ChatGPT

Mathematical Methods for Detecting Scratches and Digs on a Fiber End-Face Pixel Grid

Finding scratches (linear/jagged line defects) and digs (roughly circular blob-like defects) in a pixel intensity matrix involves a range of advanced mathematical techniques. Below is an **exhaustive list of sophisticated computational methods** – spanning calculus (derivatives and differential equations), linear algebra (matrix decompositions, eigen-analysis), and physics-inspired models – that can be employed to locate and segment these defects. All methods are *non-learning-based* (suitable for manual calculation or simulation) and include strategies for robustness against noise and pixelation artifacts. Each item describes the mathematical formulation and how it helps detect **line-like scratches** or **blob-like digs** (which may appear darker, brighter, or with variable contrast against the background).

1. Gradient-Based Edge Detection (First-Order Derivatives)

Image Gradient & Edge Operators: Computing the partial derivatives of the intensity image $I(x,y)$ highlights locations of abrupt intensity change (edges). For example, *Sobel* or *Prewitt* operators approximate $\partial I/\partial x$ and $\partial I/\partial y$ via convolution kernels, yielding a gradient vector at each pixel. The **gradient magnitude** $|\nabla I| = \sqrt{I_x^2 + I_y^2}$ is high along scratches/digs edges. Edges can be identified where the gradient magnitude exceeds a threshold. This method leverages calculus (first derivatives) and linear algebra (convolution as a matrix operation). However, raw gradients are noise-sensitive – typically one **pre-smooths** the image with a Gaussian filter to create a scale-space (suppressing high-frequency noise). Notably, the *Canny edge detector* formalizes this: it smooths with a Gaussian, computes gradients, then finds **local maxima of gradient magnitude** (edge pixels) and applies **dual thresholds with hysteresis** to robustly connect edge segments [1]. Canny's design was mathematically optimized for good detection and localization under noise. **Result:** Gradient methods mark intensity edges outlining scratches (often long, possibly jagged lines) and the boundaries of digs (approximate circular outlines). These edges form the basis for higher-level detection.

Gradient Direction Consistency: Scratches are elongated, so along a scratch the gradient orientations are consistent (normal to the scratch direction). By checking that a sequence of edge pixels has similar gradient directions, one can reinforce true linear features and suppress spurious edges. This uses vector calculus on the gradient field and linear algebra for correlation of orientation.

Noise Robustness: Smoothing (Gaussian convolution) is integral – it implements a *linear diffusion* (the heat equation) to reduce noise. In fact, the Gaussian-filtered image $I(x,y,\sigma)$ as σ increases defines a **scale-space**; derivatives on this multi-scale representation ensure that only structures of a certain size "pop out" [2] [3]. One may choose a larger smoothing scale for very noisy images, trading off edge localization. Additionally, **non-maximum suppression** (as in Canny) refines thick gradient responses to single-pixel thin edges by analytically finding where the gradient magnitude attains a peak perpendicular to the edge.

2. Second-Order (Laplacian) and Multi-Scale Blob Detection

Laplacian of Gaussian (LoG) for Blobs: The *second derivative* (Laplacian) highlights intensity extrema. The Laplacian $\nabla^2 I = I_{xx} + I_{yy}$ can detect blob-like features as spots of intensity different from surroundings. In practice, to handle noise and scale, one applies a Gaussian smoothing then the Laplacian – the **Laplacian of Gaussian (LoG)** operator [2] [4]. LoG produces strong positive responses at dark **blobs** (digs if they appear as darker spots) and strong negative responses at bright blobs [4]. By looking for **zero-crossings** of the LoG output (the classic Marr-Hildreth edge detection), one finds closed contours around blobs or scratches (since a scratch could appear as a long valley or ridge producing LoG zero-crossings along its edges).

Multi-Scale Extrema & Blob Segmentation: A single Laplacian filter is tuned to a particular feature size. To detect digs of unknown radius (or scratches of various thickness), a **scale-space search** is done: the image is convolved with LoG at multiple scales (Gaussian kernel with variance t). The Laplacian response must be *normalized* by scale (e.g. $t\,\nabla^2 I$) to compare across scales [5]. Blobs are detected as **local extrema in the 3D space of (x,y,scale)** [6] – points where the normalized LoG is maximal w.r.t. both position and scale. Such points give the blob center and an estimate of its size (scale ${\hat t}$ relates to blob radius $r \approx \sqrt{2\,{\hat t}}$ for 2D images [7]). This multi-scale Laplacian method is analytically derived from Gaussian scale-space theory and yields a **mathematically precise definition of "blob"** [8] [9]. It is robust to noise due to initial smoothing, and **invariant to image scale and rotation**, meaning a dig will be found regardless of slight size/angle differences [9]. (In fiber images, digs of varying diameter can thus be detected by their Laplacian peaks at corresponding scales.)

Difference of Gaussians (DoG): The DoG is a convenient approximation to LoG, computed as the difference of two Gaussian-smoothed images at nearby scales [10] [11]. It also detects blobs via scale-space extrema and is computationally cheaper. DoG and LoG both rely on calculus (2nd derivatives) and are solved via convolution (linear algebra operations on the intensity matrix).

Laplacian Edge Detection: Beyond blob centers, the Laplacian $\nabla^2 I$ also highlights **edges as zero-crossings**. For example, a scratch appearing as a dark line (intensity valley) will produce a Laplacian sign-change crossing the line. Thus, zero-crossing contours of LoG can outline both scratch lines and dig perimeters. This is a more **analytical formulation** of edge detection, where one solves $\nabla^2 I = 0$ for edge curves.

3. Hessian Matrix and Differential Geometry of Ridges

Hessian Eigenvalues (Ridge/Valley Detection): The **Hessian matrix** $H(x,y)$ of second-order partials gives rich geometric information about intensity variations [12] [13]. For an image $I(x,y)$,

$$H = \begin{pmatrix} I_{xx} & I_{xy} \\ I_{xy} & I_{yy} \end{pmatrix},$$

and its eigenvalues λ_1, λ_2 (with $|\lambda_1| \le |\lambda_2|$) indicate curvature along principal directions [14]. Line-like **ridges or valleys** can be identified by specific Hessian conditions. For instance, a **ridge (bright line)** is a curve of local intensity maxima in one cross-section: mathematically,

along the ridge the gradient in the direction of the ridge is zero and the second derivative across the ridge is negative (indicating a local maximum) [15]. Similarly, a **valley (dark line)** like a dark scratch on a bright background is a curve of local minima: gradient along it zero, second derivative across it positive [16]. In formal terms, if one rotates coordinates (p,q) so that q-axis is along the ridge, the conditions for a ridge are:

$$I_p = 0, \qquad I_{pp} < 0, \qquad |I_{pp}| \geq |I_{qq}|,$$

and for a valley (dark line):

$$I_q = 0, \qquad I_{qq} > 0, \qquad |I_{qq}| \geq |I_{pp}|\,.$$

These are precise differential equations describing line features [15]. They essentially say one principal curvature (second derivative) is large in magnitude (and of appropriate sign for ridge vs valley) while the orthogonal curvature is smaller.

In practice, one computes H (often on a Gaussian-smoothed image for noise robustness) and examines its eigenvalues λ_{1},λ_{2} at each pixel [17]. **Line Detection via Eigenvalues:** For a line (ridge/valley) oriented through a pixel, one eigenvalue will be large in magnitude and the other much smaller. For example, a dark line yields $\lambda_{1} \approx 0$ (along the line direction, little curvature) and $\lambda_{2} \gg 0$ (perpendicular to line, intensity has a steep trough). A blob, by contrast, gives eigenvalues of similar magnitude (curvature in all directions). Hence a **"vesselness" or line-likeliness measure** can be defined as a function that is high when $\lambda_{1} \ll \lambda_{2}$ (with the sign of λ_{2} indicating dark or bright line). The **Frangi vesselness filter** is a classical implementation: it uses the Hessian eigenvalues to compute the probability that a pixel is on a tubular structure (line) as opposed to noise or blob [18]. Essentially, Frangi's formula penalizes λ_{1} (to favor one dominant curvature) and looks for λ_{2} of the appropriate sign (negative for bright ridge, positive for dark valley) [18]. By scanning multiple scales (since a scratch can have varying width), the Hessian-based filter at each scale highlights line structures and is robust to noise (Gaussian smoothing is inherent).

Determinant of Hessian (DoH) for Blobs: A related measure is the **Hessian determinant** $\det(H) = I_{xx} I_{yy} - I_{xy}^2$. This is used in blob detection – e.g. the SURF feature detector uses the determinant of Hessian as a blob indicator [19]. The scale-normalized $\det(H)$ can be treated similar to LoG for finding blob extrema in scale-space [20] [21]. It will respond strongly to both bright and dark blobs (and even saddle points), offering a blob-detection method alternative to LoG [22]. DoH tends to have good localization and some affine invariance [23]. We include it here as another **matrix equation** (computing a determinant) derived from second derivative calculus that assists in identifying blob-like digs.

By using Hessian eigen-analysis and related invariants, one exploits **advanced linear algebra** (eigendecomposition) and **differential geometry** of the intensity surface to find candidate scratches (as curvilinear ridges/valleys) and digs (as 2D curvature maxima or blobs). These methods are **robust to intensity variations** – they rely on local second-derivative structure rather than absolute intensity, so a scratch that is slightly lighter or darker than the background is detected as long as it forms a ridge/valley shape in the intensity landscape.

4. Structure Tensor and Orientation Analysis

While the Hessian captures *second-order* structure, the **structure tensor** focuses on *first-order* statistics of the gradient in a neighborhood. The structure tensor $S(x,y)$ is the locally averaged outer product of the gradient vector with itself [24] [25]. In 2D:

$$S = \begin{pmatrix} \overline{I_x^2} & \overline{I_x I_y} \\ \overline{I_x I_y} & \overline{I_y^2} \end{pmatrix},$$

where the overline denotes averaging (often with a Gaussian window) over a neighborhood [26]. This symmetric positive-semidefinite matrix's eigenvalues μ_1,μ_2 (with $\mu_1 \ge \mu_2 \ge 0$) encode the **energy and anisotropy** of local gradients [27] [28]. Specifically:

- If **one eigenvalue is much larger** than the other ($\mu_1 \gg \mu_2$), the gradient vectors largely align in one direction – indicating an **edge or line structure** in that orientation [29]. The corresponding eigenvector gives the dominant orientation (perpendicular to a line or along the edge) [27]. A scratch, being a coherent linear feature, will produce a high anisotropy measure (nearly all gradients perpendicular to the scratch, so $\mu_1 \gg \mu_2$).
- If **both eigenvalues are large and comparable**, gradients point in many directions (corner or blob region – e.g. a dig's interior might have isotropic texture) [28].
- If **both are small**, the region is nearly uniform (no significant gradients).

Thus, computing the structure tensor (which involves calculus for gradients and linear algebra for eigen-decomp) and then the **coherence measure** $c = (\mu_1 - \mu_2)/(\mu_1 + \mu_2)$ [30] provides a robust indicator of line-like features: $c \approx 1$ for a clear line, $c \approx 0$ for an isotropic blob or flat area. One can map each pixel to a coherence value and threshold it to **segment candidate scratches** (high coherence pixels forming elongated regions). This method inherently averages over neighborhoods, making it **robust to noise and pixelation** – opposite gradient directions on either side of a line do not cancel out because the outer-product captures orientation without sign ambiguity [25] [31].

Orientation Field: Additionally, the structure tensor's principal eigenvector yields the **orientation** of the scratch. By continuity, one can track a line by following the local dominant orientation (this is akin to tracing a ridge via gradient flow). Physically, this resembles how **coherence-enhancing diffusion** works: using the structure tensor to guide smoothing along the direction of a line (and not across it) preserves scratches while reducing noise [32] [33]. While diffusion is discussed below, it's worth noting here that the structure tensor is often an intermediate calculation in advanced noise-reduction that keeps line structures.

In summary, the structure tensor provides a **matrix equation approach** (eigen-analysis of a 2×2 tensor at each pixel) to detect and characterize edges/lines versus blobs. It is a **PhD-level tool** widely used in computer vision for corner detection (e.g. Harris detector uses the trace and determinant of S) and for anisotropic filtering. For our purpose, it helps find scratches as oriented, anisotropic features, distinguishing them from digs (more isotropic) or noise (low energy).

5. Hough and Radon Transforms for Parametric Line Detection

Hough Transform (HT) for Lines: The Hough transform is a *voting algorithm* that detects geometric shapes by mapping image pixels to a parameter space [34] [35]. For straight lines, the **normal form** $x\cos\theta + y\sin\theta = r$ is used [36] [37], where θ is the line orientation and r the perpendicular distance from origin [38] [39]. Each edge pixel votes for all (r,θ) pairs that could describe a line through that pixel, effectively drawing a sinusoidal curve in Hough (parameter) space [40]. **Collinear pixels** (i.e. a scratch forming a line) yield curves that **intersect at a common (r,θ)** – a peak in the accumulator array [40] [41]. By finding accumulator peaks (via thresholding or peak finding algorithms), one extracts line candidates (the most supported (r,θ) values correspond to lines in the image) [42] [43]. The HT is **robust to noise and gaps**: even if a scratch is faint or broken, as long as enough edge pixels align roughly linearly, the votes accumulate constructively. It can detect **approximately straight** scratches; a jagged scratch will often be detected as a series of linear segments or a line covering its general direction.

Mathematically, the Hough transform implements an **integration in a transformed domain**, closely related to the continuous **Radon transform** (which integrates image intensity along lines at given angles) [44]. In fact, *the Hough transform is mathematically equivalent to the Radon transform* [45] – both project the image onto lines of various orientations. The difference is HT uses a *discrete accumulator* and is usually applied to a binary edge map. This is a clear example of using an **integral geometry approach** (from calculus) to detect shapes.

For our case, after getting an edge map (via any method from Section 1 or 2), applying the Hough transform will yield peaks corresponding to the main scratch lines. It provides line equations (angle and distance) even under noisy conditions [34] [35]. One can then analytically compute the line overlay or segment the pixels near those parameters. The standard HT has extensions: the *probabilistic Hough* (randomly sample votes), *Hough for circles* (3D parameter space for center and radius), etc., which could detect circular digs as well.

Circular/Blob Detection via Hough: A dig that is perfectly circular can be detected by a circular Hough transform: edge points vote in a (a,b,R) space for potential circle centers (a,b) and radius R. Peaks in that accumulator give circle parameters. This is computationally heavy (3D accumulator), so in practice blob detection (LoG/DoH methods above) or connected-component analysis (below) are often used for digs. But it's worth noting as a *parametric geometric detection* method.

In summary, the Hough transform is a **highly computational voting procedure** (often $O(nr\theta)$ operations for an n-pixel edge map) but very powerful for extracting **global line structures** amidst noise. It complements local methods (gradient/Hessian) by accumulating evidence of collinearity. The use of trigonometry ($\cos\theta,\sin\theta$) and accumulator arrays can be seen as applying a **discrete inverse Radon transform** to infer lines [44] – a direct application of linear algebra and calculus concepts for feature extraction.

6. Morphological Image Processing Techniques

Mathematical Morphology: Morphological operations treat the image as a set and use *set algebra* on pixel neighborhoods. They are effective for emphasizing certain shapes (lines or blobs) and removing noise without blurring edges. Two fundamental operations: **erosion** (shrink bright regions or grow dark regions)

and **dilation** (grow bright regions or shrink dark regions), defined via a *structuring element* (SE). By choosing line or disk-shaped SEs, one can target scratches or digs:

- **Line SE Filtering:** Using a linear structuring element oriented in a certain direction, one can perform an *opening* (erosion followed by dilation) to remove thin lines not aligned with that SE. Conversely, a *morphological hit-or-miss* with a line SE can detect linear segments of that orientation. Scanning multiple orientations, scratches show strong responses when the SE is aligned. This is a brute-force shape convolution in the spatial domain (somewhat like a template match using set operations). Jagged or curved scratches can be detected by shorter line SEs or iteratively bridging small segments.

- **Top-hat Transform:** This is a powerful tool for blob detection and uneven background correction. A **white top-hat** is defined as the original image minus its opening (with a certain SE) [46]. For instance, using a *disk SE roughly the size of expected digs*, the white top-hat will yield an image that highlights small bright spots (or dark spots, if using the black top-hat variant) that were not removed by the opening [47] [48]. In other words, it extracts small elements (like digs) from a background. If scratches appear dark, one might invert the image or use the black top-hat (which is original minus a closing) to get dark defects on bright background. **Morphological gradient** (dilation minus erosion) is another related operation that highlights edges of objects (like an edge detector but robust to noise speckles).

Connected Component Labeling: A simpler mathematical approach (once a preliminary segmentation exists) is to apply connectivity rules. For example, threshold the image to isolate darker-than-background regions (for dark defects) or use the edge map to fill enclosed regions. Then use **connected component analysis** to label continuous pixel groups. This yields candidate defect regions. One can compute shape descriptors (e.g. eccentricity, area) for each component: scratches will be long and thin, whereas digs are compact and round. Although threshold selection is sensitive, one can incorporate **iterative thresholding** or **morphological reconstruction** to gradually peel off background illumination and isolate defects. These are algorithmic but grounded in set theory and topology.

Morphological methods are typically **discrete mathematics** approaches, but they can be very sophisticated (e.g., morphological reconstruction solves for regional maxima maintaining connectivity). They excel at handling **discretization artifacts**: e.g., an opening can remove single-pixel noise; a closing can fill single-pixel gaps in a scratch line. They don't rely on gradients, so they are **robust to intensity fluctuations** (a scratch that is only slightly darker than background can still be picked out if it forms a contiguous path of dark pixels).

7. Template Matching and Correlation Methods

This class involves *analytical convolution/correlation calculations* using a predefined template that resembles the defect shape:

- **Matched Filters for Lines:** One can design a filter (kernel) that looks like a short line segment. For example, a straight-line template of a certain length L (with values +1 along the line and 0 elsewhere, or a bar of dark/bright intensity) can be correlated with the image. Where the image contains a similar aligned segment, the correlation output is high. By rotating the template, different orientations are checked. The mathematical operation is cross-correlation, essentially a sum of

products (inner product) between the template matrix and image patches – a linear algebra operation. This is *computationally heavy* if done exhaustively, but it directly uses the known shape of scratches. It's robust if scratches have roughly known width/contrast. For jagged scratches, smaller line segments can be used and then linked.

- **Bank of Oriented Filters (Gabor filters):** A more advanced version uses *Gabor filters*, which are Gaussians modulated by a sinusoid, oriented at various angles. A Gabor filter at orientation θ and frequency f will respond strongly to line patterns oriented at θ (acting as a band-pass filter in the Fourier domain for that orientation). By applying a bank of Gabor filters (or other oriented wavelets) and measuring the energy response, one can detect oriented features like scratches. Mathematically, Gabor filters implement a localized Fourier transform – an integral of image intensity times a complex exponential – capturing specific frequency/orientation content. They are known to be optimal in the sense of jointly localized in space and frequency. High response from a Gabor filter indicates presence of an oriented edge/line at the chosen scale. Summing responses across scales can even handle multi-scale lines. This approach is resilient to noise due to the filtering's band-pass nature (noise is broad-spectrum and can be attenuated outside the passband).

- **Circular Template for Digs:** Similarly, a circular or Gaussian spot template can be used to find blobs. Correlating a Gaussian of a certain sigma with the image is analogous to LoG but simpler: peaks in the correlation indicate similarity to that blob size. This is essentially template matching for digs. One can also perform normalized cross-correlation to handle varying brightness (subtract mean intensity in the window).

The above correlation methods involve **convolution integrals** and can be seen as implementing a **filter bank**. They are straightforward to conceptualize (find where image looks like this template) and can be very effective if defect shapes are consistent. They also allow using *physics intuition*: for example, modeling a scratch's intensity profile as a line-spread function and using that as a filter. However, they are computational (trying many positions, orientations, scales), though techniques like FFT convolution can accelerate matching by using the convolution theorem (multiplying spectra in Fourier domain).

Robustness: Template matching is sensitive if the defect shape deviates from the template or if the background isn't uniform (leading to false matches). To improve robustness, one can use correlation after high-pass filtering the image (so only local contrast matters), or incorporate multiple templates (e.g., line with slight curvature). It's a deterministic approach that can complement the more general edge detectors by explicitly checking the expected shape.

8. Variational Models and PDEs for Segmentation

Advanced segmentation can be achieved by formulating an **energy functional** that encodes desirable properties (e.g. defect regions have different intensity than background, and boundaries are smooth or of minimum length) and then minimizing this energy via calculus of variations or graph methods. Two influential approaches:

- **Active Contours (Snakes):** An active contour is a parametric curve $\mathbf{v}(s)$ that moves through the image, driven by energies [49] [50] . The energy functional typically has an **internal regularization term** (encouraging smooth, minimal-length curves) and an **external image term**

that attracts the curve to edges or specific intensity regions [51] [52]. For example, a basic snake energy is:

$$E_{\text{snake}} = \int_0^1 \Big[\underbrace{\alpha |\mathbf{v}'(s)|^2 + \beta |\mathbf{v}''(s)|^2}_{E_{\text{internal (elasticity \& rigidity)}}} \; - \; \underbrace{\gamma \, |\nabla I(\mathbf{v}(s))|^2}_{E_{\text{image}}} \Big] ds \,,$$

where the image term is designed to be low (minimal energy) at strong edges (or one can use $-|\nabla I|^2$ so energy is minimized when snake sits on edges) [53]. Through **Euler-Lagrange equations**, one derives a gradient descent PDE for the curve's coordinates (often solved iteratively) [54]. The snake will lock onto object boundaries – in our case, if initialized around a suspected defect, it will tighten onto the scratch or dig edges, even if jagged. There are region-based snakes too: the **Chan–Vese model** assumes two regions (inside and outside the contour) each have roughly constant intensity. It defines an energy based on the difference of each region's intensity to some mean, plus a length penalty. Minimizing that functional by **gradient flow** (level set method) can segment a defect even without clear edges, using intensity homogeneity. These methods involve calculus of variations, functionals, and **partial differential equations (PDEs)** – definitely graduate-level math. Snakes and level sets are powerful because they naturally impose smoothing (robust to noisy edges) and can handle complex shapes (level sets can split/merge to capture multiple blobs or branching scratches).

Geodesic Active Contours: A variant where the evolving contour is treated like a curve moving in a Riemannian metric derived from the image (often $g(x,y)=1/(1+|\nabla I|^2)$ so that it moves slower at strong edges). The PDE resembles a curve shortening flow weighted by image features. This is grounded in differential geometry and mean-curvature flow concepts.

- **Level Set Method:** Rather than explicitly parameterize a curve, one embeds it as the zero level of a higher-dimensional function $\phi(x,y)$ and evolves ϕ by a PDE. For example, a mean-curvature flow with an advection term from image gradient can capture edges. The level set formulation seamlessly handles topological changes (important if the defect segmentation has multiple regions). Chan–Vese can be implemented in level sets: $\partial \phi/\partial t = \delta(\phi)\left[\nu + \lambda_1(I-c_1)^2 - \lambda_2(I-c_2)^2\right]$ (where c_1,c_2 are average intensities inside/outside the zero level and ν is a length penalty coefficient). The mathematics involves solving this PDE iteratively to steady-state.

Noise and Artifact Robustness: Variational methods include regularization inherently (the length term discourages fitting to spurious noisy boundaries). They can incorporate *edge-stopping terms* (so they lock onto real edges) and *region-based terms* (so they won't get distracted by minor intensity fluctuations). They can also enforce that a scratch region is thin/long by adding shape constraints or additional penalties for area or curvature. These models are computationally heavy (solving PDEs iteratively or large linear systems), but they yield a very **clean segmentation** of defects when properly tuned.

Graph Cuts (Energy Minimization): Although graph cut is discrete, it fits here conceptually as it also minimizes an energy. One constructs an energy $E = E_{\rm data} + E_{\rm smooth}$ defined on pixel labels (foreground=defect or background). $E_{\rm data}$ might assign a cost for a pixel being labeled defect based on its intensity (e.g. deviation from background intensity), and $E_{\rm smooth}$ imposes a penalty for neighboring pixels having different labels (promoting continuity) [55] [56]. This can model that scratches/digs are contiguous regions that stand out in intensity. The energy is minimized by constructing a flow network and finding a **minimum cut**, thanks to the max-flow/min-cut theorem [57]. The min-cut globally

minimizes the energy, giving an optimal segmentation (for binary labels) with respect to that cost function [58] [59]. For example, one could set up $E_{\rm data}$ via intensity histograms (bright/dark distribution of defects vs background) and $E_{\rm smooth}$ proportional to gradient (so cutting along strong image gradients is cheaper, aligning the segmentation boundary to edges) [56]. The result is a segmented map of scratches and digs. Graph cut methods are **robust to noise** because they consider global energy – they won't select an isolated noisy pixel as "defect" if that dramatically increases boundary length cost. They also naturally fill gaps (due to the smoothness term) unless evidence suggests otherwise, which addresses discretization gaps.

In summary, variational and PDE-based methods bring heavy mathematical artillery: **calculus of variations, Euler-Lagrange equations, level set PDEs, and combinatorial optimization**. They are often at the level of research papers (indeed Chan–Vese, Mumford–Shah, etc., are classic PhD-level topics). These approaches can be tailored to be extremely robust against noise, at the cost of solving complex equations. They are suitable when one needs an automated, principled segmentation of defects rather than just edge highlighting.

9. Anisotropic Diffusion and Edge-Preserving Smoothing

Before detection, applying **edge-preserving noise reduction** is often beneficial. *Anisotropic diffusion* (Perona–Malik) is a PDE that iteratively smooths the image while **avoiding smoothing across edges** [60] [61]. The diffusion coefficient $D(x,y)$ is made a function of the local gradient: high (near 1) in uniform regions (to blur noise) and near 0 at strong edges (to keep edges sharp) [62]. The PDE: $\partial I/\partial t = \nabla \cdot (D(x,y)\,\nabla I)$ is nonlinear (because D depends on I) and is solved over a few iterations. This process reduces random noise and small fluctuations but **enhances edge contrast** (by not washing it out) [60]. By preserving line structures, it prepares the image such that subsequent edge or line detection is more robust.

A more advanced variant is **coherence-enhancing diffusion** [32]: here D is a tensor (matrix) aligned with the local structure (using the structure tensor from Section 4). It diffuses *along* a scratch (to strengthen its continuity) but not across it (so it doesn't blur its contrast) [32] [33]. This literally treats line structures as if they were "vessels" or "flows" in a physical system, letting diffusion act like flowing along the vessel. After such diffusion, scratches become smoother, more continuous, and noise within digs is smoothed, making threshold or gradient-based detection far more reliable.

Another PDE-based filter is the **shock filter**, which does essentially the opposite of diffusion at edges: it sharpens edges by increasing gradient where a contrast exists (creating a shock discontinuity). Shock filters (Osher-Rudin) use a sign of second derivative term to steepen intensity transitions. This can make faint scratches more pronounced by *nonlinear wave propagation* in the intensity profile.

While these techniques do not directly "detect" features, they **prepare and enhance** the image for detection. They involve solving PDEs or iterative equations – definitely advanced mathematical processes (connected to diffusion equations in physics and nonlinear dynamics). Using them means your subsequent segmentation (with any method above) is more robust to noise and aliasing. For example, anisotropic diffusion can remove compression artifacts or sensor pixel noise on the fiber image while preserving the scratch/dig boundaries, which are crucial for accurate detection.

10. Low-Rank + Sparse Matrix Decomposition (Robust PCA)

A fiber end-face image can be thought of as mostly smooth background (perhaps illumination gradient or fiber core reflection) plus localized defects. This lends itself to a **model**: image matrix $X = L + S$ where L is a *low-rank matrix* (the smooth, slowly-varying background) and S is a *sparse matrix* of anomalies (scratches and digs occupy relatively few pixels) [63] [64]. **Robust Principal Component Analysis (RPCA)** is a modern linear algebra technique to recover L and S given X. One formulation is solving:

$$\min_{L,S} \ \|L\|_* + \lambda \|S\|_1 \quad \text{s.t. } L + S = X,$$

where $|\cdot|_*$ is the nuclear norm (sum of singular values, promoting low rank) and $|\cdot|_1$ is the sum of absolute values (promoting sparsity) [63]. Despite this being a convex optimization in many variables, under certain conditions it can exactly recover the background and the sparse outliers [64] [65]. In our context, the output "sparse" component S would ideally contain bright/dark pixels corresponding to scratches and digs (and maybe some noise, but mostly the defects if they are sufficiently different from the smooth background).

This method is **highly computational** (it involves iterative solvers like Augmented Lagrange Multipliers or singular value decompositions in each step) and is rooted in advanced matrix theory and optimization. It's "PhD-level" in that it was developed in recent academic research. However, conceptually it is a powerful way to **separate structured background from anomalies**. It does not require choosing thresholds; it finds the decomposition by exploiting global data patterns (low-rank means there is redundancy in the image intensities – e.g., large uniform areas or gradients can be represented by a few principal components).

For example, suppose the fiber end-face has an overall illumination gradient or concentric pattern, and only small scratches and digs break that pattern. L (low-rank) will model the smooth illumination and perhaps slight lensing effects, while S flags the pixel regions that don't fit (the scratches/digs which "disturb the low-rank property" of the image [63]). Indeed, RPCA has been used in applications like **background subtraction** in video (treat background as low-rank across frames, moving objects as sparse) [65] – by analogy, the fiber background is "static" low-rank, and defects are "moving objects" sparsely distributed. Once S is obtained, one can post-process it: e.g., apply a small morphological dilation to connect neighboring sparse points and thus outline each defect region.

Noise Robustness: RPCA is inherently robust to moderate noise – the optimization can be modified to allow a small dense noise matrix too. It shines especially when defects are outliers in an otherwise smooth image. It might struggle if the background isn't low-rank (e.g., very textured surfaces), but for an optical fiber endface (usually mostly clean apart from defects) it's a fair assumption. Since it's a global method, it's not fooled by local pixel noise the way a derivative might be; it's looking for overall minimal-rank structure.

In summary, robust PCA provides a **matrix decomposition approach** to anomaly detection. It transforms the defect-finding task into a problem of linear algebra and convex optimization, with solutions drawing on eigenvalues (for SVD in low-rank) and L1 minimization techniques. It's quite far from classical edge detection, but it can be extremely effective for segmentation of scratches/digs when applicable.

11. Frequency-Domain and Multiresolution Analysis

Fourier Analysis: A scratch being a line is an image feature that has orientation – in the Fourier frequency domain, line patterns correspond to localized high-frequency content oriented perpendicular to the line. One could take the 2D Fourier transform of the image and look for an excess of energy along certain directions in frequency space. For instance, a perfectly straight horizontal scratch would produce a stripe of high-frequency coefficients aligned horizontally in the Fourier magnitude spectrum. By analyzing the spectrum (perhaps projecting it polar-wise), one might detect dominant orientations (this is essentially related to the Radon transform as well). However, for most practical purposes, Fourier global analysis is too coarse (scratches are local features, and Fourier doesn't localize spatial position). A bandpass filter could isolate features of a certain size (removing very low frequencies removes smooth background, removing very high frequencies removes noise).

Wavelet and Curvelet Transforms: Wavelets provide a multi-scale representation of the image. Unlike Fourier's infinite sinusoidal waves, wavelets are localized wave-like basis functions. Applying a wavelet transform (e.g. a dyadic wavelet decomposition) separates the image into sub-bands of different scales and orientations (for certain wavelet families). Edges and lines often appear prominently in specific wavelet sub-bands (e.g., horizontal vs vertical vs diagonal detail coefficients). One could detect scratches by thresholding wavelet coefficients – large coefficients in a given orientation band imply an edge/line at that orientation and location. This approach is related to **sparse representation**: a line is a singular feature that wavelets can represent with a few big coefficients. Denoising by wavelet thresholding (set small coefficients to zero) can be done first to remove noise, then invert the transform to get a cleaned image where lines are sharp.

More specialized are **ridgelet and curvelet transforms**. The **curvelet transform** is an advanced multiresolution transform designed to represent images with curves and edges much more efficiently than wavelets. Curvelets have anisotropic elements (very elongated shapes at fine scales) and are known to approximate curve-like edges with fewer coefficients. A line or scratch, being essentially a straight or gently curved edge, is exactly the kind of feature curvelets excel at. Indeed, curvelet transform was shown to handle curve singularities better (while wavelets handle point singularities) [66]. In practical terms, one could perform a curvelet transform and simply look for the highest-magnitude curvelet coefficients – they will correspond to the significant line structures (the math behind it is that curvelets perform something akin to a localized Radon transform combined with wavelet smoothing). Researchers have proposed edge detection via thresholding curvelet coefficients and reconstructing an image of edges [67]. Similarly, **ridgelets** (a precursor to curvelets) directly represent linear singularities via a Radon transform followed by a wavelet transform; they could theoretically identify a single straight scratch in one coefficient.

Using these transforms involves heavy math – integral transforms, functional analysis, and often specialized inversion algorithms. They provide a **different domain to analyze the image's content**, where noise and signal can be separated by frequency content or sparsity. For noise robustness, frequency filtering (like low-pass to remove noise, or high-pass to isolate edges) is straightforward. For example, applying a high-pass filter (like subtracting a blurred version of the image) will emphasize fine details including scratches/digs – a simpler frequency-domain approach related to the Laplacian filtering mentioned before.

Phase Congruency: This is a technique where edges are identified by the *phase alignment* of Fourier components rather than their magnitude. Points in the image where many frequency components have the same phase are likely features (edges or lines). Phase congruency is illumination-invariant (doesn't matter how bright/dark the line is, if it produces a phase alignment it's detected) – a highly robust measure. It

requires computing local Fourier or wavelet transforms and analyzing phase, which is mathematically advanced. But it can detect subtle scratches that vary in intensity because it looks for structural presence rather than gradient magnitude.

In summary, frequency and multiresolution methods give **analytical tools** for defect detection: - They allow filtering (to remove noise or background frequencies). - They allow detection by looking at coefficient patterns (or significant coefficients). - They are grounded in signal processing theory (Fourier analysis, basis decompositions) and require linear algebra (e.g., computing transforms via matrix multiplications or FFT) and sometimes advanced function space theory (as with curvelets). They complement spatial-domain analyses and can be especially useful if the defect has a known frequency signature or if one needs to *denoise* the image as a pre-processing step.

In conclusion, detecting scratches and digs on a fiber end-face involves a rich toolbox of mathematical methods:

- *Calculus-based:* gradients and Laplacians for basic edge and blob detection.
- *Linear algebra:* eigen-decomposition of Hessians or structure tensors to quantify line/blob nature, low-rank approximations to isolate anomalies.
- *Integral transforms:* Hough/Radon to aggregate evidence of lines; Fourier/wavelet to filter and detect patterns.
- *PDEs and variational calculus:* to iteratively evolve solutions that segment defects while imposing smoothness and stability.
- *Discrete math and algebraic methods:* morphological operations and graph cuts to solidify detections into clean segments.

Each method addresses the problem from a different angle – some focus on local differential structure, others on global geometric alignment, others on optimizing an energy or decomposition. In practice, these can be combined (e.g., use anisotropic diffusion to denoise, then LoG to find blobs and structure tensor to find lines, then refine with an active contour). All these approaches are **manually calculable or simulatable** with enough computational resources and represent the state-of-the-art mathematical techniques (outside of machine learning) for image defect detection. By employing these, one can robustly locate and segment scratches (linear features) and digs (blob features) even under varying intensity contrasts and noise conditions, directly on the pixel intensity grid of the fiber's image.

Sources:

- Edge detection via gradient, Canny method [1].
- Hough transform for line detection and its equivalence to Radon transform [44] [35].
- Differential geometric definition of ridges (lines) via Hessian eigenvalues [15].
- Laplacian of Gaussian and scale-space blob detection [4] [6].
- Structure tensor eigen-analysis for orientation coherence [27] [28].
- Frangi vesselness (Hessian eigenvector method for ridges) [18].
- Graph cut energy segmentation (global optimum via min-cut) [58] [56].
- Anisotropic diffusion for noise reduction with edge preservation [60] [32].
- Robust PCA model separating low-rank background and sparse defects [63] [64].

---

[1] Canny edge detector - Wikipedia
https://en.wikipedia.org/wiki/Canny_edge_detector

[2] [3] [4] [5] [6] [7] [8] [9] [10] [11] [20] [21] [22] [23] Blob detection - Wikipedia
https://en.wikipedia.org/wiki/Blob_detection

[12] [13] [14] [15] [16] [17] Ridge detection - Wikipedia
https://en.wikipedia.org/wiki/Ridge_detection

[18] Hessian based Frangi Vesselness filter - File Exchange - MATLAB Central
https://www.mathworks.com/matlabcentral/fileexchange/24409-hessian-based-frangi-vesselness-filter

[19] [24] [25] [26] [27] [28] [29] [30] [31] Cris' Image Analysis Blog | The Hessian and the Structure Tensor
https://www.crisluengo.net/archives/1132/

[32] [33] [60] [61] [62] Anisotropic diffusion - Wikipedia
https://en.wikipedia.org/wiki/Anisotropic_diffusion

[34] [35] [36] [37] [38] [39] [40] [41] [42] [43] [44] [45] Hough transform - Wikipedia
https://en.wikipedia.org/wiki/Hough_transform

[46] Top Hat and Black Hat Transform Using OpenCV Python
https://www.tutorialspoint.com/top-hat-and-black-hat-transform-using-opencv-python

[47] Removing small objects in grayscale images with a top hat filter
https://scikit-image.org/docs/0.25.x/auto_examples/filters/plot_tophat.html

[48] TopHatTransform - Wolfram Language Documentation
https://reference.wolfram.com/language/ref/TopHatTransform.html

[49] [50] [51] [52] [53] [54] Active contour model - Wikipedia
https://en.wikipedia.org/wiki/Active_contour_model

[55] [56] [57] [58] [59] Graph cuts in computer vision - Wikipedia
https://en.wikipedia.org/wiki/Graph_cuts_in_computer_vision

[63] [64] [65] How to use Robust PCA for image inpainting? - Signal Processing Stack Exchange
https://dsp.stackexchange.com/questions/16234/how-to-use-robust-pca-for-image-inpainting

[66] SAR image edge detection using curvelet transform - ProQuest
https://www.proquest.com/scholarly-journals/sar-image-edge-detection-using-curvelet-transform/docview/1616434818/se-2

[67] (PDF) Edge detection in microscopy images using curvelets
https://www.researchgate.net/publication/24174240_Edge_detection_in_microscopy_images_using_curvelets