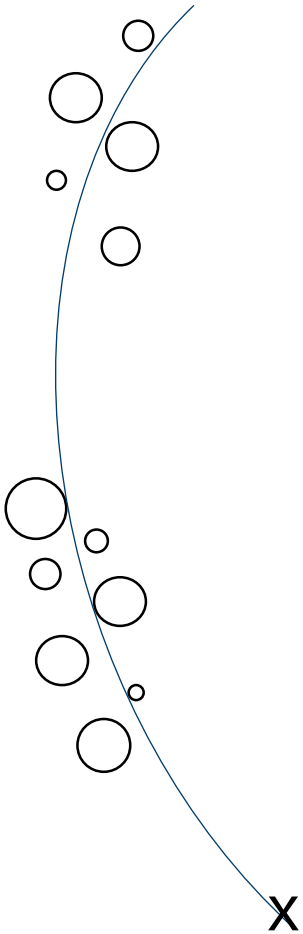
A PANDA TRACK FINDING ALGORITHM BASED ON THE APOLLONIUS PROBLEM

25.06.2019 I ANNA SCHOLL

INTRODUCTION

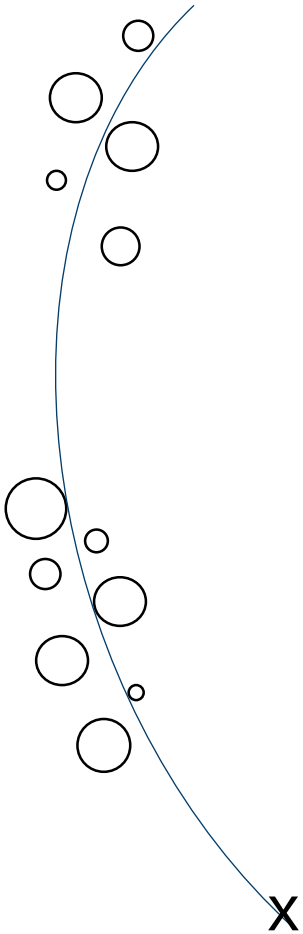


- Focussing on barrel part of the detector
- Use hits from MVD, STT, GEM detector
- Track passes through MVD and GEM hit points
- Track is tangent to STT isochrones

x interaction point (IP)

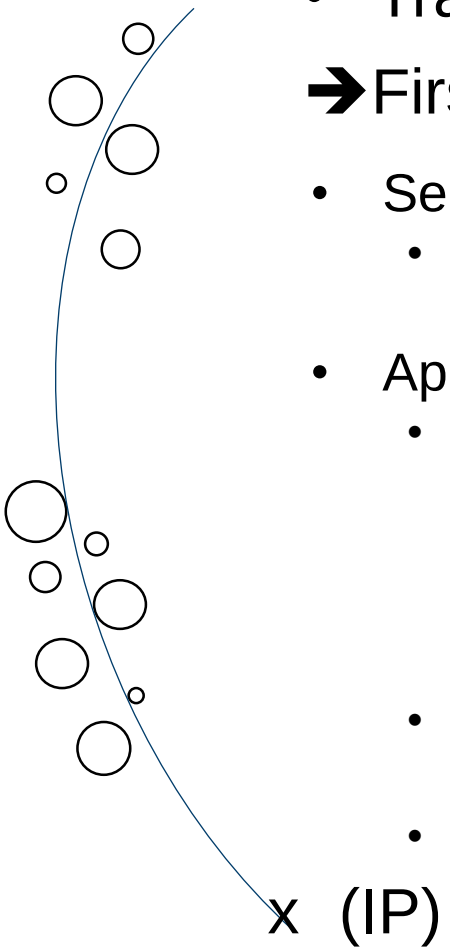
HOW TO INCLUDE ISOCHRONE INFORMATION IN TRACKING ALGORITHMS?

- Track is tangent to the isochrone



x interaction point (IP)

HOW TO INCLUDE ISOCHRONE INFORMATION IN TRACKING ALGORITHMS?



- Track is tangent to the isochrone

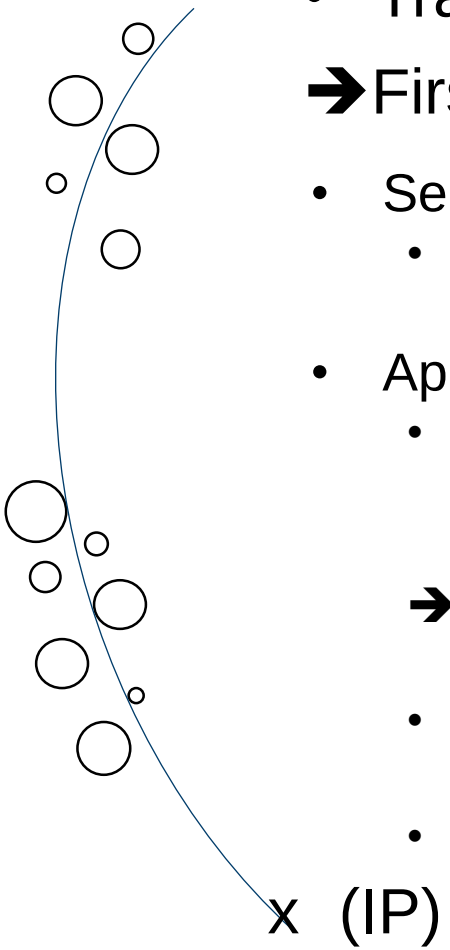
➔First idea: **Hough transformation**

- Separate dimensions
  - 3D helix (R, φ, z) ➔ 2D circle (R, φ) + line (z)

- Apply Hough transform to detect tracks in a set of hits
  - For each hit, generate all possible tracks compatible with it (Circles in xy plane, passing through IP and are tangent to the isochrone)

  - Collect generated track parameters for all hits (2D Hough Space)
  - Count: most frequent values = parameters of actual tracks

HOW TO INCLUDE ISOCHRONE INFORMATION IN TRACKING ALGORITHMS?



- Track is tangent to the isochrone

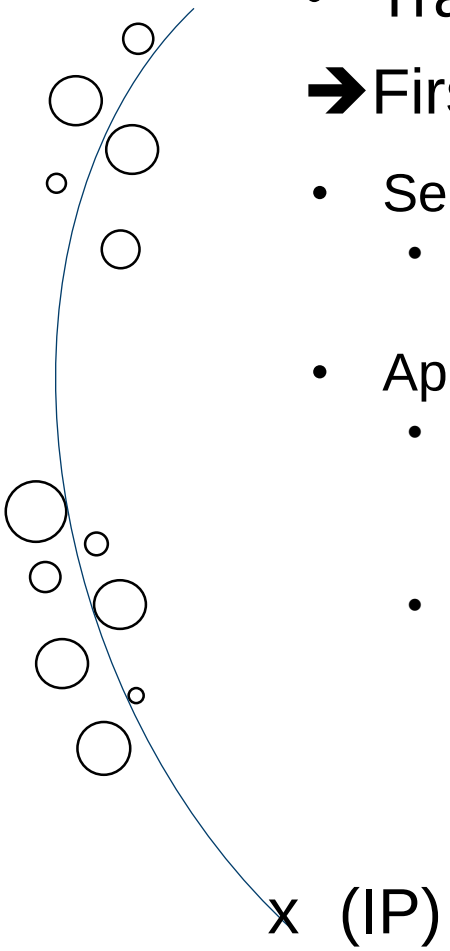
➔First idea: **Hough transformation**

- Separate dimensions
  - 3D helix (R, φ, z) ➔ 2D circle (R, φ) + line (z)

- Apply Hough transform to detect tracks in a set of hits
  - For each hit, generate all possible tracks compatible with it (Circles in xy plane, passing through IP and are tangent to the isochrone)

    ➔ **Problem**: a lot of false combinations for increasing number of tracks per event
  - Collect generated track parameters for all hits (2D Hough Space)
  - Count: most frequent values = parameters of actual tracks

HOW TO INCLUDE ISOCHRONE INFORMATION IN TRACKING ALGORITHMS?



- Track is tangent to the isochrone

➔First idea: **Hough transformation**

- Separate dimensions
  - 3D helix (R, φ, z) ➔ 2D circle (R, φ) + line (z)

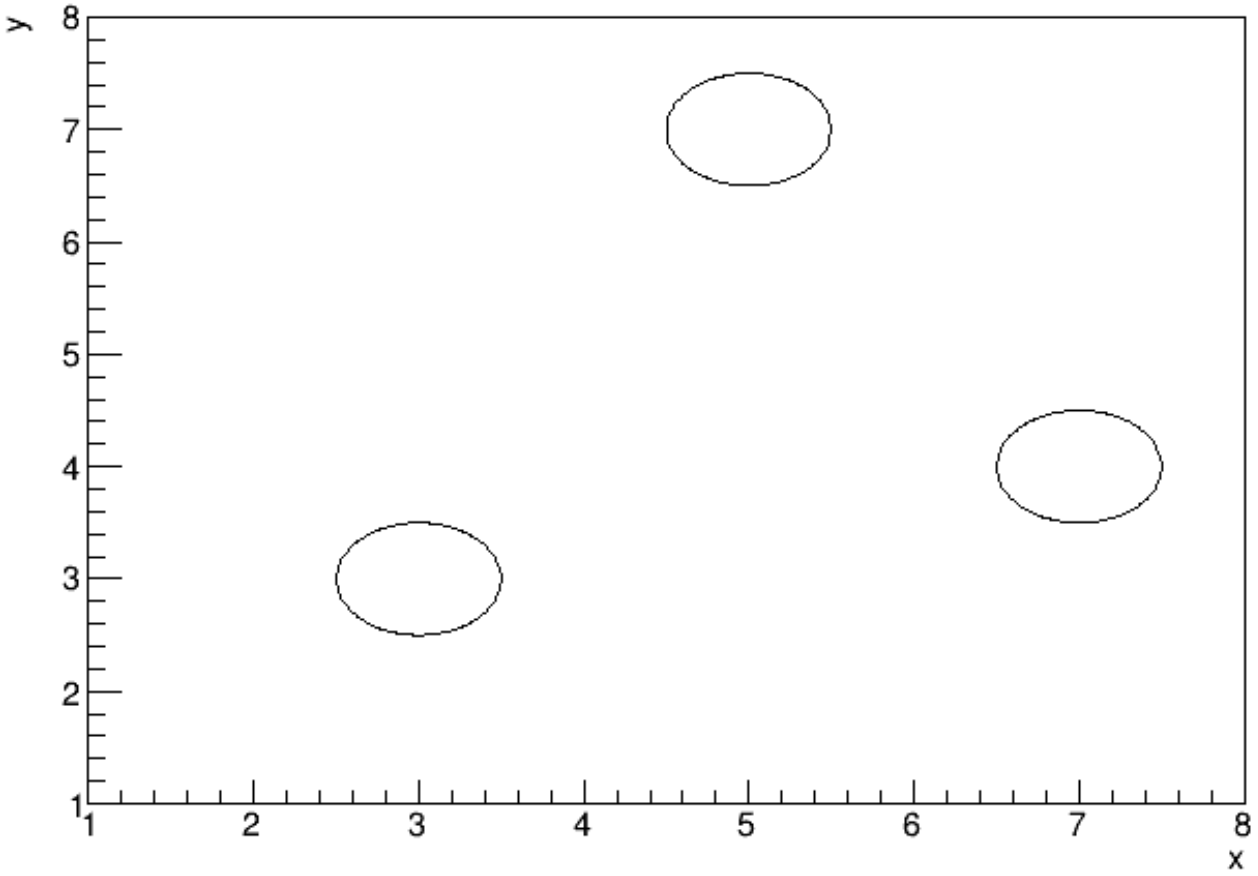
- Apply Hough transform to detect tracks in a set of hits
  - **Problem:** a lot of false combinations for increasing number of tracks per event

  - **Idea:** reduce combinatrics by using 2 Isochrones and IP

➔**problem of Apollonius**

APOLLONIUS PROBLEM

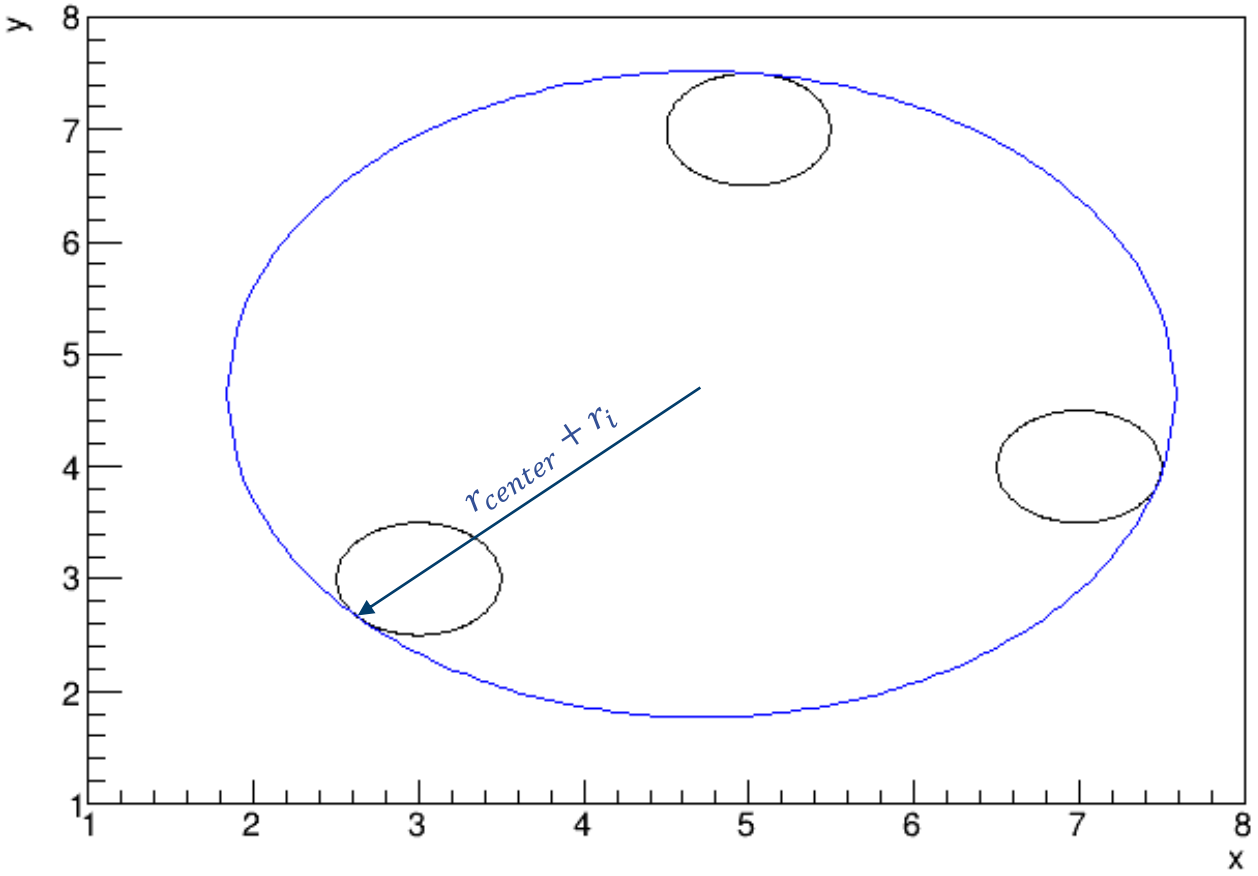
- General Apollonius problem for 3 circles:
  Find circles that are tangent to three given circles in a plane



JÜLICH Forschungszentrum

APOLLONIUS PROBLEM

- General Apollonius problem for 3 circles:
  Find circles that are tangent to three given circles in a plane

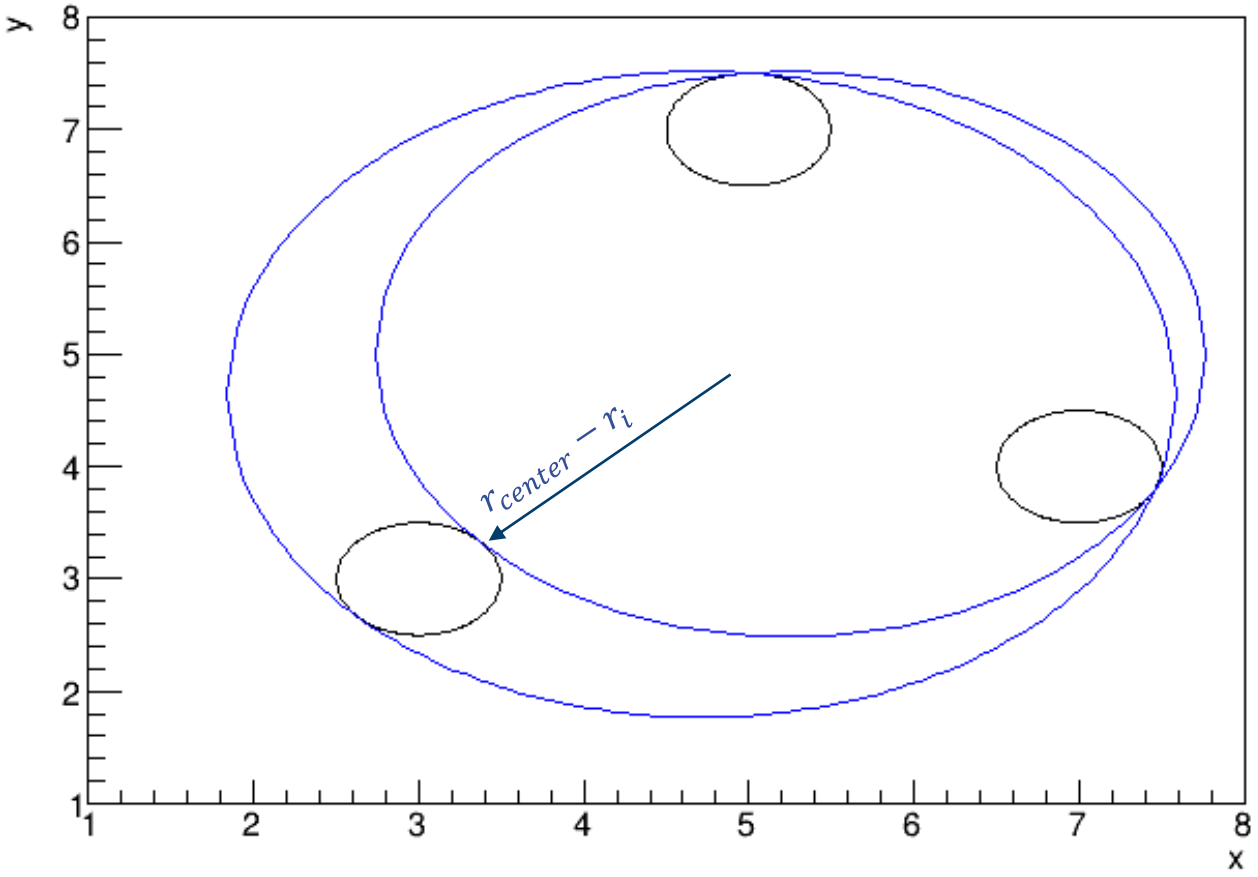


For each circle there are 2 possibilities for an Apollonius circle

1. $r_{Apollonius} = r_{center} + r_i$

APOLLONIUS PROBLEM

- General Apollonius problem for 3 circles:
  Find circles that are tangent to three given circles in a plane



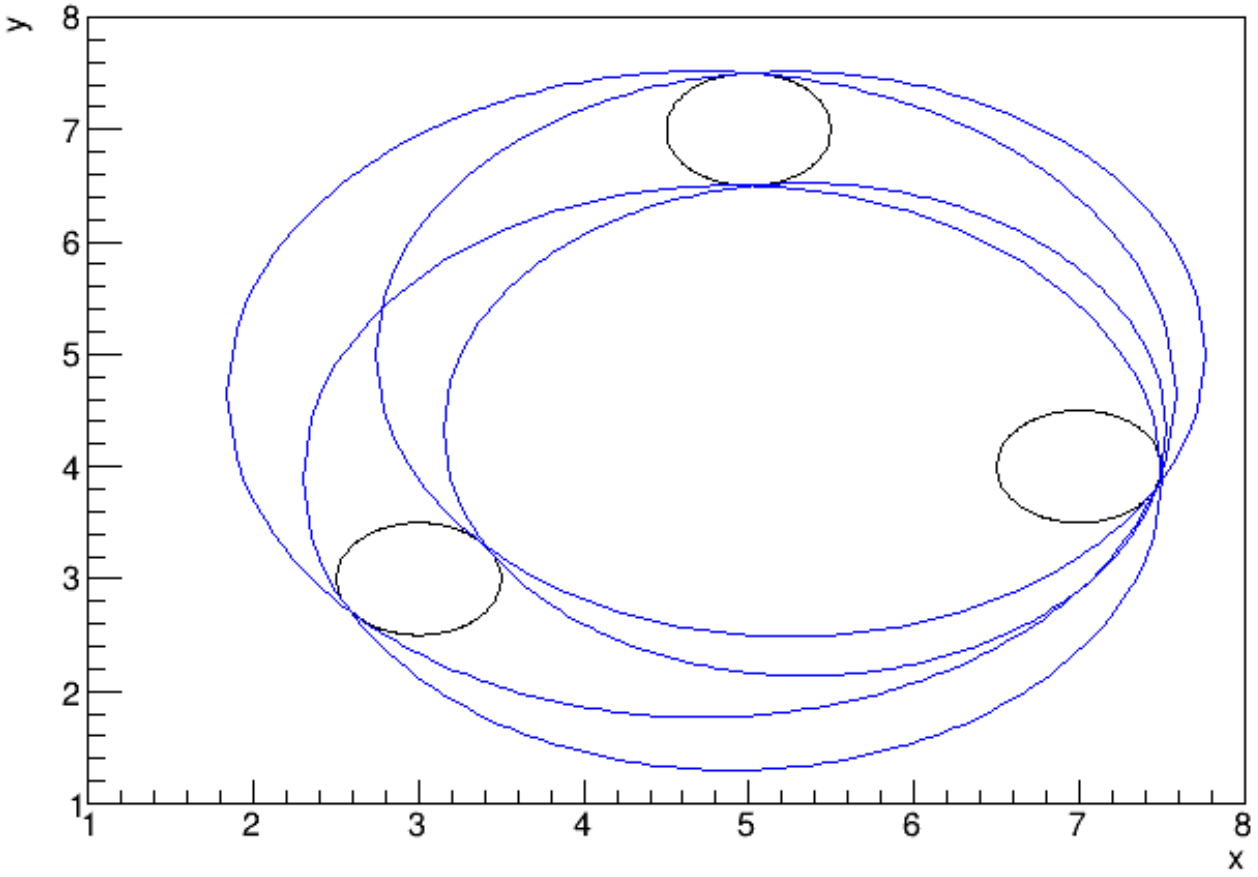
For each circle there are 2 possibilities for an Apollonius circle

1. $r_{Apollonius} = r_{center} + r_i$
2. $r_{Apollonius} = r_{center} - r_i$

JÜLICH Forschungszentrum

APOLLONIUS PROBLEM

- General Apollonius problem for 3 circles:
  Find circles that are tangent to three given circles in a plane

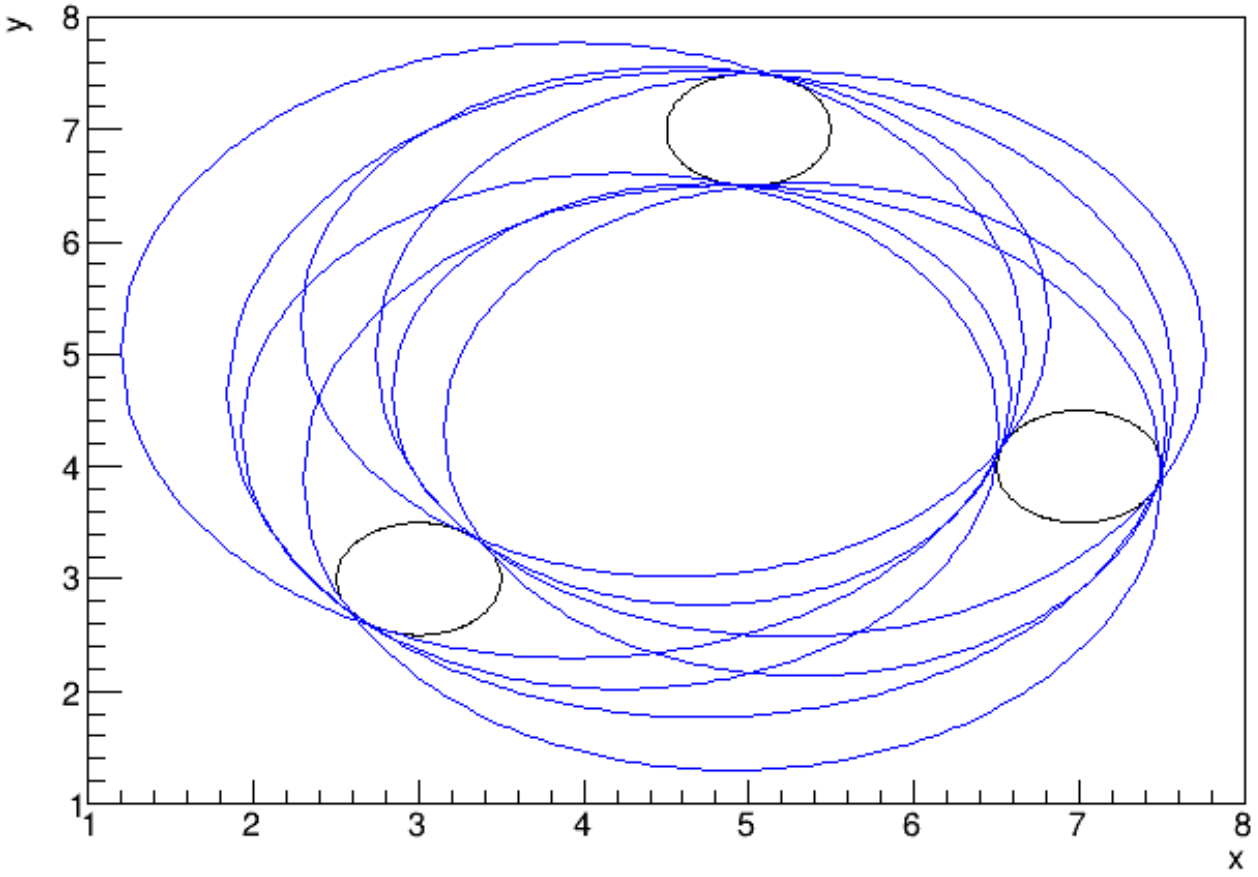


For each circle there are 2 possibilities for an Apollonius circle

1. $r_{Apollonius} = r_{center} + r_i$
2. $r_{Apollonius} = r_{center} - r_i$

APOLLONIUS PROBLEM

- General Apollonius problem for 3 circles:
  Find circles that are tangent to three given circles in a plane



For each circle there are 2 possibilities for an Apollonius circle

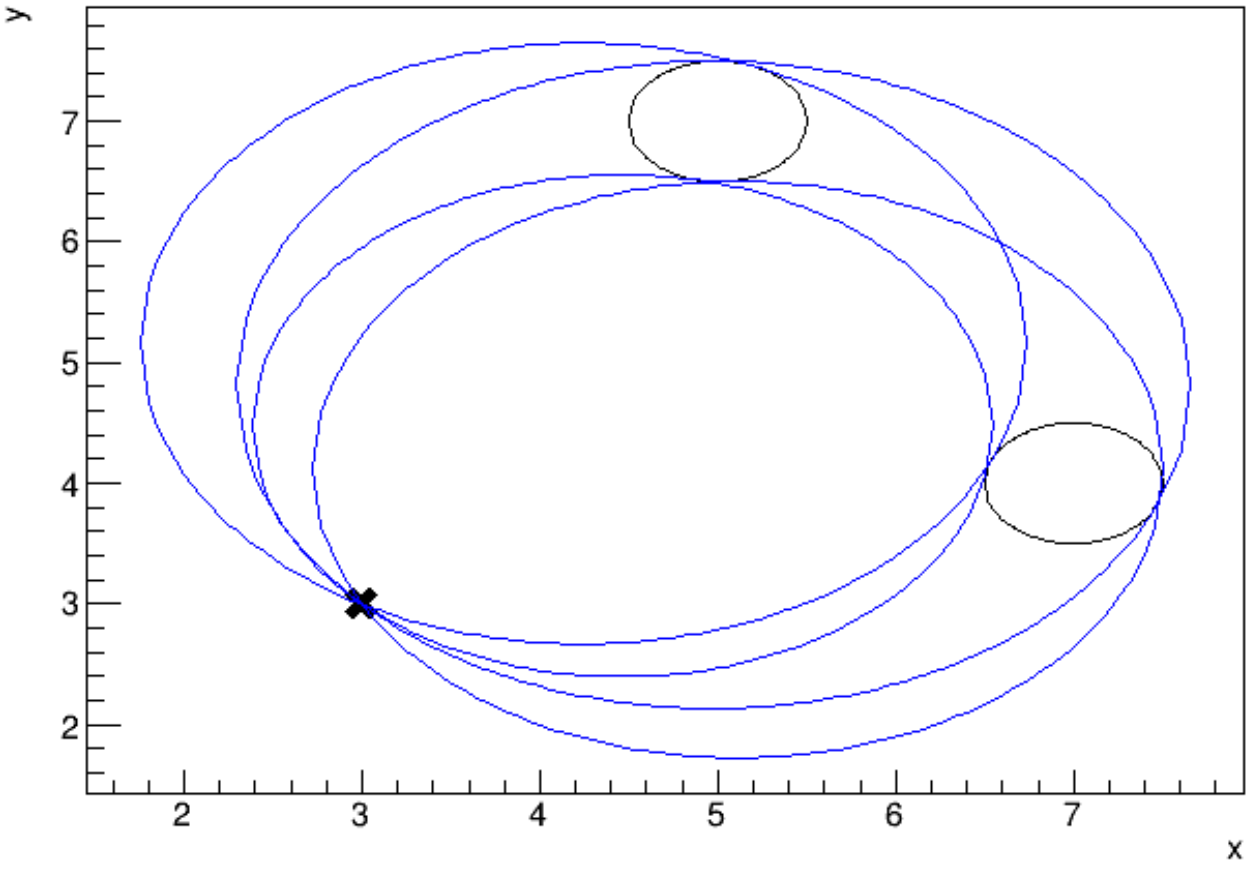
1. $r_{Apollonius} = r_{center} + r_i$
2. $r_{Apollonius} = r_{center} - r_i$

In total $2^3 = 8$ Apollonius circles

JÜLICH Forschungszentrum

APOLLONIUS PROBLEM

- Special case: two circles and one point



For each circle there are 2 possibilities for an Apollonius circle

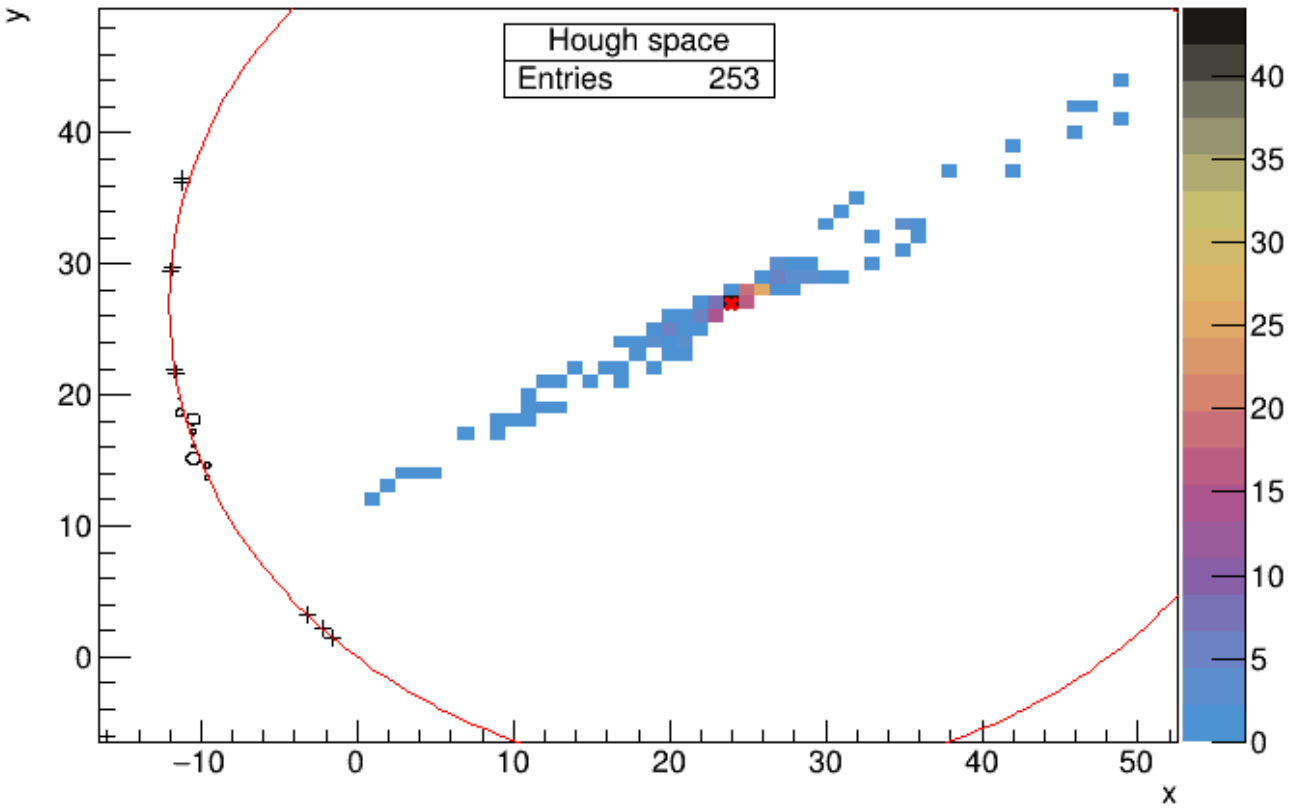
1. $r_{Apollonius} = r_{center} + r_i$
2. $r_{Apollonius} = r_{center} - r_i$

In total $2^2 = 4$ Apollonius circles

HOUGH TRANSFORMATION BASED ON APOLLONIUS PROBLEM

Implemetation in PandaRoot and testing with simulated data

Example for one Track



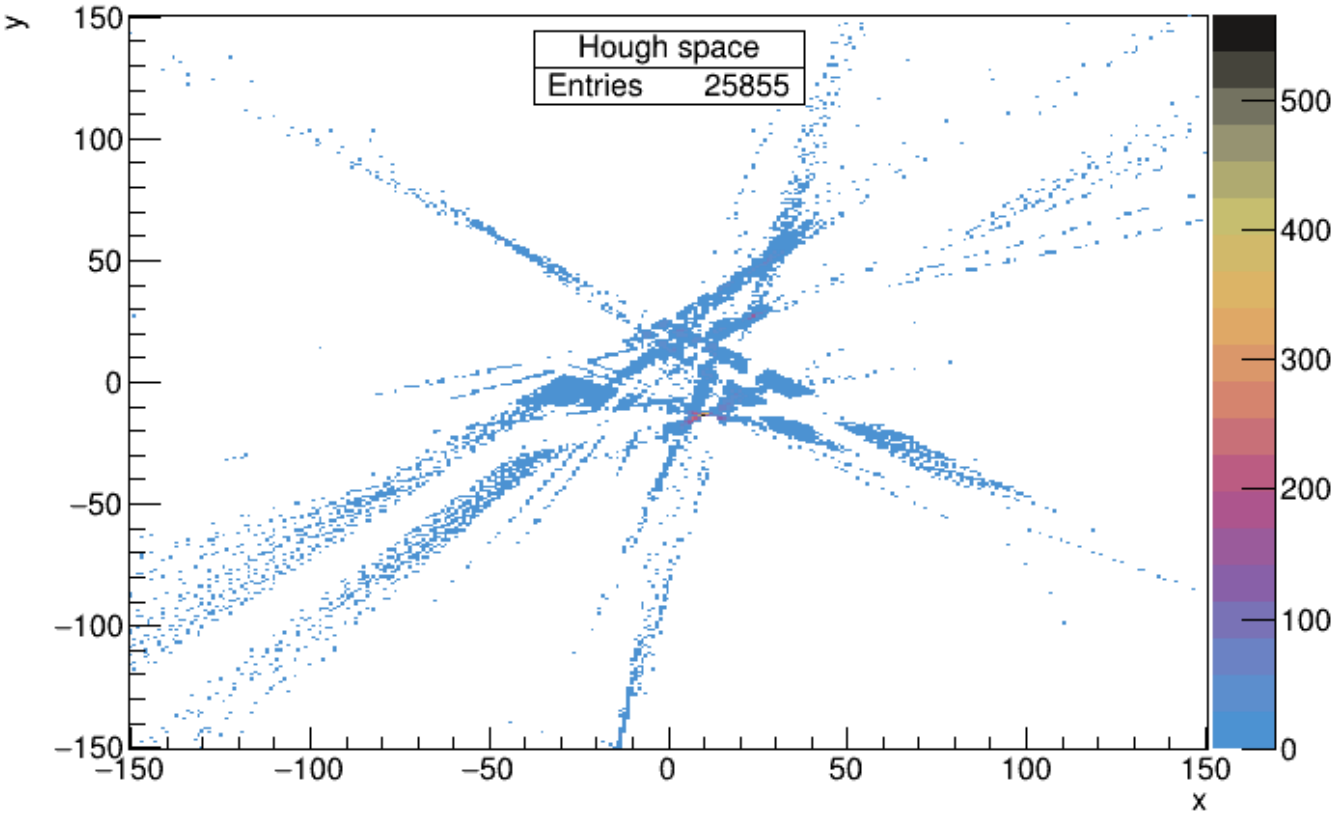
- Works quiet well if track candidate is known (IdealTrackFinder)

JÜLICH Forschungszentrum

HOUGH TRANSFORMATION BASED ON APOLLONIUS PROBLEM

Implemetation in PandaRoot and testing with simulated data

Example for one Event



- Works quiet well if track candidate is known (IdealTrackFinder)
- For one event (many tracks): high combinatorics with (still) many false combinations

JÜLICH Forschungszentrum

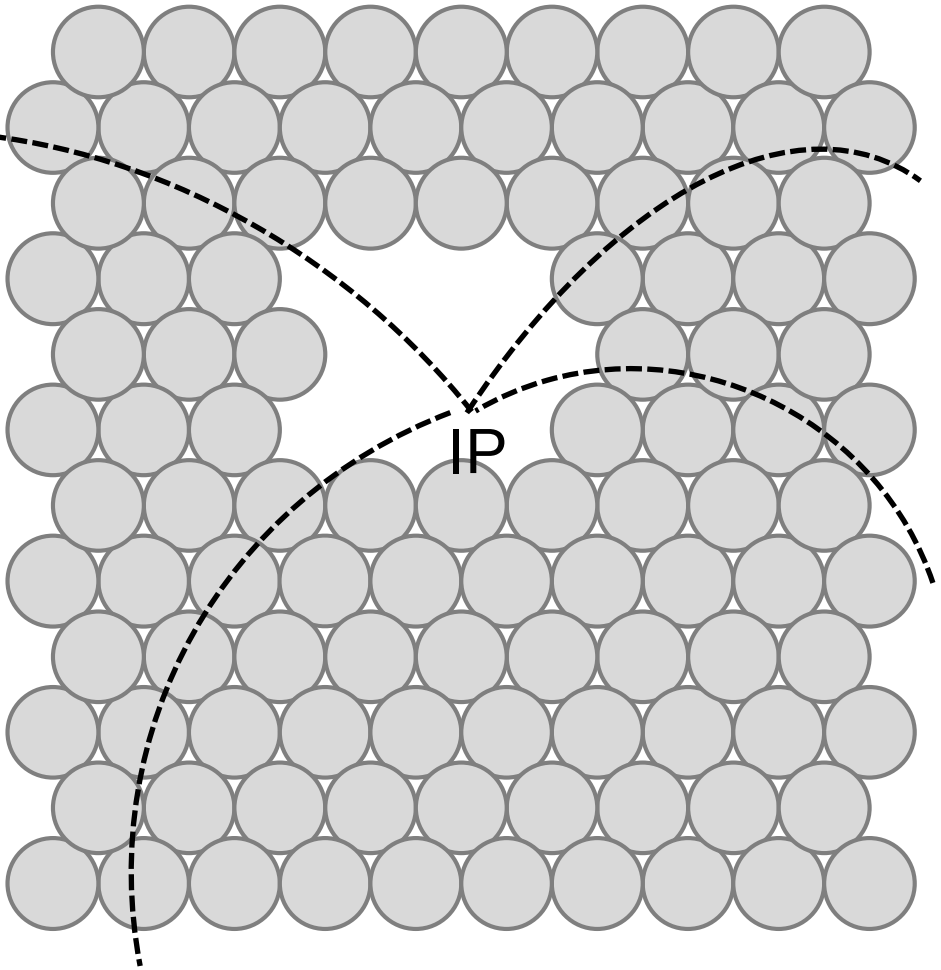
PRESELECTION

- In reality track candidates are not known
- Using Apollonius transform for all tracks in one event is very time consuming and leads to a lot of false combinations

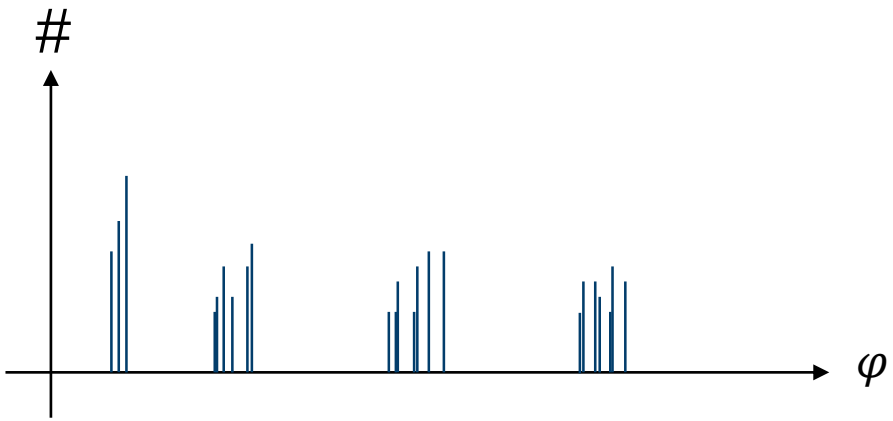
➔ preselection for possible tracklets is needed

- Idea:
  - preselection by cutting on φ-plane: dividing x-y-plane dynamicly in sectors

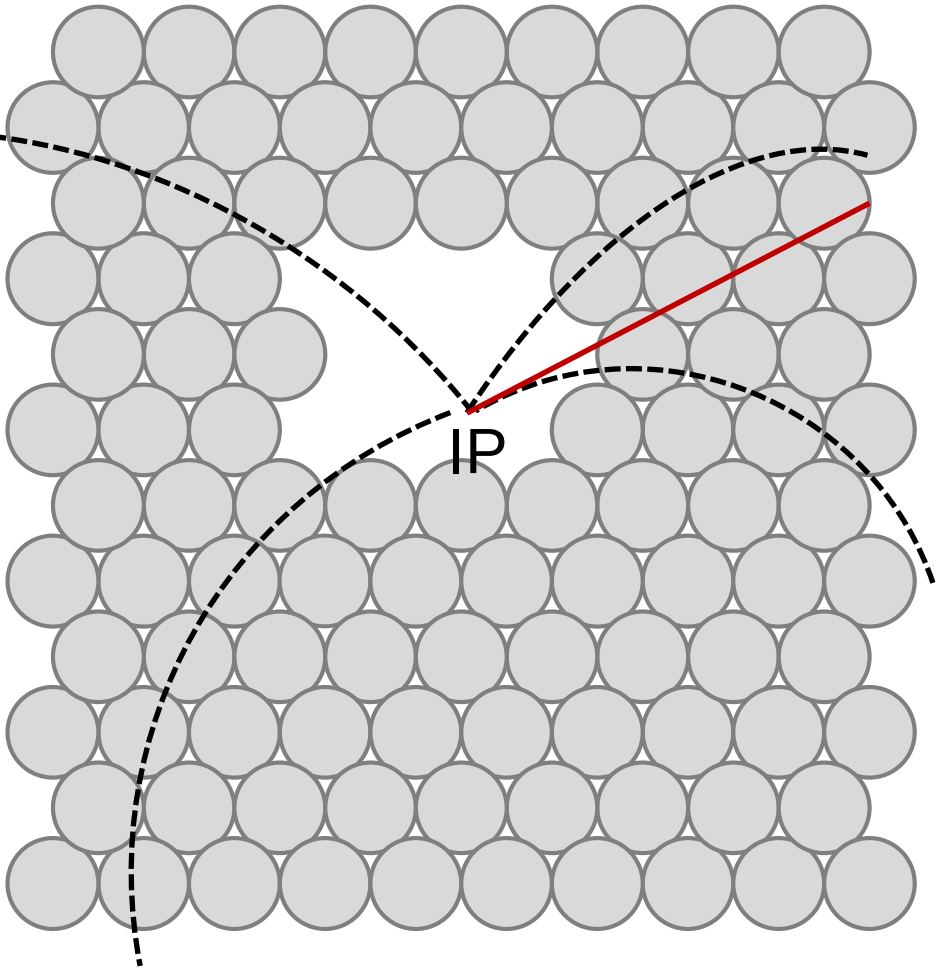
CUT ON φ-PLANE



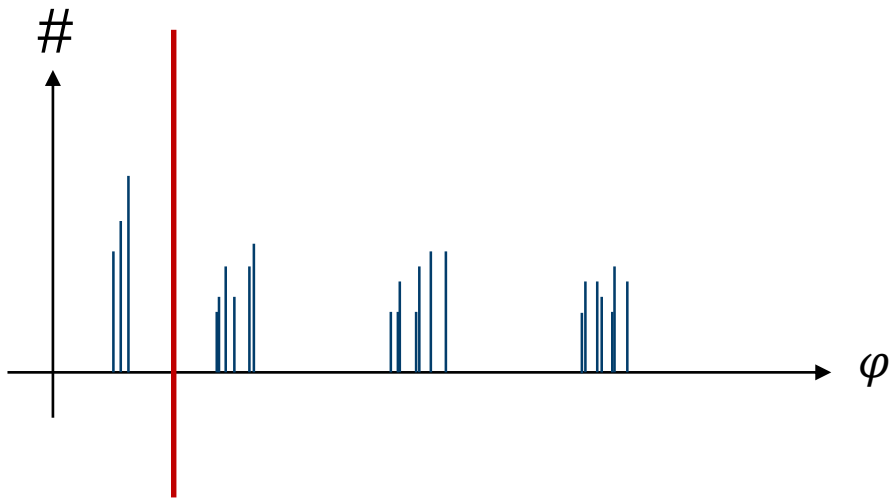
- Filling φ-values of all hits into histogram:



CUT ON φ-PLANE

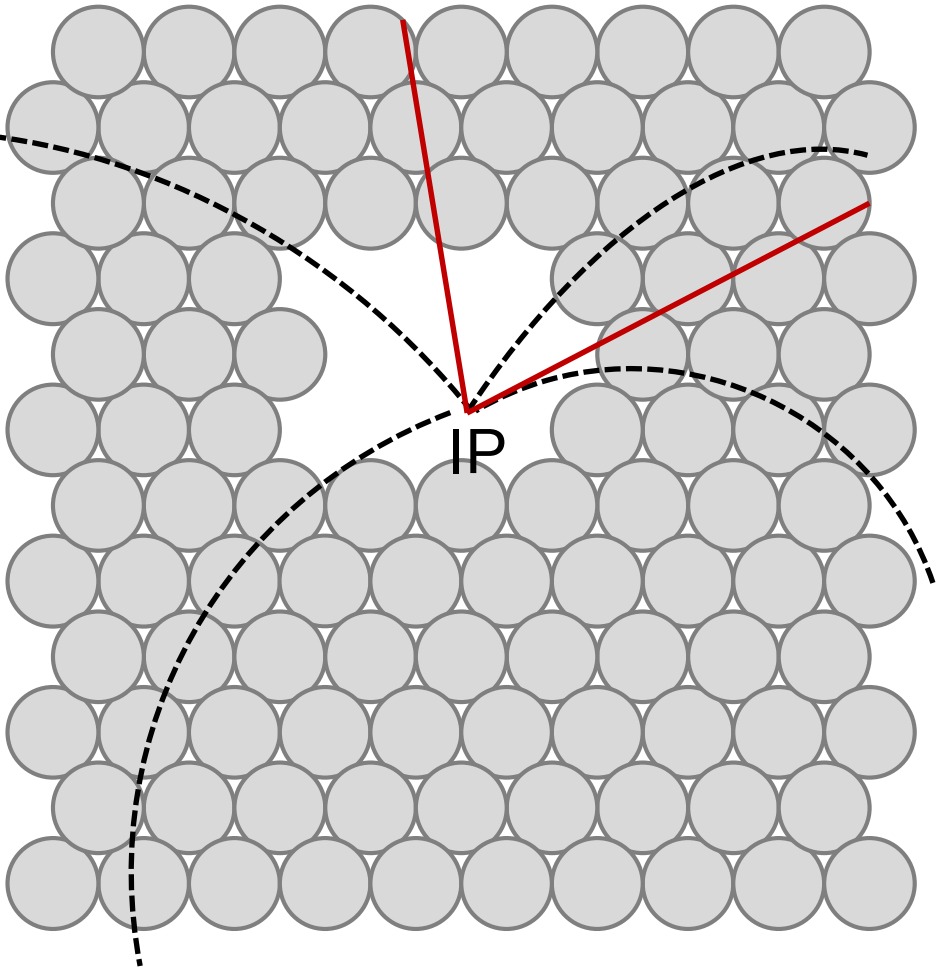


- Filling φ-values of all hits into histogram:

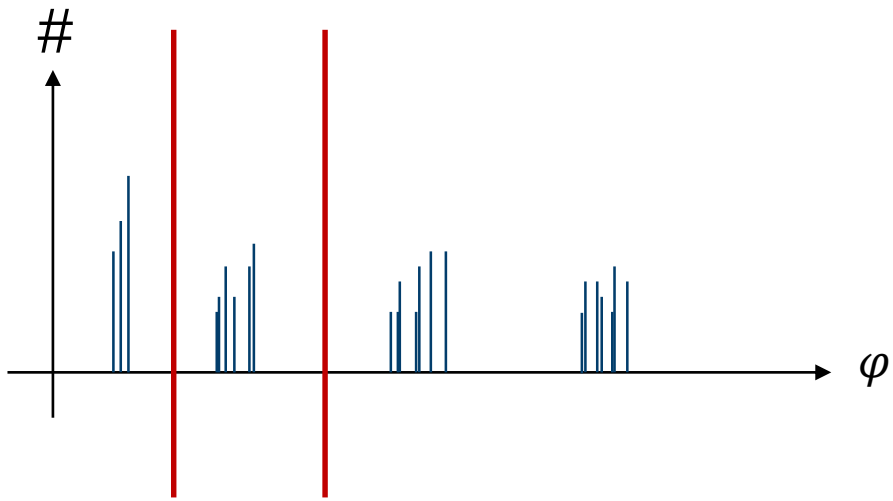


- Divide in φ - sectors

CUT ON φ-PLANE

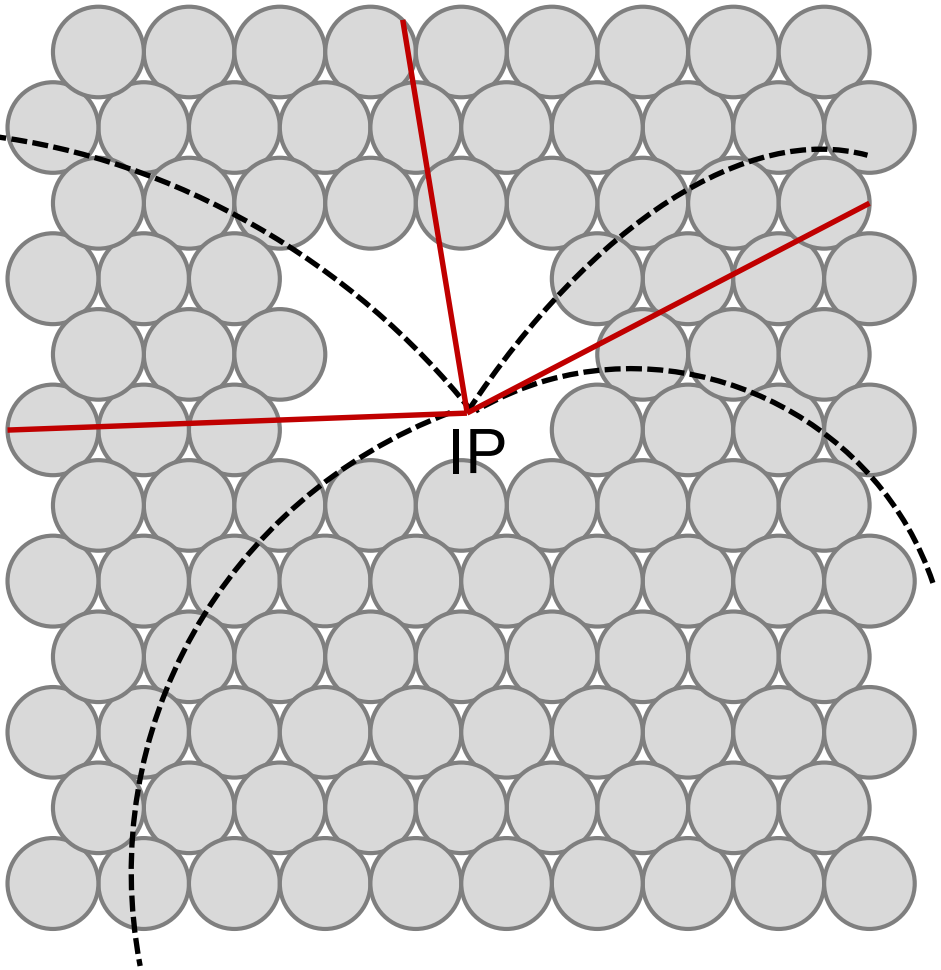


- Filling φ-values of all hits into histogram:

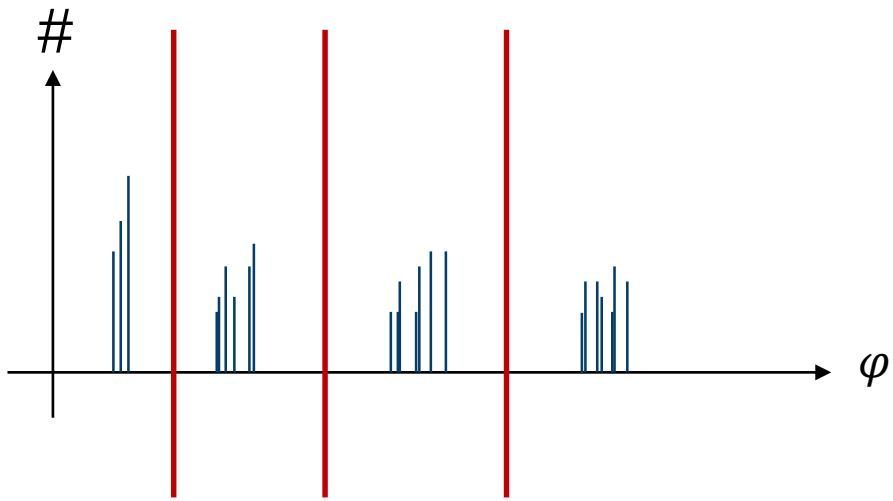


- Divide in φ - sectors

CUT ON φ-PLANE

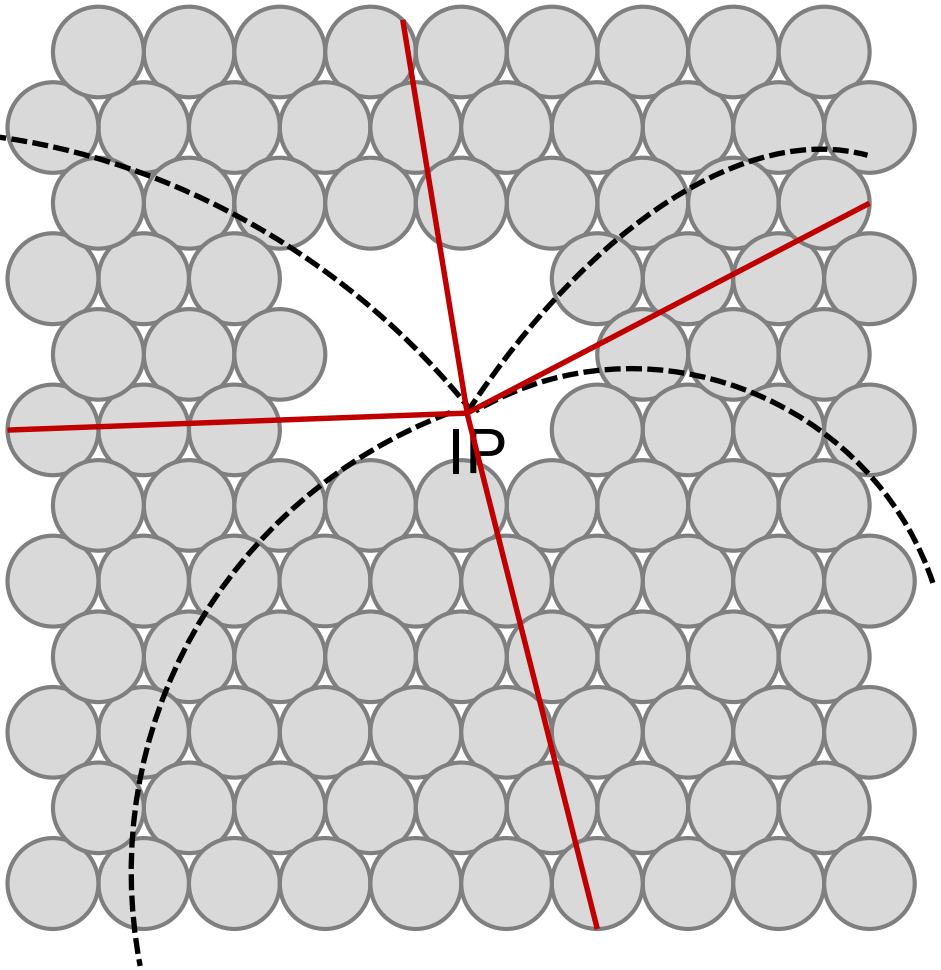


- Filling φ-values of all hits into histogram:

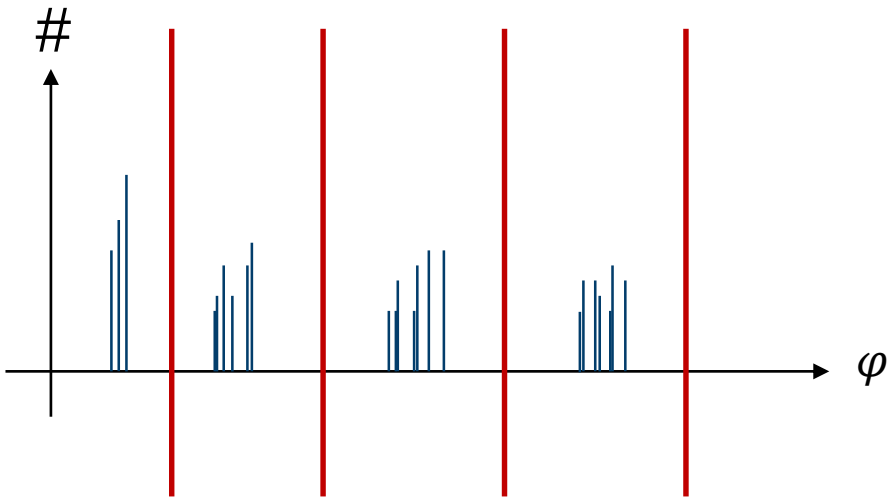


- Divide in φ - sectors

CUT ON φ-PLANE

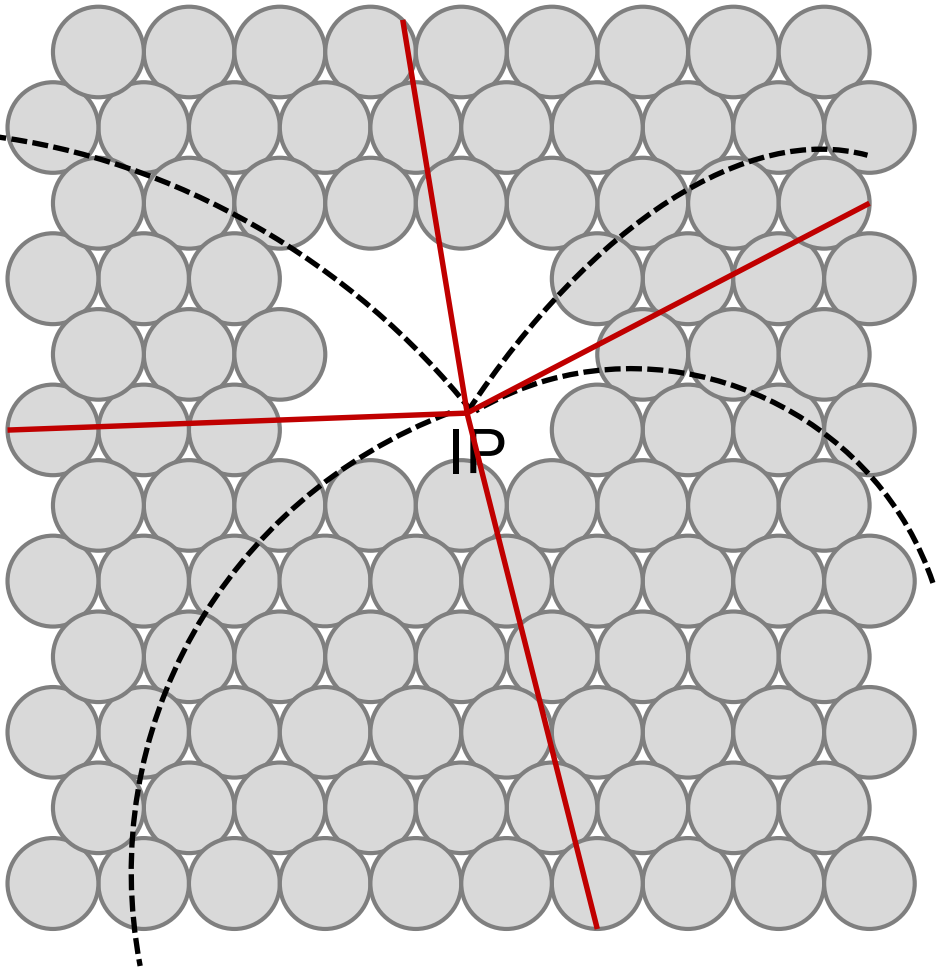


- Filling φ-values of all hits into a histogram:

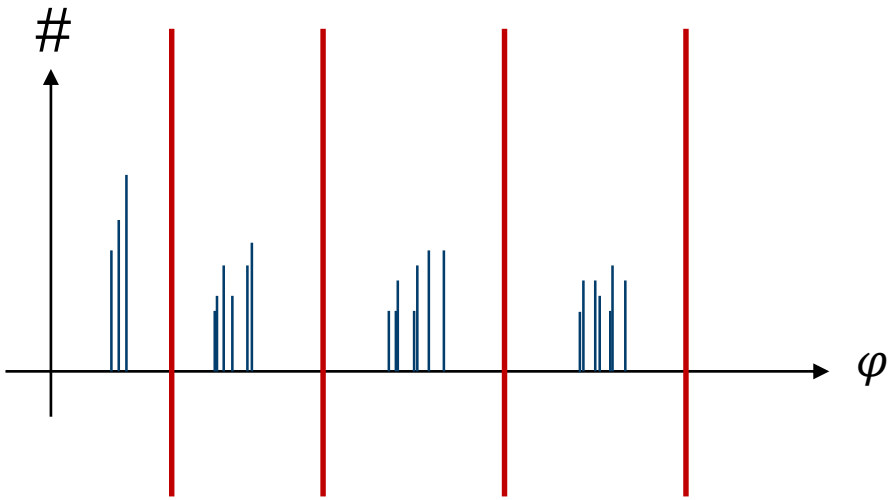


- Divide in φ - sectors
- Hough transformation for all hits in one sector

CUT ON φ-PLANE



- Filling φ-values of all hits into a histogram:



- Divide in φ - sectors
- Hough transformation for all hits in one sector
- Could be problematic for low p_T -Tracks

CONCLUSION

- Implementation of Hough transformation based on the Apollonius problem in PandaRoot
- Testing with single tracks found by the Ideal Track Finder
- High combinatorics for many tracks per events
- Idea for preselection of track candidates:
  - Cut on φ - plane