Superpixel Segmentation: An Evaluation

David Stutz

Computer Vision Group, RWTH Aachen University
david.stutz@rwth-aachen.de

Abstract. In recent years, superpixel algorithms have become a standard tool in computer vision and many approaches have been proposed. However, different evaluation methodologies make direct comparison difficult. We address this shortcoming with a thorough and fair comparison of thirteen state-of-the-art superpixel algorithms. To include algorithms utilizing depth information we present results on both the Berkeley Segmentation Dataset [3] and the NYU Depth Dataset [19]. Based on qualitative and quantitative aspects, our work allows to guide algorithm selection by identifying important quality characteristics.

1 Introduction

The term superpixel was introduced by Ren and Malik in 2003 [16] and is used to describe a group of pixels similar in color or other low-level properties. The concept of superpixels is motivated by two important aspects [16]: firstly, pixels do not represent natural entities but are merely a result of discretization; and secondly, the high number of pixels in large images prevents many algorithms from being computationally feasible.

Superpixels have actively been used for a wide range of applications such as classical segmentation [16, 17], semantic segmentation [6], stereo matching [30] or tracking [26] and numerous superpixel algorithms have been proposed. However, keeping an overview of the different approaches and their suitability for specific applications is difficult. This is caused by varying experimental setups and metrics used for evaluation [12]. Furthermore, only few publications are devoted to a thorough comparison of existing algorithms.

In this paper, we address this shortcoming and present an extensive comparison of thirteen state-of-the-art superpixel algorithms. In Section 2 we discuss important related work regarding the comparison of superpixel algorithms and subsequently we survey existing superpixel algorithms. In Section 3 we discuss relevant datasets and introduce our benchmark in Section 4. Finally, in Section 5, we present a qualitative and quantitative comparison of the superpixel algorithms, before concluding in Section 6.

Recommended for submission to YRF2015 by Prof. Dr. Bastian Leibe.

Table 1. Overview of all evaluated superpixel algorithms ordered by year of publication. In Row 3, we categorize the algorithms in either graph-based approaches (gb) or gradient ascent approaches (ga) [2]. Furthermore, in Row 4, we note the programming language of the evaluated implementations as it may influence the runtime reported in Section 5.2 (M refers to MatLab). We distinguish algorithms offering direct control over the number of superpixels (Row 5), algorithms providing a compactness parameter (Row 6) and algorithms using depth information (Row 7).

	NC	*FH*	*QS*	*TP*	*SLIC*	*CIS*	*ERS*	*PB*	*CRS*	*SEEDS*	*TPS*	*DASP*	*VCCS*
Ref.	[16]	[5]	[24]	[7]	[1]	[25]	[8]	[29]	[10]	[23]	[22]	[27]	[13]
Year	2003	2004	2008	2009	2010	2010	2011	2011	2011	2012	2012	2012	2013
Cat.	gb	gb	ga	ga	ga	gb	gb	gb	ga	ga	gb	ga	ga
Impl.	C/M	C++	C/M	C/M	C++	C++	C++	C++	C++	C++	C/M	C++	C++
Sup.	✓	✗	✗	✓	✓	✓	✓	✓	✓	✓	✓	✓	✗
Comp.	✗	✗	✗	✗	✓	✗	✗	✗	✓	✗	✗	✓	✓
Depth	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✗	✓	✓

2 Related Work

There are only few publications devoted to the comparison of existing superpixel algorithms in a consistent framework: to the best of our knowledge these are [18], [2] and [12]. However, these publications cannot include several recent algorithms (for example [23, 13, 27]). Meanwhile, authors tend to include a brief evaluation intended to show superiority of their proposed superpixel algorithm over selected existing approaches (for example [8, 25, 23, 23, 27, 13]). However, these results are not comparable across publications.

2.1 Superpixel Algorithms

Table 1 gives an overview of all evaluated superpixel algorithms. We categorize the algorithms according to criteria we find important for evaluation and algorithm selection. Roughly, the algorithms can be categorized as either graph-based approaches or gradient ascent approaches [2]. Furthermore, we distinguish algorithms offering direct control over the number of superpixels as well as algorithms providing a compactness parameter. Overall, we evaluated thirteen state-of-the-art superpixel algorithms including three algorithms utilizing depth information. We also note that there are additional superpixel algorithms [17, 11, 4, 28, 14, 20] for which evaluation was not possible due to the lack of open source code.

3 Datasets

As popular dataset for image segmentation and contour detection, the Berkeley Segmentation Dataset [3], referred to as BSDS500, consists of 500 natural images of size 482×321 (200 training images, 100 validation images, 200 test images). The provided ground truth segmentations, at least five per image, have

been obtained from different persons and reflect the difficult nature of image segmentation.

In contrast to the natural images of the BSDS500, The NYU Depth Dataset [19], referred to as NYUV2, comprises 1449 images of different indoor scenes (we chose 200 validation images and 400 test images including most of the scenes). For all images, pre-processed depth images are provided. As these images have been undistorted and aligned with the color images, we crop the original images of size 640×480 to 608×448 pixels. In addition, following Ren and Bo [15], we remove small unlabeled regions and combine class and instance labels to guarantee connected ground truth segments. Overall, difficult lighting conditions and cluttered scenes contribute to the difficulty of the NYUV2.

4 Benchmark

We use an extended version of the Berkeley Segmentation Benchmark, introduced by Arbeláez et al. [3], to evaluate superpixel algorithms. Among other metrics, the benchmark includes Boundary Recall (Rec) and Undersegmentation Error (UE) [7, 12] as primary metrics to assess superpixel algorithms.

Boundary Recall is part of the Precision-Recall Framework [9] originally used to evaluate contour detectors. Treating region boundaries of a superpixel segmentation as contours, Boundary Recall represents the fraction of boundary pixels correctly detected by the superpixel algorithm. As superpixels are expected to adhere to boundaries, high Boundary Recall is desirable.

Undersegmentation Error, as originally proposed by Levinshtein et al. [7], measures the "bleeding" of superpixels with respect to a ground truth segmentation. We implemented the corrected formulation proposed by Neubert and Protzel [12] computing an error in the range of $[0, 1]$. Low Undersegmentation Error is preferable as each superpixel is expected to cover at most one ground truth segment.

5 Evaluation and Comparison

In addition to the superpixel algorithms introduced in Section 2.1, we include a 2D and 3D re-implementation of *SEEDS*, called *reSEEDS* and *reSEEDS3D*, respectively. Further details can be found in [21].

Our comparison is split into a qualitative part, examining the visual quality of the generated superpixels, and a quantitative part based on Boundary Recall, Undersegmentation Error and runtime. To ensure a fair comparison, the parameters have been chosen to jointly optimize Boundary Recall and Undersegmentation Error using discrete grid search. Parameter optimization was performed on the validation sets while comparison is performed on the test sets.

5.1 Qualitative

The visual appearance of superpixels is determined by compactness and regularity – properties that may also have strong influence on possible applications.

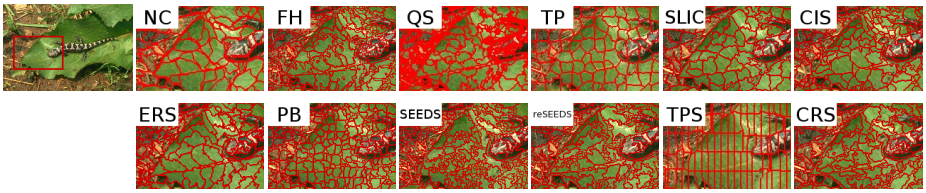

Fig. 1. Superpixel segmentations obtained for an example image from the BSDS500. From left to right, top to bottom: original image, *NC*, *FH*, *QS*, *TP*, *SLIC*, *CIS*, *ERS*, *PB*, *SEEDS*, *reSEEDS*, *TPS* and *CRS*. Approximately 600 superpixels have been generated for the whole image.

Figures 1 and 2 present results on the BSDS500 and NYUV2, respectively, obtained after parameter optimization. We note that these images are intended to be as representative as possible, however, generated superpixel segmentations may vary across different images.

Both *FH* and *QS* produce highly irregular superpixels. However, they are able to capture small details and all important boundaries. Furthermore, note that *QS* produces more superpixels in highly textured areas. In contrast, *TP* generates highly compact superpixels at the expense of missing several boundaries. *CRS* and *ERS* produce irregular superpixels with good boundary adherence and are more visually appealing than superpixels generated by *FH* or *QS*. While *PB* shows reasonable results on the BSDS500, producing irregular and small superpixels, results on the NYUV2 appear to be unfinished and of poor quality. Similarly, *TPS* produces unfinished superpixel segmentations on both datasets. Both *SLIC* and *CRS* provide a compactness parameter which has been traded off for boundary adherence during parameter optimization. Thus, the generated superpixels are irregular and not compact, especially on the NYUV2. The original implementation of *SEEDS* produces highly irregular superpixels capturing the majority of boundaries. Our implementation, *reSEEDS*, shows similar behavior.

Intuitively, depth information allows to adapt superpixels to the underlying three-dimensional structure. While *DASP* is able to resemble this structure at

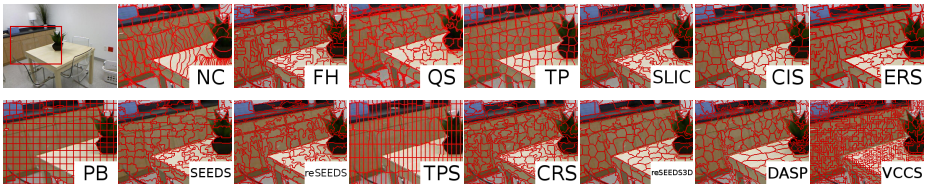

Fig. 2. Superpixel segmentations obtained for an example image from the NYUV2. From left to right, top to bottom: original image, *NC*, *FH*, *QS*, *TP*, *SLIC*, *CIS*, *ERS*, *PB*, *SEEDS*, *reSEEDS*, *TPS*, *CRS*, *reSEEDS3D*, *DASP*, *VCCS*. Note that *reSEEDS3D*, *DASP* and *VCCS* use depth information for superpixel segmentation. Approximately 840 superpixels have been computed for the whole image.

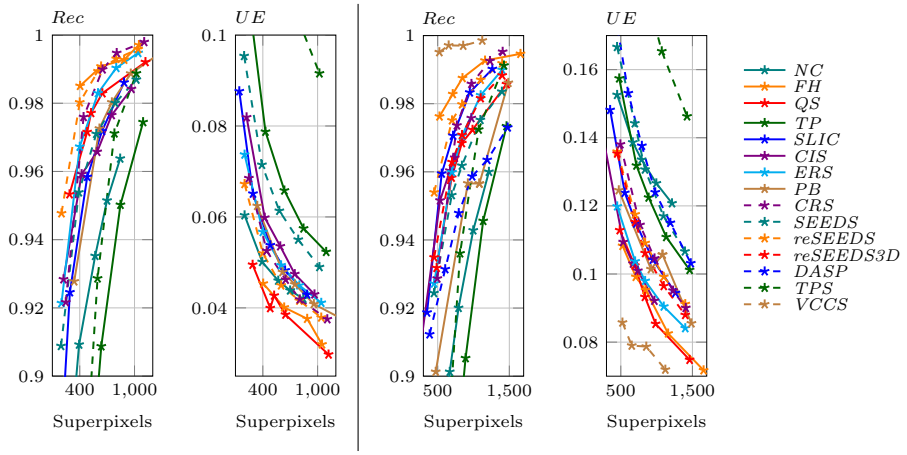

Fig. 3. Comparison of thirteen superpixel algorithms with respect to Boundary Recall (*Rec*) and Undersegmentation Error (*UE*) on the test sets of the BSDS500 (left) and the NYUV2 (right). Note that for visualization purposes only a small part of the range of both metrics is shown.

least in parts, *VCCS* produces highly irregular superpixels. This may be due to the fact that *DASP* adapts the number of superpixels according to depth, whereas *VCCS* directly operates within the point cloud and merely the back projection is shown. As result, the generated superpixels occur irregular and voxelization (see [13]) is still visible. Compared to *DASP*, *reSEEDS3D* generates slightly less irregular superpixels with better boundary adherence.

5.2 Quantitative

The quantitative comparison is based on Boundary Recall and Undersegmentation Error, averaged over the test sets, as a function of the number of superpixels. As shown in Figure 3, *FH* performs excellent on both datasets and is only outperformed by *VCCS*. However, as *VCCS* operates within point clouds, these results are hardly comparable. In addition, *FH* is closely followed by approaches such as *QS*, *SLIC*, *CRS* and *ERS*. Our implementation of *SEEDS*, *reSEEDS*, is able to keep up with *FH*, while consistently outperforming the original implementation. Unfortunately, as shown by *reSEEDS3D*, depth information may not necessarily improve performance. This is supported by the poor performance of *DASP*. In addition, our implementation of *SEEDS* demonstrates that any ranking extracted from Figure 3 is possibly challenged by revising existing implementations. This also justifies a qualitative comparison as performed in Section 5.1.

Another important aspect of superpixel algorithms is runtime (measured using a 64bit machine with Intel Core i7-3770@3.4GHz, 16GB RAM and without GPU acceleration or multi-threading). Iterative algorithms such as *SLIC* and *SEEDS* may be optimized with respect to runtime by decreasing the number of iterations. Figure 4 compares these optimized versions to algorithms requiring

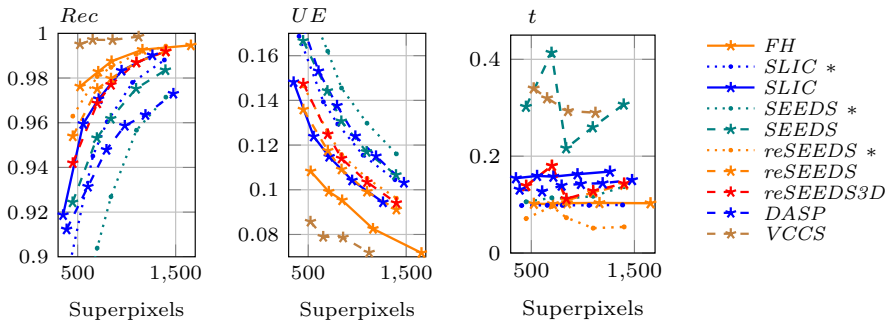

Fig. 4. Comparison of the runtime t in seconds of superpixel algorithms requiring less than $0.5s$ per image on the NYUV2. *SLIC*, *SEEDS* and *reSEEDS* may be optimized with respect to runtime by decreasing the number of iterations. These optimized versions are indicated by an $*$.

less than $0.5s$ on the NYUV2 (images of size 608×448). The optimized versions of *SLIC*, *SEEDS* and *reSEEDS*, indicated by an $*$, show significantly reduced runtime while providing similar performance in terms of Boundary Recall and Undersegmentation Error. The drop in performance is lowest for *reSEEDS* – Boundary Recall even increases – which simultaneously shows the lowest runtime with $\sim 0.05s$ per image. We also observe low runtimes for *FH* and *DASP*.

6 Conclusion

Several algorithms provide both excellent performance and low runtime. Furthermore, including additional information such as depth may not necessarily improve performance. Therefore, additional criteria are necessary to asses superpixel algorithms. In particular, we find that visual quality, runtime and the provided parameters are among these criteria. Clearly, visual appearance is difficult to measure appropriately, however, it may have serious impact on possible applications. Furthermore, low runtime is desirable when using superpixel algorithms as pre-processing step, especially in real-time settings. Finally, parameters should be interpretable and easy to tune and algorithms providing a compactness parameter are preferable. In addition, as the number of superpixels can be understood as a lower bound on performance, we prefer algorithms offering direct control over the number of superpixels.

In conclusion, while many algorithms provide excellent performance with respect to Undersegmentation Error [7, 12] and Boundary Recall [9], they lack control over the number of superpixels or a compactness parameter. Furthermore, these impressive results with respect to Boundary Recall and Undersegmentation Error do not necessarily reflect the perceived visual quality of the generated superpixel segmentations.

References

1. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: SLIC superpixels. Tech. rep., École Polytechnique Fédérale de Lausanne, Lusanne, Switzerland (June 2010)
2. Achanta, R., Shaji, A., Smith, K., Lucchi, A., Fua, P., Süsstrunk, S.: SLIC superpixels compared to state-of-the-art superpixel methods. IEEE Trans. Pattern Anal. Mach. Intell. 34(11), 2274–2281 (November 2012)
3. Arbeláez, P., Maire, M., Fowlkes, C., Malik, J.: Contour detection and hierarchical image segmentation. IEEE Trans. Pattern Anal. Mach. Intell. 33(5), 898–916 (May 2011)
4. Drucker, F., MacCormick, J.: Fast superpixels for video analysis. In: WMVC. pp. 1–8. IEEE Computer Society, Snowbird, Utah (December 2009)
5. Felzenswalb, P.F., Huttenlocher, D.P.: Efficient graph-based image segmentation. Int. J. Comput. Vision 59(2), 167–181 (September 2004)
6. Gupta, S., Arbeláez, P., Malik, J.: Perceptual organization and recognition of indoor scenes from RGB-D images. In: CVPR. pp. 564–571. IEEE Computer Society, Portland, Oregon (June 2013)
7. Levinshtein, A., Stere, A., Kutulakos, K.N., Fleet, D.J., Dickinson, S.J., Siddiqi, K.: TurboPixels: Fast superpixels using geometric flows. IEEE Trans. Pattern Anal. Mach. Intell. 31(12), 2290–2297 (December 2009)
8. Lui, M.Y., Tuzel, O., Ramalingam, S., Chellappa, R.: Entropy rate superpixel segmentation. In: CVPR. pp. 2097–2104. IEEE Computer Society, Providence, RI (June 2011)
9. Martin, D., Fowlkes, C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE Trans. Pattern Anal. Mach. Intell. 26(5), 530–549 (May 2004)
10. Mester, R., Conrad, C., Guevara, A.: Multichannel segmentation using contour relaxation: Fast super-pixels and temporal propagation. In: Heyden, A., Kahl, F. (eds.) SCIA, LNCS, vol. 6688, pp. 250–261. Springer, Berlin, Heidelberg, Germany (May 2011)
11. Moore, A.P., Prince, S.J.D., Warrell, J., Mohammed, U., Jones, G.: Superpixel lattices. In: CVPR. pp. 1–8. IEEE Computer Society, Anchorage, AK (June 2008)
12. Neubert, P., Protzel, P.: Superpixel benchmark and comparison. In: Forum Bildverarbeitung. Regensburg, Germany (November 2012)
13. Papon, J., Abramov, A., Schoeler, M., Wörgötter, F.: Voxel cloud connectivity segmentation - supervoxels for point clouds. In: CVPR. pp. 2027–2034. IEEE Computer Society, Portland, Oregon (June 2013)
14. Perbet, F., Maki, A.: Homogeneous superpixels from random walks. In: (IAPR) MVA. pp. 26–30. Nara, Japan (June 2011)
15. Ren, X., Bo, L.: Discriminatively trained sparse code gradients for contour detection. In: Bartlett, P.L., Pereira, F.C.N., Burges, C.J.C., Bottou, L., Weinberger, K.Q. (eds.) NIPS, vol. 25, pp. 584–592. Curran Associates, Red Hook, New York (December 2012)
16. Ren, X., Malik, J.: Learning a classification model for segmentation. In: ICCV. pp. 10–17. IEEE Computer Society, Nice, France (October 2003)
17. Rohkohl, C., Engel, K.: Efficient image segmentation using pairwise pixel similarities. In: A.Hamprecht, F., Schnrr, C., Jhne, B. (eds.) Pattern Recognition, LNCS, vol. 4713, pp. 254–263. Springer, Berlin, Heidelberg, Germany (September 2007)

18. Schick, A., Fischer, M., Stiefelhagen, R.: Measuring and evaluating the compactness of superpixels. In: ICPR. pp. 930–934. IEEE Computer Society, Tsukuba, Japan (November 2012)
19. Silberman, N., Hoiem, D., Kohli, P., Fergus, R.: Indoor segmentation and support inference from RGBD images. In: Fitzgibbon, A., Lazebnik, S., Perona, P., Sato, Y., C. Schmid, C. (eds.) ECCV. LNCS, vol. 7576, pp. 746–760. Springer, Berlin, Heidelberg, Germany (October 2012)
20. Siva, P., Wong, A.: Grid seams: A fast superpixel algorithm for real-time applications. In: CRV. pp. 127–134. IEEE Computer Society, Montreal, Canada (May 2014)
21. Stutz, D.: Superpixel segmentation using depth information. B.Sc. thesis, Computer Vision Group, RWTH Aachen University, Aachen, Germany (September 2014). http://davidstutz.de/projects/superpixelsseeds/
22. Tang, D., Fu, H., Cao, X.: Topology preserved regular superpixel. In: ICME. pp. 765–768. IEEE Computer Society, Melbourne, Australia (July 2012)
23. Van den Bergh, M., Boix, X., Roig, G., de Capitani, B., van Gool, L.: SEEDS: Superpixels extracted via energy-driven sampling. In: Fitzgibbon, A.W., Lazebnik, S., Perona, P., Sato, Y., Schmid, C. (eds.) ECCV, LNCS, vol. 7578, pp. 13–26. Springer, Berlin, Heidelberg, Germany (2012)
24. Vedaldi, A., Soatto, S.: Quick shift and kernel methods for mode seeking. In: Forsyth, D.A., Torr, P.H.S., Zisserman, A. (eds.) ECCV, LNCS, vol. 5305, pp. 705–718. Springer, Berlin, Heidelberg, Germany (October 2008)
25. Veksler, O., Boykov, Y., Mehrani, P.: Superpixels and supervoxels in an energy optimization framework. In: Daniilidis, K., Maragos, P., Paragios, N. (eds.) ECCV, LNCS, vol. 6315, pp. 211–224. Springer, Berlin, Heidelberg, Germany (September 2010)
26. Wang, S., Lu, H., Yang, F., Yang, M.H.: Superpixel tracking. In: ICCV. pp. 1323–1330. IEEE Computer Society, Barcelona, Spain (November 2011)
27. Weikersdorfer, D., Gossow, D., Beetz, M.: Depth-adaptive superpixels. In: ICPR. pp. 2087–2090. IEEE Computer Society, Tsukuba, Japan (November 2012)
28. Zeng, G., Wang, P., Wang, J., Gan, R., Zha, H.: Structure-sensitive superpixels via geodesic distance. In: ICCV. pp. 447–454. IEEE Computer Society, Barcelona, Spain (November 2011)
29. Zhang, Y., Hartley, R., Mashford, J., Burn, S.: Superpixels via pseudo-boolean optimization. In: ICCV. pp. 1387–1394. IEEE Computer Society, Barcelona, Spain (November 2011)
30. Zhang, Y., Hartley, R., Mashford, J., Burn, S.: Superpixels, occlusion and stereo. In: DICTA. pp. 84–91. IEEE Computer Society, Noosa, Australia (December 2011)