Keypoint Detection in Greyscale Images Based on Region Image Segmentation

Luka Lukač[1], Andrej Nerat[1], Damjan Strnad[1], Bogdan Lipuš[1], Filip Hácha[2], Borut Žalik[1]

[1]*Faculty of Electrical Engineering and Computer Science, University of Maribor*
[2]*Department of Computer Science and Engineering, University of West Bohemia*
E-mail: luka.lukac@um.si

Abstract

This paper proposes a new method for keypoint detection in greyscale images. In the beginning, an image is divided into segments using one of the established methods for the region image segmentation. The obtained segments are analysed in order to calculate the suitable number of keypoints. After that, pixels with highest gradients in their local neighbourhoods are extracted and proclaimed keypoints. The experimental results on real-life images indicate that the method extracts keypoints that adequately represent the image's structure.

1 Introduction

One of the major fields in image processing is detection of local features [1]. Nowadays, there are plenty of local feature detectors, which are utilised for many tasks, such as object recognition [2, 3, 4], image retrieval [5, 6], and texture recognition [7, 8]. The most basic elements in raster images that represent their local features are pixels. However, not all pixels carry an equal amount of information about the image structure. Pixels that represent important image features (e.g., edges and corners) usually stand out from their local neighbourhood. As they predominantly define the image structure, they are often referred to as keypoints (also known as key pixels).

Keypoint detection represents one of the oldest challenges for image analysis. Morevec proposed a simple corner detector as far back as 1977 [9]. In practice, however, it is rarely used due to its sensitivity to noise. The first widely used feature detector is the Harris corner detector [10], which extracts pixels with lowest auto-correlation values, and proclaims them keypoints. Its main downside is failing to process image scale changes [1]. In order to overcome the issues of the early keypoint detectors, Lowe introduced the scale-invariant feature transform (SIFT) [11]. The disadvantage of the approach is a poor time efficiency [12]. In the years after SIFT was proposed, numerous methods for keypoint detection were proposed, such as PCA-SIFT [13], adaptive thinning approach [14], GLOH [15], and FAST [16]. However, the real breakthrough in local feature detection was achieved with the Speeded Up Robust Features algorithm (SURF) [17] that is based on the approximation of the Hessian matrix. SURF achieved high time efficiency compared to SIFT, and, therefore, enabled feature detection in real-time applications [18]. In order to furtherly improve keypoint detection performance, various machine learning methods were proposed [19, 20, 21].

Our method performs the detection of keypoints on a regionally segmented image. Keypoints are detected separately in each image segment. After that, gradients in pixels' local neighborhoods inside the segment are calculated. In the end, pixels with the highest gradients in separate segments are proclaimed keypoints. The proposed method overcomes the issues of other methods for keypoint detection. It is fast, invariant to image rotation, and resistant to noise.

The remainder of the paper is structured as follows. In Section 2, the proposed method for keypoint detection based on the image segmentation is described, Section 3 presents the experiments and results, while Section 4 concludes the paper.

2 Keypoint Detection

Let I be a greyscale image, consisting of N pixels. The main goal of keypoint detection is to select $n \ll N$ pixels that adequately represent the structure of I. Our method calculates gradients between the current pixel and the pixels in its neighbourhood. This way, the significance of each pixel is evaluated – the larger gradients in the neighbourhood, the more significant the pixel is. Furthermore, if the distance between two pixels is larger, the impact on the evaluation value should be smaller. Size of the observed neighbourhood s is user-specified. The evaluation value e for the pixel $I(y, x)$ (located in the x-th column and y-th row) is calculated according to Equation 1.

$$e = \sum_{i=y-s}^{y+s} \sum_{\substack{j=x-s \\ j \neq x \vee i \neq y}}^{x+s} \left(\frac{|I(y,x) - I(i,j)|}{\sqrt{(x-j)^2 + (y-i)^2}} \right) \quad (1)$$

After all pixels in I are evaluated, n pixels with highest e values are proclaimed keypoints $p_i^k \in P^k;\ 1 \leq i \leq |P^k|$. However, if calculating the evaluations on a raw image, there is a possibility that pixels with highest evaluation values are located close to each other, and do not represent the entire image structure. To solve this problem, I is divided into regions using one of the established tech-

niques for region image segmentation (e.g., K-means [22] or DBSCAN [23]).

In the next step, keypoint detection is performed on segments $\mathcal{S} = \{S_i\};\ 1 \leq i \leq |\mathcal{S}|$ of the obtained image segmentation. The total number of keypoints n can be either user-given or determined using a calculation. A more convenient user input (instead of n) is the desired rate r of detected pixels, from which, the number of keypoints is calculated as $n = r \cdot N$. After that, the number of keypoints is calculated for each S_i according to its belonging number of pixels (given in Equation 2).

$$n_i = \lfloor r \cdot |S_i| \rfloor \tag{2}$$

An analysis of each segment is performed if the number of keypoints is calculated automatically. The information entropy H is calculated for each S_i in order to extract more keypoints in image segments with larger gradients, while still detecting keypoints in segments with smaller gradients. An automatic calculation of the keypoints' number n_i is performed according to Equation 3.

$$n_i = \begin{cases} \left\lfloor \frac{2^{H(S_i)}}{m} \cdot |S_i| \right\rfloor; & \frac{2^{H(S_i)}}{m} \leq 1 \\ |S_i|; & \text{otherwise} \end{cases} \tag{3}$$

where m represents an arbitrary non-negative factor.

The complete method for keypoint detection is explained in Algorithm 1. In Line 8, a region segmentation $\mathcal{S}$ is obtained from I. After that, the loop iterates through all segments $S_i \in \mathcal{S}$. The number of keypoints n_i is calculated either from a user-provided r (Line 11) or is determined automatically (Line 13 to Line 17). In Line 20, a loop iterates through all pixels of the current segment $p_{i,j} \in S_i$. The calculation of the pixel evaluation value e_j is performed in Line 21. After all pixels inside S_i are evaluated, the evaluation value vector E is sorted in a descending order. The final loop in Line 25 iterates through the first n_i pixels with the maximum evaluation values and adds them to the set of keypoints P^k. The final result of the algorithm is obtained after all segments inside $\mathcal{S}$ are processed.

Algorithm 1 Keypoint detection in a greyscale image.

```
1:  function KEYPOINT-DETECTION(I, s, r, m)
2:                                   ▷ I: a greyscale image
3:                ▷ s: size of the observed neighbourhood
4:                 ▷ r: the desired ratio between n and N
5:      ▷ m: magn. factor for automatic calculation of n_i
6:                ▷ Returns: set of the detected keypoints
7:      P^k ← {}
8:      S = ImageSegmentation(I)
9:      for i ← 1 ... |S| do
10:         if IsProvided(r) then
11:             n_i ← [r · |S_i|]
12:         else
13:             if 2^H(S_i) / m ≤ 1 then
14:                 n_i ← ⌊ 2^H(S_i) / m · |S_i| ⌋
15:             else
16:                 n_i ← |S_i|
17:             end if
18:         end if
19:         E ← {}
20:         for j ← 1 ... |S_i| do
21:             e_j ← EvaluatePixel(p_i,j)        ▷ Eq. 1
22:             E ← E ∪ {e_j}
23:         end for
24:         SortEvaluationsDescending(E)
25:         for j ← 1 ... n_i do
26:             P^k ← P^k ∪ {E_j}
27:         end for
28:     end for
29:     return P^k
30: end function
```

3 Results

The results of the proposed method are presented in this section. Keypoint detection was performed on a variety of images from the popular image dataset DIV2K [24]. In Figure 1, the comparison of keypoint extraction results is presented on the image of the famous Taj Mahal mausoleum according to different input parameters: the presence of the segmentation step and manual/automatic determination of r. The automatic calculation yielded the average value $r = 0.06768$, therefore, this value was used also as the manual input. In Figure 1(b), keypoint detection was performed on a raw image with the manually provided r. Although the basic image structure is recognisable, some of the important pixels, such as ones on the edge of the dome, were not detected. Keypoint detection on the segmented I is displayed in Figure 1(c). The equally distributed rate across segments turns out to be subpar, as many noisy pixels in the background are detected as keypoints. The visually best results are obtained with the segmentation of I and calculation of r separately for each S_i according to its H, as seen in Figure 1(d). This way, important keypoints, which describe the structure of I, are extracted.

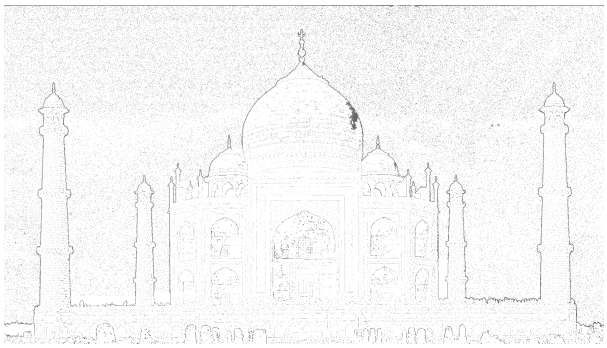
In Table 1, the results of keypoint detection on selected test images I are collected. Automatic calculation of r and region segmentation were used in order to obtain the best results. The extracted rate of pixels r depends on the properties of a separate image I: Child with a single-colour background requires only about 4.9% keypoints (Figure 2) while a significantly complex Forest is adequately represented with more than 21% of the total number of pixels N (Figure 3).

Table 1: Results of the keypoint detection on the test images I (resolution, number of extracted key pixels, and the rate between the number of key pixels and the total number of pixels).

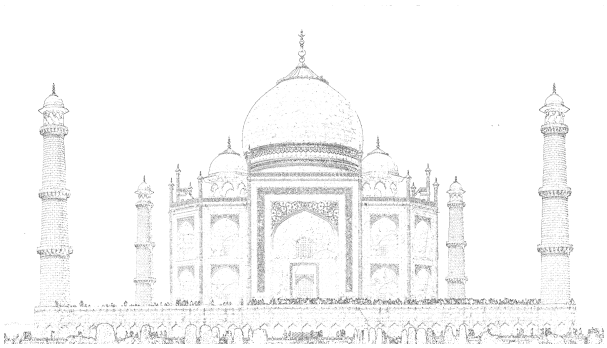
I	Resolution	$\lvert P^k \rvert$	r
Child	2,040 x 1,368	135,888	0.04869
Forest	2,040 x 1,188	511,050	0.21087
Taj Mahal	2,040 x 1,152	159,057	0.06768

(a)

(b)



(c)



(d)

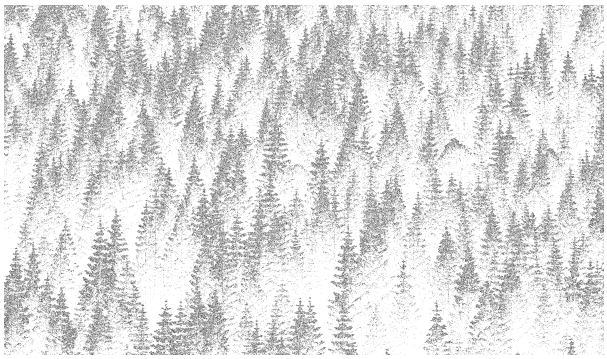
Figure 1: Keypoint detection with different parameters: (a) Original I, (b) P^k (no segmentation, manually determined r), (c) P^k (segmentation, manually determined r), (d) P^k (segmentation, automatic calculation of r for each S_i).

(a)

(b)

Figure 2: Child: (a) I, (b) P^k.

(a)



(b)

Figure 3: Forest: (a) I, (b) P^k.

4 Conclusion

In this paper, a new method for keypoint detection in greyscale images is introduced. The approach is based on a region image segmentation. The obtained segments are analysed using the information entropy metric, which determines the number of keypoints for each segment. Lastly, the neighbourhood gradients of pixels inside each segment are calculated, and the pixels with the highest gradients are proclaimed keypoints.

The proposed method could serve as an alternative approach to the established keypoint detectors. A specific area where the method could be used is the field of image compression, as keypoint detection extracts the most important image features.

Acknowledgement

This research was funded by the Slovenian Research and Innovation Agency under Research Project J2-4458 and Research Programme P2-0041, and the Czech Science Foundation under Research Project 23-04622L.

References

[1] J. Li and N. M. Allinson, "A comprehensive review of current local features for computer vision," *Neurocomputing*, vol. 71, no. 10, pp. 1771–1787, 2008.

[2] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 24, no. 4, pp. 509–522, 2002.

[3] M. Rudinac, B. Lenseigne, and P. Jonker, "Keypoint extraction and selection for object recognition," in *MVA 2009 IAPR Conference on Machine Vision Applications*, pp. 191–194, 2009.

[4] S. A. A. Shah, M. Bennamoun, and F. Boussaid, "Keypoints-based surface representation for 3D modeling and 3D object recognition," *Pattern Recognition*, vol. 64, pp. 29–38, 2017.

[5] C. Schmid and R. Mohr, "Local grayvalue invariants for image retrieval," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 19, no. 5, pp. 530–535, 1997.

[6] O. Moskvyak, F. Maire, F. Dayoub, and M. Baktashmotlagh, "Keypoint-aligned embeddings for image retrieval and re-identification," in *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 676–685, January 2021.

[7] 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.

[8] H.-G. Nguyen, R. Fablet, and J.-M. Boucher, "Spatial statistics of visual keypoints for texture recognition," in *Computer Vision – ECCV 2010*, pp. 764–777, 2010.

[9] H. P. Morevec, "Towards automatic visual obstacle avoidance," in *Proceedings of the 5th International Joint Conference on Artificial Intelligence*, vol. 2, p. 584, Morgan Kaufmann Publishers Inc., 1977.

[10] C. Harris and M. Stephens, "A combined corner and edge detector," in *Proceedings of the Alvey Vision Conference*, vol. 15, pp. 147–152, Citeseer, Alvety Vision Club, 1988.

[11] D. Lowe, "Object recognition from local scale-invariant features," in *Proceedings of the Seventh IEEE International Conference on Computer Vision*, vol. 2, pp. 1150–1157, 1999.

[12] P. Drews Jr, R. de Bem, and A. de Melo, "Analyzing and exploring feature detectors in images," in *2011 IEEE 9th International Conference on Industrial Informatics*, pp. 305–310, 2011.

[13] 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*, vol. 2, pp. 506–513, 2004.

[14] L. Demaret and A. Iske, "Advances in digital image compression by adaptive thinning," *Annals of the MCFA*, vol. 3, pp. 105–109, 2004.

[15] 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.

[16] E. Rosten and T. Drummond, "Machine learning for high-speed corner detection," in *Computer Vision – ECCV 2006*, pp. 430–443, Springer Berlin Heidelberg, 2006.

[17] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded up robust features," in *Computer Vision – ECCV 2006*, pp. 404–417, Springer Berlin Heidelberg, 2006.

[18] C. Wilson, P. Zicari, S. Craciun, P. Gauvin, E. Carlisle, A. George, and H. Lam, "A power-efficient real-time architecture for SURF feature extraction," in *2014 International Conference on ReConFigurable Computing and FPGAs*, pp. 1–8, 2014.

[19] H. Altwaijry, A. Veit, S. J. Belongie, and C. Tech, "Learning to detect and match keypoints with deep architectures," in *BMVC*, pp. 49.1–49.12, 2016.

[20] S. Wu, J. Xu, S. Zhu, and H. Guo, "A deep residual convolutional neural network for facial keypoint detection with missing labels," *Signal Processing*, vol. 144, pp. 384–391, 2018.

[21] A. Barroso-Laguna and K. Mikolajczyk, "Key.Net: Keypoint detection by handcrafted and learned CNN filters revisited," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 45, no. 1, pp. 698–711, 2023.

[22] N. Dhanachandra, K. Manglem, and Y. J. Chanu, "Image segmentation using K-means clustering algorithm and subtractive clustering algorithm," *Procedia Computer Science*, vol. 54, pp. 764–771, 2015.

[23] R. Manavalan and K. Thangavel, "TRUS image segmentation using morphological operators and DBSCAN clustering," in *2011 World Congress on Information and Communication Technologies*, pp. 898–903, 2011.

[24] E. Agustsson and R. Timofte, "NTIRE 2017 challenge on single image super-resolution: Dataset and study," in *The IEEE Conference on Computer Vision and Pattern Recognition Workshops*, July 2017.