Efficient Foreground-Background Segmentation using Local Features for Object Detection

Fabio Carrara Giuseppe Amato Fabrizio Falchi Claudio Gennaro
ISTI-CNR
Via G. Moruzzi 1
56124 Pisa - Italy
name.surname@isti.cnr.it

ABSTRACT

In this work, a local feature based background modelling for background-foreground feature segmentation is presented. In local feature based computer vision applications, a local feature based model presents advantages with respect to classical pixel-based ones in terms of informativeness, robustness and segmentation performances. The method discussed in this paper is a block-wise background modelling where we propose to store the positions of only most frequent local feature configurations for each block. Incoming local features are classified as background or foreground depending on their position with respect to stored configurations. The resulting classification is refined applying a block-level analysis. Experiments on public dataset were conducted to compare the presented method to classical pixel-based background modelling.

1. INTRODUCTION

Many computer vision application dealing with object detection, tracking, description, and recognition on static video streams are based on local features (such as SIFT, SURF and ORB). Those kind of applications usually have to apply a pixel-wise background subtraction algorithm (such as Mixture of Gaussian) to isolate the local features of interest (foreground) from the background. These background subtraction methods increase the total computational cost, already high due to keypoint detection, without exploiting local features of the scene for the foreground/background segmentation. Moreover, some features near the foreground boundaries may not be detected since some neighboring parts around the interest point are cut-off.

In this work, a block-wise local feature based background modelling for background subtraction is presented. A local feature-based model presents advantages with respect to classical pixel-based ones in terms of informativeness and robustness, providing a better foreground/background segmentation of the detected features. Moreover a feature-based model of the background is useful to computer vision applications that would anyway detect local features' keypoints, saving resources otherwise reserved to other methods and providing the already detected keypoints for further processing.

The presented foreground detection method has been implemented and tested on the Raspberry Pi platform (equipped with the Pi Camera module) as first step to perform unsupervised learning to recognize 3D objects in a distributed camera scenario.

The rest of the paper is organized as follows: Section 2 describes the main features of some studied background subtraction methods. Section 3 presents our feature based background subtraction method. Section 4 reports the experiments performed and the metrics used to evaluate our method. Conclusive remarks are addressed at the end of this paper.

ICDSC '15, September 08 - 11, 2015, Seville, Spain
© 2015 ACM. ISBN 978-1-4503-3681-9/15/09…$15.00
DOI: http://dx.doi.org/10.1145/2789116.2789136

2. RELATED WORK

Background subtraction methods differ in the type of background model maintained and in the way they evaluate and update it.

Classical methods maintain a model for each pixel independently from each other. Wren et al. [11] proposed to model each pixel with a gaussian probability density function fitting the last n pixel's values. A running (or online cumulative) average is maintained to compute the mean μ and standard deviation σ of the gaussian pdf at each new pixel value. Evaluation is done comparing the current pixel value x to the pixel model: if $|x - \mu| > k\sigma$, the current pixel is considered foreground. Cucchiara et al. [2] makes this model more robust taking μ as a temporal median of the last n values. Those methods have the advantage of a low memory footprint and a high speed, particularly useful in embedded smart camera platforms, but might provide a poor model in case of an active background since they are based on a single scalar value.

A more accurate and widely used approach for background modeling is mixtures of gaussians (MoG) [9]. In this approach, statistics of each pixel are modeled by a mixture of a variable number N of gaussians whose parameters are updated online using the EM (Expectation-Maximization) algorithm on a sliding window of pixel values. This kind of model can represent up to N different "sources" of background in a single pixel and hence are accurate in presence of an active background, but it is more demanding in terms of memory occupation and computational resources with respect to simple gaussian model. Moreover, this method

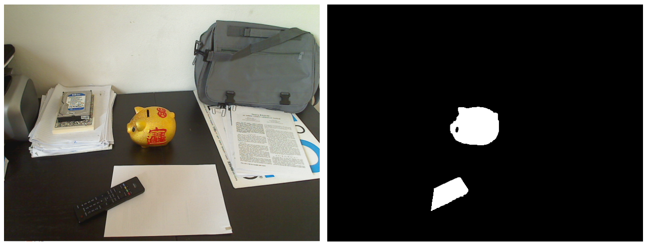


Figure 1: Example of foreground-background missclassification in background subtraction method based on MoG: half of the remote control shares color with the background and therefore is misclassified.

presents some drawbacks: a) since it is color-based, the background subtraction does not perform well if a foreground object we are trying to detect shares colors with the background (see Figure 1), b) training and updating the model with a number $N > 1$ of gaussian components may result in an unwanted *memory effect*. For instance consider the following scenario: a new object appearing in the scene is correctly detected and after a while, thanks to the online training, its pixel values becomes part of the background model. If the object is removed from the scene and then reinserted in approximately the same position, a gaussian component of the model previously trained will match with the object pixel values, marking them as background. This problem is better visualized in Figure 2.

More complex models exploit spatial correlation of pixels in the background model, dropping the assumption of indipendent pixels.

Oliver et al. [6] presents a method based on eigen-value decomposition applied to the whole image. In the learning phase, the mean μ and the covariance matrix C of n background images are computed. Principal component analysis (PCA) is applied to maintain only the M strongest eigenvectors of C, stored in an eigenvector matrix Φ_M. In the evaluation phase, Φ_M is used to project the current frame I to the eigen-space

$$I' = \Phi_M(I - \mu)$$

and than back project it to the image-space, obtaining

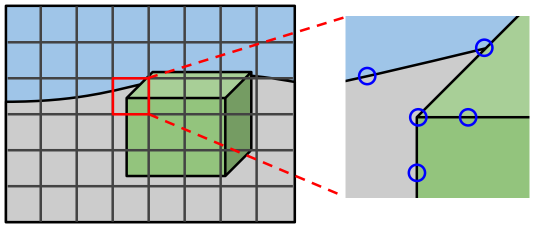
$$I'' = \Phi_M^T I' + \mu$$

Since the transformation keeps only background components, is it possible to identify foreground pixels where $|I - I''| > T$.

Seki et al. [8] exploits block-wise spatial informations stating that neighboring blocks of pixels belonging to the background should experience similar variations over time. For each block of $N \times N$ pixels, its *image variation* (a N^2-component vector) is computed as the average of the mean-subtracted samples of a block. Applying PCA, the dimensionality of an image variation is reduced from N^2 to K and an eigen-space transformation is found. Blocks are classified as background if its image variation in the eigen-space is close to the variations of its neighboring blocks.

3. FEATURE-BASED BACKGROUND MODEL

The presented method, which will be referred to as *LFBBM+* (Local Feature Based Background Model plus) in



(a)

Event Hash	Counter	Timestamp
(28,32,64,67,84)	7	143

(b)

Figure 3: (a) The visualization of an event as simultaneous observation of interest point positions inside a block. (b) Numerical representation of the event.

this paper, is an enhanced version of the method presented by Dehghani and Sutherland [3], which will be referred to as *LFBBM*. The main goal of this method is to segment the local features extracted from the current frame in two sets: the foreground features, potentially belonging to a foreground object, and the background ones, which are fixed and not interesting for our goal.

3.1 Background Model

In LFBBM, a block-wise background model is built. The image coming from the camera is divided into blocks of $W_b \times H_b$ pixels. After the position of all local feature keypoints (KPs) have been detected from the image, they are assigned to the appropriate block based on their (x, y) position in the image. In each block, the set of KPs' positions is called an **event** (Figure 3a). In order to facilitate event labeling, the 2D coordinates of the KPs are mapped in a 1D coordinate by numbering the pixels from 0 at the top left corner of the block, and then counting along each row from left to right to $W_b \times H_b - 1$ at the bottom right corner. The set of those 1D coordinate forms the event hash (see Figure 3b).

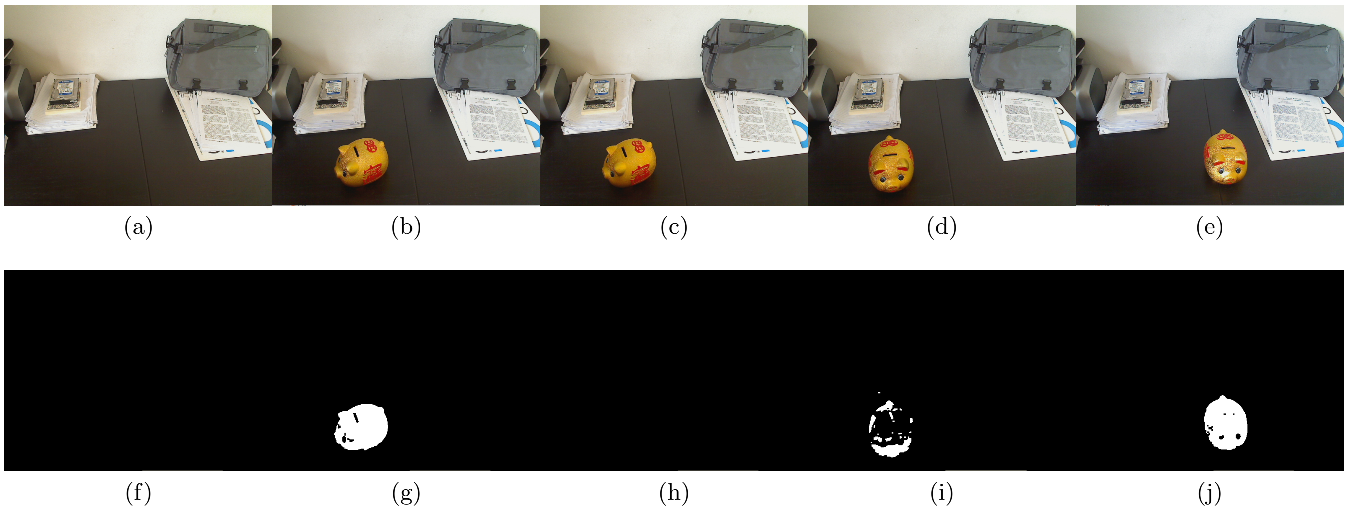
Together with the hash, a counter and a timestamp are maintained for each event. The counter associated with each event is incremented every time a particular event (that simultaneous observation of that group of KPs) occurs.

The whole background model is made by the sets of the occurred events (one set for each block) and their associated counters.

3.2 Learning Phase

The model is continuously updated at every incoming frame: each detected KP is associated with the correct block and becomes part of the current event for that block. Then for each block the event is inserted in the set of the occurred events of that block. If the event is already present in the set, its counter is incremented, otherwise a new counter is created.

In order not to store forever all KPs which occurred at least once, LFBBM+ has been enhanced with an *an aging technique*. A timestamp (frame number) is associated with each event and updated every time that event occurs: if an event does not occur again within a fixed number T_{age} of


(a) (b) (c) (d) (e)

(f) (g) (h) (i) (j)

Figure 2: Example of unwanted memory effect in background subtraction method based on MoG. Frames (a-e) and extracted foreground masks (f-j) of a testing video are shown: background is initially trained (a), then an object is inserted and detected (b). After a while it is inserted in the background model (c). The object is then removed and reinserted in a slightly different position, but its detection is not correct due to memory effect (d). No problem arises if the object is reinserted in a position not overlapping with the first one (e).

frames, it is discarded and removed from the background model. Common values for T_{age} are around 50 frames. In this way, possible memory effects due to intermittent object motion are avoided and in the long run the model will not increase its memory occupancy too much.

In LFBBM+, the data structure used to store and retrieve events of a block is an hash table having event hashes as key instead of the binary search tree suggested by LFBBM. In this way, LFBBM+ has better speed performance at the cost of higher model memory requirements. However, since in LFBBM+ the size of the model is reduced by the aging technique, its final memory requirements are comparable to LFBBM.

An example of background model for a block is reported in Table 1.

3.3 Classification Phase

In each block, a set B of background KP positions is maintained: every time a counter is above a threshold parameter T, all the 1D coordinates belonging to the associated event are inserted in B.

If a position of an incoming KP is present in B, it is considered as a background KP, otherwise it is considered as a foreground one.

After this preliminary classification of the incoming KPs, the following post-processing tasks are executed in order to decrease the misclassification rate:

3.3.1 *Background Zone Enlargement*

Due to acquisition noise, the exact positions of the KPs can slightly change during time. Therefore a background point could be incorrectly classified as a foreground one when its position is not the same as the background point present in the B set.

In LFBBM, all the non dominant events hashes (events that has not reached the minimum threshold to be considered background) are compared using a distance function with all the dominant ones. Non dominant events near enough to dominant ones are still classified as background. This task is very computational demanding since the number of non dominant events in a block can be very high, and therefore also the comparisons to be computed.

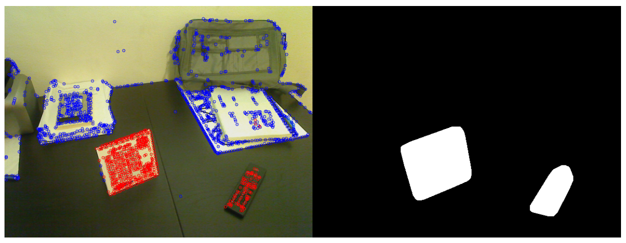

(a) (b)

Figure 4: LFBBM+ output: (a) features are partitioned in Foreground KPs (red circles) and Background KPs (blue circles). (b) The extracted foreground mask.

In LFBBM+, in order to limit this effect, the $r \times r$ neighbors pixels of any background KP are considered as background and are inserted in B. Common values of r are around 3-5. In this way, the precision of the post-processing is slightly decreased but the computational cost is affordable. We did not use more complex and computationally expensive region shapes approaches, such as superpixels [1] or MSER [5], because we expect many keypoints to lie on the edge between regions.

3.3.2 *Neighbor Blocks Analisys*

The blocks of the image can be divided in three different types: 1) *background blocks*, that contain only background KPs, 2) *foreground blocks*, containing only foreground KPs and 3) *mixed blocks*, that contain both. Some blocks are spurious and all their KPs can be correctly reclassified in

Event Hash	**Counter**	**Timestamp**
(16,19,22)	1	187
(10,16,20,23)	2	117
(20)	1	14
(10,23)	1	3
(11)	**11**	**204**
(19,23)	**29**	**220**
(16,18,20,23)	1	76
(20,22)	**5**	**212**
(11,22)	3	215
(18,23)	2	209
(20,22,26)	1	72
(12,23,25)	1	4
(11,16,23,32)	1	151
(23)	**11**	**216**
(11,16,23)	**6**	**210**
(11,23,48)	2	95
(19,22,32)	1	217
...	...	...

(a)

B: 11, 16, 19, 20, 22, 23

(b)

Table 1: (a) Example of background model for an image block (without post-processing tasks), containing all the events occurred (background events are bold). In this example, the background events are obtained applying a threshold $T = 5$ to the counter column. (b) The obtained background positions list B used to classify incoming interest points. It is obtained grouping together the positions of background events.

1	1	1
1	0	1
1	1	1

Table 2: The 2D 3×3 filter used to compute the number of foreground neighbors of a block.

foreground or background points. For example a mixed or foreground block with no foreground blocks in its neighborhood is probably spurious and has to be reclassified as background block. Similarly a background block with many foreground blocks in its neighborhood is probably a foreground one.

To do this, a binary image with one pixel per block is created: each pixel has value 1 if the correspondent block is a mixed or foreground one, 0 otherwise. A 2D 3×3 filter, reported in Table 2, is applied to compute the number of foreground neighbors for each block. If the neighbors count is below a threshold parameter T_{FG} the block and all its KPs are reclassified as background since it has not enough foreground neighbors to be considered foreground. If the neighbors count is above a threshold parameter $T_{BG} > T_{FG}$, the block is surrounded by enough foreground blocks to be considered a foreground one and all its KPs are reclassified as foreground. Common values for those parameters are $T_{FG} = 1$ and $T_{BG} = 4$.

At the end, the output of this algorithm is the partition of the initial set of KPs in foreground and background ones, respectively shown in Figure 4 as blue and red circles.

A foreground mask similar to the ones produced by the background subtraction algorithms is obtained drawing white filled circles centered in the foreground KPs' positions on a completely black mask. Morphological opening operation is applied to the mask, in order to discard singular spurious foreground spots and to fill background holes inside a foreground area. Finally, the convex hull of each isolated white spot is found and filled (see Figure 4).

The presented method is independent from the interest point detection algorithm. Among the available detectors, FAST [7] detection algorithm is suggested for the following reasons:

- its implementation does not use image pyramids. In this stage, we are searching for parts of the frame containing detectable features and we are not interested in finding features robust to scale transformations, hence image pyramids are unnecessary.
- it does not limit the number of detected points. Since the background model is based on interest point positions, detected points should not disappear from a part of the image because of the insertion of something which has stronger points.
- it is usually faster than other detectors, which is relevant in a low resource computing platform.

4. EXPERIMENTS

For evaluation of the background modelling, we have performed a foreground-background segmentation of the local features of some test videos using different methods . Experiments were performed on two public datasets:

4.1 BMC dataset

The BMC dataset[1] [10] is composed by 20 synthetic videos of two outdoor scenes (see Figure 3), divided in "learning phase" videos and "evaluation phase" videos, and 9 real videos. Camera acquisition noise, illumination changes and background movements are artificially added in synthetic videos, emulating a windy and cloudy environment. Only "learning phase" videos have been used, since they come with groundtruth foreground masks for each frame.

4.2 CDW-2012 dataset

The CDW-2012 dataset[2] [4] is composed by different categories of videos and was designed for evaluation of change detection algorithms. However, some of the categories are also suitable for background modelling evaluation, since videos come with groundtruth annotation of each pixel of evaluation frames and with already segmented region of interest. Each video is composed by a first unannotated training part and an annotated evaluation part.

Four background models are evaluated: static frame (frame difference with thresholding or FrameDiff), Mixture of Gaussian model (MoG), LFBBM and LFBBM+. Each algorithm is applied to each video: at each frame FAST keypoints are extracted and classified as foreground or background keypoint. For pixel-based methods (FrameDiff and MoG), the classification is performed looking to the foreground mask obtained by the algorithm. The correct classification is given by the groundtruth foreground masks. For videos of the

[1]Background Models Challenge: http://bmc.iut-auvergne.com/

[2]Change Detection Workshop: http://changedetection.net/

BMC dataset, models are continuously trained and evaluation metrics are computed over the entire video. For CDW-2012 videos, models are trained only during the training part of the video and metrics are computed over the evaluation part of the video only. The parameters of the evaluated methods have been tuned for each category of video used during tests. A visualization of test videos and outputs of each method is reported in Figure 3.

Measures related to binary classification problems are extracted for each video and their averages for each video category are reported in Table 4. Positive and negative classes represent respectively foreground and background points. The following measures are extracted:

Precision $Prec = \frac{\mathrm{TP}}{\mathrm{TP+FP}}$, where TP is the number of correctly classified foreground points and FP is the number of background points incorrectly classified as foreground. This measure represents the fraction of keypoints classified as foreground that are really foreground keypoints.

Recall $Rec = \frac{\mathrm{TP}}{\mathrm{P}}$, where P is the number of foreground keypoints. This measure represents the fraction of all foreground keypoints correctly classified as foreground points.

F-Score $F_1 = \frac{2pr}{p+r}$ which is the harmonic mean of precision and recall. Values reported in Table 4 are averaged for each video category.

Specificity $Spec = \frac{\mathrm{TN}}{\mathrm{N}}$, where TN is the number of correctly classified background points and N is the number of background keypoints. This measure represents the fraction of all background keypoints correctly classified as background points.

Accuracy $Acc = \frac{\mathrm{TP+TN}}{\mathrm{P+N}}$, which represents the fraction of all keypoints correctly classified.

FPS how many frames per second are processed during tests. For each frame, FAST keypoints are extracted and classified using one of the evaluated methods.

5. CONCLUSION

Although pixel-based methods (FrameDiff and MoG) are more precise in background subtraction, they tend to incorrectly classify features on the boundary between foreground and background, hence having low recall values. LFBBM+ provides a foregrond/background segmentation of the local features that is better or as good as the ones obtained with the other evaluated methods. In our tests, LFBBM+ has in general best performances in terms of accuracy and f-score and, on execution time, it is the closest to FrameDiff, a very simple and fast pixel-based method. The loss of recall in LFBBM+ with respect to LFBBM is a minor drawback we have afforded in order to get higher precision and accuracy values. LFBBM+ is more precise on foreground keypoints, whereas LFBBM has a tendency to incorrectly classify keypoints as foreground more frequently.

BMC "Learning Phase" videos

Method	**Prec**	**Rec**	F_1	**Spec**	**Acc**	**FPS**
LFBBM+	0.81	0.81	0.80	0.99	0.98	**FPS**
LFBBM	0.79	0.86	0.81	0.99	0.98	34
MOG	0.72	0.47	0.56	0.89	0.87	24
FrameDiff	0.76	0.73	0.69	0.97	0.95	37

CDW-2012 "baseline" videos

Method	**Prec**	**Rec**	F_1	**Spec**	**Acc**	**FPS**
LFBBM+	0.65	0.89	0.75	0.92	0.92	155
LFBBM	0.57	0.96	0.68	0.84	0.86	130
MOG	0.97	0.18	0.30	1.00	0.89	107
FrameDiff	0.90	0.55	0.67	0.98	0.93	196

CDW-2012 "badWeather" videos

Method	**Prec**	**Rec**	F_1	**Spec**	**Acc**	**FPS**
LFBBM+	0.66	0.40	0.48	0.71	0.54	84
LFBBM	0.76	0.36	0.45	0.84	0.57	71
MOG	0.90	0.04	0.08	0.99	0.44	45
FrameDiff	0.84	0.09	0.16	0.96	0.46	84

CDW-2012 "intermittentObjectMotion" videos

Method	**Prec**	**Rec**	F_1	**Spec**	**Acc**	**FPS**
LFBBM+	0.67	0.85	0.71	0.84	0.84	221
LFBBM	0.68	0.87	0.69	0.82	0.81	193
MOG	0.95	0.10	0.18	1.00	0.71	137
FrameDiff	0.74	0.57	0.62	0.94	0.82	254

CDW-2012 "lowFramerate" videos

Method	**Prec**	**Rec**	F_1	**Spec**	**Acc**	**FPS**
LFBBM+	0.81	0.43	0.53	0.83	0.58	113
LFBBM	0.85	0.48	0.53	0.78	0.57	68
MOG	0.88	0.11	0.19	1.00	0.48	68
FrameDiff	0.91	0.23	0.34	0.95	0.52	133

CDW-2012 "shadow" videos

Method	**Prec**	**Rec**	F_1	**Spec**	**Acc**	**FPS**
LFBBM+	0.48	0.85	0.60	0.85	0.86	192
LFBBM	0.47	0.87	0.57	0.77	0.80	170
MOG	0.98	0.23	0.36	1.00	0.88	120
FrameDiff	0.71	0.52	0.58	0.96	0.89	195

Table 4: Evaluation Metrics: the table shows averaged Precision, Recall, Specificity, F1 Score, Accuracy and FPS obtained by all algorithms tested for each video category.

References

[1] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk. Slic superpixels compared to state-of-the-art superpixel methods. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 34(11):2274–2282, 2012.

[2] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati. Detecting moving objects, ghosts, and shadows in video streams. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 25(10):1337–1342, 2003.

[3] A. Dehghani and A. Sutherland. A novel interest-point-based background subtraction algorithm. *ELCVIA*, 13 (1):Gowri Srinivasa, 2014.

[4] N. Goyette, P.-M. Jodoin, F. Porikli, J. Konrad, and P. Ishwar. Changedetection. net: A new change detection benchmark dataset. In *Computer Vision and Pattern Recognition Workshops (CVPRW), 2012 IEEE Computer Society Conference on*, pages 1–8. IEEE, 2012.

[5] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide-baseline stereo from maximally stable extremal re-

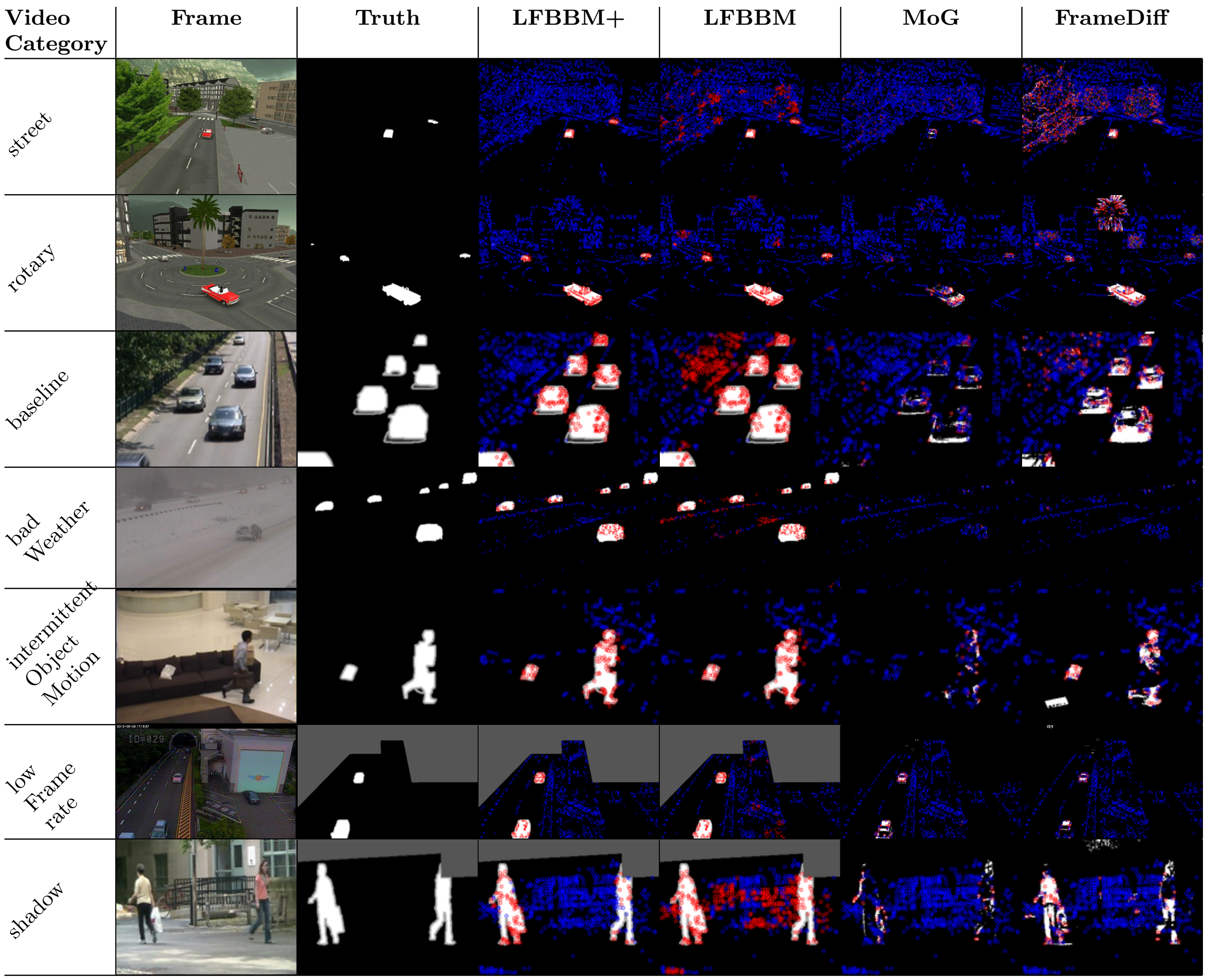


*Table 3: Test videos used for evaluation: the first two videos (*street, rotary*) belongs to BMC dataset, the others to CDW-2012. The first and second columns show the original frame and its groundtruth foreground mask. The remaining columns show the results obtained by LFBBM+, LFBBM, MoG and FrameDiff. Red and blue circles correspond to foreground and background local features. For LFBBM+ and LFBBM, local features are superimposed to the groundtruth foreground mask, while for MoG and FrameDiff, the generated foreground mask is shown.*

gions. *Image and vision computing*, 22(10):761–767, 2004.

[6] N. M. Oliver, B. Rosario, and A. P. Pentland. A bayesian computer vision system for modeling human interactions. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 22(8):831–843, 2000.

[7] E. Rosten and T. Drummond. Machine learning for high-speed corner detection. In *Computer Vision–ECCV 2006*, pages 430–443. Springer, 2006.

[8] M. Seki, T. Wada, H. Fujiwara, and K. Sumi. Background subtraction based on cooccurrence of image variations. In *Computer Vision and Pattern Recognition, 2003. Proceedings. 2003 IEEE Computer Society Conference on*, volume 2, pages II–65. IEEE, 2003.

[9] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *Computer Vision and Pattern Recognition, 1999. IEEE Computer Society Conference on.*, volume 2. IEEE, 1999.

[10] A. Vacavant, T. Chateau, A. Wilhelm, and L. Lequièvre. A benchmark dataset for outdoor foreground/background extraction. In *Computer Vision-ACCV 2012 Workshops*, pages 291–300. Springer, 2013.

[11] C. R. Wren, A. Azarbayejani, T. Darrell, and A. P. Pentland. Pfinder: Real-time tracking of the human body. *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, 19(7):780–785, 1997.