Open access • Journal Article • DOI:10.1016/S1077-2014(02)00150-X

Length-speed ratio (LSR) as a characteristic for moving elements real-time classification — Source link

Miguel A. Fernández, Antonio Fernández-Caballero, María T. López, José Mira

Institutions: University of Castilla–La Mancha, National University of Distance Education

Published on: 01 Feb 2003 - Real-time Imaging (Academic Press)

Topics: Object detection

Available online at www.sciencedirect.com
SCIENCE DIRECT®
Real-Time Imaging 9 (2003) 49–59
www.elsevier.com/locate/rti

Length–speed ratio (LSR) as a characteristic for moving elements real-time classification

Miguel A. Fernández[a], A. Fernández-Caballero[a,*], María T. López[a], José Mira[b]

[a] Departamento de Informatica Escuela Politecnica Superior, Universidad de Castilla-La Mancha, 02071 Albacete, Spain
[b] Departamento de Inteligencia Artificial, UNED, Madrid, Spain

Received 20 December 2001; received in revised form 1 July 2002; accepted 5 November 2002

Abstract

In this article, the length–speed ratio (LSR) is proposed as a basic characteristic for the real-time detection of moving objects. We define the LSR of a uniform moving zone as the relation between its length in the direction of motion and the speed of this motion. For a given zone of the image with uniform gray level (or patch), the greater its length in the direction of motion and the smaller its speed, the greater its LSR. A moving element is generally composed of various zones of uniform gray levels (or patches), which move with the same speed but which have different lengths in the direction of motion and which therefore have a characteristic set of LSR values. In this article, this "LSR footprint" is proposed as the basic characteristic for the detection and subsequent classification of moving elements in image sequences. The problem of detecting a moving element in a sequence of images is transformed into the recognition of a pattern on a static image, namely the LSR footprint. We also specify how to obtain this characteristic in real time, we discuss its invariants and we consider the cases for which LSR detection of movement is applicable. We also present its use in some significant examples and we compare it with other methods applicable to similar computational problems.

1. Introduction

The processing of image sequences is a complex task [1–13]. Firstly, the volume of information is very large: if we consider the typical figures for a television signal, images of 512×512 pixels and a new image every 40 ms, real-time treatment of this data means processing more than 6 Mbytes per second. Secondly, the objectives are more ambitious than those of the processing of one single image. Clearly, the processes can be applied to a single image (filtering, threshold, edge extraction, etc.). But others, such as those designed for the analysis of movement or for the analysis of 3D spatial relations from a sequence of images, for example, are also applicable. A great deal of techniques to process image sequences in real time have been introduced so far [14–17].

In this article, we discuss techniques for the analysis of movement in sequences of images and, more concretely, for the detection of moving elements. The techniques used in this area can be divided into two classes. The first class processes one complete image at a time and is typically a sequential process, identifying characteristic elements which can be re-identified in subsequent images; for these tasks, algorithms based on correlation, clustering, chain-coding, etc. are usually employed. The principle problem with these algorithms is that they consume an excessive amount of processing time and require expensive hardware. They are also difficult to implement in real time and are very sensitive to variations in the results of the low-level processing. The second class processes each individual point of the image along a sequence of images, involving multiple parallel processes (one for each pixel of the image). The classical algorithms of this type are those based on gradient analysis, the most well known being the optical flow model [18–20]. The problem with algorithms of this class is that the calculations are again very costly in hardware [21] and, in addition, their implementation in a real-time context requires many simplifications. Other well-known methods of this type are those based on image difference or on accumulated image difference [20]; both methods require a reference image and both are designed for use with a small sequence of images

*Corresponding author. Tel.: +34-967-59-92-00; fax: +34-967-59-92-24.
E-mail address: caballer@info-ab.uclm.es (A. Fernández-Caballero).

1077-2014/03/$ - see front matter
PII: S1077-2014(02)00150-X

rather than an indefinite sequence. Of all these mechanisms, the accumulated image difference is the most similar to the mechanisms used in the work reported on in this article. However, the mechanisms used here differ fundamentally from those of the accumulated difference methods. The latter increment the difference memory on finding a difference with the reference image, whereas, in our case, as will be seen later, the permanence memory is incremented when there is no variation with respect to the previous image. In general, in image sequence processing (or video processing), there are few characteristics that are easy to extract while contributing at the same with robust, valuable and discriminatory information. In this article we propose the use of such a characteristic for the detection of moving elements in sequences of images. As we will show later, it is robust, has good invariants and has good discriminatory properties making it well suited to the real-time detection and classification of moving elements using simple hardware.

2. Definition of the LSR characteristic

We define the length–speed ratio (LSR) of a moving image zone (or patch) as the ratio between its length in the direction of movement (L) and its speed (V) (see Fig. 1).

This descriptor can be interpreted as a measure of the permanence of a given zone (or patch) over the sampling point. In other words, the LSR value measures the time that a certain element of the image activates a particular coordinate in the array of sensors. The greater the length of a uniform gray-level patch and the smaller its speed, the greater the permanence it generates and the higher its LSR value.

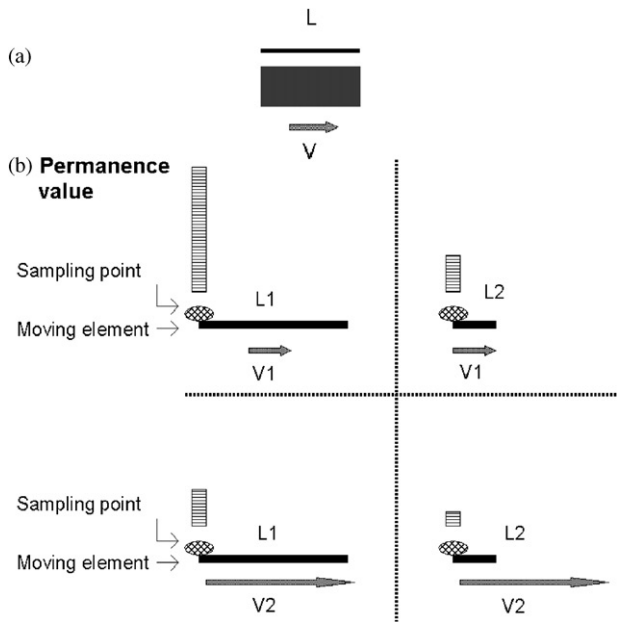

Fig. 1. Illustration of the LSR characteristic behavior: (a) Definition; (b) different moving elements together with the permanence value they generate at the sampling point.

In the case of points at which the image is static, there is no substantial modification of gray levels so that the permanence value charges up to saturation and an LSR value of zero is generated.

The rest of this article demonstrates the efficiency of this technique through the presentation of its calculation method, through the study of its invariants and through the discussion concerning its similarities to biological computation systems and that concerning the rest of its properties. The best way to interpret it is to turn to its basic geometric definition and its computational meaning.

2.1. Permanence memories

Permanence memories work on television images binarized (1 bit digitized) according to gray-level thresholds or by other methods. For each sensor point (pixel) P_{ij}, where i (respectively j) ranges between 1 and n (m), being n (m) the number of columns (rows), we denote its gray level in frame t of the sequence by $GL_{ij}(t)$ and the value corresponding to the binarization for that point in that frame by $IN_{ij}(t)$. The permanence memories define a map of data items for each frame t. The value in frame t of the permanence memory PM_{ij} [22], associated to point P_{ij}, is defined in terms of its value at time $t-1$ and the binarised input $IN_{ij}(t)$, as follows (the values $PM_{ij}(t)$ being referred to as permanence values):

$$PM_{ij}(t) = \begin{cases} PM_{ij}(t-1) + S\,(\text{up to } MAX) \\ \qquad \text{if } IN_{ij}(t) = 1, \\ PM_{ij}(t-1) - R\,(\text{down to } MIN) \\ \qquad \text{if } IN_{ij}(t) = 0, \end{cases} \tag{1}$$

where S (respectively, R) is the constant by which the permanence memory is incremented (decremented), MAX is the saturation constant, i.e. the maximum possible permanence value, and MIN is the total discharge constant, i.e. the minimum possible permanence value. The values of parameters S, R, MIN and MAX will be explained later on. An example of the behavior of the permanence memories can be studied from Fig. 2. In this figure it can be appreciated how the permanence memory values grow to saturation point when the object is over their associated pixels and slowly decrease when the object has passed.

2.2. LSR characteristic extraction process

In what follows, we see that computationally the local property LSR is basically an application of permanence memories [22]. The LSR extraction process begins with

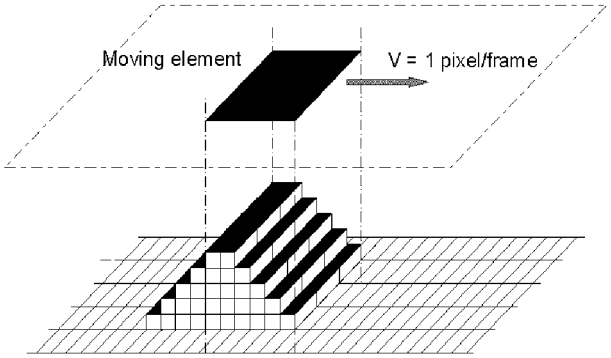

Fig. 2. Behavior of the permanence memories ($S=1$, $R=1$).

the binarization of the image, giving rise to an array of values $IN_{ij}(t)$ one for each point P_{ij} in the frame t, according to the following procedure:

(1) the spectrum of possible gray values is divided into bands;
(2) if gray level GL_{ij} recorded at sampling point P_{ij} lies in the same gray-level band GLB_{ij} as that recorded at P_{ij} in the previous frame, *t-1*, $IN_{ij}(t)$ remains 1, otherwise it is set to zero. Thus, for each sampling point in each frame t, we have

$$IN_{ij}(t) = \begin{cases} 1 & \text{if } GLB_{ij}(t) = GLB_{ij}(t-1), \\ 0 & \text{if } GLB_{ij}(t) \neq GLB_{ij}(t-1). \end{cases} \quad (2)$$

In the following we define the value of the charge decrement R to be $R = MAX$. This setting is performed to adjust the permanency effect to obtaining the LSR characteristic, as it will be explained later on. Finally, we obtain the LSR characteristic: for each sampling point P_{ij} and for each frame t, an $LSR_{ij}(t)$ value, is generated. In any given frame t, the set of values $LSR_{ij}(t)$, where i ranges between 1 and n, and j ranges between 1 and m, is characteristic of the moving element which created it and is denoted its LSR footprint. The calculation of the LSR value $LSR_{ij}(t)$ at each sampling point P_{ij} and in each frame t is carried out using the following algorithm:

$$LSR_{ij}(t) = \begin{cases} PM_{ij}(t-1) & \text{if } IN_{ij}(t) = 0, \\ 0 & \text{if } IN_{ij}(t) = 1. \end{cases} \quad (3)$$

That is, the value $LSR_{ij}(t)$ is set to zero when there has been no substantial variation in the gray level of the point P_{ij} between the previous frame *t-1* and the current frame t and takes the value $PM_{ij}(t\text{-}1)$ (the charge value of the permanence memory of the point P_{ij} in the previous frame) in the case where there has been such a variation.

The value PM_{ij} associated to each point P_{ij} charges up progressively while the corresponding input GL_{ij} remains in the same gray level band ($IN_{ij}(t)=1$) and discharges completely generating an LSR value equal to $PM_{ij}(t\text{-}1)$ when the input GL_{ij} changes from one band to another ($IN_{ij}(t)=0$).

We now determine the range of LSR values that a system based on this property can measure. Ignoring the sensor characteristics in terms of camera optics and speaking in terms of V (velocity, in pixels per frame, of the patch on the sampling field constituted by the image of the moving object), L (length, in pixels, of the patch on the sampling field constituted by the image of the moving object), and t (frames), the limitations on the motion which the system is capable of detecting are as follows:

(1) The system is not capable of detecting movements with an LSR of less than one. This means that it will not be able to detect the LSR created by patches that do not remain over any sensor point for longer than one frame. This condition is defined by the equation:

$$V_t < L. \quad (4)$$

(2) The system does not distinguish movement of patches that activate the same sensor point for longer than MAX/S frames since the permanence values for the pixels that such patches pass over reach saturation so that it is not possible to calculate their true LSR value. This condition is defined by the equation:

$$L < V_t \frac{MAX}{S} \quad (5)$$

Changes in the LSR greater than the saturation value or lower than zero are therefore not distinguishable by the system. The range of LSR values, which the system can differentiate, is thus given by the equation

$$Vt < L < V_t \frac{MAX - MIN}{S}. \quad (6)$$

We now describe the nature of the parameters used, as well as the values adequate to each application. Table 1.

Values of parameters *S, R, MAX* and *MIN* have to be fixed according to the applications characteristics. Concretely, values *MAX* and *MIN* have to be chosen by taking into account that charge values will always be between them. The value of S defines the charge increment interval. Greater values of S allow arriving in a quicker way to saturation. In order to calculate the LSR, the best is to generate a discharge in the stored permanency value of a pixel where motion is detected. To achieve this goal, the best is to define R as *MAX*. A pixel is this way discharged where motion is detected. Results offered in this paper have been obtained by used the following values: $MAX=255$, $MIN=0$, $S=1$ and $R=255$. We now explain the reason for this choice: Table 2.

Table 1
LSR parameters description

Parameter	Description
MIN	Discharge value.
MAX	Saturation value. *MAX–MIN* defines the charge capacity; associated to the number of steps that may exist between both values.
S	Charge increment (see formula (1), (5) and (6)). A high value of *S* leads to a great charge velocity, and therefore it reduces the number of steps between discharge and saturation. A low value of *S* leads to a reduced charge velocity, and therefore the number of steps between discharge and saturation augments.
R	Charge decrement (see formula (1)). A high value of *R* leads to a great discharge velocity, and therefore it reduces the number of steps between discharge and saturation. A low value of *R* leads to a reduced discharge velocity, and therefore the number of steps between discharge and saturation augments.

Table 2
LSR parameter values explanation

Parameter value	Explanation
$MIN = 0$	This way, we define the widest working zone, adequate for the charge values presented next; and, this a value easy to compute with.
$MAX = 255$	This is also a value easy to compute with. It is sufficient for the moving elements lengths (L = 20–50) and velocities (V = 1–4) used in our examples. Notice that by means of this value it is possible to obtain the *LSR* of elements with values up to $L = 254$ and $V = 1$ in 512×512 pixel images.
$S = 1$	We use this value for parameter S to define an adequate number of steps between discharge and saturation (slow charge).
$R = 255$	This way, we discharge all image pixels that do not correspond to any element of the scene (immediate discharge).

2.3. Information provided by LSR footprints

As described before, a moving element generates an LSR footprint, on the map of discharges, depending on its speed and on the different zones of uniform gray level of which it is composed. The LSR footprint is, in reality, a pattern that must be classified in order to identify the moving element that created it. However, such an analysis has important advantages over a direct analysis of the image sequence, in particular:

(a) The information concerning static elements has been suppressed.
(b) In each image frame, the LSR footprint is a static pattern containing information about the movement that has taken place in that scene in previous frames.
(c) As the volume of data is much smaller, the moving element can be identified more easily.

The problem of detecting a moving element in a sequence of images has been transformed into the recognition of a pattern on a static image, namely the LSR footprint.

The extraction process for the LSR characteristic is cheap, simple, robust and quick and the information that is obtained is sufficient for the classification of moving elements.

Once the LSR footprint has been obtained, as it is just a static pattern in the current image, it is then possible to choose from different methods of pattern classification. In our case, we have opted for a very simple classification method that we describe later on.

3. Some important facts about the LSR characteristic

3.1. Invariants of the LSR characteristic

Invariants are of great interest [23]. The invariants that the LSR characteristic possesses can be described with the aid of Fig. 3. As this figure indicates, the LSR characteristic is invariant under changes in the distance between the sensor and the moving element. That is to say, a same element at a same velocity generates the same discharge values independently from its distance to the sensor. This affirmation is valid in those moments where the distance to the sensor is constant. When the element comes closer or moves away from the sensor, its discharge values are altered. This statement can be demonstrated carrying out a trigonometric analysis of the problem. It is also invariant under changes in the direction of movement.

Nevertheless, it is not invariant under rotations of the moving element with respect to the direction of movement. These invariants are useful in a multitude of real applications, such as the detection of moving elements in landscape scenes or in industrial scenarios. The lack of invariance under rotations with respect to the direction of movement does not impose many restrictions since in

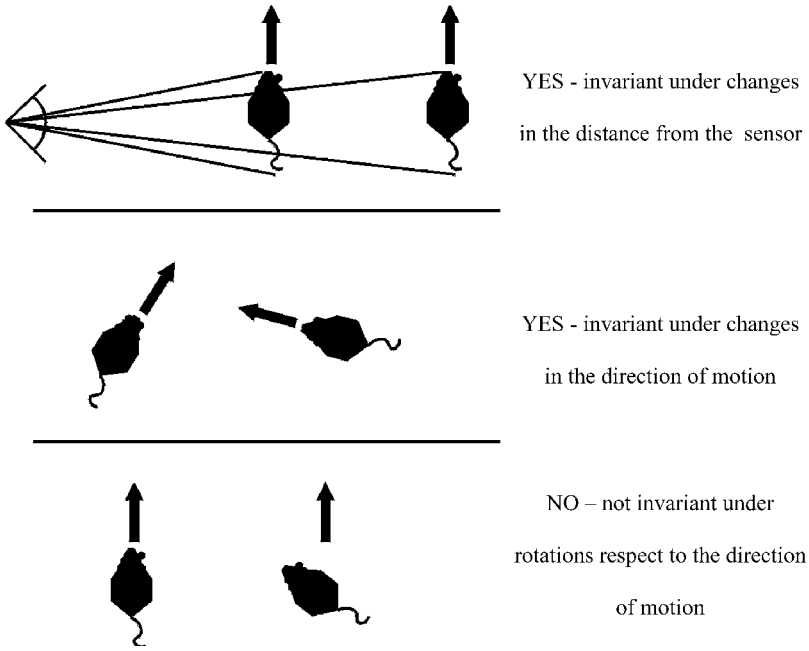

Fig. 3. Invariants of the LSR characteristic.

a very large number of applications the moving elements do not move with these characteristics (generally, the front arrives earlier than the rear). Moreover, this lack of invariance can in fact be used to differentiate different positions of the same element, since an element gives rise to a different LSR footprint if its orientation with respect to the direction of motion is modified.

3.2. Real-time measurement of the LSR characteristic

One of the main advantages of this characteristic is that it can be measured in real time in the whole image of a sequence supplied by a conventional video camera. To obtain the LSR footprint from the binarized image—according to algorithm (2)—it is sufficient to use a look-up table (LUT), a previous-band memory and a simple logic. The real-time calculation of the LSR can be performed by means of the hardware sketched in Fig. 4.

As it can be seen from the figure the following resources are sufficient: an image memory of $n \times m \times 1$ bits to store the values GLB_{ij} for the previous frame $t-1$, a memory with a capacity $n \times m \times 8$ bits to store the values PM_{ij}, and a simple logic capable of implementing algorithms (1)–(3) for the updating of the permanence values. The logic required reduces to a summation operation, a pair of simple logical functions and a multiplex. These operations can easily be carried out in 100 ns between pixel arrivals.

This hardware is sufficient to obtain the LSR value generated for each pixel in real time. At the end of each pixel frame, all LSR values for that frame's pixels will be available.

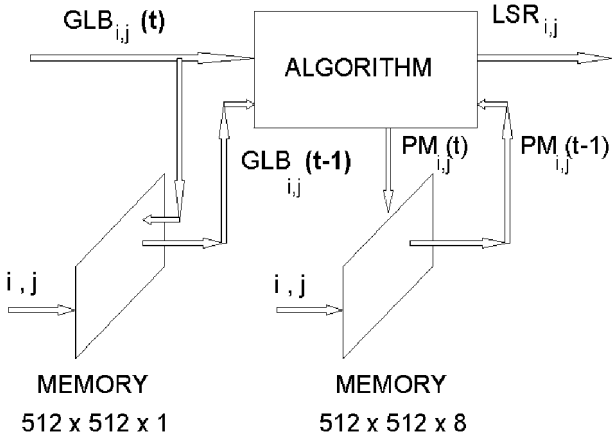

Fig. 4. Hardware for real-time extraction of the LSR characteristic.

3.3. Applicability of the LSR characteristic

In principle, using the mechanisms presented in this article, the LSR characteristic is applicable to the detection of objects or invariant forms whose movement is a plane [24]. This includes the movement of objects on a conveyor belt, the tangential movement of distant targets on landscape or sky backgrounds, as well as movement in image sequences of metropolitan or traffic scenes recorded by observation or security cameras. However, both the LSR characteristic and the permanence memories in general are applicable, using other complementary mechanisms, both to 3D-image analysis and to the analysis of variable forms, the latter generating sequences of LSR vectors.

It is of interest to note that the classification mechanisms described in this article can also be used for the detection of certain movement situations. We refer to scenes in which there is not one moving element moving in a fixed manner but various types of moving elements which can adopt movement configurations characteristic of certain concrete states. Examples of this type of application are found in the detection of rapid or slow circulation of vehicles or the detection of different states of movement of colonies of individuals in a passive or an active state [25].

3.4. Some biological connections of the LSR characteristic

When designing artificial systems which attempt to implement functions very efficiently as in biological systems—this is the case in artificial vision—it may not be essential to find biological mechanisms similar to the artificial ones used in our designs, but to do so is certainly encouraging. In the case of the LSR characteristic under consideration, the amount of encouragement that we obtain by looking at biological systems is significant. We can find important similarities between a multitude of biological processes and this characteristic, the one that perhaps stands out the most being the function implemented by type E neurons of the Pipiens frog [26], which is very similar to the extraction of the LSR. Additionally, the permanence processes discussed in this work are similar to processes of local accumulation of persistent activity at the level of the synapse [27].

4. A classification example based on the LSR footprint

We now describe a process for the classification of LSR footprint created by various different moving elements. It should be made clear that many other processes can be used. We chose this one due to the fact that, as well as being adequate for the applications in which we use it, it is cheap, robust and can be carried out in real time.

Supposing that there is only one moving element in the scene, in each frame of the image the LSR values generated for that element will be available. If this moving element is a stable form that travels with a uniform speed without any variation in its orientation with respect to the direction of movement, it will generate in each frame a LSR footprint whose position changes with time but whose form does not.

On this LSR footprint there are different parameters, both quantitative and morphological, on which to base the classification process. In our case, we have chosen to base the detection on processes that only look for the existence of a particular combination of LSR values in the scene, without carrying out any morphological or other type of analysis of the LSR footprint. We classify the moving elements according to the analysis of the LSR footprint, by considering that a given moving element is identified through the appearance of a given combination of LSR values, without considering in any way the spatial or morphological distribution of the footprint. This implies loss of information and also limits the field of application, but for the applications of interest here it is beneficial since it leads to a large reduction in the computational costs.

We describe the process of classification according to the LSR footprint using Fig. 5. Parts (b) and (c) from this figure show in a generic fashion how the LSR footprint generated by a moving element (Fig. 5a) is obtained. Parts (d) and (e) of the figure show a concrete process that we have used to classify the moving elements in real time with simple hardware based on their LSR footprint. We now describe this process explicitly.

4.1. Obtaining the discharge vector from the LSR footprint

The process that we have implemented is represented graphically in Figs. 5d and e consists of the conversion of the LSR footprint into a discharge vector of k bits and its later classification to identify the moving element that generated it. To carry out this process, the spectrum of LSR values is divided into k bands, each band being associated to one bit of the discharge vector. In the example represented, the value of the LSR is greater than or equal to one and less than or equal to 8. In this case, the bands have been chosen as small as possible, so that each band only covers one possible LSR value. The first band is associated to an LSR value of one, the second to an LSR value of two and so on up to the eight bands associated to an LSR value of eight.

The first step is to construct a histogram in which, for each of the possible bands of LSR values, the number of LSR values in that band in the current frame (Fig. 5d) is indicated, without recording the position of each of these discharges. Next, as explained below, the actual values of the discharge vector are calculated. If the number of LSR discharges in a given band of the histogram is greater than a predefined threshold, the bit corresponding to this band is set to 1, and otherwise it is set to 0. The mentioned threshold has to be defined by a value such that we eliminate those discharge intervals where the only existent discharges are due to noise or to elements that are not of our interest due to their size. In our experience, when working with 512×512 pixel images, we have noticed that a threshold between 5 and 10 is sufficient to eliminate discharges due to noise. On the other hand, when trying to eliminate those discharges due to elements lower in size than a given value, we have to adapt the threshold to the number of

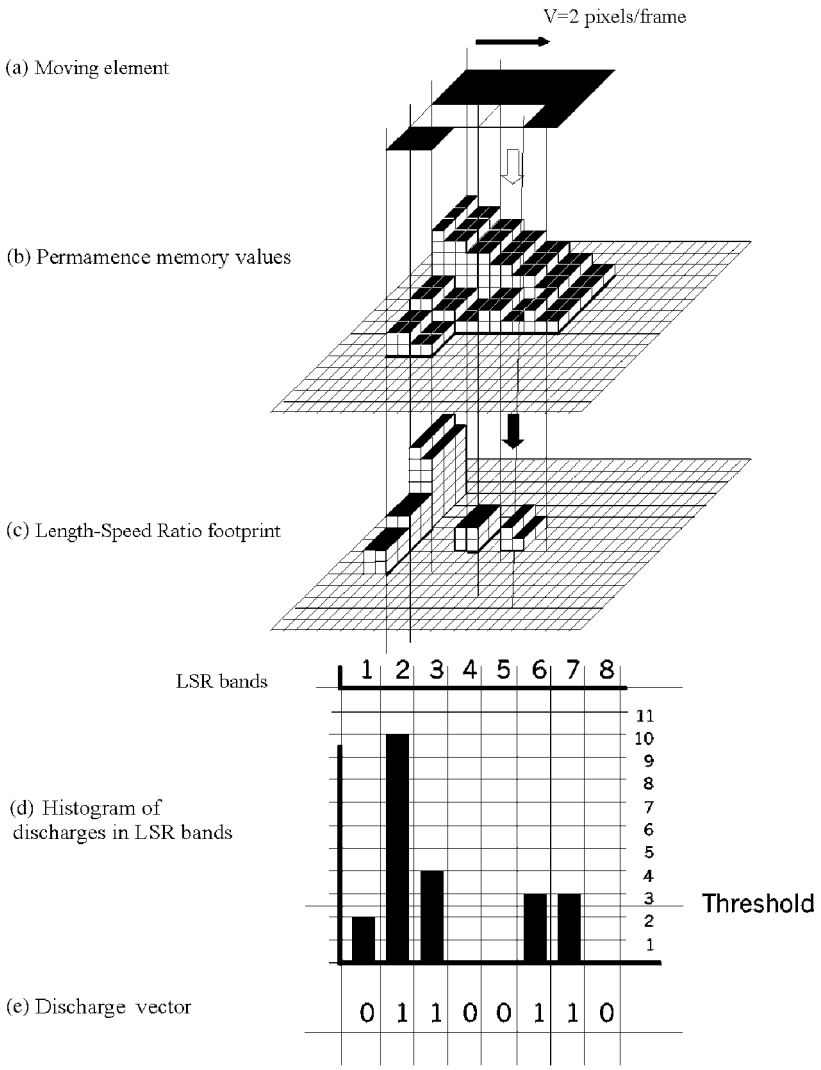

Fig. 5. (a) Moving element; (b) permanence memory values ($S=1$, $R=MAX$); (c) LSR footprint; (d) histogram of discharges in LSR bands; and (e) discharge vector.

discharge points associated to those elements. This way we have generated the set of values referred to as the discharge vector, which basically indicates the LSR bands in which the moving element traversing the image at this moment generates discharges (LSR values). Therefore, in order to detect a moving element, first we generate its LSR footprint, next we extract the discharge vector from this LSR footprint and finally we classify this vector. The detection of a moving element is reduced to the classification of the discharge vectors, this being a much smaller volume of information than the sequence of images constituting the initial input. This classification can be obtained through numerous methods, in our case a classification based on concepts similar to the Hamming distance is performed.

Once the classification process has been described, it is appropriate to comment on the parameters used in this process. Firstly, it is important to underline the key role played by the threshold value, since the choice of this value determines the number of LSR values detected. The division of the spectrum of LSR values into bands also plays an important role. In the case study treated in the next section, this division into bands is adjusted via a learning process, which we do not describe in this article, oriented towards optimizing the capacity to differentiate the elements of the training sets.

4.2. Hardware required for obtaining the discharge vector

To describe the hardware needed for obtaining the discharge vector we will refer to the diagram shown in Fig. 6. The input to this piece of hardware is an LSR value for each point of the image (an item of input data every 100 ns). To detect which discharge band this input belongs to, it is sufficient to use an LUT addressed by

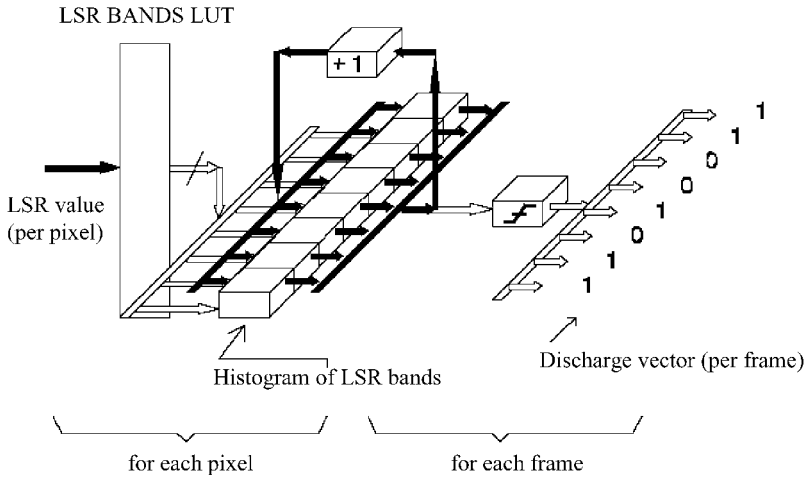

Fig. 6. Hardware to obtain the discharge vector.

the LSR values and containing data words in which each bit is associated to a discharge band. Each position of this LUT contains a word with all bits set to zero apart from the one corresponding to the band to which the value addressing it belongs. The output of this LUT is fed into a bank of registers, one for each LSR band. The register corresponding to the bit that is set to one for the current pixel is then incremented by one while the other registers are left unchanged.

Before starting each frame, the value of all the registers is set to zero. At the end of each frame, the bank of registers contains the histogram of LSR discharges in each band (the histogram of Fig. 5d). To calculate the discharge vector for each frame of the image (typically every 40 ms), the register values only have to be binarized according to whether they are above or below the predefined threshold. This can be done during the vertical blanking period.

With this simple hardware we obtain in real time a discharge vector for each frame of the image. We have transformed the problem of classifying a moving element by processing an image of $n \times m$ pixels with $n \times m \times 8$ bits of information (assuming the gray level is coded in 8 bits) to one of classifying a vector of k bits. The process of classifying the discharge vector can be performed by general-purpose hardware since the volume of information is far smaller than that of the original input and the time available to do so is the frame duration time (40 ms).

5. Experimental data and results

The system has been simulated on a general-purpose machine using various different families of moving elements. Some of these were obtained synthetically while others were extracted from CCIR-standard images taken with a black and white television camera [24]. Here, we choose to use the synthetic images since they are more illustrative as far as the presentation of the mechanisms defined here is concerned.

In the simulation, the mechanisms described in the previous sections were used with the following concrete values: $S=1$, $R=255$, $MAX=255$, $MIN=0$, $k=8$, $0<LSR<255$.

The groups of moving elements (icons) used to train the system are shown in Fig. 7. The training was carried out using the scene background also shown in this Figure and the speed of all the icons, as indicated in the Figure was 1 pixel per frame. During the training phase, the system was shown a series of sequences of images in which each icon was moved, one at a time, in front of the scene background. The presence of each icon in the scene was accompanied by an entry signal (RF in Fig. 7), indicating to the system the denomination of this icon. Once the learning phase was concluded, the system was shown the same sequences without the accompaniment of the icon denomination signal (without indicating RF). The system recognized each icon, identifying it by its corresponding reference.

Next, a series of test icons were constructed by introducing some defects into the icons of the family used for the original training. We show the results related to these defective icons to highlight that our method is capable of classifying them up to a high degree of corruption. These icons and their speed are shown in Fig. 8.

The correspondence between these icons and those of Fig. 7 is as follows:

- The icon DS0 of Fig. 8 is a defective version of icon S0 from Fig. 7.
- The icon DS1 of Fig. 8 is a defective version of icon S0 from Fig. 7.
- The icon DS2 of Fig. 8 is a defective version of icon S2 from Fig. 7.

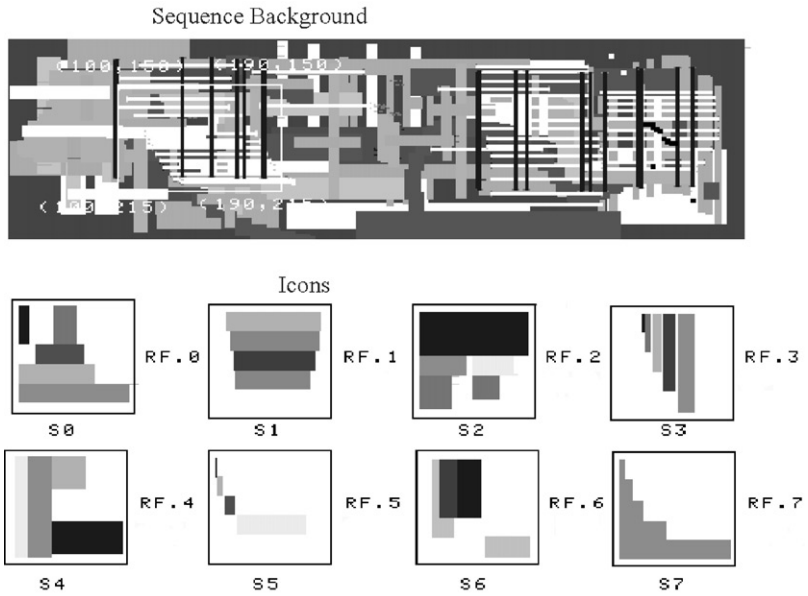

Fig. 7. Synthetic scene background and family of moving icons used for training.

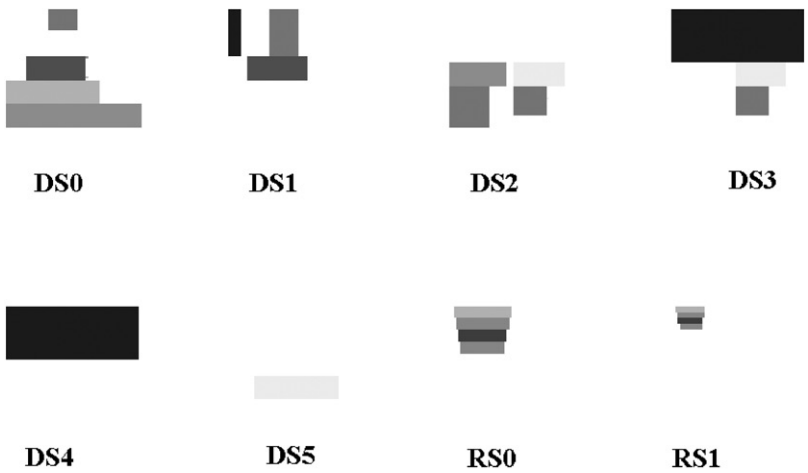

Fig. 8. Defective synthetic icons.

- The icon DS3 of Fig. 8 is a defective version of icon S2 from Fig. 7.
- The icon DS4 of Fig. 8 is a defective version of icon S2 from Fig. 7.
- The icon DS5 of Fig. 8 is a defective version of icon S2 from Fig. 7.
- The icon RS0 of Fig. 8 is icon S1 from Fig. 7 but with $\frac{1}{2}$ the size and $\frac{1}{2}$ the speed.
- The icon RS1 of Fig. 8 is icon S1 from Fig. 7 but with $\frac{1}{4}$ the size and $\frac{1}{4}$ the speed.

Of these defective icons, those that appear in Fig. 8 with references DS0, DS2, DS3, DS5, RS0 and RS1 were correctly identified. Those with reference DS1 and DS4 were incorrectly identified.

It is worth drawing attention to the behavior of the system with respect to the temporary hiding of part of the moving element behind static elements of the three dimensional scene. In these cases, the system is quite robust since maintaining the previous discharges for the short space of time for which the moving element is hidden is sufficient.

6. Conclusions

In this article we defined the LSR and proposed its use for the detection of moving elements, or of characteristic movement situations, in image sequences. We also described how this characteristic could be continuously

derived from image sequences in real time and we presented a concrete system design based on the LSR characteristic, commenting on the results obtained from its simulation.

In conclusion, we can state that the LSR characteristic is very well suited to use in the detection process for moving elements in sequences of synthetic images, as shown in this paper. We continue working on real images, and the results obtained in our tests are also satisfactory. The process of simplification of information that is carried out on extracting the LSR characteristic is of great help to the tasks of detection or classification. This is because it converts a sequence of images with moving elements into an image with a stable pattern in which the information concerning elements of the scene that are not moving has been eliminated. At the same time, it provides information concerning the speed of the moving elements. Furthermore, its invariant properties, both the invariance with respect to the direction of movement and invariance with respect to distance from the sensor, make it well suited to a range of different applications. In addition, the LSR characteristic is quite robust with respect to changes in the static scene backgrounds since these are filtered out due to its nature. The characteristic also has good discriminatory capacity, since LSR footprints are in general significantly different for different moving elements. Another important aspect of the LSR characteristic is that it is possible to extract it in real time over the whole image using simple hardware at low cost.

The limitations of the LSR characteristic are associated with the loss of information suffered in obtaining it. The use of specific descriptors does not guarantee a complete description. Not all the knowledge contained in the image sequence is stored in the LSR. This means that elements that generate the same LSR are not distinguishable by the system, In the same way, the invariants described impose limitations on the applications which can be treated. Though it is worth mentioning that the lack of invariance under rotations with respect to the direction of movement can be used to detect different orientations of the same element.

In addition, with respect to the concrete application of this characteristic to a problem of moving element detection, as described in this article, in the simulation of this system, the LSR characteristic produced the expected results. It successfully detected moving elements in both synthetic images and in images obtained from a television camera. That the proposed system based on the LSR characteristic can be implemented to work in real time with simple hardware has also been confirmed.

In summary, the LSR characteristic and the rest of the mechanisms presented in this article are valid for the detection tasks under the conditions we have stated. We are convinced that this characteristic and the information generated by the permanence memories offer great possibilities for the analysis of movement in image sequences. In our research group we are using this information both for detection of moving elements and for analysis of 3D scenes captured by moving cameras. We are also working on the study of other characteristics to support the LSR with the aim of increasing the reliability of the results and widening the field of application.

Acknowledgements

This work was supported in part by the Spanish CICYT TIC 94-95 grant. The authors would like to acknowledge the very helpful and perspicacious comments of the anonymous referees that have greatly improved this article.

References

[1] Capellini V, Mecocci A, del Bimbo A. Motion analysis and representation in computer vision. Journal of Circuits, Systems and Computers 1993;3(4):797–831.
[2] Dubuisson MP, Jain AK. Contour extraction of moving objects in complex outdoor scenes. International Journal of Computer Vision 1995;14:83–105.
[3] Fan J, Wang R, Zang L, Xing D. Image sequence segmentation based on 2D temporal entropic thresholding. Pattern Recognition Letters 1996;17(10):1101–7.
[4] Femin I, Imiya A, Ichikawa A. Randomized polygon search for planar motion detection. Pattern Recognition Letters 1996;17(10):1109–15.
[5] Hutchinson J, Koch C, Luo L. Computing motion using analog and binary resistive networks. IEEE Computer 1988;21(3):52–63.
[6] Irani M, Rousso B, Peleg S. Computing occluding and transparent motions. International Journal of Computer Vision 1994;12(1):5–16.
[7] Liter JC, Braunstei ML, Hoffman D. Detection of one versus two objects in structure from motion. Journal of The Optical Society of America A 1994;11(12):3162–6.
[8] Murase H, Sakai R. Moving object recognition in eigenspace representation: gait analysis and lip reading. Pattern Recognition Letters 1996;17(2):155–62.
[9] Nagle MG, Srinivasan MV. Structure from motion: determining the range and orientation of surfaces by image interpolation. Journal of the Optical Society of America A 1994;13(1):25–34.
[10] Park D. Adaptative bayesian model for motion segmentation. Pattern Recognition Letters 1994;15(12):1183–9.
[11] Sereno ME. Neural computation of pattern motion. Cambridge, MA: The MIT Press, 1993.
[12] Thomson WB, Pong T. Detecting Moving objects. International Journal of Computer Vision 1990;4:39–57.
[13] Zheng H, Blostein SD. Motion-based object segmentation and estimation using the MDL principle. IEEE Transactions on Image Processing 1995;4(9):1223–35.
[14] Chhabra AK. Real-time computation of optic flow along contours of significant intensity change. Real-Time Imaging 1997;3:87–99.
[15] Camus T. Real-time quantized optical flow. Real-Time Imaging 1997;3:71–86.
[16] Kim H-N, Irwin MJ, Owens RM. Motion analysis on the micro grained array processor. Real-Time Imaging 1997;3:101–10.

[17] Li Z, Wang H. Real-time 3D motion tracking with known geometric models. Real-Time Imaging 1999;5:167–87.
[18] Horn BKP, Schunck BG. Determining optical flow. Artificial Intelligence 1981;17:185–203.
[19] Ballard DH, Brown C. Computer vision. Englewood Cliffs, NJ: Prentice-Hall, 1982.
[20] Schalkoff RJ. Digital image processing and computer vision. Canada: Wiley, 1989.
[21] Lee J, Bing J, Fang W, Chellappa R. VLSI neuroprocessors for video motion detection. IEEE Transactions on Neural Networks 1993;4(2):178–91.
[22] Fernández MA, Mira J. Permanence memory: a system for real time motion analysis in image sequences. IAPR Workshop on Machine Vision Applications MVA'92, Tokyo 1992; p. 249–52.
[23] Joya G, Sandoval F. Projectivity invariant pattern recognition with high-order neural networks. In: New Trends in Neural Computation. Miva J, Cabestany J, Priesto A (eds) Germany: Springer, 1993.
[24] Fernández MA, Fernández-Caballero A, Moreno J, Sebastian G. Object classification on a conveying belt. In: Proceedings of the Third International ICSC Symposium on Soft Computing SOCO'99. Canada/Switzerland: Academic Press, 1999.
[25] Fernández MA. Una Arquitectura Modular de Inspiracion Biologica con Capacidad de Aprendizaje para el Analisis de Movimiento en Secuencias de Imagen en Tiempo Real. PhD dissertation. UNED, Madrid, Spain, 1995.
[26] Lettvin JH, Maturana H, McCulloch WS, Pitts WH. What the frog's eye tells the frog's brain. Proceedings of the IRE 1959;47(11):1940–51.
[27] Fernández MA, Mira J, López MT, et al. Local accumulation of persistent activity at synaptic level: application to motion analysis. In: From Natural to Artificial Neural Computation. Miva J, Sandoval F (eds) Germany: Springer, 1995. p.137–43.