Convexity Shape Prior for Segmentation

Lena Gorelick[1], Olga Veksler[1], Yuri Boykov[1], and Claudia Nieuwenhuis[2]

[1] University of Western Ontario, Canada
[2] UC Berkeley, USA

Abstract. Convexity is known as an important cue in human vision. We propose shape convexity as a new high-order regularization constraint for binary image segmentation. In the context of discrete optimization, object convexity is represented as a sum of 3-clique potentials penalizing any 1-0-1 configuration on all straight lines. We show that these non-submodular interactions can be efficiently optimized using a trust region approach. While the quadratic number of all 3-cliques is prohibitively high, we designed a dynamic programming technique for evaluating and approximating these cliques in linear time. Our experiments demonstrate general usefulness of the proposed convexity constraint on synthetic and real image segmentation examples. Unlike standard second-order length regularization, our convexity prior is scale invariant, does not have shrinking bias, and is virtually parameter-free.

Keywords: segmentation, convexity shape prior, high-order functionals, trust region, graph cuts.

1 Introduction

Length-based regularization is commonly used for ill-posed segmentation problems, in part because efficient global optimization algorithms are well-known for both discrete and continuous formulations, e.g. [1,2]. Nevertheless, the shrinking bias and the sensitivity to the weight of the length term in the energy are widely recognized as limitations of this form of regularization. These problems motivate active research on optimization of higher-order regularization energies, e.g. curvature [3,4,5,6], which can alleviate the shrinking bias and other issues.

Fig. 1. Segmentation with convexity shape prior: (a) input image, (b) user scribbles, (c) segmentation with contrast sensitive length regularization. We optimized the weight of length with respect to ground truth. (d) segmentation with convexity shape prior.

D. Fleet et al. (Eds.): ECCV 2014, Part V, LNCS 8693, pp. 675–690, 2014.

We propose a new higher-order regularization model: convexity shape constraint, see Fig.1. Convexity was identified as an important cue in human vision [7,8]. Many natural images have convex or nearly convex objects. Convex objects are also common in medical images. Yet, to the best of our knowledge, we are the first to introduce a convexity shape prior into discrete segmentation energy.

We develop an energy-based formulation for convexity prior in discrete optimization framework and propose an efficient optimization algorithm for the corresponding non-submodular high-order term $E_{convexity}(S)$. The overall segmentation energy $E(S)$ can combine our convexity prior with user-defined hard-constraints and linear intensity appearance [9], boundary length [1], color separation [10], or any others standard submodular terms $E_{sub}(S)$

$$E(S) = E_{convexity}(S) + E_{sub}(S). \tag{1}$$

Convexity of segment S is expressed as a penalty for all ordered triplet configurations 1-0-1 along any straight line, see Fig.2(c). Similar straight 3-cliques also appear in curvature modeling [6], but they also need 0-1-0 configurations to penalize negative curvature. Moreover, they use only local triplets to evaluate curvature. In contrast, convexity **is not** a local property of the segment boundary. Therefore, we have to penalize 1-0-1 configurations on straight intervals of any length. Consequently, our convexity energy model has a much larger number of cliques. We propose an efficient dynamic programming technique to evaluate and approximate these cliques in the context of trust region optimization [11].

Related Work. Many related shape priors were introduced in the past. Common length-based regularizer [1] penalizes segment perimeter favoring smooth solutions that are closer to circles and, therefore, more convex. However, as shown in our experiments, this prior needs to be carefully balanced with the appearance term as it has a strong shrinking bias. Connectivity regularizer [12,13] does not have shrinking bias but might suffer from connecting thin structure artifacts.

Another related regularizer is the *star shape prior* [14,15], which imposes convexity constraints only along the lines passing through a reference point given by the user: these lines are allowed to enter and exit the object only once. In contrast to our convexity prior, the star shape allows for non-convex objects.

There are also part-based shape priors [16,17,18]. A shape is partitioned into several parts and each part imposes certain constraints on the direction of the boundary with the background. This approach can model some simple convex shapes, e.g. a rectangle, but it can not represent a general convexity prior.

Most related work is in [19], which models the object as an n-sided convex polygon. It is a part-based approach that uses one foreground and n background labels. For an accurate segmentation of an arbitrary convex object, e.g. a circle, a finer discretization (i.e. more background parts) is required, significantly increasing runtime. The larger the object, the worse is the problem. In contrast, we can obtain an arbitrary convex object for any choice of orientation discretization. Moreover,

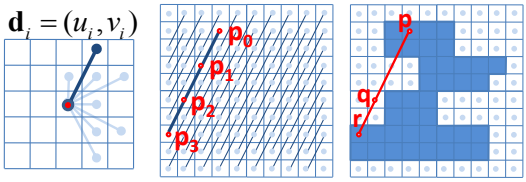


Fig. 2. Left: Example of discretized orientations given by a 5×5 stencil. One orientation d_i is highlighted. Middle: Set L_i of all discrete lines on image grid that are parallel to d_i. Right: Example of a triple clique (p, q, r) that violates convexity constraint.

[19] relies on continuous optimization and is not efficient without GPU. Additional related work on optimization is discussed in Sec. 4.3.

Contributions. We introduce a new discrete convexity shape regularizer. It is virtually parameter free: there is almost no variation in segmentation results for different values of the convexity weight, once the value is high enough, see Sec. 4. This leads to scale invariance, another desirable property. We develop an optimization algorithm based on trust region and show how to use dynamic programming to significantly improve efficiency. Finally, we experimentally validate the advantage of our covexity vs. the length regularizer for segmentation.

The paper is organized as follows. Sec. 2 formulates convexity energy and explains efficient evaluation. Sec. 3 explains trust-region optimization and its efficient implementation. Sec. 4 demonstrates the usefulness of convexity shape prior, and discusses alternative models, optimization schemes, and limitations.

2 Energy

Denote by Ω the set of all image pixels and let $S \subset \Omega$ be a segment. Let $\mathbf{x}$ be a vector of binary indicator variables $x_p \in \{0, 1\}, p \in \Omega$ such that $S = \{p \mid x_p = 1\}$. Due to one-to-one correspondence, we will use either $\mathbf{x}$ or S interchangeably.

In this paper we focus on convexity shape prior and propose a novel formulation to incorporate this prior into segmentation energy. In continuous case, segment S is convex if and only if for any $p, r \in S$ there is no q on a line between them s.t. $q \notin S$. In discrete case, we approximate convexity constrains as follows.

Let $i \in \{1, \ldots, m\}$ enumerate discrete line orientations, see Fig. 2a, and let L_i be the set of all *discrete lines* $l \subset \Omega$ of given orientation d_i such that

$$l = \{p_0, p_1, ..., p_n \mid p_t = p_0 + t \cdot d_i,\ t \in \mathcal{Z}^+,\ p_t \in \Omega\}. \tag{2}$$

Fig. 2b illustrates set L_i for one particular orientation d_i. One way to represent discrete convexity constraint can be based on potential $\phi : \{0, 1\}^3 \to \mathcal{R}$ defined for all triplets of ordered pixels (p, q, r) along any discrete line $l \in \bigcup L_i$

$$\phi(x_p, x_q, x_r) = \begin{cases} \infty & \text{if } (x_p, x_q, x_r) = (1, 0, 1) \\ 0 & \text{otherwise.} \end{cases}$$

In practice we use some finite penalty ω redefining potential ϕ algebraically as

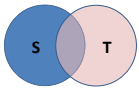
$$\phi(x_p, x_q, x_r) = \omega \cdot x_p(1 - x_q)x_r. \tag{3}$$

The convexity energy $E_{\text{convexity}}(\mathbf{x})$ integrates this triple clique potential over all orientations, all lines and all triplets:

$$E_{\text{convexity}}(\mathbf{x}) = \sum_{l \in \bigcup L_i} \sum_{(p,q,r) \in l} \phi(x_p, x_q, x_r). \tag{4}$$

As discussed below, 3^{rd}-order energy (4) is hard to optimize for two reasons: it is non-submodular and it has a prohibitively large number of cliques.

It is easy to verify that this energy is non-submodular [20]. It is enough to show that there exist segments $S, T \subset \Omega$ such that $E(S) + E(T) < E(S \cap T) + E(S \cup T)$. Consider the example on the right. Since both S and T are convex, the left hand side is zero, while the right hand side is infinite since the union of S and T is not convex. Therefore, our energy cannot be optimized with standard methods for submodular functions.



At the first glance, is seems prohibitively expensive to even evaluate our energy on reasonably sized images. For example, for an image of size 200×300, with just 8 orientations, there are roughly 32 billion triple cliques. In Sec. 2.1, 3.2 we show how to evaluate and approximate the $E_{\text{convexity}}$ in time linear wrt image size using dynamic programming. Then, in Sec. 3 we show how to optimize our energy using trust region techniques [21,11]. Other sparser convexity models and alternative optimization schemes are discussed in Sec. 4.3.

2.1 Energy Evaluation via Dynamic Programming

This section explains how to evaluate our convexity term $E_{convexity}(\mathbf{x})$ efficiently. We show how to compute the inner summation in (4) for one given line l. The idea is to use dynamic programming to efficiently count the number of triplets $(1, 0, 1)$ on a line violating convexity constraints.

Let $\mathbf{x}_l$ denote a vector of binary indicator variables on line l. Let $s, t, v \in \mathcal{Z}^+, s < t < v$ be non-negative integers enumerating the set of pixels on l as in definition (2). With some abuse of notation we denote by x_s, x_t, x_v the corresponding binary variables for pixels $p_s, p_t, p_v \in l$ and rewrite

$$E_{convexity}(\mathbf{x}_l) = \sum_{(p,q,r) \in l} \phi(x_p, x_q, x_r) = \omega \cdot \sum_t \sum_{s<t} \sum_{v>t} x_s \cdot (1 - x_t) \cdot x_v.$$

Consider pixels $p_s, p_t, p_v \in l$. We say pixel p_s *precedes* pixel p_t on line l if $s < t$. Similarly, pixel p_v *succeeds* pixel p_t if $v > t$. Let $C^-(t)$ be the number of pixels p_s preceding pixel p_t such that $x_s = 1$, and $C^+(t)$ be the number of pixels p_v succeeding pixel p_t such that $x_v = 1$:

$$C^-(t) = \textstyle\sum_{s<t} x_s, \quad C^+(t) = \textstyle\sum_{v>t} x_v. \tag{5}$$

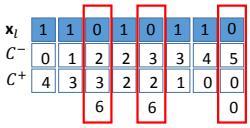


Fig. 3. Evaluation of $E_{convexity}$. The top row shows current configuration $\mathbf{x}_l$ of pixels on line l. The second and the third rows show the number of pixels p_s with $x_s = 1$ before and after each pixel p_t, that is, functions $C^-(t)$ and $C^+(t)$. The last row shows the number of violated constraints for each p_t with $x_t = 0$.

To count the number of all violating configuration $(1, 0, 1)$ for ordered triplets on line l we first consider one fixed pixel $p_t \in l$ with zero label $x_t = 0$. Each preceding pixel with label one and each succeeding pixel with label one form configuration $(1, 0, 1)$. Thus, the total combinatorial number of ordered triplets (p_s, p_t, p_v), $s < t < v$, with configuration $(1, 0, 1)$ is given by $C^+(t) \cdot C^-(t)$, see Fig. 3. Summing over all zero label pixels on line l gives

$$E_{convexity}(\mathbf{x}_l) = \omega \cdot \sum_t C^+(t) \cdot C^-(t) \cdot (1 - x_t).$$

Note that $C^-(t) = C^-(t-1) + x_{t-1}$. Hence both $C^+(t)$ and $C^-(t)$ can be computed for all pixels on a line in one pass using running sums. For a particular orientation d_i, each pixel appears in one line only. Therefore, the total number of operations needed to compute $E_{convexity}(\mathbf{x})$ is $O(mN)$, where $N = |\Omega|$ is the number of pixels in the image and m is the number of distinct orientations.

3 Optimization

This section describes our optimization algorithm for segmentation energy (1) with the convexity shape prior. In terms of indicator variables $\mathbf{x}$ this energy is

$$E(\mathbf{x}) = E_{convexity}(\mathbf{x}) + E_{sub}(\mathbf{x}) \tag{6}$$

where E_{sub} is any submodular term[1] that can be optimized with graph cuts, e.g. boundary length [1], color separation [10], etc. As mentioned earlier, our energy term $E_{convexity}$ is non-submodular. Therefore, (6) is hard to optimize. An additional difficulty is the large number of triple cliques in $E_{convexity}(\mathbf{x})$.

For optimization, we use iterative trust region framework [21], which has been shown promising for various non-submodular energies [11,6,22]. In each iteration, we construct an approximate tractable model $\tilde{E}^k$ of the energy E in (6) near current solution $\mathbf{x}^k$. The model is only accurate within a small region around $\mathbf{x}^k$ called "trust region". The approximate $\tilde{E}^k$ is then optimized within the trust region to obtain a candidate solution. This step is called *trust region sub-problem*. The size of the trust region is adjusted in each iteration based on the quality of the current approximation. See [21] for a review of trust region.

[1] The submodularity of the last term in (6) is assumed for clarity. The proposed trust region approach can approximate non-submodular terms jointly with $E_{convexity}$.

Algorithm 1 summarizes our approach. Line 2 computes unary approximate energy E^k_{approx} for the non-submodular $E_{convexity}$ around $\mathbf{x}^k$. Line 3 combines E^k_{approx} with the submodular E_{sub}. The resulting $\tilde{E}^k$ is submodular and coincides with the exact energy E on $\mathbf{x}^k$. The trust region sub-problem requires minimization of $\tilde{E}^k$ within a small region $||\mathbf{x} - \mathbf{x}^k|| \leq d_k$ around $\mathbf{x}^k$. Unfortunately, minimizing $\tilde{E}^k$ under distance constraints is NP-hard [22]. Instead, we use a simpler formulation of the trust region sub-problem proposed in [22,11] based on unconstrained optimization of submodular Lagrangian

$$L^k(\mathbf{x}) = \tilde{E}^k(\mathbf{x}) + \lambda_k ||\mathbf{x} - \mathbf{x}^k||. \tag{7}$$

Here parameter λ_k controls the trust region size indirectly instead of distance d_k. The distance term in (7) can be expressed using unary terms [11,22]. Therefore $L_k(\mathbf{x})$ is a submodular function. Line 5 solves (7) for some fixed λ_k using one graph-cut. The candidate solution $\mathbf{x}^*$ is accepted whenever the original energy decreases (line 11). The Lagrange multiplier λ_k is adaptively changed (line 13), based on the quality of the current approximation and as motivated by empirical inverse proportionality relation between λ_k and d_k (see discussion in [11]).

Algorithm 1. Trust Region Convexity

1 **Repeat Until Convergence**
2 Compute approximation $E^k_{approx}(\mathbf{x})$ for $E_{convexity}(\mathbf{x})$ around $\mathbf{x}^k$ (see Sec. 3.1)
3 $\tilde{E}_k(\mathbf{x}) = E^k_{approx}(\mathbf{x}) + E_{sub}(\mathbf{x})$ // keep the submodular part unchanged
4 **//Trust region sub-problem: optimize approximate energy**
5 $\mathbf{x}^* \longleftarrow \text{argmin}_{\mathbf{x}} L^k(\mathbf{x})$ (7)
6 **//Update current solution**
7 Evaluate $E_{convexity}(\mathbf{x}^k), E_{convexity}(\mathbf{x}^*)$ (see Sec. 2.1)
8 Evaluate $E^k_{approx}(\mathbf{x}), E^k_{approx}(\mathbf{x}^*)$ (see Sec. 3.2)
9 $R = E(\mathbf{x}^k) - E(\mathbf{x}^*)$ //actual reduction in energy
10 $P = \tilde{E}_k(\mathbf{x}^k) - \tilde{E}^k(\mathbf{x}^*)$ //predicted reduction in energy
11 $\mathbf{x}^{k+1} \longleftarrow \begin{cases} \mathbf{x}^* \text{ if } R/P > \tau_1 \\ \mathbf{x}^k \text{ otherwise} \end{cases}$
12 **//Adjust the trust region**
13 $d_{k+1} \longleftarrow \begin{cases} \lambda_k/\alpha \text{ if } R/P > \tau_2 \\ \lambda_k \cdot \alpha \text{ otherwise} \end{cases}$

3.1 Linear Approximation of $E_{convexity}$

In this section we derive linear approximation $E^k_{approx}(\mathbf{x})$ for the energy term $E_{convexity}(\mathbf{x})$ in (4) around current solution $\mathbf{x}^k$

$$E^k_{approx}(\mathbf{x}) = \sum_{l \in \bigcup L_i} \sum_{(p,q,r) \in l} \phi^k(x_p, x_q, x_r)$$

where $\phi^k(x_p, x_q, x_r)$ is a linear approximation of the corresponding triple clique potential $\phi(x_p, x_q, x_r)$ in (3) around $\mathbf{x}^k$, as explained below.

Property 1. For any potential $\phi(\mathbf{x}) : \{0,1\}^n \rightarrow \mathcal{R}$ of n binary variables $\mathbf{x} = (x_1, \ldots, x_n)$ and any subset $A \subset \{0,1\}^n$ of $n+1$ distinct binary configurations of $\mathbf{x}$, there is a linear function $L_A(\mathbf{x}) = a_0 + a_1 x_1 + a_2 x_2 + \ldots + a_n x_n$ such that $\phi(\mathbf{x}) = L(\mathbf{x})$ for any $\mathbf{x} \in A$.

For example, Fig.2 in [22] shows linear approximations of any pairwise potential ($n = 2$) that are exact for $n + 1 = 3$ out of $2^n = 4$ binary configurations.

Based on Prop. 1, one way to approximate potential $\phi(\mathbf{x})$ in (3) is to compute $n+1$ unknown coefficients of $L_A(\mathbf{x})$ by solving a system of $n+1$ equations $\phi(\mathbf{x}) = L_A(\mathbf{x})$ for $n+1$ binary configurations in some chosen A. We take an alternative practical approach that avoids solving systems of equations and implicitly selects a specific set A. Note, any discrete potential ϕ can be written as a combination of *multilinear* functions of variables $(x_1, \ldots, x_n)$, see (3). In this case, it is easy to verify that Taylor expansion ϕ^k of the potential ϕ around configuration $\mathbf{x}^k$ is a linear function satisfying Prop. 1. That is, $\phi^k(\mathbf{x})$ agrees with $\phi(\mathbf{x})$ on configuration $\mathbf{x}^k$ and n other "neighboring" configurations obtained by flipping one of the variables in $\mathbf{x}^k = (x_1^k, \ldots, x_n^k)$. Omitting the constant terms, Taylor expansion of (3) around $\mathbf{x}^k$ yields[2]:

$$\phi^k(x_p, x_q, x_r) = (1 - x_q^k) \cdot x_r^k \cdot x_p - x_r^k \cdot x_p^k \cdot x_q + (1 - x_q^k) \cdot x_p^k \cdot x_r. \qquad (8)$$

The components in (8) have an intuitive interpretation. Consider the first component $(1 - x_q^k) \cdot x_r^k \cdot x_p$. Recall that pixels p, q, r are on a line and q is between p and r. If the current configuration $\mathbf{x}^k$ is such that $x_q^k = 0$, and $x_r^k = 1$, then assigning label 1 to pixel p violates convexity, assuming q and r keep their labels unchanged from $\mathbf{x}^k$. The unary term $(1-x_q^k)\cdot x_r^k \cdot x_p$ penalizes this violation: assignment $x_p = 1$ carries a penalty, whereas $x_p = 0$ is not penalized. The other two components in (8) have similar intuitive interpretations.

Approximation in (8) gives three unary terms for each triple clique. Consider line l. Pixel p can be either the leftmost, middle, or rightmost member of a clique on that line. Sum the terms from all triple cliques on line l involving pixel p. First with p being on the left, then in the middle and finally on the right of the clique. All these terms contribute to the unary potential for a single pixel p:

$$u_p^l(x_p) = \sum_{(p,q,r)\in l} (1 - x_q^k) \cdot x_r^k \cdot x_p - \sum_{(q,p,r)\in l} x_q^k \cdot x_r^k \cdot x_p + \sum_{(q,r,p)\in l} (1 - x_r^k) \cdot x_q^k \cdot x_p. \qquad (9)$$

The full Taylor based unary term for pixel p sums $u_p^l(x_p)$ over all lines,

$$u_p(x_p) = \textstyle\sum_{l \in \bigcup L_i} u_p^l(x_p) \qquad (10)$$

Fig. 5 illustrates the resulting unary terms arising from such approximation. They encourage any holes in the foreground segment to be filled in, and any protrusions to be erased. Efficient computation of (10) is discussed in Section 3.2.

There is a relation between the Taylor unary terms in (10) and parallel ICM algorithm, noted in [23]. However, our trust region framework has many differences from parallel ICM [23]. See [22] for a detailed comparison.

[2] Here we assume $\omega = 1$. If $\omega \neq 1$, all the derived formulas should be multiplied by ω.

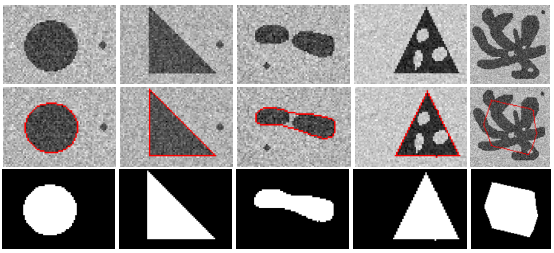


Fig. 4. First row shows synthetic images with added noise $\sigma_{noise} = 0.2$; Second and third rows show contours and masks of segmentation, $\omega = 0.1$. We used log-likelihood appearance terms, $(\mu_{fg} = 0, \sigma_{fg} = 0.1)$ and $(\mu_{bg} = 1, \sigma_{bg} = 0.1)$. The convexity prior removes noise, connects components and fills holes while preserving sharp corners.

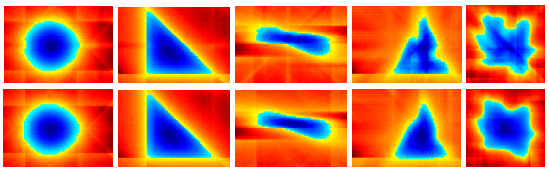


Fig. 5. First and second rows show unary approximation terms of $E_{convexity}$ as in (10) during the first and second iterations of trust region for the examples in Fig. 4. Red-yellow colors denote preference to background, and blue-cyan colors - preference to foreground. Unary terms encourage filling of holes and removal of protrusions.

3.2 Energy Approximation via Dynamic Programming

Naive computation of the summations in (10) is too costly. We now explain how to compute the unary terms in (10) efficiently. Similarly to Sec. 2.1, the speedup is achieved with running sums on each line. In (10), each u_p sums over all lines $l \in \bigcup L_i$. As in Sec. 2.1 we first show how to efficiently compute $u_p^l(x_p)$ in (9).

Let $s, t, v \in \mathcal{Z}^+$ enumerate pixels on l and rewrite (9)

$$u_t(x_t) = x_t \left(\sum_{s>t} \sum_{v>s} (1 - x_s^k) \cdot x_v^k - \sum_{s<t} \sum_{v>t} x_s^k \cdot x_v^k + \sum_{v<t} \sum_{s<v} x_s^k \cdot (1 - x_v^k) \right). \quad (11)$$

In (11), the first sum counts the number of pixel pairs (p_s, p_v) such that $t < s < v$ and $x_s^k = 0, x_v^k = 1$. The second sum counts the number of pixels pairs (p_s, p_v) such that $s < t < v$ and $x_s^k = x_v^k = 1$. The last sum counts the number of pixels pairs (p_s, p_v) such that $s < v < t$ and $x_s^k = 1$, $x_v^k = 0$.

Let C^- and C^+ be as in (5). Recall that each of them can be computed in one pass over the line. Then the second sum in (11) is simply $C^-(t) \cdot C^+(t)$. For the other two sums, we need additional running sums.

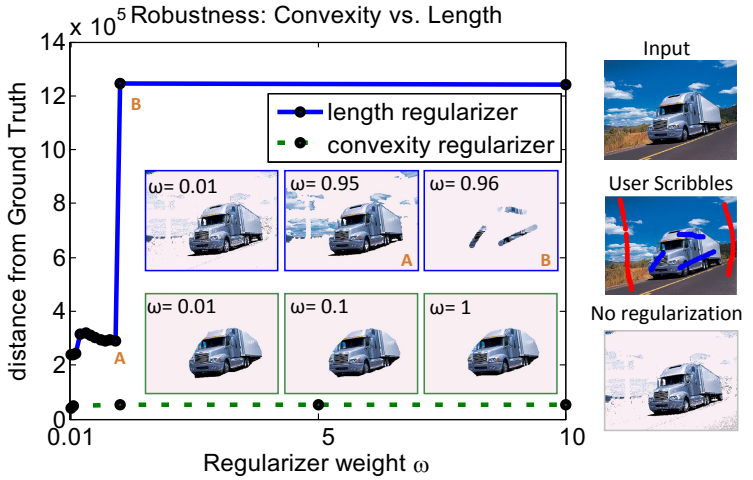


Fig. 6. Illustration of robustness to parameter ω: results for length regularization are shown with blue color and for convexity shape prior - with green. See text for details.

Denote by $A^-(t)$ the number of pixel pairs (p_s, p_v) preceding pixel p_t such that $x_s^k = 1$, $x_v^k = 0$ and pixel p_s precedes pixel p_v,

$$A^-(t) = \sum_{v<t} \sum_{s<v} x_s^k \cdot (1 - x_v^k).$$

Given C^-, we compute A^- in one pass over line l using $A^-(0) = A^-(1) = 0$ and recurrence $A^-(t) = A^-(t-1) + (1 - x_{t-1}^k) \cdot C^-(t-1)$.

Similarly, we define $A^+(t)$ as the number of pixel pairs (p_s, p_v) succeeding pixel p_t such that $x_s^k = 0$, $x_v^k = 1$ and pixel p_v succeeds pixel p_s,

$$A^+(t) = \sum_{s>t} \sum_{v>s} (1 - x_s^k) \cdot x_v^k.$$

A^+ is computed analogously to A^-, given C^+. Then the first sum in (11) is $A^+(t)$ and the third sum is $A^-(t)$, and $u_t(x_t) = A^+(t) - C^-(t) \cdot C^+(t) + A^-(t)$.

Computing A^- and A^+ is linear in the number of pixels on a line. For orientation d_i, each pixel appears on one line only. Therefore we can compute A^- and A^+ for all lines in $O(mN)$ time, where m is the number of orientations and $N = |\Omega|$. Then the unary term for each pixel is computed in $O(m)$ time. Thus the total time to compute Taylor based unary terms for all pixels is $O(mN)$.

4 Experiments

Below we apply convexity shape prior to image segmentation. We discretized orientations using 11×11 stencil yielding 40 orientations for all synthetic images. For natural images we found a 5×5 stencil yielding 8 orientations sufficient. The code is available from `http://vision.csd.uwo.ca/code/`.

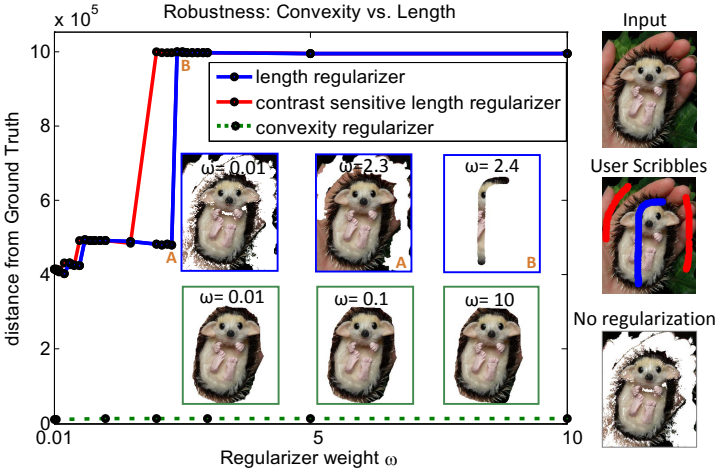


Fig. 7. Illustration of robustness to parameter ω: results for length regularization are shown with blue color and for convexity shape prior - with green. See text for details.

4.1 Synthetic Images

First we validate our method on synthetic images with noise $\mathcal{N}(0, 0.2)$, see Fig. 4. We assume given target appearance distributions for foreground and background, and combine standard log-likelihood data terms with the convexity shape prior

$$E(\mathbf{x}) = E_{app}(\mathbf{x}) + E_{convexity}(\mathbf{x}).$$

Here $E_{app}(\mathbf{x}) = \sum_{p \in \Omega} D_p(x_p)$ is the appearance term, $D_p(x_p) = -\log Pr(I_p|x_p)$ and $E_{convexity}(\mathbf{x})$ is as in (4). Fig. 4 demonstrates that our convexity prior removes noise, insures connectivity and fills in holes while preserving sharp corners.

4.2 Real Images

Next, we use convexity prior in interactive segmentation of natural images with user scribbles. The convexity prior is especially useful for images where there is an overlap between the foreground and background appearance, see Figures 6-9. Such overlap often leads to holes in the foreground or larger parts of the background erroneously segmented as the foreground, see Figures 6-8(bottom-right). The convexity prior prevents such results. Length regularization is either too weak to remove the noise or too strong causing shrinking.

We now specify the details of our interactive segmentation. For appearance we use the recent submodular L_1 *color separation* term proposed in [10]. This term is based on L_1 distance between unnormalized histograms of foreground and background colors. Unlike standard appearance models, the color separation does not require re-estimation of the parameters and can be efficiently and globally optimized. We use 16 bins per color channel and combine the color separation

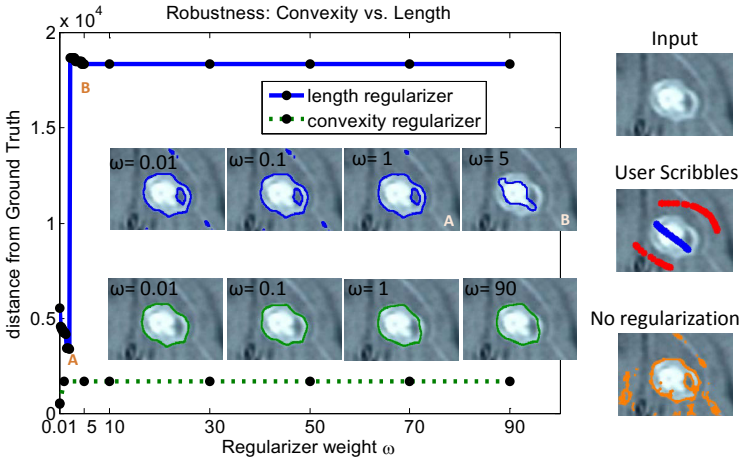


Fig. 8. Illustration of robustness to parameter ω: results for length regularization are shown with blue color and for convexity shape prior - with green. See text for details.

term with the convexity prior, subject to hard constraints on the user scribbles

$$E(\mathbf{x}) = E_{L1}(\mathbf{x}) + E_{convexity}(\mathbf{x}).$$

We then compare with the standard length regularization

$$E(\mathbf{x}) = E_{L1}(\mathbf{x}) + \sum_{(pq)\in\mathcal{N}} \omega[x_p \neq x_q].$$

Figures 6-8 show segmentation results on two natural and one medical image. We vary the weight ω for our convexity prior in (3) and optimize as discussed in Sec. 3. Similarly, we vary the weight ω for the length regularizer above and optimize with one graph-cut [10]. We then plot the distance between the resulting segment and the ground truth as a function of ω (green line - for convexity prior, blue - for length). We also show segmentations for several key points on the plot (green frames - for convexity, blue - for length) and compare them with the results obtained without regularization. The length regularization is either too weak to remove the noise or too strong and has a shrinking bias.

We experiment both with contrast sensitive length (ω depends on pixel pair (p, q) as in [9]) and pure length, see Fig. 7. There is no significant difference in their performance. The same sensitivity to the parameter values is observed; compare the red (contrast sensitive) curve with the blue one (pure length).

Our model is scale invariant due to infinity cost constraints. In practice, we have to choose finite ω. There is almost no variation in segmentation results for different values of ω, once the value is high enough, making it a virtually parameter-free regularizer. In fact, for each image we can compute finite ω such that violating a single constraint is more expensive than the initial solution. In

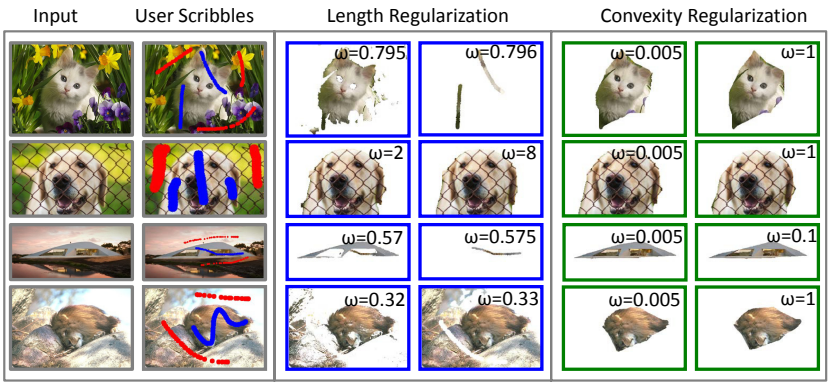


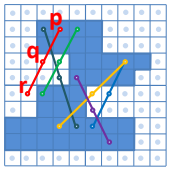
Fig. 9. Additional results comparing between length and convexity regularizers. Except for the second row, all images demonstrate sensitivity to length weight ω.

cases where using such large ω leads to poor local minimum, gradually increasing the weight ω (annealing) can potentially escape to a better solution.

4.3 Alternative Convexity Models and Optimization

Below we discuss some variations of our convexity model and optimization.

Central Cliques Model. One natural question regarding our model is whether we need *all* the triple cliques on a line to enforce convexity. Indeed, it is sufficient to use a smaller subset consisting only of *central* triple cliques (p, q, r), i.e. $|p - q| = |q - r|$, see example on the right. This reduces the number of triple cliques from $O(n^3)$ to $O(n^2)$ for a line with n pixels. However, our dynamic programming procedures for evaluation (Sec. 2.1) and for approximation (Sec. 3.2) are no longer applicable. Brute force computation takes $O(n^2)$ operations per line with n pixels, as opposed to linear time with our dynamic programming for the full model. Nonetheless, we compare between our full model and the central cliques model, see Fig.10. Since the resulting segmentations have no convexity violations their energies can be directly compared. The energy is slightly better with the central cliques, but its running time is 25-30 times longer. The difference in time will be even more significant for larger images.



Alternative Optimization Methods. The most related optimization method is LSA [22], which is also based on trust region framework. However, it was designed for non-submodular energies with only *pairwise* cliques. For this class of energies, LSA reports state-of-the-art results [22]. LSA approximates the energy by replacing non-submodular pairwise cliques with their Taylor expansions while preserving all submodular pairwise cliques. Even though our convexity prior is not pairwise, it is possible to reduce each triple clique to several pairwise po-

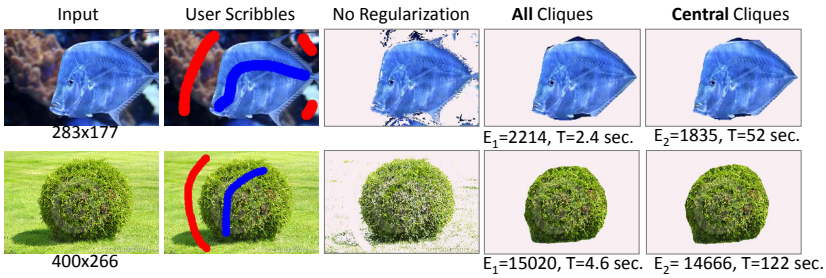


Fig. 10. Comparison between the full convexity model with all triple cliques vs. central cliques convexity model. The later does not allow efficient evaluation and approximation of $E_{convexity}$ using dynamic programming and therefore is much slower.

tentials using an additional auxiliary node [24] and optimize them with LSA. We call this reduced version r-LSA. Reducing all triple cliques would result in a prohibitively large number of auxiliary nodes and pairwise cliques for our full convexity model. Even for the central clique model the number of resulting pairwise cliques is quite large. An $n \times n$ image with m orientations for central cliques produces $O(mn^3)$ pairwise potentials, which is very costly both to optimize and to evaluate. Nonetheless, we tried this approach on a small 91×122 image. The first two rows in Figure 11 compare r-LSA approach to our method. We apply both methods to the central clique model[3] and vary ω. Note that r-LSA is an order of magnitude slower than the slow version of our method. As the value of ω increases, r-LSA fails to obtain satisfactory solutions. We believe that there could be serious drawbacks in splitting clique $\phi(x_p, x_q, x_r)$ into individual submodular and supermodular parts and then approximating the supermodular part. One sign of a problem is that there are infinitely many such decompositions and it is not clear which one would give a better approximation.

Our full model with all triple cliques is also prohibitively expensive for standard optimization methods designed for non-submodular energies, such as QPBO [25] and TRWS [26]. However, we can use these methods to optimize the more compact central clique model as well. The last two rows in Figure 11 show segmentation results of QPBO and TRWS for several values of ω. For values of ω that are not sufficiently large to enforce convexity, all four methods, QPBO, TRWS, r-LSA and Trust Region, return globally optimum, but useless solutions. However, when ω is large enough, QPBO, TRWS and r-LSA fail to obtain a satisfactory result. Our trust region approach obtains good results in all cases.

4.4 Optimization Limitations

Trust region framework is a local iterative optimization and therefore we can only guarantee a local minimum [11]. Figure 12 demonstrates some sensitivity with

[3] Even though the full model is more efficient for our method, for this experiment we use central cliques to have identical energies for direct comparison.

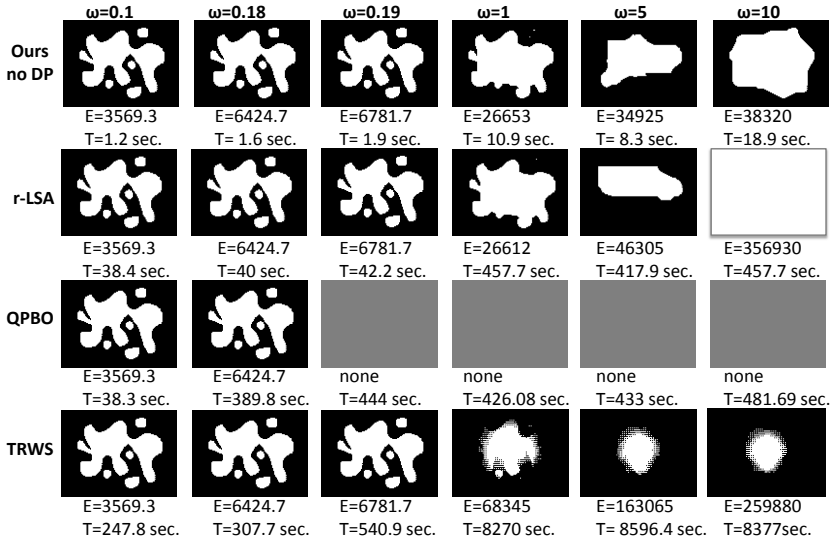


Fig. 11. Comparison between our method without dynamic programming (no DP), r-LSA, QPBO and TRWS on the central clique model with 8 orientations. We use thresholded appearance terms for initialization when needed. As ω increases, making the energy more difficult, QPBO was not able to label any pixel (shown in gray) and TRWS did not converge after 5000 iterations, which took several hours. For ω large enough to enforce convexity, all methods except ours fail.

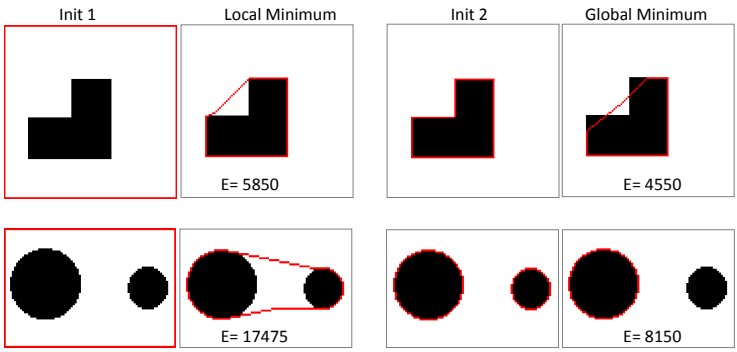


Fig. 12. Local optimization of convexity shape prior might yeild different segmentation results for different initializations. We used 40 orientations, $\omega = 10$ and given target appearance models $(\mu_{fg} = 0, \sigma_{fg} = 0.1)$, $(\mu_{bg} = 1, \sigma_{bg} = 0.1)$.

respect to initialization. A trivial initialization with all pixels in the foreground, denoted by "init 1" and delineated by the red contour, leads to a local minimum. Initializing with the maximum likelihood label per pixel, denoted by "init 2" results in a global optimum, verified by geometrical arguments. Empirically, we obtain better results starting with maximum likelihood labels. This is consistent for all the experiments, both on synthetic and real images.

5 Conclusion and Future Work

We propose convexity prior as a new regularizer and develop efficient discrete optimization based on trust region and dynamic programming. Our regularizer is scale invariant, does not have shrinking bias, and is virtually parameter-free.

In the future, we plan to explore meaningful relaxations of strict convexity. For example, we can explore contrast sensitive convexity, which is similar to contrast sensitive length. The penalty for convexity violation by triplet (p, q, r) can carry a smaller weight if there is a high contrast on a line connecting pixels p and r. This formulation is straightforward, but the main difficulty will be extending our dynamic programming algorithms to handle this case. Another direction is to extend our model to handle objects with multiple convex parts.

References

1. Boykov, Y., Kolmogorov, V.: Computing geodesics and minimal surfaces via graph cuts. In: International Conference on Computer Vision, pp. 26–33 (2003)
2. Pock, T., Cremers, D., Bischof, H., Chambolle, A.: Global solutions of variational models with convex regularization. SIAM Journal on Imaging Sciences 3, 1122–1145 (2010)
3. Schoenemann, T., Kahl, F., Masnou, S., Cremers, D.: A linear framework for region-based image segmentation and inpainting involving curvature penalization. Int. Journal of Computer Vision (2012)
4. Bredies, K., Pock, T., Wirth, B.: Convex relaxation of a class of vertex penalizing functionals. J. Math. Imaging and Vision 47(3), 278–302 (2013)
5. Olsson, C., Ulen, J., Boykov, Y., Kolmogorov, V.: Partial enumeration and curvature regularization. In: International Conference on Computer Vision (ICCV), Sydney, Australia (December 2013)
6. Nieuwenhuis, C., Töppe, E., Gorelick, L., Veksler, O., Boykov, Y.: Efficient regularization of squared curvature. In: IEEE conference on Computer Vision and Pattern Recognition (CVPR), pp. 4098–4105 (June 2014)
7. Liu, Z., Jacobs, D., Basri, R.: The role of convexity in perceptual completion: beyond good continuation. Vision Research 39, 4244–4257 (1999)
8. Mamassian, P., Landy, M.: Observer biases in the 3d interpretation of line drawings. Vision Research 38, 2817–2832 (1998)
9. Boykov, Y., Jolly, M.P.: Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In: IEEE International Conference on Computer Vision, ICCV (2001)
10. Tang, M., Gorelick, L., Veksler, O., Boykov, Y.: Grabcut in one cut. In: International Conference on Computer Vision (2013)

11. Gorelick, L., Schmidt, F.R., Boykov, Y.: Fast trust region for segmentation. In: IEEE conference on Computer Vision and Pattern Recognition (CVPR), Portland, Oregon, pp. 1714–1721 (June 2013)
12. Vicente, S., Kolmogorov, V., Rother, C.: Graph cut based image segmentation with connectivity priors. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR (2008)
13. Nowozin, S., Lampert, C.H.: Global interactions in random field models: A potential function ensuring connectedness. SIAM J. Imaging Sciences 3(4), 1048–1074 (2010)
14. Veksler, O.: Star shape prior for graph-cut image segmentation. In: Forsyth, D., Torr, P., Zisserman, A. (eds.) ECCV 2008, Part III. LNCS, vol. 5304, pp. 454–467. Springer, Heidelberg (2008)
15. Gulshan, V., Rother, C., Criminisi, A., Blake, A., Zisserman, A.: Geodesic star convexity for interactive image segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR (June 2010)
16. Winn, J., Shotton, J.: The layout consistent random field for recognizing and segmenting partially occluded objects. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 37–44 (2006)
17. Liu, X., Veksler, O., Samarabandu, J.: Order-preserving moves for graph-cut based optimization. Transactions on Pattern Analysis and Machine Intelligence (tPAMI) 32, 1182–1196 (2010)
18. Felzenszwalb, P., Veksler, O.: Tiered scene labeling with dynamic programming. In: IEEE Conference on Computer Vision and Pattern Recognition, CVPR (2010)
19. Strekalovskiy, E., Cremers, D.: Generalized ordering constraints for multilabel optimization. In: International Conference on Computer Vision, ICCV (2011)
20. Fujishige, S.: Submodular functions and optimization. Annals of Discrete Mathematics (1991)
21. Yuan, Y.: A review of trust region algorithms for optimization. In: The Fourth International Congress on Industrial & Applied Mathematics, ICIAM (1999)
22. Gorelick, L., Boykov, Y., Veksler, O., Ben Ayed, I., Delong, A.: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 1154–1161 (June 2014)
23. Leordeanu, M., Hebert, M., Sukthankar, R.: An integer projected fixed point method for graph matching and map inference. In: Neural Information Processing Systems (NIPS), pp. 1114–1122 (2009)
24. Kolmogorov, V., Zabih, R.: What energy functions can be minimized via graph cuts? IEEE Trans. Pattern Anal. Mach. Intell. 26(2), 147–159 (2004)
25. Boros, E., Hammer, P.L.: Pseudo-boolean optimization. Discrete Applied Mathematics 123, 2002 (2001)
26. Kolmogorov, V., Schoenemann, T.: Generalized sequential tree-reweighted message passing. arXiv:1205.6352 (2012)