Robust Feature Matching in the Wild

Henderson, C; Izquierdo, E; Science and Information Conference

Robust Feature Matching in the Wild

Craig Henderson
Multimedia and Vision Group
Queen Mary University of London
Email: c.d.m.henderson@qmul.ac.uk

Ebroul Izquierdo
Multimedia and Vision Group
Queen Mary University of London
Email: ebroul.izquierdo@qmul.ac.uk

***Abstract*—Finding corresponding key points in images from security camera videos is challenging. Images are generally low quality and acquired in uncontrolled conditions with visual distortions caused by weather, crowded scenes, emergency lighting or the high angle of the camera mounting. We describe a methodology to match features between images that performs especially well with real-world images. We introduce a novel *blur sensitive feature detection* method, a combinatorial feature descriptor and a distance calculation that efficiently unites texture and colour attributes to discriminate feature correspondence in low quality images. Our methods are tested by performing key point matching on real-world security images such as outdoor CCTV videos, and we demonstrate an improvement in the ability to match features between images compared with the standard feature descriptors extracted from the same set of feature points. We use key point features from Harris Corners, SIFT, SURF, BRISK and FAST as well as MSER and MSCR region detectors to provide a comprehensive analysis of our generic method. We demonstrate feature matching using a 138-dimensional descriptor that improves the matching performance of a state-of-the-art 384-dimension colour descriptor with just 40% of the storage requirements.**

***Keywords*—Feature extraction; Pattern matching**

Fig. 1: Typical images from a single CCTV camera, with poor lighting and long range camera views. When a subject appears in the distance, the colour and texture definition is poor and inconsistent with frames when the subject is closer.

I. Introduction

Feature descriptors and inter-image feature matching have been well researched areas in computer vision for many years. Most works assess the performance of descriptor matching using high quality images; for example, in the field of video analysis, popular techniques have used Hollywood movies as a test dataset [1], [2], [3]. However, security cameras work in uncontrolled environments and record constantly without continual adjustment to focus, lighting and position that a feature film is privileged with. As a result, the low resolution images generally have poor colour clarity and little discriminative or representative texture definition (Figure 1).

Fluctuating lighting conditions caused by fire and emergency vehicle lights are commonplace in video that undergoes forensic analysis. Fast camera pan or zoom, frenzied motion within a frame, or a combination of both can cause significant blurring in frame images which results in a lack of texture. Closed-circuit television (CCTV) cameras are often sited very high and cover a long field of view where objects in the distance lack colour definition. Consequently, contemporary methods are not robust to the challenges of low quality images that result from these systems. Forensic analysis of security camera video sequences is a less well studied field and demands adaptation of contemporary methods to accommodate the image quality differences that exist. The quality of images from each security camera varies considerably, and this inconsistency can cause difficulties in matching features between camera images.

Our intent is to match distinctive regions or patterns across frames, such as a brand logo, a coloured pattern on a scarf or hat, a tattoo or other distinctive marking on a person, object or clothing. Texture alone is not sufficient to find correspondences between frames in security videos, and therefore the effectiveness of popular gradient-based descriptors such as SIFT and SURF is limited. Our interest is in large-scale processing of long-running videos which demand fast processing of a very large number of features. We are therefore motivated by simple solutions to complex problems with low computation requirements and minimal storage, intentionally avoiding some of the complexities of other methods in the interest of execution speed.

In this paper we establish a method to improve the robustness of matching features by using image blur metrics and colour information to increase discriminative properties of texture feature descriptors. Our contributions are simple and fast algorithms that combine to provide memory- and processing-efficient feature matching with which we demonstrate improvements over current methods that use Euclidean distance to match intensity- and colour-feature descriptors:

Adaptive blur-sensitive feature detection An adaptive approach to the detection of features that will correspond between

two images, guided by the sharpness of the two images.

Combinatorial Texture and Colour feature matching A novel technique to combine texture and colour features and measure distance between descriptors for robust feature matching.

The rest of the paper is structured as follows. §II provides an overview of related literature on colour features descriptors, the proposed methodology is described in detail in §III and evaluated in §IV. §V assesses the storage efficiency and accuracy trade-off that is important in large-scale systems, and we conclude in §VI.

II. Related Work

Popular feature detectors such as SIFT [4], SURF [5] and the Harris Corner Detector [6] were designed to find texture in grey scale images, and the feature descriptors of SIFT and SURF are defined only by pixel intensity variations. There have been a number of proposals for colour descriptors that describe colour attributes of an image. These are conveniently small in dimensionality (30 to 45) and represent the colour information around a key point using a colour histogram. A detailed description of histogram based colour descriptors is provided in [7].

Colour alone is not robust for achieving good correspondences between images. There have been many descriptors proposed that use texture descriptors with various colour channel combinations, combining the texture from each channel. Many of these are based on the SIFT descriptor resulting in a $128 \times 3 = 384$ dimension descriptor. HSV-SIFT [8] calculates a SIFT descriptor on each of the three channels in HSV colour space and RGB-SIFT [9] is a similar algorithm using RGB channels, with values equal to the *Transformed Colour SIFT* method [9]. rgSIFT [7] builds descriptors on the r and g chromacity components of the normalised RGB colour model and C-SIFT [10] uses a normalised opponent colour space, dividing the first two channels by the intensity channel O_3, to make it invariant with respect to light intensity [7]. OpponentSIFT identifies features in opponent colour channels, red-green (RG) and yellow-blue (YB) [11] by computing SIFT descriptors in each of them. OpponentSURF uses the same technique with SURF features. The interested reader is referred to [9] for a comprehensive review of colour descriptors.

In each of these, colour information is used in detecting features and extracted descriptors are implicitly discriminative by virtue of their construction. However, the colour detail of the image area around the feature is not encoded into the descriptor and is not used to discriminate between similar features. HueSIFT [12] describes a concatenation of a quantised Hue Histogram of 37 dimensions with the SIFT descriptor, concentrating on the effective detection of features without consideration for the descriptor encoding. Our method takes a similar approach in descriptor concatenation, but we do not limit our focus on SIFT descriptors and we describe a robust approach to feature distance calculations.

Colour descriptors that use three colour channels for feature descriptions typically increase the dimensionality three times, compared with their intensity based counterparts, and the size of each descriptor becomes problematic for efficient computation and storage. Principal Component Analysis (PCA) has been used to reduce the dimensionality in PCA-SIFT [13], but is computationally expensive. A method of fast PCA calculation has been suggested [14], which finds the desired number of leading eigenvectors using less computation, but with a slightly larger mean-squared error.

III. Proposed Methodology

A. Blur sensitive feature detection

Image blur is a very significant hindrance to matching features between frames in low quality images. We use this observation to adapt feature detection to maximise correspondence accuracy in a technique we call *blur sensitive feature detection*.

1) Measuring image blur: Accurate models for calculating the motion blur of an image have been described, from estimating the parameters of a Point Spread Function [15] to using machine learning [16]. Our intent is not to accurately calculate the blur parameters such that the blurred image can be restored to a sharp image, but to quickly be able to estimate the degree to which an image, or part of an image, is blurred. We therefore use a straightforward method that is fast to calculate and is shown to give a reasonable estimation of blurriness for our purposes.

We derive an efficient technique from the intuition that a blurred image will contain fewer sharp edges than a non-blurred image. The number of edges in an image can therefore be used as an expression of image blurriness (or, conversely, image sharpness). We use a Canny edge detector [17] with a 3×3 Gaussian kernel, a lower threshold of 175 and an upper threshold of 225. The small Gaussian kernel balances execution time with sensitivity salt-and-pepper noise that can be caused by analog-to-digital converter errors or bit errors in transmission. The threshold values have been chosen empirically to avoid breaking noisy edges (if the lower threshold is too high) and reducing fragmentation if the upper threshold is too low.

The Canny edge detector [17] is used to construct a binary edge map E from image I of size $m \times n$,

$$E_I = \{e_1, e_2, e_3, \dots e_{m \times n}\}, \quad e_i \in \{0, 1\} \tag{1}$$

The *sharpness* of image I is then determined by a function $S(I)$ that calculates the fraction of non-zero pixels in the edge map E, which is the fraction of pixels representing edges in the image I.

$$S(I) = \frac{1}{m \times n} \sum_{x_i \in E_I} x_i \tag{2}$$

The determination of a *blurred image* is then to find a suitable threshold below which $S(I)$ must fall to represent an image that is blurred. Our definition of a blurred image is therefore;

$$\text{image } I \text{ is blurred} = \begin{cases} true & \text{if } S(I) \leqslant \lambda \\ false & \text{otherwise} \end{cases} \tag{3}$$

2) Adaptive feature extraction: A relationship map M_s is established to correspond the properties of a 2D Gaussian kernel G_k of size k with the sharpness measure of the query image I_Q after a convolution with G_k. The map holds

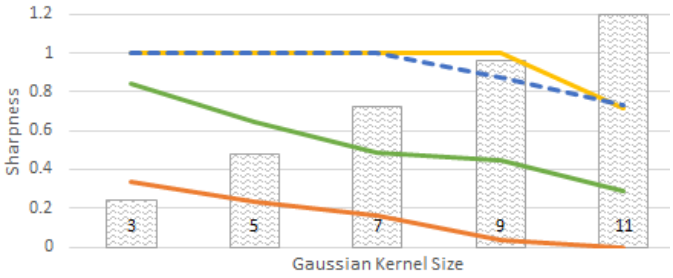

Fig. 2: Relationship between the size of a Gaussian kernel (x-axis) used to artificially blur example query images (*curves*) and the sharpness of the resulting image (y-axis). The kernel size steadily increases while the decline in sharpness (increase in image blur) varies with different query image regions. The relationship map M_s therefore needs to calculated for each query region used for correspondence matching.

sharpness values for the query image after convolution with 2D Gaussian filters of kernel size p. Let

$$\Gamma \equiv \{p \in \mathbb{N} | p = 2q - 1 \wedge q \in \mathbb{N}\} \tag{4}$$

The sharpness is calculated for each kernel size and stored in an associative map $k \rightarrow S(I)$

$$M_s(k) = S(G_k * I_Q) \quad \forall k \in \Gamma \wedge k \leq \alpha \in \mathbb{N} \tag{5}$$

where

$M_s(\cdot)$ represents an entry in an associative map
$S(I)$ image sharpness, from Equation (2)
G_k Gaussian filter of kernel size k
$*$ represents 2D convolution
α an upper bound on the Guassian kernel size

Figure 2 shows some examples of the relationships between the size of a Gaussian kernel used to artificially blur example query images and the sharpness of the resulting image in M_s. This demonstrates the variance in the correlation between the steepness in the decline in sharpness (increase in image blur) with steadily increasing kernel sizes for different query image regions, and therefore the need to calculate M_s for each query region used for correspondence matching.

A *sharpness adjustment* S_a is calculated as the difference between the sharpness of the original (unconvolved) query image region I_Q and the target image I_T to which correspondence is to be established.

$$S_a = S(I_Q) - S(I_T) \tag{6}$$

The value of S_a is used to find the corresponding Gaussian kernel size k in M_s which, when convolved with I_Q will produce an image I'_Q with sharpness that will most closely match $S(I_T)$

$$m = \arg\max_k \{S_a - S(G_k * I_Q)\} \quad m \geq 0 \tag{7}$$

$$I'_Q = G_k * I_Q \tag{8}$$

Features are detected in, and extracted from I'_Q and I_T and correspondences are found between these feature sets.

Matching performance is considerably improved by aligning sharpness of the images before feature detection. However, blurring an image reduces texture structure, which reduces the effectiveness of feature detectors, especially corner-based detectors such as FAST and BRISK. If no features are found in I'_Q, we repeat the process with I_Q as the entire query image, not bounded to the query region of interest. We do this with the understanding that the bounded region of interest contains little texture so retrying with greater kernel sizes would offer only minor improvements, whereas the sharpness of the query image as a whole provides more information with respect to camera movement induced blur. In the unlikely event that no features are found in the revised I'_Q, we fall back features found in I_Q. In all of our experiments, this fall back position is never required as the unbounded I_Q image always produces a usable feature set.

B. Combinatorial Texture and Colour feature matching

We create a new combinatorial feature descriptor representing local key point features with colour information from the surrounding region. First, any local feature detector is used to find feature locations and both a key point and a region are defined for each. In the case of a key point detector such as SIFT, a circular region is created with its center at the key point co-ordinates. For region based feature detectors such as *Maximally Stable Extremal Regions* (MSER), the region is approximated using an ellipse fitting algorithm through the region boundary points and a key point is defined at the center of the ellipse.

With the resulting set of key point locations and region definitions, we extract a texture descriptor at each key point. The texture descriptor is a standard feature descriptor that will be extended by our method to improve its discriminative capability in colour images. We then build a local histogram colour model of each region to create an extension descriptor. Using the region shape as a mask over the colour image, pixels falling within the shape are quantised into a local colour histogram representing the region. This histogram is transformed into a feature descriptor using the histogram bins to form the descriptor values. Finally, the texture descriptor and colour descriptor are concatenated into a composite descriptor.

The RGB colour space is known to be a poor representation for colour segmentation as there is no straightforward correlation between the RGB channel values and the intensity of a particular colour that lends itself to simple thresholding. We therefore transform the RGB image to the HSV colour space for our algorithm. The Hue (H) channel determines the colour, the Saturation (S) is the intensity of the colour and the brightness or luminance (V) can be used to find non-colour white, grey and black.

In counting colours, we use a standard quantisation of pixel values to their closest histogram bin by calculating a partial distance in HSV colour space. For colour entries in the histogram, the distance is determined by the Euclidean distance of the Hue and Saturation components, $d = \sqrt{H_i^2 + S_i^2}$. Distance to the additional three non-colour entries in the histogram – white, black and grey – are calculated using the Euclidean distance of the Saturation and Value (luminance) components, $d = \sqrt{S_i^2 + V_i^2}$. Measuring colour distances in

the HSV colour space in this way maintains robustness against affine illumination changes in the image.

1) Texture descriptor distance: Two features are considered equal if they are close to each other in their high-dimensional feature space. Popular feature descriptors are designed as Euclidean space vectors such that $\vec{u}$ and $\vec{v}$ represent features

$$\begin{aligned}\vec{u} &= (u_1, u_2, \ldots, u_n)\\ \vec{v} &= (v_1, v_2, \ldots, v_n)\end{aligned} \tag{9}$$

and the distance between them is given by the length of the line segment $\overline{\mathbf{uv}}$ connecting them, i.e. the Euclidean norm.

$$\|\overline{\mathbf{uv}}\|_2 = \|\vec{u} - \vec{v}\|_2 = \sqrt{\left(\sum_{i=1}^{n}(u_i - v_i)^2\right)} \tag{10}$$

A pre-defined or dynamically calculated threshold value is typically used to determine whether features are *close enough* to be considered a reasonable match.

2) Colour histogram distance: Colour histogram feature space is also multi-dimensional, but the distance between points in most colour spaces are more accurately calculated using methods such as χ^2 [19], Bhattacharyya distance [20] or the Earth Mover's Distance [21]. Our histograms have identical palettes and we want a fast calculation of distance between two histograms. We therefore elect to use the Normalised Histogram Intersection (NHI) [22], which is a light-weight calculation of similarity, and subtract the result from 1 to give a dissimilarity, or a distance measure between two histograms.

$$H(a,b) = 1 - \frac{\sum_{j=1}^{n}\min(a_j, b_j)}{\sum_{j=1}^{n} a_j} \tag{11}$$

3) Designing a combinatorial descriptor: In designing an algorithm to extend an existing feature descriptor, consideration is made to the potential of falsely matching dissimilar features of similar colour, or moving vectors in feature space closer together where neither their feature descriptor nor colour are similar. Our goal is to produce a generic extension that can be used with any underlying texture feature descriptor. We therefore focus on a method to combine an n_1-dimensional texture feature descriptor with an n_2-dimensional colour histogram in such a way as to discriminate similar features of different colours without these pitfalls.

Consider a naïve implementation that concatenates an n_2-dimensional colour-histogram onto a n_1-dimensional texture descriptor to form an (n_1+n_2)-dimensional feature descriptor, and compares the combined descriptors as single vectors. Extending the vector dimensionality to include colour information is intuitive, but flawed. This method will treat the colour histogram as an integral part of the feature, and apply Equation (10) to the vector as a whole. The unique properties of the colour histogram will be lost. Figure 3 shows the correlation between using *Euclidean distance* and *histogram similarity* to measure the closeness of features from our dataset. The low positive correlation value of 0.49 shows that a Euclidean distance will generally give a reasonable indicative result but is less accurate than the histogram similarity $(1-\text{NHI})$.

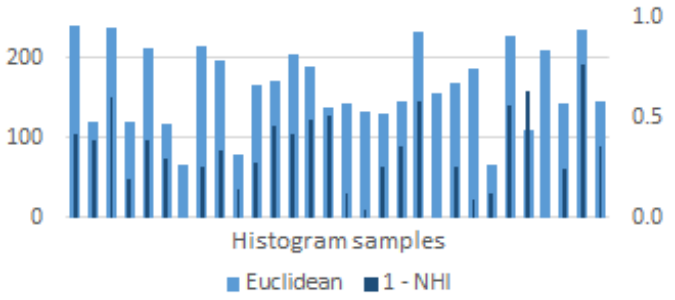

Fig. 3: Correlation of the distance between colour histograms using the Euclidean distance (left axis) and a similarity based on Normalised Histogram Intersection (right axis). Correlation ≈ 0.49 indicates a low positive correlation between the two measures.

4) Distance between descriptors: A feature can be said to correspond to its closest match in a set of candidate features, where the descriptor with the smallest distance is selected, irrespective of the value of the distance or its relationship to its neighbors. Lowe [4] refined this method using a *distance ratio* to determine if the closest match was a *good* match. The distance ratio method finds the closest two features f_c and f_{c+1} and divides the nearest distance by the second closest distance,

$$distance\ ratio = \frac{\|f - f_c\|_2}{\|f - f_{c+1}\|_2} \tag{12}$$

This ratio helps to determine how reliable the match is. If the nearest feature has another feature close to it, then there is a lesser likelihood that the match is correct. Tests in the original paper suggest that 0.8 is a reasonable threshold for this ratio, based on analysis of $40,000$ key points, and that matches with a distance ratio greater than 0.8 should be considered less reliable, thus,

$$match = \begin{cases} true & \text{if } \frac{\|f-f_c\|_2}{\|f-f_{c+1}\|_2} \leqslant 0.8 \\ false & \text{otherwise} \end{cases} \tag{13}$$

We follow this understanding in our method and use the colour information of both features to scale the distance between their descriptors. In doing this, a metric of the difference in the colour histograms logically moves the features apart, extending the line segment $\overline{\mathbf{uv}}$.

5) Distance Definition: The composite feature descriptor $\mathbf{f}$ is conveniently represented as a single n-dimensional vector, where n the sum of the lengths of the texture $\vec{t}$, and histogram $\mathbf{h}$.

$$\mathbf{f} = (\vec{t}, h) \tag{14}$$

In calculating the distance D between two composite descriptors, we first consider a distance measure between each of the two parts independently, d_1 and d_2, and combine the results. The texture descriptor distance d_1 is a standard calculation of the Euclidean distance between the two vectors

and d_2 is the distance between the two colour histogram descriptors, $H(\cdot)$ from Equation (11).

$$d_1 = \left\| \vec{t_1} - \vec{t_2} \right\|_2 \tag{15}$$

$$d_2 = H(\mathbf{h}_1, \mathbf{h}_2) \tag{16}$$

The individual distance measures d_1 and d_2 are then combined to yield a representative distance between the two composite descriptors. A simple sum $D = d_1 + d_2$ does not account for the difference in scale within each of the descriptors, which itself will be different depending on the choice of texture descriptor. The product $D = d_1 \times d_2$ down-scales the texture distance based on the colour histogram distance, effectively moving similar texture descriptors closer together. This reduces the discrimination of similar textual descriptors, increasing the number of mis-matches and reducing the overall accuracy. We derive a composition applying a multiplier to the normalised histogram distance and summing with the texture distance. In general form,

$$D = d_1 + \lambda d_2 \tag{17}$$

The selection of a suitable value for λ has been the subject of many experiments. Any empirically chosen constant value is not robust for the variety of challenging images from surveillance video images, and we therefore look to a dynamic value for λ which represents the conditions within which the feature appears.

Using d_2 as a value for λ reduces the impact of the colour histogram distance because d_2 is a normalised value, which when multiplied by itself becomes smaller, and overall less discriminative. However, d_1 is a good candidate. With $\lambda = d_1$, the colour distance is used to scale the distance measure of the texture descriptor so that it discriminates between similar descriptors of different colours.

$$\begin{aligned} D &= d_1 + d_1 d_2 \\ &= d_1(1 + d_2) \end{aligned} \tag{18}$$

We see from Equation (18) that with $\lambda = d_1$ we apply the colour distance measure as a scalar to the distance between two texture feature descriptors. Increasing the normalised value of d_2 from the range $0 \ldots 1$ into $1 \ldots 2$, thus upscaling the distance of a texture feature by multiplication. The overall distance between two composite descriptors is therefore

$$\begin{aligned} D &= d_1(1 + d_2) \\ &= \left\| \vec{t_1} - \vec{t_2} \right\|_2 \times \left(2 - \frac{\sum_{j=1}^{n} \min(a_j, b_j)}{\sum_{j=1}^{n} a_j} \right) \end{aligned} \tag{19}$$

The use of a scalar applied to the texture descriptor distance ensures that attributes of the texture descriptor such as invariance to affine scale and rotation transformations, are preserved. The calculation of the colour histogram in Hue and Saturation channels maintains invariance in affine illumination transformations.

To find the closest descriptor D_c to a given descriptor D_i it is customary to use an algorithm based on Euclidean distance, such as k-Nearest Neighbour. We perform a nearest descriptor calculation in two parts. First, the k-nearest neighbours of the texture descriptor $\vec{t}$ are found using the standard algorithm with $k = 5$, giving $\{\vec{v_1}, \vec{v_2}, \vec{v_3}, \vec{v_4}, \vec{v_5}\}$. For each of the five closest descriptors, we perform the scaling multiplication of Equation (19) and determine the descriptor with the smallest resulting distance to be the closest, D_c

$$D_c = \arg\min_i \{D_{v_i}\} \tag{20}$$

This is not guaranteed to be optimal, but in our tests increasing k to 10 does not improve the result. The *Approximate Nearest Neighbour* algorithm (ANN) is commonly used to reduce computational complexity in a k-Nearest Neighbour search. ANN uses a randomised indexing method making the result non-deterministic, but is widely accepted for many matching tasks. Our calculation has not shown to produce a worse approximation in our tests, and in deterministic in its result.

IV. Experimental Evaluation

We evaluate the performance of the proposed descriptor by measuring the accuracy of matching features between pairs of images. The definition of a feature match depends on the matching strategy that is applied [23]. Our intention is to measure the accuracy of our new composite feature descriptor and distance calculation. We therefore compare our results with a nearest neighbour matching algorithm without any threshold filtering, such as Nearest Neighbour Distance Ratio to discard poor matches.

We use seven feature detectors to find initial regions of interest. Five popular intensity based key point detectors; Harris Corners detector (HARRIS), SIFT, SURF, BRISK and FAST, and two region detectors; MSER on grey scale representations and *maximally stable colour regions* (MSCR) on colour images. For each of these sets of features, we compare feature matching performance of descriptors extracted using SIFT and SURF, with and without our combinatorial descriptor, and later using OpponentSIFT and OpponentSURF 3-channel descriptors, again with and without our combinatorial descriptor.

The key point detectors HARRIS, SIFT and SURF are chosen because of their popularity and widespread adoption in many tasks including object classification and image retrieval, and BRISK and FAST for their high performance and relevance for real-time processing. We are keen to demonstrate the universal improvements of our method and therefore also include region based detectors in our comparisons, MSER and MSCR.

A. Blur sensitive feature detection

We evaluate the *blur sensitive feature detection* technique independently using our seven selected feature detectors with state-of-the-art descriptors and Euclidean distance measurements. In our experiments, we use an empirical value $\lambda := \frac{1}{32}$ in Equation (3), so if edges are present in 3.125% of the image or less, then the image is deemed to be blurred. Our sharpness map contains convolutions with Gaussian kernels up to 11×11, thus $\alpha := 11$ in Equation (5).

Figure 4 shows the percentage improvements in matching quality achieved by applying the blur sensitive feature detection algorithm to our test database. The matching accuracy improvement is subject to the choice of feature detector, which is expected because the artificial blurring of the image will

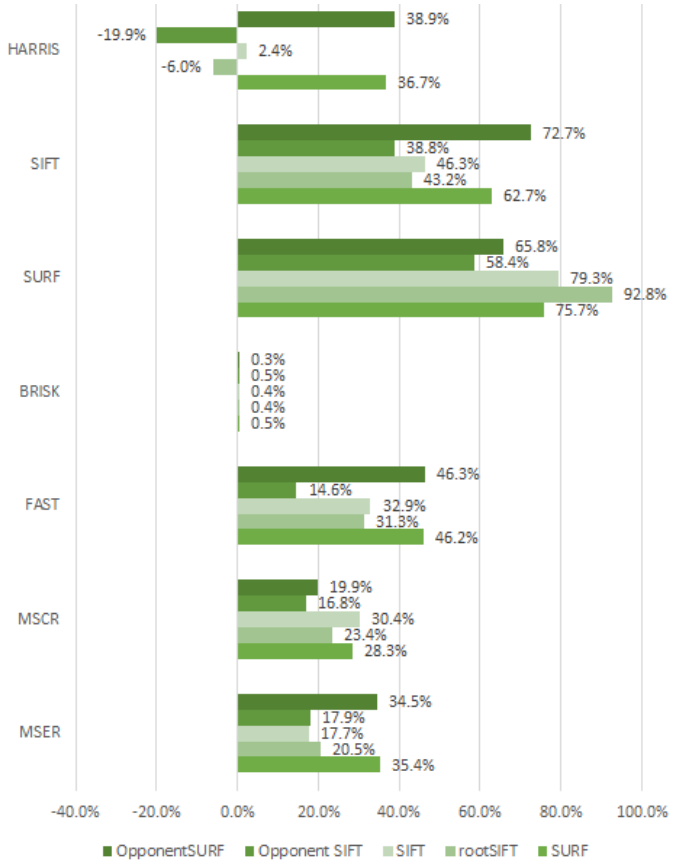

Fig. 4: Matching accuracy *blur sensitive feature detection*. Matching accuracy improvement is subject to the choice of feature detector and performance is broadly consistent across all extractors for each detector. However, Harris Corner features vary considerably for each descriptor type, and decreases matching performance in two cases.

effect each detector differently, and the matching performance is broadly consistent across all extractors for each detector. The exception are Harris Corner features which vary considerably for each descriptor type, and decreases matching performance in two cases; rootSIFT and OpponentSIFT descriptors. BRISK features yielded consistently low improvements, and matching SURF features was generally more improved, with rootSIFT descriptors extracted from SURF key points being improved the most, by 92.8%.

B. Combinatorial descriptor assessment

We use a fixed colour histogram for all images. In experiments, the 10-bin palette of Park *et al.* [18] has proven to work well; seven colours and three special considerations for intensities (Table I). This palette has been used for the experiments presented in this paper. The descriptor extension is therefore 10 dimensions in size.

1) Query by example: A rectangular area of an image is specified as a query region containing features that are to be matched in subsequent frames of the video sequence. In our first test the query region represents a distinctive two-colour back-pack being worn by a person (Figure 6 and Figure 7).

Colour	H	S	V
Red	0°	100%	100%
Brown	15.1°	74.5%	64.7%
Yellow	60°	100%	100%
Green	120°	100%	100%
Blue	240°	100%	100%
Violet	300°	45.4%	93.3%
Pink	349.5°	24.7%	100%
White	0°	0%	100%
Black	0°	0%	0%
Grey	0°	0	60%

TABLE I: Colour palette from [18] used in our experiments

This region is matched against 250 video frames, each of which has ground truth information defining the boundaries of the back-pack within it.

Descriptors are created for the query image region and each image under consideration (*candidate images*) using the method described above. The positions of the features within the candidate image that match with the query region are then assessed relative to the ground truth and determined to be a true or false positive result or a negative result. A true positive result is a feature that matched with the query region (a *query match*) lies within the ground truth region. If a query match falls outside the ground truth then the region is labeled as false negative result. A feature matched between the images from outside the query region that falls within the ground truth region is counted as a false positive result. A match between the images from outside the query region to outside the ground truth region is not used directly within our analysis but are implicitly relative to other metrics.

Results for each feature are tallied for each image, and these are then summed across all of the images in the sequence. The true positive tp, false positive fp and false negative fn totals for the images are then used to calculate the recall and precision measures of performance of each of the four descriptors with and without our extension;

$$recall = \frac{tp}{tp + tn} \tag{21}$$

$$precision = \frac{tp}{tp + fp} \tag{22}$$

In reporting our results we use the F-measure, the weighted harmonic mean of recall and precision, to measure and compare the accuracy of our combinatorial descriptor and distance measure with well-known descriptors. We favour neither precision nor recall over the other, and therefore use the F_1 score, defined as

$$F_1 = \frac{(2 \times recall \times precision)}{(recall + precision)} \tag{23}$$

2) Intensity descriptors: It is important to compare the feature matching performance with popular intensity descriptors because these have the smallest dimensionality. In a large-scale processing system, size of descriptors is important for minimizing memory and disk storage and data processing time.

Our experiments compared the matching performance of SIFT and SURF descriptors against our combinatorial descriptor based on SIFT and SURF with our distance measure,

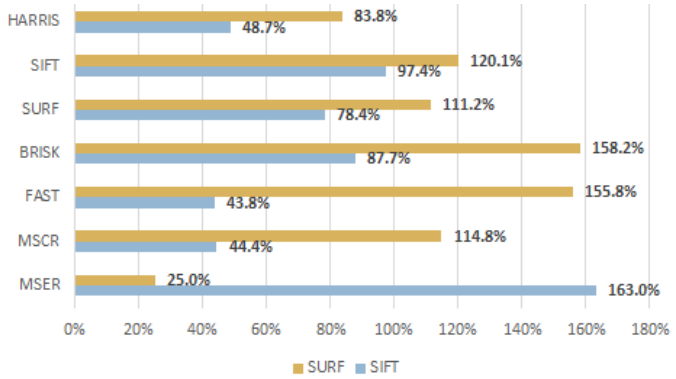

Fig. 5: Improvement of SIFT and SURF intensity descriptors using our combinatorial descriptor and distance measure. Orange bars show percentage improvements of SURF descriptors using our method, and blue bars show improvements in SIFT. The baseline uses standard descriptors with Euclidean distance measures in feature space. The overall average improvement across all of the feature descriptors in this test was 95.2%.

for features detected using Harris Corners (HARRIS), SIFT, SURF, BRISK, FAST, MSCR and MSER (Figure 5). Feature matching is determined by the nearest neighbour feature in descriptor space. The greatest improvement was achieved with SIFT descriptors extracted from MSER features where the F_1 measure increased by 163% using our method (from 0.064 to 0.167) compared to a plain SIFT descriptor on the same MSER features.

Overall, the average improvement across all of the feature descriptors in this test was 95.2%.

Figure 6 shows two examples of matching feature descriptors from a region of interest within a query image to a subsequent frame in a surveillance video, using a SURF feature detector. The top images show matches of SURF descriptors extracted from the SURF features within the region of interest in the query image (left), and a blurred frame (right). There is a notable increase in the number of features matched into the bag region in the right hand image. The bottom images show matches from the same query frame to a sharper subsequent frame and demonstrates the reduction in false-positive matches into background features. The less cluttered Figure 7 repeats the second image pairs from Figure 6 using the distance ratio filter (Equation (13)) from [4]. There are new positive matches in both images, matching points within the rucksack that are not matched in the top row. In addition, the number of false positives is visibly reduced, with fewer yellow lines matched to the background in the right-hand images.

3) Colour descriptors: We now assess our algorithm using two high-dimensional colour descriptors, OpponentSIFT (384-dimensions) and OpponentSURF (192-dimensions), with the same features from the previous section.

The F_1 measure on our test video sequence is improved using colour descriptors over using the intensity texture descriptors. This is to be expected as the colour information provides a more discriminative comparison. In our test video sequence, the best match performance was achieved using

Fig. 6: Two examples of matching SURF features on a coloured bag from a query frame (left in each pair) to a subsequent video frame. Top, matches to a blurred image perform poorly using Approximate Nearest Neighbour (blue matches) and a significant increase in matches to the target bag using our method (yellow matches). Bottom demonstrates the significantly reduced number of false positive matches to the background using our method (yellow) compared with ANN matching (blue).

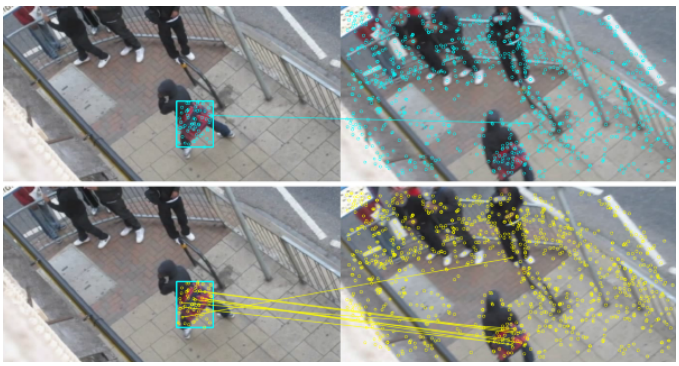

Fig. 7: *Good matches* – Equation (13) – are shown for SURF features (top row) and using our method (bottom row).

the combinatorial OpponentSIFT descriptor with FAST features, achieving an improvement of 12% over the original OpponentSIFT descriptors' accuracy of 0.415. Matching OpponentSURF descriptors of FAST features was improved the most of all colour descriptors, by 98.5% but the F_1 score is low, increasing from just 0.149 to 0.296.

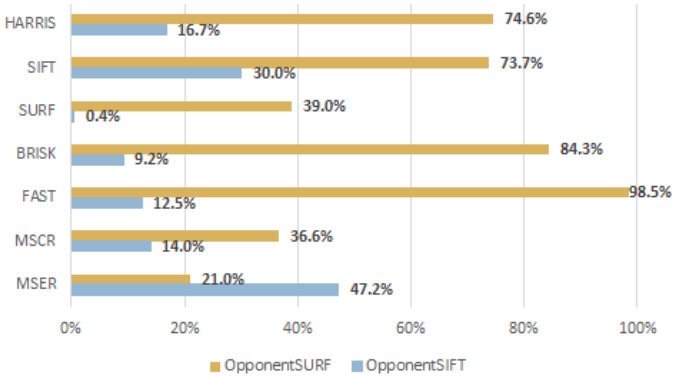

Fig. 8: Improvement of OpponentSIFT and OpponentSURF colour descriptors using our combinatorial descriptor and distance measure. Orange bars show percentage improvements of OpponentSURF descriptors using our method, and blue bars show improvements in OpponentSIFT. The baseline uses standard descriptors with Euclidean distance measures in feature space. The overall average improvement across all of the feature descriptors in this test was 95.2%.

Overall the average improvement across all of the colour feature descriptors in this test was 39.8%.

OpponentSIFT uses colour information in the extraction of the descriptor and can be expected to out-perform those that do not use colour information in a dataset in which colour is visually distinctive. In their thorough evaluation of colour feature descriptors, van de Sande *et al.* conclude that OpponentSIFT is generally a better performing descriptor and is a good choice where there is no prior knowledge of the images or object/scene categories [9]. In our tests, results show that our extension method generally improves matching with this descriptor by up to 47.2% depending on the initial feature detector (Figure 8).

C. Feature matching results

The graphs in Figure 9 summarize the results from our test database of 251 images. Each graph shows the F_1 measure of one of our seven selected feature detectors and all four of the feature extractors, comparing the matching performance of four methods of calculating correspondence. The pale blue line shows SIFT features extracted from each of the feature key points or region centers, the orange line shows rootSIFT features, SURF is in grey, and colour features of OpponentSIFT and OpponentSURF are in yellow and dark blue respectively. Each of the four methods are represented on the x-axis; the *original* correspondence using Euclidean distance of unmodified feature descriptors is the baseline against which we measure performance improvements. *Blur sensitive* applies the blur sensitive feature detection algorithm using unmodified feature descriptors. *Combinatorial* results are those achieved in using the combinatorial texture and colour feature matching descriptor extensions and matching algorithm, and finally *Combinatorial Blur sensitive* are results from the combined methodology described in this paper.

The upward left-to-right trend in each of the graphs demonstrates the improvement in matching performance that

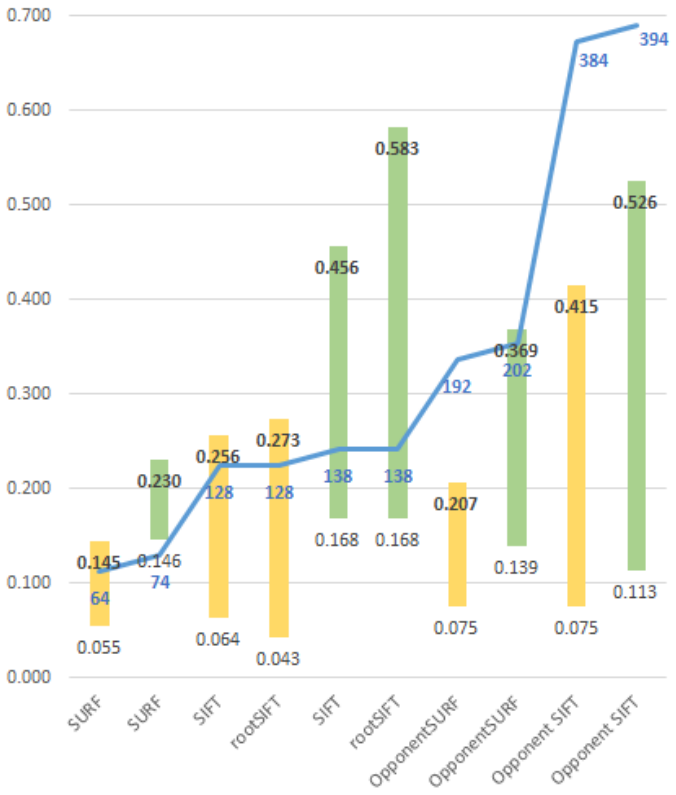

Fig. 10: The correlation between descriptor size and matching accuracy. Yellow bars show measures for established descriptors and Green bars are accuracy measures using our method. Using our method with SIFT and rootSIFT 138D combinatorial descriptors out-perform descriptors of almost 3 times the size.

is achieved with each of our method's components, and the combined methodology. The consistent closeness of the orange and yellow lines in the *Combinatorial Blur sensitive* result is particularly striking. The performance of our method using rootSIFT descriptors ($128 + 10$ dimensions) closely matches the performance of the much larger OpponentSIFT $384+10$ dimension descriptor and significantly outperforms state-of-the-art feature matching using the OpponentSIFT 128D descriptors with the Euclidean distance measure.

V. Storage efficiency vs. matching performance

The choice of feature detector to use in the initial step of the processing pipeline significantly affects the ability to match features across images. The variability of matching accuracy is observable in the results presented in this paper. Systems attempting to match features across a high volume of images are becoming increasingly common, and a key consideration for such systems is the storage efficiency of the descriptors used and the trade-off between storage and accuracy.

The accuracy of feature matching using contemporary techniques generally increases in line with the size of the descriptor that is determined by a choice of extractor, as in the yellow bars in Figure 10 where the top of the bar representing the peak performance on each descriptor is generally higher moving left-to-right. The green bars show the F_1 matching

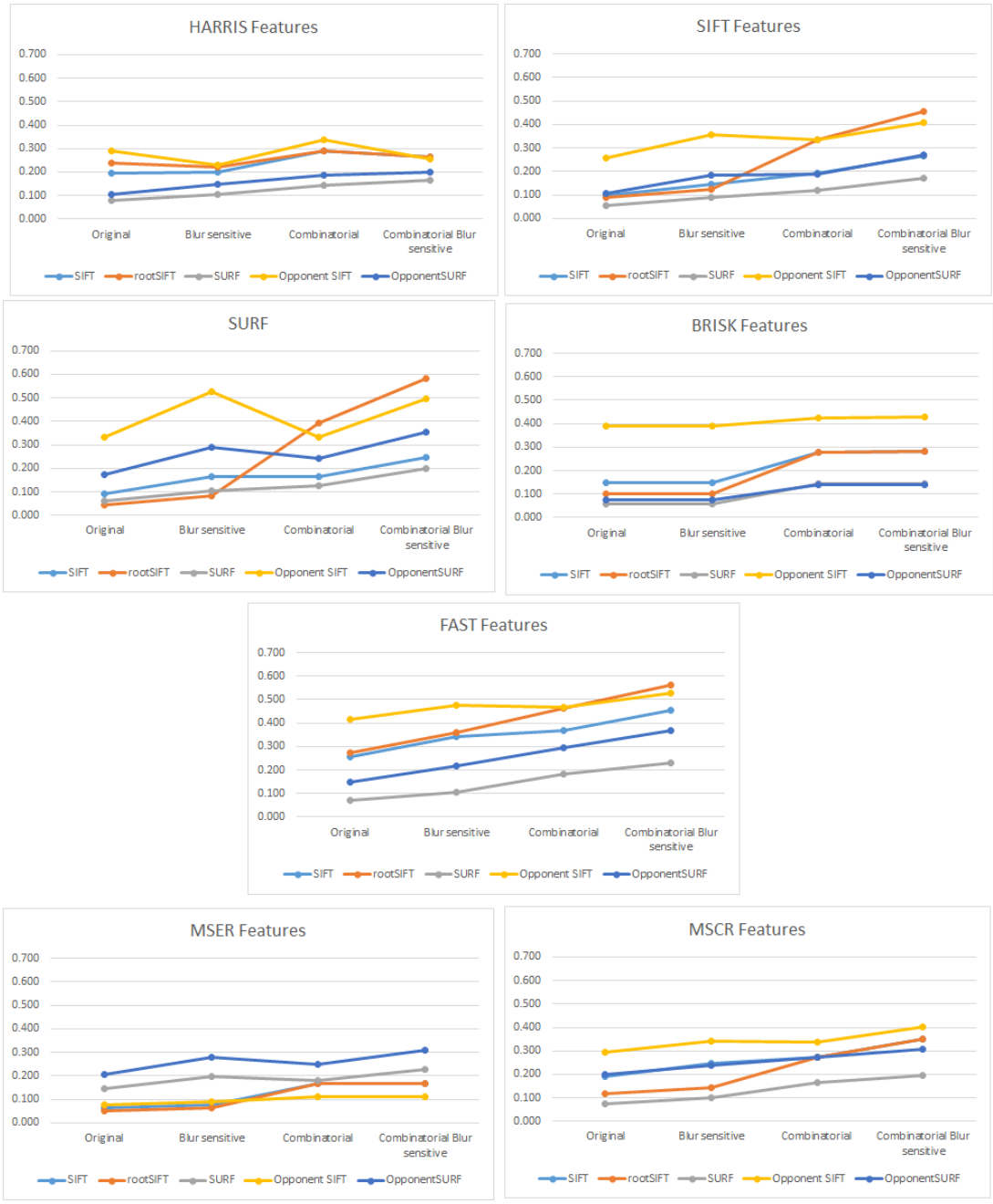

Fig. 9: Summary of the results of feature matching with each component of our method, and the combined methodology (right-most). Each graph shows results from a different feature detector, and compares results with each of four intensity and colour descriptors using four methods; *original* – using Euclidean distance of unmodified feature descriptors is the baseline against which we measure performance improvements, *Blur sensitive* – blur sensitive feature detection algorithm using unmodified feature descriptors, *Combinatorial* – combinatorial texture and colour feature matching descriptor extensions and matching algorithm, *Combinatorial Blur sensitive* – the combined methodology described in this paper.

accuracy using our method, where there is a very significant peak in matching accuracy at dimensionality $D = 138$ where our method using SIFT and rootSIFT descriptors outperforms all other state-of-the-art descriptor matching using Euclidean distance measures. The saving in storage using our *Combinatorial rootSIFT* over the performance-comparable *Combinatorial OpponentSIFT* is $394 - 138 = 256$ bytes per descriptor.

VI. Conclusion

We have introduced a methodology for improved feature correspondence in low-quality images, with an emphasis on storage optimization and execution performance. Our efficient and generic extension for feature descriptors improve the performance of feature matching and the blur sensitive feature detection method further enhances feature matching performance. We have shown the flexibility of the approach by applying it to five common key point descriptors and two popular region descriptors and we have compared the performance of all of them in matching features between images varying in quality and appearance. Our experiments have demonstrated that the introduction of colour information to the feature descriptors, a unique feature distance measure and compensating for inter-image blur differences can improve the matching accuracy over the original descriptors in most combinations that were tested, even where the colour detail is visually subtle in poor quality images.

Our method provides flexibility as it can be used with any feature descriptor extracted from any key point or region detector. Further, evaluation of the method in our problem domain of frame-to-frame feature tracking in low quality videos has demonstrated that smaller descriptors that are computationally lighter can be used to out-perform larger and more intensive feature descriptors. Our experiments have demonstrated an accuracy in matching features that out-performs all state-of-the-art methods using descriptors of less than 40% of size of the nearest performing colour descriptor.

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 also extend their thanks to the Metropolitan Police at Scotland Yard, London, UK, for the supply of and permission to use CCTV images.

References

[1] J. Sivic and A. Zisserman, "Video Google: a text retrieval approach to object matching in videos," in *Proceedings Ninth IEEE International Conference on Computer Vision*, vol. 2, Oct. 2003, pp. 1470–1477.

[2] A. Anjulan and N. Canagarajah, "A unified framework for object retrieval and mining," *IEEE Transactions on Circuits and Systems for Video Technology*, vol. 19, no. 1, pp. 63–76, Jan. 2009.

[3] S. O'Hara and B. A. Draper, "Introduction to the Bag of Features Paradigm for Image Classification and Retrieval," Jan. 2011.

[4] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International Journal of Computer Vision*, vol. 60, no. 2, pp. 91–110, Nov. 2004.

[5] H. Bay, A. Ess, T. Tuytelaars, and L. Vangool, "Computer Vision ECCV 2006," in *9th European Conference on Computer Vision*, vol. 3951, 2006, pp. 346–359.

[6] C. Harris and M. Stephens, "A Combined Corner and Edge Detector," in *Procedings of the Alvey Vision Conference 1988*. Alvey Vision Club, 1988, pp. 147–151.

[7] K. Van De Sande, T. Gevers, and C. Snoek, "Evaluating color descriptors for object and scene recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 32, pp. 1582–1596, Jun. 2010.

[8] A. Bosch, A. Zisserman, and X. Muñoz, "Scene classification using a hybrid generative/discriminative approach," *IEEE transactions on pattern analysis and machine intelligence*, vol. 30, no. 4, pp. 712–27, Apr. 2008.

[9] K. E. van de Sande, T. Gevers, and C. G. M. Snoek, "Evaluating color descriptors for object and scene recognition," *Pattern Analysis and Machine Intelligence, IEEE Transactions on*, vol. 32, no. 9, pp. 1582–96, Sep. 2010.

[10] A. E. Abdel-Hakim and A. A. Farag, "CSIFT: A SIFT descriptor with color invariant characteristics," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2, 2006, pp. 1978–1983.

[11] H. Stokman and T. Gevers, "Selection and Fusion of Color Models for Feature Detection.pdf," in *IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, vol. 1. IEEE, 2005, pp. 560–565.

[12] J. van de Weijer and C. Schmid, "Coloring Local Feature Extraction," in *9th European Conference on Computer Vision*, vol. 3952, 2006, pp. 334–348.

[13] Y. K. Y. Ke and R. Sukthankar, "PCA-SIFT: a more distinctive representation for local image descriptors," in *Proceedings of the 2004 IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2004. CVPR 2004.*, vol. 2. IEEE, 2004, pp. 506–513.

[14] A. Sharma and K. K. Paliwal, "Fast principal component analysis using fixed-point algorithm," *Pattern Recognition Letters*, vol. 28, no. 10, pp. 1151–1155, Jul. 2007.

[15] R. Dash and B. Majhi, "Motion blur parameters estimation for image restoration," *Optik - International Journal for Light and Electron Optics*, vol. 125, no. 5, pp. 1634–1640, Mar. 2014.

[16] C. Schuler and M. Hirsch, "Learning to Deblur," 2014.

[17] J. Canny, "A computational approach to edge detection." *IEEE transactions on pattern analysis and machine intelligence*, vol. 8, no. 6, pp. 679–698, Nov. 1986.

[18] U. Park, A. Jain, I. Kitahara, K. Kogure, and N. Hagita, "ViSE: Visual Search Engine Using Multiple Networked Cameras," in *18th International Conference on Pattern Recognition (ICPR'06)*, vol. 3. IEEE, 2006, pp. 1204–1207.

[19] H. L. H. Liu and R. Setiono, "Chi2: feature selection and discretization of numeric attributes," in *Proceedings of 7th IEEE International Conference on Tools with Artificial Intelligence*, 1995, pp. 388 – 391.

[20] A. Bhattacharyya, "On a measure of divergence between two statistical populations defined by their probability distributions," *Bulletin of the Calcutta Mathematical Society*, vol. 35, pp. 99–109, 1943.

[21] Y. Rubner, C. Tomasi, and L. J. Guibas, "Earth mover's distance as a metric for image retrieval," *International Journal of Computer Vision*, vol. 40, no. 2, pp. 99–121, 2000.

[22] M. J. Swain and D. H. Ballard, "Color indexing," *International Journal of Computer Vision*, vol. 7, no. 1, pp. 11–32, 1991.

[23] K. Mikolajczyk and C. Schmid, "A performance evaluation of local descriptors," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 27, no. 10, pp. 1615–1630, 2005.