Project Report - Superpixel Segmentation

Mayank Kumar (2014CSB1022)

Devendra Pratap Yadav (2014CSB1010)

CSL 462 - Computer Vision
Department of Computer Science and Engineering,
Indian Institute of Technology, Ropar

1 Abstract

Many computer vision task operate on superpixels nowadays, instead of the raw pixels. Superpixels refer to a group of pixels which share certain properties and constitute a more logical segment of the image as compared to the individual pixels. There is however no single algorithm which guarantees good superpixels. Most superpixel generation algorithms leverage the color and texture information present in images, which often bypasses the semantic information of the objects present in it. We propose a new algorithm which improvises upon some of the previously published algorithms and tries to produce semantically consistent superpixels by using both the color information of the pixels as well as object information, if any, in the image. We use a very popular object detection model, the YOLO object detector, to detect objects across 20 labels (trained on the Pascal VOC dataset).

2 Introduction

Image representation using pixels ignores the semantic information of the object/region present in it. Many Computer Vision tasks require a semantic understanding of objects in the image. Instead of processing a large number of pixels, we can group similar pixels into natural regions called Superpixels.

In general, superpixels should have the following properties:

- Each pixel belongs to only one superpixel
- Low variance in a superpixel
- Edges present at superpixel boundaries, not inside
- Regular structure, efficient to compute, controllable count

Efficient representation via superpixels can improve/speed-up computer vision tasks such as object detection, depth estimation, tracking and semantic segmentation. When finding superpixels, it is desirable to avoid too many or too less superpixels. Almost all algorithms provide some tunable parameters to control the number and size of superpixels generated. We provide a threshold parameter to control this, which ranges between 0 to 1. Refer the image given below to get a feel of what superpixels look like.

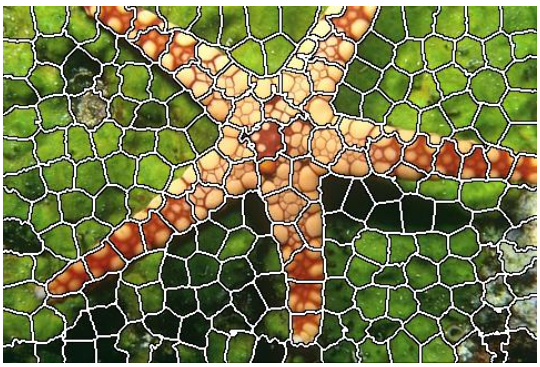


Figure 1: Superpixel segmentation using SLICO algorithm.

3 Related works

3.1 Algorithm categories

Superpixels have been extensively studied during the past decade. The algorithm proposed for this task may be broadly divided into the following categories:
1) **Graph-based algorithms** - Examples - Normalized Cuts algorithm and Felzenszwalb segmentation.
2) **Clustering-based algorithms** - Examples - SLIC and Quickshift.
3) **Energy-based algorithms** - Example - SEEDS algorithm.
4) **Some other categories** - Wavelet-based algorithms, Density-based algorithms, Contour evaluation algorithms, Path-based algorithms, Watershed algorithms.

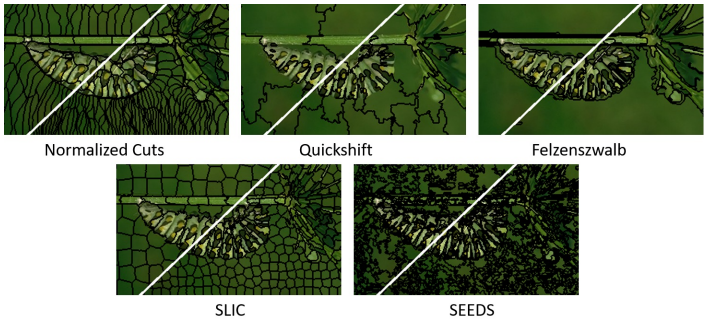


Figure 2: Superpixel segmentation results [14] of various algorithms

3.2 Earlier Proposed Methods

Normalized Cut [2] - This algorithm considers the image as a graph G(V, E). The authors define a metric, named Ncut(A, B). They aim to find the optimal vertex partition with the minimum Ncut(A, B). The algorithm is recursively applied on the subsets A and B, until we obtain the desired number of clusters

SLIC [5] - SLIC stands for Simple Linear Iterative Clustering. It is based on local clustering of the pixels in a 5-D space. The 5-D space is defined by the L, a, b parameters obtained after converting the pixels to the CIELAB color space, and their x and y coordinates.

SEEDS [7] - The initial superpixels are refined by exchanging blocks of pixels as well as single pixels between neighboring superpixels, to minimize a certain energy parameter value. The similarity between neighboring superpixels is calculated by comparing their colour histograms.

Quickshift [4] - Quick shift is a mode seeking algorithm (like mean shift) which instead of iteratively shifting each point towards a local mean instead forms a tree of links to the nearest neighbor which increases the density.

4 Method

We propose a novel method to obtain semantically coherent superpixels in an image. The paradigm we follow is essentially a bottom-up superpixel coalition technique. The core idea behind our algorithm is the fact that if we tune an existing segmentation algorithm in a particular way, we obtain a large number of small-sized superpixels. These superpixels encompass fairly smooth color regions in an image, which can then be combined to form larger and more logical superpixels.

The algorithm pipeline is discussed in detail below:
1) **Quickshift** - The initial step of our algorithm applies a chosen superpixel segmentation algorithm on the image to obtain a set of raw segments. The reason for choosing Quickshift over algorithms like SLIC or Felzenszwalb segmentation is that the former one results in superpixels which stick to the original edges present in the image, and exhibits a competitive running time. We tune kernel size required for Quickshift to obtain a large number of superpixels.
2) **Feature extraction** - The next step involves extracting relevant features from each of the superpixels for later comparisons. Since each superpixel is expected to show negligent color variation, edge based feature extractors like HoG, SURF etc. come to no avail here. We therefore extract the color histogram information for each superpixel and apply a 5x1 mean

filter on it to account for the low density of the obtained vectors. We use 'Histogram intersection' as the similarity measure for our algorithm. The distance between each histogram pair is pre-calculated for optimizing the runtime.

3) **Spatial comparison and union** - We spatially compare the adjacent superpixels only and merge those who are similar above a certain threshold. We employ the union-find data structure in our algorithm to speed up the set merging process. Empirical results show that a threshold ranging from 0.4-0.5 times the maximum similarity amongst all pairs, gives results close to the ground truth human segmented regions.

4) **YOLO object detection** - Simply unifying the superpixels based on color properties is not enough. Consider the case when we have the image of a car. Ideally, the entire car should lie within one big superpixel. But certain components of the car, say the tires, the spoilers, the body etc. may not agree in their color histograms, which make it difficult to merge them together. We therefore apply the standar YOLO object detector on the image to obtain sets of bounding boxes. For each bounding box, we consider the superpixels lying wholly inside it, and relax the merging constraints for them.

5 Results and Discussion

Figure 3: Stanford Background Dataset sample image

5.1 Dataset

We use the **Stanford Background Dataset (SBD)** to evaulate the performance of our proposed algorithm. The Stanford Background Dataset [10] combines 715 images from several datasets. The images are of various sizes and quality. The foreground objects and backgrounds are more complex than Berkeley Segmentation Data Set and Benchmarks 500 [11] dataset.

We evaluate and compare the proposed method with previous algorithms of Quickshift and SLIC. We perform qualitative evaluation by tuning the parameters of all algorithms to get best results. These results are shown at end of paper. We perform quantitative evaluation by comparing metrics such as Boundary Recall, Undersegmentation Error, and Unexplained Variation. The metrics are explained below:

1) **Boundary Recall** [12]- Consider FN(G, S) and TP(G, S) to be the number of false negative and true positive boundary pixels in S with respect to G, where S is the superpixel clustering generated by an algorithm and G is the groundtruth clustering. We define the boundary recall as follows:

BoundaryRecall(G, S) = TP(G, S) / (TP(G, S) + FN(G, S))

2) **Undersegmentation Error** [13] - This parameter measures the overlapping of superpixels with respect to the nearby superpixels in the ground-truth image. Lower values are better.

$$UndersegmentationError = \frac{1}{N}\left[\sum_{S\in GT}\left(\sum_{P:P\cap S\neq\emptyset} min(P_{in}, P_{out})\right)\right]$$

Where N is the number of pixels, $S \in GT$ is ground truth superpixel, P is predicted superpixel. P_{in} and P_{out} refer to overlapping and non-overlapping area of P if it intersects with S.

3) **Unexplained Variation** [15] - This measures the amount of variation in color within superpixels. Lower values are better.
$UnexplainedVariation = 1 - R^2$, where R^2 is:.

$$R^2 = \frac{\sum_i(\mu_i - \mu)^2}{\sum_i(x_i - \mu)^2}$$

Here, μ_i is mean color of superpixel, x_i is actual pixel value, μ is mean color of whole image.

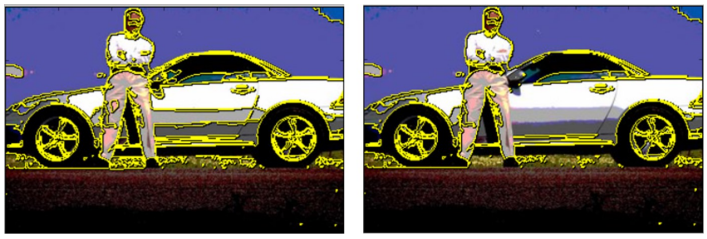
We implement all these evaluation metrics and compare algorithms on Stanford Background Dataset. Results are shown in Figure below.

Metric / Algorithm	SLIC	QuickShift	Our Method
Boundary Recall	0.7398	0.9087	0.7396
Undersegmentation Error	0.1446	0.1242	0.3190
Unexplained Variation	0.2149	0.1453	0.2550

Figure 4: Results on Stanford Background Dataset

Using Quickshift as the base method, we reduce the number of superpixels by merging them. This often leads to lack of edge adherence. However, we still obtain Boundary Recall comparable to SLIC while being qualitatively similar to ground truth. One must note that oversegmentation in SLIC and Quickshift don't add to Undersegmentation Error. Due to fixed threshold for all images, our method often get superpixels overlapping with several ground truth superpixels. This leads to high Undersegmentation Error. Since our method focuses on capturing parts of objects in the same superpixel, the colour variation in superpixels increases. Still, our method is competitive with SLIC. It must be noted that high boundary recall and low variation of Quickshift is observed due to high oversegmentation. These metrics are not a good indicator since the dataset itself is focused on object segmentation rather than superpixel segmentation. The qualitative results give a better comparison of semantic superpixel segmentation.

Using YOLO object detector to further merge superpixels gives more semantically meaningful superpixels. It merges any small, slightly varying superpixels which may be segmenting part of an object into one object superpixel. A result showing improvement in shown below:



Without YOLO Using YOLO

Figure 5: Improvement on using YOLO object detection

Overall, our proposed method is competitive with previous algorithms and our novel approach of utilizing object detection gives good sematic superpixels.

5.2 Extensions and Future Work

To remove the need to tune the merging threshold, we tried to train a classifier which given color histograms of two superpixels, predicts whether these superpixels should be merged or not. For training data, we extracted (fully contained)square patches for each superpixel in images from Stanford Background Dataset. Positive samples(should be merged) were pair of patches from same superpixel. Negative samples(shouldn't be merged) were pair of patches from different superpixels. Using two hidden layers with (400,200) neurons, we obtained classification accuracy of 85%. However, since Stanford Background Dataset images have highly object-based and not superpixel-based segmentation, the actual utility of classifier to merge superpixels was limited. We obtained slightly better results than our base method but at much higher runtime. This approach can be improved using better dataset for obtaining patches at various scales.

Further, we may aggressively merge superpixels first, then use a CNN based classifier to split a bad superpixel(one containing prominent edges and large color variation) into separate superpixels.

Lastly, we can improve our current algorithm by applying some constraints on superpixel size and number during the merging process. Sim-

ilar to object detection, we may add more secondary indicators for improving superpixel merging.

6 References

[1] X. Ren, J. Malik, Learning a classification model for segmentation, International Conference on Computer Vision, 2003

[2] J. Shi, J. Malik, Normalized cuts and image segmentation, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2000

[3] P. F. Felzenswalb, D. P. Huttenlocher. Efficient graph-based image segmentation. International Journal of Computer Vision, 2004

[4] A. Vedaldi, S. Soatto. Quick shift and kernel methods for mode seeking. European Conference on Computer Vision, 2008

[5] R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. SÃijsstrunk. SLIC superpixels. Technical report, Ecole Polytechnique Federale de Lausanne, 2010

[6] M. Y. Lui, O. Tuzel, S. Ramalingam, R. Chellappa. Entropy rate superpixel segmentation. Conference on Computer Vision and Pattern Recognition, 2011

[7] M. van den Bergh, X. Boix, G. Roig, B. de Capitani, L. van Gool. SEEDS: Superpixels extracted via energy-driven sampling. European Conference on Computer Vision, 2012

[8] J. Yao, M. Boben, S. Fidler, R. Urtasun, Real-time coarse-to fine topologically preserving segmentation, IEEE Conference on Computer Vision and Pattern Recognition, 2015

[9] R. Mester, U. Franke, Statistical model based image segmentation using region growing, contour relaxation and classification, SPIE Symposium on Visual Communications and Image Processing, 1988

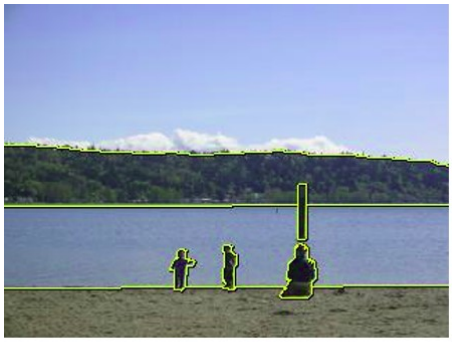
[10] S. Gould, R. Fulton, D. Koller. Decomposing a scene into geometric and semantically consistent regions. International Conference on Computer Vision, 2009

[11] P. Arbelaez, M. Maire, C. Fowlkes and J. Malik. Contour Detection and Hierarchical Image Segmentation IEEE TPAMI, Vol. 33, No. 5, 2011

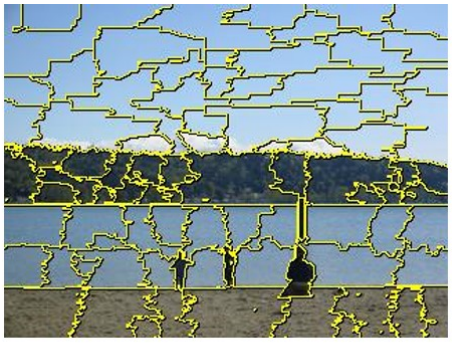
[12] D. Martin, C. Fowlkes, J. Malik, Learning to detect natural im- age boundaries using local brightness, color, and texture cues, IEEE Transactions on Pattern Analysis and Machine Intelligence, 2004

[13] A. Levinshtein, A. Stere, K. N. Kutulakos, D. J. Fleet, S. J. Dickinson, K. Siddiqi, TurboPixels: Fast superpixels using ge- ometric flows, IEEE Transactions on Pattern Analysis and Ma- chine Intelligence,2009

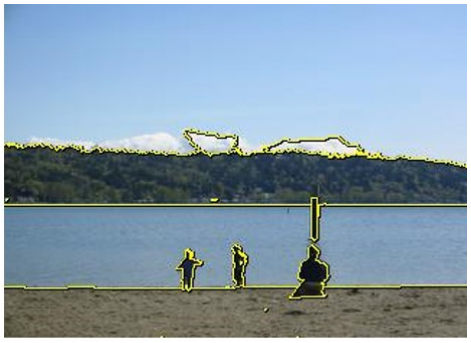
[14] D. Stutz et al, Superpixels: An Evaluation of the State-of-the-Art, arXiv:1612.01601, 2017 [15] A. P. Moore, S. J. D. Prince, J. Warrell, U. Mohammed, G. Jones. Superpixel lattices. CVPR, 2008



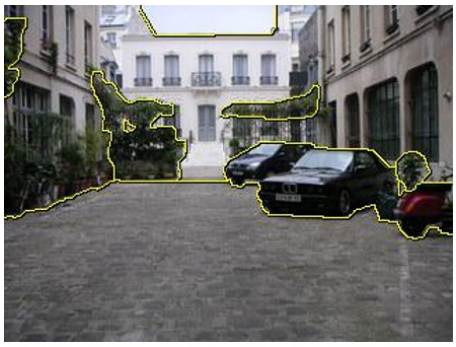
Original image



Quickshift



Superpixel Reduction

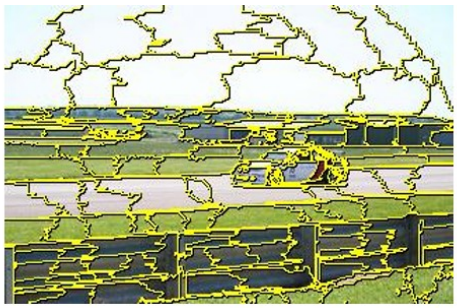


Original image

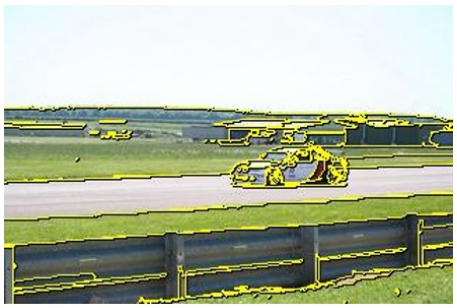
Quickshift

Superpixel Reduction

Original image



Quickshift



Superpixel Reduction