PARALLEL COORDINATES : *VISUAL* Multidimensional Geometry and its Applications

Alfred Inselberg(©1992, 2004)

Senior Fellow San Diego SuperComputing Center, CA, USA

School of Mathematical Sciences
Tel Aviv University , Israel
aiisreal@post.tau.ac.il
&
Multidimensional Graphs Ltd
Raanana 43556, Israel

Chapter 1

Multidimensional Lines

1.1 Representing Lines in $\mathbb{R}^N$

$\mathcal{T}$here are several *point* $\leftrightarrow$ *line* dualities (like the very useful Hough transform) on the plane which do not generalize well for higher dimensions. This is, of course, because in the Projective N-space $\mathbb{P}^N$ the natural duality is *Point* $\leftrightarrow$ *Hyperplane* with the *Point* $\leftrightarrow$ *Line* being the special case for $N = 2$. However, in $\|$-coords there is a useful and direct generalization which is the subject of this chapter. At first the basic idea for lines in $\mathbb{R}^N$, rather than $\mathbb{P}^N$ is presented in an intuitive way paving the way for the general case which is treated subsequently (as in [8] and [9]).

1.1.1 Elementary Approach

Adjacent Variables Form

What is meant by "a line in $\mathbb{R}^N$"? In $\mathbb{R}^3$ a line is the intersection of two planes. So a line ℓ in $\mathbb{R}^N$ is the intersection of $N-1$ non-parallel hyperplanes. Equivalently, it is the set of points (specified by N-tuples) which satisfy a set of $N-1$ linearly independent linear equations. After some manipulation, and with the exception of some special cases to be treated later, the set of equations can be put in the convenient form :

$$\ell \quad : \quad \begin{cases} \ell_{1,2} & : & x_2 & = & m_2 x_1 \ + \ b_2 \\ \ell_{2,3} & : & x_3 & = & m_3 x_2 \ + \ b_3 \\ \quad \cdots & & & & \\ \ell_{i-1,i} & : & x_i & = & m_i x_{i-1} \ + \ b_i \\ \quad \cdots & & & & \\ \ell_{N-1,N} & : & x_N & = & m_N x_{N-1} \ + \ b_N \quad , \end{cases} \tag{1.1}$$

that is each equation contains a pair of *adjacently* indexed variables. In the $x_{i-1}x_i$-plane the relation labeled $\ell_{i-1,i}$ is a line, and by our *point* $\leftrightarrow$ *line* duality which we have already found (eq. (3) in Chapter 1) it can be represented by a point $\bar{\ell}_{i-1,i}$. To clarify matters the situation is illustrated with an example in $\mathbb{R}^5$ shown in Fig. 1.1 with a polygonal line $\bar{A}$, for $A = (a_1, a_2, a_3, a_4, a_5)$, on each one the four points $\bar{\ell}_{12}, \bar{\ell}_{23}, \bar{\ell}_{34}, \bar{\ell}_{45}$ representing the 4 linear relations in eq. (1.1) for a line ℓ when $N = 5$. The $\bar{X}_1\ \bar{X}_2$ portion of $\bar{A}$ on $\bar{\ell}_{12} \Rightarrow$ that (a_1, a_2)

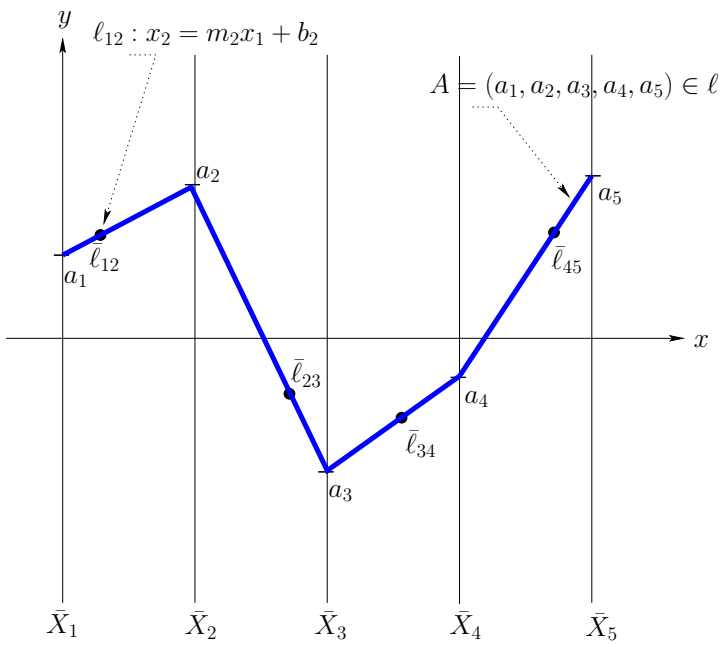

Figure 1.1: Point on line in 5-D.

satisfies the ℓ_{12} relation as for the duality in Chapter **??**. Similarly $(a_2, a_3), (a_3, a_4), (a_4, a_5)$ satisfy the $\ell_{23}, \ell_{34}, \ell_{45}$ relations respectively. Hence $A \in \ell$ when $\bar{A}$ is on all the $\bar{\ell}$ points.

Unless otherwise stated from now on the distance between each pair of adjacent axes is taken as one unit as shown in Fig. 1.2. Since the y-axis is not coincident with the $\bar{X}_{i-1}$-axis, the x-coord of the point representing the line (see eq. (**??**) and Fig. **??** in Chapter **??**) needs to be translated by $(i-2)$. That is

$$\bar{\ell}_{i-1,i} = (\frac{1}{(1-m_i)} + (i-2)\ , \frac{b_i}{(1-m_i)})$$

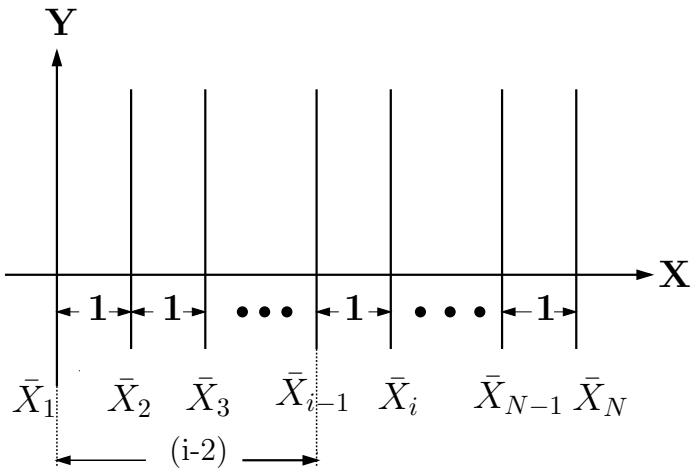

Figure 1.2: Spacing between adjacent axes is 1 unit.

or in homogeneous coordinates :

$$\bar{\ell}_{i-1,i} = ((i-2)(1-m_i)+1\,,\, b_i\,,\, 1-m_i). \tag{1.2}$$

There are $N-1$ such points for $i = 2\,,\ldots\,,N$ which represent the line ℓ. This is the first instance where the indexing arises in the representation and plays a key role in the subsequent development. For now it suffices to observe that without the indexing the representation of the line is ambiguous. The exact number of lines that can be represented by the same points indexed differently is studied later in section 1.1.4. Since the display space is at a premium, the indexing is usually not included in the picture but it must always be accessible.

We know that a polygonal line whose portions between the $\bar{X}_{i-1}$ and $\bar{X}_i$ axes (extended if necessary) are on the points $\bar{\ell}_{i-1,i}\ \forall\ i = 2\,,\ldots\,,N$, necessarily represents a point on the line ℓ since adjacent pairs of y-coordinates of vertices on the $\bar{X}_{i-1}$ and $\bar{X}_i$-axes simultaneously satisfy Equation (1.1). Such is the case in Fig. 1.3 where several polygonal lines representing points on an interval of a line in 10-D are shown. For example, the point of intersection of the polygonal lines between the $\bar{X}_2$ and $\bar{X}_3$-axes, is the point $\bar{\ell}_{2,3}$. All the nine points, representing that 10-dimensional line, can be seen (or constructed) with their horizontal positions depending on the first coordinate of eq. (1.2). Do not be mislead by the fact that all of the $\bar{\ell}$'s except $\bar{\ell}_{1,2}$ and $\bar{\ell}_{6,7}$ are in-between their corresponding axes. This is due to the choice of $m_i \leq 0$ made for display convenience only. As in the x-coord of $\bar{\ell}$ in the $\ell \leftrightarrow \bar{\ell}$ mapping of Ch. 1, and here due to the translation of x by $(i-2)$, the point $\bar{\ell}_{i-1,i}$ is in-between the $\bar{X}_{i-1}\,,\bar{X}_i$ axes when $m_i < 0$, to the left of $\bar{X}_{i-1}$ when $m_i > 1$ and to the right of $\bar{X}_i$ when $0 < m_i < 1$.

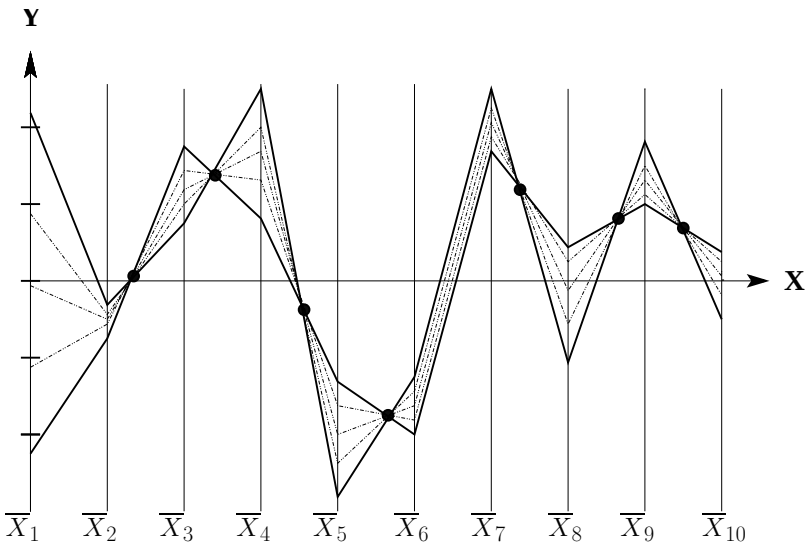

Figure 1.3: Line interval in 10-D – the thicker polygonal lines represent it's end-points. The adjacent variables representation, consisting of nine properly indexed points, is obtained by the sequential intersections of the polygonal lines' linear portions. Note that $\bar{\ell}_{1,2}$ is to the right of the X_2-axis and $\bar{\ell}_{6,7}$ is an ideal point. The remaining points are in between the corresponding pairs of axes.

Base Variable Form

Another common way of describing a line $\ell \subset \mathbb{R}^N$ is in terms of one, sometimes called the *base*, variable which after appropriate relabeling may be taken as x_1. Then

$$\ell \quad : \quad \left\{ \begin{array}{lcccl} \ell_{1,2} & : & x_2 & = & m_2^1 x_1 \; + \; b_2^1 \\ \ell_{1,3} & : & x_3 & = & m_3^1 x_1 \; + \; b_3^1 \\ & \cdots & & & \\ \ell_{1,i} & : & x_i & = & m_i^1 x_1 \; + \; b_i^1 \\ & \cdots & & & \\ \ell_{1,N} & : & x_N & = & m_N^1 x_1 \; + \; b_N^1 \end{array} \right. , \tag{1.3}$$

and the $N-1$ points representing it are :

$$\bar{\ell}_{1,i} \; = \; (i-1, \; b_i^1, \; 1-m_i^1) \; , \tag{1.4}$$

With the axes placed 1 unit apart, the distance between $\bar{X}_i$ and $\bar{X}_1$ (which is coincident with the y-axis) is $i-1$. This brings the point $\bar{\ell}_{1,i}$ in-between the axes $\bar{X}_1, \bar{X}_i$, when $m_i^1 < 0$, to the left of $\bar{X}_1$ when $m_i^1 > 1$ and to the right of $\bar{X}_i$ when $0 < m_i^1 < 1$. All the remarks about indexing pertain equally well for this or any other representation of ℓ . What about changing the parametrization? Let us consider an example where a line in $\ell \subset \mathbb{R}^5$ is specified by four indexed points of the form $\bar{\ell}_{i-1,i}$ for which it is required to find the linear relation between x_2 and x_5,

$$\ell_{2,5} : x_5 = m_{25} x_2 + b_{25} \; . \tag{1.5}$$

There is no problem returning to the system of four linear equations $\ell_{i-1,i} \; , i = 2, \ldots, 5$ specifying the line ℓ, from which the points $\bar{\ell}_{i-1,i}$ were obtained, and after doing the necessary algebraic manipulations obtain eq. 1.5. But we can do this equivalently by just following

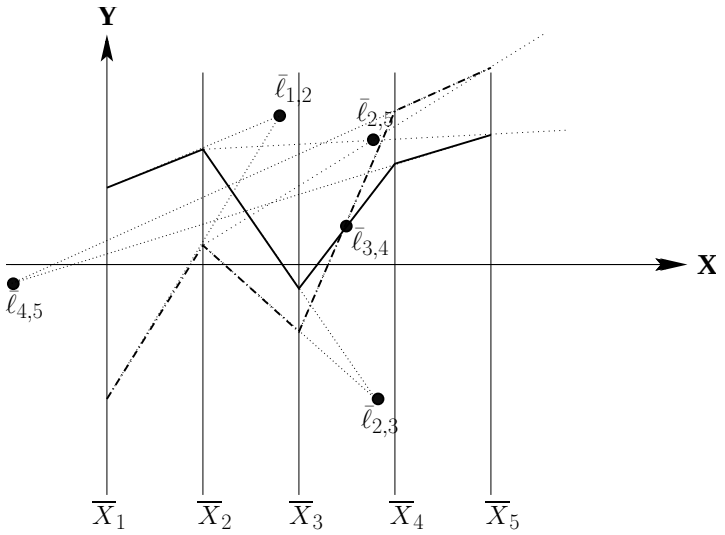

Figure 1.4: Algorithm for constructing a pairwise linear relation, in this case $\bar{\ell}_{25}$, given the $N-1$ points, $\bar{\ell}_{i-1,i}$, representing the line.

what is in the picture. By assigning any two different values to x_1 the polygonal lines representing two points on ℓ are constructed. Joining the two pairs of x_2 and x_5 values provides two lines whose intersection must, due to the line $\rightarrow$ point mapping in 2-D, be the point $\bar{\ell}_{2,5}$ whose coordinates are

$$\bar{\ell}_{25} = (\frac{3}{1-m_{25}} + 1\ ,\ \frac{b_{25}}{1-m_{25}})\ . \tag{1.6}$$

yielding the values of m_{25} and b_{25}. By the way, the 3 is for the distance between the $\bar{X}_2$ and $\bar{X}_5$ axes and the translation by 1 is for the distance between the $\bar{X}_2$ and y-axis. Incidentally, we see that all the information about a line is really contained in its $N-1$ indexed points. For the remainder, all sections marked by ** can be safely omitted at the first reading.

Two Point Form **

The representation of a line ℓ independent of parametrization can best be constructed from two of its points. Starting with

$$P_r\ :\ (p_{1,r}\ ,\ p_{2,r}\ ,...,\ p_{N,r}) \quad r = 1, 2$$

for any other point $P \in \ell$, $P = \lambda P_1 + (1-\lambda)P_2\ , \lambda \in R$. When the two points are both ideal they define an ideal line. Otherwise, if one of the points is ideal it is a simple matter to replace it by a second ordinary point distinct from the first and still obtain the same line ℓ. So without loss of generality only lines defined by two ordinary points need to be dealt with.

A pair of variables $x_i\ ,\ x_j$

$$(p_{i,r}, p_{j,r}) \quad i, j \in 1, 2, ..., N \quad r \in 1, 2\ . \tag{1.7}$$

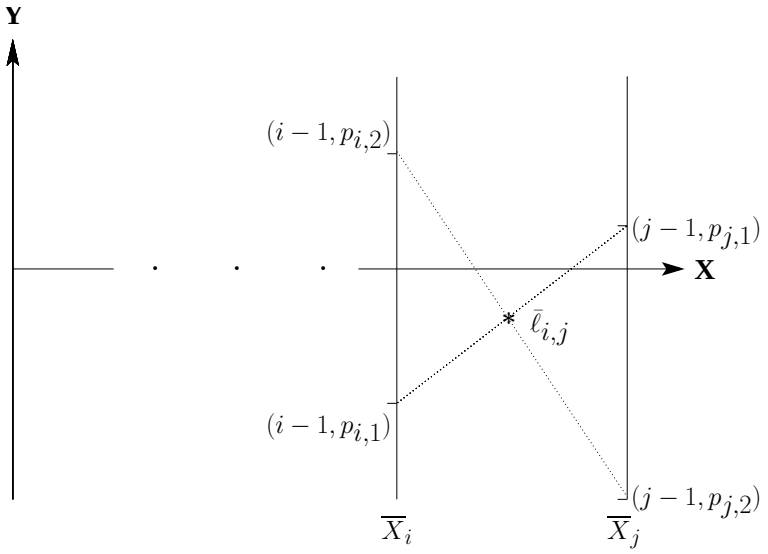

Figure 1.5: Construction of $\bar{\ell}_{i,j}$

specify the component of ℓ which corresponds to its projection on the $x_i x_j$ coordinate plane. As in the 2-D case a point, $\bar{\ell}_{i,j}$, represents this component though it is now essential to identify it with the proper indices – see Fig. 1.5. Given the data in (1.7) this component is

$$\Delta_j x_i - \Delta_i x_j + D_{i,j} = 0 \tag{1.8}$$

which is a line $\ell_{i,j}$ whose line coordinates are $[\Delta_j, -\Delta_i, D_{i,j}]$ where

$$\left\{ \begin{array}{lcll} \Delta_i & = & p_{i,2} - p_{i,1} & , \\ \Delta_j & = & p_{j,2} - p_{j,1} & , \\ D_{i,j} & = & p_{j,2}\Delta_i - p_{i,2}\Delta_j & = \quad p_{j,1}\Delta_i - p_{i,1}\Delta_j \end{array} \right. \tag{1.9}$$

for all $i, j \leq N$. When Δ_i and Δ_j are not both zero, eq. (1.8) actually defines a line which is represented by the point

$$\bar{\ell}_{i,j} : ((i-1)\Delta_j - (j-1)\Delta_i \ , \ -D_{i,j} \ , \ \Delta_j - \Delta_i) \tag{1.10}$$

where, as usual, the distance between adjacent coordinate axes is 1. The relevant transformations here are

$$\bar{\ell}_{i,j} = A_{i,j}\ell_{i,j} \quad , \quad \ell_{i,j} = A_{i,j}^{-1}\bar{\ell}_{i,j} \tag{1.11}$$

where

$$A_{ij} = \begin{pmatrix} (i-1) & (j-1) & 0 \\ 0 & 0 & -1 \\ 1 & 1 & 0 \end{pmatrix}$$

and

$$(i-j)A_{ij}^{-1} = \begin{pmatrix} 1 & 0 & (1-j) \\ -1 & 0 & (i-1) \\ 0 & (j-1) & 0 \end{pmatrix} \ .$$

There are still some important considerations missing which are completed in section 1.1.4.

1.1.2 Some properties of the indexed points

The 3-point collinearity

The representation of a line in terms of $N-1$ points can be given for *any* of the various descriptions of the line by linear equations each involving a pair of variables. Actually, once a sufficient representation of a line in terms of $N-1$ points is given any other point $\bar{\ell}_{i,j}$ can be obtained from the line representation by a geometrical construction. An example is shown in Fig. 1.4 where starting from the adjacent-variable description of a line in $\mathbb{R}^5$ the point $\bar{\ell}_{2,5}$ is constructed. It is important then to clarify that by the **representation** $\bar{S}$ of an object S is meant the **minimal** subset of the xy-plane from which the representative $\bar{P}$ of any point $P \in S$ can be constructed. For a line $\ell \subset \mathbb{P}^N$ we have seen that $N(N-1)$ indexed points in the xy-plane, corresponding to all distinct pairs of the N variables, can be constructed. However only $N-1$ of these, corresponding to $N-1$ linearly independent equations in pairs of these variables, are needed to **represent** the line ℓ.

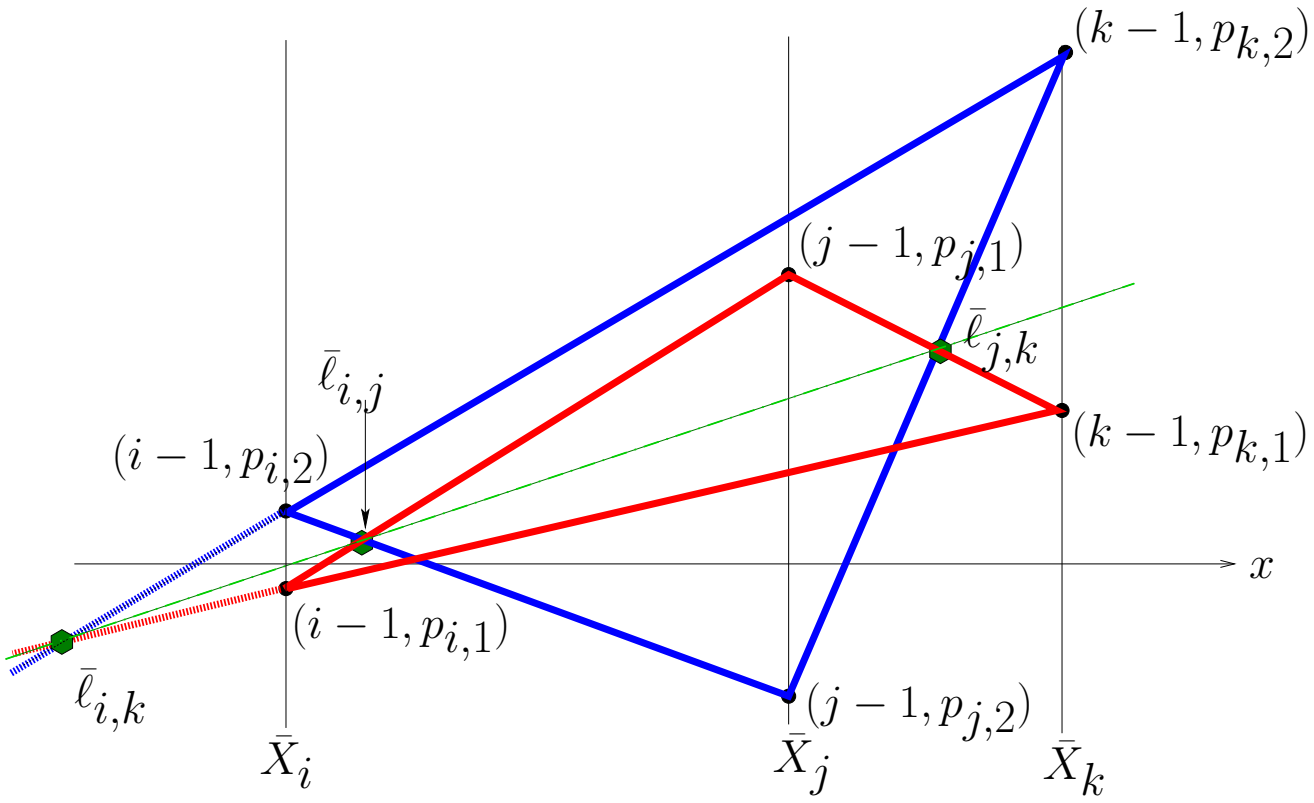

Figure 1.6: The collinearity for the 3 points $\bar{\ell}_{i,j}, \bar{\ell}_{j,k}, \bar{\ell}_{i,k}$. The two triangles are in perspective with respect to the ideal point in the vertical direction. The y-axis is offscale.

The indexed points representing the pairwise linear relation of variables in an N-dimensional line have a striking and very useful property. For $i \neq j \neq k \in [1, 2, ..., N]$ the three points $\bar{\ell}_{i,j}, \bar{\ell}_{j,k}, \bar{\ell}_{i,k}$ are always collinear. This can be seen by considering again two points $P_r = (p_{1,r}, \ldots, p_{N,r})$, $r = 1, 2$ on ℓ and their projections on the x_i, x_j, x_k three-space as shown in Fig. 1.6. The projected portions of the points are the vertices of two triangles with the collinearity of $\bar{\ell}_{i,j}, \bar{\ell}_{j,k}, \bar{\ell}_{i,k}$ being a consequence of Desargue's Theorem[1]. Here the two triangles are in perspective with respect to the ideal point in the vertical direction. This property, as will be seen, is the "back-bone" of the construction for the representation of higher dimensional p-flats in N-space (that is planes of dimension $2 \leq p \leq N-1$) and it is referred to as the *the 3-point collinearity property*. An important special case is for a line ℓ in 3-D where the 3 points $\bar{\ell}_{1,2}$, $\bar{\ell}_{1,3}$, $\bar{\ell}_{2,3}$ are always collinear. In turn, this provides an important corollary. Consider the plane

$$\pi_{123}: \; c_1 x_1 + c_2 x_2 + c_3 x_3 = c_0 \; ,$$

choose a line $\ell \subset \pi$, and denote by $\bar{L}$ the line determined by $\bar{\ell}_{1,2}$, $\bar{\ell}_{2,3}$ and $\bar{\ell}_{1,3}$. It is found by direct computation that for any other line $\bar{\ell}'$, and $\bar{L}'$ the corresponding line through

[1]Two triangles in perspective from a point are in perspective from a line.

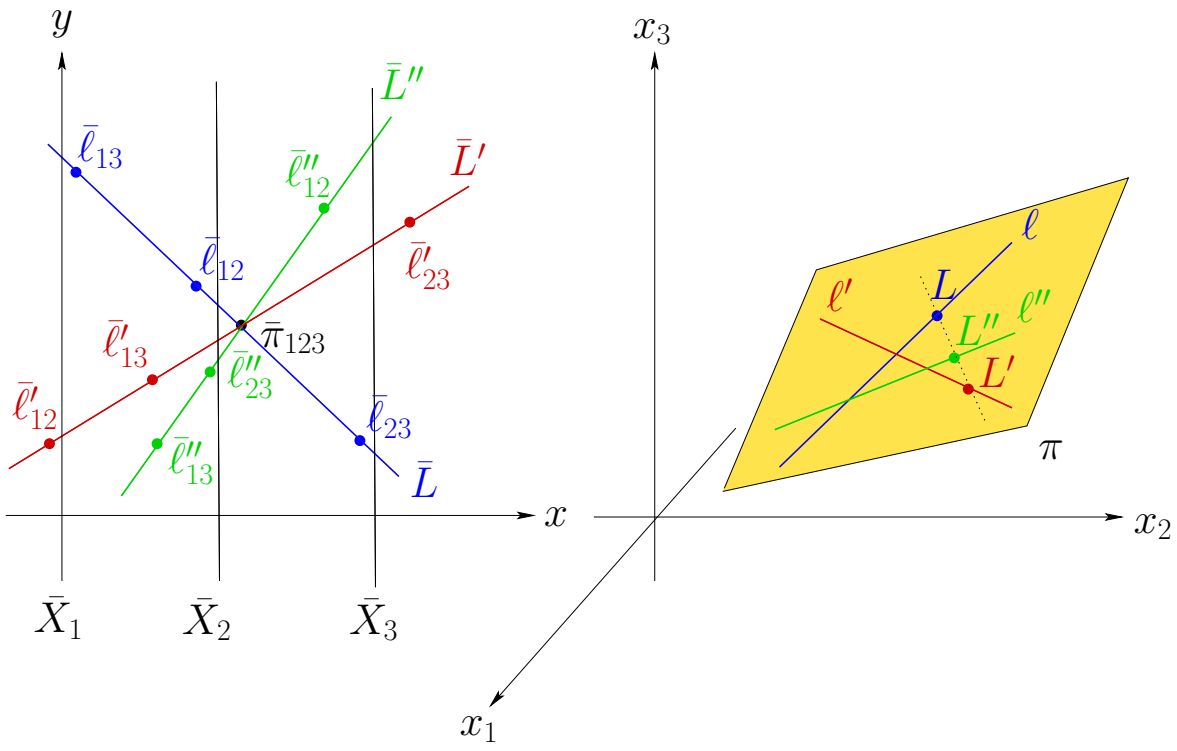

Figure 1.7: Three lines $\ell, \ell', \ell'' \subset \pi \subset \mathbb{R}^3$ are chosen. In $\|$-coords the 3pt-collinearity property applied to the three points $\bar{\ell}_{12}, \bar{\ell}_{13}, \bar{\ell}_{23}$ determines a line $\bar{L}$ and similarly ℓ' determines the line $\bar{L}'$. Denoting the intersection point by $\bar{\pi}_{123} = \bar{L} \cap \bar{L}'$ the line $\bar{L}''$, determined by **any** other line ℓ'', is **on** the point $\bar{\pi}_{123}$. The picture hints that the points L, L', L'' are on a line in the plane π. This is a prelude to the chapter on planes.

$\bar{\ell}'_{1,2}$, $\bar{\ell}'_{2,3}$, $\bar{\ell}'_{1,3}$, the intersection $\bar{\pi}_{123} = \bar{L} \cap \bar{L}'$ has coordinates:

$$\bar{\pi}_{123} = (c_2 + 2c_3\,, c_o, c_1 + c_2 + c_3)\,. \tag{1.12}$$

Since these coordinates do not depend on the choice of ℓ and ℓ', for any other line $\ell' \subset \pi$ the corresponding $\bar{L}''$ must also pass through $\bar{\pi}_{123}$ as shown in Fig. 1.7. In short, the image of every line on the plane π determines a line on the point $\bar{\pi}_{123}$, hence two of these lines suffice to determine the point. This is a preview of the next chapter where it is shown that two such points each with **three indices**, provide a representation of a plane $\pi \subset \mathbb{R}^3$.

Exercises

1. Consider a line $\ell \subset \mathbb{R}^N$ whose description, by a set of linear equations, contains $x_i = c_i$, a constant, for some i.

   (a) How would such a line be represented?

   (b) Can such a line be represented by $N-1$ indexed points?

2. Prove the property illustrated in Fig. 1.7

3. Given a line ℓ and the $N-1$ points $\bar{\ell}_{i-1,i}$, $i = 2, \ldots, N$, provide an an algorithm for determining the point $\bar{\ell}_{ij}$, for any distinct pair i, j and state its complexity – see Fig. 1.4.

4. For lines where some of the m's are close to 1 their representation becomes problematical. Find a transformation that can always convert points outside the axes to equivalent points between the axes. Write an algorithm to perform this transformation so that all points representing a line are in between the axes. What is the algorithm's complexity.

1.1.3 Representation Mapping I

The 3-point collinearity property and its higher-dimensional relatives are used in the next chapter for the representation of p-flats (i.e. "planes" of dimension $0 \leq p \leq N-1$) in $\mathbb{R}^N$ by means of indexed points. As explained shortly, the 0 lower bound pertains to the representation of points $P \in \mathbb{P}^N$ as shown in Fig. 1.8.

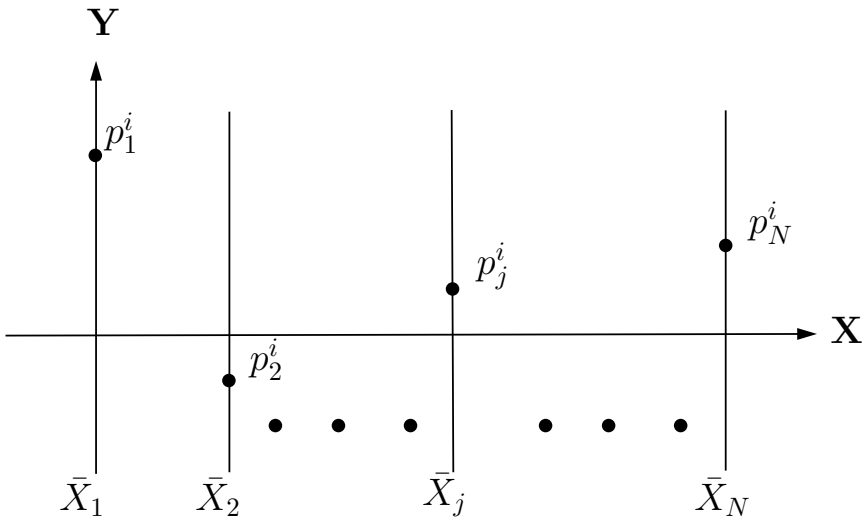

Figure 1.8: A point $P^i = (p_1^i, p_2^i, ..., p_j^i, ..., p_N^i)$ represented by N points p_j^i with one index.

The methodology's goal is the construction of a mapping $\mathcal{J}$ for the *unique representation* of objects (i.e. subsets) of $\mathbb{P}^N$ by subsets of $\mathbb{P}^2$; an object $B \subset \mathbb{P}^N$ is *represented* by its image $\mathcal{J}(B) = \bar{B}$. From the starting with the representation of a point $P \in \mathbb{P}^N$, it has been clear that representation **is not a point-to-point mapping.** Formally,

$$\mathcal{J} : 2^{P^N} \rightarrow 2^{P^2} \times 2^{[1,2,...,N]} , \tag{1.13}$$

where $2^A = [B \subseteq A]$ denotes the *power set* of a set A and $\mathbb{R}^N$ is embedded in the projective N-space $\mathbb{P}^N$. The $2^{[1,2,...,N]}$ in the product pertains to the indexing by subsets of $[1, 2, \ldots, N]$.

The answer to "how can $\mathbb{R}^N$ be mapped into the plane $\mathbb{R}^2$ without loss of information?" is found by examining the *cardinality*[2] of the domain and range of $\mathcal{J}$. Let $|A|$ denote the

[2]Only elementary considerations are needed. For example, the sets of rational and real numbers $\mathbb{R}$ have cardinality $\aleph_0$ and $\aleph_1$ respectively. The comparison $\aleph_0 < \aleph_1$ signifies the *non-existence* of a 1-1 and onto mapping from the rationals to the reals. For more on the fascinating topic of infinite sets and cardinal numbers see any good text on Topology.

cardinality of a set A and recall that by taking the finite product of a set with itself the cardinality remains invariant. For our specific case,

$$|\mathbb{R}| = |\mathbb{R}^2| = |\mathbb{R}^N| = \aleph_1 \quad \Rightarrow \quad |2^{\mathbb{R}^2}| = |2^{\mathbb{R}^N}| = \aleph_2 \ ,$$

With the domain and range having the same cardinality the construction of a 1-1 mapping $\mathcal{J}$ is possible; what is left is to actually do it. "The devil is in the details" ... as sages have said. The image $\mathcal{J}(B)$, of a set $B \subset \mathbb{P}^N$, is given in terms of the xy Cartesian coordinates superimposed on the parallel axes with $\mathcal{J}(B) = \mathcal{J}(C) \ \Leftrightarrow \ B = C$. At this stage we can get a glimpse of the mapping's *recursive construction algorithm*, the recursion being on the dimensionality of the object being represented.

1. **Non-recursive step** is the alternate representation of a point

   $$P^i = (p^i_1, p^i_2, \ , \ldots \ , p^i_j \ , \ldots \ , p^i_N).$$

   Rather than using a polygonal line for $\bar{P}^i$, for consistency with what follows, the representation is also given in terms of indexed points on the xy-plane. Specifically, $P^i \in P^N$ is mapped into (or represented by) N points with *one index* as shown in Fig. 1.8. Of course, this is completely equivalent to its polygonal line representation, where the indexing sequence provides the straight-line connectivity when needed.

   If N_r denotes the number of points and n_r the number of indices appearing in the representation, for a (point) *0-flat*

   $$N_r + n_r = N + 1 \ .$$

2. **First recursive step :** is the construction of a *1-flat* (line) $\ell \subset \mathbb{P}^N$ consisting of $N-1$ points with two indices. The steps are :

   (a) for each of two distinct *0-flats* $P_1, P_2 \subset \ell$ the N points of their representation are connected, via the indexing sequence, to provide two polygonal lines,

   (b) intersecting adjacent portions of the polygonal lines yields the $N-1$ points, $\bar{\ell}_{i-1,i} \ , i = 2, \ldots, N$.

   Alternate but equivalent representations correspond to the intersection of $N-1$ pairs of lines joining the appropriate alternate vertices of the polygonal lines.

   Checking the previous sum for *1-flats*,

   $$N_r + n_r = (N-1) + 2 = N + 1 \ .$$

3. The construction procedure turns out to be generic for the representation of p-flats of all dimensions.

The emerging relation for $N_r + n_r$ holds with the caveat that the subset of points in the representation is minimal. It is revisited as new representations are obtained.

1.1.4 The General Case **

Indexed points for arbitrary parametrization

$\mathcal{T}$he approach so far clarifies the basic representational issues. However, the results in section 1.1.1 do not cover all the possibilities. There remain complications arising due to the orientation of some lines with respect to the principal axes. Specifically, a line ℓ may be perpendicular to a principal 2-plane x_ix_j. Then both x_i and x_j are constant for ℓ so the projection of $\ell_{i,j}$ on the x_ix_j-plane is not a line but a point (p_i, p_j). It's representative in parallel coordinates, therefore, is not a point but a line through the two points $(i-1, p_i)$, $(j-1, p_j)$ and this line has the role of $\bar{\ell}_{i,j}$. These contingencies, though conceptually straight-forward, require that the foundations of the representational issues be revisited in a thorough way.

In $\mathbb{R}^N$ there are $N(N-1)$ different linear equations connecting the variables pairwise and, therefore, as N increases the number of sets of equations available for describing a line increases rapidly. A precise count as a function of N is given shortly. The reasons for choosing one description rather than another are irrelevant here. Rather, the *class* of such descriptions is of interest for which purpose a graph-theoretic approach is appropriate. Let the N variables be represented by N vertices of a graph. An edge connecting a pair of vertices indicates that a linear relation is given between the corresponding pair of variables. A set of equations describing a line ℓ corresponds to a subgraph with N vertices and N - 1 edges which must be a *spanning tree* of the complete graph of N vertices. For if the subgraph is not connected the corresponding system of equations contains at least two independent sets of variables such that their values are independent of values taken on by the variables of other sets. Also, if there is a closed loop in the graph then at least one vertex is disconnected, there being only $N-1$ edges. Once a particular variable is selected and designated as the *root* of the tree the number of different ways in which N vertices and N-1 edges can be assembled into a spanning tree is $N^{(N-1)}$ (Cayley's formula) [13]. In [3] there is a nice proof[3] that the number of spanning trees with N distinct vertices is $N^{(N-2)}$. From this result Caley's formula can immediately obtained by choosing any one of the N vertices as the root. From our viewpoint this is the number of distinct $N-1$ linearly independent equations, with the chosen variable as the root, can be selected out of the possible $N(N-1)$ pairwise linear equations. Assigning a value to the *start* variable, corresponding to the root, *initiates* the computation and suffices to determine the values of all the remaining variables for the specification of a point on the line.

Since variables which are constant are not involved in linear relations with another variable there is no edge connecting their vertex to another vertex. That is to say, such vertices are isolated. If there are $N-M$ such variables, then the remaining M vertices together with $M-1$ edges again form a spanning tree as before, *rooted*, once a start variable is chosen. Of course in this case there are $M^{(M-1)}$ ways for selecting the $M-1$ independent pairwise linear equations with the designated *start* variable. An example is shown in Fig. 1.9. Let S be the set of indices for which the variables are not constant where

$$S = (i_1, i_2, ..., i_M)$$

and $1 \leq i_1 < i_2 ... < i_M \leq N$, and let $\tilde{S}$ be the complementary set of indices. Note that two

[3]Also cited in [3] is [12] which has a collection of different proofs for the same result.

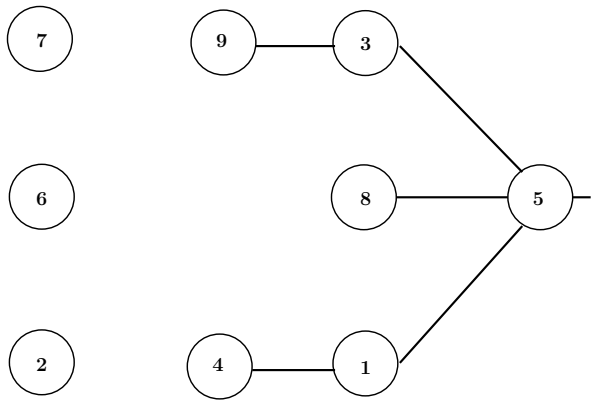

Figure 1.9: Tree with 5 as the root corresponding to x_5 being the *start* variable of the system. The isolated vertices have indices $\{2, 6, 7\} = \tilde{S}$.

lines ℓ_1 and ℓ_2 are parallel if and only if

- $S_1 \equiv S_2 = S$ and,
- for all $i_r \neq i_s$ in S_1 the $\bar{\ell}_{i_r,i_s}$ have the same abscissa for both lines.

In other words, the $\bar{\ell}_{i_r,i_s}$ for each pair $i_r \ i_s \in S$ lie on a vertical line as in the 2-D case discussed in Ch. 1. Clearly, consideration of $\tilde{S}_i \ , \ i = 1, 2$ is not needed since x_i and x_j constant implies that ℓ is orthogonal to the $x_i x_j$-plane.

We have shown that any line $\ell \subset \mathbb{R}^N$ is given, after an appropriate relabeling of the variables, by the $N-1$ equations of the form :

$$\ell_{i,j} : x_j = m_i^j x_i + b_i^j \ , \quad i < j \in S = (i_1, i_2, ..., i_M) \tag{1.14}$$

$$\ell_{o,k} : x_k = b_k \qquad , \qquad k \in \tilde{S} = (i_{M+1}, ..., i_N) \tag{1.15}$$

The "dummy" index o is introduced to preserve uniformity in indexing each relation by *two* indices rather than sometimes by two and others (i.e. the constant variables) by *one*. From this and the considerations provided in section 1.1.1 it is immediate that :

Theorem 1.1.1 Representation of a line $\ell \subset \mathbb{R}^N$ for arbitrary parametrization. *Any line $\ell \subset \mathbb{R}^N$ can be represented by $N-1$ points each of which has two indices, where $M-1$ points are given by*

$$\begin{cases} \bar{\ell}_{i,j} = (j - i \ , \ b_i^j \ \ 1 - m_i^j) \quad for \ \ i, j \ \in S = (i_1, i_2, ..., i_M) \ , \ ; i \neq j \ \ and \\ \bar{\ell}_{o,k} = (k-1, b_k, 1) \qquad\qquad\qquad for \ \ k \in \tilde{S} = (i_{M+1}, ..., i_N) \ . \end{cases} \tag{1.16}$$

As in the previous cases, it is clear that the horizontal position of the point $\bar{\ell}_{i,j}$ is determined by the slope m_i^j. Specifically for $i < j$, $\bar{\ell}_{i,j}$ is to the left of the $\bar{X}_i$ axis for $m_i^j > 1$, in between the $\bar{X}_i$ and $\bar{X}_j$ axes for $m_i^j < 0$ and to the right of $\bar{X}_j$ axis for $0 < m_i^j < 1$. The construction of a point on ℓ given the $\bar{\ell}$ is illustrated in section 1.1.5. Once a value is assigned to any variable with index in S via the $\bar{\ell}_{i,j}$'s all other variables with indices in S are known. Since variables in $\tilde{S}$ are also known all the N coordinates of a point on ℓ are known.

Two Point Form

The general *two-point form* representation, obtained in a completely analogous way, is :

Theorem 1.1.2 Representation of a line $\ell \subset \mathbb{R}^N$ – Two-Point Form. *Any line in $\mathbb{R}^N$ can be represented by $N-1$ points where, $M-1$ points (some possibly ideal), one for each edge connecting variable i and variable j are given by*

$$\bar{\ell}_{i,j} = (x_{i,j}, y_{i,j})$$

where for $\Delta_i \neq \Delta_j$ the point coordinates are

$$x_{i,j} = \frac{(j-i)\Delta_i - (i-1)(\Delta_i - \Delta_j)}{\Delta_i - \Delta_j} \quad , \quad y_{i,j} = \frac{p_{i,1}p_{j,2} - p_{i,2}p_{j,1}}{\Delta_j - \Delta_i} \quad , \tag{1.17}$$

and for $\Delta_i = \Delta_j$, the ideal point (direction) is

$$x_{i,j} = j - i \quad , \quad y_{i,j} = p_{j,1} - p_{i,1} \quad . \tag{1.18}$$

In this case, $p_{j,2} - p_{i,2} = p_{j,1} - p_{i,1}$ and there $N - M$ points given by:

$$(k-1, x_k) \tag{1.19}$$

one for each $k \in \tilde{S}$. By the definition of $\bar{\ell}_{i,j}$ the three points $(i-1, x_i)$, $\bar{\ell}_{i,j}$, $(j-1, x_j)$ lie on a straight line, so that as a consequence of eq. (1.17).

$$(i - 1 - x_{i,j})x_j + (j-i)y_{i,j} - (j - 1 - x_{i,j})x_i = 0 \ , \tag{1.20}$$

and as a consequence of eq. (1.18)

$$x_{i,j}x_j - x_{i,j}x_i = (j-i)y_{i,j} \ . \tag{1.21}$$

Once a value is assigned to any variable with index in S then via the $\bar{\ell}_{i,j}$'s all other variables with indices in S are found. But variables in $\tilde{S}$ being constant are also known and so are all the N coordinates of a point. Given x_i let λ satisfy the equation

$$x_i = \lambda p_{i,1} + (1-\lambda)p_{i,2} \tag{1.22}$$

which is always possible since by hypothesis $p_{i,1} \neq p_{i,2}$. It follows directly from eqs. (1.17), (1.18), (1.20) and (1.21) that

$$x_j = \lambda p_{j,1} + (1-\lambda)p_{j,2} \ . \tag{1.23}$$

That is, any point so defined satisfies the definition and belongs to the straight line.

1.1.5 Construction Algorithms **

$\mathcal{A}$nd now, after all this preparation, we provide "pencil and paper" construction algorithms for viewing the objects that we have been discussing. This entails the representation of the line (in terms of indexed points), as well as the construction of points on the line. Together with the rooted tree representation for the desired parametrization (i.e. the $\ell_{i,j}$) and the isolated vertices corresponding to the variables with constant values, an array D with $N-1$ rows and five columns is used as data structure. Essentially the order of rows is unimportant, although programming considerations may dictate otherwise. There are two distinct types of rows in the structure i.e.

i	j	$x_{i,j}$	$y_{i,j}$	$h_{i,j}$
0	k	0	b_k	1

The first type provides the indices of variables connected by an edge in the first two columns. The next three columns are the homogeneous coordinates $(x_{i,j}, y_{i,j}, h_{i,j})$ of the point $\bar{\ell}_{i,j}$, $h_{i,j} = 1$ for regular (i.e. finite) and $h_{i,j} = 0$ when it is ideal in which case the coordinates $(x_{i,j}, y_{i,j})$ define the slope of the direction. The second row type pertains to $x_k = b_k$ a constant, with $x_{i,j} = 0$, $y_{i,j} = b_k$ and $h_{i,j} = 1$. In other words, the variable's index is in column two and the variable's value is in column four. Column three and five have only 0's and 1's respectively.

In the complete data structure there are $M-1$ rows of the first type and $N-M$ rows of the second type. Of course, for fixed variables there is some waste in the structure. But since the occurrence of such variables will certainly be a rare event it is not useful to maintain this part of the data in a separate structure. For the time being it is assumed that no Δ_i is zero, that is, all variables are unconstrained. It is apparent that there is sufficient data in the data structure to compute a point on the line. To guide the computation of such a point a framework, called *adjacency data* is provided. It consists of a set of arrays denoted by A_i , one for each $i \in S$. The A_i has two rows and as many columns as there are edges connecting the ith variable to other variables having the form :

$$A_i = \begin{pmatrix} i_1 & j_2 & \dots & i_j \\ r_1 & r_2 & \dots & r_j \end{pmatrix}$$

The first column of the array signifies that there is an edge connecting variables with indices i and i_1 and that $\bar{\ell}_{i,i_1}$ is found in columns 3, 4, 5 of row r_1 of the data structure, likewise for the remaining columns of the array. Note that reordering rows of the data structure requires a like reordering of the *second* rows of the adjacency arrays. The algorithm constructing a point on a line can now be stated.

Input : consists of the adjacency data, the data structure and an index k for which x_k, *start variable* is to be assigned a value v_k.

Output : is a list of N values, the coordinates of a point on a line.

1. **for** $i = 1$ to N mark all x's new
2. put k on queue (Q), **for** v_k on $x_k \leftarrow$ old
3. **while** Q not empty **do**
4. **for** adjacency array A_i **for** first i in Q

(a) **while** adjacency array not empty **do**
(b) **for** x_j not old, j first in row 1
  i. find the value of x_j
  ii. append j to Q
  iii. $x_j \leftarrow$ old
  iv. delete first column of A
(c) delete first index in Q

5. stop

The treatment of the queue Q is such that each variable is dealt with at most once. The graph is connected and, via the adjacency data, each variable enters the queue. Thus each variable is dealt with exactly once. Given the value of one variable there are $N-1$ calculations required for the remaining variables. Now each edge (i,j) appears twice in the adjacency data, once in A_i and once in A_j. Since there are $N-1$ edges, this means there are $2(N-1)$ edge occurrences in the algorithm. But once a variable is calculated it is not calculated again. Hence there are $N-1$ calculations of variables with $N-1$ bypasses and the number of calculations is $O(N)$. The extension to the general case is obvious. It is also worth noting that for the case eq. (1.14) the adjacency set is implicit in the data structure as is of course is the special cases of eqs. (1.1) and (1.3). The various aspects and nuances of the above are illustrated and clarified with the following example.

An Example

Here a line in $\mathbb{R}^9$ is specified by two of its points whose coordinates appear in the first 3 columns of Table 1.1. As already discussed, a number of descriptions of the line in terms of linear equations involving pairs of variables is possible. The detailed construction of a the chosen set of $\bar{\ell}_{i,j}$ is shown in Fig. 1.10. First the polygonal lines representing the two points P_1 and P_2 are drawn. Then the $\bar{\ell}_{i,j}$, corresponding to the choice of x_5 for *start* variable, are

Coords	*Point 1*	*Point 2*		i	j	$x_{i,j}$	$y_{i,j}$	$h_{i,j}$
x_1	-4.0	2.0		1	5	2.4	-0.7	1
x_2	2.0	2.0		3	5	3.2	-0.3	1
x_3	-3.0	3.0		4	1	7.5	2.0	1
x_4	-1.6	2.0		8	5	5.5	0.25	1
x_5	1.5	-2.5		9	3	6.0	-1.0	0
x_6	-2.0	-2.0		0	2	1	2.0	1
x_7	-0.5	-0.5		0	6	5	-2.0	1
x_8	-1.0	3.0		0	7	6	-0.5	1
x_9	-4.0	2.0						

Table 1.1: On the left 3 columns are the coordinates of two points specifying a line ℓ from which a set of 8 $\bar{\ell}_{i,j}$ is constructed. The data structure used in the point construction algorithm, given in the next 5 columns, provides the $\bar{\ell}$ and their locations. Note the values $h_{i,j} = 1$ indicating finite and $h_{i,j} = 0$ ideal points. The points designating the $x_k =$ constant (i.e. $k \in \tilde{S}$) have $i = x_{i,j} = 0$.

obtained by the intersection of pairs of thin dashed lines seen in the figure. For example, in order to obtain $\bar{\ell}_{1,5}$ the x_1 and x_5 coordinates of each point are joined. The intersection of these two lines is $\bar{\ell}_{1,5}$. The corresponding rooted tree for the *start* variable x_5 is the one already shown in Fig. 1.9. Altogether for this tree the points required are $\bar{\ell}_{1,5}$, $\bar{\ell}_{3,5}$, $\bar{\ell}_{1,4}$, $\bar{\ell}_{5,8}$, and $\bar{\ell}_{3,9}$. The last is an ideal point since $x_9 = x_3 = -1$, hence the slope is 1, and it is indicated in the figure as a direction. The corresponding data structure appears in the rightmost 5 columns of the table. Here the number of unconstrained variables is $M = 5$ and there are $N - M - 1 = 3$ constant variables.

The $\bar{\ell}_{i,j}$ together with the points $\bar{\ell}_{0,j}$, on the $\bar{X}_2$, $\bar{X}_6$ and $\bar{X}_7$ axes, for the constant x_j are also shown in Fig. 1.11 together with the construction of a point $P \in \ell$ for a given value of x_5. Recall that x_5, the *start* variable, corresponds to the root of the tree. The construction can be easily modified for another choice of, possible $M = 5$, *start* variable.

Identifying the $\bar{\ell}_{i,j}$

There are certain practical problems that must be dealt with. One difficulty arises in labelling these indexed points. The following example illustrates the nature of the problem. Given the two points

$$P_1 = (a, a + b, ..., a + (N - 1)b)$$
$$P_2 = (c, c + d, ..., c + (N - 1)d)$$

where $b \neq d$ for any pair i, j

$$\bar{\ell}_{i,j} = (\frac{a - c}{d - b}\ ,\ \frac{ad - bc}{d - b})\ . \tag{1.24}$$

That is, *all* the $\bar{\ell}_{i,j}$ may be congruent (a situation which is revisited in Chapter **??**). Even without such unusual contigencies the labeling of the points may cause overcrowding in the

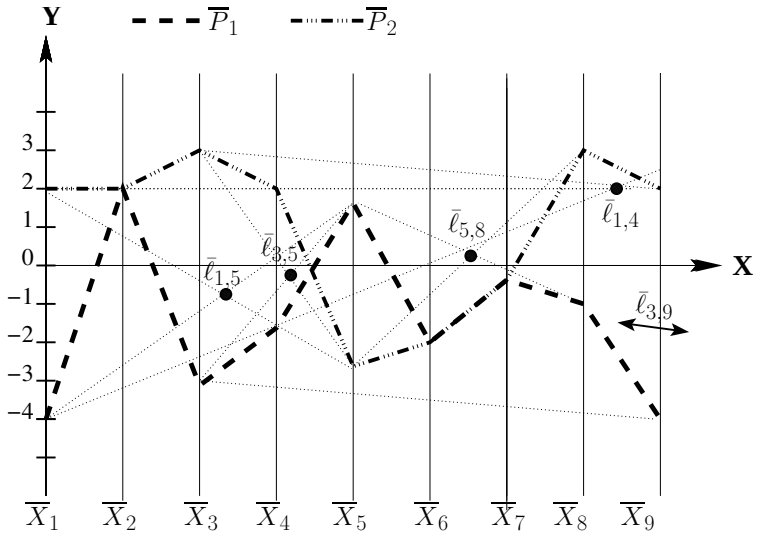

Figure 1.10: Construction of the $\bar{\ell}_{i,j}$ for the example with the rooted tree in Fig. 1.9 and the data shown in Table 1.1.

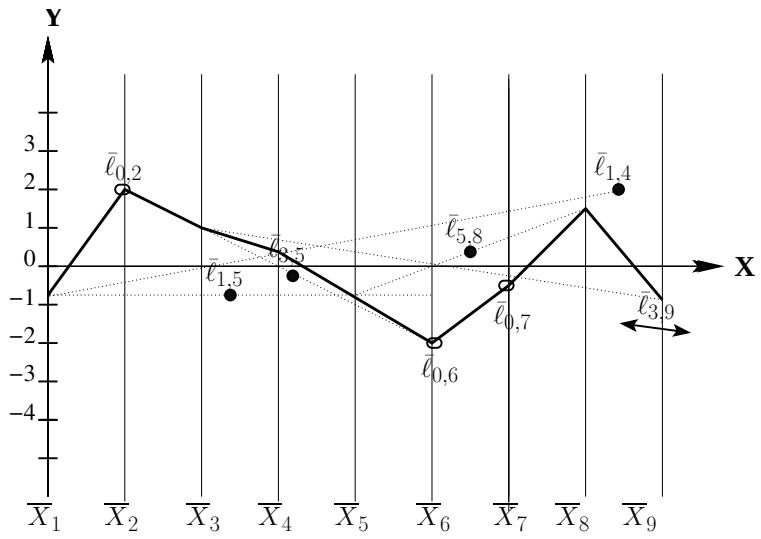

Figure 1.11: The $\overline{\ell}_{i,j}$ for the example and the construction of a new point $P \in \ell$ for $x_5 = -0.75$.

figure. This suggests that a separate tabulation of the $\bar{\ell}_{i,j}$ may be preferable rather than attempting to label these points directly on the diagram.

Lines with positive slopes

Often the $\bar{\ell}$ are out of the scope of the picture. For such cases two line segments (portions of the polygonal lines representing two points on the line) may be shown, whose intersection (when extended) defines the $\bar{\ell}$. The question arises as to whether it is possible to recast the problem of line representation in such a way that these "distant" points are not required for the representation. It is not difficult to see that the following approach not only eliminates ideal points from the representation, but has a number of other advantages.

Make the transformation of variables

$$x'_i = e_i x_i \tag{1.25}$$

where $e_i = +1$ for index $i \in \tilde{S}$. For the variables in S suppose the vertices for x_i, x_j define an edge of the tree, and that the vertex for x_j is either the root or lies between the vertex for x_i and the root, so that the linear equation in this case is

$$x_i = m_i^j x_j + b_j.$$

The remaining e_i are developed recursively. Starting at the root index i set $e_i = +1$ and visiting the remaining nodes in order setting $e_1 = +1$ or -1 to ensure that the slope m'^j_i of the line

$$x_i = m'^j_i x_j + b_j. \tag{1.26}$$

is negative. Here $m'^j_i = e_i e_j m_i^j$ and $b'^j_i = e_i b_i^j$. Since the slope is negative it follows immediately that the x coordinate of $\bar{\ell}'_{i,j}$ is between $i-1$ and $j-1$, and that the y coordinate is less in absolute value than the absolute value of b_i^j. With everything nicely bounded all the

i	j	$x_{i,j}$	$y_{i,j}$
1	5	2.4	-0.7
3	5	3.2	-0.3
4	1	1.875	-0.5
8	5	5.5	0.25
9	3	5.0	0.5
0	2	2.0	0
0	6	-2.0	0
0	7	-0.5	0

Table 1.2: The $\bar{\ell}'_{i,j}$'s where the location of $x_{i,j}$ is now between the corresponding axes.

$\bar{\ell}_{i,j}$'s can be placed within the frame of a picture. Calculation of a point on the line proceeds as in the preceding algorithm for the $\bar{X}$-system, followed by appropriate changes of sign and reflection in the $\bar{X}_i$-axis of those coordinates for which e_i is negative. Thus the algorithm is still of order $O(N)$. Likewise construction proceeds with the x'-system, followed by reflection in the x axis of those coordinates for which e_i is negative. However, while adjacency data depends only on the underlying graph structure, the numbers e_i are determined by the signs of the slopes, and hence must be recalculated for each case. Evidently this does not alter the fact that the algorithm is of order $O(N)$. For the above example it turns out that the numbers e_i are successively 1,1,1,-1,1,1,1,1,-1. The data structure is given in Table 1.2. where now no fifth column is needed. It is really a simple matter to notice by inspecting Fig. 1.10 that only the points $\bar{\ell}_{1,4}$ and $\bar{\ell}_{3,9}$ do not lie in between their corresponding two axes. Hence only for the indices 4 and 9 is $e_i = -1$ as indicated. The $\bar{\ell}'$ are constructed from the

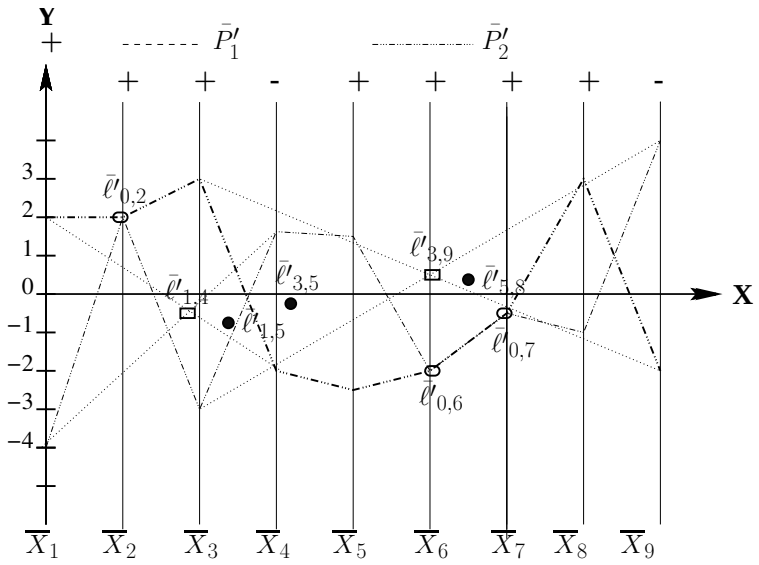

Figure 1.12: The $\overline{\ell}'_{i,j}$ for the example. First the points $\bar{P}'_1$ and $\bar{P}'_2$ are drawn. Then the two points $\bar{\ell}'_{1,4}$, $\bar{\ell}'_{3,9}$, marked by rectangles, are constructed as indicated. All other points are unchanged though the prime (') is added for notational consistency. Note the absence of ideal points.

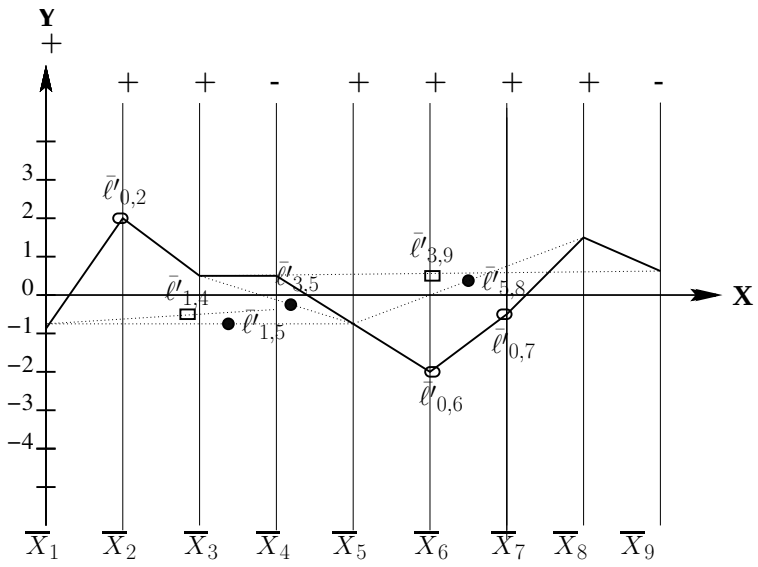

Figure 1.13: The polygonal line for the point P now denoted by P' in the new system x'_i.

transformed points $\bar{P}_i{}'$, $i = 1, 2$ and shown in Fig. 1.12.The previously contructed point P for $x_5 = .75$ is shown in the x'_i-coordinate system in Fig 1.13.

Whereas the transformation (1.25) $x_i \to x'_i$ works well for a single line its application to several lines simultaneously can be become problematical. By allowing different pairwise linear relations for each line greatly increases the prospect that their index-point representation can be brought within the framework of the picture (see exercises below).

Exercises

1. For the example of section 1.1.5 choose another *start* variable other than x_5 and repeat the constructions. Namely,

   (a) exhibit the rooted tree,

   (b) construct and show the corresponding $\bar{\ell}$,

   (c) construct and show the point P',

   (d) construct and show the $\bar{\ell}'$,

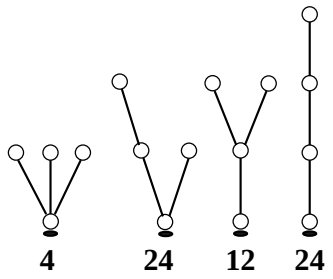

Figure 1.14: The 4 kinds of distinct rooted-trees with all vertices labeled (i.e. 1, 2, 3, 4) – the number of trees allowing permutations of the vertex labels is below each type.

(e) construct and show the point P in the x'_i system.

2. This is an example of the tree enumeration for $N = 4$. Show that,

   (a) only four distinct kinds of trees are possible – see Fig. 1.14 – and,

   (b) the number of distinct rooted trees (allowing permutation of vertex labels) are those shown below each tree.

   How many distinct (not rooted) trees are they?

3. The questions below pertain to the line representational problem of section 1.1.5 and the transformation (1.25).

   (a) For $N = 3$ let ℓ and ℓ' be two lines such $m_1^2, m_2^3 < 0$ and ${m'_1}^2, {m'_2}^3 > 0$. Is it possible via transformation eq. (1.25) to bring **all** the four $\bar{\ell}'$ in between the corresponding axes? If not give a counterexample.

   (b) For $N = 3$ and 3 lines ℓ, ℓ', ℓ'' state and prove necessary and sufficient conditions for the existence of a transformation $x_i \to x'_i$ such that the $\bar{\ell}_{i,j}, \bar{\ell}'_{r,s}, \bar{\ell}''_{u,v}$ are in between the corresponding axes. Provide a specific example.

   (c) Answer the previous question for specific numbers $N \geq 4$ and the number of lines $L \geq 4$.

   (d) Generalize the result of the previous question for arbitrary N and L — this is an open question and likely to be **hard**.

1.1.6 Rotations & Translations

In ‖-coords projective transformations in $\mathbb{R}^N$ are dual to other projective transformations [2]. In $\mathbb{R}^2$ we saw the duality between the rotation of a line ℓ about one of its points O with the translation of the point $\bar{\ell}$ on the line $\bar{O}$. This generalizes nicely in $\mathbb{R}^N$. As an example, consider the rotation of a line ℓ about one of its points P, as shown in Fig. 1.15, with the rotated line denoted by ℓ'. This corresponds to the simultaneous translation of the $\bar{\ell}$s to the new positions $\bar{\ell}'$s all being on the polygonal line $\bar{P}$. In fact, the pencil of **all** lines on P is represented by the collection of all quadruples $\bar{\ell}_{i,i-1}, i = 1, 2, 3, 4$ on the polygonal line $\bar{P}$ with ℓ' being one of these lines. This discussion leads nicely to the next topic.

Exercise

1. With reference to Fig. 1.15 show the translation of the point P to a new position P' and the line *unrotated* on P'.

1.2 Distance & Proximity Properties

1.2.1 Intersecting Lines

$\mathcal{I}$t is clear from Fig. 1.15 that $P = \ell \cap \ell'$, and this in fact suggests an easy intersection algorithm for the adjacent-variables description. Specifically the two lines ℓ , ℓ' intersect

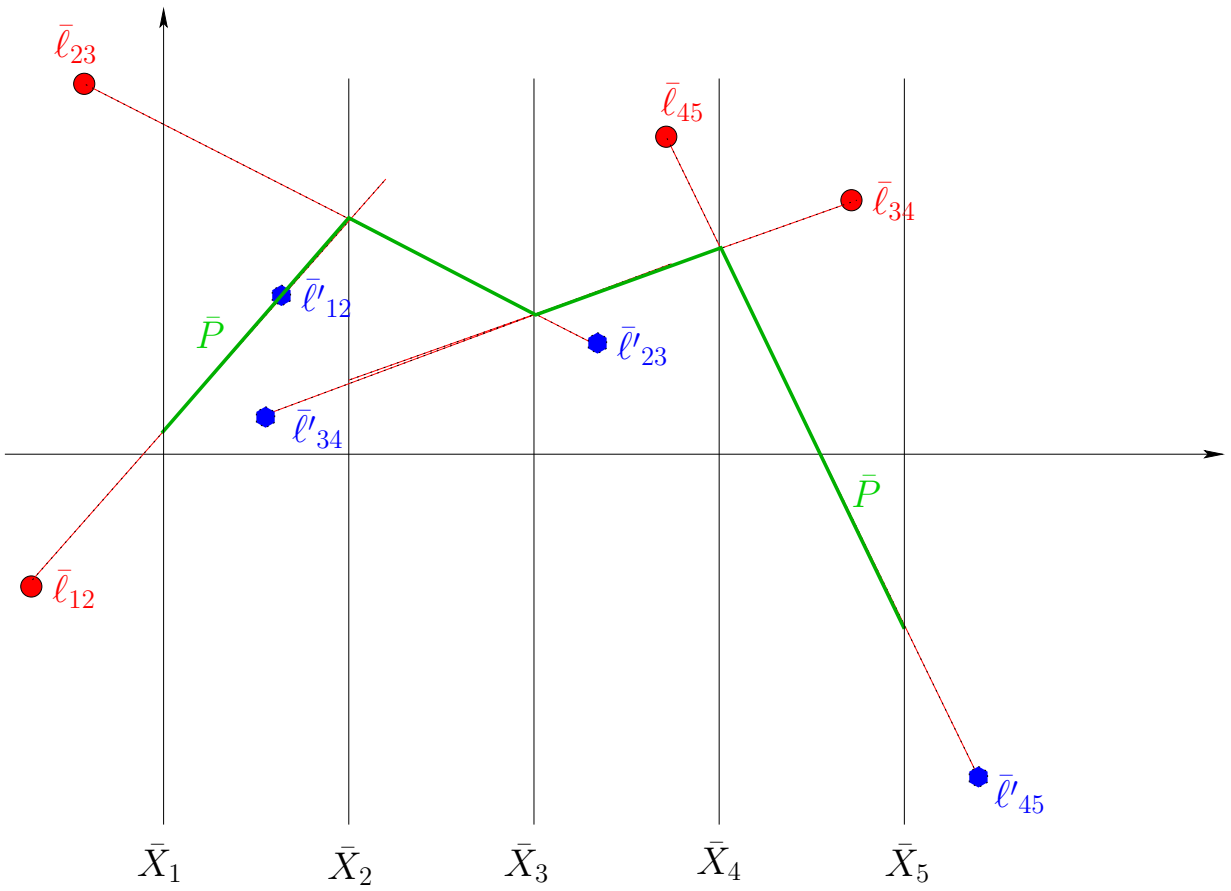

Figure 1.15: Two intersecting lines in 5-D. A different interpretation is that the line ℓ rotated about one of it's points shown here by the polygonal lin. This corresponds to the translation of the $\bar{\ell}$s to the new positions $\bar{\ell}'$s on the same polygonal line.

$\Longleftrightarrow$ the line $\bar{P}_{i,i+1}$ joining the points $\bar{\ell}_{i,i+1}$ and $\bar{\ell}'_{i,i+1}$ intersect the x_i-axis at the same point as the line $\bar{P}_{i+1,i+2}$ $\forall i$. The polygonal line so constructed represents the point of intersection of the two lines. Formally, for two lines ℓ and ℓ', described in terms of adjacent-variables,

$$\exists P = \ell \cap \ell' \Leftrightarrow \bar{P}_{i,i+1} \cap \bar{P}_{i+1,i+2} = (i, p_{i+1}) \;\; \forall i = 1, \ldots, N-2, \tag{1.27}$$

where $x_{i+1}(P) = p_{i+1}$ the ith-coordinate. Actually, $\bar{P}_{1,i} = (p_1\,, p_i)$ is the projection on the x_1x_i-plane of the point of intersection between the two lines – see Fig. 1.16 where T is the base-variable. The intersection conditions for the base-variable description of the lines as given by

$$\left.\begin{array}{rcl} \ell\,, \ell_{1i} & : & x_i = m_i x_1 + b_i\,, \\ \ell'\,, \ell'_{1i} & : & x'_i = m'_i x_1 + b'_i\,, \end{array}\right\} \tag{1.28}$$

are :

$$\exists P = \ell \cap \ell' \Leftrightarrow \alpha_i \;=\; -\frac{b'_i - b_i}{m'_i - m_i} \;=\; \frac{\Delta b_i}{\Delta m_i} \;=\; p_1\,,\; \forall i \;=\; 2,\; \ldots\,,N \tag{1.29}$$

where $x_1(P) \;=\; p_1$. Obtaining an intersection algorithm with $O(N)$ time-complexity and generalizations to other line parametrizations are straight-forward. There are special cases

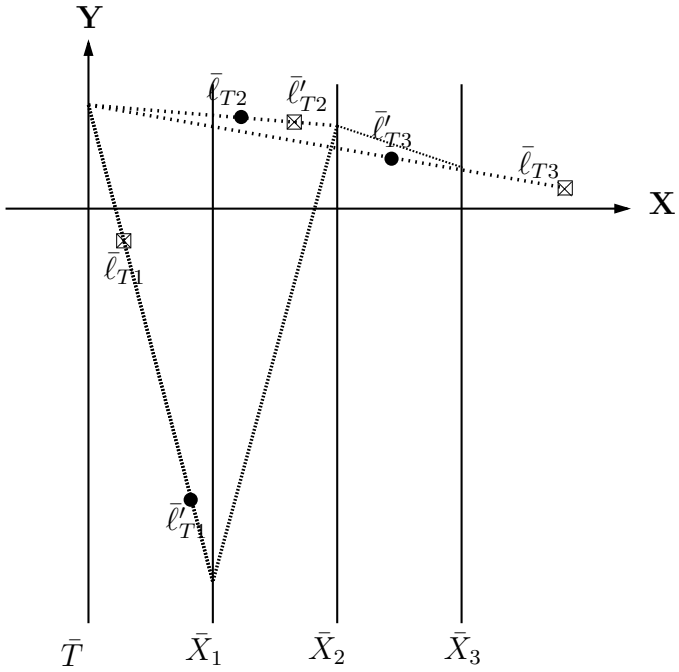

Figure 1.16: Intersection, for the base-variable line description, of two lines in 4-D. This provides the space **and time** coordinates of the place where two particles moving with constant velocity collide.

not covered by these conditions whose treatment is simply not warranted at this stage (see [9] for a complete treatment, and [7] for an early exposition).

1.2.2 Non-intersections

$\mathcal{H}$ere it is convenient to illustrate the situation in 4-D using the base-variable representation of a line:

$$x_i = v_i T + s_{o,i} \ , \ i = \ 1, 2, 3 \ . \tag{1.30}$$

and shown in Fig. 1.16. There the intersection of two lines described by eq. (1.30), each represented by 3 indexed points $\bar{\ell}_{Ti}$, is constructed. For T denoting time and $x_1 \ x_2 \ x_3$ the space coordinates of a particle moving with constant velocity $\vec{V} = (v_1 \ v_2 \ v_3)$ and initial position $S_o = (s_{o,1} \ , s_{o,2} \ , s_{o,3})$ eq. (1.30), and equivalently it's 3 point representation, provide the complete trajectory information of the particle. The two sets of triple points $\bar{\ell}_{Ti}$ and $\bar{\ell}'_{Ti}$ describe the trajectories of two moving particles. The construction in Fig. 1.16 shows that two such particles collide since they go through the same point in space **at the same time** (i.e. there is a time-space intersection). Perhaps some of the power of the ‖-coordinate representation can be appreciated from this simple example.

Clearly intersections in space (3-D and certainly higher) are very rare, so it behooves us to study non-intersections and the intersection algorithm can, of course, be used to verify non-intersections. In the sequence of figures 1.17, 1.18 and 1.19 with the minimum distance **D** between them computed and shown, the same algorithm is applied by connecting the pairs of $\bar{\ell}_{Ti} \ \ \bar{\ell}'_{Ti}$ and noticing that the three lines so formed do not intersect the T-axis at

the same value. Hence the lines ℓ and ℓ' do not intersect. Let

$$\alpha_i = (T - axis) \ \cap \ (line - passing - though \ \bar{\ell}_{Ti} \ \bar{\ell'}_{Ti}) \tag{1.31}$$

Let $I_{\bar{\ell},\bar{\ell'}} = [max \ \alpha_i \ , min \ \alpha_i]$. Observing the figures, it looks like $I_{\bar{\ell},\bar{\ell'}} \to 0$ as $\mathbf{D} \to 0$, which is reasonable since we already know that $I_{\bar{\ell},\bar{\ell'}} = 0$ when the two lines intersect ($\mathbf{D} = 0$).

1.2.3 Minimum distance between two lines in $\mathbb{R}^N$

$\mathcal{N}$ow we consider the more general problem of finding and, if possible, visualizing the minimum distance between the two lines as well as the points one on each line where the minimum occurs. In many problems what is required is the minimum distance when one or more of the variables is *constrained* to have the same value for both lines. For example, in Air Traffic Control and Motion Planning in general one is interested in knowing the time, and position, when two aircract are the closest. Therefore, it is appropriate for such applications to constraint the *time* and find the minimum distance and position under that constraint. In that case, the minimum distance is zero when there is a time-space collision. For if time is not constrained there will be two points $P_m = (t_m \, , x_1 \ , \ldots \ , x_{N-1}) \ \in \ \ell \ , \ P'_m = (t'_m \, , x'_1 \ , \ldots \ , x'_{N-1}) \ \in \ \ell'$ where the minimum distance occurs with $t_m \neq t'_m$ that would not be the closest distance between the two moving objects at the *same time*.

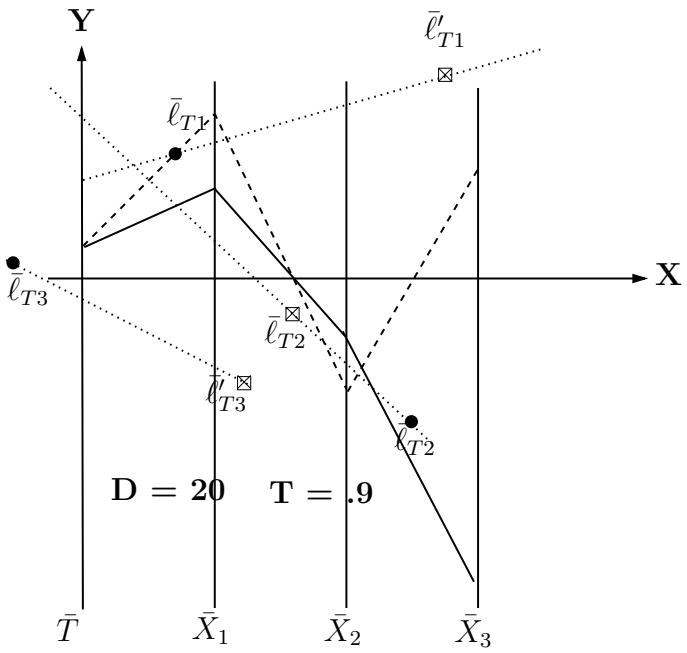

Figure 1.17: Non-intersection between two lines in 4-D. Here the minimum distance is 20 and occurs at time = .9. Note the maximum gap on the $\bar{T}$-axis formed by the lines joining the $\bar{\ell}$'s with the same subscript. The polygonal lines representing the points where the minimum distance occurs are shown and they have the *same* value of T.

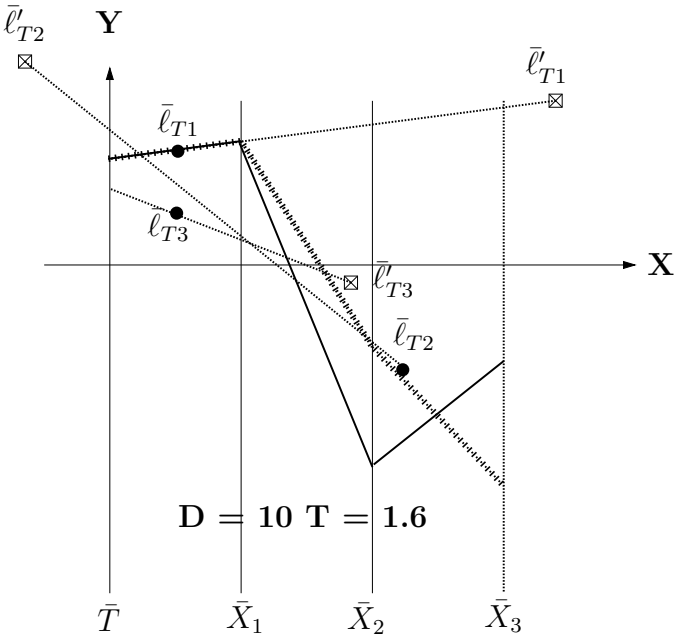

Figure 1.18: Non-intersection between two lines in 4-D. Here the minimum distance is 10 and occurs at time = 1.6. Note the the diminishing maximum gap on the $\bar{T}$-axis formed by the lines joining the $\bar{\ell}$'s with the same subscript and compare with Fig. 1.17. The polygonal lines representing the points where the minimum distance occurs are shown.

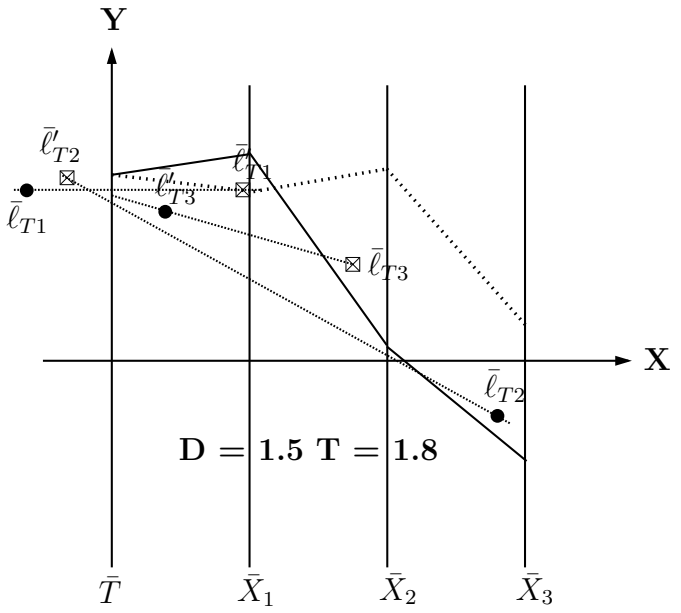

Figure 1.19: Near intersection between two lines in 4-D. Here the minimum distance is 1.5 and occurs at time = 1.8. Note the the diminished maximum gap on the $\bar{T}$-axis formed by the lines joining the $\bar{\ell}$'s with the same subscript. The polygonal lines representing the points where the minimum distance occurs are shown.

Constrained $\mathcal{L}_1$ metric minimum distance

The following result suggests that in some way the "natural" metric for ‖-coords is the $\mathcal{L}_1$ (or "Manhattan") metric where the various components can actually be seen – see Fig. 1.20. The $\mathcal{L}_1$ distance between two points $P = (x_1\ x_2\ ,\dots\ x_N)$ and $P' = (x'_1\ x'_2\ ,\dots\ x'_N)$ is given by

$$\mathcal{L}_1(P\ ,P') = \sum_{i=1}^{N} |x_i - x'_i|$$

Now if $P \in \ell$ and $P' \in \ell'$ for the lines given by eq. (1.28) and the base variable x_1 is

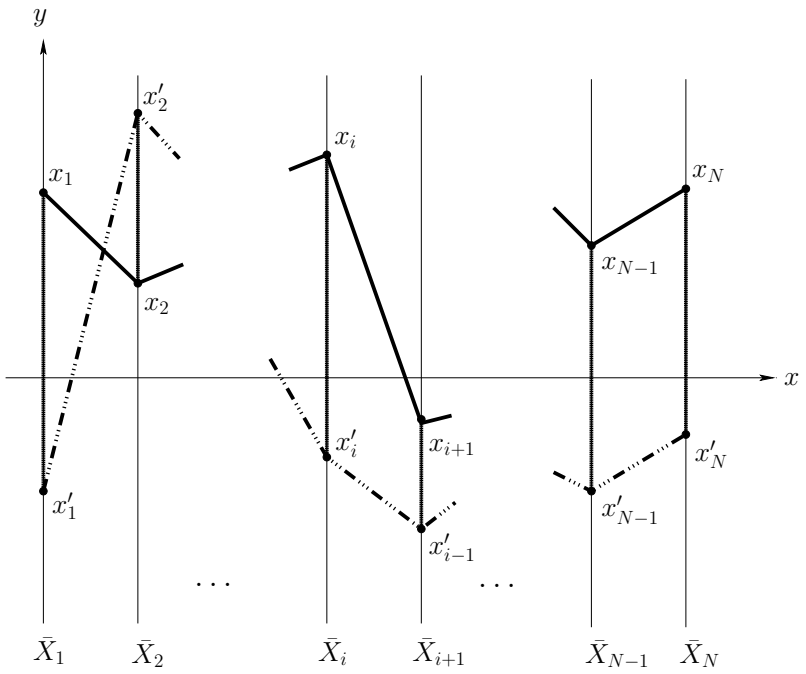

Figure 1.20: L_1 distance between the points $P = (x_1, ..., x_i, ..., x_N)$ and $P' = (x'_1, ..., x'_i, ..., x'_N)$.

constrained to be the same for both points, then the distance is given by ,

$$\mathcal{L}_1(x_1) = \sum_{i=2}^{N} |x_i - x'_i| = \sum_{i=2}^{N} |\Delta m_i||x_1 - \alpha_i| \ , \tag{1.32}$$

where the "intercepts" α_i are defined in eq. (1.29).

Theorem 1.2.1 *(Constrained Min-Dist) – The unique minimum value of $\mathcal{L}_1(x_1)$ is attained at $x_1 = \alpha_i$ for at least one $i = 2\,,\dots\,,N$.*

Proof: On the interval $\alpha_k \le x_1 \le \alpha_{k+1}$

$$\mathcal{L}_1(x_1) = \sum_{i=2}^{k} |\Delta m_i|(x_1 - \alpha_i) + \sum_{k+1}^{N} |\Delta m_i|(\alpha_i - x_1)$$

is a linear function of x_1 with slope

$$T_k = \sum_{i=2}^{k} |\Delta m_i| - \sum_{i=k+1}^{N} |\Delta m_i| \ .$$

It attains it's minimum at one of its end-points unless $T_k = 0$ in which case, $\mathcal{L}_1(x_1)$ is constant over the interval $[\alpha_k \ , \alpha_{k+1}]$.

Step 2 – Reorder the index i, if necessary, so that $\alpha_i \leq \alpha_{i+1} \ \forall \ i$. First we consider the case when $\alpha_i < \alpha_{i+1} \ \ \forall i$ and no $\Delta m_i = 0$. It is clear that for $x_1 \leq \alpha_1$, $\mathcal{L}_1(x_1)$ increases monotonically as x_1 decreases since the $|x_1 - \alpha_i|$ increase $\forall \ i$. Similarly, if $\alpha_N \leq x_1$ again $\mathcal{L}_1(x_1)$ increases monotonically as x_1 increases since the $|x_1 - \alpha_i|$ increase $\forall \ i$. So if $\mathcal{L}_1(x_1)$ has a minimum it occurs at an $x_1 \in [\alpha_2 \ , \alpha_N]$. However, this together with the conclusion of Step 1 $\Rightarrow$ that the minimum occurs at an $x_1 = \alpha_i$ for at least one value of i.

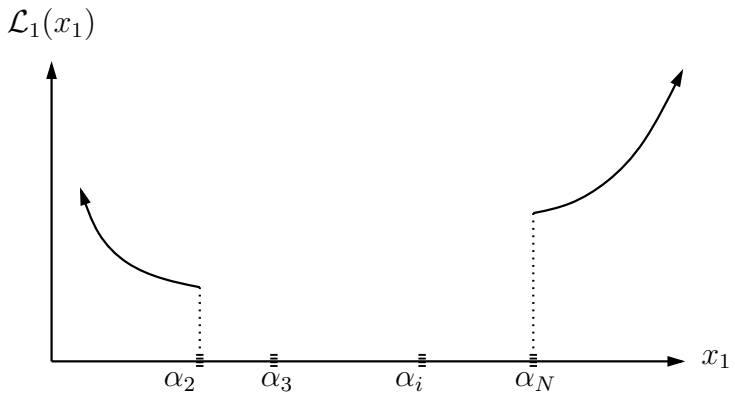

Figure 1.21: Monotone increasing portions of $L_1(x_1)$.

Step 3 – Let $S_I \ = \ \mathcal{L}_1(\alpha_{I+1}) \ - \ \mathcal{L}_1(\alpha_I)$, and by the reordering in Step2,

$$S_I = \sum_{i=2}^{N} |\Delta m_i| |\alpha_{I+1} \ - \ \alpha_i \ | \ - \ \sum_{i=2}^{N} |\Delta m_i| |\alpha_I \ - \ \alpha_i \ |$$

$$= \sum_{i=2}^{I} |\Delta m_i| (\alpha_{I+1} \ - \ \alpha_i) \ - \ \sum_{i=I+1}^{N} |\Delta m_i| (\alpha_{I+1} \ - \ \alpha_i)$$

$$- \ \sum_{i=2}^{I} |\Delta m_i| (\alpha_I - \alpha_i) \ + \ \sum_{i=I+1}^{N} |\Delta m_i| (\alpha_i - \alpha_I)$$

$$= \sum_{i=2}^{I} |\Delta m_i| (\alpha_{I+1} - \alpha_I) \ - \ \sum_{i=I+1}^{N} |\Delta m_i| (\alpha_{I+1} - \alpha_I)$$

$$= \ (\alpha_{I+1} - \alpha_I) (\sum_{i=2}^{I} |\Delta m_i| - \sum_{i=I+1}^{N} |\Delta m_i|) \ .$$

Therefore,

$$S_I = (\alpha_{I+1} \ - \ \alpha_I)T_I \ . \tag{1.33}$$

Since $\alpha_{I+1} - \alpha_I > 0$ the sign of S_I is determined by the slope of the line segment over the interval $(\alpha_I \ , \ \alpha_{I+1})$.

Step 4 – By Step 1,

$$T_I = \sum_{i=2}^{I} |\Delta m_i| - \sum_{i=I+1}^{N} |\Delta m_i| \ , \ T_{I+1} = \sum_{i=2}^{I+1} |\Delta m_i| - \sum_{i=I+2}^{N} |\Delta m_i|$$

. That is, T_{I+1} is found by moving the term $|\Delta m_{I+1}|$ from the negative to the positive sum. Therefore, T_I is monotone increasing with I. Hence from eq. (1.33), if $T_I \neq 0$ the minimum of S_I is attained at a single point. Further for $T_I = 0$, the minimum is attained over an entire interval $(\alpha_I \ , \ \alpha_{I+1})$.

Step 5 – Claim that the smallest value of $I \ \ni$

$$\sum_{i=2}^{I} |\Delta m_i| \geq 0.5 \sum_{i=2}^{N} |\Delta m_i| \tag{1.34}$$

is the value of i for $\alpha *_i$ where $\mathcal{L}_1(x_1)$ attains its minimum. For this is the first value of i for which the slope T_I can change sign. This observation provides an algorithm for finding the $\alpha *_i$ by a construction. This algorithm is described and shown shown in Fig. 1.22 .

Step 6 – When some of the α_i are equal, by eq. (1.33), there will be stationary values of $\mathcal{L}_1$ (i.e regions where it is constant) and which may or may not be at the min – see Fig. 1.23.

Step 7 – If there are $\Delta m_i| = 0$ for i in a set E, then

$$\mathcal{L}_1(x_1) = \sum_{E} |\Delta b_i| \ + \ \sum_{\tilde{E}} |\Delta m_i||x_1 - \alpha_i| \ , \tag{1.35}$$

where E denotes the complement of E. This can be carried out by a modification of the above procedure. It turns out that for small $|\Delta m_i|$ and $|\Delta b_i|$ not small, the corresponding α_i are very large and, if not all the $|\Delta m_i|$s are small, then the procedure simply ignores such values. ■

On the $\mathcal{L}_2$ minimum distance – Monotonicity with N **

Rather than the standard minimization argument, consider "fitting" a sphere of minimum radius with center at a point $P \in \ell$ and tangent to ℓ' at a point P'. The radius r of this sphere is the distance between the two lines with P and P' are the points in question. To accomplish this "fit" for any number α let $P \in \ell$ where $x_1(P) = \alpha$ and $\mathcal{S}$ the sphere with radius r and centered at P i.e.

$$\mathcal{S} : \sum_{i=1}^{N} (x_i - m_i\alpha - b_i)^2 = r^2$$

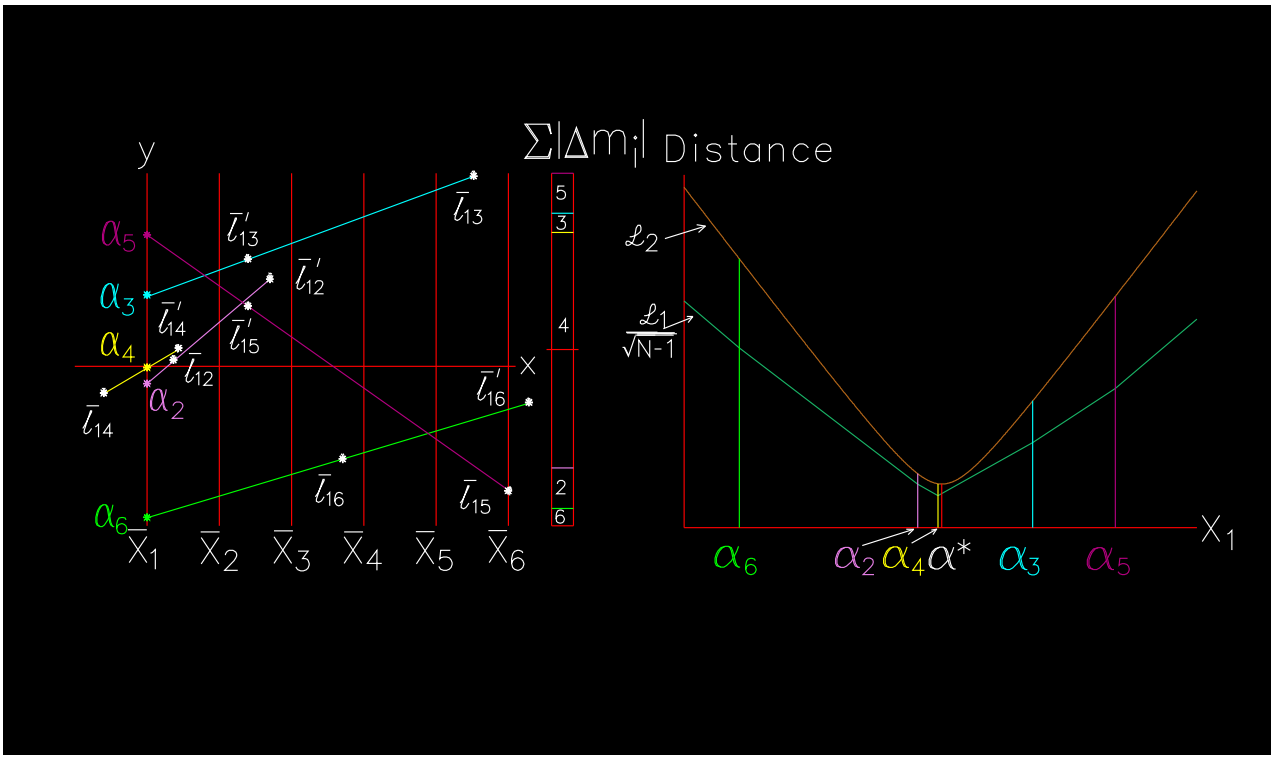

Figure 1.22: Constructing the $x_1 = \alpha_I$ minimizing the $\mathcal{L}_1$ distance between two lines. For comparison the minimum $\mathcal{L}_2$ distance occurs at $x_1 = \alpha*$. The bar chart, on the left of the $\bar{X}_6$-axis, shows the construction of the slopes T_I. There the $|\Delta m_i|$ are added in the order 6,2,4,3,5 obtained from the order of increasing α_i (as shown in the $\bar{X}_1$-axis). The index I of the interval where the mid-value of the $\sum_{i=2}^{N} |\Delta m_i|$ provides the correct $x_1 = \alpha_I$. Here $|\Delta m_4|$ dominates the sum yielding $I = 4$.

and for $P' \in \ell'$ with $x_1(P') = \alpha'$,

$$\mathcal{S} \cap \ell' : \sum_{i=1}^{N} (m_i'\alpha' + b_i' - m_i\alpha - b_i)^2 = r^2$$

Let's now "shrink" the sphere so that it is just tangent to ℓ'. Expanding and expressing in matrix form yields,

$$r^2 = (\alpha \quad \alpha' \quad 1) \begin{pmatrix} M & -C & -D \\ -C & M' & D' \\ -D & D' & B \end{pmatrix} \begin{pmatrix} \alpha \\ \alpha' \\ 1 \end{pmatrix} \tag{1.36}$$

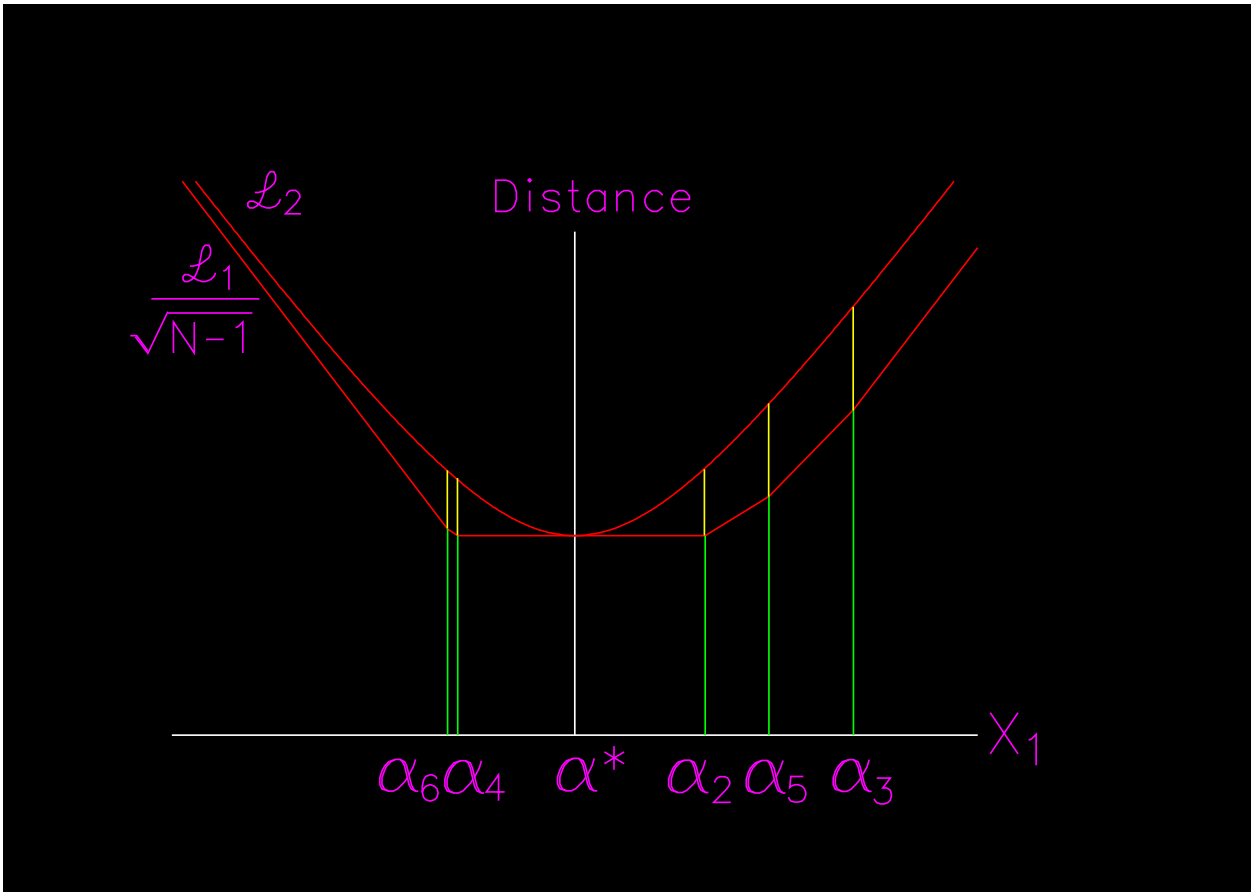

Figure 1.23: Here the $\mathcal{L}_1$ and $\mathcal{L}_2$ minimum distances are identical and occur for the same value of the constrained variable

where,

$$M' = \sum_{i=1}^{N} {m'_i}^2 \ , \ M = \sum_{i=1}^{N} {m_i}^2 \ , \ C = \sum_{i=1}^{N} m_i m'_i$$
$$B = \sum_{i=1}^{N} {\Delta b_i}^2 \ , \quad \Delta b_i = b'_i - b_i \tag{1.37}$$
$$D = \sum_{i=1}^{N} \Delta b_i m_i \ , \quad D' = \sum_{i=1}^{N} \Delta b_i m'_i \ .$$

The gist in eq. (1.36) is the quadratic relationship between the three unknowns α , α' , r . The minimum, since $Q = MM' - C^2 \geq 0$ by the Cauchy-Schwarz inequality, is reached when

$$r^2 = B + D'\alpha' - D\alpha \ , \ \alpha = \frac{M'D - CD'}{Q} \ , \ \alpha' = \frac{CD - MD'}{Q} \ . \tag{1.38}$$

When $Q = 0$ and $B \neq 0$ the lines are parallel and the constant square distance between them is $r^2 = B - D^2/M$. For $\mathbb{R}^2$ with $m_2 \neq m_2'$ the minimum $r \equiv 0$ since there the lines always intersect.

One constrained variable and monotonicity with N

The minimization of the distance now follows for $x_1 = \alpha^*$ constrained to be the same for both ℓ, ℓ'. By a similar argument s the minimum distance in this case, and the α^* which minimizes it are found to be:

$$\alpha^* = \frac{(D - D')}{V} = -\frac{W}{V} = -\frac{\sum_{i=2}^N \Delta b_i \Delta m_i}{\sum_{i=2}^N {\Delta m_i}^2} \ , \quad s^2 = B + (D - D')\alpha^*, \tag{1.39}$$

where $V = \sum_{i=2}^N (m_i' - m_i)^2, W = \sum_{i=2}^N \Delta b_i \Delta m_i$ or

$$s^2 = \sum_{i=2}^N {\Delta b_i}^2 - \frac{(\sum_{i=2}^N \Delta b_i \Delta m_i)^2}{\sum_{i=2}^N {\Delta m_i}^2} \ . \tag{1.40}$$

Applying Lagrange's identity yields the alternate form

$$s^2 \sum_{i=2}^N {\Delta m_i}^2 = \sum_{2 \leq k < j \leq N} (\Delta b_j \Delta m_k - \Delta b_k \Delta m_j)^2 \ . \tag{1.41}$$

How does the minimum distance s vary as a function of the dimension N. Let ℓ and ℓ' be two lines in $\mathbb{R}^{N+1}$ with P and P' respectively their closest points and consider the projections ℓ_N, ℓ_N' on $\mathbb{R}^N$ with closest points P_N, P_N' respectively. In eq. (1.40) denote the distance s for $\mathbb{R}^N$ by s_N and the corresponding α^* by α_N^*, $\Delta b_i = B_i, \Delta m_i = M_i$ and

$$D_N = \sum_{i=2}^N {M_i}^2 \ , \ C_N = \sum_{i=2}^N B_i M_i.$$

Subscript the analogous variables by $N+1$ for $\mathbb{R}^{N+1}$. Then

$$s_N^2 = \sum_{i=2}^N B_i^2 - \frac{C_N^2}{D_N} \ , \ \alpha_N^* = -\frac{C_N}{D_N}. \tag{1.42}$$

It is easily found that

$${s_{N+1}}^2 - {s_N}^2 = \frac{(B_{N+1} D_N - C_N M_{N+1})^2}{D_N (D_N + M_{N+1}^2)} \geq 0$$

that $s_N \uparrow N$ (i.e. monotone increasing) with $s_N = s_{N+1} \iff$

$$\alpha_{N+1}^* = -\frac{\frac{C_N}{D_N} + B_{N+1}\frac{M_{N+1}}{D_N}}{1 + \frac{M_{N+1}^2}{D_N}} = -\frac{B_{N+1}}{M_{N+1}} = \alpha_N^* \ ,$$

with $x_{N+1}(P) = x_{N+1}(P')$. All this shows that not only does s_N increase monotonically with N but also that in $\mathbb{R}^N$ the closest points between ℓ_N and ℓ_N' are **not** in general the projections of the closest points $P \ , \ P' \in \mathbb{R}^{N+1}$.

$\mathcal{L}_1$ versus $\mathcal{L}_2$ constrained minimum distance

For the constrained $\mathcal{L}_2$ distance on x_1 we consider the square :

$$\mathcal{L}_2(x_1)^2 \;=\; \sum_{i=2}^{N} (\Delta m_i)^2 (x_1 - \alpha_i)^2 \;.$$

It is left as an exercise to prove for the *unconstrained* case that in $\mathbb{R}^N$

$$\mathcal{L}_1(x_1) \geq \mathcal{L}_2(x_1) \geq \frac{\mathcal{L}_1(x_1)}{\sqrt{N}} \;. \tag{1.43}$$

Therefore for the constrained case where the dimensionality is $N-1$,

$$\mathcal{L}_1(x_1)^2 \;\leq\; (N-1)\mathcal{L}_2(x_1)^2 \;.$$

Hence if α minimizes $\mathcal{L}_1$ on a particular interval I i.e.

$$\mathcal{L}_1(\alpha) \;\leq\; \mathcal{L}_1(x_1) \;\; \forall x_1 \in I \;,$$

and β minimizes ${\mathcal{L}_2}^2$ on I i.e.

$$\mathcal{L}_2(\beta)^2 \;\leq\; \mathcal{L}_2(x_1)^2 \;\; \forall x_1 \in I \;,$$

altogether then,

$$\mathcal{L}_2(\alpha)^2 \;\geq\; \mathcal{L}_2(\beta)^2 \;\geq\; \frac{{\mathcal{L}_1}^2(\beta)}{(N-1)} \;\geq\; \frac{{\mathcal{L}_1}^2(\alpha)}{(N-1)} \;. \tag{1.44}$$

From eq. 1.43 and eq. 1.44 we obtain upper and lower bounds for the minimum $\mathcal{L}_2(x_1)$ in terms of the minimum of $\mathcal{L}_1(x_1)$. It turns out that these bounds are tight since there exist cases where these minima are equal for the same x_1. Still, it seems like the $\mathcal{L}_2$ minimum distance does not lend itself easily to visualization in $\|$-coords as the $\mathcal{L}_1$ distance does... take this as a challenge!

Finding the minimum distance between lines raises the question on how to measure the "closeness" – according to some criteria – between lines. This is matter of considerable interest in various applications like Computer Vision, Geometric Modeling and others. Gauging the "closeness" between things of the same kind is really the subject matter of *Topology* whose original name was *Analysis Situs*. In a later section we study various topologies for lines and show that it is better to do so using $\|$ rather Cartesian coordinates.

Exercises

1. Write an algorithm for constructing a point on a line given the $\bar{\ell}$ and state its complexity.

2. Write an algorithm for constructing the representation of a line given two points P_1, P_2 on the line. Be careful and delimit the special cases.

3. Write an algorithm for constructing the intersection of two lines or verifying non-intersection and state its complexity. Clarify the special cases.

4. Write an algorithm for constructing the constrained $\mathcal{L}_1$ minimum distance and state its complexity. List all special cases where the algorithm fails and why?

5. Prove eq. 1.43. – Hint use induction on the dimension N.

6. Construct a case where $\mathcal{L}_1(x_1)$ has stationary portions. State conditions for this to occur in general

7. Construct a case where the $\mathcal{L}_2(x_1)$ and $\mathcal{L}_1(x_1)$ minima are equal for the same x_1.

8. Write an algorithm for minimizing eq. (1.35) and state its complexity.

1.2.4 Air Traffic Control

*A*round 1985 the interest generated by the impending design and construction of a new Air Traffic Control (abbr. ATC) system in the USA lead, among other things, to exploration for new information displays. One of them was based on the ||-coords methodology (USA patent # 4,823,272) and it is briefly illustrated here in this the concluding section of the chapter. The reader interested on some non-technical background on ATC is referred to [5] and for an early description of the then projected new system AAAS to [6].

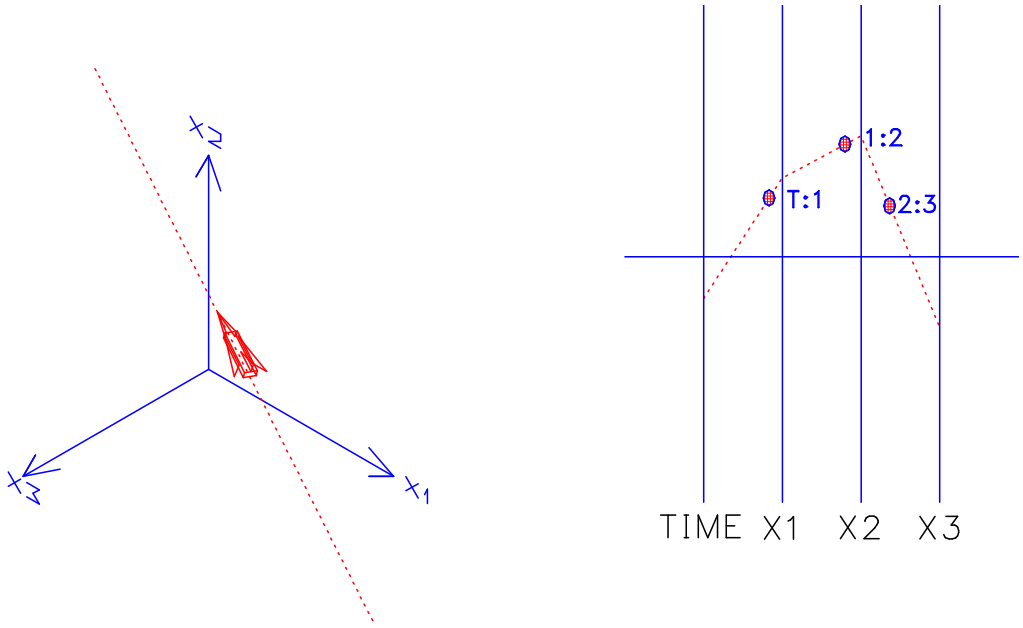

Figure 1.24: Path (left) and Trajectory (right) of an aircraft. In ||-coords the position at any given **time** may be displayed.

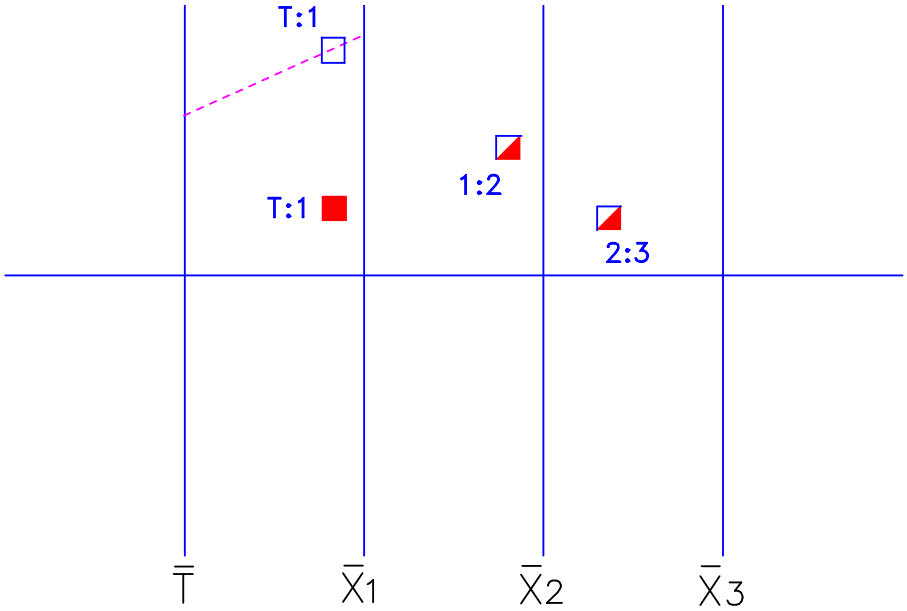

Figure 1.25: Two aircraft flying on the same path since their **1:2** and **2:3** points coincide. They have a constant separation their velocity being the same since **T:1** points have the same x-coordinate.

Displaying Trajectory Information

The trajectory of an aircraft (considered as a point) with constant velocity as occurs for much of the time in civil aviation is a line segment in 4-D which may be described by:

$$\left.\begin{array}{rcl} x_1 & = & v_1 T + s_1 \\ x_{i+1} & = & v_2 x_i + s_2 \\ x_{i+1} & = & v_3 x_i + s_3 \end{array}\right\} . \tag{1.45}$$

where the x_i are the space coordinates, T is time $\mathbf{V} = (v_1, v_2, v_3)$ the velocity vector and $S = (s_1, s_2, s_3)$ the initial position at $T = 0$. We already know that in parallel coordinates the complete trajectory information is given by 3 *stationary* points which may be labeled[4] **T:1**, **1:2** and **2:3**. Using the $\bar{T}$(time)-axis as a clock, the present position of the aircraft as well as where it was and where it will be at any given time can be found and displayed as in Fig.1.24. Recall that the horizontal coordinate of the points representing the trajectory given by eq. (1.45) is determined only from the quantities m_i with m_1 being the velocity component in the x_1 direction. With x_3 being the altitude m_2 is the tangent of, what in aviation parlance is called, the *heading*, and horizontal position of **T:3** provides the vertical velocity.

Certain properties of the trajectories are immediately evident from the display. Consider the situation where aircraft are flying on the same *path* as is often the case in civil aviation.

[4]For brevity we skip the $\bar{\ell}$ part whenever it is clear from the context.

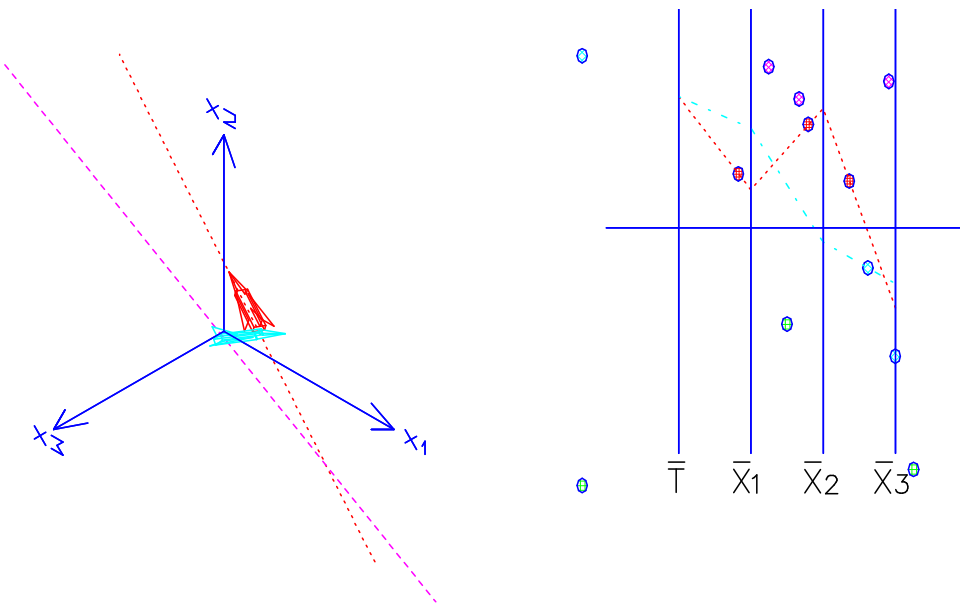

Figure 1.26: Closest approach of two aircraft. The time and closest positions are clearly seen in ‖-coords. Appearances can be misleading in a 3-D (near perspective) display where the aircraft appear to be nearly colliding. It is even more uninformative in 2-D where only a projection is displayed.

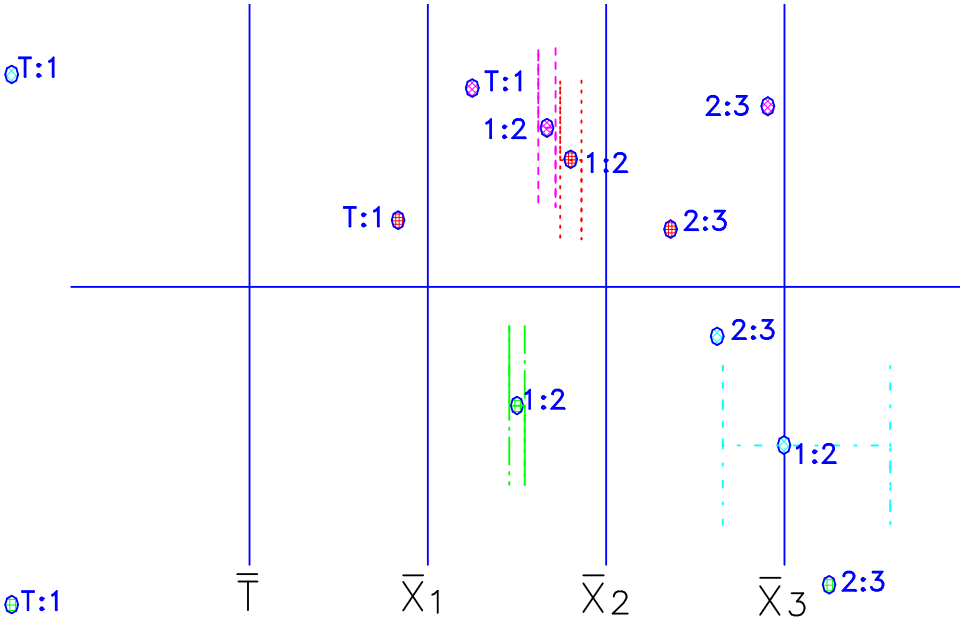

Figure 1.27: Transforming deviations in heading (angle) to lateral deviations

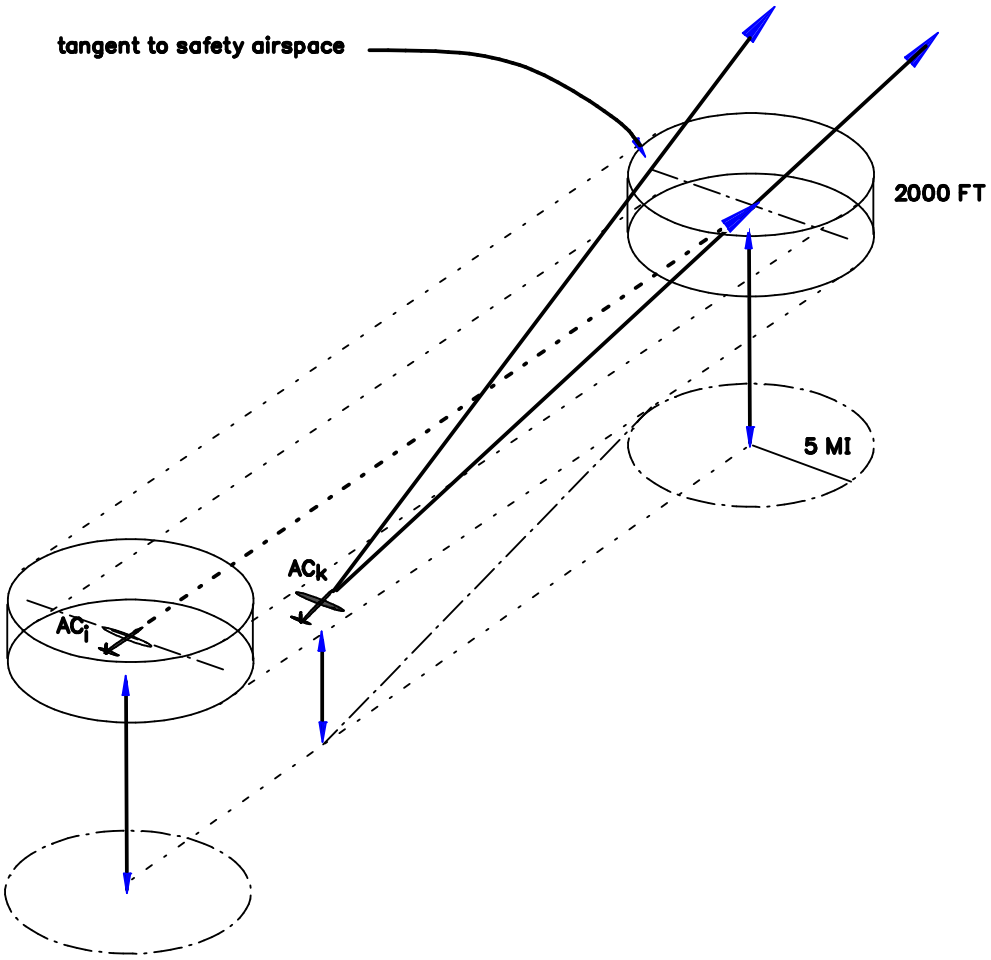

Figure 1.28: Protected Space in 3-D is a thin cylinder

An example is shown in Fig.1.25, where the points **1:2**, and **2:3** are shared since these are the points describing the path in ||-coords. Further, it is evident that they have the same velocity since the points **T:1** have the same horizontal coordinate. Otherwise, if one point is between the $\bar{T}$ and $\bar{X}_1$ axes and the other outside their velocities have opposite signs, and since they have the same path, they are flying in opposite directions. If they are both between the axes or both outside the axes then the leftmost point corresponds to the greater velocity. Other properties are also "visible".

The time and positions at which the distance between two aircraft is minimum is displayed in Fig.1.26. This critical situation is acurately portrayed in ||-coords, while in 3-D displays not to speak of 2-D projections it can appear misleading. The possibility of transforming angular to lateral deviations is illustrated in Fig.1.27. Since our eyes can better distinguish lateral rather than angular deviations, as those that may occur in the assigned path, had some "human factors" advantages but which may be partially offset by the the non-linear scale involved by the $(1-m_i)$ in the denominator of the coordinates of the points representing the path.

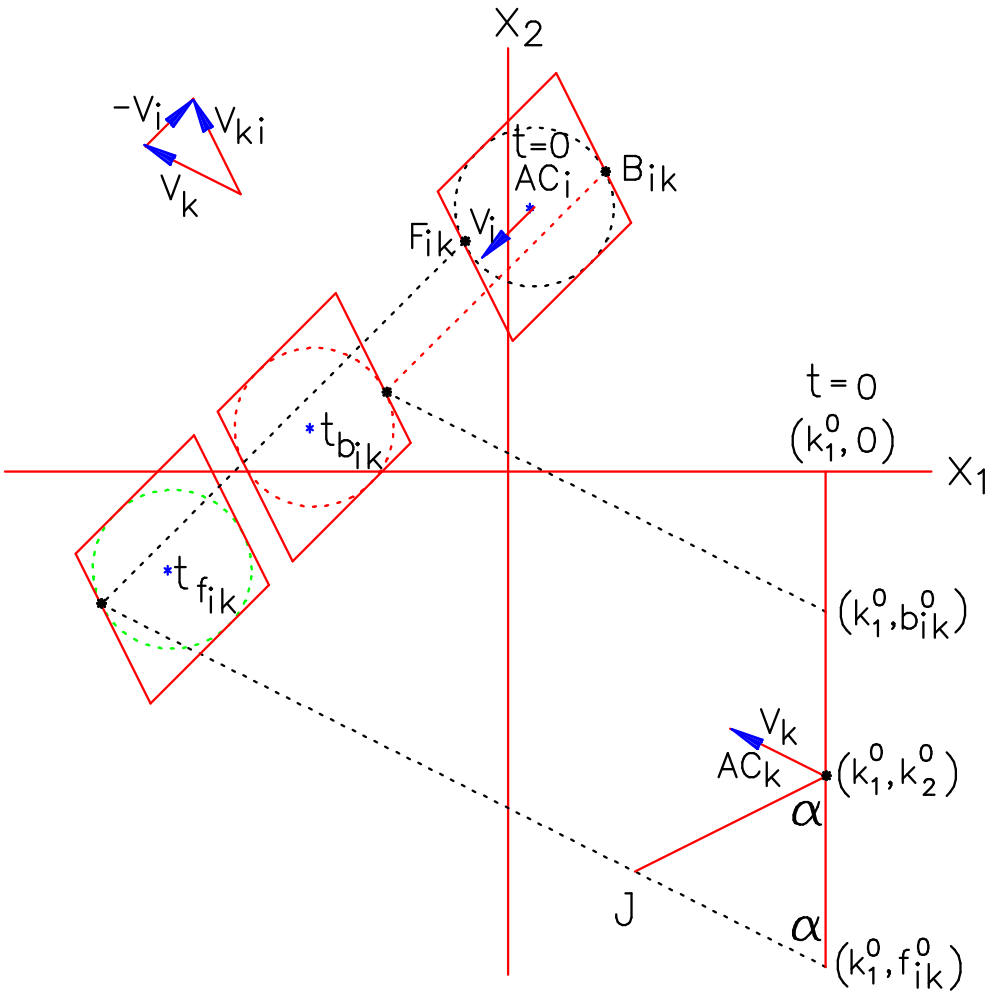

Figure 1.29: Relation between maneuver-speed and turn-angle. An equal-speed maneuver results with turn-angle α, whereas turn-angles greater than or less than α require speeds slower or faster than $|\vec{V}_k|$ respectively.

Particle Fields, Scrapes & Conflict Intervals

The goal of ATC is to direct aircraft safely to their destination while minimally interfering with their planned trajectories. This involves maintaining a minimum separation between the aircraft, detecting impending *conflicts* which are violations of the minimum separation, and *resolving* the conflicts by changing the trajectories subject to certain constraints and priorities. The typical safety requirements in 3-D (sometimes referred to as "separation standard") involve horizontal and vertical separations are shown in Fig. 1.28. The protected airspace is a thin cylinder usually 2,000 feet high and diameter of 5 nautical miles. This shape is sometimes called a "puck" due to its similarity with the hokey puck. Conflict resolution is an instance of the theoretical *Asteroid Avoidance Problem (AAP)* which turns out in general to be NP-Hard [1]. In ATC for a number of good reasons it is preferred to resolve conflicts if possible without altitude changes. An algorithm for planar (i.e. no altitude change) conflict detection and resolution in ATC (USA patent # 5,058,024) was proposed in conjunction with the design competition for the new ATC system (AAAS). It was tested on some complex groupings of flying aircraft ("complex scenarios" for short) which were provided by the USA Federal Aviation Administration [10]. We'll give the rudiments of the algorithm and then

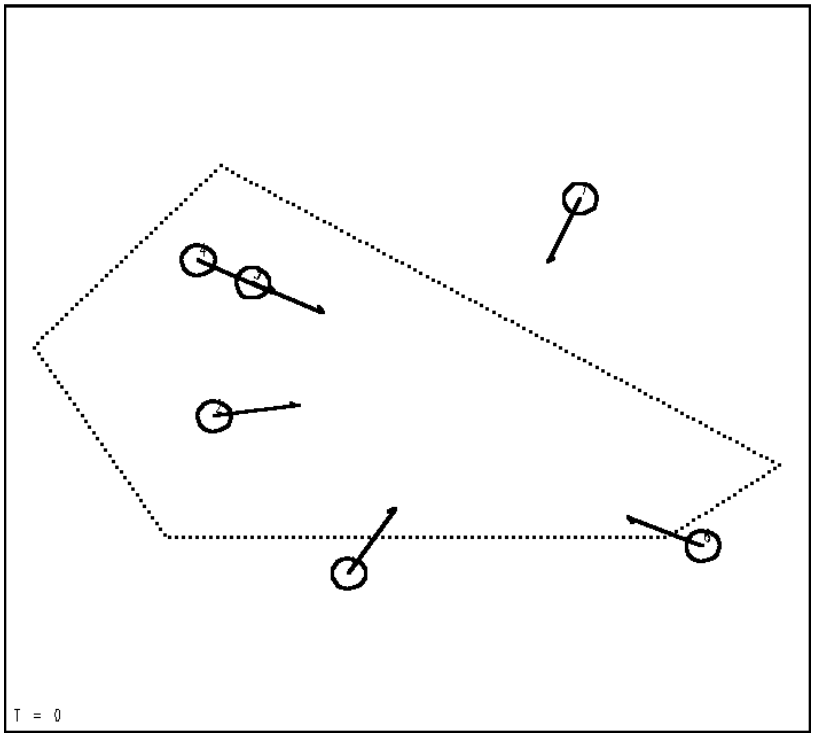

Figure 1.30: Six aircraft flying at the same altitude. These positions are at a certain time (taken as 0 seconds and shown on the left-hand-corner). Circles centered at each aircraft are the protected airspaces with the diameter being the minimum allowable separation. The arrows represent the velocities.

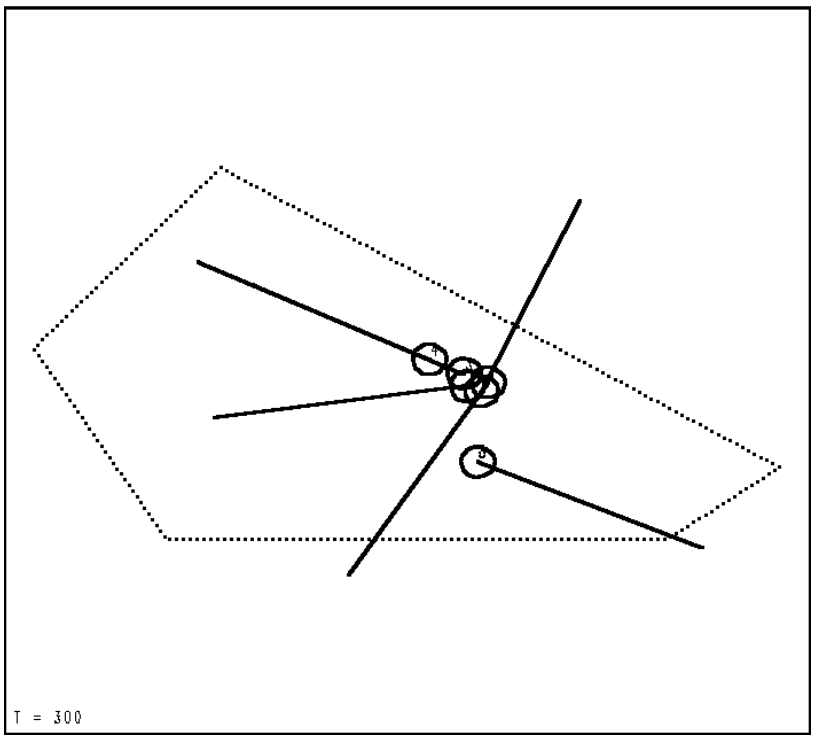

Figure 1.31: Conflicts, indicated by overlaping circles, within the next 5 minutes.

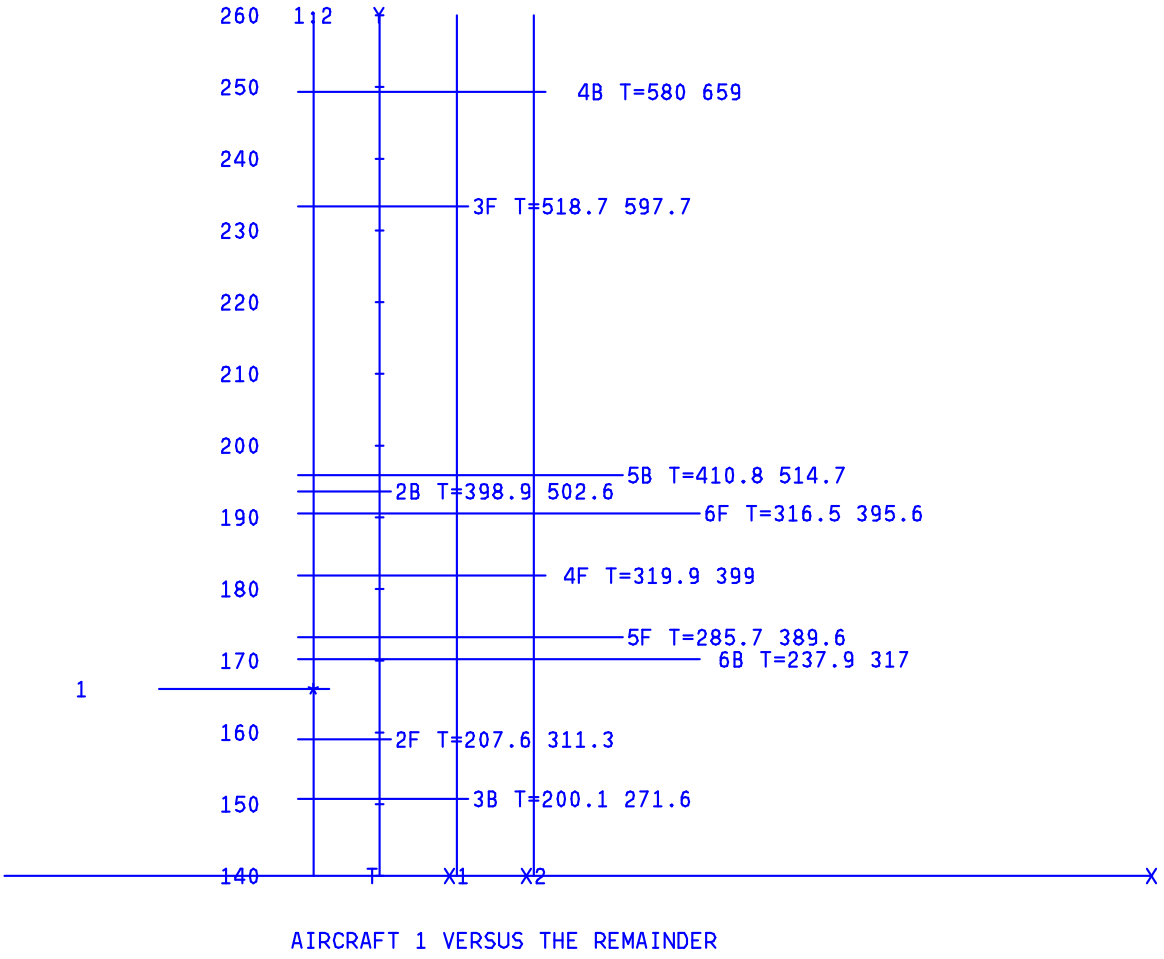

Figure 1.32: Conflict Intervals I_{1k}, $k = 2, 3, 4, 5, 6$, with respect to aircraft # 1 for the complex scenario shown in Fig. 1.30 and the conflicts shown in Fig. 1.31. The path of aircraft # 1 is shown as a point on the vertical line. Lowest point shown is **3B** and represents the path of which **B**ack-scrapes # 3 and next to it is the *time* when this occurs. In general, **B** or **F** denotes back or front-scrapes for the indicated aircraft # and the time at which the scrape occurs.

apply it to one of the complex scenarios.

Given a set of aircraft considered as points moving with a constant velocity, if the minimum separation to be maintained is denoted by d then the protected airspaces, for the planar case, are circles with radius $d/2$ centered at each aircraft and moving with the same velocity. The ensuing conflict and resolution analysis is done first for a pair of circles and then for an arbitrary number. There is no loss of generality in reducing this to the point-circle configuration above (by shrinking one circle to) a point(the aircraft itself), AC_k with velocity $\vec{V}_k$ while the circle C_i for the aircraft AC_i has double the original radius and velocity $\vec{V}_i$ as shown in Fig. 1.29. Here the velocities are assumed to be constant as when the aircraft are flying en route having reached their cruising altitude. A line ℓ, taken here for convenience to be vertical, going through AC_k is shown at the initial time $t = 0$. On ℓ a vector *field* is defined with the points on the line now being considered as *"particles"* endowed with the same velocity $\vec{V}_k$ as AC_k for $t \geq 0$. It is clear that there exists exactly one such particle, initially

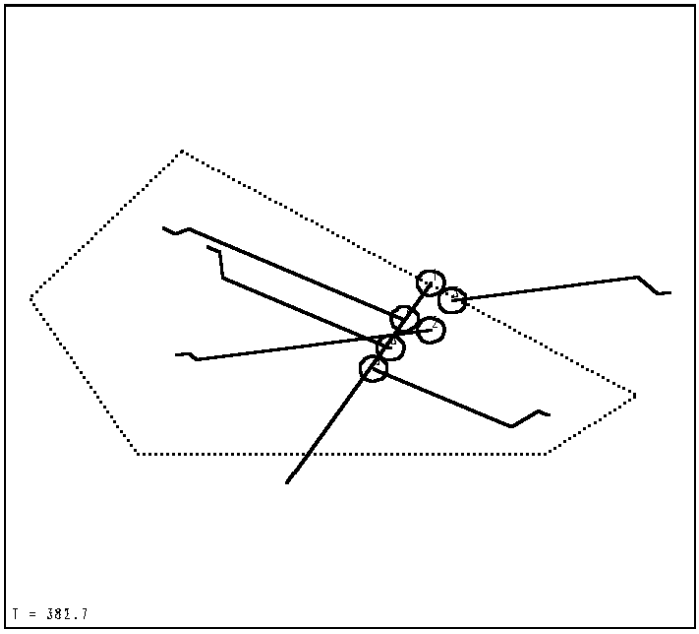

Figure 1.33: Conflict resolution with parallel-offset maneuvers. Three pairs of tangent circles.

in position $(k_1^0\ ,\ b_{ik}^0)$ which just "scrapes" (i.e. is tangent) to the circle C_i at time $t_{b_{ik}}$. Similarly, there exists a unique particle initially in position $(k_1^0\ ,\ f_{ik}^0)$ which exactly scrapes C_i from the front. The particles starting above b_{ik}^0 pass the circle safely from the back, particles below f_{ik}^0) pass in the front while those in the interval $I_{ki} = [f_{ik}^0\,,\, b_{ik}^0]$ eventually intersect the circle. This provides the conflict *detection* for of any aircraft on ℓ only those on I_{ki} will be in conflict with AC_i. For lack of a better name I_{ki} is called the *conflict interval of i with respect to k*. There is more information in Fig. 1.29 worth pointing out. The parallelogram enclosing C_i has two sides parallel to $\vec{V}_i$ and two parallel to $\vec{V}_{ki}$ the velocity of k relative to i. That is an observer sitting on C_i will see the particles passing by with velocity $\vec{V}_{ki}$. Only the particles entering this parallelogram will intersect C_i and these are the particles on I_{ki}. So the particle scraping the circle from the back will be seen by such an observer traveling on the top (back) tangent to C_i and touching it at the point B_{jk}. That, in fact, is the point where the back scrape will occur. The intersection of this side with ℓ occurs precisely at the point $(k_1^0\ ,\ b_{ik}^0)$ (why?) providing the particle responsible for the back-scrape. Since this particle has velocity $\vec{V}_k$, the intersection of a line in this direction with the path of B_{jk} (which has velocity $\vec{V}_i$) must be the position where the back-scrape occurs from which, in turn, the time $t_{b_{jk}}$ can be found. All this is easily seen by noticing that the triangle so formed is similar to the triangle shown in the upper-left with the vector subtraction. The situation is exactly the same vis-a-vis the front scrape. In this way, running through all the indices $i = 1, \ldots, N\ \ i \neq k$ the corresponding conflict intervals can be found.

In ‖-coords where the paths of the particles on I_{ki} are transformed into points. These paths all being parallel they are represented by points on will, in turn, correspond to an interval, say $\bar{I}_{ki}$ on a vertical line at $x = \frac{1}{1-m}$ with m being the slope of $\vec{V}_k$. The situation becomes more interesting when this is considered $\forall i = 1, \ldots, N\ \ i \neq k$. Since m is still the

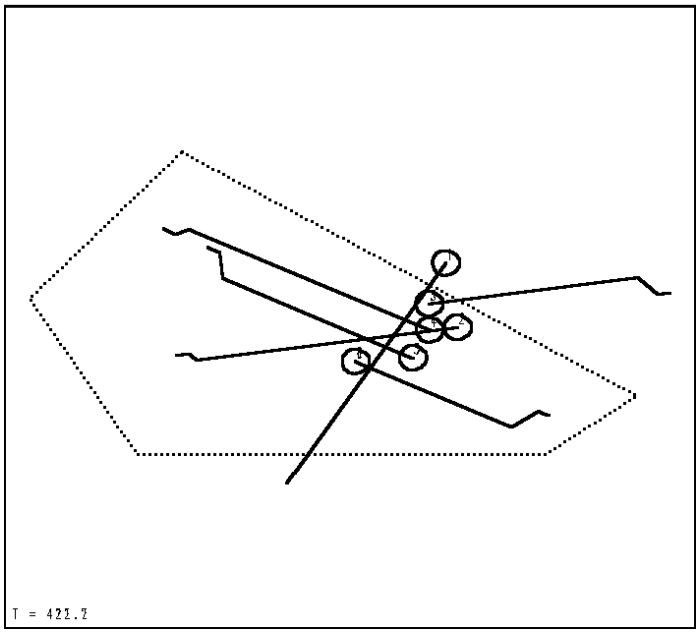

Figure 1.34: A triple tangency

same all the corresponding $\bar{I_{ki}}$ fall on the **same** vertical line. Let's see how this works on one of the complex scenarios (# 8) from [10] where 10 aircraft are involved. What makes it "interesting" is that 6 of these, shown in Fig. 1.30 are flying at the same altitude and somewhat menacingly towards each other; several impending conflicts are seen in Fig. 1.31. The corresponding conflict intervals are shown in Fig. 1.32 showing the paths for the **B**ack-scrapes and **F**ront-scrapes of the aircraft #'s $2, \ldots, 6$ and the times when they occur. The times are computed from eq. (1.45) or directly from its point representation in $\|$-coords.

Conflict Resolution

Now the fun begins! Let's denote by $\bar{k}$ the point representing the path of AC_k. Clearly, AC_k is conflict with aircraft # i $\Leftrightarrow \bar{k} \in I_{ki}$. For example, from Fig. 1.32 it is easily seen that aircraft # 1 is in conflict with # 2 and # 3 and no others. The challenge is to resolve the conflicts with the allowable maneuvers which in our case are *parallel offsets* which are commonly used en route. Such a maneuver consists of :

1. a turn (left or right) which is bounded above say by a θ_{MAX} – that is a turn which is relatively "small",

2. followed by a relatively short straight path – this is the "offset",

3. then a second turn returning the aircraft to a path parallel to the original one (i.e. the same heading). The new path is not allowed to be "too far" from the original – i.e. the distance between the path is bounded above say by a s.

4. At all times the aircraft's speed i.e. $|\vec{V}|$ is constant – that is no slow-downs or speed-ups are allowed even during the maneuvers.

The turn here is idealized in the sense that it is considered "instantaneous" in time and occurs at a point in space. All this can be adjusted to more realistic models based on the capabilities of various airplanes, policies for various airlines and air-sectors, as well as many other constraints and priorities like time-to-conflict ("validation time") etc. all of which are beyond our scope here. The strategy for conflict resolution then is to use parallel offset manuevers subject to the given constraints in order to redirect the aircraft to conflict-free paths. The conflict intervals are well-suited for this task. Since eventually the new path is parallel to the original the idea, when # j is in conflict with some aircraft, is to find a the nearest conflict-free path, say $\bar{j}'$, and check to see if under the constraints the maneuvers allow # j to reach this path *prior to the scrape*. That is we want # j to behave like the particle originaly with path j' in a neighborhood of the scrape. Specifically, for # 1 we see in Fig. 1.32 that a good candidate is the path of the particle causing the back-scrape **3B**. This is found by taking the union of all conflict intervals containing $\bar{1}$ which is also an interval (from a basic theorem in topology) and examining the closest end-point. In general, the new interval's end point may lie on conflict intervals not included in the union, such as **3F** $\in I_{14}$, in which case this intervals are added to the union until a conflict-free path is found. This process terminates since there is a finite number of intervals. Then the closest end-point is checked to see if it is *feasible*; that is if the aircraft in conflict – such as # 1 here – can be redirected to the new path subject to the constraints. It is not difficult to apply the maximum-offset constraint.

To see how equal-speed maneuvers can be achieved refer again to Fig. 1.32 where the turn-angle to maneuver-speed relation is illustrated. When AC_k makes a turn with angle α an isosceles triangle is formed so that, after the turn, the aircraft traveling with speed $|\vec{V}_k|$ arrives at J at the *same time* as the particle originaly at f^0 for they both travel with the

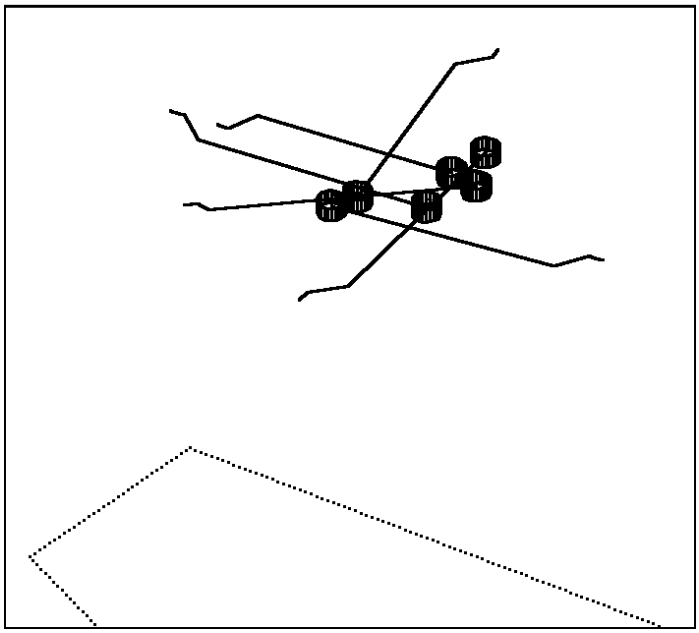

Figure 1.35: Resolution in 3-D

same speed over an *equal distance*. Hence after the point J AC_k, now in its new path, mimics the behavior of the particle – namely the scrape – in the neighborhood of some circle. From this picture we can also see that turns of more or less than α will reach the path of f_{ik}^0 sooner or later than the particle necessitating a slow-down or speed-up by AC_k to accomplish the scrape as the particle. Of course, a choice of lines other than the vertical provides more and different options and this is exploited in the implementation.

The general AAP is NP-Hard so even its specializations may have very high complexity. Since Conflict Resolution requires a real-time solution, one way to handle the need for fast response is by *cascading* the algorithm in various levels, which we briefly outline below, of increasing complexity and power. Starting with the first level (having the lowest complexity), the set of aircraft is processed by successive levels until either a resolution is found or no resolution is found by the highest level though, in in our experience, satisfactory resolutions were found at the lowest level. It is, of course, even better to run the algorithms for the various levels in parallel. The algorithm is *greedy* and in testing we found different resolutions are found by the various levels. Different resolutions (or no resolutions) may also be found by reordering aircraft on input. Listing the aircraft in order of decreasing number of conflicts, they are involved in, turns out to be a "good" ordering. Other ordering heuristics, which also consider the time-to-conflict, have been tried with varying success. When several resolutions for a conflict scenario are available rating them as to their "desirability" has turned out to be very elusive and very interesting problem.

In the simplest case ("simple rules") an aircraft in conflict, say AC_j is placed on the path found as was indicated above satisfying all the constraints. This is repeated with the *updated* (due to the repositioning of AC_j) conflict intervals. If this works out for all the aircraft in the input list a resolution is found, otherwise the next level is invoked. The computational worst-case complexity for this level is $O(N^2 logN)$ where N is the number of aircraft.

When the simple rules level finds no resolution for a particular aircraft AC_k a sweep of the conflict intervals determines which aircraft prevented it (i.e. blocking it) from being resolved – and there may be more than one. A criterion is applied in order to chose the "most troublesome" aircraft blocking the resolution,temporarily removing it and resolving the conflicts, if possible, in this way using the simple rules. Such potential conflict resolution is provisionally taken and a resolution for the remaining, including the one that was removed is attempted. If successful, the resolution spawns additional maneuvers for the blocking aircraft one at a time. This backtracking can be nicely implemented with recursion.

The worst-case complexity of the recursion is exponential. In order to terminate, the maximum number of times R which the removal of a blocking aircraft is allowed to take place is fixed a priori yielding the maximum complexity is $O(N^{R+1} logN)$. It was also found profitable to experiment with different reordering heuristics rather than invoke backtracking. Though frequently successful, no clear scenario independent criteria for the reordering were found. However, using several particle lines rather than one lead to resolutions for the vast majority of scenarios tried using only the simple rules. For equal-speed maneuvers, every particle line corresponds to a specific maneuver turn. In addition to using "idealized" turns here, a pilot can not execute a turn with absolute precision. So a certain tolerance say θ for the error is involved, So placing the particle lines in multiples of θ and covering in this way the allowable turn range (i.e. $\pm\ \theta_{MAX}$) is a way to check for all *implementable* resolutions without increase in the time complexity.

Let us illustrate all this with the chosen scenario shown in Fig. 1.30. To "judge" it's difficulty in a practical way an expert Air Traffic controller was asked to resolve it. He could only resolve 4 aircraft without altitude change. Even that required a great deal of time and the solution was certainly not "real-time". Our algorithm was able to resolve all conflicts without altitude changes using only two particle lines the resolution being shown at two time-frames in Figs. 1.33 and 1.33. There are several instances of tangent circles. This is because the algorithm is built to utilize the scrapes and which, in turn, result in minimally disturbing the aircraft from their original paths. The figures hopefully illustrate the difficulty in finding constrained maneuvers which allow the circles to possible touch but always exquisitely avoiding overlapping.

As an interesting aside notice, from Fig. 1.32 that $\cap_{k=2}^{6} I_{1k} = [4\mathbf{F}, 6\mathbf{F}]$. So any point on the interval $[4\mathbf{F}, 6\mathbf{F}]$ represents a path which eventually intersects *each one of the circles*. This is an instance of the *One-Shot Problem (OSP)* which in general is NP-Complete [11]. In 1989 another algorithm using particle fields on the *whole plane*, rather than just particle lines, was derived (USA patent # 5,173,861). Also it accomplished some resolutions with altitude changes Fig. 1.35. It was reported[5] that the U.S.A. Air Force is testing the use of these algorithms for positioning several aircraft in formation flying [4] and for conflict detection and resolution [14].

Exercises

1. In Fig. 1.32 most of the associated times with the scrapes *increase* as the point representing the scrape path is higher on the vertical line. Provide the conditions needed for these times to increase or decrease monotonicaly.

2. Provide the formulation for using several line fields $\ell(\theta)$ at different angles θ with the horizontal axis. Describe how you would efficiently construct the corresponding *conflict intervals* and their functional dependence on θ.

3. Define a particle field on the whole plane – as was done only for ℓ. Find the allowable particle regions (i.e. the set of particles satisfying the constraints) associated with:

   (a) Maximum offset

   (b) Maximum Angle

   (c) Conditions where the approach may fail

4. Find the conditions where the approach of "particle lines" fails

5. Formulate the "one-shot problem" carefully. Describe conditions where the conflict intervals can be used interactively to find one-shot solutions.

.

[5] I am grateful to D.Sobiski for this information.

Index

References

[1] J. Canny and J. Reif. *New Lower Bound Techniques for Robot Motion Planning Problems, in Proc. of 28th Symp. on Found. of Comp. Sci.* IEEE Comp. Soc., Washington, D. C., 1987.

[2] A. Chatterjee. *Visualizing Multidimensional Polytopes and Topologies for Tolerances.* Ph.D. Thesis, Dept. Comp. Sci., Univ. of S. Calif., 1995.

[3] S. Even. *Graph Algorithms.* Computer Science Press, Rockville MD, 1979.

[4] LORAL FSD. *Test Report for the Quidk-Look FLight Demonstration of the IntraFormation Positioning System, Document Nos. 93-A37-002, Contract f33615-90-C-3609.* LORAL Federal Systems Co, Oswego, 1994.

[5] H. S. Guyford and J.J. Haggerty. *Flight – Life Science Library.* Time Inc., New York, 1965.

[6] IEEE. The faa's advanced automation program — special issue. *IEEE Computer J.*, 20-2, 1987.

[7] A. Inselberg. *N-Dimensional Graphics, Part I – Lines and Hyperplanes, IBM LASC Tech. Rep. G320-2711, 140 pages.* IBM LA Scientific Center, 1981.

[8] A. Inselberg and B. Dimsdale. Multidimensional lines i: Representation. *SIAM J. of Applied Math.*, 54-2:559–577, 1994.

[9] A. Inselberg and B. Dimsdale. Multidimensional lines ii: Proximity and applications. *SIAM J. of Applied Math.*, 54-2:578–596, 1994.

[10] R. O. Lejeune. *Government Provided Complex Scenarios for the Advanced Automated System Design Competition Phase, MTR-85W240.* MITRE Co, McLean, Virginia, 1985.

[11] N. Megido. *On the Complexity of some Geometric Problems in Unbounded Dimension, in IBM Res. Rep. RJ5744(58033).* IBM Research, 1987.

[12] J. W. Moon. *Various Proofs of Cayley's Formula for Counting Trees, Seminar in Graph Theory, F. Harary (ed.).* Holt, Rinehart & Winston, 1967.

[13] J. Riordan. *An Introduction to Combinatorial Analysis.* John Wiley, New York, 1958.

[14] D. J. Sobiski. *Collision Detection and Collision Avoidance, Document Nos. 196A798.* LORAL Federal Systems Co, Oswego, NY, 1995.