TRANSPORTATION RESEARCH RECORD 787

Computer Graphics Applications

TRANSPORTATION RESEARCH BOARD

COMMISSION ON SOCIOTECHNICAL SYSTEMS
NATIONAL RESEARCH COUNCIL

NATIONAL ACADEMY OF SCIENCES
WASHINGTON, D.C. 1980

Transportation Research Record 787
Price $4.20
Edited for TRB by Naomi Kassabian

modes
1 highway transportation
4 air transportation

subject areas
12 planning
13 forecasting

Library of Congress Cataloging in Publication Data
National Research Council. Transportation Research Board.
Computer graphics applications.

(Transportation research record; 787)
Reports presented at the 59th annual meeting of the Transportation Research Board.
1. Transportation—Research—Data processing—Addresses, essays, lectures. 2. Computer graphics—Addresses, essays, lectures. I. National Research Council (U.S.). Transportation Research Board. II. Series.
TE7.H5 no. 787 [HE191.5] 380.5s 81-9570
ISBN 0-309-03201-6 ISSN 0361-1981 [380.5'072] AACR2

Sponsorship of the Papers in This Transportation Research Record

GROUP 1—TRANSPORTATION SYSTEMS PLANNING AND ADMINISTRATION
Leon M. Cole, Library of Congress, chairman

Transportation Forecasting Section
George V. Wickstrom, Metropolitan Washington Council of Governments, chairman

Committee on Computer Graphics and Interactive Graphics
Dietrich R. Bergmann, General Motors Transportation Systems Division, chairman
Carter W. Brown, Amir Eiger, Richard E. Esch, Gordon P. Fisher, Claus D. Gehner, David A. Green, Yehuda Gur, Hubert A. Henry, Everett S. Joline, Jerome M. Lutin, Harold Moellering, Granville E. Paules, Ludie G. Richard, Jerry B. Schneider, Robert L. Siegel, Robert A. White

James A. Scott, Transportation Research Board staff

The organizational units, officers, and members are as of December 31, 1979.

Contents

Authors of the Papers in This Record

Couture, Michael R., Transportation Systems Center, U.S. Department of Transportation, Kendall Square, Cambridge, MA 02142
Esch, Richard, Bureau of Transportation Planning, Michigan Department of Transportation, P.O. Box 30050, Lansing, MI 48909
Gotts, Terry, Bureau of Transportation Planning, Michigan Department of Transportation, P.O. Box 30050, Lansing, MI 48909
Kobialka, Edward J., School of Engineering and Applied Science, Princeton University, Princeton, NJ 08544
Livingston, Dale, National Aviation Facilities Experimental Center, Federal Aviation Administration, Atlantic City, NJ 08405
Lutin, Jerome M., Department of Civil Engineering, Princeton University, Princeton, NJ 08544
Polhemus, Neil W., School of Engineering and Applied Science, Princeton University, Princeton, NJ 08544

Computer Graphics for Transit Planning: An Empirical Study

MICHAEL R. COUTURE

This paper documents the results of a case study in which an interactive graphics system was applied to solve a bus transit design problem. The aim of the study was to gain insights into the applied use of the interactive graphics system and to assess its utility as a practical sketch-planning tool. The particular system used was the Interactive Graphic Transit Design System (IGTDS), and the problem to be addressed was the design of a bus system in the Southfield-Jeffries Corridor in southeast Michigan. The study consisted essentially of a data-base development phase and a system-application phase. The results of the case study are reported in terms of (a) the user requirements for data-base development, (b) the general on-line problem-solving experience by using IGTDS, (c) the efficiency characteristics of the IGTDS design process, and (d) the on-line times and costs associated with IGTDS operation. The major findings were that IGTDS was simple to use, that the application costs were relatively low in terms of both data-base development and system operation, and that the IGTDS design process was quite efficient in generating design solutions to the relatively complex bus transit problem (i.e., 81 designs in 24 h of on-line time). These findings suggest the strong potential of interactive graphics systems like IGTDS as practical and cost-effective sketch-planning tools.

The application of interactive computer graphics in designing public transportation services is a relatively new and important research effort. The potential benefits of interactive graphics methods include fast response to planning and design questions (and thus the capability for exploring in a timely manner a wide range of alternative solutions to a particular design problem) and the capability for assimilating graphic as well as alphanumeric information. Since transportation problems in general--and transit design problems in particular--are spatially oriented, the graphic capability can ease the user burden in problem definition and analysis and can help eliminate errors prevalent in coding spatial information (e.g., network coding errors).

Despite these apparent advantages, application of interactive graphics to transit system planning and design has been limited. Several prototypical interactive graphics systems have been developed to aid in the transit system design process (1-5), but for the most part little has been reported regarding the potential of these systems as practical design tools.

This paper documents a case-study application in which an interactive graphics system was used by a group of designers to solve a bus transit design problem by using actual transit system data. The system used was the Interactive Graphic Transit Design System (IGTDS) (5). Developed principally at the University of Washington by Rapp (6), IGTDS is currently distributed to the public by the Urban Mass Transportation Administration (UMTA) of the U.S. Department of Transportation. The case study consisted of two major phases: a data-base development phase and a system-application phase. The data-base development phase of the study was carried out at the General Motors (GM) Technical Center in Warren, Michigan, by the GM Transportation Systems Division (GMTSD). The site selected for the case study was the Southfield-Jeffries Corridor in southeast Michigan. The system-application phase was carried out at Northwestern University by a group of transportation engineering graduate students who were asked to use IGTDS to assist in designing a bus system that would serve trips that originated in the Southfield-Jeffries Corridor and were destined for the Detroit Medical Center.

The primary objectives for conducting the case study were as follows:

1. To ascertain and document the user requirements involved in IGTDS data-base development,
2. To observe the general problem-solving experience that characterized the IGTDS design process and to document the various design strategies employed and the quality of the resulting design solutions,
3. To determine the design efficiency characteristics of the IGTDS design process, and
4. To determine the user costs involved in IGTDS operation.

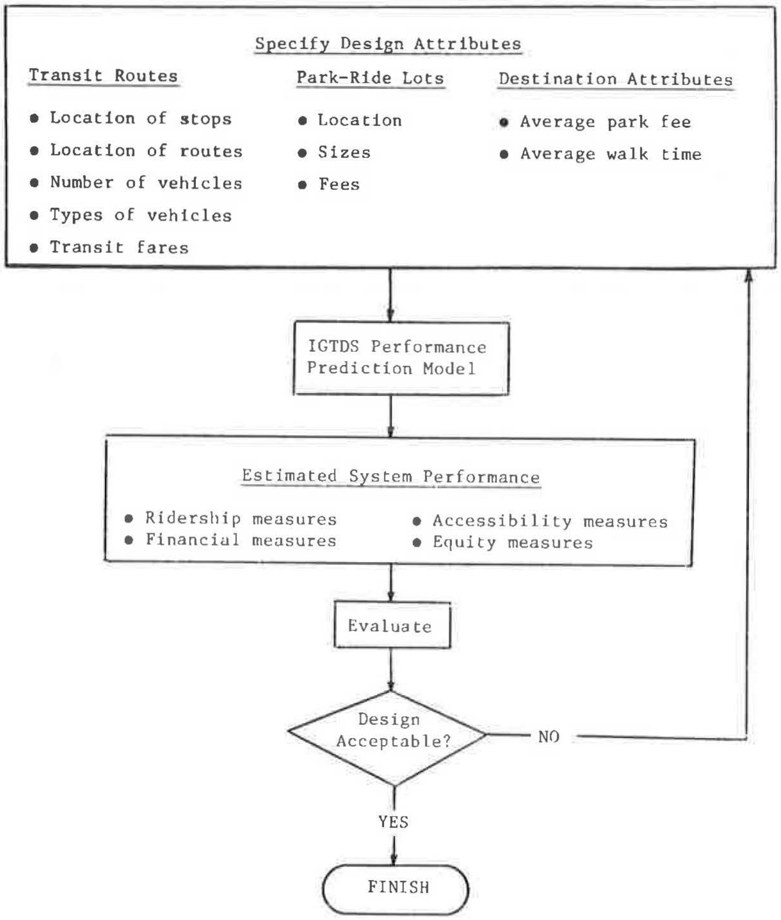
THE SYSTEM (IGTDS)

IGTDS is a tool for assisting in the design and evaluation of alternative transit systems that serve many trip origins and a single major destination (i.e., many-to-one services). IGTDS is designed to be used principally as a sketch-planning aid by which design changes (i.e., changes in transit level of service) can be executed rapidly and corresponding measures of transit performance can be made available almost instantantly to the planner and designer for evaluation. At the heart of IGTDS are a mode-choice model called the n-dimensional logit model (7) and a capacity-restrained transit assignment model. For any given design specification, this model estimates the number of trip makers likely to use each of the modes available in IGTDS (i.e., drive, park-and-ride, and walk-and-ride) and assigns these trips to the transportation network via the shortest impedance paths.

IGTDS allows the user to manipulate the various design variables easily and thus to change the relative trip impedances and predicted shares for the given modes. Any element--from a complete redesign of the transit route structure to a modification of the fare at a single stop--can be handled at the computer graphics terminal, through either graphic input (i.e., via the terminal display screen cursor) or alphanumeric input (i.e., via the terminal keyboard). Based on the estimated modal shares and subsequent assignment of transit system loadings for a given design, a set of performance measures is computed and can be displayed graphically. These performance measures are used to evaluate the given transit system design. The general transit system design process that uses IGTDS is illustrated in Figure 1. This iterative process consists essentially of three steps: (a) specifying design attributes through interactive input by using IGTDS, (b) obtaining system performance estimates from IGTDS, and (c) evaluating the design and modifying it if it is unacceptable. This is the basic design process used by the student designers in the case-study application.

The basic hardware requirements for operating IGTDS software are either a PDP-10 or an IBM series 370 computer and a Tektronix series 4010 graphics terminal.

Figure 1. Transit design process that uses IGTDS.



DATA-BASE DEVELOPMENT

The data base required for operating IGTDS consists primarily of information that describes the geography of the study region, which includes the regional demand for travel to the specified activity area (i.e., the destination) and the underlying transportation network (e.g., the highway system) that serves trips within the region. The transportation network serves as a geographic base map for the IGTDS design process, since all regional attributes (including trip origins and some transit system attributes) are assigned directly to elements (i.e., links and nodes) of this network. Associated with each network node are Cartesian (i.e., X,Y) coordinates (for facilitating spatial displays of the network) and the average land value and trip-making demand associated with the subarea represented by the given node. Associated with each link are the travel times that correspond to each of the competing travel modes (i.e., walk, automobile, and transit). When this network information and some additional transit-vehicle-specific data (i.e., vehicle types, capacities, and operating costs) are available, a transit system may be designed (essentially by overlaying the design on the base network), the mode-choice--network-assignment procedure can be executed for the given design, and the various performance measures can be computed and displayed.

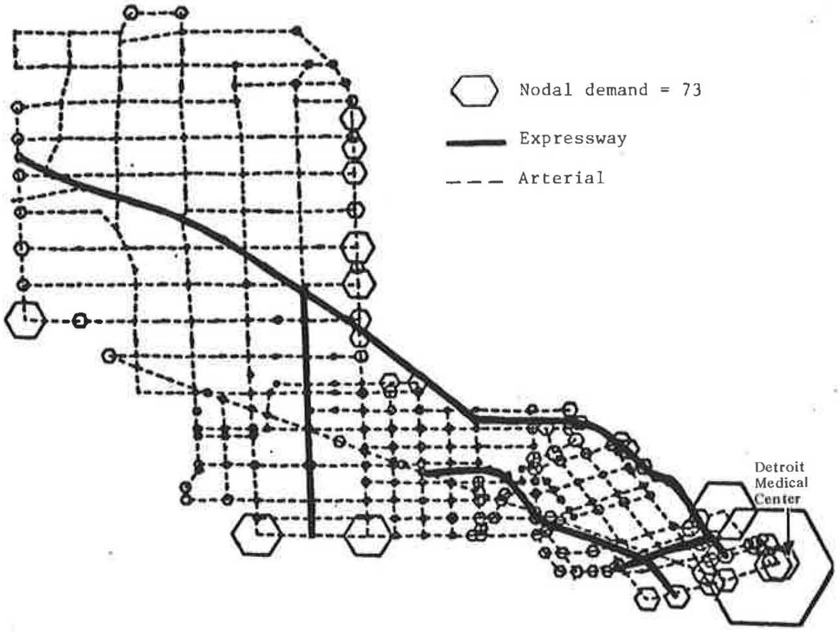
The primary tasks involved in developing the IGTDS data base for this case study were found to be (a) the creation of a link-node network at a suitable level of detail for the application, (b) the assignment of transit and automobile link travel times, and (c) the assignment of trip origins to the network nodes. The methods used to carry out these tasks are described in detail in a GMTSD report (8).

The entire IGTDS data-base development phase for this study required approximately three person weeks of a graduate student's effort. This requirement can be expected to vary in other cases depending on the problem size and the availability of data and data-processing capabilities. However, it appears that in most planning environments the level of effort required for this task would not exceed one person month.

SYSTEM APPLICATION

Following the data-base development phase, IGTDS was applied by a group of Northwestern University graduate students to aid in solving a bus system design problem. Six design teams of two to three students each were formed, and each was asked to design a bus system to serve peak-period trips that orginated in the Southfield-Jeffries Corridor and terminated at the Detroit Medical Center. An explicit procedure was specified for evaluating the bus system design solutions, and each team attempted to develop the best possible solution under this scheme. The teams were each given two 2-h on-line sessions in which to use IGTDS to develop their designs. Prior to the on-line design sessions, the entire group met for a 2-h instruction session and a 2-h on-line demonstration of IGTDS. Each team worked independently on the problem, and competition among teams was encouraged. None of the participants had had any prior experience in using IGTDS.

Figure 2. IGTDS representation of network and demand data for Southfield-Jeffries Corridor.



The formal definition of the design problem (as specified to the design teams) consisted essentially of the following three elements:

1. Network and demand characteristics,
2. Trip-maker behavioral assumptions, and
3. Design evaluation procedure.

Network and Demand Characteristics

The network and demand characteristics of the study area were basic elements in defining the spatial structure of the problem. The area of the Southfield-Jeffries Corridor (Figure 2) is approximately 347 km^2 (125 $miles^2$). The total peak-period demand for trips to the medical center for this problem was 4158. As indicated in Figure 2, the demand at each node is represented by a hexagon whose size is proportional to the magnitude of the demand.

Trip-Maker Behavioral Assumptions

The IGTDS mode-choice--network-assignment model divides all travel to the destination point among the three modes available (i.e., drive, walk-and-ride, and park-and-ride). The share of trip makers who choose a particular mode is based on the relative disutility to travel by that mode. The disutility for a given mode is described by a linear disutility equation that has behavioral coefficients assigned to each component of a trip by that mode. The disutility equations for each mode for the case-study design problem were known explicitly by each of the student designers.

Design Evaluation Procedure

In order to guide the development of the bus system designs, a specific evaluation procedure was devised. The procedure was quite simple and facilitated rapid evaluation of alternatives during the on-line design sessions. The form of the evaluation scheme is discussed below.

The principal objective for designing the bus system was that of maximizing the number of transit patrons. The performance standards represented minimum and maximum values of the performance measures for other important transit objectives (i.e., those that pertained to net operating loss, accessibility and equity of service, and user costs) to ensure that these objectives were fulfilled to an acceptable degree. The design constraints consisted of hardware and software limitations (i.e., maximum number of bus stops, bus lines, and park-and-ride lots allowed by the model) and environmental restrictions (i.e., bus fleet size characteristics, average destination parking fee and walk-access time, and daily park-and-ride lot operating cost). The overall quality score for a given bus system design was thus equivalent to the total number of transit patrons who used the system; however, a score of zero was given if the acceptable standards were not satisfied.

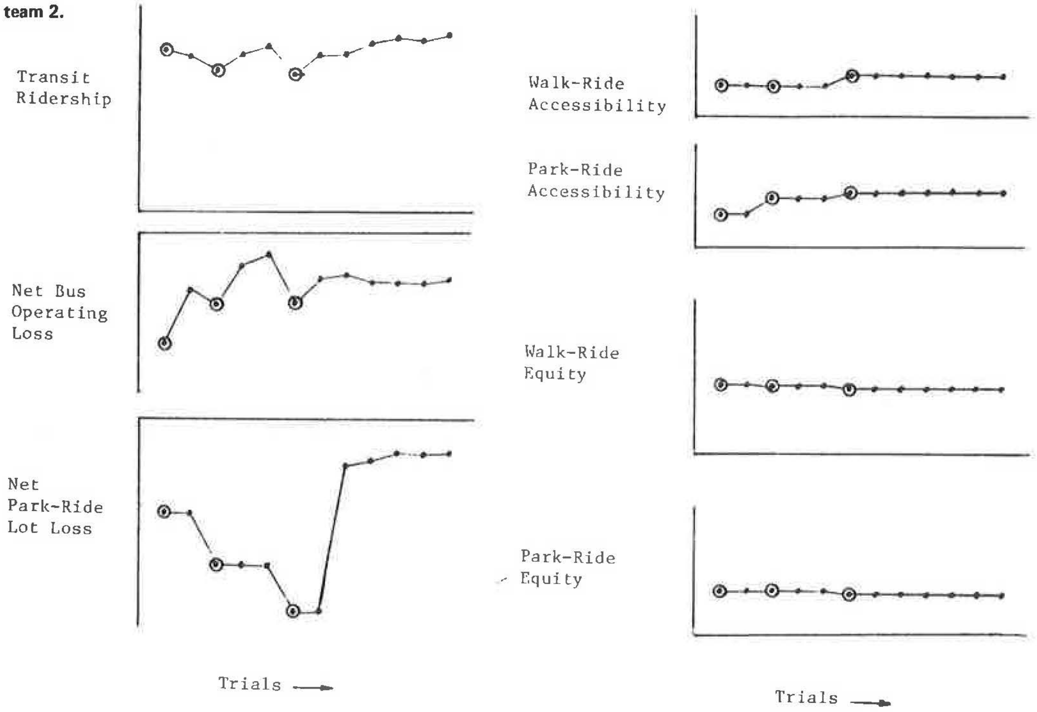
RESULTS

The results of the case-study application are presented below in terms of the project objectives outlined in the introduction to this paper.

General Problem-Solving Experience by Using IGTDS

The behavior of the individual teams during the on-line design sessions was quite diverse and interteam comparisons are difficult. Generally, the teams spent the first of their two on-line sessions in an exploratory manner, becoming familiar with the operation of IGTDS and gaining insights into the relationships among the design variables and performance measures. No team showed evidence of a well-defined plan of attack, although most teams had thought out their first-cut designs to some extent prior to their on-line time. All first-cut designs exhibited poor performance in terms of accessibility to users, since most teams either had underestimated the required density of bus stops or had failed to locate the stops strategically (i.e., near large demand clusters). Subsequent trial configurations improved the accessibility attributes so that they eventually surpassed the acceptable standards. Each team tried from one to three different route configurations and tested from two to six sets of operating and pricing policies (i.e., lot sizes and

Figure 3. Performance profiles for team 2.



fees, bus frequencies, and bus fares) on these configurations during their first sessions. No design team achieved a feasible solution (i.e., a solution that met the acceptable standards) during their first session.

The design teams appeared to be more systematic in their second on-line sessions than they had in their first. Each team had had a chance to study the hard-copy prints of their design decisions and concomitant performance characteristics from the first session. Each team was now familiar with the use and capabilities of IGTDS. Hence, the teams entered the second session with clearer strategies for laying out their routes and for making subsequent modifications to their designs. For example, the two teams that eventually developed the highest-performing designs each formulated a strategy of implementing short high-capacity express park-and-ride routes to the destination. These two teams subsequently concentrated their efforts on modifying the operating and pricing policies for those routes.

In order to monitor its progress during each on-line session, each team plotted the values of the performance measures for successive trial designs. Typical performance plots, or profiles, are illustrated in Figure 3 for team 2. As can be seen, significant shifts in performance generally occurred when a new route configuration was attempted; this is indicated by the circled points on the profiles in Figure 3. As for the accessibility and equity measures, variations occurred only when there were new route layouts (as expected) and these were generally small unless radical route changes were implemented. For ridership and net system losses, variations in performance occurred for nearly any type of modification; this included changes in the system operating and pricing policies.

For every team, the best design was obtained on the last trial. The teams were each able to produce from 7 to 11 feasible designs, and in all cases these constituted more than half the total number of trial designs by each team. A total of 81 trial designs was developed by all teams combined; 50 of these were considered feasible.

The characteristics of the final or best designs of each team were quite diverse, but similarities did appear to exist along some dimensions. Visually, all the designs were complex. Figure 4 provides a graphic example of one of the final designs. Each of the designs consisted of some routes along the southern and eastern perimeters of the corridor, and all exhibited a greater density of stops and routes near the medical center (where the distribution of demand was denser). The physical attributes, or design-parameter quantities, of the final designs are listed in Table 1. As can be observed, there was no variability in the number of buses employed in each design, since the constraint on bus-fleet size was binding in all cases. The only other constraint that was binding was the one placed on the number of bus stops, but this was not a limitation in all cases (e.g., the design produced by team 2 employed only 39 stops, or 10 fewer than were allowed). In terms of the remaining three physical attributes, there was a greater degree of variability among the six designs. The differences in these attribute values were significant and gave an indication of the wide range of solution paths taken by each team.

In terms of the performance attributes of the final designs, the values for total estimated transit patronage ranged from 1432 (i.e., transit mode split = 34.4 percent) to 2010 (i.e., transit mode split = 48.3 percent). Below is a list of the estimates of transit patronage by each design team.

	Total Transit Patronage		Walk-and-Ride	Park-and Ride
Team	No.	Percentage of Total	(%)	(%)
1	1432	34.4	10.3	24.1
2	1733	41.6	15.8	25.8
3	2010	48.3	10.5	37.8
4	2007	48.3	14.1	34.2
5	1491	35.8	18.8	17.0
6	1483	35.6	17.1	18.5

Figure 4. Route display of a final design.

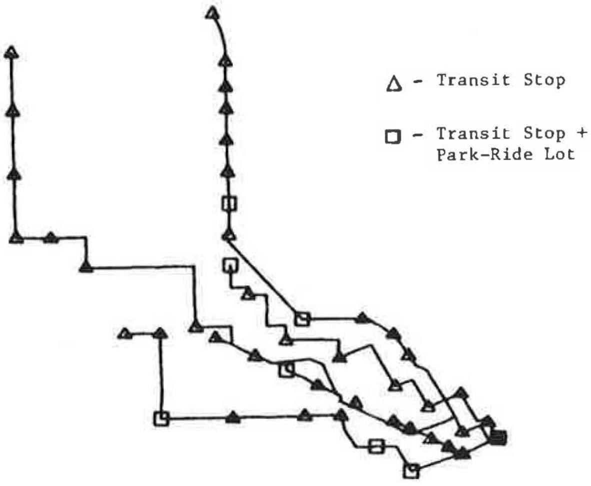


Table 1. Physical attributes of final designs.

Team	No. of Stops (constraint ≤ 49)	No. of Lines (constraint ≤ 20)	No. of Buses (constraint ≤ 40)	No. of Lots (constraint ≤ 20)	No. of Stalls (no constraint)
1	47	6	40	7	1084
2	39	7	40	8	1200
3	46	5	40	12	1718
4	46	6	40	7	1470
5	46	7	40	10	950
6	49	9	40	9	940

In all but one case, the estimated share of the park-and-ride mode was greater than that for the walk-and-ride mode, and the share of park-and-ride users was generally greater for the designs that attracted greater total transit ridership. The reason for this was that the higher-performing designs (i.e., those developed by teams 2, 3, and 4) took advantage of the utility of a short large-capacity express park-and-ride route near the destination. The other designs did not exploit this resource; rather, park-and-ride demand was distributed in these designs to several smaller-sized lots located in areas farther out from the destination, at which the disutility for the park-and-ride mode was greater (since bus rides were longer). Hence, lower patronage estimates were achieved in these latter designs. In general, it was difficult to draw strong inferences as to the relationships between the design variables and performance results, since so many interaction effects were taking place and so many possible combinations of route structures, operating policies, and pricing policies (including zone fares) were attempted.

Design Efficiency Characteristics

Two basic questions posed prior to conducting the case-study application were how efficient the design process was by using IGTDS or, more specifically, whether use of IGTDS encouraged systematic development of improved transit designs. To address these questions, the performance profiles (see Figure 3) for each team were analyzed. From examining the curves of the ridership profiles, it was found that the ridership scores increased monotonically for each team once the initial feasible designs had been achieved (i.e., the first designs that satisfied all the performance standards and design constraints). To measure the amount of improvement in design quality experienced by each team, the ridership levels of the final designs were compared with those of the corresponding initial feasible designs.

In terms of percentage of improvement from the initial ridership levels, the teams averaged an improvement of 26 percent from their initial feasible scores. Team 1 appeared far superior to the other teams; they had an improvement score of 60 percent. It would not be strictly correct to assert that team 1 was more efficient than the other teams in improving its design, however, since the values of the initial scores of each team significantly influenced their efficiency scores. For example, team 1, although it had the highest improvement score, had the lowest initial quality score. Thus, team 1 had theoretically a greater capacity for improvement than did the other teams. In general, the teams that had smaller initial quality scores tended to improve their designs by a greater percentage than did the teams that had larger initial quality scores.

On-Line Time and Cost Characteristics

Detailed information on user and computer time costs was recorded during all on-line sessions. The time costs to users for each on-line design session were broken down into three components--interaction time, thought time, and error time. The interaction time, or input and output time, was the time during which the teams were either transmitting information to the computer via the terminal keyboard (input) or receiving information via the cathode-ray tube screen (output). The thought time was the time during which the teams were contemplating their design decisions and were not interacting with the computer. The error time represented the time lost due to user input errors, such as mistyping a key or issuing commands in an improper sequence. The component on-line times are listed in Table 2.

As can be seen, the amount of thought time exceeded the amount of interaction time in nearly all cases. The portion of on-line time devoted to interaction with IGTDS ranged from 27 percent to 49 percent and averaged 38 percent. The portion of on-line time devoted to thought about the problem ranged from 48 percent to 69 percent and averaged 56 percent. The time lost due to error was relatively minimal; it consumed a maximum of 13 percent and an average of 7 percent of the total on-line time. The central-procesing-unit (cpu) time used by each team varied between 2.29 and 5.64 min; the total was 20.55 min. The cpu time required to develop each trial design ranged from 0.174 min to 0.343 min, which averaged 0.255 min per design. The differences in cpu times for each trial were due primarily to differences in the number of interactions (i.e., executions of IGTDS program modules) required to develop each design. The average number of interactions per design was four. These cpu times that characterize the IGTDS design experience were much lower than had been anticipated given the size and complexity of the design problem.

CONCLUSIONS

The data-base development requirements for applying IGTDS in this case study appeared to be relatively moderate (i.e., three person weeks of graduate-student effort), especially considering the size and complexity of the design problem addressed. These requirements also seem reasonable when the unlimited number of alternative design

Table 2. Component on-line times.

	Interaction Time		Thought Time		Error Time	
Team	Minutes	Percentage of Total	Minutes	Percentage of Total	Minutes	Percentage of Total
1	81	33.8	154	64.2	5	2.0
2	90	37.5	127	52.9	23	9.6
3	117	48.8	114	47.5	9	3.7
4	91	37.9	118	49.2	31	12.9
5	97	40.4	124	51.7	19	7.9
6	65	27.1	165	68.8	10	4.1
Mean	90	37.5	134	55.8	16	6.7

Note: Total time for each team was 240 min.

solutions that could have been produced by using the original data base is considered. It is hypothesized that, as use of interactive systems like IGTDS becomes more prevalent, even greater economies can be achieved if a central data base is developed initially for an entire region rather than for just a corridor or subarea. Such a strategy would obviate the need for creating a new data base from scratch each time a corridor study was initiated; rather, the required corridor data base could essentially be extracted from the existing regionwide data base. As tools become increasingly available for automating the acquisition and management of transportation data (e.g., graphic tablets and network editor systems) the data-base development task will decrease even further in significance.

In regard to the application of IGTDS in the transit design process, several important findings were produced. In the identification of a useful problem-solving technique to be applied in conjunction with IGTDS, the results of the design experience provided a number of insights. A reasonably effective design strategy appeared to be one characterized by the following elements:

1. Thorough knowledge of the problem definition,
2. Preparation of initial designs on paper prior to on-line sessions,
3. Review of the results from preceding design sessions prior to each on-line session,
4. Use of systematic incremental design modifications, and
5. Use of quick intuitive judgements in making individual design decisions.

Although the above strategy may provide some guidance in applying the interactive graphics system to the transit design process, it is clear that a more-structured approach to interactive problem solving is desired.

The on-line user-time and computer-time costs associated with the IGTDS design process were much lower than anticipated, and the design efficiency characteristics were much better than expected. The error times of the teams averaged only 7 percent of the total on-line time, which speaks well for the ease of using IGTDS, especially considering the short period of time in which the teams were exposed to the system. The computer connect and cpu time figures that characterize the design process were quite reasonable if we consider the complexity of the design problem addressed. In most automated data-processing environments, these computer use figures would translate into relatively modest monetary charges (i.e., on the order of $5 or less per trial design). In terms of design efficiency, IGTDS enabled the teams to design and evaluate 81 alternative designs in 24 h of on-line time and obtain improvements estimated at close to 30 percent over their initial feasible designs.

In summary, IGTDS was found to be easy to use, the application costs were relatively low in terms of both data-base development and system operation, and the design process appeared to be quite efficient in generating design solutions to a relatively complex bus system problem. These findings indicate the strong potential of an interactive graphics system like IGTDS as a practical transit sketch-planning tool.

ACKNOWLEDGMENT

This study was conducted as part of a Master's thesis research project at Northwestern University. Access to IGTDS and related computer support was provided by the General Motors Transportation Systems Division, and interactive graphics equipment was furnished by Tektronix, Inc. Special thanks are given for these sources of support. I also gratefully acknowledge the contributions of my fellow graduate students who participated in the design experiment.

REFERENCES

1. Interactive Transit Assignment Model. Office of Planning Methods and Support, Urban Mass Transportation Administration, UTPS tape, n.d.
2. J. Hess and A. Kornhauser. Interactive Sketch Planning for Automated Guideway Transit. Transportation Program, Princeton Univ., Rept. 76-TR-13, 1976.
3. C. Gehner and J. Clark. The Urban Region Transit Analysis System (UTRANS)--Vol. 1: Documentation; Vol. 2: User's Manual. Urban Transportation Program, Univ. of Washington, Seattle, Res. Rept. 72-7, 1973.
4. Transportation Network Evaluation System (TRANES). Comprehensive Planning Organization, San Diego, CA, Tech. Rept., Feb. 1978.
5. Interactive Graphic Transit Design System. Office of Planning Methods and Support, Urban Mass Transportation Administration, UTPS tape, n.d.
6. M. Rapp. Transit System Planning: A Man-Computer Interactive Graphic Approach. HRB, Highway Research Record 415, 1972, pp. 49-61.
7. P. Rassam, R. Ellis, and J. Bennett. The n-Dimensional Logit Model: Development and Application. HRB, Highway Research Record 369, 1971, pp. 135-147.
8. M. Couture. Interactive Graphic Transit Design System (IGTDS): Test Application. GM Transportation Systems Division, General Motors Technical Center, Warren, MI, EP-77111, 1977.

Interactive Graphics Model for Testing Stochastic Traffic Assignment Algorithms

JEROME M. LUTIN

This paper reports on the use of interactive computer graphics to investigate properties of a stochastic traffic assignment algorithm used to model pedestrian flow in the Urban Mass Transportation Administration's transit station simulation model (USS). The use of an interactive computer language (APL) to transform networks and models into easily manipulable matrix forms is discussed. Graphical calibration of model parameters is discussed and illustrated by using computer plots. The development of an interactive network builder and stochastic traffic assignment model is reported. The exploration of various model forms through interactive graphics is discussed and illustrated by using computer graphics plots of typical networks.

There are many instances in the design of transportation facilities and pedestrian networks in which one would like to estimate the volume of pedestrians who will pass a given point. In recent years, this question has most often been asked by transit station planners and designers who have traditionally had few analytical tools to aid in estimating pedestrian flow within stations. Recently, however, the Urban Mass Transportation Administration (UMTA) has sponsored development of a transit station simulation model (USS) to model pedestrian flow and to determine at what point queues will form. An innovative multiple-path traffic assignment model is incorporated into the USS program to simulate pedestrian flow. This paper reports on interactive computer research that is leading to improvements in the model.

Most traffic assignment models use minimum-path algorithms to simulate the choice of route by the driver or shipper. However, these models do not appear to be very realistic in their replication of human behavior, especially that of pedestrians. First, they typically yield routes structured only as trees that have one path per origin-destination (O-D) pair. Second, they apply the same cost-minimizing rule at each decision point regardless of the choice set. Third, they always select the minimum-cost route, assuming perfect knowledge of costs and perfect rational decision making by the driver or shipper.

In many instances, however, a pedestrian does not have perfect information about the route ahead. He or she must anticipate congestion and queueing and attempt to guess at the best available route. In addition, there may be several routes that appear to have equal costs between O-D pairs. Empirical work has indicated that for many types of flows there tends to be a set of routes used between O-D pairs (1,2). As noted by several researchers (1,3,4), distance estimation is subject to distortion and the minimum-time path is not always chosen by travelers.

MULTIPLE-PATH MODELS

In recent years, a number of researchers have developed methods of allocating traffic flows to competing routes (5-7). Of these so-called "stochastic traffic assignment" algorithms, Dial's analytical method, which is based on a multinomial logit formulation, has attracted a considerable amount of attention. Although a number of criticisms have been brought forth concerning the ability of this model to produce reasonable results (8-10), it lends itself to some interesting behavioral interpretations. The model has been implemented in UMTA's recently released USS model to simulate pedestrian path choice in transit stations. Dial's multiple-route traffic assignment model takes the form shown below:

$$P_i = \left[1 + \exp(\theta t_i) \cdot \sum_{j=1}^{n} \exp(-\theta t_j)\right]^{-1} \quad (1)$$

where

P_i = probability path i will be chosen,
exp = base of Naperian logarithms,
θ = parameter,
t = travel time,
j = family of competing paths (j = 1, ... , n), and
$i \neq j$.

Basically, the model allocates a percentage of all trips between an O-D pair to each efficient route based on the travel time for that route. Efficient routes are based on the calculation of an unconditional arc likelihood A(e) that a given link in the network will be traversed, as shown in Equation 2:

$$A(e) = \begin{cases} \exp\theta\,[P(j) - P(i) - T(i,j)] & \text{if } P(j) > P(i) \\ 0 & \text{otherwise} \end{cases} \quad (2)$$

where

A(e) = arc likelihood,
(e) = link index for (i,j) pair,
P(j) = shortest path distance from origin to node j,
P(i) = shortest path distance from origin to node i, and
T(i,j) = length of link from i to j.

MODELING PEDESTRIAN FLOWS IN TRANSIT STATIONS

Research by Lutin (11) on the UMTA USS model showed that the use of Dial's stochastic traffic assignment algorithm was very sensitive to the value of θ used. If θ was set too high, only one path--the minimum path--was used by each individual who moved through the station. This led to the embarrassing result in which all passengers were modeled as leaving a subway train from a single door. Similarly, passengers all tended to queue at only one location on a platform. On the other hand, setting θ too low permitted individuals to wander through the station without much regard to signs or walk time to reach the exit. These effects were perplexing. While the multiple-path model seemed appropriate to model pedestrian paths, it needed to be fine-tuned to model spreading of flows on platforms and concentration of flows elsewhere in the station. In order to test various modifications to the multiple-path model, it was decided to develop an interactive graphics version of the model by using APL, an interactive programming language based on tensor algebra.

INTERACTIVE GRAPHICS IN MODEL RESEARCH

An entire interactive package was developed that

Figure 1. Multipath assignment probabilities as a function of path length and θ.

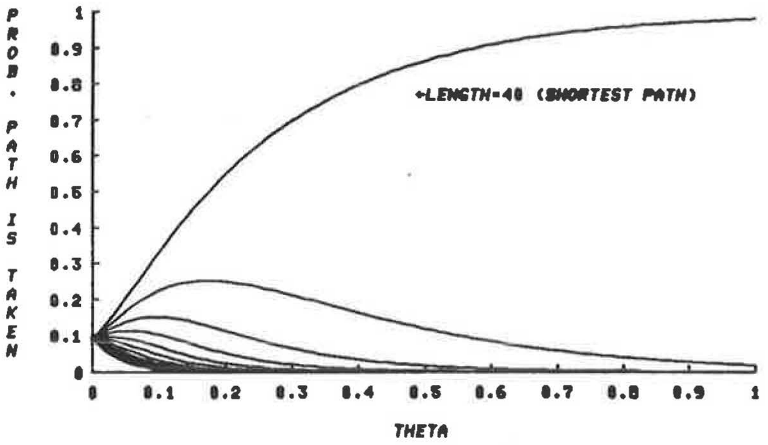
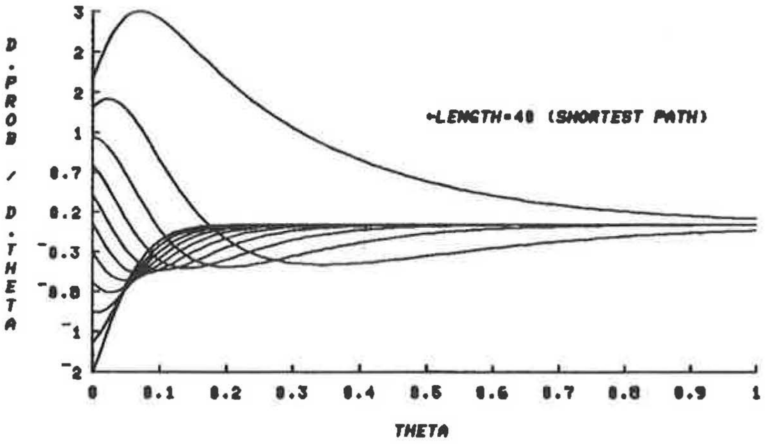


Figure 2. First partial derivative of path choice probability as a function of θ.



permitted one to create test networks, to edit networks, to calculate minimum paths, to input various modifications to the multiple-path model, and to examine output graphically on a cathode-ray tube (CRT) terminal. In this research, graphics was not used solely as pictorial output to display final results. Graphics was used as a means to explore alternatives and to determine whether various program elements were working correctly. Without the use of interactive graphics, this work would have taken considerably longer or might not have been done at all.

The first step in the research was to examine the sensitivity of the model to various settings of θ for a specified set of paths. This was accomplished by creating an interactive plotting program that would permit the user to input a set of paths by length and to specify a range of values for θ. The program would then plot the set of curves and indicate the probability of selecting each path as a function of θ. Figure 1 shows a plot of one set of curves for a set of paths. Each curve represents 1 of 11 paths. Path lengths are 40, 44, 48, 52, 56, 60, 64, 68, 72, 76, and 80 min. In addition, one can find the first partial derivative of the model with respect to θ:

$$\partial P_i/\partial\theta = \left\{1 + \sum_{j=1}^{n} \exp[\theta(t_i - t_j)]\right\}^{-2} \times \left\{\sum_{j=1}^{n} (t_i - t_j) \exp[\theta(t_i - t_j)]\right\} \qquad (3)$$

where $i \neq j$.

Figure 2 shows the plot of the first partial derivative of path-selection probability with respect to θ. For very small values of θ ($0 < \theta < 0.2$), one finds local maxima for the path-selection probability of five of the nonminimum paths. For $\theta > 0.2$, all derivatives are negative, except for the shortest path. Thus, in a very small range of θ, it is possible to divert trips away from the longer paths to several of the next-shortest paths. Discovery of the existence of local maxima for the selection probability of nonminimum paths was made possible by interactively plotting the derivatives for a set of paths simultaneously.

USE OF APL IN INTERACTIVE NETWORK MODELING

As noted earlier, the interactive language used for this work was APL, a tensor algebra-based language that permits one to treat and manipulate vectors and matrices as easily as scalars. The power of APL lies in five areas. First, one does not need to deal with matrices on an element-by-element basis, as in most other languages. Most primitive functions (e.g., +, −, x, ÷) can be applied to matrices and vectors as well as to scalars. Second, interactive use is accomplished through an APL interpreter, which permits one to halt programs in the middle and even rewrite them or change data before continuing execution. Both data and programs are accessible simultaneously within a single workspace, and virtual-storage computers permit one to use up to 16 000 000 bytes of direct-access virtual storage on-line for the manipulation of large files of data. Third, the language has developed the concept of the linear operator, or generalized operator, which can be applied to vectors or matrices by using a two- or three-character symbol. By using these generalized operators, it is possible to manipulate networks in symbolic ways and map scalars, vectors, and matrices onto other vectors and matrices quite simply. Fourth, logical operators are developed in APL in a manner that permits filtering matrices through Boolean algebra. Finally, the data structures generated by APL are amenable to computer graphics display without much transformation.

A discussion of the application of APL linear operators to network modeling follows. Two specific algorithms were developed; one determined minimum costs of travel between nodes, and a second performed the arc-likelihood calculations found in Dial's algorithm. By using these matrix techniques, it became clear that forming a diagonal matrix of parameters was a convenient way of performing the calculations in APL.

By creating a diagonal matrix of θ-parameters, it became obvious that one could insert a unique value of θ for each node. Thus, in areas of the station in which one wished to simulate spreading of flows (such as on platforms), one could supply low values of θ. At key junctions and corridors in which persons might be directed by signs to the shortest paths, one could simulate a greater sensitivity to distance by increasing θ at selected nodes.

Algorithm to Determine Minimum-Path Cost

Lutin developed the following algorithm for finding the minimum-path cost and implemented it interactively in APL computer code on an IBM 370/158 computer (12). Basically, it is derived from the algorithm shown by Aho, Hopcroft, and Ullman (13) for transitive closure of a semiring. It is of interest not because it is efficient, but because it preserves the algebraic properties of the algorithm in its translation to APL.

The algorithm begins with the definition of a generalized linear operator $@_1 \cdot @_2$ so that for vectors V and V' of length n,

$$V(@_1 \cdot @_2)V' = S^* \tag{4}$$

where S* is a scalar. Likewise, for square (n x n) matrices, M and M' are as follows:

$$M(@_1 \cdot @_2)M' = M^* \tag{5}$$

where M* is a square matrix of order n. The operation shown in Equation 5 is termed a generalized inner product (14). Expanding on the operation of $@_1 \cdot @_2$, $@_2$ is specified as +, the element sum of two vectors V and V^T of length n, so that

$$\begin{bmatrix} V_1 & @_2 & V_1{}^T \\ V_2 & @_2 & V_1{}^T \\ . & . & . \\ . & . & . \\ . & . & . \\ V_n & @_2 & V_n{}^T \end{bmatrix} = \begin{bmatrix} V_1 & + & V_1{}^T \\ V_2 & + & V_2{}^T \\ . & . & . \\ . & . & . \\ . & . & . \\ V_n & + & V_n{}^T \end{bmatrix} = V^* \tag{6}$$

The first element of the linear operator $@_1$ is specified as a monadic operation (L) on vector V* so that

$$\min V^* = L(V^*) \tag{7}$$

where L(V*) is known as the floor of V* and represents the smallest element of V*. Thus the operator L.+ is called the minimum-cost operator. Given a matrix of costs (C), the algorithm establishes a vector of sums that represents the costs (i,k) + (k,j) of travel from node i to node j via each node k and selects the minimum cost, which is placed in the i,jth cell of C*. The operation is expressed below:

$$CL. + C = C^* \tag{8}$$

The operation shown in Equation 5 is repeated and C is replaced by C* in subsequent iterations:

$$CL. + C_{i-1}{}^* = C_i{}^* \tag{9}$$

It can be shown that C* is closed under the operation L.+, as in Equation 9. By using this property of closure, Equation 9 is repeated until $C_{i-1}{}^* = C_i{}^*$; that is, the matrix C* is unchanged for further iterations. C* is said to be idempotent. In APL, the entire algorithm is implemented in only six lines of code; Equation 9 appears in the code as a single line and is virtually identical to the algorithmic form presented here.

Dial's Algorithm

Turning to Dial's multiple-path algorithm, it too was transformed to matrix form by using linear operators as found in APL. In this case, it was necessary to use an outer-product operator.

Define : @ as a generalized outer product operator (14) so that, for vectors V and V' of length n,

$$V : @ V' = \begin{bmatrix} V_1 @ V_1{}' \ldots V_1 @ V_n{}' \\ . \quad . \; . \; . \; . \; . \; . \; . \; . \quad . \\ . \quad . \; . \; . \; . \; . \; . \; . \; . \quad . \\ . \quad . \; . \; . \; . \; . \; . \; . \; . \quad . \\ V_n @ V_1{}' \ldots V_n @ V_n{}' \end{bmatrix} \tag{10}$$

Let V be a vector of distances from the destination node d to all other nodes j. Let @ be a dyadic operator (-). Then

$$D_d = [V : -V] = [q(i) - q(j)] \quad \text{for all } i, j \in N \tag{11}$$

where

D_d = matrix of cost differences with respect the destination node d,
q(i) = shortest-path distance from node i to destination node d,
q(j) = shortest-path distance from node j to destination node d, and
N = set of all nodes, N = 1, ..., i,j,n.

Let C be the cost matrix for the network so that

$$[c_{ij}] = \begin{cases} c_{ij} & \text{if } (i,j) \in E \text{ and } i \neq j \\ 0 & \text{if } (i,j) \in E \text{ and } i = j \\ \infty & \text{if } (i,j) \notin E \end{cases} \tag{12}$$

where

c_{ij} = cost or time of travel between nodes i and j,
i,j = node numbers,
(i,j) = link that connects node i to node j, and
E = set of all links (i,j) in the network.

Further, let θ be a diagonal matrix of the sensitivity parameters for each node i so that

$$\theta_{ij} = \begin{cases} \theta_{ij} & \text{when } i=j \\ 0 & \text{otherwise} \end{cases} \tag{13}$$

Then θ is shown as follows:

$$\theta = \begin{bmatrix} \theta_{1,1} & 0 & \cdots & & 0 \\ 0 & \theta_{2,2} & & & \cdot \\ \cdot & & \ddots & & \cdot \\ \cdot & & & \ddots & \cdot \\ \cdot & & & \cdot & 0 \\ 0 & \cdots & & 0 & \theta_{n,n} \end{bmatrix} \tag{14}$$

The arc-likelihood calculation can now be restated:

$$A(E) = \exp\,[\theta \cdot (D_d - C)] \tag{15}$$

The calculation $(D_d - C)$ represents the elementwise subtraction of the two matrices and is equivalent to $\{[q(i) - q(j)] - T(i,j)\}$, the arc likelihoods in Equation 2. A(E) represents the matrix of arc likelihoods and is the matrix product of θ and $(D_d - C)$. To test for the necessary condition $q(i) > q(j)$, a Boolean matrix, $D_d > 0$, is created. Likewise, another Boolean matrix is created to test for connectedness $(C \neq \infty)$. These matrices are used to form the elementwise product of $A(E)^*$, as follows:

$$A(E)^* = (C \neq \infty) \times (D_d > 0) \times A(E) \tag{16}$$

GRAPHICAL TESTING OF MODEL RESULTS

In order to see how well the model worked and to determine how difficult it would be to calibrate, it was decided to create a test network by using an interactive-network building program. The interactive-network builder permitted one to enter links and nodes graphically by using a crosshair cursor controlled by thumb wheels on a Tektronix 4013 CRT terminal. Link costs were automatically calculated as the Euclidean distance between points on the Tektronix screen or entered separately as a vector of link costs. One-way links could be specified if desired.

An interactive program was created in which the user was asked to specify an O-D node pair. The user was queried to supply the value of θ desired. One could input a single value of θ for the entire network or a vector of θ-parameters that assigned individual values to each node. The program permitted the user to display various link data, such as minimum-path costs, A(e) values, or the percentage of travelers that used a given link. The user was queried to supply a title for each graphic display.

After receiving optional link-cost data from the user, the program calculated the minimum-cost matrix by following the algorithm described above. When possible, the program alternated between executing various functions and querying the user for information. This technique permitted us to cut down the length of the periods when the machine was processing data and appeared dead to the user. In addition, it made the execution seem faster, since the user was thinking about responses to program queries while the program was being executed.

After calculating the minimum paths and obtaining the desired O-D node pair and θ-value, the program calculated the probability of using each link in all feasible paths from origin to destination and stored the results in a vector. Prior to the display of the results, the user was asked to specify the title. Thus, immediately after receiving this last user input, the program began to draw the results on the screen; this emphasized the rapidity of response.

In the display, nodes were indicated by circles; the node number was displayed in the circle. Links were drawn as lines that connected each node. Passenger flow volumes were posted at the center of each link. To aid in visual interpretation of results, rectangles were drawn to the right of each link so that the width of the rectangle was proportional to the expected passenger flow volume. Parameter values were posted above and to the right of each node. To calibrate the model, one examined the display from a run and changed the vector of θ-values that had been used in the previous run.

STATION NETWORK CALIBRATION EXAMPLE

Figure 3 (11) shows a computer-drawn plan of a typical transit station; a platform, a concourse, and several possible exit corridors are shown. In Figure 4 (11), this station layout is shown as a computer-drawn network representation that has 22 nodes that are required for analysis by the simulation model. Node numbers are circled. The train is represented by node 1; train doors are indicated by nodes 2-9. Links that connect the train node 1 to the door nodes have been given dummy lengths of zero. In this example, only the flow of passengers from the train to the exit (node 21) is modeled. Figure 5 (11) shows an assignment of passenger flows by using a single parameter value, θ = 1, at all nodes. Flow volumes are shown by rectangles of proportional widths, and the percentage of flow is posted at the midpoint of each link. This figure shows a minimum-path all-or-nothing assignment; all passengers are represented as leaving the train via one door. No spreading of flow among train doors occurs.

In Figure 6 (11), the simulation was rerun, again with a single parameter (θ = 0). All efficient paths from node 1 to node 21 are included. Passenger flows are distributed fairly uniformly across all train doors and the platform. However, in the corridors that lead out of the station, one sees that the main diagonal corridor, which is the shortest route from the concourse to the exit for most passengers, receives only 33 percent of the flow. In addition, nodes 10 and 11, which represent train doors, receive smaller flows than the other door nodes.

In Figure 7, multiple-parameter assignment has been run. Parameter values are set to θ = 1 at nodes 12, 18, 19, 20, 21, and 22 and to θ = 0 at all other nodes. In this example, most detraining passengers are distributed uniformly across the platform and use the main corridor to exit the station. However, the distribution of passengers who leave the train is still not uniform, and passengers detraining at node 10 are shown backtracking to nodes 19 and 20. In Figure 8, parameters were increasd to θ = 1 for nodes 10 and 11 and decreased to θ = 0 for node 19. In this example, an equal volume of passengers leaves from each train door. Those detraining from doors at nodes 4 through 9 are all routed through the main concourse and the major diagonal corridor. Passengers detraining at nodes 10 and 11 are all routed out the side corridor via node 20. Thus, by selectively changing parameter values, one can fine-tune the station simulation to more accurately model passenger flows.

One of the major problems with the original Dial model is the fact that the parameter θ produces the same probabilities for a set of paths that differ by similar increments of length regardless of the lengths of the paths. For example, the selection probabilities for a set of three paths with

Figure 3. Transit station plan.

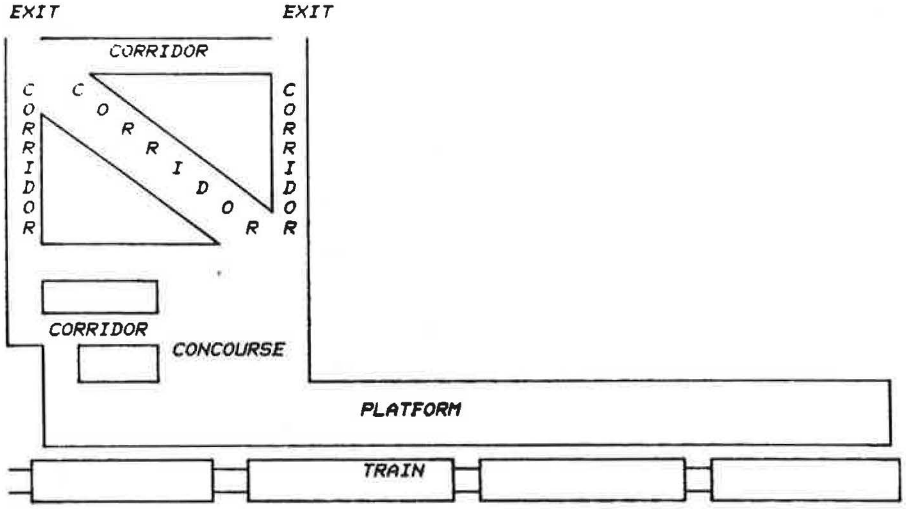
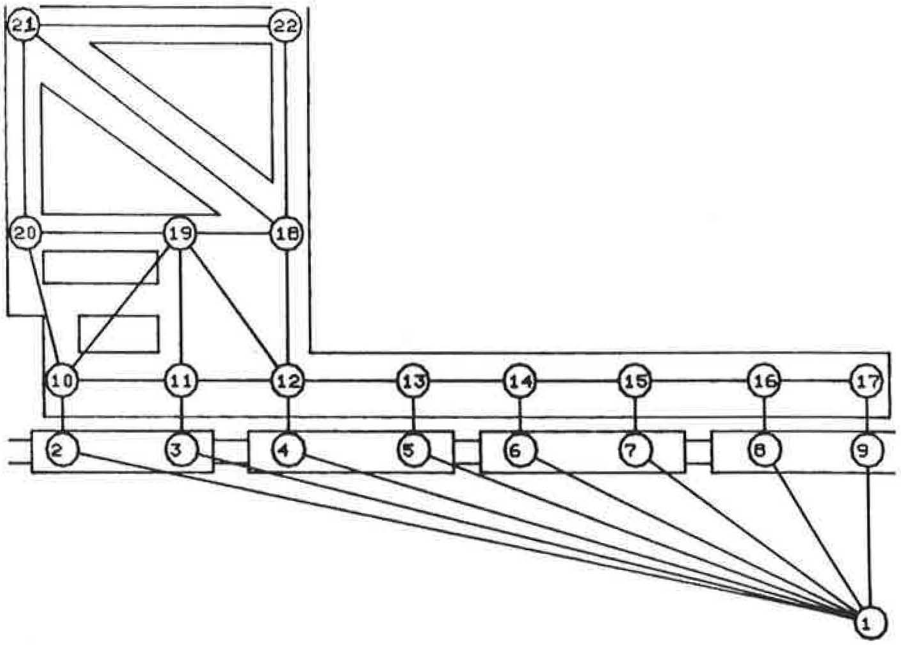


Figure 4. Network representation of transit station.



respective lengths of 5, 6, and 7 min are identical to the path-selection probabilities for paths 15, 16, and 17 min long, respectively. According to Dial's model, for any value of θ, the same proportion of individuals who choose the 17-min path, which is 13 percent longer than the shortest path, will choose the 7-min path, which is 40 percent longer than the shortest path in that set of choices. However, in the multiple-parameter model, each set of links that emanates from a node can be evaluated by using a parameter value appropriate to the set of choices at that node. Although this multiple-parameter formulation still contains some of the undesirable properties of the original model, it can be used more flexibly and calibrated much more easily.

REDUCING THE IMPACT OF ROUTE OVERLAP

One of the most severe criticisms of Dial's model is that unreasonable assignments are obtained when routes overlap, especially when these competing routes have identical or nearly identical travel times (_7_). By using a well-worn example (Figure 9), one would expect flow that emanates from node 1 to be split equally between link 1-7 and link 1-2. However, Dial's model produces twice as much flow on link 1-7 because this link appears in two distinct paths (1-7-6-5-4 and 1-7-6-8-4) as shown in Figure 10 (flow volumes may not total 100 due to rounding).

Essentially, in calculating the link weights, the model is calculating the binomial distribution of all possible paths between origin and destination, and the weight calculated for each link represents the number of feasible paths in which that link participates. This effect can be ameliorated by introducing a weight (λ_{ij}) for each link that is proportional to the number of paths that cross that link. Let Ψ^p be an n x n matrix (where n is the number of nodes) that represents the set of feasible links (i.e., "efficient links" in Dial's terminology) for the set of possible paths between O-D pair p. The elements ($\psi_{i,j}^p$) of Ψ^p are as follows:

Figure 5. Station network with minimum-path flow from node 1 to node 21.

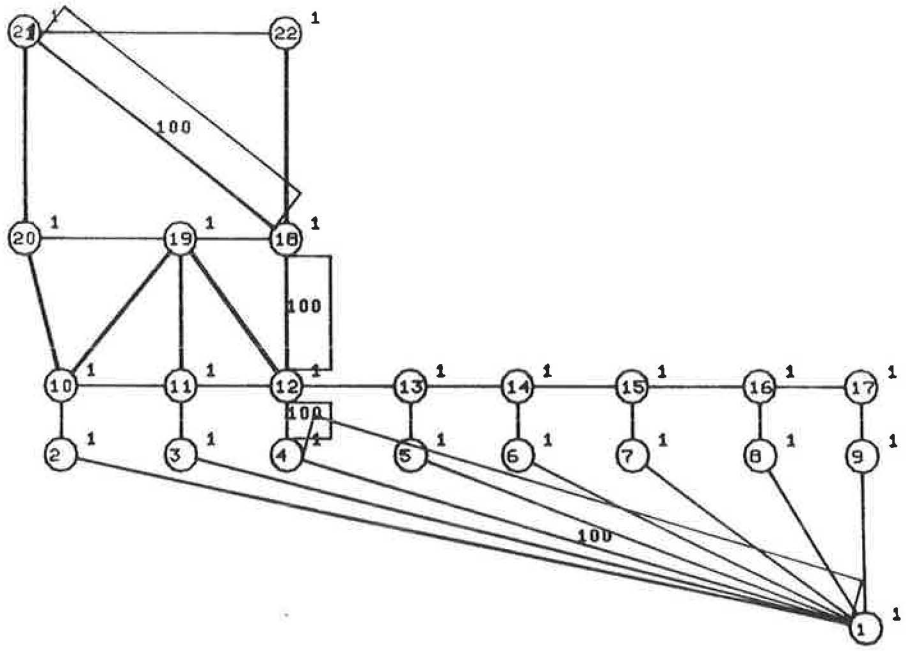


Figure 6. Station network with loading on all efficient paths.

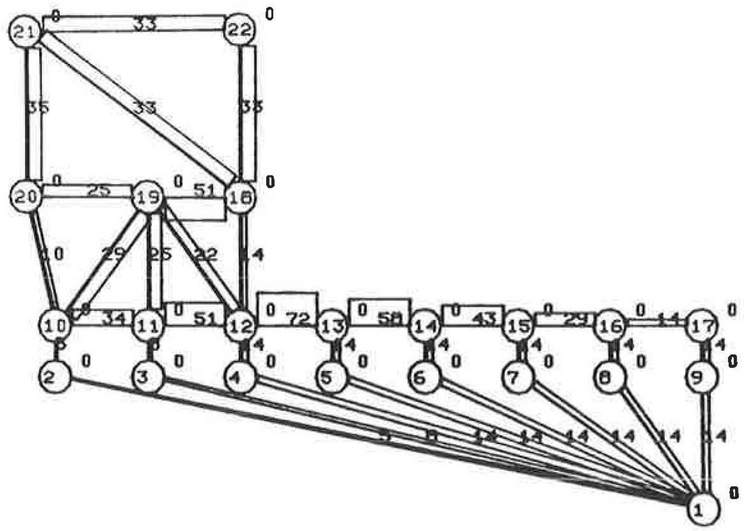


Figure 7. Multiple-parameter assignment (first run).

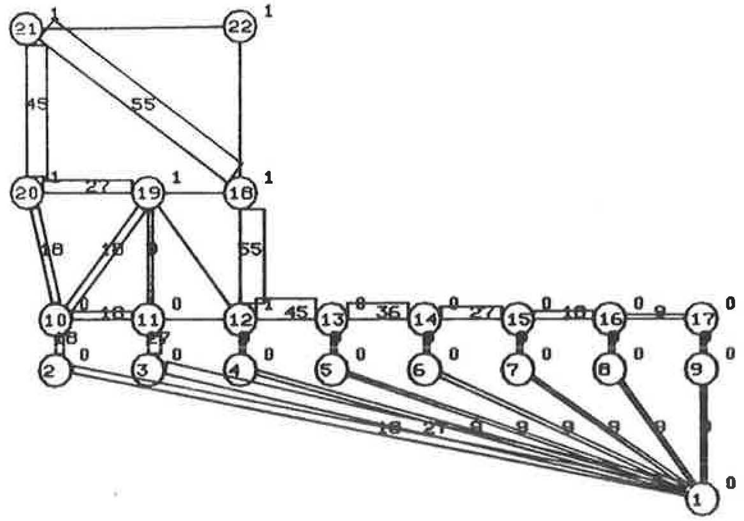


Figure 8. Multiple-parameter assignment (second run).

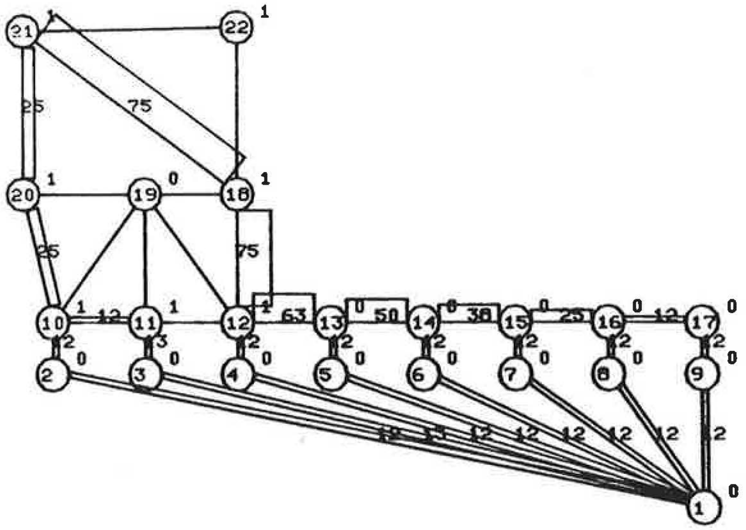


Figure 9. Expected flows from node 1 to node 4.

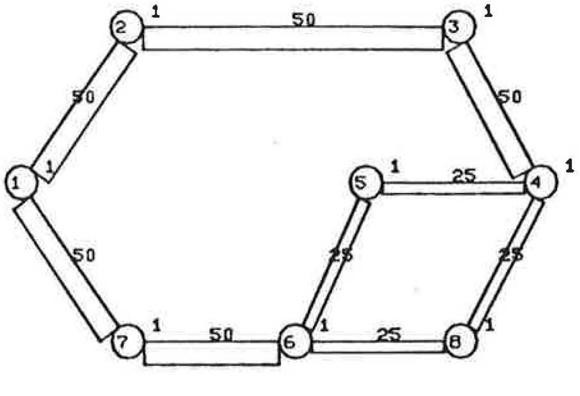
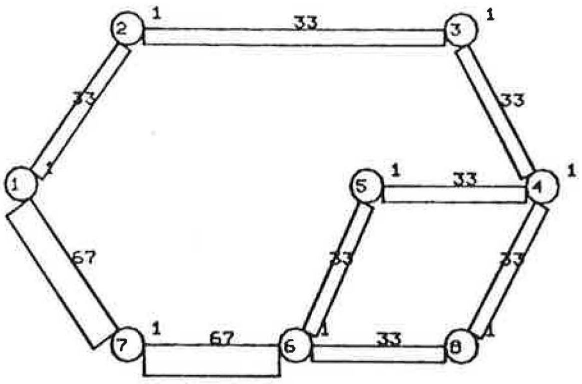


Figure 10. Flows generated by Dial's algorithm from node 1 to node 4.



$$\psi_{i,j}{}^{p} = \begin{cases} 1 & \text{if } A(E)_{i,j}{}^{*} > 0 \\ 0 & \text{otherwise} \end{cases} \tag{17}$$

where A(E)* is the n x n matrix of arc likelihoods as in Equation 9. Then, for each node i in N, let

$$\lambda_i = \sum_{j=1}^{n} \psi_{i,j}{}^{p} \quad \text{for all } i,j \in N \tag{18}$$

A new set of weighted arc likelihoods Γ(E) that has elements $\gamma_{i,j}$ can be calculated as follows:

$$\gamma_{i,j} = A(E)_{i,j}{}^{*} / \lambda_j \tag{19}$$

The remaining calculations follow Dial's algorithm; weighted arc likelihoods Γ(E) are substituted for A(E), the conventional likelihoods.

Figure 11 shows a network that contains three diversion points; the loadings were calculated by using Dial's algorithm. One would expect the upper route (1-8-7) to attract 50 percent of the trips, given that two paths of equal length emanate from node 1. Dial's model, however, finds four distinct paths through the lower set of links and allocates 80 percent of the trips to link 1-2. Figure 12 shows the results of the modified version in which both upper and lower routes share equally in the traffic.

Figure 13 shows a slight modification of the network in Figures 11 and 12. Here, a shortcut, link 9-10, has been added. The value of θ has been reduced to 0.02 to permit multiple-route simulation. (If θ = 1 in this example, only the minimum path would be selected.) Note that only 33 percent of the trips are assigned to the upper route, and 42 percent take the shortcut. For the sake of a contrived example, we assume that 50 percent of the observed flow travels the upper route, whereas everything out of node 2 follows the minimum path, but at node 9 traffic splits equally among all paths to the destination. To achieve flow concentration at node 2, θ_2 is increased to 1, and, to achieve equal dispersion at node 9, θ_9 is set to zero. Through iterative analysis (i.e., trial and error by using interactive graphics), it is found that adjusting θ_1 downward to 0.0085 will balance the arc weights of all paths to achieve the desired flow (Figure 14).

CONCLUSIONS AND DISCUSSION

Although many researchers have cast doubt on the ability of Dial's model to predict flows accurately or even to behave reasonably, there still seems to be life left in the model. This paper has shown that several modifications are possible that improve its behavior. First, the link-efficiency criterion was changed to select links closer to the destination. Second, it was shown that it was possible to include a unique parameter for each node. Third, the overallocation of trips to overlapping paths was reduced by weighting the arc likelihoods.

Others have suggested that path enumeration, simu-

Figure 11. Flows generated by Dial's algorithm on network with three cycles.

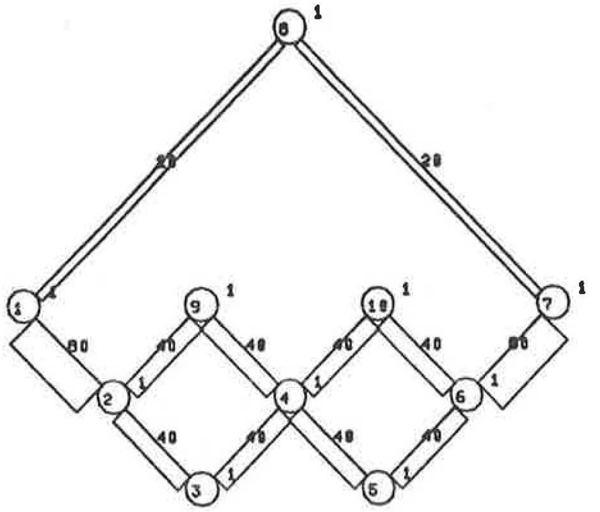


Figure 12. Flows generated by modified algorithm by using gamma weights.

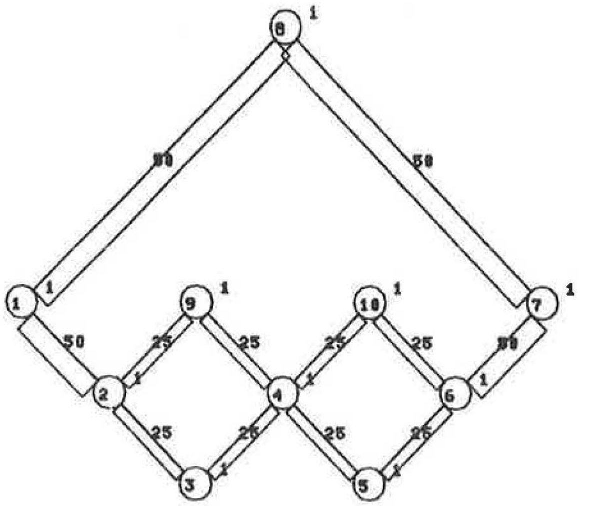


Figure 13. Single-parameter assignment, modified algorithm.

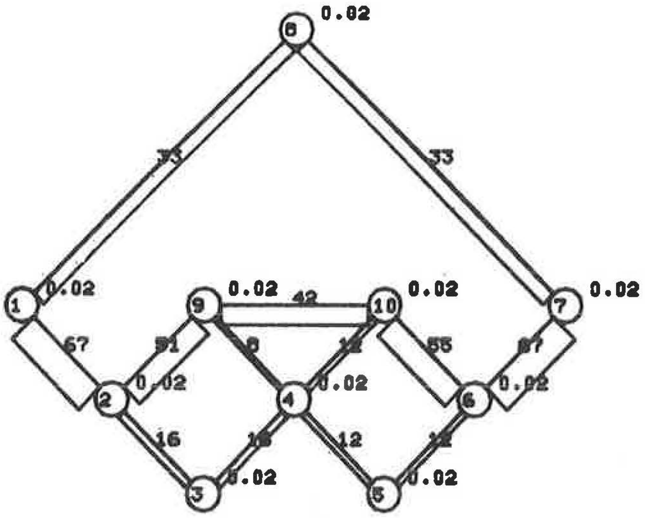
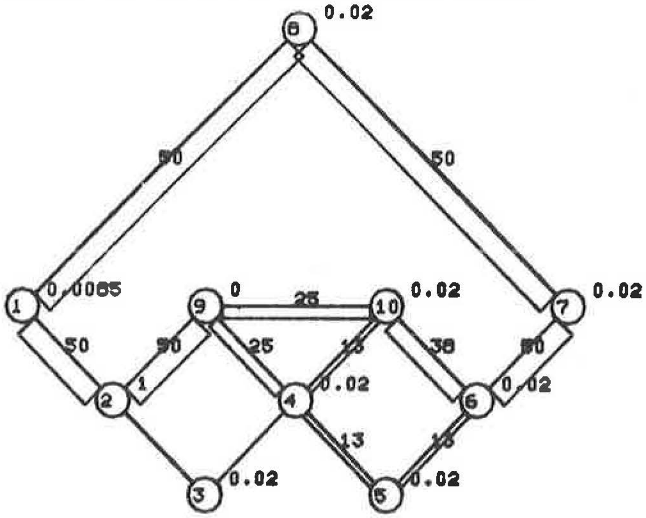


Figure 14. Flow balancing and concentration, modified algorithm.



lation, or interactive editing combined with a stochastic model may yield a more-fruitful approach to multiple-route traffic assignment. Daganzo and Sheffi (7) showed that Dial's logit model implied that route error terms are independent and identically distributed. Their approach involved developing the actual variance-covariance matrix of route travel times. Although this seems theoretically more appealing, it nevertheless requires calculation of the link-route incidence matrix for all feasible paths for each O-D pair. For most networks, this is now computationally infeasible. For example, a simple square grid network that has 10 nodes and 9 links on a side has 48 620 possible paths that connect the extreme nodes on the diagonal. It is questionable whether a simulation approach can be substituted for path enumeration and can still achieve reasonable results for large networks. In spite of all its shortcomings, the modified version of Dial's model still holds promise of becoming an efficient, accurate, and easily implemented multiple-route traffic assignment algorithm.

By introducing multiple parameters, a number of degrees of freedom have been incorporated into the model. With the multiple-parameter model, calibration to fit actual route itineraries for individual travelers may be possible. In addition, it is possible to provide a better, more-appropriate means of calibrating traffic assignment models, since one now has much better control over modeling the decision-making process. Previously, in order to make traffic assignment models agree more closely with reality, one had to adjust link travel times and costs, essentially distorting the network to account for behavior. When the multiple-parameter model is used, one can use measured link times and there is no need to tinker with the network itself.

In the model developed in this work, the graphic capabilities of the computer system have been used to provide visual output. Since the subject of the investigation is a network, which is a representation of a spatially determined transport system, the use of graphic output enables the analyst to link numerical results to the spatial structure. Looking at a picture of the network annotated with numerical results provides an easier means of interpreting data than scanning through tables of numbers. In fact, the graphic display capability proved valuable in the development of the model itself, since one could directly assess several aspects of the model's performance by inspecting the graphic output. Numerous errors were detected in this way that might not otherwise have been caught.

In developing the algorithms, it was found that the ability to program interactively gave greater flexibility in modifying the models and encouraged more experimentation. It was not necessary to submit a program and let the machine grind away. Execution could be interrupted at any time, intermediate output could be examined, and data or programs could even be modified before operation continued. Many ideas could be tested in the time needed for a single run of a conventional model. In addition, the power of APL enabled dealing with networks as matrices and vectors without having to program tedious intermediate steps and do-loops. The algebra of the problem could be dealt with almost directly rather than having to translate the mathematics into a computer language.

ACKNOWLEDGMENT

All model development work was accomplished on an IBM 370/158 computer at the Interactive Computer Graphics Laboratory (ICGL), Princeton University. All programming was done interactively by using VSAPL computer language under a virtual machine configuration that had the conversational monitor system. All figures were drawn by using assembler language drawing functions developed by the ICGL staff. Curve plots used functions of the LINPLOT package developed by Tom Hahn and Yehonathan Hazony of the ICGL. I wish to thank Granville Paules, Linda Moore, and Robert B. Dial of the Office of Planning Methods and Support, Urban Mass Transportation Administration, for their assistance in obtaining support for this research. This research was performed under a research, development, and demonstration grant from the Urban Mass Transportation Administration, U.S. Department of Transportation.

REFERENCES

1. B. Marchand. Pedestrian Traffic Planning and the Perception of the Urban Environment: A French Example. Environment and Planning, Vol. 6, 1974.
2. A. L. Kornhauser and S. E. Still. Analysis of the Flow of Freight Traffic on the U.S. Railroad System for the Year 1974: Final Report. Transportation Program, Princeton Univ., Princeton, NJ, 1977.
3. T. Lee. Perceived Distance as a Function of Direction in the City. Environment and Behavior, Vol. 2, 1970, pp. 40-51.
4. M. Wachs. Relationships Between Drivers' Attitudes Toward Alternative Routes and Driver and Route Characteristics. HRB, Highway Research Record 197, 1967, pp. 70-84.
5. J. E. Burrell. Multipath Route Assignment and Its Application to Capacity Restraint. Proc., Fourth International Symposium on the Theory of Traffic Flow (Karlsruhe, Germany, 1968).
6. R. B. Dial. Probabilistic Assignment: A Multipath Traffic Assignment Model Which Obviates Path Enumeration. Univ. of Washington, Seattle, Ph.D. dissertation, 1970
7. C. F. Daganzo and Y. Sheffi. On Stochastic Models of Traffic Assignment. Transportation Science, Vol. 11, 1977, pp. 253-274.
8. M. Schneider. Probability Maximization in Networks. Proc., International Conference on Transportation Research (Bruges, Belgium, 1973).
9. R. L. Tobin. An Extension of Dial's Algorithm Utilizing a Model of Tripmakers' Perceptions. Transportation Research, Vol. 11, 1977, pp. 337-342.
10. M. Florian and B. Fox. On the Probabilistic Origin of Dial's Multipath Traffic Assignment Model. Transportation Research, Vol. 10, 1976, pp. 339-341.
11. J. M. Lutin. Transit Station Simulation Model (USS) Testing and Evaluation. Princeton Univ., Princeton, NJ, Transportation Rept. 76-TR-9, 1976.
12. J. M. Lutin. A Perceptual Model of Path Choice in Urban Transportation Networks. Princeton Univ., Princeton, NJ, Ph.D. dissertation, 1977.
13. A. V. Aho, J. E. Hopcroft, and J. D. Ullman. The Design and Analysis of Computer Algorithms. Addison-Wesley, Reading, MA, 1974.
14. R. P. Polivka and S. Pakin. APL: The Language and Its Usage. Prentice-Hall, Englewood Cliffs, NJ, 1975.

Use of Computer Graphics to Analyze Navigational Performance and Jet-Route Separation

DALE LIVINGSTON, EDWARD J. KOBIALKA, AND NEIL W. POLHEMUS

Formulation of jet-route separation standards to achieve adequate levels of safety requires careful analysis of aircraft navigational performance. In the processing and analysis of extensive data obtained from the continental U.S. air traffic control (ATC) system, computer graphics techniques were used to develop a clean data base, to provide descriptive summaries of achieved navigational performance, and to portray the relationship between collision risk and track-system geometry. This paper describes these applications and their importance in ATC systems analysis.

As part of a general study of aircraft separation standards, the Federal Aviation Administration (FAA) has collected a large amount of radar data on aircraft navigational performance in the high-altitude continental U.S. jet-route system. These data were collected to serve as the empirical base for assessing the adequacy of current route-spacing criteria. In all, more than 10 000 flight records were collected along route segments in the Cleveland, Memphis, and Albuquerque air traffic control (ATC) centers.

The task of taking these data and using them to make recommendations regarding jet-route separation requires a careful step-by-step process of data verification, analysis, and interpretation. To assist in this process, FAA's National Aviation Facilities Experimental Center and the Department of Civil Engineering at Princeton University have relied heavily on the use of computer graphics. As described in this paper, the ability to portray interactively the recorded ATC environment and analytical results is essential to ensure a meaningful and useful analysis of methodologies for establishing separation standards.

Initially, the raw radar returns needed to be compiled into a data base representative of actual navigational performance along selected jet routes in the study area. To screen out aircraft that were not actually attempting to navigate along the routes--for example, those that had been given a direct clear, a holding pattern, or some other maneuver by ATC--visual inspection of the recorded tracks on an interactive graphics terminal proved to be invaluable. Summaries of daily traffic on selected routes were similarly generated, as will be described later.

It is of considerable interest to determine the likely explanation for the times when ATC did intervene in an aircraft's performance and the resulting consequences. To reconstruct the environment that surrounds a particular aircraft at a specified time, an interactive graphics technique was developed.

Several graphical procedures designed to provide descriptive summaries of observed navigational performance and also to refer aggregate performance to the current very-high-frequency omnidirectional radio-ranging (VOR) system-use accuracy standard are described. The important use of graphical display in guiding the estimation of lateral overlap probabilities is illustrated and, finally, a computer-generated movie that was developed to describe the modeling procedure for the case of intersecting jet routes is described. In each of these stages of the analysis, graphical display is crucial to ensure effective use of the mathematical results by those who eventually make the decisions.

QUALIFYING THE DATA

It is characteristic of any data collection that most of the data is useful and some is not. Of that which is useful, a portion requires special attention to render it acceptable. This data collection was no exception and certain graphical tools were designed and implemented in order to identify those data that required additional attention or deletion from the sample set. These tools proved to be valuable for indicating features of the data that may have been overlooked by summary processes or other attempts to estimate the character of the data through quantification. As used here, "qualification" of the data is a term that relates to the graphical methods used to examine data and determine suitability for the sample.

The raw radar returns used for estimation of aircraft position were collected from three separate radar sources. Given the error inherent in reporting radar position, the collation of these data required smoothing for accurate depiction of aircraft position. The smoothing technique chosen was a method of robust nonlinear smoothing tailored expressly for radar-derived data (1-4). Figure 1 shows both the raw and the smoothed radar data as plotted on a flat-bed plotter. The aircraft at the ends of the data trace indicate the direction of flight with respect to the route. The triangles represent the raw data, and the smoothed data points are connected by a solid line. Data traces such as these were produced in digital form for each of the 10 000 aircraft in the sample, but only those that reflected special circumstances were reviewed.

Figure 1 also provides an example of the robust performance of the smoother. This aircraft exhibits a standard delay tactic used by ATC; a linear smoother would not have been able to handle this situation but the smoothing technique used tracks the data well. Such a maneuver was not suitable for the description of aircraft navigational performance as considered in this study and would have been deleted from the sample after detection. Although it may have been possible to detect the tactic through other analytical techniques, it was easier to confirm the maneuver through this display.

Once a preliminary data base had been amassed, certain investigative techniques were used in order to delete from the sample those aircraft whose courses had been directly influenced by ATC. To provide the data used in this investigation, observers had been placed at each controller position during the data collection to provide a log of the aircraft sampled and voice recordings of the conversations between pilots and ATC. When initial screening of an aircraft's navigational performance indicated the possibility of direct intervention by ATC, another graphical aid was used. Figure 2 is a view of one such interrogation of the data base, copied directly from a cathode-ray tube (CRT) display. The information displayed on the screen shows the jet route and protected airspace, the position data trace, and the aircraft identification. The data block in the upper right corner identifies the aircraft and supplies additional information about the trace (from top to bottom): the aircraft identifier, jet route flown, VOR route segment displayed, beginning and ending trace times (in hours and minutes), aircraft beacon code, altitude (in hundreds of feet), and Julian date (the number of days elapsed in the year). The intended route of the aircraft and an area surrounding it known as the protected airspace relate the aircraft's track to the local control area.

After the data are presented on the screen, an analyst may use the terminal's crosshairs to identify portions of the smoothed aircraft trace for closer examination. By moving the crosshairs to a particular place on the data trace and pressing the return key, a small data block appears that contains the time at that position, the perpendicular distance to course (in nautical miles), and the distance to the nearest navigational aid (in nautical miles). Five such blocks are displayed in Figure 2. The ability to determine maximum excursion from course and the associated time, the point of initiation of a turn, or some other characteristic of the data trace and to compare the observed behavior with commands given by ATC proved to be a valuable supplement in the decision to retain an aircraft in the sample. Since the reduction of a voice-recorded tape (the next step in qualifying the data) meant many person hours of dedicated effort, the possibility of removing a candidate from the sample at this point saved a large amount of human resource for alternate analytical work.

As a further step in the analysis of the data, multiple aircraft traces were plotted on the same route to present a sample of aggregate navigation performance. Figure 3 is a presentation of multiple traces plotted on a single route in which the angular nature of signal propagation as well as other properties of navigational performance may be noted. This illustration presents a day's worth of aircraft traces plotted on the plan of the route and the protected airspace. The display presents a qualitative view of navigational performance as well as an idea of the proportion of aircraft that escape the protected airspace; this quantity is crucial to the evaluation of collision risk. General trends

Figure 1. Nonlinear smoothing of aircraft position data.

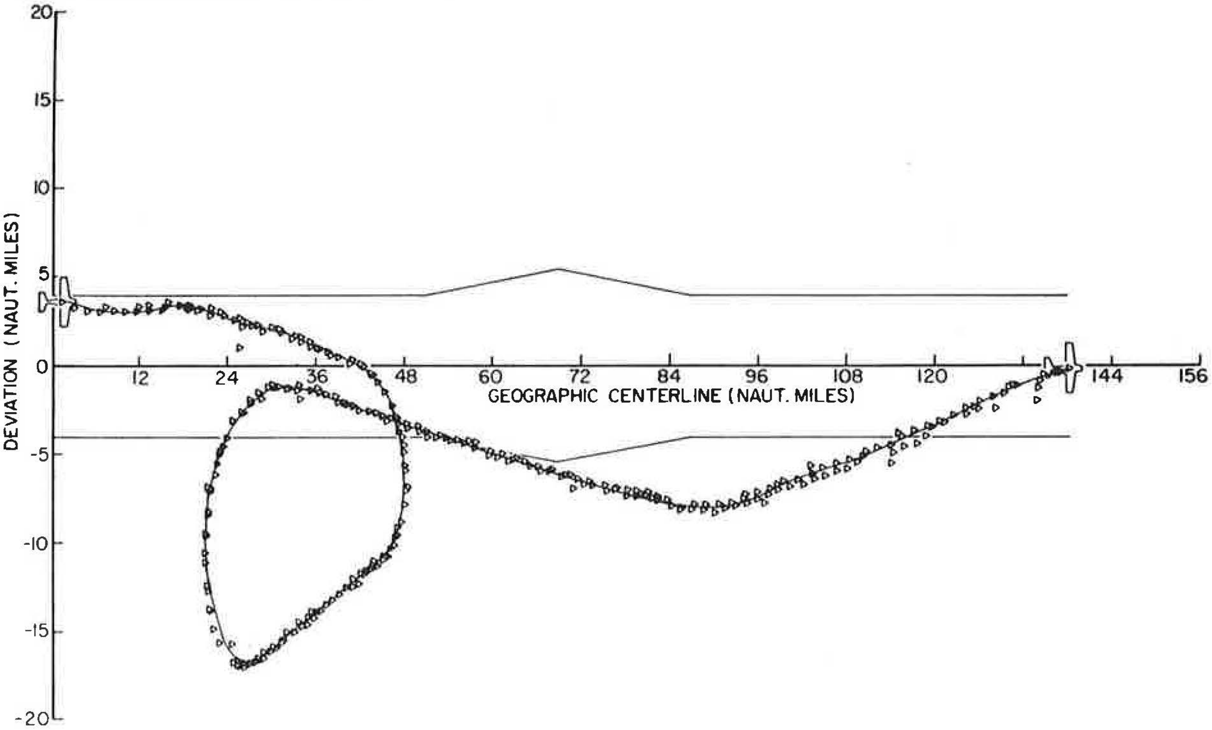
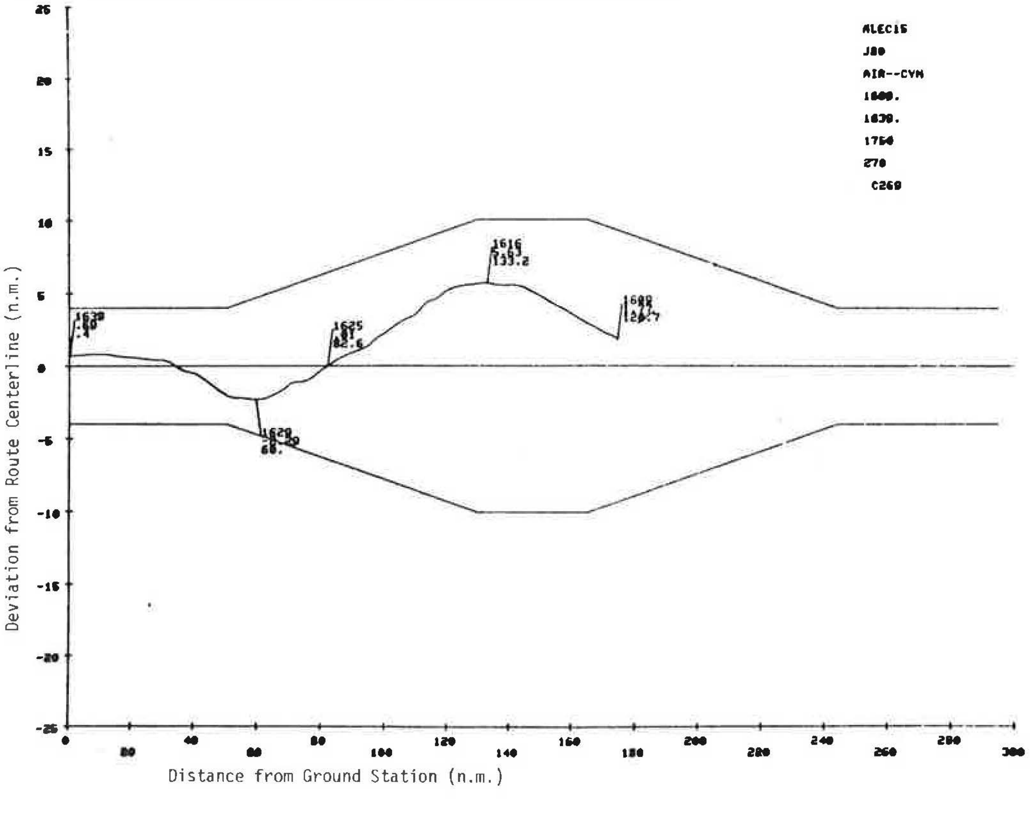


Figure 2. Trace of single aircraft that shows data blocks generated by interrogation.



and patterns associated with navigation performance in the aggregate become apparent after lengthy examination of multiple-trace plots for each route under consideration, which has allowed the study of daily trends in navigation performance and on occasion has uncovered meteorological conditions that affect other flights in the same time period.

Figure 4 shows an additional graphical tool

Figure 3. Multiple aircraft traces generated from fast-time data replay.

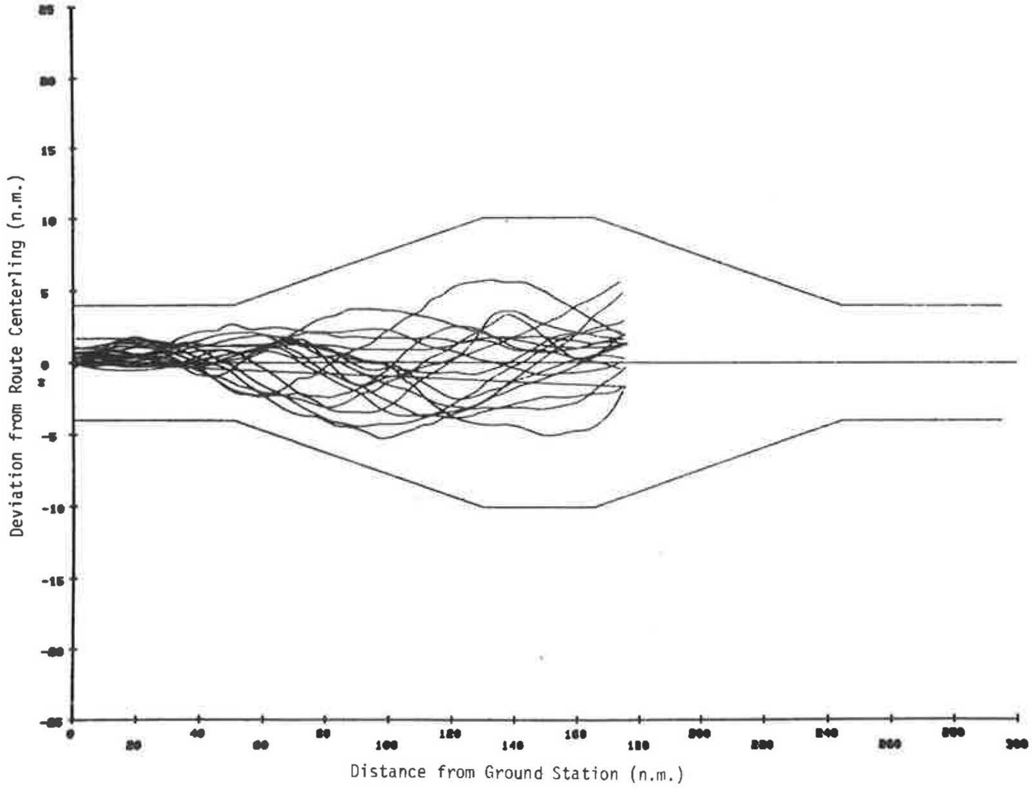
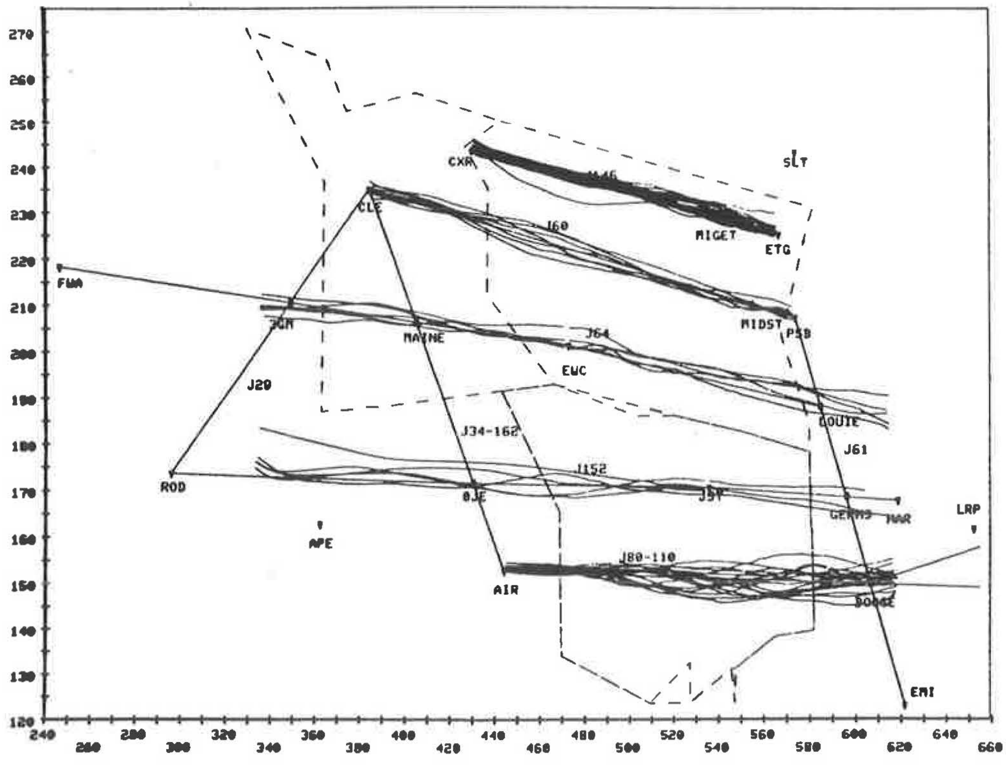


Figure 4. Representation of one day's sample plotted on the local control area.



designed to help relate aggregate performance on one route to that noted on adjacent routes in the same time period. The geometry of the local control area and the applicable routes is projected onto the screen and the analyst recalls an interval of time during which an event of interest occurred. The interaircraft dynamics of the situation may be observed to aid in determining those factors that may have influenced the aircraft's position with respect to the route. Again, a fast-time examination of the actual scenario that uses any number of aircraft may be portrayed. By allowing the field to be enlarged beyond the area of interest, certain anticipatory aspects of pilotage could be observed and analyzed. This analysis provided background for the explanation of observed behavior.

EXAMINING THE ATC ENVIRONMENT

Proximate trace plotter (PROXPLOT) is a FORTRAN program designed to examine the air traffic environment around a specified aircraft. Although previous data processing had proceeded with the goal of characterizing the navigation performance of selected aircraft with respect to their intended flight paths, a need to examine the traffic environment about an aircraft of interest was perceived. Satisfying this need would allow a more-thorough investigation of the possible impact of ATC or navigation errors on the safety of the system.

A graphical display of the ATC environment seemed to provide the most efficient way of transmitting this information. The initial concept called for software that would read a radar data record (RDR) tape and present radar position data on a CRT display; essentially it would play back the plan video display that had appeared at the controller's position. This concept was rejected as infeasible; hardware limitations prevented the presentation of such large amounts of data in a reasonable amount of time. These limitations lay not only in the graphic display, but also in processing time; an RDR tape contains raw radar data in an encoded format, and much processing must be done before the data can be effectively displayed.

These restrictions are circumvented by the design of PROXPLOT, which is in reality two separate programs. First, a file of data is collected by the program named PROXBOX, which operates under a batch monitor. The analyst specifies to PROXBOX a specific aircraft (the candidate) and a set of times that may be of interest. PROXBOX reads through the radar data and follows this particular aircraft as it flies through radar coverage. A proximity disk of specified radius and thickness is carried with the candidate aircraft; all other aircraft that enter this disk are noted along with the time and distance of their closest approach to the candidate. A hard-copy summary of these data is made available for analysis. Second, PROXBOX also acts as the data collector for the graphical display of PROXPLOT. At any set of times specified by the programmer, PROXBOX writes to disk file the current position of the candidate aircraft and any aircraft within its proximity disk as well as a short history (about 10 radar hits or approximately 1 min) of the preceding positions of all aircraft. It is this file that provides the data to be displayed by PROXPLOT.

Figure 5 is a copy of the display produced by PROXPLOT obtained from a xerographic copier connected to the CRT. The flight trace of the candidate aircraft is approximately in the center of the display area and is marked by a line of asterisks. The symbol ϕ next to the data point farthest on the right identifies the leading side of the trace (the most-recent data) and indexes the trace to the data block in the upper right-hand corner. This data block indicates the radar beacon code that identifies the aircraft, the last recorded flight level, the time of the last data point for that trace, and the x-coordinates and y-coordinates of the last point in that trace. The display area is automatically scaled so that the candidate trace is plotted in the center of the area and the diameter of its proximity disk fills about 75 percent of the screen. ATC environment data for that center are stored on line, automatically accessed, and plotted. These data include the location of the navigational aids (navaids, shown by triangles), jet routes defined by these navaids (solid lines), and the sector boundaries of the controlling center (dashed lines). The dashed-line circle (plotted at the discretion of the analyst) shows the extent of the proximity disk with reference to the last point of the candidate trace.

Once the basic information has been displayed, the analyst has the opportunity to obtain more information from the plot by using the display-screen crosshairs and single-letter commands. Examples of these are also shown in Figure 5. The operator may obtain the plot of the x-, y-, z-, and t-coordinates of any one radar data point by moving the crosshairs to the point and giving the appropriate command letter. The selected point is circled, and the operator moves the crosshairs a short distance away and enters a label. The program labels the point as shown and writes the appropriate information on the right-hand side of the screen. In Figure 5, the most-recent point of trace 1, labeled A, has x-coordinates and y-coordinates 337.38 and 245.61, respectively; records flight level 350; and was taken at 20:03:06.7.

Other functions are available that work in much the same manner. The operator locates the point of interest by using the crosshairs and chooses a label to index the event. Current program configuration includes the capability to locate in terms of x and y any point on the screen or to measure the distance between any two points, as is shown for the distance between the candidate aircraft and the navaid SSO (labeled B). The operator also has the option of replotting the current display without the entered labels, of listing all radar data points displayed on the screen, and of skipping over any of the plots recorded on the input file.

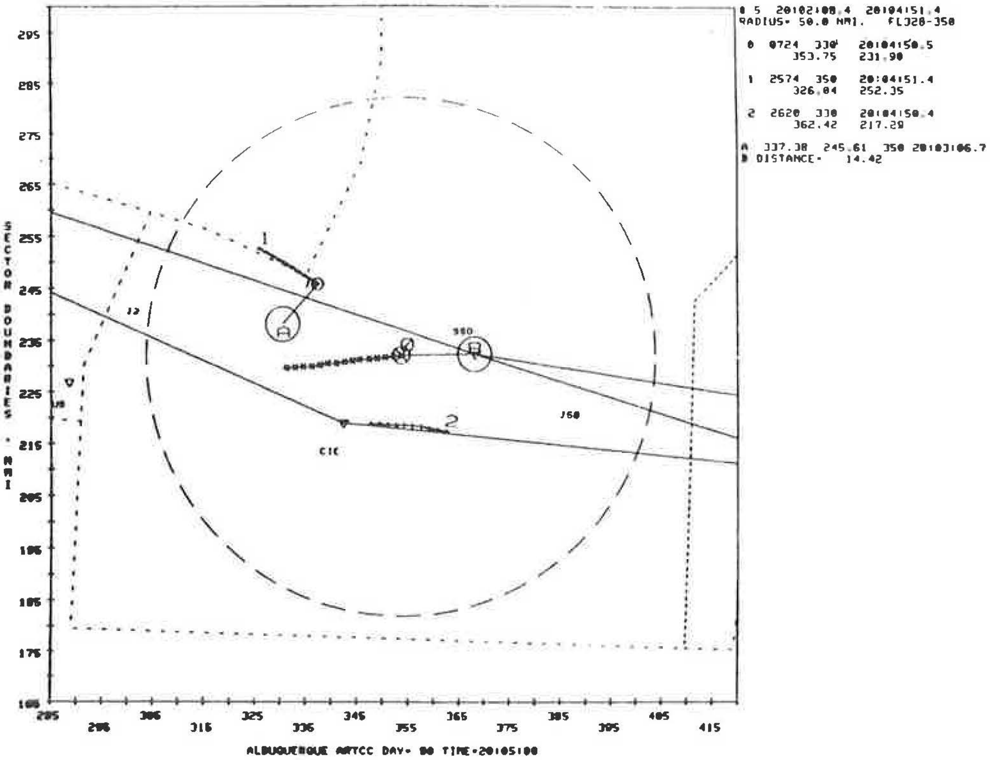
The PROXPLOT program has proved useful in processing navigation data in the aggregate. Each aircraft that exceeded a certain distance from route centerline during its flight was screened to ascertain the origin of the error; specifically, a deviation that occurred under ATC cognizance or direction was not included in the navigation performance data base. Normal methods for determining ATC intervention include scrutinizing data collection logs and recording pilot-controller communications. Since these sources are not infallible, use of the above display in determining the traffic environment around the perceived error can often be helpful in determining the controller's intent and the source of the deviation.

PRESENTING SUMMARIES OF OBSERVED LATERAL DEVIATIONS

Having limited the sample to those flights determined to be navigating without direct ATC intervention, it was then of interest to characterize the statistical properties of the data. One particular question centers around the effect of distance from ground station on the magnitude of lateral deviation from route centerline. In particular, the current VOR system-use accuracy standard requires 95 percent of all traffic to be within a protected airspace centered about the route; the width of that airspace depends on the distance from the nearest ground station.

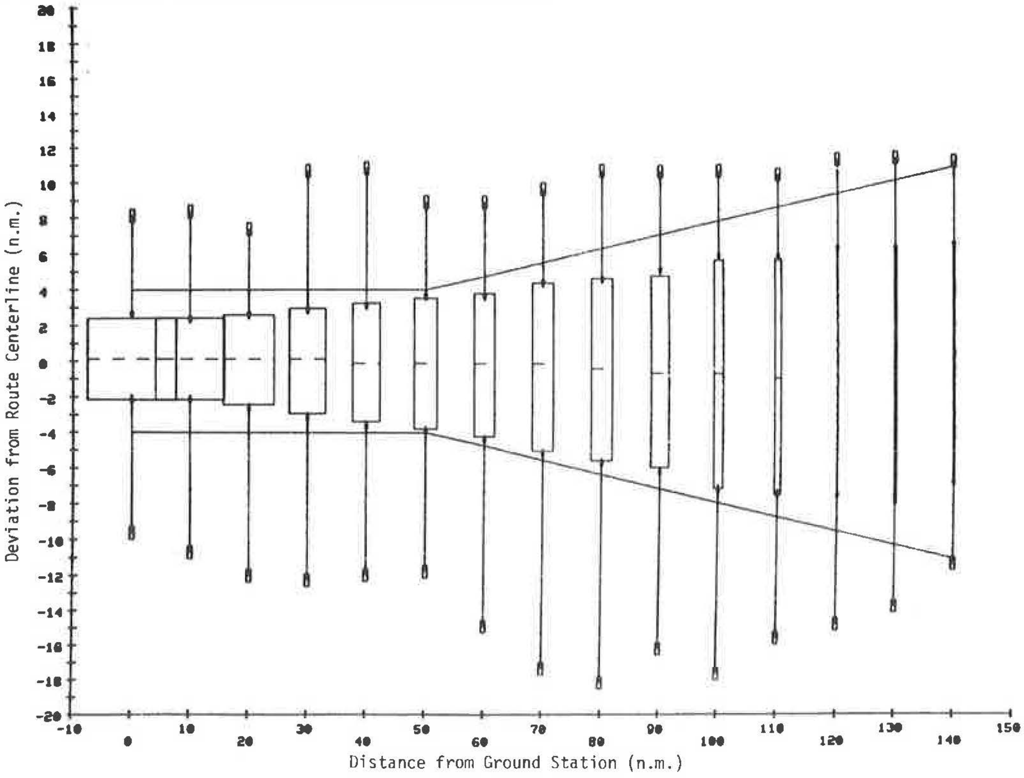
To illustrate the effect of distance on the distribution of lateral deviations, a graphical tool

Figure 5. Interactive examination of proximate aircraft.



```
0 5  20102108.4  20104151.4
RADIUS- 50.0 NMI.   FL320-350

 0  0724  330   20104150.5
          353.75  231.90

 1  2574  350   20:04151.4
          326.04  252.35

 2  2620  330   20104150.4
          362.42  217.29

A  337.38  245.61  350 20103106.7
B  DISTANCE-   14.42
```

Figure 6. Summary characteristics of lateral-deviation distributions.



was developed similar to the box-and-whisker plot suggested by Tukey (5). This particular plot (Figure 6) shows both the protected airspace (funnel-shaped lines) and various summary statistics; it has the dual purpose of exhibiting the changing distributional form and determining whether the 95 percent containment was being achieved. At each milepost, the central rectangle is centered at the mean of the observed data, whereas its height is ±2 SD. If the distributional form is Gaussian, as postulated by the standard, then the box ought to enclose about 95 percent of the observations and be contained within the protected airspace.

The central dashed line in each box represents the median of the data and if not near the center could indicate excessive skewness. The kurtosis calculated from the observed data at each milepost is used to scale the width of the box. The narrowing of the boxes as the distance from the VOR increases indicates a flattening of the distribution; this is also illustrated in Figure 7, which shows a more-conventional illustration by using relative-frequency histograms.

Of particular interest during this study was the information in the tails of the distribution, which is a crucial input in later calculations of overlap probabilities. The tails of the distribution are defined by the outermost 2.5 percent of the recorded deviations on either side of the route centerline, indicated in Figure 7 by the "whiskers." To present this aspect of the data more clearly, Figure 6 was modified (not illustrated here) by removing the large rectangles that represent summary distribution statistics so that the relationship of the tails of the distribution to the protected airspace might be more easily perceived. This is an example of a type of graphics modification called "erasure," which allows selected information to be better assimilated by the viewer.

CHARACTERIZING THE DISTRIBUTION OF LATERAL DEVIATIONS

Many factors lead to the occurrence or avoidance of midair collisions. Some factors, such as escape maneuvers when two pilots judge that a collision is imminent, are difficult to model since the possible actions are so numerous. In contrast, for aircraft assigned to laterally adjacent routes, a factor directly related to the risk of collision that can be quantified is the magnitude of lateral deviations of the aircraft from their respective route centerlines. Determination of adequate lateral spacing between routes should be directly related to the resulting frequency with which aircraft will come close to each other laterally.

To state the problem formally, consider the task of determining an adequate spacing S_y between two parallel routes. If lateral overlap is defined to be the condition in which the centroids of a pair of aircraft are within a wingspan of each other in the direction perpendicular to the routes, then the probability of lateral overlap P_y is clearly a function of S_y. Determining the form of this functional relationship would allow for better specification of adequate separation between the routes.

In the absence of controller intervention, $P_y(S_y)$ could be determined by convolving the distributions of lateral deviations from route centerline for two aircraft on adjacent routes. If we let $f_1(Y)$ and $f_2(Y)$ denote the probability density functions for lateral deviations on the two routes, then

$$P_y(S_y) = 2\lambda_y \int_{\infty}^{\infty} f_1(Y) f_2(Y - S_y) dy \qquad (1)$$

where λ_y is the average wingspan of the aircraft.

Given the data base described in previous sections of this paper, the above convolution could be estimated either by observing lateral deviations directly or by fitting an appropriate density function initially and then performing the required integration. In either approach, the resulting estimate will be dominated by the tails of the lateral-deviation distributions (which represent very large deviations). Even given the extensive data base available, observed deviations in the extreme tails are very sparse. Consequently, if the data are used directly through a numerical convolution, values of $P_y(S_y)$ for large S_y must be obtained by smoothing and extrapolation (6). If density functions are fitted, the resulting estimates will depend greatly on how well these functions fit the tails of the observed data. In either case, graphical techniques have been found to be crucial in determining which mathematical models adequately describe the observed behavior.

Based on the current data collection, it was decided to take the second of the above approaches [reasons for this choice were documented by Polhemus (7)] and fit an appropriate density function to the lateral deviations from route centerline. To determine initially whether a simple distribution such as the normal or double exponential might be appropriate, graphical displays were generated that showed how the proportion of deviations greater than or equal to Y varied as a function of Y (equivalent to 1 minus the empirical cumulative distribution). Figure 8 shows such a plot for deviations at 50 nautical miles from the nearest VOR ground station; the equivalent curves for a normal distribution and a double-exponential distribution with standard deviation are equal to that implied by the current VOR system-use accuracy standard. With logarithmic vertical scaling, it is immediately apparent that the tail of the actual distribution is considerably heavier than implied by the normal. The double-exponential distribution, which does have heavier tails, is not supported by the data either. This simple graphical technique suggests immediately that a more-complicated distributional form is required; this conclusion was later substantiated by more-sophisticated goodness-of-fit tests.

Two natural extensions of the normal and double-exponential distributions, obtained by introducing additional parameters in the density function, are the exponential power distribution (8) and the generalized t-distribution (9). Both these distributions require numerical procedures in order to estimate the parameters. As is the case with all numerical procedures, it is essential that the analyst monitor the results carefully to ensure that the algorithm used behaves correctly.

Here graphical procedures were employed to assess the adequacy of the distributions as fit by the algorithm. Initially, the numerical algorithms were used to refit a normal distribution as a means of verifying the computer programs. The numerical fit was then compared graphically to the observed data by plotting the actual and predicted number of deviations in increments of 0.25 nautical mile. Figure 9a shows such a plot that uses logarithmic vertical scaling, from which it is evident that the data are more peaked in the center than a normal distribution would imply and that the observed tail is also much heavier than the normal fit. In contrast, Figure 9b shows the results of a fitted generalized t-distribution. Both the core and the tail of the fitted distribution correspond reasonably well to the observed data. Again, the graphical display is essential in allowing the analyst to judge the acceptability of the fit.

Once an acceptable fit has been obtained for the observed distribution of lateral deviations from route centerline, the resulting probability of lateral overlap can be obtained as a function of route separation S_y by convolving the distributions as indicated earlier. To summarize the results, one final graph is relevant--Figure 10, which shows how the probability of lateral overlap varies with route separation and assumed distribution. To convert the calculated probability into a measure of risk obviously requires many additional calculations and assumptions, which are outside the scope of this paper. Nevertheless, Figure 10 shows much about the relative relationship between route spacing and the potential for a collision, which, when combined with other factors, could be meaningfully employed to establish acceptable route-spacing criteria.

ANIMATED PRESENTATION OF THEORETICAL RISK MODEL FOR INTERSECTING ROUTES

Although the applications of collision-risk methodology to date have been solely for parallel routes, many route systems have nonparallel and intersecting routes. In explaining the derivation of overlap probabilities for such situations to nontechnical audiences, graphical procedures are again essential. Since the probability of overlap for a pair of aircraft that traverse an intersection

Figure 7. Relative-frequency histograms by milepost.

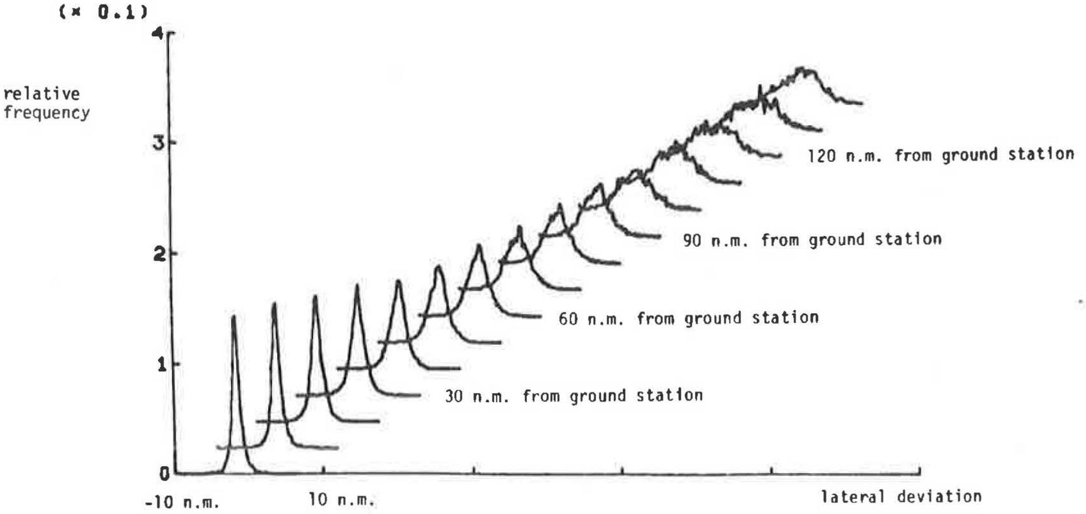


Figure 8. Comparison of observed and theoretical cumulative distributions (Cleveland).

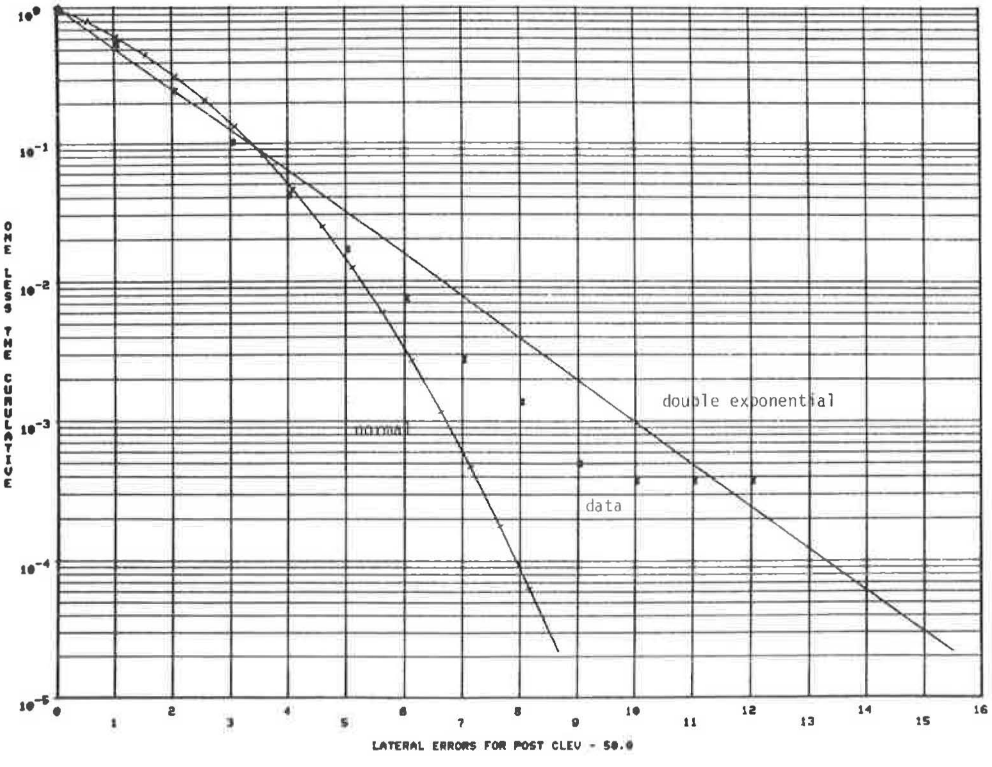


Figure 9. Normal and generalized fitted distribution at the 50th nautical milepost in Cleveland.

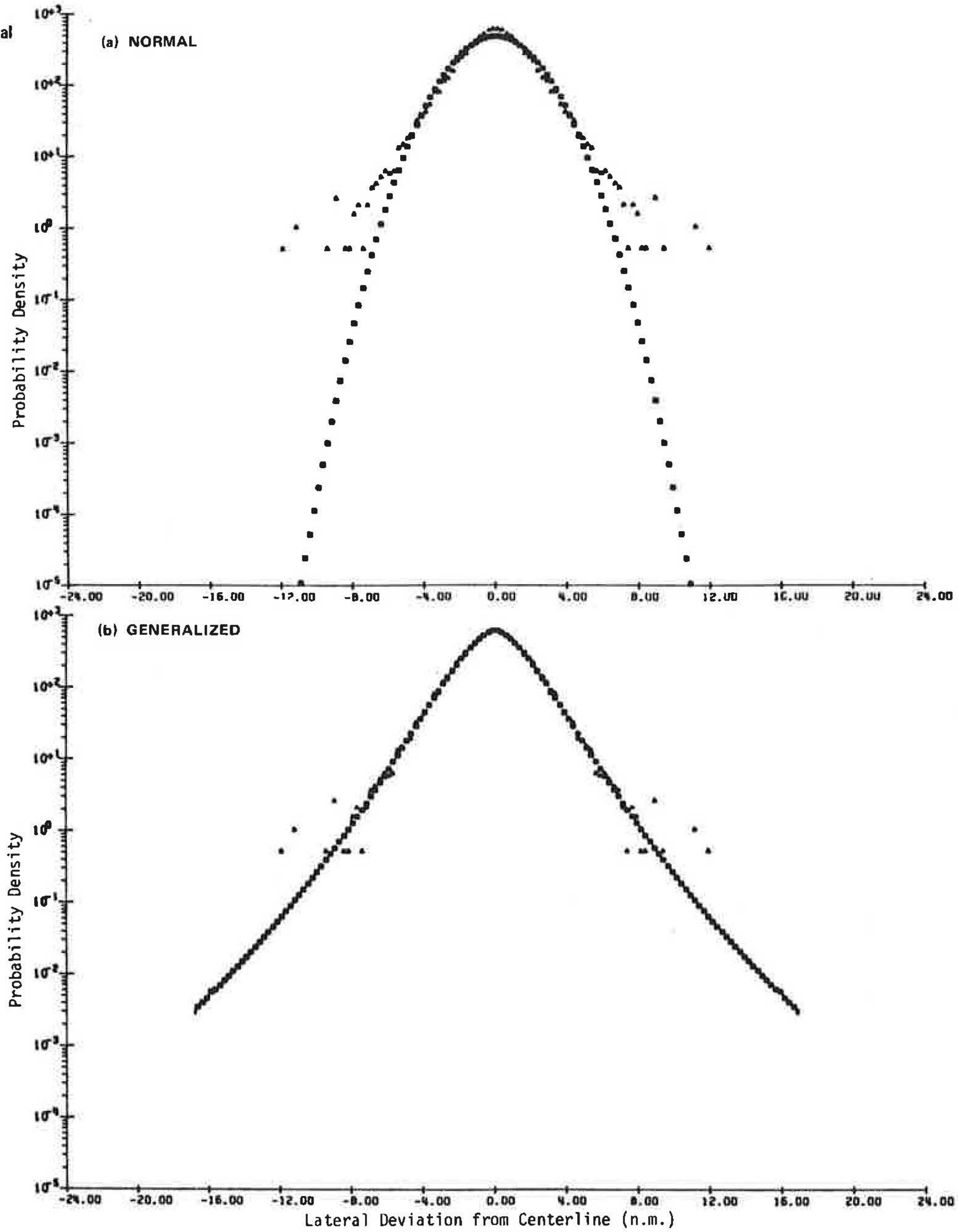


Figure 10. Probability of lateral overlap as function of route separation.

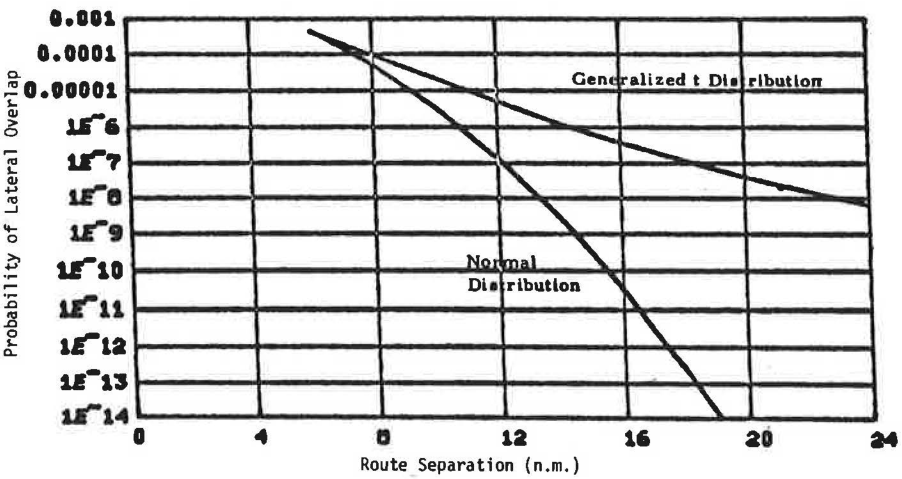
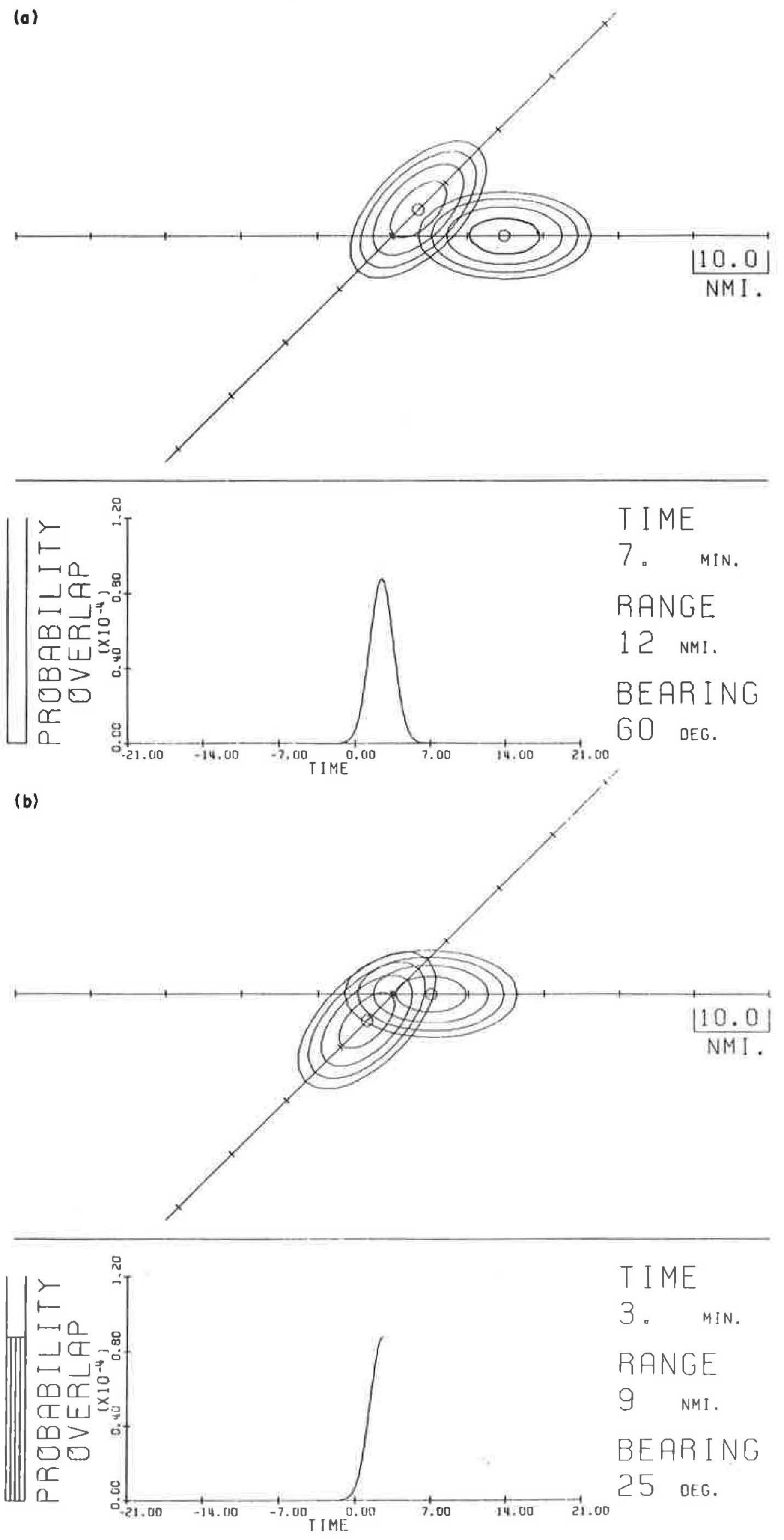


Figure 11. Selected frames from computer-generated movie that illustrate overlap probability for intersecting routes.



changes over time, a useful portrayal is through a computer-generated movie. Such a movie has been produced; two frames are shown in Figure 11. The example illustrated is a 45° intersection. The contours of the bivariate probability density function of along-track and cross-track errors are shown as ellipses about the planned position of each aircraft.

As the movie evolves, the ellipses move at a constant velocity through the intersection. As the first aircraft nears the center of the intersection the probability of overlap increases, as illustrated by the rising thermometer on the left. At the same time, a plot of probability versus time is created. The movie has been found to be particularly effective in explaining the general concepts of the model to a nontechnical audience, which is crucial if the analytical results are to be accepted.

CONCLUSIONS

Computer graphics has played a central role in the analysis of data and development of methodology for determining adequate jet-route separation standards. Development of a clean data base, summary description of navigational performance, estimation of lateral-overlap probability, and dissemination of the model have all been greatly enhanced by the availability of computer graphics. The use of graphical tools throughout the study resulted in substantial savings of analytical resources and an understanding of the problem that would otherwise have been very difficult to achieve.

REFERENCES

1. A. E. Beaton, Jr., and J. W. Tukey. The Fitting of Power Series, Meaning Polynomials, Illustrated on Band Spectroscopic Data. Presented at 133rd Annual Meeting of the American Statistical Association, New York, 1973.
2. P. F. Velleman. Robust Non-Linear Data Smoothers: Theory, Definitions, and Applications. Princeton Univ., Princeton, NJ, Ph.D. dissertation, June 1975.
3. J. D. Van Ness. Non-Linear Smoothing of Long-Tailed Aircraft Radar Data. Department of Civil Engineering, Princeton Univ., Princeton, NJ, M.E. thesis, 1975.
4. D. A. Livingston and E. H. Stevens. Robust, Non-Linear Smoothing of Noisy Time Series: VOR-Defined Air Route Separation Study. National Aviation Facilities Experimental Center, Federal Aviation Administration, U.S. Department of Transportation, Tech. Letter Rept. NA-78-72-L.R., 1978.
5. J. W. Tukey. Exploratory Data Analysis. Addison-Wesley, Reading, MA, 1977.
6. Methodology for the Derivation of Separation Minima Applied to the Spacing Between Parallel Tracks on ATS Route Structures. International Civil Aviation Organization, Montreal, PQ, Canada, ICAO Circular 120-AN/89/2, 1976.
7. N. W. Polhemus. Estimation of Lateral Overlap Probabilities for VOR-Defined Jet Routes. Department of Civil Engineering, Princeton Univ., Princeton, NJ, Tech. Rept., Jan. 1980.
8. G. E. P. Box and G. C. Tiao. Bayesian Inference in Statistics. Addison-Wesley, Reading, MA, 1973.
9. D. Livingston and N. W. Polhemus. Statistical Modeling of Aircraft Collision Risk for High-Altitude Jet Routes. Department of Civil Engineering, Princeton Univ., Princeton, NJ, Tech. Rept. 79-TR-19, Aug. 1979.

Applications of Interactive Graphics in Michigan's Statewide Transportation Modeling System

TERRY GOTTS AND RICHARD ESCH

Modern transportation agencies face increasing responsibilities and decreasing revenues. The only viable solution is increased productivity and efficiency. NETEDIT, an operational interactive method of updating, displaying, and interrogating networks, has allowed Michigan's Department of Transportation to increase its analysis capabilities without sacrificing production time. The utility of such a process is easily seen by contrasting manual coding and batch updating with the interactive update process. NETEDIT gives the user instant pictorial feedback, which enables correction of errors on the spot. This allows the average elapsed time for generation of alternatives to drop from two weeks to 4 h. In addition, the ability to vary the way a link is drawn—solid, dotted, dashed, crosshatched, even varying bandwidths—based on its attributes lends versatility. The addition of a digitizer tablet has allowed creation of 2000-node networks in two weeks. A tree-plotting subroutine eliminates much of the elapsed time in network calibration. Most importantly, NETEDIT, together with Michigan's Statewide Transportation Modeling System, has been used in more than 350 actual planning applications over the past two and one-half years. As a result, NETEDIT has become a valuable transportation planning technique in Michigan.

Governments face a challenge in meeting modern political realities. On the one hand, legislation and public expectations demand thorough planning and implementation of an expanding range of governmental responsibilities. On the other hand, the economic realities of the foreseeable future dictate a governmental belt-tightening; personnel and budgets will be maintained or cut back rather than increased.

Michigan's Department of Transportation (MDOT) is meeting just such a challenge in its Bureau of Transportation Planning. Although budget and personnel are not rapidly expanding, the demands on the planning process from federal legislation, MDOT's action plan, and concerned citizens have grown tremendously. MDOT must now consider many alternatives and modes and must examine an ever-widening range of impacts--travel, social, economic, and environmental--for every major transportation project it undertakes. Moreover, the people who live in the region of the proposed project and who will therefore be most affected by the final solution must be involved in each step. Based on the evaluation of the first set of alternatives, new alternatives may

Figure 1. CRT, digitizer tablet, and hard copy.

be proposed and the whole cycle repeated until a consensus is reached.

With conventional methods of network updating, interrogation, and display, the emphasis on public involvement becomes self-defeating. As more and more enthusiastic citizens start asking questions, the planning machinery begins to get bogged down. Elapsed time between questions and answers keeps growing, and public enthusiasm declines. The result is interested but frustrated private citizens who feel they cannot get answers and well-meaning but frustrated transportation planners who would like to answer the questions but cannot get to them for two months.

The logical solution to the dilemma of increasing responsibility and decreasing resources is increased productivity. Michigan has a unique opportunity in this respect. Michigan's Statewide Transportation Modeling System (STMS) is an operational multimodal impact-analysis system that has been used in more than 350 applications over the past two and one-half years. This system has allowed MDOT to take full advantage of interactive graphics through the development of the NETEDIT computer program. NETEDIT is an interactive method of creating, displaying, and analyzing alternate transportation networks and of relating associated socioeconomic data to them. Through interactive graphics, NETEDIT has (a) decreased the workload, (b) decreased elapsed time, and (c) increased accuracy, thereby allowing MDOT to plan, design, and construct a transportation system more efficiently (1-4).

HARDWARE REQUIREMENTS

There are only two basic hardware requirements for implementing an interactive system of network display and updating. First, the user needs a cathode-ray-tube (CRT) computer terminal (Figure 1), which is similar in appearance to a television screen and is capable of drawing pictures as well as printing alphanumeric characters. Second, there must be access to a computer that supports a high-level programming language and allows a computer program to be linked with routines that actually do the drawing on the screen. These routines can usually be purchased from the manufacturer of the CRT at a nominal cost.

It is important that the screen of the CRT be large enough to display a natural unit of data--say, a county--without appearing cluttered. If it is difficult for the operator to figure out what he or she is looking at, the objectives of increased efficiency and accuracy will not be realized. The most popular screen sizes range from a rectangle about 10 in measured diagonally to one that is about 25 in diagonally. The examples in this report were made on a Tektronix 4014-1, which has a 19-in screen.

Resolution is important if the program is to calculate the actual length of a link from the screen coordinates of its end points. For example, the 4014 has 4096 x 4096 addressable points. By comparison, the small-screen 4010 has only 1024 x 1024 addressable points, which somewhat increases the chance of approximation error. The user may decide approximately how many square miles should be displayed on the screen and then compute possible errors for each model of CRT being considered.

A third consideration is computer speed. Most high-level machines are quite fast; however, if a time-sharing system is overloaded, one should think very carefully about investing the time and money required to put up an interactive graphics system. Moreover, even a fast line will not help in such a situation: What good does it do to draw at 9600 baud if there is a 30-s pause between bursts of drawing?

Finally, the reliability of the time-sharing system to be used must be looked at objectively. If the system is prone to frequent periods of down time, this will produce many false starts and restarts in a network update. In that case, it would probably be less frustrating and more productive to stay with manual coding techniques.

NETEDIT is now running (with excellent results) on a large-scale Burroughs B-7700 computer with three processors and a 4.7-megabyte memory. An earlier version was run on a Control Data 6400-series machine. It is written in FORTRAN-IV and uses Tektronix Plot-10 graphics subroutines. The program is currently limited to networks of not more than 13 000 links and 16 000 nodes.

STMS NETWORK CONCEPTS

Collection, storage, and retrieval of all data employed within STMS are tied to a 547-zone system. Of this total, 508 are in-state zones. The zonal concept is of extreme importance in that it provides a dynamic link between information retrieval and actual model procedures. The conversion of raw data from storage within the information files into accurate travel, social, economic, and environmental indicators has been effectively accomplished as a result of gearing the entire system to the zonal format.

The transportation network model is the means by which the transportation planner describes to the computer in its own language the transportation system under study. The network is defined in the network model as a set of links and nodes. Nodes are numbered points located by X-coordinates and Y-coordinates that reference a statewide coordinate grid. A link is defined by its connection of the nodes of two networks. Figure 2 shows a conceptual drawing of a portion of the highway network within several zones of a zone system. The illustration indicates that there are two basic types of system links--regular links and pseudolinks, which are known as centroid links. A regular link is used to describe a section of the transportation facility (e.g., highway or rail segment), whereas a centroid, in connecting itself to a node of the base network, allows the feeding of traffic to and from a zone and off and onto the system.

The transportation planner must differentiate between types of links according to certain physical and travel characteristics. Each discrete piece of link-specific descriptive data, or link attribute, is stored on a computer file in what is known as a volume field. An understanding of the volume-field concept is critical to one's comprehension of the

Figure 2. Network description.

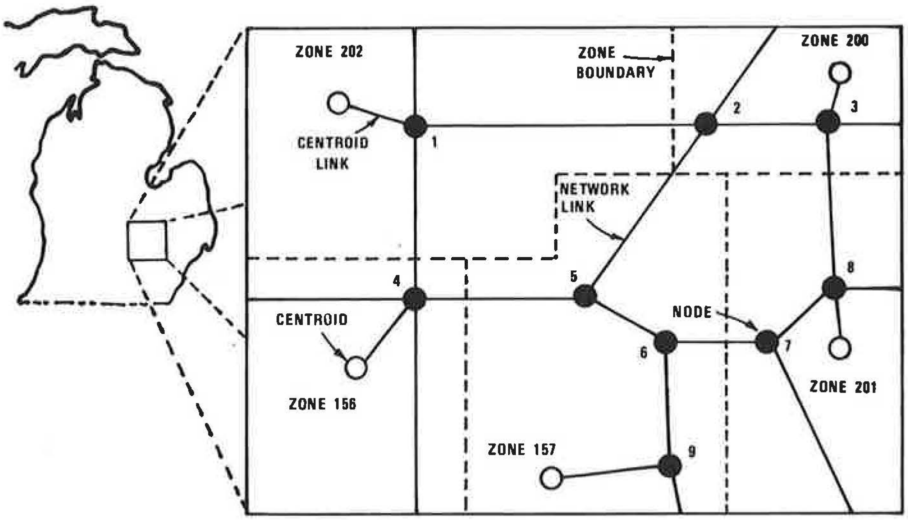
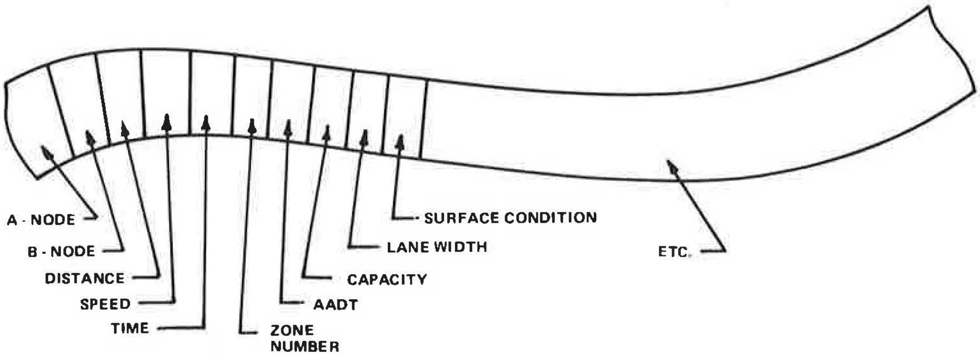


Figure 3. Link description.



modeling system, for it is consistently employed throughout the process. Figure 3 illustrates how a highway link's attributes might appear on a segment of magnetic tape if it were visible to the human eye. In creating the network model, data records are taken from punched cards and recorded sequentially on the network data file. First, a link's A- and B-nodes are recorded to distinguish it from other links within the network. This initial portion of a link's data file contains, in volume fields, other information pertinent to its description, e.g., type (existing or newly created) and jurisdiction (who funded the construction and maintains the facility).

NETWORK UPDATING: MANUAL VERSUS INTERACTIVE

Once the network file is created through standard coding procedures and specialized computer programs, it may be modified to simulate alternative transportation proposals. Many such changes are performed when comparisons of travel impacts are desired. The transportation network model provides a primary input not only for the travel forecasting model, but for all models developed within the statewide system. The steps involved in creating and modifying networks are exacting and time consuming. These are the steps that the interactive network updating system was developed to replace:

Step 1: Cards must be punched to update a base network into the alternative network. For the analyst, this means coding (i.e., writing on forms from which cards will be punched) all the links and nodes to be added, deleted, or changed in the updated alternative network and the link attributes for the new or modified links.

Step 2: After the cards have been keypunched, they are input into a series of computer programs that (a) alter the base network to produce an updated alternative network, (b) change or add volume-field (link-attribute) information, and (c) create a computer-readable version of the network that can be plotted on paper.

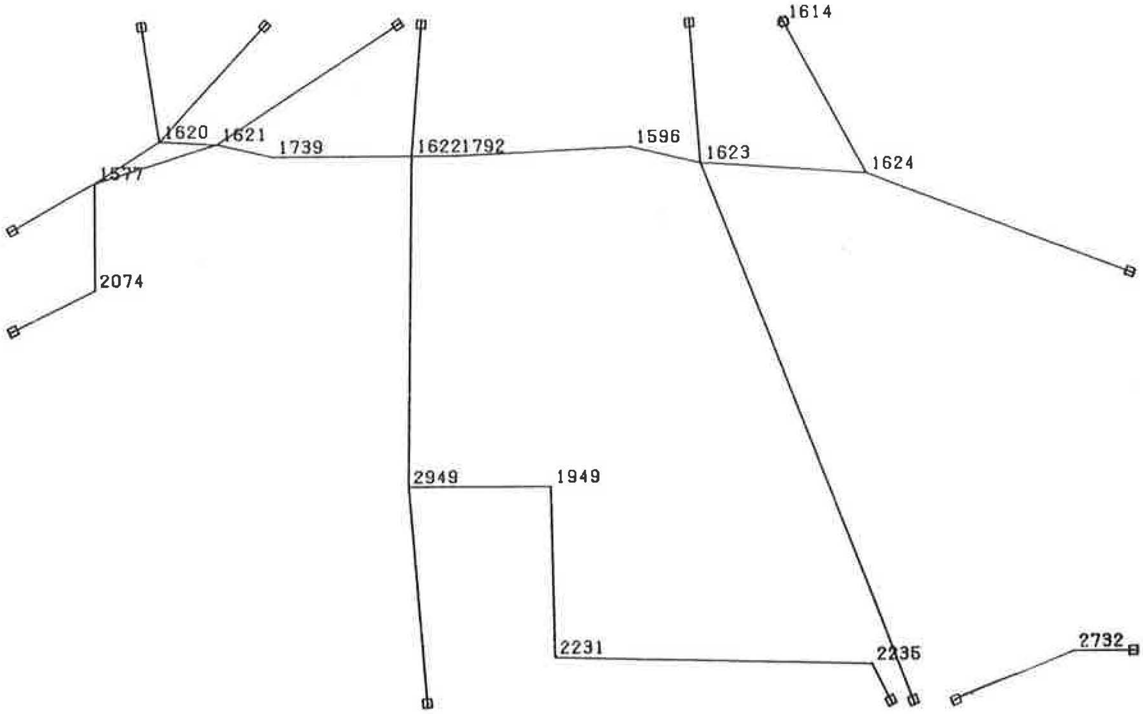
Step 3: After the network has been plotted, which produces a plot such as that in Figure 4, the analyst pores over it to make sure that no errors have been made. Typically, the analyst must check (a) the location of modified nodes, (b) the correct addition or deletion of all links, and (c) the correct volume-field updating. If errors are discovered, further updating is begun by going back to step 1.

Often the entire process is repeated two or three times. Even if the procedure is gone through only once, it usually takes up to two weeks for coding, computer runs, plotting, and plot checking.

In comparison, the statewide interactive network updating system is simpler and less laborious and provides for immediate verification. From the user's perspective, the system can be a simple one-input one-output system such as that shown in Figure 5. The user merely specifies the necessary information to access the base network at the beginning of the program; all other file manipulation occurs internally. The user need only pay attention to the location of roads when the desired network alternative is being built.

NETEDIT currently manipulates networks stored in

Figure 4. Plot made by using Cal Comp network plotter.



Burroughs' TP-System packed form. However, the program is modular enough and its concepts are general enough so that modifying the packing and unpacking subroutines should be all that is necessary to allow it to read and write networks in other forms, such as DCO/TRANPLAN or the Urban Transportation Planning System (UTPS). After the updated network has been written to a disk or tape file, it is used like any network created by batch processes. Thus, NETEDIT uses no special interfacing mechanisms other than its own packing and unpacking routines. Internally, the program creates and manipulates several background files (Figure 6):

1. Link attributes--capacity, base-year counts, and up to 45 other pieces of link-specific data--are stored on a randomly accessed disk file to save core space.
2. At any point between unpacking and packing, the current intermediate state of the network may be written to a disk file by giving the command SAVE. That stored state may be restored at any time by the command RCVR (recover), which provides a check-point and restart capability.
3. As the user executes network modification commands, they are encoded and written to a "tank" on a disk. After any system failure short of a disk crash, the user need only unpack the network that was being worked on when the system went down and execute the command TANK. Then the network will be changed as if every command had been reentered.
4. A record of changes made by every user to every network is kept in a log file on a disk pack. The file is periodically printed out and rewound. Besides providing informational backup, the log is also handy for answering questions about why the program does not work correctly.

SYSTEM DEMONSTRATION

There are five primary application categories for which NETEDIT is typically used on a daily basis: (a) network updating, (b) network display and presentation, (c) network generation, (d) tree (path) plotting, and (e) socioeconomic data display. By considering each category in turn, the reader may see the extent and versatility of the NETEDIT command set. Commands may usually be given in any order, so that the user may base the next action on the results of the last. This characteristic also makes NETEDIT a useful tool in dealing with private citizens' inquiries, in which the answer to one question often prompts another. Figures 8-19 were created by a hard-copy unit similar to the one shown in Figure 1. More than 9000 such copies have been used in day-to-day planning over the past two and one-half years in Michigan.

Updating

Consider now the portion of the highway net around Grand Rapids shown in Figure 4. Suppose MDOT is asked to evaluate the impacts of putting in a bypass around Grand Rapids (Figure 7). The planner would begin by adding a link (ADDL) from I-196 to US-131 (nodes 2074-2949), as shown in Figure 8. In Figure 9, the change-parameter command CHGP is used to reconstruct county road 2949-1949 to a four-lane freeway. The command CJ, 3 changes the road to jurisdiction 3 and puts in all volume-field defaults for that jurisdiction; then the command 3, 12 changes parameter 3 (link type) to 12, and the command 5, 1 changes parameter 5 (freeway code) to 1. The command DISP redraws the picture.

In Figure 10, an interchange is inserted on 1623-1961 at node 2231 by using the split command SPLT. This is equivalent to adding node 2231, deleting link 1623-1961, and adding links 1623-2231 and 2231-1961; during interactive operation, 1623-1961 is Xed out, which shows the deletion. The two new links are given all the characteristics of the parent, and the sum of their distances is made equivalent to that of 1623-1961. Finally, links 1949-2231 and 2231-1624 are added to complete the bypass.

This small example points out an additional ad-

Figure 5. Network updating from the user's perspective.

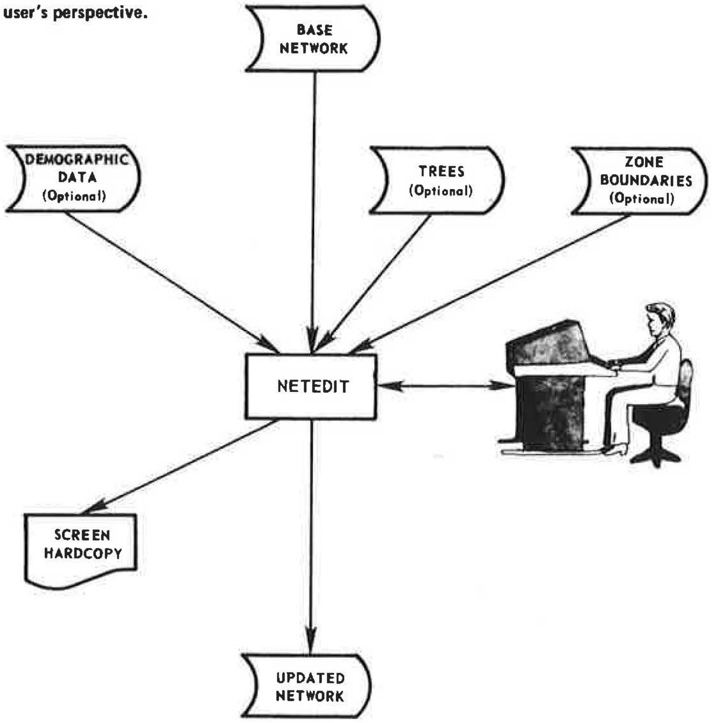
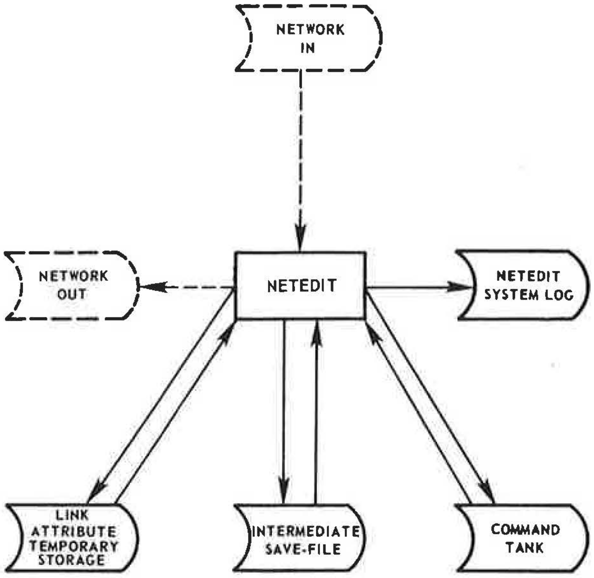


Figure 6. Program background files.



vantage to interactive updating--instant feedback. The user knows immediately whether the correct link has been deleted or whether a link to be added was placed correctly.

Display

Whether the planner is investigating a portion of a network in detail, finalizing dimensions and parameters in preparation for an ink plot, or merely producing quick 8.5x11-in plots for a meeting, NETEDIT cuts elapsed plot time from hours or days to seconds. For example, it is possible for the program to vary the manner in which a link is drawn based on link attributes. In Figure 11, the solid lines are state trunk lines, dashed lines are county roads, crosshatches are rail links, and links marked by triangles are rail-to-highway connectors. One may also choose, or filter, links into or out of the display based on attributes. Figure 12 is the result of filtering out all but rail links from Figure 11. A portion of the current display may be enlarged for further study: Figure 13 is the result of selectively windowing in on the right half of Figure 11. Zone boundaries can be superimposed by using dot-dash lines (Figure 14). Link attributes can be displayed either numerically (Figure 15) or by bandwidth (Figure 16--attribute 7 is 1975 average daily traffic).

Network Generation

A digitizer tablet attached to a graphics terminal (such as the one shown in Figure 1) can be used to great advantage both in the initial creation of a network and in the addition of an existing network. The user simply tapes a map to the tablet, enters two fixed points by using the bull's-eye cursor or stylus, and informs the program where these points lie in the master coordinate system. From then on, one-letter cursor commands control the flow of the subroutines as the user moves with the cursor from node to node, automatically creating links between nodes. In this manner, it has been possible for persons who have only moderate familiarity with computers in general--and NETEDIT in particular--to digitize detailed networks of approximately 2000 nodes and 2000-3000 links in two weeks. Because the tablet routines run as a subset of NETEDIT, one need not jump back and forth between programs to edit links entered from the tablet and, because the final product is a packed net, no further batch processing is necessary. This capability now allows Michigan to digitize an A-node, B-node network for all roads in the state. This network will be the basis for future impact analysis and needs studies.

Tree Plotting

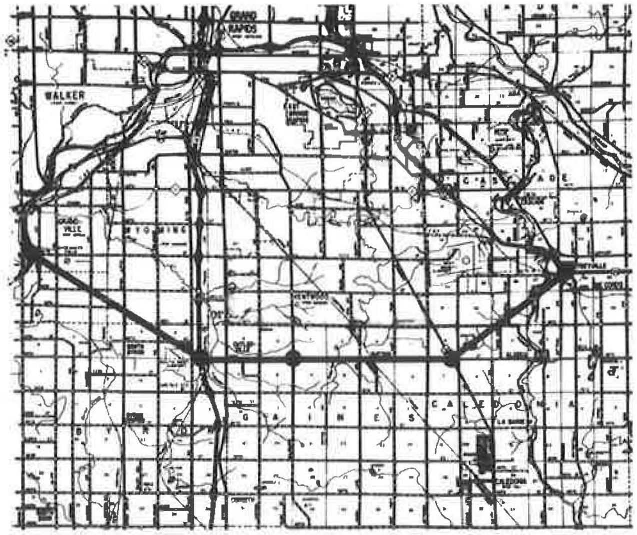
In network calibration as well as in alternative analysis, a large part of the job involves verifying that driving paths on the model reflect reality. Usually this is done by a batch-driven off-line plotter package. However, because most agencies have only one or two plotters to serve all users, a bottleneck often results.

The use of interactive graphics to plot trees offers several advantages, not the least of which is the reduction of turnaround time from days to minutes. In fact, interactive tree plotting has proved to eliminate as much as 90 percent of the time elapsed in highway network calibration.

Also, because the tree plotter is a subroutine of NETEDIT, all the options available in editing or displaying networks are accessible--windowing, changing line type based on link characteristics, and link-attribute annotation. One could even plot trees by bandwidth, the width of the line being determined by, say, accident rate.

Figure 17 shows a portion of a multimodal tree from Michigan's STMS. Note that in the interchange from zone 33 to zone 404, the tree proceeds on state trunk line to the rail-to-highway connector, continues onto the rail line, leaves the rail line, and finishes its path on trunk line.

Figure 7. Proposed Grand Rapids south bypass.



Socioeconomic Data Display

Any data that can be associated with a node on a network can be superimposed on a network plot. Figures 18 and 19 show the same data--projected 1980 zone population--plotted as vertical bars and as circles; the height of the bar or size of the circle indicates the magnitude of the zone's population relative to the other zones in the state. In these examples, zone boundaries have been drawn as dot-dash lines by NETEDIT. This subroutine becomes useful in attempting to relate a particular travel impact to the socioeconomic classes it affects.

These examples have demonstrated only a few of the NETEDIT commands. A more comprehensive list follows.

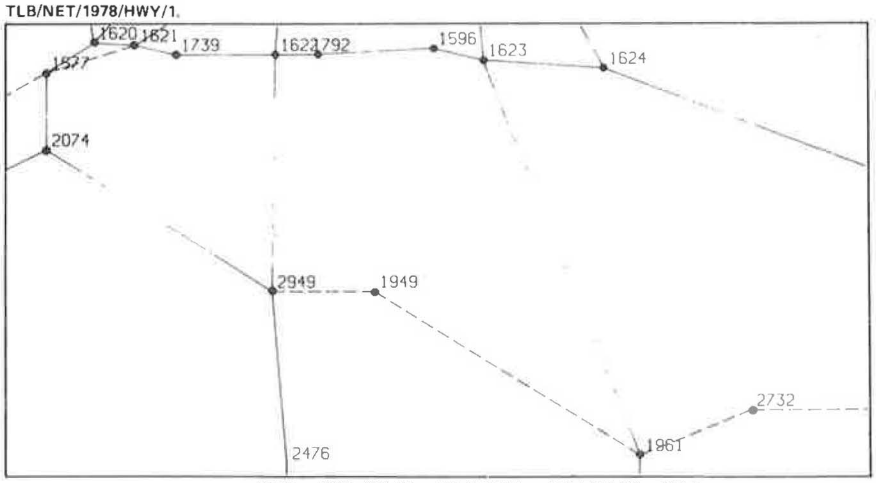
Network Updating

Command	Definition
ADDL	Add new link
ADDN	Add new node
CHGP	Change link parameters
DELL	Delete link
DELN	Delete node and any link that uses it
MOVN	Move node
SPLT	Split link in two; add new node

Network Display

Command	Definition
DISP	Display network in current coordinate window
ENLG	Enlarge portion of network
GOTO	Center window around specified nodes
OLVW	Return to previous virtual window
ZONE	Superimpose zone boundaries
NOZN	Turn off zone boundary overlay
NODE	Display node numbers
CENT	Display centroid numbers only
BLNK	Turn off node annotation
PAGE	Move window one screen page in one of eight directions
PARA	Display link attributes
FLTR	Set filter criteria for inclusion of link in display
CHGF	Change existing filter criteria
LINE	Change line type for link type or jurisdiction group
BSET	Set key attribute and band limits for bandwidth plotting
LGND	Display legend for bandwidth plot

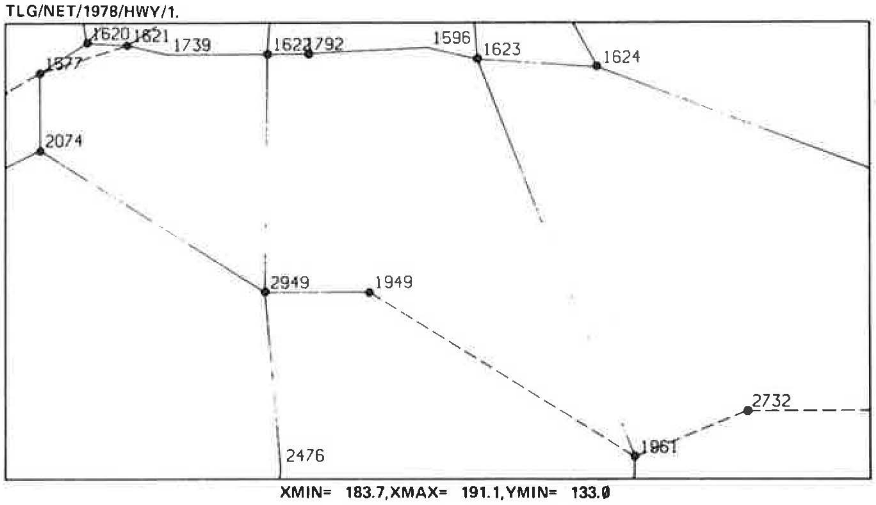
Figure 8. Network update: adding a link.



COMMAND:
ADDL, 2074, 2949

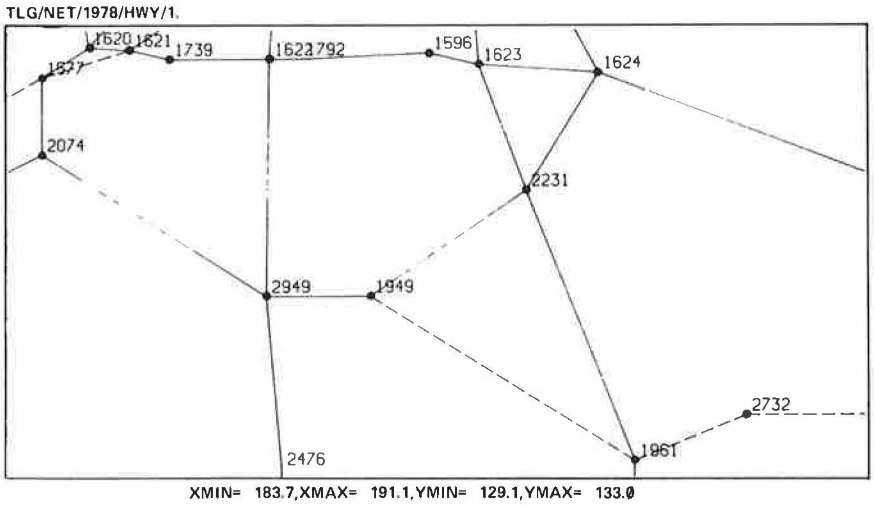
XMIN= 183.7,XMAX= 191.1,YMIN= 129.1,YMAX= 133.0

Figure 9. Network update: reconstructing county road to freeway standards.



```
COMMAND:
CHGP, 2949, 1949
CJ, 3
3, 12
5, 1
```

Figure 10. Network update: adding an interchange.



```
COMMAND:
SPLT, 1623, 1961
ADDL, 1949, 2231
ADDL, 2231, 1624
```

Figure 11. Line types based on link attributes.

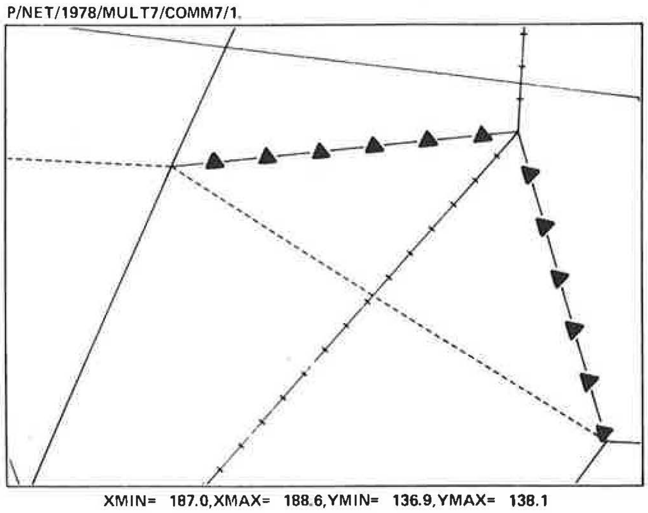


Figure 12. Filtering.

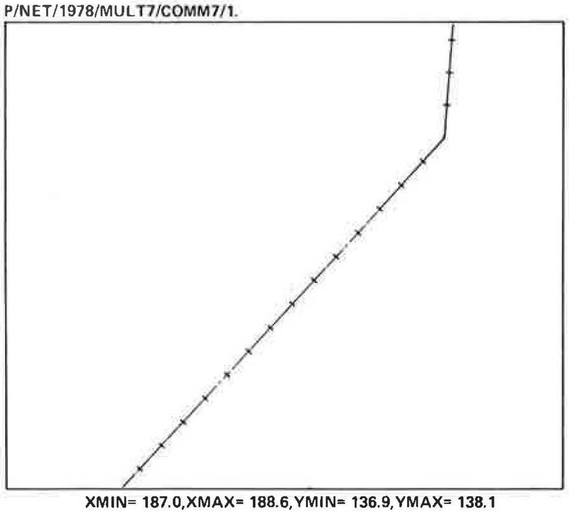


Figure 13. Windowing.

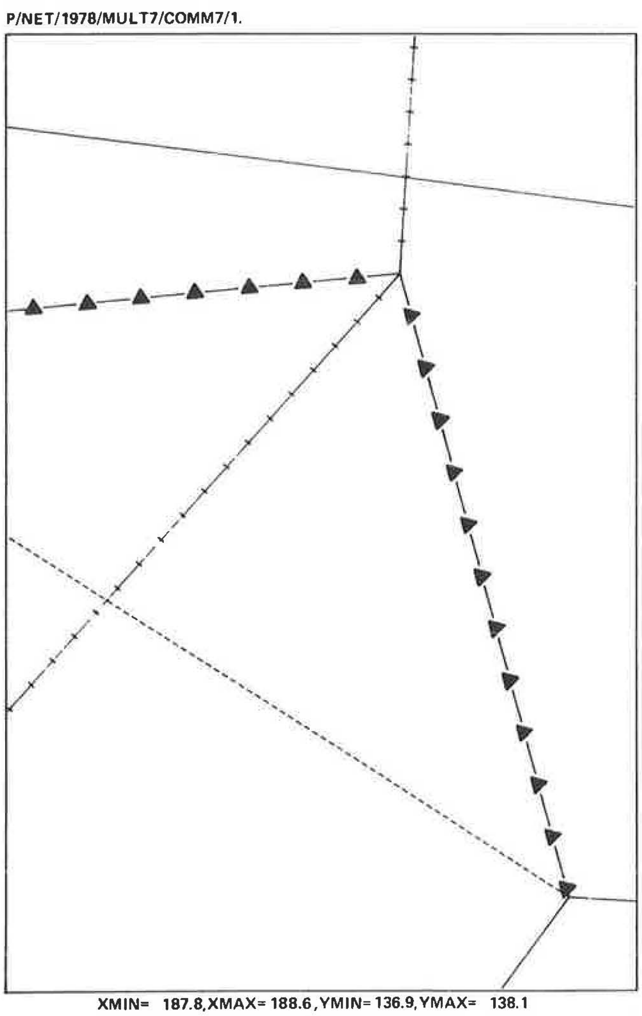


Figure 14. Zone boundary overlay.

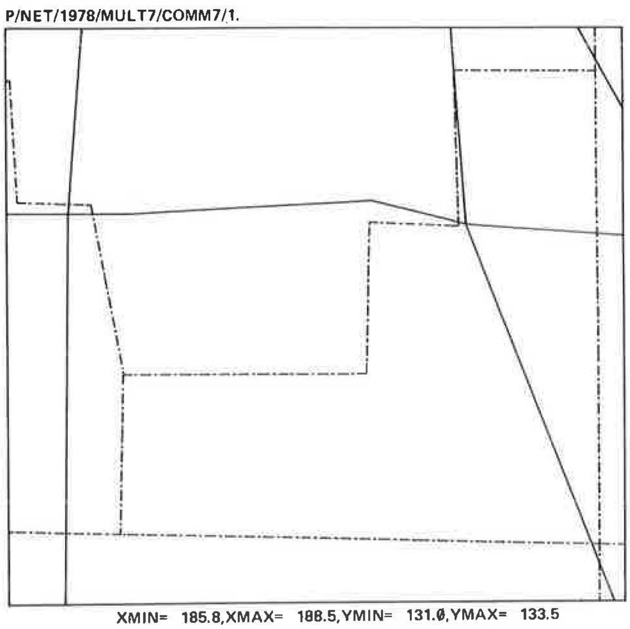


Figure 15. Numeric display of link attributes.

Figure 16. Bandwidth plotting.

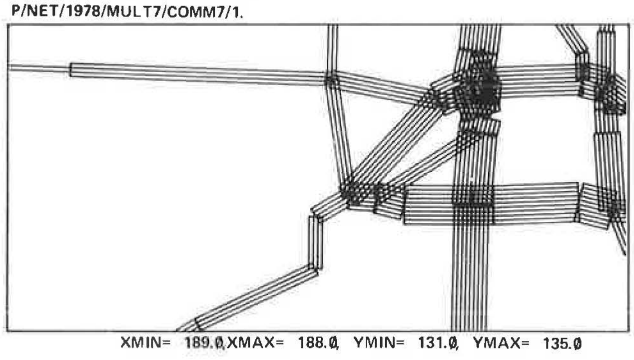


Figure 17. Multimodal tree plotting.

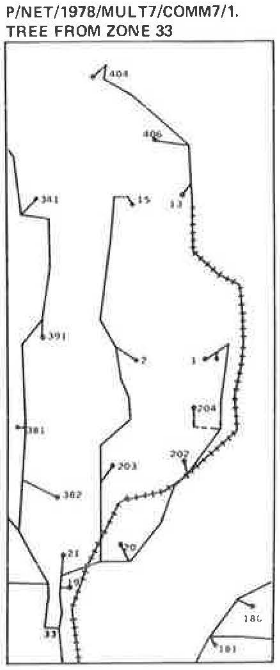


Figure 18. Socioeconomic data displayed as vertical bars.

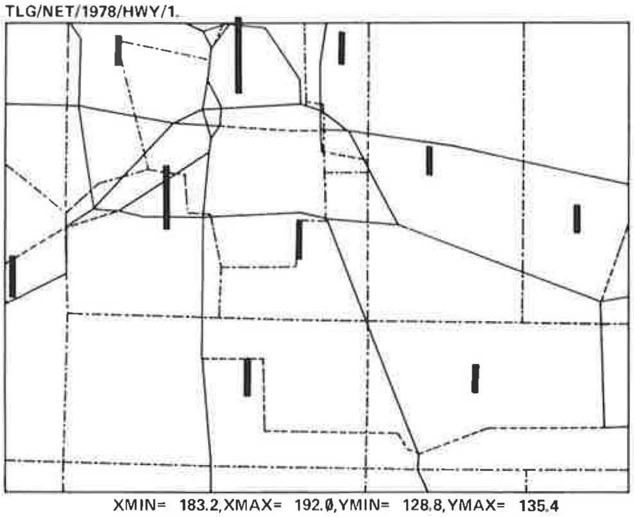
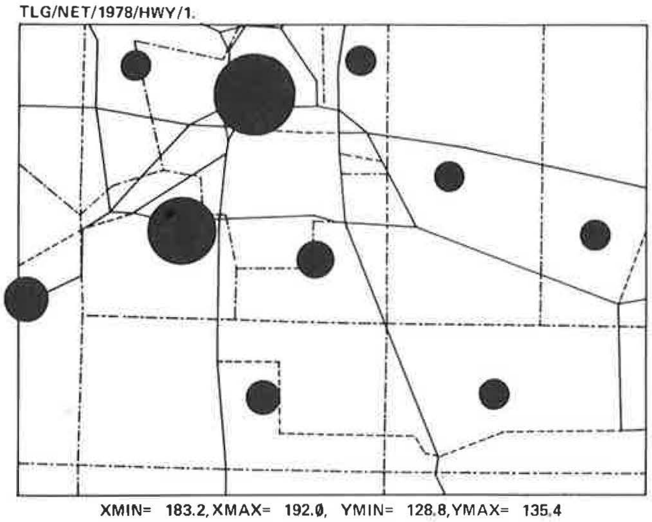


Figure 19. Socioeconomic data displayed as circles.



In addition to all regular commands, for network creation the following cursor commands may be entered while the graphics tablet is receiving points:

Network Display

Command	Definition
A	Automatically number new nodes
D	Delete links or nodes
E	Enlarge
F	Window full county or region
M	Move node
N	Return node sequence to existing node
O	Return to previous virtual window
P	Plot; redraw picture
R	Return to main program
S	Split link
#	Same as NODE
B	Same as BLNK

Tree Plotting

Command	Definition
TREE	Read tree file and plot portion of tree in current window
NEWT	Close old tree file, open new one, and plot tree
TIME	Annotate cumulative times on links of tree

Socioeconomic-Demographic Data Plotting

Command	Definition
DEMR	Read node data from disk
DSET	Establish level cutoffs for node data
DEMB	Plot demographic data as vertical bars
DEMC	Plot demographic data as concentric circles

Miscellaneous Commands

Command	Definition
FIND	Print out X-coordinates and Y-coordinates for up to four nodes
SAVE	Save current network configuration in temporary form on disk
RCVR	Return to last SAVE
TANK	Recover command tank and execute encoded commands
CHGH	Change information in network header file
PACK	Create output network in bit-packed form

CONCLUSIONS

In a time of increased emphasis on public involvement in the transportation planning process, it is important to ensure that the process of generating alternatives remains responsive to public feedback. Although the number of alternative plans to be considered increases dramatically, the planning system cannot allow itself to get bogged down. Time elapsed between citizen question and system answer must not become unreasonably long; this would have the ultimate result of killing the public's newly found enthusiasm.

The interactive network graphics program described in this paper has helped relieve a potentially crippling burden on Michigan's planning procedure. It has eliminated a major part of the cost and time elapsed in generation and evaluation of alternatives. Because NETEDIT is much more interesting to use than manual methods, it also tends to eliminate a majority of the errors that can occur after hours of network coding.

Finally, it must be stressed that NETEDIT is a production technique used in hundreds of statewide model applications. It is the proven productivity that makes NETEDIT such a valuable aid in transportation planning.

REFERENCES

1. T. L. Gotts and L. G. Scott. Michigan's Statewide Transportation Modeling System, Volume 2-A: Efficient Network Updating with Interactive Graphics. Michigan Department of Transportation, Lansing, May 1976.
2. T. L. Gotts. Michigan's Statewide Transportation Modeling System, Volume 2-A, Appendix A: NETEDIT Command Directory. Michigan Department of Transportation, Lansing, June 1977.
3. T. L. Gotts. Michigan's Statewide Transportation Modeling System, Volume 2-B: Tree Plotting with Interactive Graphics. Michigan Department of Transportation, Lansing, Jan. 1977.
4. J. Newell. Michigan's Statewide Transportation Modeling System, Volume 3-B: Automatic Network Generator Using Interactive Graphics. Michigan Department of Transportation, Lansing, Aug. 1977.