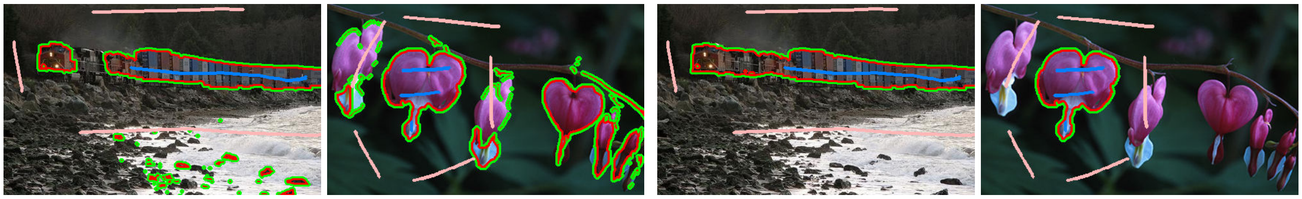
Geodesic Star Convexity for Interactive Image Segmentation

Varun Gulshan[†], Carsten Rother[‡], Antonio Criminisi[‡], Andrew Blake[‡] and Andrew Zisserman[†]

[†]Dep. of Engineering Science
University of Oxford, UK
{varun,az}@robots.ox.ac.uk

[‡]Microsoft Research Ltd.
Cambridge, UK
{carrot,antcrim,ablake}@microsoft.com



(a) No Shape Constraint

(b) Geodesic Star Convexity Constraint

Figure 1. *Shape constraints for Interactive Segmentation.* The blue and pink strokes represent foreground(FG) and background (BG) brushes respectively. The output segmentations are overlaid as a two-colored boundary, with the red boundary towards the FG side and green towards the BG. The segmentations on the left are obtained using Boykov Jolly energy minimization [3]. The segmentations on the right are obtained by minimizing the same energy function subject to Geodesic star-convexity constraints. Notice how the star-convexity constraint helps to remove disconnected FG islands, and also to connect up FG islands into a single component.

Abstract

In this paper we introduce a new shape constraint for interactive image segmentation. It is an extension of Veksler's [25] star-convexity prior, in two ways: from a single star to multiple stars and from Euclidean rays to Geodesic paths. Global minima of the energy function are obtained subject to these new constraints. We also introduce Geodesic Forests, which exploit the structure of shortest paths in implementing the extended constraints. The star-convexity prior is used here in an interactive setting and this is demonstrated in a practical system. The system is evaluated by means of a "robot user" to measure the amount of interaction required in a precise way. We also introduce a new and harder dataset which augments the existing Grabcut dataset [1] with images and ground truth taken from the PASCAL VOC segmentation challenge [7].

1. Introduction

This paper* deals with the problem of interactive image segmentation – the process of power assisting a human user in cutting out a desired object from the image. The main aim of any interactive image segmentation is to improve user experience and the most crucial aspect of improving user experience is the amount of effort taken to segment a desired object. This paper is a step in that direction and also attempts at quantifying the rather subjective and immeasurable quantity of 'effort' required.

Figure 1 demonstrates the power of having shape constraints in an interactive segmentation system. Having shape constraints such as connectivity can restrict the space of possible segmentations to a smaller subset, helping to eliminate false segmentations. This is the main focus of this paper: to introduce shape constraints in interactive segmentation, by means of a powerful extension of the star-convexity prior proposed by Veksler [25].

It is well known that shape is a powerful cue for object recognition. Shape models have been used for object specific segmentation [10, 11] and help in making the problem well posed. In such applications, shapes are represented as some kind of distance transform from a template, and such representations are limited and need good initializations. There has been some work in generic connectivity constraints for segmentation by Vicente *et al.* [26] and Nowozin *et al.* [15]. They show that obtaining globally optimal solutions under the connectivity constraint is NP-hard, and then try to find approximate solutions. Connectivity can also be imposed as a post-processing step as done in paint-select [13], but that is not principled and cannot be expressed as an energy function. Our work is closest to Veksler [25], where a star-convexity constraint is imposed on the segmentation, and globally optimal solutions are

*This work was supported in part by Microsoft Research through the European PhD Scholarship Programme.

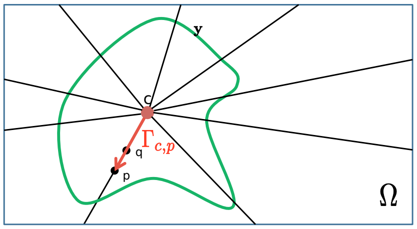


Figure 2. A star-convex object $\mathbf{y}$ (in green) wrt star center c. p is any point in the image domain Ω and $\Gamma_{c,p}$ is the straight line segment connecting c to p. An object is star-convex wrt center c if for every point p in the object, all points on the line segment $\Gamma_{c,p}$ also lie inside the object.

achieved subject to the star convexity constraint. The star convexity constraint also ensures connectivity to brushes, and is a stronger assumption than plain connectivity. In this work, we extend the idea of star-convexity in a way that gets rid of the limitations of Veksler [25], and yet being more meaningful than just plain connectivity [26, 15]. We introduce the idea of Geodesic forests in order to implement our constraints. Note that we still obtain globally optimal solutions while breaking free from Veksler's limitations.

2. Star-convexity

This section reviews the ideas of star-convex sets (Section 2.1) and how this concept is extended to multiple stars (Section 2.2) and Geodesic stars (Section 2.3).

2.1. Single star-convexity

Star-convex sets have been defined in the geometry and math community [24, 21]. Recently Veksler [25] used such sets as a shape prior in image segmentation. A point p is said to be visible to c via a set $\mathbf{y}$ if the line segment joining p to c lies in the set $\mathbf{y}$. A set $\mathbf{y}$ is star-convex with respect to center c, if every point $p \in \mathbf{y}$ is visible to c via $\mathbf{y}$ (ref. Figure 2). Denote by $S^*(\{c\})$, the set of all shapes which are star-convex wrt to center c. The star constraint is expressed as an energy:

$$E^*(\mathbf{y}|c) = \begin{cases} 0 & \text{if } \ \mathbf{y} \in S^*(\{c\}) \\ \infty & \text{if } \ \mathbf{y} \notin S^*(\{c\}) \end{cases} \tag{1}$$

The set $\mathbf{y} \subseteq \Omega$ can also be represented as a function $y : \Omega \to \{0, 1\}$, where Ω is the domain of the image and $\forall p \in \Omega : p \in \mathbf{y} \Leftrightarrow y_p = 1$. The energy $E^*(\mathbf{y}|c)$ can be expressed with pairwise terms. As shown in Fig. 2, for the line segment $\Gamma_{c,p}$ joining p to c, the star constraints are written as:

$$\forall q \in \Gamma_{c,p},\ E^*_{pq}(y_p, y_q) = \begin{cases} \infty & \text{if } y_p = 1 \text{ and } y_q = 0 \\ 0 & \text{otherwise} \end{cases}$$

$$E^*(\mathbf{y}|c) = \sum_{p \in \Omega} \sum_{q \in \Gamma_{c,p}} E^*_{p,q}(y_p, y_q) \tag{2}$$

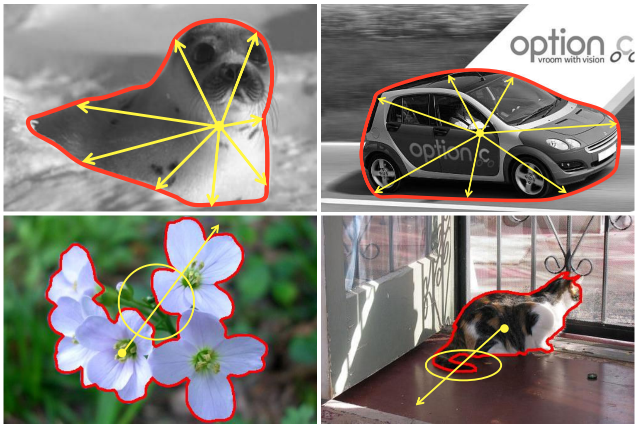


Figure 3. *Star-convex and objects that are not star-convex.* The object of interest is outlined in red. The top row depicts examples of star-convex objects, with a plausible star center marked. The bottom row shows deviations from star-convexity, either in the fine details or gross deviations due to multiple object instances. A plausible star center is shown for such objects with the region that prohibits visibility being encircled.

It is easy to see that such an energy is submodular as the labeling $(y_p = 1, y_q = 0)$ has an ∞ energy, and will always satisfy the submodularity criteria. In practice, the domain Ω is discrete, points p and q correspond to pixels and the paths $\Gamma_{c,p}$ are rasterized versions of continuous lines. As discussed in [25], the constraint need not be implemented $\forall q \in \Gamma_{c,p}$, but needs to be imposed only between 8-connected neighboring pixels (p, q), which makes the constraint efficient to implement. It is also possible to define the constraint directly in the discrete domain (see section 2.3) which avoids rasterizing and other implementation issues discussed in [25].

The star-convexity constraint is quite flexible in the set of shapes it allows as opposed to fixed geometries like ellipses and boxes [20]. It includes the set of all convex shapes containing the point c. Figure 3 shows examples taken from [25] which depict fairly complex objects as being star-convex. The same figure also shows limitations of star-convexity, which limits its applications to real images. The next two sections discuss how to extend the star-convexity constraint to overcome these limitations.

2.2. Multiple stars

A natural extension of single star-convexity is to use multiple stars to define a more general class of shapes. This section explores two alternative ways of defining multiple star-convexity and explains why the second way is more practical than the first. Consider for simplicity star-convexity wrt to two star centers $\{c_1, c_2\}$. The first definition is an extension of the visibility argument presented in Section 2.1 – A set $\mathbf{y}$ is star-convex wrt $\{c_1, c_2\}$ if every point $p \in \mathbf{y}$ is visible to at-least one of the star centers via $\mathbf{y}$ (ref. Fig-

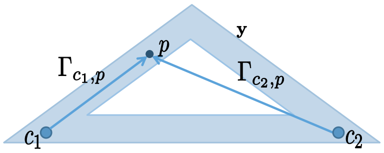


Figure 4. *Multiple star semantics – visibility.* $\mathbf{y}$ (shaded blue) is an object which is star convex wrt. centers $\{c_1, c_2\}$, p denotes an arbitrary point in $\mathbf{y}$ and $\Gamma_{c_1,p}$ and $\Gamma_{c_2,p}$ are the two straight lines joining p to the star centers. Every point $p \in \mathbf{y}$ should be visible to at least one star center for the object to be star-shaped wrt $\{c_1, c_2\}$.

ure 4). Another way of characterizing such a set is by expressing it as a finite union of starshaped sets [23] – A set $\mathbf{y}$ is star-convex wrt $\{c_1, c_2\}$ if $\exists \mathbf{y}_1, \mathbf{y}_2 \subseteq \Omega$ $s.t.$ $\mathbf{y} = \mathbf{y}_1 \cup \mathbf{y}_2$ and $\mathbf{y}_1 \in S^*(\{c_1\})$ and $\mathbf{y}_2 \in S^*(\{c_2\})$. This definition directly translates to an implementation as:

$$E^*(\mathbf{y}|\{c_1,c_2\}) = \min_{\mathbf{y}_1,\mathbf{y}_2} E^*(\mathbf{y}_1|c_1)+E^*(\mathbf{y}_2|c_2)+\delta_{\mathbf{y}}(\mathbf{y}_1\cup\mathbf{y}_2)$$

$$\delta_{\mathbf{y}}(\mathbf{y}_1\cup\mathbf{y}_2) = \begin{cases}0 & \text{if } \mathbf{y}=\mathbf{y}_1\cup\mathbf{y}_2\\ \infty & \text{otherwise}\end{cases} \tag{3}$$

where $E^*(\mathbf{y}_1|c_1)$ and $E^*(\mathbf{y}_2|c_2)$ are as defined in (2). It is certainly possible to express this energy with pairwise terms and additional variables $\mathbf{y}_1$ and $\mathbf{y}_2$ but the energy is not submodular ($\delta_{\mathbf{y}}(\mathbf{y}_1 \cup \mathbf{y}_2)$ causes the submodularity to break). This is not surprising as these visibility semantics are closely related to the classical NP-complete "Art Gallery Problem" – the problem of placing the minimum number of guards which together can observe the whole gallery [16]. Also it is not obvious how to extend these semantics from a finite discrete set of star centers to an infinite continuous set of star centers – e.g a brush stroke.

The first definition of star-convexity for multiple centers, above, is arguably a natural extension. The second definition, described next, is both computationally more tractable and also extends naturally to an infinite set of star centers. The definition of the line segment joining star center c to p (denoted $\Gamma_{c,p}$) is extended to the line segment joining the set of star centers $\mathbf{c}$ to point p (denoted $\Gamma_{\mathbf{c},p}$). Observing that $\Gamma_{c,p}$ is the shortest path between point p and center c, we define $\Gamma_{\mathbf{c},p}$ as the shortest path between point p and set $\mathbf{c}$:

$$c(p) = \arg\min_{c\in\mathbf{c}} d(c,p) \quad , \quad \Gamma_{\mathbf{c},p} = \Gamma_{c(p),p} \tag{4}$$

where $d(c, p)$ is the Euclidean distance between c and p and $c(p)$ denotes the closest star center to point p. Figure 5 visualizes these shortest paths for the case of discrete and continuous star centers. This construction is a restriction of the previous visibility semantics – here each point $p \in \mathbf{y}$ should be visible to its nearest star center $c(p)$. The star energy $E^*(\mathbf{y}|\mathbf{c})$ can now be written exactly as in (2) with the shortest paths now defined as in (4). Note that the star energy remains submodular which keeps things tractable as explained in section 4. Also these semantics extend nicely to a having brush strokes as star centers. To our knowledge, such a definition of star-convexity has not yet been proposed in the literature.

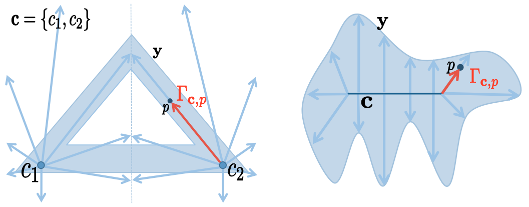


Figure 5. *Alternate multiple star semantics – visibility to nearest center.* $\mathbf{c}$ denotes the set of star centers, $\mathbf{c} = \{c_1, c_2\}$ in the triangle on the left, and $\mathbf{c}$ is a line segment in the shape on the right. For every point $p \in \mathbf{y}$, star-convexity is imposed on $\Gamma_{\mathbf{c},p}$ – the segment joining p to the nearest star center.

2.3. Geodesic stars

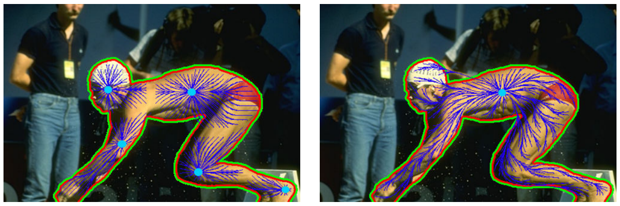
The previous section defined $\Gamma_{\mathbf{c},p}$ as the shortest path between star centers $\mathbf{c}$ and point p. In this section we generalize the notion of shortest path from Euclidean to geodesic, and also define this directly in the discrete domain. Geodesic paths can bend and adapt to image data as opposed to straight Euclidean rays, thus extending visibility and reducing the number of star centers required. In the case of image segmentation the gradients in the underlying image provide information to compute such paths (ref. figure 6). To define the geodesic distance, we first define the length of a discrete path:

$$L(\Gamma) = \sum_{i=1}^{n-1} \sqrt{(1-\gamma_g) d(\Gamma^i, \Gamma^{i+1})^2 + \gamma_g \|\nabla I(\Gamma^i))\|^2} \tag{5}$$

where Γ is an arbitrary parametrized discrete path with n pixels given by $\{\Gamma^1, \Gamma^2, \cdots, \Gamma^n\}$, $d(\Gamma^i, \Gamma^{i+1})$ is the Euclidean distance between successive pixels, and the quantity $\|\nabla I(\Gamma^i))\|^2$ is a finite difference approximation of the image gradient between the points (Γ^i, Γ^{i+1}). The parameter γ_g weights the Euclidean distance with the geodesic length. It is also possible to use gradients in the likelihood image to get rid of texture edges as done in [2]. Using the above definition, one can define the geodesic distance as in [4]:

$$d_g(a,b) = \min_{\Gamma\in\mathcal{P}_{a,b}} L(\Gamma) \quad , \quad \Gamma_{a,b} = \arg\min_{\Gamma\in\mathcal{P}_{a,b}} L(\Gamma) \tag{6}$$

where $\mathcal{P}_{a,b}$ denotes the set of all discrete paths between two grid points a and b. Note that unlike [2], we are interested in the geodesic path $\Gamma_{a,b}$ rather than geodesic distance $d_g(a, b)$. The above definition of geodesic distance between two points also extends to distance between a set of points $\mathbf{c}$ and a point p exactly as in (4). Notice that everything is now defined in the discrete domain, including the shortest paths $\Gamma_{a,b}$. Thus there is no need of any rasterization step while setting up the star energy, this saves computation time. If one were to connect up every point $p \in \Omega$ to the star center



(a) Euclidean rays (b) Geodesic paths

Figure 6. *Geodesic stars improve visibility.* (a) Shows how the object outlined can be seen by multiple Euclidean star centers (b) Shows how most of the object (except the head) can be seen by just one geodesic star center. Notice how the forest bends around image gradients.

$\mathbf{c}$ using the shortest paths $\Gamma_{\mathbf{c},p}$, the structure obtained would be a collection of trees rooted at $\mathbf{c}$, and we call this structure a *Geodesic Forest*. Geodesics are efficiently computed using shortest path algorithms ($O(n \log n)$, with n being the number of pixels). Also note that the star energy $E^*(\mathbf{y}|\mathbf{c})$ now needs to be written as $E^*(\mathbf{y}|\mathbf{x}, \mathbf{c})$ as it now depends on the underlying image. This energy can be again expressed as in (2) with the shortest paths being given by (6).

It is important to emphasize the role of the parameter γ_g, as it controls the weighing between geodesic and Euclidean. Indeed setting $\gamma_g = 0$ results in a Euclidean shape (albeit in discrete domain), and $\gamma_g = 1$ relies purely on image gradients. We also rescale the gradients such that the average gradient is the same as the average Euclidean distance between neighboring pixels – this is necessary to give γ_g a sensible meaning. This parameter is discussed in detail later in section 6. We use the shorthand GSC (Geodesic Star-Convexity) to refer to this shape constraint for $\gamma_g > 0$ and the shorthand ESC (Euclidean Star-Convexity) to refer to the case $\gamma_g = 0$.

3. Visibility experiments

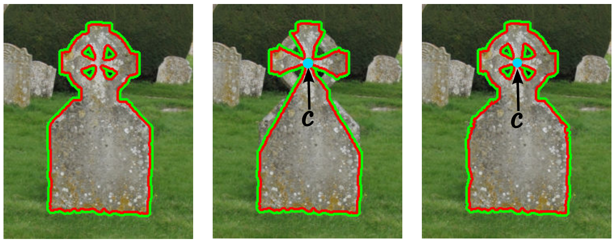
In order to capture the extent of the visibility of an object from a star center, we define $Y^*(\mathbf{y}, \mathbf{c})$, the $\mathbf{c}$-star of set $\mathbf{y}$. It is the set of all points in $\mathbf{y}$ which are visible to $\mathbf{c}$ via $\mathbf{y}$ [24]. Denoting the ground truth object as $\mathbf{y}_{gt}$, the extent of its visibility $Y^*(\mathbf{y}_{gt}, \mathbf{c})$ can be computed as the following optimization:

$$\begin{array}{lll} \min\limits_{\mathbf{y}} \Delta(\mathbf{y}, \mathbf{y}_{gt}) & \Leftrightarrow \min\limits_{\mathbf{y}} \Delta(\mathbf{y}, \mathbf{y}_{gt}) + E^*(\mathbf{y}|\mathbf{x}, \mathbf{c}) & (7) \\ \quad s.t\ \mathbf{y} \in S^*(\mathbf{c}) & \quad s.t\ \mathbf{y} \subseteq \mathbf{y}_{gt} & \\ \qquad \mathbf{y} \subseteq \mathbf{y}_{gt} & & \end{array}$$

where $\Delta(\mathbf{y}, \mathbf{y}_{gt})$ measures the Hamming distance between $\mathbf{y}$ and $\mathbf{y}_{gt}$:

$$\Delta(\mathbf{y}, \mathbf{y}_{gt}) = \sum_{i \in \Omega} (y_i \neq y_i^{gt}) \qquad (8)$$

Note the constraint $\mathbf{y} \subseteq \mathbf{y}_{gt}$ is equivalent to the hard constraint $\forall i : i \notin \mathbf{y}_{gt}, y_i = 0$ and essentially means that one cannot see through the background pixels. The cost function in (7) can be optimized globally with graph cuts, as all the terms are sub-modular. The $\mathbf{c}$-stars for a single star center c are visualized for the case of Euclidean and geodesic star-convexity in Figure 7. A similar cost function (but without the star constraint), was minimized for the purpose of parameter learning in Szummer *et al.* [22].



(a) Ground truth (b) ESC c-star (c) GSC c-star

Figure 7. *c-stars for Euclidean and geodesic star-convexity.* (a) The ground truth object outlined. (b) The Euclidean c-star of the ground truth. (c) The corresponding geodesic c-star with the same star center.

	Percentage Occlusion Rates			
	Full dataset		Complex Shapes	
Method	1 star	2 stars	1 star	2 stars
ESC	4.50±0.58	2.85±0.39	10.95±1.26	6.88±0.89
GSC	4.16±0.54	2.22±0.30	9.48±1.15	4.79±0.66

Table 1. *Visibility experiment.* The occlusion rates denote the percentage of pixels of the object not visible to the star center. Left two columns report performance on the whole dataset and the right two columns on a subset of complex shapes in the dataset. Observing along the rows, GSC has much lower error rates than ESC, suggesting that geodesic improves the visibility of the star centers. We also see that 2 star centers have better visibility than just one. Also note the error rates are higher for complex shapes as one would have expected.

In our quantitative evaluation, we find the $\mathbf{c}$-stars for the case of 1 and 2 star centers on a dataset of images introduced later in section 5.2. The question here is how to choose the star centers $\mathbf{c}$. In our experiments, the star centers were chosen manually to be favorable to the Euclidean case, so as to maximize the visibility of the star center with Euclidean rays. It would also be possible to find the globally optimal location of star centers using Branch-and-mincut [12] by optimizing (7) wrt $\mathbf{c}$ (in addition to $\mathbf{y}$). The occlusion rates for this *Visibility Experiment* are reported in Table 1. We make the following three observations from this table: (i) It is beneficial to have multiple stars as that improves the visibility of the star centers. (ii) Geodesic stars have better visibility than Euclidean stars, and thus lower occlusion rates. (iii) Geodesics improve visibility more for more complex shapes.

One must remember that lower occlusion rates in Table (1) do not automatically mean the shape constraint works better in a practical system. Indeed, having no shape con-

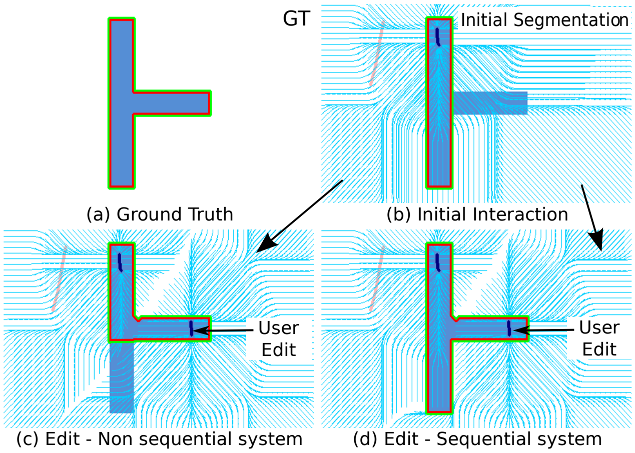


Figure 8. *Sequential system.* (a) Ground truth shape is outlined. (b) User places initial brush strokes, blue for FG and pink for BG. The output segmentation is shown outlined, with the underlying geodesic forest (note that $\gamma_g = 0$ here, so that the forest is actually Euclidean). (c) User makes an edit and the shape constraint changes non-intuitively. The bottom of the T-shape which was visible to the star center on the top now is closer to the newly added star center and not visible to it. (d) The sequential system ensures that visibility is incremental, that things which were visible before do not lose visibility. The geodesic forest is not permitted to enter the previous segmentation, so it bends/terminates around it.

straint, or just a simple connectivity constraint, would give an occlusion rate of zero in this experiment. In practice the shape constraint must be combined with other considerations, as discussed in the next section.

4. Star-convexity in a segmentation system

The star-convexity constraints introduced above are combined with the Boykov Jolly [3] energy formulation as done by Veksler [25]. We first describe the energy minimization of Boykov Jolly:

$$E(\mathbf{y}|\mathbf{x}) = \sum_{i\in\Omega} U(y_i|\mathbf{x}) + \lambda \sum_{(i,j)\in\mathcal{N}} V(y_i, y_j|\mathbf{x}) \qquad (9)$$
$$\bar{\mathbf{y}} = \arg\min_{\mathbf{y}\in\mathcal{Y}} E(\mathbf{y}|\mathbf{x})$$

where $\mathbf{x}$ denotes the image, Ω is the set of all pixels, λ is a weighting on the pairwise terms and $\mathcal{N}$ represents the set of neighboring pixel-pairs. The data term $U(y_i|\mathbf{x})$ is computed using the negative log likelihood of color models learned from the user provided brush strokes, and the pairwise terms are the usual contrast dependent terms $V(y_i, y_j|\mathbf{x}) = \exp(-\beta\|x_i - x_j\|^2)$ [18]. The color models use GMMs with 5 components each for FG/BG. Also as the brush strokes are sparse, the GMMs are mixed with uniform color models as suggested in [5]. In the final system, the energy in (9) is minimized subject to $\mathbf{y}$ being star-convex:

$$\begin{array}{ll} \min_{\mathbf{y}} & E(\mathbf{y}|\mathbf{x}) \\ s.t & \mathbf{y} \in S^*(\mathbf{c}) \end{array} \Leftrightarrow \min_{\mathbf{y}} E(\mathbf{y}|\mathbf{x}) + E^*(\mathbf{y}|\mathbf{x},\mathbf{c}) \quad (10)$$

It is possible to obtain global minima of (10) efficiently as both $E(\mathbf{y}|\mathbf{x})$ and $E^*(\mathbf{y}|\mathbf{x},\mathbf{c})$ are submodular. The star-convexity constraint above restricts the space of shapes. For an $n \times n$ pixel image, the cardinality of the unconstrained output space is 2^{n*n}, whereas with a single Euclidean star, the output space is reduced significantly to $O(n^n)$. A good regularizer would allow the ground truth object while restricting the output space as much as possible.

Finally we consider the choice of star centers $\mathbf{c}$. In an interactive system, the FG brush strokes are a natural choice for star centers, since that avoids the need for any separate interaction to select the centers. In that way, unlike [25], the set of allowed shapes changes as the user adds brush strokes, and this can sometimes cause shape constraints to change un-intuitively – see Fig. 8. To resolve this issue, a sequential system is developed which ensures that the shape constraints change progressively as the user interacts. This is done by enforcing two properties at every edit: (i) The current segmentation should always be valid under the new star-convexity constraint obtained after the user edit and (ii) the shape constraint should only change locally, around the newly placed brush stroke. (i) can be implemented by ensuring that no paths are allowed to enter the FG segmentation, in the geodesic forest computed after the edit. This can be enforced by adding an ∞ cost on edges going from BG to FG in the geodesic computation. (ii) is implemented by only incrementally computing geodesics from the newly placed brush strokes.

5. Quantitative evaluation

We evaluate interactive segmentation systems quantitatively by means of a "robot user" [14]. Existing evaluation of interactive segmentation is either qualitative [2, 9] or is restricted to a fixed set of user interactions with seed points obtained by eroding and dilating the ground truth segmentation [1, 6]. The robot user generates a flexible sequence of user interactions, according to well-defined rules, that model the way in which residual error in segmentation is progressively reduced in an interactive system.

An ideal evaluation system would measure the amount of effort required to segment an image, in a user study. Alternatively, the robot user simulates user interaction by placing brushes automatically. It starts with an initial set of brush-strokes (chosen manually with one stroke for FG and 3 strokes for BG) and computes a segmentation. It then places a circular brush stroke (diameter 17 pixels) in the largest connected component of the segmentation error area, placed at a point farthest from the boundary of the component. The process is repeated up to 20 times, generating a sequence of 20 simulated user strokes. In this way a custom sequence of brush strokes is generated for each algorithm. Refer to the video at `http://www.robots.ox.ac.uk/~vgg/research/iseg/` for a demonstration of the robot

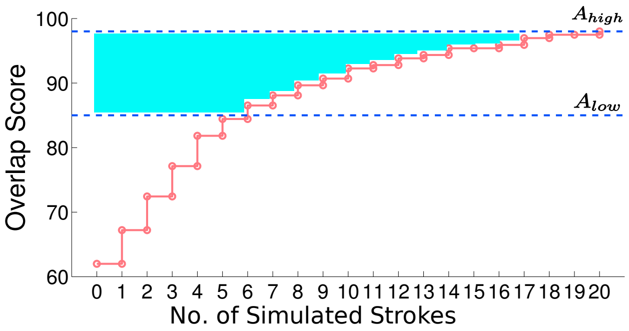


Figure 9. *Plotting overlap score vs. no. of strokes, in order to measure interaction effort.* The area above the curve is a measure of the average number of strokes required for user interaction. Since we are interested in the degree of interaction required to achieve high segmentation accuracy, the average is restricted to the band $[A_{low}, A_{high}]$, as illustrated (shaded in blue).

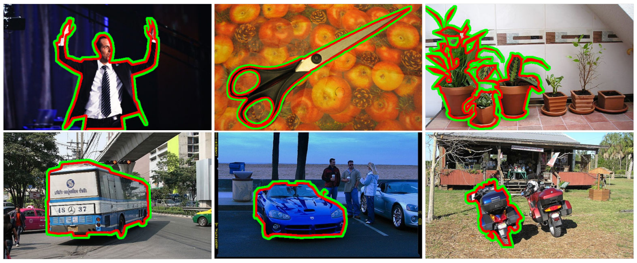


Figure 10. *Dataset.* 6 out of 151 images are shown. The top row shows images with complex shapes; the bottom row has images with relatively simple shapes that can be modeled with just one Euclidean star center.

and [14] for further details.

5.1. Evaluation of interactive segmentation quality

Interactive system quality is evaluated as the average number of strokes required to achieve segmentation quality within a certain band. This is illustrated in figure 9. The graph of overlap score v. no. of brush strokes captures how the accuracy of the segmentation varies with successive user interactions, and the average no of strokes summarizes that in a single score. Here overlap score, the measure used to evaluate segmentation quality in the VOC segmentation challenge [7], is given by $100 \cdot \frac{\mathbf{y} \cap \mathbf{y}_{gt}}{\mathbf{y} \cup \mathbf{y}_{gt}}$ (with $\mathbf{y}$ denoting output segmentation and $\mathbf{y}_{gt}$ denoting ground truth). The average is computed over a certain range of scores, and we take $A_{low} = 85$, $A_{high} = 98$.

5.2. Dataset

The dataset (see Figure 10) consists of the GrabCut dataset (49 images) [1] augmented with images from the PASCAL VOC'09 segmentation challenge (99 images) [7] and 3 images from the alpha-matting dataset [17]. The VOC images come with ground truth labeling of object classes, and are cleaned up for the task of figure-ground segmentation. Images from the GrabCut dataset contain complex shapes but the foreground and background tend to have disjoint color distributions. The VOC dataset on the other hand has simpler shapes (car, bus) but more complex appearances, in which the color distributions of foreground and background overlap.

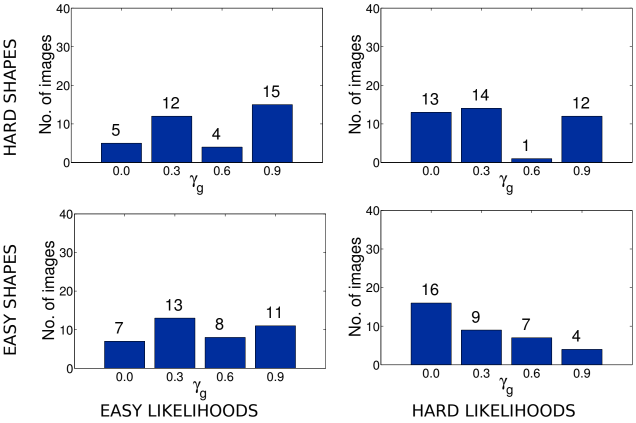


Figure 11. *Optimal γ_g as a function of shape and likelihood difficulty.* The above histograms visualize how often a particular γ_g is chosen as a function of likelihood and shape difficulty. Shapes are classified as easy or hard by sorting the per image occlusion rates in the visibility experiment and splitting at the median. The likelihood difficulty sorting is done using performances of a segmentation system that uses only likelihoods (i.e $\lambda = 0$ in (9)). With easy shapes and hard likelihoods (bottom right), $\gamma_g = 0$ is the majority choice and with hard shapes and easy likelihoods (top left), $\gamma_g = 0.9$ is the clear choice. For hard shapes+hard likelihoods and easy shapes+easy likelihoods, it's a trade off between shape and likelihood difficulty and neither is the clear winner.

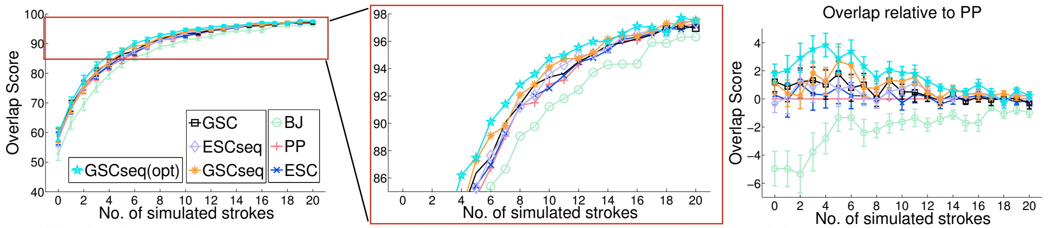
5.3. Parameter cross validation

To set free parameters such as γ_g in (5) and λ in (9), the image set is split into validation and test sets, and line search for the best parameters is performed on the validation set. Parameters are selected to minimize the area above the accuracy overlap vs. brush strokes curve. This is repeated for 10 splits, to estimate the stability of parameters.

6. Results

This section analyzes results obtained using the simulated user interaction on the introduced dataset. We first compare our different shape constraints with the Boykov Jolly energy function, and later compare our best system with existing methods [2, 9].

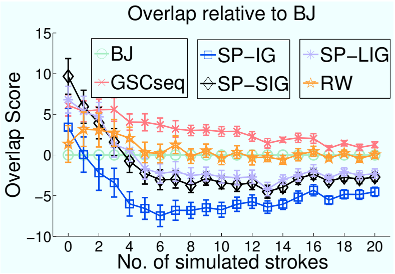
The first set of results compares: (i) *BJ* – Boykov Jolly with no shape constraint [3] (ii) *PP* – Post-Processing output of *BJ* to remove disconnected foreground islands [13]. (iii) *ESC* – Boykov Jolly with multiple star Euclidean Star-Convexity, implemented using geodesic forests with $\gamma_g = 0$. (iv) *GSC* – Boykov Jolly with Geodesic Star-Convexity



(a) Absolute performances (b) Absolute performances (Zoom) (c) Relative performances

Method	Avg. Effort
BJ	12.35
PP	10.66
ESC	10.57
GSC	10.23
ESCseq	10.23
GSCseq	9.63
GSCseq(opt)	8.97

Figure 12. *Overlap score vs. No. of strokes plots.* (a) Overlap scores for various methods are plotted as a function of no. of strokes with error bars for each stroke. (b) Zoom in of the region $[A_{low} = 85, A_{high} = 98]$ in the plot, error bars removed for clarity. (c) Overlap score measure relative to *PP*. The table next to the plots shows our measure – the interaction effort required to reach an accuracy of 98, in units of brush strokes. *GSCseq(opt)* is noted separately in the table as it involves additional effort in choosing γ_g.



Method	Avg. Effort
SP-IG	17.78
SP-SIG	15.77
SP-LIG	15.14
BJ	12.35
RW	12.31
GSCseq	9.63

Figure 13. *Comparison with other methods.* The plot compare *GSCseq* with existing methods. The performance differences are more remarkable here compared to Fig. 12 because this plot compares algorithms with different cost functions, whereas Fig. 12 compares algorithms with same cost function subject to different constraints. *GSCseq* defines the **state of the art** on this dataset.

implemented with geodesic forests computed on the likelihood image. The parameter γ_g is set by validation as described in section 5.3. (v) *ESCseq* – Sequential system (as described in sec. 4) on top of *ESC* (vi) *GSCseq* – Sequential system on top of *GSC* and (vii) *GSCseq(opt)* – Same as *GSCseq*, but with γ_g being set optimally for each image.

Before moving on to the discussion of these results, the system *GSCseq(opt)* needs further explanation. In our experiments, we observed that the optimal choice of the parameter γ_g can be image dependent. This is because of the varying difficulty of likelihoods and shapes in our dataset. We know from the visibility experiment (section 3) that for complex shaped objects, it is better to have a high γ_g – i.e be close to geodesic. However, that experiment is purely theoretical – in a real system the cost function also includes the likelihoods. The images in our dataset have complex appearances, and that can lead to noisy likelihoods. Such likelihoods can be a problem with high γ_g, as the geodesic constraint then connects up all the noisy likelihoods resulting in poor segmentations. In such cases, Euclidean ($\gamma_g = 0$), can behave better by ensuring that the shape does not bend arbitrarily and stays reasonable. Thus the choice of an optimal γ_g depends on the quality of the likelihoods and the complexity of the object shape. Our quantitative experiment (ref. fig. 11) confirms that – for complex shapes geodesic is preferred, and for complex likelihoods Euclidean . Thus, one could potentially let the user choose γ_g and this is what happens in *GSCseq(opt)* – the simulated user varies and chooses the γ_g that minimizes effort. Choosing γ_g requires additional effort, which means $GSCseq(opt)$ should not be compared with other systems. However, *GSCseq(opt)* still provides a useful benchmark as it lower bounds the effort required by any system that sets γ_g automatically.

Figure 12 visualizes results for these systems. We make the following observations: (i) The best system on the whole dataset is *GSCseq* – taking 9.63 brush strokes to reach an accuracy of 98%. (ii) All of our shape constraints perform better than simple post-processing. (iii) The geodesic system (*GSC*) improves over Euclidean (*ESC*) as demonstrated by the reduction in effort from 10.57 strokes to 10.23. (iv) The sequential system *GSCseq* has a definite advantage over the non-sequential system *GSC*, reducing effort from 10.23 strokes to 9.63 strokes. On a closer examination of Fig. 12, one would say that all methods except for *GSCseq(opt)* and *BJ* are perhaps too close and std-errs too large to draw definite conclusions. The Std-errs are an indication that our dataset is still very small and has a large variability on a per image basis. However, the performance order matches our intuition and theory, which is encouraging.

Our system is also compared with other algorithms. *GSCseq* is chosen as our system and compared with Bai and Sapiro [2] and Random Walker [9]. In [2], geodesics are computed on the likelihood image and each point assigned the label of its nearest stroke. In addition to that we also try variations of this method with geodesics computed on RGB gradients of raw and smoothed images. We use the following shorthands: (i) SP-LIG - geodesics computed on Likelihood Image Gradients, (ii) SP-IG - geodesics computed on raw image gradients. (iii) SP-SIG - geodesics computed on smoothed image gradients and (iv) RW - Random walker. The results are plotted in Figure 13. This evaluation really brings out a significant difference in performance across these methods. Shortest paths methods (*SP-LIG,SP-IG,SP-SIG*) take around 15 brush strokes and fail due to lack of any boundary regularization and sensitivity to brush lo-

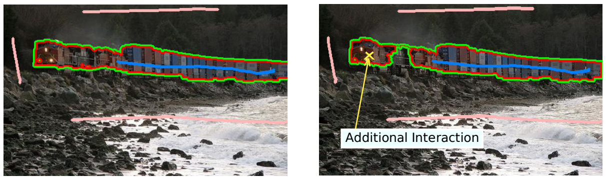


Figure 14. *Comparison with DijkstraGC* [26]. Both *GSC*(left) and *DijkstraGC*(right) are run with exactly the same energy function and user interaction. To connect the engine of the train to the rest, *DijskstraGC* needs further interaction as shown above. However, it only finds a very narrow path connecting the two sections of the train

cation [19]. *RW* only performs as well as BJ (≈12 brush strokes). Though *RW* is based on only pairwise terms, there is an extension which incorporates color models [8]. That extension however is a relaxation of BJ into the continuous domain, and is expected to have similar shortcomings to BJ.

We also show a qualitative comparison with *DijkstraGC* [26] in Fig. 14. This example clearly demonstrates the benefit of having more than just a simple connectivity constraint. *DijkstraGC* only connects up the segments with a narrow path, but GSC imposes a more sensible constraint and generates a better segmentation.

7. Conclusions and Further work

We summarize our contributions as follows. We extended the notion of star-convexity from single to multiple centers in a tractable way, and further generalized this notion from Euclidean to geodesic. To our knowledge, such definitions of star-convexity do not exist in the literature and we are the first to define them and apply them in a real system. We demonstrate using a rigorous evaluation system, how such extensions reduce interaction effort, defining the *state-of-the-art* system on a newly introduced dataset.

As future work, we would like to exploit star-convexity constraints to represent shapes. Indeed, we saw in this work that the shape constraint can model arbitrarily complex shapes and could be extended to object specific constraints as opposed to interactive constraints. These shape constraints can also be exploited in other ways suggested in [25] – such as reducing the shrinking bias using negative edge weights, ratio optimization and area constrained segmentation.

The datasets and code used in the quantitative evaluation can be found at: `http://www.robots.ox.ac.uk/~vgg/research/iseg/`

References

[1] Grabcut image dataset. http://research.microsoft.com/en-us/um/cambridge/projects/visionimagevideoediting/segmentation/grabcut.htm.

[2] X. Bai and G. Sapiro. Geodesic matting: A framework for fast interactive image and video segmentation and matting. *IJCV*, 2009.

[3] Y. Boykov and M. P. Jolly. Interactive graph cuts for optimal boundary and region segmentation of objects in N-D images. In *Proc. ICCV*, volume 2, 2001.

[4] A. Criminisi, T. Sharp, and A. Blake. GeoS: Geodesic image segmentation. In *Proc. ECCV*, 2008.

[5] P. Das, O. Veksler, V. Zavadsky, and Y. Boykov. Semiautomatic segmentation with compact shape prior. *Image Vision Comput.*, 2009.

[6] O. Duchenne, J.-Y. Audibert, R. Keriven, J. Ponce, and F. Segonne. Segmentation by transduction. In *Proc. CVPR*, 2008.

[7] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The PASCAL Visual Object Classes Challenge 2009 (VOC2009) Results.

[8] L. Grady. Multilabel random walker image segmentation using prior models. In *Proc. CVPR*, 2005.

[9] L. Grady. Random walks for image segmentation. *IEEE PAMI*, 28(11):1768–1783, 2006.

[10] P. Kohli, J. Rihan, M. Bray, and P. Torr. Simultaneous segmentation and pose estimation of humans using dynamic graph cuts. *IJCV*, 2008.

[11] M. P. Kumar, P. H. S. Torr, and A. Zisserman. OBJ CUT. In *Proc. CVPR*, volume 1, pages 18–25, 2005.

[12] V. Lempitsky, A. Blake, and C. Rother. Image segmentation by branch-and-mincut. In *Proc. of ECCV*, 2008.

[13] J. Liu, J. Sun, and H.-Y. Shum. Paint selection. In *Proc. ACM SIGGRAPH*, 2009.

[14] H. Nickisch, P. Kohli, and C. Rother. Learning an interactive segmentation system. *Technical Report*, 2009.

[15] S. Nowozin and C. H. Lampert. Global connectivity potentials for random field models. In *CVPR*, 2009.

[16] J. O'Rourke. *Art gallery theorems and algorithms*. Oxford University Press, Inc., New York, NY, USA, 1987.

[17] C. Rhemann, C. Rother, J. Wang, M. Gelautz, P. Kohli, and P. Rott. A perceptually motivated online benchmark for image matting. In *Proc. CVPR*, pages 1826–1833, 2009.

[18] C. Rother, V. Kolmogorov, and A. Blake. Grabcut: interactive foreground extraction using iterated graph cuts. *Proc. ACM SIGGRAPH*, 23(3):309–314, 2004.

[19] A. K. Sinop and L. Grady. A seeded image segmentation framework unifying graph cuts and random walker which yields a new algorithm. In *Proc. ICCV*, Oct. 2007.

[20] G. Slabaugh and G. Unal. Graph Cuts Segmentation Using an Elliptical Shape Prior. In *IEEE ICIP*, 2005.

[21] C. Smith. A characterization of star-shaped sets. *American Mathematical Monthly*, 75(4):386, 1968.

[22] M. Szummer, P. Kohli, and D. Hoeim. Learning CRFs using graph cuts. In *ECCV08*, 2008.

[23] F. Toranzos and A. Cunto. Sets expressible as finite unions of starshaped sets. *Journal of Geometry*, 2004.

[24] F. Valentine. *Convex sets*. McGraw-Hill, 1964.

[25] O. Veksler. Star shape prior for graph-cut image segmentation. In *Proc. ECCV*, 2008.

[26] S. Vicente, V. Kolmogorov, and C. Rother. Graph cut based image segmentation with connectivity priors. In *Proc. CVPR*, 2008.