REDUCTION OF FALSE ALARMS TRIGGERED BY SPIDERS/COBWEBS IN SURVEILLANCE CAMERA NETWORKS

R. Hebbalaguppe[⋆†], *K. McGuinness*[⋆], *J. Kuklyte*[⋆], *R. Albatal*[⋆], *C. Direkoglu*[⋆], *and N. E. O'Connor*[⋆]

[⋆] Insight Centre for Data Analytics, Dublin, Ireland [†] TCS Innovation Labs, Delhi, India

ABSTRACT

The percentage of false alarms caused by spiders in automated surveillance can range from 20-50%. False alarms increase the workload of surveillance personnel validating the alarms and the maintenance labor cost associated with regular cleaning of webs. We propose a novel, cost effective method to detect false alarms triggered by spiders/webs in surveillance camera networks. This is accomplished by building a spider classifier intended to be a part of the surveillance video processing pipeline. The proposed method uses a feature descriptor obtained by early fusion of blur and texture. The approach is sufficiently efficient for real-time processing and yet comparable in performance with more computationally costly approaches like SIFT with bag of visual words aggregation. The proposed method can eliminate 98.5% of false alarms caused by spiders in a data set supplied by an industry partner, with a false positive rate of less than 1%.

Index Terms— Spider detection, False alarm reduction, Computer Vision, Surveillance, Descriptor fusion

1. INTRODUCTION

Security companies offer network cameras equipped with intrusion detection software to protect customers' property e.g. when the perimeter of the area monitored is breached, an alert is flagged to a human operator. True events are typically considered to be those that might cause potential hazard to the monitored environment e.g. people, animals, and vehicles. However, in many systems, operators can become overwhelmed by false alarms triggered by insects, foliage changes, changes in lighting conditions, etc.

Working closely with a security company, we obtained a data set of over 2,000 events triggered across 275 camera views. Analysis of the data showed: 30% of alarms were triggered by spiders/webs, 33% by people, 4% by animals, 23% by vehicles and 10% due to other sources like cloud or foliage motion (see Section 3). Spiders are attracted to cameras due to the heat emitted and the availability of prey such as houseflies and other insects [1]. The ability to detect and suppress alarms caused by spiders/insects could help dramatically reduce false alarm rates. Our analysis showed false alarms are triggered by spiders when they crawl over the surface of surveillance camera lens or when the spiderwebs shake due to wind. The false alarms raised by spiders can result in: (a) increased workload for human operators validating the alarms; (b) increased labor cost associated with regular cleaning of the lens to avoid frequent build up of spiderwebs; (c) costly erroneous notification of police forces.

Detecting spiders in a surveillance context is challenging due to varying environment conditions (rain, snow, day/night), compressed low resolution images, and limited number of key frames. Erratic spider movements in successive image frames and varying viewpoints based on how the surveillance camera is mounted make it difficult to analyze spider shape and structure. Spiders typically are seen close to lens surface unlike true event candidates like people and vehicles.

The technique proposed in this paper determines if an image contains a spider/spiderweb using computer vision techniques. At the time of submission, we are unaware of any attempt to reduce false alerts by spiders in surveillance system using computer vision. Existing approaches are based on generating frequencies to deter pests [2] or changing the camera to a dome shaped camera to facilitate the reduction in the formation of webs [3]. However, all these solutions involve replacement of currently deployed cameras, which would be expensive. Spider deterrent sprays are available but this requires expensive and repetitive human effort given that spiders tend to reappear frequently even when using chemical sprays.

Our contributions are as follows: We propose an algorithm that can distinguish between events triggered by *spiders* and those triggered by other causes like people, vehicles and animals. The method also assists the human operator by associating a confidence score to the detected events using Support Vector Machines produced using a variant of Platt's method [4]. These confidence scores can then be used to filter events that have high probability of being caused by spiders or spiderwebs, while ensuring true events are very unlikely to be classified incorrectly.

2. PROPOSED METHOD

To develop a spider classifier, we formulate the problem as a binary classification task. The manually annotated set for training data takes the form $\{(x_1, y_1), (x_2, y_2), \ldots, (x_n, y_n)\}$,

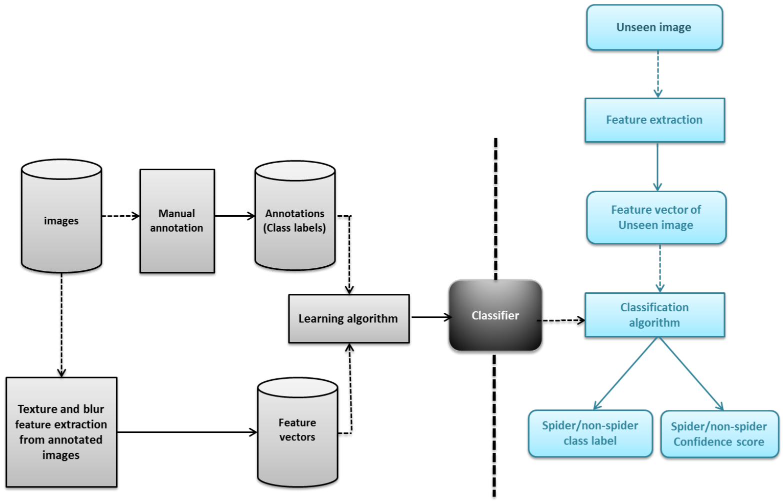
images
Manual annotation
Annotations (Class labels)
Learning algorithm
Texture and blur feature extraction from annotated images
Feature vectors
Classifier
Unseen image
Feature extraction
Feature vector of Unseen image
Classification algorithm
Spider/non-spider class label
Spider/non-spider Confidence score

Fig. 1: Block diagram showing the various components of the proposed spider classification system: vertical dashed line separates *Learning* and *Classification* blocks.

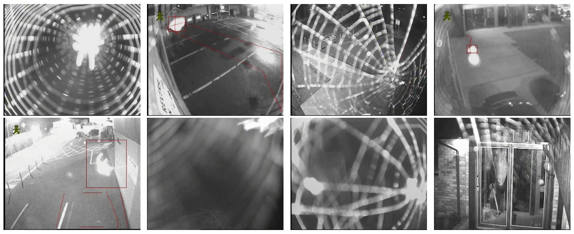

Fig. 2: Positive examples used in image classification: *spider class* comprising of spiders and spiderwebs.

where $x_i \in X$ is a vector of feature values computed for a test image *i* and $y \in \{0, 1\}$ is a binary label of example *i*. Positive examples are images belonging to *spiders* category comprising of spiders and spiderwebs; negative examples are the *non-spider* category comprising of people, animals and vehicles. A function $f : X \rightarrow \{0, 1\}$ learns to map every test image in X to a class label. Figure 1 outlines the proposed method.

2.1. Visual Feature Extraction

We learn a function, f based on the fact that spiderwebs tend to have a repetitive coarse texture (see Figure 2) and we use extent of blur as another important discriminatory feature leveraging the fact that that most spiders are outside the field of depth and hence appear blurry. In addition, the coarse regular pattern of webs provides valuable cues for selection of statistical texture features. We combine features into a single feature vector. No motion features (e.g. optical flow) are used , since the smoothness constraints of optical flow computation are usually violated due to the low temporal resolution of surveillance cameras.

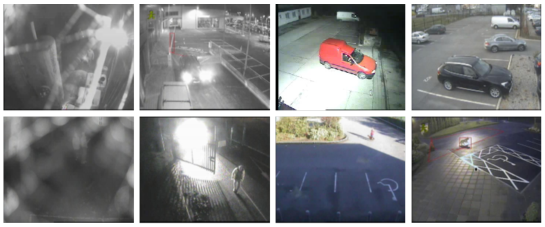

Fig. 3: Negative examples used in image classification: *non-spider class* comprise of animals, people and vehicle. Note the presence of spiderweb with human is manually annotated into *non-spiders* category.

2.1.1. Cumulative Probability of Blur Detection (CBPD)

As evident from Figure 2, spiders/spiderwebs are most likely closer to the surveillance camera lens. This range is outside the depth-of-focus and hence, spiders and webs appear defocussed and blurry. We choose a blur metric based on CPBD [5] as it is non-referential in nature and takes a human attention model into consideration. The threshold to mark a block as edge/non-edge was empirically chosen as 0.002 and filtering parameter, β to be 3.6.

2.1.2. Haralick Texture Features

Haralick et al. [6] propose a method to describe textural properties by computing a set of gray-tone spatial-dependence probability distribution matrices and suggest a set of textural features which is extracted from these matrices – homogeneity measured by angular second moment given by f_1, linear structure, contrast measured by difference moment of that matrix given by f_2, number of edge boundaries present and complexity of an image given by f_3:

$$f_1 = \sum_{i=1}^{N_g} \sum_{j=1}^{N_g} \frac{P(i,j)}{R}^2, \tag{1}$$

$$f_2 = \sum_{i=0}^{N_g-1} n^2 \left\{ \sum_{|i-j|}^{n} \frac{P(i,j)}{R} \right\}, \tag{2}$$

$$f_3 = \frac{\sum_{i=1}^{N_g} \sum_{j=1}^{N_g} [ijP(i,j)/R] - \mu_x \mu_y}{\sigma_x \sigma_y}, \tag{3}$$

where, N_g is the number of quantized gray tones or distinct gray levels, and P_{ij} is the relative frequency with two neighboring resolution cells and μ_x, μ_y, σ_x, and σ_y are the means and standard deviations of marginal distributions associated with $P(i,j)/R$ and R is a normalizing constant. We have chosen to use the 13 significant texture features out of 28 for fast calculation of Haralick features as described in [7].

2.1.3. *SIFT with Bag of Visual Words Aggregation (SIFT/BoVW)*

We use the BoVW approach [8] to aggregate a variable number of SIFT descriptors [9] for an image into a fixed length histogram. This is done by first clustering the descriptors for all images in the training set to produce a codebook. Given this codebook a visual word histogram descriptor for an image is computed by assigning each SIFT descriptor to the nearest cluster center in this codebook.

2.1.4. *RootSIFT*

The performance of SIFT histogram for image classification can be boosted by using a better distance measure. RootSIFT [10] is simply a L1 norm of SIFT vectors followed by an element-wise square root of SIFT descriptor.

2.1.5. *LBP Variance*

Local Binary Pattern (LBP) features have the drawback of losing global spatial information, while global features preserve little local texture information. Using LBP Variance [11] an alternative hybrid scheme useful for globally rotation invariant matching is investigated along with locally variant LBP texture features. LBP codes are computed on sample points on a circle with user specified radius – in our case, we used a radius of one. LBP variance was used on (8,1) neighborhood and uniform rotation invariant LBP scheme was chosen for mapping.

2.1.6. *Fusion of Haralick texture features with CPBD (Haralick/CPBD)*

The Haralick texture features and the CPBD blur measure provide complementary information about the image content. As such, fusing the descriptors is likely to provide more relevant information to the classifier and produce a higher-accuracy result. We propose a simple early fusion strategy in which we simply concatenate the feature vectors obtained from CPBD [5] and Haralick texture features [6] followed by normalization of feature vectors. Since CPBD is a scalar, this simply increases the Haralick descriptor by one.

Descriptor	Dim.	Acc.
CPBD	1	65.8%
Haralick	13	91.6%
SIFT with Bag of Visual Words	100	98.9%
RootSIFT	100	99.28%
LBP Variance	10	98.5%
Fusion of Haralick and CPBD	14	98.82%

Table 1: Comparison of the classification accuracy of tested approaches. *Dim.* refers to the descriptor dimension, *Acc.* to the classification accuracy

2.2. Learning and Classification phase

The learning algorithm explores the feature space to identify separators that partition the feature space into positive and negative zones corresponding to *spider* and *non-spider* class. The result of the learning phase is a classifier that is able to associate confidence score to a previously unseen image, a normalized confidence score indicating the existence (≥ 0.5) or the absence (< 0.5) of a spider/spiderweb in an image.

3. EXPERIMENTS

Our dataset was gathered from CCTV surveillance footage from a surveillance company that uses video analytics to detect events that are passed to a human operator for manual verification. The existing analytics software generates three images taken a second apart for every event triggered (event triggering is based on frame differencing). Figure 4 shows some events detected by the existing software including artifacts introduced by the existing analytics system. The dataset was selected from a large number of events created by the existing analytics software after manual annotation. 2273 images caused by spider related events were found via manual annotation. An equal number of true events were randomly chosen producing a dataset containing 4546 images in total. The dataset is partitioned into two sets: 70% of the data (3182 images) is used for training and the remaining 30% (1364 images) is used for testing the out-of-sample performance of the classifiers. We use support vector machines classifiers and a variation of Platt's method to produce probability outputs. In our experiments, we used the soft margin SVM implementation provided by LIBSVM with a Radial Basis Function (RBF) kernel [12]. To find the optimal parameters (soft-margin con-

Method	feature extraction time(ms)	classification time(ms)	total time(ms)
CPBD	2106	0.11	2106.11
Haralick	31.2	0.25	31.45
LBP Variance	2464	0.368	2464.368
SIFT with Bag of Visual Words	8735.8	0.622	8736.4
Fusion of LBP Variance and CPBD	4570.8	0.204	4571.004
Fusion of Haralick and CPBD	2137	0.158	2137.158

Table 2: Computational times taken by each method per image in milliseconds.

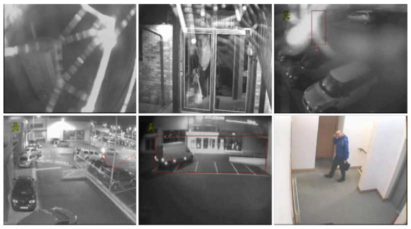

Fig. 4: The first row contains three images that were categorized as true positives (spiders classified as spiders); the second row shows true negatives. Artifacts (red trails and boxes) correspond to motion events detected by the existing analytics software.

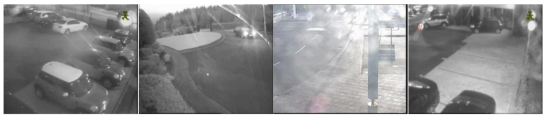

Fig. 5: The first two image illustrate false positives (non-spiders classified as spiders): observe that reflections and lighting produce an effect very similar in appearance to a spiderweb, which explains the misclassification. The second two images are false negatives (spiders classified as non-spiders); it appears that the extremely low contrast in these images may be responsible for the classifier failing to recognize the spiders correctly by our proposed algorithm

stant, C and inverse width parameter, γ) we perform a grid search on C and γ using five fold cross-validation.

3.1. Results

Figures 4 and 5 show examples of correctly and incorrectly classified images. Table 1 shows the classification accuracies (percentage of correct classifications) on the test data for each of the different types of features tested. The best performing methods are the fusion of Haralick and CPBD and SIFT or RootSIFT with BoVW, which achieve comparable accuracy. The Haralick/CPBD descriptor has lower dimension when compared to SIFT/RootSIFT, but slightly higher than LBP variance. Given the importance of computation time in surveillance applications, we propose fusion of Haralick and CPBD as the desired method providing favorable results with lower computation time compared to SIFT/RootSIFT - BoVW and LBP variance. The proposed descriptor, which fuses easily computable Haralick texture features and a blur metric based on cumulative probability of blur detection, produced a classification accuracy of 98.82%, which is comparable with the more computationally expensive SIFT/BoVW descriptor whose classification accuracy was 98.9%. There is a clear merit in fusion of complementary features contributing to better classification rates as seen in the Table 1 e.g. individual Haralick and CPBD classification rates were 91.6% and 65.8% while fusion of those features increased this to 98.82%. Considering the computational time shown in Table 2, we conclude fusion of CPBD with Haralick texture features presents lower test times as compared with SIFT with bag of visual Words and outperforms LBPVariance methods.

4. CONCLUSIONS AND FUTURE WORK

We have presented a cost effective method to reduce false alarms triggered by spiders. To the best of our knowledge, this is the first time computer vision has been used to reduce spider-triggered false alarms. The proposed method uses various computer vision and machine learning techniques to aid the human operators in validating and prioritization of events triggered. We found the algorithm to be generically applicable for insects close to the lens. In the future, we would like to convert online learning for an incrementally trained classification framework to recognize re-occurring events.

5. ACKNOWLEDGEMENTS

This research was conducted with financial support from the Enterprise Ireland Innovation Partnership programme and from Science Foundation Ireland (SFI) under grant number SFI/12/RC/2289 . We gratefully acknowledge the support of Netwatch Business Solutions.

6. REFERENCES

[1] “The sydney morning herald: City dwelling spiders getting all warm and fuzzy - and bigger,” December 3 2012.

[2] Yoon Hwa Ko, “Security camera capable of preventing spiders,” 2008.

[3] “Shdow security, ultimate digital video surveillance solutions,” .

[4] John C. Platt, “Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods,” in *Advances in Large margin Classifiers*. 1999, pp. 61–74, MIT Press.

[5] N. D. Narvekar and L. J. Karam, “A No-Reference Image Blur Metric Based on the Cumulative Probability of Blur Detection (CPBD),” *IEEE Transactions on Image Processing*, vol. 20, no. 9, pp. 2678–2683, 2011.

[6] Robert Haralick, K Shanmugam, and D Itshak, “Textural features for image classification,” *IEEE Trasnactions on Systems, man and Cybernetics*, vol. SMC-3, no. 6, pp. 610–621, 1973.

[7] Eizan Miyamoto and Thomas Merryman, “Fast calculation of haralick texture features,” .

[8] J. Sivic and A. Zisserman, “Video Google: A text retrieval approach to object matching in videos,” in *IEEE International Conference on Computer Vision*, 2003, vol. 2, pp. 1470–1477.

[9] D. G. Lowe, “Object recognition from local scale-invariant features,” in *The Proceedings of the Seventh IEEE International Conference on Computer Vision*, Los Alamitos, CA, USA, 1999, vol. 2.

[10] R. Arandjelović and A. Zisserman, “Three things everyone should know to improve object retrieval,” in *IEEE Conference on Computer Vision and Pattern Recognition*, 2012.

[11] Lei Zhang Zhenhua Guo and David Zhang, “Rotation invariant texture classification using lbp variance (LBPV) with global matching,” *Pattern Recognition*, vol. 43, no. 2010, pp. 706–719, 2010.

[12] Chih-Chung Chang and Chih-Jen Lin, “LIBSVM: A library for support vector machines,” *ACM Trans. Intell. Syst. Technol.*, vol. 2, no. 3, pp. 27:1–27:27, May 2011.