0 0 0 4 3 5 1 3 4 2

RECEIVED
LAWRENCE
RADIATION LABORATORY

MAR 24 1975

LIBRARY AND
DOCUMENTS SECTION

LBL-3653 c. 1
UC-32
TID-4500-R62

JUBB: RECOGNIZING PARTICLE TRACKS IN CYLINDRICAL SPARK CHAMBERS

Ed Fourt

January 1975

Prepared for the U. S. Atomic Energy Commission
under Contract W-7405-ENG-48

For Reference

Not to be taken from this room

LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY

LBL-3653 c. 1

DISCLAIMER

This document was prepared as an account of work sponsored by the United States Government. While this document is believed to contain correct information, neither the United States Government nor any agency thereof, nor the Regents of the University of California, nor any of their employees, makes any warranty, express or implied, or assumes any legal responsibility for the accuracy, completeness, or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately owned rights. Reference herein to any specific commercial product, process, or service by its trade name, trademark, manufacturer, or otherwise, does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or any agency thereof, or the Regents of the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or any agency thereof or the Regents of the University of California.

0 0 0 0 4 3 0 1 3 4 3

LBL-3653

LAWRENCE BERKELEY LABORATORY
University of California
Berkeley, California

JUBB: RECOGNIZING PARTICLE TRACKS IN CYLINDRICAL SPARK CHAMBERS*

by

Ed Fourt
Mathematics and Computing

January, 1975

*Work performed under the auspices of the U. S. Energy Research and Development Administration

LBL-3653

JUBB: RECOGNIZING PARTICLE TRACKS IN CYLINDRICAL SPARK CHAMBERS

JUBB is the collective name for a group of Fortran routines that recognize tracks traveling through cylindrical spark chambers. It has been written specifically for the SPEAR Magnetic Detector Experiment SP1, but with a considerable degree of generality, so that it may be readily used in other, similar situations. It uses only space coordinates, and does not take into account information coming out of other parts of the detector -- the shower counters, muon chambers, etc. It does geometrical reconstruction exclusively.

We will describe JUBB in a very idealized fashion, abstracting the algorithms and the intentions from the grubby matrix of the actual SP1 data analysis program and the physical detector. We first outline the general problem of JUBB must deal with; we then describe the data structures that JUBB sets up and manipulates, which is to say that we describe the information JUBB receives and produces and the method by which it is passed on for further analysis; and finally we describe the algorithms JUBB uses for recognizing tracks.

1. The General Problem

SPEAR, the Stanford Positron-Electron Accelerating Ring, is a circular colliding-beam particle accelerator, located at the Stanford Linear Accelerator Center (SLAC). A beam of electrons circulates one way around the ring, and a beam of positrons simulatneously circulates the other way. At specific interaction regions the beams are made to intersect; thus the particles in the beams are made to collide. The particles in the two beams have equal but opposite momenta; furthermore, the two particles in a collision are antiparticles of one another. Therefore, for purposes of this description, we may consider that they utterly annihiliate one another resulting in a motionless, featureless blob of pure energy.

This energy must manifest itself as particles; these particles must fly away from the locus of the blob. From the detection and analysis of these resultant particles, physicists attempt to understand what actually went on during the collision.

To detect these particles, a series of concentric cylindrical spark chambers have been built surrounding one interaction region, with the axis of the cylinders being the line of the beam. (We will ignore all other parts of the detector besides the these spark chambers, since these are all that JUBB is concerned with.) A coordinate system is defined, with its origin being at the center of the interaction region (which equals the center of the chambers, the midpoint of their axis). The 'z' coordinate is taken as running along this chamber axis. See fig. 1.

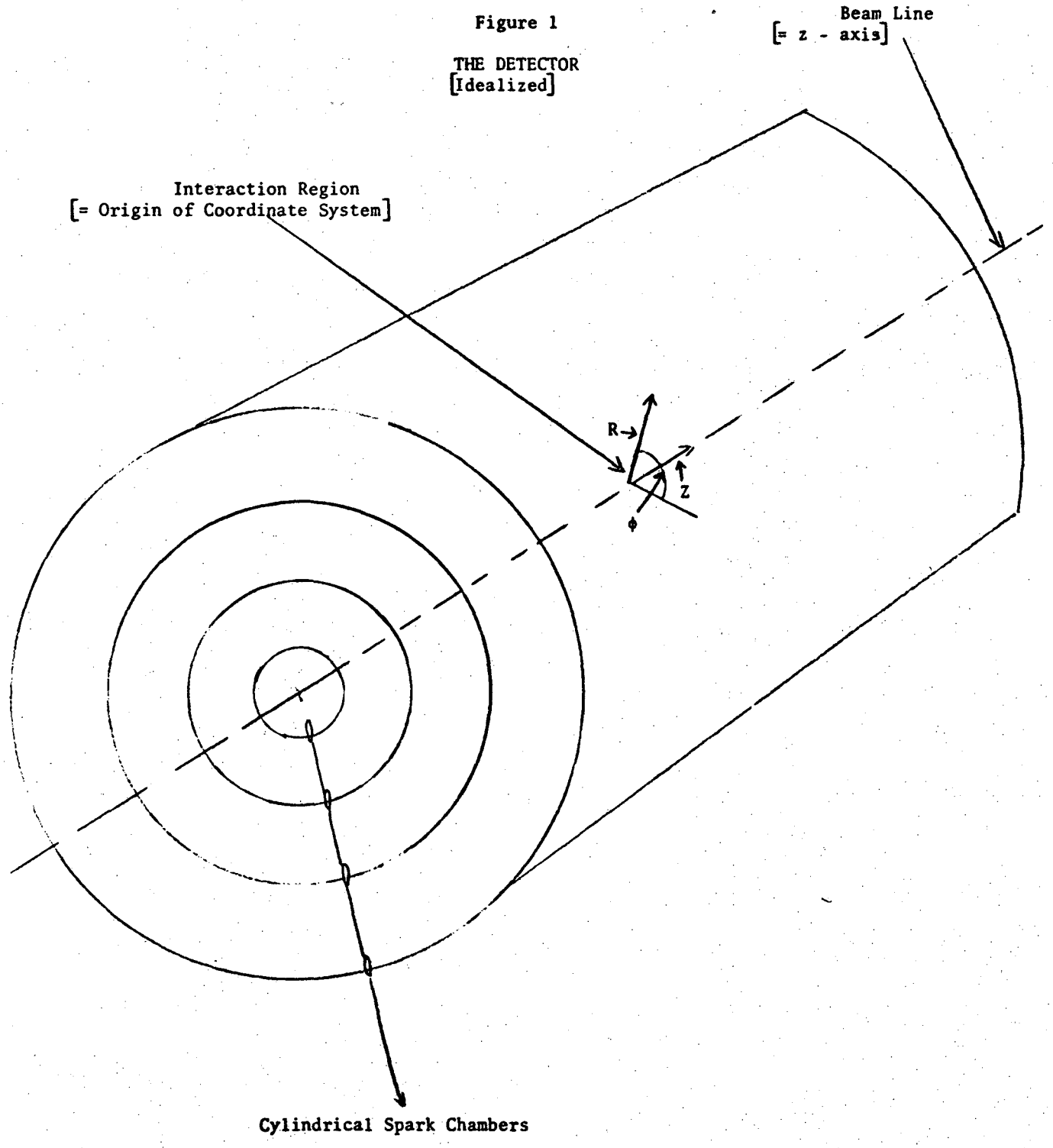
There is a constant magnetic field applied along this z axis. Charged particles coming out of the interaction region therefore move circularly transversely to the beam, while proceeding linearly along the direction of the beam line. Thus their orbit is a helix, and the axis of each helix is parallel to the z-axis of our coordinate system.

As a charged particle passes thru a spark chamber, it causes that chamber to fire; via a logically-straight-forward process, the location in space of this spark is determined, and thus a point along (or near) the orbit of some particle is found. The coordinates of all the space points for a particular event are placed into a bank of words in the computer. It is JUBB's job to take such a bank and thread valid particle orbits thru the points. Once such orbits are defined, one may calculate the momenta of the particles that made these points, and thus understand what went on during the collision.

Thus, we see that our problem reduces that of recognizing helices from their intersections with concentric cylinders, given that the axis of each helix must be parallel to the axis of the cylinders.

Figure 1

THE DETECTOR
[Idealized]



A general constraint on the program is the necessity of using it on-line in monitoring the accelerator, as well as off-line in analyzing the data. To see that the accelerator and detector are running properly, one must check that they are turning out bataches of points that do indeed seem to define a valid event. Thus, a version of JUBB exists which is extremely fast and compact, for use in the small computer that controls the accelerator. For ease of maintaining the software, this version differs as little as possible from the expanded, rigorous version used in off-line event analysis. Ideally, the on-line version would be exactly a subset of the off-line one.

JUBB begins with a bank of coordinates for all the space points for an event. Each coordinate contains several words of information; as we use the work 'coordinate' below, we mean this set of several (>3) computer words. The full contents of each coordinate are outlined in comments appearing in Fortran routine FLAMEON. The following words are of interest to JUBB:

(1, 2, 3) The cylindrical coordinates of the space point, (r, ϕ, z)

(4, 5) The (x,y) representation of r, ϕ). Some of the algorithms used in track-finding and -fitting work most naturally in rectangular space.

(6) A counter of the number of tracks the point is used in. This counter is intialized to zero; as JUBB puts this point into a track, this counter is bumped.

These coordinates are sorted by chamber, with those for the outermost chamber coming first. There is no ordering of points within each chamber.

Two points not on the z-axis are sufficient to define a helix constrained to pass thru the z-axis (that is, the beam line); three points determine a general helix. JUBB's general strategy is to take pairs (or triplets) of points on adjacent chambers, determine the constrained (or unconstrined) helix they define, calculate the intercepts of this helix with the ideal cylinders

that represent the spark chambers, and then see if sparks were indeed made on the chambers at approximately the predicted points in space. If so, then a track is formed from the sparks -- the usage counters for each of the coordinates is increased by one, and a 'track-list' is formed. ('Track-lists' are defined below.)

We, of course, wish to perform the track-finding in as short a time as possible. The different algorithms that JUBB uses, vary widely in the amount of computer time they burn up in building or rejecting a track. Therefore, we may require that the initial pair (or triple) of points be unused in tracks previously found; though subsequently-added points may have been used before. (In some cases, we only require one of the initial points to be unused.) Furthermore, we perform the fastest algorithms first; therefore, the set of possible pairs may be much reduced by the time the slower algorithms begin to test them.

The "track-lists" which JUBB turns out are defined as follows:

1) First, there is a count of the number of points in the track; call this number N.

2) Then come N words, each containing a pointer to one of the coordinates in the coordinate bank. The coordinate bank is part of a Fortran array; each pointer is the subscript in that array at which the first word of the coordinate for the point in question is stored. The pointers are stored in "line-of-flight" order -- that is to say, what we assume was the first spark created, will be pointed to by the first pointer word; and so on.

Thus, if a track begins in the interaction region and passes out thru the outermost chamber, then the pointer to the coordinate on the innermost chamber will appear first in the list. Then will appear the pointer for the coordinate on the second-intermost chamber, and so on; with the pointer for the point on the outermost chamber being the Nth and last word of the list. We assume that the direction of flight of the particle is away from the interaction region;

in a vanishingly-small proportion of intersting tracks that assumption will be incorrect, and the pointers will be stored in reverse line-of-flight order. There is no way this can be discovered by JUBB, however; it would not become apparent until a later part of the analysis process, when one attempted to hook the tracks up into vertices.

For example, suppose we have a two-point track. The coordinate of the outermost point is stored at works 116, 117, 118 ... in the array within which the coordinate is stored, and that for the innermost is in words 228, 229, 230 ... in that array. Then the first word of the track-list would be 2, since there are two points in the track. The second word would be 228, since that is the subscript of the first word of the coordinate of the point closest to the interaction region; and the third word would be 116.

It sometimes occurs that a sprak chamber does not fire when a particle crosses it, resulting in a missing coordinate for a track. Some of JUBB's scanners are allowed to construct track-lists in which points are missing, to take care of this situation. The position in the track-list for which no point was found is filled with a number whose value is less than the lowest possible value of a valid pointer; this dummy number carries some information about where the missing point was expected to be. Also, it often occurs that a presumed orbit found by JUBB has an unacceptably-bad least-squares fit to a helix; in such a case, the point whose contribution to the chi-square is greatest will be thrown out of the list, and its place in the list will be taken by another dummy subscript, linking in its own dummy fashion to the bad point. (This helix-fitting and point-heaving is done by subroutines subordinate to BUTT which are not discussed in this writeup.) These dummy subscripts are useful both for reconstructing the machinations of the program as it attempted to unravel a particular event, and also for attempting to recover from a bad initial reconstruction as discussed below.

3) Then come a series of words describing the track as a whole. It was originally intended that there should be perhaps a half-dozen of these words, containing quantities such as the radius of curvature and dip angle of the helix. In the off-line version, as of this writing, the number of such words has ballooned past 90. A listing of what each of these words contain is found as comments to the Fortran listing of subroutine BUTT.

JUBB's philosophy is to pick up all the helices it can find -- to make track-lists for all feasible combinations of points. Some of these are known to be irrelevant even at the time they are made; others must be thrown out further down the line, after JUBB is all thru. Therefore, a further level of pointers is provided, to pick out those track-lists which we wish to consider in further analysis of the event. The track-lists are kept in a Fortran array (the same array that the coordinate bank is kept in, in fact); this further array of pointers contains, for each track-list of current interest, the subscript of the array member which the first word of the track-list occupies.

There are currently seven track-finding algorithms that have been programmed. Each of them is embodied in a Fortran subroutine whose name is a girl's name; each of these routines is also given a 'scanner number'. The operative fiction is that JUBB (the name is taken from Keith Waterhouse's novel of the same name) bosses a gang of scanners, who pick over the events in turn, each pulling out just as much information as she can. Thus, the subroutine JUBB consists simply of calls to other subroutines that do the actual work. As a rule, it is not desirable to use every scanner in every run; therefore, ordinarily, some of these calls are deleted.

When one of the scanners finds a track, she puts together a track-list, using subroutine PLANT to make up the list; and when all the points are emplated, the scanner calls subroutine BUTT to compute and fill in the values of the after-pointer words. Thus, a basic flow-chart of the program looks like fig. 2.

2. Methods of Finding Tracks

We, here, describe the track-finding algorithms embodied in the various scanning subroutines. By "transverse section", below, we shall mean that section of the detector taken transversely to the beam -- the r-phi or x-y plane.

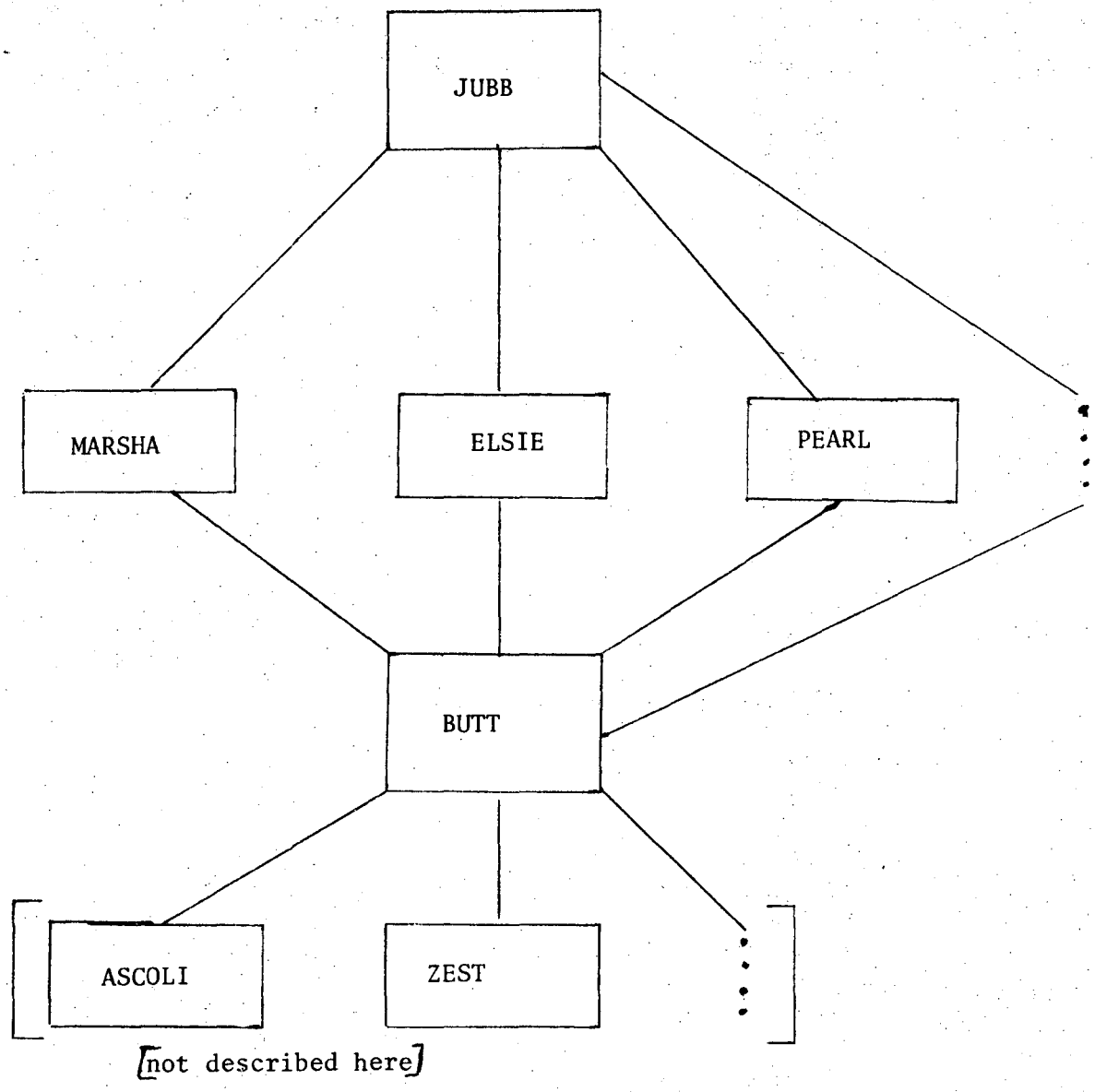
Scanner number 1, Subroutine MARSHA, is the fastest and crudest of them all. She can find the tracks of two-body events. Since the blob from which the two particles come has zero mementum, the two particles of a two-body event must have equal but opposite momenta. Therefore, as seen from the end of the chambers, the oppositely-charged particles will describe two halves of one large circle. Thus, in transverse section, the event will appear as a set of points lying along a single circular arc that passes thru the origin; and in longitudinal section, as a since curve passing thru the event vertex (x_o, y_o, z_o) in such fashion that $f(z_o + z) = -f(z_o - z)$, where $\underline{f}$ describes the trajectory. In most cases, of course, the longitudinal momentum of the particles is so great that the portion of the sine-curve within the detector volume is virtually a straight line.

To detect this configuration of points, MARSHA hunts for a chamber with exactly two points (r_1, ϕ_1, z_1) and (r_2, ϕ_2, z_2) on it. When such a chamber is found, we check that $z_o = z_1 + z_2$ is within the interaction region. If it is, we hold our presumed z_o and calculate $\gamma = (\phi_1 + \phi_2)/2$. Gamma is then either the azimuth of the center of curvature of the supposed tracks, or that azimuth + π.

We then check the other chambers for coordinate pairs with z-coordinates symmetrical about z_o, and ϕ's symmetrical about γ. If such are found, we list the pair of tracks.

Figure 2

BASIC FLOWCHART



MARSHA's great and only virtue is extreme speed. Since the vast majority of events seen are indeed two-body events, this algorithm is valuable for on-line use in checking that the accelerator and detector are working properly. In off-line event analysis, however, MARSHA is generally not called.

Scanner Number 2 is ELSIE. This routine finds fast, clean tracks coming out of the interaction region, by looking for sets of points that form straight lines in both r versus ϕ and r versus z.

The rationale behind this process is as follows: As expressed in polar (r, ϕ) coordinates, the equation of a circle whose circumference passes thru the origin, having radius ρ and center of curvature (ρ, α), is ordinarily written

$$r = 2\rho \cos(\phi + \alpha)$$

which immediately transforms to

1) $$\phi = \alpha \pm \text{arccosine}(r/2\rho).$$

But using the ordinary series expansion for arc-cosine, this is immediately equivalent to

2) $$\phi = \alpha \pm (\pi/2 - r/2\rho - 1/6(r/2\rho)^3 - 3/40(r/2\rho)^5 - \ldots)$$

Furthermore, starting with the parametric equations

3) $$r(t) = \sqrt{2\rho^2 - 2\rho^2 \cos(\omega t)}$$

4) $$z(t) = z_o + k\,t$$

one may derive the equation

5) $$z(r) = \pi/2 + p \pm q \text{ arc-cosine}(r/2\rho), \text{ where}$$

$$q = \omega/2\,k,$$

$$p = q^2\, z_o;$$

this equation has a similar expansion in odd powers of $r/2\rho$ with diminishing coefficients.

Now, from the definition of a circle it is obvious that any r-coordinate of a circle that passes thru the origin can never exceed 2ρ, its diameter

of curvature. Therefore, $r/2\rho \leq 1$; indeed, for most tracks in the SPEAR experiment, $r/2\rho << 1$ for even the coordinates on the outermost chamber. Thus, we see that for tracks of a greater-than-minimal transverse momentum, both z and ϕ are virtually linear in r, for we may ignore all terms of the expansion beyond the first.

ELSIE's tactic, then, is to pick a point (r_1, ϕ_1, z_1) on the outermost chamber and match it with each point (r_2, ϕ_2, z_2) on the next chamber in. Then, using the radius r_3 of the third chamber in, we calculate the predicted ϕ for the point on this chamber, via

$$\phi_3 = \phi_1 + (r_1 - r_3) * u,$$

$$u = (\phi_1 - \phi_2)/(r_2 - r_1)$$

u is then the inverse of the slope of the track's image in r-ϕ space.

If a point with such a ϕ is found, we then check that its z-coordinate is also reasonable, via a similar process. If points pass on all the chambers down to the innermost, then we figure we have a fast, clean track, and create a track-list for it.

We also check for tracks that never make it to the outermost chamber, by seeing if they leave the detector in z rather than in r. Such tracks are also listed. As currently written, ELSIE will not accept a track in which sparks are missing.

Like the first algorithm, this one's greatest virtue is speed. It is sufficient to handle the vast majority of events that appear in the SPEAR detector. If a particular event has a great number of points associated with it, however, this routine should not be used on it, because it quite deliberately emphasizes speed over exhaustive checking.

Scanner Number 3 is PEARL. This routine embodies the most rigorous search we make for tracks that pass thru (or near) the interaction region. It picks

up tracks with missing sparks and tracks which loop thru more than 180° in the chamber -- that is, particles whose transverse momentum is so small that they never make it to the outermost radius of the detector, but instead turn back towards the center. This routine should find any discernible helical segment whose extension passes reasonably close to the interaction region.

PEARL begins by matching pairs of previously-unmatched points on adjacent chambers, starting with the outermost. Now if these two points do indeed lie on a single orbit passing thru the interaction region, then we can calculate the diameter of curvature this track must have. We can either use the approximation derived from 2),

$$6) \qquad 2\rho = (r_1 - r_2)/(\phi_2 - \phi_1)$$

or for slower (that is, smaller-radius) tracks, where this approximation breaks down, we can use the exact relation

$$7) \qquad 2\rho = \sqrt{(2\cos(\phi_2 - \phi_1)r_1 r_2 - r_1^2 - r_2^2)/\sin^2(\phi_2 - \phi_1)} \ .$$

Once we have this 2ρ defined by the pair of points, then we may check that it is not greater than the maximum possible radius of curvature for a particle coming out of an actual colliding-beam interaction. This maximum is 33.3 E/B, with E the interaction energy in Mev and B the magnetic field in gauss. If the pair passes this test, then we may calculate the azimuth of the center of curvature, α, from 1). If this α lies within the smaller wedge subtended by ϕ_1 and ϕ_2, then the supposed track would loop outside the outer cylinder of the pair. This would mean that the points could not be 'neighbors' on the orbit, so we consider the pair no further.

For pairs that pass these tests, we can calculate the distance traveled along the path in transverse section from the origin to each point via

$$8) \qquad s_n = 2\rho \text{ arc-sine}(r_n/2\rho).$$

Now at any point along the track, the distance traveled in transverse section,

which is s, must be proportional to the distance traveled in z. We pick up this constant of proportionality from

$$k = (z_1 - z_2)/(s_1 - s_2)$$

and since then $dz = k\, ds$, we have that

$$z_o = z_n - k\, s_n \quad .$$

If this z_o falls within the interaction region, then the two points do indeed lie on a physically-meaningful possible orbit, and we can go on searching for the rest of the points such a particle would have made.

We continue on in this fashion, working first in transverse section and then expanding our view to take in the z dimension also. We have started at the outside of the detector, so we work our way inwards towards the origin cylinder-by-cylinder. For each cylinder, we predict the coordinates of the spark that the track should have made, and then search the coordinate bank to see if such a point was indeed measured. If so, we add this point to the track-list; if not, then we go on looking on the next cylinder, and do not quit until too many points have turned up missing.

After tracing our track back to the interaction region, we then reverse our direction of search and try to add on points to the outer end of the track. This is necessary, since scattering errors can accumulated along the track to the extent that a good approximation to 2ρ cannot be obtained from the outermost pair of points on the track. And this is also necessary for tracing out 'loopers' -- tracks that turn thru more than 180°. Note that it is easy for PEARL to tell when a track loops back -- it happens when the radius of the next outward cylinder to be crossed is greater than 2ρ.

It happens fairly often that PEARL will pick up a few points on a track that does not, in fact, pass thru the interaction region, and make a partial track from them. She will often be able to trace them thru three chambers, but miss the point expected on the inner chamber. Also, it is conceivable that

errors from wire scattering can build up to such an extent that the orbit defined by the momentum vector at a point far downstream from the vertex will no longer pass thru the origin in xy, even though the particle came from the primary vertex. This is especially likely in low-momentum tracks that loop in the chamber. Therefore, provision has been made for PEARL to call routine CONNIE when she misses expected points. This allows the orbit to be re-calculated from the last three points found, removing the constraint that the track should pass thru the interaction region. CONNIE is further described below.

If PEARL calls CONNIE and CONNIE finds some points, then we say that the track was found by scanner number 4. This is a way of picking up relatively-slow cosmic rays, electrons resulting from gamma-ray conversion, and stray particles that pass fairly near the interaction region.

PEARL works. If helices are there, she will find them. For a complex event, one with many points, it is best to let PEARL be the first to actually work on it; ELSIE and MARSHA should be disabled for such events.

Scanner Number 5 is MILDRED. This routine finds relatively straight tracks that do not pass thru the interaction region. In particular, it will find most cosmic ray tracks. It forms a track-list for these, even though they are not really of interest, as a means of helping to account for everything that goes on in the chamber.

MILDRED begins by finding a pair of previously-unused points which lie on adjacent chambers. It checks that the one on the inner chamber of the pair lies within the cone defined by the tangents to this chamber drawn from the point on the outer chamber, since otherwise the two would not be neighbors on a straight track. It then calculates the slope and intercepts of the straight line these two define, using cartesian coordinates. Then it calculates the distance of

closest approach of this line to the origin in transverse section; this distance tells us the innermost chamber the straight line would intercept.

The routine then goes off hunting points along the straight line, first one way, then the other. For the next chamber in line it calculates the two points of interception of the straight line with the ideal cylinder that represents the chamber. Then it chooses the predicted point which is closer to the last point added to the track, and hunts thru the coordinate banks for a point sufficiently close to the ideal. If such is found, it is added to the track-list and we proceed to the next chamber, or if we are on the innermost chamber reached, we simply look again for the other point predicted on the chamber and continue on looking outwards thru chambers of increasing radius.

MILDRED does not necessarily stop when she misses a point. Instead, she figures that the track is fact has some curvature, enought for instance that even though three points on it line up to form a straight line within the errors prescribed, a fourth point lies outside the allowable tolerances. So she will refigure the heading of the straight line, using the last two points added as the definers of the line, and with this new heading predict anew what the point on the next chamber should be. MILDRED only quits when no points lies sufficiently close to this second prediction.

MILDRED attempts to trace its tracks out two sides of the detector. Thus, after it has traced the line from the original two points as far as possible in one direction, it will begin trying to find points lying in the other direction from the original two also.

MILDRED is quite fast and is usually called in the off-line version. It handles most cosmic rays that are not picked up by PEARL -- those which pass too far from the interaction region for PEARL to be able to begin them. However, as is obvious from its method of search, it has a minimum xy momentum cutoff -- that is, the tracks it finds can deviate only so far from a straight line. The

fact that it requires a pair of points on neighboring chambers to be previously unmatched will sometimes cause it to miss a track since PEARL is sometimes a little over-enthusiastic in assembling partial tracks. This has not been a problem in the SPEAR experiment, since virtually all the tracks that MILDRED finds are, in fact, of no physical interest, but are only of interest in accounting for all the uninteresting things that happen in the chamber.

Scanner Number 6 is DOREEN. This routine applies the most general search of all to the remaining unmatched points. It should be able to find any helix that has a sufficiently great radius of curvature to intercept at least three chambers. Many of the tracks she finds are of no physical interest -- many are "knock-on electrons", that is, electrons in the body or atmosphere of the chamber which are sprung free from their atoms by a particle passing near by. But she could also find particles emerging from the decay of long-lived strange particles -- "vees" and "kinks", and electron pairs resulting from the conversion of a gamma ray that resulted from the decay of a neutral pion.

We begin with one point which has not been previously used and consider this point to be the origin of a new translated coordinate system. We will be working with the cartesian (x, y, z) representation of the coordinates, since we will be continually translating. Having this one point, we now try to find two more points such that they define a helix whose axis is parallel to the chamber axis. To do this we need a pair of points such that, in the translated coordinate system, the ratio z_2'/z_1' -- the displacements of the two points from the new origin in z -- is equal to the ratio s_2/s_1 -- the distances along the circle defined by the two new points and the new origin, of each from the new origin. To get this second ratio, we must first calculate the center of curvature of the circle, via straight-forward formulae we give here for reference. (x_2', y_2', z_2') is the translation of coordinate (x_2, y_2, z_2) to the new coordinate system in which (x_1, y_1, z_1) is taken to be the new origin. The formulae for center of curvature (x_c', y_c') then are

$$x_c' = (x_3'^2\, y_2' + y_3'^2\, y_2' - x_2'^2\, y_3' - y_2'^2\, y_3')/(2\, x_3'\, y_2' - 2\, x_2'\, y_3')$$

$$y_c' = (x_2'^2 - 2\, x_2'\, x_c' + y_2'^2)/2\, y_2'$$

Then ρ, the track's radius of curvature, is simply found --

$$\rho = \sqrt{x_c'^2 + y_c'^2}$$

At this point, we call CONNIE to add more points onto the helix.

From the criteria she uses, DOREEN cannot be sure that the three points she has used to define the helix are, in fact, neighbors on the orbit they define; for any three points on the orbit will satisfy the relation $dz_1/ds_1 = dz_2/ds_2$. She tries to cope with the situation by considering first the point on an adjacent chamber that is nearest to the original point in z, then the point next-nearest, and so on; for, as a general rule, tracks are monotonic in their z-coordinates and when this is so we'll pick up the first points nearest two neighbors first. But this is not an infallible criteria. Therefore, we allow CONNIE to refind the first three points, for by predicting the next point on the orbit and then searching for it, CONNIE can pick up any intervening points that lie between two of the first three.

The first thing CONNIE must decides is which will be the next cylinder to be crossed by the track. At any point along the orbit there are only two possibilities -- a track either hits the next cylinder in the radial order in which it has been traveling, or else it must pass back thru the cylinder last passed thru. Therefore, at any point it is sufficient to check whether the arc of the track in xy intersects the next cylinder in order or not.

Now, if D is the distance between the track's center of curvature and the origin in xy, and ρ is its radius of curvature, then the track's closest approach to the center is $D - \rho$ and its furthest regress, $D + \rho$. Therefore, if r_c is the nominal radius of the next chamber in order, we need merely check that

$$|D - \rho| \leq r_c \leq D + \rho \ .$$

If not, then the track must loop back thru the same cylinder just passed thru.

CONNIE predicts the ϕ of the next point using the equation for a general circle (not constrained to pass thru the origin) in polar coordinates, which may be written

$$\phi = \alpha \pm \text{arc-cosine}((r^2 + D^2 - \rho^2)/2\ r\ D)$$

Since r is merely the nominal radius of the chamber in question, it is simply a matter of direct substitution to predict ϕ. (D, α) is the coordinate of the track's center of curvature, and ρ is its radius of curvature.

The DOREEN/CONNIE combination has not been used in actual production runs at this writing, and it is known that a few bugs remain.

Scanner Numbers 7 and 8 do not currently exist. It is planned that Number 7 (LUANNE) will pick up points that were either missed by one of the earlier scanners or for which the earlier scanner made a poor choice -- the poorness of the choice being revealed by a high point χ-square when a least-squares fit to a helix is made (by routines ASCOLI and ZEST, subordinate to BUTT). Its implementation awaits a further rationalization of the coordinate-bank and a more sophisticated track list. And this waits upon the time when the pace of data-taking for the SPEAR magnetic-detector experiment becomes less frantic.

LUANNE has the advantage over the earlier scanners in that she can use the preliminary best-fit orbits to predict the locations of further points, rather than simply using the orbit defined by two (or three) points. We anticipate it will be of some value in cleaning up events with a large number of sparks.

Scanner Number 8 (IDA MAE) was originally planned to choose the best track(s) out of a sheath of tracks which have many of the same points in common. However, it is now clear that this task is inextricibly intertwined with that of topological reconstruction; that is, hooking the tracks up into vertices; and thus IDA MAE is permanently out to pasture.

We have no figures on the actual running times of the various routines of JUBB; however, we do know that their operation is a gratifyingly-small part of the total time used in completely analyzing an event. That is, the time required for JUBB to find the tracks is considerably less than the time required to do a least-squares fit of the found event. We may also note that the time used in analyzing an event can be varied over orders of magnitude, depending upon which scanners are called on to analyze it.

It is hard to give any figures for accuracy, since we have no other method of reconstructing tracks that approaches these routines in accuracy. When it can be seen that JUBB made a mistake, we change the program to take account of the problem; and when JUBB cannot make sense of an event, neither can anybody else. A study using simulated events created by program HOWL reported virtually 100% efficiency for the tracking routines -- which is to say, if helices are created, they will be found. This is not to say that JUBB can make sense of anything like 100% of the real events it is faced with, however.

APPENDIX

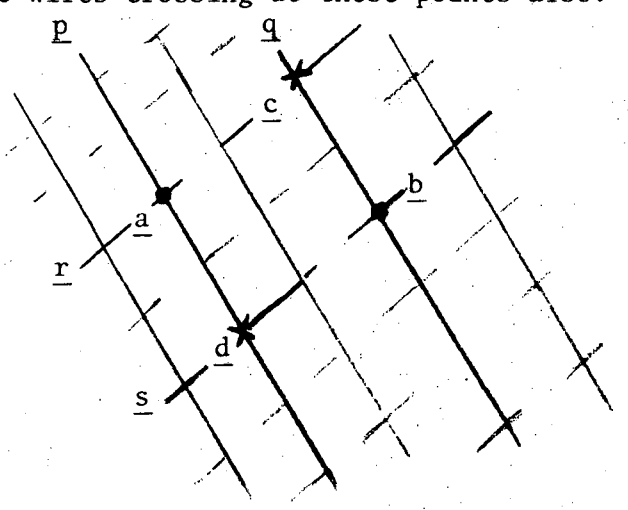
"Twins"

As we have said, the SPEAR magnetic detector has four cylindrical "chambers" from which space coordinates are read out. Each of these "chambers" has two spark gaps, however, not one. We now explain the rationale behind this, and the complexities arising from it.

The idea behind building the detector this way was to avoid the construction of spurious space points. Consider the small section of a spark chamber drawn below. The solid lines may be taken to be wires of one charge lying in a plane parallel to the page; the dotted lines are then wires of the opposite charge, lying in a plane parallel to the first plane and perhaps a centimeter removed from the first plane. Two charged particles pass perpendicularly through the chambers; sparks then travel from a wire in one plane to a wire in the other, approximately along the lines-of-flight of the particles. These sparks, we shall say, are at points a and b on our drawing; these sparks then cause read-outs on wires p, q, r, and s.

Now, of course, all our program knows about is which wires of the chamber actually pulsed. We presume a spark occurred at the point where two pulsed wires cross, and we then attempt to reconstruct tracks on the basis of the spatial locations of these hypothesized sparks. But note on our drawing that sparks could equally well have occurred at points c and d, given merely that wires p, q, r, and s fired; for we have two wires crossing at these points also.

Small section of a spark chamber



It is fair to say that the difficulty of reconstructing tracks in this detector was considerably over-estimated at the time this experiment was designed. In particular, confusion of the (at-that-time-unwritten) track reconstructing program with spurious points was a possibility that the original designers wanted to avoid at all costs. Therefore, the four chambers were designed with dual spark gaps, the hope being that 'real' points would show up on both gaps, while the spurious wire-crossings would not have a mate on the alternate spark gap, and could on this basis be discarded.

In fact, this scheme seems to have gained us little. Few such spurious points arise; and when they do, they do not seem to confuse JUBB. (No figures are possible, since we have no way of doing a 'check-scan' -- eyeball methods seem to be less accurate than the programs.) A complimentary problem also arises, in that spark gap efficiencies are far from 100%, and, therefore, we cannot throw away points which appear on only one of the two spark gaps of a chamber; rather, we ordinarily assume that the missing spark simply got lost in the electronics.

This is the origin of the numerous comments in the source-listing referring to "two-wire crossings", "four-wire coordinates", etc. This is merely an additional slight complexity, and not noteworthy in itself. However, a great mistake was made in the setting up of data structures to take care of this complexity and anyone involved in understanding, maintaining, or modifying JUBB needs to know about this mistake, in order to work around it or (hopefully) to correct it. (It has not seemed feasible to do the extensive recoding necessary for correction while the experiment is actually running, which at this writing it still is.)

For each entry in the coordinate bank -- that is, for each point that JUBB actually sees -- there is a flag-word giving various information about the coordinate. We have not discussed this flag-word before this point. If a point

on one spark gap of a chamber has no mate on the alternate spark gap, the spatial location of the one present spark is stored in the coordinate bank, with an entry in the flag-word indicating that this is a two-wire crossing. If sparks on the two gaps should match up, however -- 'matching up' means that the straight line drawn thru the two passes approximately thru the center of the interaction region, the coordinate-system origin -- the location of only one spark is stored in the coordinate bank, with its flag-word indicating that this is one gap of a four-wire crossing. The spatial location of the other spark is stored in an array contained in labeled common block /TWIN/, with the subscript of the first word of the coordinate in /TWIN/ being the same as the subscript of the first word for the coordinate in the main coordinate bank -- this is how twins are accessed.

This method of storage has a sort of sophomoric niftiness to it, but has led to innumerable problems in coding -- besides being utterly wasteful of storage. Time that should have been spent on understanding and improving the track-reconstruction process has had to be squandered on coding around this mistake. Indeed, it is felt that no further development of JUBB should take place until this correction is made -- which effectively means that no development is, in fact, taking place.

It is recommended that anyone attempting to adopt JUBB to a new experiment make the abolishment of /TWIN/ his first order of business. Besides improving the understandibility and maintainability of the program immeasurably, this will have the incidental effect of acquainting one with virtually every nook and cranny of the routines; for almost everything has had to deal with the /TWIN/ block somehow.

0 0 0 4 3 0 1 3 5 5

LEGAL NOTICE

This report was prepared as an account of work sponsored by the United States Government. Neither the United States nor the United States Atomic Energy Commission, nor any of their employees, nor any of their contractors, subcontractors, or their employees, makes any warranty, express or implied, or assumes any legal liability or responsibility for the accuracy, completeness or usefulness of any information, apparatus, product or process disclosed, or represents that its use would not infringe privately owned rights.

TECHNICAL INFORMATION DIVISION
LAWRENCE BERKELEY LABORATORY
UNIVERSITY OF CALIFORNIA
BERKELEY, CALIFORNIA 94720