An implement of Kalman filtering in MarlinTPC

LI Bo

Center for High Energy Physics
Tsinghua University

Equations

System equation:

$$x_{k+1} = \phi_k x_k + w_k$$

x_k – system state vector at discrete time k

Φ_k – state transfer matrix

w_k – process noise (e.g. multiple scattering)

Measurement equation:

$$z_k = H_k x_k + v_k$$

H_k – measurement matix (projection)

v_k – measurement error

Covariance:

$$E[w_i w_j^T] = Q_i \delta_{ij} \quad E[v_i v_j^T] = R_i \delta_{ij} \quad E[w_i v_j^T] = 0$$

A simple deduction of Kalman Filter

- Prediction:

$$\hat{x}_k^- = \phi_{k-1}\hat{x}_{k-1}$$

$$\begin{aligned} P_k^- &= \operatorname{cov}[\hat{x}_k - x_k] = \operatorname{cov}[\phi_{k-1}\hat{x}_{k-1} - \phi_{k-1}x_{k-1} - w_{k-1}] \\ &= \operatorname{cov}[\phi_{k-1}\left(\hat{x}_{k-1} - x_{k-1}\right)] + \operatorname{cov}[w_{k-1}] \\ &= \phi_{k-1}P_{k-1}^-\phi_{k-1}^T + Q_{k-1} \end{aligned}$$

- Filtering:

Let $L\left(\hat{x}_k\right) = \left(\hat{x}_k - \hat{x}_k^-\right)^T \left(P_k^-\right)^{-1}\left(\hat{x}_k - \hat{x}_k^-\right) + \left(z_k - H_k\hat{x}_k\right)^T R_k^{-1}\left(z_k - H_k\hat{x}_k\right)$

$\equiv \min$

Then the least square estimator is,

$$\hat{x}_k = \hat{x}_k^- + K_k\left(z_k - H_k\hat{x}_k^-\right)$$

in which, $K_k = P_k^- H_k^T\left(H_k P_k^- H_k^T + R_k\right)^{-1}$ (Kalman gain matrix)

and $P_k = \left(1 - K_k H_k\right)P_k^-$

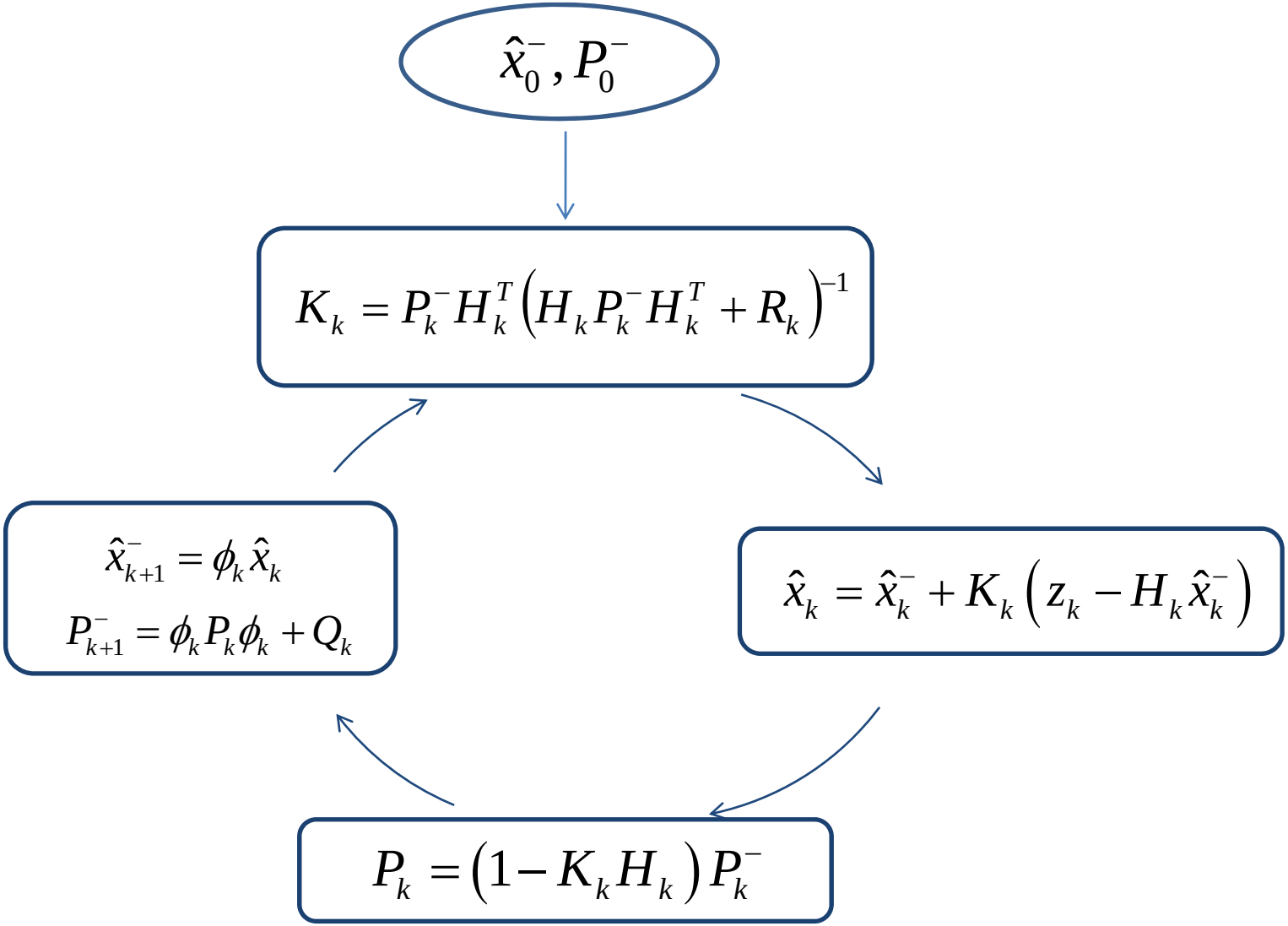
Filtered Residual $r_k = z_k - H_k\hat{x}_k = \left(1 - H_k K_k\right)\left(z_k - H_k\hat{x}_k^-\right)$

Its covariance

$$\begin{aligned} C_k &= \mathrm{cov}[\hat{r}_k] = \mathrm{cov}\left[\left(1 - H_k K_k\right)\left(z_k - H_k\hat{x}_k^-\right)\right] \\ &= \left(1 - H_k K_k\right)\mathrm{cov}\left[z_k - H_k\hat{x}_k^-\right] \\ &= \left(1 - H_k K_k\right)R_k \end{aligned}$$

χ^2 update $\chi_k^2 = \chi_{k-1}^2 + \chi_+^2 = \chi_{k-1}^2 + r_k^T C_k^{-1} r_k$

Kalman Filtering Loop



Straight model

The line equation is:

$$y_k = a_k x + b_k$$

re-writes it for kalman filtering:

$$X_k = \begin{bmatrix} a_k \\ b_k \end{bmatrix} \qquad \phi_k = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} \qquad H_k = \begin{bmatrix} k & 1 \end{bmatrix}$$

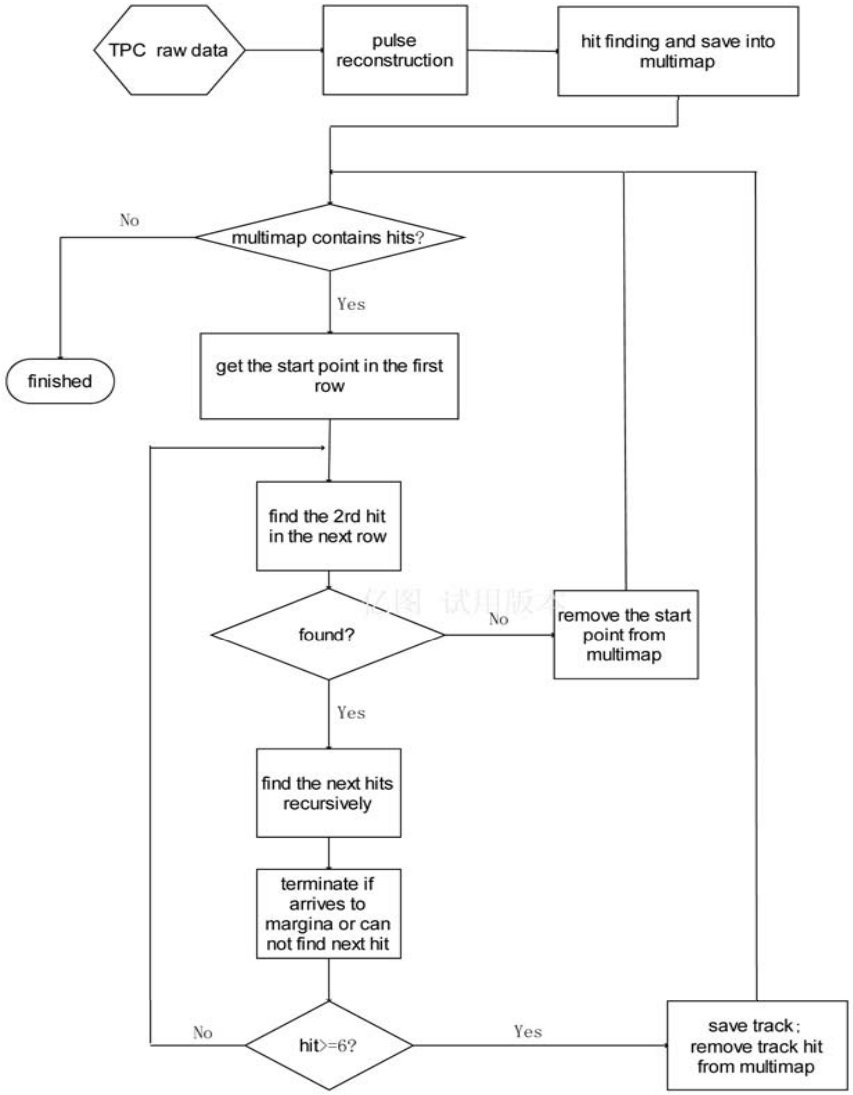
So both slope and intercept can be got.

In track following method in which kalman filter is used, track finding and track fitting are completely simultaneously. And the estimator

$$\hat{y}_k = \hat{a}_k x + \hat{b}_k$$

is used to discriminate wrong hit at every step.

Flowchart of Algorithm in MarlinTPC



Test condition

- The spatial resolution is not considered.
- Raw data is simulated by a C++ code
- Readout geometry is for TU-TPC(10 rows)
- Track definition: hit size>=6, MaxSkipRow=1.

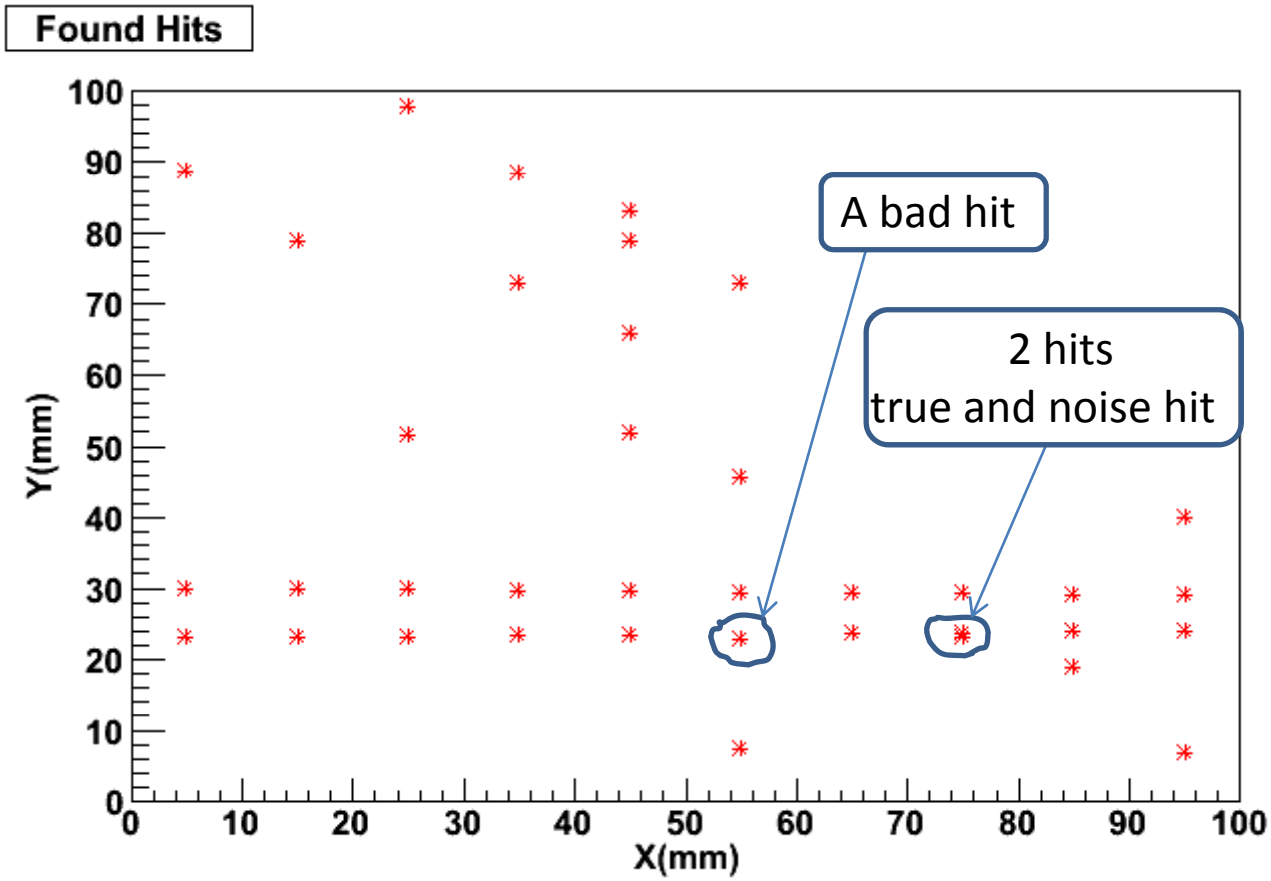
Test 1

- Formula of two tracks:

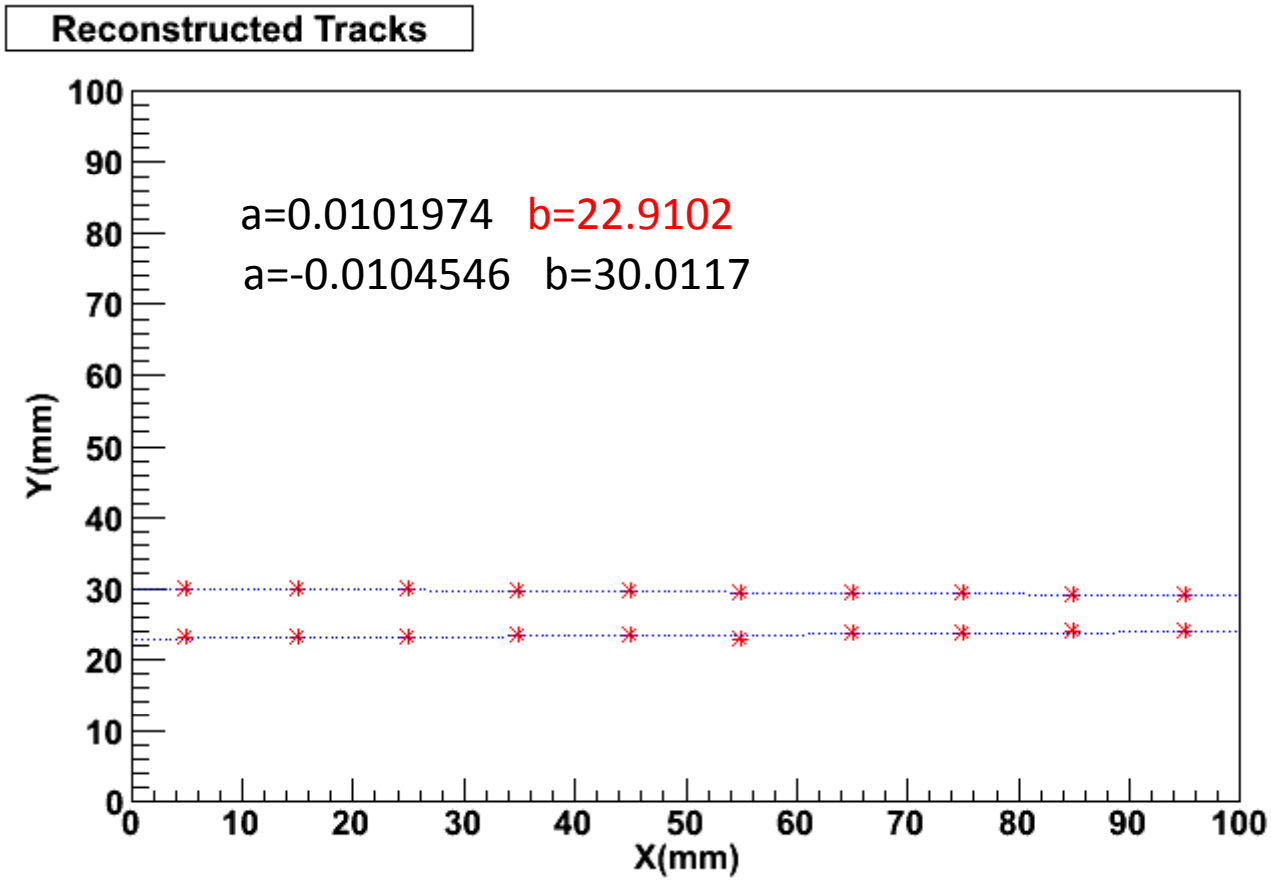
$$y_1 = 0.01x + 23$$
$$y_2 = -0.01x + 30$$

- 21 noise hits are added to the true hits.
- Discrimination value in the propagate step is 1.

Test 1-1

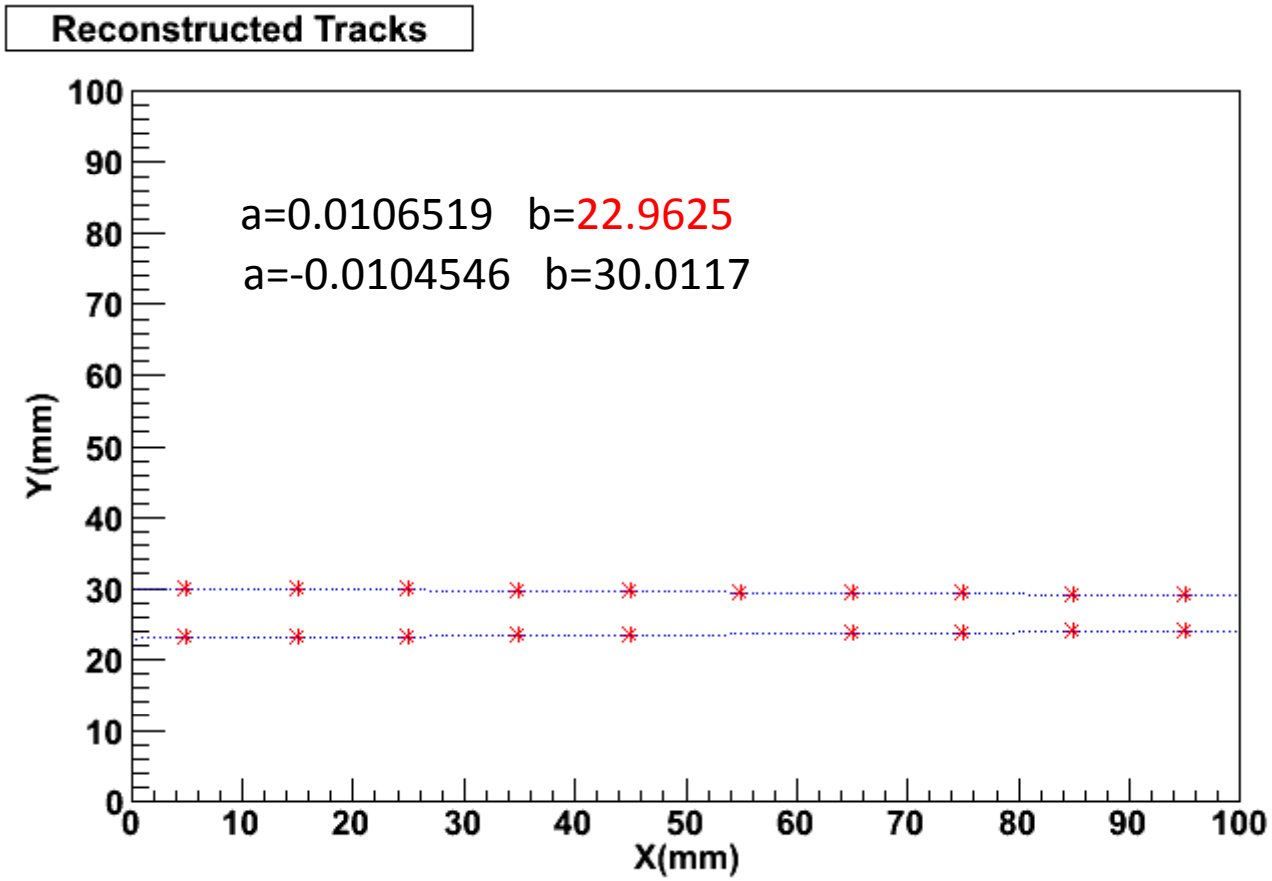


Test 1-1


Reconstructed Tracks
a=0.0101974 b=22.9102
a=-0.0104546 b=30.0117
Y(mm)
X(mm)
0
10
20
30
40
50
60
70
80
90
100

Test 1-2

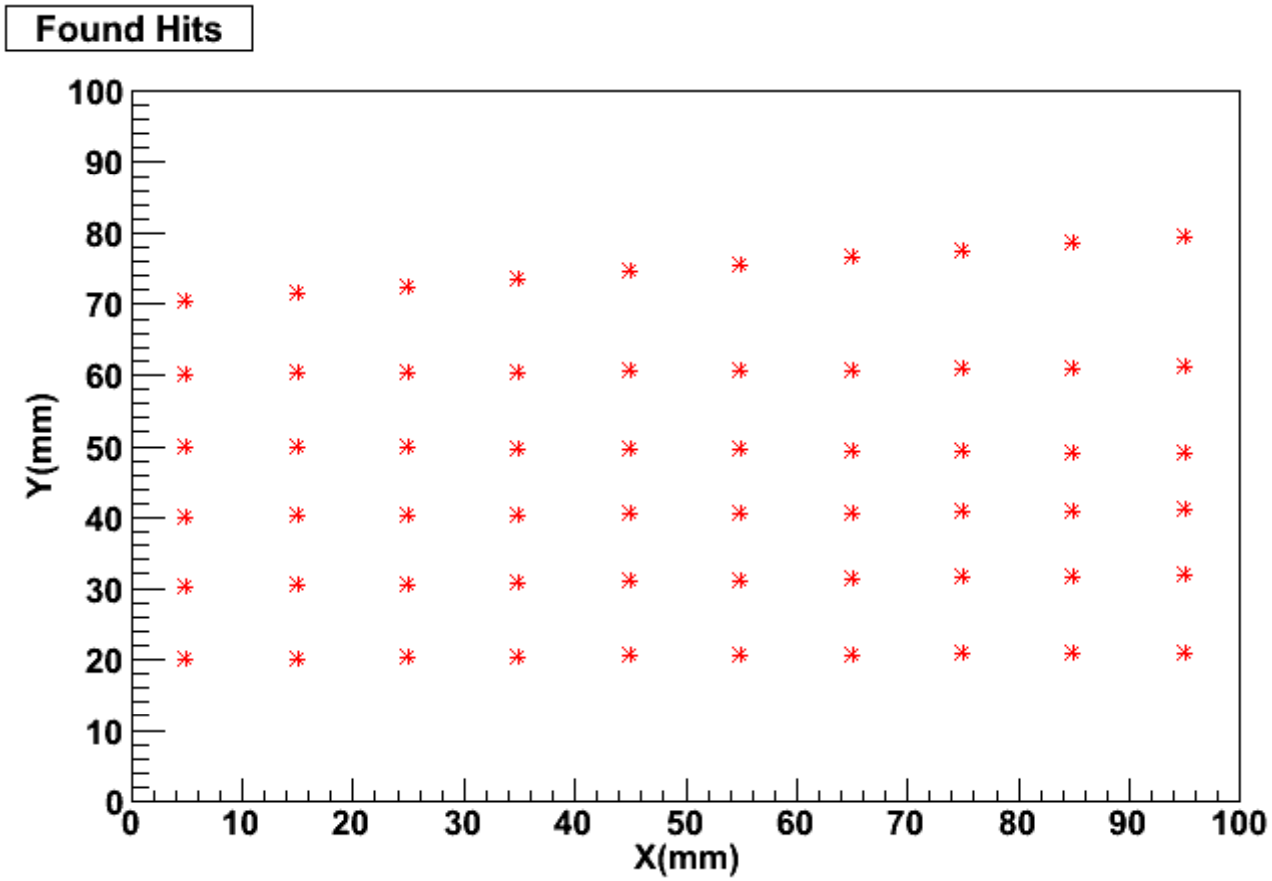
Discrimination value is 0.1, other two conditions is the same as test 1-1.



Conclusions from test 1

- The track finding code can have good capability to discriminate noise, if a proper value is chosen.
- Track fitting result is improved by abandoning bad hit.

Test 2-1



y=0.1x+70

y=0.01x+60

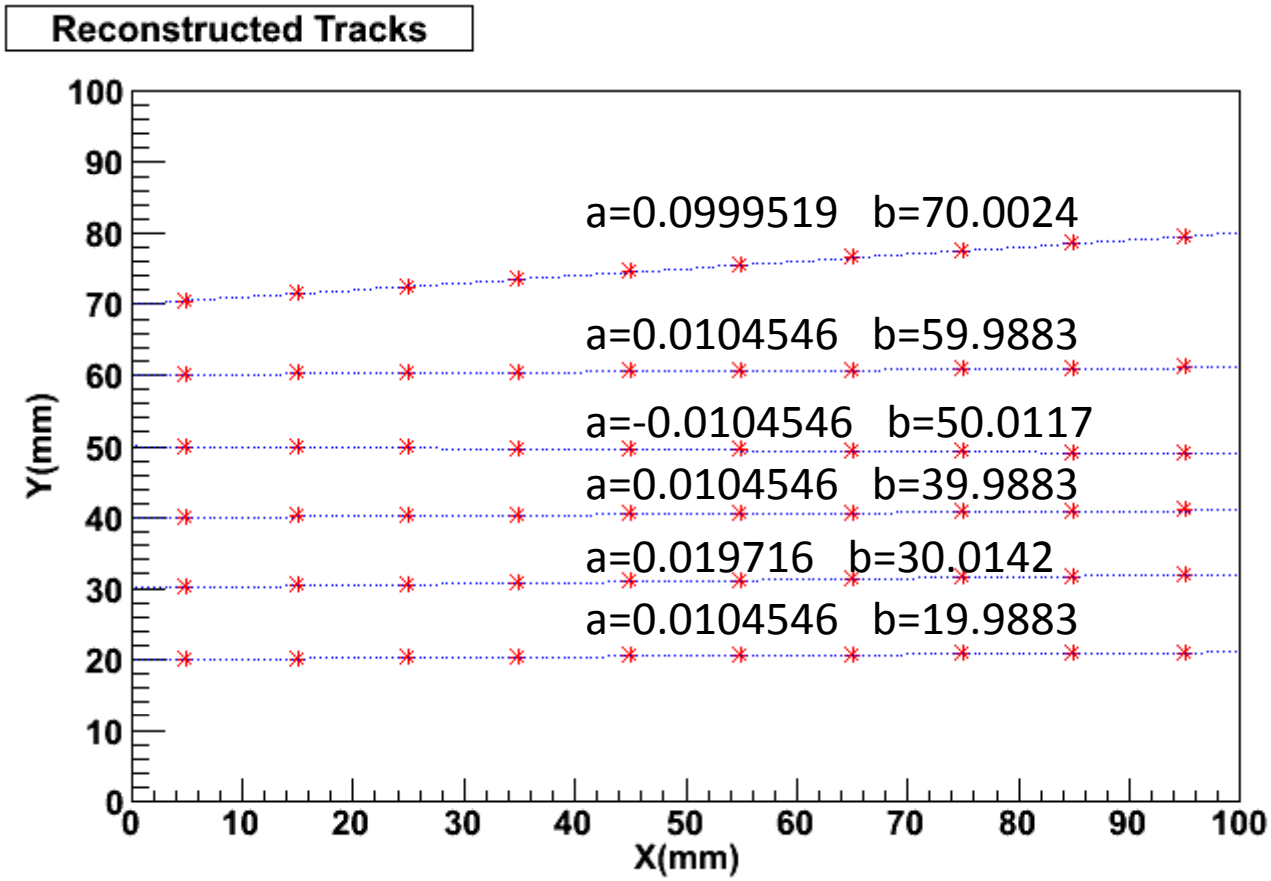
y=-0.01x+50

y=0.01x+40

y=0.02x+30

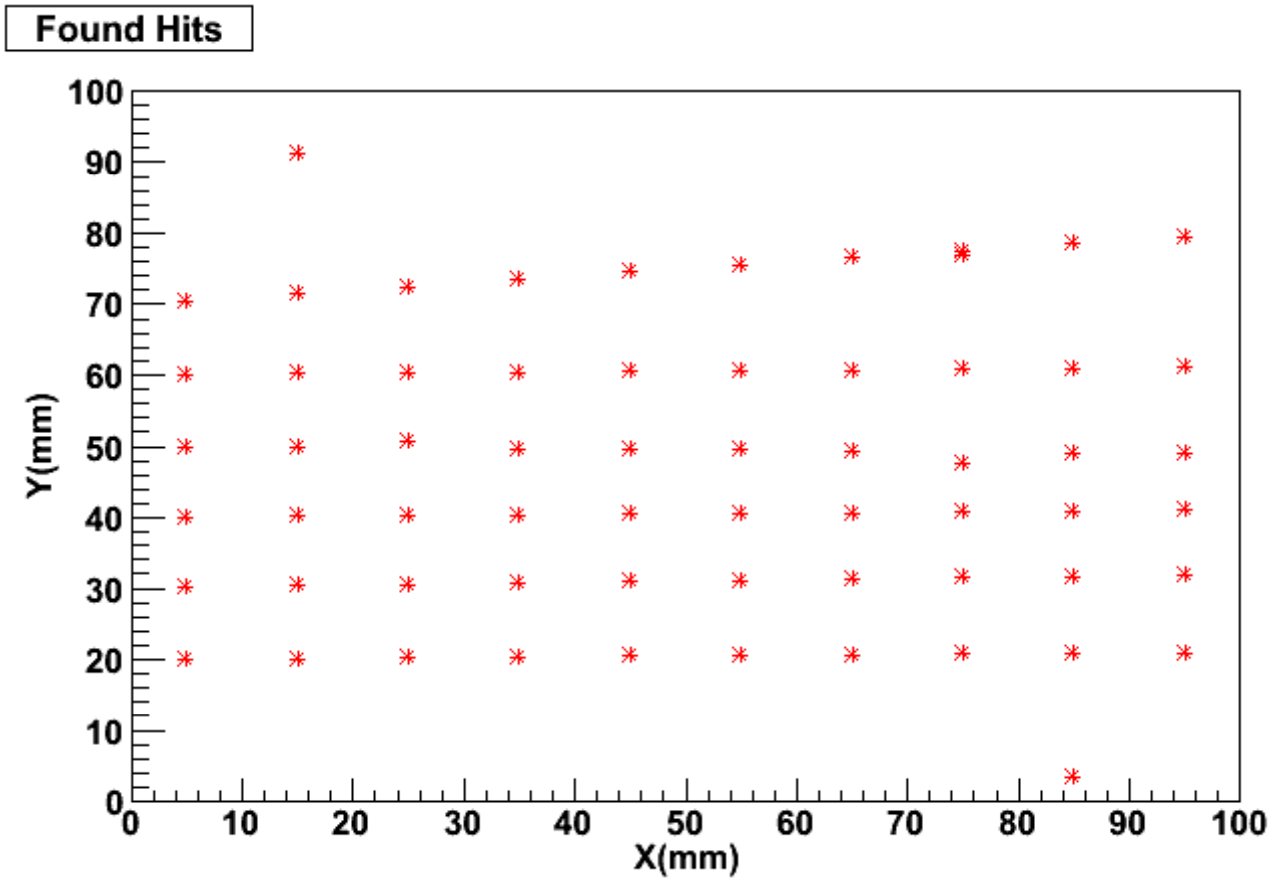
y=0.01x+20

Test 2-1

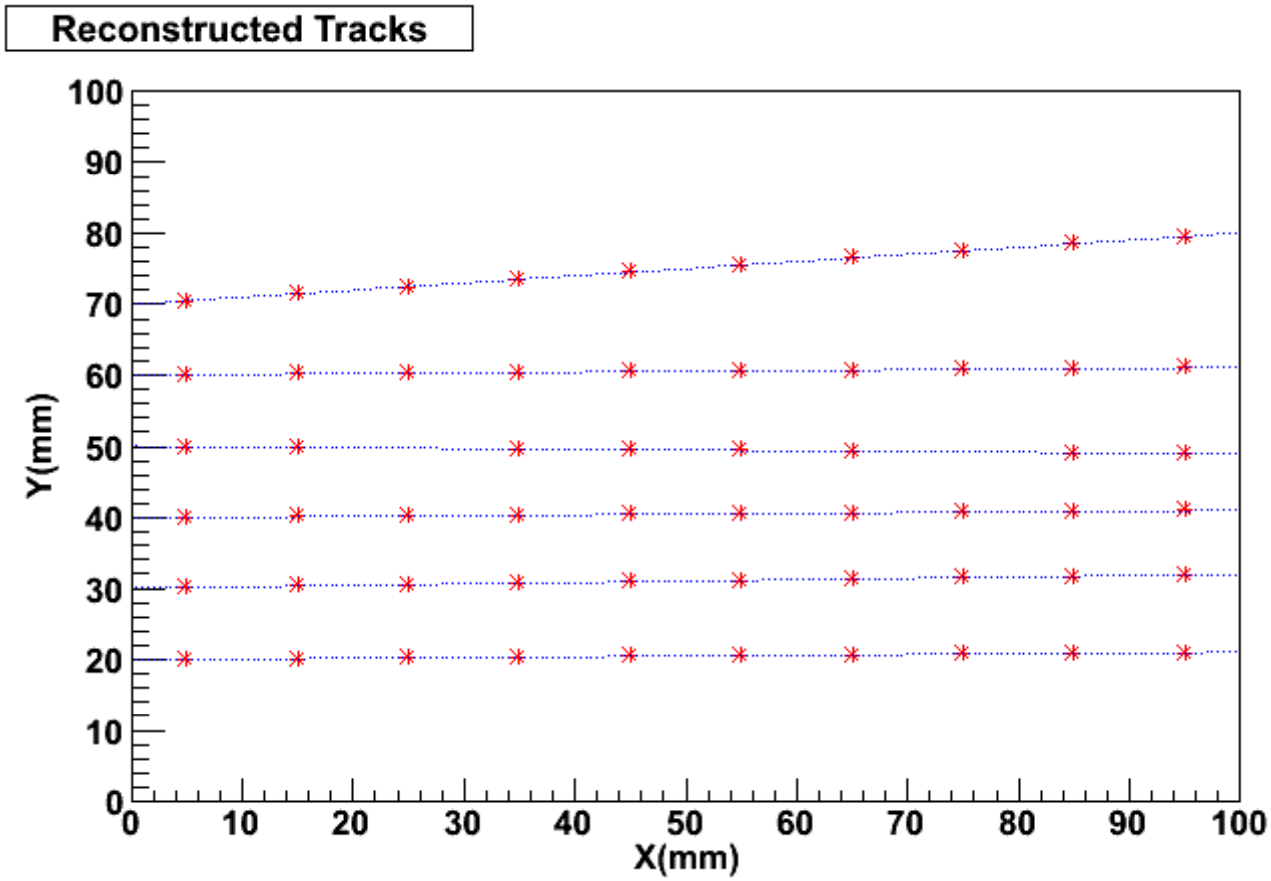

Reconstructed Tracks
a=0.0999519 b=70.0024
a=0.0104546 b=59.9883
a=-0.0104546 b=50.0117
a=0.0104546 b=39.9883
a=0.019716 b=30.0142
a=0.0104546 b=19.9883
Y(mm)
X(mm)
0
10
20
30
40
50
60
70
80
90
100

Test 2-2

Add 5 noise hits to test 2-1

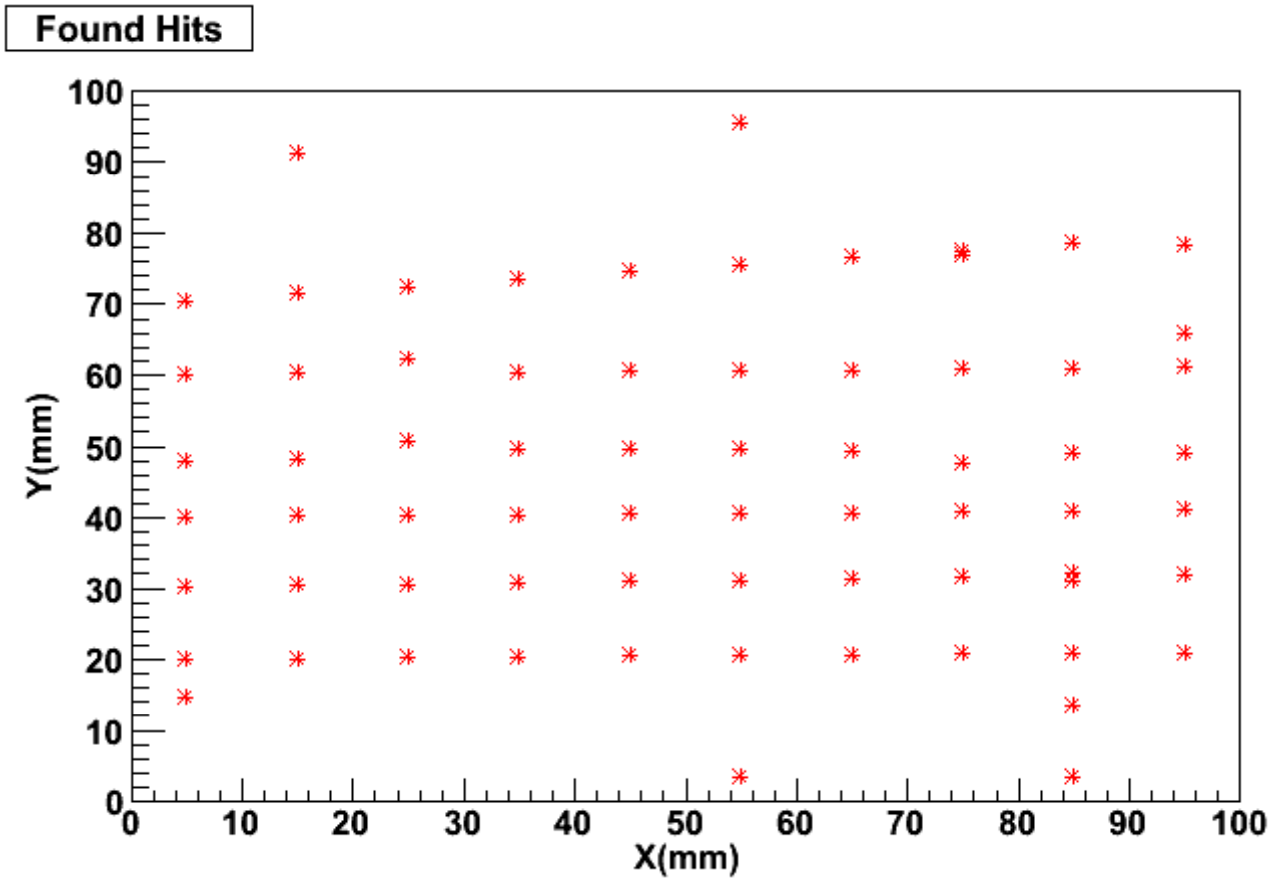


Test 2-2

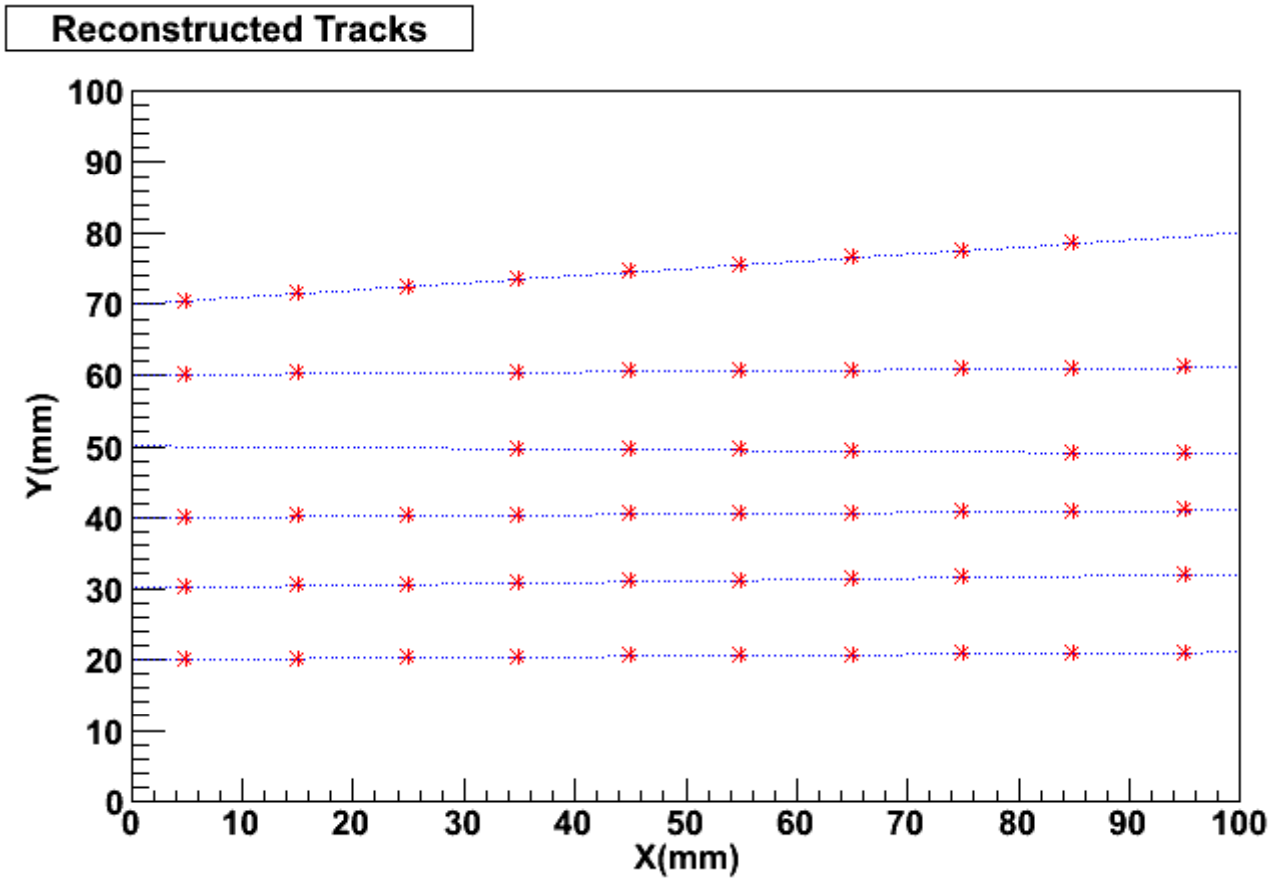


Test 2-3

Add 15 noise hits

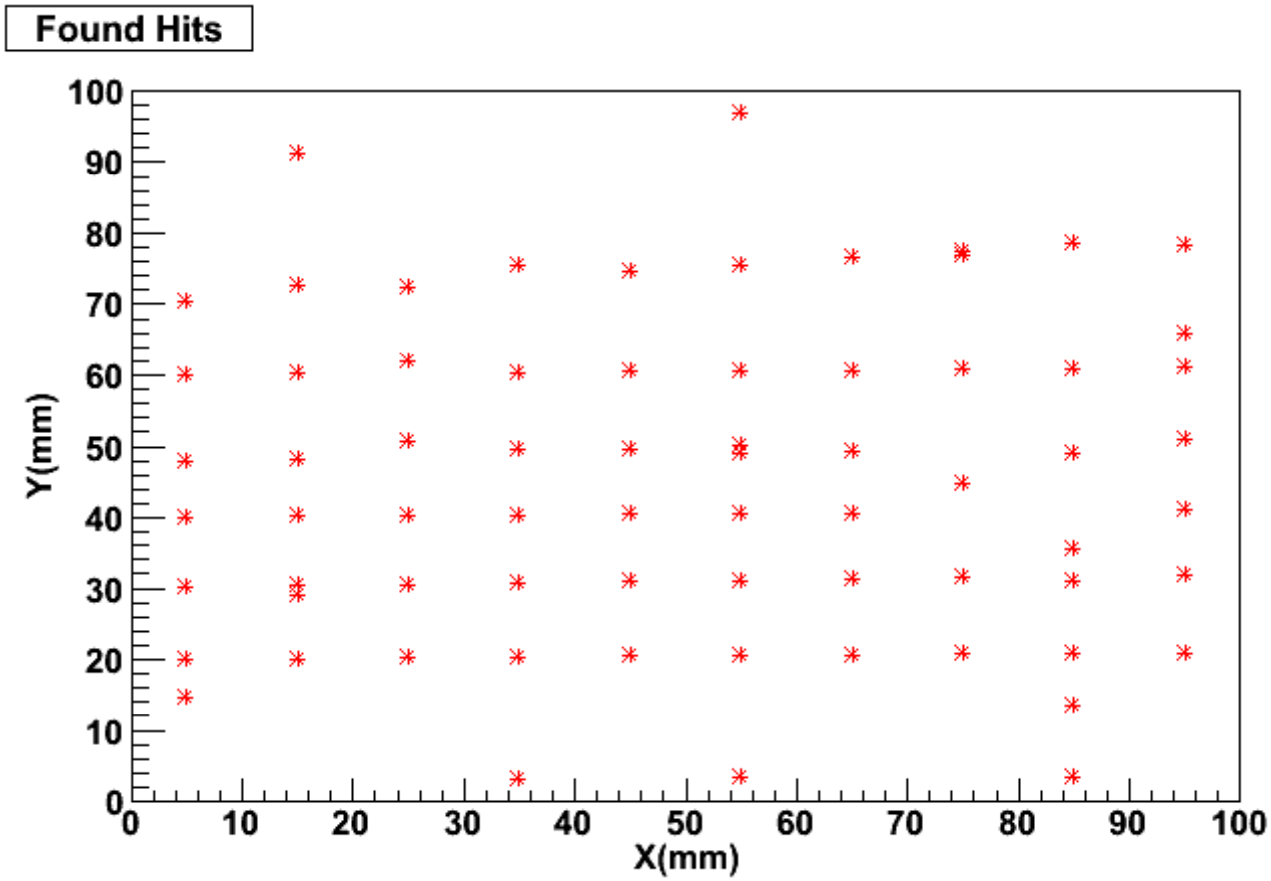


Test 2-3

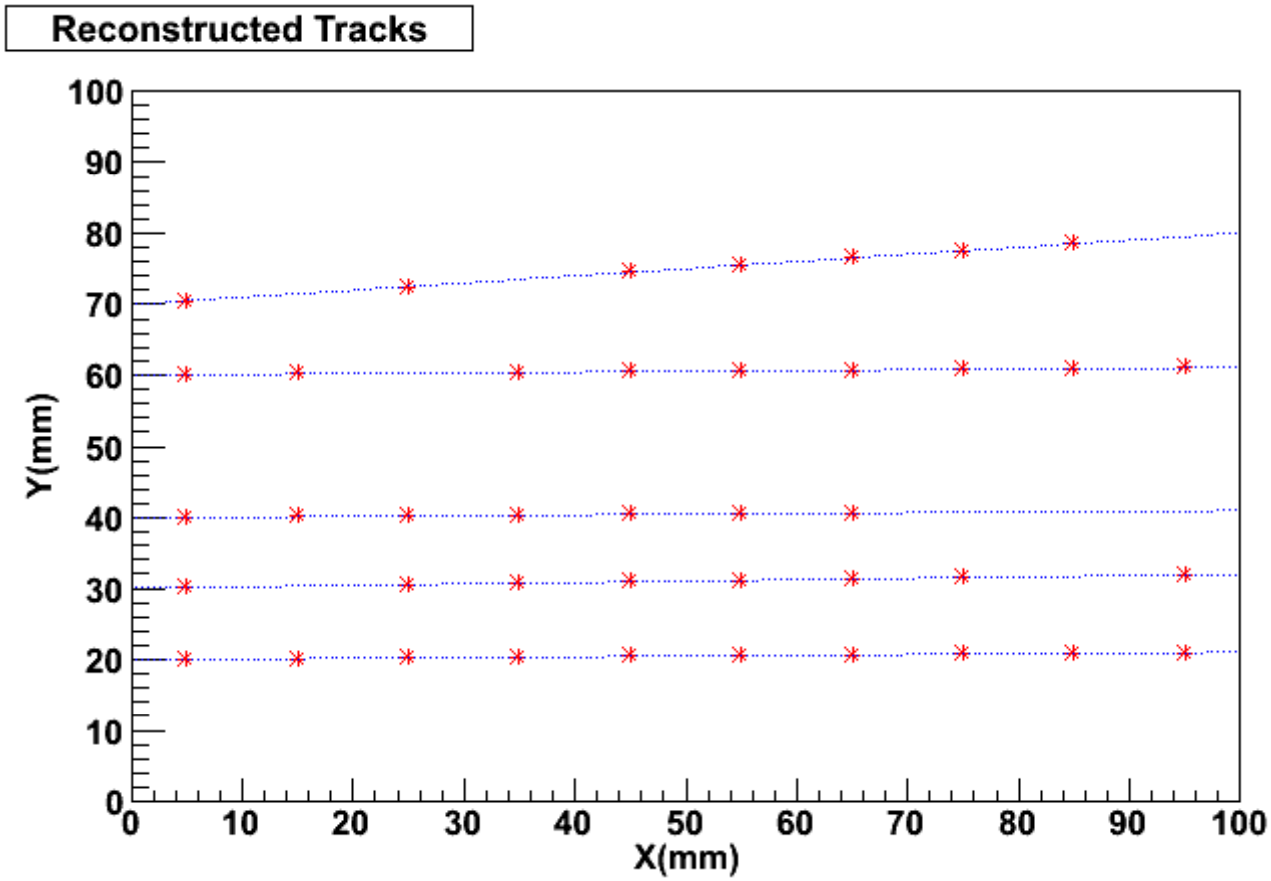


Test 2-4

Add 25 noise hits

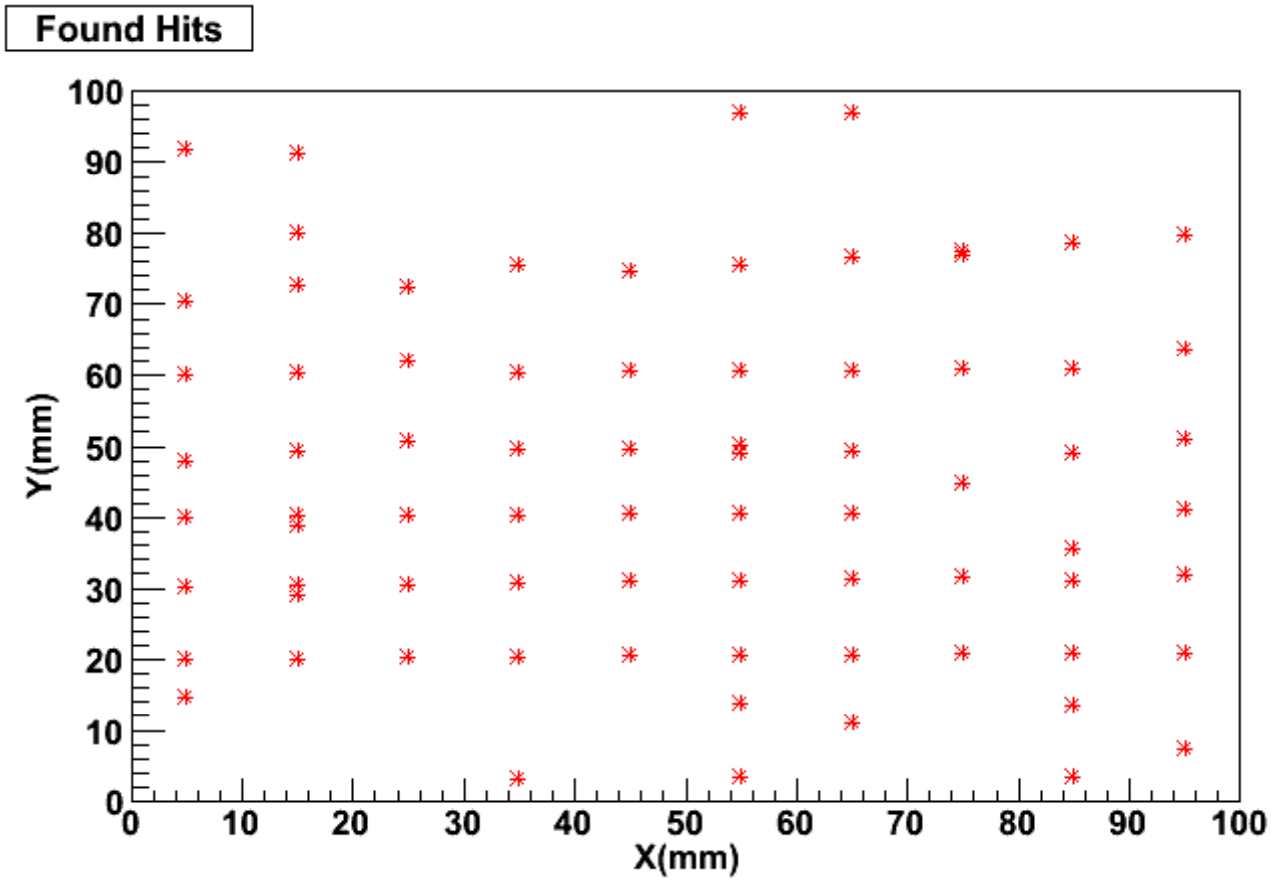


Test 2-4

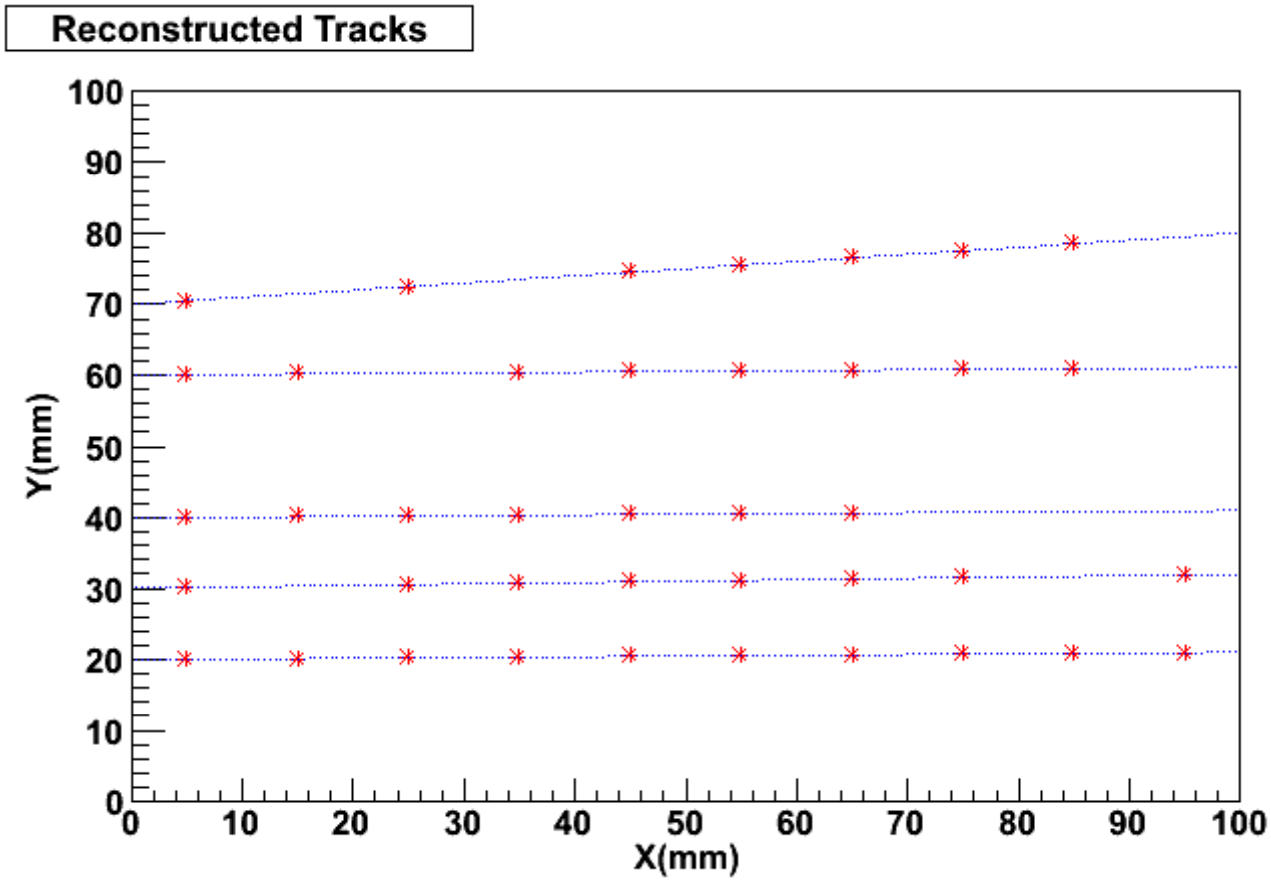


Test 2-5

Add 35 noise hits

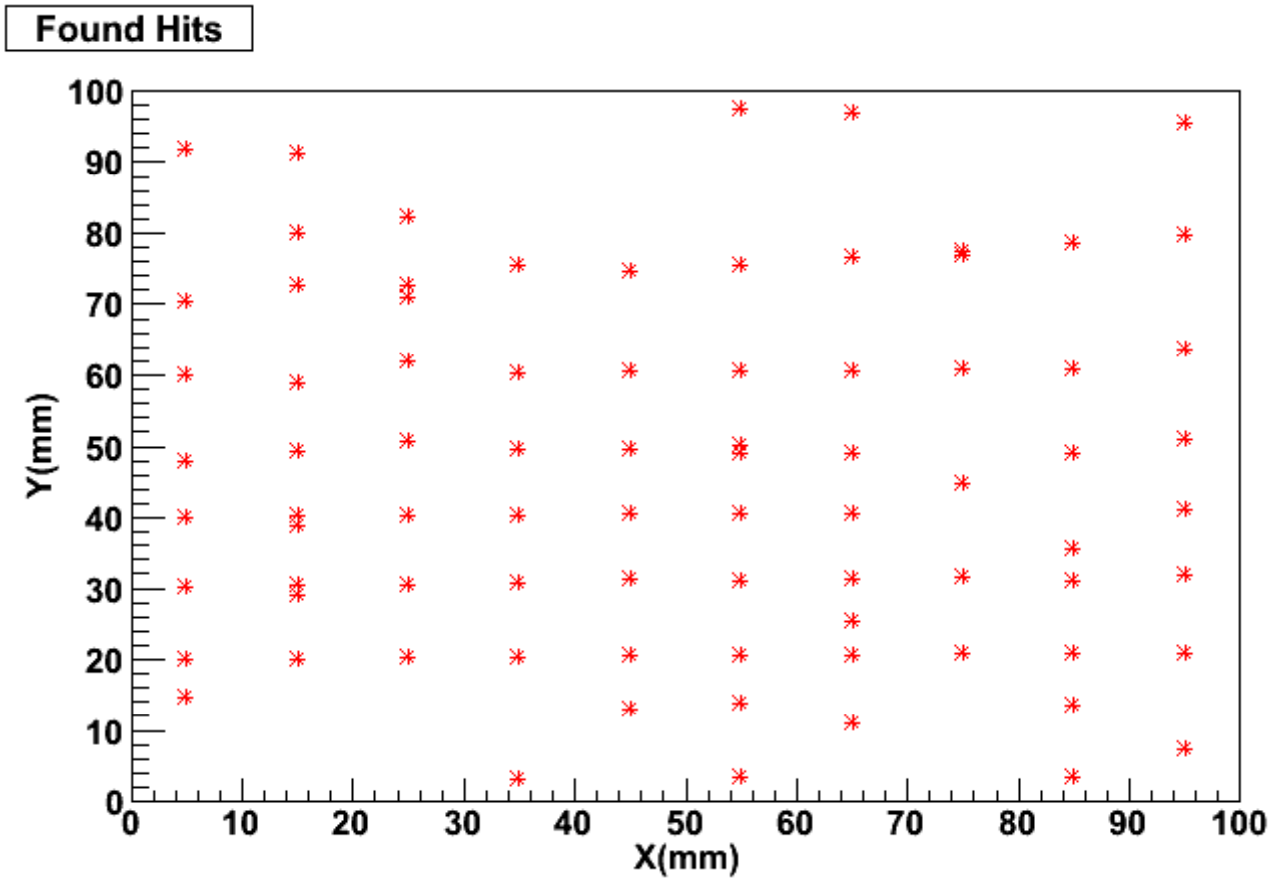


Test 2-5

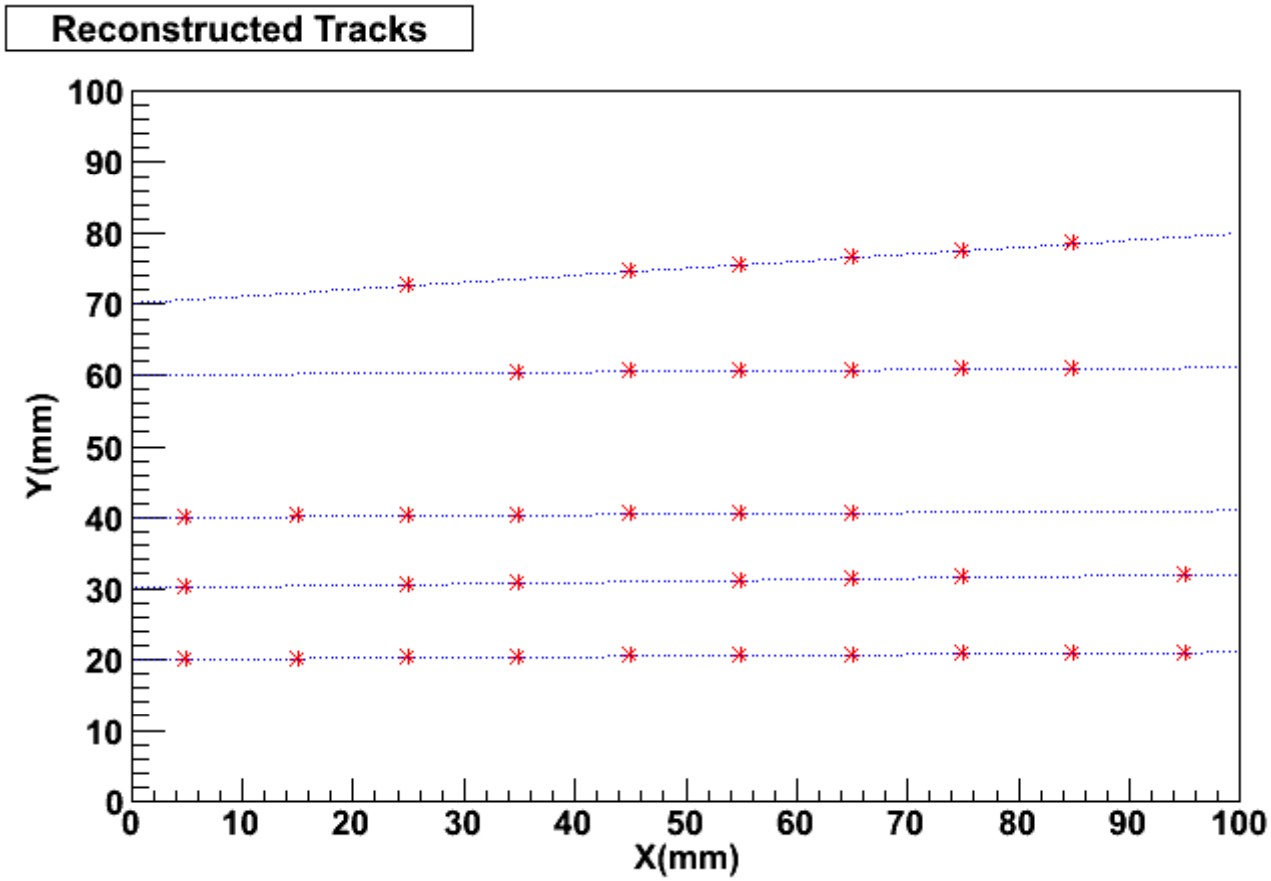


Test 2-6

Add 45 noise hits

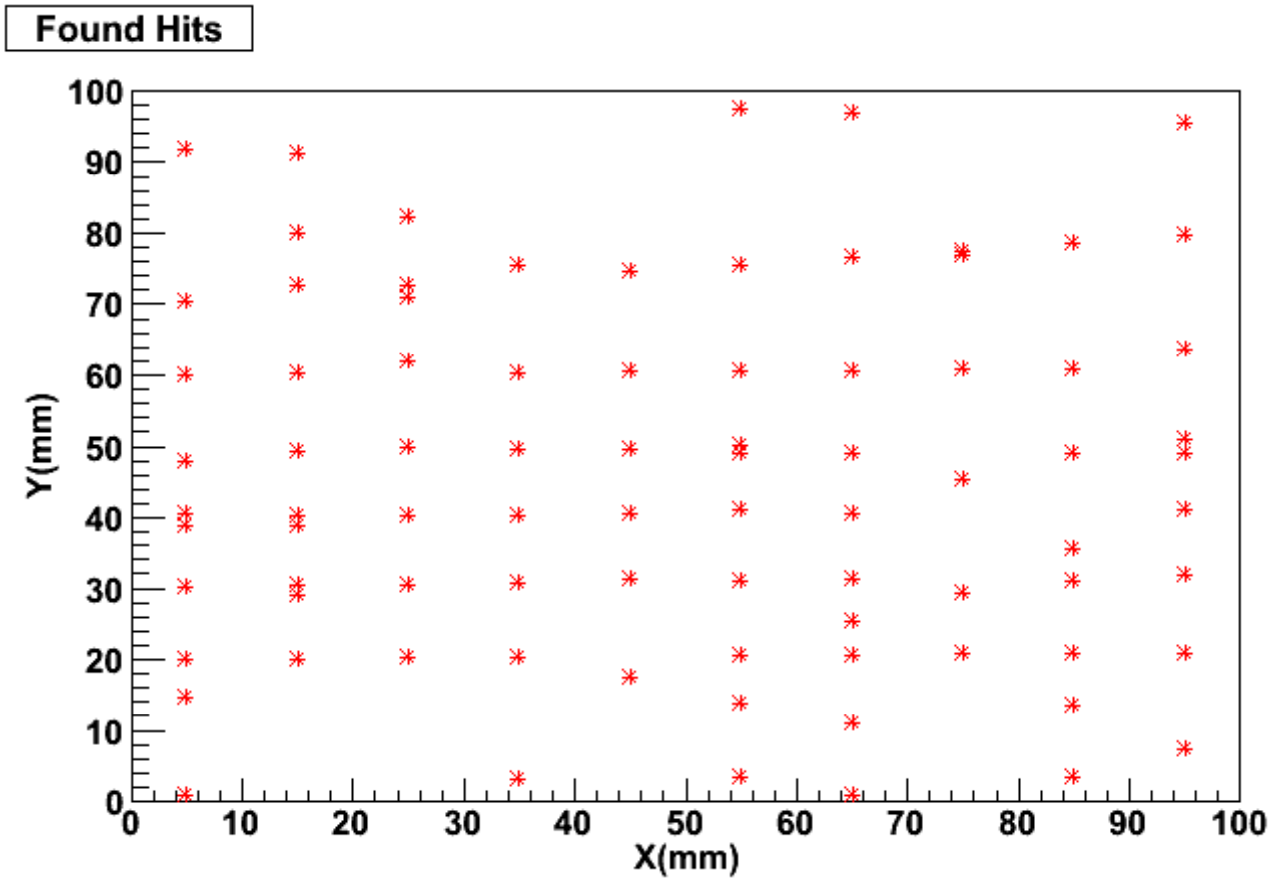


Test 2-6

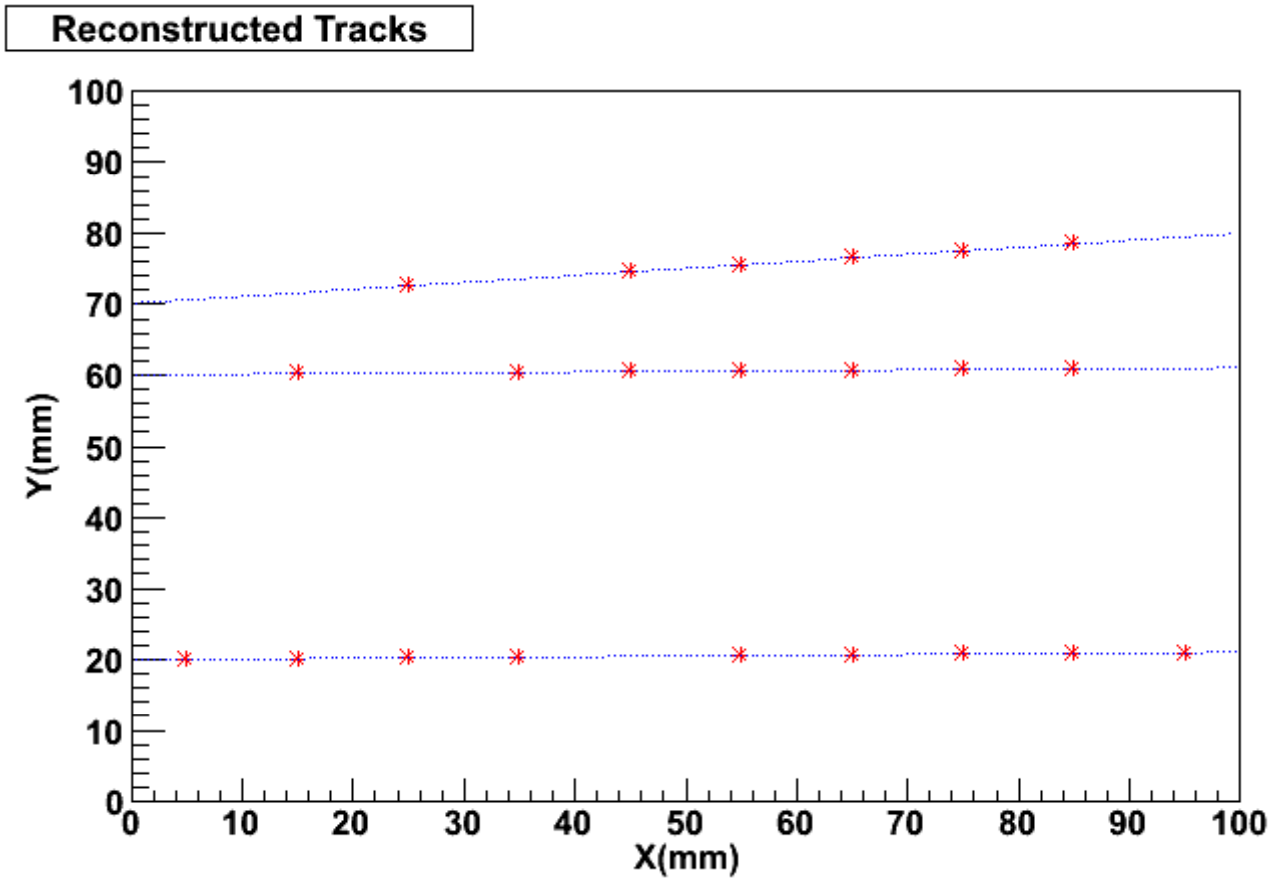


Test 2-7

Add 55 noise hits

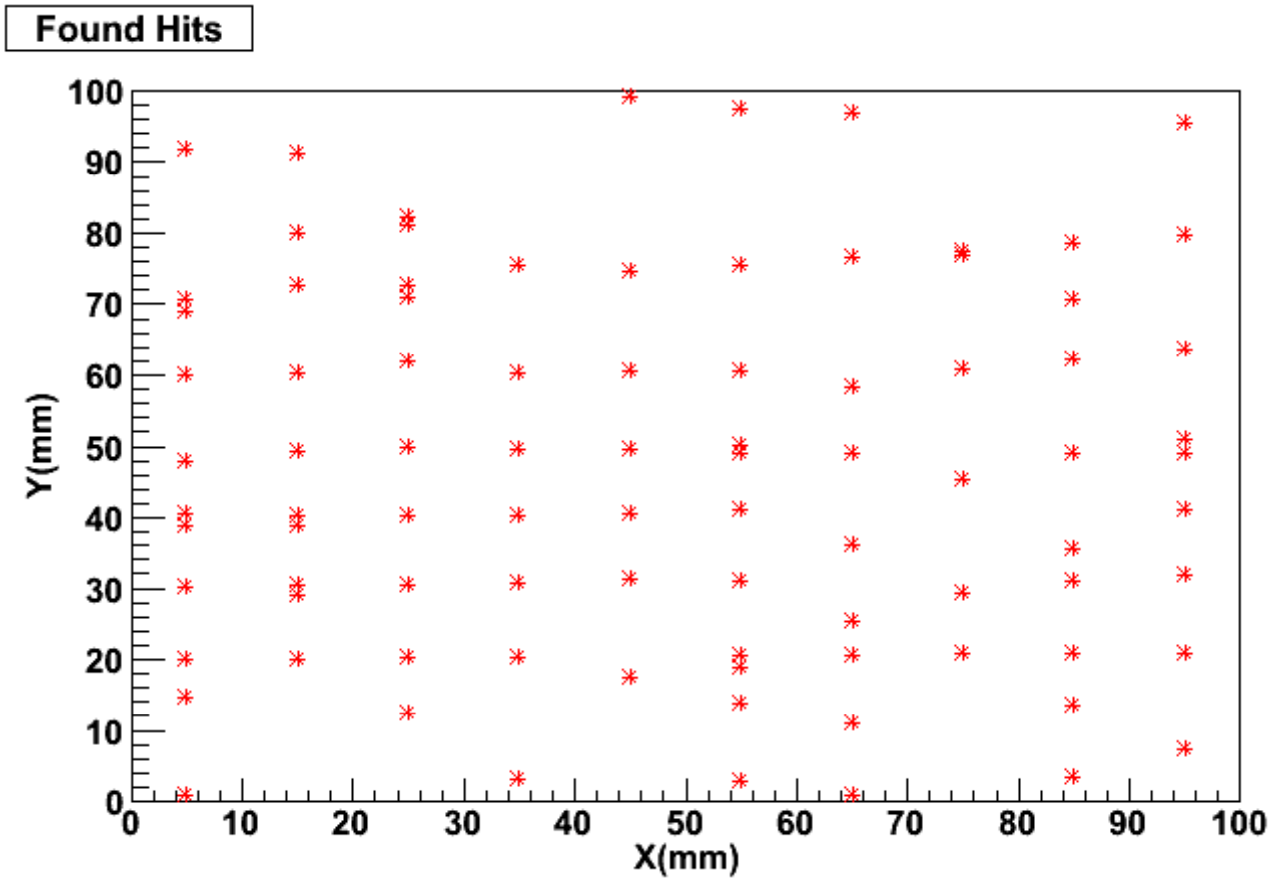


Test 2-7

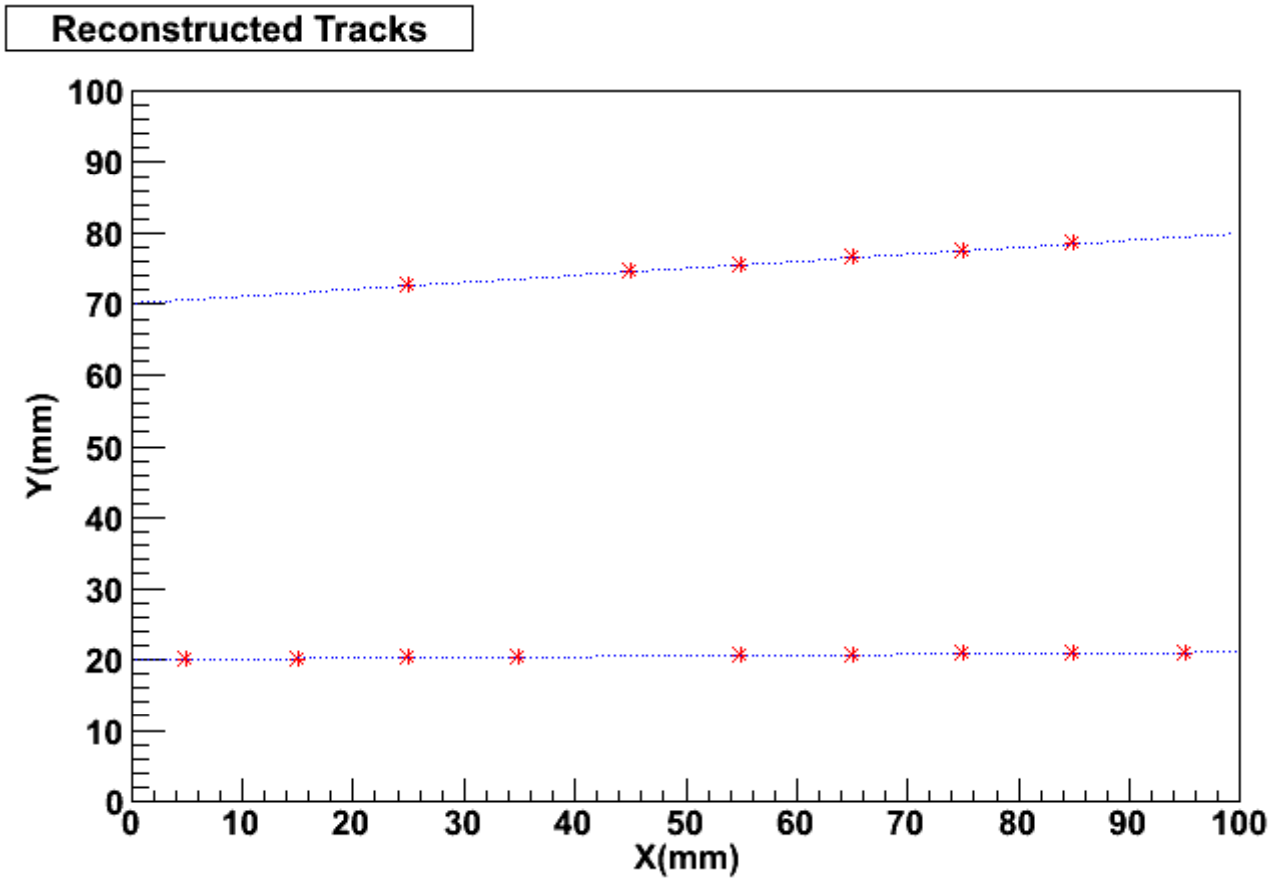


Test 2-8

Add 65 noise hits



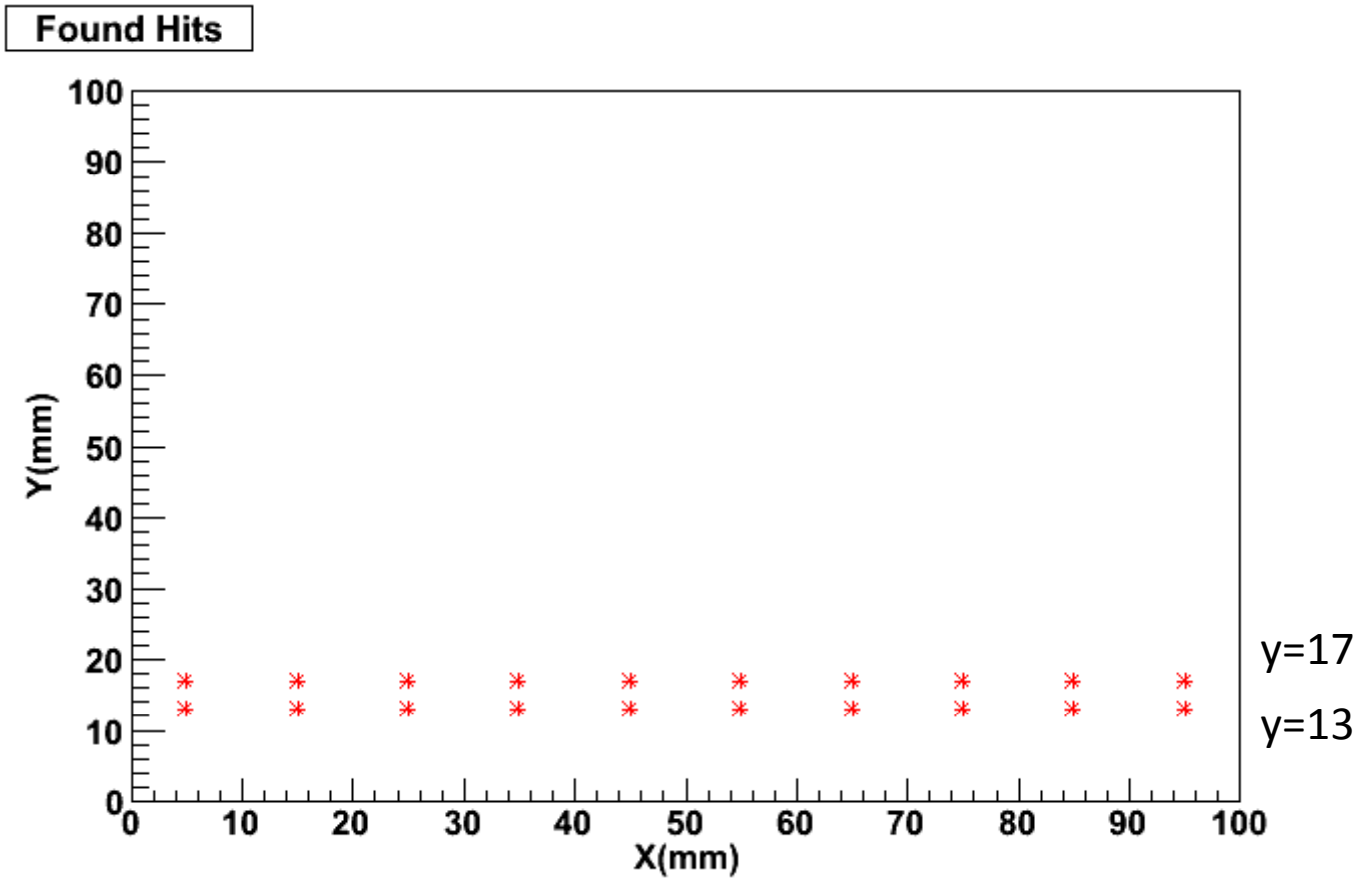
Test 2-8



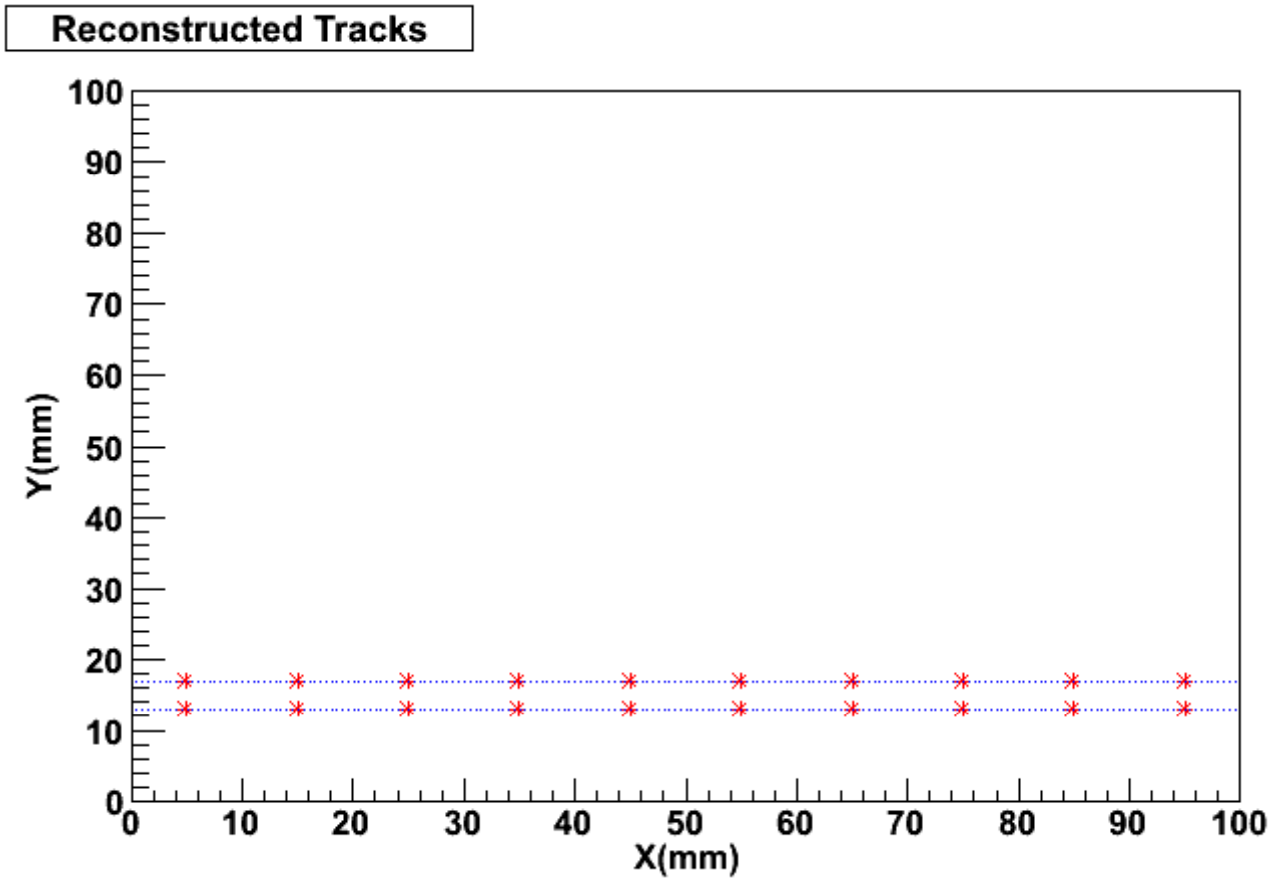
Conclusions from test 2

- Multi-track with noise hits can be recognized in different levels.
- With noise hits increasing, the number of found tracks decreases.

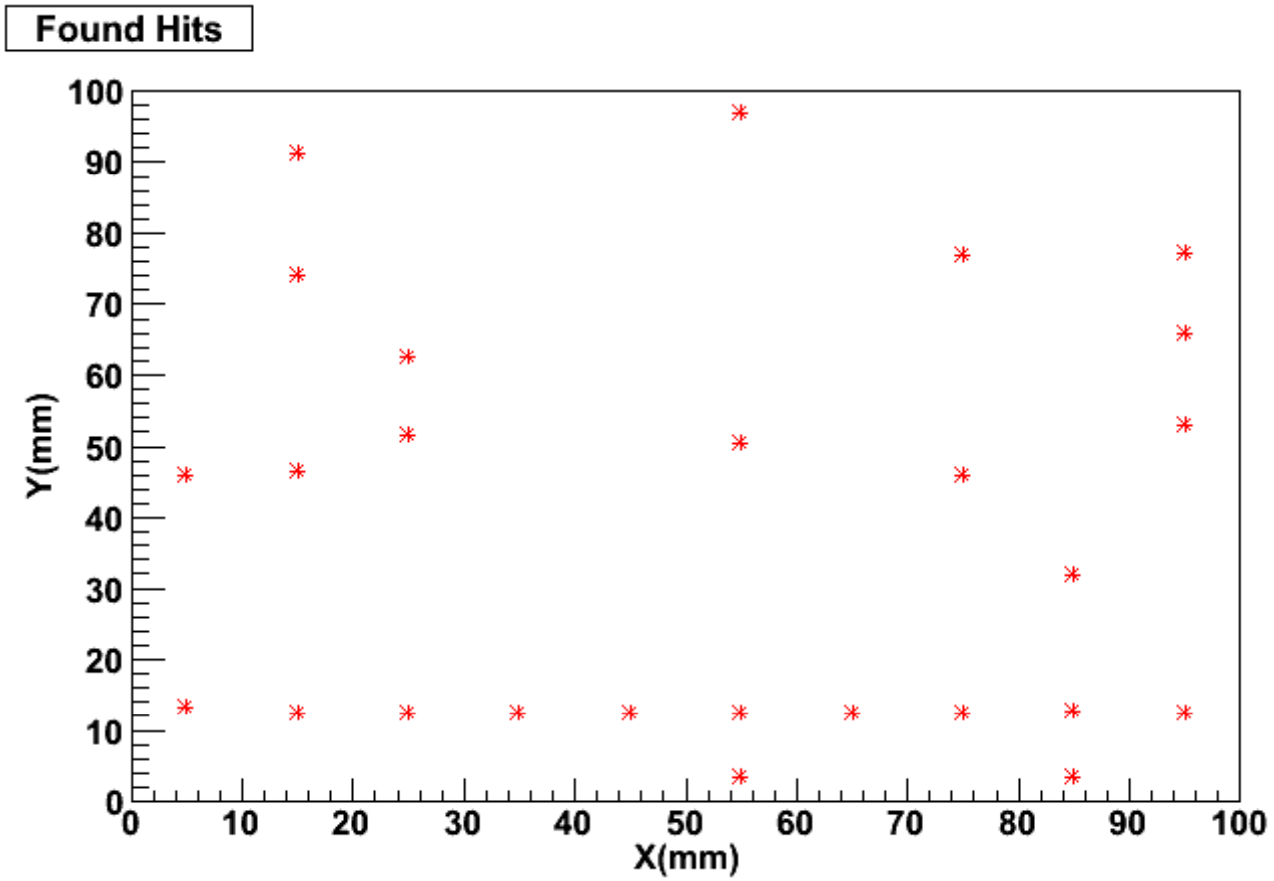
Test 3-1



Test 3-1



Test 3-2



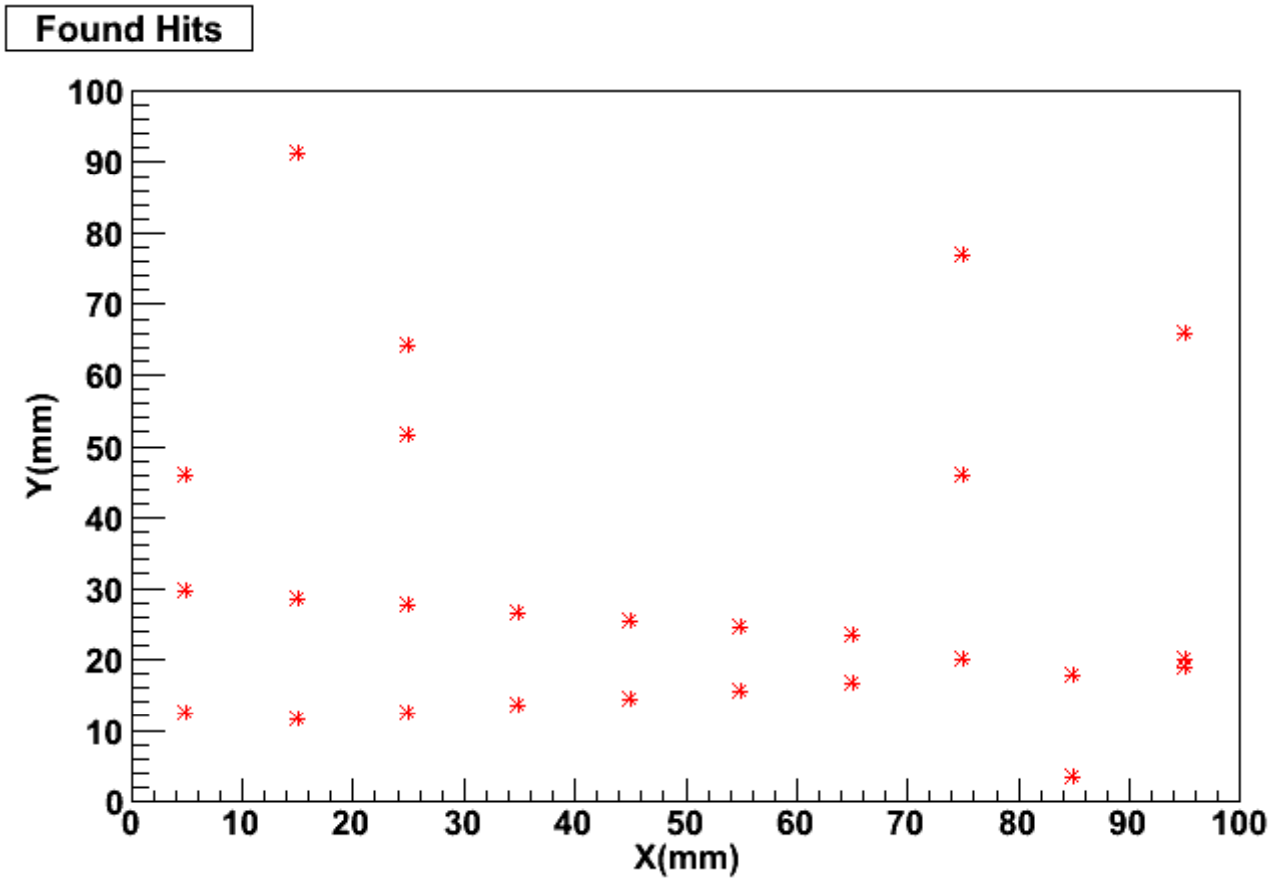
y=-0.01x+15

y=0.01x+10

The min. separate is 4mm in this environment?

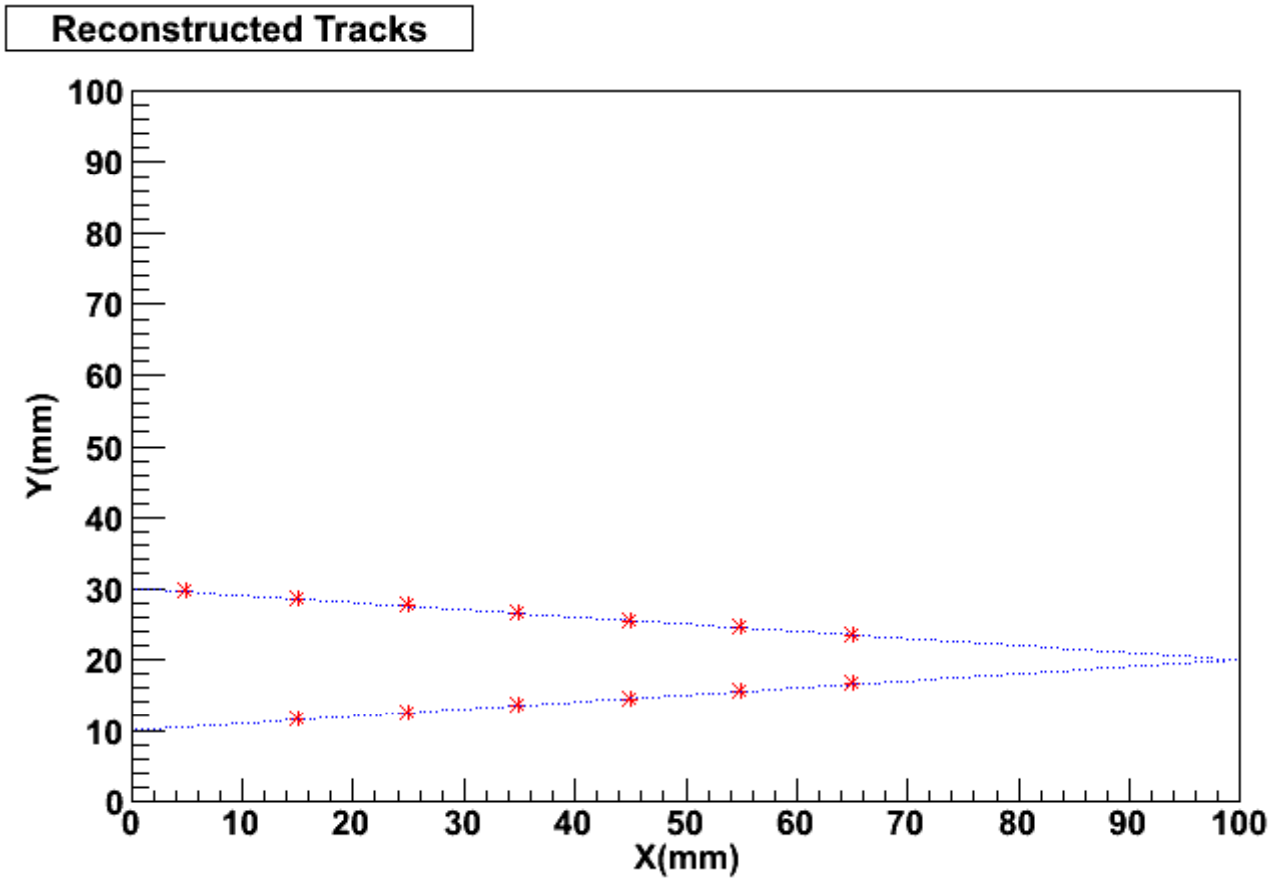
Hit reconstruction?

Test 3-3



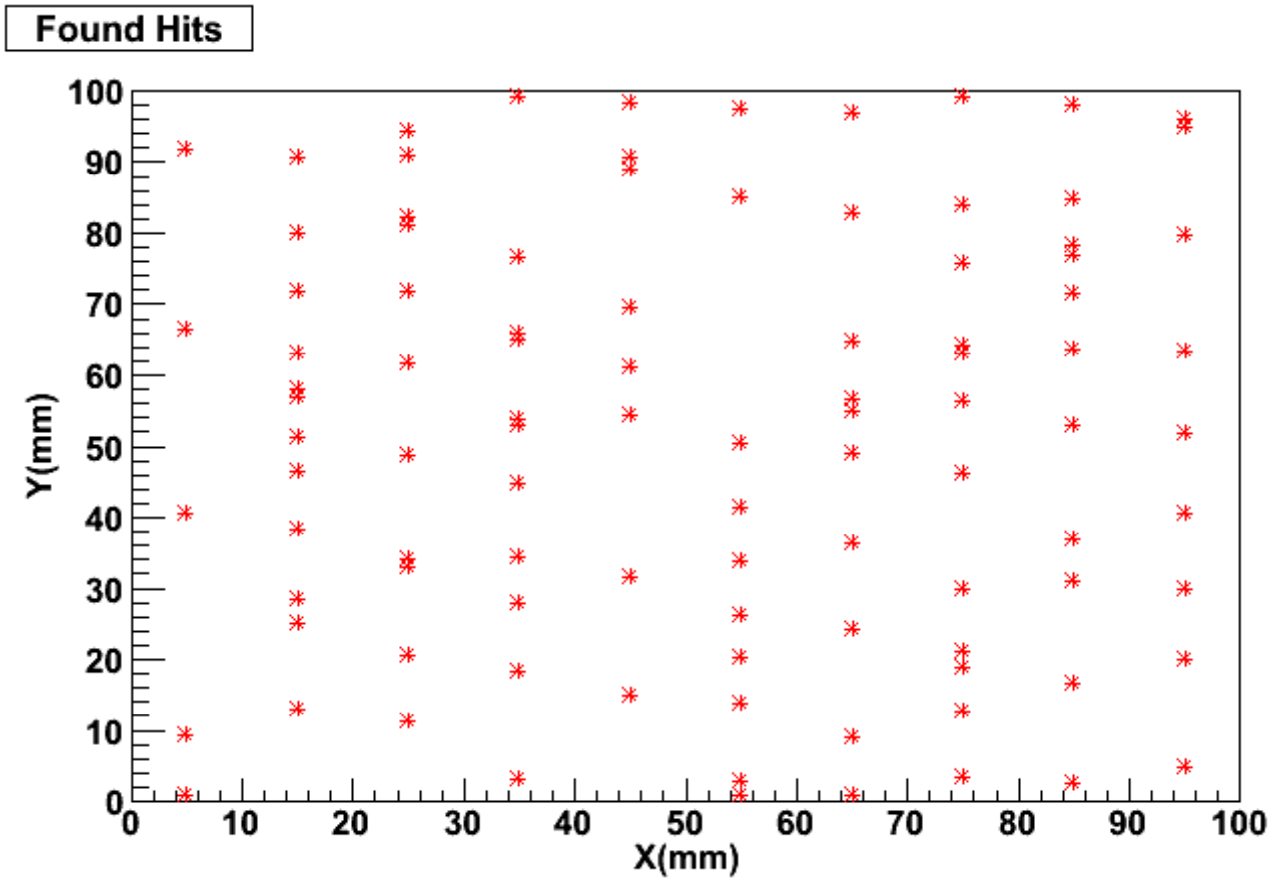
y=-0.1x+30
y=0.1x+10

Test 3-3



Test 4

200 noise hits



No track is found.

Conclusions from test 4

- In case the data is totally noise hit, track find code can get right result.
- But this progress will take more time for checking every noise hit.

Thank you!