NASA Contractor Report 201602

Guidance and Control Design for High-Speed Rollout and Turnoff (ROTO)

S. H. Goldthorpe, R. D. Dangaran, J. P. Dwyer, L. S. McBee,
R. M. Norman, J. H. Shannon, and L. G. Summers
McDonnell Douglas Aerospace, Long Beach, California

Contract NAS1-19703

August 1996

National Aeronautics and
Space Administration
Langley Research Center
Hampton, Virginia 23681-0001

TABLE OF CONTENTS

1.0 SUMMARY

A ROTO architecture, braking and steering control law and display designs for a research high speed Rollout and Turnoff (ROTO) system applicable to transport class aircraft are described herein. Minimum surface friction and FMS database requirements are also documented. The control law designs were developed with the aid of a non-real time simulation program incorporating airframe and gear dynamics as well as steering and braking guidance algorithms.

An attainable objective of this ROTO system, as seen from the results of this study, is to assure that the studied aircraft can land with runway occupancy times less than 53 seconds. Runway occupancy time is measured from the time the aircraft crosses the runway threshold until its wing tip clears the near side of the runway. Turnoff ground speeds of 70 knots onto 30 degree exits are allowed with dry and wet surface conditions.

Simulation time history and statistical data are documented herein. Parameters which were treated as variables in the simulation study include aircraft touchdown weight/speed/location, aircraft CG, runway friction, sensor noise and winds. After further design and development of the ROTO control system beyond the system developed in reference 1, aft CG MD-11 aircraft no longer require auto-asymmetric braking (steering) and fly-by-wire nose gear steering. However, the auto ROTO nose gear hysteresis must be less than 2 degrees.

The 2 sigma dispersion certified for MD-11 CATIIIB is acceptable. Using this longitudinal dispersion, three ROTO exits are recommended at 3300, 4950 and 6750 feet past the runway threshold. The 3300 foot exit is required for MD-81 class aircraft.

Designs documented in this report are valid for the assumptions/models used in this simulation. It is believed that the results will apply to the general class of transport aircraft; however further effort is required to validate this assumption for the general case.

2.0 INTRODUCTION

OBJECTIVES

The objective of this study was to design a research ROTO guidance and control system which will be used to develop operational ROTO requirements for both automatic and manual piloted operation under normal and reduced visibility conditions. This ROTO system will utilize the satellite navigation system called Global Position System (GPS) operated in a differential mode.

The study addresses the design of :

- a variable auto-braking control law,
- a steering control law,
- ROTO exit prediction logic,
- ROTO arming procedures, and
- ROTO displays.

Another objective is to develop the runway/exit friction and navigational database requirements.

Manual ROTO steering and deceleration guidance control laws were not designed as part of this study.

REQUIREMENTS

The requirements used in this study are outlined in reference 1. The requirements were:

- Nose Wheel/Rudder Steering
- Braking/Reverse Thrust
- Touchdown Dispersion
- Navigation Noise and Update Rates
- Longitudinal/Lateral Acceleration and Jerk
- Control Law Update Rates

3.0 ROTO CATIIIB ARCHITECTURE

A preliminary assessment of the impact of CATIIIB requirements on the High Speed ROTO system requirements has been made and is herein described. The first step was to create a preliminary Functional Hazard Analysis to assign levels of criticality to the various functions and components of the ROTO system. A top level system diagram was prepared to facilitate the analysis and identify the main hardware or software elements of the system.

The results of the FHA were used to help determine the level of redundancy of various system elements and identify special monitoring or procedural requirements. The results of the following Functional Hazard Analysis suggest that most of the ROTO system components supply a critical function and erroneous outputs or loss of function at a critical time can produce hazardous or even catastrophic results. In order to satisfy FAA and other safety requirements it is therefore necessary to show that no single failure or combination of failures not shown to be extremely improbable will produce these results. In some cases this is accomplished with simple redundancy and monitoring. In other cases where adequate redundancy alone is impractical, it will be necessary to rely on showing that a more modest level of fault tolerance combined with a short exposure time to the critical case is sufficient.

In fact for much of the system components and interfaces, the redundancy and monitoring required to satisfy the concerns elicited by the attached FHA are already provided by the basic CATIIIB autoland system. Only the new components/tasks imposed by auto and manual High Speed ROTO function are addressed here. The shaded areas in figures 3.1 and 3.2 identify these items which are discussed below.

It is assumed that any software/hardware failure on the runway would revert to non-ROTO and a failure on the exit would remain in its current mode; auto or manual. The pilot would be free to take over from auto ROTO if he chooses.

FMS Data Base

The accuracy of the airport runway and taxiway geometric information provided by the FMS is critical to the ROTO guidance commands. In addition to assuring the data accuracy when loading the data into the database, detection of any failure affecting the data during use will be necessary. It would be unrealistic to assume more than two FMS functions can be provided. If there are only two FMS data bases, it is not clear how this can be construed fail operational unless all necessary data is transmitted to the ROTO computers prior to the start of the autoland after being validated by cross checking. FMS data may also contribute to estimating aircraft weight, drag and thrust characteristics; which are inputs to the ROTO software exit prediction logic.

[A dual FMS and data base is assumed.]

GPS

Positioning data provided by the DGPS receiver and associated local area augmentation signal source must be fail operational and provide the necessary reliability level to meet the "extremely improbable" criteria for loss of function. It is assumed this would also be the source of position data for the autoland guidance. No additional integrity requirements would be imposed as a result of the ROTO function.

[The DGPS airborne elements are assumed to be triplex with outputs voted within each using computer (ROTO and/or FCC).]

VHF Communications Link with Ground Station

Transmittal of estimated runway friction coefficient to the approaching aircraft is not a critical element since its absence would not affect the safety of the rollout. However, its accuracy and integrity must satisfy some minimal reliability requirement in order to take credit for advance prediction of the turnoff to be used.

The integrity of the voice communications with the control tower may be more critical since it is conceivable that a last minute abort may be required as a result of a taxiway encroachment incident. The tower would be expected to detect this situation and alert the approaching aircraft. The combination of the probability of this event and the probability of loss of VHF communications should be sufficiently remote that no new requirements are imposed on the communications link.

[VHF communications are assumed to be dual.]

ROTO Computer

All steering and deceleration commands originate in this unit both for automatic and manual mode of operation. The accuracy of commands and continuity of function must be assured throughout the rollout. This function would likely be incorporated within the same computers as the autoland functions (viz.; flight control computer), although it has been shown as a separate element on the reference diagram for clarity of function.

[A dual-dual or triplex implementation is assumed.]

Servo Backdrive for Rudder Pedals and Tiller

For pilot situational awareness, backdrive of all control functions except braking has been deemed to be necessary during the automatic ROTO mode. For the rudder pedals this should impose no new requirements since it would also be required for normal CATIIIB autoland.

In addition to backdriving the tiller for situational awareness, auto ROTO now requires the

ROTO computer to command the tiller via a dual servo which is fail passive (similar to MD-80 autopilot servo). The tiller signal compensates for the hysteresis in the rudder to nose gear mechanical control system. A dual fail passive servo is deemed adequate because a servo failure on the runway would revert to non-ROTO. Unlike other failures, this particular tiller servo failure on the exit would revert to manual ROTO, due to the undesirable MD-11 auto ROTO nose gear hysteresis greater than 2 degrees without tiller compensation.

[Dual backdrive for rudder pedals and simplex for tiller backdrive is assumed.]

Servo Drive for Reverse Thrust Control

If reverse thrust is judged to be necessary for adequate deceleration performance in automatic mode, then the position of the throttle levers will have to be moved to reflect the engine commands even with reversers deployed. This could be a significant complication unless a clever mechanical design of the throttle quadrant is contrived that allows the autothrottle servo[s] to move the levers while in reverse thrust command position as well as normal position.

[A dual throttle servo is assumed.]

Braking System

Braking action in response to variable automatic or manual commands must be dependable and consistent in order to achieve the deceleration profiles required to make the designated high speed turnoffs. Some current aircraft have dual braking systems. This should be sufficient to satisfy reliability and safety requirements if it can be shown that there is no loss of control for a single system failure and that the system adjusts adequately to the reduced braking pressure available. The braking computer is also required to be fail operational.

[A dual braking system is assumed.]

Nose Gear Steering Actuation

This (along with baseline rudder control) is the most critical function of the high speed ROTO system. Failure of steering function at a critical moment would probably be catastrophic. All portions of the control path from and including the ROTO computer/FCC to the actuators and connecting linkages must be shown to be fail operational. An anti-hysteresis software/tiller cable grabber system will reduce MD-11 auto-ROTO nose gear hysteresis down to an acceptable 2 degrees.

[A dual installation is assumed.]

ROTO Display(s)

If the ROTO maneuver is required to be initiated manually, then the every element required for satisfactory control must be fail operational. This includes the mechanism for display of runway map and/or command guidance. Loss of displayed information at a critical time is unacceptable. Dual display mechanisms are certainly a requirement. Furthermore, it must be shown either that the loss of the pilot's display at a critical moment is extremely improbable, or that it is possible to transfer control between pilots without significant disturbance even during the critical time.

[A dual display arrangement with fail obvious characteristics is assumed].

A preliminary reliability analysis has been performed to show that the redundancy levels selected for critical system components is adequate. This analysis is included following the FHA. In addition to the simple component redundancy and theoretical reliability analysis, it will also be necessary to show that no single point failures or common failure modes exist that could invalidate the redundancy. For example, possible software errors must be eliminated either by stringent design and test measures or by application of dissimilar design techniques or a combination of both.

PRELIMINARY FUNCTIONAL HAZARD ANALYSIS

INTRODUCTION

This preliminary functional hazard analysis (FHA) was prepared in accordance with DAC Design Safety Manual PD-503 and FAA Advisory Circular 25.1309-1A. It is an initial risk assessment of the functions and evaluates and identifies critical safety areas for potential hazards. This information is used in developing system safety requirements and establishing the framework for other analyses. While the FHA is not a compliance document, it is the basis for certain design decisions and does provide guidance for decisions on other analyses for certification.

The purpose of the FHA is to develop safety design requirements for the system(s) which will perform the function being considered and establish the framework for the Certification Plan and subsequent assessments. It provides information about potential functional failure conditions which should be used for establishing the required system architecture, software integrity level requirement, system separation and isolation requirement, and minimum equipment list (MEL) requirements. The subsequent design and development process must result in a system which, as installed, can be shown to meet applicable FAR Part 25 safety standards.

SYSTEM DESCRIPTION

The baseline high-speed rollout and turnoff (ROTO) guidance and control system architecture approach is shown in figure 3.1 and 3.2. The ROTO system architecture provides for an automatic ROTO mode as well as a manual ROTO mode. Both modes require the following equipment/functions:

1. A ROTO navigation input which is provided primarily by a global positioning system (GPS) operated in a differential mode and supplemented by an inertial reference system.
2. Ground operational data such as available exits, exit speeds, winds, temperature, surface conditions, other aircraft, runway friction, aircraft weight, etc.
3. ROTO exit locations/geometry which is permanently stored in the flight management system (FMS).
4. Control panel for the selection or de-selection of ROTO automatic or manual mode as well as ROTO abort.
5. ROTO displays that provides the ROTO guidance information on the head-up-display (HUD) and/or primary flight display (PFD).

6. Flight control computer to provide nose gear steering control, rudder actuation, and feedback signals to backdrive the tiller, rudder pedals, and throttle levers.
7. Brake computer to provide the braking function
8. Reverse thrust function.
9. ROTO computer

In the automatic ROTO mode the following operational sequence of events occurs:

1. Air traffic controller (ATC) offers all available ROTO exits to the ROTO software.
2. ROTO software offers the pilot one ROTO exit.
3. Pilot arms (accepts) ROTO exit.
4. ROTO deceleration provided by the auto braking and auto reverse thrust
5. Auto ROTO steering using the rudder, and nosewheel steering.
6. Pilot monitors the ROTO automatic mode by the HUD and/or PFD.

In the manual mode, the following operational sequence of events occurs:

1. ATC offers all available ROTO exits to the ROTO software.
2. ROTO software offers the pilot one ROTO exit.
3. Pilot arms (accepts) ROTO exit.
4. Pilot brakes, commands reverse thrust, and steers onto the ROTO exit by following the ROTO guidance information provided by the ROTO displays either on the HUD and/or PFD. If the pilot preselects auto brake/auto reverse thrust manual ROTO mode, the sequence 4 event is changed and an event 5 is added as follows:
4. ROTO deceleration provided by auto braking and auto reverse thrust
5. Pilot steers onto ROTO exit by following the ROTO guidance information provided by the ROTO display either on the HUD and/or PFD

Refer to figure 7.2 for a pictorial representation of the automatic and manual ROTO modes operational concept.

Preliminary ROTO guidance and control system architecture definition has identified the ROTO computer, the backdrive servos for the tiller, throttle levers in the reverse thrust mode, and variable auto-braking as the major unique ROTO system hardware additions to the baseline large and/or heavy transport aircraft. Modifications to the baseline aircraft system equipment are anticipated to be needed also to meet reliability and safety requirements, but the specific modifications have not been determined at this time.

SYSTEMS FUNCTIONS

The initial step in the development of the preliminary ROTO FHA consists of the identification of the ROTO system functions. The following functions list has been established for the preliminary FHA:

1. ROTO navigation
2. Runway acquisition clearance
3. ROTO steering guidance
4. ROTO deceleration
5. ROTO displays

HAZARD CLASSIFICATION

The hazard classifications used in this FHA are derived from requirements set forth in the Code Of Federal Regulations (CFR) and are defined as follows:

Class IV (Minor): Failure conditions which would not significantly reduce airplane safety, and which involve crew actions that are well within their capabilities. Minor failure conditions may include, for example, a slight reduction in safety margins or functional capabilities, a slight increase in crew workload, such as routine flight plan changes, or inconvenience to occupants.

Class III (Major): Failure conditions that would reduce the capability of the airplane or the ability of the crew to cope with adverse operating conditions to the extent that there would be, for example, a significant reduction is safety margins or functional capabilities, a significant increase in crew workload or in conditions impairing crew efficiency, or some discomfort to occupants.

Class II (Hazardous): Failure condition which would reduce the capability of the airplane or the ability of the crew to cope with adverse operating conditions to the extent that there would be, for example, a large reduction in safety margins or functional capabilities, higher workload or physical distress such that the crew could not be relied upon to perform its tasks accurately or completely, or adverse effects on occupants.

(Note: AC 25.1309-1A refers to this condition as "severe major.")

Class I (Catastrophic): Failure conditions that would prevent continued safe flight and landing.

AC 25.1309-1A defines continued safe flight and landing as:

> "The capability for continued controlled flight and landing at a suitable airport, possibly using emergency procedures, but not requiring exceptional pilot skill or strength. Some airplane damage may be associated with a failure condition during flight or landing"

SUMMARY OF RESULTS/RECOMMENDATIONS

The following summary of results/recommendations are based upon the functional hazard analysis worksheets contained in Appendix A beginning on page A-133. Tables 3.1 and 3.2 present the summary of Class I and Class II hazards. The FHA worksheets have been prepared based upon the high level ROTO guidance and control system functional and operational description presented in the SYSTEM DESCRIPTION section above. The objective is to identify top level critical system hazards and provide a high level system design approach to address the identified hazards in terms of system architecture recommendations/ considerations, such as selective application of redundancy, increased equipment reliability, failure monitoring concepts/schemes, etc.. Based upon tables 3.1 and 3.2 summary of Class I and Class II hazards, the following system architecture recommendation/.considerations have been developed and are provided in tables 3.3 and 3.4. Tables 3.3 and 3.4 include the system function, hazard description, baseline aircraft or unique ROTO system critical equipment that contributes to the Class I or Class II hazards, and the specific system architecture recommendations/ considerations for each identified critical system.

CONTINUING STUDIES/ANALYSES

One of the primary tasks of the reliability and safety effort for the design development process of the ROTO guidance and control system architecture will be the continuing development of the quantitative and qualitative tradeoff parameters for the selection process of fault-tolerant architecture. This initial preliminary FHA and the summary of system architecture recommendation/considerations for Class I and Class II hazards presented in tables 3.3 and 3.4, form the basis for these continuing studies/analyses. As mentioned in the preceding System Description section, modification to the baseline aircraft system equipment have not been determined at this time, but the development of theses unique modifications will be part of the continuing studies/analyses effort. In addition, the continuing reliability and safety studies/analyses will provide the reliability and safety design criteria for establishing the fault tolerant system architecture, software integrity level requirement, system separation and isolation requirement, and minimum equipment list (MEL) requirements that are part of the process in meeting the system safety certification requirements contained in the Federal Aviation Administration (FAA) Code of Federal Regulation (CFR) Chapter I, Federal Aviation Regulation (FAR) Part 25.1309.

Examples of some of the ROTO guidance and control system architecture and critical equipment issues that may be addressed in future studies/analyses include:

1. Selective application of reliability and safety enhancements such as redundancy, increased equipment reliability, etc. for the following:
    a. Global positioning system
    b. ROTO computer
    c. VHF data and communication channels
    d. Flight control computer
    e. Brake computer
    f. ROTO display electronics that drives the HUD and/or PFD

2. Failure monitoring concepts/schemes that addresses:
    a. Erroneous navigation data
    b. Erroneous deceleration commands
    c. Erroneous reverse thrust
    d. Erroneous steering guidance
    e. Erroneous rudder
    f. Erroneous nosewheel steering
    g. Erroneous aircraft weight information

ROTO SYSTEM RELIABILITY ANALYSIS

A preliminary reliability analysis was prepared based upon the current high speed rollout and turnoff (ROTO) guidance and control system description. The preliminary ROTO system functional hazard analysis established that the ROTO system quantitative safety certification requirement shall meet the extremely improbable case as defined in the Federal Aviation Regulation (FAR) Part 25.1309 and the Advisory Circular 25.1309.1A as follows:

> The occurrence of any failure condition which would prevent the continued safe flight and landing of the airplane is extremely improbable. Extremely improbable failure conditions are those having a probability on the order of 1 x 10E-09 or less.

This preliminary reliability analysis is an integral part of the overall ROTO system design development process by establishing the quantitative reliability probability estimate to validate that the proposed ROTO system architecture has the potential to meet the extremely improbable quantitative requirement.

The results of the ROTO system reliability analysis indicates that the estimated probability of failure is 2.06 x 10E-10 as shown in table 3.5 ROTO SYSTEM PROBABILITY OF FAILURE.

The reliability analysis was developed based upon the mean-time-between-failure values for the applicable equipment as shown in the table 3.5, the ROTO system reliability block diagram shown in figure 3.3, and an operational time of five minutes.

4.0 MODELING

The model used in this study was outlined in reference 1. The model was implemented in both FORTRAN 77 and MATLAB SIMULINK diagrams, which have been delivered to NASA Langley. The figure found in the appendix on page A-63 illustrates the MATLAB SIMULINK ROTO top level diagram. The aircraft simulation is a 3 degree of freedom (yaw, forward, lateral). It calculates aerodynamic, thrust and tire forces on the airplane and solves the resulting equations of motion to determine aircraft accelerations, velocities and positions during a simulated turnoff. The simulation also includes hydraulic models of the nosewheel steering, rudder and autobrakes. The simulation begins at touchdown. The model includes the following items:

ROTO Exit Geometry

Nosewheel, Rudder and Autobrake Actuation

Tire-runway Coefficient of Friction

Forces - Aerodynamic, Thrust, Braking Drag, Main & Nose Gear (Vertical & Side)

Aircraft Equations of Motion - Acceleration, Velocity, Position

Navigation

Winds

ROTO Control Laws

Exit Prediction Logic

An addition was made to the model to include rudder cable hysteresis (automatic ROTO) and nose gear cable hysteresis. The figure in Appendix A on page A-112 illustrates the cable hysteresis. The anti-hysteresis block is a software/hardware feature to minimize the auto ROTO nose gear hysteresis to 2 degrees. This feature is described in the Steering Control Law ROTO Design section 7.

A variety of aircraft types may be simulated by providing the simulation with unique aircraft characteristics (i.e. dimensions, actuator, gear, aero and thrust characteristics). Some of these characteristics are described below for an MD-11 and MD-81:

Unique Aircraft Characteristics	MD-11	MD-81
Landing Weight (1000 lb.)	340 to 480	82 to 128
CG	12% to 34%	-0.8% to 33.4%
Landing Air Speed (Keas)	130 to 166	110 to 143
Wing Span (ft)	165	108
Aircraft Yaw Inertia (million slug-ft^2)	26	4.1
Wing Gear:		
Number	2	2
Wheels/Gear	4	2
Center Gear:		
Number	1	0
Wheels/Gear	2	0
Nose Gear:		
Number	1	1
Wheels/Gear	2	2
Distance between wing and nose gear (ft)	81	72
Distance between wing gear (ft)	35	17

5.0 DATABASE REQUIREMENTS

It is assumed that long term permanent data (ROTO exit geometry data) would be stored in the Flight Management System (FMS) and temporary uplinked data (used in approach and roll-out) would be stored in the Flight Control Computer (FCC).

FMS ROTO EXIT GEOMETRY REQUIREMENTS

The MD-11 FMS database currently stores its permanent navigation data in 1 MB. Storage capacity is therefor at a premium. Each 3 ROTO exit airport would require approx. 400 bytes of FMS permanent storage. ROTO exit prediction logic requires approximately 20 aircraft parameters (4 bytes each) which could be permanently stored in the FMS.

1. The FMS database currently stores the runway threshold and end points in latitude/longitude coordinates. ROTO requires that the resolution of these coordinates be an inch. This would require the use of 1000ths of a second in polar notation or 7-8 decimal places using a 4 byte real number notation.

2. ROTO requires that aircraft DGPS position accuracy be at least +/- 2 feet. The resolution of the aircraft location should also be an inch. If the resolution is as much as 1 foot, the accuracy should tighten to not be greater than +/- 1 foot.

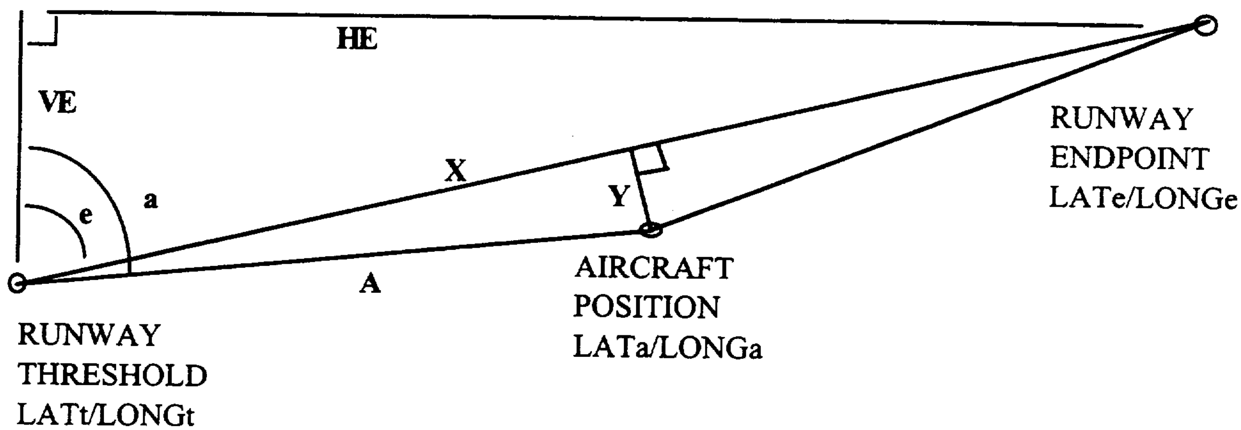
3. ROTO, developed in this study, uses an X,Y coordinate system to represent the runway and its exits. The runway is represented by the X axis with the threshold at the origin and the runway extending along the positive X axis. The right side of the runway, as it appears to a landing aircraft, has positive Y values and, therefore, a right hand exit would have positive Y values. The runway centerline has zero Y values. The units (resolution) of X and Y should be 1/2 foot. The accuracy of the X,Y coordinates should be an inch. The endpoint of a 12,000 foot runway would have coordinates of X = 24000 and Y = 0. The resolution of 1/2 foot allows X and Y to be represented by 2 byte integers, thereby minimizing X,Y pair storage requirements.

4. An array of 2 byte integer X,Y coordinate pairs in units of 1/2 foot represents an exit as a series of line segments. It is assumed that the exits cannot necessarily be represented by exact formulae such as an arc. The length of any particular line segment must not cause any point of the line segment to have a perpendicular error from the actual exit path greater than 1/2 foot. Using this criteria requires about 22 X,Y pairs to represent a 2200 foot long ROTO spiral (reference 2) 30 degree exit, reference 2. The exit centerline radius (2 byte integer, required by steering control law) is stored with each X,Y pair. Each 3 ROTO exit airport would require about 400 bytes of

FMS storage (3 * 2 bytes (X,Y,R) * 22 (2200 feet/~100 foot spacing) * 3 (3 exits/airport)). The exit centerline radius values may be stored in units of feet.

5. The aircraft's latitude/longitude position must be projected onto the runway's X,Y coordinate system using the following steps: (all trig functions must be accurate to 8 decimal places)

   a) Convert latitude/longitude positions from the FMS data base (polar coordinates) to floating point numbers.



$$\mathbf{DEG2FT} = \frac{6080\text{ft}}{\text{min}} * \frac{60\,\text{min}}{\text{deg}}$$

   b) Calculate the angles **e** and **a**.

Angle **e** only needs to be **calculated once** as follows:

$$\mathbf{HE} = \text{abs}(\text{LONGt} - \text{LONGe}) * \mathbf{DEG2FT} * \cos\left(\frac{(\text{LATt} + \text{LATe})}{2}\right) \text{(feet)}$$

$$\mathbf{VE} = \text{abs}(\text{LATt} - \text{LATe}) * \mathbf{DEG2FT}\text{(feet)} \qquad \mathbf{e} = \tan^{-1}\left(\frac{\text{HE}}{\text{VE}}\right)$$

Angle **a** would be calculated as:

$$\mathbf{HA} = \text{abs}(\text{LONGt} - \text{LONGa}) * \mathbf{DEG2FT} * \cos\left(\frac{(\text{LATt} + \text{LATa})}{2}\right) \text{(feet)}$$

$$\mathbf{VA} = \text{abs}(\text{LATt} - \text{LATa}) * \mathbf{DEG2FT}(\text{feet}) \qquad \mathbf{a} = \tan^{-1}\left(\frac{\text{HA}}{\text{VA}}\right)$$

c) Account for the affect of runway direction in defining left and right side of the runway.

if (LONGe is to the east of LONGt) then
    sign = 1
else
    sign = -1
endif

d) Calculate the distance **A** from the runway threshold to the aircraft location.

$$\mathbf{A} = \sqrt{\text{VA}**2 + HA**2}\,(feet)$$

e) Calculate the aircraft's **X** and **Y** position (4 byte real number) in units of 1/2 feet.

Aircraft's **X** position = **A** * cos(**a** - **e**) * 2 (1/2 feet)

Aircraft's **Y** position = **sign*A** * sin(**a** - **e**) * 2 (1/2 feet)

f) After calculating the aircraft's lateral deviation/deviation rate from the runway/exit centerline, the lateral deviation/deviation rate should be divided by 2 to convert them to units of feet for use by ROTO control law algorithms.

FCC ROTO UPLINKED DATA REQUIREMENTS

The uplinked data required by ROTO are DGPS ground-computed corrections for the continuous onboard update of the aircraft position and ground velocity and other data specific to the current approach (runway friction measurements, winds, available exits and exit speeds) for use by the ROTO exit prediction logic. If exit prediction were performed by ATC on the ground, the aircraft would only need to know its designated ROTO exit and exit speed. However, the aircraft weight may then have to be sent to the ground. All other data would still be required by the ATC. Memory space for approximately 20 aircraft parameters (4 bytes each) need to be reserved in the FCC (calculated) or FMS (permanent). A block diagram of the exit prediction algorithm is shown in figure 7.1.

Uplinked

1. All currently available ROTO exit locations and their accompanying exit entrance speeds. Exit geometries are permanently stored in the FMS navigation database.
2. Friction: An array of speed dependent surface conditions at locations along the runway (approx. 250 feet spacing). The surface conditions should be an average or represent the entire 250 foot length, otherwise exit prediction may result in false conclusions. The measurement spacing may not be as important as its accurate representation of the distance it covers.
3. Uplinked estimated winds and onboard DGPS position and velocity data would be used to predict the aircraft touchdown position and ground speed. Estimated flare distance may depend on approach mode (autoland, ILS manual, non-ILS manual). The touchdown location of a non-ILS manual approach may be difficult to predict. Use of ROTO exit prediction logic in ROTO operation may require the engagement of ILS manual or autoland before ROTO can be engaged. Estimated crosswind is also used within the exit prediction logic.

Onboard

1. Known (modeled) Aircraft Parameters:

   Thrust: Arrays of the following thrust profiles:

   - Approach Idle versus Time
   - Approach Idle versus Airspeed
   - Ground Idle versus Airspeed
   - Reverse Idle versus Airspeed
   - Maximum Reverse versus Airspeed

   Aircraft Wing Area

Percent MAC

Distance from nose gear to CG

Distance from main center gear to CG

Distance from main wing gear to CG

Aircraft CG Height

Forward most CG

Aft most CG

Drag Coefficient at forward CG (based on aircraft configuration, surfaces deployed)

Drag Coefficient at aft CG (based on aircraft configuration, surfaces deployed)

Average Anti-skid Efficiency

Expected Nose Gear T.D. Time (time starts after main gear touchdown)

Reverse Thrust Available Flag

Automatic Reverse Thrust Available Flag

Maximum allowed deceleration

2. Estimated Aircraft Parameters:

Aircraft Weight

Aircraft CG

6.0 FRICTION REQUIREMENTS

This section will describe friction requirements which were found through simulation and a literature search.

SIMULATION RESULTS

The friction requirements from simulation results were developed with non-grooved runway and exit models. Grooved runways and exits have a higher friction coefficient than non-grooved ones and should be considered for the ROTO operation.

Minimum Allowed Runway Friction

The runway occupancy times documented in this report were determined with exit locations at 3300, 4950, 6750 and 8000 feet past the runway threshold, using the non-grooved dry and wet friction curves found in figure 6.4. Different exit locations may be required for lower surface friction conditions (e.g. flooding, and ice). The exit locations used in this study will accommodate some amount of flooding and ice less than 10% and 5%, respectively, of the wet friction condition.

Some low-probability-of-occurrence, heavy/late MD-11 landings on a wet surface required a ROTO exit past 8000 feet and, thus, were considered for this study to be non-ROTO landings.

70 Knot ROTO Exit

For acceptable deceleration and steering on the 70 knot exit, the exit must have friction at or better than the following runway condition combinations: (see Maximum Ground Coefficient Friction for MD-11 Diagram, figure 6.4)

> 90% Textured Concrete Wet & 10% Textured Concrete Flooded
>
> OR
>
> 95% Textured Concrete Wet & 5% Ice (0.02*Dry)
>
> OR
>
> At or above an average constant friction coefficient of 0.2 (i.e. alternating 0.35 and 0.05 patches smaller than 15 feet in length). The Minimum Allowed Runway Friction section above allows a friction coefficient less than 0.2 on the runway. The Textured Concrete Wet condition has a lower friction coefficient than 0.2 at runway speeds.

If we assume that the entire exit has the Textured Concrete Wet surface condition, the **largest single contamination patch** centered at 700 feet past the exit entrance (full exit width) is the following: (see Maximum Ground Coefficient Friction for MD-11 Diagram, figure 6.4)

50 foot long patch of Textured Concrete Flooded

OR

15 foot long patch of Ice (0.02*Dry)

Friction Patches affect Exit Prediction Logic

The exit prediction logic using speed dependent measured friction along the runway at 250 foot intervals appears to be quite reliable. However, studying varying patches of friction has shown that the prediction can be false if the measured runway friction has errors due to friction patches not included in the measurements. Either the friction interval should be as small as the smallest reasonable patch size (5-15 feet) or the measured friction should be an average of the friction over the runway interval (250 feet) it represents. The current exit prediction logic makes a few more incorrect predictions as available friction becomes more critical with lower friction conditions.

LITERATURE SEARCH

Introduction

The ROTO system was designed to provide safe runway operation on dry and wet runway surfaces. The problems of operation for conditions or runways being contaminated with water (flooded), ice, snow-packed, or slush have been identified and are discussed in this section.

A goal of the research described herein was to develop a ROTO system to decelerate and steer the aircraft onto high-speed runway exits and onto taxiways at entrance speeds to the high-speed exit up to 70 knots. A suitable guidance and control (G&C) law to automatically steer a transport aircraft along runway centerline, onto a high-speed runway exit, and finally along a runway taxiway was to be developed. The system was also to be capable of decelerating the aircraft by means of autobraking and auto-reverse thrust to the desired turn-off and taxi speeds.

Adequate tire-runway interface friction for braking must be available for the system to effect necessary longitudinal deceleration and lateral control. This section addresses the tire-runway interface friction.

Operations

General

The runway friction during the high-speed turnoff is critical; therefore, it is recommended that the runway in the vicinity of the turn be grooved. Further, the grooving should be in a "diamond" pattern. This type pattern will improve the friction coefficient in all directions.

During the course of this study, CUSHION CUT, a manufacturer of pavement grinding and grooving equipment, in near-by Torrance, CA, was visited. It was observed that all passage-ways in their factory area were grooved. At the intersections there was cross-grooving. From walking on the floor in shoes with hard leather soles it was noticed, where the grooving was longitudinal, the lateral friction was high; whereas, in passage way intersections, with a "diamond" groove pattern, the friction was high in all directions.

This small example is believed to illustrate that cross grooving in the vicinity of the turn would resist the tendency of the aircraft to skid laterally when the centrifugal forces are highest.

If moisture is on the pavement, grooving will also prevent reverted rubber hydroplaning which may be started prior to the aircraft's reaching the point of turning. In this type of hydroplaning a pad of steam develops under the tire eliminating virtually all friction between the main tires and the pavement preventing braking and directional control; therefore it must be avoided.

Wet and Dry Runway

In beginning this discussion it is worthwhile to classify the "wet" runway as one which is well soaked but with no measurable amounts of water.

An international group consisting of the Federal Aviation Administration (FAA), Joint Aviation authorities (JAA) (European equivalent of FAA), Civil Aviation Authority (CAA) (British equivalent of FAA), Transport Canada (TC), National Aeronautics and Space Administration (NASA), and Manufacturers: Douglas, Boeing, Airbus, Gulfstream, Alenia, & Aerospatiale have been endeavoring to develop civil regulations for take off field lengths that account for degraded stopping performance on wet runways. To derive a conservative wet to dry friction ratio, data from some B737 tests demonstrated wet runway landing certification results, in which the average airplane braking coefficient on wet, grooved pavement exceeded 90% of the demonstrated dry runway value. This is illustrated by figure 6.1. Additionally, selected pages from NASA report SP-5073 (Reference 3 is shown by figures 6.2 and 6.3). These illustrate various runway surfaces and degrees of wetness and flooding. On each of these is a curve of 70% of dry friction coefficient. It may be seen that this curve shows adequate friction margin for all depths of wetness when the surfaces are grooved.

From this discussion, the conclusion may be drawn that for the WET and DRY RUNWAY, when grooved and adequate coefficient of friction margin exists, no problem will exist in accomplishing the high-speed rollout and turnoff at 70 knots.

Contaminated Runway and High-Speed Exit

Advisory Circular No. 91-6B (Ref. 4) defines a contaminated runway as one with more than 25% of the required field length, within the width being used, covered by standing water or slush more than 0.125 inch (3.2 mm) deep, or that has accumulation of snow or ice. Other situations would be appropriate to say it is contaminated if there is an accumulation in critical areas such as the high speed for lift-off, or the high-speed exit. The runway friction is as an average value. An average value is suitable for the runway, but not for the exit. The specific exit friction must be determined. For, if the aircraft encounters a slick area while turning, it may skid out of control.

Normal Deceleration

An assumption for ROTO operations is that the runway will have a first, second, and third high-speed exit. Nominally on dry or wet runways, the airplane would decelerate sufficiently to approach the first or second exit at 70 knots or less. The highspeed exit would then be accomplished.

Low Deceleration

If the airplane could not decelerate to 70 knots at the first exit because of contamination and a low friction level, the first exit would be aborted. If the deceleration is low enough that 70 knots is exceeded at the second and third exits, these would be aborted also. The aircraft would have to continue to the end of the runway before turnoff at a low speed.

If an exit is aborted, a closely-spaced following aircraft would likely have to execute a go-around, and the ROTO sequence would need to be modified.

Runway/Taxiway Surface Condition Sensors

It is recommended that Runway Surface Condition Sensors in accordance with (Ref. (5) be installed in the runway and highspeed turnoff at locations and in quantities as necessary to monitor critical areas. The highspeed turnoffs would likely be the most critical locations requiring the greatest number of sensors. Otherwise, the general locale of the airport -- its proximity to mountains, bodies of water, plains, etc.-, would dictate their need and location on the runway.

Signals from the surface condition sensors, embedded in the exit surface, would be monitored by the airport control tower. Evaluation of the sensors measurements would allow the ROTO system to determine whether or not an automatic high-speed turnoff could be accomplished.

Processing of the surface sensor measurements could indicate the acceptable speeds for use of the high-speed exit. As may be seen on friction curves in figure 6.4, at this speed adequate friction is available except for snow conditions to prevent lateral skidding of the airplane.

Figure 6.4 also shows ice to have a coefficient of approximately 0.05; therefore, movement of the airplane at the end of the runway would likely be accomplished with an airplane tow-bar and tug.

In addition to data from the surface sensors, friction information about the runways and exits should be determined as necessary with continuous friction measuring equipment as described in reference 1.

Discussion of Contamination

General

Under this heading the problems for potential operation on runways which are contaminated by flooding, ice, packed snow, or slush are discussed. These are the basic natural contaminants, which may be present in various forms, primarily due to temperature. The biggest problem arises from any one of these contaminates if the quantity is so great that it cannot be controlled. This results in closure of an airport, and if the airport is a major hub, airports throughout the nation will be affected. Early in January 1996, this was the case with many airports in the Eastern and Northeastern part of the United States. This was extreme, but lesser extremes can and will affect the operation of a High-Speed Rollout and Turnoff Guidance and Control System. The effects of these extremes can be reduced and the system can be back in operation sooner when certain precautions are made. Grooving the runway and exits is a highly important consideration.

The basic effect of these extreme conditions is the closure of the airport or a particular runway. The decision for closure can likely be made with more certainty than whether or not an exit is capable of supporting a high-speed turnoff. Several factors affect the precise determination of the friction level -- the particular contaminate; its roughness and hardness; the condition of the runway surface, macro and/or micro surface texture; temperature; condition of the airplane tires; calibration of the friction measuring devices; their correlation with the aircraft being considered; and more. The FFA (The Aeronautical Research Institute of Sweden) has made extensive studies concerning snow, ice and slush on runways (Reference 6). One result showed that runway temperature around 0 degrees C in combination with precipitation very often caused great discrepancies between experienced and measured runway friction.

An average friction value for the highspeed turnoff is inadequate. The minimum available friction value for a high-speed exit must be known. If a higher value is required at any point during the turn for the desired turnoff speed, loss of lateral control of the airplane could result. Thus, an average friction value for the high-speed turnoff is inadequate.

Flooding

Reference 7 summarized the research done relative to flooding during rainstorms, hydroplaning, identification of slippery runways, progress in developing antihydroplaning procedures, and rubber deposit removal. Grooving is extolled -- particularly grooves made with saws having diamond chips embedded in their cutting edge.

The exit speed of 70 knots is below the full hydroplaning speed of the MD-11; however, at this speed there is a wedge of water under a portion of the tire. If the pavement is grooved, the area of the wedge is reduced and more of the tire is in contact with the pavement even though the rate of rain fall is sufficient to cause complete flooding. Thus, some data supports the capability of making a high-speed exit on a grooved flooded runway. This capability may depend on the details of all the variables involved. Notwithstanding, based on the figure 6.4 friction curves the ROTO Operation at 70 knots cannot be accomplished on flooded textured concrete.

Pack-Snow, Ice and Slush

Advisory Circular No. 91-6B gives guidelines, and recommendations concerning takeoff and landing performance. It indicates that provisions should be for test measurements to establish the runway braking coefficients for standing water, packed snow, slush and ice. If such data is not available, appropriate data supplied by the manufacturer should be used. For the aircraft being considered in this study, test data for all conditions are not available, but training simulators data has been derived figure 6.4. This figure indicates the coefficient for packed-snow is marginal, for slush, the friction is high; but for ice, the coefficient is too low to consider.

These data correlate reasonably well with data from Ref. (8). This reference reports the Aircraft and Ground Vehicle Friction Correlation Test Results Obtained Under Winter Runway Conditions During Joint FAA/NASA Runway Friction Program. The aircraft used for these tests were the B-727 and the B-737. The mu effective values recorded for packed snow at 70 knots were between 0.15 and 0.21. The values for solid ice were 0.03 to 0.05 at 70 knots. There was no slush tested during this program.

Procedures for Minimizing the Affects of Contaminates

Weather in various parts of the world can be highly variable. Rain squalls can cause one part of a runway to be flooded for some time while the other end is completely dry. Freezing rain will stick to the pavement if it is already at or below freezing temperatures. High gusting wind can make precise control of the aircraft difficult. The snow fall can be so great that it exceeds the capability of the maintenance equipment to remove it adequately. Aircraft cannot land or take off from the airport. In extreme circumstances it must be closed. If precipitation rates are not too high and the temperature is not too low, then with judicious maintenance the airport surfaces can be kept free of contaminate such that they are not worse then wet conditions.

Two important advisory circulars are AC No. 150/5320-12B (Ref. 9), and AC No. 150/5200-30 (Ref. 10). The first gives guidelines and procedures for design and construction of skid-resistant pavements; pavement evaluation, and maintenance of high-skid-resistant pavements. This circular covers both the Asphaltic Concrete and Portland Cement Concrete. There is a good section on grooving for both types of concrete. Information is given relative to friction deterioration, evaluation and frequency of survey. The need for continuous friction measuring equipment (CFME) and general specifications are given.

The second advisory circular provides information to assist the airport owner/and operators in the development of acceptable airport snow and ice control programs. Chapters are given on personnel and equipment organization for Winter Operations on Airports; Airport Snow Removal Equipment -- Equipment selection, Services, and minimum equipment; Snow and Ice Removal Procedures -- Snow control procedures, Mechanical Methods and Chemicals for control and removal of snow and ice, and chemical affects on friction; also, use of abrasives and Runway Friction Improvements.

Neither of these circulars are strictly directed toward, or provide specific instructions relative to the High-speed Turnoff, but they serve to show the great amount of organization, equipment and care necessary to assure winter safety. The requirements for ROTO are more critical than addressed by the circulars and will require more study to assure that all facets are recognized and satisfactory solutions are developed.

Summary and Conclusions

Problems which will be encountered during the ROTO Operation with contamination of flooded water, ice snow-packed, and slush are discussed. Grooving of the pavement surface will enhance the friction, particularly for contaminated conditions. A diamond groove pattern will tend to cause improvement in all directions and is recommended in the area of the exit turnoff.

Dry and wet (well soaked, but with no measurable water) runway/taxiway appear to be satisfactory for accomplishing the high-speed rollout and turnoff at 70 knots.

Possible scenarios are discussed with varying amounts of contamination and location on the runway & exits and the use of surface condition sensors. These devices strategically placed can be monitored to indicate whether or not a safe turnoff can or cannot be made at a particular exit or subsequent exit locations.

The runway friction coefficient must be measured at regular intervals along its length. This data would be used by approaching aircraft.

Precipitation rates cannot be controlled. If the airports' available maintenance/cleaning equipment can cope with the rate, the pavement will be no worse than wet for most contaminates except for deluges of rainfall, snowfall and freezing rain.

Highspeed exits may be made for dry and wet conditions, slush, and for "flooded" if the pavement surfaced is grooved and depth of water is not too great.

Highspeed exits should not be made on ice or packed-snow. The latter appears to be marginal and should not be considered until further research and/or procedures are developed.

Recommendations

The goal of the ROTO concept is to provide a capability for landing airplanes every 50 seconds. Undoubtedly there will be some weather and contamination conditions which will prevent this desire from being met. The following considerations may help in meeting this criteria and/or clean-up of the runway and turnoff during or after severe weather conditions.

On various runway surfaces the depth of tire tread may make a considerable difference in the tire-runway interface friction. A method of determining and reporting the tire tread conditions would be beneficial.

Reverted rubber hydroplaning is hazardous. Precise reasons for this occurrence should be determined and measures taken to prevent.

When the 70 knot turn off cannot be met, a procedure for turnoff at lower speeds should be determined. This will reduce the runway occupancy time to less than the airplane being taxied to the very end of the runway.

There is a potential for the intersection of the runway and the highspeed turnoff, when the turnoff has a large radius, to have a large flat area which would drain poorly. Cross sections of such intersections should be studied to obtain optimum drainage.

Timing for the cleaning of the runway and taxiway after the subsidence of severe weather may be critical, so, the coordination for removal of contaminant and the maintenance of the airport/runway must be precise.

7.0 ROTO DESIGN

All of the ROTO control laws have update rates at 20 Hz. In this study, all control calculations that create the rate of an input are suspended if the input data is not fresh or is dropped out. The navigation inputs are updated at 10 Hz. All control laws appeared to be equally suitable for both MD-11 and MD-81 type aircraft.

AUTOBRAKING CONTROL LAW

The autobraking control law provides a variable deceleration command to the autobrake system. The purpose is to decelerate the aircraft to the selected ROTO exit's entrance speed and then continue to decelerate on the ROTO exit so that the aircraft may stop prior to the taxiway if desired. The deceleration command is limited to 9 ft/sec^2 (heavy braking) and rate limited so as to minimize longitudinal jerk. Some MD-11 simulation runs approached 9 ft/sec^2 during a portion of the runway. Inputs to the autobraking control law are the aircraft ground speed, aircraft X,Y position, selected ROTO exit entrance X position, selected ROTO exit entrance ground speed (70 knots used in this study), taxiway entrance X,Y position and taxiway entrance ground speed. The top level diagram of the autobraking control law is illustrated in Appendix A on page A-88, with lower diagrams thereafter. A simplified representation is shown in figure 7.9.

This study assumed that the pilot engages reverse thrust and speed brakes as soon as possible after touchdown, which will minimize the required braking. Auto-Reverse Thrust is discussed in the next section. The autobrake system is referred to in reference 1 or in Appendix A on page 125. A commanded deceleration results in brake pressure. The actuation system is identical to the existing MD-11 autobrake system except that instead of 3 available settings (high, medium, low), the actuation model allows the setting to be continuously varied by the autobraking control law. As in the existing system, the model delays all braking until after spoiler deployment to ensure adequate vertical loading of the main gear. Full braking ramp rates and gains are only available after nose wheel touchdown, which is approximately 6 seconds after main gear touchdown.

In order to minimize runway occupancy time (ROT), it is desired to delay the onset of braking until a ROT less than 53 seconds is assured. Safety issues might argue against this practice. ROT begins counting when the aircraft is at the runway threshold. The logic to begin braking is as follows:

1. Estimated final ROT is less than 53 seconds. OR
2. Estimated final ROT is increasing, ignore reverse thrust spool up time period. OR
3. Estimated final deceleration is greater than 6 - 8 ft/sec^2 (logic dependent). OR
4. A 53 second ROT requires a constant deceleration greater than 7.5 ft/sec^2.

Logic item 1 minimizes the deceleration magnitude once it is guaranteed that the estimated ROT is under our desired value. For logic item 2, if estimated ROT is increasing we assume that it is best to begin braking for the current exit rather than consider any later exit, even though the other logic may show that the estimated ROT is greater than our desired value. This logic should use a steady state estimated ROT, so we ignore the time during reverse thrust spool up. This study assumed reverse thrust spool up has ended 6 seconds after main gear touchdown. Logic item 3 uses current deceleration and deceleration rate to estimate a final deceleration. Logic item 4 attempts to keep a constant required deceleration under 7.5 ft/sec^2 for the desired ROT value.

When braking begins, a linearly decreasing speed profile versus the distance to the selected exit is created. The deceleration command is created by a PI controller, whose input is the aircraft's ground speed minus the desired speed at that location on the runway. This speed versus runway location profile is more easily modeled by the exit prediction logic described in a later section, rather than an initial deceleration method (reference 1) which decelerated the aircraft to reach a final speed at the exit entrance only. Because there is a target speed all along the runway, intermittent runway low friction patches are more quickly corrected for. The speed profile has a distance buffer (~100 feet) just prior to the selected exit.

The selected exit is aborted, up until 250 feet prior to the exit, if the aircraft's current required deceleration or estimated final deceleration is greater than 9 ft/sec^2. At the exit entrance the exit would be aborted if the aircraft's ground speed is greater than 2 knots above the exit entrance speed or that the aircraft has a lateral displacement from the runway centerline of X feet (needs to be investigated). MD-11 simulations on a wet runway with a steady 15 knot crosswind did not see a lateral deviation too great to cause an exit steering problem. If an exit is aborted, the aircraft would revert to non-ROTO. More discussion is required to decide whether pilots would then again want to again select auto ROTO for the next available ROTO exit.

The exit prediction logic, prior to touchdown, attempts to minimize the occurrence of aborting an exit on the runway by the deceleration exit-abort logic in the preceding paragraph. Before the exit prediction logic was implemented, the affect of the preceding paragraph's exit abort logic was:

1. The aircraft's current required deceleration would either abort an exit(s) immediately at touchdown or during the last third of the distance to a selected exit.
2. The aircraft's estimated final deceleration generally would abort an exit during the first third of the distance to a selected exit.

ROT is minimized by aborting appropriate exits as soon as possible, preferably prior to touchdown.

AUTO-REVERSE THRUST CONTROL LAW

Reverse thrust is needed for operations in low friction runway conditions and to minimize brake usage. An assumption for this study was that the pilot moves the throttle levers through the pedestal inter-locks to engage reverse thrust soon after touchdown and stows reverse thrust (or at a minimum sets it to idle) **at 70 knots ground speed**. Currently pilots generally stow reverse thrust based on airspeed. It is suggested that pilots stow reverse thrust just prior to entering the exit to negate that task while on the exit.

When reverse thrust is engaged, the auto-reverse thrust control law commands the auto-throttle servo to drive the throttle levers to achieve the desired reverse thrust. The control law adjusts reverse thrust so that the symmetric brake pressure stops between 20 - 30 % of maximum *supply* pressure during the ROTO deceleration phase. Because the engines have slower dynamics than the brakes, the brakes provide the longitudinal damping. The reverse thrust is allowed to vary between idle and maximum reverse thrust (Note: These negative thrust limits become more positive as airspeed decreases). When the deceleration command is zero, symmetric brake pressure decreases to *return* pressure.

The auto-reverse thrust command is integrated to slowly decrease reverse thrust to idle if brake pressure is below 20 % or the aircraft's ground speed is below the desired speed profile. There is a command deadband when the brake pressure is between 20 - 30 %. The command is integrated even more slowly to increase reverse thrust to maximum if brake pressure is above 30 %. The figure in Appendix A on page A-87 illustrates the auto-reverse thrust control law.

STEERING CONTROL LAW

The PID steering control law used for this study started with a basic auto rollout control law similar to localizer type control laws. Lateral centerline deviation is the main command error with lateral centerline deviation rate and yaw rate as damping feedbacks. On the ROTO exit, exit centerline radius and radius rate were also added as feedbacks to lead the aircraft into the curve. Aircraft ground speed was used for gain scheduling. The output of the control law is the position limited rudder command. . The top level diagram of the steering control law is illustrated in Appendix A on page A-106, with lower diagrams thereafter. A simplified representation is shown in figure 7.10. The nose gear is geared to the rudder through the rudder pedals and is limited to +/- 8 degrees. Navigation inputs (both the aircraft position and the data stored for path centerline) were converted to a X,Y reference frame for lateral deviation calculations. Nose gear loading was maximized by commanding a positive down elevator, scheduled with ground speed and aircraft CG.

For typical exit path geometry, lateral jerk normally occurs as the aircraft begins to follow the centerline of the exit because the physics of the path geometry require a step change in lateral acceleration to follow the exit path centerline. Gradual spiral exit (reference 2) entrances help to minimize this effect. If the exit has a constant radius entrance, the control law can contain a feature to smoothly increase the lateral acceleration to nominal lateral acceleration as defined by ROTO exit radius and desired exit speed. The lateral acceleration is increased at maximum allowable lateral jerk. This procedure is begun at a distance prior to the exit based on the aircraft ground speed and how many seconds are required to ramp up the lateral acceleration. During the pre-exit lateral acceleration, the lateral deviation of the aircraft changes less than a foot due to the commanded lateral acceleration.

With further steering control law development, auto asymmetric braking, geared to the rudder command to augment steering, was found in this study to be unnecessary. This study showed that nose gear steering with no more than 2 degrees of hysteresis is suitable for an aft CG MD-11 on a wet runway. It is suggested that the tiller be used to compensate the nose wheel such that the MD-11 auto ROTO nose gear hysteresis will be effectively reduced to 2 degrees. This suggestion for nose gear steering was implemented for this study in a tiller control law. The tiller control law subtracts measured rudder pedal angle from the desired nose gear (rudder) command. After washing out this signal, it is used to command a "cable grabber" servo which is connected to the tiller-to-nose gear cable. The tiller handle is back driven and the servo is commanding the nose wheel actuator through the tiller cables. This signal is mechanically summed with the rudder pedal nose wheel command. The tiller control law is shown in the figure in Appendix A on page A-113. It is believed that the tiller control law would be less costly than a fly-by-wire nose gear to decrease hysteresis of the baseline MD-11 auto nose wheel steering.

In auto ROTO the tiller control law activates prior to turning onto the exit entrance. The tiller control law is not active for manual ROTO. In manual ROTO, the pilot follows the flight director guidance with the yaw damper on.

AUTO ASYMMETRIC BRAKING CONTROL LAW

Reference 1 refers to the need for asymmetric braking to assist in steering an aft C.G. MD-11 on a wet high speed ROTO exit. Development of the baseline steering control law has now alleviated the need for this feature, as well as a 1-degree hysteresis, fly-by-wire nose gear (assuming auto ROTO tiller control law requirement). However, asymmetric braking would improve steering if it were available. All simulation runs described in this report do not include auto asymmetric braking, nor a fly-by-wire nose gear. The developed auto asymmetric braking control law is illustrated in the figure in Appendix A on page A-94.

The auto asymmetric brake command is geared to the rudder command and summed with the symmetric braking command at the brake. The asymmetric command passes through a deadband to minimize deceleration braking. An opposite command is sent to the opposing brake. If the command is in the direction to drive a brake's pressure below its return pressure, the command is doubled to the opposing brake.

The command is rate limited to minimize longitudinal jerk. Longitudinal jerk due to the brake pressure threshold cannot be helped, except to not let the brake pressure fall below the threshold during the asymmetric braking phase. Whenever the command is zero, a temporary command is created to drive the brake pressure asymmetry back to zero. Because the brakes act as integrators, if this were not done, some type of asymmetry would persist thereafter and fight the commanded rudder and nose gear.

EXIT PREDICTION LOGIC

In order to minimize runway occupancy time by controlled deceleration, it is desired to predict which available ROTO exits the aircraft is capable of using at touchdown. For ROTO pilot arming/work load reasons, this prediction would most likely occur up to a half minute prior to touchdown if touchdown parameter predictions are accurate enough to allow this. Whether a ROTO exit is selected by the pilot or by an exit prediction algorithm; selecting too early an exit would cause the exit to be aborted, causing the aircraft to coast to the next exit. Selecting a late exit would increase runway occupancy time above what necessary. Both of these occurrences may cause the following aircraft to go-around.

The exit prediction logic has the following predicted/estimated inputs: touchdown location, touchdown ground speed, aircraft weight, aircraft CG, aircraft drag characteristics and aircraft thrust versus airspeed/time profiles. Estimated steady airport winds and speed dependent runway friction along its length are also required by the algorithm. It may be possible to predict autoland touchdown location more easily than ILS manual or non-ILS landings due to the flare segment. MD-11 ROTO simulations have recommended the following MD-11 predicted value accuracies. The Exclusive Accuracy column is the maximum allowed error for a predicted value with that error being tested by itself. The Combined Accuracy column is the maximum allowed error for a predicted value with all of the errors being tested together.

Predicted Value	Exclusive Accuracy Relative to Actual	Combined Accuracy
Touchdown location	0 to +300 feet	0 to +100 feet
Touchdown ground speed	0 to +3 knots	0 to 1.0 knots
Touchdown weight	0 to +100,000 lb.	0 to 30,000 lb.
Runway friction coefficient	-0.03 to 0	-0.01 to 0

It appears that the current modeled MD-11 dispersion and exit locations only allow the prediction errors to lie on the conservative side. If the predicted touchdown location error were -100 feet, the exit prediction logic would perhaps accept too early of an exit which might be eventually aborted. If the predicted touchdown location error were +400 feet, the exit prediction logic would perhaps accept to late of an exit than necessary which would unnecessarily increase the runway occupancy time.

These prediction accuracies are dependent on the simulation accuracy, the number and location of ROTO exits and aircraft type. It may not yet be technically feasible to achieve these accuracies necessary to correctly predict ROTO exits, which is essential to minimize following aircraft go-arounds which are currently at about 1 per 1000 landings. The prediction errors would decrease as the prediction is made nearer to touchdown.

These MD-11 simulations assumed speed dependent friction was measured along the runway approximately 250 feet apart. It was found that this friction measurement should represent/average the friction value of the entire 250 feet. If there are friction patches on the runway and the friction measurement is in or out of a patch, these non-representative measurements may accumulate to cause a false exit prediction.

The exit prediction algorithm attempts to simulate the aircraft's onset of deceleration and deceleration (due to brakes, thrust, drag, speed brakes) down the runway until the aircraft CG reaches the target exit entrance. The logic to begin aircraft braking is referenced in the auto-braking section above. The exit prediction simulation steps down the runway in terms of distance rather than time. Because the actual aircraft attempts to decelerate according to a linear speed profile versus distance rather than target a final speed at the exit entrance, it is more easily modeled by the exit prediction algorithm.

When the simulated aircraft reaches the predicted exit, if the required friction is greater than the available friction or a deceleration greater than 9 ft/sec^2 is required, the predicted exit is aborted. The prediction algorithm then repeats until a suitable available exit is found.

The exit prediction logic path is illustrated in figure 7.1 and MATLAB script code in Appendix A on page A-27.

FLIGHT DIRECTOR MANUAL ROTO GUIDANCE

Manual ROTO steering and deceleration flight director guidance control laws were not designed as part of this study. The flight director commands would most likely be a filtered output of the auto ROTO steering and deceleration control law's forward paths. The yaw damper should be active with manual ROTO steering. The HUD display provides both command and raw data guidance.

CURRENT ROTO OPERATIONS

Some U.S. airports, e.g. Dallas-Ft. Worth, under flight crew discretion conduct manual high-speed ROTO under daylight VMC conditions with no surface contamination on the exits. This applies to both narrow and wide body aircraft on 30 degree exits at exit entrance ground speeds up to 60 mph.

The current runway clearance definition which would apply to continuous ROTO operation is as follows. The runway would be clear for a following aircraft at the runway threshold, if all preceding aircraft have cleared the runway and are past their exit hold lines (figure 7.11) or are in the process of rolling past their exit hold lines without obstruction. The runway would not be clear if a preceding aircraft has stopped on an exit prior to an exit hold line.

If an aircraft stops on an exit past the hold lines, the taxiway is open, but that exit is not available as a ROTO exit for following aircraft. The runway would not be clear for a following aircraft that required that occupied ROTO exit, unless the following aircraft performs a NON-ROTO landing.

ROTO MODE LOGIC

Figure 7.2 illustrates the ROTO modes (states) and the transitions between those modes. For any mode, the arrow originating from a darkened circle is the default transition. In approach, ROTO begin in the NON ROTO state. After

ATC has provided the expect ROTO clearance AND

ROTO exit prediction software has identified a ROTO exit

, ROTO transitions into the ROTO READY mode. The pilot may arm ROTO and an exit causing the transition into the AUTO ROTO SELECTED mode by pressing the ROTO button on the glareshield. AUTO ROTO provides automatic steering and deceleration on the runway and onto and through the ROTO exit, while the pilot monitors out-the-window through the HUD and other displays.

The pilot may transition between the READY, AUTO ROTO and MANUAL ROTO (manual steering with guidance/while maintaining auto deceleration) modes by repeatedly pressing the tri-state ROTO button on the glareshield. Additionally, after touchdown the pilot may also press the yoke autopilot disengage button to select MANUAL ROTO.

After touchdown in AUTO ROTO SELECTED mode, if the pilot overrides the rudder pedals, ROTO would transition to MANUAL ROTO (manual steering with guidance/auto deceleration) mode. After touchdown in AUTO or MANUAL ROTO, if the pilot overrides the brake pedals or throttle levers, ROTO would transition to FULL MANUAL ROTO (manual steering with guidance/manual deceleration with guidance).

At any time, AUTO or MANUAL ROTO would revert to NON ROTO if

Pilot deselects ROTO OR

ATC rescinds ROTO Clearance OR

Aircraft is in GO-AROUND OR

ROTO exit is aborted OR

Any mechanical/software ROTO function has **failed on the runway prior to the exit**. The failure must be positively indicated.

Prior to touchdown the pilot may select NON ROTO by pressing the yoke autopilot disconnect button. After touchdown the pilot may toggle the glareshield ROTO button to the READY state.

ROTO would remain in its current mode if any mechanical/software ROTO function fails on the exit. A failure annunciation would notify the pilot of the condition, whereupon he/she is free to override the controls and manually control using ROTO guidance.

Under certain timing conditions, if an exit is aborted (ROTO mode reverts to NON-ROTO), there may be enough remaining runway occupancy time to transition again from NON-ROTO to AUTO ROTO by selecting a second available ROTO exit, then exiting that second exit while still keeping the runway occupancy time under 50 seconds, thereby not requiring a close following aircraft to GO-AROUND. This topic needs further discussion and pilot input.

ROTO OPERATIONS AND CREW INTERFACE

The ROTO crew interface concept was developed in response to anticipated flight crew and Air Traffic Control (ATC) operational requirements associated with ROTO's implementation in terminal area operations. More specifically, development of the crew interface concept for ROTO operation was accomplished in two phases. In conjunction with a McDonnell Douglas engineering test pilot and two commercial airline pilots, a targeted operational analysis was conducted articulating approach, landing, and roll out functions required of the crew and ATC. Based on this analysis, a design effort was performed -- again with substantial pilot contribution and review -- yielding the specification of control and display system elements needed to perform these required activities.

Operational Analysis

To conduct a more substantive operational analysis, crew system design personnel and the engineering test pilot familiarized themselves with the major functional capabilities and system logic parameters of the ROTO system: Relationships between ROTO-specific system calculations and performance, and potential modifications to normal guidance, brake, and reverse thrust control; automatic and manual modes; and deceleration and steering profiles. Of specific interest were estimates of ROTO calculation times, crew-ROTO coordination in the application of thrust-reversing and braking, and the requirement for out-the-window monitoring of ROTO progress.

After acquiring general familiarity with ROTO system performance, a representative functional timeline was developed to delineate how ROTO operation should be integrated with standard terminal area arrival, approach, and landing procedures. This timeline, presented in Appendix A on page A-139, identifies the necessary communication events between the crew and ATC, and indicates the probable sequencing and timing of crew and ROTO system activities. The timeline was particularly useful for articulating the relatively critical timing constraints associated with ATC's clearance for ROTO and the subsequent execution of ROTO-directed landing, roll out, and turn off. (It should be noted that, in the timeline, ATC maintains ~50 second separation between ROTO-controlled landings by issuing the ROTO clearance for the aircraft under discussion immediately after a preceding aircraft has cleared the exitway needed for the landing aircraft -- in the last five to ten seconds of flight before touch down. This proposed procedure must be carefully evaluated in subsequent ROTO operational analyses because of concerns about workload, crew situation awareness, and the actual feasibility of ATC issuing such clearances in such small and critical 'windows' of time. Alternative procedures, such as earlier clearance issuance with possible (last minute) recision, would need to be explored in on-line piloted simulation activities.)

ROTO Control and Display Elements

From this operational analysis, control and display elements were designed and their operations defined. Following a design philosophy of minimal modification of existing procedures, and correspondingly minimal modifications of control and display elements, the ROTO system as shown in Figure 7.3 comprises several components:

1. A modified Head Up Display.
2. A glareshield control head (including ARM and READY settings) and annunciation panel.
3. A Primary Flight Display containing a backup (head-down) representation of ROTO-related information displayed on the HUD.
4. A Navigation Display (ND) showing a plan view representation of the aircraft's progress on the ROTO deceleration profile and steering track.
5. A slightly modified Flight Management System page for specifying and executing ROTO calculations/estimates (to prepare ROTO for initiation after clearance).
6. An optional data link system for receiving ROTO-relevant airport information, and for crew acknowledgment of ROTO's clearance.
7. A modified Autopilot Disengage switch on the yoke for ground-operational de-selection of Automatic ROTO steering (thus releasing steering control to the crew for Manual-mode ROTO operation).
8. A "ROTO" setting on the Autobrake Control panel.

Owing to the limited and relatively preliminary scope of this activity, the design effort was focused exclusively on the HUD and glareshield components -- elements critical to ROTO's implementation, and requiring relatively extensive operational/procedural analysis. Modifications to the FMS page(s), PFD and ND formats, data link, and Autopilot Disengage switch were viewed as straightforward, and were, therefore, not developed in this design effort.

As can be seen in Figure 7.4 the HUD contains a number of ROTO operation-specific display elements. Positioned above the horizon line and in the center of the HUD field are the ROTO and exit number designation and status elements. The word "ROTO" is followed by the system-selected exit (in this case, "1"), and both are enframed by a box symbol. The box is written in a solid line and is labeled, in this example, with the designation "AUTO," indicating ROTO's present status (i.e., being armed to execute a ROTO procedure in which steering is automatically controlled by ROTO). Before the crew had armed ROTO, the box was shown in a dashed line, and the system's "READY" status was displayed. This information is replicated on the glareshield (described subsequently).

To the left of the ROTO/exit number status indication elements, and also above the HUD horizon line, is the Airport and Active Runway designation. This information (also repeated on the glare-shield) is simply provided as a head-up verification that the FMS-specified selection is indeed correct (this feature is a useful safety enhancement element, especially in cases of reduced visibility, problematic ATC communications, areas with closely co-located airports, and for airports with multiple runways).

Two symbols commonly employed (in various forms) on 'glass cockpit' PFDs and HUDs -- the Deceleration Cue (a ">" symbol positioned to the left of the Flight Path Vector symbol) and the Speed Deviation Indicator (a deviation bar rising from, or descending from, the left wing of the Flight Path Vector symbol) -- have been co-opted to support guidance and control capabilities associated with following the ROTO system-calculated speed schedule. As in standard usage, the Deceleration Cue drifts below alignment with the Flight Path Vector symbol's left wing, commensurate with the aircraft's current (ROTO-commanded) deceleration. Precision guidance and/or control is provided by the Speed Deviation Indicator which, in turn, is driven by the autobraking control law. Length of the deviation bar indicates magnitude of the instantaneous speed deviation (from directed speed schedule), and direction (up or down) of the bar indicates instantaneous speed error, with an upward extended bar indicating that the instantaneous speed is above commanded and a downward extended bar indicating below commanded.

The Flight Path Vector symbol, a circle with two down angled "wings" (one to the circle's left and one to its right), operates in a standard HUD-operational manner, indicating the instantaneous flight path of the aircraft. ROTO operation extends the use of this symbol to the indication of the ground path vector. This information facilitates runway and exitway centerline tracking, and assists in exit- and taxiway turn guidance (especially in reduced visibility). The Flight Director symbol (a circle sized to be clearly distinguishable inside the Flight/Ground Path Vector circle) is also used as guidance for pursuit of the desired ROTO path centerline, and is of critical importance for precision tracking in the complex ground turn profiles (spiral profiles) prescribed by ROTO.

Strategic head up awareness of ROTO exit turn parameters is provided by an Exit Turn symbol shown in true-perspective graphical form, super-imposed on the real world exitway visible through the HUD. On final approach, and before ROTO is armed, this symbol is shown as a dashed line conformal with the selected ROTO exitway centerline. Upon ROTO being armed, this line changes from dashed to solid in form. After touch down, the Exit Turn Indication "line" must be modified for clear, unambiguous visibility because of the drastically minimized grazing angle resulting from the flight deck's relatively low height above the runway. This is accomplished by giving the line a dimension of height (forming a low fence-like symbol). Showing the shape of the turn is accomplished by making the 'fence' uniformly segmented; at those points where the curve is increasingly more severe in angle (and is also relatively farther away from the pilot's eye point) the segments (and gaps between them) will be perceived as progressively wider and diminishing in height. (It should be noted here that this solution is viewed as provisional, and is not considered completely satisfactory in its present version. Chief among the concerns regarding this format symbol are the possibility that this element will clutter the HUD's critically important central display field (even with an algorithm for blanking out portions of the element when it competes with other symbology), and that it clearly violates the design decision to have dashed lines indicate Ready status and solid lines indicated Armed.) This centerline information is supplemented by Runway Edge markings to assist the pilot in perceiving position, altitude, track accuracy, speed, and perspective cues.

As an additional source of tactical awareness, Tire Position Indication information is provided at the bottom of the HUD. This data indicates current gear position relative to the runway or exitway edge and the commanded centerline tracking. This information was provided as a safety enhancement employed in correcting for significant deviations from intended centerline tracking that could occur in low visibility, with unanticipated runway/exitway conditions, or as a result of control errors.

The final significant modification to a standard HUD format suite is the automatic removal of all format elements having exclusively airborne operational functions (e.g., the roll indicator). This declutter mode will enable the pilot to more easily attend to and track ROTO's precision deceleration, steering, position, and timing. This decluttering function would itself be automatically reverted to full-up flight information immediately upon the selection of a go-around maneuver.

Crew Procedures for ROTO Operation

In this section, a detailed procedural account is provided depicting crew actions and ROTO system functioning specifically related to the control and display interfaces for ROTO. This description is designed to complement the overall functional timeline developed in the operational/mission requirements activity discussed previously (see Appendix A), and to better articulate the principal crew procedures and coordination efforts involved in ROTO operation.

A Representative Implementation of ROTO Operation

On approach into Los Angeles International Airport (LAX), ATC clears the aircraft for the ILS (or DGPS) approach to Runway 24 Right. The crew is told to expect to be cleared for ROTO later in the approach. The Pilot Not Flying (PNF) accesses the FMS page associated with runway ILS selection, specifies the cleared approach information, and sets the ROTO system for calculation of appropriate deceleration and steering profiles predicated on the system-selected high speed exit. Upon selection of Exit 1, the ROTO system enunciates airport, runway, and exitway designations on the glareshield (for viewing by both crew members) and on the HUD (and on the head down and FMS displays as well). Additionally, ROTO's operational status, "Ready," is indicated. Figure 7.5 portrays this status of the ROTO system while the aircraft is turning to intercept the Localizer signal.

After capture of the localizer and glideslope signals, the aircraft is stabilized on its approach to Runway 24 Right, tracking the Glideslope signal in to the predicted touch down point. As is apparent in figure 7.6, ROTO is still in Ready status, and the runway edge markings and exitway centerline are clearly discernible. The crew awaits clearance for ROTO.

Just prior to the Pilot Flying (PF)'s execution of the flare maneuver in the final seconds of flight, the crew has been cleared for the ROTO procedure and the system has been armed. Figure 7.7 depicts this change in ROTO system status -- ROTO has been armed for Automatic (i.e., system-controlled) operation upon touch down, and the exitway centerline symbology has consequently turned from dashed to solid line format. The PF flares the aircraft and lands.

Upon touch down, the HUD automatically declutters, leaving symbology relevant only to ground maneuvering and to monitoring ROTO progress (or tracking ROTO guidance). As is shown in figure 7.8, the exitway centerline symbol has transitioned to the "fence" format to provide the crew with compelling perspective and angular cues for accurately perceiving (and monitoring or tracking) the complex exitway ground path dictated by ROTO. Just prior to initiation of the exitway turn maneuver, a positive indication of this maneuver's imminent execution is enunciated by a computer-generated voice message. Landing gear position is displayed, indicating current position with regard to the ROTO-specified centerline of track, and the runway and exitway edges.

After the aircraft has cleared the runway and continues the commanded deceleration and steering profiles on the exitway, the PNF requests clearance to enter the upcoming taxiway. In those cases when trailing aircraft are also executing ROTO landings and exits, the clearance will need to be timely. In effect, the ROTO procedure is not completed until the aircraft has entered the taxiway within the time window required by ROTO.

8.0 SIMULATED ROTO PERFORMANCE RESULTS

TIME HISTORIES

Figures 8.1 - 8.12 document ROTO time histories using 30 degree spiral exits (reference 2) while varying aircraft type, surface condition and crosswind. Each aircraft landed at the mean of their dispersion and landing weight as follows:

	MD-11	MD-81
Weight (klb.)	410	105
% CG	23	16.5
Airspeed (knots)	148	126.5
Tailwind (knots) (negative is headwind)	-7.5	-7.5
Touchdown Location relative to Threshold (feet)	1375	1375

Table 1 lists the runway occupancy times for these runs. These runs show satisfactory braking and steering performance. Runway occupancy time is measured from the runway threshold until the aircraft wing tip clears the near side of runway.

Table 1

Run #	Aircraft Type	Surface Condition	Crosswind (knots)	Runway Exit Location (feet)	Runway Occupancy Time (sec)
1	MD-11	Dry	0	4950	38.1
2	MD-11	Wet	15	6750	49.5
3	MD-11	Wet	12.5 +/- 2.5	6750	49.5
4	MD-81	Dry	0	4950	41.6
5	MD-81	Wet	15	4950	41.6
6	MD-81	Wet	12.5 +/- 2.5	4950	41.1

Note: Positive crosswind direction is from left to right for landing aircraft. A crosswind of increasing magnitude causes greater lateral centerline deviation. The simulation studies found a positive crosswind caused greater deviation than a negative crosswind.

Note: Runway Exit Location is relative to runway threshold.

Note: Runway Occupancy Time is calculated from the time the aircraft crosses the runway threshold until the aircraft wing tip clears the near side of the runway.

Each simulation run is documented with two pages of time histories. When a plot shares more than one variable, the second variable is usually plotted on the right hand Y axis. The zero origin of the left and right axis are usually offset so that the variable time histories do not cross each other. The X axis of all plots is the runway longitudinal axis in feet. 0 feet is at the runway threshold.

Page 1; Bottom Plot

This plot shows two views of the aircraft position relative to the runway with a right hand ROTO turnoff. The left axis shows the aircraft Y position in feet. The runway centerline is along the top of the plot. The desired path (dashed line) is along the centerline and then curves to the right as the right-handed ROTO exit. Any small perturbations in the dashed curves represent exit entrances which the aircraft did not enter. The solid line represents the aircraft position. For MD-81 and MD-11 simulation runs the first ROTO exit is at position 3300 feet and 4950 feet, respectively.

The right axis shows the aircraft Y lateral displacement (solid line) in feet from the runway centerline and exit path. The straight-lined funnel shape represents the allowable lateral width in which the aircraft can move without running off the pavement. The funnel width is the runway and ROTO exit widths minus the aircraft main gear offset, which varies with aircraft type.

Page 1; 2nd from Bottom Plot

The left axis plots the aircraft ground speed in knots (decreasing trace). The right axis plots the aircraft runway occupancy time in seconds. The runway occupancy time at touchdown begins at a value greater than zero because it begins counting at the runway threshold. The runway occupancy time stops increasing when the aircraft wing tip clears the near side of the runway.

Page 1; Middle Plot

The left axis plots the aircraft lateral acceleration in G's (lower trace). The right axis plots the aircraft lateral jerk in G/sec. Gust cases do not plot the lateral jerk because it is too excessive. This study did not ascertain the cause of the gust related jerk (simulation model, control laws, sensors) or find a solution for this occurrence.

Page 1; 2nd from Top Plot

The left axis plots the aircraft longitudinal acceleration in G's (lower trace). The right axis plots the aircraft longitudinal jerk in G/sec. Gust cases do not plot the longitudinal jerk because it is too excessive. This study did not ascertain the cause of the gust related jerk (simulation model, control laws, sensors) or find a solution for this occurrence.

Page 1; Top Plot

The left axis plots the percent of main gear brake supply pressure commanded (lower trace). When the plot shows 100%, the deceleration command is commanding all of the brake supply pressure. The percent of brake supply pressure commanded does not reflect the amount of brake supply pressure in use if anti-skid (required by ROTO) is active.

Please refer to the 3rd and 4th plots on the bottom of plot page 2 for the amounts of available mu being used by the main gear. When runway surface friction decreases below that required (resulting in skidding), anti-skid decreases brake pressure used just until skidding is alleviated. One would not expect 100% supply pressure in use when braking at high speeds on a wet surface. The ROTO simulation used in this study implemented the anti-skid function in the drag code (for modeling complexity reasons), after its proper location in the brake pressure code.

The right axis plots the aircraft total thrust in pounds (upper trace).

Page 2; Bottom Plot

The left axis plots the aircraft rudder position in degrees (lower trace). The right axis plots the nose gear position in degrees.

Page 2; 2nd from Bottom Plot

The left axis plots the amount of μ being used by the aircraft nose gear (lower trace). The right axis plots the available aircraft nose gear μ.

Page 2; 3rd from Bottom Plot

The left axis plots the amount of μ being used by the aircraft main right gear μ (lower trace). The right axis plots the available aircraft main right gear μ.

Page 2; 4th from Bottom Plot

The left axis plots the amount of μ being used by the aircraft main center gear μ (lower trace). The right axis plots the available aircraft main center gear μ.

Page 2; 3rd from Top Plot

The left axis plots the aircraft track angle relative to the aircraft heading in degrees. The right axis plots the aircraft elevator angle in degrees (gradually rising trace).

Page 2; 2nd from Top Plot

The left axis plots the steady tailwind in knots. A headwind would have a negative value. The right axis plots the crosswind in knots. If the crosswind is steady it will have a straight line value. Gust cases will show a varying crosswind. A positive crosswind blows in a negative Y to positive Y direction (left to right as viewed by a landing aircraft).

Page 2; Top Plot

The left axis plots the navigation X position data noise content (lower trace). The right axis plots the navigation Y position data noise content.

DISPERSION RESULTS

The ROTO FORTRAN simulation was modified with an outer loop to allow multiple runs. Approximately 900 AUTO ROTO runs were simulated that covered the range of expected aircraft ground speeds and touchdown locations, spaced 2 knots and 100 feet apart respectively. The two data values used for dispersion analysis from these runs were the runway occupancy time (ROT) and the ROTO exit location used by the aircraft.

An effort was then made to calculate the relative probability of one run occurring relative to the others. Assuming that the aircraft landing ground speed and touchdown locations are normally distributed and independent of each other, the calculations were done as follows:

1. The aircraft landing ground speed mean and standard deviation were created by adding an aircraft's landing airspeed and expected wind means and variances, respectively, and then taking the square root of the summed variance to obtain the standard deviation.
2. The combined effect of aircraft landing ground speed and touchdown location on the relative probability of an individual simulation run occurring was calculated by creating a normal probability density function (PDF) for each of the two individual variables. This was done by subtracting a cumulative density function (CDF) from the next CDF spaced 2 knots apart for the ground speed variable and 100 feet apart for the touchdown location variable. A joint PDF, based on the two variables, was created by multiplying the individual PDF values together at the intersection values of ground speed and touchdown location for each run. A normal CDF is calculated as follows:

$$\frac{1}{\sqrt{2\pi\sigma}} \exp^{-\left(\frac{(x-\mu)^2}{2\sigma^2}\right)}$$

The joint normal PDF then represented the relative probability of a run occurring based on the aircraft landing ground speed and touchdown location.

By multiplying a simulation run's ROT and ROTO exit location by the probability of the run occurring, ROT and ROTO exit location statistics could be calculated. Mean, standard deviation and PDF's were then calculated for ROT and ROTO exit location. A spreadsheet was used to manipulate the data and create the figures.

Figure Descriptions

There are five types of figures referred to below. The 3-D plots show the ROT value for the simulations (e.g. fig. 8-14), the exit location used (e.g. fig. 8-15), and the joint probability density function (PDF) of touchdown location and TD ground speed (e.g. fig. 8-16). The ROT figure generally shows that the slow and early, and last and fast, landing aircraft have the high ROT values. The Exit location figure shows the exit locations used for the simulations runs.

The other two figures are the PDF's for Exit location (e.g. fig. 8-17) and ROT (e.g. fig. 8-18). The Y axis represents the relative occurrences of one outcome (exit used or ROT value) to the others. The X axis of the Exit location figure is the possible runway exit locations in feet. The X axis of the ROT figure is the possible rounded (to nearest second) ROT values in seconds. PDF figures were also created for Exit location and ROT assuming equal probabilities for each simulation run.

Results

The following table lists the figure numbers for the figures described above:

Figure Descriptions	MD-11 Wet	MD-11 Dry	MD-81 Wet
ROT	8-13	8-21	8-27
Exit Location	8-15	8-22	8-28
Joint PDF of simulation inputs	8-16	8-16	8-29
Exit Location PDF	8-17	8-23	8-30
ROT PDF	8-18	8-24	8-31
Exit Location PDF assuming equal probabilities	8-19	8-25	8-32
ROT PDF assuming equal probabilities	8-20	8-26	8-33
ROT showing incorrect exit predictions	8-14		

Figure 8-13 uses the same data as figure 8-14, but does not include the spiked ROTs caused by 4 incorrect exit predictions. The exit prediction logic used for all the simulation runs was continuously improved during this study, but needs further development to further minimize incorrect exit predictions.

Comparison of the Exit & ROT results show that mean and standard deviations decrease in the order of MD-11/Wet, MD-11/Dry to MD-81/Wet as shown in the table below. The lower standard deviation of MD-81 ROT and Exit # taken may be explained by the fact that the MD-81 has smaller standard deviations for its touchdown location and TD ground speed.

Aircraft/Conditions	Exit # Taken; Exit 0 = 3300 ft		ROT (seconds)	
	Mean	STDDEV	Mean	STDDEV
MD-11/Wet	1.65	0.53	45.0	4.04
MD-11/Dry	1.30	0.46	41.3	3.99
MD-81/Wet	0.95	0.40	41.3	3.50

The ROT PDF shows grouped ROTs for each exit location, represented by each line style. The left most line style is the 3300 and 4950 foot exit locations, respectively, for the MD-11 and MD-81 aircraft. The Exit location and ROT PDF's both show that the MD-11/Dry aircraft make more use of the 4950 foot exit than the MD-11/Wet. The MD-81/Dry aircraft would most likely use the 3300 foot exit more often than the MD-81/Wet. The Exit location and ROT PDF's using equal probability for all runs show all of the exits being more equally utilized.

The following table shows the percent of MD-11 & MD-81 AUTO ROTO aircraft using an exit.

Aircraft/Conditions	Exit Locations Used by Aircraft (feet)			
	3300	4950	6750	8000
MD-11/Wet	0%	37.8%	59.8%	2.4%
MD-11/Dry	0%	70.0%	30.0%	0%
MD-81/Wet	10.4%	84.0%	5.6%	0%

It appears that a 3 ROTO exit (@ 3300, 4950 & 6750 feet past runway threshold) runway would satisfy 99.8% (3 sigma dispersion) of the MD-11 Dry and MD-81 Wet landings. These exits would satisfy ~97.5% (> 2 sigma dispersion = 95.5%) of the MD-11 Wet landings.

9.0 CONCLUSIONS AND RECOMMENDATIONS

A ROTO architecture and control laws have been designed for a research auto and manual ROTO system capable of CATIIIB conditions. System additions to most aircraft would include DGPS, FMS exit geometry data, data link, one HUD display, glareshield button(s), backdriven tiller, ROTO auto & manual control laws and auto variable braking. To achieve the runway occupancy time performance documented in this report; aircraft would also need auto-reverse thrust (includes backdriven throttle levers), exit prediction capability and consistent reverse thrust stowing techniques based on ground speed.

Control law designs developed in this study were validated with a 3-DOF aircraft dynamic simulation using representative landing gear, engine, rudder actuator, and nose wheel actuator models. Approximately 900 simulation runs were performed varying the touchdown point, landing speed, aircraft weight and center-of-gravity, wind speeds (headwind/tailwind, crosswinds, and gusts), and runway surface conditions (wet & dry). In addition, the simulations were run using two different weight class aircraft – an MD-81 and MD-11 aircraft – using the same guidance and control system. This report documents representative time histories of these simulations. The study developed ROTO statistics from the simulated data including probabilities on runway occupancy time and the use of the given available high-speed exits.

This study recommended ROTO exit positions (@ 3300, 4950 & 6750 feet past runway threshold) for MD-11 and MD-81 type aircraft used in this study. These exits satisfy from 2 to 3 sigma touchdown dispersions for aircraft landing weights ranging from light MD-81's up to heavy MD-11's. The maximum runway occupancy time for the 6750 foot exit was under 53 seconds for a heavy MD-11 on a wet/non-grooved concrete surface. This study did not investigate other operational airport factors which may not allow for consistent runway occupancy times less than 53 seconds.

This study assumed that ROTO exit entrances would be spiral in nature as described in reference 2. This allows for a more smooth transition onto the exit, but adds 1 to 2 seconds to the runway occupancy time. It is recommended that ROTO exits be grooved with a diamond pattern. If possible, ROTO runways should also be grooved.

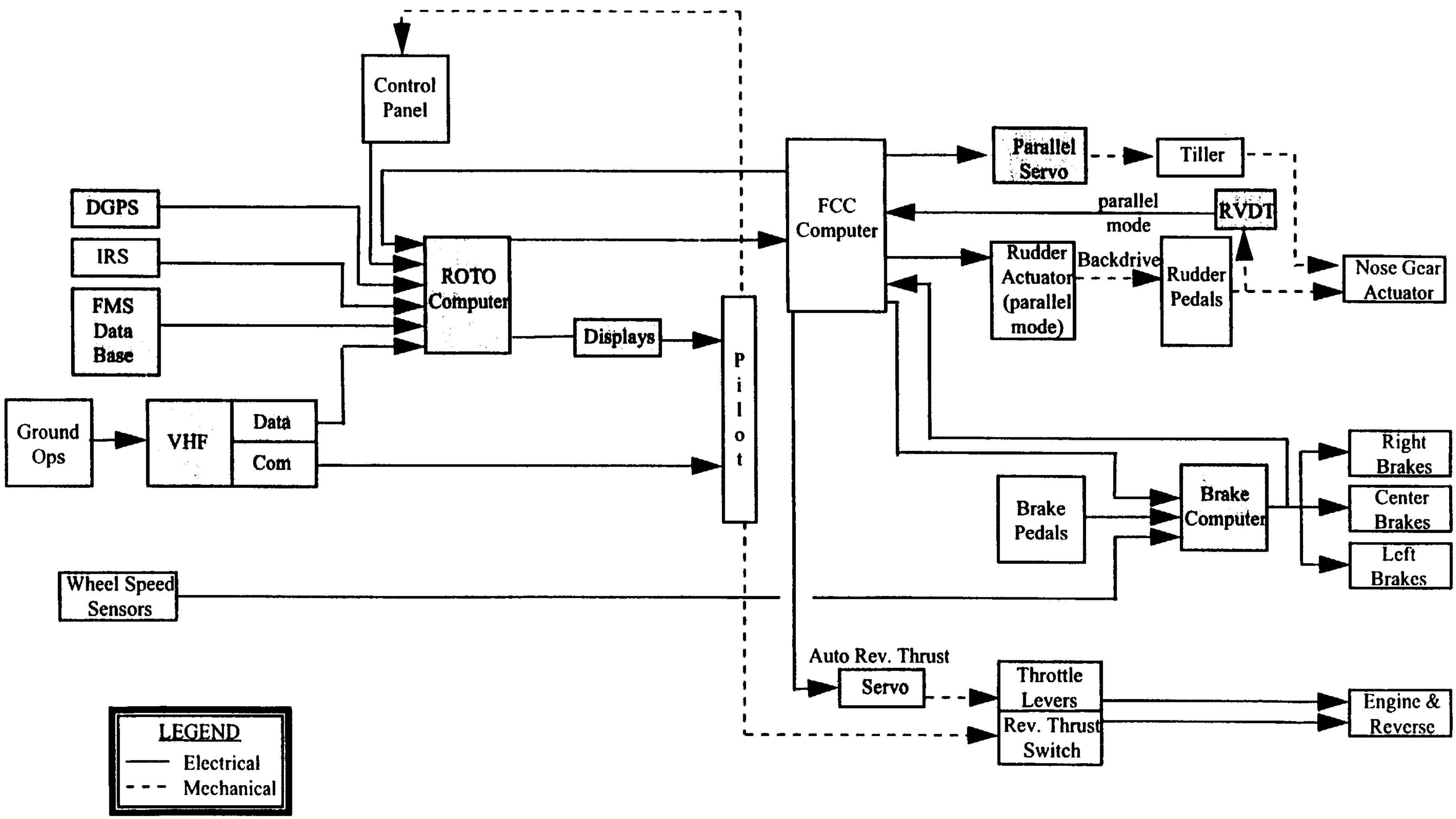
Auto asymmetric braking and fly-by-wire nose gear are no longer required for aft CG MD-11 aircraft on a wet surface as mentioned in reference 1. An anti-hysteresis tiller control law, commanding the tiller to augment the auto ROTO nose gear, was needed to decrease the hysteresis to 2 degrees. It is believed that the tiller control law would cost less than the fly-by-wire nose gear alternative.

Future study is required to find the sensitivity of sub-optimal ROTO designs and operations on runway occupancy time. Dynamic piloted evaluation is required of display format/guidance elements, look-ahead limits for CATIIIB operations and continuos ROTO operations with runway occupancy times less than 53 seconds. Functions required by the exit prediction logic need further development: accurate runway friction measurement, predicted runway touchdown location/ground speed and general exit prediction algorithm.

REFERENCES

1. Goldthorpe, Kernik, McBee, Preston, GUIDANCE AND CONTROL REQUIREMENTS FOR HIGH-SPEED ROLLOUT AND TURNOFF (ROTO), NCR 195026, January 1995, NAS1-19703 Task 3, Langley Research Center, Hampton, VA, NASA.
2. Anon, AIRPORT DESIGN STANDARDS - AIRPORTS SERVED BY AIR CARRIERS - TAXIWAYS, Advisory Circular AC No. 150/5300 - B, U S Department of Transportation, Federal Aviation Administration, Washington DC, 5/9/80.
3. Anon, PAVEMENT GROOVING AND TRACTION STUDIES, NASA SP-5073, Langley Research Center, Hampton, VA, NASA, November 1968.
4. Anon, PERFORMANCE INFORMATION FOR OPERATION WITH WATER, SLUSH, SNOW, OR ICE ON THE RUNWAY, AC No. 91-6B, FAA, June 1986.
5. Anon, RUNWAY SURFACE CONDITION SENSOR SPECIFICATION GUIDE, AC No. 150/5220-13B, FAA 3/27/91.
6. Norrbom, Bo and Fistedt, Knut, STUDIES CONCERNING SNOW, ICE AND SLUSH ON RUNWAYS, FFA Memo 106, The Aeronautical Reserch Institude of Sweden, Stockholm, 1975.
7. Horne, Walter B., PAPER-STATUS OF RUNWAY SLIPPERINESS RESEARCH, NASA SP-416, Hampton, VA, October 1976.
8. Yager, Thomas J., Vogler, William A., and Baldasare, Paul., SUMMARY REPORT OF AIRCRAFT AND GROUND VEHICLE FRICTION CORRELATION TEST RESULTS OBTAINED UNDER WINTER RUNWAY CONDITIONS DURING JOINT FAA/NASA RUNWAY FRICTION PROGRAM, NASA TM-100506, March 1988. NASA SP-416, Hampton, VA, October 1976.
9. Anon, MEASUREMENT, CONSTRUCTION, AND MAINTENANCE OF SKID RESISTANT AIRPORT PAVEMENT SURFACES, AC No. 150/5320-12B, FAA, Washington, DC, 11/21/91.
10. Anon, AIRPORT WINTER SAFETY AND OPERATIONS, AC No. 150/5200-30, FAA, Washington, DC 4/20/88.

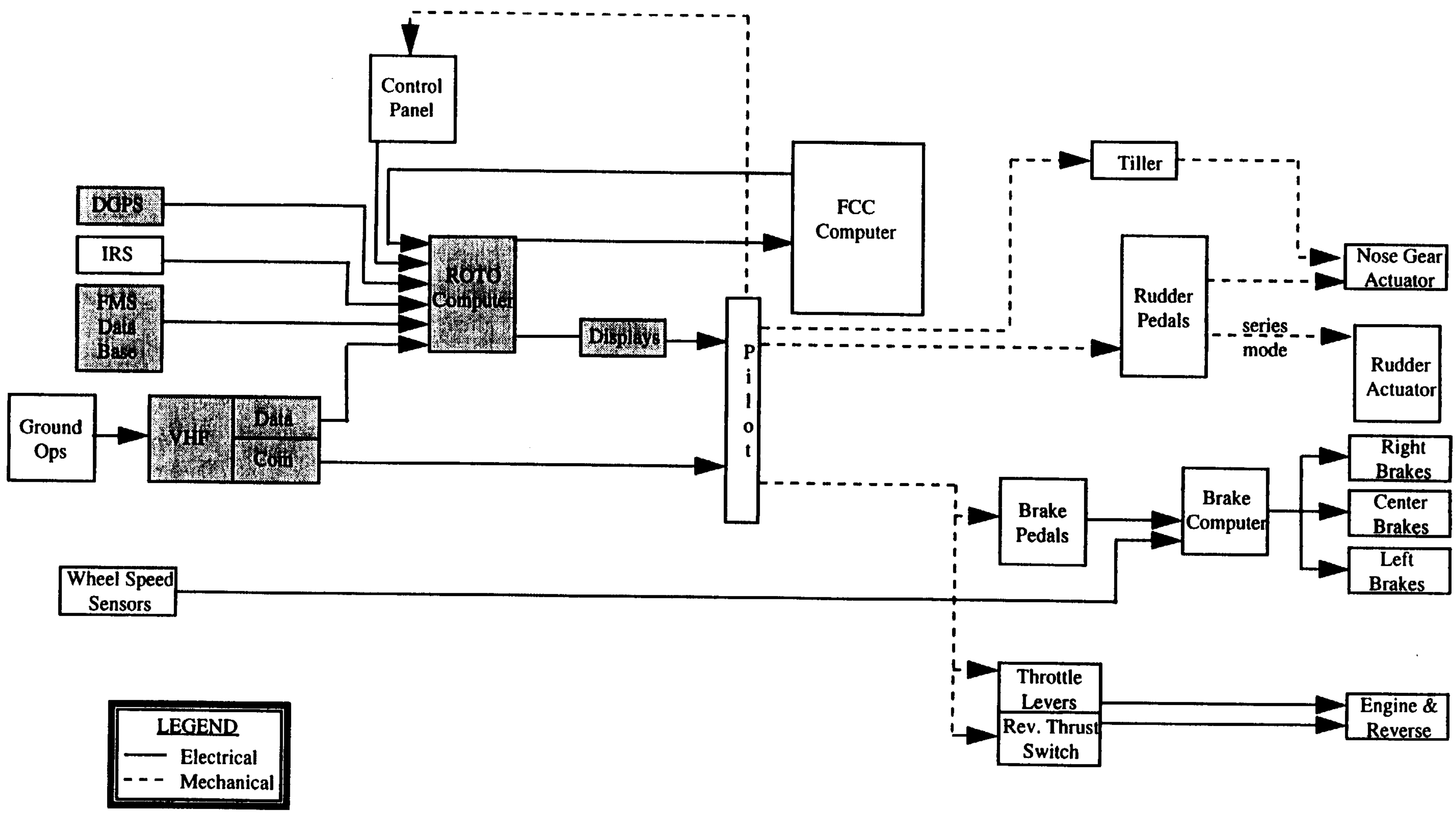
BASELINE AUTO HIGH-SPEED ROLLOUT / TURNOFF CONTROL SYSTEM



The ROTO Computer may be part of the FCC Computer.
The Pilot may override the brake pedals, throttle levers, rudder pedals and/or tiller.

Figure 3.1

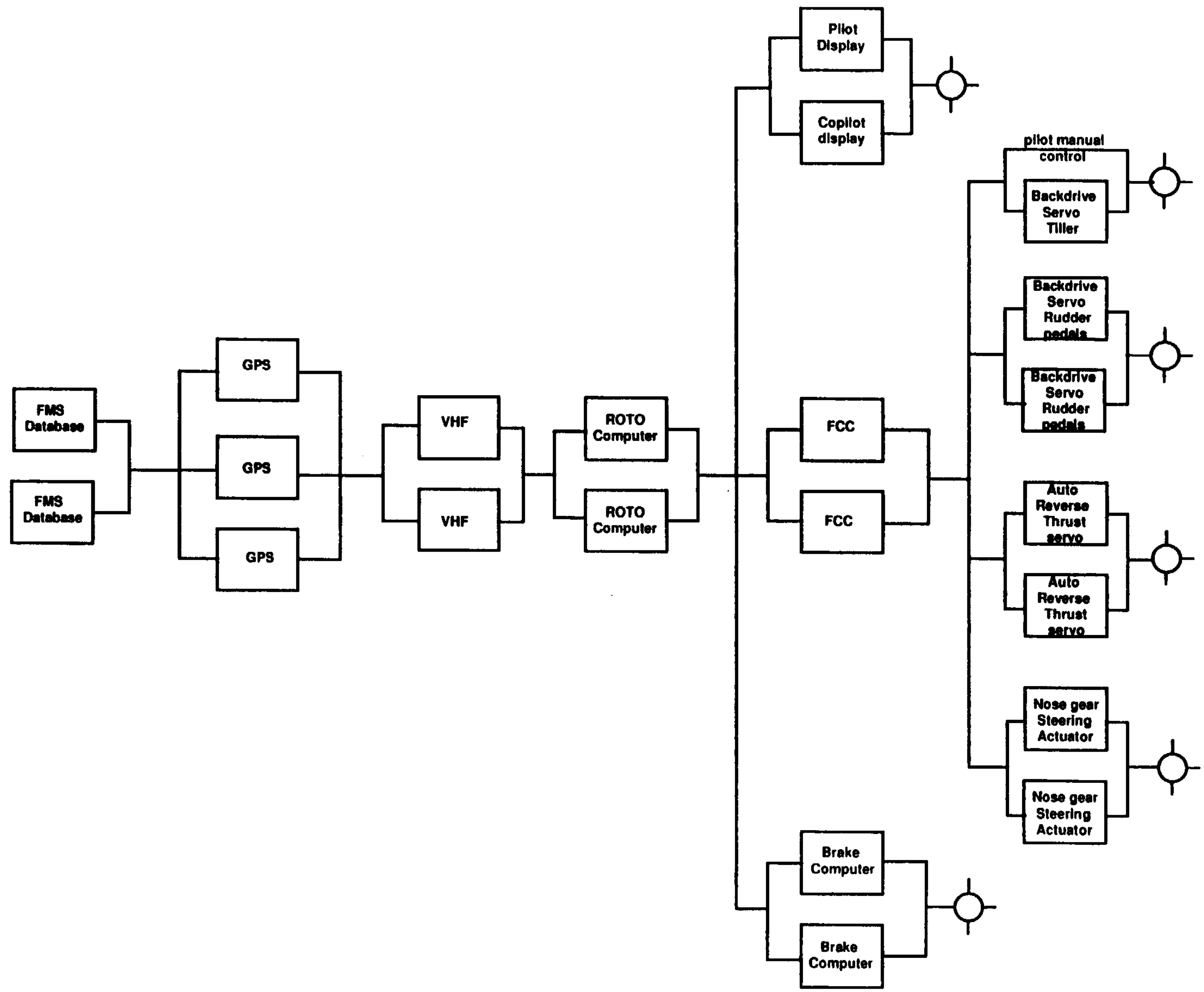
BASELINE FULL MANUAL HIGH-SPEED ROLLOUT / TURNOFF CONTROL SYSTEM



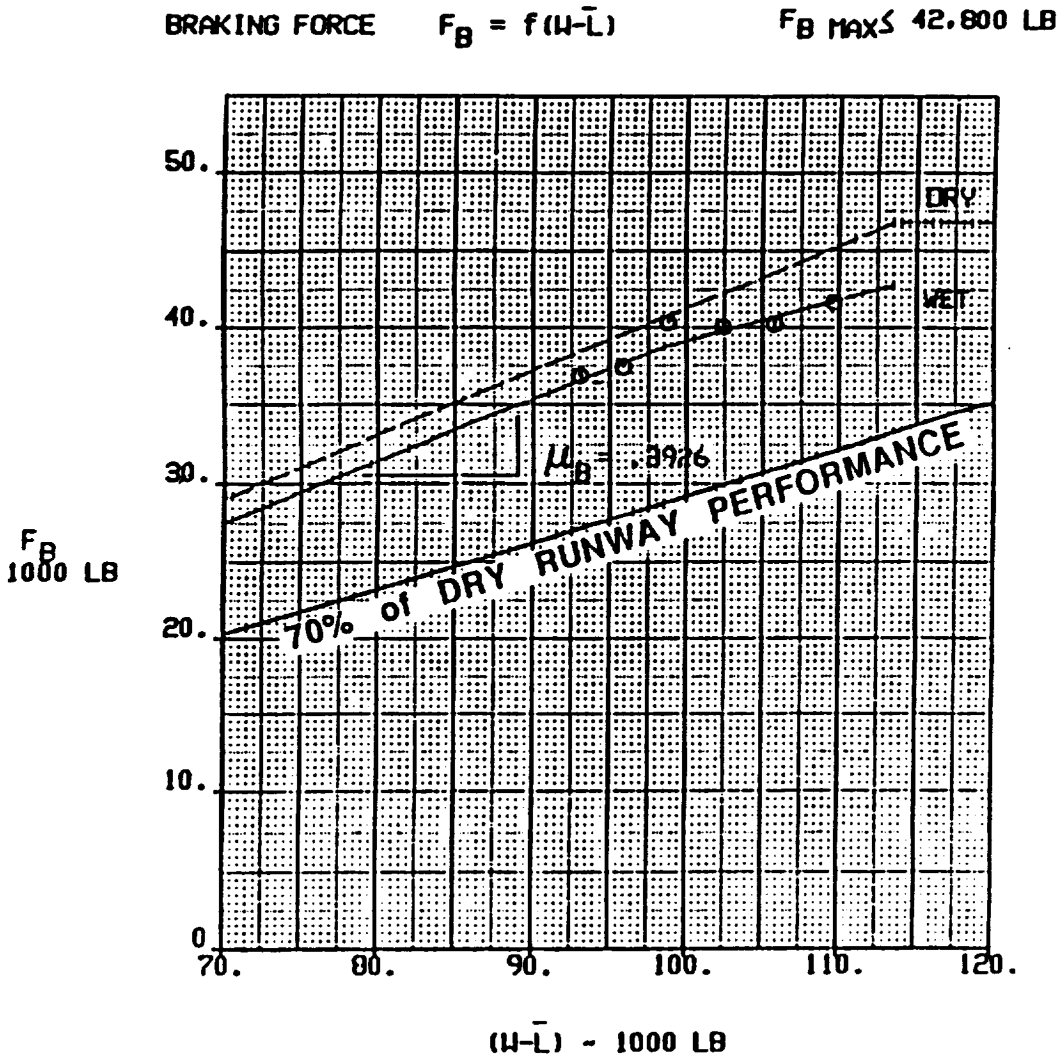
The ROTO Computer may be part of FCC Computer.

Figure 3.2

FIGURE 3.3. ROTO SYSTEM RELIABILITY BLOCK DIAGRAM



ANTISKID AND ALL BRAKES OPERATIVE
WET, GROOVED (OR EQUIVALENT) RUNWAY
FLAP 40



WET GROOVED RUNWAY CERTIFICATION TESTS FOR B737

Figure 6.1

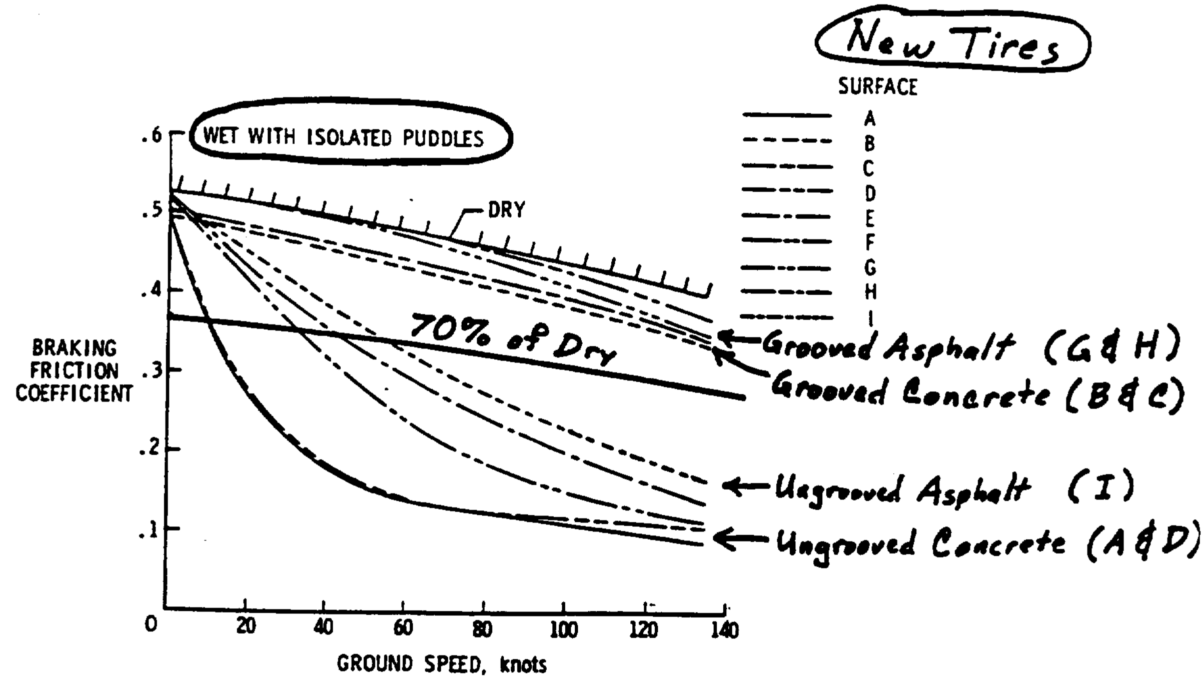


Figure 16.- Variation of 990 aircraft braking friction coefficient with ground speed on wet grooved and ungrooved surfaces. 5-groove, type VIII, 41 × 15.0-18 main tires; inflation pressure, 160 lb/in^2.

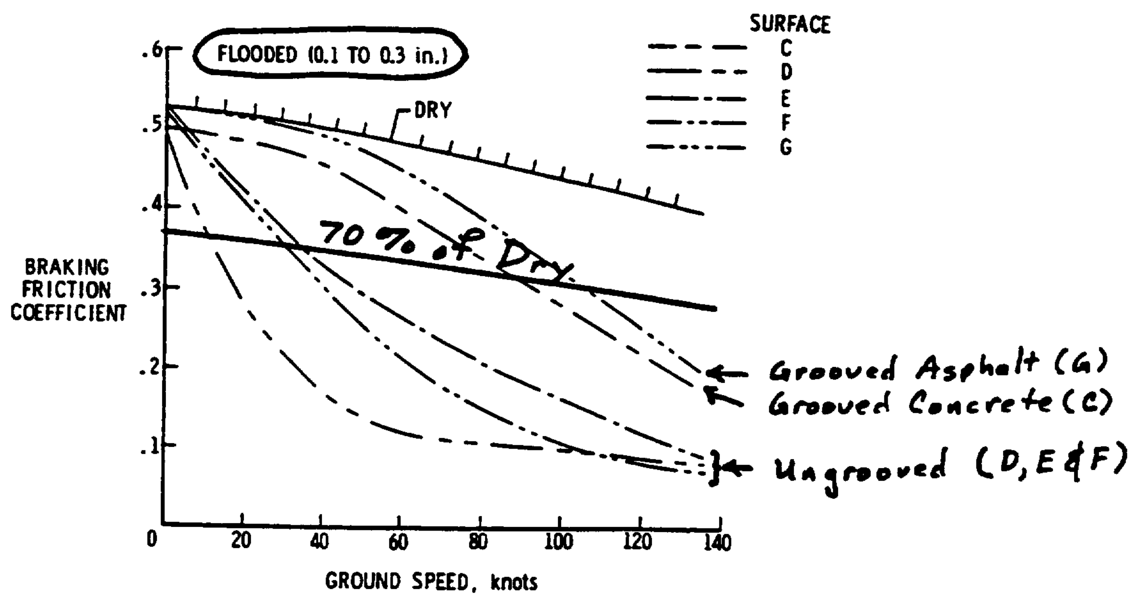


Figure 17.- Variation of 990 aircraft braking friction coefficient with ground speed on flooded grooved and ungrooved surfaces. 5-groove, type VIII, 41 × 15.0-18 main tires; inflation pressure, 160 lb/in^2.

Selected Pages from NASA Report SP-5073 (Ref 3)

Figure 6.2

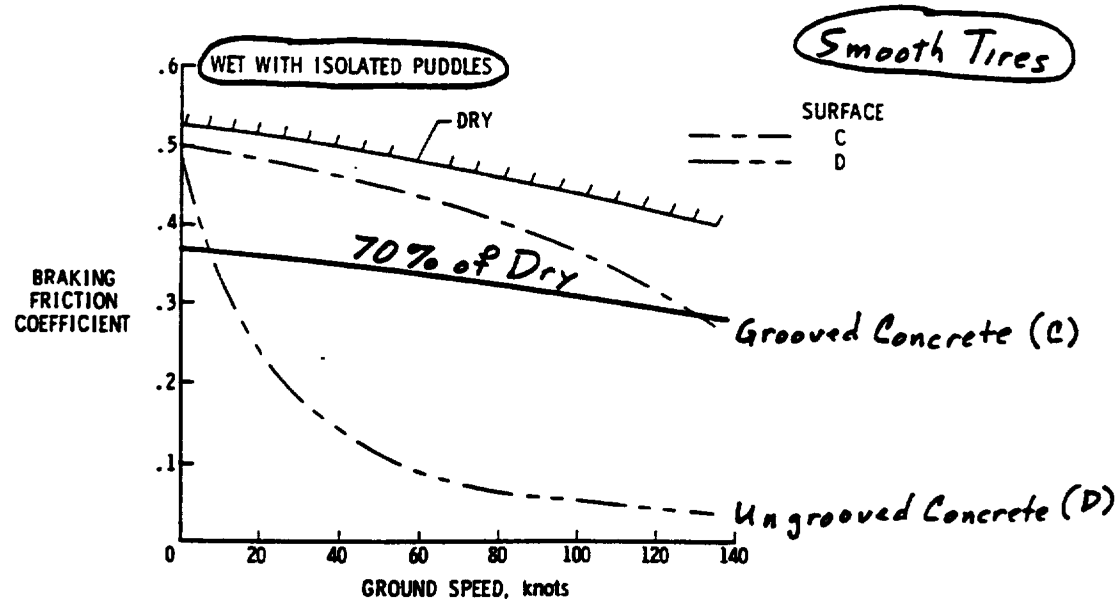


Figure 18.- Variation of 990 aircraft braking friction coefficient with ground speed on wet grooved and ungrooved surfaces. Smooth, type VIII, 41 × 15.0-18 main tires; inflation pressure, 160 lb/in².

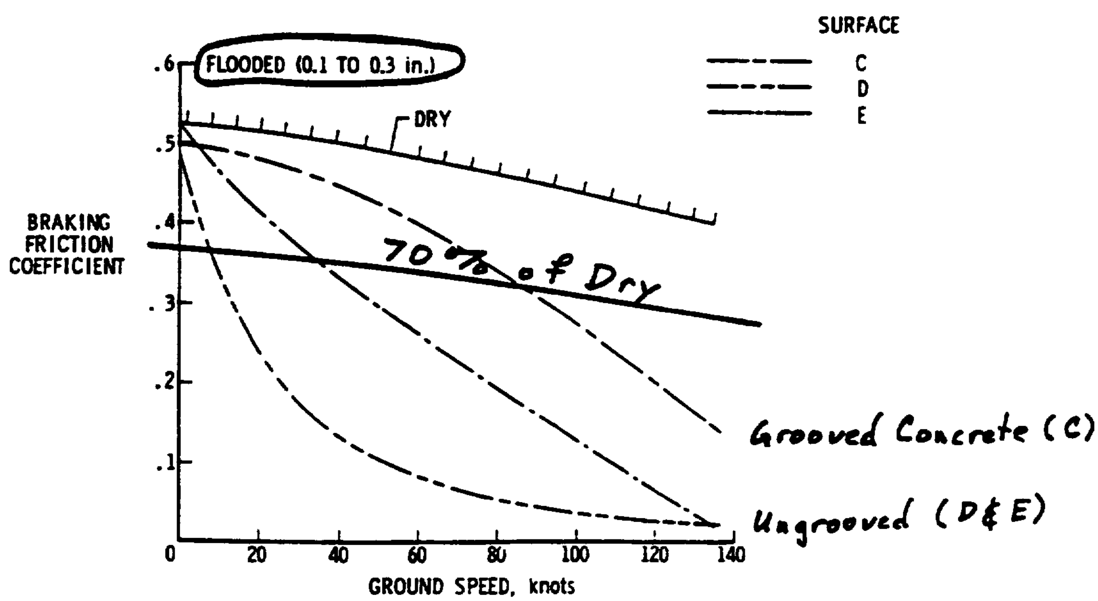


Figure 19.- Variation of 990 aircraft braking friction coefficient with ground speed on flooded grooved and ungrooved surfaces. Smooth, type VIII, 41 × 15.0-18 main tires; inflation pressure, 160 lb/in².

Selected Pages from NASA Report SP-5073 (Ref 3)

Figure 6.3

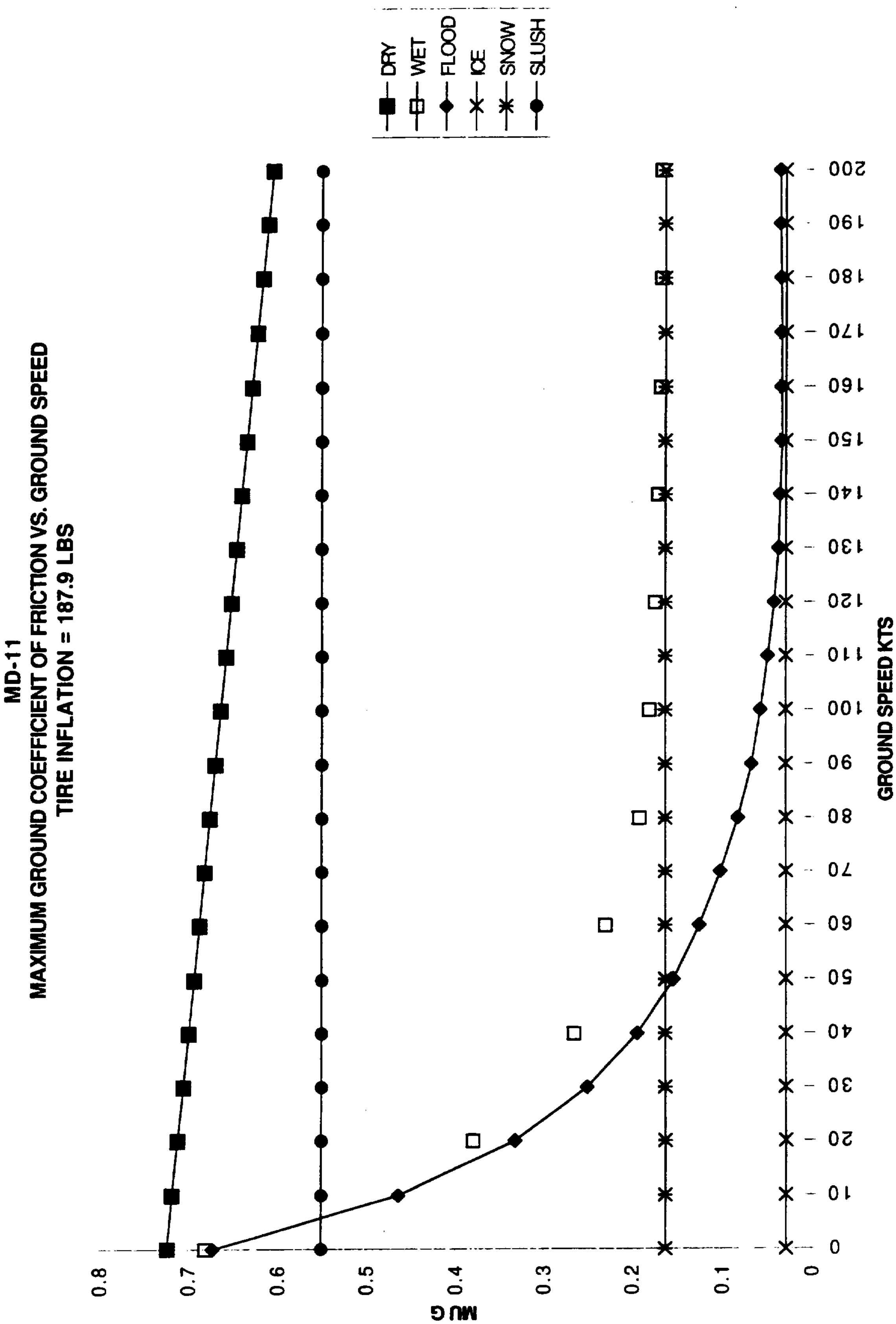

MD-11
MAXIMUM GROUND COEFFICIENT OF FRICTION VS. GROUND SPEED
TIRE INFLATION = 187.9 LBS
DRY
WET
FLOOD
ICE
SNOW
SLUSH
MU G
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
GROUND SPEED KTS
0
10
20
30
40
50
60
70
80
90
100
110
120
130
140
150
160
170
180
190
200

Figure 6.4

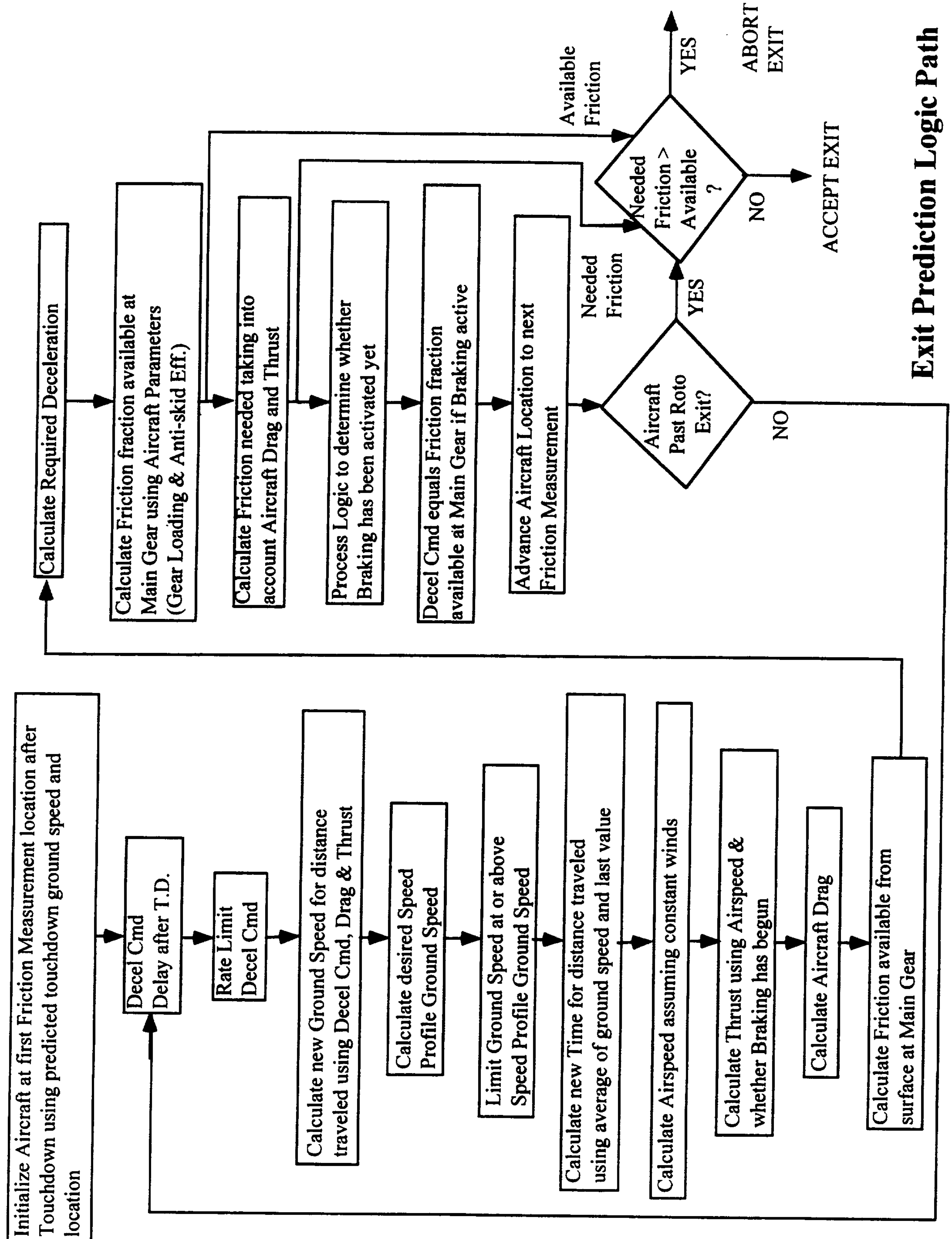

Exit Prediction Logic Path
Initialize Aircraft at first Friction Measurement location after Touchdown using predicted touchdown ground speed and location
Decel Cmd Delay after T.D.
Rate Limit Decel Cmd
Calculate new Ground Speed for distance traveled using Decel Cmd, Drag & Thrust
Calculate desired Speed Profile Ground Speed
Limit Ground Speed at or above Speed Profile Ground Speed
Calculate new Time for distance traveled using average of ground speed and last value
Calculate Airspeed assuming constant winds
Calculate Thrust using Airspeed & whether Braking has begun
Calculate Aircraft Drag
Calculate Friction available from surface at Main Gear
Calculate Required Deceleration
Calculate Friction fraction available at Main Gear using Aircraft Parameters (Gear Loading & Anti-skid Eff.)
Calculate Friction needed taking into account Aircraft Drag and Thrust
Process Logic to determine whether Braking has been activated yet
Decel Cmd equals Friction fraction available at Main Gear if Braking active
Advance Aircraft Location to next Friction Measurement
Aircraft Past Roto Exit?
YES
NO
Needed Friction
Available Friction
Needed Friction > Available ?
YES
NO
ABORT EXIT
ACCEPT EXIT

Figure 7.1

ROTO MODE DIAGRAM

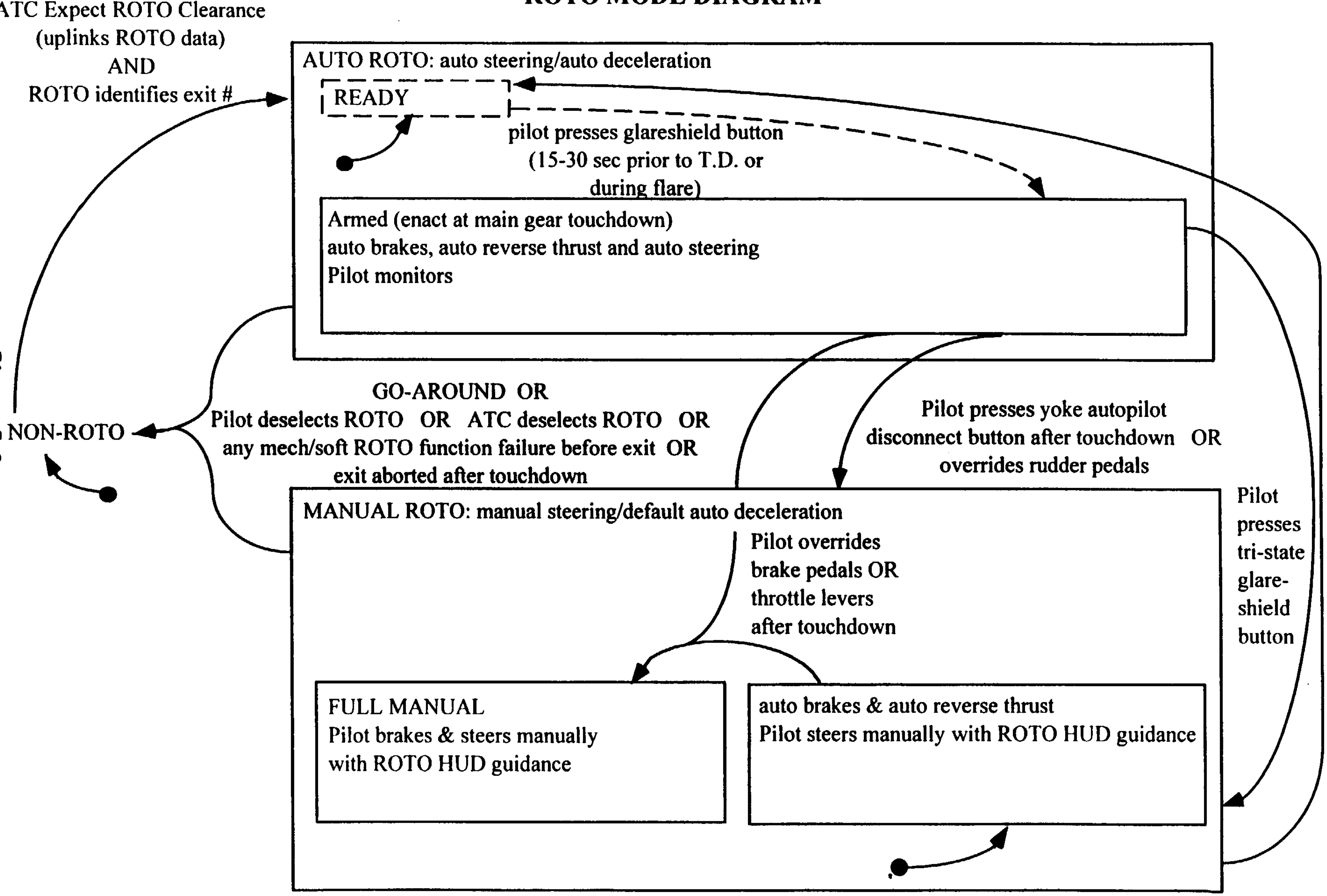


Figure 7.2

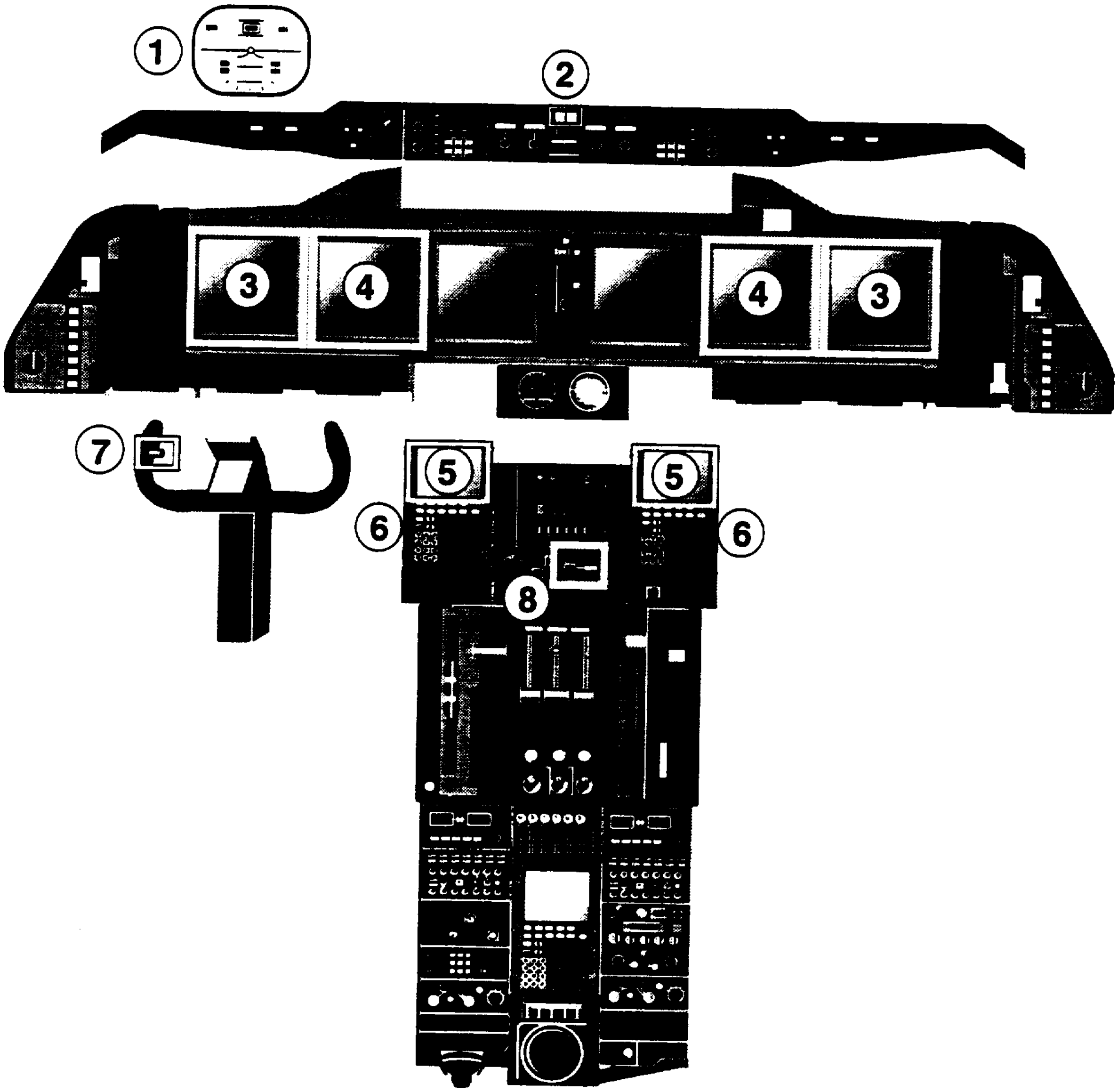


Figure 7.3. ROTO control and display elements

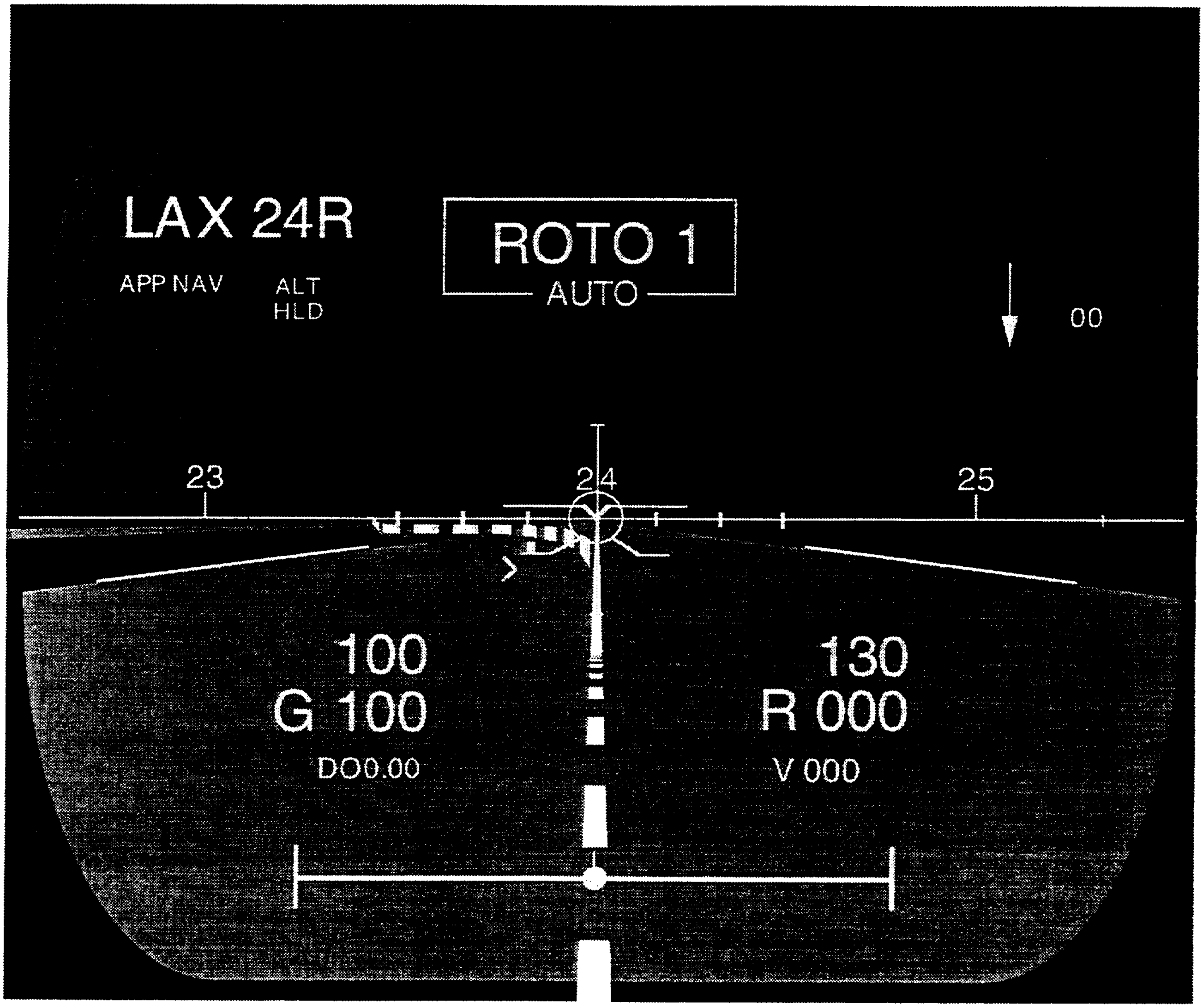


Figure 7.4. ROTO-operational Head Up Display elements

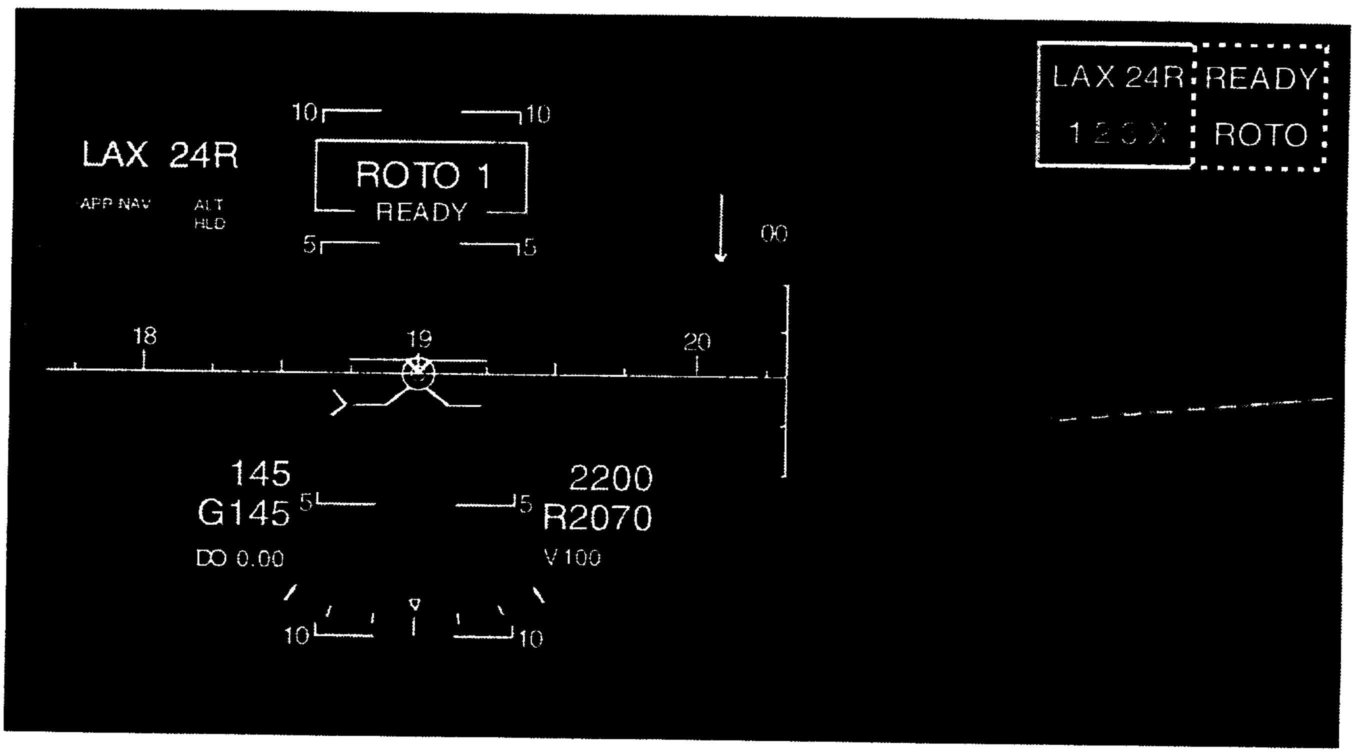


Figure 7.5. Terminal area operational elements for ROTO implementation: Localizer intercept

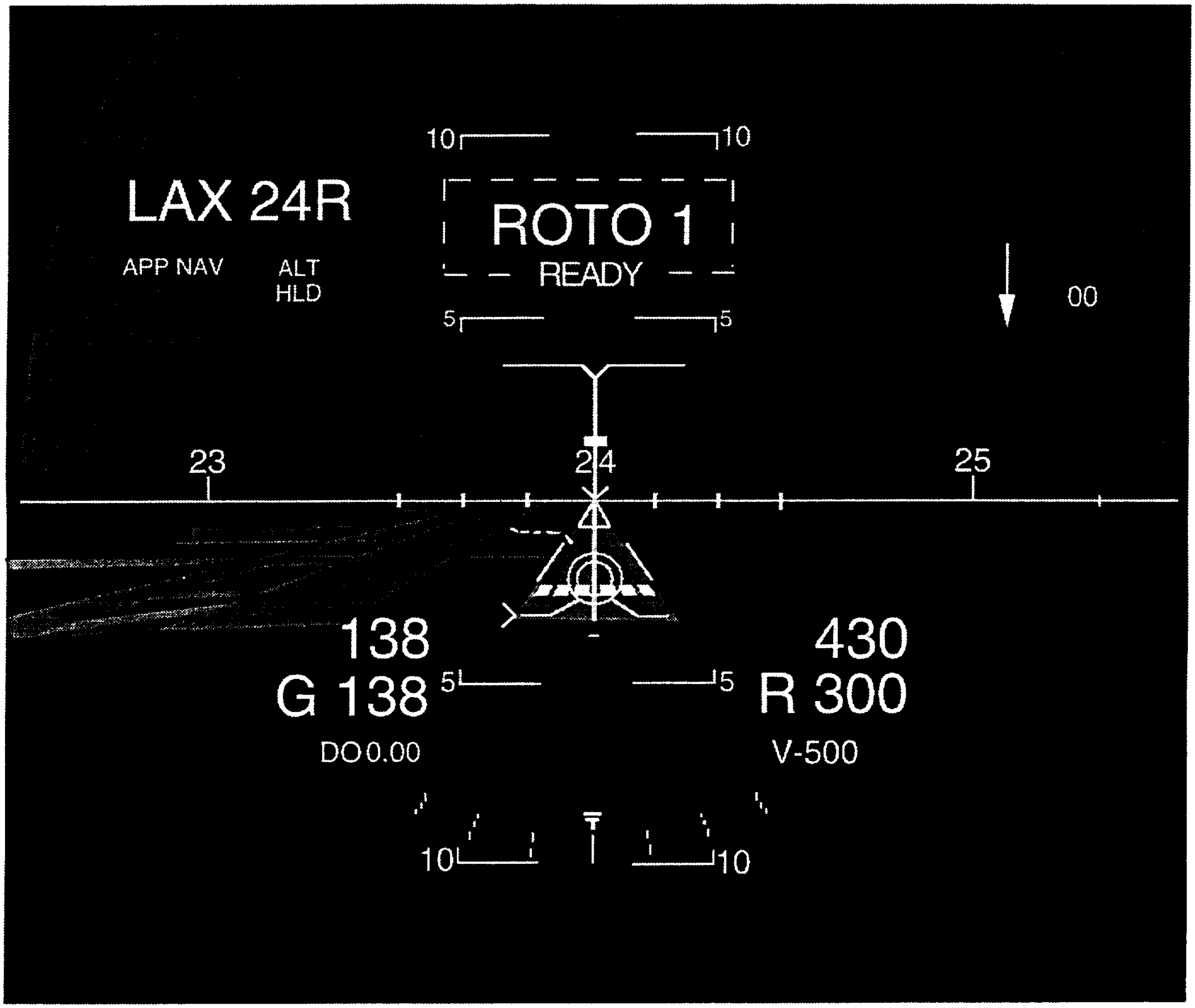


Figure 7.6. Terminal area operational elements for ROTO implementation: ILS capture, on glideslope

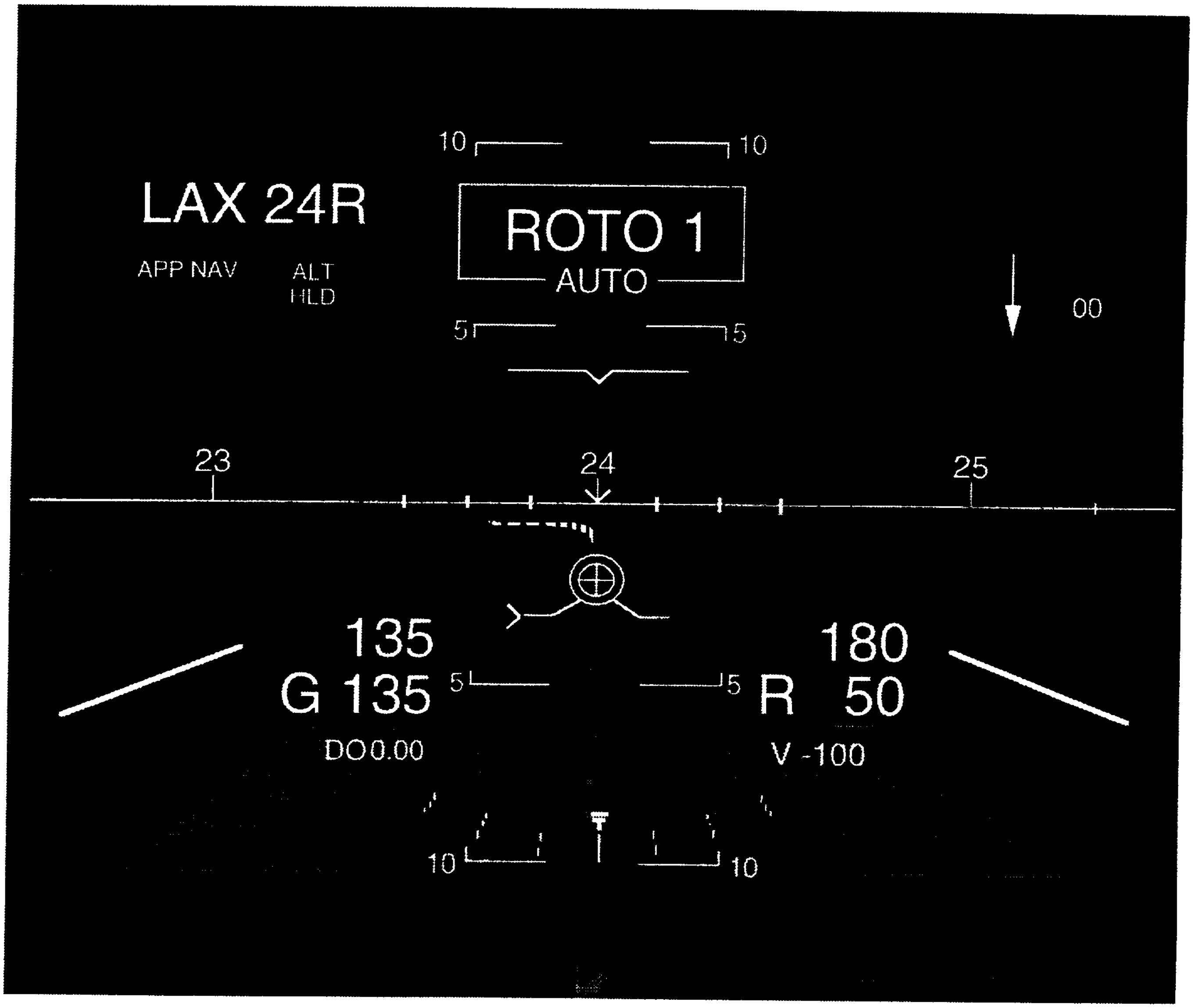


Figure 7.7. Terminal area operational elements for ROTO implementation: At aircraft flare maneuver

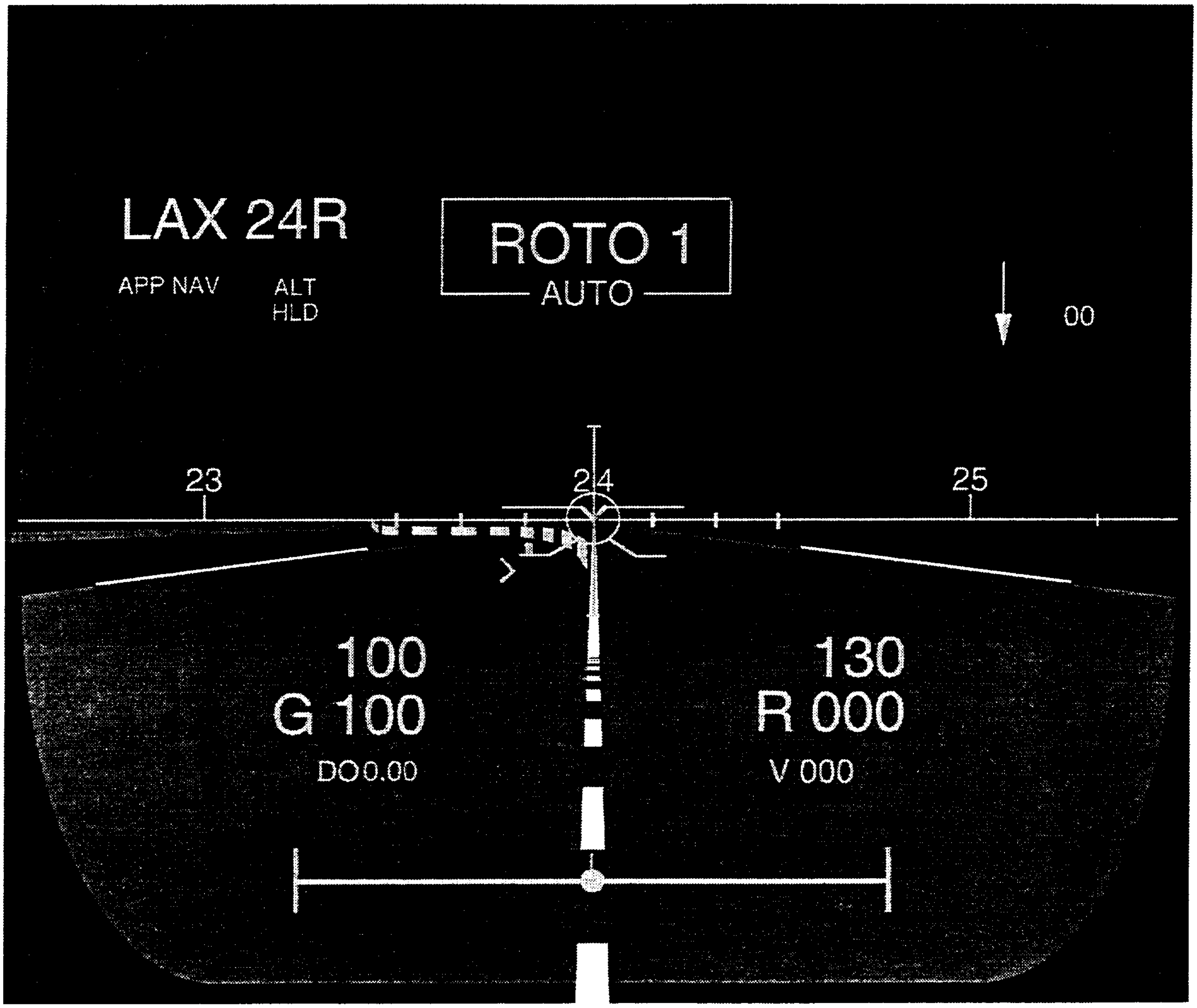
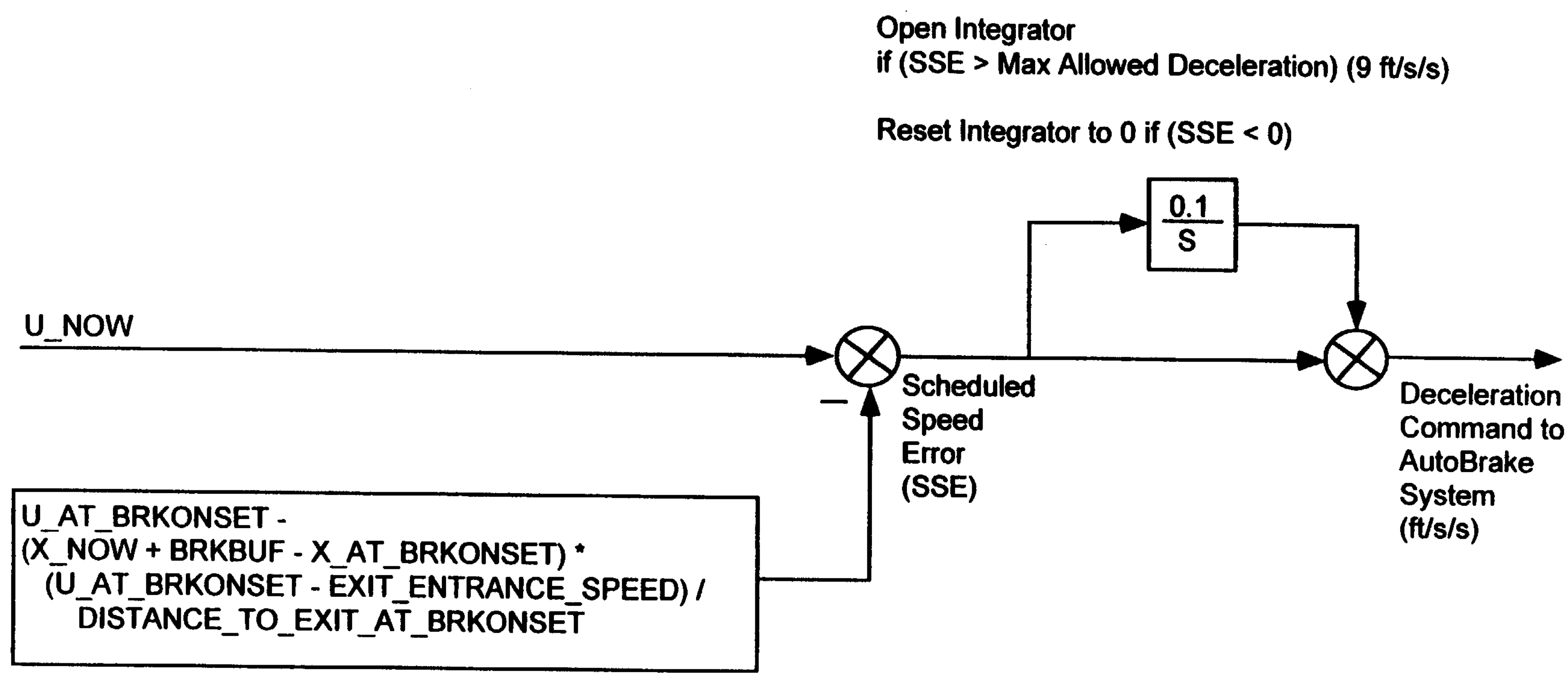


Figure 7.8. Terminal area operational elements for ROTO implementation: Runway roll out



U = AIRCRAFT LONGITUDINAL SPEED
X = AIRCRAFT LONGITUDINAL LOCATION
BRKBUF = BRAKING BUFFER DISTANCE BEFORE EXIT ENTRANCE
EXIT_ENTRANCE_SPEED = VEXIT
DISTANCE_TO_EXIT = BRKDIST

Figure 7.9 Simplified ROTO Auto-Braking Control Law

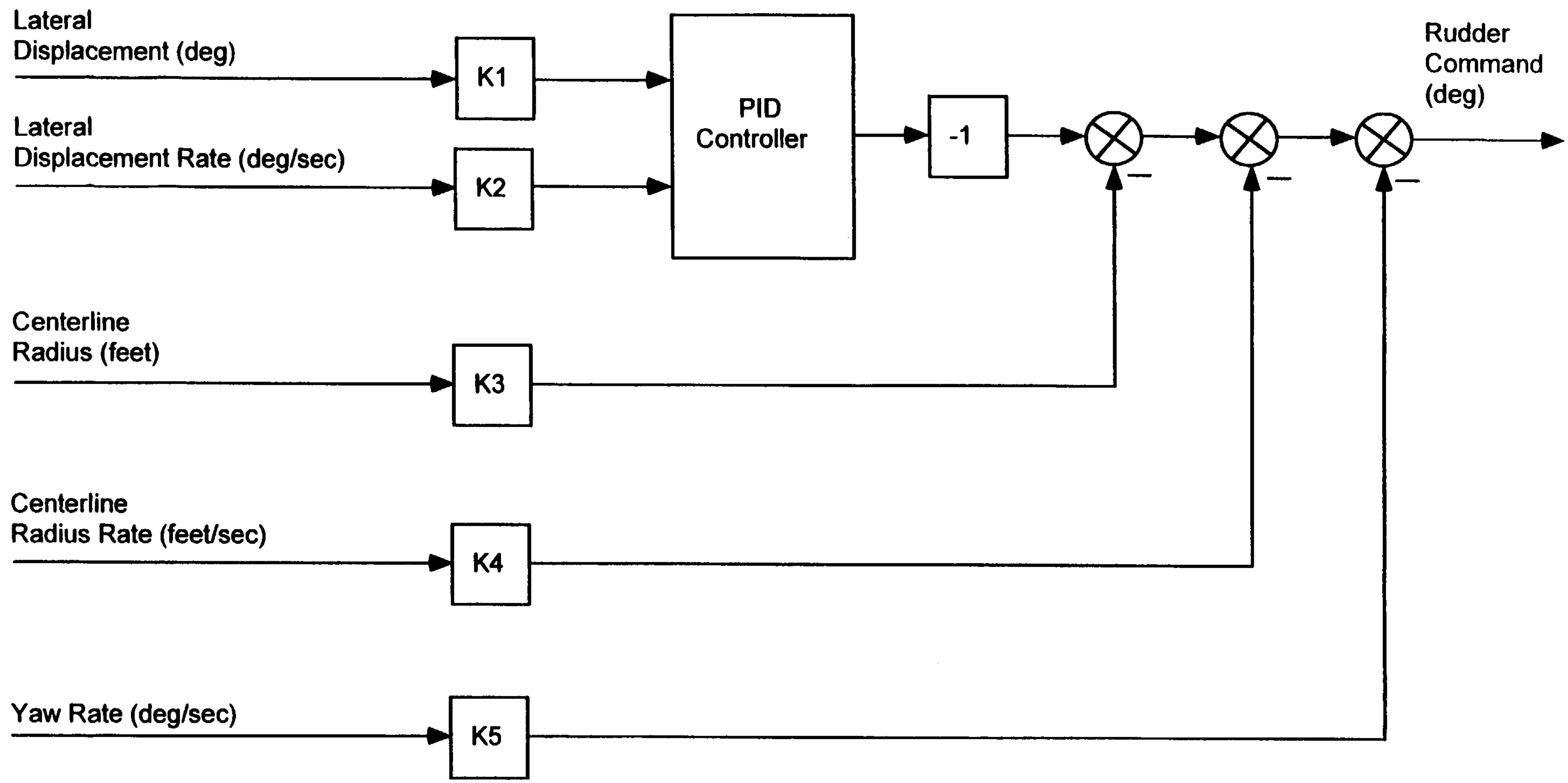


Figure 7.10 Simplified ROTO Steering Control Law

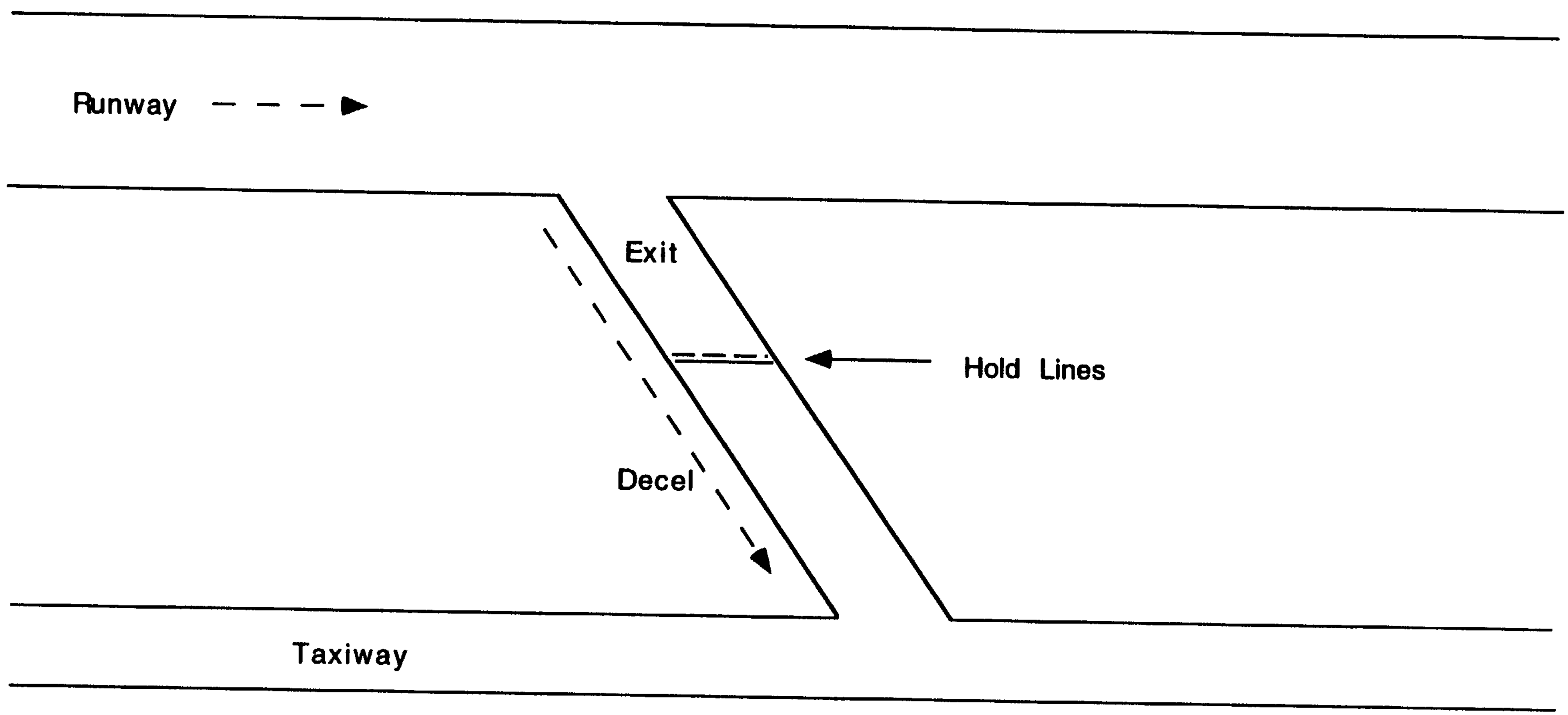


Figure 7.11 Hold Lines depicted on ROTO Exit

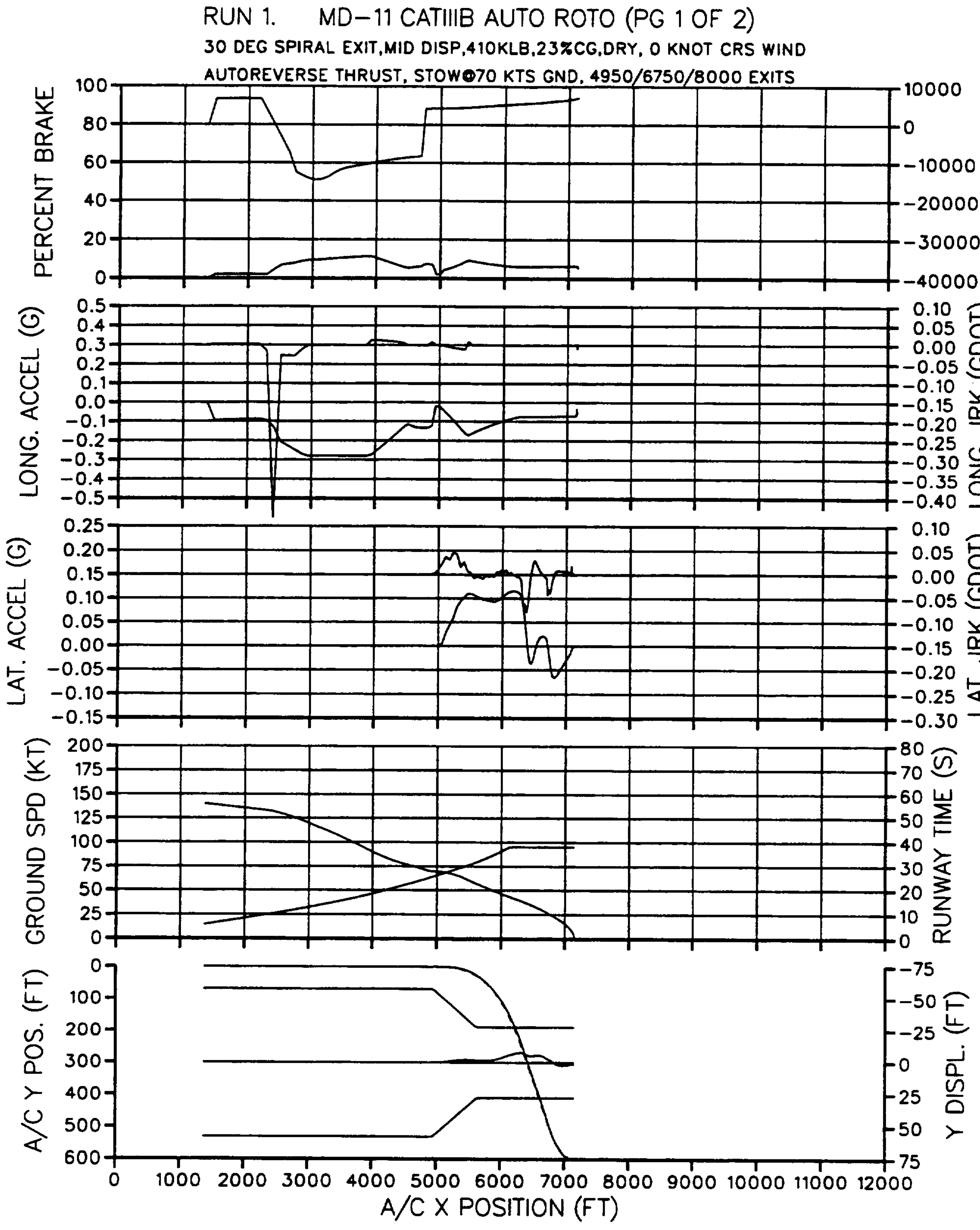

RUN 1. MD-11 CATIIIB AUTO ROTO (PG 1 OF 2)
30 DEG SPIRAL EXIT,MID DISP,410KLB,23%CG,DRY, 0 KNOT CRS WIND
AUTOREVERSE THRUST, STOW@70 KTS GND, 4950/6750/8000 EXITS
PERCENT BRAKE
THRUST (LBS)
LONG. ACCEL (G)
LONG. JRK (GDOT)
LAT. ACCEL (G)
LAT. JRK (GDOT)
GROUND SPD (KT)
RUNWAY TIME (S)
A/C Y POS. (FT)
Y DISPL. (FT)
A/C X POSITION (FT)

Figure 8.1

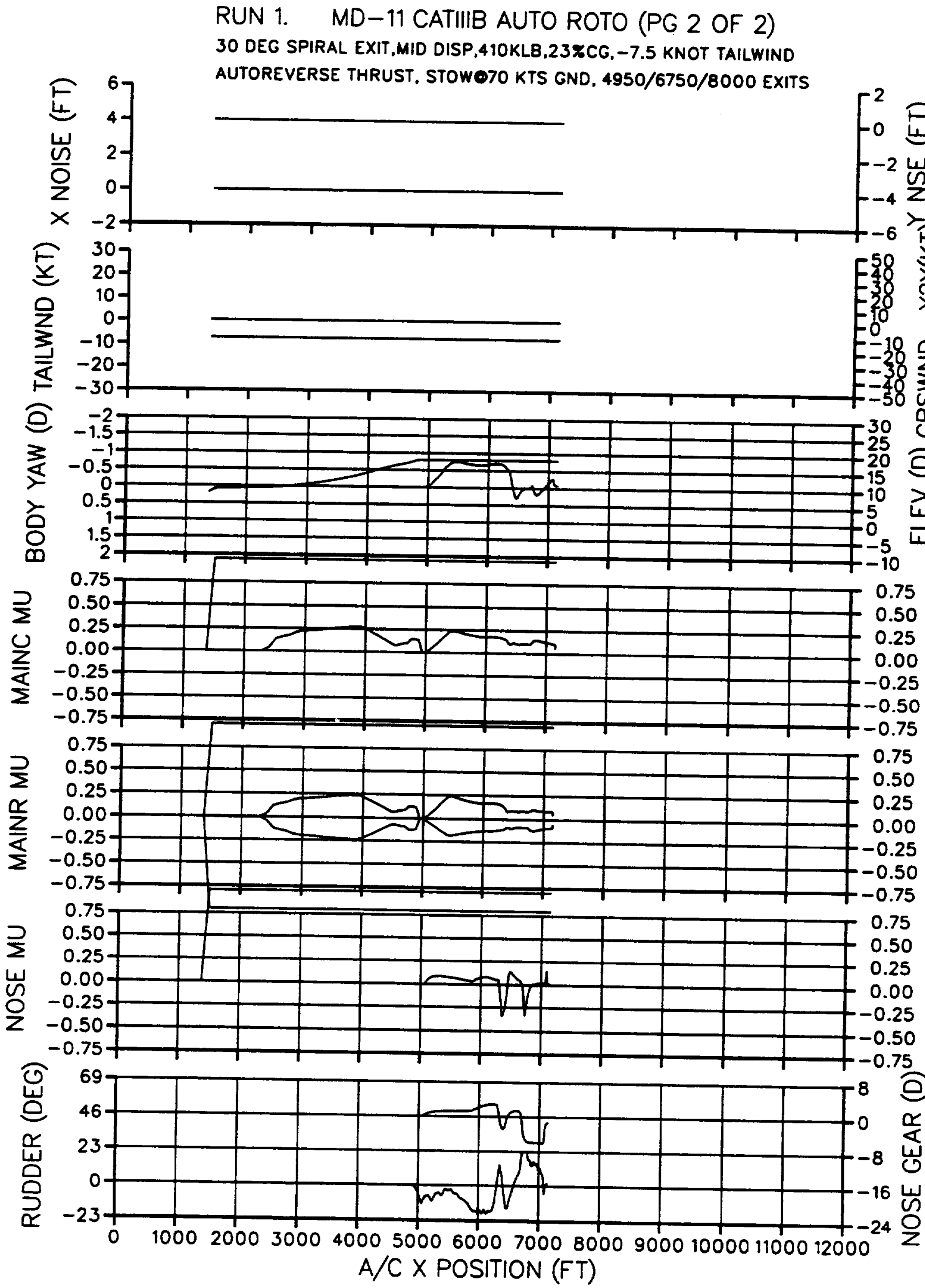
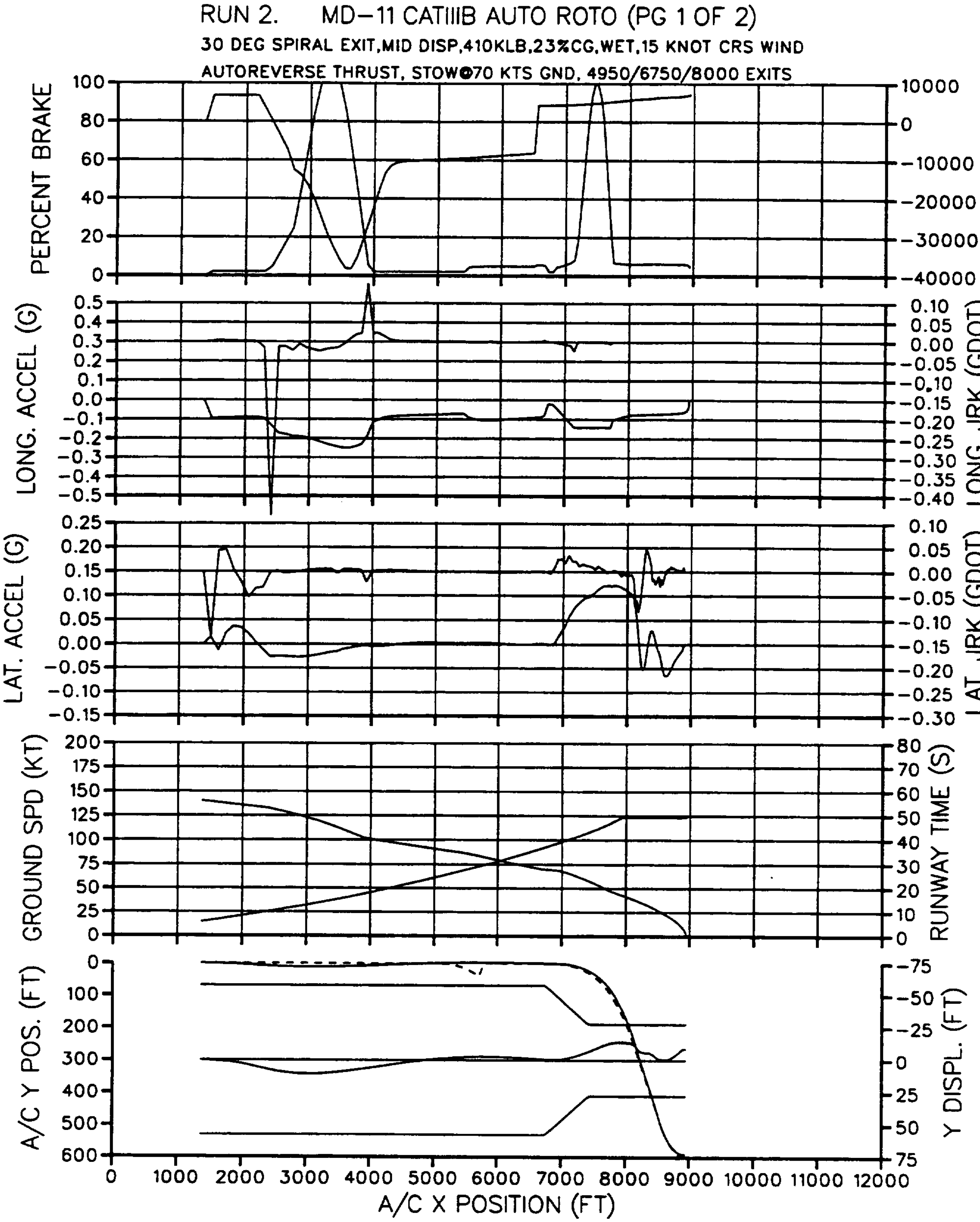

RUN 1. MD−11 CATIIIB AUTO ROTO (PG 2 OF 2)
30 DEG SPIRAL EXIT,MID DISP,410KLB,23%CG,−7.5 KNOT TAILWIND
AUTOREVERSE THRUST, STOW@70 KTS GND, 4950/6750/8000 EXITS
X NOISE (FT)
Y NSE (FT)
TAILWND (KT)
CRSWND −Y2Y(KT)
BODY YAW (D)
ELEV (D)
MAINC MU
MAX C MU
MAINR MU
−MAINL MU
NOSE MU
MAX N MU
RUDDER (DEG)
NOSE GEAR (D)
0
1000
2000
3000
4000
5000
6000
7000
8000
9000
10000
11000
12000
A/C X POSITION (FT)

Figure 8.2



RUN 2. MD-11 CATIIIB AUTO ROTO (PG 1 OF 2)
30 DEG SPIRAL EXIT,MID DISP,410KLB,23%CG,WET,15 KNOT CRS WIND
AUTOREVERSE THRUST, STOW@70 KTS GND, 4950/6750/8000 EXITS
PERCENT BRAKE
THRUST (LBS)
LONG. ACCEL (G)
LONG. JRK (GDOT)
LAT. ACCEL (G)
LAT. JRK (GDOT)
GROUND SPD (KT)
RUNWAY TIME (S)
A/C Y POS. (FT)
Y DISPL. (FT)
A/C X POSITION (FT)

Figure 8.3

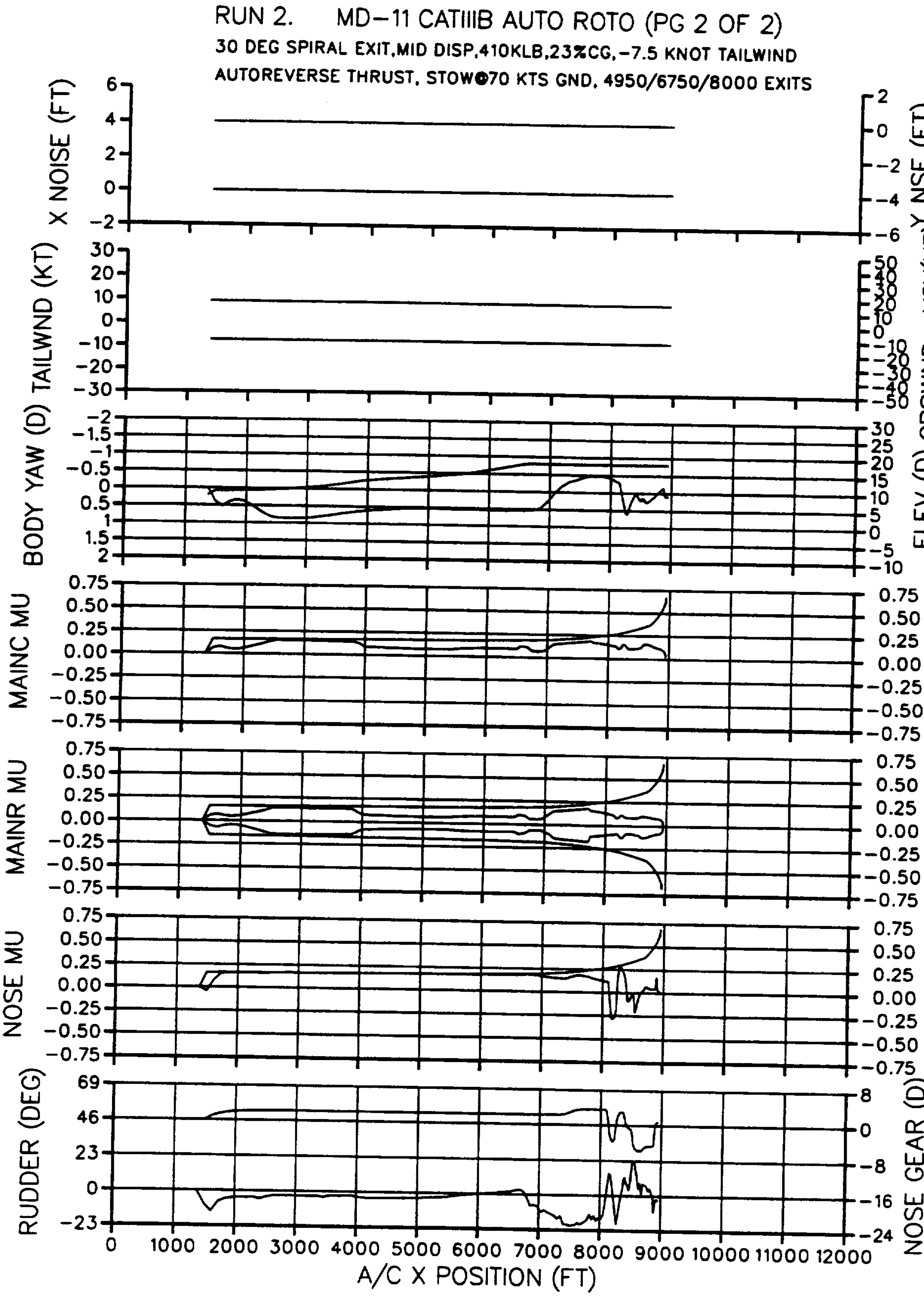

RUN 2. MD-11 CATIIIB AUTO ROTO (PG 2 OF 2)
30 DEG SPIRAL EXIT,MID DISP,410KLB,23%CG,-7.5 KNOT TAILWIND
AUTOREVERSE THRUST, STOW@70 KTS GND, 4950/6750/8000 EXITS
X NOISE (FT)
Y NSE (FT)
TAILWND (KT)
CRSWND -Y2Y(KT)
BODY YAW (D)
ELEV (D)
MAINC MU
MAX C MU
MAINR MU
-MAINL MU
NOSE MU
MAX N MU
RUDDER (DEG)
NOSE GEAR (D)
A/C X POSITION (FT)

Figure 8.4

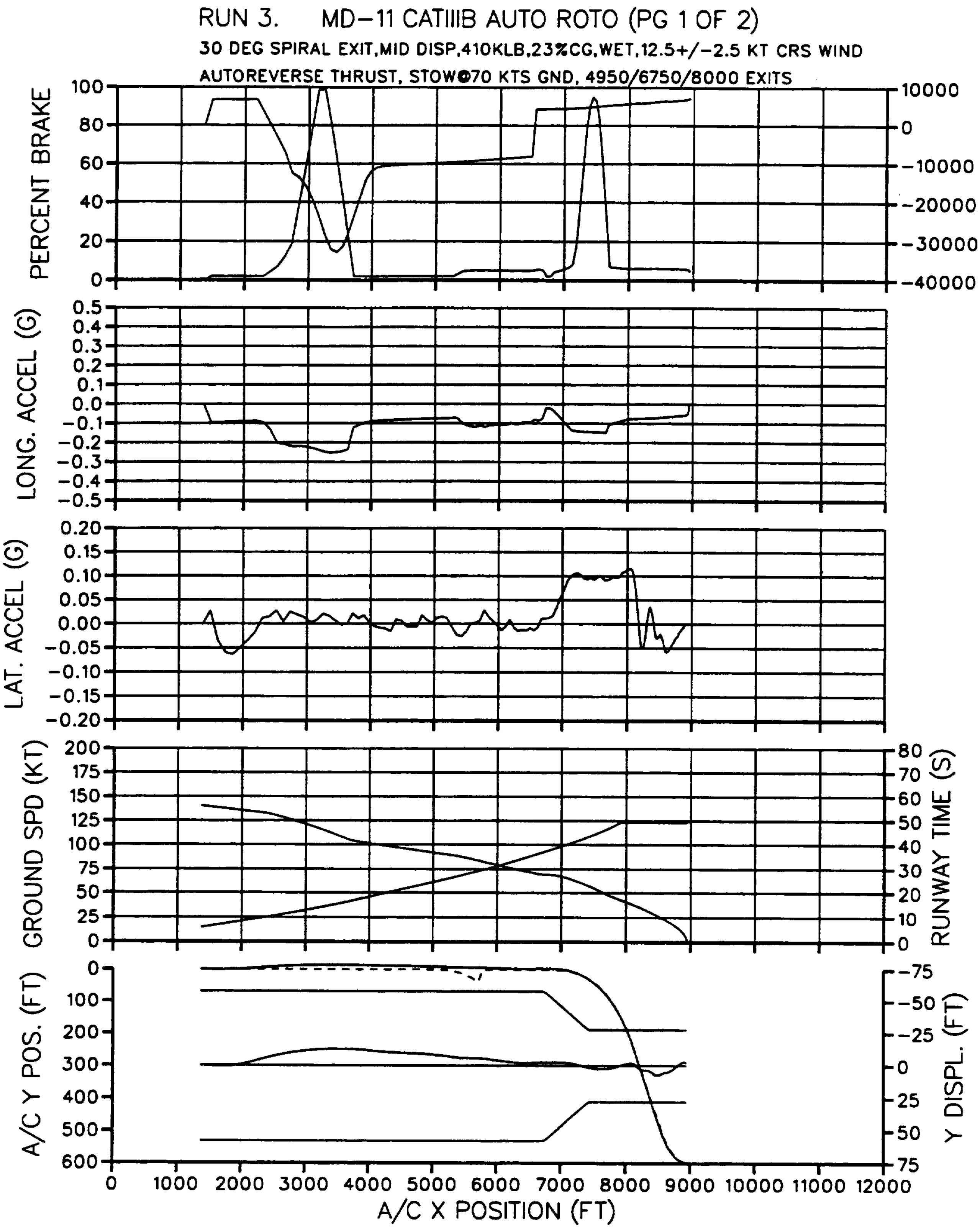
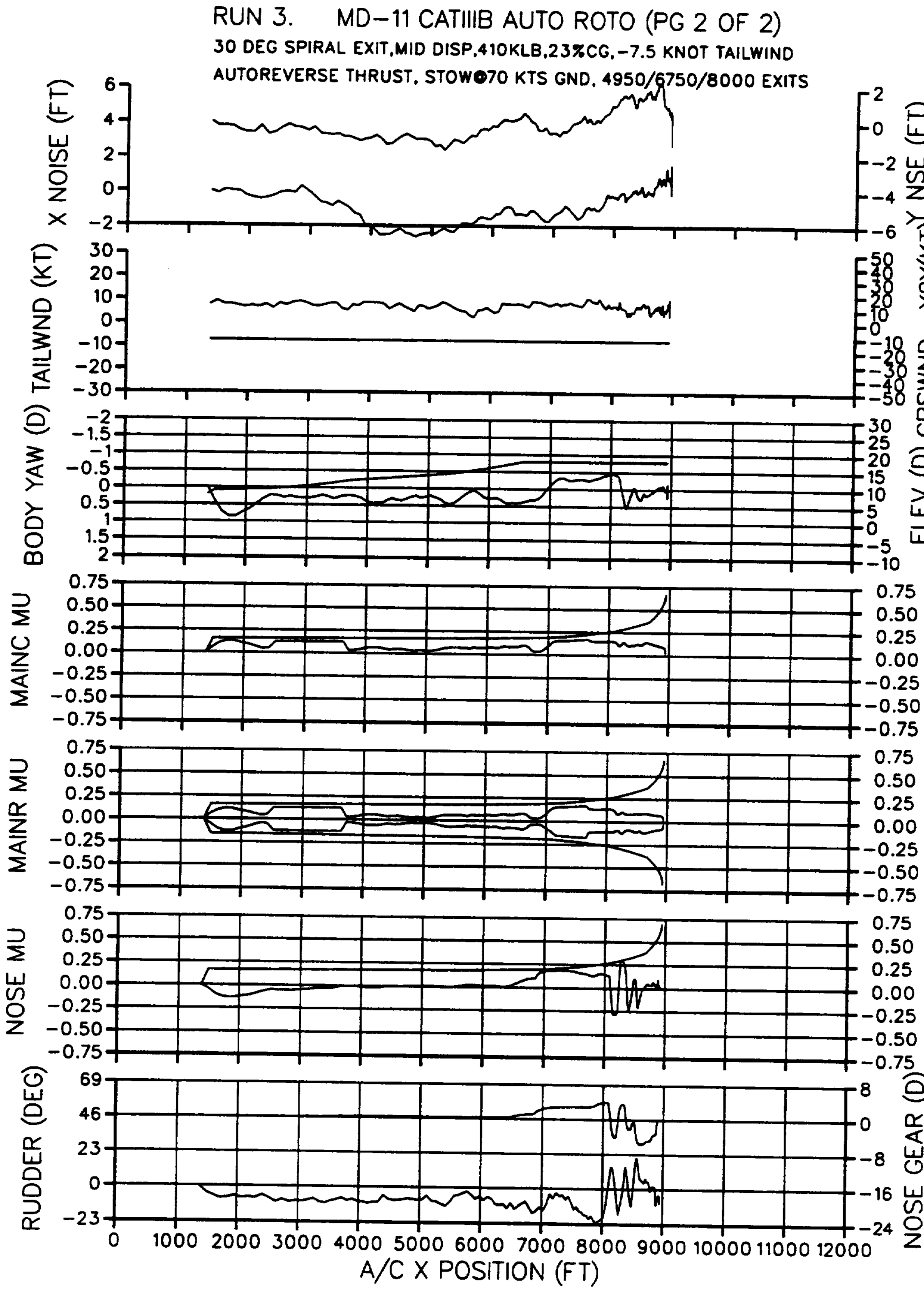

RUN 3. MD-11 CATIIIB AUTO ROTO (PG 1 OF 2)
30 DEG SPIRAL EXIT,MID DISP,410KLB,23%CG,WET,12.5+/-2.5 KT CRS WIND
AUTOREVERSE THRUST, STOW@70 KTS GND, 4950/6750/8000 EXITS
PERCENT BRAKE
THRUST (LBS)
LONG. ACCEL (G)
LAT. ACCEL (G)
GROUND SPD (KT)
RUNWAY TIME (S)
A/C Y POS. (FT)
Y DISPL. (FT)
A/C X POSITION (FT)

Figure 8.5



RUN 3. MD-11 CATIIIB AUTO ROTO (PG 2 OF 2)
30 DEG SPIRAL EXIT,MID DISP,410KLB,23%CG,-7.5 KNOT TAILWIND
AUTOREVERSE THRUST, STOW@70 KTS GND, 4950/6750/8000 EXITS
X NOISE (FT)
Y NSE (FT)
TAILWND (KT)
CRSWND -Y2Y(KT)
BODY YAW (D)
ELEV (D)
MAINC MU
MAX C MU
MAINR MU
-MAINL MU
NOSE MU
MAX N MU
RUDDER (DEG)
NOSE GEAR (D)
0 1000 2000 3000 4000 5000 6000 7000 8000 9000 10000 11000 12000
A/C X POSITION (FT)

Figure 8.6

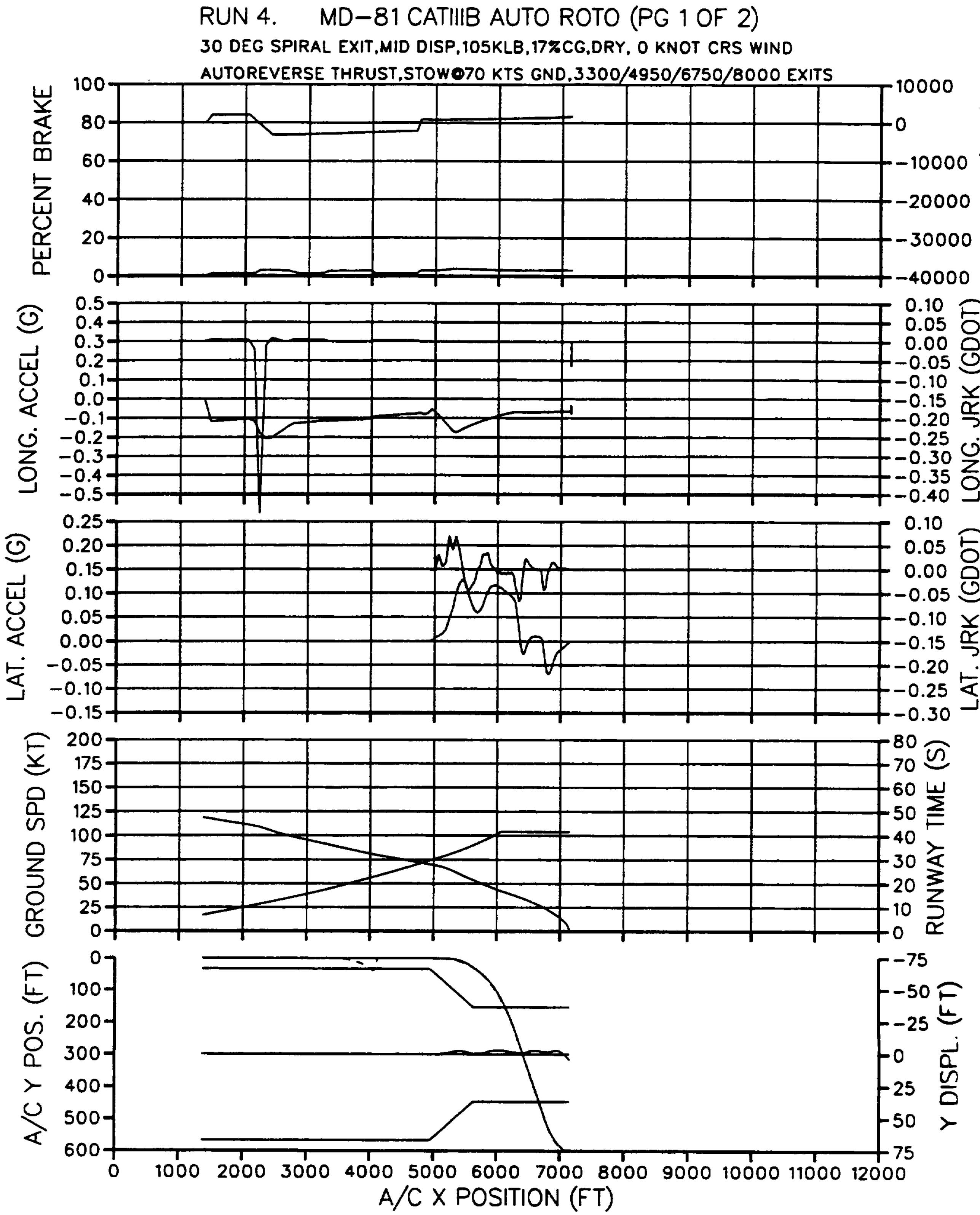

RUN 4. MD-81 CATIIIB AUTO ROTO (PG 1 OF 2)
30 DEG SPIRAL EXIT,MID DISP,105KLB,17%CG,DRY, 0 KNOT CRS WIND
AUTOREVERSE THRUST,STOW@70 KTS GND,3300/4950/6750/8000 EXITS
PERCENT BRAKE
THRUST (LBS)
LONG. ACCEL (G)
LONG. JRK (GDOT)
LAT. ACCEL (G)
LAT. JRK (GDOT)
GROUND SPD (KT)
RUNWAY TIME (S)
A/C Y POS. (FT)
Y DISPL. (FT)
A/C X POSITION (FT)

Figure 8.7

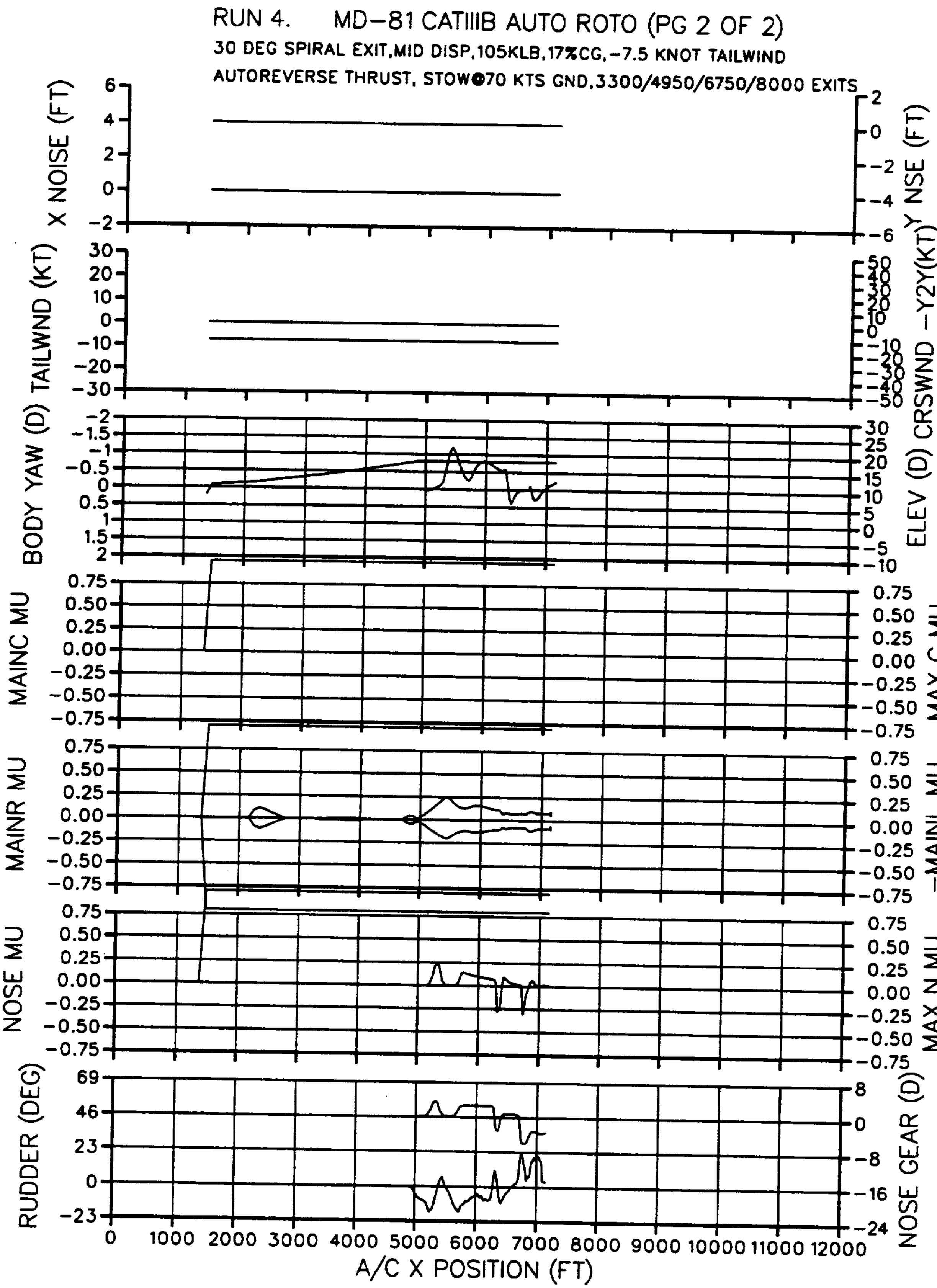

RUN 4. MD-81 CATIIIB AUTO ROTO (PG 2 OF 2)
30 DEG SPIRAL EXIT,MID DISP,105KLB,17%CG,-7.5 KNOT TAILWIND
AUTOREVERSE THRUST, STOW@70 KTS GND,3300/4950/6750/8000 EXITS
X NOISE (FT)
Y NSE (FT)
TAILWND (KT)
CRSWND -Y2Y(KT)
BODY YAW (D)
ELEV (D)
MAINC MU
MAX C MU
MAINR MU
-MAINL MU
NOSE MU
MAX N MU
RUDDER (DEG)
NOSE GEAR (D)
A/C X POSITION (FT)

Figure 8.8

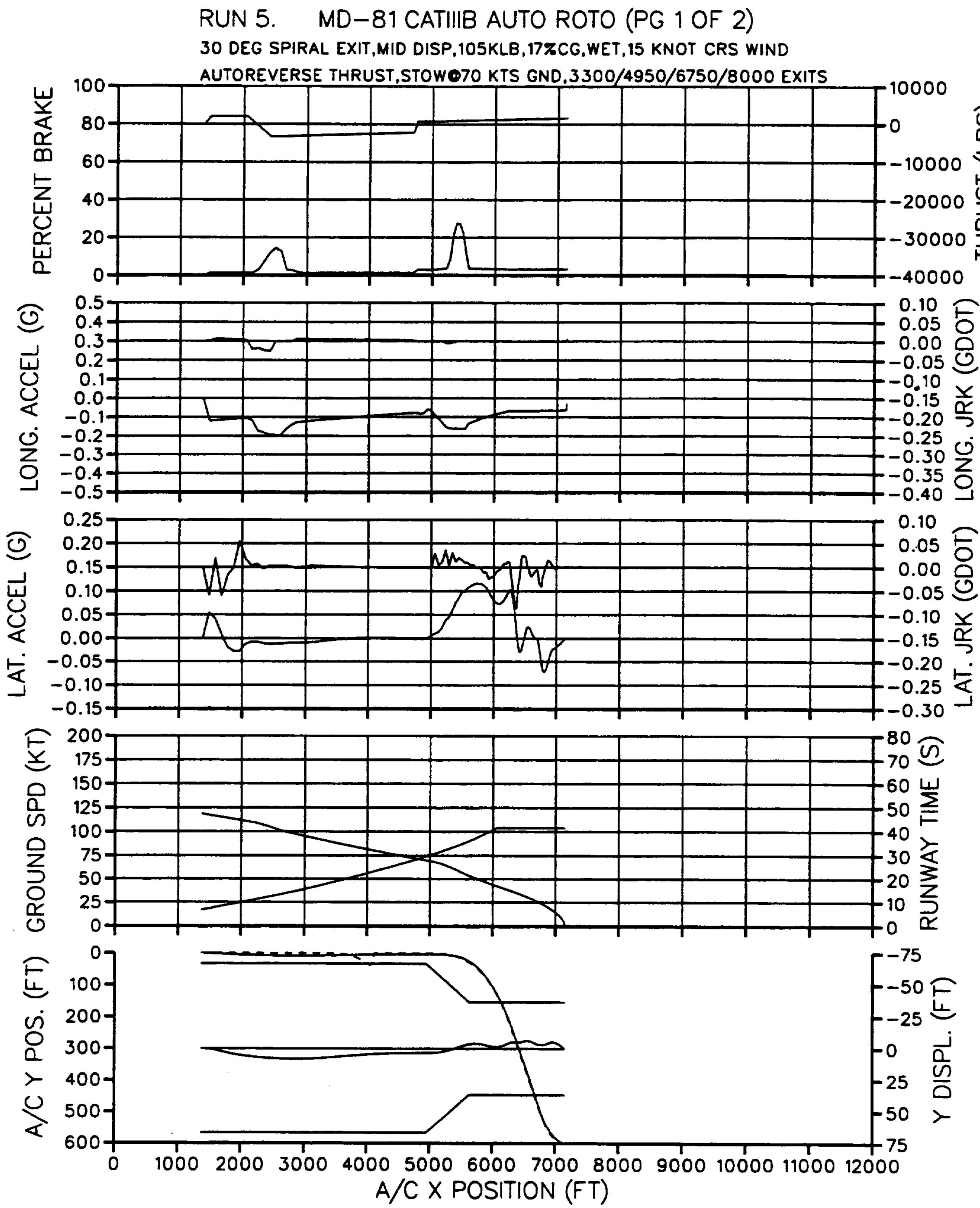

RUN 5. MD-81 CATIIIB AUTO ROTO (PG 1 OF 2)
30 DEG SPIRAL EXIT,MID DISP,105KLB,17%CG,WET,15 KNOT CRS WIND
AUTOREVERSE THRUST,STOW@70 KTS GND,3300/4950/6750/8000 EXITS
PERCENT BRAKE
THRUST (LBS)
LONG. ACCEL (G)
LONG. JRK (GDOT)
LAT. ACCEL (G)
LAT. JRK (GDOT)
GROUND SPD (KT)
RUNWAY TIME (S)
A/C Y POS. (FT)
Y DISPL. (FT)
A/C X POSITION (FT)

Figure 8.9

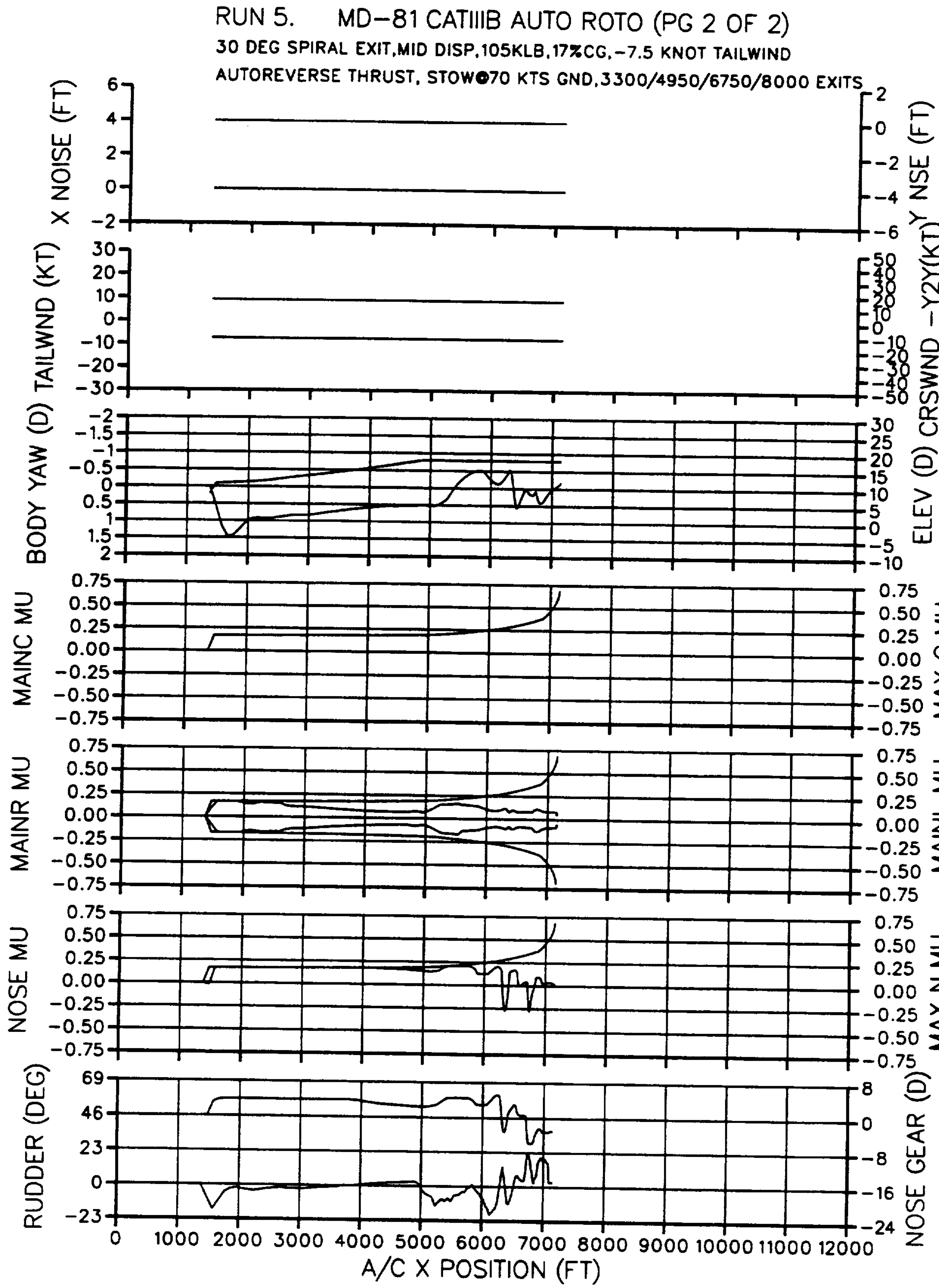

RUN 5. MD-81 CATIIIB AUTO ROTO (PG 2 OF 2)
30 DEG SPIRAL EXIT,MID DISP,105KLB,17%CG,-7.5 KNOT TAILWIND
AUTOREVERSE THRUST, STOW@70 KTS GND,3300/4950/6750/8000 EXITS
X NOISE (FT)
Y NSE (FT)
TAILWND (KT)
CRSWND -Y2Y(KT)
BODY YAW (D)
ELEV (D)
MAINC MU
MAX C MU
MAINR MU
-MAINL MU
NOSE MU
MAX N MU
RUDDER (DEG)
NOSE GEAR (D)
0 1000 2000 3000 4000 5000 6000 7000 8000 9000 10000 11000 12000
A/C X POSITION (FT)

Figure 8.10

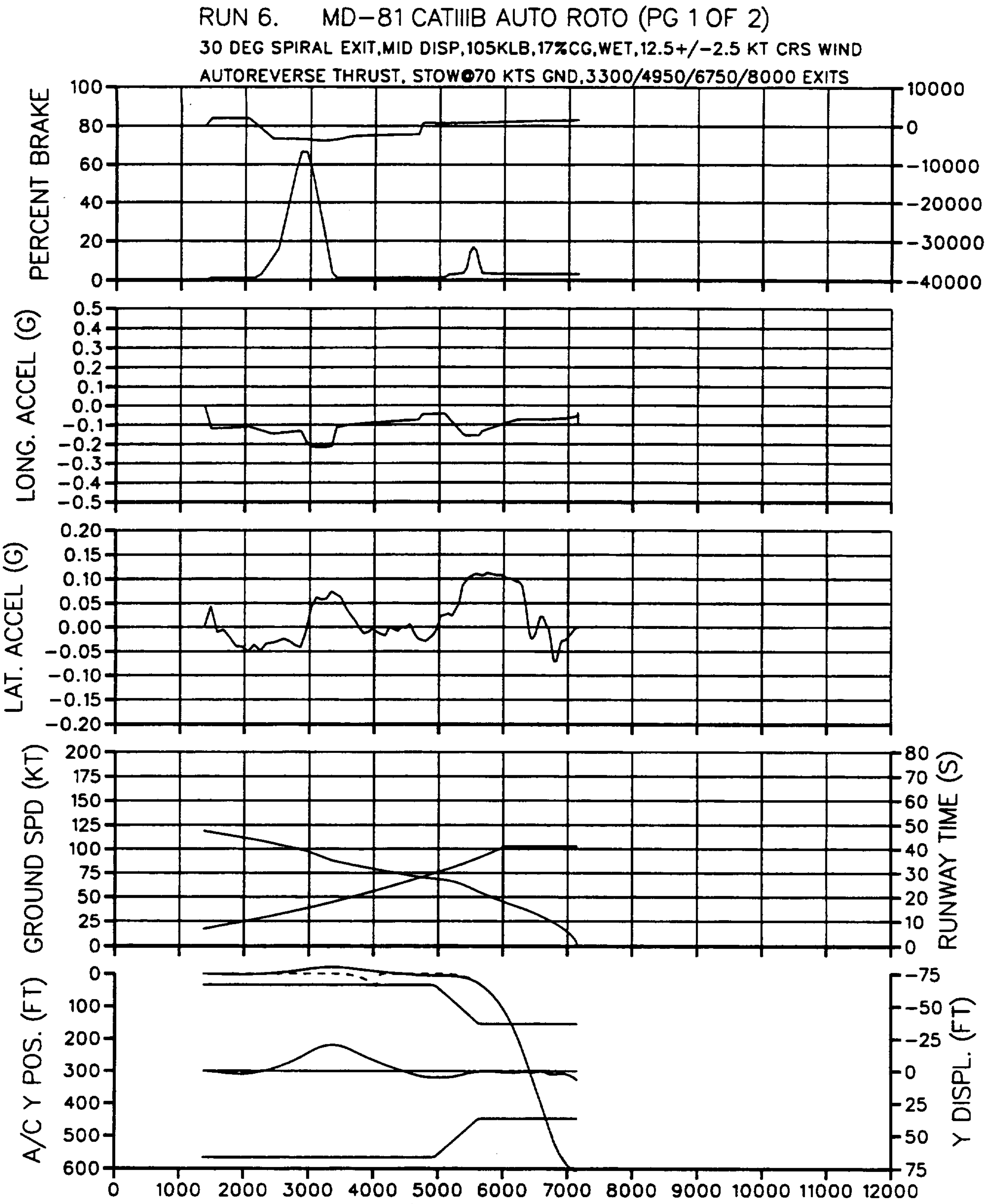
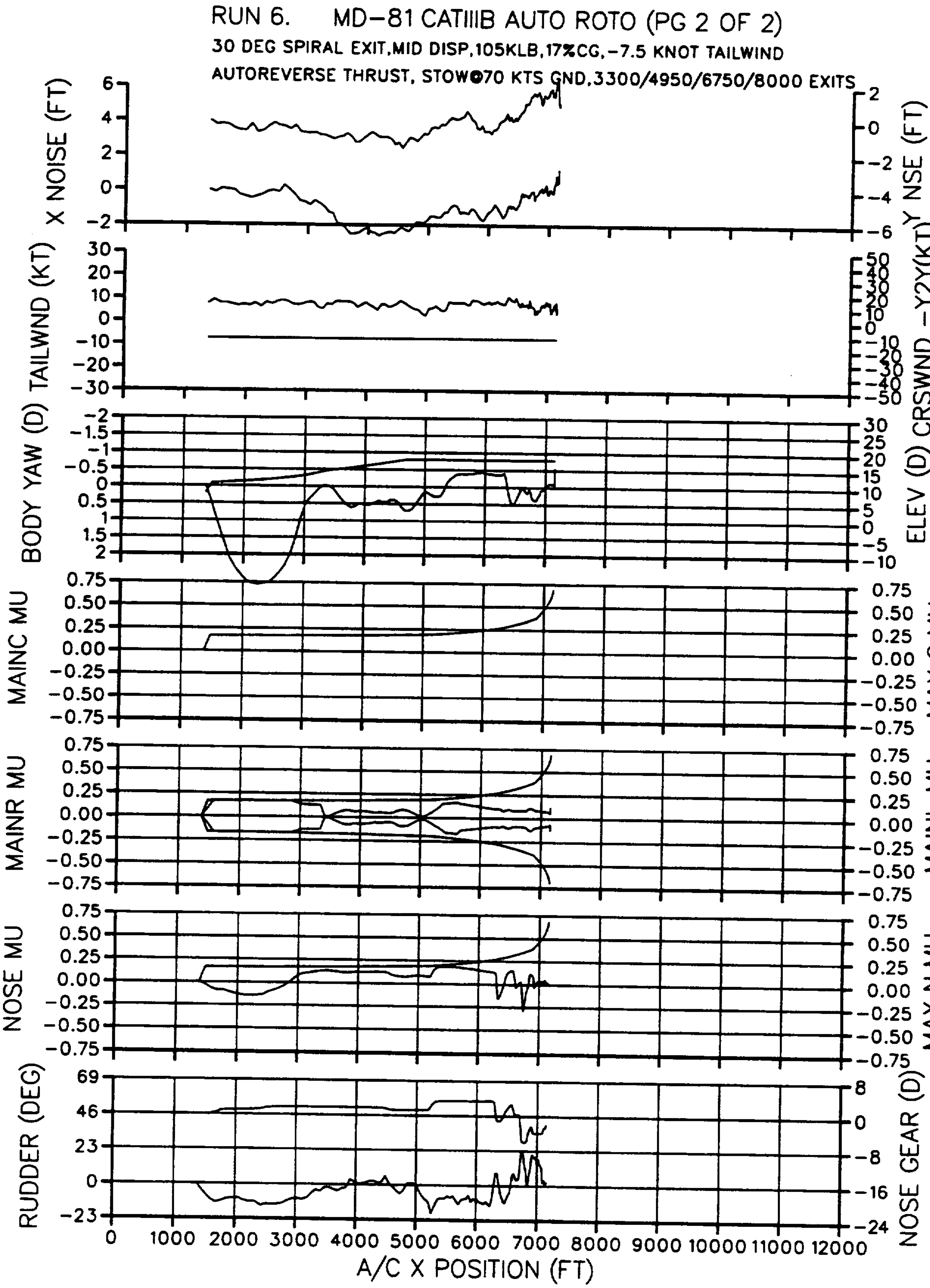

RUN 6. MD-81 CATIIIB AUTO ROTO (PG 1 OF 2)
30 DEG SPIRAL EXIT,MID DISP,105KLB,17%CG,WET,12.5+/-2.5 KT CRS WIND
AUTOREVERSE THRUST, STOW@70 KTS GND,3300/4950/6750/8000 EXITS
PERCENT BRAKE
THRUST (LBS)
LONG. ACCEL (G)
LAT. ACCEL (G)
GROUND SPD (KT)
RUNWAY TIME (S)
A/C Y POS. (FT)
Y DISPL. (FT)
A/C X POSITION (FT)

Figure 8.11



RUN 6. MD-81 CATIIIB AUTO ROTO (PG 2 OF 2)
30 DEG SPIRAL EXIT,MID DISP,105KLB,17%CG,-7.5 KNOT TAILWIND
AUTOREVERSE THRUST, STOW@70 KTS GND,3300/4950/6750/8000 EXITS
X NOISE (FT)
Y NSE (FT)
TAILWND (KT)
CRSWND -Y2Y(KT)
BODY YAW (D)
ELEV (D)
MAINC MU
MAX C MU
MAINR MU
-MAINL MU
NOSE MU
MAX N MU
RUDDER (DEG)
NOSE GEAR (D)
0 1000 2000 3000 4000 5000 6000 7000 8000 9000 10000 11000 12000
A/C X POSITION (FT)

Figure 8.12

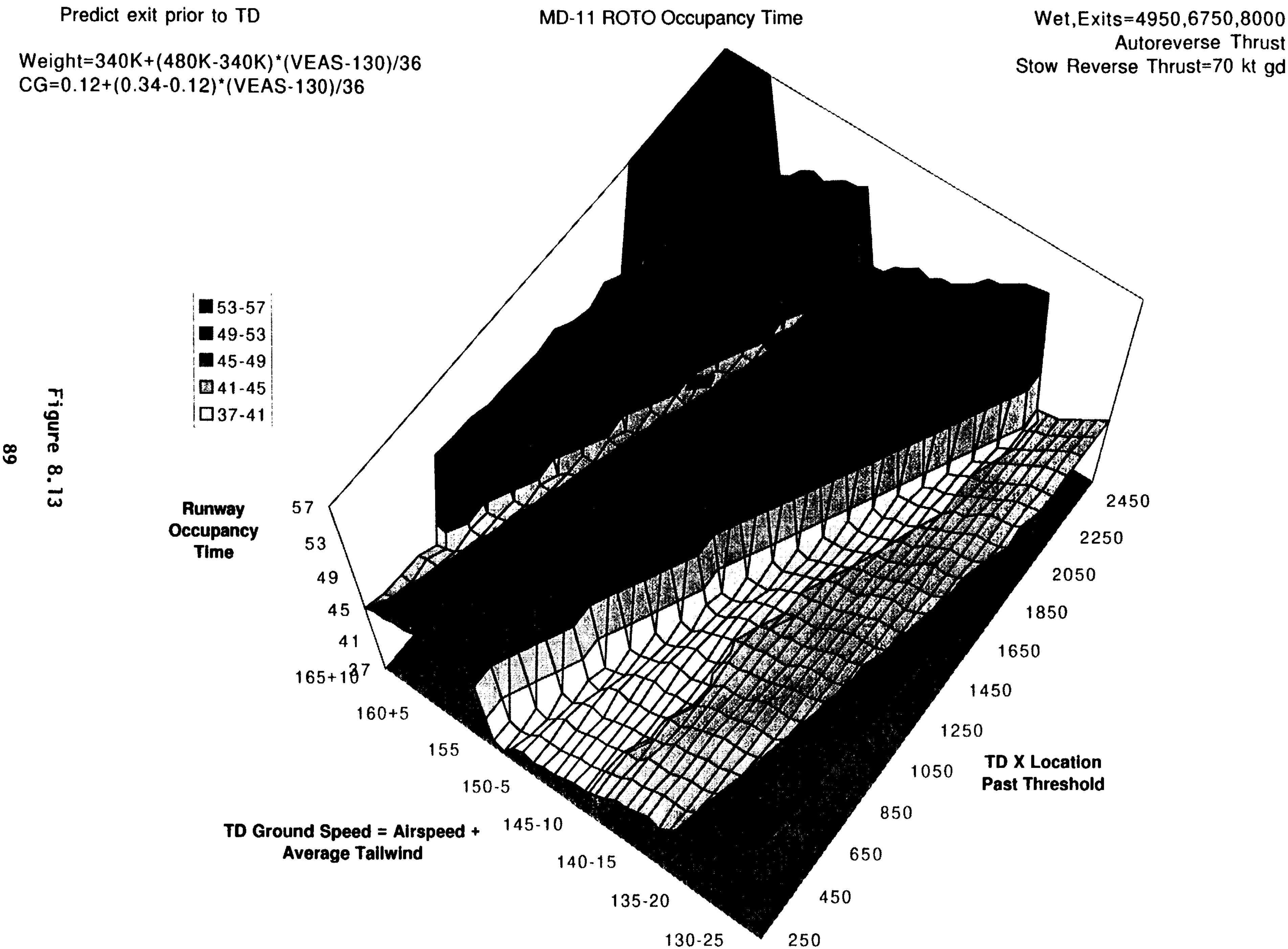

Predict exit prior to TD
MD-11 ROTO Occupancy Time
Wet,Exits=4950,6750,8000
Autoreverse Thrust
Stow Reverse Thrust=70 kt gd
Weight=340K+(480K-340K)*(VEAS-130)/36
CG=0.12+(0.34-0.12)*(VEAS-130)/36
53-57
49-53
45-49
41-45
37-41
Runway Occupancy Time
57
53
49
45
41
37
165+10
160+5
155
150-5
145-10
140-15
135-20
130-25
TD Ground Speed = Airspeed + Average Tailwind
250
450
650
850
1050
1250
1450
1650
1850
2050
2250
2450
TD X Location Past Threshold

Figure 8.13

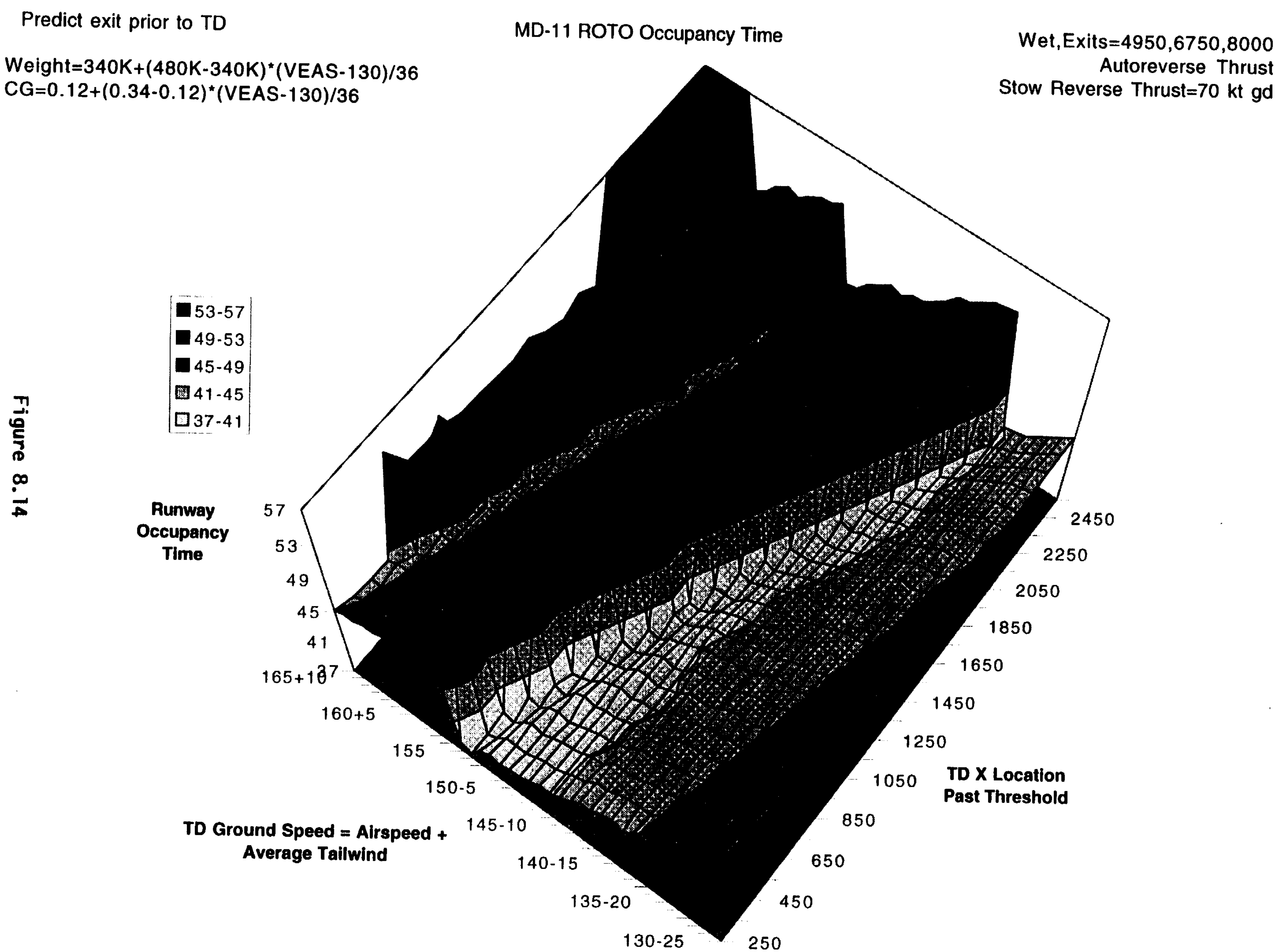

Predict exit prior to TD
Weight=340K+(480K-340K)*(VEAS-130)/36
CG=0.12+(0.34-0.12)*(VEAS-130)/36
MD-11 ROTO Occupancy Time
Wet,Exits=4950,6750,8000
Autoreverse Thrust
Stow Reverse Thrust=70 kt gd
53-57
49-53
45-49
41-45
37-41
Runway Occupancy Time
57
53
49
45
41
37
165+10
160+5
155
150-5
145-10
140-15
135-20
130-25
TD Ground Speed = Airspeed + Average Tailwind
250
450
650
850
1050
1250
1450
1650
1850
2050
2250
2450
TD X Location Past Threshold

Figure 8.14

Predict exit prior to TD

MD-11 ROTO Exit Used

Wet,Exits=4950,6750,8000
Autoreverse Thrust
Stow Reverse Thrust=70 kt gd

Weight=340K+(480K-340K)*(VEAS-130)/36
CG=0.12+(0.34-0.12)*(VEAS-130)/36

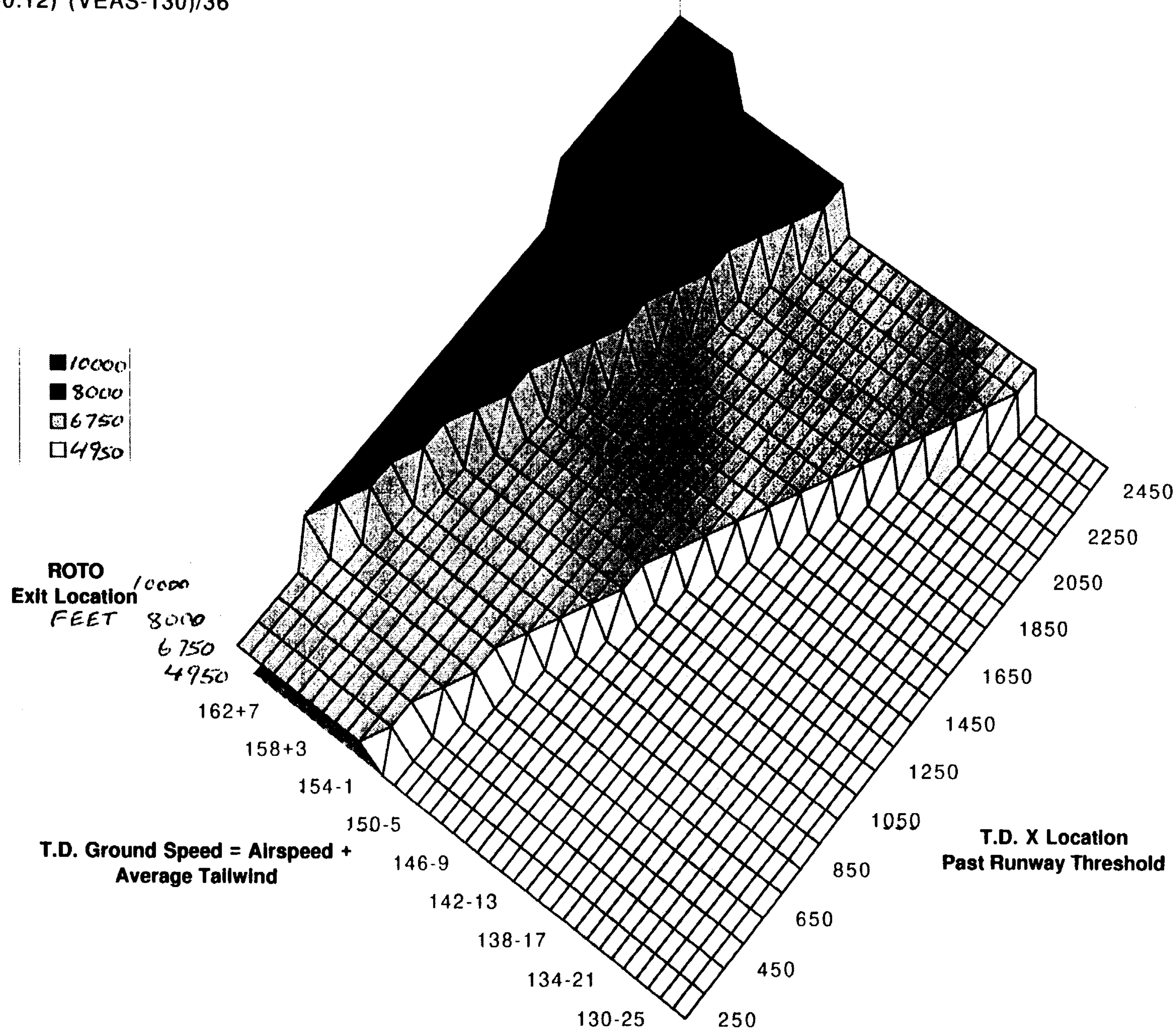


Figure 8.15

MD-11 Joint Probability Density Function for 2 T.D. inputs (assume independent, gaussian distributed)

Weight=340K+(480K-340K)*(VEAS-130)/36
CG=0.12+(0.34-0.12)*(VEAS-130)/36

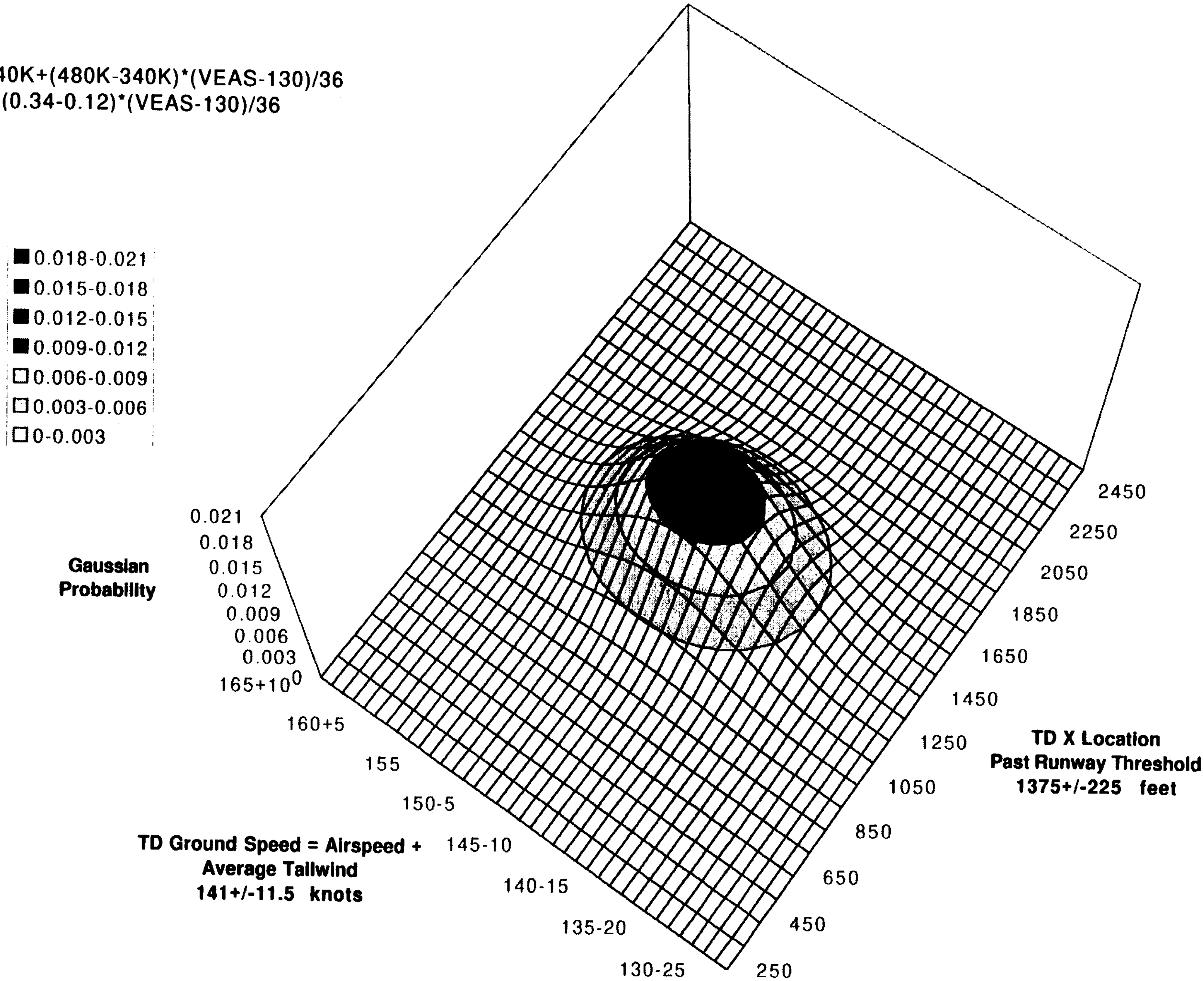


Figure 8.16

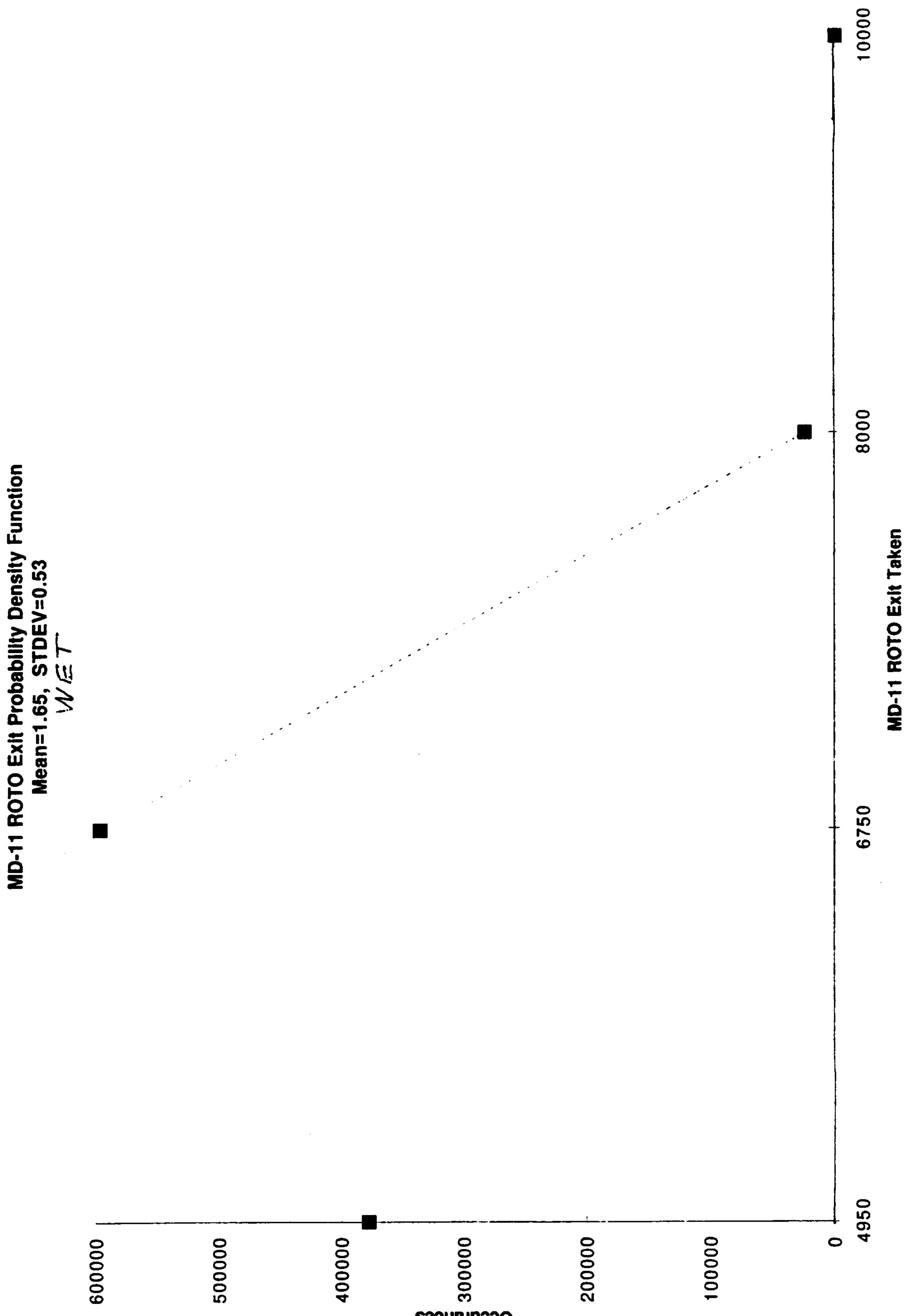

MD-11 ROTO Exit Probability Density Function
Mean=1.65, STDEV=0.53
WET
Occurances
600000
500000
400000
300000
200000
100000
0
4950
6750
8000
10000
MD-11 ROTO Exit Taken

Figure 8.17

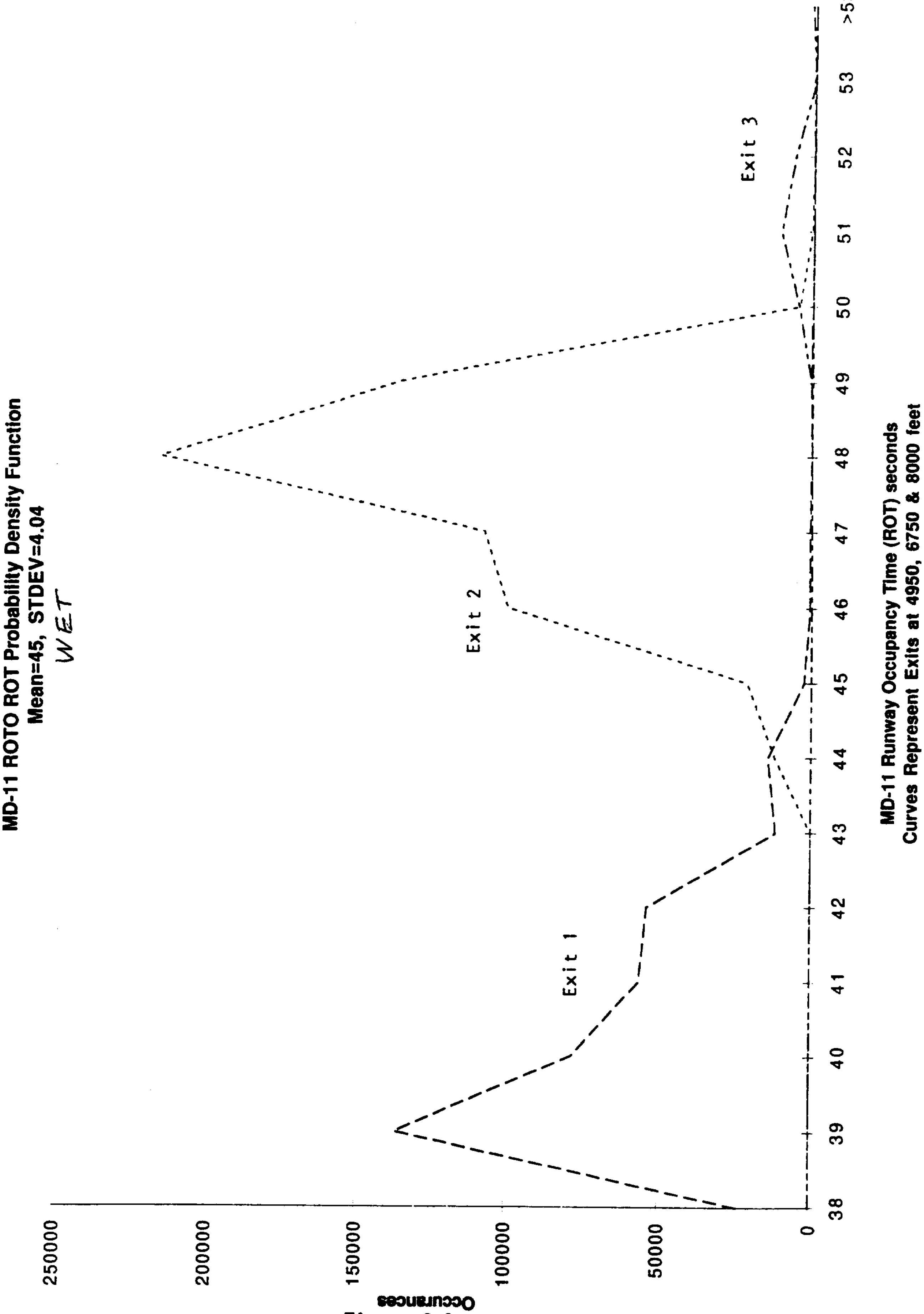

MD-11 ROTO ROT Probability Density Function
Mean=45, STDEV=4.04
WET
Exit 1
Exit 2
Exit 3
250000
200000
150000
100000
50000
0
Occurances
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
>53
MD-11 Runway Occupancy Time (ROT) seconds
Curves Represent Exits at 4950, 6750 & 8000 feet

Figure 8.18

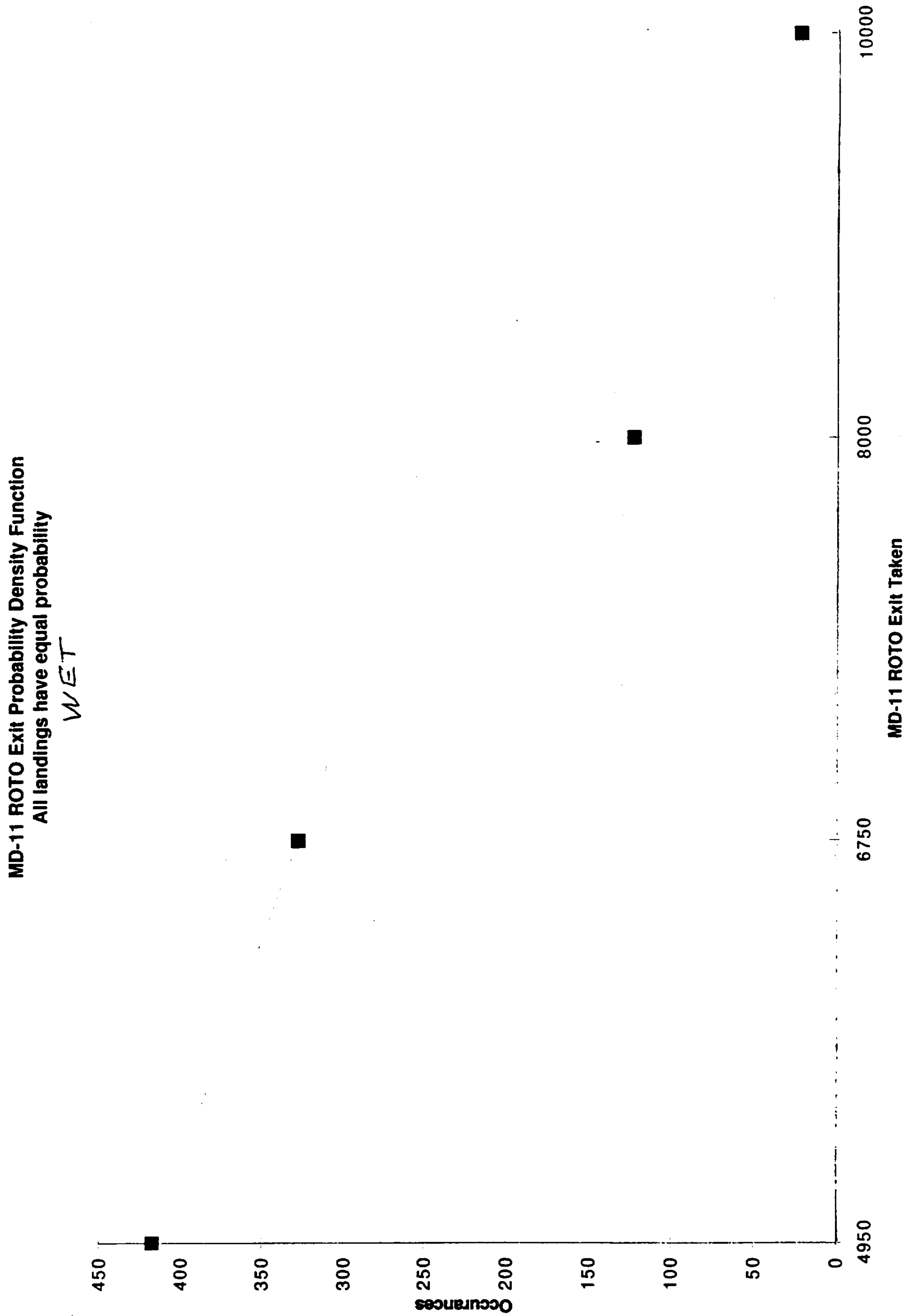

MD-11 ROTO Exit Probability Density Function
All landings have equal probability
WET
Occurances
450
400
350
300
250
200
150
100
50
0
4950
6750
8000
10000
MD-11 ROTO Exit Taken

Figure 8.19

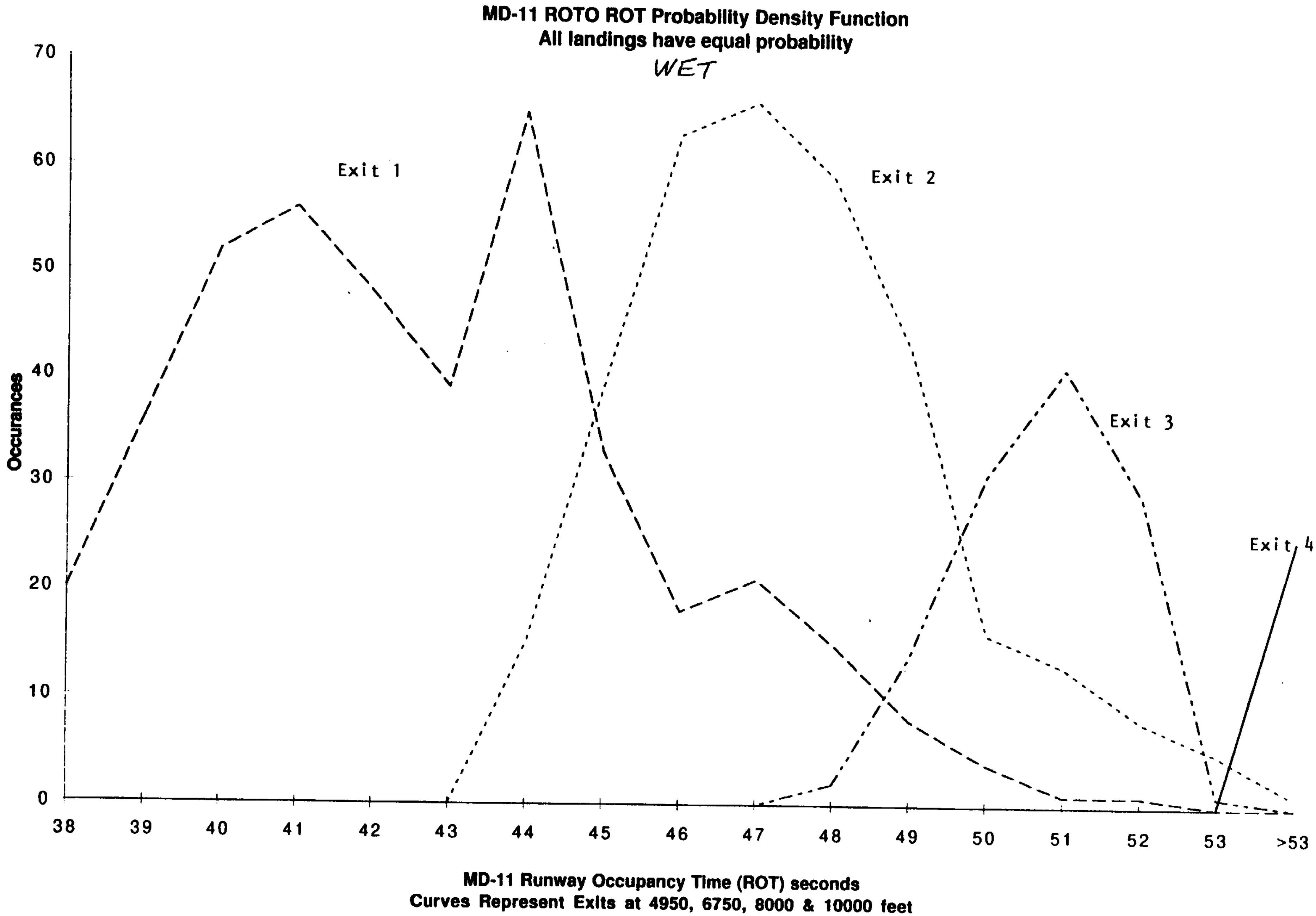

MD-11 ROTO ROT Probability Density Function
All landings have equal probability
WET
Occurances
70
60
50
40
30
20
10
0
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
>53
Exit 1
Exit 2
Exit 3
Exit 4
MD-11 Runway Occupancy Time (ROT) seconds
Curves Represent Exits at 4950, 6750, 8000 & 10000 feet

Figure 8.20

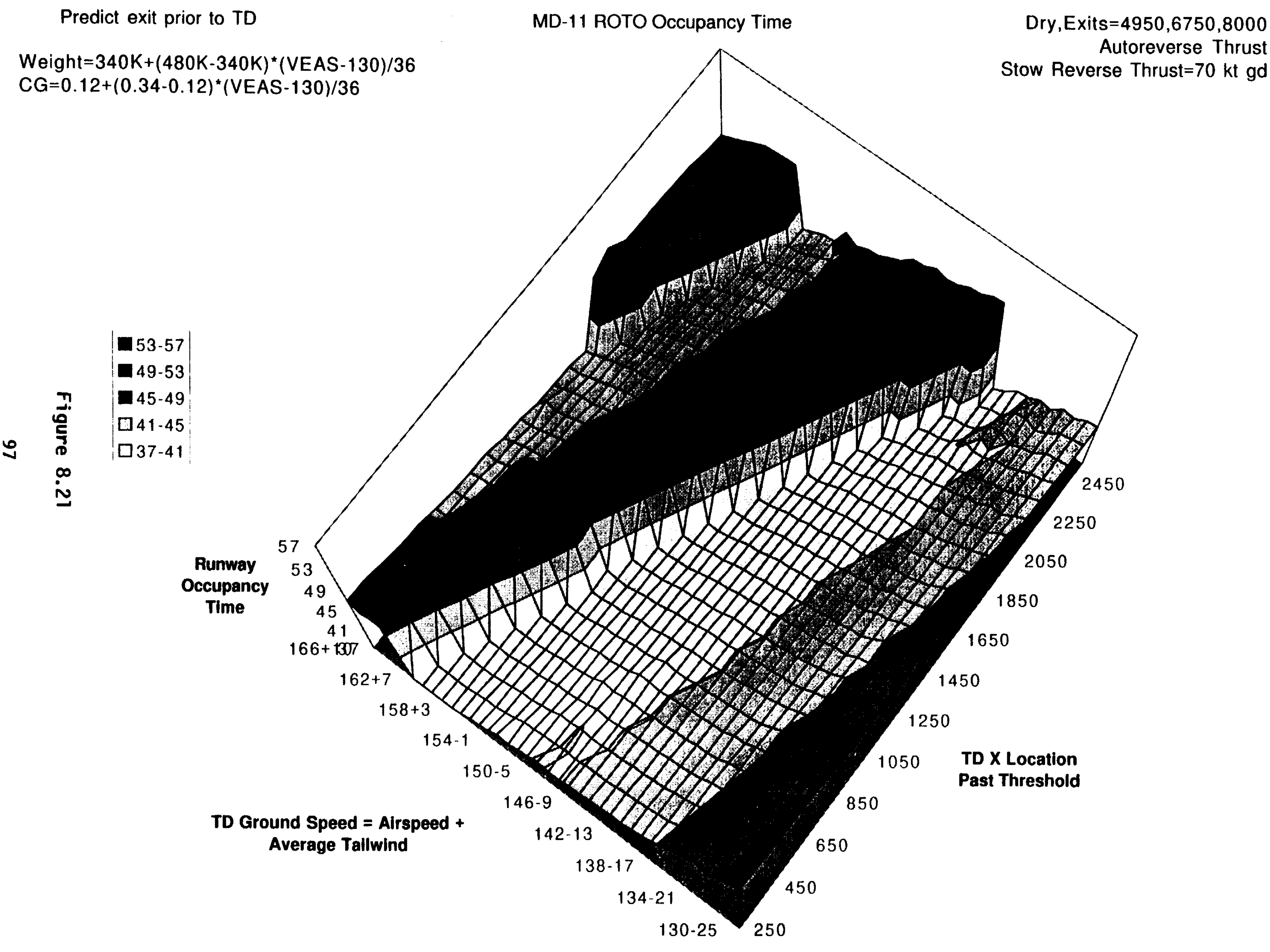

Predict exit prior to TD
MD-11 ROTO Occupancy Time
Dry,Exits=4950,6750,8000
Autoreverse Thrust
Stow Reverse Thrust=70 kt gd
Weight=340K+(480K-340K)*(VEAS-130)/36
CG=0.12+(0.34-0.12)*(VEAS-130)/36
53-57
49-53
45-49
41-45
37-41
Runway Occupancy Time
57
53
49
45
41
37
166+11
162+7
158+3
154-1
150-5
146-9
142-13
138-17
134-21
130-25
TD Ground Speed = Airspeed + Average Tailwind
250
450
650
850
1050
1250
1450
1650
1850
2050
2250
2450
TD X Location Past Threshold

Figure 8.21

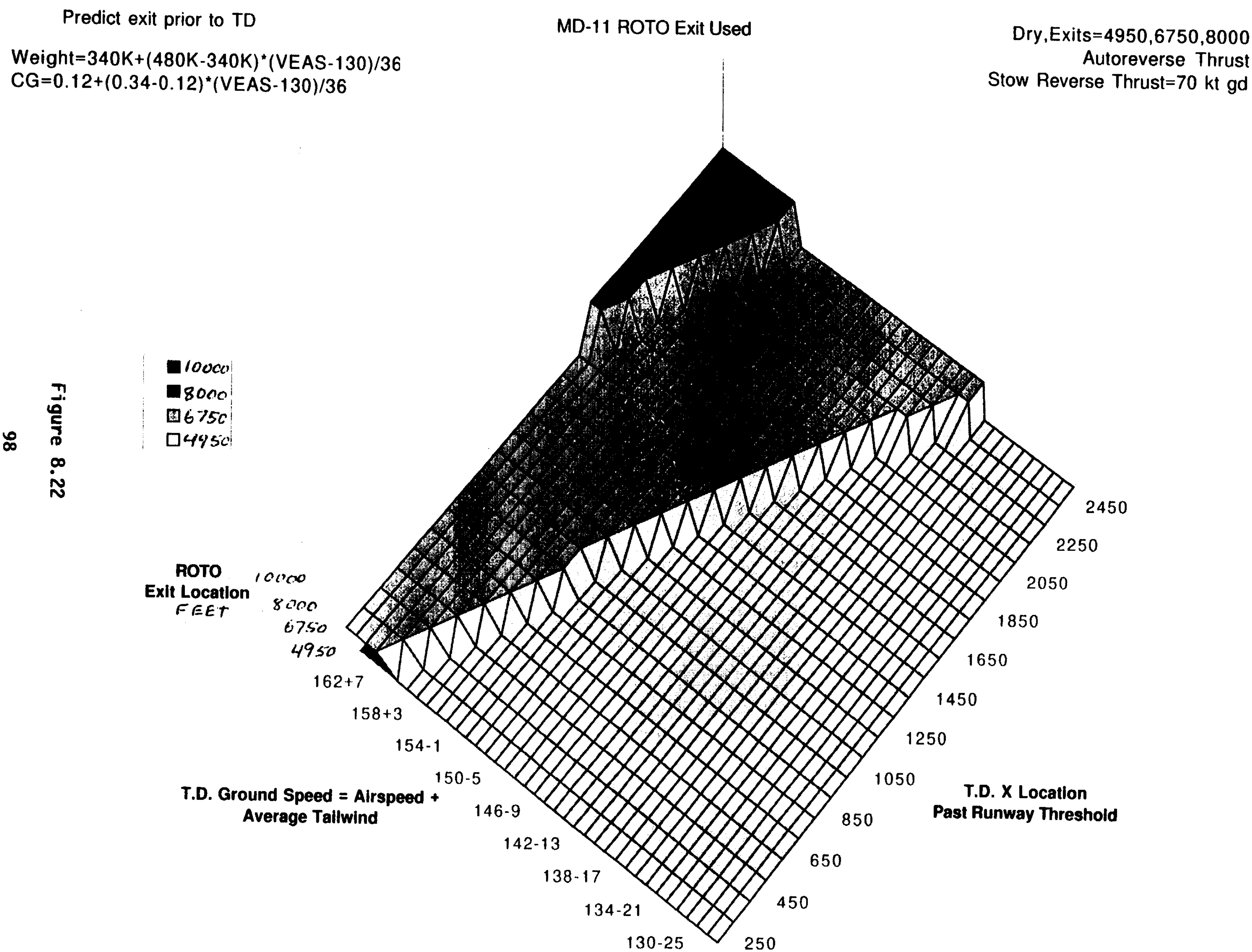

Predict exit prior to TD
Weight=340K+(480K-340K)*(VEAS-130)/36
CG=0.12+(0.34-0.12)*(VEAS-130)/36
MD-11 ROTO Exit Used
Dry,Exits=4950,6750,8000
Autoreverse Thrust
Stow Reverse Thrust=70 kt gd
10000
8000
6750
4950
ROTO
Exit Location
FEET
10000
8000
6750
4950
162+7
158+3
154-1
150-5
146-9
142-13
138-17
134-21
130-25
T.D. Ground Speed = Airspeed + Average Tailwind
250
450
650
850
1050
1250
1450
1650
1850
2050
2250
2450
T.D. X Location Past Runway Threshold

Figure 8.22

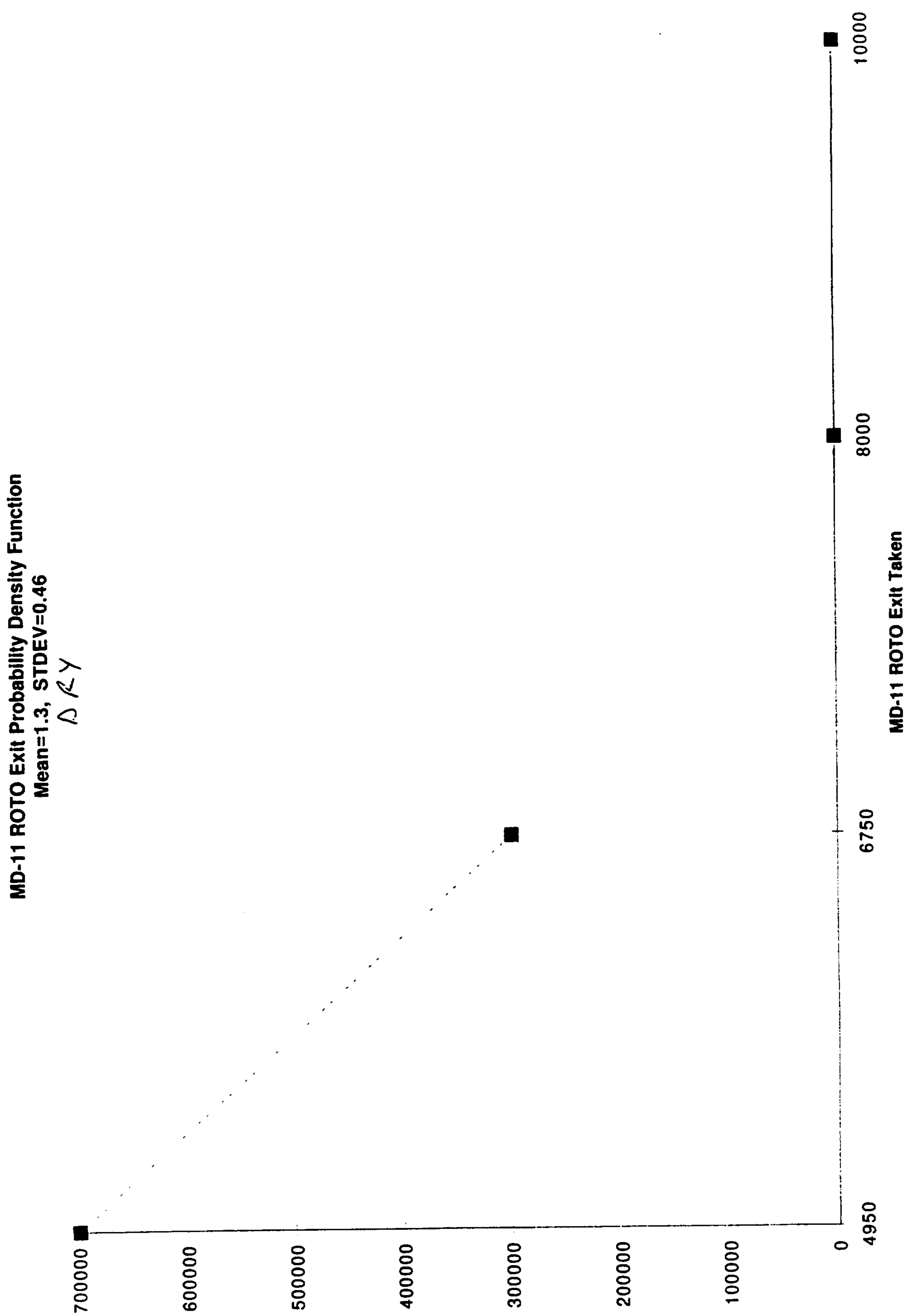
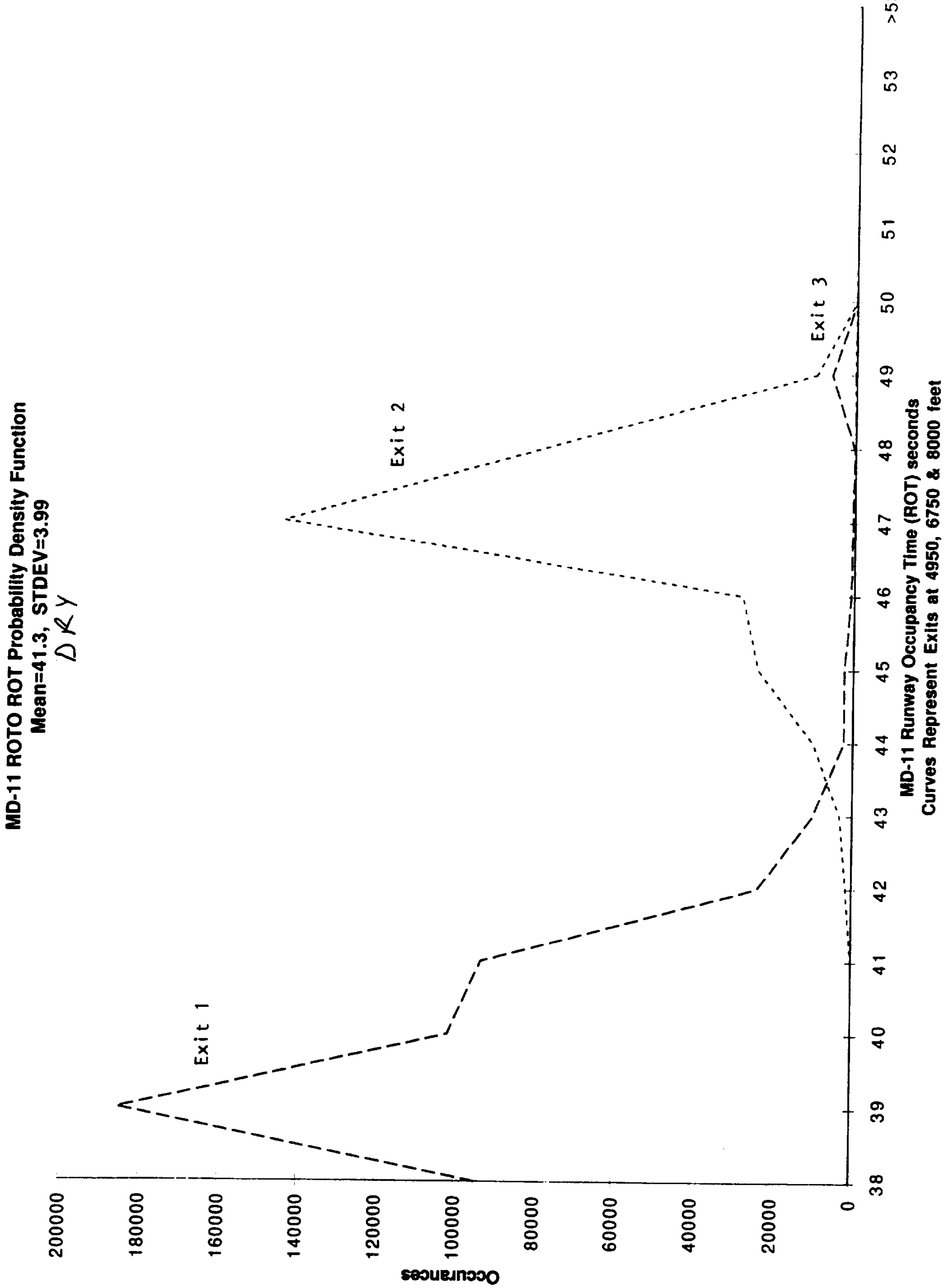

MD-11 ROTO Exit Probability Density Function
Mean=1.3, STDEV=0.46
DRY
Occurances
700000
600000
500000
400000
300000
200000
100000
0
4950
6750
8000
10000
MD-11 ROTO Exit Taken

Figure 8.23



MD-11 ROTO ROT Probability Density Function
Mean=41.3, STDEV=3.99
DRY
Exit 1
Exit 2
Exit 3
Occurances
200000
180000
160000
140000
120000
100000
80000
60000
40000
20000
0
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
>53
MD-11 Runway Occupancy Time (ROT) seconds
Curves Represent Exits at 4950, 6750 & 8000 feet

Figure 8.24

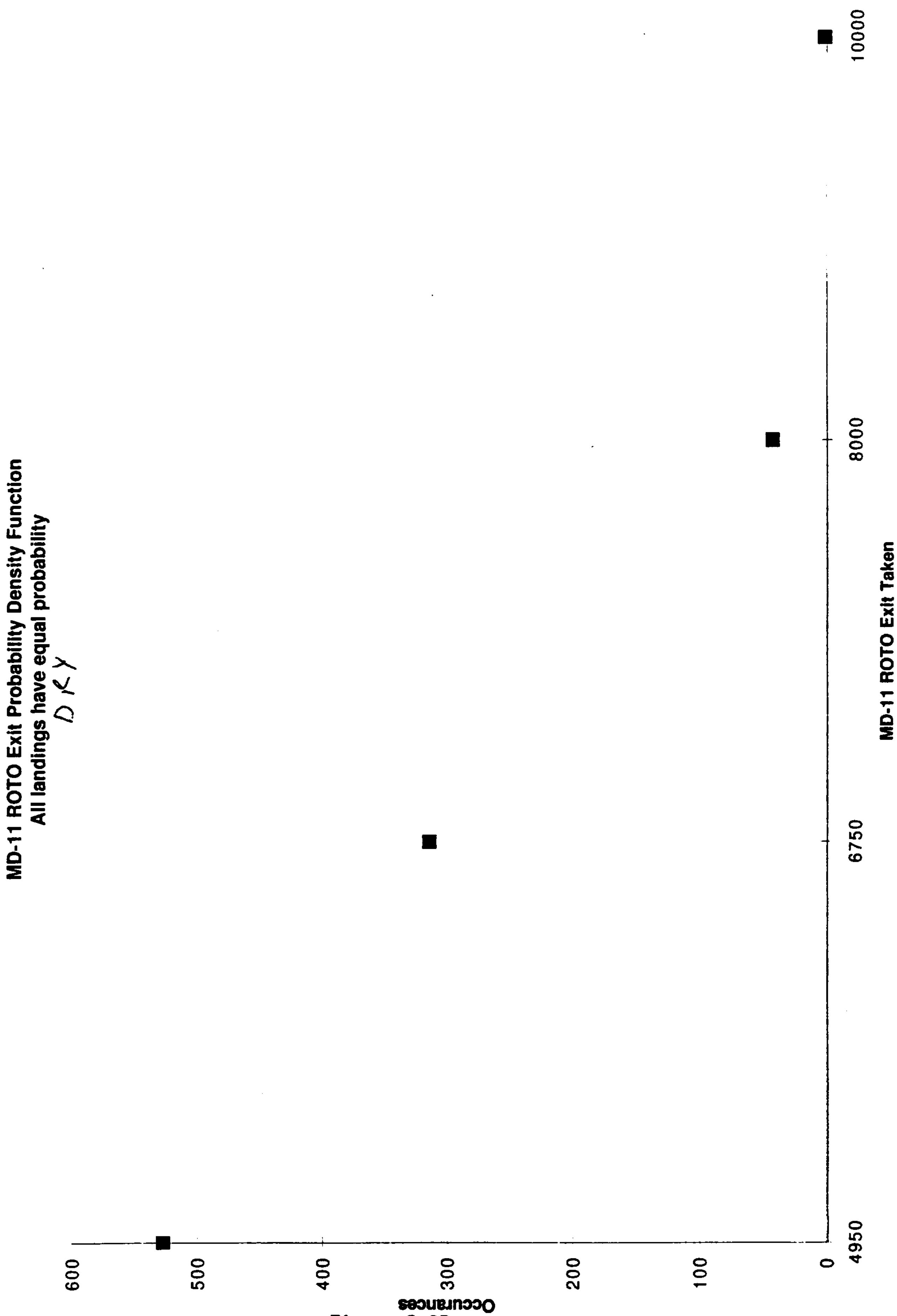

MD-11 ROTO Exit Probability Density Function
All landings have equal probability
DRY
600
500
400
300
200
100
0
Occurances
4950
6750
8000
10000
MD-11 ROTO Exit Taken

Figure 8.25

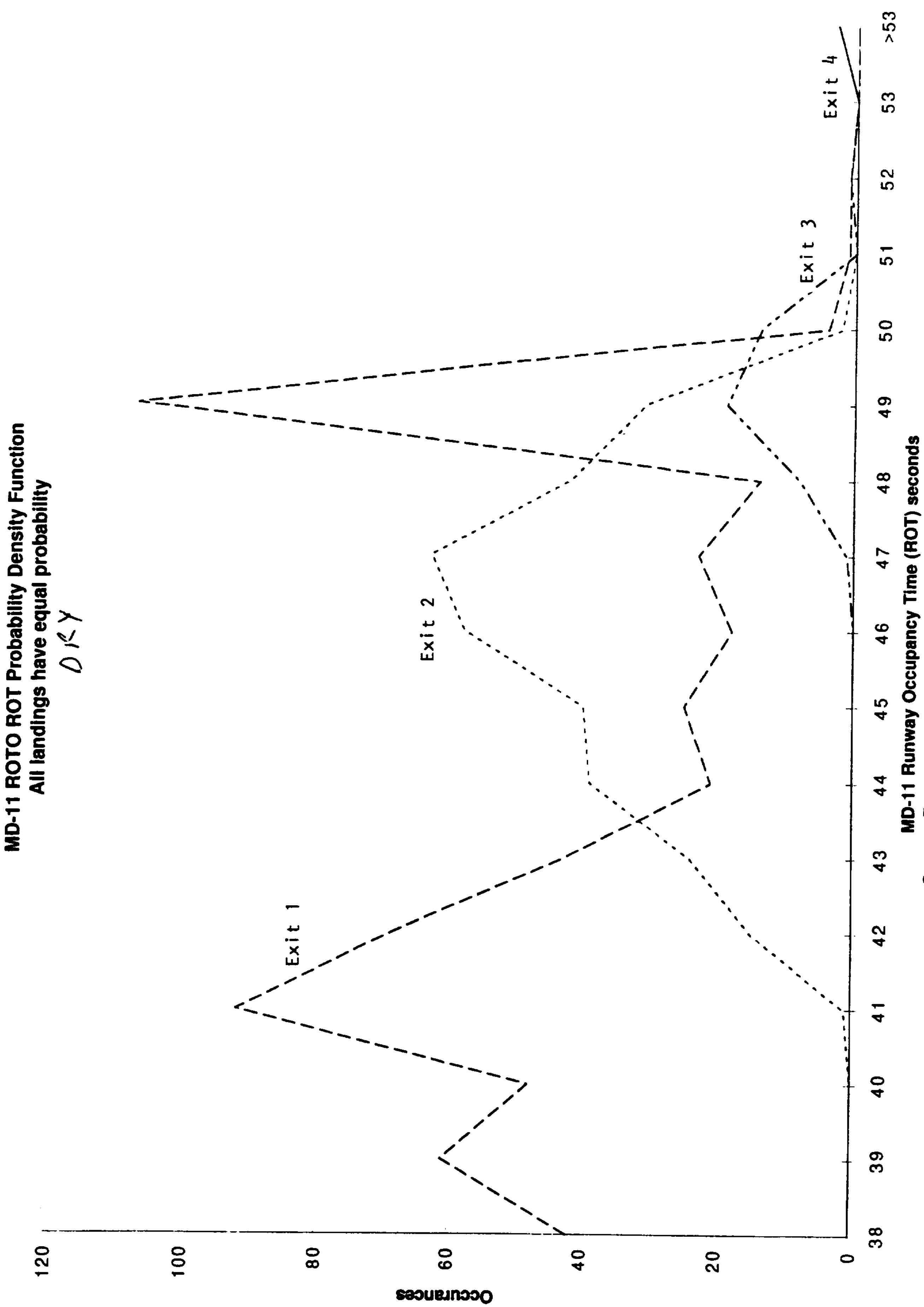

MD-11 ROTO ROT Probability Density Function
All landings have equal probability
DRY
Occurances
0
20
40
60
80
100
120
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
>53
Exit 1
Exit 2
Exit 3
Exit 4
MD-11 Runway Occupancy Time (ROT) seconds
Curves Represent Exits at 4950, 6750, 8000 & 10000 feet

Figure 8.26

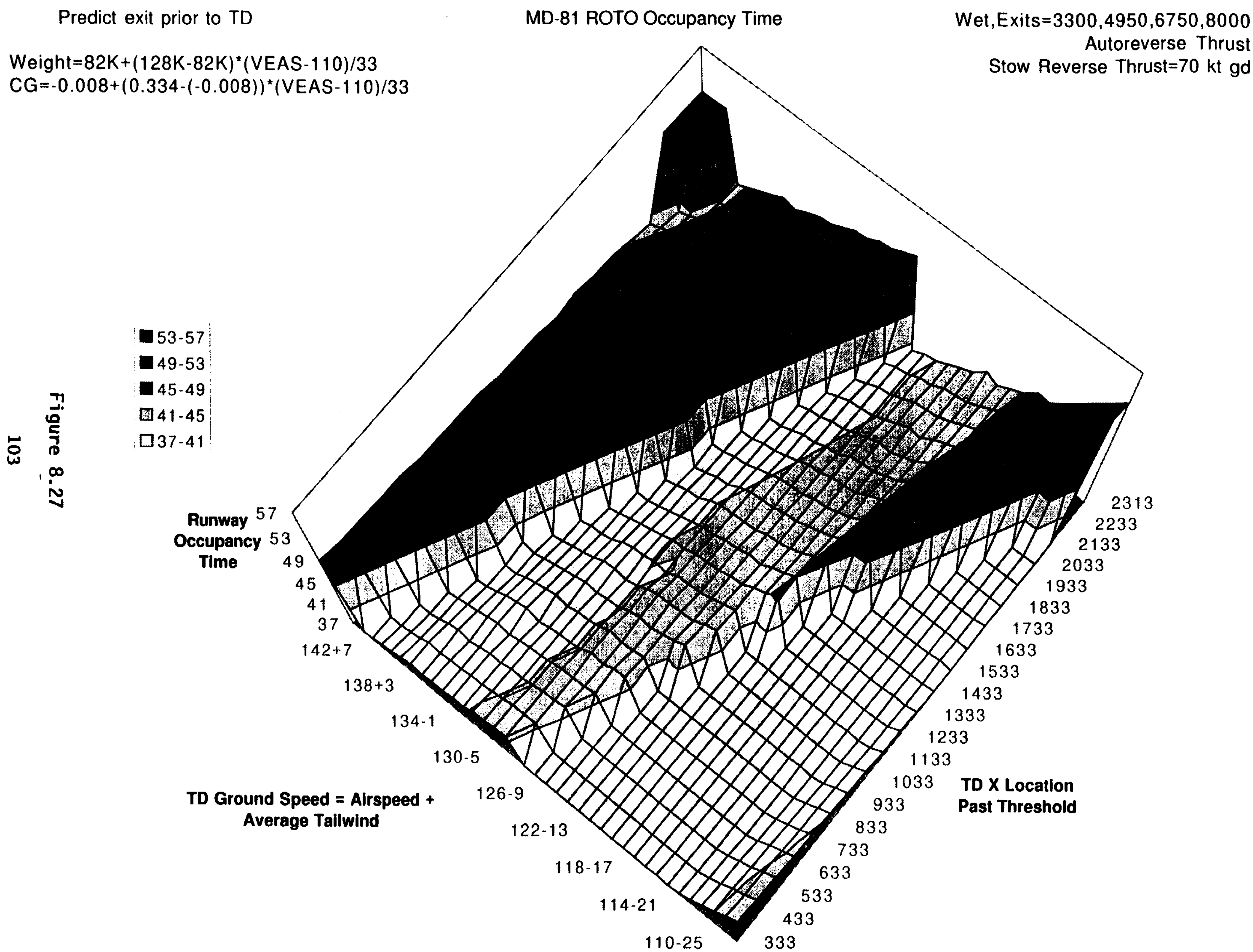

Predict exit prior to TD
MD-81 ROTO Occupancy Time
Wet,Exits=3300,4950,6750,8000
Autoreverse Thrust
Stow Reverse Thrust=70 kt gd
Weight=82K+(128K-82K)*(VEAS-110)/33
CG=-0.008+(0.334-(-0.008))*(VEAS-110)/33
53-57
49-53
45-49
41-45
37-41
Runway Occupancy Time
57
53
49
45
41
37
142+7
138+3
134-1
130-5
126-9
122-13
118-17
114-21
110-25
TD Ground Speed = Airspeed + Average Tailwind
2313
2233
2133
2033
1933
1833
1733
1633
1533
1433
1333
1233
1133
1033
933
833
733
633
533
433
333
TD X Location Past Threshold

Figure 8.27

Predict exit prior to TD

MD-81 ROTO Exit Used

Wet,Exits=3300,4950,6750,8000
Autoreverse Thrust
Stow Reverse Thrust=70 kt gd

Weight=82K+(128K-82K)*(VEAS-110)/33
CG=-0.008+(0.334-(-0.008))*(VEAS-110)/33

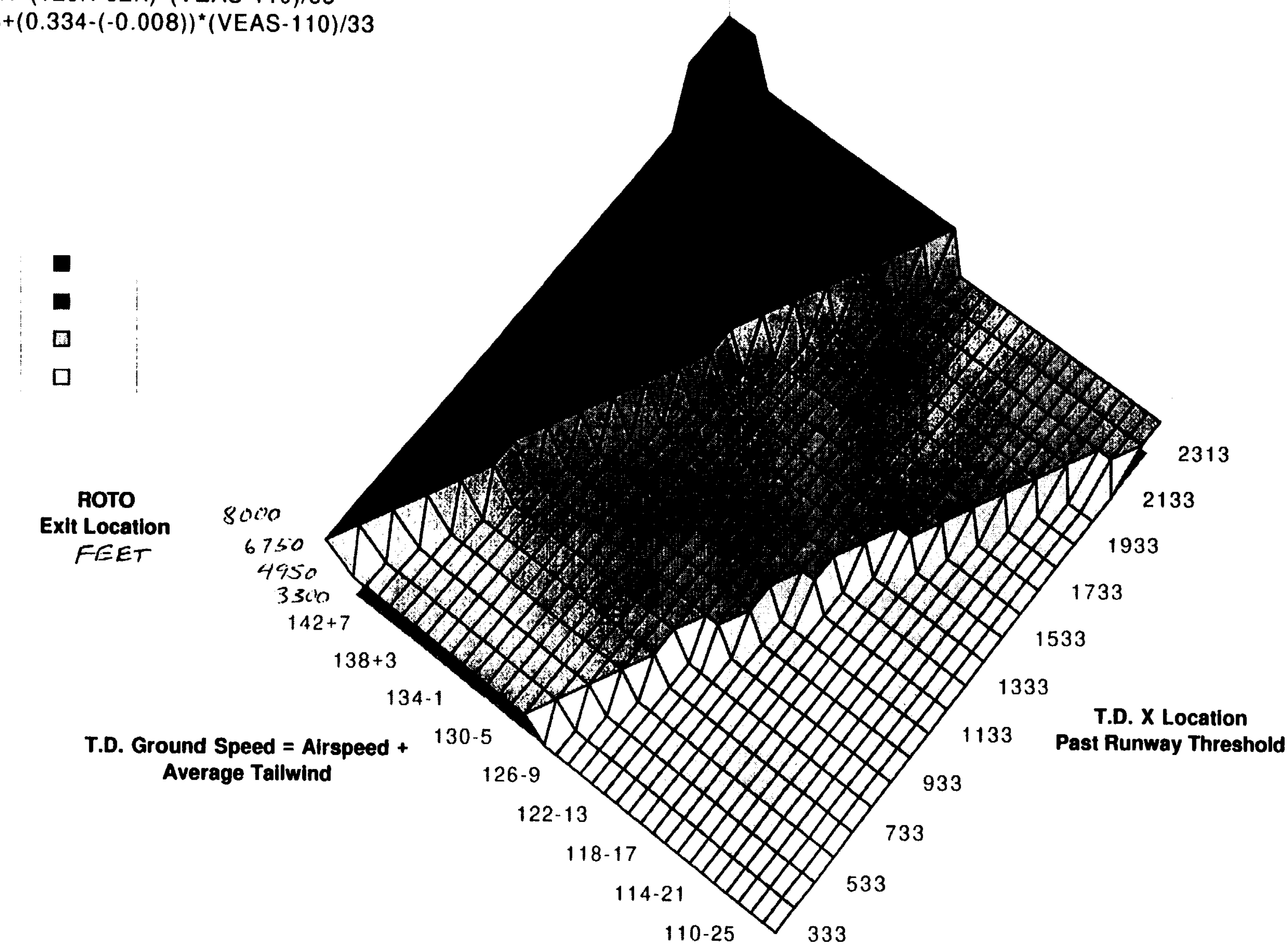
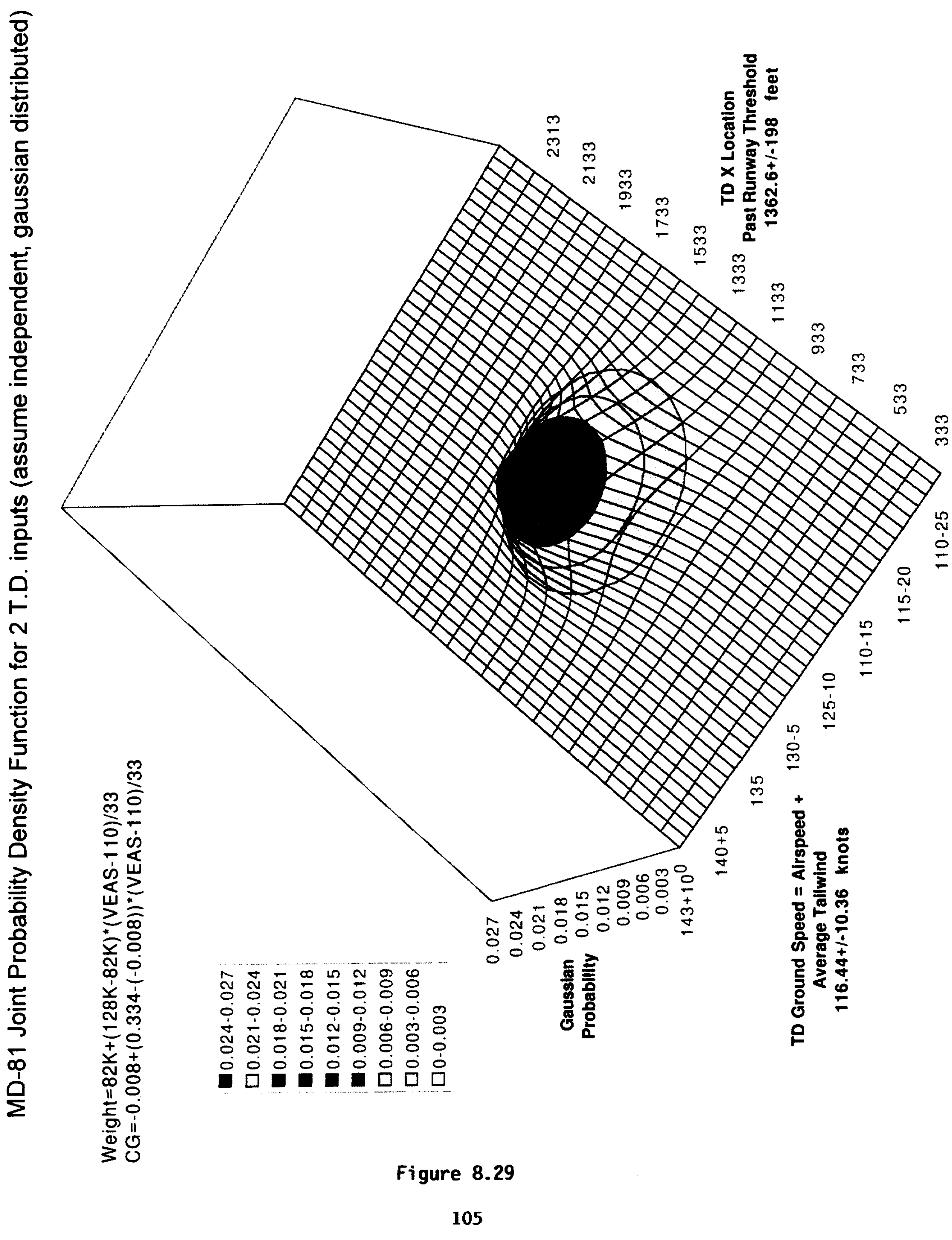


Figure 8.28



MD-81 Joint Probability Density Function for 2 T.D. inputs (assume independent, gaussian distributed)
Weight=82K+(128K-82K)*(VEAS-110)/33
CG=-0.008+(0.334-(-0.008))*(VEAS-110)/33
0.024-0.027
0.021-0.024
0.018-0.021
0.015-0.018
0.012-0.015
0.009-0.012
0.006-0.009
0.003-0.006
0-0.003
Gaussian Probability
0.027
0.024
0.021
0.018
0.015
0.012
0.009
0.006
0.003
0
143+10
140+5
135
130-5
125-10
110-15
115-20
110-25
TD Ground Speed = Airspeed + Average Tailwind
116.44+/-10.36 knots
333
533
733
933
1133
1333
1533
1733
1933
2133
2313
TD X Location
Past Runway Threshold
1362.6+/-198 feet

Figure 8.29

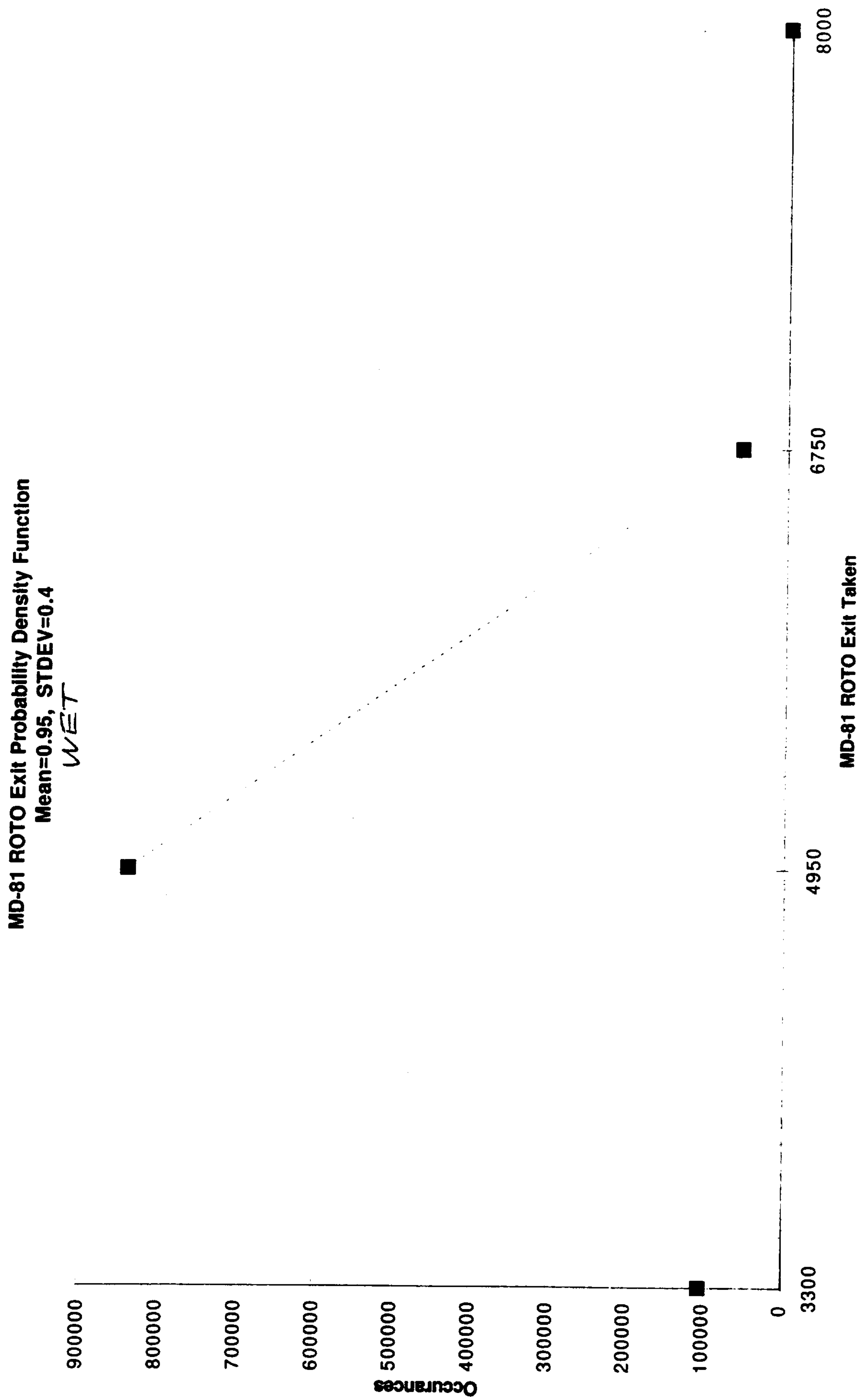

MD-81 ROTO Exit Probability Density Function
Mean=0.95, STDEV=0.4
WET
Occurances
900000
800000
700000
600000
500000
400000
300000
200000
100000
0
3300
4950
6750
8000
MD-81 ROTO Exit Taken

Figure 8.30

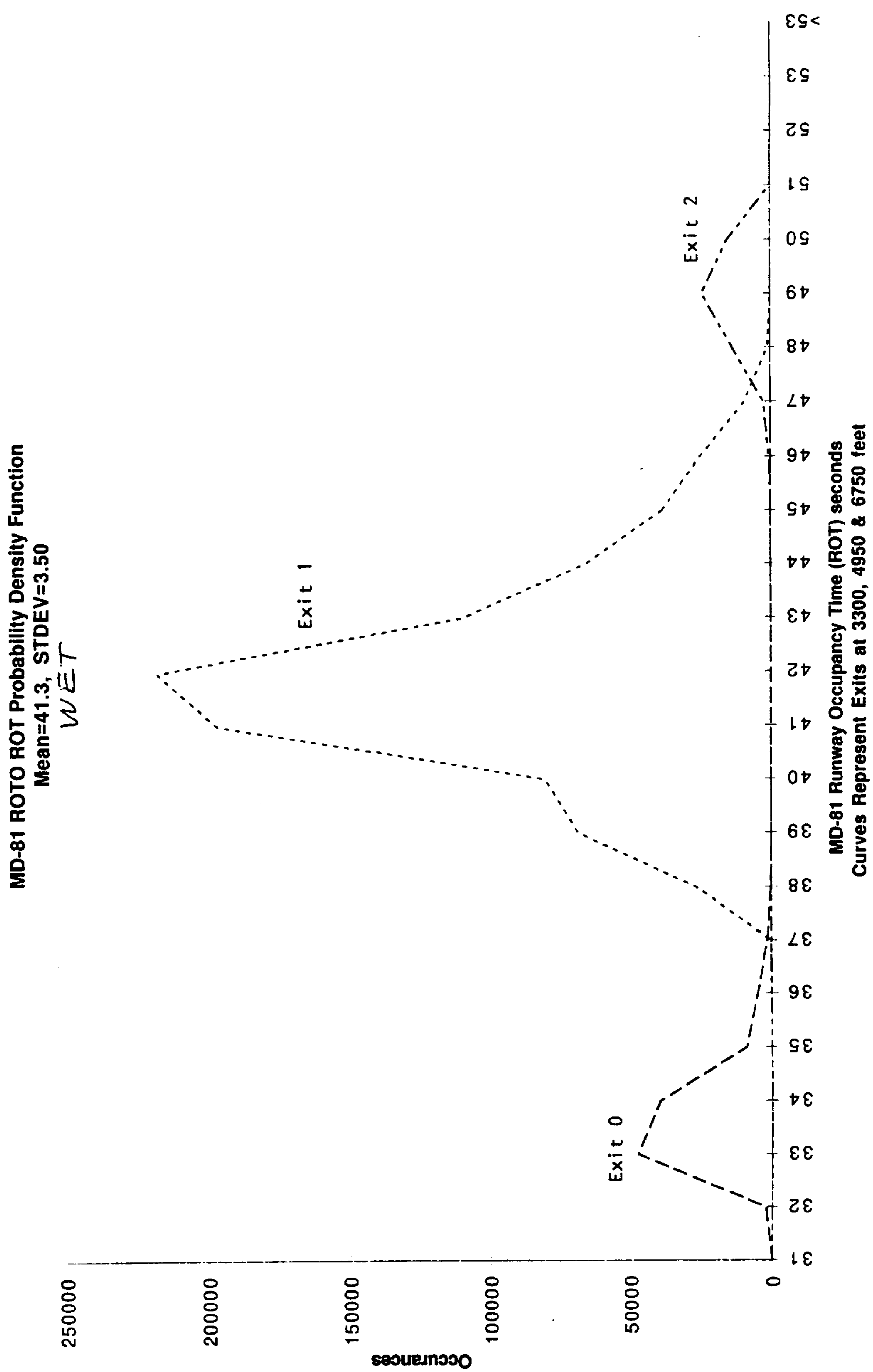

MD-81 ROTO ROT Probability Density Function
Mean=41.3, STDEV=3.50
WET
Exit 0
Exit 1
Exit 2
250000
200000
150000
100000
50000
0
Occurances
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
>53
MD-81 Runway Occupancy Time (ROT) seconds
Curves Represent Exits at 3300, 4950 & 6750 feet

Figure 8.31

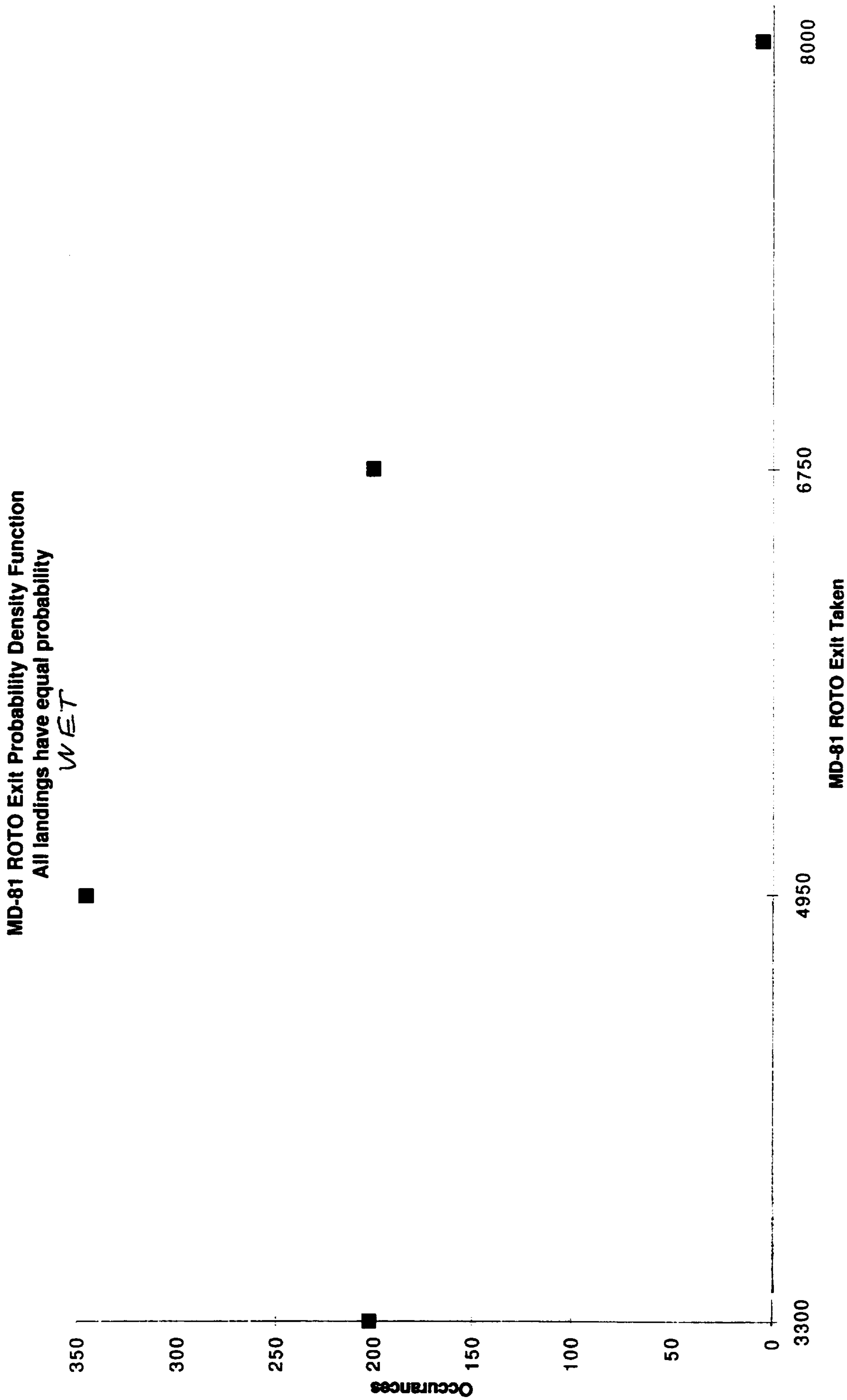

MD-81 ROTO Exit Probability Density Function
All landings have equal probability
WET
Occurances
350
300
250
200
150
100
50
0
3300
4950
6750
8000
MD-81 ROTO Exit Taken

Figure 8.32

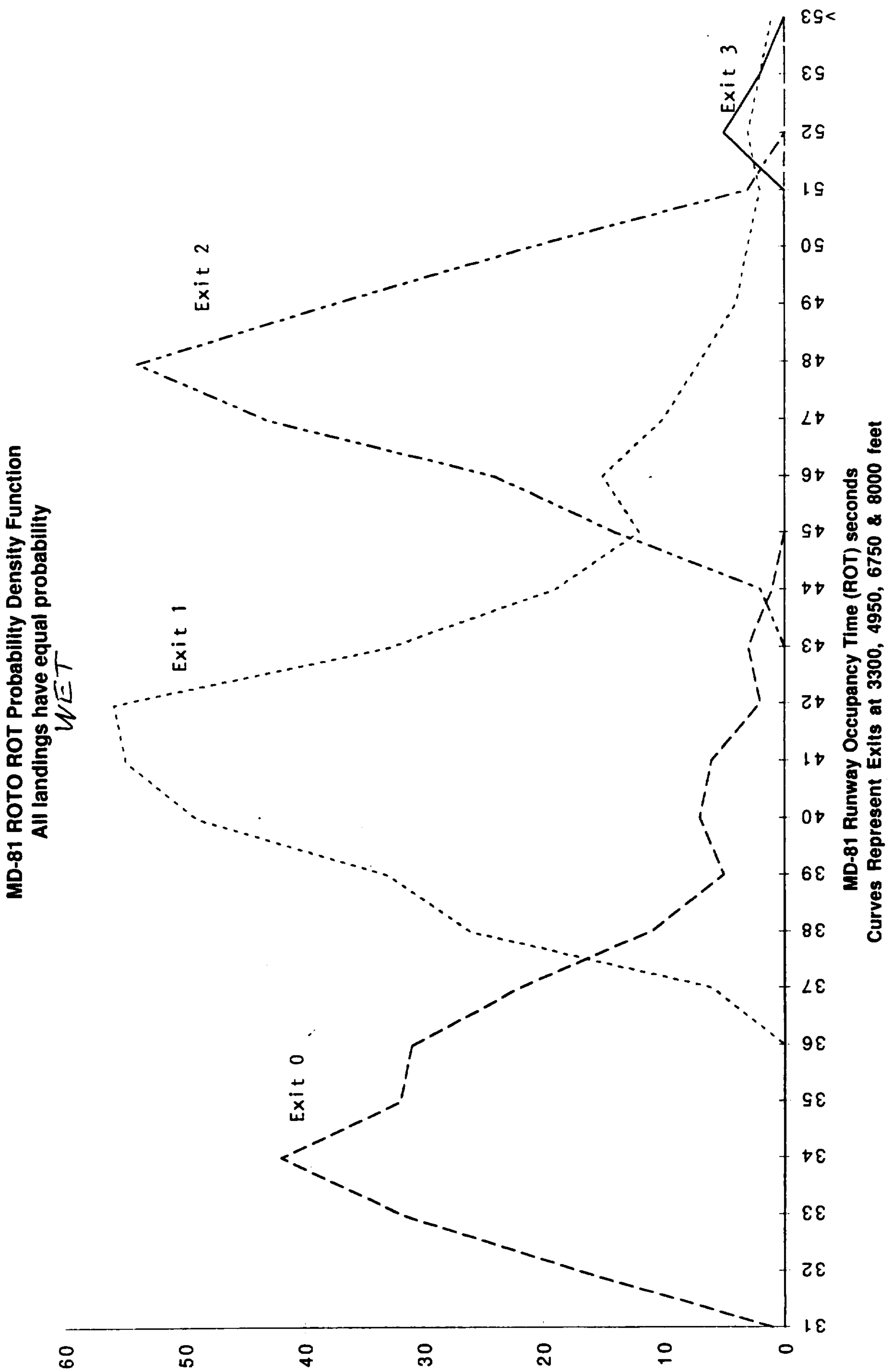

MD-81 ROTO ROT Probability Density Function
All landings have equal probability
WET
Occurances
0
10
20
30
40
50
60
31
32
33
34
35
36
37
38
39
40
41
42
43
44
45
46
47
48
49
50
51
52
53
>53
Exit 0
Exit 1
Exit 2
Exit 3
MD-81 Runway Occupancy Time (ROT) seconds
Curves Represent Exits at 3300, 4950, 6750 & 8000 feet

Figure 8.33

Table 3.1. SUMMARY - HAZARD CLASS I

SYSTEM FUNCTION	HAZARD DESCRIPTION	FAILURE CONDITION
Runway Acquisition Clearance	Object on runway	Possible crash
ROTO Steering Guidance	Erroneous steering guidance	Loss of control
ROTO Steering Guidance	Erroneous rudder	Loss of control
ROTO Steering Guidance	Erroneous nosegear steering	Loss of control

Table 3.2. SUMMARY - HAZARD CLASS II

SYSTEM FUNCTION	HAZARD DESCRIPTION	FAILURE CONDITION
ROTO Navigation	Loss of navigation data (GPS, IRS, airport data base)	Loss of ROTO capability Loss of control if loss occurs at critical time
ROTO Navigation	Erroneous navigation data	Loss of control
ROTO Steering Guidance	Loss of all guidance command	Loss of ROTO capability Loss of control if loss occurs at critical time
ROTO Steering Guidance	Loss of rudder	Loss of ROTO capability Loss of control if loss occurs at critical time
ROTO Steering Guidance	Loss of nosegear	Loss of ROTO capability Loss of control if loss occurs at critical time
ROTO Deceleration	Loss of deceleration command	Loss of ROTO capability Loss of control if loss occurs at critical time
ROTO Deceleration	Erroneous deceleration commands	Loss of ROTO capability Loss of control if loss occurs at critical time
ROTO Deceleration	Erroneous auto brake	Loss of ROTO capability Loss of control if loss occurs at critical time
ROTO Deceleration	Erroneous reverse thrust	Loss of ROTO capability Loss of control if loss occurs at critical time
ROTO Display	Loss of pilot display	Loss of ROTO capability Loss of control if loss occurs at critical time
ROTO Display	Loss of aircraft weight	Loss of ROTO capability

Table 3.3. SUMMARY - SYSTEM ARCHITECTURE RECOMMENDATIONS/CONSIDERATIONS HAZARD CLASS I

SYSTEM FUNCTION	HAZARD DESCRIPTION	CRITICAL SYSTEM BASELINE AIRCRAFT	CRITICAL SYSTEM ROTO SYSTEM	RECOMMENDATIONS/CONSIDERATIONS
Runway acquisition clearance	Object on runway	VHF com,data system	ROTO command system	The VHF communications and data system and the ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause loss of the VHF communications and data system and ROTO command system functions.
ROTO Steering Guidance	Erroneous steering guidance	Nosegear steering system	ROTO command system	The nosegear steering system, rudder control system, and the ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause erroneous steering guidance function.
		Rudder control system	Failure monitoring system	The ROTO system failure monitoring system must be designed so that no single failure or combination of failures not shown to be extremely improbable will cause loss of the failure monitoring system (undetected failure, misleading information, etc,)
ROTO Steering Guidance	Erroneous rudder	Rudder control system	ROTO command system	The rudder control system and the ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause erroneous rudder function.
			Failure monitoring system	The ROTO system failure monitoring system must be designed so that no single failure or combination of failures not shown to be extremely improbable will cause loss of the failure monitoring system (undetected failure, misleading information, etc,)

Table 3.3. SUMMARY - SYSTEM ARCHITECTURE RECOMMENDATIONS/CONSIDERATIONS HAZARD CLASS I (continued)

SYSTEM FUNCTION	HAZARD DESCRIPTION	CRITICAL SYSTEM		RECOMMENDATION/CONSIDERATIONS
		BASELINE AIRCRAFT	ROTO SYSTEM	
ROTO Steering Guidance	Erroneous nosegear steering	Nosegear steering system	ROTO command system	The nosegear steering system and the ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause erroneous nosegear steering function.
			Failure monitoring system	The ROTO system failure monitoring system must be designed so that no single failure or combination of failures not shown to be extremely improbable will cause loss of the failure monitoring system (undetected failure, misleading information, etc,)

Table 3.4. SUMMARY - SYSTEM ARCHITECTURE RECOMMENDATIONS/CONSIDERATIONS HAZARD CLASS I I

SYSTEM FUNCTION	HAZARD DESCRIPTION	CRITICAL SYSTEM BASELINE AIRCRAFT	CRITICAL SYSTEM ROTO SYSTEM	RECOMMENDATION/CONSIDERATIONS
ROTO Navigation	Loss of navigation data (GPS,IRS, airport data base)	Navigation data system	ROTO command system	The navigation data system and the ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause loss of navigation data.
			Failure monitoring system	The ROTO system failure monitoring system must be designed so that no single failure or combination of failures not shown to be extremely improbable will cause loss of the failure monitoring system (undetected failure, misleading information, etc,)
ROTO Navigation	Erroneous navigation data	Navigation data system	ROTO command system	The navigation data system and the ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause erroneous navigation data.
			Failure monitoring system	The ROTO system failure monitoring system must be designed so that no single failure or combination of failures not shown to be extremely improbable will cause loss of the failure monitoring system (undetected failure, misleading information, etc,)
ROTO Steering Guidance	Loss of all guidance command	Nosegear steering system Rudder control system	ROTO command system	The nosegear steering system, rudder control system, and the ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause the loss of all steering guidance command.
ROTO Steering Guidance	Loss of rudder	Rudder control system	ROTO command system	The rudder control system and the ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause loss of the rudder.
ROTO Steering Guidance	Loss of nosegear	Nosegear steering system	ROTO command system	The nosegear and the ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause loss of the nosegear.

Table 3.4. SUMMARY - SYSTEM ARCHITECTURE RECOMMENDATIONS/CONSIDERATIONS HAZARD CLASS I I (continued)

SYSTEM FUNCTION	HAZARD DESCRIPTION	CRITICAL SYSTEM BASELINE AIRCRAFT	CRITICAL SYSTEM ROTO SYSTEM	RECOMMENDATION/CONSIDERATIONS
ROTO Deceleration	Loss of deceleration command	Auto brake system Auto reverse thrust system	ROTO command system	The auto brake system, auto reverse thrust system , and ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause loss of the deceleration command.
ROTO Deceleration	Erroneous deceleration commands	Auto brake system Auto reverse thrust system	ROTO command system	The auto brake system, auto reverse thrust system, and ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause erroneous deceleration commands.
			Failure monitoring system	The ROTO system failure monitoring system must be designed so that no single failure or combination of failures not shown to be extremely improbable will cause loss of the failure monitoring system (undetected failure, misleading information, etc,)
ROTO Deceleration	Erroneous auto brake	Auto brake system	ROTO command system	The auto brake system and the ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause erroneous auto brake.
			Failure monitoring system	The ROTO system failure monitoring system must be designed so that no single failure or combination of failures not shown to be extremely improbable will cause loss of the failure monitoring system (undetected failure, misleading information, etc,)
ROTO Deceleration	Erroneous reverse thrust	Auto reverse thrust system	ROTO command system	The auto reverse thrust system and ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause erroneous reverse thrust.
			Failure monitoring system	The ROTO system failure monitoring system must be designed so that no single failure or combination of failures not shown to be extremely improbable will cause loss of the failure monitoring system (undetected failure, misleading information, etc,)

Table 3.4. SUMMARY - SYSTEM ARCHITECTURE RECOMMENDATIONS/CONSIDERATIONS HAZARD CLASS I I (continued)

SYSTEM FUNCTION	HAZARD DESCRIPTION	CRITICAL SYSTEM BASELINE AIRCRAFT	CRITICAL SYSTEM ROTO SYSTEM	RECOMMENDATION/CONSIDERATIONS
ROTO Display	Loss of pilot display	ROTO pilot display system	ROTO command system	The ROTO pilot display system and the ROTO command system must be designed so that no single failure or combinations of failures not shown to be extremely improbable will cause the loss of pilot display.
			Failure monitoring system	The ROTO system failure monitoring system must be designed so that no single failure or combination of failures not shown to be extremely improbable will cause loss of the failure monitoring system (undetected failure, misleading information, etc,)

N E T W O R K   R E L I A B I L I T Y   A N A L Y S I S   P R O G R A M

TABLE   ROTO SYSTEM PROBABILITY OF FAILURE

TIME(HRS)   0.1
P(FAIL)   0.206E-09

FILE: ROTO2
DATE: 08/30/95
LIMIT: 15

ID1	ID2	NOMENCLATURE	QTY	MTBF	P(F/PRFCT)	CONTRIB(%)
1	2	FMS DATA BASE	2	25000.	0.195D-09	5.4
3	5	GPS	3	48000.	0.206D-09	0.0
6	7	VHF	2	20000.	0.189D-09	8.4
8	9	ROTO COMPUTER	2	14000.	0.171D-09	17.2
10	11	FCC	2	12000.	0.158D-09	23.4
12	13	PILOT DISPLAY	2	15000.	0.175D-09	15.0
14	15	BRAKE COMPUTER	2	14000.	0.171D-09	17.2
16	16	BACKDRIVE SERVO TILLER	1	35000.	0.206D-09	0.0
17	18	BACKDRIVE SERVO RUDDER P	2	35000.	0.200D-09	2.8
19	20	AUTO THRUST REVERSE SERV	2	19000.	0.187D-09	9.3
21	22	NOSEGEAR STEERING ACTUAT	2	50000.	0.203D-09	1.3

Table 3.5

Appendix A ROTO MODEL TABLE OF CONTENTS

MAKEFILE is a file used by MAKE to create C object files required by MODELROTOBIG.M, which are then converted to CMEX files.

```
CLIBS = -lm -lmalloc -lc
CFLAGS = +z -z -Aa -Ae -g -DDAY_FMEX -I$(HOME)/matlab/extern/include
atmos : funcnosetireforces.o
        cmex funcnosetireforces.o
.f.o:
        cc $(CFLAGS) funcnosetireforces.c

atmos : funcinput.o
        cmex funcinput.o
.f.o:
        cc $(CFLAGS) funcinput.c

atmos : funcinputnav.o
        cmex funcinputnav.o
.f.o:
        cc $(CFLAGS) funcinputnav.c
atmos : funcinterpr.o
        cmex funcinterpr.o
.f.o:
        cc $(CFLAGS) funcinterpr.c
```

RUNROTO.M is the main MATLAB script file to run the ROTO simulation from the MATLAB command line. It calls the MATLAB SIMULINK model MODELROTOBIG.M, which you may view by typing modelrotobig from the MATLAB command line. The following pages contain the input MATLAB script files and exit prediction algorithm files.

```
% Initialize all input variables
clear all;
format short e;
% if you were to do multiple runs, do them here with   clear variables;
% this will not reload the model and functions
centerline;
centerline22;
 % generic airplane
 inputdef;
 % unique md11 characteristics
 indefmd11;
 % run specifics
 inmd1170;
 % some additional initializations
 beforeloop;
save md1170;
createarrays;
globalfile;

% Predict exit selection before touchdown
% create U_TD,UT_TD,X_TD; input files currently set these values,
% In reality they would need to be predicted on the glideslope
U_TD=U;
UT_TD=UT;
X_TD=X;
VWSS_TD=VWSS;
lastnexit=0;
NONROTO=0;
while ((lastnexit~=NEXIT)&(NEXIT<5))
 lastnexit=NEXIT;
 abortearly=funcpredictabort(U_TD,UT_TD,X_TD,VWSS_TD,VEXIT,EXITPOS,NEXIT);
 if(abortearly)
   NEXIT=NEXIT+1;
 end;
 if(NEXIT>4)
   NONROTO=1;
 end;
 EXITPOS=EXITLOC(NEXIT);
end;


% ROTO Simulation
if(NONROTO==0)
 % Run roto simulation
 TIME=0;
 timt=0;
 TSTEP=0;
 [T,AEROX,AEROY]=linsim('modelrotobig',70,[],[DT,DT,DT,0,3,2]);
 % Write out results
 fid=fopen('nose.output1.data','w');
 scriptwriteplotdata;
```

```
else
  disp('NON-ROTO LANDING'); end;
```

VARIABLE DEFINITIONS

Variable definitions used in the ROTO simulation.

The following defines output variables:

Variable	Definitions
X	Aircraft longitudinal position on the runway
Y	Aircraft lateral position on the runway
YSTRIPE	Runway and exit centerline lateral position
U	Aircraft ground speed in knots (inside sim it is fps)
PSI	Aircraft heading
DV	Aircraft lateral accelleration
DU	Aircraft longitudinal accelleration
DY	Aircraft lateral deviation rate
DPSI	Aircraft heading rate
SIGSTR	Aircraft nose wheel angle relative to aircraft heading
SIG	Aircraft nose wheel angle
DELRC	Rudder command
DPERP	Aircraft lateral deviation
VPERP	Aircraft lateral deviation rate
DDPSI	Aircraft heading acceleration
FNOSE	Aircraft nose tire side force
FC	Aircraft center tire side force, after limited to avail. friction
FM	Aircraft R&L tire side force, after limitied to avail. friction
PMM	Aircraft R&L tire vertical force
PSIP	Aircraft heading relative to centerline heading
THST, THRUST	Aircraft total thrust
TR	Aircraft rudder moment
MNAERO	Aircraft side slip moment
PSIC	Aircraft center gear heading
NC	Aircraft center gear cornering power
PSICGV	Aircraft center gear heading relative to aircraft velocity vector
MUMAX	Runway surface available friction coef. at nose gear
MUMAXR	Runway surface available friction coef. at right main gear
MUMAXL	Runway surface available friction coef. at left main gear
MUMAXC	Runway surface available friction coef. at center main gear
MUNOSE	Friction used by nose gear
MUR	Friction used by right main gear
MUL	Friction used by left main gear
MUC	Friction used by center main gear
LATG	Aircraft lateral G
LONGG	Aircraft longitudinal G
LATJ	Aircraft lateral jerk
LONGJ	Aircraft longitudinal jerk
YAW	Aircraft yaw angle
DRU	Aircraft upper rudder angle
DRUR	Aircraft upper rudder angle rate
DRUDEAD	Aircraft upper rudder value after deadband
DRMPUU	Aircraft upper rudder mod piston position
DECELCOM	Aircraft symmetrical deceleration command
NEXIT	Designnated exit # (1,2,or 3)
RUNWAYTIME	Aircraft runway occupancy time
POSEXIT,NEGEXIT	Aircraft main gear must stay within these lateral bounds
PERCENT BRK	Avg. Percent of maximum brake pressure used by all main gear
TAILWIND	knots
CROSSWIND	knots

XNAV	DGPS noise added to actual aircraft X position
YNAV	DGPS noise added to actual aircraft Y position
ELEV	Aircraft elevator angle

The following describes input contants:
*

Variable	Definition	
* A	DISTANCE--NLG TO CG	FT
* ASEFF	ANTISKID EFFICIENCY	—
* B	DISTANCE--CG TO MLG	FT
* BC	DISTANCE--CG TO CLG	FT
* BW	DISTANCE -- CG TO	FT
* CMR	RUDDER YAW MOMENT COEFF.	1./DEG
* C	LIFT MOMENT ARM TO CG	FT
* CDRAG	A/C AERODYNAMIC DRAG COEFF.	—
* CLIFT	A/C AERODYNAMIC LIFT COEFF.	—
* CMOM	A/C AERODYNAMIC MOMENT COEFF.	—
* CNB	AERODYNAMIC SIDE-SLIP MOMENT COEFF.	—
* CYB	AERODYNAMIC SIDE-SLIP FORCE COEFF.	—
* DELB	NLG TIRE DEFL., AT LOAD RB	IN
* DELS	NLG TIRE DEFL., RATED	IN
*DECELSET	AUTOBRAKE DECELERATION SETTING	FT/SEC
* HCG	CG HEIGHT	FT
* HS	NLG TIRE SECTION HEIGHT	IN
* IYAW	A/C YAW MOMENT OF INERTIA (ABOUT CG)	SLUG-FT**2
* LMAC	LENGTH OF THE MAC	FT
* LTAIL	DISTANCE--CG TO TAIL CP	FT
* NM	MLG TIRE CORNERING POWER (PER TIRE)	LB/DEG
* OD	NLG TIRE OD	IN
* RB	NLG TIRE VERT LOAD AT DEFL. DELB (FOR SPRING RATE CALC.)	LB
* RP	NLG TIRE RATED PRESS. (LOADED)	PSI
* RS	NLG TIRE RATED LOAD	LB
* SP	NLG TIRE STATIC PRESSURE (LOADED)	PSI
* SPM	MLG TIRE STATIC PRESSURE (LOADED)	PSI
* S	NOSEWHEEL SPACING	IN
* SW	A/C WING AREA	FT**2
* THETA	NLG STRUT CANT ANGLE	DEG
* TREAD	DISTANCE BETWEEN MLG'S	FT
* VEAS	INITIAL AIRSPD A/C VELOCITY	KTS
* VW	CROSSWIND VELOCITY	KTS
* VTW	TAILWIND VELOCITY	KTS
* W	A/C WEIGHT	LB
* WET	SWITCH TO USE WET RUNWAY FRICTION (1 IS ON)	—
* ICE	SWITCH TO USE ICY RUNWAY FRICTION (2 IS ON)	—
* SNOW	SWITCH TO USE SNOWY RUNWAY FRICTION (3 IS ON)	—
* SLUSH	SWITCH TO USE SLUSHY RUNWAY FRICTION (4 IS ON)	—
* FLOOD	SWITCH TO USE FLOODED RUNWAY FRICTION (5 IS ON)	—
* WS	NLG TIRE SECTION WIDTH	IN
* FLATMU	SWITCH FOR FLAT CORNERING MU VS. SKID VEL. CURVE (1 IS FLAT)	-----
* KS1	(STEERING VALVE SPOOL DISPLACEMENT)/(STEER ERR)	IN/DEG
* KS2	(STEERING RATE)/(VALVE FLOW)	(DEG.SEC)/(IN**3/SEC)
* KS3	(STEERING ACTUATOR PRESSURE)/(STRT GD MOM)	(PSI/IN-LB)
* PSUP	HYDRAULIC SUPPLY PRESSURE	PSI
* PRET	HYDRAULIC RETURN PRESSURE	PSI
* TMGD	MAIN GEAR TOUCHDOWN TIME	SEC
* TNGD	NOSE GEAR TOUCHDOWN TIME	SEC
* TSPOIL	TIME BETWEEN MAIN GEAR TOUCHDOWN & SPL DEPLOYMENT	SEC
* TDELAY	TIME BETWEEN SPOILER DEPLOYMENT & START OF BRKE RP	SEC
*RRPHASE1	PHASE 1 BRAKE PRESSURE RAMP RATE	PSI/SEC
*RRPHASE2	PHASE 2 BRAKE PRESSURE RAMP RATE	PSI/SEC
*KBPHASE1	PHASE 1 BRAKE PRESSURE GAIN	(PSI/SEC)/(FT/SEC**2)
*KBPHASE2	PHASE 2 BRAKE PRESSURE GAIN	(PSI/SEC)/(FT/SEC**2)

* NWWLG	NUMBER OF WING GEAR WHEELS	—
* NWCLG	NUMBER OF CENTER GEAR WHEELS	—
* BDEXP	CONSTANT USED TO DEFINE BRAKE DRAG VS. PRESSURE CURVE	-----
* BDP	CONSTANT USED TO DEFINE BRAKE DRAG VS. PRESSURE CURVE	PSI
* BDK	CONSTANT USED TO DEFINE BRAKE DRAG VS. PRESSURE CURVE	LBS/PSI

CENTERLINE.M is a file that gives coordinates for the spiral exit centerline every 2 feet apart along its length and is called by RUNROTO once. Because of it many pages of data it is not included in this report. NASA has the file in electronic form. This file is not used by the aircraft navigation. It is used in the simulation by FUNCINPUT to find the actual centerline lateral error of the aircraft for performance evaluation purposes.

CENTERLINE22.M is a file containing the coordinates and radius of the spiral exit centerline and is called by RUNROTO once. The points are spaced so that the perpendicular error from the actual path is less than 0.5 feet. There are about 26 points. The points get closer together as the path radius decreases. The database requirements section recommends that the exit's X,Y coordinates be in units of 1/2 foot using 2 byte integers. This particular input file has stored the X,Y coordinates in units of feet, but the X,Y coordinates have no greater resolution than 1/2 foot.

```
spiralexit22 = [
 0.00000D+00...
 0.00000D+00...
 0.32767D+5...
;
 0.17600D+03...
 0.5D+00...
 0.10634D+05...
;
 0.354D+03...
 0.400D+01...
 0.52872D+04...
;
 0.490D+03...
 0.105D+02...
 0.38197D+04...
;
 0.6015D+03...
 0.195D+02...
 0.31091D+04...
;
 0.7105D+03...
 0.32D+02...
 0.26287D+04...
;
 0.8055D+03...
 0.47D+02...
 0.23164D+04...
;
 0.9075D+03...
 0.675D+02...
 0.20523D+04...
;
 0.985D+03...
 0.865D+02...
 0.18868D+04...
;
 0.10695D+04...
 0.1115D+03...
 0.17330D+04...
;
 0.1145D+04...
 0.1375D+03...
 0.16135D+04...
;
```

```
 0.1219D+04...
 0.1675D+03...
 0.15094D+04...
;
 0.1290D+04...
 0.201D+03...
 0.14201D+04...
;
 0.1343D+04...
 0.229D+03...
 0.13582D+04...
;
 0.1362D+04...
 0.2395D+03...
 0.13369D+04...
;
 0.1362D+04...
 0.2395D+03...
 0.32767D+5...
;
 0.1800D+04...
 0.4925D+03...
 0.32767D+5...
;
 0.1800D+04...
 0.4925D+03...
 -.80000D+03...
;
 0.18495D+04...
 0.519D+03...
 -.80000D+03...
;
 0.1901D+04...
 0.5415D+03...
 -.80000D+03...
;
 0.19555D+04...
 0.5615D+03...
 -.80000D+03...
;
 0.2009D+04...
 0.577D+03...
 -.80000D+03...
;
 0.2064D+04...
 0.588D+03...
 -.80000D+03...
;
 0.21195D+04...
 0.596D+03...
 -.80000D+03...
;
 0.21775D+04...
 0.5995D+03...
 -.80000D+03...
;
```

```
 0.21995D+04...
 0.60000D+03...
 -.80000D+03...
;

 0.22015D+04...
 0.60000D+03...
 0.32767D+5...
;
];
spiralexitx22=spiralexit22(:,1);
spiralexity22=spiralexit22(:,2);
spiralexitr22=spiralexit22(:,3);
clear spiralexit22;
```

INPUTDEF.M is a MATLAB script file which initializes the default simulation variables for all aircraft types and flight conditions and is called by RUNROTO once.

```
    false=0;
    true=1;
ASEFF =  0.75;
DT =  1.0E-03;
DECELSET =  0.0;
ENDTIM =  350.0;
INTVL =  0.5;
VEAS =  166.;
VSTOP =  1.0;
VWSS =  15;
VTWSS =  10.;
WET =          1;
MUROLL =  0.15;
PSUP =  3000.;
PRET =  60;
TMGD =  0.0;
TNGD =  6.0;
TSPOIL =  1.3;
TDELAY =  3.0;
RRPHASE1 =  400.;
RRPHASE2 =  1200.;
KBPHASE1 =  600.;
KBPHASE2 =  1800.;
FLATMU =          1;
RUNWAYWIDTH=150;
EXITWIDTH=90;
% table lookup uses last four points to extrapolate out of table, if you want
% a limit in the table, use four points at each end with that limit.
MUSKIDT = [
 0.000000000E+00;
  20;
  40;
  60;
  100;
  100.0;
];
MUSKID = [
 1.0;
 0.6;
 0.42;
 0.36;
 0.32;
 0.32;
];
STEERT = [
 -23;
 -18.9;
 -12.6;
 -6.3;
 -3.15;
 0.0;
  3.15;
  6.3;
  12.6;
  18.9;
  23;
```

```
];
STEER = [
  8.4;
  7.;
  4.85;
  2.55;
  1.3;
 0.0;
 -1.3;
 -2.55;
 -4.85;
 -7.;
 -8.4;
];
 WETMUT = [
 0.0;
  20.;
  40.;
  50.;
  70.;
  80.;
  100.;
  120.;
  140.;
  160.;
  180.;
];
WETMU = [
 0.941;
 0.533;
 0.378;
 0.333;
 0.282;
 0.271;
 0.266;
 0.266;
 0.266;
 0.27;
 0.274;
];
ICEMUT = [
 0.0;
  200.;
];
ICEMU = [
 0.27E-01;
 0.27E-01;
];
SNOWMUT = [
 0.0;
  200.;
];
SNOWMU = [
 0.162;
 0.162;
];
SLUSHMUT = [
 0.0;
```

```
  200.;
 ];
 SLUSHMU = [
  0.55;
  0.55;
 ];
 FLOODMUT = [
  0.0;
  10.;
  20.;
  30.;
  40.;
  50.;
  60.;
  70.;
  80.;
  90.;
  100.;
  110.;
  120.;
  130.;
  140.;
  150.;
  160.;
  170.;
  180.;
  190.;
  200.;
 ];
 FLOODMU = [
  0.929;
  0.645;
  0.465;
  0.353;
  0.276;
  0.22;
  0.18;
  0.147;
  0.119;
  0.98E-01;
  0.84E-01;
  0.72E-01;
  0.62E-01;
  0.55E-01;
  0.53E-01;
  0.5E-01;
  0.51E-01;
  0.52E-01;
  0.53E-01;
  0.54E-01;
  0.55E-01;
 ];
```

INDEFMD11.M is a file initializes MD-11 aircraft characteristics independent of flight contidion and is called by RUNROTO once.

```
A =  78.2562256;
B =  2.45735645;
BC =  5.00903320;
BW =  165.369904;
C =  2.21834278;
CDRAG =  0.1651;
CLIFT =  0.29;
CMOM =  0.160003304E-01;
CMR = -0.262E-02;
CNB =  0.37E-02;
CYB = -0.24E-01;
DELB =  1.1;
DELS =  3.4;
HCG =  2.71658039;
HS =  9.8;
IYAW =  26000000.0;
LMAC =  24.6479797;
LTAIL =  83.7390137;
NC =  4426;
NM =  4806.0;
OD =  39.6;
RB =  8000;
RP =  203;
RS =  39500;
SP =  167;
SPM =  188;
S =  25.;
SW =  3647.5;
WINGSPAN=83;
TREAD =  34.677;
THETA =  9.5;
W =  480000.;
KS1 =  0.873E-02;
KS2 =  0.965;
KS3 =  0.842E-02;
BDEXP=0.7;
BDP=125;
BDK=190.6;
NWWLG =          4;
NWCLG =          2;
WS =  15.5;
% table lookup uses last four points to extrapolate out of table, if you want
% a limit in the table, use four points at each end with that limit.
THS1T = [
 0.0;
 66.1249695;
 132;
];
THS1 = [
 15048;
 10602;
 7335;
];
THSRT = [
 85.0;
 123.84;
```

```
126;
135;
140;
];
THSR = [
-28740;
-40700;
-40900;
-41460;
-42630;
];
THSTIT = [
9.5;
7.0;
5.5;
3.5;
0.0;
];
THSTI = [
-39350;
-23340;
-7060;
6800;
6750;
];
THSNT = [
0.0;
66.125;
132;
];
THSN = [
7581;
4680;
2535;
];
THSRNT = [
0.0;
60.889;
65.65;
84.96;
123.84;
125.97;
134.85;
140.18;
147.29;
150.0 ;
];
THSRN = [
-1400.0;
-6350.0;
-6750.0;
-8425.0;
-11900.0;
-12150.0;
-13050.0;
-14200.0;
-15475.0;
-16200.0;
```

```
 ];
GAMMAT = [
  200000.;
  257500.;
  280000.;
  300000.;
  330000.;
  365000.;
  405000.;
  440000.;
  480000.;
  510000.;
  550000.;
 ];
 GAMMA = [
  0.224;
  0.2;
  0.192;
  0.186;
  0.178;
  0.17;
  0.162;
  0.156;
  0.15;
  0.146;
  0.142;
 ];
 VALVET = [
-0.13;
-0.12;
-0.11;
-0.11;
-1.E-01;
-0.9E-01;
-0.8E-01;
-0.7E-01;
-0.6E-01;
-0.5E-01;
-0.4E-01;
-0.3E-01;
-0.2E-01;
-0.1E-01;
 0.0;
 0.1E-01;
 0.2E-01;
 0.3E-01;
 0.4E-01;
 0.5E-01;
 0.6E-01;
 0.7E-01;
 0.8E-01;
 0.9E-01;
 1.E-01;
 0.11;
 0.11;
 0.12;
 0.13;
];
```

```
VALVE = [
-0.91;
-0.91;
-0.91;
-0.91;
-0.875;
-0.765;
-0.6;
-0.455;
-0.34;
-0.245;
-0.165;
-0.9E-01;
-0.3E-01;
 0.0;
 0.0;
 0.0;
 0.3E-01;
 0.9E-01;
 0.165;
 0.245;
 0.34;
 0.455;
 0.6;
 0.765;
 0.875;
 0.91;
 0.91;
 0.91;
 0.91;
];
```

INMD1170.M is a file initializes variables specific to a flight condition and is called by RUNROTO once.

```
MD11=true;
 DT =  0.005;
 DECELSET =  0.0;
 ENDTIM =   100;
 INTVL =  0.5;
% veas range 130 kt to 166 kt
 VEAS =   166;
 VTWSS =   10.;
 VWSS =   15;
% weight range 340k to 480k
 W =  480000;
 WET =          1;
 KYAWRT =   7.75;
 KYDI =  5;
 KY =  12;
 KRAD =  -40000;
 XSTART =  2500;
 RCMDLM =   1000.0;
 EXSTOP =  1.0;
 RADTAU =   1.0;
 IG3 =  0.1;
 IG5 =  0.002;
 IG16 =  0.06;
 IG14 =   23.0;
 IKBANDRD =  0.85;
 IKMECH =          0;
 NSDEAD =  0.01;
 NSRTLM =  0.3;
 NSPSLM =   8.0;
 KBANDNS =  0.7;
 DECLIM =  9.0;
 COMPUP =  0.05;
 ILATBIAS =  0.0;
 ITURB =          false;
 NAVUP =  0.1;
 IRANDM =          false;
 ACCURRMS =  4.0;
 ACCURTAU =  30.0;
% CG range .12 to .34
 CG =  0.34;
 % CALCULATE NEW CG VARIABLES BASED ON FRACTION CG
      CGIN=1311.947+CG*295.779;
      A=(CGIN-473.437)/12.;
      B=(1442-CGIN)/12.;
      BC=(1472.62-CGIN)/12.;
      C=(CGIN-(1311.947+0.25*295.779))/12.;
     HCG=(209.32-(sqrt(CGIN^2+(-21)^2))*sin(atan(21/CGIN)+0.00193))/12.;
      LTAIL=(2417.38-CGIN)/12.;
% SET IH BASED ON CG
     PCTMAC=100.*CG;
    ELEV= 8;
    CGFWD=12;
    CGAFT=34;
    CLFWD=0.123;
    CLAFT=0.226;
    CMFWD=0.515;
```

```
      CMAFT=0.216;
      CDFWD=0.1746;
      CDAFT=0.1651;
      DCLDE=0.008;
      DCMDE=-0.025;

 % other constants
  BUFSIZ = 0.15;
 % DT WOULD HAVE BEEN COMPUP IF IT WERE RUNNING AT COMPUP
 % RADLAG=exp(-DT/RADTAU);
  RUDLIM = 23.;
  C5     = 0.07;
  KBYR2  = 7.75;
  KLAG   = 0.9;
  GSPFLL = 130.0;
  R2D    = 180./3.141593;
  REXP2  = exp(-DT / 10.0);
  REXP3  = exp(-DT / 1.0);
  C16    = .084;
  RUDRLM = 30.0;
  KBYRDT=10.0;
  VTW=VTWSS;
  AUTOREVERSE=1;
 % von karman filter
      WNYQ = 3.1416/DT;
      SRNYQ = sqrt(WNYQ);
      WD = 647.9/ 1750.;
      SG=VWSS/1.5;
      GD = SG*sqrt(1.0/(WD*3.1416));
      WF = 100. * WD;
 % VON KARMAN COEFFICIENTS
      VAPB=(1.339+1.118)/WD;
      VATB=1.339*1.118/WD/WD;
      VCPD=(0.1277+0.0146)/WD;
      VCTD=0.1277*0.0146/WD/WD;
      VEPF=(2.187+0.1833)/WD;
      VETF=2.187*0.1833/WD/WD;
      VG=0.021/WD;
```

BEFORELOOP.M is a file initializing additional inputs prior to the simulation beginning and is called by RUNROTO once. This file is not aircraft type specific. Some if statements are aircraft type specific.

```
% FORCE MU TO BE FLAT FOR NON-DRY CONDITIONS
    if(WET>0)
     FLATMU=true;
    else
     FLATMU=false;
    end;

    G3=IG3;
    G20=IG3;
    G5=IG5;
    G16=IG16;
    G14=IG14;
    KBANDRD=IKBANDRD;
    LATBIAS=ILATBIAS;
    VTW=VTWSS;
    VW=VWSS;
    CLIFT=CLFWD+((CLAFT-CLFWD)/(CGAFT-CGFWD))*(PCTMAC-CGFWD)+DCLDE*ELEV;
    CDRAG=CDFWD+((CDAFT-CDFWD)/(CGAFT-CGFWD))*(PCTMAC-CGFWD);
    CMOM =CMFWD+((CMAFT-CMFWD)/(CGAFT-CGFWD))*(PCTMAC-
CGFWD)+DCMDE*ELEV;


% INITIALIZATION  AND CALCULATIONS
    PI    = 3.141592654;
    DPSI=0.0;
    PSI= 0.0;
% FOR SOME REASON, EXITPOS VARIABLE DOES NOT LIKE TO BE NEGATIVE
% MAKE XSTART SO THAT FIRST EXIT IS AT 0
    X=XSTART;
    Y=0.0;
    SIG=0.0;
    DELRC=0.0;
    K1= (RS-RB)/(DELS-DELB);
    K2 = RB/DELB;
    SST=S*sin(THETA/57.3);
    D1 = DELB-RB/K1;
%  U IS A GRND SPD
    U=(VEAS+VTWSS)*1.689;
    V=0.0;
    DX = U*cos(PSI) - V*sin(PSI);
    DY = V*cos(PSI) + U*sin(PSI);
    TRQLAG=exp(-DT/0.005);
    COUNT=0.0;
    ACCURLAG=exp(-DT/ACCURTAU);
    ASYMBRAKE=false & (NWCLG~=0);
% XY ACCUR      SETUP
    ACCURRMS = sqrt(3.1416/DT)*ACCURRMS;
    BUFSIZ=0.15*0;
% patch constants
    WETPATCH=WET;
    PATCHNOW=false;
    PATCHLEN=10;
    PATCHTYPE=5;
    FIRSTPATCH=0;
```

```
LASTPATCH=11500;
PATCHOFTEN=10;
DISPLN=2250;
if( MD11)
 if(DISPLN<1150)
    EXITLOC(1)=4950;
    EXITLOC(2)=6750;
    EXITLOC(3)=8000;
    EXITLOC(4)=10000;
 else
    EXITLOC(1)=4950;
    EXITLOC(2)=6750;
    EXITLOC(3)=8000;
    EXITLOC(4)=10000;
 end;
else
 if(DISPLN<1150)
    EXITLOC(1)=3300;
    EXITLOC(2)=4950;
    EXITLOC(3)=6750;
    EXITLOC(4)=8000;
    EXITLOC(5)=10000;
 else
    EXITLOC(1)=3300;
    EXITLOC(2)=4950;
    EXITLOC(3)=6750;
    EXITLOC(4)=8000;
    EXITLOC(5)=10000;
 end;
end;
%other
 AUREVSEC= 3.0;
 AUREVTHIRD= 1;
 if(ITURB==1)
   RUDDEF= 0;
 else
   RUDDEF=VWSS;
 end;
 DECELON=false;
 LASTDECCR=0;
 LASTDECCL=0;
 DECELSET=0;
 REVERSE=true;
 AUTOREV=true;
 USEMUABORT=true;
 NEXIT=1;

 EXITPOS=EXITLOC(NEXIT);
% ASSUME PLANE IS DECRABED
 UT =(U - sin(0)*(VWSS*1.689) - cos(0)*(VTWSS*1.689))/1.689;
        VEXIT= 70.0*1.689;
INITRUNTIME=1/(U/X);
OUTARRSIZE=ENDTIM/DT;
 TSTEP=0;
timt=0;
TIMEOCCLAG=(9000-X)/((VEXIT+U)/2.2)+INITRUNTIME;
SWITCHDIV=0.0000000001;
ALSAMPLE=DT;
```

```
LOWSPD=20*1.689;
BRKBUF=100;
MANUAL=0;
DRATE=8.0 ;
DTEMP=7.5;
DTIME=41;
```

CREATEARRAYS.M is a file which declares array variables (mostly output). which will decrease the simulation run time. It called by RUNROTO once.

```
RWFC = zeros(size(1:5000));
    for  i = 1:5000
       RWFC(i)=WET;
    end;
  breakoutin=(BDP:3000+BDP);
 breakout=BDK*(breakoutin-BDP).^BDEXP;

    XTH = zeros(OUTARRSIZE,1);
    DXTH = zeros(OUTARRSIZE,1);
    DUTH = zeros(OUTARRSIZE,1);
    UTH = zeros(OUTARRSIZE,1);
    YTH = zeros(OUTARRSIZE,1);
    DYTH = zeros(OUTARRSIZE,1);
    DVTH = zeros(OUTARRSIZE,1);
    PSITH = zeros(OUTARRSIZE,1);
    DPSITH = zeros(OUTARRSIZE,1);
    DDPSITH = zeros(OUTARRSIZE,1);
    SIGTH = zeros(OUTARRSIZE,1);
    SIGSTRTH = zeros(OUTARRSIZE,1);
    LATGTH = zeros(OUTARRSIZE,1);
    LATJTH = zeros(OUTARRSIZE,1);
    DRUTH = zeros(OUTARRSIZE,1);
    DRURTH = zeros(OUTARRSIZE,1);
    DRUDEADTH = zeros(OUTARRSIZE,1);
    DRMPUUTH = zeros(OUTARRSIZE,1);
    DECELTH = zeros(OUTARRSIZE,1);
    EXITPOSTH = zeros(OUTARRSIZE,1);
    NEXITTH = zeros(OUTARRSIZE,1);
    VEXITTH = zeros(OUTARRSIZE,1);
    RUNTIMETH = zeros(OUTARRSIZE,1);
    LONGGTH = zeros(OUTARRSIZE,1);
    LONGJTH = zeros(OUTARRSIZE,1);
    POSEXITTH = zeros(OUTARRSIZE,1);
    NEGEXITTH = zeros(OUTARRSIZE,1);
    ELEVTH = zeros(OUTARRSIZE,1);
    DELRCTH = zeros(OUTARRSIZE,1);
    DPERPTH = zeros(OUTARRSIZE,1);
    VPERPTH = zeros(OUTARRSIZE,1);
    TIMETH = zeros(OUTARRSIZE,1);
    FCTH = zeros(OUTARRSIZE,1);
    FNOSETH = zeros(OUTARRSIZE,1);
    FMTH = zeros(OUTARRSIZE,1);
    PMMTH = zeros(OUTARRSIZE,1);
    TRTH = zeros(OUTARRSIZE,1);
    MNAEROTH = zeros(OUTARRSIZE,1);
    PSICTH = zeros(OUTARRSIZE,1);
    NCTH = zeros(OUTARRSIZE,1);
    PSICGVTH = zeros(OUTARRSIZE,1);
    THSTTH = zeros(OUTARRSIZE,1);
    PBPERTH = zeros(OUTARRSIZE,1);
    VWTH = zeros(OUTARRSIZE,1);
    VTWTH = zeros(OUTARRSIZE,1);
    PSIPTH = zeros(OUTARRSIZE,1);
    YAWTH = zeros(OUTARRSIZE,1);
    MUNOSETH = zeros(OUTARRSIZE,1);
    MUMAXTH = zeros(OUTARRSIZE,1);
```

```
MUMAXXTH = zeros(OUTARRSIZE,1);
MUMAXRTH = zeros(OUTARRSIZE,1);
MUMAXLTH = zeros(OUTARRSIZE,1);
MUMAXCTH = zeros(OUTARRSIZE,1);
MUCTH = zeros(OUTARRSIZE,1);
MURTH = zeros(OUTARRSIZE,1);
MULTH = zeros(OUTARRSIZE,1);
FMRTH = zeros(OUTARRSIZE,1);
FMLTH = zeros(OUTARRSIZE,1);
BDRAGCTH = zeros(OUTARRSIZE,1);
BDRAGRTH = zeros(OUTARRSIZE,1);
BDRAGLTH = zeros(OUTARRSIZE,1);
PMCTH = zeros(OUTARRSIZE,1);
PMRTH = zeros(OUTARRSIZE,1);
PMLTH = zeros(OUTARRSIZE,1);
YNAVTH = zeros(OUTARRSIZE,1);
XNAVTH = zeros(OUTARRSIZE,1);
DECELCOMTH = zeros(OUTARRSIZE,1);
DRAGCOMCTH = zeros(OUTARRSIZE,1);
RMPRATETH = zeros(OUTARRSIZE,1);
EABTH = zeros(OUTARRSIZE,1);
PBTH = zeros(OUTARRSIZE,1);
SIGCTH = zeros(OUTARRSIZE,1);
YSTRIPETH = zeros(OUTARRSIZE,1);
ZEROTH = zeros(OUTARRSIZE,1);
PBRTH = zeros(OUTARRSIZE,1);
PBLTH = zeros(OUTARRSIZE,1);
FCRTH = zeros(OUTARRSIZE,1);
FCLTH = zeros(OUTARRSIZE,1);
PSIMRTH = zeros(OUTARRSIZE,1);
PSIMLTH = zeros(OUTARRSIZE,1);
DRAGCOMLTH = zeros(OUTARRSIZE,1);
DRAGCOMRTH = zeros(OUTARRSIZE,1);
DRAGMAXLTH = zeros(OUTARRSIZE,1);
DRAGMAXRTH = zeros(OUTARRSIZE,1);
RADIUSTH = zeros(OUTARRSIZE,1);
TEMP1 = zeros(OUTARRSIZE,1);
TEMP2 = zeros(OUTARRSIZE,1);
TEMP4 = zeros(OUTARRSIZE,1);
TEMP5 = zeros(OUTARRSIZE,1);
TEMP6 = zeros(OUTARRSIZE,1);
TEMP3 = zeros(OUTARRSIZE,1);
```

GLOBALFILE.M is a file which declares as global so that they can be passed to a script file function in a common block rather than as a parameter and is called by RUNROTO once.

```
global SP
global SPM
global     WETMUT
global  WETMU
global  ICEMUT
global  ICEMU
global  SNOWMUT
global  SNOWMU
global  SLUSHMUT
global  SLUSHMU
global  FLOODMUT
global  FLOODMU
global  CONSTMUT
global  CONSTMU
global  CONST2MUT
global  CONST2MU
global PATCHNOW FIRSTPATCH LASTPATCH PATCHLEN PATCHTYPE PATCHOFTEN
global DT
global  TMGD TNGD USEMUABORT
global COMPUP DECLIM SST K2 RB OD HS DELS RP WS S MUROLL D1 K1 K2 DELB
global spiralexitx spiralexity spiralexitx22 spiralexity22 spiralexitr22
global TIMEOCCLAG FLATMU
global tmpfy111 tmpfy222 mulim11st mulim21st MUSKIDT MUSKID
global SW RWFC A B BC W
global HCG ASEFF VEAS VTW XSTART REVERSE AUTOREV
global CDFWD CDAFT CDFWD CGAFT CGFWD PCTMAC CGFWD
global THS1T THS1 THSRT THSR THSTIT THSTI THSNT THSN THSRNT THSRN fy111
fy222
global  psisk2 vskid2 murat2 mulim2 MD11 CG
global  psisk1 vskid1 murat1 mulim1 thstmulast BRKBUF DRATE DTIME WINGSPAN
```

FUNCPREDICTABORT.M is a MATLAB script function file called from RUNROTO.M once, containing the exit prediction algorithm. The parameters passed to it are variables predicted at touchdown. The following files are additional script file functions called by this algorithm.

```
function abortearly = funcpredictabort(upre,utpre,xnavpre,vwsspre,vexit,exitpos,nexit)
% these inputs are predicted values at touchdown
% in the simulation time=0 at maingear touchdown

global SW RWFC  W  ASEFF   REVERSE AUTOREV
global CDFWD CDAFT CGAFT CGFWD PCTMAC DECLIM TMGD TNGD
global USEMUABORT TIMEOCCLAG
global thstmulast BRKBUF DRATE DTIME MD11 CG

% initialize variables
    thstmulast=0;
   abortearly=0;
   decelpreon=0;
   runtimepre=1/(upre/xnavpre);
   lenfcms=250;
   disfcms=0;
   lenx   =50;
   futt=0;
   fudt=0;
   tempw  = utpre*1.689-upre;
   futut=utpre;
   deccnstu=upre;
   deccnstx=xnavpre;
   deccnstbd=exitpos-xnavpre;
   futu=upre;
   futut=(futu+tempw)/1.689;
   thstpre=functhrust(REVERSE,AUTOREV,futt,futut,xnavpre,exitpos,0.,futu,0.5);
    CDRAG=CDFWD+((CDAFT-CDFWD)/(CGAFT-CGFWD))*(PCTMAC-CGFWD);
   adragmu = (utpre^2)/295.37*CDRAG*SW;
   itemp=1+xnavpre/lenfcms;

% how does weight and cg error affect exit prediction
   wpre=W+0;
   cgpre=CG+0;
   if(MD11)
    if(wpre>480000)wpre=480000;end;
    if(wpre<340000)wpre=340000;end;
    if(cgpre>.34)cgpre=.34;end;
    if(cgpre<.12)cgpre=.12;end;
   else
    if(wpre>128000)wpre=128000;end;
    if(wpre<82000)wpre=82000;end;
    if(cgpre>.34)cgpre=.34;end;
    if(cgpre<-0.008)cgpre=-.008;end;
   end;
% CALCULATE NEW cgpre VARIABLES BASED ON FRACTION CG
  if(MD11)
     cginpre=1311.947+CG*295.779;
    apre=(cginpre-473.437)/12.;
    bpre=(1442-cginpre)/12.;
    bcpre=(1472.62-cginpre)/12.;
   hcgpre=(209.32-(sqrt(cginpre^2+(21)^2))*sin(atan(21/cginpre)+0.00193))/12.;
  else
```

```
     cginpre=885.547+CG*158.512;
     apre=(cginpre-97.998)/12.;
     bpre=(967.1-cginpre)/12.;
     bcpre=0;
    hcgpre=(83.029-(sqrt(cginpre^2+(5.1)^2))*sin(atan(-5.1/cginpre)+0.018))/12.;
   end;
   temp=apre+bpre;
   [mumax,mumaxx]=funcfc(RWFC(itemp),upre,xnavpre,temp);
% drag contributions due to crosswind
   dragcrs=abs(vwsspre)/57.3*4;
% drag contributions
   otherd=((+thstpre-adragmu)/wpre)*32.2+dragcrs;
   tempdlast=0;

   tempd= (vexit^2- futu^2)/(exitpos-xnavpre)/2.0 ;
   if(nexit>=3 | -tempd>7.0)
     DRATE=8.0;
   else
     DRATE=6.0;
   end;
   tempd=0;
   timeocclag=TIMEOCCLAG;
   timepreocclag=timeocclag;
   decelprerate=0;
   lastprexnav=xnavpre;
   lastpredeccalc=0;
% advance aircraft location to next friction measurement (every lenx feet)
   for i = 200:lenx :11450
 % start after touchdown point
   if(i > xnavpre & i < exitpos)
    futulast=futu;
  % delay decel cmd after touchdown
    if(-TMGD+futt < TNGD)
      tempd=0;
    end;
    if(-tempd>DECLIM)
      tempd=-9;
    end;
  % rate limit decel cmd
     if(abs(-tempd-tempdlast) > 1.33*fudt)
       if(-tempd-tempdlast > 0.)
         tempdset=tempdlast+1.33*fudt;
       else
         tempdset=tempdlast-1.33*fudt;
       end;
     else
       tempdset=-tempd;
     end;
     tempdlast=tempdset;
  % advance ground speed for travel distance using decel cmd, drag & thrust
     if(i-xnavpre    > lenx)
      futu=sqrt(futu*futu+2*(-tempdset+otherd)*(lenx));
     else
      futu=sqrt(futu*futu+2*(-tempdset+otherd)*(i-xnavpre));
     end;

 % calculate desired speed profile ground speed
    if(decelpreon)
```

```
      temp=deccnstu-(i+BRKBUF-deccnstx)*(deccnstu-vexit)/deccnstbd;
     else
       temp=0;
     end;
 % limit ground speed at or above speed profile ground speed
     if(temp < vexit)
       temp=vexit;
     end;
     if(futu <  temp)
       futu=temp;
     end;
 % advance time for travel distance using average of ground speed & last value
     if(i-xnavpre  >  lenx)
      fudt=lenx/(futulast+futu)*2;
     else
      fudt=(i-xnavpre )/(futulast+futu)*2;
     end;
      futt=futt+fudt;
 % this is how you would normally calculate time except that we limit
 % futu above to const vel line
 %       futt=futt+(futu-futulast)/(-tempdset);

 %  calculate airspeed, assumes winds are constant
     futut=(futu+tempw)/1.689;

 % calculate thrust using airspeed & whether braking is engaged
      temp=i;
    if(decelpreon)
       thstmu=functhrust(REVERSE,AUTOREV,futt,futut,temp,exitpos,1.,futu,0.5);
     else
       thstmu=functhrust(REVERSE,AUTOREV,futt,futut,temp,exitpos,0.,futu,0.5);
     end;
 % calculate aircraft drag along runway using airspeed
     adragmu = (futut^2)/295.37*CDRAG*SW;

 % calculate friction available from surface at main gear (mumaxx)
 % lenx stepsize may be smaller than friction measurement spacing
    if(disfcms<=0)
     disfcms=lenfcms;
     itemp=1+i/lenfcms;
      temp=i;
     temp2=apre+bpre;
    [mumax,mumaxx]=funcfc(RWFC(itemp),futu,temp,temp2);
    end;
   disfcms=disfcms-lenx;

 % calculate required deceleration
     tempd = ((vexit    )^2- futu^2)/(exitpos-i)/2.0;
     deceltemppre=-tempd;
 % calculate AVAILABLE friction fraction available at main gear using
 % aircraft parameters (gear loading & avg anti-skid eff)
     muavailarr=mumaxx*ASEFF*...
       (apre+hcgpre*(-adragmu/wpre+tempd/32.2))/(apre+(8*bpre+2*bcpre)/10.);
 % calculate NEEDED friction taking into account aircraft drag and thrust
 % *2 increases predict accuracy
    muneedarr=(-tempd/32.2-(-thstmu+adragmu)/wpre)*2;
    otherd=((+thstmu-adragmu)/wpre)*32.2+dragcrs;
```

```
  % engage braking logic
      temp=( (exitpos-i   )*(-tempd  -lastpredeccalc)/...
            ( i-lastprexnav)-tempd   );
      decelprerate = temp;
      lastprexnav= i;
      lastpredeccalc=-tempd;
      timepreocclast=timepreocclag;
      timepreocc=(exitpos-i)/((vexit+futu)/2.2)+futt+runtimepre;
% TIME FOR DECEL AT 7.5
       tempt= (futu-vexit)/7.5;
% DISTANCE OF DECEL AT 7.5
       tempdis=-(vexit^2- futu^2)/2.0/7.5;
% TIME REQUIRED
       temp=BRKBUF/vexit+(exitpos-i-BRKBUF-tempdis)/futu+tempt;
% TIME REMAINING
       dtemp=DTIME-futt-runtimepre;

     timepreocclag= timepreocc+(timepreocclag-timepreocc)*exp(-0.5);
   % set decel command to available friction if braking is engaged
      if((decelpreon) |...
        ((decelprerate>DRATE) |(timepreocc<DTIME)|(temp>dtemp)|...
       ((timepreocclag>timepreocclast)&(futt>6))))
       if(~decelpreon)
         deccnstu=futu;
         deccnstx=i;
         deccnstbd=exitpos-i;
        end;
       decelpreon=1;
       tempd=-muavailarr*32.2;
     else
       tempd=0;
     end;
    end;
  end;
% exit for loop if aircraft is past ROTO exit
% ABORT EXIT if NEEDED friction > AVAILABLE friction
 if(USEMUABORT)
  abortearly=muneedarr >  muavailarr;
%   [muneedarr,muavailarr,futu]
 end;
% ABORT EXIT if required deceleration > allowed deceleration
 if((~abortearly)&(deceltemppre>DECLIM))
  abortearly=1;
 end;
```

FUNCTHRUST.M is a script function calculating forward and reverse thrust and is called by FUNCPREDICTABORT.

```
function thst = functhrust(reverse,autorev,time,ut,xnav,exitpos,aurevmult,u,dttau)

  global THS1T THS1 THSRT THSR THSTIT THSTI THSNT THSN THSRNT THSRN
thstmulast

% auto-reverse thrust control logic
%    do not use auto-reverse thrust until brakes start
   if((~reverse)|((xnav+250)>exitpos)|(u<70*1.689))
     if (time<8.0)
       thst=funcinterp(ut,THS1T,THS1,length(THS1T));
     else
       thst=funcinterp(ut,THSNT,THSN,length(THSNT));
     end;
   else
%     reverse thrust for non-dry runway
     if (time<THSTIT(1))
       thst=funcinterp(time    ,THSTIT,THSTI,length(THSTIT));
% thrust arrays have to be set right for this to work,scale  spool up
       if(ut<THSRT(2))
         temp=funcinterp(ut ,THSRT,THSR,length(THSRT));
         temp=temp/THSR(2);
       else
         temp=1;
       end;
       thst=thst*temp;
     else
         thst=funcinterp(ut    ,THSRT,THSR,length(THSRT));
     end;
     thsttemp=funcinterp(ut,THSRNT,THSRN,length(THSRNT));
% add reverse idle limit here
     if((thst<thsttemp) & autorev & (time>5.))
       temp=    aurevmult*(thst-thsttemp)+thsttemp;
% engine lag, first 10 sec already lagged
       thst = temp+(thstmulast -temp)*exp(-dttau );
     end;
   end;
    thstmulast=thst;
```

FUNCFC.M is a script function which calculates surface friction and is called by FUNCPREDICTABORT.

```
function [mumax,mumaxx]=funcfc(wet,U,x,nlen)

 global SP
 global SPM
 global      WETMUT
 global  WETMU
 global  ICEMUT
 global  ICEMU
 global  SNOWMUT
 global  SNOWMU
 global  SLUSHMUT
 global  SLUSHMU
 global  FLOODMUT
 global  FLOODMU
 global  CONSTMUT
 global  CONSTMU
 global  CONST2MUT
 global  CONST2MU
 global PATCHNOW FIRSTPATCH LASTPATCH PATCHLEN PATCHTYPE PATCHOFTEN

munose=1;
mumain=1;
% get nose gear mu
% assume  PSIP=0
if(PATCHNOW&((x+nlen)>FIRSTPATCH)&((x+nlen)<LASTPATCH)& ...
   ( fix( fix((x+nlen)/PATCHLEN)/PATCHOFTEN)== ...
     fix((x+nlen)/PATCHLEN)/PATCHOFTEN))
   wetpatch=PATCHTYPE;
else
   wetpatch=wet;
end;
% dry runway
mumax = (0.93 - 0.0011* SP )*(1. - 0.0013*( U/1.689));
% wet runway
if(wetpatch==1)
   munose=funcinterp(U/1.689,WETMUT,WETMU,length(WETMUT));
% icy runway
else
 if(wetpatch==2)
   munose=funcinterp(U/1.689,ICEMUT,ICEMU,length(ICEMUT));
% snowy runway
 else
  if(wetpatch==3)
    munose=funcinterp(U/1.689,SNOWMUT,SNOWMU,length(SNOWMUT));
% slushy runway
  else
   if(wetpatch==4)
     munose=funcinterp(U/1.689,SLUSHMUT,SLUSHMU,length(SLUSHMUT));
% flooded runway
   else
    if(wetpatch==5)
      munose=funcinterp(U/1.689,FLOODMUT,FLOODMU,length(FLOODMUT));
% const  runway , read friction coefficient in from input directly
    else
     if(wetpatch==6)
       munose=funcinterp(U/1.689,CONSTMUT,CONSTMU,length(CONSTMUT));
```

```
            else
             if(wetpatch==7)
                munose=funcinterp(U/1.689,CONST2MUT,CONST2MU,length(CONST2MUT));
              end;
            end;
          end;
        end;
      end;
    end;
  end;
% get main gear mu
   if(PATCHNOW&((x)>FIRSTPATCH)&((x)<LASTPATCH)& ...
    ( fix( fix((x)/PATCHLEN)/PATCHOFTEN)== ...
              fix((x)/PATCHLEN)/PATCHOFTEN))
     wetpatch=PATCHTYPE;
  else
    wetpatch=wet;
  end;
% dry runway
  mumaxx= (0.93 - 0.0011* SPM)*(1. - 0.0013*( U/1.689));
  if(wetpatch==1)
    mumain=funcinterp(U/1.689,WETMUT,WETMU,length(WETMUT));
% icy runway
  else
   if(wetpatch==2)
     mumain=funcinterp(U/1.689,ICEMUT,ICEMU,length(ICEMUT));
% snowy runway
    else
     if(wetpatch==3)
       mumain=funcinterp(U/1.689,SNOWMUT,SNOWMU,length(SNOWMUT));
% slushy runway
      else
       if(wetpatch==4)
         mumain=funcinterp(U/1.689,SLUSHMUT,SLUSHMU,length(SLUSHMUT));
% flooded runway
        else
         if(wetpatch==5)
           mumain=funcinterp(U/1.689,FLOODMUT,FLOODMU,length(FLOODMUT));
% const  runway , read friction coefficient in from input directly
          else
           if(wetpatch==6)
             mumain=funcinterp(U/1.689,CONSTMUT,CONSTMU,length(CONSTMUT));
            else
             if(wetpatch==7)
               mumain=funcinterp(U/1.689,CONST2MUT,CONST2MU,length(CONST2MUT));
             end;
            end;
          end;
        end;
      end;
    end;
  end;
   if(wetpatch>5)
     mumaxx=mumain;
     mumax=munose;
   else
     mumaxx=mumaxx*mumain;
     mumax=mumax*munose;
```

```
end;
```

FUNCINTERPY.M is a script function returning the y value of the centerline and is called by FUNCPREDICTABORT.

```
function y=funcinterpy(x,exitpos)

    global spiralexitx spiralexity

    newx=x-exitpos;
    if (newx<0)
       y=0.;
     else
      if (newx>2200.688)
        y=600.;
      else
       if (newx<1362.1)
%         disp(' Point 1, funcinterpy'),keyboard
        y=funcinterp(newx,spiralexitx,spiralexity,length(spiralexitx));
       else
        if (newx<1800.517)
          y=-460.397+0.5*(newx-149.669)/0.8660254;
        else
          endcrv=2200;
          smrad=800;
          a=1.;
          b= 2.*200.;
        c=newx*newx-2.*endcrv*newx+endcrv*endcrv-smrad*smrad+200.*200.;
         y=(-b+sqrt(b*b-4.*a*c))/(2.*a);
        end;
       end;
      end;
     end;
```

FUNCINTERP.M is a script function interpolating vectors and is called by FUNCPREDICTABORT.

```
function y = funcinterp(x,xarray,yarray,npts)
if (xarray(1) < xarray(npts))
  for i=2:npts
   if (xarray(i) > x)|(i==npts)
   y = (x-xarray(i-1))/(xarray(i)-xarray(i-1))*(yarray(i)-yarray(i-1)) + yarray(i-1);
     break;
    end;
  end;
else
 for i=npts-1:-1:1
  if(xarray(i) > x)|(i==1)
  y=(x-xarray(i+1))/(xarray(i)-xarray(i+1))*(yarray(i)-yarray(i+1))+yarray(i+1);
    break;
   end;
 end;
end ;
```

FUNCPERPDIST.M is a script function returning the aircraft's perpendicular distance to the centerline and is called by FUNCPREDICTABORT. The XPATH input is already determined to be the X location of the centerline perpendicular to the aircraft.

```
function dist = funcperpdist(x,y,xpath,exitpos)

%     REAL*8 DIST


  ypath = funcinterpy(xpath,exitpos);
  if(abs(y-ypath)~=0)
    dist=sqrt((x-xpath)^2+(y-ypath)^2)*abs(y-ypath)/(y-ypath);
  else
    dist=abs(x-xpath);
  end;
```

SCRIPTWRITEPLOTDATA.M is a script file to create and write output variables from the simulation run and is called by RUNROTO. The following files are used by scriptwriteplotdata.

```
global fid

    i=TIME/INTVL+1;
    if(i>1000)i=1000;end;
    for TSTEP=1:i
      skip=((TSTEP-1)/DT*INTVL)+1;
% create variables that are created from others before
% creating others



% DISPLAY WHEN WING TIP EXITS RUNWAY SIDE, VARIABLE ON RUNWAY WIDTH=75
      clrdist=WINGSPAN*cos(PSITH(skip)/57.3)+75;
     if(abs(YTH(skip))>clrdist)
        RUNTIMETH(TSTEP)=RUNTIMETH(TSTEP -1);
     else
        RUNTIMETH(TSTEP)=RUNTIMETH(skip);
     end;
     RUNTIMETH(1)=1/((VEAS+VTW)*1.689/XSTART);
     if(LONGJTH(TSTEP)>0.15)LONGJTH(TSTEP)=0.15;end;
     if(LONGJTH(TSTEP)< -0.45)LONGJTH(TSTEP)= -0.45;end;

    if((XTH(skip)<800+EXITLOC(1))&(XTH(skip)>EXITLOC(1))&(XTH(skip)<EXITPOSTH(skip)))
      YSTRIPETH(TSTEP) =funcinterpy(XTH(skip)+EXITPOSTH(skip)-
EXITLOC(1),EXITPOSTH(skip));
    else

if((XTH(skip)<800+EXITLOC(2))&(XTH(skip)>EXITLOC(2))&(XTH(skip)<EXITPOSTH(skip)))
        YSTRIPETH(TSTEP) =funcinterpy(XTH(skip)+EXITPOSTH(skip)-
EXITLOC(2),EXITPOSTH(skip));
      else

if((XTH(skip)<800+EXITLOC(3))&(XTH(skip)>EXITLOC(3))&(XTH(skip)<EXITPOSTH(skip)))
          YSTRIPETH(TSTEP) =funcinterpy(XTH(skip)+EXITPOSTH(skip)-
EXITLOC(3),EXITPOSTH(skip));
        else
          YSTRIPETH(TSTEP) =funcinterpy(XTH(skip),EXITPOSTH(skip));
        end;
      end;
    end;
    TEMP=XTH(skip)-EXITPOSTH(skip);
    if(TEMP<0.0)
      POSEXITTH(TSTEP) = ((150-TREAD)/2. -LATBIAS);
    else
      if (TEMP<680   )
        POSEXITTH(TSTEP) = (((90+60*(680 -TEMP)/680 ) -TREAD)/2. -LATBIAS);
      else
        POSEXITTH(TSTEP) = ((90-TREAD)/2. -LATBIAS);
      end;
    end;
    NEGEXITTH(TSTEP) = -POSEXITTH(TSTEP) ;
    ZEROTH(TSTEP) =0;
```

```
XTH(TSTEP) =   XTH(skip);
PBPERTH(TSTEP) =(PBTH(skip) +PBRTH(skip) +PBLTH(skip) )/3./PSUP*100.;
MUMAXRTH(TSTEP) =MUMAXXTH(skip) ;
MUMAXLTH(TSTEP) = -MUMAXXTH(skip) ;
MUMAXCTH(TSTEP) =MUMAXXTH(skip) ;
MUCTH(TSTEP) =sqrt(FCTH(skip) ^2+BDRAGCTH(skip) ^2)/PMCTH(skip) ;
MURTH(TSTEP) =sqrt(FMRTH(skip) ^2+BDRAGRTH(skip) ^2)/PMRTH(skip) ;
MULTH(TSTEP) = -sqrt(FMLTH(skip) ^2+BDRAGLTH(skip) ^2)/PMLTH(skip) ;

TIMETH(TSTEP)=TIMETH(skip);
DXTH(TSTEP) =  DXTH(skip);
DUTH(TSTEP) =  DUTH(skip);
UTH(TSTEP) =  UTH(skip)/1.689;
YTH(TSTEP) =  YTH(skip);
DYTH(TSTEP) =  DYTH(skip);
DVTH(TSTEP) =  DVTH(skip);
PSITH(TSTEP) =  PSITH(skip);
DPSITH(TSTEP) =  DPSITH(skip);
DDPSITH(TSTEP) =  DDPSITH(skip);
SIGTH(TSTEP) = SIGTH(skip) ;
SIGSTRTH(TSTEP) =  SIGSTRTH(skip);
LATGTH(TSTEP) =  LATGTH(skip);
LATJTH(TSTEP) = LATJTH(skip) ;
DRUTH(TSTEP) =  DRUTH(skip);
DRURTH(TSTEP) = DRURTH(skip) ;
DRUDEADTH(TSTEP) = DRUDEADTH(skip) ;
DRMPUUTH(TSTEP) =  DRMPUUTH(skip);
DECELTH(TSTEP) =  DECELTH(skip);
NEXITTH(TSTEP) =  NEXITTH(skip);
VEXITTH(TSTEP) =  VEXITTH(skip);
RUNTIMETH(TSTEP) =  RUNTIMETH(skip);
LONGGTH(TSTEP) =  LONGGTH(skip);
LONGJTH(TSTEP) =  LONGJTH(skip);
ELEVTH(TSTEP) = ELEVTH(skip) ;
DELRCTH(TSTEP) = DELRCTH(skip) ;
DPERPTH(TSTEP) =  DPERPTH(skip);
VPERPTH(TSTEP) =  VPERPTH(skip);
FCTH(TSTEP) =  FCTH(skip);
FNOSETH(TSTEP) =  FNOSETH(skip);
FMTH(TSTEP) = FMTH(skip) ;
PMMTH(TSTEP) =  PMMTH(skip);
TRTH(TSTEP) =  TRTH(skip);
MNAEROTH(TSTEP) = MNAEROTH(skip) ;
PSICTH(TSTEP) =  PSICTH(skip);
NCTH(TSTEP) =  NC;
PSICGVTH(TSTEP) = PSICGVTH(skip) ;
THSTTH(TSTEP) =  THSTTH(skip);
VWTH(TSTEP) =  VWTH(skip);
VTWTH(TSTEP) =  VTWTH(skip);
PSIPTH(TSTEP) =  PSIPTH(skip);
YAWTH(TSTEP) =  YAWTH(skip);
MUNOSETH(TSTEP) =  MUNOSETH(skip);
MUMAXTH(TSTEP) = MUMAXTH(skip) ;
FMRTH(TSTEP) =  FMRTH(skip);
FMLTH(TSTEP) =  FMLTH(skip);
BDRAGCTH(TSTEP) =  BDRAGCTH(skip);
BDRAGRTH(TSTEP) =  BDRAGRTH(skip);
BDRAGLTH(TSTEP) =  BDRAGLTH(skip);
```

```
     PMCTH(TSTEP) = PMCTH(skip);
     PMRTH(TSTEP) = PMRTH(skip);
     PMLTH(TSTEP) = PMLTH(skip);
    YNAVTH(TSTEP)=YNAVTH(skip);
    XNAVTH(TSTEP) = XNAVTH(skip);
    DECELCOMTH(TSTEP) = DECELCOMTH(skip);
    DRAGCOMCTH(TSTEP) = DRAGCOMCTH(skip);
     RMPRATETH(TSTEP) = RMPRATETH(skip);
    EABTH(TSTEP) = EABTH(skip);
    PBTH(TSTEP) = PBTH(skip);
    SIGCTH(TSTEP) = SIGCTH(skip);
    PBRTH(TSTEP) = PBRTH(skip) ;
    PBLTH(TSTEP) = PBLTH(skip);
    RADIUSTH(TSTEP) = RADIUSTH(skip);
  end;
   TSTEP=i;

% PRINT OUT TIME HISTORY ARRAYS AND CROSS PLOT ARRAYS
   fprintf(fid,'        ********* OUTPUT DATA **************    \n');
   fprintf(fid,'        X  Y                         \n');
   funccrossplot(XTH,YTH,TSTEP);
   fprintf(fid,'        X  YSTRIPE                         \n');
   funccrossplot(XTH,YSTRIPETH,TSTEP);
   fprintf(fid,'        TIME   U    Y    PSI    YSTRIPE \n');
   functimhist(TIMETH,UTH,YTH,PSITH,YSTRIPETH,TSTEP);
   fprintf(fid,'        X     DV    U    PSI    YSTRIPE \n');
   functimhist(XTH,DVTH,UTH,PSITH,YSTRIPETH,TSTEP);
   fprintf(fid,'        TIME   DV   DY   DPSI   DU    \n');
   functimhist(TIMETH,DVTH,DYTH,DPSITH,DUTH,TSTEP);
   fprintf(fid,'        TIME   DV   SIGSTR DELRC   DPERP  \n');
   functimhist(TIMETH,DVTH,SIGSTRTH,DELRCTH,DPERPTH,TSTEP);
   fprintf(fid,'        TIME   DDPSI FC   FNOSE   FM    \n');
   functimhist(TIMETH,DDPSITH,FCTH,FNOSETH,FMTH,TSTEP);
   fprintf(fid,'        TIME   PMM  PSIP  THST    TR    \n');
   functimhist(TIMETH,PMMTH,PSIPTH,THSTTH,TRTH,TSTEP);
   fprintf(fid,'        TIME   MNAERO PSIC  NC     PSICGV \n');
   functimhist(TIMETH,MNAEROTH,PSICTH,NCTH,PSICGVTH,TSTEP);
   fprintf(fid,'        X MUNOSE   MUMAX    DPERP  VPERP  \n');
   functimhist(XTH,MUNOSETH,MUMAXTH,DPERPTH,VPERPTH,TSTEP);
   fprintf(fid,'        X  LATG   SIG   YAW    DV       \n');
   functimhist(XTH,LATGTH,SIGTH,YAWTH,DVTH,TSTEP);
   fprintf(fid,'        SIGSTR  FNOSE LATG  MUNOSE  DV\n');
   functimhist(SIGSTRTH,FNOSETH,LATGTH,MUNOSETH,DVTH,TSTEP);
   fprintf(fid,'        X   DRU   DRUR   DRUDEAD  DRMPUU   \n');
   functimhist(XTH,DRUTH,DRURTH,DRUDEADTH,DRMPUUTH,TSTEP);
   fprintf(fid,'        X   DECELCOM  NEXIT  RUNWAYTIME  LONGG   \n');
   functimhist(XTH,DECELCOMTH,NEXITTH,RUNTIMETH,LONGGTH,TSTEP);
   fprintf(fid,'        X  POSEXIT NEGEXIT  LATJ      LONGJ   \n');
   functimhist(XTH,POSEXITTH,NEGEXITTH,LATJTH,LONGJTH,TSTEP);
   fprintf(fid,'        XPERCENT BRK  THRUST  TAILWIND CROSSWIND\n');
   functimhist(XTH,PBPERTH,THSTTH,VTWTH,VWTH,TSTEP);
   fprintf(fid,'        X XNAV    YNAV    ELEV    CROSSWIND\n');
   functimhist(XTH,XNAVTH,YNAVTH,ELEVTH,VWTH,TSTEP);
   fprintf(fid,'        X  MUMAXR  MUR     MUMAXC   MUC    \n');
   functimhist(XTH,MUMAXRTH,MURTH,MUMAXCTH,MUCTH,TSTEP);
   fprintf(fid,'        X  MUMAXL  MUL     DX   ZERO      \n');
   functimhist(XTH,MUMAXLTH,MULTH,DXTH,ZEROTH,TSTEP);
  fclose(fid);
```

FUNCCROSSPLOT.M is a script function write format and is called by SCRIPTWRITEPLOTDATA.

```
function funccrossplot(para1,para2,tstep)

global fid



fprintf(fid,'PLOTDEF\n');
fprintf(fid,'XREAD(1)=1\n');
fprintf(fid,'YREAD(1)=2\n');
fprintf(fid,'END\n');
fprintf(fid,'DATA\n');
for i=1:tstep
fprintf(fid,'%13.6e %13.6e\n',para1(i),para2(i));
end;
fprintf(fid,'END\n');
```

FUNCTIMEHIST.M is a script function write format and is called by SCRIPTWRITEPLOTDATA.

```
function functimhist(tim,para1,para2,para3,para4,tstep)

global fid

fprintf(fid,'PLOTDEF\n');
fprintf(fid,'XREAD(1)=1\n');
fprintf(fid,'YREAD(1)=2\n');
fprintf(fid,'YREAD(2)=3\n');
fprintf(fid,'YREAD(3)=4\n');
fprintf(fid,'YREAD(4)=5\n');
fprintf(fid,'END\n');
fprintf(fid,'DATA\n');
for i=1: tstep
 fprintf(fid,'%13.6e %13.6e %13.6e %13.6e %13.6e\n',tim(i),para1(i),para2(i),para3(i),para4(i));
end ;
fprintf(fid,'END\n');
```

FUNCINPUT.C is C function file which is converted to a CMEX file and called by ROTO SIMULINK picture AERO in file MODELROTOBIG.M. It returns the aircraft lateral deviation from the centerline and aircraft heading. It uses CENTERLINE.M for spiral exit centerline coordinates (2 feet apart).

```
#include <math.h>
#include <stdio.h>
#include "mex.h"
 /* Input Arguments */

#define  UP_IN  prhs[0]

/* Output Arguments */

#define  YP_OUT        plhs[0]

#define inputvec 4


   double funcinterp(double x,double xarray[],double yarray[],int npts)
    {
     double y;
     int i;


     y=-1.0;

   if (xarray[0] < xarray[npts-1])
    {
     for (i=1;i<=npts-1;i++)
      {
       if ((xarray[i] > x) ||(i==npts-1))
        {
       y = (x-xarray[i-1])/(xarray[i]-xarray[i-1])*(yarray[i]-yarray[i-1]) + yarray[i-1];
         return y;
        };
      };
    }
    else
    {
     for (i=npts-2;i>=0;i--)
      {
       if((xarray[i] > x)||(i==0))
        {
       y=(x-xarray[i+1])/(xarray[i]-xarray[i+1])*(yarray[i]-yarray[i+1])+yarray[i+1];
         return y;
        };
      };
    }
   return y;
    }


   double funcinterpy(double x,double exitpos)
    {
     double y;
```

```
  double newx,endcrv,smrad,a,b,c;

 double *spiralexitx,*spiralexity;
 int lenspiralexit;

spiralexitx=mxGetPr(mexGetGlobal("spiralexitx"));
spiralexity=mxGetPr(mexGetGlobal("spiralexity"));
lenspiralexit=mxGetM(mexGetGlobal("spiralexitx"));


newx=x-exitpos;
 if (newx<0)
 {
   y=0.;
 }
 else
 {
  if (newx>2200.688)
   {
    y=600.;
   }
   else
   {
    if (newx<1362.1)
     {
     y=funcinterp(newx,spiralexitx,spiralexity,lenspiralexit);
     }
    else
     {
      if (newx<1800.517)
       {
        y=-460.397+0.5*(newx-149.669)/0.8660254;
       }
      else
       {
        endcrv=2200;
        smrad=800;
        a=1.;
        b= 2.*200.;
      c=newx*newx-2.*endcrv*newx+endcrv*endcrv-smrad*smrad+200.*200.;
       y=(-b+sqrt(b*b-4.*a*c))/(2.*a);
      };
    };
  };
 }

  return y;
}



double funcperpdist(double x,double y,double xpath,double exitpos)

{
   double dist,ypath;
```

```
  ypath = funcinterpy(xpath,exitpos);
  if(fabs(y-ypath)!=0)
   {
    dist=sqrt(pow((x-xpath),2)+pow((y-ypath),2))*fabs(y-ypath)/(y-ypath);
   }
  else
   {
    dist=fabs(x-xpath);
   }
  return dist;

 }




/*    function [dperp,vperp,psip] = funcinput(u)  */
  void funcinput(double  yout[], double u[])
  {
   double x,y,exitpos,psi;
   double  olddp;
   double  DT;
   double fpathx,dlat,far,near,mid,fardst,neardst,middst,dperp,vperp;
   double lastmiddst,xclose,yclose,xplus,yplus,psip,dyp,dxp;




   DT=*mxGetPr(mexGetGlobal("DT"));
/*    olddp=*mxGetPr(mexGetGlobal("olddp"));*/

   x=u[0];
   y=u[1];
   exitpos=u[2];
   psi=u[3];
/*    printf("%f %f %f %f\n",x,y,exitpos,psi);*/

/* AIRCRAFT DISPLACEMENT PERPENDICULAR TO PATH, CLOSEST POINT ON PATH */
   fpathx= funcinterpy(x,exitpos);
   dlat=fabs(y-fpathx);
   if(dlat < 1.0)
    {
     dlat=1.0;
    };
   far=(x+dlat);
   near=(x-dlat);
    mid=x;
   fardst=funcperpdist(x,y,far,exitpos);
   fardst=fabs(fardst);
   neardst=funcperpdist(x,y,near,exitpos);
   neardst=fabs(neardst);
   middst=funcperpdist(x,y,mid,exitpos);
   dperp=10000;
   lastmiddst=0.0;
   while (((fabs(fabs(middst)-fabs(dperp))) > 0.0000005) && (middst != lastmiddst))
    {
     dperp=middst;
```

```
        if (fardst-neardst < 0)
         {
          near=mid;
         neardst=fabs(middst);
         }
        else
         {
         far=mid;
        fardst=fabs(middst);
         } ;
       mid=(far+near)/2.0;
        lastmiddst=middst;
        middst = funcperpdist(x,y,mid,exitpos);
    };      /*% end of while loop*/

    dperp=middst;
    xclose=mid;
    yclose=funcinterpy(xclose,exitpos);

/*% AIRCRAFT VELOCITY PERPENDICULAR TO PATH
 cannot take derivatives in cmex file. file called more than once per iteration.
    vperp=(dperp-olddp)/DT;
    olddp=dperp;
    mxSetPr(mexGetGlobal("olddp"),&olddp);*/
/*% AIRCRAFT YAW ANGLE WITH RESPECT TO PATH*/
    dxp=.001;
    xplus=xclose+dxp;
    yplus=funcinterpy(xplus,exitpos);
    dyp=yplus-yclose;
    psip=psi-57.3*atan(dyp/dxp);


    yout[0]=dperp;
/*     yout[1]=vperp;*/
    yout[1]=psip;

}


void mexFunction(
        int             nlhs,
        Matrix  *plhs[],
        int             nrhs,
        Matrix  *prhs[]
        )
{

        double  *yp;
        double  *u;
        unsigned int    m,n;

        /* Check for proper number of arguments */

        if (nrhs !=1) {
          mexErrMsgTxt("Funcinput routine requires one vector input argument.");
        } else if (nlhs > 1) {
          mexErrMsgTxt("Funcinput routine produces one vector output argument.");
        }
```

```
    /* Check the dimensions of u. */

    m = mxGetM(UP_IN);
    n = mxGetN(UP_IN);

    if (!mxIsNumeric(UP_IN) || mxIsComplex(UP_IN) ||
    !mxIsFull(UP_IN)  || !mxIsDouble(UP_IN) ||
    (!(m == inputvec && n == 1) &&!(m == 1 && n == inputvec)) )
  {
      mexErrMsgTxt("Funcinput routine requires 4 input vector.");
    }

    /* Create a Matrix for the return argument */

    YP_OUT = mxCreateFull(1,2,  REAL);

    /* Assign pointers to the various parameters */

    yp = mxGetPr(YP_OUT);
    u = mxGetPr(UP_IN);

    /* Do the actual computations in a subroutine */

    funcinput(yp,u);

    return;
}
```

FUNCINPUTNAV.C is C function file which is converted to a CMEX file and called by ROTO SIMULINK picture CTRLLAWS20HZ in file MODELROTOBIG.M. It returns the aircraft lateral deviation from the centerline and aircraft heading. It uses CENTERLINE22.M for spiral exit centerline coordinates. (~100 feet apart)

```
#include <math.h>
#include <stdio.h>
#include "mex.h"
 /* Input Arguments */

#define  UP_IN  prhs[0]

/* Output Arguments */

#define  YP_OUT         plhs[0]

#define inputvec 4



    double funcinterpnav(double x,double xarray[],double yarray[],int npts)
     {
     double y;
     int i;


     y=-1.0;

    if (xarray[0] < xarray[npts-1])
     {
      for (i=1;i<=npts-1;i++)
       {
       if ((xarray[i] > x)||(i==npts-1))
        {
       y = (x-xarray[i-1])/(xarray[i]-xarray[i-1])*(yarray[i]-yarray[i-1]) + yarray[i-1];
         return y;
        };
      };
    }
    else
    {
     for (i=npts-2;i>=0;i--)
      {
      if((xarray[i] > x) ||(i==0))
       {
      y=(x-xarray[i+1])/(xarray[i]-xarray[i+1])*(yarray[i]-yarray[i+1])+yarray[i+1];
        return y;
       };
     };
    }
   return y;
    }


   double funcintrpynav(double x,double exitpos)
    {
```

```
  double y;
  double newx,endcrv,smrad,a,b,c;

 double* spiralexitx22,*spiralexity22;
 int lenspiralexit22;

spiralexitx22=mxGetPr(mexGetGlobal("spiralexitx22"));
spiralexity22=mxGetPr(mexGetGlobal("spiralexity22"));
lenspiralexit22=mxGetM(mexGetGlobal("spiralexitx22"));


newx=x-exitpos;
 if (newx<0)
 {
   y=0.;
 }
  else
  {
   if (newx>2200.688)
    {
     y=600.;
    }
   else
    {
     y=funcinterpnav(newx,spiralexitx22,spiralexity22,lenspiralexit22);
    };
   };


   return y;
 }



double funcperpdistnav(double x,double y,double xpath,double exitpos)

 {
     double dist,ypath;


  ypath = funcintrpynav(xpath,exitpos);
  if(fabs(y-ypath)!=0)
   {
    dist=sqrt(pow((x-xpath),2)+pow((y-ypath),2))*fabs(y-ypath)/(y-ypath);
   }
   else
   {
     dist=fabs(x-xpath);
   };
   return dist;

 }
```

```
/*    function [dperp,vperp,psip] = funcinputnav(u)  */
   void funcinputnav(double  yout[], double u[])
   {
    double x,y,cgx,cgy,exitpos,psi;
    double  olddp1,A,B;
    double  DT;
   double fpathx,dlat,far,near,mid,fardst,neardst,middst,dperp,vperp;
    double lastmiddst,xclose,yclose,xplus,yplus,psip,dyp,dxp;

   DT=*mxGetPr(mexGetGlobal("DT"));
/*   olddp1=*mxGetPr(mexGetGlobal("olddp1"));*/
   A=*mxGetPr(mexGetGlobal("A"));
   B=*mxGetPr(mexGetGlobal("B"));

   cgx=u[0];
   cgy=u[1];
   exitpos=u[2];
   psi=u[3];

   x=cgx+(A+B)*cos(psi/57.3);
   y=cgy+(A+B)*sin(psi/57.3);


/* AIRCRAFT DISPLACEMENT PERPENDICULAR TO PATH, CLOSEST POINT ON PATH */
   fpathx= funcintrpynav(x,exitpos);
   dlat=fabs(y-fpathx);
   if(dlat < 1.0)
    {
     dlat=1.0;
    };
   far=(x+dlat);
   near=(x-dlat);
    mid=x;
   fardst=funcperpdistnav(x,y,far,exitpos);
   fardst=fabs(fardst);
   neardst=funcperpdistnav(x,y,near,exitpos);
   neardst=fabs(neardst);
   middst=funcperpdistnav(x,y,mid,exitpos);
   dperp=10000;
   lastmiddst=0.0;
  while (((fabs(fabs(middst)-fabs(dperp))) > 0.0000005) && (middst != lastmiddst))
   {
     dperp=middst;
    if (fardst-neardst < 0)
     {
      near=mid;
      neardst=fabs(middst);
     }
    else
     {
     far=mid;
     fardst=fabs(middst);
     } ;
    mid=(far+near)/2.0;
     lastmiddst=middst;
    middst = funcperpdistnav(x,y,mid,exitpos);
  };      /*% end of while loop*/
```

```
      dperp=middst;
      xclose=mid;
     yclose=funcintrpynav(xclose,exitpos);

/*% AIRCRAFT VELOCITY PERPENDICULAR TO PATH
 DT would be COMPUP if it were running at 20 hz
 cannot create derivatives in mex files because it is called more than once per iteration
     vperp=(dperp-olddp1)/DT;
     olddp1=dperp;
     mxSetPr(mexGetGlobal("olddp1"),&olddp1);*/
/*% AIRCRAFT YAW ANGLE WITH RESPECT TO PATH*/
     dxp=.001;
     xplus=xclose+dxp;
     yplus=funcintrpynav(xplus,exitpos);
     dyp=yplus-yclose;
     psip=psi-57.3*atan(dyp/dxp);


     yout[0]=dperp;
/*     yout[1]=vperp;*/
     yout[1]=xclose;
}


void mexFunction(
          int                  nlhs,
          Matrix   *plhs[],
          int                  nrhs,
          Matrix   *prhs[]
          )
{

          double   *yp;
          double   *u;
          unsigned int      m,n;

          /* Check for proper number of arguments */

          if (nrhs !=1) {
            mexErrMsgTxt("Funcinputnav routine requires one vector input argument.");
          } else if (nlhs > 1) {
            mexErrMsgTxt("Funcinputnav routine produces one vector output argument.");
          }

          /* Check the dimensions of u. */

          m = mxGetM(UP_IN);
          n = mxGetN(UP_IN);

          if (!mxIsNumeric(UP_IN) || mxIsComplex(UP_IN) ||
          !mxIsFull(UP_IN)  || !mxIsDouble(UP_IN) ||
          (!(m == inputvec && n == 1) &&!(m == 1 && n == inputvec)) )
      {
             mexErrMsgTxt("Funcinputnav routine requires 4 input vector.");
          }

          /* Create a Matrix for the return argument */
```

```
        YP_OUT = mxCreateFull(1,2,  REAL);

        /* Assign pointers to the various parameters */

        yp = mxGetPr(YP_OUT);
        u = mxGetPr(UP_IN);

        /* Do the actual computations in a subroutine */

        funcinputnav(yp,u);

        return;
}
```

FUNCINTERPR.C is C function file which is converted to a CMEX file and called by ROTO SIMULINK picture CTRLLAWS20HZ in file MODELROTOBIG.M. It returns the centerline radius and uses CENTERLINE22.M. (~100 feet apart)

```
#include <math.h>
#include <stdio.h>
#include "mex.h"
 /* Input Arguments */

#define  UP_IN  prhs[0]

/* Output Arguments */

#define  YP_OUT          plhs[0]

#define inputvec 3



    double interpr(double x,double xarray[],double yarray[],int npts)
     {
     double y;
     int i;


     y=-1.0;

    if (xarray[0] > xarray[npts-1])
     {
      for (i=1;i<=npts-1;i++)
       {
        if ((xarray[i] < x)||(i==npts-1))
         {
/* This logrithmic interpolation technique tried to represent the spiral nature of the exit radius.
  Without it, the interpolated radius would provide more lead.*/
        y = -(pow(10,1-((x-xarray[i-1])/(xarray[i]-xarray[i-1])))-1.0)/9.0*(yarray[i]-yarray[i-1]) + yarray[i];
          return y;
         };
       };
     }
    else
     {
      for (i=npts-2;i>=0;i--)
       {
        if((xarray[i] < x) ||(i==0))
         {
        y=-(pow(10,1-((x-xarray[i+1])/(xarray[i]-xarray[i+1])))-1.0)/9.0*(yarray[i]-yarray[i+1])+yarray[i];
          return y;
         };
       };
     };
    return y;
    }
```

```
void funcinterpr(double  yout[], double u[])
 {

  double xclose, exitpos;

  double y;
  double newx;
  double x;

 double* spiralexitx22,*spiralexitr22;
 int lenspiralexit22;

spiralexitx22=mxGetPr(mexGetGlobal("spiralexitx22"));
spiralexitr22=mxGetPr(mexGetGlobal("spiralexitr22"));
lenspiralexit22=mxGetM(mexGetGlobal("spiralexitx22"));

xclose=u[0];
exitpos=u[1];
x=u[2];

newx=x-exitpos;
 if (newx<0)
 {
   y=1.0E+10;
 }
 else
 {
     y=interpr(xclose-exitpos,spiralexitx22,spiralexitr22,lenspiralexit22);
     if ((y>32766)&&(y<32768))
      {
       y=1.0E+10;
      };
  };

yout[0]=y;


}

void mexFunction(
        int                 nlhs,
        Matrix   *plhs[],
        int                 nrhs,
        Matrix   *prhs[]
        )
{

        double   *yp;
        double   *u;
        unsigned int       m,n;

        /* Check for proper number of arguments */

        if (nrhs !=1) {
         mexErrMsgTxt("Funcinterpr routine requires one vector input argument.");
        } else if (nlhs > 1) {
         mexErrMsgTxt("Funcinterpr routine produces one vector output argument.");
        }
```

```
    /* Check the dimensions of u. */

    m = mxGetM(UP_IN);
    n = mxGetN(UP_IN);

    if (!mxIsNumeric(UP_IN) || mxIsComplex(UP_IN) ||
    !mxIsFull(UP_IN)  || !mxIsDouble(UP_IN) ||
    (!(m == inputvec && n == 1) &&!(m == 1 && n == inputvec)) )
  {
      mexErrMsgTxt("Funcinterpr routine requires 2 input vector.");
    }

    /* Create a Matrix for the return argument */

    YP_OUT = mxCreateFull(1,1,  REAL);

    /* Assign pointers to the various parameters */

    yp = mxGetPr(YP_OUT);
    u = mxGetPr(UP_IN);

    /* Do the actual computations in a subroutine */

    funcinterpr(yp,u);

    return;
}
```

FUNCNOSETIREFORCES.C is C function file which is converted to a CMEX file and called by ROTO SIMULINK picture FUNCNOSETIREFORCES in file MODELROTOBIG.M. It returns ground moment GM and nose gear forces FNOSE and MUNOSE.

```
/*   function [fnose,gm,MUNOSE] = funcnosetireforces(u)*/
/*    function yout3 = funcnosetireforces(u)*/

#include <stdio.h>
#include <math.h>
#include "mex.h"
 /* Input Arguments */

#define  UP_IN   prhs[0]

/* Output Arguments */

#define  YP_OUT         plhs[0]

#define inputvec 13


    double funcinterp(double x,double xarray[],double yarray[],int npts)
     {
     double y;
     int i;


     y=-1.0;

    if (xarray[0] < xarray[npts-1])
     {
      for (i=1;i<=npts-1;i++)
      {
       if ((xarray[i] > x) ||(i==npts-1))
        {
       y = (x-xarray[i-1])/(xarray[i]-xarray[i-1])*(yarray[i]-yarray[i-1]) + yarray[i-1];
         return y;
        };
      };
     }
    else
     {
      for (i=npts-2;i>=0;i--)
      {
       if((xarray[i] > x)||(i==0))
        {
       y=(x-xarray[i+1])/(xarray[i]-xarray[i+1])*(yarray[i]-yarray[i+1])+yarray[i+1];
         return y;
        };
      };
     }
    return y;
    }

void  funcnosetireforces(double yout[], double u[])
{
```

```
int lenmuskidt;
double U,sig,psin,pn,deldef,mumax,tvert,term2;
double dell,delh,rl,rh,r111,r222,del111,del222,h111,h222,term1,n111,n222;
double phi111,phi222,q111,q222,fnose,MUNOSE,deldel,stcs,tyaw,troll,tilt,gm;

double DT, SST ,K2 ,RB, OD ,SP ,HS, DELS, RP ,WS, S, MUROLL ,FLATMU ,D1. K1.
DELB;
double tmpfy111, tmpfy222, mulim1lst, mulim2lst, fy111, fy222;
double  psisk2, vskid2 ,murat2 ,mulim2;
double  psisk1, vskid1, murat1 ,mulim1;
double* MUSKIDT,*MUSKID;

DT=*mxGetPr(mexGetGlobal("DT"));
SST=*mxGetPr(mexGetGlobal("SST"));
K2=*mxGetPr(mexGetGlobal("K2"));
RB=*mxGetPr(mexGetGlobal("RB"));
OD=*mxGetPr(mexGetGlobal("OD"));
SP=*mxGetPr(mexGetGlobal("SP"));
HS=*mxGetPr(mexGetGlobal("HS"));
DELS=*mxGetPr(mexGetGlobal("DELS"));
RP=*mxGetPr(mexGetGlobal("RP"));
WS=*mxGetPr(mexGetGlobal("WS"));
S=*mxGetPr(mexGetGlobal("S"));
MUROLL=*mxGetPr(mexGetGlobal("MUROLL"));
FLATMU=*mxGetPr(mexGetGlobal("FLATMU"));
D1=*mxGetPr(mexGetGlobal("D1"));
K1=*mxGetPr(mexGetGlobal("K1"));
DELB=*mxGetPr(mexGetGlobal("DELB"));
MUSKIDT=mxGetPr(mexGetGlobal("MUSKIDT"));
MUSKID=mxGetPr(mexGetGlobal("MUSKID"));
lenmuskidt=mxGetM(mexGetGlobal("MUSKIDT"));
/*not working, causes a memory problem
tmpfy111=*mxGetPr(mexGetGlobal("tmpfy111"));
tmpfy222=*mxGetPr(mexGetGlobal("tmpfy222"));
mulim1lst=*mxGetPr(mexGetGlobal("mulim1lst"));
mulim2lst=*mxGetPr(mexGetGlobal("mulim2lst"));
fy111=*mxGetPr(mexGetGlobal("fy111"));
fy222=*mxGetPr(mexGetGlobal("fy222"));
mulim2=*mxGetPr(mexGetGlobal("mulim2"));
mulim1=*mxGetPr(mexGetGlobal("mulim1"));*/

U=u[0];
sig=u[1];
mumax=u[2];
pn=u[3];
psin=u[4];

tmpfy111=u[5];
tmpfy222=u[6];
mulim1lst=u[7];
mulim2lst=u[8];
fy111=u[9];
fy222=u[10];
mulim2=u[11];
mulim1=u[12];

deldef = fabs(SST*sin(sig/57.3));
```

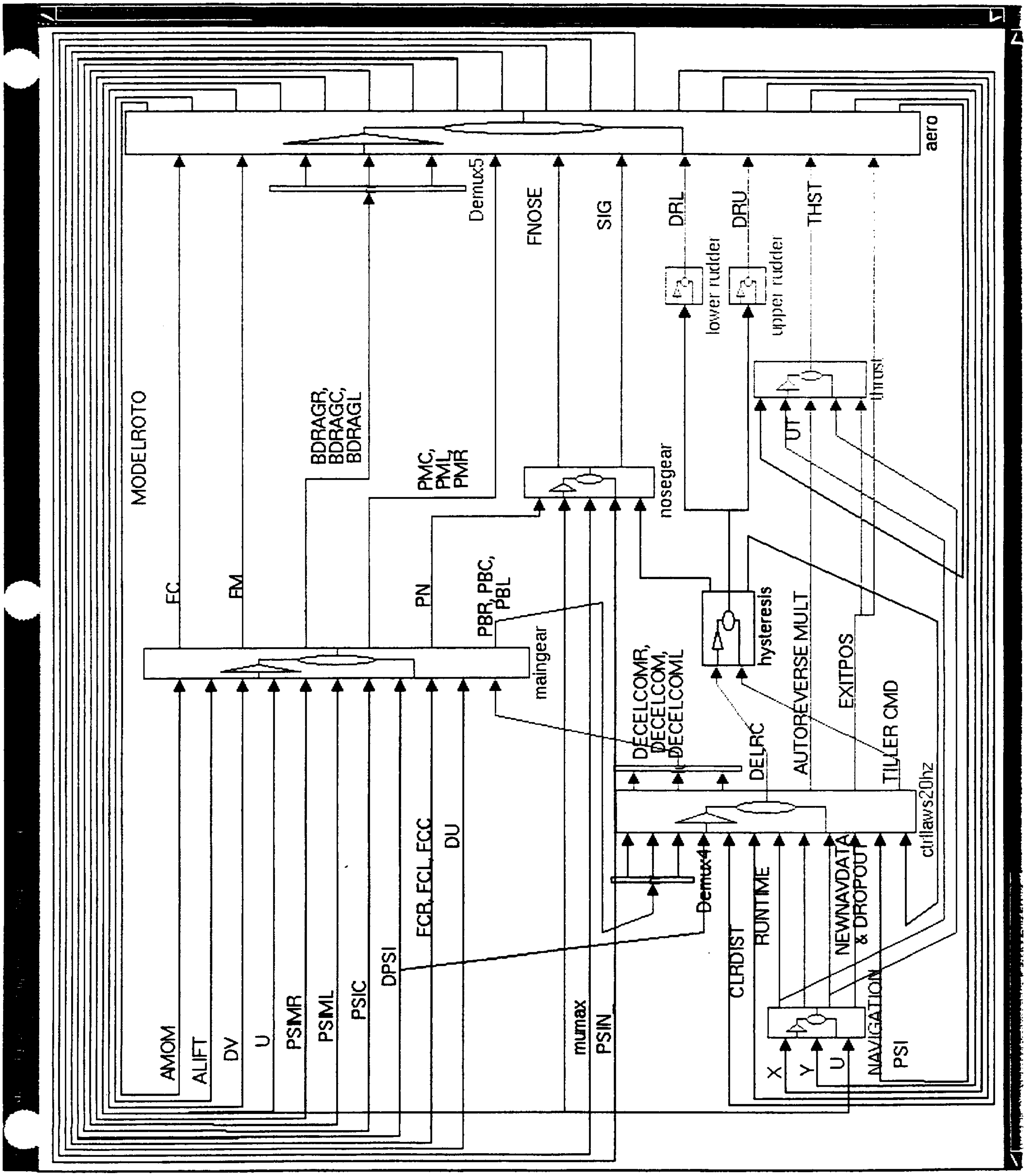
```
 /*% initially set loads and deflections equal to zero */
 /*% ('h' stands for high, 'l' stands for low.)*/
      dell = 0.0;
      delh = 0.0;
      rl = 0.0;
      rh = 0.0;
/*% check whether or not the nose is off the ground. */
      if(pn > 0.0)
      {
/*% assume all load is on one tire */
        rh = pn;
        delh = rh/K2;
        if(rh >= RB)
        {
         delh = rh/K1+D1;
        };
       if(delh > deldef)
        {
/*% assume both tires on high curve */
         dell = D1-0.5*(deldef-pn/K1);
         rl = K1*(dell-D1);
         rh = pn-rl;
         delh = dell+deldef;
         if(dell < DELB)
          {
           if(rh < RB)
            {
/*% both tires on low curve*/
             dell = 0.5*(pn/K2-deldef);
             rl = K2*dell;
             delh = dell+deldef;
             rh = pn-rl;
            }
           else
            {
/*% one tire on low and one on high */
             dell = (D1-deldef+pn/K1)/(1+K2/K1);
             rl = dell*K2;
             delh = dell+deldef;
             rh = pn-rl;
            }
          }
        }
      };
/*% assign r and del according to steering direction*/
    r111 = rl;
    r222 = rh;
    del111 = dell;
    del222 = delh;
    if(sig < 0.0)
    {
     r111 = rh;
     r222 = rl;
     del111 = delh;
     del222 = dell;
    };
    h111 = 0.85*OD*sqrt(del111/OD - (del111/OD)*(del111/OD));
    term1 = SP+1.4/HS/OD*(del111*del111-DELS*DELS)*(SP+15.) + 0.44*RP;
```

```
 if(del111/HS <= 0.338)
  {
   n111 =term1*(WS*WS/3.32*(del111/HS-1.84*(del111/HS)*(del111/HS)));
  }
 else
  {
    n111 = term1*(WS*WS/15.77*(1.-1.163*(del111/HS)));
  };
h222 = 0.85*OD*sqrt(del222/OD - (del222/OD)*(del222/OD));
 term2 = SP+1.4/HS/OD*(del222*del222-DELS*DELS)*(SP+15.) + 0.44*RP;
 if (del222/HS <= 0.338)
  {
   n222 =term2*(WS*WS/3.32*(del222/HS-1.84*(del222/HS)*(del222/HS)));
  }
 else
  {
   n222 = term2*(WS*WS/15.77*(1.-1.163*(del222/HS)));
  };
 if(n111 > 0.0)
  {
   psisk1 = psin - fy111/n111;
   vskid1 = fabs(U/1.689*(sin(psisk1/57.3)));
   murat1 = funcinterp(vskid1,MUSKIDT,MUSKID,lenmuskidt);
   if (FLATMU == 1)
    {
     murat1  = 1.0;
    };
   mulim1 = mumax*murat1;
  }
 if(n222 > 0.0)
  {
   psisk2 = psin  - fy222/n222;
   vskid2 = fabs(U/1.689*(sin(psisk2/57.3)));
   murat2 = funcinterp(vskid2,MUSKIDT,MUSKID,lenmuskidt);
   if (FLATMU == 1)
    {
     murat2  = 1.0;
    };
   mulim2 = mumax*murat2;
  };
 if (r111 > 0.0)
  {
   phi111 = n111*fabs(psin)/mulim1/r111;
   if(phi111 <= 0.1)
    {
     q111 = 0.8*h111/(1.0-.1482*phi111*phi111);
    }
   else
    {
     if  (phi111 <= 0.55)
      {
       q111 = h111*(phi111-phi111*phi111-0.01)/(phi111-0.1482*phi111*phi111*phi111);
      }
     else
      {
       if (phi111 <= 1.5)
        {
         q111 = h111*(0.2925-0.1*phi111)/(phi111-0.1482*phi111*phi111*phi111);
```

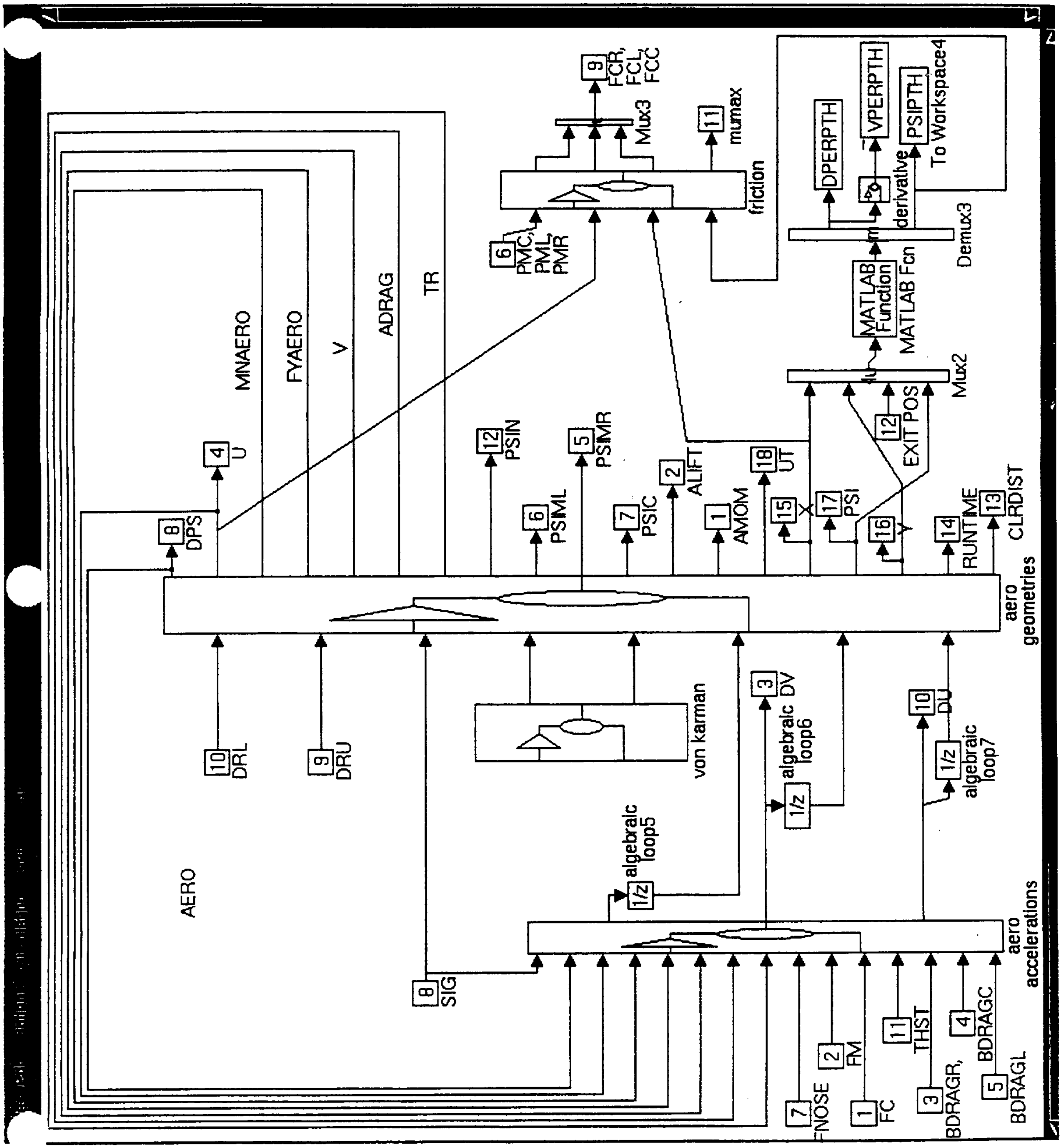
```
        }
        else
        {
        q111 = h111*(0.2925-0.1*phi111);
        };
      }
    }
  }
  else
  {
   phi111 = 0.0;
   q111 = 0.0;
  };
  fy111 =mulim1*r111*(phi111-0.1482*phi111*phi111*phi111);
  if(phi111 > 1.5)
  {
   fy111 = mulim1*r111;
  };
  if(psin < 0.0)
  {
  fy111 = -fy111;
  };
 if(r222 > 0.0)
  {
   phi222 = n222*fabs(psin)/mulim2/r222;
   if(phi222 <= 0.1)
   {
    q222 = 0.8*h222/(1.0-.1482*phi222*phi222);
   }
   else
   {
    if (phi222 <= 0.55)
    {
     q222 = h222*(phi222-phi222*phi222-0.01)/(phi222-0.1482*phi222*phi222*phi222);
    }
    else
    {
     if(phi222 <= 1.5)
     {
      q222 = h222*(0.2925-0.1*phi222)/(phi222-0.1482*phi222*phi222*phi222);
     }
     else
     {
      q222 = h222*(0.2925-0.1*phi222);
     };
    }
   }
 }
 else
 {
  phi222 = 0.0;
  q222 = 0.0;
 };
 fy222 =mulim2*r222*(phi222-0.1482*phi222*phi222*phi222);
 if (phi222 > 1.5)
 {
  fy222 =mulim2*r222;
 };
```

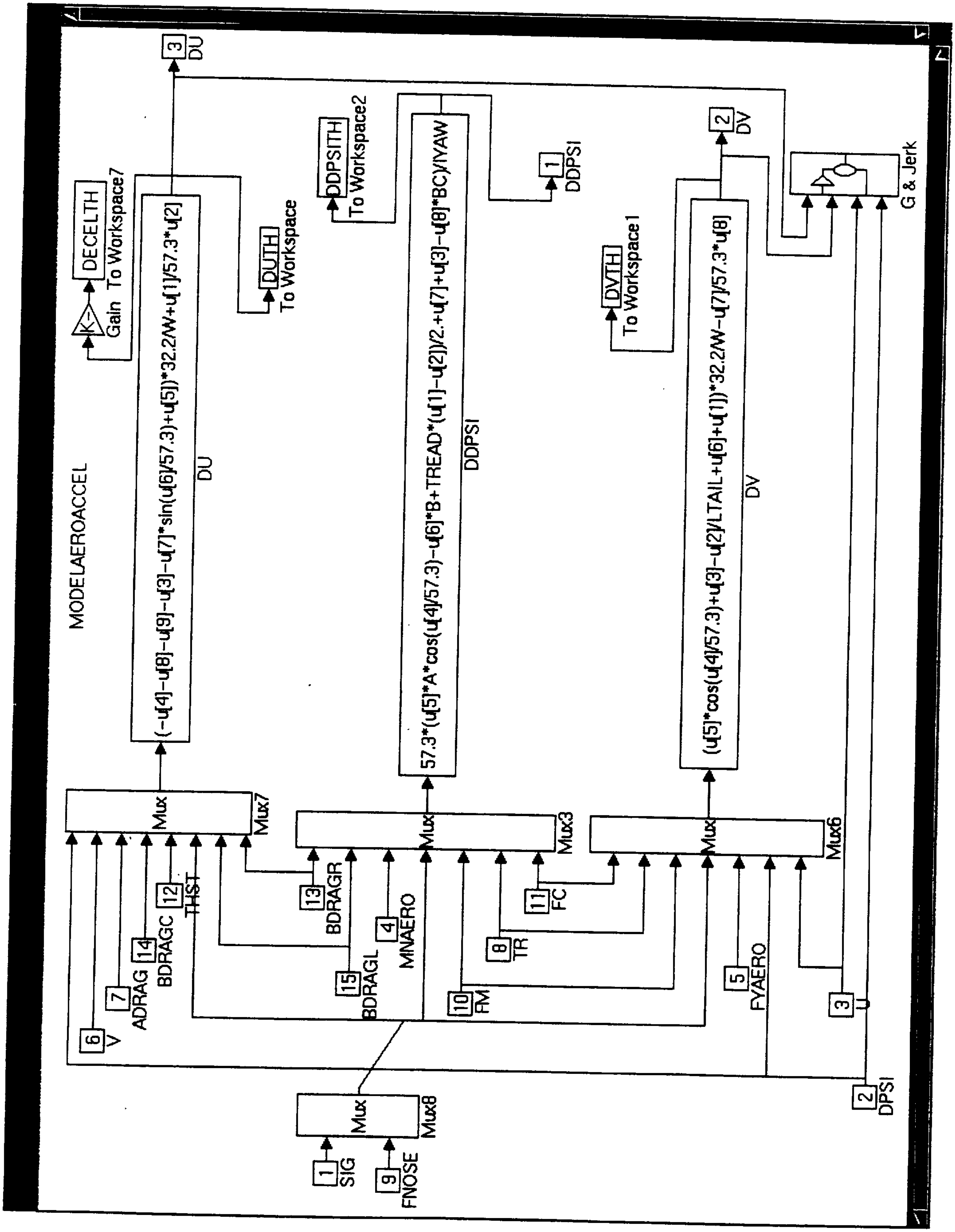
```
    if(psin < 0.0)
    {
     fy222 = -fy222;
    };
    if(U > 0)
    {
/*% relaxation lengtHS if mumax increasing*/
      if(mulim1*1.05>=mulim1lst)
      {
      tmpfy111=fy111+(tmpfy111-fy111)*exp(-DT/(3/U));
       fy111=tmpfy111;
      }
      else
      {
       tmpfy111=fy111;
      };
      mulim1lst=mulim1;
      if(mulim2*1.05>=mulim2lst)
      {
      tmpfy222=fy222+(tmpfy222-fy222)*exp(-DT/(3/U));
       fy222=tmpfy222;
      }
      else
      {
       tmpfy222=fy222;
      };
      mulim2lst=mulim2;
    };
    fnose=fy111+fy222;
    MUNOSE=fnose/pn;
/*% GROUND MOMENT ON NOSEWHEEL STRUT */
     deldel = SST*sin(sig/57.3);
     stcs = SST/S*cos(sig/57.3);
    tyaw = fy111*((OD/2.-del111)*stcs + q111) + fy222*((OD/2.-del222)*stcs + q222);
    troll = MUROLL*(r111-r222)*S;
     tilt = asin(deldel/S);
    tvert = r222*stcs*(0.5*S-(0.5*OD-del222)*tan(tilt))-r111*stcs*(0.5*S+(0.5*OD-del111)*tan(tilt));
     gm = tyaw + troll + tvert;

  /*for some reason this is not working, causing memory problems

  mxSetPr(mexGetGlobal("tmpfy111"),&tmpfy111);
    mxSetPr(mexGetGlobal("tmpfy222"),&tmpfy222);
     mxSetPr(mexGetGlobal("mulim1lst"),&mulim1lst);
     mxSetPr(mexGetGlobal("mulim2lst"),&mulim2lst);
    mxSetPr(mexGetGlobal("fy111"),&fy111);
    mxSetPr(mexGetGlobal("fy222"),&fy222);
     mxSetPr(mexGetGlobal("mulim2"),&mulim2);
     mxSetPr(mexGetGlobal("mulim1"),&mulim1);*/


    yout[0]=fnose;
     yout[1]=gm;
     yout[2]=MUNOSE;
  yout[3]=tmpfy111;
    yout[4]=tmpfy222;
     yout[5]=mulim1lst;
     yout[6]=mulim2lst;
```

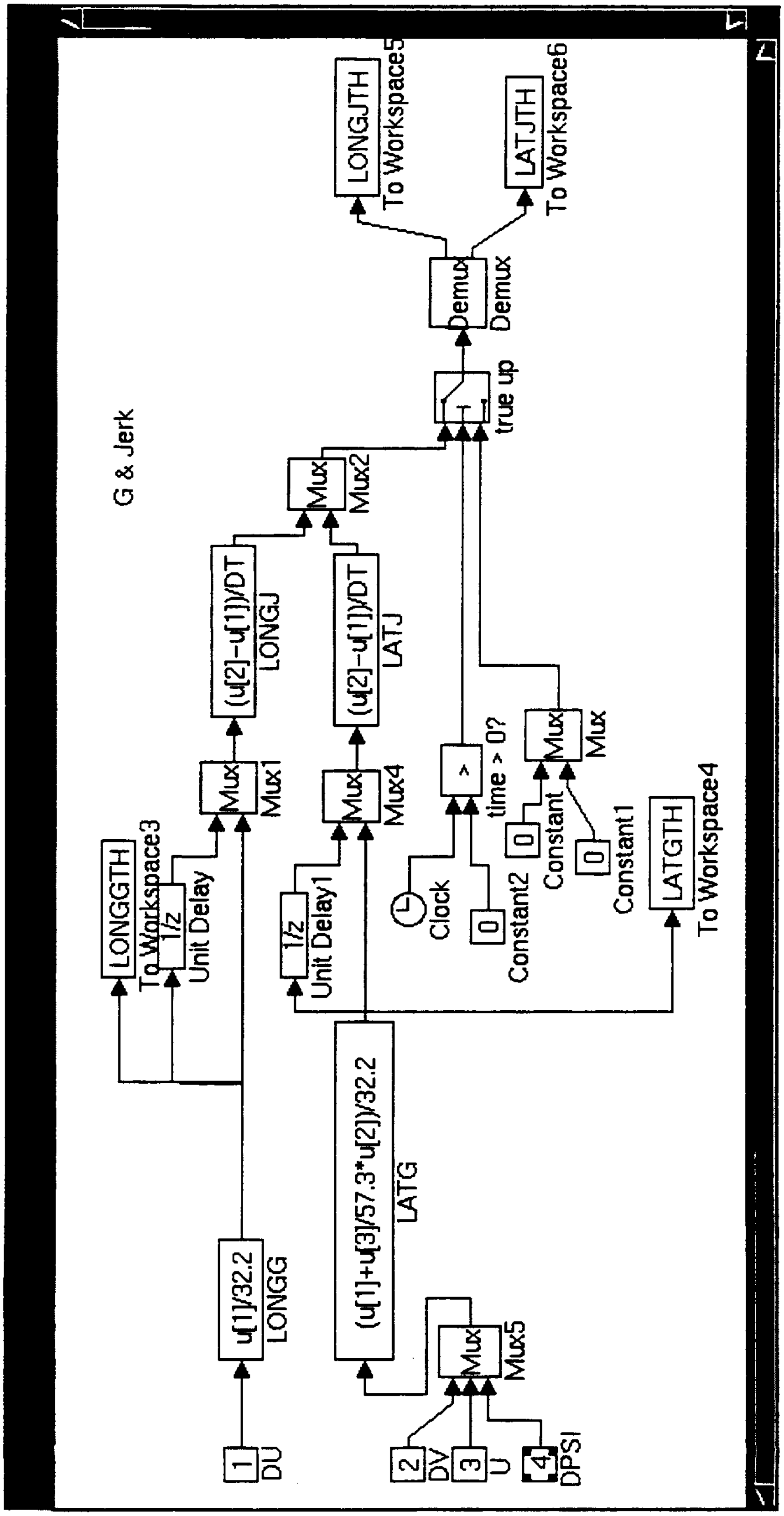
```
    yout[7]=fy111;
    yout[8]=fy222;
    yout[9]=mulim2;
    yout[10]=mulim1;

  }

void mexFunction(
        int             nlhs,
        Matrix   *plhs[],
        int             nrhs,
        Matrix   *prhs[]
        )
{

        double   *yp;
        double   *u;
        unsigned int      m,n;

        /* Check for proper number of arguments */

        if (nrhs !=1) {
          mexErrMsgTxt("Funcnosetireforces routine requires one vector input argument.");
        } else if (nlhs > 1) {
          mexErrMsgTxt("Funcnosetireforces routine produces one vector output argument.");
        }

        /* Check the dimensions of u. */

        m = mxGetM(UP_IN);
        n = mxGetN(UP_IN);

        if (!mxIsNumeric(UP_IN) || mxIsComplex(UP_IN) ||
        !mxIsFull(UP_IN)  || !mxIsDouble(UP_IN) ||
        (!(m == inputvec && n == 1) &&!(m == 1 && n == inputvec)))
    {
          mexErrMsgTxt("Funcnosetireforces routine requires a 13 input vector.");
        }

        /* Create a Matrix for the return argument */

        YP_OUT = mxCreateFull(1,11, REAL);

        /* Assign pointers to the various parameters */

        yp = mxGetPr(YP_OUT);
        u = mxGetPr(UP_IN);

        /* Do the actual computations in a subroutine */

        funcnosetireforces(yp,u);

        return;
}
```

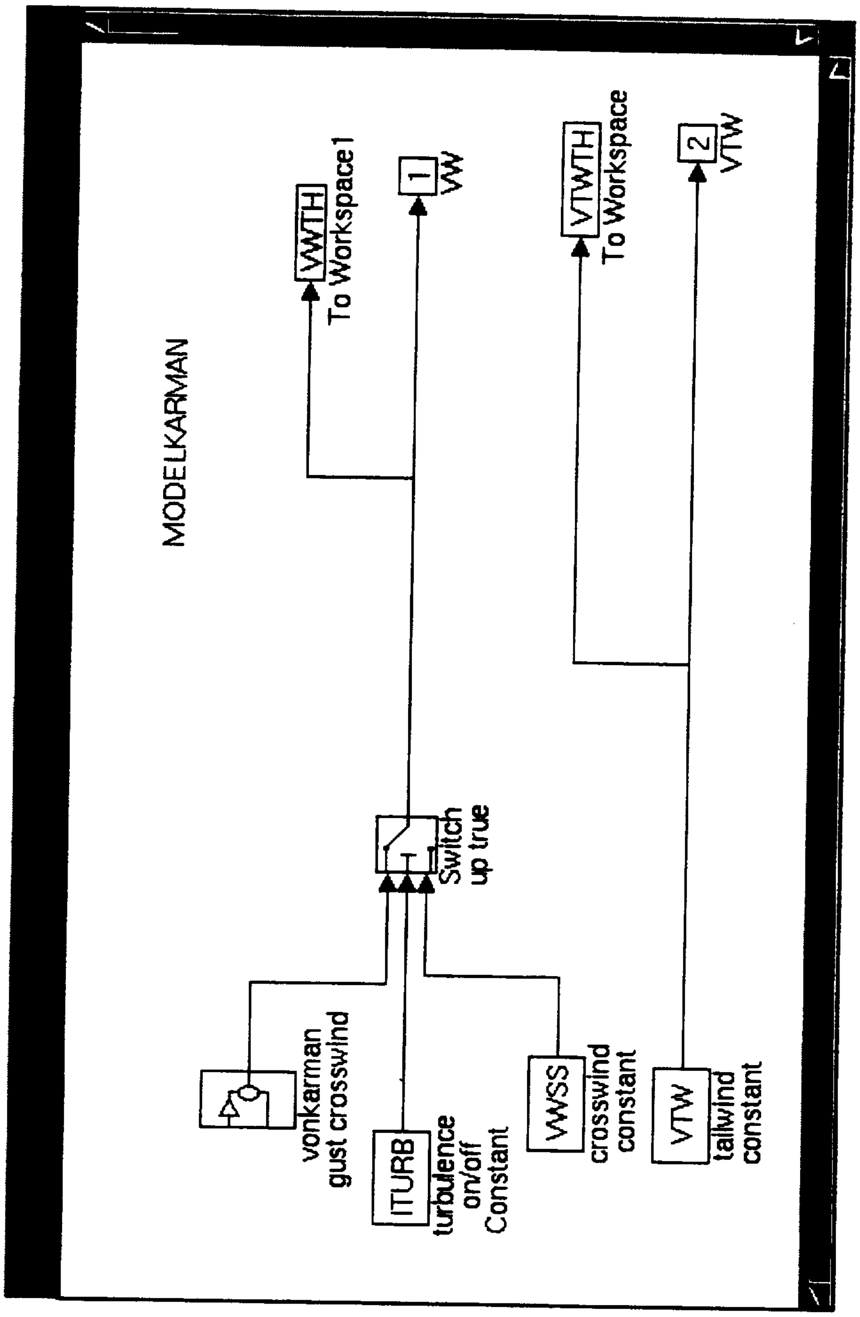


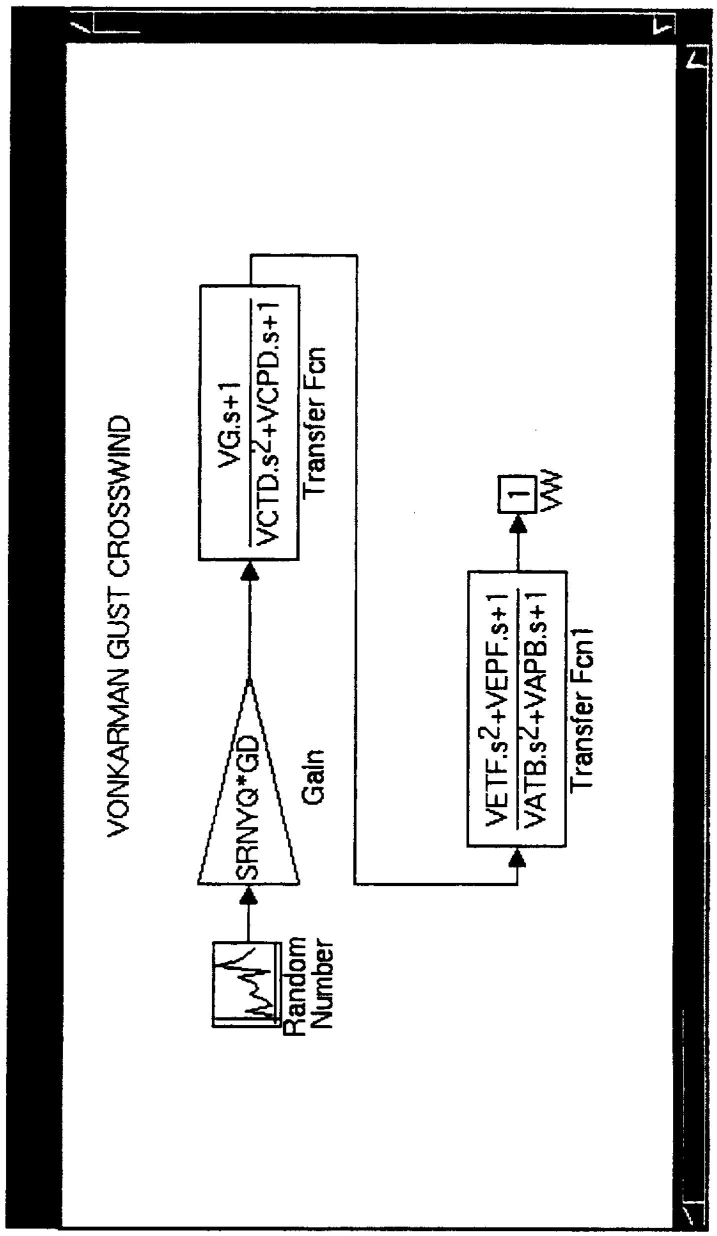
MODELROTO
FC
FM
BDRAGR,
BDRAGC,
BDRAGL
PN
PMC,
PML,
PMR
PBR, PBC,
PBL
maingear
Demux5
FNOSE
SIG
DRL
DRU
THST
aero
lower rudder
upper rudder
thrust
UT
nosegear
hysteresis
DECELCOMR,
DECELCOM,
DECELCOML
DELRC
AUTOREVERSE MULT
EXITPOS
TILLER CMD
ctrllaws20hz
AMOM
ALIFT
DV
U
PSIMR
PSIML
PSIC
DPSI
ECR, ECL, ECC
DU
mumax
PSIN
Demux4
CLRDIST
RUNTIME
NEWNAVDATA
& DROPOUT
NAVIGATION
X
Y
U
PSI

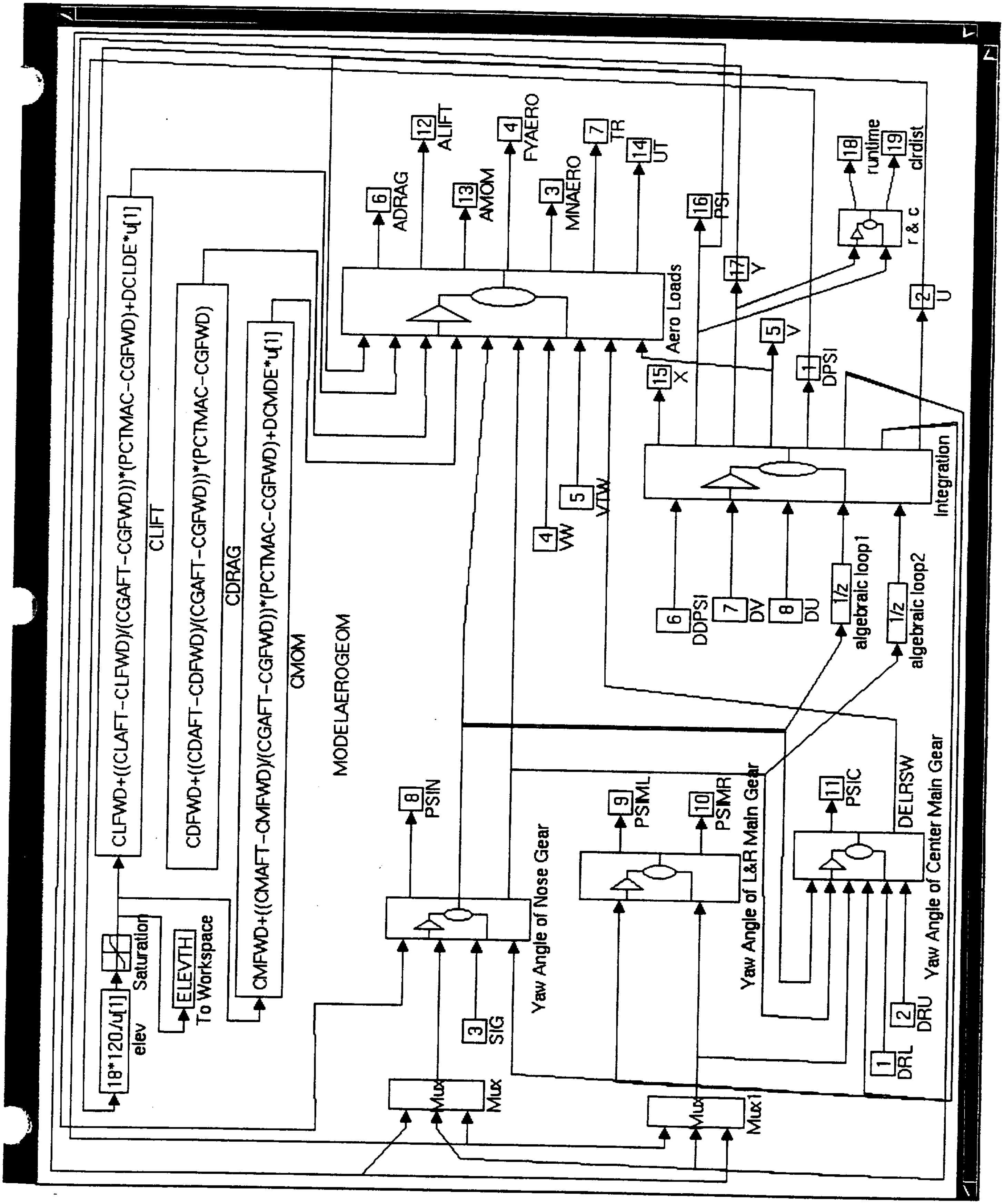

AERO
aero accelerations
aero geometries
von karman
friction
algebraic loop5
algebraic loop6
algebraic loop7
1/z
MATLAB Function
MATLAB Fcn
Mux2
Mux3
Demux3
derivative
DPERPTH
VPERPTH
PSIPTH
To Workspace4
mumax
FCR, FCL, FCC
PMC, PML, PMR
MNAERO
FYAERO
V
ADRAG
TR
DPS
U
PSIN
PSIMR
PSIML
PSIC
ALIFT
AMOM
UT
PSI
EXIT POS
RUNTIME
CLRDIST
DRL
DRU
DV
DU
SIG
FNOSE
FC
FM
THST
BDRAGR,
BDRAGC
BDRAGL


MODELAEROACCEL
(-u[4]-u[8]-u[9]-u[3]-u[7]*sin(u[6]/57.3)+u[5])*32.2/W+u[1]/57.3*u[2]
DU
57.3*(u[5]*A*cos(u[4]/57.3)-u[6]*B+TREAD*(u[1]-u[2])/2.+u[7]+u[3]-u[8]*BC)/IYAW
DDPSI
(u[5]*cos(u[4]/57.3)+u[3]-u[2]/LTAIL+u[6]+u[1])*32.2/W-u[7]/57.3*u[8]
DV
Gain
K-
DECELTH
To Workspace7
DUTH
To Workspace
DDPSITH
To Workspace2
DVTH
To Workspace1
3
DU
1
DDPSI
2
DV
G & Jerk
Mux
Mux7
Mux
Mux3
Mux
Mux6
Mux
Mux8
6
V
7
ADRAG
14
BDRAGC
12
THST
13
BDRAGR
15
BDRAGL
4
MNAERO
10
FM
8
TR
11
FC
5
FYAERO
3
U
2
DPSI
1
SIG
9
FNOSE

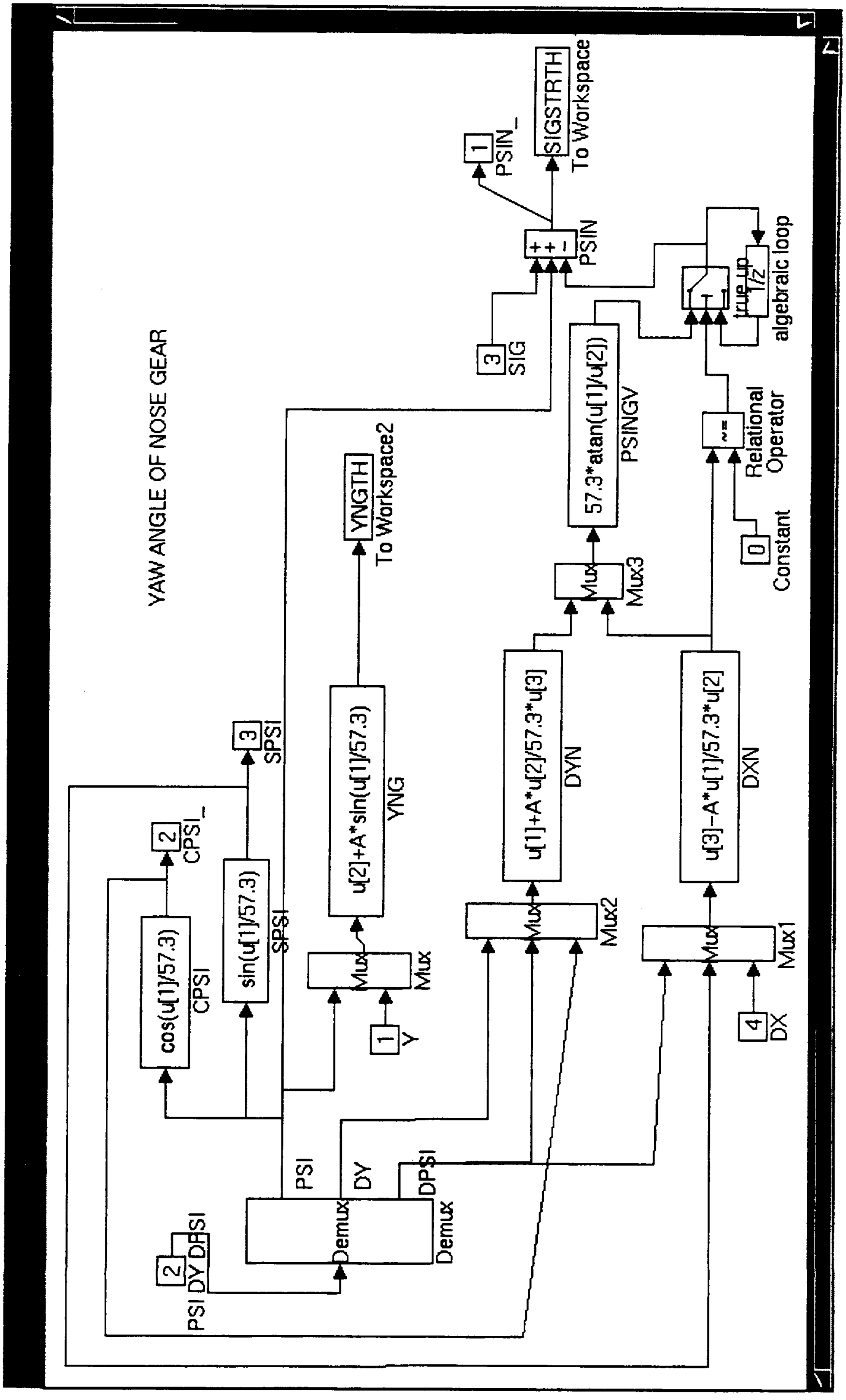

G & Jerk
1
DU
2
DV
3
U
4
DPSI
Mux
Mux5
u[1]/32.2
LONGG
(u[1]+u[3]/57.3*u[2])/32.2
LATG
LONGGTH
To Workspace3
1/z
Unit Delay
1/z
Unit Delay1
Mux
Mux1
Mux
Mux4
(u[2]-u[1])/DT
LONGJ
(u[2]-u[1])/DT
LATJ
Clock
0
Constant2
>
time > 0?
0
Constant
0
Constant1
Mux
Mux
LATGTH
To Workspace4
Mux
Mux2
true up
Demux
Demux
LONGJTH
To Workspace5
LATJTH
To Workspace6


MODELKARMAN
vonkarman gust crosswind
ITURB
turbulence on/off Constant
VWSS
crosswind constant
VTW
tailwind constant
Switch up true
VWTH
To Workspace1
1
VW
VTWTH
To Workspace
2
VTW

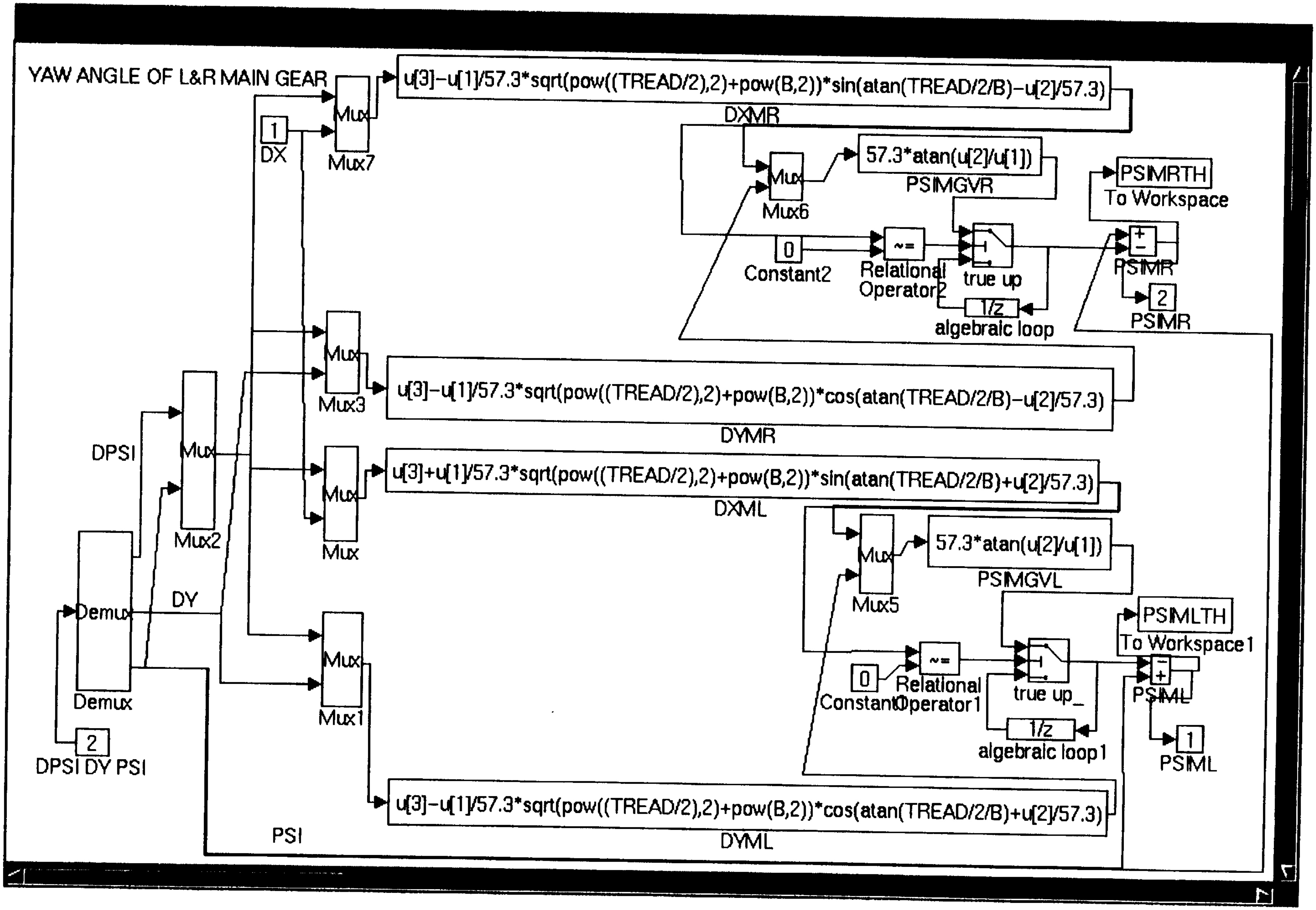

VONKARMAN GUST CROSSWIND
Random Number
SRNYQ*GD
Gain
VG.s+1
VCTD.s2+VCPD.s+1
Transfer Fcn
VETF.s2+VEPF.s+1
VATB.s2+VAPB.s+1
Transfer Fcn1
1
VW

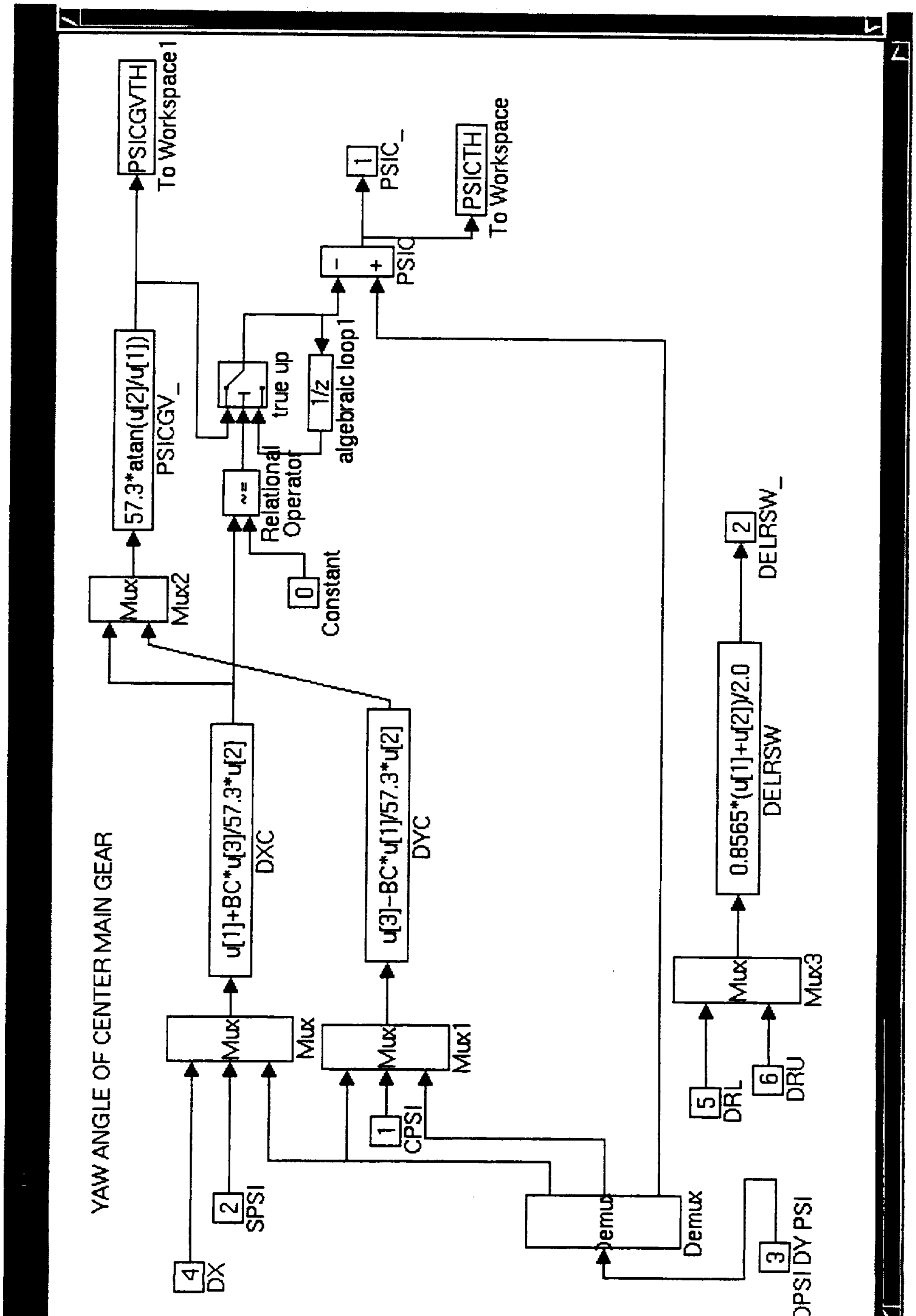


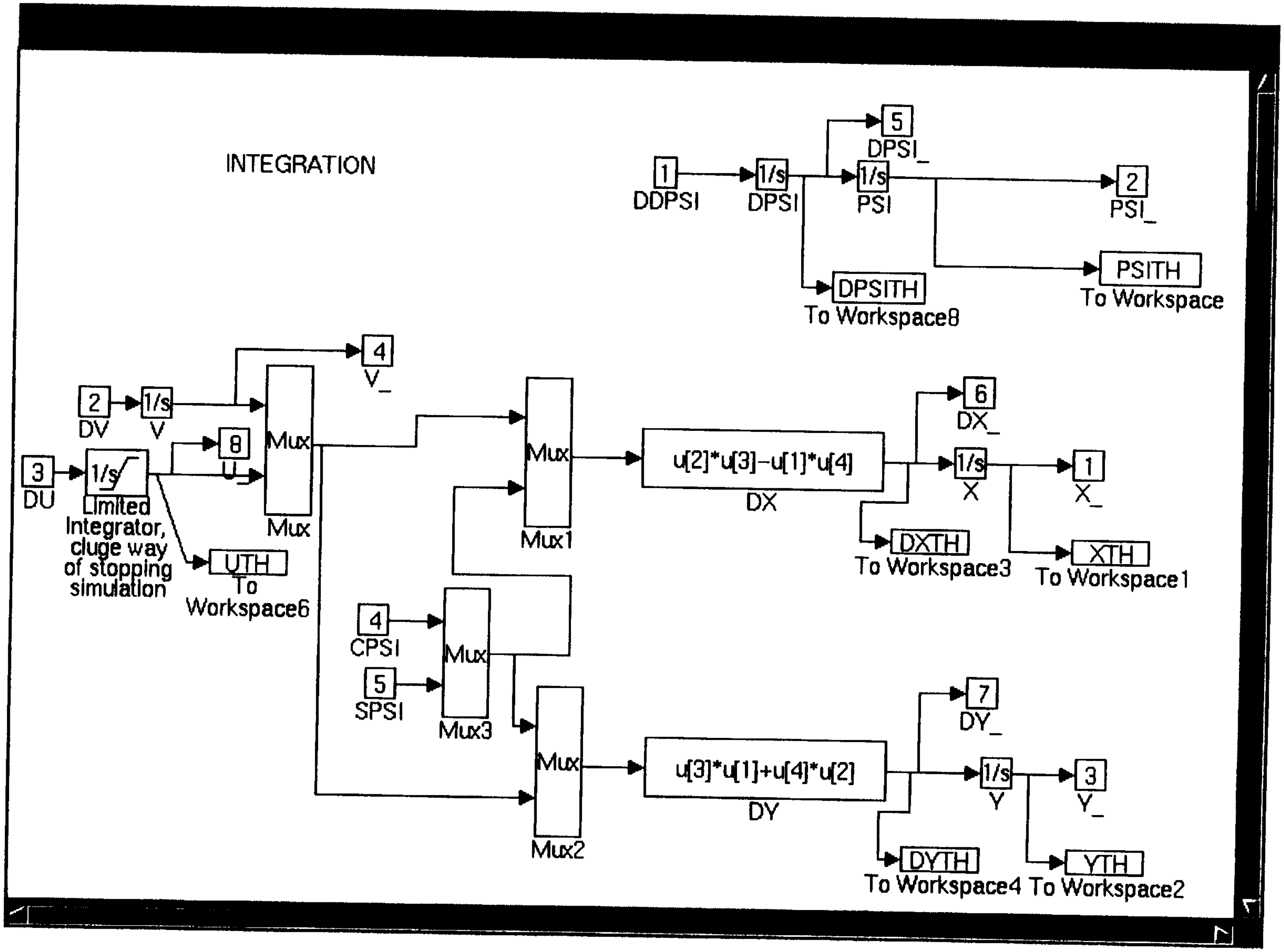
18*120/u[1]
elev
Saturation
ELEVTH
To Workspace
CLFWD+((CLAFT-CLFWD)/(CGAFT-CGFWD))*(PCTMAC-CGFWD)+DCLDE*u[1]
CLIFT
CDFWD+((CDAFT-CDFWD)/(CGAFT-CGFWD))*(PCTMAC-CGFWD)
CDRAG
CMFWD+((CMAFT-CMFWD)/(CGAFT-CGFWD))*(PCTMAC-CGFWD)+DCMDE*u[1]
CMOM
MODELAEROGEOM
6 ADRAG
12 ALIFT
13 AMOM
4 FYAERO
3 MNAERO
7 TR
14 UT
Aero Loads
4 VW
5 VTW
16 PSI
17 Y
5 V
15 X
1 DPSI
2 U
18 runtime
19 clrdist
r & c
Integration
6 DDPSI
7 DV
8 DU
1/z
algebraic loop1
1/z
algebraic loop2
8 PSIN
Yaw Angle of Nose Gear
3 SIG
Mux
Mux
9 PSIML
10 PSIMR
Yaw Angle of L&R Main Gear
Mux
Mux1
11 PSIC
DELRSW
Yaw Angle of Center Main Gear
2 DRU
1 DRL

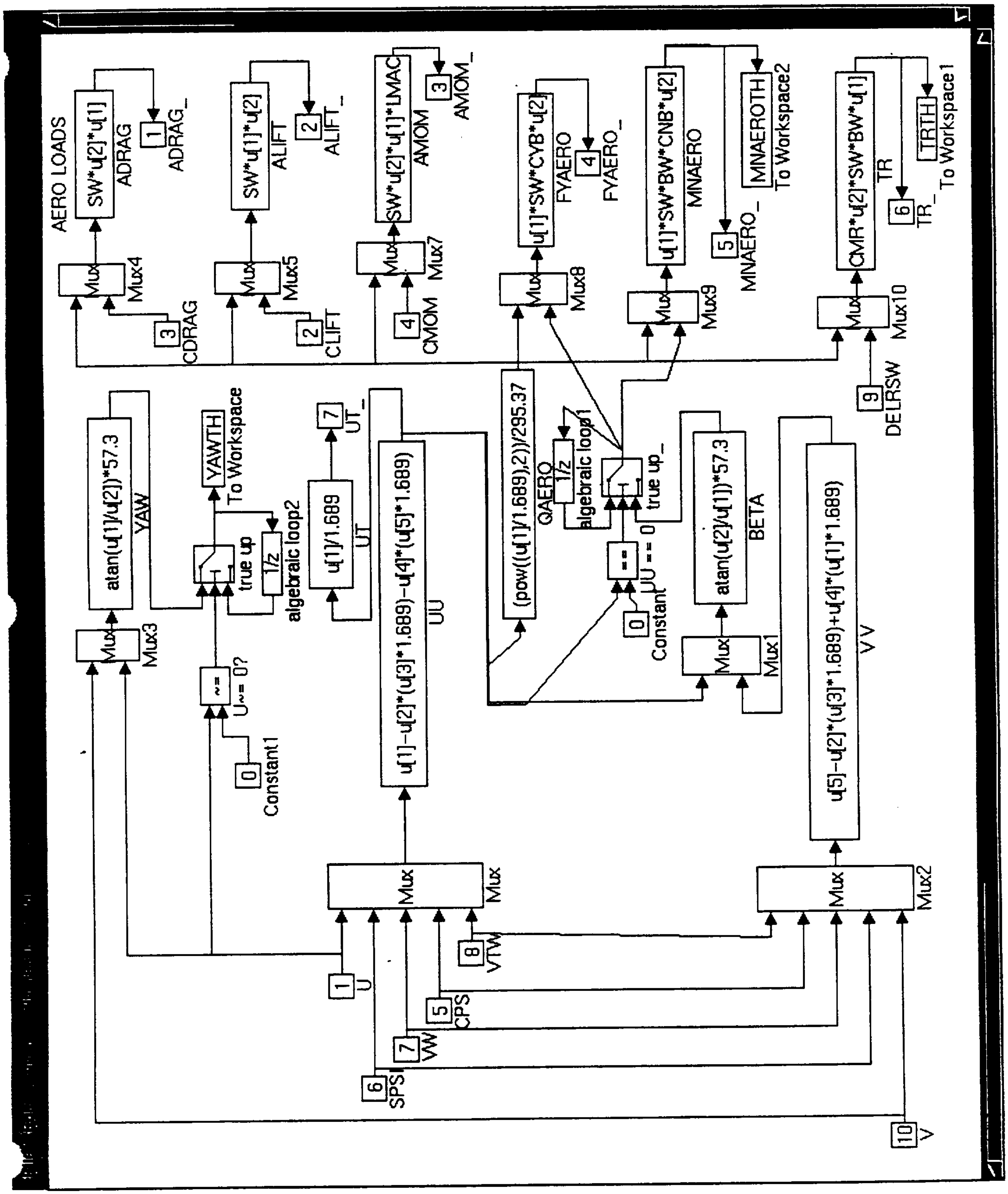


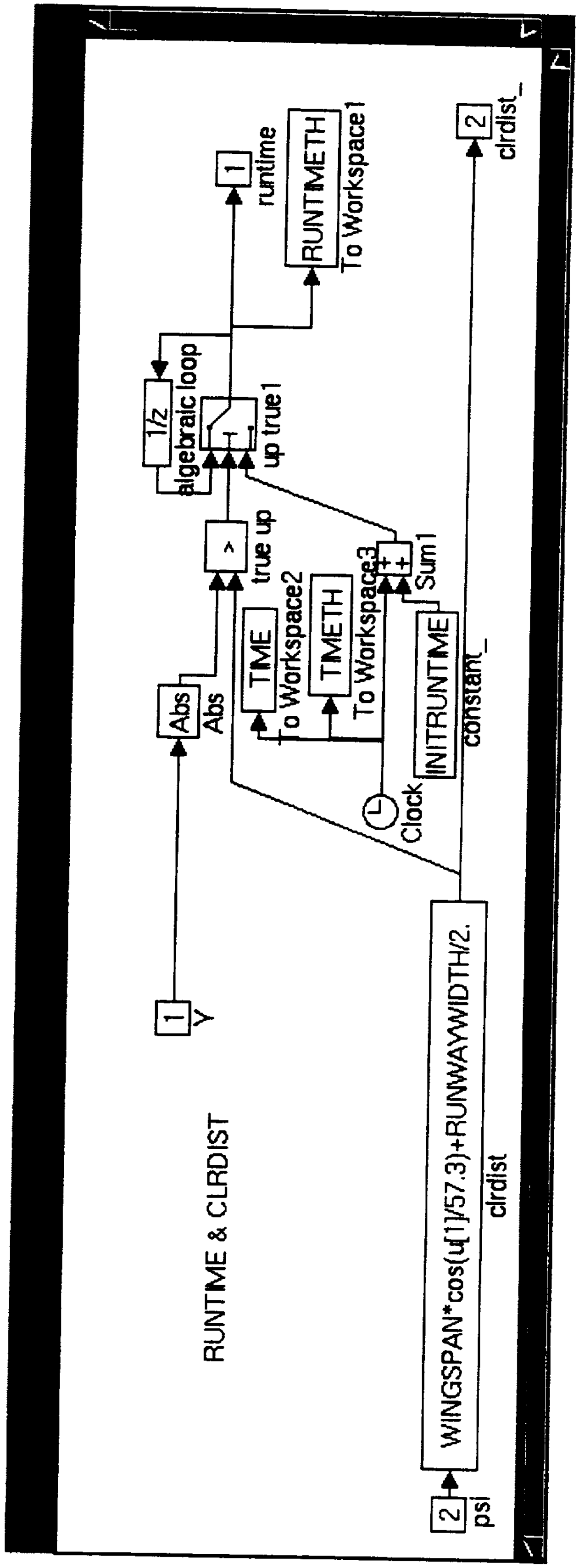
YAW ANGLE OF NOSE GEAR
PSI DY DPSI
2
Demux
Demux
PSI
DY
DPSI
cos(u[1]/57.3)
CPSI
2
CPSI_
sin(u[1]/57.3)
SPSI
3
SPSI
1
Y
Mux
Mux
u[2]+A*sin(u[1]/57.3)
YNG
YNGTH
To Workspace2
Mux
Mux2
u[1]+A*u[2]/57.3*u[3]
DYN
4
DX
Mux
Mux1
u[3]-A*u[1]/57.3*u[2]
DXN
Mux
Mux3
57.3*atan(u[1]/u[2])
PSINGV
0
Constant
~=
Relational Operator
true up
1/z
algebraic loop
3
SIG
PSIN
1
PSIN_
SIGSTRTH
To Workspace


YAW ANGLE OF L&R MAIN GEAR
u[3]-u[1]/57.3*sqrt(pow((TREAD/2),2)+pow(B,2))*sin(atan(TREAD/2/B)-u[2]/57.3)
DXMR
u[3]-u[1]/57.3*sqrt(pow((TREAD/2),2)+pow(B,2))*cos(atan(TREAD/2/B)-u[2]/57.3)
DYMR
u[3]+u[1]/57.3*sqrt(pow((TREAD/2),2)+pow(B,2))*sin(atan(TREAD/2/B)+u[2]/57.3)
DXML
u[3]-u[1]/57.3*sqrt(pow((TREAD/2),2)+pow(B,2))*cos(atan(TREAD/2/B)+u[2]/57.3)
DYML
57.3*atan(u[2]/u[1])
PSIMGVR
57.3*atan(u[2]/u[1])
PSIMGVL
Mux7
Mux3
Mux
Mux1
Mux2
Mux6
Mux5
Demux
1
DX
2
DPSI DY PSI
DPSI
DY
PSI
0
Constant2
Relational Operator2
true up
1/z
algebraic loop
PSIMRTH
To Workspace
PSIMR
2
PSIMR
0
Constant
Relational Operator1
true up_
1/z
algebraic loop1
PSIMLTH
To Workspace1
PSIML
1
PSIML

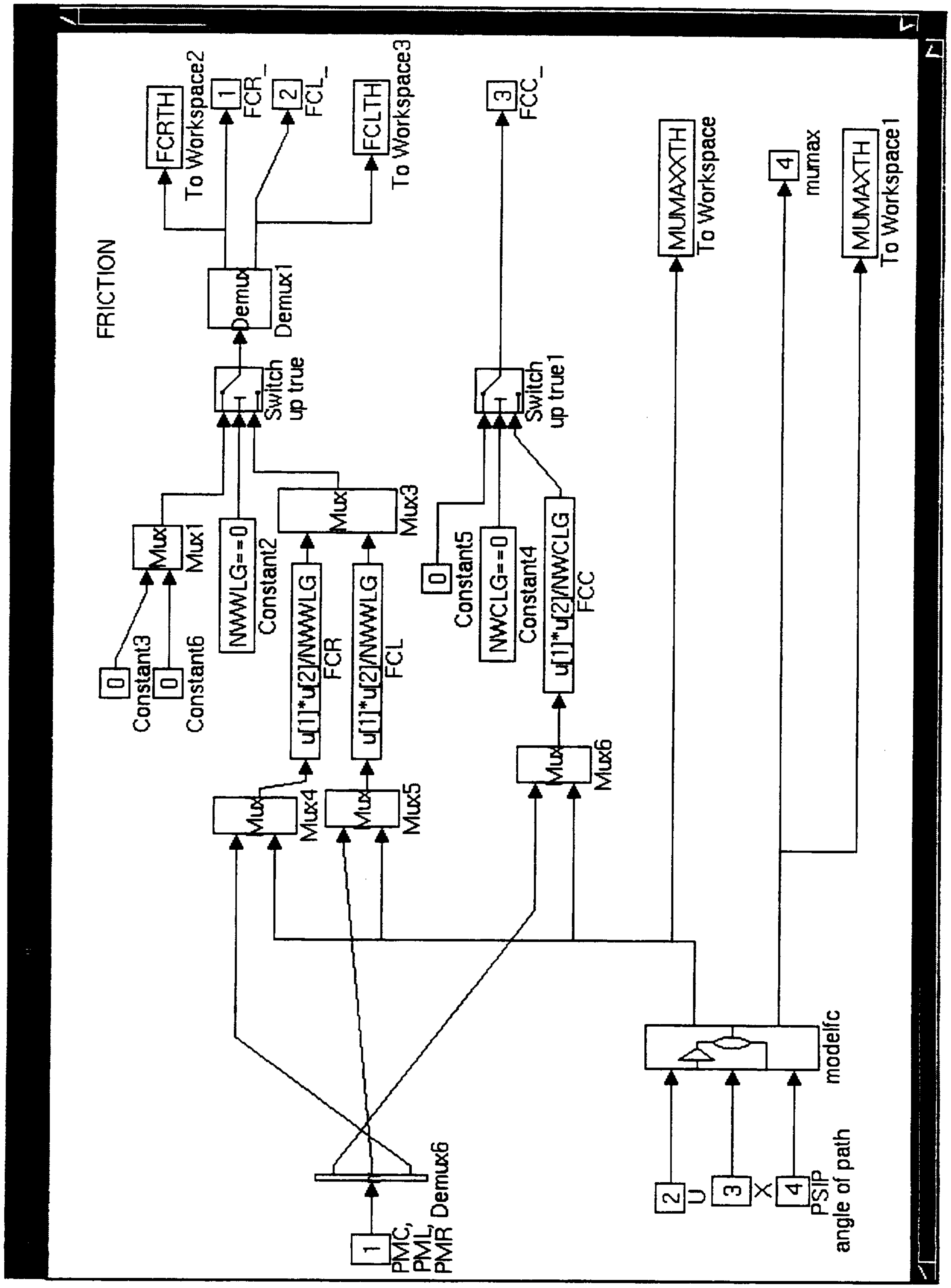

YAW ANGLE OF CENTER MAIN GEAR
4
DX
2
SPSI
Mux
Mux
u[1]+BC*u[3]/57.3*u[2]
DXC
Mux
Mux2
57.3*atan(u[2]/u[1])
PSICGV_
PSICGVTH
To Workspace1
1
CPSI
Mux
Mux1
u[3]-BC*u[1]/57.3*u[2]
DYC
~=
Relational
Operator
0
Constant
true up
1/z
algebraic loop1
-
+
PSIC
1
PSIC_
PSICTH
To Workspace
Demux
Demux
3
DPSI DY PSI
5
DRL
6
DRU
Mux
Mux3
0.8565*(u[1]+u[2])/2.0
DELRSW
2
DELRSW_


INTEGRATION
1
DDPSI
1/s
DPSI
1/s
PSI
5
DPSI_
2
PSI_
DPSITH
To Workspace8
PSITH
To Workspace
3
DU
1/s
Limited Integrator, cluge way of stopping simulation
2
DV
1/s
V
8
U_
UTH
To Workspace6
Mux
Mux
4
V_
Mux
Mux1
4
CPSI
5
SPSI
Mux
Mux3
Mux
Mux2
u[2]*u[3]-u[1]*u[4]
DX
u[3]*u[1]+u[4]*u[2]
DY
6
DX_
1/s
X
1
X_
DXTH
To Workspace3
XTH
To Workspace1
7
DY_
1/s
Y
3
Y_
DYTH
To Workspace4
YTH
To Workspace2

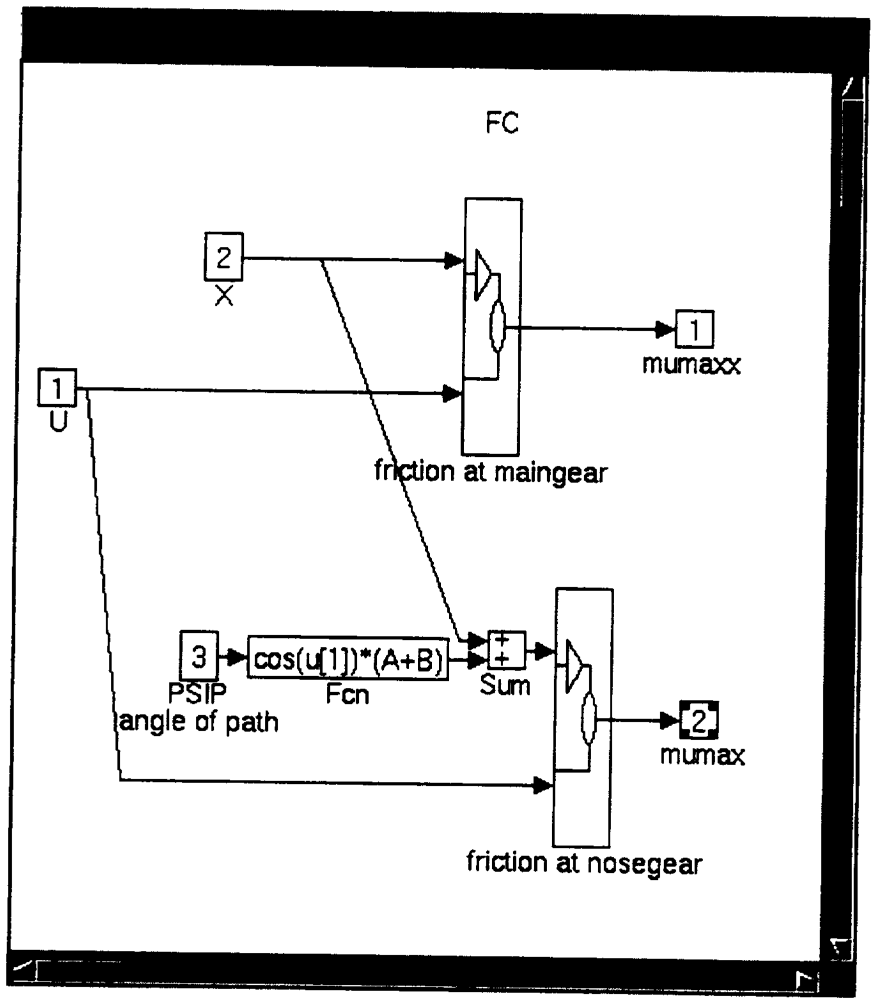

AERO LOADS
SW*u[2]*u[1]
ADRAG
1
ADRAG_
SW*u[1]*u[2]
ALIFT
2
ALIFT_
SW*u[2]*u[1]*LMAC
AMOM
3
AMOM_
u[1]*SW*CYB*u[2]
FYAERO
4
FYAERO_
u[1]*SW*BW*CNB*u[2]
MNAERO
5
MNAERO_
MNAEROTH
To Workspace2
CMR*u[2]*SW*BW*u[1]
TR
6
TR_
TRTH
To Workspace1
Mux4
Mux5
Mux7
Mux8
Mux9
Mux10
3
CDRAG
2
CLIFT
4
CMOM
9
DELRSW
atan(u[1]/u[2])*57.3
YAW
YAWTH
To Workspace
true up
1/z
algebraic loop2
u[1]/1.689
UT
7
UT_
(pow((u[1]/1.689),2))/295.37
QAERO
1/z
algebraic loop1
true up_
==
UU == 0
0
Constant
atan(u[2]/u[1])*57.3
BETA
Mux3
~=
U~= 0?
0
Constant1
u[1]-u[2]*(u[3]*1.689)-u[4]*(u[5]*1.689)
UU
Mux1
u[5]-u[2]*(u[3]*1.689)+u[4]*(u[1]*1.689)
VV
Mux
Mux2
1
U
8
VTW
5
CPS
7
VW
6
SPS
10
V

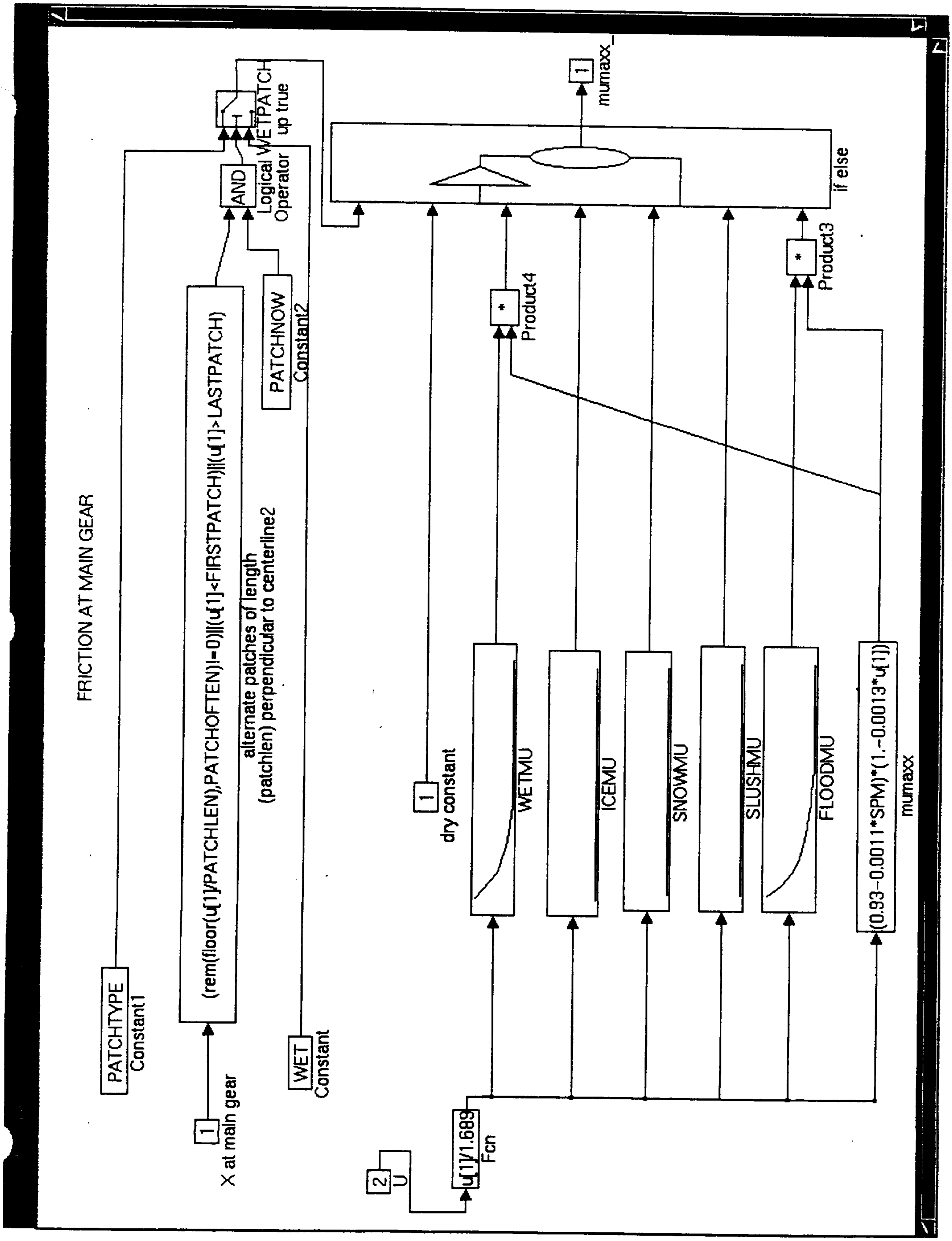


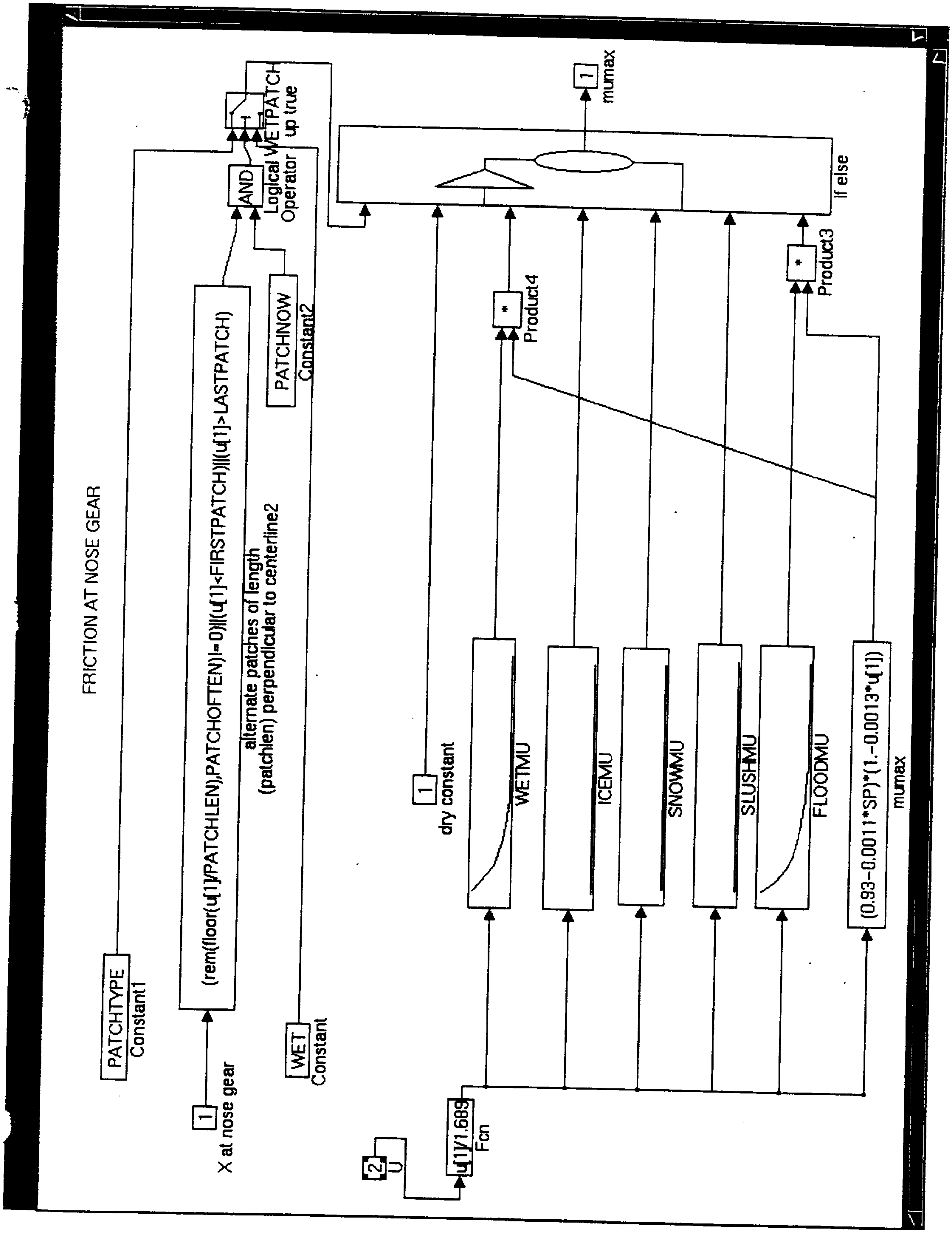
RUNTIME & CLRDIST
1
Y
Abs
Abs
TIME
To Workspace2
TIMETH
To Workspace3
>
true up
Clock
INITRUNTIME
constant_
+
+
Sum1
1/z
algebraic loop
up true1
RUNTIMETH
To Workspace1
1
runtime
WINGSPAN*cos(u[1]/57.3)+RUNWAYWIDTH/2.
clrdist
2
psi
2
clrdist_

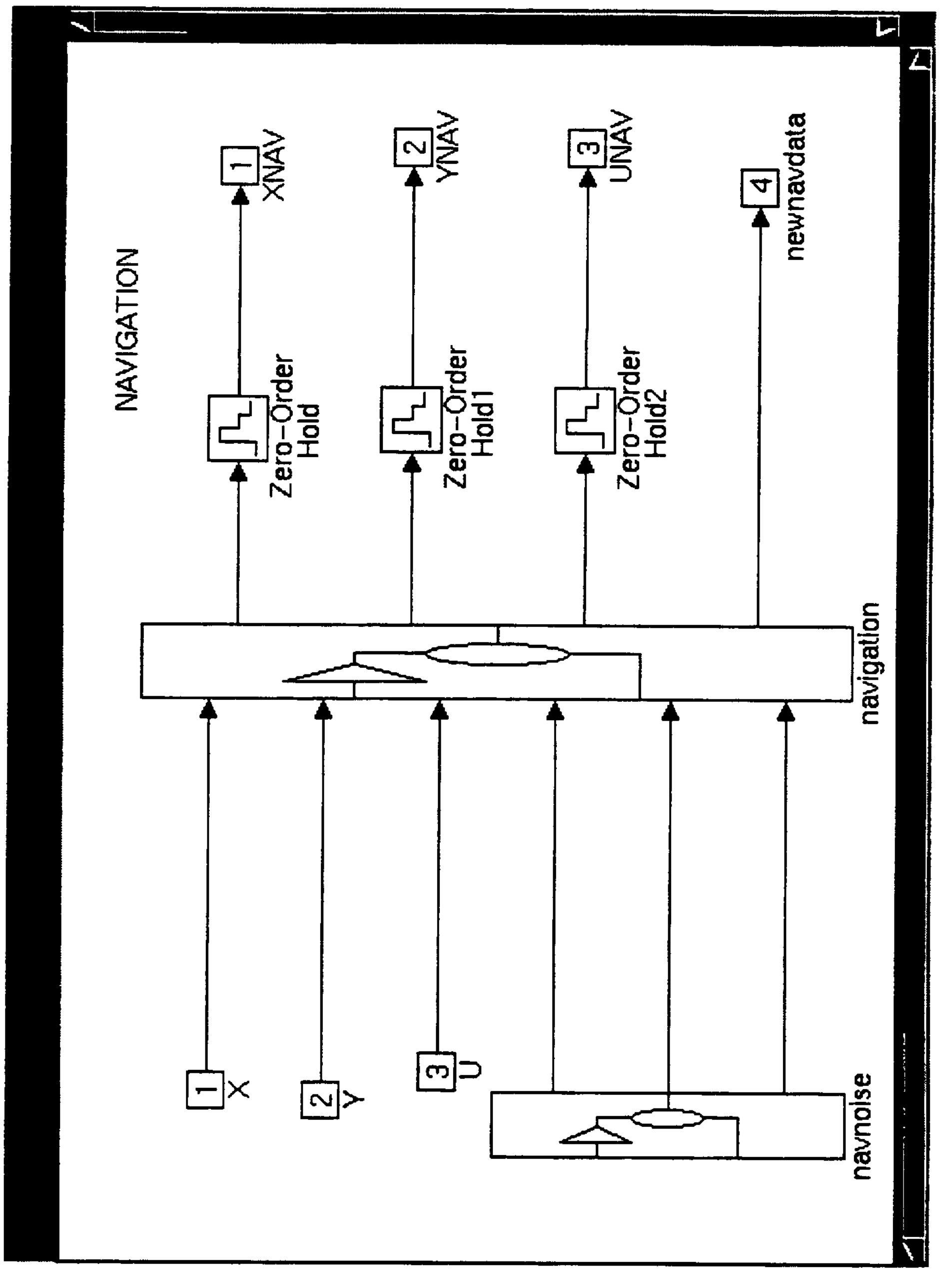


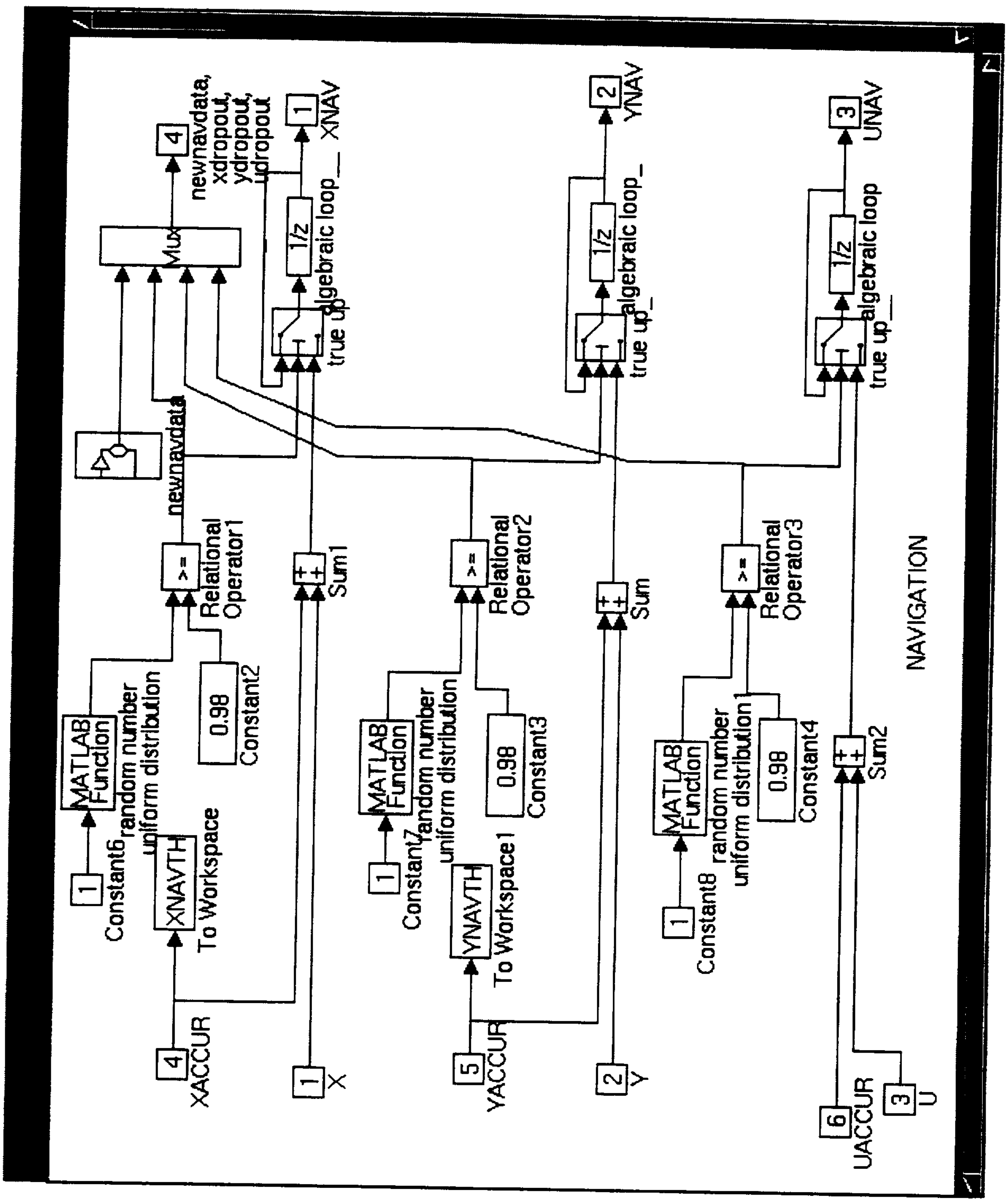
FRICTION
1
PMC,
PML,
PMR
Demux6
2
U
3
X
4
PSIP
angle of path
modelfc
Mux
Mux4
Mux
Mux5
Mux
Mux6
u[1]*u[2]/NWWLG
FCR
u[1]*u[2]/NWWLG
FCL
u[1]*u[2]/NWCLG
FCC
0
Constant3
0
Constant6
Mux
Mux1
NWWLG==0
Constant2
Mux
Mux3
0
Constant5
NWCLG==0
Constant4
Switch
up true
Switch
up true1
Demux
Demux1
FCRTH
To Workspace2
1
FCR_
2
FCL_
FCLTH
To Workspace3
3
FCC_
MUMAXXTH
To Workspace
4
mumax
MUMAXTH
To Workspace1

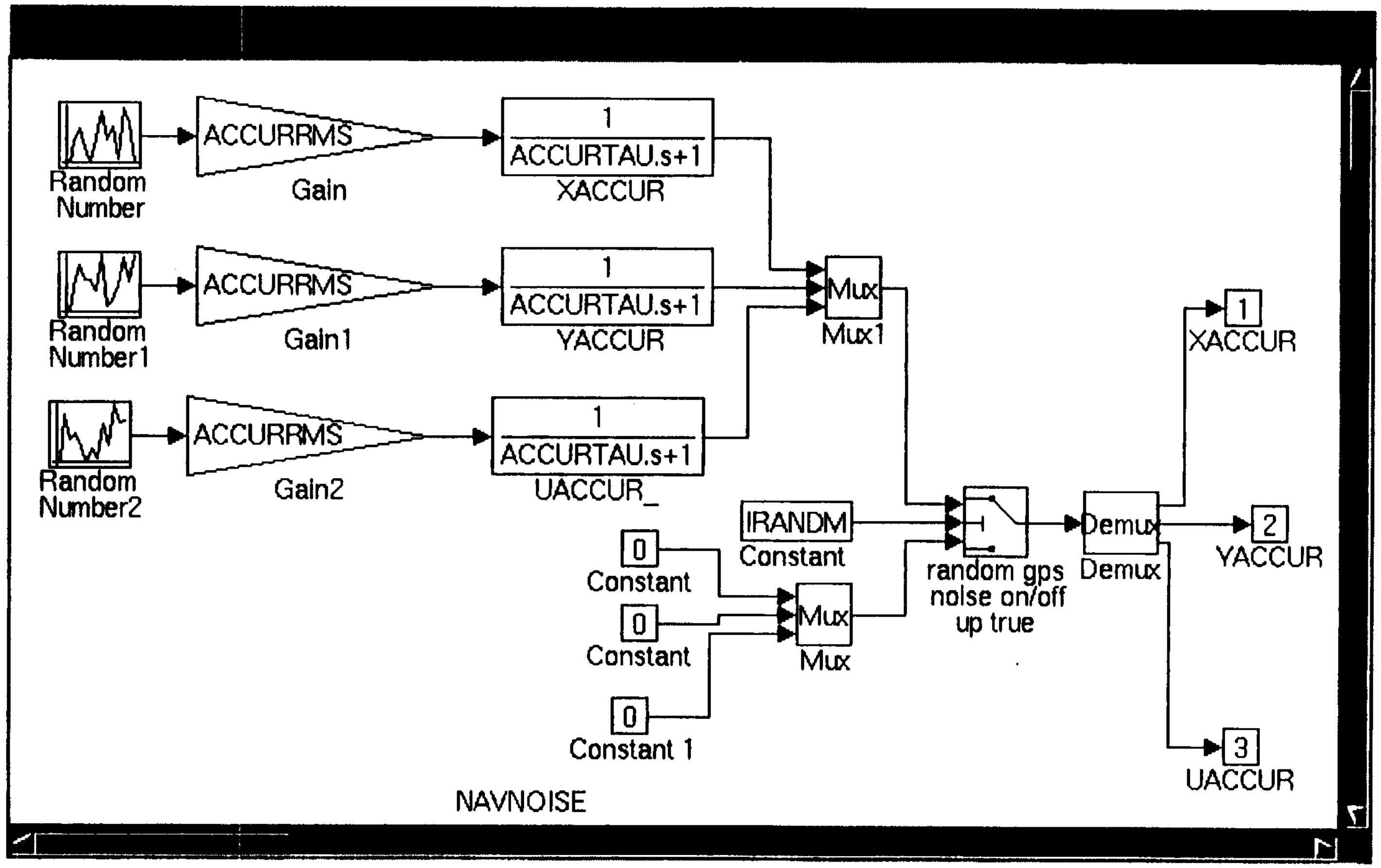

FC
2
X
1
U
friction at maingear
1
mumaxx
3
PSIP
angle of path
cos(u[1])*(A+B)
Fcn
+
+
Sum
friction at nosegear
2
mumax

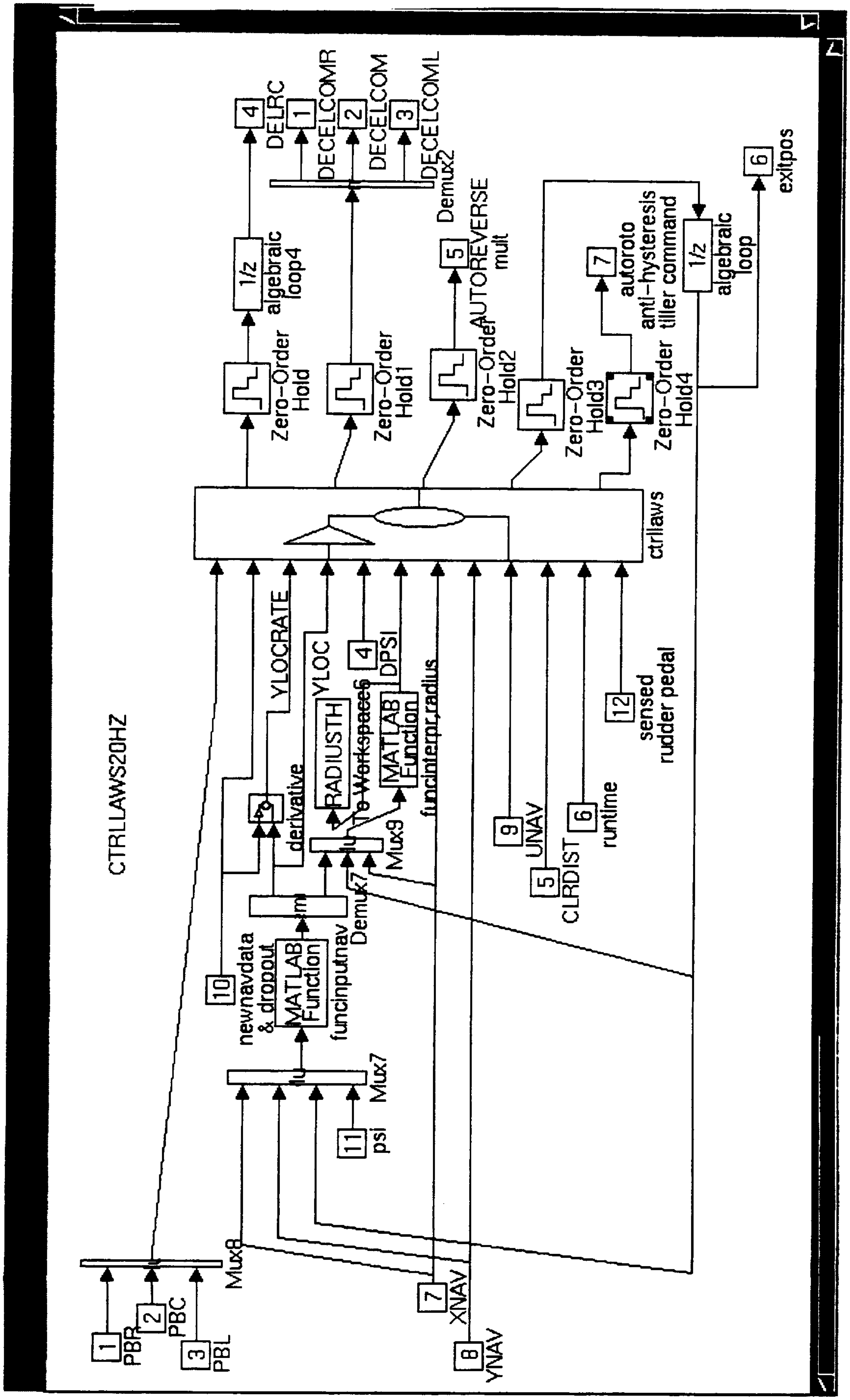

FRICTION AT MAIN GEAR
PATCHTYPE
Constant1
(rem(floor(u[1]/PATCHLEN),PATCHOFTEN)!=0)||(u[1]<FIRSTPATCH)||(u[1]>LASTPATCH)
alternate patches of length
(patchlen) perpendicular to centerline2
PATCHNOW
Constant2
AND
Logical
Operator
WETPATCH
up true
WET
Constant
1
X at main gear
2
U
u[1]/1.689
Fcn
1
dry constant
WETMU
ICEMU
SNOWMU
SLUSHMU
FLOODMU
(0.93-0.0011*SPM)*(1.-0.0013*u[1])
mumaxx
*
Product4
*
Product3
if else
1
mumaxx_

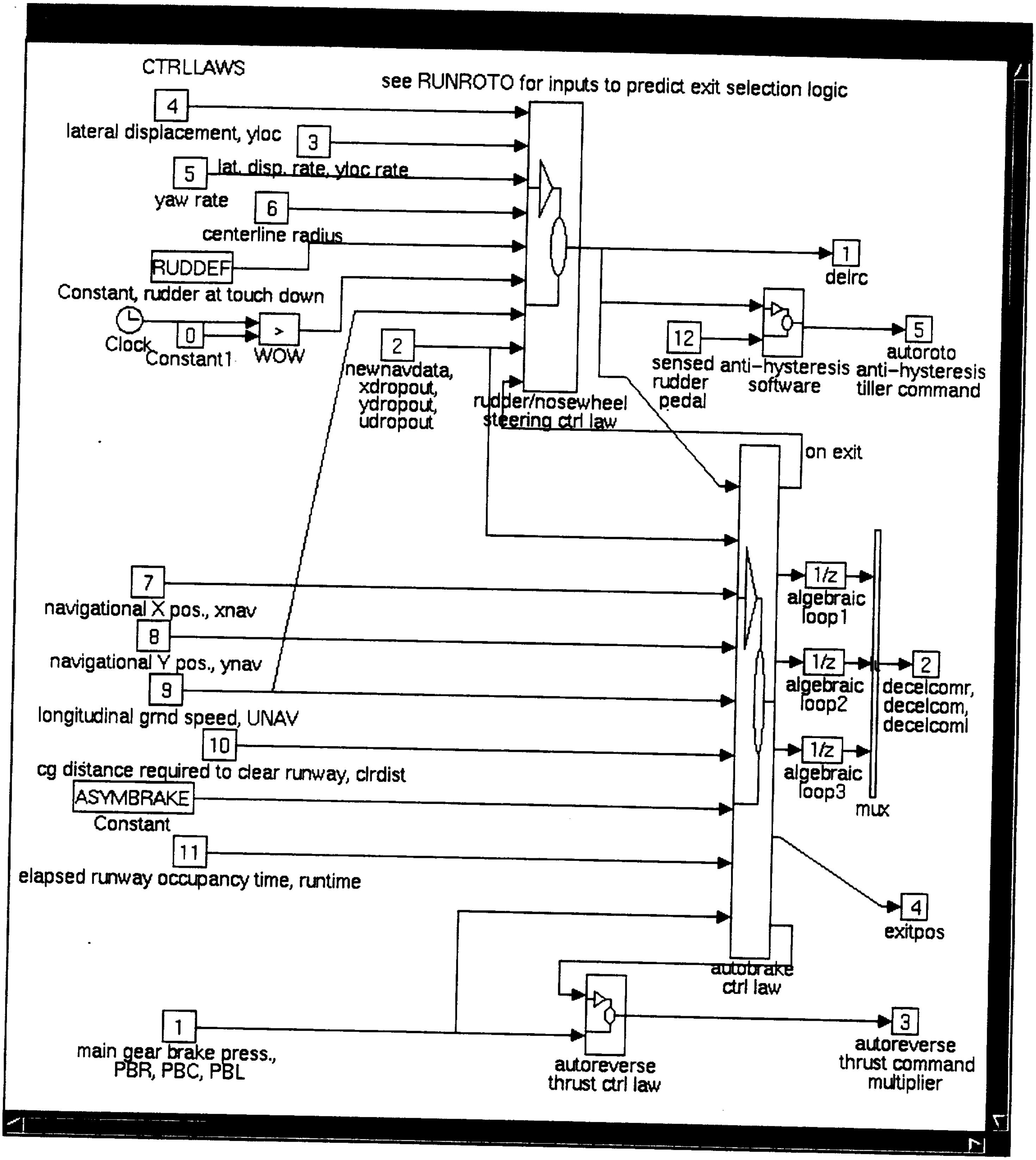

FRICTION AT NOSE GEAR
PATCHTYPE
Constant1
(rem(floor(u[1]/PATCHLEN),PATCHOFTEN)!=0)||(u[1]<FIRSTPATCH)||(u[1]>LASTPATCH)
alternate patches of length
(patchlen) perpendicular to centerline2
1
X at nose gear
WET
Constant
PATCHNOW
Constant2
AND
Logical
Operator
WETPATCH
up true
1
mumax
if else
*
Product4
*
Product3
1
dry constant
WETMU
ICEMU
SNOWMU
SLUSHMU
FLOODMU
(0.93-0.0011*SP)*(1.-0.0013*u[1])
mumax
2
U
u[1]/1.689
Fcn

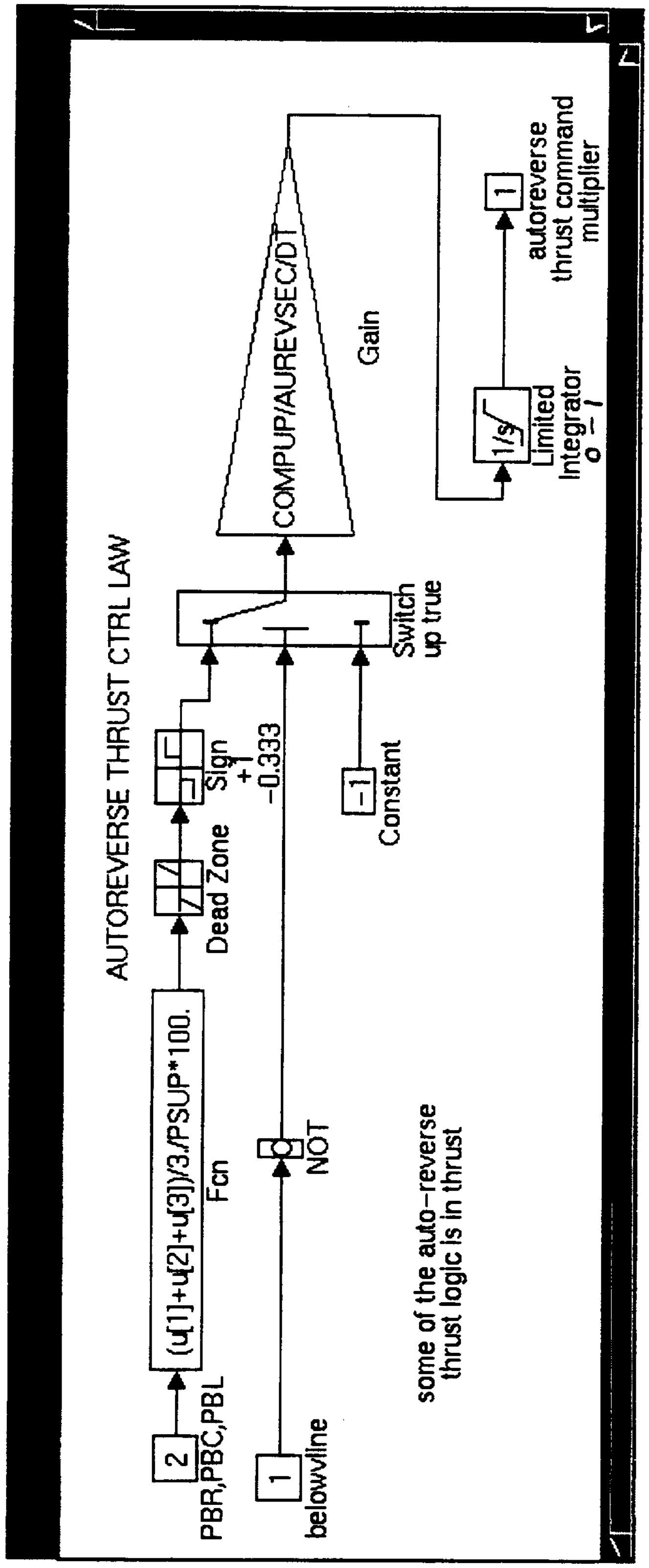

NAVIGATION
1
X
2
Y
3
U
navnoise
navigation
Zero-Order
Hold
Zero-Order
Hold1
Zero-Order
Hold2
1
XNAV
2
YNAV
3
UNAV
4
newnavdata

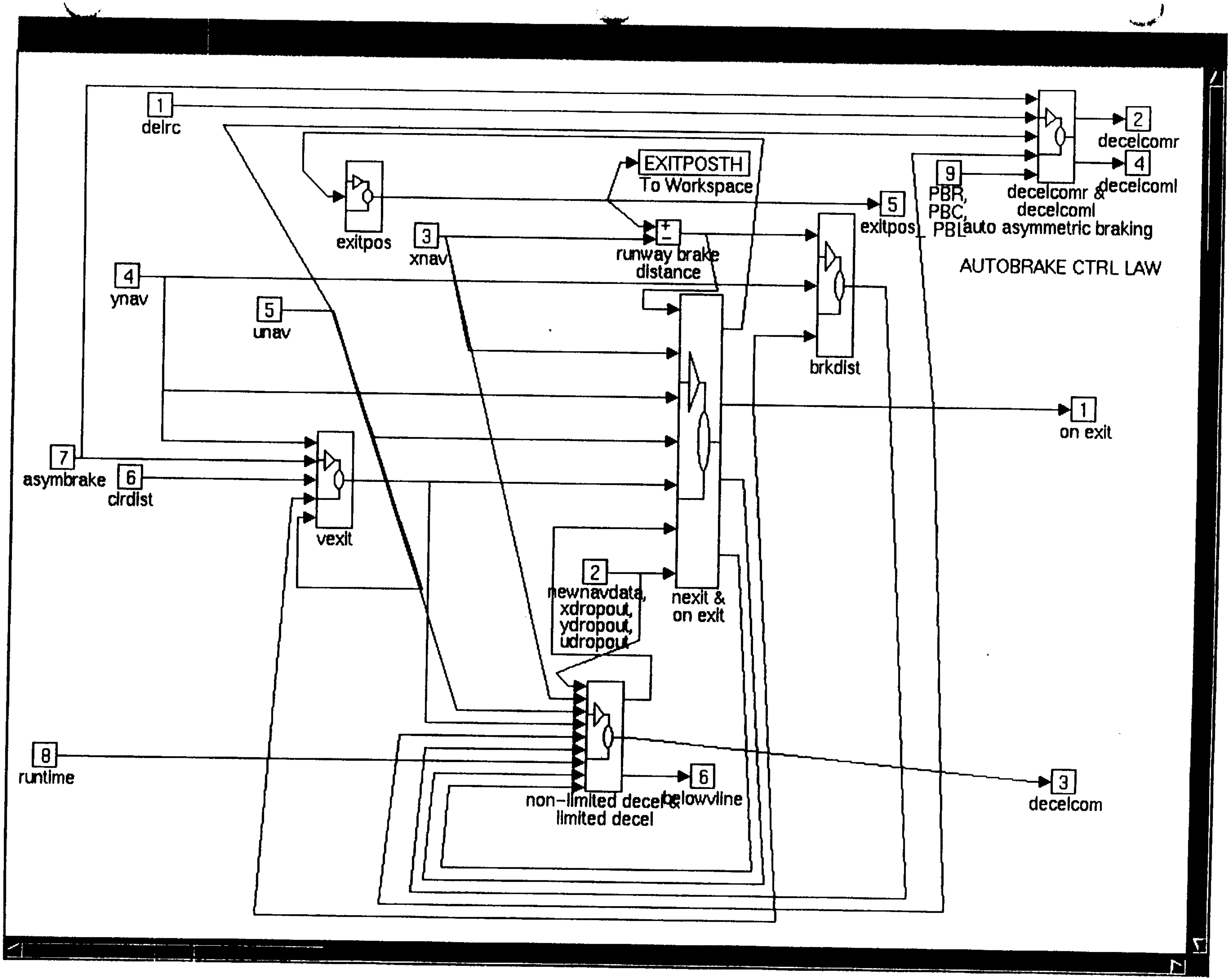

NAVIGATION
4 XACCUR
1 X
5 YACCUR
2 Y
6 UACCUR
3 U
1 Constant6
MATLAB Function
random number uniform distribution
XNAVTH
To Workspace
0.98
Constant2
Relational Operator1
Sum1
newnavdata
1 Constant7
MATLAB Function
random number uniform distribution
YNAVTH
To Workspace1
0.98
Constant3
Relational Operator2
Sum
1 Constant8
MATLAB Function
random number uniform distribution1
0.98
Constant4
Relational Operator3
Sum2
Mux
4 newnavdata, xdropout, ydropout, udropout
true up
1/z
algebraic loop__
1 XNAV
true up_
algebraic loop_
2 YNAV
true up__
algebraic loop
3 UNAV


Random Number
ACCURRMS
Gain
1
ACCURTAU.s+1
XACCUR
Random Number1
ACCURRMS
Gain1
1
ACCURTAU.s+1
YACCUR
Random Number2
ACCURRMS
Gain2
1
ACCURTAU.s+1
UACCUR_
Mux
Mux1
IRANDM
Constant
0
Constant
0
Constant
0
Constant 1
Mux
Mux
random gps
noise on/off
up true
Demux
Demux
1
XACCUR
2
YACCUR
3
UACCUR
NAVNOISE

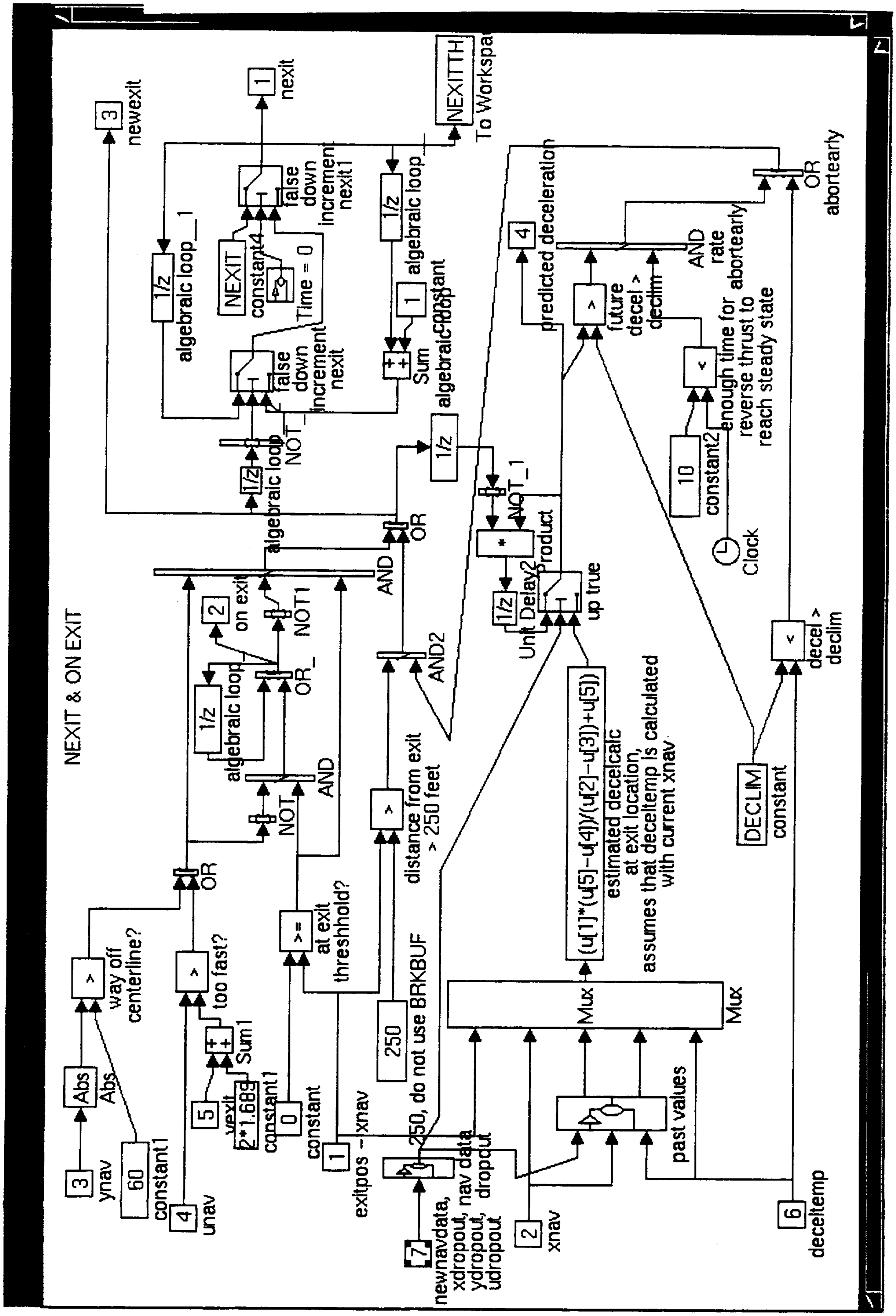

CTRLLAWS20HZ
PBR
PBC
PBL
Mux8
XNAV
YNAV
psi
Mux7
newnavdata & dropout
MATLAB Function
funcinputnav
Demux7
derivative
YLOCRATE
YLOC
RADIUSTH
To Workspace6
MATLAB Function
funcinterpr,radius
Mux9
DPSI
UNAV
CLRDIST
runtime
sensed rudder pedal
ctrllaws
Zero-Order Hold
Zero-Order Hold1
Zero-Order Hold2
Zero-Order Hold3
Zero-Order Hold4
1/z
algebraic loop4
algebraic loop
DELRC
DECELCOMR
DECELCOM
DECELCOML
Demux2
AUTOREVERSE mult
autoroto anti-hysteresis tiller command
exitpos


CTRLLAWS
see RUNROTO for inputs to predict exit selection logic
4
lateral displacement, yloc
3
5
lat. disp. rate, yloc rate
yaw rate
6
centerline radius
RUDDEF
Constant, rudder at touch down
Clock
0
Constant1
>
WOW
2
newnavdata, xdropout, ydropout, udropout
rudder/nosewheel steering ctrl law
1
delrc
12
sensed rudder pedal
anti-hysteresis software
5
autoroto anti-hysteresis tiller command
on exit
7
navigational X pos., xnav
8
navigational Y pos., ynav
9
longitudinal grnd speed, UNAV
10
cg distance required to clear runway, clrdist
ASYMBRAKE
Constant
11
elapsed runway occupancy time, runtime
1/z
algebraic loop1
1/z
algebraic loop2
1/z
algebraic loop3
mux
2
decelcomr, decelcom, decelcoml
4
exitpos
autobrake ctrl law
1
main gear brake press., PBR, PBC, PBL
autoreverse thrust ctrl law
3
autoreverse thrust command multiplier

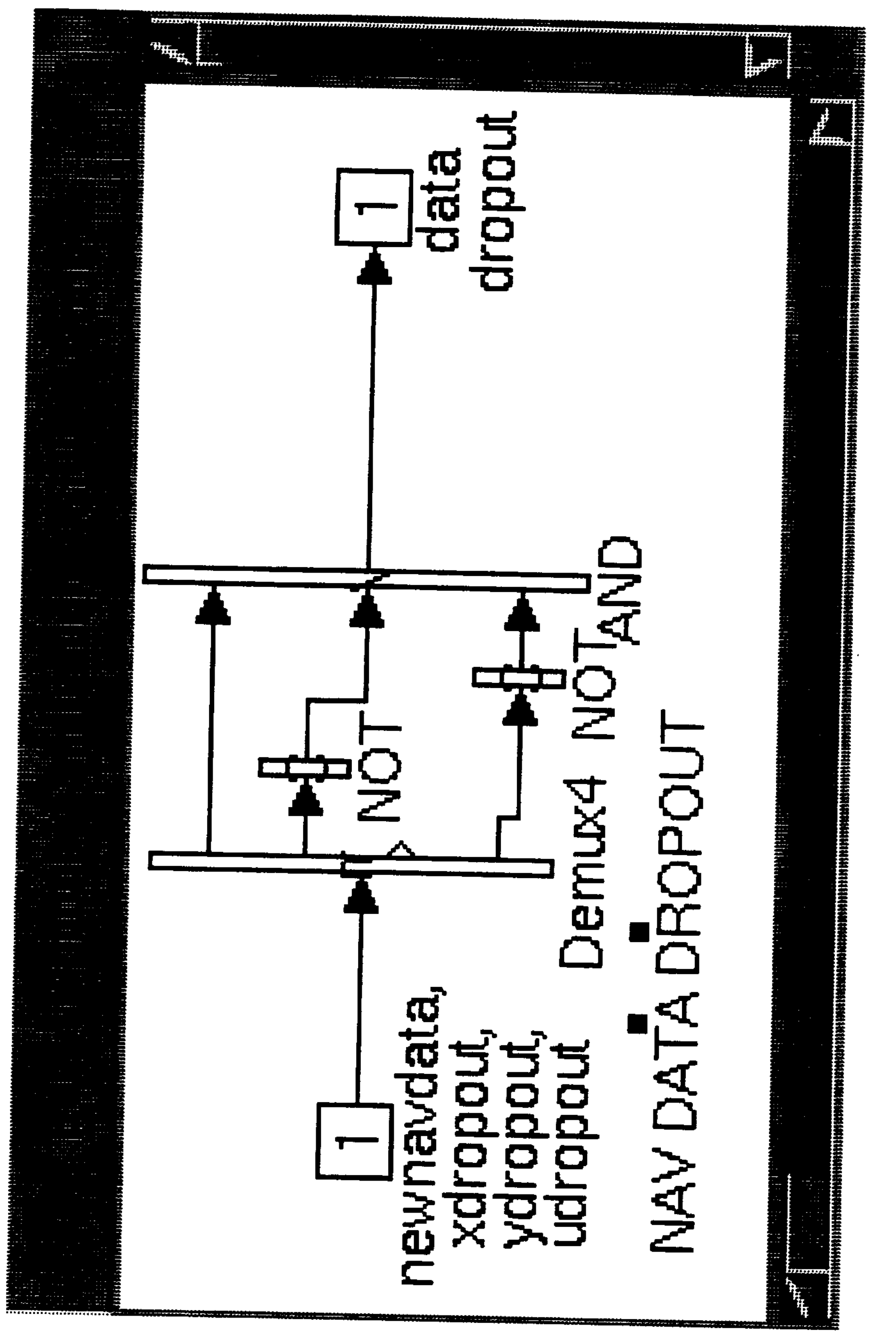

AUTOREVERSE THRUST CTRL LAW
2
PBR,PBC,PBL
(u[1]+u[2]+u[3])/3./PSUP*100.
Fcn
Dead Zone
Sign
+1
-0.333
1
belowvline
NOT
-1
Constant
Switch
up true
COMPUP/AUREVSEC/DT
Gain
1/s
Limited
Integrator
0 - 1
1
autoreverse
thrust command
multiplier
some of the auto-reverse
thrust logic is in thrust

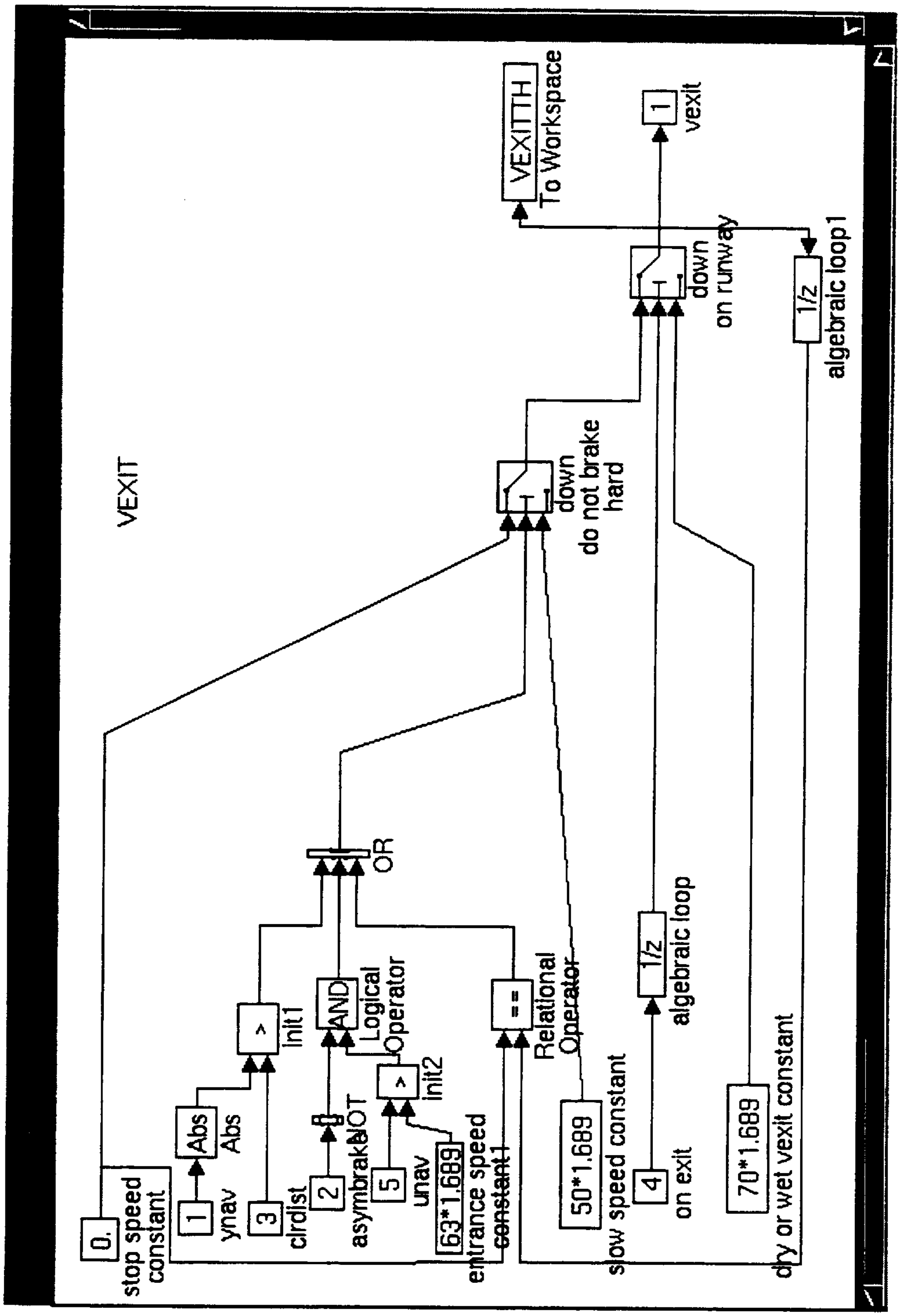


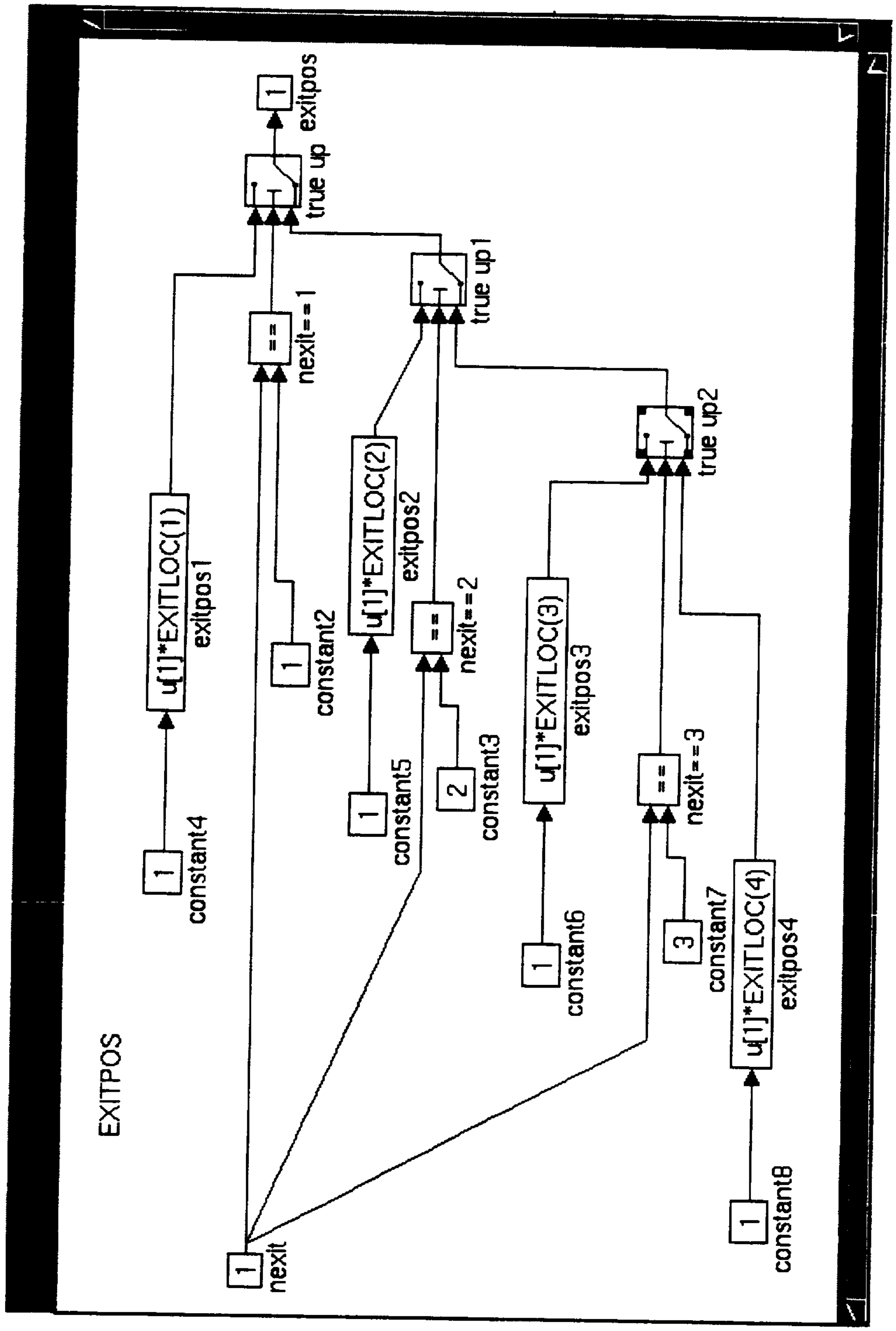
1
delrc
4
ynav
5
unav
3
xnav
7
asymbrake
6
clrdist
8
runtime
exitpos
EXITPOSTH
To Workspace
runway brake
distance
vexit
2
newnavdata,
xdropout,
ydropout,
udropout
nexit &
on exit
brkdist
5
exitpos
9
PBR,
PBC,
PBL
decelcomr &
decelcoml
auto asymmetric braking
2
decelcomr
4
decelcoml
AUTOBRAKE CTRL LAW
1
on exit
non-limited decel &
limited decel
6
belowvline
3
decelcom



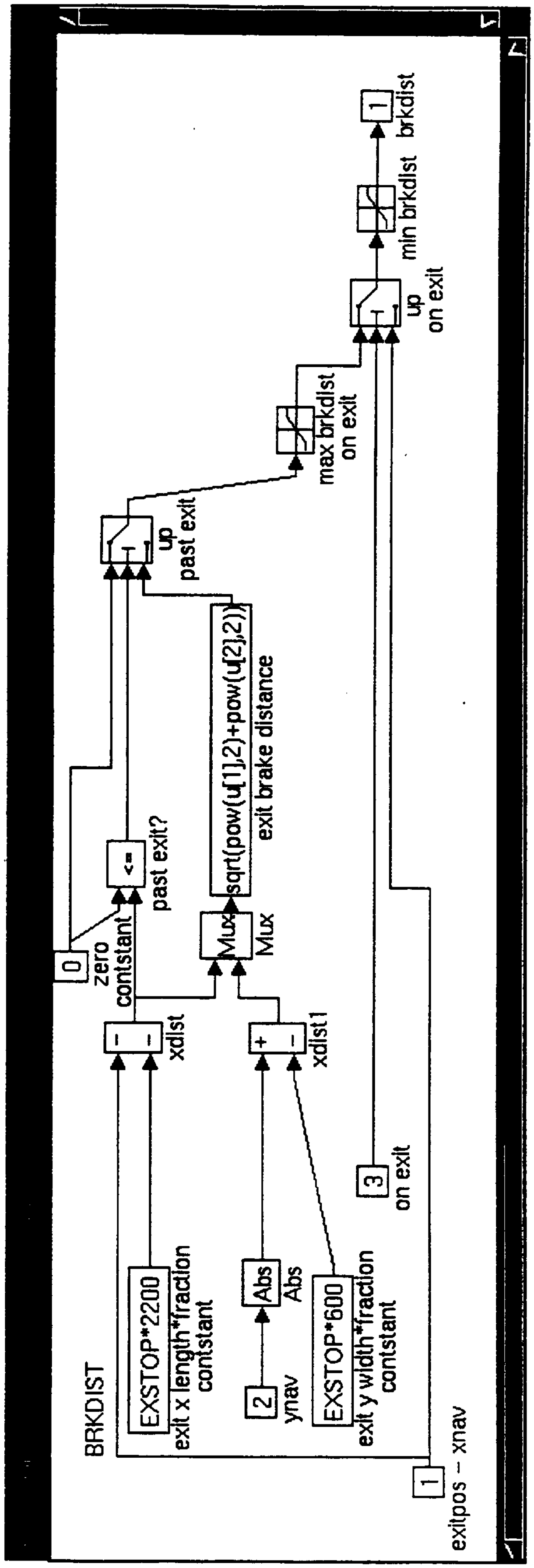
NEXIT & ON EXIT
3
ynav
60
constant1
Abs
Abs
way off centerline?
4
unav
5
vexit
2*1.689
constant1
Sum1
too fast?
0
constant
at exit threshhold?
1
exitpos - xnav
250
250, do not use BRKBUF
distance from exit > 250 feet
7
newnavdata, xdropout, ydropout, udropout
nav data dropout
2
xnav
6
deceltemp
past values
Mux
Mux
(u[1]*(u[5]-u[4])/(u[2]-u[3])+u[5])
estimated decelcalc at exit location, assumes that deceltemp is calculated with current xnav
OR
NOT
AND
1/z
algebraic loop
OR_
2
on exit
NOT1
AND
OR
AND2
3
newexit
1/z
algebraic loop__1
false down increment nexit
NEXIT
constant4
Time = 0
false down increment nexit1
1
nexit
NEXITTH
To Workspa
1/z
algebraic loop
NOT_
1/z
algebraic loop__
Sum
1
constant
algebraic loop
1/z
NOT_1
*
Product
1/z
Unit Delay2
up true
4
predicted deceleration
future decel > declim
AND
rate
abortearly
enough time for reverse thrust to reach steady state
10
constant2
Clock
OR
abortearly
DECLIM
constant
decel > declim

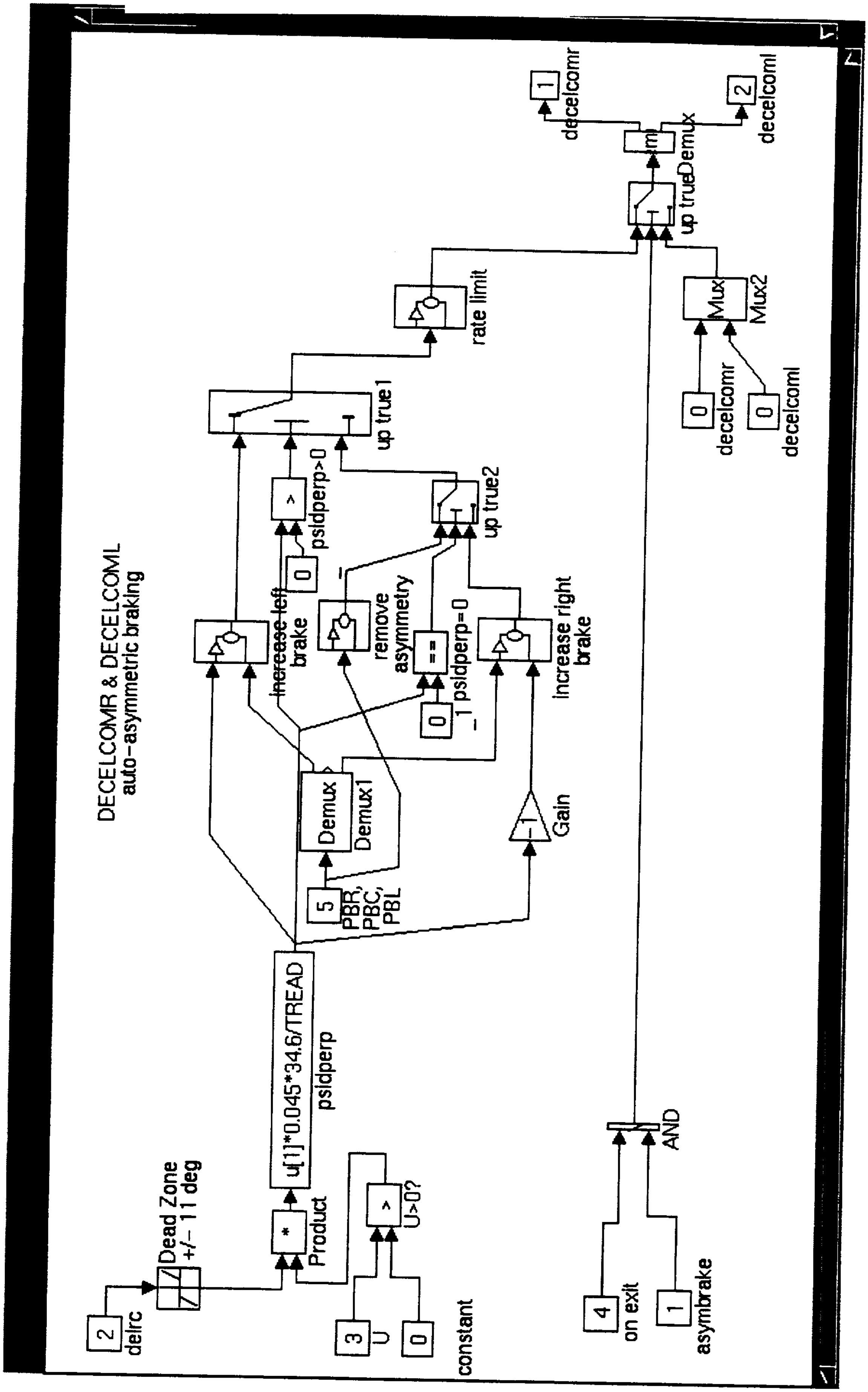

1
newnavdata,
xdropout,
ydropout,
udropout
Demux4
NOT
NOT
AND
1
data
dropout
NAV DATA DROPOUT

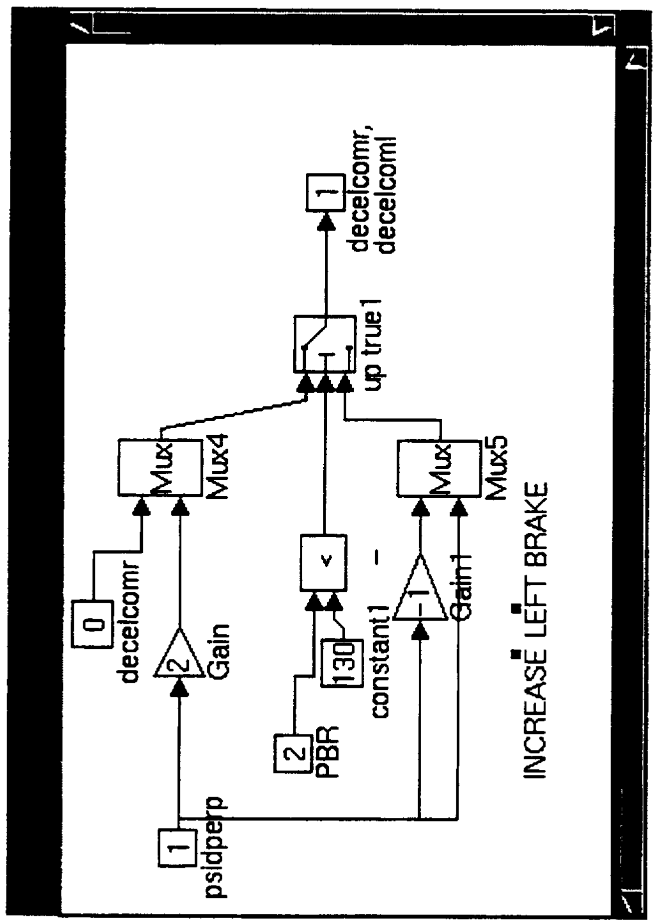

VEXIT
VEXITTH
To Workspace
1
vexit
down
on runway
1/z
algebraic loop1
down
do not brake
hard
OR
>
init1
AND
Logical
Operator
==
Relational
Operator
1/z
algebraic loop
0.
stop speed
constant
1
ynav
Abs
Abs
3
clrdist
2
asymbrake
NOT
5
unav
>
init2
63*1.689
entrance speed
constant1
50*1.689
slow speed constant
4
on exit
70*1.689
dry or wet vexit constant

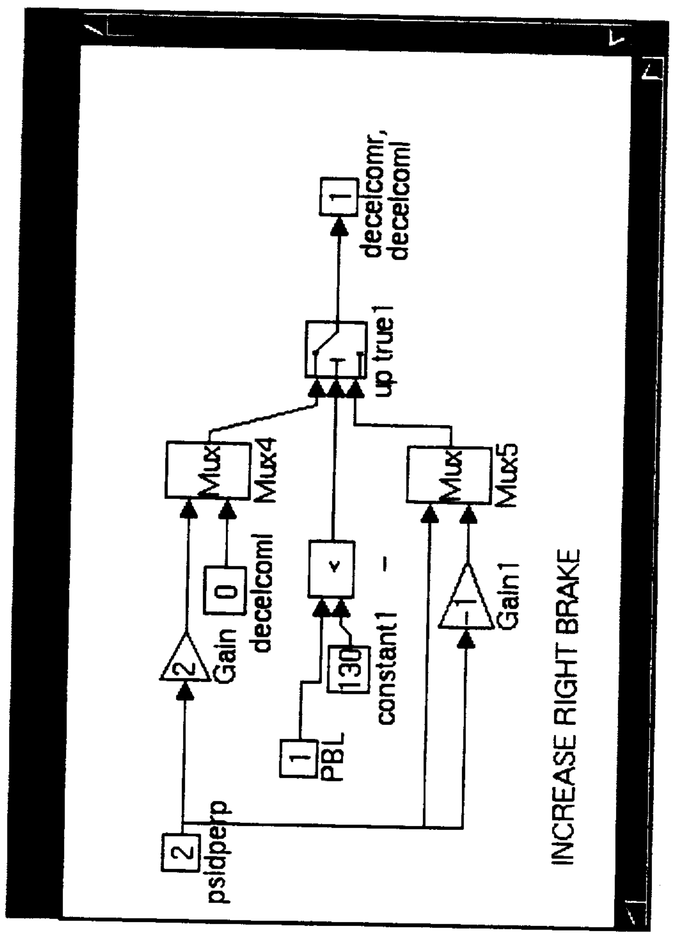


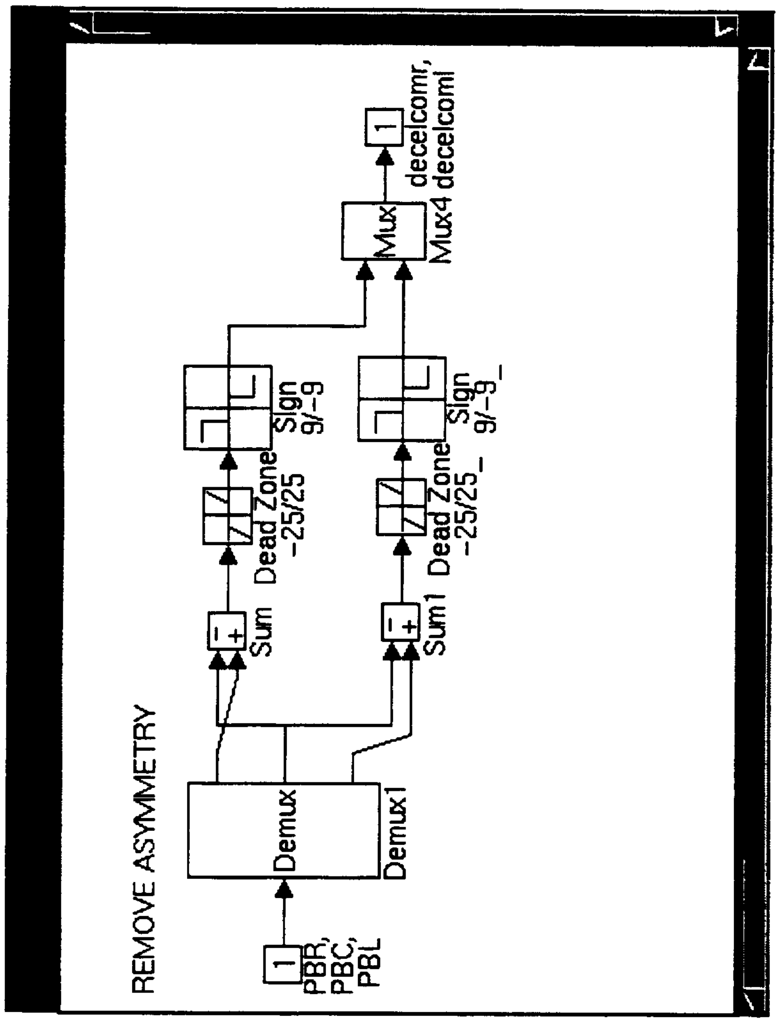
EXITPOS
1
nexit
1
constant4
u[1]*EXITLOC(1)
exitpos1
1
constant2
==
nexit==1
true up
1
exitpos
1
constant5
u[1]*EXITLOC(2)
exitpos2
2
constant3
==
nexit==2
true up1
1
constant6
u[1]*EXITLOC(3)
exitpos3
3
constant7
==
nexit==3
true up2
1
constant8
u[1]*EXITLOC(4)
exitpos4


BRKDIST
EXSTOP*2200
exit x length*fraction
contstant
2
ynav
Abs
Abs
EXSTOP*600
exit y width*fraction
contstant
1
exitpos -- xnav
xdist
xdist1
0
zero
contstant
Mux
Mux
<=
past exit?
sqrt(pow(u[1],2)+pow(u[2],2))
exit brake distance
up
past exit
max brkdist
on exit
3
on exit
up
on exit
min brkdist
1
brkdist

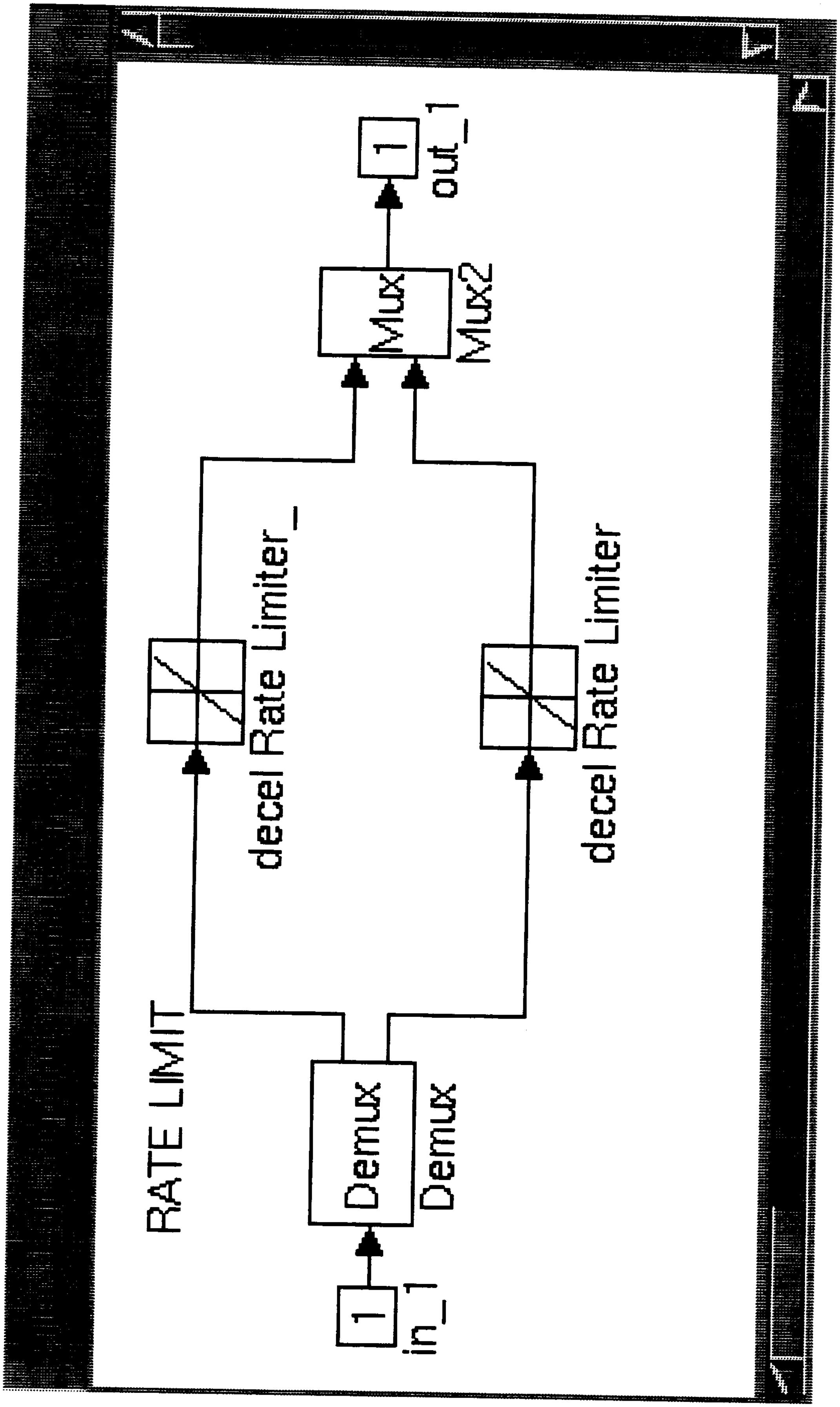

DECELCOMR & DECELCOML
auto-asymmetric braking
2
delrc
Dead Zone
+/- 11 deg
3
U
0
constant
Product
U>0?
u[1]*0.045*34.6/TREAD
psidperp
5
PBR,
PBC,
PBL
Demux
Demux1
Gain
-1
increase left
brake
remove
asymmetry
increase right
brake
_1 psidperp=0
psidperp>0
up true1
up true2
rate limit
4
on exit
1
asymbrake
AND
decelcomr
decelcoml
Mux
Mux2
up true
Demux
1
decelcomr
2
decelcoml


1
psidperp
2
PBR
0
decelcomr
2
Gain
130
constant1
<
-1
Gain1
Mux
Mux4
Mux
Mux5
up true1
1
decelcomr,
decelcoml
INCREASE LEFT BRAKE

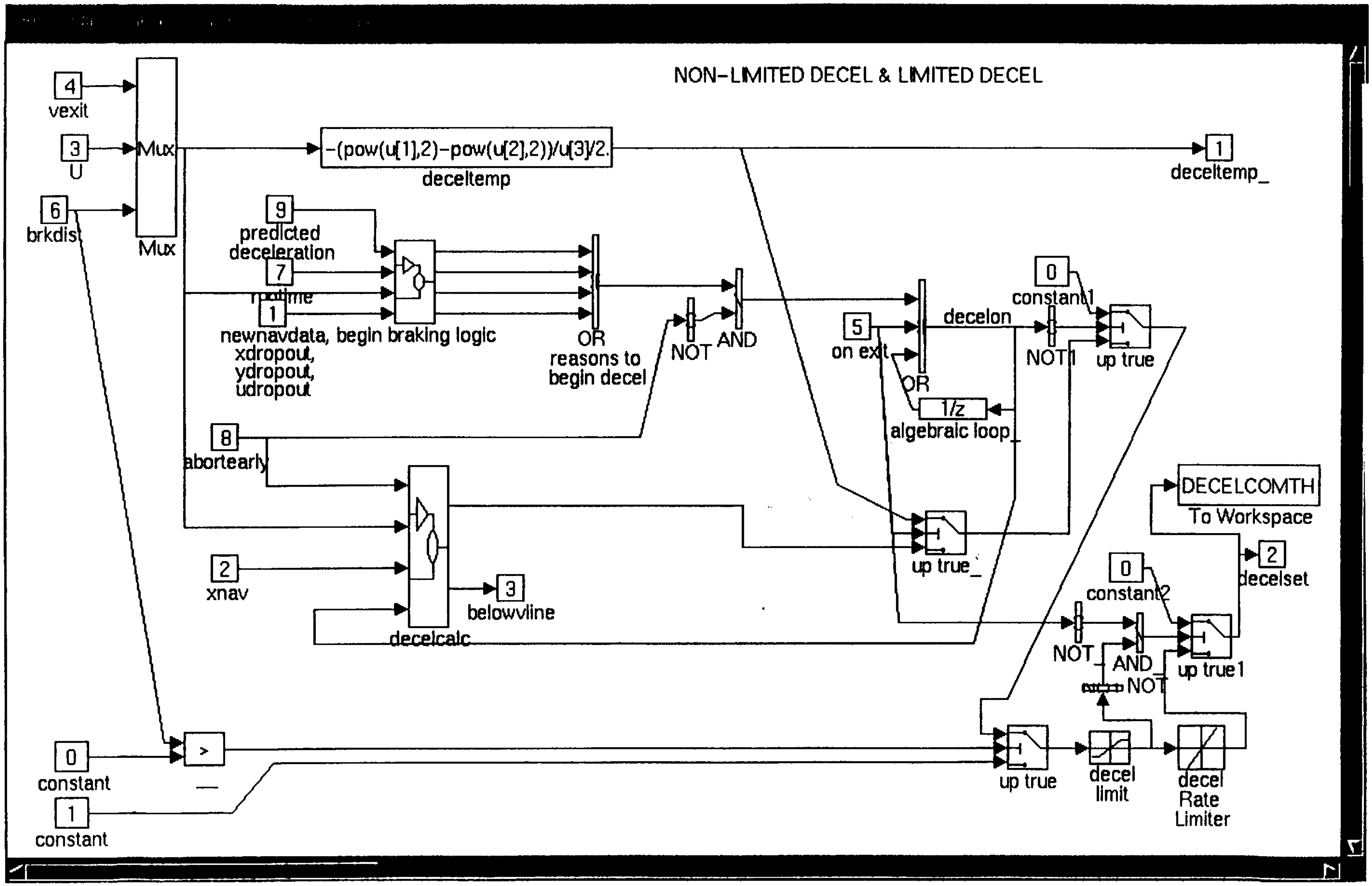

2
psidperp
2
Gain
0
decelcoml
Mux
Mux4
1
PBL
130
constant1
<
Mux
Mux5
-1
Gain1
up true1
1
decelcomr,
decelcoml
INCREASE RIGHT BRAKE



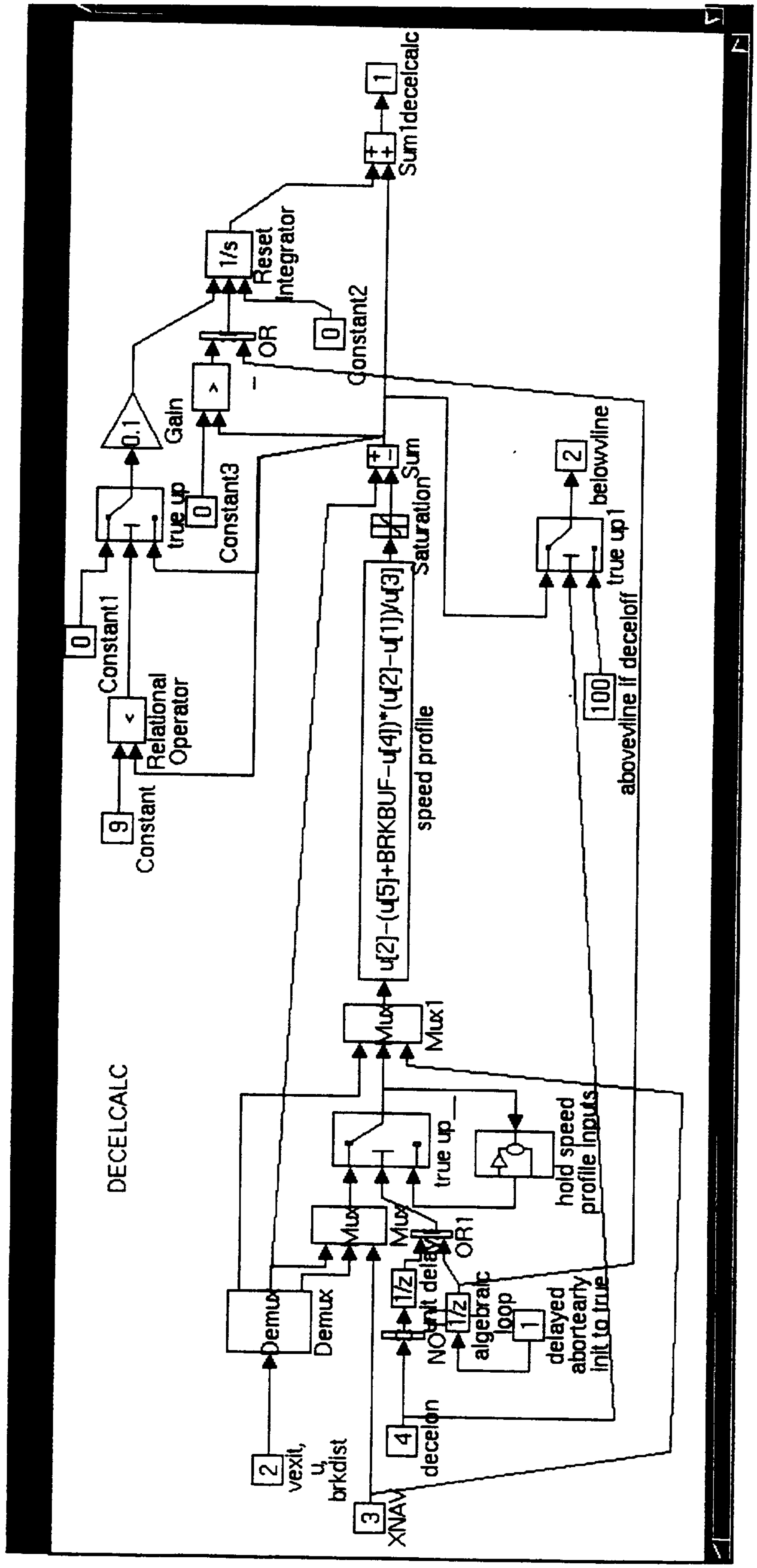
REMOVE ASYMMETRY
1
PBR,
PBC,
PBL
Demux
Demux1
Sum
Sum1
Dead Zone
-25/25
Dead Zone
-25/25_
Sign
9/-9
Sign
9/-9_
Mux
Mux4
1
decelcomr,
decelcoml

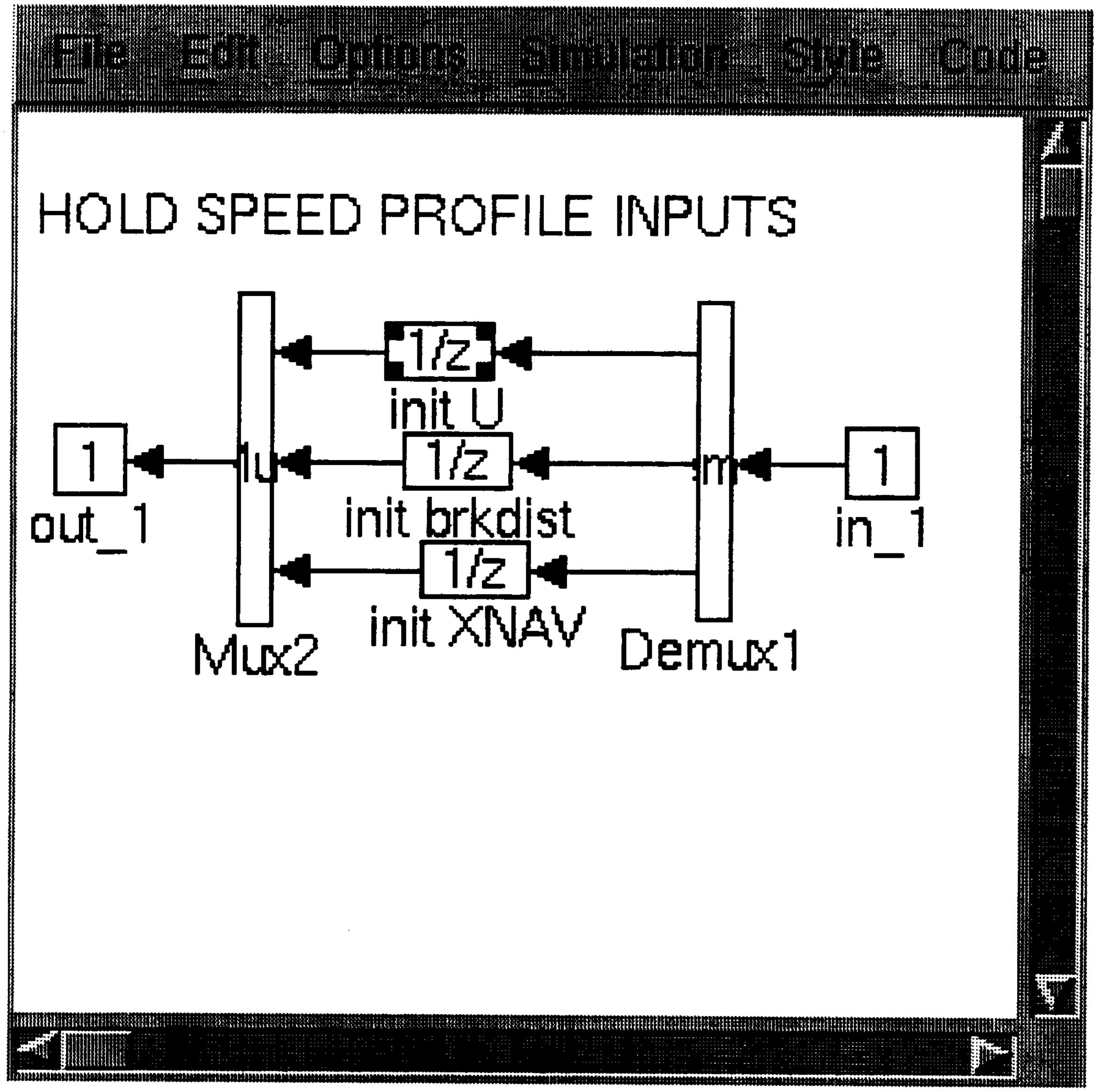


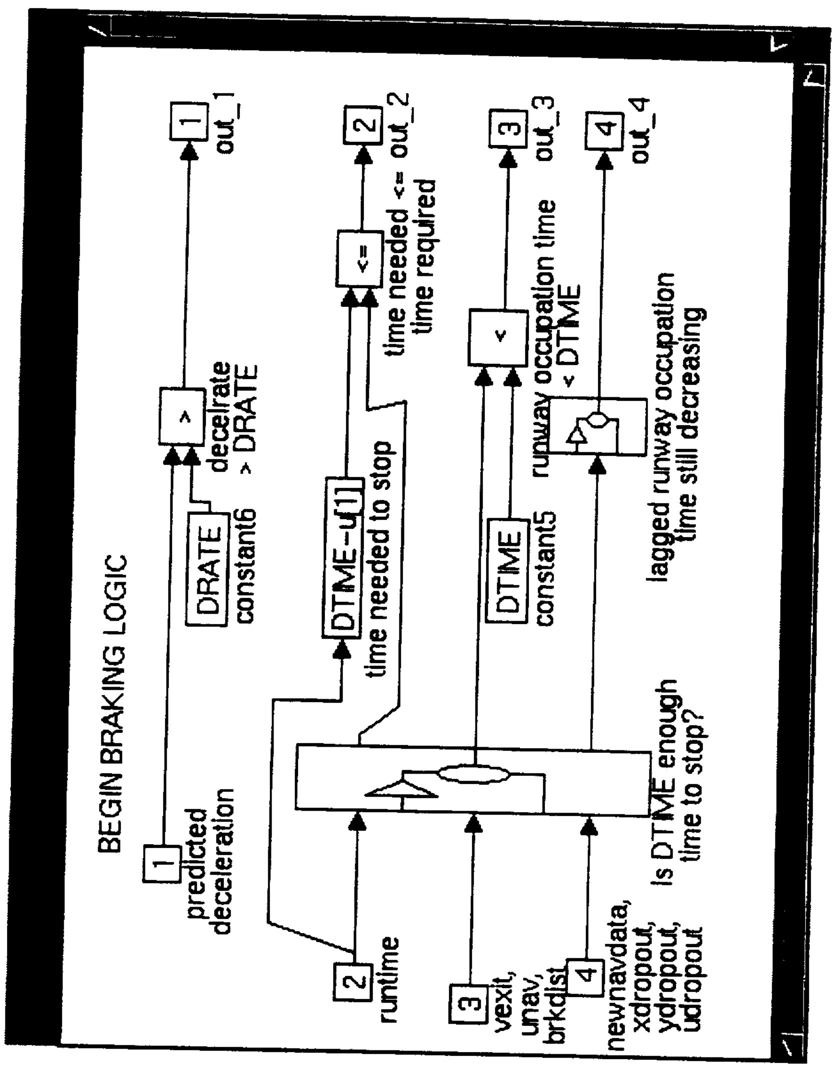
RATE LIMIT
1
in_1
Demux
Demux
decel Rate Limiter_
decel Rate Limiter
Mux
Mux2
1
out_1

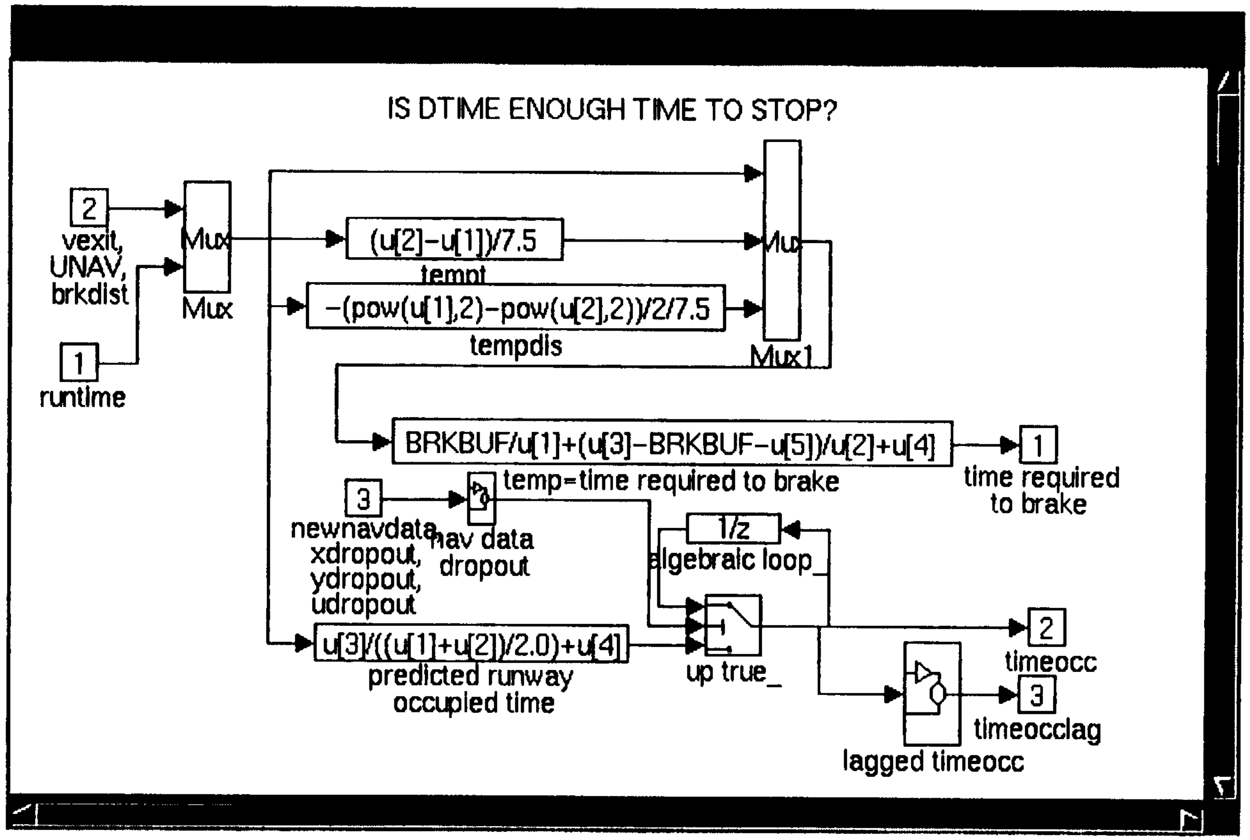


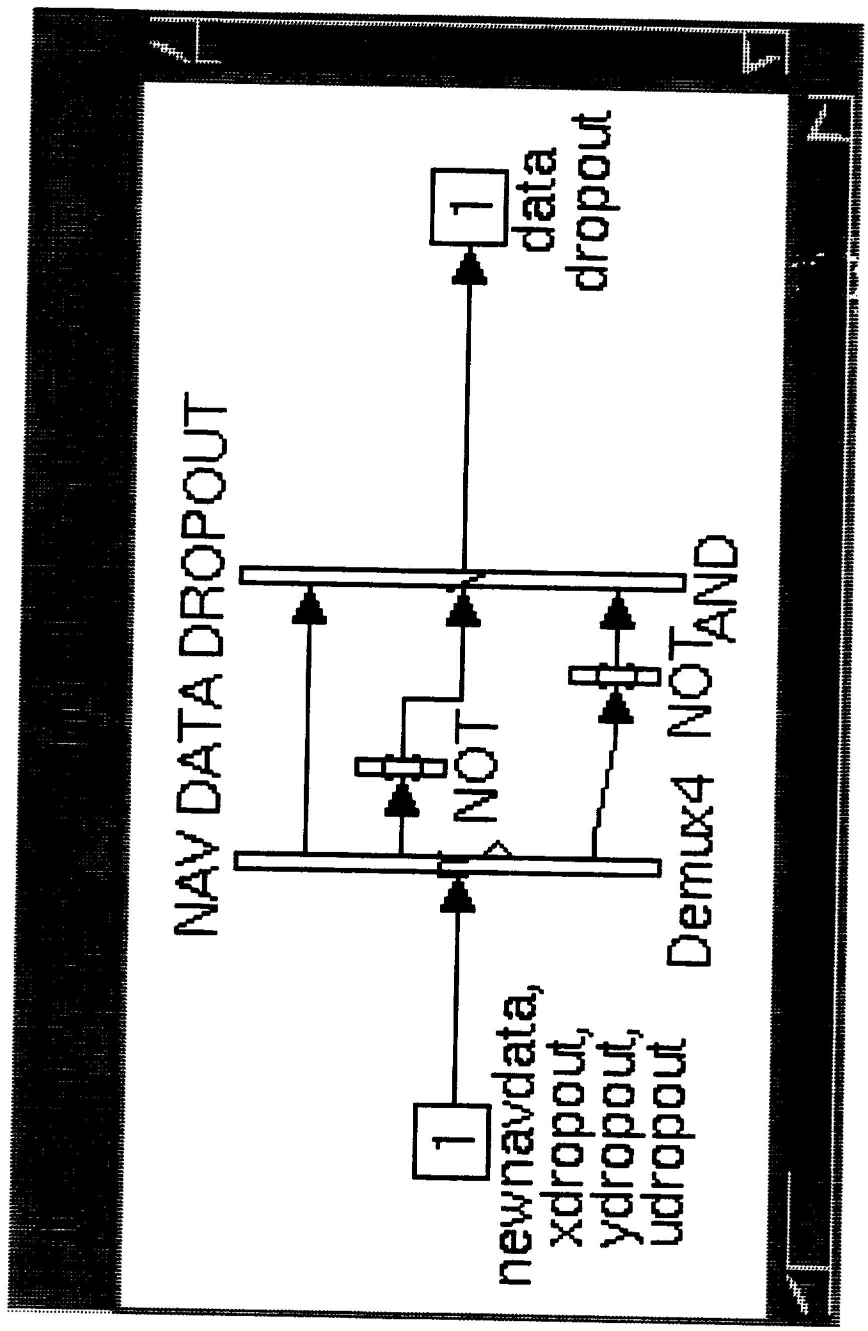
NON-LIMITED DECEL & LIMITED DECEL
4
vexit
3
U
6
brkdis
Mux
Mux
-(pow(u[1],2)-pow(u[2],2))/u[3]/2.
deceltemp
1
deceltemp_
9
predicted deceleration
7
1
newnavdata, begin braking logic
xdropout,
ydropout,
udropout
OR
reasons to begin decel
NOT
AND
5
on exit
OR
decelon
1/z
algebraic loop_
0
constant1
NOT1
up true
8
abortearly
2
xnav
decelcalc
3
belowvline
up true_
DECELCOMTH
To Workspace
2
decelset
0
constant2
NOT_
AND_
NOT
up true1
0
constant
1
constant
>
up true
decel
limit
decel
Rate
Limiter

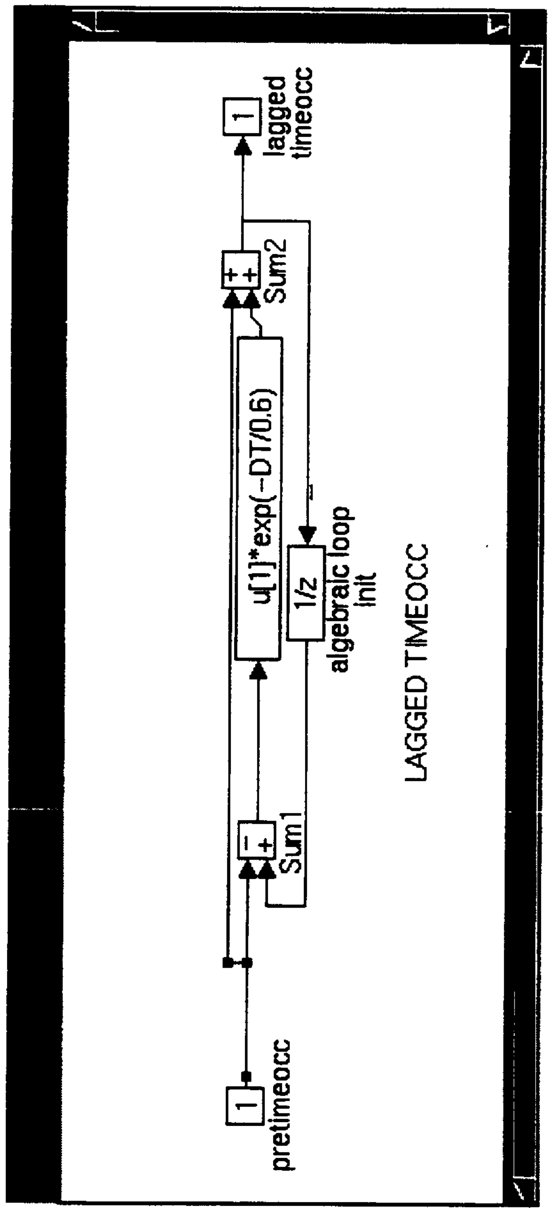

DECELCALC
vexit,
u,
brkdist
XNAV
decelon
Demux
Mux
Mux1
NOT
unit delay
1/z
algebraic
loop
delayed
abortearly
init to true
OR1
true up__
hold speed
profile inputs
u[2]-(u[5]+BRKBUF-u[4])*(u[2]-u[1])/u[3]
speed profile
Constant
Relational
Operator
Constant1
true up
Constant3
Saturation
Sum
Gain
0.1
OR
Constant2
1/s
Reset
Integrator
Sum1
decelcalc
100
aboveline if deceloff
true up1
belowvline

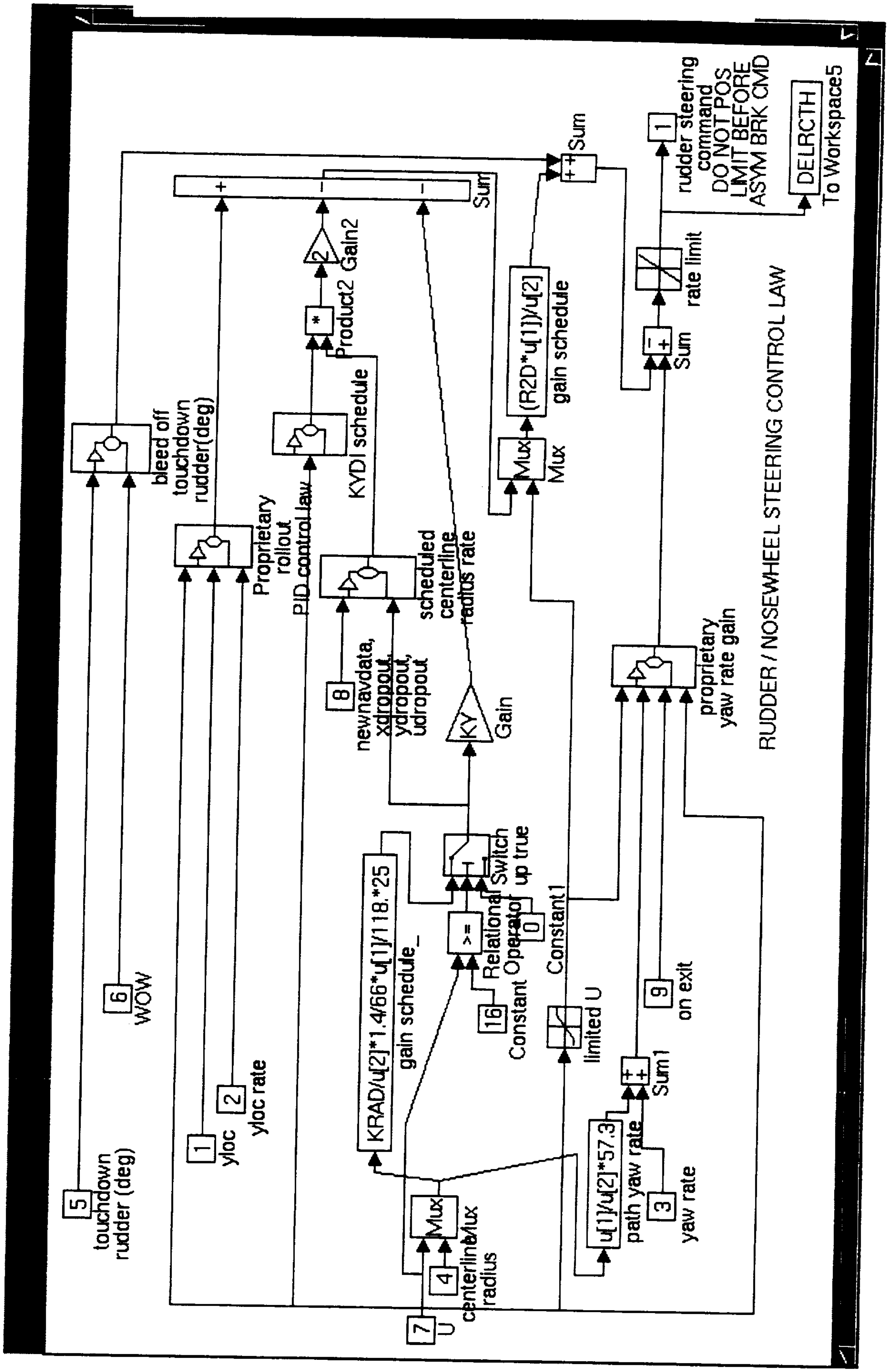

File
Edit
Options
Simulation
Style
Code
HOLD SPEED PROFILE INPUTS
1/z
init U
1
out_1
1/z
init brkdist
1
in_1
1/z
init XNAV
Mux2
Demux1

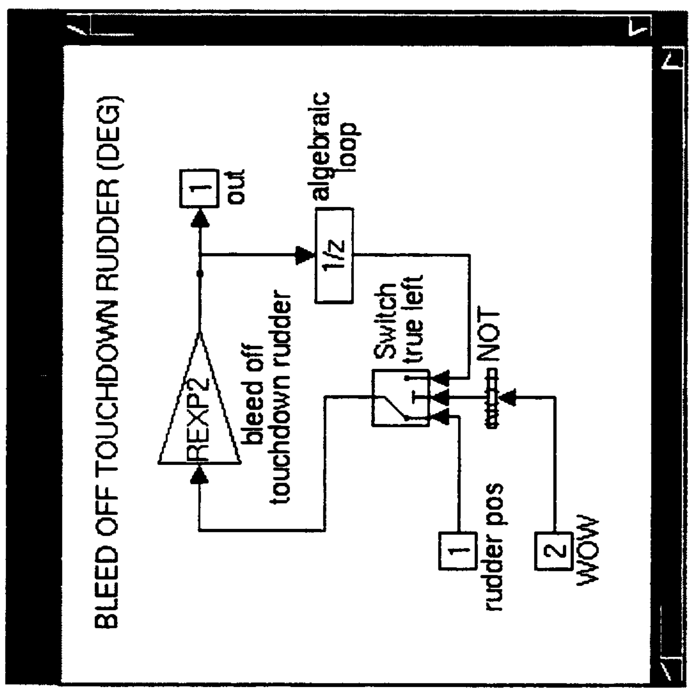

BEGIN BRAKING LOGIC
1
predicted deceleration
DRATE
constant6
>
decelrate > DRATE
1
out_1
2
runtime
DTIME-u[1]
time needed to stop
<=
time needed <= time required
2
out_2
3
vexit, unav, brkdist
DTIME
constant5
<
runway occupation time < DTIME
3
out_3
4
newnavdata, xdropout, ydropout, udropout
Is DTIME enough time to stop?
lagged runway occupation time still decreasing
4
out_4

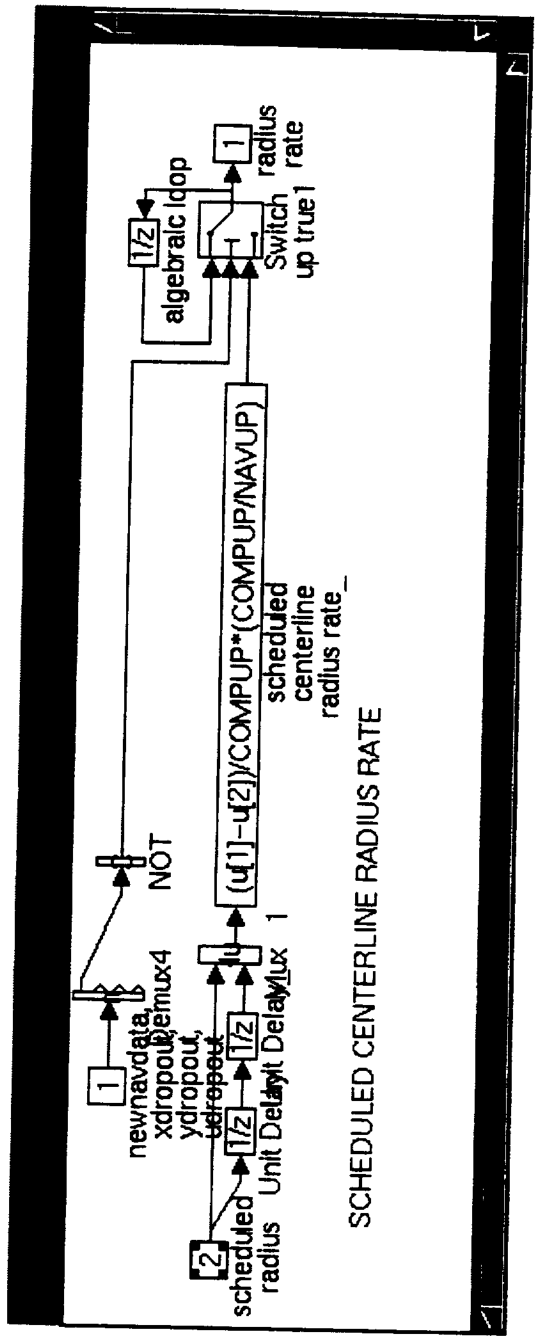

IS DTIME ENOUGH TIME TO STOP?
2
vexit,
UNAV,
brkdist
1
runtime
Mux
Mux
(u[2]-u[1])/7.5
tempt
-(pow(u[1],2)-pow(u[2],2))/2/7.5
tempdis
Mux
Mux1
BRKBUF/u[1]+(u[3]-BRKBUF-u[5])/u[2]+u[4]
temp=time required to brake
1
time required
to brake
3
newnavdata,
xdropout,
ydropout,
udropout
nav data
dropout
1/z
algebraic loop_
up true_
u[3]/((u[1]+u[2])/2.0)+u[4]
predicted runway
occupied time
2
timeocc
3
timeocclag
lagged timeocc

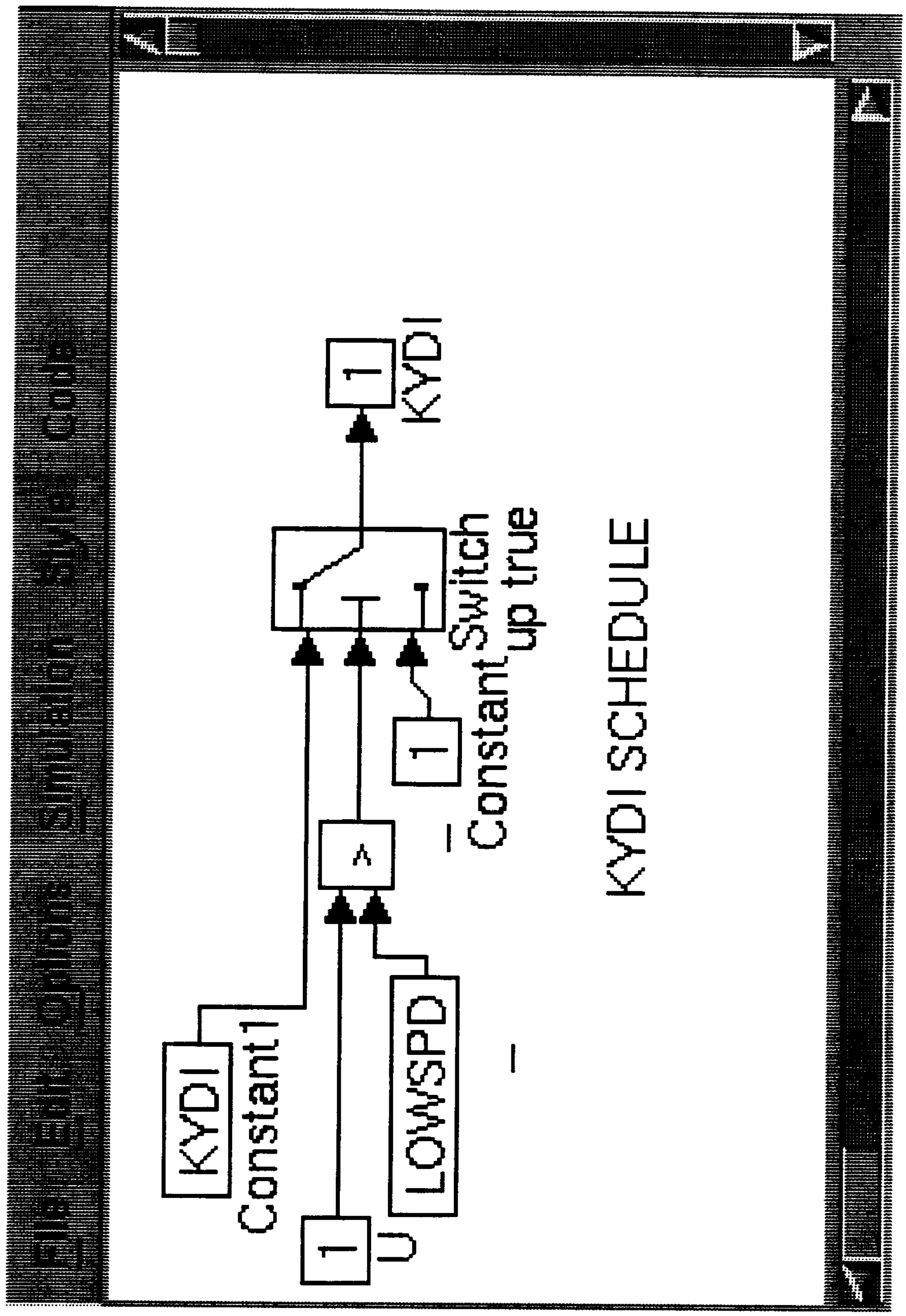

NAV DATA DROPOUT
1
newnavdata,
xdropout,
ydropout,
udropout
Demux4
NOT
NOT
AND
1
data
dropout

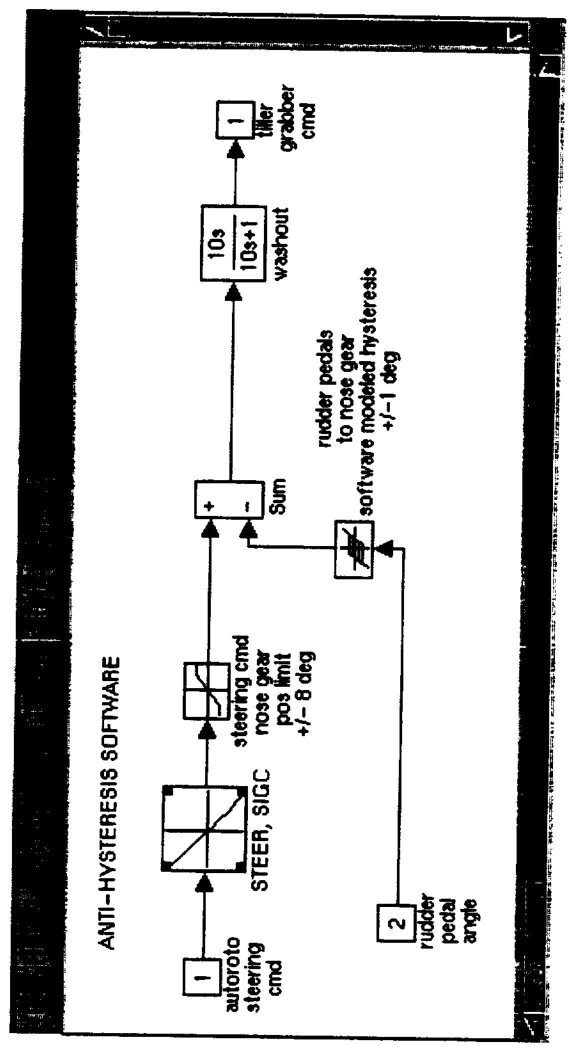

1
pretimeocc
Sum1
u[1]*exp(-DT/0.6)
1/z
algebraic loop
Init
Sum2
1
lagged
timeocc
LAGGED TIMEOCC

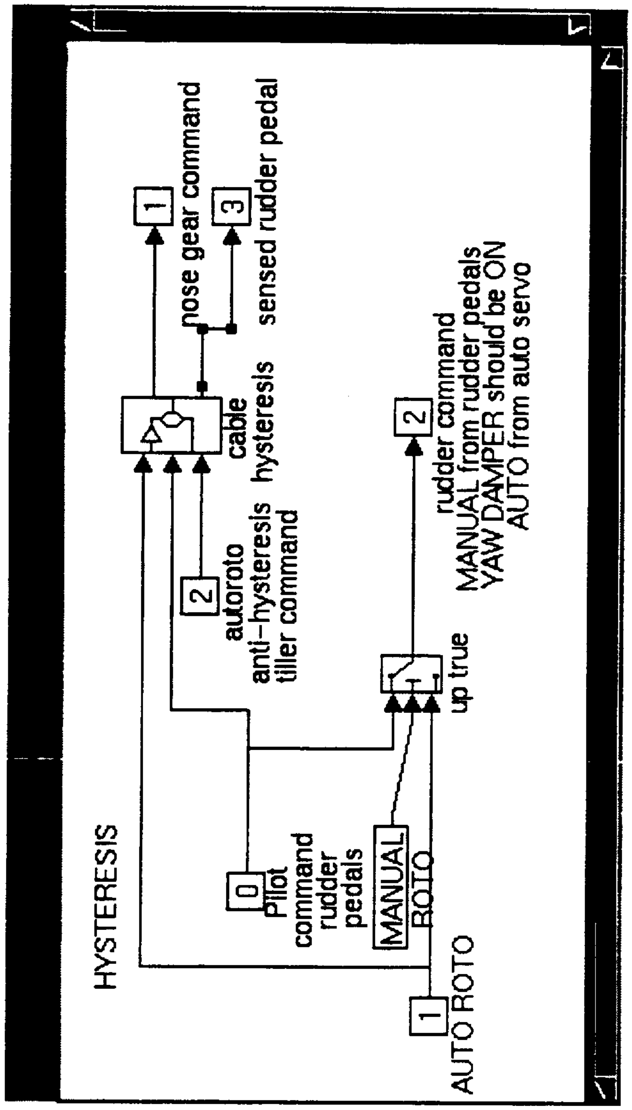

RUDDER / NOSEWHEEL STEERING CONTROL LAW
touchdown rudder (deg)
WOW
yloc
yloc rate
centerline radius
U
Mux
KRAD/u[2]*1.4/66*u[1]/118.*25
gain schedule_
Relational Operator
Constant
Constant1
Switch up true
limited U
Sum1
path yaw rate
u[1]/u[2]*57.3
yaw rate
on exit
newnavdata, xdropout, ydropout, udropout
Gain
KY
bleed off touchdown rudder(deg)
Proprietary rollout PID control law
KYDI schedule
scheduled centerline radius rate
Product2
Gain2
Sum
(R2D*u[1])/u[2]
gain schedule
proprietary yaw rate gain
rate limit
rudder steering command DO NOT POS LIMIT BEFORE ASYM BRK CMD
DELRCTH
To Workspace5

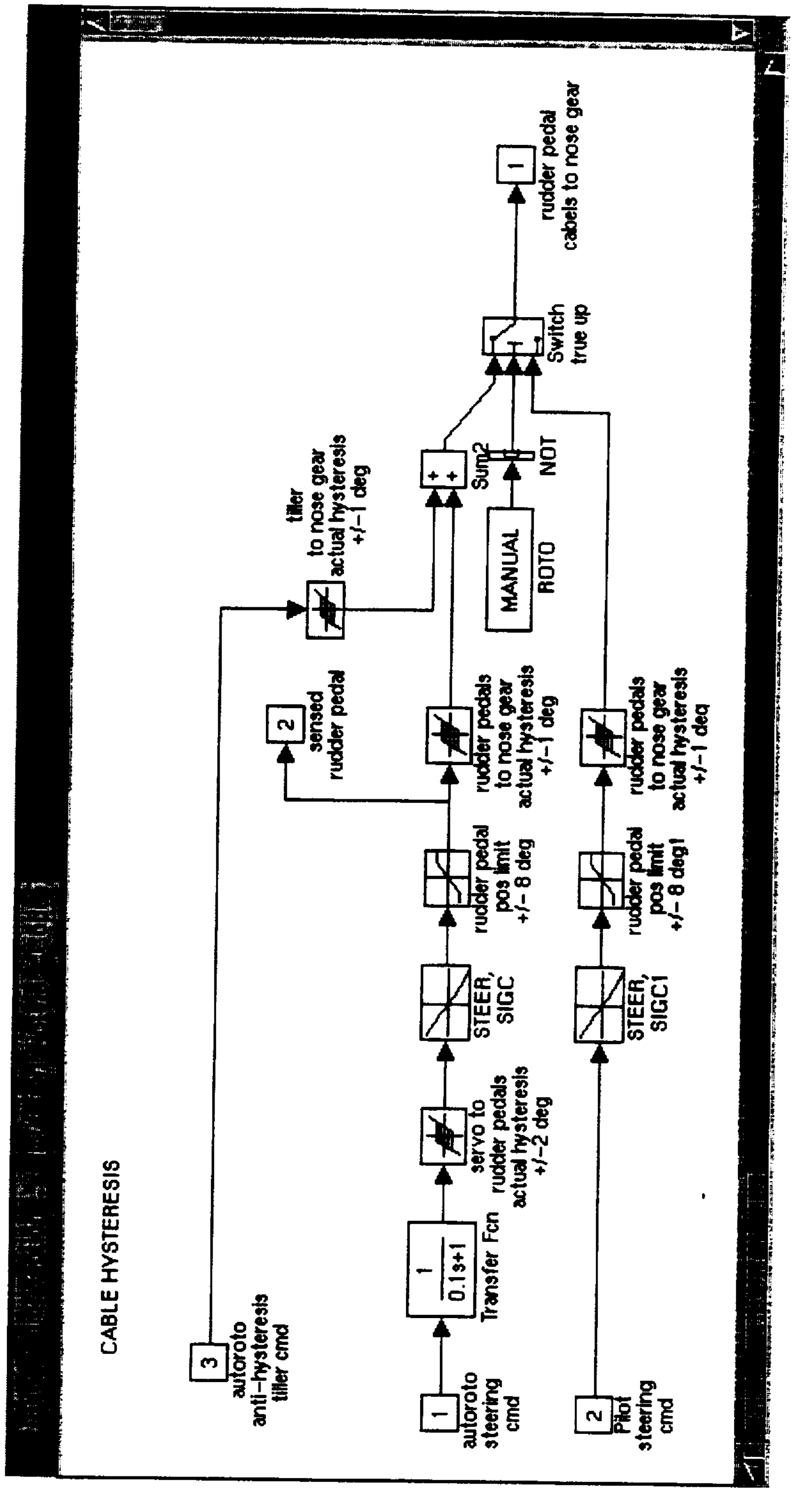

BLEED OFF TOUCHDOWN RUDDER (DEG)
1
out
algebraic
loop
1/z
REXP2
bleed off
touchdown rudder
Switch
true left
NOT
1
rudder pos
2
WOW

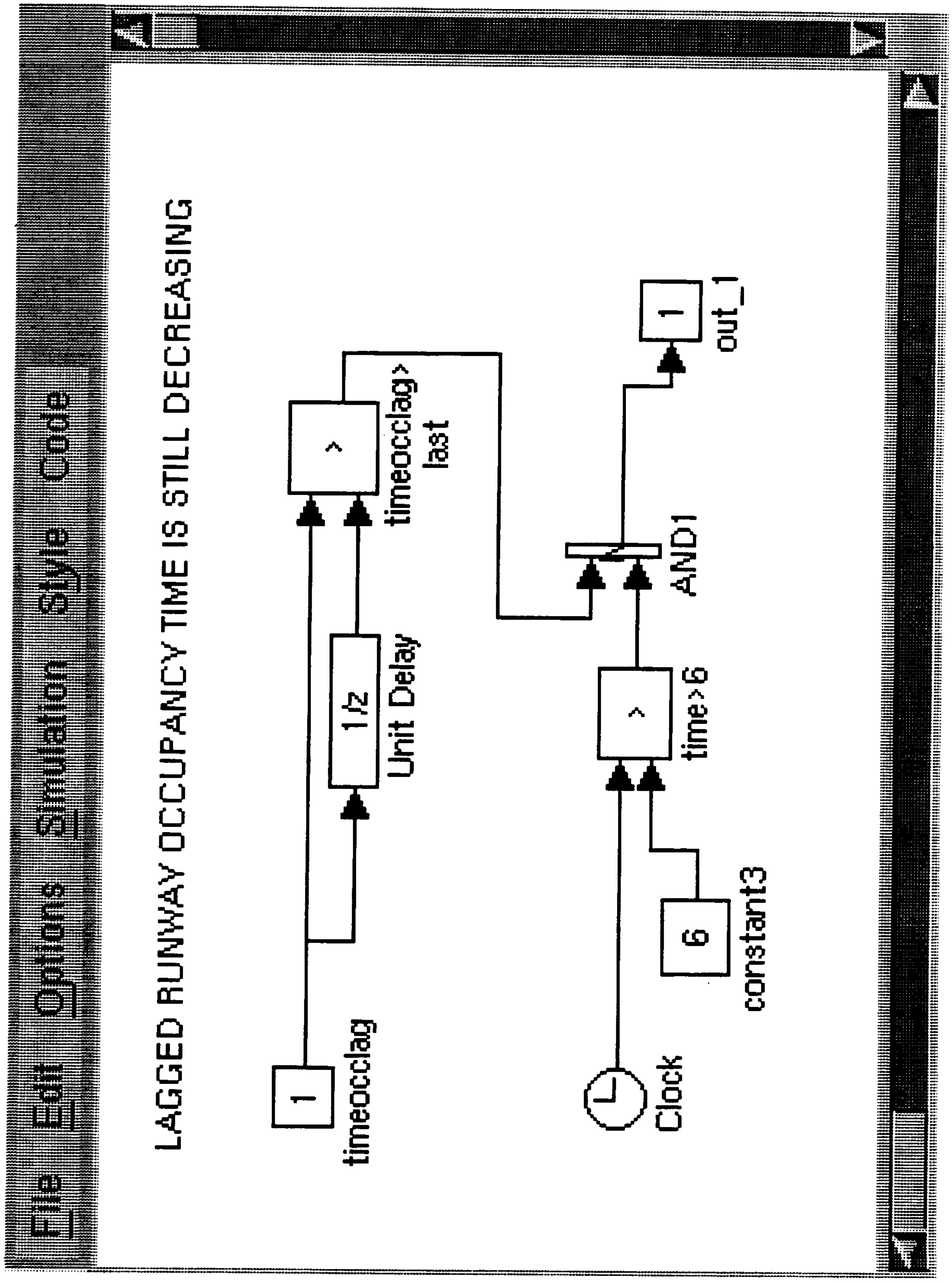

SCHEDULED CENTERLINE RADIUS RATE
2
scheduled radius
1/z
Unit Delay
1/z
Unit Delay
Mux
1
newnavdata, xdropout, ydropout, udropout
Demux4
NOT
(u[1]-u[2])/COMPUP*(COMPUP/NAVUP)
scheduled centerline radius rate_
1/z
algebraic loop
Switch up true1
1
radius rate

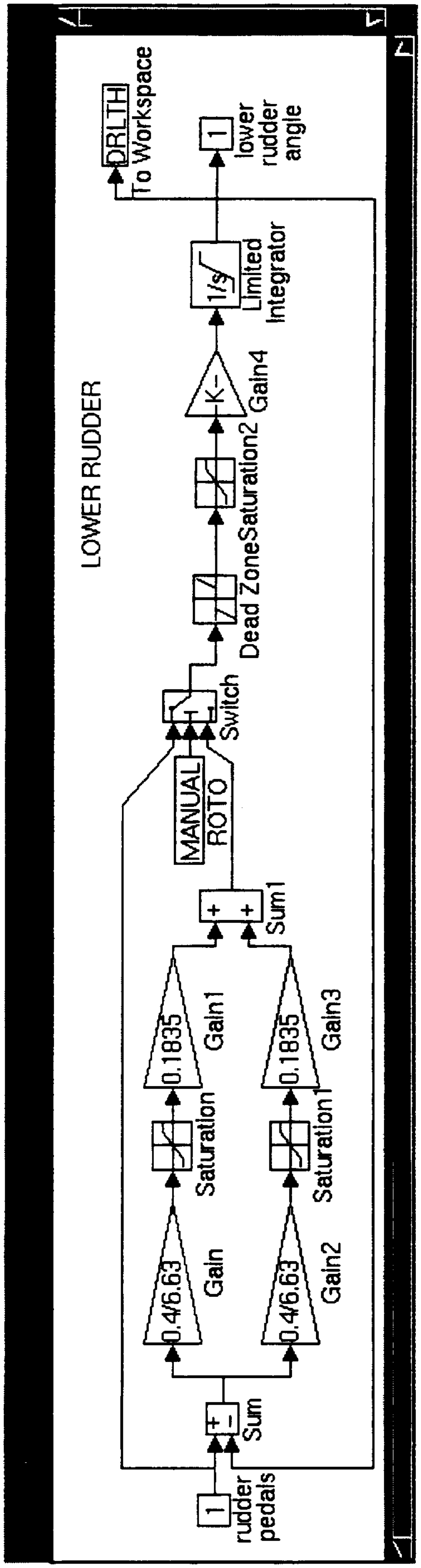

File
Edit
Options
Simulation
Style
Code
KYDI
Constant1
1
U
LOWSPD
>
1
Constant
Switch
up true
1
KYDI
KYDI SCHEDULE

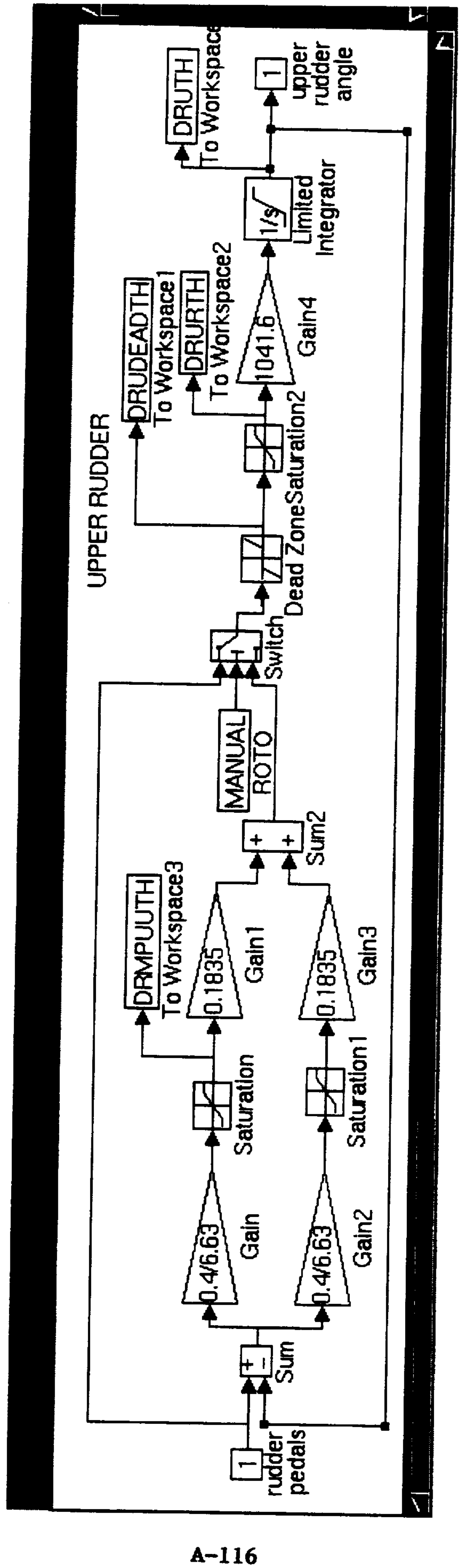

ANTI-HYSTERESIS SOFTWARE
1
autoroto steering cmd
STEER, SIGC
steering cmd nose gear pos limit +/- 8 deg
+
-
Sum
rudder pedals to nose gear software modeled hysteresis +/-1 deg
2
rudder pedal angle
10s
10s+1
washout
1
tiller grabber cmd

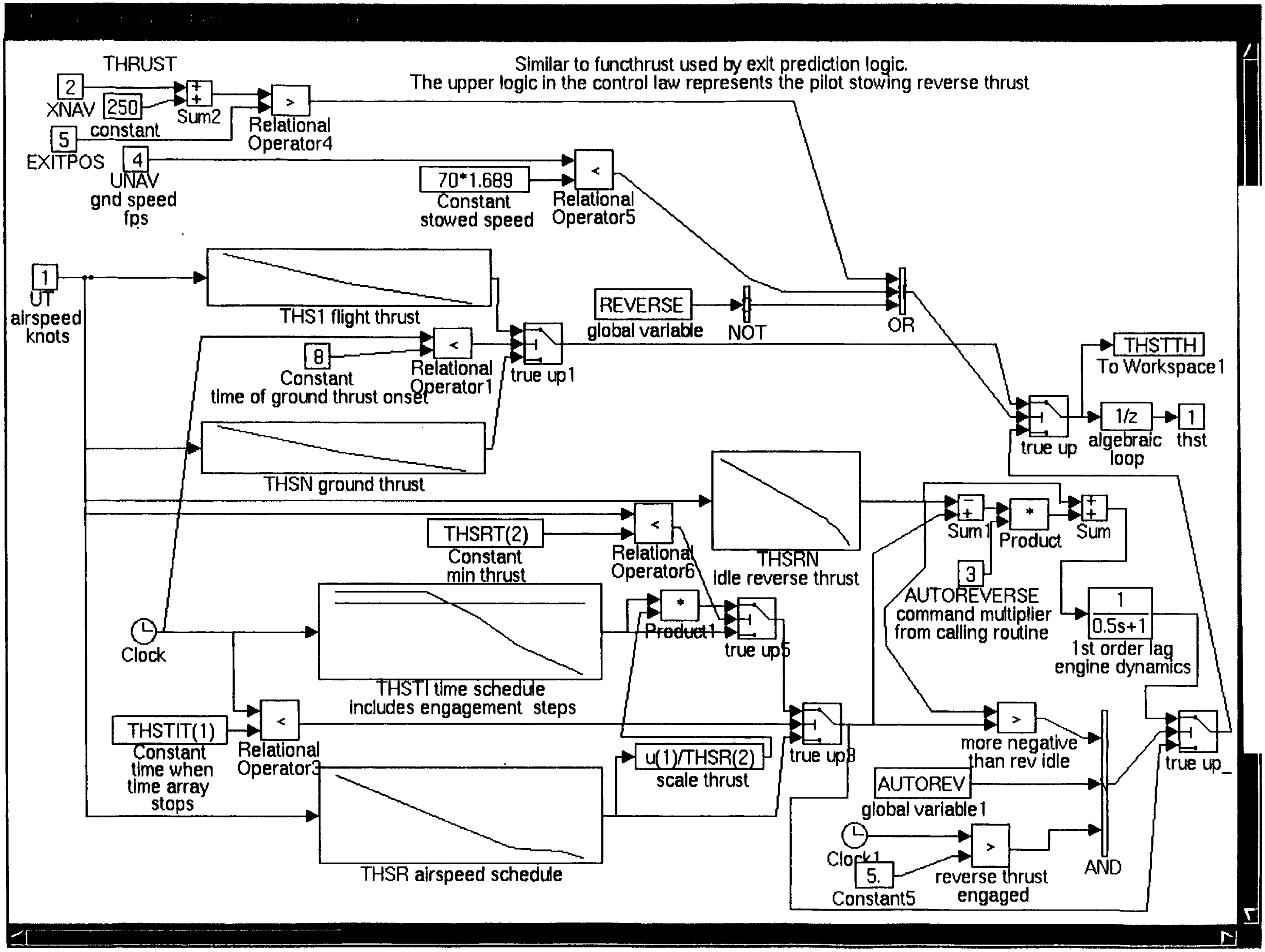

HYSTERESIS
1
AUTO ROTO
0
Pilot
command
rudder
pedals
MANUAL
ROTO
up true
2
autoroto
anti-hysteresis
tiller command
cable
hysteresis
1
nose gear command
3
sensed rudder pedal
2
rudder command
MANUAL from rudder pedals
YAW DAMPER should be ON
AUTO from auto servo


CABLE HYSTERESIS
1
autoroto steering cmd
2
Pilot steering cmd
3
autoroto anti-hysteresis tiller cmd
1
0.1s+1
Transfer Fcn
servo to rudder pedals actual hysteresis +/-2 deg
STEER, SIGC
STEER, SIGC1
rudder pedal pos limit +/- 8 deg
rudder pedal pos limit +/- 8 deg1
rudder pedals to nose gear actual hysteresis +/-1 deg
rudder pedals to nose gear actual hysteresis +/-1 deq
2
sensed rudder pedal
tiller to nose gear actual hysteresis +/-1 deg
MANUAL
ROTO
Sum2
NOT
Switch true up
1
rudder pedal cabels to nose gear

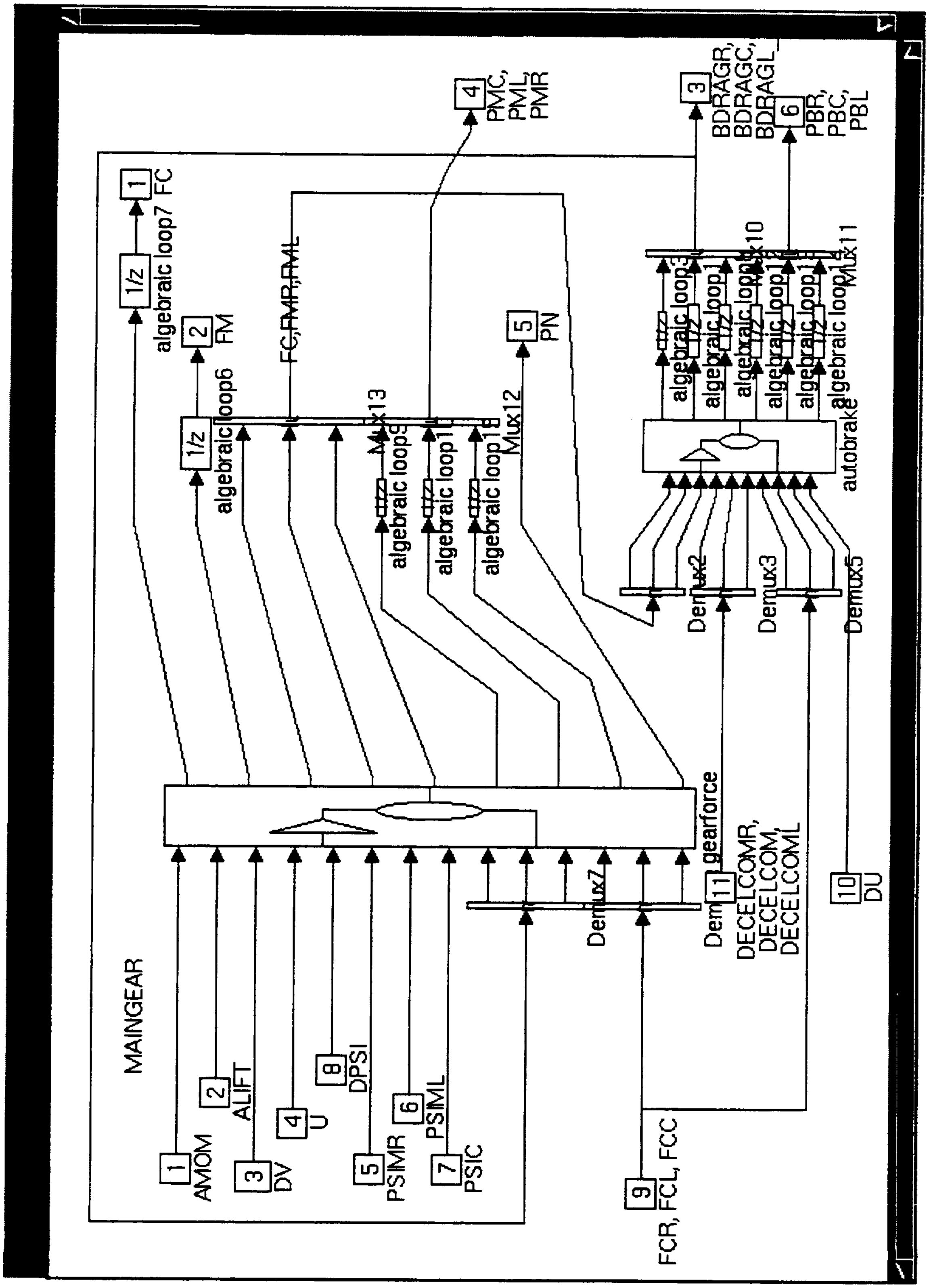

File Edit Options Simulation Style Code
LAGGED RUNWAY OCCUPANCY TIME IS STILL DECREASING
1
timeocclag
1/z
Unit Delay
>
timeocclag>
last
AND1
1
out_1
Clock
6
constant3
>
time>6

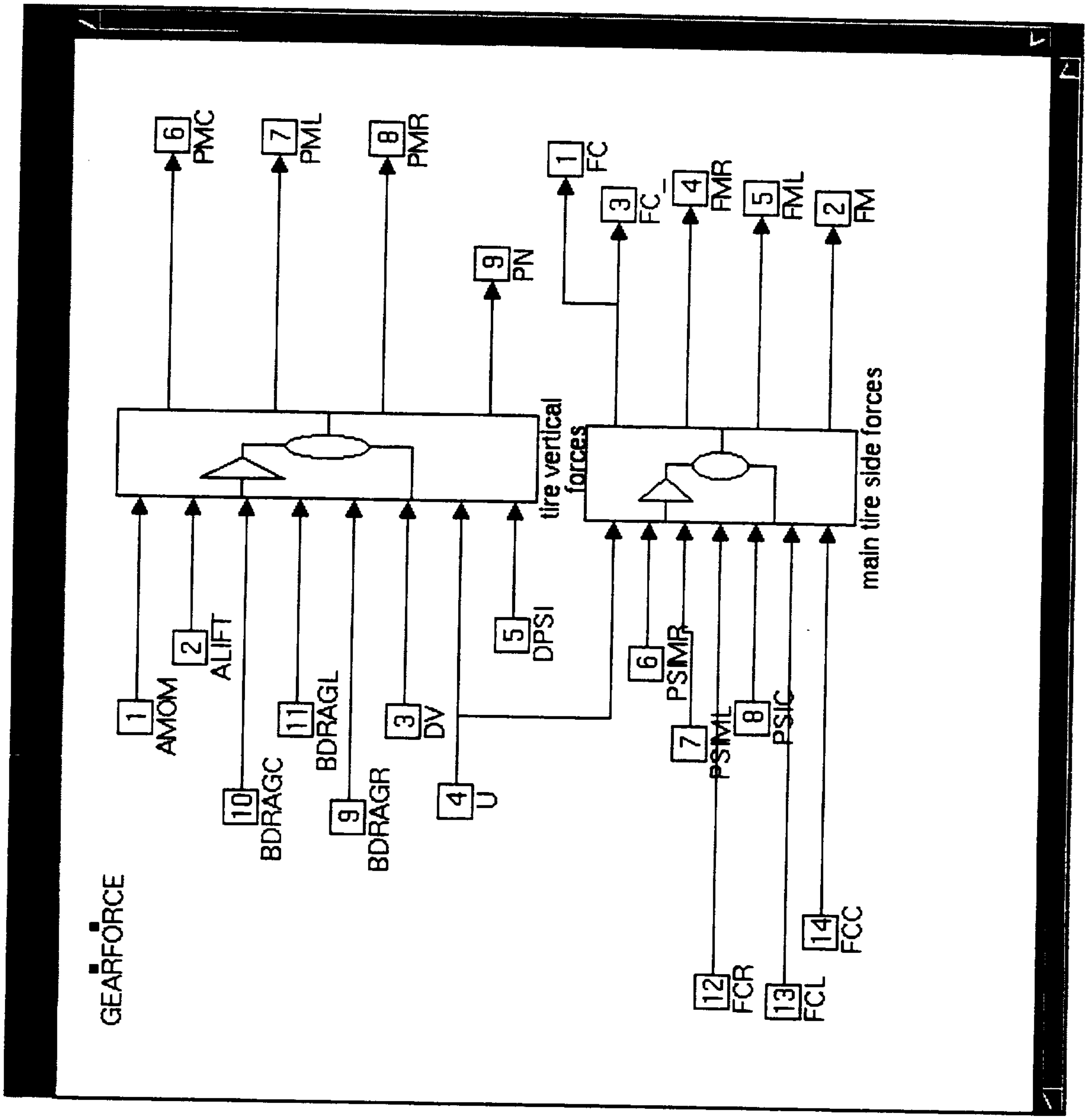

LOWER RUDDER
1
rudder pedals
Sum
0.4/6.63
Gain
0.4/6.63
Gain2
Saturation
Saturation1
0.1835
Gain1
0.1835
Gain3
Sum1
MANUAL
ROTO
Switch
Dead Zone
Saturation2
-K-
Gain4
1/s
Limited Integrator
1
lower rudder angle
DRLTH
To Workspace

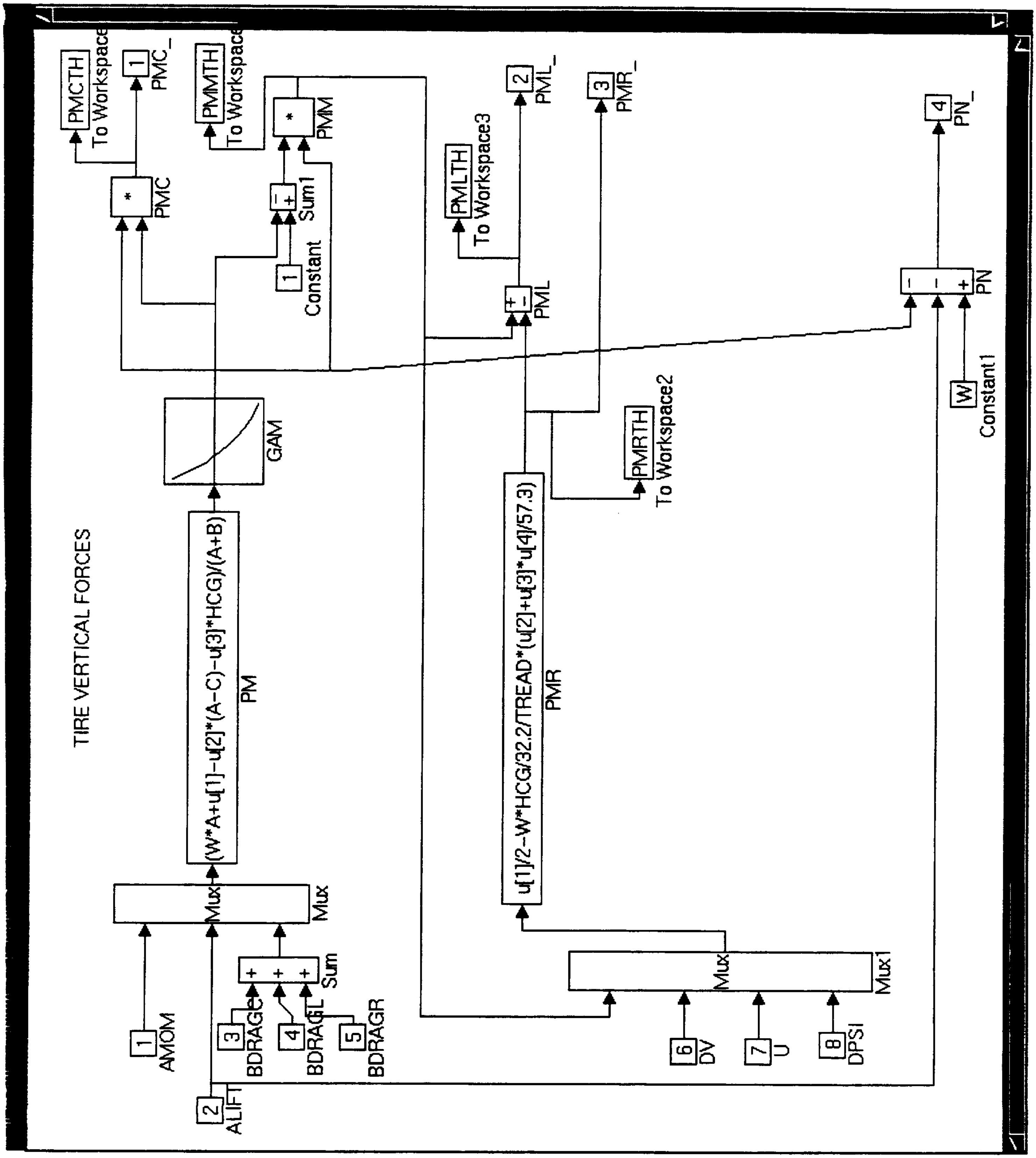

UPPER RUDDER
1
rudder pedals
Sum
0.4/6.63
Gain
0.4/6.63
Gain2
Saturation
Saturation1
DRMPUUTH
To Workspace3
0.1835
Gain1
0.1835
Gain3
Sum2
MANUAL
ROTO
Switch
Dead Zone
Saturation2
DRUDEADTH
To Workspace1
DRURTH
To Workspace2
1041.6
Gain4
1/s
Limited Integrator
DRUTH
To Workspace
1
upper rudder angle

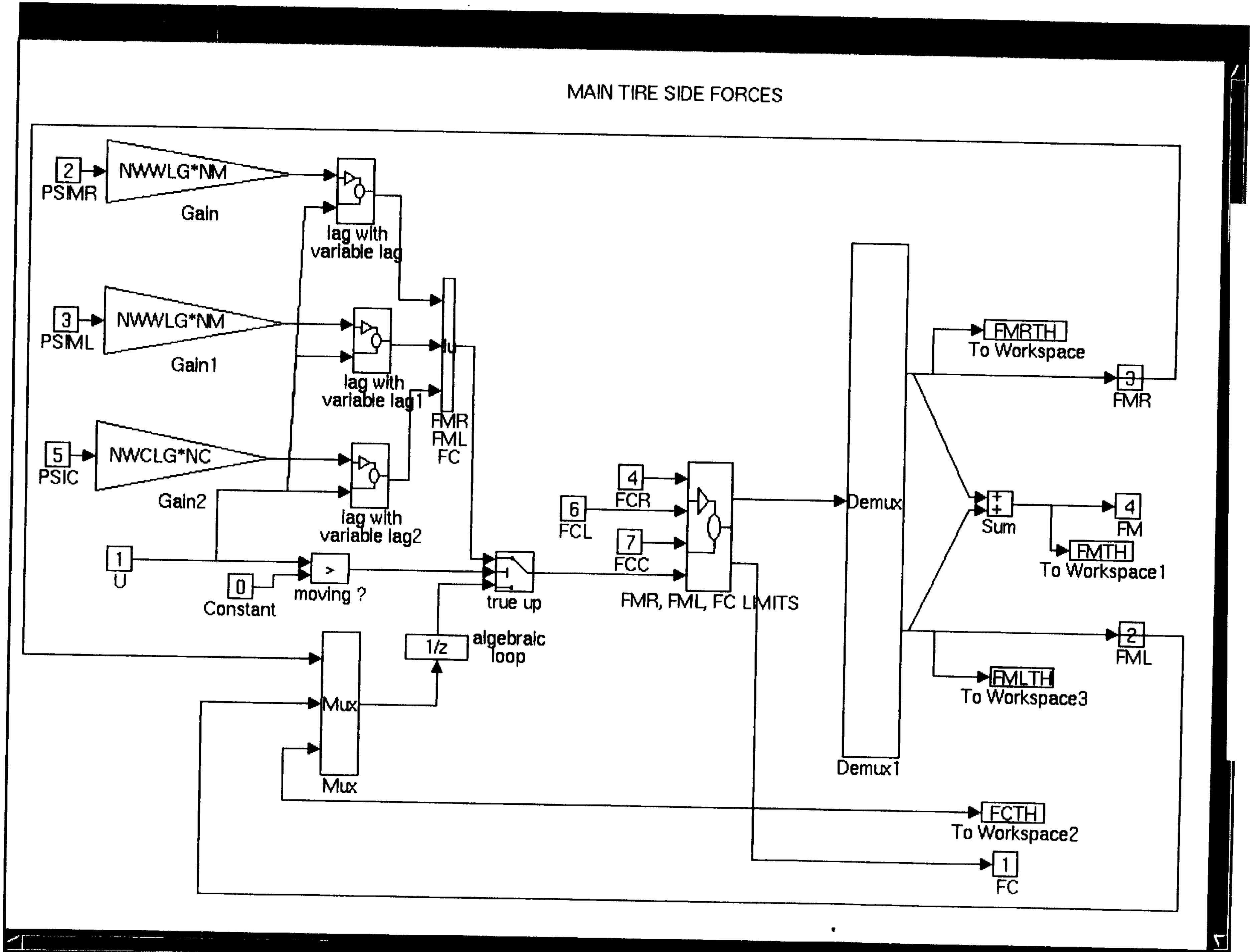


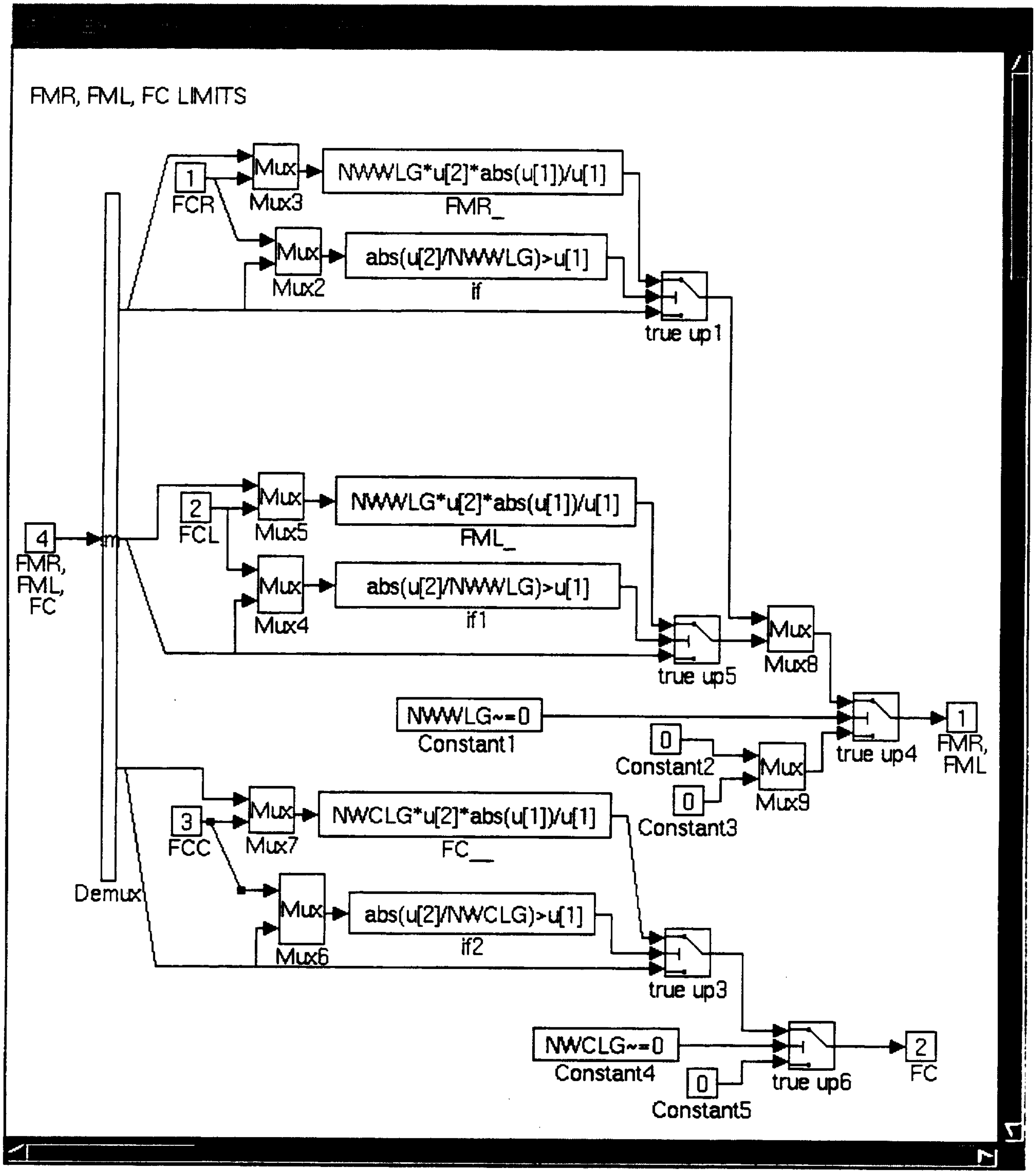
Similar to functhrust used by exit prediction logic.
The upper logic in the control law represents the pilot stowing reverse thrust
THRUST
2
XNAV
250
constant
Sum2
Relational
Operator4
5
EXITPOS
4
UNAV
gnd speed
fps
70*1.689
Constant
stowed speed
Relational
Operator5
1
UT
airspeed
knots
THS1 flight thrust
REVERSE
global variable
NOT
OR
8
Constant
time of ground thrust onset
Relational
Operator1
true up1
THSTTH
To Workspace1
true up
1/z
algebraic
loop
1
thst
THSN ground thrust
THSRT(2)
Constant
min thrust
Relational
Operator6
THSRN
Idle reverse thrust
Sum1
Product
Sum
3
AUTOREVERSE
command multiplier
from calling routine
1
0.5s+1
1st order lag
engine dynamics
Clock
Product1
true up5
THSTI time schedule
includes engagement steps
THSTIT(1)
Constant
time when
time array
stops
Relational
Operator3
true up8
u(1)/THSR(2)
scale thrust
more negative
than rev idle
true up_
AUTOREV
global variable1
Clock1
5.
Constant5
reverse thrust
engaged
AND
THSR airspeed schedule

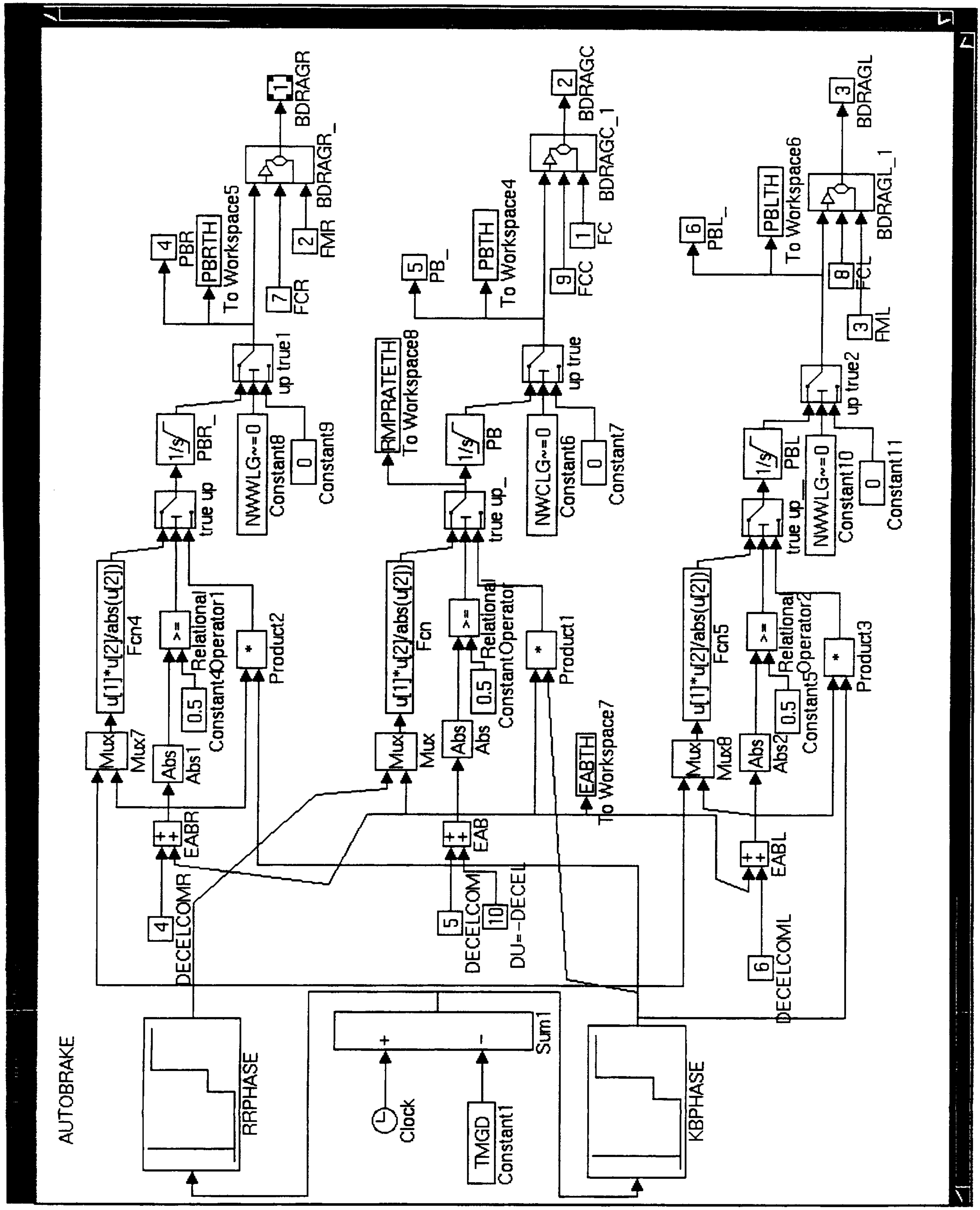

MAINGEAR
1 AMOM
2 ALIFT
3 DV
4 U
5 PSIMR
6 PSIML
7 PSIC
8 DPSI
9 FCR, FCL, FCC
10 DU
11 DECELCOMR, DECELCOM, DECELCOML
gearforce
Demux7
Demux2
Demux3
Demux5
autobrake
Mux10
Mux11
Mux12
Mux13
1/z
algebraic loop6
algebraic loop7
1 FC
2 FM
FC,FMR,FML
5 PN
4 PMC, PML, PMR
3 BDRAGR, BDRAGC, BDRAGL
6 PBR, PBC, PBL

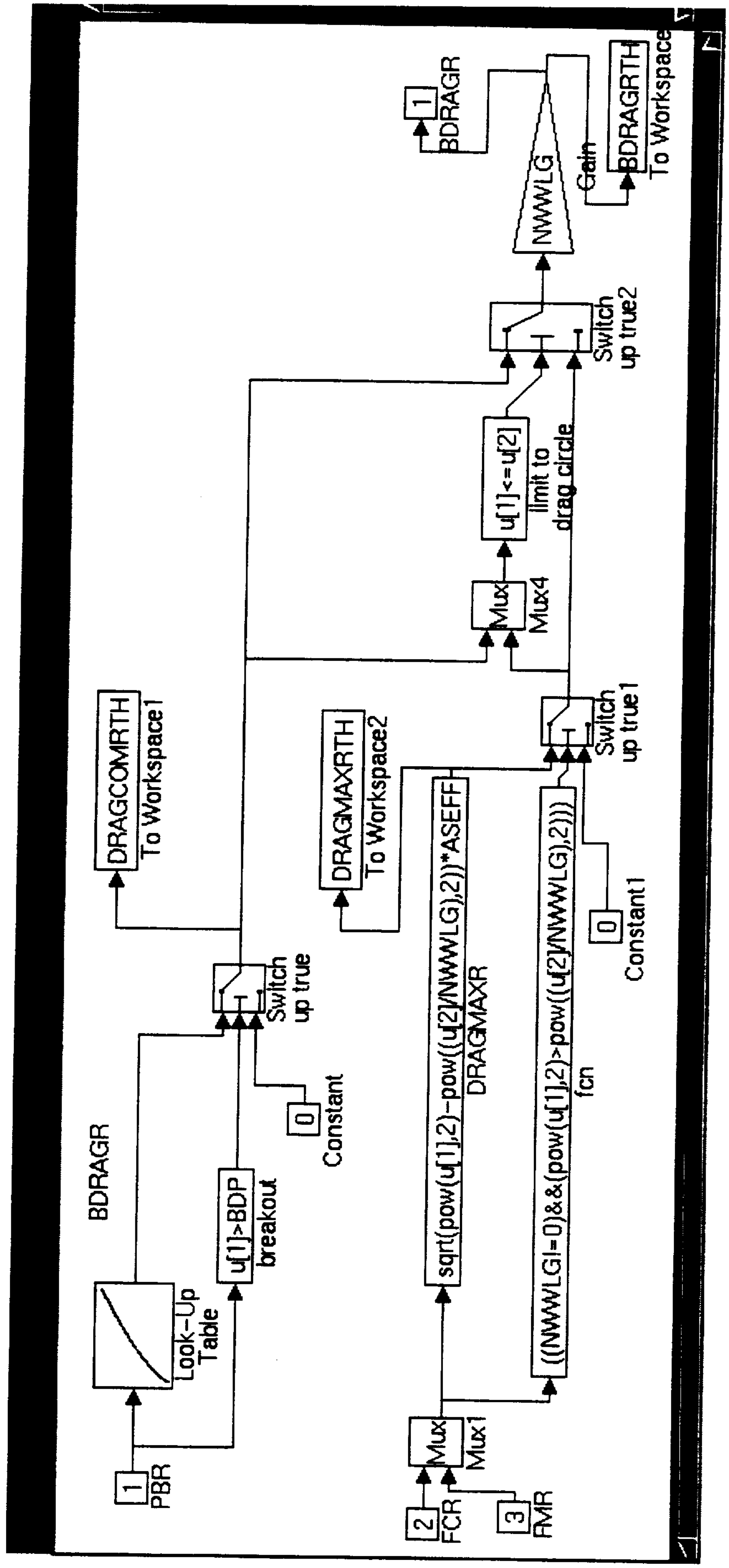

GEARFORCE
1 AMOM
2 ALIFT
10 BDRAGC
11 BDRAGL
9 BDRAGR
3 DV
4 U
5 DPSI
6 PSIMR
7 PSIML
8 PSIC
12 FCR
13 FCL
14 FCC
tire vertical forces
main tire side forces
6 PMC
7 PML
8 PMR
9 PN
1 FC
3 FC_
4 FMR
5 FML
2 FM

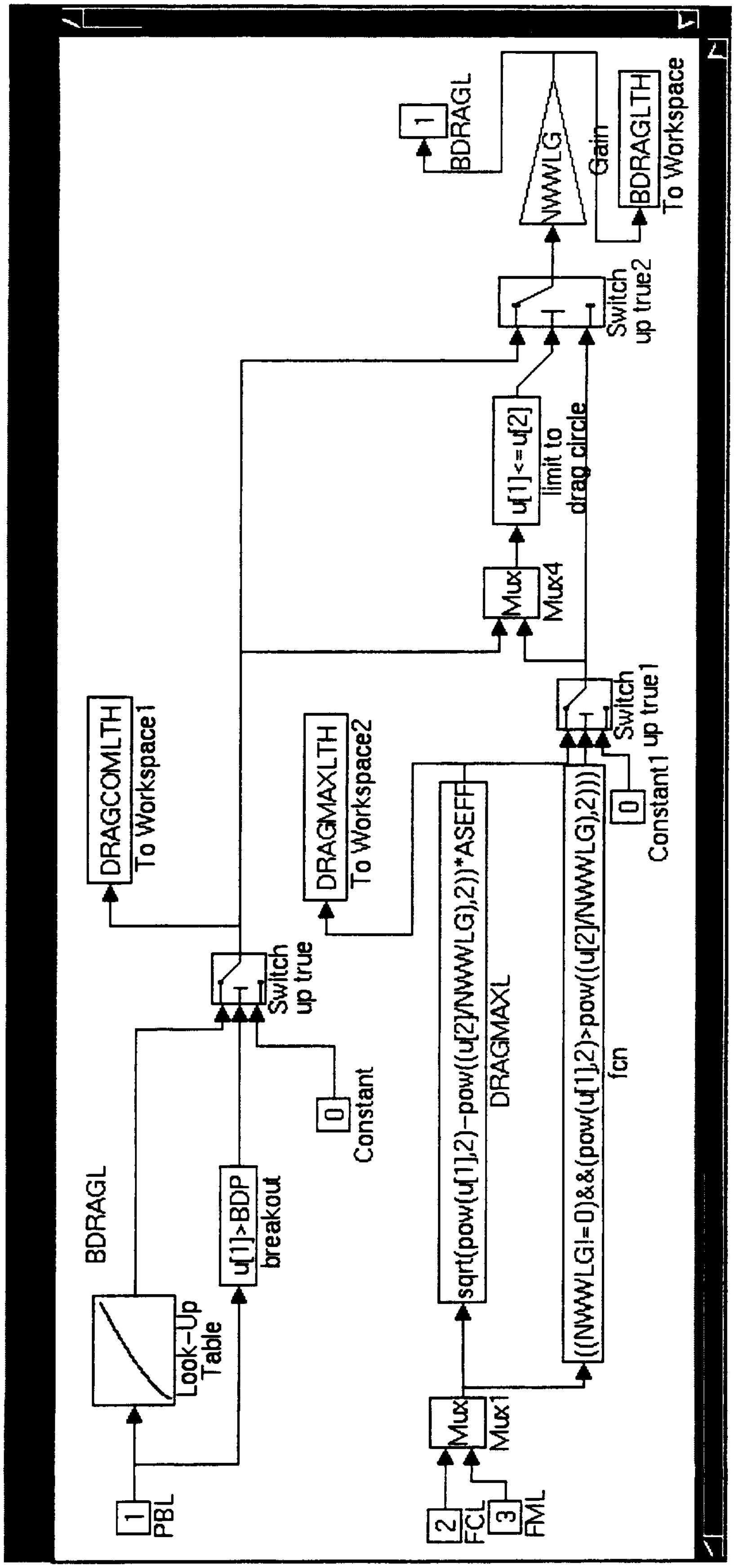

TIRE VERTICAL FORCES
AMOM
ALIFT
BDRAGC
BDRAGL
BDRAGR
DV
U
DPSI
Sum
Mux
Mux1
(W*A+u[1]-u[2]*(A-C)-u[3]*HCG)/(A+B)
PM
GAM
u[1]/2-W*HCG/32.2/TREAD*(u[2]+u[3]*u[4]/57.3)
PMR
PMC
PMCTH
To Workspace
PMC_
Constant
Sum1
PMM
PMMTH
To Workspace
PML
PMLTH
To Workspace3
PML_
PMR_
PMRTH
To Workspace2
PN
PN_
W
Constant1


MAIN TIRE SIDE FORCES
2
PSMR
NWWLG*NM
Gain
lag with variable lag
3
PSML
NWWLG*NM
Gain1
lag with variable lag1
5
PSIC
NWCLG*NC
Gain2
lag with variable lag2
Mu
FMR
FML
FC
1
U
0
Constant
>
moving ?
true up
1/z
algebraic loop
Mux
Mux
4
FCR
6
FCL
7
FCC
FMR, FML, FC LIMITS
Demux
Demux1
FMRTH
To Workspace
3
FMR
+
+
Sum
4
FM
FMTH
To Workspace1
2
FML
FMLTH
To Workspace3
FCTH
To Workspace2
1
FC

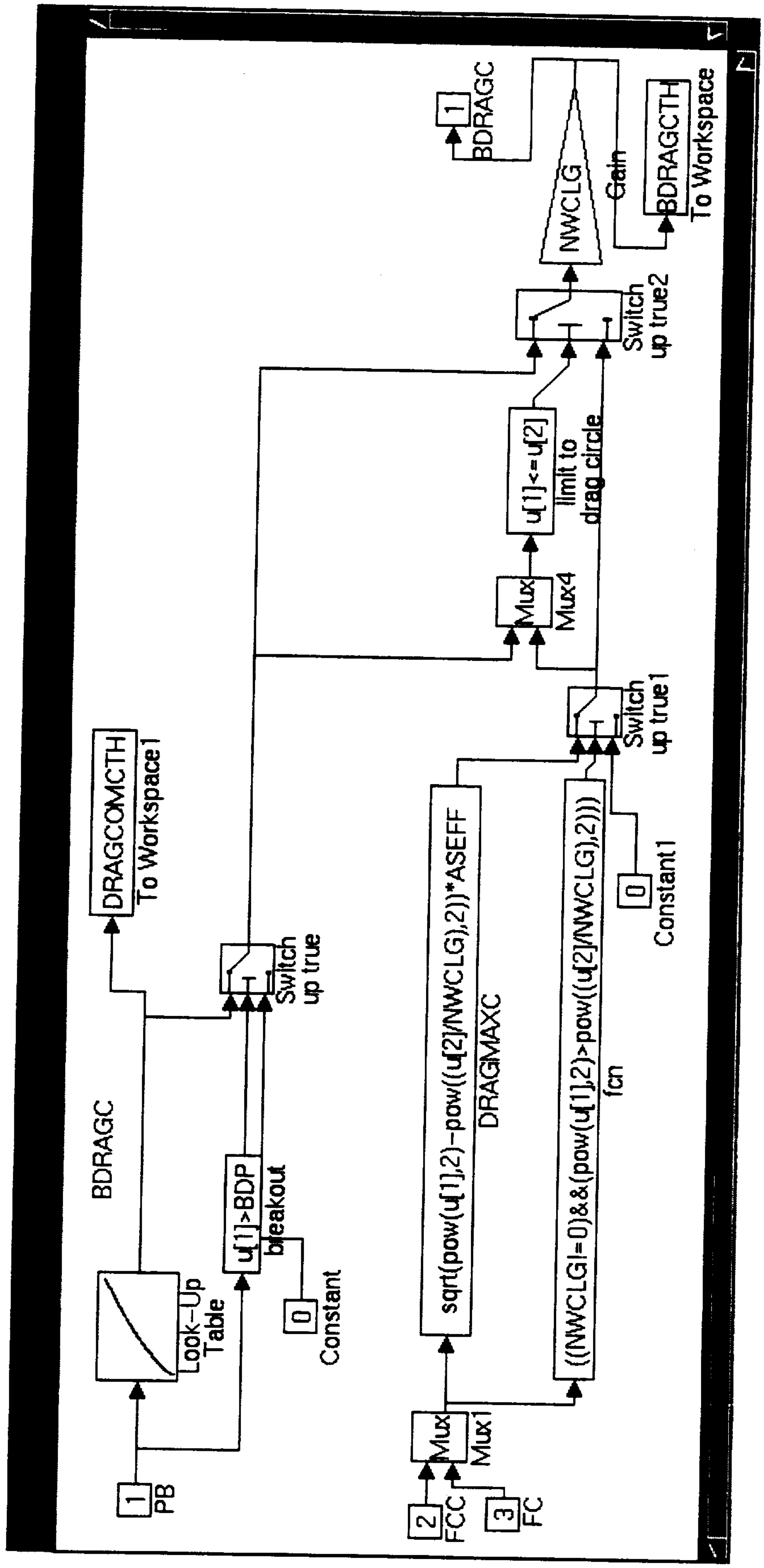

FMR, FML, FC LIMITS
1
FCR
Mux
Mux3
NWWLG*u[2]*abs(u[1])/u[1]
FMR_
Mux
Mux2
abs(u[2]/NWWLG)>u[1]
if
true up1
2
FCL
Mux
Mux5
NWWLG*u[2]*abs(u[1])/u[1]
FML_
Mux
Mux4
abs(u[2]/NWWLG)>u[1]
if1
true up5
Mux
Mux8
4
FMR,
FML,
FC
NWWLG~=0
Constant1
0
Constant2
Mux
Mux9
0
Constant3
true up4
1
FMR,
FML
3
FCC
Mux
Mux7
NWCLG*u[2]*abs(u[1])/u[1]
FC__
Demux
Mux
Mux6
abs(u[2]/NWCLG)>u[1]
if2
true up3
NWCLG~=0
Constant4
0
Constant5
true up6
2
FC

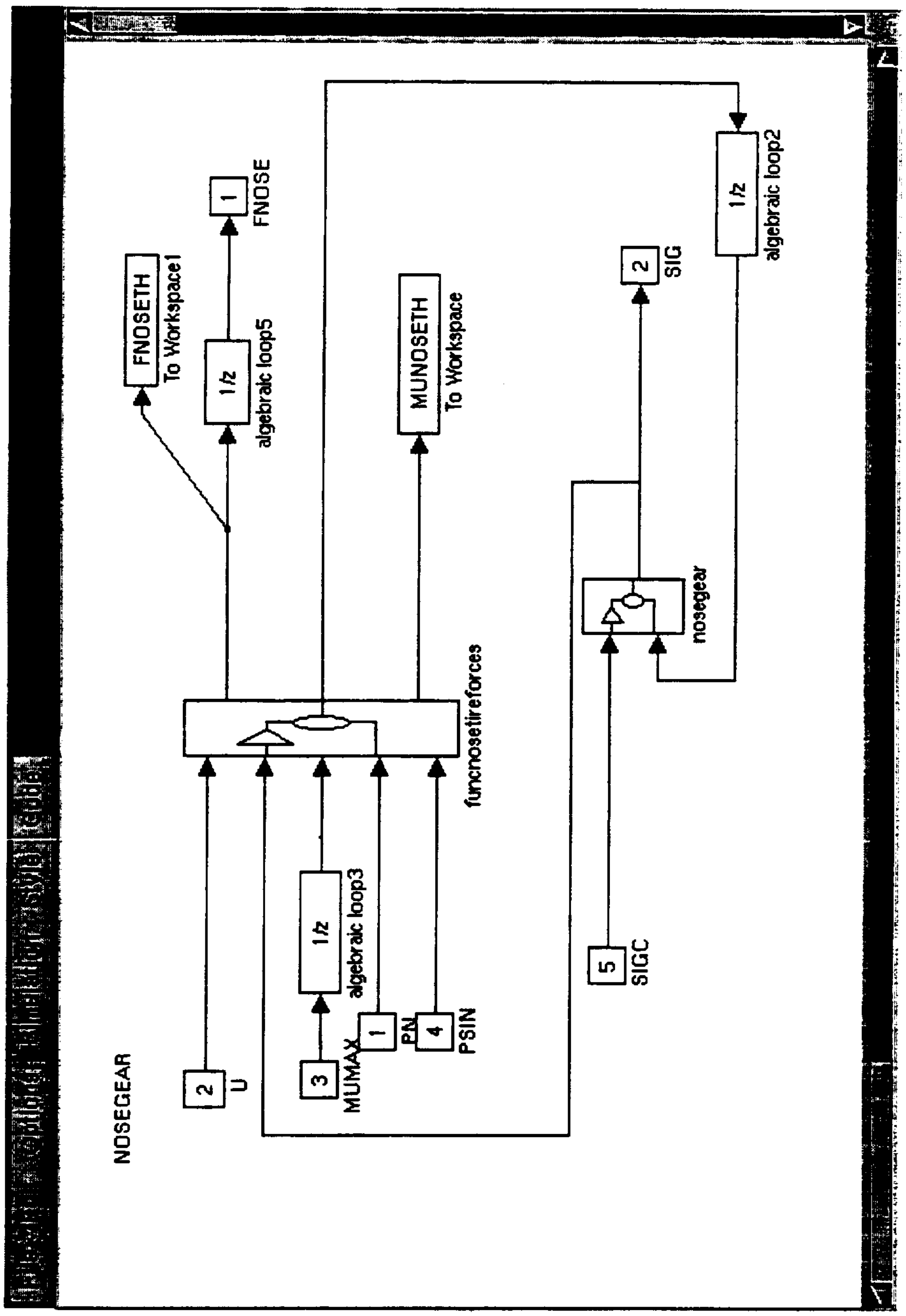

AUTOBRAKE
RRPHASE
KBPHASE
Clock
TMGD
Constant1
Sum1
4
DECELCOMR
5
DECELCOM
10
DU=-DECEL
6
DECELCOML
EABR
EAB
EABL
EABTH
To Workspace7
Mux
Mux7
Mux8
Abs
Abs1
Abs2
0.5
Constant4
Constant
Constant5
>=
Relational Operator1
Relational Operator
Relational Operator2
*
Product2
Product1
Product3
u[1]*u[2]/abs(u[2])
Fcn4
Fcn
Fcn5
true up
true up_
true up__
1/s
PBR_
PB
PBL
RMPRATETH
To Workspace8
NWWLG~=0
Constant8
NWCLG~=0
Constant6
Constant10
0
Constant9
Constant7
Constant11
up true1
up true
up true2
4
PBR
PBRTH
To Workspace5
5
PB_
PBTH
To Workspace4
6
PBL_
PBLTH
To Workspace6
7
FCR
2
FMR
9
FCC
1
FC
8
FCL
3
FML
BDRAGR_
BDRAGC_1
BDRAGL_1
1
BDRAGR
2
BDRAGC
3
BDRAGL

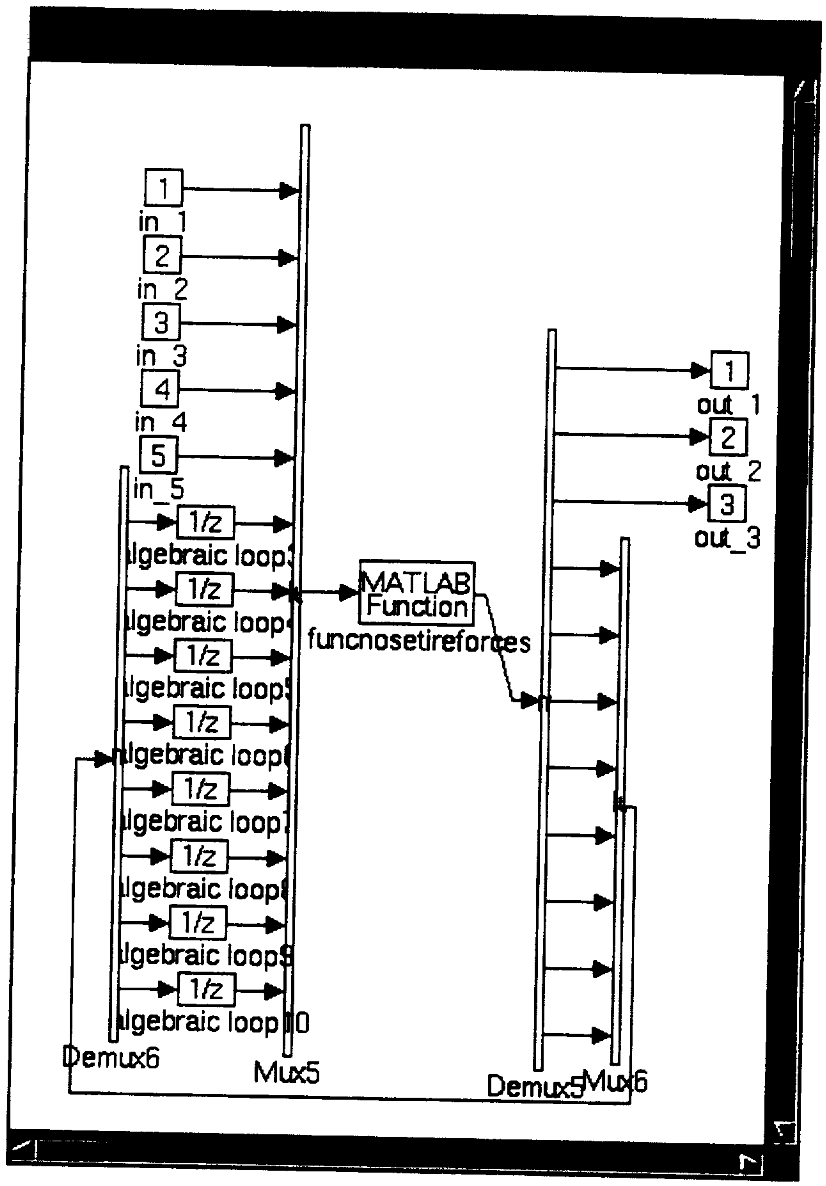

1
PBR
2
FCR
3
FMR
Mux
Mux1
Look-Up Table
u[1]>BDP
breakout
BDRAGR
0
Constant
Switch up true
DRAGCOMRTH
To Workspace1
sqrt(pow(u[1],2)-pow((u[2]/NWWLG),2))*ASEFF
DRAGMAXR
((NWWLG!=0)&&(pow(u[1],2)>pow((u[2]/NWWLG),2)))
fcn
DRAGMAXRTH
To Workspace2
0
Constant1
Switch up true1
Mux
Mux4
u[1]<=u[2]
limit to drag circle
Switch up true2
NWWLG
Gain
1
BDRAGR
BDRAGRTH
To Workspace

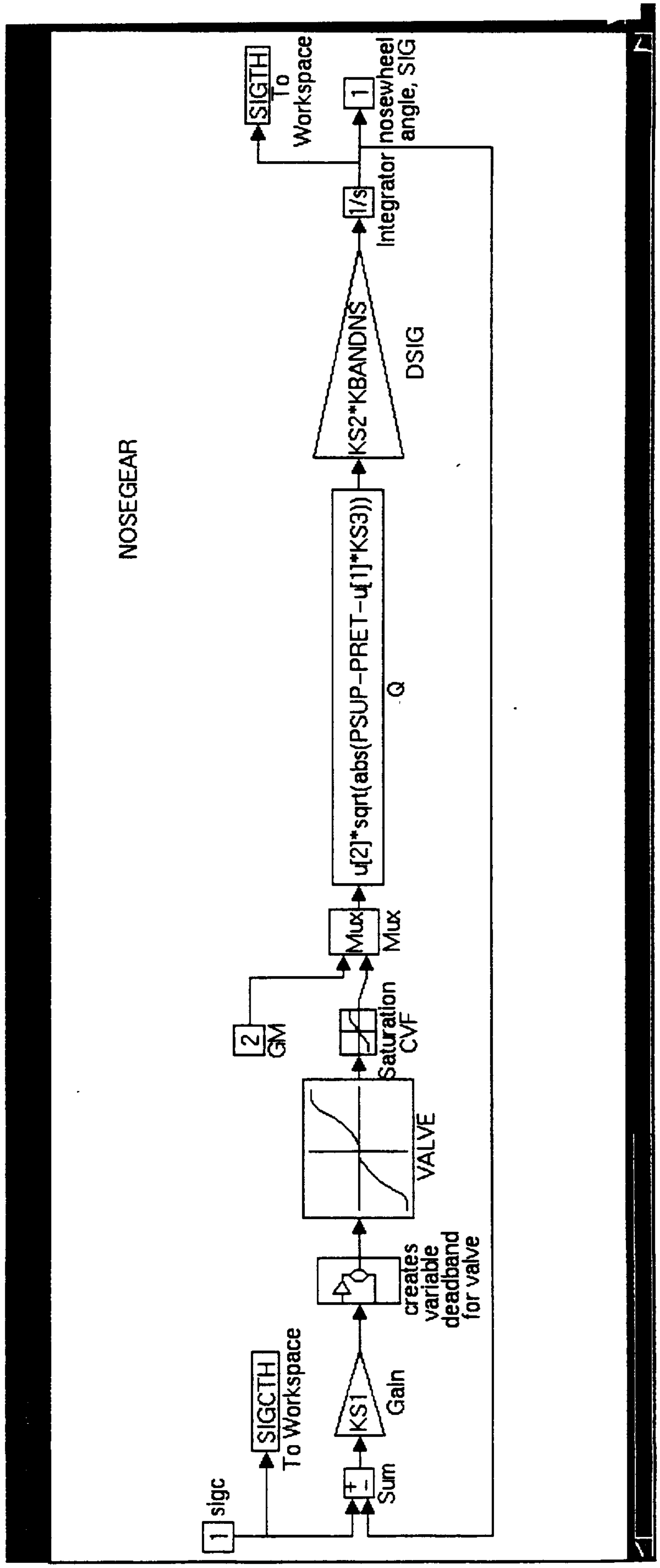


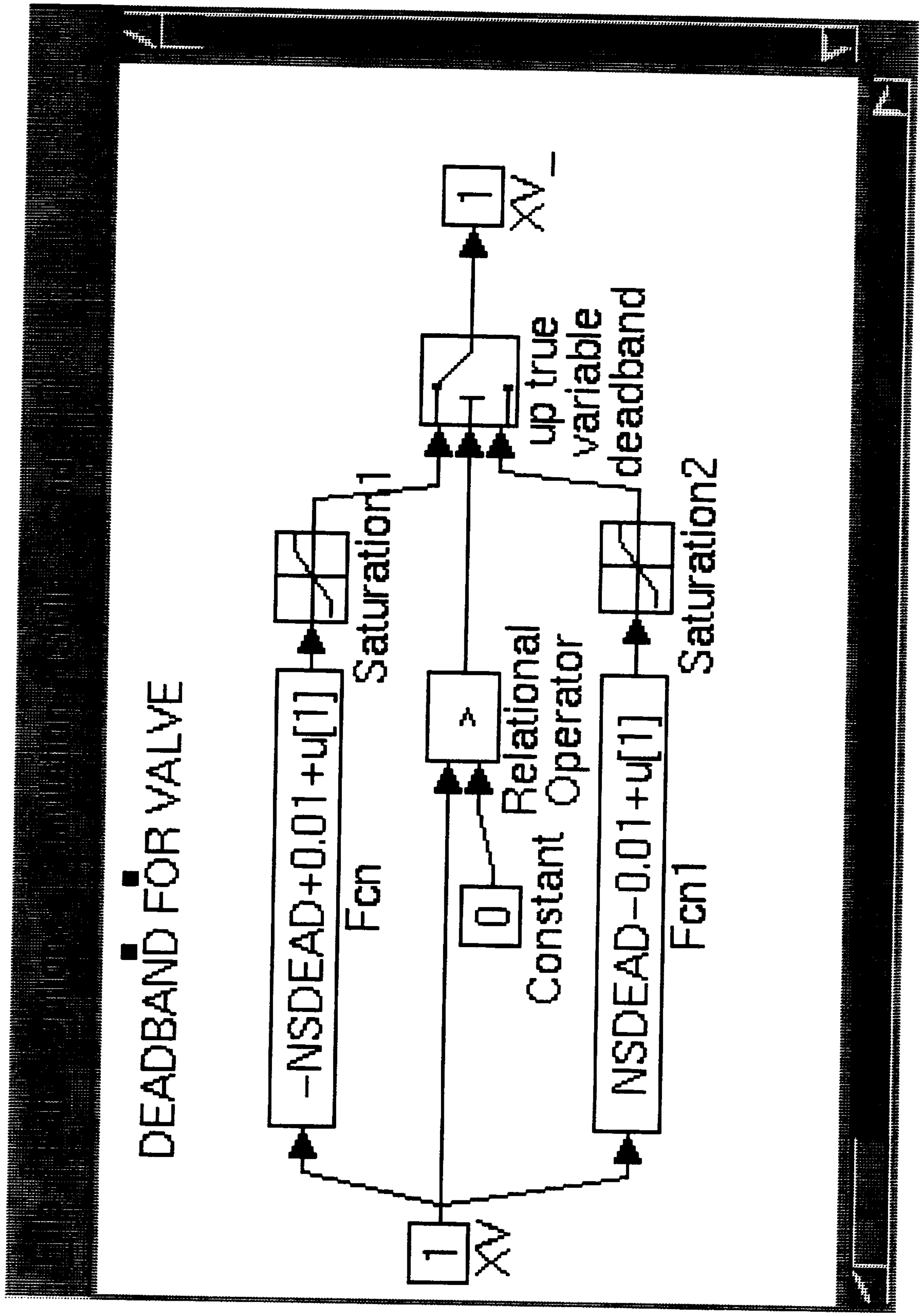
1
PBL
Look-Up Table
BDRAGL
u[1]>BDP
breakout
0
Constant
Switch up true
DRAGCOMLTH
To Workspace1
2
FCL
3
FML
Mux
Mux1
sqrt(pow(u[1],2)-pow((u[2]/NWWLG),2))*ASEFF
DRAGMAXL
((NWWLG!=0)&&(pow(u[1],2)>pow((u[2]/NWWLG),2)))
fcn
DRAGMAXLTH
To Workspace2
0
Constant1
Switch up true1
Mux
Mux4
u[1]<=u[2]
limit to drag circle
Switch up true2
NWWLG
Gain
1
BDRAGL
BDRAGLTH
To Workspace


1
PB
2
FCC
3
FC
Mux
Mux1
Look-Up Table
u[1]>BDP
breakout
0
Constant
BDRAGC
sqrt(pow(u[1],2)-pow((u[2]/NWCLG),2))*ASEFF
DRAGMAXC
((NWCLG!=0)&&(pow(u[1],2)>pow((u[2]/NWCLG),2)))
fcn
0
Constant1
Switch up true
DRAGCOMCTH
To Workspace1
Switch up true1
Mux
Mux4
u[1]<=u[2]
limit to drag circle
Switch up true2
NWCLG
Gain
BDRAGCTH
To Workspace
1
BDRAGC


NOSEGEAR
2
U
3
MUMAX
1
PN
4
PSIN
1/z
algebraic loop3
funcnosetireforces
FNOSETH
To Workspace1
1/z
algebraic loop5
1
FNOSE
MUNOSETH
To Workspace
5
SIGC
nosegear
2
SIG
1/z
algebraic loop2


1
in_1
2
in_2
3
in_3
4
in_4
5
in_5
1/z
lgebraic loop3
1/z
lgebraic loop4
1/z
lgebraic loop5
1/z
lgebraic loop6
1/z
lgebraic loop7
1/z
lgebraic loop8
1/z
lgebraic loop9
1/z
lgebraic loop10
Demux6
Mux5
MATLAB
Function
funcnosetireforces
Demux5
Mux6
1
out_1
2
out_2
3
out_3


NOSEGEAR
1
sigc
SIGCTH
To Workspace
Sum
KS1
Gain
creates variable deadband for valve
VALVE
Saturation CVF
2
GM
Mux
Mux
u[2]*sqrt(abs(PSUP-PRET-u[1]*KS3))
Q
KS2*KBANDNS
DSIG
1/s
Integrator
SIGTH
To Workspace
1
nosewheel angle, SIG


DEADBAND FOR VALVE
1
XV
-NSDEAD+0.01+u[1]
Fcn
0
Constant
>
Relational
Operator
NSDEAD-0.01+u[1]
Fcn1
Saturation1
Saturation2
up true
variable
deadband
1
XV_

FUNCTIONAL HAZARD ANALYSIS
COMMERCIAL PRODUCTS

SYSTEM ROTO in CAT IIIB Environment

SYSTEM ATA NO. 34

REPORT NUMBER ____

PREPARED BY ____ DATE ____

REVIEWED BY ____ DATE ____

1. FUNCTION (SYSTEM LEVEL)	2. HAZARD DESCRIPTION CONSIDER: A) FUNCTION LOSS B) MALFUNCTION C) OTHER SYSTEMS D) MISUSE/EXTERNAL EVENTS	3. PHASE	4. EFFECT OF HAZARD ON OTHER SYSTEMS	5. FAILURE CONDITION FAR/JAR 25.1309 (EFFECT OF HAZARD ON AIRPLANE)	6. HAZ CLASS	7. CERTIFICATION APPROACH	8. REMARKS
1) ROTO Navigation	A) Loss of nav data (GPS ,IRS,airport data base)			A) Loss of ROTO capability Loss of control if loss occurs at cricical time	II	FMEA FTA	Short ROTO critical exposure time period
	B) Erroneous nav data			B) Loss of control	II	FMEA FTA	System architecture will incorporate redundancy and monitoring
2)Runway Acquisition Clearance	A1) Loss of ATC communications			A1) Loss of ROTO capability	IV	FHA	
	A2)Loss of up-link friction data			A2) Could result in on-ground reversion from ROTO to non-ROTO operation	IV	FHA	
	B1) Object on runway			B1) Possible crash	I	FMEA FTA	Procedural - same as CAT IIIB autoland
	B2) Erroneous friction data			B2) Same as A2	IV	FHA	
3)ROTO Steering Guidance	A1) Loss of all guidance command			A1) Loss of ROTO capability Loss of control if	II	FMEA FTA	System architecture will incorporate redundancy and monitoring

SYSTEM OPERATING PHASES (COL. 3)

GROUND
G1 TAXI
G2 TRANSITION TO REVERSE THRUST
G3 IN REVERSE THRUST
G4 TRANSITION TO STOW & ROLL OUT
G5 AIRPLANE STATIC-SYSTEM OPERATING

TAKEOFF
T1 TAKEOFF ROLL-PRIOR TO ROTATION
T2 TAKEOFF-AFTER ROTATION
T3 REJECTED TAKEOFF

IN-FLIGHT
F1 CLIMB
F2 GEAR DOWN
F3 GEAR UP
F4 CRUISE
F5 DESCENT
F6 APPROACH
F7 GO AROUND
F8 ICE PROTECTION ON
F9 BEFORE LANDING
F10 OTHER (DESCRIBE)

LANDING
L1 LANDING ROLL
L2 GROUND ROLL BRAKING

HAZARD CLASS (COL 6)
CLASS I - CATASTROPHIC
CLASS II - HAZARDOUS
CLASS III - MAJOR
CLASS IV - MINOR

REV LTR.

SHEET 1 OF 5

FUNCTIONAL HAZARD ANALYSIS
COMMERCIAL PRODUCTS

SYSTEM ROTO in CAT IIIB Environment

SYSTEM ATA NO. 34

REPORT NUMBER ______

PREPARED BY ______ DATE ______

REVIEWED BY ______ DATE ______

1. FUNCTION (SYSTEM LEVEL)	2. HAZARD DESCRIPTION CONSIDER: A) FUNCTION LOSS B) MALFUNCTION C) OTHER SYSTEMS D) MISUSE/EXTERNAL EVENTS	3. PHASE	4. EFFECT OF HAZARD ON OTHER SYSTEMS	5. FAILURE CONDITION FAR/JAR 25.1309 (EFFECT OF HAZARD ON AIRPLANE)	6. HAZ CLASS	7. CERTIFICATION APPROACH	8. REMARKS
				loss occurs at critical time			
	A2) Loss of rudder			A2) Same as A1	II	FMEA FTA	System architecture will incorporate redundancy and monitoring
	A3) Loss of nosegear			A3) Same as A1	II		System architecture will incorporate redundancy and monitoring
	A5) Loss of rudder pedal backdrive			A5) Could cause unnecessary pilot takeover	IV	FHA	
	A6) Loss of tiller handle backdrive			A6) Same as A5	IV		
	B1) Erroneous steering guidance			B1) Loss of control	I	FMEA FTA	System architecture will incorporate

SYSTEM OPERATING PHASES (COL. 3)

GROUND
G1 TAXI
G2 TRANSITION TO REVERSE THRUST
G3 IN REVERSE THRUST
G4 TRANSITION TO STOW & ROLL OUT
G5 AIRPLANE STATIC-SYSTEM OPERATING

TAKEOFF
T1 TAKEOFF ROLL-PRIOR TO ROTATION
T2 TAKEOFF-AFTER ROTATION
T3 REJECTED TAKEOFF

IN-FLIGHT
F1 CLIMB
F2 GEAR DOWN
F3 GEAR UP
F4 CRUISE
F5 DESCENT
F6 APPROACH
F7 GO AROUND
F8 ICE PROTECTION ON
F9 BEFORE LANDING
F10 OTHER (DESCRIBE)

LANDING
L1 LANDING ROLL
L2 GROUND ROLL BRAKING

HAZARD CLASS (COL 6)
CLASS I - CATASTROPHIC
CLASS II - HAZARDOUS
CLASS III - MAJOR
CLASS IV - MINOR

REV LTR.

SHEET 2 OF 5

FUNCTIONAL HAZARD ANALYSIS
COMMERCIAL PRODUCTS

REPORT NUMBER ______

SYSTEM ROTO in CAT IIIB Environment

SYSTEM ATA NO. 34

PREPARED BY ______ DATE ______

REVIEWED BY ______ DATE ______

1. FUNCTION (SYSTEM LEVEL)	2. HAZARD DESCRIPTION CONSIDER: A) FUNCTION LOSS B) MALFUNCTION C) OTHER SYSTEMS D) MISUSE/EXTERNAL EVENTS	3. PHASE	4. EFFECT OF HAZARD ON OTHER SYSTEMS	5. FAILURE CONDITION FAR/JAR 25.1309 (EFFECT OF HAZARD ON AIRPLANE)	6. HAZ CLASS	7. CERTIFICATION APPROACH	8. REMARKS
							redundancy and monitoring
	B2) Erroneous Rudder			B2) Same as B1	I	FMEA FTA	System architecture will incorporate redundancy and monitoring
	B3) Erroneous nosegear steering			B3) Same as B1	I	FMEA FTA	System architecture will incorporate redundancy and monitoring
	B5) Erroneous rudder pedal backdrive			B5) Same as A5	IV	FTA	
	B6) Erroneous tiller handle backdrive			B6) Same as A6	IV	FHA	
4) ROTO Deceleration	A1) Loss of deceleration command			A1) Loss of ROTO capability Loss of control if	II	FMEA FTA	System architecture will incorporate redundancy and monitoring

SYSTEM OPERATING PHASES (COL. 3)

GROUND
G1 TAXI
G2 TRANSITION TO REVERSE THRUST
G3 IN REVERSE THRUST
G4 TRANSITION TO STOW & ROLL OUT
G5 AIRPLANE STATIC SYSTEM OPERATING

TAKEOFF
T1 TAKEOFF ROLL-PRIOR TO ROTATION
T2 TAKEOFF-AFTER ROTATION
T3 REJECTED TAKEOFF

IN-FLIGHT
F1 CLIMB
F2 GEAR DOWN
F3 GEAR UP
F4 CRUISE
F5 DESCENT
F6 APPROACH
F7 GO AROUND
F8 ICE PROTECTION ON
F9 BEFORE LANDING
F10 OTHER (DESCRIBE)

LANDING
L1 LANDING ROLL
L2 GROUND ROLL BRAKING

HAZARD CLASS (COL 6)
CLASS I - CATASTROPHIC
CLASS II - HAZARDOUS
CLASS III - MAJOR
CLASS IV - MINOR

REV LTR.

SHEET 3 OF 5

FUNCTIONAL HAZARD ANALYSIS
COMMERCIAL PRODUCTS

SYSTEM ROTO in CAT IIIB Environment

SYSTEM ATA NO. 34

REPORT NUMBER ______

PREPARED BY ______ DATE ______

REVIEWED BY ______ DATE ______

1. FUNCTION (SYSTEM LEVEL)	2. HAZARD DESCRIPTION CONSIDER: A) FUNCTION LOSS B) MALFUNCTION C) OTHER SYSTEMS D) MISUSE/EXTERNAL EVENTS	3. PHASE	4. EFFECT OF HAZARD ON OTHER SYSTEMS	5. FAILURE CONDITION FAR/JAR 25.1309 (EFFECT OF HAZARD ON AIRPLANE)	6. HAZ CLASS	7. CERTIFICATION APPROACH	8. REMARKS
				loss occurs at critical time			Short ROTO critical exposure time period
	A2) Loss of auto brake			A2) Reversion to manual	IV	FHA	
	A3) Loss of auto reverse thrust			A3) Reversion to manual	IV	FHA	
	A4) Loss of reverse thrust handle backdrive			A4) Reversion to manual	IV	FHA	
	B1) Erroneous deceleration commands			B1) Same as A1	II	FMEA FTA	System architecture will incorporate redundancy and monitoring Short ROTO critical exposure time period
	B2) Erroneous auto brake			B2) Same as B1	II	FMEA FTA	
	B3) Erroneous reverse thrust			B3) Same as B1	II	FMEA FTA	
	B4) Erroneous reverse thrust handle backdrive			B4) Same as A4	IV	FHA	

SYSTEM OPERATING PHASES (COL. 3)

GROUND
G1 TAXI
G2 TRANSITION TO REVERSE THRUST
G3 IN REVERSE THRUST
G4 TRANSITION TO STOW & ROLL OUT
G5 AIRPLANE STATIC-SYSTEM OPERATING

TAKEOFF
T1 TAKEOFF ROLL-PRIOR TO ROTATION
T2 TAKEOFF-AFTER ROTATION
T3 REJECTED TAKEOFF

IN-FLIGHT
F1 CLIMB
F2 GEAR DOWN
F3 GEAR UP
F4 CRUISE
F5 DESCENT
F6 APPROACH
F7 GO AROUND
F8 ICE PROTECTION ON
F9 BEFORE LANDING
F10 OTHER (DESCRIBE)

LANDING
L1 LANDING ROLL
L2 GROUND ROLL BRAKING

HAZARD CLASS (COL 6)
CLASS I - CATASTROPHIC
CLASS II - HAZARDOUS
CLASS III - MAJOR
CLASS IV - MINOR

REV LTR.

SHEET 4 OF 5

FUNCTIONAL HAZARD ANALYSIS
COMMERCIAL PRODUCTS

REPORT NUMBER ____________

SYSTEM ROTO in CAT IIIB Environment

SYSTEM ATA NO. 34

PREPARED BY ____________ DATE ______

REVIEWED BY ____________ DATE ______

1. FUNCTION (SYSTEM LEVEL)	2. HAZARD DESCRIPTION CONSIDER: A) FUNCTION LOSS B) MALFUNCTION C) OTHER SYSTEMS D) MISUSE/EXTERNAL EVENTS	3. PHASE	4. EFFECT OF HAZARD ON OTHER SYSTEMS	5. FAILURE CONDITION FAR/JAR 25.1309 (EFFECT OF HAZARD ON AIRPLANE)	6. HAZ CLASS	7. CERTIFICATION APPROACH	8. REMARKS
	C1) Loss of anti-skid			C1) Reversion to manual. Probable loss of ROTO capability	IV	FHA	
5) ROTO Displays (on HUD or PFD)	A1)Loss of pilot displays			A1)Loss of ROTO capability. Possible loss of control at critical time	II	FMEA FTA	System architecture will incorporate redundancy and monitoring Short ROTO critical exposure time period
	A2)Loss of aircraft weight			A2)Loss of ROTO capability to select exit	II		System architecture will incorporate redundancy and monitoring

SYSTEM OPERATING PHASES (COL. 3)

GROUND
G1 TAXI
G2 TRANSITION TO REVERSE THRUST
G3 IN REVERSE THRUST
G4 TRANSITION TO STOW & ROLL OUT
G5 AIRPLANE STATIC-SYSTEM OPERATING

TAKEOFF
T1 TAKEOFF ROLL-PRIOR TO ROTATION
T2 TAKEOFF-AFTER ROTATION
T3 REJECTED TAKEOFF

IN-FLIGHT
F1 CLIMB
F2 GEAR DOWN
F3 GEAR UP
F4 CRUISE
F5 DESCENT
F6 APPROACH
F7 GO AROUND
F8 ICE PROTECTION ON
F9 BEFORE LANDING
F10 OTHER (DESCRIBE)

LANDING
L1 LANDING ROLL
L2 GROUND ROLL BRAKING

HAZARD CLASS (COL 6)
CLASS I - CATASTROPHIC
CLASS II - HAZARDOUS
CLASS III - MAJOR
CLASS IV - MINOR

REV LTR.

SHEET 5 OF 5

APPENDIX A: EVENT TIMELINE FOR A REPRESENTATIVE ROTO IMPLEMENTATION -- MANUAL LANDING / AUTOMATIC ROTO DECELERATION AND RUNWAY EXIT

Event	Time, Speed, Position	Flight Crew: Pilot Flying, Pilot Not Flying	Flight Deck: ROTO Automation	ATC: Controller, Automation
i	By 11:48, ~7000 ft., 250 Kts., inbound to SMO VOR from BAYST intersection	Crew determines (e.g., from ATIS) that ROTO is in operation, observes indications that terminal area traffic spacing is consistent with ROTO-required separations		
ii	Between SMO and initiation of base-leg turn, between ~11:48 and ~11:50, 6500-4000 ft., 250-200 Kts.	Crew is assigned a runway and is cleared for the approach; readies relevant ROTO information (using FMS, etc.)		
iii	Upon entering base-leg turn, at 200 Kts. and 4000 - 2900 ft. at ~11:55:15 - 11:56:00	Although not required for ROTO operations, crew will endeavor to visually acquire and keep track of lead aircraft in order to provide backup monitoring of ATC-dictated separation		
iv	11:55:00 - 12:00:00; 4000 - 50 ft., 200 - ~135 Kts.	Crew visually acquires airport environment, and runway and touchdown zone, if possible		
v	11:48:00 - 11:56:30; 6500 - 2200 ft., 250 - 160 Kts.			Approach Control issues "Expect ROTO" clearance, and information regarding anticipated available exits, surface conditions, other aircraft, etc.

EVENT TIMELINE FOR A REPRESENTATIVE ROTO IMPLEMENTATION (CONTINUED): MANUAL LANDING / AUTOMATIC ROTO DECELERATION AND RUNWAY EXIT

Event	Time, Speed, Position	Flight Crew: Pilot Flying, Pilot Not Flying	Flight Deck: ROTO Automation	ATC: Controller, Automation
vi	11:48:05 - 11:56:40; ~6500 ft. at or below 250 Kts. to 2200 ft. at ~160 Kts.	Pilot Not Flying acknowledges "Expect ROTO" clearance		
vii	11:55:30 - 11:56:00; ~4000 ft.; ~200 - ~185 Kts.			Approach Control issues "Cleared for the approach / contact Tower" clearance
viii	1:55:33 - 11:56:33; ~4000 ft.; ~200 - ~185 Kts.	Pilot Not Flying acknowledges hand-off and clearance		
ix	1:55:38 - 11:56:38; ~4000 ft.; ~200 - ~185 Kts.	Pilot Not Flying contacts Tower Control		
x	Any time after hand-off to Tower Control (and probably before Glide Slope intercept)			Tower acknowledges aircraft's entry into final approach; issues "Cleared to land / expect ROTO" clearance
xi	Any time after hand-off to Tower Control (and probably before Glide Slope intercept)	Pilot Not Flying acknowledges clearances		

EVENT TIMELINE FOR A REPRESENTATIVE ROTO IMPLEMENTATION (CONTINUED): MANUAL LANDING / AUTOMATIC ROTO DECELERATION AND RUNWAY EXIT

Event	Time, Speed, Position	Flight Crew: Pilot Flying, Pilot Not Flying	Flight Deck: ROTO Automation	ATC: Controller, Automation
xii	11:48:10 - 11:56:45; (could be later ... but is not likely); ~6500 ft. at or below 250 Kts. to 2200 ft. at ~160 Kts.	Pilot Not Flying engages ROTO system calculations; calculated guidance loaded into preselect buffer (for automatic mode, guidance and deceleration control also placed in preselect)		
xiii	11:48:11 - 11:56:46; (could be later ... but is not likely); ~6500 ft. at or below 250 Kts. to 2200 ft. at ~160 Kts.		ROTO executes a B.I.T.; reports Ready status to crew	
xiv	11:48:20 - 11:56:55; (could be later ... but is not likely); ~6400 ft. at or below 250 Kts. to 2100 ft. at ~160 Kts.	Pilot Not Flying checks ROTO readiness, informs Pilot Flying; Pilot Flying evaluates initial estimate for touchdown point and exit selection		
xv	Continually	Pilot Flying tracks Localizer and Glide Slope; controls speed, altitude, and rate		

EVENT TIMELINE FOR A REPRESENTATIVE ROTO IMPLEMENTATION (CONTINUED): MANUAL LANDING / AUTOMATIC ROTO DECELERATION AND RUNWAY EXIT

Event	Time, Speed, Position	Flight Crew: Pilot Flying, Pilot Not Flying	Flight Deck: ROTO Automation	ATC: Controller, Automation
xvi	Before threshold (in all probability, from 10 - 5 seconds before touchdown, preserving 50 second separation with leading aircraft; ~100 - 50 ft)			Tower issues "Clear for ROTO" clearance, including available exits, exit speeds, and runway friction information
xvii	Before threshold (in all probability, from 8 - 3 seconds before touchdown, preserving 50 second separation with leading aircraft; ~100 - 50 ft)	Pilot Flying arms ROTO for activation during landing / roll out sequence		
xviii	Five seconds before touchdown to ~15 - 20 seconds after touchdown	Crew acknowledges "Clear for ROTO" clearance		
xix	11:59:40 - 11:59:54; just prior to Middle Marker up to ~ edge of runway (i.e., ~200 - 50 ft.); ~135 Kts.	Crew keeps lead aircraft in sight, if possible; watches for clearing of runway (visibility permitting); Pilot Flying controls aircraft attitude, airspeed, altitude, and sink rate; initiates flare maneuver		

EVENT TIMELINE FOR A REPRESENTATIVE ROTO IMPLEMENTATION (CONTINUED): MANUAL LANDING / AUTOMATIC ROTO DECELERATION AND RUNWAY EXIT

Event	Time, Speed, Position	Flight Crew: Pilot Flying, Pilot Not Flying	Flight Deck: ROTO Automation	ATC: Controller, Automation
xx Flare	11:59:54; ~ edge of runway; ~60 - 50 ft.; ~135 Kts.	Pilot Flying conducts flare maneuver, touchdown, and begins roll out		
1 Main gear down	12:00:00; ~200 ft. past displaced threshold;			
2	12:00:02; ~300 - 350 ft. past displaced threshold; ~130 Kts.		Sensors report touchdown position, ground speed to ROTO automation for prediction update of selection and deceleration schedule calculations	
3		Pilot "Flying" monitors ROTO's control of centerline tracking; engages thrust reversing; engines move to idle reverse thrust settings		
4	12:00:02 to initiation of turn on to exitway; from thruster onset to stowing; ~130 to as slow as 70 Kts.		ROTO monitors thrust reverser deployment and utilizes reverser / deceleration data to augment overall ROTO routine	

EVENT TIMELINE FOR A REPRESENTATIVE ROTO IMPLEMENTATION (CONTINUED): MANUAL LANDING / AUTOMATIC ROTO DECELERATION AND RUNWAY EXIT

Event	Time, Speed, Position	Flight Crew: Pilot Flying, Pilot Not Flying	Flight Deck: ROTO Automation	ATC: Controller, Automation
5 Nose gear down	12:00:01 - 12:00:03; ~250 - ~400 ft. past displaced threshold; ~133 - ~128 Kts.			
6	12:00:03 - 12:00:15; ~400 - 2000 ft. past displaced threshold; ~131 - ~94 Kts. (depending on aircraft weight, runway conditions, and exit		ROTO automation initiates auto braking command; brakes develop braking pressure to comply	
7	Continually throughout roll out and turn off	Pilot "Flying" monitors automatic control of centerline tracking and of deceleration		
8	Continually throughout roll out	Pilot "Flying" keeps hand on throttles; monitors effect of auto braking and thrust reversing; is ready to control reverse thrust for emergency stopping or for supplementing a reversion to auto- or manual braking, or to set up for a go-around		
9	Continually throughout roll out	Pilot "Flying" observes track and deceleration "guidance;" monitors speed; watches for runway obstacles, etc.		

EVENT TIMELINE FOR A REPRESENTATIVE ROTO IMPLEMENTATION (CONTINUED): MANUAL LANDING / AUTOMATIC ROTO DECELERATION AND RUNWAY EXIT

Event	Time, Speed, Position	Flight Crew: Pilot Flying, Pilot Not Flying	Flight Deck: ROTO Automation	ATC: Controller, Automation
10	Periodically throughout roll out		ROTO determines updated estimate of turn-commit point (with some accommodation for pilot reaction time)	
11	Continually throughout roll out and turn off		Sensors continue to provide ROTO system with runway friction estimates and aircraft position; actual deceleration continues to be compared to required rate; necessary compensatory control inputs are generated	
12	At or before 12:00:00 - 12:00:37 (if at all); distance depends on exit; at or above 70 Kts.	If possible (considering visual conditions that could be as poor as 300 ft. RVR), crew visually acquires ROTO exit turn point ground track guidance and other cues; monitors position, speed, and deceleration data, and observes up-coming turn-commit indication		
13	12:00:27 - 12:00:35; distance depends on deceleration profile; ~80 - ~70 Kts.	Pilot "Flying" sets reverse thrust at idle reverse at or before 70 Kts. ground speed, if safe to do so		
14	12:00:28 - 12:00:36; distance depends on deceleration profile; ~80 - ~70 Kts.		ROTO records thrust reversers being set to idle and accounts for resulting change in system capability	

EVENT TIMELINE FOR A REPRESENTATIVE ROTO IMPLEMENTATION (CONTINUED): MANUAL LANDING / AUTOMATIC ROTO DECELERATION AND RUNWAY EXIT

Event	Time, Speed, Position	Flight Crew: Pilot Flying, Pilot Not Flying	Flight Deck: ROTO Automation	ATC: Controller, Automation
15	12:00:32 - 12:00:38; distance depends on exit; ~70 Kts.	At exit turn point, Pilot "Flying" monitors exit track and is ready to take over if needed		
16	From ~12:00:47 - ~12:00:52 to...; distance depends on exit; ~50 Kts - taxi speed	Pilot "Flying" continues to monitor ROTO's tracking of exit path guidance and is prepared for manual take over if needed		
17	12:00:48 - 12:00:53		Immediately upon clearing the runway (determined by GPS or some other means of position verification), ROTO communicates (probably via data link) a "Clear of runway" message to the tower and ATC automation	
18	12:00:50 - 12:00:55			ATC communicates (probably via data link) acknowledgment of message
19	12:00:51 - 12:00:56		Tower's receipt of message is annunciate to crew	
20	12:00:52 - 12:00:57	Crew observes ROTO's annunciation of message receipt		

EVENT TIMELINE FOR A REPRESENTATIVE ROTO IMPLEMENTATION (CONTINUED): MANUAL LANDING / AUTOMATIC ROTO DECELERATION AND RUNWAY EXIT

Event	Time, Speed, Position	Flight Crew: Pilot Flying, Pilot Not Flying	Flight Deck: ROTO Automation	ATC: Controller, Automation
21	12:00:47 - 12:00:52 to...; distance depends on exit; ~50 Kts. - taxi speed or stop		After clearing the runway, the ROTO exit routine executes a deceleration schedule that slows the aircraft in anticipation of pilot take over for taxi	
22	12:00:57 - 12:01:02; taxi speed or stop	During final deceleration on exitway, Pilot "Not Flying" contacts Ground Control for taxi instructions, terminal assignment (if not already designated)	•	
23	12:01:02 - 12:01:07 to no longer than 12:01:37 - 12:01:42		Ground Control either holds the aircraft on the exitway or clears the aircraft for taxi; Ground informs Tower of exitway occupancy status; aircraft clears exitway before exitway is needed for subsequent (~50 second lag) ROTO-governed landings	

REPORT DOCUMENTATION PAGE

Form Approved
OMB No. 0704-0188

1. AGENCY USE ONLY (*Leave blank*)

2. REPORT DATE
August 1996

3. REPORT TYPE AND DATES COVERED
Contractor Report

4. TITLE AND SUBTITLE
Guidance and Control Design for High-Speed Rollout and Turnoff (ROTO)

5. FUNDING NUMBERS
NAS1-19703
Task 7
538-04-13-02

6. AUTHOR(S)
S. H. Goldthorpe, R. D. Dangaran, J. P. Dwyer, L. S. McBee,
R. M. Norman, J. H. Shannon, L. G. Summers

7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES)
McDonnell Douglas Corporation
McDonnell Douglas Aerospace
Transport Aircraft
2401 E. Wardlow Road
Long Beach, CA 90807-4418

8. PERFORMING ORGANIZATION REPORT NUMBER
CRAD-9206-TR-1659

9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES)
National Aeronautics and Space Administration
Langley Research Center
Hampton, VA 23681-0001

10. SPONSORING / MONITORING AGENCY REPORT NUMBER
NASA CR-201602

11. SUPPLEMENTARY NOTES
Langley Technical Monitor: Richard M. Hueschen
Final Report

12a. DISTRIBUTION / AVAILABILITY STATEMENT
Unclassified - Unlimited
Subject Category 08

12b. DISTRIBUTION CODE

13. ABSTRACT (*Maximum 200 words*)

A ROTO architecture, braking and steering control law and display designs for a research high speed Rollout and Turnoff (ROTO) system applicable to transport class aircraft are described herein. Minimum surface friction and FMS database requirements are also documented. The control law designs were developed with the aid of a non-real time simulation program incorporating airframe and gear dynamics as well as steering and braking guidance algorithms. An attainable objective of this ROTO system, as seen from the results of this study, is to assure that the studied aircraft can land with runway occupancy times less then 53 seconds. Runway occupancy time is measured from the time the aircraft crosses the runway threshold until its wing tip clears the near side of the runway. Turnoff ground speeds of 70 knots onto 30 degree exits are allowed with dry and wet surface conditions. Simulation time history and statistical data are documented herein. Parameters which were treated as variables in the simulation study include aircraft touchdown weight/speed/location, aircraft CG, runway friction, sensor noise and winds. After further design and development of the ROTO control system beyond the system developed in reference 1, aft CG MD-11 aircraft no longer require auto-asymmetric braking (steering) and fly-by-wire nose gear steering. However, the auto ROTO nose gear hysteresis must be less than 2 degrees. The 2 sigma dispersion certified for MD-11 CATIIIB is acceptable. Using this longitudinal dispersion, three ROTO exits are recommended at 3300, 4950 and 6750 feet past the runway threshold. The 3300 foot exit is required for MD-81 class aircraft. Designs documented in this report are valid for the assumptions/models used in this simulation. It is believed that the results will apply to the general class of transport aircraft; however further effort is required to validate this assumption for the general case.

14. SUBJECT TERMS
Runway Guidance and Control, Differential GPS, Rollout, Turnoff, Runway Occupancy Time, Aircraft Ground Operations

15. NUMBER OF PAGES
270

16. PRICE CODE
A12

17. SECURITY CLASSIFICATION OF REPORT	18. SECURITY CLASSIFICATION OF THIS PAGE	19. SECURITY CLASSIFICATION OF ABSTRACT	20. LIMITATION OF ABSTRACT
Unclassified	Unclassified		

NSN 7540-01-280-5500

Standard Form 298 (Rev. 2-89)
Prescribed by ANSI Std Z39-18
298-102