The 2010 IEEE/RSJ International Conference on Intelligent Robots and Systems
October 18-22, 2010, Taipei, Taiwan

A Novel Approach for Object Extraction from Video Sequences Based on Continuous Background/Foreground Classification

Thiago C. Bellardi, Jorge Rios-Martinez, Dizan Vasquez and Christian Laugier

Abstract— **In many computer vision related applications it is necessary to distinguish between the background of an image and the objects that are contained in it. This is a difficult problem because of the constraints imposed by the available time and the computational cost of robust object extraction algorithms.**

This report describes a new method that benefits from state of the art background/foreground classification combined with the strong theoretical foundations of clustering. The pixels on the scene background are modeled as Mixtures of Gaussians and the output of the classification process are continuous values representing the likelihood that each pixel belongs to the foreground. The clustering is based on a Self Organizing Network (SON) which has a robust initialization schema and is able to find the number of objects in an image or grid. The algorithm's complexity is linear with respect to the number of pixels or cells.

I. Introduction

Detection of moving objects is a fundamental task in video surveillance applications like tracking, traffic monitoring and activity recognition. One common approach to perform moving object detection from static cameras is resumed in two steps: (1) *background subtraction* then (2) *object extraction*.

The *background subtraction* step aims to label pixels as belonging to one of two classes - background and foreground [1], and constitutes an active research domain. The interested reader is referred to [2] for an overview of the field's state of the art. Usually, the output of most background segmentation techniques consists of a binary bitmap image, where values of 0 and 1 correspond to background and foreground, respectively (*eg* [3], [4], [5]).

Having such a bitmap the *object extraction* step consists of grouping together foreground pixels to obtain candidate objects. One common approach to object extraction proceeds by finding 4 or 8-connected components. This is done using efficient algorithms whose time complexity is linear with respect to the number of pixels in the bitmap [6], [7]. A problem with this approach is that it usually produces many small regions which may correspond to noise or to larger regions which failed to merge.

One approach to dealing with this situation is to filter out regions composed of less than a given number of pixels [8]. Although this approach is fast, it has the drawback of assuming that all small regions are noise, which, in many situations, is clearly not the case. A second approach consists of relaxing the neighborhood criterion by assuming, for example, that regions separated by one background pixel are still connected. The usual way of doing this is by pre-processing the bitmap image using morphological operators (*eg* dilation, closing), which have the effect of "thickening" the pixels and "filling in" the holes [9]. Two problems with this approach are the difficulty of finding the appropriate parameters for the operators and the lack of clear physical interpretation of the operators' parameters. A third approach to object extraction is the use of clustering techniques to group pixels. This opens up the possibility of choosing between a plethora [10] of different algorithms having well understood theoretical properties. On the other hand, most of the robust clustering algorithms (*eg* [11], [12]) have three problems when applied to object extraction: a) the number of objects to be found should be known beforehand, b) the algorithms' performance is strongly dependent on the initialization and c) most algorithms are just too complex to be used in systems subject to demanding real-time constraints.

In our approach to the *background subtraction* task we use Mixture of Gaussians (MoG) to model the background pixels on the scene. The output of the classification process is a continuous gray scale image, where the pixel intensity, which varies between 0 and 1, reflects the likelihood that the pixel belongs to the foreground (fig.1(e)).

After that, object extraction is done by means of a clustering algorithm based on Self Organizing Networks (SON) which, in previous works, has been applied to images [13] and occupancy grids [14], showing that it is able to produce good results in real time. This paper improves the clustering algorithm by enabling it to process continuous input pixel values while maintaining a linear complexity with respect to the size of the input image.

This work has been partially supported by the european BACS Project and CONACYT 250140/308006.

T. C. Bellardi, J. Rios-Martinez and C. Laugier are with LIG & INRIA Rhone-Alpes, France
Dizan Vasquez is with ITESM, Mexico
`tbellardi@gmail.com`
`jorge.rios-martinez@inrialpes.fr`
`dichodaemon@gmail.com`
`Christian.Laugier@inrialpes.fr`

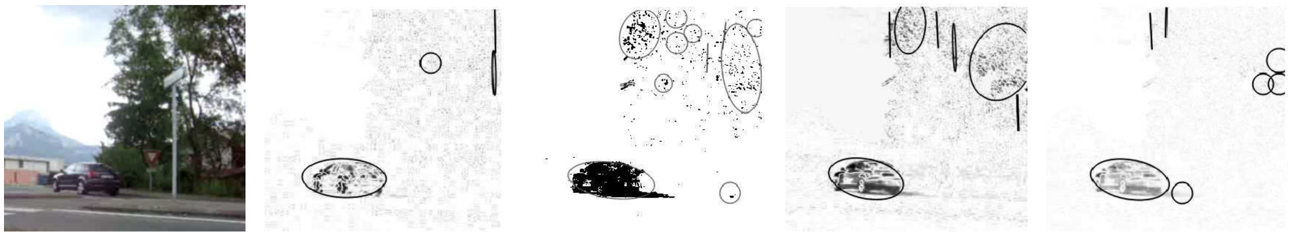

(a) Original Image (b) Absolute difference output (c) MoG binary output (d) Single Gaussian continuous output (e) MoG continuous output

Fig. 1. Examples of output from the different classification bg/fg algorithms compared. Ellipses correspond to the output of the clustering algorithm.

II. Background subtraction

A. *Modeling the background with MoG*

The Mixture of Gaussians (MoG) method was first proposed in [15] and has been implemented, reviewed and improved in many approaches presented in the literature. This method has the capacity to represent multimodal and time-varying backgrounds which are common in outdoors scenarios.

The values for a particular pixel in the image are represented as a Mixture of Gaussians. Based on the mean value and the variance of each Gaussian of the mixture it's possible to determine which Gaussian may correspond to background colors. Pixel values that don't match any of the Gaussians in the mixture are considered to belong to the foreground but, if they continue to be observed, the algorithm is able to include them in the background model by creating a new Gaussian in the mixture. Our implementation of this model mainly follows the adaptation suggested by [16], which gives us fast parameter stabilization. We have introduced as well some alternatives proposed in [17] in order to get a faster learning rate adaption.

We begin by describing the basic approach as presented in [15]. The method models the pixels as a mixture of K Gaussian distributions where the probability of observing a current pixel X_t, is

$$P(X_t) = \sum_{i=1}^{K} \omega_{i,t} N(X_t, \mu_{i,t}, \Sigma_{i,t}) \tag{1}$$

where K is the number of distributions, $\omega_{i,t}$ is an estimate of the weight of the i^{th} Gaussian in the mixture at time t, $\Sigma_{i,t}$ is the covariance matrix of the i^{th} Gaussian in the mixture at time t, and N is a Gaussian probability density function:

$$N(X,\mu,\Sigma) = \frac{1}{(2\pi)^{\frac{1}{2}}|\Sigma|^{\frac{1}{2}}} e^{-\frac{1}{2}(X-\mu)^T \Sigma^{-1}(X-\mu)} \tag{2}$$

K is typically a value between 3 and 5.

Taking in account the RGB color space and assuming an axis aligned covariance, we can write the covariance matrix for Gaussian k in an specific instant as

$$\Sigma_k = diag[\sigma^2_{k,R} \quad \sigma^2_{k,G} \quad \sigma^2_{k,B}] \tag{3}$$

Every new pixel value X_t is checked against the existing K Gaussian distributions, until a match is found. A match is defined as a pixel value within $\lambda = 2.5$ standard deviations of a distribution. The weight of distributions is updated as

$$\omega_{k,t} = \omega_{k,t-1} + \alpha(M_{k,t} - \omega_{k,t-1}) \tag{4}$$

where α is the learning rate and $M_{k,t}$ is one for the matching model and zero for the remaining models. In [15] the Gaussians in the mixture must be ordered according to weight divided by standard deviation, $\omega_{k,t}/\sigma_{k,t}$, but we use only $\omega_{k,t}$ to order the Gaussians, as proposed by [18].

Once a match is found, the parameters μ and σ for the corresponding distribution are updated as follows:

$$\mu_t = \mu_{t-1} + \rho(X_t - \mu_{t-1}) \tag{5}$$

$$\sigma^2_t = \sigma^2_{t-1} + \rho\left((X_t - \mu_t)^T (X_t - \mu_t) - \sigma^2_{t-1}\right) \tag{6}$$

$$\rho = \alpha N(X_t, \mu_k, \sigma_k) \tag{7}$$

The parameters for unmatched distributions remain the same. If none of the K distributions match the current pixel value, the least probable distribution is replaced with a distribution with the current value as its mean value, an initially high variance and low prior weight, in our case 900 and 0.05 respectively.

TABLE I
PARAMETERS OF THE MOG

Parameter	Value	Assumption
No. of Gaussian Models (K)	3	-
No. of standar deviations away (λ)	2.5	-
Learning rate (α)	0.005	-
Background threshold (T)	0.7	$K = 3$
Weight of created gaussians (ω_{init})	0.05	$K = 3$
Initial standard deviation (σ_{init})	30	Intensity pixel is between 0 and 255

The MoG parameters for our implementation were chosen according to [17] and are shown on table I, in practice the parameter ρ could be approximated dividing the learning rate, α, by the weight as suggested in the same work, but we chose the definition of [16] because it yields a better adaptation of the model when foreground objects appear in the first computed frames. For this reason, in the next equations ρ dissappears and α depends on the number of frames taken as a window.

In the next section we present the alternative equations used to update the MoG model. The complete approach is listed in Algorithm 1.

B. Updating the model

As mentioned above, we use the modified update equation proposed by [16]. First we choose a number L of frames to be sampled with a faster learning rate and the following expressions are used until L frames have been processed:

$$\omega_{k,n} = \omega_{k,n-1} + \frac{1}{n}(p(G_k|X_n) - \omega_{k,n-1}) \tag{8}$$

$$\mu_{k,n} = \mu_{k,n-1} + s(X_n - \mu_{k,n-1}) \tag{9}$$

$$\Sigma_{k,n} = \Sigma_{k,n-1} + s\big((X_n - \mu_{k,n-1})(X_n - \mu_{k,n-1})^T - \Sigma_{k,n-1}\big) \tag{10}$$

$$s = \frac{p(G_k|X_n)}{\sum_{i=1}^{n} p(G_k|X_i)} \tag{11}$$

where n is the frame number. When more than L frames have been processed we update the parameters as follows:

$$\omega_{k,n} = \omega_{k,n-1} + \frac{1}{L}(p(G_k|X_n) - \omega_{k,n-1}) \tag{12}$$

$$\mu_{k,n} = \mu_{k,n-1} + \frac{p(G_k|X_n)}{L}(X_n - \mu_{k,n-1}) \tag{13}$$

$$\Sigma_{k,n} = \Sigma_{k,n-1} + \frac{p(G_k|X_n)}{L}((X_n - \mu_{k,n})(X_n - \mu_{k,n})^T - \Sigma_{k,n-1}) \tag{14}$$

In the precedent equations the function p is defined as:

$$p(G_k|X_n) = \begin{cases} 1 & if \quad X_n \quad matches \quad Gaussian \quad G_k \\ 0 & otherwise \end{cases} \tag{15}$$

C. Foreground classification

Instead of using the traditional binary classification to decide if a pixel is part of the foreground, we calculate the Mahalanobis distance (MAH) between the pixel current value obtained from the input image and its correspondent background model. This way, the output of the classification process is a continuous value that represents how likely is that the pixel belongs to the foreground.

Having a given pixel represented by $I_{i,j} = (r,g,b)^T$, the MAH distance of $I_{i,j}$ to the k Gaussian in the mixture is computed as:

$$MAH(I_{ij},k) = \sqrt{(I_{ij} - \mu_k)^T \Sigma_k^{-1} (I_{ij} - \mu_k)} \tag{16}$$

where Σ_k and μ_k are the covariance matrix and the mean for the k-th gaussian, respectively.

III. Clustering-based object extraction

In this paper we use an object extraction approach which combines a Self-organizing Network inspired by the Growing Neural Gas [19] combined with a graph theoretic algorithm used to cut edges in the network's graph. The network is built from $M = W \times H$ nodes connected with undirected edges, arranged in a grid with H rows and W columns. This means that, with the exception of nodes located in the borders, every node i will be connected to four other nodes or neighbors ($neigh(i)$), individually denoted by $u(i)$, $d(i)$, $r(i)$ and $l(i)$ for up, down, right and left, respectively. Every node i has two associated variables: its mean value $\mu_i = (x_i, y_i)$ and an accumulator c_i. In a similar manner, for every edge connecting nodes i and j there will be an accumulator $e_{i,j}$. Besides W and H, the algorithm has two other parameters: $0 < \varepsilon_n < \varepsilon_w \leq 1$ which are the learning rates for node mean adaption.

The following subsections describe the steps that our algorithm performs *for every video frame*, using the grayscale image produced by Algorithm 1.

1) Initialization: The network is initialized by assigning values to all the μ_i node centers in order to form a regular grid. Also, the values of all the weights are set to zero:

$$\{c_i \leftarrow 0, e_{i,j} \leftarrow 0 \,\forall\, i,j \mid i \in [1,M], j \in neigh(i)\} \tag{17}$$

2) Learning: The learning stage takes as input a continuous valued bitmap I, where the pixel intensity reflects how likely is that it belongs to the foreground image(fig.1(e)). Pixels from the input image, are processed starting from the upper-left corner and then sweeping every row from left to right. For every pixel i, its coordinates p_i and value $I(p_i)$ are used to update the SON in four steps:

a. Find the node whose mean value is closest to p_i, referenced as winner (w_i). The search is restricted to a subset of nodes surrounding the previous winner w_{i-1} (*eg* the one corresponding to the previous processed pixel). This subset, that we call the search boundary, is represented by $sbound(i-1)$ (see fig. 2).

$$w_i = \underset{j \in sbound(i-1)}{\operatorname{argmin}} \|p - \mu_j - 1\| \tag{18}$$

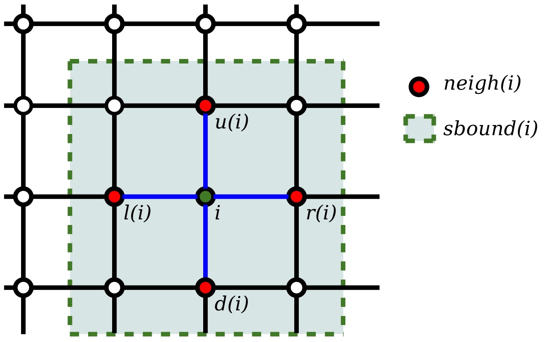


Fig. 2. Neighborhood and search boundary relative to node i

Algorithm 1. Summary of the background classification algorithm

```
Input: color frame RGB from video sequence ($I_{M\times N}$)
Output: gray frame ($IG_{M\times N}$) where every pixel $(i,j)$ represents distance from the background estimated in
        $MOG_{i,j}$
1.1  $MOG_{ij}$: Mixture of k gaussians for every pixel $(i,j)$;
1.2  The first r gaussians are background;
1.3  if Numberframe = 0 then
1.4      Initialize every $MOG_{ij}$ with the first frame
1.5  else
1.6      foreach $I_{ij}$ do
1.7          if $I_{ij}$ match any Gaussian in $MOG_{ij}$ then
1.8              update Gaussian matched according section II-B;
1.9          else
1.10             create new Gaussian and add to $MOG_{ij}$ replacing the last one;
1.11         end
1.12         update the other $k-1$ Gaussians according section II-B;
1.13         order $k$ Gaussians in $MOG_{ij}$;
1.14     end
1.15     foreach $I_{ij}$ do
1.16         $weight = weight_1 + ... + weight_r$ ;
1.17         $IG_{ij} = \frac{weight_1 * MAH(I_{ij},1) + ... + weight_r * MAH(I_{ij},r)}{weight}$;
1.18     end
1.19 end
```

b. Look in $neigh(w_i)$ for the second nearest node to p_i:

$$s_i = \underset{j \in neigh(w_i)}{\text{argmin}} \|p_i - \mu_j\| \quad (19)$$

c. Increment the accumulators e_{w_i,s_i} and c_{w_i} by the pixel value $I(p_i)$:

$$e_{w_i,s_i} \leftarrow e_{w_i,s_i} + I(p_i) \quad (20)$$

and

$$c_{w_i} \leftarrow c_{w_i} + I(p_i) \quad (21)$$

d. Adapt the mean of w_i and all his neighbors:

$$\mu_{w_i} \leftarrow \mu_{w_i} + \varepsilon_w \frac{I(p_i)}{c_{w_i}}(p_i - \mu_{w_i}) \quad (22)$$

$$\mu_j \leftarrow \mu_j + \varepsilon_n \frac{I(p_i)}{c_j}(p_i - \mu_j) \quad \forall j \in neigh(w_i) \quad (23)$$

3) Relabeling nodes: As a result of the learning step, the network adapts its form to represent the objects in the input. The last step of the algorithm, identifies individual objects by assigning a discrete value to every node in the SON, so that nodes having the same label belong to the same node. At the end the algorithm finds groups of nodes by merging nodes according to the weight of their common edges $e_{i,j}$. The idea is that a higher value of $e_{i,j}$ corresponds to a higher likelihood that nodes i and j belong to the same object. Under this assumption, it is possible to compute a maximum likelihood estimation of the probability, denoted by $P_{i,j}$, that two nodes "belong together" by using the Laplace law of succession, see [20] for a more detailed explanation. It is important to highlight the fact that the labels are just identifiers used to distinguish one region (*ie* object) from the other and that new labels are obtained by incrementing a counter.

4) Computing cluster representations: Having labeled the nodes, the probability that a pixel, given by its image coordinates p belongs to a cluster m may be represented as a mixture of gaussians, corresponding to individual nodes in the cluster:

$$P^*(p \mid m) = \sum_{i \in m} P_i \eta(p; \mu_i, S_i) \quad (24)$$

In order to compute the covariance matrices S_i, the points located halfway between i and its neighbors can be used:

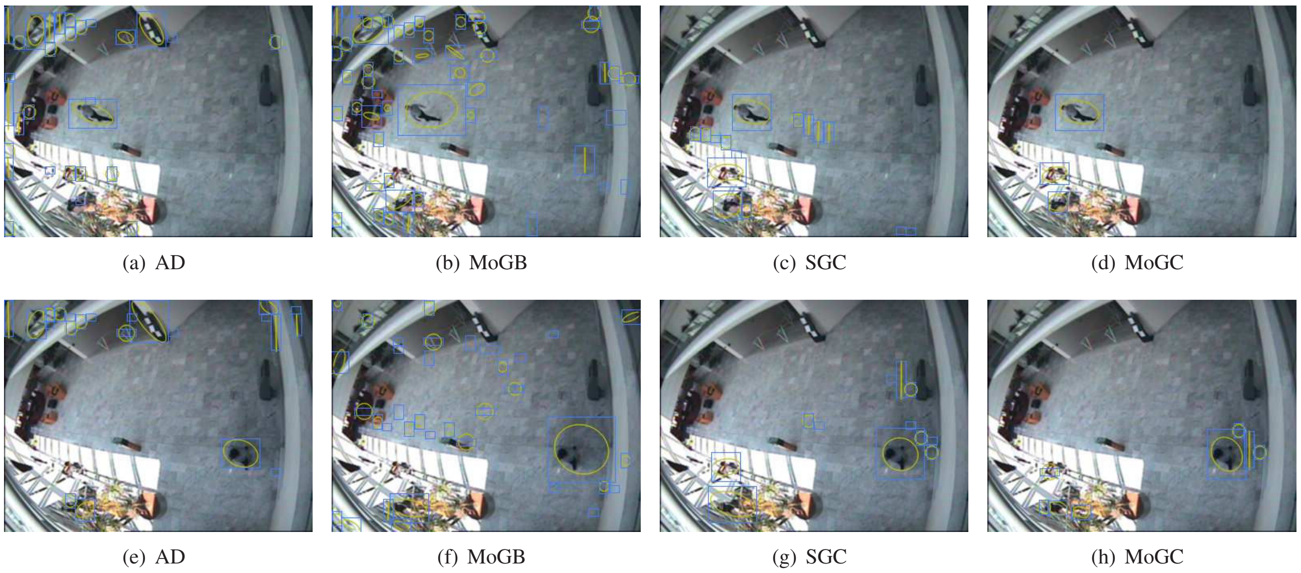
$$S_i = \sum_{j \in neigh(i)} \frac{P_j}{K} \begin{pmatrix} \left(\frac{x_j + x_i}{2}\right)^2 & \frac{(x_j+x_i)(y_j+y_j)}{4} \\ \frac{(x_j+x_i)(y_j+y_i)}{4} & \left(\frac{y_j+y_i}{2}\right)^2 \end{pmatrix} \quad (25)$$

Where $K = \sum_{j \in neigh(i)} P_j$ is a normalization constant.

In cases where the algorithm is required to produce interest regions it is often convenient to produce bounding boxes which are slightly larger than the contained object. We have computed the size of these regions using the difference between the maximum and the minimum mean values of the cluster nodes as they were *before learning* this may be regarded as finding the area bounded by nodes which have not been adapted.

A. Complexity Analysis

The local search, shown on the equations 18 and 19 gives a complexity of $O(N)$, while a global search would


(a) AD (b) MoGB (c) SGC (d) MoGC

(e) AD (f) MoGB (g) SGC (h) MoGC

Fig. 3. Detections over the foreground methods. Yellow ellipses representing the resulting detections with the respective bounding boxes in blue.

give $O(N_f M)$, where N corresponds to the total number of pixels in the image, N_f the number of foreground pixels and M the size of the SON. Noting that now it takes into account all the pixels in the image, instead of just the ones marked as being part of the foreground, and its complexity is independent of the SON grid size. This allows us to weight the pixels background/foreground contribution in a continuous fashion, instead of using a hard threshold, which makes the approach much more robust. The key idea for complexity independence on the SON size is to exploit the fact that the network is processed in a top-bottom, left-right sequence, and to limit the set of nodes in the SON which need to be compared with each pixel by looking in the neighborhood of the last processed pixel.

Thanks to the existence of efficient algorithms, the cost of labeling is linear with respect to the number of nodes in the SON, moreover, the computation of the cluster representation (*i.e.* gaussian parameters, mixture of gaussian parameters and bounding boxes) may be performed at the same time as labeling. For Algorithm 1, updating every MoG model for each pixel has a constant cost and, since the number of MoG models is fixed, its complexity is also linear with respect the number of pixels in the image. Thus, the algorithm's overall complexity is $O(N)$.

IV. Experimental Results

To evaluate the proposed method we are combining the clustering algorithm with 4 different background/foreground classification techniques:

(a) A binary bitmap obtained by thresholding the absolute difference (AD) between the intensity level of the current and previous video frames;

(b) Traditional MoG classification, with 3 Gaussians and binary output (MoGB);

(c) Background as a single Gaussian (SGC); foreground classification similar to the proposed approach.

(d) MoG with 3 Gaussians and the proposed foreground classification (MoGC).

The SON size and learning factors are the same for the three cases: 30x30 nodes, $\varepsilon_w = 0.1$ and $\varepsilon_n = 0.01$. The tests were conducted using the CAVIAR test case scenarios [21], which consist of a number of video sequences of people moving in the INRIA Lab's entry hall. The videos come with data files containing the ground truth of the sequences, which has been obtained by hand-labeling the images. Typical images of our detector running on one of these videos is shown in fig. 3.

For each frame processed the result of the extraction is compared with the ground truth and the following parameters are computed:

1) Detection ratio (η_{dt})

$$\eta_{dt} = \frac{\text{number of detections}}{\text{number of labeled objects}} \tag{26}$$

2) Matching ratio (η_{match})

$$\eta_{match} = \frac{\text{detection and ground truth matching area}}{\text{ground truth area}} \tag{27}$$

3) False positive ratio (η_{fp})

$$\eta_{fp} = \frac{\text{detected false positive area}}{\text{ground truth area}} \tag{28}$$

4) False negative ratio (η_{fn})

$$\eta_{fn} = \frac{\text{detected false negative area}}{\text{ground truth area}} \tag{29}$$

The mean values for these parameters, obtained from 1042 effective frames processed from the CAVIAR [21] 'Browse1' dataset is shown in table II.

We can see on table II that the proposed approach produces considerably better results than the other ones.

TABLE II
MEAN RESULTS FOR 1042 EFFECTIVE FRAMES PROCESSED FROM THE CAVIAR "BROWSE1" DATASET

input	$\bar{\eta}_{dt}$	$\bar{\eta}_{match}$	$\bar{\eta}_{fp}$	$\bar{\eta}_{fn}$
ideal	1	1	0	0
(a)	2.52	0.62	1.28	0.38
(b)	8.73	0.88	5.9	0.12
(c)	0.96	0.73	7.08	0.27
(d)	1.72	0.93	1.06	0.07

It is important to notice that traditional MoG, with a binary output, may in some cases produce better results if combined with post processing techniques to filter the noisy detections. On the other hand, this post processing step is unnecessary in our approach, since the noise tends to have lower significance during the foreground classification phase and the clustering algorithm can naturally filter out noisy input. In [20] we have shown as well that for a single-Gaussian background model, the application of the continuous input yielded slightly better detections than the thresholded foreground.

With respect to processing time, under the described experiment, the detection was performed at 15fps, running on Ubuntu 10.04 32-bit with an Intel^{TM} Core 2 Duo Processor P7450 at 2.13 GHz.

V. CONCLUSIONS AND FUTURE WORK

In this work we have discussed object extraction from continuous valued bitmaps emphasizing the advantages, but also the three big problems of cluster based algorithms (*ie* need to know the number of objects to be detected beforehand, sensibility to initialization and complexity) and extended previous work on a Self Organizing Network based on the Growing Neural Gas algorithm which solves the above mentioned problems and keeps the strong theoretical properties of clustering algorithms. Our extension permits makes the complexity of the algorithm independent of the size of the underlying SON, and eliminates the need of obtaining a binary image through a threshold.

We have explained the details of our algorithm, and shown how it may be used to find clusters and represent them using gaussians, mixtures of gaussians or bounding boxes.

Finally, we have discussed the experimental results we have obtained by comparing our approach to a ground truth consisting of hand-labeled data. Our results seem to confirm that our approach is fast, robust and general.

Is still important to notice the path to simplicity, reducing the number of parameters and post processing steps, that our solution promotes.

Future work includes continuing our experimental work, specially in a way to improve the metrics to compare the methods. We plan also extend the detection task to detection and tracking. Finally, we would like to explore the use of our SON to perform data fusion on a multicamera system.

REFERENCES

[1] D. Koller, J. Weber, T. Huang, J. Malik, G. Ogasawara, B. Rao, and S. Russell, "Towards robust automatic traffic scene analsis in real-time," in *Proceedings of the Int.Conf. on Pattern Recognition*, Israel, November 1994.

[2] M. Piccardi, "Background subtraction techniques: a review," in *Proceedings of the IEEE Int. Conf. on Systems, Man and Cybernetics*, The Hague, NL, October 2004, pp. 3099–3103.

[3] N. Friedman and S. Rusell, "Image segmentation in video sequences: A probabilistic approach," in *Proceedings of the 13th Conf. on Uncertainty in Artificial Intelligence*, Providence, USA, August 1997.

[4] C. Stauffer and E. L. Grimson, "Learning patterns of activity using real-time tracking," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 8, pp. 747–757, August 2000.

[5] R. Cucchiara, C. Grana, M. Piccardi, and A. Prati, "Detecting moving objects, ghosts, and shadows in video streams," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 25, no. 10, pp. 1337–1442, 2003.

[6] K. Suzuki, I. Horiba, and N. Sugie, "Fast connected-component labeling based on sequential local operations in the course of forward-raster scan followed by backward-raster scan," in *Proceedings of the 15th Int. Conf. on Pattern Recognition*, vol. 2, Barcelona, September 2000, pp. 434–437.

[7] F. Chang, C.-J. Chen, and C.-J. Lu, "A linear-time component-labeling algorithm using contour tracing technique," *Computer Vision and Image Understanding*, vol. 93, no. 2, pp. 206–220, 2004.

[8] L.-H. Chen and J.-R. Chen, "Object segmentation for video coding," in *Proc. of the 15th Int. Conf. on Pattern Recognition*, vol. 3, Barcelona, Spain, September 2000, pp. 383–386.

[9] F. Meyer and S. Beucher, "Morphological segmentation," *Journal of Visual Communication and Image Representation*, vol. 1, no. 1, pp. 21–46, 1990.

[10] A. Jain, M. Murty, and P. Flynn, "Data clustering: A review," *ACM Computing Surveys*, vol. 31, pp. 265–322, September 1999.

[11] J. MacQueen, "Some methods for classification and analysis of multivariate observations," in *Proceedings of the 5th Berkeley Symposium on Mathematical Statistics and Probability*, L. L. Cam and J. Neyman, Eds., vol. 1. University of California Press, 1967, pp. 281–297.

[12] N. Dempster, A.and Laird, , and D. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society*, vol. 9, no. 1, pp. 1–38, 1977, series B.

[13] D. Vasquez and T. Fraichard, "A novel self organizing network to perform fast moving object extraction from video streams," in *Proc. of the IEEE-RSJ Int. Conf. on Intelligent Robots and Systems*, Beijing (CN), October 2006. [Online]. Available: http://emotion.inrialpes.fr/bibemotion/2006/VF06

[14] D. Vasquez, F. Romanelli, T. Fraichard, and C. Laugier, "Fast object extraction from bayesian occupancy grids using self organizing networks," in *Proc. of the Int. Conf. on Control, Automation, Robotics and Vision (ICARCV)*, Singapore (SG), December 2006. [Online]. Available: http://emotion.inrialpes.fr/bibemotion/2006/VRFL06

[15] C. Stauffer and W. E. L. Grimson, "Adaptive background mixture models for real-time tracking," vol. 2, 1999, p. 252 Vol. 2. [Online]. Available: http://dx.doi.org/10.1109/CVPR.1999.784637

[16] P. Kaewtrakulpong and R. Bowden, "An improved adaptive background mixture model for real-time tracking with shadow detection," in *Proceedings of 2nd European Workshop on Advanced Video Based Surveillance Systems*, vol. 5308, 2001.

[17] W. P. Power and J. A. Schoonees, "Understanding background mixture models for foreground segmentation," 2002. [Online]. Available: http://www.is.irl.cri.nz/pubdoc/2002/JSPP2002.pdf

[18] Z. Tang and Z. Miao, "Fast background subtraction and shadow elimination using improved gaussian mixture model," in *Haptic, Audio and Visual Environments and Games, 2007. HAVE 2007. IEEE International Workshop on*, Oct. 2007, pp. 38–41.

[19] B. Fritzke, "A growing neural gas network learns topologies," *Advances in Neural Information Processing Systems*, 1995.

[20] T. Craesmeyer Bellardi, D. A. Vasquez Govea, and C. Laugier, "Frame rate object extraction from video sequences with self organizing networks and statistical background detection," 2008. [Online]. Available: http://hal.inria.fr/inria-00333071/en/

[21] "Datasets and videos of the european project caviar," http://homepages.inf.ed.ac.uk/rbf/CAVIAR/, July 2003.