УДК 519.673/6+539.1.08

TRACK FINDING WITH NEURAL NETWORKS IN ALICE ITS

B.V.Batyunya, Yu.A.Belikov, A.G.Fedunov, A.I.Zinchenko

A program based on Neural Networks technique was developed for track recognition in ALICE ITS. The efficiency of this program was estimated on samples of Monte Carlo simulated events for particles with $p_T \leq 100$ MeV/c. The advantages of this program are the high performance, the capability of track finding in conditions of the high particle multiplicity and the ability to find tracks distorted by the multiple Coulomb scattering and the energy losses.

The investigation has been performed at the Laboratory of High Energies, JINR.

Поиск треков в ITS установки ALICE с помощью нейронных сетей

Б.В.Батюня и др.

Описана программа для распознавания треков во внутренней трековой системе (ITS) установки ALICE в CERN. В основу программы положена модель нейронных сетей. Приводится оценка эффективности распознавания треков мягких ($p_T \leq 100$ МэВ/с) частиц, выполненная с помощью метода Монте Карло. Достоинствами описанной программы являются высокое быстродействие, возможность поиска треков в условиях большой множественности частиц, образующихся в ядро-ядерных столкновениях при высоких энергиях, и способность находить искаженные многократным кулоновским рассеянием и потерями энергии треки частиц с низким импульсом.

Работа выполнена в Лаборатории высоких энергий ОИЯИ.

1. Introduction

One of the most complicated problems during off-line processing of tracking detector data is the pattern recognition. After preliminary treatment, these data represent a large number of points in space. The problem is how to separate an initial set of unordered points into some subsets, which contain the points belonging to a given particle track.

Several years ago quite a new approach has begun to be developed. These are Neural Networks (*NN*) investigations. The *NN* technique has emerged from programming, mathematics and electronics [1,2]. Being applied to the pattern recognition problem, this method offers fast convergence and possibility to recognize tracks of any shape (see for example [3,4]). So we can create (in principal) one program for straight lines and helices and, may be, change only some control input constants, when switching on/off the magnetic

field. At last, it seems that the problem of finding kinks and distorted tracks from particles with large dE/dx — losses could be solved in this way also.

This note demonstrates the performance of the track recognition program based on the simplest kind of *NN*. The track finding capabilities were studied for the events obtained by a generation and the GEANT based simulation of the ALICE ITS.

2. Generation of Events and ALICE ITS Simulation

To generate the events of *PbPb* central collisions at LHC energy, the HIJING-code [5] was used. As the first step, we considered only the tracks with the limited transverse momenta ($p_T \leq 100$ MeV/c), which can be reconstructed only in the Inner Tracking System (ITS) of the ALICE detector [6], because they don't leave the ITS or have not enough a number of measurements in the outer tracker (TPC). Thirty events were generated with the mean number of charged particles equaled to ~400 with $p_T \leq 100$ MeV/c within the ITS acceptance, $|\eta| \leq 1$ (η is a pseudorapidity). We excluded the tracks with $p_T > 100$ MeV/c by assuming that these ones can be recognized in the TPC with an efficiency of ~100%.

Track finding capability of the ITS was evaluated using the GEANT-based simulation package with the performance parameters of six silicon cylinder layers. The layer radii are presented in the second column of the Table. The beam pipe with a radius of 3 cm and a thickness of 0.06 cm (~0.17%X_0 for Be) has been included as well. Multiple scattering in the beam pipe, silicon layers and also in mechanical support and cooling systems and in electronics was taken into account (~4.0%X_0). Secondary interactions and delta-electron production were included in the simulation. The tracks were processed in the magnetic field of 0.2 *T*. We used, as the hits, the GEANT explicit coordinates smeared by Gaussian distributions with the mean resolution values presented in the Table for cylindrical $r\,\varphi$ and z coordinates and for the silicon detectors of different types [7].

Table

Layer number	Radius (mm)	Detector type	$\sigma_{(r\varphi)}$ (mm)	σ_z (mm)
1	33	pixel	0.02	0.05
2	73	pixel	0.02	0.05
3	140	derift	0.03	0.02
4	240	drift	0.03	0.02
5	400	strip	0.03	1.00
6	450	strip	0.03	1.00

3. Recognition Program

Our recognition program was written on *C++* language. The main idea is similar to the one described in [4]. But because the mean distance between tracks in ITS is much less than the distance between consecutive points on a given track, we are forced to take into account the parametric form of expected tracks. We choose the track model as follows

$$\begin{cases} x(t) = R \cdot [\sin(\text{sign}(R)t + \varphi_0) - \sin(\varphi_0)] \\ y(t) = R \cdot [-\cos(\text{sign}(R)t + \varphi_0) + \cos(\varphi_0)] \\ z(t) = z_0 + R \cdot \operatorname{tg}\lambda \cdot t \end{cases} \qquad t \in [0, \pi] .$$

So our tracks are helices with the common vertex at $(0, 0, z_0)$, radius of curvature in (XY)-plane and emission angles φ_0 and λ (see Fig.1). Of course, straight lines crossing the vertex satisfy our track model.

The program consists of three steps:

1. reading data points from an input file and creating of the NN;
2. evolution of the NN (to the minimum of the network energy function);
3. exctraction of track from the NN when the minimum of the energy function has been achieved.

Because of the azimuthal symmetry of the events, we treat hits in the cylindrical coordinate system (r, φ, z). At the step of the **NN creation,** when all the points (hits) belonging to a given event (triplets of r-, φ- and z-coordinates) are in computer memory, we sort them in increasing order of r and φ (for fixed r). Then we create the neurons between pairs of the points if some conditions are true. These conditions are:

(a) the hits of the current pair are at the consecutive silicon layers;

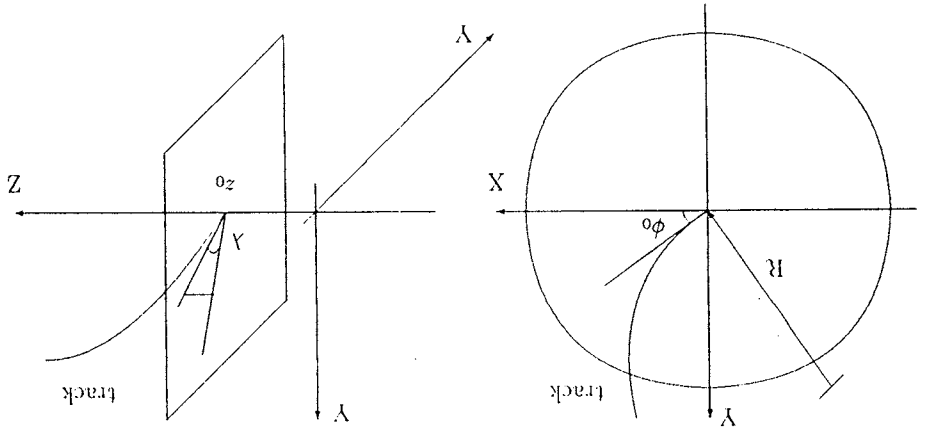

Fig.1

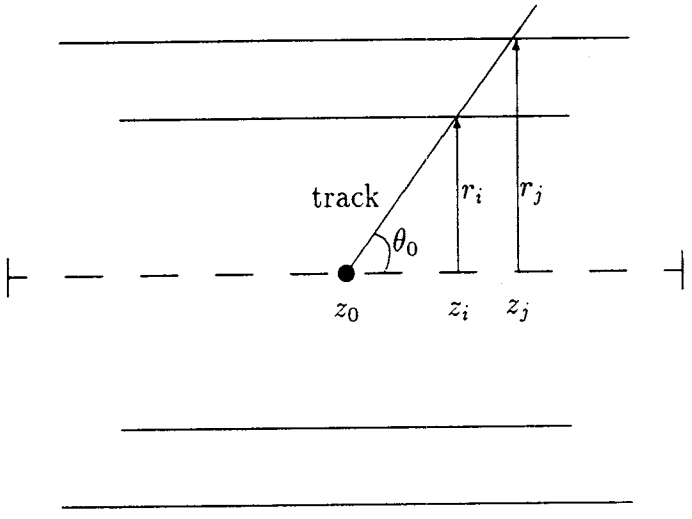

Fig.2

(b) circle diameter (in $r\varphi$-plane) passing through the tested hits and the vertex is greater than the threshold value $D_{\min}$;

(c) these two points and the estimated vertex $(0, 0, \hat{z}_0)$ should lie approximately on a certain helix.

If for any point there was no pair at the neightbour layer, the program tries to find it at the next cylinder.

Let us consider the last condition in more detail. First of all, we have to have the $\hat{z}_0$ value in spite of the absence of the reconstructed tracks at this step. To find the approximate $\hat{z}_0$ value, the following procedure is used. We can write for a straight line track (see Fig.2):

$$z_i = z_0 + r_i \operatorname{ctg} \theta_0, \tag{1}$$

where z_i is a z-coordinate of the crossing point of the layer by the track, θ_0 is a polar emission angle and r_i is the layer radius.

The equation (1) does not depend on azimuthal angle φ_0 therefore we can average it with respect to φ_0. The averaged values $\langle z_i \rangle$, $\langle z_0 \rangle$ and $\langle \operatorname{ctg} \theta_0 \rangle$ for two given cylinders with indices i and j satisfy the system of equations

$$\begin{cases} \langle z_i \rangle = \langle z_0 \rangle + r_i \langle \operatorname{ctg} \theta_0 \rangle \\ \langle z_j \rangle = \langle z_0 \rangle + r_j \langle \operatorname{ctg} \theta_0 \rangle. \end{cases}$$

This is a solvable system of two equations for two unknown values $\langle z_0 \rangle$ and $\langle \operatorname{ctg} \theta_0 \rangle$, because we can calculate $\langle z_i \rangle$ and $\langle z_j \rangle$ from input data.

In our case tracks are helices, but the difference between helices and straight lines is not essential if we get two innermost layers. We can solve the above system with respect to $\langle z_0 \rangle$ and take it as first $\hat{z}_0$ approximation.

The known property of the chosen track parametrization is $\frac{dz}{dl} = \text{const}$. So the increment of z-coordinates (dz) is strictly proportional to the corresponding arc lengths (dl) in (r, φ)-plane. But due to multiple scattering and measurement errors we have to use the inequality:

$$\left| \frac{z_1 - \hat{z}_0}{l_{10}} - \frac{z_2 - \hat{z}_0}{l_{20}} \right| < \varepsilon. \tag{2}$$

In this inequality ε is the program parameter and l_{10}, l_{20} are the length of the track (r, φ)-projection between the first point and the vertex and between the second point and the vertex, respectively. The condition (c) is true if the inequality (2) is fulfilled for the two tested points (r_1, φ_1, z_1) and (r_2, φ_2, z_2).

The values l_{10} and l_{20} can be estimated from the equations

$$\begin{aligned} l_{01} &= r_1 \\ l_{02} &= r_1 + l \end{aligned} \quad \text{for } \varphi_2 - \varphi_1 < 0.15,$$

$$\begin{aligned} l_{01} &= r_1 \cdot \alpha / \sin(\alpha) \\ l_{02} &= r_2 \cdot [\alpha + 2(\varphi_2 - \varphi_1)] / \sin[\alpha + 2(\varphi_2 - \varphi_1)] \end{aligned} \quad \text{otherwise.}$$

In these equations

$$l = \sqrt{r_1^2 + r_2^2 - 2r_1 r_2 \cos(\varphi_2 - \varphi_1)},$$

$$\alpha = 2\arcsin[\sin(\varphi_2 - \varphi_1) \cdot r_1 / l]$$

and $r_{1,2}$, $\varphi_{1,2}$ are the input data.

After the creation of the neurons we try to connect them by the following way. Let us mark a neuron created between points number i and j by the pair (ij). If the start point of one neuron is the same as the end point of another one, we connect these neighbour neurons with the weight

$$w_{ijk} = \cos^{\xi}(\theta_{ijk}),$$

where θ_{ijk} — the space angle between neurons (ij) and (jk); ξ — the program parameter.

We assign $w_{ijk} = 0$ when

$$\left| \frac{z_j - z_i}{l_{ij}} - \frac{z_k - z_j}{l_{jk}} \right| > \varepsilon,$$

or if the vector products

$$[\mathbf{r}_1 \times \mathbf{l}_{ij}] \quad \text{and} \quad [\mathbf{l}_{ij} \times \mathbf{l}_{jk}]$$

have the different signs. Here we used the next new designations: $\mathbf{l}_{ij} = \mathbf{r}_j - \mathbf{r}_i$ and $\mathbf{r}_i$, $\mathbf{r}_j$ are the radius-vectors of the hits number i and j in (XY)-plane.

The next step of the program is the **NN evolution**. We choose the equation of neuron activation renewal in the form

$$V_{ij} = \frac{1}{2}\left[1 + th \frac{\text{«}c - \text{term»} - \text{«}\alpha - \text{term»} + b}{T} \right] \tag{3}$$

where

$$\text{«}c\text{–term»} = c(\Sigma_m w_{mij} V_{mi} + \Sigma_n w_{ijn} V_{jn})$$

$$\text{«}\alpha\text{–term»} = \alpha(\Sigma_{l \neq j} V_{il} + \Sigma_{k \neq i} V_{kj});$$

(c, α, b and T are parameters of the program).

The neurons vary asynchronously their activations according to the equation of evolution (3). We stop iterations if the mean difference of the neuron activations between the iterations n and $n+1$ began to be small, i.e.

$$\frac{1}{N} \Sigma_{ij} \left| V_{ij}^n - V_{ij}^{n+1} \right| < 0.00001,$$

where N is the total number of neurons.

Our list of points is sorted in increasing order of r and φ. The procedure of **track extraction** is as follows. We take the point at the end of list, find the most excited incoming neuron, take the start point of this neuron and find the most excited incoming neuron for this point, and so on while incoming neurons exist. We mark these points as used and consider this chain as a track candidate. Then we return to the end of our list, take the next unused point and repeat the procedure unitl all the points were used.

The program parameters are tuned empirically, and depend on detector peculiarities. The following values of the parameters were found for the ALICE ITS.

- $\varepsilon = 0.04 \div 0.08$ for $D_{\min} = 800 \div 2500$ (mm)
- $c = 400$, $\alpha = 100$, $b \sim 0$, $T = 10^{-3}$;
- $\xi = 2$

The most critical parameter is ε. For example, if this one is not small enough, many unnecessary neurons are created, and the time of recognition (and the computer memory) increases while the recognition efficiency decreases. But if ε is too small, some of required neurons are lost, and some of tracks begin to be broken.

The optimal value of $D_{\min}$ depends on the momentum region of interest. If we take a greater value for this parameter, the recognition efficiency for tracks with $D > D_{\min}$ is greater (as compared with smaller $D_{\min}$), but the efficiency for tracks with $D < D_{\min}$ falls to zero.

4. Results and Discussion

To obtain the highest recognition efficiencies, we processed different p_T-regions separately, because of the large track distortion (by dE/dx) for lower momenta. Three passes of the recognition program for momentum regions of $p_T > 75$ MeV/c, $p_T > 50$ MeV/c and $p_T > 25$ MeV/c were done for each event. When any pass was completed the hits of the «found» (see below) tracks were removed from the input data and the program began the next pass with the new parameter values. Thus, we could find the maximum number of tracks for all p_T-regions. Then we got remained (the most distorted) tracks in the whole area $25 < p_T < 100$ MeV/c after the additional fourth pass. The following optimized values of $D_{\min}$ and ε parameters were obtained for all four passes:

1. $D_{\min} = 2500$ $\varepsilon = 0.04$,
2. $D_{\min} = 1666$ $\varepsilon = 0.04$,
3. $D_{\min} = 833$ $\varepsilon = 0.04$,
4. $D_{\min} = 833$ $\varepsilon = 0.08$.

After the recognition we considered a track candidate as a «*found*» track if

- track candidate consists of 4 or more hits;
- one of the hits must be from the external ITS layer;
- track candidate contains not more than one hit from another track.

The second condition is essential to connect tracks found in the TPC and ITS. If for any track candidate the first two conditions were fulfilled but the third one was not satisfied, we considered it as a «*fake*» track. The results of such a four pass procedure for the «found» tracks, summing up thirty events, are shown in Fig.3.

The recognition efficiency determination has been done separately for e^+/e^- and for π^+/π^- tracks, because of the special physical interest to the e^+/e^- pair production [6]. It should be noted also that very significant part of hadrons are lost because of the high dE/dx losses in the p_T-region under study. That is why we have calculated only π^+/π^- track recognition efficiency in case of hadron tracks.

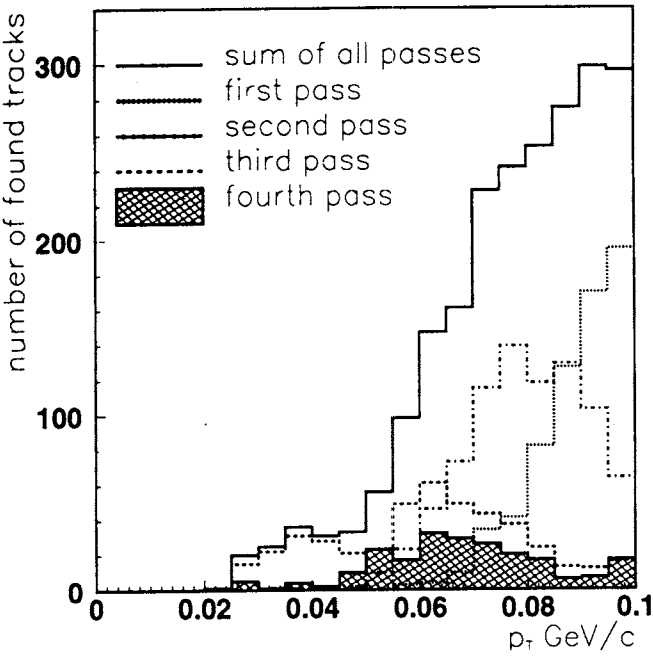
number of found tracks
300
200
100
0
sum of all passes
first pass
second pass
third pass
fourth pass
0
0.02
0.04
0.06
0.08
0.1
p_T GeV/c

Fig.3

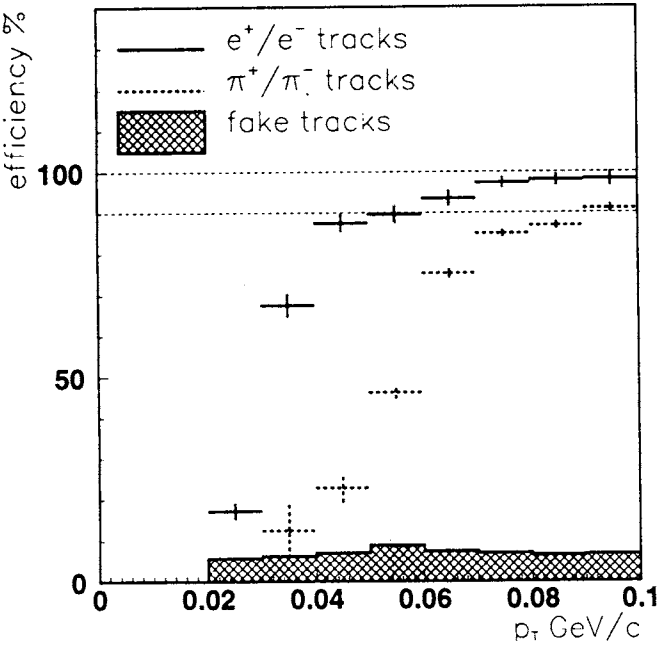
efficiency %
100
50
0
e^+/e^- tracks
π^+/π^- tracks
fake tracks
0
0.02
0.04
0.06
0.08
0.1
p_T GeV/c

Fig.4.

Thus we calculated efficiencies for electron, pion and fake tracks using the next formulae:

$$\zeta^e = \frac{N^e_{\text{found}}}{N^e_{\text{initial}}} 100\%,$$

$$\zeta^\pi = \frac{N^\pi_{\text{found}}}{N^\pi_{\text{initial}}} 100\%,$$

$$\zeta^{\text{fake}} = \frac{N^{\text{fake}}_{\text{found}}}{N^{\text{fake}}_{\text{initial}}} 100\%,$$

where N^e_{found}, N^π_{found} and $N^{\text{fake}}_{\text{found}}$ are numbers of the «found» e^+/e^-, π^+/π^- and the «fake» tracks, respectively. The N^e_{initial}, N^π_{initial} and N_{initial} are numbers of the simulated electron, pion and all charged particle tracks satisfied to the first and second conditions considered above and related to the primary vertex. It is clear that the secondary tracks (related to secondary vertices or delta electrons) play only the role of a «noise» for our recognition program.

Figure 4 shows ζ^e, ζ^π, and ζ^{fake} as function of p_T. It can be seen from this figure that the efficiency for electron tracks is about $80 + 100\%$ at $p_T \geq 40$ MeV/c and the efficiency for «fake» tracks is less than 10% in the whole region of p_T. We can see also that the ζ^π is significantly worse for pions at $p_T \leq 70$ MeV/c as a consequence of the track distortion by dE/dx.

One of the most important properies of all recognition programs is computer memory and time consumption. This program needs about 500k RAM and spends about 5 CPU sec per event (with ~400 tracks and ~6 hits per track) on CONVEX 220 or about 30 CPU sec per event on AT386 33 MHz.

Acknowledgements

We are very grateful to K.Safaric, S.Khorozov for useful discussions and suggestions and to N.Slavin for his help with event generation.

References

1. Treleaven P., Vellasco M. — Comput. Phys. Commun., 1989, v.57, p.543.
2. Humpert B. — Comput. Phys. Commun., 1990, v.58, p.223.
3. Dendy B., Linn S. — Comput. Phys. Commun., 1990, v.56, p.293.

4. Stimpfl-Abele G., Garrido L. — Comput. Phys. Commun., 1991, v.64, p.46.
5. Wang N.X. et al. — Phys. Rev., 1991, v.D44, p.3521; Phys. Rev. Lett., 1992, v.68, p.1480.
6. Letter of Intent for a Large Ion Collider Experiment, CERN/LHCC/93-16, LHCC/14, March 1993.
7. Batyunya B., Zinchenko A. — Internal Note/SIM ALICE/94-31, 1994; JINR Rapid Communication, 1995, No.3[71]-95, p.5.

Received on May 14, 1996.