Machine Vision and Applications (2016) 27:1187–1196
DOI 10.1007/s00138-016-0785-3

SPECIAL ISSUE PAPER

An extension to the brightness clustering transform and locally contrasting keypoints

Jaime Lomeli-R.[1] · **Mark S. Nixon**[1]

Received: 27 November 2015 / Revised: 5 April 2016 / Accepted: 19 May 2016 / Published online: 9 July 2016

Abstract The need for faster feature matching has left as a result a new set of feature descriptors to the computer vision community, ORB, BRISK and FREAK amongst others. These new descriptors allow reduced time and memory consumption on the processing and storage stages, mitigating the implementation of more complex tasks. The problem is now the lack of fast interest point detectors with good repeatability to use with these new descriptors. A blob-detection algorithm was recently presented that uses an innovative non-deterministic low-level operator called the Brightness Clustering Transform (BCT) (Lomeli-R. and Nixon in The brightness clustering transform and locally contrasting keypoints. In CAIP. Springer, Berlin, pp 362–373, 2015). This algorithm is easy to implement and is faster than most of the currently used feature detectors. The BCT can be thought as a coarse-to-fine search through scale spaces for the true derivative of the image. The new algorithm is called *Locally Contrasting Keypoints* detector (LOCKY). Showing good robustness to image transformations included in the Oxford affine-covariant regions dataset, LOCKY is amongst the fastest affine-covariant feature detectors. In this paper, we present an extension of the BCT that detects larger structures maintaining timing and repeatability; this extension is called the BCT-S.

Keywords Feature detection · Low-level operator · Affine-invariant · Blob detection

✉ Jaime Lomeli-R.
jlr2g12@soton.ac.uk

Mark S. Nixon
msn@ecs.soton.ac.uk

1 Electronics and Computer Sciences, University of Southampton, Southampton, UK

1 Introduction

The detection of points of interest in images is one of the most basic operations in computer vision. The detection of such points is a relatively fast process, with timings usually varying in the range of the milliseconds in implementations in modern computers. Nonetheless, it is necessary to consider that feature detection is in many cases the first step in more complicated tasks, and therefore more rigorous time constrains apply.

In 2008, Tuytelaars and Mikolajczyk presented a survey on local invariant feature detectors discussing many of their characteristics [1]. Perhaps the most important topic is the criteria used to decide which features are more appropriate for particular applications. Nevertheless, it was shown that some features are more suitable for different tasks. Features can be categorised into three general groups, corners, blobs and regions. For instance, corner detection has long been researched and therefore many approaches to solve this problem exist. The *Harris corner detector* [2] is arguably the most well-known feature detector; based on the eigenvalues of the second-order moment matrix, corners can be detected with rotation invariance. Faster solutions have been proposed, *SUSAN* [3], *FAST* [4] and more recently *AGAST* [5], amongst others.

The main problem with corner points is that, because of their persistence through changes in scale, they are ill-suited for describing the size of keypoint they represent; one solution to this problem is the use of blobs. The fact that blobs are contrasting regions implies that their shape carries information about both the scale and affine transformations. Moreover blobs are known to be more robust to noise and blurring than corners. The use of such features is limited nowadays because the detection of blobs is usually slower than the detection of other features.

Some of the most well-known blob detectors are the *Laplacian of Gaussian* [6] (and its approximation by *difference of Gaussians*) and the *determinant of Hessian*. A fast implementation of the Hessian approach was presented by Bay et al. in [7], this algorithm is well known as *Speeded up Robust Features* (SURF). More recently, Agrawal et al. presented in [8] an algorithm known as *CenSurE* (also known as STAR detector); this algorithm, using integral and rotated-integral images, sets a polygon in every pixel and calculates an approximation of the Laplacian of Gaussian.

Detectors are called *invariant* to some transformations, Tuytelaars and Mikolajczyk suggest the term should be *covariant* when the features change covariantly with the image transformation [1]. Mikolajczyk and Schmid [9] present an affine-covariant solution that, based on the second-order moments of the regions surrounding the detected points, the features are affine-normalised; their solution is robust and elegant but very slow. The *Maximally Stable Extremal Regions* (MSER) [10] is an affine-covariant interest region detector that improves the computation time over Mikolajczyk's work; however, it lacks robustness against blurring and noise. The use of affine features is related to the intention of making detectors robust to perspective changes. Human vision is very robust to image transformations like blurring, rotation and scaling, and also to changes in perspective. Although perspective transformations are different from affine transformations, affine-covariant local feature detectors are good approximations due to their local nature.

Image retrieval [11], matching of stereo pairs of images [10, 12], object detection [13], tracking [14], and many other applications are aided by the use of feature matching. The usual process that follows the detection of points of interest is description; by extracting information from the region surrounding a point of interest, it is possible to *measure* the similarity between two points in an image. The algorithm that introduced the idea of feature description is SIFT [15], recently a new set of efficient algorithms for description was introduced, ORB [16], BRISK [17], FREAK [18] and others. The work presented in [19] by Heinly et al., analyses characteristics of the stages of feature detection and description, and it is left clear that the area of feature detection has not received as much attention as description in the last decade. The problem is now the lack of fast interest point detectors with good repeatability to use with these new descriptors.

This paper is an extension of the work presented by Lomeli-R. and Nixon [20]. The *Brightness Clustering Transform* (BCT) is a novel low-level operator that transforms the image into an accumulator matrix (a blob map), and it benefits from the use of integral images to perform a fast search through different scale spaces. Information from the accumulator matrix is extracted to detect points of interest. The new algorithm is called *Locally Contrasting Keypoints* (LOCKY). LOCKY features contain information about scale and affine

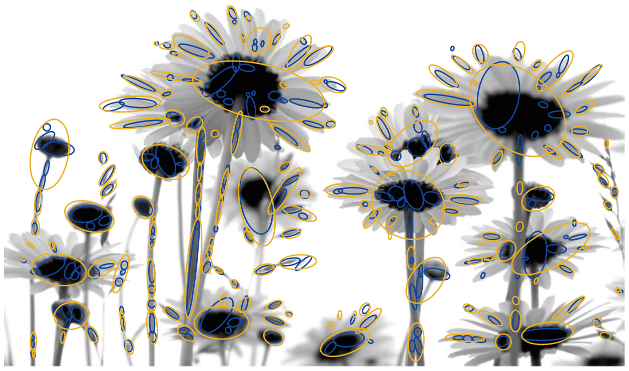


Fig. 1 Blobs detected on an image of daisies. LOCKY features are shown in *blue* (detected using the BCT). In *yellow* are the LOCKY-S features (detected using the BCT-S) (color figure online)

transformations and are up to three times faster to detect than the MSER regions. We introduce the BCT-S, an extension to the regular BCT; this extension concentrates on the extraction of larger structures. We refer to the features extracted using the BCT-S as LOCKY-S features. LOCKY and LOCKY-S features show good repeatability scores using the measure presented in [21]. Figure 2 shows a flowchart of the presented algorithm. The spacial dispersion of the detected features is analysed, and we also discuss how some applications can benefit from the sparseness of features in images. Figure 1 illustrates the difference between LOCKY and LOCKY-S features.

2 The brightness clustering transform

The BCT is a non-deterministic low-level operator that employs a voting scheme on an integral image for improving performance. Each vote is randomly initialised to extract information from a region on the image; when the next vote is initialised, the algorithm changes the location of attention.

The integral image as presented by Viola and Jones [22] is a very useful tool to calculate the sum of rectangular areas of pixels in only three operations disregarding the size of the rectangles; it is widely used because it allows to make calculations at different scales without added computational cost.

Equation 1 shows the definition of the integral image of the image function $\mathbf{P}(x, y)$. The recurrences in Eqs. 2 and 3 allow the calculation of $ii(x, y)$ in one pass over the image, ($s(x, y)$ is the cumulative row sum, $s(x, -1) = 0$ and $ii(-1, y) = 0$).

$$ii(x, y) = \sum_{x' \le x, y' \le y} \mathbf{P}(x', y'). \tag{1}$$

$$s(x, y) = s(x, y - 1) + \mathbf{P}(x, y). \tag{2}$$

$$ii(x, y) = ii(x - 1, y) + s(x, y). \tag{3}$$

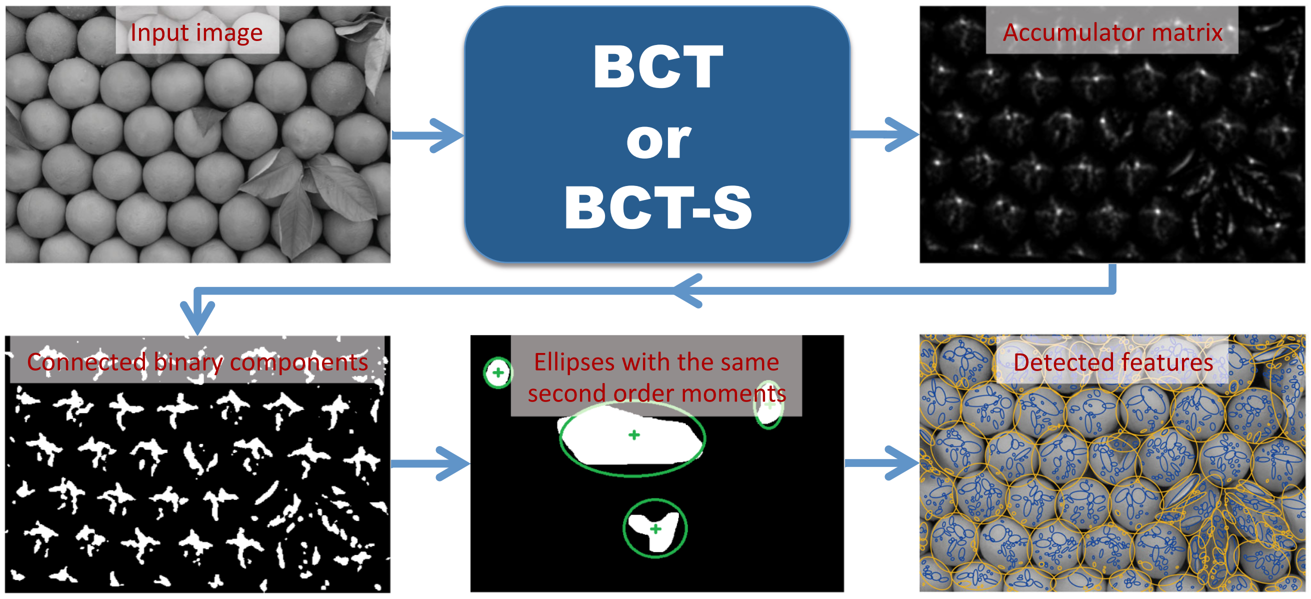

Input image
BCT or BCT-S
Accumulator matrix
Connected binary components
Ellipses with the same second order moments
Detected features

Fig. 2 The process for extracting the LOCKY features begins with the transformation of the image into a blob map using the BCT (*blue*) or the BCT-S (*yellow*). The accumulator matrix is thresholded, and the set of connected components is then extracted. The LOCKY features are the *ellipses* with the same second-order moments as the connected binary components (color figure online)

The result of the BCT is an accumulator matrix that is initialised as a null-matrix of the same size as the input image; a vote means incrementing the value of an element in the accumulator matrix by one; each vote is obtained in three steps. First, a rectangle with random position and size is initialised within the image. For creating blob maps, we select the width and the height of the rectangle to be a power of two i.e. $width = 2^n$ and $height = 2^m$ $\{n, m \in \mathbb{N}\}$. The second step is to divide the rectangle into four smaller subregions, and the subregion with the biggest sum of brightness is now considered to be the initial rectangle; the sum is calculated using the integral image. Consider the rectangle R_t where $t = 0$ for the initial position and size, and its subregions r_0, r_1, r_2 and r_3; the next region will have an initial rectangle R_{t+1}. The second step is repeated until R_t has either $width = 2$ or $height = 2$.

$$R_{t+1} = \arg\max_{r_i} \sum_{x,y \in r_i} \mathbf{P}(x, y). \quad i = 0, 1, 2, 3. \tag{4}$$

Suppose the last R_t is situated in (x_f, y_f) and has $width = w_f$ and $height = h_f$; in the third step, the pixel in $loc = (x_f + w_f/2, y_f + h_f/2)$ is voted. This sequence of steps is graphically presented in Fig. 3. Algorithm 1 shows the pseudocode for the BCT.

After a user-defined number of votes, the accumulator matrix is smoothed with a small Gaussian kernel and then normalised. Smoothing the accumulator matrix removes the effects of noise in the voting process and helps to find the true shape of the extracted blobs. As mentioned in [23], intermediate shape priors yield discriminative shape structures; these structures can improve recognition tasks.

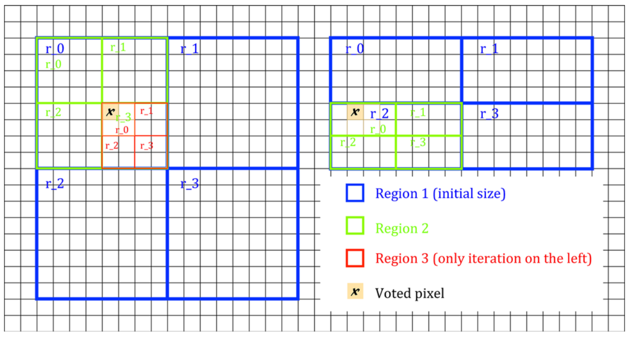


Fig. 3 A squared vote on the *left* and a rectangular vote on the *right*. R_0 in *blue*, R_1 in *green* and R_2 in *red*. The subregions of every step are marked as *r_i* with the *same colour* as the step they belong to (color figure online)

The BCT can be thought as a coarse-to-fine search through scale spaces for the true derivative of the image. Every subdivision of a rectangle is in the next smaller octave, and thus, the votes start at a big scale and refine until they get to the smallest scale possible in the image. The use of rectangles benefits affine locality; for example, a horizontal rectangle discards some information in the y axis and operates in the same scale in the x axis; this allows an improvement on the detection of oval blobs. Suppose a vote lies in (x_v, y_v) with an initial rectangle in (x_0, y_0), another vote will most likely lie in $(x_v + 1, y_v)$ if a rectangle of the same size is set in $(x_0 + 1, y_0)$. This property clusters votes around the centre of blobs, and so the shape of the blobs is extracted.

Algorithm 1 BCT.

```
$integ$ = calculate integral of the input image
$accum$ = initialise accumulator matrix

for ($vote = 1$ to $max\_votes$) do
    First Step:
    Init region $R$:
        $n = \text{rand}(rangeMin, rangeMax)$
        $m = \text{rand}(rangeMin, rangeMax)$
        $width = 2^n$
        $height = 2^m$
        $x = \text{rand}(0, imWidth - width)$
        $y = \text{rand}(0, imHeight - height)$

    Second Step:
    while ($width > 2$ & $height > 2$) do
        divide $R$ in 4 subregions $r_0, r_1, r_2$ and $r_3$
        $r_{max}$ = max-brightness($r_0, r_1, r_2, r_3$) (use $integ$)
        $R = r_{max}$   this implies:
            $width = width/2$
            $height = height/2$
            $x = x_{r_{max}}$
            $y = y_{r_{max}}$
    end while

    Third Step:
    $loc = (x + width/2, y + height/2)$
    $accum[loc] = accum[loc] + 1$

end for
```

The implementation of the BCT requires the selection of two parameters, i.e. the size of the rectangles and the amount of votes. The most commonly used values for the width and height of the rectangles range from 2^3 to 2^7, and this range may be modified depending on the size of the image and the size of the blobs to be extracted. From experiments, we deduce that 5×10^4 to 1×10^5 votes are enough to extract the blobs in a 1024×768 image; nonetheless, amounts as small as 2×10^4 votes suffice to extract significant blobs from the same image.

So far the bright blobs are extracted by finding the subregions with the biggest sum of pixels; if we want to find dark blobs, we could either modify Eq. 4 to be

$$R_{t+1} = \arg\min_{r_i} \sum_{x,y \in r_i} \mathbf{P}(x, y), \quad i = 0, 1, 2, 3, \tag{5}$$

or, find the bright blobs of the inverted image, i.e. considering the image is an 8-bits representation, we do $\mathbf{P}' = 255 - \mathbf{P}(x, y)$ or $\mathbf{P}'$ =not($\mathbf{P}(x, y)$).

Continuing the voting process while the width and the height are greater than two concentrates the votes in a compact manner around locally contrasting regions. This compactness achieves good shape description of small regions, and therefore the detection of larger features in images with higher resolution is difficult.

Sometimes fine features are desired, when this is not the case the second and third steps can be modified to be able to

Algorithm 2 BCT-S.

```
Modified Second Step:
for ($div = 0$; $div < 3$; $div++$) do
    divide $R$ in 4 subregions $r_0, r_1, r_2$ and $r_3$
    $r_{max}$ = max-brightness($r_0, r_1, r_2, r_3$) (use $integ$)
    $R = r_{max}$   this implies:
        $width = width/2$
        $height = height/2$
        $x = x_{r_{max}}$
        $y = y_{r_{max}}$
end for

Modified Third Step:
$accum[r_{max}] = accum[r_{max}] + 1$
```

detect larger features. The extension of the BCT for extracting larger structures (BCT-S) is described in Algorithm 2. In the second step, instead of iterating the rectangle reduction until either the width or the hight equal two, the reduction is only performed three times ($rangeMin > 2$). On the third step, all the pixels within the last r_{max} region are voted. This is the equivalent to running the algorithm at multiple scales. Figure 4 shows the detected features using the BCT and the BCT-S.

3 Locally contrasting keypoints

LOCKY features are blob keypoints extracted directly from the accumulator image result of the BCT. After the normalisation process, the accumulator image is thresholded; the set of all the connected components in the binary image are the detected blobs. The threshold is the third parameter of the entire process required to detect LOCKY features, the *strength* of the detected features is related to this threshold, and the lower it is the more features will be detected. Common values for the threshold vary from 0.1 to 0.3 (if the transform is normalised in the range [0, 1]).

Finding the ellipse with the same second-order moments as the connected components is a fast way of extracting information from them. If F is a $2 \times N$ matrix ($N > 1$) containing the coordinates of the pixels in a connected component ($f_0, f_1, \ldots, f_{N-1}$), the mean is the centre of the feature (Eq. 7).

$$Q = \frac{1}{N-1} \sum_{n=0}^{N} (f_n - M_F)(f_n - M_F)^T. \tag{6}$$

$$M_F = \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}. \tag{7}$$

The eigenvalues of the sample covariance matrix Q (Eq. 6) represent the size of the axes of the ellipse with the same second-order moments; the eigenvectors define the direction

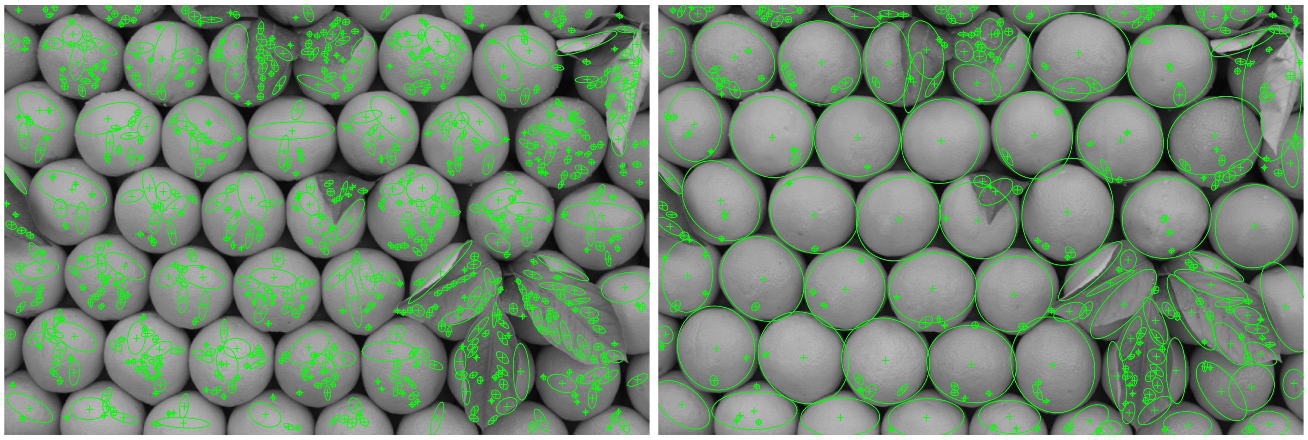


Fig. 4 LOCKY features on the left (extracted using the regular BCT), and LOCKY-S features on the right (extracted using the BCT-S). The BCT-S promotes the detection of larger structures in the image

Fig. 5 The first images of the sequences in the Oxford affine-covariant regions dataset [21]

of the axes. This step is similar to the ellipses of the MSER regions. The advantages of this method are that one can detect the scale of the features by the size of the axes and it is also possible to extract information about affine transformations and rotation of the blobs.[1]

4 Results

The Oxford affine-covariant regions dataset [21] (Fig. 5) presents a good challenge for interest point detection, and it is widely used for evaluation; eight sequences composed of six images each with increasing image transformations including decreasing illumination, change in perspective, blurring, jpeg compression and a mix of scale and rotation. We use the measure of *repeatability* defined in the same paper to compare LOCKY and LOCKY-S with both, affine-covariant and non-affine-covariant interest point detectors. In the affine-covariant group, we use as comparison the *Harris-Affine* and *Hessian-Affine* detectors [9] and *MSER* [10]. In the non-affine-covariant group, we use the BRISK detector [17], the SURF (fast-Hessian) detector [7], and the STAR detector [8]. LOCKY-1 uses 1×10^5 rectangles of size ranging from 2^3 to 2^5 and a threshold of 12 %; LOCKY-2 uses 1×10^6 rectangles of the same characteristics. Both LOCKY-1 and LOCKY-2 use the regular BCT; LOCKY-S uses the same settings as LOCKY-1 with the only difference that it uses the BCT-S to extract the features.

The measure of repeatability consists of projecting the detected features to the same basis as the original image (the first image of the sequence) using an homography matrix, then we compare correspondences of features and measure how well the detected regions overlap. For more information on this measure, see [21]. We use the correspondences with

[1] http://www.robots.ox.ac.uk/~vgg/research/affine.

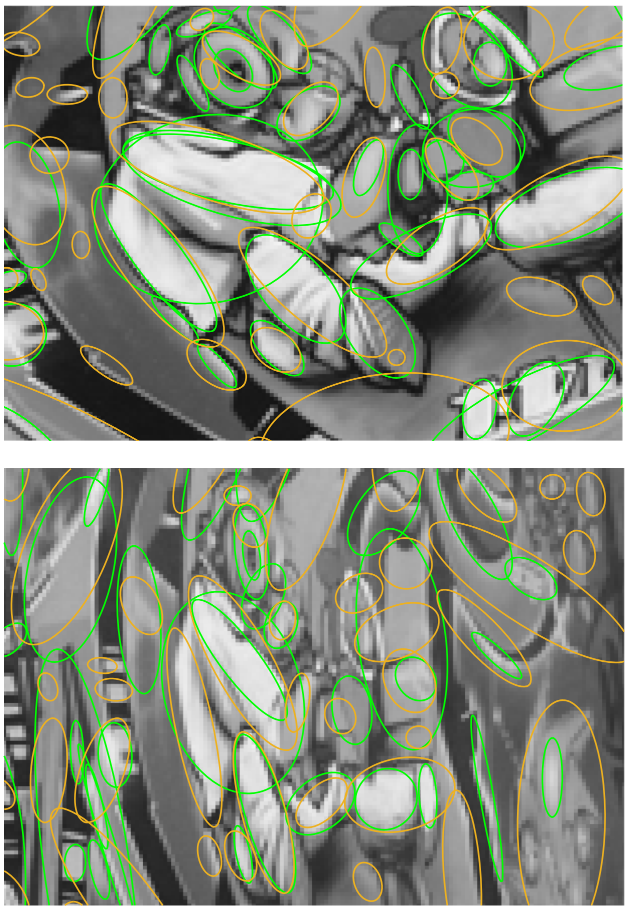


Fig. 6 LOCKY-S features in yellow and MSER regions in *green*, on the second (*top*) and fifth (*bottom*) images of the graffiti sequence. The detected features change covariantly with the change in perspective (color figure online)

40 % overlap. To be able to compare LOCKY and LOCKY-S with non-affine-covariant detectors, we "disable" the measure by using a circle with a radius equal to half the size of the major axis of the ellipses; these results are shown in Fig. 7. Since the detection of the LOCKY features is based in a non-deterministic transform, every run matches different features; the repeatability results of LOCKY-1, LOCKY-2 and LOCKY-S include the mean and variance over 100 runs of the test (examples of the detected features are shown in Fig. 6).

The timing results shown in Table 1 were obtained using the OpenCV implementations of the algorithms (using mex files in MATLAB) with a 2 GHz $i7$ processor (the used Harris-Affine and Hessian-Affine implementations are the binaries provided in www.robots.ox.ac.uk/~vgg/research/affine/index.html). Note that BRISK uses a multi-scale version of the AGAST detector which is a faster implementation of the FAST detector; LOCKY and LOCKY-S have similar timings while being able to provide information on affine transformations of the features.

In 2008, Tuytelaars and Mikolajczyk suggested that the density of features should reflect the information content of the image [1]. This is a useful property for tasks such as object detection; nevertheless, for tasks such as navigation or landmark matching, occlusion can become a big problem if features cluster on regions of the image. Spacial dispersion of features and the use of other evaluation measures were addressed in [19]. The measure of dispersion presented in Table 1 is obtained by dividing the image into $B = 100$ non-overlapping bins. We count the number of features that lie in each bin as a two-dimensional histogram of locations. If the location of a feature is represented with a two-dimensional vector l_n having N features, the count of features in each bin (a region called Γ_b) is represented by C_b (Eq. 8).

$$C_b = \sum_{n=0}^{N} a_b(l_n), \tag{8}$$

$$a_b(l_n) = \begin{cases} 1 & \text{when } l_n \subset \Gamma_b, \\ 0 & \text{otherwise.} \end{cases} \tag{9}$$

The dispersion index D is calculated as

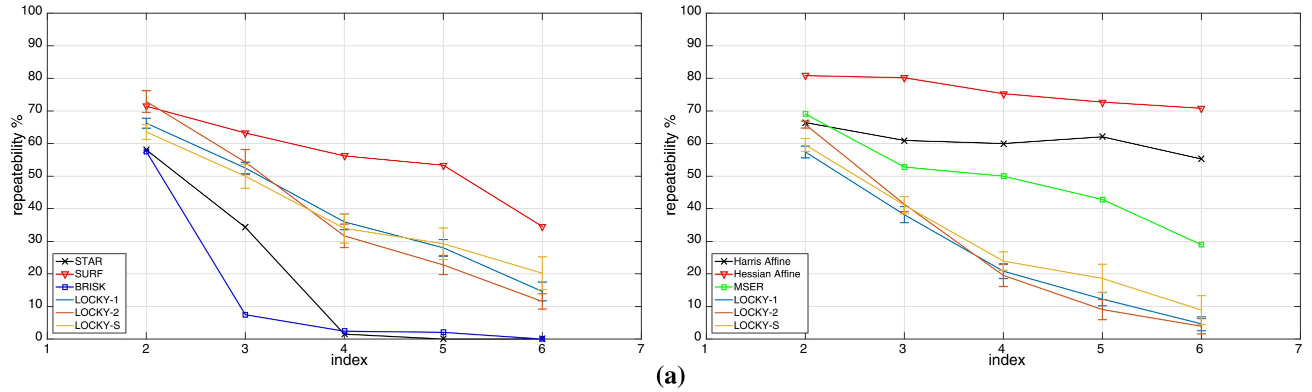
$$M_C = \frac{1}{B}\sum_{b=0}^{B} C_b, \tag{10}$$

$$D = \frac{\sum_{b=0}^{B}(C_b - M_C)^2}{M_C}. \tag{11}$$

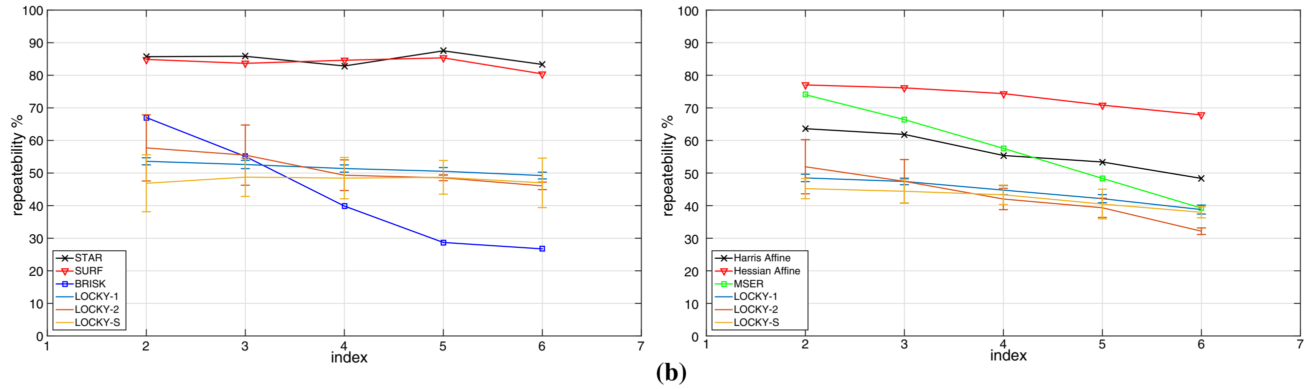
The lower the value of D, the more sparse are the features in the image. Spacial-frequency energy is usually not evenly distributed within an image, and this happens more commonly in natural images. When features cluster in certain regions of an image, it can be understood that locality is not successfully achieved. That is, more features tend to be detected in regions with more spacial-frequency energy; this could be the result of normalisation or thresholding processes, and as a consequence, features in regions with less energy are lost. Therefore, small values of the dispersion index indicate that features are detected independently from the spacial-frequency energy distribution.

5 Discussion

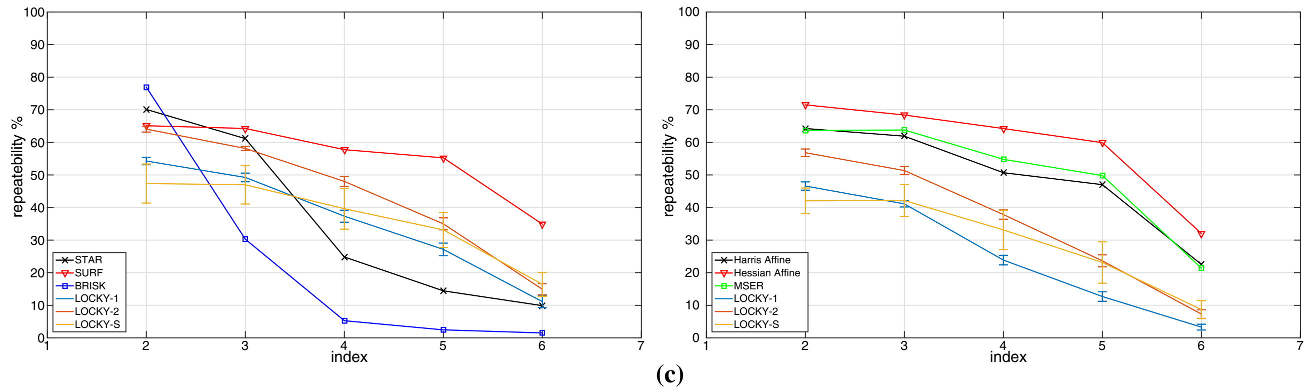
LOCKY and LOCKY-S provide a good alternative for fast feature detection, and timing results show that the presented algorithm is amongst the fastest feature detectors available and provides a trade-off between run-time and performance. On average, the LOCKY approach can deliver good performance: it is not dependent on initialisation and competes



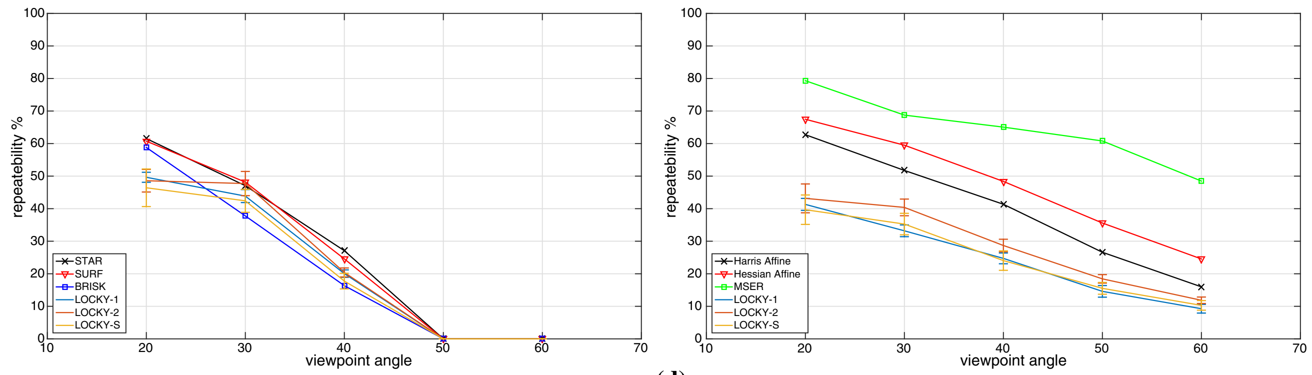
(a)



(b)

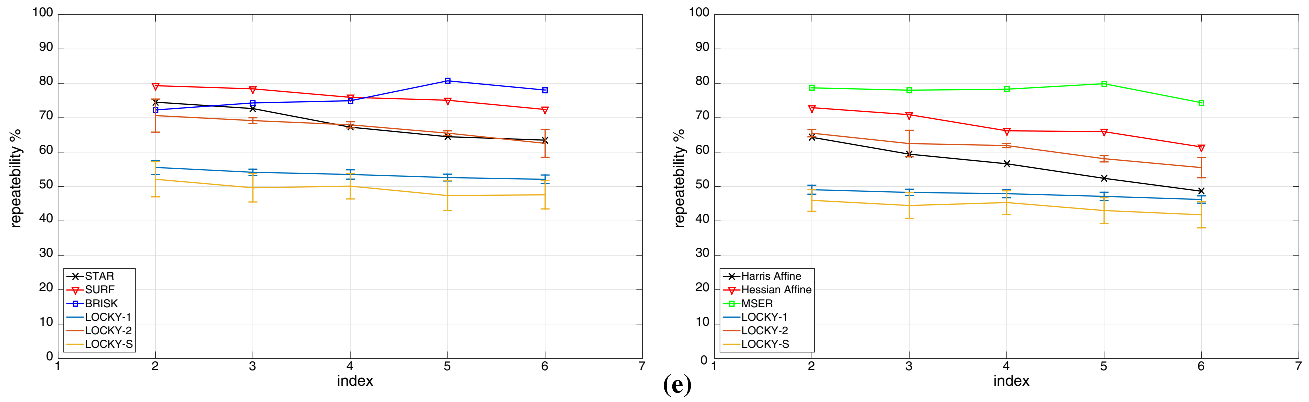


(c)

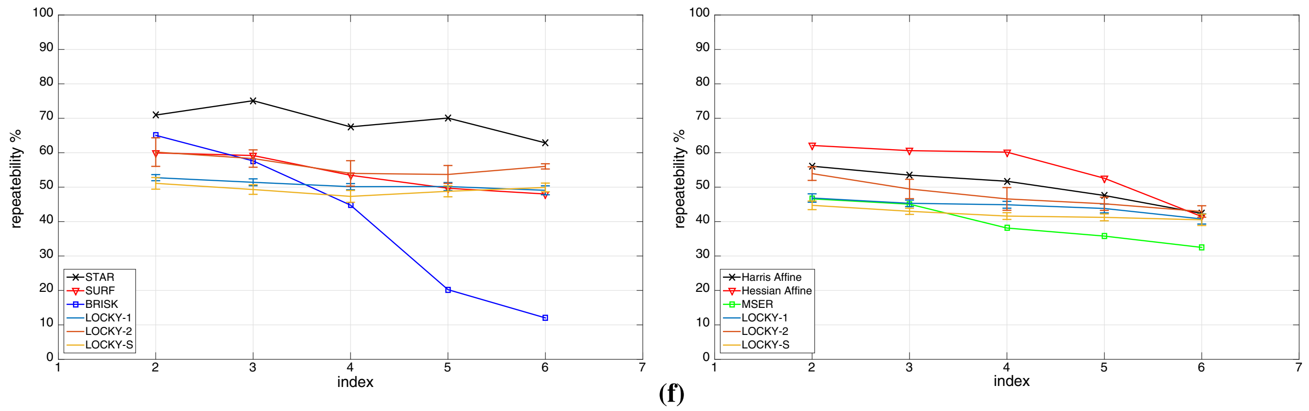


(d)

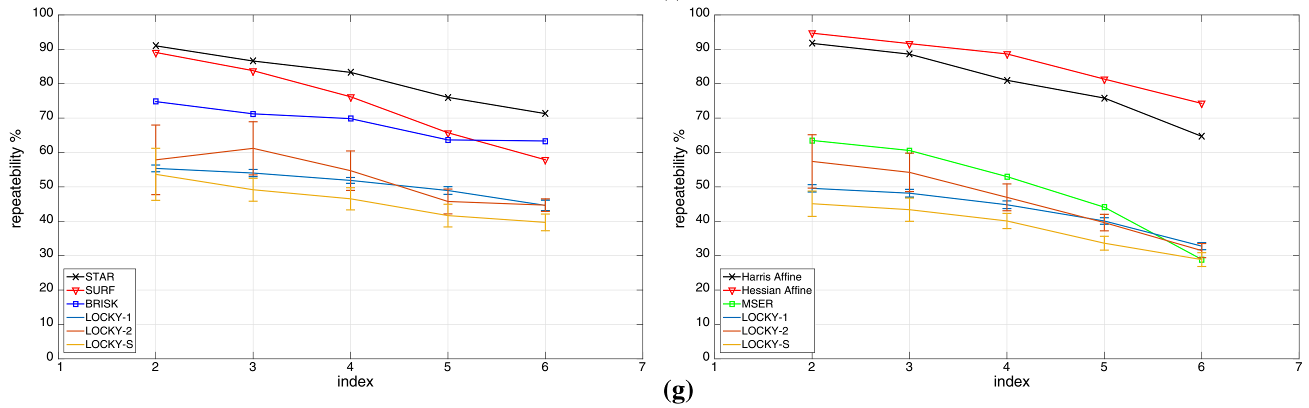
Fig. 7 The repeatability test presented in [21]. Figures on the *left column* present the results of the repeatability test with no affine-covariance (features are *circles*); the *right column* shows the figures with the results using affine-covariance (features are *ellipses*). LOCKY-1 uses 1×10^5 rectangles of size ranging from 2^3 to 2^5 and a threshold of 24 %; LOCKY-2 uses 1×10^6 rectangles of the same characteristics. **a** Bark sequence. **b** Bikes sequence. **c** Boat sequence. **d** Graffiti sequence. **e** Leuven sequence. **f** Trees sequence. **g** UBC sequence. **h** Wall sequence



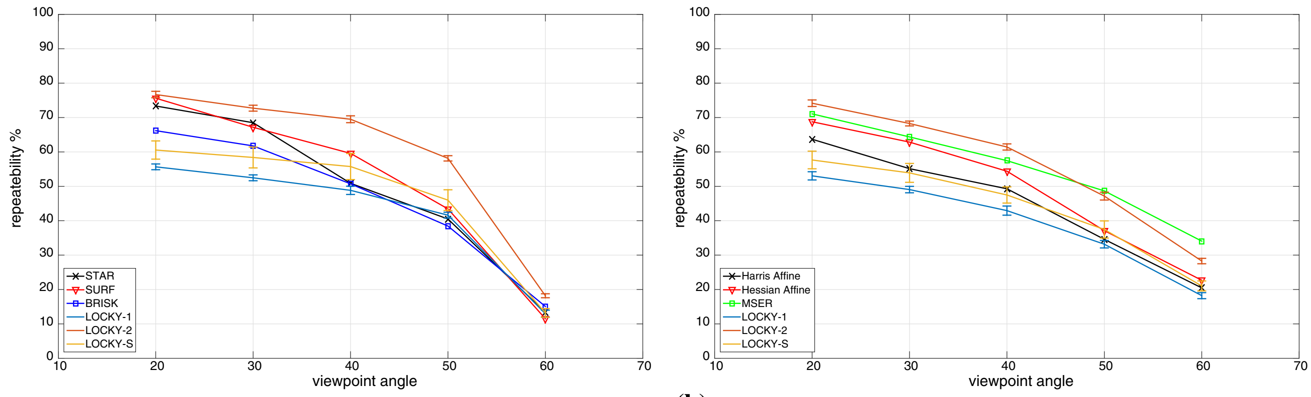
(e)



(f)



(g)



(h)

Fig. 7 continued

Table 1 The average factor of time for a set of 127 images

Detector	Tim	Typ	Affine	Disp.
SURF [7]	4.53	Blobs	×	12.95
BRISK [17]	0.98	Corners	×	10.42
STAR [8]	0.72	Blobs	×	12.57
LOCKY-1	1	Blobs	✓	1.26
LOCKY-2	4.79	Blobs	✓	1.65
LOCKY-S	1.05	Blobs	✓	1.04
MSER [10]	3.26	Regions	✓	8.67
Hessian-Affine [9]	6.04	Blobs	✓	19.03
Harris-Affine [9]	8.53	Corners	✓	16.16

The images were converted to grayscale with 1024 × 768 pixels. The dispersion index measures sparseness of the features on the image (calculated with Eq. 11 averaged over the same set of 127 images, the lower the index the more evenly distributed)

with those techniques already available and with its own advantages, namely speed and sparseness. LOCKY detector might appear better on rectilinear structures (the wall) than on objects (the boat), and this can be investigated further.

Features detected using the BCT-S tend to be more congruent with what we recognise as individual objects, while the ones detected with the regular BCT are smaller (see the image of oranges in Fig. 4). Timings and sparseness results indicate that LOCKY is a good alternative for several applications including landmark selection and image matching.

References

1. Tuytelaars, T., Mikolajczyk, K.: Local invariant feature detectors: a survey. In: Foundations and Trends in Computer Graphics and Vision, vol. 3, pp. 177–280. Now Publishers Inc. (2008)
2. Harris, C., Stephens, M.: A combined corner and edge detector. In: Mathews, M.M., (ed.) Proceedings of the 4th ALVEY Vision Conference, Manchester, UK, vol. 15, pp. 147–151 (1988)
3. Smith, S.M., Brady, J.M.: SUSAN—a new approach to low level image processing. In: IJCV, vol. 23, pp. 45–78. Springer (1997)
4. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In: ECCV, pp. 430–443. Springer (2006)
5. Mair, E., Hager, G.D., Burschka, D., Suppa, M., Hirzinger, G.: Adaptive and generic corner detection based on the accelerated segment test. In: ECCV, pp. 183–196. Springer (2010)
6. Marr, D., Hildreth, E.: Theory of edge detection. In: Proceedings of the Royal Society of London. Series B. Biological Sciences, vol. 207, pp. 187–217. The Royal Society (1980)
7. Bay, H., Tuytelaars, T., Van Gool, L.: Surf: Speeded up robust features. In: ECCV, pp. 404–417. Springer (2006)
8. Agrawal, M., Konolige, K., Blas, M. R.: Censure: Center surround extremas for realtime feature detection and matching. In: ECCV, pp. 102–115. Springer (2008)
9. Mikolajczyk, K., Schmid, C.: An affine invariant interest point detector. In: ECCV, pp. 128–142. Springer (2002)
10. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust Wide-baseline stereo from maximally stable extremal regions. In: Image and Vision Computing, vol. 22, pp. 761–767. Elsevier (2004)
11. Schaffalitzky, F., Zisserman, A.: Automated location matching in movies. Comput. Vis. Image Underst. **92**(2), 236–264 (2003)
12. Baumberg, A.: Reliable feature matching across widely separated views. In: CVPR, vol. 1. IEEE (2000)
13. Lepetit, V., Fua, P.: Keypoint recognition using randomized trees. In: Pattern Analysis and Machine Intelligence. 28.9, pp. 1465–1479. IEEE (2006)
14. Sivic, J., Zisserman, A.: Video Google: A text retrieval approach to object matching in videos. In: Proceedings on Computer Vision. IEEE (2003)
15. Lowe, D.: Object recognition from local scale-invariant features. In: Proceedings on Computer Vision, vol. **2**. IEEE (1999)
16. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: an efficient alternative to SIFT or SURF. In: ICCV, pp. 2564–2571. IEEE (2011)
17. Leutenegger, S., Chli, M., Siegwart, R.Y.: BRISK: binary robust invariant scalable keypoints. In: ICCV, pp. 2548–2555. IEEE (2011)
18. Alahi, A., Ortiz, R., Vandergheynst, P.: Freak: fast retina keypoint. In: CVPR, pp. 510–517. IEEE (2012)
19. Heinly, J., Dunn, E., Frahm, J.-M.: Comparative evaluation of binary features. In: ECCV, pp. 759–773. Springer (2012)
20. Lomeli-R., J., Nixon, M.S.: The Brightness clustering transform and locally contrasting keypoints. In: CAIP, pp. 362–373. Springer, Berlin (2015)
21. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Van Gool, L.: A comparison of affine region detectors. In: IJCV, vol. 65, pp. 43–72. Springer (2005)
22. Viola, P., Jones, M.: Robust real-time object detection. In: IJCV, vol. 4, pp. 34–47. Springer (2001)
23. Dickinson, S.J., Pizlo, Z.: Shape Perception in Human and Computer Vision. Springer, Berlin (2013)

Jaime Lomeli-R. received his degree in Electronics Engineering from the Technology Institute of Queretaro, Mexico, in 2011. He then graduated with honours from the University of Southampton receiving an MSc degree in Artificial Intelligence in 2013. He is now a PhD student at the same university. His research interests are feature detection and the detection of symmetry in images.

Mark S. Nixon is the Professor in Computer Vision at the University of Southampton, UK. His research interests are in image processing and computer vision. His team develops new techniques for static and moving shape extraction which have found application in automatic face and automatic gait recognition and in medical image analysis. His team were early workers in face recognition, later came to pioneer gait recognition and more recently joined the pioneers of ear biometrics. His vision textbook, with Alberto Aguado, Feature Extraction and Image Processing (Academic Press) reached 3rd Edition in 2012 and has become a standard text in computer vision. With Tieniu Tan and Rama Chellappa, their book Human ID based on Gait is part of the Springer Series on Biometrics and was published in 2005. He has chaired/program chaired many conferences and given many invited talks. Mark is a member of IAPR TC4 Biometrics and of the IEEEBiometrics Council. He is a Fellow IET and FIAPR and a Distinguished Fellow of the BMVA.