December 1992
LU TP 92-28

Extensions and Explorations of the Elastic Arms Algorithm

Mattias Ohlsson[1]
Department of Theoretical Physics,
University of Lund
Sölvegatan 14A,
S-22362 Lund, Sweden

(Submitted to *Computer Physics Communications*)

Abstract:

The deformable templates method for track finding in high energy physics is reviewed and extended to handle multiple and secondary vertex positions. An automatized minimization method that handles different types of parameterizations is derived. It is based on the gradient descent method but modified with an explicit calculation of the natural metric. Also a simplified and more intuitive derivation of the algorithm using Potts mean field theory equations is given.

[1] mattias@thep.lu.se

1 Motivation and Results

An important problem in the area of pattern recognition and computer vision is detection of curves. In the context of high energy physics track finding represents such a problem. This is a combinatorial optimization problem; given a set of detector signals reconstruct particle trajectories subject to smoothness constraints. In most cases, the parametric form of the tracks are known in advance - straight lines or helices.

Artificial Neural Networks (ANN) techniques, and variations thereof, have shown great power in finding approximate solutions to difficult combinatorial optimization problems [1, 2]. The ANN technique has also been used for the track finding problem with encouraging results. In the pure neural approach [3, 4] one considers tracks as a number of consecutive line segments. The idea is to assign a binary decision element (neuron) s_{ij} between two signals points i and j, which is *one* if i is connected to j and *zero* otherwise. An energy function in terms of s_{ij} is then constructed such that smooth tracks with no bifurcations correspond to minima. Initial explorations have given encouraging results with respect to solution quality for toy-sized problems with no noise [3, 4]. In ref. [5] realistic cuts on the number of degrees of the freedom were made on real TPC data from the CERN ALEPH detector and the performance turned out to be compatible with the conventional method used in the ALEPH detector, both with respect to quality and speed for large problem sizes. Another approach for the track finding problem is the use of rotors [6]. A rotor $\vec{s}_i$, which is a unit vector, is associated with each signal point i. In a local approach [7] each rotor interact with other rotors (within some neighborhood) and with line segments vectors $\vec{r}_{ij}$. Again, an energy function is constructed that lines up the rotors to form smooth tracks.

The above methods are constructed to solve the combinatorial part of the track finding problem, i.e. to assign signal points to tracks. In reality one also needs to know the momenta corresponding to the tracks. In the neural approach one then has to augment the algorithm with some fitting procedure. In most track finding problems the parametric form of the tracks are know in advance, but in the neural approach the network has no such prior knowledge. It would be advantageous to have an algorithm that does both the assigning and the fitting simultaneously.

One step in this direction has been taken in ref. [8] where the rotor approach [7] has been modified to take into account the parametric form of the tracks. This algorithm has been tested on real data with encouraging results.

This paper is based on the deformable templates [9], or the elastic nets [10], approach. The basic idea of this approach for track finding was presented in ref. [11] and in ref. [12] it was developed further with respect to theoretical understanding and experimental applications. Another, closely related, approach was independently pursued in ref. [13].

In this paper the elastic arm approach is extended and generalized in different directions with the following main results:

- The main task of the elastic arms algorithm is to minimize the energy function E_{eff}. The efficiency of the algorithm depends on how fast it converges towards the global minimum of E_{eff}. In ref. [12] we used the standard gradient descent method but with individual updating parameters[2]. In this paper we modify the gradient descent method as to compensate for the different metric imposed by the parameters describing the track. As a result the convergence towards the minimum is faster and the number of updating parameters are reduced to one, it is basically a "black box" algorithm.

- Furthermore, we show how the elastic arms algorithm also can be understood more intuitively in the standard mean field theory. In this treatment *zero*-neurons, with the corresponding V_{i0} as the probability of i being a noise signal, appear in a very natural way. Thus, an event can be viewed as a set of real tracks with a given parameterization plus an extra "noise-track" consisting of all noise signals.

- Even though "real" data was used in the simulations of ref. [12] we assumed that all tracks passed trough the *a priori* known origin. In reality there may be more than one vertex position and also secondary vertex positions coming from decaying particles. We therefore extend the formalism to include multiple and secondary vertex positions.

- A complication that may occur in some detectors is the presence of mirror charges in the data set. Following ref. [14] we describe how to handle this problem.

The paper is organized as follows: In Section 2 we review the elastic arms algorithm plus a simplified derivation using Potts mean field theory equations. Section 3 deals with geometry and parameterization and in section 4 we modify the gradient descent method by metric considerations. The necessary extensions of the algorithm to handle multiple and secondary vertex positions are presented in Section 5. Section 6 contains a small discussion concerning the initialization of the algorithm and finally a brief summary can be found in section 7.

[2] In the context of ANN these parameters are often called learning rates.

2 The Elastic Arms Algorithm

2.1 Review of the Algorithm

An event is defined as a set $\vec{x} = \{\vec{x}_1, ..., \vec{x}_N\}$ of N signal points. These $\vec{x}_i$'s can be either 2- or 3-dimensional. Each event also corresponds to a set of particle trajectories (tracks) and the task is to find these tracks, given $\vec{x}$. For this purpose a set $\vec{\pi} = \{\vec{\pi}_1, ..., \vec{\pi}_M\}$ of M template tracks, or elastic arms, are introduced, where an arm a is completely described by its P parameters $(\pi_a^{(1)}, ..., \pi_a^{(P)})$.

We state the track finding problem as finding minima to the energy function

$$E(\{S_{ia}\};\vec{\pi}) = \sum_{i=1}^{N}\sum_{a=1}^{M} S_{ia}M_{ia}(\vec{x},\vec{\pi}) + \lambda\sum_{i=1}^{N}\left(\sum_{a=1}^{M} S_{ia} - 1\right)^2 , \tag{1}$$

where S_{ia}[3] is a binary decision unit (neuron) defined as

$$S_{ia} = \begin{cases} 1 & \text{if point } i \text{ is assigned to arm } a \\ 0 & \text{otherwise} \end{cases} \tag{2}$$

$M_{ia}(\vec{x},\vec{\pi})$ (or just M_{ia}) is the minimal *squared* Eucledian distance between signal point i and arm a. Since each signal i can belong to only one track or no track at all, E must be minimized with the condition

$$\sum_a S_{ia} = 1 \text{ or } 0 \quad \forall\, i\ . \tag{3}$$

Finally, the parameter λ imposes a penalty if $\sum_a S_{ia} = 0$, that is, if signal i is not assigned to any arm. In this way the parameter λ governs the amount of noise the algorithm allows for.

In order to avoid local minima when minimizing E one often introduces noise[4] into the system. A commonly used procedure for doing this is *simulated annealing* [15] where the system is allowed to thermalize for a sequence of temperatures $T_n > T_{n-1} > ... > T_0$ according to the Boltzmann distribution

$$P(\{S_{ia}\};\vec{\pi}) = \frac{1}{Z}e^{-\beta E(\{S_{ia}\};\vec{\pi})} , \tag{4}$$

where $\beta = 1/T$ and Z is the usual partition function.

We proceed by calculating the marginal probability distribution

$$P_M(\vec{\pi}) = \sum_{\{S_{ia}\}} P(\{S_{ia}\};\vec{\pi}) , \tag{5}$$

[3] We use the notation S_{ia} rather than V_{ia} as in ref. [12] for binary variables.
[4] Not to be confused with noise signals

by summing out the neuronic degrees of freedom, S_{ia}. Doing this (for details see ref. [12]) we end up with

$$P_M(\vec{\pi}) = \frac{1}{Z} e^{-\beta E_{\mathit{eff}}(\vec{\pi})} , \tag{6}$$

where we have introduced the *effective* energy E_{eff} as

$$E_{\mathit{eff}}(\vec{\pi}) = -\frac{1}{\beta} \sum_i \log(e^{-\beta\lambda} + \sum_a e^{-\beta M_{ia}}) . \tag{7}$$

The most probable configurations according to eq. (6) corresponds to the minima of E_{eff}. Using a gradient descent method to minimize E_{eff} one gets an updating rule according to

$$\Delta\pi_a^{(k)} = -\eta_a^{(k)} \frac{\partial E_{\mathit{eff}}}{\partial \pi_a^{(k)}} = -\eta_a^{(k)} \sum_i \hat{V}_{ia} \frac{\partial M_{ia}}{\partial \pi_a^{(k)}} , \tag{8}$$

where $\eta_a^{(k)}$ is the updating parameter and the *Potts factor* $\hat{V}_{ia}$ is given by

$$\hat{V}_{ia} = \frac{e^{-\beta M_{ia}}}{e^{-\beta\lambda} + \sum_{b=1}^{M} e^{-\beta M_{ib}}} . \tag{9}$$

This algorithm has similarities with a collective self-organizing network [16]. The main difference is the neighborhood function. In our approach every signal is taken into account when fitting a certain arm, but they are weighted according to the Potts factor, eq. (9). In the low temperature limit the competitive winner-takes-all updating is retrieved. Another difference is the *zero*-neuron which enters the denominator of the Potts factor (9), as $e^{-\lambda\beta}$. The parameter λ governs the amount of noise points the algorithm allows for.

V_{ia} can be interpreted as the probability for signal i to belong to arm a. In the limit $T \to 0$ ($\beta \to \infty$) V_{ia} reduces to S_{ia}, which means that V_{ia} can also be viewed as the thermal average of S_{ia}; $V_{ia} = < S_{ia} >_T$.

2.2 Mean Field Theory Picture

Eq. (8) can also be understood in a simple and more intuitive way using the Potts mean field theory (MFT) equations [1]. Let us introduce N *zero*-neurons S_{i0} ($i = 1, ..., N$), such that $M_{i0} = \lambda \;\; \forall \; i$. A suitable energy function E', equivalent to eq. (1), is given by

$$E'(\{S_{ia}\}; \vec{\pi}) = \sum_{i=1}^{N} \sum_{a=0}^{M} S_{ia} M_{ia} , \tag{10}$$

which should be minimized together with the Potts condition

$$\sum_{a=0}^{M} S_{ia} = 1 \quad \forall\, i \,. \tag{11}$$

This condition is, because of the N *zero*-neurons, completely equivalent to the previous condition, eq. (3). Using gradient descent to minimize E', considering S_{ia} as a logical constant, gives

$$\Delta\pi_a^{(k)} = -\eta_a^{(k)} \sum_i S_{ia} \frac{\partial M_{ia}}{\partial \pi_a^{(k)}} \,. \tag{12}$$

Now, replace the S_{ia} with its thermal average $V_{ia} = < S_{ia} >_T$. According to the MFT equations [1] V_{ia} is calculated as

$$V_{ia} = \frac{e^{U_{ia}}}{\sum_{b=0}^{M} e^{U_{ia}}} \,, \tag{13}$$

where the local field U_{ia} is given by

$$\begin{aligned} U_{ia} &= -\frac{\partial E'}{\partial V_{ia}} \beta \\ &= -\beta M_{ia} \,. \end{aligned} \tag{14}$$

Substitute this into eq. (13),

$$\begin{aligned} V_{ia} &= \frac{e^{-\beta M_{ia}}}{\sum_{b=0}^{M} e^{-\beta M_{ib}}} \\ &= \frac{e^{-\beta M_{ia}}}{e^{-\beta\lambda} + \sum_{b=1}^{M} e^{-\beta M_{ib}}} \,, \end{aligned} \tag{15}$$

which is the same as eq. (9).

The N *zero*-neurons were introduced because of the presence of noise signals in the data, i.e. signals which should not be assigned to any real track. It is therefore natural to interpret V_{i0} as the probability for signal i to be a noise signal, and V_{i0} is given by

$$V_{i0} = \frac{e^{-\beta\lambda}}{e^{-\beta\lambda} + \sum_{b=1}^{M} e^{-\beta M_{ib}}} \,. \tag{16}$$

With this interpretation V_{i0} can used as a tool to identify non-fitted signals. If $V_{i0} \approx 1$ after the annealing of the algorithm ($T \to 0$), then i is a possible noise signal or a signal belonging to a track not included in the formalism. This observation will be used in section 5.2 to identify secondary tracks.

It is also interesting to see how the different V_{i0}'s develop with decreasing temperature (iteration step). If we choose λ to be the square of some typical distance representing the error in the initialization of the algorithm then, at high temperature, $V_{i0} > 0$ for most of the signals. As the temperature decreases "decisions" are made whether signal i belongs to a track or not. This can be seen in fig. (1) where the different V_{i0}'s are plotted against the number of iterations. The V_{i0}'s goes to either 0, meaning that signal i is assigned to a track, or 1, corresponding to i being a noise signal. Decisions are not made at a common temperature, instead the different V_{i0}'s converge at different temperatures (iteration steps). We also see that, already at a high temperature, $V_{i0} \approx 1$ for some signals, meaning that there is no initial arm close to these signals - they can form possible secondary tracks (see section 5.2).

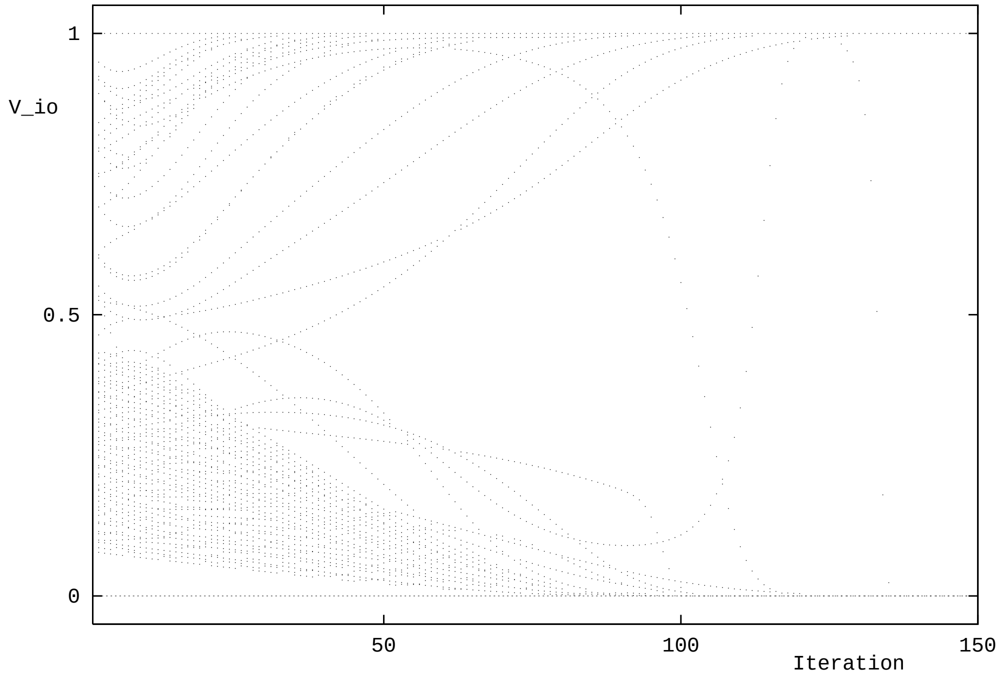

Figure 1: Development of the V_{i0}'s for a generated CERN DELPHI TPC event (371 signal points).

3 Geometry and Parameterization

The idea of the elastic arms algorithm is to match the observed events into a known parameterized model. The type of model, of course, depends on what kind of tracks we want to describe. In the case of straight tracks in three dimensions a track a is defined by $(\theta_a, \phi_a, x_a^o, y_a^o, z_a^o)$ through the obvious parameterization

$$\begin{aligned} x &= t\cos\theta_a \sin\phi_a + x_a^o \\ y &= t\sin\theta_a \sin\phi_a + y_a^o \\ z &= t\cos\phi_a + z_a^o \end{aligned} \tag{17}$$

where $t \geq 0$. In this paper, however, we only consider tracks coming from detectors with a constant magnetic field in the $\hat{z}$-direction, $\vec{B} = (0, 0, B)$. In three dimensions, a track is thus a spiral emerging from the vertex position (x^o, y^o, z^o) with an emission angle θ in the $\hat{x}\hat{y}$-plane, the curvature κ (also referring to the $\hat{x}\hat{y}$-plane) and finally the parameter γ governing the extension in the longitudinal direction. In terms of these parameters $(\kappa, \theta, \gamma, x^o, y^o, z^o)$ a point (x, y, z) on the track a is given by

$$\begin{aligned} x &= \frac{1}{\kappa_a}[\sin(\theta_a + \kappa_a t) - \sin\theta_a] + x_a^o \\ y &= \frac{1}{\kappa_a}[-\cos(\theta_a + \kappa_a t) + \cos\theta_a] + y_a^o \\ z &= \gamma_a t + z_a^o \ , \end{aligned} \tag{18}$$

where $t \in [0, \pi/|\kappa_a|]$. Note that t has the dimension of length, making γ a dimensionless scale parameter. The distance measure M_{ia} is given by [5]

$$\begin{aligned} M_{ia}(\vec{x}_i, \vec{\pi}_a) &= M_{ia}^{(xy)} + M_{ia}^{(z)} \\ &= \frac{1}{\kappa_a^2}\left(1 - \sqrt{[\kappa_a(x_i - x_a^o) + \sin\theta_a]^2 + [\kappa_a(y_i - y_a^o) - \cos\theta_a]^2}\right)^2 \\ &+ (z_i - z_a^o - \gamma_a t)^2 \end{aligned} \tag{19}$$

where t is computed from

$$t = \frac{1}{\kappa_a}\arctan\left(\frac{(x_i - x_a^o)\cos\theta_a + (y_i - y_a^o)\sin\theta_a}{\frac{1}{\kappa_a} + (x_i - x_a^o)\sin\theta_a - (y_i - y_a^o)\cos\theta_a}\right) . \tag{20}$$

4 A Refined Gradient Descent Method

4.1 Metrical Considerations

The efficiency of the algorithm depends upon how fast E_{eff} converges towards its global minimum. A straight forward way to minimize E_{eff} is to use the gradient descent method; minimize by taking small steps in the negative gradient direction (see eq. (8)). The gradient descent update equation can be written in a general way as

$$\Delta\pi^{(k)} = -\epsilon \sum_{j=1}^{P} g_{kj}^{-1} \frac{\partial E_{\mathit{eff}}}{\partial \pi^{(j)}} \ , \tag{21}$$

where ϵ is a small number. In the standard gradient descent method, $\boldsymbol{g}$ is simply the identity matrix. Since the parameters $\vec{\pi} = (\kappa, \theta, \gamma, x^o, y^o, z^o)$ are different with

[5] M_{ia} is slightly different from that of ref. [12]

respect to dimensions and bounds, a natural extension is to have different learning rates for different parameters. This was done in ref. [12] where we used a diagonal matrix $\boldsymbol{g}^{-1}$ according to

$$\boldsymbol{g}^{-1} = \frac{1}{\epsilon}\begin{pmatrix} \eta^{(\kappa)} & 0 & 0 \\ 0 & \eta^{(\theta)} & 0 \\ 0 & 0 & \eta^{(\gamma)} \end{pmatrix} , \tag{22}$$

where the update parameters (learning rates) $(\eta^{(\kappa)}, \eta^{(\theta)}, \eta^{(\gamma)})$ were chosen on an intuitive and exploratory ground. We will now extend the choice of $\boldsymbol{g}$ to include "off diagonal" elements. Each $\Delta\pi^{(k)}$ will then be a mixture of all partial derivatives $\partial E_{\mathit{eff}}/\partial\pi^{(j)}$ $(j = 1, ..., P)$. One way of doing this is to use the second derivative $\mathbf{H}$ of E_{eff} and simply let $\boldsymbol{g} = \mathbf{H}$. To show this let us expand E_{eff} around a point $\vec{\pi}_p$

$$E_{\mathit{eff}}(\vec{\pi}) \approx E_{\mathit{eff}}(\vec{\pi}_p) + \nabla E_{\mathit{eff}}|_{\vec{\pi}_p} \cdot (\vec{\pi} - \vec{\pi}_p) + \frac{1}{2}(\vec{\pi} - \vec{\pi}_p) \cdot \mathbf{H} \cdot (\vec{\pi} - \vec{\pi}_p) , \tag{23}$$

with the Hessian $\mathbf{H}$ given by

$$[\mathbf{H}]_{ij} = \left.\frac{\partial^2 E_{\mathit{eff}}}{\partial\pi^{(i)}\partial\pi^{(j)}}\right|_{\vec{\pi}_p} . \tag{24}$$

Let $\vec{\pi}_m$ denote the point for which E_{eff} has a minimum, then

$$\nabla E_{\mathit{eff}}|_{\vec{\pi}_m} = 0 . \tag{25}$$

This condition, put into eq. (23), gives the desired updating equation

$$\Delta\vec{\pi} = \vec{\pi}_m - \vec{\pi}_p = -\, \mathbf{H}^{-1} \cdot \nabla E_{\mathit{eff}}\Big|_{\vec{\pi}_p} . \tag{26}$$

The calculation of $\mathbf{H}$ is in our case, unfortunately, very computationally demanding and would therefore not result in a faster algorithm. We will instead make another approach and use matrix elements g_{ij} defined by the equation

$$ds^2 = \sum_{ij} g_{ij} d\pi^{(i)} d\pi^{(j)} , \tag{27}$$

where ds and $d\pi$ is the distance between two neighboring points in the usual space and the space defined by the coordinates $\vec{\pi}$ respectively (the right-hand member of the above equation is called the *metric* of the Riemannian space). If $(\pi^{(1)}, ..., \pi^{(P)})$ are the usual Cartesian coordinates (x, y, z), then $ds^2 = dx^2 + dy^2 + dz^2$ and $\boldsymbol{g}$ reduces to the identity matrix. In our formalism, however, we use the coordinates $(\kappa, \theta, \gamma, x^o, y^o, z^o)$ rather than (x, y, z) to describe the elastic arms. The point-wise transformation between the two coordinate systems is given by the parameterization, eq. (18), from which $d\vec{s}$ can be calculated as

$$d\vec{s} = \sum_k \frac{\partial\vec{s}}{\partial\pi^{(k)}} d\pi^{(k)} = \frac{\partial\vec{s}}{\partial\kappa} d\kappa + \ldots + \frac{\partial\vec{s}}{\partial z^o} dz^o . \tag{28}$$

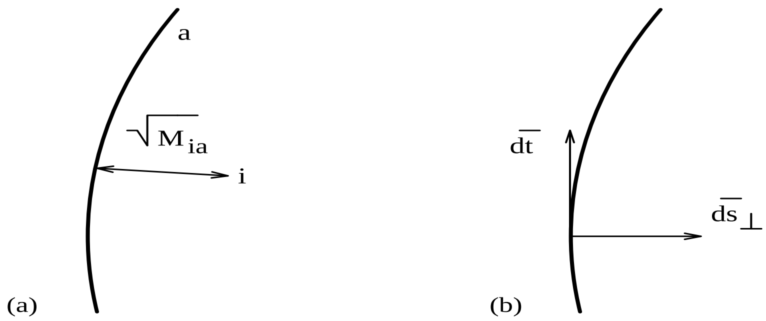

Figure 2: **(a).** The measure M_{ia}, which is the closest distance between arm a and point i. **(b).** The small displacement $d\vec{t}$ along the arm and the corresponding displacement $d\vec{s}_\perp$ orthogonal to the arm.

Since M_{ia} is a measure of the minimal distance between an arm a and a point i the relevant component of $d\vec{s}$ is the one orthogonal to the elastic arm, denoted $d\vec{s}_\perp$ (see fig. (2)). Let dt be a small displacement along the arm, then $d\vec{t} = (\partial\vec{s}/\partial t)dt$ and from this $d\vec{s}_\perp$ is given by

$$\begin{aligned} d\vec{s}_\perp &= d\vec{s} - \left[\frac{(d\vec{s}\cdot d\vec{t})}{dt^2}\right] d\vec{t} \\ &= \sum_k \left(\vec{v}_k - \left[\frac{\vec{v}_k \cdot d\vec{t}}{dt^2}\right] d\vec{t}\right) d\pi^{(k)} , \end{aligned} \tag{29}$$

where

$$\vec{v}_k \equiv \frac{\partial \vec{s}}{\partial \pi^{(k)}} \tag{30}$$

Finally, $d\vec{s}_\perp$ can be expressed in the desired form

$$ds_\perp^2 = \sum_{kj} g_{kj}\, d\pi^{(k)} d\pi^{(j)} , \tag{31}$$

with the matrix elements

$$g_{kj} = (\vec{v}_k \cdot \vec{v}_j) - \frac{(\vec{v}_k \cdot d\vec{t})(\vec{v}_j \cdot d\vec{t})}{dt^2} \tag{32}$$

that are tabulated in appendix A. As given by eq. (32) g_{kj} still depends on the parameter t and varies as we move along the trajectory. We take this into account by replacing g_{kj} with its average

$$g_{kj} \to \frac{1}{(t_2 - t_1)} \int_{t_1}^{t_2} g_{kj}\, dt , \tag{33}$$

where t_1 and t_2 defines the beginning and the end of the elastic arm, respectively.

The above procedure provides a robust and straightforward way of implementing gradient descent. The number of undetermined updating parameters has also been reduced to only one (ϵ) which is the the global time-scale in the updating eq. (26).

This refined gradient descent method has been compared with the gradient descent method used in ref. [12] for a number of events from the CERN DELPHI TPC event generator. With a fixed iteration length (20) this new gradient descent method gave consistently lower final energies than the previous method.

5 Extensions of the Algorithm

Until now all tracks emerging from other points than the assumed vertex position have been considered noise. This restriction is sometimes unrealistic from an experimental point of view, where we have no knowledge about the number of vertices and their positions. In this section we extend the formalism of the elastic arms algorithm to include, (i) secondary tracks coming from decaying particles and (ii) multiple vertex positions. Using the ideas in ref. [14] we also show how to solve the problem of mirror charges in the data.

5.1 Multiple Vertex Positions

It is possible with events where there are more than one (primary) vertex position [6]. Suppose there are K vertex positions $\vec{r}_j^{\,o} = (x_j^o, y_j^o, z_j^o)$ $(j = 1, ..., K)$, then arm a is parameterized by $(\kappa_a, \theta_a, \gamma_a, \vec{r}^{\,o}_{k(a)})$, where $k(a) = j$ if arm a belongs to vertex position j. Each $\vec{r}_j^{\,o}$ is updated with contributions from all arms belonging to it, according to

$$\Delta\vec{r}_j^{\,o} = \frac{\sum_{a=1}^{M} \Delta\vec{r}_a^{\,o}\delta_{k(a),j}}{\sum_{a=1}^{M} \delta_{k(a),j}} \,, \tag{34}$$

where $\delta_{i,j}$ is the Kronecker δ-symbol and $\Delta\vec{r}_a^{\,o} = (\Delta x_a^o, \Delta y_a^o, \Delta z_a^o)$ is calculated using the update eq. (21) derived in section 4. (More explicit formulas can be found in appendix B.)

[6] Multiple events are expected for the next generation of accelerators (LCH,SSC).

5.2 Secondary Vertex Positions

We now focusing on the problem of decaying particles. Due to physics involved, a secondary track can then only occur in one of the following three situations (see fig. (3)):

- A neutral particle decays into a positive and a negative particle (fig. 3a).
- A positive charged particle decays into a neutral and a positive particle (fig. 3b).
- A negative charged particle decays into a neutral and a negative particle (fig. 3c).

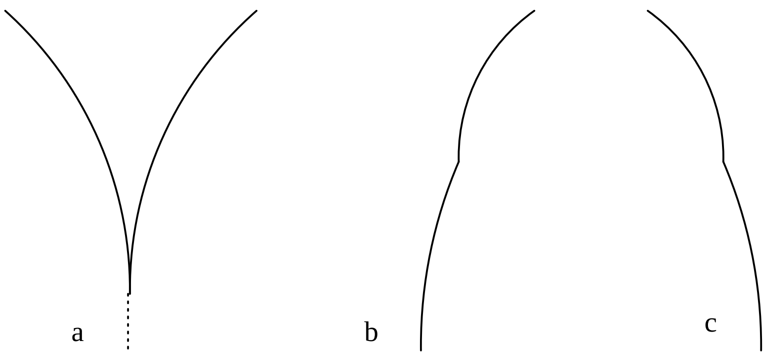

Figure 3: **(a)**. Neutral $\rightarrow$ Positive + Negative. **(b)**. Positive $\rightarrow$ Neutral + Positive. **(c)**. Negative $\rightarrow$ Neutral + Negative.

The last two types are the easiest to handle, because the secondary tracks have vertex positions that lie on an already visible track. This fact reduces the number of parameters needed to describe the secondary track. If track a decays, somewhere inside the detector, into the secondary track $\tilde{a}$ then $(\tilde{\kappa}_a, \tilde{\theta}_a, \tilde{\gamma}_a, \tilde{t}_a)$ completely describes $\tilde{a}$ through

$$\begin{aligned} x &= \frac{1}{\tilde{\kappa}_a}[\sin(\tilde{\theta}_a + \tilde{\kappa}_a t) - \sin\tilde{\theta}_a] + \tilde{x}_a^o \\ y &= \frac{1}{\tilde{\kappa}_a}[-\cos(\tilde{\theta}_a + \tilde{\kappa}_a t) + \cos\tilde{\theta}_a] + \tilde{y}_a^o \\ z &= \tilde{\gamma}_a t + \tilde{z}_a^o\ , \end{aligned} \tag{35}$$

with $(\tilde{x}_a^o, \tilde{y}_a^o, \tilde{z}_a^o)$ given by

$$\tilde{x}_a^o = \frac{1}{\kappa_a}[\sin(\theta_a + \kappa_a \tilde{t}_a) - \sin\theta_a] + x_a^o$$

$$\begin{aligned}\tilde{y}_a^o &= \frac{1}{\kappa_a}[-\cos(\theta_a + \kappa_a \tilde{t}_a) + \cos\theta_a] + y_a^o \\ \tilde{z}_a^o &= \gamma_a \tilde{t}_a + z_a^o \ .\end{aligned} \tag{36}$$

We locate all possible pairs $(a, \tilde{a})$ by looking at the quantity σ_a defined as

$$\sigma_a = \frac{\sum_i V_{ia} M_{ia}}{\sum_i V_{ia}} \ . \tag{37}$$

A small σ_a means that arm a has been well fitted to a track, while a large σ_a indicates a possible secondary track. The parameters $(\tilde{\kappa}_a, \tilde{\theta}_a, \tilde{\gamma}_a, \tilde{t}_a)$ can be updated using the method derived in section 4. The corresponding derivatives $\partial \tilde{M}_{ia}/\partial \tilde{\pi}_a^{(k)}$ can be found in appendix B.

Tracks corresponding to the first type of decay must be parameterized by $(\kappa, \theta, \gamma, \vec{r}^{\,o})$, since they are coming from an invisible track. The vertex position $\vec{r}^{\,o} = (x^o, y^o, z^o)$ for each track pair is updated using eq. (34). As mentioned in section 2.2 V_{i0} can now be used to locate possible secondary tracks. If $V_{i0} \approx 1$ for a set of signals close to each other, then they may form possible secondary tracks.

5.3 Mirror Charges [14]

In some detectors there are ambiguities in the measurement of the position of a particle. This means that a coordinate for a given signal may be double-valued - it has a mirror signal, $(x, y) \rightarrow (x^+ \text{ or } x^-, y)$. This ambiguity problem can be solved [14] in the same manner as noise signals are handled, but on different levels of resolution. This is seen by rewriting the original energy function, eq. (1) as

$$E(\{S_{ia}, s_{ia}^+, s_{ia}^-\}, \vec{\pi}) = \sum_{i=1}^{N}\sum_{a=1}^{M} S_{ia}\left(s_{ia}^+ M_{ia}^+ + s_{ia}^- M_{ia}^-\right) + \lambda \sum_{i=1}^{N}\left(\sum_{a=1}^{M} S_{ia} - 1\right)^2 \ , \tag{38}$$

where the new variables s_{ia}^+ and s_{ia}^- are defined by

$$s_{ia}^+ \ (s_{ia}^-) = \begin{cases} 1 & \text{if signal } i^+ \ (i^-) \text{ is assigned to arm } a \\ 0 & \text{otherwise} \end{cases} \ , \tag{39}$$

and with the corresponding distance measure M_{ia}^+ (M_{ia}^-). A derivation analogous to that of section 4.1 then gives new updating equations with the associated *Potts factors* $\hat{v}_{ia}^+$ and $\hat{v}_{ia}^-$. For further details we refer to [14].

6 Initialization of the Algorithm

To ensure a fast convergence towards a high quality solution the elastic arms algorithm must be initialized with approximate values for the parameters used to parameterize the track. Since there are a lot of different types of tracks (straight or curved tracks in 2- or 3-dimensions) in different experimental environments, a universal procedure for initiating the algorithm is difficult to find. One initialization method may work for one experimental setup, but may be useless in another. However, once an initialization procedure has been found, which may require some engineering work, the elastic arms algorithm gives high quality solutions.

In this paper a general track is described by the parameters $(\kappa, \theta, \gamma, x^o, y^o, z^o)$. To initialize the algorithm one generally needs 6 initial values. In ref. [12] this was simplified by the knowledge of (x^o, y^o, z^o) for the primary tracks. The initial values of (κ, θ, γ) could be found using Hough Transforms [17] (which essentially are variants of "histogramming" or "binning" techniques in parameter space). When there is no knowledge about the vertex position such a simple binning technique is harder to find and will probably require specific knowledge about the experimental setup. However, if the primary vertex position is known to be close to some origin, the simple Hough transform used in ref. [12] can still be used.

7 Summary

We have extended the formalism of the elastic arms algorithm to include secondary tracks coming from decaying particles and to handle events with more than one primary vertex position. With the explicit parameterization of the tracks it is possible to refine the gradient descent method with a calculation of the natural metric. This substantially reduces the number of parameters in the minimization procedure, giving an almost "black box" like algorithm. This method hence requires almost no parameter-tuning by the user.

The introduction of *zero*-neurons with the corresponding probability V_{i0} has given us a better understanding of how the algorithm really works.

The elastic arms approach is easy to adapt to specific situations. If, for example, measurement precisions vary with different pad-layers, then the formalism can be generalized to allow for different i-dependent λ's. Even though we have used a constant λ it is possible to have a temperature dependent λ.

It is showed that although the core algorithm is extremely robust and generalizable to new situations, the initialization procedure has to be custom made for different experimental configurations.

In this paper we have focused on specific generalizations of the algorithm and methods for fast convergence. Due to lack of real data, these extensions has been tested on the same data set as in ref. [12]. To test the new features of the algorithm in an exploratory way other data sets should be used.

Particle physics tracking codes always end up "dirty" with ample of exceptions etc. The elastic arms approach has the advantage that the code based on it starts out from a clean base with global constraints built in. Nevertheless it is flexible to host a variety of experimental setups. The only "engineering" needed concerns the initialization. Furthermore the approach has the advantage of being intrinsically parallel, facilitating design of custom made hardware for real-time execution.

Acknowledgements

I would like to thank Richard Blankenbecler and Carsten Peterson for valuable discussions and Bo Söderberg in particular for the idea on how to use the natural metric. I would also like to thank O. Bärring for providing me with the DELPHI TPC simulation data.

Appendix A

Explicit expressions for the matrix elements g_{kj} corresponding to the parameterization, eq. (18):

$$
\begin{aligned}
g_{\kappa\kappa} &= \left(3+2\gamma^2\left(2+\kappa^2t^2\right)-4\cos(\kappa t)(1+\gamma^2)+\cos\left[2\kappa t\right]-4\kappa\gamma^2 t\sin(\kappa t)\right)\frac{1}{2\kappa^4(1+\gamma^2)} &\text{(A1)}\\
g_{\kappa\theta} &= 2\sin(\frac{\kappa t}{2})^2\left(\kappa\gamma^2 t+\sin(\kappa t)\right)\frac{1}{\kappa^3(1+\gamma^2)} &\text{(A2)}\\
g_{\kappa\gamma} &= (\sin(\kappa t)-\kappa t)\frac{\gamma t}{\kappa^2(1+\gamma^2)} &\text{(A3)}\\
g_{\kappa x^o} &= \cos(\theta+\kappa t)\frac{t}{\kappa}+(\sin\theta-\sin(\theta+\kappa t))\frac{1}{\kappa^2}+(\sin(\kappa t)-\kappa t)\frac{\cos(\theta+\kappa t)}{\kappa^2(1+\gamma^2)} &\text{(A4)}\\
g_{\kappa y^o} &= \sin(\theta+\kappa t)\frac{t}{\kappa}+(\cos(\theta+\kappa t)-\cos\theta)\frac{1}{\kappa^2}+(\sin(\kappa t)-\kappa t)\frac{\sin(\theta+\kappa t)}{\kappa^2(1+\gamma^2)} &\text{(A5)}\\
g_{\kappa z^o} &= (\sin(\kappa t)-\kappa t)\frac{\gamma}{\kappa^2(1+\gamma^2)} &\text{(A6)}\\
g_{\theta\theta} &= 2\sin(\frac{\kappa t}{2})^2\left(1+2\gamma^2+\cos(\kappa t)\right)\frac{1}{\kappa^2(1+\gamma^2)} &\text{(A7)}\\
g_{\theta\gamma} &= -2\sin(\frac{\kappa t}{2})^2\frac{\gamma t}{\kappa(1+\gamma^2)} &\text{(A8)}\\
g_{\theta x^o} &= (\cos(\theta+\kappa t)-\cos\theta)\frac{1}{\kappa}-2\sin(\frac{\kappa t}{2})^2\frac{\cos(\theta+\kappa t)}{\kappa(1+\gamma^2)} &\text{(A9)}\\
g_{\theta y^o} &= (\sin(\theta+\kappa t)-\sin\theta)\frac{1}{\kappa}-2\sin(\frac{\kappa t}{2})^2\frac{\sin(\theta+\kappa t)}{\kappa(1+\gamma^2)} &\text{(A10)}\\
g_{\theta z^o} &= -2\sin(\frac{\kappa t}{2})^2\frac{\gamma}{\kappa(1+\gamma^2)} &\text{(A11)}\\
g_{\gamma\gamma} &= \frac{t^2}{(1+\gamma^2)} &\text{(A12)}\\
g_{\gamma x^o} &= -\cos(\theta+\kappa t)\frac{\gamma t}{(1+\gamma^2)} &\text{(A13)}\\
g_{\gamma y^o} &= -\sin(\theta+\kappa t)\frac{\gamma t}{(1+\gamma^2)} &\text{(A14)}\\
g_{\gamma z^o} &= \frac{t}{(1+\gamma^2)} &\text{(A15)}\\
g_{x^o x^o} &= 1-\frac{\cos(\theta+\kappa t)^2}{(1+\gamma^2)} &\text{(A16)}\\
g_{x^o y^o} &= -\frac{\sin(2\left(\theta+\kappa t\right))}{2(1+\gamma^2)} &\text{(A17)}\\
g_{x^o z^o} &= -\cos(\theta+\kappa t)\frac{\gamma}{(1+\gamma^2)} &\text{(A18)}\\
g_{y^o y^o} &= 1-\frac{\sin(\theta+\kappa t)^2}{(1+\gamma^2)} &\text{(A19)}\\
g_{y^o z^o} &= -\sin(\theta+\kappa t)\frac{\gamma}{(1+\gamma^2)} &\text{(A20)}
\end{aligned}
$$

$$g_{z^o z^o} = \frac{1}{(1+\gamma^2)} \tag{A21}$$

Appendix B

The formulae below are valid only for the parameterization given by eq. (18). An elastic arm a is given by $(\kappa, \theta, \gamma, x^o, y^o, z^o)$ (the index a is suppressed below) and a signal i by (x_i, y_i, z_i).

Some abbreviations to shorten the formulas:

$$\Delta x \equiv x_i - x^o \tag{B1}$$

$$\Delta y \equiv y_i - y^o \tag{B2}$$

$$\Delta z \equiv z_i - z^o \,. \tag{B3}$$

With these we define

$$c \equiv \Delta x \cos\theta + \Delta y \sin\theta \tag{B4}$$

$$d \equiv \frac{1}{\kappa} + \Delta x \sin\theta - \Delta y \cos\theta \tag{B5}$$

$$s \equiv \sqrt{(\kappa\Delta x + \sin\theta)^2 + (\kappa\Delta y - \cos\theta)^2} \,. \tag{B6}$$

The distance measure M_{ia} between signal i and track a is then given by

$$M_{ia} = \frac{1}{\kappa^2}(1-s)^2 + (\Delta z - \gamma t)^2 \,, \tag{B7}$$

where t is computed from

$$\kappa t = \arctan\left(\frac{\Delta x \cos\theta + \Delta y \sin\theta}{\frac{1}{\kappa} + \Delta x \sin\theta - \Delta y \cos\theta}\right) \,. \tag{B8}$$

The derivatives of M_{ia} with respect to the parameters $(\kappa, \theta, \gamma, x^o, y^o, z^o)$ are:

$$\frac{\partial M_{ia}}{\partial \kappa} = -\frac{2}{\kappa^3}(1-s)\left(1 - \frac{d\kappa}{s}\right) - \frac{2\gamma}{\kappa}(\Delta z - \gamma t)\left(\frac{c}{\kappa^2(c^2+d^2)} - t\right) \tag{B9}$$

$$\frac{\partial M_{ia}}{\partial \theta} = \frac{2(s-1)}{\kappa s}c - \frac{2\gamma}{\kappa}(\Delta z - \gamma t)\left(\frac{d}{\kappa^2(c^2+d^2)} - 1\right) \tag{B10}$$

$$\frac{\partial M_{ia}}{\partial \gamma} = -2t(\Delta z - \gamma t) \tag{B11}$$

$$\frac{\partial M_{ia}}{\partial x^o} = \frac{2(1-s)}{\kappa s}(\kappa\Delta x + \sin\theta) - \frac{2\gamma}{\kappa^2}\frac{(\Delta z - \gamma t)}{(c^2+d^2)}(\kappa\Delta y - \cos\theta) \tag{B12}$$

$$\frac{\partial M_{ia}}{\partial y^o} = \frac{2(1-s)}{\kappa s}(\kappa\Delta y - \cos\theta) + \frac{2\gamma}{\kappa^2}\frac{(\Delta z - \gamma t)}{(c^2+d^2)}(\kappa\Delta x + \sin\theta) \tag{B13}$$

$$\frac{\partial M_{ia}}{\partial z^o} = -2(\Delta z - \gamma t) \,. \tag{B14}$$

A secondary track $\tilde{a}$ is described by $(\tilde{\kappa}, \tilde{\theta}, \tilde{\gamma}, \tilde{t})$ through the parameterization of eqs. (35,36). The derivatives $\partial M_{ia}/\partial\tilde{\kappa}, \partial M_{ia}/\partial\tilde{\theta}, \partial M_{ia}/\partial\tilde{\gamma}$ are given above (eqs. (B9 - B11)) with the substitution

$$(\kappa, \theta, \gamma, x^o, y^o, z^o) \rightarrow (\tilde{\kappa}, \tilde{\theta}, \tilde{\gamma}, \tilde{x}^o, \tilde{y}^o, \tilde{z}^o) \ , \tag{B15}$$

and $\partial M_{ia}/\partial\tilde{t}$ is given by

$$\frac{\partial M_{ia}}{\partial\tilde{t}} = \frac{2}{\tilde{\kappa}}\left(\frac{1}{s} - 1\right)\left(\tilde{\kappa}\left(\Delta x \cos\left(\theta + \kappa\tilde{t}\right) + \Delta y \sin\left(\theta + \kappa\tilde{t}\right)\right) + sn\right) + \tag{B16}$$

$$+ \quad 2\left(\Delta z + \tilde{\gamma}t\right)\left(\gamma + \frac{\tilde{\gamma}}{\tilde{\kappa}\left(c^2 + d^2\right)}\left(\Delta y \cos\left(\theta + \kappa\tilde{t}\right) - \Delta x \sin\left(\theta + \kappa\tilde{t}\right) - \frac{cs}{\tilde{\kappa}}\right)\right) \tag{B17}$$

where $sn \equiv \sin(\tilde{\theta} - \theta - \kappa\tilde{t})$ and $cs \equiv \cos(\tilde{\theta} - \theta - \kappa\tilde{t})$.

References

[1] C. Peterson and B. Söderberg, "A New Method for Mapping Optimization Problems onto Neural Networks", *International Journal of Neural Systems* **1**, 3 (1989).

[2] C. Peterson, "Parallel Distributed Approaches to Combinatorial Optimization Problems - Benchmark Studies on TSP", *Neural Computation* **2**, 261 (1990).

[3] B. Denby, "Neural Networks and Cellular Automata in Experimental High Energy Physics", *Computer Physics Communications* **49**, 429 (1988).

[4] C. Peterson, "Track Finding with Neural Networks", *Nuclear Instruments and Methods* **A279**, 537 (1989).

[5] G. Stimpfl-Abele and L. Garrido, "Fast Track Finding with Neural Nets", *Computer Physics Communication* **64**, 46 (1991).

[6] L. Gislén, C. Peterson and B. Söderberg, "Rotor Neurons - Basic Formalism and Dynamics", *Neural Computation* **4**, 737 (1992)

[7] C. Peterson, "Neural Networks and High Energy Physics", *Proc. of International Workshop on Software Engineering, Artificial Intelligence and Expert Systems for High Energy and Nuclear Physics, Lyon Villeurbanne, France, March , 1990*, eds. D. Perret-Gallix and W. Woijcik, Editions du CRNS (Paris 1990).

[8] A.A. Glazov, I.V. Kisel, E.V. Konotopskaya, V.N. Neskoromnyi and G.A. Ososkov, "Track Reconstruction in Discrete Detectors by Neural Networks", JINR Communication E10-92-352, Dubna 1992.

[9] A.L. Yuille, "Generalized Deformable Models, Statistical Physics, and Matching Problems", *Neural Computation* **2**, 1 (1990).

[10] R. Durbin, and D. Willshaw, "An Analog Approach to the Traveling Salesman Problem Using an Elastic Net Method", *Nature* **326**, 689 (1987).

[11] A. Yuille, K. Honda and C. Peterson, "Particle Tracking by Deformable Templates", *Proceedings of 1991 IEEE INNS International Joint Conference on Neural Networks*, Vol. 1, pp 7-12, Seattle, WA (July 1991).

[12] M. Ohlsson, C. Peterson, A. Yuille, "Track Finding with Deformable Templates - The Elastic Arms Approach", *Computer Physics Communications* **71**, 77 (1992).

[13] M. Gyulassy and H. Harlander, "Elastic Tracking and Neural Network Algorithms for Complex Pattern Recognition", *Computer Physics Communications* **66**, 31 (1991).

[14] R. Blankenbecler, manuscript in preparation.

[15] S. Kirkpatrick, C.D. Gelatt and M.P. Vecchi, "Optimization by Simulated Annealing", *Science* **220**, 671 (1983).

[16] J. Hertz, A. Krogh and R. G. Palmer, *Introduction to the Theory of Neural Computation*, Addison-Wesley, Redwood City, Ca. (1991).

[17] R.O. Duda and P.E. Hart, *Pattern Classification and Scene Analysis*, John Wiley and Sons, New York (1973).