Calorimeter clustering with minimal spanning trees

G.Mavromanolakis, University of Cambridge

Outline

- ▶ **Introduction – theory**
- ▶ **Clustering algorithm**
- ▶ **Gallery**
- ▶ **Summary – future planning**

Introduction – theory

- **minimal spanning tree**

  : a tree which contains all nodes with no circuits and of which the sum of weights of its edges is minimum

- **properties**

  : unique for the given set of nodes and the chosen metric

  : deterministic, no dependence on random choices of nodes

  : invariant under similarity transformations that preserve the monotony of the metric

Introduction – theory

▶ MST and clustering

: *theorem 1:* any MST contains at least one edge from each link-set between P and Q partitions

: *theorem 2:* all MST edges are links of some partition of graph

: *theorem 3:* if S denotes the nodes of graph and C is a non-empty subset of S with the property that ρ(P,Q) $<$ ρ(C,S-C) for all partitions P, Q of C, then the restriction of any MST to the nodes of C forms a connected subtree of the MST

: *theorem 4:* if T is an MST for graph G and X, Y are two nodes of G then the unique path in T from X to Y is a minimax path from X to Y

References

[1] C.T.Zahn, *IEEE Trans.Comput. C20(1971)68*

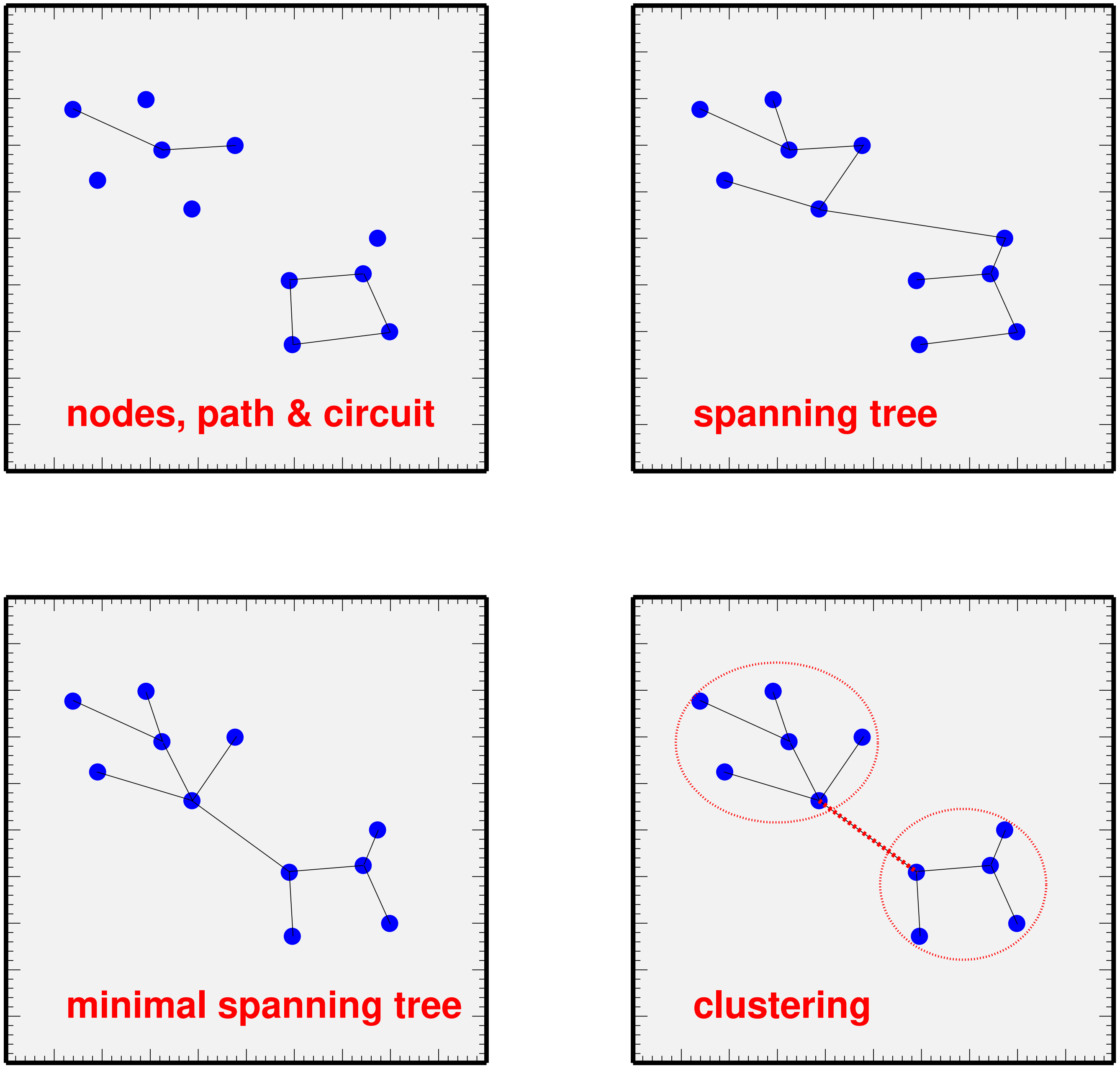
[2] J.C.Gower, G.J.S.Ross, *Appl.Statis. 18(1969)54*

[3] G.J.S.Ross, *Appl.Statis. 18(1969)103*

[4] R.C.Prim, *Bell System Techn.Jour. 36(1957)1389*

[5] K.DeWinter etal. (CHARM II Collaboration) *Nucl.Instr.Meth. A277(1989)170*

[6] N.Saoulidou *Ph.D. thesis, Univ. of Athens 2003*


nodes, path & circuit
spanning tree
minimal spanning tree
clustering

Prim's algorithm

R.C.Prim, *Bell System Techn.Jour. 36(1957)1389*

- **construct the MST**

  : exploit the key properties of MST theory and construct the MST of N nodes with a $(N-1)^2$-order loop

  : **NOTICE:** if you construct all possible spanning trees out of N nodes, and then find the minimum one, it will be a $\mathcal{O}(N!)$ loop !!!

Prim's algorithm

- **pseudocode**

```
.   start with setA containing 1 node and setB with N-1 nodes;
.   while(setA contains less than N nodes)
.   {
..  find the node of setB with the shortest edge connecting
    to any node already in setA;
..  move this node to setA;
..  the edge connecting the two nodes belongs to MST;
.   }
```

- **... in plain ...**

  : start with any one of the given nodes and initially assign to setA the node with the shortest edge to this node

  : then continue to add to setA the nodes from setB with the shortest edge and which connect to at least one node of setA without forming a circuit

  : iterate until all nodes have been assigned to setA

Clustering algorithm – General description

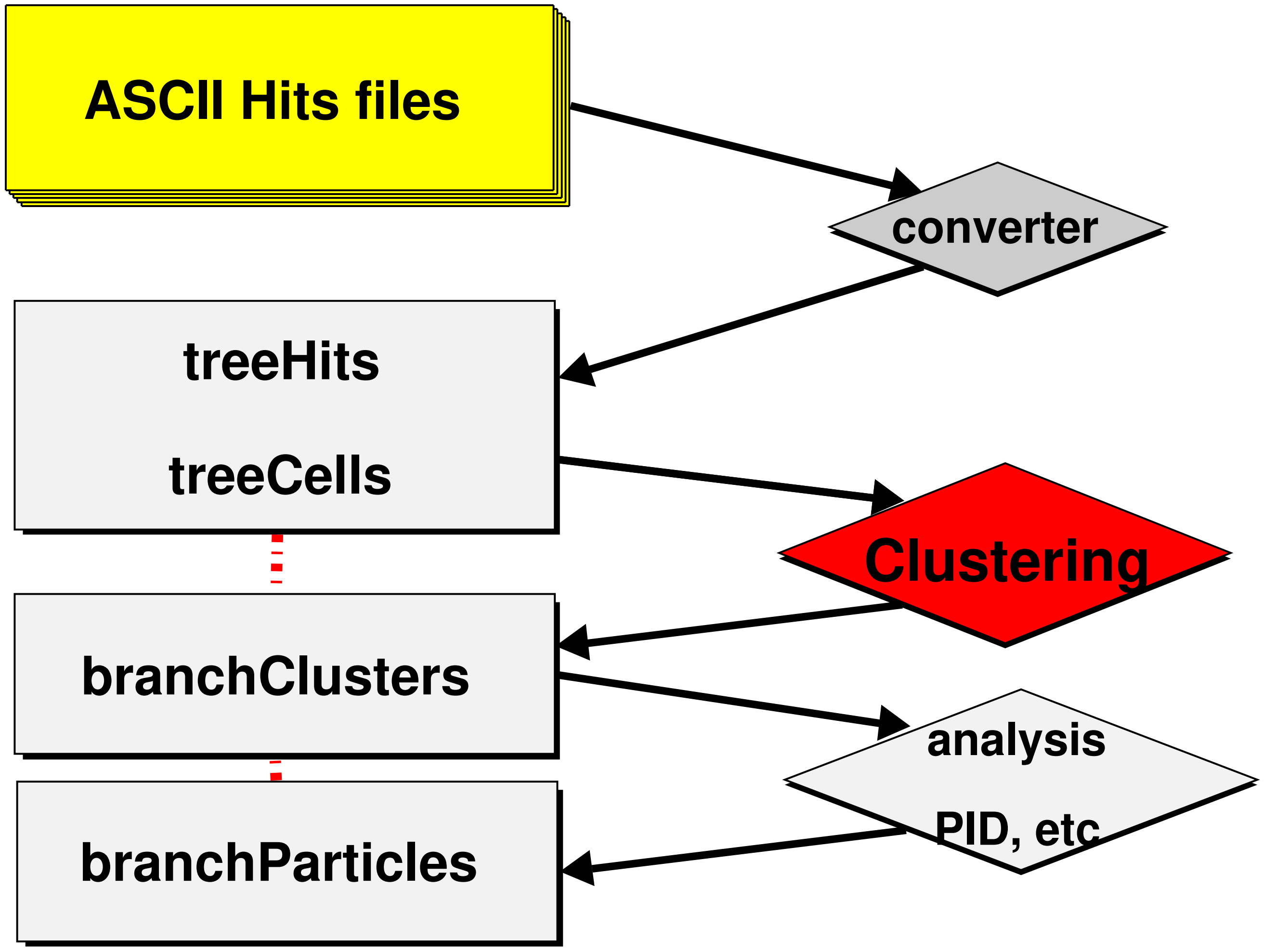
- code is arranged in two classes, **MST** and **MSTclusters**

  MST objects give birth to MSTclusters objects through MST::mergeClusters()

- main methods

defineMetric()	:	metric defines configuration space
	:	not necessarily euclidean
fillDistanceMatrix()	:	lower triangular distance matrix with diagonal elements omitted
constructMST()	:	apply Prim's algorithm to construct the minimal spanning tree
setLevel()	:	"proximity" bound between nodes belonging to the same cluster
mergeClusters()	:	single linkage cluster analysis
	:	i.e. go through MST and cut branches with length above Level

Reconstruction – From Hits to Particles



Clustering algorithm – General description

- **clustering with MST**

  : algorithm has been coded in C++

  : interface with data stored in ROOT trees

  : after defining an "appropriate" metric to fill the distance matrix, the rest of the algorithm has no dependency on data format and detector geometry since only the metric deals with these

  : so very easy to switch to different geometries/detectors

Application to calorimeter

▶ **test, check, debug**

: single particle runs
with CALICE ECAL, HCAL prototype geometries,
with full TESLA-ECAL, HCAL detectors

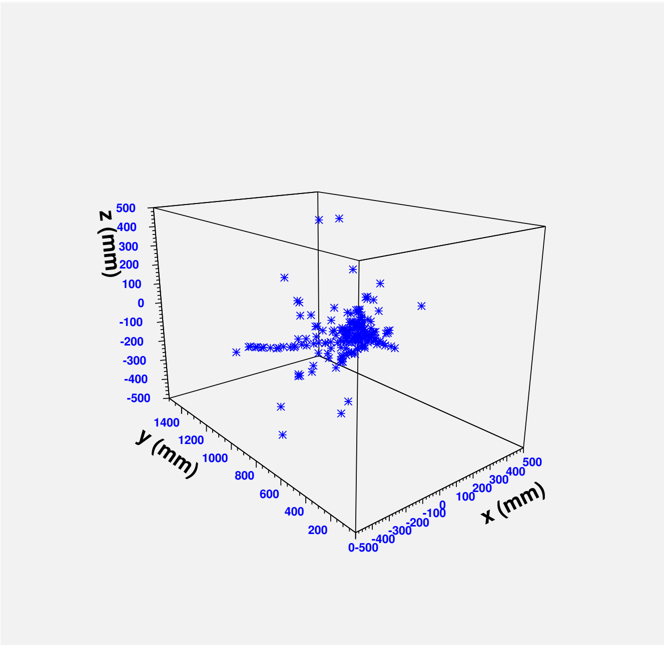
: common clustering behavior observed
(one main cluster plus several small ones)

: debugging runs with multiparticle events

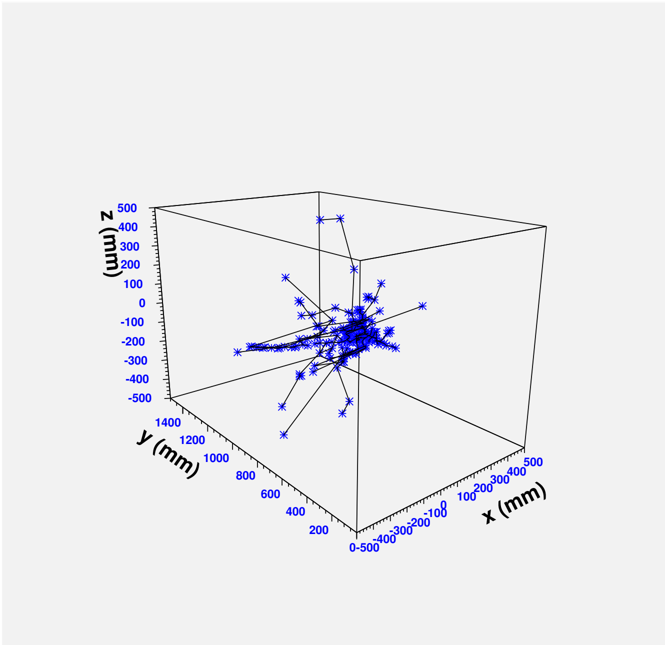
: first studies with non-euclidean metrics
(e.g. "gravity-like" metric, $D_{ij} \propto \frac{|\vec{r_i}-\vec{r_j}|}{f(E_i,E_j)}$)

: some example plots for illustration follow

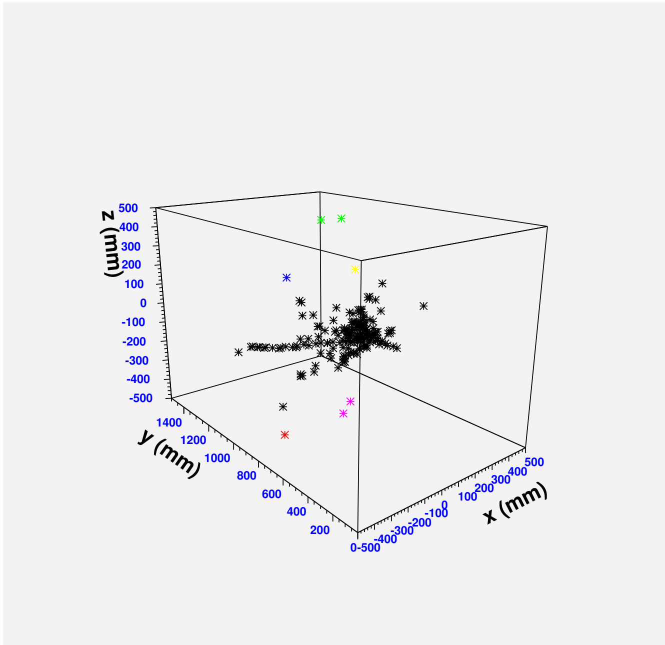
nodes

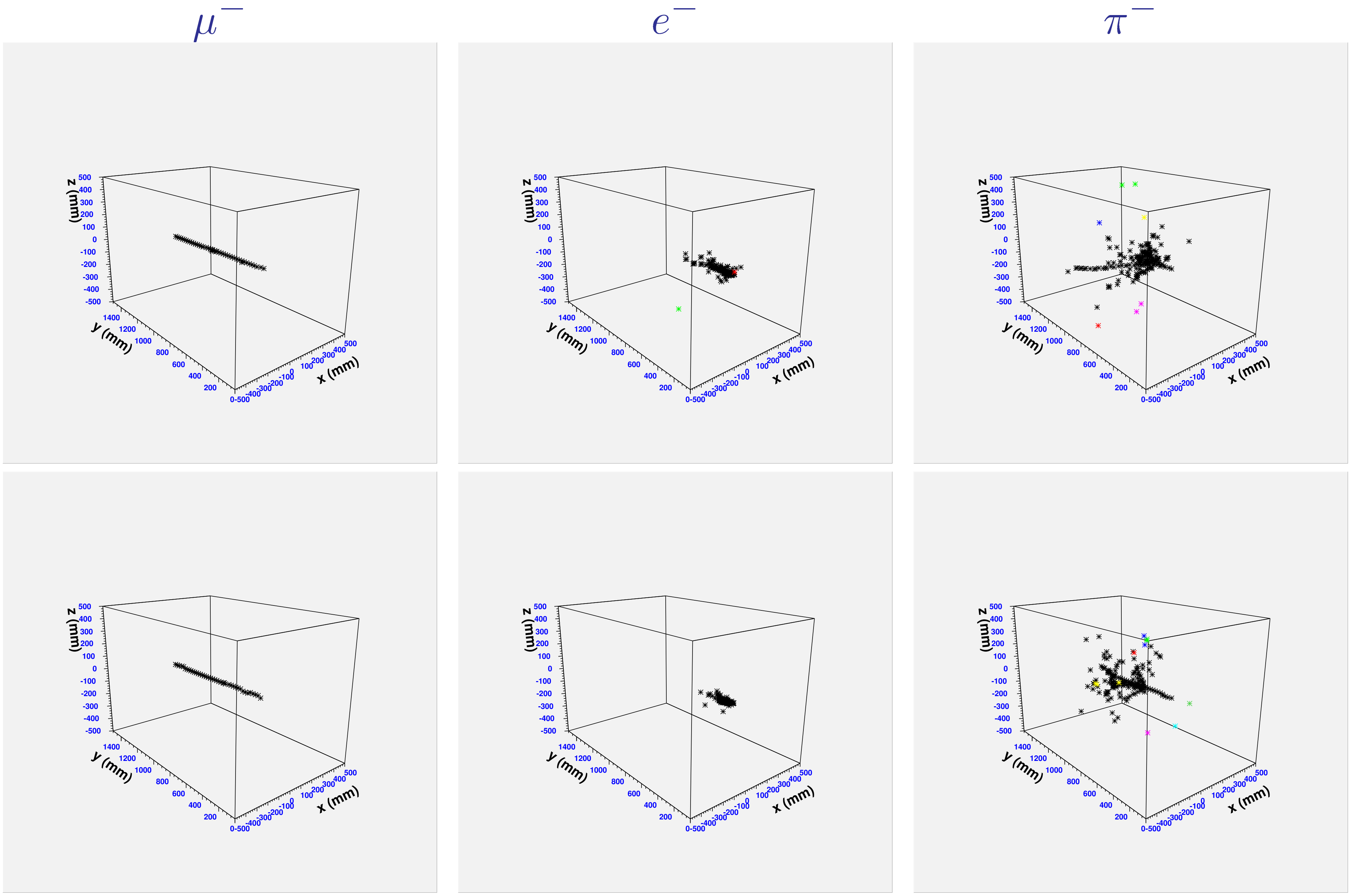


minimal spanning tree



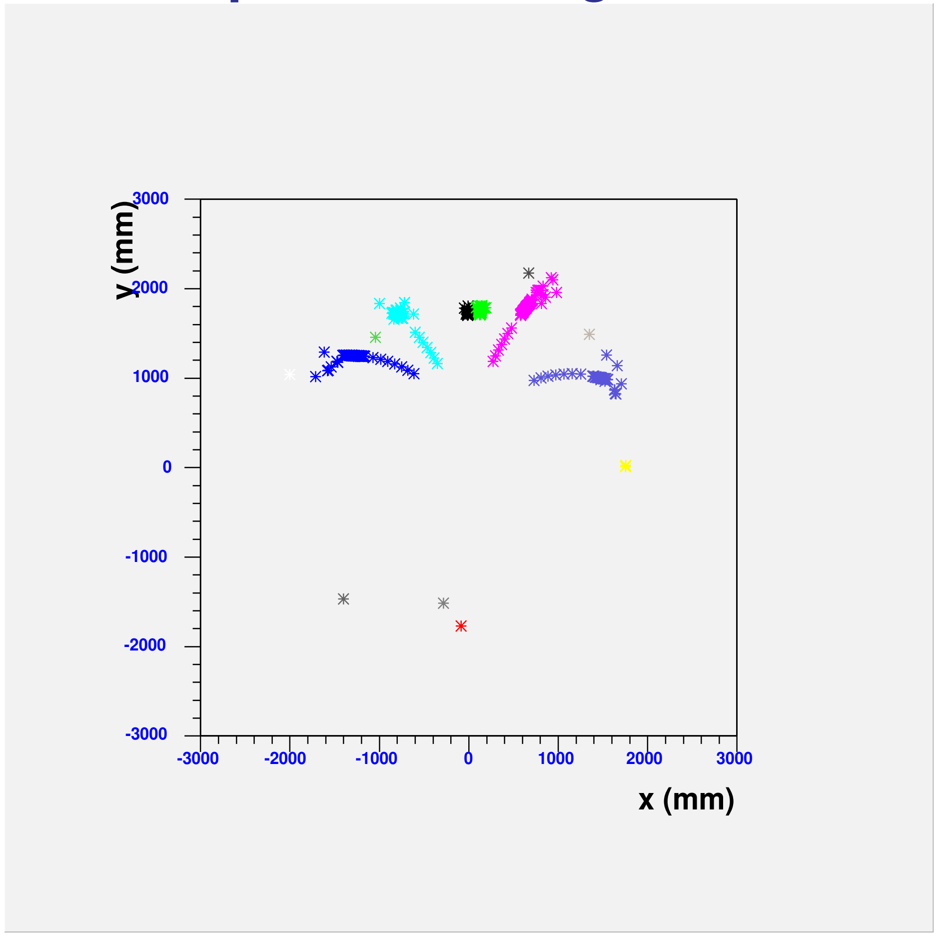
clustering



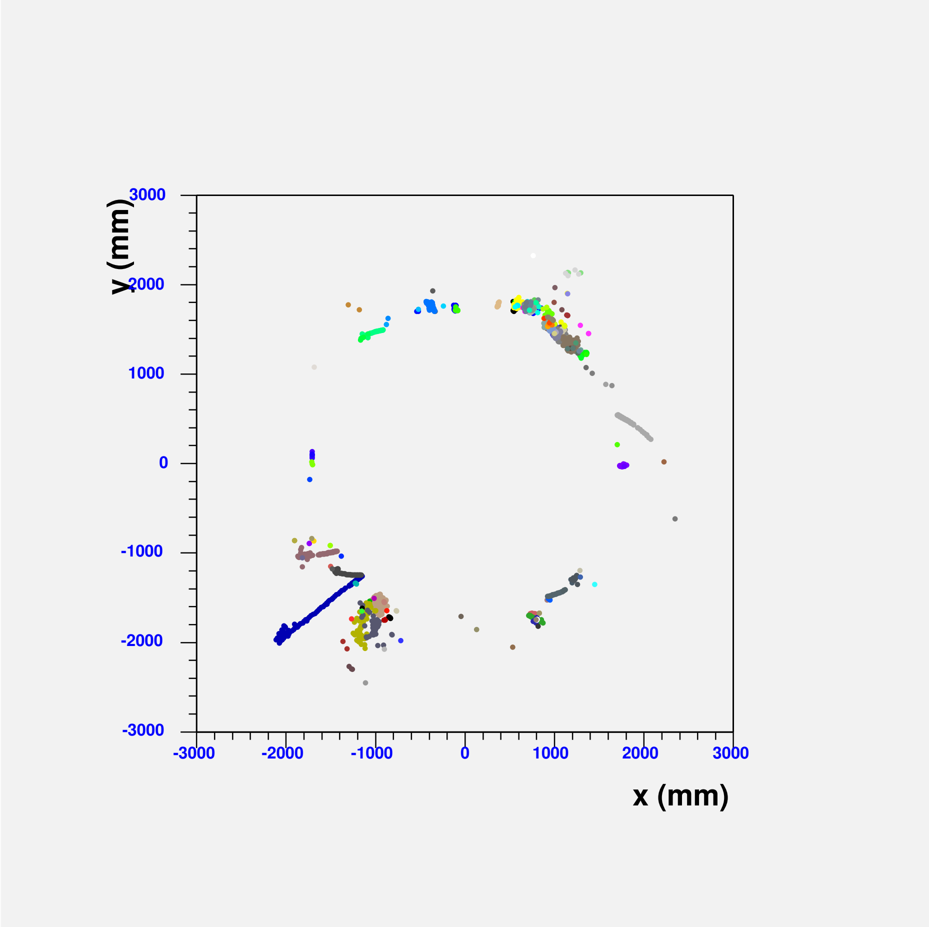


μ^-
e^-
π^-
z (mm)
y (mm)
x (mm)

example – including TPC hits



example – multiparticle event



Summary – Future planning

- **clustering with MST**
  - clustering algorithm based on MST approach has been coded in C++
  - interface with data stored in ROOT trees
  - kernel has minimum dependency on data format and detector geometry, only the metric deals with these
  - first debugging phase passed
- **general planning**
  - study clustering efficiency - robustness
  - implement particle identification
  - evaluate reconstruction performance