Superpixel Segmentation Based Gradient Maps on RGB-D Dataset

Lixing Jiang, Huimin Lu, Vo Duc My, Artur Koch and Andreas Zell

***Abstract*— Superpixels aim to group homogenous pixels by a series of characteristics in an image. They decimate redundancy that may be utilized later by more computationally expensive algorithms. The most popular algorithms obtain superpixels based on an energy function on a graph. However, these graph-based methods have a high computational time consumption. This study presents a fast and high quality over-segmentation method by a watershed transform based on computing the dissimilarity of pixels among RGB(D) cues and gradient maps. Specifically, we first capture a gradient map based on an image to enhance and explain directional variations in the image scene. A distance function then measures the similarity among adjacent pixels, which is calculated according to RGB(D) values. A fast marker-controlled watershed (MCW) algorithm traverses the entire image based on the distance function. Finally, we acquire all watersheds consisting of superpixel contours. Experimental results compare state-of-the-art algorithms and highlight the effectiveness of the proposed method. As an application, the proposed superpixel algorithm can be used in applications aiming for real-time, like mobile robot saliency detection and segmentation.**

I. Introduction

Recently, an increasing amount of vision and navigation research involving robots has been based on superpixels [1], [2]. As a middle-level image feature, superpixels comprise homogeneous color, texture and spatial regions in an image. Superpixels belong to over-segmentation methods, which are shown to be better than a rectangular patch box as they can be aligned effectively. Two fundamental reasons allow superpixels to be adapted widely to vision applications. On the one hand, superpixels present more natural entities in a scene including efficient features than pixels resulting in discretization. On the other hand, computation can be reduced by the size of superpixels.

Since superpixels are able to represent natural entities much better than normal pixels, they can be widely and effectively used in pre-processing for segmentation and for detection in computer and robot vision. To be useful, this over-segmentation method should produce effective superpixels with high compactness and a low computational overhead. Current methods capture superpixels based on the visual features of an RGB image [3]–[9]. However, there are only a few superpixel algorithms that can handle superpixel segmentation in real-time for practical applications. Therefore, we have focused on finding a real-time and robust superpixel method useful for computer vision and robotics tasks.

L. Jiang, D. M. Vo, A. Koch and Dr. H. Lu are with the Chair of Cognitive Systems, headed by Prof. A. Zell, Computer Science Department, University of Tuebingen, Sand 1, D-72076 Tuebingen, Germany {lixing.jiang, duc-my.vo, artur.koch, andreas.zell}@uni-tuebingen.de {lhmnew}@nudt.edu.cn

(a) Raw RGB image (b) Magnitude of RGB-D gradient image (c) Superpixels

Fig. 1. Superpixel segmentation examples.

In addition, with the broad deployment of Microsoft Kinect RGB-D sensors for vision applications, the requirement of RGB-D-based algorithms has become more universal [9], [10]. By using supplemental depth cues and derived features, superpixel segmentation can be more precise and thus more feasible for practical applications [9]. Depth values can be assigned even to separate objects with a homogeneous visual appearance. This advantage inspires us to combine visual features with depth values in order to group superpixels together. A simple practical application would be the pre-processing stage for classical segmentation and salience detection for a mobile service robot in a real-world indoor scenario.

The main focus of this work is the development of an efficient superpixel segmentation method to adjust the requirements of preprocessing for object detection for indoor mobile robots. Fig.1(a) shows three scenes obtained by an RGB-D sensor from [11]. Multiple objects with complex context are shown in these scenes. Instead of analyzing every pixel, we expect that the robot can quickly process compact and more natural entities in an image. The concept of superpixels, as shown in Fig.1(c), gives a basic theory to solve these problems.

For this purpose, we use marker-controlled watershed (MCW) transform based on a gradient map to produce superpixel boundaries. In addition to analyzing and using color information, we also can incorporate depth values as measurements. Superpixels sharply decrease the computational complexity for the later stages of the processing pipeline. Specifically, we first introduce an RGB-D gradient map combining color with depth values to capture directional variations in RGB and depth image. The new RGB-D

gradient image provides complementary cues for boundaries, even with a homogeneous visual distribution. As shown in Fig. 2(d), although the pendant lamp in the image has a similar color to the background, the depth data still reliably gives distinct depth borders for preprocessing. A distance function for evaluating the dissimilarity of pixels is then proposed based on the color channels or together with depth cue. Because the used sensor data is composed of both color and depth values, it retains the uniqueness and consistency of different pixels. Consequently, a fast MCW transform is used to traverse every pixel in the image according to the distance function. Ultimately, we evaluate the proposed method against different state-of-the-art alternatives using RGB-D datasets. The results show that watershed superpixel segmentation based on gradient image furnishes fast and robust superpixel results. It is adapted to be able to cluster approximate color and depth information.

The advantages of our new superpixel method are:

- The method employs a fast MCW algorithm on color and gradient map. Due to its low computational costs, the proposed approach is well suited for real-time applications.
- The approach of gradient based watershed in images is applied to merge complementary color and depth information to produce a gradient map. It achieves a more general means for distinguishing objects in complex environments.

The remainder of the paper is structured as follows. In Sect. II, we give an overview of related work. Sect. III describes our proposed method, including the technique of creating an RGB-D gradient map and a fast MCW transform. In Sect. IV we then present the experimental results based on an RGB-D dataset and show some visual superpixel results by comparison of different superpixel approaches. Finally, we draw conclusions in Sect. V.

II. Related Work

Research on superpixel algorithms has drawn much attention during the last few years. The term superpixel was first advocated by Ren and Malik in [12] and is helpful for grouping pixels that feature similarities in color or low-level features. Superpixel algorithms can be roughly classified into two categories: graph-based and gradient-ascent-based [5]. Graph-based approaches are used to create an undirected, weighted graph. Typically, an energy function relying on a graph is proposed and optimized using graph cuts or similar techniques. Most algorithms enforce color homogeneity by minimizing objective functions. In contrast, regardless of being built around an energy function, gradient ascent methods are not dependent on a graph structure. Instead, they use a variety of different approaches to optimize the proposed energy in order to generate a superpixel segmentation.

Felzenswalb and Huttenlocher (FH) [13] present a graph-based approach to segment an image into superpixels. An image is first represented by a graph with vertices as pixels and weighted edges. It is a greedy algorithm that can construct a *minimum spanning tree* in the graph and merges minimum weighted edges between two neighboring points. The constant intensity superpixels (*CIS*) [14] algorithm generates superpixels by partitioning the problem in an energy minimization function and optimizing it with graph cuts. In the end, an expansion algorithm [15] is used to optimize every pixel. Comparable to CIS, superpixels (via pseudo-boolean optimization (*PB*) [8]) obtain higher computational efficiency by simplifying two pseudo-boolean functions. Furthermore, another graph-based approach for images, name the entropy rate superpixels (*ERS*) [7], generates superpixels utilizing an objective entropy rate function. The entropy rate function can be computed based on the elements of a color term and a boundary term. Conrad *et al.* [6] describe contour relaxed superpixels (*CRS*) based on maximum accordance of the contours. The maximum accordance of the contours is calculated by the image content and a *Gibbs-Markov random field* model.

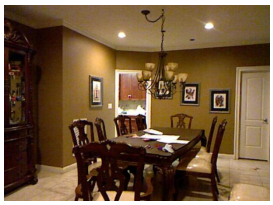
In contrast to graph-based methods, a classical gradient ascent based algorithm named *simple linear iterative clustering* (*SLIC*) [5] adapts a local k-means clustering algorithm to group superpixels in five-dimensional space defined by *L, a, b* in the CIELAB color space. Subsequently, Bergh *et al.* [4] introduce superpixels extracted via the energy-driven sampling (*SEEDS*) algorithm, which clusters superpixels based on a hill-climbing optimization framework. Inspired by SLIC and the watershed algorithm [16], Neubert *et al.* [3] achieve faster *preemptiveSLIC* and compact watershed (*CW*) for superpixels. Recently, Weikersdorfer *et al.* introduced the depth-adaptive superpixels algorithm (DASP) [9]. This method calculates the density of superpixel clusters from a depth map and clusters pixels using k-means. Nevertheless, this method is time-consuming and therefore not suited for real-time applications. In a review of superpixel approaches [17], Stutz *et al.* summarized and compared several superpixel segmentation methods from 2003 to 2013.

Our over-segmenting method belongs to the category of gradient ascent based approaches with a focus on improved runtime and quality based on a gradient map by the MCW transform.

III. Gradient Map Based Superpixel Segmentation

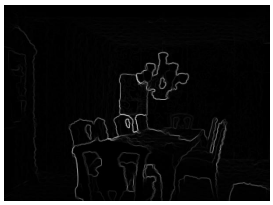
In this section, we describe the details of our approach. Our superpixel segmentation method incorporates color and depth values using a gradient map on a MCW transform. Hence, the herein proposed improvements regarding the watershed transform may be divided into two parts:

- Firstly, we calculate a gradient map from RGB or RGB-D images, which should give distinct contours. The depth information enhances curves even if the color distributions are homogeneous.
- Secondly, we optimize the termination condition of the watershed transform by a gradient map to accelerate the algorithm.



(a) Color image

(b) Magnitude map of RGB



(c) Magnitude of depth cues

(d) Magnitude map of RGB-D

Fig. 2. Gradient images.

A. Gradient Map

A gradient map characterizes the directional variations in an intensity image. It captures the variation and orientation from a disorganized image matrix. In our case, we first convert the color image into a gray image and extract the gradient map from gray and depth cues by a *Sobel* filter.

Based on gradient images, boundary pixels correspond to large gradient values, the direction of a boundary pixel is perpendicular to the gradient direction. Fig. 2(b) and Fig. 2(c) show gradient magnitude maps from RGB and depth image respectively. Additionally, we can obtain an enhanced gradient map by combining gradient maps from the color image and the depth image, as shown in Fig. 2(d). As mentioned above, the gradient map therefore captures boundary features, that may be used to initialize markers in order to accelerate the watershed transform.

B. Marker-controlled Watershed Transform

In the segmentation stage, we use the MCW transform, which is a fast and efficient technique proposed by Beucher and Meyer [16]. Neubert *et al.* [3] integrate a controllable compactness constraint to improve the compactness and the shape of watershed superpixels. Our approach is an extension of Neubert's method, incorporating the gradient map with a watershed transform.

In our method, we first choose local gradient minima as markers by a gradient map. The markers are essentially seed points which are able to initialize the segmentation algorithm. Every initial marker has a corresponding watershed region, thus the size of markers is the same as the resultant number of superpixel regions. The distribution of markers exerts an influence on the segmentation result. The markers then start to spread repeatedly pixel by pixel until they catch a border around another marker.

We utilize a simple distance function $Dist$ measuring the dissimilarity of two adjacent pixels (p, q) in an image. We approximate the dissimilarity by the maximum absolute distance in the three color channels (RGB) as:

$$Dist_{RGB}(p,q) = \max\{|p_R - q_R|, |p_G - q_G|, |p_B - q_B|\} \quad (1)$$

Similarly, an distance function in an depth image can be calculated as:

$$Dist_D(p,q) = |p_D - q_D| \quad (2)$$

As a consequence, the measurement of RGB-D is defined as follows:

$$Dist_{RGBD}(p,q) = \alpha \cdot Dist_{RGB}(p,q) + (1-\alpha) \cdot Dist_D(p,q) \quad (3)$$

D is a scaled depth map from a raw depth image. We set the mixture parameter α to 0.5, effectively leading to a uniform influence of RGB and depth values during MCW segmentation.

In addition, a 4-neighborhood distance function ($Dist_{4n}$) of pixel p with its four adjacent points in an image can be defined as:

$$Dist_{4n}(p) = \min_{n=l,t,r,b} \{Dist_{RGB(D)}(p, p_n)\} \quad (4)$$

p hereby represents the current pixel, n represent one of the four adjacent neighbors, with p_l, p_t, p_r and p_b being adjacent pixels from left, top, right and bottom of p, respectively. The gradient based superpixels method (gWatersheds) is summarized in Algorithm 1.

IV. Experimental Results

For our experiments, we benchmarked on the publicly available NYU RGB-D dataset published by Silberman *et al.* [11]. The dataset includes 869 RGB-D images for training, acquired from diverse indoor scenarios with low-contrast objects and complex background. It also includes ground truth data. Due to depth image preprocessing, the effective resolution of the images in the dataset is cropped to $[608 \times 448]$. Therefore, the dataset comprises same sensor data conditions as we expect in the case of our mobile MetraLabs Scitos G5 indoor service robot equipped with a Microsoft Kinect [18], [19].

We compare our method to seven state-of-the-art superpixel algorithms, namely: constant intensity superpixels (CIS) [14], superpixels via pseudo-boolean optimization (PB) [8], entropy rate superpixels (ERS) [7], contour relaxed superpixels (CRS) [6], simple linear iterative clustering (SLIC) [5], superpixels extracted via energy-driven sampling (SEEDS) [4], [17], preemptiveSLIC and compact watershed (CW) [3], respectively. For evaluation of segmentation quality, we focus on three evaluation metrics: runtime, boundary recall and undersegmentation error.

A. Runtime Analysis

Runtime is a vital performance indicator in real-world applications for robots. Real-time superpixel algorithms are fundamental for subsequent image processing. Hence, we first give the average runtime comparison in Table I. All experiments were performed on an Intel Core i5-4590 CPU at 3.30 GHz and 4 GB RAM.

The results show that our method may run on common hardware at about 60 Hz (15 ms per frame), and therefore

TABLE I

RUNTIMES OF SEVERAL STATE-OF-THE-ART ALGORITHMS FOR DIFFERENT SEGMENT NUMBERS *N* ON THE NYU DATASET.

Time in ms	$N = 25$	$N = 50$	$N = 100$	$N = 200$	$N = 400$	$N = 800$	$N = 1600$	$N = 3200$	Average time
CIS [14]	7934.86	9305.06	9885.98	10086.10	10197.10	10086.60	9880.60	9621.69	9624.75
CRS [6]	235.52	279.92	309.22	368.06	456.39	575.59	726.78	926.19	484.71
PB [8]	42.35	42.46	42.49	42.54	42.60	42.61	42.64	42.68	42.55
SEEDS [4], [17]	138.12	146.05	136.46	113.31	156.64	147.44	187.59	172.35	149.74
SLIC [5]	129.90	134.21	135.54	139.00	140.77	141.79	142.18	142.49	138.23
preemptiveSLIC [3]	34.22	32.65	32.67	32.95	33.60	34.28	34.85	35.20	33.80
CW [3]	15.00	15.10	15.17	15.23	15.24	15.25	15.25	15.27	15.19
gWatershed (ours)	15.01	15.27	15.42	15.53	15.62	15.72	15.92	15.96	15.56
gRGBDWatershed (ours)	17.78	17.82	17.83	17.87	17.92	17.94	17.91	17.82	17.87

Algorithm 1 gWatersheds

Input: RGB or RGB-D images I
Output: Superpixels map S

1: Initialize array of queues $Q[256]$: each queue $Q[h]$ serves as a *FIFO* queue over the heightmap values $h \in [0, 256)$;
2: Generate an initial watersheds area: initialize a zero matrix S, set all the border values of S to $S_{border} = -1$ and sample $K > 0$ seeds over S using gradient values;
3: // Initially process S and fill queues
4: **for** each pixel $p(i,j) \in I$ and $s(i,j) \in S$ **do**
5: **if** $s(i,j) == 0$ **then** compute gradient $G(i,j)$;
6: **if** $G(i,j)$ is border **then** $s(i,j) = S_{border}$;
7: **end if**
8: **end if**
9: **if** $s(i,j) == 0 \wedge (s(adj(i,j)) > 0)$ **then**
10: Calculate queue index $h_p = Dist_{4n}(p)$;
11: Push pixel (i,j) into $Q[h_p]$;
12: $s(i,j) = S_{inQ} = -2$;
13: **end if**
14: **end for**
15: // Process elements in queues until empty
16: **for** h from 0 to 255 **do**
17: **while** $Q[h]$ not empty **do**
18: Pop pixel index (x,y) from $Q[h]$;
19: **for** $\forall a_i \in adj(x,y)$ **do**
20: **if** $s(a_i) == 0$ **then**
21: $h_{a_i} = Dist_{RGB(D)}(p(x,y), p(a_i))$;
22: Push pixel a_i into $Q[h_{a_i}]$;
23: $s(a_i) = S_{inQ} = -2$;
24: **else**
25: **for** $\forall a_j \in adj(x,y) \wedge i < j$ **do**
26: **if** $s(a_i) > 0 \wedge s(a_j) > 0$
27: $\wedge (s(a_i) \neq s(a_j))$ **then**
28: $s(x,y) = S_{border}$;
29: **end if**
30: **end for** // adjacent a_j
31: **end if**
32: **end for** // adjacent a_i
33: **end while** // $Q[h]$
34: **end for** // h
35: **if** Queues not empty **then** Repeat from 16;
36: **end if**
37: **return** S;

gives us the desired real time performance. This is mainly due to the watershed method, our focus on simple distance metrics and the use of a gradient map to initialize the markers to speedup the segmentation process. It is also the fastest method, although only a minor improvement is achieved in comparison with CW. CIS and CRS belong to more sophisticated graph-based superpixel algorithms and feature higher runtimes (> 480 ms), as compared to other methods. This is reasonable due to the computational complexity of those methods.

Table I also summarizes the dependency of the tested methods on the number of superpixels. Besides CRS, none of the evaluated methods features major run time impact with a higher number of segments.

Overall, PB, preemptive SLIC, CW and the proposed method may be considered as suitable for real time applications. Nevertheless, taking into account superpixels as a preprocessing step in a multi-stage application pipeline, even small gains in performance may be considered crucial for the real time suitability of the whole system.

B. Boundary Recall

Boundary recall is defined as the part of ground truth edges, which is within a certain distance *d* from a superpixel boundary. In order to evaluate the length of the boundaries in the image, we use both boundary precision as well as recall. Assuming I_{gt} to be a ground truth boundary of the image, and I_b to be the boundary calculated by superpixel segmentation method, *TP*(I_b, I_{gt}), *FN*(I_b, I_{gt}) and *BR*(I_b, I_{gt}) represent true positives, false negatives and boundary recall, respectively:

$$BR = \frac{TP(I_b, I_{gt})}{TP(I_b, I_{gt}) + FN(I_b, I_{gt})} \tag{5}$$

where *TP* is defined as the number of boundary pixels in I_{gt} with at least one boundary pixel in I_b in range d.

In contrast, the number of boundary pixels in I_{gt} is denoted by *FN* so that there is no boundary pixel found in I_b in range d. The boundary recall is expected to be high in order to make superpixels respect boundaries in an image.

As shown in Fig. 3, our methods (red curve from RGB image and green curve from RGB-D images) perform slightly better than SEEDS and CW. It is worth mentioning that the

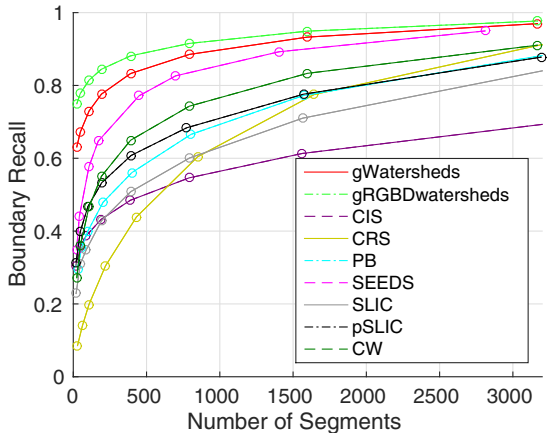


Fig. 3. Boundary recall with different algorithms on the NYU database.

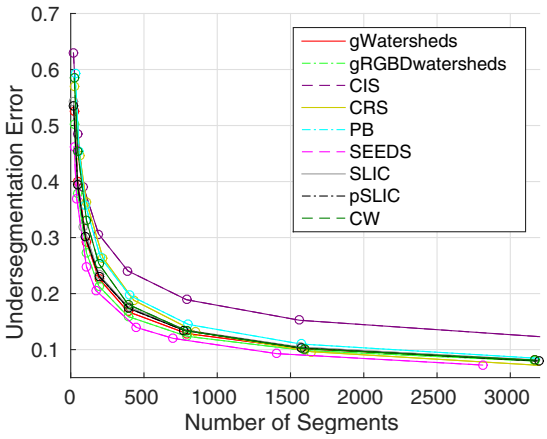


Fig. 4. Under-segmentation errors on the NYU database.

running time of our method is more than 8 times faster than SEEDS.

C. Under-segmentation Error

Under-segmentation error (UE) is used for evaluating the ability of recovering the ground truth object segments using the combination of superpixels. For this purpose, a significant penalty is applied on each superpixel which does not properly overlap with a ground truth segmentation. Algorithms with a high boundary recall are usually subject to a high under-segmentation error. According to [5], this computation of the UE metric is quantified as follows:

$$UE = \frac{1}{N} \sum_{G \in GT} \sum_{SP: SP \cap G \neq \phi} min(SP_i, SP_j) \qquad (6)$$

where N is the number of pixels in an image, G is a ground truth segment, and SP_i as well as SP_j are superpixel segments of SP divided by G.

The results in Fig.4 indicate that the under-segmentation error of our method is slightly higher compared with SEEDS, which exhibits the lowest under-segmentation error, and comparable to most other methods.

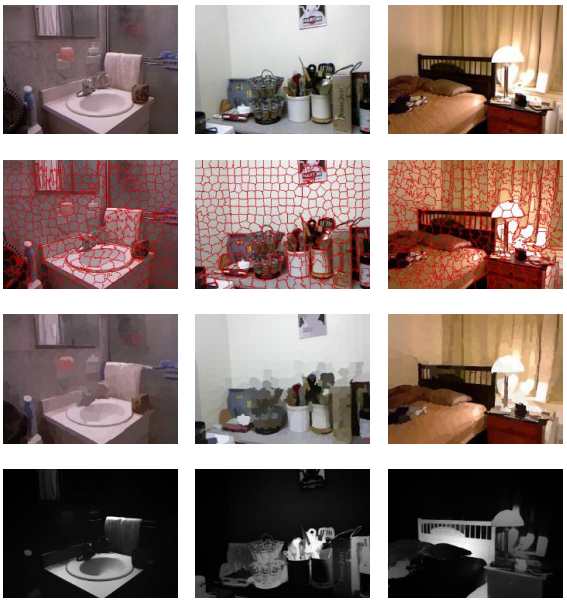


Fig. 6. Saliency map examples. Raw color image samples (top row), superpixel segmentation results (400 superpixels) (second row), mean of image elements (the third row) and saliency maps based on [20] using superpixels (bottom row).

Overall, our method features the best boundary recall with moderate undersegmentation errors at lowest run times, and therefore may be considered a good alternative to other existing state-of-the-art approaches.

In addition to the quantitative performance evaluation, we also give a visual comparison of the segmentation results of all participating oversegmentation methods. As shown in Fig. 5, rows represent the different superpixel algorithms and columns three different image samples, respectively. Image samples were taken from the NYU RGB-D dataset [11].

D. Sample Application

As we mentioned earlier, superpixels are an effective technique to abstract large numbers of pixels from perceptually uniform region in an image. These clustered segments may for example be used to drastically reduce the complexity of salient object detection. Fig. 6 shows the results of object saliency detection using superpixels as proposed in [20] and serves as a proof of concept as well as an interesting topic for future work.

V. CONCLUSION

In this paper we proposed a fast over-segmentation method for growing homogeneous pixels into superpixels in an image. In contrast to previous algorithms, we utilized the MCW transform based on a gradient map. Due to the low runtime complexity of the watershed segmentation, we obtained an efficient superpixel method that delivers results comparable to the state-of-the-art. Experimental results highlight the speed and robustness of the proposed method, with average runtimes being at approximately 15 ms per frame. As such, the proposed method may represent a serious alternative

Fig. 5. Visual appearance of the segments obtained by different state-of-the-art algorithms. Image samples taken from the NYU Depth Dataset [11]. For all approaches parameters were chosen to produce around 400 superpixels. The algorithm name and reference from left-top-row to right-bottom-row: CIS [14], CRS [6], SEEDS [17], SLIC [5], PB [8], preemptiveSLIC [3], CW [3] and the proposed gWatershed algorithm.

to other state-of-the-art approaches, especially if runtime is crucial. Since superpixels maintain the uniqueness and consistency of different objects and decrease computational complexity for later pipeline stages, the application of the proposed method is also highly suitable for vision-based tasks in mobile robotics.

REFERENCES

[1] R. Roberts and F. Dellaert, "Direct superpixel labeling for mobile robot navigation using learned general optical flow templates," in *Proc. IEEE/RSJ Int. Conf. Intelligent Robots and Systems (IROS)*, pp. 1032–1037, Sep. 2014.

[2] A. Concha and J. Civera, "Using superpixels in monocular SLAM," in *Proc. IEEE Int. Conf. Robotics and Automation (ICRA)*, pp. 365–372, May 2014.

[3] P. Neubert and P. Protzel, "Compact watershed and preemptive slic: On improving trade-offs of superpixel segmentation algorithms," in *Proc. IEEE Int. Conf. Pattern Recognition (ICPR)*, pp. 996–1001, Aug 2014.

[4] M. Van den Bergh, X. Boix, G. Roig, B. de Capitani, and L. Van Gool, "SEEDS: Superpixels extracted via energy-driven sampling," in *Proc. IEEE Pro. European Conf. Computer Vision (ECCV)*, (Berlin, Heidelberg), pp. 13–26, 2012.

[5] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, and S. Susstrunk, "SLIC superpixels compared to state-of-the-art superpixel methods," *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, vol. 34, no. 11, pp. 2274–2282, 2012.

[6] C. Conrad, M. Mertz, and R. Mester, "Contour-relaxed Superpixels," in *Proc. Int. Conf. Energy Minimization Methods in Computer Vision and Pattern Recognition (EMMCVPR)*, vol. 8081, (Sweden), pp. 280–293, August 2013.

[7] M.-Y. Liu, O. Tuzel, S. Ramalingam, and R. Chellappa, "Entropy rate superpixel segmentation," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition (CVPR)*, (Washington, DC, USA), pp. 2097–2104, 2011.

[8] Y. Zhang, R. Hartley, J. Mashford, and S. Burn, "Superpixels via pseudo-boolean optimization," in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, pp. 1387–1394, Nov. 2011.

[9] D. Weikersdorfer, D. Gossow, and M. Beetz, "Depth-adaptive superpixels," in *Proc. IEEE Int. Conf. Pattern Recognition (ICPR)*, pp. 2087–2090, Nov 2012.

[10] J. Papon, A. Abramov, M. Schoeler, and F. Wrgtter, "Voxel cloud connectivity segmentation-supervoxels for point clouds," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 2027–2034, Jun. 2013.

[11] N. Silberman, D. Hoiem, P. Kohli, and R. Fergus, "Indoor segmentation and support inference from RGB-D images," in *Proc. IEEE Pro. European Conf. Computer Vision (ECCV)*, (Berlin, Heidelberg), pp. 746–760, 2012.

[12] X. Ren and J. Malik, "Learning a classification model for segmentation," in *Proc. IEEE Int. Conf. Computer Vision (ICCV)*, (Washington, DC, USA), pp. 10–16, 2003.

[13] P. F. Felzenszwalb and D. P. Huttenlocher, "Efficient graph based image segmentation," *Int. J. Comput. Vision*, vol. 59, pp. 167–181, sep. 2004.

[14] O. Veksler, Y. Boykov, and P. Mehrani, "Superpixels and supervoxels in an energy optimization framework," in *Proc. IEEE Pro. European Conf. Computer Vision (ECCV)*, (Berlin, Heidelberg), pp. 211–224, 2010.

[15] Y. Boykov, O. Veksler, and R. Zabih, "Fast approximate energy minimization via graph cuts," *IEEE Trans. Pattern Analysis and Machine Intelligence (PAMI)*, vol. 23, no. 11, pp. 1222–1239, 2001.

[16] S. Beucher and F. Meyer, "The morphological approach to segmentation: the watershed transformation," *Optical Engineering*, vol. 34, pp. 433–481, 1993.

[17] D. Stutz, A. Hermans, and B. Leibe, "Superpixel segmentation using depth information." http://davidstutz.de/, Sep. 2014.

[18] L. Jiang, A. Koch, S. A. Scherer, and A. Zell, "Multi-class fruit classification using RGB-D data for indoor robots," in *Proc. IEEE Int. Conf. Robotics and Biomimetics (ROBIO)*, (Shenzhen, China), Dec. 2013.

[19] L. Jiang, A. Koch, and A. Zell, "Object recognition and tracking for indoor robots using an RGB-D sensor," in *Intelligent Autonomous Systems 13 (IAS-13)*, (Padova, Italy), Jul. 2014.

[20] F. Perazzi, P. Krähenbühl, Y. Pritch, and A. Hornung, "Saliency filters: Contrast based filtering for salient region detection," in *Proc. IEEE Int. Conf. Computer Vision and Pattern Recognition (CVPR)*, pp. 733–740, 2012.