Computational Visual Media
DOI 10.1007/s41095-016-0039-3 Vol. 2, No. 2, June 2016, 143–152

Research Article

Rethinking random Hough Forests for video database indexing and pattern search

Craig Henderson[1] (✉), **Ebroul Izquierdo**[1]

Abstract Hough Forests have demonstrated effective performance in object detection tasks, which has potential to translate to exciting opportunities in pattern search. However, current systems are incompatible with the scalability and performance requirements of an interactive visual search. In this paper, we pursue this potential by rethinking the method of Hough Forests training to devise a system that is synonymous with a database search index that can yield pattern search results in near real time. The system performs well on simple pattern detection, demonstrating the concept is sound. However, detection of patterns in complex and crowded street-scenes is more challenging. Some success is demonstrated in such videos, and we describe future work that will address some of the key questions arising from our work to date.

Keywords Hough Forests; pattern detection; pattern search; machine learning

1 Introduction

Randomised Hough Forests were introduced in 2009 and demonstrated to detect instances of an object class, such as cars or pedestrians [1]. This, and subsequent works, have shown that Hough Forests can perform effectively in object detection tasks, and we see potential to translate their use to exciting opportunities in pattern search in large-scale data corpora.

Current systems are, however, incompatible with the scalability and performance requirements of an interactive visual search system. Contemporary research trains a forest using variants of an object and regression is performed using an unseen image in which instances of the pattern are probabilistically identified. We propose to conceptually invert the use of the forest by using the corpus data that is *to be searched* as training data, and a query image of an unseen pattern to be *searched for* (Fig. 1). The new schema results in a very fast search of a large set of data whereby a single pass of a small query image through the forest yields probabilistic hypotheses of pattern detection in every image in the training data. Conventionally, regression is performed on each image and while each one can take only a few hundred milliseconds, the time to run over a large corpus invalidates its use for search at large scale.

We make the following contributions:

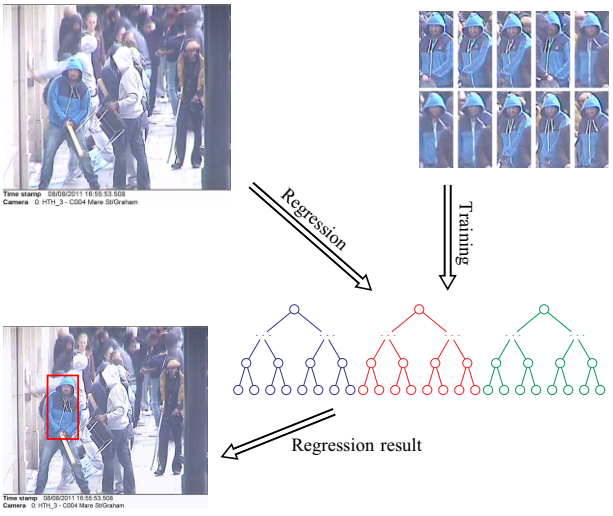
1. a new approach to Hough Forests training and pattern detection is described, suitable for large-scale pattern search in an interactive system;
2. a technique to train forests without explicit negative training images, eliminating the need for an otherwise arbitrary set of negative training images to be used to counter positive training images;
3. a method to select positive and negative training patches to filter noise from the background of images;
4. a flexible and scalable pattern search index that data can be added to or removed from at any time without need for any re-training of the existing forests.

1.1 Motivation

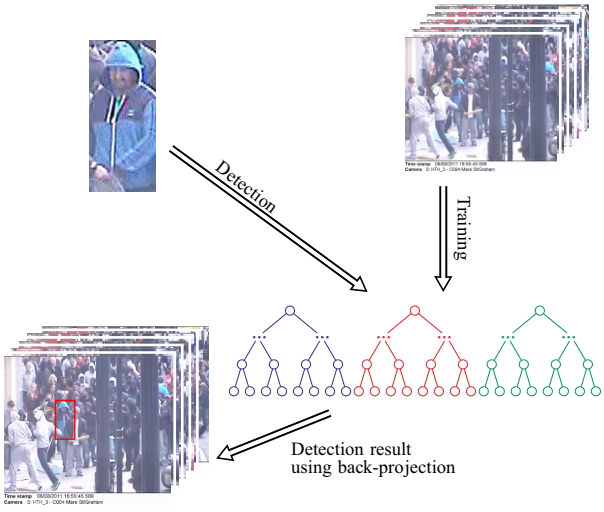
We are interested in the problem of searching a large corpus of video and still images for a previously

1 Multimedia and Vision Research Group, School of Electronic Engineering and Computer Science, Queen Mary University of London, London, E1 4NS, UK. E-mail: C. Henderson, c.d.m.henderson@qmul.ac.uk (✉); E. Izquierdo, e.izquierdo@qmul.ac.uk.

Manuscript received: 2015-11-24; accepted: 2015-12-18



(a) A typical forest training and regression schema. The forest is trained using random patches from images depicting an object or pattern and a regression is run on a query image to detect the object or pattern.



(b) Our schema inverts the training data and pattern image. The forest is trained using random patches from frames of a video sequence, and testing is run on a query image containing the object or pattern to be searched for. Using back-projection, one pass of the pattern through the forest simultaneously identifies the pattern in all frames of the video.

Fig. 1 (a) A typical forest training and regression process and (b) our novel process for video indexing.

unknown distinctive pattern such as a fashion or corporate logo, tattoo, or bag for left-luggage search. The genericity of the problem definition eliminates the opportunity to restrict the search, for example, by reducing the problem using a person detector to first find a suitable search area. The mixture of video and still images in the database also restricts the general use of spatio-temporal information available only in the video subset of data. We therefore seek a generic solution to a search problem that is flexible and fast enough to be an interactive search with user input defining a previously unknown pattern for which to search.

1.2 Nomenclature

Literature to date describe Hough Forests used in *object detection*. We are interested in a more general *pattern detection* regardless of the structure of the pattern, and therefore refer to *pattern* where it can be referred to an object in the case of other literature.

Our data corpus consists of video sequences and collections of related still images. Each video or collection of images is treated as a unit of data for indexing. For brevity, we refer to the unit as a *video*, where it can also mean a collection of images.

2 Background

Hough Forests [2] use a random forest framework [3] that is trained to learn a mapping from densely-sampled D-dimensional feature *cuboids* in the image domain to their corresponding votes in a Hough space $\mathcal{H} \subseteq \mathbb{R}^H$. The Hough space accumulates votes for a hypothesis $\boldsymbol{h}(c, \boldsymbol{x}, s)$ of an object belonging to class $c \in C$ centred on $\boldsymbol{x} \in \mathbb{R}^D$ and with size s. Object hypotheses are then formed by local maxima in the Hough space when voting is complete. Votes contributing to a hypothesis are called *support*. The term *cuboid* refers to a local image patch ($D = 2$) or video spatio-temporal neighbourhood ($D = 3$) depending on the task. Introduced in 2009 [1], Hough Forests have gained interest in many areas such as object detection ($D = 2$) [4–8], object tracking [9], segmentation [5, 10], and feature grouping [7]. With $D = 3$, action recognition is an active research area that has used Hough Forests [11, 12], and in Ref. [13], more general randomised forests were used as part of a solution to index short video sequences using spatio-temporal interest points.

Hough Forests have shown to be effective in class-specific object detection [1], multi-class object detection [14], tracking object instances [2], and pattern detection [15], and their performance is suitable for real-time tasks with sub-second regression time on 800 × 600 dimension images. The high performance and effective accuracy make Hough Forests an attractive proposition for large-scale video database pattern searching. However, the conventional use of forests in object detection tasks

does not scale well; a forest is trained with example images of the object or class of object to be found, and then a regression is run on unseen image to detect the object. In an interactive search system, the query *pattern* is unknown until runtime, when it is specified by the user. In this paper, we seek to address this conflict, rethink the method, and use of a randomised Hough forest for high-performance visual search.

3 Rethinking forests

In a departure from the established method, we conceptually invert the use of the forest and consider the image domain to consist not of instances of an object class or variations of the pattern to be detected, but the corpus to be *searched*. In our case, a forest's image domain is a set of frame images from a single video within the corpus.

We use a collection of forests to build a complete index of our video and image corpus. Each forest is trained using a single video (Fig. 2)—which can be just a few frames or 90,000 frames for one hour sequence at 25 fps—and can therefore be considered a sub-index of the database relating exclusively to a single video. Each forest $\mathcal{F} = \{\mathcal{T}_1, \mathcal{T}_2, \mathcal{T}_3, \ \ldots, \ \mathcal{T}_N\}$ consists of N trees where each tree $\mathcal{T}_i$ is independent of all other trees for training and testing.

Forests are trained using a novel scheme to identify positive and negative training samples (Section 3.2) from the frames of a video, thus removing the usual need for an explicit set of negative training images. Training is the most time-consuming function, and can be done as an offline activity for each video, independently of all other videos in the index. Training each forest is therefore consistent with building a video index in a more conventional retrieval system. A trained forest provides fast access to the patterns contained within the video such that it is searchable on-demand for unseen patterns of any size and dimension.

To perform a search, patches are extracted from a query image (or sub-image defined by a query region) using a sliding window, and passed through the forest. Rather than accumulating votes in the two dimensions of the query image, we accumulate votes in three dimensions of width and height of the training dataset and depth of the number of training images. The *support* of the leaf is used to trace the contributing vote back to the source frames, and the vote is accumulated in each of them (Section 3.3).

The independence of the components within a collection of forests is important for large-scale searching, providing scalability and flexibility.

Scalability. To support large video database search, the index must be highly scalable. The independence of components in the forest collection means that it is massively scalable and processing can be extended across many cores, processors, and machines. Training trees can be performed in parallel as there is no dependence between individual trees. Pattern search is less time-consuming, but similarly scalable—each forest can pass the query image patches through all of its trees simultaneously and accumulate the results as they complete.

Flexibility. New videos can be added easily without need for any re-training of existing forests. A new forest is simply created, trained with the new

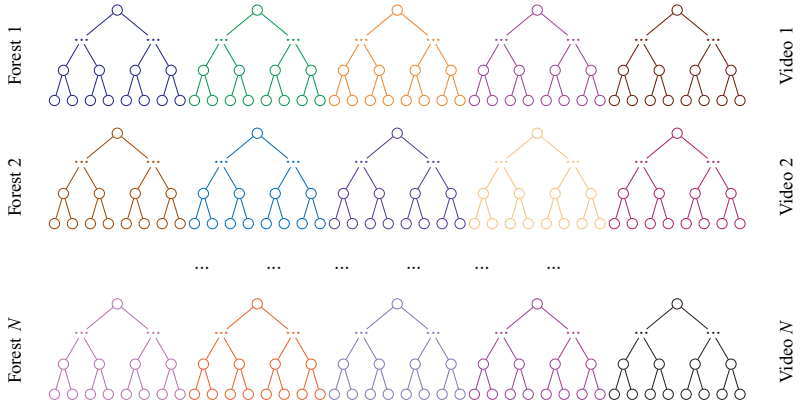


Fig. 2 The video database index is a collection of independent forests. Each row represents a forest of five independent trees, trained using frames from single video sequence.

data and then added to the collection. If a video is no longer required to be searched, then the relevant forest can simply be removed from the collection and will no longer be included in future searches. No re-training is necessary. Where available, the date of the video can be added as a property of the forest to further increase performance and search result relevance. A user can specify a time frame and forests containing videos from outside of the date range can easily be excluded from the search.

3.1 Hough Forests algorithm

We make some small amendments to the Hough Forests algorithm to achieve our goals. First, an *offset vector* is used [14] to translate the centre of the object in the training set to the centre in the regression step. With the inversion of training and query data, voting is accumulated at the patch position in the original frame dimension (Section 3.2) and the centre point offset into 2D Hough space of dimension of the query is not used, so we omit storing vector. Second, we do not scale images such that the longest spatial dimension of the bounding box is a known length. We work in the original frame dimensions of the video to avoid image data loss caused by resizing. Third, at each leaf, we record the training image to which each patch belongs, and finally, voting weights are accumulated in a multi-dimensional Hough space that includes the frame number.

3.2 Training the forest

A forest is trained using sets of *patches* that are extracted from the training images at random positions. Two sets of training images are conventionally used: a *positive* set of images containing examples of the object class or pattern to be detected, and a *negative* training set consisting images that do not contain any examples. Patches are extracted from each set yielding positive and negative *training patches*, respectively. The original authors of Hough Forests object detection [1] and subsequent research use a pre-determined patch size of 16×16 pixels, and extract 50 patches from each training image (resized such that the longest side is 100 pixels) to use in training the forest.

From each image, 32 feature channels are created: three colour channels of the Lab colour space, the absolute values of the first- and second-order derivatives in the x- and y-directions, and nine HOG [16] channels. Each HOG channel is obtained as the soft bin count of gradient orientations in a 5×5 neighbourhood around a pixel. To increase the invariance under noise and articulations of individual parts these 16 channels are processed by applying the min and the max filtration with 5×5 filter size, yielding 32 feature channels (16 for the min filter and 16 for the max filter).

Patches are extracted from random positions within a training image, overlapping to build up a patchwork representation of the image (Fig. 4(b)). It therefore follows that the quantity and size of patches as well as the size of the training image determine the coverage. Our recent study [15] demonstrated some increase in detection accuracy can be achieved by dynamically selecting the size and quantity of training patches without resizing the training images. A large number of patches in a small training image will cause a lot of overlap, leading to redundancy in the forest. Although random forests do not suffer from over-training [3], bloating the trees with excessive patches does impede runtime performance. In an image of dimension 704×625, a saturated coverage using 16×16 dimension patches with one patch in each position on a grid would require ρ patches, with some patches overlapping because the height is not divisible by the patch height.

$$\begin{aligned} \rho &= \left\lceil \frac{704}{16} \right\rceil \times \left\lceil \frac{625}{16} \right\rceil \\ &= 44 \times 40 \\ &= 1760 \end{aligned} \tag{1}$$

where $\lceil \cdot \rceil$ represents the *ceiling* function that maps a real number to the smallest following integer. Training time increases 1256% with this 3420% increase in the number of patches from $\rho = 50$.

However, saturated coverage with little or no overlap is a poor solution for us; the pattern is unlikely to align neatly to a grid and with each patch casting a single weighted vote, the accumulation to indicate the presence of the pattern will be weak. Positioning patches is therefore a critical step in the algorithm.

Randomness is used extensively in computer science as a means to avoid bias. Consider the training images stacked one on top of another. If patches were always placed in the same position, then each patch would represent a tunnel view through the images. All image data outside of these tunnels would be ignored. Furthermore, algorithms are usually trained using a corpus of images containing

objects that contain neatly cropped and centred views of the objects. In such cases, if the position of the patch extracted from every image was in the same position, then the model would be trained only on a subset of tunnel-vision views of an object, resulting in *over-training* and ineffective detection. Variation is therefore important, and randomness is used to achieve variation, in this case to avoid over-training. However, randomness is non-deterministic, and repeated execution of the same algorithm will produce different results, which makes measuring the effect and change in accuracy of incremental adjustments impossible to compare. To overcome this, the random number generator can be seeded to a fixed value, to yield a repeatable sequence of psuedo-random numbers. This is the approach adopted in experiments in this paper.

Selecting patches. Patches are extracted from training images at random positions, thus leaving to chance the amount of coverage of the image that is used for training. If the number and size of patches are large relative to the dimensions of the training images, and the random number generator has a uniform distribution, then the chances of the coverage being spread evenly over the image and sufficient to capture the image detail are increased.

We are interested in indexing video sequences from street-scenes such as closed circuit television (CCTV) images, often filmed from high in the air and, although cluttered with respect to large crowds, still containing non-distinct areas for which searching is unlikely. Random sampling from the image is too hit-and-miss (Fig. 3) to be reliable in extracting patches of use for a search system. Patches are therefore selected based on a statistical measure of how distinctive they may be in any subsequent search. The *grey-level co-occurrence matrix* (GLCM) is a well-known statistical method of examining texture that considers the spatial relationship of pixels. We use the derived *GLCM-Contrast* statistic that measures the local variations within the matrix. For brevity, we refer to *GLCM-Contrast* simply as the *contrast* of a patch. For each of the randomly positioned 16×16 patches, the normalised contrast τ_i of the patch is calculated. A patch is selected as a positive training patch if $\tau_i > \lambda$, otherwise the patch is considered too low-contrast for positive training. This selection restricts the random placement of positive training patches to be within high-contrast areas which are more distinctive and therefore more searchable (see green patches in Fig. 4(b)).

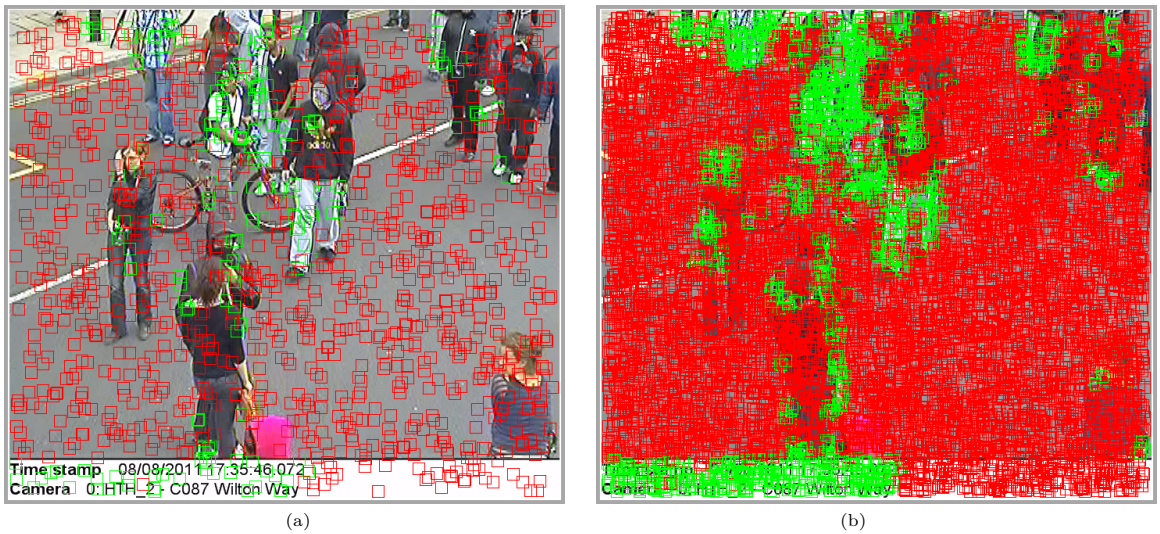
Fig. 3 Extracting patches at random is unreliable for selecting sufficient training patches to yield a good search results. Only 1 of 200 randomly placed patches will extract any of the Adidas® logo on the black hoodie.

Training the forest with the corpus leaves no sensible choice for negative training images to counter the positive training images. However, the use of the contrast measure has rejected a number of random patches that will not be used for positive training. We therefore use these rejected patches as negative training patches, and overcome a significant problem by providing a contextual set of negative training patches. Each patch is therefore selected as a *positive* or *negative* training patch based on the normalised GLCM-Contrast of the patch, thus:

$$\text{patch} = \begin{cases} \text{positive training}, & \text{if } \tau_i > \lambda \\ \text{negative training}, & \text{otherwise} \end{cases} \quad (2)$$

where $\lambda := 0.015$ for 16×16 patches in our experiments.

Figure 4 shows extracted patch regions that are selected positive (green) and negative (red) in the example image of Fig. 3, when searching for 200 (Fig. 4(a)) and 1760 (Fig. 4(b)) positive patches. The positive patches are clustered in the high-contrast areas of the image, which are those that contain searchable distinctive patterns. Light clothing is selected because the patch contrast is affected by shadows caused by creases, etc. Low-contrast areas are ignored as they are not distinctive patterns. The high density of 1760 patches in Fig. 4(b) shows so much overlapping of patches that the low-contrast



(a) (b)

Fig. 4 Positive and negative training patches. Green patches are positive training patches and red patches are selected as negative training patches because the GLCM-Contrast is very low (Section 3.2). (a) Left, 200 positive patches are selected, and (b) right, 1760 positive patches are selected (Eq. (1)) and show the low- and high-contrast areas of the image. In each case the number of negative patches is determined by the search for positive patches—negative patches are discarded by low-contrast considerations.

and high-contrast regions become visible.

3.3 Detection

A forest, trained from all of the frames from a video, is synonymous with an index of patterns for searching the video. A single pass of patches from the query image through the tree finds a probabilistic hypothesis for the pattern's occurrence in *each frame*. The interactive visual search is therefore very fast. A given query region is defined by the user by drawing a rubber band around a pattern of interest—for example, a distinctive pattern or logo on a piece of clothing. A sliding window of size 16×16 (the same as the training patch size) is then passed over the query image and votes are accumulated for each image in the training set.

To visualise the results, we back-project votes to the *support*. Back-projection is an established method of tracing the leaves of a forest backwards through the tree to establish the source data contributing to the leaf, and has previously been used for verification [17], meta-data transfer [18], and visualisation [19]. Back-projected votes are aggregated at the patch positions to create a heat-map of votes overlaid onto the image domain (Fig. 8). This is not an integral part of the algorithm, but enables a visual inspection of the accuracy of the pattern detection.

4 Experiments

We ran four experiments to examine the feasibility and accuracy performance of the new Hough Forests video database index. Three experiments were conceived to validate the concept without being distracted by the complexity of the search domain. In these, very simple synthetic images were constructed to train a forest and perform pattern detection in uncluttered images. We feel this is important as, although a toy example in pattern detection terms, it is used as a proof of concept for the new method of training a forest using the search domain and detection of an unseen query image. First is a very simple anti-aliased black text on a white background. Patches extracted from all the images shown in Fig. 5(a) were used as positive training samples and negative training images came from cartoon images from the *Garfield* and *Snoopy* categories of Ref. [20] and *104.homer-simpson* from Ref. [21]. Second, using the same images (Fig. 5(a)) training patches were extracted from all frames and selected as positive or negative training samples based on our algorithm (Eq. (2)). No additional negative training images were used. Third, a simple anti-aliased black text on a red-and-white high-contrast checkerboard background (Fig. 6(a)). While the first three experiments demonstrate the viability of the new use of Hough Forests and negative training method, using synthetic images, the fourth experiment tests a real-world use case. The forest is trained with 256 frames of a video sequence and detection of a pattern—an Adidas® logo on a jumper (Fig. 7)—during the

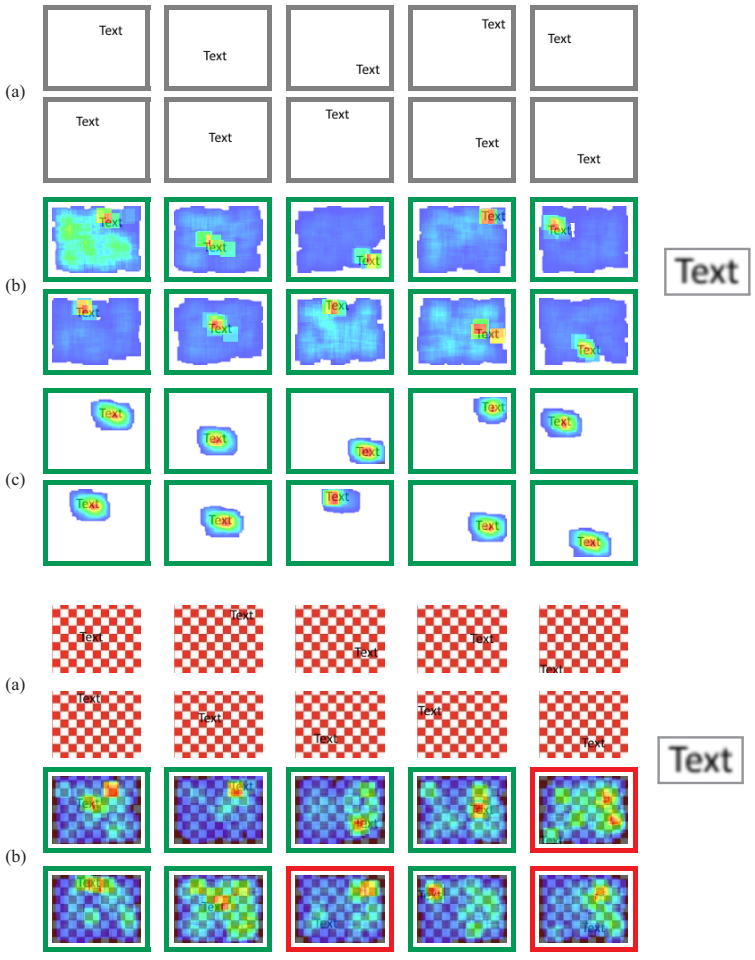


Fig. 5 (a) 10 images in the search corpus; (b) detection results using random patches through the image and negative training images from Refs. [20] and [21]; (c) detection results using a forest trained with random patches from the images and GLCM-Contrast selected positive/negative patches. Right, query image (shown with a border).

Fig. 6 (a) 10 images in the search corpus; (b) pattern detection results using a forest of five trees, each with a maximum depth of five. The coloured bordered represent successful (green) and unsuccessful (red) detection of the pattern. All patterns are detected successfully if the forest shape is changed to twenty trees, each with a maximum depth of ten.

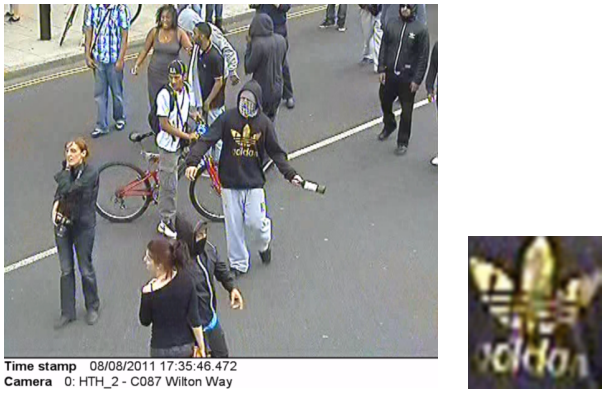


Fig. 7 The video frame (left) from which a query region (right, actual size) is selected to perform our fourth experiment.

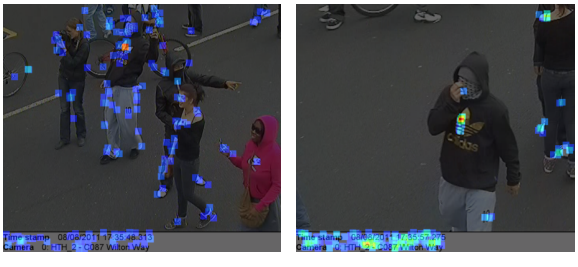


Fig. 8 Two frames of a video showing correctly identified positions of the search pattern. The offset hot-spot may be related to the training of the forest (see text).

London riots of 2011.

For each experiment, a forest of five trees was trained using the dataset as positive training images, extracting 200 positive training patches per image. The efficacy of pattern detection was then assessed by using a pattern that is known to be in the training data. This protocol mirrors our motivating use case.

5 Empirical results

In each of our result figures, coloured squares highlight the patch positions that contribute to a vote in favour of a pattern position. The colour reflects the accumulated vote from the patch, or series of overlaid patches, from blue (few votes) to red (many votes) on a heat map scale.

Detection of a piece of text on a plain white background is a simple enough task, but valuable

as a baseline experiment for a novel method such as ours. Figure 5(b) shows the correct detection of the text with some noise in the background (a perfect detection is a hot-spot in the centre of the text). We repeat the exercise using our low-contrast filtering method to select negative training patches from low-contrast areas of the images without using arbitrary negative training images, and the result is improved considerably (Fig. 5(c)) with hot-spots more central and no votes at all accumulated in the background away from the pattern area.

Results from the third test are shown in Fig. 6. The red-and-white checker-board background is high-contrast and the background is therefore not eliminated using our patch selection technique. Accuracy is measured by the position of the most intense red region, not by the size of the region. The intensity represents the number of votes cast, so a small intense red region depicts a higher probability of the centre of the pattern than a larger, less intense colour. This is evident in the first result image (top-left). A larger red/orange area has accumulated to the north-east of the pattern ground truth, but a smaller, more intense region shows on the right-hand edge of the letter **x**. The result is therefore determined a successful detection. In total, the pattern has been detected in seven out of the ten images. A successful detection in ten out of the ten images is achieved when the number of trees in the forest is increased to twenty, and the maximum depth of each tree is ten.

Figure 8 shows the successful detection of a sportswear company logo on clothing in two frames of a video sequence, from a realistically trained forest. The images have been darkened to highlight the results. The scatter of patches is restricted to high-contrast regions because of the training patch selection algorithm. The hot-spot (intensity red) is the algorithm's predicted centre of the logo. In both frames, the centre point is within the logo, but off-centre. However, consider the patch positions of the training image in Fig. 4(a); these randomly positioned patches only cover the top-left of the logo that was searched for. Although this is only one training frame, it indicates that the detection algorithm has not received enough good training patches for the selected query, and this is affecting the detection performance. Detection in some other frames performed less well (Fig. 9). The voting hot-spot is outside the ground truth position, accumulating on a different jumper's logo. Patches in the ground truth area are very sparse indicating that the forest has not been trained sufficiently.

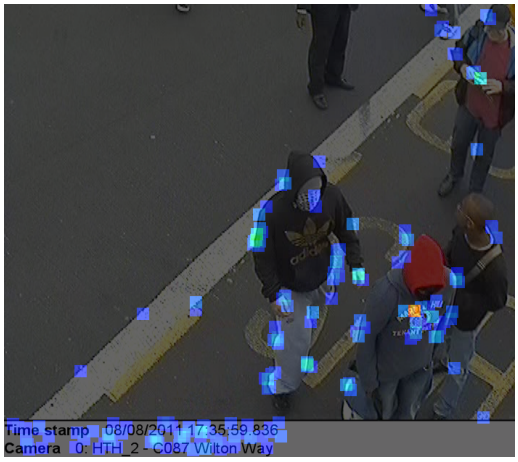


Fig. 9 An example of a false detection in an image where the query pattern is clearly visible. The voting hot-spot is on a jumper of another person, but the accumulated vote is lower showing less confidence in this detection.

6 Conclusions

This paper reports on new techniques for applying an established framework of randomised Hough Forests to large-scale pattern detection. By rethinking how a forest is trained and used in pattern detection, we have shown some initial results that validate the approach, and show promise of future success in applying the problem not only at large scale, but also to sequences of videos of complex scenes.

Training the forest effectively is an open research area to achieve good general pattern detection accuracy in cluttered street-scene images. Training with negative patches from low-contrast areas of the image works well in our experiments to date, but experience from textual retrieval systems suggest this may not always hold.

> *Studies of* [textual] *retrieval effectiveness show that all terms should be indexed . . . any visible component of a page might reasonably be used as a query term . . . Even stopwords—which are of questionable value for bag-of-words queries—have an important role in phrase queries* [22].

Relating this experience to the image search domain may suggest that low-contrast, and even background patches, should be included in the searchable forest and therefore used as positive patches for training.

Further experimentation is needed in this area. Open questions remain:

1. *How can the shape of the forest (number of trees, maximum depth, etc.) be determined per-video to maximise pattern detection accuracy?*

   All our experiments in this paper have used a consistent forest structure of five trees with a maximum depth of 5 levels, each trained using 200 positive training patches from each image. This structure generally performs well, but could theoretically be optimised based on the video image contents [15]. For example, videos containing more complex scenes should benefit from a larger forest (more trees) or more complex trees (greater depth) as experiment four demonstrates. The balance of runtime complexity, memory consumption and accuracy will be a trade-off consideration.

2. *Can a Hough Tree be learned incrementally without visibility of all its training data together?*

   One aspect of the scalability of the system to very large videos remains an open research area; To train a forest, the patches for all training data has to be accessible. This is by algorithm design and could present a scalability limit for videos with a large number of frame. The Hoeffding algorithm [23] describes a method to train a tree without visibility of all data, which may benefit our system. Some recent work has been published on incremental learning of forests for classification [24], and an opportunity exists to extend this to the problem of pattern detection.

3. *Can the granularity of forest and training data be better chosen to improve pattern detection accuracy?*

   Thus far we have used one forest per video in the corpus. This choice is made with the prior knowledge of properties of street-scene CCTV videos, for example that a video will be a sequence from a single camera without any shot change that temporally changes significantly between two consecutive frames of video [25]. While this choice is valid, in our view, it may not be optimal. We would like to investigate whether a single forest can be used to index multiple (perhaps related) videos, or if any videos can be temporally segmented and distributed to multiple forests. In such a system, each forest then becomes an index for a pre-classified set of video segments which may yield better detection performance.

As well as investigating these questions, a full assessment of the system, measuring accuracy against a ground truth is required in future work to produce a quantitative evaluation of our method.

Acknowledgements

This work is funded by the European Union's Seventh Framework Programme, specific topic "framework and tools for (semi-) automated exploitation of massive amounts of digital data for forensic purposes", under grant agreement number 607480 (LASIE IP project). The authors extend their thanks to the Metropolitan Police at Scotland Yard, London, UK, for the supply of and permission to use CCTV images.

References

[1] Gall, J.; Lempitsky, V. Class-specific Hough forests for object detection. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 1022–1029, 2009.

[2] Gall, J.; Yao, A.; Razavi, N.; Van Gool, L.; Lempitsky, V. Hough forests for object detection, tracking, and action recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 33, No. 11, 2188–2202, 2011.

[3] Breiman, L. Random forests. *Machine Learning* Vol. 45, No. 1, 5–32, 2001.

[4] Barinova, O.; Lempitsky, V.; Kholi, P. On detection of multiple object instances using Hough transforms. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 34, No. 9, 1773–1784, 2012.

[5] Payet, N.; Todorovic, S. Hough forest random field for object recognition and segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence* Vol. 35, No. 5, 1066–1079, 2013.

[6] Razavi, N.; Alvar, N. S.; Gall, J.; Van Gool, L. Sparsity potentials for detecting objects with the Hough transform. In: Proceedings of British Machine Vision Conference, 11.1–11.10, 2012.

[7] Srikantha, A.; Gall, J. Hough-based object detection with grouped features. In: Proceedings of IEEE International Conference on Image Processing, 1653–1657, 2014.

[8] Yokoya, N.; Iwasaki, A. Object detection based on sparse representation and Hough voting for optical remote sensing imagery. *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing* Vol. 8, No. 5, 2053–2062, 2015.

[9] Godec, M.; Roth, P. M.; Bischof, H. Hough-based tracking of non-rigid objects. *Computer Vision and Image Understanding* Vol. 117, 1245–1256, 2013.

[10] Rematas, K.; Leibe, B. Efficient object detection and segmentation with a cascaded Hough Forest ISM. In: Proceedings of IEEE International Conference on Computer Vision Workshops, 966–973, 2011.
[11] Waltisberg, D.; Yao, A.; Gall, J.; Van Gool, L. Variations of a Hough-voting action recognition system. In: *Lecture Notes in Computer Science, Vol. 6388*. Ünay, D.; Çataltepe, Z.; Aksoy, S. Eds. Springer Berlin Heidelberg, 306–312, 2010.
[12] Yao, A.; Gall, J; Van Gool, L. A Hough transform-based voting framework for action recognition. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 2061–2068, 2010.
[13] Yu, G.; Yuan, J.; Liu, Z. Unsupervised random forest indexing for fast action search. In: Proceedings of IEEE Conference on Computer Vision and Pattern Recognition, 865–872, 2011.
[14] Gall, J.; Razavi, N.; Van Gool, L. An introduction to random forests for multi-class object detection. In: Proceedings of the 15th International Conference on Theoretical Foundations of Computer Vision: Outdoor and Large-scale Real-world Scene Analysis, 243–263, 2012.
[15] Henderson, C.; Izquierdo, E. Minimal Hough forest training for pattern detection. In: Proceedings of International Conference on Systems, Signals and Image Processing, 69–72, 2015.
[16] Dalal, N.; Triggs, B. Histograms of oriented gradients for human detection. In: Proceedings of IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Vol. 1, 886–893, 2005.
[17] Leibe, B.; Leonardis, A.; Schiele, B. Robust object detection with interleaved categorization and segmentation. *International Journal of Computer Vision* Vol. 77, No. 1, 259–289, 2008.
[18] Thomas, A.; Ferrari, V.; Leibe, B.; Tuytelaars, T.; Van Gool, L. Using multi-view recognition and metadata annotation to guide a robot's attention. *The International Journal of Robotics Research* Vol. 28, No. 8, 976–998, 2009.
[19] Razavi, N.; Gall, J.; Van Gool, L. Backprojection revisited: Scalable multi-view object detection and similarity metrics for detections. In: *Lecture Notes in Computer Science, Vol. 6311*. Daniilidis, K.; Maragos, P.; Paragios, N. Eds. Springer Berlin Heidelberg, 620–633, 2010.
[20] Li, F.-F.; Fergus, R.; Perona, P. Learning generative visual models from few training examples: An incremental Bayesian approach tested on 101 object categories. *Computer Vision and Image Understanding* Vol. 106, No. 1, 59–70, 2007.
[21] Griffin, G.; Holub, A.; Perona, P. Caltech-256 object category dataset. Technical Report. California Institute of Technology, 2007. Available at http://authors.library.caltech.edu/7694/1/CNS-TR-2007-001.pdf.
[22] Zobel, J.; Moffat, A. Inverted files for text search engines. *ACM Computing Surveys* Vol. 38, No. 2, Article No. 6, 2006.
[23] Domingos, P.; Hulten, G. Mining high-speed data streams. In: Proceedings of the 6th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 71–80, 2000.
[24] Ristin, M.; Guillaumin, M.; Gall, J.; Van Gool, L. Incremental learning of random forests for large-scale image classification. *IEEE Transactions on Pattern Analysis and Machine Intelligence* DOI: 10.1109/TPAMI.2015.2459678, 2015.
[25] Henderson, C.; Blasi, S. G.; Sobhani, F.; Izquierdo, E. On the impurity of street-scene video footage. In: Proceedings of the 6th International Conference on Imaging for Crime Prevention and Detection, 1–7, 2015.

Craig Henderson received his B.S. degree in computing for real time systems from the University of the West of England in Bristol, UK, in 1995. From 1995 to 2014 he worked in a variety of organisations as software engineer and engineering manager.

Since 2014, he is a Ph.D. candidate in the Multimedia and Computer Vision Laboratory, School of Electronic Engineering and Computer Science at Queen Mary University of London, UK. His research interests include computer vision, machine learning, and scalable systems.

Ebroul Izquierdo received his M.S., Ph.D., C.Eng., FIET, SMIEEE, MBMVA degrees. For his thesis on the numerical approximation of algebraic-differential equations, he received the Dr. Rerum Naturalium (Ph.D.) degree from Humboldt University, Berlin, Germany.

He is the head of the Multimedia and Vision Group, School of Electronic Engineering and Computer Science at Queen Mary University of London, UK. He has published over 500 technical papers including book chapters and holds several patents.