Continuous Learning of Simple Visual Concepts using Incremental Kernel Density Estimation *

Danijel Skočaj, Matej Kristan, Aleš Leonardis

Faculty of Computer and Information Science, University of Ljubljana
Tržaška 25, SI-1001 Ljubljana, Slovenia
{danijel.skocaj,matej.kristan,ales.leonardis}@fri.uni-lj.si
http://vicos.fri.uni-lj.si/

Abstract. In this paper we propose a method for continuous learning of simple visual concepts. The method continuously associates words describing observed scenes with automatically extracted visual features. Since in our setting every sample is labelled with multiple concept labels, and there are no negative examples, reconstructive representations of the incoming data are used. The associated features are modelled with kernel density probability distribution estimates, which are built incrementally. The proposed approach is applied to the learning of object properties and spatial relations.

1 Introduction

An important characteristic of a system that operates in a real-life environment is the ability to expand its current knowledge. The system has to create and extend concepts by observing the environment – and has to do so continuously, in a life-long manner. An integral part of such a *continuous learning* system is a method for incremental updating of already learned representations, which has to fulfill several requirements: (i) the learning algorithm should be able to update the current representations (and create new ones if necessary), (ii) it should not require access to old (already processed) original data, (iii) the representations should be kept compact, and (iv) the computational effort needed for a single update should not depend on the amount of previously observed data.

In this paper we present an algorithm that addresses these requirements in the context of learning associations between low-level visual features and higher-level concepts. In particular, we address the problem of continuous learning of visual properties (such as colour or shape) and spatial relations (such as 'to the left of' or 'far away'). The main goal is to find *associations* between *words* describing these concepts and simple *visual features* extracted from images.

In our setting, the inputs in the learning process are partial descriptions of the scene; descriptions of the objects and relationships between them. An object can be labeled with several concept labels (e.g., object A can be 'yellow' and

* This research has been supported in part by the EU project CoSy (FP6-004250-IP) and Research program Computer Vision P2-0214 (RS).

'round' and positioned 'in the middle' of the image and 'to the left of' object B). These descriptions serve as learning examples. There are no negative examples (e.g., the information that 'yellow' is not 'red' is not provided). Therefore, the algorithm should build *reconstructive* representations without relying on *discriminative* information, which would discriminate between different classes (i.e., concepts) [1]. It can only exploit *consistency* of the feature values extracted from the objects labelled with the same concept and *specificity* of these values with respect to the rest of the feature values related to other concept labels.

The contribution of this paper is twofold. First, we present a method that is able to continuously process input feature vectors and learn corresponding associations based on the consistency and specificity criteria. Similar problems are commonly referred to as examples of the *symbol grounding problem* and have often been tackled by different researchers in different settings [2–6]. Several papers have also been published addressing online learning, particularly with regard to object recognition [7–9]. In our system, however, we utilize, in a unified framework, continuous online learning of qualitative object properties and spatial relations in a setting with no negative examples and when every sample is labelled with multiple concept labels, with a view toward using this as a basis for further learning and facilitating unlearning as well.

To facilitate this type of learning, our method models the values of features, that are associated with individual concepts, by estimating the probability density function that generated them. To that end, we employ kernel density estimates (KDE), which apply a mixture of kernels to approximate the underlying density. Since the models have to be estimated continuously from the arriving data, we construct a kernel for each incoming data and use it to update the corresponding distributions. This boils down to estimating the single parameter of the kernel – its bandwidth. We propose a method for estimating the bandwidth of each incoming kernel, which is the second contribution of this paper. A number of bandwidth selection methods have been proposed previously which aim to minimize the asymptotic-mean-integrated-squared-error (AMISE) between the unknown original distribution and its approximation based on a set of the observed samples (e.g., Wand and Jones [10]). However, these approaches are not directly applicable in incremental settings. To that end, various incremental approaches have been proposed which usually incorporate some constraints or prior knowledge of the relation between the consecutive samples [11, 12], like temporal coherence on the incoming data [13, 14]. Szewczyk [15] applies a Dirichlet process prior on components and applies a Gamma density prior to sample the bandwidth of the incoming data. A drawback of this approach, however, is that the parameters of the prior need to be specified for a given problem. Here, we propose an incremental bandwidth selection approach that does not assume any temporal coherence and does not require setting a large number of parameters.

The paper is organised as follows. First we introduce the main incremental learning algorithm. Then we explain the algorithm for incremental updating of KDE representations. In section 4 we then present the evaluation of the proposed methods. Finally, we summarize and outline some work in progress.

2 Main Incremental Learning Algorithm

The main task of the incremental algorithm is to *assign associations* between extracted visual features and the corresponding visual concepts. Since our system is based on positive examples only (we do not have negative examples for the concepts being learned), and each input instance can be labelled with several concept labels, the algorithm can not exploit discriminative information and can rely only on reconstructive representations of observed visual features. Each visual concept is associated with a visual feature that best models the corresponding images according to the *consistency* and *specificity* criteria. It must determine which of the automatically extracted visual features are *consistent* over all images representing the same visual concept and that are, at the same time, *specific* for that visual concept only. The learning algorithm thus selects from a set of one-dimensional features (e.g., median hue value, area of segmented region, coordinates of the object center, distance between two objects, etc.), the feature whose values are most consistent and specific over all images representing the same visual concept (e.g., all images of large objects, or circular objects, or pairs of objects far apart etc.). Note that this process should be performed incrementally, considering only the current image (or a very recent set of images) and learned representations – previously processed images cannot be re-analysed.

Therefore, at any given time, each concept is associated with one visual feature, i.e., with the representation built from previously observed values of this feature. A Kernel Density Estimate (KDE) is used to model the underlying distribution that *generated* these values. The KDE models, in our case Gaussian mixture models, are updated at every step by considering the current model and new samples using the algorithm presented in the next section. However, after new samples have been observed, it may turn out that some other feature would better fit the particular concept. The system enables such switching between different features by keeping simplified representations of all features. Assuming that the data that has to be modeled is coarsely normally distributed the proposed algorithm keeps updating the Gaussian representations of all features for every concept being learned[1]. These updates can be performed without loss of information. When at some point the algorithm determines that some other feature is to be associated with the particular concept, it starts building a new KDE, starting with one component obtained from the corresponding Gaussian model[2].

The only question that still remains is how to select the best feature. The algorithm fulfills the consistency and specificity criteria mentioned above by selecting the feature with a distribution that is most distant from the distributions of the corresponding feature values of all other concepts. The distances between two distributions are measured using the Hellinger distance [16]. Note that while the Hellinger distance can be calculated exactly for two Gaussians, there is no

[1] In practice, only the representations of a number of potentially interesting features could be maintained.

[2] In practice, several KDEs of the most interesting features could be maintained.

closed-form solution for Gaussian mixtures. We therefore apply the unscented-transform [16] to approximate the Hellinger distance between two mixtures. The proposed incremental learning algorithm is presented[3] in Algorithm 1.

Algorithm 1 : Main incremental learning algorithm

Input: $mC_{t-1}^{(i)}; i = 1 \dots nc_{t-1}$, f_t, c_t // current models and current input

Output: $mC_t^{(i)}, i = 1 \dots nc_t$ // updated models

```
// Update Gaussian models
for i ∈ c_t do // for all current concept labels
  if i ∉ mC_{t-1} then // previously not encountered concept
    nc_t = nc_{t-1} + 1 // increase the number of concepts
    init(mC_t^(i)) // initialize a new concept
  else // existing concept
    for j = 1...nf do // for all features
      mC_t^(i).G^(j) = updateG(mC_{t-1}^(i).G^(j), f_t^(j)) // update Gaussian models
    end for
  end if
end for
// Select best features
for i = 1...nc_t do // for all learned concepts
  for j = 1...nf do // for all features
    d_ij = Σ_{k=1}^{nc_t} d_Hel(pdf1, pdf2), where // calculate Hellinger distances
    pdf1 = { mC_t^(i).KDE , if mC_t^(i).B = j
           { mC_t^(i).G^(j) , if mC_t^(i).B ≠ j ,pdf2 = { mC_t^(k).KDE , if mC_t^(k).B = j
                                                        { mC_t^(k).G^(j) , if mC_t^(k).B ≠ j
  end for
  mC_t^(i).B = arg max_j d_ij // determine the best feature
end for
// Update KDE models
for i = 1...nc_t do // for all learned concepts
  if mC_t^(i).B ≠ mC_{t-1}^(i).B then // new best feature
    mC_t^(i).KDE = mC_t^(i).G^(mC_t^(i).B) // initialize KDE with corresp. Gaussian
  else // still the same best feature
    if i ∈ c_t then // current concept label
      mC_t^(i).KDE = updateKDE(mC_{t-1}^(i).KDE, f_t^(mC_t^(i).B)) using Algorithm 2
    else // not current concept label
      mC_t^(i).KDE = mC_{t-1}^(i).KDE // keep the old model
    end if
  end if
end for
```

[3] The following notation is used: $mC^{(i)}$ represent nc models of concepts being learned, where $mC^{(i)}.G^{(j)}$ is a Gaussian model of the j-th feature and $mC^{(i)}.KDE$ is a KDE model of the best feature ($mC^{(i)}.B$) for the i-th concept. f is a $nf-$dimensional training feature vector, c is a list of the corresponding concept labels. The subscripts $t-1$ and t indicate the time step.

The learned concepts are than used for analysis of new images. Feature vectors are extracted from the image and evaluated with the recognition algorithm that for every concept returns a belief, a value in the range from 0 to 1. This value is obtained by verifying how well the value of the feature, which is assigned to the concept, fits the particular KDE distribution, i.e., $p^{(i)} = \text{getBelief}(mC_t^{(i)}.KDE, f^{(mC_t^{(i)}.B)})$. This estimation can be done in different ways; our algorithm returns the integral over the pdf values that have lower probability than the current sample.

3 Incremental Estimation of KDE

In the previous section we described the main algorithm for incremental learning of simple visual concepts. Since these concepts are described using kernel density estimates (in our case mixtures of one-dimensional Gaussians), we present here an approach to incremental construction of these estimates.

Formally we define a *Gaussian-kernel-based* KDE as an M-component mixture of Gaussians

$$p(x) = \sum_{j=1}^{M} w_j K_{h_j}(x - x_j), \tag{1}$$

where w_j is the weight of the j-th component and $K_\sigma(x-\mu)$ is a Gaussian kernel

$$K_\sigma(z) = (2\pi\sigma^2)^{\frac{-1}{2}} exp(-\frac{1}{2}z^2/\sigma^2), \tag{2}$$

centered at mean μ with standard deviation σ; note that σ is also known as *the bandwidth* of the kernel. Suppose that we observe a set of n_t samples $\{x_i\}_{i=1:n_t}$ up to the current time-step t. We seek a kernel density estimate with kernels of equal bandwidths h_t

$$\hat{p}_t(x; h_t) = \frac{1}{n_t} \sum_{i=1}^{n_t} K_{h_t}(x - x_i), \tag{3}$$

which is as close as possible to the underlying distribution that generated the samples. A classical measure used to define closeness of the estimator $\hat{p}_t(x; h_t)$ to the underlying distribution $p(x)$ is the *mean integrated squared error* (MISE)

$$\text{MISE} = \text{E}[\hat{p}_t(x; h_t) - p(x)]^2. \tag{4}$$

Applying a Taylor expansion, assuming a large sample-set and noting that the kernels in $\hat{p}_t(x; h_t)$ are Gaussians ([10], p.19), we can write the *asymptotic* MISE (AMISE) between $\hat{p}_t(x; h_t)$ and $p_t(x)$ as

$$\text{AMISE} = \frac{1}{2\sqrt{\pi}}(h_t n_t)^{-1} + \frac{1}{4}h^4 R(p''(x)), \tag{5}$$

where $p''(x)$ is the second derivative of $p(x)$ and $R(p''(x)) = \int p''(x)^2 dx$. Minimizing AMISE w.r.t. bandwidth h_t gives AMISE-optimal bandwidth

$$h_{t\text{AMISE}} = [\frac{1}{2\sqrt{\pi}R(p''(x))n_t}]^{\frac{1}{5}}. \tag{6}$$

Note that (6) cannot be calculated exactly since it depends on the second derivative of $p(x)$, and $p(x)$ is exactly the unknown distribution we are trying to approximate. Several approaches to approximating $R(p''(x))$ have been proposed in the literature (see e.g. [10]), however these require access to *all* observed samples, which forgoes the possibility of incremental learning where we wish to discard previous samples and retain only compact representations of them. Thus in our setting we have to estimate the bandwidth of the kernel corresponding to the incoming sample and update the existing density estimate using that kernel; we propose a plug-in rule to achieve that.

Let x_t be the currently observed sample and let $\hat{p}_{t-1}(x)$ be an approximation to the underlying distribution $p(x)$ from the previous time-step. Note that in incremental learning we compress the estimated distributions once in a while to maintain low complexity [17]. Therefore, in general, the bandwidths of the kernels in $\hat{p}_{t-1}(x)$ may vary. The current estimate of the $p(x)$ is initialized using the distribution from the previous time-step $\hat{p}_t(x) \approx \hat{p}_{t-1}(x)$. The bandwidth $\hat{h}_t$ of the kernel $K_{\hat{h}_t}(x - x_t)$ corresponding to the current observed sample x_t is obtained by approximating the unknown distribution $p(x) \approx \hat{p}_t(x)$ and applying (6)

$$\hat{h}_t = c_{\text{scale}}[2\sqrt{\pi}R(\hat{p}_t''(x))n_t]^{-1/5}, \tag{7}$$

where c_{scale} is used to increase the bandwidth a little and thus avoid undersmoothing. In our experience, values of the scale parameter $c_{\text{scale}} \in [1, 1.5]$ in (7) give reasonable results and in all our experiments $c_{\text{scale}} = 1.3$ is used. The resulting kernel $K_{\hat{h}_t}(x - x_t)$ is then combined with $\hat{p}_{t-1}(x)$ into an improved estimate of the unknown distribution

$$\hat{p}_t(x) = (1 - \frac{1}{n_t})\hat{p}_{t-1}(x) + \frac{1}{n_t}K_{\hat{h}_t}(x - x_t). \tag{8}$$

Next, the improved estimate $\hat{p}_t(x)$ from (8) is plugged back in the equation (7) to re-approximate $\hat{h}_t$ and thus equations (7) and (8) are iterated until convergence; usually, five iterations suffice. The entire procedure is outlined in Algorithm 2.

Algorithm 2 : Incremental density approximation algorithm.

Input: $\hat{p}_{t-1}(x)$, x_t ... the current density approximation and the new sample
Output: $\hat{p}_t(x)$ ... the new approximation of density

1: Initialize the estimate of the current distribution $\hat{p}_t(x) \approx \hat{p}_{t-1}(x)$.
2: Estimate the bandwidth h_t of $K_{\hat{h}_t}(x - x_t)$ according to (7) using $\hat{p}_t(x)$.
3: Reestimate $\hat{p}_t(x)$ according to (8) using $K_{\hat{h}_t}(x - x_t)$.
4: Iterate steps 2 and 3 until convergence.
5: If the number of components in $\hat{p}_t(x)$ exceeds a threshold N_{comp}, compress $\hat{p}_t(x)$.

4 Experimental Results

In this section we present two sets of experiments, which were conducted to evaluate the proposed methods. The aim of the first experiment was to demonstrate the incremental bandwidth selection method proposed in Section 3. We generated 1000 samples from a 1D mixture of Gaussians and a uniform distribution (Figure 1(a)). These samples were then used one at a time to incrementally build the approximation to the original distribution using Algorithm 2. At each time-step two other KDE approximations were also built for reference: an optimal and a suboptimal. These were *batch approximations*, and were thus built by processing *all* samples observed up to the given time-step simultaneously. The optimal bandwidth was estimated using the solve-the-equation plug-in method [18], which is currently theoretically and empirically one of the most successful bandwidth selection methods. For the suboptimal bandwidth selection we have chosen Silverman's rule-of-thumb ([10], pg. 60), which was also used to initialize our algorithm from the first two samples.

The results are shown in Figure 1. The final KDEs, after observing all 1000 samples, estimated by the proposed incremental method and the two reference methods, are shown in Figure 1(a). It is clear that Silverman produced an undersmoothed approximation to the ground-truth distribution. The incrementally constructed KDE and the batch-estimated KDE using the solve-the-equation plug-in both visually agree well with the ground-truth. This can be further quantitatively verified from Figure 1(b) where we show how the integrated squared error (ISE) between the three approximations and the ground-truth was changing as new samples were observed. While initially errors are high for all three approximations they decrease as new samples arrive. As expected, the error of the KDE calculated using Silverman's rule remains high even after all 1000 samples have been observed. On the other hand, the error decreases faster for the KDE constructed by the proposed method and comes close to the error of the optimally selected bandwidth with increasing numbers of samples. Note that the optimal bandwidth was calculated using *all* samples observed up to a given step. On the other hand, the proposed incremental method produced similar bandwidths using only a low-dimensional *representation* of the observed samples (Fig.1(c)).

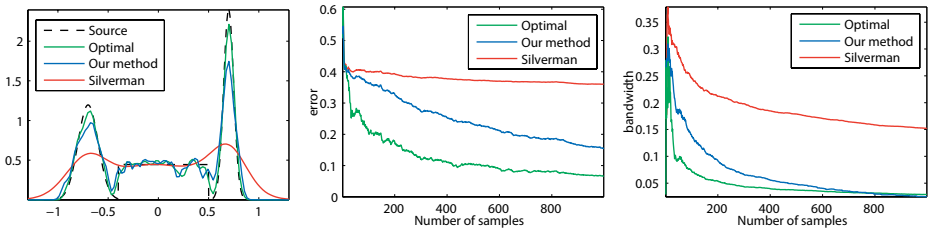


Fig. 1. Illustration of Incremental KDE algorithm: (a) final estimated distributions, (b) MISE with respect to the source distribution, (c) estimated bandwidth.

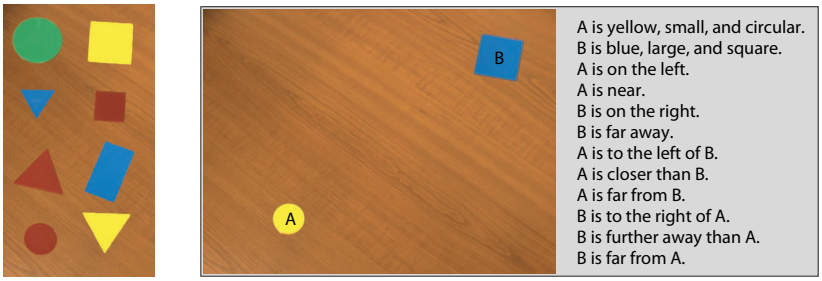
We evaluated the proposed method for learning visual concepts in a task that involved learning concepts of several object properties and spatial relations using simple objects of basic shapes and of different colours and sizes (Fig. 2(a)). This image domain is quite suitable for such analysis, since the object properties are very diverse and well defined. The input images contained pairs of objects that were put on a table in different configurations. For every object we considered ten visual concepts related to the object's colour, size and shape (red, green, blue, yellow; small, large; square, circular, triangular, and rectangular). Next, we also considered eleven different spatial relations – six binary relations between the two objects (with respect to the observer/camera): 'to the left of', 'to the right of', 'closer than', 'further away than', 'near to', 'far from', and five unary relations describing the position of the object in the scene: 'on the left', 'in the middle', 'on the right', 'near', and 'far away'. Fig. 2(b) depicts one image with the corresponding description, which gave two training samples (for object A and object B). From every training sample 11 features were extracted (three appearance features, three shape features, and five distance features), which were to be associated with 21 visual concepts being learned.

We randomly divided a set of 300 samples into two halves consisting of training and test samples respectively. Then we added training samples one by one and at each step updated the models of object properties and spatial relations using the proposed method. We evaluated the current models by trying to recognize all training samples and observing the achieved accuracy. To limit the complexity of the models, the current KDE was compressed whenever the number of components exceeded the number 20.

The experiment was repeated 20 times with different sequences of input samples. The average results are depicted in Fig. 3. From Fig. 3(a) it can be seen that the overall accuracy increases by adding new samples. The growth of the accuracy is very rapid at the beginning when new models of newly introduced concepts are being added, but still remains positive even after all models are formed due to refinement of the corresponding associations and representations. Fig. 3(b) plots the average number of Gaussian components in KDE distributions of all models. One can observe that after a while this number does not grow any more, thus the model size remains limited. The models do not improve by increasing their complexity, but rather due to refinement of the underlying representations. Fig. 4 depicts kernel density estimates of their best features for five concepts at the end of the learning process. These trained models can be used for automatic generation of scene descriptions as presented in Fig. 2(b).

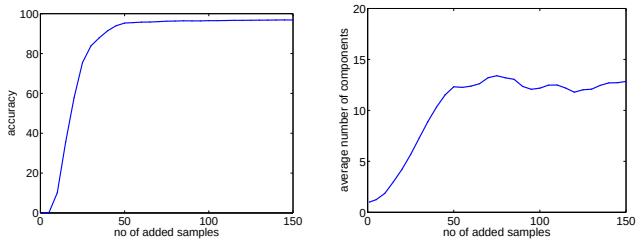
5 Conclusion

In this paper we proposed a method for continuous learning of simple visual concepts and applied it to learning object properties and spatial relations. The method keeps continuously establishing associations between automatically extracted visual features and words describing the observed scenes. The associated



(a) (b)

Fig. 2. (a) Input shapes. (b) Training / automatically generated scene description.



(a) (b)

Fig. 3. Experimental results: (a) accuracy. (b) average number of components.

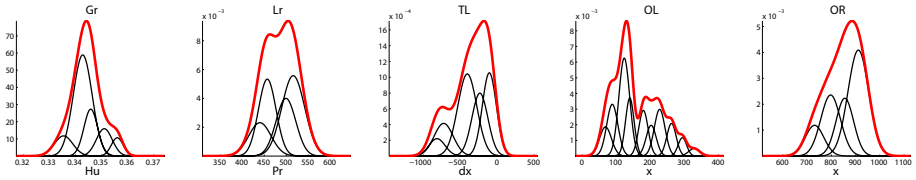


Fig. 4. Final models for 'green', 'large', 'to the left of', 'on the left', and 'on the right'.

features are modelled with kernel density probability distribution estimates using the proposed incremental KDE algorithm.

The proposed method fulfills four requirements for incremental learning presented in the introduction: (i) the learning algorithm is able to update the current representations and create new ones when new concepts occur, (ii) it does not require access to old data (it uses only their representations), (iii) the representations are being kept compact and do not grow any more once they reach an adequate complexity, and, consequently (iv) the computational effort needed for a single update does not depend on the amount of data observed sofar.

The work presented in this paper is a part of a larger framework for continuous learning of concepts that we have been developing. We will embed this algorithm in an interactive setting, where the concept labels (descriptions of scenes) will be obtained in an interactive dialogue with the tutor. In this way negative training examples will also be introduced, which will enable correction of erroneous updates. The proposed algorithm was tailored to support such operations.

In addition we also plan to extend the proposed incremental KDE algorithm to multiple dimensions and to advance the method to handle associations with several features, as well as to improve the feature selection method. Our ultimate goal is to develop a general, scalable and robust method for continuous learning of visual concepts.

References

1. Fidler, S., Skočaj, D., Leonardis, A.: Combining reconstructive and discriminative subspace methods for robust classification and regression by subsampling. IEEE Transactions on Pattern Analysis and Machine Intelligence **28** (2006) 337–350
2. Harnad, S.: The symbol grounding problem. Physica D: Nonlinear Phenomena **42** (1990) 335–346
3. Ardizzone, E., Chella, A., Frixione, M., Gaglio, S.: Integrating subsymbolic and symbolic processing in artificial vision. Journal of Intelligent Systems **1(4)** (1992) 273–308
4. Roy, D.K., Pentland, A.P.: Learning words from sights and sounds: a computational model. Cognitive Science **26** (2002) 113–146
5. Roy, D.K.: Learning visually-grounded words and syntax for a scene description task. Computer Speech and Language **16(3)** (2002) 353–385
6. Vogt, P.: The physical symbol grounding problem. Cognitive Systems Research **3** (2002) 429–457
7. Kirstein, S., Wersing, H., Körner, E.: Rapid online learning of objects in a biologically motivated recognition architecture. In: 27th DAGM. (2005) 301–308
8. Steels, L., Kaplan, F.: AIBO's first words. the social learning of language and meaning. Evolution of Communication **4** (2001) 3–32
9. Arsenio, A.: Developmental learning on a humanoid robot. In: IEEE International Joint Conference On Neural Networks. (2004) 3167–3172
10. Wand, M.P., Jones, M.C.: Kernel Smoothing. Chapman & Hall/CRC (1995)
11. Elgammal, A., Duraiswami, R., Harwood, D., Davis, L.: Background and foreground modeling using nonparametric kernel density estimation for visual surveillance. In: Proceedings of the IEEE. (2002) 1151– 1163
12. Han, B., Comaniciu, D., Davis, L.: Sequential density approximation through mode propagation: Applications to background modeling. In: Asian Conf. Computer Vision. (2004)
13. Arandjelovic, O., Cipolla, R.: Incremental learning of temporally-coherent gaussian mixture models. In: British Machine Vision Conference. (2005) 759–768
14. Song, M., Wang, H.: Highly efficient incremental estimation of gaussian mixture models for online data stream clustering. In: SPIE: Intelligent Computing: Theory and Applications. (2005) 174–183
15. Szewczyk, W.F.: Time-evolving adaptive mixtures. Technical report, National Security Agency (2005)
16. Julier, S., Uhlmann, J.: A general method for approximating nonlinear transformations of probability distributions. Technical report, Department of Engineering Science, University of Oxford (1996)
17. Leonardis, A., Bischof, H.: An efficient MDL-based construction of RBF networks. Neural Networks **11** (1998) 963 – 973
18. Jones, M.C., Marron, J.S., Sheather, S.J.: A brief survey of bandwidth selection for density estimation. J. Amer. Stat. Assoc. **91** (1996) 401–407