Background subtraction using competing models in the block-DCT domain

Mathieu Lamarre, James J. Clark
Centre For Intelligent Machines
McGill University, Montréal, QC, Canada, H3A 2A7
E-mail: {mlamarre,clark}@cim.mcgill.ca

Abstract

Many image analysis applications rely on background subtraction as a pre-processing step. Hence it should be efficient and robust. We present a background subtraction algorithm that uses multiple competing Hidden-Markov-Models (HMMs) over small neighbourhoods to maintain a locally valid background model in all situations. We use the DCT coefficients of JPEG encoded images directly to minimize computation and to use local information in a principled way. Region level processing is reduced to the minimum so that the extracted information that goes to higher level processing is unbiased.

1. Introduction

The goal of our work is to develop an efficient algorithm for segmenting a sequence of images into foreground regions, through suppression of the background. Simple solutions can be implemented given an initialization sequence free of foreground objects and a completely static background scene. However, in practice, such ideal situations are rarely encountered, and the background subtraction algorithm must work without clean training sequences and continuously update its reference model. Our approach is to use *background models*, in which we construct statistical predictions of which parts of the image correspond to background regions.

Much of the work on background subtraction comes from the field of video surveillance [2, 5]. Like previous researchers, we face the major problem of adaptation: that is, when do we adapt the background model, and how is this adaption done? If we continuously adapt the background model we get the degenerate case of frame differencing, while with no adaptation we are left with naive background subtraction. To avoid the known problems of these special cases a decision process must be included in the algorithm. Ideally, some form of temporal constraint should be used to take this decision. One intuitive approach is to specify an immobility period that an object must satisfy before being integrated in the background. This makes more sense then using some *ad hoc* update parameter. For example, the immobility time parameter can be specified according to the security level, specifying an infinite period for a high security location down to a few minutes for a low security room where people move objects frequently. Recently, many probabilistic background models have been proposed to solve the adaptation problem. Elgammal et al. [3] use a mixture of normal kernel functions to quickly learn the background distribution of scenes that contain moving objects (such as tree branches). Toyama et al. [8] use a Wiener filter to predict background values using the 50 previous intensity measurements.

We propose to use multiple models in our algorithm because they are needed in many application contexts. For example, indoor surveillance footage will necessarily require the surveillance of doors, as they are critical regions. Therefore, it would be appropriate to learn both the "opened" and "closed" background model of the doors. Another example is the light switch problem described by Toyama *et al.*[8], they used a frame level algorithm to switch between models, working at the pixel level would be an even stronger approach.

One very popular technique to find the best model given a sequence of observations are Hidden Markov Models (HMM) (see [7] for a tutorial), more precisely the Forward algorithm. Rittscher *et al.* [6] used HMMs for background subtraction in a traffic monitoring application, using one HMM per pixel. However, as our goal is to accept many models at the pixel level we need many HMMs per pixel. The amount of processing and memory space for such an implementation is currently not appropriate for a pre-processing algorithm. Therefore, as Rittscher *et al.*, we divide the image into small non-overlapping regions. We found that using images encoded with JPEG (see [9]) is a good way to reduce the amount of processing that our algorithm requires. Indeed, networked cameras that directly compress images in JPEG are now inexpensive and commonplace. JPEG compression operates by finding a linear

1051-4651/02 $17.00 (c) 2002 IEEE

combination of 64 orthogonal 2D basis functions that synthesize an 8×8 image block. The Discrete Cosine Transform (DCT) is used in JPEG to give such a decomposition. Each of the basis functions in the DCT contains one of the 64 unique two-dimensional frequencies found in 8×8 image blocks. By directly using the block encoded data of JPEG images, we reduce the number of HMMs we need by a factor of 64 and also access, with minimal computational cost, the average of each 8×8 neighborhood, as well as 63 coefficients that can be used to robustly discriminate background from foreground.

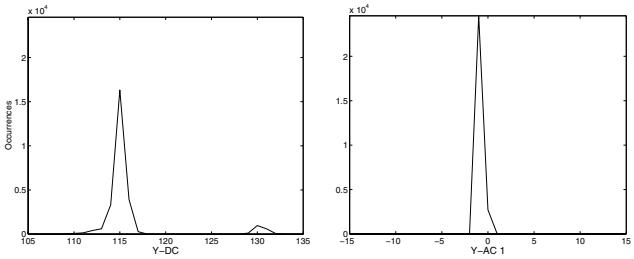

Figure 1. Histograms collected on a 3 hour period for a sampled DCT block.

2. Block-DCT domain observations

One major advantage of using DCT coefficients is that it's possible to reduce the amount of input data in a principled way. Indeed, reading only lower frequency DCT coefficients is a reasonable way to reduce the dimensionality of the observations. Intuitively, within a DCT block the DC (zero frequency) component codes the average luminance or chrominance, while the AC coefficients explain the depicted pattern. Lower frequency coefficients are less sensitive to noise or small changes in the images, while they preserve some edge information that is sufficient to detect occlusion edges at the boundaries of foreground object.

We obtain the DCT coefficients by entropy decoding of the JPEG source data. There is no need to de-quantize the values, as it does not bring any new information. The number of AC coefficients to use is somewhat system dependant. It depends on the targeted sensitivity and on the specific camera used. Inspection of the quantization table is a straightforward way to automatically determine how much or which coefficients to use (e.g. take the N less quantized coefficients). Our current implementation use color images, in YCrCb color space, where all three DC coefficients are included in the observations, but the chrominance AC coefficients are left out because they don't bring much more information.

3. Probabilistic background model

An HMM is built for each 8×8 block provided by the JPEG encoding. We use a very simple model with only two states: background and foreground. The transition matrix is fixed by hand based on the following hypothesis: blocks are more likely to stay in background state than to switch to the foreground state, and when they are in the foreground state there is no bias on the next state. We use a weak initialization prior which assumes that all blocks start out in the background state.

At any given time t the state of a 8×8 pixel block is estimated using an on-line Viterbi algorithm, which computes the most likely sequence of state transitions ending at state Q_t given the past history of observations. The last state Q_t for each block are combined to form the segmentation mask. This computation is only performed for the winning model for each block. The state estimate at each time step given by the other models are not needed because competition occurs at the sequence level.

The background probabilities are estimated empirically, by collecting histograms over a long period for a set of image blocks that we manually selected in the scene being viewed. Two samples of such histograms are shown in Fig.1. In this case, the data was collected on a three hour period for an indoor scene similar to the one shown in Fig.2 during an active period of the day, i.e. there was a lot of human activity in the room while recording the data. We show the histograms for the most interesting sampled block, and only for the luminance (Y) DC and first AC components. Over such a long period the background distribution should completely dominate, so it's assumed that the histograms correctly depict the true background models. The histograms show that the Y-DC component can be modelled by a mixture of Gaussians. The sampled block is located near a door, which explains the two modes of the distribution associated with the state of the door. On the other hand, the distribution of the first AC coefficient is modelled by a Gaussians of very small variance. This is caused by the quantization step in JPEG compression. Using very narrow (e.g. 5 units) uniform distributions is an acceptable approximation of this distribution that greatly reduce the complexity of the algorithm.

Let $\tau_{\text{max}}^{(i)}$ and $\tau_{\text{min}}^{(i)}$ be the maximum and minimum acceptable values for the i^{th} AC coefficient, these thresholds can be set such that the acceptable range is the same for all AC coefficients, i.e. $\{r_{\text{ac}} = \tau_{\text{max}}^{(i)} - \tau_{\text{min}}^{(i)} \forall (i)\}$. Furthermore, if we assume that the DC and AC coefficients are uncorrelated, we get the following background distribution:

$$p_b(\mathbf{z}) = \begin{cases} 0 & \text{if } \exists z_{ac}^{(i)} \notin [\tau_{\text{min}}^{(i)} \dots \tau_{\text{max}}^{(i)}] \\ \frac{G(\mathbf{z_{dc}})}{(r_{ac})^{n_{ac}}} & \text{otherwise} \end{cases} \quad (1)$$

where G is a Gaussian distribution:

$$G(\mathbf{z_{dc}}) = \frac{e^{-\frac{1}{2}(\mathbf{z_{dc}}-\mu)^T R^{-1}(\mathbf{z_{dc}}-\mu)}}{(2\pi)^{\frac{3}{2}}|R|^{\frac{1}{2}}} \quad (2)$$

And where $\mathbf{z_{dc}}$ is the 3-dimensional DC component vector, n_{ac} is the number of AC coefficients, and $1/(r_{ac})^{n_{ac}}$ is the probability of an observation given that all AC coefficients are within range. This normalizes $p_b(\mathbf{z})$ such that its integral over all $\mathbf{z}$ is equal to 1.

We assume a uniform distribution for the foreground model:

$$p_f(\mathbf{z}) = \frac{1}{256^3}\prod_{i=1}^{i=n_{ac}}\frac{1}{d_{ac}(i)} \quad (3)$$

where $d_{ac}(i)$ is the effective dimension of the i^{th} AC coefficient, which can be computed from the quantization table.

4. Background model adaptation

To be successful, a background subtraction algorithm must consider both smooth and sudden transitions in illumination. To cover both problems we use two separate approaches. Smooth transitions of the DC coefficients are integrated into the background model with a steady-state Kalman filter [4] and sudden changes are detected and analyzed by the competing HMMs. The AC coefficient distributions are not updated because of their very low variance.

4.1. Competing models to detect sudden transitions

At regular intervals, the best model is elected. Given a sequence of observations O and many models λ_i we can compute the probability $p(O|\lambda_i)$ for each HMM using the Forward algorithm [7], the model with the highest probability is elected. We use an on-line version of the algorithm to achieve a constant processing load. When a challenger wins against the current best model both the Viterbi and Forward algorithms are initialized with a new observation and the prior that the pixel block is currently in background state.

4.2. Adaptation to smooth transitions

A simple approach to track small illumination changes without integrating outliers is to compute some range of validity using the background distribution variance and then update the mean of the distribution for observations that fall within that range. However, note that our observations are averaged on 8×8 blocks. Therefore, all transitions caused by objects moving through some block will be smooth enough to seriously corrupt the background distribution. To avoid updating the background pdf. with observations that are near sudden transitions, we inspect the state sequence and update the background only if there are no state transitions in the near past or future. This is possible when using HMMs because the lattice structure keeps the history of state transitions. In practice, we keep a very limited record of state transitions.

Once a valid observation is found we update the mean of the 3D Gaussian part of the model. We use a steady-state Kalman filter. The dynamics of the distribution are modelled as constant. The measurement error covariance is computed during the training process. The process variance is set to the identity matrix.

5. Bootstrapping the training process

Having multiple competing HMMs is crucial for our algorithm to work. We need a module that will automatically detect stable changes to the scene and build new models for the affected blocks. We consider the detection of persistent changes and the training process as two different functions.

Detection of stable changes is the only module within the algorithm that operates at the region level. The idea is to look for large regions that do not change for a given period of time. This is implemented by adding an alternate HMM model for each block. This model adapts rapidly to any changes to the DC coefficients. We use a gain matrix $K = 0.5I$. The AC coefficients are left out of this model. If the main model for a block is in the foreground state the alternate model is used to see if the foreground object stays stable. It runs as follows:

- Flag blocks for which $Q_t^{main} = F$ and $Q_t^{alt} = B$;
- Compute the mask of the flagged blocks;
- Remove unconnected blocks from the mask;
- If the mask has not changed for X seconds train new models for all flagged blocks.

The training process is itself straightforward. A fixed number of sample observation vectors are recorded. The mean and covariance matrix are then computed for the three DC coefficients inside the vectors. The range for each AC coefficient is computed as the samples are obtained. Singular value decomposition is performed on the covariance matrix. We inspect singular values and invert the covariance matrix. Finally, the steady state Kalman gain matrix is computed.

For the initial training, all the observations are used; but when a new model is being trained we use a prior obtained from the persistent change detection algorithm. This prior is a 3D Gaussian distribution of the DC coefficients of the new background. We use it to coarsely filter the incoming observations taking only observations that fall within 5 standard deviations of the prior estimate. If for some reason the fixed number of training samples cannot be reached because

there is too much variance, the training is interrupted, the previous model is selected and the Viterbi and Forward algorithm are initialized with a foreground prior. These rules adequately implement the learning process even for training sequences with moving objects.

The number of models is limited. We use a *Least Recently Used* algorithm to discard old models as new ones are being learned. When a model wins a competition or is created we put it on top of the list. When there is no more space for a new model we replace the LRU model. In our experiment we used 3 models per block, which is enough in most contexts.

6. Results

We tested our system in many indoor configurations: computer labs with much activity, corridors, and rooms with large window. We used the Axis 200 network camera [1] to provide the JPEG encoded image sequenced. The algorithm behaved robustly, giving good foreground segmentation over full day periods.

Representative results are given in figure 2, which provides a set of snapshots of the background modelling process for the case of an opening-closing door. The model learns to switch quickly between the "closed" and "opened" door background models.

Figure 2. Snapshots of the background modelling process run on a scene containing an opening and closing door. All images come from a single sequence.

7. Conclusions

We showed that we can use the DCT coefficients of precoded JPEG images to inexpensively obtain good observations on 8×8 blocks. We described a background subtraction algorithm that uses competing HMMs to represent the background model accurately over time. This algorithm includes two levels of background adaptation. The small illumination variations are tracked with a steady state Kalman filter while model switching adapts to sudden change between two learned background distributions. A learning algorithm automatically creates and updates model over time to get reliable performance in all situations. However, the algorithm has many parameters that need to be tuned to fit the application context. This approach is particularly well suited for indoor automated video surveillance, due to the nature of the scene changes encountered in such applications.

References

[1] Axis Communications AB, Lund, Sweden. *Technical Guide to Axis Network Video Solutions*.

[2] R. Collins, A. Lipton, T. Kanade, H. Fujiyoshi, D. Duggins, Y. Tsin, D. Tolliver, N. Enomoto, and O. Hasegawa. A system for video surveillance and monitoring: Vsam final report. Technical Report TR CMU-RI-TR-00-12, Robotics Institute CMU, May 2000.

[3] A. Elgammal, D. Harwood, and L. S. Davis. Non-parametric model for background subtraction. In *ECCV*, pages 751–767, 2000.

[4] A. Gelb, editor. *Applied Optimal Estimation*. MIT Press, Cambridge, MA, 1974.

[5] I. Haritaoglu, D. Harwood, and L. Davis. W4: Who? when? where? what? a real time system for detecting and tracking people. In *Inter. Conf. on Automatic Face and Gesture Recognition*, pages 222–227, Apr. 1998.

[6] S. J. J. Rittscher, J. Kato and A. Blake. A probabilistic background model for tracking. In *ECCV*, 2000.

[7] L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proc. of the IEEE*, 77(2):257–286, 1989.

[8] K. Toyama, J. Krumm, B. Brumitt, and B. Meyers. Wallflower: Principles and practice of background maintenance. In *ICCV*, 1999.

[9] G. Wallace. The jpeg still picture compression standard. *IEEE Trans. Cons. Elec.*, 38:18–34, Feb. 1992.