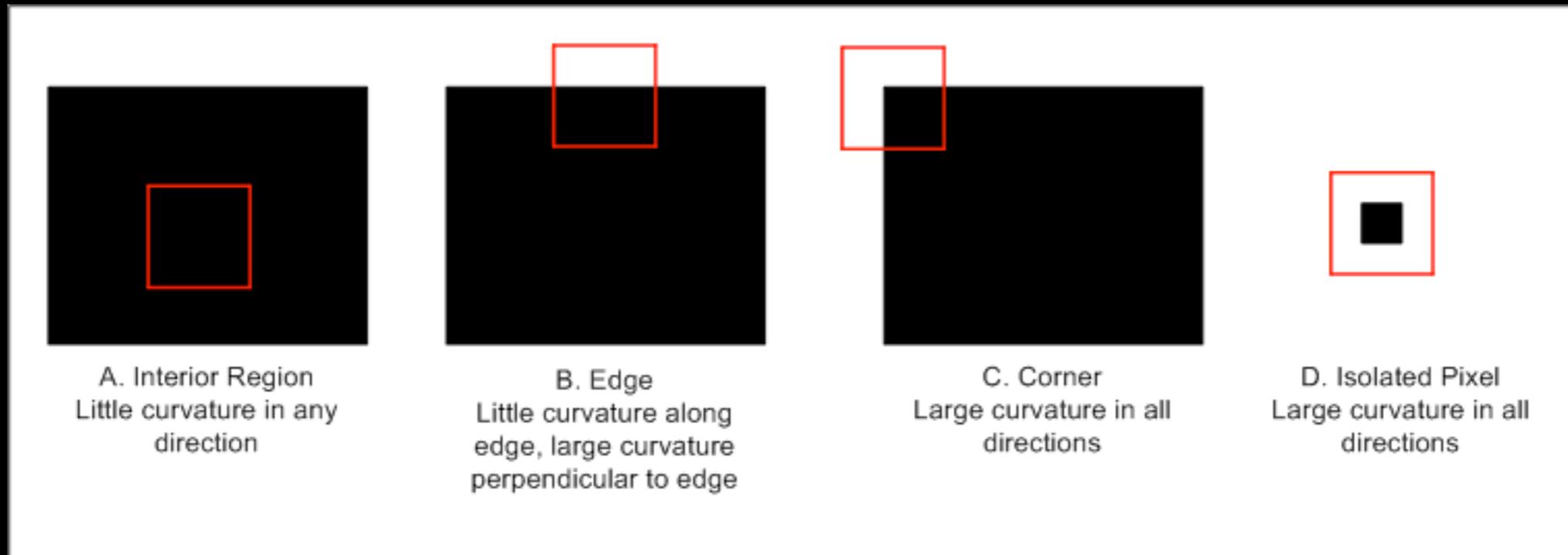


# Vertex finding update

Joshua Spitz

6/24/2010

# The Harris corner detector\*



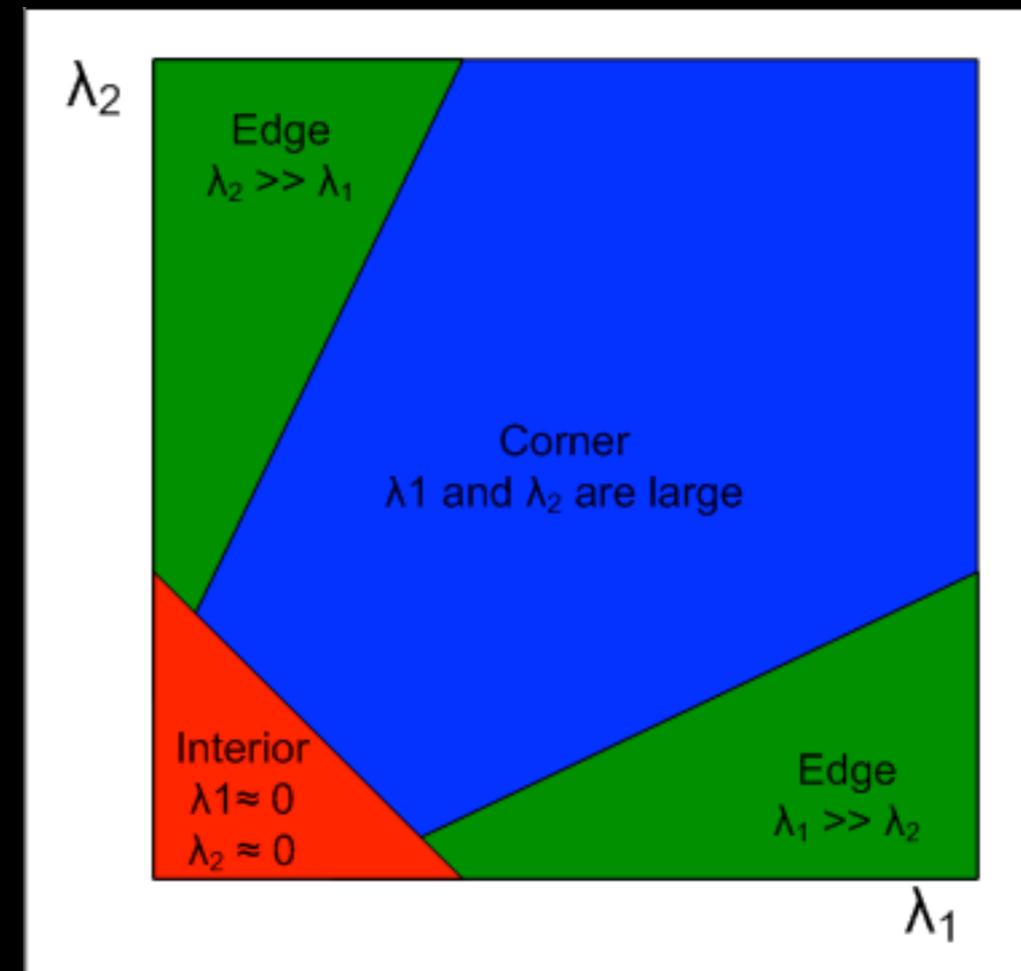
$$H(x, y) = (\Delta x \Delta y) A \begin{pmatrix} \Delta x \\ \Delta y \end{pmatrix}$$

$$A = w(x, y) \begin{pmatrix} (\partial_x)^2 & (\partial_x)(\partial_y) \\ (\partial_x)(\partial_y) & (\partial_y)^2 \end{pmatrix}$$

$\Delta x$ =shift in x     $\partial_x$ =intensity variation in x direction

$w(x, y)$ =2D Gaussian weight     $\lambda$ =eigenvalues of A

\*Motivated by U. Warwick group



# Recent upgrades to HarrisVertexFinder

- A Gaussian derivative is now used instead of the Prewitt operator to determine each pixel's gradient, necessary to determine corneriness.

Sensitive to noise and not rotationally invariant!

↑  
Prewitt operator in x-direction

$$\partial_x \sim \begin{pmatrix} -1 & 0 & 1 \end{pmatrix}$$

Gaussian derivative ( $\sigma=1$ , arbitrary normalization) in x-direction

$$\partial_x \sim \begin{bmatrix} -.0016 & -.0065 & 0 & .0065 & .0016 \\ -.0130 & -.0547 & 0 & .0547 & .0130 \\ -.0266 & -.1109 & 0 & .1109 & .0266 \\ -.0130 & -.0547 & 0 & .0547 & .0130 \\ -.0016 & -.0065 & 0 & .0065 & .0016 \end{bmatrix}$$

$$A = w(x, y) \begin{pmatrix} (\partial_x)^2 & (\partial_x)(\partial_y) \\ (\partial_x)(\partial_y) & (\partial_y)^2 \end{pmatrix}$$

↙  
~Corneriness

$$\begin{bmatrix} 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 \end{bmatrix}$$

Low corneriness

$$\begin{bmatrix} 1 & 1 & 1 & 1 & 1 \\ 1 & 1 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

Low corneriness

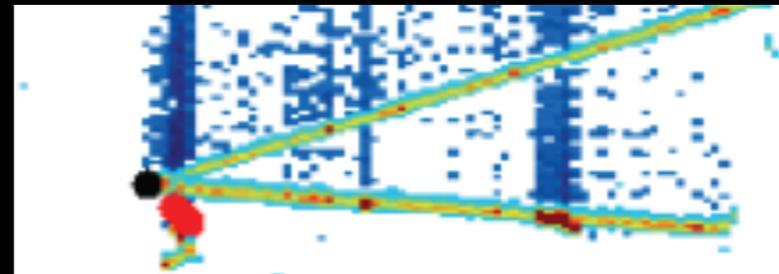
$$\begin{bmatrix} 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

High corneriness

# More upgrades

- Non-maximal suppression added.
- If many potential vertices are found close to each other, the vertex with the largest strength is kept and the others are dismissed.

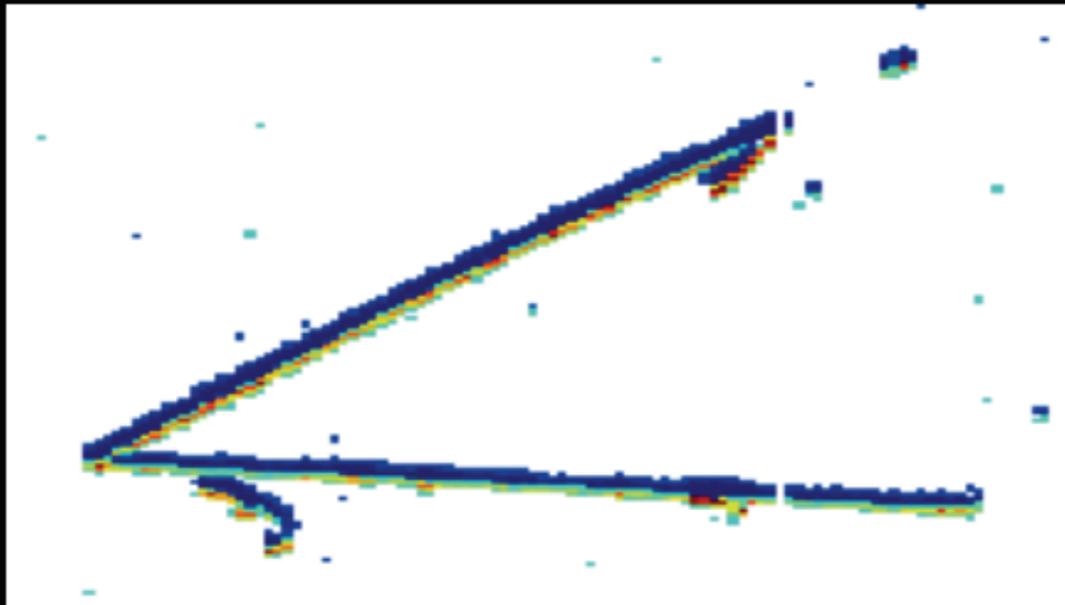
An example of many found vertices close together. Non-maximal suppression chooses the strongest vertex to keep within a user-specified window.



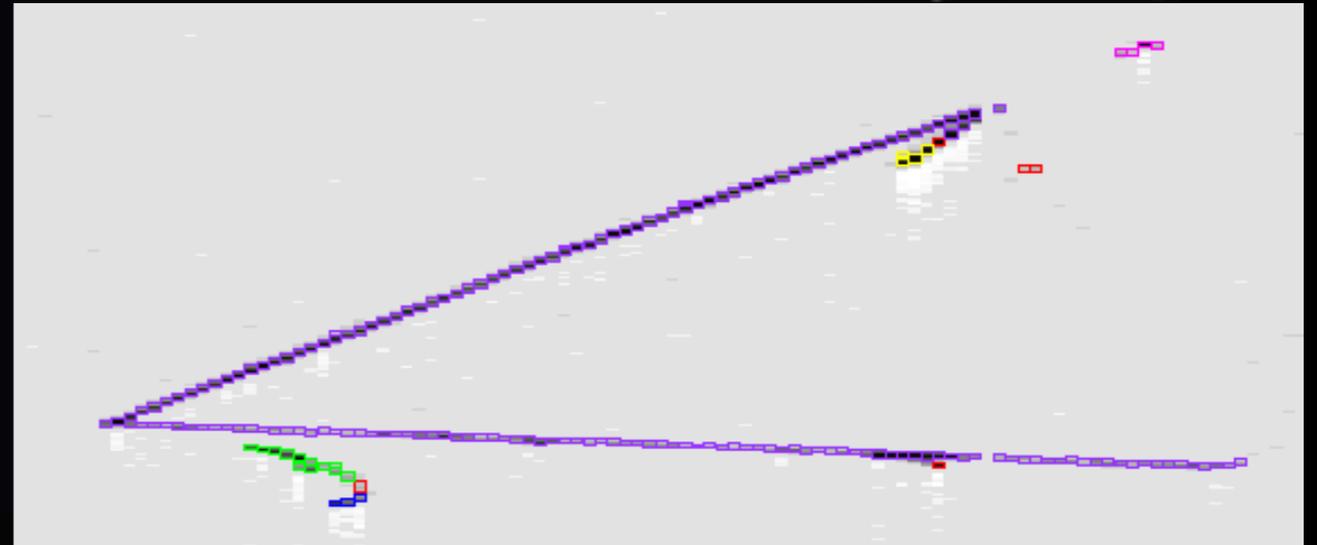
- Each “weak” vertex found with HarrisVertexFinder is now given a quantitative vertex strength.

# The Corner finding input

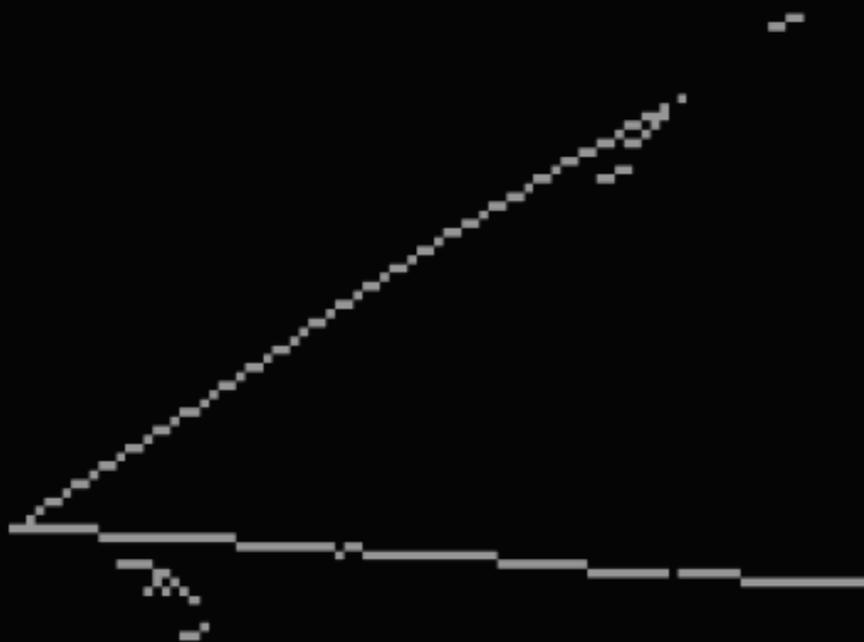
Step 1. Raw data



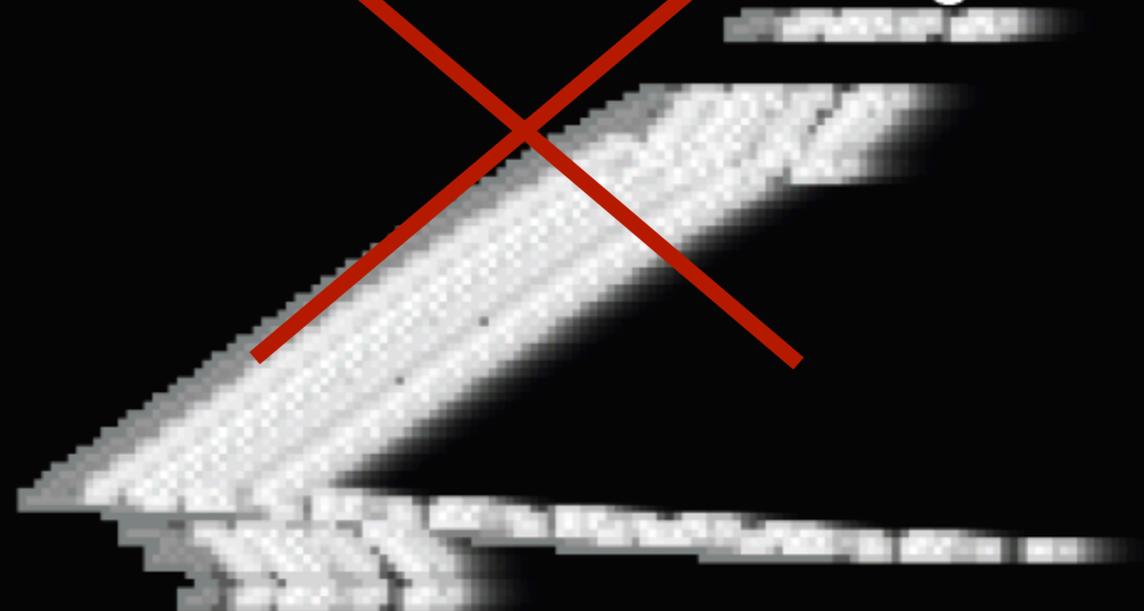
Step 2. Hits associated with DBSCAN clusters



Step 3. Pixelized, binary hits



Step 4. Pixelized hits w/ one-sided Gaussian shading



Manipulating the image to assist the algorithm in differentiating one region from another is no longer necessary!

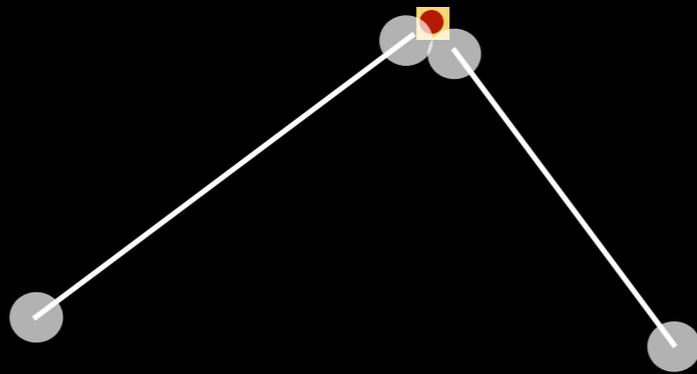
# Recent upgrades to VertexMatch

- VertexMatch takes those vertices found with HarrisVertexFinder and attempts to match them to the endpoints (within a window) of HoughLineFinder lines.
- Using two completely different methods to define a vertex allows “true” vertices (event vertex, decay points) to be differentiated from “false” ones (endpoints, delta rays, noise).
- If a vertex is matched to the endpoints of 2 or more Hough lines, it is considered a “strong” vertex.
- The strongest “strong” vertex is found considering the HarrisVertexFinder vertex strength and the sum of the length of the Hough lines associated with the vertex.

# Definitions

- Window around a hit
- Harris-vertex
- Hough-line
- Window around endpoint of Hough-line

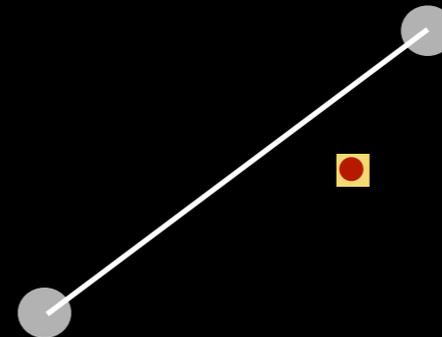
## Strong vertex



A Harris-vertex associated with the endpoints of at least 2 Hough lines.

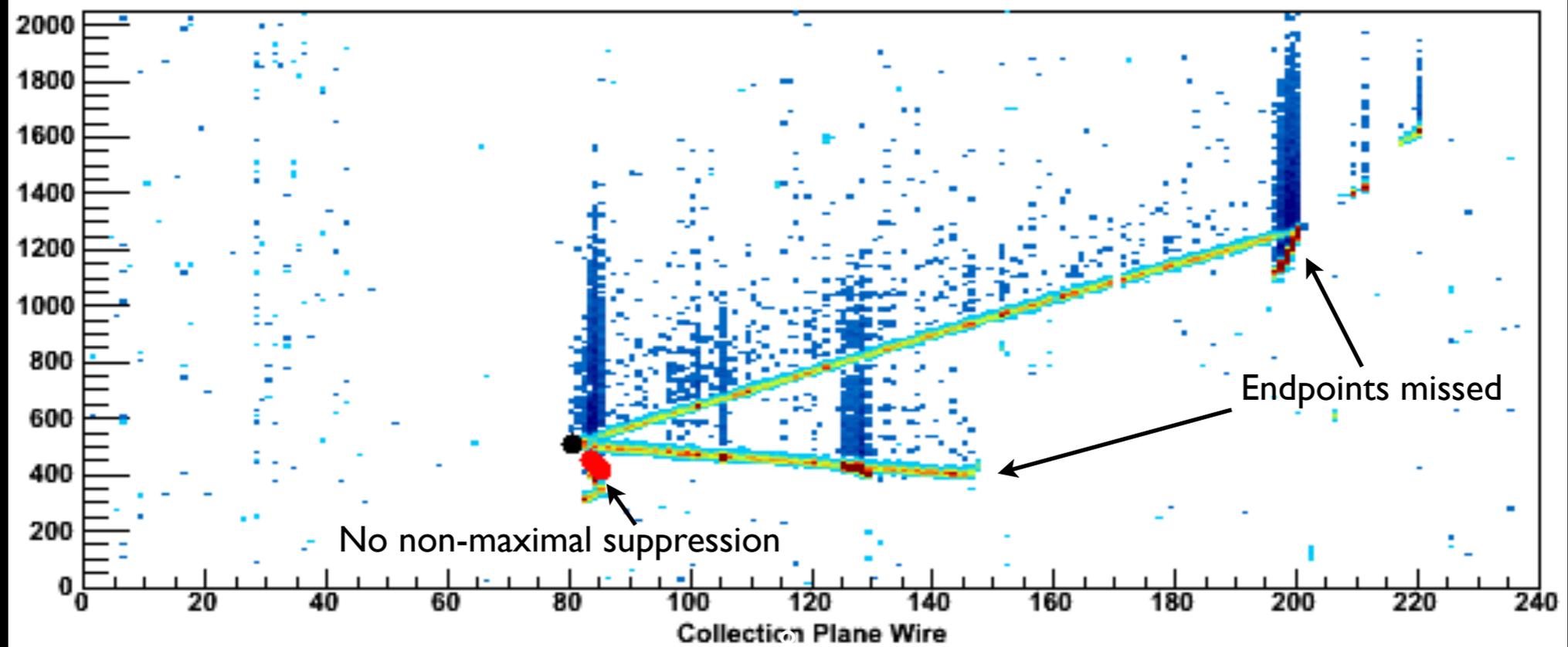
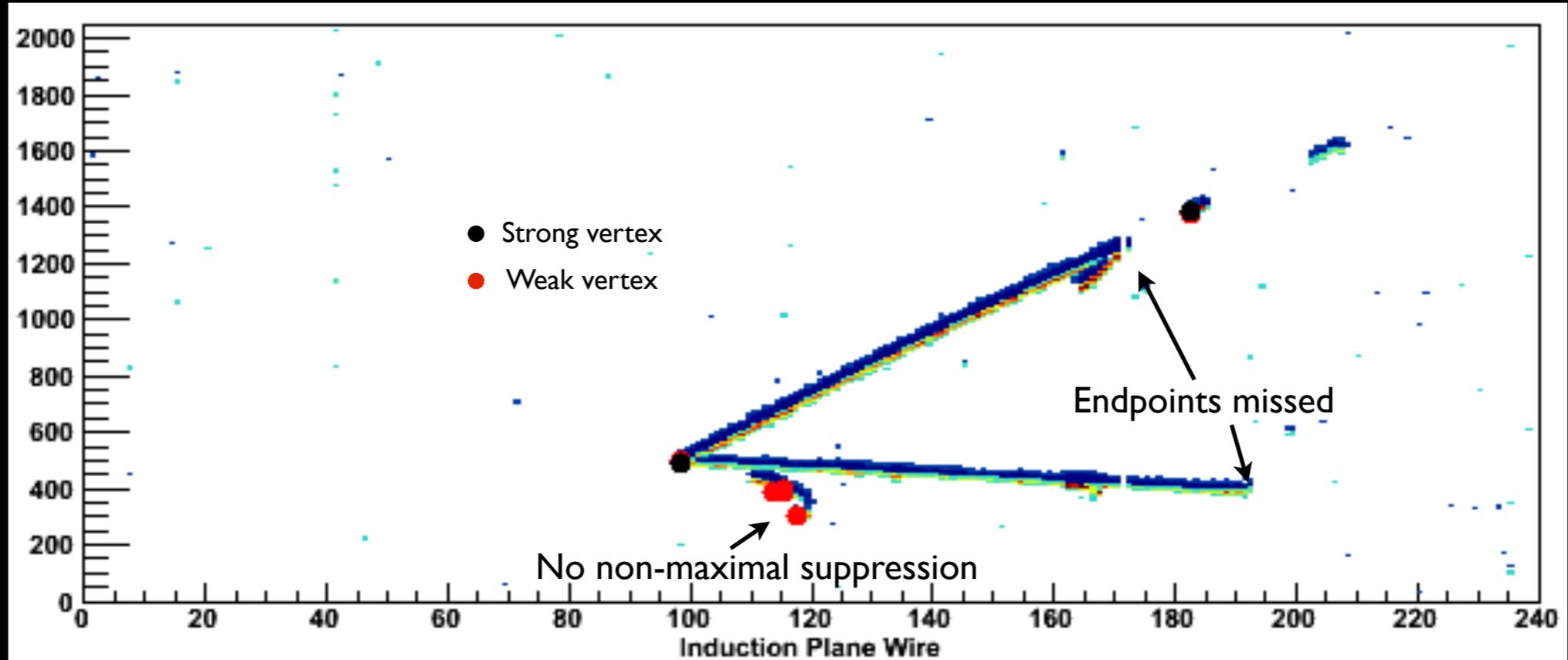
The strength of this vertex is given as the sum of the length of the Hough lines associated with it multiplied by the Harris strength.

## Weak vertex

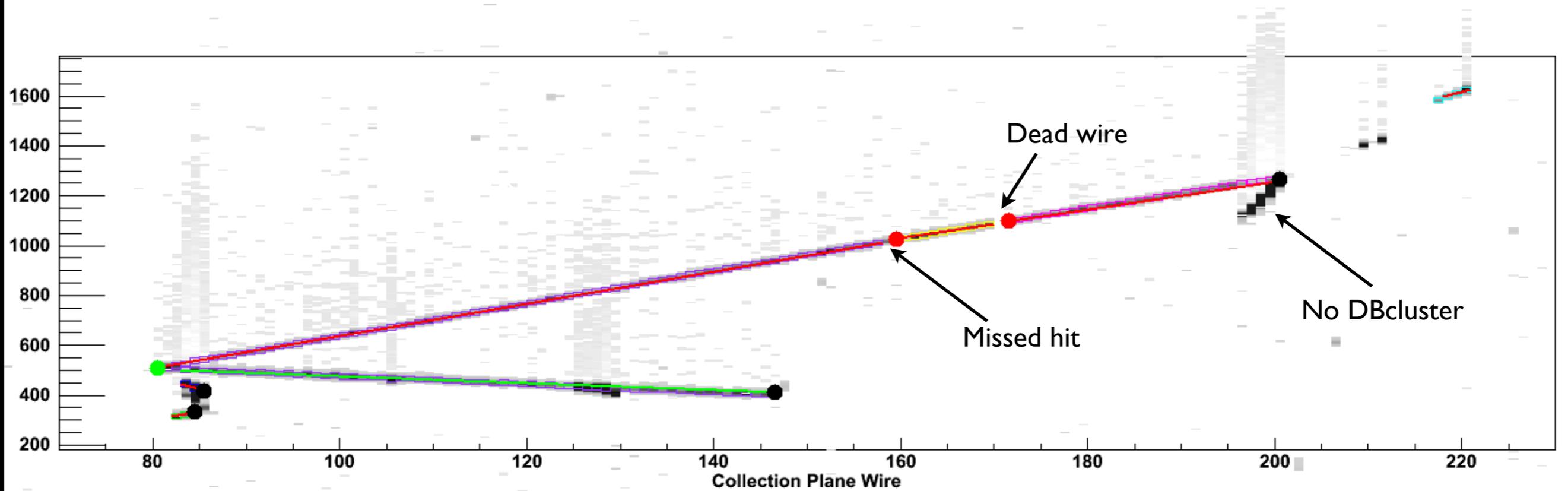
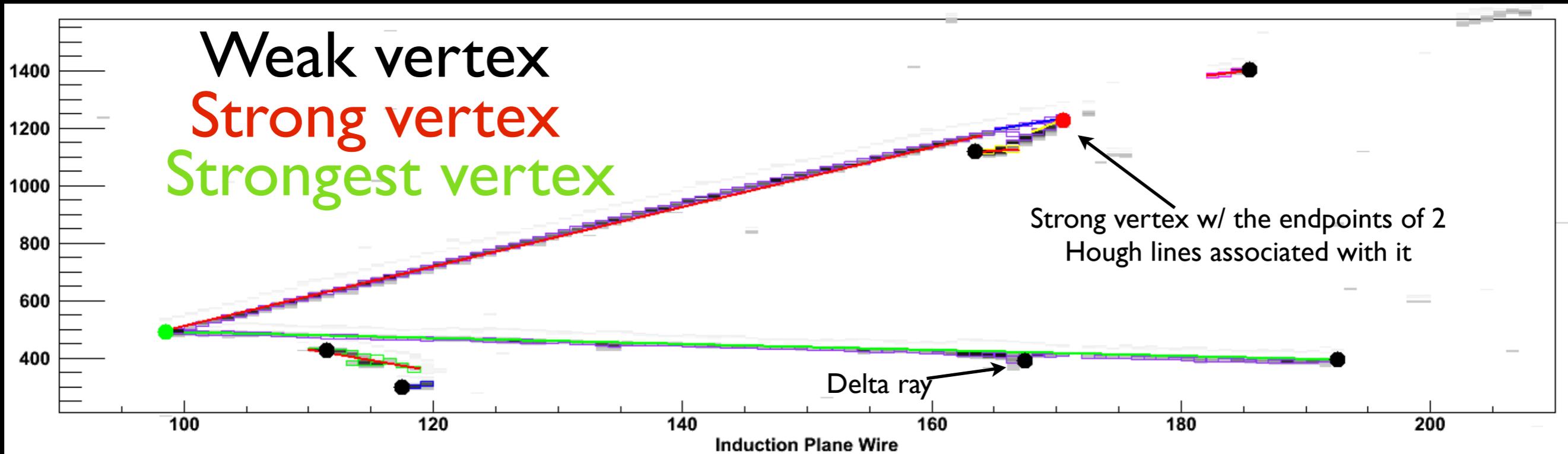


A Harris-vertex associated with less than 2 Hough lines.

# From my last talk



# The upgraded vertex finder



Note that Hough lines and DBSCAN clusters are visible in the event display

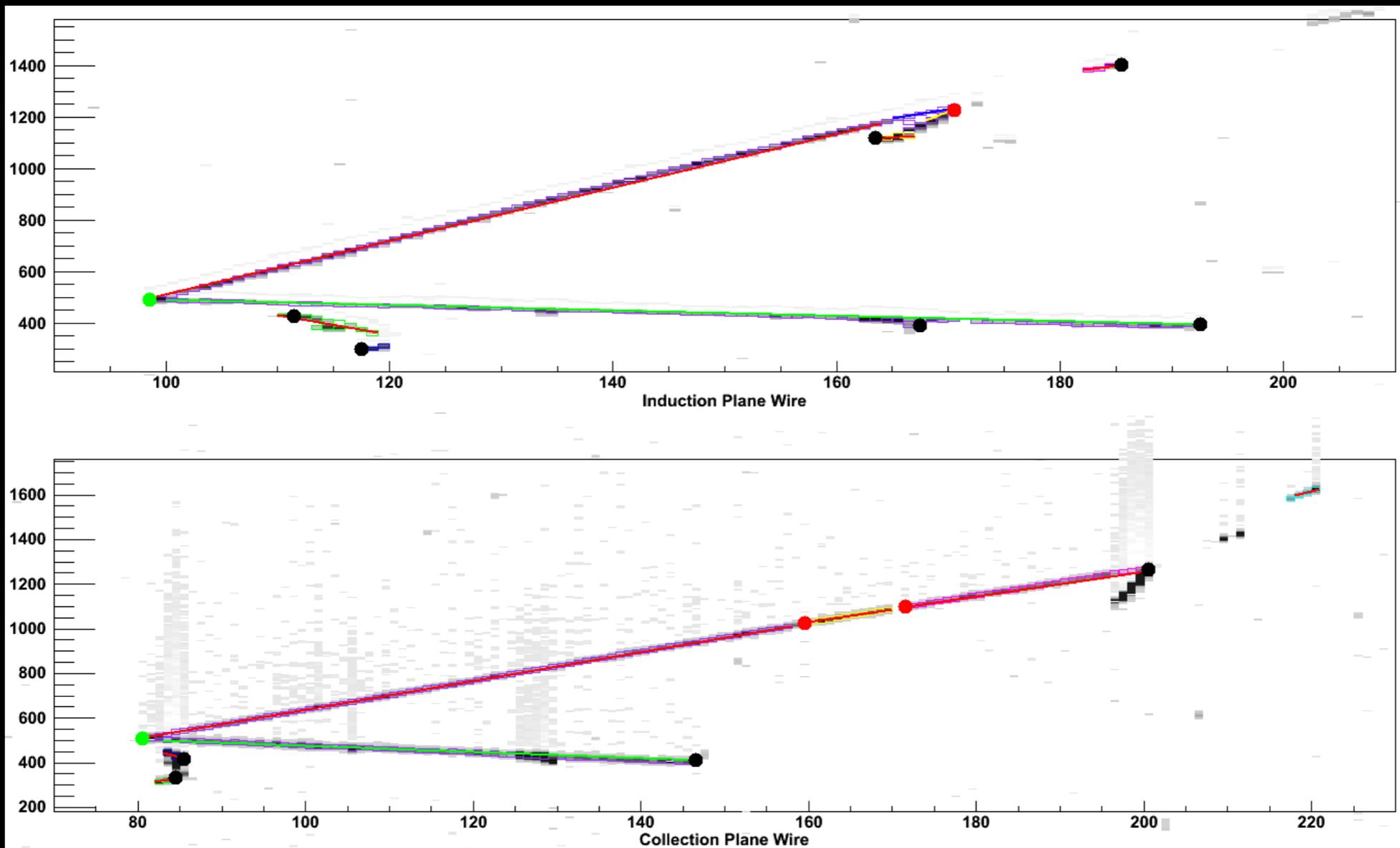
# HarrisVertexFinder

- A LArsoft module to find vertices from hits associated with DBSCAN clusters.
- Relevant xml parameters:
  - TimeBins, the number of time bins to use in pixelization.
  - Gsigma, sigma of the Harris algorithm's Gaussian window and the sigma of the Gaussian derivative window.
  - Window, the size of the non-maximal suppression window (the time and wire units are made consistent) in wire/time.
  - MaxCorners, maximum number of vertices to find (before non-maximal suppression).

# xml parameters for VertexMatch

- A LArSoft module to match hits, Hough line crossings, and Harris vertices and define the strength of a vertex.
- Relevant xml parameter:
  - Maximum distance between a vertex and a line to be considered a match.





- Useful for:
  - 3D matching and reconstruction.
  - Seeding track fitting algorithms.
  - Vertex activity characterization.
  - Event filtering.
  - Knowing if neutrino vertex is inside fiducial volume.

# Thank you

- Big thanks to B. Morgan at U. of Warwick for comments/suggestions.
- B. Morgan, "Interest Point Detection for Reconstruction in High Granularity Tracking Detectors", 2010. arXiv:1006.3012v1 [physics.ins-det]

# Backup

# Vertex.h in RecoBase

```
public:

Vertex(); ///Default constructor
Vertex(std::vector<const recob::Hit*> hits);
~Vertex();
void SetDriftTime(double drifttime) {fDriftTime=drifttime;}
void SetWire(int wire) {fWire=wire;}
void SetID(int ID) {fID=ID;}
void SetStrength(double Strength) {fStrength=Strength;}
bool Add(const recob::Hit* hit);
bool Remove(const recob::Hit* hit);
bool Clear();
std::vector<const recob::Hit*> Hits(int plane, int wire=-1) const;
std::vector<const recob::Hit*> Hits(gio::View_t view=gio::kUnknown) const;
double Charge(gio::View_t view=gio::kUnknown);
gio::View_t View() { return fView; }
double DriftTime() const {return fDriftTime; }
int Wire() const {return fWire; }
int ID() const {return fID; }
double Strength() const {return fStrength; }

private:
double fDriftTime; ///vertex's drift time
int fWire; ///vertex's wire number
int fID; ///vertex's ID
double fStrength; ///vertex's strength
gio::View_t fView; ///view for this vertex, gio::3D if 3D
TRefArray fHits; ///vector of hits for each plane
```