The Brightness Clustering Transform and Locally Contrasting Keypoints

Jaime Lomeli-R. Mark S. Nixon

University of Southampton, Electronics and Computer Sciences
jlr2g12@ecs.soton.ac.uk

Abstract. In recent years a new wave of feature descriptors has been presented 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 of tasks such as image matching or visual odometry, enabling real time applications. The problem is now the lack of fast interest point detectors with good repeatability to use with these new descriptors. We present a new blob-detector which can be implemented in real time and is faster than most of the currently used feature-detectors. The detection is achieved with an innovative non-deterministic low-level operator called the *Brightness Clustering Transform* (BCT). The BCT can be thought as a coarse-to-fine search through scale spaces for the true derivative of the image; it also mimics trans-saccadic perception of human vision. We call the new algorithm *Locally Contrasting Keypoints* detector or *LOCKY*. Showing good repeatability and robustness to image transformations included in the Oxford dataset, LOCKY is amongst the fastest affine-covariant feature detectors.

1 Introduction

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 criterion used to decide which features are more appropriate for particular applications. Nevertheless, it was shown that some features are more appropriate 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. In need of faster algorithms other solutions were 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 of the lack of efficient algorithms that can find and extract information from blobs in real time. Recently a new set of descriptors was introduced, ORB [14], BRISK [15] and FREAK [16] amongst others; these descriptors can be implemented in real time in mobile processors. In [18], Heinly *et al.* presented a comparison of detectors and descriptors, it is left clear that the area of feature detection has not received as much attention as description in the last decade.

Some of the most well known blob detectors are the *Laplacian of Gaussian* [17] (and its approximation by *difference of Gaussians*) and, the *determinant of Hessian*. A fast implementation of the Hessian approach was presented in [11], this algorithm is well known as *Speeded up Robust Features* (SURF). More recently, Agrawal *et al.* presented in [7] an algorithm known as *CenSurE*; 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 suggests the term should be *covariant* when they change covariantly with the image transformation [1]. Mikolajczyk *et al.* [12] 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) [6] is an affine-covariant interest region detector that improves the computation time over Mikolajczyk's work, although 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. We introduce a rapid novel affine-covariant blob detector in response to the wave of new binary descriptors; figure 1 shows some of the keypoints detected with the proposed algorithm.

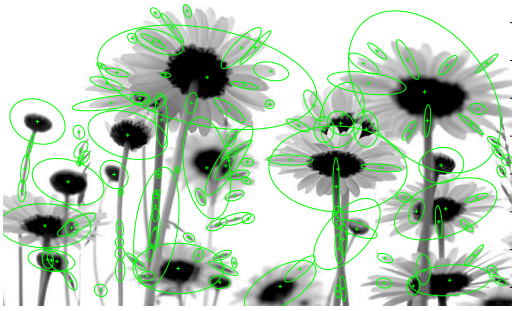


Fig. 1: LOCKY blobs detected on an image of daisies.

This paper is divided in two main sections. First, we present the *Brightness Clustering Transform*. The BCT is an efficient algorithm that can be used to create both blob maps and ridge maps. It benefits from the use of integral images to perform a fast search through different scale spaces. Secondly, information from the blob maps is extracted; the detected blobs are called *Locally Contrasting Keypoints* (LOCKY). These keypoints contain information about the scale and affine transformations of the detected blobs and, are up to three times faster to detect than the MSER regions.

2 The Brightness Clustering Transform

Most of the complex human visual activities require alternations between eye fixations and significantly rapid eye movements known as saccades. The information humans extract from a scene is a series of *snapshots* (fixations), however, we perceive scenes as single views [19]. There exist a temporal memory which integrates those *snapshots* into one *memory image*, preserving the extracted information from the fixations; the process is known as *trans-saccadic integration*.

The BCT is a non-deterministic algorithm that employs a voting scheme using rectangles on the integral image space. Each vote is randomly initialised, then it extracts information from the region being attended resembling eye fixations. When the next vote is initialised, the algorithm changes the location of attention, this is a saccade.

2.1 Blob Detection

The integral image as presented in [10], 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.

$$ii(x,y) = \sum_{x' \leq x, y' \leq y} Im(x', y'). \tag{1}$$

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

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

Equation 1, shows the definition of the integral image. The recurrences in equations 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 $ii(-1,y) = 0$).

The BCT is initialised as a null-matrix of the same size as the input image (the output *memory image*). A vote means incrementing the value of a matrix element in the BCT by one; each vote is obtained in three steps. First, a rectangle with random position and size is initialised within the image. For

finding blobs, 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 in four smaller sub-regions, the sub-region with the biggest sum 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 $height = 2$.

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

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

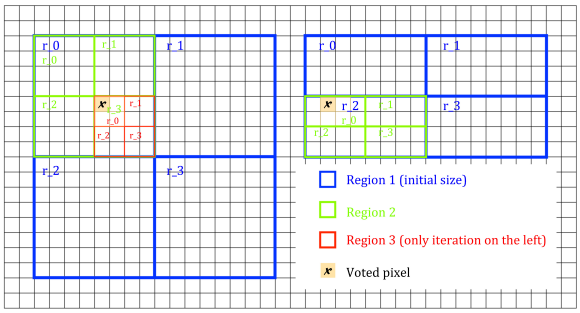


Fig. 2: A squared vote on the left and a rectangular vote on the right. R_0 in blue, R_1 in green and R_2 in red. The subregions of every step are marked as r_i with the same colour as the step they belong to.

After a user-defined number of votes, the BCT is smoothed with a small Gaussian kernel with $\sigma = 2$ and then normalised. Smoothing the BCT removes the effects of noise in the voting process and helps to find the true shape of the extracted blobs. As mentioned in [20], intermediate shape priors can yield discriminative shape structures; these structures can improve recognition tasks.

The BCT can be thought as a coarse-to-fine search through scale spaces for the true derivative of the image. Every sub-division 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 on the image. The use of rectancles benefits affine locallity; 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

Algorithm 1 The Brightness Clustering Transform

integ_Im = calculate integral of input_Im
output_Im = initialise output image

for $vote = 1$ **to** max_votes **do**

First Step:
Init region R:
$width = 2^{rand(rangeMin, rangeMax)}$
$height = 2^{rand(rangeMin, rangeMax)}$
$x = rand(0, imWidth - width)$
$y = rand(0, imHeight - height)$

Second Step:
while $width > 2$ & $height > 2$ **do**

divide R in 4 subregions r_0, r_1, r_2 and r_3
$r_{max} = \max(r_0, r_1, r_2, r_3)$ *(use integ_Im)*
$R = r_{max}$ this implies:
$width = width/2$
$height = height/2$
$x = x_{r_{max}}$
$y = y_{r_{max}}$
end while

Third Step:
$loc = (\text{round}(x + width/2), \text{round}(y + height/2))$
$\text{output\_Im}(loc) = \text{output\_Im}(loc) + 1$

end for

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.

In our experiments we discovered that amounts ranging from 5×10^4 to 1×10^5 votes are enough to extract the blobs in a 1024×768 image. We also noted that amounts as small as 2×10^4 votes can extract most of the significant blobs on the same image. The most commonly used values for the width and height of the rectangles range from 2^3 to 2^7; this range may be modified depending on the size of the image and the size of the blobs to be extracted.

So far the bright blobs are extracted by finding the sub-regions with the biggest sum of pixels; if we want to find dark blobs we could either modify equation 4 to be

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

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

2.2 Ridge Detection

Ridges are one-dimensional blobs. The BCT can be modified to extract this kind of features. Half of the votes are obtained with the width set constant and equal to two, and the height again set to a power of two; this time, the rectangles are only divided in two parts along the height. The other half of the votes are obtained in the same manner, but setting the height to be constant and equal to two and varying only the width.

This operator yields a new fast ridge detector that can be used for any kind of applications from matching to extraction and segmentation.

2.3 Operator Invariance

The scale invariance of the BCT, is related to the size of the initial rectangles; as the voting process goes on, the centres of the blobs are found. The random aspect ratio of the rectangles helps to disperse the votes near the centre of the features and not only cluster them at the exact centre.

Changes in illumination are considered as multiplicative noise. We modify equation 4 as follows.

$$R_{t+1} = \arg\max_{r_i} \sum_{x,y \in r_i} K(x,y) Im(x,y). \tag{6}$$

Here, $K(x,y)$ is the illumination function. If the illumination function can be considered to be approximately constant *i.e.* $K(x,y) \approx k$ in the region covered by R_0, then k can be extracted from the equation and we see that the operator is invariant to approximately constant changes in illumination.

From the same logic we obtain that the BCT is invariant to noise with zero-mean distributions as it cancels itself in the sub-regions r_i.

3 Locally Contrasting Keypoints

The Locally Contrasting Keypoints (LOCKY) are blob keypoints extracted directly from the BCT of an image. After the normalisation process, the BCT is thresholded; the set of all the connected components in the binary image, are the detected blobs. Figure 3 describes the process for detecting the LOCKY blobs in an image.

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_1, $f_2, \ldots,$ f_N); the mean is the centre of the feature (eq. 8).

$$Q = \frac{1}{N-1} \sum_{n=1}^{N} (f_n - \bar{F})(f_n - \bar{F})^T. \tag{7}$$

$$\bar{F} = \begin{bmatrix} \mu_x \\ \mu_y \end{bmatrix}. \tag{8}$$

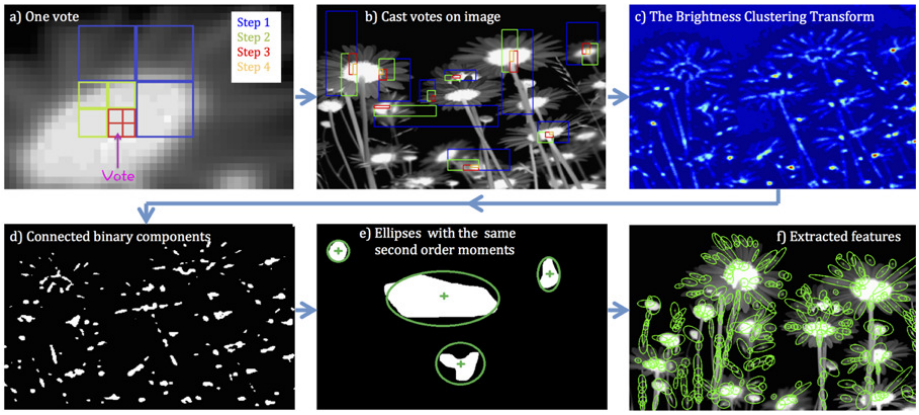


Fig. 3: The process for extracting the LOCKY features starts with the voting process to obtain the BCT. The transform is thresholded and the set of connected components is then extracted. The LOCKY features are the ellipses with the same second order moments as the connected binary components.

The eigenvalues of the sample covariance matrix Q (eq. 7) represent the size of the axes of the ellipse with the same second order moments; the eigenvectors define the direction of the axes. In practice, the eigenvalues of Q are scaled up by a factor of five to enlarge the size of the features.

This step is similar to the ellipses of the Maximally Stable Extremal 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.

4 Results

The Oxford dataset [9] (figure 4) presents a good challenge for interest point detection and it is now 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 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 [12] and, the *Maximally Stable Extremal Regions* [6]. In the non-affine-covariant group, we use the BRISK detector [15], the SURF (fast-Hessian) detector [11] and, CenSurE [7] (known as the STAR detector in *openCV*). 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.

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

Fig. 4: The first images of the sequences in the Oxford Dataset [9].

homography matrix; comparing correspondences of features and measuring how well the region detected on the images overlap. For more information on this measure see [9]. We use the correspondences with 40% overlap. To be able to compare LOCKY 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.

Detector	Time Factor	Type	Affine
Fast-Hessian [11]	4.53	Blobs	×
BRISK [15]	0.98	Corners	×
CenSurE [7]	0.72	Blobs	×
LOCKY1	1	Blobs	✓
LOCKY2	4.93	Blobs	✓
MSER [6]	3.26	Regions	✓
Hessian-Affine [12]	6.04	Blobs	✓
Harris-Affine [12]	8.53	Corners	✓

Table 1: The average factor of time for a set of 127 images. The images were converted to grayscale with 1024×768 pixels.

The timing results shown in table 1, were obtained using the OpenCV implementations of the algorithms (using mex files in MATLAB) in a computer with a 2 GHz $i7$ processor. Note that BRISK uses a multi-scale version of the AGAST detector which is a faster implementation of the FAST detector; LOCKY has similar timings while being able to provide information on affine transformations of the features.

Since the detection of the LOCKY features is based in the BCT which is a non-deterministic transform every run matches different features. The results

shown in Figure 4 (at the bottom) include the mean and variance over 100 runs of the test, showing that on average the LOCKY approach can deliver excellent performance: it is not dependent on initialisation and competes with those techniques already available and with its own advantages, especially speed. LOCKY detector might appear better on rectilinear structures (the wall) than on objects (the boat) and this can be investigated further.

5 Discussion

We presented a new algorithm that (via a novel non-deterministic low level operator, the Brightness Clustering Transform), can detect blobs and extract information about image transformations including affine changes in very small amounts of time. From the results we conclude that non-deterministic algorithms can help accelerate the search through different scale spaces. Although the literature contains a good amount of information on affine-covariant local feature detectors, the pairing of these with the set of descriptors available, has to be researched in more depth to reach the robustness necessary for real-world tasks. The term *invariant* imposes a hard restriction on perfect measurement after image transformations; on the other hand, the term *covariant* allows for a more flexible definition of measurement.

While LOCKY uses only rectangles as the shape for the search of blobs, the use of different polygons (using rotated integral images) can be an advantage in terms of precision on the detection although it could cause an increment in the running time. The detection of ridges is shown as a potential application and can also be extended and used for feature detection. A texture is a repetitive pattern, therefore, two votes will progress in a similar manner if both are wholly contained in a single texture; the BCT can be extended for fast analysis of textures.

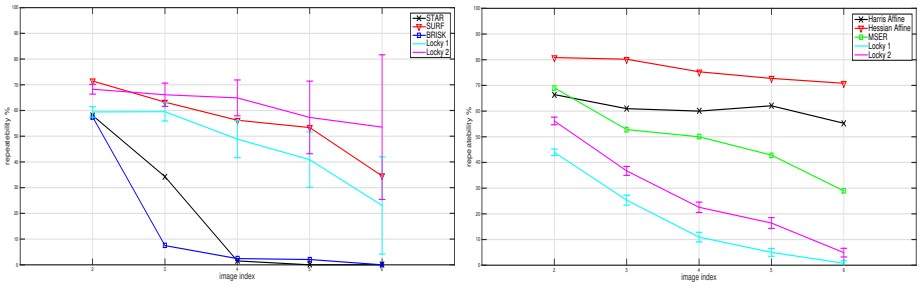
Robustness means persistance of features through image transformations. Although the non-deterministic nature of LOCKY causes a dificulty to achieve high repeatability scores, the results show that LOCKY is consistent. Also, its repeatability score tends to decay slower than the score of other approaches.

Shape is an important factor, Dickinson and Pizlo [20] talk about the perception of shape and explain how objects are composed of intermediate shape priors. The detection of shape has to be researched in more depth to achieve more domain-independent object recognition.

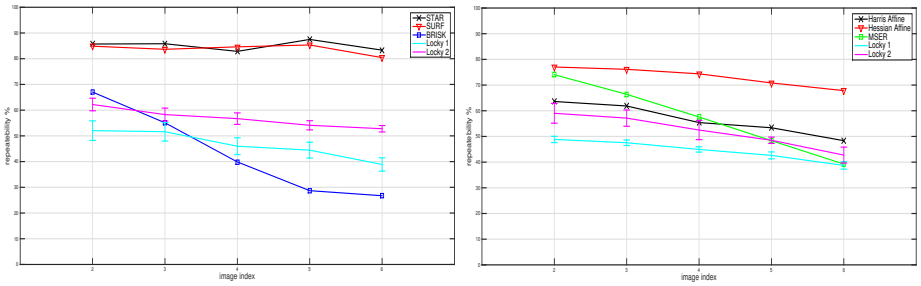
References

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 Alvey Vision Conference, vol. 15, p. 50. (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).

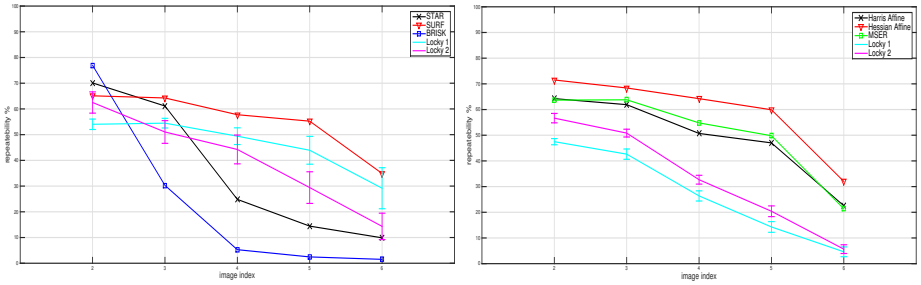
4. Rosten, E., Drummond, T.: Machine learning for high-speed corner detection. In ECCV, pp. 430–443. Springer (2006).
5. Mair, E., Hager, G. D., Burschka, D., Suppa, M., Hirzinger, G.: Adaptive and generic corner detection based on the accelerated segment test. In ECCV, pp. 183–196. Springer (2010).
6. 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).
7. Agrawal, M., Konolige, K., Blas, M. R.: Censure: Center surround extremas for realtime feature detection and matching. In ECCV, pp. 102–115. Springer (2008).
8. Gunn, S. R.: On the discrete representation of the Laplacian of Gaussian. In Pattern Recognition, vol. 32, pp. 1463–1472. Elsevier (1999).
9. 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).
10. Viola, P., Jones, M.: Robust real-time object detection. In IJCV, vol. 4, pp. 34–47. Springer (2001).
11. Bay, H., Tuytelaars, T., Van Gool, L.: Surf: Speeded up robust features. In ECCV, pp. 404–417. Springer (2006).
12. Mikolajczyk, K., Schmid, C.: An affine invariant interest point detector. In ECCV, pp. 128–142. Springer (2002).
13. Calonder, M., Lepetit, V., Strecha, C., Fua, P.: Brief: Binary robust independent elementary features. In ECCV, pp. 778–792. Springer (2010).
14. Rublee, E., Rabaud, V., Konolige, K., Bradski, G.: ORB: an efficient alternative to SIFT or SURF. In ICCV, pp. 2564–2571. IEEE (2011).
15. Leutenegger, S., Chli, M., Siegwart, R. Y.: BRISK: Binary robust invariant scalable keypoints. In ICCV, pp. 2548–2555. IEEE (2011).
16. Alahi, A., Ortiz, R., Vandergheynst, P.: Freak: Fast retina keypoint. In CVPR, pp. 510–517. IEEE (2012).
17. 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 Soc. (1980).
18. Heinly, J., Dunn, E., Frahm, J.-M.: Comparative evaluation of binary features. In ECCV, pp. 759–773. Springer (2012).
19. Jonides, J., Irwin, D. E., Yantis, S.: Integrating visual information from successive fixations. In Science, vol. 215, pp. 192–194. (1982).
20. Dickinson, S. J., Pizlo, Z.: Shape perception in human and computer vision. Springer (2013).



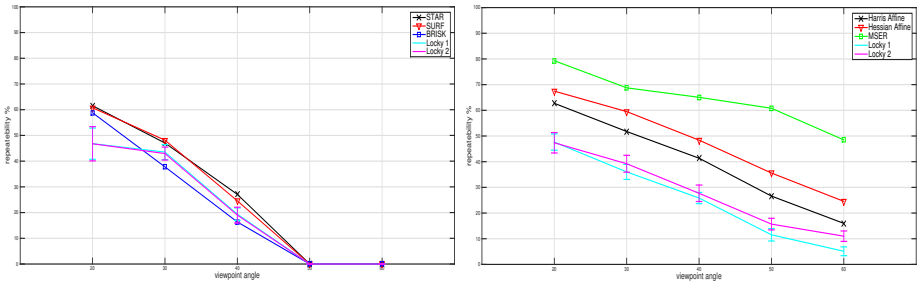
(a) Bark sequence



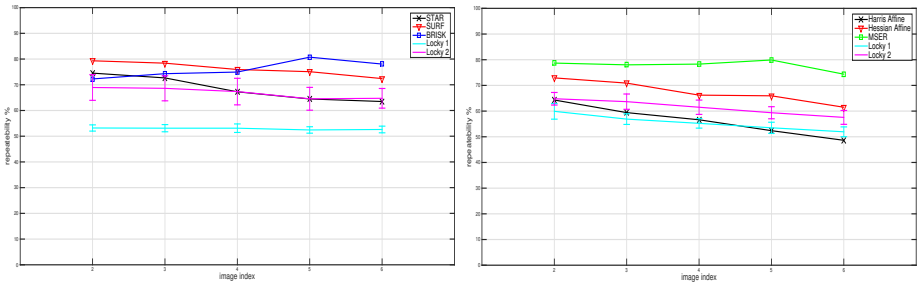
(b) Bikes sequence



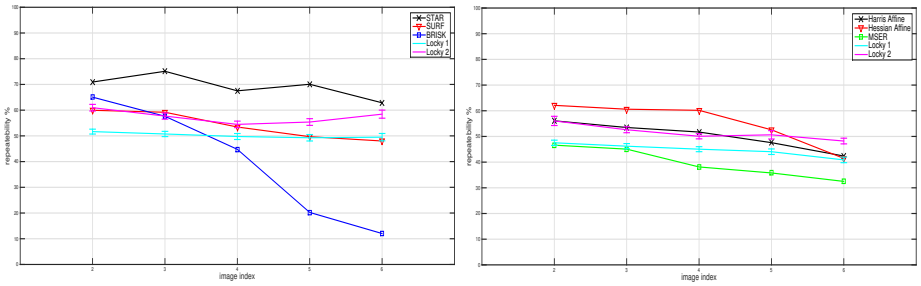
(c) Boat sequence



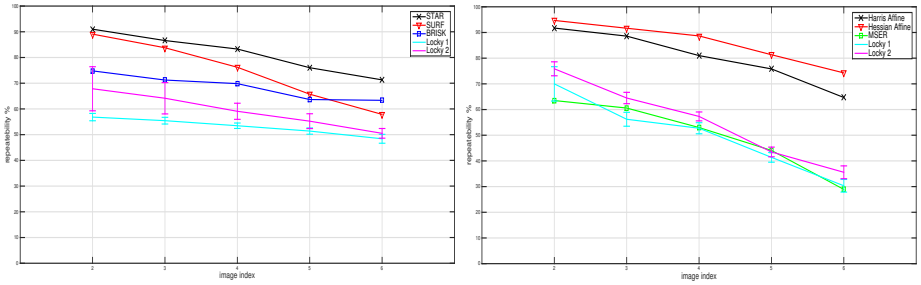
(d) Graf sequence



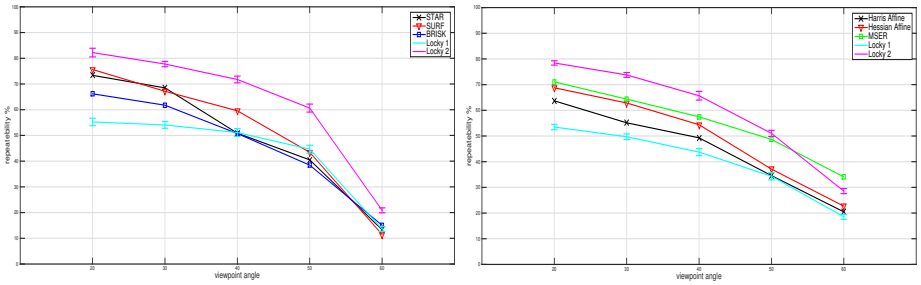
(e) Leuven sequence



(f) Trees sequence



(g) UBC sequence



(h) Wall sequence

Fig. 4: The repeatability test presented in [9]. 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.