

Local Symmetry Detection in Natural Images Using a Particle Filtering Approach

Nicolas Widynski, Antoine Moevus, and Max Mignotte

Abstract—In this paper, we propose an algorithm to detect smooth local symmetries and contours of ribbon-like objects in natural images. The detection is formulated as a spatial tracking task using a particle filtering approach, extracting one part of a structure at a time. Using an adaptive local geometric model, the method can detect straight reflection symmetries in perfectly symmetrical objects as well as smooth local symmetries in curved elongated objects. In addition, the proposed approach jointly estimates spine and contours, making it possible to generate back ribbon objects. Experiments for local symmetry detection have been conducted on a recent extension of the Berkeley segmentation data sets. We also show that it is possible to retrieve specific geometrical objects using intuitive prior structural information.

Index Terms—Particle filter, local symmetry detection, ribbon detection.

I. INTRODUCTION

IN HUMAN perception, symmetry is a key element for object recognition as it is one of the fundamental law of the Gestalt theory. Its importance is easy to underline as it is ubiquitous in nature: plants, animals, humans; as well as in man-made objects, buildings, and art [1], [2].

In computational science, reflection symmetry detection has received an increasing amount of attention in the last decade. In this paper, we focus on *local symmetries*, which locally approximate reflection symmetries. Two points form a local symmetry if the angles between their outward normal and the line joining the points are the same [3], [4]. This definition covers *global* as well as *purely local* symmetries. The former refers to a symmetry that holds a whole object, whereas the latter relies on a subset of an object. In other terms, although the definition is local, it does not mean that the detected symmetry cannot be global. While global reflection symmetries can be difficult to retrieve due to noise, occlusion and local deformations, purely local symmetries offer more flexibility and may be more suited for natural images.

A symmetry axis is a spatial characteristic of the shape. However, the symmetry axis does not encode the scale and

contours of the shape. On the other hand, contours only can hardly describe regional properties (e.g. symmetry) and descriptions (e.g. “elongated and curve”) [3]. Ribbon-like objects, or simply *ribbons*, are a representation of 2D plane shapes. A ribbon can be defined by a smooth local symmetry curve, called *spine* (the black plain curve in Fig. 1), and a geometric figure, such as a segment or a disk, called the *generator* (the black dashed structure in Fig. 1). Ribbon shapes benefit of a dual representation: a contour-based and a region-based representation.

In this article, we propose to detect both contour and symmetry curves of ribbon objects in natural images. Ribbons being strongly related to the notion of local symmetry, they are in fact common in natural images, as they can describe a large amount of objects [1]. They are also of interest for the detection of specific tubular structures, for example in medical image analysis (vessels, arteries, colon, ...) [5]–[9], and remote sensing (urban structures, road networks) [10], [11]. Ribbons being a basic element structure that represents well any local object part, their detection could also be employed to retrieve more sophisticated non-ribbon objects using a higher abstraction level detection framework: contours can be used for object segmentation [12], [13]; local symmetries can serve object recognition by providing candidates for structural-based graph matching techniques [14], [15]; and local contours and symmetries could be jointly exploited by a dedicated algorithm, as it has been done in [16] in the context of knowledge transfer between object classes, and in [17] and [18] in the context of object detection using a hierarchical representation.

Our approach spatially tracks symmetries and contours using a local geometric model of ribbons. Local symmetries are assumed to form smooth 1D lines (spines). The use of an iterative algorithm for this task is particularly well-suited to extract smooth local symmetries and to preserve local connectivity properties. We propose to implement a sequential Monte Carlo method, namely a particle filter, to extract one piece of the structure at a time. The particle filter naturally embeds a prior and a spatial transition, which enable to geometrically control the structure of the shape to be extracted. The particle filter is also able to maintain several hypotheses during time, thus overcoming local ambiguities that may happen in occlusions and clutter, for example. The last component of the proposed recursive Bayesian approach is the likelihood function, which is adaptive to the image, in order to confer more importance on visually salient geometric patterns.

This paper is organized as follows. In Section II, we present a brief analysis of previous works proposed in the

Manuscript received December 14, 2013; revised May 3, 2014 and August 18, 2014; accepted October 17, 2014. Date of publication October 30, 2014; date of current version November 6, 2014. The associate editor coordinating the review of this manuscript and approving it for publication was Dr. Adrian G. Bors.

The authors are with the Department of Computer Science and Operations, University of Montreal, Montreal, QC H3T 1J4, Canada (e-mail: widynski@iro.umontreal.ca; moevusan@iro.umontreal.ca; mignotte@iro.umontreal.ca).

Color versions of one or more of the figures in this paper are available online at <http://ieeexplore.ieee.org>.

Digital Object Identifier 10.1109/TIP.2014.2365140

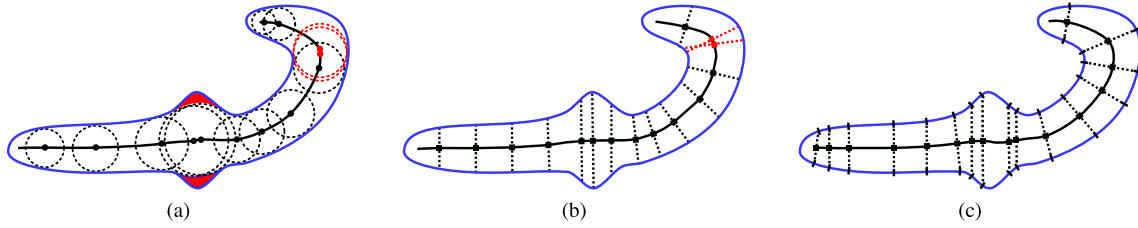


Fig. 1. Examples of spine's recovery for the three well-known types of ribbons. (a) Blum ribbon does not fully recover a shape with a high local positive curvature or strong turn. The red highlighted areas are missed and the red circles intersect at the same point, hence, this shape cannot be recovered with a Blum ribbon. (b) With the Brooks ribbon, generators also intersect. (c) Only the Brady ribbon can fully recover the shape. It is important to notice that Brady ribbons were specifically dedicated to recovery.

symmetry and medial axis detection literature. In Section III, we describe our model of ribbons. Applied sequentially, our model provides consistent and well-formed ribbons. We also define the ribbon prior and transition probability density functions. In Section IV, we detail the contour and object features. These features rely on the information of local gradient, oriented gradient, and textural color. Their purposes are twofold: to evaluate the accuracy of the localization of a contour pair candidate, and to ensure that the ribbon spine is inside an object. In Section V, we explain the ribbon detection algorithm. The detection framework originates from the recent work introduced by Widynski and Mignotte [19], [20], in which the authors proposed to detect contours using a particle filtering technique. We adapt this methodology to our purpose. In Section VI, we show the results obtained on the LS-BSDS300. We also illustrate examples of symmetrical object retrieval using a prior information on their structure. We finally conclude in Section VII.

II. RELATED WORK

We focus on symmetry and medial axes detection algorithms in the recent literature, before making an overview of the definition of a ribbon.

A. Symmetry Detection

Reflection, rotation, and translation symmetries have been the subject of numerous detection methods these recent years. Among the three, reflection symmetry is definitely the most common, as it may be the most ubiquitous in the real world. Reflection symmetry detection methods can be separated into two groups: the first starts from segmented data, and aims at coherently gathering pieces in order to form symmetries with nice geometric properties. The second does not require any pre-processing step, and thus aims at extracting local symmetry features directly from the image, possibly using a shape model. In this section, we mainly focus on un-segmented reflection symmetry detection methods. The reader can find extended bibliography on these subjects in [1] and [21].

Literature on shape representation methods has been particularly prolific these recent years [22], [23]. Their scope of application is appealing: object detection, data compression, tracking, segmentation, ... Among these methods, Trinh and Kimia addressed the object detection problem by extracting medial axis of specific objects in natural images [24]. The shape is learned from a dataset and is modeled using a

graph representation. Since they rely on a structural prior rather than a potentially changing appearance model, such methods are very promising in object detection. But although skeletons contain local symmetries [23], [25], these methods are not intended to extract them all since the detections are driven by a specific shape model.

Detection of symmetries without integrating a model has notably been studied in [21], [26], and [27]. These papers aim at finding major symmetry axes from extracted feature points. Thus, these methods are more adapted to recover global reflection symmetries rather than multiple local symmetries from images [28]. In [29], Podolak *et al.* proposed a planar reflective symmetry transform to measure the degrees of symmetries of arbitrary 3D shapes. All the planes passing through the object bounding volume being evaluated, this representation creates an interesting bridge between local and global symmetries, yet it remains to be applied to the detection problem in natural images.

Recently, Tsogkas and Kokkinos proposed a soft local symmetry detector using a learning-based approach operating jointly on several scales [30]. The features are based on the well known gPb contour detection algorithm [12]. Although we compare our approach to theirs, the problems are not exactly the same. They focus on local symmetry detection, and not on ribbon detection, meaning that their approach cannot recover an object nor a scale information. The authors also proposed to use the Berkeley Segmentation Dataset (BSDS300) [31] to learn and evaluate their local symmetry detection algorithm. To create the symmetry ground truth dataset, they first manually selected elongated objects from the set of human-annotated segments available in the BSDS300. Then, they used a skeletonization algorithm to extract the medial structure of the objects. We further refer to this dataset as the LS-BSDS300, which stands for Local Symmetries in the Berkeley Segmentation Dataset 300. As in [30], we also compare our approach to the methods proposed by Lindeberg [32] and Levinshtein *et al.* [33]. The former defines a multi-scale ridge detector, hence yielding to a 1D output of the structure. The latter approximates symmetrical regions using fitted ellipses from which it retrieves the major axes.

The problem of detecting symmetries and ribbons together has been previously addressed in the context of the detection of thin elongated structures such as roads, vessels or other tubular objects [5]–[8], [10], [34], but to our knowledge, not in natural images. The comparison with our approach is

even more relevant in the case of [5]–[7], since the proposed methods employ a particle filtering technique to extract vessels and arteries in 3D computation tomography data. However, the aforementioned methods are often semi-automatic, thus making them designed for single extraction purposes, and/or are object-specific, which make their adaptation in natural images not trivial.

In this work, we stress that simultaneously retrieving the contours and their associated local symmetries into a single framework is an interesting challenge as it aims at representing objects by compact and consistent structures at different levels of abstraction. Our goal is also to propose a generic geometric model that could be easily exploited to extract specific ribbons from natural images. This prior could be based on many basic filtering criteria, such as the length of the symmetry, its minimal/maximal curvature, the object area, its perimeter, its thickness, ...

B. Ribbon

The definition of a local symmetry is related to the notion of ribbon. Past literature [1], [3], [35], [36] reports several definitions of local symmetries, each one being associated with a different type of ribbon [4], [37]. According to the definition proposed in [4], a ribbon is a plane shape generated by translating a geometric figure, the *generator*, along a plane curve, the *spine*. To ensure that the generated shape is a proper ribbon, the generation process should respect the following rules [37]:

- the size and the orientation of generators should vary smoothly along the spine;
- generators should not intersect each others;
- generators should not contain one another.

Smooth variations of the size and the orientation imply the continuity of the shape contours. The two latter rules ensure that the shape is well formed: the non-intersection rule forbids objects with holes (which are not ribbons objects), and the non-inclusion rule imposes each generator center to contribute to the global shape of the ribbon (this way, the ribbon structure is very similar to that of the spine).

Fig. 1 illustrates the most popular types of ribbons: *Blum ribbons*, *Brooks ribbons*, and *Brady ribbons*. Each of these is defined by a different generator: *Blum ribbons* use a circle as a generator [35], *Brooks ribbons* use a line segment with a fixed angle (often perpendicular) to the spine [36] and *Brady ribbons* use a line segment whose extremities make equal angles with the contour of the ribbons [3].

Rosenfeld [37] explains that these ribbons have different properties because they were designed for different purposes. For instance, Blum ribbons have the property to be uniquely recoverable from the contours, *i.e.* only one spine can be recovered from a given ribbon. However, these ribbons can represent a limited class of objects. Brooks ribbons are more flexible than Blum's, and are easy to generate, but are locally ambiguous due to their lack of constraint on the contour layouts. Thus, they may not result in a unique recovery of the spine. Finally, Brady ribbons define the most flexible class of ribbons. They describe well a wide range of

shapes, but their generation is neither unique nor obvious compared to other ribbons. To generate Brooks and Blum ribbons only the size (radius) of the generator, l , is required before sweeping it out on the spine, whereas for Brady ribbons the slope of the generator with the spine, $\tan \theta$, is also required. Moreover, the end points of the generator are the sides of the ribbon and their slopes are equal to [37]:

$$\frac{d(y \pm l \sin \theta)}{d(x \pm l \cos \theta)} = \frac{y' \pm l' \sin \theta \pm l \theta' \cos \theta}{1 \pm l' \cos \theta \mp l \theta' \sin \theta}$$

with u' the local derivative of u . These slopes are referred to by $\tan \alpha_a$ and $\tan \alpha_b$, and they must respect the equal angle condition, meaning that $(\alpha_a + \alpha_b)/2 = \theta$. In practice, this equation is generally not easy to solve and this confirms that Brady's intention was not to define a generative model of ribbons.

The process of retrieving ribbons from images is different regarding the type of ribbon considered. Blum and Brady ribbons belong to the bottom-up category, since their detection algorithm may be divided into the following procedures: 1) a pre-processing step is used to extract binary contours, 2) multiple local symmetries are obtained according to their respective definitions, and finally, 3) the middle points on the axes of symmetry, the locus, are joined all together to obtain the medial axis. Brooks ribbons may be used in a top-down approach, by 1) generating candidates, and 2) validating or invalidating the candidates by matching their local characteristics to the image. The former type of approach may not be suited in the case of natural images, since retrieving binary contours may produce erroneous results, and yet these noisy measurements may increase the detection errors of the ribbon extraction algorithm. Conversely, generating candidates and then estimating their likelihood is not vulnerable to propagation errors from a pre-processing step and may benefit from a higher level of decision process.

To summarize, we expect that a ribbon satisfies the following three properties: 1) it should be expressive enough in order to be able to model a large range of ribbon-like objects; 2) it must be generative in order to be able to propagate ribbon candidates; 3) it should not be ambiguous (otherwise, it could lead to an ill-posed problem). Blum ribbons offer neither 1) nor 2), since they can represent a limited class of objects and are not generative. Brooks ribbons do not offer 3), since they are highly ambiguous. Finally, Brady ribbons do not offer 2). Therefore, we need to define a simple generative but yet expressive model of ribbons.

In our detection algorithm, ribbons are detected sequentially: candidate parts of a ribbon are generated and then weighted in a Bayesian framework. We base our ribbon definition on that of Brooks, but we add the constraint that the generator makes equal angles with the contours. This constraint is formulated in the Brady ribbons, and results in reducing the local ambiguities of the spine location. This way, we define a convenient generative ribbon model which can easily reconstruct contours of the shape and keep a good description capacity of smooth local symmetries for a large class of ribbons.

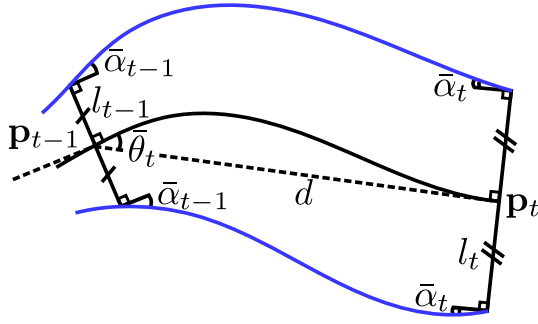


Fig. 2. Geometric model of a ribbon $\mathbf{r}_t = (\mathbf{p}_{t-1}, \theta_t, l_t, \bar{\alpha}_t)$. $\bar{\theta}_t$ is the orientation shift $|\theta_t - \theta_{t-1}|$, l_t is the generator radius, and $\bar{\alpha}_t$ the deviation angle from θ_t , i.e. $\alpha_t = \theta_t \pm \bar{\alpha}_t$ with α_t being the angles between the contour curves and the abscissa axis. Contour points generated at the spine points \mathbf{p}_{t-1} and \mathbf{p}_t define local symmetries. Spine and contours are interpolated using cubic Hermite splines.

III. A MODEL OF RIBBONS

We present in this section a geometric model used to represent ribbons. Since we want to efficiently retrieve these structures from natural images, and be able to generate suitable candidates, a good geometric model would confer a trade-off between simplicity and power expressivity.

A. Geometric Model

Let us introduce some notations. Since the detection method is stochastic, the random variables are indexed by the time t . However, in our case, the tracking is not spatio-temporal but only spatial, thus, the time t is a t^{th} step iteration of our detection algorithm. Hence the set $\{\mathbf{p}_t, t \in \mathbb{N}\}$ represents a sequence of spine points. A spine point is a local smooth symmetry point, with respect to the Brady's definition [3], i.e. it is defined as the barycenter of the line segment joining two contour points that makes equal angles with the sides of the shape. The smooth spine is interpolated using a cubic Hermite spline whose data points and tangents are defined by the spine points. Hence, the tangent direction at the point \mathbf{p}_t is equal to $(\mathbf{p}_t - \mathbf{p}_{t-1})/\|\mathbf{p}_t - \mathbf{p}_{t-1}\|$.

Let θ_t be the angle defined by the tangent direction, i.e. θ_t is the angle between \mathbf{p}_t and the abscissa axis with respect to the center point \mathbf{p}_{t-1} . We consider that the distance between two consecutive spine points is a parameter d of the method, so that the point \mathbf{p}_t is defined by the polar system (θ_t, d) and with respect to \mathbf{p}_{t-1} . We would also like to retrieve the symmetric contours from the spine. To do that, we add a local symmetry radius parameter, l_t , into our model. At a certain point \mathbf{p}_t , the contour points are defined along the normal of $(\mathbf{p}_{t-1}, \mathbf{p}_t)$, i.e. at an angle $\theta_t \pm \pi/2$ and at a distance l_t . These contour points are the extremities of the ribbon at time t . The angles between their respective contour curves and the abscissa axis are defined as $\alpha_t = \theta_t \pm \bar{\alpha}_t$, which is consistent with the Brady ribbons. The angles α_t are expressed as a deviation angle from θ_t because we assume that for most of natural ribbons, contours are locally parallel to the spine points, implying in that case $\bar{\alpha}_t = 0$. Finally, the contours are interpolated using cubic Hermite splines, whose angles of the tangent are defined by the angles α_t . The model is illustrated in Fig. 2.

B. State Model

In conformity with the notation system introduced in the previous section, the vector state \mathbf{s}_t of a spine axis at a time t is defined as $(\mathbf{p}_t, \theta_t, d)$, where \mathbf{p}_t is the spine point, θ_t the angular coordinate, and d the radial coordinate. These three variables define the axis of a local symmetry. Since the point \mathbf{p}_t is defined by the first point \mathbf{p}_0 and the polar coordinates (θ_t, d) , we discard it from the notation system. Plus, the distance d is fixed, and thus does not need to be estimated. In these conditions, the spine trajectory $\mathbf{s}_{1:t}$ is simply characterized by the sequence $(\mathbf{p}_0, \theta_1, \dots, \theta_t)$. However, in Section V, we will see that the first point of a trajectory is not necessarily \mathbf{p}_0 , since the trajectory corresponds to a set of spines instead of only one spine.

From the spine axis \mathbf{s}_t , we can define a ribbon portion, \mathbf{r}_t , as $\mathbf{r}_t = (\mathbf{s}_t, l_t, \bar{\alpha}_t) = (\theta_t, l_t, \bar{\alpha}_t)$, where l_t is the radius of the ribbon, and $\alpha_t = \theta_t \pm \bar{\alpha}_t$ the angles at the contours.

C. Probabilistic Model

There are two ways to define the prior and transition distributions $p(\mathbf{r}_t)$ and $p(\mathbf{r}_t|\mathbf{r}_{t-1})$. The first one is to learn these distributions on a ribbon shapes database. The spine can for example be retrieved using the Brady's algorithm [3], and the other conditional distributions can be approximated using a shape reconstruction error criterion. However, in this work, we propose to define generic distributions whose parameters are estimated by a trial-and-error procedure on the LS-BSDS300 training dataset. The reason is that in this way, the shape parameters can intuitively be tuned, which is suitable when one wants to detect a specific category of objects based on some shape criteria, as we will experiment in Section VI.

1) *Prior Model*: We define the prior distribution $p(\mathbf{r})$ as a product of independent terms:

$$p(\mathbf{r}) = p(\mathbf{p}_0) p(\bar{\alpha}) p(\theta) p(l) \\ = \mathcal{U}(\mathbf{p}_0; \Omega) \mathcal{N}(\bar{\alpha}; 0, \sigma_{\bar{\alpha}}) \mathcal{U}(\theta; [0, 2\pi[) \mathcal{U}(l; [l^m, l^M]) \quad (1)$$

Here we assume that the prior distribution of the spine point \mathbf{p}_0 follows a uniform distribution in the image domain Ω . The distribution of the angle θ is uniform in $[0, 2\pi[$, meaning that no initial direction is suggested. The prior distribution $p(l)$ is uniform in the interval $[l^m, l^M]$. The angle $\bar{\alpha}$ on the contours follows a normal distribution centered in 0 (in that case the tangent contours are equal to the one of the spine), and of standard deviation $\sigma_{\bar{\alpha}}$.

2) *Transition Model*: The joint transition distribution is used to propagate a spatial information from $t-1$ to t and is defined as:

$$p(\mathbf{r}_t|\mathbf{r}_{t-1}) = p(l_t, \bar{\alpha}_t|\mathbf{s}_t, l_{t-1}) p(\mathbf{s}_t|\mathbf{s}_{t-1}) \\ = p(\bar{\alpha}_t) p(\theta_t|\mathbf{r}_{t-1}, \theta_{t-1}, l_t) p(l_t|l_{t-1}) \\ = \mathcal{N}(\bar{\alpha}_t; 0, \sigma_{\bar{\alpha}}) \tilde{\mathcal{N}}(\theta_t; \theta_{t-1}, \sigma_{\theta}) \tilde{\mathcal{N}}(l_t; l_{t-1}, \sigma_l) \quad (2)$$

The transition of the radius is defined as a truncated normal distribution $\tilde{\mathcal{N}}_{l^m, l^M}(l_t; l_{t-1}, \sigma_l)$, with $[l^m, l^M]$ the support interval. The transition of θ_t is then defined as a normal truncated distribution $\tilde{\mathcal{N}}_{\theta_t}(\theta_t; \theta_{t-1}, \sigma_{\theta})$ whose endpoints are

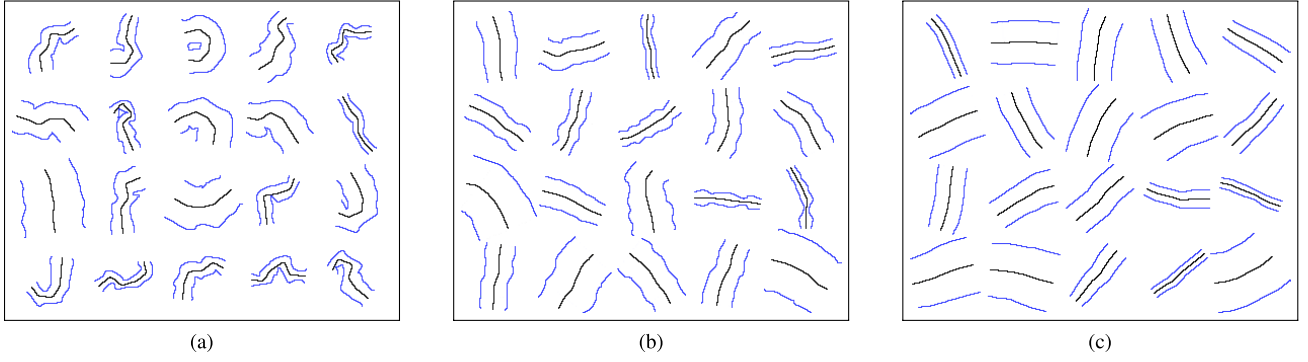


Fig. 3. Simulation of ribbons using different parameters for the prior and transition models. Other parameters have been set to $d = 9$ pixels, $t = 1, \dots, 5$, for all the simulations. Each simulation has been laid out on a 75×75 pixel grid. The simulations in (b) correspond to the estimated parameters on the LS-BSDS300 training dataset. (a) ($\sigma_{\bar{\alpha}} = \pi/4, \sigma_{\theta} = \pi/2, \sigma_l = 1.5$). (b) ($\sigma_{\bar{\alpha}} \simeq \pi/8, \sigma_{\theta} \simeq \pi/16, \sigma_l = 0.18$). (c) ($\sigma_{\bar{\alpha}} = 0, \sigma_{\theta} = \pi/32, \sigma_l = 0.05$).

computed using $(\mathbf{r}_{t-1}, \theta_{t-1}, l_t)$ in order to induce a generator (connecting line of the two contour point extremities at t) that does not intersect with the one at $t - 1$. This ensures generating locally well formed ribbons, and implies that the thickness of a generated ribbon cannot exceed twice the radius of curvature of its spine. Fig. 3 illustrates ribbons generated using the prior model (Eq. 1) and the transition model (Eq. 2) for several combinations of parameters.

IV. COMPUTING FEATURES ON RIBBONS

In this section, we propose to extract five symmetry features. We note $\overleftarrow{c}(\mathbf{r})$, $\overrightarrow{c}(\mathbf{r})$, the left and right contours, respectively, associated with the medial axis \mathbf{r} , and $\overleftrightarrow{c}(\mathbf{r})$ whether it can be one or the other. The sides of the contours are arbitrarily defined relatively to the direction of traversal of the ribbon. The time subscript t is omitted for more clarity. The first three features are computed near the contours: the local gradient f^1 , the oriented gradient f^2 , and the textural gradient f^3 . The textural features f^4 and f^5 ensure that the medial axis is correctly located inside an object. All the features are illustrated in Fig. 4.

A. Contour Features

First, we need to define a merging operator to combine a pair of contour features $(f^i(\overleftarrow{c}(\mathbf{r})), f^i(\overrightarrow{c}(\mathbf{r})))$. We define it as the harmonic mean between these two quantities:

$$f^i(\mathbf{r}) = \frac{2f^i(\overleftarrow{c}(\mathbf{r}))f^i(\overrightarrow{c}(\mathbf{r}))}{f^i(\overleftarrow{c}(\mathbf{r})) + f^i(\overrightarrow{c}(\mathbf{r}))}, \quad \forall i = 1, 2, 3$$

The choice of a harmonic mean relies on the fact that when computed on just two values, this merging operator is a good trade-off between a minimum and an average operator. It forces the contour features to get a high and consistent response, but it is also less drastic than a minimum operator when one of the two contours is cluttered or occulted.

1) *Local Gradient*: The local gradient is a smoothed gradient norm $|g * \nabla I|$ of the image I . The feature $f^1(\overleftrightarrow{c}(\mathbf{r}))$ is computed along a contour $\overleftrightarrow{c}(\mathbf{r})$ of length L :

$$f^1(\overleftrightarrow{c}(\mathbf{r})) = \frac{1}{L} \sum_{i=1}^L \left| \nabla g * I(\overleftrightarrow{c}^i(\mathbf{r})) \right|$$

where $\overleftrightarrow{c}^i(\mathbf{r})$ is the i^{th} point of the contour $\overleftrightarrow{c}(\mathbf{r})$. For an image I , we take the maximum gradient value on each point among the different channels.

2) *Oriented Gradients*: We propose to compute a histogram distance between two oriented gradient patches. The histogram of oriented gradients [38] summarizes a textural information. The distance between two adjacent patches of oriented gradients is known to give a good indication of the presence of a contour [12]. The procedure proposed here is slightly different from that of the pioneers Dalal and Triggs [38] and has been used in [39]. For a point $\overleftrightarrow{c}^i(\mathbf{r})$ of a contour $\overleftrightarrow{c}(\mathbf{r})$, we consider the two sides of its normal segment, one related to the outside the object, n_{out} , and one related to the inside, n_{in} . Let $h_O[a] = \{h_O^r[a]\}_{r=1}^R$ be the histogram of a set of pixels a , where r is the bin index of a histogram of length $R = 4 \times R_M$, with 4 the number of considered orientations (vertical, horizontal, and two diagonals) and R_M the number of magnitude bins. The distance between two adjacent oriented gradient patches is defined as:

$$f^2(\overleftrightarrow{c}(\mathbf{r})) = \frac{1}{L} \sum_{i=2}^{L-1} d_B(h_O[n_{\text{in}}(\overleftrightarrow{c}^i(\mathbf{r}))], h_O[n_{\text{out}}(\overleftrightarrow{c}^i(\mathbf{r}))])$$

with

$$d_B(h[a], h[b]) = \left[1 - \sum_{r=1}^R \sqrt{h^r[a] h^r[b]} \right]^{1/2}$$

the Bhattacharyya distance between two histograms. In practice, we consider normal segments with 5 pixels width in order to get enough samples to compute the histograms.

3) *Textural Gradients*: The last contour feature integrates the color distribution of the neighborhood of a contour to help determine whether the candidate is located on a true contour or not. We consider two CIE Lab color histograms of adjacent patches, ideally one located inside the contour, the other one outside. The dimension of the outside patch is fixed, while the inside patch depends on the radius of the object. Let h_T be a textural color histogram of length R equals $R_T \times R_T \times R_T$, with R_T the number of bins by channel. The distances between the pair of histograms along the barycenter contour curve is defined as:

$$f^3(\overleftrightarrow{c}(\mathbf{r})) = d_B(h_T[n_{\text{in}}(\overleftrightarrow{c}^b(\mathbf{r}))], h_T[n_{\text{out}}(\overleftrightarrow{c}^b(\mathbf{r}))])$$

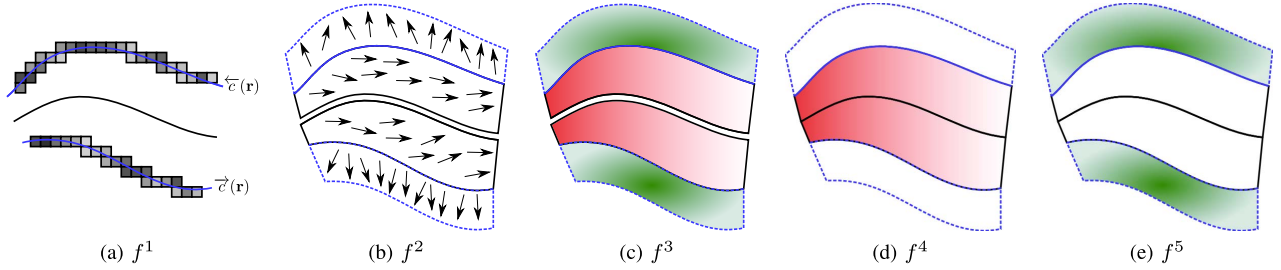


Fig. 4. Features. $\overleftarrow{c}(\mathbf{r})$ and $\overrightarrow{c}(\mathbf{r})$ are the left-side and right-side contours, respectively. The figures represent the five proposed features f^j , $j = 1, \dots, 5$. (a) f^1 is the local gradient on both sides of the contours. (b) f^2 is a combination of two histogram distances of the oriented gradients computed inside and outside the object. (c)-(e) The three last features are extracted from the color histograms of these regions: f^3 ensures the presence of the contours by a combination of the two distances inside / outside the object, the other ones ensure the presence of an object by a distance inside / inside (f^4) and another outside / outside (f^5).

with $\overleftarrow{c}^b(\mathbf{r})$ the barycenter point of the contour $\overleftarrow{c}(\mathbf{r})$. In practice, to limit the additional computational cost of considering a radius-dependent patch size, the regions are approximated by considering rectangles centered on the barycenter points, and 16 possible orientations in $[0, \pi]$. This feature is computed at four different size patches in order to integrate a more global information.

B. Object Features

The feature f^4 ensures that the medial axis divides a region into parts sharing the same textural information. It helps detecting symmetries inside consistent regions. This is done by considering color histograms extracted from two patches located on the inner sides of the contour:

$$f^4(\mathbf{r}) = 1 - d_B(h_T[n_{in}(\overleftarrow{c}^b(\mathbf{r}))], h_T[n_{in}(\overrightarrow{c}^b(\mathbf{r}))])$$

Finally, the feature f^5 assumes that the object is superposed to a unique background. The same hypothesis is made in [30], and from experiments, we found that it greatly reduces the number of undesired symmetry detections in the background. We also use color histograms to check that the background histograms on each contour side match:

$$f^5(\mathbf{r}) = 1 - d_B(h_T[n_{out}(\overleftarrow{c}^b(\mathbf{r}))], h_T[n_{out}(\overrightarrow{c}^b(\mathbf{r}))])$$

V. RIBBON DETECTION ALGORITHM

In this Section, we present a local symmetry detection algorithm using a particle filtering technique. The algorithm is based on previous developments in which we proposed the PFCF framework [19], [20], standing for Particle Filter for Contour Detection, in order to detect contours in natural images. We turn the ribbon detection problem into the estimation of the distribution of the joint local reflection symmetries and radiuses $\mathbf{r}_{1:t} = (\theta_{1:t}, l_{1:t}, \tilde{a}_{1:t})$ considering a set of observations \mathbf{y} . The detection algorithm is composed of three estimation steps:

- The approximation of the likelihood functions, which are defined using the features described in Section IV.
- The approximation of the initialization distribution of the local symmetries $p(\mathbf{r}_1|\mathbf{y})$ is performed using a Monte-Carlo sampling approach. Ribbons are first generated using the prior distributions defined in Section III, and are then weighted using the likelihood functions.

The joint likelihood function and the initialization distribution are both estimated during this step.

- The approximation of the trajectory distribution $p(\mathbf{r}_{1:t}|\mathbf{y})$ is performed using a sequential Monte-Carlo technique, the so-called particle filter. For each step t , ribbons are first generated using an importance function that is related to the transition distributions defined in Section III, and are then weighted using, in particular, the same joint likelihood function that was used in the initialization step. Each time a ribbon has been entirely extracted, the detection is automatically reinitialized on a new ribbon. Local symmetries are therefore iteratively and spatially tracked until the particle filter reaches a stopping criterion.

A. Likelihood Function

This section presents how estimating tail distributions of the features, or complementary cumulative distribution functions of the features, are used to overcome two classical drawbacks of particle filtering techniques. First, if a particle gets lost, there is no mechanism in the particle filter to handle it. This may happen, for example, when the extraction of the last ribbon was completed at $t-1$, and then needs to move on a new spine. This is partially due to the weight normalization step, intrinsic to particle filters, which avoid any global comparison. Second, particle filters do not embed a stopping criterion, which is necessary in our application. We denote by $\mathbf{y}^j(\mathbf{r})$ the observation associated to the ribbon \mathbf{r} . The observation $\mathbf{y}^j(\mathbf{r}_t)$ is given by the tail distribution associated to the feature j :

$$\mathbf{y}^j(\mathbf{r}_t) = P(\mu^j > f^j(\mathbf{r}_t)) \quad (3)$$

where μ^j is a random variable taking values in \mathbb{R}^+ and f^j is the j^{th} feature previously defined in Section IV. In this case, the tail distribution of the j^{th} feature response can be seen as a false alarm distribution. Its value tends to 0 as the feature response tends to $+\infty$. In this last example, we say that the feature response is *meaningful*, conveying the idea that significant events, according to the human perception, are rare. This is the idea of the Helmholtz principle, and has notably been used in an *a contrario* framework [40], [41]. Using this contextual information, these J distributions will be exploited to decide whenever some particles need to be reassigned to new objects, and when to stop the detection algorithm.

We define the likelihood joint $p(\mathbf{y}|\mathbf{r}_t)$ which measures the adequation between the joint observation $\mathbf{y} = (\mathbf{y}^1, \dots, \mathbf{y}^J)$,

and the state \mathbf{r}_t at a time t . Although the vector observation \mathbf{y}^j designates observations from the whole set of ribbons included in an image, the likelihood is only evaluated on the ribbon \mathbf{r}_t [20], [42]:

$$p(\mathbf{y}|\mathbf{r}_t) = \prod_{j=1}^J p(\mathbf{y}^j(\mathbf{r}_t)) \quad (4)$$

where the observations $(\mathbf{y}^1, \dots, \mathbf{y}^J)$ are assumed conditionally independent given (\mathbf{r}_t) . The marginal likelihood $p(\mathbf{y}^j(\mathbf{r}_t))$ is defined using its associated tail distribution:

$$p(\mathbf{y}^j(\mathbf{r}_t)) \propto \exp\left(-\lambda^j P(\mu^j > f^j(\mathbf{r}_t))\right) \quad (5)$$

where f^j is the j^{th} feature defined in Section IV, and $\lambda^j \in \mathbb{R}^+$ a coefficient that weights the importance of the feature j . The marginal likelihood $p(\mathbf{y}^j(\mathbf{r}_t))$ gives low density values for high values of the tail distribution, i.e., the density values are low for common events, thus giving more weight to rare relevant events. The tail distributions are estimated using a Monte Carlo procedure, along with the initialization distribution, as we will see in the next section.

B. Initialization Distribution

Particle filters assume that an initialization distribution is known. In real-world applications, this might not be the case, so we need to estimate it. We could also simply use the prior distribution $p(\mathbf{r}_1)$ defined in Section III, but since by definition they are not conditioned by the observation \mathbf{y} , it could be inefficient. This leads the estimating of the initialization distribution $p(\mathbf{r}_1|\mathbf{y})$ using a classic Bayesian decomposition:

$$\begin{aligned} p(\mathbf{r}_1|\mathbf{y}) &\propto p(\mathbf{y}|\mathbf{r}_1) p(\mathbf{r}_1) \\ &= p(\mathbf{y}|\mathbf{r}_1) p(\bar{\alpha}_1) p(\theta_1) p(l_1) p(\mathbf{p}_0) \end{aligned} \quad (6)$$

with the prior distribution $p(\mathbf{r}_1)$ defined in Equation 1 and the likelihood $p(\mathbf{y}|\mathbf{r}_1)$ defined in Equation 4. To alleviate the problem of dimensionality, generated ribbons from the initialization distribution are rectangular, i.e. $l_0 = l_1$ and $\bar{\alpha}_0 = \bar{\alpha}_1$. The tail and initialization distributions are approximated using a Monte Carlo importance sampling procedure by:

- Generating samples $\{\mathbf{r}_1^{(n)}\}_{n=1, \dots, N_i}$ according to the prior distribution $p(\mathbf{r}_1)$;
- Evaluating the feature responses $\{f^j(\mathbf{r}_1^{(n)}), \forall j = 1, \dots, 5\}_{n=1, \dots, N_i}$ of every sample, which results in the approximation of the tail distributions $\{P(\mu^j > f^j(\mathbf{r}_t)), \forall j = 1, \dots, 5\}$;
- Computing the joint likelihood $p(\mathbf{y}|\mathbf{r}_1^{(n)})$ of every sample, which results in the approximation of the initialization distribution $p(\mathbf{r}_1|\mathbf{y})$.

The initialization distribution implies that the spatial tracking of the ribbons can begin at any location of an object. As we will see in Section V-C, the potential negative effect of this behavior is coped by the sequential aspect of the detection algorithm that allows the tracking to pursue the detection on other object parts at different times, and by the use of multiple particle filters to add diversity to the posterior distribution. Moreover, the method implicitly favors to find the longest

symmetry axis of a ribbon: if the probability of initialization is identical at any location of the ribbon axes, the effect of this tendency is proportional to the length of the symmetry axes. This means that minor axes can still be detected, but with a lower probability. From a ribbon representation perspective, this is consistent with the fact that one generally wants to limit the ambiguities of the ribbon model, and that the longest axes should be preferred, since they represent the object more intuitively [37].

C. Detection Algorithm: Trajectory Distribution

First, we need to make an adjustment of the hidden state. In order to detect all the ribbon objects of an image, the algorithm needs a mechanism to detect when the extraction of a ribbon has been completed. Let j_t be a binary random variable that designates the current work state of the ribbon extraction. If $j_t = 0$, the spatial tracking goes on normally. Otherwise, it means that the extraction of the current object is completed and needs to be reassigned to a new object. This point will be further clarified in the current section. The purpose of this section is to estimate the trajectory distribution $p(\mathbf{x}_{1:t}|\mathbf{y})$ using a particle filtering technique, with $\mathbf{x}_t = (\mathbf{r}_t, j_t) = (\theta_t, \bar{\alpha}_t, l_t, j_t)$. The modeling of the state and the definition of the prior and transition distributions in the original PFCD approach [19] are conceptually different from the one proposed here.

In sequential Monte Carlo methods, $(\mathbf{x}_t)_{t \in \mathbb{N}^+}$ and $(\mathbf{y}_t)_{t \in \mathbb{N}^+}$ are modeled as stochastic processes, where the index t represents the time. In image processing, particle filters are essentially known for tracking applications. Basically, the goal is to estimate the position of an object over time by empirically approximating the posterior density distribution and next by computing the Monte Carlo expected value. However, in the context of symmetry and ribbon detections, the tracking is purely spatial as it aims at estimating the positions of ribbons by locally propagating small parts of ribbons. Thus, the time index is removed from (\mathbf{y}_t) , as the observations could all be gathered at the beginning of the estimation process. The recursive trajectory estimation consists in estimating the following posterior distribution:

$$p(\mathbf{x}_{1:t}|\mathbf{y}) \propto p(\mathbf{x}_{1:t-1}|\mathbf{y}) p(\mathbf{y}|\mathbf{x}_t) p(\mathbf{x}_t|\mathbf{x}_{t-1}) \quad (7)$$

where the likelihood function $p(\mathbf{y}|\mathbf{x}_t) = p(\mathbf{y}|\mathbf{r}_t)$ has been defined in Equation 4. The transition density distribution $p(\mathbf{x}_t|\mathbf{x}_{t-1})$ is defined as:

$$p(\mathbf{x}_t|\mathbf{x}_{t-1}) = p(\mathbf{r}_t|\mathbf{r}_{t-1}) p(j_t) \quad (8)$$

with the ribbon transition $p(\mathbf{r}_t|\mathbf{r}_{t-1})$ as defined in Equation 2. We assume that the transition of the jump variable is independent from its past state, i.e. $p(j_t|j_{t-1}) = p(j_t)$. This way, $p(j_t)$ implicitly controls the length of the detected spines. We note $\kappa = p(j_t = 1)$ the probability to make a jump.

The particle filter aims at recursively approximating the posterior distribution $p(\mathbf{x}_{1:t}|\mathbf{y})$ using a finite set of N samples $\{\mathbf{x}_t^{(n)}\}_{n=1, \dots, N}$. The resulting empirical distribution is [43]:

$$P_N(d\mathbf{x}_{1:t}|\mathbf{y}) = \sum_{n=1}^N w_t^{(n)} \delta_{\mathbf{x}_{1:t}^{(n)}}(d\mathbf{x}_{1:t}) \quad (9)$$

where $\delta_{\mathbf{x}_{1:t}}^{(n)}(\cdot)$ is a Dirac mass centered on a hypothetical state realization $\mathbf{x}_{1:t}^{(n)}$ of the state $\mathbf{x}_{1:t}$, also called particle, $w_t^{(n)}$ is its weight, and $d\mathbf{x}_{1:t}$ is an event of infinitesimal support. The recursion estimation of the trajectory distribution can be carried out by three steps: 1) propagating the particle set from $t-1$ to t using an importance function; 2) updating the particle weights using in particular the likelihood, and resampling the particles if needed [43]; 3) checking if the particle filter has completed its task, i.e. if no further ribbon needs to be extracted. These three steps are detailed below.

1) *Generating the Particles*: The first step of a particle filter algorithm consists in propagating the particles from the previous particle set $\{\mathbf{x}_{t-1}^{(n)}\}_{n=1,\dots,N}$. Particles are generated using a function of importance $q(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \mathbf{y})$:

$$q(\mathbf{x}_t|\mathbf{x}_{1:t-1}, \mathbf{y}) = q(\mathbf{r}_t|\mathbf{r}_{1:t-1}, j_t, \mathbf{y}) q(j_t|\mathbf{x}_{t-1}, \mathbf{y}) \quad (10)$$

The ribbon proposal density is proportional to the initialization density (Eq. 6) if a jump is made. Otherwise, the proposal density is proportional to the ribbon transition (Eq. 2). Also, it depends on a trajectory term that gives little probability to echo detections:

$$q(\mathbf{r}_t|\mathbf{r}_{1:t-1}, j_t, \mathbf{y}) = \frac{1}{n_q} f(\mathbf{s}_t, \mathbf{s}_{1:t-1}) \times [j_t p(\mathbf{r}_t|\mathbf{y}) + (1 - j_t) p(\mathbf{r}_t|\mathbf{r}_{t-1})] \quad (11)$$

The trajectory constraint $f(\cdot)$ is set to 10^{-5} , if any point of the spine curve defined by \mathbf{s}_t is closer than a Manhattan distance of 5 with any spine point of the past ribbons $\mathbf{s}_{0:t-1-k}$, with k the length of the current ribbon, otherwise $f(\cdot)$ is set to 1. While the spines from current and past ribbons cannot intersect, the ribbons themselves can. This way, the algorithm can detect symmetries at different scales, which may for example happen with objects presenting local and global symmetries. There is no restriction on the current ribbons, since it is already done for the past state in the definition of $p(\mathbf{r}_t|\mathbf{r}_{t-1})$. Hence, self-crossing ribbons are unlikely, but remain possible in order to add a little flexibility in the detection process. Generating the particles can be done using a rejection sampling, i.e. generating a candidate $\mathbf{s}_t^{(n)}$ with the right-side of Equation 11, and accepting it with a probability of $f(\mathbf{s}_t^{(n)}, \mathbf{s}_{1:t-1}^{(n)})$. Finally, the normalizing term n_q can be approximated using an importance sampling method, but we found in practice that ignoring it resulted in a non-significant bias, while saving great computational time.

The jump proposal density $q(j_t|\mathbf{x}_{t-1}, \mathbf{y})$ uses the probabilities of the tail distributions (Eq. 3) computed at $t-1$ to determine if the ribbon \mathbf{r}_{t-1} is still relevant. Therefore, a jump is made with probability $\sum_{j=1}^J \tilde{\lambda}^j P(\mu^j > f^j(\mathbf{r}_{t-1}))$, where $\tilde{\lambda}^j = \lambda^j / \sum_{k=1}^J \lambda^k$.

2) *Computing the Particle Weights*: The joint likelihood (Eq. 4), the transition model (Eq. 8) and the proposal distribution (Eq. 10) are used to weigh the particles:

$$w_t^{(n)} \propto w_{t-1}^{(n)} \frac{p(\mathbf{y}|\mathbf{r}_t^{(n)}) p(\mathbf{x}_t^{(n)}|\mathbf{x}_{t-1}^{(n)})}{q(\mathbf{x}_t^{(n)}|\mathbf{x}_{1:t-1}, \mathbf{y})} \quad (12)$$

with $\sum_{n=1}^N w_t^{(n)} = 1$. Thus, a particle is assigned to a high weight when its likelihood is high, and when prior and

importance function are in agreement. This shows the importance of defining an importance function dependent on both the transition and the observation, and whose support includes the one of the filtering distribution. The particles may then be resampled when their effective number $[\sum_{n=1}^N (w_t^{(n)})^2]^{-1}$ becomes lower than a threshold, typically $0.75 \times N$ [43].

3) *Algorithm Completion*: A particle filter is automatically stopped when the number of jumps reaches a learned threshold. The number of jumps being related to the meaningfulness of an extracted ribbon, this means that the algorithm automatically stops when all relevant ribbons have been extracted from the image. Also, L several independent particle filters are used to estimate the trajectory distribution in order to add diversity to the trajectory, hence yielding $\{\mathbf{r}_t^l = (\mathbf{p}_t^l, \theta_t^l, l_t^l, \tilde{\alpha}_t^l, j_t^l)\}_{l=1,\dots,L; t=1,\dots,t_l}$, with t_l the last step performed by the l^{th} particle filter. Let $\tilde{\mathbf{r}}_{1:t_l}^l$ be the ribbon trajectory of the particle of maximum weight of the l^{th} particle filter, and $\tilde{w}_{t_l}^l$ its associated weight. The soft ribbon map is a real-valued image in $[0, 1]$, giving the probability of a pixel $z \in \Omega$ to be part of a ribbon:

$$O(z) = \frac{1}{L} \sum_{l=1}^L \tilde{w}_{t_l}^l \mathbb{1}_{\tilde{\mathbf{r}}_{1:t_l}^l}(z) \quad (13)$$

with l the index of a particle filter, and $\mathbb{1}_{\mathbf{r}}(z) = 1$ if z belongs to the ribbon \mathbf{r} , 0 otherwise. Contour and spines soft maps can be accordingly computed replacing \mathbf{r} by the adequate variables in Equation 13.

VI. EXPERIMENTS

This section presents some experimental results in symmetry, contour, and ribbon detections. First, we talk about implementation details, parameters tuning, and their influence.

A. Implementation

The proposed algorithm uses parameters of three different kinds. They originate from: 1) the geometric model, 2) the likelihoods, 3) the particle filter framework. For the geometric model, parameters were either optimized using a trial-and-error procedure on the LS-BSDS300 training dataset for the detection of local symmetries (Section VI-B) or manually set (although they also could be learned) for the detection of specific ribbons (Section VI-C). The trial-and-error procedure gave the best results for ($d = 9, \sigma_{\tilde{\alpha}} = 0.4, \sigma_{\theta} = 0.2, \sigma_l = 0.18$). Closely related to the geometric model since it controls the length of the spines, the prior probability of making a jump was estimated to $\kappa = 0.0005$. The minimal radius l^m was manually set to 3 and the maximal radius l^M to $\max(w, h)/2$, which ranges the detection thickness from thin tubular objects to objects as wide as the image. For the likelihoods, the weights $\{\lambda^j\}_j$ and the histogram computation internal parameters have also been optimized using a trial-and-error procedure, and were found optimal for ($\lambda^1 = 25, \lambda^2 = 10, \lambda^3 = 25, \lambda^4 = 13, \lambda^5 = 13$), for color histograms of $6 \times 6 \times 6$ bins and for oriented gradient histograms of 8 bins by direction, respectively. These parameters have been fixed for all the different experiments. As for the particle filters, $N_i = 10^7$ samples in the tail and initialization distributions

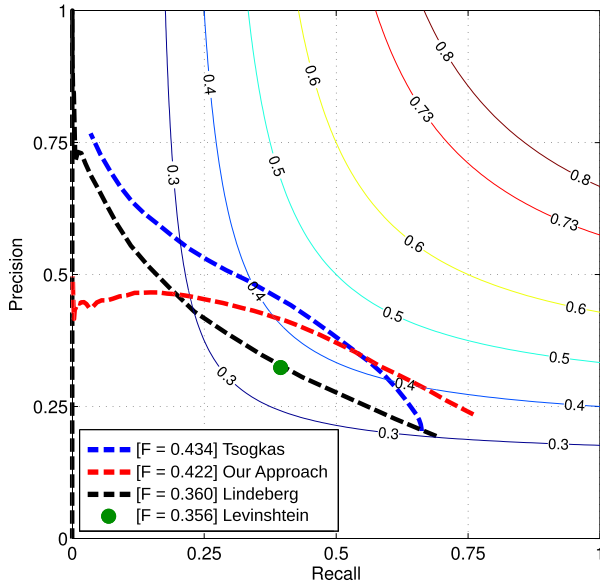


Fig. 5. Scores obtained by state-of-the-art methods and our proposed approach on the LS-BSDS300. The F-Measure curves correspond to the harmonic means of the recall and the precision scores. The final score is the optimal threshold computed among the 89 test images.

and 150 particle filters of 150 particles were needed to achieve the estimation of the posterior distribution. These parameters have been set to offer a good trade-off between estimation accuracy and computation time. Finally, a particle filter stops when $t \geq 100$ and its proportion of jump on the last 200 steps reaches 14%. These parameters have been estimated using a trial-and-error procedure.

The algorithm is well suited for parallel programming: the samples in the approximation of the tail and initialization distributions are independent and identically distributed (i.i.d.), the samples in a particle filter are also i.i.d. (before the normalization step), and the particle filters are mutually independent. Hence, the algorithm has the advantage of getting benefits from any hardware based on multi-threaded CPUs and/or GPU cards. Our implementation has been done in Cuda, and runs in about 22 seconds on a Nvidia GTX 670 GPU card. In contrast, the code of Tsogkas and Kokkinos [30] is a mixed C++ and Matlab implementation and takes about 270 seconds on a quad cores i7-920 (2.66 GHz) processor. Code and results are available at <http://www.iro.umontreal.ca/~mignotte/pfsd/>.

B. Local Symmetry Detection

Since our approach deals with local symmetry detection, we first compare our results on the LS-BSDS300 to the ones of [32], [33], and [30]. The LS-BSDS300 test dataset contains 89 of the 100 test images of the BSDS300 [30]. The LS-BSDS300 ground truth is extracted from an automatic skeletonization algorithm from the manual selections of symmetrical objects. For these objects, the skeleton medial axes are assumed to be good approximations of the local symmetry curves. The ground truth score is indicated by a plain red curve at $F = 0.73$. The results of Lindeberg's method [33] come from the implementation proposed in [17]. The F-Measure curves are illustrated in Fig. 5, and pairwise score comparisons are presented in Table I. We performed the experiment several

TABLE I
ONE-TO-ONE AND ONE-AGAINST-ALL COMPARISONS OF SYMMETRY DETECTION METHODS ON THE LS-BSDS300. SCORES ARE COMPARED BASED ON THE OPTIMAL THRESHOLD OBTAINED BY EACH IMAGE. EACH ROW POINTS OUT THE NUMBER OF IMAGES THAT OBTAIN BETTER SCORES THAN THE METHOD IN THE CORRESPONDING COLUMN. THE LAST COLUMN PRESENTS A ONE-AGAINST-ALL COMPARISON

	Levinshstein	Lindeberg	Tsogkas	Our approach	All
Levinshstein	—	28	11	14	4
Lindeberg	61	—	19	23	9
Tsogkas	78	70	—	56	47
Our approach	75	66	33	—	29

times and obtained a variance of 2.3×10^{-6} attributed to the stochastic nature of our algorithm. Our approach (FM: 0.422) compares favorably to Levinshstein *et al.* [33] (FM: 0.356) and Lindeberg [32] (FM: 0.36) algorithms, and obtains a slightly inferior F-Measure score compared to the one obtained by Tsogkas and Kokkinos [30] (FM: 0.434). One-against-all comparisons show that Tsogkas and Kokkinos method obtains the best performance on 47 of the 89 test images. Our approach obtains the highest scores for 29 images. Fig. 6 depicts the ground truth for several images and their associated results for the comparison methods and ours. For Lindeberg, Tsogkas and Kokkinos, and our method, thresholds were optimized over the whole dataset. The strength of the approach proposed by Tsogkas and Kokkinos is to result in well structured spine detection, even for the thresholded maps. In comparison, our method produces less misled spines originating from textures and contours. Fig. 7 presents several grayscale results of the extracted spines and their corresponding contours. The detected spines are the ones evaluated by the LS-BSDS300. We also display a degree of symmetry in an image by defining it as the ratio between the area of the extracted ribbons and the number of pixels in the image. We can especially see that images of human constructions, such as statues and buildings, have high degrees of symmetry.

As mentioned in [30], it is relevant to compare the algorithms [30], [32], [33] with ours since all these algorithms aim at detecting local symmetries. However, it is worth noticing that these algorithms produce different outputs: the detector proposed by Levinshstein *et al.* [33] approximates symmetric objects by fitted ellipses from which symmetric axes are retrieved, thus the output is binary, whereas in [30]–[32], the outputs are 1D, and in grayscale. For the proposed method, the output is also a probability, but on ribbons, not just spines. The two-scale parameters and the symmetry axes extracted by the Levinshstein *et al.* approach could be directly used as a preprocessing step of a segmentation algorithm. As for the other approaches, which include ours, a thresholding step may be needed before applying a segmentation algorithm. As it happens, this threshold may be the one maximizing the F-Measure score in Fig. 5. Our approach not only provides the spines but also the structural information of the object, namely its scale and its contours (Fig. 7). Finally, although the precision-recall curves obtained by Tsogkas and Kokkinos [30] and our approach are very



Fig. 6. Local symmetry ground truth (1st column on the left) and several thresholded detection results obtained by (2nd to the last column on the right): Levinstein *et al.* [33], Lindeberg [32], Tsogkas and Kokkinos [30], and our approach, respectively. Thresholds were optimized over the whole dataset.

different, the F-Measure scores are quite similar. Moreover, we can see from Table I that these methods perform well for different images. This can certainly be explained by the distinct natures of these two algorithms: one is a data-driven learning method, the other is a geometric-based model method. This could suggest that the former could be embedded as a likelihood into our Bayesian approach to achieve better results.

Failure modes of the proposed approach include incorrect and meaningless symmetry axis detections, e.g. between the raven and the vertical twig (last image of the third row in Fig. 7), or in the snow (in the middle of the second row in the same figure). This behavior can be explained by our definition of symmetry, which is purely local, and thus both the geometric model and the likelihood functions have not the adequate level of abstraction to discriminate relevant from insignificant symmetry axes. To overcome these limitations, we propose to integrate prior geometric information in order to detect specific ribbons. This is the subject of the following section.

C. Ribbon Detection

In this section, we address the problem of detecting symmetric objects. As opposed to state-of-the-art symmetry detection

methods [21], [26], [27], [30], [32], [33], our approach yields to a ribbon detection output.

1) *Filtering the Posterior Trajectory*: A first way to recover specific ribbons from an image is to filter the output trajectory $\{\mathbf{r}_t^l\}_{l=1,\dots,L; t=1,\dots,t_l}$ using some geometric criteria. This is illustrated in Fig. 8. The ribbons have been detected from the trajectories displayed in Fig. 8(b) using criteria such as minimum/maximum ribbon radiuses, and minimum/maximum spine lengths. In Fig. 8(d)–(f), radius and length have been measured for specific objects of the image, and their corresponding range parameters have been set to $\pm 10\%$ of their values. The analysis of the posterior trajectory distribution could also be exploited in an object recognition system based on symmetric-, or ribbon-object parts.

2) *Constraining the Geometric Model*: The proposed geometric model is designed in such way that each component can be intuitively tuned in order to constrain the detection. In Fig. 9, we study the effect of tuning the geometric model parameters on the detected spines. Fig. 9(b) shows the results obtained using the parameters optimized on the training set of the LS-BSDS300 (Section VI-B). This set of parameters detects curved and local symmetries. As expected, lowering the probability of jumps κ leads to longer

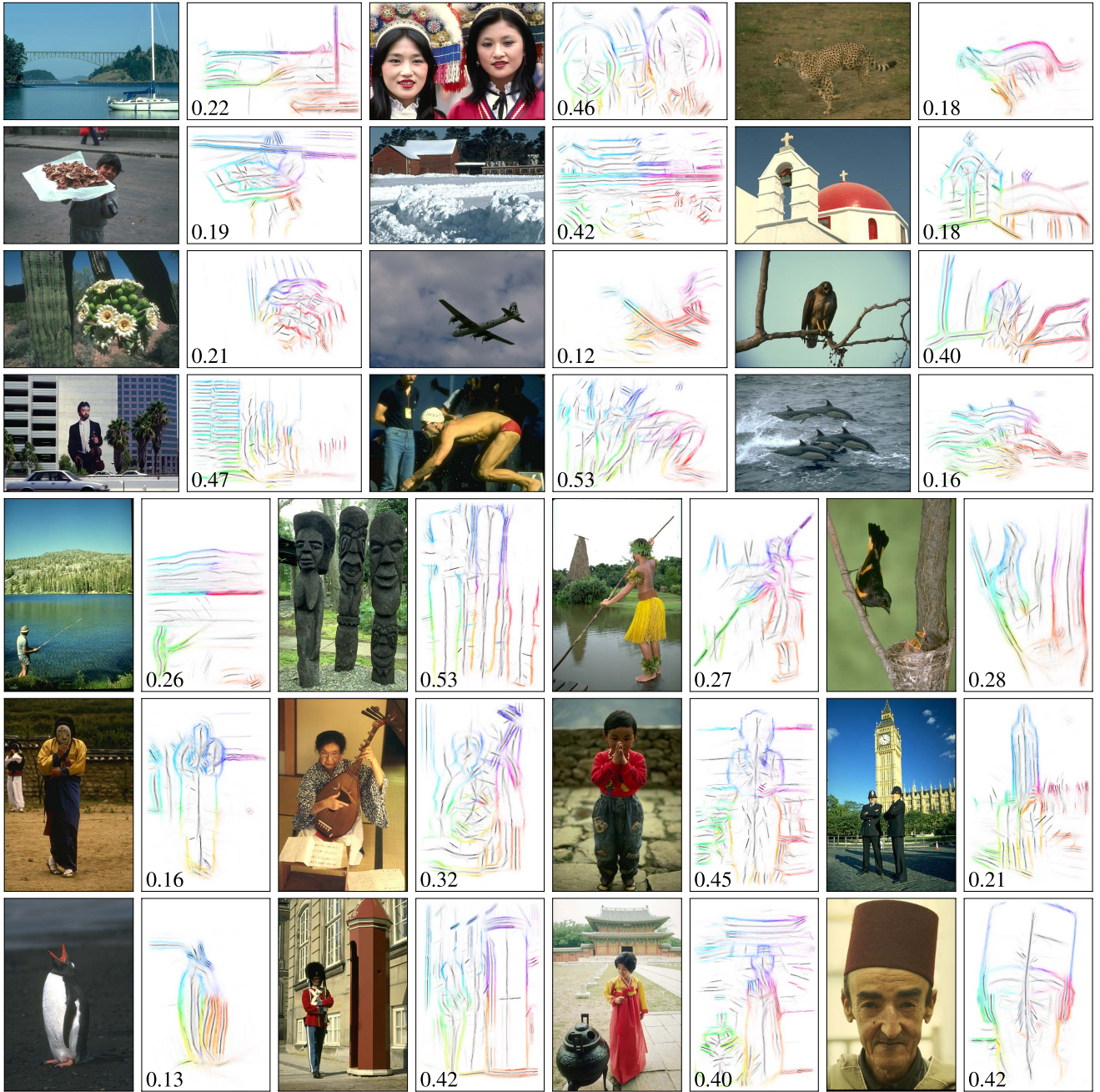


Fig. 7. Examples of local symmetry (black) and contour (color) detections obtained by the proposed algorithm on the BSDS300. Values correspond to the degrees of symmetry estimated on the images. Contour points associated with the same spine point share the same color.

spines (Fig. 9(d)). Straight and more global symmetries can be retrieved by tuning the spine curvature and the distance parameter d (Fig. 9(g)). By also tuning the radius parameter, one can obtain a cone or a cylinder geometric model (Fig. 9(h)).

The last experiment is an attempt to demonstrate the ability of the approach to only detect class-specific ribbons using local prior geometric information. This prior geometric information is directly embedded into the geometric model. Hence, only the objects satisfying this prior information are detected by the algorithm. This is illustrated in Fig. 10. The top row is especially challenging as the original image contains a lot of

symmetric objects with comparable structures. The pepperpot, the box, and the bottle have been detected on three independent runs of the algorithm, each one corresponding to specific constraints on the geometric model. This figure also illustrates the detection of ribbon objects from the BSDS300. Objects have been detected by tuning several parameters of the detection algorithm: the geometrical parameters ($l^m, l^M, \sigma_l, \sigma_\theta$), and the probability of jumps κ , the latter having an impact on the detected spine lengths. The other parameters have been set identically for all the images. Namely, σ_α equals 0.4, and the particle filter stops when $t \geq 20$ and its proportion of jump on the last 200 steps reaches 6%. Moreover, detections

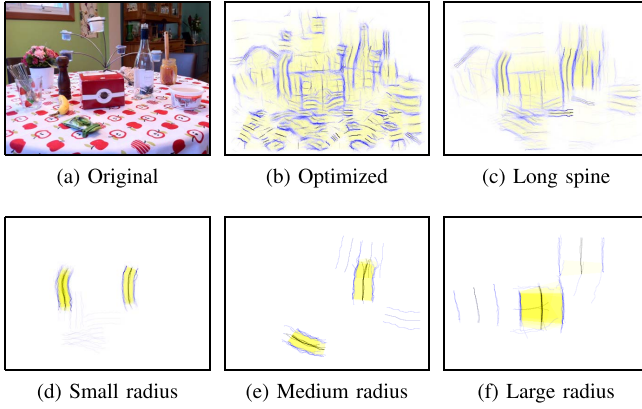


Fig. 8. Ribbon detection. Contours (blue), symmetries (black), and objects (yellow). The brightness of the color indicates the probability of detection. (b) Result using the parameters optimized on training data of the LS-BSDS300. (c) Trajectories with a spine length less than 36 pixels have been filtered from (b). (d-f) Trajectories with a spine length between 45 and 72 pixels and radiuses between (d) 10 and 15, (e) 15 and 25, and (f) 35 and 50 have been kept from (b).

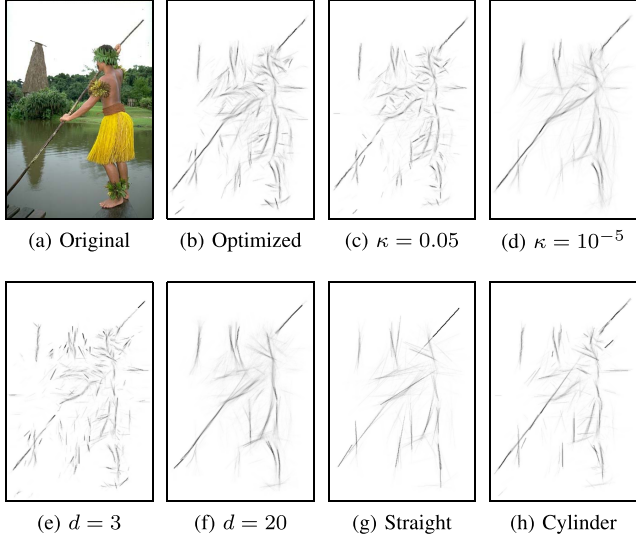


Fig. 9. Parameter influence. (b) Parameters have been optimized on the training set of the LS-BSDS300. (c)-(f) These results have been obtained by changing one parameter of (b). (g),(h) These results have been obtained by changing several parameters of (b): (g) ($d = 20, \sigma_\theta = 0$), and (h) ($\sigma_{\bar{a}} = 0.04, \sigma_\theta = 0.02, \sigma_l = 0.02$).

with a spine length of less than 45 pixels have been filtered. The other parameters remained unchanged from the set of optimized parameters. For example, the doll in the first image of this set has been detected by setting the radius interval $[l^m, l^M]$ at [25], [40], the standard deviation of the radius σ_l at 5, and the standard deviation of the orientation σ_θ at 0.005. Compared to filtering the posterior trajectory, constraining the geometric model to detect specific ribbons has the advantage of being computationally efficient: the detection is stopped earlier as it focuses only on the ribbons satisfying the geometric model. This experiment suggests that constraining the geometric model could be employed to detect class-specific ribbon objects. For practical purposes, these local geometric parameters could also be learned directly from a dataset of specific objects.

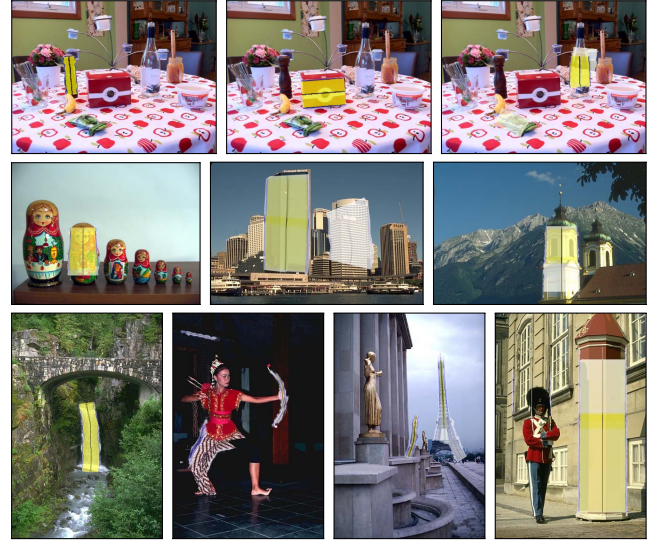


Fig. 10. Ribbon detection. Contours (blue), symmetries (black), and objects (yellow). The brightness of the color indicates the probability of detection. Ribbons are detected by tuning the geometric model parameters and the prior probability of jumps.

VII. CONCLUSION

We proposed in this article a ribbon detection algorithm using a particle filtering method. Ribbons have the particularity of exhibiting smooth local symmetries. Our contributions were threefold: the generative geometric model, which can represent a large quantity of objects; the feature extraction, which is designed for the local reflection symmetries in natural images; and the ribbon detection algorithm, which is an adaptation of the work proposed in [19] and [20]. Experiments conducted on the LS-BSDS300 [30] and on ribbon detection demonstrated the ability of the approach to extract low level features (local symmetries and contours) and symmetric, ribbon-like objects. These results could be used as an input of a segmentation or class-specific object detection algorithm.

The aim of this article is to detect local symmetries using local geometric models. In contrast to a global geometric model, the use of a local geometric model is suitable to represent a large variety of ribbon objects. However, a local geometric model is often poorly specific, which can be a limitation when one wants to recover complex ribbon shapes. It is also a limitation when one is only interested by the detection of global symmetries. In this case, state-of-the-art global symmetry detection algorithms proposed in [21], [26], and [27] are more naturally suited.

These limitations bring several perspectives to this work, in which the geometric model could be adapted to serve different aims. First, modeling more global structural information would enable to detect class-specific ribbons. This global prior information could be, for example, structure-based [23], [24], contour-based [44]–[46], or appearance-based [47], [48]. Second, adapting the likelihood in order to integrate spatial texture information, such as it has been done in [26], [27], and [21] using SIFT points, would be relevant for global symmetry detection and for 3D shape detection.

In this latter case, it would also require to model skewed symmetries as a subclass of Brooks ribbons [36], as proposed in [4].

ACKNOWLEDGMENTS

The authors would like to thank the anonymous reviewers for their helpful comments that greatly contributed to improve this paper.

REFERENCES

- [1] Y. Liu, H. Hel-Or, C. S. Kaplan, and L. Van Gool, "Computational symmetry in computer vision and computer graphics," *Found. Trends Comput. Graph. Vis.*, vol. 5, nos. 1–2, pp. 1–195, 2010.
- [2] G. Kootstra, A. Nederveen, and B. De Boer, "Paying attention to symmetry," in *Proc. Brit. Mach. Vis. Conf.*, 2008, pp. 1115–1125.
- [3] M. Brady and H. Asada, "Smoothed local symmetries and their implementation," *Int. J. Robot. Res.*, vol. 3, no. 3, pp. 36–61, 1984.
- [4] J. Ponce, "On characterizing ribbons and finding skewed symmetries," *Comput. Vis., Graph., Image Process.*, vol. 52, no. 3, pp. 328–340, 1990.
- [5] C. Florin, N. Paragios, and J. Williams, "Particle filters, a quasi-Monte Carlo solution for segmentation of coronaries," in *Proc. 8th Int. Conf. Med. Image Comput. Comput. Assist. Intervent.*, 2005, pp. 246–253.
- [6] C. Florin, N. Paragios, and J. Williams, "Globally optimal active contours, sequential Monte Carlo and on-line learning for vessel segmentation," in *Proc. 9th Eur. Conf. Comput. Vis.*, 2006, pp. 476–489.
- [7] D. Lesage, E. D. Angelini, I. Bloch, and G. Funka-Lea, "Medial-based Bayesian tracking for vascular segmentation: Application to coronary arteries in 3D CT angiography," in *Proc. 5th IEEE Int. Symp. Biomed. Imag.*, May 2008, pp. 268–271.
- [8] M. Schaap, R. Manniesing, I. Smal, T. Van Walsum, A. Van Der Lugt, and W. Niessen, "Bayesian tracking of tubular structures and its application to carotid arteries in CTA," in *Proc. 10th Int. Conf. Med. Image Comput. Comput. Assist. Intervent.*, 2007, pp. 562–570.
- [9] J.-H. Jang and K.-S. Hong, "Detection of curvilinear structures and reconstruction of their regions in gray-scale images," *Pattern Recognit.*, vol. 35, no. 4, pp. 807–824, 2002.
- [10] C. Steger, "An unbiased detector of curvilinear structures," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 2, pp. 113–125, Feb. 1998.
- [11] X. Hu and C. V. Tao, "A reliable and fast ribbon road detector using profile analysis and model-based verification," *Int. J. Remote Sens.*, vol. 26, no. 5, pp. 887–902, 2005.
- [12] P. Arbelaez, M. Maire, C. Fowlkes, and J. Malik, "Contour detection and hierarchical image segmentation," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 33, no. 5, pp. 898–916, May 2011.
- [13] M. Mignotte, "A non-stationary MRF model for image segmentation from a soft boundary map," *Pattern Anal. Appl.*, vol. 17, no. 1, pp. 129–139, 2012.
- [14] K. Siddiqi, A. Shokoufandeh, S. J. Dickinson, and S. W. Zucker, "Shock graphs and shape matching," *Int. J. Comput. Vis.*, vol. 35, no. 1, pp. 13–32, 1999.
- [15] M. Pelillo, K. Siddiqi, and S. W. Zucker, "Matching hierarchical structures using association graphs," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 21, no. 11, pp. 1105–1120, Nov. 1999.
- [16] M. Stark, M. Goesele, and B. Schiele, "A shape-based object class model for knowledge transfer," in *Proc. 12th IEEE Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 373–380.
- [17] I. Kokkinos, P. Maragos, and A. Yuille, "Bottom-up & top-down object detection using primal sketch features and graphical models," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 2, 2006, pp. 1893–1900.
- [18] I. Kokkinos and A. Yuille, "Inference and learning with hierarchical shape models," *Int. J. Comput. Vis.*, vol. 93, no. 2, pp. 201–225, 2011.
- [19] N. Widynski and M. Mignotte, "A particle filter framework for contour detection," in *Proc. 12th Eur. Conf. Comput. Vis.*, vol. LNCS-7572, 2012, pp. 780–794.
- [20] N. Widynski and M. Mignotte, "A multiscale particle filter framework for contour detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 36, no. 10, pp. 1922–1935, Oct. 2014.
- [21] S. Lee and Y. Liu, "Curved glide-reflection symmetry detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 34, no. 2, pp. 266–278, Feb. 2012.
- [22] S. Belongie, J. Malik, and J. Puzicha, "Shape matching and object recognition using shape contexts," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 24, no. 4, pp. 509–522, Apr. 2002.
- [23] N. H. Trinh and B. B. Kimia, "Learning prototypical shapes for object categories," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit. Workshop*, Jun. 2010, pp. 1–8.
- [24] N. H. Trinh and B. B. Kimia, "Skeleton search: Category-specific object recognition and segmentation using a skeletal shape model," *Int. J. Comput. Vis.*, vol. 94, no. 2, pp. 215–240, 2011.
- [25] S. Thrun and B. Wegbreit, "Shape from symmetry," in *Proc. 10th IEEE Int. Conf. Comput. Vis.*, vol. 2, Oct. 2005, pp. 1824–1831.
- [26] J. Liu and Y. Liu, "Curved reflection symmetry detection with self-validation," in *Proc. 10th Asian Conf. Comput. Vis.*, 2010, pp. 102–114.
- [27] G. Loy and J.-O. Eklundh, "Detecting symmetry and symmetric constellations of features," in *Proc. 9th Eur. Conf. Comput. Vis.*, 2006, pp. 508–521.
- [28] M. Park, S. Leey, P.-C. Cheny, S. Kashyap, A. A. Butty, and Y. Liu, "Performance evaluation of state-of-the-art discrete symmetry detection algorithms," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2008, pp. 1–8.
- [29] J. Podolak, P. Shilane, A. Golovinskiy, S. Rusinkiewicz, and T. Funkhouser, "A planar-reflective symmetry transform for 3D shapes," *ACM Trans. Graph.*, vol. 25, no. 3, pp. 549–559, 2006.
- [30] S. Tsogkas and I. Kokkinos, "Learning-based symmetry detection in natural images," in *Proc. 12th Eur. Conf. Comput. Vis.*, vol. LNCS-7572, 2012, pp. 41–54.
- [31] 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. 8th IEEE Int. Conf. Comput. Vis.*, vol. 2, 2001, pp. 416–423.
- [32] T. Lindeberg, "Edge detection and ridge detection with automatic scale selection," *Int. J. Comput. Vis.*, vol. 30, no. 2, pp. 117–156, 1998.
- [33] A. Levinstein, S. Dickinson, and C. Sminchisescu, "Multiscale symmetric part detection and grouping," in *Proc. 12th IEEE Int. Conf. Comput. Vis.*, Sep./Oct. 2009, pp. 2162–2169.
- [34] D. Terzopoulos, A. Witkin, and M. Kass, "Symmetry-seeking models and 3D object reconstruction," *Int. J. Comput. Vis.*, vol. 1, no. 3, pp. 211–221, 1988.
- [35] H. Blum, "A transformation for extracting new descriptors of shape," *Models Perception Speech Vis. Form*, vol. 19, no. 5, pp. 362–380, 1967.
- [36] R. A. Brooks, "Symbolic reasoning among 3D models and 2D images," *Artif. Intell.*, vol. 17, nos. 1–3, pp. 285–348, 1981.
- [37] A. Rosenfeld, "Axial representations of shape," *Comput. Vis., Graph., Image Process.*, vol. 33, no. 2, pp. 156–173, 1986.
- [38] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *Proc. IEEE Comput. Soc. Conf. Comput. Vis. Pattern Recognit.*, vol. 1, Jun. 2005, pp. 886–893.
- [39] M. Mignotte, "MDS-based multiresolution nonlinear dimensionality reduction model for color image segmentation," *IEEE Trans. Neural Netw.*, vol. 22, no. 3, pp. 447–460, Mar. 2011.
- [40] A. Desolneux, L. Moisan, and J.-M. Morel, "Edge detection by Helmholtz principle," *J. Math. Imag. Vis.*, vol. 14, no. 3, pp. 271–284, 2001.
- [41] N. Widynski and M. Mignotte, "A contrario edge detection with edgelets," in *Proc. IEEE Int. Conf. Signal Image Process. Appl.*, Nov. 2011, pp. 421–426.
- [42] P. Pérez, A. Blake, and M. Gangnet, "JetStream: Probabilistic contour extraction with particles," in *Proc. 8th IEEE Int. Conf. Comput. Vis.*, vol. 2, 2001, pp. 524–531.
- [43] A. Doucet, N. De Freitas, and N. Gordon, Eds., *Sequential Monte Carlo Methods in Practice*. New York, NY, USA: Springer-Verlag, 2001.
- [44] V. Ferrari, L. Fevrier, F. Jurie, and C. Schmid, "Groups of adjacent contour segments for object detection," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 30, no. 1, pp. 36–51, Jan. 2008.
- [45] C. Xu, J. Liu, and X. Tang, "2D shape matching by contour flexibility," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 31, no. 1, pp. 180–186, Jan. 2009.
- [46] T. Ma and L. J. Latecki, "From partial shape matching through local deformation to robust global shape similarity for object detection," in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit.*, Jun. 2011, pp. 1441–1448.
- [47] T. Deselaers, B. Alexe, and V. Ferrari, "Localizing objects while learning their appearance," in *Proc. 11th Eur. Conf. Comput. Vis.*, 2010, pp. 452–466.
- [48] L. Fei-Fei, R. Fergus, and P. Perona, "One-shot learning of object categories," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 28, no. 4, pp. 594–611, Apr. 2006.



with the Centre de Recherche du Centre Hospitalier de l'Université de Montréal, Montréal.

Nicolas Widynski received the Engineering degree from the École Pour l'Informatique et les Techniques Avancées, Paris, France, the master's degree in image processing from Pierre and Marie Curie University, Paris, in 2007, and the Ph.D. degree from Télécom ParisTech, Paris, and the Laboratoire d'Informatique de Paris 6, Paris, in 2010. He was a Post-Doctoral Fellow with the Département d'Informatique et de Recherche Opérationnelle, University of Montréal, Montréal, QC, Canada, from 2011 to 2013. He is currently a Post-Doctoral Fellow



Antoine Moevus received the dual bachelor's degree in electrical engineering and computer science from the École Polytechnique de Montréal, Montréal, QC, Canada, and the École Supérieure d'Électricité, Paris, France, in 2012, and the master's degree in computer vision from the Département d'Informatique et de Recherche Opérationnelle, University of Montréal, Montréal, QC, Canada, in 2014.



Max Mignotte received the D.E.A. degree in digital signal, image, and speech processing from the Grenoble Institute of Technology, Grenoble, France, in 1993, and the Ph.D. degree in electronics and computer engineering from the University of Bretagne Occidentale, Brest, France, and the Digital Signal Laboratory, French Naval Academy, Brest, in 1998.

He was an INRIA (French Institute for Research in Computer Science and Automation) Post-Doctoral Fellow with the Département d'Informatique et de Recherche Opérationnelle, University of Montréal, Montréal, QC, Canada, from 1998 to 1999, where he is currently an Associate Professor with the Computer Vision and Geometric Modeling Laboratory. He is also a member of the Laboratoire de Recherche en Imagerie et Orthopédie, Centre de Recherche du Centre Hospitalier de l'Université de Montréal (CHUM), Hôpital Notre Dame, Montréal, and a Researcher at CHUM. His current research interests include statistical methods, Bayesian inference, and hierarchical models for high-dimensional inverse problems, such as segmentation, parameters estimation, fusion, shape recognition, deconvolution, 3D reconstruction, and restoration problems.