Updates to the Circle Hough Algorithm

Ludovico Bianchi, Forschungszentrum Jülich

P̄ANDA Collaboration Meeting #54, GSI, Sep 08 2015

Circle Hough Algorithm

General principle

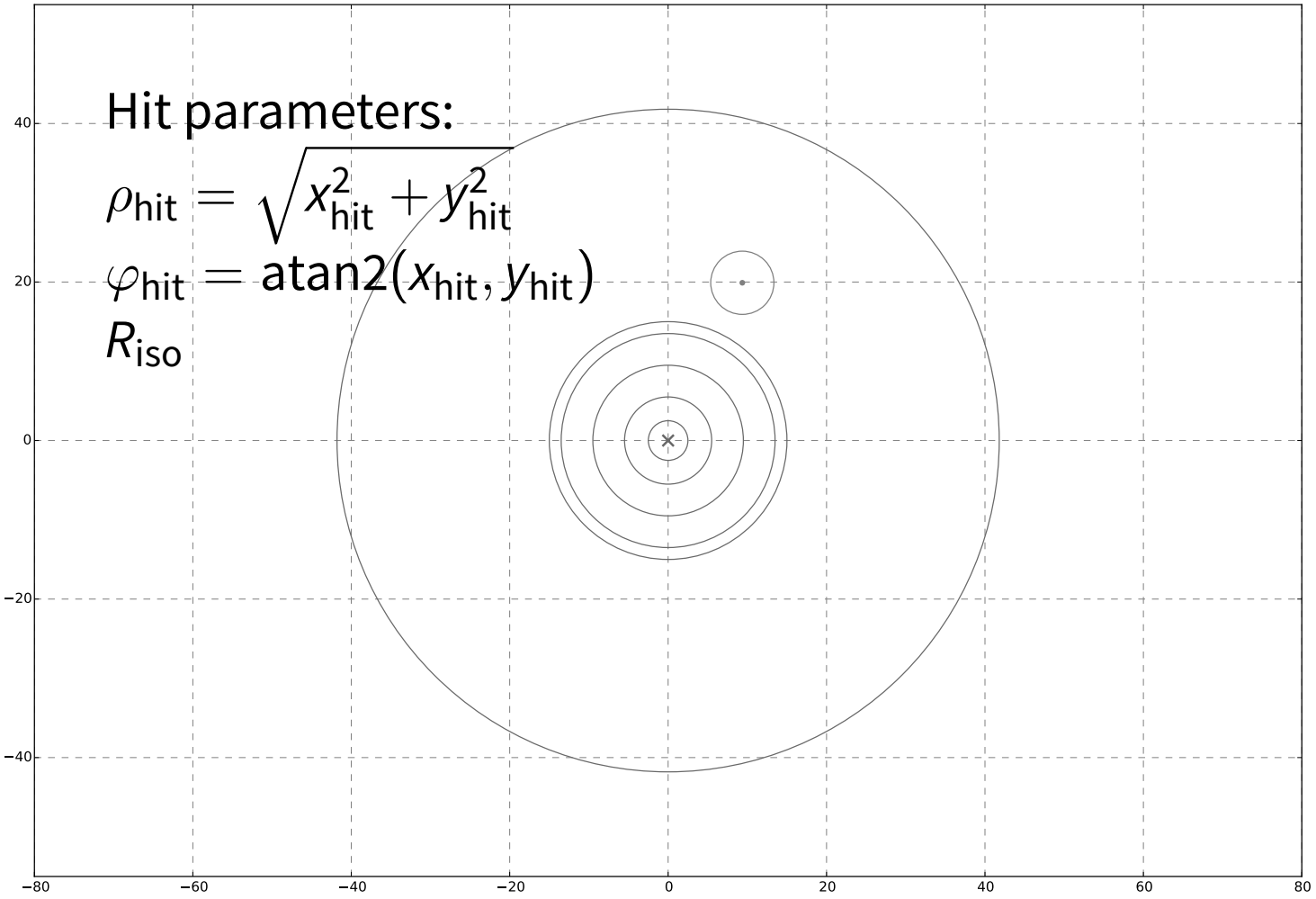
- Based on the Hough transform algorithm for detection of a shape S within a data set D
  - Parametrize target shape S with set of parameters
  - For each point in D, calculate all possible instances of S
  - Collect votes in parameter space P (Hough space)
  - Most voted sets of parameters in $P \Longrightarrow$ instances of S in D
- For trackfinding in P̄ANDA
  - Calculate all possible tracks compatible with one hit
  - Repeat for many hits
  - Most voted track parameters $\Longrightarrow$ real tracks

CH Algorithm

Current approach

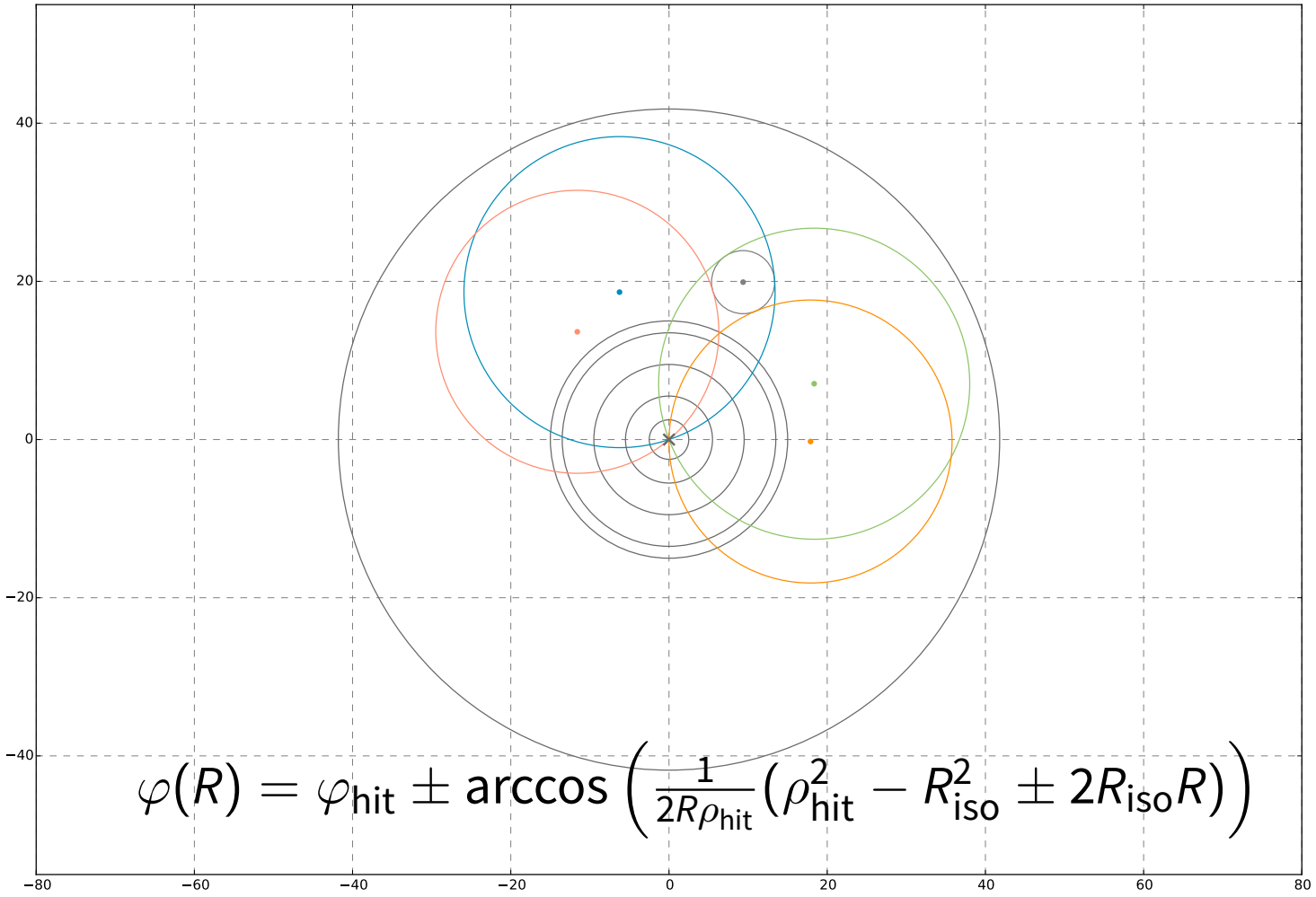
- Shape to find: primary tracks in 2D
  $\Longrightarrow$ circles in the (x, y) plane, passing through IP at $(0, 0)$ and hit at $(x_{\text{hit}}, y_{\text{hit}})$
  - Point hit (MVD)
  - Extended hit (STT): tangent to isochrone radius
- Parameters: coordinates of circle centers $\Longrightarrow$ 2D Hough space
  - Use one parameter as sampling parameter
  - Calculate second parameter from hit contact condition

Hough Circles



Hough Circles

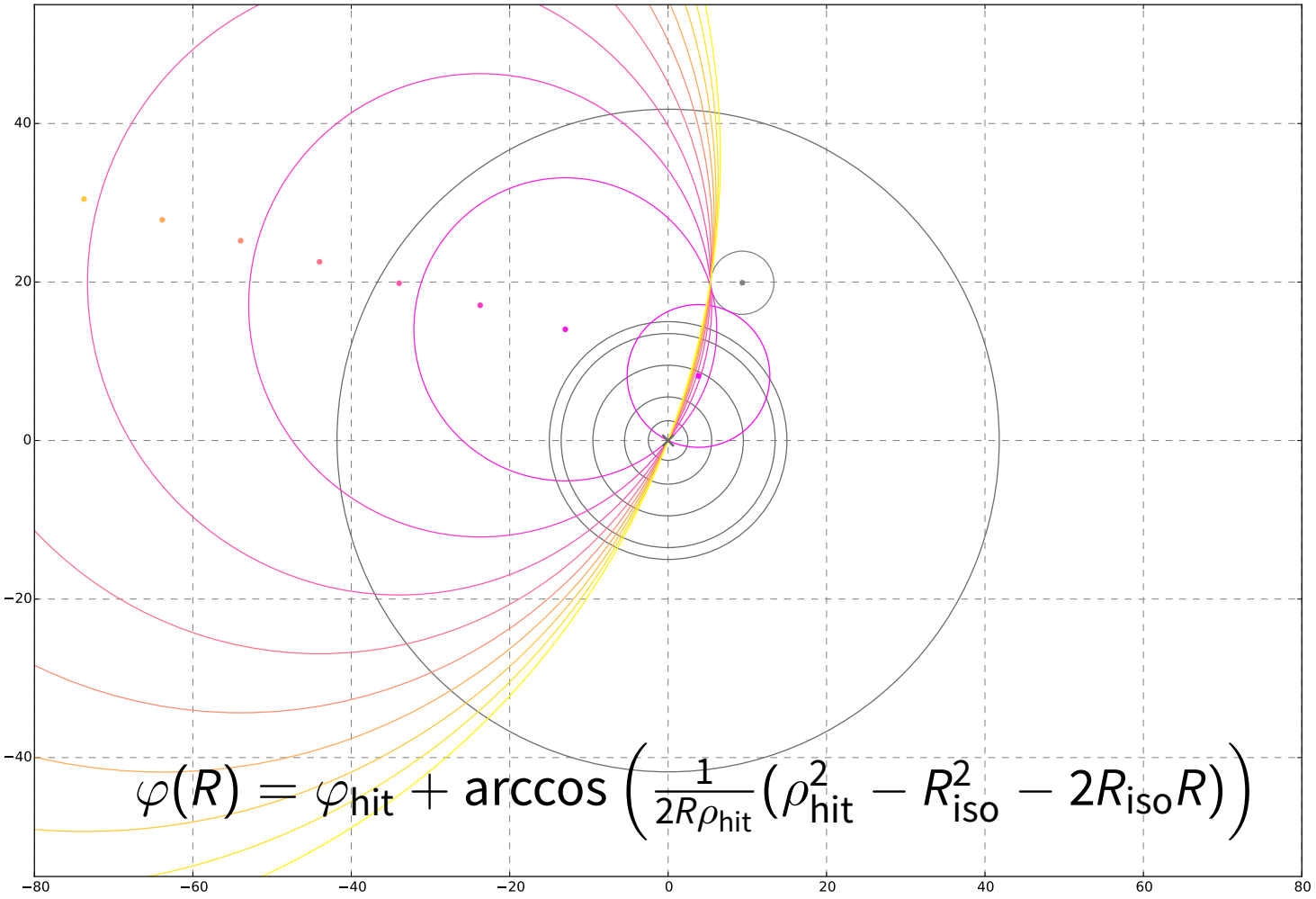
For each hit, 4 possible circles



$$\varphi(R) = \varphi_{\text{hit}} \pm \arccos\left(\frac{1}{2R\rho_{\text{hit}}}\left(\rho_{\text{hit}}^2 - R_{\text{iso}}^2 \pm 2R_{\text{iso}}R\right)\right)$$

Hough Circles, “External” Series

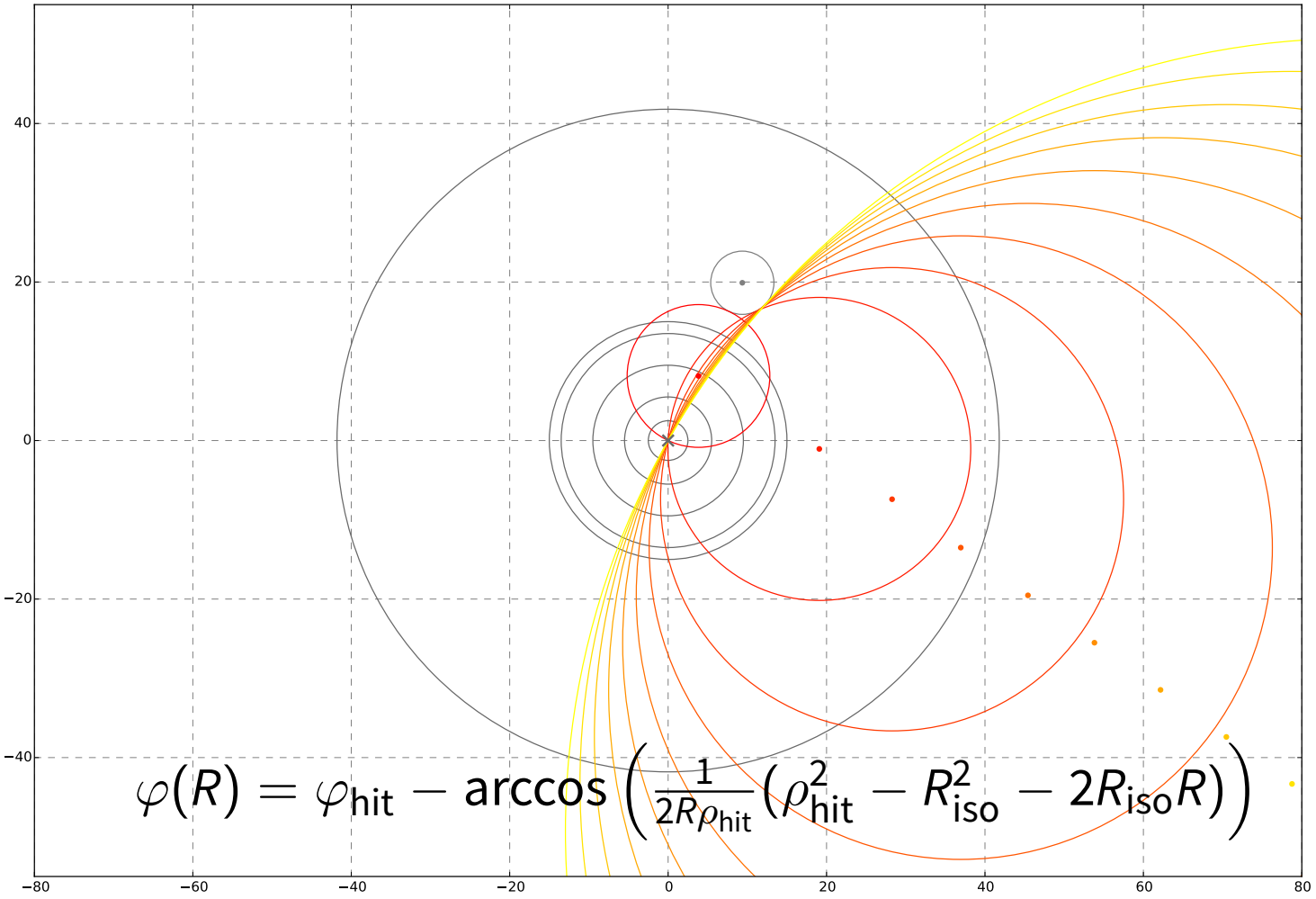
*Hit is **external** w.r.t. the Hough circle; counterclockwise direction*



$$\varphi(R) = \varphi_{\text{hit}} + \arccos\left(\frac{1}{2R\rho_{\text{hit}}}\left(\rho_{\text{hit}}^2 - R_{\text{iso}}^2 - 2R_{\text{iso}}R\right)\right)$$

Hough Circles, “External” Series

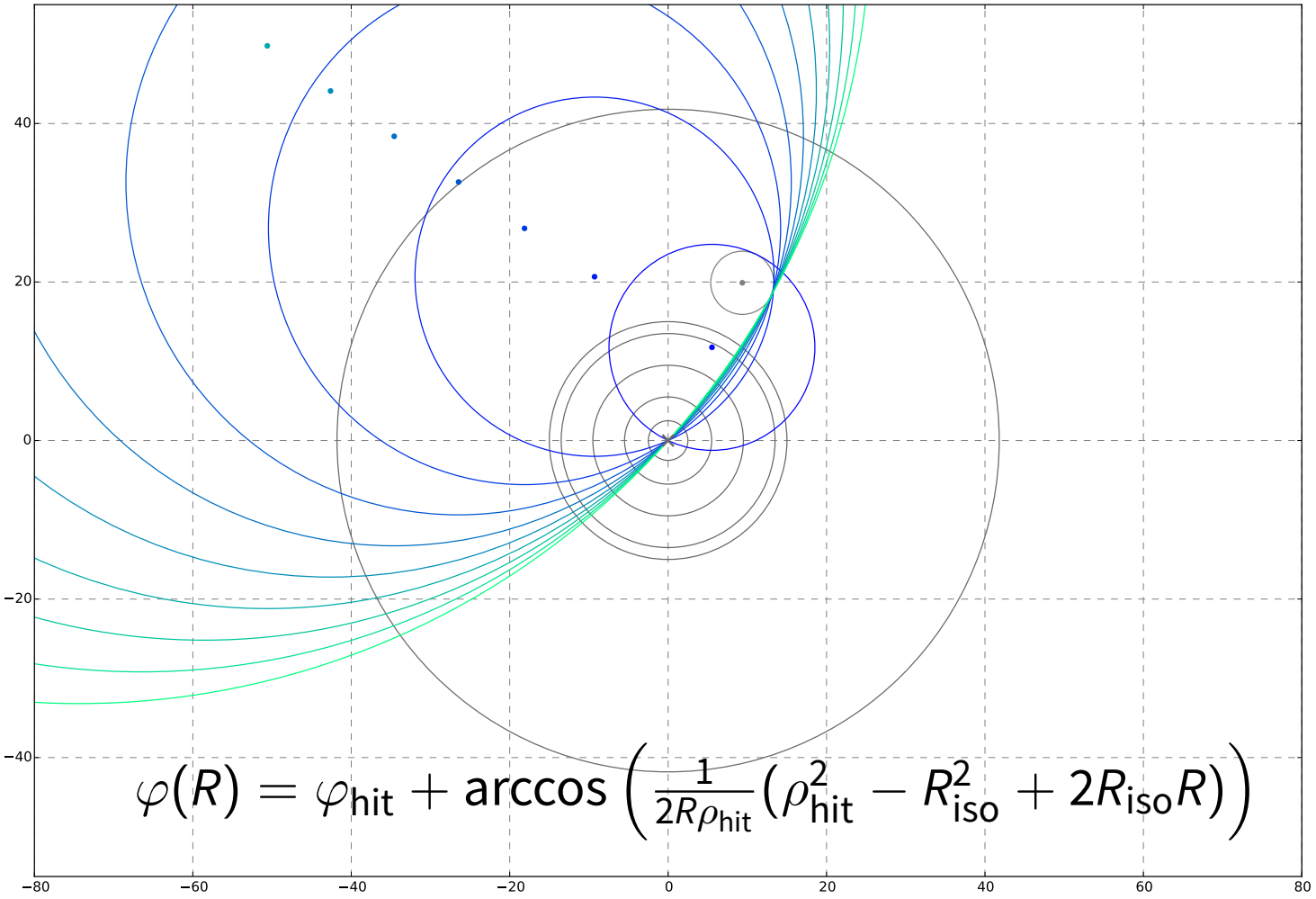
*Hit is **external** w.r.t. the Hough circle; clockwise direction*



$$\varphi(R) = \varphi_{\text{hit}} - \arccos\left(\frac{1}{2R\rho_{\text{hit}}}\left(\rho_{\text{hit}}^2 - R_{\text{iso}}^2 - 2R_{\text{iso}}R\right)\right)$$

Hough Circles, “Internal” Series

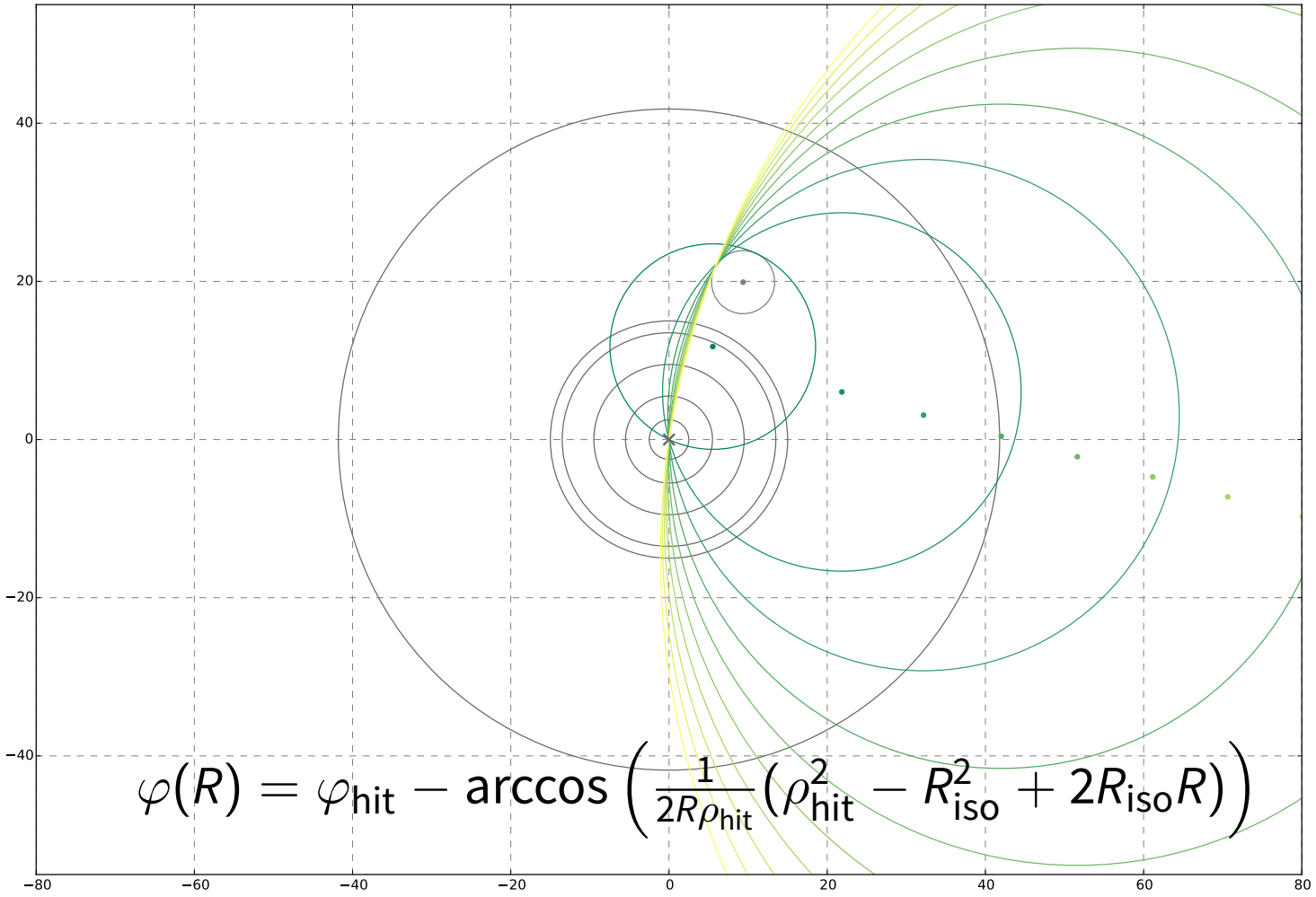
*Hit is **internal** w.r.t. the Hough circle; counterclockwise direction*



$$\varphi(R) = \varphi_{\text{hit}} + \arccos\left(\frac{1}{2R\rho_{\text{hit}}}\left(\rho_{\text{hit}}^2 - R_{\text{iso}}^2 + 2R_{\text{iso}}R\right)\right)$$

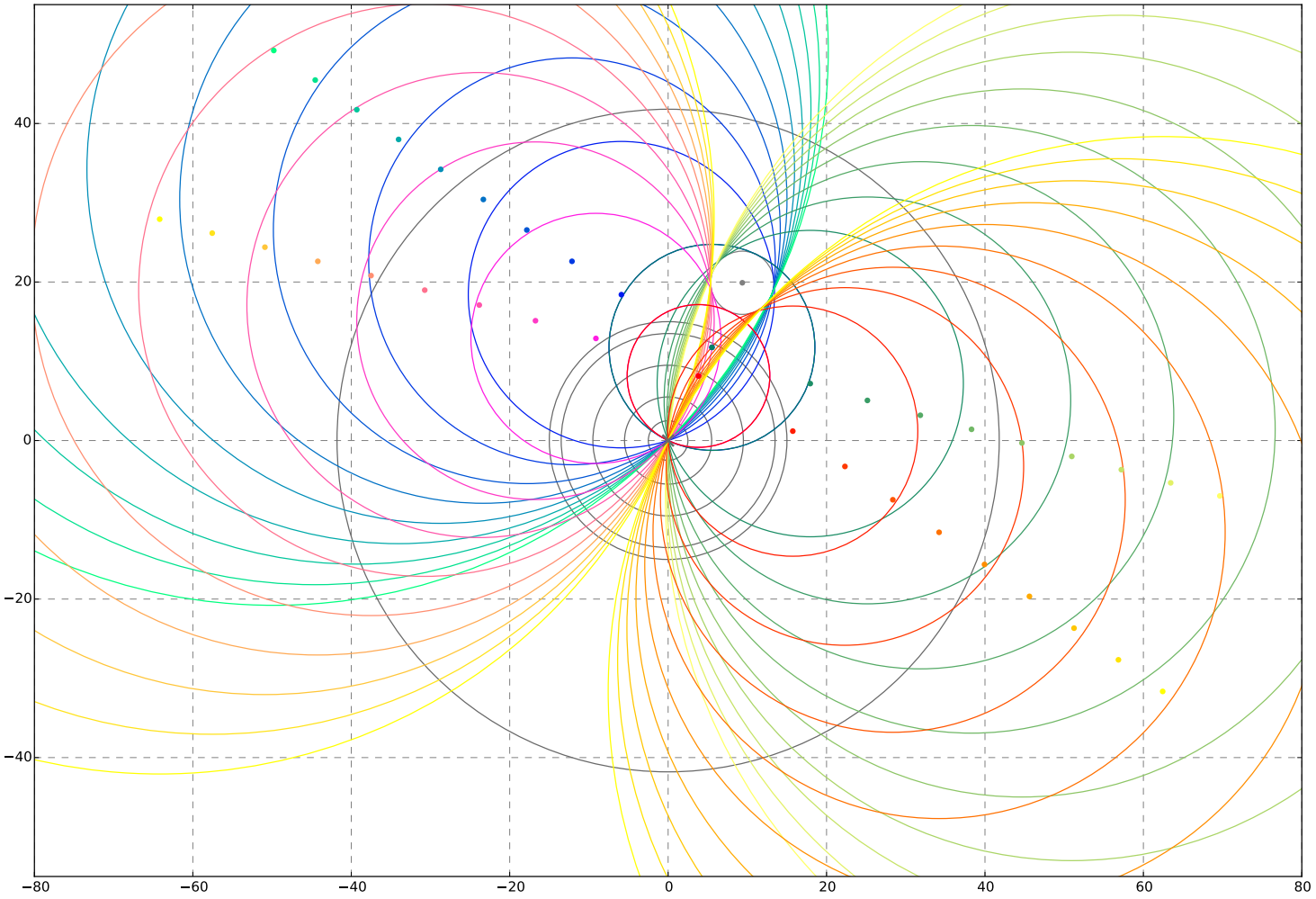
Hough Circles, “Internal” Series

*Hit is **internal** w.r.t. the Hough circle; clockwise direction*

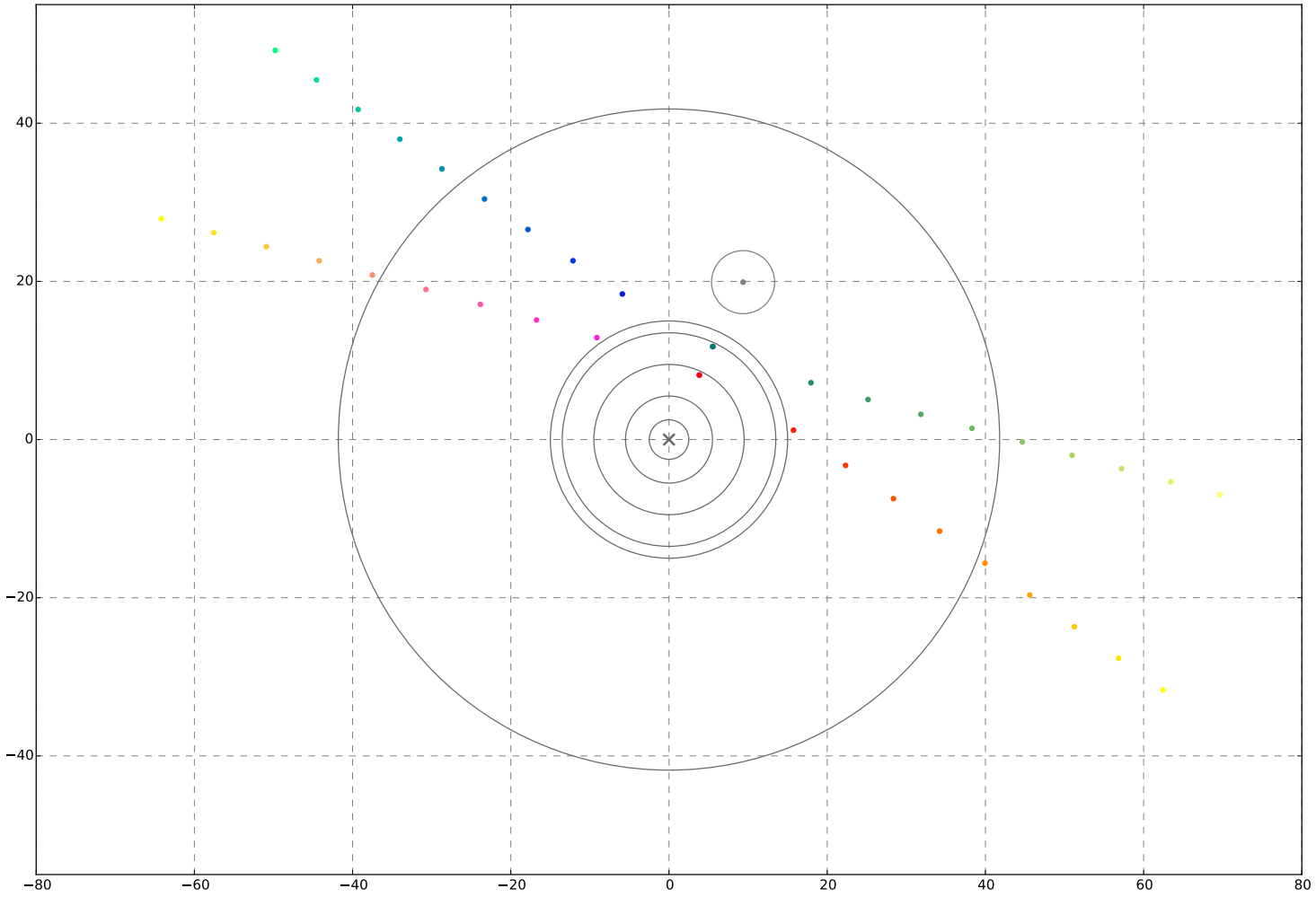


$$\varphi(R) = \varphi_{\text{hit}} - \arccos\left(\frac{1}{2R\rho_{\text{hit}}}\left(\rho_{\text{hit}}^2 - R_{\text{iso}}^2 + 2R_{\text{iso}}R\right)\right)$$

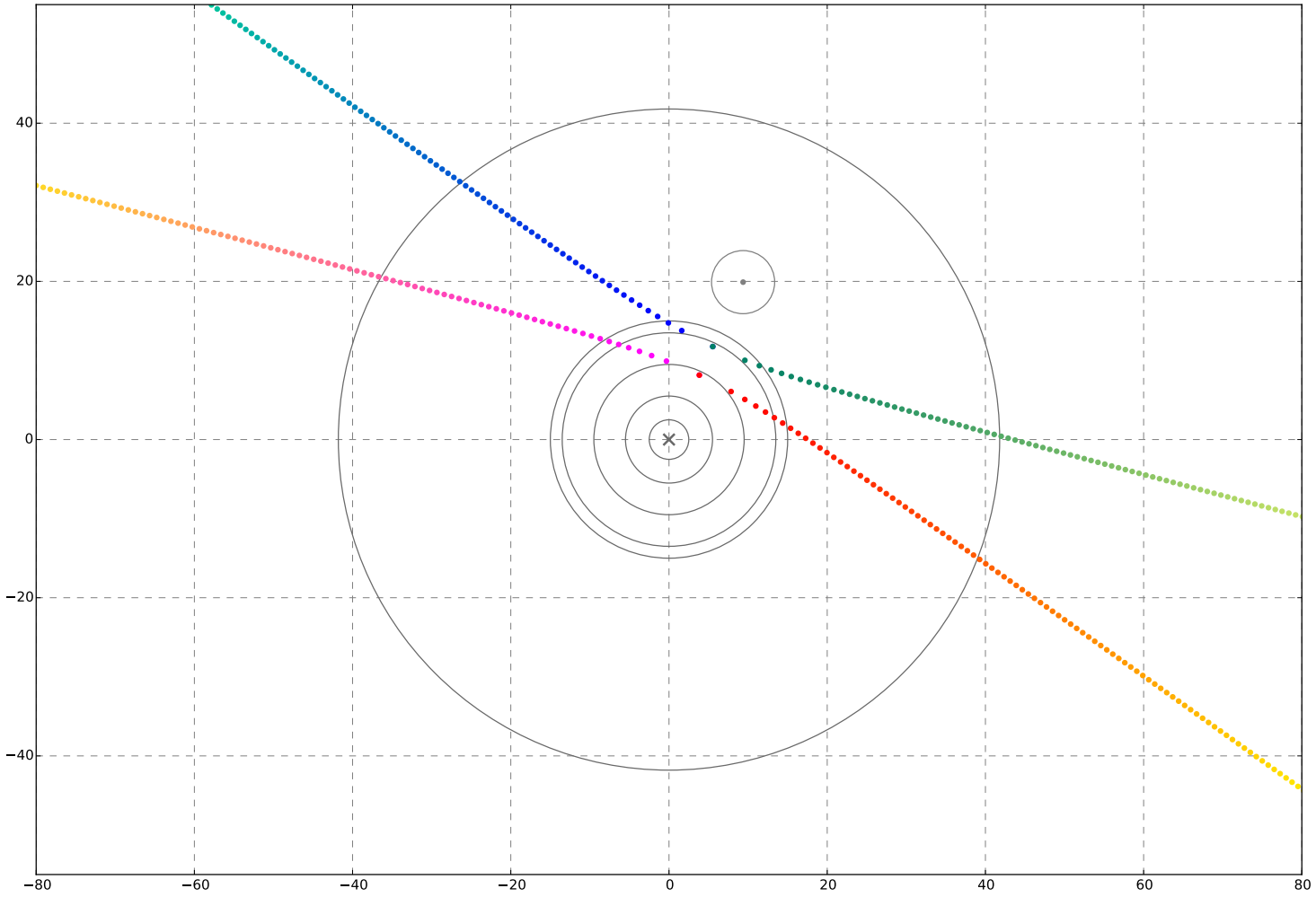
Hough Circles, Both Series



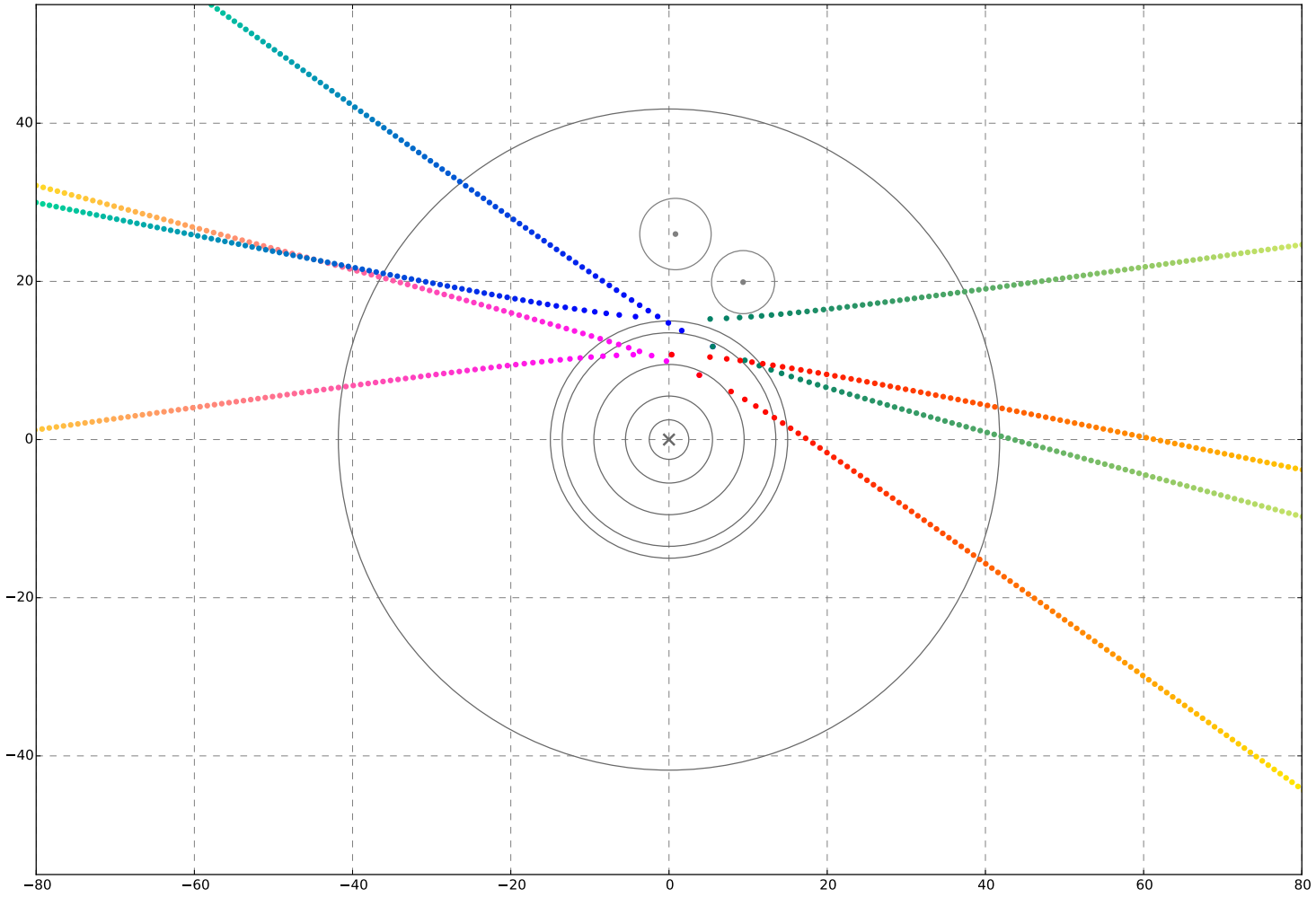
Hough Circles, Multiple Hits



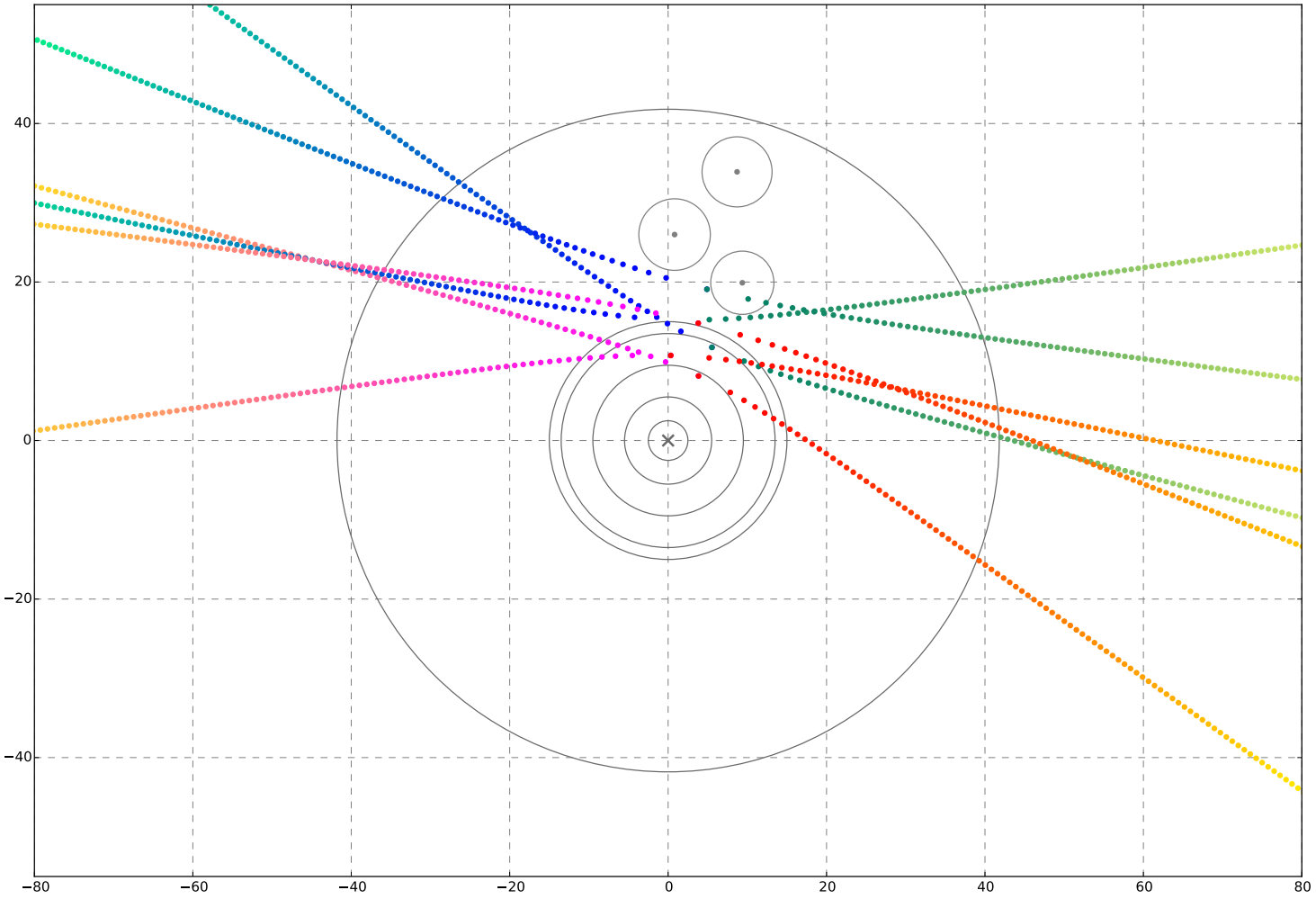
Hough Circles, Multiple Hits



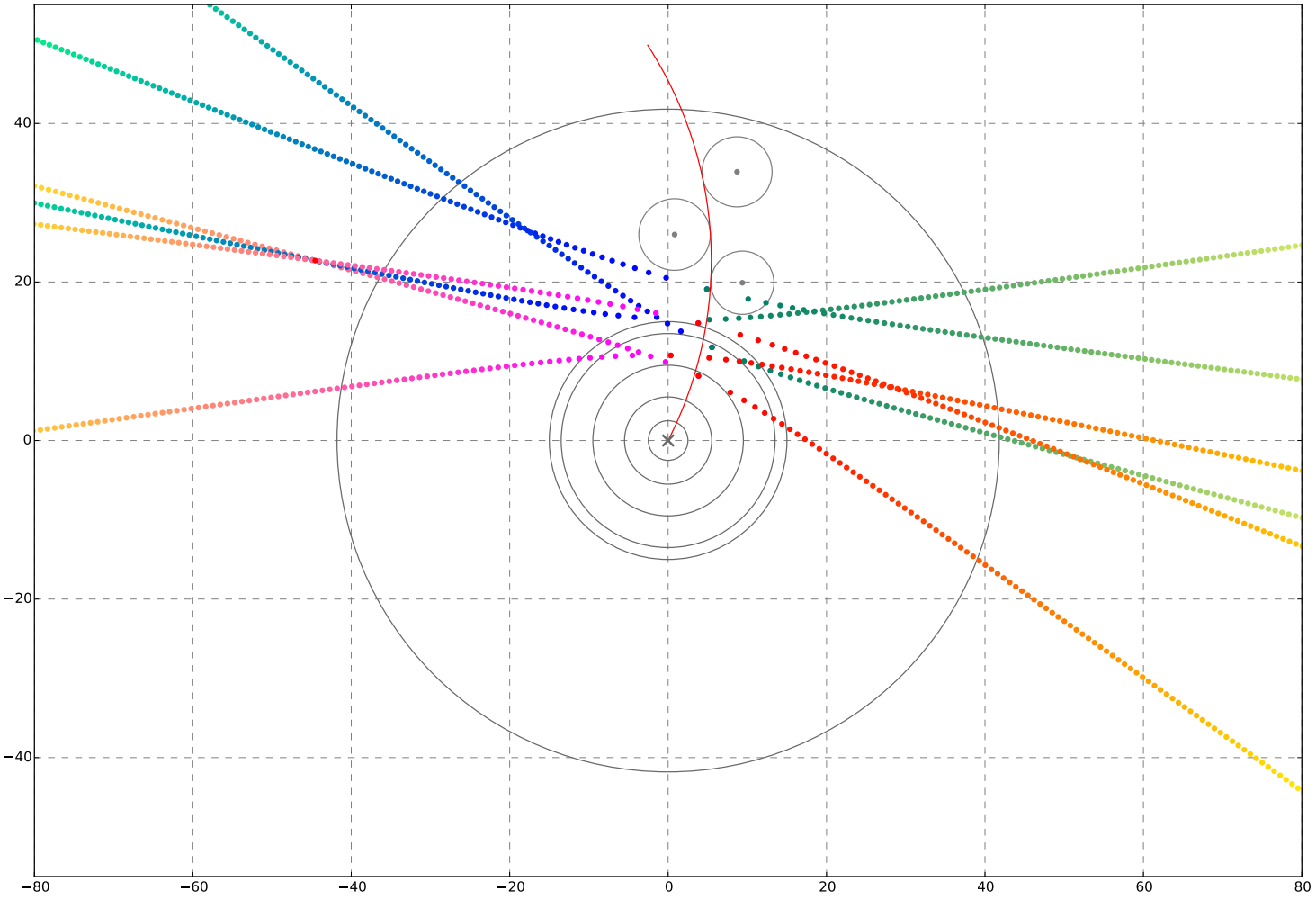
Hough Circles, Multiple Hits



Hough Circles, Multiple Hits



Hough Circles, Multiple Hits

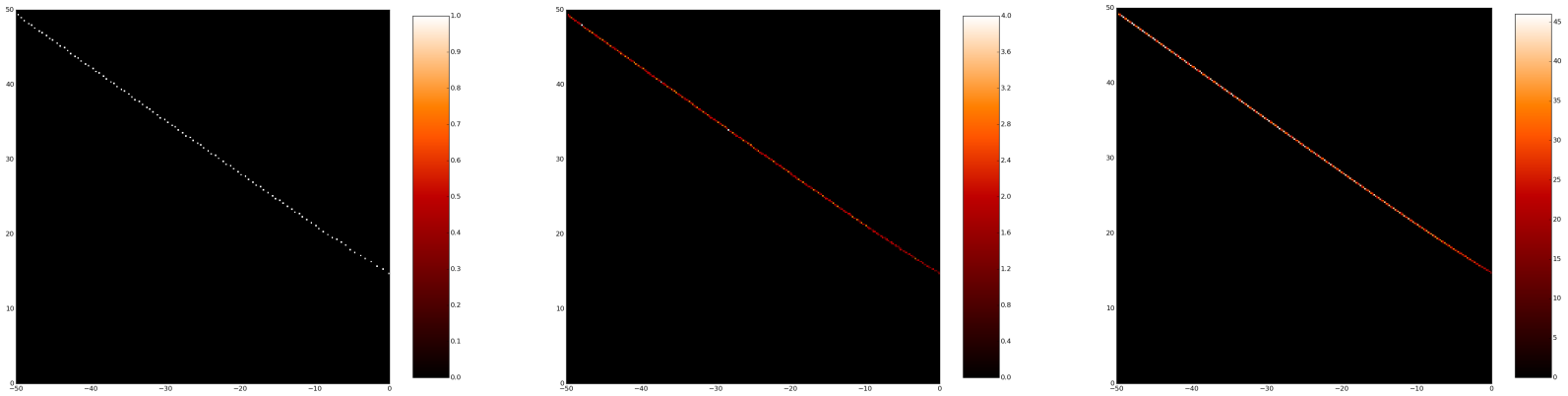


Accumulator Array

- Collect parameters in accumulator array
- Discrete 2D domain

⟹ Fine tune necessary between radius sampling and array size

⟹ Dishomogeneous structure makes peakfinding more difficult



- Each coordinate pair must fill the array exactly once

⟹ Current solution:

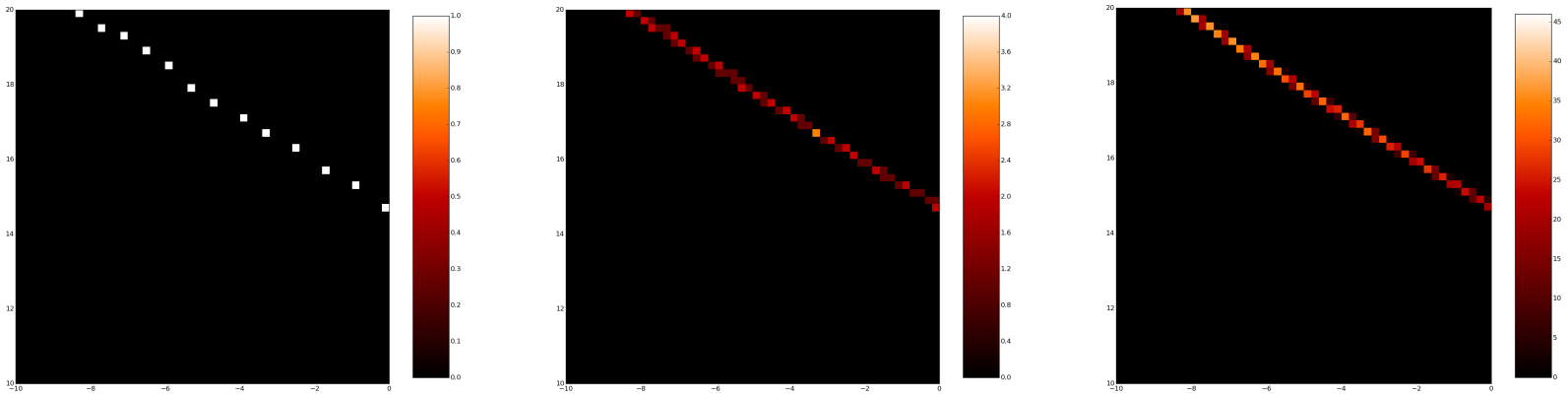
  — Reject values ending up in the same bin
  — Fill empty values by interpolation

Accumulator Array

- Collect parameters in accumulator array
- Discrete 2D domain

⟹ Fine tune necessary between radius sampling and array size

⟹ Dishomogeneous structure makes peakfinding more difficult



- Each coordinate pair must fill the array exactly once

⟹ Current solution:

— Reject values ending up in the same bin
— Fill empty values by interpolation

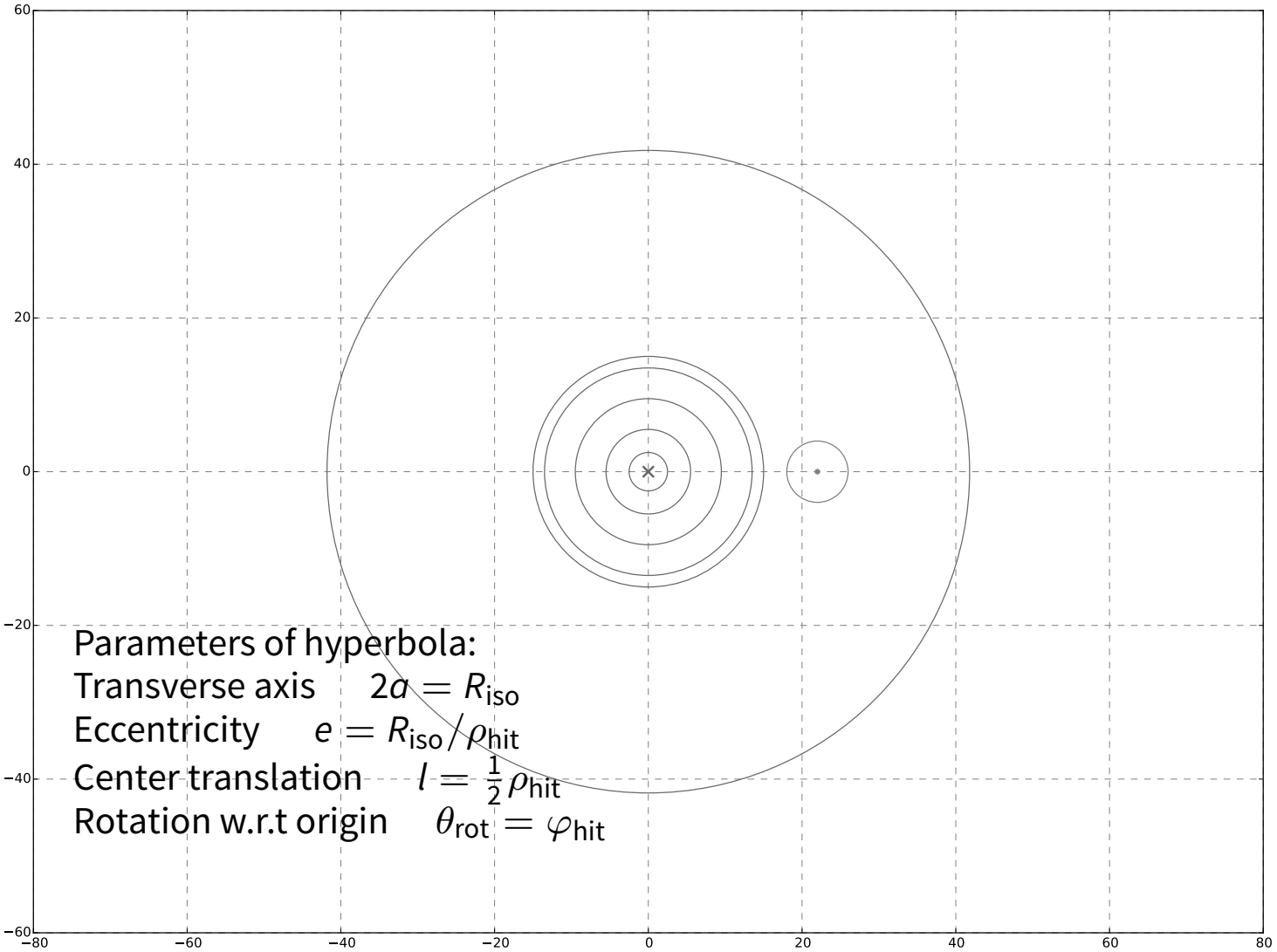
Alternative Approach

Based on two properties of our problem:

- Geometric locus of circle centers is known
  - Hyperbola, whose parameters are easily computed from hit data
- Analogy between Hough space and 2D image
  - Bin in 2D accumulator array $\Longrightarrow$ pixel in bitmap
  - Take advantage of image processing algorithms
  - Especially important for implementation on parallel architectures

$\Longrightarrow$ Operate directly on locus rather than on individual circles

Locus of Hough circle centers



Parameters of hyperbola:

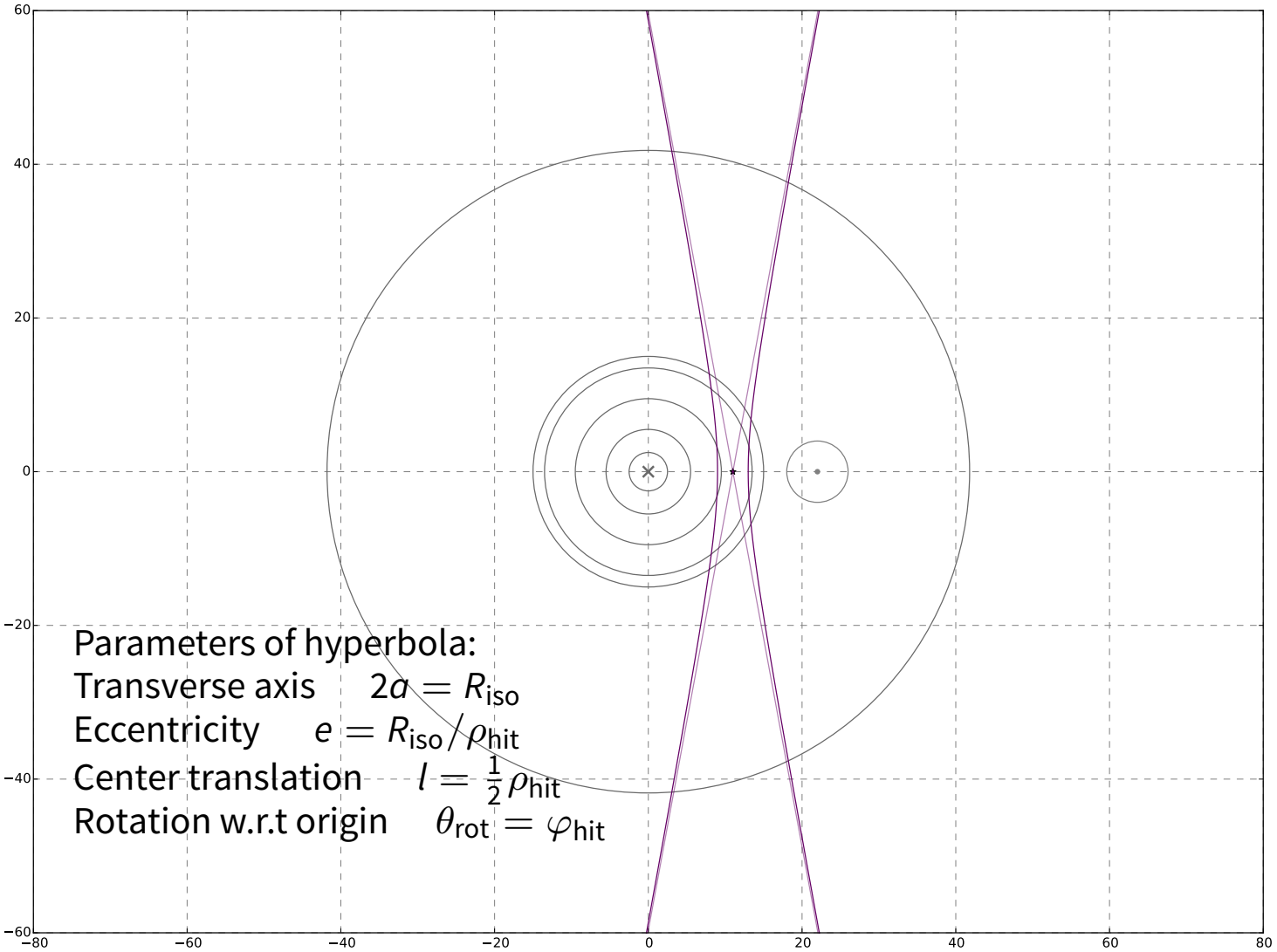
Transverse axis $2a = R_{\text{iso}}$

Eccentricity $e = R_{\text{iso}} / \rho_{\text{hit}}$

Center translation $l = \frac{1}{2}\rho_{\text{hit}}$

Rotation w.r.t origin $\theta_{\text{rot}} = \varphi_{\text{hit}}$

Locus of Hough circle centers



Parameters of hyperbola:

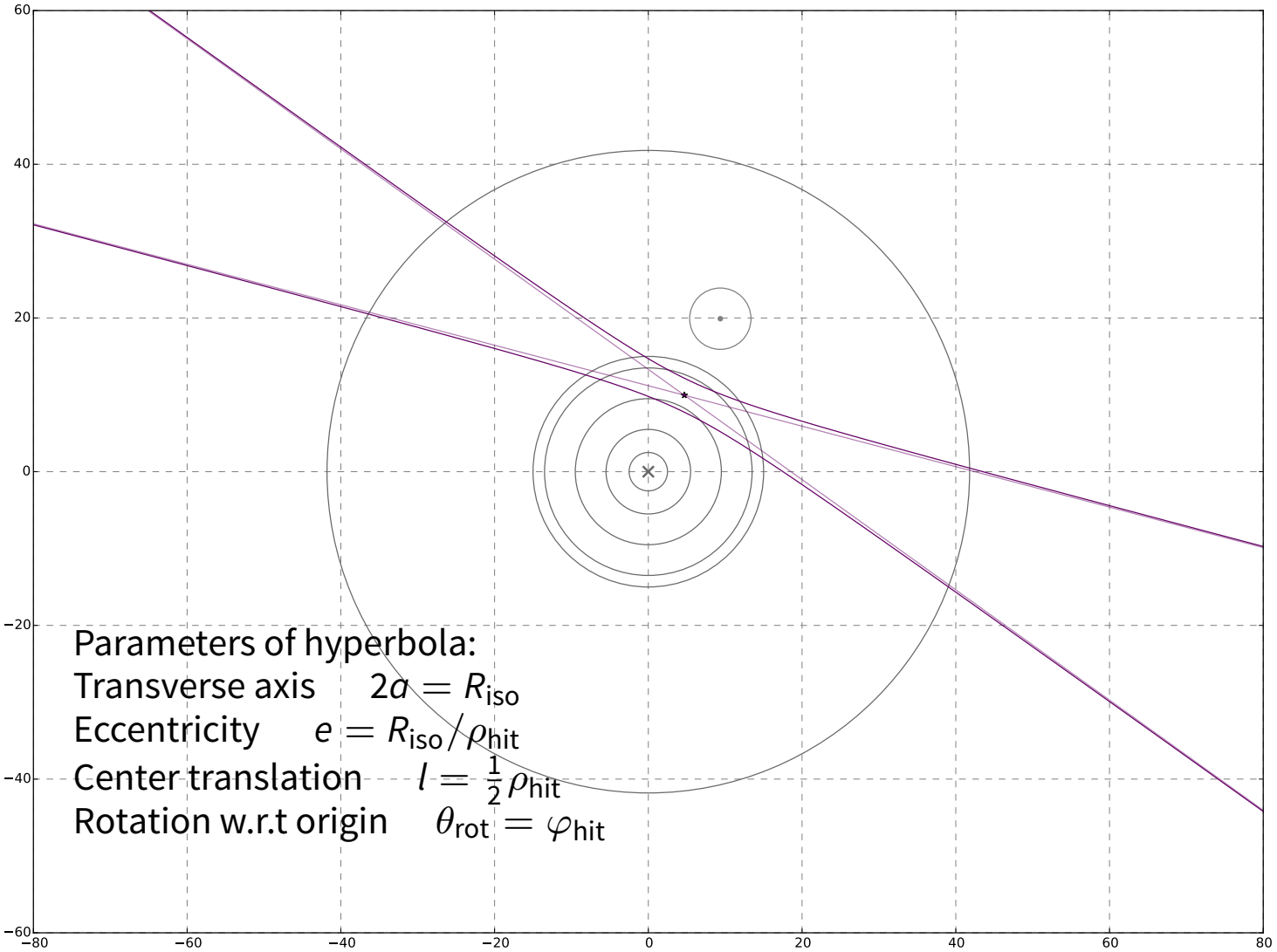
Transverse axis $2a = R_{\text{iso}}$

Eccentricity $e = R_{\text{iso}} / \rho_{\text{hit}}$

Center translation $l = \frac{1}{2}\rho_{\text{hit}}$

Rotation w.r.t origin $\theta_{\text{rot}} = \varphi_{\text{hit}}$

Locus of Hough circle centers



Parameters of hyperbola:

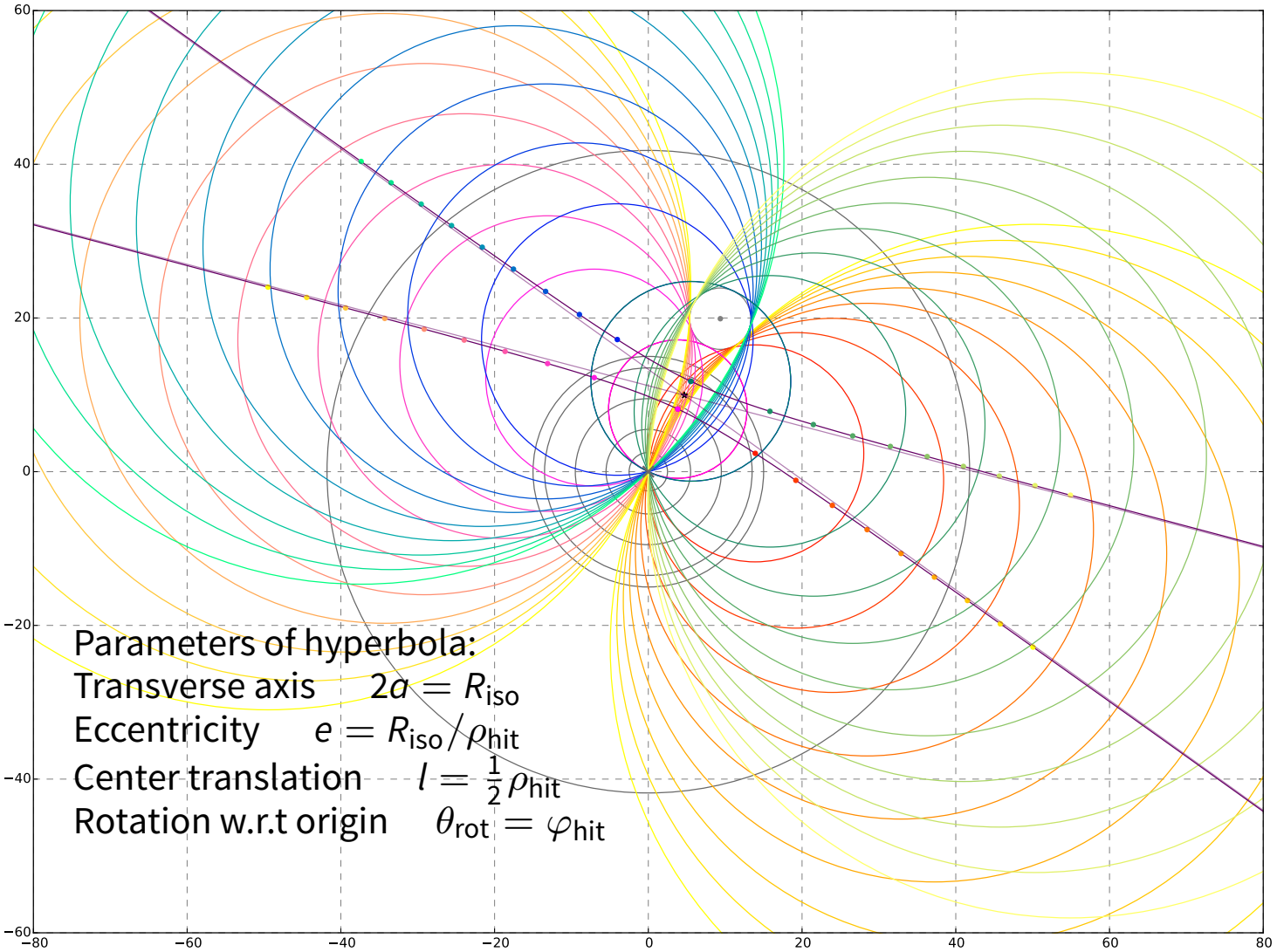
Transverse axis $2a = R_{\text{iso}}$

Eccentricity $e = R_{\text{iso}} / \rho_{\text{hit}}$

Center translation $l = \frac{1}{2}\rho_{\text{hit}}$

Rotation w.r.t origin $\theta_{\text{rot}} = \varphi_{\text{hit}}$

Locus of Hough circle centers



Parameters of hyperbola:

Transverse axis $2a = R_{\text{iso}}$

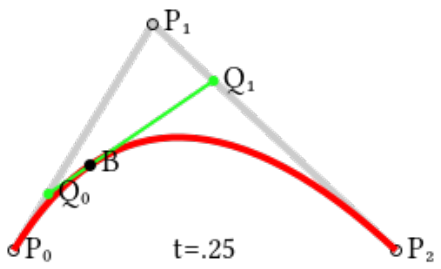
Eccentricity $e = R_{\text{iso}} / \rho_{\text{hit}}$

Center translation $l = \frac{1}{2}\rho_{\text{hit}}$

Rotation w.r.t origin $\theta_{\text{rot}} = \varphi_{\text{hit}}$

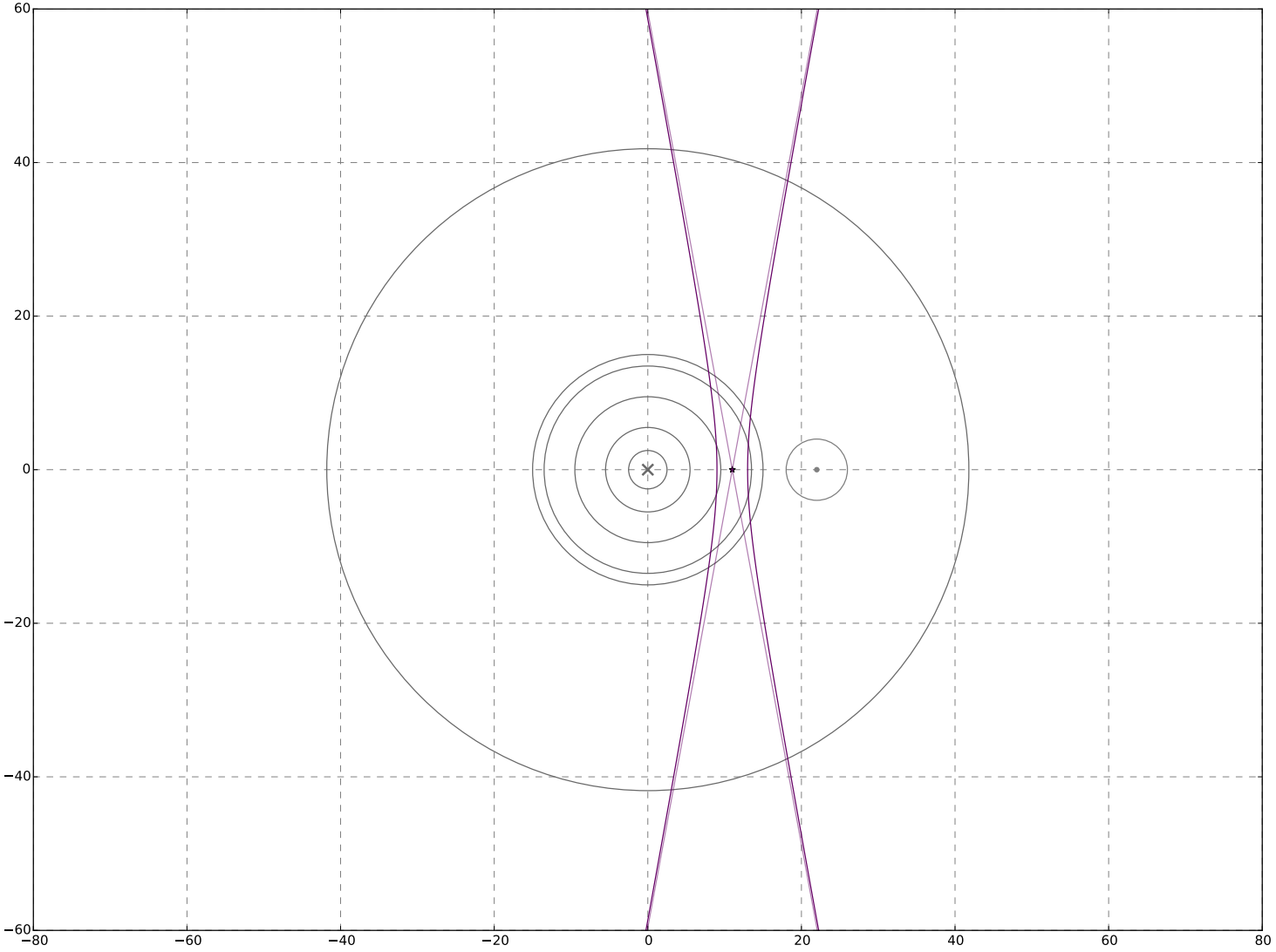
Bezier Curves

- Bezier curves: parametric polynomial curves of degree N, defined by their $N+1$ control points $\mathbf{P}_0 \ldots \mathbf{P}_N$
- Rational Bezier curves of degree 2 represent conic sections exactly $\Longrightarrow$ "lossless" description of hyperbola
  - Convenient parametrization
  - Easy linearization
  - Widely used in computer graphics



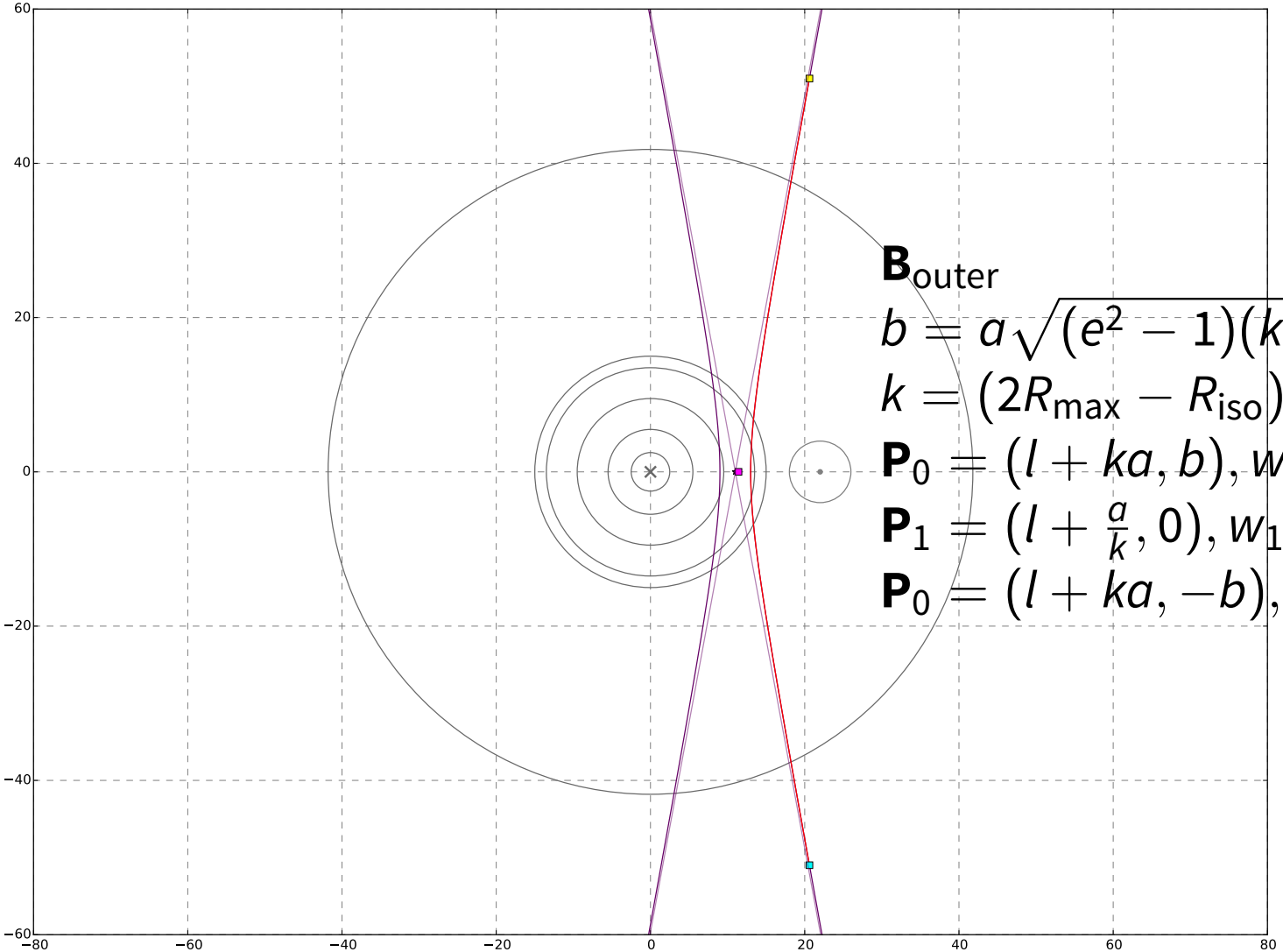
$$\mathbf{B}(t) = (1-t)^2 w_0 \mathbf{P}_0 + 2(1-t)t w_1 \mathbf{P}_1 + t^2 w_2 \mathbf{P}_2, \qquad 0 \leq t \leq 1$$

Parametrization with Bezier Curves



Parametrization with Bezier Curves

"Outer" curve



$$\mathbf{B}_{\text{outer}}$$

$$b = a\sqrt{(e^2-1)(k^2-1)}$$

$$k = (2R_{\text{max}} - R_{\text{iso}})/\rho_{\text{hit}}$$

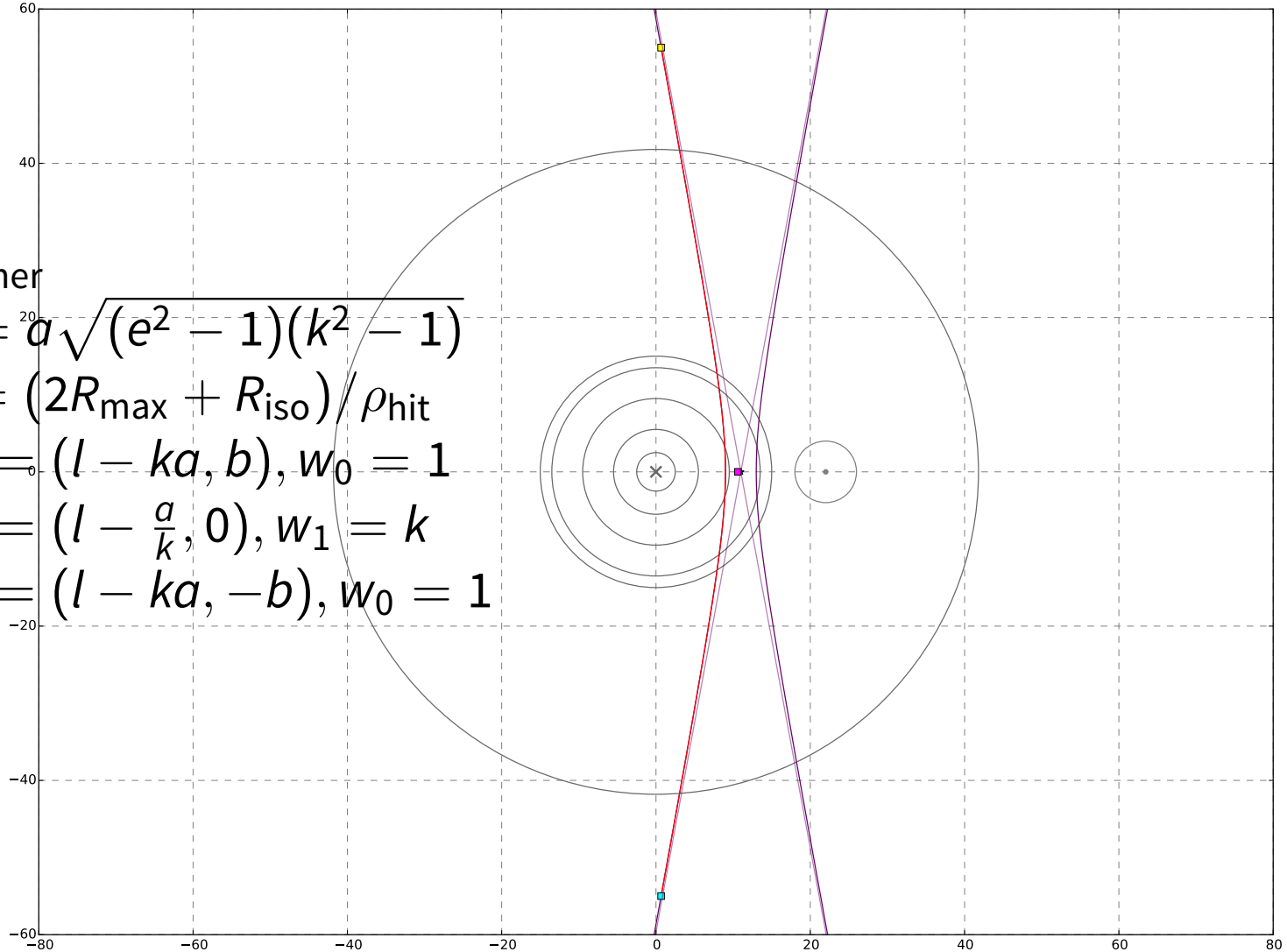
$$\mathbf{P}_0 = (l + ka, b), w_0 = 1$$

$$\mathbf{P}_1 = (l + \tfrac{a}{k}, 0), w_1 = k$$

$$\mathbf{P}_0 = (l + ka, -b), w_0 = 1$$

Parametrization with Bezier Curves

"Inner" curve



$\mathbf{B}_{\text{inner}}$

$b = a\sqrt{(e^2-1)(k^2-1)}$

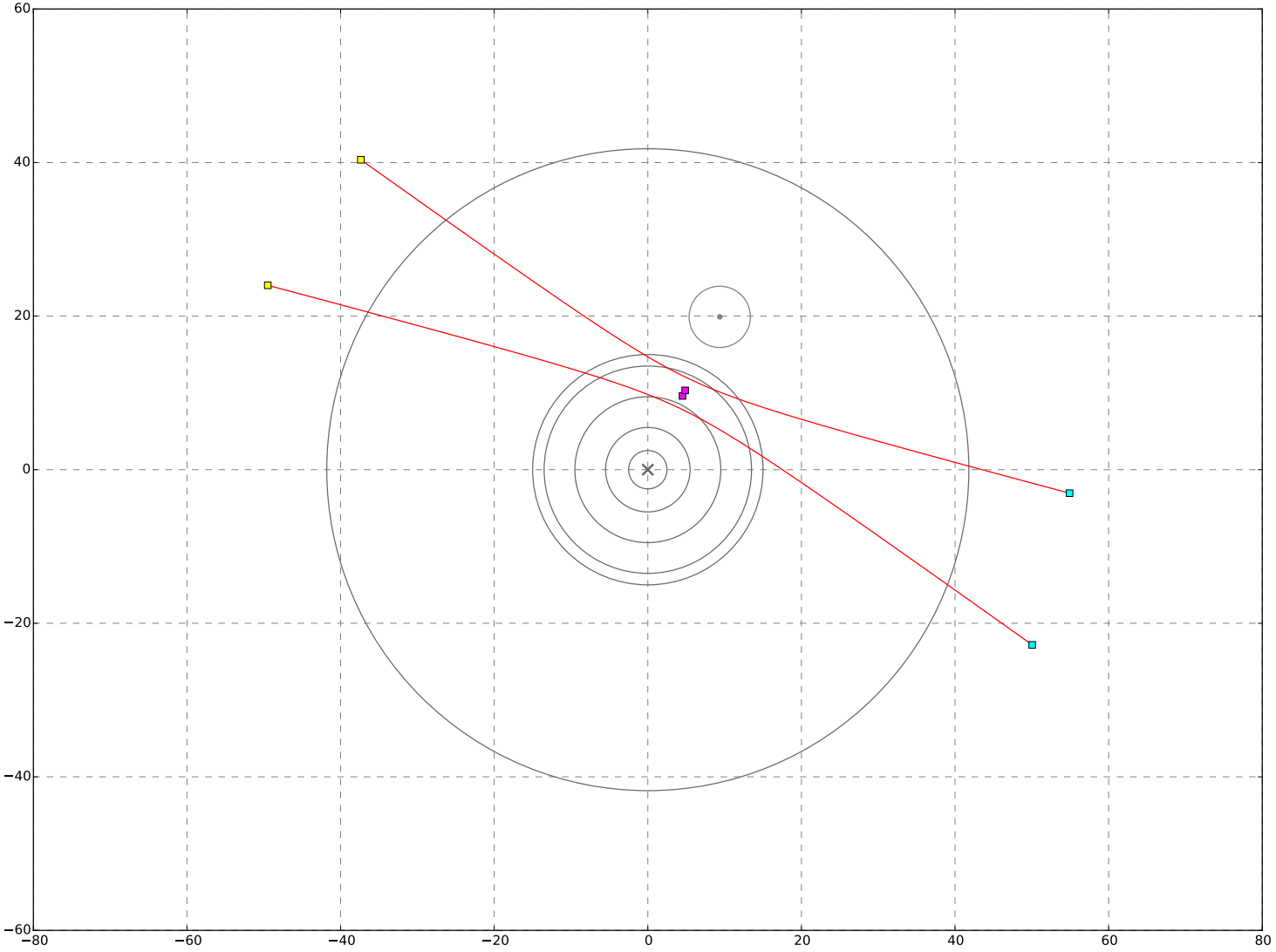
$k = (2R_{\text{max}} + R_{\text{iso}})/\rho_{\text{hit}}$

$\mathbf{P}_0 = (l - ka, b), w_0 = 1$

$\mathbf{P}_1 = (l - \frac{a}{k}, 0), w_1 = k$

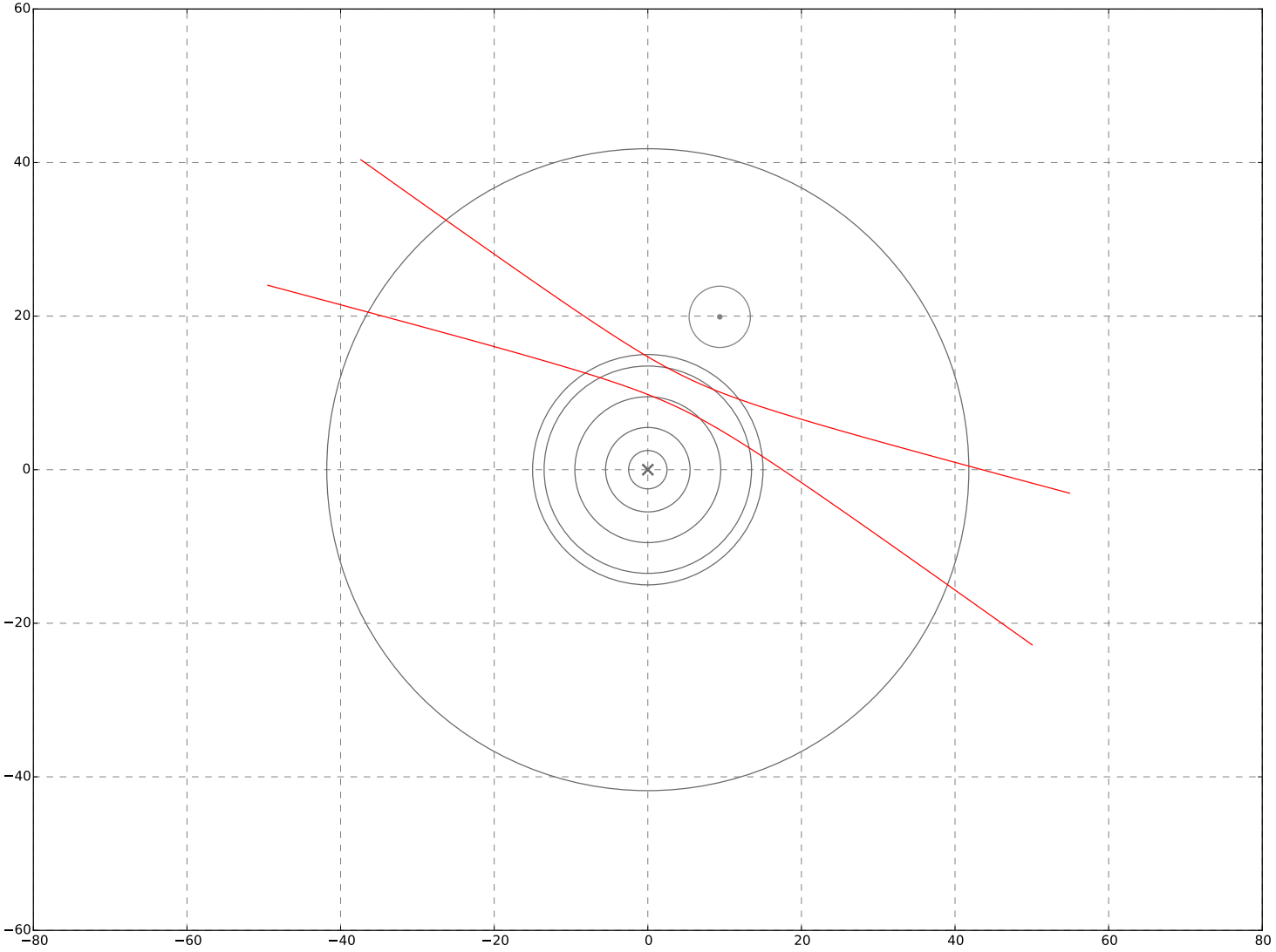
$\mathbf{P}_0 = (l - ka, -b), w_0 = 1$

Parametrization with Bezier Curves

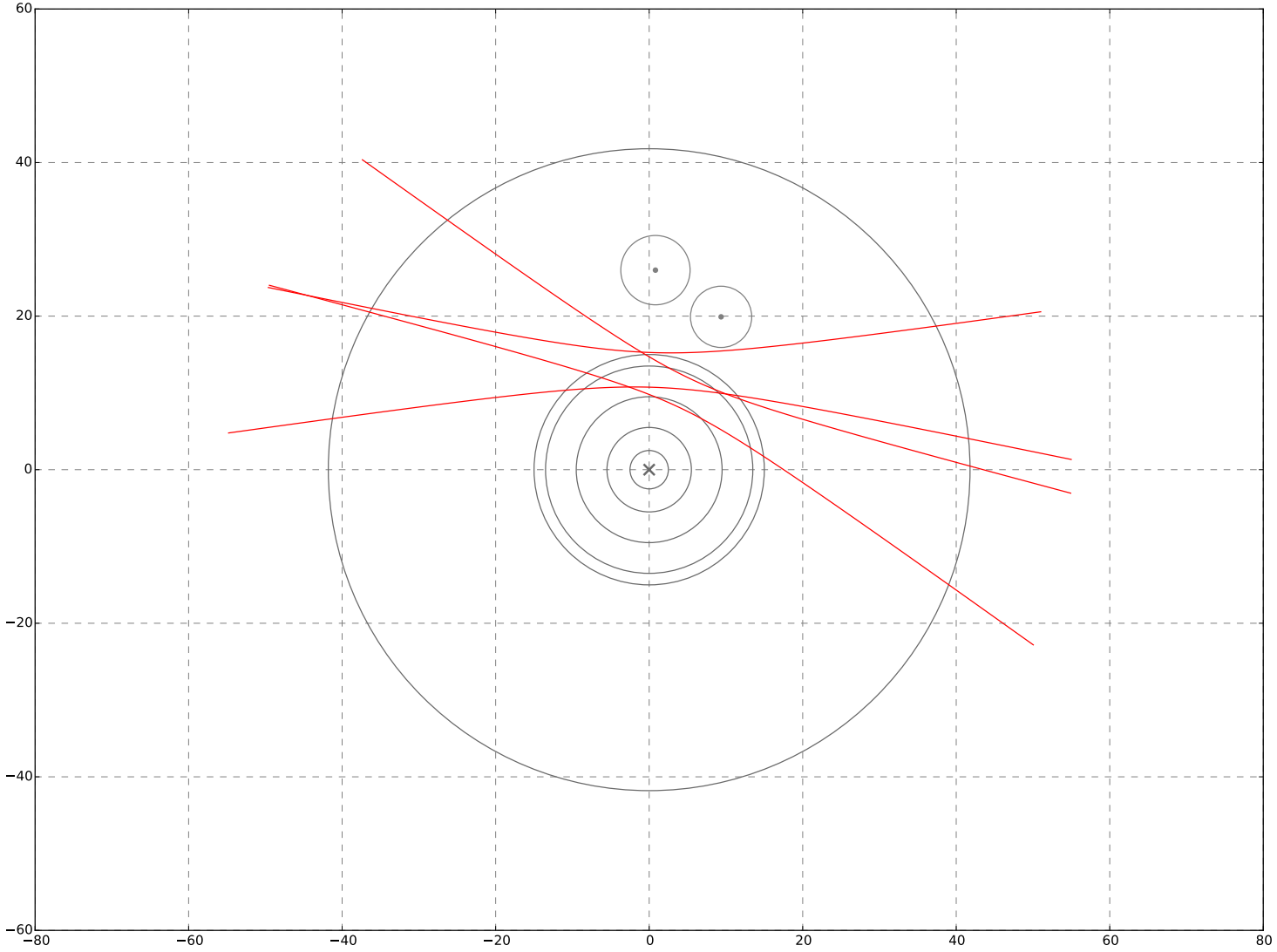


Parametrization with Bezier Curves

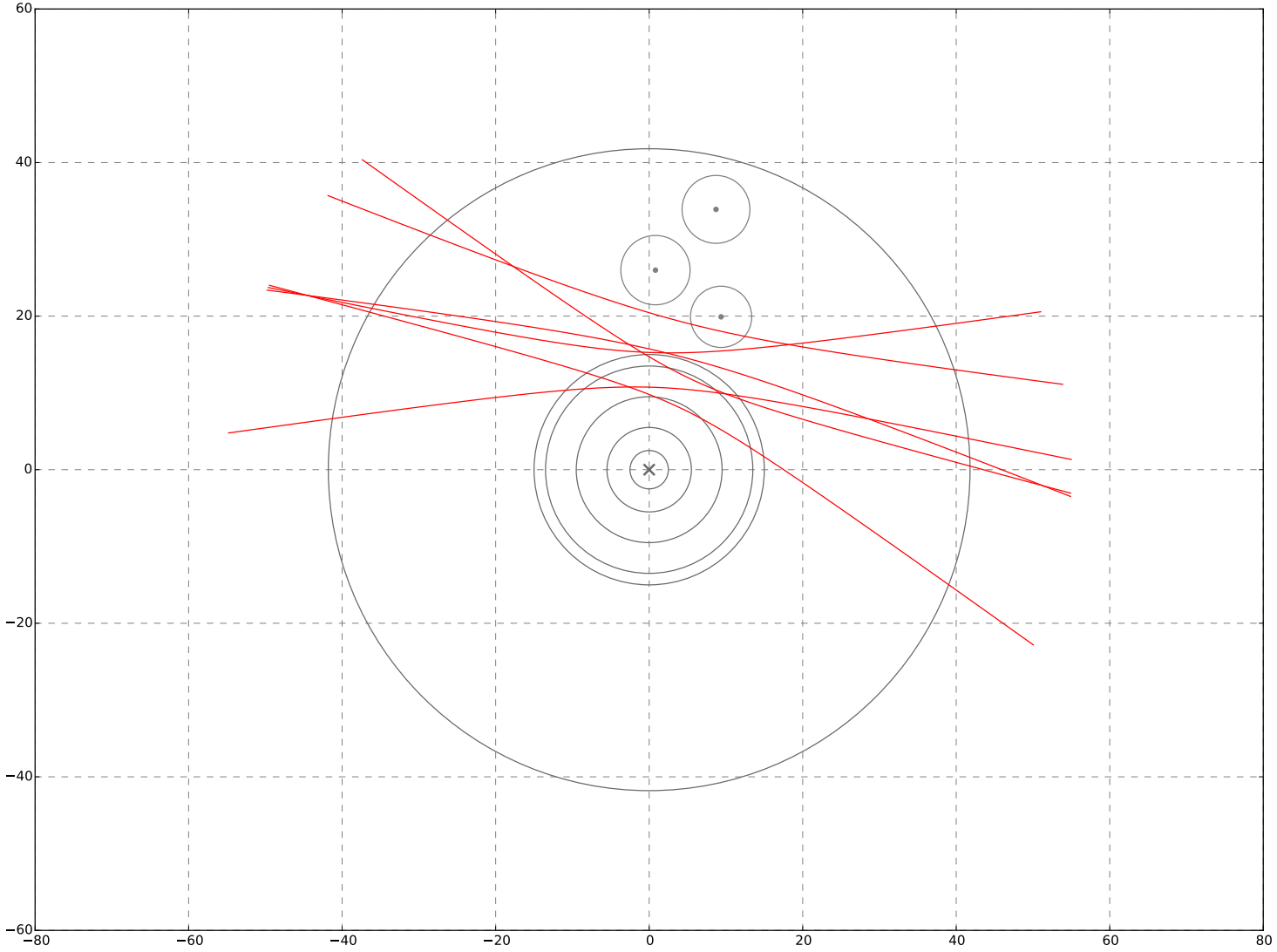
Parametrization with Bezier Curves



Parametrization with Bezier Curves



Parametrization with Bezier Curves



Rasterization

- Draw a figure to a bitmap
- Bresenham algorithm: simple, fast algorithm for rasterization of straight lines
- Adaptable to Bezier curves (directly or via polyline)

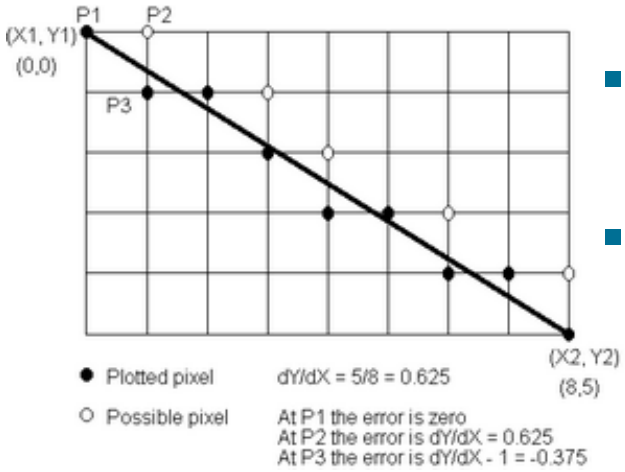
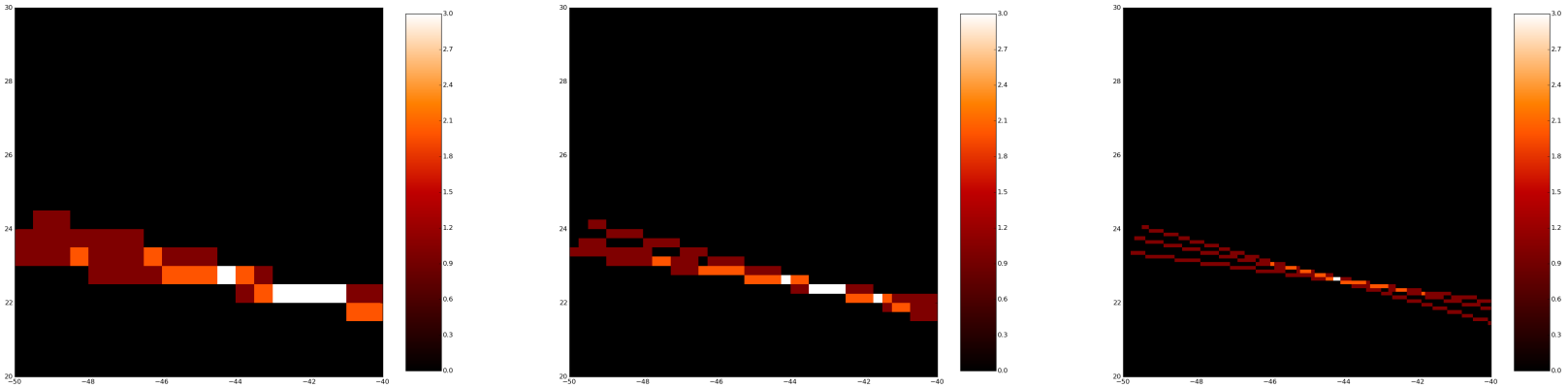


Image credit: http://kobi.nat.uni-magdeburg.de/patrick/pmwiki.php?n=BEng.TheLCDController

- Calculate slope $(y_1 - y_0)/(x_1 - x_0)$
- At each step calculate deviation from ideal line ε
- Only two possible choices:
  - $\varepsilon < 0.5$: increment x, y unchanged
  - $\varepsilon > 0.5$: decrement y instead

Accumulator Array

- Uniform filling of accumulator array, independent of bin density
- Sharper, more regular peaks



Click on the images to download PNG image of the entire frame ($\sim$ 1 MB)

Conclusion & Outlook

- Circle Hough algorithm for trackfinding
  - Compute all tracks compatible with one hit
  - Combine results from *N* hits
  - Find accumulation of track parameters
- Developed alternative approach based on geometric properties of track centers
  - Parametrize locus (hyperbola) using Bezier curves calculated from hit data
  - Rasterize Bezier curves on accumulator array
  - Perform peakfinding on accumulator array
- Work in progress:
  - ☐ Finalize peakfinding
  - ☐ Write optimized implementation CPU, GPU
  - ☐ Physics studies