Comparative analysis between OpenCV's and a custom MSER implementation

Alessandro Acquilino (5198425)

June 2025

1 Abstract

Maximally Stable Extrema Regions is a technique for blob detection proposed by Matas et al. for image correspondence that can also be used for text detection. In this report, the definitions for the algorithm are outlined, along with a sketch of the steps involved. A high-level comparison between OpenCV's implementation and my own is presented, also using purposefully developed visualizations. The results are also compared with OpenCV's implementation using a bidirectional testing approach on a dataset containing mainly license plate images. My implementation returns a smaller set of strong regions, as opposed to OpenCV's, that is more permissive given the parameters used. Ultimately, a possible license plate recognition process is described as a simplified version of a text classification task.

2 Introduction

The original MSER paper [Mat+04] introduces the technique with the goal of finding correspondences between stereo images by means of feature matching. The paper proposes a new kind of feature called *extremal regions*. This kind of features can be defined in the following way:

Image I is a mapping $I : \mathcal{D} \subset \mathbb{Z}^2 \to \mathcal{S}$. Extremal regions are well defined on images if:

1. $\mathcal{S}$ is totally ordered, i.e. reflexive, antisymmetric and transitive binary relation $\leq$ exists. In this paper only $S = \{0, 1, ..., 255\}$ is considered, but extremal regions can be defined on e.g. real-valued images ($\mathcal{S} = \mathbb{R}$).
2. An adjacency (neighbourhood) relation $A \subset \mathcal{D} \times \mathcal{D}$ is defined. In this paper 4-neighbourhoods are used, i.e $p, q \in \mathcal{D}$ are adjacent (pAq) iff $\sum_{i=1}^{d} |p_i - q_i| \leq 1$.

Region $\mathcal{Q}$ is a contiguous subset of $\mathcal{D}$, i.e. for each $p, q \in \mathcal{Q}$ there is a sequence $p, a_1, a_2, ..., a_n, q$ and $pAa_1, a_iAa_{i+1}, a_nAq$.

(Outer) Region Boundary $\partial\mathcal{Q} = \{q \in \mathcal{D} \setminus \mathcal{Q} : \exists p \in \mathcal{Q} : qAp\}$, i.e. the boundary $\partial\mathcal{Q}$ of $\mathcal{Q}$ is the set of pixels being adjacent to at least one pixel of $\mathcal{Q}$ but not belonging to $\mathcal{Q}$.

Extremal Region $\mathcal{Q} \subset \mathcal{D}$ is a region such that for all pixels $p \in \mathcal{Q}, q \in \partial\mathcal{Q} : I(p) > I(q)$ (maximum intensity region) or $I(p) < I(q)$ (minimum intensity region).

Maximally Stable Extremal Region (MSER): Let $\mathcal{Q}_1, \ldots, \mathcal{Q}_{i-1}, \mathcal{Q}_i, \ldots$ be a sequence of nested extremal regions, i.e., $\mathcal{Q}_i \subset \mathcal{Q}_{i+1}$. An extremal region Q_{i^*} is maximally stable iff $q(i) = \frac{|\mathcal{Q}_{i+\Delta} \setminus \mathcal{Q}_{i-\Delta}|}{|\mathcal{Q}_i|}$ has a local minimum at $i = i^*$.

Table 1: Definition from the original paper.

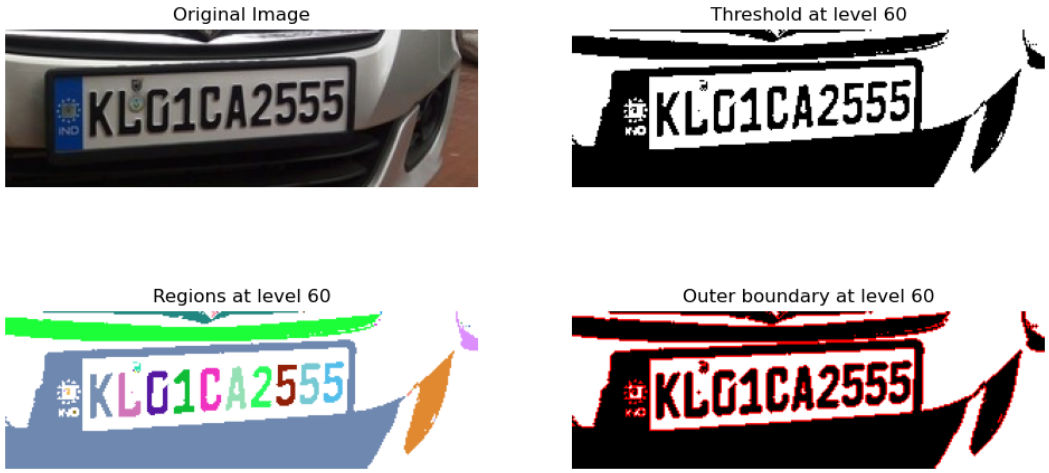

Figure 1: Visualizations of some of the definitions.

Section 3 provides a high-level overview of the algorithm.
Section 4 compares the implementations and the results of OpenCV's version and custom implementation, with a focus on the problem of duplicates with an improvement on the definition of *maximally stable*. The results are compared with bidirectional testing: first considering OpenCV's results as ground truth; then inverting the roles.
Section 5 presents the task of license plate recognition as a simplified text detection problem and provides an overview of a possible procedure.

3 MSER Algorithm

The version of the algorithm that I developed [Acq] is divided in the following broad steps:

1. Convert the image to grayscale.
2. Sort the pixels in $O(n)$ using `BINSORT`.
3. Iterate over the bins in increasing or decreasing order and update a `UNION-FIND` data structure to efficiently maintain connectivity between extremal regions.
4. Compute the stability of each region at each intensity level and ignore those above a certain threshold.
5. For each sequence of nested, stable enough regions (which, from now on, will be referred to as *lineage*) find the most stable region.

4 Comparison

4.1 Implementations

4.1.1 OpenCV

OpenCV implements two versions of the MSER algorithm: one for gray-level images and one for color images. We will focus on gray-level images for this comparison.

The gray-level implementation uses an improved linear version [NS08] of the algorithm that runs in true $O(n)$ worst case, unlike the quasi-linear $O(\alpha(n))$ version of the original paper. This improved version is also much more memory-efficient and more cache-friendly.

OpenCV does not seem to properly suppress the non-maxima regions. Instead, it uses a parameter called MIN_DIVERSITY to compare the sizes of the nested regions: If they do not differ enough, one of them is pruned. To compare the results, I kept all the default parameters taken from the OpenCV implementation and applied a custom post-processing based on the overlap of the bounding boxes of the regions to prune the duplicates.

4.1.2 Custom

For my report, I followed an implementation derived from the original with an improvement that greatly reduced the number of duplicates.

First, the hierarchy of the regions (Fig. 2) is built using the `UNION-FIND` data structure, then, for each region $Q_{i,j}$ at intensity level i, the corresponding $q_i(j)$ is computed from the definition above.
Unlike the original paper and the OpenCV version, where all i^* corresponding to $q(i^*)$ local minima are taken, my implementation only takes the global minima of each *lineage*.

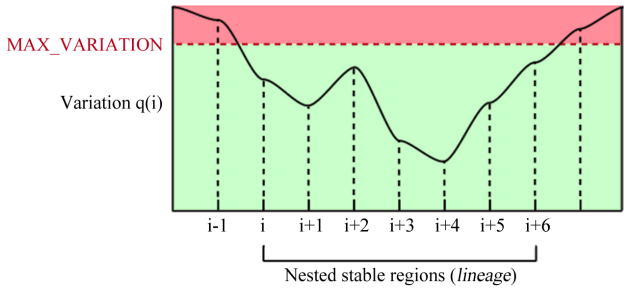

Figure 3: Example of lineage over intensity levels.

From the original definition (Tab. 1) both $\mathcal{Q}_{i+1}$ and $\mathcal{Q}_{i+4}$ (Fig. 3) would be considered *maximally stable extremal regions*, even though they are extremely similar regions, since they are stable over a small range of intensity levels, while my implementation only returns $\mathcal{Q}_{i+4}$.

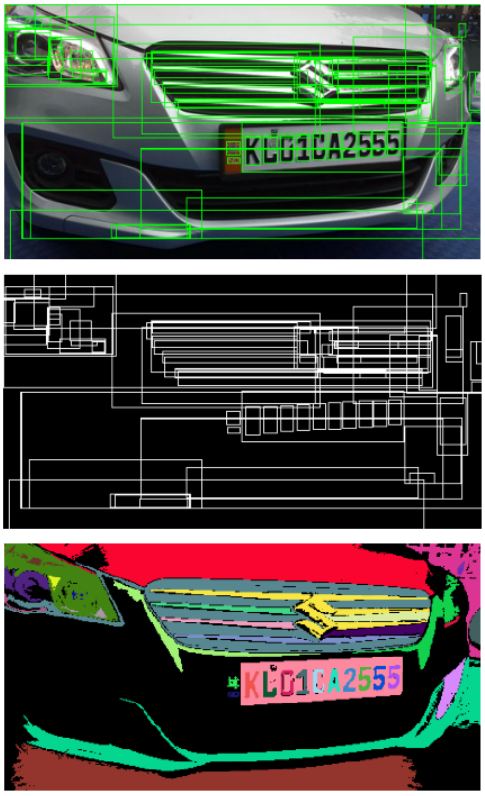

Figure 4: Sample output with highlights.

Figure 2: Example of tree visualization of region evolutions and lineages in a forward pass. Each chain of colored nodes is a lineage.

4.2 Experiments & Results

To compare the results, I first extracted the MSER regions with both implementations on the first 100 images of the dataset [Mar].

For each region found by my custom implementation, I then applied an `Intersection-over-Union` metric with every OpenCV region after pruning to find the best match (if any). This IoU is not calculated on the bounding boxes, but rather on the single pixels of the regions to provide more accurate results. If the best IoU for a specific region was below a threshold of 0.5, a *miss* would be accounted for since the region was not detected by OpenCV. If the match is exact (IoU = 1) a *perfect match* is counted.

The MSER parameters used are:

- DELTA: 5
- MIN_AREA: 60
- MAX_AREA: 14400
- MAX_VARIATION: 0.25
- MIN_DIVERSITY: 0.2 (only relevant for the OpenCV implementation)

I examined the results twice: first assuming OpenCV's implementation as the ground truth to compare with my implementation, then using the same metrics but with the roles reversed.

4.2.1 Regions detected

#Custom regions	#CV Regions (before pruning)	#CV Regions (after pruning)	Reduction %
97.32	440.45	103.69	76.5%

Table 2: Average number of regions detected over 100 samples.

As expected (Table 2) the OpenCV implementation yields over 4X the number of MSER regions compared to my custom implementation, however, once the duplicates are removed, the numbers become much more comparable. Further analysis would be required to verify if an efficient implementation of my custom version could outperform the OpenCV's MSER + pruning in terms of execution speed.

4.2.2 OpenCV as ground truth

Avg. IoU	Median IoU	Min. IoU	#Misses	#Perfect matches
0.8496	0.8903	0.1586	3.01	3.58

Table 3: Average results with CV as the ground truth over 100 samples.

4.2.3 Custom implementation as ground truth

Avg. IoU	Median IoU	Min. IoU	#Misses	#Perfect matches
0.8703	0.9031	0.1597	2.21	3.63

Table 4: Average results with my custom implementation as the ground truth over 100 samples.

The average and median IoU metrics show a strong agreement of 90%, indicating that the two implementations mostly return matching regions with similar areas. This value is dependent on the adopted pruning strategy and could potentially show improvements depending on which duplicate regions are removed.
If we analyze the number of misses and perfect matches, OpenCV appears able to match my custom implementation more than vice-versa. This suggests that my implementation is able to find more fundamental regions, as supported by (Fig. 5) where the missed regions are either irrelevant or a merge between two detected regions.

Figure 5: Original sample with the 4 CV regions missed by my custom implementation highlighted.

5 Conclusions

In this report, I presented an outline of the MSER algorithm, from the formal definitions to a sketch of the implementation. I then compared at an high level

In future work, I intend to develop an optimized version of the algorithm and to compare the algorithms with different sets of parameters; possibly exploiting the parallel processing capabilities of GPUs.

5.1 MSER for license plates detection

General text detection is a complicated task. It requires being able to identify text in a wide variety of scenarios, for instance:

- Different font styles.
- Different orientations.
- Different illuminations.
- Irregular layouts.

And many more.

However, if we restrict the detection to license plates, we can greatly reduce the scope of the problem and introduce some conditions that the detector must abide by.

Every country has a set of rules regarding the fonts allowed, the dimension of the characters, the number of characters, and the set of characters available for each spot of the plate. Since license plates are meant to be seen from far away, we can expect easily identifiable letters with good contrast over a wide range of intensities, which is exactly the main strength of MSER.

If we further restrict the search to, for instance, standard sized italian license plates for commercial vehicles, we can impose other conditions:

- The plate must have dimensions 360mm $\times$ 110mm (so a $36 : 11$ aspect ratio).
- In the bottom left corner there must be an "I".
- The code has a fixed structure of: 2 capital letters from A to Z, 3 digits from 0 to 9, 2 capital letters from A to Z.
- The characters are black on a white background.
- The characters have a fixed size, font and aspect ratio.

We can simplify more general text detection procedures [Che+11].
A procedure for identifying license plates could be the following:

1. Run MSER to detect extremal regions.
2. Remove all regions with non-plausible aspect ratios (Fig. 6).
3. From a dictionary of license plate structures, try to match the remaining regions. This can be done in the following way:
   - 3.1 For each region $\mathcal{Q}_i$, check that a region $\mathcal{Q}_j$ exists on it's right side within a certain distance τ. That region should also have a bounding box very similar in shape and size to the one currently considered, otherwise, it can be discarded. Further analysis of stroke width [Che+11] can be done to refine the findings.
   - 3.2 Repeat this procedure recursively on $\mathcal{Q}_j$ until there are no more regions on the right within distance τ.
   - 3.3 At the end, take any chain of regions that has a compatible number of regions.
4. Use a classical OCR technique or a small NN classifier to identify the characters within the regions.

Figure 6: Regions after aspect ratio cleanup.

We could further improve the method by considering that, by design, the background of the text of a license plate is contrastive with respect to the text. We can thus search within the MSER regions extracted by the backward pass to find a region whose bounding box closely encapsulates all the characters and automatically discard the chains that do not have such a region.

References

[Mat+04] Jiri Matas et al. "Robust wide-baseline stereo from maximally stable extremal regions". In: *Image and vision computing* 22.10 (2004), pp. 761–767.

[NS08] David Nistér and Henrik Stewénius. "Linear Time Maximally Stable Extremal Regions". In: *Computer Vision – ECCV 2008*. Ed. by David Forsyth, Philip Torr, and Andrew Zisserman. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 183–196. ISBN: 978-3-540-88688-4.

[Che+11] Huizhong Chen et al. "Robust text detection in natural images with edge-enhanced Maximally Stable Extremal Regions". In: *2011 18th IEEE International Conference on Image Processing*. 2011, pp. 2609–2612. DOI: 10.1109/ICIP.2011.6116200.

[Acq] Alessandro Acquilino. *Code and samples used*. URL: https://github.com/YeeveX/CV-Project.

[Mar] Andrew Maranhão. *Car License Plates Dataset*. URL: https://www.kaggle.com/datasets/andrewmvd/car-plate-detection.