ASTES

Advances in Science, Technology and Engineering Systems Journal Vol. 2, No. 3, 321-326 (2017)

www.astesj.com

Special Issue on Recent Advances in Engineering Systems

ASTESJ

ISSN: 2415-6698

Real Time Advanced Clustering System

Giuseppe Spampinato[1*], Arcangelo Ranieri Bruna[1], Salvatore Curti[1], Viviana D'Alto[2]

[1]*STMicroelectronics, Advanced System Technology, Catania, Italy*

[2]*STMicroelectronics, Advanced System Technology, Agrate Brianza, Italy*

ARTICLE INFO

Article history:
Received: 30 March, 2017
Accepted: 24 April, 2017
Online: 16 May, 2017

Keywords:
Automotive
Control traffic alert
Optical flow
Clustering
Moving object detection

ABSTRACT

This paper describes a system to gather information from a stationary camera to identify moving objects. The proposed solution makes only use of motion vectors between adjacent frames, obtained from any algorithm. Starting from them, the system is able to retrieve clusters of moving objects in a scene acquired by an image sensor device. Since all the system is only based on optical flow, it is really simple and fast, to be easily integrated directly in low cost cameras. The experimental results show fast and robust performance of our method. The ANSI-C code has been tested on the ARM Cortex A15 CPU @2.32GHz, obtaining an impressive fps, about 3000 fps, excluding optical flow computation and I/O. Moreover, the system has been tested for different applications, cross traffic alert and video surveillance, in different conditions, indoor and outdoor, and with different lenses.

1. Introduction

Clustering is the task of grouping a set of information in such a way that information in the same group (called a cluster) are more similar to each other than to those in other groups. A clustering system, in particular regarding detection and tracking of moving objects, is a really common and important task, widely used in computer vision as low-level task for different applications like: camera surveillance [1], Cross Traffic Alert (CTA) [2], Intelligent Transport System (ITS) [3], object detection and recognition [4], people detection and recognition [5], people behavior analysis (e. g. anti-theft, falling detection, etc.) [6] and user interfaces by gesture [7]. Efficiency and robustness of clustering, considered as low-level task, is really important because it influences the performance of all higher level tasks of the aforementioned applications.

To implement the clustering task, there are in general two types of approaches: region based and boundary based. For what concerns region based, different approaches have been proposed: combined color based and region based particle filters, using multiple hypotheses for tracking objects [8], motion-based segmentation and a region-based Mean-shift tracking approach, fused by a Kalman filter, to extract object independent motion trajectory under uncontrolled environment [9], and so on. However the most popular and recent region based approaches is background subtraction [10-14]. Once the background is removed from the scene, through different kind of filters applied on the foreground, it is possible to identify moving object. Since these methods are pixel-based, they are not suitable for real time applications, also because they require a long time for estimating the background models.

In boundary based approaches, different features have been proposed: color feature descriptors, based on the spectral power distribution of the illuminant and object's surface reflectance property [15], multi-feature descriptors [16], and so on. However, many recent approaches use edge based optical flow [17-20]. Optical flow estimation gives a two-dimensional motion vector, which represents the movement of some point of an image in the following one of the sequence. Basic idea in boundary based approaches is to consider vectors with similar spatial and motion information belong to the same object [21]. Since these approaches are feature-based, they are a good choice for all real time applications, even if usually the do not reach high performances.

In this work, we propose a very low-cost clustering system, working with a generic optical flow and reaching high performances in different application scenarios. This paper is organized as follows: Section II describes the proposed system, block by block, also showing visual impact on a CTA example; Section III shows the experimental results for different application

*Corresponding Author: Giuseppe Spampinato, STMicroelectronics, Advanced System Technology, Catania, Italy | Email: giuseppe.spampinato@st.com

cases; finally, conclusion and future work are exposed in Section IV.

2. Proposed System

Since we are interested in light systems, we started our investigation with optical flow based algorithm, which system [22,23] is described in Fig. 1. The standard system takes as input the optical flow, that is the list of motion vectors between adjacent frames. First step is usually a pre-filtering to eliminate noise and/or spikes. Afterwards, labelling step inserts same label for each "similar" vector. The role of clustering step is to group vectors with the same label into clusters. At the end, "similar" clusters are merged together to obtain the final list of identified clusters. In this system, some steps (in particular pre-filtering) can be computationally heavy and no previous vector information is used. Moreover, no temporal cluster information is used to enhance clustering, but the previous clustering information is usually used only for tracking.

To overcome the aforementioned drawbacks of typical optical flow based system, we propose a novel low-cost system only based on optical flow, described in Fig. 2. The new blocks have been added to increase performances of the system. Moreover, previous frame is not necessary, but just a list of previous vectors and clusters are needed for processing. Detailed description of each block is explained in the next sub-paragraphs. Moreover, to better understand the peculiarity of various blocks, example images are shown.

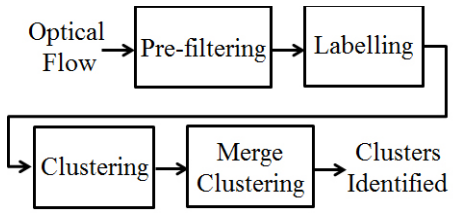

Fig. 1. Typical optical flow based system.

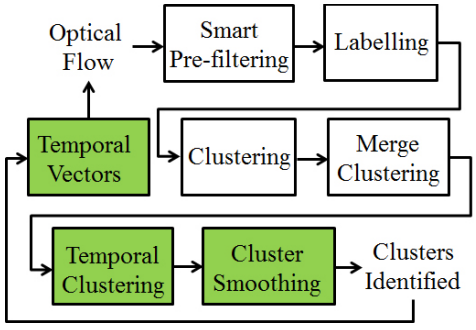

Fig. 2. Proposed system

2.1. Smart Pre-filtering

This starting block takes as input the optical flow, the set of motion vectors generated in any manner, and its role is to remove from this set all vectors which can be considered as noise. Usually complex filtering is used as for example median filters [22] and so on. We propose a new simple filtering with two different options, applied depending on the quality of incoming optical flow.

First option, called *brute-force*, just eliminates motion vectors with small movements, which are considered as noise. This option is indicated just for low quality optical flow. In particular, a motion vector $v=(dx,dy)$ is removed from optical flow if the following condition is satisfied:

$$(ABS(dx) \leq MAX\_DX) \text{ and } (ABS(dy) \leq MAX\_DY) \quad (1)$$

Where MAX_DX and MAX_DY are thresholds chosen depending on the precision of the optical flow and ABS is the absolute value.

Second option, called *smart*, apart applying (1), do not remove in the output optical flow, the motion vector v with the following characteristics: it lies in the area around a cluster C of previous frame and its orientation is similar to the orientation of C, that is:

$$ABS((orientation(v) - orientation(C)) \leq MAX\_ORIENT1) \quad (2)$$

Where *MAX_ORIENT1* threshold should be opportunely chosen, depending on specific application. In particular, for rigid objects, as vehicles, a small value of *MAX_ORIENT1* is a good choice, while for not rigid objects, such as people, we need to increase its value.

About *orientation*, we have to distinguish between orientation of a vector $v=(dx,dy)$ and orientation of a cluster C. In the first case, the formula is the following:

$$orientation(v) = tan^{-1}(dy/dx) \quad (3)$$

Instead, orientation of a cluster C, it is defined as the mean of the orientation of all motion vectors included in C. To execute the tan^{-1} function at real-time a fixed point arctangent algorithm was implemented (http://www.coranac.com/documents/arctangent/).

Figure 3 shows an input optical flow example. In this example we have three cars moving: the middle one to the left, the other two to the right. We can easily notice in the scene some noisy vectors in particular around the leafs of the surrounding trees, which is a typical problem in these kind of applications.

Fig. 3. CTA example: input optical flow

2.2. Labelling

This block takes as input the filtered optical flow. Its role is to insert a label for each motion vector, assuring to insert the same label for motion vectors witch potentially belong to the same cluster. Two motion vectors $v_1=(dx_1,dy_1)$, spatially situated in point $p_1=(x_1,y_1)$, and $v_2=(dx_2,dy_2)$, spatially situated in point $p_2=(x_2,y_2)$, have the same label if the following three conditions are satisfied:

$$(ABS(x1-x2) + ABS(y1-y2)) \leq MAX\_DIST\_POINT \quad (4)$$

$$(ABS(dx1-dx2) + ABS(dy1-dy2)) \leq MAX\_MOD \quad (5)$$

$$ABS((orientation(v1) - orientation(v2)) \leq MAX\_ORIENT2) \quad (6)$$

Where the thresholds should be opportunely chosen: *MAX_DIST_POINT*, depending on specific application and typical objects distance; *MAX_MOD* depending on precision of optical flow; *MAX_ORIENT2* depending on specific application.

The above formulas indicate that two motion vectors are considered to belong to the same cluster if they are spatially near (4), have got similar module (5) and similar orientation (6). While (4) and (5) are commonly used for labelling [23], (6) has been added for robustness of the system.

2.3. Clustering

This block takes as input the filtered optical flow, obtained by *smart pre-filtering* block and the related labels, coming from *labelling* block elaboration. The aim of this step is to group vectors with the same label into clusters. To avoid having clusters with dimensions too small, it is important to extend the cluster found to a minimum dimension N×*M*. This dimension depends on various factors: typical object size we want to cluster, typical distance of the object from the camera and the type of the camera (liner or fish-eye).

Figure 4 shows in boxes the nine clusters, obtained from *clustering* step in the CTA example of Fig. 3. From the center of each cluster, its *orientation* is displayed to better understand the next *merge clustering* step, which will reduce the number of clusters.

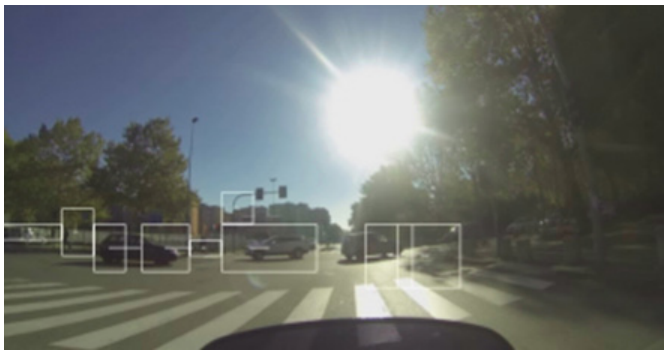

Fig. 4. CTA example: clustering step output

2.4. Merge Clustering

This block takes as input a list of clusters, coming from previous *clustering* block, and merges together similar clusters to obtain a new list of identified clusters. Two clusters C_1 and C_2 are considered similar and then merged if the following conditions are satisfied:

$$distance\_X(C_1,C_2) \leq MAX\_DIST\_CLUSTER\_X \quad (7)$$

$$distance\_Y(C_1,C_2) \leq MAX\_DIST\_CLUSTER\_Y \quad (8)$$

$$ABS((orientation(C_1) - orientation(C_2)) \leq MAX\_ORIENT3) \quad (9)$$

Where the thresholds should be opportunely chosen: *MAX_DIST_CLUSTER_X* and *MAX_DIST_CLUSTER_Y*, depending on application and typical objects distance; *MAX_ORIENT3*, depending on application.

In (7) and (8), the distance for clusters concept is analogous to distance for points concept in (4) in *labelling* step; while (9) is analogous to (6). The above formula indicates that two clusters are similar if they are spatially near (7-8) and have got similar orientation (9). Since for each cluster we have its orientation, it is easy to extend the similarity also to the module, in analogous way of (5).

Looking at Fig. 4, from left to right, the first car is spit in two cluster and these clusters have got the similar orientation, so the *merge clustering* step will join these two clusters. The middle car is well detected, but since there are near clusters with similar orientation, these clusters will be joined together. In the last car, we can notice two near clusters, but since they have opposite orientation, they will be not joined together. Figure 5 shows, marked by ellipses, similar clusters that *merge clustering* step will join, while Fig. 6 shows the merged clusters, now reduced to six.

Fig. 5. CTA example: similar clusters to merge

Fig. 6. CTA example: merge clustering step output

2.5. Temporal Clustering

This block takes as input two lists of clustering: current frame list and previous frame list. The roles of this block are two: the first one is similar to the *merge clustering* block, that is to merge similar clusters in current and previous frame; while the second role is to eliminate clusters in current frame which do not have similar clusters in the previous frame. Similarly to the method described in Sub-section D, two clusters are similar if condition (7), (8) and (9) are satisfied. The difference, compared to *merge clustering* block, are the following: for *temporal clustering* block C_1 and C_2 are two clusters coming from two adjacent frames; *MAX_DIST_CLUSTER_X* and *MAX_DIST_CLUSTER_Y* thresholds can be different in value. Of course, as said before, in this block we can use the module as similarity condition. Another important note is that, to increase reliability of the system, assuming constant speed of objects to identify, the clusters of previous frame can be motion compensated.

In the CTA example, looking Fig. 6 from left to right, the first two and the last clusters will be removed, since they do not have similar clusters in previous frame. Figure 7 shows the resulting clusters, reduced to three, as the real moving objects in the scene.

Fig. 7. CTA example: temporal clustering step output

2.6. Cluster Smoothing

This block takes as input the list of clusters, coming from the previous *temporal clustering* block, and the previous frame clustering list. Clusters marked as similar, in the temporal clustering block, to avoid big discrepancy in dimensions, should be tuned according to the similar clusters in the previous frame. In particular, a weighted average between previous C_1 and current C_2 similar clusters should be calculated as follows:

$$C_2 = (C_2 * CURRENT\_W + C_1 * PREVIOUS\_W)/SUM\_W \quad (10)$$

With:

$$SUM\_W = CURRENT\_W + PREVIOUS\_W \quad (11)$$

Fig. 8. CTA example: cluster smoothing output

The weights *CURRENT_W* and *PREVIOUS_W* should be opportunely chosen, usually giving more weight on the previous cluster. As said for *temporal clustering* step, to increase reliability of the system, the clusters of previous frame can be motion compensated.

The final effect of this block is shown in Fig. 8, where we can note that dimensions of the three clusters are more similar to the real dimensions of objects. This is the final output of the proposed schema.

2.7. Temporal Vectors

It is important, in particular for not dense optical flow, at the end of the proposed system, to insert in the optical flow of the next frame all vectors contained in the current cluster, to limit clusters disappearing in the scene. Of course, these inserted vectors can be motion compensated, assuming constant speed.

2.8. Tracking

Since in *temporal clustering* step, we obtain similarity and then relationship between previous and current frame clusters, it is easy to calculate the trajectory of the identified object in the scene. The cluster centroid can be obtained and, according to relationship between clusters, displayed in the scene.

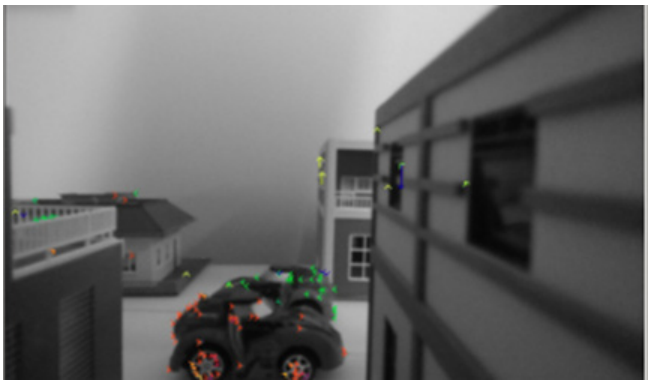

Fig. 9. Simulated example: input optical flow.

Fig. 10. Simulated example: proposed system output.

3. Experimental Results

Lots of tests have been executed with different scenarios, as CTA and video surveillance, and different cameras at different resolutions, with both linear and fish-eye lenses, obtaining really good visual results.

Speed testing has been made calculating computation time for VGA sequences on a 2.3GHz processor (ARM Cortex-A15, using a single core). About 3000 frame/sec were obtained, executing only the described ANSI-C algorithm (excluding the optical flow and I/O calculation).

Considering that the memory required by the system is really low, about 96K considering 4K optical flow for frame, and considering the aforementioned speed test, we can assure that the system is suitable also for really low-cost devices.

An example of simulated CTA with partially occluded objects is shown in Fig. 9 and Fig. 10. In particular the proposed system, with the input motion vectors in Fig. 9, correctly identifies and well spots the two partially occluded crossing cars, as shown in Fig. 10.

An example of real CTA with fish-eye lens, with multi objects moving, is shown in Fig. 11 and Fig. 12. In particular the proposed system, with the input motion vectors in Fig. 11, correctly identifies the three crossing cars shown in Fig. 12.

Fig. 11. Fish-eye example: input optical flow.

Fig. 12. Fish-eye example: proposed system output.

An example of video surveillance is shown in Fig. 13 and Fig. 14. In particular, the proposed system, with the input motion vectors in Fig. 13, correctly identifies the person in Figure 14. It is to underline that the proposed system not only correctly identifies the man leaving the bag, but also bag itself.

The last case can be useful also to recognize the unattended luggage application, really important for example for airport security.

Fig. 13. Video surveillance example: input optical flow.

Fig. 14. Video surveillance example: proposed system output.

4. Conclusion and future work

The proposed approach has been experimentally tested on a representative dataset of scenes obtaining effective results in terms of accuracy. A very reliable and low-cost clustering system has been developed with the characteristic to be very low power consumption, since it works only with optical flow information. The proposed system is also very flexible: can be used with any algorithm which estimates motion vectors between adjacent frames. An important characteristic is the consideration of previous frame clustering and vectors information for a more robust clustering determination in the current frame. Moreover, average of current and previous related cluster, to maintain more graceful clustering output for the user.

A limitation of this work is that we consider just stationary camera, so can be used for some specific applications, like CTA and video surveillance. Another limitation is related to input optical flow: if the image is too dark or blurred, or moving object is too speed, poor optical flow will be produced and the proposed method will fail.

Future work will be orientated on overcoming this limitations, leaving the system reliable also in the case of moving cameras, introducing robust filtering to remove from input optical flow the global motion vector, opportunely calculated.

Conflict of Interest

The authors declare no conflict of interest regarding the publication of this paper.

References

[1] D. Coppi, S. Calderara, R. Cucchiara, "Transductive People Tracking in Unconstrained Surveillance" in IEEE Transactions on Circuits and Systems for Video Technology, 2016.

[2] V. Balisavira, V. K. Pandey, "Real-time Object Detection by Road Plane Segmentation Technique for ADAS" in Eighth International Conference on Signal Image Technology and Internet Based Systems (SITIS), 2012.

[3] N. Seenouvong, U. Watchareeruetai, C. Nuthong, K. Khongsomboon, N. Ohnish, "A computer vision based vehicle detection and counting system" in Eighth International Conference on Knowledge and Smart Technology (KST), 2016.

[4] V. Kalogeiton, V. Ferrari, C. Schmid, "Analysing domain shift factors between videos and images for object detection" in IEEE Transactions on Pattern Analysis and Machine Intelligence, 2016.

[5] A. García-Martín, J. M. Martínez, "People detection in surveillance: classification and evaluation" in IET Computer Vision, 2015.

[6] K. Ozcan, S. Velipasalar, "Wearable Camera- and Accelerometer-Based Fall Detection on Portable Devices" in IEEE Embedded Systems Letters, 2016.

[7] X. Dang, W. Wang, K. Wang, M. Dong, L. Yin, "A user-independent sensor gesture interface for embedded device" in IEEE Sensors, 2011.

[8] S. Kumar, M.S. Narayanan, P. Singhal, J.J. Corso, V. Krovi, "Product of tracking experts for visual tracking of surgical tools" in IEEE International Conference on Automation Science and Engineering (CASE), 2013.

[9] X. Wu, X. Mao, L. Chen, A. Compare, "Combined Motion and Region-Based 3D Tracking in Active Depth Image Sequence" in IEEE International Conference on Cyber, Physical and Social Computing, 2013.

[10] D. K. Panda, S. Meher, "Detection of Moving Objects Using Fuzzy Color Difference Histogram Based Background Subtraction" in IEEE Signal Processing Letters, 2016.

[11] M. Anandhalli, V. P. Baligar, "Improvised approach using background subtraction for vehicle detection" in IEEE International Advance Computing Conference (IACC), 2015.

[12] O. E. Harrouss, D. Moujahid, H. Tairi, "Motion detection based on the combining of the background subtraction and spatial color information" in Intelligent Systems and Computer Vision (ISCV), 2015.

[13] T. Huynh-The, O. Banos, S. Lee, B. H. Kang, E. Kim, T. Le-Tien, "NIC: A Robust Background Extraction Algorithm for Foreground Detection in Dynamic Scenes" in IEEE Transactions on Circuits and Systems for Video Technology, 2016.

[14] S. Battiato, G. M. Farinella, A. Furnari, G. Puglisi, A. Snijders, J. Spiekstra, "An Integrated System for Vehicle Tracking and Classification" in Expert Systems with Applications, Vol. 42, Issue 21, pp. 7263–7275, 2015.

[15] A. Yilmaz, O. Javed, M. Shah, "Object Tracking: A Survey" in ACM Computing Surveys, 2006.

[16] H. Yang, L. Shao, F. Zheng, L. Wangd, Z. Song, "Recent advances and trends in visual tracking: A review" in Elsevier Neurocomputing, 2011.

[17] C. Huang, M. Hung, "Target motion compensation with optical flow clustering during visual tracking" in IEEE 11th International Conference on Networking, Sensing and Control (ICNSC), 2014.

[18] C. Wong, W. C. Siu, S. Barnes, P. Jennings, B. Fong, "Shared-use motion vector algorithm for moving objects detection for automobiles" in IEEE International Conference on Consumer Electronics (ICCE), 2016.

[19] J. Hariyono, K. Jo, "Detection of pedestrian crossing road" in IEEE International Conference on Image Processing (ICIP), 2015.

[20] C. Liang, C. Juang, "Moving Object Classification Using a Combination of Static Appearance Features and Spatial and Temporal Entropy Values of Optical Flows" in IEEE Transactions on Intelligent Transportation Systems, 2015.

[21] W.-C. Lu, Y.-C. F. Wang, C.-S. Chen, "Learning dense optical-flow trajectory patterns for video object extraction" in IEEE International Conference on Advanced Video & Signal-based Surveillance (AVSS), 2010.

[22] S. Aslani, H. Mahdavi-Nasab, "Optical Flow Based Moving Object Detection and Tracking for Traffic Surveillance" in World Academy of Science, Engineering and Technology International Journal of Electrical, Robotics, Electronics and Communications Engineering, 2013.

[23] M. Yokoyama, T. Poggio, "A Contour-Based Moving Object Detection and Tracking" in Proceedings of the 14th International Conference on Computer Communications and Networks (ICCCN), 2005.