TRACKING EDGES, CORNERS AND VERTICES IN AN IMAGE

PETER HALL
Centre for Mathematics and its Applications
Australian National University

PEIHUA QIU
School of Statistics
University of Minnesota

CHRISTIAN RAU
Department of Mathematics
Hong Kong Baptist University

23 January 2007

ABSTRACT. In a range of imaging problems, particularly those where the images are of man-made objects, edges join at points which comprise three or more distinct boundaries between textures. In such cases the set of edges in the plane forms what a mathematician would call a planar graph. Smooth edges in the graph meet one another at junctions, called "vertices", the "degrees" of which denote the respective numbers of edges that join there. Conventional image reconstruction methods do not always draw clear distinctions among different degrees of junction, however. In such cases the algorithm is, in a sense, too locally adaptive; it inserts junctions without checking more globally to determine whether another configuration might be more suitable. In this paper we suggest an alternative approach to edge reconstruction, which combines a junction classification step with an edge-tracking routine. The algorithm still makes its decisions locally, so that the method retains an adaptive character. However, the fact that it focuses specifically on estimating the degree of a junction means that it is relatively unlikely to insert multiple low-degree junctions when evidence in the data supports the existence of a single high-degree junction. Numerical and theoretical properties of the method are explored, and theoretical optimality is discussed. The technique is based on local least-squares, or local likelihood in the case of Gaussian data. This feature, and the fact that the algorithm takes a tracking approach which does not require analysis of the full spatial dataset, mean that it is relatively simple to implement.

Key words. Boundary estimation, edge detection, edge representation, image processing, image segmentation, kernel methods, local least squares, local likelihood, nonparametric curve estimation, planar graph, spatial statistics.

Short Title. Edge, corner and vertex tracking.

1. Introduction

Edges are critical features in many important applications of imaging. An edge often indicates the boundary of an object in the field of view. This extremity can provide crucial information about the object, sometimes more important than its "texture" (e.g. its grey shade or its colour). Not surprisingly, some of the earliest methods for image enhancement focused on delineating edges, for example by adding a multiple of the Laplacian back onto the image from which it was computed.

In many contemporary applications of image analysis a more accurate, and more explicit, estimate of the edge is required, often after part of the edge has been identified. One wishes to start an edge estimator at a given point that we believe is close to the edge, and follow the edge through places where it splits into two or more edges, tracking it up each branch, as well as following it in the reverse direction, away from the starting point. Edge-splitting is very commonly encountered in real images; a simple example arises in a non-orthogonal view of a cube, where a corner of the cube is represented in the planar image by three edges converging at a point. The angles the edges make to one another in the image depend on the position from which the cube is viewed. Making decisions about the number of edges into which a tracked edge splits, and successfully following each of these, is an intrinsically more difficult problem than estimating a smooth edge from spatial data, or tracking a smooth edge that does not have any junctions.

We suggest a tracking-based solution to the problem, founded on computing elementary local-likelihoods at discrete points along a piecewise-linear edge estimate. The points are uniquely determined by an algorithm for defining the piecewise-linear steps; the local likelihoods are produced by elementary local-linear approximations to the texture surface, under the temporary assumption of Gaussian noise; and the results of each likelihood analysis dictate whether the edge we are tracking is smooth in the neighbourhood of that point, or whether it splits into two or more edges. In the latter case an estimate of the number of edges is also given by elementary likelihood analysis. Since the likelihood is constructed under a Gaussian assumption then the method is equivalently based on local least-squares.

The edge-splitting models can be quite general, however. In particular, the edges might meet at arbitrary angles at a point, or their junction may be more narrowly defined. For example, a "T-intersection" model for three edges meeting

at a point, with two of the edges collinear, is readily included in the likelihood analysis, as are many others. The technique is equally appropriate for problems of image analysis and for nonparametric regression in the case of surfaces with complex fault-type discontinuities.

An image can be represented abstractly, and globally, as a function f from the plane $\mathbb{R}^2$ to d-variate space $\mathbb{R}^d$, where the value of f in the jth coordinate represents the grey-shade at the jth frequency or wavelength. For example, in a naive model for a colour image the number of components would be $d = 3$, representing the three primary colours. In remote-sensing applications the value of d can be relatively large. For clarity and simplicity we shall treat only the case $d = 1$, noting that methodology is similar in other cases.

For either $d = 1$ or $d \geq 2$, and despite the fact that the abstract model is defined in the continuum, we of course have only discrete data on the model, with the x variables restricted to a regular grid or, in the case of some statistical applications, to a stochastic point process in the plane. An example of the latter is undersea sampling, for example to determine water temperature. (Sea temperature gradients can be so steep that temperature can be interpreted as having discontinuities.) Here it is usually impossible to confine the spatial explanatory variables to a grid. However, in either case the algorithm we suggest is the same.

Taking $d = 1$ for definiteness, the function f is assumed to be a smooth (e.g. differentiable) function of $x \in \mathbb{R}^2$, except for the presence of fault-type discontinuities along a network of lines in the plane, and x is a point in that plane. The fault lines, which represent the edges, form a planar graph $\mathcal{G}$ in which the edges are, appropriately, represented as graph-theoretic edges, and the points at which edges join are vertices of the graph. The number of edges that join at a particular point is that vertex's graph-theoretic degree. This abstraction will form the basis of our model for edges in images. We shall develop an algorithm which uses discrete noisy data on textures (that is, on the values taken by f at points x not on $\mathcal{G}$) to estimate the locations and degrees of all vertices, and the locations of edges linking vertices.

Recent monograph-length accounts of computer vision include those of Jähne & Haussecker (2000), Forsythe & Ponce (2003), Hartley & Zisserman (2003) and Qiu (2005). Earlier monographs on computer vision and image analysis, paying particular attention to edge detection, include those by Gonzalez & Woods (1992),

Haralick & Shapiro (1992), Morel & Solimini (1995) and Parker (1997). Wavelet-based methods are discussed by Mallat (1999). Existing approaches to edge detection in computer vision usually begin with a pre-processing step, addressing issues such as illumination correction and noise suppression, before attempting to track the edge. By way of contrast, the approach taken in the present paper derives edge maps directly from the current data, in a single algorithm, reflecting statistical arguments that it is optimal to fit simultaneously all explanatory effects (illumination angle, noise, edges etc). The output of conventional edge detectors can be quite poor in the neighbourhood of corners and junctions, because those features of edges are not explicitly accommodated by standard edge models. Moreover, much of the literature on edge detection is rather informal, with relatively little rigorous theoretical backing. The present paper avoids these pitfalls through using edge models which incorporate corners and junctions, and by providing a theoretical account of performance.

Among recent contributions to smooth edge estimation based on spatial data we mention those of Sinha & Schunck (1992), Bhandarkar *et al.* (1994), Yi & Chelberg (1995), Qiu & Yandell (1997, 1998), Miyajima & Mukawa (1998), Qiu (1998), Bhandarkar & Zeng (1999) and Pedersini *et al.* (2000). O'Sullivan & Qian (1994), Chan *et al.* (1996), Hall & Rau (2000) and Hall *et al.* (2001) have suggested methods for fault-line tracking. Techniques proposed in the context of Markov random field models include those for fitting predetermined curved objects to boundaries, and for imputing line patterns; see e.g. Li (1995a, pp. 119, 177; 1995b; 2000), Achour *et al.* (2001), Kang & Kweon (2001), Kang & Roh (2001) and Park & Kang (2001).

2. Methodology

2.1. Features of a planar graph

A planar graph, $\mathcal{G}$, is a set of smooth planar curves, called *edges*, joined together at points called *vertices*. A vertex of degree $k \geq 1$ in $\mathcal{G}$ is a point from which radiate just k edges. We shall assume for simplicity that no two of these edges have the same orientation at the vertex, although this restriction can be removed; see subsection 2.4 for discussion. The main virtue of this "sharp vertex" condition is that it removes the possibility that multiple edges join vertices in a collinear way, i.e. that their tangents there have the same orientations. Our method would produce statistically consistent vertex estimators in this case, but the estimators

would not enjoy the convergence rates given in section 4.

The case $k = 2$, where just two edges meet at a vertex, is that of a corner. In this setting the "sharp vertex" assumption removes the degenerate case, where the angle between the two edges equals 0 or π, and so ensures that the corner has the character that it would in popular parlance. Of course, when $k = 2$ but the tangents meet at an angle that is very close to 0 or π, the corner may be relatively difficult to detect, and may be missed altogether unless the level of noise is relatively low. Analogously, in the context of hypothesis testing, if the null hypothesis is false but the alternative hypothesis is close to the null, the power of the test will be relatively low, and the alternative may not be detected.

A vertex of degree 3 or more, i.e. for which $k \geq 3$, will be called a knot. An edge in $\mathcal{G}$ will be assumed to be a segment of a planar curve of bounded curvature which joins two vertices. (The "sharp vertex" assumption removes degeneracy, by implying that the edge contains no vertices other than those at its ends.) The graph will be assumed to be connected, to contain only a finite number of vertices (and, consequently, only a finite number of edges), and to have finite total length. All graphs that we shall consider will be assumed to have these properties, which we shall denote by (G). The assumption of connectivity serves only to ensure that we can track $\mathcal{G}$ in its entirety by starting at a single point.

Note that each edge ends in a vertex. In particular, a vertex of degree 1 is the endpoint of an edge which, at that end, is linked to no other edge. Figure 1 shows a graph, $\mathcal{G}$, having vertices of differing degrees.

Place Figure 1 here, please

2.2. Global model for surfaces

Given a graph $\mathcal{G}$ satisfying (G), let f be a function from $\mathbb{R}^2$ to $\mathbb{R}$ such that (a) $\mathcal{G}$ is contained in an open subset, $\mathcal{S}$, of the support of f, (b) all first derivatives of f are uniformly bounded on $\mathcal{S} \setminus \mathcal{G}$, and (c) for each $\eta > 0$ the height of the jump discontinuity at $x \in \mathcal{G}$ is bounded away from zero uniformly in points x that are distant at least η from all vertices of degree 1 in $\mathcal{G}$. (If x is a vertex of degree $k \geq 2$ then this assumption is taken to mean that the k limits of $f(v)$, as $v \to x$ through points in any one of the k segments of $\mathcal{S} \setminus \mathcal{G}$ that meet at the vertex x, are all distinct.)

We shall denote these properties by (F). Given a function f satisfying (F), our

global model for the surface is defined by $y = f(x)$, $x \in \mathcal{S}$. The value of $f(x)$ might represent the colour, or grey shade, of an image at x. In this case $\mathcal{G}$ would denote the set of sharp boundaries between different colours in the image. Figure 2 depicts, for a graph $\mathcal{G}$ simpler than that in Figure 1, a function f satisfying (F) for this $\mathcal{G}$.

Place Figure 2 here, please

2.3. Local model for surfaces

Let Θ denote the set of all bivariate unit vectors. Our local (linear) model, M_k, for a surface will resemble the k wedge-shaped slices of a circular cake, the wedges meeting at a point $x_0 \in \mathbb{R}^2$. The height of the local surface model, above the plane $\mathbb{R}^2$, will be taken to have constant value c_i on the ith wedge. The model will be used to approximate a more elaborate surface, such as that described in subsection 2.2, where surface height has discontinuities along the edges of a graph $\mathcal{G}$ of which x_0 is a vertex of degree k, for $k \geq 2$.

Specifically, let $M_k = M_k(x_0, \theta_1, \ldots, \theta_k, c_1, \ldots, c_k)$, for $k \geq 2$, be the surface defined by $y = f_k(x)$, where $x \in \mathbb{R}^2$ and $f_k = f_k(\,\cdot \,|\, x_0, \theta_1, \ldots, \theta_k, c_1, \ldots, c_k)$ is the function that equals c_i on the infinite wedge that has its vertex at x_0 and its two straight sides in the directions of θ_i and θ_{i+1}. Each $\theta_i \in \Theta$, and we take $\theta_{k+1} = \theta_1$. It is assumed that the θ_i's are arranged such that passing from θ_i to θ_{i+1} involves rotating counterclockwise about the origin, without crossing any other unit vectors θ_j. We define f_k arbitrarily at x_0 and on the infinite rays that point in directions $\theta_1, \ldots, \theta_k$.

2.4. Local surface estimators

In this subsection we construct local approximations to the function f, using a least-squares method with kernel weights. The approximations will be used in subsection 2.5 to guide an algorithm for tracking, from data, the graph $\mathcal{G}$.

Noisy observations Y_i are made of f at points $X_i \in \mathbb{R}^2$. Specifically, we record the values of $Y_i = f(X_i) + e_i$, $1 \leq i \leq n$, where f satisfies (F) and corresponds to a graph $\mathcal{G}$ that satisfies (G), and the errors e_i are independent and identically distributed random variables, independent also of $X_1, \ldots X_n$, with zero mean and finite variance. (The value of n may be infinite, if the process of points X_i covers the entire plane.) Using the data (X_i, Y_i), $1 \leq i \leq n$, we wish to construct an estimator of f which respects the vertices of $\mathcal{G}$. In particular, we wish to estimate the number of vertices of $\mathcal{G}$, and their respective locations and degrees, and to construct an

estimator $\widehat{\mathcal{G}}$ of $\mathcal{G}$ whose vertices have the estimated properties. Our first step is to construct a sequence of local estimators of f.

To this end, let K denote a kernel function, which we take to be a compactly supported, bivariate probability density, and let h be a bandwidth. Let f_k denote the function defined in subsection 2.3, and put $K_i(x_0) = K\{(x_0 - X_i)/h\}$ and

$$S_k(x_0 \,|\, \theta_1, \ldots, \theta_k, c_1, \ldots, c_k) = \sum_{i=1}^{n} \left\{Y_i - f_k(X_i \,|\, x_0, \theta_1, \ldots, \theta_k, c_1, \ldots, c_k)\right\}^2 K_i(x_0)\,. \tag{1}$$

Then $S_k(x_0 \,|\, \theta_1, \ldots, \theta_k, c_1, \ldots, c_k)$ is proportional to a negative log-likelihood in the case where the true function is representable by the local model,

$$f_k(\cdot \,|\, x_0, \theta_1, \ldots, \theta_k, c_1, \ldots, c_k)\,,$$

in the vicinity of x_0, and the data (X_i, Y_i) are generated in the form $Y_i = f_k(X_i) + e_i$, with the errors e_i being Gaussian with zero mean and fixed variance. Of course, the methodology suggested by these very specialised assumptions is going to be useful in much more general settings, as we shall show in sections 3 and 4.

Returning to the one-dimensional case we choose $\hat\theta_1, \ldots, \hat\theta_k, \hat c_1, \ldots, \hat c_k$ to minimise the sum of squares at (1), and write simply $S_k(x_0)$ for the corresponding value of the left-hand side. The surface defined by $y = f(x)$ is estimated locally, near x_0, as the surface with equation $y = \hat f_k(x \,|\, x_0, \hat\theta_1, \ldots, \hat\theta_k, \hat c_1, \ldots, \hat c_k)$. If the kernel K is smooth and unimodal, with a uniquely defined mode, then the parameter estimators are uniquely defined with probability 1.

We take f_1 to be the function corresponding to a local linear model for the surface at a point x_0 that is on an edge of $\mathcal{G}$ but not a vertex. Thus,

$$f_1(x) = f_1(x \,|\, x_0, \theta, c_+, c_-) = \begin{cases} c_+ & \text{if } x \in \mathcal{H}_+(x_0, \theta) \\ c_- & \text{if } x \in \mathcal{H}_-(x_0, \theta)\,, \end{cases} \tag{2}$$

where $\mathcal{H}_\pm(x_0, \theta)$ are the half-planes on opposite sides of the infinite line that passes through x_0 and is parallel to θ. The function f_0, which is simply constant throughout $\mathbb{R}^2$, corresponds to our zeroth surface model, which is flat everywhere. The local surface model M_k, represented by the equation $y = f_k(x)$, will be said to be of degree k. Figure 3 shows a graph of the function f_1.

Place Figure 3 here, please

Our local models are fitted without any constraints on the angles at which edges approach vertices. We can of course impose such conditions, for example if we have prior information about edge configurations at vertices. In particular, we might suppose that at some vertices of degree 3 the edges are arranged so that two of them are collinear. Also, some vertices of degree 4 might involve two pairs of collinear edges. Such constraints are readily incorporated into the local edge model. They reduce, rather than add to, the computational (and theoretical) burden.

Note that

$$T_k(x_0) \equiv S_k(x_0) - S_{k+1}(x_0) \geq 0\,, \tag{3}$$

where the inequality follows from the fact that f_k is a constrained form of f_{k+1}. Observe too that $T_0(x)$ may equivalently be defined by the formulae

$$T_0(x) = \sup_\theta t(x,\theta)\,, \quad t(x,\theta) = \frac{N_+(x,\theta)\,N_-(x,\theta)}{N(x)}\,\{\bar{Y}_+(x,\theta) - \bar{Y}_-(x,\theta)\}^2\,, \tag{4}$$

and, for respective choices of the plus and minus signs,

$$N_\pm(x,\theta) = \sum_{i \in \mathcal{I}_\pm(x,\theta)} K_i(x)\,, \quad \bar{Y}_\pm(x,\theta) = N_\pm(x,\theta)^{-1} \sum_{i \in \mathcal{I}_\pm(x,\theta)} Y_i\,K_i(x)\,,$$

$\mathcal{I}_\pm = \{i : \pm\theta_\perp \cdot (X_i - x) \geq 0\}$, $\theta_\perp$ is the unit vector perpendicular to θ with its direction defined by rotating θ anticlockwise, $N(x) = N_+(x,\theta) + N_-(x,\theta) = \sum_i K_i(x)$ and $\bar{Y}(x) = N(x)^{-1} \sum_i Y_i\,K_i(x)$. See, for example, Hall, Peng and Rau (2001).

If a local model of degree k is approximately correct in the neighbourhood of x_0 then the value of $S_k(x_0)$, and hence of $T_k(x_0)$, will tend to be relatively small. The converse is also true: when a local model of degree k or less is not a good approximation to the response surface near x_0, then $T_k(x_0)$ will tend to be large. In particular, we can deduce that x_0 is close to $\mathcal{G}$ if $T_0(x_0)$ is large. This property will form the basis for tracking, and hence estimating, edges of the graph $\mathcal{G}$. We can recognise that we are close to a vertex of degree k when $T_{k-1}(x_0)$ is large but $T_k(x_0)$ is small.

2.5. Algorithm

We shall give the algorithm in a form that is appropriate for our theory in section 4. In practice, better performance is often obtained if some of the steps are slightly altered from the definitions given here. Optimal adjustments depend, for example, on the variance of the error distribution.

Assume we are given an integer $k_0 \geq 1$ such that each vertex of $\mathcal{G}$ is of degree not exceeding k_0. Suppose too that we have determined, generally from prior experience with data of the type we are analysing, two thresholds t_1 and t_2, satisfying $0 < t_1, t_2 < \infty$, which we shall use to determine whether we are in the vicinity of a vertex. We shall use the thresholds to make decisions about vertices. Take K to be a symmetric, unimodal, continuous, bivariate probability density with support equal to the unit disc, and let $\delta \in (0,1)$ be fixed; δh will equal the length of steps in the algorithm. In particular, choosing δ smaller generally gives a smoother trajectory, looking less like a polygonal path.

The five-part algorithm below produces an estimator $\widehat{\mathcal{G}}$ of the graph $\mathcal{G}$.

(1) *First step along $\mathcal{G}$.* We start at a point $\hat{x}_0$ that has been located close to an edge of $\mathcal{G}$. (Subsection 2.6 discusses empirical procedures for determining $\hat{x}_0$. Step (2) below includes a part which checks whether $\hat{x}_0$ is a vertex.) Recalling the definition of T_k at (2.3), we may write, in the case $j = 0$,

$$T_0(\hat{x}_j) = \sum_{i=1}^{n} (Y_i - \hat{c})^2 \, K_i(\hat{x}_j) - \sum_{i=1}^{n} \left\{ Y_i - f_1\big(X_i \,\big|\, \hat{x}_j, \hat{\theta}, \hat{c}_+, \hat{c}_-\big) \right\}^2 K_i(\hat{x}_j) \,. \tag{5}$$

The first series on the right-hand side denotes $S_0(\hat{x}_j)$, and the second, $S_1(\hat{x}_j)$. In the second series on the right-hand side of (5), $(\hat{\theta}, \hat{c}_+, \hat{c}_-)$ denotes the quantity that minimises $\sum_i \{Y_i - f_1(X_i \,|\, \hat{x}_j, \theta, c_+, c_-)\}^2 \, K_i(\hat{x}_j)$ with respect to (θ, c_+, c_-).

The next point in our excursion along the edges is $\hat{x}_1$, defined to be distant δh from $\hat{x}_0$ in the direction of $\hat{\theta}$. (We would also track $\mathcal{G}$ in the opposite direction, but of course we need only describe the tracking procedure in one direction.)

(2) *General step along an edge of $\mathcal{G}$.* Assume that in the previous step we tracked an edge of $\mathcal{G}$ to a point $\hat{x}_j$, where $j \geq 1$. We determine that $\hat{x}_j$ is in the vicinity of a vertex of degree 1 if $T_0(\hat{x}_j) \leq t_1$. In this case we terminate the current edge at $\hat{x}_j$, taking that point to be our estimator of the first-degree vertex at which the edge ends; and we join $\hat{x}_j$ to the previous point, $\hat{x}_{j-1}$, in the edge trace. On the other hand, if $\max_{2 \leq k \leq k_0} T_k(\hat{x}_j) > t_2$ then we conclude we are in the vicinity of a vertex of degree exceeding 1, and pass to step (3) of the algorithm.

If $T_0(\hat{x}_j) > t_1$ and $\max_{2 \leq k \leq k_0} T_k(\hat{x}_j) \leq t_2$ then we join $\hat{x}_j$ to the previous point $\hat{x}_{j-1}$, we compute $\hat{\theta}$ such that (5) holds, and we take $\hat{x}_{j+1}$ to be the point on the infinite line, containing the vector $\hat{x}_j + \hat{\theta}$, which is δh from $\hat{x}_j$ and in the direction of travel. Replacing $\hat{x}_j$ by $\hat{x}_{j+1}$, we repeat the current step of the algorithm.

(3) *Locating a vertex of degree* $k \geq 2$. If, at the previous step, we computed a point $\hat{x}_j$ for which $\max_{2 \leq k \leq k_0} T_k(\hat{x}_j) > t_2$, we go back one step to the previous point, $\hat{x}_{j-1}$. This is potentially the last point we estimate to be interior to the current edge; the next point estimator is likely to be an approximation to the vertex at which the edge ends.

As noted in step (2), the point $\hat{x}_{j-1}$ is associated with a vector $\hat{\theta}$, defined by (5) with j there replaced by $j-1$. Let $\mathcal{R}(\hat{x}_{j-1}, \hat{\theta})$ denote the ray that starts at $\hat{x}_{j-1}$ and points in the direction of $\pm\theta$, the sign being chosen to preserve the previous direction of travel. Write $\mathcal{L}(\hat{x}_{j-1}, \hat{\theta})$ for the line segment of length h along $\mathcal{R}(\hat{x}_{j-1}, \hat{\theta})$ that has one of its ends at the point on the ray that is distant δh from $\hat{x}_{j-1}$. Let $\hat{k} \in [2, k_0]$ denote the largest k for which $T_{k-1}(x) \geq t_2$ for all $x \in \mathcal{L}(\hat{x}_{j-1}, \hat{\theta})$. (If no such $\hat{k}$ exists then we continue to the next step in our traverse of the current edge.) Let $\hat{z}$ be the value of $x \in \mathcal{L}(\hat{x}_{j-1}, \hat{\theta})$ that maximises $T_{\hat{k}}(x)$. Then $\hat{z}$ is our estimator of the location of the vertex of degree $\hat{k}$ at which the line segment we have recently been tracking ends. We join $\hat{x}_{j-1}$ to $\hat{z}$.

(4) *Moving away from the vertex estimator at* $\hat{z}$. Fit a local surface model of degree $\hat{k}$ at $\hat{z}$, constructed so that one of its rays $\hat{\theta}_1, \ldots, \hat{\theta}_{\hat{k}}$ is in the direction of $\pm\hat{\theta}$. Move distance $(1+\delta)h$ along each of the other $\hat{k}-1$ rays, and place there a point that corresponds to $\hat{x}_0$, introduced in step (1). Join each of the $\hat{k}-1$ versions of $\hat{x}_0$ to $\hat{z}$. Now start tracking each of the $\hat{k}-1$ edges, arguing as in step (1) and moving in the direction away from $\hat{z}$.

(5) *Linking current edge estimators to vertices.* If, while tracking an edge, we compute a point $\hat{x}_j$ which is within distance $(1+\delta)h$ of a point previously estimated to be on $\mathcal{G}$, we join $\hat{x}_j$ directly to that point and assume the currently-tracked edge has ended. This step is the key to terminating the algorithm.

More generally, we link adjacent points $\hat{x}_j$ that have been computed as approximations to points on edges (see step (2)), and we link vertex estimators to the edge-point estimators immediately preceding them (see step (3)). The resulting piecewise-linear graph, $\widehat{\mathcal{G}}$, is our estimator of $\mathcal{G}$.

2.6. Starting point for algorithm

There is a variety of ways of locating a suitable starting point. They range from using a conventional edge detection algorithm (see section 1 for references) to applying a statistical change-point algorithm (see e.g. Müller, 1992; Eubank & Speckman,

1994; Müller & Song, 1994) employing a line transect traversing $\mathcal{S}$. Under conditions given in our account of theory in section 4, the latter approach produces an estimator $\hat{x}_0$ of a point x_0 at which the transect cuts $\mathcal{G}$, which with probability 1 satisfies $\hat{x}_0 - x_0 = O(h^2)$. A theoretical derivation of this property is similar to that given in Remark 3.4 of Hall & Rau (2000).

3. Numerical examples

3.1. Simulation study

Here we treat the model $Y_i = f(X_i) + e_i$, $i = 1, \ldots, 10000$, where

$$f(x) = f\bigl(x^{(1)}, x^{(2)}\bigr) = \begin{cases} -1 & \text{for } x^{(2)} < 0.25 \text{ and } x^{(1)} \in \bigl[x^{(2)}, 1 - x^{(2)}\bigr) \\ & \text{or } x^{(2)} > 0.75 \text{ and } x^{(1)} \in \bigl[1 - x^{(2)}, x^{(2)}\bigr)\,, \\ 1 & \text{for } x^{(1)} < 0.25 \text{ and } x^{(2)} \in \bigl[x^{(1)}, 1 - x^{(1)}\bigr) \\ & \text{or } x^{(1)} > 0.75 \text{ and } x^{(2)} \in \bigl[1 - x^{(1)}, x^{(1)}\bigr)\,, \\ 0 & \text{otherwise}\,. \end{cases}$$

The design points $\{X_i\}$ were independent and Uniformly distributed on the unit square $[0,1] \times [0,1]$, and the errors e_i were independent and Normally distributed with mean zero and standard deviation $\sigma = 0.25$. In the regression surface defined by f, the edge graph

$$\mathcal{G} = \bigl\{\|x - (0.5, 0.5)\|_\infty = 0.25\bigr\} \cup \Bigl\{\|x - (0.5, 0.5)\|_\infty > 0.25\,, x^{(2)} = x^{(1)}\Bigr\} \\ \cup \Bigl\{\|x - (0.5, 0.5)\|_\infty > 0.25\,, x^{(2)} = 1 - x^{(1)}\Bigr\}$$

has four knots of order three. Figure 4(a) shows a typical realisation of noisy regression surface data, where increasing brightness corresponds to a larger surface value. For the kernel K, we used the function $K(x^{(1)}, x^{(2)}) = (3/\pi)\,\{1 - (x^{(1)})^2 - (x^{(2)})^2\}^2$ for $\|x\| \le 1$ and vanishing elsewhere.

Place Figure 4 here, please

Bandwidths were selected in the interval $[0.01, 0.08]$, using the automated method discussed in subsection 3.2. The thresholds t_1 and t_2 would typically depend on the bandwidth h, and could hence vary as well. In the present example, however, simply choosing the thresholds as constants yielded good results, as illustrated in Figure 4(b). We took $\delta = 0.3$ and $t_2 = 0.03$, and chose t_1 very small; almost identical results were obtained for values of t_1 within a wide range. Figure 4(c) shows the aggregated result over 20 realisations, which is essentially a histogram

over 400×400 bins with vertices coinciding with those of a regular grid on the unit square. For easier visual interpretation, we set all non-zero values of histogram bins equal to a single value, represented by black dots in Figure 4(c).

We performed cyclic location re-fitting, which meant that we shifted the current estimate at the edge by a small amount along the normal to the estimated tangent direction. This idea was used by Hall & Rau (2000). Tracking of an edge was terminated if the estimate came within 0.08 of the boundary of the unit square. The initial point x_0 was located by searching in the left half of the unit square with the constraint $x^{(2)} = 0.5$.

3.2. Application to real image

In the computer vision literature, the 'Cameraman' image has been used extensively as a benchmark. We employed a greyscale, 256 $\times$ 256 pixel version of this image. The image domain was the unit square $[0,1] \times [0,1]$, and we restricted ourselves to estimating the edge graph within the sub-window $\Pi = [0.25, 0.75] \times [0.37, 0.87]$. Figure 5 depicts the original image restricted to Π.

Place Figure 5 here, please

Panels (a) and (b) of Figure 3.3 depict the two estimates of the boundary obtained by applying the Sobel and the Laplacian of Gaussian (LoG) methods, respectively. See Qiu (2005, section 6.2) for discussion of LoG. For both panels (a) and (b), thresholds were chosen to give the best visual impression. The parameters used in Figure 6(a) and (b) were selected for the original image, where 256 potential grey levels were coded as the integers $0, 2, \ldots, 255$. Pixel intensities were divided by 256, so as to yield a regression function with values in $[0, 1)$.

Place Figure 6 here, please

As was to be expected from the visual impression, a simulation using a single starting point did not suffice to estimate more than a part of the edge graph. To maximise coverage, we therefore started the algorithm at several points determined as change points in a line search, resulting in starting points $\hat{x}_0 = \hat{x}_{0,r} = \big(\hat{x}^{(1)}_{0,r}, \hat{x}^{(2)}_{0,r}\big)$. Here $\hat{x}^{(2)}_{0,r} = 0.38 + 0.015\, r$, and $\hat{x}^{(1)}_{0,r}$ was estimated within the interval (a_r, b_r) where $a_{3m} = a_0 = 0$, $a_{3m+1} = a_1 = 1/3$, $a_{3m+2} = a_2 = 2/3$ $(m \geq 1)$, and $b_r = a_r + 1/3$, $r = 0, \ldots, 29$. If the added edge graph component was too short (which we took as less than five points) or there was no edge detected at all, no points were added to the graph estimate. Tracking was stopped as soon as the estimate reached the

boundary of Π. Since data outside Π were retained, no boundary effects needed to be taken into account.

We took the thresholds t_1 and t_2 to vary in proportion to $h^{5/2}$, with h denoting the bandwidth. We estimated h locally, as follows. For a given bandwidth and at a given point, we estimated derivatives as regression coefficients in a piecewise-linear approximation on each of four square-shaped quarter-neighbourhoods of that point. These neighbourhoods are defined as in Qiu (2004) in the context of surface estimation, but were rotated by the estimated slope (called $\hat{\theta}$ at (5)) from the previous step. The Euclidean norm of the gradient in the two subsquares with the smallest residual sum of squares from the previous surface fitting step, was then minimised as a function of h. This gave bandwidths in the interval $[0.01, 0.03]$. To lessen the variation, bandwidths were calculated as moving averages, with the currently computed bandwidth and the two previous ones being given weights 0.6, 0.3 and 0.1, respectively. We took $\delta = 0.3$, and did not stop tracking until the algorithm got as close as 0.006 to a previously tracked point. The rule for taking a point to be a corner was that in subsection 3.1.

Figure 6(c) shows the obtained estimate as the raw points, which numbered 734 in total, and panel (d) shows the estimate obtained by linearly connecting the points in (c), alongside the detected corners and knots. The salient features of the images appear more clearly outlined than for the Sobel and LoG methods, with detected edges being both thin and well connected. Also, the front part of the camera, which has a very diffuse mixture of grey levels, is rendered surprisingly well, except for the fact that the top lens is missing. Performance on other real-world images of similar overall complexity also yielded good results, with generally only few parameters besides t_1, t_2 requiring adjustment.

An important issue is to assess the quality of estimation. This may be done visually, as is commonly done in practice, but this is well known to be in discordance with performance in various L_p metrics. In principle, a suitable metric should take the number of false positives and negatives of knots and corners into account; if there is a simple procedure for this then we have not found it. If one assumed a ground truth of edges in this or another image, we think that one of the distances suggested in Qiu (2002; 2005, section 4.4) would be suitable, with the caveats mentioned there.

4. Theory

4.1. Main results

We suppose that the response surface is given by the equation $y = f(x)$, where f is a function from $\mathbb{R}^2$ to $\mathbb{R}$. Data pairs (X_i, Y_i) are observed, generated by the model $Y_i = f(X_i) + e_i$, where the X_i's are points in the plane $\mathbb{R}^2$. Conditional on the set $\mathcal{X}$ of all X_i's, the errors e_i are assumed to be independent and identically distributed random variables with zero mean and a distribution which does not depend on $\mathcal{X}$ and has finite moment generating function in a neighbourhood of the origin. Further regularity conditions, (a)–(g), are given in Appendix 1. In particular, (a) implies that the points X_i are distributed with density approximately equal to ν per unit area of $\mathbb{R}^2$.

Under these assumptions we generate a sequence of datasets $\mathcal{Z} = \{(X_i, Y_i), -\infty < i < \infty\}$, indexed by the positive real parameter ν which, for simplicity, we take to be integer valued and which will be allowed to diverge to infinity. As ν increases, the points X_i become increasingly dense in $\mathbb{R}^2$. The assumptions require us to interpret $\mathcal{G}$ as a particular set of curves in the Cartesian plane, and so confer on $\mathcal{G}$ a metric rather than a merely topological character.

Recall that the estimator $\widehat{\mathcal{G}}$ of $\mathcal{G}$ was defined in subsection 2.5 by a five-part tracking algorithm. Suppose $\mathcal{G}$ has exactly m_k vertices of degree k for $1 \le k \le k_0$. Part (i) of the theorem below states that, with probability converging to 1 as the density of design points increases, the tracking algorithm correctly identifies the numbers of vertices of each degree, and in particular does not add any extra vertices. Part (ii) declares that vertices of degrees 2 or more are estimated to within $O(\epsilon)$, where $\epsilon = (\nu h)^{-1} \log \nu + h^2$, of their true locations. This degree of accuracy is not necessarily achieved for vertices of degree 1, although part (iii) shows that vertices of degree 1 are estimated consistently. Part (iv) states that edges of $\mathcal{G}$ are estimated with accuracy $O(\epsilon)$, and part (v) shows that the tangents of edges of $\mathcal{G}$ are estimated with accuracy $O(\epsilon/h)$.

Let D_0, D_1 and D_2 denote, respectively, the distance of an identified vertex from the nearest actual vertex of $\mathcal{G}$, the supremum of D_0 over all identified vertices of degree 1, and the supremum of D_0 over all identified vertices of degree at least 2. Given $x \in \widehat{\mathcal{G}}$, let $D(x)$ equal the distance of x to the nearest point of $\mathcal{G}$, and write $G(x)$ for the absolute value of the difference between the estimated gradient

of $\mathcal{G}$ at x, and the actual gradient of $\mathcal{G}$ at the point of $\mathcal{G}$ nearest to x. Define $D_{\sup} = \sup_{x \in \widehat{\mathcal{G}}} D(x)$ and $G_{\sup} = \sup_{x \in \widehat{\mathcal{G}}} G(x)$. An outline proof of the following theorem is given in Appendix 2.

Theorem. *Assume conditions (a)–(g). Then with probability 1, (i) for all sufficiently large ν the algorithm identifies exactly m_k vertices of degree k for $1 \le k \le k_0$; (ii) $D_2 = O\{(\nu h)^{-1} \log \nu + h^2\}$; (iii) $D_1 = o(1)$; (iv) $D_{\sup} = O\{(\nu h)^{-1} \log \nu + h^2\}$; and (v) $G_{\sup} = O\{(\nu h^2)^{-1} \log \nu + h\}$.*

Choosing h to be of size $\alpha = (\nu^{-1} \log \nu)^{1/3}$, for example to equal $C\alpha$ for some $C > 0$, we deduce from the theorem that in the Poisson case, all the errors in estimating points on $\mathcal{G}$, be they vertices or points on smooth edges, equal $O(\alpha^2)$. This is within the factor $(\log \nu)^{2/3}$ of the minimax-optimal pointwise rate, $\nu^{-2/3}$, for estimating a smooth edge that has bounded curvature. Likewise, we see from the theorem that for Poisson-distributed data, the convergence rate of our slope estimator, at smooth parts of edges, is $O\{\nu^{-1/3} (\log \nu)^{2/3}\}$. For gridded data, taking $h = C\beta$, where $\beta = (\nu^{-1} \log \nu)^{1/4}$, we see from the theorem that the convergence rate of our estimator, at smooth parts of edges, is $O(\beta^{1/4})$, which again is within a logarithmic factor of the minimax-optimal pointwise rate.

5. Discussion

In this final section, we summarise some observations on the application of the algorithm (subsection 2.5) which emerged from our experiments with real-world images, and which we hope to be helpful in practice. We first comment on aspects of the algorithm, in the order described in subsection 2.5, and conclude with some remarks of a general nature.

Regarding step (1): Among the recommendations that we make, one of the easiest is with regard to the choice of the starting point; see subsection 2.6. We recommend the use of many line transects rather than a single one. These may be chosen deterministically or even at random; as the example in subsection 3.2 illustrates, this enables capturing as large a portion of $\mathcal{G}$ as possible. Note that each of these starting points will have a relatively large value of T_0 (defined at (4)), and it is therefore plausible to select the threshold t_1 as a fraction of this initial value.

Regarding step (2): We suggest that it is a good idea to use a rather small

value of δ, not much larger than $\delta = 0.1$ which was used in section 3. Also, it may be expedient (though this is arguable) not to allow any knots to be detected if one had been detected within the last q (say) steps. We took $q = 5$.

Regarding step (3): In our simulations in subsection 3.2, we detected rather many knots, which were the result of small-scale image intensity variation and ensuing small edge segments. In order to be more sure of detecting 'genuine' knots, one may increase t_2 but in this way, many 'genuine' knots are not detected either. In images somewhat 'simpler' than this, with a well connected edge graph, this may not be too problematic if the (visually anticipated) knot is reached from another direction (if the edge graph is sufficiently connected or many starting points are used, as suggested above). Such a point may then be declared a knot even if this had not been done when the algorithm first passed that point.

In the detection of knots as well as corners, it is necessary to specify the minimum angle of the ray(s) that separates emanating from the point in question. Note that this may be seen as the finite-sample analogue of the "sharp vertex" assumption from the first paragraph of subsection 2.1. We found that using a value below 40°, we would get spurious corners on the top of the head and shoulders of the cameraman, which arise to due small-scale undulations. For similar reasons, we found it efficacious to restrict angles between rays at knots of order three to values no less than 45°.

Regarding step (4): We found that the line segment $\mathcal{L}(\hat{x}_{j-1}, \hat{\theta})$ is generally too long to give good performance. Specifically, the first new point on a ray at a knot is placed too far away from the knot and hence the emanating new edge may be missed altogether. Shortening that segment to half of its length is, as our experiments suggest, advantageous in practice.

Regarding step (5): We note that the matter of linking up edge pieces is far from straightforward. As a case in point, the presence of small features with high curvature parts such as the ear of the cameraman may make it advisable to link edges only when they are much closer than $(1 + \delta)h$. Of course, when checking for the nearest neighbours in the edge hitherto tracked, one always has to exclude several immediately preceding points.

We conclude with general comments. The potential of automation already emerges from the discussion above, but the most intricate problem in this regard

concerns the choice of bandwidth, and we see a lot of scope and necessity for future work here. Assumption (g) in Appendix 1 prescribes that $t_j = o(\nu h^2)$ for $j = 1$ and 2, and the choice $t_j \sim h^{5/2}$ from subsection 3.2 seems to work well in practice. On the question of how to determine the implied constants in those thresholds, in the case of t_1, the method described above may be a satisfactory answer. As to the choice of t_2, we recommend to check — possibly within a smaller sub-window only — for the number of knots detected with a relatively large threshold, and gradually decreasing the multiplier and hence t_2 until the number of knots roughly matches the possible expectation (if only a rough one) of the observer. To us, at least a partial automation of this procedure seems possible.

Acknowledgement. We are grateful to two reviewers for helpful comments.

References

Achour, K., Zenati, N. & Laga, H. (2001). Contribution to image and contour restoration. *Real-Time Imaging* **7**, 315–326.

Bhandarkar, S.M. & Zeng, X. (1999). Evolutionary approaches to figure-ground separation. *Appl. Intell.* **11**, 187–212.

Bhandarkar, S.M., Zhang, Y.Q. & Potter, W.D. (1994). An edge-detection technique using genetic algorithm-based optimization. *Patt. Recognition* **27**, 1159–1180.

Chan, F.H.Y., Lam, F.K., Poon, P.W.F., Zhu, H. & Chan, K.H. (1996). Object boundary location by region and contour deformation. *IEEE Proc. Vis. Image Sig. Process.* **143**, 353–360.

Eubank, R.L. & Speckman, P.L. (1994). Nonparametric estimation of functions with jump discontinuities. In: *Change-point problems*, Eds. E. Carlstein, H.-G. Müller & D. Siegmund, IMS Lecture Notes **23**, pp. 130–144. Institute of Mathematical Statistics, Hayward, CA.

Forsyth, D. & Ponce, J. (2003). *Computer vision: a modern approach*, 2nd edn. Prentice Hall, London.

Gonzalez, R.C. & Woods, R.E. (1992). *Digital image processing*. Addison-Wesley, Reading, MA.

Hall, P., Peng, L. & Rau, C. (2001). Local likelihood tracking of fault lines and boundaries. *J. R. Statist. Soc.* Ser. B **63**, 569–582.

Hall, P. & Rau, C. (2000). Tracking a smooth fault line in a response surface. *Ann. Statist.* **28**, 713–733.

Haralick, R.M. & Shapiro, L.G. (1992). *Computer and robot vision*, vol. 1 Addison-Wesley, Reading, MA.

Hartley, R. & Zisserman, A. (2001). *Multiple view geometry in computer vision*. Cambridge University Press, Cambridge, UK.

Jähne, B. & Haussecker, H. (2000). *Computer vision and applications: a guide for students and practitioners*. Academic Press, San Diego.

Kang, D.J. & Kweon, I.S. (2001). An edge-based algorithm for discontinuity adaptive color image smoothing. *Patt. Recog.* **34**, 333–342.

Kang, D.J. & Roh, K.S. (2001). A discontinuity adaptive Markov model for color image smoothing. *Image Vis. Comput.* **19**, 369–379.

Li, S.Z. (1995a). *Markov random field modeling in computer vision*. Springer, Tokyo.

Li, S.Z. (1995b). On discontinuity-adaptive smoothness priors in computer vision. *IEEE Trans. Patt. Anal. Mach. Intell.* **7**, 576–586.

Li, S.Z. (2000). Roof-edge preserving image smoothing based on MRFs. *IEEE Trans. Image Process.* **9**, 1134–1138.

Miyajima, K. & Mukawa, N. (1998). Surface reconstruction by smoothness map estimation. *Patt. Recog.* **31**, 1969–1980.

Mallat, S.G. (1999). *A wavelet tour of signal processing*, 2nd edn. Academic Press, San Diego.

Morel, J.-M. & Solimini, S. (1995). *Variational methods in image segmentation*. Birkhäuser, Boston.

Müller, H.-G. (1992). Change-points in nonparametric regression analysis. *Ann. Statist.* **20**, 737–761.

Müller, H.-G. & Song, K.S. (1994). Maximin estimation of multidimensional boundaries. *J. Multivar. Anal.* **50**, 265–281.

O'Sullivan, F. & Qian, M.J. (1994). A regularized contrast statistic for object boundary estimation — implementation and statistical evaluation. *IEEE Trans. Patt. Anal. Mach. Intell.* **16**, 561–570.

Park, S.C. & Kang, M.G. (2001). Noise-adaptive edge-preserving image restoration algorithm. *Opt. Eng.* **39**, 3124–3137.

Parker, J.R. (1997). *Algorithms for image processing and computer vision*. Wiley, New York.

Pedersini, F., Sarti, A. & Tubaro, S. (2000). Visible surface reconstruction with accurate localization of object boundaries. *IEEE Trans. Circ. Syst. Video Tech. 10*, 278–292.

Qiu, P.H. (1998). Discontinuous regression surfaces fitting. *Ann. Statist.* **26**, 2218–2245.

Qiu, P.H. (2002). A nonparametric procedure to detect jumps in regression surfaces. *J. Comput. Graph. Statist.* **11**, 799–822.

Qiu, P.H. (2004). The local piecewisely linear kernel smoothing procedure for fitting jump regression surfaces. *Technometrics* **46** (1), 87–98.

Qiu, P.H. (2005). *Image processing and jump regression analysis.* Wiley, New York.

Qiu, P.H. & Yandell, B. (1997). Jump detection in regression surfaces. *J. Comput. Graph. Statist.* **6**, 332–354.

Qiu, P.H. & Yandell, B. (1998). A local polynomial jump-detection algorithm in nonparametric regression. *Technometrics* **40**, 141–152.

Sinha, S.S. & Schunck, B.G. (1992). A 2-stage algorithm for discontinuity-preserving surface reconstruction. *IEEE Trans. Patt. Anal. Mach. Intell.* **14**, 36–55.

Yi, J.H. & Chelberg, D.M. (1995). Discontinuity-preserving and viewpoint invariant reconstruction of visible surfaces using a first-degree regularization. *IEEE Trans. Patt. Anal. Mach. Intell.* **17**, 624–629.

C. Rau, Department of Mathematics, Hong Kong Baptist University, Kowloon Tong, Hong Kong.
E-mail: rau@math.hkbu.edu.hk

Appendix 1: Regularity conditions for theorem

(a) The X_i's are either arranged deterministically on a regular triangular, square or hexagonal grid with density ν per unit area of $\mathbb{R}^2$, or are the points of a Poisson process with intensity $\nu\psi$ in the plane. The function ψ, not depending on ν, is assumed bounded away from zero on an open set $\mathcal{S} \subseteq \mathbb{R}^2$, and is taken to be identically equal to 1 when the points X_i are arranged on a grid. If the points X_i are all located at vertices of a regular grid then it is assumed the grid is supported throughout $\mathbb{R}^2$.

(b) The planar graph $\mathcal{G} \subseteq \mathcal{S}$ is connected and contains only finite numbers of edges and vertices; each edge between two vertices has a continuously turning tangent and bounded curvature; the degree of each vertex does not exceed a given integer $k_0 \geq 1$; and, for each $k \geq 3$, any point x of $\mathcal{G}$ which can be written as the limit along exactly k nondegenerate edge segments of $\mathcal{G}$, all of the segments distinct except that each contains x, is a vertex of degree k. If exactly two of these smooth edges meet at a point where the angle between their tangents does not equal 0 or π, then that point is a vertex of degree $k = 2$.

(c) The function f is continuous and has a uniformly bounded derivative in $\mathcal{S} \setminus \mathcal{G}$. For each $x \in \mathcal{G}$, $f(y)$ has a well-defined limit as $y \to x$ through any line segment $\mathcal{L}(x)$ that has one end at x and, excepting this point, is entirely contained in $\mathcal{S} \setminus \mathcal{G}$.

(d) For each $x \in \mathcal{G}$, write $\mathcal{L}_1(x)$ and $\mathcal{L}_2(x)$ for two versions of $\mathcal{L}(x)$, as introduced in (c) but having the following additional properties: given a small number $\eta_1 > 0$, let $v_r \in \mathcal{L}_r$ be distant exactly η_1 from $\hat{x}_j$; and suppose that for all sufficiently small η_1 the line that joins v_1 and v_2 does not lie entirely in $\mathcal{S} \setminus \mathcal{G}$. Let $d\{\mathcal{L}_1(x), \mathcal{L}_2(x)\} = |u_1 - u_2|$, where u_r denotes the limit of $f(y)$ as $y \to x$ through points on $\mathcal{L}_r(x)$, and write $d(x)$ for the infimum of $d\{\mathcal{L}_1(x), \mathcal{L}_2(x)\}$ over all choices of $\mathcal{L}_1(x)$ and $\mathcal{L}_2(x)$

that satisfy the conditions in the previous sentence. We assume that for each $\eta_2 > 0$, $d(x)$ is bounded away from zero uniformly in all points $x \in \mathcal{G}$ that are distant at least η_2 from each vertex of degree 1.

(e) The bandwidth $h = h(\nu)$ satisfies $h \to 0$ and $\nu^{1-\eta}h^2 \to \infty$, for some $\eta > 0$, as $\nu \to \infty$. The kernel K is a nonnegative, radially symmetric probability density, Hölder continuous in the plane and with its support equal to the unit disc centred at the origin. Moreover, the constant δ, used to define the step length δh, lies in the interval $(0, 1)$.

(f) For some point $y \in \mathcal{G}$ which is not a vertex, for some constant $C > 0$, and at each stage ν of the sequence of datasets $\mathcal{Z}$, the initial point $\hat{x}_0$ is within Ch^2 of y.

(g) (Recall that t_1 and t_2 are the two thresholds.) For some $c > 0$, and for $j = 1$ and 2, $\nu^c = O(t_j)$ and $t_j = o(\nu h^2)$.

Appendix 2: Outline proof of theorem

A.1. Outline proof of theorem. To appreciate why it is possible to assert, in the theorem, that an event occurs "with probability 1" as $\nu \to \infty$, rather than merely "with probability converging to 1", we note that it is necessary only to show that the probability of the event, for given ν, equals $1 - O(\nu^{-2})$ as $\nu \to \infty$. (This is readily done using the exponential bounds employed by Hall *et al.* (2001).) Then it will follow from the discreteness of ν, via the Borel-Cantelli lemma, that the desired assertion indeed holds true with probability 1.

We shall show that, if the algorithm starts at a point $\hat{x}_0$ that satisfies condition (f); and if the initial step is in the direction towards a vertex of degree $k_1 \geq 2$; then with probability 1, for all sufficiently large ν, the algorithm correctly deduces the degree of the vertex [call this result (R_1)], and with probability 1, for all sufficiently large ν, gets its location right to within $O(\epsilon)$ [result (R_2), say], where $\epsilon = (\nu h)^{-1} \log \nu + h^2$. We shall also prove that if the vertex towards which we move at the first step is of degree 1, then with probability 1, for all sufficiently large ν, the algorithm gets the degree of this vertex correct [result (R_3)], and gets its position right to within $o(1)$ [result (R_4)]. Moreover, at those points of the trajectory before we determine that we are close to a vertex, the location and the slope of the edge estimator are accurate to within $O(\epsilon)$ and $O(\epsilon/h)$, respectively [result (R_5)].

These results can be developed into a complete proof of the theorem. This is done by considering restarting the algorithm (as prescribed in subsection 2.5) at the vertex estimator, moving away from that vertex along each edge estimator; using methods identical to those below to prove the analogous properties over the subsequent edges of the graph; and noting that with probability 1, for all sufficiently large ν only a finite number of such stages is involved. If this is done, then properties (i)–(v) in the theorem follow from, respectively, results (R_1) & (R_3), (R_2), (R_4), (R_5) and (R_5).

Observe that we may write

$$T_k(x_0) = 2 \sum_i \left\{ Y_i - \tfrac{1}{2} \left(\hat{f}_{ki} + \hat{f}_{k+1,i} \right) \right\} \left(\hat{f}_{k+1,i} - \hat{f}_{ki} \right) K_i \,, \tag{6}$$

where $\hat{f}_{ki} = \hat{f}_k(X_i \,|\, x_0, \hat{\theta}_1, \ldots, \hat{\theta}_k, \hat{c}_1, \ldots, \hat{c}_k)$, with an analogous formula applying for $\hat{f}_{k+1,i}$, and, here and below, we write simply K_i for $K_i(x_0)$. Put

$$I_{kji} = \begin{cases} 1 & \text{if } \hat{f}_k(X_i \,|\, x_0, \hat{\theta}_1, \ldots, \hat{\theta}_k, \hat{c}_1, \ldots, \hat{c}_k) = \hat{c}_j \\ 0 & \text{otherwise}\,, \end{cases}$$

and note that $\sum_i (Y_i - \hat{c}_j)\, K_i\, I_{kji} = 0$. It follows that $\sum_i Y_i\, \hat{f}_{ki}\, K_i = \sum_i \hat{f}_{ki}^2\, K_i$, with the analogous result holding if k is replaced by $k+1$. Therefore, by (6),

$$T_k(x_0) = \sum_i \left(\hat{f}_{k+1,i}^2 - \hat{f}_{ki}^2\right) K_i\,. \tag{7}$$

Now, $\hat{f}_{0i}$ does not depend on i, from which property, and (7) in the case $k = 0$, it follows that $T_0(x_0) = \sum_i (\hat{f}_{1i} - \hat{f}_{0i})^2\, K_i$. A similar argument shows that if we define $\hat{c}_{1\theta}$ and $\hat{c}_{2\theta}$ to be the quantities c_1 and c_2 that minimise $\sum_i \{Y_i - \hat{f}_1(X_i \,|\, x_0, \theta, c_1, c_2)\}^2\, K_i$, for fixed θ, and put $\hat{f}_{1i}(\theta) = \hat{f}_1(X_i \,|\, x_0, \theta, \hat{c}_{1\theta}, \hat{c}_{2\theta})$, then the following is a definition alternative, but equivalent, to that at (4):

$$T_0(x_0) = \sup_\theta t(\theta) \quad \text{where} \quad t(\theta) = t(x_0, \theta) = \sum_i \left\{\hat{f}_{1i}(\theta) - \hat{f}_{0i}\right\}^2 K_i\,. \tag{8}$$

Using the representation (8), and employing the argument in the appendix of Hall *et al.* (2001), the following can be proved. If we start the algorithm at a point $\hat{x}_0$ which satisfies assumption (f) then, with probability 1, the subsequent points generated by the algorithm, up until the point immediately preceding the one ($\hat{x}_j$, say; the same $\hat{x}_j$ as in step (3) of the algorithm in subsection 2.5) at which we conclude we are close to a vertex, are uniformly within $O(\epsilon)$ of the nearest points on $\mathcal{G}$, and the gradient estimates are uniformly within $O(\epsilon/h)$ of the gradients at the respective nearest points. This gives result (R_5).

Let c denote the constant appearing in condition (g), imposed on the thresholds t_1 and t_2. If the vertex towards which the first of the above sequence of steps is taking us is of degree 1 then the following can be proved to hold. (I) With probability 1, for all sufficiently large ν, one of the sequence of values $T_0(\hat{x}_\ell)$, for $\ell \geq 1$, will fall below ν^c strictly before one of the sequence of values of $\max_{2 \leq k \leq k_0} T_k(\hat{x}_\ell)$ exceeds t_2. (II) If Δ denotes the distance from the vertex (of degree 1) at which $T_0(\hat{x}_\ell)$ first drops below ν^c, then $\Delta \to 0$ with probability 1. Properties (I) and (II) imply that with probability 1, for all sufficiently large ν the algorithm correctly identifies the degree of the vertex of degree 1 [result (R_3)]; and moreover, the distance between the point at which the algorithm stops, and the vertex of degree 1, converges to 0 [result (R_4)].

Assume next that the first in the sequence of steps discussed two paragraphs above takes us in the direction of a vertex, z say, of degree $k_1 \geq 2$. Recall that δh equals step length. It can be proved that with probability 1, for all sufficiently large ν the point $\hat{x}_{j-1}$ at which the sequence terminates is within $(1+\delta)h$ of z. Let $\zeta = \zeta(\nu)$ denote any positive sequence of constants decreasing to 0 as $\nu \to \infty$, so

slowly that $\zeta\nu^{1-c}h^2 \to \infty$, where c is the constant in condition (g) on the threshold values. Recall the definition of $\mathcal{L}(\hat{x}_{j-1}, \hat{\theta})$ given in step (3) of the algorithm, in subsection 2.5. Since the vertex at z is of degree $k_1 \geq 2$ then we may prove from (A.2) that with probability 1, the following are true: (III) for all $\eta > 0$, $\sup \inf T_k(x) = O(\nu^\eta)$, where the supremum is over all $k \in [k_1, k_0]$ and the infimum is over all $x \in \mathcal{L}(\hat{x}_{j-1}, \hat{\theta})$; and (IV) for all sufficiently large ν, $\zeta\nu h^2 \leq \inf T_k(x)$, where now the infimum is over all $k \in [0, k_1 - 1]$ and all $x \in \mathcal{L}(\hat{x}_{j-1}, \hat{\theta})$.

Hence, noting the definition of $\hat{k}$ in step (3) of the algorithm, and the conditions on t_2 imposed in assumption (g), we deduce that with probability 1, $\hat{k} = k_1$ for all sufficiently large ν. This gives result (R_1). Noting that with probability 1, (V) $\hat{x}_{j-1}$ is distant $O(\epsilon)$ from the nearest point on $\mathcal{G}$, (VI) the slope estimator $\hat{\theta}$ at $\hat{x}_{j-1}$ is in error by only $O(\epsilon/h)$, and (VII) for all sufficiently large ν, z is itself within $(1+\delta)h$ from $\hat{x}_{j-1}$, we conclude that: with probability 1, the point z is perpendicularly distant $O(h^2)$ from the line segment along which we search for our estimator of z. This fact, and property (7), may be used to establish result (R_2).

Caption for Figure 1: Planar graph, $\mathcal{G}$. Vertices A, B, C and D in the graph are of degrees 1, 2, 3 and 4, respectively. In particular, vertex A is the endpoint of an edge.

Caption for Figure 2: Perspective depiction of the surface represented by $y = f(x)$, for $x \in \mathbb{R}^2$. This f satisfies condition (F) for a graph $\mathcal{G}$ simpler than the one shown in Figure 1.

Caption for Figure 3: Perspective representation of surface defined by $y = f_1(x)$, for $x \in \mathbb{R}^2$, with f_1 given by (2).

Caption for Figure 4: Panel (a) shows the noisy response surface defined by the function f from section 3, where the error distribution is $N(0, \sigma^2)$ with $\sigma = 0.25$. Panel (b) shows a realisation of the tracking estimate $\widehat{\mathcal{G}}$, corresponding to (a). The estimate was computed as described in subsection 2.5. The four estimated knots are circled. Panel (c) shows the result of aggregating 20 realisations as described in section 3.

Caption for Figure 5: Central part of original Cameraman image, as used for estimation.

Caption for Figure 6: Panels (a)–(c) show estimates of the edge obtained by the Sobel, Laplacian of Gaussian (LoG), and the method of this paper, respectively. The circled points in panel (c) are the knots of order three. Panel (d) shows the results after linearly connecting the sequence of points from (c), alongside the knots and corners. In each of panels (a)–(d) a small rim around the estimation window Π is added for visual purposes.

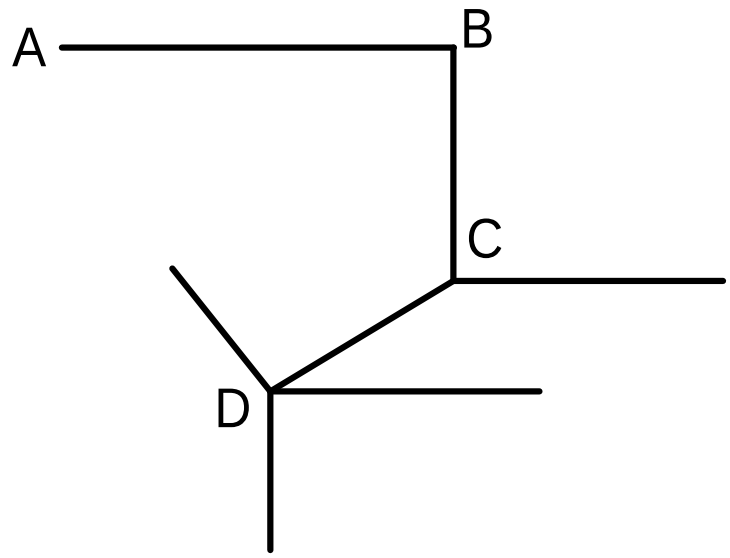


Fig. 1. Planar graph, $\mathcal{G}$. Vertices A, B, C and D in the graph are of degrees 1, 2, 3 and 4, respectively. In particular, vertex A is the endpoint of an edge.

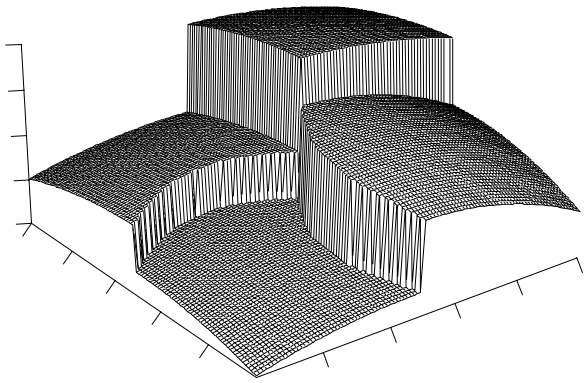


Fig. 2. Perspective depiction of the surface represented by $y = f(x)$, for $x \in \mathbb{R}^2$. This f satisfies condition (F) for a graph $\mathcal{G}$ simpler than the one shown in Figure 1.

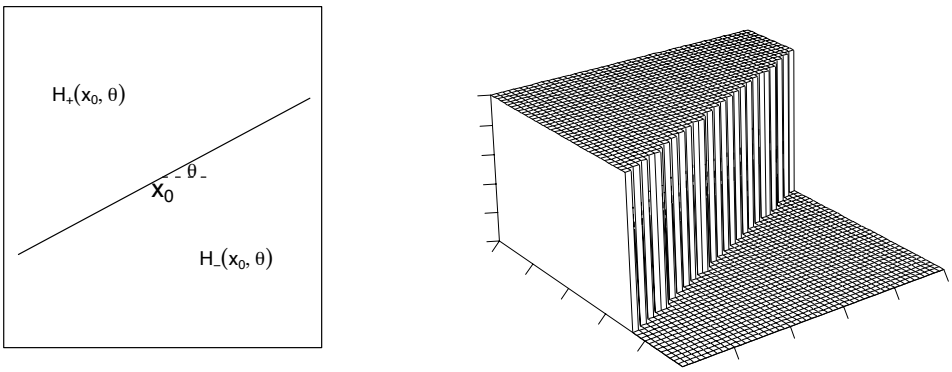


Fig. 3. Perspective representation of surface defined by $y = f_1(x)$, for $x \in \mathbb{R}^2$, with f_1 given by (2).

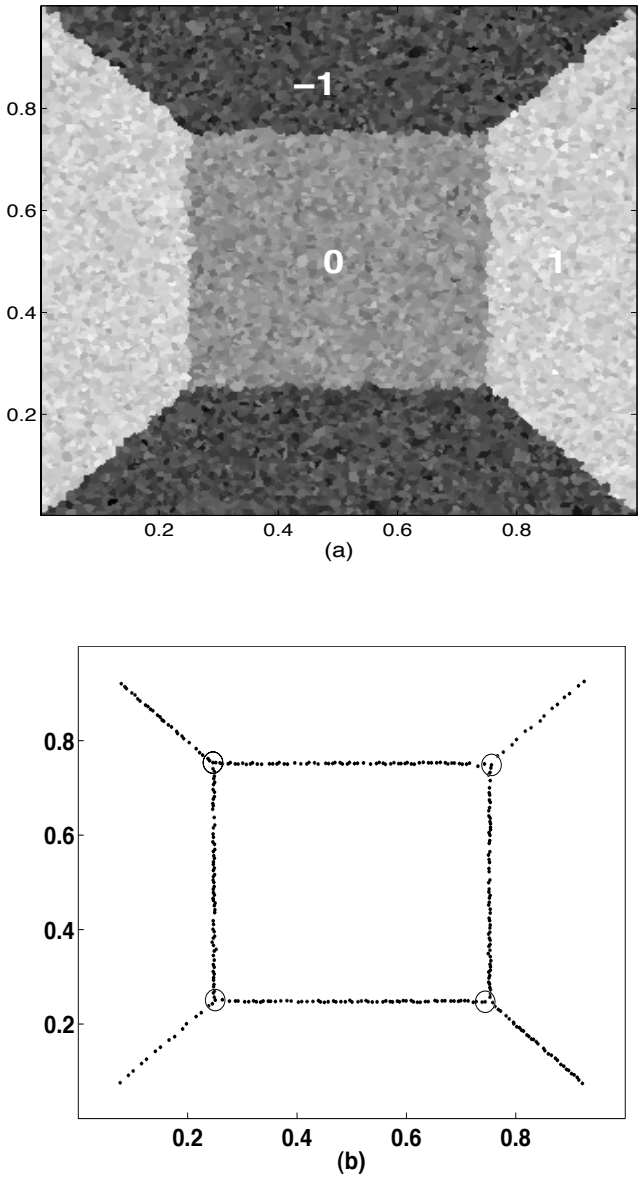
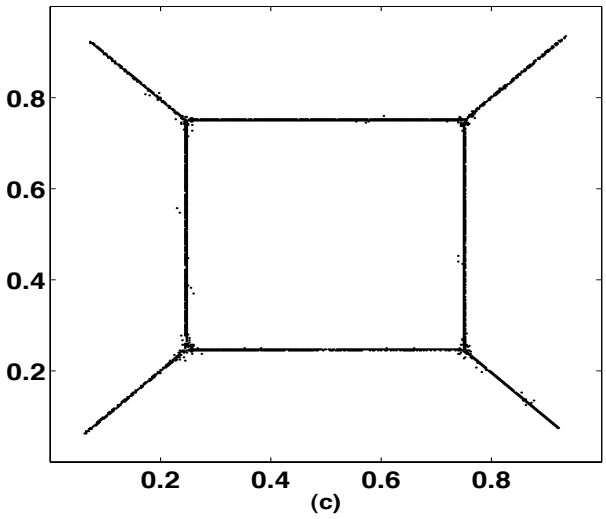


Fig. 4. Panel (a) shows the noisy response surface defined by the function f from section 3, where the error distribution is $\mathrm{N}(0, \sigma^2)$ with $\sigma = 0.25$. Panel (b) shows a realisation of the tracking estimate $\widehat{\mathcal{G}}$, corresponding to (a). The estimate was computed as described in subsection 2.5. Panel (c) shows the result of aggregating 20 realisations as described in section 3.


0.8
0.6
0.4
0.2
0.2
0.4
0.6
0.8
(c)

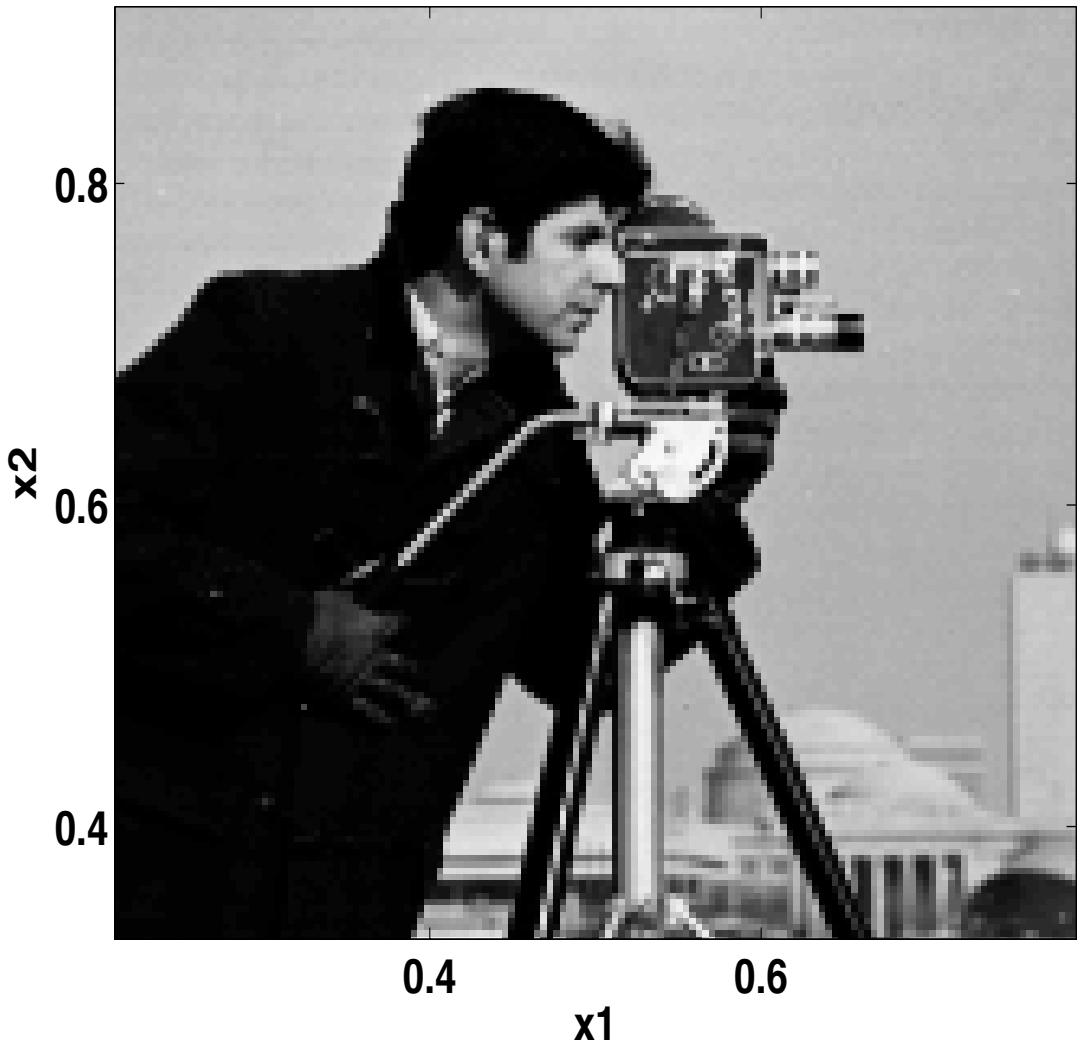


Fig. 5. Central part of original Cameraman image, as used for estimation.

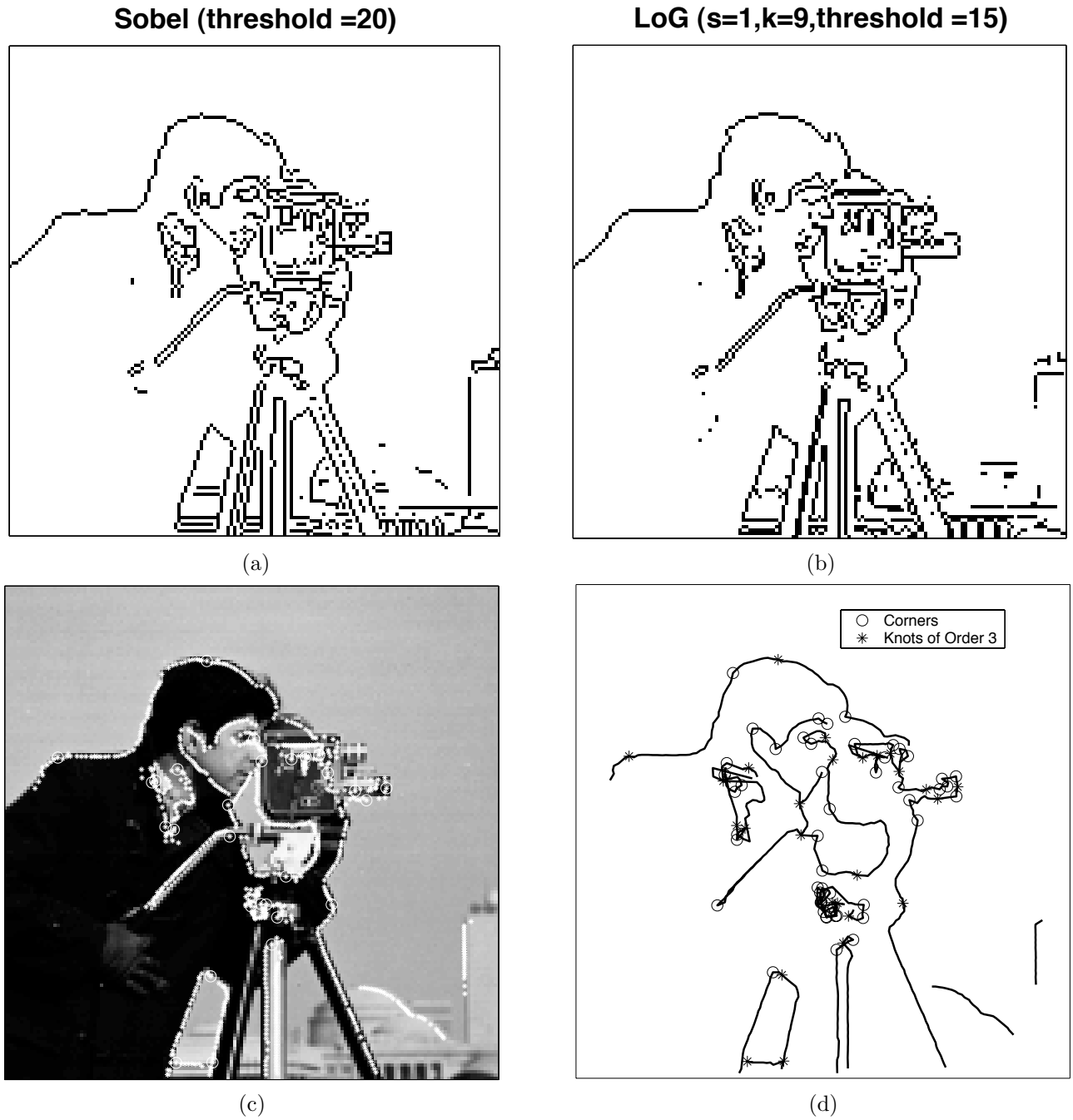


Fig. 6. Panels (a)–(c) show estimates of the edge obtained by the Sobel, Laplacian of Gaussian (LoG), and the method of this paper, respectively. The circled points in panel (c) are the knots of order three. Panel (d) shows the results after linearly connecting the sequence of points from (c), alongside the knots and corners. In each of panels (a)–(d) a small rim around the estimation window Π is added for visual purposes.