Internation Journal in Computer Vision manuscript No.
(will be inserted by the editor)

Optimal Image and Video Closure by Superpixel Grouping

Alex Levinshtein · Cristian Sminchisescu · Sven Dickinson

Received: date / Accepted: date

Abstract Detecting independent objects in images and videos is an important perceptual grouping problem. One common perceptual grouping cue that can facilitate this objective is the cue of contour closure, reflecting the spatial coherence of objects in the world and their projections as closed boundaries separating figure from background. Detecting contour closure in images consists of finding a cycle of disconnected contour fragments that separates an object from its background. Searching the entire space of possible groupings is intractable, and previous approaches have adopted powerful perceptual grouping heuristics, such as proximity and co-curvilinearity, to constrain the search. We introduce a new formulation of the problem, by transforming the problem of finding cycles of contour fragments to finding subsets of superpixels whose collective boundary has strong edge support (few gaps) in the image. Our cost function, a ratio of a boundary gap measure to area, promotes spatially coherent sets of superpixels. Moreover, its properties support a global optimization procedure based on parametric maxflow. Extending closure detection to videos, we introduce the concept of *spatiotemporal closure*. Analogous to image closure, we formulate our spatiotemporal closure cost over a graph of spatiotemporal superpixels. Our cost function is a ratio of motion and appearance discontinuity measures on the boundary of the selection to an internal homogeneity measure of the selected spatiotemporal volume. The resulting approach automatically recovers coherent components in images and videos, corresponding to objects, object parts, and objects with surrounding context, providing a good set of multiscale hypotheses for high-level scene analysis. We evaluate both our image and video closure frameworks by comparing them to other closure detection approaches, and find that they yield improved performance.

Keywords Closure · Segmentation · Perceptual Grouping · Superpixels · Figure/Ground Segmentation · Motion Segmentation

University of Toronto
E-mail: {babalex,sven}@cs.toronto.edu ·
University of Bonn
E-mail: cristian.sminchisescu@ins.uni-bonn.de

1 Introduction

One of the key challenges in perceptual grouping is computing contour closure, i.e., linking together a set of fragmented contours into a cycle that separates an object from its background. What makes the problem particularly hard is the intractable number of cycles that may exist among the contours extracted from an image of a real scene. Early perceptual grouping researchers [61] identified a set of non-accidental contour relations, such as symmetry, parallelism, collinearity, co-curvilinearity, etc., that can be used to link together causally related contours. Such non-accidental grouping rules can serve as powerful heuristics to help manage the complexity of greedily searching for a contour closure that is unlikely to have arisen by chance [19, 20]. However, the space of possible closures is still overwhelming, particularly when one allows larger and larger boundary gaps in a closure. Finding an optimal solution is intractable without somehow reducing the complexity of the problem.

The closure cue for images can be extended to the spatiotemporal domain, facilitating video segmentation. In spatiotemporal closure, image contour fragments would correspond to spatiotemporal surface fragments that lie on appearance or motion boundaries. Segmenting the

video would then correspond to grouping such surface fragments into closed spatiotemporal surfaces. What makes video segmentation challenging is the strong coupling that exists between the estimation of an object's spatial support and the estimation of its motion parameters. On one hand, local motion estimates may be unreliable, especially in untextured regions, and larger spatial support is needed for accurate motion estimation. On the other hand, appearance alone may not be enough to recover the object's spatial support in cases of heterogeneous object appearance or low contrast with the background, and we may need to rely on motion to define the correct spatial support for objects. This chicken and egg problem forces most video segmentation techniques to resort to restrictive modeling assumptions or suboptimal solutions to the problem.

In this paper, we introduce a novel framework for efficiently searching for optimal image and spatiotemporal closures. This is an extension of our work in [32] about image closure and our work in [33] about spatiotemporal closure. The current paper unifies the closure cost functions from the previous two papers, offers more related work discussion, and additional analysis of the results. Fig. 1 illustrates an overview of our approach for image closure. Given an image of extracted contours (Fig. 1(a)), we begin by restricting contour closures to pass along boundaries of superpixels computed over the contour image (Fig. 1(b)). In this way, our first contribution is to reformulate the problem of searching for cycles of contours as the problem of searching for a subset of superpixels whose collective boundary has strong contour support in the contour image; the assumption we make is that those salient contours that define the boundary of the object (our target closure) will align well with superpixel boundaries. However, while a cycle of contours represents a single contour closure, our reformulation exploits a mechanism to encourage superpixel subsets that are spatially coherent.

Spatial coherence is an inherent property of a cost function that computes the ratio of perimeter to area. We modify the ratio cost function of Stahl and Wang [52] to operate on superpixels rather than contours, and extend it to yield a cost function that: 1) promotes spatially coherent selections of superpixels; 2) favors larger closures over smaller closures; and 3) introduces a novel, learned gap function that accounts for how much agreement there is between the boundary of the selection and the contours in the image. The third property adds cost as the number and sizes of gaps between contours increase. Given a superpixel boundary fragment (e.g., a side of a superpixel) representing a hypothesized closure component, we assign a gap cost that's a function of the

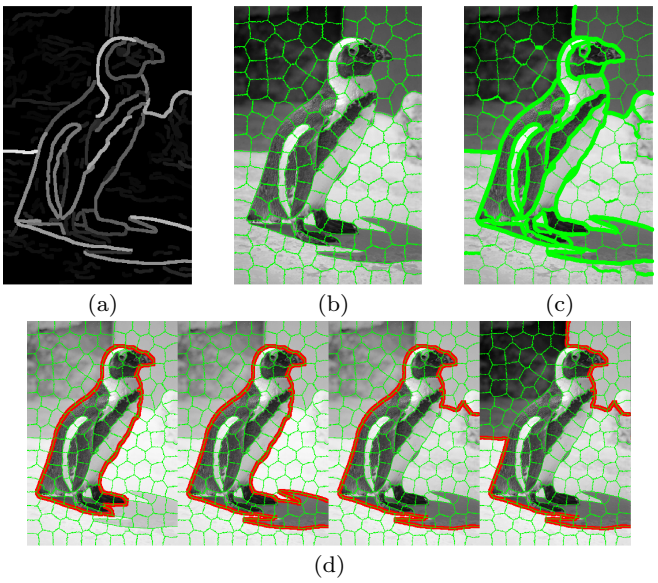


Fig. 1: Overview of our approach for image closure: (a) contour image – while we take as input only this contour image, we will overlay the original image in the subsequent figures to ease visualization; (b) superpixel segmentation of contour image, in which superpixel resolution is chosen to ensure that target boundaries are reasonably well approximated by superpixel boundaries; (c) a novel, learned measure of gap reflects the extent to which the superpixel boundary is supported by evidence of a real image contour (line thickness corresponds to the amount of agreement between superpixel boundaries and image contours); (d) our cost function can be globally optimized to yield the largest set of superpixels bounded by contours that have the least gaps. In this case the solutions, in increasing cost (decreasing quality), are organized left to right.

proximity of nearby image contours, their strength, and their orientation (Fig. 1(c)). It is in this third property that our superpixel reformulation plays a second important role – by providing an appropriate scope of contour over which our gap analysis can be conducted.

In our third contribution, the two components of our cost function, i.e., area and gap, are combined in a simple ratio that can be efficiently optimized using parametric maxflow [29] to yield the global optimum. The optimal solution yields the largest set of superpixels bounded by contours that have the least gaps (Fig. 1(d)). Moreover, parametric maxflow can be used to yield the top k solutions (see [8], for example). In an object recognition setting, generating a small set of such solutions can be thought of as generating a small set of promising shape hypotheses which, through an indexing process, could invoke candidate models that

could be verified (detected). The use of such multiscale hypotheses was shown to facilitate state-of-the-art object recognition in images [35].

In our fourth and final contribution, we extend the framework to detect spatiotemporal closure. Similar to detecting contour closure in images, we formulate spatiotemporal closure detection inside a spatiotemporal volume (Fig. 2a) as selecting a subset of spatiotemporal superpixels whose collective boundary falls on such discontinuities (Fig. 2b). Our spatiotemporal superpixels, extending our superpixel framework in [34], provide good spatiotemporal support regions for the extraction of appearance and motion features, while limiting the undersegmentation effects exhibited by other superpixel extraction techniques due to their lack of compactness and temporal stability.

We proceed by forming a superpixel graph whose edges encode appearance and motion similarity of adjacent superpixels (Fig. 2c). Next, we formulate spatiotemporal closure. The notion of contour gap from image closure detection is generalized to the cost of a cut of a set of spatiotemporal superpixels from the rest of the spatiotemporal volume, where the cut cost is low for superpixel boundaries that cross appearance and motion boundaries. Similarly, instead of normalization by area, we choose to normalize by a measure of internal motion and appearance homogeneity of the selection, which is more appropriate for video segmentation. The cost is again minimized using parametric maxflow [29] that is not only able to efficiently find a globally optimal closure solution, but returns multiple closure hypotheses (Fig. 2e). This not only eliminates the need for estimating the number of objects in a video sequence, as all objects with the best closure are extracted, but can result in hypotheses that oversegment objects into parts or merge adjacent objects. Multiple spatiotemporal segmentation hypotheses can serve tasks such as action recognition, video synopsis and indexing [45].

In the following sections, we begin by reviewing related work on image and video segmentation (Section 2). Next, in Section 3, we formulate closure detection as the selection of an optimal subset of superpixels and define our cost function. We will also present an efficient procedure for finding the global minimum of our ratio-based cost function using parametric maxflow. In Sections 4 and 5, we describe details specific to image and spatiotemporal closure detections, respectively. In Section 6, we evaluate our framework. For image closure, we compare it to two competing closure detection approaches. For spatiotemporal closure, we compare our method to Normalized Cuts [51] on the same superpixel graph. In Section 7, we discuss the strengths and weaknesses of our approach, and outline plans for future work. We draw conclusions in Section 8.

2 Related Work

Image and video segmentation pose two of the main challenges in computer vision. In image segmentation, one has to consider a prohibitive number of possible pixel groupings or cycles of contour fragments that separate the figure from the background. In video segmentation, the combinatorial nature of possible groupings is even larger, and one needs to jointly estimate not only an object's spatial extent but also its motion. Using prior information about object appearance, motion or other scene content significantly simplifies the problem. For instance, many segmentation techniques are formulated as energy minimization problems that could be solved using min-cut in an efficient manner. However, the corresponding energy functions typically include terms that require prior object knowledge in terms of user interaction [6,48,30] or knowledge about object appearance. We think of segmentation as a necessary bottom-up preprocessing step for indexing or recognition, providing substantial reduction in the computational complexity of these tasks. It is therefore unclear how segmentation methods that use strong prior knowledge are applicable for object recognition from large databases. To that end, we will not cover methods that employ strong object or motion priors. We will split our review into separate discussions of image and video segmentation.

2.1 Contour Closure

We will begin our review with related work on closure detection in images. Detecting closed contours in an image has been addressed by many researchers in different ways. One possible taxonomy for categorizing related work is based on the nature of the prior information used to constrain the grouping process. We focus on methods that make no assumptions about scene content, although as we will see, many make assumptions about the nature of parts that make up the objects in the scene. In fact, some methods incorporate low-,mid-, and high-level shape priors, as exemplified by Ren et al. [47]. We will also not cover methods focused solely on contour completion, e.g., Ren et al. [46] and Williams and Jacobs [63], although the regularities exploited by such approaches can clearly play a powerful role in detecting closure.

Many researchers have exploited the classical Gestalt cues of parallelism and symmetry to group contours.

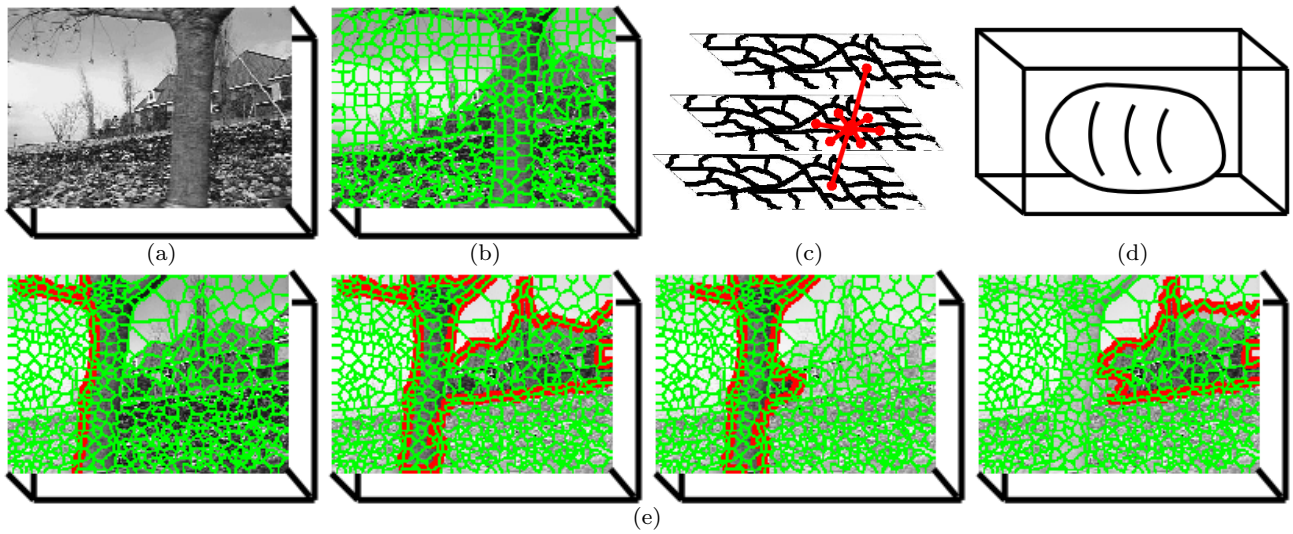


Fig. 2: Overview of our approach for spatiotemporal closure. (a) Spatiotemporal volume; (b) Spatiotemporal superpixels; (c) Superpixel graph with edges encoding appearance and motion affinity; (d) Optimizing our spatiotemporal closure corresponds to finding a closed surface cutting low affinity graph edges; (e) Our optimization framework results in multiple multiscale hypotheses, corresponding to objects, objects with their context, and object parts.

Lowe's [36] early work on perceptual grouping was one of the first to develop a computational model for parallelism, collinearity, and proximity. Many computational models exist for symmetry-based grouping, including Brady and Asada [7], Cham and Cipolla [9], Saint-Marc et al. [49], Ylä-Jääski and Ade [64], and more recently, Stahl and Wang [53]. One significant challenge faced by these systems is the complexity of pairwise contour grouping to detect symmetry-related contour pairs. Our work in [31] attempts to overcome this computational complexity limitation by constraining the symmetric parts to collections of superpixels. This paper draws on this idea of grouping superpixels, but will relax the symmetry constraint and focus on the more generic perceptual grouping rule of closure.

Further down the spectrum of prior knowledge are methods based on weaker shape priors than parallelism and symmetry. For example, Jacobs [25] uses convexity as well as gap to extract closed contours by grouping straight line segments. A less restrictive measure is that of compactness, which can be attained by normalizing the gap by area (Estrada and Jepson [19,20], Stahl and Wang [52]). Some measure of internal homogeneity can also be used (Estrada and Jepson [20], Stahl and Wang [52]), provided that the inside of the region is easily accessible.

Finally, we come to the most general methods that compute closure using only very weak shape priors, such as continuity and proximity. The most basic closure-based cost function uses a notion of boundary gap, which is a measure of missing image edges along the closed contour. Elder and Zucker [17] model the probability of a connection between two adjacent contour fragments, and find contour cycles using a shortest path algorithm. Wang et al. [57] optimize a measure of average gap using the ratio cut approach. However, a measure based purely on the total boundary gap is insufficient for perceptual closure. Elder and Zucker [16] argue that the distribution of gaps along the contour is also important. Williams and Hanson [62] addressed the problem of perceptual completion of occluded surfaces, formulated as the problem of computing a labeled knot-diagram representing a set of occluded surfaces from observed image contours. While formulated as an elegant combinatorial optimization problem, for which an optimal solution was available, the approach was only tested on synthetic images.

All the above methods suffer from the high complexity of choosing the right closure from a sea of contour fragments. To cope with complexity, they either resort to heuristics to prune the search (e.g., [25]) or constrain the search space by other means (e.g., restricting the closure to alternating gap/non-gap cycles [52]). Zhu et al. [65] propose a solution that embeds the edge fragments into polar coordinates such that closed contours correspond to circles in that space; however, their goal is to better detect object contours, and they do not group the contours into closed boundaries. The method of Jermyn and Ishikawa [27] is perhaps the closest to our work. Similar to [57,52], they minimize closure cost using ratio cuts, but unlike [57,52] who operate on contour fragments, [27] works directly with pixels in a 4-

connected image grid. This allows for the minimization of many different closure costs (including our own) by globally optimizing ratio cuts in a simply connected planar graph. However, individual pixels provide poor scope for gap computation. In contrast, our superpixels not only provide greater scope for gap computation (which in our case is learned), but provide greater scope for the incorporation of internal appearance-based affinity. Finally, while their solution is optimal, it does not provide a set of optimal solutions that capture closures at multiple scales.

There are two concurrent works that do extract multiple figure/ground, similar to our method. The first is the work of Endres and Hoiem [18]. They too start with superpixels, but generate multiple proposals by varying the parameters of a Conditional Random Field built over a superpixel graph. In contrast, we are not changing the parameters of our energy function and minimize a closure cost based on the powerful and ubiquitous intermediate shape prior of closure. Similarly, while Carreira and Sminchisescu [8] employ the same parametric maxflow approach to efficiently extract multiple figure/ground hypotheses, they do not relate their cost to a known perceptual grouping cue such as closure. Moreover, unlike our approach, [8] works with image pixels, rather than superpixels or contour fragments, making it harder to encode more promising affinity information during grouping. Finally, neither method is extended into spatiotemporal segmentation.

In this paper, our goal is to find closed contour groups in an efficient manner. To that end, we use superpixels to constrain the search space of the resulting closures. Superpixels also provide an easy way to access internal region information (such as region area). Moreover, superpixel boundaries provide better scope for gap computation, as opposed to most previous methods that linearize the output of an edge detector and fill the gaps with straight line fragments. On the optimization side, we show that parametric maxflow [29] can be used not only to recover the global optimum of closure costs similar to that of Stahl and Wang [52] and Jermyn and Ishikawa [27], but can also be used to recover a multiscale set of closure hypotheses.

2.2 Spatiotemporal Closure

A full interpretation of a dynamic scene is a great challenge in computer vision. Tracking methods often adopt a high-level probabilistic scene representation, where objects are modeled as low-dimensional state vectors whose probability at any given instance is a function of the observed data and the temporal dynamics. Inferring object states in real world motion sequences is difficult due to occlusion, camera motion, and variability in object appearance, dynamics, and shape. As a result, tracking techniques are forced to restrict their models of observed data likelihood and motion [60, 4], or resort to approximation techniques to infer object states [24, 11]. In contrast, our focus in this paper is on spatiotemporal segmentation. Unlike tracking, where objects are represented at a high level, spatiotemporal segmentation is a low-level task that aims to automatically extract precise object boundaries given generic perceptual grouping regularities, such as similarity, proximity and common fate (motion similarity).

Spatiotemporal segmentation methods can be divided into two categories, layer-based approaches and generic segmentation techniques (a good review is provided in Megret and Dementhon [40]). In the first category, a scene is represented using overlapping layers, with each layer capturing a coherently moving object or part [56, 59, 58, 28, 26]. Most such approaches are limited by either assuming a fixed number of layers, assuming a restricted motion model per layer, or resorting to suboptimal techniques that iteratively estimate the spatial extent and the motion of each layer. Nevertheless, this strong global model of a scene enables layer-based methods to successfully segment objects in video sequences in the presence of occlusion, appearance changes, and other effects. In this work, however, we will focus on more generic, less constrained models for spatiotemporal segmentation.

The second category of approaches does not use strong models and attempts to segment a video based on generic spatiotemporal information. Methods mainly differ in their segmentation algorithms and their treatment of the spatiotemporal volume, with some methods analyzing the volume in a framewise manner and others treating it as a single 3D entity. One set of techniques models moving objects with active contours. In Bascle and Deriche [3], motion is modeled with a global warp which is computed by correlating internal region appearance in successive frames. Following the warp, however, only appearance information is used to update the region's contour. Paragios and Deriche [43] propose an elegant geodesic active contour formulation. Unlike [3], both motion and appearance information are used in active contour evolution and their level-set framework enables them to easily handle the splitting and merging of contours. However, they assume a static background model to facilitate automatic contour initialization and tracking. A similar method is proposed by Chung et al. [10], who employ the EM framework to iterate between region motion estimation and segmentation using active contours, but unlike [43] do not rely on a static background. Cremers and Soatto [12] propose a more

global approach and treat the spatiotemporal volume as a single entity instead of working with pairs of frames. However, their approach provides no automatic initialization and does not estimate the number of objects in a scene.

A different set of techniques opts for a dominantly bottom-up approach, and finds spatially and temporally coherent clusters. Similar to methods based on active contours, some of these approaches handle the spatiotemporal volume in a framewise fashion [42,55, 14,1,44]. While such techniques are applicable to real-time segmentation, some opt to treat the video stack as a single entity facilitating more global constraints. Dementhon [13] and Greenspan et al. [22] are examples of two techniques that represent videos with distributions in a low-dimensional feature space (7D in [13] and 6D in [22]). While such model allows support the efficient segmentation of videos by employing non-parametric (a mean-shift-based technique in [13]) or parametric (GMM in [22]) clustering, low-dimensional models can be restrictive for many motion sequences. Instead of explicitly modeling video sequences in some Euclidean space, segmentation can be formulated as an optimization of a global cost that is based on pairwise similarities between neighboring points in the spatiotemporal stack. For example, in [50,21], video segmentation is formulated as a normalized cuts problem, further extended by Huang et al. [23] to handle more global interactions. Our approach falls in this category as it also defines a global cost function. However, unlike Ncuts-based techniques that are forced to resort to approximate solutions, we are able to find an exact global optimum of our cost. Moreover, the number of partitions does not have to be specified a priori, as we automatically detect a multiscale set of spatiotemporal clusters.

3 Problem Formulation

3.1 Closure Cost

We formulate closure detection as a superpixel selection problem. Detecting closure is formulated as finding a subset of superpixels that minimizes a cut-based closure cost over a superpixel graph, with graph edges encoding superpixel similarity. For contour closure detection in images, the graph consists of image superpixels, while for video segmentation, the graph will consist of spatiotemporal superpixels. However, minimizing the cut alone unfairly penalizes larger selections. It was previously shown (see [52] or [51], for example) that normalization of the cut by a measure that is proportional to the size of the selection yields better results. We adopt this measure and design a closure cost that is a ratio of a cut to a selection size, where size can measure the area or the appearance homogeneity of the selection, for example. Optimizing this cost over superpixels enables us to efficiently recover coherent spatial and spatiotemporal segments out of an exponential number of superpixel subsets.

Formally, let $SG = (V, W)$ be a superpixel graph with nodes V corresponding to superpixels and W corresponding to weights on graph edges. Let $\mathbf{X} \in \{0,1\}^N$ be a superpixel indicator vector, where $N = |V|$ is the number of superpixels. We further define an edge weight W_{ij} to encode the similarity between two superpixels i and j that are connected by an edge. $W_{ij} = 0$ if superpixels i and j have no edge between them. For convenience, let $D_i = \sum_j W_{ij}$. Finally, let $S : V \rightarrow R^+$ be an arbitrary non-negative function over superpixels, corresponding to some measure of superpixel size. For simplicity we will denote $S_i = S(v_i)$. Our closure cost is defined as follows:

$$C(\mathbf{X}) = \frac{cut(\mathbf{X})}{size(\mathbf{X})} = \frac{\sum_{ij} X_i(1-X_j)W_{ij}}{\sum_i S_i X_i} = \frac{\sum_i D_i X_i - 2\sum_{i<j} X_i X_j W_{ij}}{\sum_i S_i X_i} \quad (1)$$

where $cut(\mathbf{X})$ is the sum of the affinities of all the edges between selected ($X_i = 1$) and unselected ($X_i = 0$) superpixels, and $size(\mathbf{X})$ is the total size of the selected superpixels. Minimizing the ratio $C(\mathbf{X})$ is equivalent to minimizing the numerator $cut(\mathbf{X})$ while maximizing the denominator $size(\mathbf{X})$. The cut between selected and unselected superpixels is small when selected superpixels are strongly dissimilar from the rest. Normalization by size pushes the solution towards large and compact subsets of superpixels. Manipulating the semantics of the edge weights W and the size S makes our framework generic for various types of closure costs. For example, by setting W_{ij} to measure the gap between spatially adjacent superpixels and setting S_i to be the area of i-th superpixel, we reduce the cost in Eqn. 1 to that of Stahl and Wang [52]. If we let W_{ij} be a generic measure of similarity and set $S_i = D_i$, our cost becomes the unbalanced Normalized Cuts cost. We can also set $S_i = 1$ to normalize by the number of superpixels in the selection. Sections 4 and 5 provide details on the graph construction and the settings for W and S that we used for contour closure and spatiotemporal closure detection, respectively. Note that in the current form Eqn. 1 has a trivial solution by setting $\mathbf{X}$ to be a vector of ones. This issue can be resolved by penalizing some superpixels. Specific details on this penalty will be given in sections 4 and 5.

3.2 Optimization using Parametric Maxflow

It has been known for some time that ratios of real variables that adhere to certain constraints can be minimized globally [15] . Instead of minimizing the ratio $R(x) = \frac{P(x)}{Q(x)}$ directly, one can minimize a parametrized difference $E(x, \lambda) = P(x) - \lambda Q(x)$. It can be shown that the optimal λ corresponds to the optimal ratio $\frac{P(x)}{Q(x)}$ and can be efficiently recovered using a binary search or Newton's method for fractional optimization. The constraints on the ratio guarantee that the resulting difference is concave and thus can be minimized globally.

In the case of binary variables, ratio minimization can be reduced to solving a parametric maxflow problem. Kolmogorov et al. [29] showed that under certain constraints on the ratio $R(x)$, the energy $E(x, \lambda)$ is submodular and can thus be minimized globally in polynomial time using min-cuts. Converting our closure cost $C(\mathbf{X})$ in Eqn. 1 to a parametrized difference results in a submodular cost $C(\mathbf{X}, \lambda)$, making the method in [29] applicable for minimizing the ratio $C(\mathbf{X})$.

$$\begin{aligned} C(\mathbf{X}, \lambda) &= cut(\mathbf{X}) - \lambda \cdot size(\mathbf{X}) \\ &= \sum_i D_i X_i - 2 \sum_{i<j} X_i X_j W_{ij} - \lambda \sum_i S_i X_i \end{aligned} \tag{2}$$

In fact, the method in [29] does not simply optimize the ratio $R(x)$, but finds all intervals of λ (and the corresponding x) for which the solution x remains constant. Different λ's correspond to different weights of the cut against the selection size. Parametric maxflow can optimize the above parametrized cost, and efficiently find all the different *breakpoints* (interval boundaries) of λ with stationary optimal solution X, resulting in a monotonically increasing sequence of breakpoints $\lambda_0, \lambda_1, \lambda_2, \ldots, \lambda_K$. Kolmogorov et al. [29] show that while the solution $\mathbf{X}^*$ in range $0 \leq \lambda \leq \lambda_0$ corresponds to the global minimum of $C(\mathbf{X})$, consecutively larger breakpoints $\lambda_1, \lambda_2, \ldots, \lambda_K$ are also related to ratio optimization. In fact, the optimal solution $\mathbf{X}^i$ of $C(\mathbf{X}, \lambda)$ in the interval $[\lambda_i, \lambda_{i+1}]$, is also an optimal solution of $\min_{size(\mathbf{X}) \geq T} C(\mathbf{X})$, where $T = size(\mathbf{X}^i)$. Therefore, employing parametric maxflow results in several closure solutions where optimal cuts are found with increasing size constraints. We refer the reader to [29] for more details on the parametric maxflow method.

Solving for all breakpoints can be exponential if the number of breakpoints is exponential, but is polynomial for obtaining a global optimum. Moreover for monotonic parametric maxflow (as is the case of the model given by Eqn. 2) the number of breakpoints is linear in N (number of nodes in the graph). In our experiments, a solution is obtained in a fraction of a second for a graph of 200 superpixels nodes (contour closure), as there are typically less than 10 breakpoints. The method takes approximately one second for a graph of several tens of thousands of superpixels (as it arises in spatiotemporal closure problems).

4 Image Closure

4.1 Image closure cost

As mentioned in Section 1, our framework reduces grouping complexity by restricting closure to lie along superpixel boundaries. This restriction provides us with a better context to incorporate mid-level cues in our closure cost computation. Furthermore, as we show below and illustrate in Fig. 3 , having disjoint superpixels enables us to formulate the total gap around a region as a quadratic function of superpixel indicator variables, which in turn allows us to efficiently minimize the closure cost using parametric maxflow.

Given a contour image $I(x, y)$[1], we first segment it into N superpixels. We use a modified version of the superpixel segmentation method of Mori et al. [41] ([41] uses the Pb edge detector [39], while we use globalPb [38]), but any other fast superpixel extraction method can be used. Note that our spatiotemporal closure in the next section uses a modified version of Turbopixels [34], as Turbopixels are reasonably fast and naturally extendable to the spatiotemporal domain. For image closure we choose the method of Mori et al. [41] as it provides slightly better boundaries at the expense of computational speed. Recall that our goal will be to select a maximal set of superpixels which have high spatial coherence and whose boundary has strong contour support in the image. Drawing on Stahl and Wang [52], we define the closure cost to be $C(\mathbf{X}) = \frac{G(\mathbf{X})}{A(\mathbf{X})}$, where $G(\mathbf{X})$ is the boundary gap along the perimeter of (the "on" superpixels of) $\mathbf{X}$, and $A(\mathbf{X})$ is its area. Boundary gap is a measure of the disagreement between the boundary of $\mathbf{X}$ and is defined to be $G(\mathbf{X}) = P(\mathbf{X}) - E(\mathbf{X})$, where $P(\mathbf{X})$ is the perimeter of $\mathbf{X}$ and $E(\mathbf{X})$ is the "edginess" of the boundary of $\mathbf{X}$. Out of the total number of pixels along the boundary of $\mathbf{X}$, $P(\mathbf{X})$, edginess is the number of edge pixels, with the edginess of image boundary pixels defined to be 0.

The above closure cost can be reduced to the generic closure cost formulation in Eqn. 1 by appropriately setting the edge weights W and the size measure S. Specifically, let P_{ij} be the length of the shared edge between superpixels i and j. Similarly, let E_{ij} be the edginess

[1] The contour image takes the form of a globalPb image [38].

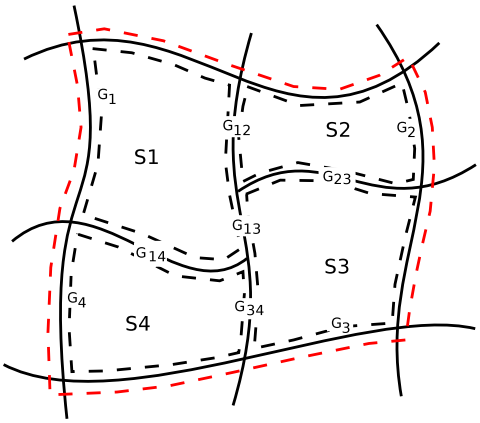


Fig. 3: Boundary gap computation over superpixel graph. S_1, S_2, S_3, and S_4 correspond to superpixels that were selected. G_i and G_{ij} are the boundary gap of superpixel i and the gap on the edge between superpixels i and j, respectively. The gap along the boundary of the selection (red) is then $G_{1234} = G_1 + G_2 + G_3 + G_4 - 2\,(G_{12} + G_{13} + G_{14} + G_{23} + G_{34})$.

of the shared boundary between the two superpixels. Given these definitions, $G_{ij} = P_{ij} - E_{ij}$ is the superpixel gap on the boundary between superpixels i and j. Note that $G_i = \sum_j G_{ij}$ is the total gap on the boundary of the i-th superpixel. Finally, let A_i be the area of superpixel i. By setting the edge weight $W_{ij} = G_{ij}$ and $S_i = A_i$, the cost in Eqn. 1 becomes:

$$C(\mathbf{X}) = \frac{gap(\mathbf{X})}{area(\mathbf{X})} = \frac{\sum_i G_i X_i - 2\sum_{i<j} X_i X_j G_{ij}}{\sum_i A_i X_i} \quad (3)$$

The denominator in the above ratio simply adds the individual areas of all selected superpixels. Normalization by area not only promotes spatial coherence[2] but also promotes compactness. In Section 6 we show that given two possible paths (with strong edge support) the method will prefer a compact path over one with deep concavities. Specifically, for two solutions $\mathbf{X}$ and $\mathbf{Y}$ of the same size ($A(\mathbf{X}) = A(\mathbf{Y})$) and with the same amount of contour support ($E(\mathbf{X}) = E(\mathbf{Y})$), $C(\mathbf{X}) < C(\mathbf{Y})$ if $P(\mathbf{X}) < P(\mathbf{Y})$. This in turn means that $\mathbf{X}$ is the more compact of the two solutions. The numerator of the cost is more complicated. To compute the gap along the perimeter, we first accumulate the individual gaps for all the selected superpixels. However, for selected superpixels that share boundaries, adding individual superpixel gaps would incorrectly include gaps that are not on the boundary of the selection. Moreover, for every internal boundary, the boundary gap is doublecounted (once for each of the superpixels that share that boundary). Therefore, we need to correct the measure by subtracting the gap twice for all internal boundaries. Note that if two superpixels do not have a shared boundary, then both P_{ij} and E_{ij} will be 0. As a result, $W_{ij} = G_{ij} = 0$, indicating that superpixels i and j have no edge in the superpixel graph SG. Furthermore, superpixels that touch the image boundary incur a natural penalty as all image boundary pixels have 0 edginess. As a result, closures are discouraged from touching the image boundary thereby avoiding the trivial solution discussed in section 3.1. Fig. 3 illustrates the gap computation for a simple superpixel graph. In the next section, we introduce our gap measure, and show how it can be learned from training data.

[2] While spatial coherence is promoted, it is not guaranteed. Since minimizing Eqn. 3 can occasionally result in disconnected sets of superpixels, we further guarantee connectedness by selecting the largest-area connected component of $\mathbf{X}$.

4.2 Learning the gap measure

Most approaches to detecting contour closure (e.g., [52]) typically define gap as simply the length of the missing contour fragments, i.e., the length of that portion of the closure that covers no image edges. In order to ground our gap measure using image evidence, as well as incorporate multiple contour features for gap computation, we learn it from ground truth. In Section 4.1 we define the gap on the boundary between a pair of superpixels i and j as $G_{ij} = P_{ij} - E_{ij}$. Specifically, if $\mathbf{EP_{ij}}$ is the set of pixels on the boundary between superpixels (i, j), then $P_{ij} = |\mathbf{EP_{ij}}|$ and $E_{ij} = \sum_{p \in \mathbf{EP_{ij}}} E_{ij}^p$, where $E_{ij}^p = [P(\mathbf{f^p}) > T_e]$ is an edge indicator for pixel p ($P(\cdot)$ is a logistic regressor and $\mathbf{f^p}$ is a feature vector for pixel p). Notice that we threshold the edginess measure instead of using it directly. T_e is a necessary threshold on the edginess measure. Since the distribution of edges in the training set is not necessarily the same as for test images, this parameter controls the contribution of weak edges. Moreover, T_e lets us control the relative (to the area) effect of the gap on the closure cost (similar to α in [52]). Decreasing it results in smaller structure generation and increases the number of potential solutions generated by the algorithm. We analyze the performance of our method as a function of this parameter in Section 6.

Given a pixel p on the superpixel boundary, the feature vector $\mathbf{f^p}$ is a function of both the local geometry of the superpixel boundary and the detected image edge response in its neighbourhood. The feature vector consists of three components (see Fig. 4):

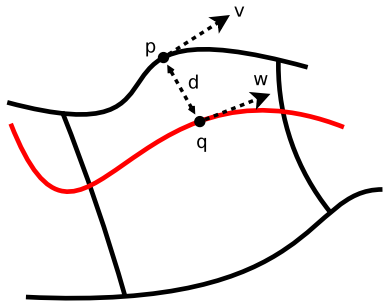


Fig. 4: Contour features for learning the gap measure. Black curves correspond to superpixel boundaries, while the red curve corresponds to detected image edges. The features that are used for edge weight computation at superpixel boundary pixel p are: 1) distance d between p and q, where q is the closest point to p among the detected image edges; 2) image edge strength at q; and 3) the alignment, computed as the absolute value of the cosine of the angle between v and w.

1. Distance to the nearest image edge; closer edges provide stronger evidence.
2. Strength of the nearest image edge; stronger edges provide stronger evidence.
3. Alignment between the tangent to the superpixel boundary point and the tangent to the nearest image edge; aligned edges provide stronger evidence.

Given a dataset of images with manually labeled figure/ground, we map the ground truth onto superpixels. Our training data consists of all pixels falling on superpixel boundaries where positive training data consists of pixels that fall on figure/ground boundaries and negative training data consists of all the other pixels on superpixel boundaries. We collect our training data from the Weizmann Horse dataset and use it to train a logistic classifier over the feature vector $\mathbf{f^p}$ to predict the likelihood of p being on an object boundary. Note that our aim is to simply learn the best generic weights for combining the gap features and not train a boundary detector for horses. The same logistic parameters are used for the gap measure throughout this paper. In addition to learning from all of the above features, we also learn from feature subsets. Fig. 5 illustrates the effect of incrementally adding more features by evaluating boundary precision and recall on a test set of images from the Weizmann Horse dataset; the thickness of the boundary between each pair of superpixels in Fig. 5(b) corresponds to the average edge probability of its superpixel boundary pixels, accumulated for all pixels on the superpixel boundary. Using all three features results in the best performance, in terms of retaining object boundary edges while suppressing other edges.

5 Spatiotemporal Closure

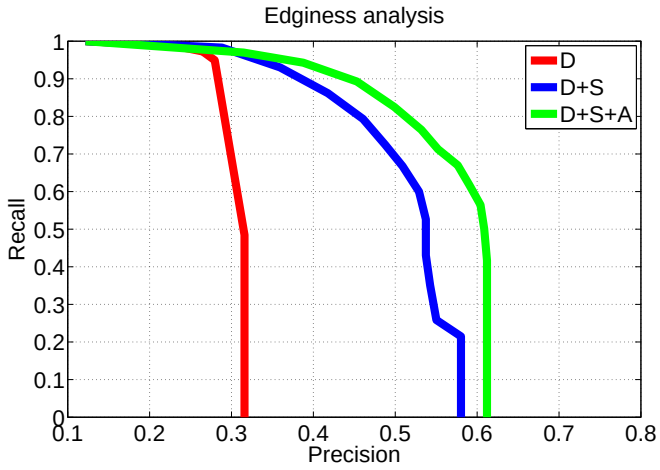
5.1 Spatiotemporal closure cost

For spatiotemporal closure detection, we define our closure cost to be the unbalanced normalized cuts cost over a superpixel graph. Given a superpixel segmentation of every frame in a video, we start by building a superpixel graph with spatial and temporal connections. Note that in the case of spatiotemporal closure, the set of vertices V in the superpixel graph SG consists of superpixels from all the frames combined. We connect each superpixel to its spatial and temporal neighbors and define an affinity W_{ij} for each pair of neighboring superpixels i and j, encoding their appearance and motion similarity. Our spatiotemporal closure cost becomes:

$$C(\mathbf{X}) = \frac{cut(\mathbf{X})}{volume(\mathbf{X})} = \frac{\sum_i D_i X_i - 2\sum_{i<j} X_i X_j W_{ij}}{\sum_i D_i X_i} \tag{4}$$

where $cut(\mathbf{X})$ is the sum of the affinities of all the edges between selected and unselected superpixels, and $volume(\mathbf{X})$ is the sum of all the affinities originating from the selected superpixels. The cut between selected and unselected superpixels is small when selected superpixels are strongly separated from the rest in terms of their appearance and motion. Normalization by volume pushes the solution towards large and compact subsets of superpixels that have homogeneous appearance and motion. The spatiotemporal closure cost in Eqn. 4 is similar to our 2D contour closure cost in Eqn. 3, with the exception that the numerator measures the cut instead of the gap and is normalized by affinity volume instead of area. We will also show that the affinities W_{ij} can include the length of the boundary between superpixels or their area. This gives larger superpixels greater influence.

Our closure detection algorithm consists of several stages. We start by extracting superpixels for each frame of the video. Subsequently, we construct a graph where each superpixel is a node connected to its spatial and temporal neighbors. Each edge in the graph is assigned a weight that measures the degree of superpixel similarity. Once the graph is built, the cost in Eqn. 4 is optimized using parametric maxflow. Finally, we post-process the solutions to detect connected components, remove similar or spurious results, and generate other potentially good solutions. The following subsections



(a)

Only distance + strength + alignment

(b)

Fig. 5: Superpixel boundary edginess as a function of different features. Adding more features results in a better edginess measure (a) Quantitative evaluation - precision/recall of contour points using different features (D - Distance, S - Strength, A - Alignment); (b) Qualitative evaluation (ordered left to right). For example, the edginess of superpixel boundaries that cross the legs becomes weaker as alignment is added and the shadow edge on the body becomes weaker as strength is added (red ellipses mark edges where the change is particularly visible).

describe each of these stages[3]. Next, we'll provide more details about spatiotemporal superpixel extraction, superpixel graph construction, as well as some post processing steps.

5.2 Superpixel Extraction

We begin by extracting superpixels from every frame using the TurboPixels approach of Levinshtein et al. [34]. Instead of using the algorithm in its standard form, we modify it to obtain more temporally coherent superpixels. We start by extracting superpixels in the first frame using the original form of the superpixel algorithm in [34]. Instead of reseeding the superpixels in the next frame on a regular grid, we use the current frame's superpixels to drive the seeding procedure. We first compute the optical flow using the Lucas-Kanade (LK) algorithm [37]. The LK algorithm returns the flow for every pixel in every frame, together with a confidence measure. For every superpixel, we compute a weighted average of the flow over all reliable pixels, where pixels that are closer to the superpixel centroid have larger weights. Superpixels with an insufficient number of reliably flowing pixels get a flow of $(0, 0)$. The result is a *superpixel flow*, with motion flow vector $\mathbf{V_i}$ for every superpixel i (Fig. 6).

Taking the superpixel flow for every superpixel, we project the center of each superpixel to the next frame according to the computed flow. These projected centers serve as the initial seeds for the superpixel evolution in the next frame. We repeat this process for all video frames, giving us a much more temporally stable

[3] See the Approach Overview section at `http://www.cs.toronto.edu/~babalex/SpatiotemporalClosure/supplementary_material.html` for a graphical overview of the method.

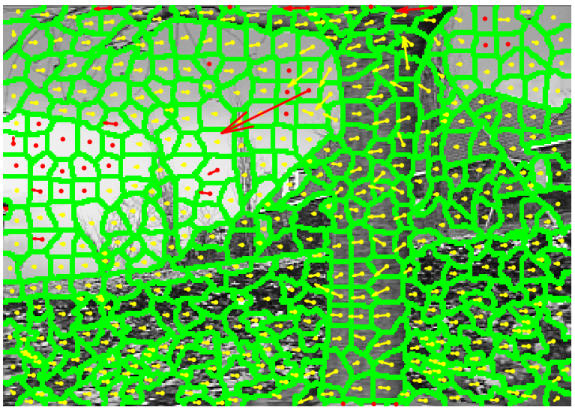


Fig. 6: Superpixel flow. The arrow within each superpixel indicates the motion flow vector of this superpixel. Yellow arrows indicate reliable flows, while red arrows correspond to unreliable flows.

superpixel segmentation. In addition, we also modify the superpixel algorithm to use a Pb-based [39] affinity rather than the original grayscale gradient-based affinity proposed in [34][4].

5.3 Superpixel Affinity

Once superpixels are extracted, we connect them using spatial and temporal edges. Every edge is assigned a weight W_{ij} that measures the similarity of the two superpixels (Fig. 2c). To form spatial connections, we find the immediate spatial neighbors of each superpixel in each frame. Spatial neighbors of superpixel i are defined as superpixels in the same frame that share some boundary with superpixel i. The formation of temporal connections follows the same approach as used in the superpixel extraction technique. Each superpixel in frame f (except the superpixels in the last frame) is connected to one superpixel in frame $f+1$. The correspondence is determined based on the superpixel flow vectors. The center of superpixel i from frame f is projected to frame $f+1$ according to the superpixel flow $\mathbf{V_i}$. We form an edge between superpixels i and j, where superpixel j is the superpixel in frame $f+1$ that contains the projected center of superpixel i.

Motivated by [23], our superpixel affinity W_{ij} for a spatial edge (i,j) is defined as the combination of appearance (W^a_{ij}) and motion (W^m_{ij}) affinities. The appearance affinity is obtained by computing the intersection of the grayscale histograms (or color, if available) histograms of the two superpixel regions (we use 30 bin histograms for grayscale and $4\times4\times4$ histograms for RGB). Motion affinity is computed using the flow vectors, $\mathbf{V_i}$ and $\mathbf{V_j}$, and is equal to $W^m_{ij} = 1 - \frac{\|\mathbf{V_i}-\mathbf{V_j}\|}{\max\{\|\mathbf{V_i}\|,\|\mathbf{V_j}\|\}}$ capped to the range $(0,1)$. Since our superpixel graph incorporates superpixel flow already, we include the motion affinity only for spatial edges. Finally, to give larger superpixels more influence, we augment the affinity by weighting it with the product of areas of the two superpixels (A_i and A_j). Combining that with the restriction of not grouping two superpixels if their appearance or motion is dissimilar, we obtain:

$$W_{ij} = \begin{cases} A_i A_j \min\left(W^a_{ij}, W^m_{ij}\right), & (i,j) \text{ are in the same frame} \\ A_i A_j W^a_{ij}, & (i,j) \text{ are in different frames} \end{cases} \quad (5)$$

Since our graph has edges for only a small spatial neighborhood of superpixels with edge affinities encoding both appearance and motion, we will refer to it as S-AM. In Section 6, we will compare this graph construction to other graphs with modified spatial connectivity and different superpixel affinities.

5.4 Optimal Cuts for Each Shot

At this point, we have a superpixel graph and apply the parametric maxflow framework to optimize the cost given in Eqn. 4. Prior to running the optimization framework, we first detect the shot boundaries in the video with the goal of independently finding closures for each shot.

Temporal superpixel edges across shot boundaries are unreliable. Thus if a video is composed of multiple shots, running the optimization on the whole video results in undesirable solutions. Since this is not the focus of this work, we take a very simplistic approach to shot boundary detection. Similar to the appearance affinity between superpixels, we compute an appearance affinity between consecutive frames by comparing the grayscale histograms of whole frames using the histogram intersection kernel. This results in a $F-1$ dimensional vector of consecutive frame affinities (where F is the number of frames). The shot boundaries correspond to the detected minima in this vector (Fig. 7). Given the detected shots, we build a subgraph for every shot by selecting the superpixels and the edges that are contained in the shot. We optimize the cost in Eqn. 4 for all the subgraphs and concatenate the results.

[4] See the Superpixel Extraction section at `http://www.cs.toronto.edu/~babalex/SpatiotemporalClosure/supplementary_material.html` for a better visualization of superpixel extraction and comparison with the original Turbopixels approach.

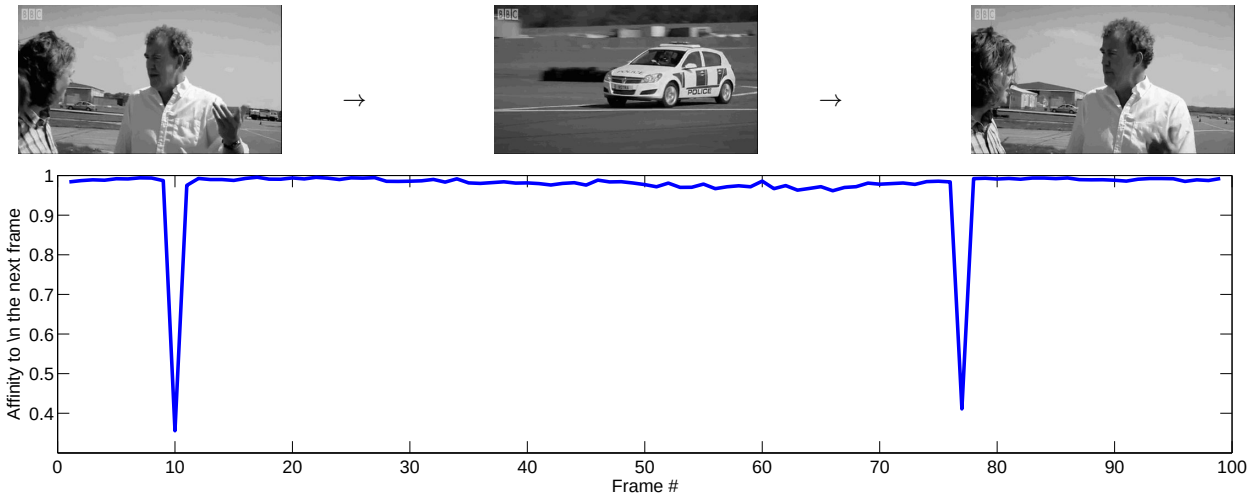


Fig. 7: Shot detection by finding minima in consecutive frame affinities. The top row shows a video containing 3 shots. The shot changes from people to car at frame 11 and back to people at frame 78. The bottom row shows a corresponding drop in consecutive frame affinity for these frames. These minima are detected in order to find the shot boundaries.

Note that as mentioned in section 3.1, optimizing the cost in Eqn. 4 directly results in a trivial solution where all the superpixels are selected for which $cut(\mathbf{X}) = 0$ and $volume(\mathbf{X}) > 0$, resulting in $C(\mathbf{X}) = 0$. Moreover, we want to be able to weaken affinities in order to handle the cases of potential bleeding between foreground and background due to appearance or motion similarity. We solve the first problem by introducing infinite penalties for a subset of superpixels in the graph, preventing the trivial solution. Specifically, for each superpixel i that we want to exclude from the selection, the term $D_i X_i$ in the numerator of Eqn. 4 is replaced with ∞. We run the optimization six times for each shot. In the first four runs, all the superpixels on the left, right, top, and bottom frame boundary respectively, are assigned an infinite penalty. In the two additional runs, we assign infinite penalties first to all top and bottom superpixels, and then to all left and right superpixels. To handle the second issue, we augment the closure affinity in Eqn. 5 to :

$$W'_{ij} = \begin{cases} A_i A_j \left(\min\left(W^a_{ij}, W^m_{ij}\right)\right)^\alpha, & (i,j) \text{ are in the same frame} \\ A_i A_j \left(W^a_{ij}\right)^\alpha, & (i,j) \text{ are in different frames} \end{cases} \tag{6}$$

The exponent α controls the contribution of weak affinities. Increasing the exponent effectively lowers all the affinities towards 0, thereby preventing bleeding, but also increases the relative difference between weak and strong affinities. In the results section, we will analyze the effect of changing α on performance and suggest an optimal value for α.

5.5 Post-processing

Running parametric maxflow on the spatiotemporal superpixel graph results in hundreds and sometimes thousands of breakpoints. Some of the solutions differ by a very minor increase in area, while others contain multiple connected components. Furthermore, there are cases where the closure costs of some desirable solutions are small but not minimal, causing such solutions to be missed. Since our goal is to yield a small number of spatiotemporal hypotheses that capture coherently moving objects in the scene, we post-process the results to narrow down the number of solutions to a smaller number and in the process generate additional good solutions. Post-processing consists of the following 3 stages:

1. **Filtering solutions and generating new ones by analyzing the area change:** As previously stated, parametric maxflow results in solutions that minimize the cut with increasing area constraints. Some solutions corresponding to consecutive breakpoints $(\lambda_i, \lambda_{i+1})$ are almost equivalent in their superpixel selections and differ by a very small increase in area. We filter out the solutions where such an increase is insignificant (less then 1% of relative area increase). Conversely, for all other solutions, we detect consecutive solution pairs where the relative area increase is above a threshold (more than 5%)

and generate a new solution subtracting one superpixel subset from another.

2. **Selecting connected components and removing small solutions:** Some solutions up to this point contain only a few superpixels or select superpixels in a very small number of frames. We filter out these solutions by keeping only the solutions with at least 2 superpixels, with total area that is at least 1% of the frame area, and that participate in at least 5 frames. We run a connected components analysis for all the remaining solutions. Each solution that contains multiple connected components in space-time is split, generating one solution for each connected component.
3. **Removing duplicate solutions:** The above post-processing steps can result in the generation of duplicate solutions. In this final step, we remove duplicate solutions.

For our test videos, this post-processing step reduces the number of solutions of a single run of parametric maxflow from several hundreds to an average of $20-80$ solutions.

6 Results

6.1 Image closure results

We compare our work, which we refer to as *superpixel closure*[5] (SC), to two other contour grouping methods: Estrada and Jepson (EJ) [20] and a version of ratio contours (RRC) from Stahl and Wang [52]. We provide a qualitative evaluation on various images (see Fig. 10), as well as quantitative evaluation on two datasets, the Weizmann Horse Database (WHD) [5] and the Weizmann Segmentation Database (WSD) [2]. Learning the gap measure (Section 4.2) is accomplished on the first 30 images from WHD. For testing, we use 170 additional images from WHD and all 100 images from WSD.

6.1.1 Quantitative Evaluation

For a quantitative evaluation of the results, we use the **F-measure**, $F = \frac{2RP}{R+P}$, where R and P are recall and precision, respectively, of the solution relative to the ground truth. Specifically, if A is the set of pixels corresponding to the solution and A_{gt} is the ground truth, then $R = \frac{|A \cap A_{gt}|}{|A_{gt}|}$ and $P = \frac{|A \cap A_{gt}|}{|A|}$. Given K solutions, the algorithm is said to perform well if the object's occluding boundary is well-captured by one of the proposed solutions. Thus, we select the solution with the best F-measure relative to the ground truth. We average the "per-image" F-measure for all the images (and three ground truth segmentations in WSD) in a dataset and report the result.

Fig. 8 shows the results of the three methods for increasing values of K. We chose the parameters that lead to the best performance for $K = 10$ for all three algorithms and fixed them for the entire experiment. For EJ, we used a Normalized Affinity Threshold (τ_{affty}) of 0.01, with the line segments generated by fitting the globalPb output. For RRC, we used $\lambda = 0$ and $\alpha = 1$. We have modified the RRC code to use globalPb-based line segments instead of the original Canny edge-based lines segments. For our method, we fixed the number of superpixels to 200 and set $T_e = 0.05$, giving us best performance at the high range of K. Since the resulting solutions can be thought of as shape hypotheses for object recognition, we believe that the performance for some reasonably small value of $K > 1$ is more important than obtaining a single best contour ($K = 1$)[6]. For $K = 10$, SC (EJ, RRC) obtains an average F-measure of 79.72% (77.23%, 69.86%) on WHD and 87.19%[7] (78.44%, 76.84%) on WSD.

We outperform the competing approaches on both datasets for a setting of $K = 10$ (obtaining a comparable performance to EJ on the horses dataset), which we attribute to the superpixel formulation, as well as the optimal closure finding method in our framework. On the WHD, both SC and RRC perform significantly worse than on WSD, while EJ performs similarly. This is likely due to the lower compactness of objects in the horse dataset (average isoperimetric ratio of 0.15, compared to 0.4 in WSD). Moreover, in many images there is a more compact path that includes the gap between the horse's legs due to shadow or ground edges. In addition, a significant number of images in the horse dataset have a picture frame boundary around the image. These boundaries provide the largest and most compact solutions, and are therefore found by SC instead of finding the horse. Interestingly, EJ still performs well on the horse dataset (unlike SC and RRC). This is most likely due to its reliance on internal appearance, which is definitely homogeneous for horses. Since T_e is set so low ($T_e = 0.05$), we detect many small structures, capturing texture elements or object parts whose closure cost is lower than that of the actual figure object. As a result,

[5] Matlab code for our method is available at `http://www.cs.toronto.edu/~babalex/closure_code.tgz`.

[6] SC can be tuned (see Fig. 9) to perform better for $K = 1$ at a small expense of performance for higher K's.

[7] For WSD, there are three ground truth segmentations per image. If we instead choose the closest of the three ground truth segmentations per image (as opposed to taking the average), our score on WSD improves to 88.76%.

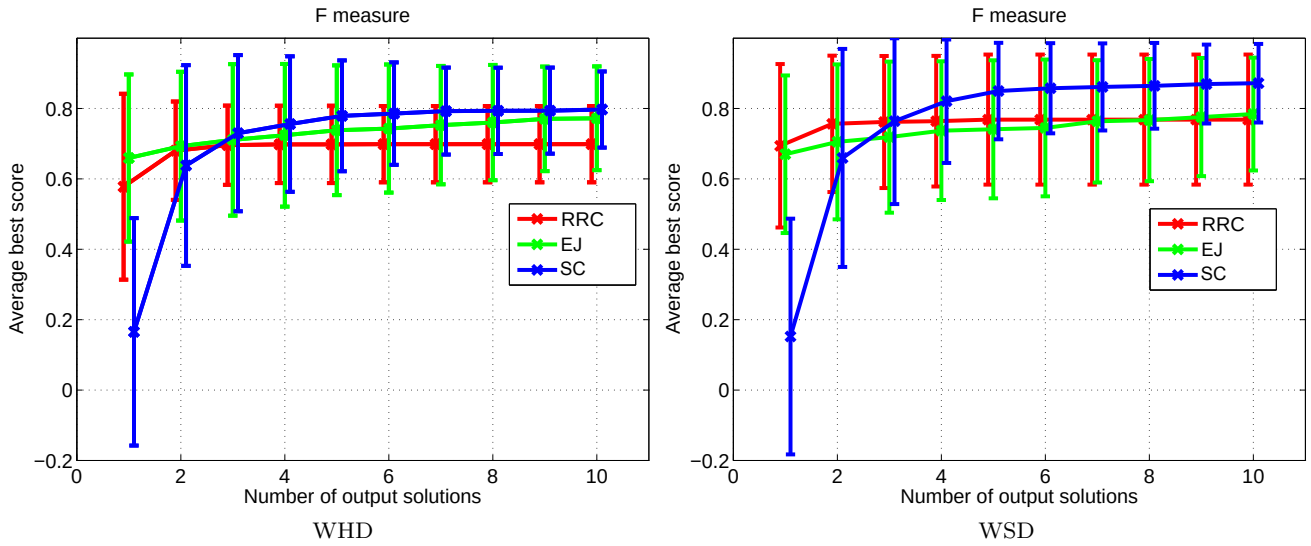


Fig. 8: Quantitative results. We compare our results (SC) to two other algorithms: Estrada and Jepson [20] (EJ) and Ratio Contours [52] (RRC). For different number of output solutions K, we show the average (across ground truth segmentations) best-out-of-K F-measure with standard error bars.

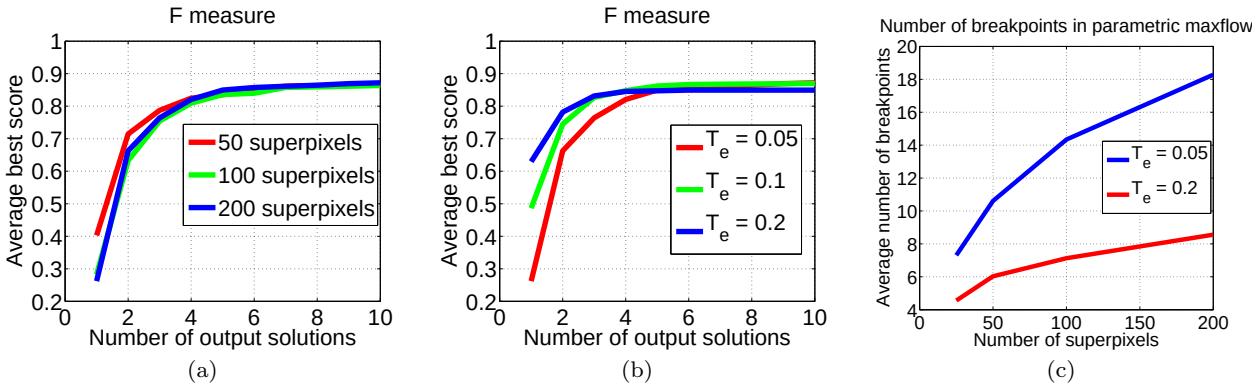


Fig. 9: Effects of varying the parameters of our method (evaluation on WSD). (a) Varying the number of superpixels for a fixed edge threshold $T_e = 0.05$. (b) Varying the edge threshold T_e for a fixed number of superpixels (200). (c) Number of breakpoints as a function of superpixel density and edge threshold T_e.

our performance is poor for low values of K, but it is better in the regime of high K values.

Fig. 9 shows the performance of our algorithm as we change the number of superpixels and vary T_e. Note that our dataset contains mostly large objects with relatively strong boundary support. Therefore, coarse superpixel resolutions, which prevent the detection of small structures, lead to good performance for low values of K (Fig. 9(a)). In general, higher superpixel density results in a marginal performance gain for large values of K. Similar effects are observed when changing the threshold T_e, since objects in our dataset typically have stronger edge support compared to other structures. Increasing the threshold T_e (Fig. 9(b)) reduces the detection of small objects and improves performance in the low range of K. However, it hurts the detection of objects with weak edges and results in slightly poorer performance at the high range of K. Nevertheless, in cases of large figure objects with strong boundary edges, coarser superpixel resolutions and higher edge thresholds are preferable. Given the above analysis, we cannot recommend a single good setting for all possible scenarios. In a qualitative sense, the number of superpixels should be as large as possible so as to not undersegment the objects of interest. This would ensure that the closure can still pick the correct path and at the same time eliminate some of the smaller structures from consideration. Similarly, the edginess threshold, T_e, should be set according to the expected contrast of object edges in the image.

The complexity of our approach is directly proportional to the number of breakpoints returned by parametric maxflow. Fig. 9(c) shows how the number of breakpoints varies as a function of superpixel resolution and edge threshold. We also compare the running times of the three methods on WSD (average image size of 300×290 pixels). On a 2.6GHz Dual Core Intel CPU with 4GB of memory, setting the methods to retrieve $K = 10$ best contours, the average running times per image are: SC (not including globalPb edge detection and superpixel segmentation) – 1.3 sec, EJ (not including globalPb edge detection) – 23 sec, RRC – 59 sec.

6.1.2 Qualitative Evaluation

In addition to the quantitative evaluation, we also provide a qualitative evaluation of our method by testing it on images from the two datasets, as well as other images obtained from the internet. Fig. 10 illustrates the performance of our method compared to the two competing approaches[8]. We manually select the best among 10 solutions for each method. Notice that detected contours in our framework lie closer to the true object contours since the superpixel boundaries, even in the presence of a gap, lie closer to object edges than the linearized contours detected by the other algorithms.

We also observe that our framework is not constrained to produce compact solutions as is usually the case when one is normalizing perimeter by area. This is clearly visible in the image of a spider, where very thin legs are segmented, representing the best closure solution. However, this is not always the case, for if there is a more compact contour that is not losing on gap, it will be preferred. This is why the gap is filled between the horse's legs, as well as the carriage's wheels, in the first two images. Note that for the horse image, EJ obtains a better solution by relying on the homogeneous appearance inside the horse.

Our method relies on superpixels to oversegment the object. This might not be the case for thin structures or when weak object contours are present. We still detect thin structures, such as the spider's legs, if good superpixels were found due to strong image edges. For weaker edges, however, thin structures are harder to capture (the bat of the baseball player, for example). Weak edges are also the cause of bleeding seen in the elephant example, where the upper portion of the front leg has weak edge contrast w.r.t. the background.

[8] Supplementary material (`http://www.cs.toronto.edu/~babalex/closure_supplementary.tgz`) contains the results of our algorithm for all the images in both datasets.

Fig. 10: Qualitative results. We compare our results (left) to two other algorithms: Estrada and Jepson [20] (middle) and Ratio Contours [52] (right).

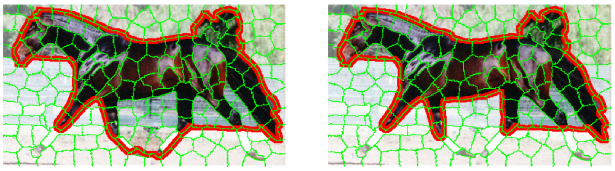


Fig. 11: Using internal appearance homogeneity. For objects with strong internal homogeneity of appearance, optimizing the cost in Eqn. 7 results in better performance (right) than optimizing the cost in Eqn. 3 (left). Note that the gap between the horse's legs was not included on the right due to its heterogeneous appearance w.r.t. the rest of the horse.

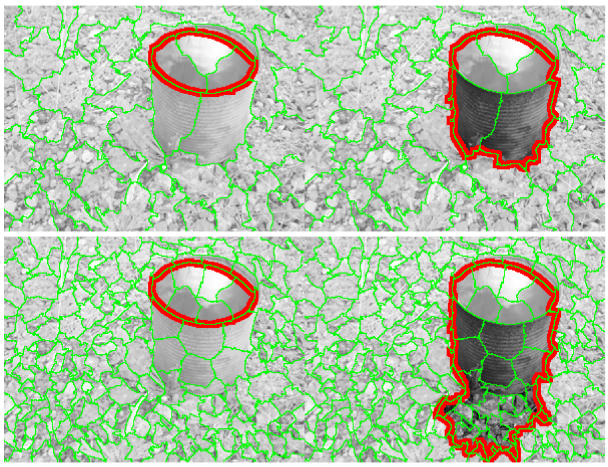


Fig. 12: Multiscale results. Choosing the $K = 2$ top solutions yields better results in the case of 50 superpixels (top) than in the case of 200 superpixels (bottom).

6.2 Extensions

6.2.1 Using internal homogeneity

As mentioned in Section 1, our superpixel formulation also facilitates the incorporation of appearance information, when it is both available and appropriate. The cost function in Eqn. 3 can be easily modified to incorporate a term which reflects the degree to which adjacent superpixels *inside* the selection, i.e., inside the closed contour, have high affinity. Assuming we are given an affinity matrix W, such that W_{ij} is the appearance similarity between two superpixels i and j, we can define our closure cost to be:

$$C_{affty}(\mathbf{X}) = \frac{\sum_i G_i X_i - 2\sum_{i<j} G_{ij} X_i X_j}{\sum_{i<j} W_{ij} X_i X_j} \qquad (7)$$

Compared to the cost in Eqn. 3, the numerator remains the same whereas the denominator changes to an internal homogeneity measure instead of the total object area. Minimizing this ratio results in minimizing the gap while maximizing the total affinity between the selected superpixels. Fig. 11 shows an example where better results were achieved by exploiting appearance homogeneity. Note, however, that the above is purely a local measure of appearance homogeneity. Unlike the method of Estrada and Jepson [20], we are currently unable to incorporate a global homogeneity measure in our cost. While the above local homogeneity-based closure cost helps to improve results in some cases, as the figure suggests, our tests have shown that in most cases there was no significant difference compared to our original closure cost in Eqn. 3.

6.2.2 Multiple superpixel scales

Though it might first seem that the more superpixels we use, the better our method will perform, it is not always so. As seen in Fig. 9(a), coarser superpixel scales constrain the solution more and thus perform better for low values of K. However, there is one additional advantage of using coarser superpixel scales. Since our superpixel algorithm does not produce hierarchical superpixels (since new superpixel boundaries may be introduced from finer to coarser scales), it is possible to encounter a lower degree of undersegmentation at coarser scales. Fig. 12 illustrates a situation where an object was segmented better at a coarser scale and consequently detected by our algorithm.

We tried a simple multiscale version of our algorithm where we merge the results from all scales. Specifically, we use four superpixel scales, obtaining 25, 50, 100, and 200 superpixels for each image. Setting $K = 10$ for each scale results in 40 solutions once the results are merged. Since the performance of our method for a given scale does not significantly vary for $K > 10$, we do not select 10 of 40 solutions for the multiscale version, but instead retain all 40. Using the multiscale version increases the performance on WSD from 87.19% to 89.53%.

6.3 Spatiotemporal closure results

To evaluate spatiotemporal closure, we first perform a qualitative analysis of our approach on several short video sequences. Some sequences (such as the flower garden sequence) are grayscale, while others contain color. In the case of color sequences, we make use of this additional information, comparing color histograms instead of grayscale when computing superpixel affinities. The frame size for each video is on the order of 300×300 pixels, with the length of a video ranging

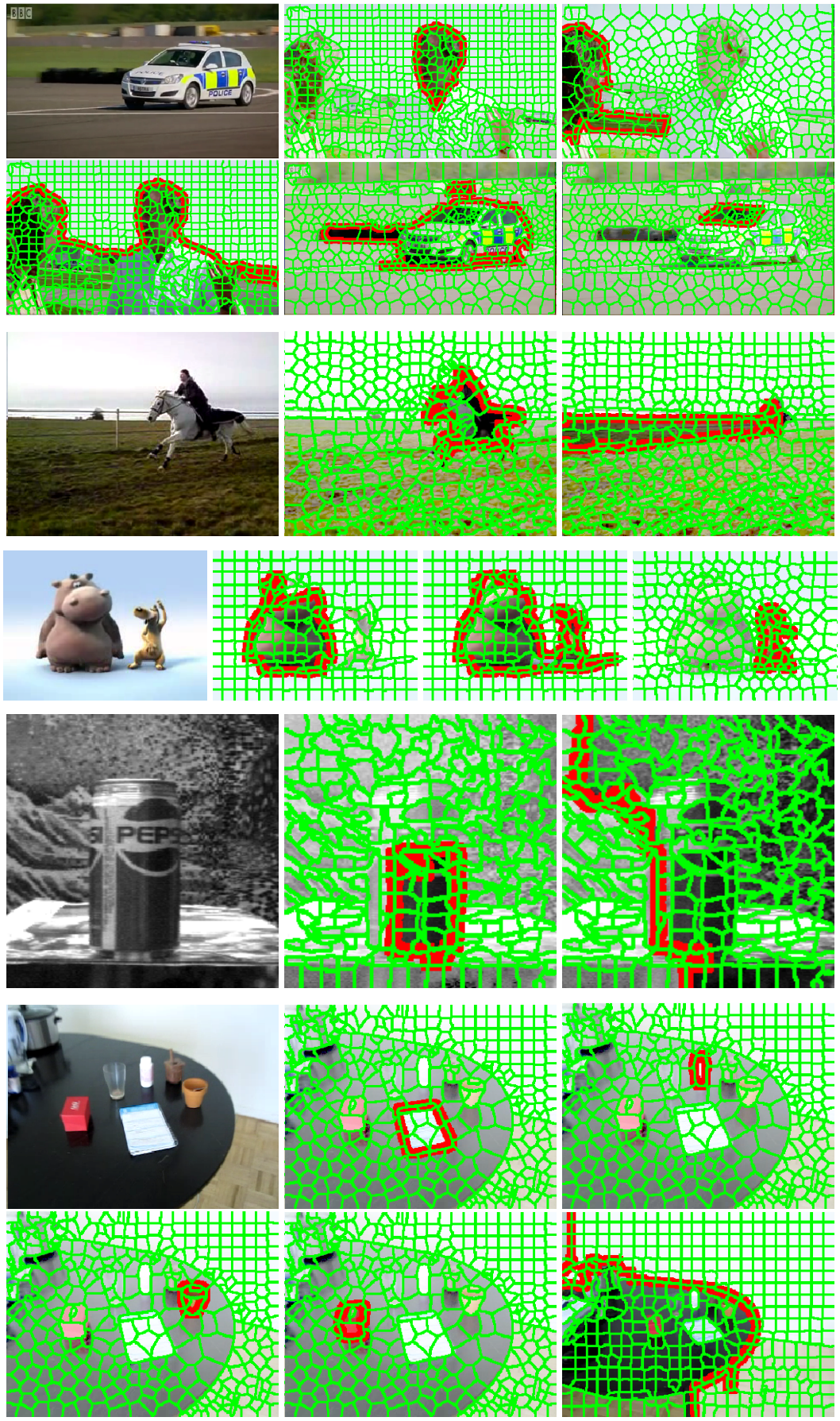


Fig. 13: Qualitative video figure/ground segmentation results. We display one sample frame from a sequence, followed by several interesting solutions.

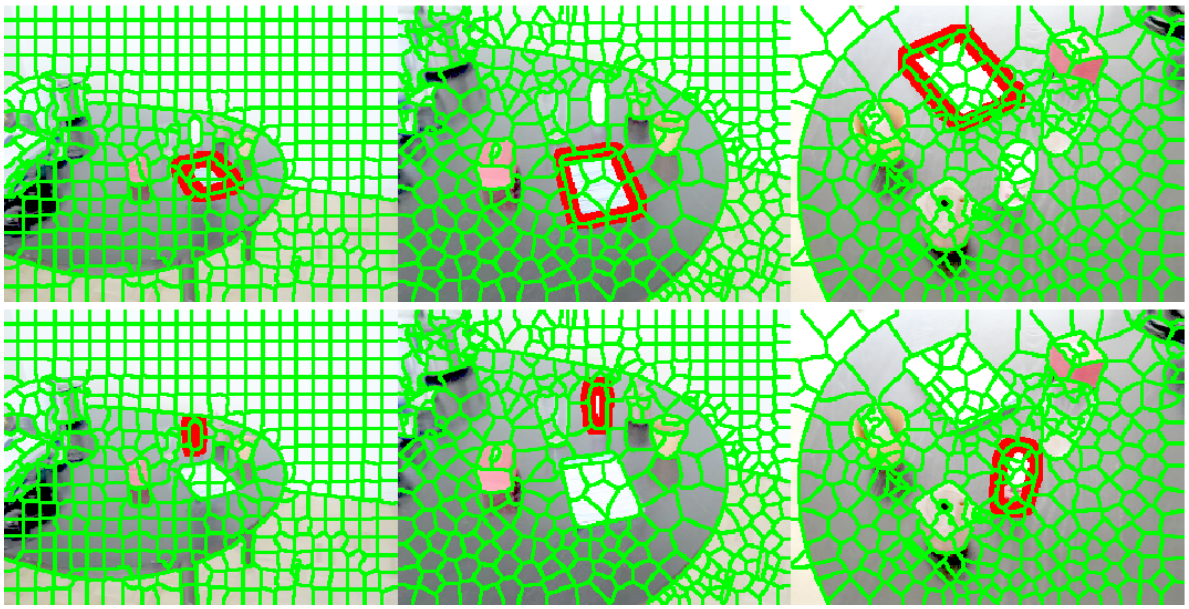


Fig. 14: Qualitative video figure/ground segmentation results illustrating two solutions from Fig. 13 over multiple frames.

from around 10 frames to 250 frames (hippo sequence). Based on quantitative evaluation (described next), we set $\alpha = 6$ for the qualitative experiments. We also perform a quantitative evaluation on a test dataset [54], where we compare different graph constructions and affinity variations, and evaluate our approach against standard normalized cuts on the same graphs. The computational bottlenecks of the approach are the preprocessing steps: Pb edge detection, superpixel extraction, and optical flow computation, each taking several seconds per frame. Once the superpixel graph is built, each run of the optimization using parametric maxflow finishes in less than 5 seconds on the entire video, followed by all the post-processing steps taking approximately 1 second.

Fig. 13 shows our qualitative results. For each sequence we show a frame from the original video and visualize several interesting solutions[9]. In the car sequence, several objects of interest were successfully recovered, such as the car and the heads of the people. Moreover, a part of the car (windshield) is also recovered in one of the solutions, indicating that if object parts exhibit good closure, our method can be used for part-based object recognition in videos or for action recognition that requires tracking parts. In the galloping horse sequence, the horse was correctly recovered in the middle of the sequence. A fence is also discovered as one of the solutions. However, in the beginning of the sequence it is partially merged with the horse due to poor superpixel boundaries and weak affinities between the horse and the background. This is also the reason for the incomplete solution in the Pepsi sequence. The horse example also illustrates that our framework works best for large objects, as small objects usually have higher closure cost and tend to be undersegmented by superpixels. The table sequence illustrates that our framework can detect most objects in the scene. Finally, the hippo sequence illustrates how an additional solution (dog) can be generated by subtracting one solution (hippo) from another (hippo and dog). Fig. 14 illustrates some of the solutions from the last sequence of Fig. 13 over multiple frames. The reader is referred to supplementary material for video visualization of the results.

For quantitative evaluation, we use 27 sequences from the dataset of Stein et al. [54]. Each sequence has a ground truth video segmentation mask, marking one foreground object. Given a set of detected spatiotemporal figures for a sequence, we choose the solution with the maximal F measure ($\frac{2 \cdot Precision \cdot Recall}{Precision + Recall}$) relative to the ground truth. We report the average F measure across all sequences.

We compare different variations of our algorithm, as well as replace our parametric maxflow minimization of the unbalanced normalized cuts cost with standard normalized cuts. Unlike our method, normalized cuts requires the user to specify the number of clus-

[9] See the Results at `http://www.cs.toronto.edu/~babalex/SpatiotemporalClosure/supplementary_material.html` for a video visualization of the results.

ters. Therefore, to compare with our approach we run normalized cuts with 5, 10, 15, 20, and 25 clusters and concatenate all the results. Recall that our previously described graph construction (S-AM) includes only the immediate spatial neighbors and adds the motion affinity W_{ij}^m for spatial edges. We define additional variations over this construction:

- S-A - Same graph as S-AM, but with affinity only including **appearance** $W_{ij} = A_i A_j \left(W_{ij}^a\right)^\alpha$.
- L-AM - Same as S-AM but with **larger** spatial connectivity. In addition to the edges in S-AM, we add edges between all superpixels in the same frame whose centroids are less than R apart, where R is five times the radius of an average superpixel.
- L-A - Same as L-AM, but with affinity only including appearance $W_{ij} = A_i A_j \left(W_{ij}^a\right)^\alpha$.

We compare our method (SC) to normalized cuts (NCuts) for all the above graph constructions. While we are able to solve the unbalanced normalized cuts problem in a globally optimal fashion, normalized cuts cost is NP-hard to optimize and therefore only an approximation is provided. Despite that, the cut balancing in NCuts makes the solutions compact and helps to avoid bleeding, while our closure cost pushes the solutions to contain more superpixels which may result in undersegmentation. Fig. 15 illustrates the performance as we vary α. We also observe that our method achieves comparable results using S-AM and L-AM, indicating that our increase of spatial connectivity has only a marginal effect on performance. Note that the video sequences in the test dataset mostly contain large objects. Thus undersegmentation, as a result of incorrect superpixels or our unbalanced normalized cuts closure cost, is less of a concern, resulting in SC outperforming the standard NCuts.

7 Limitations and Future Work

We have presented a closure detection method in 2D images as well as spatiotemporal domains. Relying on compactness allows us to use closure to group large subsets of superpixels in a purely bottom-up fashion. However, while our closure cost is shown to be effective, it is by no means "correct" in the sense that its minima must correspond to real objects. Since it uses a purely bottom-up closure cue for grouping, it should be thought of as a proposal mechanism rather than a decision rule. Additional post-processing of the optimal closure solutions could potentially improve the results despite the fact that the post-processed solutions are no longer guaranteed to have optimal closure. Ultimately, top-down approaches are probably necessary to correctly recover the objects.

Some limitations of our cost can be seen in performance on non-compact objects, such as a horse or fast-moving objects that are temporally non-compact. Moreover, both the image cost and the spatiotemporal cost give preference to objects with larger area. This feature enables us to effectively form very large groups of superpixels bottom-up, but hinders performance for small objects.

Finally, our spatiotemporal cost includes only local constraints for temporal coherence. Tracking approaches that maintain a global shape and/or appearance model of the target can improve results. That said, both our 2D and spatiotemporal costs could be augmented to include global shape or appearance constraints. This would allow us to handle more global effects, and segment objects in the presence of occlusion. We plan to explore this extension in future work.

Perhaps our most significant contribution is the use of superpixels and redefining closure detection as finding groups of superpixels instead of grouping edgels. This reformulation decreases the complexity of the problem and provides a better context for feature extraction. Unfortunately, errors in superpixel segmentation propagate throughout our framework. Superpixel bleeding prevents the accurate segmentation of figure from ground, but an even larger issue is that it provides incorrect scope for feature extraction, thereby weakening gaps or strengthening affinities between figure and ground. In future work, we plan to explore additional cues for gap and affinity computation to strengthen the robustness of our approach to weak object edges or similar foreground/background motion. For example, we plan to use superpixel junctions to learn an affinity measure between pairs of superpixels that are both inside and adjacent to the boundary. Such an affinity measure can encode a learned measure of continuity and T-junction, and could significantly strengthen our cost function.

Ultimately, our goal is to detect closure in cases such as the Kanizsa triangle, where long range closure reasoning is needed, or detect spatiotemporal closures in the presence of large occlusions. Currently this is beyond the capabilities of our framework and only objects with a relatively small amount of gap, whether spatial or temporal, can be detected. However, we believe that we can approach this goal through the use of more global shape cues in our closure cost and multiscale superpixel segmentations. Coarser superpixels will be more suitable for capturing large gaps.

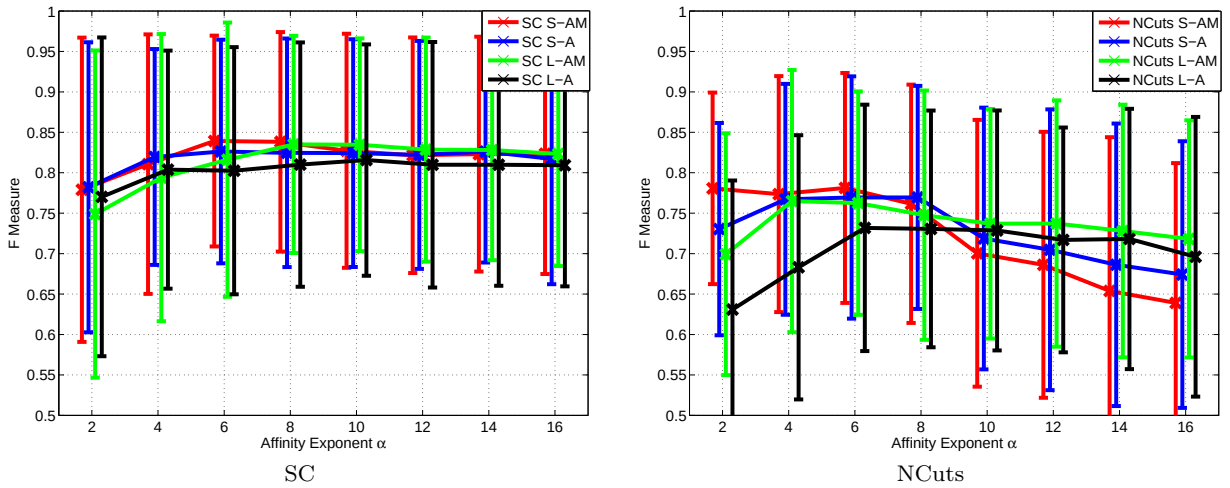


Fig. 15: Quantitative evaluation of spatiotemporal closure detection. We compare the performance of each method (SC on the left and NCuts on the right) on four different graph constructions. We show the F-measure with standard error bars as a function of the affinity exponent α.

8 Conclusions

We present a generic detection method for contour closure and spatiotemporal closure that is applicable to a variety of closure detection scenarios. Our reformulation of the problem of finding cycles of contours in images as the problem of finding spatially coherent subsets of superpixels, whose collective external boundary has strong image edge evidence, yields an optimal framework for closure detection that compares favorably with two prior leading approaches. The same holds for our spatiotemporal closure detection framework, where our method was adjusted to recover coherent spatiotemporal segments that are separated from the background by strong appearance and motion discontinuities. In contrast to competing approaches that focus on the detection of a single, best, closed contour, our optimization framework generates a small number of solutions that can be thought of as promising object hypotheses, better serving high-level recognition tasks.

While superpixels provide an ideal scope for learning a gap measure from training data, they offer a number of additional advantages that we are currently exploring. In an extension to the main 2D closure detection approach, we show that superpixels also provide a convenient mechanism for incorporating appearance information, if appropriate and if available. For example, if the object was known to be homogeneous in appearance, our modified cost function can easily incorporate such a prior, as discussed in Section 6.2.1. We also provide a simple approach for using multiscale superpixel information, with the plan of pursuing a more elegant coarse-to-fine framework for finding contour closure using multiple superpixel scales. Finally, for our spatiotemporal closure cost, superpixels play an even greater role, facilitating stable appearance and motion-based affinity computation. To conclude, our closure detection method efficiently recovers a small number of 2D and spatiotemporal figure/ground hypotheses and paves the way to better solutions of high-level vision problems.

Acknowledgements We thank David Fleet and Allan Jepson for discussion about closure cost functions and optimization procedures, James Elder for providing valuable advice, and Yuri Boykov and Vladimir Kolmogorov for providing their parametric maxow implementation. This research was sponsored in part by the Army Research Laboratory and was accomplished under Cooperative Agreement Number W911NF-10-2-0060. The views and conclusions contained in this document are those of the authors and should not be interpreted as representing the official policies, either express or implied, of the Army Research Laboratory or the U.S. Government. The U.S. Government is authorized to reproduce and distribute reprints for Government purposes, notwithstanding any copyright notation herein. This work was also supported by the European Commission under a Marie Curie Excellence Grant MCEXT-025481 (Cristian Sminchisescu), CNCSIS-UEFISCU under project number PN II- RU-RC-2/2009 (Cristian Sminchisescu), CNCSIS-UEFISCU under project number PN II-RU-RC-2/2009 (Cristian Sminchisescu), NSERC (Alex Levinshtein, Sven Dickinson), MITACs (Alex Levinshtein).

References

1. A region-level motion-based graph representation and labeling for tracking a spatial image partition. Pattern Recognition **33**(4), 725 – 740 (2000)
2. Alpert, S., Galun, M., Basri, R., Brandt, A.: Image segmentation by probabilistic bottom-up aggregation and

cue integration. In: IEEE International Conference on Computer Vision and Pattern Recognition (2007)
3. Bascle, B., Deriche, R.: Region tracking through image sequences. ICCV **0**, 302 (1995)
4. Black, M., Jepson, A.: Eigentracking: Robust matching and tracking of articulated objects using a view-based representation. IJCV **26**(1), 63–84 (1998)
5. Borenstein, E., Ullman, S.: Class-specific, top-down segmentation. In: European Conference on Computer Vision, pp. 109–124 (2002)
6. Boykov, Y.Y., Jolly, M.P.: Interactive graph cuts for optimal boundary and region segmentation of objects in n-d images (2001)
7. Brady, M., Asada, H.: Smoothed local symmetries and their implementation. International Journal of Robotics Research **3**(3), 36–61 (1984)
8. Carreira, J., Sminchisescu, C.: Constrained parametric min-cuts for automatic object segmentation. In: IEEE International Conference on Computer Vision and Pattern Recognition (2010)
9. Cham, T.J., Cipolla, R.: Geometric saliency of curve correspondences and grouping of symmetric contours. In: European Conference on Computer Vision, pp. 385–398 (1996)
10. Chung, D., MacLean, W., Dickinson, S.: Integrating region and boundary information for spatially coherent object tracking. IVC **24**(7), 680–692 (2006)
11. Comaniciu, D., Ramesh, V., Meer, P.: Kernel-based object tracking. PAMI **25**(5), 564–577 (2003)
12. Cremers, D., Soatto, S.: Motion competition: A variational approach to piecewise parametric motion segmentation. IJCV **62**(3) (2005)
13. DeMenthon, D.: Spatio-temporal segmentation of video by hierarchical mean shift analysis. In: SMVP (2002)
14. Deng, Y., Manjunath, B.: Unsupervised segmentation of color-texture regions in images and video. PAMI **23**, 800–810 (2001)
15. Dinkelbach, W.: On nonlinear fractional programming. Management Science **13**, 492–498 (1967)
16. Elder, J., Zucker, S.: A measure of closure. Vision Research **34**, 3361–3369 (1994)
17. Elder, J.H., Zucker, S.W.: Computing contour closure. In: European Conference on Computer Vision, pp. 399–412 (1996)
18. Endres, I., Hoiem, D.: Category independent object proposals. In: ECCV, pp. 575–588 (2010)
19. Estrada, F.J., Jepson, A.D.: Perceptual grouping for contour extraction. In: IEEE International Conference on Pattern Recognition, pp. 32–35 (2004)
20. Estrada, F.J., Jepson, A.D.: Robust boundary detection with adaptive grouping. In: Computer Vision and Pattern Recognition Workshop, p. 184 (2006)
21. Fowlkes, C., Belongie, S., Malik, J.: Efficient spatiotemporal grouping using the nyström method. In: CVPR, pp. 231–238 (2001)
22. Greenspan, H., Goldberger, J., Mayer, A.: Probabilistic space-time video modeling via piecewise gmm. PAMI **26**(3), 384–396 (2004)
23. Huang, Y., Liu, Q., Metaxas, D.: Video object segmentation by hypergraph cut. IEEE International Conference on Computer Vision and Pattern Recognition **0**, 1738–1745 (2009)
24. Isard, M., Blake, A.: Condensation - conditional density propagation for visual tracking. IJCV **29**, 5–28 (1998)
25. Jacobs, D.: Robust and efficient detection of salient convex groups. IEEE Transactions on Pattern Analysis and Machine Intelligence **18**(1), 23–37 (1996)
26. Jepson, A.D., Fleet, D.J., Black, M.J.: A layered motion representation with occlusion and compact spatial support. In: ECCV, pp. 692–706 (2002)
27. Jermyn, I., Ishikawa, H.: Globally optimal regions and boundaries as minimum ratio weight cycles. IEEE Transactions on Pattern Analysis and Machine Intelligence **23**, 1075–1088 (2001)
28. Jojic, N., Frey, B.J.: Learning flexible sprites in video layers. CVPR **1**, 199 (2001)
29. Kolmogorov, V., Boykov, Y., Rother, C.: Applications of parametric maxflow in computer vision. In: IEEE International Conference on Computer Vision, pp. 1–8 (2007)
30. Lempitsky, V., Kohli, P., Rother, C., Sharp, T.: Image segmentation with a bounding box prior (2009)
31. Levinshtein, A., Sminchisescu, C., Dickinson, S.: Multiscale symmetric part detection and grouping pp. 2162–2169 (2009)
32. Levinshtein, A., Sminchisescu, C., Dickinson, S.: Optimal contour closure by superpixel grouping. In: ECCV, pp. 480–493 (2010)
33. Levinshtein, A., Sminchisescu, C., Dickinson, S.: Spatiotemporal closure. In: ACCV, pp. 369–382 (2010)
34. Levinshtein, A., Stere, A., Kutulakos, K.N., Fleet, D.J., Dickinson, S.J., Siddiqi, K.: Turbopixels: Fast superpixels using geometric flows. PAMI **31**(12), 2290–2297 (2009)
35. Li, F., Carreira, J., Sminchisescu, C.: Object Recognition as Ranking Holistic Figure-Ground Hypotheses. In: CVPR (2010)
36. Lowe, D.G.: Perceptual Organization and Visual Recognition. Kluwer Academic Publishers, Norwell, MA, USA (1985)
37. Lucas, B.D., Kanade, T.: An iterative image registration technique with an application to stereo vision. In: Proceedings of Imaging Understanding Workshop, pp. 674–679 (1981)
38. Maire, M., Arbelaez, P., Fowlkes, C., Malik, J.: Using contours to detect and localize junctions in natural images. In: IEEE International Conference on Computer Vision and Pattern Recognition (2008)
39. Martin, D.R., Fowlkes, C.C., Malik, J.: Learning to detect natural image boundaries using local brightness, color, and texture cues. IEEE Transactions on Pattern Analysis and Machine Intelligence **26**, 530–549 (2004)
40. Megret, R., DeMenthon, D.: A survey of spatio-temporal grouping techniques. Tech. rep., University of Maryland, College Park (2002)
41. Mori, G., Ren, X., Efros, A.A., Malik, J.: Recovering human body configurations: Combining segmentation and recognition. In: IEEE International Conference on Computer Vision and Pattern Recognition, pp. 326–333 (2004)
42. Moscheni, F., Bhattacharjee, S., Kunt, M.: Spatiotemporal segmentation based on region merging. PAMI **20**(9), 897–915 (1998)
43. Paragios, N., Deriche, R.: Geodesic active contours and level sets for the detection and tracking of moving objects. PAMI **22**(3), 266–280 (2000)
44. Patras, I., Lagendijk, R.L., Hendriks, E.A.: Video segmentation by map labeling of watershed segments. PAMI **23**(3), 326–332 (2001)
45. Pritch, Y., Rav-Acha, A., Peleg, S.: Nonchronological video synopsis and indexing. PAMI **30**, 1971–1984 (2008)
46. Ren, X., Fowlkes, C.C., Malik, J.: Cue integration in figure/ground labeling. In: Advances in Neural Information Processing Systems (2005)

47. Ren, X., Fowlkes, C.C., Malik, J.: Scale-invariant contour completion using conditional random fields. In: IEEE International Conference on Computer Vision, vol. 2, pp. 1214–1221 (2005)
48. Rother, C., Kolmogorov, V., Blake, A.: "grabcut": interactive foreground extraction using iterated graph cuts. SIGGRAPH **23**(3), 309–314 (2004)
49. Saint-Marc, P., Rom, H., Medioni, G.: B-spline contour representation and symmetry detection. IEEE Transactions on Pattern Analysis and Machine Intelligence **15**(11), 1191–1197 (1993)
50. Shi, J., Malik, J.: Motion segmentation and tracking using normalized cuts. In: ICCV, p. 1154 (1998)
51. Shi, J., Malik, J.: Normalized cuts and image segmentation. IEEE Transactions on Pattern Analysis and Machine Intelligence **22**(8), 888–905 (2000)
52. Stahl, J., Wang, S.: Edge grouping combining boundary and region information. IEEE Transactions on Image Processing **16**(10), 2590–2606 (2007)
53. Stahl, J., Wang, S.: Globally optimal grouping for symmetric closed boundaries by combining boundary and region information. IEEE Transactions on Pattern Analysis and Machine Intelligence **30**(3), 395–411 (2008)
54. Stein, A., Hoiem, D., Hebert, M.: Learning to find object boundaries using motion cues. In: IEEE International Conference on Computer Vision, pp. 1–8 (2007)
55. Wang, D.: Unsupervised video segmentation based on watersheds and temporal tracking. CirSysVideo **8**(5), 539–546 (1998)
56. Wang, J., Adelson, E.: Representing moving images with layers. TIP **3**(5), 625–638 (1994)
57. Wang, S., Kubota, T., Siskind, J., Wang, J.: Salient closed boundary extraction with ratio contour. IEEE Transactions on Pattern Analysis and Machine Intelligence **27**(4), 546–561 (2005)
58. Weiss, Y.: Smoothness in layers: Motion segmentation using nonparametric mixture estimation. In: CVPR, p. 520 (1997)
59. Weiss, Y., Adelson, E.H.: A unified mixture framework for motion segmentation: Incorporating spatial coherence and estimating the number of models. In: CVPR, p. 321 (1996)
60. Welch, G., Bishop, G.: An introduction to the kalman filter. Tech. rep. (1995)
61. Wertheimer, M.: Laws of organization in perceptual forms. In: W. Ellis (ed.) Source Book of Gestalt Psychology. Harcourt, Brace, New York, NY (1938)
62. Williams, L.R., Hanson, A.R.: Perceptual completion of occluded surfaces. Computer Vision and Image Understanding **64**(1), 1–20 (1996)
63. Williams, L.R., Jacobs, D.W.: Stochastic completion fields: a neural model of illusory contour shape and salience. In: IEEE International Conference on Computer Vision, p. 408 (1995)
64. Ylä-Jääski, A., Ade, F.: Grouping symmetrical structures for object segmentation and description. Computer Vision and Image Understanding **63**(3), 399–417 (1996)
65. Zhu, Q., Song, G., Shi, J.: Untangling cycles for contour grouping. In: IEEE International Conference on Computer Vision (2007)