An Implementation of Multi-Dimensional Maximally Stable Extremal Regions

Andrea Vedaldi

February 7, 2007

Contents

1 Introduction

We describe an implementation of the Maximally Stable Extremal Region ([3], Sect. 2, MSER) feature detector[1] and an immediate multi-dimensional generalization ([1], Sect. 2). We propose an algorithm (Sect. 3) that is essentially union-tree with path-compression and union-by-rank (see for instance [4]). However we do not use the N-tree graph of [4] as for the purpose of fitting ellipses to the MSERs a much simpler data structure turns out to be sufficient (Sect. 3.4). Finally we describe a few experiments where 3-D MSERs are used to track regions in video sequences (Sect. 4). This is different from [2] as we do not extract regions from each frame independently, but directly a 3-D region from the stacking of such frames.

[1] The implementation can be downloaded at `http://vision.ucla.edu/~vedaldi/code/mser/mser.html`

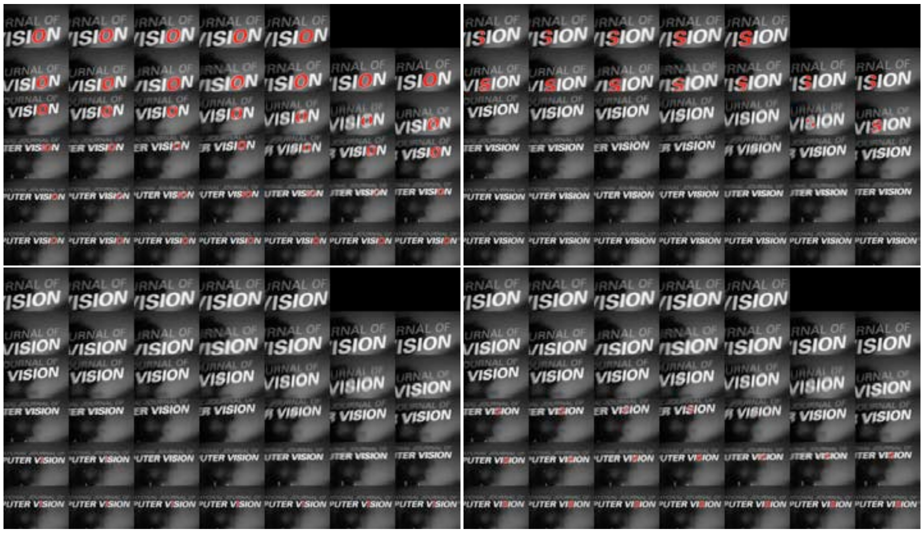

Figure 1: **MSER tracker.** By computing 3-D MSERs on the stacking of the frames of a video sequence we obtain a simple tracker (Sect 4).

2 Maximally stable extremal regions

Here an image $I(x), x \in \Lambda$ is a real function of a finite set Λ with a topology τ. Elements of Λ are called *pixels*. For simplicity, we take $\Lambda = [1, 2, \ldots, N]^n$ and the topology τ induced by the 4-way or 8-way neighborhoods, but we do not restrict ourselves to $n = 2$ as [3].

A *level set* $S(x),\ x \in \Lambda$ of the image $I(x)$ is the set of pixels that have intensity not greater than $I(x)$, i.e.

$$S(x) = \{y \in \Lambda : I(y) \leq I(x)\}.$$

A *path* $(x_1, \ldots, x_n)$ is a continuous sequence of pixels (i.e. such that x_i and x_{i+1} are 4-way or 8-way neighbors for $i = 1, \ldots, n-1$). A *connected component* C of the set Λ is a subset $C \subset \Lambda$ for which each pair $(x_1, x_2) \in C^2$ of pixels is connected by a path fully contained in C. The connected component is *maximal* if any other connected component C' containing C is equal to C. An *extremal region* R is a maximal connected component of a level set $S(x)$. We denote by $\mathcal{R}(I)$ the set of all extremal regions of image I.

Stability criteria. Among all extremal regions $\mathcal{R}(I)$, we are interested in the ones that satisfy certain stability criteria which we introduce next. Let the *level* $I(R)$ of the extremal region R be the maximum image value attained in the region R, i.e.

$$I(R) = \sup_{x \in R} I(x). \tag{1}$$

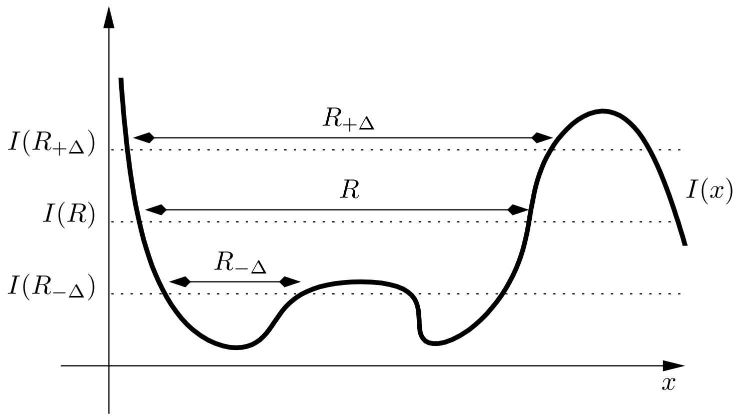

Figure 2: **Stability criteria.** We show an extremal region R of a one dimensional image $I(x)$ and the corresponding extremal regions $R_{+\Delta}$ and $R_{-\Delta}$ (see text). Stability is computed based on the area variation of such regions (Sect. 2).

Let $\Delta > 0$. Let $R_{+\Delta}$ be the smallest extremal region that contains iR and has intensity which exceeds of at least Δ the intensity of R (Fig. 2), i.e.

$$R_{+\Delta} = \operatorname{argmin}\{|Q| : Q \in \mathcal{R}(I),\ Q \supset R,\ I(Q) \geq I(R) + \Delta\}. \tag{2}$$

Similarly, let $R_{-\Delta}$ be the biggest extremal region containing R that has intensity which is exceeded by at least Δ by R, i.e.

$$R_{-\Delta} = \operatorname{argmax}\{|Q| : Q \in \mathcal{R}(I),\ Q \subset R,\ I(Q) \leq I(R) - \Delta\}. \tag{3}$$

Consider the area variation

$$\rho(R;\Delta) = \frac{|R_{+\Delta}| - |R_{-\Delta}|}{|R|}.$$

The region R is *maximally stable* if it is a minimum for the area variation, in the following sense: $\rho(R;\Delta)$ is smaller than $\rho(Q;\Delta)$ for any extremal region Q “immediately contained” or “immediately containing” R. We say that an extremal region R *immediately contains* another extremal region Q if $R \supset Q$ and if R' is another extremal region with $R \supset R' \supset Q$, then $R' = R$. Note that this notion makes sense because the base set Λ is finite.

3 Regions computation

We describe an efficient algorithm for the computation of the maximally stable extremal regions of an image $I(x)$ defined on a discrete domain Λ.

3.1 Enumerating extremal regions

We describe first a method to enumerate all extremal regions of a given image I. Let $x_1, x_2, \dots, x_N \in \Lambda$ be a sorting[2] of the image pixels by increasing inteisty value, i.e.

$$I(x_1) \leq I(x_2) \leq \dots I(x_N).$$

We compute extremal regions incrementally, by considering larger and larger image subdomains $\Lambda_t = \{x_1, x_2, \dots, x_t\} \subset \Lambda$ for $t = 1, \dots, N$. Denote by $I_t = I|_{\Lambda_t}$ the restriction of the image I to the subset Λ_t.

For $t = 1$, $\Lambda_1 = \{x_1\}$ is trivially an extremal region of the image $I_!$ and level $I(x_1)$. For $t = 2$, either x_1 and x_2 are connected and Λ_2 is an extremal region of I_2, or they are not and $\{x_2\}$ is an extremal region of Λ_2. Moreover Λ_1 is an extremal region of I_2 if, and only if, $I(x_2) \neq I(x_1)$. This is captured in general by:

Lemma 1. *Let t be one of $1, 2, \dots, N-1$. Let $R_1, \dots, R_K$ be all the extremal regions of I_t. Let*

- *$\mathcal{K}_1$ the subset of indices k for which $I(R_k) \neq I(x_{t+1})$ and*
- *$\mathcal{K}_2$ the subset of indices k for which $I(R_k) = I(x_{t+1})$ but x_{t+1} is* not *connected to R_k and*
- *let $\mathcal{K}_3$ be the subset of indices k for which x_{t+1} is connected to R_k.*

Then

1. *for all $k \in \mathcal{K}_1 \cup \mathcal{K}_2$ the set R_k is an extremal region of I_{t+1};*
2. *the set $R = \{x_{t+1}\} \cup_{k \in \mathcal{K}_3}$ is an extremal region of I_{t+1};*
3. *all extremal regions of I_{t+1} are obtained either as (1) or (2).*

Proof. By definition each R_k is a maximal connected component of the set $S_t(R_k) = \{x \in \Lambda_t : I(x) \leq I(R_k)\}$. If $k \in \mathcal{K}_1$, then $I(R_k) \neq I(x_{t+1})$, $S_t(R_k) = S_{t+1}(R_k)$ and R_k is a maximal connected component of $S_{t+1}(R_k)$ as well. If $k \notin \mathcal{K}_1$, then $S_{t+1}(R_k) = S(R_k) \cup \{x_{t+1}\}$. However if $k \in \mathcal{K}_2$, then R_k and x_{t+1} are not neighbors and R_k is still maximal in $S_{t+1}(R_k)$. Finally, $\{x_{t+1}\}$ together with all the regions R_k of level $I(R_k) \leq I(x_{t+1})$ which are neighbors of x_{t+1}, i.e. $k \in \mathcal{K}_3$, constitute a new extremal region. To see this, note that (i) $R \subset S(x_{t+1})$, (ii) R is connected because the subregions R_k are connected and any two points in two different subregions are connected through x_{t+1} by construction and (iii) R is maximal as if not, one could add a pixel $y \in \Lambda_t = \Lambda_{t+1} - \{x_{t+1}\}$ to R that would be either an extension of one of the extremal regions R_k of image I_t or $\{y\}$ would be a new extremal region of image I_t by itself.

Finally, we need to show that the listing is exhaustive. So let R be an extremal region of image I_{t+1}. If $R \subset S_t(R)$, then $x_{t+1} \notin R$ and R is equal to some R_k for $k \in \mathcal{K}_1 \cup \mathcal{K}_2$ by the inductive hypotesis. If, on the other hand, $x_{t+1} \in R$, then R is obtained as (2). $\square$

[2] This can be done in linear time by using bucket-sort.

Lemma 1 suggests a simple algorithm to enumerate extremal regions. The idea is to consider one pixel at time in the order $x_1, x_2, \dots$ growing extremal regions for the intermediate images I_t until $I_N = I$ is reached.

Formally, this process can be implemented by means of a forest of pixels. At time t the forest represents all the union operations that have been performed so far according to point (2) of Lemma 1. Since extremal regions are only generated by such union operations, the tree stores all the extremal regions of all intermediate images $I_1, \dots, I_t$.

Let us consider the addition of pixel x_{t+1} to the forest. Following Lemma 1, we must search for all extremal regions $R_1, \dots, R_k$ of image I_t which are neighbors of x_{t+1} and join them to x_{t+1} to obtain the new region R. This is done by scanning the neighbors $y \in \Lambda_t$ of x_{t+1} and, for each of them, climbing the tree in search for the appropriate extremal regions R_k. In practice, we simply take the union of all sets $S(y) \cup S(\pi(y)) \cup S(\pi^2(y)) \cup \cdots \cup S(\mathrm{root}(y)) = S(\mathrm{root}(y))$, where $S(y)$ is the subtree rooted at y, $\pi(y)$ is the parent of y and $\mathrm{root}(y)$ is the root of the tree that contains y. While only some of $S(\pi^n(y))$ are indeed extremal regions of image I_t, $S(\mathrm{root}(y))$ always is and, since it covers all other subsets anyway, it is sufficient to join that. The join operation is then encoded in the forset by making x_{t+1} parent of $\mathrm{root}(y)$, i.e. $\pi(\mathrm{root}(y)) \leftarrow x_{t+1}$.

This basic algorithm can be improved significantly by keeping the tree balanced. This is an optimization of the join operation, for which x_{t+1} is not necessarily added to the forest as root; instead one uses as root one of the nodes $\mathrm{root}(y)$ with the goal keeping the tree height short. Although this disrupts partially the property of the forest (some of the extremal regions of the intermediate images $I_1, I_2, \dots$ are lost), the relevant information (i.e. the regions that are extremal regions of I) is preserved, as it can be verified. In particular, regions can be emitted as soon as condition (1) of the Lemma is encountered, which correspond to the case $I(y) \neq I(\pi(y))$.

3.2 Computing the stability score

Once the extremal region tree is computed, we need to calculate the area variation for each region and then selecting the maximally stable ones. The area $|R|$ of each region is computed efficiently as explained in Sect. 3.4. In order to compute the area variation of a region R, we need to figure out the regions $R_{-\Delta}$ and $R_{+\Delta}$. To do this we begin by arranging the extremal regions into a tree where R is parent of R' if R immediately contains R'. Then each region R is considered and the tree is explored to find a region Q for which $R = Q_{-\Delta}$ and the region $R_{+\Delta}$. This is done by scanning the regions $R_0 = R$, $R_1 = \pi(R_0)$, $R_2 = \pi(R_1)$ and so on. If a region $Q = R_i$ satisfies $Q_{-\Delta} = R_0$, then

$$I(R_0) \leq I(R_i) - \Delta < I(R_1).$$

The condition is not necessary though; according to (3) we need to keep the region of maximum area among all the candidate ones. Similarly, if $R_i = R_{+\Delta}$, then

$$I(R_i) \leq I(R_0) + \Delta < I(R_{i+1}).$$

In this case the condition is also sufficient as at most one of such regions exist.

3.3 Cleaning up

The stability score alone may not be sufficient to select only useful regions. In the cleanup phase we

- remove very small and very big regions;
- remove regions which have too high area variation (even if they are indeed minima of the variation score);
- remove duplicated regions.

Duplicated regions arise because, due to noise, the same mode of the local minima score may correspond to more than one local minimum. Duplicated regions are easily found by comparing each MSER R with the MSER R' immediately containing R and removing R if they are too similar.

3.4 Fitting elliptical regions

Fitting elliptical regions amount to computing for each maximally stable extremal region R the first and second order moments, i.e.

$$\mu(R) = \frac{1}{|R|} \sum_{x \in R} x, \qquad \Sigma(R) = \frac{1}{|R|} \sum_{x \in R} (x - \mu)(x - \mu)^\top.$$

Rather than considering directly the centered moment $\Sigma(R)$, it is computationally more convenient to compute

$$M(R) = \frac{1}{R} \sum_{x \in R} x x^\top$$

and use the fact that $\Sigma(R) = M(R) - \mu(R)\mu(R)^\top$. The advantage is that any quantity which is obtained by integrating a function $f(x)$, $x \in \Lambda$ of the image domain (in particular $f(x) = x$ and $f(x) = xx^\top$) can be computed for all regions at once by visiting (in breath first order and from the leaves) each pixel of the forest and summing its value to the parent. The visit order is determined (and can be recorded for later use) during the construction of the forest itself. This simple idea achieves the same efficiency of [4] for the purpose of fitting ellipses.

4 Experiments

Multi-dimensional extremal regions can be compute for instance on volumetric images or video sequences. Here we explore the latter possibility, which should yields to a dynamic extension of MSER, or region tracker. Some results are shown in Fig. 1 and in Fig. 3.

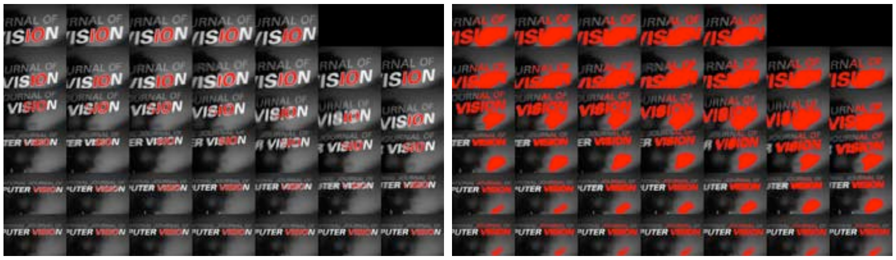

Figure 3: **Examples of incorrectly tracked regions.** Since the shape of the region is not constrained in any way across frames, due to cross-frame overlapping, regions may bleed yielding to inconsistent tracking.

References

[1] M. Donoser and H. Bischof. 3D segmentation by maximally stable volumes. In *ICPR*, 2006.

[2] M. Donoser and H. Bischof. Efficient maximally stable extremal region (MSER) tracking. In *CVPR*, 2006.

[3] J. Matas, O. Chum, M. Urban, and T. Pajdla. Robust wide baseline stereo from maximally stable extremal regions. In *BMVC*, 2002.

[4] E. Murphy-Chutorian and M. Trivedi. N-tree disjoint-set foreset for maximally stable extremal regions. In *BMVC*, 2006.