Background Subtraction for Temporally Irregular Dynamic Textures

Gerald Dalley, Joshua Migdal, and W. Eric L. Grimson
Massachusetts Institute of Technology, Computer Science and Artificial Intelligence Laboratory
77 Massachusetts Ave., Cambridge, MA 02139
{dalleyg,jmigdal,welg}@csail.mit.edu
http://people.csail.mit.edu/dalleyg/

Abstract

In the traditional mixture of Gaussians background model, the generating process of each pixel is modeled as a mixture of Gaussians over color. Unfortunately, this model performs poorly when the background consists of dynamic textures such as trees waving in the wind and rippling water. To address this deficiency, researchers have recently looked to more complex and/or less compact representations of the background process. We propose a generalization of the MoG model that handles dynamic textures. In the context of background modeling, we achieve better, more accurate segmentations than the competing methods, using a model whose complexity grows with the underlying complexity of the scene (as any good model should), rather than the amount of time required to observe all aspects of the texture.

1. Introduction

A typical approach in current scene analysis systems is to build an adaptive statistical model of the background image. When a new frame is presented, pixels that are unlikely to have been generated by this model are labeled as foreground. Stauffer and Grimson [11] represent the background as a mixture of Gaussians (MoG). At each pixel, a collection of Gaussians emits values in RGB (red, green, blue) or some other colorspace. When a pixel value is observed in a new frame, it is matched to the Gaussian most likely to emit it. The Gaussian is then updated with this pixel value using an exponential forgetting scheme that approximates an online k-means algorithm. This allows online adaptation to changing imaging conditions such as shifts in lighting or objects that stop moving. Pixel values are labeled as foreground when they are associated with uncommon Gaussians or when they do not match any Gaussian well. This approach lends itself to realtime implementation and works well when the camera does not move and neither does the "background." However, for most applications, objects such as branches and leaves waving in the wind, and waves in water, should be considered as background even though they involve motion. Because these dynamic textures cause large changes at an individual pixel level, they typically fail to be modeled well under a fully independent pixel model. In the middle column of Fig. 5, we see how the MoG foreground mask not only (correctly) includes both pedestrians and the vehicle, but also includes many other pixels due to image noise and moving trees.

More recently, Mittal and Paragios [5] used the most recent T frames to build a non-parametric model of color and optical flow, with care taken to handle measurement uncertainty when estimating kernel density bandwidths. Uncertainty management is especially important here due to the inherent ambiguities in local optical flow estimation. While their approach still models the image as a collection of independent pixels, they produce impressive results when the same motions are observed many times in every block of T frames. Challenges are likely to occur when infrequent motions occur, such as trees rustling periodically (but not constantly) due to wind gusts. Better classification performance results in a cost linear in T. For a 200-frame window, their highly optimized implementation is one to two orders of magnitude slower than typical MoG implementations.

Sheikh and Shah [10] have also developed a kernel-based model of the background using the most recent T frames. Their kernels are Gaussians over the pixel color and location. By allowing observed pixels to match kernels centered at neighboring pixel locations, they are able to interpret small spatial motions such as trees waving in the wind as being part of the background. Like Mittal and Paragios, they must maintain a long enough kernel history to represent all modes in the local background distribution. Fortunately, for many types of scenes, this history length will be shorter for Sheikh and Shah since information can be "shared" by kernels spawned by nearby pixels. We will show that our approach is able to achieve similar sharing benefits, and we do so by including a small set of easily implemented modifications to any standard MoG system.

Nam and Han [6] recently published a background subtraction method that uses particle filtering to track the positions of the generative model's pixel processes. They use a constant velocity (plus Gaussian noise) motion model, and they represent the appearance distribution of an individual pixel process as a color histogram. In order to make the problem tractable, they make several simplifying assumptions that allow for independent decisions in the inference and update stages. Limited quantitative results are given.

Zhong and Sclaroff [15] use an autoregressive moving average model for scenes with highly regular dynamic background textures. For training clips of 96 frames, they retain 80 eigenimages.

Jojic and Frey [3] have taken a radically different approach, extending a model proposed by Wang and Adelson [13]. They consider an image to be generated by a collection of layers, where near layers occlude far ones. Their model assumes that the number of layers and their depth ordering are known and fixed. Each layer is free to translate across the image. Extensions include Winn and Blake's [14] affine motion model. Because finding the optimal solution is intractable, they employ variational approximations to their model. Unlike the other methods mentioned, their approach is batch-mode, so it cannot be used as-is on continuous video feeds.

Our work is most closely related to that of Stauffer and Grimson and of Sheikh and Shah. We combine the usage of a spatial neighborhood in background likelihood estimation with the compactness of a semi-parametric MoG representation. In §2, we describe our generative model and how we perform inference on it. In §3, we then highlight experiments we have performed on our algorithm. We conclude in §4.

2. Our model

We model the image generation process as arising from a mixture of components that have a Gaussian distribution in color and some spatial distribution:

$$p(c_i \mid \Phi) \propto \sum_{j \in N_i} w_j \mathcal{N}(c_i; \mu_j, \Sigma_j)\,, \tag{1}$$

where c_i is the observed color at pixel location i, $\Phi = \{w_j, l_j, \mu_j, \Sigma_j\}_j$ is our model, and N_i is the set of indices of mixture components that lie in the local spatial neighborhood of pixel i. Each component j in our model has an associated mixture weight w_j, a discrete pixel location l_j, mean color μ_j, and color covariance matrix Σ_j. We assume that each observed pixel value is sampled from our model independently. This same assumption is made with nearly all non-layered approaches, including ones that have a spatial component (*e.g.* Sheikh and Shah [10]).

Note that we are not restricted to the RGB colorspace for observations. As with other models, we are free to use other colorspaces (such as YCrCb) or build an observation space over more exotic features such as spatio-temporal gradients [7] or optical flow [5]. In our experience we have found that when a proper neighborhood size is chosen, the background-foreground labeling is less sensitive to the choice of colorspace or the inclusion of optical flow features.

2.1. Foreground-background classification

The primary purpose of most background models is to determine the likelihood that each pixel was generated from the background process. In classic MoG approaches, the model is the same as Eqn. 1, with the constraint that the neighborhood function is degenerate and only selects mixture components at the same location where the colors are sampled, *i.e.* when $N_i = \{j \mid l_j = i\}$. A collection of mixture components is maintained, where only those with the highest weights are considered part of the model and used in the likelihood evaluation. Under the assumptions that all Gaussians have similar covariances, all background Gaussians have comparable weights, and that they do not overlap significantly, the squared Mahalanobis distance

$$d_{ij} = (c_i - \mu_j)^T \Sigma_j^{-1} (c_i - \mu_j) \tag{2}$$

serves as a good proxy for the negative log likelihood, and it can be computed much more efficiently than the precise likelihood value. For the experiments presented in this paper, we have followed this tradition and used the squared Mahalanobis feature for foreground-background classification.

After the model returns the pixelwise likelihood estimates, a higher-level procedure is responsible for classifying each pixel as foreground or background. Common choices for the external classifier often include some combination of a simple thresholder, a Markov Random Field optimizer to perform uncertainty-aware label smoothing, morphological operations to remove isolated foreground detections and merge disjoint blobs, and higher-level detection, tracking, or explicit foreground modeling to filter the results. The external classifier choice is outside the scope of the model itself; we will discuss our choice in §3.

2.2. Model update

A model consisting of a single Gaussian may be updated online as new observations are obtained in an optimal manner by retaining its sufficient statistics. Mixture distributions add the complexity of needing to know which observations were generated from which mixture components. Stauffer and Grimson [11] use an online approximation of expectation maximization. Given a pixel location i, they find the observation likelihood $\mathcal{N}(c_i; \mu_j, \Sigma_j)$ for each mixture component j, and then update its sufficient statistics by assuming an evidentiary weight of $(1-\rho)$ for the old statistics and ρ for the new data point, where $\rho = \alpha\mathcal{N}(c_i; \mu_j, \Sigma_j)$ for some exponential learning rate α. Typical hard EM-like implementations simplify this further by only updating the most likely Gaussian using $1-\alpha$ and α as evidentiary weights. Depending on initialization and the order of online updates, the second approach tends to yield tighter Gaussian distributions that overestimate the covariance less. More recently, Porikli and Thornton [8] used a richer prior model with a greedy update scheme to improve the updates and reduce the effects of the order of updates. All three of these update mechanisms have been used on likelihood models essentially equivalent to Eqn. 1, with the neighborhood size restricted to only consider Gaussians and observations at the same pixel location.

In our model, we allow pixels to be generated from nearby Gaussians. This means each Gaussian has the possibility of independently generating multiple observations and thus potentially needs to be updated from multiple simultaneous measurements. One way of accomplishing this is to retain the time-weighted sample sum $s_j^{(t)}$, squared sample sum $corr_j^{(t)}$, and total effective sample size $e_j^{(t)}$ as follows:

$$s_j^{(t)} = (1-\alpha)s_j^{(t-1)} + \alpha \sum_{i \in N_j} \rho_{ij}^{(t)} c_i^{(t)} \tag{3}$$

$$corr_j^{(t)} = (1-\alpha)corr_j^{(t-1)} + \alpha \sum_{i \in N_j} \rho_{ij}^{(t)} c_i^{(t)} \left(c_i^{(t)}\right)^T \tag{4}$$

$$e_j^{(t)} = (1-\alpha)e_j^{(t-1)} + \alpha \sum_{i \in N_j} \rho_{ij}^{(t)}, \tag{5}$$

where α recursively downweighs old samples, ρ_{ij} is the contribution of the observation at pixel i to Gaussian j, and $\sum_j \rho_{ij} = 1$, and our model parameters are derived as

$$\mu_j^{(t)} = s_j^{(t)} / e_j^{(t)} \tag{6}$$

$$\Sigma_j^{(t)} = corr_j^{(t)} / e_j^{(t)} - \mu_j^{(t)} \left(\mu_j^{(t)}\right)^T \tag{7}$$

$$w_j^{(t)} = e_j^{(t)} / \sum_{j'} e_{j'}^{(t)}. \tag{8}$$

The question at this point is how to assign the update weights ρ_{ij}. There are several logical possibilities, including:

- *Pure Soft:* For each observation, c_i, we update all mixture components that could have generated it, weighting by the likelihood of being generated by that Gaussian, *i.e.*

$$\rho_{ij} \propto \begin{cases} \mathcal{N}(c_i; \mu_j, \Sigma_j) & \text{if } j \in N_i \bigwedge d_{ij} < \tau \\ 0 & \text{otherwise} \end{cases} \tag{9}$$

  where τ is some threshold that allows us to avoid updating poor matches. If no mixture components pass the τ test, we assume some previously-unseen mixture component generated the pixel and we instantiate a new component j' at $l_{j'} = i$ instead of performing an update.

- *Pure Hard:* We choose the single mixture component which was most likely to have generated the sample and update it alone, *i.e.*

$$\rho_{ij} = \begin{cases} 1 & \text{if } j = \arg\max_{j' : j' \in N_i} \mathcal{N}(c_i; \mu_{j'}, \Sigma_{j'}) \\ 0 & \text{otherwise.} \end{cases} \tag{10}$$

  If $d_{ij} \geq \tau$ for the selected component, we perform the new component instantiation as was done for *pure soft* updates.

- *Soft Local:* We perform Stauffer- and Grimson-style soft updates by only updating mixture components at the same location as the observation, *i.e.*

$$\rho_{ij} \propto \begin{cases} \mathcal{N}(c_i; \mu_j, \Sigma_j) & \text{if } l_j = i \bigwedge d_{ij} > \tau \\ 0 & \text{otherwise,} \end{cases} \tag{11}$$

  handling outliers in the usual fashion.

- *Hard Local:* We perform Stauffer- and Grimson-style hard updates, handling outliers in the usual fashion, *i.e.*

$$\rho_{ij} = \begin{cases} 1 & \text{if } j = \arg\max_{j' : l_{j'} = i} \mathcal{N}(c_i; \mu_{j'}, \Sigma_{j'}) \\ 0 & \text{otherwise.} \end{cases} \tag{12}$$

While the *pure soft* scheme is appealing from a Bayesian perspective, it (and *soft local*) also requires that we actually evaluate the likelihoods, $\mathcal{N}(c_i; \mu_j, \Sigma_j)$. The *hard* approaches only require evaluating the much more computationally-efficient squared Mahalanobis distances.

In Fig. 1, we have plotted the relative computational costs of the various update methods, relative to the baseline standard MoG approach (*hard local* with $W = 1$). These plots aggregate the results from running with a variety of parameter settings on several different machines while processing the traffic sequence from Mittal and Paragios [5]. The *local* approaches are relatively unaffected by the neighborhood size, W, since they do not iterate over the whole neighborhood during the update phase. It is clear that the *pure soft* approach incurs a significant additional performance penalty due to its requirements of full likelihood evaluation and that potentially all mixture components in the local neighborhood about a pixel must be updated.

For the same set of experiments, we show in Fig. 2 the costs of producing the Mahalanobis distance maps required

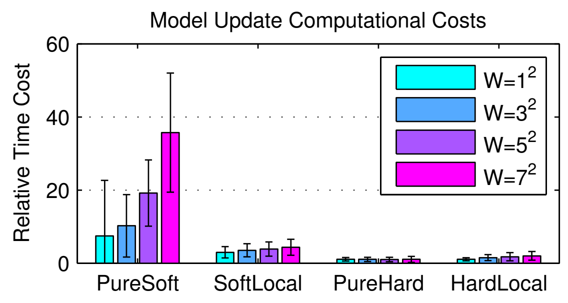


Figure 1. This plot shows the relative length of time required to update the background model after background subtraction has been performed. All times are multiples of the fastest: *pure hard* updates with a neighborhood of size $W = 1$, which is equivalent to the standard optimized MoG approach.

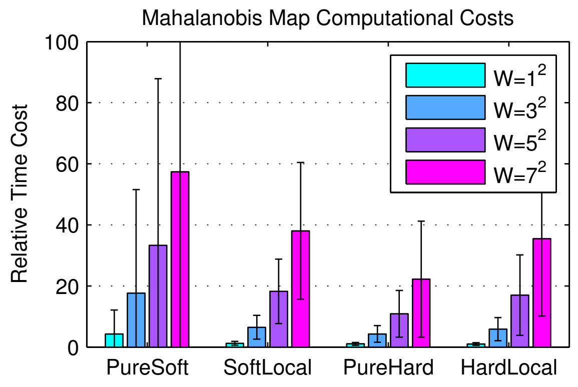


Figure 2. This plot shows the relative length of time required to compute the Mahalanobis distance map used as an input to the foreground/background classification (see §2.1). All times are multiples of the fastest: *pure hard* updates with a neighborhood of size $W = 1$, which is equivalent to the standard optimized MoG approach.

by the foreground/background classifier. Given an update scheme, we expect the the computational cost to rise linearly with the neighborhood size. The update schemes are independent of this step, so it is interesting to note that by either choosing *hard* updates and/or *local* updates, the Mahalanobis calculations become faster. When we do *hard* updates, we force each sampled pixel to effect exactly one mixture component. Similarly, *local* updates can only affect a smaller pool of components. The net affect is that a slightly more compact model can be learned. This more compact model is more computationally efficient as well because fewer Mahalanobis distance calculations are necessary.

3. Experiments

To test our algorithm, we selected several videos from recent publications which attempt to provide better background subtraction in the face of waving trees and/or rippling water. We then hand-labeled all foreground pixels in an evenly-spaced set of frames. Pixels that are ambiguous or are alpha blends of foreground and background are marked as "don't-care" in our labeling and are ignored in our evaluation. Sample frames from the videos are given in Fig. 5.

Like many modern background subtraction systems, we first compute the background squared Mahalanobis distance map

$$m_i = \min_{j \in N_i} d_{ij} \tag{13}$$

and process it with a Markov Random Field (MRF) to classify pixels as foreground or background (see [2], [4]). Our MRF minimizes a standard Potts energy function:

$$E\left(\left\{l_i^{(t)}\right\} \middle| \left\{l_i^{(t-1)}\right\}\right) = \tag{14}$$
$$\sum_i \left(D_i\left(l_i^{(t)}\right) + U\left(l_i^{(t)}, l_i^{(t-1)}\right) + \sum_{k \in \mathfrak{N}_i} V\left(l_i^{(t)}, l_k^{(t)}\right)\right)$$

where

$$D_i\left(l_i^{(t)}\right) = m_i\left(1 - l_i^{(t)}\right) + \lambda_D l_i^{(t)}, \tag{15}$$
$$U\left(l_i^{(t)}, l_i^{(t-1)}\right) = \lambda_T \left|l_i^{(t)} - l_i^{(t-1)}\right|, \tag{16}$$
$$V\left(l_i^{(t)}, l_k^{(t)}\right) = \lambda_S \left|l_i^{(t)} - l_k^{(t)}\right|, \tag{17}$$

$l_i^{(t)}$ is the label of pixel i in frame t (background= 0, foreground= 1), $\mathfrak{N}_i$ is the set of pixel locations in the neighborhood of pixel i, λ_D is the energy applied for choosing a foreground label, λ_T is the energy applied for each temporally-mismatched labels, and λ_S is the energy applied for each pair of 8-connected spatial neighbors with disagreeing labels. There are readily-available implementations which efficiently find the global minimum for Eqn. 14 [1].

3.1. ROC Analysis

If we vary the MRF parameters over the course of a collection of experiments and record the per-pixel classification rates, we are sampling points in the receiver-operator characteristics (ROC) curve. We then may estimate the overall ROC characteristics of the system by taking the convex hull of these points [9].

In Fig. 3, we show the ROC curve for a collection of experiments on the Wallflower waving trees video clip [12]. Each curve uses a different neighborhood size. For this clip, our method shows a clear advantage over the standard MoG in foreground detection rates. Using a 3×3 window is sufficient in this case because the tree motion is limited to a few pixels.

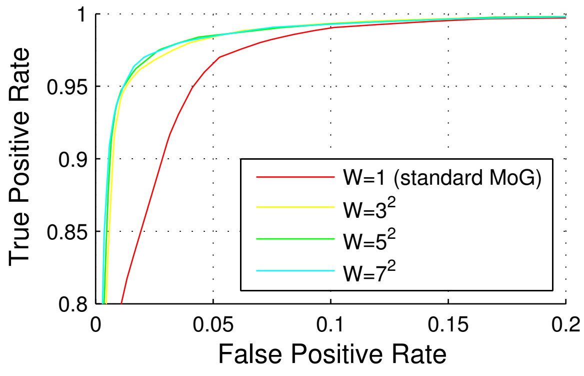


Figure 3. ROC curve for various neighborhood sizes for the Wallflower waving trees clip.

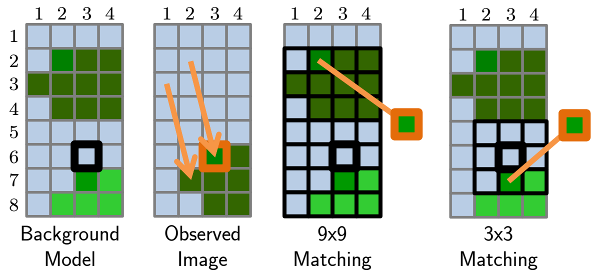


Figure 4. *Exploiting repetitive texture under large inter-frame motion to allow smaller windows:* Consider a background model with one Gaussian per pixel learned from a stationary pair of leaves against the blue sky (leftmost image). We will be concentrating on what happens at the pixel location with the bold outline. Suppose a sudden gust of wind moves the top leaf down and right, as shown in the middle-left image. We now wish to find the best matching background Gaussian for the pixel outlined in brown. To match the Gaussian that actually generated it, we would need to have a 9×9 window (middle-right image); however, a smaller 3×3 window allows matching to a similar leaf in the model and labeling the pixel as background.

For videos where background objects move several pixels between pairs of frames, larger windows can provide additional benefits; however, they are often unnecessary. There is no expected benefit for choosing a window size greater than $W = \max(|dx|, |dy|)$ where (dx, dy) is the largest expected motion vector arising from the dynamic texture. Fortunately, many types of dynamic textures are spatially repetitive and allow us to use much smaller windows, as illustrated in Fig. 4. For most scenes, we have found that a 3×3 window lowers the false positive rate of foreground detection without unduly raising the false negative rate or becoming too computationally expensive.

3.2. Experiments on Various Scenes

In Fig. 5 we show comparative results from a few selected video frames of our test videos. In the first two rows, we have the input frames and the best known results to date. The third and fifth rows show the Mahalanobis distance to the closest matching appearance model when using a MoG and our model, respectively. The images are thresholded and intensity-scaled for visualization purposes. The fourth and sixth rows are final masks after applying an MRF and morphological operations to the Mahalanobis distance map. The final row contains hand-labeled ground truth where each object is given a different hue and don't-care pixels are shown in a lighter shade. The video clips are ordered from left-to-right as easiest to hardest.

The first column is from the classic wallflower paper by Toyama *et al.* [12]. For 200 frames, the scene is empty and an off-camera person vigorously and continuously shakes the tree, producing a semi-regular dynamic texture. A person enters the scene and a single frame is labeled with ground truth. The "Best Published" results are the best results from the original paper. For the MoG and our method, we used the same parameter settings (except the neighborhood size). When looking at the Mahalanobis distance maps, our approach suppresses the waving trees much more effectively than the traditional MoG and is still able to pick up the person very well.

The second and third columns are from frames 410 and 150 of Mittal and Paragios' traffic video [5] and we present their results in the second row. The graded appearance of these results suggests they have used higher-level modules to detect the vehicles and suppress any noise not corresponding to a vehicle detection. Also note that the detected foreground regions are significantly smaller than the entire vehicle. An MoG model is unable to fully suppress the false positives due to waving foliage, even as we allow its parameters to be optimized independently. Our model is able to suppress all false positives in these two frames and can even correctly detect the two pedestrians in frame 150.

The fourth column is from frame 766 of a very challenging sequence courtesy of Sheikh and Shah. We are unaware of any existing published results on the sequence. The scene consists of rippling water, large-leafed tropical plants blowing in the wind, and two small ducks swimming by that are very close in color to the water. Our model is able to produce cleaner silhouettes and have fewer blob-level mistakes than the best MoG results.

The final column is from frame 36 of a Zhong and Sclaroff dynamic texture clip [15]. Our foreground detections are cleaner and more fully capture the bobbing jug.

In practice, we have found the performance of our method to be consistent with these results as it has been employed in traffic surveillance, indoor activity monitoring, and coastal ship tracking (with mild to moderate waves). For more challenging water scenery or places with very consistent dynamic textures, we have found the usage of optical flow and/or spatio-temporal derivatives to be useful as additional modeling features.

4. Conclusions

In this paper, we have introduced a new image generation model that takes into account the spatial uncertainty of dynamic background textures. Our model is much more compact than recent methods ([15],[10],[5]) that have been introduced to handle this problem. Ours can be readily implemented in the familiar mixture of Gaussians framework and it performs better than competing methods.

5. Acknowledgements

This work was funded in part by the Defense Advanced Research Projects Agency (DARPA).

References

[1] Yuri Boykov and Vladimir Kolmogorov. An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision. *PAMI*, 26(9), September 2004.

[2] Yuri Boykov, Olga Veksler, and Ramin Zabih. Fast approximate energy minimization via graph cuts. *PAMI*, Nov. 2001.

[3] Nebojsa Jojic and Brendan Frey. Learning flexible sprites in video layers. In *CVPR*. IEEE, 2001.

[4] Joshua Migdal and W. Eric L. Grimson. Background subtraction using markov thresholds. In *MVC*, 2005.

[5] Anurag Mittal and Nikos Paragios. Motion-based background subtraction using adaptive kernel density estimation. In *CVPR*. IEEE, July 2004.

[6] Woonhyun Nam and Joonhee Han. Motion-based background modeling for foreground segmentation. In *VSSN*. ACM, Sept. 2006.

[7] Robert Pless, John Larson, Scott Siebers, and Ben Westover. Evaluation of local models of dynamic backgrounds. In *CVPR*, pages 73–78, 2003.

[8] Fatih Porikli and Jay Thorton. Shadow flow: A recursive method to learn moving cast shadows. In *ICCV*, 2005.

[9] M. Scott, M. Niranjan, and R. Prager. Realisable classifiers: improving operating performance on variable cost problems. In *BMVC*, 1998.

[10] Yaser Sheikh and Mubarak Shah. Bayesian object detection in dynamic scenes. In *CVPR*. IEEE, 2005.

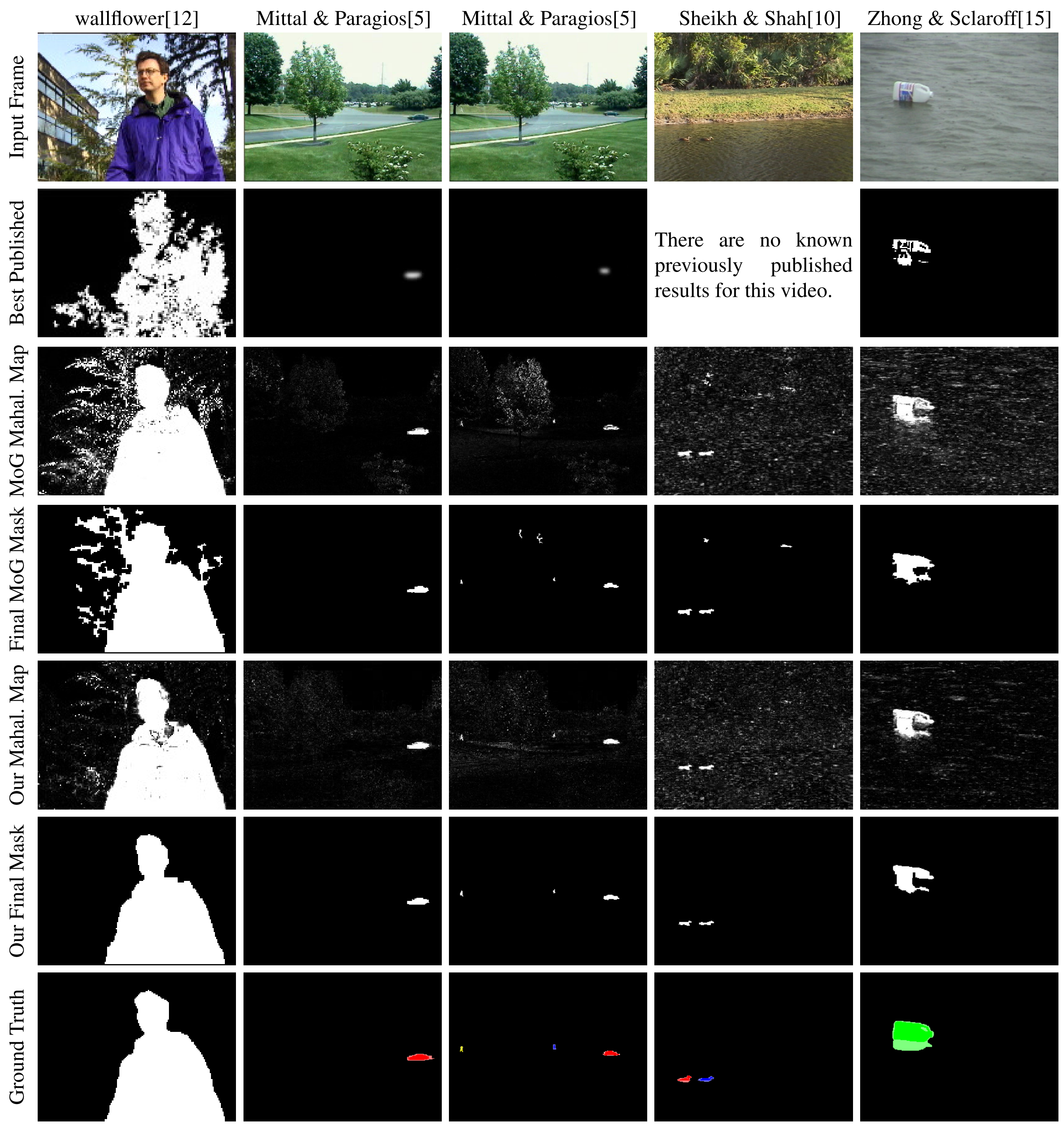
[11] Chris Stauffer and W.E.L. Grimson. Adaptive background mixture models for real-time tracking. In *CVPR*, 1999.

[12] Kentaro Toyama, John Krumm, Barry Brumitt, and Brian Meyers. Wallflower: Principles and practice of background maintenance. *International Conference on Computer Vision*, 1:255, 1999.

[13] John Wang and Edward Adelson. Representing moving images with layers. In *Transactions on Image Processing*, Sept. 1994.

[14] John Winn and Andrew Blake. Generative affine localisation and tracking. In *NIPS*, 2004.

[15] J. Zhong and Stan Sclaroff. Segmenting foreground objects from a dynamic textured background via a robust kalman filter. In *ICCV*. IEEE, 2003.



There are no known previously published results for this video.

Figure 5. Selected background subtraction results comparing the best existing methods with a standard MoG model and our extended MoG model. For the Wallflower example, the "best published" result refers to the result from the proposed method in that paper [12]. Refer to the text in §3 for a discussion of these images.