Low Complexity Motion Detection with Background Modeling

Alberto Dassatti, Guido Masera, Gianluca Piccinini
VLSI Lab, Electronic Department,
Politecnico di Torino, Torino, Italy
{alberto.dassatti,guido.masera,gianluca.piccinini}@polito.it

Abstract— **Motion Detection is of utmost importance in several applications such as video surveillance, remote sensing and medical diagnosis, and a large number of algorithms have been proposed for implementation. In this paper we focus on low resources scenarios where standard solutions are inapplicable. We present a novel algorithm that exploits spacial correlation of neighbor pixels in an image to get the most accurate view of the moving scene without increasing computational burden. In conjunction with the proposed algorithm we investigate, to the best of our knowledge for the first time, several background models and update policies, comparing available approaches and exploring trade-offs between complexity and detection capability.**

I. Introduction

Motion Detection (MD) deals with identifying part of video sequences that capture moving objects. This kind of information is a crucial ingredient in a growing set of applications, ranging from well known surveillance appliance to sensors networks. Particularly, the availability of low cost wireless video sensors to be spread in a proper area opens up new application possibilities in contexts such as environmental monitoring, domotics, marine biology, habitat studies, etc .... All these applications are characterized by remote sensors with extremely limited processing capabilities and reduced activity of the RF (Radio Frequency) interface. Therefore low complexity and low energy MD algorithms resident in the sensor and capable of triggering the sensor only when necessary are a key enabling technology for this kind of monitoring applications.

The most popular method for motion detection and image segmentation is based on the evaluation of frame differences; basically the absolute difference between a reference image and the frame under test is computed and, if the difference is above a threshold, motion is detected. A lot of variants of this method were proposed in the literature and a complete survey is available in [1]. These methods also need the choice of a reference model and some thresholds; interesting examples are in [2] and [3] where thresholds are chosen dynamically , based on the statistical properties of the video sequence. Another example is [4]; here authors propose a statistical adaptive update scheme for the reference frame, combining different methods that operate at pixel, region and frame levels. All the recently published techniques focus on detection accuracy, not considering complexity as key point. In [5], [6] and [7] under-sampling with a fixed grid is used to remarkably reduce the number of performed operations. In [6] authors modified the algorithm proposed in [8] to achive better resilience to illumination changes; in [6] and [7] homomorphic filtering techniques proposed in [9] are exploited to combat the effects of illumination changes.

A completely different approach is presented in [10], [11], where authors target a low power solution: to this purpose, the image is processed in the analog domain and a single bit plane is used to decide if motion has occurred or not. This approach reduces considerably the overall complexity of the MD, but requires custom hardware and cannot be easily adopted in existing systems.

As proved by the large number of papers published on this topic in the last ten years, one of the most challenging problems in motion detection and segmentation is the background modeling (for a recent and complete survey the interested reader can refer to [1]). Two problems arise when background modeling is considered: i) how to build the model and ii) how often the model has to be updated Both these issues have a deep impact on processing complexity as well as on the overall detection performance, so that possible trade-offs have to be carefully investigated.

In this work we present a reduced complexity algorithm with automatic noise calibration, able to detect motion and trigger alarm, with the computational power of few Mega Instructions per second (MIPS) on a CIF (in the PAL system 352x288 pixel) video sequence.

Our method is described in Section II, while section III presents, to the best of our knowledge for the first time, a study of background modeling techniques, comparing their complexity and evaluating the update policy impact on the overall processing. Finally Section IV summarizes performance and trade-offs for the implementation of efficient low cost MD.

II. Motion Detection Algorithm

Detection of moving objects in image sequences is often based on difference between a reference image and the current frame. Both real motion or noise generate these differences and the purpose of MD is to discriminate them and label those changes that are due to motion. Noise–related changes, on the other hand, have to be neglected.

The proposed MD algorithm is designed with low complexity in mind and it works as follows: at the beginning of the procedure a reference image is stored and successive frames are compared with it.

Some thresholds need to be defined:

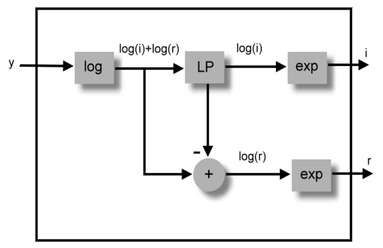
- N, noise threshold
- M, motion threshold
- P, max number of randomly selected pixels
- S, size (in bit) of the minimum object of interest.

The process starts sampling a pixel location; for each sampled pixel, the difference between the current pixel and the reference one is computed. If the difference is below the threshold N, the pixel is classified as noise and neglected, otherwise we select recursively neighbor pixels, not yet tested, and repeat the same process. Once we have tested all adjacent non-noise pixels, if they are more than S, the homomorphic filtering proposed in [12] is applied and the mean difference is evaluated in the logarithm domain, as proposed in [7].

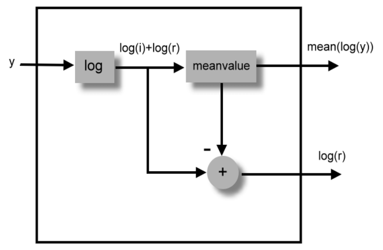
Pixels can be considered as composed by the product of two components: illumination and reflectance. Only changes in the reflectance influence the detection capability, so if we are able to separate the two components we can neglete the illumination changes. Illumination is placed at low spatial frequencies whereas the reflectance is located at higher frequencies, thus a simple filter can separate them and two exponentials can recover them separately (see Fig. 1a). Image 1 describes the standard processing and the simplified computation in the logarithm domanin (Fig. 1b). If this mean is larger than M the block is classified as Motion Block, it is discarded otherwise. As soon as a Motion Block is discovered the process can be terminated. This early termination is useful when this algorithm is used as alarm trigger; in scenarios where identification of several moving objects is of concern this step can be easily omitted.

Comparing only a small subset of pixels per frame is a powerful method to reduce complexity in MD as demonstrated in [6] and [7]. A fixed under-sampling grid can be applied to this purpose and different criteria can be adopted to choose the grid. Once chosen the maximum number (P) of pixels to be tested for each frame, we select randomly and uniformly pixels along the frame. The computational complexity associated to the genration of a random grid is negligible; moreover this method brings two positive consequences: 1) no easy untested paths are present and 2) the minimum size of a detectable moving object is reduced to a single pixel; it is important noticing that the grid proposed in [6] results into a minimum size detectable object of 6×6 pixels in round 1, (decreased to 4×4 and 3×3 in round 2 and 3 respectively). A further advantage of the proposed method comes from the possibility of limiting the required computation around a region of interest, that is the area where motion occurrence is more probable. Image adjustments and optimizations of the moving object view are not performed in this application.

M and N thresholds can be dynamically adjusted by means of a calibration process. In [7] an effective procedure to adapt N was proposed: in the presented method, the algorithm is run for some frames, ensuring that no motion occurs; all the differences are then stored in one vector and a normalized histogram is generated to determine N. The same procedure is adopted in the present work and periodically re-called in order to re-calibrate the system.



(a) Complete form



(b) Simplified form

Fig. 1. The homomorphic filter

The calibration of S is application dependent and it can address either a single pixel or a wide slice of the image under test. The choice of this parameter has almost no impact on the computational complexity, but it is critical from the performance point of view; too small values of S bring to a high number of homomorfic filtering operations to be performed, while large values tend to reduce the probability of discover small moving objects. Whenever the resiliency to light changes is of concern, S has to be large enough to give statistical significance to the mean value computed in the homomorphic filter.

There is no limitation in the choice of parameter P; it is obvious that a small P makes the algorithm extremely fast, but it increases the probability of missing detection in the frame under test. On the other hand, large values of P reduce the achievable speed-up while increasing the probability of detection, defined in this case as the probability that a single pixel within a moving object is tested and its difference with the reference frame is larger than N.

The random sampling process is not reset after a frame processing: this ensures that all the spatial location will be tested within few frames with high and uniform probability. This consideration links the value of P to the system frame rate. Intuitively similar detection capabilities can be achieved either with high frame rates and low P values or with low frame rates and high values of P. This makes the proposed method extremely flexible and adaptable to different system scenarios.

If the system is used for alarm generation, high values of P are suggested coupled width high values of S and low frame rates. This combination ensures low probabilities of miss-detection and low number of false alarms, providing at the same time limited complexity.

III. Background modelling

The algorithm described in II requires a reference frame in order to compute temporal differences. Unfortunately, to the best of our knowledge, no previous studies investigated the complexity of background modeling and all the low complexity MD algorithms presented in section I do not make use of any background technique.

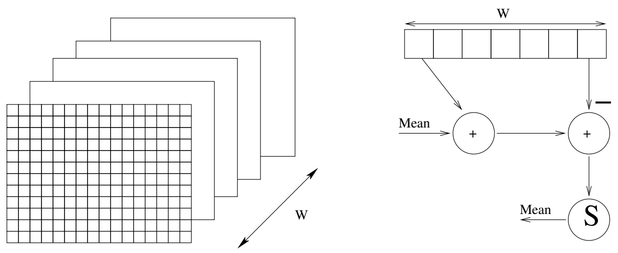


Fig. 2. Frame Window processing

In this Section, we specifically address the background modeling issue, evaluating in particular low cost methods, with limited requirements of both computational and storage requirements.

From [4] and [1] we selected the following seven methods:
Trivial (T): a single frame is used as background.
Temporal Mean (TM): first order statistic is collected for each pixel computing the mean of the values assumed by the pixel in the same position of successive frames.
Temporal Mean and Variance (TMV): second order statistic is added to the previous method computing temporal variance for each pixel.
Spatial Mean (SM): the image is divided in blocks of size B; each pixel in the block is represented by the mean values of intensity within the block.
Spatial Mean and Variance (SMV): the second order spatial statistic is computed for each block.
Space-Time Mean and Variance (STMV): this method combine the SMV and TMV methods.
Temporal Derivative (TD): maximum and minimum values in intensity are stored as well as maximum inter-frame distance between two successive frames. Then the MD algorithm classifies the pixel as motion if it deviates from maximum and minimum more then maximum inter-frame distance.

The algorithm described in Section II has to be slightly modified when a second order background model is used: the pixel is classified as motion if the difference between the pixel and the mean is larger than the variance plus M.

Complexity of this model can be compared in terms of memory accesses, memory storage and number of operations performed. Table III provides normalized values for these metrics. In Table III, W is the size of a sliding window (we use power of two values for W in order to avoid divisions) used to compute moving average and the pixel variance as illustrated in Figure III. The same is true for B, where $B \times B$ represents the dimension, measured in pixels, of a block in which spatial statistics are collected. All values are normalized with respect to the image size and obtained manually analyzing our unoptimized C implementation. It has to be stressed that the whole processing has to be repeated accordingly to the update policy adopted.

	Memory access	**Memory Storage**	**Basic Operation**
T	1	1	0
TM	2	$1+W$	$2A+S$
TMV	$1+W$	$2+W$	$(W+1)A+(W+1)S$ $+WM$
SM	1	$1/B$	$A+BS$
SMV	2	$2/B$	$A+2BS+1M$
STMV	$2+W$	$2/B+2+W$	$(W+2)A+(W+1)M$ $(W+1+2B)S$
TD	3	3	$3A$

TABLE I
COMPLEXITY OF BACKGROUND MODELING TECHNIQUES IN TERMS OF MEMORY REQUIREMENTS AND OPERATIONS PERFORMED (A FOR SUM OR SUBTRACTION, S FOR SHIFT AND M FOR MULTIPLICATIONS). THIS VALUES ARE NORMALIZED BY THE IMAGE SIZE.

Background modeling is often undervalued in motion detection problems and when limiting complexity is one of the main goals it becomes a key processing component. For example, if we consider the TM model applied to a CIF movie sequence with a frame rate of 30 fps, even this simple method requires more than 9 $MIPS$.

Comparing models in Table III in terms of detection performance is an hard task. Figure 3 depicts a visual example of performance obtained with the MD algorithm described in Section II and using different background models. Figure 3(a) reports original frames from three different movie sequences while Figures 3(b) .. 3(h) are the output of the MD algorithm on the same reference frames presented in 3(a) with the seven background models. Figure 3 reveals that temporal correlation (sub-figures c,d,g) performs better than spatial one (sub-figures e,f): for example if we compare images 3(c) and 3(e) we can see how SM is unable to detect some moving objects (first and second sequences) and add a lot of noise generating false alarms. Due to the difficulties of comparing visually the performance of these techniques, we also compare them in terms of false positive and false negative detected motion pixels. For each background model, we counted the number of pixels wrongly labeled as motion, and motion pixels not recognized. In order to compute these metrics we need a reference segmentation of each frame. Instead of the manual segmentation adopted in [4], we employed the segmentation made by the more sophisticated method (STMV) as reference.

All the experiments are executed with P equal to half frame size. Figure III reports numbers of false positive and false negative pixels detected in 100 frames, chosen from 5 different video sources.

Numeric results, presented in Figure 3, confirm the visual impression, but they add one new element of information: methods based on spacial analysis are mainly affected by false detection.

A. Update Policy

In order to evaluate the complexity of the complete detection systems we have to include the background model updating. Therefore, in conjunction with the aforementioned background schemes, we studied four update policies: α) update every frame, β) update after a fixed timeout, γ) update every time motion is detected and δ) random partial update. The former two policies are proposed here for the first time in order to mitigate the complexity of the background modeling; in policy δ) the update of the background model for a specific spatial location occurs every time the MD algorithm visits that pixel. This policy reduces by a factor P the complexity reported in Table III regardless the used method.

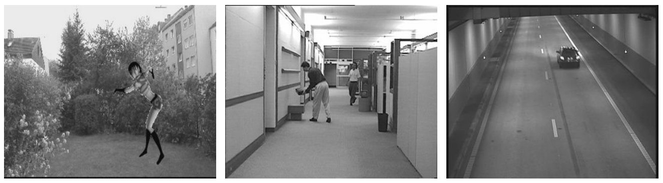
As done for the comparison among background modeling methods, we evaluate the effect of the update policy on the system performance in terms of false detections. Figure III-A shows performance of the considered methods combined with the four update policies. From the image analysis it is clear that there is no significant impact of the policy over the detection capability; on the other hand Table III-A highlights a remarkable variation in terms of complexity.

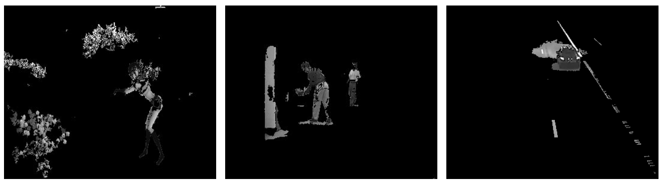
	α	β	γ	$\delta(16)$	$\delta(64)$	$\delta(128)$
T	1052.17	752.36	906.94	1033.99	1714.47	2350.42
TM	608.48	903.50	897.28	784.17	1231.41	1686.76
TMV	246.95	589.29	584.01	575.80	1140.73	1608.88
SM	508.28	629.47	543.28	524.98	1192.29	1853.29
SMV	306.37	496.04	482.23	398.43	1253.49	1849.96
TD	562.72	553.57	701.53	893.40	1698.36	2256.45
STMV	173.24	737.53	637.55	660.74	1396.54	1392.90

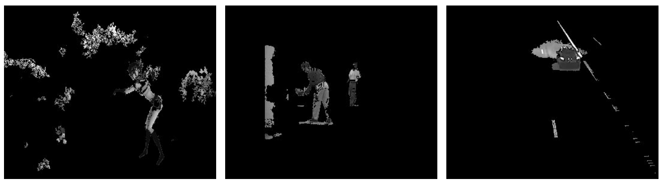
TABLE II

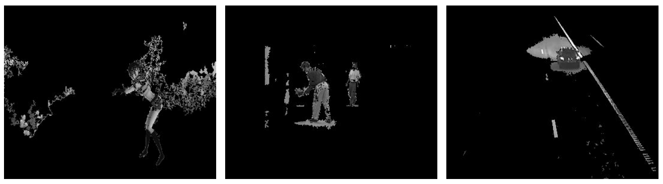
CIF FRAME PROCESSED PER SECOND ACHIEVED WITH DIFFERENT COMBINATIONS OF BACKGROUND MODEL AND UPDATE POLICY. IN THE β CASE THE BACKGROUND MODEL IS COMPUTED FOR ONE HUNDRED TRAINING FRAME AND NEVER UPDATED AGAIN.

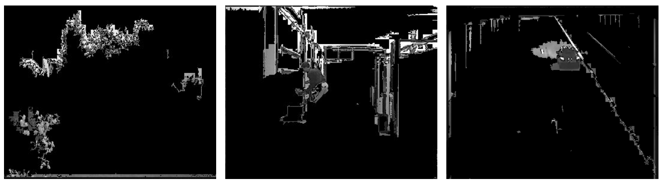
Some conclusions can be drawn from the results presented in Table III-A and Figure III-A. Although a throughput of thousands frames per seconds (fps) is far beyond any practical use, this measure provides a meaningful value for comparing complexity of different combinations. Interestingly high frame rates imply low frame processing time and potentially good power saving in systems equipped with advanced power management features. As clear is Table III-A, the simple TM method used with policy α is almost as complex as the STMV method updated in agreement with policy δ, while the former is absolutely better in detection capabilities. The proposed δ-policy has the advantage of computing the background

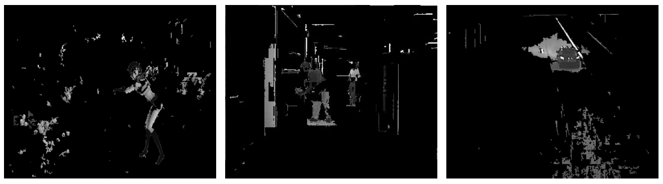

(a) Reference frames.

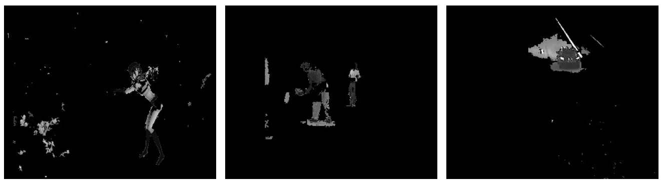

(b) T method


(c) TM


(d) TMV


(e) SM


(f) SMV


(g) STMV

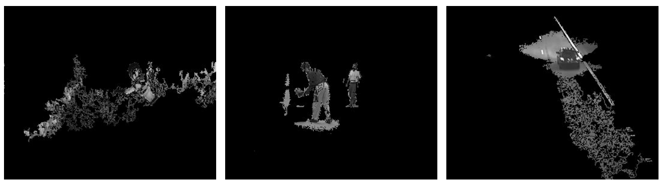

(h) TD

Fig. 3. Experimental results with several videos sources changing the Background model.

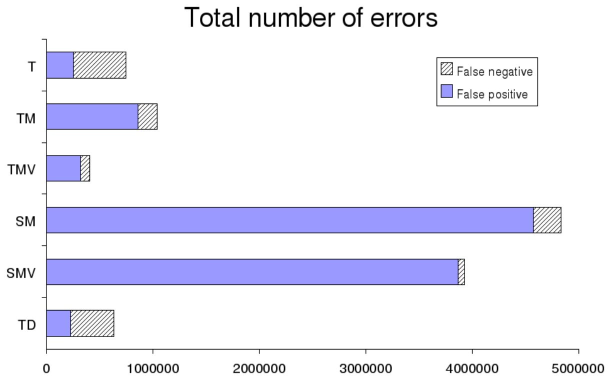


Fig. 4. Number of false and unrevealed motion pixels computed over 100 frames from 5 video sequences.

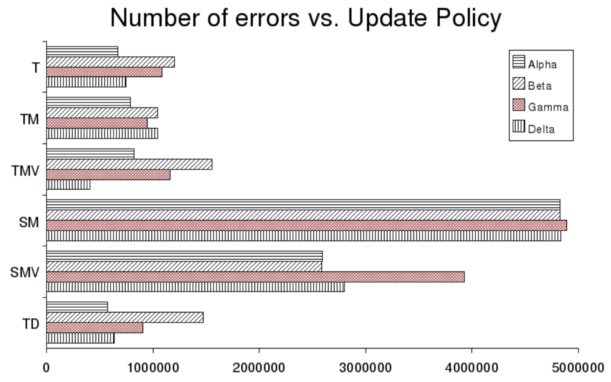


Fig. 5. Number of false motion and unrevealed motion pixels computed over 100 frames from 5 video sequences; different update policies are compared.

model only for those pixels that are strictly necessary for the MD processing, so guaranteeing a remarkable saving of complexity.

IV. Results

We evaluate the performance of the proposed algorithm in terms of MIPS (Millions of Instructions per Second) required for a real time implementation and compare obtained results to other low complexity MD algorithms.

The performed experiments make use of several benchmark video sequences and test suite from VSSN 2006 [13]. The algorithm described in section II was set with P equal to $1/16$ of the image size (making our performance comparable to simulation results presented in [7]). We adopted STMV background model and δ-policy, which offer the best trade-off between performance and detection capabilities.

The frame rate achieved by algorithm in [6] is as high as 300 $PAL\ fps$ (720×576 pixel), while in [7] authors claim 950 $PAL\ fps$ on a Pentium IV 2.4GHz. In [3] the average time for computing a CIF (352×288) frame is reported for several algorithms, ranging from $0.0263s$ (equal to 38 fps) to $0.538s$ on the same processor.

In our experiments the algorithm described in Section II reaches a frame rate of 119 $PAL\ fps$ on an Intel Pentium-M $1.86GHz$; the measured processing complexity includes the background model and update, which is not present in [6], [7];

The choice of P can help to further reduce the complexity; values up to 256 have been tested with reasonable degradation in detection capability. Figure 6 reports some frames grabbed during the execution of our algorithm with different P values and algorithm presented in [7]. The visual comparison denotes small degradation in performance for $P = 64$; in this case the frame rate increases up to 264 $PAL\ fps$. If this algorithm is used only for alarm generation, setting $P = 128$ brings to the image presented in Figure 6(e), with a measured frame rate of 344.39 $PAL\ fps$.

Our system was tested on an ARM processor ($ARM9JSE$ $400MHz$) to demonstrate the applicability of our approach in embedded systems. On this platform, running at the same time Linux and network I/O, the MD can perform up to 57 $PAL\ fps$ demonstrating relatively high real time performance.

In this paper a new low complexity MD technique is proposed. It is analyzed in terms of complexity and detection capability, presenting to the reader data and images to figure out reasonable trade-offs.

References

[1] R. Radke, S. Andra, O. Al-Kofahi, and B. Roysam, “Image change detection algorithms: a systematic survey,” *Image Processing, IEEE Transactions on*, vol. 14, pp. 294 – 307, March 2005.

[2] G. Jing, D. Rajan, and C. E. Siong, “Motion detection with adaptive background and dynamic thresholds,” in *Information, Communications and Signal Processing, 2005 Fifth International Conference on*, December 2005, pp. 41 – 45.

[3] M.-C. Chang and Y.-J. Cheng, “Motion detection by using entropy image and adaptive state-labeling technique,” in *Circuits and Systems, 2007. ISCAS 2007. IEEE International Symposium on*, May 2007, pp. 3667–3670.

[4] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers, “Wallflower: principles and practice of background maintenance,” in *Computer Vision, 1999. The Proceedings of the Seventh IEEE International Conference on*, September 1999, pp. 255 – 261.

[5] C.-F. Chiasserini and E. Magli, “Energy-efficient coding and error control for wireless video-surveillance networks,” *Telecommunication Systems*, vol. 26, no. 2-4, pp. 369–387, june-august 2004.

[6] P. Bassignana, M. Martina, G. Masera, A. Molino, and F. Vacca, “DSP implementation of a low complexity motion detection algorithm,” in *Conference Record of the Thirty-Ninth Asilomar Conference on Signals, Systems and Computers*, November 2005, pp. 1352–1355.

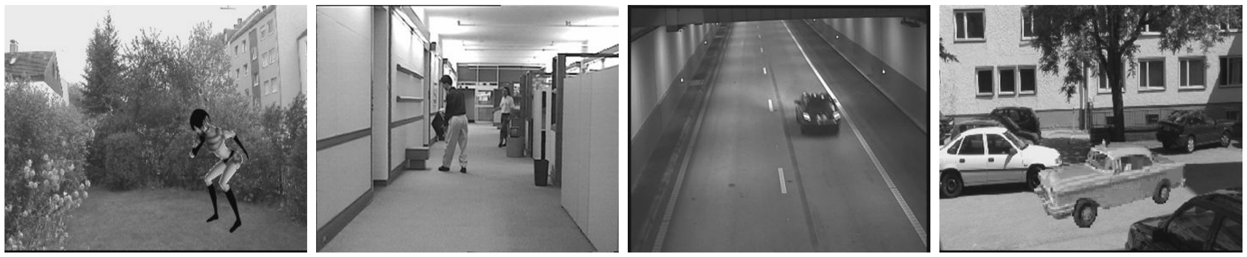
[7] A. Dassatti, G. Masera, M. Nicola, and F. Vacca, “Low resources algorithm for video survelliance,” in *IEEE International Symposium on Communications and Information Technologies, ISCIT*, October 2007.

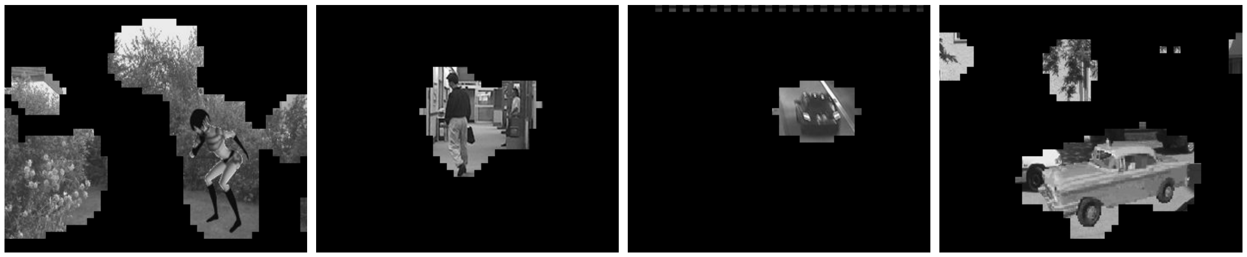
[8] E. Magli, M. Mancin, and L. Merello, “Low-complexity video compression for wireless sensor networks,” in *IEEE International Conference on Multimedia and Expo,*, 2003, pp. 585–588.

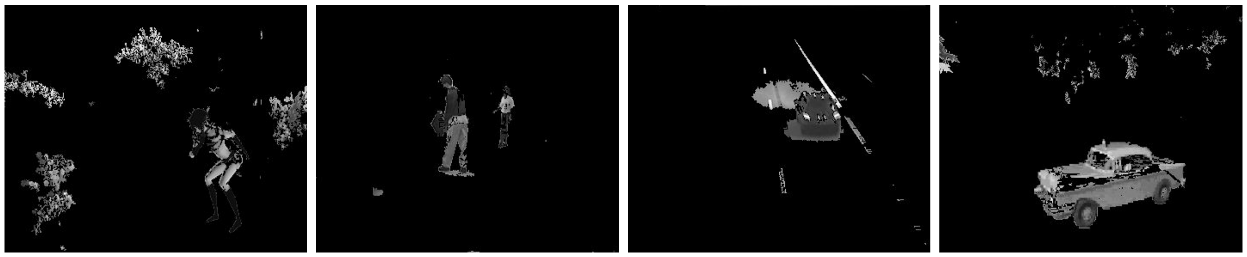
[9] Y. Matsushita, K. Nishino, K. Ikeuchi, and M. Sakauchi, “Illumination normalization with time-dependent intrinsic images for video surveillance,” *IEEE Transaction on Pattern Analysis and Machine Intelligence*, vol. 26, no. 10, pp. 1336–1347, October 2004.

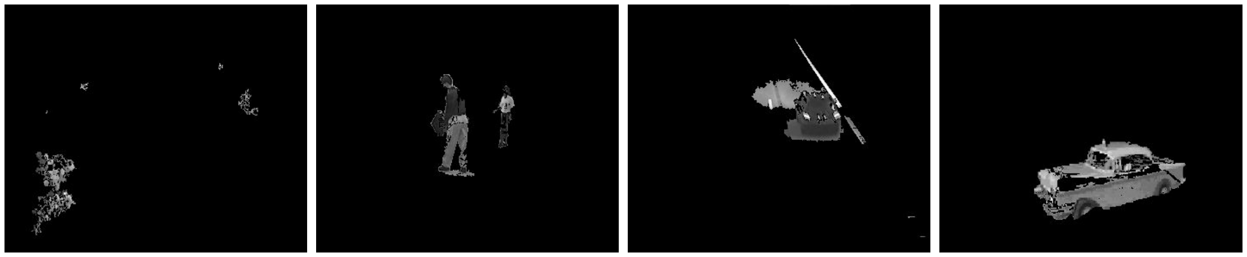
[10] S. H. Lee, S. W. Kim, and S. Kim, “Implementation of a low power motion detection camera processor using a CMOS image sensor,” in *Circuits and Systems, 2004. ISCAS ’04. Proceedings of the 2004 International Symposium on*, May 2004, pp. 737–40.

[11] S.-M. Sohn, S.-H. Kim, S.-H. Lee, K.-J. Lee, and S. Kim, “A cmos image sensor (cis) architecture with low power motion detection for portable security camera applications,” *Consumer Electronics, IEEE Transactions on*, vol. 49, pp. 1227 – 1233, November 2003.


(a) Original Video Frames


(b) Algorithm [7]


(c) Proposed $P = 16$


(d) Proposed $P = 64$

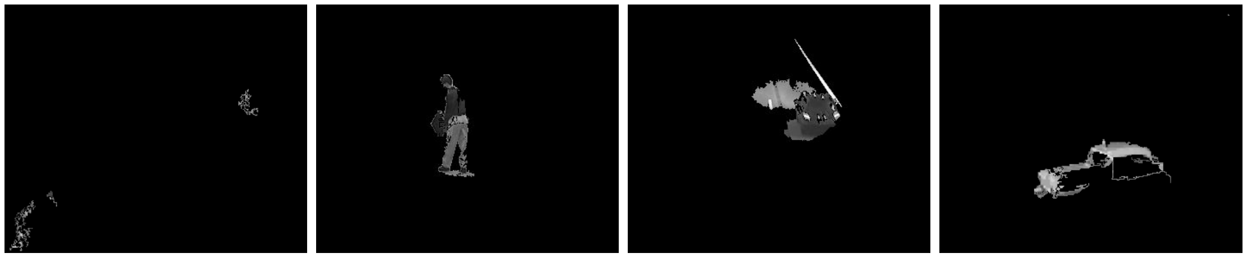

(e) Proposed $P = 128$

Fig. 6. Experimental results with different videos sequences

[12] D. Toth, T. Aach, and V. Metzler, "Illumination-invariant change detection," in *IEEE Southwest Symposium on Image Analysis and Interpretation*, 2000, pp. 3–7.

[13] [Online]. Available: http://imagelab.ing.unimo.it/vssn06/