Fast Learning Algorithm for Gaussian Models to Analyze Video Objects with Parameter Size

GuoQing Yin and Dietmar Bruckner

Institute of Computer Technology
Vienna University of Technology, Austria, Europe
{yin, bruckner}@ict.tuwien.ac.at

Abstract

Size of objects in scenes is an important parameter of video surveillance systems. From the analysis of object's size we can build an objects size model in scenes. The basic idea derives from automatic calibration to different perspectives.

To build such a model of object's size from the real time video data we utilize Gaussians and real-time fast learning algorithm from literature. The built model is used for real-time surveillance systems[1].

Keywords: Gaussian Models, Machine Learning, Parameter Analysis, Real-Time Applications.

1. Introduction

There are many resources about Gaussian models, fast learning algorithms and video object tracking: [1] is a unifying review of linear Gaussian model, [2] using kernel predictive linear Gaussian models for nonlinear stochastic dynamical systems, [3] proposed a fast learning algorithm for dynamic fuzzy neural networks based on extended radial basis function neural networks, [4] introduced an iterative learning algorithm for estimating coefficient vector in a given setting of statistical models with unknown hyper parameters, [5] gives a fast fixed-point learning algorithm for efficiently implementing maximization of the harmony function on Gaussian mixture with automated model selection, [6] propose an algorithm to track moving objects in video sequences, [7] present a Gaussian model-based approach for robust and automatic extraction of roads from very low-resolution satellite imagery, [8] demonstrate a real-time system for image registration and moving object detection, but to our knowledge there is no paper using Gaussian models and real-time fast learning algorithms for analyzing the video object's size for object's parameter analysis in surveillance systems.

1.1. Project SENSE background

This work is part of the SENSE project. The SENSE [9] project (Smart Embedded Network of Sensing Entities) is a European project about security for public spaces. The project has an airport as test environment.

Through unsupervised learning the network learns the "normality" in the environment and uses the "model of normality" to detect unusual behavior (e.g. people "lurking" in an area), situations (e.g. baggage left unattended) and to inform the user [10], [11]. This paper is about how to build a "normality" size model: use Gaussians and real-time fast learning algorithm to learn a model of the location-dependent objects size.

1.2. Basic Definition: Video objects, parameters, cluster, size.

The video modality of the SENSE system detects objects (O) for example: persons, group of persons, luggage, or unclassified objects. These are characterized by a bounding box. There are many parameters about these bounding box: object frame number (F, from 0 to 9; for each second) and time (T, second), ID of objects (L, from 1 to n; provided by a tracking mechanism), the object's position (X, Y) in pixel, and its width (W) and height (H), again in pixels.

At first we segment the video frame (480 x 640 pixels) in smaller pixels cluster (each cluster is 32 x 32 pixels) and therefore get 15 x 20 clusters in the frame. When objects move or stay in some of the clusters, the size of the object (O_{size}) will be gathered in these clusters, see Fig.1.

In Fig.1 there are two different kinds of objects: a blue one (thick lines) and a red one; the blue object moved from left to right with different Size (S_1, S_2, S_3). The red object moved with Size (S_1, S_2, S_3, S_4, here the size from blue and red objects have same index but different value). These clusters that covered by object will be get size parameter from the same object and with time flowing each cluster will be gather different size parameter. Such as Fig.1 the clusters covered by blue and red objects gathered two different size value (S_2 from blue and red objects).

[1] This work is partially funded by the European Commission under contract No. 033279.

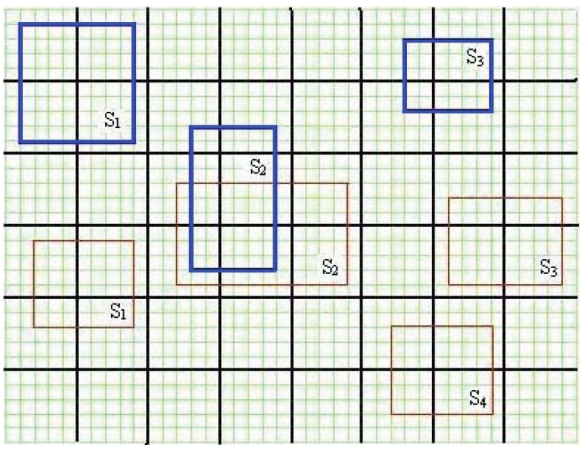

Fig.1. Depiction of data input. The small squares are pixel, the big grid represents the clusters, and the blue and red rectangles are various objects at different points in time with different size.

In a time interval each cluster ($C_{i,j}$, here i is cluster value in y direction, from 1 to 15; j is cluster value in x direction, from 1 to 20) collects a set of size {$S_{((i,j),\ m)}$, m≥0, m is the objects count}. A size model models the characteristic for this cluster ($C_{i,j}$). With Gaussian models and their real-time fast learning algorithm we can build the size model.

2. Gaussian Models and fast learning Algorithm

2.1. Gaussian model

The standard Gaussian function is:

$$\varphi_{\mu,\sigma 2}(S) = \frac{1}{\sigma\sqrt{2\pi}}\ e^{-\frac{(S-\mu)^2}{2\sigma^2}} \qquad (1)$$

μ being the expected size value (mean value of the cluster-dependent size), σ is the standard deviation of size for each cluster, S is the size value itself.

Before we apply the learning algorithm for clustering the size set, we have to initialize the parameters in each cluster, for example in cluster $C_{i,j}$: $\mu_{i,j}$ = S, which means the first size value which is covered in the cluster will be used as the initial value ($\mu_{i,j}$), $\sigma_{i,j}$ is set to 0 in one experiment and to $4*10^4$ in a second to show the differences of that choice (compare Figures 9 and 10 below). The maximum number of observations for lowering the learning rate is T_d and the current count is T'. Each new observation size value that gets into the set is S.

2.2. Fast learning Algorithm

1) Compute new size means

$$\mu_{i,j} = \mu_{i,j} \cdot (T'-1)/T' + S / T' \qquad (2)$$

2) Compute new size variances

$$\sigma_{i,j} = \sigma_{i,j} \cdot (T'-1)/T' + |\mu_{i,j} - S| / T' \qquad (3)$$

3) Keep the learning rate

$$\text{If } T' \geq T_d,\ T' = T_d \qquad (4)$$

4) With the new values go to step 1) (5)

This is a simple and real-time learning algorithm, but it delivers a reliable result and can be executed on embedded devices with limited computational resources.

3. Results

In a test video data the duration of the video is 779.8 seconds with 7798 frames (pictures, 10 frames each second, Fig. 2 is one example frame from the video data). In total there are 1456 different objects moving in the frames. In Fig. 2 the person with blue bounding box will be treated as an object. The size and position of the bounding box in the picture are the basic parameters.

Fig. 2. Camera view in a surveillance system.

In Fig.3 the object move to the right top location of the frame and the size of the object is smaller than the same object in Fig. 2, this is just because the location of the camera and the perspective.

Fig. 3. Camera view in a surveillance system. The same object as in Fig .2

If we learn all the 1456 different objects from all the 7798 frames in a time interval 779.8 seconds, use the Gaussian function (1) and fast learning algorithm (2) - (5) we can get the objects' size model for the Camera view in the time interval.

At first we will see how many objects gathered in each cluster and how much is the gathered objects size in each cluster.

Fig. 4 shows the gathered objects count in each cluster, at the left top and right bottom there are seldom objects but an "objects hill" running from left bottom to right top and the objects count increased in that direction then reduced at the right top location.

This is because, first: these location are the main path location; second: objects far away from the camera have a longer duration in these clusters, so there will be counted more often; third: new objects appear at the border will not be recognized at once, so at the border location there are generally always less objects counts.

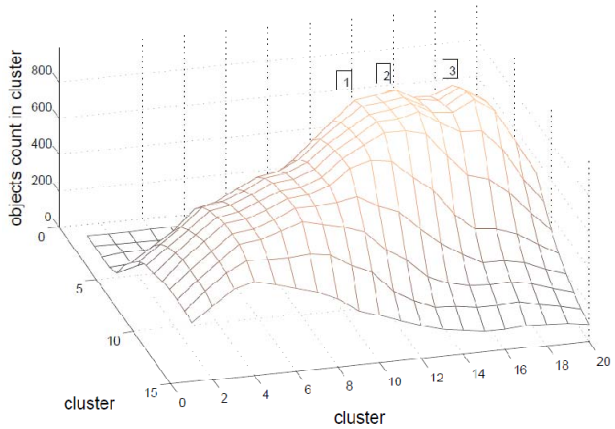

Fig. 4. Object's count model in clusters

If we rotate the objects model, we can observe an "objects path" from left bottom to right top in Fig.5. It corresponds to the video image: objects move in these areas and on the top side there is wall, without moving objects (but some wall areas were covered by moving objects) and on the right bottom location there are only seldom moving objects.

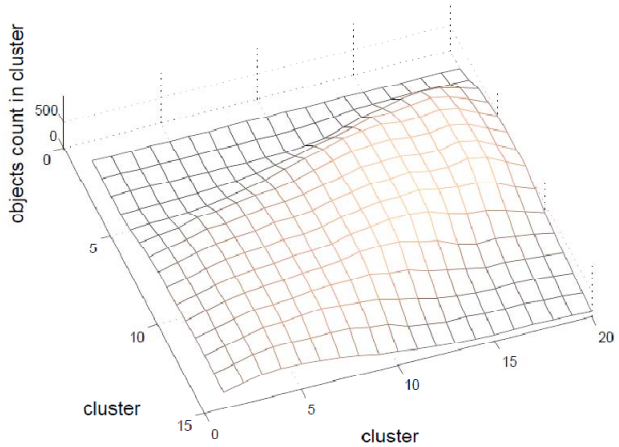

Fig. 5. Object's count model in clusters

Fig.6 is the gathered objects size summed up in each cluster. It has a similar shape as Fig. 4. This is because more objects have bigger sums in general, however not absolutely. It is related to the location of the clusters. For example there are three "peaks" depicted with location indices "1" - "3" in Fig.4, location 2 and 3 have nearly the same object count, location 1 has a smaller value, but in Fig.6 at the same locations the sum of sizesshows a different picture: location 3 has a smaller sum than location 1 and location 2. Further, if we compare these locations in Fig.8, we see that these "peaks" even disappear.

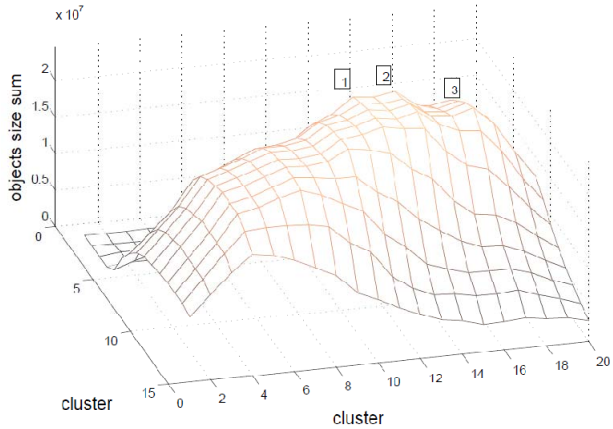

Fig. 6. Object's size model in clusters

If we rotate the objects size model we can get nearly the same shape just like Fig. 5. It is shown in Fig.7.

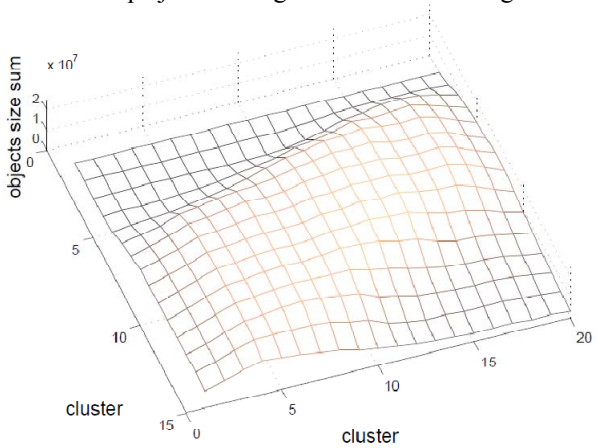

Fig. 7. Object's size model in clusters

Fig. 8 is the learned size model in each cluster: from left bottom to right top the size mean value reduced continuous. It corresponds to the Figures 2 and 3, the perspective[2]. But what about the left bottom location, right bottom location and the middle top location?

In the left bottom location the size mean value is smaller than the right neighbor areas. It is because in these areas the objects appearance was not completely, so these objects have a smaller size value than they should be. In the right bottom location the size value is bigger than its top and left side neighbor areas, it is because the camera mounted there, according perspective the objects there has a relative bigger size value. In the middle top location the objects size value is bigger than its right and bottom side and on the left side of the areas there are without objects occupied, the size mean value there are zeros. If we observe the Fig. 5 and Fig. 7 together, the answer is: there are seldom objects and these objects with relative bigger size value, according equations (2) to (5) the learning result is not reliable with seldom data. So there should be some bigger std value, the result just showed in Fig. 9: in these areas std value bigger than its

[2] This is the reason why the above mentioned "peaks" cannot be shown here: time does not play a role for that model.

neighbors. Extremely situation is if there just one object with a bigger size value occupied a cluster, the result should be the size mean value should be bigger but the size std value in the cluster will not changed (stay at the initial value). The situation happened in cluster (2, 10) and cluster (3, 9). In the two clusters there are bigger size values but the std value is just the initial value (Fig. 9 and Fig. 10).

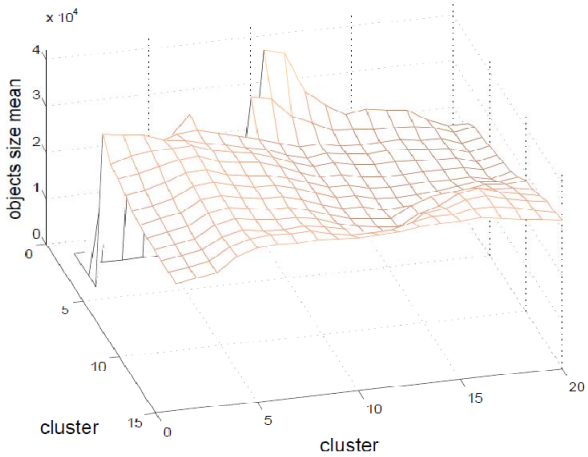

Fig. 8. Learned size model for objects in clusters

Fig. 9 shows the general size std value in the whole frame with initial std value $\sigma_{i,j} = 0$. For contrast, we set $\sigma_{i,j} = 4*10^4$ as initial std value and run the program with the same data again, will see how changed the std value after learning, Fig. 10 shows the result. The two Figs have nearly the same shape but Fig. 10 has generally bigger std value than Fig. 9, it is because these value reduced from a bigger initial value.

This is more convenient for further processing since it directly shows that the model is not well adapted to the data, whereas with the initialization with 0 we have to consider the data count to know how well the model potentially fits the data.

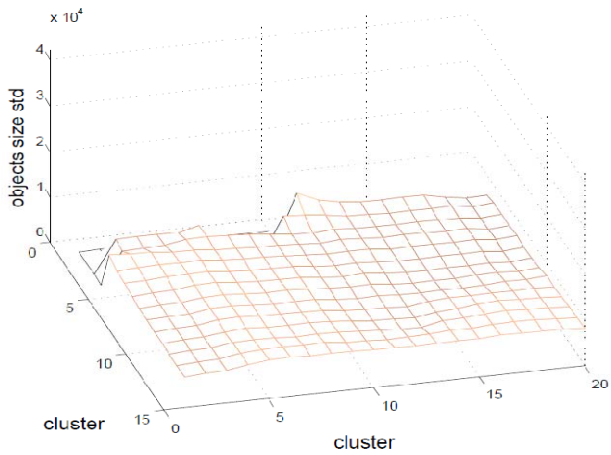

Fig. 9. Learned size std model for objects in clusters with initial $\sigma_{i,j}$ = 0

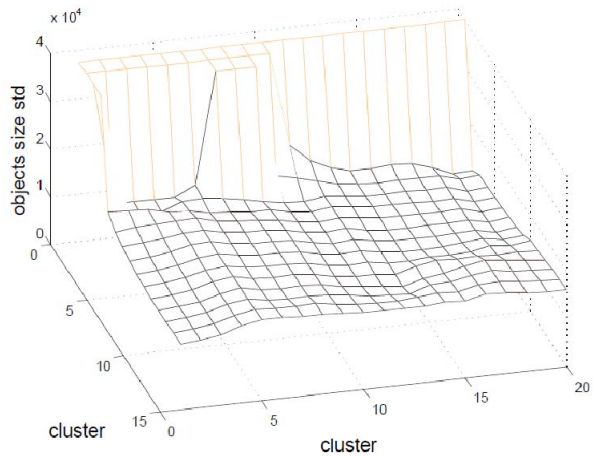

Fig. 10. Learned size std model for objects in clusters with initial $\sigma_{i,j}$ = 4*10^4

4. Conclusion

From the presented results we can conclude that Gaussians together with fast learning algorithms for embedded devices are a powerful tool for unsupervised learning in real-time automatic surveillance systems.

References

[1] Sam Roweis and Zoubin Ghahramani: "A Unifying Review of Linear Gaussian Model", in Neural Computation 11, 305-345 (1999).

[2] David Wingate and Satinder Singh: "Kernel Predictive Linear Gaussian Model for Nonlinear Stochastic Dynamical Systems" in Proceedings of the 23rd International Conference on Machine Learning, Pittsburgh, PA, 2006.

[3] Meng Joo Er and Shiqian Wu: " A fast learning algorithm for parsimonious fussy neueral system" in Fuzzy Sets and System 126 (2002) 337-351.

[4] Guang Deng: "Iterative Learning Algorithms for Linear Gaussian Observation Models", in IEEE TRANSACTIONS ON SIGNAL PROCESSING, VOL 52, NO. 8, AUGUST 2004.

[5] Jinwen Ma, Xuefeng He: "A fast fixed-point BYY harmony learning algorithm on Gaussian mixture with automated model selection", in Pattern Recognition Letter 29 (2008) 701-711.

[6] Yiwei Wang, John F. Doherty and Robert E. Van Dyck: " Moving Object Tracking in Video".

[7] Fatih Porikli: "Road Extraction by Point-wise Gaussian Model", in Mitsubishi Electric Research Laboratories, Murray Hill, 07974 USA.

[8] Yoav Rosenberg and Michael Werman: "Real-Time Object Tracking from a Moving Video Camera: A Software Approach on a PC", Institute of Computer Science, The Hebrew University of Jerusalem.

[9] http://www.sense-ist.org (SENSE project website).

[10] D. Bruckner, J. Kasbi, R. Velik, and W. Herzner: "High-level Hierarchical Semantic Processing Framework for Smart Networks", In: Proceedings of the 2008 IEEE Conference on Human System Interaction (HSI), S. 6, Krakow, 2008.

[11] B. Sallans, D. Bruckner, and G. Russ: "Statistical Detection of Alarm Conditions in Building Automation Systems", In: Proceedings of 2006 IEEE International Conference of Industrial Informatics INDIN´06, S. 6, Singapore, 2006.