Superpixel Gridization via Minimum Topological Discrepancy for Fast Object Localization

Wei Feng, *Member, IEEE,* Liang Li, Qing Guo, Liang Wan, *Member, IEEE,* and Jiawan Zhang, *Member, IEEE,*

Abstract—In computer vision, many algorithms can be significantly accelerated by leveraging the regular grid structure of image pixels. These successful techniques, however, cannot be easily used to handle superpixels (SPs), due to their inherent irregularity in size, shape, and topology. In this paper, we propose a general and tractable criterion, namely minimum topological discrepancy (MTD), to optimally transforming arbitrary SP-graphs into regular grids, thus enabling the usage of various grid-based acceleration techniques at superpixel-level.

In our formulation, the topological discrepancy from an SP-grid to the original SP-graph is measured by both positional and structural discrepancy scores, which respectively reflect the locality deviations from SP regions to regular grid cells and the deviations of pairwise SP connections in the SP-grid to those in the original SP-graph. To minimize the grid topological discrepancy, we present a hybrid dynamic programming approach to efficiently constructing the optimal MTD SP-grid for a given arbitrarily-structured SP-graph, which averagely needs only $0.06s$ for common-sized images. Extensive experiments on object localization show that our MTD SP-grid outperforms state-of-the-art pixel- and superpixel-level alternatives in accuracy, efficiency, and the capability of handling large-scale images.

Index Terms—Superpixel gridization, minimum topological discrepancy, dynamic programming, object localization.

1 Introduction

The rapid increase of images and videos in present cyberworld has made *efficiency* much more important than ever, which together with *accuracy* become two essential marks in measuring the feasibility of computer vision algorithms to handle real-world problems.

Last decades have witnessed a number of successful methods, by leveraging the regular grid structure of image pixels, to achieve significant improvement in speed. Typical methods include the integral image [1] and efficient subwindow search [2] in object localization [1][3][4] and tracking [3], [5], dynamic programming in solving variational and low-level vision problems [6][7], and some practical techniques in fast energy minimization [8][9][10][11]. On the other hand, superpixels have also exhibited great potentials in fast and accurate image analysis. A lot of recent feasible algorithms are based on superpixels (or over-segmented regions) in various vision tasks, *e.g.* cosegmentation [12][13], image parsing [14], object detection [15][16], and large displacement optic flow [17]. The popularity of superpixel-level algorithms mainly results from two reasons: (1) superpixels largely reduce the number of variables that need to be handled, thus can cause apparently better efficiency; (2) superpixels usually well respect the local image structures, thus may lead to notably better accuracy than pixel-level alternatives [12][13][15]. However, current superpixel-level algorithms cannot share the great success of the mature grid-based acceleration techniques. The major obstacle lies in the topology irregularity of superpixels.

As shown in Fig. 1(e), superpixels (SPs) generated by grouping similar pixels into perceptually meaningful atomic regions [20], may

- *W. Feng, L. Li and Q. Guo are with the Tianjin Key Laboratory of Cognitive Computing and Application, the School of Computer Science and Technology, Tianjin University, Tianjin, China. Email: wfeng@ieee.org, liangli@tju.edu.cn.*
- *L. Wan and J. Zhang (* Corresponding author, Tel: +86-22-87401540) are with the School of Computer Software, Tianjin University, Tianjin, China. Email: {lwan,jwzhang}@tju.edu.cn.*

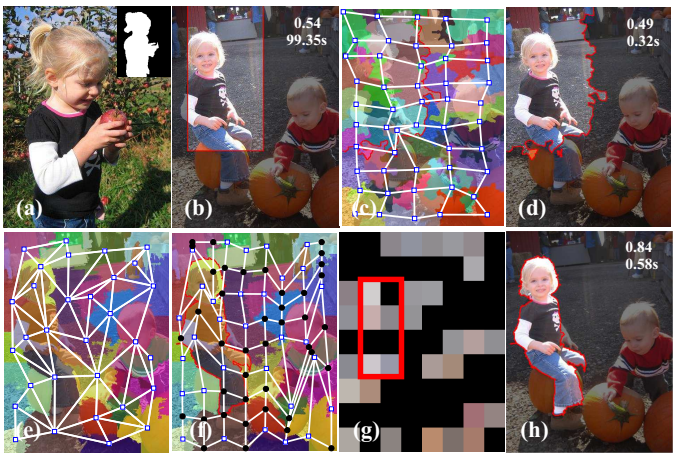


Fig. 1: MTD superpixel gridization. (a) Query image (with query object mask in the corner). (b) Pixel-level RC detection [1] result. (c) SP-grid generated by SEEDS [18]. (d) Superpixel-level RC detection result via SEEDS SPs. (e) Irregular SP-graph generated by SLIC [19]. (f-g) The proposed MTD SP-grid. (h) The RC detection result of our MTD SP-grid. In (b), (d) and (h), the F_1-measure of detection results and overall running time (including SPs generation, gridization and matching time) are shown on the left-up corner. The images are best viewed in color.

have quite different sizes, shapes, and more importantly, topological structures. To unleash the immense potentials of various grid-based acceleration methods at the superpixel-level, we study how to optimally transform arbitrary SP-graphs into regular SP-grids, which we call *superpixel gridization*. The first attempt to the general-purpose superpixel gridization was reported in our previous work [21], which constructs maximum cohesive SP-grids via cascade dynamic programming. In this paper, we propose a general and tractable criterion, namely *minimum topological discrepancy* (MTD), under which our previous work can be regarded as a special case.

Specifically, we formulate a SP-grid as a regular 4-connected grid composed of real *SP nodes* and *dummy nodes*. The goodness of a particular gridizing transformation is measured by the overall topological discrepancy from the output SP-grid, *e.g.* Fig. 1(f), to the original SP-graph, *e.g.* Fig. 1(e). In our model, the topological discrepancy function consists of two parts: (1) *positional discrepancy* $\mathrm{P}(\mathcal{L})$ that reflects the locality deviations from SP regions to the regular grid cells; and (2) *structural discrepancy* $\mathrm{S}(\mathcal{L})$ that measures the deviations of pairwise SP connections in the SP-grid to those in the original SP-graph. To minimize the grid topological discrepancy, we present a hybrid dynamic programming approach to efficiently constructing the optimal MTD SP-grid for a given arbitrarily-structured SP-graph. First of all, we consider only the unary positional discrepancy $\mathrm{P}(\mathcal{G})$ and use min-cost bipartite graph matching [7] to obtain a high-quality initial SP-grid. We then conduct cascade dynamic programming to sub-optimally allocate the SP and dummy nodes in each grid column/row by minimizing the overall discrepancy function.

We empirically find that, for images of common size (*e.g.* 800×600) and fair number of SPs (*e.g.* 100), our approach averagely needs only $0.06s$ to converge to good MTD SP-grids. We have tested our approach in the task of fast object detection and localization on real-world benchmark dataset and have compared with state-of-the-art pixel-level and superpixel-level alternatives. Extensive experiments validated the superior performance of our approach in both accuracy and efficiency.

2 Related Work

2.1 Grid-based acceleration

Since image pixels are evenly sampled into a grid structure, a lot of efforts in last decades have been devoted to explore specific algorithms that leverage the simple and regular grid structure to achieve good efficiency and accuracy. One general and widely-used technique is the integral image [1] that enables fast $\mathrm{O}(1)$ sum evaluation of arbitrary rectangles in an image, thus making brute-force searching based object detection and segmentation practical [1], [5]. Another famous grid-based acceleration is efficient subwindow search (ESS) that parameterizes a rectangle with 4 parameters and uses a branch-and-bound scheme to rapidly seek the target region of interest [2], [3], [4]. Recently, dynamic programming is found to be applicable in solving general labeling

problems, given the hidden variables are grid-structurally correlated [22]. Before that, DP has already been used in solving many variational and low-level vision problems [6], [7]. Besides, there are a number of particular techniques that can be used to apparently speed-up some popular energy minimization algorithms, *e.g.* s-t graph cuts [8], [11] and efficient BP [9].

2.2 Regular and near-regular SPs

The notion of superpixels originates from the over-segmented homogeneous subregions in an image, *e.g.* EGS [23]. Hence, SPs of an image usually form an irregular graph, with the boundaries well-aligned to image edges. Because of this, SPs are widely used by many recent algorithms in solving various vision problems, such as object segmentation and detection [12], [13], [15], [24], [16], image parsing [14], and large displacement optic flow [17]. These algorithms have shown the large potentials of superpixel-level processing in achieving higher efficiency and accuracy than pixel-level alternatives. Recently, researchers start to realize the importance of SP regularities [25], [26]. Typical regular and near-regular SP generation methods include SLIC [19], TurboPixel [27], SuperLattice [28], LatticeCut [29], SEEDS [18] and LSC [30]. Among all these methods, SLIC, LSC and TurboPixel can only produce near-regular SPs, SuperLattice and LatticeCut need pre-computed image edge maps as guidance, only SEEDS can independently produce strict SP-grids without extra help.

The first attempt to converting arbitrary SP-graphs into regular SP-grids was proposed in our previous work [21]. After introducing dummy nodes with position-taking function only, we constructed an optimal SP-grid via maximizing the SP-grid coherence, which is defined as the sum of edge weights, with SP locality and appearance encoded, along all direct paths connecting any two nearest neighbouring real nodes in the grid. As discussed later, it can be regarded as a special case of the MTD criterion proposed in this paper.

2.3 Object detection and localization

Sliding window search is a classical way and has been well studied in object detection and localization [31], [32], [33]. Most sliding window based algorithms seek an optimal bounding box to localize the query object in the target image, by minimizing some proper distance function. Recently, region covariance (R-C) searching [1] and efficient subwindow search (ESS) [2], [3], [4] are two types of successful methods in quickly detecting and localizing complex objects in large-scale images. In our experiments, we have tested the proposed approach and several state-of-the-art pixel- and superpixel-level alternatives in the framework of RC searching.

3 Problem Formulation

The topology structure of any SPs can be represented as a weighted SP-graph $\mathcal{G} = \langle \mathcal{P}, \mathcal{E}_{\mathcal{P}}, W_{\mathcal{P}} \rangle$, where $\mathcal{P}$ is the set of SPs; $\mathcal{E}_{\mathcal{P}}$ is the set of pairwise SP connections that can be established via proper neighboring or closeness relationships in feature spaces;[1] and $W_{\mathcal{P}} = [W_{\mathcal{P}}(p,q)]_{p,q=1}^{|\mathcal{P}|}$ is the edge weight matrix. Specifically, for two neighboring SPs p and q, we set the edge weight $W_{\mathcal{P}}(p,q)$ as their regularized dissimilarity, given by

$$\mathrm{dif}(p,q) = \omega_d \mathrm{D}(p,q) - \omega_f \mathrm{F}(p,q) + \omega_e \mathrm{E}(p,q) + C, \quad (1)$$

where $\mathrm{D}(p,q)$ is the L_2 distance between the regional centroids of p and q. $\mathrm{F}(p,q)$ is the SP color similarity measured by the Bhattacharyya coefficient $\mathrm{Ba}(H_p, H_q) = \sum_{b=1}^{8^3} \sqrt{H_p(b) \cdot H_q(b)}$, where H_p and H_q are RGB-channel independently quantized color histograms of p and q, respectively. $\mathrm{E}(p,q)$ is the average gradient magnitude along the common boundary of p and q that measures the inbetween edge strength. Clearly, Eq. (1) encourages small SPs similarity if they are spatially far from each other, with dissimilar color histograms and strong inbetween boundary. In Eq. (1), non-negative parameters ω_d, ω_f and ω_e control the relative influences of spatial distance, feature similarity and edge strength in $\mathrm{dif}(p,q)$, and C is a constant ensuring all edge weights in the graph are non-negative.

1. In our experiments, we evaluated four types of SP edges, *i.e.* spatial n-connections, $k$nn, ϵnn, and L_1-graph [34]. See Sec. 6 for more details.

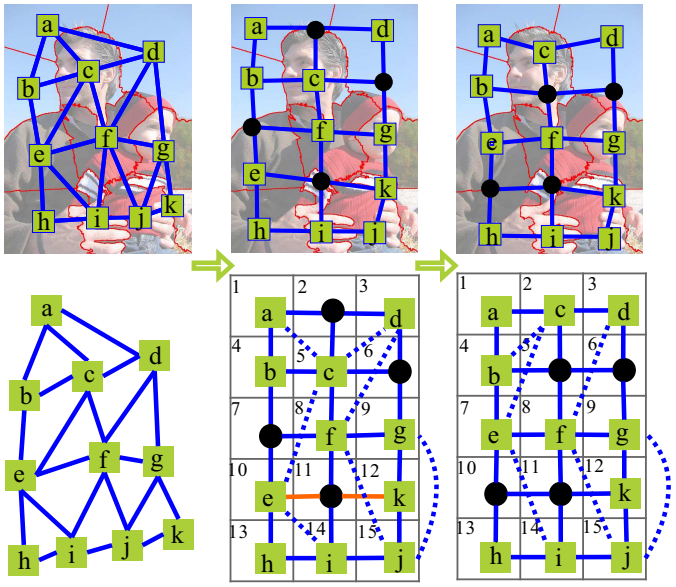


Fig. 2: Illustration of superpixel gridization by optimally adding dummy nodes. The images are best viewed in color.

For a given SP-graph $\mathcal{G}$, our objective, *i.e. SP gridization*, is to optimally transform $\mathcal{G}$ into a regular 4-connected grid (or lattice equivalently) $\mathcal{L} = \langle \mathcal{C}, \mathcal{E}_\mathcal{C}, W \rangle$, with $\mathcal{C}$ as the set of grid cells and $\mathcal{E}_\mathcal{C}$ being the gird edge set. Furthermore, $\mathcal{E}_\mathcal{C}$ can be fully determined by 2D-indexed $\mathcal{C}$, equivalently grid height h and width w (*i.e.* the number of cells in a column and a row on the grid). That is, a grid $\mathcal{L}$ can be simply expressed as $\mathcal{L} = \langle \mathcal{C}, (h, w), W \rangle$. As a result, in SP gridization, if considering h and w as manually set parameters, we need to construct two important mapping relationships from $\mathcal{C}$ to $\mathcal{P}$, and from W to $W_\mathcal{P}$, respectively. As shown in Fig. 2, a reasonable and practical way is to add *dummy nodes* (marked as black circles) [21] and to explore the *d-linked paths*.

***Definition** 1 (d-linked SP nodes & d-linked path):* In an SP-grid $\mathcal{L}$, two SP nodes p and q are *d-linked* to each other, iff: (1) they lie in the same column (or row) in the grid; and (2) all inbetween cells in the column (or row) are filled by dummy nodes. As a special case, two directly connected SP nodes in the grid can also be called d-linked. The path connecting two d-linked SP nodes p and q, composed of (if any) only co-column (or co-row) inbetween dummy nodes and p, q themselves, is called a *d-linked path*.

Through gridization, the dummy nodes have two major roles in the SP-grid $\mathcal{L}$: (1) position taking, which means all grid cells should be filled by either SP nodes $p \in \mathcal{P}$ or dummy nodes $d \in \mathcal{D}$, thus $\mathcal{C} = \mathcal{P} \cup \mathcal{D}$; and (2) linking SP nodes, which means in SP-grid, the d-linked SP nodes can still be viewed as direct neighbors to each other, thus their dissimilarity should be maintained in the corresponding d-linked path. Accordingly, for any two directly neighbored nodes i and j in SP-grid $\mathcal{L}$, the edge weight can be defined as:

$$W(i,j) = \begin{cases} \mathrm{dif}(i,j), & \text{both } i \text{ and } j \text{ are SPs,} \\ 0, & \text{both } i \text{ and } j \text{ are dummy,} \\ \frac{1}{2}\mathrm{dif}(i,t), & i \text{ is SP and } j \text{ is dummy,} \\ \frac{1}{2}\mathrm{dif}(t,j), & i \text{ is dummy and } j \text{ is SP,} \end{cases} \tag{2}$$

where t is the SP node d-linked to the SP node i (or j), with j (or i) as the first or last inbetween dummy node in the d-linked path.

Till now, we can formulate *SP gridization* as the following minimization problem:

$$\begin{aligned} \hat{\mathcal{L}} &= \arg_\mathcal{L} \min f(\mathcal{L}|\mathcal{G}, h, w) \\ &= \arg_{\mathcal{C},W} \min[f_1(\mathcal{C}|\mathcal{P}, h, w) + \alpha f_2(W|W_\mathcal{P})] \end{aligned} \tag{3}$$

where $\alpha > 0$ is a weighting parameter controlling the relative importance of $f_1(\cdot)$ and $f_2(\cdot)$, which measures the cost of two mappings from $\mathcal{P}$ to $\mathcal{C}$ and from $W_\mathcal{P}$ to W, respectively.

4 Minimizing Grid Topological Discrepancy

In this section, we introduce a general yet tractable criterion for SP gridization, namely minimum topological discrepancy (MTD), which aims at minimizing the topological deviations from resultant SP-grid $\mathcal{L}$ to the original SP-graph $\mathcal{G}$. Specifically, we consider the positional discrepancy $\mathrm{P}(\mathcal{L})$ and structural discrepancy $\mathrm{S}(\mathcal{L})$, both of which are elaborated detailedly in the following.

4.1 Positional discrepancy $\mathrm{P}(\mathcal{L})$

We first consider the *positional discrepancy* $\mathrm{P}(\mathcal{L})$, which reflects the locality deviations from SP regions to the regular grid cells. Let $\mathrm{reg}(p)$ denote the irrgular subregions of SP p, and $\mathrm{cell}(i)$

indicate the rectangle region of the i-th grid cell. Their positional discrepancy is defined as:

$$\mathrm{dev}(p,i) = \exp\Big(-\frac{|\mathrm{reg}(p)\cap \mathrm{cell}(i)|}{|\mathrm{cell}(i)|}\Big) + \beta \mathrm{D}(p,i), \tag{4}$$

where $|\cdot|$, in this paper, indicates the set cardinality or region area; $\mathrm{D}(p,i)$ is identical to its definition in Eq. (1) representing the L_2 distance between the regional centroids of SP p and grid cell i, $\beta > 0$ is a weighting parameter. $\mathrm{dev}(p,i)$ measures the positional deviation of SP p to grid cell i, based on which the overall grid positional discrepancy can be measured by

$$\mathrm{P}(\mathcal{L}) = \mathrm{f}_1(\mathcal{C}|\mathcal{P},h,w) = \sum_{p\leftrightarrow i, p\in\mathcal{P}, i\in\mathcal{C}} \mathrm{dev}(p,i), \tag{5}$$

where $p \leftrightarrow i$ means SP p is allocated to grid cell i. Note that, since $\mathrm{dev}(p,i)$ can be independently evaluated for different SPs, the positional discrepancy $\mathrm{P}(\mathcal{L})$ actually defines unary energy terms in the SP gridization model Eq. (3).

4.2 Structural discrepancy $\mathrm{S}(\mathcal{L})$

For the given SP-graph $\mathcal{G}$, the gridization result $\mathcal{L}$ should not severely deviate from the original SP connections, which usually correspond to important image structures. Thus, we define structural discrepancy $\mathrm{S}(\mathcal{L})$ to penalize the pairwise topological differences between SP-grid $\mathcal{L}$ and the original SP-graph $\mathcal{G}$. Specifically, structural discrepancy $\mathrm{S}(\mathcal{L})$ can be measured by

$$\mathrm{S}(\mathcal{L}) = \mathrm{f}_2(W|W_{\mathcal{P}}) = \sum_{\substack{p\in\mathcal{P}\\ q\in\mathcal{P}}} |W_{\mathrm{dLink}}(p,q) - W_{\mathcal{P}}(p,q)|, \tag{6}$$

where matrix W_{dLink} is derived from the grid edge weights matrix W and records the dissimilarity of all d-linked SP pairs in the SP-grid $\mathcal{L}$:

$$W_{\mathrm{dLink}}(p,q) = \begin{cases} \mathrm{dif}(p,q), & p,q \text{ are d-linked in } \mathcal{L} \\ 0. & \text{else} \end{cases} \tag{7}$$

Note, the rationale of W_{dLink} is guaranteed by the grid edge weights defined in Eq. (2), which ensures $\sum_{\substack{s\in\mathrm{dPath}(p,q)\\ s\neq p, s\in\mathcal{P}\cup\mathcal{D}}} W(p,s) = \mathrm{dif}(p,q)$ for any two d-linked SPs in $\mathcal{L}$. $\mathrm{dPath}(p,q)$ denotes the d-linked path of p and q.

Besides, the structural discrepancy $\mathrm{S}(\mathcal{L})$ defined in Eq. (6) can also be viewed as the sum of weights of newly added SP connections in the SP-grid $\mathcal{L}$ and the deleted SP connections from the SP-graph $\mathcal{G}$. See Fig. 2 for illustration. Clearly, $\mathrm{S}(\mathcal{L})$ defines pairwise energy terms in the SP gridization model Eq. (3).

4.3 Hybrid dynamic minimization

From Eq. (5) and (6), the objective of SP gridization in Eq. (3) becomes:

$$\mathrm{Gtd}(\mathcal{L}|\mathcal{G},h,w) = \mathrm{P}(\mathcal{L}) + \alpha \mathrm{S}(\mathcal{L}). \tag{8}$$

For this regularized grid topological discrepancy function, we use a two-step hybrid dynamic programming (DP) approach to obtain the final MTD SP-grid. We first consider only the unary positional discrepancy $\mathrm{P}(\mathcal{L})$ and use min-cost bipartite graph matching to obtain a good initial SP-grid $\mathcal{L}^{(0)}$ [7]. See Fig. 2 for an example. Then, we conduct iterative dynamic programming to sub-optimally allocate the SP and dummy nodes in each grid column/row by minimizing the overall discrepancy function $\mathrm{Gtd}(\mathcal{L}|\mathcal{G},h,w)$.

4.3.1 Dynamic initialization

To initialize, we construct a bipartite graph $\mathcal{G}_B = \{\mathcal{P},\mathcal{L},\mathbf{C}_{\mathrm{A}}\}$, where the across-cost matrix $\mathbf{C}_{\mathrm{A}} = (c_{ij})_{|\mathcal{P}|\times|\mathcal{L}|}$ between $\mathcal{P}$ and $\mathcal{L}$ is defined as

$$c_{ij} = \mathrm{dev}(i,j), \tag{9}$$

where i denotes the #i real-node, and j denotes grid cell j. The bipartite graph is illustrated in Fig. 3. Then we adapt the Hungarian algorithm to assign each cell in $\mathcal{L}$ exactly one SP in $\mathcal{P}$. The cells that do not have corresponding SPs are assigned dummy nodes.

4.3.2 Iterative dynamic programming

Starting from $\mathcal{L}^{(0)}$, our MTD SP-grid $\mathcal{L}^*$ is progressively constructed by optimizing its every column/row under the current configurations of other columns/rows. For a particular column/row ρ of current $\mathcal{L}^*$, we follow the iterative DP procedure [21] to update it under the context of current configurations of other

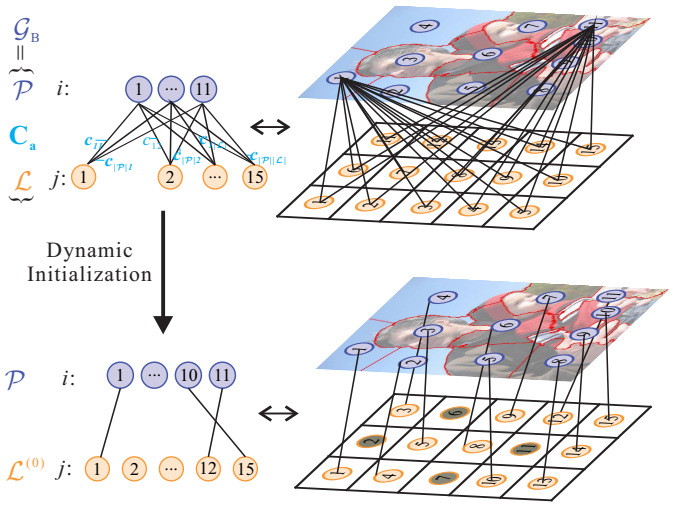


Fig. 3: Illustration of the bipartite graph $\mathcal{G}_B$.

columns/rows in $\mathcal{L}^*$, while strictly maintaining the order of SP nodes in ρ:

$$\mathrm{S}(k,n) = \max_{k\le p\le n+1}[\mathrm{S}(k-1,p-1)+\mathrm{inc}(p,k,n)]. \tag{10}$$

$$\mathrm{B_S}(k,n) = \arg\max_{k\le p\le n+1}[\mathrm{S}(k-1,p-1)+\mathrm{inc}(p,k,n)]. \tag{11}$$

$\mathrm{S}(k,n)$ is the minimum overall grid discrepancy of related subgraph, asserting minimum increments caused by allocating k dummy nodes in the first n SP nodes of current column/row ρ. The subgraph corresponding to such change is composed of all nodes in the first $n+k$ rows/columns of $\mathcal{L}^*$ whose states (*i.e.* being either SP node or dummy node) jointly contribute to decreasing $\mathrm{Gtd}(\mathcal{L}^*)$. That is, $\mathrm{S}(k,n)$ is actually the change of $\mathrm{Gtd}(\mathcal{L}^*)$. $\mathrm{inc}(p,k,n)$ denotes extra discrepancy increments by assigning #k dummy node right in front of #p real node of ρ and is calculated as

$$\mathrm{inc}(p,k,n) = \sum_{i=p}^{n} \mathrm{Gtd}(i,i_r) + \mathrm{Gtd}(i,i_l) + \mathrm{Gtd}(k_l,k_r), \tag{12}$$

where i_l (k_l) and i_r (k_r) indicate left-first and right-first real nodes of #i (#k) real node.

4.3.3 The algorithm

Given a SP-graph $\mathcal{G}$ whose construction is discussed in subsection 5.1, we get the initial grid $\mathcal{L}^{(0)}$ from $\mathcal{G}$ via bipartite graph matching introduced in subsection 4.3.1. Then, we get an optimized grid $\mathcal{L}^*$ via minimizing Eq. (8) with dynamic programming procedure to refine the dummy nodes location of $\mathcal{L}^{(0)}$ over all columns, as described in subsection 4.3.2. The algorithm is illustrated in Algorithm. 1.

Algorithm 1 SP-grid via hybrid DP

Require: The superpixel graph $\mathcal{G}$
Ensure: Optimal minimum topological discrepancy grid $\mathcal{L}^*$
Compute the edge weight matrix W according to Eq. (2);
Compute the grid positional discrepancy $P(\mathcal{L})$ and the structural discrepancy $S(\mathcal{L})$ according Eq. (10) and Eq. (11) respectively;
Construct the bipartite graph $\mathcal{G}_B$ and solve the assignment problem to get the initial grid $\mathcal{L}^{(0)}$
repeat
  for each column ρ of $\mathcal{G}$ **do**
    /* Stage 1: compute $\mathrm{coh}(p,k,n)$ of current column */
    for $n=1$ to $|\rho|$, $k=1$ to m, $p=k$ to $n+1$ **do**
      Compute $\mathrm{inc}(p,k,n)$ as the overall coherence of its correlated subgraph;
    end for
    /* Stage 2: compute $\mathrm{S}(k,n)$ and $\mathrm{B_S}(k,n)$ */
    for $n=1$ to $|\rho|$, $k=1$ to m **do**
      Compute $\mathrm{S}(k,n)$ and $\mathrm{B_S}(k,n)$ using Eqs. (10)–(11);
    end for
    /* Stage 3: updating $\mathcal{G}^*$ */
    Updating $\mathcal{G}^*$ according to $\mathrm{B_S}(\cdot,\cdot)$;
  end for
until No changing to $\mathcal{G}^*$;

The complexity of Hungarian algorithm during initialization is $\mathrm{O}(|\mathcal{P}|^3)$. The dynamic programming process of optimizing column ρ is $\mathrm{O}(|\rho|^3 m)$, where $|\rho|$ is the number of real nodes and m is the number of dummy nodes to be added in the column.

5 Fast Object Localization

With the proposed MTD, we thus can transform the grid-based acceleration methods from pixel-level to SP-level to get efficient yet approximate segmentation results for various applications. Here, we focus on the object localization, *i.e.* given an object mask in an image, we need to automatically localize the object in another image with possible huge deformation, scale variation, point view change and so on. We first present four SP-graphs as the input

of MTD for arbitrary SP segmentation results. Then, we introduce how to use the MTD SP-grid to realize fast yet accuracy object localization with the original pixel-level method, *i.e.* region covariance (RC) detection [1].

5.1 SP-graph construction

For our approach, we have constructed four types of SP-graphs based on the given SP set $\mathcal{P}$ and the edge weight matrix $W_{\mathcal{P}}$:

- $k$nn. If p is the one of k nearest neighbor of q (according to dif(p,q)) and vice verse , then link them.
- n-con. Link the spatially n-nearest SPs. Formally, for SP p and q in $\mathcal{P}$, the adjoint matrix $L = (l_{pq})_{|\mathcal{P}|\times|\mathcal{P}|}$ is defined as

$$l_{pq} = \begin{cases} 1; & \exists x \in \Omega(I), \{p,q\} \subset \mathcal{P}_{N_x} \\ 0; & others \end{cases}, \quad (13)$$

  where N_x is the 4-connected neighborhood of pixel x and $\mathcal{P}_{N_x}$ is the superpixel label set of N_x. We regard p and q is connected in n-con if and only if $L^n(p,q) \neq 0$;
- ϵnn. Link p and q if dif$(p,q) < \epsilon$;
- L_1 graph. This type of graph is described in [34] and here we introduce it shortly. For SP p in $\mathcal{P}$, we calculate a color histogram with 8 bins per RGB channel h_p. For each p, we solve the following ℓ_1 norm minimization to get a sparse representation h_p:

$$\min_{\alpha_p} ||\alpha_p||_1, \qquad \text{s.t.} \qquad h_p = B_p \alpha_p, \quad (14)$$

  where $B_p = [h_1, \ldots, h_{p-1}, h_{p+1}, \ldots, h_{\mathcal{P}}, E] \in \mathbb{R}^{24\times(24+|\mathcal{P}|-1)}$ and E is the identity matrix. The graph is constructed based on the adjacent matrix

$$L(p,q) = \begin{cases} 1; & \alpha_p(q) \neq 0 \ or \ \alpha_p(q-1) \neq 0 \\ 0; & others \end{cases}.$$

We quantize the first 3 graphs into 5 levels, namely $k = 1, k = 5, k = 10, k = 15, k = 25; n = 1, n = 2, n = 3, n = 4, n = 6$ and $\epsilon = 100, \epsilon = 200, \epsilon = 300, \epsilon = 400, \epsilon = 500$ [2].

2. dif(p,q) is regularized to [0, 1000]

5.2 SP-grid based object localization

We extend pixel level region covariance (RC) detection to SP level. Following the notation of [1], given the input image I, let $F \subset \mathbb{R}^{W\times H\times d}$ be the corresponding feature image:

$$F(x,y) = \phi(I,x,y), \quad (15)$$

where ϕ can be any mapping of intensity, color, gradients, filter responses or feature statistics. The key of RC-based object localization is to efficiently compute $F(\mathcal{R})$ for any rectangle $\mathcal{R}$ of I using the integral-image acceleration [1]. Similarly, due to the regular structure of our SP-grid, for any *logic rectangle* $\mathcal{R}$ on the SP-grid, we can directly use the integral-image technique to compute the corresponding RC feature of $\mathcal{R}$ in $\mathrm{O}(1)$ time. We need only to define the superpixel-level feature image $F(u,v)$ for an SP-grid $\mathcal{G} = \langle \mathcal{C}, \mathcal{E} \rangle$, where $\mathcal{C} = \mathcal{P} \cup \mathcal{D}$ is the vertices set of the grid, $\mathcal{P}$ is the set of real nodes (*i.e.* all SPs) and $\mathcal{D}$ represents all dummy nodes in the grid. The $\#(u,v)$ entry of superpixel-level feature image $F(u,v)$ is defined as the sum of all features of the corresponding image regions:

$$\mathcal{F}(u,v) = \begin{cases} \sum\limits_{(x,y)\in\mathcal{V}_{uv}} F(x,y), & \mathcal{C}_{uv} \in \mathcal{P}, \\ 0, & \mathcal{C}_{uv} \in \mathcal{D}, \end{cases} \quad (16)$$

where $\mathcal{V}_{uv}$ denotes the corresponding image region of $\#(u,v)$ superpixel in the SP-Grid, if it is a real node. Clearly, the superpixel-level feature image $F \subset \mathcal{R}^{c\times r\times d}$, where c and r denote the width and height of the SP-grid $\mathcal{G}$, respectively. In our experiments, we used the Lab color, spatial location, and color gradients as the feature.

6 Experimental Results

We validate the effective and efficiency of our MTD SP-grid method via object localization experiments. To get the most sophisticated analysis and the fairest comparison, we first propose to use differential evolution (DE) [35] to get the best and fixed parameters for all methods. Then, we demonstrate that the topological discrepancy do decrease during the optimization process, which also leads to the best accuracy localization result. We also compare the localization performance considering different

graph construction methods and show that ϵnn based SP-graph can achieve best performance. Finally, we compare and analyse the localization performance based on pixel-level, SP-level and our MTD SP-grid, respectively.

6.1 Experiment setting

We construct our MTD SP-grid based on the SLIC [19] and EGS [23] and denote the corresponding object localization methods as MTD-SLIC and MTD-EGS. We choose these two methods mainly due to its simplicity and efficiency, and the MTD can be readily applied to arbitrary SPs. Besides, with different SP-graphs introduced in subsection 5.1, we can get many variants of MTD-SLIC and MTD-EGS, as discussed in subsection 6.4.

Our test data mainly comes from MFC dataset [36] containing 14 scenes each of which has 15 to 20 images with huge foreground changes including deformation, scale variation, rotation and so on. Besides, to further evaluate our methods, we prepare 3 extra scenes whose images are all grabbed from the Flickr website to form an extended-MFC dateset. The total number of images in extended-MFC are 341. For a single foreground object in one image, we search the same foreground in other images of the same scene. Since one image may contain multiple foreground objects, the actual number of query cases are around 2600.

We develop five object localization methods as baselines based on state-of-the-art representations: pixel-level [1], near-regular SPs, *e.g*. TurboPixel [27], and the regular SPs including SEEDS [18], SPLattice [28] and MCG [21] that also derives two variants with different SP methods, *i.e*. MCG-EGS and MCG-SLIC. Specifically, as introduced in section 5, we can directly use the RC [1] descriptor for the regular SPs, *i.e*. SEEDS, SPLattice and MCG, to localize the object. For near-regular SPs, *i.e*. TurboPixel [27], we use its initial uniformly placed seeds indices to approximate their virtual grid indices and realize object localization with the same RC descriptor used above. Note that, strictly speaking, near-regular SPs cannot support the RC-based searching if without the approximation.

TABLE 1: The parameters of MTD-SLIC&EGS.

Parameter		Symbol	Description
Generating SP	SLIC	c	Superpixel smoothness factor
		n_s	Average superpixel number
	EGS	σ	Smooth the input image
		k	Value for the threshold function
		m_s	Minimum component size
Measuring dissimilarity between SPs		ω_d	Weight for spatial distance
		ω_f	Weight for appearance dissimilarity
		ω_e	Weight for average intensity of edges
Superpixel gridization		β	Weight for distance between regional centroid and grid cell Eq. (4)
		α	Weight for structural discrepancy Eq. (8)

We evaluate all methods with the pixel-level segmentation ground truth instead of bounding box, which is more accurate and fairer for both comparison and analysis. We use $F1$-measure, precision and recall as the metrics being commonly used in object localization.

6.2 Parameters tuning

Since each method have several parameters that need to be set, *e.g*. Tab. 1 which summarizes the parameters of MTD-SLIC and MTD-EGS, it is clearly unreasonable and unfair to analyse or compare different methods with the empirical selected parameters. To alleviate this problem, we propose to use the differential evolution (DE) [35] to tune the parameters of all methods to get the most sophisticated analysis and the fairest comparison. Specifically, we first randomly select 538 cases from 2600 query cases to tune the parameters of each method to get the best performance. Then, the parameters are fixed to evaluate the rest cases. To make the result comparable, we assume the population size and generation number (the two parameters of DE) are the same and empirically set to 150 and 80 for all training. In our experiments, the most methods get convergent within 70 generations. As the limitation of space, we just show the tuning result of our MTD-SLIC and MTD-EGS for different graph type in Table **??** and Table **??**.

6.3 Validation and analysis

As we apply dynamic programming to optimize the topological discrepancy, the MTD method is theocratically converged. Fig. 6

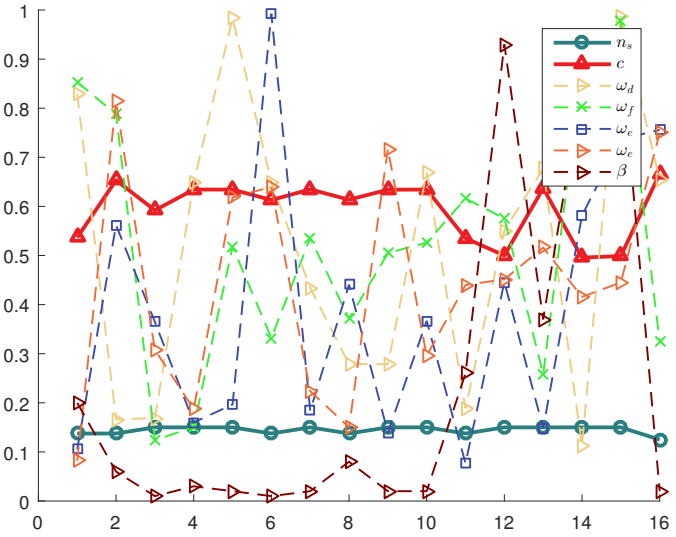


Fig. 4: Best parameters for SLIC.

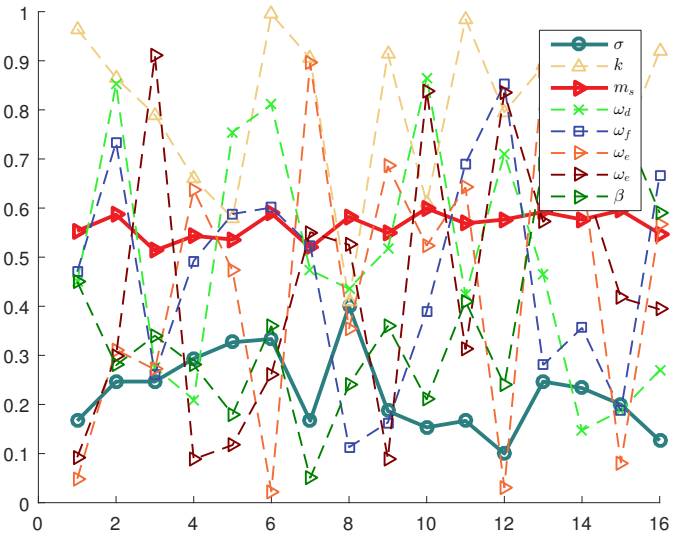
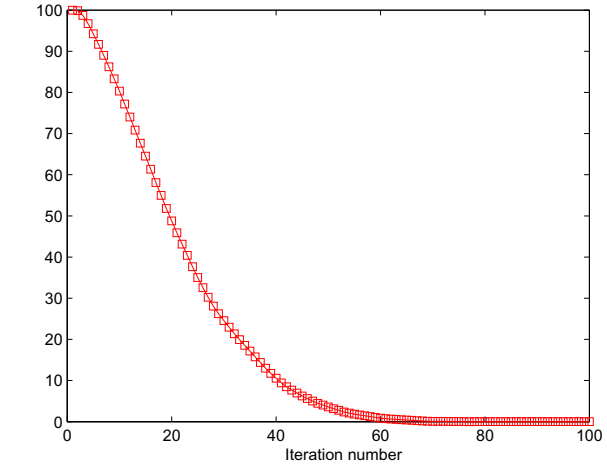
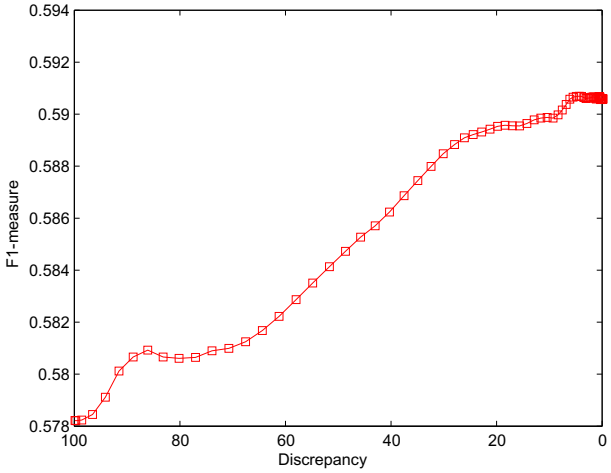


Fig. 5: Best parameters for EGS.



(a)



(b)

Fig. 6: MTD validation. (a) Topological discrepancy converging process; (b) Inverse relationship between discrepancy and F_1-measure.

(a) demonstrates this convergence property in practice. To be a general analysis, we randomly select 10 images, and record all the intermediate discrepancy values during the minimization. As the iteration number and discrepancy values may differ largely for different images, we normalize both values of each image into $[0, 100]$. The average of normalized discrepancy values are computed and plotted against the normalized iteration number. As shown, the discrepancy value is monotonically decreasing versus the iteration number. What's more, the discrepancy value drops rapidly in the first 40 iterations, and almost converges at the 60-th iteration.

In this experiment, we exploit the relationship between the discrepancy value (Eq. (8)) and the F_1- measure in object localization. Specifically, during the MTD minimization, we perform object localization on each intermediate SP-grid, and measures the corresponding discrepancy value and F_1-measure. Again, the discrepancy values from 10 query cases are normalized, and the average F_1 measure is plotted against the average regularized discrepancy in Fig. 6 (b). The F_1-measure increases rapidly when the discrepancy value decreases to around 30 (regularized value), and improves slightly until the discrepancy value converges.

6.4 Comparison of MTD variants

In this experiment, we evaluate the localization performance of variants of MTD-SLIC and MTD-EGS with different SP-graphs. As introduced in subsection 5.1, we can build eight variants via four SP-graphs, including spatial n-connections (n-con), $k$nn, ϵnn, and L_1-graph based on both MTD-SLIC and MTD-EGS. For each graph type, we quantize the connection parameter into 5 different levels. To be specific, we choose $k \in [1, 20]$ for $k$nn, $n \in [1, 5]$ for n-con, and regularize the similarity into [0, 1000] for ϵnn. Thus, we obtain 32 sub-variants in total, as shown in horizontal axis of Fig. 7.

We evaluate the performance of different sub-variants with random parameters, *i.e*. Fig. 7 and optimal parameters, *i.e*. Fig. 8, respectively. For random version, we first randomly generate 1000 parameter groups for both MTD-SLIC (7000 parameters in total) and MTD-EGS (8000 parameters in total) against the parameters in Tab. 1. For each parameter group, we obtain the mean of $F1$-measure over all 2600 query cases. Thus, we obtain 1000 results for each sub-variant and calculate the mean and variance of these results which are presented in Fig. 7. For optimal version, as introduced in subsection 6.2, we use DE to get the optimal parameters for each sub-variant and calculate the mean and variance of $F1$-measure over all query cases excluding the ones used to tune parameters. The results are shown in the left of Fig. 8. In Fig. 7 and 8, the average performance of each sub-variant is marked with a red circle, while the rectangle represents the data variance, the red line segment denotes the median value, and the mark "+" represents outlier.

As showed in Fig. 7, for each type of SP-graph with random parameters, when the graph is more sparse, the localization performance is better. This means that using a smaller connection parameter is more favourable. Given the same SP-grid construction method, ϵnn (at the most sparse level) achieves the best performance among the four graphs, followed by $k$nn (k=1), and in the next n-con (n=100) is slightly better than L_1-graph. On the other hand, when the SP-graph type is fixed, we found that the average performance due to EGS superpxiel generation method is consistently better than that due to SLIC, although the variation range increases. Also note that when using EGS, the F_1-measure variation among different connection levels for one graph type is lower than those using SLIC, which means EGS supports more robust performance. What's more, the localization using SLIC suffers much outliers than using EGS.

When parameters are optimized via DE and lead to results in Fig. 8, sub-variants using the same SP-graph have similar performance. Then, comparing with Fig. 7, we conclude that the most sparse SP-graph leads to the closest performance on optimized parameters and random parameters, which implies that more spare graph achieves more robust performance. For different SP methods, although the performance of MTD-EGS and MTD-SLIC is quite close, MTD-EGS still does consistently better than MTD-SLIC on all SP-graphs. Besides, as compared to Fig. 7, n-con (n=5) outperforms ϵnn a little bit, reporting the best performance among the EGS-based variants and the SLIC-based variants.

6.5 Comparison with state-of-the-arts

We now make comparison with state-of-the-art methods with the analysis of metrics, *e.g*. $F1$-measure, precision, recall and timing, discussion of robustness to the change of scale, rotation and image quality as well as the visual comparison for subjective evaluation.

6.5.1 Quantitative measures and timing

We report quantitative measures (including precision, recall and F_1-measure) and timings for different methods in Tab. 2. Our MTD variants are based on EGS SP method. The precision, recall, and F_1-measures are reported as the average value across all the query cases excluding the cases for tuning parameters. We can see that the all SP-level methods have higher precision and $F1$-measure than pixel-level RC, although the latter has a high recall. Our MTD variants outperform SEEDS, TurboPixel, SPLattice, and MCG in terms of all three metrics. In particular, MTD based on graphs, *i.e*.

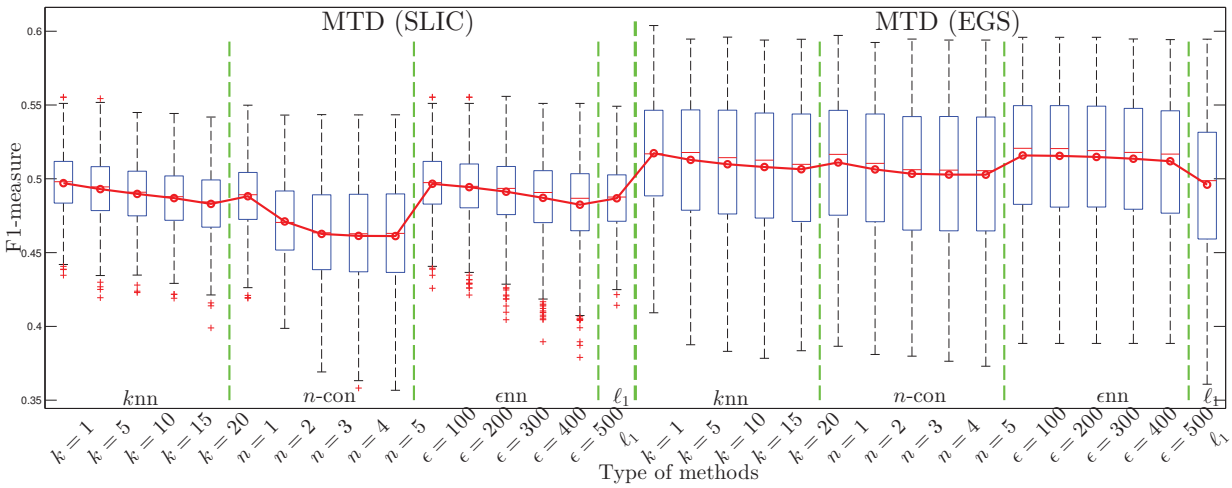


Fig. 7: F_1-measures of 32 sub-variants of MTD with 1000 random parameter groups.

$L1$,$k$nn,n-con achieve the best performance on precision, recall and $F1$-measure, respectively.

The timing performance is measured from three aspects, *i.e.* the superpixel generation and gridization time (SP-Time), object matching time (via brute-force searching), and precomputation time (only invoked in SPLattice). In comparison, although SP level methods have to precalculate SP, the total cost is still less than pixel level method. Besides, our four MTD variants are all efficient, achieving comparable or better timing performance than the state-of-the-art methods. The fastest one is MTD with $k$nn ($k = 3$), taking 0.26 seconds on average.

The right part of Fig. 8 plots the F_1-measure of all object localization methods. As for state-of-the-art methods, our previous M-CG achieves similar performance to SEEDS (around 0.53). It is worthy pointing out that MCG-EGS is slightly better than MCG-SLIC, which is consistent with MTD variants. In comparison, TurboPixel gets a lower average F_1-measure (0.49), followed by SPLattice (0.45 on average). More importantly, our MTD variants outperforms all the above methods with apparent improvements. The maximum F_1-measure is reported to be 0.56 and 0.60 for n-con MTD-SLIC and n-con MTD-EGS respectively.

6.5.2 Robustness

We then evaluate the robustness of all methods to three disturbing factors, *i.e.* noise, scale variation and rotation. For each factor, we consider 10 disturbing levels w.r.t. additive Gaussian noise variance, scale value and rotation degree, respectively, as showed in the horizontal axis of Fig.9. Thus, we can build 30 datasets in total for three factors by transforming all the test cases w.r.t different disturbing levels. Then, we evaluate all methods on each dataset with the DE optimized parameters used in Fig. 8 and calculate the average $F1$-measure, as showed in Fig. 9. Note that, since MTD variants with different graph types have similar performance, we calculate the average $F1$-measure of MTD-SLIC and MTD-EGS across all graphs,respectively.

As demonstrated in Fig. 9, pixel-level RC is highly sensitive to noise and image scaling, and it drops quickly and consistently when the noise variance increases or the scaling factor decreases. TurboPixel, SPLattice, and SEEDS have similar performance when considering image noise. However, TurboPixel and SPLattice are sensitive to image rotation and have low F_1-measure similar to that of pixel-level RC. It is interesting to point out that SEEDS are robust to image scaling and rotation, and it have even improved F_1-measure when the scaling factor drops from 0.78 to 0.56.

In comparison, our MTD methods and previous MCG methods are more stable under varying image scaling factors, rotation, and noise corruptions. Futhermore, MTD-EGS achieves the best performance.

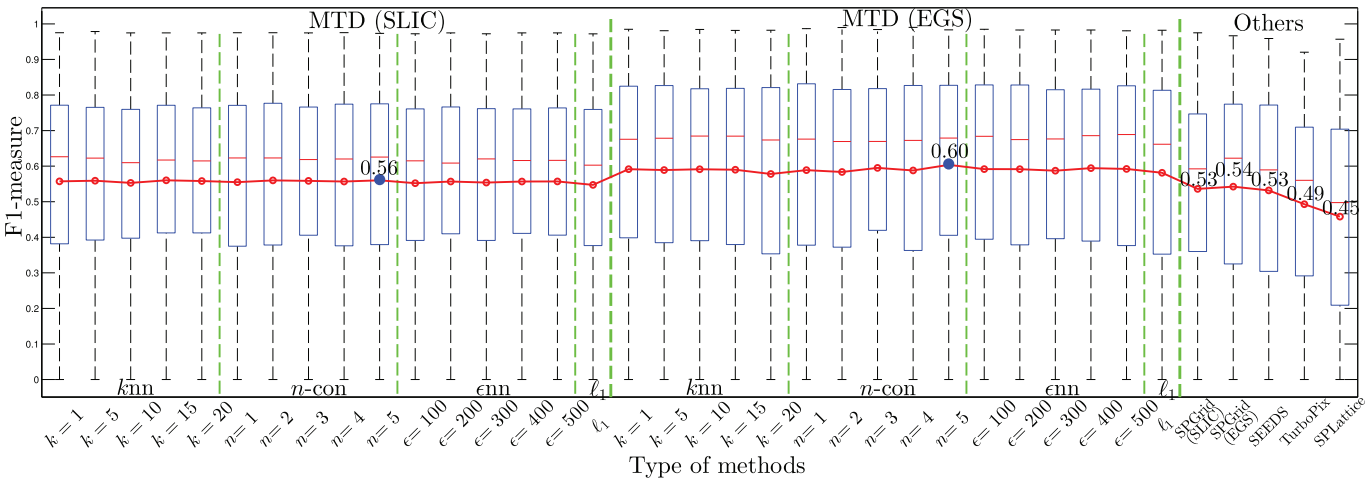


Fig. 8: $F1$-measures of 32 sub-variants of MTD and 5 state-of-the-art methods with DE optimized parameters.

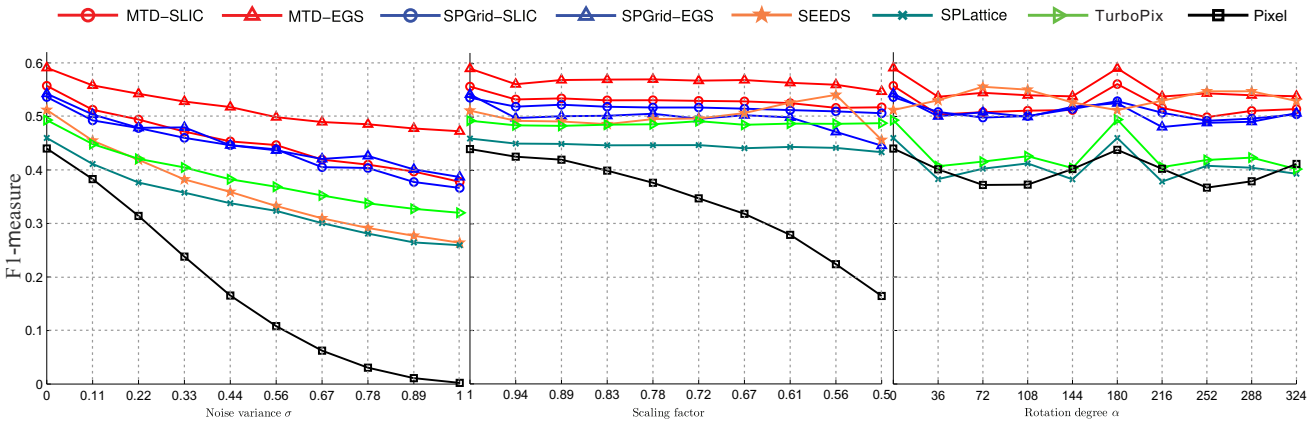


Fig. 9: Robustness to scaling, rotation and noise for pixel-level RC, SPLattic, SEEDS, TurboPixle, MCG and our two MTD variants,*i.e*. MTD-SLIC and MTD-EGS.

6.5.3 More results

The visual comparison among different methods is presented in Fig. 10. Note that, the results come from previous quantitative evaluation in subsection 6.5 with optimal parameters. For our MTD methods, as MTD with different SP-graphs have similar performance, we just show the results of MTD-SLIC and MTD-EGS based on n-con graph. Both the F_1-measure and the whole timing are marked in each result. The leftmost of Fig. 10 shows the inputs of each case, *i.e*. an image with a mask indicating the target object. The rest parts of Fig. 10 shows the retrieval results on another image with different methods.

Pixel-based RC returns rectangular matched regions, which covers only a part of objects under retrieval. SPLattice, TurboPixel, SEEDS have similar localization results in many cases, yet they are not able to well grasp the object boundary. MCG-SLIC and MCG-EGS can capture most of the object boundary with some background included and foreground lost. Obviously, MTD methods (both SLIC-based and EGS-based) improve our previous MCG methods. Let us take the last third and fourth image for example. The man can be identified with a complete boundary using MTD methods, yet miss some parts using MCG. The similar performance is shown in the dog case. We can also see that MTD-EGS gets more complete object boundaries than MTD-SLIC, for instance, the eagle in the last image.

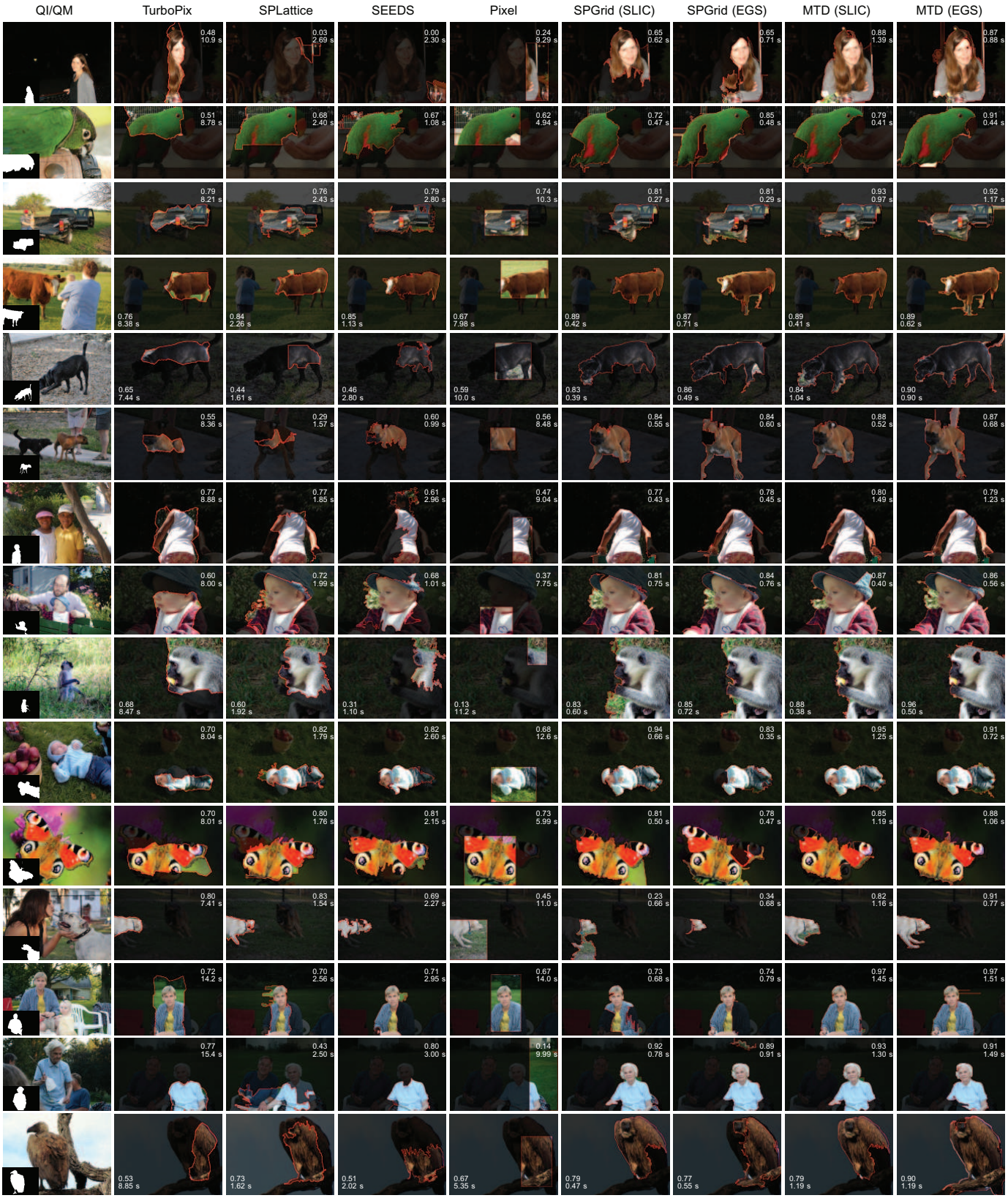


Fig. 10: 15 results of 8 methods. QI and QM represent the query image and query mask. Our MTD-SLIC and MTD-EGS is based on...

TABLE 2: Average accuracy and speed of object localization using region covariance [1] on our extended MFC dataset.

Method	Precision	Recall	F_1	Precom-Time(sec)	SP-Time(sec)	Matching-Time(sec)	Total-Time(sec)
MTD $k$nn 3	0.59	0.60	0.59	-	0.06	0.20	0.26
MTD n-con 5	0.63	0.57	0.60	-	0.07	0.24	0.31
MTD ϵnn 400	0.61	0.58	0.59	-	0.06	0.22	0.28
MTD L_1-graph	0.65	0.55	0.58	-	0.06	0.92	0.98
MCG-SLIC	0.61	0.48	0.53	-	0.16	0.24	0.40
MCG-EGS	0.64	0.50	0.54	-	0.06	0.25	0.31
SEEDS	0.62	0.50	0.53	-	0.10	0.25	0.35
TurboPixel	0.61	0.44	0.49	-	3.58	0.25	3.83
SPLattice	0.57	0.42	0.45	0.17	0.19	0.29	0.48
Pixel	0.40	0.61	0.42	-	-	7.38	7.38

7 Conclusion and Future Work

References

[1] O. Tuzel, F. Porikli, and P. Meer, "Region covariance: A fast descriptor for detection and classification," in *ECCV*, 2006.

[2] C. Lampert, M. Blaschko, and T. Hofmann, "Efficient subwindow search: A branch and bound framework for object localization," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 12, pp. 2129–2142, 2009.

[3] C. Lampert, "Detecting objects in large image collections and videos by efficient subimage retrieval," in *ICCV*, 2009.

[4] S. An, P. Peursum, W. Liu, and S. Venkatesh, "Efficient algorithms for subwindow search in object detection and localization," in *CVPR*, 2009.

[5] F. Porikli, "Integral histogram: A fast way to extract histograms in cartesian spaces," in *CVPR*, 2005.

[6] A. Amini, T. Weymouth, and R. Jain, "Using dynamic programming for solving variational problems in vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 12, no. 9, pp. 855–867, 1990.

[7] P. Felzenszwalb and R. Zabih, "Dynamic programming and graph algorithms in computer vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 4, pp. 721–740, 2011.

[8] Y. Boykov and V. Kolmogorov, "An experimental comparison of min-cut/max-flow algorithms for energy minimization in vision," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 26, no. 9, pp. 1124–1137, 2004.

[9] P. Felzenszwalb and D. Huttenlocher, "Efficient belief propagation for early vision," in *CVPR*, 2004.

[10] P. Carr and R. Hartley, "Minimizing energy functions on 4-connected lattices using elimination," in *ICCV*, 2009.

[11] Y. Lim, K. Jung, and P. Kohli, "Efficient energy minimization for enforcing label statistics," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 36, no. 9, pp. 1893–1899, 2014.

[12] J. Rubio, J. Serrat, A. López, and N. Paragios, "Unsupervised co-segmentation through region matching," in *CVPR*, 2012.

[13] S. Vicente, C. Rother, and V. Kolmogorov, "Object cosegmentation," in *CVPR*, 2011.

[14] J. Tighe and S. Lazebnik, "Superparsing: Scalable nonparametric image parsing with superpixels," in *ECCV*, 2010.

[15] S. Gould, T. Gao, and D. Koller, "Region-based segmentation and object detection," in *NIPS*, 2009.

[16] J. Yan, Y. Yu, X. Zhu, Z. Lei, and S. Z. Li, "Object detection by labeling superpixels," in *CVPR*, 2015, pp. 5107–5116.

[17] T. Brox and J. Malik, "Large displacement optical flow: Descriptor matching in variational motion estimation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 3, pp. 500–513, 2011.

[18] M. V. den Bergh, X. Boix, G. Roig, B. de Capitani, and L. V. Gool, "SEEDS: Superpixels extracted via energy-driven sampling," in *ECCV*, 2012.

[19] 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 Anal. Mach. Intell.*, vol. 34, no. 11, pp. 2274–2282, 2012.

[20] X. Ren and J. Malik, "Learning a classification model for segmentation," in *ICCV*, 2003.

[21] L. Li, W. Feng, L. Wan, and J. Zhang, "Maximum cohesive grid of superpixels for fast object localization," in *CVPR*, 2013.

[22] P. Felzenszwalb and O. Veksler, "Tiered scene labeling with dynamic programming," in *CVPR*, 2010.

[23] P. Felzenszwalb and D. Huttenlocher, "Efficient graph-based image segmentation," *Int. J. Comput. Vision*, vol. 59, no. 2, pp. 167–181, 2004.

[24] A. Iscen, G. Tolias, P. H. Gosselin, and H. Jgou, "A comparison of dense region detectors for image search and fine-grained classification," *IEEE Transactions on Image Processing*, vol. 24, no. 8, pp. 2369–2381, 2015.

[25] H. Fu, X. Cao, D. Tang, Y. Han, and D. Xu, "Regularity preserved superpixels and supervoxels," *IEEE Transactions on Multimedia*, vol. 16, no. 4, pp. 1165–1175, 2014.

[26] P. Neubert and P. Protzel, "Beyond holistic descriptors, keypoints, and fixed patches: Multiscale superpixel grids for place recognition in changing environments," *IEEE Robotics and Automation Letters*, vol. 1, no. 1, pp. 484–491, 2016.

[27] A. Levinshtein, A. Stere, K. Kutulakos, D. Fleet, S. Dickinson, and K. Siddiqi, "TurboPixels: Fast superpixels using geometric flows," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 12, pp. 2290–2297, 2009.

[28] A. Moore, S. Prince, J. Warrell, U. Mohammed, and G. Jones, "Superpixel lattices," in *CVPR*, 2008.

[29] A. Moore, S. Prince, and J. Warrell, ""Lattice cut" - constructing superpixels using layer constraints," in *CVPR*, 2010.

[30] Z. Li and J. Chen, "Superpixel segmentation using linear spectral clustering," in *CVPR*, 2015, pp. 1356–1363.

[31] M. Blaschko and C. Lampert, "Learning to localize objects with structured output regression," in *ECCV*, 2008.

[32] A. Gonzalez-Garcia, A. Vezhnevets, and V. Ferrari, "An active search strategy for efficient object class detection," in *CVPR*, 2015.
[33] Z. Zhang and P. H. S. Torr, "Object proposal generation using two-stage cascade svms," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 38, no. 1, pp. 102–115, 2016.
[34] B. Cheng, J. Yang, S. Yan, and T. Huang, "Learning with L1-graph for image analysis," *IEEE Trans. Image Process.*, vol. 19, no. 4, pp. 858–866, 2010.
[35] R. Storn and K. Price, "Differential evolution - a simple and efficient heuristic for global optimization over continuous spaces," *Journal of Global Optimization*, vol. 11, no. 4, pp. 341–359, 1997.
[36] G. Kim and E. Xing, "On multiple foreground cosegmentation," in *CVPR*, 2012.