SIAM J. IMAGING SCIENCES
Vol. 10, No. 3, pp. 1475–1510

Theoretical Analysis of Active Contours on Graphs*

Christos Sakaridis†, Kimon Drakopoulos‡, and Petros Maragos§

Abstract. Active contour models based on partial differential equations have proved successful in image segmentation, yet the study of their formulation on arbitrary geometric graphs, which place no restrictions in the spatial configuration of samples, is still at an early stage. In this paper, we introduce geometric approximations of gradient and curvature on arbitrary graphs, which enable a straightforward extension of active contour models that are formulated through level sets to such general inputs. We prove convergence in probability of our gradient approximation to the true gradient value and derive an asymptotic upper bound for the error of this approximation for the class of random geometric graphs. Two different approaches for the approximation of curvature are presented, and both are also proved to converge in probability in the case of random geometric graphs. We propose neighborhood-based filtering on graphs to improve the accuracy of the aforementioned approximations and define two variants of Gaussian smoothing on graphs which include normalization in order to adapt to graph nonuniformities. The performance of our active contour framework on graphs is demonstrated in the segmentation of regular images and geographical data defined on arbitrary graphs, using geodesic active contours and active contours without edges as representative models in our experiments.

Key words. active contours, graph segmentation, random geometric graphs, image segmentation

AMS subject classifications. 35Q68, 35R02, 60D05, 65D25, 68U10

DOI. 10.1137/16M1100101

1. Introduction.

Evolution of curves via active contour models has been applied extensively in computer vision for image segmentation and object detection. In the standard image setting which involves a regular grid of pixels, the discretization of PDEs governing the motion of active contours is well-established and ensures proper convergence of the contour to object boundaries. Recently, active contours have been extended to handle more general inputs in the form of graphs whose vertices are arbitrarily distributed in a two-dimensional Euclidean space. In particular, the input in this case consists of

1. an undirected graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$, where $\mathcal{V} = \{\mathbf{v}_i \in \mathbb{R}^2 : i = 1, \dots, n\}$ and $\mathcal{E} \subseteq \mathcal{V} \times \mathcal{V}$, and
2. a real-valued function $I : \mathcal{V} \to \mathbb{R}$ defined on the vertices of the graph, which resembles

*Received by the editors October 24, 2016; accepted for publication (in revised form) June 26, 2017; published electronically September 7, 2017. This work was primarily performed while the first author was at National Technical University of Athens.

http://www.siam.org/journals/siims/10-3/M110010.html

Funding: The research of the third author was partially supported by the European Union under the projects I-SUPPORT with grant H2020-643666 and BabyRobot with grant H2020-687831.

†Department of Information Technology and Electrical Engineering, ETH Zürich, 8092 Zürich, Switzerland (csakarid@vision.ee.ethz.ch).

‡Marshall School of Business, University of Southern California, Los Angeles, CA 90089-0809 (drakopou@marshall.usc.edu).

§School of Electrical and Computer Engineering, National Technical University of Athens, 15773 Athens, Greece (maragos@cs.ntua.gr).

the image function in the standard grid setting. For the rest of the paper, we will use the term *image function* to refer to I.

This arbitrary spatial configuration poses a significant challenge to the approximation of continuous operators that are essential for active contour models using level sets. Applications of segmentation of such graphs span not only image processing, but also geographical information systems and generally any field where data can assume the form of a set of pointwise samples of a real-valued function.

Our work focuses mainly on the theoretical study of fundamental geometric terms in active contours, primarily *gradient* and *curvature*, and the introduction of novel, neighborhood-based approximations of them on arbitrary graphs, which improve upon previous approaches. We prove the *consistency* of our approximations in the limit of infinite vertices for the class of *random geometric graphs* [38]. Additionally, we derive an asymptotic bound for the error of our gradient approximation with respect to the number of vertices of the graph. Another important contribution is the usage of neighborhood-based smoothing filtering on graphs as an empirical method to reduce the error of our approximations. Last, we propose normalized versions of Gaussian filtering—which is essential for initialization of several active contour schemes—on graphs, suited to handle nonuniform vertex spacing.

The paper is structured as follows. Section 2 reviews previous work on active contours, graph-based morphology and segmentation, and PDE-based methods on graphs and provides the necessary background on active contour models. In section 3 we introduce the basic quantities of our framework and present our geometric approximation of gradient on arbitrary graphs. We provide conditions for convergence in probability of our approximation in the case of random geometric graphs and analyze the asymptotic behavior of the approximation error, which enables an advised selection of parameters for graph construction. In section 4 we give two methods to approximate curvature on graphs and state theorems about their convergence in probability for random geometric graphs. Section 5 is dedicated to defining neighborhood-based smoothing filters on graphs, introducing normalized Gaussian filtering and Gaussian derivative filtering on graphs, and demonstrating their use in smoothing synthetic gradient, curvature, or image functions. In section 6 we equip two exemplar active contour models, namely geodesic active contours [7] and active contours without edges [9], with our approximations of gradient and curvature and apply these models to segment arbitrary graphs defined from regular images or containing geographical data. We assess different methods to create the set of vertices and/or edges of these graphs and compare our approach with a concurrent work on active contours on graphs based on finite elements [29]. It should be noted that due to space limitations, a lengthy proof of convergence in probability for our first method for curvature approximation is deferred to the accompanying supplemental material file M110010_01.pdf [local/web 254KB]. For the same reason, some additional results for smoothing filtering and segmentation of synthetic data are also included in M110010_01.pdf [local/web 254KB].

2. Background and related work.

Active contour models for curve evolution toward image edges originate from "snakes" [28]. These early approaches could not, in general, handle topological changes of the contour, for instance, splitting into two disjoint parts to detect the boundaries of two distinct objects. PDE-based methods using level sets were proposed as an

alternative in [6, 7], where the geometric active contour model was initially introduced and subsequently complemented to establish the geodesic active contour (GAC) framework. The former model involves two forces that govern curve motion: a balloon force that expands or shrinks it, and a curvature-dependent force that maintains its smoothness. The latter model adds an extra spring force that attracts the contour toward salient image edges. Both methods embed the active contour as a *level set* of function u, which is termed the embedding function and is the unknown in the PDE that models curve evolution, allowing the use of numerical schemes of the type proposed in [37]. The curvature-dependent force is also used for regularization in active contours without edges (ACWE) [9], which abandon the aforementioned edge-driven paradigm and leverage the Mumford–Shah functional [36] to formulate a piecewise constant segmentation model, with automatic detection of interior contours and reduced sensitivity to initialization. This model is further studied in [8], where the authors prove that the global minimizers of the original nonconvex problem can be recovered via solving a convex reformulation of it.

Graphs have long been connected to image processing, in part through their study in terms of mathematical morphology. The application of morphological transforms on neighborhood graphs was established in [44], while a wide variety of graph structures, algorithms for their construction, and early applications in computer vision were surveyed in [27]. The notion of structuring element in classical morphology was extended to graphs in [25], where the proposed structuring graph enables a generalization of neighborhood functions on a graph beyond the one induced by its set of edges. Morphological operators on graphs have been studied further in [15], where the lattice of the subgraphs of a graph is considered in order to define filters that treat the graph as a whole.

Recently, several works, including [42, 32, 31, 17, 16, 20, 18], have focused on the construction of PDE-based rather than algebraically defined morphological operators on graphs, which are then used to define active contour models on graphs. All these works are based on the definition of a gradient operator on graphs. However, [42, 32, 31, 16, 20, 18] work on weighted graphs, and all define a *discrete* gradient at a vertex as a vector whose dimensionality is the same as the cardinality of the vertex's neighborhood. This type of gradient is a difference operator, which *replaces* the original continuous gradient operator and enables the adaptation of continuous PDE schemes on graphs by defining analogous partial difference equations. This approach sacrifices consistency of the resulting scheme with the original, continuous PDE formulation, as it removes the link between the discrete graph domain and the underlying continuum. Its foundation lies in the theory of *discrete calculus* [24, 26], which provides a framework for reformulating continuous energies on graphs such that the corresponding solutions share similar properties [10, 12, 33, 23, 3, 19]. On the other hand, the authors of [17] consider unweighted graphs and *approximate* the original, continuous gradient at each vertex. Their approach effectively constitutes a generalized numerical scheme for solving the original, continuous active contour PDE on graphs that have arbitrary spatial vertex configuration and hence greater complexity than the standard 2D image grid, as was described in section 1. We follow the line of [17] and carefully treat the geometric quantities involved in active contour models, such as gradient and curvature. Our aim is to establish approximations of these operators on graphs which are *asymptotically consistent* in the limit of large, dense geometric graphs in order to ensure a stable *discretization* of level set–based

active contour models on arbitrary 2D graphs, as has been previously achieved for the 2D image grid. In particular, to the best of our knowledge, the asymptotic upper bound for the error of our gradient approximation for random geometric graphs is the first of its kind.

A different class of approaches to graph segmentation which has gained a lot of interest in the image processing community is based on graph cuts. These approaches, in contrast to ours, usually operate on a regular image grid and define weighted edges between image pixels based on certain cues, such as spatial or appearance proximity, in order to find a cut of minimal cost for the resulting weighted graph. The cost of a cut is normalized in [40] so that balanced partitions are preferred. Approximate solutions to multilabel problems are proposed in [5], guaranteeing constant-factor optimality. A link between geodesic active contours and graph cuts is established in [4], where the graph is constructed so that the cost of the cut corresponds to the contour's length under the induced anisotropic metric, and this link is extended to the arbitrary graph setting in [17]. The random walk algorithm of [22] assigns unlabeled pixels to user-defined seeds interactively. Efficient algorithms for watershed-like segmentation that are formulated as graph cuts are introduced in [13, 14]. The power watershed framework of [11] unites and generalizes several graph-based optimization methods for image segmentation by expressing their energies in a common, parametric form.

The analysis of consistency of algorithms operating on point clouds, or graphs which are constructed from point clouds, has received a lot of attention, especially for machine learning tasks such as clustering. The focus of this line of research is on proving that optimizers of functionals which are defined on the discrete input converge to optimizers of the limiting functional which is defined on the underlying continuum as the number of points or graph vertices goes to infinity. The consistency of k-means is studied in [39]. More recently, [45, 2, 43] examined spectral clustering and graph Laplacians in terms of consistency by analyzing the convergence of the respective eigenvalues and eigenvectors. The works of [1, 34, 21] on the consistency of various graph-cut methods are more closely related to our analysis: they define a spatial scale parameter, which controls the connectivity of the graph through the edge weights and depends on the number of vertices, and derive certain conditions on this parameter in order for the respective graph-cut algorithm to be consistent. This analysis is performed in the setting of pointwise convergence in [1, 34], whereas [21] obtains results on Γ-convergence. In comparison, the analysis of consistency of our approximations also involves a spatial scale parameter, the radius of the graph, which is presented in Definition 2, with a function similar to that in the above works. An important distinction between these works and ours is that we prove consistency at the level of operators that are used in active contour models on graphs, not at the level of active contour algorithms themselves. Given that the aforementioned approaches do not cover the case of input which is presented in section 1 and includes an image function I defined on the vertices of the graph, we believe that studying the consistency of level set–based active contour algorithms on graphs constitutes an interesting topic for future research; our results are a promising first step in this direction.

Finally, our work bears some resemblance to unsupervised clustering of data that are represented as graphs embedded in Euclidean domains, where techniques based on nonnegative matrix factorization [30, 47, 46] have been applied successfully. Nonetheless, our method is not applicable to this setting, since the input is limited to the set of vertex locations $\mathcal{V}$ and does not include an image function I defined on the vertices (cf. section 1), which is essential for

formulating active contour models. In the same sense, our framework cannot be extended to stochastic block models [41] for graph segmentation, given that these models do not associate each vertex with a scalar value (which would model the image function) either.

3. Gradient approximation on graphs.

The first term of the active contour evolution model that needs to be approximated is the gradient of the bivariate embedding function. We thus develop a general method for calculating the gradient of a real-valued, bivariate function that is implicitly defined on a continuous domain, although its values are known only at a finite set of points, which constitute the vertices of the graph.

3.1. Main idea, notation, and definitions.

Compared to the proposals of Drakopoulos and Maragos [17] for gradient approximation, we attempt to incorporate our knowledge about the local spatial configuration of vertices in the approximation. More specifically, we introduce the concept of the *angle* around a vertex which is "occupied" by each of its neighbors and use this concept directly in our novel geometric gradient approximation. Our motivation for this approach comes from the following lemma in bivariate calculus.

Lemma 1. *The gradient of a differentiable function $u : \mathbb{R}^2 \to \mathbb{R}$ at point $\mathbf{x}$ is*

$$\nabla u(\mathbf{x}) = \frac{\displaystyle\int_0^{2\pi} D_\phi u(\mathbf{x})\, \mathbf{e}_\phi \, d\phi}{\pi}, \tag{1}$$

where $\mathbf{e}_\phi$ is the unit vector in direction ϕ and $D_\phi u(\mathbf{x})$ is the directional derivative of u at $\mathbf{x}$ in this direction, defined by

$$D_\phi u(\mathbf{x}) = \lim_{h \to 0} \frac{u(\mathbf{x} + h\mathbf{e}_\phi) - u(\mathbf{x})}{h}.$$

Based on Lemma 1, the goal of this section is to approximate the gradient at a vertex of the graph by substituting the integral

$$\mathcal{I} = \int_0^{2\pi} D_\phi u(\mathbf{x})\, \mathbf{e}_\phi \, d\phi \tag{2}$$

with a sum over all the neighbors of the vertex. To this end, we start by introducing several key concepts.

The Euclidean distance between vertices v and w of a graph $\mathcal{G}$ is denoted by $d(v, w)$, and the unit vector in the direction of the edge vw starting at v is denoted by $\mathbf{e}_{vw}$. We define $\phi(w) \in [0, 2\pi)$ as the angle between the vector $\mathbf{e}_{vw}$ and the horizontal axis, as in Figure 1. A vertex v will be alternatively denoted by $\mathbf{v}$ to declare its position vector. Moreover, we denote by $\mathcal{N}(v)$ the set of neighbors of v in $\mathcal{G}$, with cardinality $N(v)$. For the sake of brevity in notation, this cardinality will be written simply as N. We write $\mathcal{N}(v) = \{w_1, w_2, \ldots, w_N\}$ so that the angles $\phi(w_i)$ are in ascending order. Based on this ordering, we define the angle

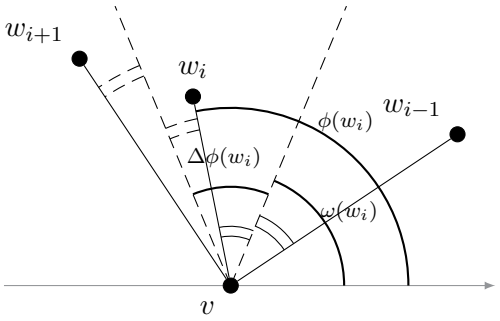


Figure 1. *Angles $\phi(w_i)$, $\Delta\phi(w_i)$, and $\omega(w_i)$.*

around v "occupied" by w_i, which we call *neighbor angle*, as

$$\Delta\phi(w_i) = \begin{cases} \dfrac{\phi(w_{i+1}) - (\phi(w_N) - 2\pi)}{2} & \text{if } i = 1, \\ \dfrac{\phi(w_1) + 2\pi - \phi(w_{i-1})}{2} & \text{if } i = N, \\ \dfrac{\phi(w_{i+1}) - \phi(w_{i-1})}{2} & \text{otherwise.} \end{cases} \tag{3}$$

In a similar fashion, we define the angle corresponding to the bisector between two consecutive neighbors as

$$\omega(w_i) = \begin{cases} \dfrac{\phi(w_i) + \phi(w_N) - 2\pi}{2} & \text{if } i = 1, \\ \dfrac{\phi(w_i) + \phi(w_{i-1})}{2} & \text{otherwise.} \end{cases} \tag{4}$$

A visual representation of the neighbor angle is provided in Figure 1. Using the above notation, we propose the following formula as the geometric gradient approximation at v:

$$\nabla u(v) \approx \frac{\displaystyle\sum_{i=1}^{N} \frac{u(w_i) - u(v)}{d(v, w_i)} \mathbf{e}_{vw_i} \Delta\phi(w_i)}{\pi}. \tag{5}$$

The directional derivative term in (1) is approximated by the difference quotient of the function along each edge. On the other hand, the angle differential is handled through the neighbor angles, which effectively constitute a Voronoi tessellation of the circle around v, created from its neighbors. The reasoning behind this approach is to use information about the change of u along each particular direction that comes from the neighbor which is *closest* to this direction.

If we chose not to take the neighbor angles into account, we would place equal importance on all the neighbors of the vertex, and we would return to an approximation similar to the

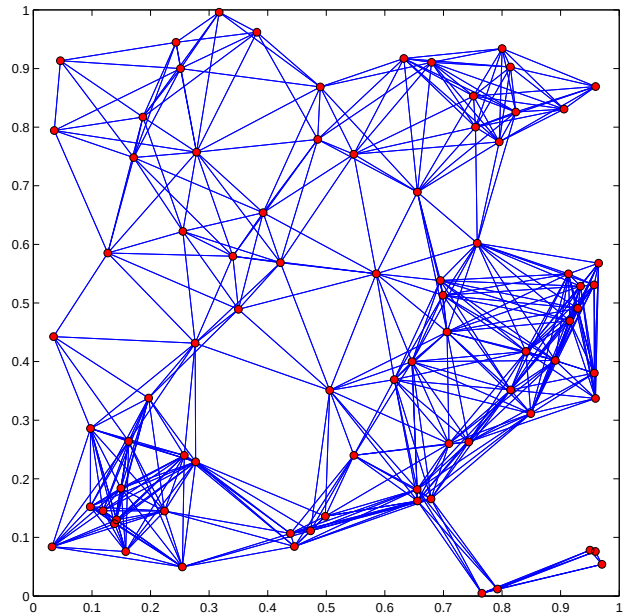


Figure 2. *An RGG embedded in* $D = [0,1]^2$*, with* $n = 80$ *vertices and radius* $\rho = 0.25$*.*

weighted sum that was introduced in [17]:

$$\nabla u(v) \approx \frac{\sum_{i=1}^{N} \frac{u(w_i) - u(v)}{d(v,w_i)} \mathbf{e}_{vw_i}}{N}. \tag{6}$$

3.2. Convergence for random geometric graphs. In the following, we will mainly focus on a certain type of graphs, namely *random geometric graphs* [38], to study the proposed gradient approximation theoretically.

Definition 2. *A random geometric graph (RGG)* $\mathcal{G}(n, \rho(n))$ *is comprised of a set* $\mathcal{V}$ *of vertices and a set* $\mathcal{E}$ *of edges. The set* $\mathcal{V}$ *consists of* n *points distributed uniformly at random and independently in a bounded region* $D \subset \mathbb{R}^2$*. The set* $\mathcal{E}$ *of edges is defined through the radius* $\rho(n)$ *of the graph: an edge connects two vertices* v *and* w *if and only if their distance is at most* $\rho(n)$*, i.e.,* $d(v,w) \leq \rho(n)$*.*

An instance of an RGG is given in Figure 2. We show that for this type of graphs, the approximation of (5) converges in probability to the true value of the gradient as the number of vertices increases, under some conditions on the radius, which constrain the density of the graph. Before stating the related theorem, we remind the reader of some definitions for the asymptotic notation which is used in the following analysis.

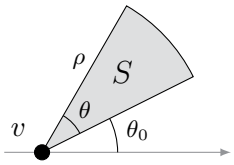


Figure 3. *Sector $S(v, \rho, \theta_0, \theta)$.*

Definition 3. *Let f and g be two nonnegative functions. Then*

$$\begin{aligned} f(n) \in O(g(n)) &\Leftrightarrow \exists k > 0\ \exists n_0\ \forall n \geq n_0 :\ f(n) \leq k g(n), \\ f(n) \in \Theta(g(n)) &\Leftrightarrow f(n) \in O(g(n)) \wedge g(n) \in O(f(n)), \\ f(n) \in o(g(n)) &\Leftrightarrow \forall k > 0\ \exists n_0\ \forall n \geq n_0 :\ f(n) < k g(n), \\ f(n) \in \omega(g(n)) &\Leftrightarrow \forall k > 0\ \exists n_0\ \forall n \geq n_0 :\ f(n) > k g(n). \end{aligned}$$

Theorem 4. *Let $u : \mathbb{R}^2 \to \mathbb{R}$ be a differentiable function, and let $\mathcal{G}(n, \rho(n))$ be an RGG embedded in $D = [0,1]^2$, with $\rho(n) \in \omega\left(n^{-1/2}\right) \cap o(1)$. For every vertex v of $\mathcal{G}$, the approximation of (5) converges in probability to $\nabla u(v)$.*

Proof. It suffices to prove that the sum

$$\mathcal{S} = \sum_{i=1}^{N} \frac{u(w_i) - u(v)}{d(v, w_i)}\, \mathbf{e}_{v w_i}\, \Delta\phi(w_i) \tag{7}$$

in (5) converges in probability to the integral (2). First, we show that the norm $\Lambda_n = \sup_{w \in \mathcal{N}(v)}\{\Delta\phi(w)\}$ of the partition of $[0, 2\pi]$ induced by the neighbor angles converges in probability to 0 in the limit of large graphs.

Let $S(v, \rho, \theta_0, \theta)$ be a circular sector centered at v, with radius ρ, occupying an angle $\theta > 0$ and whose rightmost radius is in the direction θ_0 (see Figure 3). For every vertex z_i, $i = 1, \ldots, n$, other than v, we can define a Bernoulli random variable indicating whether this vertex is inside S: $Z_i \sim \text{Bern}\left(\rho^2\, \theta/2\right)$. The sum of these variables follows a binomial distribution:

$$Z = \sum_{\substack{i=1 \\ z_i \neq v}}^{n} Z_i \sim \text{Bin}\left(n - 1, \frac{\rho^2(n)\, \theta}{2}\right).$$

Therefore, the probability that S is empty of vertices other than v is $\text{P}(Z = 0) = \left(1 - \rho^2(n)\, \theta/2\right)^{n-1}$, which converges to 0, because $\rho(n) \in \omega\left(n^{-1/2}\right)$. Let us consider a neighbor w of v and the event

$$A = \exists\, y, z \in \mathcal{V} \setminus \{v, w\} :\ y \neq z \wedge y \in S\left(v, \rho, \phi(w), \frac{\theta}{2}\right) \wedge z \in S\left(v, \rho, \phi(w) - \frac{\theta}{2}, \frac{\theta}{2}\right);$$

i.e., there is another neighbor of v "closer" than $\theta/2$ on each side of w. According to the above analysis, it is straightforward that $\lim_{n \to +\infty} \text{P}(A) = 1$. Moreover, A implies $\Delta\phi(w) \leq \theta$. Thus,

it holds that $\lim_{n\to+\infty} \mathrm{P}(\Delta\phi(w) \leq \theta) = 1$, and consequently

$$\lim_{n\to+\infty} \mathrm{P}(\Lambda_n \leq \theta) = 1 \; \forall \theta > 0, \tag{8}$$

which concludes the first part of the proof.

Second, we show that the sum

$$\mathcal{S}_1 = \sum_{i=1}^{N} D_{\phi(w_i)} u(v)\, \mathbf{e}_{vw_i}\, \Delta\phi(w_i)$$

converges in probability to $\mathcal{I}$. $\mathcal{S}_1$ constitutes a Riemann sum of $\mathbf{f}(\phi) = D_\phi u(v)\, \mathbf{e}_\phi$ over $[0, 2\pi]$ with respect to the partition induced by the neighbor angles. Therefore, (8) directly implies that $\mathcal{S}_1$ converges in probability to $\mathcal{I}$.

The third step of the proof is to show that the difference $\mathcal{S} - \mathcal{S}_1$ converges in probability to $\mathbf{0}$. Using the triangle inequality, we obtain

$$\|\mathcal{S} - \mathcal{S}_1\| \leq \sum_{i=1}^{N} \left| \frac{u(w_i) - u(v)}{d(v, w_i)} - D_{\phi(w_i)} u(v) \right| \Delta\phi(w_i).$$

For every $w \in \mathcal{N}(v)$, the first order Taylor approximation of u at v in the direction $\phi(w)$ yields

$$\left| u(w) - u(v) - d(v, w) D_{\phi(w)} u(v) \right| \in O\left(d^2(v, w)\right).$$

We divide both sides by $d(v, w)$ and use the fact that $0 \leq d(v, w) \leq \rho(n)$ to arrive at

$$\left| \frac{u(w) - u(v)}{d(v, w)} - D_{\phi(w)} u(v) \right| \in O(\rho(n)).$$

The last result holds for every neighbor of v, so we can substitute each term of the sum to get

$$\|\mathcal{S} - \mathcal{S}_1\| \in \sum_{i=1}^{N} O(\rho(n)) \Delta\phi(w_i) = O(\rho(n)) \sum_{i=1}^{N} \Delta\phi(w_i).$$

The sum of neighbor angles over all neighbors is constant and equals 2π, which in turn implies that

$$\|\mathcal{S} - \mathcal{S}_1\| \in 2\pi O(\rho(n)) = O(\rho(n)). \tag{9}$$

If we further make use of the fact that $\rho(n) \in o(1)$, we get that $\|\mathcal{S} - \mathcal{S}_1\| \in o(1)$. Consequently, $\mathcal{S} - \mathcal{S}_1$ converges to $\mathbf{0}$ almost surely and hence in probability as well.

Finally, we combine the results from the second and third steps to conclude the proof. For arbitrary $\epsilon > 0$, we use the triangle inequality to get

$$\mathrm{P}(\|\mathcal{S} - \mathcal{I}\| > \epsilon) \leq \mathrm{P}(\|\mathcal{S} - \mathcal{S}_1\| + \|\mathcal{S}_1 - \mathcal{I}\| > \epsilon) \leq \mathrm{P}\left(\|\mathcal{S} - \mathcal{S}_1\| > \epsilon\right) + \mathrm{P}\left(\|\mathcal{S}_1 - \mathcal{I}\| > \epsilon\right).$$

The above inequalities hold in the limit as well:

$$\lim_{n\to+\infty} \mathrm{P}(\|\mathcal{S} - \mathcal{I}\| > \epsilon) \leq \lim_{n\to+\infty} \mathrm{P}\left(\|\mathcal{S} - \mathcal{S}_1\| > \epsilon\right) + \mathrm{P}\left(\|\mathcal{S}_1 - \mathcal{I}\| > \epsilon\right) = 0,$$

where the last equality is due to convergence in probability of $\mathcal{S} - \mathcal{S}_1$ to $\mathbf{0}$ and of $\mathcal{S}_1$ to $\mathcal{I}$. ■

3.3. Asymptotic analysis of approximation error. Going one step further, we decompose the error introduced by the geometric gradient approximation in order to obtain a bound on the *rate* of convergence to the true gradient as the size of the RGG grows large. Since the framework in RGGs is stochastic, our results involve *expectations* for the various quantities.

Let us denote the error in approximating $\mathcal{I}$ with $\mathcal{S}$ by $\mathcal{E} = \mathcal{S} - \mathcal{I}$. Comparing the two expressions, we deduce that the approximation in $\mathcal{S}$ is threefold:

1. Directional derivatives along edges are approximated with difference quotients.
2. The approximate value for the directional derivative along each edge is used as a constant estimate for all the directions "falling into" the respective neighbor angle.
3. The unit vector in the direction of each edge is also used for all the directions corresponding to the respective neighbor angle.

To isolate the above sources of error, we construct intermediate expressions between $\mathcal{S}$ and $\mathcal{I}$ and bound the magnitude of the resulting differences.

Theorem 5. *Let $u : \mathbb{R}^2 \to \mathbb{R}$ be a differentiable function, and let $\mathcal{G}(n, \rho(n))$ be an RGG embedded in $D = [0,1]^2$, with $\rho(n) \in \omega\left(n^{-1/2}\right) \cap o(1)$. For every vertex v of $\mathcal{G}$, it holds that*

$$\mathrm{E}[\|\mathcal{E}\|] \in O\left(\rho(n) + \frac{1}{n\,\rho^2(n)}\right). \tag{10}$$

Proof. The first intermediate expression is $\mathcal{S}_1$, corresponding to the first part of the error, which can be expressed as

$$\mathcal{E}_1 = \mathcal{S} - \mathcal{S}_1 = \sum_{i=1}^{N} \left(\frac{u(w_i) - u(v)}{d(v, w_i)} - D_{\phi(w_i)} u(v) \right) \mathbf{e}_{vw_i}\, \Delta\phi(w_i). \tag{11}$$

We have shown in the proof of Theorem 4 that $\|\mathcal{E}_1\|$ is bounded asymptotically by the radius of the graph (cf. (9)), as the difference quotients which are used in $\mathcal{S}$ are first order approximations of the corresponding directional derivatives:

$$\|\mathcal{E}_1\| \in O(\rho(n)). \tag{12}$$

We define the second intermediate expression as

$$\mathcal{S}_2 = \sum_{i=1}^{N} \int_{\omega(w_i)}^{\omega(w_i)+\Delta\phi(w_i)} D_\phi u(v)\, \mathbf{e}_{vw_i}\, d\phi.$$

If we denote the direction of $\nabla u(v)$ by θ, we can write $D_\phi u(v) = \|\nabla u(v)\| \cos(\theta - \phi)$, and the second part of the error can be expressed as

$$\mathcal{E}_2 = \mathcal{S}_1 - \mathcal{S}_2 = \|\nabla u(v)\| \sum_{i=1}^{N} \mathbf{e}_{vw_i} \int_{\omega(w_i)}^{\omega(w_i)+\Delta\phi(w_i)} \left(\cos(\theta - \phi(w_i)) - \cos(\theta - \phi)\right) d\phi.$$

After performing some calculations, we obtain

$$\int_{\omega(w_i)}^{\omega(w_i)+\Delta\phi(w_i)} \cos(\theta - \phi)\, d\phi = 2\cos\left(\theta - \omega(w_i) - \frac{\Delta\phi(w_i)}{2}\right) \sin\left(\frac{\Delta\phi(w_i)}{2}\right).$$

The first factor on the right-hand side can be expanded as follows using the identity $\cos(a-b) = \cos(a)\cos(b) + \sin(a)\sin(b)$:

$$\cos\left(\theta - \omega(w_i) - \frac{\Delta\phi(w_i)}{2}\right) = \cos(\theta - \phi(w_i))\cos\left(\omega(w_i) + \frac{\Delta\phi(w_i)}{2} - \phi(w_i)\right) + \sin(\theta - \phi(w_i))\sin\left(\omega(w_i) + \frac{\Delta\phi(w_i)}{2} - \phi(w_i)\right).$$

Furthermore, if we apply the definitions of angles $\omega(w_i)$ and $\Delta\phi(w_i)$, we get the following bound:

$$\left|\omega(w_i) + \frac{\Delta\phi(w_i)}{2} - \phi(w_i)\right| \leq \frac{\Delta\phi(w_i)}{2} \Rightarrow \omega(w_i) + \frac{\Delta\phi(w_i)}{2} - \phi(w_i) \in O(\Delta\phi(w_i)).$$

We combine the Taylor expansions of the sine and cosine functions around 0,

$$\cos(a) = 1 + O\left(a^2\right) \text{ and } \sin(a) = a + O\left(a^3\right) = O(a),$$

with the above bound to obtain

$$\begin{aligned}
&2\cos\left(\theta - \omega(w_i) - \frac{\Delta\phi(w_i)}{2}\right)\sin\left(\frac{\Delta\phi(w_i)}{2}\right) \\
&= 2\left(\cos(\theta - \phi(w_i))\left(1 + O\left(\Delta\phi(w_i)^2\right)\right)\right. \\
&\quad \left. + \sin(\theta - \phi(w_i))O(\Delta\phi(w_i))\right)\left(\frac{\Delta\phi(w_i)}{2} + O\left(\Delta\phi(w_i)^3\right)\right) \\
&= \cos(\theta - \phi(w_i))\Delta\phi(w_i) + \cos(\theta - \phi(w_i))O\left(\Delta\phi(w_i)^3\right) + \sin(\theta - \phi(w_i))O\left(\Delta\phi(w_i)^2\right).
\end{aligned}$$

We substitute the above expression into $\mathcal{E}_2$ and use the triangle inequality and the fact that $\|\mathbf{e}_{vw_i}\| = 1\ \forall i \in \{1, \ldots, N\}$ to obtain

$$\|\mathcal{E}_2\| \in \|\nabla u(v)\| \sum_{i=1}^{N} \left|\cos(\theta - \phi(w_i))O\left(\Delta\phi(w_i)^3\right) + \sin(\theta - \phi(w_i))O\left(\Delta\phi(w_i)^2\right)\right|.$$

Additionally, the absolute values of the sine and the cosine in the last result are bounded from above by 1. Thus, the second part of the error is bounded by

$$\|\mathcal{E}_2\| \in \|\nabla u(v)\| \sum_{i=1}^{N} O\left(\Delta\phi(w_i)^2\right) = \|\nabla u(v)\| O\left(\sum_{i=1}^{N} \Delta\phi(w_i)^2\right). \tag{13}$$

The next step is to take the expectation for both sides of (13). Expectation preserves inequalities, and it is straightforward that $f \in O(g)$ implies $\mathrm{E}[f] \in O(\mathrm{E}[g])$ for two sequences of random variables f and g. As a result, it holds that

$$\mathrm{E}\left[\|\mathcal{E}_2\|\right] \in \|\nabla u(v)\| O\left(\mathrm{E}\left[\sum_{i=1}^{N} \Delta\phi(w_i)^2\right]\right).$$

Since N is itself a random variable, we employ the law of total expectation and take advantage of the fact that all $\Delta\phi(w_i)$ are identically distributed to write

$$\begin{aligned}\mathrm{E}\left[\sum_{i=1}^{N}\Delta\phi(w_i)^2\right] &= \mathrm{E}\left[\mathrm{E}\left[\sum_{i=1}^{N}\Delta\phi(w_i)^2\middle|N\right]\right] = \mathrm{E}\left[\sum_{i=1}^{N}\mathrm{E}\left[\Delta\phi(w_i)^2\middle|N\right]\right] \\ &= \mathrm{E}\left[N\,\mathrm{E}\left[\Delta\phi(w)^2\middle|N\right]\right]\end{aligned}$$

for some $w \in \mathcal{N}(v)$. Let us focus on the term $\mathrm{E}\left[\Delta\phi(w)^2\middle|N\right]$. In order to calculate this conditional expectation, we examine the distribution of the random variable $\Delta\phi(w)$. The probability that $\Delta\phi(w) \le x$ is equal to the probability that at least two neighbors of v other than w fall inside the $2x$ radial interval. Taking into account all possible combinations, it follows that for $N \ge 3$,

$$\mathrm{P}(\Delta\phi(w) \le \pi x) = \sum_{k=2}^{N-1}\binom{N-1}{k}x^k(1-x)^{N-1-k} = 1-(N-1)x(1-x)^{N-2}-(1-x)^{N-1}$$

for $x \in [0,\,1]$. The corresponding PDF of the random variable $\Delta\phi(w)/\pi$ is

$$(N-1)(N-2)x(1-x)^{N-3}, \quad x \in [0,\,1]\,,$$

and therefore $\Delta\phi(w)/\pi$ follows a Beta distribution with parameters $\alpha = 2$ and $\beta = N-2$. We use the formulas for the mean and variance of a Beta distribution with known parameters to write

$$\mathrm{E}\left[\left(\frac{\Delta\phi(w)}{\pi}\right)^2\middle|N\right] = \mathrm{Var}\left[\frac{\Delta\phi(w)}{\pi}\right] + \left(\mathrm{E}\left[\frac{\Delta\phi(w)}{\pi}\right]\right)^2 = \frac{2(N-2)}{N^2(N+1)} + \frac{4}{N^2} = \frac{6}{N(N+1)}.$$

Due to linearity of expectation, the conditional expectation we are after is

$$\mathrm{E}\left[\Delta\phi(w)^2\middle|N\right] = \frac{6\pi^2}{N(N+1)}. \tag{14}$$

Using (14), we obtain

$$\mathrm{E}\left[\|\mathcal{E}_2\|\right] \in \|\nabla u(v)\|\,O\left(\mathrm{E}\left[N\frac{6\pi^2}{N(N+1)}\right]\right) = \|\nabla u(v)\|\,O\left(\mathrm{E}\left[\frac{1}{N+1}\right]\right).$$

We compute the expectation $\mathrm{E}\left[1/(N+1)\right]$ using the binomial distribution of the number of

neighbors of v, $N \sim \text{Bin}\left(n-1, \pi\rho^2\right)$. The definition of this expectation is

$$\begin{aligned}
\mathrm{E}\left[\frac{1}{N+1}\right] &= \sum_{k=0}^{n-1} \frac{1}{k+1}\binom{n-1}{k}\left(\pi\rho^2\right)^k\left(1-\pi\rho^2\right)^{n-1-k} \\
&= \frac{1}{n\pi\rho^2}\sum_{k=0}^{n-1}\binom{n}{k+1}\left(\pi\rho^2\right)^{k+1}\left(1-\pi\rho^2\right)^{n-(k+1)} \\
&= \frac{1}{n\pi\rho^2}\sum_{k=1}^{n}\binom{n}{k}\left(\pi\rho^2\right)^{k}\left(1-\pi\rho^2\right)^{n-k} \\
&= \frac{1}{n\pi\rho^2}\left(1-\binom{n}{0}\left(\pi\rho^2\right)^0\left(1-\pi\rho^2\right)^n\right) = \frac{1}{n\pi\rho^2}\left(1-\left(1-\pi\rho^2\right)^n\right).
\end{aligned}$$

Since $\rho(n) \in \omega\left(n^{-1/2}\right) \cap o(1)$, it follows that

$$\left(1-\left(1-\pi\rho^2(n)\right)^n\right) \in \Theta(1).$$

Consequently, the expectation of the second part of the error is bounded through

$$\mathrm{E}[\|\mathcal{E}_2\|] \in \|\nabla u(v)\|\, O\left(\frac{1}{n\,\rho^2(n)}\right). \tag{15}$$

Finally, the third part of the approximation error is

$$\mathcal{E}_3 = \mathcal{S}_2 - \mathcal{I} = \|\nabla u(v)\| \sum_{i=1}^{N} \int_{\omega(w_i)}^{\omega(w_i)+\Delta\phi(w_i)} \cos(\theta-\phi)\left(\mathbf{e}_{vw_i} - \mathbf{e}_\phi\right) d\phi.$$

For the ith term of the above sum, it holds that $|\phi - \phi(w_i)| \le \Delta\phi(w_i)$, which further implies that

$$\|\mathbf{e}_{vw_i} - \mathbf{e}_\phi\| \le \left\|\mathbf{e}_{vw_i} - \mathbf{e}_{\phi(w_i)+\Delta\phi(w_i)}\right\| = 2\sin\left(\frac{\Delta\phi(w_i)}{2}\right).$$

Thus, the magnitude of $\mathcal{E}_3$ can be bounded using the triangle inequality as follows:

$$\begin{aligned}
\|\mathcal{E}_3\| &\le \|\nabla u(v)\| \sum_{i=1}^{N} \int_{\omega(w_i)}^{\omega(w_i)+\Delta\phi(w_i)} \|\mathbf{e}_{vw_i} - \mathbf{e}_\phi\|\, d\phi \\
&\le \|\nabla u(v)\| \sum_{i=1}^{N} \int_{\omega(w_i)}^{\omega(w_i)+\Delta\phi(w_i)} 2\sin\left(\frac{\Delta\phi(w_i)}{2}\right) d\phi \\
&= \|\nabla u(v)\| \sum_{i=1}^{N} 2\Delta\phi(w_i)\sin\left(\frac{\Delta\phi(w_i)}{2}\right) \in \|\nabla u(v)\| \sum_{i=1}^{N} O\left(\Delta\phi(w_i)^2\right).
\end{aligned}$$

The last bound is the same as that derived in (13) for $\|\mathcal{E}_2\|$, which yields

$$\mathrm{E}[\|\mathcal{E}_3\|] \in \|\nabla u(v)\|\, O\left(\frac{1}{n\,\rho^2(n)}\right). \tag{16}$$

The total error is $\mathcal{E} = \mathcal{E}_1 + \mathcal{E}_2 + \mathcal{E}_3$, and due to the triangle inequality and the fact that expectation preserves inequalities, it follows that

$$\mathrm{E}[\|\mathcal{E}\|] \leq \mathrm{E}[\|\mathcal{E}_1\|] + \mathrm{E}[\|\mathcal{E}_2\|] + \mathrm{E}[\|\mathcal{E}_3\|].$$

Based on the asymptotic bounds in (12), (15), and (16) and the sum property of the O symbol, we derive the bound of the total error

$$\mathrm{E}[\|\mathcal{E}\|] \in O\left(\rho(n) + \frac{1}{n\,\rho^2(n)}\right). \qquad \blacksquare$$

The radius of the graph is effectively the factor that determines the strictness of this bound. To provide better intuition, we study the case when $\rho(n) \in \Theta\left(n^{-a}\right)$, $a \in (0, 1/2)$. Substituting in (10), we obtain

$$\mathrm{E}[\|\mathcal{E}\|] \in O\left(n^b\right), \quad b = \begin{cases} -a & \text{if } a \in \left(0, \frac{1}{3}\right], \\ -1 + 2a & \text{if } a \in \left(\frac{1}{3}, \frac{1}{2}\right). \end{cases} \tag{17}$$

We visualize this expression for the error bound in Figure 4. The strictest upper bound is $O\left(n^{-1/3}\right)$, which is achieved for $a = 1/3$ and constitutes a trade-off between minimizing the first error term, which calls for small radii, and the other two terms, which require more neighbors and consequently larger radii. On the other hand, when $a \notin (0, 1/2)$, the conditions of Theorems 4 and 5 for $\rho(n)$ are not met, and convergence to the true value of the gradient is not guaranteed in general. For instance, for $a = 0$, we get $\rho(n) \in \Theta(1)$, which means that the radius does not approach 0 in the limit. In turn, this implies that in the first part of the error $\mathcal{E}_1$ in (11), the difference quotients are not guaranteed to converge to the respective directional derivatives, since the distance $d(v, w) \leq \rho(n)$ does not go to 0 in the limit. In addition, for $a = 1/2$, it holds that $\rho(n) \in \Theta(n^{-1/2})$ and hence $\rho^2(n) \in \Theta(1/n)$. Using the same notation as in the proof of Theorem 4, this implies that for every $\theta > 0$, the probability that sector $S(v, \rho(n), \theta_0, \theta)$ is empty of vertices other than v

$$\mathrm{P}(Z = 0) = \left(1 - \rho^2(n)\,\theta/2\right)^{n-1} \xrightarrow{n \to +\infty} p, \quad 0 < p < 1.$$

As a result, the norm Λ_n of the partition induced by the neighbor angles does not converge in probability to 0, and thus the sum $\mathcal{S}_1$ does not converge in probability to $\mathcal{I}$. A related observation which provides further intuition for the case where $a = 1/2$ is that the expected number of neighbors $\mathrm{E}[N] = (n-1)\pi\rho^2(n) \in \Theta(1)$. In other words, the expected number of terms of the Riemann sum $\mathcal{S}_1$ is finite in the limit, which forbids convergence of $\mathcal{S}_1$ to the respective integral $\mathcal{I}$.

From a practical point of view, (15) and (16) indicate that the error in the approximation of (5) increases as the magnitude of the true gradient grows large, i.e., when the function exhibits abrupt variations. This does not pose a problem for the calculation of gradient direction, since the latter does not depend on the range of the function's variation around the examined vertex. Utilizing all incident edges in the weighted sum of (5) ensures that all available information in the neighborhood of the vertex is used to estimate which direction

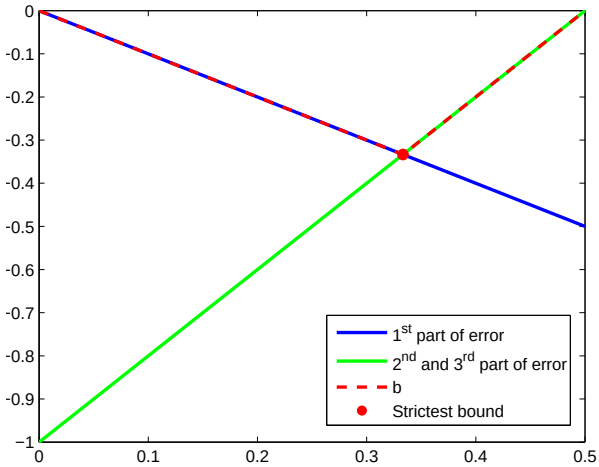


Figure 4. *Variation of exponent b of the asymptotic bound for gradient approximation error with respect to exponent a of the radius. Smaller values of b mean stricter error bounds. The exponents of the individual parts of the error are also presented.*

the gradient points to, as emphasized in [17]. However, the estimated gradient *magnitude* with our approximation is prone to greater error, as it depends on the range of the function's variation. The use of difference quotients in (5) accentuates this effect for dense graphs, where distances between neighboring vertices that appear in the denominator of the quotients approach zero. To circumvent this issue in practice, we adopt the approximation of [17] for gradient magnitude, namely the maximum absolute difference of values of the function along edges that are incident on v:

$$\|\nabla u(v)\| \approx \max_{w \in \mathcal{N}(v)} \{|u(w) - u(v)|\}. \tag{18}$$

4. Curvature approximation on graphs. After having devised an approximation scheme for the gradient of an embedding function, the next step is to build upon this scheme in order to estimate the curvature of the level sets of this function. This differs from the gradient case in that the input gradient values for curvature approximation are already approximate themselves, i.e., a *cascaded* approximation is attempted. Therefore, the error in curvature approximation on a graph is expected to accumulate compared to the gradient approximation error on the same graph, since the estimated curvature at a vertex inherits the error of the estimated gradients at its neighboring vertices.

4.1. Geometric approximation. In this type of curvature approximation, we follow an approach similar to [17]. More specifically, we exploit the expression of curvature as the divergence of the unit gradient field $\mathbf{F} = \nabla u / \|\nabla u\|$ of the embedding function u:

$$\kappa(v) = \operatorname{div} \mathbf{F}(v), \qquad \nabla u(v) \neq \mathbf{0}. \tag{19}$$

An integral definition of divergence as

$$\operatorname{div} \mathbf{F}(v) = \lim_{S \to \{\mathbf{v}\}} \frac{\oint_{\Gamma(S)} \mathbf{F} \cdot \mathbf{n} \, d\ell}{|S|} \tag{20}$$

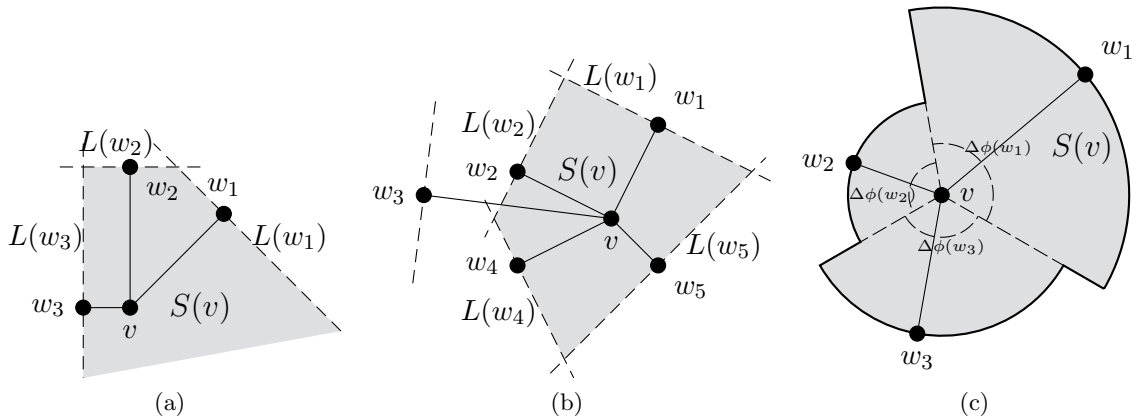


Figure 5. *Deficiencies of original curvature approximation of* [17] *are shown in* (a) *and* (b) *and solution through the new geometric approximation is shown in* (c). *In* (a) *the defined region has infinite area, while in* (b) *neighbor* w_3 *causes an unintuitive shape for* $S(v)$. *These ill cases are handled properly by defining region* $S(v)$ *through the neighbor angles, as done in* (c).

can then be used as a basis for geometric approximations of curvature, where S is a region with area $|S|$ and boundary $\Gamma(S)$ and $\mathbf{n}$ is the outward unit normal to this boundary. In [17], the integral in (20) is approximated using a polygonal region to form a finite sum over the neighbors of the vertex (as shown in [17, p. 8]). However, certain arrangements of the neighbors of the examined vertex, which are shown in Figure 5, can lead to regions with ill-defined area, boundary, and normals.

To tackle these problems, we employ again the *neighbor angles* that were introduced in section 3, in order to define the region S in (20) in a more compact and principled fashion. For vertex v, $S(v)$ is formed as a union of circular sectors, each corresponding to a neighbor of v, as we show in Figure 5(c). More formally, for each neighbor w of v, the respective circular sector is $S(v, d(v,w), \omega(w), \Delta\phi(w))$. The area of $S(v)$ can then be expressed as

$$|S(v)| = \sum_{i=1}^{N} \frac{\Delta\phi(w_i)}{2}\, d^2(v, w_i). \tag{21}$$

The challenge imposed by our construction of $S(v)$ is the choice of suitable values for $\mathbf{F}$ along the boundary of this region, given only its values at the locations of neighbors of v. The resulting boundary consists of arcs, each of which contains a neighbor of v, and line segments which connect these arcs. We fix the value of $\mathbf{F}$ along each arc at the geometric approximation computed for the corresponding neighbor w using (5), $\mathbf{F}_g(w)$. Moreover, for every line segment, we use the normalized mean of the approximate values of $\mathbf{F}$ along the two neighboring arcs. The idea is again to use information from the closest vertex, which should be more reliable. We visualize the described configuration in Figure 6.

Using the above approximations, we substitute the line integral in (20) with a sum of simple line integrals over single arcs and line segments, which have closed analytical forms. If we denote the integral over the arc $C_a(w_i)$ containing neighbor w_i by $I_a(w_i)$ and the integral

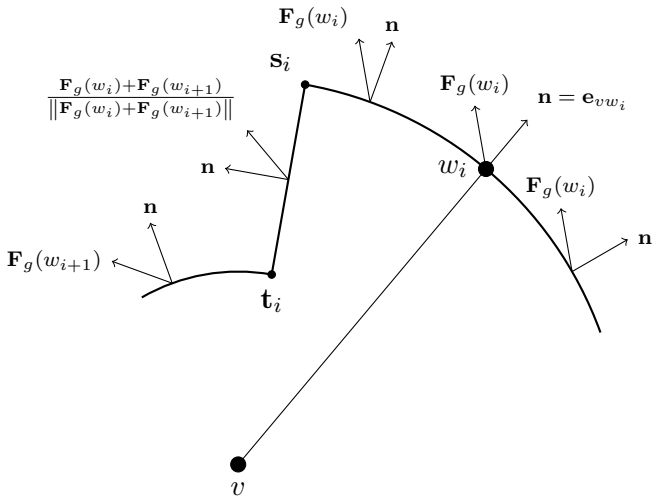


Figure 6. *The picture of values of* $\mathbf{F}_g$ *at a part of the boundary of* $S(v)$ *which corresponds to neighbor* w_i *of* v.

over the line segment $C_l(w_i)$ that connects the arcs $C_a(w_i)$ and $C_a(w_{i+1})$ by $I_l(w_i)$, we obtain

$$\begin{aligned} I_a(w_i) &= \int_{C_a(w_i)} \mathbf{F}_g(w_i) \cdot \mathbf{n}\, d\ell \\ &= d(v, w_i)\, \mathbf{F}_g(w_i) \cdot (\sin(\omega(w_{i+1})) - \sin(\omega(w_i)), \cos(\omega(w_i)) - \cos(\omega(w_{i+1}))) \end{aligned} \tag{22}$$

and

$$\begin{aligned} I_l(w_i) &= \int_{C_l(w_i)} \frac{\mathbf{F}_g(w_i) + \mathbf{F}_g(w_{i+1})}{\|\mathbf{F}_g(w_i) + \mathbf{F}_g(w_{i+1})\|} \cdot \mathbf{n}\, d\ell \\ &= (d(v, w_{i+1}) - d(v, w_i)) \frac{\mathbf{F}_g(w_i) + \mathbf{F}_g(w_{i+1})}{\|\mathbf{F}_g(w_i) + \mathbf{F}_g(w_{i+1})\|} \cdot (\sin(\omega(w_{i+1})), -\cos(\omega(w_{i+1}))). \end{aligned} \tag{23}$$

The proposed geometric approximation of curvature is given by

$$\kappa(v) \approx \frac{\sum_{i=1}^{N} I_a(w_i) + I_l(w_i)}{|S(v)|}. \tag{24}$$

For RGGs, this approximation is exact in the limit of large graphs like in the gradient approximation case, although the conditions are now stronger.

Theorem 6. *Let* $\mathcal{G}(n, \rho(n))$ *be an RGG embedded in* $D = [0,1]^2$, *with* $\rho(n) \in \omega\left(n^{-1/2}\right) \cap o(1)$ *and* v *a vertex of* $\mathcal{G}$. *If* $u : \mathbb{R}^2 \to \mathbb{R}$ *is continuously differentiable and* $\nabla u(v) \neq \mathbf{0}$, *then the approximation of* (24) *converges in probability to* $\kappa(v)$.

The full proof of Theorem 6 is given in M110010_01.pdf [local/web 254KB]. A brief outline

of the proof with its key ideas follows. The main task in the proof is to show that

$$\sum_{i=1}^{N} I_a(w_i) + I_l(w_i) - \oint_{\Gamma(S(v))} \mathbf{F} \cdot \mathbf{n}\, d\ell \xrightarrow{\mathrm{P}} 0. \tag{25}$$

We treat each component of $\Gamma(S(v))$, i.e., each arc and line segment, separately and prove the following convergence in probability results:

$$I_a(w_i) - \int_{C_a(w_i)} \mathbf{F} \cdot \mathbf{n}\, d\ell \xrightarrow{\mathrm{P}} 0 \quad \text{and} \quad I_l(w_i) - \int_{C_l(w_i)} \mathbf{F} \cdot \mathbf{n}\, d\ell \xrightarrow{\mathrm{P}} 0.$$

Afterward, these convergence results are combined through the sum property of convergence in probability to obtain (25). To prove the above results, we make use of the continuity of both $\mathbf{F}$ and ∇u at v, which is ensured by the conditions on u in Theorem 6. In addition, we use the law of total probability to prove that certain probabilities vanish in the limit, by expanding the examined probability with respect to the mutually exclusive events of $\mathbf{F}$ being continuous or discontinuous at a point (or on a curve), and exploiting the continuity of $\mathbf{F}$ at v to show that the probability of discontinuity vanishes. The last step relies on the fact that $\rho(n) \in o(1)$, which implies that the distance between the aforementioned point (or curve) and v converges in probability to 0.

4.2. Gradient-based approximation. We have seen that the divergence of a vector field can be used to compute the curvature. An alternative way to approximate this divergence is through its differential definition, which avoids handling the geometric quantities of subsection 4.1. More specifically, the unit gradient field can be expressed through its components as $\mathbf{F} = (F_1, F_2)$, so that its divergence is

$$\operatorname{div} \mathbf{F} = \frac{\partial F_1}{\partial x} + \frac{\partial F_2}{\partial y}. \tag{26}$$

As a result, a second application of the gradient approximation of (5), this time on the components of the approximate unit gradient field $\mathbf{F}_g = (F_{1,g}, F_{2,g})$, is adequate for calculating the curvature. The full expression for this approximation is

$$\kappa(v) \approx \frac{\displaystyle\sum_{i=1}^{N} \frac{F_{1,g}(w_i) - F_{1,g}(v)}{d(v,w_i)} \cos(\phi(w_i))\Delta\phi(w_i) + \sum_{i=1}^{N} \frac{F_{2,g}(w_i) - F_{2,g}(v)}{d(v,w_i)} \sin(\phi(w_i))\Delta\phi(w_i)}{\pi}. \tag{27}$$

This gradient-based curvature approximation also converges in probability for RGGs, under slightly stricter conditions than the geometric curvature approximation.

Theorem 7. *Let $\mathcal{G}(n, \rho(n))$ be an RGG embedded in $D = [0,1]^2$, with $\rho(n) \in \omega\left(n^{-1/2}\right) \cap o(1)$ and v a vertex of $\mathcal{G}$. Let $u : \mathbb{R}^2 \to \mathbb{R}$ be twice differentiable and $\nabla u(v) \neq \mathbf{0}$. Then, the approximation of (27) converges in probability to $\kappa(v)$.*

Proof. With the same argument as in the proof of Theorem 6, it can be shown that

$$\mathbf{F}_g(v) - \mathbf{F}(v) \xrightarrow{\mathrm{P}} \mathbf{0} \quad \text{and} \quad \mathbf{F}_g(w_i) - \mathbf{F}(w_i) \xrightarrow{\mathrm{P}} \mathbf{0} \quad \forall i \in \{1, \ldots, N\}.$$

These results can be combined into

$$\mathbf{F}_g(w_i) - \mathbf{F}_g(v) - (\mathbf{F}(w_i) - \mathbf{F}(v)) \xrightarrow{\mathrm{P}} \mathbf{0} \quad \forall i \in \{1, \ldots, N\}.$$

Using the above convergence and the product and sum properties of convergence in probability, it follows that

$$\sum_{i=1}^{N} \frac{F_{1,g}(w_i) - F_{1,g}(v)}{d(v, w_i)} \mathbf{e}_{vw_i} \Delta\phi(w_i) - \sum_{i=1}^{N} \frac{F_1(w_i) - F_1(v)}{d(v, w_i)} \mathbf{e}_{vw_i} \Delta\phi(w_i) \xrightarrow{\mathrm{P}} \mathbf{0}.$$

Furthermore, since u is twice differentiable and $\nabla u(v) \neq \mathbf{0}$, $\mathbf{F}$ is differentiable at v, as the quotient of differentiable functions with nonzero denominator. As a result, F_1 is also differentiable at v, and Theorem 4 applies:

$$\frac{\displaystyle\sum_{i=1}^{N} \frac{F_1(w_i) - F_1(v)}{d(v, w_i)} \mathbf{e}_{vw_i} \Delta\phi(w_i)}{\pi} \xrightarrow{\mathrm{P}} \nabla F_1(v).$$

This result can be combined with the previous one through the sum property of convergence in probability to obtain

$$\frac{\displaystyle\sum_{i=1}^{N} \frac{F_{1,g}(w_i) - F_{1,g}(v)}{d(v, w_i)} \mathbf{e}_{vw_i} \Delta\phi(w_i)}{\pi} \xrightarrow{\mathrm{P}} \nabla F_1(v).$$

Analysis identical to that above leads to the same result for F_2.

The last step of the proof is to isolate from the last convergence results the x-component of $\nabla F_1(v)$ and the y-component of $\nabla F_2(v)$, which appear in (26), and use the sum property of convergence in probability to show that the expression on the right-hand side of (27) converges in probability to $\partial F_1(v)/\partial x + \partial F_2(v)/\partial y$. ∎

5. Smoothing filtering on graphs. In the previous two sections, we developed certain approximations on arbitrary graphs to compute quantities that are essential for active contour models. Despite convergence of these approximations to the true values of the respective quantities in the case of RGGs, in practice there is a nonnegligible error for graphs with finite number of vertices. This error is propagated to the embedding function of the active contour after each update, and therefore it may be accumulated after several iterations. Consequently, it is intuitive to apply smoothing filtering on the approximate values at a local, neighborhood level as an empirical means to eliminate potential outliers by taking into account the values at neighboring vertices. Of course, this type of filtering induces an additional computational burden, especially when performed at every iteration, but we are willing to trade some speed for accuracy. We analyze this neighborhood-based filtering in subsection 5.1.

Moreover, a different type of smoothing is involved in the GAC model [7] as a form of regularization of the image to distinguish predominant "edges" from small-scale variations. This regularization typically uses an isotropic Gaussian filter, for which we propose two modified versions that are tailored for the arbitrary graph setting in subsection 5.2.

5.1. Neighborhood-based average/median filtering. To compute a smoothed version of a function defined on a graph, one option is to operate in the same neighborhood-based framework that we presented in the previous sections and apply a simple filter on the original version of the function. This filter can be either an average or a median filter, receiving as input the set of function values at the vertex itself and all its neighbors. In the case of curvature this is straightforward, while for gradient, we filter each of the two vector components separately.

In the following analysis, we provide theoretical intuition on the benefit of smoothing at a neighborhood level by considering the case of smoothing the gradient approximation with an average filter on an RGG. Our analysis is focused on the x-component of gradient for the sake of simplicity, but the conclusions hold for the complete gradient vector as well. From a statistical point of view, we show that using the neighbors of a vertex to form an ensemble of estimators of the gradient at that vertex reduces the variance of the estimation compared to the basic geometric approximation, while not changing the bias.

More formally, consider the sum $\mathcal{S}$ in (7) which is involved in the geometric gradient approximation at vertex v. For the purposes of this analysis, we denote this sum by $\mathcal{S}(v) = (\mathcal{S}_x(v), \mathcal{S}_y(v))$ and the integral $\mathcal{I}$ in (2) (which is approximated by $\mathcal{S}(v)$) by $\mathcal{I}(v) = (\mathcal{I}_x(v), \mathcal{I}_y(v))$. The term $\mathcal{S}_x(v)$ can be interpreted statistically as an estimator of $\mathcal{I}_x(v)$, with mean squared error

$$\text{MSE}\left(\mathcal{S}_x(v)\right) = \mathrm{E}\left[\left(\mathcal{S}_x(v) - \mathcal{I}_x(v)\right)^2\right] = \mathrm{E}\left[\left(\mathcal{S}_x(v) - \mathrm{E}\left[\mathcal{S}_x(v)\right]\right)^2\right] + \left(\mathrm{E}\left[\mathcal{S}_x(v)\right] - \mathcal{I}_x(v)\right)^2, \tag{28}$$

where $k = \mathrm{E}\left[\left(\mathcal{S}_x(v) - \mathrm{E}\left[\mathcal{S}_x(v)\right]\right)^2\right]$ is the variance of $\mathcal{S}_x(v)$ and $l = \mathrm{E}\left[\mathcal{S}_x(v)\right] - \mathcal{I}_x(v)$ is its bias.

For a neighbor w of v, the same notation as above can be used to denote the respective sum $\mathcal{S}(w) = (\mathcal{S}_x(w), \mathcal{S}_y(w))$ and integral $\mathcal{I}(w) = (\mathcal{I}_x(w), \mathcal{I}_y(w))$. Let us also denote $\overline{\mathcal{N}}(v) = \mathcal{N}(v) \cup \{v\}$, which is the set of neighbors of vertex v, augmented with v. Then, our proposed neighborhood-based average filtering replaces $\mathcal{S}_x(v)$ with

$$\mathcal{S}_x^a(v) = \frac{1}{N+1} \sum_{w \in \overline{\mathcal{N}}(v)} \mathcal{S}_x(w). \tag{29}$$

At this point, we assume that function u, whose gradient is approximated, is differentiable, as required in Theorem 4. This assumption implies that u is locally linear at $\mathbf{v}$, i.e., its gradient is approximately constant in the "continuous," $\mathbb{R}^2$-neighborhood of $\mathbf{v}$. Since this analysis concerns RGGs, every $w \in \mathcal{N}(v)$ belongs to the aforementioned $\mathbb{R}^2$-neighborhood of $\mathbf{v}$, as the distance $d(v, w)$ is bounded by the radius ρ. We can therefore write $\mathcal{I}_x(w) \approx \mathcal{I}_x(v)$. In addition, the isotropy of the generating point process of the vertices of an RGG (cf. Definition 2) and the locally linear profile of u at $\mathbf{v}$ imply that the statistical quantities of geometric gradient approximation at w are approximately equal to those at v:

$$l \approx \mathrm{E}\left[\mathcal{S}_x(w)\right] - \mathcal{I}_x(w) \approx \mathrm{E}\left[\mathcal{S}_x(w)\right] - \mathcal{I}_x(v) \quad \text{and} \quad k \approx \mathrm{E}\left[\left(\mathcal{S}_x(w) - \mathrm{E}\left[\mathcal{S}_x(w)\right]\right)^2\right]. \tag{30}$$

Based on (30), we compute the bias and variance of the average filter estimator in (29). The

bias is

$$\mathrm{E}\left[\mathcal{S}_x^a(v)\right] - \mathcal{I}_x(v) = \frac{1}{N+1} \sum_{w \in \overline{\mathcal{N}}(v)} \left(\mathrm{E}\left[\mathcal{S}_x(w)\right] - \mathcal{I}_x(v)\right) \approx \frac{(N+1)l}{N+1} = l, \tag{31}$$

where the first equality follows from linearity of expectation. Before computing the variance, we elaborate on the covariance $\mathrm{Cov}\left(\mathcal{S}_x(w), \mathcal{S}_x(z)\right)$ for w, $z \in \overline{\mathcal{N}}(v)$ with $w \neq z$. Even though w and z may share some neighbors, implying that the values of u that are input to the respective terms of (7) are identical for the two vertices, the rest of the factors in these terms, in particular the neighbor angles, still differ significantly as they depend on the spatial configuration of the *complete* set of neighbors of w and z. Consequently, $\mathrm{Cov}\left(\mathcal{S}_x(w), \mathcal{S}_x(z)\right) \approx 0$, and we obtain

$$\begin{aligned} \mathrm{E}\left[\left(\mathcal{S}_x^a(v) - \mathrm{E}\left[\mathcal{S}_x^a(v)\right]\right)^2\right] &= \frac{1}{(N+1)^2} \sum_{w \in \overline{\mathcal{N}}(v)} \mathrm{E}\left[\left(\mathcal{S}_x(w) - \mathrm{E}\left[\mathcal{S}_x(w)\right]\right)^2\right] \\ &\quad + \frac{1}{(N+1)^2} \sum_{w \in \overline{\mathcal{N}}(v)} \sum_{\substack{z \in \overline{\mathcal{N}}(v) \\ z \neq w}} \mathrm{Cov}\left(\mathcal{S}_x(w), \mathcal{S}_x(z)\right) \\ &\approx \frac{k}{N+1}. \end{aligned} \tag{32}$$

The mean squared error of the average filter estimator is thus $\mathrm{MSE}\left(\mathcal{S}_x^a(v)\right) \approx k/(N+1) + l^2$, indicating that the proposed average filtering reduces the variance by a factor of $N+1$ compared to the original geometric gradient approximation and leaves the bias unchanged, i.e., the overall mean squared error of the estimation is reduced.

To validate the benign effects of smoothing filtering on both gradient and curvature, we experiment with certain closed-form functions defined on RGGs. For each graph, we compute the function's gradient and the curvature of its level sets at each vertex using the proposed approximations, and afterward we filter the results with an average or median filter. Deriving the analytical expressions of the function's gradient and curvature, we are able to compare them with our estimates. To enable a quantitative assessment of our approximations and smoothing filters, we define a suitable error metric which we call *relative error* and denote by e_r. If the error in approximating, e.g., the gradient at each vertex is defined as the difference between the computed approximate value and the true value, then the relative error is simply the ratio of the energy of the error signal on the entire graph to the energy of the true gradient signal on the graph:

$$e_r = \frac{E_{\text{error}}}{E_{\text{analytical}}}. \tag{33}$$

In the experiments that follow in the rest of the paper and in M110010_01.pdf [local/web 254KB], the radius of an RGG is chosen as $\rho(n) = 0.6n^{-1/3}$ unless otherwise mentioned, so as to achieve the strictest asymptotic bound for gradient approximation error according to the results of section 3. In the rest of this section, all RGGs are embedded in $[0,1]^2$. To verify the enhancement of gradient approximation with smoothing filtering quantitatively, we evaluate

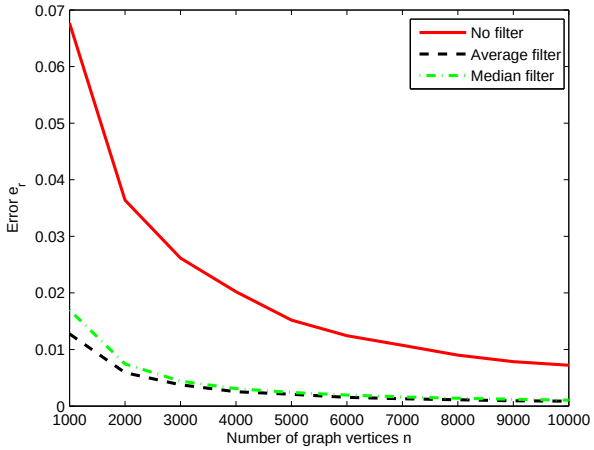


Figure 7. *Relative error of gradient approximations for a Gaussian function defined on RGGs of increasing size. The geometric approximation with no filtering is compared to its filtered versions with an average or median filter.*

the relative error for an isotropic Gaussian on RGGs whose size ranges from 1000 to 10,000 vertices. The analytical form of the Gaussian is

$$\exp\left\{-\left[(x-x_0)^2+(y-y_0)^2\right]\ /\ 2\sigma^2\right\}, \tag{34}$$

with $\sigma = 0.25$ and $x_0 = y_0 = 0.5$. Figure 7 shows average values of e_r over 10 different graphs for each size, which leads to a reduced variance in the estimation. Using either an average or a median filter substantially reduces the relative error irrespective of size. This leads us to apply smoothing on the gradient and feed the smoothed version to curvature computation. Detailed illustrations of approximate gradient vector fields which are computed in this experiment are given in M110010_01.pdf [local/web 254KB].

In Figure 8(a), we present the results of an experiment similar to that in Figure 7, this time focusing on the curvature of a function that corresponds to an elliptical cone, which we will call conic function for short. The form of this conic function is

$$\sqrt{\frac{(x-x_0)^2}{\alpha^2}+\frac{(y-y_0)^2}{\beta^2}}, \tag{35}$$

where $\alpha = 0.4$, $\beta = 0.3$, $x_0 = -0.25$, and $y_0 = 0.5$. Median filtering appears superior: the median-filtered approximations exhibit lower error than the corresponding average-filtered ones over almost the entire range of graph sizes (except for the smallest sizes). Even more important, the relative error of median-filtered curvature is steadily decreasing for increasing graph size with both the geometric and the gradient-based approximation, in contrast to the average-filtered cases, where the error stops decreasing around 4000 vertices. Due to these facts, we use median filtering for smoothing gradient and curvature for the GAC model in the rest of the paper. We also observe that the geometric approximation induces a relatively smaller error than the gradient-based approximation for both types of filters.

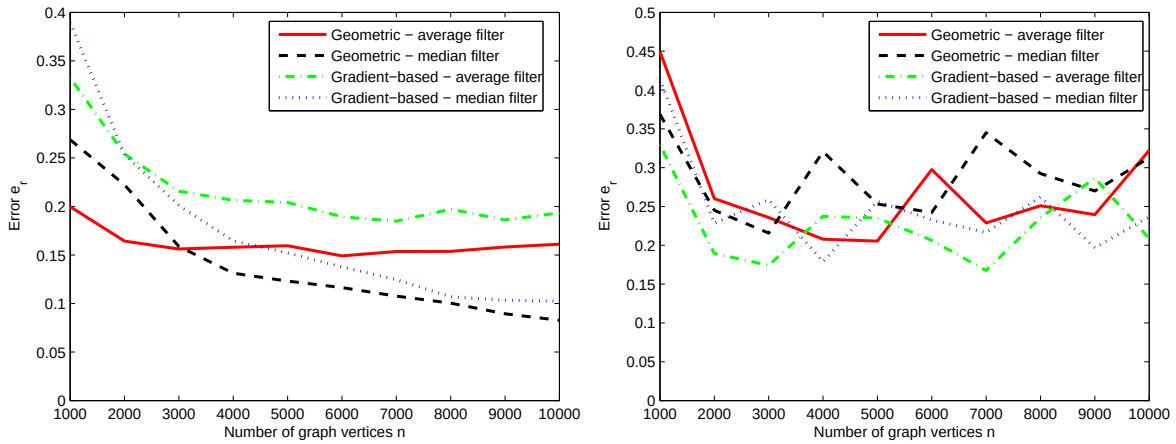


Figure 8. *Relative error of curvature approximations for a conic function defined on RGGs of increasing size. In (a), the conic function does not assume a local extremum in the interior of the graph's region, whereas in (b) it does. All four combinations of approximation type (geometric or gradient-based) and smoothing filter (average or median) are compared in both cases.*

Illustrations of approximate curvature profiles from this experiment are provided in M110010_01.pdf [local/web 254KB].

To emphasize the importance of the assumptions made for convergence of the curvature approximations in section 4, we repeat the last experiment, setting $x_0 = 0.5$. This way, the conic function is not differentiable at $(0.5, 0.5)$, which lies in the interior of the graph's region, and therefore the assumptions of Theorems 6 and 7 do not hold necessarily for every vertex of the graph. In fact, near this point, the true curvature of the level sets approaches infinity. Indeed, the evolution of relative error depicted in Figure 8(b) confirms that all approximations are less accurate, and they do not converge in this case.

5.2. Gaussian smoothing. At the initialization stage of the GAC algorithm, one of the tasks is to process the original image function I so as to obtain a smoother version of it. In this way, the stopping function g which drives the active contour to object boundaries is aided to capture only salient edges in the image and ignore small local variations.

Following [17], we employ Gaussian smoothing defined on graphs for this task. The filter is an isotropic 2D Gaussian with standard deviation σ:

$$G_\sigma(\mathbf{x}) = \frac{1}{2\pi\sigma^2} \exp\left(-\frac{\|\mathbf{x}\|^2}{2\sigma^2}\right). \tag{36}$$

We will denote by I_σ the smoothed image function which is obtained by using such a filter. The authors of [17] use a simple graph-based convolution of (36) with the image function to perform smoothing:

$$I_\sigma(\mathbf{v}) = \sum_{\mathbf{w}\in\mathcal{V}} I(\mathbf{w}) G_\sigma(\mathbf{v} - \mathbf{w}). \tag{37}$$

A simple formula is then used to calculate the stopping function g:

$$g(\|\nabla I_\sigma\|) = \frac{1}{1 + \frac{\|\nabla I_\sigma\|^2}{\lambda^2}}. \tag{38}$$

However, the arbitrary graph setting introduces nonuniformities: in some parts of the graph, the vertices might be distributed more densely than in other parts. This implies that (37) will operate counter-intuitively, introducing variations to the smoothed image in regions of the graph where the original image function is constant. To demonstrate this behavior, we use a simple binary image of a disk, shown in Figure 9(a). The result of applying (37) on this image is shown in Figure 9(b). Not only has the range of image values changed, but the interior of the original disk also exhibits significant variations in image values. This shortcoming is propagated to $\|\nabla I_\sigma\|$ and g, as we present in Figure 9(e) and Figure 9(h), respectively. There is a deviation of g from the ideal value of 1 inside the area corresponding to the disk, and a variation in its values as well, which means that the gradient of the stopping function is not $\mathbf{0}$ as it should be.

We tackle this issue by adding a normalization term to (37) to address nonuniformities:

$$I_\sigma(\mathbf{v}) = \frac{\sum_{\mathbf{w}\in\mathcal{V}} I(\mathbf{w}) G_\sigma(\mathbf{v}-\mathbf{w})}{\sum_{\mathbf{w}\in\mathcal{V}} G_\sigma(\mathbf{v}-\mathbf{w})}. \tag{39}$$

We term this method *normalized Gaussian filtering* and show its result for the examined disk image in Figure 9(c). The smoothed image is now very similar to the output of simple Gaussian filtering in the usual image processing setting with regularly spaced pixels. As a result, the corresponding magnitude of the gradient of I_σ and g functions (shown in Figure 9(f) and Figure 9(i), respectively) matches our expectations.

An important observation at this point is that in the stopping function computation pipeline, we are interested in the smoothed image's gradient rather than the smoothed image itself. Since the derivatives of the Gaussian filter have closed analytical forms, it is appealing to exchange the convolution with the gradient operator and convolve the image directly with Gaussian derivatives in order to obtain the gradient of I_σ. In this case, normalization is not straightforward as in normalized Gaussian filtering: Gaussian derivatives assume both positive and negative values. We circumvent this difficulty by splitting the vertices into two sets, according to the sign of the Gaussian derivative with respect to the processed vertex, and perform separate normalization for each of these sets. This separation can be easily expressed in terms of the vertices' coordinates. If we denote $\mathbf{v} = (v_1, v_2)$, then Gaussian derivative

filtering with separate normalization is defined as

$$\nabla I_\sigma(\mathbf{v}) = \left(\frac{\displaystyle\sum_{\substack{\mathbf{w}\in\mathcal{V}:\\ w_1\geq v_1}} I(\mathbf{w})\frac{\partial G_\sigma(\mathbf{v}-\mathbf{w})}{\partial x}}{\displaystyle\sum_{\substack{\mathbf{w}\in\mathcal{V}:\\ w_1\geq v_1}} \frac{\partial G_\sigma(\mathbf{v}-\mathbf{w})}{\partial x}} + \frac{\displaystyle\sum_{\substack{\mathbf{w}\in\mathcal{V}:\\ w_1< v_1}} I(\mathbf{w})\frac{\partial G_\sigma(\mathbf{v}-\mathbf{w})}{\partial x}}{-\displaystyle\sum_{\substack{\mathbf{w}\in\mathcal{V}:\\ w_1< v_1}} \frac{\partial G_\sigma(\mathbf{v}-\mathbf{w})}{\partial x}}, \right. \tag{40}$$

$$\left. \frac{\displaystyle\sum_{\substack{\mathbf{w}\in\mathcal{V}:\\ w_2\geq v_2}} I(\mathbf{w})\frac{\partial G_\sigma(\mathbf{v}-\mathbf{w})}{\partial y}}{\displaystyle\sum_{\substack{\mathbf{w}\in\mathcal{V}:\\ w_2\geq v_2}} \frac{\partial G_\sigma(\mathbf{v}-\mathbf{w})}{\partial y}} + \frac{\displaystyle\sum_{\substack{\mathbf{w}\in\mathcal{V}:\\ w_2< v_2}} I(\mathbf{w})\frac{\partial G_\sigma(\mathbf{v}-\mathbf{w})}{\partial y}}{-\displaystyle\sum_{\substack{\mathbf{w}\in\mathcal{V}:\\ w_2< v_2}} \frac{\partial G_\sigma(\mathbf{v}-\mathbf{w})}{\partial y}} \right).$$

The application of Gaussian derivative filtering with separate normalization on the examined image produces the results shown in Figure 9(d) for $\|\nabla I_\sigma\|$ and Figure 9(g) for g. The quality of the stopping function is at least as satisfactory as in the normalized Gaussian filtering case of Figure 9(i). Consequently, both our novel methods for Gaussian smoothing on graphs outperform the simple Gaussian filtering approach of [17] and can be readily used in the GAC framework.

6. Results. Having approximated the main terms of active contour models with level sets in the arbitrary graph setting, we are ready to apply iterative algorithms for segmentation on graphs that stem from the PDEs which are used in these models. The input comprises a graph $\mathcal{G} = (\mathcal{V}, \mathcal{E})$ and an image function I, as specified in section 1. In the following experiments, we select two representative active contour models, the GAC model [7] and the ACWE model [9], to demonstrate the effectiveness of our approximations.

An important practical aspect of applying active contour models on graphs is the method used to create the graph. The original input often consists only of $\mathcal{V}$ and I, without any information about the edges of the graph. This setting leaves us free to choose the underlying model for the edge structure of the graph. In our experiments, we used random geometric graphs and Delaunay triangulations (DTs). In other cases, one may have at her disposal a regular image defined on a grid; however, the graph framework is still relevant. In particular, subsampling the original image by placing a number of vertices which is much smaller than the total number of pixels uniformly at random in the original image domain brings us to the previous setting and at the same time reduces the size of the input compared to the standard grid-based active contour framework. An attractive alternative to random sampling is to extract vertex locations via watershed transformation. More specifically, we apply watershed transformation directly to the gradient of the image and place the vertices at the centroids (ultimate erosions) of the resulting superpixels. We show that this approach leads to far better graph segmentation results than does randomly sampling the image, as it captures image particularities into the spatial structure of the graph and "compresses" the image function into its part that is crucial for segmentation.

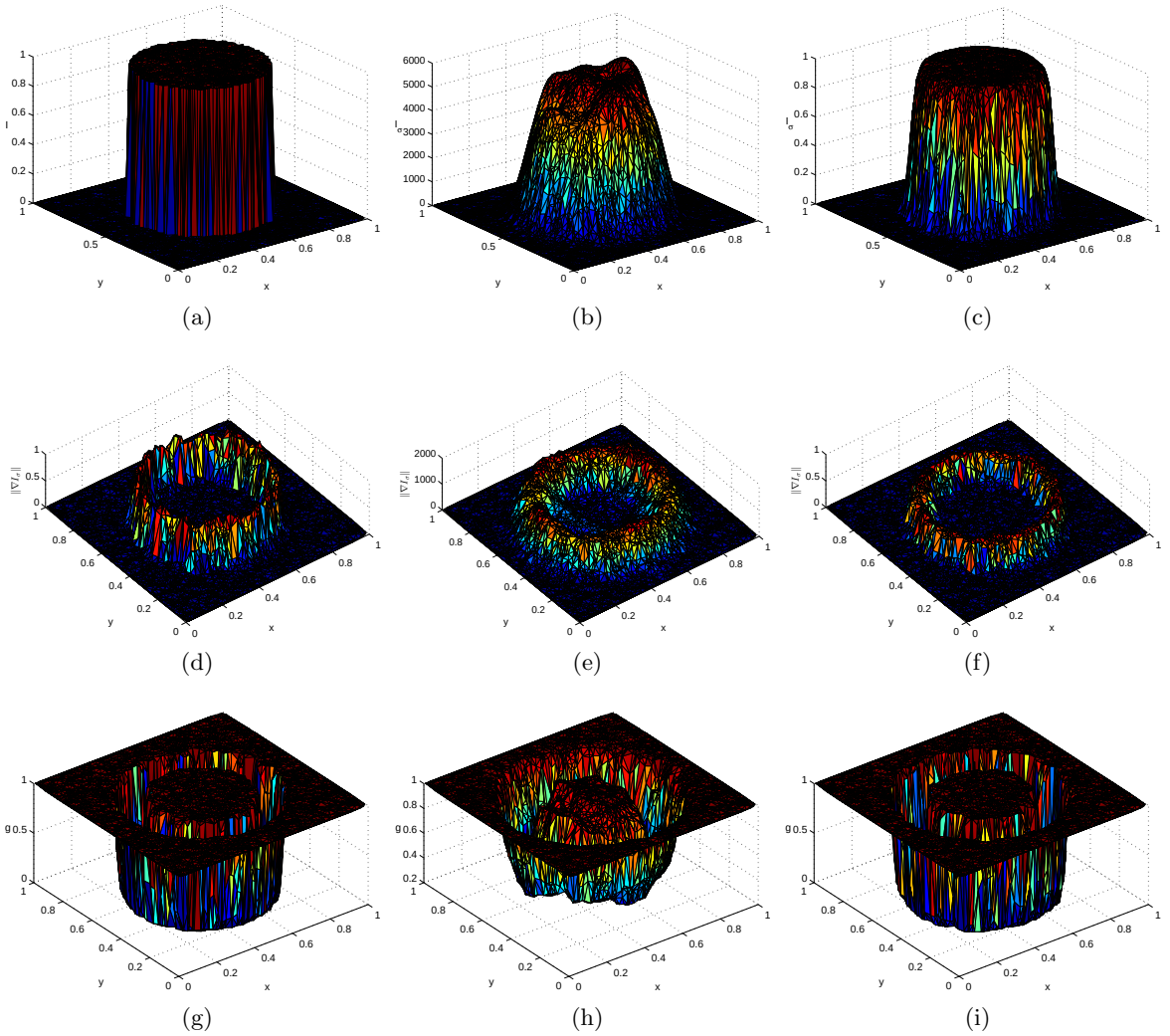


Figure 9. *Comparison of methods for Gaussian smoothing and computation of stopping function. The original image on the graph (a disk) is shown in* (a)*. The rest of the figure is organized as follows: the two rightmost plots of the top row contain smoothed versions I_σ of the original image, the middle row contains gradient magnitudes of I_σ, and the bottom row contains g values based on these gradient magnitudes. The results in the left column pertain to our Gaussian derivative filtering with separate normalization using $\sigma = 0.02$ and $\lambda = 0.05$, those in the middle column pertain to the simple Gaussian filtering of* [17] *using $\sigma = 0.05$ and $\lambda = 1000$, and those in the right column correspond to our normalized Gaussian filtering with $\sigma = 0.02$ and $\lambda = 0.05$. We use the approximation of* (18) *to compute the gradient magnitude in* (e) *and* (f)*. For all three methods, we filter $\|\nabla I_\sigma\|$ with a median filter before feeding it to* (38) *for computing g.*

6.1. Geodesic active contours. The algorithm for GACs on graphs with our approximations includes the following steps:

1. Compute $g(\|\nabla I_\sigma\|)$, using either (39) and (18) or (40) to compute $\|\nabla I_\sigma\|$. In both cases, median filtering is applied to $\|\nabla I_\sigma\|$ before plugging it into the formula for g. Then, compute the magnitude of g's gradient using (18) and its direction using (5).

2. Choose a subset X of $\mathcal{V}$ which *contains* the objects to be detected and initialize the embedding function with the signed distance function from the boundary of X, denoted by u_0. By convention, u_0 is positive inside X.
3. Iterate for $r \in \mathbb{N}$,

$$u_r = u_{r-1} + \Delta t((\kappa - c)\,\|\nabla u_{r-1}\|\,g + \nabla g \cdot \nabla u_{r-1}), \tag{41}$$

until convergence. In the difference equation (41), Δt and c are positive constants and κ is the curvature of the level sets of u_{r-1}.

In practice, after each iteration of step 3 of the algorithm, we smooth u_r with a median filter before proceeding to the next iteration. The parameters involved in the algorithm are the time step Δt of the difference equation, the balloon force constant c, the scale σ of the Gaussian smoothing filter, and parameter λ in g's formula. Tuning their values depending on the particular input is pivotal in obtaining satisfactory segmentation results. In the following experiments of this subsection, unless otherwise specified, we set $\Delta t = 0.005$ and $c = 2$.

A brief analysis of the computational complexity of each iteration (41) of our algorithm follows. We examine each term on the right-hand side of (41) and aggregate the individual contributions of all terms to obtain the total complexity. More precisely, g and ∇g are constant across iterations, so they are computed only once at initialization. The same applies to the geometric quantities, such as angles or distances, which are involved in our approximations of gradient and curvature. Therefore, computing the gradient ∇u_{r-1} at a single vertex v with (5) has linear complexity in the number of neighbors of v: $\Theta\left(N(v)\right)$. Summing over all vertices results in a linear number of operations in the number of edges $m = |\mathcal{E}|$: $\Theta(m)$. The same, linear complexity in the number of edges holds for computing the curvature κ with (24) or (27), and the gradient magnitude $\|\nabla u_{r-1}\|$ with (18). The evaluation of (41) itself at every vertex requires $\Theta(n)$ operations. We also need to take into account the median filtering that is applied to ∇u_{r-1}, κ, and u_r. The calculation of the median of a set of L values requires $\Theta(L)$ operations. Since our neighborhood-based median filter operates on the values from the current vertex v plus all its neighbors, it requires $\Theta(N(v) + 1)$ operations to compute the filtered version of each of the three above quantities at v. Summing over all vertices and over the three applications of the filter yields a total number of $\Theta(n + m)$ operations for median filtering. The same analysis holds for average filtering. Overall, the total complexity for each iteration is

$$\Theta(n + m), \tag{42}$$

which is linear in the number of vertices and in the number of edges.

Our first experiment involves a full grayscale image with four distinct coins (Figure 10(a)). We make a twofold comparison, on the one hand between placing the graph's vertices at random or via watershed transformation, and on the other hand between using random geometric or DT structure for the edges. Results from the four experiments corresponding to all possible combinations are shown in Figures 10(c)–(e) and Figure 10(i). To ensure a fair comparison, the number of randomly placed vertices is approximately the same as in the watershed case. The most accurate segmentation is achieved with DT and watershed-placed vertices (Figure 10(i)), as the boundaries of the objects are captured very well. Using randomly placed vertices and DT structure, or watershed and random geometric structure, also

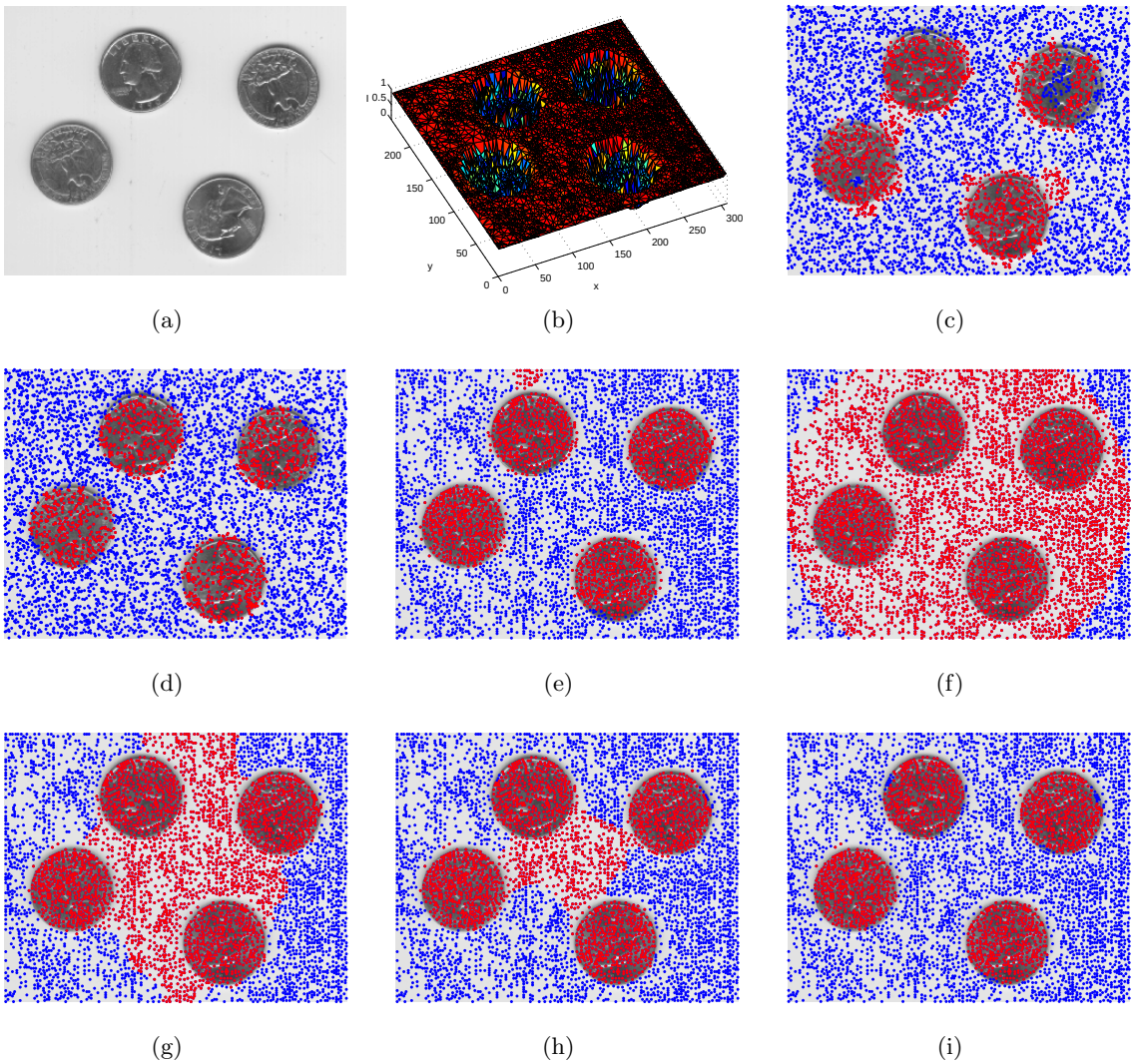


Figure 10. *Detection of grayscale objects on graphs with our GAC algorithm.* (a) *Full grayscale image with four coins;* (b) *image function on watershed-based DT;* (c)–(e) *and* (i) *final detection results overlaid on original image, with detected objects shown in red and background in blue. We use* (c) *randomly placed vertices with random geometric structure,* $\sigma = 0.02$ *and* $\lambda = 0.03$*;* (d) *randomly placed vertices with DT structure,* $\sigma = 0.02$ *and* $\lambda = 0.03$*;* (e) *watershed-placed vertices with random geometric structure,* $\sigma = 0.008$ *and* $\lambda = 0.07$*; and* (i) *watershed-placed vertices with DT structure,* $\sigma = 0.01$ *and* $\lambda = 0.07$*. The total number of iterations to obtain the final segmentation result is* (c) 2200, (d) 4000, (e) 3000, *and* (i) 4000. (f)–(h) *Instances of active contour evolution for watershed-placed vertices with DT structure corresponding to the final detection result in* (i) *after* (f) *initialization of the contour,* (g) 1200 *iterations, and* (h) 2400 *iterations.*

yields decent results (Figure 10(d) and Figure 10(e)). On the contrary, the combination of randomly placed vertices and random geometric structure (i.e., a proper RGG) leads to poor segmentation, in which close objects are not separated and others have holes in their interior (Figure 10(c)). Consequently, the DT structure is preferable to the random geometric one, and

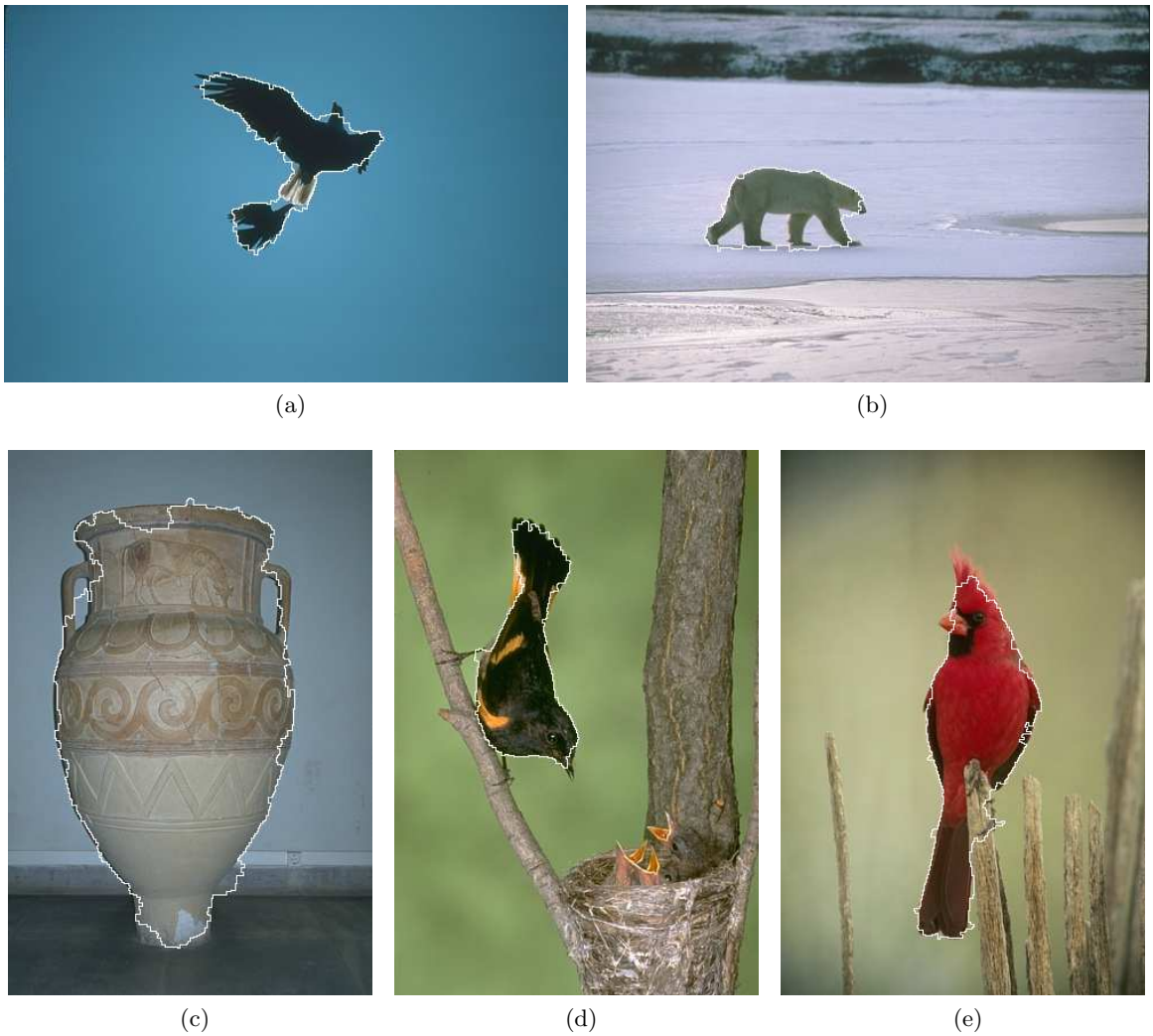
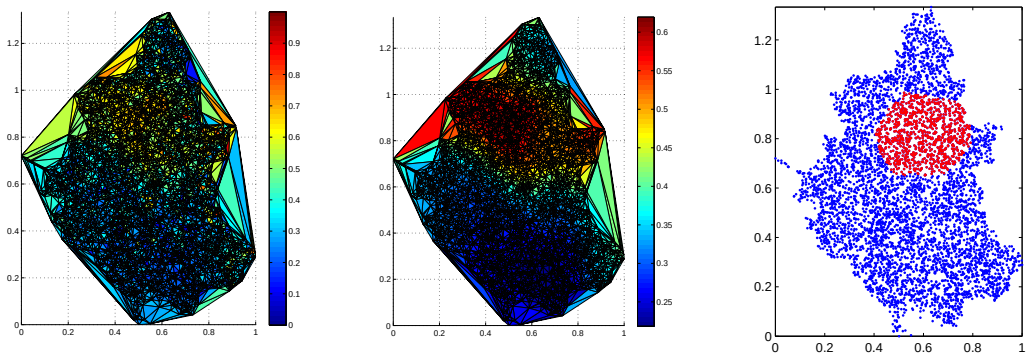


Figure 11. *Segmentation of images from* BSDS500 *dataset with our GAC algorithm. The detected boundary, marked white, is defined as the boundary of the union of the watershed superpixels for which the corresponding graph vertex belongs to the final contour's interior. We use* $\sigma = 0.005$ *and* $\lambda = 0.03$ *in all cases except* (d)*, for which we set* $\lambda = 0.05$*, and* (e)*, for which we set* $\sigma = 0.01$ *and* $\lambda = 0.02$*. The total number of iterations to obtain the final segmentation result is* (a) 3400*,* (b) 2000*,* (c) 2600*,* (d) 4400*, and* (e) 5600*.*

using watershed transformation to place vertices when a full image is available is better than using random placement.

We use the combination of watershed-placed vertices and DT structure for the edges to repeat the above experiment for a collection of natural color images from the Berkeley Segmentation Dataset BSDS500 [35]. The images were converted to grayscale for the application of our method. Segmentation results on the images are presented in Figure 11. In general, our algorithm successfully detects the dominant objects in the images, even though background clutter and thin protrusions or concavities of the objects' boundaries may cause minor inaccuracies.

An interesting application of our method is related to geographical data, where the two spatial coordinates are longitude and latitude and the image function can encode information



(a) (b) (c)

Figure 12. *Segmentation of wind speed data on a graph with our GAC algorithm.* (a) *Normalized data on the graph.* (b) *Smoothed wind speed.* (c) *Final detection result after* 1000 *iterations, with vertices in the contour's interior shown in red and the rest in blue.*

about any type of real-valued signal defined at the vertices of the graph. Such a signal is the average annual wind speed, which is particularly important for locating regions with high wind power potential. Figure 12 presents the application of our algorithm for GACs on graphs to average annual wind speed data on a graph constructed as a DT. We use $\sigma = 0.05$, $\lambda = 0.8$, and $c = 5$ and normalize the coordinates and the values of wind speed. The algorithm detects a cluster that corresponds to a region with relatively uniform and quite high wind speed. Another example regards the signal strength of a cellular network. Figure 13 demonstrates the result of our method for such data on a DT. We again normalize the data and use a very small time step $\Delta t = 10^{-4}$ to guarantee convergence, which requires more iterations until termination than in the previous experiments. In addition, we set $\sigma = 0.03$, $\lambda = 0.02$, and $c = 20$. The segmented set of vertices comprises two regions, the southern of which is characterized by an increased signal strength compared to the rest of the graph. Our approach is tailored for geographical data with arbitrary spatial configuration such as in the above cases, which grants greater flexibility when collecting measurements.

In the concurrent work of [29], the finite element method (FEM) is used to apply the GAC model to arbitrary graphs. Even though this approach inherits the advantages of the well-established framework of finite elements, our method features greater generality and lower computational complexity.

In particular, our method is applicable for any type of connectivity pattern defined through the set of edges $\mathcal{E}$ of the input graph. For instance, when $\mathcal{E}$ is given as part of the input and models connections of vertices that follow an arbitrary, nonlocal pattern which is known a priori, our method still applies. On the contrary, the method in [29] only applies to triangulations; notably, it cannot handle the aforementioned case with arbitrary edge structure, or other commonly used types of graphs such as k nearest neighbor graphs or RGGs.

Furthermore, in case of triangulations, the complexity of each iteration of our algorithm for geodesic active contours is $O(n)$, i.e., *linear* in the number of vertices n, whereas the respective complexity of each iteration of the FEM algorithm in [29] is $O\left(n^2\right)$ according to the analysis therein, i.e., *quadratic* in n. The $O(n)$ complexity for our iterations follows from

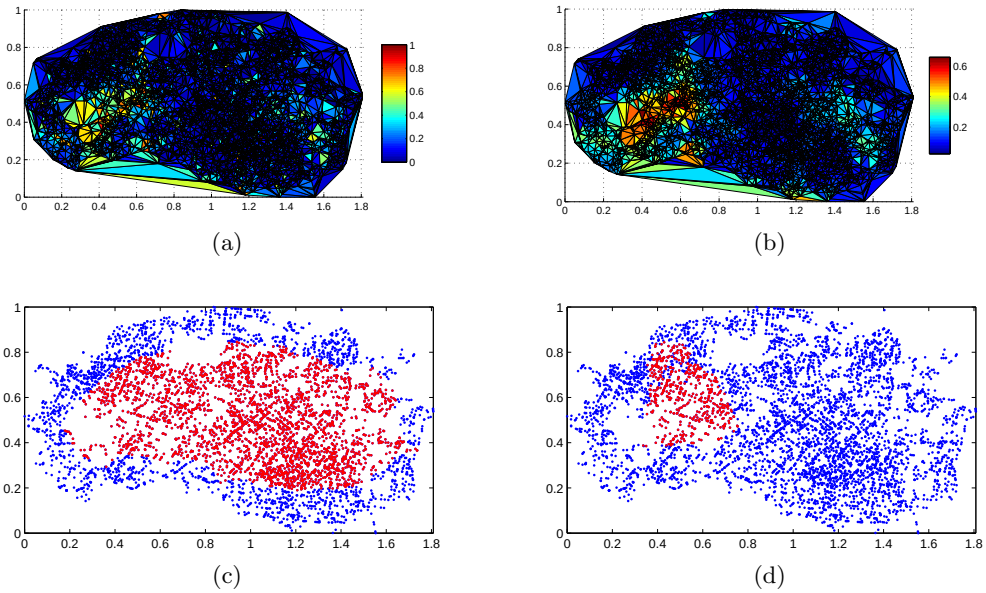


Figure 13. *Segmentation of signal strength data of a cellular network on a graph with our GAC algorithm.* (a) *Normalized data on the graph.* (b) *Smoothed signal strength.* (c) *Initial contour.* (d) *Final detection result after* 40,000 *iterations, with vertices in the contour's interior shown in red and the rest in blue.*

(42) and planarity of triangulations, which bounds the number of edges through $m \leq 3n - 6$. Consequently, our method is much faster than that in [29] even for moderately sized graphs with $n \sim 10^4$, and at the same time it achieves the same segmentation quality as in [29], as shown in the experiments that follow.

In Figure 14 we compare the two approaches on a pair of images from our previous experiments. The graphs are formed as DTs with watershed-placed vertices. The results of our method in Figure 14(a) and Figure 14(c) have already been presented in Figure 10(i) and Figure 11(d), respectively, and we present them again side by side with the respective results of [29] in Figure 14(b) and Figure 14(d) to facilitate comparison. Wherever possible, for [29] we use parameter values that are the same as or at least similar to those we use for our method, so that a fair comparison is ensured. The two methods demonstrate similar segmentation performance on both images. However, our method is faster: for the bird image, the constructed graph comprises $n = 14797$ vertices and the running time is 1001 ± 4 sec for our method and 3159 ± 7 sec for [29].

6.2. Active contours without edges. The algorithm for ACWE [9] on graphs with our approximations includes the following steps:

1. Choose a subset X of $\mathcal{V}$ and initialize the embedding function with the signed distance function from the boundary of X, denoted by u_0. By convention, u_0 is positive inside X.
2. Iterate for $r \in \mathbb{N}$:

$$c_1 = \text{average}\left\{I(v) : u_{r-1}(v) \geq 0\right\}, \quad c_2 = \text{average}\left\{I(v) : u_{r-1}(v) < 0\right\}, \tag{43}$$

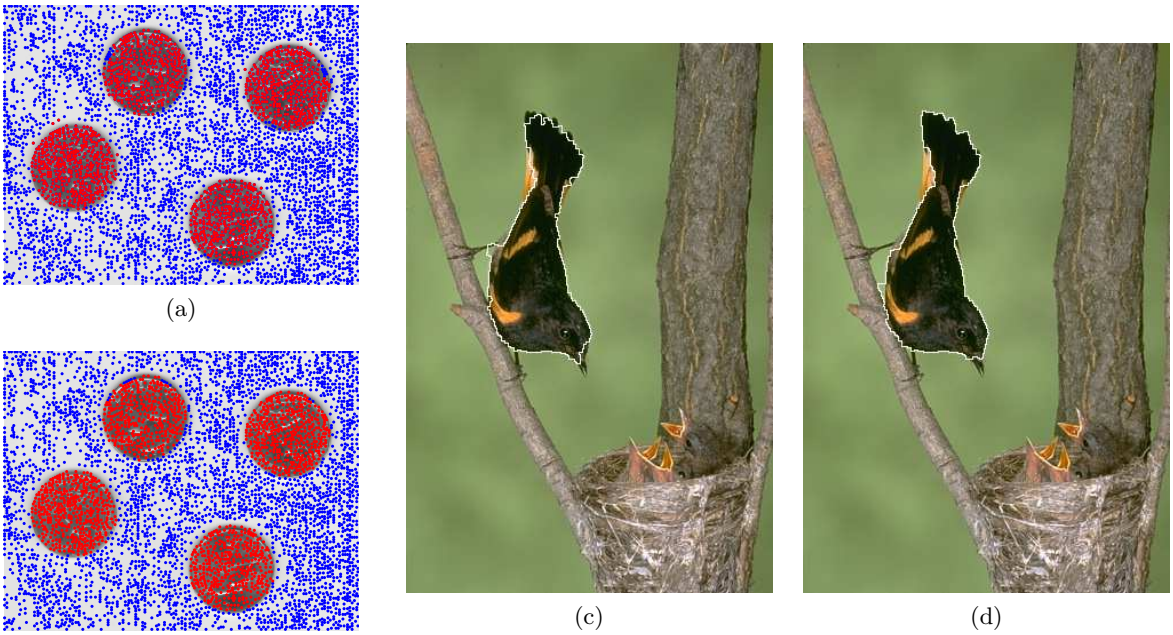


Figure 14. *Comparison of our algorithm for geodesic active contours with the FEM algorithm in* [29]. *For the coins image, the foreground is shown in red and the background in blue and we use* $\sigma = 0.01$ *and* $\lambda = 0.07$ *for both methods.* (a) *Our result after* 4000 *iterations.* (b) *The result of* [29] *after* 301 *iterations, using* $c = 10$ *and* $\Delta t = 0.001$. *For the bird image, the detected boundary is marked white and we use* $\sigma = 0.005$ *for both methods.* (c) *Our result after* 4400 *iterations, using* $\lambda = 0.05$. (d) *The result of* [29] *after* 4001 *iterations, using* $\lambda = 0.07$, $c = 10$, *and* $\Delta t = 0.001$.

$$u_r = u_{r-1} + \Delta t \left(\delta_\epsilon \left(u_{r-1} \right) \left(\mu\kappa - \nu - \lambda_1 (I - c_1)^2 + \lambda_2 (I - c_2)^2 \right) \right) \tag{44}$$

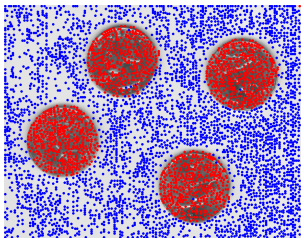
until convergence. In the difference equation (44), Δt, ϵ, μ, ν, λ_1, and λ_2 are positive constants and κ is the curvature of the level sets of u_{r-1}. Moreover, δ_ϵ is the derivative of a regularized version of the Heaviside function which was introduced in [9] as

$$\delta_\epsilon(x) = H'_\epsilon(x) = \left(\frac{1}{2} \left(1 + \frac{2}{\pi} \arctan\left(\frac{x}{\epsilon} \right) \right) \right)' = \frac{\epsilon}{\pi \left(x^2 + \epsilon^2 \right)}. \tag{45}$$

We modify the standard ACWE model for color images, so that it uses all three color channels, instead of reducing the image to a simpler grayscale version. In order for the averages and Euclidean distances to be meaningful, the original RGB values of the vector-valued image function $\mathbf{I}$ at graph vertices are transformed to CIELAB values, as the latter color space is more perceptually uniform. The update equations are modified to operate on multiple color channels:

$$\mathbf{c}_1 = \text{average}\left\{ \mathbf{I}(v) : \, u_{r-1}(v) \geq 0 \right\}, \quad \mathbf{c}_2 = \text{average}\left\{ \mathbf{I}(v) : \, u_{r-1}(v) < 0 \right\} \tag{46}$$

(a)



(b)

(c)

Figure 15. *Segmentation of images with our algorithm for ACWE on graphs.* (a), (c) *Images from* BSDS500 *with detected boundaries marked red and white, respectively.* (b) *Coins image with detected foreground shown in red and background in blue. We use* $\Delta t = 0.05$, $\nu = 0$, $\lambda_1 = \lambda_2 = 1$, *and* $\epsilon = \text{median}_{vw \in \mathcal{E}}\{d(v,w)\}$ *in all cases,* $\mu = 0.1$ *in* (a), *and* $\mu = 0.5$ *in* (b) *and* (c). *The total number of iterations to obtain the final segmentation result is* (a) 1000, (b) 7000, *and* (c) 2000.

and

$$u_r = u_{r-1} + \Delta t \left(\delta_\epsilon \left(u_{r-1} \right) \left(\mu\kappa - \nu - \lambda_1 \left\| \mathbf{I} - \mathbf{c}_1 \right\|^2 + \lambda_2 \left\| \mathbf{I} - \mathbf{c}_2 \right\|^2 \right) \right). \tag{47}$$

In Figure 15, we perform experiments that are analogous to Figure 10(i) and Figures 11(a),(e) by replacing GAC with ACWE and demonstrate the *generality* of our approximations as a base for translating various active contour models to graphs. For the color images from BSDS500 in Figures 15(a),(c), we use our modification of ACWE in (46) and (47) for exploiting multiple color channels instead of converting the images to grayscale. The segmentation results are of high quality, e.g., capturing the thin feathers at the eagle's wing very precisely.

7. Conclusion. In this paper, we introduce approximations of the gradient and curvature terms involved in the level set formulation of active contour evolution models for the case of arbitrary graphs. We provide the conditions under which these approximations are consistent with the true values and, in the case of gradient, the respective rate of convergence in the limit of large graphs. At an algorithmic level, we propose neighborhood-based smoothing filters as an empirical means to improve the accuracy of our approximations, and provide improved implementations of Gaussian smoothing on graphs which account for potential nonuniformi-

ties. We also demonstrate the applicability of active contours on graphs, equipped with our approximations, for segmentation of regular images as well as raw geographical data, using GAC and ACWE as representative models in our experiments.

A remaining challenge in our work is related to the curvature term of active contour models. Both the geometric and the gradient-based approximation which are proposed are proved to converge in probability to the true value of curvature at points with nonzero gradient, which is the first result of this kind to the best of our knowledge. However, due to the cascaded approximation that we perform, a larger amount of noise is injected in the approximate values, which is also reflected in the slower convergence observed in our empirical tests compared to gradient approximation. This forces us to take a very small step in time in some cases when updating the embedding function, which leads to much slower convergence of the active contour algorithm. To overcome this difficulty, a deeper analysis of the curvature term needs to be accomplished, ideally establishing asymptotic bounds on the respective approximation error similar to our bound for the gradient approximation error.

Although our framework applies to any type of graph, our theoretical analysis considers random geometric graphs, whose definition simplifies convergence proofs for our approximations. However, judging from segmentation quality, more regular graph structures, such as DTs, lead to more accurate results. Therefore, an interesting extension of our work is the theoretical study of our approximations on graphs formed as DTs. Furthermore, our initial Gaussian smoothing of the image corresponds to isotropic diffusion, which blurs predominant edges and rounds corners. More faithful preservation of edges in the final segmentation can be ensured by defining anisotropic smoothing on graphs.

Acknowledgments. The authors wish to thank the anonymous reviewers for providing constructive comments which helped to improve this paper. They would also like to thank Nikolaos Kolotouros for making his code available for comparisons.

REFERENCES

[1] E. Arias-Castro, B. Pelletier, and P. Pudlo, *The normalized graph cut and Cheeger constant: From discrete to continuous*, Adv. Appl. Probab., 44 (2012), pp. 907–937.

[2] M. Belkin and P. Niyogi, *Convergence of Laplacian eigenmaps*, in Advances in Neural Information Processing Systems (NIPS), 2006, pp. 129–136.

[3] S. Bougleux, A. Elmoataz, and M. Melkemi, *Discrete regularization on weighted graphs for image and mesh filtering*, in Proc. International Conference on Scale Space and Variational Methods in Computer Vision (SSVM), 2007, pp. 128–139.

[4] Y. Boykov and V. Kolmogorov, *Computing geodesics and minimal surfaces via graph cuts*, in Proc. International Conference on Computer Vision, vol. 1, 2003, pp. 26–33.

[5] Y. Boykov, O. Veksler, and R. Zabih, *Fast approximate energy minimization via graph cuts*, IEEE Trans. Pattern Anal. Mach. Intell., 23 (2001), pp. 1222–1239.

[6] V. Caselles, F. Catté, T. Coll, and F. Dibos, *A geometric model for active contours in image processing*, Numer. Math., 66 (1993), pp. 1–31.

[7] V. Caselles, R. Kimmel, and G. Sapiro, *Geodesic active contours*, Int. J. Comput. Vis., 22 (1997), pp. 61–79.

[8] T. F. Chan, S. Esedoḡlu, and M. Nikolova, *Algorithms for finding global minimizers of image segmentation and denoising models*, SIAM J. Appl. Math., 66 (2006), pp. 1632–1648, https://doi.org/10.1137/040615286.

[9] T. F. Chan and L. A. Vese, *Active contours without edges*, IEEE Trans. Image Process., 10 (2001), pp. 266–277.

[10] C. Couprie, L. Grady, L. Najman, J.-C. Pesquet, and H. Talbot, *Dual constrained TV-based regularization on graphs*, SIAM J. Imaging Sci., 6 (2013), pp. 1246–1273, https://doi.org/10.1137/120895068.

[11] C. Couprie, L. Grady, L. Najman, and H. Talbot, *Power watershed: A unifying graph-based optimization framework*, IEEE Trans. Pattern Anal. Mach. Intell., 33 (2011), pp. 1384–1399.

[12] C. Couprie, L. Grady, H. Talbot, and L. Najman, *Combinatorial continuous maximum flow*, SIAM J. Imaging Sci., 4 (2011), pp. 905–930, https://doi.org/10.1137/100799186.

[13] J. Cousty, G. Bertrand, L. Najman, and M. Couprie, *Watershed cuts: Minimum spanning forests and the drop of water principle*, IEEE Trans. Pattern Anal. Mach. Intell., 31 (2009), pp. 1362–1374.

[14] J. Cousty, G. Bertrand, L. Najman, and M. Couprie, *Watershed cuts: Thinnings, shortest path forests, and topological watersheds*, IEEE Trans. Pattern Anal. Mach. Intell., 32 (2010), pp. 925–939.

[15] J. Cousty, L. Najman, and J. Serra, *Some morphological operators in graph spaces*, in Proc. International Symposium on Mathematical Morphology, 2009, pp. 149–160.

[16] X. Desquesnes, A. Elmoataz, and O. Lézoray, *Eikonal equation adaptation on weighted graphs: Fast geometric diffusion process for local and non-local image and data processing*, J. Math. Imaging Vision, 46 (2013), pp. 238–257.

[17] K. Drakopoulos and P. Maragos, *Active contours on graphs: Multiscale morphology and graphcuts*, IEEE J. Sel. Topics Signal Process., 6 (2012), pp. 780–794.

[18] A. Elmoataz, F. Lozes, and M. Toutain, *Nonlocal PDEs on graphs: From tug-of-war games to unified interpolation on images and point clouds*, J. Math. Imaging Vision, 57 (2017), pp. 381–401.

[19] A. Elmoataz, O. Lézoray, and S. Bougleux, *Nonlocal discrete regularization on weighted graphs: A framework for image and manifold processing*, IEEE Trans. Image Process., 17 (2008), pp. 1047–1060.

[20] A. Elmoataz, M. Toutain, and D. Tenbrinck, *On the p-Laplacian and ∞-Laplacian on graphs with applications in image and data processing*, SIAM J. Imaging Sci., 8 (2015), pp. 2412–2451, https://doi.org/10.1137/15M1022793.

[21] N. García Trillos and D. Slepcev, *Continuum limit of total variation on point clouds*, Arch. Ration. Mech. Anal., 220 (2016), pp. 193–241.

[22] L. Grady, *Random walks for image segmentation*, IEEE Trans. Pattern Anal. Mach. Intell., 28 (2006), pp. 1768–1783.

[23] L. Grady and C. Alvino, *The piecewise smooth Mumford-Shah functional on an arbitrary graph*, IEEE Trans. Image Process., 18 (2009), pp. 2547–2561.

[24] L. Grady and J. Polimeni, *Discrete Calculus: Applied Analysis on Graphs for Computational Science*, Springer Science & Business Media, 2010.

[25] H. J. A. M. Heijmans, P. Nacken, A. Toet, and L. Vincent, *Graph morphology*, J. Visual Commun. Image Represent., 3 (1992), pp. 24–38.

[26] A. N. Hirani, *Discrete Exterior Calculus*, Ph.D. thesis, Caltech, 2003.

[27] J. Jaromczyk and G. Toussaint, *Relative neighborhood graphs and their relatives*, Proc. IEEE, 80 (1992), pp. 1502–1517.

[28] M. Kass, A. Witkin, and D. Terzopoulos, *Snakes: Active contour models*, Int. J. Comput. Vis., 1 (1988), pp. 321–331.

[29] N. Kolotouros and P. Maragos, *A finite element computational framework for active contours on graphs*, submitted.

[30] D. D. Lee and H. S. Seung, *Algorithms for non-negative matrix factorization*, in Advances in Neural Information Processing Systems (NIPS), 2001, pp. 556–562.

[31] F. Lozes, A. Elmoataz, and O. Lézoray, *Partial difference operators on weighted graphs for image processing on surfaces and point clouds*, IEEE Trans. Image Process., 23 (2014), pp. 3896–3909.

[32] O. Lézoray, A. Elmoataz, and V.-T. Ta, *Nonlocal PdEs on graphs for active contours models with applications to image segmentation and data clustering*, in Proc. 2012 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP), 2012, pp. 873–876.

[33] O. Lézoray and L. Grady, *Image Processing and Analysis with Graphs: Theory and Practice*, Digital Imaging and Computer Vision, CRC Press, 2012.

[34] M. Maier, U. von Luxburg, and M. Hein, *How the result of graph clustering methods depends on the construction of the graph*, ESAIM Probab. Statist., 17 (2013), pp. 370–418.

[35] D. Martin, C. Fowlkes, D. Tal, and J. Malik, *A database of human segmented natural images and its application to evaluating segmentation algorithms and measuring ecological statistics*, in Proc. International Conference on Computer Vision, vol. 2, 2001, pp. 416–423.
[36] D. Mumford and J. Shah, *Optimal approximations by piecewise smooth functions and associated variational problems*, Comm. Pure Appl. Math., 42 (1989), pp. 577–685.
[37] S. Osher and J. A. Sethian, *Fronts propagating with curvature dependent speed: Algorithms based on Hamilton-Jacobi formulations*, J. Comput. Phys., 79 (1988), pp. 12–49.
[38] M. Penrose, *Random Geometric Graphs*, Oxford University Press, 2003.
[39] D. Pollard, *Strong consistency of k-means clustering*, Ann. Statist., 9 (1981), pp. 135–140.
[40] J. Shi and J. Malik, *Normalized cuts and image segmentation*, IEEE Trans. Pattern Anal. Mach. Intell., 22 (2000), pp. 888–905.
[41] T. Snijders and K. Nowicki, *Estimation and prediction for stochastic blockmodels for graphs with latent block structure*, J. Classification, 14 (1997), pp. 75–100.
[42] V.-T. Ta, A. Elmoataz, and O. Lézoray, *Nonlocal PDEs-based morphology on weighted graphs for image and data processing*, IEEE Trans. Image Process., 20 (2011), pp. 1504–1516.
[43] D. Ting, L. Huang, and M. I. Jordan, *An analysis of the convergence of graph Laplacians*, in Proc. International Conference on Machine Learning, 2010, pp. 1079–1086.
[44] L. Vincent, *Graphs and mathematical morphology*, Signal Process., 16 (1989), pp. 365–388.
[45] U. von Luxburg, M. Belkin, and O. Bousquet, *Consistency of spectral clustering*, Ann. Statist., 36 (2008), pp. 555–586.
[46] Z. Yang, T. Hao, O. Dikmen, X. Chen, and E. Oja, *Clustering by nonnegative matrix factorization using graph random walk*, in Advances in Neural Information Processing Systems (NIPS), 2012, pp. 1079–1087.
[47] Z. Yuan and E. Oja, *Projective nonnegative matrix factorization for image compression and feature extraction*, in Proc. Scandinavian Conference on Image Analysis, 2005, pp. 333–342.