Correntropy-based adaptive filter used as background subtractor

Carlos Gonzalez
MSc Electrical Engineering
Universidad Tecnologica de Pereira
Email: caal.0522@gmail.com

Manuel Pineda
MSc Electrical Engineering
Universidad Tecnologica de Pereira
Email: manuel.felipe.pineda@gmail.com

***Abstract*—The background subtraction task is widely used in computer vision for video segmentation of moving scenes from static cameras. This is the basis for several real world applications as the surveillance systems or the automatic counting of vehicles and pedestrians. We are analyzing an adaptive kernel model to determine if a pixel is background or foreground based on the correntropy as cost function and an automatic tuning strategy for the bandwidth param of the kernel. In this report we describe other methods for background subtraction in order to generate better comparatives. We also describe the experiments that we performed and the way we profiled them.**

***Keywords*—Background subtraction, adaptive filters, correntropy, cost function, kernel methods, RKHS.**

I. Introduction

The *background subtraction* (or *foreground detection*) problem is one of the most common in computer vision, as it is the basis of many common tasks and real world applications, which range from simple object recognition and classification, to much more complex tasks like classifying objects of interest and counting them, for example in the context of traffic applications [1], or recognizing human motion and gestures [2], for example, to detect unusual human activity in security footage.

Being a basis to so many applications, many approaches have been tried in order to solve the problem, ranging from purely statistical methods, to methods based in Frame difference, and including some methods based on a Gaussian approach, or even Neural Networks which have been introduced to deal with the non-stationary aspects of the background detection (e.g. sudden changes in illumination.) [3], we are particularly interested in methods that can be easily taken into the standard *Online Learning Framework*, since this is the particular field this course is interested in.

As this is a report on the chosen problem and possible methodologies and approaches to a solution using the *Online Learning Framework*, a small summary of the state of the art is written in section 2, along with some of the most commonly used models and a simple mathematical description of each one.

Our main proposal will be based on the approach developed in the paper by Alvarez *et al.* [4], as it is built around the *Online Learning Framework*, whilst also providing a Single Gaussian approach, and dealing with the non-stationary aspects, the methods proposed here was implemented in C++ and put to test on real databases in order to provide data on its performance, in terms of computational resources and accuracy.

The remainder of the report is concerned with the way we performed the tests, the different databases used, and the metrics that were taken into account.

A. Formal Problem definition

Given $\mathbf{x_t} \in \mathcal{X}$ as the features vector of one pixel in the time $t = 1 \dots T, t \in \mathbb{N}$ being T the number of frames, our goal is to learn a mapping function $f : \mathcal{X} \mapsto \mathcal{Y}$ in order to determine if a pixel either is background or foreground. $\mathcal{Y} = \{0, 1\}$, being 0 the representation of background and 1 the foreground.

II. State of the art

A. Proposed models of background subtraction

- **Minimum, Maximum and Maximum Inter-Frame Difference (MinMax):**
  The W4 videosurveillance system [5] uses a background model made of a minimum $\mathbf{m_s}$, a maximum $\mathbf{M_s}$, and a maximum of consecutive frames difference $\mathbf{D_s}$. For MinMax, a pixel s in the image $\mathbf{I_t}$ belongs to the background if: $|\mathbf{M_s} - \mathbf{I_{s,t}}| < \tau d_\mu$ or $|\mathbf{m_s} - \mathbf{I_{s,t}}| < \tau d_\mu$ where τ is an user-defined threshold and d_μ is the mean of the largest interframe difference over pixels. Note that MinMax operates on grayscale videos only.
- **Gaussian Mixture Models [3]:** In this algorithm, distributions of each pixel color are represented by the sum of weighted Gaussian distributions defined in a given colorspace.
  As a new image is processed, the GMM parameters (for all pixels) are updated to explain the colors variations. In fact, at time t, it is considered that the model $\mathbf{m_t}$ generated for each pixel from the measures $\{\mathbf{y_0}, \mathbf{y_1}, \dots \mathbf{y_{t_1}}\}$ of a pixel is correct. The likelihood that a pixel is a background pixel is:

$$P(y_t|m_t) = \sum_{n=1}^{N} \frac{\alpha_n \exp\left(\frac{-(y_t-\mu_n)^\top \Sigma_n^{-1}(y_t-\mu_n)}{2}\right)}{2\pi^{d/2}|\Sigma_n|^{d/2}}$$

  Where d is the dimension of the color space. Each Gaussian is described by its mean μ_n and covariance

matrix Σ_n, all the coefficients α_n must hold that $\sum_{n=1}^{N} \alpha_n = 1$.
The update steps with respect to a new image $\mathbf{I}$ are:

- $\alpha'_n = (1-\delta)\alpha_n + \delta$
- $\mu'_n = (1-\rho)\mu_n + \rho\mathbf{I}$
- $\Sigma n' = (1-\rho)\Sigma n + \rho(\mathbf{I}-\mu_n)^\top(\mathbf{I}-\mu_n)$

Where δ is a user defined learning rate and ρ is estimated as $\delta\mathcal{N}(y_t|\mu_n, \Sigma_n)$

- **Kernel Density estimation: [6]** This method uses a non-parametric technique (Parzen window) to estimate the probability density P over the value of the pixel.

$$P(\mathbf{x_t}) = \frac{1}{N}\sum_{i=t-N}^{t-1} K(\mathbf{x_i}, \mathbf{x_t})$$

A pixel is labeled as foreground if it is unlikely to come from this distribution, i.e. when $P(\mathbf{x_t})$ is smaller than a predefined threshold. The hyper parameters of the kernel can be assumed or computed on the fly.

- **Eigenbackground Subtraction [7]:** The eigenspace model is formed by taking a sample of N images and computing both the mean μ_b background image and its covariance matrix C_b.
The covariance matrix can be diagonalized by performing the eigenvalue decomposition $L_b = \Phi_b C_b \Phi_b^\top$ where Φ_b is the eigenvector matrix of the covariance of the data.
In order to reduce the dimensionality of the space, in principal component analysis (PCA) only M eigenvectors (eigenbackgrounds) are kept, corresponding to the M largest eigenvalues to give a Φ_M matrix.
They key idea is that the moving objects do not appear frequently in the same position and for this reason they do not have significant contributions to the mean of the background.
Once the eigenbackground images (stored in a matrix called Φ_{M_b}) are obtained, as well as their mean μ_b, we can project each input image I_i onto the space expanded by the eigenbackground images static parts of the scene, pertaining to the background.
Therefore, by computing and thresholding the Euclidean distance (distance from feature space DFFS) between the input image and the projected image, we can detect the moving objects (foreground) present in the scene: $\mathbf{D_i} = \mathbf{I_i} - \mathbf{B_i} > t$. The output of the pixel j, k for the image i is $\mathbf{y_i} = \mathbf{D_{i_{j,k}}}$

B. Review of publications

1) Approaches based on Frame Difference: In 1998 Lai and Yung proposed a relatively simple and efficient method [8] for detecting stationary backgrounds in an image sequence, which used variance in pixel intensity or color over time to determine the if a pixel formed part of the background or not, this method used a so call "scoreboard" to keep track of this variance over time, in which small positive numbers represented small changes in intensity or color, and negative numbers represented significant changes, for small variances, the average intensity of the pixels is used for the background detection calculation, whilst for larger variances, what they denominate a "running mode" in which all the intensities of the pixel in each image up until are taken into account is used. In general all of this algorithms are similar to the one proposed in [5] are considered to be in the category of **Inter Frame difference**, which use the differences between pixels in each frame to determine the background, these methods, however are not great when handling non-stationary situations [3] such as lighting changes, objects suddenly stopping, camera motion, etc.

Newer methods suggest the use of other concepts such as texture in conjunction with intensity and color for the same purpose[9], and this approach have been even extended to motion detection field [10]. However since we are mostly interested in approaches that involve kernel usage in the online learning framework, as such these methods are only mentioned as to give some background on other methods, and maybe a source of ideas to enhance our implementation.

2) Approaches based on Gaussians, or Mixtures of Gaussians: In 1999 Stauffer, Grissom, *et al*, proposed to model the intensity of each pixel in the RGB space as a mixture of K Gaussians, denominated **GMM** or **Gaussian Mixture Model** [11], in order to handle dynamic backgrounds, this adaptive model remains relevant to date, as numerous authors have taken this model as basis, and enhanced it through various other ideas, including the detection of shadows [12], the use of other concepts, such as textures as means of generalization [13], using Feature detectors such as SURF to suppress unwanted ghosts [14], and was even included in a recent paper [15] that provides a comparison between different implementations of this model (including the aforementioned ones).

Another interesting method, also compared on the article argues that traditional methods do not provide adequate adaptation to both sudden and slow changes in lighting due to a fixed learning rate, the method provides way to adapt said learning rate, so that in can handle sudden changes in illumination[16].

The conclusion of the review paper is that while the original algorithm is still relevant and can handle a lot of changes, including slow illumination changes and objects entering and leaving the scene, any of the other implementations would be better suited for real world usage in this date and age, due to faster convergence times or the ability to handle sudden illumination changes or more dynamic backgrounds, especially the updated versions from 2014 (Shah, Chen) [15].

As a side note, methods using a single Gaussian are a simplified version of the the mixture of Gaussians cases where $K = 1$.

3) Approaches based on Eigenvalues: This method was originally proposed by Oliver, *et al*, in their paper "A Bayesian Computer Vision System for Modeling Human Interactions", as a part of their system to model Human Interactions, and, while their idea is still relevant to this day, being cited in multiple articles about background subtraction and motion detection [17] [18] [19] [9], we could say that the main idea of this approach was absorbed into bigger background subtraction frameworks, rather than stand on its own.

4) Small summary of Comparisons: This small subsection provides a summary of the results provided by Sobral and Vacant [3], comparing different a approaches to background subtraction, specifically the results obtained by using BMC dataset:

- Static Frame Difference method is still one the most inaccurate, yet improving it to a Frame by Frame difference based approach improves it considerably, however, they are still not in the top 5 however due to its poor recall.
- Mixture of Gaussian methods do reach the top 5 methods, and their execution time, memory, and CPU usage are very reasonable considering the results.
- Other methods provide higher overall precision and recall than the mixture of Gaussians, for example some neural based methods, statistical models based on textures, and Non-parametric methods, however their execution time and general resource consumption is far greater.

C. Proposed method to evaluate based on Kernel Adaptive Filtering

The section 2 of [4] describes a Correntropy-based adaptive learning framework for background modeling. This assumes that the considered videos constitute a stochastic process composed by two main dynamics: foreground and background. This method aims to take advantage of the new existing information at each time instant to learn the function f_t (defined in the introduction). The learning process follows the conventional online learning framework as follows:

$$f_t = f_{t-1} - \eta_t \partial f_{t-1}(l(x_t, f_{t-1}))$$

Being $\eta_t \in \mathbb{R}^+$ the learning rate, $l : \mathcal{X} \mapsto \mathbb{R}$ is a cost function and ∂_f is the gradient with respect to f.

Owing to computational cost and model mathematical tractability, pixel statistical distribution is commonly assumed as either Gaussian or Gaussian mixture models. $f_t = \mathcal{N}(\mu_t, \Sigma_t)$. f can be also adapted by updating simultaneously μ_t and Σ_t, particularly they use the following updating rule:

$$\mu_t = \mu_{t-1} - \eta_t \partial \mu_{t-1}(l(x_t, \mu_{t-1}))$$

1) Correntropy-based cost function for adaptive online learning: The authors proposed a Correntropy-based cost function that deals with the non-Gaussian noise fluctuations because of the non-stationarity of the measured process. It is worth noting that the robustness of the cost function is ruled by its parameters. Moreover, its proper tuning influences the learning algorithm performance even more than the choice of the same kernel in terms of reaching the local optimum, rate of convergence, and robustness to impulsive noise during adaption.

The correntropy between two concrete random variables is computed as:

$$C_\phi(U, V) = \mathbb{E}\{K(U - V)\}$$

Where K is a positive definite kernel, generally a Gaussian kernel which is scaled by its bandwidth param ϕ.

Updating the cost function in the previous updating rule we obtain:

$$\mu_t = \mu_{t-1} - \eta_t \partial \mu_{t-1}(C_\phi(x_t, \mu_{t-1}))$$

which can be written as :

$$\mu_t = \mu_{t-1} + \frac{\eta_t}{\phi^2} e_t K_\phi(x_t, \mu_{t-1})$$

with $e_t = x_t - \mu_{t-1}$

Mostly, the kernel bandwidth is selected as a tradeoff between outlier rejection and estimation efficiency.

The authors then propose the following strategy to update the param ϕ taking into account the kurtosis of e_t and the kurtosis of Gaussian distribution.

$$\phi_t = \alpha \phi_{t-1} + (1 - \alpha)\sigma_{e_r}\sqrt{\beta_G / \beta_{e_r}}$$

Due to the updating rule in this strategy is computed over time windows where pixel dynamics are assumed as stationary, the kernel bandwidth ϕ_t allows considering those main video backgrounds varying over time, for example the illumination changes and the background motion objects.

Finally, in order to determine if a pixel is foreground or background, we check if the product of the probabilities of the pixel in each channel is lower that a given threshold.

III. Experiments and discussion

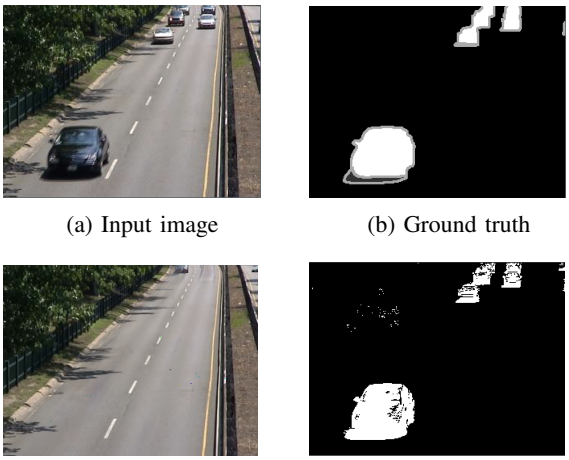
To test the implemented algorithm, we used real datasets (described in the following subsections), running the algorithm on them multiple times to get average metrics on resource usage and performance (the metrics are fully described in subsections III-D and III-C), we also tested an alternative solution, using the same methodology and datasets to get the same metrics, this will be done for purposes of comparison.

A. Real live data sets

We used a set of videos of real-word scenarios provided by the Change Detection Workshop challenge [1], which include videos with non-static cameras and dynamic backgrounds. Each of the videos has a ground truth that we used to profile the algorithms. In particular we used the following three videos:

- Highway: Used as baseline with static camera and static background.
- Boulevard: Video with with heavy camera jitter.
- Canoe: Video with dynamic background.

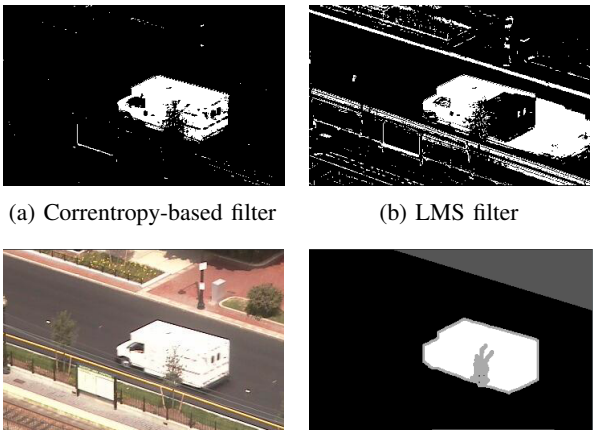
[1] http://www.changedetection.net/



(a) Input image (b) Ground truth

(c) Background model (d) Background mask

Fig. 1: Background subtraction using a correntropy-based adaptive filter



(a) Correntropy-based filter (b) LMS filter

(c) Input image (d) Ground truth

Fig. 2: Correntropy-based filter agains LMS filter in a video with heavy camera jitter

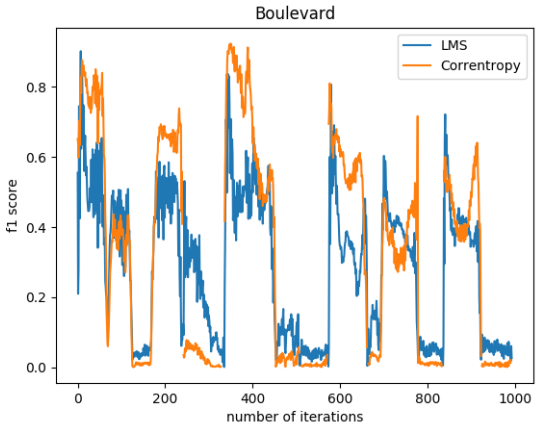


Fig. 3: f1 score boulevard

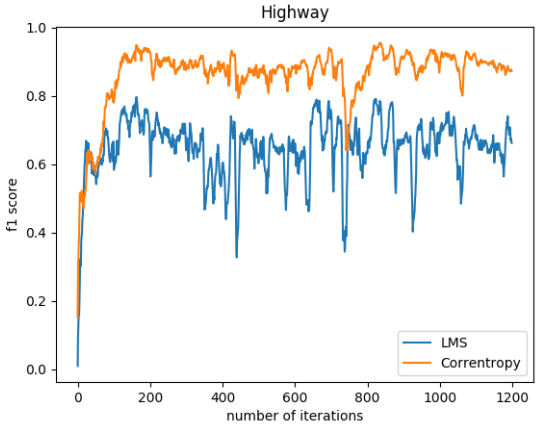


Fig. 4: f1 score highway

B. Implementation details

As we put a filter in each pixel, there is no dependency between the filters. For this reason the code can be parallelized quite straightforward, in this particular implementation we used OpenMP[2] to distribute the filters between all the threads of the computer. All the code used in this work can be found online [3] and it is free to be used in any form.

C. Quality Evaluation

After the background subtraction task we plotted the background mask of each filter in order to visualize their performance.

The figure 1 shows the visual results of the background subtractor using the proposed method based on kernel adaptive filtering, here we can see how the filter learns the background model and correctly detects the moving objects in the scene.

The figure 2 shows a very interesting behavior in which we can observe that the correntropy-based filter is more robust against the small movements of the camera, contrary to the LMS filter which detects other edges in the image as foreground.

As numeric evaluation, we also computed the f1 score of each filter compared against the ground truth, the figures 3, 4 and 5 show the performance of the filters through the time, it is worth noting that the correntropy-based filter performs consistently better on the videos “highway” and “canoe”.

D. Computer Performance

This section shows the performance of the filters in terms of computational resources, the tests where done in a Lenovo ThinkPad x240 with 4 giga bytes of RAM and a Intel(R) Core(TM) i5-4300U CPU @ 1.90GHz processor. Whe measured the following two metrics:

- CPU time: This reflects the complexity of the inner operations needed by each method.

[2]http://www.openmp.org/
[3]https://github.com/pin3da/background-subtraction

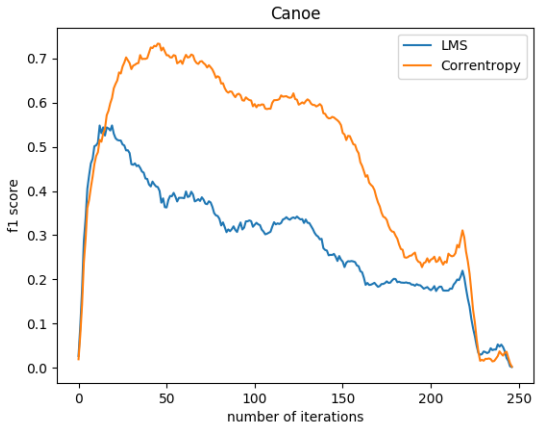


Fig. 5: f1 score canoe

TABLE I: Execution time per frame in seconds

Video	LMS	Correntropy-based
Highway	0.00158312686056	0.436359151839
Canoe	0.00165275015198	0.356236319453
Boulevard	0.00173456433041	0.420305861076

- RAM: This reflects how much information the methods need to perform the task.

The table I shows the average execution time of the filters, in seconds.

The table II compares the memory usage of the filters in mega bytes.

IV. Conclusions

The correntropy-based filter performed better in the datasets than the LMS filter.

The parameter ϕ is very important in the correntropy-based filter, however the calculations can be numerically unstable because we estimate most of the parameters from a small time window.

The threshold used to determine if a pixel is either foreground or background was very effective in both filters, but in the case of the correntropy-based filter, that value was too small (around $1e^{-8}$). This can be a problem in other applications because the precision of the computers.

Thanks to the underlying gaussian model of the correntropy-based filter, it is very robust to the small variations in the camera or the background. We can see for example in the figure 5 how this filter outperforms the basic LMS.

TABLE II: Used RAM in mega bytes

Video	LMS	Correntropy-based
Highway	368	1521
Canoe	365	1591
Boulevard	399	1668

The correntropy-based filter demands more RAM and CPU time, but this can be a good trade-off for its robustness. We need to clarify that the code is not really full optimized as it can be, and maybe the gap between the filters can be smaller.

References

[1] S.-C. S. Cheung and C. Kamath, "Robust background subtraction with foreground validation for urban traffic video," *EURASIP Journal on Advances in Signal Processing*, vol. 2005, no. 14, p. 726261, 2005.

[2] L. Wang, W. Hu, and T. Tan, "Recent developments in human motion analysis," *Pattern recognition*, vol. 36, no. 3, pp. 585–601, 2003.

[3] A. Sobral and A. Vacavant, "A comprehensive review of background subtraction algorithms evaluated with synthetic and real videos," *Computer Vision and Image Understanding*, vol. 122, pp. 4–21, 2014.

[4] A. M. Álvarez-Meza, S. Molina-Giraldo, and G. Castellanos-Dominguez, "Background modeling using object-based selective updating and correntropy adaptation," *Image and Vision Computing*, vol. 45, pp. 22–36, 2016.

[5] I. Haritaoglu, D. Harwood, and L. Davis, "W4— real time detection and tracking of people and their parts,"," *University of Maryland technical report*, 1997.

[6] Y. Benezeth, P.-M. Jodoin, B. Emile, H. Laurent, and C. Rosenberger, "Review and evaluation of commonly-implemented background subtraction algorithms," in *Pattern Recognition, 2008. ICPR 2008. 19th International Conference on*. IEEE, 2008, pp. 1–4.

[7] N. M. Oliver, B. Rosario, and A. P. Pentland, "A bayesian computer vision system for modeling human interactions," *IEEE transactions on pattern analysis and machine intelligence*, vol. 22, no. 8, pp. 831–843, 2000.

[8] A. H. Lai and N. H. Yung, "A fast and accurate scoreboard algorithm for estimating stationary backgrounds in an image sequence," in *Circuits and Systems, 1998. ISCAS'98. Proceedings of the 1998 IEEE International Symposium on*, vol. 4. IEEE, 1998, pp. 241–244.

[9] X. Jian, D. Xiao-qing, W. Sheng-jin, and W. You-shou, "Background subtraction based on a combination of texture, color and intensity," in *Signal Processing, 2008. ICSP 2008. 9th International Conference on*. IEEE, 2008, pp. 1400–1405.

[10] P. Chiranjeevi and S. Sengupta, "Neighborhood supported model level fuzzy aggregation for moving object segmentation," *IEEE Transactions on Image Processing*, vol. 23, no. 2, pp. 645–657, 2014.

[11] 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.*, vol. 2. IEEE, 1999, pp. 246–252.

[12] P. KaewTraKulPong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," *Video-based surveillance systems*, vol. 1, pp. 135–144, 2002.

[13] A. B. Chan, V. Mahadevan, and N. Vasconcelos, "Generalized stauffer–grimson background subtraction for dynamic scenes," *Machine Vision and Applications*, vol. 22, no. 5, pp. 751–766, 2011.

[14] M. Shah, J. D. Deng, and B. J. Woodford, "Video background modeling: recent approaches, issues and our proposed techniques," *Machine vision and applications*, vol. 25, no. 5, pp. 1105–1119, 2014.

[15] K. Goyal and J. Singhai, "Review of background subtraction methods using gaussian mixture model for video surveillance systems," *Artificial Intelligence Review*, pp. 1–19, 2017.

[16] Z. Chen and T. Ellis, "A self-adaptive gaussian mixture model," *Computer Vision and Image Understanding*, vol. 122, pp. 35–46, 2014.

[17] T. Maeda and T. Ohtsuka, "Reliable background prediction using approximated gmm," in *Machine Vision Applications (MVA), 2015 14th IAPR International Conference on*. IEEE, 2015, pp. 142–145.

[18] A. Shimada, H. Nagahara, and R.-i. Taniguchi, "Background modeling based on bidirectional analysis," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2013, pp. 1979–1986.

[19] M. A. Rahman, B. Ahmed, M. A. Hossian, and M. N. I. Mondal, "An adaptive background modeling based on modified running gaussian average method," in *Electrical, Computer and Communication Engineering (ECCE), International Conference on*. IEEE, 2017, pp. 524–527.