Detecting Regions from Single Scale Edges

Konstantinos Rapantzikos, Yannis Avrithis, and Stefanos Kollias

National Technical University of Athens,
School of Electrical and Computer Engineering
{rap,iavr}@image.ntua.gr,stefanos@cs.ntua.gr

Abstract. We believe that the potential of edges in local feature detection has not been fully exploited and therefore propose a detector that starts from single scale edges and produces reliable and interpretable blob-like regions and groups of regions of arbitrary shape. The detector is based on merging local maxima of the distance transform guided by the gradient strength of the surrounding edges. Repeatability and matching score are evaluated and compared to state-of-the-art detectors on standard benchmarks. Furthermore, we demonstrate the potential application of our method to wide-baseline matching and feature detection in sequences involving human activity.

1 Introduction

Local features and descriptors are very successful in providing a compact representation for image matching, with applications to registration, wide baseline matching, retrieval, object recognition and categorization [1]. Affine-covariant regions are among the most popular features, being robust to occlusion and viewpoint changes. Their performance is typically measured by *repeatability* and *matching score* under a number of transformations. According to the study by Mikolajczyk *et al*. [2], the best performing detectors have been the *maximally stable extremal region* (MSER) [3] and *Hessian-affine* [4] detectors. More recent studies have identified a number of weaknesses in existing detectors or suggested new measures or desired properties like *compactness* [5] or *image coverage* [6].

We make an attempt to address a number of these issues in this work, while at the same time paying particular attention to a fast implementation, achieving competitive performance and in certain cases outperforming the state of the art. Surprisingly, our starting point is quite simple and, in a sense, one of the least expected: *single scale edges*. While powerful in providing an abstraction mechanism for many computer vision tasks, edges have not been quite successful in affine region detection so far. One example is *edge-based region* (EBR) detector [7], performing among the lowest in [2]. On the other hand, mainly motivated by Lindeberg's theory [8], one of the key ingredients of many region detectors is *scale selection*. Notable examples are the Hessian-affine and Lowe's SIFT [9] detectors, while the MSER detector's single scale operation may be connected to region density issues and the effect of blur [6]. The question is

then, *how can we use a fragile structure like a set of single scale edges to detect repeatable features?*

We choose to start from the binary map of the finest scale edges and detect local maxima of its *Euclidean distance transform* (EDT). In areas of uniform intensity and in the absence of spurious edges, such points are expected to lie on the interior of *blob-like regions* or close to *ridges* [10]. It then makes sense to use them as region seeds, much like local intensity extrema are employed in the *intensity extrema-based region* (IBR) detector [11]. On the other hand, while spurious structures appear almost everywhere in the finest scale, their *edge strength* is informative of their potential evolution in scale space. This is justified by the monotone decrease of edge strength with scale, as a result of smoothing, and subsequent decrease in the amplitude of local variations [10].

We therefore *greedily merge local maxima of the distance transform, guided by the strength of surrounding edges*. If our assumption is correct, this process will be similar to iteratively removing spurious edges and eventually *reproduce the effect of scale space evolution, by merely manipulating a sparse set of points*. The merging process is the one used in graph-based image segmentation [12]. However, instead of constructing a disjoint partition, we select multiple overlapping regions according to a number of different criteria during merging.

Finally, a region is defined by a selected set of maxima and a surrounding set of points, to which an ellipse is fitted. This is again similar to IBR detector, where the intensity pattern along rays emanating from seed points is analyzed. The main difference here is that a selected region corresponds to the *union* of a set of initial seeds, thus adapting to areas of *arbitrary shape*, much like MSER. Further, in contrast to EBR and other edge related methods, *disconnected edges* due to weak responses and thresholding are not an issue here, since the method automatically selects appropriate regions. Although not affine-covariant by construction, the performance of our detector is quite competitive in relevant benchmarks. We also qualitatively evaluate our approach in wide-baseline matching on sequences involving human activity.

2 Related Work

Recent advances on region detectors have focused on a number of properties or criteria while keeping repeatability close to the state of the art. We follow a similar structure in our analysis below.

An important property is *speed*, and has been particularly investigated for detectors based on the *Laplacian of Gaussian* (LoG) and subsequent scale selection [8], as well as affine adaptation using the *second moment matrix* [13] or the *Hessian matrix* [4]. An example is the approximation of LoG with *difference of Gaussians* (DoG) in SIFT detector [9], also seen as a *center-surround* operation in CSDD detector [14]. Other examples are the approximations in image space using *integral images* for rectangular region approximation of the Hessian matrix as in SURF detector [15], or even octagonal region approximation of the DoG in

CenSurE detector [16]. Clearly, the increased speed comes at the cost of certain performance drop due to the coarser approximation of region shape.

The MSER detector has the ability to fit regions of arbitrary shape. Building on the entropy-based *salient region* detector of Kadir and Brady and its affine adaptation of [17], the *structure guided salient region* (SGSR) detector of [5] supports this view, by employing MSER regions as seeds. On the other hand, MSER typically detects a small number of regions that, though repeatable, sample only a small part of the image. Clearly, this is problematic when *e.g.* the limited set of detected regions in an image is occluded in another. *Stable affine frames* (SAF) [6], extend to possibly unstable regions in an attempt to increase *image coverage*. The latter is, informally, the spatial extent of detector responses for a given number of correspondences and has been evaluated by visual inspection.

The performance of many detectors drops significantly when they are requested a small number of features. This is particularly true for the Hessian-affine detector, which densely detects regions of different scales or orientations at the same location. Clearly, a compact representation is crucial in large scale applications where computation and memory requirements are of primary importance. *Compactness* has been identified in [5] as the ability of a detector to preserve a competitive performance (measured *e.g.* in repeatability or matching score) under such limitations.

Figure-ground segmentation is discussed in Collins and Ge [14], where the idea is to detect not only blobs of uniform intensity but *groups* of such blobs as well; this is illustrated in the example of the yin-yang symbol in [14], which we repeat for our detector in section 3. *Feature grouping* is another line of research typically considered at higher levels of analysis and often during matching, *e.g.* the *semi-local* approaches of [18], [19], hyperfeatures [20] and many others. However, if appropriately integrated in a detector it *is* related to repeatability performance. *Segmentation*, like edge detection, has often been considered an abstraction mechanism that is not stable enough for the purpose of repeatable region detection. The recent experiments of Koniusz and Mikolajczyk [21] do not look promising either. Higher-level tasks typically combine multiple segmentations [22]. However, we claim that *selecting multiple overlapping regions during the segmentation process can be repeatable.*

3 The algorithm

Given an image, we would like to decompose it into a set of sparse but repeatable and interpretable features that will represent well the underlying local structure and will not focus only on corners or homogenous regions. The work of Tuytelaars *et al.* has highlighted several drawbacks in using edges for feature detection and this is why they have proposed the IBR detector [11] as a complementary to EBR [7]. We believe that the potential of edges in local feature detection has not been fully exploited and propose a detector that overcomes most of the weaknesses of previous work. In particular, the distance transform can help

recover from broken edges, and relaxed edge grouping can help recover from partially undetected edges or other topological changes across images.

Inspired by [14], Fig. 1 presents regions detected in an image of the yin-yang symbol by our detector along with other common detectors, namely Harris-affine and Hessian-affine [4], Lindeberg's primal sketch [8], MSER [3] and SIFT [9]. It is clear that with a very small number of detected regions all elements of the figure are captured at all scales by our detector. The large circular region is included despite the intensity variation. We will show in section 4 that this property is achieved without sacrificing repeatability.

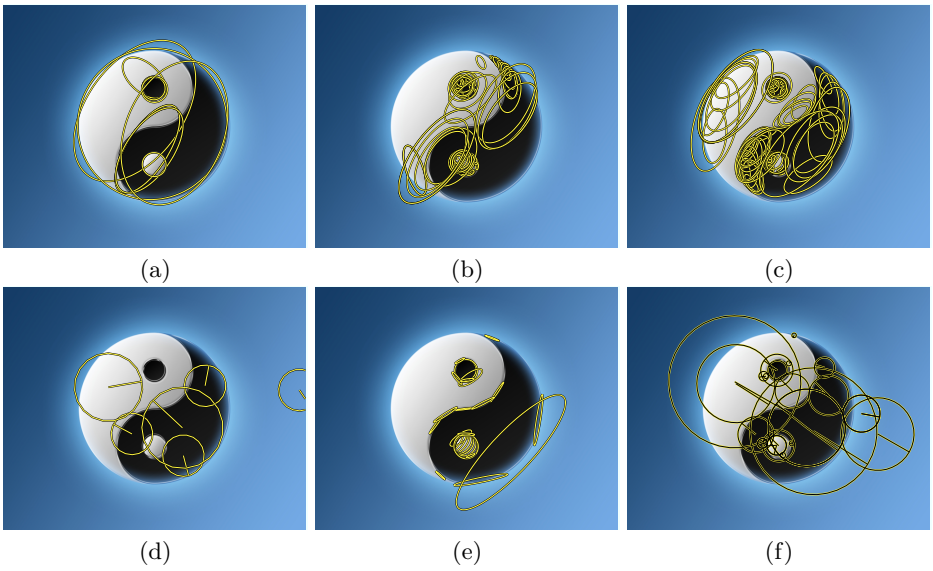


Fig. 1. Detection results for different detectors. (a) DistDetector (#12); (b) Harris-affine (#102); (c) Hessian-affine (#125); (d) Primal sketch (#6); (e) MSER (#27); (f) SIFT (#25). The number in parentheses indicate the number of detected features.

3.1 Image decomposition

We will use the same example of the yin-yang symbol to illustrate the main steps of the algorithm. Given an input image I defined on a lattice Q as shown in Fig. 2a, we apply Canny *edge detection* [23] to get a binary edge map. Let $F \subset Q$ be the set of pixels on the edge map. If I is convolved with a Gaussian derivative ∇G_σ of scale σ, let $g = \|\nabla G_\sigma \star I\|$ be the relevant gradient magnitude, signifying the *edge strength*. A clean edge map is not crucial. On the contrary, we set the scale parameter σ and the hysteresis thresholds τ_h, τ_ℓ of the detector such that all distinctive image structures along with a large number of spurious edges are detected, as in Fig. 2b. This edge map corresponds to a fine scale and we will use g to reproduce coarser scales in subsequent steps.

The next step is to compute the *Euclidean distance transform* D on the edge map. That is, for each $p \in Q$, the distance $D(p)$ to the nearest image edge:

$$D(p) = \min_{q \in F} (\|p - q\|), \quad p \in Q. \tag{1}$$

We use the linear-time algorithm Felzenszwalb and Huttenlocher [24] to compute the EDT, as in Fig. 2c. We then detect the *local maxima* of D, typically lying on the interior of blob-like regions or ridges. The set V of the maxima is obtained using gray-scale morphological gradient:

$$V = \{p \in Q : (D \oplus B)(p) - D(p) = 0\}, \tag{2}$$

where $(D \oplus B)$ denotes gray-scale dilation of D by structuring element B. In most cases V is a sparse set of points as shown in Fig. 2d, superimposed on the EDT. A large number of maxima lies between spurious edge fragments, and a merging process on the maxima will be equivalent to iteratively removing the fragments in non-decreasing order of gradient g.

We represent geometry using the *Delaunay triangulation* of the initial set V of maxima, as shown in Fig. 2e. Seeing points of V as vertices in the Delaunay graph G such that $G = (V, E)$, two adjacent vertices $u, v \in V$ will correspond to two neighboring local maxima of D. If there is no edge fragment lying between u, v, this means that u and v are likely to lie within the same region or along a ridge; otherwise, the strength of the edge fragment will determine at which iteration u and v will be merged, which would be equivalent to removing the fragment.

We capture this information in the graph by assigning a weight to each Delaunay edge, observing its intersections with image edges. Formally, for each Delaunay edge $e = (u, v) \in E$, let $\ell(e)$ denote the set of points in Q lying on a linear segment from u to v (*e.g.* by line drawing on Q). Define the *intersection set* $S(e) = \ell(e) \cap F$ as the set of points of this segment intersecting with the binary edge map of I. These points are shown as green dots in Fig. 2e. Then, the *weight function* $w : E \to \mathbb{R}$ assigns a weight to e that is equal to the maximum strength found along $S(e)$:

$$w(e) = \max_{p \in S(e)} g(p), \quad e \in E. \tag{3}$$

This weight signifies that vertices u, v will be merged when the strongest edge fragment between them is removed. Equipped with weight function w, the Delaunay graph becomes a *weighted undirected graph*.

3.2 Feature detection

To reproduce the effect of edge evolution in scale space, we sort graph edges $e = (u, v)$ by non-decreasing weight $w(e)$ and then iteratively merge vertices u, v in the same order. We adopt the merging process used in the *graph segmentation* method of Felzenszwalb *et al.* [12]. Using a structure of disjoint sets, the output

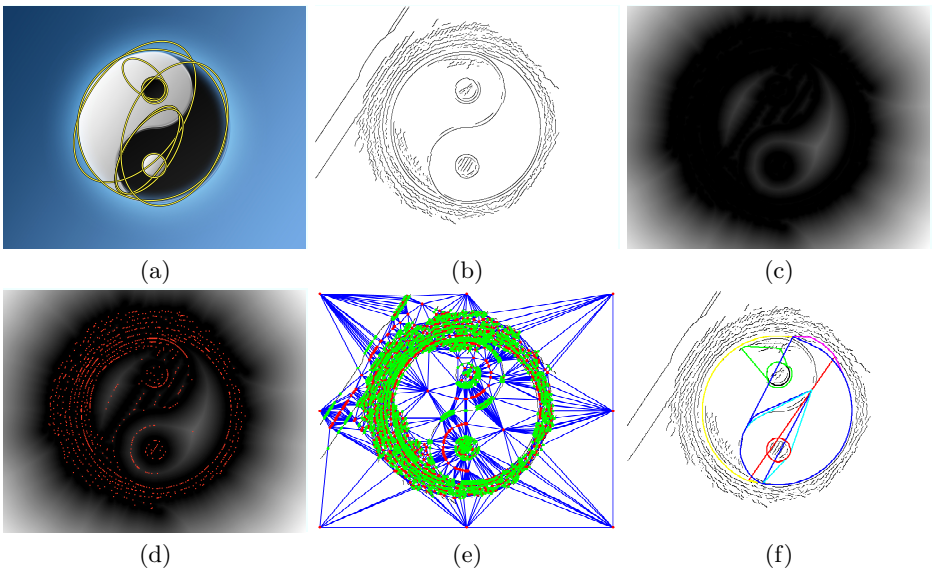
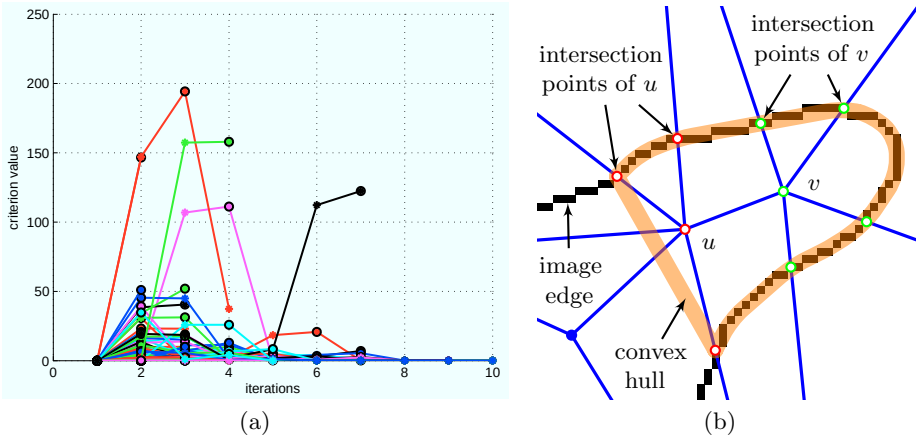


Fig. 2. Detection steps for a yin-yang symbol. (a) detected features; (b) edge map F; (c) distance map D; (d) local maxima V; (e) Delaunay triangulation G. Green dots correspond to intersecting points (see text); (f) convex hulls (in color).

is a partition of graph G into components. The process is similar to Kruskal algorithm for minimum spanning trees, with the addition of a merging criterion controlling the grouping of components. The algorithm decides which edges to cut by comparing the minimum weight edge connecting two components with the maximum weight within each component. To avoid over-segmenting, the algorithm includes a penalty $\tau(C) = k/|C|$ on each component C, where $|C|$ is the order of C and k is a constant. In general, the larger the penalty, the coarser the resulting partition. Nevertheless, this penalty does not impose a fixed size limit.

We are not interested in the final partition of the graph here because we target *multiple overlapping regions* rather than a disjoint set. Any component produced during the merging process is a candidate for producing a region. Let $\mathcal{C}$ be the set of all components corresponding to the final partition, and t an iteration of the merging process. For each $C \in \mathcal{C}$, we denote by C^t the instance of C at iteration t. Apparently, C^t grows larger with increasing iteration until it is merged into another component. When component C_i^t is merged into C_j^t, then C_j^{t+1} grows to become the union of the two, while C_i stops growing and remains constant until termination.

Setting a constant scale of observation k, the idea is to define a *measure* $\mu(C^t)$ of each component $C \in \mathcal{C}$ at each iteration t and keep track of its evolution during the process. For instance, Fig. 3a shows the change of $\mu(C^t)$ when measure μ is defined as the *compactness* of the region shape corresponding to component C^t. When there is a significant change in the topology of a component, we expect a



(a) (b)

Fig. 3. (a) Indicative distribution of the weights assigned to components. Each color represents the evolution of a single component and the dark circle indicates the selected weight; the selected values by black circles, which correspond to the global maxima. (b) Graph vertices and corresponding intersection points. The orange line is not the actual convex hull, but a sketch for illustrative purposes.

peak in the corresponding curve. Hence, we choose to *select* a component C^t if iteration t corresponds to the maximum peak in its lifetime.

If $\hat{\mathcal{C}} \subset \mathcal{C}$ is the subset of *selected* components, what remains is to construct an image region R corresponding to each $C \in \hat{\mathcal{C}}$. The set $\mathcal{R}$ of such regions will be the output of our region detector. To compute the actual spatial extent of a region, we use the edge fragments surrounding the vertices of the component, as depicted in Fig. 3b. Given a vertex $u \in V(C)$ and an adjacent edge e with a non-empty intersection set $S(e)$, define the *nearest intersection point*

$$n(u, e) = \arg \min_{p \in S(e)} \|p - u\|. \tag{4}$$

For instance, vertex u and its nearest intersection points are shown in red in Fig. 3b, while v and its nearest intersection points are shown in green. Then, given a component $C \in \hat{\mathcal{C}}$, we collect all its vertices $u \in V(C)$ and all their nearest intersection points to construct its *neighborhood*

$$N(C) = \bigcup_{u \in V(C)} \bigcup_{e \in E(u)} n(u, e). \tag{5}$$

Referring to Fig. 3b, and since vertices u, v have been merged to one component, the neighborhood comprises all red and green intersection points. Finally, we compute the *convex hull* of all intersecting points, as depicted by the thick orange line in Fig. 3b and by the colored lines in Fig. 2f, and *fit ellipses*. All steps of our region detector are summarized in Algorithm 1.

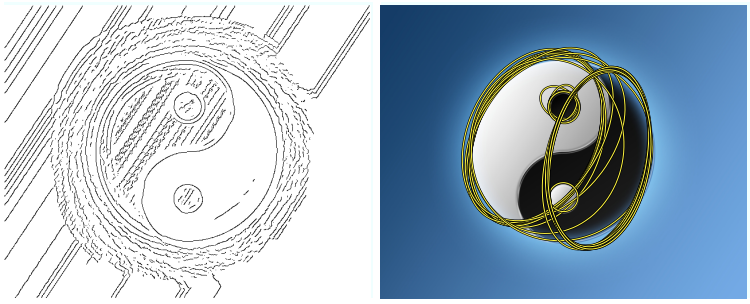
Algorithm 1 Distance Transform Detector

procedure DTD(image I, regions $\mathcal{R}$)
  $g \leftarrow \|\nabla G_\sigma \star I\|$ ▷ gradient magnitude at scale σ
  $F \leftarrow \textsc{EdgeMap}(I)$
  $D \leftarrow \textsc{DistanceTransform}(F)$
  $V \leftarrow \textsc{LocalMaxima}(D)$
  $G \leftarrow \textsc{DelaunayTriangulation}(V)$
  $\textsc{Sort}(E,w)$ ▷ sort edges by non-decreasing weight
  $t \leftarrow 0$
  $\mathcal{C} \leftarrow V$
  for all $e = (u, v) \in E(G)$ **do**
    $t \leftarrow t + 1$
    if $w(e) \leqslant \min\{\rho(C^{t-1}(u)), \rho(C^{t-1}(v))$ **then** ▷ check penalty
      $C^t \leftarrow \textsc{Merge}(C^{t-1}(u), C^{t-1}(v))$ ▷ merge comp. of v into comp. of u
      $\rho(C^t) \leftarrow w(e) + k/|C^t|$ ▷ update penalty
    end if
  end for
  $\mathcal{R} \leftarrow \emptyset$
  for all $C \in \mathcal{C}$ **do**
    $t \leftarrow \arg\max_s(\Delta\mu(C^s))$ ▷ iteration where measure change is maximized
    $H \leftarrow \textsc{ConvexHull}(N(C^t))$ ▷ neighborhood $N(C)$ defined in (5)
    $R \leftarrow \textsc{FitEllipse}(H)$
    $\mathcal{R} \leftarrow \mathcal{R} \cup R$
  end for
  return $\mathcal{R}$
end procedure

4 Experiments

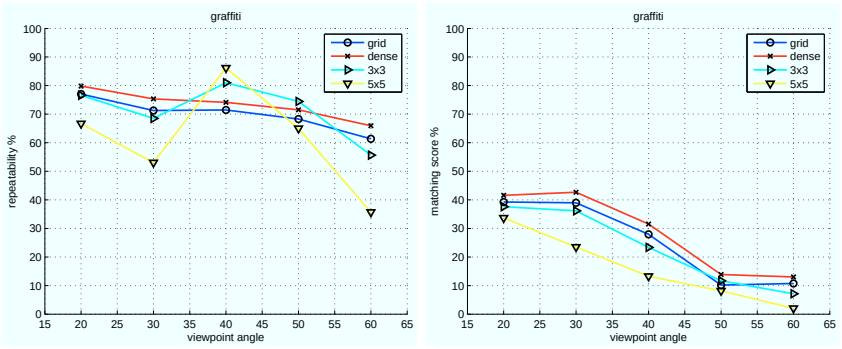
4.1 Tuning and experimental setup

We first carry out a set on experiments to study the influence of different parameters of our detector, hereafter termed as *DistDetector*. As *qualitative* parameters –mainly related to region shape– we identify the weights w of graph edges and the measure μ related to the evolution of graph components. Although different choices exist, we use the gradient magnitude g and the shape *compactness* of the component. Also, we define penalty $\tau(C) = k/|C|$ with $k = 0.25\min(r, c)$ where $r \times c$ is the size of input image I in pixels. As *quantitative* parameters – rather related to the trade-off between computational complexity and statistical performance– we identify the Canny hysteresis thresholds and the density of the local maxima of EDT. Fig. 4 shows detected regions hysteresis thresholds lower than those of Fig. 1. Although spurious edges result in more regions, similar image areas are covered. Fig. 5 shows the performance under four different sampling strategies for EDT local maxima. Overall, we use the "3×3" strategy as default in our experiments, achieving the best trade-off between computational complexity and performance.



$\tau_l = 0.15, \tau_h = 0.6$

Fig. 4. Robustness to hysteresis thresholds. Canny edges for a different set of hysteresis thresholds than the ones used in Fig. 2b and the corresponding detection results.



(a) (b)

Fig. 5. Comparison of the four methods to extract local maxima. (a) repeatability score(%); (b) matching score(%).

We then carry out a set of experiments focusing on the benchmarks proposed of [2] and region interpretability by visual inspection. We evaluate the statistical performance on image sequences *graffiti*, *boat* and *leuven*, with running time $2s$, $3s$ and $0.8s$ respectively. Apparently the running time is higher for strongly textured scenes like *boat*. Repeatability and matching score results of our detector are measured against results in [2]. For all other comparisons we use publicly available implementations with default parameters if not otherwise specified. We use the SIFT descriptor for all experiments.

4.2 Repeatability and matching score experiments

Figures 6, 7 compare our detector against the state of the art on four performance indicators: repeatability, number of correspondences, matching score and number of correct matches obtained with the Nearest-Neighbor (1-NN) strategy. As shown in Fig. 6, we obtain higher repeatability scores than all other detectors for the *graffiti* and matching scores above Hessian-affine. The repeatability is high for the *boat* and average for *leuven*, while the number of detected fea-

tures is low. On the other hand, the matching score is high for *leuven* and low for *boat*. Overall, DistDetector achieves a good trade-off between performance and number of features.

This good balance led us to carry out another experiment to comparing the ratio of correct to false matches of DistDetector, Hessian-Affine and MSER under two matching strategies, namely NN and similarity threshold (SIM). Both strategies have been used in [25]. Fig. 8 shows results for the *graffiti* pair $(0°, 30°)$. As expected, DistDetector performs quite well for a small number of total matches and ranks second after the MSER for the NN strategy. Interestingly enough, it outperforms both other detectors for approximately the same number of detected features under the similarity threshold strategy. The Hessian-affine detector with a small threshold produces a large number of features (2640), but still performs only slightly better than DistDetector.

Finally, in Fig. 9 we illustrate the qualitative performance of the proposed detector in feature matching using a set of three different image pairs. We perform spatial matching measuring RANSAC inliers after 10 runs. For the well known *graffiti* pair $(0°, 50°)$, shown in Fig. 9a, we get the following results: (a) DistDetector (#681 NN-matches, 32 inliers); (b) Hessian-Affine (#2352 NN-matches, 65 inliers) and (c) MSER (#780 NN-matches, 52 inliers). Fig. 9bc show inliers for two (non-neighboring) frames of a face sequence and a stereo pair of a gesture sequence respectively. A human face can be easily detected by its edges. The edge-based nature of DistDetector fits well with local feature detection from faces. Indeed we have shown that it does not depend on the quality of edges and provides a good estimate of feature scale regardless of the initial edge map.

5 Discussion

We have proposed a new feature detector based on single-scale edges. The detector performs competitively on established evaluation benchmarks and produces a compact set of interpretable and repeatable features. Visually, the detected features are more similar to the ones produced by MSER, but exhibit a higher overlap factor, a wider area coverage. The ability to detect composite regions is also very important, in contrast to uniform intensity regions of other detectors. Since all major algorithmic components, namely distance transform, Delaunay triangulation and merging, are still efficient to compute in 3D, it is straightforward to extend DistDetector to spatiotemporal data. It is our plan to pursue this direction and apply the extended detector to action detection using the framework of our early work [26].

References

1. Tuytelaars, T., Mikolajczyk, K.: Local invariant feature detectors: A survey. Foundations and Trends in Computer Graphics and Vision **3** (2007) 177–280
2. Mikolajczyk, K., Tuytelaars, T., Schmid, C., Zisserman, A., Matas, J., Schaffalitzky, F., Kadir, T., Gool, L.: A comparison of affine region detectors. IJCV **65** (2005) 43–72

3. Matas, J., Chum, O., Urban, M., Pajdla, T.: Robust wide-baseline stereo from maximally stable extremal regions. Image and Vision Computing **22** (2004) 761–767
4. Mikolajczyk, K., Schmid, C.: Scale & affine invariant interest point detectors. IJCV **60** (2004) 63–86
5. Fan, S., Ferrie, F.: Structure guided salient region detector. In: BMVC. (2008)
6. Perdoch, M., Matas, J., Obdrzalek, S.: Stable affine frames on isophotes. In: ICCV. Volume 104. (2007)
7. Tuytelaars, T., Van Gool, L.: Content-based image retrieval based on local affinely invariant regions. In: Visual Information Systems. (1999)
8. Lindeberg, T.: Feature detection with automatic scale selection. IJCV **30** (1998) 79–116
9. Lowe, D.: Distinctive image features from scale-invariant keypoints. IJCV **60** (2004) 91–110
10. Lindeberg, T.: Edge detection and ridge detection with automatic scale selection. IJCV **30** (1998) 117–154
11. Tuytelaars, T., Van Gool, L.: Wide baseline stereo matching based on local, affinely invariant regions. In: BMVC. (2000)
12. Felzenszwalb, P., Huttenlocher, D.: Efficient graph-based image segmentation. IJCV **59** (2004) 167–181
13. Lindeberg, T., Garding, J.: Shape-adapted smoothing in estimation of 3-d shape cues from affine deformations of local 2-d brightness structure. Image and Vision Computing **15** (1997) 415–434
14. Collins, R.T., Ge, W.: CSDD features: Center-Surround Distribution Distance for feature extraction and matching. In: European Conference on Computer Vision. (2008)
15. Bay, H., Tuytelaars, T., Van Gool, L.: SURF: Speeded up robust features. In: ECCV. (2006)
16. Agrawal, M., Konolige, K., Blas, M.R.: CenSurE: Center surround extremas for realtime feature detection and matching. In: ECCV. (2008)
17. Kadir, T., Zisserman, A., Brady, M.: An affine invariant salient region detector. In: ECCV. (2004) 228–241
18. Brown, M., Lowe, D.: Invariant features from interest point groups. In: BMVC. (2002)
19. Lazebnik, S., Schmid, C., Ponce, J.: Semi-local affine parts for object recognition. In: BMVC. (2004)
20. Agarwal, A., Triggs, B.: Hyperfeatures-multilevel local coding for visual recognition. In: ECCV. (2006)
21. Koniusz, P., Mikolajczyk, K.: Segmentation based interest points and evaluation of unsupervised image segmentation methods. In: BMVC. (2009)
22. Russell, B., Efros, A., Sivic, J., Freeman, W., Zisserman, A.: Using multiple segmentations to discover objects and their extent in image collections. In: CVPR. (2006)
23. Canny, J.: A computational approach to edge detection. PAMI **8** (1986) 679–698
24. Felzenszwalb, P., Huttenlocher, D.: Distance transforms of sampled functions. Technical report (2004)
25. Mikolajczyk, K., Schmid, C.: A performance evaluation of local descriptors. Pattern Analysis and Machine Intelligence **27** (2005) 1615–1630
26. Rapantzikos, K., Avrithis, Y., Kollias, S.: Dense saliency-based spatiotemporal feature points for action recognition. In: CVPR'09. (2009)

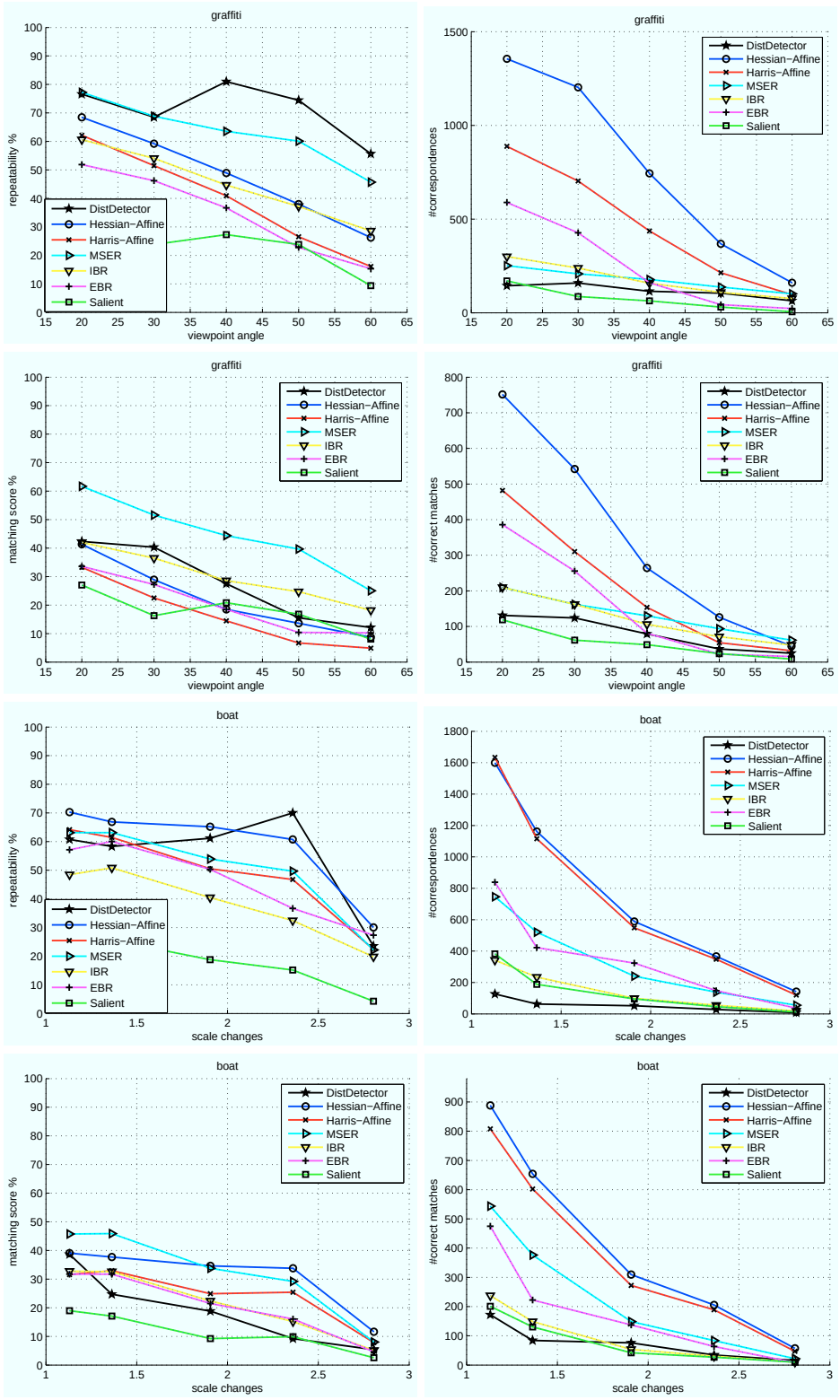


Fig. 6. Repeatability score, number of correspondences, matching score and number of correct matches for the *graffiti* and *boat* sequences.

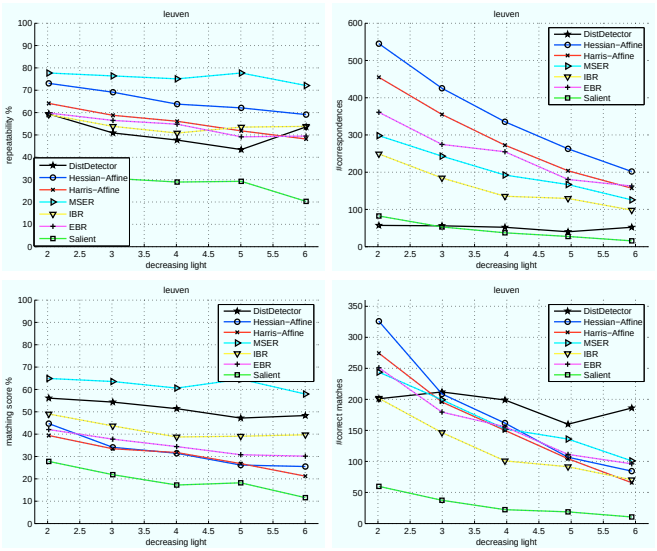


Fig. 7. Repeatability score, number of correspondences, matching score and number of correct matches for the *leuven* sequence.

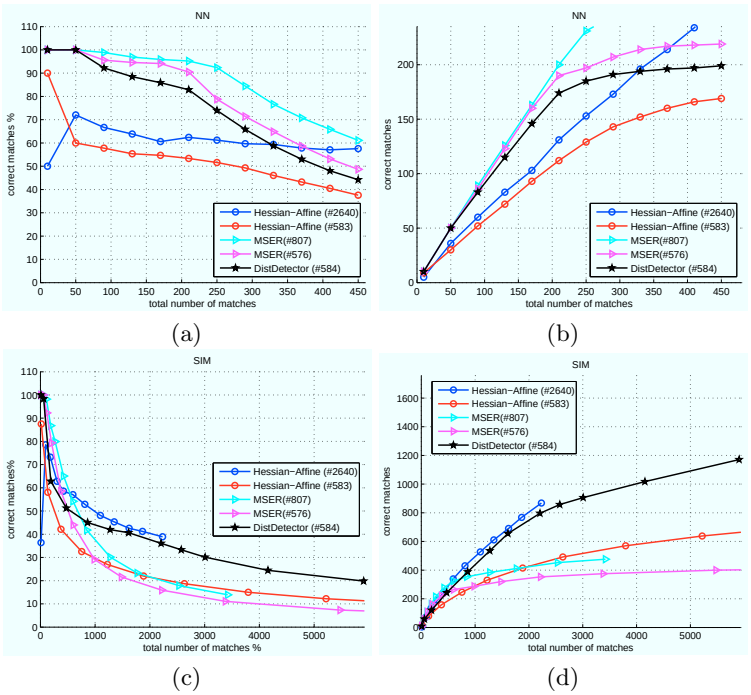
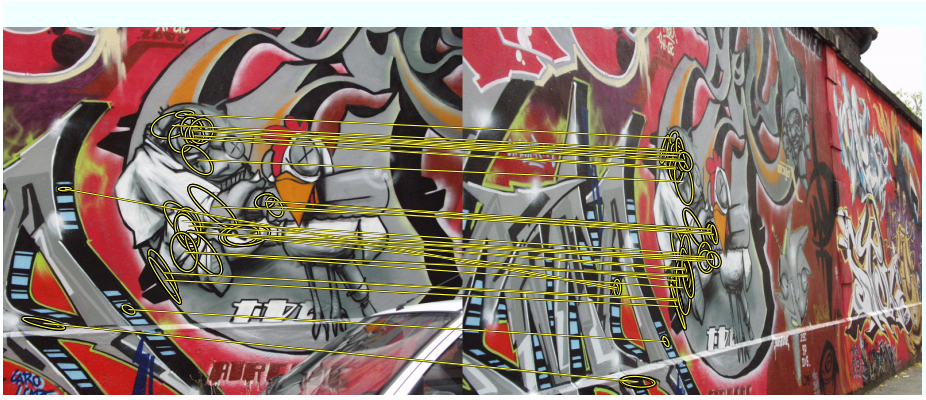
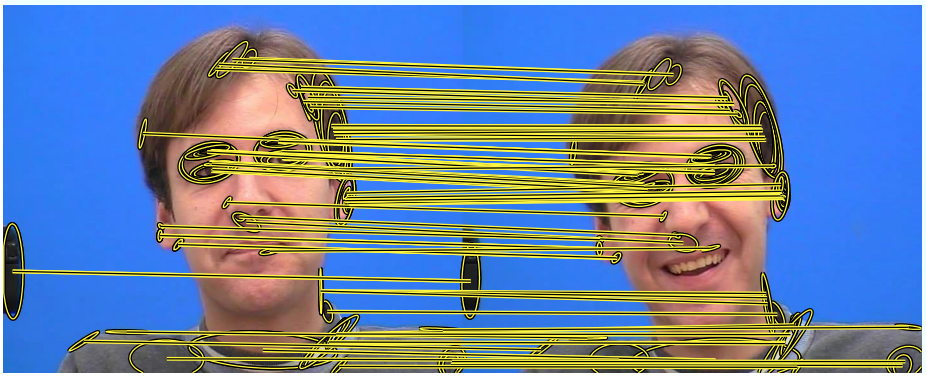


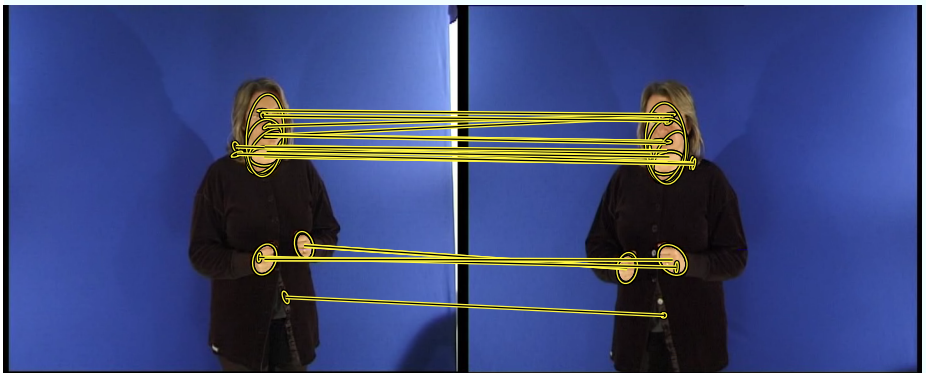
Fig. 8. Viewpoint change (Graffiti image pair at 0° and at 30°). Percentage of correct matches and number of correct matches versus total number of matches for Nearest-Neighbor (NN) and threshold-based (SIM) matching.



(a)



(b)



(c)

Fig. 9. RANSAC Inliers for the DistDetector for three different pairs of images. (a) Graffiti pair (0°, 50°); (b) two frames of a moving face sequence; (c) a stereo pair of a gesture sequence.