Regularised region-based mixture of Gaussians for dynamic background modelling

Varadarajan, S., Wang, H., Miller, P., & Zhou, H. (2014). *Regularised region-based mixture of Gaussians for dynamic background modelling*. Paper presented at AVSS, Seoul, Korea, Republic of.

Regularised Region-based Mixture of Gaussians for Dynamic Background Modelling

Sriram Varadarajan, Hongbin Wang, Paul Miller and Huiyu Zhou
The Centre for Secure Information Technologies (CSIT)
Queen's University Belfast
`svaradarajan01@qub.ac.uk, h.wang@ecit.qub.ac.uk, p.miller@qub.ac.uk, h.zhou@ecit.qub.ac.uk`

Abstract

This paper introduces a momentum-like regularisation term for the region-based Mixture of Gaussians framework. Momentum term has long been used in machine learning, especially in backpropagation algorithms to improve the speed of convergence and subsequently their performance. Here, we prove the convergence of the online gradient method with a momentum term and apply it to background modelling by using it in the update equations of the region-based Mixture of Gaussians algorithm. It is then shown with the help of experimental evaluation on both simulated data and well known video sequences that these regularised updates help improve the performance of the algorithm.

1. Introduction

The online Mixture of Gaussians (MoG) algorithm proposed by Stauffer and Grimson [11] uses a Robbins-Monro type stochastic approximation technique [10] to update the mixture parameters over time. These updates are controlled by a learning rate parameter α that can be tuned to vary the speed of learning of the mixture model. These updates can also be viewed as analogous to gradient methods that are widely used in online learning. The objective of these methods is to minimise the difference between the samples and the means of the mixtures. The standard form for these types of updates is usually

$$\theta^{(t)} = \theta^{(t-1)} - \alpha \nabla f(\theta^{(t)}) \tag{1}$$

The drawback of this approach however is that it is quite slow to converge [4]. One way to speed up the convergence is by introducing an additional term to the update equation called the momentum term. Gradient methods have often been combined with a momentum term that adds a fraction of the difference between the two previous values of the parameters. The momentum term can be seen frequently applied to backpropagation methods for training artificial neural networks. The update equation (1) can thus be modified to include the momentum term as

$$\theta^{(t)} = \theta^{(t-1)} - \alpha \nabla f(\theta^{(t)}) + \beta(\theta^{(t-1)} - \theta^{(t-2)}) \tag{2}$$

The additional term can be viewed as inertia as it takes a small step further in the same direction of the previous update. This inertia reduces oscillations when the direction of the gradient keeps changing. It can also aid the learning rate by pushing the update further towards the optimum value if there is no change in the direction of the gradient. This helps to increase the speed of convergence to the optimum value. This momentum can also help the system escape local minima better than conventional gradient methods.

This was first introduced by Polyak [9] and was called the heavy-ball approach. The heavy-ball method is a batch version of the gradient descent algorithm with an additional momentum term that accelerates convergence. An optimal version of the heavy-ball method was derived by Nesterov [8]. He obtained a superior convergence rate of $O(1/t^2)$ compared to the gradient descent method which had a convergence rate of $O(1/t)$. This is the fastest convergence rate achievable by a first-order method. Beck and Teboulle [2] extended this algorithm to composite functions by using proximal gradient updates and this algorithm is known as FISTA and it is used widely in image processing for image deblurring and image deconvolution. In 2008, Paul Tseng combined all these algorithms and more into a unified framework for accelerated proximal gradient algorithms [13]. All these algorithms are for batch learning, but we show that similar benefits can be gained by using the momentum term in online learning.

Momentum term has also been used previously in various applications like object tracking by Le et al. [6], image segmentation by Andersson et al. [1], image deconvolution by Wang and Miller [16], scene labelling and denoising by Domke [5].

This term can also be used as a regularisation term and was introduced to regularise the classification Expectation-

Maximisation (cEM) based Mixture of Gaussians algorithm by Wang and Miller [15]. In this paper, we apply momentum as a regularisation term to the region-based Mixture of Gaussians approach proposed by Varadarajan et al. [14] to improve the performance of the algorithm.

The spatio-temporal modelling approach with region-based MoG (RMoG) algorithm is briefly introduced in section 2. The convergence of the online gradient method with a momentum term is derived in section 3. This extrapolation step is then applied to the RMoG framework and the background subtraction algorithm is explained in detail in section 4. This is followed by experiments and results of well-known video sequences and comparison with the baseline RMoG approach.

2. Region-based Mixture of Gaussians algorithm (RMoG)

Here, we start by outlining the online RMoG algorithm for dynamic background that was introduced in [14]. This algorithm is a complete framework for MoG modelling extending the standard per-pixel approach to larger regions thus enabling it to model highly dynamic regions in the scene. The standard MoG algorithm considers pixels to be independent of one another hence important spatial cues in the scene where there is dynamic motion like waves rippling or leaving swaying are not captured. The RMoG algorithm takes into consideration this spatial relationship between pixels in a region and builds the model accordingly. It was shown in [14] that this framework models dynamic motion in the scene quite effectively.

The RMoG model can be written in mathematical terms as

$$p(\boldsymbol{x}|\Theta) \propto \sum_{q\in\mathcal{R}_{\boldsymbol{x}}} \sum_{t} \omega_{qk} * \mathcal{N}(\boldsymbol{x}|\mu_{qk}, \Sigma_{qk}) \qquad (3)$$

where $\boldsymbol{x}$ is the pixel (or a feature vector of pixels) under observation, $\mathcal{N}(\bullet)$ denotes a Gaussian distribution and the parameter set is $\Theta = \{\omega, \mu, \Sigma\}$ namely the weight, mean and variance of each mixture in the distribution. $\mathcal{R}_{\boldsymbol{x}}$ denotes the neighbourhood of the feature vector $\boldsymbol{x}$. This usually includes $r \times r$ pixels (or feature blocks) around $\boldsymbol{x}$. The subscript q indicates the location of the chosen mixture component k in the neighbourhood $\mathcal{R}$. The algorithm is also customisable in terms of the feature size and the region size to consider for modelling. For instance, a block of pixels can also be considered as features instead of the conventional approach of using a single pixel within a given region. It can be easily shown that by using a single pixel as a feature in a 1×1 window, the algorithm reduces to the standard MoG approach.

The mixture parameters are updated by using the following equations:

$$\mu_{qk}^{(t)} = (1-\rho)\,\mu_{qk}^{(t-1)} + \rho\left(x^{(t)}\right) \qquad (4)$$

$$\Sigma_{qk}^{(t)} = (1-\rho)\,\Sigma_{qk}^{(t-1)} + \rho\Big(x^{(t)} - \mu_{qk}^{(t-1)}\Big)^2 \qquad (5)$$

$$\omega_{qk}^{(t)} = (1-\alpha)\,\omega_{qk}^{(t-1)} + \alpha \qquad (6)$$

3. Convergence of Online Gradient method with Momentum

In this section, we prove the convergence of the online gradient method with a momentum term. This allows us to use the momentum term in the online RMoG algorithm. The algorithm is usually of the form

$$y^{(t)} = x^{(t)} + \beta^{(t)}(x^{(t)} - x^{(t-1)}) \qquad (7)$$

$$x^{(t+1)} = P_x(y^{(t)} - \alpha\nabla_{\omega^{(t)}} f(y^{(t)})) \qquad (8)$$

where P_x denotes the projection of X on to a closed convex set. Notice the subscript ω for the gradient that indicates that the gradient is equivalent to the expectation of the current observation error i.e. $\nabla_\omega f(y^{(t)}) = E(H(\omega^{(t)}, y^{(t)}))$. Here, $\omega^{(t)}$ are the observations at each instant t.

In order to prove the convergence, we first introduce the Supermartingale Convergence Theorem [3].

Lemma 1 (Supermartingale Convergence Theorem) Let $Y^{(t)}$, $Z^{(t)}$ and $W^{(t)}$, $t = 0, 1, 2, ...$, be three sequences of random variables and let $F^{(t)}$, $t = 0, 1, 2, ...$, be sets of random variables such that $F^{(t)} \subset F^{(t+1)}$ for all t. Suppose that:

1. The random variables $Y^{(t)}$, $Z^{(t)}$ and $W^{(t)}$ are non-negative, and are functions of the random variables in $F^{(t)}$.
2. For each t, we have
$$E\{Y^{(t+1)}|F^{(t)}\} \leq Y^{(t)} - Z^{(t)} + W^{(t)}$$
3. There holds, with probability 1, $\sum_{t=0}^{\infty} W^{(t)} < \infty$

Then, we have $\sum_{t=0}^{\infty} Z^{(t)} < \infty$, and the sequence $Y^{(t)}$ converges to a non-negative random variable Y, with probability 1. ■

For the online gradient method with the momentum term to converge, we assume the following:

1. f is convex, differentiable and finite for all x
2. There exists a finite solution x^*
3. The gradient is Lipschitz continuous with a constant L, i.e.
$$||\nabla f(x) - \nabla f(y)||_2 \leq L||x-y||_2 \qquad \forall x, y \quad (9)$$

The proof continues by expanding the term for $x^{(t+1)}$ and setting the upper limit

$$\begin{aligned} f(x^{(t+1)}) &= f(y - \alpha \nabla_{\omega^{(t)}} f(y)) \\ &= f(y - \alpha E(H(\omega^{(t)}, y^{(t)}))) \\ &\leq f(y) + \nabla f_{\omega^{(t)}}(y)^T (y - \alpha \nabla f_{\omega^{(t)}}(y) - y) \\ &\quad + \frac{L}{2} ||y - \alpha E(H(\omega^{(t)}, y^{(t)})) - y||_2^2 \\ &= f(y) - \alpha ||f_{\omega^{(t)}}(y)||_2^2 + \frac{L\alpha}{2} E(H(\omega^{(t)}, y^{(t)})^2) \end{aligned} \tag{10}$$

The first inequality is by using the quadratic upper bound due to the third assumption.

Now, applying conditional expectation on both sides,

$$\begin{aligned} E(f(x^{(t+1)})) \leq E(f(y)) - \alpha ||f_{\omega^{(t)}}(y)||_2^2 \\ + \frac{L\alpha}{2} E(H(\omega^{(t)}, y^{(t)})^2) \end{aligned} \tag{11}$$

Now, the Supermartingale Convergence Theorem can be applied to this equation as we are dealing with positive values as long as the final term of the equation converges with a probability of 1. Now, assuming $\alpha = \frac{1}{L}$, it is enough to show that the term $E(H(\omega^{(t)}, y^{(t)})^2)$ is bounded.

In online mode, this term refers to the second moment of the updates [4] and can be decomposed as

$$E(H(\omega^{(t)}, y^{(t)})^2 = (\nabla_{\omega^{(t)}} f(y))^2 + var_{\omega^{(t)}} H(\omega^{(t)}, y^{(t)}) \tag{12}$$

This second term given by the variance indicates the noise due to the stochastic nature of the algorithm. This term remains positive at all locations, even at the optimum value. Since the gradient is Lipschitz continuous, it imposes the bound required for the almost sure convergence of the final term of Equation (11).

Therefore, by Lemma 1, it can be said that

$$x_t \xrightarrow[t \to \infty]{a.s.} x^* \tag{13}$$

This proves that the online gradient method with momentum almost surely converges to an optimum value with the probability of 1.

4. Regularisation for Region-based Mixture of Gaussians algorithm

Since hard Expectation-Maximisation type classification is used to assign a pixel to a particular mixture, the non-convex cost function of MoG is converted into smaller convex optimisation problems [15] and hence, the momentum term can be applied to the update equations after assigning the pixel to a particular mixture. The online RMoG can be regularised by using an extrapolation involving the direction of the difference between the previous two values of the mixture parameter. Now, the question would be whether to apply it on all the parameters or apply it on certain parameters. The three parameters under consideration are the Means, Variances and Weights of the mixture components. The weights of the distribution are constrained by normalisation within a given neighbourhood, hence an additional regularisation term will have little or no influence on them. This can also be seen in the experiments from the simulated dataset of [15]. We also noticed empirically that applying the momentum term on the second order Variance term can cause the values to become negative at times thus rendering the system unstable. Even at times when the variance remained positive, it did not have a big influence on the performance. Therefore, we apply the regularisation only on the Mean update equation of the RMoG algorithm given in (4) which can be written as

$$\mu_{qk}^{(t)} = \mu_{qk}^{(t-1)} + \rho \left(x^{(t)} - \mu_{qk}^{(t-1)} \right) - \beta_t (\mu_{qk}^{(t-1)} - \mu_{qk}^{(t-2)}) \tag{14}$$

This application is justifiable because the mean parameter is the one that defines the cluster centers. In the case of background modelling, the means of the clusters are updated based on the pixel values and in regions having highly dynamic motion, the regularisation term helps smooth the mean parameter even if there is a high fluctuation of colour samples.

The momentum parameter β_t is usually between 0 and 1. The momentum term helps to find the optimum value faster, however, by maintaining a large momentum value for a long period of time, it can cause the values to diverge away from the optimum value. Therefore, two choices are available to us. β can either be a decreasing parameter over time [15], or it can increase from 0 to 1 as in the batch approach [3] for a time period proportional to the learning rate and then stopped with subsequent updates taking place without the momentum term so that this divergence does not occur.

Algorithm Regularised Region-based Mixture of Gaussians

1. Consider a series of T images $Y = \{Y_1, Y_2, ..., Y_t, ..., Y_T\}$ where t is the index of the image at the current time instant. The image Y_t (of size $I \times J$ feature blocks) at location (i, j) can be denoted by $Y_t = \{\boldsymbol{y}_{i,j} : i = 1 : I, j = 1 : J\}$ where $\boldsymbol{y}_{i,j}$ are the different feature vectors. The model is given by the parameters $\{\theta_{t,i,j,h} : \mu_{t,i,j,h}, {\sigma^2}_{t,i,j,h}, \omega_{t,i,j,h}\}$ where $h = 1 : H$ is the index of the mixture and K is the total number of mixtures at each location. The size of each

feature vector is dependent on the number of pixels in the feature blocks.

2. Initialise the first mixture of the model with the means equal to the pixel values of Y_1, variances are initialised by a suitable high value scaled by the number of pixels in each block and weights normalised over each region $\mathcal{R}_{i,j}$ given by r. Initialise the difference in the mean values to 0. Initialise the β value.

3. For every subsequent image Y_t, calculate the most likely mixture $\theta_{t,k,l,h}$ where $(k,l) \in \mathcal{R}_{i,j}$ for the reference feature block $\boldsymbol{y}_{i,j}$ (over its entire neighbourhood). This can be calculated by using Euclidean distance.

4. Compare the distance of the most likely Gaussian mixture with a threshold D which is usually a scaled factor of the standard deviation of the mixture. This indicates whether the pixel matches the mixture model or falls outside the model.

5. If a match is found, update the mixture parameters of the above Gaussian $\theta_{t,k,l,h}$ by using Equation (5) for the variance and Equation (14) for the mean with q corresponding to (k,l) and k corresponding to h.

6. Recalculate the differences in the mean values for the next image and update the beta value.

7. If no match is found, create a new Gaussian mixture if there aren't already H mixtures at the reference block location (i,j) or else replace the Gaussian mixture with the lowest weight (at the current block location (i,j)) with a new mixture by initialising it again. Reset the β value for the Gaussian mixture that is reinitalised.

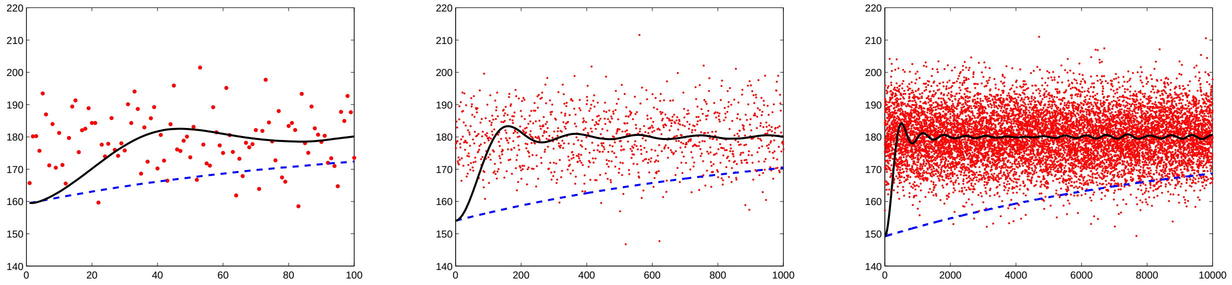
8. The weights are updated with Equation (6) and normalised over the corresponding region.

9. The background model is built with Gaussians having high weights in the region. If the observation falls within this model, it is classified as a background pixel; otherwise it is classified as a foreground pixel.

5. Experimental results

For the first experiment, we compared the convergence of two online gradient methods, one without the momentum term and the other with the additional extrapolation step. We simulated three different sets of data with 100, 1000 and 10000 samples respectively. The samples were generated from a Gaussian distribution with mean 180 and standard deviation 8. This data can be seen analogous to pixel values falling into a particular cluster in a mixture distribution. The parameter was initialised to the minimum value of the samples from each dataset. The value for $\beta^{(t)}$ was chosen as $(t-1)/(t+2)$ that helps to optimally converge in $O(1/t^2)$. Figure 1 shows the convergence for the two different types of updates. In each case, the update method using the momentum term learns at a higher speed compared to the standard gradient method. It has to be noted that the online gradient method with momentum is not a descent algorithm as its batch counterpart. Hence, it will oscillate around the optimum value once it gets close to it. This behaviour is usually not a concern, but it can be easily regulated by applying the momentum term for a specific time period and then reverting to the original form of updates. From Figure 1, it can be seen that, with the momentum term, the algorithm reaches the optimum value around hundred samples while the standard online gradient method takes a few thousand samples to reach the optimum value around 180. Adding the momentum term to the update is not computationally expensive as it only requires the calculation of the difference between the two previous iterates.

In addition to the simulated data, we compared the performance of the regularised RMoG with the baseline RMoG algorithm from [14] on four well known video sequences. These datasets are the bottle sequence [17], the beach sequence [7], the waving trees sequence [12] and a CCTV sequence that was captured by us on board a moving bus.The ROC curves for each of the sequences are shown in Figure 2. We show the results for two different values of the neighbourhood term r in the RMoG algorithm, r=1 and r=8. r=1 corresponds to the standard MoG approach while r=8 is the RMoG with 8×8 regions. It can be seen from the ROC curves that adding the momentum term increases the performance of the algorithm in both cases. An example output from each of the video sequences for different approaches are shown in Figure 3. In the bottle sequence, though the performance of RMoG is very good even without the momentum term, applying regularisation on the updates helps reduce false positives further in the image. Not only does the regularisation term aid in reducing the false positives, but it also helps in reducing the false negatives in some cases as is evident from the output of the bus sequence and the beach sequence. Even in cases where the number of samples are small like the waving trees sequence, adding the momentum term does not affect the performance and the regularised method will work at least as good as the baseline method. The performance gain due to the momentum term will be even higher if the algorithms have to be cold started or there is an increase in the number of samples. This is because there will be a larger contribution from the difference term when the mixtures are randomly initialised leading to faster acceleration. Also, it was seen in the results of the simulated data that as the number of samples increases, the difference between the convergence speed of the two algorithms becomes more pronounced. This be-



(a) Simulated data with 100 samples (b) Simulated data with 1000 samples (c) Simulated data with 10000 samples

Figure 1: Convergence of Gradient methods without a momentum term (Blue/Dashed Line) and with a momentum term (Black/Solid Line) for simulated data samples from a Normal Distribution with Mean - 180, Variance - 64.

haviour will also be true for the regularised RMoG algorithm and hence, there will be a greater difference in the performance between the two algorithms for longer video sequences.

6. Conclusions

In this paper, we proposed a regularisation term for the region-based Mixture of Gaussians background subtraction algorithm by adopting the momentum based acceleration commonly used in backpropagation algorithms for learning artificial neural networks. We proved the convergence of the online gradient descent algorithm with an additional momentum term and applied it to the update equations of the RMoG algorithm. Experiments on both simulated data and video sequences show that adding this term helps learning the dynamic background model faster thereby improving the performance of the algorithm.

References

[1] T. Andersson, G. Lathen, R. Lenz, and M. Borga. Modified gradient search for level set based image segmentation. *IEEE Transactions on Image Processing*, 22:621–630, 2013.

[2] A. Beck and M. Teboulle. A fast iterative shrinkage thresholding algorithm for linear inverse problems. *SIAM Journal on Imaging Sciences*, 2:183–202, 2009.

[3] D. P. Bertsekas. Convex optimization theory by supplementary chapter 6 on convex optimization algorithms, 2013.

[4] L. Bottou. Online algorithms and stochastic approximations. In *Online Learning and Neural Networks*. Cambridge University Press, 1998.

[5] J. Domke. Generic methods for optimization-based modeling. In *Proceedings of the Fifteenth International Conference on Artificial Intelligence and Statistics (AISTATS-12)*, pages 318–326, 2012.

[6] H. Le, L. Hu, and Y. Feng. Momentum based level set method for accurate object tracking. *International Journal of Intelligent Systems and Applications*, 2:10–16, 2010.

[7] L. Li, W. Huang, I. Y. H. Gu, and Q. Tian. Foreground object detection from videos containing complex background. In *Proceedings of the eleventh ACM international conference on Multimedia*, pages 2–10, 2003.

[8] Y. Nesterov. Introductory lectures on convex optimization. a basic course, 2004.

[9] B. T. Polyak. *Introduction to Optimization*. Optimization Software, Inc., Publications Divisions, New York, 1987.

[10] H. Robbins and S. Monro. A stochastic approximation method. *The Annals of Mathematical Statistics*, 22:400–407, 1951.

[11] C. Stauffer and W. E. L. Grimson. Adaptive background mixture models for real-time tracking. In *IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pages 2246–2252, 1999.

[12] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *Seventh International Conference on Computer Vision*, pages 255–261, 1999.

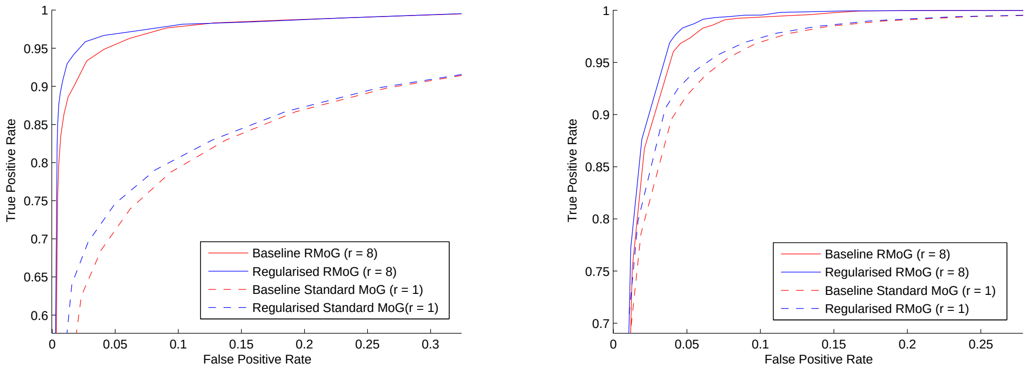
[13] P. Tseng. On accelerated proximal gradient methods for convex-concave optimization. *SIAM Journal on Optimization*, 2008.

[14] S. Varadarajan, P. Miller, and H. Zhou. Spatial mixture of gaussians for dynamic background modelling. In *Advanced Video and Signal Based Surveillance (AVSS), 2013 10th IEEE International Conference on*, pages 63–68, Aug 2013.

[15] H. Wang and P. Miller. Regularized online mixture of gaussians for background subtraction. In *8th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, pages 249–254, 2011.

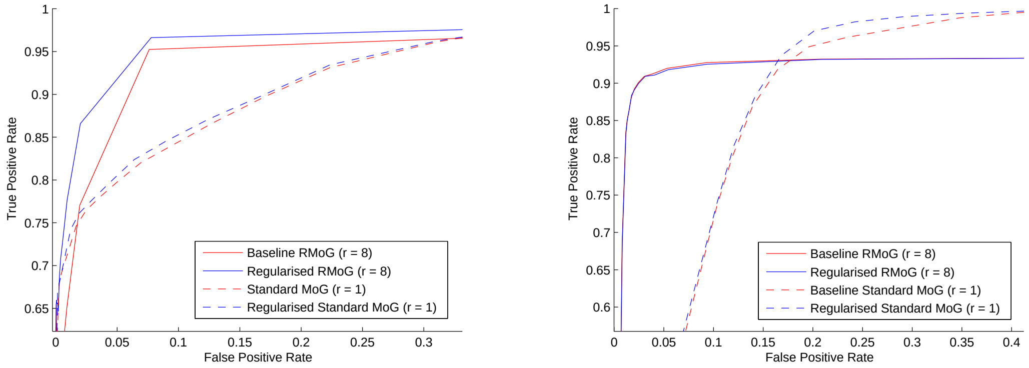
[16] H. Wang and P. Miller. Scaled heavy-ball acceleration of the richardson-lucy algorithm for 3d microscopy image restoration. *IEEE Transactions on Image Processing*, 23:848–854, 2014.

[17] J. Zhong and S. Sclaroff. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. In *Computer Vision, 2003. Proceedings. Ninth IEEE International Conference on*, pages 44–50, October 2003.



(a) Bottle Sequence

(b) Bus Sequence



(c) Beach Sequence

(d) Waving Trees Sequence

Figure 2: ROC curves for the different video sequences

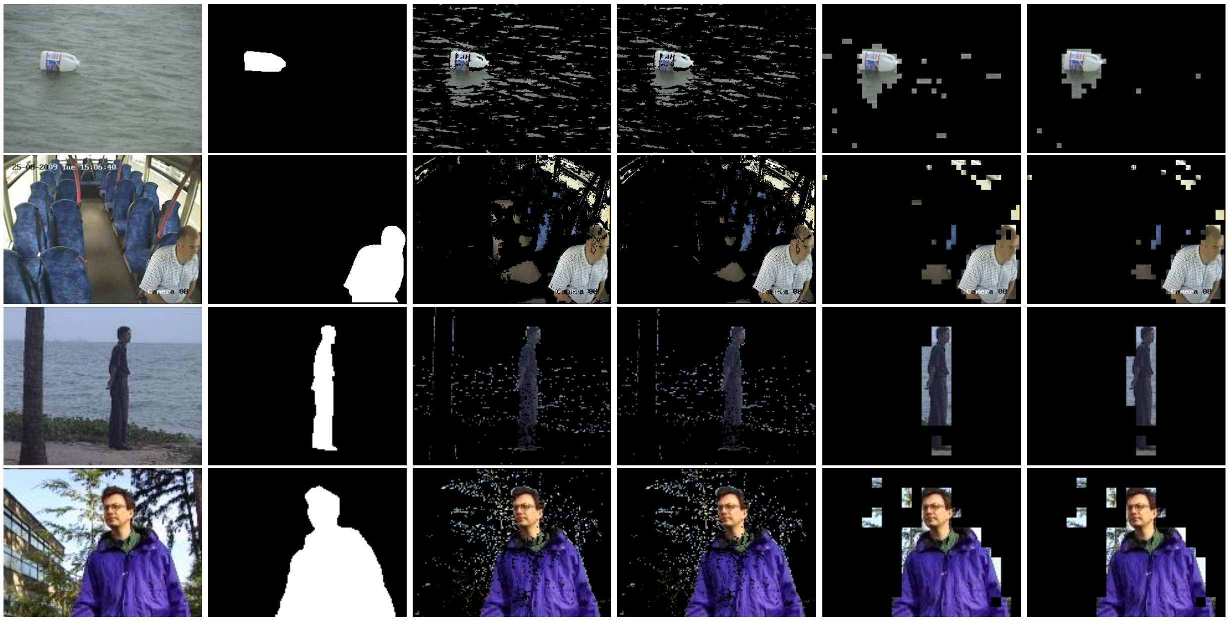


Figure 3: Sample outputs for different video sequences. First Column: Input frames; Second Column: Ground Truth; Third Column: Baseline MoG (r=1); Fourth Column: Regularised MoG (r=1); Fifth Column: Baseline RMoG (r=8); Sixth Column: Regularised RMoG (r=8)