Vehicle Detection and Tracking Using the Block Matching Algorithm

Luigi Di Stefano, Enrico Viarani
Department of Electronics, Computer Science and Systems (DEIS)
University of Bologna
Via Risorgimento, 2 - 40136 Bologna
ITALY

Abstract: - The paper describes an approach to vehicle detection and tracking based on the Block Matching Algorithm (BMA), which is the motion estimation algorithm employed in the MPEG compression standard. BMA partitions the current frame in small, fixed size blocks and matches them in the previous frame in order to estimate blocks displacement (referred to as motion vectors) between two successive frames. The vehicle detection and tracking approach is as follows. BMA provides motion vectors, which are then regularised using a Vector Median Filter. After the regularisation step, motion vectors are grouped based on their adjacency and similarity, and a set of vehicles is built for each frame. Finally, the tracking algorithm establishes the correspondences between the vehicles detected in each frame of the sequence, allowing the estimation of their trajectories as well as the detection of new entries and exits. The tracking algorithm is strongly based on the BMA. We consider the BMA output as the basic tracking information associated with each block and combine this already available block-level tracking information with the grouping output so as to achieve the tracking of vehicles.

Key-Words: - Computer vision, traffic monitoring, vehicle tracking *CSCC'99 Proceedings*, Pages:4491-4496

1 Introduction

This paper describes the state of the research carried out at the Department of Electronics Informatics and Systems (DEIS), University of Bologna, on the development of computer vision techniques for urban traffic data recovering. The perspective goal of this activity is the development of a low-cost, vision-based traffic monitoring system capable of fast and trusty fundamental traffic parameters measurements (e.g. low level information such as queue lengths and vehicle turning rates at intersections and higher level information such as assessment of the traffic situation).

The initial goals of this research activity have been identified in the development of fast and reliable algorithms for detecting, tracking and classifying vehicles. In this paper we describe the current vehicle detection and tracking approach.

We have chosen to rely on the extraction of motion from image sequences so as to extract vehicles (i.e. moving objects) from the scene. Our approach is based on the Block Matching Algorithm [11] (abbreviated as BMA), which is used for motion estimation in the MPEG [5] image sequences compression standard.

BMA has been employed for vehicle detection in an application aimed at evaluating turning rates at urban crossroads [4]. Unlike techniques based on spatio-temporal derivatives [9] [2] or background subtraction [10], BMA provides not only detection of moving pixels but also motion estimation (i.e. a motion vector is associated with image points). This improves the discrimination power in case of partially occluding vehicles and crowded traffic scenes.

World-wide diffusion of MPEG makes specialised hardware for BMA easily available nowadays [6] and this could be used for fastening substantially BMA computation.

Once motion has been estimated by BMA, we filter noise (by means of a Vector Median Filter), detect vehicles by grouping together blocks having similar motion vectors (labelling algorithm) and then track vehicles by correlating BMA and labelling data.

The tracking step is based on the following idea. The BMA yields block displacements between two frames. Since the blocks are image subparts, we consider each non-null displacement vector of a frame as the tracking information concerning the block; that is, after the BMA we have already tracked individually each image sub-part. The labelling algorithm provides the information needed to track the objects (which are larger sub-images made out of blocks having similar motion characteristics). More precisely, since the labelling

step yields lists of blocks belonging to the same object, we known which object each block belongs-to, and, based on BMA output, where each block is coming from. By merging the BMA and the labelling outputs we are able to track objects.

As for other tracking approaches, some forecasts the position of the vehicle for the future frame and then assigns it a "link" with an object in the previous one, based on centroid nearness [4] or object texture similarity [2]. Instead, we look back at the position where this object is coming from, because, thanks to multiple consistent blocks tracking, we have reasonable confidence to find it in a well defined area of the previous frame.

Since vehicle's position, displacement, speed and sizes provided by BMA are the only "features" used in our tracking algorithm we consider it as "fully" BMA-based.

2 The Block Matching Algorithm

BMA consists in partitioning each frame of a given sequence into square blocks of a fixed size (in pixel: 6x6 or 8x8 or...) and detecting blocks displacement between the actual frame and the previous one, searching inside a given scan area. It provides a field of displacement vectors (DVF) associated with the frame. Each block encloses a part of the image and is a matrix containing the grey tones of that image part. What we have to do is estimating each block position in the previous frame in order to calculate displacements and speeds (knowing Δt between frames). Each block defines in the previous frame, a "scan area", centered in the block center.

The block is shifted pixel-by-pixel inside the scan area, calculating a match measure at each shift position. The comparison is aimed at determining the pixel set most similar to the block between its possible positions in the scan area. Between these positions, the scan area subpart defined by its center will be the matrix with the best match measure. The similarity between blocks can be evaluated on the basis of several matching measures [11], among which we have adopted the Normalised Cross-Correlation Function:

$$NCCF = \frac{\sum_{i,j}[P(i,j)\cdot q(i,j)]}{\sqrt{\left[\sum_{i,j}P^2(i,j)\right]\cdot\left[\sum_{i,j}q^2(i,j)\right]}} \qquad (1)$$

where "$\cdot$" represents the product between the corresponding elements of matrixes "*q*" and "*P*", and indexes *i,j* rolls all over the matrix size.

The size of the scan area should account for the highest possible block displacement between two frames. However, since computation time largely depends on this size, it is convenient to keep it as small as possible. The matching process yields a displacement vector $\boldsymbol{V}$ between the position of the block in the actual frame and that of the best matching in the previous. In the actual frame, the reflected vector $\boldsymbol{V'=-V}$, applied in the block center, represents the block displacement as well as its tracking information.

The DVF matrix contains a vector per each location. DVF has (*image_rows/N*)-rows and (*image_columns/N*)-columns, were *N* is the number of pixel in each block dimension and *image_rows* and *image_columns* are the size of the original frame. Therefore, the DVF matrix size is smaller than the original image.

Unfortunately, due to noisy grey tone fluctuations, BMA generates many wrong vectors over static blocks located on the background of the road lane. This is typically handled by biasing the selection of the best matching displacement, towards the null displacement [8]; in particular, a fixed value is subtracted to the mismatch measure calculated for the same position in the previous frame (i.e. the null displacement case). Hence, for those blocks having not a reasonably good matching position within the scan area, the displacement vector will tend to be null.

Yet, with this approach the matching process is still carried out over the whole scan area even though the displacement vector turns out to be null. Instead, we have addressed the problem so as to achieve also a significant computational saving. In fact, we calculate first the correlation of a block with the block in the same position of the previous frame (i.e. null displacement) and compare the result with a fixed threshold: if the correlation is higher than the threshold, then the null displacement vector is chosen for the block; otherwise we proceed by calculating the correlation over the scan area and then searching for its maximum value. By using a relatively low threshold we favour null vectors in low textured static areas; moreover we compute just a single correlation step when the vector turns out to be null. We found that this approach is effective in preventing wrong matches within static areas, yielding at the same time a computational saving for the Block Matching Algorithm which is usually of the order of 90%.

As mentioned in the introduction, BMA is very effective in discriminating between close vehicles.

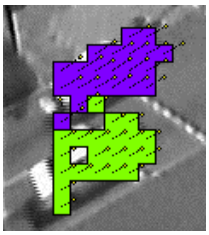

Fig.1 - Two vehicles very close each-other correctly segmented.

This is shown in Fig.1, when two vehicles runs very close. Even though their labels are 8-connected, thanks to the different DVF, it is possible to distinguish between the two vehicles since they move in a different way: the vehicle above is riding faster than the bottom one. The result are two different labels which correctly represents the real situation.

3 Cleaning the DVF

The BMA usually produces noisy DVFs due to the unavoidable grey tone fluctuations of moving objects between successive frames as well as to excessive vehicle speed. Noisy vectors may cause significant over-segmentation of vehicles (i.e. splitting of a single vehicle into different objects) during the labelling process (which will be described in Section 4).

Since the noise found in the DVF may be modelled as the presence of outliers into a rather regular vector field, we have chosen to rely on vector median filtering to smooth DVFs. This approach has been employed for regularisation of velocity [1] and optical flow fields [3] as well as DVFs [4].

The median value of a set of n scalars is defined as $(n/2)$th element of the ordered set. Evidently, this definition does not embody any straightforward extension to vector-valued signals. However, Astola et. al. have introduced a "median" operator for vector valued signals derived from a bi-exponential two-dimensional probability density function using the maximum likelihood approach [1]. This vector operator has properties very similar to those of the standard median for one-dimensional signals [1]. The vector median operator is obtained by taking the element of a set of vectors which minimises the sum of the distances from all the other elements, with distances evaluated on the basis of the L_2-norm: in the $\Re^2$ space, this distance between two vectors, $u(x_u,y_u)$ and $v(x_v,y_v)$, is given by:

$$\left\|\vec{u}-\vec{v}\right\|_{L_2} = \sqrt{(x_u - x_v)^2 + (y_u - y_v)^2} \qquad (2)$$

In our application of the vector median operator, each displacement vector is replaced by the vector median of the set given by the vector itself and its eight nearest neighbour's displacement vectors. In the distance calculation we consider only non null vectors, that is each non-null vector is turned into the vector median computed on the basis of its non-null 8-neighbours. This is done to avoid the changing of non-null vectors into null-ones when the neighbourhood is dominated by null vectors. Fig.2 shows an example of the regularisation action provided by the filter.

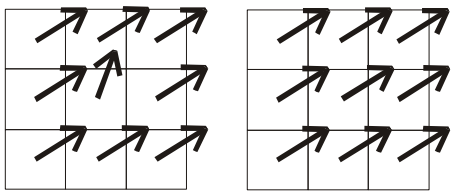

Fig.2 - Vector median based on an 8-neighbourhood

Regularisation by vector median filtering significantly reduce the noise and produces sets of blocks having more similar displacement vectors

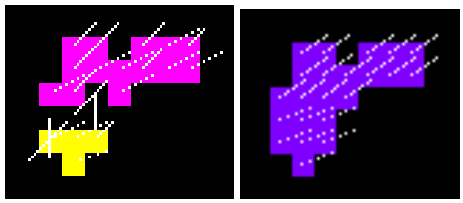

Fig.3 - noisy DVF input (Left), and final VMF output (Right)

Fig.3-Left shows a wrongly recovered vehicle. In Fig.3-Right the DVF has been regularised and the vehicle is correctly detected as a single moving object.

4 The labelling algorithm

After obtaining the restored DVF, we detect vehicles by grouping together 8-connected clusters of blocks having similar (and non-zero) displacement vectors. Fig.4 shows the eight 8-connected neighbours of a given block ***B***.

1	2	3
4	**B**	5
6	7	8

Fig.4 - Block ***B*** and its 8-connected neighbours

The grouping process is basically a sequential, iterative connected components labelling algorithm [7] in which the blocks having non-null displacement play the role of "foreground points" and two blocks are considered neighbours if they are 8-connected and the L_2-norm of their displacement vectors is less than a threshold, *Th*:

$$\left\|\vec{u}-\vec{v}\right\|_{L_2}=\sqrt{(x_u-x_v)^2+(y_u-y_v)^2}<Th \qquad (3)$$

This process is sometimes referred to as δ-labelling. Moreover, we have chosen to filter out the groups made out of very few blocks in order to increase robustness with respect to noise. The value of the threshold on the group size depends on the size of the block with respect to the size of vehicles.

The labelling algorithm is composed of a *main* loop and a subroutine called *Mark-8*. The labelling is run on every frame and produces a matrix, called *label_map*, having the same size as the DVF matrix. As described in the pseudo-code reported below, the *main* loop scans the *label_map* in order to find the first unlabelled block (the loop ends when all the blocks have been labelled). For each unlabelled block found by the main loop, referred to as a *seed*, the label counter, *new_label*, is incremented and its value assigned to the block in the *label_map*. Then the main invokes *Mark-8* which annexes to the *seed* all its 8-connected neighbours having non null DVF value and satisfying equation (3). The annexation is done by coping *new_label* on these blocks.

PSEUDO CODE

```
Per each frame of the sequence:
BEGIN MAIN
{
  FOR i,j scan the whole DVF's block
  matrix:
  {
    IF the block(i,j) is yet unlabelled AND
    it has a non null vector THEN
    REM (i,j) is a seed for the new label

      new_label = new_label + 1;

      label_map(i,j) = new_label;

      CALL "Mark-8"(i,j,new_label);

  }END FOR i,j scan

}END MAIN


****************************************
PROCEDURE "Mark-8"(i,j,new_label)
BEGIN PROCEDURE
{
  WHILE scanning on (r,c), there are
  8-connected to (i,j) blocks
  DO BEGIN{

    IF (r,c) is an un-labelled block
    AND has a non-null associated vector
    AND L2_norm[vector(i,j)-vector(r,c)]
      LOWER THAN Threshold
    THEN
      label_map(r,c) = new_label;

  }END WHILE;
}END PROCEDURE
RETURN;
```

After obtaining the *label_map*, another matrix, called *resized_map*, is produced. It is obtained enlarging the *label_map* to the size of the original frame (see the Sect. 2) so as to have a matrix with labelled objects at the same resolution as the image. This matrix will be used in the tracking algorithm, which is described in the following Section.

6 The tracking algorithm

The tracking algorithm relies on the actual-frame block matching and labelling outputs. The labelling output consists indeed in a set of "temporary" labels which will be updated by the tracking step according to the block-level tracking information provided by the BMA and embedded into the DVF.

Given the set of the blocks belonging to a labelled object in the actual frame, we translate every block in the previous-frame's *resized_map* by a vector equal to its reflected DVF entry (***V'=-V***) and evaluate the amount of overlap of the "temporary" label assigned to the object with the labels in the previous frame.

The process is described in Fig.5 where the right part shows the temporary labelling of an object in the actual frame and the left one the "established" labeling of objects in the previous frame. To track object "*3*" we reflect the DVF and we shift back the blocks by the reflected vectors over the previous frame's *resized_map*.

Each shifted block will overlap *NxN* pixels of the previous *resized_map* pixels: these pixels represents the label entries for the considered block which will be used to determine the final global label value. As shown in Fig.5-left the block may overlap labelled pixels (grey-hatched) as well as unlabelled ones (white)

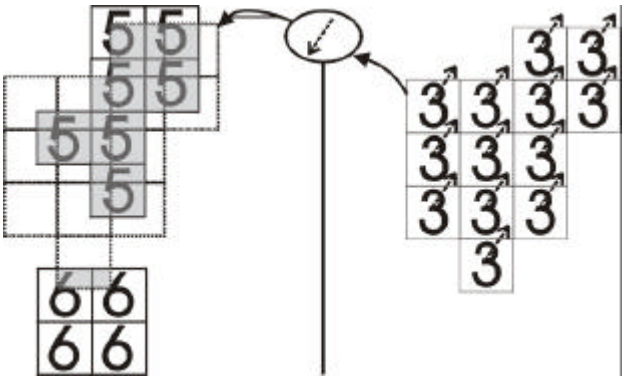

Fig.5 - Tracking an object on the basis of its component blocks

Scanning all the pixels of the shifted label "*3*" we calculate the number of entries for each different pixel label. After repeating these operations for each block we finally obtain an array of label values, where each cell contains the number of the entries for that label value. The number of the cell containing the maximum number of entries will be the new label for all the blocks of the actual object. With reference to Fig.5-left, it is evident that the maximum number of entries will be found at the location number "5" of the mentioned array. Consequently, Fig.6 shows the result of tracking the temporary label "*3*", which has been turned into label "5".

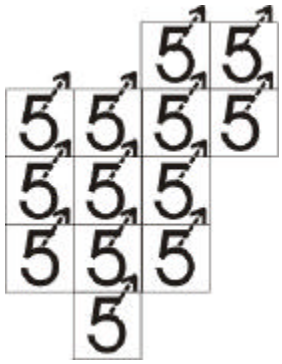

Fig.6 - Result of tracking label "*3*" of Fig.5.

The process is repeated for each temporary label of the actual frame in order to track all the object in the frame. Fig.7 shows the tracking of two vehicles over two contiguous shots: time = *t* on the left and time = *t+1* on the right. The vehicles in the right image hold the same labels as the corresponding vehicles in the left image.

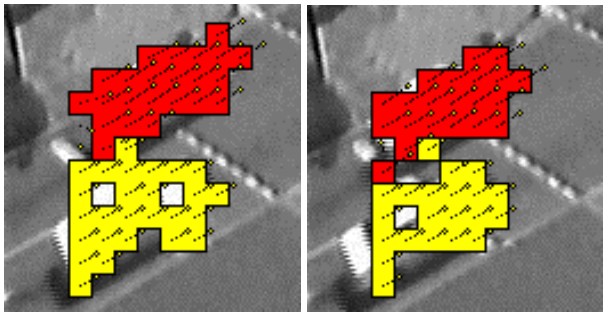

Fig.7 - Tracking output in two successive frames of a traffic sequence

If for a certain temporary label we find no overlap with the *resized_map* of the previous frame, then a new unused label value will be given to this label (it is a "new born").

Moving objects evolves in the scene. Frame by frame we can recover their shape at the resolution of BMA blocks grid. Objects motion, together with low-resolution shape recovery, noise and consequent filtering operations, may causes substantial changes in vehicles shape and size.

For these reasons it may happens that the maximum number of overlapped pixels corresponds to the "unlabelled" label. The problem arises especially in case of objects made out of a few blocks. If we simply assigned the object the label corresponding to the maximum number of overlapped pixels we might label as a "new born" an object which conversely could be tracked. Hence, in this situation we look for the presence labelled pixels and choose as the new label for our object the label value with the maximum number of pixel entries.

7 Conclusion

Our initial experimental results shows that the developed procedure (i.e. BMA with the correlation threshold, adaptive filtering, grouping and tracking) is effective in detecting and tracking vehicles.

The worst problem with BMA seems to be its computational load. For this reason, we plan to develop a new BMA approach in which "interest blocks" are pre-selected on the basis of a rough motion estimate provided by frame differencing. We are also working on improvements of the tracking step based on exploiting the feedback coming from several high-level rules.

As for vehicle detection, there can be still some over-segmentation effect when vehicles are too big with respect to block size. Examples are buses or, when images are significantly affected by perspective, vehicles very close to the cameras. To deal with this problem we are implementing a

perspective correction algorithm in order to estimate motion correctly in case of far and near vehicles.

In addition, our future work will address vehicle detection and tracking in night images.

References:

[1] J. Astola, P. Haavisto, Y. Neuvo, Vector Median Filters, Proceedings of IEEE, vol. 78, No. 4, April 1990.

[2] M. Barattin, R. Cucchiara, M. Piccardi, A Rule-based Vehicular Traffic Tracking System, CVPRIP '98, Research Triangle Park, North Carolina.

[3] F. Bartolini, V. Cappellini, C. Colombo, A. Mecocci, Enhancement of Local Optic Flow Techniques, Proceedings 4th International Workshop on Time Varying Image Processing and Moving Object Recognition, June 10-11, Florence, Italy, 1993, pp. 359-366.

[4] F. Bartolini, V. Cappellini, C. Giani, Motion Estimation and Tracking for Urban Traffic Monitoring, Proceedings 3rd IEEE International Conference on Image Processing ICIP'96,Lausanne, Switzerland, September 16-19 1996, pp. 787-790.

[5] P. Baglietto, M. Maresca, M. Migliardi, N. Zingirian, Customizing MPEG Video Compression Algorithm to Specific Application Domains: the Case of Highway Monitoring, Lectures Notes in Computer Science, Vol. 1311, Springer Verlag, 1997, pp. 46-53.

[6] De Vos and M. Stegherr, Parametrizable VLSI Architectures For Full-search Block-matching Algorithms, IEEE Transactions on Circuits and Systems, vol. 36, October 1989, pp. 1309-1316.

[7] R. M. Haralick, L. G. Shapiro, *Computer and Robot Vision*, Addison-Wesley, Vol.1, 1992, pp.28-33.

[8] ITU-T, sl Video Codec Test Model, TMN5, ITU-TSS LBC-95, Study Group 15, Working Party 15/1, Expert's Group on Very Low Bitrate Visual Telephony, Telenor Research, January, http://web.fou.telenor.no/fou/DVC/tmn5/tmn5.html, 1995.

[9] D. Koller, J. Weber, T Huang, J. Malik, G. Ogasawara, B. Rao, S. Russel, Towards Robust Automatic Traffic Scene Analysis in Real-Time, Proc. Int'l Conf. Pattern Recognition, 1994, pp. 126-131.

[10] P. G. Michalopulos, Vehicle Detection Video Through Image Processing: The Autoscope System, IEEE Transactions on vehicular technology, Vol. 40, No. 1, February 1991, pp. 21-29.

[11] H. G. Musmann, P. Pirsch, H. J. Grallert, Advances in Picture Coding, Proceedings of the IEEE, Vol. 73, No. 4, April 1994, pp. 523-546.

[12] R. J. Shalkoff, *Digital Image Processing and Computer Vision*, Wiley, 1989, pp.137-144.