Charged Particle Track Reconstruction using Artificial Neural Networks

Charles Glover, Peter Fu, Tony Gabriel

Oak Ridge National Laboratory
Oak Ridge, Tennessee 37831 USA

Thomas Handler

Physics Dept.
University of Tennessee
Knoxville, Tennessee 37996-1200 USA

Abstract

This paper summarizes the current state of our research in developing and applying artificial neural network (ANN) algorithms to reconstruct charged particle tracks from tracking chamber data. The ANN algorithm described here is based on a crude model of the retina. It takes as input the coordinates of each charged particle's interaction point ("hit") in the tracking chamber. The algorithm's output is a set of vectors pointing to other hits that most likely to form a track.

1. Introduction

Two general observations can be made about tracking problems: algorithms to achieve global solutions are non-polynomial (NP) in complexity and it is a two stage process.

The global solution to the tracking problem is NP-Hard. The complexity of an optimal tracking algorithm scales as a polynomial of order n, where n represents the number of measured points (e.g., measured points may represent hits in a central tracking chamber). These classes of problems are intractable in general for large n.

To illustrate the tracking problem's complexity, note that three or more labels can be attached to a measured point: the point is a new detection (ND), the point could be a false detection (FD), or the point could be a member of an existing track (T_i). The 3^2 is the minimum cardianlity of a hypothesis set for two measurement points. The cardinality of a hypothesis set for n points is of order 3^n (see Ref. [1] for a full discussion of hypothesis trees). Clearly, the growth of the hypothesis tree must be limited and pruned. A wealth of literature exists (see Ref. [1]) on techniques to do this. However, these methods require large computers and are not suitable for triggers in detector systems.

Artificial Neural Network (ANN) paradigms are completely parallel algorithms that do not require large computers to implement; they can be implemented in VLSI or optical processing circuits. Thus, ANN tracking algorithms offer the potential to be incorporated into detector trigger systems. This is part of the motivation behind the research reported in this paper.

The solution to the tracking problem is a two stage process. In the first stage, the hypothesis set is pruned according to some heuristics (i.e., Kalman Filtering, tracks form trajectories of constant curvature, etc.). The second stage assigns a score (usually a probability measure) to each surviving hypotheses and selects the highest scoring hypothesis as the solution to the tracking problem.

ANN tracking algorithms also require two stages to provide tracking solutions. To date, most of the published ANN tracking algorithms [Ref. 2] have concentrated on the second stage; selecting a high scoring hypothesis. These algorithms start with an *a priori* set of segments and link the segments according to a set of constraints to form a set of tracks. First, the hypothesis space is reduced by combining points into a segment according to some good heuristics such as locality – two points form a segment if they are 'near' each other, and if the segment they form points away from the collision vertex. This preprocessing is essentially a sequential operation and not easily amenable to hardware. Next, the ANN algorithm assigns segments from this set to a set of tracks until some optimality conditions is satisfied, such as the ANN's energy is minimized.

This paper reports the status of our research to find a completely parallel ANN algorithm that is amenable to hardware that will prune the hypothesis space. This should provide a first stage to an ANN tracking trigger.

2. Retinal Tracking Algorithm

To reduce the hypothesis space an ANN tracking algorithm must incorporate locality and direction constraints. The ANN algorithm used in this research is based on a crude model of the retina and its implementation to multi-target tracking by Kuczewski[3].

The retina incorporates locality and direction through oriented receptive fields. In a highly simplified view, one can view the retina as several layers of direction sensitive neurons. Neurons within a layer form a field of oriented neurons. Neurons in this oriented field are sensitive to motion in one particular direction. We can represent this oriented field as a vector field where each neuron has a unit vector pointing in the direction of its orientation. If an object is moving with a sufficiently large velocity component in the direction of the field's orientation vector then the neuron will fire. Its firing rate or the magnitude of its output will be some function of the dot product between the object's velocity and the neuron's orienting vector. Overlaying this field are several other layers of oriented fields. No two layers are sensitive to motion in the same direction. Thus, two differently oriented neurons will produce equal magnitude outputs if an object's velocity bisects two neuron's orienting vectors. By projecting these oriented vector fields on a plane, one can picture the retina as a finite set of unit vectors located at each point in a plane (see Fig. 1).

Locality is established by allowing neurons in a finite size neighborhood to affect the neuron centered in the neighborhood. Consider the i^{th} neuron located at a given coordinate that is sensitive to motion in the μ direction (μ = north, say). The neighborhood N_μ for this neuron is shown in Figure 1. The input to the neuron from the other neurons in the neighborhood can be expressed by

$$I_i^\mu(t) = \sum_k^{N_\mu} W_{ik}^\mu \, v\left[x_k^\mu(t)\right] \qquad (1)$$

where W_{ik} is a weighting function, $x(t)$ is the neuron's state at time t, and $v(\bullet)$ is neuron's output and is taken to be a sigmoid function. The form of the weighting function is taken from Kuczewski[3] as

$$W_{ik}^\mu = \exp\left[-\alpha\left(\Delta\vec{r}_{ik}\right)^2 \quad + \quad -\beta\left(\cos\Delta\theta_{ik}\right)^2\right]$$

where $\boldsymbol{r}$ is the vector connecting neuron i and k, and θ is the angle between $\boldsymbol{r}$ and i^{th} neuron's oriented unit vector. A neuron remains active as long as its neighbors remain active. In addition, the state of each neuron is affected by its counterpart $v(x_i^\nu)$ in the other layers. This competition can be expressed as

$$C_i(t) = -\sum_{\nu \neq \mu}^{Layers} W_{ii}^{\nu}\, v\left[x_i^{\nu}(t)\right] \qquad (2)$$

The competition tends to reduce a neuron's output $v(x_i^{\mu})$. As a consequence, the neuron with the largest output, $v(x_i^{\mu}). > v(x_i^{\nu})$; $\mu \neq \nu$, will shutdown the other oriented neurons at a given point on the plane. Thus, we perceive an object moving in only one direction.

Every neuron's state x_i^{μ} is affected by the output of its neighbors within a layer and its counterparts in other layers. The equation describing the evolution the neuron's state is given by

$$\dot{x}_i^{\mu}(t) = -a x_i^{\mu}(t) + b I_i^{\mu}(t) + c C_i(t). \qquad (3)$$

The equation of motion for the neuron's state is obtained by integrating (3).

This ANN tracking algorithm was applied to the charged particle tracking problem for central tracking chambers. A set of oriented neurons was associated with straw tube in small region of the chamber (see Fig. 1). A value of 1.0 was assigned initially to all oriented neurons associated with straw tube that contained a hit. All other neurons were assigned 0.0. A straw tube was then picked at random and equation (3) was integrated for all associated neurons. This cycle was repeated until the outputs of all neurons stabilized (asynchronous updating). All vectors shown in Fig. 2 represent the output from oriented neurons. The vector's length is proportional to the neuron's output and it points in the direction of the oriented neuron's unit vector. This vector forms a segment. Other ANN tracking algorithms can use this set of segments to form charged particle tracks.

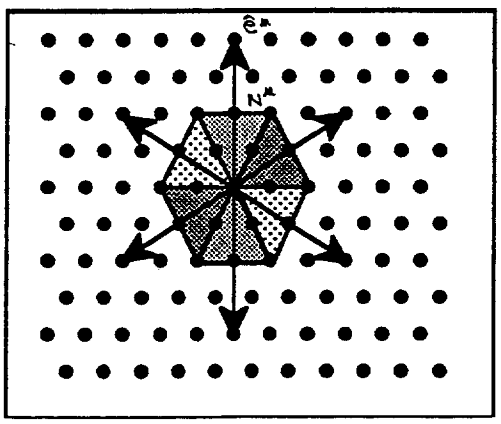

Figure 1

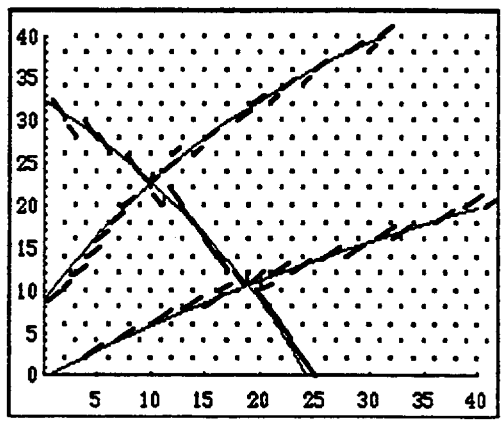

Figure 2

3. References

1. Samuel. S. Blackman, *Multiple-Target Tracking with Radar Applications*, Artech House, Norwood, MA (USA), (1986). See references within.

2. Georg Stimpl-Abele, "Fast Track Finding with Neural Networks", accepted for publication in Comp. Phys. Comm. (1991).. See references within.

3. Robert M. Kuczewski, "Neural Network Approaches to Multi-Target Tracking", *IEEE First International Conference on Neural Networks*, Maureen Caudill and Charles Butler (eds.) IEEE, Piscataway, NJ (USA), (1987).