ELSEVIER

Expert Systems with Applications 31 (2006) 570–594

Expert Systems with Applications

www.elsevier.com/locate/eswa

Algorithmic lateral inhibition method in dynamic and selective visual attention task: Application to moving objects detection and labelling

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

[a] *Departamento de Informática, Escuela Politécnica Superior de Albacete, Universidad de Castilla-La Mancha, 02071 Albacete, Spain*
[b] *Departamento de Inteligencia Artificial, E.T.S.I. Informática, Universidad Nacional de Educación a Distancia, 28040 Madrid, Spain*

Abstract

In a recent article, knowledge modelling at the knowledge level for the task of moving objects detection in image sequences has been introduced. In this paper, the algorithmic lateral inhibition (ALI) method is now applied in the generic dynamic and selective visual attention (DSVA) task with the objective of moving objects detection, labelling and further tracking. The four basic subtasks, namely feature extraction, feature integration, attention building and attention reinforcement in our proposal of DSVA are described in detail by inferential CommonKADS schemes. It is shown that the ALI method, in its various forms, that is to say, recurrent and non-recurrent, temporal, spatial and spatial-temporal, may perfectly be used as a problem-solving-method in most of the subtasks involved in the DSVA task.

Keywords: Dynamic and selective visual attention; Algorithmic lateral inhibition; Feature extraction; Feature integration

1. Modelling the dynamic and selective visual attention task

1.1. Modelling by algorithmic lateral inhibition method

In a recent article (Mira, Delgado, Fernández-Caballero, & Fernández, 2004), knowledge modelling at the knowledge level for the task of moving objects detection in image sequences has been introduced. Three items were the focus of the approach: (1) the convenience of knowledge modelling of tasks and methods in terms of a library of reusable components and in advance to the phase of operationalization of the primitive inferences, (2) the potential utility of looking for inspiration in biology, (3) the convenience of using these biologically inspired problem-solving methods (PSMs) to solve motion detection tasks. In this paper, the approach is the same, and the algorithmic lateral inhibition (ALI) method is now applied in the generic dynamic and selective visual attention task with the objective of moving objects detection, labelling and further tracking.

A computational system within a media made up of visual sensors is said to possess the faculty of dynamic and selective visual attention (DSVA) when it is capable of processing the sequence of images coming from this media, selecting in every moment a series of objects from the current scene at that precise moment in time and focusing on the selected objects through time, at least while the criteria used in the process of selection are fulfilled. In this paper the algorithmic lateral inhibition (ALI) method adapted to the different subtasks involved in the DSVA work is introduced. It is shown that the ALI method, in its various forms, that is to say, recurrent and non-recurrent, temporal, spatial and spatial-temporal, may perfectly be used in most of the subtasks involved in the DSVA task. From (Mira et al., 2004) firstly remember the non-recurrent case. Each calculation element samples its data in the central (C) and periphery (P) part of the volume that its RF (receptive field) specified in the input space *V*. On these two data fields, the calculation element carries out evaluation inferences and results comparison. This comparison inference is made according to a set of criteria to generate a set of discrepancy classes as input to the final selection, where the output is obtained from the set of outputs associated with the different discrepancy classes, according to the specific discrepancy classes generated by the previous comparison inference. In an analogous manner there is the inferential scheme for the recurrent ALI circuits. Now each element of calculus starts to infer from data sampled in the central (C*) and periphery (P*) parts of its feedback receptive fields in the output space. The values in C* (individual opinion before

[*] Corresponding author. Tel.: +34 967 599200; fax: +34 967 599224.
E-mail address: caballer@info-ab.uclm.es (A. Fernández-Caballero).

0957-4174/$ - see front matter © 2005 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2005.09.062

dialogue) are compared with the evaluation of the 'opinions' of all the elements in the periphery. This comparison is made according to a set of rules for consensus to produce a discrepancy class. Finally, as in the non-recurrent case, this discrepancy is the input to a selection to provide the consensus output.

In the inferential schemes used in the writing of this work we will use the usual agreement in CommonKADS (Schreiber, Akkermans, Anjewierden, de Hoog, Shadbolt, van de Velde, et al., 2001) to represent the static and dynamic roles and the operational meaning of the inferential verbs proposed by Breuker and van de Velde (1994) (evaluate, compare, select,…) The dynamic roles are represented as rectangles with solid lines and the static roles are represented as rectangles with broken lines. On some occasions a dynamic role resulting from an inference can play the part of a static role in another later inference within the information flow. Accordingly, we will use a double rectangle where a solid line is the end of the arrow that brings the rectangle from the inference generating it as a dynamic role and where a broken line (outside the other rectangle) is the origin of the arrow finishing in the inference where the rectangle plays a static role.

1.2. Modelling the dynamic and selective visual attention task

Attention is the cognitive process of selectively concentrating on one thing, while ignoring other things. Of the many cognitive processes associated with the human mind (decision-making, memory, emotion, etc.) attention is considered the most concrete because it is tied so closely to perception. One of the most influential theories about the relation between attention and vision is the feature integration theory (Treisman & Gelade, 1980). In the 1960s, Anne Treisman found that certain object features such as colour or orientation could be detected in parallel, while conjunctions of these features could not. According to this model, attention is responsible for binding different features into consciously experienced wholes. Attention remains a major area of investigation within psychology and neuroscience, and many computational models have been proposed for selective attention so far.

The models for selective attention may be divided into two broad groups: (a) models based exclusively on the scene (bottom-up), and, (b) models based on the scene (bottom-up) and on the task (control top-down). The first bottom-up neurally plausible architecture of selective visual attention was proposed by Koch and Ullman (1985), and it is related to the feature integration theory. In (Itti, Koch, & Niebur, 1998) a visual attention system inspired by the behaviour and the neural architecture of the early primate visual system is presented. Multi-scale image features are combined into a single saliency map. A dynamical neural network then selects attended locations in order of decreasing saliency. The connectionist model called SLAM (selective attention model) (Phaf, van der Heijden, & Hudson, 1990) assumes an interactive-activation network consisting of input, hidden, and output nodes. Input nodes represent words and colours in a particular spatial position. Processing occurs through activation spreading from colour input nodes via hidden nodes to output nodes, and directly from word input to output nodes, whereby nodes change their activation with time in a continuous non-linear manner. There are excitatory links between nodes representing compatible information, and there are inhibitory links between nodes standing for incompatible information. In (Heinke, Humphreys, & diVirgilo, 2002) a neural network (connectionist) model called the selective attention for identification model (SAIM) is introduced. The function of the suggested attention mechanism is to allow translation-invariant shape-based object recognition. One goal of SAIM is to explain neuropsychological data on different versions of attention disorders, that is, 'space- and object-based' neglect. The model of Guided-Search (GS) by Wolfe (1994) uses the idea of 'saliency map' to realize the search in scenes. GS assumes a two-stage model of visual selection. The first, pre-attentive stage of processing has great spatial parallelism and realizes the computation of the visual simple features. The second stage is spatially serial and it enables more complex visual representations to be computed, involving combinations of features.

The approach to the DSVA task is explained next. The selection of the elements of interest in the scene necessarily starts by setting up the criteria based on the features extracted from the elements (feature extraction). Firstly, all the necessary mechanisms to provide sensitivity to the system are included in order to succeed in centring the attention. Frame to frame attention is captured (attention capture) on elements (blobs) constructed from image pixels that fulfil the requirements established by the user and obtained after a feature integration. On the other hand, stability has been provided to the system. This has been achieved by including mechanisms to reinforce attention (attention reinforcement), in such a way that the elements accepting the user's predefined requirements (figures) are strengthened up to be shaped as the system attention centre.

Fig. 1 shows the block diagram that illustrates the two components of 'sensitivity' and 'stability' in the DSVA task, as an initial step towards the construction of the complete conceptual model. Notice that we name components of

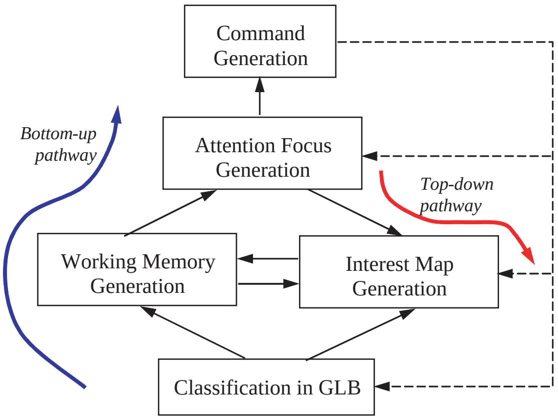


Fig. 1. DSVA block diagram.

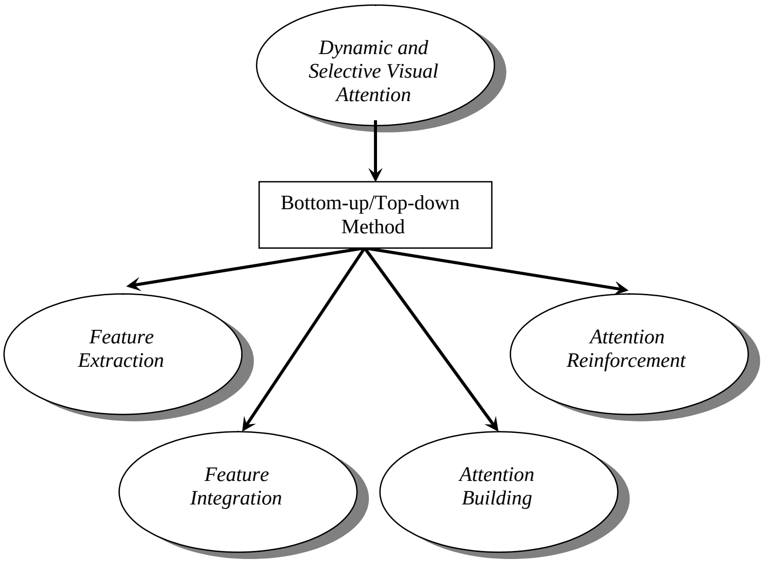


Fig. 2. Decomposition of DSVA into its four subtasks.

'sensitivity' to what neurophysiology calls a data-driven or bottom-up organization. Similarly, we call 'stability' to what in Psychology is referred to as a knowledge-driven or top-down organization.

The terminology used to describe the task is the following one:

- *Pixel of interest*: image pixel that fulfils the dynamic features (motion features) defined by the user.
- *Point of interest*: pixel of interest which also fulfils the blob and figure features defined by the user.
- *Grey level band*: step or range of grey levels.
- *Blob*: set of connected pixels belonging to a same grey level band, and which includes at least one point of interest.
- *Blob of interest*: spot that fulfils the blob features defined by the user.
- *Figure*: set of connected blobs of interest.
- *Figure of interest*: figure that fulfils the features defined by the user.
- *Attention focus*: set of figures of interest.

In the Fig. 2 you may distinguish the four basic subtasks in the DSVA work, where:

- *Feature extraction*: processing of the dynamic features of the image pixels, the features related to the blobs, and the figures of an image capable of capturing attention.
- *Feature integration*: application of the criteria established by the user to the features extracted in the Feature Extraction subtask, consisting of filtering pixels, blobs and figures to setup the interest points.
- *Attention building*: construction of blobs starting from the interest points calculated in Feature Integration.
- *Attention reinforcement*: construction of figures and keeping attention on certain figures (or objects) of the image sequence that are of real interest to the user.

Next each of the subtasks is described with the help of a simple running example, as shown in Fig. 3. It consists of a scene where a vehicle and a pedestrian are moving independently. The aim is to hold the attention on the objects that fulfil certain conditions of size and shape, in addition to fulfilling a series of dynamic parameters.

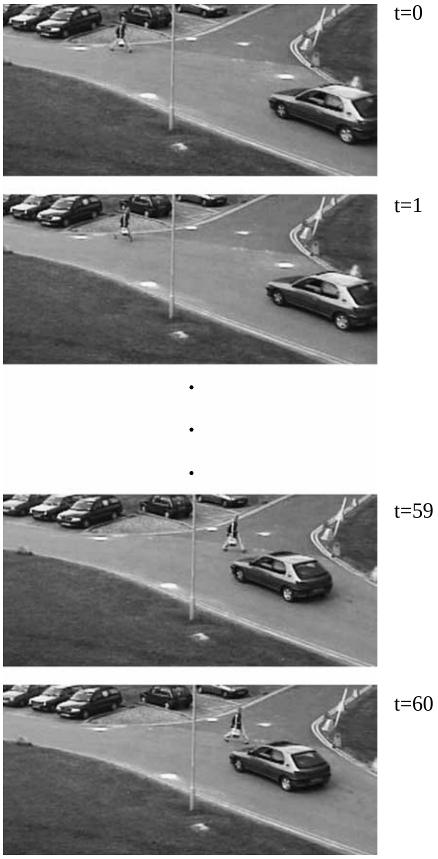


Fig. 3. Running example input sequence.

2. Attention building

The purpose of the attention building subtask is to select and to label zones (blobs) of the objects (figures) to pay attention on. See, therefore, that after processing attention building, the whole figures are not classified. Instead, each one of the blobs, understood as homogeneous connected zones that form the figures, are marked with different labels. Obviously, the blobs are built from image points that fulfil the requirements established by the guidelines of the observer (points of interest).

Fig. 4 shows a process scheme for the attention building subtask. The output of this subtask is called Working Memory. The selection of the term working memory stems from its resemblance with the concept used in Psychology, where the working memory—also called short-term or functional memory (Awh, Anllo-Vento, & Hillyard, 2000; Awh & Jonides, 2001; Awh, Matsukura, & Serences, 2003; O'Reilly, 2003; O'Reilly, Braver, & Cohen, 1999)—stores and processes during a brief period of time the chosen information coming from the sensory records. In our case, only blobs constructed in the Working Memory will potentially form the figures of the system's attention focus.

In our proposal the blobs of the Working Memory are built from the information in the so-called Interest Map and from the input image divided into grey level bands. The interest map is obtained, as it will be explained later on, by performing a feature integration, both of motion and shape features. For each image pixel, in the interest map the result of a comparison among three discrepancy classes—'activator', 'inhibitor' and 'neutral'—is stored. The interest points are those points of the interest map labelled as 'activator' points. Grey-level bands are obtained as a result of the classification in grey-level bands subtask, which is explained next.

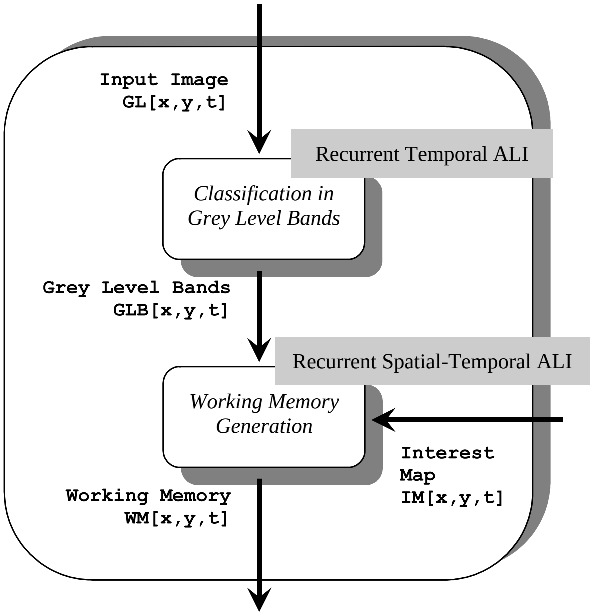


Fig. 4. The 'attention building' subtask.

2.1. Classification in grey-level bands

The classification in grey-level bands subtask transforms the 256 grey-level-input images into images with fewer levels. In particular, good results are usually got with eight levels. These eight level images are called images segmented into eight grey-level bands (GLB). The reason of working with grey-level bands is two-fold. (1) A now traditional method of motion detection is based on image differencing. The noise level diminishes with little changes in grey level (or luminosity) of a same object between two consecutive images, when joining a range of grey levels into a single band. (2) On the other hand, a decrease in the computational complexity is achieved, bearing in mind the great parallelism used in the algorithms of the proposed model. We now calculate in parallel in the order of magnitude of grey-level bands n, and not of grey levels N, where $N > n$.

The inferential scheme of classification in grey level bands (Fig. 5) follows the layout of a Recurrent-Temporal ALI, as the output in the current time instant t, GLB[x,y,t], takes into account its proper output in the previous instant, GLB[x,y,$t-1$]. As shown in Fig. 5, the static roles are the number of grey-level bands (n) in which the image is split, the overlap between the bands (O) and the maximum (GL_{max}) and minimum (GL_{min}) grey levels of the input image. The overlap value between the bands, O, is used to augment the size of the bands without diminishing the number of bands. This way the level of noise decrease due to little changes in luminosity between two consecutive image frames is even better adapted.

Fig. 6 graphically shows the underlying idea in the overlap between bands. The figure shows in the left side the division in N grey levels, where $N = GL_{max} - GL_{min} + 1$, and in the rest of the figure, the division in n grey level bands, ranging from GLB_1 to GLB_n, where obviously $n < N$. As it may be noticed, there is an overlap between the grey level bands, so that an image point whose grey level is GL could be thought beforehand to belong to a single grey-level band—if the overlap does not affect the grey level—or to two different grey-level bands—if the grey level is included in an overlap zone. Obviously, at each instant t, an image point may only belong to one grey-level band. As it has already been pointed out, the use of grey-level bands reduces the noise level. The overlap between bands aims that a point belonging to a grey-level band remains in the same band when the change of luminosity is 'small'. 'Small' means that the luminosity of an image point included in an overlap zone does not fall out of the same zone in the next moment. Thus, noise decreases without diminishing the number of bands. Also observe in Fig. 6 the values E_{2max}, the maximum band value (in this case, for band 2), and E_{2min}, the minimum band value.

Next an explanation of all inferences present in Fig. 5 is provided. The first evaluate inference performs the computation of value B over the centre C. This value is the output of the subtask if there is no change of grey-level band. B is the computed value of stepping from grey-level to grey-level band before asking for the overlap. The calculus of B is expressed in formula (1). As you may notice, this is just an easy scale

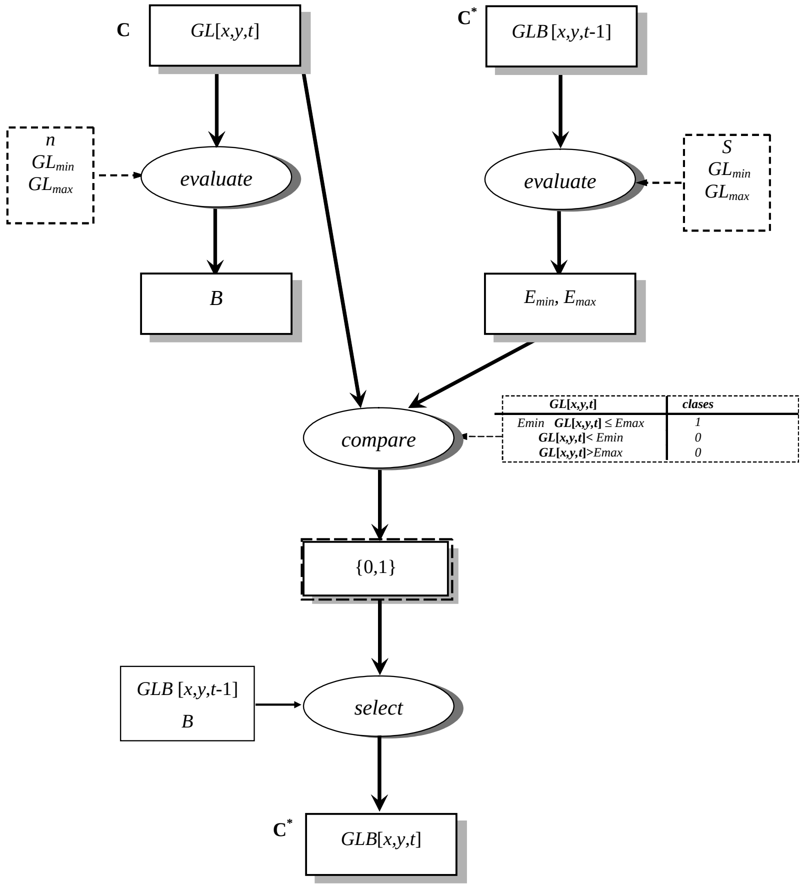


Fig. 5. Inferential scheme for ‘classification in grey-level bands’.

transformation. Notice also that we are still not deciding if there is a change of grey-level band in the point.

$$B = \left\lfloor \frac{\mathrm{GL}[x,y,t]n}{(\mathrm{GL}_{\max} - \mathrm{GL}_{\min} + 1)} \right\rfloor + 1 \tag{1}$$

The second evaluate inference performs the calculation of the extreme values of the grey-level band, $E_{\max}$ and $E_{\min}$, on the central part of the output space. In other words, it performs on its proper output in the previous time moment. When interpreting the formulae (2) and (3), notice that $E_{\min}$ and $E_{\max}$ correspond to a grey level, the lowest grey level of the grey-level band at instant $t-1$ and the highest grey level of the grey-level band at the same time instant $t-1$.

$$E_{\min} = \max\left(\frac{(\mathrm{GLB}[x,y,t-1]-1)(\mathrm{GL}_{\max} - \mathrm{GL}_{\min} + 1)}{n} - O, \mathrm{GL}_{\min} \right) \tag{2}$$

$$E_{\max} = \min\left(\frac{\mathrm{GLB}[x,y,t-1](\mathrm{GL}_{\max} - \mathrm{GL}_{\min} + 1)}{n} + O, \mathrm{GL}_{\max} \right) \tag{3}$$

Now, the compare inference verifies whether the value of the grey level, GL[x,y,t], produces a change of band with respect to the grey-level-band value obtained at $t-1$, GLB[x,y,$t-1$]. For it, the criterion is the following one: if GL[x,y,t] is inside the range established between $E_{\min}$ and $E_{\max}$, then the output of this inference, called *variation* as it detects a change of grey-level band between time instants t and $t-1$ at a pixel, takes value 0, and 1 in any other case:

$$\text{Variation} = \begin{cases} 0, & \text{if } E_{\min} \le \mathrm{GL}[x,y,t] < E_{\max} \\ 1, & \text{otherwise} \end{cases} \tag{4}$$

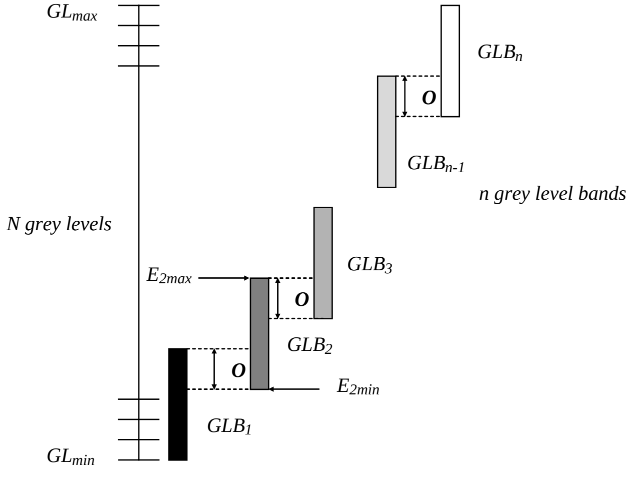


Fig. 6. Relation between grey-level bands and overlap.

Finally, the select inference offers as output value the calculated value B, if the output of inference compare was 1, and GLB[x,y,$t-1$] if the output of inference compare was 0:

$$\mathrm{GLB}[x,y,t] = \begin{cases} \mathrm{GLB}[x,y,t-1], & \text{if variation} = 0 \\ B, & \text{otherwise} \end{cases} \tag{5}$$

To sum up, the result of the classification in Grey-Level Bands subtask is, for each input image pixel, the transformation of the grey level into its corresponding grey-level band, but always bearing in mind its grey-level band in the previous time instant.

2.2. Working-memory generation

The objective of this subtask is, firstly, to select and to label (to classify numerically) image blobs associated to pixels of interest-pixels that possess dynamic features in the numerical intervals established by the guidelines of the observer. Secondly, the subtask eliminates the blobs whose shape features do not correspond with the guidelines. In order to achieve these aims, the images in Grey-Level Bands are segmented into regions composed of connected points whose luminosity level belongs to a same interval (or grey-level band), and only connected regions that include some 'activator' point (or, point of interest) in the Interest Map are selected. Each region or zone of uniform grey level is a blob of potential interest in the scene.

In previous works of our research team some methods based on image segmentation from motion have already been used. These methods are the permanency effect and the lateral inhibition (Fernández-Caballero, Mira, Fernández, & López, 2001; López, Fernández, Fernández-Caballero, & Delgado, 2003). Based on the satisfactory results of these algorithms (Fernández-Caballero, Mira, Férnandez & Delgado, 2003), in this paper we propose to use mechanisms of charge and discharge together with mechanisms of lateral inhibition to solve the current task of DSVA.

Fig. 7 shows the inputs and the output of the Working Memory Generation subtask. The inputs are the image in *Grey-Level Bands* and the Interest Map, whereas the output is the Working Memory. The Working Memory stores a common number for each pixel belonging to a blob. Value 0 is given to pixels that do not belong to any blob.

The idea consists in overlapping, as with two superposed transparencies, the Grey-Level Bands image of the current frame (t) with the Interest Map image built at the previous frame ($t-1$). At t, only blobs of the Grey-Level Bands image at t are selected where at least one point of interest fell at $t-1$. Nevertheless, not the total blob is taken, but pixels that coincide with points of the Interest Map classified as 'inhibitors' are eliminated. The computational model used to perform the preceding steps incorporates the notion of lateral inhibition, which enables that the points of interest flood their zones of uniform grey levels whilst eliminating all points classified as 'inhibitors'.

In order to achieve the aims of this subtask, the processes shown in Fig. 8 are performed. Firstly, by means of the process

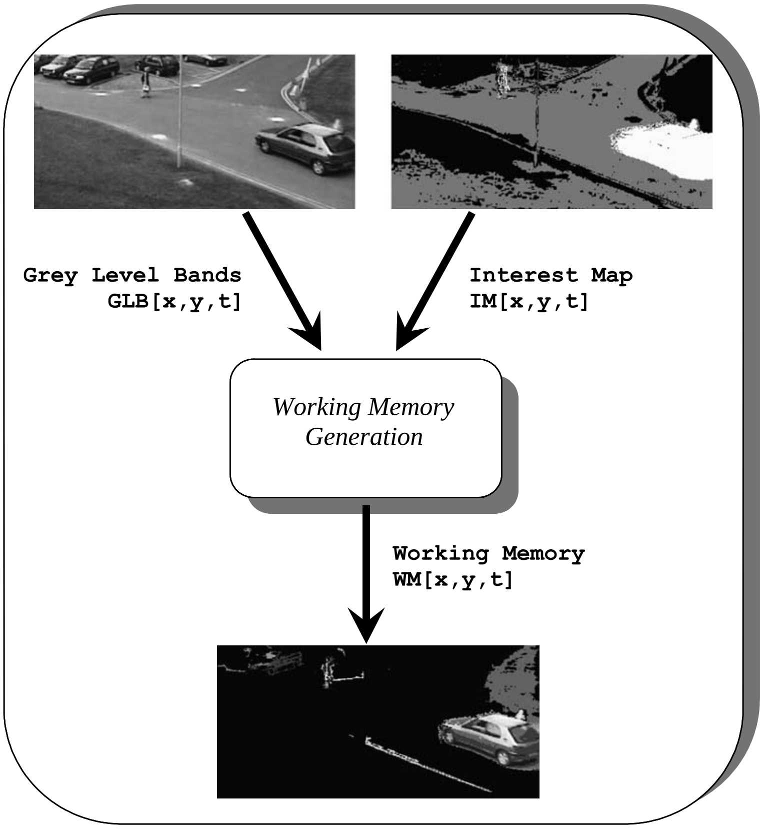


Fig. 7. The 'working memory generation' subtask.

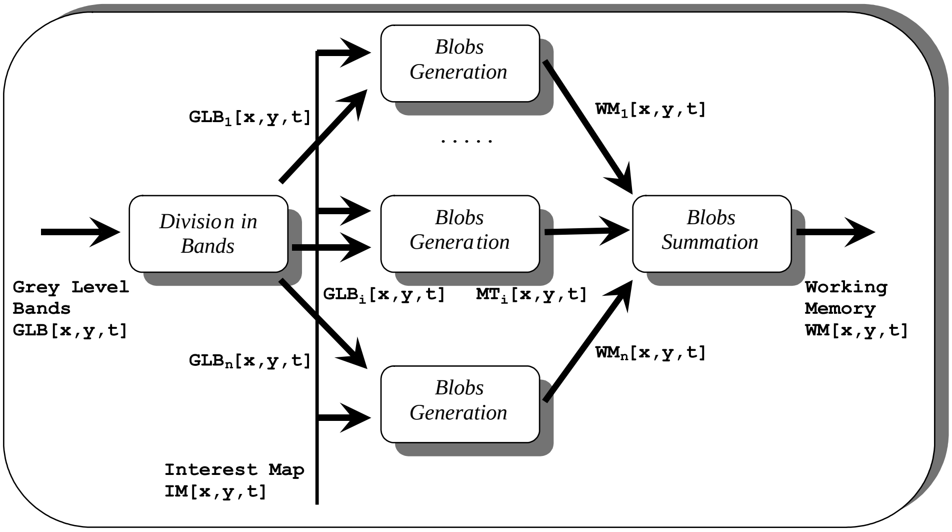


Fig. 8. Scheme for 'working-memory generation'.

called Division in Bands, n images $\mathrm{GLB}_i[x,y,t]$, $1 \le i \le n$, (one image per band) are obtained from the grey-level-bands image, $\mathrm{GLB}[x,y,t]$.

Each one of the $\mathrm{GLB}_i[x,y,t]$ images stores a 1 if $\mathrm{GLB}[x,y,t]=i$ (that is to say, for each pixel whose grey-level band is equal to band i), and 0 in the opposite case, as shown in Fig. 9. Next, for each $\mathrm{GLB}_i[x,y,t]$ the different connected regions that include an 'activator' point in the Interest Map and which do not correspond to 'inhibitor' points in the Interest Map are labelled. This process has been called Blobs Generation and its output is the Working Memory for Grey-Level Band i, $\mathrm{WM}_i[x,y,t]$. $\mathrm{WM}_i[x,y,t]$ stores the label corresponding to the generated blob to pixels belonging to blobs, and value 0 for the rest of the pixels. The output of this subtask, the Working Memory, $\mathrm{WM}[x,y,t]$, is the result of adding up all the obtained blobs in all $\mathrm{WM}_i[x,y,t]$ (through the Blobs Summation subtask).

Fig. 9 shows the application of Division in Bands to the running example. In this figure pixels in black in each $\mathrm{GLB}_i[x,y,t]$ represent that their grey-level-band value is equal to band i.

Now, Fig. 10 shows the result of applying the Blobs Generation subtask to the running example.

A detailed description of all subtasks composing Working-Memory Generation is provided in the following paragraphs.

2.2.1. Division in bands

The only inference of Division in Bands, inference evaluate, shown in Fig. 11, obtains n binary images $\mathrm{GLB}_i[x,y,t]$ from an image in grey-level bands $\mathrm{GLB}_i[x,y,t]$ (one image for each band). Each one of these images, $\mathrm{GLB}_i[x,y,t]$ stores a value of 1 for a pixel whose grey-level band is i and a 0 in the opposite case. That is to say:

$$\mathrm{GLB}_i[x,y,t] = \begin{cases} 1, & \text{if } \mathrm{BNG}[x,y,t]=i \\ 0, & \text{otherwise} \end{cases}, \quad \forall i, \quad 1 \le i \le n \tag{6}$$

2.2.2. Blobs generation. Recurrent-spatial-temporal ALI

The blobs generation subtask, as shown in Fig. 12, gets and labels for each grey-level band, pixels belonging to connected regions that include any 'activator' point in the interest map but do not correspond with 'inhibitor' points of the interest map. Its output, working memory for grey-level band i, $\mathrm{WM}_i[x,y,t]$, stores for each pixel the label corresponding to the generated blob if it belongs to the blob, or the value 0.

The static roles of the inferences have the following meanings: $v_{\mathrm{activator}}$ is the value given to the points of interest ('activators') of the interest map, v_{neutral} is the value for the 'neutral' points of the interest map, $v_{\mathrm{inhibitor}}$ is the value for the 'inhibitor' points of the interest map, NR is the number of rows of the image, and NC is the number of columns of the image. The dynamic roles of the inference are the grey-level band i, $\mathrm{GLB}_i[x,y,t]$, the interest map, $\mathrm{IM}[x,y,t]$, the label value of the centre, z_c, and the label value of the periphery, z_p. Centre has to be understood as each element used to compute the working memory—in this case, the grey-level bands—and periphery as the eight neighbours that surround each of the centres. The idea is to get a consensus label for all pixels of a common blob. As shown in Fig. 12, there are two time scales, a global scale, t, and a local scale, τ, typical of recurrent algorithmic lateral inhibition (ALI).

The primitive evaluate in time scale t, which operates on the input data of the centre, C, formed by $\mathrm{GLB}_i[x,y,t]$ and $\mathrm{IM}[x,y,t]$, assigns at each time increment t to all points where $\mathrm{GLB}_i[x,y,t]=1$ an initial and provisional value as shown in formula (7):

$$z_c(t) = \begin{cases} x \times \mathrm{NC} + y + 1, & \text{if } \mathrm{GLB}_i[x,y,t] = 1 \wedge \mathrm{IM}[x,y,t] \\ & \quad = v_{\mathrm{activator}} \\ \mathrm{NR} \times \mathrm{NC} + 1, & \text{if } \mathrm{GLB}_i[x,y,t] = 1 \wedge \mathrm{IM}[x,y,t] \\ & \quad = v_{\mathrm{neutral}} \\ 0, & \text{otherwise} \end{cases} \tag{7}$$

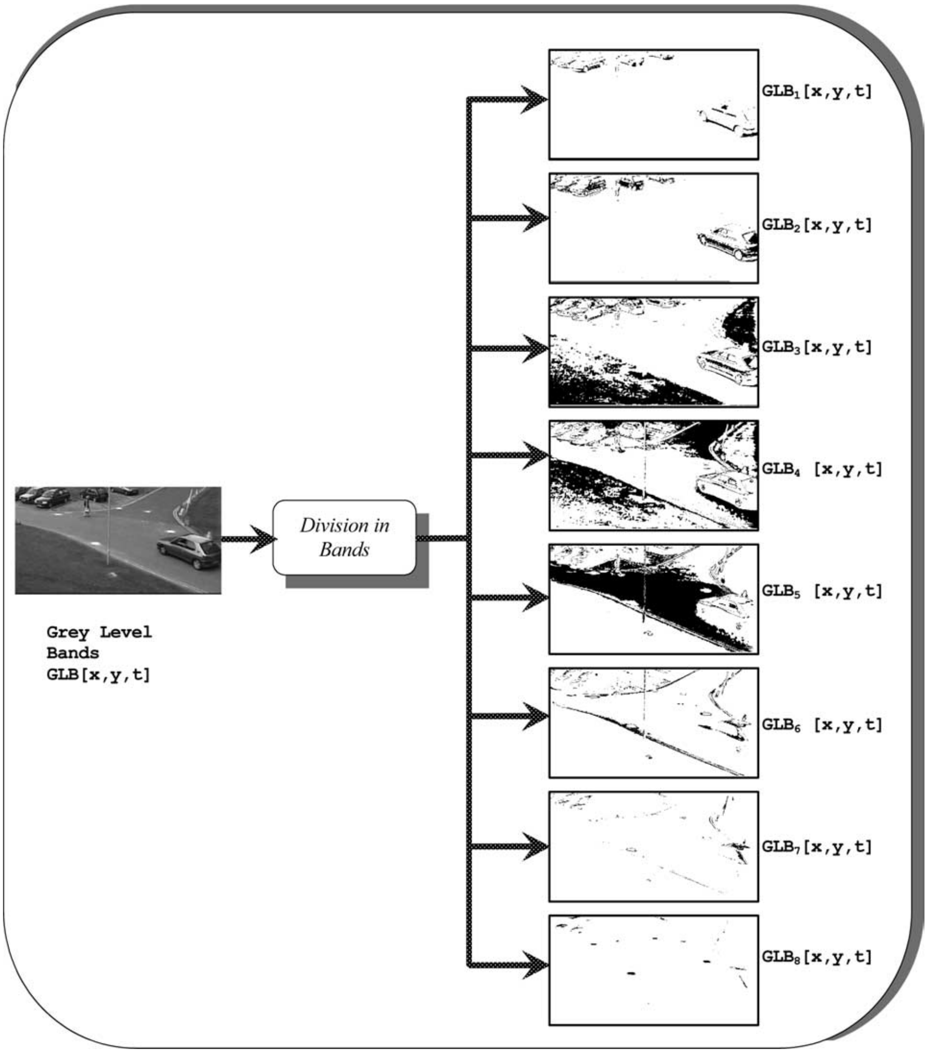


Fig. 9. Application of 'division in bands' to the running example.

This value corresponds to $(x \times \mathrm{NC}+y+1)$ when $\mathrm{IM}[x,y,t]=v_{\mathrm{activator}}$, to a greater value than NR×NC when $\mathrm{IM}[x,y,t]=v_{\mathrm{neutral}}$, and to value 0 in the rest of the cases. In other words, if the centre belongs to the grey level band and corresponds to a point of interest of the interest map, the tentative value for the centre is a function of its proper coordinate. Now, if the centre belongs to the grey level band but corresponds to a 'neutral' point of the interest map, the provisional value given to the centre is a value greater than any possible value of the coordinate function. In any other case, the value is 0. This initial value assignment to all calculus elements in time scale t, serves to get an agreement in the labels of the blobs after the negotiation in time scale τ. Of course, there will only be collaboration among calculus elements that possess an initial value greater than 0.

The primitive evaluate operates on the output data of the periphery, P*, in time scale τ. It is made up of the eight neighbours of each $\mathrm{WM}_i[x,y,\tau]$, and gets the minimum value of their values, called z_{p}.

$$z_{\mathrm{p}}=\min(v_{\mathrm{p}(\alpha,\beta)})\,\forall\,[\alpha,\beta]\in[x\pm 1,y\pm 1]|([\alpha,\beta]\neq[x,y]) \wedge(0<z_{\mathrm{p}(\alpha,\beta)}\leq z_{\max}),\quad \text{where } z_{\max}=\mathrm{NR}\times\mathrm{NC}+1 \tag{8}$$

Inference compare generates two discrepancy classes $D1$ y $D2$ by comparing values $z_{\mathrm{c}}(\tau)$ and $z_{\mathrm{p}}(\tau)$, as shown in Table 1.

$D1$ means that the value of the centre does not change in the interaction, whereas $D2$ means that z_{c} takes the value of z_{p} in the present pass. As you may notice, the centre talks with the elements of the periphery whose value is different from 0 and always retains the smallest value. Thus, blobs are labelled with the ordinal corresponding to the point with the lowest coordinate (if the superior left image point is taken as origin). The values generated by the primitive compare fulfil the input dynamic role to inference select, which determines the output value $z_{\mathrm{c}}(\tau)$ as a function of the resulting discrepancy class ($D1$, $D2$). Thus,

$$z_c(\tau)=\begin{cases} z_{\mathrm{c}}(\tau-\Delta\tau), & \text{if } D1\\ z_{\mathrm{p}}(\tau-\Delta\tau), & \text{if } D2\end{cases} \tag{9}$$

2.2.3. Blobs summation

The blobs summation subtask gets the working memory, $\mathrm{WM}[x,y,t]$, as the result of adding up all the blobs computed at

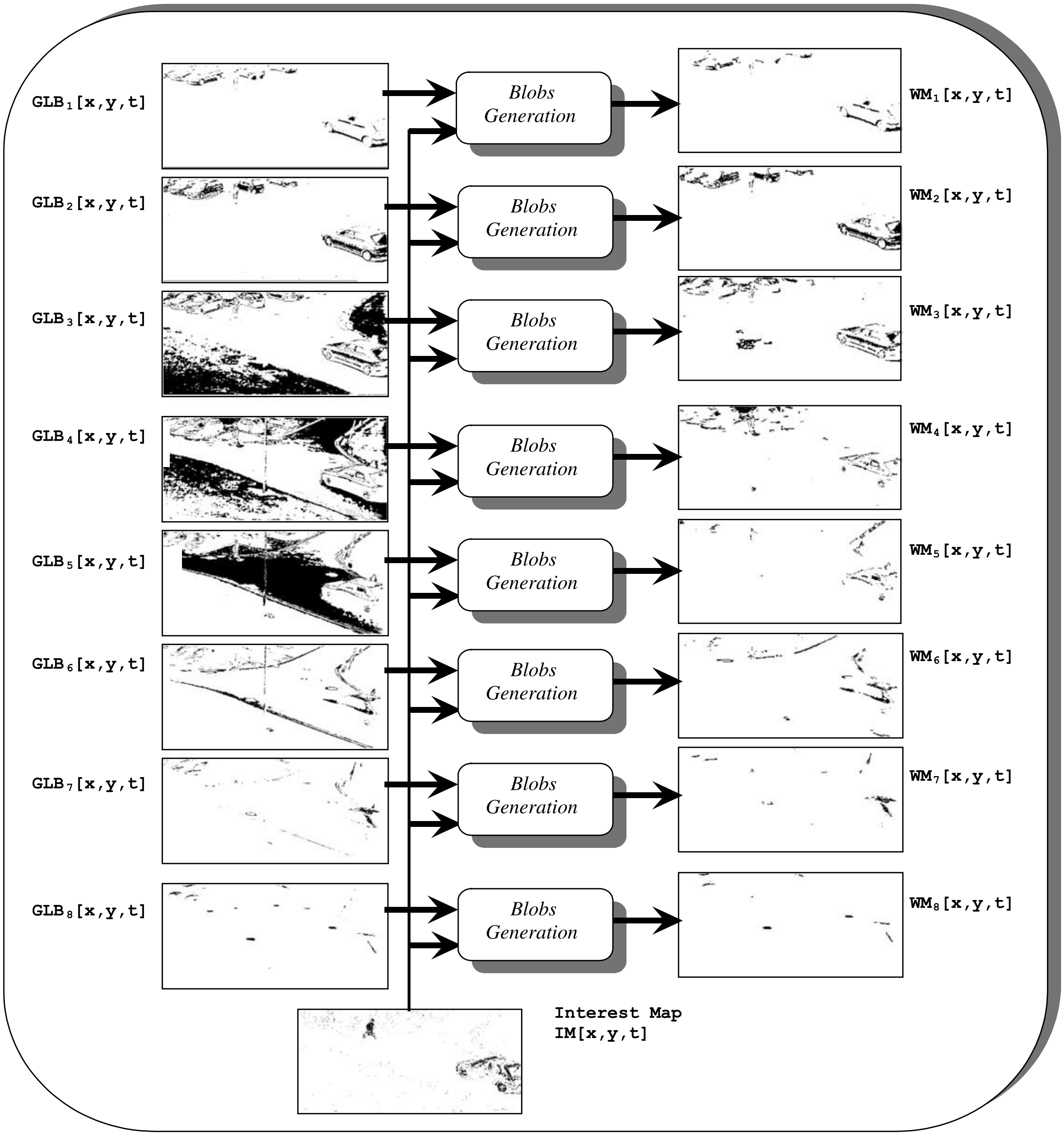


Fig. 10. Application of 'blobs generation' to the running example.

each of the eight working memories for grey-level band i, $WM_i[x,y,t]$, where $i=1,2,\ldots,8$.

Firstly, each $WM_i[x,y,t]$ whose output value in blobs generation is $NR \times NC+1$ (impossible value) is put to 0, as shown in Eq. (10) and in the evaluate inference of Fig. 13.

$$z_i[x,y,t] = \begin{cases} WM_i[x,y,t], & \text{if } WM_i[x,y,t] < NR \times NC + 1, \\ 0, & \text{otherwise} \end{cases} \quad (10)$$

The following primitive is a select that obtains the maximum value of all $z_i[x,y,t]$ (see formula (11)).

$$WM[x,y,t] = \max_i z_i[x,y,t], \quad \forall i \in [1\ldots8] \quad (11)$$

Notice that this maximum selection operation has to be performed for all the elements of matrixes $WM_i[x,y,t]$ to obtain the corresponding element in a single matrix of blobs,

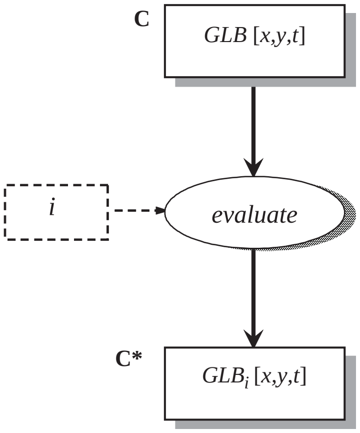


Fig. 11. Inferential scheme for 'division in bands'.

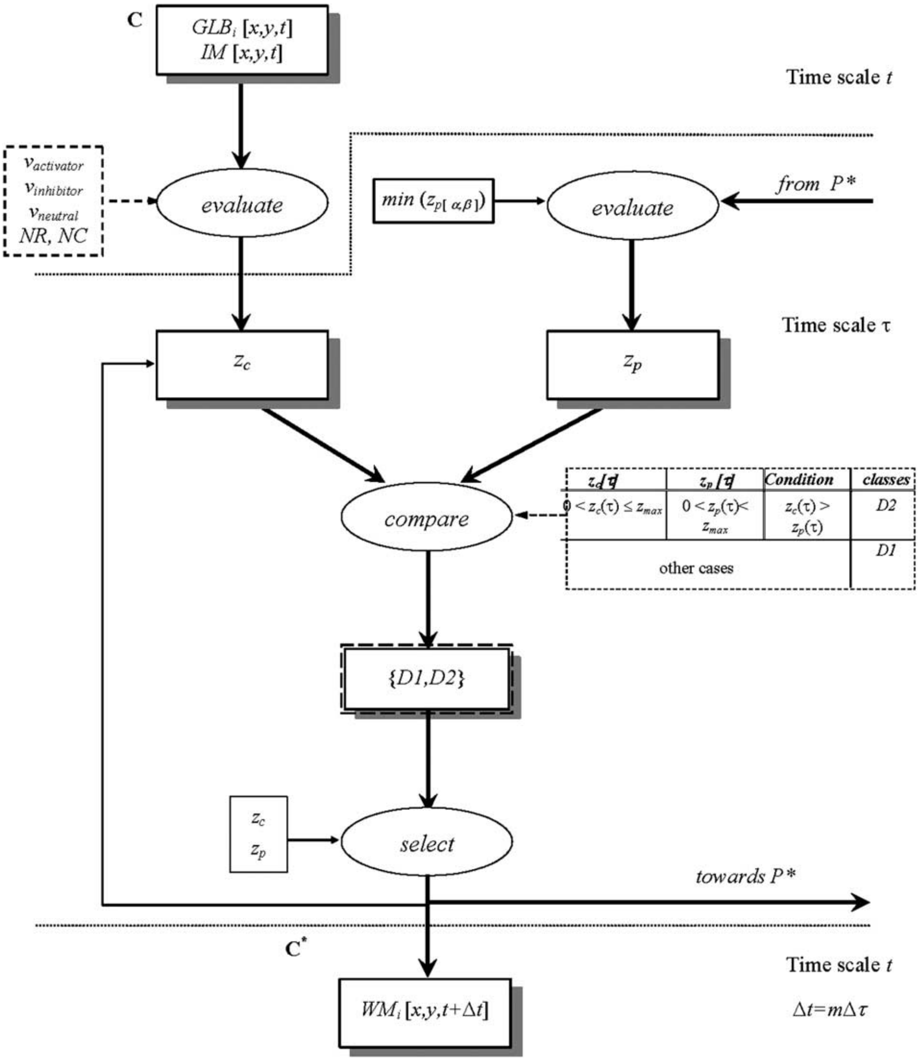


Fig. 12. Inferential scheme for ‘blobs generation’.

WM[x,y,t]. This way all blobs of all grey level bands have been united and labelled with a common value.

Fig. 14 shows the result of applying the blobs summation subtask to the running example. In this figure all the generated blobs are represented in black on a white background.

3. Feature extraction

The feature extraction subtask calculates the properties related to motion present in the input image sequence at pixel level and to the shape of the objects in the sequence. This is why the subtask has been divided into (a) motion feature extraction, at pixels level in the image sequence, and (b) shape-feature extraction, at blobs and figures level of the images.

3.1. Motion-feature extraction

The motion feature extraction subtask acquires the dynamic features of the image pixels; in particular, the extracted features are motion presence, velocity and acceleration.

Due to our experience (Fernández, Fenández-Caballero, López, & Mira et al., 2003; Mira, Fernández, Lopez, Delgado & Fernández-Caballero, 2003), we know some methods to partially generate this information partially. As it was explained before, working with grey levels diminishes the effects of noise generated by small luminosity variations, and may be used to detect motion presence in a more accurate manner. By using accumulative computation methods studied by our research team during the last years (Fernández & Mira, 1992), it is possible to calculate the velocity and the acceleration in those image points where motion has been previously detected from variations in their grey level bands.

Table 1
Discrepancy classes for ‘blobs generation’

$z_c(\tau)$	$z_p(\tau)$	Condition	Classes
$0<z_c(\tau)\le z_{max}$	$0<z_p(\tau)<z_{max}$	$z_c(\tau)>z_p(\tau)$	D2
Other cases			D1

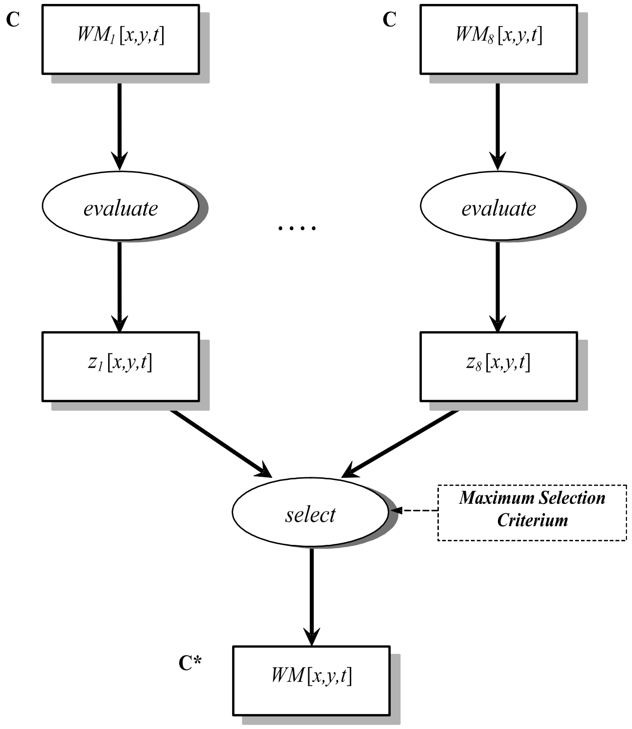


Fig. 13. Inferential scheme for 'blobs summation'.

The more general variety of accumulative computation is the charge/discharge mode, used successfully in works on moving-objects detection, classification and tracking in video sequences so far. This mode may be described by means of the following generic formula (12):

$$\mathrm{Ch}[x,y,t]=\begin{cases}\min(\mathrm{Ch}[x,y,t-\Delta t]+C,\mathrm{Ch}_{\max}), & \text{if property } P[x,y,t]\\ \max(\mathrm{Ch}[x,y,t-\Delta t]-D,\mathrm{Ch}_{\min}), & \text{otherwise}\end{cases} \tag{12}$$

That is to say, temporal accumulation of the persistency of binary property $P[x,y,t]$ measured at every time instant t at each pixel $[x,y]$ of the data field is computed. Generally, if the property is fulfilled at pixel $[x,y]$, charge value at that pixel $\mathrm{Ch}[x,y,t]$ goes increasing by the increment in charge value C up to reaching $\mathrm{Ch}_{\max}$, whilst, if property P is not fulfilled, charge value $\mathrm{Ch}[x,y,t]$ goes decreasing by decrement in charge value D down to $\mathrm{Ch}_{\min}$. All pixels of the data field have charge values in the range between the minimum charge, $\mathrm{Ch}_{\min}$, and the maximum charge, $\mathrm{Ch}_{\max}$.

Obviously, values C, D, $\mathrm{Ch}_{\min}$ and $\mathrm{Ch}_{\max}$ are configurable depending on the different kinds of applications, giving raise to all different operating modes of the accumulative computation. $\mathrm{Ch}_{\max}$ and $\mathrm{Ch}_{\min}$ have to be chosen by taking into account that charge values will always fall into this range. The value of C defines the charge increment interval between time instants $t-1$ and t. Greater values of C allow arriving in a quicker way to saturation. On the other hand, D defines the charge-decrement interval between time instants $t-1$ and t. So, notice that the charge stores motion information as a quantified value, which can be used for several classification purposes. In (Mira et al., 2003) the whole architecture of the accumulative computation module is introduced. In that paper all the functioning modes are described, showing the versatility and the computational power of the method.

Fig. 15 permits to observe the scheme of the subtask. The values calculated and associated to motion are the motion presence, the motion-charge memory, the velocity and the acceleration. The motion-charge memory is obtained by means of accumulative computation methods on the negative of property motion presence. Velocity and acceleration, in turn, are calculated from the values stored in the motion-charge memory. Under velocity we understand the calculus of the module and the angle of the velocity vector. The same is true with the acceleration.

3.1.1. Motion-presence calculation. Non-recurrent-temporal ALI

Motion presence, $\mathrm{Mov}[x,y,t]$, is obtained from a point-to-point comparison between a pair of images segmented in grey-level bands in successive time instants. If point $[x,y]$ at time t belongs to the same GLB as at the previous time instant, $t-1$, we consider that there has been no motion, whilst if there has been no change in the GLB, then we consider that motion has been detected.

Motion presence, $\mathrm{Mov}[x,y,t]$, is obtained from input-dynamic roles grey-level band at time instants t and $t-1$ (Fig. 16). Inference compare verifies whether $\mathrm{GLB}[x,y,t]$ and $\mathrm{GLB}[x,y,t-1]$ are equal. The output of this inference, called motion presence, $\mathrm{Mov}[x,y,t]$, is 0 if $\mathrm{GLB}[x,y,t]$ and $\mathrm{GLB}[x,y,t-1]$ are equal, and 1 in the other case:

$$\mathrm{Mov}[x,y,t]=\begin{cases}0, & \text{if } \mathrm{GLB}[x,y,t]=\mathrm{GLB}[x,y,t-1]\\ 1, & \text{if } \mathrm{GLB}[x,y,t]\neq\mathrm{GLB}[x,y,t-1]\end{cases} \tag{13}$$

Fig. 17 illustrates by means of real images the result of the carrying out of this inference. The scheme is the one of a non-recurrent-temporal ALI, as there is no recurrence and the input receptive field extends over the time axis.

3.1.2. Motion-charge-memory calculation. Recurrent-temporal ALI

As we stated before, in this subtask motion-charge memory is calculated by means of accumulative computation on the negative of property motion presence associated to the accumulation process. Out of accumulative computation operation modes, the one used in this case is the LSR (length–speed ratio) mode (Fernández et al., 2003). The property measured in this case is equivalent to 'no motion' at pixel of coordinates $[x,y]$ at instant t.

The functioning mode is explained by means of the inferences shown in Fig. 19. The static roles shown have the following meanings: $\mathrm{Ch}_{\min}$ and $\mathrm{Ch}_{\max}$ are now the minimum and maximum value, respectively, that the values stored in the motion charge memory can reach, and C_{MM} is now the charge-increment value. Notice that in D_{MM}, (formerly D) the decrease-charge value does not appear explicitly, as we consider that $D_{\mathrm{MM}}=\mathrm{Ch}_{\max}$. The idea behind the LSR is that if there is no motion on pixel $[x,y]$ charge value $\mathrm{Ch}_{\mathrm{MM}}[x,y,t]$

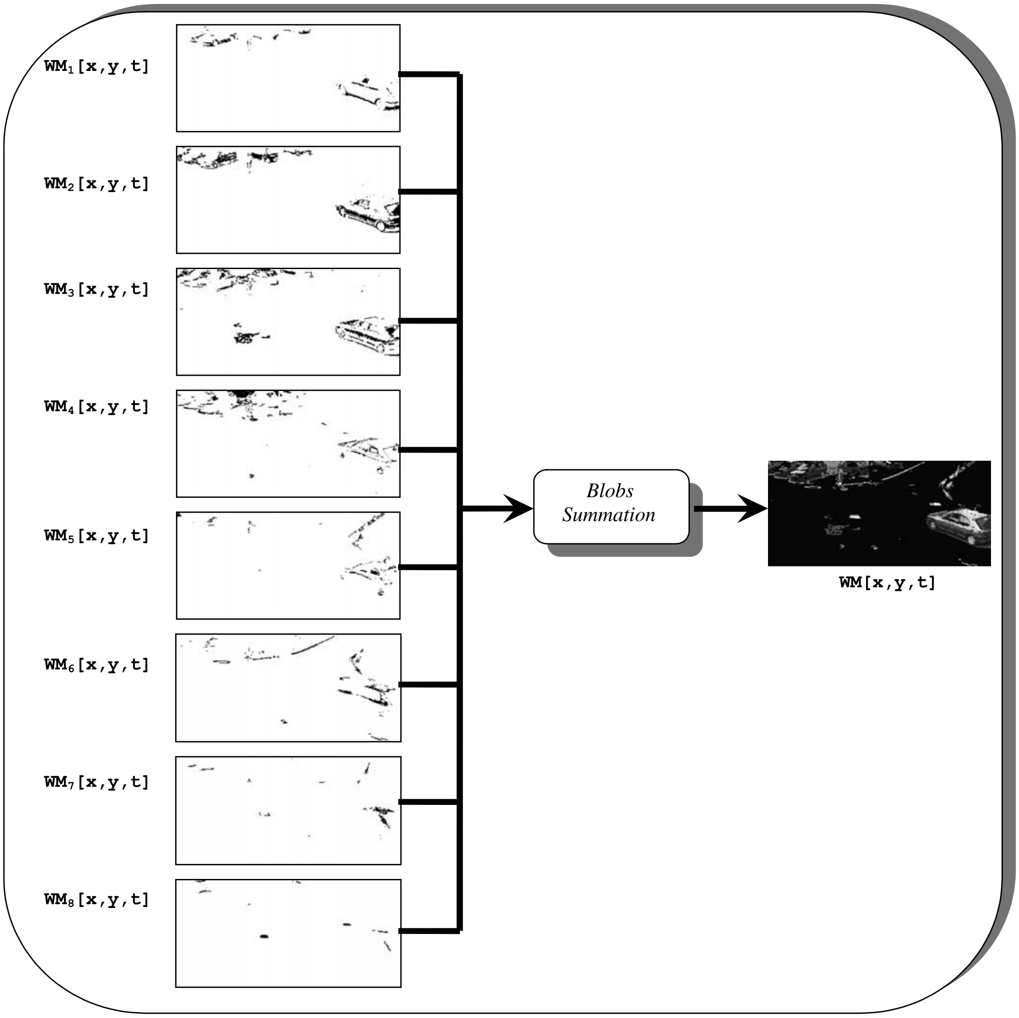


Fig. 14. Application of ‘blobs summation’ to the running example.

goes increasing up to $\mathrm{Ch}_{\max}$, and if motion exists there is a complete discharge (the charge value is given, value $\mathrm{Ch}_{\min}$).

Points where movement has existed recently are charged between the complete discharge, $\mathrm{Ch}_{\min}$, and the maximum charge, $\mathrm{Ch}_{\max}$, being closer to $\mathrm{Ch}_{\min}$ the more recent the movement has taken place. On the contrary, it will take a value closer to $\mathrm{Ch}_{\max}$ the more time has elapsed since motion has been detected on this point (Fig. 18). Thus, charge value $\mathrm{Ch}_{\mathrm{MM}}[x,y,t]$ represents a measure of time elapsed since the last significant variation in brightness on image pixel $[x,y]$ (Fig. 19).

Inference evaluate performs the calculus of values Ch1 and Ch2, which are the possible values of the motion charge memory on the centre at different time intervals. Next, by means of inference select the output value, $\mathrm{Ch}_{\mathrm{MM}}[x,y,t]$, which takes one of the values of set {Ch1,Ch2} depending on the result of Mov[x,y,t], is obtained. Thus, the selection criterion is:

$$\begin{aligned}\mathrm{Ch1} &= \mathrm{Ch}_{\min}\\ \mathrm{Ch2} &= \min(\mathrm{Ch}_{\mathrm{MM}}[x,y,t-1] + C_{\mathrm{MM}}, \mathrm{Ch}_{\max})\end{aligned} \tag{14}$$

Ch1 represents the case for complete discharge, whilst Ch2 is the case when the charge value is augmented when no motion is detected. Inference select selects as output value Ch1 if Mov[x,y,t], motion presence, is 1, and Ch2 in the other case.

$$\mathrm{Ch}_{\mathrm{MM}}[x,y,t] = \begin{cases}\mathrm{Ch1}, & \text{if } \mathrm{Mov}[x,y,t] = 1\\ \mathrm{Ch2}, & \text{if } \mathrm{Mov}[x,y,t] = 0\end{cases} \tag{15}$$

The scheme of the subtask is the one of a recurrent-temporal ALI, as for the production of output $\mathrm{Ch}_{\mathrm{MM}}[x,y,t]$ the proper response at the previous time instant $\mathrm{Ch}_{\mathrm{MM}}[x,y,t-1]$ is provided. In the next sections the way of calculating the velocity and the acceleration from the motion charge-memory is described.

3.1.3. Velocity calculation

Velocity calculation is obtained in two steps. Firstly, velocities in Cartesian x and y axis are calculated; then, the module and the angle of the velocity that the last moving object held when passing on coordinate point $[x,y]$ is calculated. Velocity calculation is performed starting from the values stored in the motion-charge memory, as explained in Table 2.

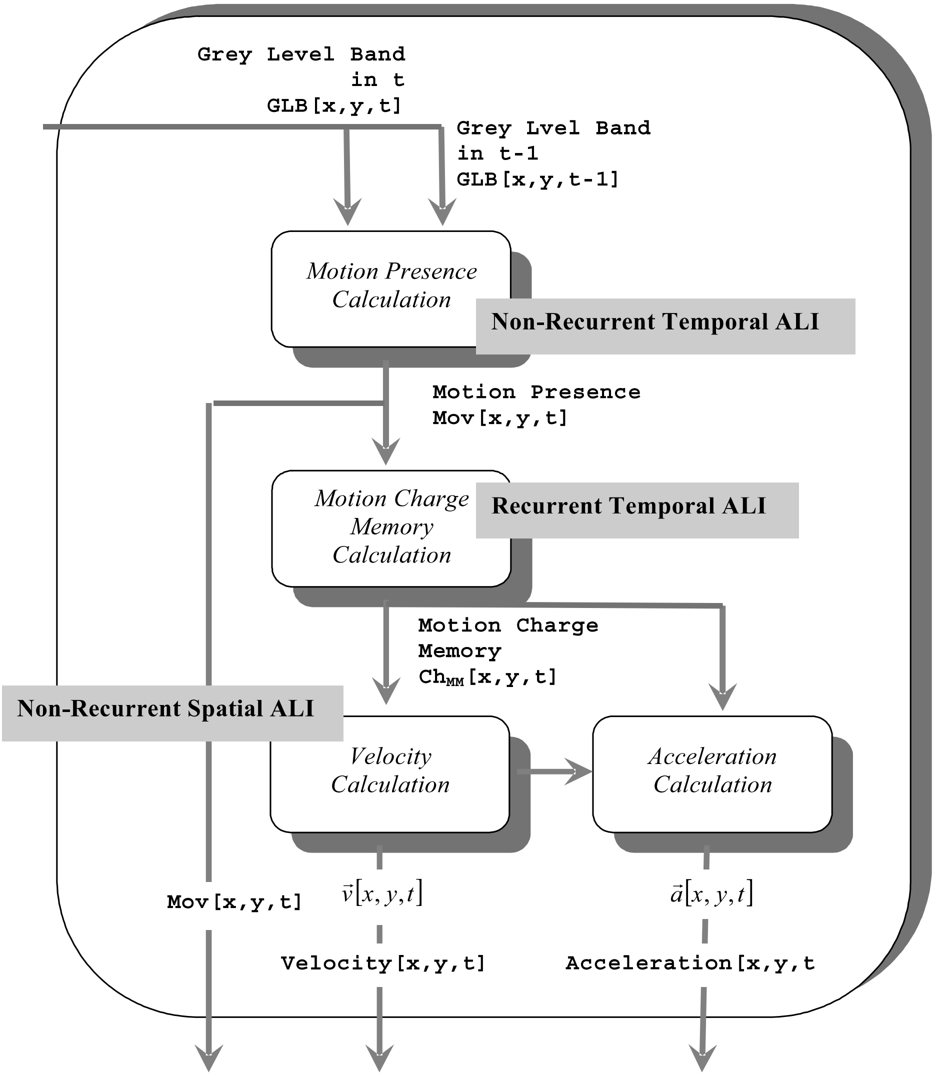
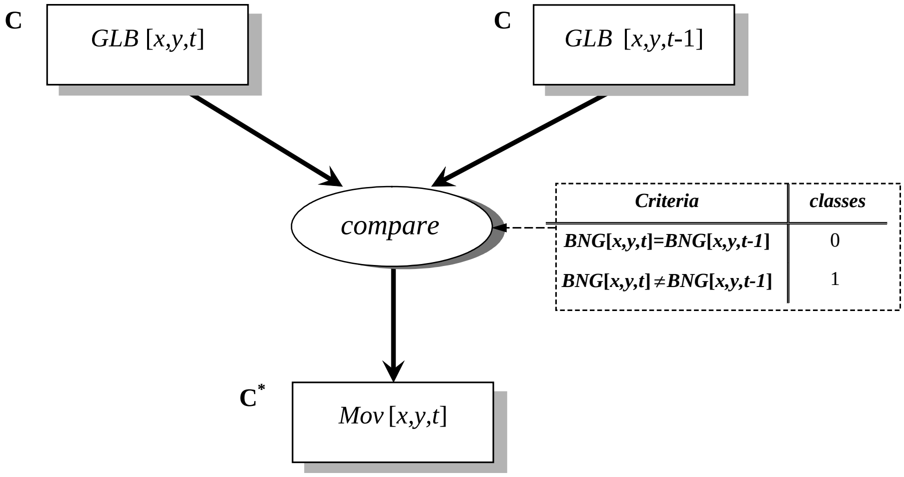


Fig. 15. Scheme for 'motion-feature extraction'.



Criteria	classes
BNG[x,y,t]=BNG[x,y,t-1]	0
BNG[x,y,t]≠BNG[x,y,t-1]	1

Fig. 16. Inference compare used for 'motion-presence calculation'.

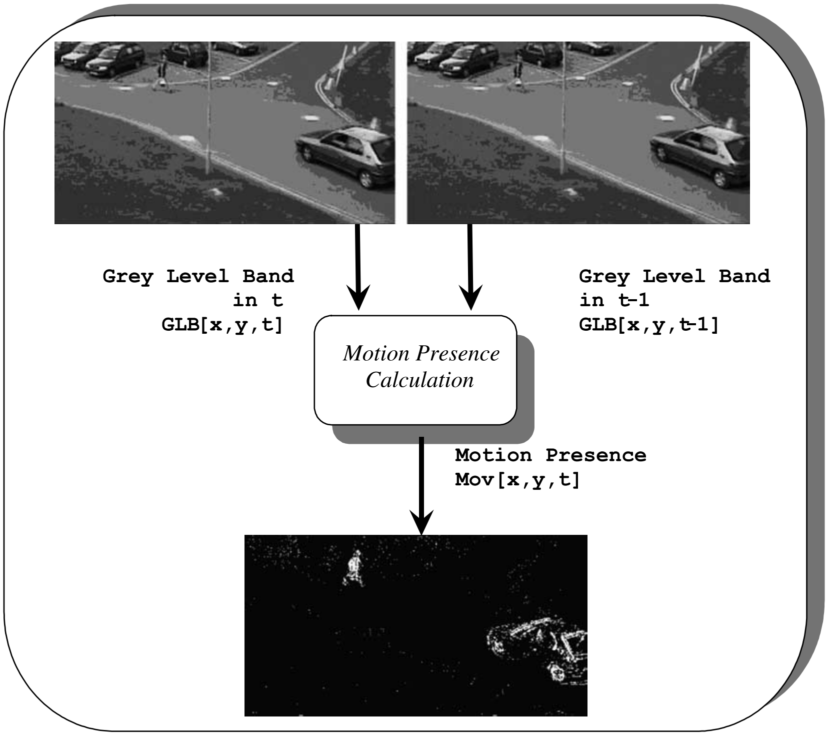


Fig. 17. Illustration of the effect of the compare inference on 'motion-presence calculation'.

Remember that the charge value $\mathrm{Ch_{MM}}[x,y,t]$ is proportional to the time spent since the last significant illumination change on image pixel $[x,y]$. It is important to highlight that the velocity obtained from motion-charge memory is not the velocity of an object point that occupies pixel $[x,y]$ in time t, but rather the velocity of an object point that caused motion-presence detection when it passed over pixel $[x,y]$ a number of

$$k = \frac{\mathrm{Ch_{MM}}[x,y,t] - \mathrm{Ch_{min}}}{C_{\mathrm{MM}}}$$

time units before. Thus, notice that motion-charge memory shows the same value for all those pixels where a simultaneous motion occurred at a given time.

3.1.3.1. Calculation of v_x. Non-recurrent-spatial ALI. First of all, in order to calculate velocity in x-axis, charge value in $[x,y]$, where an object is currently passing, is compared to charge value in another coordinate of the same row $[x+l,y]$, where the same object is passing. In the best of the cases, that is to say, when both values are different from $\mathrm{Ch_{max}}$, the time that elapsed since motion was lastly detected in instant $t-k_{[x,y]}\Delta t$ at $[x,y]$ up to the time when motion was detected in instant $t-k_{[x+l,y]}\Delta t$ in $[x+l,y]$ can be calculated as:

$$\begin{aligned}\mathrm{Ch_{MM}}[x,y,t] - \mathrm{Ch_{MM}}[x+l,y,t] \\ = (\mathrm{Ch_{min}} + k_{[x,y]}C_{\mathrm{MM}}) - (\mathrm{Ch_{min}} + k_{[x+l,y]}C_{\mathrm{MM}}) \\ = (k_{[x,y]} - k_{[x+l,y]})C_{\mathrm{MM}} \qquad (16)\end{aligned}$$

This computation can obviously not be performed if any of both values are $\mathrm{Ch_{max}}$, as we do not know how many time

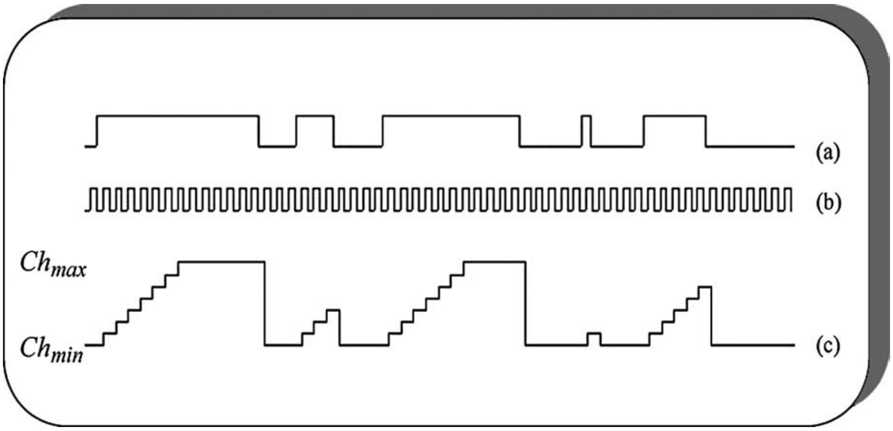


Fig. 18. Illustration of the accumulative computation model, where we represent, for a pixel, (a) the thresholded-binary-input-image-sequence value, (b) the clock signal, and (c) the charge value in LSR mode.

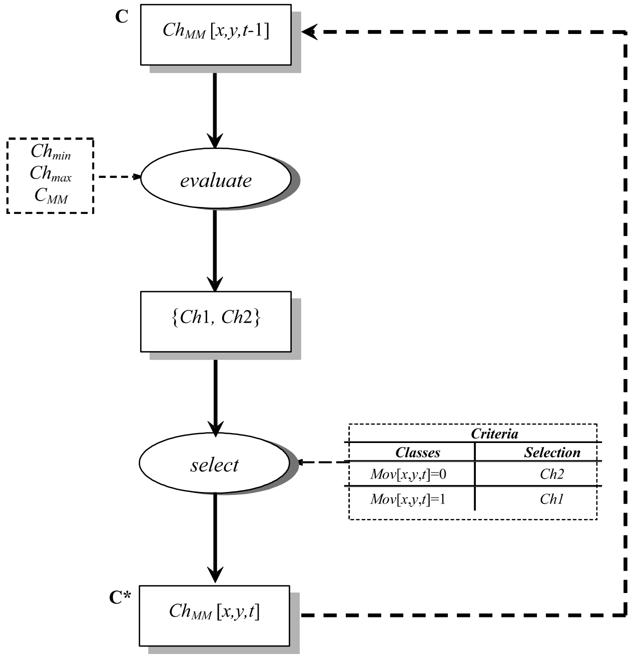


Fig. 19. Inferential scheme for ‘motion-charge-memory calculation’.

intervals have elapsed since the last movement. Hence, for valid charge values, it is known that:

$$\Delta t = \frac{(k_{[x,y]} - k_{[x+l,y]})C_{\mathrm{MM}}}{C_{\mathrm{MM}}} = k_{[x,y]} - k_{[x+l,y]} \tag{17}$$

And, as velocity can always be obtained as $v_x = ((x/(t) = (l/\Delta t)$, finally:

$$v_x[x,y,t] = \frac{C_{\mathrm{MM}} l}{\mathrm{Ch}_{\mathrm{MM}}[x,y,t] - \mathrm{Ch}_{\mathrm{MM}}[x+l,y,t]} \tag{18}$$

That is to say, the velocity in the x-axis of a pixel [x,y], understood as the velocity in the x-axis that the last moving object had on coordinate [x,y], is calculated as the value of the charge increment by the space in pixels to pixel [$x+l$,y], l, divided by the difference of charges between both pixels.

After having introduced the general theory, let us focus in detail on this subtask, whose inferential scheme is shown in Fig. 20. The calculation of the velocity in the x-axis, $v_x[x,y,t]$, is restricted to those pixels whose value in motion-charge memory is of real interest to the subtask. For this, it has been included a parameter, called evaluation value, v_{eval}, which will limit the range of the time elapsed since the last motion. The evaluation value is defined through the formula (19):

$$v_{\mathrm{eval}} = \mathrm{Ch}_{\min} + kC_{\mathrm{MM}} < \mathrm{Ch}_{\max} \tag{19}$$

where value k is selected in such a way that $v_{\mathrm{eval}} < \mathrm{Ch}_{\max}$ has to be fulfilled. That is to say, velocity is only calculated at points, which have detected motion in the last k time intervals. In the rest of the points there is no interest in calculating the velocity, even if motion has been detected. Thus, v_{eval} is directly proportional to the charge increment. Generally speaking, with a little value of k, and hence with a little v_{eval}, the system works with the most recent motion information, whilst with a great k (great v_{eval}) the system plays with more historic information. In the generic task of dynamic-and selective-visual attention, where it is important to capture attention on everything that is moving at each instant, it is recommended to work with low values of v_{eval}.

Next the roles and inferences shown in Fig. 20 are introduced. Role v_{eval}, evaluation value, is the maximum value permitted as charge value for the centre, $\mathrm{Ch}_{\mathrm{MM}}[x,y,t]$, to calculate its velocity. Thus, if charge value of the centre is valid, that value is accepted. On the opposite side, an undefined value, v_{undef}, which is the value given to pixels where the velocity cannot be calculated, is assigned. Role C_{MM} is once more the charge increment value and role $\mathrm{Ch}_{\max}$ is the maximum charge value used to calculate the motion charge memory.

$$z_{\mathrm{c}} = \begin{cases} \mathrm{Ch}_{\mathrm{MM}}[x,y,t], & \text{if } \mathrm{Ch}_{\mathrm{MM}}[x,y,t] \le v_{\mathrm{eval}} = kC_{\mathrm{MM}} < \mathrm{Ch}_{\max} \\ v_{\mathrm{undef}}, & \text{otherwise} \end{cases} \tag{20}$$

if considering that $\mathrm{Ch}_{\min} = 0$.

The periphery of each central element [x,y] is now the closest neighbour to the right, that is to say, [$x+1$,y]. Thus, the receptive space has been restricted to $l=1$. The validation of the periphery charge is not as restrictive as for the centre. This way:

$$z_{\mathrm{p}} = \begin{cases} \mathrm{Ch}_{\mathrm{MM}}[x+1,y,t], & \text{if } \mathrm{Ch}_{\mathrm{MM}}[x+1,y,t] < \mathrm{Ch}_{\max} \\ v_{\mathrm{undef}}, & \text{otherwise} \end{cases} \tag{21}$$

The last calculus for the velocity in the x-axis is eventually:

$$v_x[x,y,t] = \begin{cases} v_{\mathrm{undef}}, & \text{if } (z_{\mathrm{c}} = z_{\mathrm{p}}) \vee (z_{\mathrm{c}} = v_{\mathrm{undef}}) \vee (z_{\mathrm{p}} = v_{\mathrm{undef}}) \\ \dfrac{C_{\mathrm{MM}}}{z_{\mathrm{c}} - z_{\mathrm{p}}}, & \text{otherwise} \end{cases} \tag{22}$$

The scheme is a non-recurrent-spatial ALI, as the input receptive field extends over the space, in particular over the x-axis.

Table 2
Relation between the value in motion-charge memory and motion detection

Value in motion-charge memory	Explanation
$\mathrm{Ch}_{\mathrm{MM}}[x,y,t] = \mathrm{Ch}_{\min}$	Motion is detected at pixel [x,y] in t. Value in memory is the minimum charge value
$\mathrm{Ch}_{\mathrm{MM}}[x,y,t] = \mathrm{Ch}_{\min} + kC_{\mathrm{MM}} < \mathrm{Ch}_{\max}$	No motion is detected at pixel [x,y] in t. Motion was detected for the last time in $t-k\ \Delta t$. After k charge increments the maximum charge has not yet been reached
$\mathrm{Ch}_{\mathrm{MM}}[x,y,t] = \mathrm{Ch}_{\max}$	No motion is detected at pixel [x,y] in t. We do not know when motion was detected for the last time. Value in memory is the maximum charge value

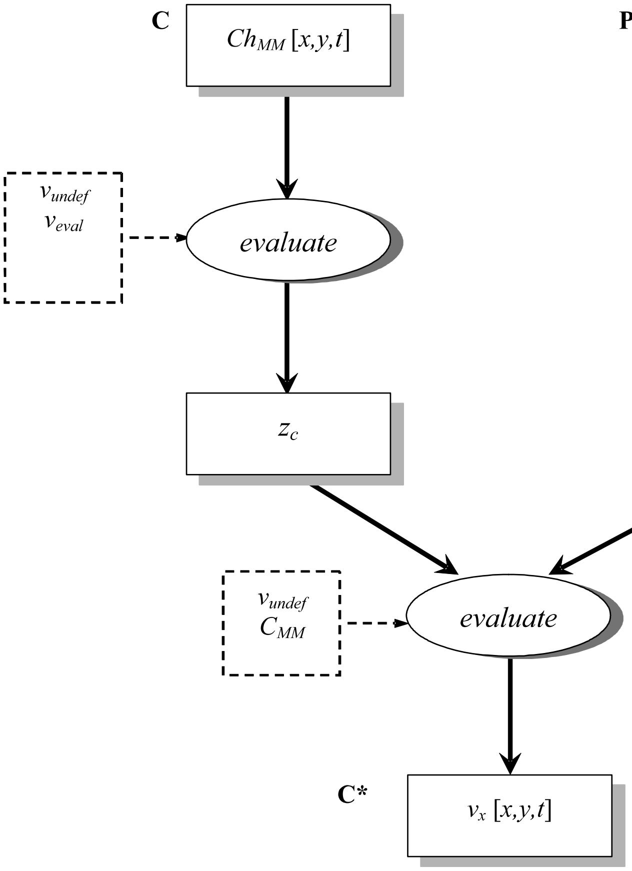


Fig. 20. Inferential scheme for 'calculation of v_x'.

3.1.3.2. Calculation of v_y. Non-recurrent-spatial ALI. In the same way the velocity in y, $v_y[x,y,t]$, is obtained from the motion-charge memory. The reasoning is similar, just changing neighbour of the same row by neighbour of the same column. The equations are in this case:

$$z_c = \begin{cases} \mathrm{Ch_{MM}}[x,y,t], & \text{if } \mathrm{Ch_{MM}}[x,y,t] \le v_{\mathrm{eval}} = kC_{\mathrm{MM}} < \mathrm{Ch_{max}} \\ v_{\mathrm{undef}}, & \text{otherwise} \end{cases} \tag{23}$$

$$z_p = \begin{cases} \mathrm{Ch}[x,y+1,t], & \text{if } \mathrm{Ch}[x,y+1,t] < \mathrm{Ch_{max}} \\ v_{\mathrm{undef}}, & \text{otherwise} \end{cases} \tag{24}$$

$$v_y[x,y,t] = \begin{cases} v_{\mathrm{undef}}, & \text{if } (z_c = z_p) \vee (z_c = v_{\mathrm{undef}}) \vee (z_p = v_{\mathrm{undef}}) \\ \dfrac{C}{z_c - z_p}, & \text{otherwise} \end{cases} \tag{25}$$

3.1.3.3. Calculation of the module and the angle of the velocity. This inference gets the values for the module of the velocity at each pixel $[x,y]$ in time t, $|\vec{v}[x,y,t]|$, and the angle of the velocity, $\beta[x,y,t]$, from the values of the velocity in x-axis, $v_x[x,y,t]$ and the velocity in y-axis, $v_y[x,y,t]$.

As you may grasp, in Fig. 21 there is only one inference, *evaluate*, where the classic computation for the module and the angle are performed:

$$\beta[x,y,t] = \arctan\frac{v_y[x,y,t]}{v_x[x,y,t]} \tag{26}$$

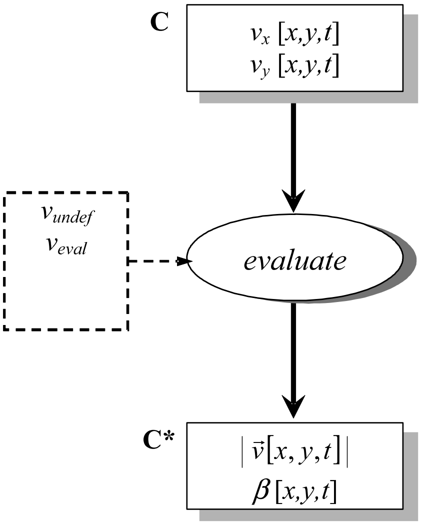


Fig. 21. Inferential scheme for 'calculation of the module and the angle of the velocity'.

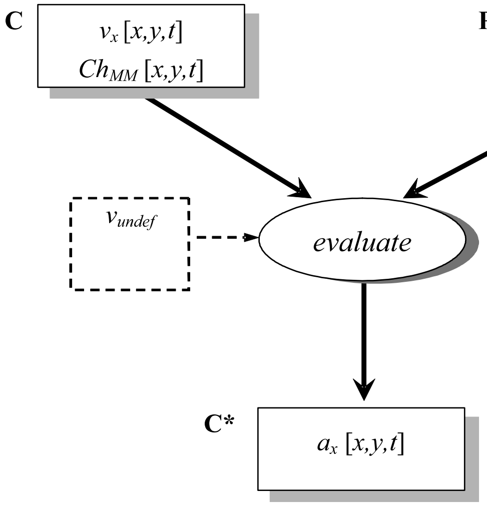


Fig. 22. Inferential scheme for 'calculation of the acceleration in *x*'.

$$|\vec{v}[x,y,t]| = \sqrt{v_x[x,y,t]^2 + v_y[x,y,t]^2} \tag{27}$$

3.1.4. Calculation of the acceleration

Calculation of the acceleration is performed in a similar way to calculation of the velocity. First, the acceleration in the *x*-axis, $a_x[x,y,t]$, and then in the *y*-axis, $a_y[x,y,t]$, are calculated. Then the module and the angle of the vector acceleration are inferred.

3.1.4.1. Calculation of a_x and a_y. The *x* component of the acceleration is calculated from the velocity values on the *x*-axis and the values present in the motion-charge memory (Fig. 22). In a way similar to the calculus of the velocity, now $a_x = \partial v_x / \partial t$ is calculated by operating on the values obtained at v_x. As those points where $v_x = v_{\mathrm{undef}}$ the acceleration will also be undefined.

$$a_x[x,y,t] = \begin{cases} \dfrac{C_{\mathrm{MM}}\left(v_x[x,y,t] - v_x[x+1,y,t]\right)}{\mathrm{Ch}_{\mathrm{MM}}[x,y,t] - \mathrm{Ch}_{\mathrm{MM}}[x+1,y,t]}, & \text{if}(v_x[x,y,t] \neq v_{\mathrm{undef}}) \wedge (v_x[x+1,y,t] \neq v_{\mathrm{undef}}) \\ v_{\mathrm{undef}}, & \text{otherwise} \end{cases} \tag{28}$$

Similarly, the *y*-component of the acceleration is got, substituting $[x+1,y]$ by $[x,y+1]$.

$$a_y[x,y,t] = \begin{cases} \dfrac{C_{\mathrm{MM}}(v_y[x,y,t] - v_y[x,y+1,t])}{\mathrm{Ch}_{\mathrm{MM}}[x,y,t] - \mathrm{Ch}_{\mathrm{MM}}[x,y+1,t]}, & \text{if } (v_y[x,y,t] \neq v_{\mathrm{undef}}) \wedge (v_y[x,y+1,t] \neq v_{\mathrm{undef}}) \\ v_{\mathrm{undef}}, & \text{otherwise} \end{cases} \tag{29}$$

3.1.4.2. Calculation of the module and the angle of the acceleration. This inference gets the values for the module of the acceleration at each pixel $[x,y]$ in time *t*, $|\vec{a}[x,y,t]|$, and the angle of the acceleration, $\alpha[x,y,t]$, from the values of the acceleration in the *x*-axis, $v_x[x,y,t]$ and the acceleration in the *y*-axis, $v_y[x,y,t]$, the same way as in calculation of the module and the angle of the velocity. Formulas are also very similar:

$$\alpha[x,y,t] = \arctan \frac{a_y[x,y,t]}{a_x[x,y,t]} \tag{30}$$

$$|\vec{a}[x,y,t]| = \sqrt{a_x[x,y,t]^2 + a_y[x,y,t]^2} \tag{31}$$

4. Shape-feature extraction

The shape feature extraction subtask, Fig. 23, calculates the values of the various shape properties as indicated by the observer. The input information to extract the shape features are stored as blobs in the working memory and as figures in the attention focus.

For the case of the blobs stored in the working memory, features size, width and height are extracted. As the figures in the attention focus are approximations to complete objects, additionally to extracting the same features than for the blobs, the width–height relation and the compactness features are extracted for the figures as well. Notice that these features are nonsense for the blobs. This is why the shape-feature extraction subtask is split into two subtasks: blob-shape-feature extraction and figure-shape-feature extraction.

4.1. Blob-shape-feature extraction

As indicated, this subtask evaluates for every blob of the working memory the size, $s\_B|v_{\mathrm{etiq}}|$, the width, $w\_B|v_{\mathrm{etiq}}|$ and

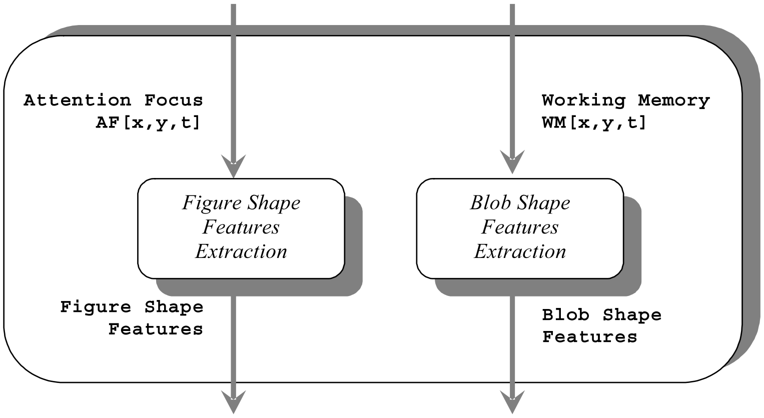


Fig. 23. 'Shape-feature extraction' subtask.

the height, $h\_B|v_{\text{etiq}}|$, features by means of some easy evaluate inferences, where v_{label} is the value of the label associated to the blob. In our ALI scheme, the centre is formed by the whole blob and the periphery is the rest of the image. Eqs. (32)–(34) show how to evaluate those features.

$$s\_B\,[v_{\text{label}}] = \sum_i p_i[x,y,t] | p_i[x,y,t] = \begin{cases} 1, & \text{if } \mathrm{WM}[x,y,t] = v_{\text{label}} \\ 0, & \text{otherwise} \end{cases} \tag{32}$$

$$w\_B[v_{\text{label}}] = \max(x) - \min(x) |\ \mathrm{WM}[x,y,t] = v_{\text{label}} \tag{33}$$

$$h\_B[v_{\text{label}}] = \max(y) - \min(y) | \mathrm{WM}[x,y,t] = v_{\text{label}} \tag{34}$$

The size of the blob is the number of pixels that form the blob (formula (32)). The width of the blob is defined as shown in formula (33), that is to say, as the difference between the coordinate x of the pixel more to the right and the coordinate x of the pixel more to the left of the blob. Equally, as you may notice in Eq. (34), the height of the blob is defined as the difference between the coordinate y of the pixel more at the top and the coordinate y of the pixel more to the bottom of the blob (Fig. 24).

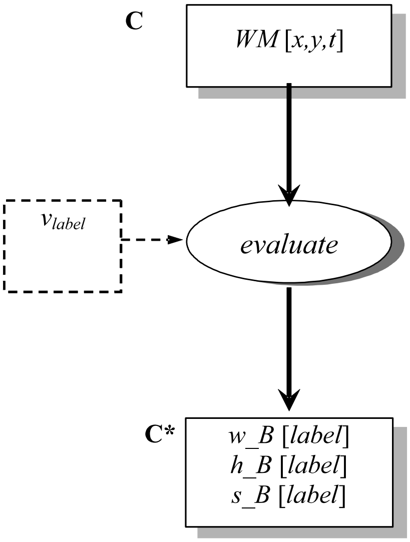


Fig. 24. Inferential scheme for 'blob-shape-feature extraction'.

4.2. Figure-shape-feature extraction

This inference evaluates for every figure of the attention focus the size, $s\_F[v_{\text{etiq}}]$, the width, $w\_F[v_{\text{etiq}}]$, the height, $h\_F[v_{\text{etiq}}]$, the height/width relation, $hw\_F[v_{\text{etiq}}]$, and the compactness, $c\_F[v_{\text{etiq}}]$. This features are shown in Fig. 25, and by means of formulas (35)–(39). Now, v_{label} *is* the label of the studied figure.

The size, the width and the height are calculated in the same way than the features of the same names of the blobs.

$$s\_F[v_{\text{label}}] = \sum_i p_i[x,y,t] | p_i[x,y,t] = \begin{cases} 1, & \text{if } \mathrm{AF}[x,y,t] = v_{\text{label}} \\ 0, & \text{otherwise} \end{cases} \tag{35}$$

$$w\_F\,[v_{\text{label}}] = \max(x) - \min(x) | \mathrm{AF}[x,y,t] = v_{\text{label}} \tag{36}$$

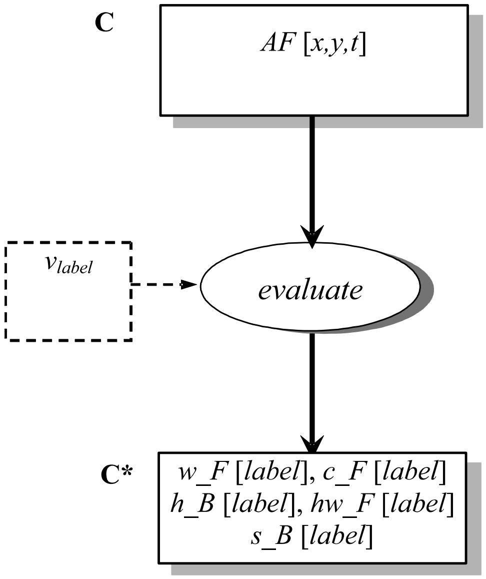


Fig. 25. Inferential scheme for 'figure-shape-feature extraction'.

$$h\_F\,[v_{\mathrm{label}}] = \max(y) - \min(y)|\mathrm{AF}[x,y,t] = v_{\mathrm{label}} \quad (37)$$

The height/width relation is a parameter of great interest, which is used in object classification as it discriminates among different objects of the real world in a very easy way. It is obtained by means of the formula (38).

$$hw\_F[v_{\mathrm{label}}] = \frac{h\_F\,[v_{\mathrm{label}}]}{w\_F\,[v_{\mathrm{label}}]} \quad (38)$$

The compactness of a figure is defined as the quotient between the number of pixels belonging to this figure and the entire number of pixels of the minimal rectangle that includes it:

$$c\_F[v_{\mathrm{label}}] = \frac{s\_F[v_{\mathrm{label}}]}{h\_F[v_{\mathrm{label}}] * w\_F[v_{\mathrm{label}}]} \quad (39)$$

5. Feature integration

The feature integration subtask gets as output the interest map (Fig. 26) by performing an integration of motion features and shape features. As it may be noticed, feature integration is decomposed into five subtasks, namely:

- *Motion-features evaluation:* This subtask values the adequacy of the extracted motion features to the values of the parameters imposed by the observer. The result of this evaluation is a map for each one of the three features (motion presence, velocity and acceleration), where we have the value $v_{\mathrm{activator}}$ in all image pixels that fulfil the restrictions to the motion features imposed by the observer and the value v_{neutral} in all pixels that do not fulfil the restrictions.
- *Blob-shape-features evaluation:* This subtask determines the adequacy of the extracted blob-shape features to the values of the blob-shape parameters imposed by the observer. The result of this evaluation is a unique map where value $v_{\mathrm{activator}}$ is assigned to all pixels of the analyzed blobs that fulfil the restrictions on the blob shape parameters imposed by the observer and the value $v_{\mathrm{inhibitor}}$ in all pixels of the same blobs that do not fulfil the restrictions. Value v_{neutral} is deserved to the rest of the image pixels, that is to

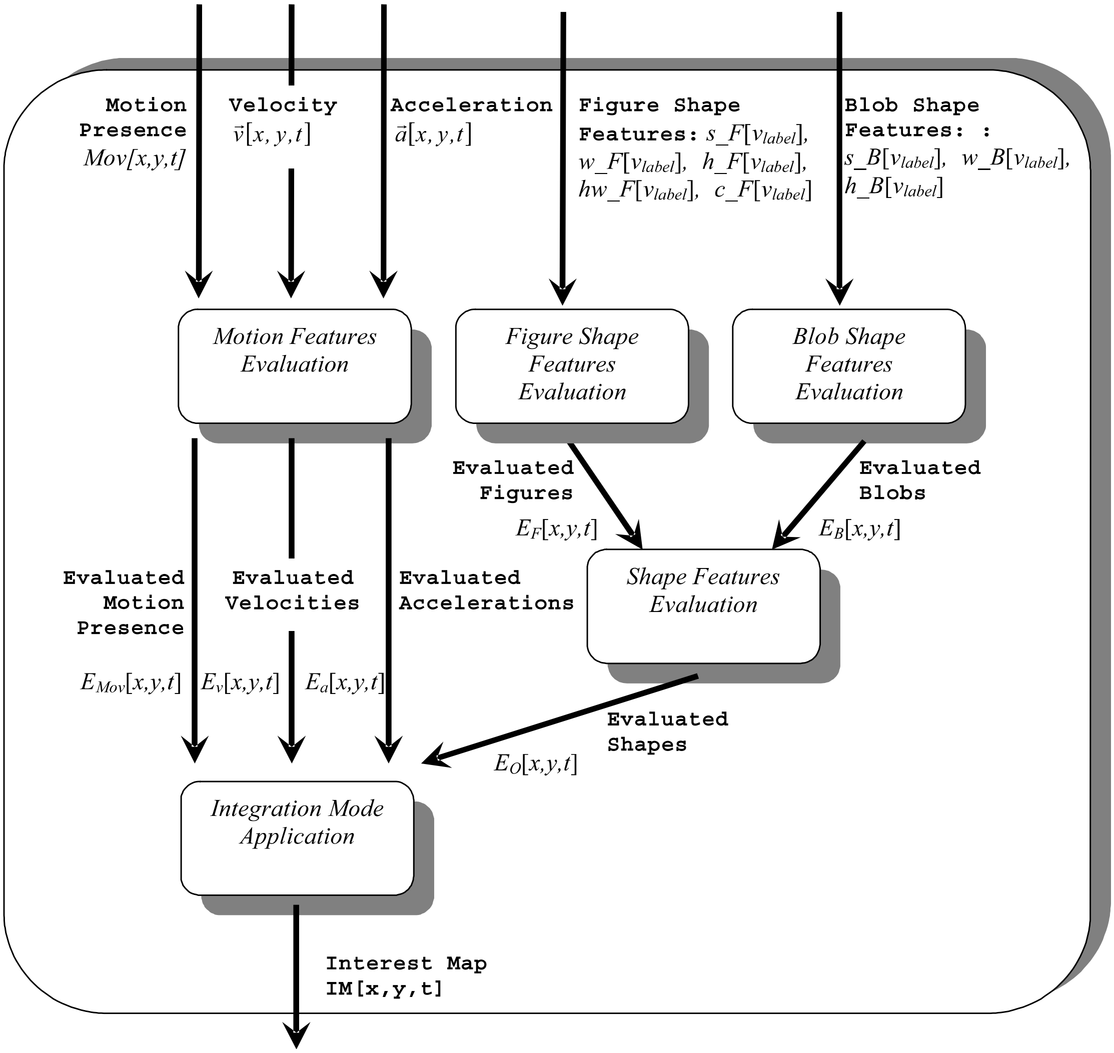

Motion Presence Mov[x,y,t]
Velocity $\vec{v}[x,y,t]$
Acceleration $\vec{a}[x,y,t]$
Figure Shape Features: $s\_F[v_{label}]$, $w\_F[v_{label}]$, $h\_F[v_{label}]$, $hw\_F[v_{label}]$, $c\_F[v_{label}]$
Blob Shape Features: : $s\_B[v_{label}]$, $w\_B[v_{label}]$, $h\_B[v_{label}]$
Motion Features Evaluation
Figure Shape Features Evaluation
Blob Shape Features Evaluation
Evaluated Figures
$E_F[x,y,t]$
Evaluated Blobs
$E_B[x,y,t]$
Shape Features Evaluation
Evaluated Motion Presence
Evaluated Velocities
Evaluated Accelerations
$E_{Mov}[x,y,t]$
$E_v[x,y,t]$
$E_a[x,y,t]$
Evaluated Shapes
$E_O[x,y,t]$
Integration Mode Application
Interest Map IM[x,y,t]

Fig. 26. The ‘feature integration’ subtask.

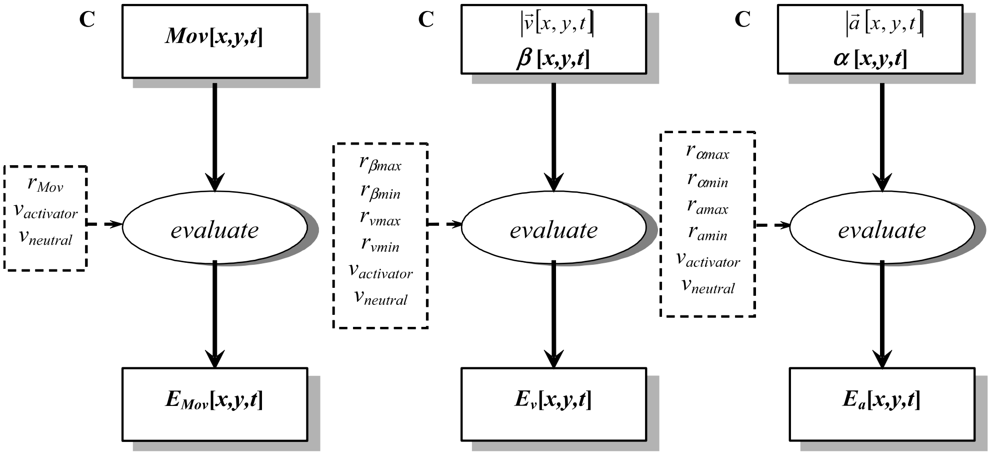


Fig. 27. Inferential scheme for 'motion-features evaluation'.

say, to those pixels that do not belong to the subtask's input blobs.

- *Figure-shape-features evaluation*: This subtask works exactly the same way than the previous one does, but it is applied to figures instead of blobs.
- *Shape-features evaluation*: This subtask combines the outputs of the two previous subtasks into a new map where the following priority relation is imposed: $v_{\text{inhibitor}} > v_{\text{activator}} > v_{\text{neutral}}$, when comparing the values of a same pixel in both input maps.
- *Integration-mode application*: The guidelines of the observer not only establish the superior and inferior limits of all dynamic and shape features, but also determine the so-called integration mode. This integration mode, *M*, assigns more importance to the fulfilment of the input values of a few inputs than to others. Therefore, it is highly dependent on specific DSVA applications.

5.1. Motion-features evaluation

Motion-features evaluation performs on motion presence, Mov[x,y,t], velocity, $\vec{v}[x,y,t]$, and acceleration, $\vec{a}[x,y,t]$, obtaining a different map for each of these evaluations. In Fig. 27 the scheme of this evaluation is shown.

The evaluation of motion presence is calculated from the map related to the feature motion presence, by means of comparing the value of the feature at each pixel to the imposed restriction, r_{Mov}, which may have the values 0 for 'no motion' and 1 for 'motion'.

$$E_{\text{Mov}}[x,y,t] = \begin{cases} v_{\text{activator}}, & \text{if } \text{Mov}[x,y,t] = r_{\text{Mov}} \\ v_{\text{neutral}}, & \text{otherwise} \end{cases} \tag{40}$$

In the other two maps the result of verifying whether the value of the feature falls between the inferior and superior limits established by the observer for that property is obtained.

$$E_v[x,y,t] = \begin{cases} v_{\text{activator}}, & \text{if}(r_{v\ \min} \leq |\vec{v}[x,y,t]| \leq r_{v\ \max}) \\ & \quad \wedge (r_{\beta\ \min} \leq \beta[x,y,t] \leq r_{\beta\ \max}) \\ v_{\text{neutral}}, & \text{otherwise} \end{cases} \tag{41}$$

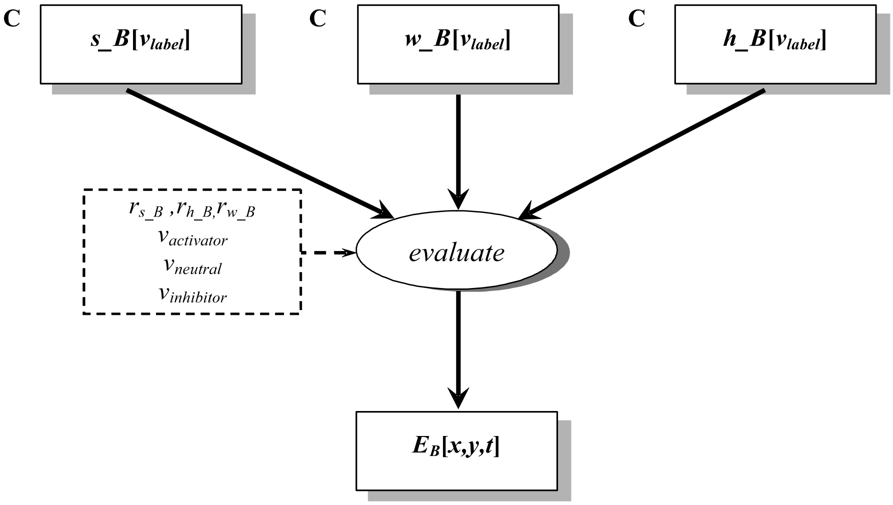


Fig. 28. Inferential scheme for 'blob-shape-features evaluation'.

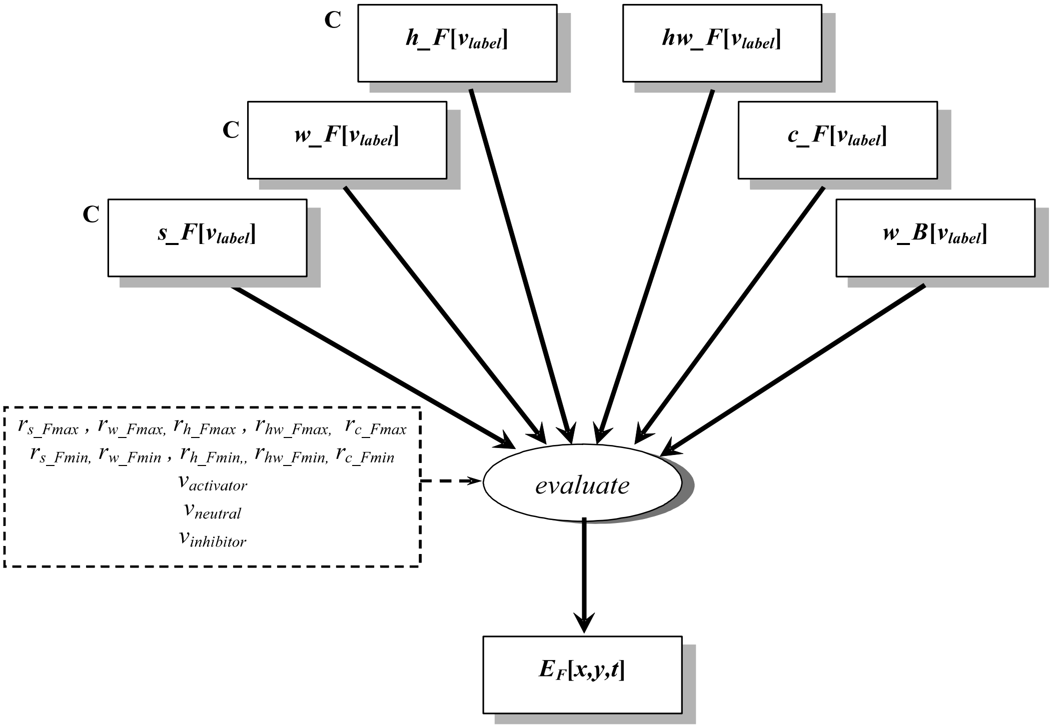


Fig. 29. Inferential scheme for 'figure-shape-features evaluation'.

$$E_a[x,y,t] = \begin{cases} v_{\text{activator}}, & \text{if}(r_{a_{\min}} \le |\vec{a}[x,y,t]| \le r_{a_{\max}}) \\ & \quad \wedge (r_{\alpha_{\min}} \le \alpha[x,y,t] \le r_{\alpha_{\max}}) \\ v_{\text{neutral}}, & \text{otherwise} \end{cases} \tag{42}$$

In all the cases, the fulfilment of the restrictions leads to value $v_{\text{activator}}$ and the non-fulfilment to v_{neutral}.

5.2. Blob-shape-features evaluation

Blob-shape-features evaluation, as offered in the inferential scheme of Fig. 28, consists of one *evaluate*. All evaluations of the blob-shape features are grouped into a unique map.

Again, the result of the verification if the values of all features are below a superior limit established by the observer in order to obtain that property is gotten. The fulfilment of all restrictions leads again to value $v_{\text{activator}}$ and the non-fulfilment of any of the restrictions to $v_{\text{inhibitor}}$. Value v_{neutral} is assigned to all pixels that do not belong to any blob.

$$E_B[x,y,t] = \begin{cases} v_{\text{activator}}, & \text{if}\,[x,y] \in \cup_i z_i | (s\_B[v_{\text{label}}] \le r_{s\_B}) \\ & \wedge (w\_B[v_{\text{label}}] \le r_{w\_B}) \wedge (h\_B[v_{\text{label}}] \le r_{h\_B}) \\ v_{\text{inhibitor}}, & \text{if}\,[x,y] \in \cup_i z_i | (s\_B[v_{\text{label}}] > r_{s\_B}) \\ & \vee (w\_B[v_{\text{label}}] > r_{w\_B}) \vee (h\_B[v_{\text{label}}] > r_{h\_B}) \\ v_{\text{neutral}}, & \text{if}\,[x,y] \notin \cup_i z_i \end{cases} \tag{43}$$

5.3. Figure-shape-features evaluation

Figure-shape-features evaluation is very similar to blob-shape-features evaluation. The main difference is the number of parameters used, and hence the number of restrictions imposed (Fig. 29).

Once again, all evaluations of the figure shape features are grouped into a unique map, where the result of the verification if the values of all features for each pixel are between a superior and inferior limit established by the observer to obtain that property. This way, each element of $E_{\text{F}}[x,y,t]$ takes the value of v_{neutral}, $v_{\text{activator}}$ or $v_{\text{inhibitor}}$ depending on expression (44).

$$E_F[x,y,t] = \begin{cases} v_{\text{neutral}}, & \text{if}\,[x,y] \notin \cup_i v_i \\ v_{\text{activator}}, & \text{if}\,[x,y] \in \cup_i v_i | (r_{s\_F_{\min}} \le s\_F[v_{\text{label}}] \\ & \le r_{s\_F_{\max}}) \wedge (r_{w\_F\,\min} \le w\_F[v_{\text{label}}] \\ & \le r_{w\_F\,\max}) \wedge (r_{h\_F\,\min} \le h\_F[v_{\text{label}}] \\ & \le r_{h\_F\,\max}) \wedge (r_{hw\_F\,\min} \le \text{hw}\_F[v_{\text{label}}] \\ & \le r_{hw\_F\,\max}) \wedge (r_{c\_F\,\min} \le c\_F[v_{\text{label}}] \\ & \le r_{c\_F\,\max}) \\ v_{\text{inhibitor}}, & \text{if}\,[x,y] \in \cup_i v_i | (s\_F[v_{\text{label}}] < r_{s\_F_{\min}}) \\ & \vee (s\_F[v_{\text{label}}] > r_{s\_F_{\max}}) \vee (w\_F[v_{\text{label}}] \\ & < r_{w\_F\,\min}) \vee (w\_F[v_{\text{label}}] > r_{w\_F\,\min}) \\ & \vee (h\_F[v_{\text{label}}] < r_{h\_F\,\min}) \vee (h\_F[v_{\text{label}}] \\ & > r_{h\_F\,\max}) \vee (hw\_F[v_{\text{label}}] < r_{hw\_F\,\min}) \\ & \vee (hw\_F[v_{\text{label}}] > r_{hw\_F\,\max}) \\ & \vee (c\_F[v_{\text{label}}] < r_{c\_F\,\min}) \vee (c\_F[v_{\text{label}}] \\ & > r_{c\_F\,\max}) \end{cases} \tag{44}$$

Remember that the fulfilment of all the restrictions leads to the value $v_{\text{activator}}$ and the not fulfilment of anyone of them gives place to $v_{\text{inhibitor}}$. The value v_{neutral} is assigned to the pixels that do not belong to any figure.

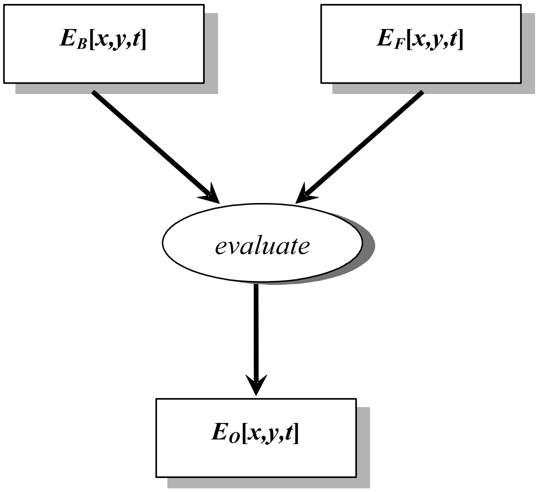


Fig. 30. Inferential scheme for 'shape-features evaluation'

5.4. Shape-features evaluation

Shape-features evaluation, as you may notice in Fig. 30, consists of the fusion into a unique map, E_{O}, of the maps coming from blob-shape-features evaluation, E_{B}, and figure-shape-features evaluation, E_{F}.

The algorithm associated to this inference is shown in Table 3, where it may be observed that value $v_{\mathrm{inhibitor}}$ occupies the first place on $v_{\mathrm{activator}}$, which in turn occupies the first place on v_{neutral}. In other words, it is enough that $E_{\mathrm{B}}[x,y,t]$ or $E_{\mathrm{F}}[x,y,t]$ take value $v_{\mathrm{inhibitor}}$ so that $E_{\mathrm{O}}[x,y,t]$ also takes the same value, $v_{\mathrm{inhibitor}}$. If both evaluations, in which on $E_{\mathrm{B}}[x,y,t]$ and $E_{\mathrm{F}}[x,y,t]$ are v_{neutral}, *Shape-Features Evaluation* also takes value v_{neutral}. In the rest of the cases value $v_{\mathrm{activator}}$ is applied, which corresponds to the fact that the two evaluations are activated or that one of them is activated and the other one takes value v_{neutral}.

5.5. Integration-mode application

After integration-mode application the interest map, where for every image pixel one of the three classes—'activator', 'inhibitor' and 'neutral'—is stored, is finally obtained. The classification is performed, in first place, in accordance with the observer's guidelines in form of restrictions on the extracted features. On the other hand, in addition to establishing the limits for each feature, the guidelines of the observer introduce the so-called integration mode. the integration mode, M, assigns more importance to the fulfilment of the input values of a few inputs than to others and is highly dependent on the specific application (Table 4).

Table 3
Fusion of shape features

$E_{\mathrm{B}}[x,y,t]$	$E_{\mathrm{F}}[x,y,t]$	$E_{\mathrm{O}}[x,y,t]$
$v_{\mathrm{inhibitor}}$	*	$v_{\mathrm{inhibitor}}$
*	$v_{\mathrm{inhibitor}}$	$v_{\mathrm{inhibitor}}$
$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$
$v_{\mathrm{activator}}$	v_{neutral}	$v_{\mathrm{activator}}$
v_{neutral}	$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$
v_{neutral}	v_{neutral}	v_{neutral}

*Any value.

Table 4
Most common integration modes

Integration mode	Output of evaluation for				Classes
	Motion $E_{\mathrm{Mov}}[x,y,t]$	Velocity $E_{\mathrm{v}}[x,y,t]$	Acceleration $E_{\mathrm{a}}[x,y,t]$	Shape $E_{\mathrm{O}}[x,y,t]$	
$M=0$	$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$
	$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$	v_{neutral}	$v_{\mathrm{activator}}$
	*	*	*	$v_{\mathrm{inhibitor}}$	$v_{\mathrm{inhibitor}}$
	Rest of combinations				v_{neutral}
$M=1$	*	*	*	$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$
	$v_{\mathrm{activator}}$	*	*	v_{neutral}	$v_{\mathrm{activator}}$
	v_{neutral}	*	*	$v_{\mathrm{inhibitor}}$	$v_{\mathrm{inhibitor}}$
	v_{neutral}	*	*	v_{neutral}	v_{neutral}
$M=2$	*	*	*	$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$
	*	$v_{\mathrm{activator}}$	*	v_{neutral}	$v_{\mathrm{activator}}$
	*	v_{neutral}	*	$v_{\mathrm{inhibitor}}$	$v_{\mathrm{inhibitor}}$
	*	v_{neutral}	*	v_{neutral}	v_{neutral}
$M=3$	*	*	*	$v_{\mathrm{activator}}$	$v_{\mathrm{activator}}$
	*	*	*	v_{neutral}	v_{neutral}
	*	*	*	$v_{\mathrm{inhibitor}}$	$v_{\mathrm{inhibitor}}$

*Any value.

Mode $M=0$ is the mode that gives an equal importance to all four maps. In integration mode $M=0$, the value of the interest map is $v_{\mathrm{activator}}$ when no restriction is broken. Thus, mode $M=0$ is the most restrictive one. Integration mode $M=1$ takes only motion presence and shape features into consideration, and does not impose restrictions to velocity and acceleration. It has the singularity of giving priority to values $v_{\mathrm{activator}}$ in opposite to other cases. Integration mode $M=2$ only takes into account velocity and shape features, and does not control motion presence and acceleration restrictions. Again, $v_{\mathrm{activator}}$ values have a greater priority than the others. Integration mode $M=3$ only minds of shape features. This mode is only used for tracking purpose once an object has been selected in accordance to the rest of restrictions given by the observer.

6. Attention reinforcement

The mechanisms used to generate the working memory endow the system of sensitivity, as it includes elements associated to interest points ('activators') in the memory at each frame. Unfortunately, in the working memory scene blobs whose shape features do not correspond to those defined by the observer may appear at a time instant t. This is precisely because their shape features have not yet been studied. But, if these blobs-shape features do not really seem to be interesting for the observer, they will appear as 'inhibitors' in $t+1$ in the interest map (now, in $t+1$, their shape features will have been obtained). And this means that in $t+1$ they will disappear from the working memory. Thus, the working memory has to be considered as a noisy memory. Scene blobs appear and disappear at each input image frame, as they fulfil or do not

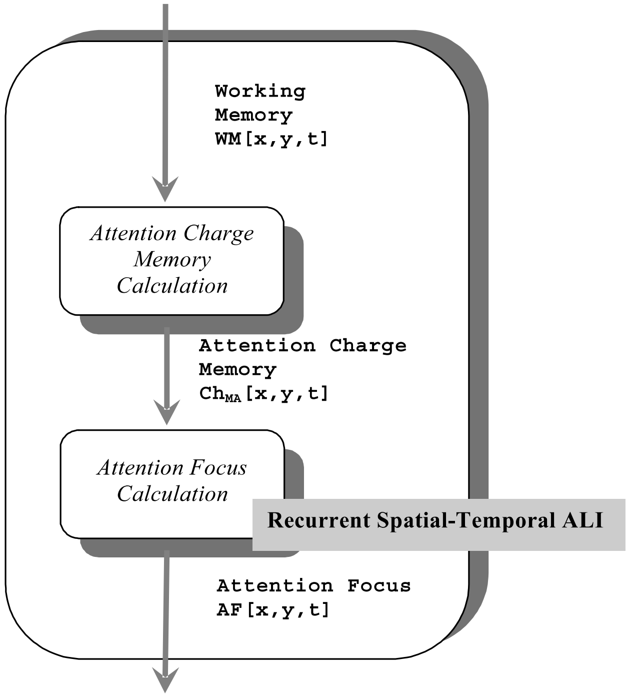

Working
Memory
WM[x,y,t]
Attention Charge Memory Calculation
Attention Charge
Memory
Ch_MA[x,y,t]
Attention Focus Calculation
Recurrent Spatial-Temporal ALI
Attention Focus
AF[x,y,t]

Fig. 31. Scheme for ‘attention reinforcement’.

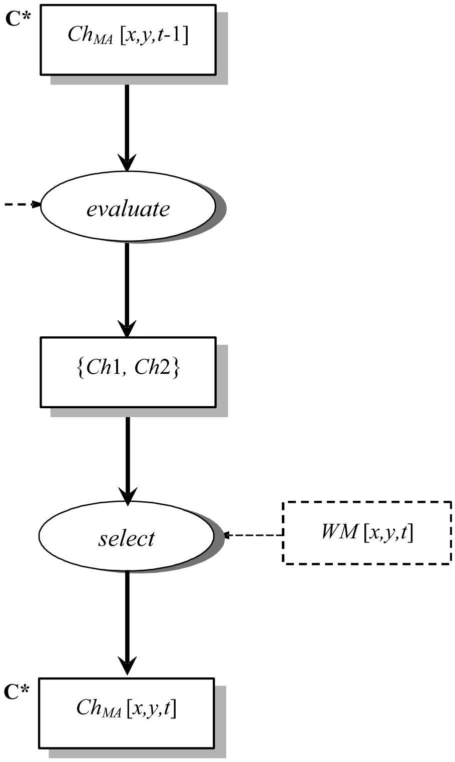


Fig. 32. Inferential scheme for ‘attention-charge-memory calculation’.

fulfil the desired blob-shape features. The same way we have got sensitivity, we need some mechanism to endow stability to the system.

In order to obtain only blobs with the desired features at each frame, attention reinforcement performs an accumulative mechanism followed by a threshold. Accumulation is realized on pixels that have a value different from 0 (pixels that do not belong to labelled blobs) in the working memory. The result of this accumulative process followed by a threshold offers as output the attention focus, AF[*x*,*y*,*t*]. More precisely, pixels that appear with a value different from 0 in the working memory reinforce attention, whilst those that appear with a value 0 diminish the attention value. The process manages to keep activated in a stable way a set of pixels that belong to a group of objects (figures) of the scene that are interesting for the observer.

Fig. 31 shows the decomposition of the attention reinforcement subtask into the attention-charge-memory calculation and attention-focus calculation subtasks.

6.1. Attention-charge-memory calculation

This subtask performs an accumulative computation on the working memory to get the attention-charge memory. As it has been explained before, pixels that belong to a blob of the working memory reinforce attention whilst all the other ones decrease attention. The inferential scheme is offered at Fig. 32.

The accumulative computation takes in this case the form of the Eqs. (45) and (46), based on the more general charge/discharge-accumulative-computation mode, as explained in formula (12). The evaluate inference shows the charge and the discharge and the inference *select* offers the property measured for the calculation.

The static roles have the following meanings: $\mathrm{Ch}_{\mathrm{min}}$ and $\mathrm{Ch}_{\mathrm{max}}$ are the minimum and maximum values, respectively, that the values stored in the attention charge memory can reach, and C_{MA} and D_{MA} are now the charge increment and decrement, respectively, in the memory computation.

$$\begin{aligned}\mathrm{Ch1} &= \max(\mathrm{Ch}_{\mathrm{MA}}[x,y,t-1]-D_{\mathrm{MA}},\mathrm{Ch}_{\mathrm{min}})\\ \mathrm{Ch2} &= \min(\mathrm{Ch}_{\mathrm{MA}}[x,y,t-1]+C_{\mathrm{MA}},\mathrm{Ch}_{\mathrm{max}})\end{aligned} \tag{45}$$

$$\mathrm{Ch}_{\mathrm{MA}}[x,y,t] = \begin{cases}\mathrm{Ch1}, & \text{if } \mathrm{WM}[x,y,t]=0\\ \mathrm{Ch2}, & \text{if } \mathrm{WM}[x,y,t]>0\end{cases} \tag{46}$$

The charge value $\mathrm{Ch}_{\mathrm{MA}}[x,y,t]$ goes increasing up to $\mathrm{Ch}_{\mathrm{max}}$, if pixel [*x*,*y*] belongs to a blob of the working memory, and goes decreasing down to $\mathrm{Ch}_{\mathrm{min}}$ if the pixel does not. Charge value $\mathrm{Ch}_{\mathrm{MA}}[x,y,t]$ represents a measure of the persistency of a blob in the working memory on each image pixel [*x*,*y*].

6.2. Attention-focus calculation. Recurrent-spatial-temporal ALI

This subtask produces, starting from the attention-charge memory, the points that shape the attention focus, labelling the obtained figures. The focus is on the figures, obtained by the union of the connected blobs that have appeared successively in the working memory and whose value in the attention-charge memory is greater or equal to a given threshold, θ. In the output, the label corresponding to the figure is stored; value 0 is

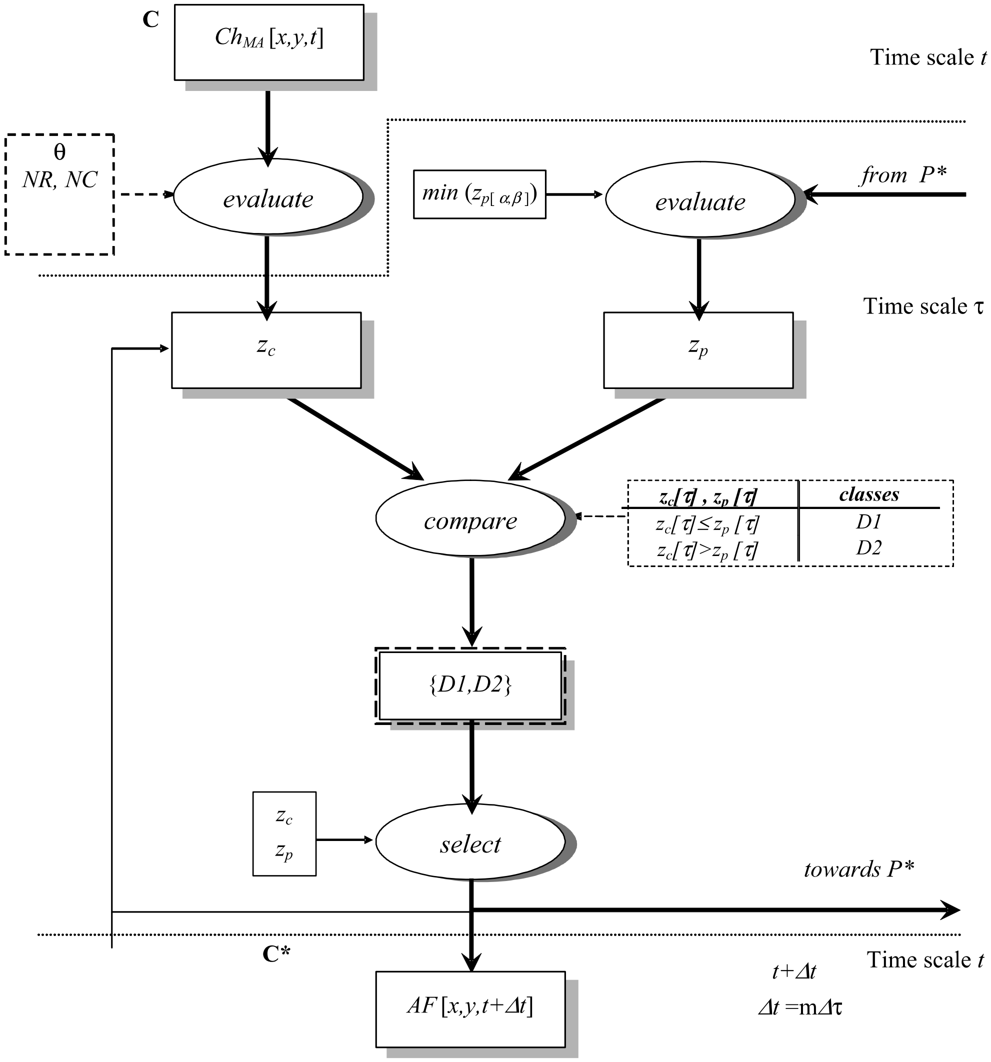


Fig. 33. Inferential scheme for ‘attention focus calculation’.

assigned to all pixels that do not belong to any figure (see inferential scheme of Fig. 33).

The inference evaluate in time scale t, which operates with the input data of the centre, formed by the value of the attention-charge memory at pixel $[x,y]$, assigns to each increment of t an initial and provisional value—yet not agreed with the periphery—corresponding to a function of the coordinate of the pixel, if the charge value overcomes the threshold θ:

$$z_c(t) = \begin{cases} x \times \mathrm{NC} + y + 1, & \text{if } \mathrm{Ch}_{\mathrm{MA}}[x,y,t] > \theta \\ 0, & \text{otherwise} \end{cases} \tag{47}$$

The primitive evaluate, which operates with the output data of the periphery in time scale τ, formed by the eight neighbours of each pixel, obtains the minimum of these values according to formula (48):

$$z_{\mathrm{p}} = \min(z_{\mathrm{p}(\alpha,\beta)})\, \forall\, [\alpha,\beta] \in [x \pm 1, y \pm 1] | ([\alpha,\beta] \neq [x,y]) \wedge (0 < z_{\mathrm{p}(\alpha,\beta)} \le z_{\max}) \tag{48}$$

The primitive compare generates the discrepancy value at each time instant τ by comparing the values of z_{c} y z_{p}, as shown in formula (49):

$$D = \begin{cases} D1, & \text{if } z_{\mathrm{c}} \le z_{\mathrm{p}} \\ D2, & \text{if } z_{\mathrm{c}} > z_{\mathrm{p}} \end{cases} \tag{49}$$

Lastly, here is the inference select that assigns the new value to z_{c}.

$$z_{\mathrm{c}}(\tau) = \begin{cases} z_{\mathrm{c}}(\tau - \Delta\tau), & \text{if } D1 \\ z_{\mathrm{p}}(\tau - \Delta\tau), & \text{if } D2 \end{cases} \tag{50}$$

Once again in time scale t, after the competition phase whose duration is $\Delta t = m\Delta\tau$, the value of the attention focus is obtained; this is the value of the centre.

$$\mathrm{AF}[x,y,t] = Z_c(t + m\Delta\tau) = Z_c(t + \Delta t) \tag{51}$$

7. Conclusions

We have presented the algorithmic lateral inhibition (ALI) method in dynamic and selective visual attention (DSVA) task with application to moving objects detection and labelling. The proposed solution defines a model with two types of processes: bottom-up processes—based on the scene—which enable to extract the pixels, blobs and figures features (feature extraction), and allow to create the blobs (attention building) and figures (attention reinforcement); and top-down processes—based on the object—by means of which from the intention of the observer the search parameters are modified up to satisfying his expectations with regard to the attention focus. These parameters correspond with the integration mode and with the dynamic restrictions—at pixel level—and the shape restrictions—at blob and figure level. The parameters are the input, together with the outputs of the feature extraction subtask, to the feature integration, subtask, engaged in generating the interest map.

The paper shows the convenience of modeling knowledge of tasks and methods in terms of a library of reusable components (inferential verbs 'evaluate', 'compare' and 'select') and a set of input and output roles played by the entities of the application domain. The paper also highlights the possibility to use a PSM, namely the ALI method in any of its forms—recurrent and non-recurrent, temporal, spatial and spatial-temporal—to solve a specific problem in artificial vision where the final configuration of a PSM is always dependent on the particular balance between data and knowledge available for the specific case under consideration. For each one of the subtasks we have illustrated the results of the inferential scheme.

Acknowledgements

This work is supported in part by the Spanish CICYT TIN2004-07661-C02-01 and TIN2004-07661-C02-02 grants.

References

Awh, E., Anllo-Vento, L., & Hillyard, S. A. (2000). The role of spatial selective attention in working memory for locations: Evidence from event-related potentials. *Journal of Cognitive Neuroscience*, *12*(5), 840–847.

Awh, E., & Jonides, J. (2001). Overlapping mechanisms of attention and spatial working memory. *Trends in Cognitive Sciences*, *5*(3), 119–126.

Awh, E., Matsukura, M., & Serences, J. T. (2003). Top-down control over biased competition during covert spatial orienting. *Journal of Experimental Psychology: Human Perception and Performance*, *29*, 52–63.

Breuker, J., & van de Velde, W. (1994). *CommonKADS library for expertise modelling*. Amsterdam: IOS Press.

Fernández, M. A., Fenández-Caballero, A., López, M. T., & Mira, J. (2003). Length–speed ratio (LSR) as a characteristic for moving elements real-time classification. *Real-Time Imaging*, *9*, 49–59.

Fernández, M. A., & Mira, J. (1992). Permanence memory: A system for real time motion analysis in image sequences. IAPR Workshop on Machine Vision Applications. p. 249–252.

Fernández-Caballero, A., Fernández, M. A., Mira, J., & Delgado, A. E. (2003). Spatio-temporal shape building from image sequences using lateral interaction in accumulative computation. *Pattern Recognition*, *36*(5), 1131–1142.

Fernández-Caballero, A., Mira, J., Delgado, A. E., & Fernández, M. A. (2003). Lateral interaction in accumulative computation: A model for motion detection. *Neurocomputing*, *50C*, 341–364.

Fernández-Caballero, A., Mira, J., Férnandez, M. A., & Delgado, A. E. (2003). On motion detection through a multi-layer neural network architecture. *Neural Networks*, *16*(2), 205–222.

Fernández-Caballero, A., Mira, J., Fernández, M. A., & López, M. T. (2001). Segmentation from motion of non-rigid objects by neuronal lateral interaction. *Pattern Recognition Letters*, *22*(14), 1517–1524.

Heinke, D., Humphreys, G. W., & diVirgilo, G. (2002). Modeling visual search experiments: Selective attention for identification model (SAIM). *Neurocomputing*, *44*, 817–822.

Itti, L., Koch, C., & Niebur, E. (1998). A model of saliency-based visual attention for rapid scene analysis. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, *20*, 1254–1259.

Koch, C., & Ullman, S. (1985). Shifts in selective visual attention: Towards the underlying neural circuitry. *Human Neurobiology*, *4*, 219–227.

López, M.T., Fernández, M.A., Fernández-Caballero, A., & Delgado, A. E. (2003). Neurally inspired mechanisms for the dynamic visual attention map generation task. In computational methods in modeling computation (pp. 694–701). Berlin:Springer.

Mira, J., Delgado, A. E., Fernández-Caballero, A., & Fernández, M. A. (2004). Knowledge modelling for the motion detection task: The algorithmic lateral inhibition method. *Expert Systems with Applications*, *27*(2), 169–185.

Mira, J., Fernández, M. A., López, M. T., Delgado, A. E., & Fernández-Caballero, A. (2003). *A model of neural inspiration for local accumulative computation Ninth international conference on computer aided systems theory*. Berlin:Springer, pp. 427–435.

O'Reilly, R. C. (2003). Making working memory work: A computational model of learning in the prefrontal cortex and basal ganglia. ICS technical report 03-03.

O'Reilly, R. C., Braver, T. S., & Cohen, J. D. (1999). A biologically-based computational model of working memory. In models of working memory: Mechanisms of active maintenance and executive control (pp. 375–411). New York: Cambridge University Press.

Phaf, R. H., van der Heijden, A. H. C., & Hudson, P. T. W. (1990). SLAM: A connectionist model for attention in visual selection. *Cognitive Psychology*, *22*(3), 273–341.

Schreiber, A., Akkermans, H., Anjewierden, A., de Hoog, R., Shadbolt, N., van de Velde, W., et al. (2001). *Knowledge engineering and management: The common KADS methodology*. Cambridge, MA: The MIT Press.

Treisman, A. M., & Gelade, G. (1980). A feature-integration theory of attention. *Cognitive Psychology*, *12*, 97–136.

Wolfe, J. M. (1994). Guided search 2.0. A revised model of visual search. *Psychonomic Bulletin and Review*, *1*, 202–238.