Detecting Suspicious Behavior in Surveillance Images

Daniel Barbará
Carlotta Domeniconi
Zoran Durić
George Mason University
Computer Science Department
Fairfax, VA 22030

Maurizio Filippone*
Richard Mansfield
Edgard Lawson
George Mason University
Computer Science Department
Fairfax, VA 22030

E-mail: dbarbara,cdomenic,zduric,rmansfi2,wlawson@gmu.edu

Abstract

We introduce a novel technique to detect anomalies in images. The notion of normalcy is given by a baseline of images, under the assumption that the majority of such images is normal. The key of our approach is a featureless probabilistic representation of images, based on the length of the codeword necessary to represent each image. Such codeword's lengths are then used for anomaly detection based on statistical testing. Our techniques were tested on synthetic and real data sets. The results show that our approach can achieve high true positive and low false positive rates.

1 Introduction

Surveillance applications are becoming commonplace. However, having to browse through a large number of frames in search of worthy images is a labor-intensive and error-prone task. A technique that could highlight which images are worth a deeper and closer look would be of outmost importance. Additionally, such a technique could highlight crucial information that is otherwise missed. Although video and image surveillance have received a lot of attention in the computer vision community, the issue of detecting anomalous images is largely ignored.

In this paper, we introduce the design of a novel technique that achieves this goal. The method is capable of flagging anomalous images, where anomalies are defined as situations that are not encountered in a baseline of images used to train the technique. Our implementation uses standard techniques for foreground and background object detection in images (which can be substituted for other methods), but its novelty resides in the way it compares images to find those that contain anomalies.

A common approach for anomaly detection is to represent entities as vectors in a given feature space, and compute distances therein. However, defining a meaningful set of features, and a proper distance measure between images is a challenging problem. Inevitably, the selected features will bias the kind of anomalies we are able to detect. To address this issue, we introduce an anomaly detection approach based on a *featureless* probabilistic representation of images. The underlying principle is rooted in information theory and code design. Optimal codes are built so that the most common words receive shorter descriptions. We apply a similar concept to images, where words are replaced by the detected objects. The collection of objects in an image (along with their probabilities) contributes to the length of the *codeword* used to represent that image. Such codewords' lengths become the key information for anomaly detection based on statistical testing.

Our approach to the problem has many desirable properties. The methodology does not employ a model to define normalcy (or abnormal behavior). This characteristic is highly desirable, since techniques that utilize models (either learned through data or manually built) tend to be brittle and produce a large number of false positives. We assume that baseline images, against which anomalies will be flagged, are available. The size of the baseline sample is related to the confidence level at which one wishes to operate. As a rule of thumb, if the sample has N images, the maximum confidence at which we can operate is $1 - \frac{1}{N}$ (for instance, with 20 images one can utilize a confidence of 95%). The baseline set itself provides the definition of normality, as we shall explain later. Furthermore, our approach does not require the recognition of specific objects (e.g., cars, people).

*CS Dept., University of Sheffield, UK, filippone@dcs.shef.ac.uk

Rather, we represent objects as geometric shapes that can be compared without attempting to recognize them. This assumption increases the robustness of the overall technique, as object recognition is a difficult task prone to misclassifications. As a result, our methodology can be applied independently of the image context, of which we make no assumptions.

The paper is organized as follows. In Section 2, we describe our anomaly detection approach, how to obtain a featureless probabilistic description of images, and the image processing techniques used for object detection. In Section 3, we describe how the individual components are integrated together in the overall approach. Specifically, we describe two different methods. Experiments and results are discussed in Section 4, and conclusions are given in Section 5.

2 Technical Description

2.1 Anomaly Detection

In anomaly detection, the goal is to find entities that are *different* from most other entities. These entities, called outliers, are defined by Hawkins [5] as "an observation that deviates so much from other observations that arouse the suspicion that it was generated by another mechanism." This definition leaves the concept of deviation unexplained, and therefore is general enough to apply to many different situations. In our application, we are looking for images that can be identified as outliers. Using the definition above, that means images that capture events generated by a "mechanism" other than the one that generates the images we consider normal.

Many anomaly detection algorithms (see for example [7, 8]) find outliers by measuring the distance between entities, where the entities are vectors of attributes. However, defining distances between two images is a difficult problem. Some researchers have attempted to solve it by describing an image as a vector of features (e.g., [3]). This approach detects anomalies based on the extracted features: for instance, an image with predominantly blue color (where color is one of the features) would be an outlier among images that do not have a lot of blue in them. Clearly, these are not the kinds of anomalies we are interested in. Instead, we utilize an anomaly detection technique we have recently developed presented in [1]. The approach, called StrOUD (for Strangeness-based Outlier Detection algorithm) compares the entity that we wish to diagnose to a sample of entities, and uses statistical testing to decide if the entity in question is an outlier. The comparison is made by using a measure, called *strangeness* (α), that represents how "strange" the entity is. Strangeness is defined as a function, and can take many forms, a characteristic that makes the method adaptable to many different scenarios.

Once a strangeness function has been chosen, the next step is to compute the strangeness for a sample of data. This sample of data, called baseline, will "represent" what normalcy is. Notice that since anomalies are strange, it is reasonable to assume that most of the data in that sample will be normal, and therefore the strangeness values of that data will approximate what the normal distribution of strangeness would look like. (If an abnormal data observation would occur too often, it would cease to be an anomaly.) Following the paradigm of transduction [11, 4], the computation of the α for each of the observations in the baseline data is done by taking out that observation from the sample, and computing the function that defines its strangeness. This mechanism is called transduction in the literature [11, 4]. After this step, a sample distribution of α values becomes available and we are ready to diagnose new entities.

To diagnose an entity using the available distribution of α values, we make use of a test of randomness. The most widely applied test is called hypothesis testing and it involves the calculation of the p-value. A p-value is the smallest level at which we can reject the null hypothesis. Informally, is the measure of evidence against the hypothesis: the smaller the p-value the greater the evidence against the null hypothesis. In our case, the p-value is calculated as follows. If the entity we are trying to diagnose has strangeness α_n and the distribution of α values in the baseline data is $\alpha_1, \alpha_2, \ldots, \alpha_{n-1}$ the p-value can be calculated as:

$$p\text{-value} = \frac{\#\{i = 1, \ldots, n, \alpha_i \geq \alpha_n\}}{n} \qquad (1)$$

The symbol $\#$ in Eq. 1 indicates the cardinality of the set of α values that are greater than or equal to the α value of the test entity. In other words, the p-value is calculated as the fraction of entities that exhibit strangeness at least as large as the strangeness of the test entity.

The null hypothesis is H_0*: The entity is not an outlier*. For the application we are describing, it translates as "The image is drawn from the same distribution that 'generated' the images we use as a baseline." With the help of a confidence level δ we can use the p-value to accept or reject this hypothesis. If the p-value is less than $1 - \delta$, the null hypothesis is rejected. Otherwise, the alternative hypothesis H_1*: The entity is an outlier* is accepted.

For the problem under consideration, the entities are images, and we need to define a suitable function for α. As mentioned before, functions that measure the distance between two images typically do not capture meaningful anomalies. To avoid this problem, we rely on the length of the description of the image in probabilistic terms to measure its strangeness. The principle is deeply rooted in information theory and code design [2]. Optimal codes are built

in such a way that the most common words receive shorter descriptions. A similar concept can be applied to images, replacing words with the objects in the image. In the next section we explain how to achieve this.

2.2 Probabilistic Description of Images

In this section we explain our choice of strangeness function and its relationship with information theory and code design. An image can be considered to be composed of a series of geometrical objects (later on we will explain how we recover these objects in practice). One can argue that, over all the possible normal images of a given area, each object follows a spatial distribution (which is unknown). This spatial distribution indicates the probability of the object being located at a specific location (x, y) within the image. It is easy to represent the whole set of possible objects as a dictionary of "words." A particular arrangement of objects constitutes an image and it is akin to a code composed of symbols in a given dictionary.

In the field of Information Theory [2], it is well known that data compression can be achieved by the assignment of short descriptions to the most frequent symbols in the vocabulary. In the Morse code, for instance, the most frequent symbol is represented by a single dot. One can prove that a lower bound for optimal codes assigns a codeword of length $-\log p_i$ to the symbol i if its probability of occurrence is p_i.

Similarly, we could describe each image as a codeword composed of symbols, one for each object in the image. The length assigned to a symbol representing an object in a particular position would be a function $(-\log p_i)$ of the likelihood of the object being in that location. This likelihood is given by the (unknown) spatial probability density of the object. While the spatial density of all the objects over all the possible normal images is unknown, we can estimate it from the baseline sample of images. The problem becomes a *density estimation* problem [10].

Equipped with estimates of the spatial density functions for the objects we observe in an image, we can compute the length of the image by adding the contributions $-\log p_i$ for each of the objects. The quantity $(-\sum_i \log p_i)$, which we call the size of the image codeword, is what we use as strangeness measure for the image. Notice that, if an object is located in an unlikely position in the image, its contribution to the codeword size is going to be large, potentially making the image abnormal. This is precisely what we are trying to achieve: images with objects in unlikely positions, or with objects that have rarely or never been seen in the baseline collection, are going to have large strangeness, and most likely will be diagnosed as outliers.

There are many methods available for density estimation, from parametric methods that assume that the data is drawn from a known parametric family of distributions (e.g., Gaussian), to non-parametric methods such as histograms. We will later describe the density estimation methods we used.

2.3 Image processing and object detection

2.3.1 Background Subtraction

We assume that the camera observes a scene in which the background changes slowly relative to the motion of the people and the objects in the scene. The resulting pictures are registered. This allows to utilize *background subtraction* to identify foreground and background objects. These techniques have been successfully employed before (see [9]). (Alternatively, other techniques to obtain object descriptions from images, such as image segmentation, can be used.) The result of this process is a a series of objects, each characterized as a set of connected edges. For each pixel in the edges of the object, we use a standard computer vision ([9]) technique to compute the **color image gradient** of the pixel. Without going into too many details, we use finite differences to compute partial derivatives in the x and y directions for the red, green, and blue channels. We form a Jacobian with the derivatives $J = (r_x, r_y; g_x, g_y; b_x, b_y)$ (r_x, r_y is the first row; g_x, g_y is the second row and b_x, b_y is the third row). We form $S = J'J$ (transposed J times J). The larger eigenvalue of S is the edge strength squared and the corresponding eigenvector is the edge orientation. The exact direction (+/-) is determined by comparing the eigenvector and the rows of J. The color gradient is the edge orientation vector times the edge strength (square root of the larger eigenvalue).

With the color gradient values for each pixel computed, we can sort the pixels and select the n pixels with largest gradient. The n points are then placed in a histogram composed of thirty-six bins. Each pixel belonging to an edge contributes to one of the bins, according to the edge orientation. When the bin index is determined the bin count is incremented.

This histogram can be converted into a *probability density function* (*pdf*) by normalizing the sum of the histogram counts to unity. Two pdfs p_1 and p_2 can be compared using *Kullback-Leibler divergence* given by:

$$D(p_1|p_2) = \sum_{i=1}^{36} p_1(i) \log \frac{p_1(i)}{p_2(i)}$$

Therefore, using KL over histograms of two different objects, one can compute the distance between the objects and therefore their similarity.

3 The overall approach

We have implemented two alternative methods of anomaly detection. The difference between the methods resides on the way the density estimation is performed. In method 1 (M1), the density estimation is done over the entire image. For that, objects are clustered in similar groups and a probability density function is estimated per cluster. This estimation is performed off-line, using the baseline data, before images can be diagnosed. Given an object in a new image, its cluster membership is determined and the corresponding density function is used to compute its probability of being seen in the location where it is found. In method 2 (M2), the image is divided by using a grid. When an object is found in a given block of the grid, the probability of the object being in the observed position is computed on the spot, using a non-parametric model that utilizes distances to objects that have been seen in the same block in the baseline data. A more detailed explanation of the methods follows.

3.1 Method 1 (M1)

The first step consists in clustering the objects extracted from the training set (i.e., baseline data). Clustering is performed by means of a three level divisive clustering that uses a different distance function at each level. The three distance functions are based on: (1) dimension of the object (pixel count); (2) matching of the orientation and the size of the bounding box; and (3) edge gradient. At each level, each cluster obtained in the previous level is split (divisive clustering) by re-clustering its objects using an agglomerative approach (based on the Ward method [6]) according to the distance function of that level.

First, objects are clustered using the pixel count. This allows early separation of objects having different sizes. At the second level, objects are grouped based on the orientation and the size of the bounding box. This is useful, for instance, to separate pedestrians located close to the camera from horizontally oriented cars positioned farther away from the camera. Bounding boxes are normalized to have the same length of the longest edge, and they are positioned so that their centers overlap. The difference in orientation between two bounding boxes is captured by the smallest rotation angle alpha (ranging from 0 to 90 degrees) necessary to align the two boxes (see Fig. 1). The distance between two bounding boxes is then defined as the maximum of the two distances d_1 and d_2 as shown in Fig. 1. Finally, at level three, objects are clustered using the similarity between edge histograms.

Once the training objects are clustered, it is possible to estimate the probability density for a specific cluster. To do so, we divide the image with a grid and have each object found in the baseline image set contribute with a Gaussian distribution to the density. The standard deviation of the Gaussian for each object i is estimated as: $\gamma \frac{1}{n} \sum_i \frac{\sqrt{c_i}}{2}$ where c_i is the pixel count of object i, and n is the number of objects in the same cluster as object i. The user can set γ, to let the density estimation have the desired sparseness. For more than three standard deviations, the contribution is set to zero. Each training object is assigned to the element of the grid where it belongs. The contribution is computed only for the centers of the blocks of the grid. At the end of this process, each cluster (at the lowest level) has a density function associated with it, which will permit the calculation of the likelihood of seeing an object belonging to that cluster in a particular block of the grid.

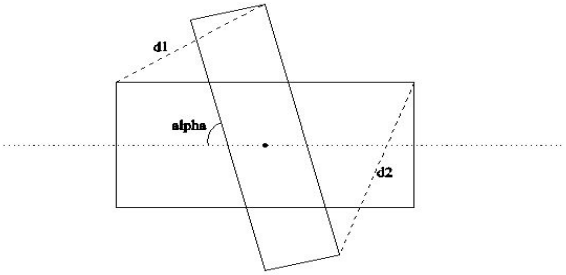

Figure 1. Bounding boxes and the second distance function.

The objects extracted from test images are assigned to the clusters of training objects as follows. For each distance level, StrOUD is first used against all the clusters. If the object does not fit any cluster for at least one distance level, it is declared an outlier; otherwise, we determine its cluster by using majority voting among the k closest clusters' members. Cluster refinements are performed at each level. This means that a test object is first assigned to one cluster based on its size. At the second level, the test object is compared only with the training objects within the same cluster, and so on. Finally, once the final cluster of a test object is determined, we estimate the probability associated to their position, using the density function previously calculated for the cluster. In the grid method, a test object takes the probability associated to the grid element it belongs to. (If the probability is zero, it is set to 10^{-100}, since the probability of outlier objects is set to 10^{-100}.)

For each image we compute the size of the codeword describing the image as $(-\sum_i \log p_i)$, where the summation is over objects i detected in an image. This also represents the strangeness of an image. Running StrOUD using the size of the codeword as strangeness, it is possible to flag each image as normal or outlier with different confidence

levels.

3.2 Method 2 (M2)

Method 2 partitions a given image into blocks. It then estimates the probability for an object to appear in a given position of a specific block by utilizing only objects seen within the same block (in the baseline data). Thus, this approach performs local estimations of probabilities, where locality is defined by the size of the blocks, without clustering objects into groups.

A grid of fixed size (for the entire process) is used, and every image is divided according to such grid. For the baseline images, the representations of objects found in each block are saved.

To compute the codeword sizes for the baseline images, we proceed as follows. For every object in an image, located in block B, we compute the probability of being in its current position using a non-parametric model of the form:

$$p_i = 1 - \frac{\min_{i,o \in B, i \neq o} d(i,o)}{\max_{i,o \in B, i \neq o} d(i,o)}$$

where o represents an object (different from i) seen in B in any of the baseline images. The model computes the probability as the complement of the ratio between the minimum distance of the object in consideration from any object in B, and the maximum distance of the object in consideration from any object in B. Notice that the two extremes of this function are 0, when the minimum and maximum distances are the same, and 1, when the minimum distance is 0. The first case corresponds to an object that has no close neighbors, the second to an object that has an identical copy in the block.

Using the probability so obtained, it is simple to compute the contribution of the object to the codeword size, and ultimately to compute the codeword size of the image. In this way, the distribution of strangeness for the baseline images can be obtained.

For a new image that needs to be diagnosed, the process is similar. For every object of the image, the non-parametric model above allows to compute its probability. The strangeness of the image is then computed. Finally, StrOUD provides a diagnosis of the image.

4 Empirical Evaluation

4.1 Data

We conducted experiments with three datasets. Two were "synthetically" created by taking digital photographs of a controlled scene consisting of a few objects on a table. We moved the objects in certain restricted patterns first to create a baseline of normal images. Later we moved the objects to areas in the image where they had never been before (in the baseline set) and introduced new objects to the scene. We also made objects (including background objects that had never moved before) disappear and background objects change their positions to create anomalies. These two datasets were designed to have a proper account of normal and outlier frames, so we could do a quantitative evaluation of the techniques. These two sets were shot with a Sigma SD9 SLR digital camera with the FOVEON®X3™image sensor that has three layers of Pixel Sensors (each pixel captures 3 colors as opposed to the standard 1 color-pixel), using high resolution (2263x1512 pixels). Details are provided below.

	Detected as Normal		Detected as Outlier	
	M1	M2	M1	M2
Normal (34)	32 94.12%	29 85.29%	2 5.88% FP	5 14.71% FP
Outlier (34)	1 2.94%	9 26.47%	33 97.06% TP	25 73.53% TP

Table 1. Confusion matrix for dataset 1.

	Detected as Normal		Detected as Outlier	
	M1	M2	M1	M2
Normal (24)	23 95.83%	24 100%	1 4.17% FP	0 0% FP
Outlier (22)	1 4.54%	1 4.54%	21 95.46% TP	21 95.46% TP

Table 2. Confusion matrix for dataset 2.

In each of the experiments the results are reported for the best selection of parameters. The value of γ is kept to 1 (variations over this parameter did not exhibit a significant change of the results). The confidence utilized is 95%. In method M1 a value of $k = 1$ neighbors is used to determine the best cluster fit.

Dataset 1. A sample frame is shown in Fig. 2. Most of the pictures contain five objects: two toy cars, a toy plane, a toy train, and a small chest that serves as part of the background. The cars, plane, and train move in specific patterns throughout the baseline set of frames as shown in the same figure (Fig 2(Lower)) the plane moves on a strip on the upper left corner of the image; the cars on a strip in the middle part of the image; the train on a strip in the right side of the image). The baseline part of the set is composed by 90 images, each with the foreground objects located in their "normal" positions (somewhere along the strips indicated in Fig 2(Lower)). The test part of the set consists of 68 images:

34 of them are normal (similar to those in the baseline), and 34 of the images contain at least one anomaly. Anomalies consist of foreground objects placed in parts of the image where they have never been before (e.g., a car in the plane's strip), background objects (i.e., the chest) disappearing or being moved to other places in the image, or new objects (e.g., a remote control device) appearing in the image. Examples of the outliers can be seen in Fig. 3. The clustering of the objects in the baseline set was set at 30 clusters.

Dataset 2. This set is similar to the first. It is composed by a series of frames shot over a controlled scene with objects following normal patterns of movement in the baseline part of the set and being moved to anomalous spots in the test part. Fig. 4 shows a sample frame for training. In the 82 baseline images, the foreground objects – the small chest and the grey remote control – are seen in the middle part of the image. In the test images, we have 24 normal frames (similar to the baseline frames) and 22 outliers with foreground objects misplaced, background objects moved or disappearing and new objects that have not been seen in the baseline. The clustering of the objects in the baseline data was set at 9 clusters. (Fig. 5 shows two examples.)

Dataset 3. This data was obtained by placing a camera close to an office window in a building on campus. The camera was pointing at a traffic intersection on campus and shooting a video of on-going activities. The camera used for this last set is an IQEye 511. The images were taken at 30 frames per second, 704x480, all of the other features (white balance, etc.) where disabled. A frame of the baseline data is shown in Fig. 6. The original set has 7680 images. The first 3680 are considered training and the remaining 4000 as test. The number of clusters of objects for the baseline data is set at 45.

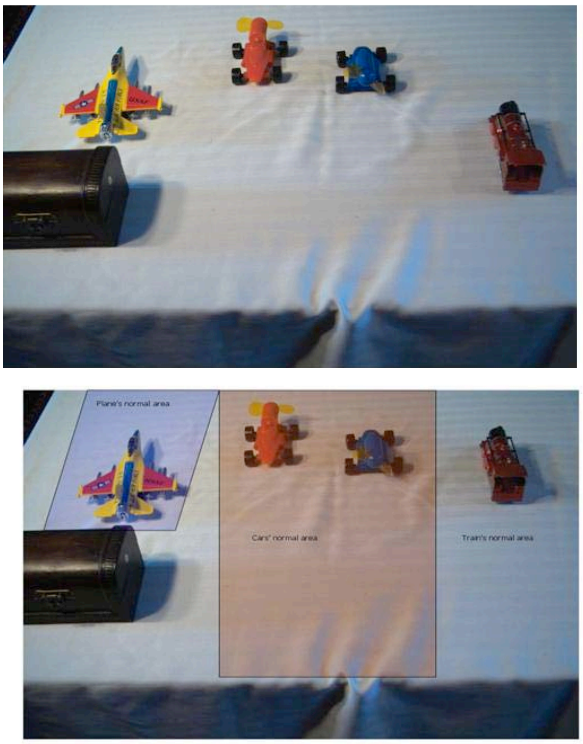

Figure 2. (Upper) A frame from dataset 1; (Lower) Normal patterns of movement for foreground objects in dataset 1.

4.2 Results

4.2.1 Basic results

In this section we show the confusion matrices (false positives and true positives) for the best results obtained on datasets 1 and 2 using the two methods described in Section 2, along with examples of outliers found for dataset 3 (we do not have the ground truth for that set, so quantitative analysis of the results are not available). The results reported in this section are obtained using the best settings for the two methods. The confidence level used in the experiments is 95%.

Table 1 shows the results obtained for dataset 1, using 90 images as baseline and with a test set composed of 34 normal images and 34 outliers.Table 2 shows the results on dataset 2 with a test set of 24 normal images and 22 outliers. M1 performs well on both datasets, with high true positive and low false positive rates. M2 does specially well on dataset 2 with 0% false positives and a high (95.46%) true positive rate.

For dataset 3, Fig. 7 shows examples of images that were flagged as outliers by our techniques. In all these images, foreground objects (or people) were detected in unlikely positions, and therefore caused a large increase in codeword sizes. Image (c) is an exception: it contains no foreground objects. It is flagged as an outlier because our statistical testing procedure considers both tails of the distribution of the α values. In this case, the strangeness (or p-value) (see eq. (1)) of the image is too small (smaller than the values for the baseline).

4.2.2 Baseline set size

Fig. 8 shows false positive and true positive rates as a function of the size of the baseline. For M1, as fewer baseline data is available, clusters contain fewer objects, and test objects are flagged as strange more often. As a result, we have more false positives and also an increased number of true detections. The trend for true positives is less marked in dataset 2, since the detection rate is quite high to begin with. For M2, as fewer data is available, each grid block gets pop-

(a) (b) (c) (d) (e) (f)

Figure 7. Images detected as outliers in dataset 3. (a) Three jaywalkers on the access road (i.e., pedestrians in a location they have not been seen in the baseline set); (b) Big truck on the access road (that object was never present there in the baseline set); (c) Lack of foreground objects; (d) Jaywalker in the main road; (e) Pedestrian on the right (no sidewalk); (f) Green truck on the sidewalk

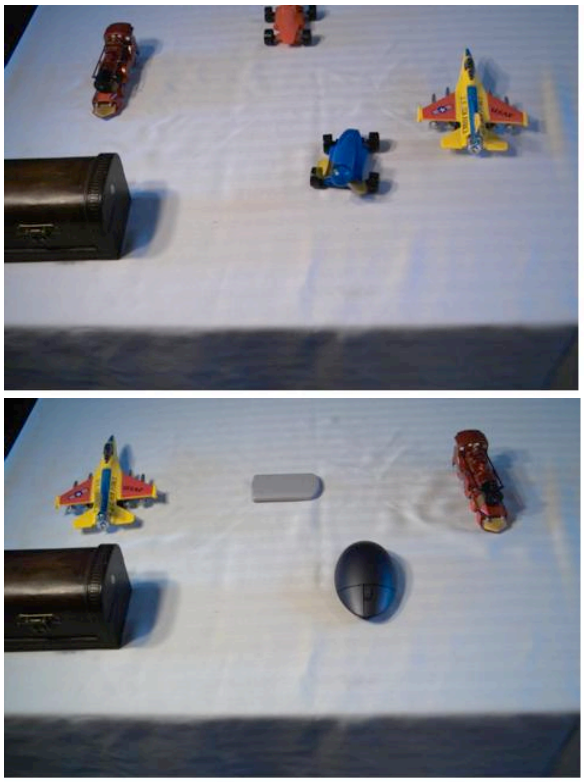

Figure 3. Examples of outliers of dataset 1.

Figure 5. Examples of outliers of dataset 2.

Figure 4. Example of a baseline image of dataset 2.

Figure 6. Example of a baseline image of dataset 3.

ulated very sparsely. Each object receives low probability (even the training ones); therefore, the size of the codewords is larger. As a result, more objects are considered as normal. Hence, as the training set gets smaller, we observe less false positives and less true positives. The trend for false positives is not noticeable in dataset 2, since the rate is 0 from the start.

4.2.3 Scale of the objects

We tested the effect of changing the scale of the images on TP and FP rates. The objective was to test the sensitivity of our approach to the quality of the images (smaller frames corresponds to less quality). Fig. 9 summarize the results. For both methods, decreasing the size of the image reduces the number of true positives. In M1, this is due to the fact

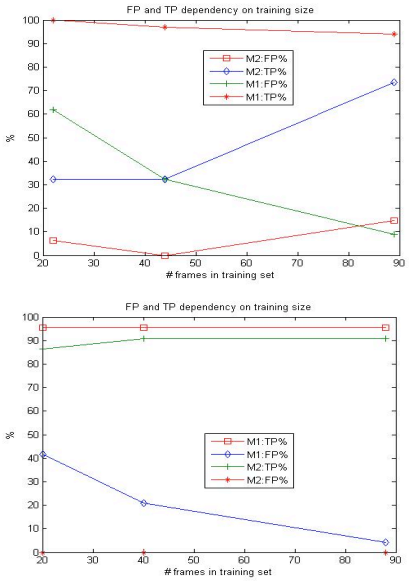

Figure 8. False positive and true positive rates as a function of the size of the baseline set. (Upper) dataset 1; (Lower) dataset 2

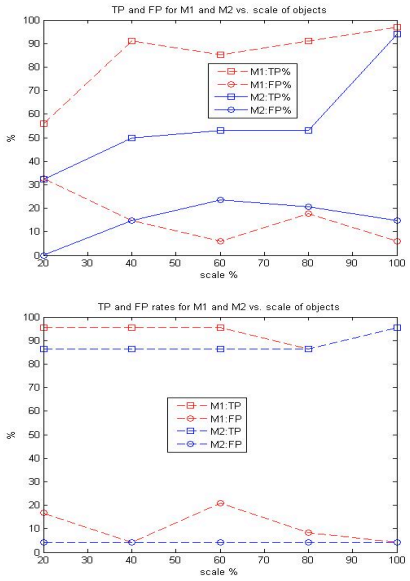

Figure 9. TP and FP rates for different scales of the frames in a) dataset 1; b) dataset 2.

that all objects start appearing more similar to each other. This results in less clusters, and an estimation of density functions closer to uniform distributions that cover the entire image. Thus, objects that should be detected as "out of place" or "never seen" are not identified as such. In M2, the effect in individual blocks is the same. Objects appear similar to each other and therefore do not get flagged as strange. The trend is less marked in dataset 2, because there are less foreground objects in that set.

4.2.4 Grid Size

The grid size affects the two methods differently. In M1, the grid size has an impact on the density estimation. In M2, however, the impact is on the objects that we will find in a particular block, and therefore, on the probability given by the non-parametric model. For this reason, we decided to conduct separate experiments for the two methods and report them in separate figures. Fig. 10 shows the effect of the grid size on M1. The x-axis reports the number of divisions made in each dimension of the image (e.g., the value three corresponds to nine blocks). In general, the method is quite robust with respect to grid sizes, maintaining similar values of TP and FP rates throughout the tested range. Fig. 11 shows the effect of the grid size on M2. The x-axis in this case represents the size (in pixels) of one block (the smaller the size the larger the number of blocks in the image). The trend in dataset 1 is a decrease of the true positive rate as the blocks get very small. This is due to the fact that at those sizes only partial objects are included in the blocks and they all look similar. The trends of TP and FP for a large range of grid sizes are quite stable.

5 Conclusions and future work

We have introduced a technique that flags anomalies in images, with high true positive and low false positive detection rates. Two methods were investigated. M1 makes use of clustering and density estimation to define compact and probabilistic representations of images. In this context, clustering can be interpreted as a compression technique useful to handling noise. M2 avoids clustering by storing the objects observed in the baseline (organized by grid cell). Local comparisons are carried out between new objects and the baseline, where locality is defined by the cell of the grid. M2 resembles "lazy learning" paradigms, where no training is carried on off-line: all the computation is performed once a test image is presented to the technique. Since objects are stored by grid cell position, their retrieval and comparisons with test objects can be carried out quite efficiently. On average, the on-line computational complexity of M2 is similar to that of M1. Nevertheless, M2 has larger memory requirements, as it needs to store the objects of the baseline (M1 discards them after performing clustering and density estimation). In the future, we plan to aid the computation of density functions with the use of domain knowledge. Fur-

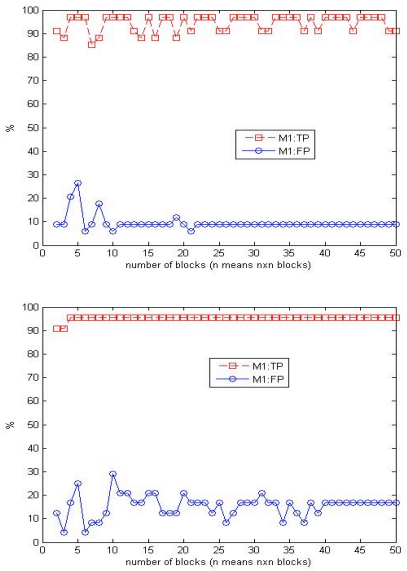

Figure 10. M1's TP and FP rates for different grid sizes in (Upper) dataset 1; (Lower) dataset 2.

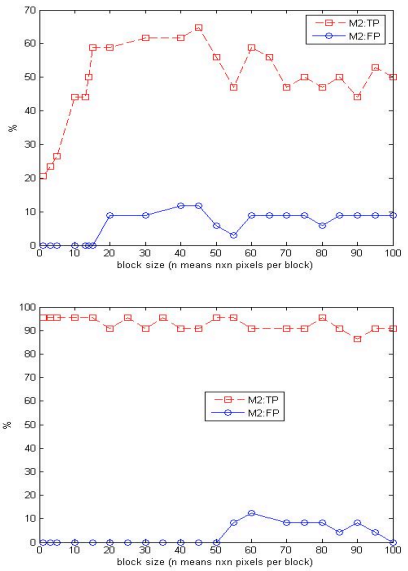

Figure 11. M2's TP and FP rates for varying grid sizes for (Upper) dataset 1; (Lower) dataset 2. The numbers in the x-axis mean the number of pixels per block (n means $n \times n$ pixels).

thermore, we plan to improve the process of discovering anomalies by using additional information such as sound, and to deal with situations where images are poorly registered.

References

[1] D. Barbará, C. Domeniconi, and J. P. Rogers. Detecting outliers using transduction and statistical testing. In *KDD '06: Proceedings of the 12th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 55–64, New York, NY, USA, 2006. ACM Press.

[2] T. M. Cover and J. A. Thomas. *Elements of Information Theory*. Wiley-Interscience, August 1991.

[3] C. Faloutsos, R. Barber, M. Flickner, J. Hafner, W. Niblack, D. Petkovic, and W. Equitz. Efficient and effective querying by image content. *Journal of Intelligent Information Systems*, 3(3/4):231–262, 1994.

[4] A. Gammerman and V. Vovk. Prediction algorithms and confidence measures based on algorithmic randomness theory. *Theoretical Computer Science*, 287(1):209–217, 2002.

[5] D. Hawkins. *Identification of Outliers*. Chapman and Hall, 1980.

[6] A. K. Jain and R. C. Dubes. *Algorithms for clustering data*. Prentice-Hall, Inc., Upper Saddle River, NJ, USA, 1988.

[7] E. M. Knorr and R. T. Ng. Algorithms for mining distance-based outliers in large datasets. In *VLDB '98: Proceedings of the 24rd International Conference on Very Large Data Bases*, pages 392–403, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

[8] S. Ramaswamy, R. Rastogi, and K. Shim. Efficient algorithms for mining outliers from large data sets. In *SIGMOD '00: Proceedings of the 2000 ACM SIGMOD international conference on Management of data*, pages 427–438, New York, NY, USA, 2000. ACM.

[9] L. Shapiro and G. Stockman. *Computer Vision*. Prentice Hall, 2001.

[10] B. W. Silverman. *Density Estimation for Statistics and Data Analysis*. Chapman & Hall/CRC, April 1986.

[11] V. N. Vapnik. *Statistical Learning Theory*. Wiley-Interscience, September 1998.