Remote Monitoring of the Calibration of a System of Tracking Arrays

ROBERT R. READ

(Invited Paper)

Abstract—Post-test analysis of vehicle tracking data taken from a three-dimensional underwater range requires the splicing together of several pieces of track individually produced by distinct transducer arrays. The location and orientation of the arrays must be known precisely in order to convert locally determined track into a coherent record in the general range coordinate system. The maintenance of calibration of the system is a problem and the paper produces some least square methodology that uses the tracking data itself to monitor it. More specifically, algorithms are developed to estimate array displacement and orientation corrections. Since such corrections must be relative to a specific array, methodology is presented to help identify the more stable arrays.

I. INTRODUCTION

This paper deals with the problem of monitoring the calibration of a multiple array underwater tracking range. The arrays in the system are of the short baseline type; each contains four sonar transducers placed rigidly at the corners of a cube in a manner that describes a Cartesian coordinate system in three dimensions (see Fig. 1). They receive a distinctive signal from a synchronously timed pinger attached to the target vehicle. The differentials of the sound wavefront’s times of arrival at the four hydrophones allow the computation of the azimuth and elevation angles of the normal to the wavefront at the origin of the local coordinate system. Then, assuming direct path propagation, one can ray trace using Snell’s law, [4], starting with the aforementioned elevation angle and utilizing a velocity-versus-depth profile for the speed of sound in the water. Finally, the time differential between the source pulse at the target vehicle and its arrival at the array is used to stop the ray-tracing algorithm and determine the location of the target relative to the array. The local track is the sequential set of these estimated positions.

Each array in the system operates over a limited radius. As the target sojourns through the range, it is tracked by a number of these arrays. See Fig. 2 for a plan view. (The zero level in the vertical is taken as mean sea level.) The overall path is constructed by transforming each piece of local track to the coordinates of the range based upon the assumed location and orientation of the various local coordinate systems. Discontinuities, or mismatches occur because the track produced by one array does not mesh exactly with that produced by a neighboring array. A major goal is to address the question of whether these mismatches can be attributed to slippages in array position and orientation, or whether some other source of error must be treated. This is our calibration monitoring problem. The present paper develops an algorithm for the simultaneous estimation of all the displacement and orientation corrections.

Generally there are a number of occasions (i.e., several sequences of time points, hereafter called point count sets) for which two arrays simultaneously produce track. These occur in the overlap regions indicated in Fig. 2. The paired tracking data of these point counts are called ‘‘crossover data.’’ It is the crossover data generated as a result of the target vehicle’s entire trip that provide the input to the estimation algorithm. Each version of track in a crossover data set is assumed to have been converted to the (common) range coordinate system.

A least squares approach is taken to the problem. In the simplest case, two arrays providing a single crossover data set, we seek a displacement and orientation correction for one of the arrays that will minimize the sum of square deviations of the two versions of track, point by point. It is this case that is treated first in Section III, following some preliminaries in Section II. Note that the corrections estimated are in the relative sense, one array relative to another. The general case of multiple crossover sets is treated in Section IV. Again the corrections are relative, but with many arrays in the problem. Only one of them needs to be designated as a reference array provided a certain ‘‘connectedness’’ criterion is met. An example involving real data is included.

If all of the assumed positions and orientations of the arrays in the crossover set are valid, then the estimated corrections will be small (i.e., within the limits of experimental error) and it matters little which of the arrays is used as the reference array. On the other hand, if one or more of them is ‘‘out of sync’’ then an identification problem appears. Presumably there will be only a few (hopefully one) miscreant arrays and the estimated corrections for them will stand out, provided the reference array is not one of them. Because of this caveat we have extended the output of the algorithm so that each array in the problem successively plays the role of the reference or base array, and the output is summarized with the magnitude of displacement and rotation estimates of the other arrays. The description of this extension and application to the example appears in Section V.

II. MATHEMATICAL PRELIMINARIES

In the formulas that follow, and V are column vectors, and B is a square matrix. The quantities $\partial z/\partial C$ (for scalar z)
Consider a set of point counts $S$ in a crossover region, and let the crossover data (in Range coordinates) be

$$y(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \end{bmatrix}$$

and

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}$$

for $t \in S$. If $B$ and $C$ are square matrices of the same order, liberal use will be made of the fact that $\text{Trace}(BC) = \text{Trace}(CB)$ and that the trace is a linear operator.

Provided by two different sensing arrays. It is preferred that these data be noise free; e.g., they might be the result of applying a smoothing filter to the raw data. Then only the

will have the same algebraic structure as $C$ and the elements will be the partial derivatives of $z$ with respect to the elements of $C$. The superscript $T$ refers to matrix transpose. The following formulas can be found in [3] and will be used in the sequel:

$$\frac{\partial V^T A}{\partial A} = V$$

$$\frac{\partial (A^T A)}{\partial A} = 2A$$

$$\frac{\partial (V^T B A)}{\partial B} = VA^T$$

$$\frac{\partial (V^T B^T B V)}{\partial B} = 2B V V^T.$$  \hspace{1cm} (1)

If $B$ and $C$ are square matrices of the same order, liberal use will be made of the fact that $\text{Trace}(BC) = \text{Trace}(CB)$ and that the trace is a linear operator.

Consider a set of point counts $S$ in a crossover region, and let the crossover data (in Range coordinates) be

$$y(t) = \begin{bmatrix} y_1(t) \\ y_2(t) \\ y_3(t) \end{bmatrix}$$

and

$$x(t) = \begin{bmatrix} x_1(t) \\ x_2(t) \\ x_3(t) \end{bmatrix}$$

for $t \in S$. If $B$ and $C$ are square matrices of the same order, liberal use will be made of the fact that $\text{Trace}(BC) = \text{Trace}(CB)$ and that the trace is a linear operator.

Provided by two different sensing arrays. It is preferred that these data be noise free; e.g., they might be the result of applying a smoothing filter to the raw data. Then only the
modeling errors could be propagated throughout the algorithm. The example in Section IV is the result of applying the method to raw data (noise included).

Let us agree that the \( y(t) \) data come from the array whose location and orientation are established and our goal is to check the calibration of the other array. In particular, does there exist a 3-vector \( A \neq 0 \), and/or a 3 \( \times \) 3 matrix \( B \) (\( \neq I \), the identity matrix) such that the adjusted track values

\[
\hat{x}(t) = A + Bx(t)
\]

are in better agreement with the \( y(t) \) than are the unmodified \( x(t) \).

The vector \( A \) is relative to a displacement of the sensing array and the matrix \( B \) is related to a correction of its orientation.

If we let \( z(t) \) be \( x(t) \) in the local coordinate system, \( \alpha \) be the location (in range coordinates) of the array, and \( \beta \) the (orthonormal) orientation adjustment to the local coordinates, then

\[
x(t) = \alpha + \beta z(t)
\]

and

\[
\hat{x}(t) = A + B\alpha + B\beta z(t).
\]

The corrected location and orientation adjustments are \( A + B\alpha \) and \( B\beta \), respectively.

Our immediate goal is to estimate \( A \) and \( B \) using the principle of least squares. We will minimize the average square deviation between \( y(t) \) and \( A + Bx(t) \) for each point count. Using the squared norm notation, \( \| V \|^2 = V^TV \), define an objective function

\[
Q = \text{Ave} \| y(t) - A - Bx(t) \|^2
\]

and, since what follows involves a modification of the standard multivariate regression development, let us outline the details.

First, the estimator for \( A \) is

\[
\hat{A} = \bar{y} - \hat{B} \hat{x}
\]

where \( \bar{y} = \text{Ave}_t y(t) \) and \( \hat{x} = \text{Ave}_t x(t) \). The proof follows from an expansion of (4), namely

\[
Q = \text{Ave}_t \left[ y(t) - Bx(t) \right]^T \left[ y(t) - Bx(t) \right] - 2 \text{Ave}_t \left[ y(t) - Bx(t) \right]^T A + A^T A
\]

to which we apply the rules in (1):

\[
\frac{\partial Q}{\partial A} = -2 \text{Ave}_t [y(t) - Bx(t) - A].
\]

Set this equal to zero, solve for \( A \) and the result follows.

Next, let us use (5) in (4), make the change

\[
Y(t) = y(t) - \bar{y}
\]

and

\[
X(t) = x(t) - \hat{x}
\]

and work with the deviations marked with capital letters. The result is a simplified appearance for \( Q \):

\[
Q = \text{Ave}_t \| Y(t) - BX(t) \|^2.
\]

Also introduce the covariance matrices

\[
D_{yy} = \text{Ave}_t \{ Y(t) Y^T(t) \}
\]

\[
D_{yx} = \text{Ave}_t \{ Y(t) X^T(t) \}
\]

\[
D_{xx} = \text{Ave}_t \{ X(t) X^T(t) \}
\]

\[
D_{xy} = D_{yx}^T.
\]

One can get the gradient of \( Q \) with respect to the elements of \( B \) by expanding (7):

\[
Q = \text{Ave}_t \{ Y^T(t) Y(t) - 2Y^T(t)BX(t) + X^T(t)B^TBX(t) \}
\]

and then use (1) and (8) (or use [1, p. 181]) to get

\[
\frac{\partial Q}{\partial B} = -2 \text{Ave}_t \{ Y(t) X^T(t) - BX(t)X^T(t) \} = -2\{ D_{yx} - BD_{xx} \}.
\]

There is a constraint that accompanies the minimization of \( Q \). Recall that the sensing arrays are rigid local Cartesian coordinate systems. If they have slipped (i.e., moved physically such as by the action of a ship's anchor hooking a cable) then they undergo a displacement and a reorientation. The matrix \( B \) should be orthonormal as is the matrix \( \beta \) in the original calibration. Thus

\[
B^TB = I
\]

and the estimation of \( B \) involves the minimization of \( Q \) subject to this constraint. An unconstrained estimate would be useful only if there were physical damage to an array, resulting in its behavior as a skewed rather than Cartesian system.

Use of (11) and (8) in (9) allows the rewrite (of 7)

\[
Q = \text{Trace} \{ D_{yy} - 2Y^T(t)B + X^T(t)B^TBX(t) \}
\]

and the minimization of \( Q \) is tantamount to the maximization of

\[
W = \text{Trace} \{ D_{yx}B^T \}
\]

subject to (11) since the trace is a linear operator and \( B \) is involved only in the second term of (12). The solution of the optimization problem (13) and its generalization in Section IV is the bulk of our effort.

Earlier it was stated that the estimated corrections are applied to one array relative to the other one. Otherwise, the solution cannot be unique. Let us show now, using analysis, why this is the case in the two-array single-crossover region.
problem. Suppose we tried to adjust both data sets, i.e.,
\[ \hat{y}(t) = A_1 + B_1 y(t) \]
and
\[ \hat{x}(t) = A_2 + B_2 x(t) \]  \hspace{1cm} \text{(14)}
where both \( B_1 \) and \( B_2 \) are constrained to be orthonormal. The least squares objective function has the appearance
\[ Q = \text{Ave } \| A_1 + B_1 y(t) - A_2 - B_2 x(t) \|_2^2. \]  \hspace{1cm} \text{(15)}
It is clear from the development of (5) that the displacement portion of the optimization can be estimated only relatively. That is, only \( A = A_2 - A_1 \) can be found uniquely. From the earlier development we can infer that \( \hat{A} = \hat{B}_1 \hat{y} - \hat{B}_2 \hat{x} \) and reexpress the objective as
\[ Q(B_1, B_2) = \text{Ave } \| B_1 Y(t) - B_2 X(t) \|_2^2. \]  \hspace{1cm} \text{(16)}
This is the average squared length of a set of vectors, however. Since the squared length of any vector is invariant under any orthonormal transformation, say \( C \), it follows that
\[ Q(B_1, B_2) = Q(CB_1, CB_2). \]
In particular we can take \( C = B_1^T \) (i.e., the inverse of \( B_1 \)) and then \( Q(B_1, B_2) = Q(I, B_1^T B_2) \). The product of orthonormal matrices is itself orthonormal. We can set \( B = B_1^T B_2 \) and note that the minimization of \( Q(I, B) \) is that of \( (7) \), the problem we are treating. The geometrical interpretation is that the orientation of a sensing array can be matched only relatively to the other array. It is convenient to assume that the latter’s orientation is known.

**III. SOLUTION ALGORITHM**

The matrix \( B \) has nine elements, but because of the constraint (11) there are only three degrees of freedom. That is, there are three length constraints and three orthogonality constraints. The three remaining degrees of freedom can be expressed in terms of the Euler angles [2].

To shorten the writing, let
\[ c_i = \cos (\phi_i) \]
and
\[ s_i = \sin (\phi_i) \]  \hspace{1cm} \text{(17)}
for \( i = 1, 2, 3 \), and the individual planar rotations can be expressed as
\[
\begin{bmatrix}
1 & 0 & 0 \\
0 & c_1 & -s_1 \\
0 & s_1 & c_1 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
c_2 & 0 & -s_2 \\
0 & 1 & 0 \\
s_2 & 0 & c_2 \\
\end{bmatrix}
\]

\[
\begin{bmatrix}
c_3 & -s_3 & 0 \\
s_3 & c_3 & 0 \\
0 & 0 & 1 \\
\end{bmatrix}
\]  \hspace{1cm} \text{(18)}
and, using the order of application described, let
\[ B = \rho_3 \rho_2 \rho_1. \]  \hspace{1cm} \text{(19)}
We make liberal use of the fact that the product of orthonormal matrices is itself orthonormal.

We note in passing that the matrices \( \rho_i \) above are the transposes (and hence inverses) of those that are usually used in studying the motion of airplanes [2]. Also, the order of application is the reverse of the customary one thus making \( B \) in (19) the inverse of the usual matrix. This choice is appropriate for our work since the angular corrections will rotate the local axes back to where they are expected to be.

This done, our optimization problem may be stated as
\[ \text{Maximize Trace } \{ D_{3x} B^T \}. \]  \hspace{1cm} \text{(20)}
The solution can be found by treating the problem as three successive two-dimensional problems. To do this, let us first characterize how this optimization problem would be solved if the tracking took place in two dimensions.

In the plane, an orthonormal matrix \( B \) has one degree of freedom and it is characterized by a rotation through an angle \( \phi \). As before, let \( c = \cos (\phi), s = \sin (\phi) \) so that
\[ B = \begin{bmatrix} c & -s \\ s & c \end{bmatrix}. \]
and
\[ W = \text{Trace } (DB^T) = (D_{11} + D_{22}) c + (D_{21} - D_{12}) s. \]  \hspace{1cm} \text{(21)}
Thus \( W \) is itself a sine wave. It has one maximum and one minimum, and these are separated by \( \pi \) radians. Since, using primes to denote derivatives,
\[ W' = -(D_{11} + D_{22}) s + (D_{21} - D_{12}) c \]
and
\[ W'' = -(D_{11} + D_{22}) c - (D_{21} - D_{12}) s \]

it is easily seen that the maximum occurs when
\[ s = k(D_{21} - D_{12}) \]
and
\[ c = k(D_{11} + D_{22}) \]
and
\[ k = [(D_{21} - D_{12})^2 + (D_{11} + D_{22})^2]^{-1/2}. \]  \hspace{1cm} \text{(22)}
This would be the solution if the tracking problem were a two-dimensional one.

Now we are positioned to treat our three-dimensional problem. Introduce matrices
\[ E = D_{3x} \rho_1^T \rho_2^T \]
\[ F = \rho_3^T D_{3x} \rho_1^T \]
\[ G = \rho_3^T \rho_2^T D_{3x} \]  \hspace{1cm} \text{(23)}
and note that, using (13) and (19)

\[ W = \text{Trace} \{ D_{\phi, B}^T \} = \text{Trace} \{ E\rho_1^T \} = \text{Trace} \{ F\rho_2^T \} = \text{Trace} \{ G\rho_3^T \}. \] (24)

These latter three representatives of \( W \) allow us to use our knowledge of the solution to the two-dimensional problem. When these three traces are written out in expanded form and compared with (21) and (22), it is seen that the optimal solution for \( \phi_1, \phi_2, \phi_3 \) must have the form

\[
\begin{align*}
s_1 &= k_3 (E_{21} - E_{12}) + c_3 = k_3 (E_{11} + E_{22}) \\
s_2 &= k_2 (F_{31} - F_{13}) + c_2 = k_2 (F_{11} + F_{33}) \\
s_1 &= k_1 (G_{32} - G_{23}) + c_1 = k_1 (G_{22} + G_{33}) \tag{25}
\end{align*}
\]

where

\[
\begin{align*}
k_3 &= \left( (E_{21} - E_{12})^2 + (E_{11} + E_{22})^2 \right)^{-1/2} \\
k_2 &= \left( (F_{31} - F_{13})^2 + (F_{11} + F_{33})^2 \right)^{-1/2} \\
k_1 &= \left( (G_{32} - G_{23})^2 + (G_{22} + G_{33})^2 \right)^{-1/2}. \tag{26}
\end{align*}
\]

The above does not provide explicit solutions because \( E = E(\phi_1, \phi_2), F = F(\phi_1, \phi_2), \) and \( G = G(\phi_1, \phi_2) \). It is necessary to get all three angles satisfying (25) and (26) at the same time. This means in addition that

\[ \nabla W = 0 \] (27)

as well. We know that each of the equations

\[
\begin{align*}
\frac{\partial W}{\partial \phi_1} &= \text{Trace} \left\{ E \frac{\partial \rho_1^T}{\partial \phi_1} \right\} = 0 \\
\frac{\partial W}{\partial \phi_2} &= \text{Trace} \left\{ F \frac{\partial \rho_2^T}{\partial \phi_2} \right\} = 0 \\
\frac{\partial W}{\partial \phi_3} &= \text{Trace} \left\{ G \frac{\partial \rho_3^T}{\partial \phi_3} \right\} = 0 \tag{28}
\end{align*}
\]

has two solutions, one maximum and one minimum. It follows that (27) has eight \( (2^3) \) solutions, one for each of the permutations of the component solutions, and this means one maximum, one minimum, and six saddle points. We require an algorithm that converges to the maximum.

This can be achieved with an iterative process. Take an arbitrary beginning set of values for \( \phi_1, \phi_2, \phi_3 \). All \( \phi_i = 0 \) will do. Compute \( E \) for this choice and then compute a new \( \phi_1 \) from the first member of (25). Use this value and the original \( \phi_1 \) to compute \( F \) and from this compute a new \( \phi_2 \) from the second member of (25). Next, use these new values for \( \phi_2, \phi_3 \) and compute \( G \), followed by a new \( \phi_1 \) from the third member of (25). This completes one cycle. At each step the appropriate member of (28) was satisfied and with a maximum. The value of \( W \) increased each time. Moreover, a second cycle beginning with the values \( \phi_1, \phi_2, \) and \( \phi_3 \) from the first cycle would produce a further increase in \( W \). We know there exists a unique maximum for \( W \) and we have an iterative method that increases \( W \) at each step. It follows that, by repeating the process, we can come arbitrarily close to this maximum and that will be signaled when all three members of (28) are negligibly small in magnitude, i.e., less than \( \epsilon \). Our experience has been that the surface \( W \) is rather flat and quite small values for \( \epsilon \) are required \((10^{-7} \text{ seems to produce stable results})\).

In order to develop intuition, let us take pause and make a heuristic interpretation of our iterative process. The objective function \( W \) is a sine wave as a function of each angle \( \phi_i \) (with the other two angles held fixed). As such it is concave in half of this restricted angular domain. Hence, in the full three-dimensional space of \( (\phi_1, \phi_2, \phi_3) \), the function \( W \) is concave in one-eighth of its domain. A more general (and perhaps more rapidly converging) gradient search algorithm should begin in this part of the domain so that when convergence takes place, it is toward a maximum. Although the algorithm presented does not require the user to worry about this point, it is a point of concern for speed of convergence and for other iterative algorithms [5].

IV. General Case

To treat the general case we must consider several arrays, say \( K \) in number, and several crossover regions. Also, we must allow for the target to be tracked in a given crossover region either not at all, exactly once, or more than once since it may maneuver back into a given region during a later point count set. Finally, the target, during a given point count set in a crossover region, may be tracked by more than two sensing arrays. We proceed to develop a notational structure that can handle the fully general case.

Let \( S_1, S_2, \ldots, S_R \) represent a collection of \( R \) point count sets. It is convenient to assume that to each individual \( S_i \) (for \( s = 1, \ldots, R \)) there is associated exactly one pair of sensing arrays. Thus, if only one array tracks the target in a particular crossover region, there will be no corresponding point count set in the collection. If exactly two arrays track the target, then the particular \( S_i \) is well defined. If three or more arrays track the target at about the same time, then things become a little fuzzy because the point count set for one pair of arrays may not be exactly congruent with the point count set of the crossover data of one of the two arrays with a third array, etc. This possible lack of congruence does not affect the computation of the cross covariance (8), as they refer only to pairs of arrays. Thus there is no harm in allowing the sets \( S_1, \ldots, S_R \) to be unique only in terms of the array pair that they represent. The repetition of some (even all) individual point count sets is allowed in two distinct point count sets provided the array pairs are not the same.

Next, for \( i = 1, \ldots, K \), let \( C_i \) be the subcollection of \( S_1, \ldots, S_R \) which contain tracking data from the \( i \)th array. In this way we can identify a three-dimensional vector of track data, \( x_i(t, s) \), as being produced by the \( i \)th array at point counts which belongs to \( S_i \). These quantities exist only if \( S_i \in C_i \).

Our goal is to estimate the displacement and reorientation parameter pairs for each of the \( K \) arrays.Earlier, with \( K = 2 \), we learned that there is an identifiability problem and it was convenient to assume that the location and orientation of one of the arrays was fixed. The same is true in the general case (i.e., only one array is fixed) provided that the data satisfy a
connectivity condition. This condition can be described in the parlance of graph theory. The $K$ sensing arrays form the nodes. An arc exists between two nodes if crossover data exist between them. A path is an unbroken sequence of arcs connecting one array (start of path) to another (end of path). We wish to consider only connected graphs. This means that there is no partition of the nodes into two nonempty sets such that a path cannot be found connecting a node of one set to a node of the other set.

If this condition of connectedness is not satisfied by our problem, then the overall data must be decomposed into smaller groups so that they hold for each group. Such decompositions are unique, and each member of the decomposition is to be treated separately. This done, we can fix our analysis on the case that involves a single connected graph. Here it will be seen that the estimation of all displacements and reorientation parameters will be unique once that one array is specified as the reference array.

Our notation needs to be expanded. Let $x_i(t; s)$ be the three-dimensional track, in range coordinates, produced by the $i$th array at point count $t$ belonging to $S_i$. Our goal is to estimate the location vectors $A_i$ and reorientation matrices $B_i$ so that (compare (2))

$$ x_i(t; s) = A_i + B_i x_i(t; s) \quad (29) $$

are as compatible as possible with the overall track of the target in the range. Also let $\bar{x}_i(s)$ be the vector of averages of the $x_i(t; s)$ taken over $t$ in $S_i$, and analogously to (6), let

$$ \bar{X}_i(t; s) = x_i(t; s) - \bar{x}_i(s) \quad (30) $$

be the deviations from the mean. Our new objective function can be expressed as an extension of (7) or (16):

$$ Q = \sum_{i<j} \sum_s N_s \text{Ave} \| B_i X_i(t; s) - B_j X_j(t; s) \|^2 \quad (31) $$

where the outer double sum is over $1 \leq i < j \leq K$, the inner sum is over $s \in C_i \cap C_j$, and the average is to be taken over $t \in S_i$. The weights $N_s = \text{number of point counts in } S_i \text{ for the array pair } (i, j)$. The requirement that $i < j$ keeps the array pairs unique as well as the associated $B$ matrices.

We can view (31) as

$$ Q = Q(B_1, B_2, \cdots, B_K) \quad (32) $$

and, because of the connectedness assumption, (31) cannot be decomposed into the sum of two terms

$$ Q_1(B_{i_1}, \cdots, B_{i_p}) $$

and

$$ Q_2(B_{j_1}, \cdots, B_{j_p}) $$

for which the subscript sets $i_1, \cdots, i_p$ and $j_1, \cdots, j_p$ form a partition of $1, 2, \cdots, K$. Now we can repeat the argument used at the end of Section II. If $C$ is any orthonormal transformation, then

$$ Q(CB_1, CB_2, \cdots, CB_K) = Q(B_1, \cdots, B_K). $$

Using $C = B_1^T$, say, then

$$ Q(B_1, \cdots, B_K) = Q(I, B_1^T B_2, \cdots, B_1^T B_K) \quad (33) $$

and for convenience, it is assumed that the orientation of the first array is known, i.e., $B_1 = I$ in (31).

Define covariance matrices

$$ D_{ij}(s) = \text{Ave} \{ X_i(t; s) X_j^T(t; s) \} \quad (34) $$

where the average is taken over the point counts $t$ in $S_i$. These quantities exist only for $s \in C_i \cap C_j \neq \emptyset$. Then (31) may be expanded to the more useful form

$$ Q = \sum_{i<j} \sum_s N_s \{ D_{ij}(s) - 2D_{ij}(s)B_i^TB_i + D_{ij}(s) \} $$

and again, because the unknown matrices $B_{ik}$ appear only in the second term, we may choose to maximize

$$ W = \sum_{i<j} \text{Trace} \{ D_{ij}B_i^TB_j \} \quad (36) $$

for $1 \leq i < j \leq K$ and

$$ D_{ij} = \sum_s N_s D_{ij}(s) \quad (37) $$

or zero, according to whether the summation for $s \in C_i \cap C_j$ has content, or is empty.

A word about the computation of $D_{ij}(s)$ in (37). It seems wise to use the pooled-within-groups covariance in those cases for which $S_i$ is the union of rather diverse point counts. An example should make the point. Suppose the array pair $(i, j)$ tracks the target at point counts $\{1, \cdots, 25\}$ and also $\{86, \cdots, 135\}$. Rather than apply (8) to produce $D_{ij}$ as an average cross-product deviation matrix of all the values measured from their center of gravity, it is better to produce two such values; one for the first point count set using its own center of gravity, and another from the second point count set using its center of gravity. For use in (36) these two need to be combined into one by using a weighted (by sample size) average. In our example, the weights would be 25/75 and 50/75.

For each fixed value of $r(t = 1, \cdots, K)$ the right-hand side of (36) contains an expression of the form

$$ W_r = \sum_{i<j} \text{Trace} \{ D_{ij}B_i^TB_j \} + \sum_{j<r} \text{Trace} \{ D_{ij}B_i^TB_r \} $$

$$ = \text{Trace} \left\{ \sum_{i<r} B_i D_{ir} B_i^T + \sum_{j>r} D_{ij} B_j^TB_j \right\}. \quad (38) $$

To shorten the writing let

$$ E_r = \sum_{i<r} B_i D_{ir} $$

and

$$ F_r = \sum_{j>r} D_{ij} B_j^T \quad (39) $$

Using $C = B_1^T$, say, then

$$ Q(B_1, \cdots, B_K) = Q(I, B_1^T B_2, \cdots, B_1^T B_K) \quad (33) $$

and for convenience, it is assumed that the orientation of the first array is known, i.e., $B_1 = I$ in (31).

Define covariance matrices

$$ D_{ij}(s) = \text{Ave} \{ X_i(t; s) X_j^T(t; s) \} \quad (34) $$

where the average is taken over the point counts $t$ in $S_i$. These quantities exist only for $s \in C_i \cap C_j \neq \emptyset$. Then (31) may be expanded to the more useful form

$$ Q = \sum_{i<j} \sum_s N_s \{ D_{ij}(s) - 2D_{ij}(s)B_i^TB_i + D_{ij}(s) \} $$

and again, because the unknown matrices $B_{ik}$ appear only in the second term, we may choose to maximize

$$ W = \sum_{i<j} \text{Trace} \{ D_{ij}B_i^TB_j \} \quad (36) $$

for $1 \leq i < j \leq K$ and

$$ D_{ij} = \sum_s N_s D_{ij}(s) \quad (37) $$

or zero, according to whether the summation for $s \in C_i \cap C_j$ has content, or is empty.

A word about the computation of $D_{ij}(s)$ in (37). It seems wise to use the pooled-within-groups covariance in those cases for which $S_i$ is the union of rather diverse point counts. An example should make the point. Suppose the array pair $(i, j)$ tracks the target at point counts $\{1, \cdots, 25\}$ and also $\{86, \cdots, 135\}$. Rather than apply (8) to produce $D_{ij}$ as an average cross-product deviation matrix of all the values measured from their center of gravity, it is better to produce two such values; one for the first point count set using its own center of gravity, and another from the second point count set using its center of gravity. For use in (36) these two need to be combined into one by using a weighted (by sample size) average. In our example, the weights would be 25/75 and 50/75.

For each fixed value of $r(t = 1, \cdots, K)$ the right-hand side of (36) contains an expression of the form

$$ W_r = \sum_{i<j} \text{Trace} \{ D_{ij}B_i^TB_j \} + \sum_{j>r} \text{Trace} \{ D_{ij}B_j^TB_i \} $$

$$ = \text{Trace} \left\{ \sum_{i<r} B_i D_{ir} B_i^T + \sum_{j>r} D_{ij} B_j^TB_j \right\}. \quad (38) $$

To shorten the writing let

$$ E_r = \sum_{i<r} B_i D_{ir} $$

and

$$ F_r = \sum_{j>r} D_{ij} B_j^T \quad (39) $$
and use the fact that \( \text{Trace} \{ F_i B_r \} = \text{Trace} \{ B_r^T F_i^T \} = \text{Trace} \{ F_i^T B_r^T \} \). Then

\[
W_r = \text{Trace} \{ (E_r + F_r^T) B_r^T \}. \tag{40}
\]

Also, it is interesting to note that each term of (36) appears in two, and only two, distinct \( W_r \). It follows that \( \Sigma W_r = 2W \).

The vector of partial derivatives of \( W \) with respect to the three Euler angles in \( B_r \), that is \( \phi_{r1}, \phi_{r2}, \phi_{r3} \), is the same as the gradient of \( W_r \) (with \( B_1, \cdots, B_{r-1}, B_r, B_{r+1}, \cdots, B_K \) held fixed). Moreover, the structure of (40) is the same as that of (20). We know from Section II that \( \nabla W ; r \) has eight zeroes, exactly one of which corresponds to a maximum.

The analysis above leads to the construction of a gradient search that can ferret out the maximum of \( W \). To fix one array we set \( B_1 = I \) and keep this throughout. Choose starting matrices for the set \( B_2, \cdots, B_K \). Compute \( E_r = E_r(B_1, \cdots, B_{r-1}) \) and \( F_r = F_r(B_{r-1}, \cdots, B_K) \) for each \( r = 1, 2, \cdots, K \) \( (E_1 = 0 = F_1) \). Use the algorithm in Section II applied to (40), for each \( r \), to produce the new Euler angles, i.e., the system \( \{ B_r \} \). Use these to recompute the \( \{ E_r, F_r \} \) and repeat. Stop when the gradients of all \( W_r \) are sufficiently small in magnitude. Each \( W_r \) will be at a local maximum for its argument \( B_r \), with all of the other orientation matrices held fixed.

Let us return to the objective function \( Q \) of (31) and use (29) in it for purposes of estimating the location parameters \( \{ A_r \} \). Formally, we have

\[
Q = \sum \sum N_s \text{Ave} \left\| A_i + B_i x_i(t; s) - A_j - B_j x_j(t; s) \right\|^2
\]

\[
(41)
\]

where \( 1 \leq i < j \leq K; s \in C_i \cap C_j \); the average is taken over the point counts \( t \) in \( S_i \); and \( N_s = \) number of points in \( S_i \). Again it is useful to use terms

\[
Q_r = \sum \sum N_s \text{Ave} \left\| A_i + B_i x_i(t; s) - A_j - B_j x_j(t; s) \right\|^2
\]

\[
+ \sum \sum N_s \text{Ave} \left\| A_j + B_j x_j(t; s) - A_i - B_i x_i(t; s) \right\|^2
\]

\[
(42)
\]

and recognize that \( \Sigma Q_r = 2Q \). In a manner similar to the one that produced (3) we can develop

\[
\frac{\partial Q_r}{\partial A_i} = -2 \sum \sum N_s \left\{ B_i x_i(s) - B_j x_j(s) - (A_r - A_i) \right\}
\]

\[
+ 2 \sum \sum N_s \left\{ B_i x_i(s) - B_j x_j(s) - (A_j - A_i) \right\}.
\]

Setting (43) equal to zero results in a \( 3K \) linear system in the \( A_1, \cdots, A_K \). Let \( M_{ij} = \sum_s N_s \) for \( s \in C_i \cap C_j \). Then the left-hand side of the linear system is

\[
- \sum_{i < r} M_{ip} A_i + \left\{ \sum_{i < r} M_{ip} + \sum_{j > r} M_{pj} \right\} A_r - \sum_{j > r} M_{jr} A_j \tag{44}
\]

and the right-hand side is

\[
\sum_{i < r} B_i \left\{ \sum_s N_i x_i(s) + \sum_{j > r} N_j x_j(s) \right\} - B_r \left\{ \sum_s N_r x_r(s) + \sum_{j > r} N_j x_j(s) \right\}.
\]

Notice that the coefficient matrix in (44) is the same for all three components of the \( \{ A_r \} \). Also, the columns of the matrix add to zero and its rank is \( K - 1 \). It follows that the system is underdetermined. We anticipated this when we designated the first array as a reference. Set \( A_1 = 0 \); a unique solution for \( A_2, \cdots, A_K \) is obtained after the reduced system (removal of first row and column of \( M \); removal of first row of the right-hand side) is solved.

Each particular vector \( A \) is related to an estimated location correction for its array in the range coordinate system. Referring to (3), we express the corrected track in terms of the original local track as

\[
\hat{x}(t) = A + B \alpha + B \beta \xi(t)
\]

where \( \alpha \) is the assumed location of the array in range coordinates. The estimated displacement vector of this location is

\[
D = A + B \alpha - \alpha = A + (B - I) \alpha.
\]

For a simple example of the output of the algorithm, we offer the result of \( R = 2 \) crossover data sets and \( K = 3 \) arrays. The number of point counts in the two sets are 45 and 20, respectively, and the three assumed array locations form the vertices of an equilateral triangle in the plan view of Fig. 2. The raw data were not smoothed and consequently the noise component has been propagated through the algorithm. Table I contains the estimated array displacement vectors (feet) and Euler angle corrections (degrees) for the indicated arrays. The down range is \( x \), cross range is \( y \), and vertical is \( z \). As of this writing, we do not know what part of these estimated corrections has permanence (if any) and what part represents noise and transient effects.

As stated earlier, the range monitoring analyst is concerned with the issue of examining the estimated corrections successively as each of the arrays is viewed as the base array (i.e., the array not requiring correction). If the orientation corrections were all zero, then this goal could be achieved by differencing the estimated displacement vectors. In general, these angles are not zero; the estimates are rather sensitive to small changes in orientation, and one must work a bit harder.

Conceptually, one need only repeat the computation \( K - 1 \) times, changing the order of data input appropriately each time. However, this approach is rather intensive computationally. The iterative gradient search to compute the \( B_2, \cdots, B_K \) consumes computer resources and there can be great variability in the number of cycles to convergence. Also it is unnecessary to use this part of the algorithm more than once because we can exploit the discussion surrounding (33) and (36). Suppose we want the orientation correction matrices relative to the \( q \)th array, rather than the first. It is easily seen
that one need only make the replacements
\[ \bar{B}_j = B_q^T B_j, \quad \text{for } j = 1, \ldots, K. \] (47)

For the estimates of the vectors \( A_1, \ldots, A_k \), we must solve the linear systems each time. Let us use the notation
\[ MA = R^T \] (48)
for the original system described by (44) and (45); \( R \) being the \( 3 \times K \) matrix of (45). Use of (47) entails the replacement
\[ R \rightarrow B_q R \] (49)
in (48). Since \( A_q \) must be zero, the solution of this new singular system is performed by removing the \( q \)th row and column of \( M \), the \( q \)th element of \( A \), and the \( q \)th row of the right-hand side from (48). The solution will be unique and represent \( A_1, A_2, A_{q-1}, A_{q+1}, \ldots, A_k \).

V. INFORMATION SUMMARY TECHNIQUES

Having developed the capability of producing \( K \) versions of Table I, one for each choice of the base array, it is wise to go a step further and present the analyst with simplified summaries of this information. Six numbers can be replaced by two. First, the three components of displacement, (46), can be replaced by their magnitude:
\[ MD = \{ || D ||^2 \}^{1/2}. \] (50)

Second, the three Euler angles can be replaced by one maximum angle of rotation. An eigenvalue problem needs to be solved in order to do it. The orientation correction may be viewed as a rigid turn in three space, i.e., the turning of the entire space through an angle of \( \theta \) about an axis \( x \). This axis is a fixed point, i.e.,
\[ Bx = x \] (51)
and is identified as the eigenvector of \( B \) having eigenvalue one. In order to make it unique, let us agree that \( x \) is normalized so that \( || x ||^2 = 1 \) and it lies in the half-space defined by a nonnegative first component, \( x_1 \geq 0 \). Note there are two degrees of freedom in this \( x \).

To determine the rotational angle \( \theta \), let us set up a right-handed coordinate system with \( x \) as its first axis, and then determine the angle of rotation resulting from applying \( B \) to any vector orthogonal to \( x \). Our goal is to construct a change of basis matrix \( C \) which has \( x \) as its first row. It remains to specify the second and third rows \( y \) and \( z \). Since \( x \) is pointed in the half-space \( x_1 > 0 \), let us apply the Gram–Schmidt process to \( e_2^T = (0, 1, 0) \) in order to specify \( y \); i.e., \( y = e_2 - (x_1, e_2) x \)

Table I

<table>
<thead>
<tr>
<th>Array</th>
<th>( x )</th>
<th>( y )</th>
<th>( z )</th>
<th>( \phi_1 )</th>
<th>( \phi_2 )</th>
<th>( \phi_3 )</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
<td>0.0</td>
</tr>
<tr>
<td>2</td>
<td>-8.4</td>
<td>-13.6</td>
<td>-23.1</td>
<td>-0.09</td>
<td>-0.31</td>
<td>-0.09</td>
</tr>
<tr>
<td>3</td>
<td>21.2</td>
<td>-5.0</td>
<td>9.5</td>
<td>0.13</td>
<td>-0.02</td>
<td>-0.21</td>
</tr>
</tbody>
</table>

That figures is necessarily symmetric.

To sum up, to each array other than the base array there is estimated a magnitude of displacement and rotation. These computations, using the data that were used to produce Table I, are presented in Table II. The summary in the \( i \)th row provides the magnitude of displacement and rotation corrections for each array when the \( i \)th array is viewed as the base array. Again these values are in feet and degrees. The matrix of rotation figures is necessarily symmetric.

Table II

\begin{array}{ccc|ccc}
\hline
\text{Displacement} & \text{Rotation} \\
\hline
\text{Base} & 1 & 2 & 3 & 1 & 2 & 3 \\
\hline
1 & 0.0 & 28.1 & 23.8 & 0.0 & 0.33 & 0.25 \\
2 & 23.1 & 0.0 & 33.5 & 0.33 & 0.0 & 0.38 \\
3 & 19.9 & 49.2 & 0.0 & 0.25 & 0.38 & 0.0 \\
\hline
\end{array}

which allows the determination of \( \theta \). Thus \( x \) and \( \theta \) can be constructed from the orthonormal matrix \( B \).

To sum up, to each array other than the base array there is estimated a magnitude of displacement and rotation. These computations, using the data that were used to produce Table I, are presented in Table II. The summary in the \( i \)th row provides the magnitude of displacement and rotation corrections for each array when the \( i \)th array is viewed as the base array. Again these values are in feet and degrees. The matrix of rotation figures is necessarily symmetric.

Summaries of the type illustrated in Table II can be very useful to the analyst. Let us discuss this under the ideal conditions that the data are noise free and there are no systematic errors in processing or any transient effects. If all arrays in the problem are properly calibrated, then all entries in Table II are zero. If exactly one array is out of kilter, it will be easily identified. Using the miscreant array as the base array will leave the impression that all other arrays need more
correction. Using any other array as the base array will provide zero corrections for all arrays save the miscreant one.

If two of the three arrays are miscreant not only relative to the good one but relative to each other, then it is unlikely that we will be able to identify the good one.

To generalize to $K (>3)$ arrays, the appropriate counterpart to Table II will be useful in identifying the arrays that are out of calibration provided the number of such arrays is not large compared to $K$, and provided the other aspects of the range are in good working order. The daily collection of information of this type will serve as a monitoring system to watch calibration and perhaps other performance characteristics of the range.

REFERENCES


Robert R. Read was born in Columbus, OH, in 1929. He received the B.S. degree in mathematics from Ohio State University, Columbus, in 1951 and the Ph.D. in mathematical statistics from the University of California at Berkeley in 1957. His dissertation research was under the direction of Professor Jerzy Neyman in the area of probabilistic model building of cloud chamber drop count data.

For the following three years he researched civil defense and airport runway design problems at the University of California's Institute of Engineering Research and the Institute of Transportation and Traffic Engineering. For the academic year 1960-1961, he served as a Visiting Assistant Professor in the Statistics Department at the University of Chicago, Chicago, IL. In September 1961, he joined the Naval Postgraduate School as an Associate Professor in the Department of Mathematics and from 1971 to the present has been Professor of Probability, Statistics, and Operations Research. His research, both scholarly and applied, has been in the areas of probability, statistical estimation, and data analysis.