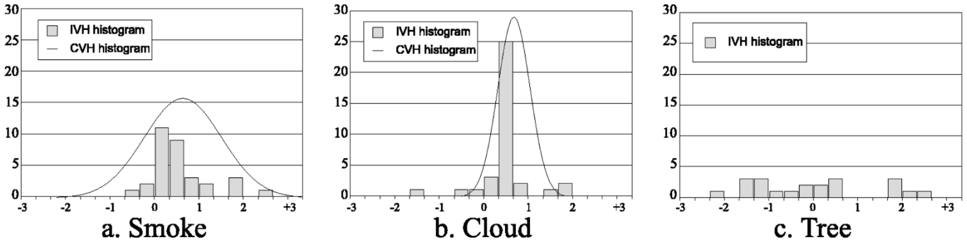


Fig. 14 Extracted trajectory inside a spot of smoke.



a. Smoke

b. Cloud

c. Tree

Fig. 15 Instantaneous velocity histograms (IVH) for one point neighborhood and cumulative velocity histograms (CVH) for all the point neighborhoods.

of the cumulative histogram per embedded point. This criterion is the most selective one, as it separates smoke from a lot of various pseudo-dynamic envelopes that are not eliminated by the envelope extraction system

In addition to CMD and ME, other criteria are used, for example, those based on the shape and on the smoothness of the cumulative histogram; but they are more questionable and have less generality. Among the other criteria is the sense derived from the angular distribution (top or bottom); the smoke always moves to the top. Then instantaneous and cumulative angular histograms are also calculated and used like the velocity.

3 Discussion and Conclusion

One of the main interests of the temporal embedding algorithm we have presented in this paper is that it is sensitive enough to detect local motions by means of only a few pixels. So it is particularly adapted to the analysis of small dynamic deformable objects with multiple transitory motion paths, such as small envelopes of smoke. The four processing steps described before lead first to a local-motion diagnosis at the level of one pixel, and second to a global-motion diagnosis at the level of the whole envelope. These diagnoses are used for a complete smoke identification.

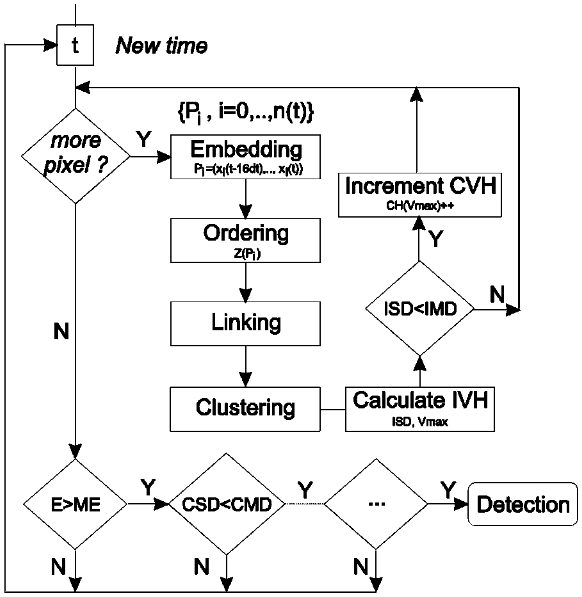


Fig. 16 Real-time detection algorithm using the four steps: temporal embedding, fractal ordering, linking points, and motion clustering.

The smoke identification uses a cumulative analysis of the instantaneous motion data. We have found that the most efficient data for smoke identification are the velocity distribution in the envelope, whose energy, or average number of instantaneous motion diagnoses per embedded point, is higher than the energy of most other landscape phenomena except clouds. But for clouds, the standard deviation of velocity distribution is generally lower than for smoke. Thus our main criterion for smoke detection is based on the analysis of the velocity distribution, using a minimum energy threshold and a minimum standard-deviation threshold. We have shown that our method is able to provide direct access to useful correlated data when a fast process is needed. The fractal embedding method does not use any *a priori* model or parameter in the calculation for the extraction of a result. For example, no threshold is used to define the gray-level variations above which the presence of a smoke puff may be suspected, and this is an advantage, because smoke can be sometimes dense, sometimes nearly transparent, and can produce slow, fast, and more often mixed gray-level variations.

It is of interest to point out the main aspects of our method from a general point of view, by examining its different processing steps as follow:

- First step: gearing the embedding type to the information sought. Temporal embedding shows various fleeting motions within a complex dynamic spot. In contrast, other embedding methods would bring out the most stable features of any complex object.
- Second step: calculating the fractal rank through *Z* curve indexing. *Z* curve indexing permits very fast calculation of the fractal rank through transposition of a matrix of bit coordinates. This very simple operation can be carried out by a specialized low-level specialized routine.
- Third step: fractal chaining. By truncating the fractal rank so as to classify all possible ranks within a limited number of classes, thus not exceeding the memory of ordinary computers, each new point may be chained inside a hypercube containing a very small number of already chained points. This limits the number of rank comparisons to be carried out, and therefore the computing time.
- Fourth step: extracting cluster data. From the minima of the isolation function, or more simply from the near neighborhood of any incoming point, the presence of a cluster can be detected, and then the relevant data associated to it can be analyzed.

One should be especially aware that a definite advantage of fractal chaining and clustering as they appear in the third and fourth steps is to reduce the computing time considerably. This is of paramount importance when processing large amounts of data, such as image sequences, in particular in the case of real-time computing. One of the conditions for the embedding method to be used is for enough correlation to exist to generate point clusters. This condition has to do with the redundancy of the information, which is due to the piling up of consecutive images of an object and to the multiplication of the measuring points constituted by all the pixels of the phenomenon's spot. That is the reason why temporal embedding is particularly well suited to an illustration of the fractal embedding method.

References

1. P. Bouthemy, ''Motion segmentation and qualitative dynamic scene analysis from an image sequence,'' *Int. J. Comput. Vis.* **10**(2), 157–182 (1993).
2. D. Terzopoulos, A. Witkin, and M. Kass, ''Constraints on deformable models: Recovering 3D shape and nonrigid motion,'' *Artif. Intel.* **36**, 91–123 (1988).
3. F. Takens, *Detecting strange attractors in turbulence*, in *Dynamical Systems and Turbulence*, Lecture Notes in Mathematics 898, pp. 366–381, Springer-Verlag (1981).
4. R. Agrawal, K. I. Lin, H. S. Sawhney, and K. Shim, ''Fast similarity search in the presence of noise, scaling and translation in time series databases,'' in *Proc. 21st Int. Conf. on Very Large Databases*, pp. 490–501, Zurich, (1995).
5. C. Faloutsos, M. Ranganathan, and Y. Monolopoulos, ''Fast subsequence matching in time-series databases,'' in *Proc. SIGMOD*, pp. 419–429 (1994).
6. G. H. Cha and C. W. Chung, ''Object-oriented retrieval mechanism for semistructured image collections,'' in *Proc. 6th ACM Int. Multimedia Conf. (ACM Multimedia '98)* (1998).
7. F. Korn, N. Sidiropoulos, C. Faloutsos, and E. Siegel, ''Fast nearest-neighbor search in medical image databases,'' in *Int. Conf. on Very Large Data Bases*, Bombay (1996).
8. A. Guttman, ''R-trees: a dynamic index structure for spatial searching,'' in *Proc. 1984 ACM SIGMOD Int. Conf. on Management of Data*, pp. 45–57 (1984).
9. N. Roussopoulos, C. Faloutsos, and S. Timos, ''The R+-trees: A dynamic index for multi-dimensional objects,'' presented at 13th Int. Conf. on Very Large Databases, Brighton (1987).
10. T. Brinkhoff, H. P. Kriegel, and B. Seeger, ''Efficient processing of spatial joins using R-trees,'' in *Proc. SIGMOD* (1993).
11. J. A. Orenstein, ''Spatial query processing in an object-oriented database system,'' in *Proc. ACM SIGMOD*, pp. 326–336 (1986).
12. J. A. Orenstein, ''Redundancy in spatial databases,'' in *Proc. ACM SIGMOD Int. Conf. on the Management of Data*, Portland, OR (1989).
13. J. A. Orenstein and T. H. Merret, ''A class of data structures for associative searching'' in *Proc. SIGACT-SIGMOD*, pp. 181–190, Waterloo, Canada (1984).
14. C. Faloutsos, ''Gray codes for partial match and range queries,'' *IEEE Trans. Softw. Eng.* (1987).
15. H. Sagan, *Space-Filling Curves*, Springer-Verlag (1994).
16. C. Faloutsos and S. Roseman, ''Fractals for secondary key retrieval,'' in *Proc. ACM Conf. on the Principles of Databases Systems*, pp. 247–252 (1989).

Philippe Guillemant is an ECP engineer working as a researcher at CNRS in France. His main interest is the analysis of complex and dynamic images for artificial vision applications. He has developed different apparatus, using neural network and fractal analysis methods, that has been utilized in industry. His main contribution is an ocular movement analyzer, the videonystagmograph, used for medical exploration of vertigo.

Jérôme Vicente received his PhD from IUSTI in France. In his thesis he developed a forest fire detector using CCD captors. His main interest is the characterization of complex or chaotic dynamical phenomena using signal or image processing. These techniques, based on fractal analysis, are used as well for medical applications in the characterization of sleep stages and the location of epilepsy crisis sources.