Monograph:
Edwards, J.B. and Mazandarani, M. (1986) For a Simulation of the Steering of Solid-Based Mining Structures. Research Report. Acse Report 291 . Dept of Automatic Control and System Engineering. University of Sheffield

[redacted]

PROGRAM DESCRIPTION;

For a simulation of the steering of solid-based mining structures

J. B. Edwards

& M. Mazandarani

February 1986

Research Report No. 291.

Department of Control Engineering

University of Sheffield

Mappin Street

Sheffield S1 3JD.

PROGRAM DESCRIPTION:

for a Simulation of the Steering of Solid-Based Mining Structures

J. B. Edwards and M. Mazandarani

CONTENTS

1. Introduction

The effect of varying the geometry of coal-winning machines and machine systems on their vertical steering ability is the subject of increasing speculation, prompted by the advent of ranging-drum shearers and underplated (closed-bottomed) armoured face-conveyors (a.f.c's). Some doubt generally exists as to exactly how solid-based structures, like underplated a.f.c's ride over the steps cut by a vertically ranging drum and the attitudes they take up when resting on the stepped cut-floors produced.

The situation with earlier open-bottomed a.f.c's was much simpler in that it could reasonably be assumed that front-and rear-edges of the structure formed the principal points of contact with the floor and that intermediate high-spots were bridged or planed off by the structure. Rolling underframes for steering fixed-drum shearers were also simpler to analyse even with under-plated conveyors since again, points of floor-contact were readily predictable.

With ranging-drum machines and underplated a.f.c's, the situation is complicated by the existence of geometrical factors additional to drum and conveyor widths, such as pick-to-pan distance, incomplete pushover, machine system centre-of-gravity etc., all of which can vary from one installation to another. On automatic systems, the coal-sensor location in line-of-advance is a further design variable.

Similar questions arise also in the case of tunnelling machines and roadheaders. Whilst relative dimensions differ from the power-loader situation, the basic geometrical arrangement still pertains. Namely, we have a solid machine base resting on high spots in a floor cut previously by the cutting-head which is located some distance ahead of the base and the depth of cut (the sumping distance) may not equal the length of the cutting head. Vertical steering is again attempted by raising or lowering the head relative to the projected underside of the base.

Fig.1. illustrates the general situation, be it a coal-face or tunnelling problem, and identifies the basic problem parameters viz:

W_b = base-length

W_d = drum - (or cutting-head) length

W_p = separation between drum and base (pick-to-pan distance)

W_a = advance distance

W_g = location of machine's centre of gravity from base-front

W_s = location of height (coal) sensor, if any

1.1 Program Capabilities

A general simulation program has been developed and written by the authors of this document that accept any practical values of W_d, W_b, W_p, W_a ($>0.5\ W_d$), W_b and W_s. It then computes and displays, with each advance of the machine, the height pattern, y, of the cut-floor at each step created by raising or lowering the cutting-head in response to vertical steering action, J. The steering actions may be applied manually from the keyboard by pressing 'raise' or 'lower' keys before each advance or they may be applied automatically, based on height-sensor, roof-follower and/or tilt-transducer measurements. The program also computes and displays the front and rear base heights h_1, h_2 at each advance. The output is displayed graphically on the screen of a monochrome storage visual display unit or alternatively as a list of numerical height values. A list of the control actions, J applied manually or automatically is also available on request.

The program assumes no breakage of the cut-floor whatsoever and works by storing the cut-floor high-spots until these pass under the base, whereupon the height and tilt of the system is adjusted to minimise its total potential energy, without penetrating the cut-floor.

The program may be used at the face-design stage to investigate whether or not any proposed geometry is steerable (not all arrangements are steerable) or it may be used to obtain a recommended escape procedure (a list of J's) for an existing underground face or machine whose control has been lost temporarily.

The program will also assist in tuning up the height, tilt and integral gains of automatic systems before commissioning and will also check whether the system is indeed controllable (not all arrangements are controllable).

1.2 Document Layout

The purpose of this document is to explain the method of modelling the steering and floor-fitting process. The physics of the problem is described in Section 2 and the appropriate mathematical equations and height constraints developed. These are each numbered to allow cross reference to the comments in the program listing reproduced in Section 6. The comments also attempt to convey a physical appreciation of the operations being carried out in any particular program segment e.g. calculations of height ordinates, comparing heights at overlap points to determine high spots in the cut floor, storing and shifting crucial ordinates backwards by one step at each advance of the machine control they pass under the base etc. It is strongly recommended that Section 2 be read before and again in conjunction with the program listing, particularly if changes are contemplated by a purchaser of the source code.

The actual fitting of the base to the cut floor is effected using a standard DUOPLEX Linear Programming routine not provided here in source-code form (this not being the property of the Department of Control Engineering). Linear programming is a well-known optimisation technique however, and merely minimises a linear objective function of variables (here the base-end heights) within a set of linear constraint inequalities. The linear objective function in this context is the potential energy of the machine and base structure, and the parameters of the linear constraints are related to crucial heights in the cut floor: the derivation of which is explained in Section 2. Readers wishing to know more about Linear Programming are referred to References 1 and 2 in Section 5.

Section 3 provides some specimen results obtained by the authors from running the system with various parameters whilst Section 4 lists and defines the principle symbols used in Section 2. Their FORTRAN equivalents given in the program listing have been chosen to match the algebraic symbols as closely as possible.

2. Steering Problem Formulation

Machine Equations

Fig.2. shows the basic variables specifying the behaviour of the cutting machine itself and the flat base-structure upon which it is mounted. The machine is shown making cut number n after making n-1 advances from left to right having started with front-and rear-base heights = $h_1(o)$ and $h_2(o)$ respectively. Small-angle geometry is assumed throughout this analysis so that the horizontal advance-distance can be taken as W_a. Each advance is assumed to be constant from cut to cut furthermore. As shown, y(n) denotes the height of the leading edge of the cutting drum's underside and this is related to the base front height $h_1(n)$, tilt $\alpha(n)$ and steering control action J(n) thus

$$y(n) = h_1(n) + (W_d+W_p) \alpha(n) + J(n) \qquad (1)$$

where $$\alpha(n) = \{h_1(n) - h_2(n)\} /W_b \qquad (2)$$

assuming small-angle geometry (as already emphasised). Parameters W_d, W_p and W_b denote the drum width, drum-to-base spacing and base-width respectively whilst $h_2(n)$ denotes the rear height of the base.

Given a value for J(n) from either an automatic control law or from a manual input therefore, the new cut-floor height y(n) may be calculated if the base coordinates $h_1(n)$ and $h_2(n)$ are also known. These depend on the heights of the high-spots (created previously by the drum) presently lying beneath the base and on the position W_g of the centre of gravity of the machine + base measured back from the leading edge of the base: as shown in Fig.2. A steering model for the overall system clearly can only be developed if the height and location of all the high spots created in the cut floor can first be determined.

This is the subject of the following Sections of this paper.

Location of Floor Break-Points

The cut-floor clearly comprises a sequence of piecewise-linear segments with upward and downward steps between the breakpoints. As Fig.2. demonstrates, however, the spacing of these steps is not necessarily constant despite the constancy of advance distance, W_a. A sequence of upward and downward movements of the drum with respect to its previous position, is shown in Fig.2. such movements being brought about by two causes generated by equations (1) and (2), viz;

(a) deliberate near-vertical adjustment J(n) of the drum with respect to the base structure on which the machine currently rides

and

(b) displacement of the present base position $h_1(n)$, $h_2(n)$ with respect to the previous cut-floor position produced during cut n-1.

A succession of downward movements yield break points of type A in Fig.2, produced by the trailing edge of the cutting head, whilst breakpoints of type B, produced by the leading edge are generated by a sequence of upward movements. Although of the same spatial frequency W_a^{-1}, the two breakpoint sequences are obviously phase shifted with respect to each other by drum width, W_d, so that when an upward sequence is followed by a sequence of downward movements, two successive high-spots are produced spaced at a distance of only

$$X = 2\,W_a - W_d \tag{3}$$

as typified by the two steps preceeding A in Fig (1) whilst the reverse situation produces two high-spots spaced at

$$Y = W_d \tag{4}$$

as Fig.2 demonstrates two steps prior to step B.

A simulation model must therefore keep track of the drum position at its front and rear-edge locations taking due account of overlap distance

$$W_o = W_d - W_a \tag{5}$$

and overlap is now considered in more detail. Before proceeding, however, we should note that, to keep spacing X positive, we shall restrict attention to situations where

$$W_a > W_d/2 \tag{6}$$

i.e. where $W_o < W_a$ (7)

Otherwise the number of step positions increases causing greater complexity. Fig.3. defines four ordinates per drum location: y(i) and y'''(i) = the front and rear heights of the drum respectively, produced during cut number i and intermediate heights y'(i) and y''(i) produced during the same cut but which clearly require comparison with overlapping ordinates y'''(i+1) and y(i-1) respectively to determine which of each pair represents the true high-spot in the cut floor and which therefore requires storage pending the arrival of the advancing base structure, one or several cuts later. Consideration of Fig.4, however, reveals that the high spot altidudes can only be y'(i) {not y'''(i+1)} and y''(i+1) {not y(i)} irrespective of whether the drum rises or falls on cut i+1 with respect to its previous position on cut i. The only ordinates generated on cut n with the potential to ultimately affect the base position (apart from ordinates beneath its front and rear toes, to be considered later) are therefore:

$$y'(n) = h_1(n) + \alpha(n)\ (W_p+W_a) + J(n) \tag{8}$$

and

$$y''(n) = h_1(n) + \alpha(n)\ (W_p+W_o) + J(n) \tag{9}$$

It is now necessary to consider the number of advances between the generation of y' and y'' and the commencement and conclusion of their effect on base position i.e. the intervals over which these ordinates should be stored prior to and during their use in predicting the base position.

Storage Intervals

In considering the advance of the base from one cut to the next it is important to realise that two fundamentally different situations can occur as regards the location of the base front with respect to the breakpoints produced by the cutting drum. The system geometry determines which situation arises. As illustrated in Fig.5. (a), the front of the base may straddle both $y'(n)$ and $y''(n)$ after making a definite number, p, advances from the creation of $y'(n)$ and $y''(n)$, or, as shown in Fig.5 (b), only the trailing breakpoint ordinate $y''(n)$ may be straddled first after, say, r advances, and $y'(n)$ not straddled until r+1 advances have occurred. Defining p as the number of advances needed for the base to first encounter $y'(n)$ (after its creation) and r as the number for $y''(n)$ to be first encountered, it is clear that, if $r=p$, the base front always lies in the overlap region between the front and rear of the drum on successive cuts whilst $p=r+1$ corresponds to the base-front always falling in the nonoverlap region. It is readily deduced from Figs. 5(a) and 5(b) respectively that integers p and r are given by

$$W_p/W_a + 2>p>W_p/W_a+1 \tag{10}$$

and $$(W_p+W_d)/W_a>r>(W_p+W_d)/W_a+1 \tag{11}$$

As Figs. 6(a) and 6(b) illustrate, a similar alternative pair of situations are possible (again dictated only by the fixed geometry of the system, this time involving base length W_b). If integer s is defined as the maximum number of system advances for the base-end to straddle $y''(n)$ (following the creation of $y''(n)$) and q = that maximum number to straddle $y'(n)$ then s and q are given by

$$(W_p+W_b+W_d)\ /W_a-2<S<(W_p+W_b+W_d)/W_a-1 \tag{12}$$

and $$(W_b+W_p)/W_a<q<(W_b+W_p)/W_a+1 \tag{13}$$

Hence, if $q=s$, the rear of the base always lies in overlap region but, if $s=q-1$, it will occupy the nonoverlap region. These two situations are illustrated in Figs. 6(a) and 6(b) respectively. {Figs.5 and 6 are intended to show horizontal locations of ordinates only and the heights of the ordinates sketched carry no significance. For this reason, no tilts of the structure are shown}.

Height constraints at the breakpoints

Thus, during cut number n, two sets of breakpoint ordinates lie beneath the base, viz

$$f'(n,i) = y'(n-p-i+1),\ 1\leq i\leq q-p+1 \tag{14}$$

and $$f''(n,i) = y''(n-r-i+1),\ 1\leq i\leq s-r+1 \tag{15}$$

and to ensure that the base does not penetrate the breakpoints following the next advance, the following constraints must apply:

$$h_1(n+1) + \{h_2(n-1)-h_1(n+1)\}\{(p+i-2)W_a-W_p\}\ /W_b\leq f'(n+1,i),$$
$$1 \leq i \leq q - p + 1 \tag{16}$$

and

$$h_1(n+1)+ \{h_2(n+1)-h_1(n+1)\}\ \{(r+i)W_a-W_p-W_d\}\ /W_b\geq f''(n+1,i)$$
$$1 \leq i \leq s - r + 1 \tag{17}$$

These are readily deduced from the geometry of Figs. 5 and 6 by comparing breakpoint heights with base heights, recalling that tilt

$$\alpha(n+1) = \{h_1(n+1) - h_2(n+1)\}\ /W_b \tag{18}$$

Two additional constraints must also apply: namely that the ends of the base must not penetrate the cut surface either. We must therefore establish the horizontal location of the base ends with respect to the sequence $y'(i)$ and $y''(i)$ and the possible floor heights at these points in terms of $y'(i)$ and $y''(i)$.

Height constraints at the base-ends

As Figs 5 (a) and (b) show the front toe of the base may lie in the overlap or nonoverlap region depending on the fixed geometry of the system. The front toe length W_{TF} may be defined as the extent to which the base front overlaps the leading breakpoint ordinate f' or f'' respectively and from Fig.5 is readily deduced to be:

$$W_{TF} = (p-1)W_a-W_p \quad , \quad p = r \tag{19}$$

or $$W_{TF} = rW_a-W_p-W_o \quad , \quad p=r+1 \tag{20}$$

Similarly the rear toe length W_{TB} may be deduced from Fig.6, and found to be given by

$$W_{TB} = W_p+W_o+W_b-sW_a \quad , \quad s=q \tag{21}$$

or $$W_{TB} = W_a+W_p+W_b-qW_a \quad , \quad s=q-1 \tag{22}$$

Now in the regions of drum overlap, two possibilities exist for the height of the cut floor depending on whether the drum cuts lowest (at the point in question) during its first or second occupation of the overlap region. Obviously the base ends need only clear or contact the lowest of these two possible ordinates to avoid their penetration of the cut floor. In the nonoverlap region only one possibility exists since the drum cuts only once in this region. Careful consideration of the system geometry therefore shows that, if $f_{df}(n-r)$ is the cut floor height at the location of $h_1(n-r)$ then

$$f_{df}(n-r) = \text{Min}\ \left[y'(n)+\{(r-1)W_a-W_p\}\alpha(n) \text{ and } y'(n+1)+((r-2)W_a-W_p)\alpha(n+1)\right],\ p=r \tag{23}$$

or

$$f_{df}(n-r) = y'(n)+\{(r-1)W_a-W_p\}\alpha(n),\ p=r+1) \tag{24}$$

Similar considerations applied to the tail end of the base show that, if $f_{dr}(n-s)$ denotes the cut floor height at the location of $h_2(n-s)$ then

$$f_{dr}(n-s) = \text{Min}\ \left[y'(n)+\{(s-1)W_a-W_p-W_b\}\alpha(n) \text{ and } y'(n-1)+\{sW_a-W_p-W_b\}\alpha(n-1)\right],\ q=s \tag{25}$$

or

$$f_{dr}(n-s) = y'(n-1)+\{sW_a-W_p-W_b\}\alpha(n-1),\ q=s+1 \tag{26}$$

Thus the two additional constraints on the base height during cut n+1 are

$$h_1(n+1) \geq f_{df}(n+1) \tag{27}$$

$$\text{and}\quad h_2(n+1) \geq f_{dr}(n+1) \tag{28}$$

Fitting the Base

The base will settle to a position of minimum potential energy on the cut floor beneath it and thus, if W_g is the distance of the centre-of-gravity of the system, measured back from the leading toe of the base, the potential energy function

$$E(n+1) = h_1(n+1) + \{h_2(n+1) - h_1(n+1)\}W_g/W_b \tag{29}$$

will be minimised in the fitting process by automatic adjustment of base-end-heights $h_1(n+1)$, $h_2(n+1)$, subject to the hard constraints imposed by conditions (16), (17), (27) and (28). These constraints clearly total q-p+s-r+4 in number.

Control Law

Automatic control is conventionally based on feedback measurements from a roof coal thickness-sensor, a tilt transducer (to provide derivative action) and a roof height sensor which measures any difference between the cut roof height and the base-height (both projected to a point beneath the thickness sensor). {The purpose of the roof-height follower is primarily to detect any deviation between the base and the cut floor beneath arising from the presence of fine coal left behind by the cutting drum, upon which the base may climb. Other factors can also cause such a deviation of course, not least the high spots produced earlier by the drum itself, and upon which the base now rests.}

Whilst the drum is making cut number n (i.e. whilst producing a drum height y(n) at the face side) the coal thickness $y_c(n)$ signal used may be one or more passes out of date depending on the sensor location. The control equations are therefore

$$J(n) = k_h \{y_{ref} - y_c(n)\} - k_g \{h_1(n) - h_2(n)\} + k_r [y(n-1) - h_1(n) - W_p\{h_1(n) - h_2(n)\}/(W_p+W_b)] \quad (43)$$

where k_h, k_g and k_r are the thickness, tilt and roof-height gain settings, y_{ref} is the desired drum height and y(n-1) is obtained from previous base height, tilt and control values using equations (1) and (2) whilst

$$y_c(n) = y'(n-cs) \;,\; cs=1,2,3 \text{ etc.} \quad (44)$$

the integer cs defining the prespecified sensor location shown in Fig.1., i.e.

$$W_S = (cs-1)\, W_a \quad (45)$$

The reader will find most of the foregoing equations cross-referenced in the program listing of Section 6. A flowchart of the program is given in Fig.7.

3 Specimen Results

Fig.8 shows the computed response of an automatic system having

W_d = 22	W_s = 0 +	
W_b = 48	height gain k_h = 0.5	
W_p = 37	tilt gain k_g = 1.0	
W_a = 15	integral gain k_i = 0.0	
W_g = 24	roof follower gain k_r = 0.0	

The relatively large ratio: W_p/W_d is clearly more appropriate to tunnelling machines than to shearers. The system clearly behaves in a stable manner and takes 25 advances to correct an initial height error by 80%.

Fig.9 shows the response of a coal-face system, again on automatic control, with the parameters:

W_d = 68	W_s = 0 +
W_b = 96	k_h = 0.5
W_p = 22	k_g = 1.0
W_a = 40	k_i = 0.0
W_g = 30	k_r = 0.0

(It is the ratios of the dimensions that dictates the performance of the system, of course, and here the W_d/W_b ratio is larger than used previously whilst the ratios W_p/W_d and W_p/W_b are much smaller, though still significant). In this case the system is unsteerable, both height and tilt going progressively out of control despite the actions of the ranging drum. The trouble appears to be the failure of the base to reach the upward steps created by the upward ranging drum. Increasing W_a to, say, 60 does not cure the problem.

Fig.10 shows the behaviour of the system as for Fig.9 but with pick-to-pan distance W_p now reduced from 22 to 10. Control is now achieved as can be seen.

Fig.11 shows the effect of introducing the roof-follower (i.e. setting k_r=1.0) to control the system of Fig.9. The pick-to-pan distance is left set at 22. The system remains uncontrollable despite the introduction of this additional control device.

No obvious pattern for steerable geometry has yet emerged. In Fig.8 for instance, a very large W_p/W_b ratio is steerable as is the much smaller value for Fig.10 whilst the intermediate value for Fig.9 (and 11) fails. For the moment, therefore, it would appear that each new geometry should be assessed on its own merits. There are no safe rules-of-thumb as yet.

4. List of Principal Symbols

$\alpha(n)$	=	tilt of base (in radians) during cut n
b_1, b_2; b_{i1}, b_{i2}, b_{i3}; b_{j1}, b_{j2}, b_{j3}	=	coefficients in Linear programming constraints.
b_{o1}, b_{o2}	=	objective function coefficients for Linear programming.
cs	=	integer $= W_s/W_a^{+1}$ defining coal sensor location
$f'(n,i)$	=	height of ith breakpoint associated with y' sequence beneath base during cut n.
$f''(n,i)$	=	height of ith breakpoint associated with y'' sequence beneath base during cut n.
$f_{df}(n)$	=	height of cut floor beneath front of base during cut n
$f_{dr}(n)$	=	height of cut floor beneath rear of base during cut n.
$h_1(n)$	=	height of front of base during cut n.
$h_2(n)$	=	height of rear of base during cut n
$J(n)$	=	drum deflection (jack extension) applied during cut n
k_g	=	tilt gain of auto control system
k_h	=	coal thickness gain of auto control system
k_r	=	roof-follower gain of auto control system
n	=	cut number
p	=	integer minimum number of advances for base to straddle $y'(n)$
q	=	integer number of advances before rear of base leaves $y'(n)$
r	=	integer minimum number of advances for base to straddle $y''(n)$

s	=	integer number of advances before rear of base leaves y''(n)
v(p)	=	storage away for y' sequence before reached by advancing base
w(r)	=	storage array for y'' sequence before reached by advancing base
W_a	=	advance distance
W_b	=	base width
W_d	=	drum width
W_g	=	distance between front of base and system's centre of gravity
W_o	=	overlap distance
W_p	=	spacing between drum and base front
W_s	=	distance of coal-sensor from rear edge of drum
W_{TF}	=	front toe length
W_{TB}	=	back toe length
$x_{(r)}$	=	storage array for cut floor ordinates beneath front of base (on arrival)
x_1, x_2	=	Variables in Linear programming
y(n)	=	height of front of drum on cut n
y'(n)	=	height of floor cut by drun on cut n at leading overlap point
y''(n)	=	height of floor cut by drum on cut n at trailing overlap point
y'''(n)	=	height of rear of drum on cut n
y_{ref}	=	reference drum height for auto control system
z(r)	=	storage array for cut floor ordinates beneath rear of base (on arrival)
z	=	objective function in Linear programming

5. References

(1) Künzi, H.P., Tzschack, H.G. and Zehnder, C.A. "Numerical methods of mathematical optimisation". Academic Press, London 1968, 170 pp.

(2) Michell, G.H. "Operational research" English Universities Press, London, 1972, 275 pp.

6. Program Listing

26 - FEB - 1986

L1

```
C                              SBS.FTN
C                              *******
C
C               ************************************
C               *                                  *
C               *  SOLID BASED STRUCTURES PROGRAM  *
C               *                                  *
C               ************************************
C
C               ******** WITH MANUAL OPTION ********
C
C        This listing is best understood by reference to University
C        of Sheffield Control Eng. Research Report No.  291  Feb 1986
C
C        Equation numbers stated on this listing are those in the report
C
C        The Program uses a DUOPLEX Linear Programming (L. P.) routine
C        for fitting the base to the stored Cut Floor high spots.
C        The SOURCE routine is NOT included, and therefore not
C        documented here.
C
C        The variables Xn are the integer representations of variables Wn
C        (n = A, B, D, G, etc)
C
C
C       List of Varaiable Names:-
C       *************************
C
C       AGN     is        Stores user reply to RUN program again
C       ANS     is        Stores user reply to a prompt
C       CNST    is        Constant added to heights to avoid
C                         negative heights (for L. P. purposes)
C       CODE    is        Code for Manual option to indicate
C                         whether or not JD has entered in a pass
C       CS      is        Coal Sensor position
C       FD1D    is        Final value of drum-rear height
C       FD2D    is        Final value of drum-front height
C       FDF     is        Cut floor height beneath front edge of base
C       FDR     is        Cut floor height beneath rear edge of base
C       JD      is        Drum deflection
C       H1      is        Optimized height of front-edge of the base
C       H2      is        Optimized height of rear-edge of the base
C       KG      is        Tilt gain
C       KH      is        Height gain
C       KR      is        Roof sensor gain
C       KYREF   is        Integer of reference horizon to be reached
C       M       is        Total number of the constraints
C       M1      is        Number of constraints of type 1
C       M2      is        Number of constraints of type 2
C       MPSS    is        Maximum pass number
C       NPSS    is        Current pass number
C       NSTR    is        No. of machine pictures saved at each run
C       OPTN    is        Auto or Manual switch
C       P       is        Number of advances needed for the base-front to
C                         reach the current Y1D
C       Q       is        Number of advances needed for the base-end to
C                         reach the current Y1D
C       QRY     is        Stores user reply to change of structure query
C       R       is        Number of advances needed for the base-front to
C                         reach the current Y2D
C       RSLT    is        Stores user reply to save results in a file
C       S       is        Number of advances needed for the base-end to
C                         reach the current Y2D
C       SAV     is        Stores user reply to save/quit/final question
```

i2 -

```
C       SXB     is          Integer of saved base X-coordinate
C       SXD     is          Integer of saved drum X-coordinate
C       SZ      is          Size of constraint tableau used by L. P.
C       XA      is          Advance distance
C       XB      is          Base length
C       XCL     is          X-coordinate of front edge of the base
C       XCR     is          X-coordinate of front edge of the drum
C       XD      is          Drum length
C       XG      is          Position of C. of G. from base front edge
C       XO      is          Overlap distance
C       XOFST   is          Offset to be added to all X-coordinates
C       XSB     is          X-coordinate of base
C       XSD     is          X-coordinate of drum
C       Y0D     is          Height of front-edge of drum (for drawing)
C       Y1D     is          Height of rear of the drum
C       Y2D     is          Height of front of the drum
C       Y3D     is          Height of rear-edge  of drum (for drawing)
C       YBF     is          Potential Heights beneath front edge of base
C       YBR     is          Potential Heights Beneath rear edge of base
C       YG      is          Height of centre of gravity
C       YOFST   is          Offset to be added to all Y-coordinates
C       YREF    is          reference horizon to be reached
C
C
C
C
C
C
C
C       List of Array Name:-
C       ********************
C
C       A       is          Constraint tableau in the form which
C                           L. P. takes
C       B       is          Tableau of all constraints
C       F1D     is          Storage for drum rear heights
C       F2D     is          Storage for drum front heights
C       L1      is          Internal array of the L. P.
C       L2      is          Internal array of the L. P.
C       L3      is          Internal array of the L. P.
C       LP1     is          Internal array of the L. P.
C       LP2     is          Internal array of the L. P.
C       MOD     is          Mode for different cases of cut floor
C       SFD1D   is          Storage for FD1D
C       SFD2D   is          Storgae for FD2D
C       SPSS    is          Storage for pass numbers of saved machines
C       SY      is          Storage for heights of saved machines
C       V       is          Storage For  Y1D
C       W       is          Storage For  Y2D
C       WE      is          Storage for overlap distances (for drawing)
C       X       is          Storage For  YBF
C       XX      is          Optimized heights returned from L. P.
C       Y       is          Storage for height beneath drum (for drawing)
C       YC      is          Storage for coal sensor signal
C       Z       is          Storage For  YBR
C
C
C
C
        PROGRAM SBS
C
        BYTE YES,NO
        BYTE AGN,OPTN,CODE,AUTO,MANU,SAV,RSLT,QRY,ANS,FINL,QUIT
C
        REAL V(10),W(10),X(10),Z(10),F1D(10),F2D(10),YC(10),JD
        REAL KH,KG,KR,A(300),B(100,3),XX(3),SY(4,200)
```

```
      REAL SFD1D(200),SFD2D(200),Y(5,200),WE(200)
C
      INTEGER LP1(3),LP2(100),L1(3),L2(100),L3(100),YOFST,XOFST
      INTEGER XD,XB,XA,XO,XU,XSB,XSD,SXB,SXD,XG,YG
      INTEGER MOD(200),P,Q,R,S,CNST,SZ,CS,SPSS(200)
C
      COMMON/AREA 1/V,W,X,Z,F1D,F2D,FD1D,FD2D,Y,Y0D,Y3D
      COMMON/AREA 2/H1,H2,WD,WB,WP,WA,WO,WU
      COMMON/AREA 3/BETA,Y1D,Y2D,JD,WE,MOD,P,Q,R,S,NPSS,FDF,FDR
      COMMON/AREA 4/YC,SFD1D,SFD2D,WG,CS,YREF,KH,KG,KR,MPSS
      COMMON/AREA 5/AGN,RSLT,OPTN,YES,AUTO,MANU
      COMMON/AREA 6/YOFST,XOFST
C
      DATA AUTO,MANU,FINL,QUIT //'A','M','F','Q'/
      DATA YES,NO /'Y','N'/
C
      CALL TKINIT
      AGN=NO
      YOFST=400
      XOFST=50
 100  CNST=100
      NPSS=0
      NSTR=0
      JD=0.0
      SAV=NO
      CALL ERASE
      CALL HEADER
C
C     INITLZ is parameter entry and parameter calculation routine
C     (Listed below)
C
      CALL INITLZ
C
C     FIND VALUES OF P, Q, R & S  equations 10 to 13
C     and convert to INTEGER
C
      P=0
180   IF((P-((WP+WA)/WA)).GT.0) GOTO 185
         P=P+1
         GOTO 180
185   R=0
190   IF((R-((WP+WO)/WA)).GT.0) GOTO 195
         R=R+1
         GOTO 190
195   Q=0
200   IF((Q-((WP+WB)/WA)).GT.0) GOTO 205
         Q=Q+1
         GOTO 200
205   S=0
210   IF((S-((WP+WB+WD-2*WA)/WA)).GT.0) GOTO 215
         S=S+1
         GOTO 210
215   NQ=Q+1
      NS=S+1
C
      XCR=XOFST+WB+WP+WD
C
C     FIND VALUES OF base front and rear Toe Length WTF, WTR
C     equation 19 to 22
C
      WTF=R*WA-WP-WO
      IF(P.EQ.R) WTF=(P-1)*WA-WP
      WTB=WO+WP+WB-Q*WA
      IF(S.EQ.Q) WTB=WP+WB+WO-S*WA
C
C     Initialize base front and rear heights and stored intermediate
```

```
C       heights
C
        FDF=X(R)
        FDR=Z(Q)
        SFD1D(NQ)=V(P-1)
        SFD2D(NS)=W(R-1)
C
C       Scale vertical display heights
C
        FDMX=SFD1D(NQ)
        N=2
C
C       For successive passes the program repeats from statement 220
C
220     IF(SFD1D(NQ).GE.FDMX) FDMX=SFD1D(NQ)
        IF(SFD2D(NS).GE.FDMX) FDMX=SFD2D(NS)
        IF(H1.GT.FDMX) FDMX=H1
        IF(H2.GT.FDMX) FDMX=H2
C
C       Find Y and X scaling factors
C
        YSCF=150.0/FDMX                     ! 150 max value of heights
        XSCF=950.0/XCR                      ! 950 max value of advance
C
C       SCALE ALL THE M/C'S GEOMETRY horizontally
C
        XB=INT(XSCF*WB+0.5)
        XD=INT(XSCF*WD+0.5)
        XA=INT(XSCF*WA+0.5)
        XO=INT(XSCF*WO+0.5)
        IF((XA+XO).NE.XD) XO=XD-XA
        XU=INT(XSCF*WU+0.5)
        IF((XO+XU).NE.XA) XU=XA-XO
C
C       PREPARE THE CONSTRAINT TABLEAU FOR LINEAR PROGRAM (DUOPLEX)
C
C
C       Find parameters of OBJECTIVE FUNCTION 34 using equations 41,42
C
        I=1
        B(I,1)=0.0
        COFF=WG/WB
        B(I,2)=-(1-COFF)
        B(I,3)=-COFF
C
C       Find COFFICIENTS OF CONSTRAINTS EQUATION 30 using equation 37
C
        DO 230 J=(I+1),(I+Q-P+1)
        B(J,1)=-(CNST+F1D(J-I))
        COFF=((P+J-I-2)*WA-WP)/WB
        B(J,2)=1-COFF
230     B(J,3)=COFF
C
C       Find COFFICIENTS OF CONSTRAINTS EQUATION 31 using equation 38
C
        I=I+Q-P+1
        DO 240 J=(I+1),(I+S-R+1)
        B(J,1)=-(CNST+F2D(J-I))
        COFF=((R+J-I)*WA-WP-WD)/WB
        B(J,2)=1-COFF
240     B(J,3)=COFF
C
C       Find COFFICIENTS OF CONSTRAINTS EQUATION 32 using equation 39
C
        I=(I+S-R+1)+1
        B(I,1)=-(CNST+FDF)
```

```
      B(I,2)=1.0
      B(I,3)=0.0
C
C     Find COFFICIENTS OF CONSTRAINTS EQUATION 34 using equation 40
C
      I=I+1
      B(I,1)=-(CNST+FDR)
      B(I,2)=0.0
      B(I,3)=1.0
      M1=I-1
      M2=0
      M=M1+M2
C
C     AUXILARY OBJECTIVE FUNCTION (STORAGE SPACE)
C
      I=I+1
      DO 250 J=1,3
250   B(I,J)=0.0
C
C     SIZE OF  A  TABLEAU
C
      SZ=(M+2)*(N+1)
C
C     PREPARE  A  TABLEAU FROM  B  ARRAY
C     IZSCHR =(N+1) & ISSCHR=1 MEANS THAT  A  TABLEAU IS
C     TAKEN ROW BY ROW FROM  B  ARRAY
C
      DO 260 I=1,(M+2)
      DO 260 J=1,(N+1)
      K=(I-1)*(N+1)+J
      A(K)=B(I,J)
260   CONTINUE
C
C     Instructions down to 350 are for Screen communication and
C     plotting (Bypasses on auto and final pass demanded)
C
      XCL=XCR-WD-WP
      IF(OPTN.EQ.AUTO.AND.SAV.EQ.FINL.AND.NPSS.LT.MPSS) GOTO 350
      CALL ERASE
      CALL HEADER
C
C     CALCULATE X-COORDINATES OF BEAM, DRUM AND
C     COORDINATE OF CENTER OF GRAVITY
C
      XSB=INT(XSCF*XCL+0.5)+XOFST
      XSD=INT(XSCF*XCR+0.5)+XOFST
      XG=XSB-INT(XSCF*(WG/COS(BETA))+0.5)
      YG=INT(YSCF*H1+0.5)+YOFST-INT(YSCF*(WG*(SIN(BETA)/COS(BETA))))
C
C     Plot floor heights
C
      CALL HGHTS(0,XCR,SFD1D,NQ,XA,XO,XSCF,YSCF)
      CALL HGHTS(1,XCR,SFD2D,NS,XA,XA,XSCF,YSCF)
C
C     Plot floor profile
C
      CALL FLOOR(MOD,Y,(NPSS+1),XCR,XD,XO,XA,XU,WE,XSCF,YSCF)
C
C     Draw shape of base and drum
C
      CALL SHAPE(1,XSB,H1,XB,H2,10,XSCF,YSCF)
      CALL SHAPE(0,XSD,Y0D,XD,Y3D,25,XSCF,YSCF)
      KYREF=INT(YSCF*YREF+0.5)+YOFST
C
C     Plot reference line
C
```

```
      CALL TPLOT(0,0,KYREF)
      CALL TPLOT(1,1023,KYREF)          ! 1023 IS MAX. X-AXIS VAUE
C
C     Mark the centre of gravity of the base
C
      CALL MARKER(3,XG,(YG+5))
C
C     DRAW THE SAVED M/C AND DRUMS WHICH HAVE SELECTED PREVIOUSLY
C
      IF(NSTR.NE.0.AND.(SAV.EQ.FINL.AND.NPSS.EQ.MPSS)) GOTO 275
C
C     OTHERWISE WRITE PASS NO AND INFORMATIONS
C
      GOTO 282
275     DO 280 I=1,NSTR
        SXB=XSB-(NPSS-SPSS(I))*XA
        SXD=XSD-(NPSS-SPSS(I))*XA
        CALL SHAPE(1,SXB,SY(1,I),XB,SY(2,I),10,XSCF,YSCF)
280     CALL SHAPE(0,SXD,SY(3,I),XD,SY(4,I),25,XSCF,YSCF)
282   CALL TPLOT(0,400,700)
      CALL ANMODE
      WRITE(6,283)NPSS
283   FORMAT(T54,' PASS NO. = ',I4)
      CALL TPLOT(0,800,700)
      CALL ANMODE
      WRITE(6,290)OPTN
290   FORMAT(T15,' OPTION = ',A1)
      CALL TPLOT(0,0,270)
      CALL ANMODE
      WRITE(6,901)KH,KG,KR,CS
      WRITE(6,902)P,Q,R,S
      WRITE(6,903)WD,WB,WA,WO
      WRITE(6,904)WG,WP,WTF,WTB
      WRITE(6,913)NQ,NS,JD
C
C     ON MANUAL OPTION READ IN THE DRUM DEFLECTION AND
C     SET CODE = NO (I.E. FALSE) INDICATING THAT VALUE
C     OF THE DRUM DEFLECTION IS READ IN ON THIS PASS
C
      IF(OPTN.NE.AUTO.AND.CODE.EQ.YES) GOTO 292
      GOTO 342
292     CODE=NO
        MANU=YES
        CALL TPLOT(0,1,XOFST)
        CALL ANMODE
        WRITE(6,295)
295     FORMAT(' ENTER DRUM DEFLECTION < JD >')
        READ(6,919)JD
C
C       Shift all storage arrays to right by one value, to avoid
C       shifting left twice (on manual picture is drawn twice)
C
        DO 300 J=1,P-1
300     V(J)=V(J+1)
        DO 310 J=1,R-1
        W(J)=W(J+1)
310     X(J)=X(J+1)
        DO 320 J=1,Q-1
320     Z(J)=Z(J+1)
        DO 330 J=1,Q-P
330     F1D(J)=F1D(J+1)
        DO 340 J=1,S-R
340     F2D(J)=F2D(J+1)
        GOTO 370
342    IF(NPSS.NE.0) GOTO 346
C
```

```
C       INITIALLY ASK THE QUESTION OF CHANGE OF STRUCTURE
C       START FROM TOP OF PROGRAM IF USER WANTS TO CHANGE          L7
C
          CALL TPLOT(0,1,XOFST)
          CALL ANMODE
          WRITE(6,344)
344       FORMAT(' ENTER  Y  TO CHANGE THE STRUCTURE')
          READ(6,900)QRY
          IF(QRY.EQ.YES) GOTO 100
C
C       WRITE ALL THE NECCESSARY PARAMETERS AND ARRAY TO FILE
C       JUST FOR PRINTING AND TESTING PURPOSES
C       (FILE IS SET TO SCREEN)
C
346     IF(RSLT.EQ.YES.AND.NPSS.EQ.0) GOTO 348
        GOTO 350
C
C       Write parameters at start each run only
C
348       WRITE(6,901)KH,KG,KR,CS
          WRITE(6,902)P,Q,R,S
          WRITE(6,903)WD,WB,WA,WO
          WRITE(6,904)WG,WP,WTF,WTB
C
C       RSLT = YES allows results to go to a file
C       future adaptation by BRETBY
C
350     IF(RSLT.NE.YES) GOTO 352
           WRITE(6,905)NPSS,JD
           WRITE(6,906)(V(I),I=1,10)
           WRITE(6,907)(W(I),I=1,10)
           WRITE(6,908)(X(I),I=1,10)
           WRITE(6,909)(Z(I),I=1,10)
           WRITE(6,910)(F1D(I),I=1,10)
           WRITE(6,911)(F2D(I),I=1,10)
           WRITE(6,912)FDF,FDR,H1,H2
C
C       CARRY ON IF THE SELECTED MAX PASS NUMBER IS NOT REACHED
C
352     IF(NPSS.GE.MPSS) GOTO 450
C
C       Fit base to cut floor heights (by Linear programming)
C       for next pass
C
        CALL DUOPLX(A,SZ,N,M1,M2,(N+1),1,IFAIL,
     & LP1,N,LP2,M,L1,N,L2,M,L3,M,XX)
        IF(IFAIL.NE.0) GOTO 990
        DO 355 J=1,3
355     XX(J)=XX(J)-CNST
C
C       User communication for next instructions
C
        IF(SAV.NE.FINL.AND.NPSS.NE.0) GOTO 360
        GOTO 364
360       CALL TPLOT(0,1,105)
          CALL ANMODE
          WRITE(6,361)
          IF(OPTN.EQ.AUTO) WRITE(6,362)
C
C         Request whether to (SAV)E present machine display for
C         output later
C
          WRITE(6,363)
          READ(6,900)SAV
C
C       Do NOT accept final picture command on MANUAL option
```

L8

```
C
364     IF(OPTN.NE.AUTO.AND.SAV.EQ.FINL) SAV=NO
        IF(SAV.EQ.QUIT) GOTO 460
C
C       Save current machine heights and pass number if requested
C
        IF(SAV.NE.YES) GOTO 365
          NSTR=NSTR+1
          SPSS(NSTR)=NPSS
          SY(1,NSTR)=H1
          SY(2,NSTR)=H2
          SY(3,NSTR)=Y0D
          SY(4,NSTR)=Y3D
C
C       Load output of Linear Program into new base-end heights
C
365     H1=XX(1)
        H2=XX(2)
        NPSS=NPSS+1
C
C       Calculate AUTO control if selected using equation 43
C
        IF(OPTN.NE.AUTO) GOTO 366
           JD= KH*(YREF-YC(CS))-KG*(H1-H2)
     &         +KR*(YC(CS)-H1-((WP*(H1-H2))/(WP+WB)))
        GOTO 370
366     CODE=YES
C
C       UPDATE routine contains the floor cutting and shifting
C       equations (listed below)
C
370     CALL UPDATE
C
C       On AUTO shift coal sensor signal and load with
C       new value equation 44
C
        IF(OPTN.EQ.AUTO.AND.CS.GT.1) GOTO 375
        GOTO 442
375        DO 440 I=1,CS-1
440        YC(CS+1-I)=YC(CS-I)
442     YC(1)=Y1D
C
C       On MANUAL option re-draw the picture with new
C       entered drum deflection
C
        IF(MANU.NE.YES) GOTO 444
          MANU=NO
          SFD1D(NQ)=Y1D
          SFD2D(NS)=Y2D
          GOTO 220
C
C       increment to next pass, store front and rear of the drum
C       jump back for display and base fitting to high spots just
C       calculated in the update routine
C
444     NQ=NQ+1
        NS=NS+1
        SFD1D(NQ)=Y1D
        SFD2D(NS)=Y2D
        XCR=XCR+WA
        GOTO 220
C
C       Terminate if requested, otherwise jump to 100 to start again
C
450     CALL TPLOT(0,1,30)
        CALL ANMODE
```

L9

```
      READ(6,900)ANS
460   CALL ERASE
      WRITE(6,462)
462   FORMAT(' ENTER  Y  TO RUN THE PROGRAM AGAIN')
      READ(6,900)AGN
      IF(AGN.EQ.YES) GOTO 100
      GOTO 999
361   FORMAT(' ENTER  Y  TO SAVE THE CURRENT M/C')
362   FORMAT('         F  FOR FINAL PASS (I.E. ENTERED MAX PASS)')
363   FORMAT('         Q  TO QUIT THE CURRENT SIMULATION')
900   FORMAT(A1)
901   FORMAT('  KH=',F9.3,3X,' KG=',F9.3,3X,' KR=',F9.3,3X,' SP=',I9)
902   FORMAT('   P=',I9,3X,'  Q=',I9,3X,'  R=',I9,3X,'  S=',I9)
903   FORMAT('  WD=',F9.3,3X,' WB=',F9.3,3X,' WA=',F9.3,3X,' WO=',F9.3)
904   FORMAT('  WG=',F9.3,3X,' WP=',F9.3,3X,'WTF=',F9.3,3X,'WTB=',F9.3)
905   FORMAT(' PASS NO.=',I3,'  JD=',F6.2)
906   FORMAT('  V   ',10(1X,F6.2))
907   FORMAT('  W   ',10(1X,F6.2))
908   FORMAT('  X   ',10(1X,F6.2))
909   FORMAT('  Z   ',10(1X,F6.2))
910   FORMAT(' F1D  ',10(1X,F6.2))
911   FORMAT(' F2D  ',10(1X,F6.2))
912   FORMAT(' FDF=',F6.2,'  FDR=',F6.2,'  H1=',F6.2,'  H2=',F6.2)
913   FORMAT('  NQ=',I9,3X,' NS=',I9,3X,' JD=',F9.3)
919   FORMAT(F6.2)
990   WRITE(6,991)
991   FORMAT(' ERROR HAS OCCURED IN LINEAR PROG.')
C
C     Re-initialize screen on exit
C
999   CALL ANSINT
      CALL EXIT
      END
C
C
C     INITLZ  is used for parameter entry and parameter initialization
C
C
      SUBROUTINE INITLZ
      BYTE YES,NO
      BYTE AGN,OPTN,AUTO,MANU,RPLY,RPT,RSLT
      REAL V(10),W(10),X(10),Z(10),F1D(10),F2D(10),JD
      REAL KH,KG,KR,Y(5,200),WE(200),YC(10),SFD1D(200)
      REAL SFD2D(200)
      INTEGER P,Q,R,S,CS
      COMMON/AREA 1/V,W,X,Z,F1D,F2D,FD1D,FD2D,Y,Y0D,Y3D
      COMMON/AREA 2/H1,H2,WD,WB,WP,WA,WO,WU
      COMMON/AREA 4/YC,SFD1D,SFD2D,WG,CS,YREF,KH,KG,KR,MPSS
      COMMON/AREA 5/AGN,RSLT,OPTN,YES,AUTO,MANU
      BETA=0.0
      IF(AGN.NE.YES) GOTO 102
        WRITE(6,101)
101     FORMAT(' KEEPING THE EXISTING STRUCTURE ? (Y/N)')
        READ(6,900)RPLY
        IF(RPLY.EQ.YES) GOTO 120
102   WRITE(6,103)
103   FORMAT(' ENTER  Y  FOR INITIAL VALUES FROM KEYBOARD')
      READ(6,900)RPT
      IF(RPT.NE.YES) GOTO 120
         WRITE(6,105)
105      FORMAT(' ENTER I.V. FOR  V, W, X, Z, F1D, F2D & YC')
         DO 110 I=1,10
110      READ(6,919)V(I),W(I),X(I),Z(I),F1D(I),F2D(I),YC(I)
      GOTO 142
C
C     Initialize all storage arrays and machine heights
```

L10

```
C
120       DO 130 I=1,10
          V(I)=10.0
          W(I)=10.0
          X(I)=10.0
          Z(I)=10.0
          F1D(I)=10.0
          F2D(I)=10.0
          YC(I)=10.0
          SFD1D(I)=10.0
130       SFD2D(I)=10.0
          DO 140 I=1,4
          DO 140 J=1,10
          Y(I,J)=10.0
140       CONTINUE
          Y0D=10.0
          Y3D=10.0
          FD1D=10.0
          FD2D=10.0
          H1=10.0
          H2=10.0
          IF(RPLY.EQ.YES) GOTO 170
142    RSLT=NO
C142    WRITE(6,143)
C143    FORMAT(' ENTER  Y  TO PRINT RESULTS IN A FILE')
C       READ(6,900)RSLT
       WRITE(6,144)
144    FORMAT(' ENTER DRUM  &  BASE  WIDTH < WD & WB >')
       READ(6,919)WD,WB
       WRITE(6,145)
145    FORMAT(' ENTER PICK TO PAN DISTANCE < WP >')
       READ(6,919)WP
150    WRITE(6,151)
151    FORMAT(' ENTER ADVANCE DISTANCE < WA >')
       READ(6,919)WA
       IF(WA.LE.WD) GOTO 155
          WRITE(6,152)
152       FORMAT(' NOT POSSIBLE, CAN''T ADVANCE MORE THAN WD')
          GOTO 150
C
C      equation 5
C
155    WO=WD-WA
160    WRITE(6,161)
161    FORMAT(' ENTER  C OF G   OF BASE   < WG >')
       READ(6,919)WG
       IF(WG.GE.0.AND.WG.LE.WB) GOTO 165
          WRITE(6,162)
162       FORMAT(' NOT POSSIBLE, C OF G SHOULD BE ON BASE')
          GOTO 160
165    WRITE(6,166)
166    FORMAT(' ENTER COAL SENSOR POSITION   &    YREF')
       READ(6,920)CS
       READ(6,919)YREF
       IF(CS.GT.0) GOTO 170
         WRITE(6,167)
167      FORMAT(' COAL SENSOR POS. SHOULD>0 SO IT TAKEN AS 1')
         CS=1
170    WRITE(6,171)
171    FORMAT(' ENTER   A  FOR  AUTO')
       READ(6,900)OPTN
       IF(OPTN.NE.AUTO) GOTO 174
          WRITE(6,172)
172       FORMAT(' KH  &  KG')
          READ(6,919)KH,KG
          WRITE(6,173)
```

L11

```
173         FORMAT(' ENTER ROOF SENSOR GAIN  < KR >')
            READ(6,919)KR
            GOTO 175
174         OPTN=MANU
175      WRITE(6,176)
176      FORMAT(' ENTER MAX. PASS(ES) NO.')
         READ(6,920)MPSS
         WU=WA-WO
         RETURN
900      FORMAT(A1)
919      FORMAT(F8.2)
920      FORMAT(I4)
         END
C
C
C        UPDATE routine is used for calculating floor cutting heights
C        and shifting equations
C
C
         SUBROUTINE UPDATE
         REAL V(10),W(10),X(10),Z(10),F1D(10),F2D(10),JD
         REAL KH,KG,KR,Y(5,200),WE(200)
         INTEGER MOD(200),P,Q,R,S
         COMMON/AREA 1/V,W,X,Z,F1D,F2D,FD1D,FD2D,Y,Y0D,Y3D
         COMMON/AREA 2/H1,H2,WD,WB,WP,WA,WO
         COMMON/AREA 3/BETA,Y1D,Y2D,JD,WE,MOD,P,Q,R,S,NPSS,FDF,FDR
C
C        NOTE   BETA used rather than ALPHA in the report equations
C        equation 2
C
         BETA=(H1-H2)/WB
C
C        equation 1
C
         Y0D=H1+(WD+WP)*BETA+JD
C
C        equation 8,9
C
         Y1D=H1+(WP+WA)*BETA+JD
         Y2D=H1+(WP+WO)*BETA+JD
C
C        Y3D needed for display only
C
         Y3D=H1+WP*BETA+JD
C
C        Save above ordinate in plotting arrays, used in the
C        FLOOR routine
C
         Y(1,(NPSS+1))=Y3D
         Y(2,(NPSS+1))=Y2D
         Y(3,(NPSS+1))=Y1D
         Y(4,(NPSS+1))=Y0D
         Y(5,(NPSS+1))=Y1D
         WE(NPSS+1)=0.0
C
C        Identify whether or not extra floor cut in overlap region
C        and any extra break points for plotting and set the CASE
C        number (MOD) accordingly
C
         MOD(NPSS)=1                                               !CASE  1
         IF(Y(3,NPSS).LE.Y(1,(NPSS+1))) MOD(NPSS)=2                !CASE  2
         IF((Y3D.LT.Y(3,NPSS)).AND.(Y2D.GT.Y(4,NPSS))) MOD(NPSS)=3 !CASE  3
         IF((Y3D.GT.Y(3,NPSS)).AND.(Y2D.LT.Y(4,NPSS))) MOD(NPSS)=4 !CASE  4
         IF(MOD(NPSS).EQ.3.OR.MOD(NPSS).EQ.4) GOTO 372
         GOTO 374
372         DY1=Y(3,NPSS)-Y(1,(NPSS+1))                 ! Y1D - Y3D
```

```
            DY2=Y(2,(NPSS+1))-Y(4,NPSS)          ! Y2D - Y0D
            DY3=Y(2,(NPSS+1))-Y(1,(NPSS+1))      ! Y2D - Y3D
            WE(NPSS+1)=(W0*DY1)/(DY1+DY2)
            Y(5,(NPSS+1))=Y(1,(NPSS+1))+WE(NPSS+1)*DY3/W0
374       IF(WE(NPSS+1).GT.W0) WE(NPSS+1)=W0
          IF(WE(NPSS+1).LT.0) WE(NPSS+1)=0
C
C         Calculate potential floor heights (in equation 23-26) beneath
C         front and rear edges of base
C
          YBF1=Y1D+((R-1)*WA-WP)*BETA
          YBF2=Y1D+((R-2)*WA-WP)*BETA
          YBR1=Y1D+((Q-1)*WA-WP-WB)*BETA
          YBR2=Y1D+(Q*WA-WP-WB)*BETA
C
C         Load and shift V-array holding Y1D (i.e. drum rear) ordinates
C         and hold final value in FD1D for entry into F1D-array
C         (equation 23,24)
C
          FD1D=Y1D
          IF(P.EQ.1) GOTO 380
            FD1D=V(P-1)
            DO 380 J=1,P-1
            V(P+1-J)=V(P-J)
380         CONTINUE
          V(1)=Y1D
C
C         Load and shift W-array holding Y2D (i.e. drum front) ordinates
C         and hold final value in FD2D for entry into F2D-array
C         equations 25,26
C
          FD2D=Y2D
          IF(R.EQ.1) GOTO 390
            FD2D=W(R-1)
            DO 390 J=1,R-1
            W(R+1-J)=W(R-J)
390         CONTINUE
          W(1)=Y2D
C
C         Find and shift stored heights beneath front and rear of base
C
          FDF=YBF1
          IF(R.EQ.1) GOTO 402
            IF(P.EQ.R.AND.X(1).GT.YBF2) X(1)=YBF2
            DO 400 J=1,R-1
            X(R+1-J)=X(R-J)
400         CONTINUE
            FDF=X(R)
402       X(1)=YBF1
          FDR=YBR1
          IF(Q.EQ.1) GOTO 410
            FDR=Z(Q)
            IF(Q.NE.S.OR.Z(1).GT.YBR1) Z(1)=YBR1
            DO 410 J=1,Q-1
            Z(Q+1-J)=Z(Q-J)
410         CONTINUE
          Z(1)=YBR2
C
C         Shift and load F1D and F2D arrays with outputs from Y1D and Y2D
C         previously stored in FD1D and FD2D equations 14,15
C
          DO 420 J=1,Q-P
          F1D(Q-P+2-J)=F1D(Q-P+1-J)
420       CONTINUE
          F1D(1)=FD1D
          DO 430 J=1,S-R
```

```
      F2D(S-R+2-J)=F2D(S-R+1-J)                                       L13
430   CONTINUE
      F2D(1)=FD2D
      RETURN
      END
C
C
C     SHAPE routine display shape of the base and drum depending on MOD
C     MOD = 1  displays base
C     MOD = 0  displays drum
C
C
      SUBROUTINE SHAPE(MOD,XX1,YY1,XX2,YY2,HGT,XSCF,YSCF)
      INTEGER Y1,X2,Y2,XX1,XX2,HT,HGT,YOFST,XOFST
      COMMON/AREA 6/YOFST,XOFST
      Y1=INT(YSCF*YY1+0.5)+YOFST
      X2=XX1-XX2
      Y2=INT(YSCF*YY2+0.5)+YOFST
      HT=INT(HGT+0.5)
      CALL TPLOT(0,XX1,Y1)
      CALL TPLOT(1,X2,Y2)
      CALL TPLOT(1,X2,(Y2+HT))
      CALL TPLOT(MOD,XX1,(Y1+HT))
      CALL TPLOT(1,XX1,Y1)
      RETURN
      END
C
C
C     HGHTS routine will display cut floor heights
C     MOD = 0 solid  lines
C     MOD = 1 dotted lines
C
C
      SUBROUTINE HGHTS(MOD,X,Y,N,XA,X1,XSCF,YSCF)
      REAL Y(N)
      INTEGER XA,X1,YOFST,XOFST
      COMMON/AREA 6/YOFST,XOFST
      IX=INT(XSCF*X+0.5)-X1+XOFST
      DO 110 I=N,1,-1
      IY=INT(YSCF*Y(I)+0.5)+YOFST
      CALL TPLOT(0,IX,YOFST)
      IF(MOD.NE.1) GOTO 105
         J=IY-YOFST
         DO 100 K=1,J,5
         M=1
         IF((K/2).EQ.(K/2.0)) M=0
100      CALL TPLOT(M,IX,(K+YOFST))
105   CALL TPLOT(1,IX,IY)
110   IX=IX-XA
      RETURN
      END
C
C
C     FLOOR routine will join the high spots of the cut floors
C     to form the floor profile
C
C
      SUBROUTINE FLOOR(MOD,Y,N,X,XD,XO,XA,XU,WE,XSCF,YSCF)
      INTEGER MOD(N),XO,XA,XE,XD,XU,XX,YY,YOFST,XOFST
      REAL Y(5,N),WE(N)
      COMMON/AREA 6/YOFST,XOFST
      XX=INT(XSCF*X+0.5)-(N*XA+XO)+XOFST
      YY=INT(YSCF*Y(1,1)+0.5)+YOFST
      CALL TPLOT(0,XX,YY)                               !PUT  TO POINT  A(1)
      XX=XX+XO
      YY=INT(YSCF*Y(2,1)+0.5)+YOFST
```

L14

```
      CALL TPLOT(1,XX,YY)                          !DRAW TO POINT  B(1)
      IF(N.LE.1) GOTO 150
      I=2
100   XX=XX+XU
      YY=INT(YSCF*Y(3,(I-1))+0.5)+YOFST
      CALL TPLOT(1,XX,YY)                          !DRAW TO POINT  C(N-1)
      IF(MOD(I-1).EQ.4) GOTO 130
      IF(MOD(I-1).EQ.3) GOTO 120
      IF(MOD(I-1).EQ.2) GOTO 110
      IF(MOD(I-1).NE.1) GOTO 140
        YY=INT(YSCF*Y(1,I)+0.5)+YOFST
        CALL TPLOT(1,XX,YY)                        !DRAW TO POINT  A(N)
        XX=XX+XO
        YY=INT(YSCF*Y(2,I)+0.5)+YOFST
        CALL TPLOT(1,XX,YY)                        !DRAW TO POINT  B(N)
        GOTO 140
C
110     XX=XX+XO
        YY=INT(YSCF*Y(4,(I-1))+0.5)+YOFST
        CALL TPLOT(1,XX,YY)                        !DRAW TO POINT  D(N-1)
        YY=INT(YSCF*Y(2,I)+0.5)+YOFST
        CALL TPLOT(1,XX,YY)                        !DRAW TO POINT  B(N)
        GOTO 140
C
120     YY=INT(YSCF*Y(1,I)+0.5)+YOFST
        CALL TPLOT(1,XX,YY)                        !DRAW TO POINT  A(N)
        XE=INT(XSCF*WE(I)+0.5)
        XX=XX+XE
        YY=INT(YSCF*Y(5,I)+0.5)+YOFST
        CALL TPLOT(1,XX,YY)                        !DRAW TO POINT  E(N)
        XX=XX+(XO-XE)
        YY=INT(YSCF*Y(4,(I-1))+0.5)+YOFST
        CALL TPLOT(1,XX,YY)                        !DRAW TO POINT  D(N-1)
        YY=INT(YSCF*Y(2,I)+0.5)+YOFST
        CALL TPLOT(1,XX,YY)                        !DRAW TO POINT  B(N)
        GOTO 140
C
130     XE=INT(XSCF*WE(I)+0.5)
        XX=XX+XE
        YY=INT(YSCF*Y(5,I)+0.5)+YOFST
        CALL TPLOT(1,XX,YY)                        !DRAW TO POINT  E(N)
        XX=XX+(XO-XE)
        YY=INT(YSCF*Y(2,I)+0.5)+YOFST
        CALL TPLOT(1,XX,YY)                        !DRAW TO POINT  B(N)
C
140   I=I+1
      IF(I.LE.N) GOTO 100
150   XX=XX+XU
      YY=INT(YSCF*Y(3,N)+0.5)+YOFST
      CALL TPLOT(1,XX,YY)                          !DRAW TO POINT C(N)
      XX=XX+XO
      YY=INT(YSCF*Y(4,N)+0.5)+YOFST
      CALL TPLOT(1,XX,YY)                          !DRAW TO POINT D(N)
      RETURN
      END
C
C
C     TPLOT routine will draw or move to specified point on screen
C     MOD = 0   moves to a point
C     MOD = 1   draws to a point
C
C
      SUBROUTINE TPLOT(MOD,IX,IY)
      IF(MOD.EQ.0) CALL MOVABS (IX,IY)
      IF(MOD.EQ.1) CALL DRWABS (IX,IY)
      RETURN
```

```
      END
C
C
C     Initialize screen to graghic
C
C
      SUBROUTINE TKINIT
      INTEGER TPS
      BYTE ESC,K,AA
      DATA ESC,K,AA,TPS /27,'K','A',0/
      CALL INITT(480)
      CALL CHRSIZ(1)
      WRITE(5,100)ESC,ESC                       ! Erase ANSI Screen
100   FORMAT(1H ,A1,'[2J',A1,'%!0')              ! Change to TEXTRONIX mode
      WRITE(5,200)ESC,K,AA,TPS
200   FORMAT(1H ,3A1,I1)
      RETURN
      END
C
C
C     Initialize screen to alphanumeric
C
C
      SUBROUTINE ANSINT
      BYTE ESC,FF
      DATA ESC,FF /27,12/
      WRITE(5,100)ESC,FF,ESC                    ! Clear Graphic Screen
100   FORMAT(1H ,3A1,'%!1')                     ! Change to ANSI mode
      RETURN
      END
C
C
C     Clear both mode of the screen
C
C
      SUBROUTINE ERASE
      BYTE ESC,FF
      DATA ESC,FF /27,12/
      WRITE(5,100)ESC,ESC,FF                    ! Erase ANSI Screen
100   FORMAT(1H ,A1,'[2J',2A1)                  ! Clear Graphic Screen
      RETURN
      END
C
C
C     Mark the centre of the gravity of the base
C
C
      SUBROUTINE MARKER(K,IX,IY)
      INTEGER MM,LH,I,K
      BYTE ESC
      DATA ESC,MM,LH,I /27,'MM','LH',0/
      WRITE(5,100)ESC,MM,K
      CALL TPLOT(0,IX,IY)
      CALL TPLOT(1,IX,IY)
      WRITE(5,100)ESC,MM,I
100   FORMAT(1H ,A1,A2,I1)
      RETURN
      END
C
C
      SUBROUTINE HEADER
      CALL TXTCOL(2)
      CALL TPLOT(0,0,750)
      CALL ANMODE
      WRITE(6,100)
      CALL TXTCOL(6)
```

L16

```
      CALL TPLOT(0,0,725)
      CALL ANMODE
      WRITE(6,200)
      CALL TPLOT(0,0,725)
      CALL ANMODE
      WRITE(6,300)
      CALL TXTCOL(3)
100   FORMAT(T25,' Solid Base Structure Program')
200   FORMAT(T50,'University of Sheffield')
300   FORMAT(T5,'By J.B. Edwards  &  M. Mazandarani',//)
      RETURN
      END
C
C
      SUBROUTINE TXTCOL(COLOUR)
      INTEGER COLOUR,MT
      BYTE ESC
      DATA ESC,MT /27,'MT'/
      WRITE(5,100)ESC,MT,COLOUR
100   FORMAT(1H ,A1,A2,I1)
      RETURN
      END
C
C
      SUBROUTINE GRFCOL(COLOUR)
      INTEGER COLOUR,ML
      BYTE ESC
      DATA ESC,ML /27,'ML'/
      WRITE(5,100)ESC,ML,COLOUR
100   FORMAT(1H ,A1,A2,I1)
      RETURN
      END
```

7. Illustrations

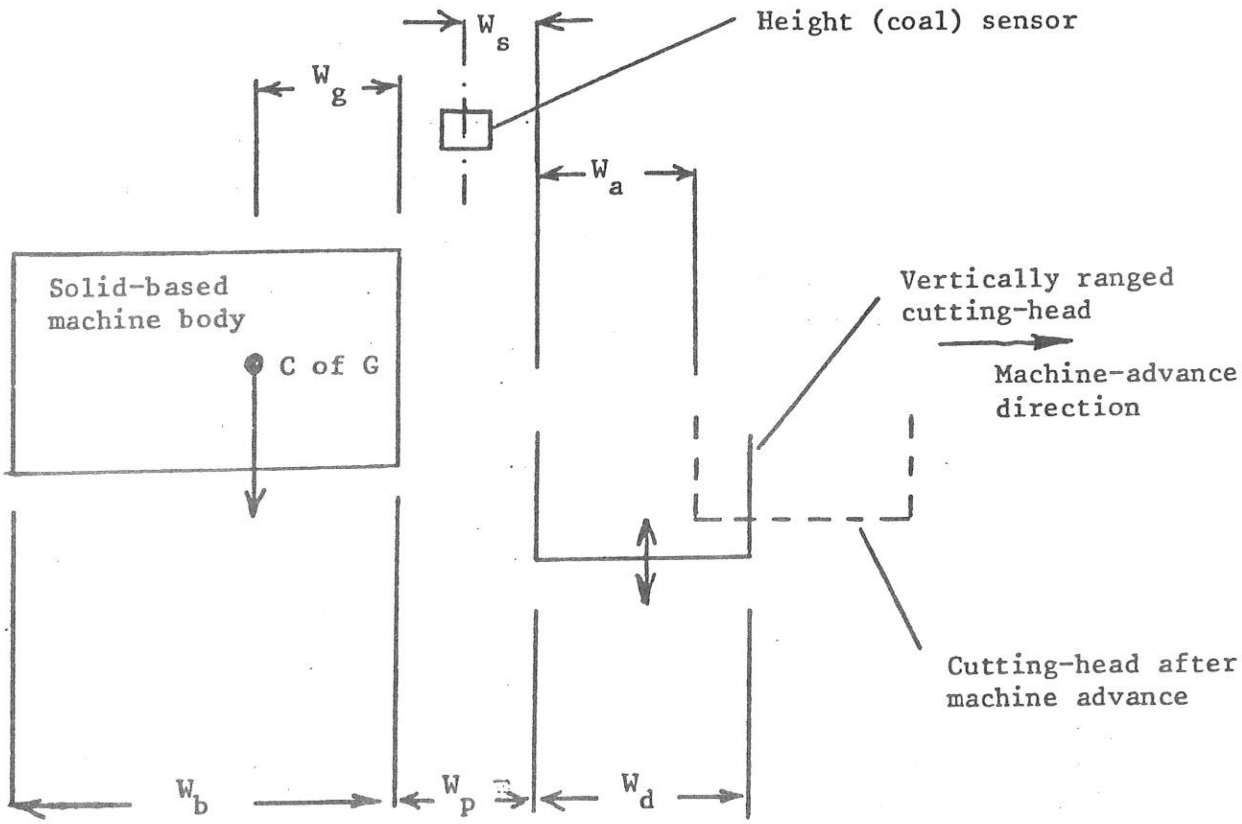
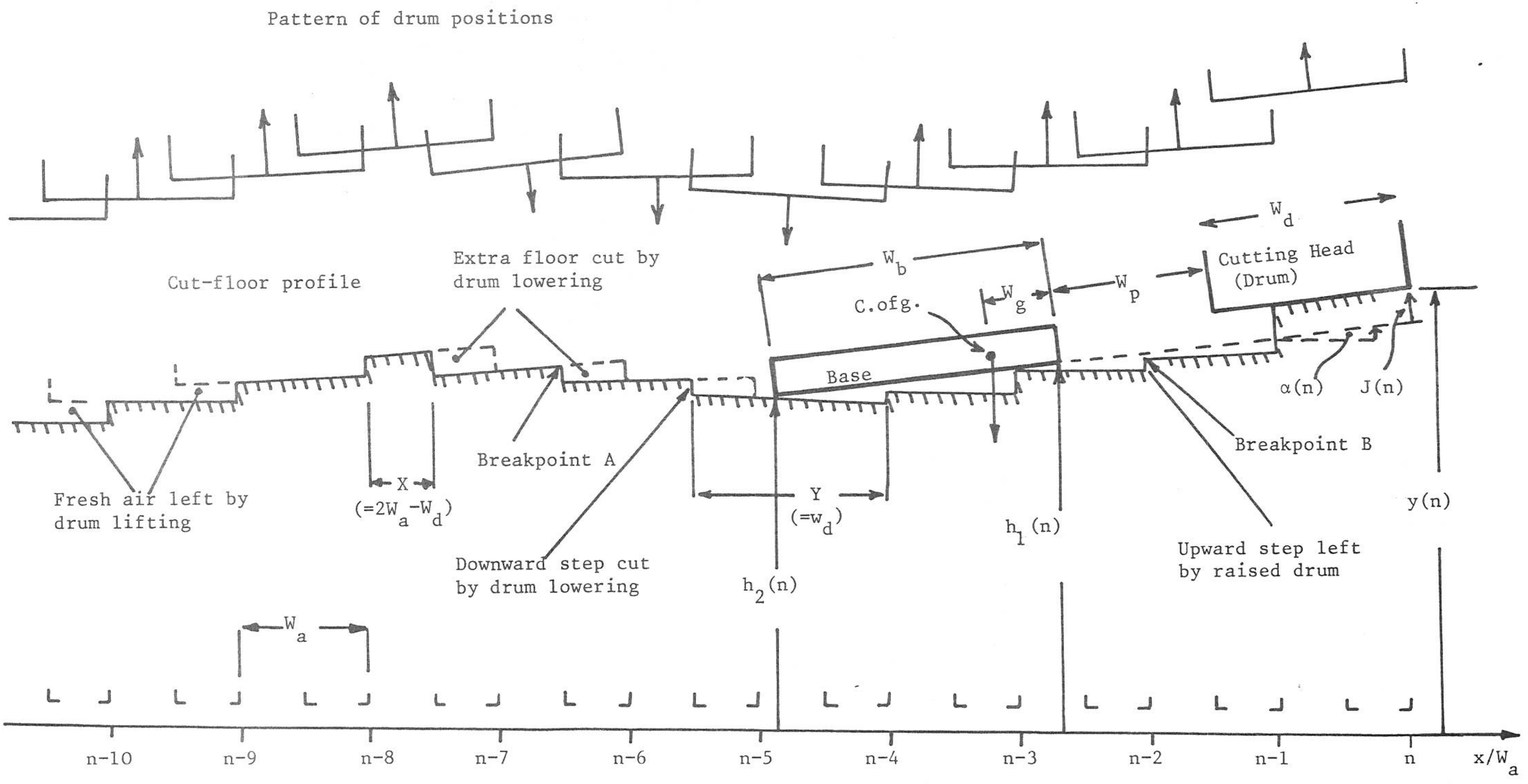


Fig.1. Showing geometrical parameters for Machine-steering simulation program

Fig.2. Showing machine variables and possible variation of break-point spacing in the cut floor



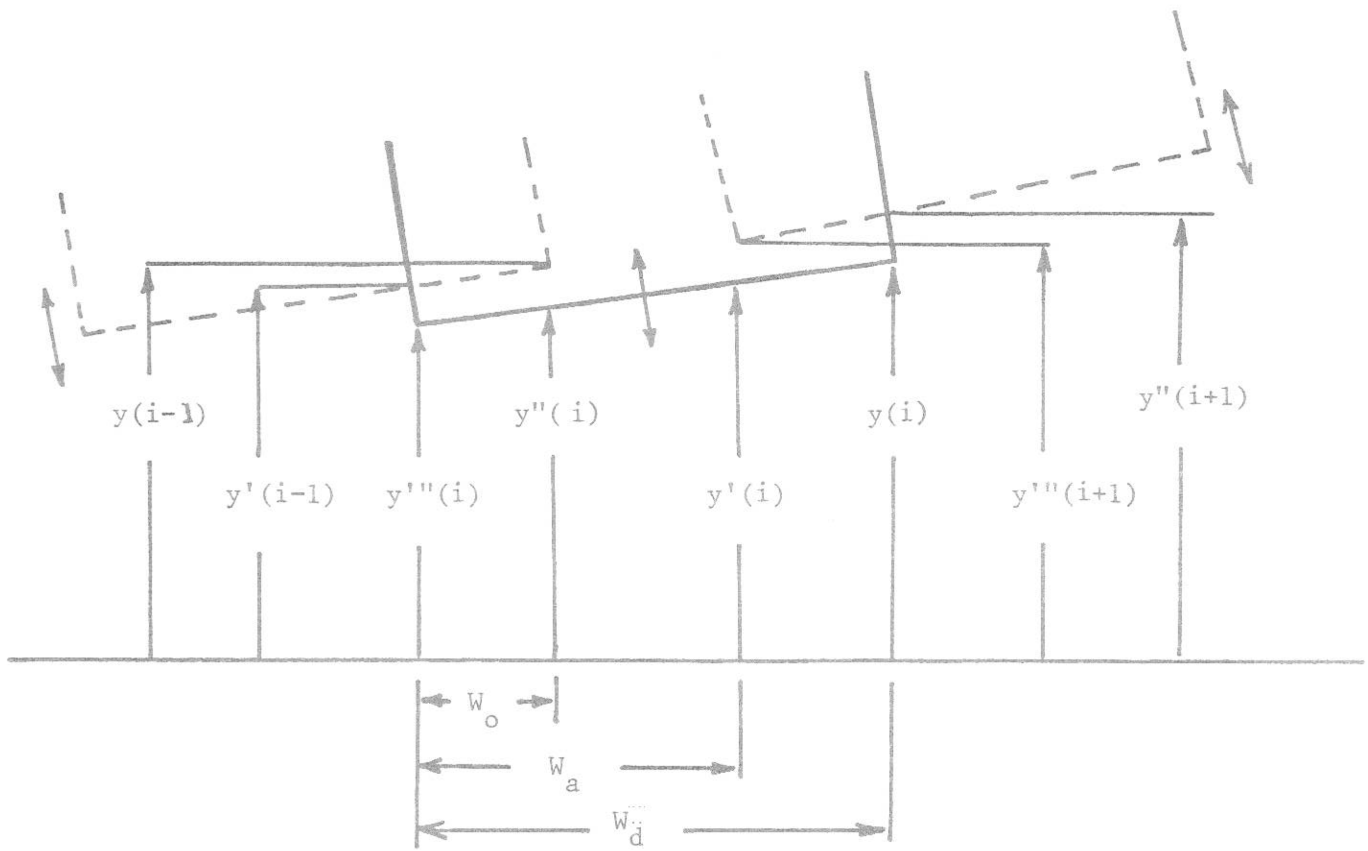


Fig.3. Potential heights of breakpoints in cut floor

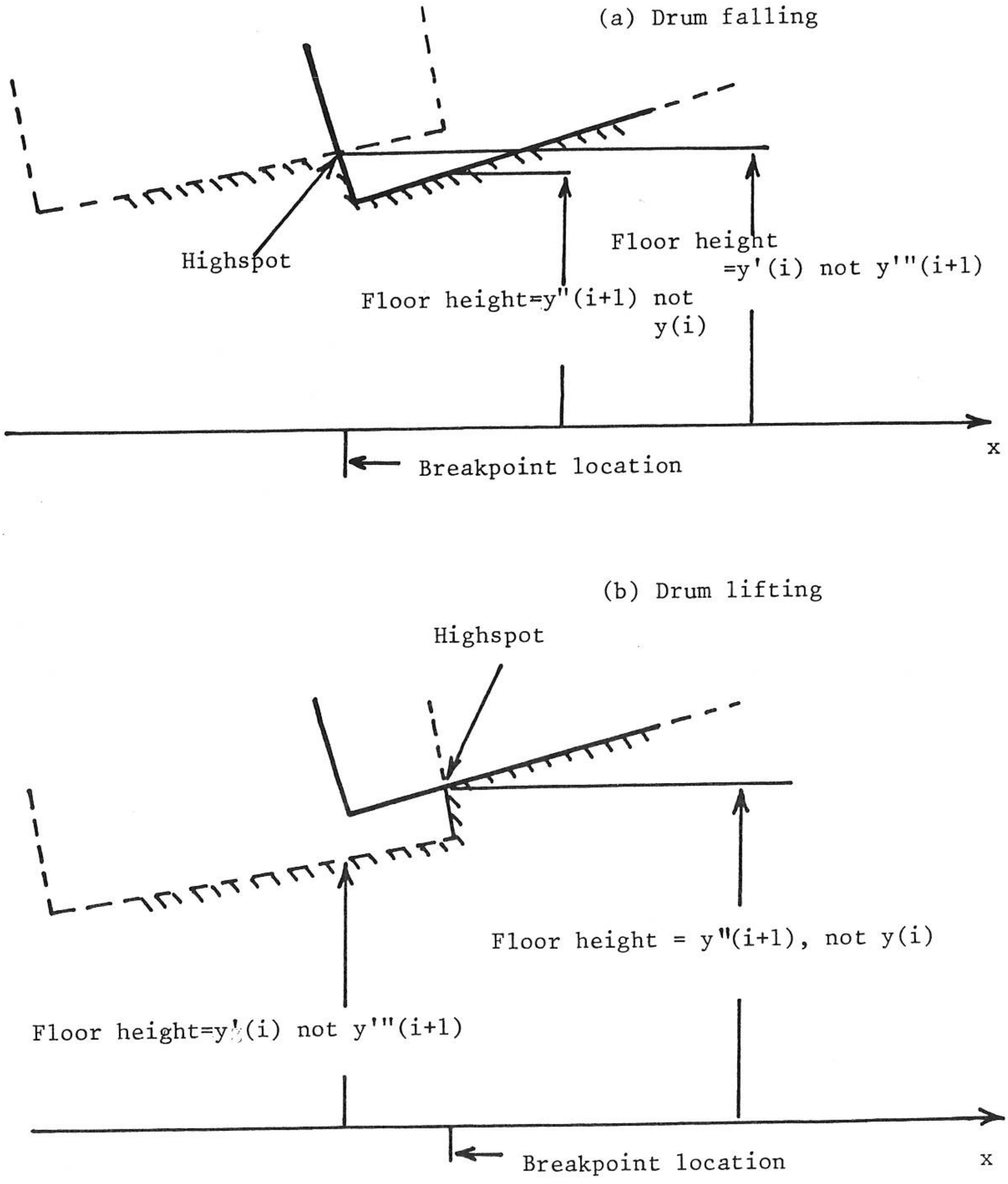
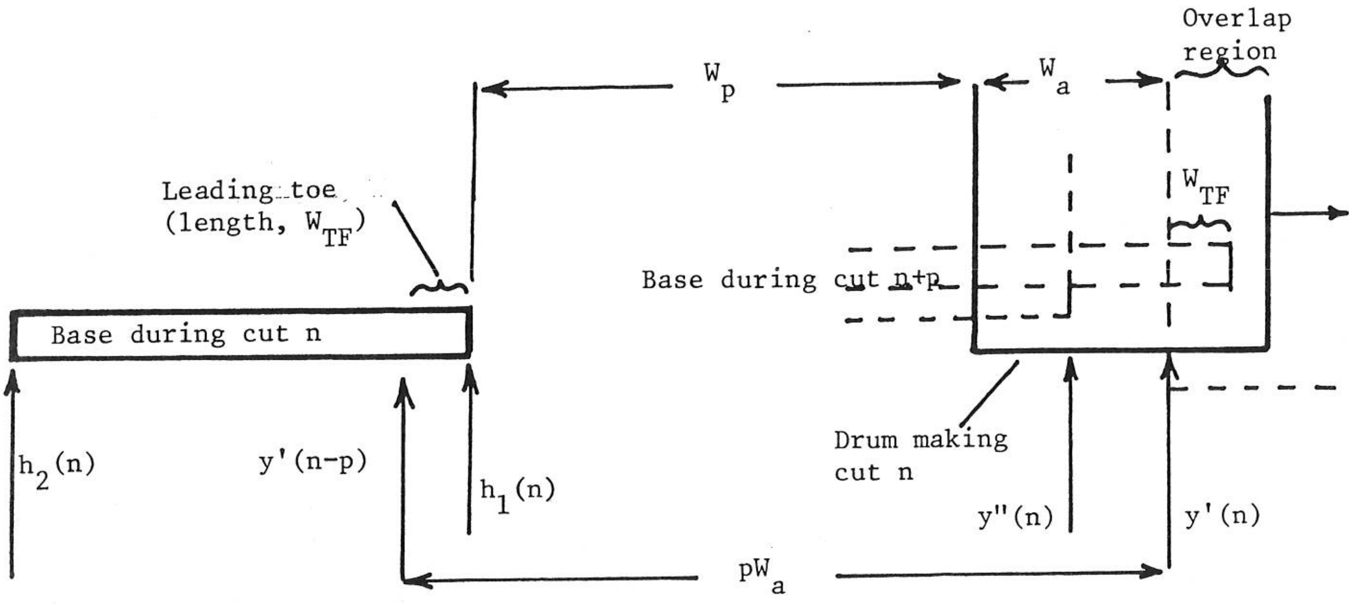
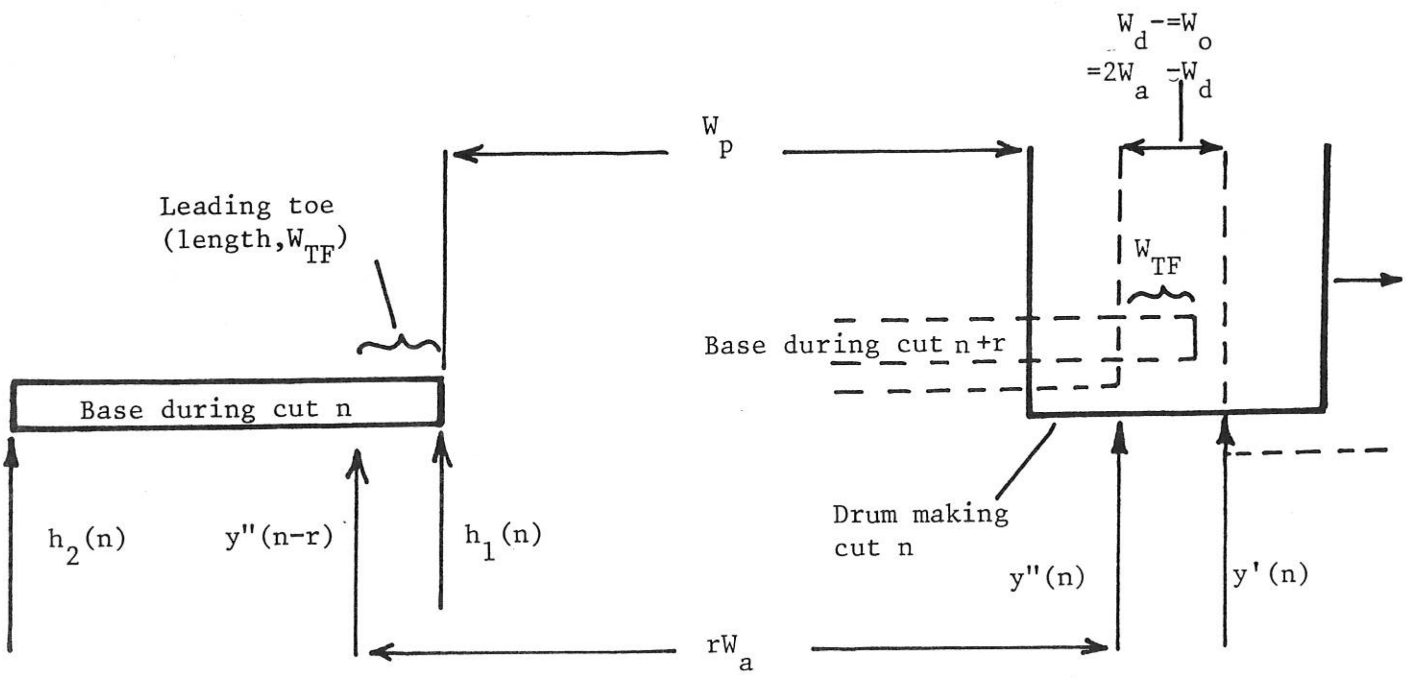


Fig.4. Showing breakpoint ordinates = y' or y" only.
(never y or y'").

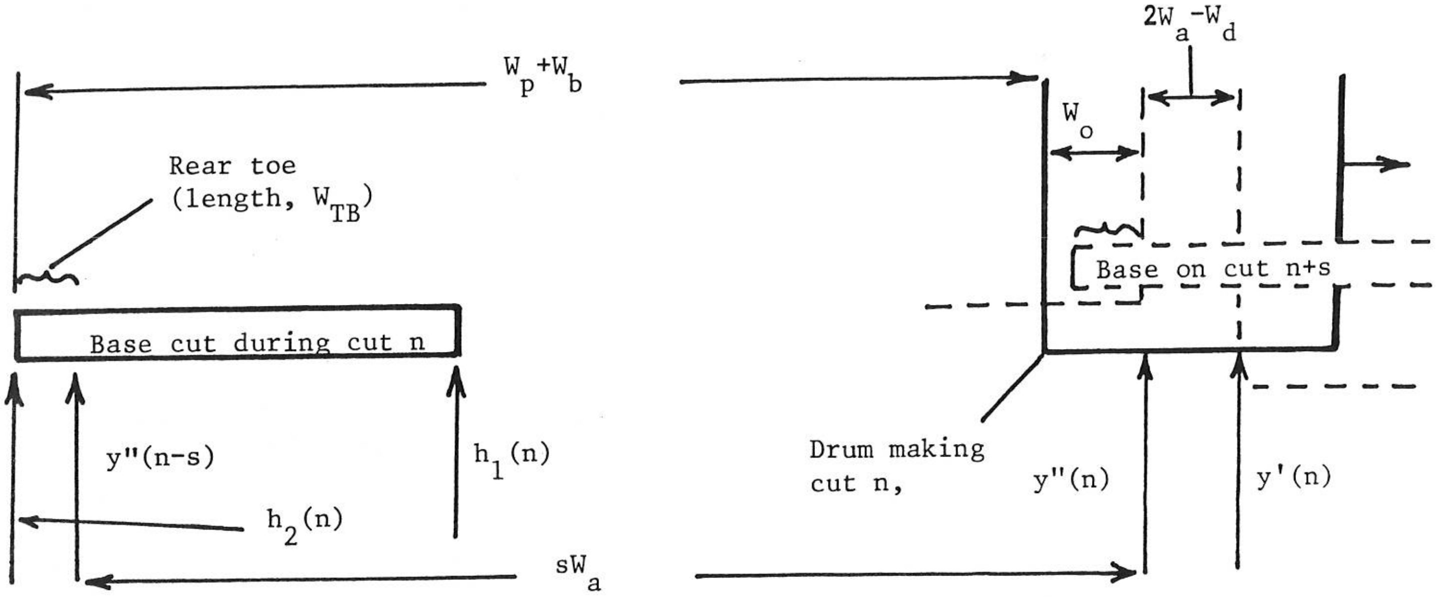


(a) Base front in overlap region (r = p)

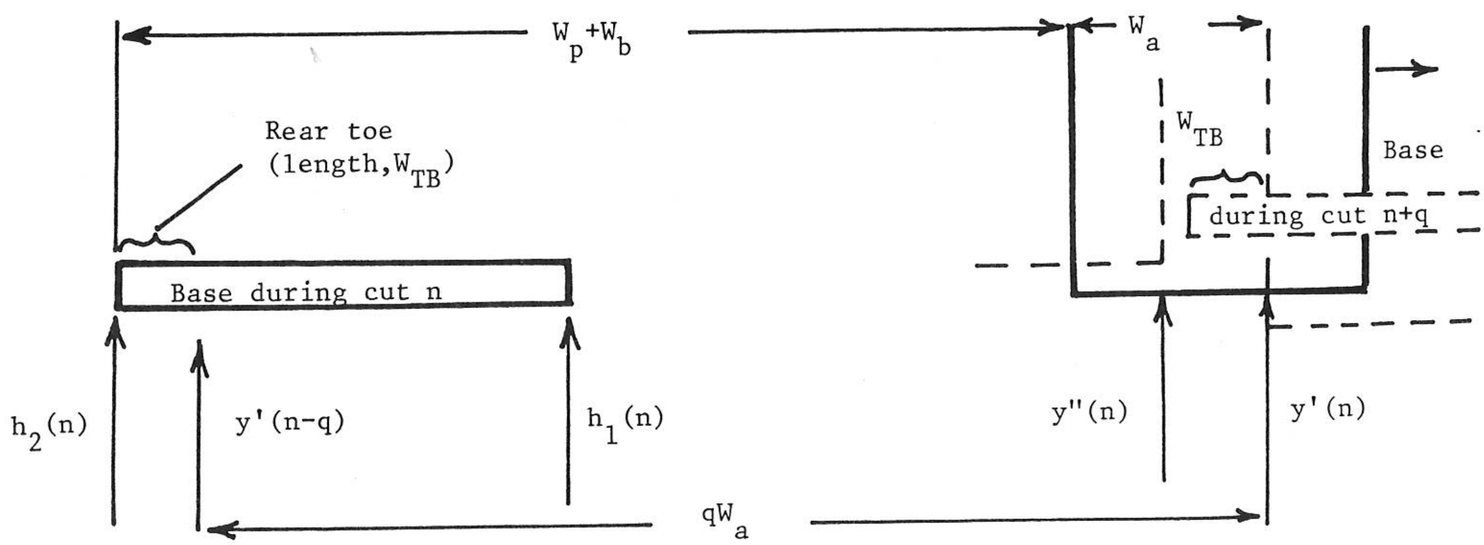


(b) Base front in non-overlap region (p=r+1)

Fig.5.- Showing the two possible locations for the front of the base.



(a) Rear of base in overlap region (q=s)



(b) Rear of base in non-overlap region (s=q-1)

Fig.6. Showing the two possible locations for the rear of the base

F7 sh 1

Fig.7. Outline Flowchart (Sheet 1)

(Details of user conversation, machine- and profile-plotting omitted).

Integer: p, q, r, s, cs

Arrays: v(10), w(10), x(10), z(10), f'(10), f"(10), (storage arrays for crucial floor heights), y_c(10).

Special initial conditions required ?

Yes

Enter required initial elements in arrays v to f"

No

$v(i) = 10$, $z(i) = 10$

$w(i) = 10$, $f'(i) = 10$

$x(i) = 10$, $f''(i) = 10$

$i = 1,10$

$y_c = 10$

Enter desired horizon height y_{ref} and coal sensor position integer cs(=1,2,3...)

A

Enter machine and base parameters W_a, W_b, W_c, W_d, W_g.

$W_o = W_d - W_d$ (eqn.(5))

p=o, r=o, q=o, s=o

(Find size of height-storage arrays (between drum and base and under base).

If $(p - W_p/W_a)>1$? (eqn.(10)).

No

p=p+1

Yes

If $(r - W_p/W_a)> W_o/W_a$? (eqn.(11))

No

r=r+1

Yes

GO TO Sheet 2

Fig.7. Sheet 2

If $(q > (W_p + W_b)/W_a)$? (eqn.13)

No: $q = q+1$

Yes

If $(s > (W_p + W_b + W_d)/W_a - 2)$? (eqn.12)

No: $S = S+1$

Yes

C. Find front and rear toe lengths

$p = r$?

Yes: $W_{TF} = (p-1)W_a - W_p$ (eqn.19) , No: $W_{TF} = rW_a - W_p - W_o$ (eqn.20)

$s = q$?

Yes: $W_{TB} = W_p + W_o + W_b - sW_a$ (eqn.21) , No: $W_{TB} = W_a + W_p + W_b - qW_a$ (eqn.22)

$f_{df} = x(r)$, $f_{dr} = z(q)$

Print p, q, r, s, W_a, W_b, W_p, W_d, W_g, W_o, W_{TF}, W_{TB}

Draw machine outline

OK ?

No: A

Yes

B

Call Linear Programming routine to minimise

$E = h_1 + (h_2 - h_1)W_g/W_b$ (eqn.29) subject to:

$h_1 + (h_2 - h_1)\{(p+i-2)W_a - W_p\}/W_b \geq f'(i)$, (eqn. 16)

$i = 1, q-p+1$

$h_1 + (h_2 - h_1)\{(r+i)\, W_a - W_p - W_d\}/W_b \geq f''(i)$, (eqn.17)

$i = 1, s-r+1$

$h_1 \geq f_{df}$, (eqn.27) $h_2 \geq f_{dr}$, (eqn. 28)

GO TO Sheet 3

Fig.7. Sheet 3

Get base heights h_1 and h_2 from L.P.

Control mode ?

Auto

$J = k_h(y_{ref} - y_c(cs)) - k_g(h_1 - h_2)$

$+k_r\{y_c - h_1 - W_p(h_1 - h_2)/(W_p + W_b)\}$ (eqn.43)

Manual

Read J from keyboard

$\alpha = (h_1 - h_2)/W_b$ (eqn.2)

$y' = h_1 + \alpha(W_p + W_a) + J$ (eqn.8)

$y'' = h_1 + \alpha(W_p + W_o) + J$ (eqn.9)

Display new floor profile

$y_c(cs+1-i) = y_c(cs-i)$ - (store coal sensor reading)

$i = 1, cs-1$

$y_c(1) = y'$ (eqn 44)

$y_{bf1} = y' + \alpha\{(r-1)W_a - W_p\}$ (for eqn. (23) or (24))

$y_{bf2} = y' + \alpha\{(r-2)W_a - W_p\}$ (for eqn. (23))

$y_{br1} = y' + \alpha\{(q-1)W_a - W_p - W_b\}$ (for eqn. (25) or (26))

$y_{br2} = y' + \alpha\{qW_a - W_p - W_b\}$ (for eqn. (25))

C. Store y' in array v for p passes pending base arrival

$p = 1$?

No

$f_d' = v(p-1)$

$v(p+1-i) = v(p-i)$

$i=1, p-1$

Yes

$f_d' = y'$

$v(1) = y'$

GO TO Sheet 4

Fig.7. Sheet 4

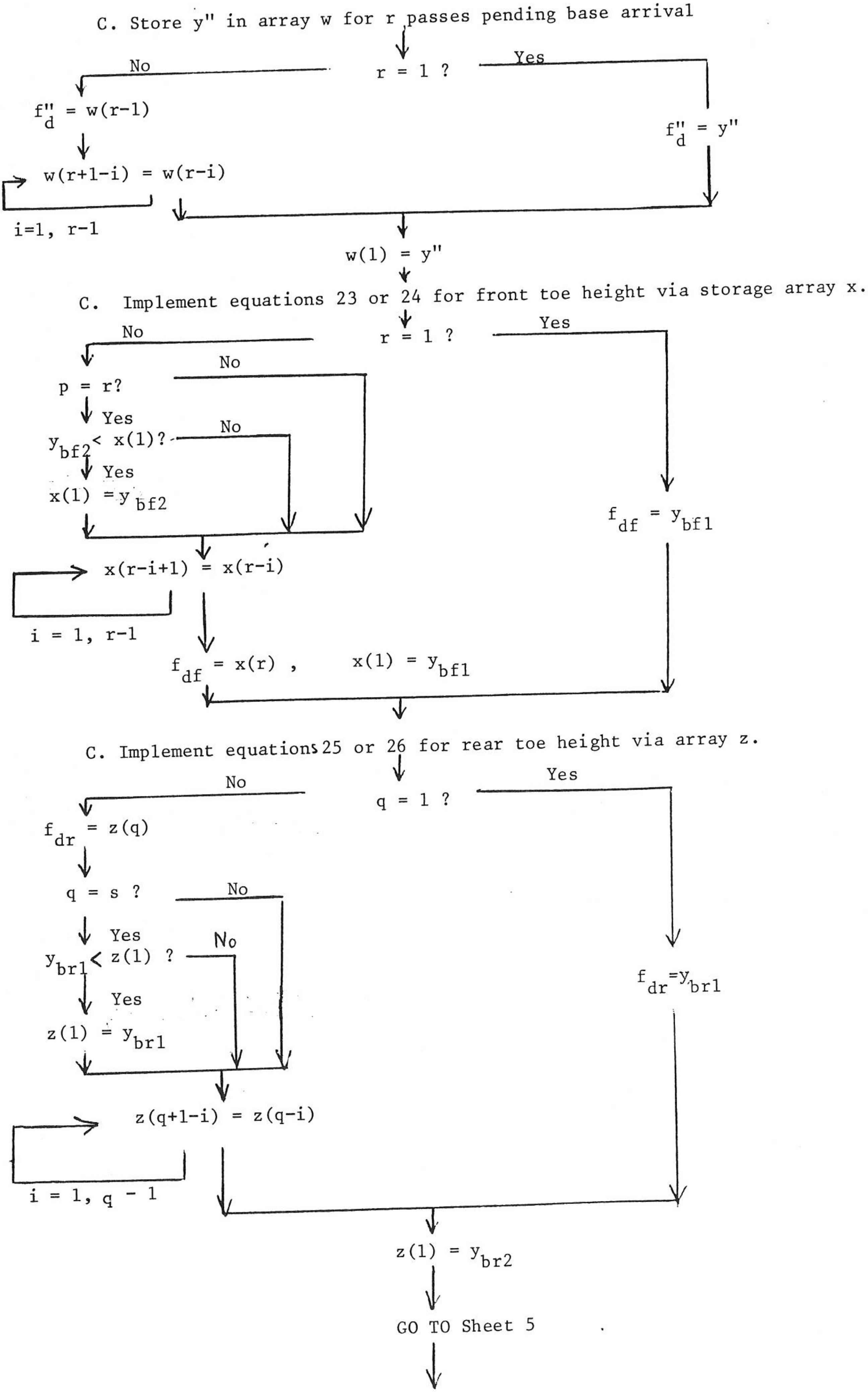

C. Store y" in array w for r passes pending base arrival
r = 1 ?
No
Yes
$f''_d = w(r-1)$
$w(r+1-i) = w(r-i)$
i=1, r-1
$f''_d = y''$
$w(1) = y''$
C. Implement equations 23 or 24 for front toe height via storage array x.
r = 1 ?
No
Yes
p = r?
No
Yes
$y_{bf2} < x(1)$?
No
Yes
$x(1) = y_{bf2}$
$x(r-i+1) = x(r-i)$
i = 1, r-1
$f_{df} = y_{bf1}$
$f_{df} = x(r)$, $x(1) = y_{bf1}$
C. Implement equations 25 or 26 for rear toe height via array z.
q = 1 ?
No
Yes
$f_{dr} = z(q)$
q = s ?
No
Yes
$y_{br1} < z(1)$?
No
Yes
$z(1) = y_{br1}$
$f_{dr} = y_{br1}$
$z(q+1-i) = z(q-i)$
i = 1, q - 1
$z(1) = y_{br2}$
GO TO Sheet 5

Fig 7 Sh 5

Fig.7. Sheet 5

C. Store y' ordinates from array v in array f' beneath base, on arrival

$f'(q-p+2-i) = f'(q-p+1-i)$

$i = 1, q - p$

$f'(1) = f'_d$

C. Store y" ordinates from array w in array f" beneath base, on arrival.

$f''(s-r+2-i) = f''(s-r+1-i)$

$i = 1, s - r$

$f''(1) = f''_d$

Maximum number of passes complete?

No

Return to L.P. on Sheet 2

B

Yes

Present final display

STOP

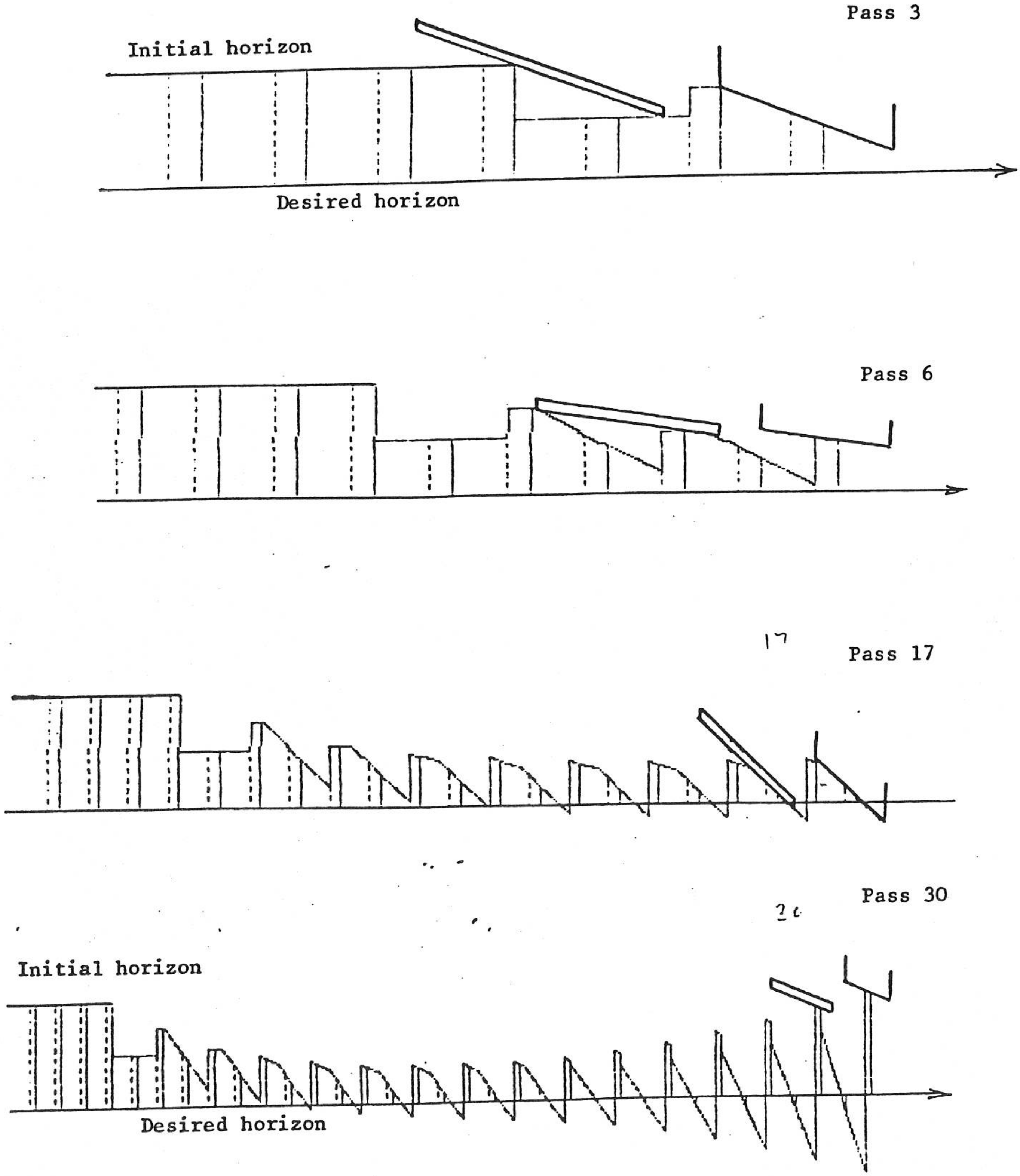


Fig. 9 Unstable system response

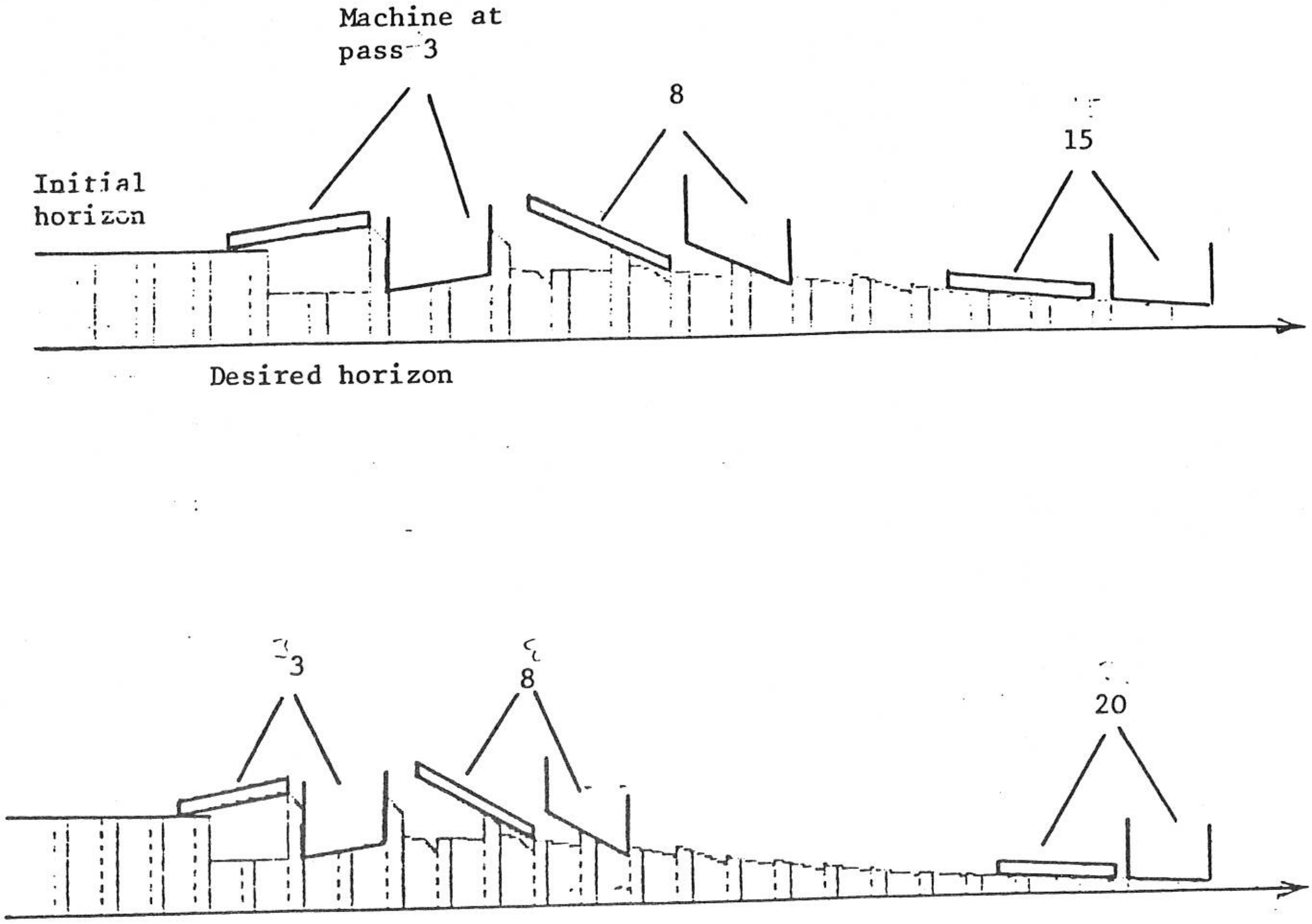


Fig. 10 System response stabilised by reduction of pick-to-pan distance