ChatGPT

Advanced Mathematical Methods for Detecting Concentric Circles in Intensity Data

In this technical report, we explore a wide range of **advanced mathematical techniques** (excluding machine learning) for detecting two concentric circles within a given pixel intensity matrix. We organize the methods by branches of mathematics and modeling approaches, providing detailed explanations, relevant equations, and suggestions for how each method could be implemented or simulated. Key considerations such as noise robustness, discretization artifacts, and iterative refinement are also addressed.

1. Analytical Geometry and Parameter-Space Methods

1.1 Classical Geometry and Direct Fitting

Circle Equation and Direct Solutions: A circle in Cartesian coordinates satisfies an equation $(x-h)^2 + (y-k)^2 = R^2$, where (h, k) is the center and R the radius. For two *concentric* circles, the centers (h, k) are the same but with two different radii R_1, R_2 . If the intensity matrix reveals circular edge features, one could analytically solve for the circle parameters by selecting points on these edges. In an ideal noise-free case, three non-collinear points are sufficient to determine a circle exactly via solving the system of equations for h, k, R . This can be extended to two concentric circles by first determining the common center (for example, by intersecting perpendicular bisectors of chord segments from each ring). In practice with many points, one can set up an overdetermined system and solve in a least-squares sense (see §3.1).

Least-Squares Circle Fitting: Expanding the circle equation gives a quadratic form $x^2 + y^2 + Ax + By + C = 0$. Gathering many edge points (x_i, y_i) (for example, from a thresholded gradient image) yields a linear system $B\,u \approx 0$ for the coefficient vector $u = (A, B, C)$ [1] [2]. Enforcing a constraint (e.g. fix one parameter or norm of u) leads to a least-squares solution. This algebraic fit can be found via linear algebra techniques (solvable by finding the singular vector of B corresponding to the smallest singular value [3]). From the solved coefficients, one recovers the center and radius by formulas (for example, $h = -A/2,\ k = -B/2,\ R = \sqrt{(A^2 + B^2)/4 - C}$). This direct approach provides an initial estimate, which can be refined iteratively (see §7). It is an **analytical** approach leveraging geometry and linear algebra, but it may be sensitive to noise and outliers if used in isolation.

1.2 Hough and Radon Transform Techniques

Hough Transform (Parameter Space Voting): The Hough transform is a classic parameter-space method for detecting geometric shapes by an accumulator voting process [4]. For circle detection, each edge pixel (at coordinates i, j) votes for all circles that could pass through that point. A circle can be parameterized by its center (a, b) and radius r , so the parameter space is 3D [4]. If the radius is known (or fixed), the accumulator is 2D for the center; if unknown, a 3D accumulator (a, b, r) is used [5]. Conceptually, for each potential radius r , one can draw a circle of radius r around the edge point and increment votes in the accumulator at those (a, b) coordinates. Peaks in the accumulator indicate likely circle centers and radii that correspond to actual circles in the image [6]. For two concentric circles, one would expect a common (a, b) with two distinct radius peaks. The Hough transform is robust to moderate noise since voting integrates evidence from many points, and missing or spurious edge points tend to be averaged out. It handles discretization by accumulating fractional votes (or using interpolation) to achieve sub-pixel precision in the parameters. However, it can be computationally heavy (due to a 3D parameter search) and may require smoothing or thresholding to identify the strongest peaks corresponding to the two circles.

Generalized Radon Transform: Mathematically, the Hough transform can be viewed as a discretized form of the Radon transform for curves [7]. The Radon transform integrates a function (the image intensity or an edge map) over a family of curves or surfaces. For circle detection, a **circular Radon transform** can be defined, which integrates image values along the circumference of a circle of a given radius centered at (a, b). Formally, one could define $R(a, b, r) = \oint_{(x-a)^2+(y-b)^2=r^2} I(x, y)\, ds$, where ds is an element of arc length. Peaks of $R(a, b, r)$ occur when a circle of radius r and center (a, b) matches a high-intensity or edge pattern in the image. This provides a more analytical (integral) approach to circle detection. The Radon approach is **more mathematically sound** (treating the problem in continuous terms) and can be more **accurate but slower** than the brute-force Hough voting [8]. It is inherently robust to noise because integration along the circle averages out pixel-level fluctuations. In practice, implementing a circular Radon transform might involve testing a range of radii and convolving the image with circular kernels or using FFT-based correlation in polar coordinates. The Radon transform provides a framework to detect circles by treating it as a problem of finding maximal alignment of intensity along closed curves.

Polar Coordinate Mapping: As a related analytical trick, if the approximate center of the concentric circles is known (or guessed), one can transform the image to polar coordinates centered at that point. In polar coordinates (r, θ), circles centered at the origin appear as horizontal lines (at fixed radius r). By summing or averaging intensities over angle θ for each radius, one obtains a radial intensity profile. Concentric ring patterns manifest as peaks in this radial profile. This method is essentially a **projection** of the image data onto a one-dimensional function of radius, simplifying detection of multiple ring radii. It can be implemented numerically by interpolation of the image onto a polar grid and then analyzing the 1D function $I_{\text{avg}}(r) = \frac{1}{2\pi} \int_0^{2\pi} I(r, \theta)\, d\theta$. Sharp changes or local maxima in $I_{\text{avg}}(r)$ would correspond to the bright concentric circles. This approach reduces noise by angular averaging and is straightforward to implement, though it assumes a known or approximate center (which could be found by an initial Hough search for one circle or by image moments).

2. Calculus of Variations and PDE-Based Methods

2.1 Energy Minimization and Active Contours (Snakes)

Variational Formulation: Calculus of variations can be used to detect shapes by defining an energy functional that is minimized when a contour aligns with desired image features (such as edges or intensity differences). The *active contour model* or "snake" is a prime example [9]. One defines an energy $E[C(s)]$ for a parametric curve $C(s) = (x(s), y(s))$ (the candidate circle in this case) which typically has internal regularization terms and external image-based terms. For example, a snake energy can be written as:

$$E = \int_0^1 \left[\alpha |\mathbf{C}'(s)|^2 + \beta |\mathbf{C}''(s)|^2\right] ds \; + \; \gamma \int_0^1 F_{\text{ext}}(C(s))\, ds,$$

where the first terms (with weights α, β) control the contour's length or curvature (regularization to prefer smooth, compact shapes) and the external force $F_{\text{ext}}(x, y)$ is derived from the image (for example, $-|\nabla I(x, y)|^2$ or a balloon force pushing outward) to attract the contour to edges or specific intensity regions [9] . Detecting a circle would involve initializing the contour and allowing it to deform under these forces. The Euler-Lagrange equations for this functional yield a force-balance condition: the contour moves in the normal direction proportional to a combination of **curvature** (internal force making it smoother or shorter) and **image gradient forces** (external forces from intensity) until equilibrium is reached at an edge. In practice, this leads to an *iterative solver*: one discretizes the contour and at each iteration updates it according to $\partial \mathbf{C}/\partial t = -\frac{\delta E}{\delta \mathbf{C}}$, which is essentially a **gradient descent** on the energy.

Circle-Specific Active Contour: For concentric circles, one could incorporate prior knowledge of circular shape into the model. For instance, one can restrict the snake to remain circular (perhaps by parameterizing $C(s)$ as a circle with variable center and radius) and then derive simpler evolution equations for those parameters. Alternatively, one can use a more general deformable contour but include a shape prior that penalizes deviations from circularity [10] . The snake framework is powerful: it is **adaptive and can handle noise** by virtue of the smoothing terms, and it can even detect *partial* or *ill-defined* circles by bridging gaps (snakes can find illusory contours [11]). However, a traditional parametric snake requires an initial guess fairly close to the actual circle. For our problem, one might initialize two circular loops and let them evolve to lock onto the two ring edges. Alternatively, using a multi-scale or automated initialization (like starting with a small circle at the center and expanding) can help capture the rings.

2.2 Level Set Methods and Evolving Curves via PDE

Geometric (Level Set) Active Contours: Geometric active contours address some limitations of parametric snakes by representing the contour implicitly as the zero level set of a higher-dimensional function $\phi(x, y, t)$. One then evolves ϕ under a partial differential equation so that its zero-level (the moving front) propagates toward object boundaries. A popular formulation is the *geodesic active contour* (Caselles et al.), which treats the evolving contour as a geodesic (shortest path) in a Riemannian space defined by the image content [12] . The level set evolution equation for a geodesic active contour can be written as:

$$\frac{\partial \phi}{\partial t} = \mu\, \kappa(\phi)\, g(I)\, |\nabla \phi| \; + \; \nu\, g(I)\, |\nabla \phi| \; + \; \lambda\, \nabla g(I) \cdot \nabla \phi,$$

where κ is the curvature of the level set, $g(I)$ is an edge stopping function (e.g. $g = 1/(1 + |\nabla I|^2)$ that is small near edges), and μ, ν, λ are parameters. This rather complex PDE essentially causes the contour to move (normal velocity proportional to curvature and to edge attraction) and to **stop on edges** of the image [12] . The level set framework can handle **topological changes**, meaning a single evolving contour can split or merge as needed [12] . This is useful in case the rings are not completely separate or if one wants to detect multiple objects without knowing their number in advance. For two concentric circles, one strategy could be to run a level set segmentation that *bipartitions* the image into three regions: inside the inner circle, the annulus between the two circles, and outside the outer circle. This could be achieved with a **multiphase level set** (using two level set functions to represent two boundaries) or by sequentially segmenting: first detect the outer boundary, then the inner.

Region-Based Variational Methods: Another calculus of variations approach is to use **region-based energies** like the Chan–Vese model (a variant of the Mumford–Shah segmentation). Instead of relying on gradient edges, Chan–Vese defines an energy that favors fitting regions of nearly constant intensity inside vs. outside a closed contour [13] [14]. In our scenario, if the two concentric circles correspond to bright rings on a darker background (or vice versa), one could set up a multiphase Chan–Vese functional to segment an inner dark region, a bright annulus, and an outer dark region. The Euler–Lagrange equations from this functional again yield level set update rules (essentially forcing the level sets to move to minimize the variance of intensity within each region). Such methods are robust to noise because they aggregate information over regions (not just rely on local gradients). They also handle **discretization artifacts** by implicitly smoothing region boundaries via a length penalty term in the energy. The trade-off is that one must assume a model of intensities (e.g., approximately constant or known means in each region), but if applicable, this yields a powerful way to detect concentric structures as distinct regions.

Simulation Considerations: PDE-based methods require careful numerical simulation. For example, one can simulate the level set evolution by finite differences: initialize $\phi(x,y,0)$ as a signed distance function around an initial guess of the circles (or a small circle), then iterate $\phi^{n+1} = \phi^n + \Delta t\, F(\phi^n)$ with $F(\phi)$ the right-hand side of the chosen PDE. The choice of time step Δt and regularization (like reinitializing ϕ as a distance function periodically) is critical for stability [15]. For variational snakes, one could simulate the gradient descent on the contour by moving each point a small step according to the snake equations at each iteration. Modern implementations often use **fast marching or fast sweeping methods** (for Eikonal-type equations) to simulate wavefront propagation efficiently [16]. In fact, some segmentation can be viewed in terms of **wave propagation**: a front expanding from a seed point that slows down at image edges, analogous to a physical wave moving through a medium with varying speed [17] [18]. This *physical analogy* is further discussed in §6.

3. Linear Algebra and Optimization Techniques

3.1 Eigenvalue Problems and Algebraic Circle Fitting

Algebraic Circle Fit via Matrix Equations: As introduced in §1.1, fitting a circle to points can be formulated as a linear algebra problem. Given many candidate edge points believed to lie on the circles, one forms a design matrix B where each row is $[x_i^2 + y_i^2,\ x_i,\ y_i,\ 1]$ corresponding to the coefficients in $x^2 + y^2 + Ax + By + C = 0$. We seek a vector $u = [A,B,C,D]^T$ (with $D = 1$ if we enforce the coefficient of $x^2 + y^2$ to be 1 for uniqueness) that satisfies $Bu \approx 0$. In a least-squares sense this becomes $Bu = \mathrm{minimize}$ with a constraint on u [2]. The solution can be obtained via the **singular value decomposition (SVD)** or equivalently by solving the normal equations. In fact, the optimal algebraic fit (minimizing $\sum (Ax_i + By_i + C + D(x_i^2 + y_i^2))^2$) is given by the **eigenvector** (right singular vector) of B corresponding to the smallest singular value [3]. This is a straightforward linear algebraic method to get an initial circle estimate. It is fast and can be done in closed-form (e.g., the famous Pratt method or Taubin method for circle fitting uses such algebraic constraints).

For two concentric circles, one approach is to first determine the common center using all edge points from both circles. This can be done by a slight modification: allow two different radii but a single center (h,k) . One can set up equations $(x_i - h)^2 + (y_i - k)^2 = R_{c(i)}^2$ where $c(i)$ indicates which circle (inner or outer) point i belongs to. If that classification is unknown, one could first fit one circle and then the other. Alternatively, treat h,k as unknowns and solve $\nabla_{h,k} \sum_i [(x_i - h)^2 + (y_i - k)^2 - R^2]^2 = 0$. Expanding leads to linear conditions relating h,k to the averages of points. In simpler terms, the optimal concentric center could be found by minimizing the variance of radii: one can guess a center and compute all distances $d_i = \sqrt{(x_i - h)^2 + (y_i - k)^2}$, then the best center minimizes the spread of these distances (bimodal in case of two distinct radii). This is a 2D nonlinear least-squares problem that can be solved via **gradient-based optimization** or grid search. After finding h, k , the radii R_1, R_2 can be estimated as the average distances of each cluster of points (or via another linear fit now with center fixed).

Total Least Squares and Eigen-decomposition: The above fitting is an example of **total least squares**, since both x, y contribute to error. The eigen-decomposition approach inherently handles that by finding the smallest singular value solution. In matrix terms, if $Bu = 0$ has no exact solution, we solve $Bu = r$ for minimal $\|r\|$ under a normalization of u [19]. This can be cast as an eigenvalue problem: solve $(B^T B)u = \lambda_{\min} u$. The solution u (with $a = 1$ in the notation of [1]) gives the algebraic circle. While this algebraic solution is not exactly the geometric best-fit (it minimizes algebraic distance, not the true orthogonal distance to the circle), it is often used as an initial guess [20] [21]. One can then apply a **nonlinear refinement** (see §3.2) to minimize geometric distance.

3.2 Optimization Frameworks and Iterative Solvers

Nonlinear Least Squares Optimization: Once an initial circle (or two circles) are estimated, one can frame the circle detection as an optimization problem: find parameters $\theta = (h, k, R_1, R_2)$ that minimize some cost function. A suitable cost is the sum of squared distances from each candidate edge point to the nearest of the two circles (for example) – this directly measures geometric error. The cost might be:

$$J(h, k, R_1, R_2) = \sum_{i \in \text{edges}} \min\{(\sqrt{(x_i - h)^2 + (y_i - k)^2} - R_1)^2,\ (\sqrt{(x_i - h)^2 + (y_i - k)^2} - R_2)^2\}.$$

This is a **non-convex** optimization problem, but it can be tackled with iterative solvers like **Gauss–Newton** or **Levenberg–Marquardt**, which linearize the residuals at each step [22] [23]. Gauss–Newton requires computing the Jacobian of residuals (distances) with respect to the parameters and solving a 4×4 linear system per iteration. One must also be careful to ensure points are associated with the correct circle; this could be handled by alternating optimization (assign points to nearest circle given current parameters, then update parameters). This resembles the **Expectation-Maximization** approach (assigning points to clusters – here two radii – and refining the cluster parameters).

Projection and Subspace Methods: Another linear algebra angle is to use **projection techniques**. For instance, one can project the image or edge data onto a basis of circular harmonics or radial basis functions. If the image contains bright rings, projecting onto circular harmonic functions (e.g., Bessel functions or Fourier–Bessel series which naturally represent circular patterns) might reveal modes corresponding to those rings. Similarly, constructing an image of the same size that contains an ideal ring (annulus) and using correlation (which is equivalent to a projection in function space) will yield a peak when the rings align in radius and position. This is essentially **template matching** in a linear algebra sense: treat the flattened image as a vector and compute the inner product with template vectors (one template for each possible radius ring at each location). While brute-force template matching is expensive, using the Fourier domain can accelerate it (the convolution theorem allows correlating a circular mask over the image efficiently). One can also use the eigen-images approach: compute the principal components of a set of basis images that are concentric rings; the given image projected into that basis can then be analyzed for significant components indicating rings.

Eigenfunction Decomposition: A more theoretical approach is to consider eigenfunctions of differential operators that are sensitive to circular shapes. For example, the Laplacian operator Δ on a 2D domain has eigenfunctions in polar coordinates that are Bessel functions times angular sinusoids. If the intensity pattern contains circles, it might have significant energy in eigenfunctions corresponding to those radial frequencies. In practice, one might perform a 2D Fourier transform: concentric rings in the spatial domain correspond to Bessel patterns in frequency domain. By examining the power in circularly symmetric frequency components (e.g., summing the Fourier magnitude on circles in frequency space), one can detect radial structures. This approach connects to the idea of using the **Fourier–Bessel transform** or circular Zernike polynomials (which form an orthogonal basis for images on a disk) to detect circular features.

Robust Solvers (RANSAC and M-Estimators): Real data often contain outliers (e.g., edges from other structures or noise). Robust linear algebra techniques like **RANSAC (Random Sample Consensus)** can be used to fit circles. RANSAC would randomly pick minimal subsets of points (three points define a circle), solve for the circle, then check how many points from the entire set lie near that circle (within a tolerance). By many iterations, it finds the circle model with the most support (inliers). This can be extended to finding two circles by finding one circle's inliers and then finding another among the remaining points. Similarly, within an optimization framework, one can use robust loss functions (like Huber loss or absolute loss) instead of squared error to reduce the influence of outliers – this falls under **M-estimation** in statistics. These techniques improve noise and artifact robustness by not letting a few bad points skew the solution.

4. Differential Geometry and Conformal Mapping Approaches

4.1 Curvature, Differential Geometry, and Shape Analysis

Curvature-Based Detection: In differential geometry, a circle in the plane is characterized by *constant curvature*. The curvature of a curve defined implicitly (like an edge in the image) can be computed from second derivatives. One approach is to first perform edge detection (e.g., Canny or Laplacian-of-Gaussian) to get curve pixels, then estimate the curvature along these edge curves. High-precision estimation might involve fitting local osculating circles to the curve or using derivatives of the curve's parametric representation. Segments of the edge with roughly constant curvature that persists over many degrees of arc are likely part of a circle. By grouping edge pixels with similar curvature and center of curvature, one can recognize circular arcs. Differential geometric descriptors (like curvature and also torsion in 3D, though here we are 2D) can thus be used to **classify an edge as circular** or not. In practice, one could slide a window along the edge points and fit a circle (via the methods of §3.1) to each small window; if the fit radius is stable and consistent, the entire set of points is a circle of that radius. The advantage is this uses **local differential properties** (less sensitive to global noise) and can distinguish circles from other shapes (e.g., an ellipse's curvature varies with angle, a circle's curvature is constant). This method may require good edge extraction and can be affected by discrete pixelation (necessitating sub-pixel smoothing of the edge curves).

Shape Space and Geodesics on Shape Manifolds: An abstract differential-geometric approach is to consider the space of all closed curves (each curve is a point on an infinite-dimensional manifold) and define a metric on this space. One can then define the distance between an arbitrary shape and the subset of circular shapes. The problem of finding the best-fitting circle to an observed shape can be viewed as projecting the observed shape onto the manifold of circles. Some works in shape analysis define metrics where geodesics on the shape manifold correspond to smooth deformations. While highly theoretical, one could imagine constructing a flow that deforms an arbitrary closed contour toward a perfectly circular contour while fitting the data. This relates to gradient flows on shape manifolds and can involve solving differential equations in the space of Fourier descriptors or other shape coefficients. Such methods ensure the result is exactly concentric circles (if that is the constrained manifold) and can incorporate differential geometry concepts like invariances (e.g., under rotation or scaling).

Intrinsic Differential Operators: Another angle is using differential operators on the image surface itself. For example, treat the intensity matrix as defining a surface $z = I(x, y)$ in $\mathbb{R}^3$. Concentric rings might appear as closed level sets of this surface (if the rings are uniform intensity) or as ridges/valleys. One can compute the **mean curvature** or Gaussian curvature of the intensity surface; concentric rings might appear as specific patterns (like concentric ridge lines). One could also use **differential invariants** such as the Hessian matrix eigenvalues of the image (which indicate ridge-like structures). Filtering the image by eigen-directions (e.g., the second derivative operator known as the Laplacian-of-Gaussian will highlight circular edges as closed zero-crossings). In summary, differential geometry provides tools like curvature flows and invariant operators that can be applied either to the contour or the intensity function to extract circular features.

4.2 Conformal Maps and Complex Analysis

Circle-Line Duality via Inversion: In the complex plane, a circle not passing through the origin can be transformed into a line by an **inversion** map (a type of conformal map). For instance, the transformation $w = 1/(z - z_0)$ (where z_0 is a suitably chosen point related to the circle's center) will send a circle in the z -plane to a line in the w -plane. More generally, Möbius transformations (inversions and combinations thereof) map circles to circles or lines [24] . This suggests a strategy: apply a conformal map to the image such that concentric circles become parallel lines or concentric circles of a fixed, simpler form. If one could perfectly invert about the center of the concentric circles, those circles would map to radial lines or something trivial, but the center is unknown a priori. However, one could attempt a sweep of possible centers (like scanning a transformation) to maximize some line-detection criterion in the transformed domain. While this is not a standard approach in image processing, it’s a theoretically intriguing method: convert the circle detection problem into a **line detection problem** via a coordinate transformation. Line detection is simpler (using Radon or Hough for lines). After detection in the transformed domain, one can invert the coordinates back to get circles in the original domain. Conformal mapping ensures that the local angles are preserved and shapes are smoothly transformed, which can help separate the circles from other features if chosen wisely.

Complex Moments and Polynomial Fitting: In complex analysis, a circle $|z - a| = R$ can be represented as the locus of solutions to a quadratic equation $z\bar{z} - \bar{a}z - a\bar{z} + |a|^2 - R^2 = 0$. One could try to identify these coefficients by using complex moments of the image. Complex moments $M_{pq} = \iint x^p y^q I(x, y)\,dxdy$ (or complex versions $\iint z^p \bar{z}^q I(x, y)\,dxdy$) contain information about symmetrical structures. For a perfect ring shape, certain moments might be distinctive. For instance, the presence of a concentric ring might be revealed by peaks in the magnitude of the Fourier transform at frequencies corresponding to that radius. Alternatively, one could construct a *conformal map of the plane to itself* that “flattens” one circle to, say, the unit circle and observe the image under that map for evidence of the second circle. This gets into advanced territory: effectively solving for a conformal map that simplifies the shape is akin to solving Laplace’s equation (since conformal maps are harmonic). While not a practical algorithm, such theoretical techniques underscore that circles have special properties under transformations.

Conformal Invariants: One may also consider invariants under Möbius transformations – for example, the cross-ratio of four points on a circle is real. By picking four edge points that are suspected to lie on a common circle, one can compute their cross-ratio. If the cross-ratio is (approximately) real, this is a necessary condition for concyclicity. This provides a test to group points into circular subsets using algebraic geometry methods.

5. Topological Methods (Algebraic Topology)

5.1 Persistent Homology for Detecting Loops

Topological Features of Rings: Two concentric circles represent a topological signature of two **1-dimensional holes** (loops) in the image intensity structure. Algebraic topology offers tools to detect such holes in data. In particular, **persistent homology** analyzes data across multiple scales (e.g., intensity thresholds) to find features like connected components, tunnels, and voids that persist [14]. For a 2D image, the first Betti number b_1 counts the number of distinct loops (holes) in a binary segmentation of the image. If the concentric circles are bright rings on a darker background, one can imagine thresholding the image at a high intensity: initially, no pixels (above threshold) are present; as the threshold lowers, pixels in the bright rings appear, forming two closed loops of high intensity. Over a range of threshold values, these loops will appear and eventually fill in. Persistent homology tracks these loops: each ring will correspond to a class in first homology that *births* when the ring appears and *dies* when it merges or fills. The longer the persistence (birth-to-death in terms of threshold), the more significant the loop [14]. In practice, one would construct a *filtration* of the image (e.g., take sublevel sets $\{(x, y) : I(x, y) \geq T\}$ for T from high to low) and compute homology at each step. The output is a **persistence diagram** or barcode, where a long bar in H_1 indicates a robust circular feature [25]. Detecting exactly two long-lived H_1 features would strongly suggest two concentric loops.

This method is highly robust to noise – small spurious edges may form loops, but they typically exist over a narrow threshold range and thus have low persistence (short bars) [26]. The topological approach is also indifferent to exact geometry or discretization; it only cares about connectivity and holes, so even if the circles are slightly irregular or broken, as long as they form closed loops eventually, the homology will capture them. One challenge is that persistent homology provides an abstract descriptor rather than explicit circle parameters. However, one can get approximate locations by looking at the representative cycles of those homology classes (the algorithm can return a set of pixels forming the loop). Another topological invariant, the **Euler characteristic** ($\chi = b_0 - b_1$ for 2D), will decrease by 1 for each hole that appears. So measuring χ of thresholded images as the threshold changes will show drops when loops form. In summary, algebraic topology offers a noise-resistant way to confirm the existence of two loops (circles) in the data by purely topological means [25] [27].

5.2 Graph and Network Analogs

Graph-Cycle Detection: One can also model the image's high-intensity pixels as a graph (vertices are pixels, edges connect neighbors). Finding rings is equivalent to finding cycles in this graph. Algorithms in graph theory (which is related to topology) like finding the **fundamental cycle basis** could be applied. Essentially, after edge detection, one could have an undirected graph of edge segments; using algorithms for cycle detection (such as looking for back-edges in a depth-first search or using union-find for connected components with cycle detection) will identify closed loops. Two distinct loops would be discovered. However, distinguishing concentric loops (one surrounding the other) might require additional spatial reasoning. Algebraic topology via homology inherently accounts for concentric (nested) loops as separate generators of H_1 . In graph terms, one can compute the cycle space of the graph (the null space of its incidence matrix) to find independent cycles. Each independent cycle corresponds to a loop in the image.

This linear-algebraic graph approach will find loops even if they are not perfect circles (any closed chain of edges). Therefore, some geometric filtering would be needed afterward to verify they are circular (e.g., using curvature as in §4.1 or comparing radii).

Topological Constraints in Energy Models: Topology can also be enforced or utilized in variational models (as a more advanced note). One could add a term to an energy functional that penalizes or favors certain topology. For example, one might want an active contour that specifically yields two loops. While standard active contours can split, controlling the final number of loops is tricky. Some research includes *topology-preserving level sets* or conversely *topology-based energies*. For instance, a penalty on b_1 (number of loops) could be added to prefer a certain number of loops. This merges topology with optimization frameworks. Conversely, one could run a segmentation and then **post-check the topology**: ensure that the segmentation result has Euler characteristic = 1 (which for two loops means likely 3 regions, hence $b_1 = 2$, if the entire image outside counts as one connected component). If not, adjust parameters or initializations and try again.

6. Physics-Inspired Analogies and Models

6.1 Wavefront Propagation and Eikonal Analogies

Wave Propagation Analogy: We can draw an analogy between the image domain and a physical medium where wavefronts propagate. Suppose we have a wave emanating from a point (e.g. the center of the image). In a homogeneous medium, the wavefront would be a growing circle. If there are boundaries (e.g. high-intensity circle acting like an interface), it might partially reflect or change speed. One way to leverage this is the **fast marching method** (a numerical algorithm to solve the Eikonal equation $|\nabla T(x, y)| \, F(x, y) = 1$). If we define a speed function $F(x, y)$ that is high inside the rings and low at the ring edges (imagine the rings are "slow" regions or absorbers), then a wave initiated at the center will propagate outward and *slow down drastically at each ring*. By recording the arrival time $T(x, y)$ of the wave at each point, one would observe level sets of T that bunch up at the rings (since the wavefront stagnates there). Essentially, the concentric circles act like equipotential barriers. This method is analogous to the **Monge soil problem or shortest path**: finding circles where the wave spent extra time indicates those circles are obstacles (edges). In practice, one can simulate wave propagation by solving $T_t = F(x, y)|\nabla T|$ or by iteratively growing a region (using something like Dijkstra's algorithm on a grid, treating darker pixels or edges as high cost). Once the wave has passed, one can inspect the gradient of the arrival time ∇T or its Laplacian; sharp changes in T radially would correspond to hitting the ring. This approach is robust to moderate noise because waves naturally circumvent small irregularities (like diffraction around small obstacles) and fill gaps, somewhat analogous to how active contours can bridge gaps.

A related concept is **Huygens' principle**: every point on a wavefront acts as a source of secondary waves. If we invert this principle for detection, we could emit waves from all edge pixels and see interference or intensification at the center if those edges form concentric circles. This is more of a thought experiment – practically, performing a circular Hough is akin to that (each edge emits a "wave" of votes back toward possible centers).

Resonance and Spectral Methods: Another physics analogy is to consider a vibrating membrane or drum. The eigenmodes of a circular membrane are Bessel functions; a strong circular pattern in initial conditions might excite certain resonance modes. If one treats the intensity profile as an initial disturbance and performs a modal analysis (e.g., solve the wave equation $u_{tt} = c^2 \Delta u$ on the image with the intensity as initial displacement), the presence of concentric circle patterns might result in higher amplitude in modes that are circularly symmetric. By analyzing the spectrum of the vibration (frequencies of oscillation or shapes of mode), one might detect rings indirectly. This is a rather elaborate approach and would require solving PDEs or doing an SVD of the Laplacian operator applied to the image, but it ties into the idea of frequency-domain analysis for circular features mentioned in §3.2.

6.2 Diffusion Models and Field Analogies

Diffusion and Heat Flow: If we treat the intensity matrix as initial temperature distribution, running a **diffusion process** (heat equation $\partial_t u = \Delta u$) will smooth out noise over time. However, edges (like ring boundaries) diffuse slower because they represent large gradients initially. One can monitor how the image evolves under diffusion: areas inside a ring vs outside will start to blur together, but sharp edges delay this mixing. One could measure the **heat content** inside a suspected ring over time; a sudden change in diffusion rate might indicate when the heat starts leaking across a boundary, thus identifying the boundary location. Alternatively, one can use *anisotropic diffusion* (Perona–Malik) which is designed to diffuse within regions but not across strong edges. Starting with an initial noisy image, anisotropic diffusion will preserve the ring edges while smoothing noise within regions. After sufficient diffusion, thresholding or gradient detection becomes easier (since noise is gone). This is more of a preprocessing, but it leverages the PDE analogy (diffusion equation) for noise robustness. In fact, anisotropic diffusion is essentially a *physical model-based filtering* to enhance edges.

Electrostatic Analogy: A static analogy is to treat the intensity as an electric potential field. If the rings are high intensity, one could imagine them as charged rings. The gradient of intensity is like an electric field. One might attempt to find circles by placing hypothetical positive charge and seeing if the "force" aligns in circular symmetry. This is stretching the analogy, but mathematically, it could involve solving Laplace's equation with boundary conditions or looking at level sets of the potential. Level sets of a harmonic function in a ring configuration could naturally be circles (since for a point charge, equipotentials are circles). However, using this for detection is indirect; a more practical approach is the inverse: treat the image edges as equipotential lines and use known results from potential theory that equipotentials in free space are circles if sources are points. But here sources are more complex.

Mechanical Analogy: Consider a membrane or plate that is constrained or has different tension at certain radii. If you pluck it, the deformation might concentrate along those radii. Another approach: imagine a **diffusion front or fluid** that fills the image but has to percolate through the intensity landscape. Concentric boundaries could act like dams holding a fluid until pressure builds enough to spill over, indicating their presence by a delay. These analogies, while interesting, generally convert the problem into another domain (time or frequency) where the presence of rings manifests as some measurable effect (delay, resonance, etc.).

The advantage of physics-based analogies is often intuition about **robustness**: waves and diffusion naturally average out noise and fill gaps, which mirrors a desirable property for detection algorithms. For example, the wave propagation segmentation by Porikli [17] [18] used an absorbing medium with varying speeds to robustly grow regions and stop at boundaries, demonstrating noise-robust boundary detection by mimicking physical wavefronts. These methods tend to involve solving PDEs numerically (like finite difference for diffusion or fast marching for Eikonal) and choosing parameters that correspond to physical properties (wave speed, diffusion coefficient) which can be tuned to the image's noise level and contrast.

7. Noise Robustness, Discretization, and Iterative Refinement

7.1 Preprocessing and Noise Reduction

Real-world intensity data invariably contain noise and discretization artifacts (pixelation, aliasing). Before applying any circle detection, it is often beneficial to preprocess the image. **Gaussian smoothing** (blurring) is a common step to reduce high-frequency noise, at the cost of slightly blurring edges. A Gaussian filter can be tuned to the scale of noise – too large a sigma will wash out the thin ring edges, so one must balance noise removal and edge preservation. More advanced is **anisotropic diffusion filtering**, which, as mentioned, runs a PDE that smooths within regions but not across strong gradients. This effectively reduces noise while keeping the concentric edges sharp. **Bilateral filtering** could also be used for similar reasons (it's an edge-preserving smoothing).

Multiscale Analysis: To handle various noise scales and potential blurring, one can analyze the image at multiple resolutions. For example, use a Gaussian pyramid: at a coarser scale (downsampled image), minor noise is eliminated and the rings (if large enough) are still visible, providing a good initial guess for radius and center. Then one can progressively refine on finer scales. Many edge detection methods use a scale-space approach (e.g., the Laplacian of Gaussian edge detector uses a certain sigma; one could try multiple sigmas to ensure the rings are detected even if fuzzy). A **Canny edge detector** with appropriate smoothing can yield a cleaner edge map of the rings, which then feeds into circle fitting or Hough. Canny essentially applies Gaussian smoothing, gradient computation, non-maximum suppression, and hysteresis thresholding – steps designed to handle noise and edge continuity. Ensuring the edge map is clean greatly aids subsequent circle detection.

Discretization and Interpolation: Pixel grids may cause a circle's edge to appear jagged or with thickness. Sub-pixel refinement techniques address this. One can fit a local quadratic curve to the edge intensity profile to estimate a sub-pixel edge location. For example, after a Sobel or Canny edge, one could do a parabolic fit of intensities across the edge to pinpoint the zero-crossing at subpixel accuracy. This yields more accurate points for circle fitting. Similarly, Hough transform accumulators can be interpolated: rather than voting to the nearest integer center, one can vote with a distribution or use gradient direction to pinpoint the center more continuously [28]. If an image is very low resolution, one might even upsample (interpolate) it to a finer grid before detection (this is effectively what subpixel methods do internally).

For the two concentric circles, discretization might cause them to appear connected or broken in places. **Morphological operations** like closing (to fill small gaps in edges) or opening (to remove spurious bits) can clean the edge patterns. This is especially useful if noise causes the rings to be not perfectly continuous. A morphological approach could be: threshold the image to get bright regions, then use a ring-shaped structuring element to detect ring-like shapes. However, designing a structuring element that captures arbitrary radius is tricky – hence transform methods (Hough) are more common.

7.2 Iterative Refinement and Hybrid Strategies

Iterative Refinement: No single method may perfectly detect the circles in one shot, especially under noise. An iterative strategy can combine methods: for example, use Hough transform to get an initial center and radii. Then use that as initialization for an active contour that precisely locks onto the edges, allowing for slight deviations from perfect circularity. The active contour will refine the shape to better align with the true edges, essentially doing a local optimization. Another example: use a coarse-to-fine Hough (first a large parameter step to find approximate solution, then narrow down with smaller steps around that solution for precision). If using optimization (Gauss–Newton on circle parameters), the iterative method itself will refine until convergence, but it might converge to a local minimum. One can rerun the optimization from multiple starting points (perhaps from different random RANSAC fits) to ensure the global minimum is found (detecting the correct pair of circles).

Combined Edge and Region Criteria: Robust detection might integrate multiple cues. For instance, one can incorporate both edge information *and* region intensity consistency in one framework. A custom objective function could include a term for how well a proposed circle matches an intensity contrast (difference between inside and outside intensity) and a term for edge alignment (how many strong gradient pixels lie on the circle). This combined objective can be optimized iteratively. Alternatively, one can alternate between detecting the center (by looking at gradient convergence or symmetry of the image) and detecting the radii (by radial profiling), iterating until both center and radii stabilize.

Gradient-Based Refinement: Many of the above approaches yield a gradient (or derivative) that can be used in a **gradient ascent/descent** algorithm. For example, consider a simpler cost: $J(h,k) = -\sum_{(x,y)\in\text{image}} w(x,y)\, I(x,y)$, where $w(x,y)$ is a weight that is positive on the expected ring and negative elsewhere (like a ring template centered at (h,k)). The gradient $\nabla_{h,k} J$ can be computed by moving the template slightly and measuring change. One could perform gradient ascent on $J(h,k)$ to find the best center. Similarly, one can fix center and optimize radii by gradient search (one could compute the derivative of some radial intensity measure with respect to radius). These gradient computations in practice might be complicated by the discrete nature of image sampling, but one can approximate them by difference or even derive an analytic form if assuming some differentiable interpolation of the image.

Stopping Criteria and Validation: An iterative detection algorithm should have criteria to stop and confirm detection. For example, if using an active contour, one stops when the contour change between iterations is below a threshold. For an optimization solver, one stops when parameter updates are below tolerance or cost change is minimal. It's also important to validate that two circles are indeed found: e.g., check that the two radii are significantly different (to ensure two distinct circles rather than one circle twice) and that a sufficient portion of each circle's circumference was supported by the data. One might also enforce *concentricity* explicitly in refinement: after each update of center, force both circles to that center; after update of radii, force an ordering (which is inner, which outer). These constraints ensure the solution remains two concentric circles.

7.3 Handling Artifacts and Bias

Bias from Pixelation: A known issue in circle detection is that naive methods can bias the radius or center due to pixel grid effects (for example, circles tend to appear larger when digitized). One mitigation is to apply a correction factor or to include a model of the imaging point-spread function. For instance, if the rings are somewhat blurred, the detected edge might be offset from the true half-maximum. One can correct the radius by a small amount based on that blur. If subpixel edge detection is used, this largely addresses the bias.

Two-Circle Interaction: Concentric circles might confuse certain algorithms: e.g., a Hough transform might vote strongly for one radius and somewhat smear votes for the other if one ring edge is much stronger than the other. A strategy is to detect circles one at a time: find the strongest circle (likely the outer if it has more circumference) then remove its edge points and detect the next. This sequential detection can be iterated if more circles were present. It's a greedy approach but often effective. Another approach is to look at **difference images**: subtract a blurred version of the image from the original to highlight high-frequency content (edges). If two circles have different thickness or intensity, sometimes a single global threshold might miss one; adaptive thresholding can help ensure both are picked up.

In summary, robust detection of concentric circles in a noisy, discretized intensity matrix benefits from **multi-faceted approaches**: smoothing and multi-scale analysis to handle noise, Hough/Radon or analytic methods to get initial guesses, variational/PDE methods to refine and handle gaps, linear algebra for efficient solving of parameters, and topological methods to double-check the presence of the loops. Each branch of mathematics contributes a unique perspective: from the **algebraic** (solving equations for circles), the **variational** (evolving solutions via gradient flows), the **geometric** (using curvature and invariants), the **topological** (counting loops), to the **physical** (simulating waves and diffusion). Combining these ensures a **comprehensive and robust solution** to detect the two concentric circles accurately even in challenging data conditions.

References to Mathematical Concepts and Techniques

- **Analytical Geometry:** Basic circle equations and direct solution methods for circle parameters using three points or overdetermined systems [1] [29].
- **Linear Algebra:** Least-squares circle fitting formulated as solving $Bu = 0$ via SVD/eigen-decomposition [3].
- **Hough/Radon Transform:** Parameter-space voting for circles (3D accumulator for center and radius) [4] [5]; relationship between Hough and continuous Radon transform [7] [8].
- **Calculus of Variations:** Active contour (snake) model as energy-minimizing spline guided by image forces [9]; Euler-Lagrange leads to gradient descent PDE for contour evolution.
- **Level Set and PDEs:** Geometric active contours using curve-shortening flows and level set methods, allowing topological changes [12].
- **Persistent Homology:** Topological analysis of image level sets to detect loops (holes) with high persistence [14].
- **Differential Geometry:** Möbius (conformal) transformations map circles to circles/lines [24]; curvature-based identification of circular arcs.
- **Physics Analogies:** Wave propagation interpreted as a segmentation mechanism (fast marching solving Eikonal equation to stop at boundaries) [17] [18]; diffusion processes for noise reduction and edge preservation.

---

[1] [2] [3] [19] [20] [21] [22] [23] [29] gander.dvi
https://www.emis.de/journals/BBMS/Bulletin/sup962/gander.pdf

[4] [5] [6] [28] Hough transform - Wikipedia
https://en.wikipedia.org/wiki/Hough_transform

[7] [8] image processing - What's the difference between the Hough and Radon transforms? - Signal Processing Stack Exchange
https://dsp.stackexchange.com/questions/470/whats-the-difference-between-the-hough-and-radon-transforms

[9] [11] [12] Active contour model - Wikipedia
https://en.wikipedia.org/wiki/Active_contour_model

[10] Robust shape based active contour for circle detection - ResearchGate
https://www.researchgate.net/publication/233621837_Robust_shape_based_active_contour_for_circle_detection

[13] [14] [25] [26] [27] Topology-based segmentation of 3D confocal images of emerging hematopoietic stem cells in the zebrafish embryo - PMC
https://pmc.ncbi.nlm.nih.gov/articles/PMC11704129/

[15] [16] [17] [18] merl.com
https://www.merl.com/publications/docs/TR2004-010.pdf

[24] Möbius transformation - Wikipedia
https://en.wikipedia.org/wiki/M%C3%B6bius_transformation