

Technical Performance of ALI-SCOUT

FAST-TRAC Phase IIB Deliverable

If24B Final Results Report
EECS-ITSLAB-FT97-118

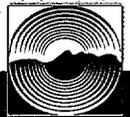
Marlin Ristenbatt, Principal Investigator
Ghassan Shahin, Engineer

FAST-TRAC Evaluation
Technology Planning and Evaluation Group (TPEG)
Intelligent Transportation Systems Laboratory
University of Michigan
Ann Arbor, MI 46109

Steve Underwood, Ph.D, Principal Investigator
Debra Demski, MUP, MSW, Project Manager

Program Offices Telephone
(313) 764 4333

Program Offices Facsimile
(313) 763 1764



Technical Performance of ALI-SCOUT

FAST-TRAC Phase IIB Deliverable

#24B Final Results Report
EECS-ITSLAB-FT97-118

Marlin Ristenbatt, Principal Investigator
Ghassan Shahin, Engineer

FAST-TRAC Evaluation
Technology Planning and Evaluation Group (TPEG)
Intelligent Transportation Systems Laboratory
University of Michigan
Ann Arbor, MI 48109

Steve Underwood, Ph.D, Principal Investigator
Debra Demski, MUP, MSW, Project Manager

Program Offices Telephone
(313) 764 4333

Program Offices Facsimile
(313) 763 1764

Technical Performance of ALI-SCOUT

FASMRAC Phase IIB Deliverable

#24 Final Results Report
EECS-ITSLAB-FT97-118

Prepared By:
Marlin Ristenbatt, Principal Investigator

Ghassan Shahin, Engineer

Table Of Contents

ACKNOWLEDGEMENTS	1
PURPOSE AND SCOPE OF REPORT1
1. Synopsis Of Ali-Scout And Its Components	2
2. Data Collection Methods	4
3. Route Description Of Data Sources6
4. General Overall System Performance13
5. Hardware Reliability Performance15
6. Dsrc Performance16
7. Mode Change Data	17
8. Commute Trip Performance Results19
9. Technical Performance Conclusions31
REFERENCES	32
APPENDICES33
Appendix A: VUC Software	1*
Appendix B: UM's Program to Collect Technical Performance Measures	8*
Appendix C: Unattended Enabling Circuit and Custom Made Computers..35*
Appendix D: Sample Of UM's Software Outputs Of Technical Performance Data..39*
Appendix E: Hardware Failure Overview	96*
Appendix F: Commute Driver Consent Form	101*

Note: These appendices are exact excerpts from previously published documents. Page numbers have been left for reference purposes, however, the numbers *are not* necessarily sequential between appendices.

ACKNOWLEDGEMENTS

We wish to acknowledge the individuals who contributed much to the ability to collect the data included herein. First, Siemens: Tom Bauer and Mel Rode, for early on volunteering use of their VUC software for our evaluation use; Peter Luchinski, for first briefing us on the VUC software, advising throughout, and responding with help to all technical memos and notes requesting specific information in using this software; Michael Wieck for supporting our effort and producing and supplying us with the hardware performance data (Appendix E); and finally the technicians, Gregg Deljudas, Benny Reed, and John Hartman whose cheerful cooperation in installing units in our commuter driver's vehicles was essential to gathering this data. We are grateful to Ivy Renga of Chrysler Corporation who enabled our recruiting of commuter drivers; without his help we could not have collected the data. In UMTRI we are exceedingly grateful to the Driver Behavior Group (Dave Eby, Lidia Kostyniuk, and Michell Hopp) for first accepting our equipment aboard their test vehicle and then giving us all cooperation for installing and giving us access to their vehicle during its operation. We also wish to acknowledge the Fast-Trac evaluation project manager at UM, Debra Demski for her help through out the course of the project, and in editing the intermediate and final reports. Finally we owe substantial debt to two UM participants: Zhihui Huang, who wrote the first version of UM software to read the VUC files, and James Daws who helped with the OrCAD software that was used to implement the printed circuit boards.

PURPOSE AND SCOPE OF REPORT

The FAST-TRAC (Faster and Safer Travel through Traffic Routing and Advanced Controls) Operational Field Test (OFT) is an Intelligent Transportation Systems (ITS) project being conducted in Southeast Michigan, managed by the Road Commission of Oakland County (RCOC). The project involves the deployment and evaluation of both an Advanced Traffic Management System (ATMS) and an Advanced Traveler Information System (ATIS). The ATMS is a traffic control system called SCATS. The ATIS is Ali-Scout, the subject of this report, a route guidance system which provides turn-by-turn directions to a destination selected by the driver. The directions seek to produce the fastest route between the origin and the destination.

This report describes the technical performance of the Ali-Scout system based on data collected during most of the 1996 calendar year. The bulk of the data reported was collected using In-Vehicle Storage (IVS) units that interfaced with the In-Vehicle Units (IVUs) of Ali-Scout. Additional data came from Ali-Scout system logs. The Driver Behavior Group (User Perceptions and Behavior), the Human Factors Group (Safety and Workload Analysis), and the Simulation and Modeling Group also performed evaluations of Ali-Scout (1,2,3,4,5,6). The plan for conducting the Technical Performance portion of the evaluation was designed to respond to FHWA's announced requirement to "measure any technical performance items that impact the evaluation of future deployments of the system".

The purpose of the Technical Performance evaluation was to assess the actual performance of all parts of the Ali-Scout system during the period of the user perception studies mentioned above. The evaluation includes the assessment of hardware, software, data bases, and operations. The Technical Performance evaluation was divided into:

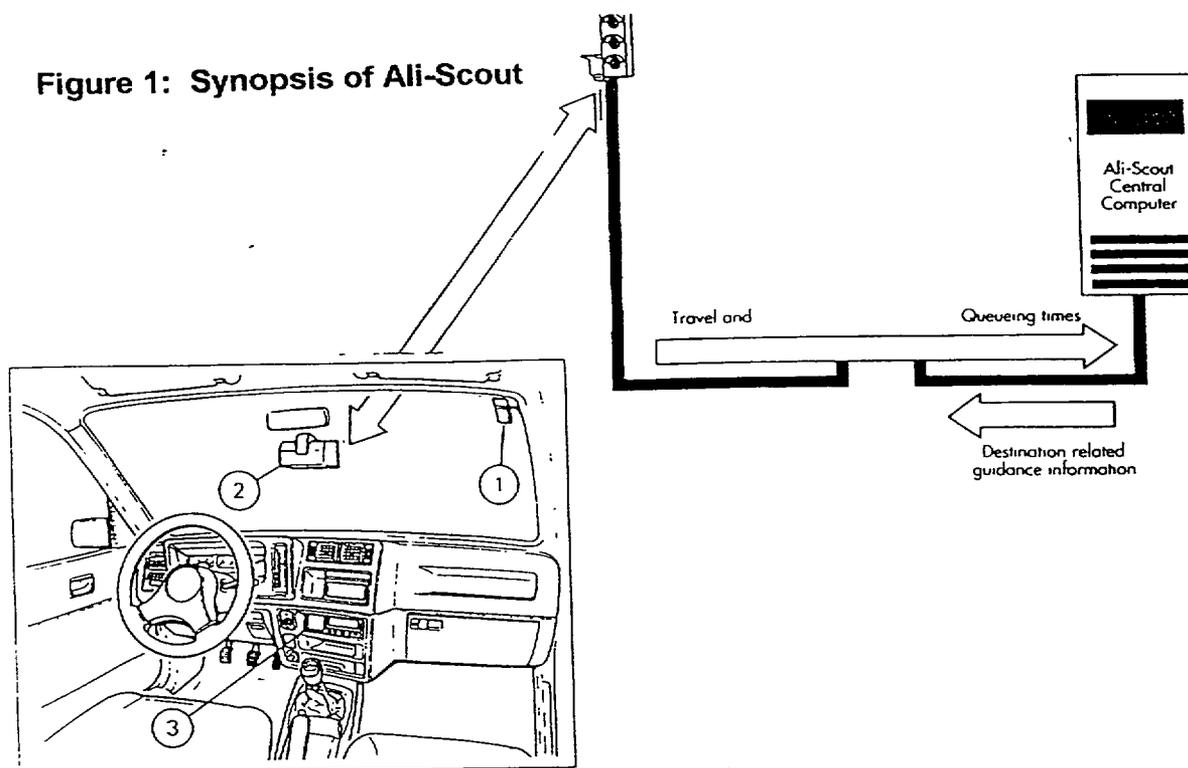
- 1) the general overall system performance; 2) hardware reliability performance; 3) the Dedicated Short Range Communications (DSRC)---formerly called vehicle-roadway communications (VRC); 4) the mode change data; and 5) the commute trip performance results.

1. SYNOPSIS OF ALI-SCOUT AND ITS COMPONENTS

The major characteristics of Ali-Scout, especially the major components whose technical performance was evaluated, are covered. Although detailed descriptions of Ali-Scout are available elsewhere (7,8) and will not be repeated here, a brief overview of the system has been provided. Ali-Scout is a centralized, beacon-based, self-contained and dynamic route guidance system. A static route guidance system uses only fixed link-times, such as speed limits and number of red lights, while a dynamic system aims to react to current traffic conditions. Ali-Scout's dynamic operation adjusts link times by time-of-day and day-of-week using historical database. We will see that Ali-Scout is capable of dynamic operation with respect to the time-of-day and day-of-week recurring congestion, but is less capable of real time adjustment in response to non-recurring incidents (see Sec. 5).

The DSRC between the roadside and the vehicle is implemented via an infrared (IR) beacon. The DSRC downlink (from central computer to vehicle) delivers recommended route information to the vehicle. Each beacon "localcasts" the recommended route links between that beacon and all other neighboring beacons; the DSRC uplink reports link travel times (and queues) experienced by the vehicle. It is the uplink probe reports from equipped vehicles that implement the self-contained traffic monitoring feature since an independent surveillance infrastructure, such as buried loops or video cameras, is not required to determine the state of the roadway network. Dynamic operation is achieved by using the current probe information in a weighted algorithm along with the historical probe data to influence the route choices sent out to the beacons

Figure 1 shows the major components of Ali-Scout (adapted from the Ali-Scout Users Guide). The In-Vehicle-Unit (IVU) is comprised of the Display Unit (DU), the Navigation Computer Unit (NAC), the Infrared Transceiver (ITR), and the electronic compass that is also called the Magnetic Field Sensor



1. an electronic compass to determine the vehicle's direction;
2. a receiver/transmitter (or transceiver) to communicate with roadside beacons;
3. the Display Unit (DU) and navigation computer to provide real-time guidance through LCD display indications and voice commands

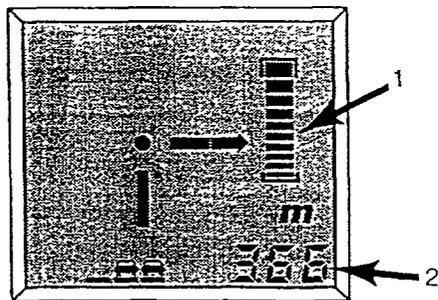
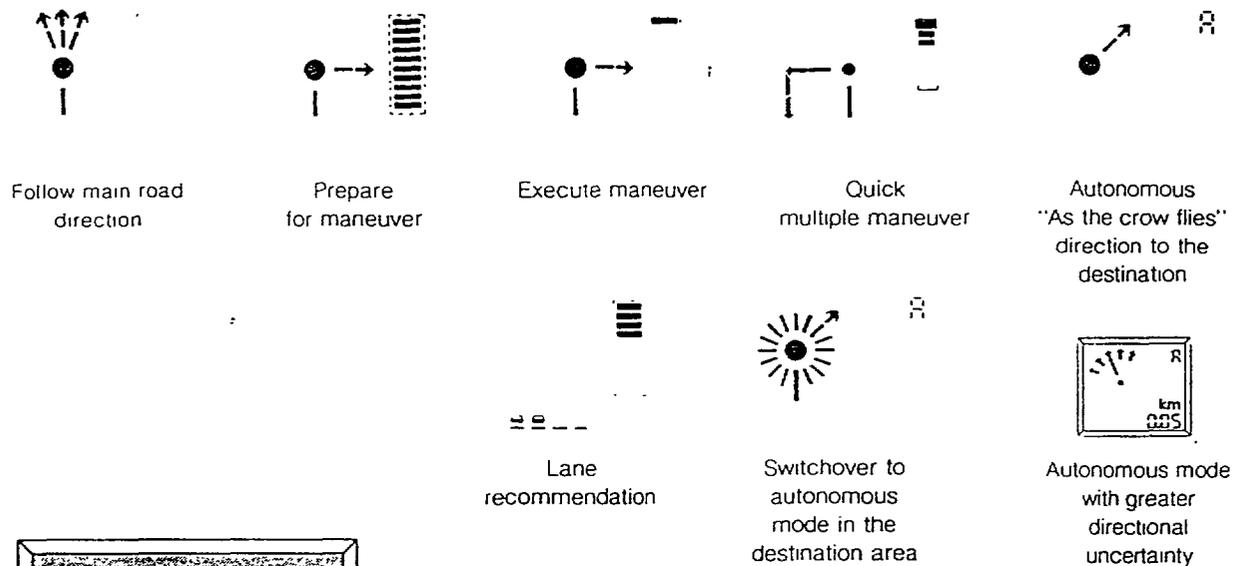
(MFS). The recommended route information from the Central computer is delivered to the vehicle via the infrared "downlink" (from beacon to vehicle), while the probe information containing vehicle experienced link times (and queue times) is delivered from the vehicle to the beacon via the "uplink."

When driving an Ali-Scout equipped vehicle the display unit is either in "autonomous" mode or in "guided mode." After the destination has been entered, or chosen from a stored list, the unit will be in autonomous mode until the first beacon is encountered. The display unit in this mode shows the "as the crow flies" direction and distance to the destination, shown in the top right display in Figure 2. While the direction alone serves as a rudimentary

guidance, the turn-by-turn guidance in the "guided" mode begins when the vehicle leaves the intersection where the first beacon is encountered.

Since guided mode starts soon after leaving the intersection, the first guided display is usually the "follow main road" forward arrows shown in the top left display of Figure 2. Sometimes a "maneuver" display is the first guided display (see Figure 2), if the maneuver is close to the initial intersection. If the driver follows the recommended steps, and the system performs properly, the unit will remain in guided mode until the vicinity of the destination is reached. During all this time the maneuvers will be communicated by the displays depicted in Figure 2, with each display change announced by an audio

Figure 2: Displays seen by driver



1. **Bargraph:** displays relative distance to the next maneuver. The number of bars decreases as your next maneuver nears.
2. **Distance:** The distance to your destination is continuously updated and displayed. "M" indicates miles.

cue. When the destination area is reached (about a quarter mile from destination) the unit will “switchover” from guided to autonomous mode, with an arrow indicating the direction to pursue. Following the arrow should bring one to within sight of the destination.

Approximately 100 beacons were deployed over an area of about 15 miles (E-W) by 17 miles (N-S); the beacon deployments appear (later) in Figures 4 through 9. The darkened circles represent the beacons. While Ali-Scout is intended to operate in a dynamic mode, with probe input from equipped vehicles, dynamic operation is only possible after historical link data is built up. Dynamic operation for the 100 beacons area began on May 1, 1996. During all the prior period of building up, the system operated in static mode. Static mode uses basic information such as speed limit of the links and number of traffic lights. In static mode the recommended route between any two points will always remain the same, no matter what time of day or day of week. In dynamic mode the drivers experienced different recommended routes, mostly responsive to recurrent congestion, and possibly responsive to non-recurrent incident congestion. On August 26, 1996 the system was again returned to static mode because the location of the Traffic Operations Center (TOC) was moved. The system was returned to dynamic mode, from the new TOC location in early November 1996.

The Technical Performance evaluation of Ali-Scout was accomplished by collecting data on the system's major components and their interconnections, which are shown in Figure 3. Each component in this diagram, and all the interconnections (including the power to the field units), must work reliably for the

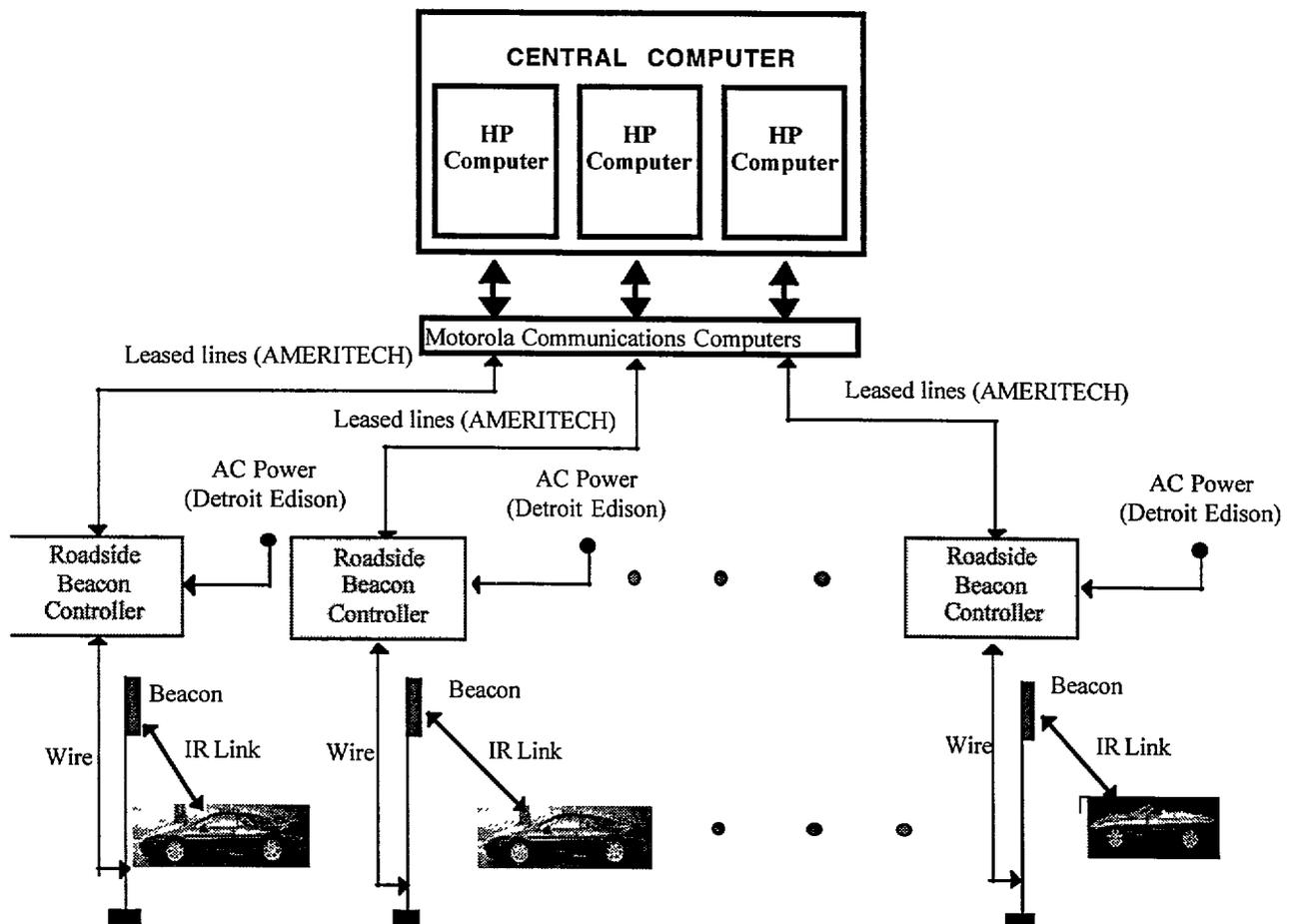
system to deliver its service. The roadside beacon controller stores the routing information from the central computer and “localcasts” it to any passing (equipped) vehicle. It also temporarily stores the incoming (from the vehicle) probe information, for relaying up to the central computer. Power is supplied to the beacon controllers (and the beacons themselves) by the Detroit Edison power company. Leased lines from the local telephone system (Ameritech) connect the central computer with the beacon controllers. The central computer communicates with the (approximately) 100 beacons via a Motorola communications computer. The central computer is comprised of three HP units, which split up the computing tasks for the entire geographical area.

2. DATA COLLECTION METHODS

The overall technical performance concept included obtaining data on the performance of each of the major elements shown in Figure 3. Data on the down time of the central computer, the connecting lines, and the beacon controller (and beacon) were obtained from Siemens records. In-vehicle data was collected in three scenarios:

- 1) During pre-planned trips in which recruited drivers drove a Mercury Sable (under the control of UMTRI's Driver Behavior Group) on a single out-and-back trip.
- 2) During the commute trips of 5 commute drivers each over a two month period).
- 3) During data runs using the UM van driven by the authors.

Figure 3



The Mercury Sable drivers were given a specific target destination and the Driver Behavior Group monitored their trip experience. The Technical Performance effort simply piggy-backed onto those trips as another means of collecting technical data on the system performance. The largest amount of Technical Performance data was collected via 5 commuters, (the second scenario) that had IVS equipment installed aboard for about 8 to 10 weeks each. Data was collected for an "apriori" period (before use of Ali-Scout) as well as during use of route guidance. The data from the UM van was sporadic in nature and was taken at intervals of every few weeks starting in January 1995 and continuing during the calendar year 1996.

The Technical Performance could only be measured from data recorded in the vehicle. The in-vehicle data was collected via In-Vehicle Storage (IVS) hardware and software that permitted recording per-

inent technical items. The measured parameters were: the beacon sequence (to detect non-functioning beacons); the mode history (from autonomous to guided and back to autonomous); the downlink success rate; the uplink success rate; and commuter trip-time experiences and speed histograms both before and after use of Ali-Scout.

The IVS used special "VUC" software provided by Siemens; this software includes the ability to replay a vehicle's trip, and store the transactions between the vehicle and the beacons. For Technical Performance purposes we were interested only in the transactions performance, thus only a part of the VUC software was actually used. The "EURO-SCOUT VUC MANUAL" specifies an "IBM compatible PC" as hardware, and in addition requires use of a proprietary HSCX communications card between the In-Vehicle Unit (IVU) and the recording computer. During the trips the VUC software creates a

trip directory that contains several sets of files. Each set of files is generated sequentially along the trip, and contains trip-specific information.

Starting with Siemens WC software (see Appendix A), UM software was written to read the WC files that were of interest to this effort (Appendix B.). Five individual computers were assembled (PC-size and 486 type) from stock parts, using fans capable of cooling the unit when inside a hot vehicle, and using a 12 volt power supply (as opposed to 110V). Since the WC software required manual operation it was necessary to design and build a custom “unattended enabling circuit” to permit automatic operation of the computer during the trips. This circuit (Appendix C) turned on the computer only after the vehicle was shifted from Park to either Reverse or Drive. It also delayed the removal of power from the computer after the vehicle was turned off, to permit registry of the files. The computer construction, the design and construction of the enabling circuit, and the data-read software were all done by Ghassan Shahin. Samples of the output data from this in-vehicle storage collection are shown in Appendix D.

3. ROUTE DESCRIPTION OF DATA SOURCES

As discussed in section 2, data was taken in three scenarios:

- 1) Pre-planned trips, using a Mercury Sable.
- 2) A set of 5 commuter drivers whose commute trips (only) were examined for technical and other trip performance.
- 3) UM van trips, generally sampling the entire network over a period of about two years.

The Technical Performance data showed some variation between the routes followed by the different sources, mainly because most of the beacons had consistent high performance while a few others had consistent poorer performance.

The pre-planned trips, using the Mercury Sable, part of the Driver Behavior Group’s Troika experiment on driver behavior, all originated at the (original) TOC on Big Beaver and Livemois and had three alternative destinations. The three destinations are indicated by “x” near the top right of Figure 4, which shows the beacon placement for the entire network. Figure 4 also shows the alternate routes for reaching the three destinations. All of these pre-planned trips were conducted during the dynamic operation of Al&scout.

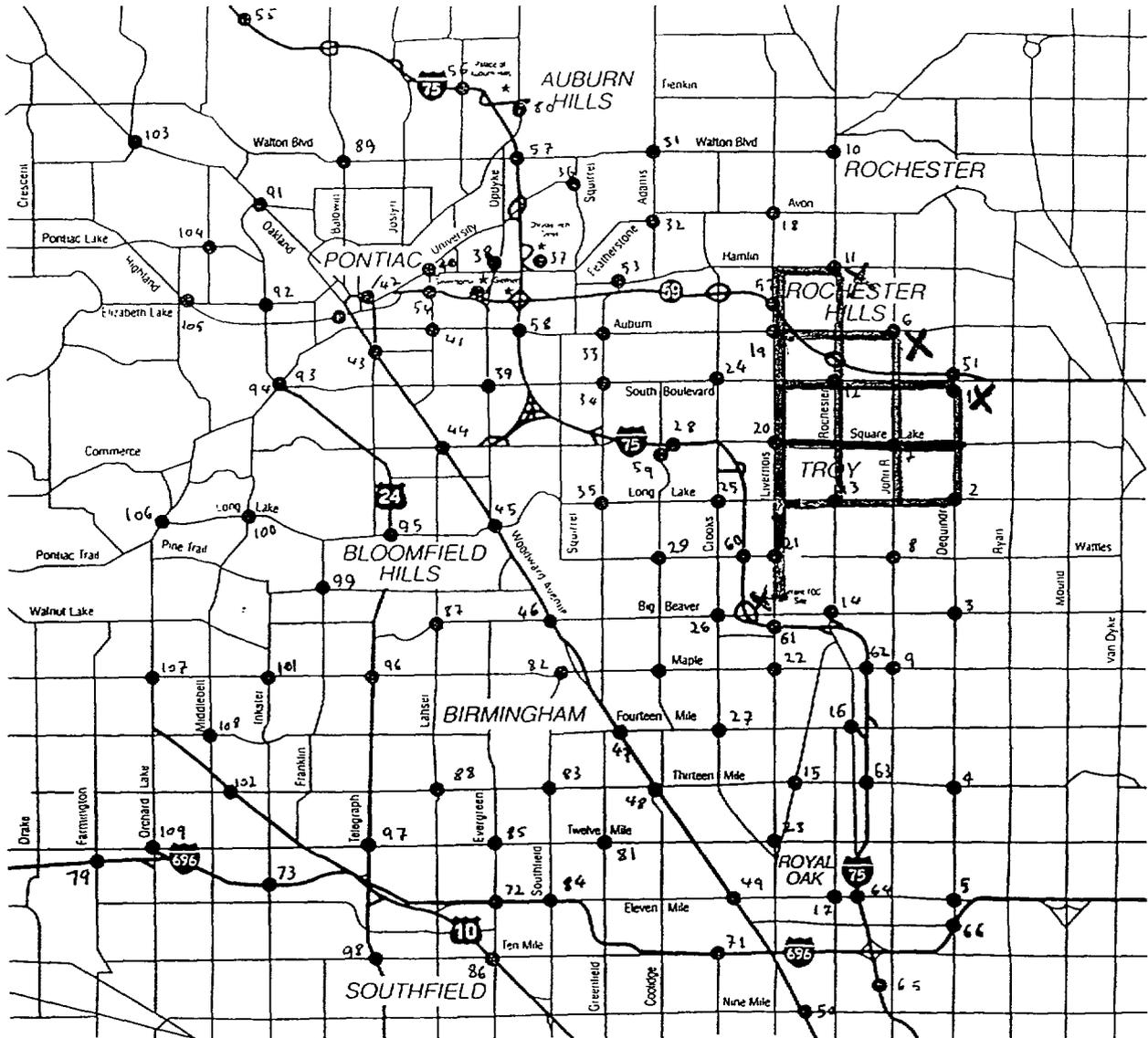
We began recruiting commuter drivers in early summer of 1996, and installation was coordinated with the construction of the computers. As noted before, IVS equipment was first used in an “apriori” period to record the pre-test trip experience of the commuter. Then the display head was installed and the drivers were asked to conform to the recommended routes, to measure any discernible change in either trip time or speed profile. The vehicle of commuter #5 had a manual shift (our custom enabling circuit was designed for automatic transmission) and kept blowing the diodes at the front end of the custom enabling circuit, so we had to abort using the IVS equipment for him. He agreed to manually record his trip-time data for his commute trips, both before and after the Ali-Scout installation, which we then used along with all the IVS data.

The general commute route for each of the five commuter drivers is shown in Figures 5 through 9. It is seen that the five different routes have little overlap and fairly represent much of the network. These routes and the particular beacons encountered will be repeatedly referred to (later) when discussing the performance.

Although the van, the third data collection source, was used to reach practically all of the network over the period of two years, many of the trips started in Ann Arbor and ended at Siemens (intersection of M59 and I-75). Hence the data from this source will be weighted by the beacon performance going east on I-696, up Telegraph, east on Square Lake road, onto I-75 north, momentarily on M59, using the Updyke exit, and into the Siemens complex.

FAST-TRAC

Oakland County, Michigan

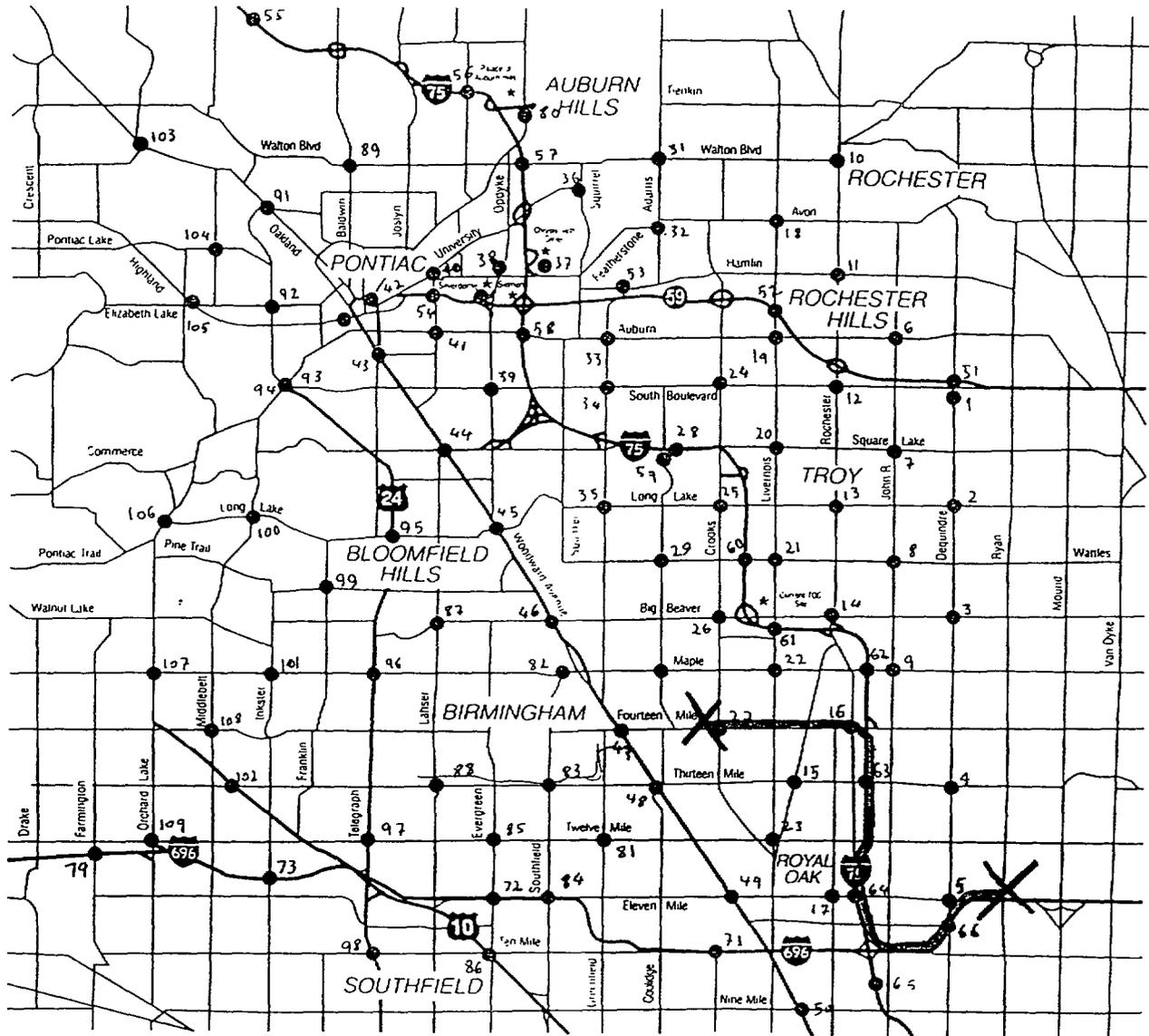


• Ali-Scout Beacon Sites

Figure 4: Pre-Planned Trips

FAST-TRAC

Oakland County, Michigan

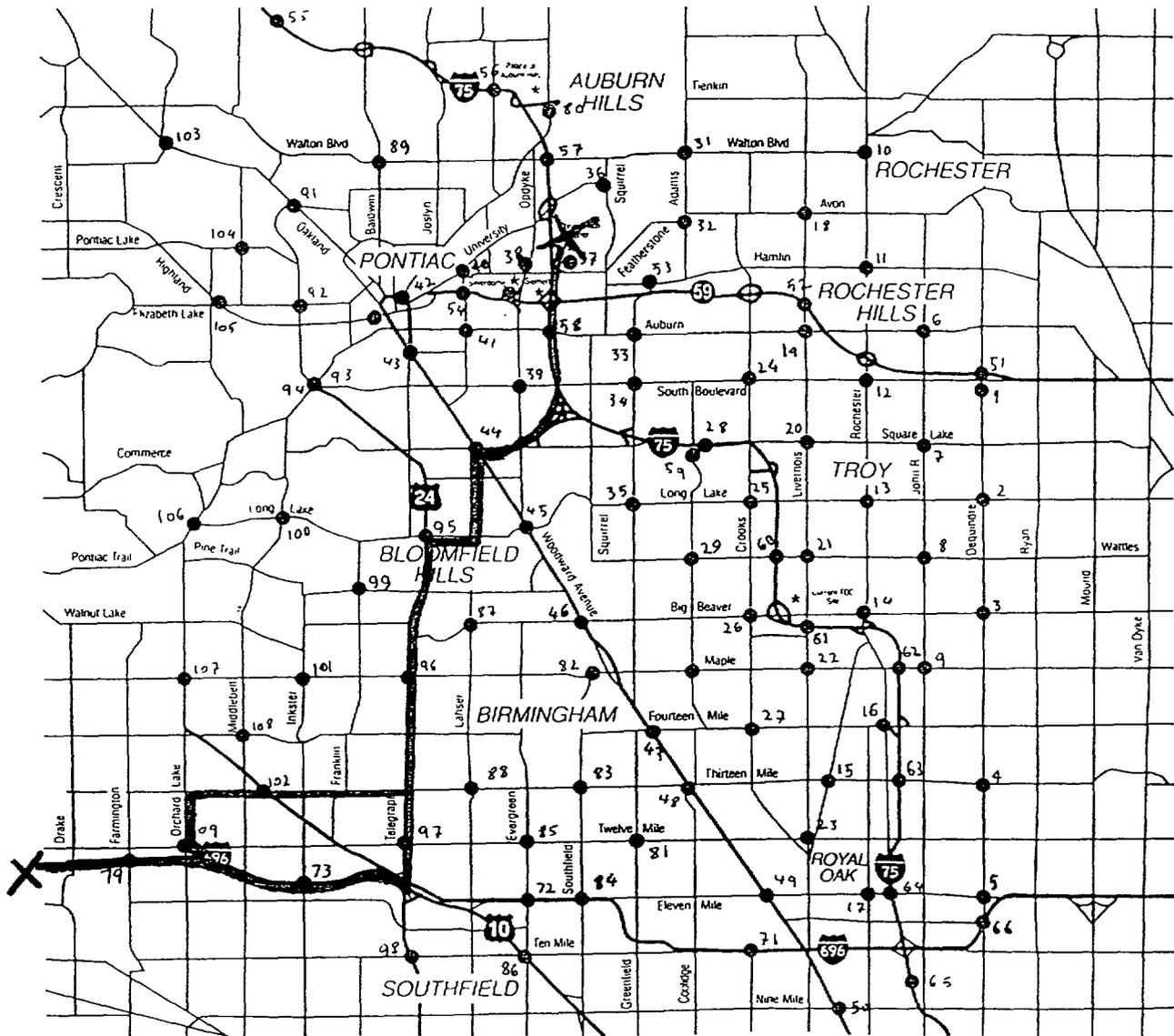


- Ali-Scout Beacon Sites

Figure 5: Commuter #1 Typical Route

FAST-TRAC

Oakland County, Michigan

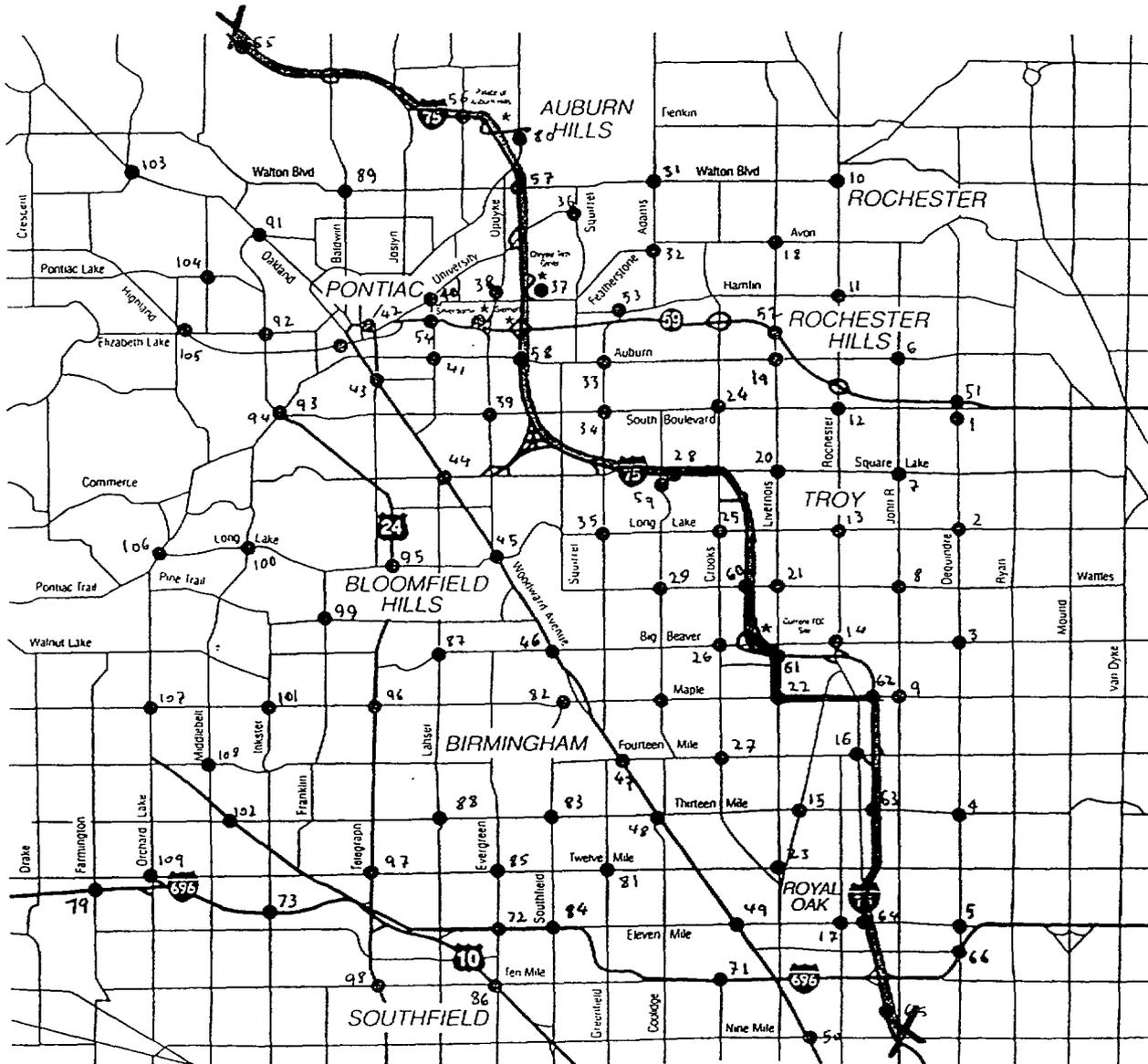


- Ali-Scout Beacon Sites

Figure 6: Commuter #2 Typical Route

FAST-TRAC

Oakland County, Michigan

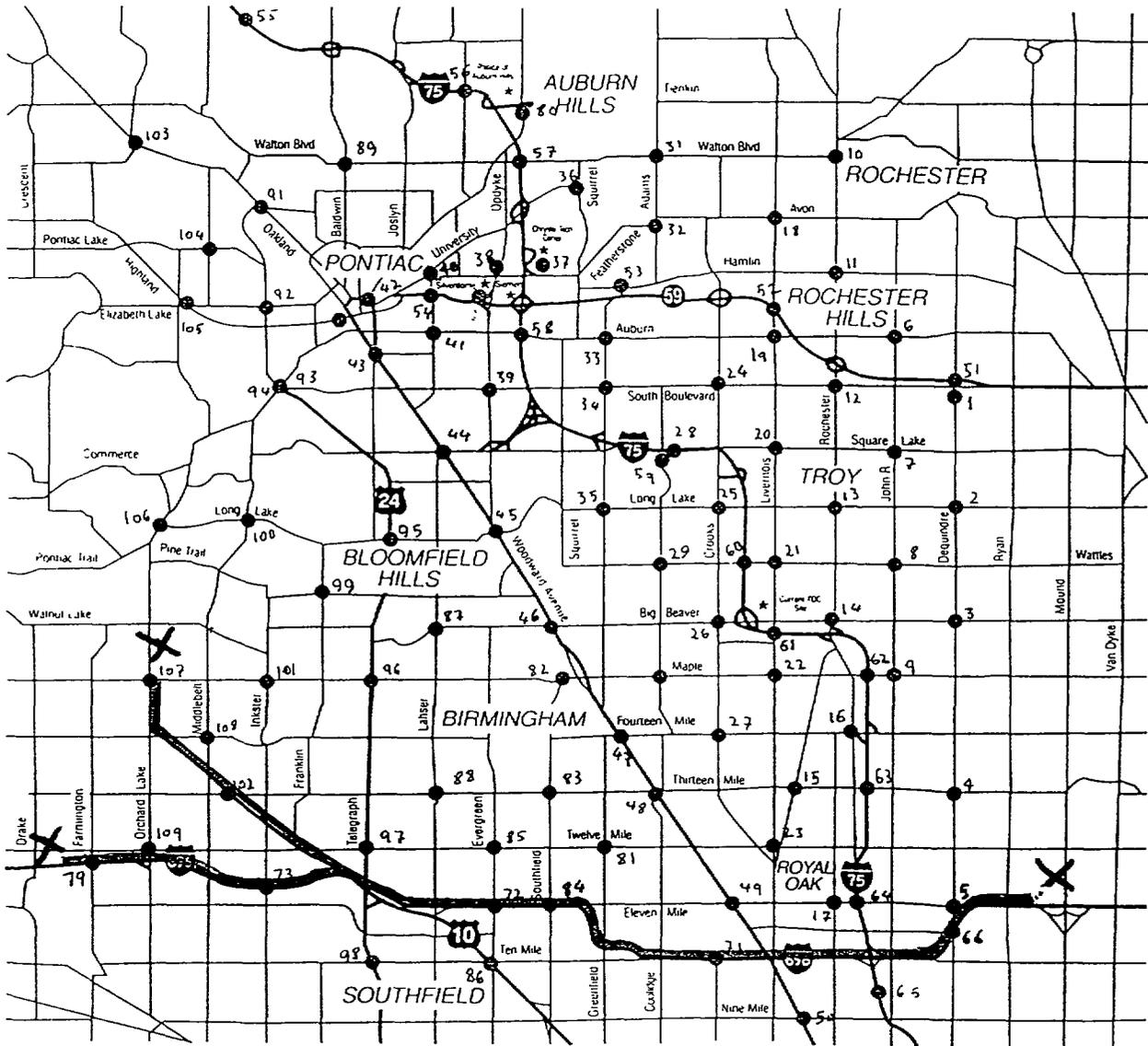


• Ali-Scout Beacon Sites

Figure 7: Commuter #3 Typical Route

FAST-TRAC

Oakland County, Michigan

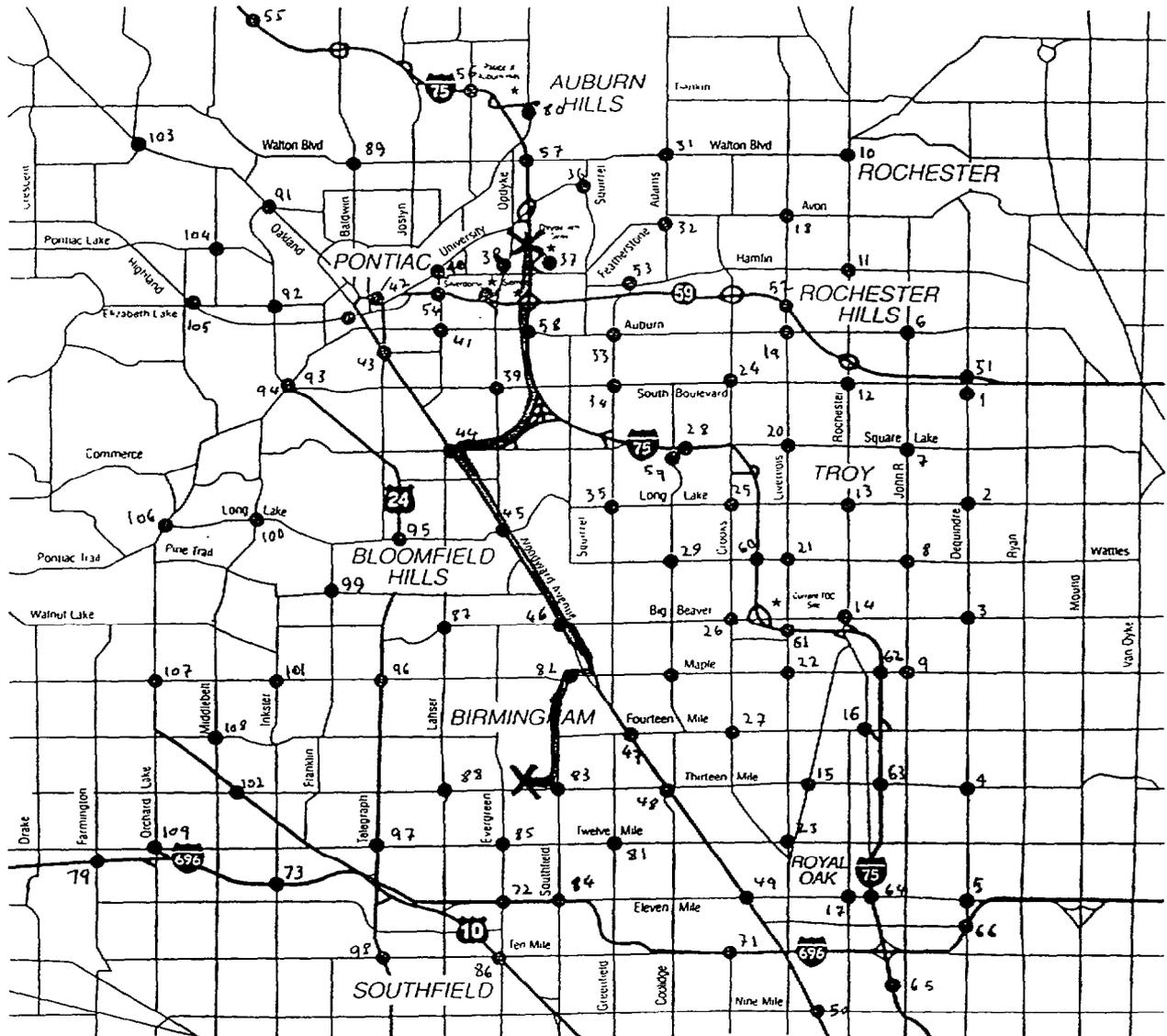


- Ali-Scout Beacon Sites

Figure 8: Commuter #4 Typical Route

FAST-TRAC

Oakland County, Michigan



* Ali-Scout Beacon Sites

Figure 9: Commuter #5 Typical Route

4, GENERAL OVERALL SYSTEM PERFORMANCE

Ali-Scout is a complex technical system comprised of a large number of field components (beacons and beacon controllers) and their linkages (see Fig. 3), in addition to the central equipment. Although the system was originated, installed, and operated by Siemens, it required also the prompt cooperation of both Detroit Edison (for power and power failures) and Ameritech (for communications and line failures). During this Operational Field Test Siemens employed dedicated effort by competent people in installing and maintaining this complex system. Also, Siemens cooperated in every way possible in supporting this Technical Performance effort. We observed that keeping the entire system in proper operating order proved to be a challenge, even with the dedicated effort. The results did show a consistent improvement in the behavior of the system comparing 1995 to 1996.

A critical aspect of any system that combines centralized route guidance with beacon communications is the delay, once in guided mode, in returning to guided mode after any interruption. The Ali-Scout display will depart from a guided mode display to the “direction and distance only” (autonomous) display when either:

- 1) The driver intentionally leaves the route (possibly not trusting the recommendation).
- 2) The driver unintentionally leaves the route (couldn't make the recommended maneuver due to traffic conditions).
- 3) The system malfunctions (possibly two successive beacons failed to downlink).

In any of these cases the display will remain autonomous until another beacon is encountered. In comparison, an autonomous (but static) vehicle-based route guidance system will recompute the route within about 7 seconds after leaving the (previous) route, but so far no dynamic in-vehicle systems have

been offered commercially in the U.S. (the VICS system in Japan does offer dynamic vehicle-based route guidance, (9)).

An ideal dynamic route guidance system will recommend, moment by moment, the best route from the present position to the destination; specifically, to react to both historically-predictable (recurring) congestion and quickly react to any enroute (non-recurring) incident so as to avoid encountering a huge slow-moving back-up or a total blockage. Rapid reaction to non-recurring incidents would require either:

1. Enough prompt probe reports, for the entire area covered.
2. A human operator that reacts to any reported (no matter the source) and verified incidents by manually inputting the data to the route selection algorithms.

In this Operational Field Test the involved vehicles were limited (about 500 vehicles total spread over the entire area) and no operator was employed. As operated, this system was unable to promptly respond to (non-retuning) incidents, for the following three reasons:

1. Using beacon-based probe information to detect incidents incurs an inherent delay; a link report for a link that has an incident experiences a delay directly proportional to the severity of the incident. If the link is completely blocked the link report is delayed until it is cleared; if partially blocked it is delayed until the vehicle can get to the next beacon reporting place. It is not possible to declare an incident if a link report is overdue since the driver may have stopped off for some purpose (gas, food, etc).
2. The second contributor to delay is the nature of the historical weighted algorithm used in Ali-Scout. During the relatively long period between incidents on a particular link the link time experienced is accumulated by day of the week and the time of day. This forms a historical database which, without the pres-

ence of non-recurring incidents, could implement a true dynamic route guidance since it is dynamic with respect to recurring congestion by period of time. Any new (current) link time report is given a 25% weighting and added to the historical database value (times.75). The rationale for this algorithm is that a single new link report cannot be trusted: the driver may have stopped for some reason. If enough vehicles traverse the affected link in a short time, the algorithm will slew over to the current value. However this feature has an inevitable slowing of response to a non-recurring incident. In this Operational Field Test there were not enough equipped vehicles to expect probe reports that would promptly slew the historical (static) link times to a current value in response to any incidents.

3. The last contributor to delay in responding to a non-recurring incident results from the series of delays in the reporting system itself (see Fig. 10). The first element is the delay from the beacon controller cycle. A beacon controller reports once every 3.64 minutes; depending on when the vehicle passed the beacon relative to that cycle, the delay can vary from practically zero up to 3.64 minutes. The next delay results from the computing cycle of the central computer (3.64 minutes). During this cycle the central computer receives all the new reports from the beacons and computes the new routes to be sent to all the beacons. The third delay in this series is the transmission time for the center to transmit the new information (64 Kbytes maximum) to the beacons (3.64 minutes).

The cumulative effect of the above three contributors limits the ability of the Ali-Scout, operating without manual intervention and only relatively sparse probe reports, to promptly respond to a non-recurring incident. Further, it slows the response to the “clearing” of any incident. Although it is difficult to measure and numerically characterize this “sluggishness” in dynamic performance, these basic facts lead us to

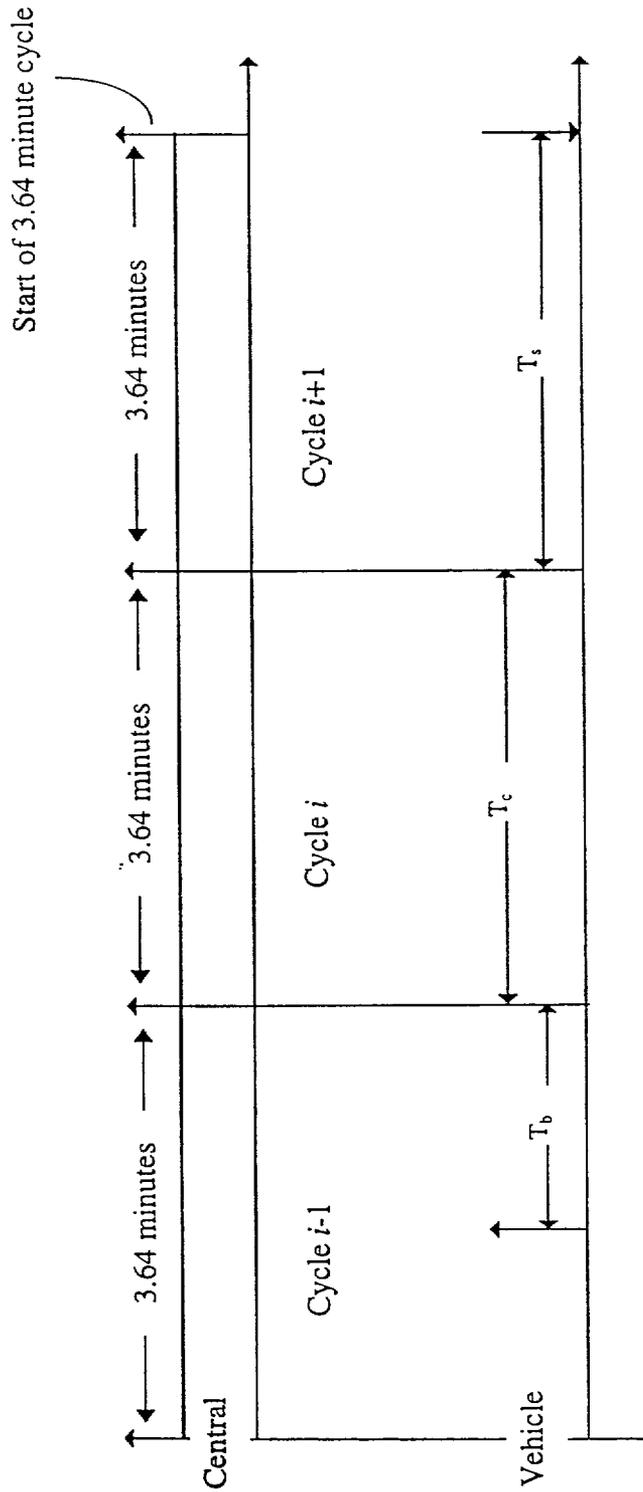
conclude that Ali-Scout was limited in its ability to provide a moment-by-moment optimum route choice in the presence of incidents that result in non-recurring congestion. Use of a human operator would improve this aspect. To be effective, a dynamic route guidance system should promptly (within a few minutes) react to any new verified incident. Many drivers currently receive dynamic information about non-recurring incidents via AM or FM broadcasts.

We see that it is difficult to implement a self-contained dynamic system that includes non-recurring congestion, due to the inherent delays. Competing in-vehicle-based route guidance systems will also be challenged in providing fast-response dynamic operation. Ali-Scout would be an effective dynamic route guidance system if only recurring congestion is present, but many commuter drivers also become expert about recurring congestion without route guidance systems.

The post-intersection decision at the first beacon encountered is another general feature warranting comment. Ali-Scout is in the autonomous mode until the first beacon is encountered, and it provides turn-by-turn guidance only after that first intersection is traversed (i.e. the intersection exit is presumably chosen by the driver on the basis of the crow-fly arrow). In some cases the optimum route from that first intersection would direct the driver to choose an intersection exit different from the one chosen by the driver on the basis of the direction arrow.

The final issue to discuss under the overall system is the role of beacon density. Although a high beacon density is more costly for both construction and maintenance it would serve the user by both:

- 1) Providing an earlier and more frequent encounter of the first beacon (to go into guided mode).
- 2) Lower the incidence of drop-outs where the guidance goes from guided to autonomous even when on the intended route, due to running out of guidance directions (too far from last beacon). With a relatively sparse beacon



Where:

T_b is the delay at the beacon which can be 0-3.64 minutes depending on when the vehicle passes the beacon.

T_c is the delay incurred at the center which is exactly one cycle long (3.64 minutes).

T_s is the time for the center to transmit the new route information (64 Kbytes max.) to the beacon (3.64 minutes.).

In the worst case, the reporting delay from the time an incident is reported to a beacon and the time the incident is made known to all beacons in the form of route information is thus $3.64+3.64+3.64=10.92$ minutes. In the best case, the delay is: $0+3.64+3.64=7.28$ minutes (shorter if route information <64 Kbytes). On average expect 9 minutes delay.

Figure 10: Ali-Scout Incident-Response Reporting Delay Times

deployment many trips within the equipped area are driven entirely or nearly without any route guidance because no beacon was encountered.

5. HARDWARE RELIABILITY PERFORMANCE

We obtained all data on hardware reliability from Siemens records; a detailed report was supplied to us covering the three months of May, June and July of 1996 (Appendix E).

This report will be the major basis for our evaluation of hardware reliability.

Although most of our Technical Performance evaluation data was collected during the last half of 1996, Ali-Scout was actually under construction and operation (initially over a smaller area) since mid-1992. We started collecting IVS data in early 1995, using the UM van, and continued through 1996. During 1995 and into early 1996 we noticed a steady improvement in the behavior of the system; Siemens added lane positions in more cases, actual guidance instructions were clearer, and fewer malfunctions (of various kinds) occurred. Such steady improvement is normal when dealing with such a complex system that was also under expansion.

Overall about 500 vehicles were eventually involved in Ali-Scout, usually driven for about a year. Thus we estimate that about 500 vehicle-years approximates the total exposure to hardware or software "failure". Appendix E shows that 45 in-vehicle units (DCU and NAC) failed up through July 1996, with about 10 more failures of either the infrared transceivers or the compass. Thus in-vehicle failures appear to be less than 10% (and may be lower since the 500 figure is only an estimate), which appears low in light of the fact that the overall system is closer to prototype stage than large production stage.

The failure report of Appendix E includes a detailed list of the beacons in a failed (non-operative) state, and the number of days they were in that state. We used that data to develop a coarse quantitative picture of the beacon reliability. The list includes bea-

con failures carried over from the previous month as well as failures from the particular month. When a beacon was "carried over" to the first reported month (May) we assumed it failed in the middle of the previous month, and assigned 12 business days in failure. We also assumed the days in failure shown in the lists are all business days. Using business days and the data from Appendix E, we estimate that out of the 66 business days in the three months multiplied by the 100 beacons, the number of beacon days in failure for May was 262,255 in June, and 146 in July. Dividing the sum of the days in failure by the 6600 beacon days results in a "failed beacon" rate of about 10%. There is reason to believe it was lower than this for ensuing months; the rate for July was 6.3%. Our general experience with driving the van over the period, over much of the network, confirms that the failure rate was in this range. In sum, we estimate the beacon failure rate to be about 7%.

We further analyzed the beacon failure data in Appendix E as to the cause of failure. Thirty-three percent of the failure days in the three months (using business days as noted) were due to "Ameritech line problems". Nineteen percent of the failure days were due to "beacon off/ bad data". Eighteen percent of the failure days were due to "not on line yet", and all appeared in the first reported month (May), as expected. Fifteen percent of the failure days were due to beacon controller card failures. "Power off," including tripped breakers, accounted for 8% of the failure days, with "power down and reset" accounting for about 3%. This breakdown gives a general picture of the cause of the beacon failures.

According to the report in Appendix E the central computer worked without problems in May and July. In June there was a period of malfunction, which ". . . happened repeatedly for approximately 20 days..." which was found to be due to two "impossibly large" files that were removed. During any such period of malfunction the system would have reverted to static mode rather than dynamic. Hence the drivers would still receive route guidance, but based on the static data base rather than the historical by time of day one.

6. DSRC PERFORMANCE

The Dedicated Short Range Communications (DSRC), previously referred to as Vehicle-Roadway Communications (VRC), is comprised of both the downlink from beacon to vehicle and the uplink from vehicle to beacon. The downlink performance is the most basic technical measure of Ali-Scout since it conveys the route instructions to the passing MJ at each beacon. We used downlink data from all of the three scenarios, as shown in Table 1, except that commuter driver #5 did not have IVS equipment aboard (as noted before). Note that the total data count is 2060 beacon encounters, and that the data from the commuters was collected during the rush hours. Three of the data sources, the van and commuter drivers #2 and #3, experienced a downlink success rate of about 98%. Commuters #1 and #4 experienced a considerably lower rate, around 70%. Note that these numbers apply only to working beacons; if a beacon is inoperative it does not register in this data.

The route of commuter #1 passes beacons 16,27, 64, and 66 (see routes in Sec.4). Beacons 16 and 27, which are on 14mile road, never missed a downlink during the 51 recorded trips. Beacon 66, located on I-696, failed to downlink 24% of the time; beacon 66, located on I-75, failed to downlink 76% of the time during these 5 1 trips. It is possible that expressway beacons are more vulnerable to malfunction than street beacons, possibly due to the speed element, although many of the expressway beacons had records in the 98% area.

Commuter #4 uses a typical route that passes beacons 107, 102, 72, 71, 65, and 66. Six beacon encounters were expected from this data, but for

much of the time that the IVS equipment was aboard two of these beacons were inoperative (recall that inoperative beacon data is not included in the downlink data). We have reason to believe that commuter #4, and also possibly #1, were fast drivers; this might have contributed to the reduced downlink success data (see Speed data in sec. 9). Nevertheless, the data indicates that the overwhelming majority of beacons had excellent downlink performance while a minority had persistent poorer performance

In interpreting the impact of downlink data it is important to note that the system is arranged so that instructions for two successive beacons are included at each beacon. Thus a single downlink failure should not result in failed guidance to the next beacon, but two in succession would likely result in leaving guided mode before the next beacon is reached. In light of this we conclude that the downlink data of Table 1 is more than adequate to support guidance of a typical trip.

The uplink carries the information about link travel experience to the central computer

Table 2 shows the success rate for the uplink transactions for each of the three sources of trips: pre-planned trips, UM van trips, and commuter driver trips. There are only a total of 1345 uplinks encounters, from all sources (compared to 2060 downlink encounters), because no uplink occurred during any static operation of the system. All the pre-planned trips occurred during dynamic operation. We had to sort the data from the UM van and the commuter trips into static and dynamic periods of operation to obtain the data in Table 2. Uplink statistics do not include encounters between August 27 1996 and November 7 1996, when the system became static to

Table 1: Down-link Performance

	Pre-Planned Trip	UM-Van Trips	Commuter#1	Commuter #2	Commuter #3	Commuter #4	Total
Encounters	224	227	720	507	204	178	2060
Successes	195	221	507	500	199	122	1744
Percent	87.0	97.4	70.4	98.6	97.6	68.5	85.0

move the TOC. The uplink performance was much poorer than the downlink performance, hovering around 50% for the pre-planned trips and the UM van and even lower for the commuter routes. The very low 19% for commuter #1 is due to the persistent poor behavior of beacons 64 and 66. The data supports the logical expectation that beacons showing a relatively poor downlink performance will also have a poor uplink performance. A logical reason for the uplink performance being poorer than that of the downlink is that the uplink IR power density at the beacon may be lower than the downlink density at the in-vehicle transceiver. If the DSRC has an uplink

failure at a given beacon encounter the in-vehicle transceiver will make another attempt to include that data at the next beacon encountered. Therefore it is not possible to conclude specifically how damaging a 50% (or lower) uplink rate is to proper performance of the dynamic operation; but a low rate might contribute to reduced performance if the number of probes itself is low. It is clear that the level of importance of a down-link is higher than an up-link. A down-link failure may lead to loss of guidance, whereas an up-link failure may be compensated by the passage of another vehicle.

Table 2: Up-link Performance

	Pre-Planned trips	UM-Van Trips	Commuter #1	Commuter #2	Commuter #4	Total
Encounters	224	160	676	107	178	1345
successes	114	73	126	34	73	420
Percent	50.9	46	19	32	41	31

7. MODE CHANGE DATA

Since Ali-Scout first starts in autonomous mode (direction and distance only) before entering guided (turn-by-turn) mode, and then returns to autonomous mode near the end, the ratio of guided time to the total time (guided plus autonomous time) should be one crude measure of the “power” of the route guidance. For example, if a large fraction of the trips have a high ratio it strengthens the contribution of the guidance; conversely, if a large fraction of the trips have a small ratio of time spent in guided mode relative to total time, the system’s contribution to the trip is minimized.

While we found no way to develop quantitative data on this crude measure we nevertheless report our perception. Our van experience demonstrated that, if one puts in a destination from within the equipped area and sets out according to the crow-fly direction indicated, a significant portion of the trip will be in the autonomous mode, in many cases, due to not encountering a beacon until late in the trip. An Ali-Scout-expert driver of course would first head for a

beacon; in fact, there is an operation possible with the system where direction to the nearest beacon is indicated, but this may be more complicated than typical users desire. For anyone approaching the equipped area on an expressway the guidance will of course begin when the first expressway beacon is passed.

We were surprised to see the amount of time spent in autonomous mode by the pre-planned trip drivers. They were told specifically how to get to the first beacon, which should have resulted in a few-minute autonomous time; many drivers seemed to wander around before finding that first beacon; possibly they were experimenting with the system.

Another mode measure of interest is the behavior between the “first and last beacon”. The intention here is that there should be no departure from “guided” mode between these two points. Any lapse into “unguided” (autonomous) mode in this interval is due either to:

1. Intentional departure from the route by the driver.
2. Unintentional (couldn't make the turn indicated even though the driver wished to) departure from the route.
3. The system ran out of route guidance due to failed downlink(s) or a "beacon-down."

This "percent of time guided between first and last beacon" is an overall measure of the system's ability to contribute to the trip, somewhat independent of the cause. Any departure from 100% that is due to the driver intentionally leaving the route would indicate driver's disagreement with the recommendation; if it were due to unintentional route departure it would be a measure of how frequently driver's are caught in situations where they can't comply; if it is due to system running out of guidance information it is a direct measure of system performance. Thus this measure has some value in judging the merit of this route guidance system if collected under typical driving conditions even though our data could not distinguish the cause. It's simply an overall indication of "ability to contribute".

The "percent time spent in guided mode between first and last beacon" data is shown in Table 3, using all the data sources appropriate for this measure. We did not include our van data since these trips were of a test nature and not a "natural use." The pre-planned trips were included since they represent a natural use. After checking that the first and last beacons were in fact working the commute driver trips were sieved to include only those in which the driver drove directly from home to work, or vice versa. Table 3 also includes the downlink success rate for that source. Most of them are quite high so that "running out of guidance" is not indicated as a significant contributor to any departure from 100%. Table 3 shows that the percent time in guided mode ranges between 70% and 84%, with an average of 79.3%. This indicates that, for the route recommended, the system has a high probability of functioning as intended, including the driver's behavior.

Table 3: Percent Time In Guided Mode Between First and Last Beacon

	Pre-Planned Trips	Commuter #1 Morning	Commuter #2 Morning	Commuter #2 Evening	Commuter #3 Morning	Commuter #3 Evening	Commuter #4 Morning	Commuter #4 Evening	Summary
Number of trips	32	21	13	13	2	3	12	8	104
Down link success rate	130/147 88.4%	53/83 63.9%	102/103 99%	100/100 100%	15/15 100%	18/19 94.7%	29/51 57%	25/25 100%	454/524 86.6%
Average time between 1 st and last beacon (minutes)	12	15	32.6	33.4	17.6	21.9	14.3	15.2	1885 total
Average guided mode time between 1 st and last beacon (minutes)	10	11.7	22.8	26.2	14.5	8.8	8.8	12	1460 total
Percent guided mode	83.3	78	70	78	83	21	84	76	79.3

8. COMMUTE TRIP PERFORMANCE RESULTS

The final data taken for this technical performance evaluation was “before and after” commute trip data for a total of 5 commuters. The IVS equipment was installed in the 4 vehicles (the 5th driver provided manual data via a manual log) for a period of 4 to 6 weeks while the drivers used their usual commute routes and decision processes. Following this, the Ali-Scout was activated for them (by installing the display unit) and they were each asked to follow the advice: in general most drivers did so, knowing they were involved in a serious evaluation. However some drivers admitted to occasional intentional departure due to disagreement with the recommendation.

Note that this data collection effort is not exactly “technical performance” but rather “commute trip experience” in nature. This was “data of opportunity” since we needed systematic drivers to collect the DSRC technical data and all the IVS equipment needed was already available for that data. Hence recruiting drivers whose commute trip took them through the equipped area permitted us to also collect this commute trip performance data.

The 5 recruited drivers were found via a computer network of the Chrysler Corporation; and each of them was technically oriented and showed a definite interest in participating in this evaluation. The first commuter IVS installation occurred in June, 1996 and the last removal was in early January, 1997. Since Ali-Scout was changed from dynamic mode to static mode from August 27, 1996 to November 7, 1996, some of the commuter drivers experienced both dynamic and static operation. We separated any dynamic mode data from static mode data in the following trip-time data.

The trip-time data here, along with the overall reactions of the commuters, appear important since the commute is the most frequent trip, and one of the most important. We obtained credible trip-time data from 4 of the 5 commuters. Commuter #3 had to

take his vehicle to a repair shop soon after the display head was inserted, resulting in only a few legitimate trips. Consequently that data is based on only a few trips.

Figure 11 shows the histogram of travel times for the morning trips of commuter #1; the evening trips almost never went from work to home, and were not useful for trip time comparisons. Some of the trips occurred during static operation of Ali-Scout, and are shown separately from those in dynamic mode. Figure 11 shows total trip time histograms on the left, and trip time histograms between first-to-last beacon on the right. We examined both in order to check whether any differences would surface. The commuter route for #1 was shown on Figure 5; it occurred in the Southeast portion of the equipped area.

The left side of Figure 11 reveals that the average total trip time for the morning commute of #1 remained the same, with a small increase in standard deviation. The right side of Figure 10 shows that the average first-to-last beacon time increased slightly with Ali-Scout use and the standard deviation increased. This commuter reported guidance inconsistencies, especially around 15 mile and 16 mile road.

Figure 12 shows the trip time histograms for the morning commute on the left and the evening commute on the right for commuter #2. We used first-to-last beacon time for this commuter since a large component of the route continued outside the equipped area. Most of the Ali-Scout guided trips were during static mode with 5 trips during dynamic mode. The commute route in the equipped area for #2 was shown in Figure 6. During static operation with Ali-Scout the average first-to-last beacon time increased slightly (over prior non-Ali-Scout use). During dynamic operation the average morning time stayed the same, but the evening time increased slightly. This driver commented that, while the guidance recommendation was generally followed, he sometimes deviated due to disagreeing with advice.

Figure 13 shows the histogram of the first-to-last beacon trip time for commuter #3; note that the number of relevant trips was small for this commuter. Ali-Scout reduced the average morning trip time by 10% and the evening average by 4.8%. The number of trips involved is too small to generalize this result. Also, this commuter mentioned that he found himself driving faster, using Ali-Scout, which could account for the lower average time.

Figure 14 shows the histogram of trip times for commuter #4, using the time from origin to destination for both morning (to work) and evening (to home) commute trips. The commute route for #4 appeared in Fig. 8; a major part of the route was on the expressway that forms the southern border of the equipped area. The part of the route that offered options was at the western part, between his home and beacon #72. The left side of Figure 14 shows the “before” and “after” trip time histogram in minutes for the morning commute, while the right side shows similar histograms for the evening commute. All of these trips experienced the dynamic mode of Ali-Scout. For the morning trips we see that the average trip time actually increased by 9.3%, but the standard deviation decreased. In the evening the average trip time stayed the same but again the standard deviation was down. The driver reported that, from time to time, different routes at the west end of the route were recommended, testifying to the dynamic operation. Also, beacon #66 was not working for the first few weeks of Ali-Scout use by commuter #4.

The data of Figure 14 suggests that Ali-Scout tended to reduce the longer trip times resulting in the lower standard deviation, but delivered essentially the same or even greater average trip times. This commute route had limited options for about two-thirds

of the trip (the expressway portion), which made it difficult for a route guidance system to improve travel times.

Figure 15 shows the total trip-time histograms for commuter #5, whose data was provided by manual logs. The commute route for this driver was shown on Figure 9. All the Ali-Scout data occurred during dynamic operation. We see that following Ali-Scout guidance increased the average morning total trip time by 11.5% while the average evening time remained the same. According to the driver, the morning increase resulted from routing abnormalities at the Woodward and Maple area. If these had been corrected, the driver predicted the travel times should have remained the same.

The commute trip data reveals that utilizing Ali-Scout guidance did not reduce the average trip time in any of the substantial data cases (it did decrease for the few trips of #3). The average trip times either remained the same or increased slightly. In many cases, the trip time standard deviation decreased which is desirable. It must be recognized that the most difficult goal of any route guidance system is to serve the drivers that already are “expert” in their commute routes and usually know the myriad alternatives between origin and destination. To be helpful, the system must tell them something they don’t know, which, as noted previously, is predominantly the non-recurrent incident congestion. But we have seen (Section 5) that Ali-Scout responds quite slowly to such non-recurrent congestion if no central operator is present and relatively few probe vehicles are dispersed throughout the network. Further, the dynamic Ali-Scout system will still operate over the “beacon to beacon” pathways, while the unequipped commuter will utilize a richer variety of possible routes and connecting roads (sometimes through neighborhoods).

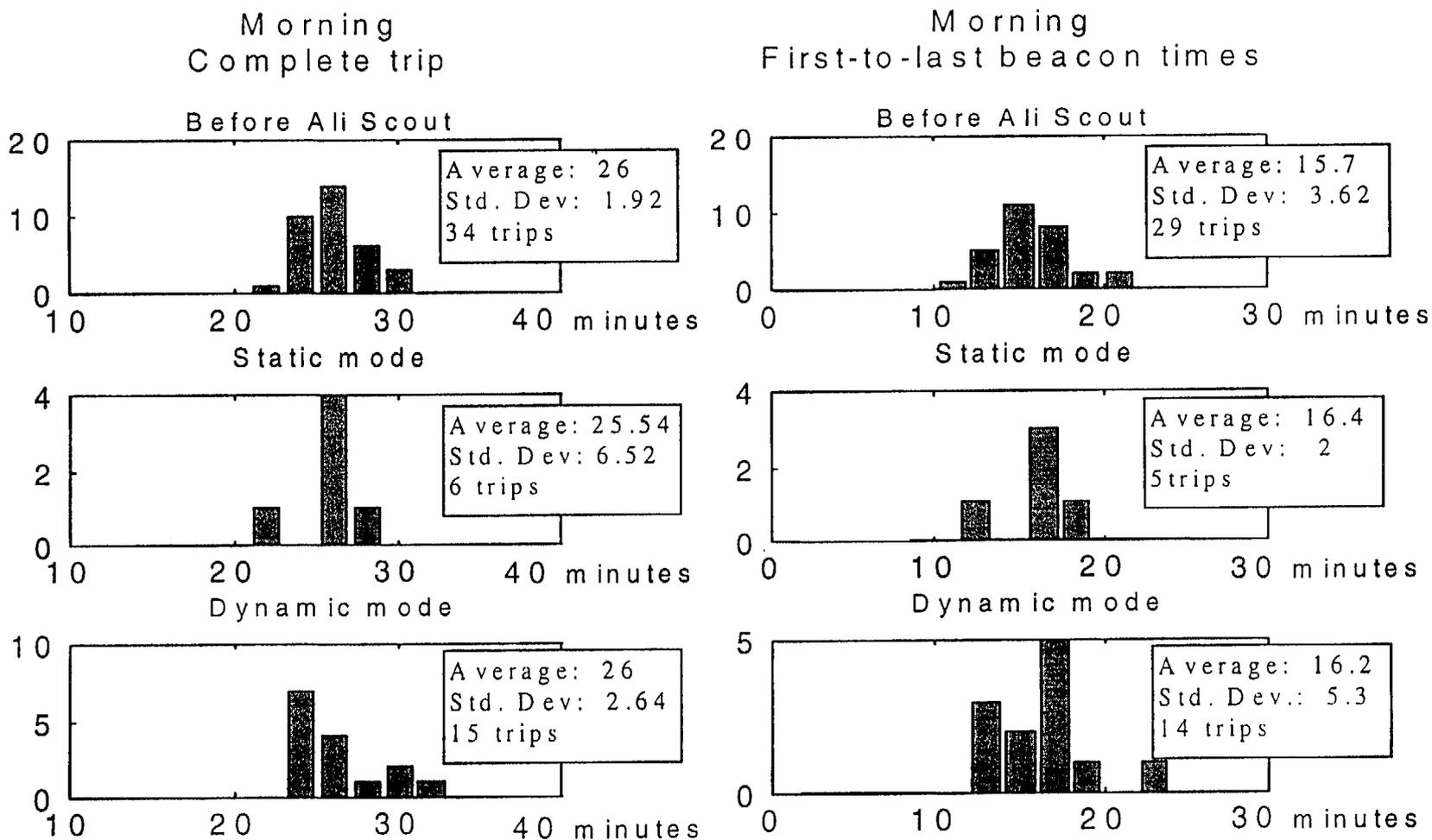


Figure 11: Trip Time Histograms for Commuter #1

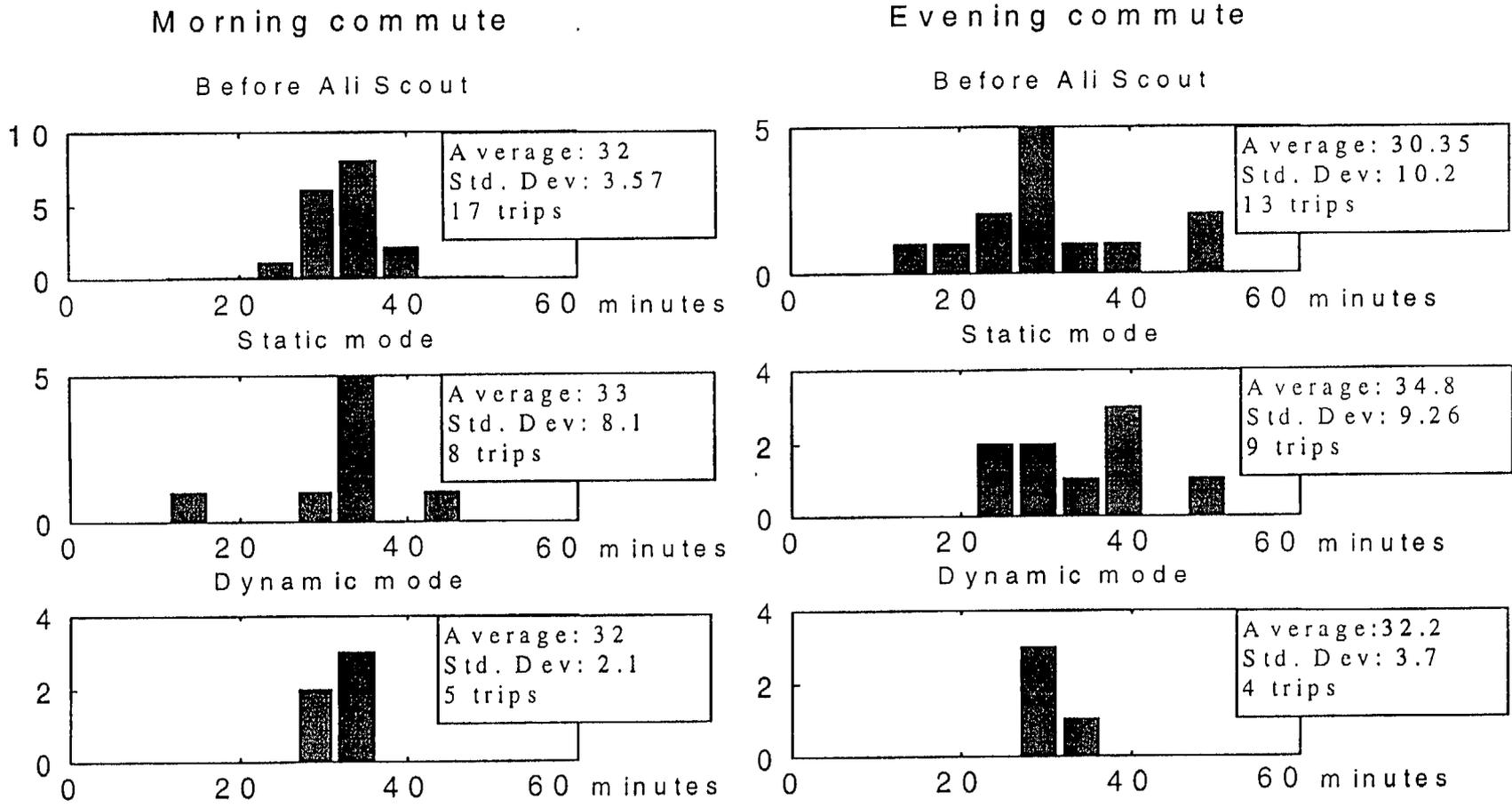


Figure 12: First-To-Last Beacon Trip Time Histograms for Commuter #2

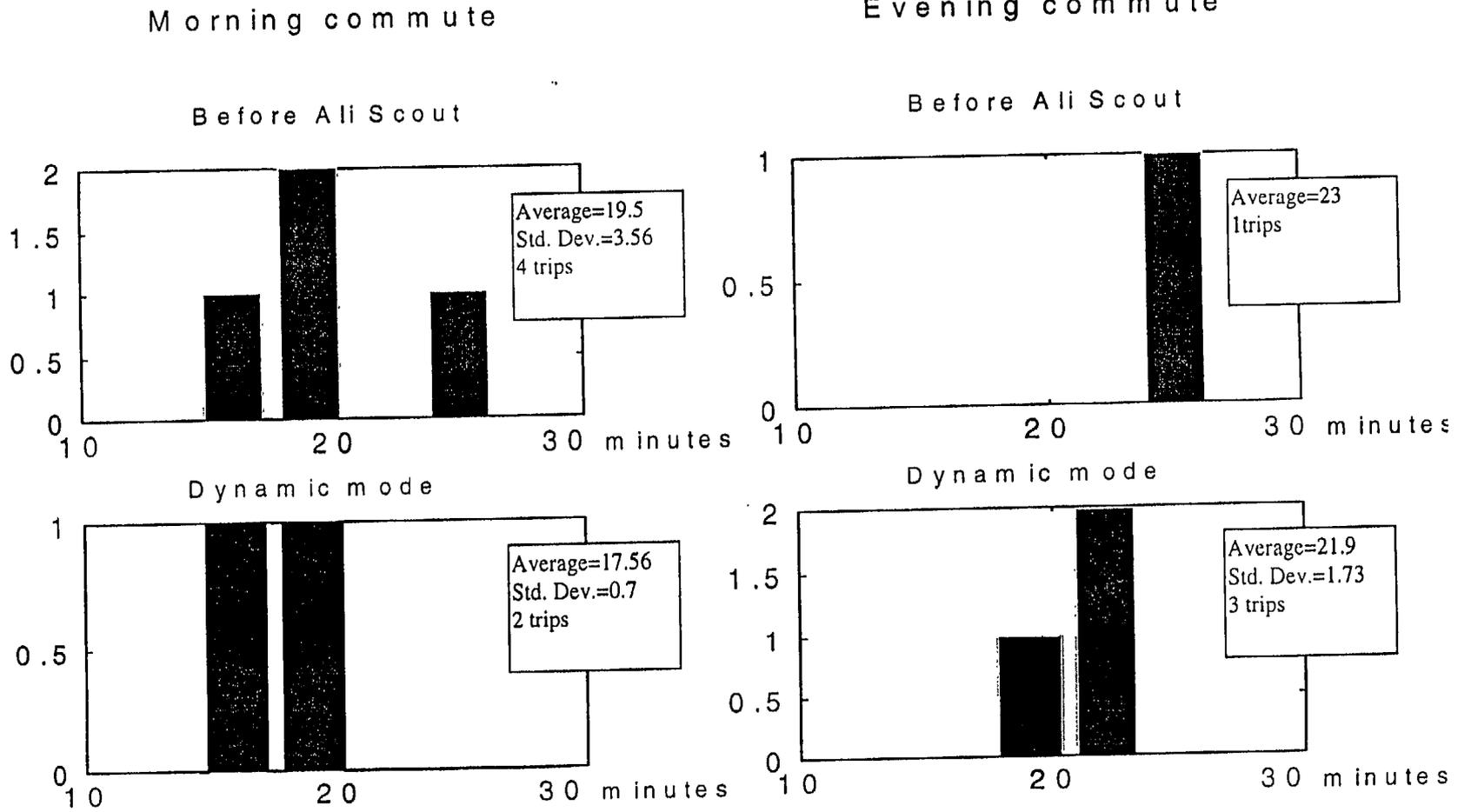


Figure 13: First-To-Last Beacon Trip Time Histograms for Commuter #3

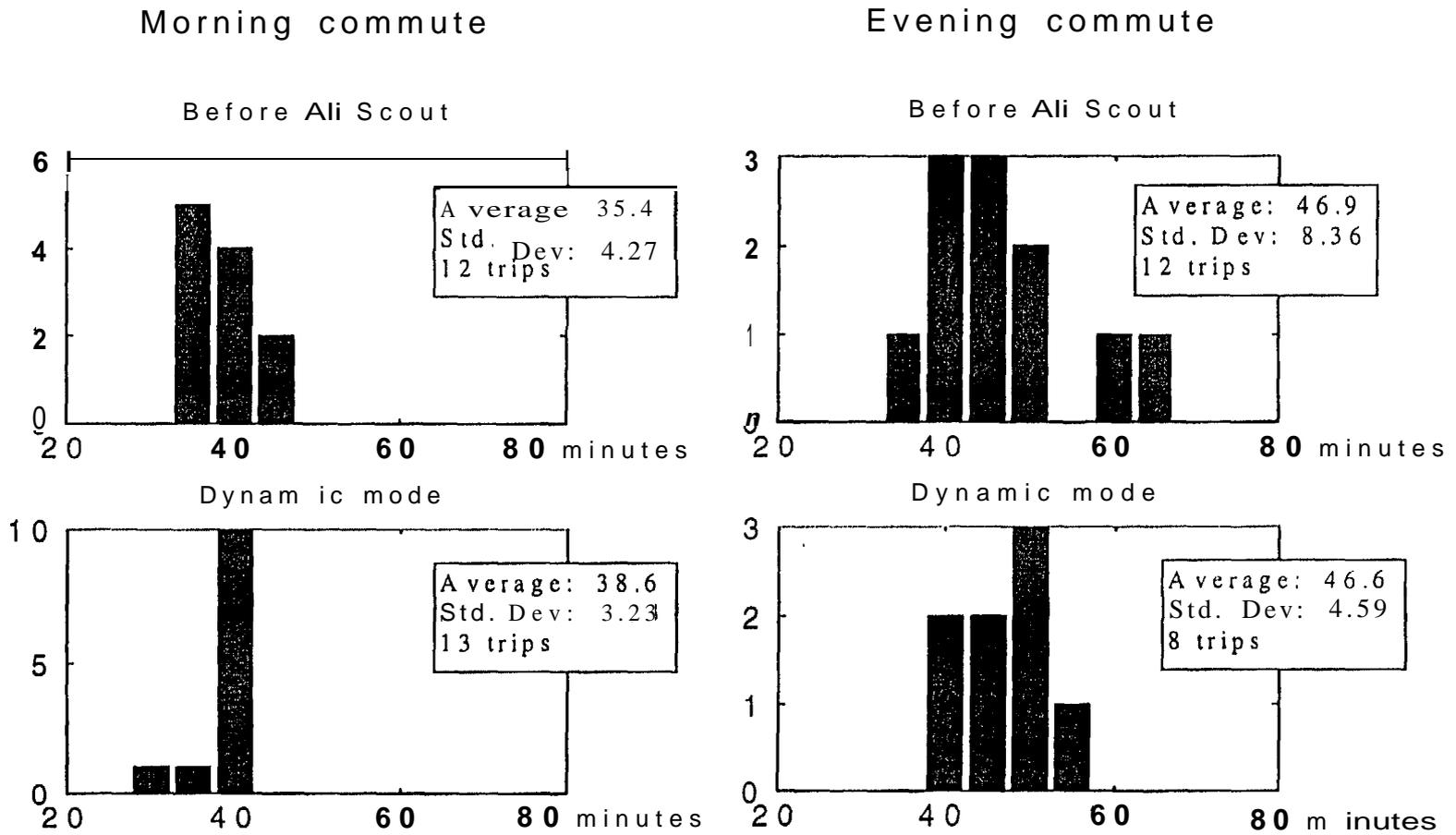


Figure 14: Total Trip Time Histograms for Corn m uter #4.

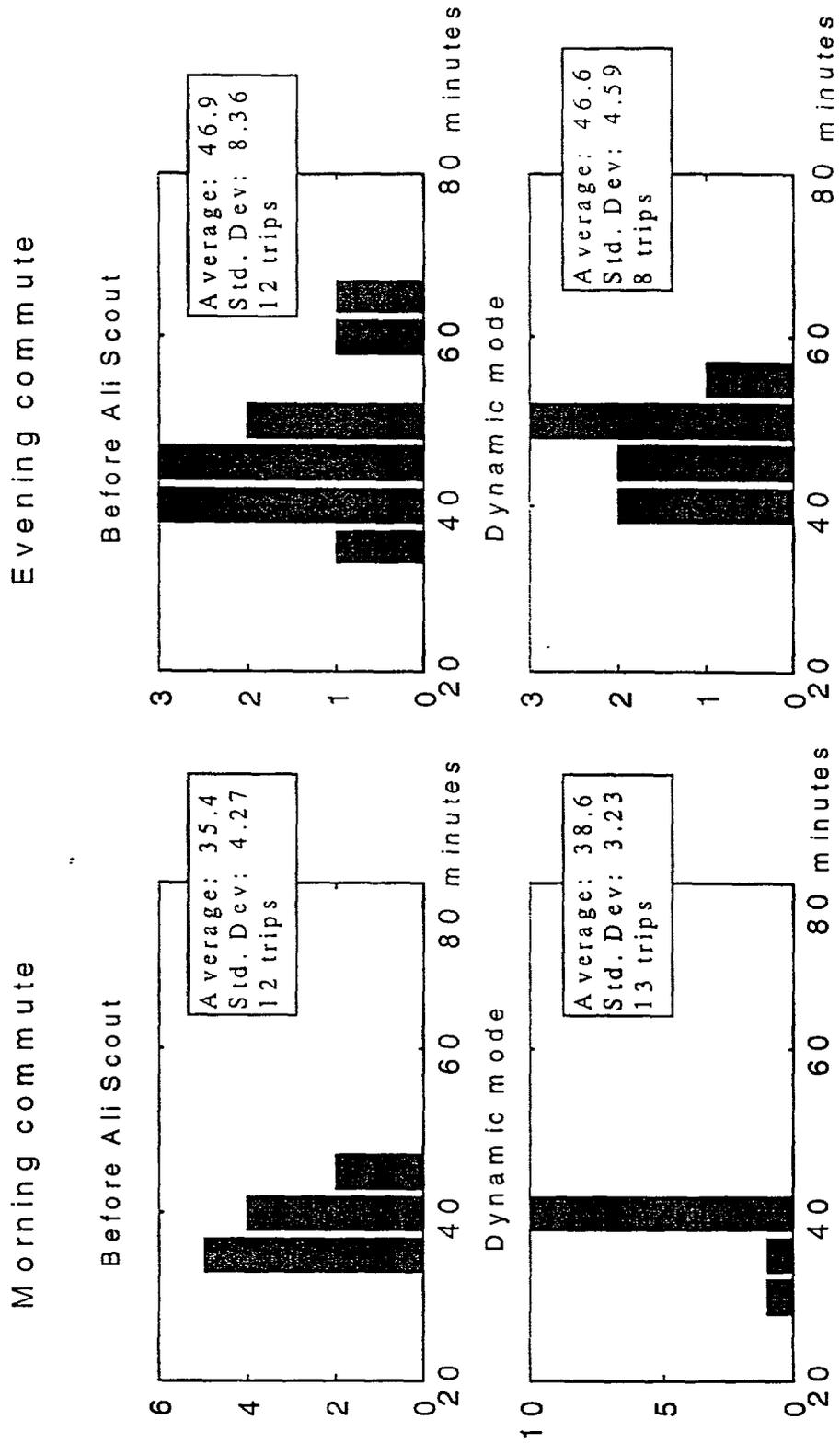


Figure 14: Total Trip Time Histograms for Commuter #4.

We conclude, then, that Ali-Scout's inability to show any consistent reduction in average commute-time is due to some combination of the following factors:

- 1) More restricted set of links, compared to expert commuter's set.
- 2) Inability to respond promptly to any non-recurrent congestion (from either incident or weather).
- 3) the presence of routing instruction abnormalities (reported by three of the five drivers).

Three of the five commuter drivers reported that they discovered at least one new fairly attractive commute route. Each of them expressed an expectation that the system would find its way around any unexpected congestion, and each of them expressed disappointment in not experiencing that. One of the

drivers complained that the "turn-warning-distance" was too short for the expressway maneuver, and we agree.

The final trip performance data that was collected concerned the speed profiles, which are shown in Figures 16 through 19. The key relating the bar intensity to the mode is shown on the right of each graph. In Figure 17 we see a slight increase in the "65 to 80" segment for both morning and evening, for commuter #2 in the static mode. Figure 19, shows a significant increase in the "above 80" segment in the morning commute. In addition, commuter #3 reported that "I find myself driving faster with Ali-Scout; I think it will take care of my turns, etc., so I am not retaining the need to be aware of where I am--I have to watch it, or I'll be ticketed". Outside of this possible issue, we have not found anything significant from the speed data,

Figure 16: Speed profile for Commuter #1 (Before, Static, Dynamic)

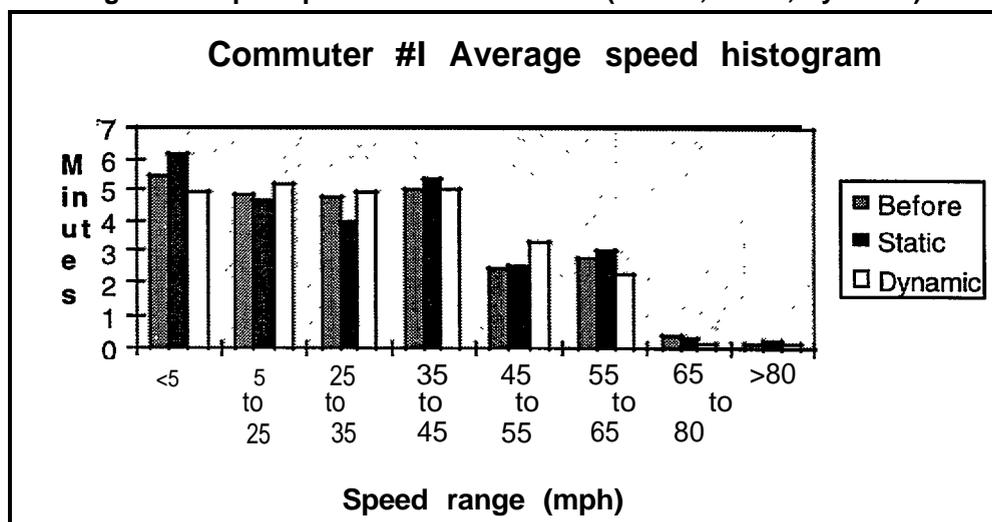


Figure 17: Speed Profiles for Commuter #2 (Before, Static, Dynamic)

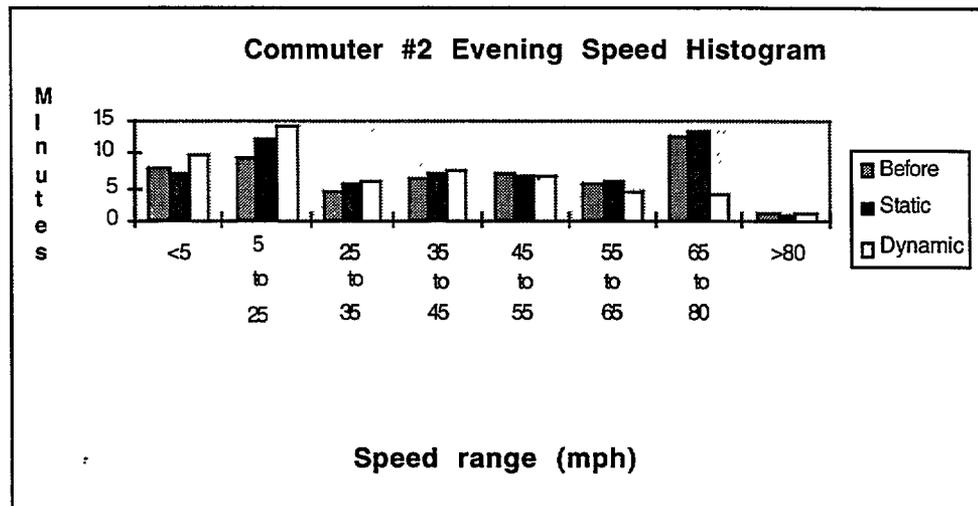
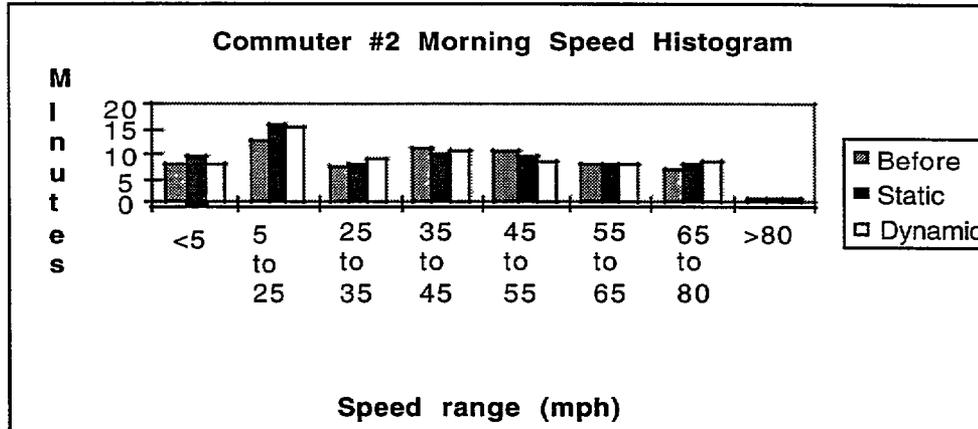


Figure 18: Speed Profile for Commuter #3

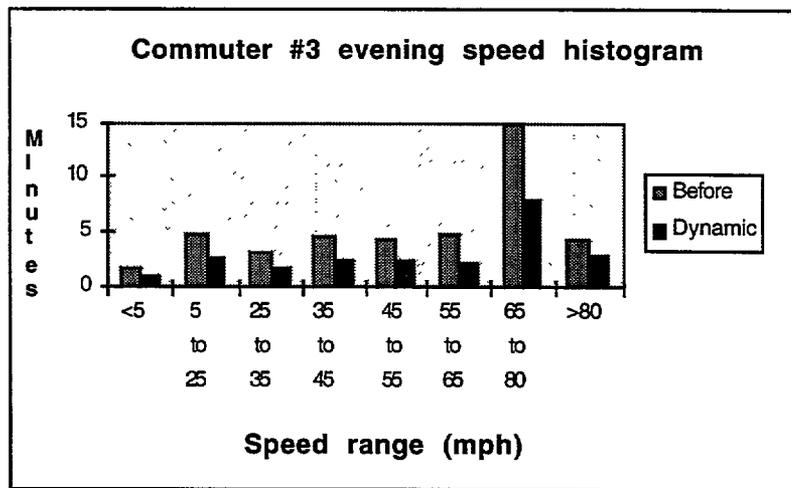
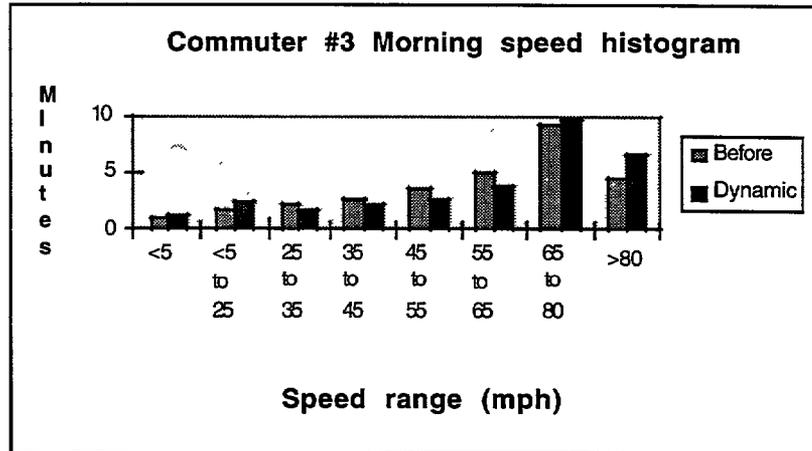
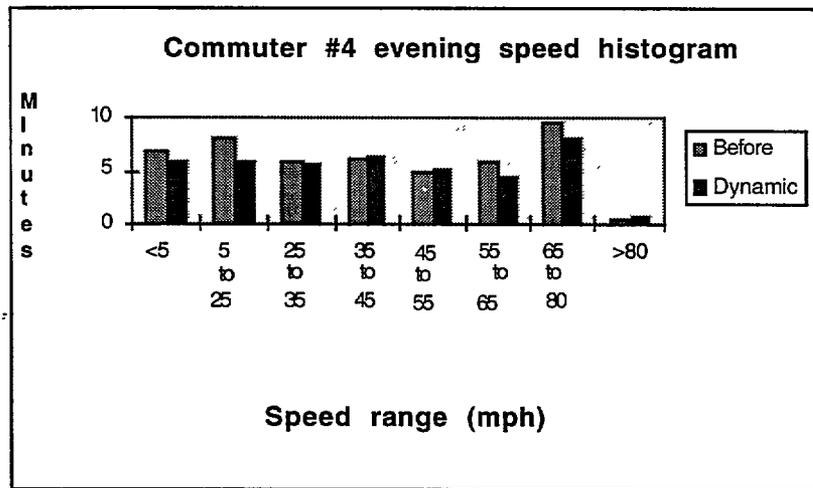
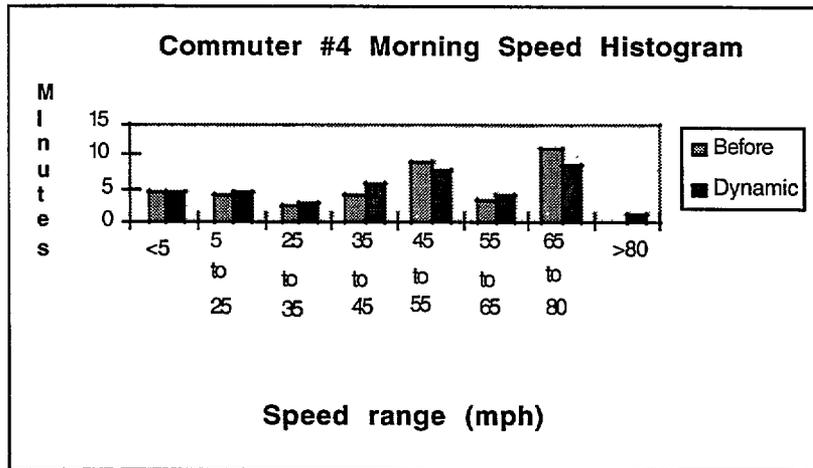


Figure 19: Speed Profiles for Commuter #4 (Dynamic)



9. TECHNICAL PERFORMANCE CONCLUSIONS

The overall concept of Ali-Scout was and is appealing since it promises to eliminate the need for additional surveillance infrastructure. Thus the relatively high cost of a beacon-based infrastructure for route guidance would be justified since it eliminates the high infrastructure cost of induction loops or overhead cameras and their connectivity to a TOC. However we have seen that the beacon-based link reporting introduces an inherent delay in responding to non-recurrent congestion, which limits the ability to provide full dynamic operation. Ali-Scout can provide dynamic response to historical recurrent congestion.

During this Operational Field Test, Ali-Scout was sufficiently complicated and ambitious that it challenged the operating personnel in trying to keep all parts in working order each and every day. The combination of field infrastructure required, and the exposure to rain, snow and thunderstorms, along with the in-vehicle equipment, resulted in a system challenging to build and maintain. We estimate that, at any one time, about 7% of the beacons were inoperative. For the operating beacons, the vast majority had a downlink success rate in the high nineties. A minority of beacons, usually on the expressways, exhibited a lower success rate. The uplink success rate of the operating beacons was much less; about 35% averaged over all sources. The system was designed to tolerate some downlink and uplink failures, so we estimate that the downlink behavior exhibited here adequately supported guidance at least 95% of the time. The lower uplink success rate, we estimate, lowered the probe report responsiveness, already weakened by a sparse number of equipped vehicles.

Considering input from all data sources, it appears that the Technical Performance during this test was sufficiently high so that the system performed as intended most of the time. However, all data sources that used the system over an extended period of time experienced malfunction episodes: not changing to

guided mode after passing first beacon; dropping out of guided mode while on recommended route; and missing (or awkward) steps in a maneuver instruction.

We estimate that the Technical Performance was barely adequate, and that it had a slight negative impact on any measured evaluation of the system. A route guidance device has to offer the same degree of reliability as other automotive electronics: radio, cruise control, anti-lock braking, and airbags (whose mean time between failures is at least three to five years).

The greatest limitation of Ali-Scout, confirmed by the commuter drivers, was the limited ability to provide what the driver's really wanted--a prompt response to the actual current conditions, especially the unpredictable non-recurrent incidents. Each of the commuter drivers commented on this, without prompting. Use of a human operator would have improved this aspect, but the system delay would still limit the responsiveness.

After considering all the data we conclude that, under the conditions of low number of probe vehicles, no operator at the console, and trying to serve drivers already expert in the road network, that this system did not deliver benefits commensurate with its cost. The system was turned off on Feb. 26, 1997.

We can make further observations about current route guidance systems in the U.S. As of late 1996 autonomous (in-vehicle-based) route guidance systems have been adopted by some major car rental companies. It is obvious that the benefit of a guidance system is maximized for drivers who are unfamiliar with the area. Such systems provide long advance notice of the "next maneuver" since the display shows the distance to next maneuver immediately after a maneuver completion. Although static in operation, the driver can implement a basic dynamic operation, especially if traffic broadcasts are utilized. If unexpected congestion is encountered, the driver may choose to leave an expressway (or street) route; soon thereafter a new route will be recommended from the diverted route; in this way, using broadcast information, the driver can arrange to be guided around non-recurrent congestion.

REFERENCES

1. L. P. Kostiniuk, D. W. Eby, C. Christoff, M. L. Hopp, F. M. Streff, "The Fast-Trac Natural Use Leased-Car Study: An Evaluation of User Perceptions and Behaviors of Ali-Scout by Age and Gender", EECS-ITS Lab-FT96-086 Part 1, The University of Michigan, 1996.
2. D. W. Eby, L. P. Kostiniuk, F. M. Streff, M.L. Hopp, "Evaluation the Perceptions and Behaviors of Ali-Scout Users in a Naturalistic Setting", EECS-ITS Lab-FT96-086 Part 2, The University of Michigan, 1996.
3. D. W. Eby, L. P. Kostyniuk, C. Christoff, M. Hopp, F. M. Streff, "An On-The-Road Comparison of In-Vehicle Navigation Assistance Systems: The FAST-TRAC Troika Study", EECS-ITS Lab-FT96-086, The University of Michigan, 1996.
4. A. Steinfeld, D. Manes, I? Green, D. Hunter, "Destination Entry and Retrieval with the Ali-Scout Navigation System", EECS-ITS-LAB-FT96-086, The University of Michigan, 1996.
5. S. Katz, J. Flemming, P. Green, D. Hunter, D. Damouth, "Initial On The Road Experiment/Calibration", EECS-ITS-LAB-FT97-087, The University of Michigan, 1997.
6. A. Hadj-Alouane, N. Hadj-Alouane, O. Juma, A. Okin, "Ah-Scout Simulation Runs at Various Levels of Market Penetration", EECS-ITS-LAB-FI'96-119, The University of Michigan, 1996.
7. Von Tomkewitsch, "Dynamic Route Guidance and Interactive Transport Management with Ali-Scout," *IEEE Transactions on Vehicular Technology*, vol. 40, p. 45, February 199 1.
8. Siemens, *Ali-Scout Vehicle Navigation System Users Guide*, FAST-TRAC, IVHS Operational Field Test, Oakland County, MI.
9. Hiroshi Kondo, "The Public Demonstration of VICS" *Proceedings of the IVHS AMERICA Annual meeting*, pp. 382-390, Atlanta, Georgia, April 1994

APPENDICES

Note: These appendices are exact excerpts from previously published documents. Page numbers have been left for reference purposes, however, the numbers are not necessarily sequential between appendices.

APPENDIX A

WC SOFTWARE

The VUC software originated by Siemens includes the ability to replay a vehicle's trip, in addition to storing the transactions between the vehicle and the beacons. For Technical Performance purposes we are only interested in the transactions performance, so are using only a part of the WC software.

The "EURO-SCOUT WC Manual" specifies an "IBM compatible PC" as hardware, and in addition requires use of a proprietary HSCX communications card between the In-Vehicle Unit (IVU) and the recording computer. During the trip the WC software creates a trip directory that contains several sets of files. Each set of files is generated sequentially along the trip, and contain trip-specific information. For the Technical Performance evaluation effort we extract data from the following set of files:

1. *hist0001.vuc* (History)
2. *travxxxx.vuc* (Travel)
3. *posixxxx.vuc* (Position)
4. *uplxxxx.vuc* (Uplink)
5. *beacxxx.vuc* (Downlink)

The name of each directory generated for every trip is *autoxxxx.rec*. The sub-field *xxxx* is numeric and increases from 0001 upwards. There are five kinds of files within each trip directory. All files have a filename followed by *.vuc* filename extension. Each filename has a filename sub-field *xxxx*. These sub-fields are generated sequentially starting from number 0001. Each file contains a set of different fields containing different kinds of information. ***The *travxxxx.vuc****, and ***posixxxx.vuc***. files fields are initiated by distinct header byte values that indicate the field type. Each field has a specific length enabling us to navigate through the file. The *hist0001.vuc*. fields are written in ASCII. The *uplxxxx.vuc* and ***beacxxx.vuc*** files contain one field with a complex structure. The specific field structure description for all the files can be found in "EURO-SCOUT VUC-Software, Layout of WC-Files."

Hist0001.vuc file

There is one HISTORY0001 .WC file per trip. It is generated at the termination of the trip when the F10 key is pressed. We extract the following information from the history file.

- Driver name
- Trip date
- Trip time
- Trip duration
- Number of blocks received by the computer from the IVU
- Number of blocks received by the IW from the computer
- Total number of uplink telegrams transmitted by the IW to the beacons during the trip

Travxxxx.vuc files

Each *trav* file within a trip directory has a varying number of fields. Some fields could repeat with different kind of information. The fields that are pertinent to the technical performance evaluation effort are:

- Vehicle data field

12 is the first byte value that identifies this field. The fourth byte contains vehicle sensor information. bit 7 in this byte when true indicates that the DCU is connected to the NAC.

- Message field

14 is the first byte value that identifies this field. The second byte value indicates one of the following events

0: Start of transmission of the infrared carrier

1: First valid IR block

2: Beacon data has completely been received

3: Vehicle uplink telegram to the beacon is sent

4: Beacon illumination zone passed

5: Uplink request by the IVU

6: Uplink acknowledged by the beacon

- Destination coordinates field

16 is the first byte value that identifies this field. This is followed by the destination longitude and latitude each occupying 4 bytes.

- Error message of the IVU

18 is the first byte value that identifies this field. When that field appears at in the file, it indicates that some error has occurred in the IVU.

- Destination coordinates global

25 is the first byte value that identifies this field. The second byte contains the length of the field, the next four bytes contain the destination longitude, and the second four bytes contain the destination - latitude.

posixxxx.vuc files

Each *posixxxx.vuc* file within a trip directory has one field category. New fields are generated to contain the value of the current longitude and latitude at 112 seconds intervals. Normally the longitude and latitude information are contained in the second four bytes and the third four bytes of the field. The next byte indicates whether the vehicle is in autonomous, tracking, or guided mode. However, if the vehicle stays in the same position more than a single 1/2 second, then the second 4 bytes will contain FFFFFFFF hex, and the third 4 bytes will contain the number of these 1/2 seconds.

Uplkxxx.vuc files

The 10 bits occurring after the 3rd byte contain a number that points to the location of data table 1 relative to the beginning of the file. Data table 1 contains general information about the vehicle and the journey. We look at the following three fields in data table 1.

- Vehicle identification

The vehicle identification field is a random number that is generated by the central office and is valid for one trip. If for some reason the driver changes the destination during the trip then

it will be assumed by the central office that a new trip is underway and a new random number is generated.

- Telegram number

This number is incremented by 1 each time an uplink telegram is transmitted to a beacon starting from 0. It is reset to zero each time a new destination is selected.

- Beacon

The specific beacon on a specific link where link statistics were transmitted.

Beacxxx.vuc files

- Beacon number

Beacon number passed is extracted from this file

DESCRIPTION OF UM'S SOFTWARE

In order to extract the Technical-Performance relevant data from WC files, software had to be written to extract and tabulate the results. The original UM software was written by Zihui Huang, which was then revised and extended by Ghassan Shahin.

The source code of the UM-software is shown in appendix B. It is written in Borland C++, and runs on any IBM compatible computer in the windows environment.

The application is a project file that combines the following different programs:

main.c

history.c

trav.c

position.c

uplk.c

beacon.c

In addition to the above programs there are four header files, namely:

main.h that is associated with ***main.c***, ***history.h*** that is associated with ***history.c***, ***trav.h*** that is associated with ***trav.c*** and ***position.h*** that is associated with ***position.c***

We will go through each of the above C programs and discuss its relevant features, the WC software fields it operates on, and the technical performance measures it generates.

Main. c

This program ties the project tile together, the main actions are:

1. It prompts the user to enter the trip directory whose files are to be operated on.
2. It allows the execution of the files ***history.c*** that analyzes the ***histxxx.vuc*** files, followed by ***trav.c*** that operates on ***travxxx.vuc*** files, followed by ***position.c*** that operates on ***posxxx.vuc*** files followed by ***uplink.c*** that operates on ***uplxxx.vuc*** files. The results of each of these operations are saved in one trip file called ***result***.
3. The result file is then displayed on the screen in slow manner to allow the user to read the results immediately.

Trav. c

This routine goes through all the ***travxxx.vuc*** files in the trip directory specified by the user in a sequential manner. On each of these files it performs the following:

1) If the current field is *vehicle data*, a check is performed on the most significant bit of that field (bit7). This bit affects three fields in the result file namely:

Starting display state

Ending display state

Number of display state changes

The rationale is as follows;

- During the first encounter of that bit, if that bit is zero then the *starting display state* is off otherwise it is on.
 - Each time that bit is encountered and is different than the previous time that it was encountered then, the *Number of display state changes* in the *result* file is incremented by 1
 - In the last encounter of that bit if it is zero then *Ending display state* is off otherwise it is on.
- 2) If the current field is *message* , then one of the following seven things would occur.
1. If the message type byte is 0 then the *Number of start IR* field is incremented
 2. If the message type byte is 1 then the *Number of first IR* field is incremented
 3. If the message type byte is 2 then the *Number of completed downlinks* field is incremented
 4. If the message type byte is 3 then the *Number of completed up links* field is incremented
 5. If the message type byte is 4 then the *Number of coutoff lines* field is incremented
 6. If the message type byte is 5 then the *Number of uplink requests* field is incremented
 7. If the message type byte is 6 then the *Number of uplink acknowledges* field is incremented.

In addition to the above the *result* file contains a printout out of the time sequence of the fields first valid *IR*, down link. Also it time stamps each of these actions by the time of creation of the specific trav file containing them.

3) If the current field is *destination coordinates* then the *destination longitude* and *destination latitude* fields will be displayed and saved.

4) If the current field is *error message of ivu*, then the mere fact that that field is encountered means that some IVU malfunction has occurred. Each time that field is encountered the field *Number of in vehicle unit errors* is incremented. In normal operation this field is never encountered.

5) If the current field is *distination coordinates global* then the *destination longitude* and *destination latitude* fields will be displayed.

6) If any of the other fields specified in the VUC document are encountered, the program pointer ignores it and the pointer is incremented to point to the next field based on the length of the current field.

Position. c

This routine goes through all the *posixxxx.vuc* files in the trip directory specified by the user in a sequential manner. The information we get from this are:

a) Driving mode information, b) Trip timing information, c) Trip distance related information, d) Cumulative trip speed histogram.

Mode Information

There are 3 different mode related information that we acquire.

- 1) Mode switching sequence, this describes how the modes are dynamically changing in time. We expect to see two mode switches, autonomous to guided followed by guided to tracking. Also we put the duration of each mode in seconds
- 2) Last mode at which the trip ended.

3) The cumulative number of mode changes. A high number of mode changes implies a difficulty of the Ali-Scout system including the driver and the network to guide efficiently to the trip destination.

Trip timing information

Each time we receive field 33, we are prompted about the current mode, and the time since previous reception of this field. This is done by updating two variables called *current_mode* and *count*. From these two pieces of information we compute the following.

1) Trip time

Trip time by simply accumulating the time intervals of all the received fields in a software counter called *total time*.

2) Time in each mode

The cumulative time spent in each mode, namely the autonomous, tracking, and guided modes. We do that by keeping 3 counters called *time-in-auto*, *time_in_tracking*, and *time_in_guided*.. As a double check, the trip time is also the sum of the 3 counts accumulated at the trip end

3) Time until first beacon

Time until first beacon is extracted by first checking that the current mode in field 33 is autonomous, then we check that no other mode occurred since the start of the trip. This is guaranteed by keeping a variable called *mode_change* that is initialized to zero. So if the current mode is autonomous and the *mode_change* variable is zero, then we increment another variable called *time_before_first* by the value of the variable *count*. Once the *mode_change* variable is incremented, which would only occur if a switch to guided has occurred, then the *time_before_first* variable will stay the same reflecting that it is the cumulative time until first beacon. The shorter the time until first beacon in the coverage zone the better. Short applies in both relative and absolute terms.

4) Time between first and last beacon

Time between- first and last beacon is accomplished by keeping two variables, namely *time_before_first* and *time-last-mode-ided*. The *time before_first* saves the time until we get out of the autonomous mode, and *time last_mode_guided* saves the time until we leave the last guidedmode. Consequently the difference between these two quantities yields the time between the first and last beacons. However before we make the computation, we check that the quantity *time_last_mode_guided* is different than zero to guarantee that during the trip the guided mode has occurred.

5) Time between first and last beacon unguided

To compute the time between first and last beacon unguided, we subtract the *time_guided* quantity from the time between first and last beacon as computed above.

6) Time beyond last beacon

Note that the time should equal to the sum of items 4, 5, and 7.

Distance related information

1. Origin coordinates
2. Destination coordinates extracted from file trav0001.vuc, but used here.
3. First autonomous position beyond last beacon, if exists.
4. First tracking position beyond last beacon, if exists.
5. Euclidean distance between origin and destination
6. Euclidean distance between first autonomous position beyond last beacon and destination

7. Euclidean distance between first tracking position beyond last beacon (if exists) and destination
The formula to compute the Euclidean distance between two points a and b is:

$$Distance = \sqrt{(long_b - long_a)^2 - (lat_b - lat_a)^2}$$

Speed Histogram

There are 8 bins in the speed histogram, they correspond to 8 speed ranges. The value accumulated in each bin corresponds to the time the vehicle spent in that speed range. The bin ranges are shown in the table

binnumber	meters per second	meters per hour	miles per hour
bin 1:	0-2.2347222	0-8045	0-5
bin 2:	2.2347222- 11.173611	8045-40225	5-25
bin 3:	11.173611- 15.643056	40225-563 15	25-35
bin 4:	15.643056-20.1125	563 15-72405	35-45
bin 5:	20.1125-24.581944	72405-88495	45-55
bin 6:	24.58 1944- 29.05 1389	88495-104585	55-65
bin 7:	29.05 1389- 35.755556	104585- 128720	65-80
bin 8:	>35.755556	>128720	50

The formula to compute the speed between two geographical points a and b with distinct longitude and latitude and are separated by a certain number of 1/2 seconds is :

$$Speed = \sqrt{\frac{(longb - longa)^2 + (latb - lata)^2}{time\ interval}}$$

In the above formula the following conversion factors are used

To convert latitude from 10⁻⁷ degrees to meters we multiply by 0.0111178

To convert longitude from 10⁻⁷ degrees to meters we multiply by 0.00826213563

To convert from 1/2 seconds count to a time interval in seconds we multiply by 1/2

The result appears in meters per second; consequently we check the table to locate the range in mph to which the new speed belongs. Now, to generate the cumulative time spent at that speed range we add the time interval spent at the current speed to the bin value. Also, in order to avoid excessive computations and round off errors, we compute the speed when more than 2 seconds have elapsed since the last reported position.

Uplk.c

This program extracts three fields from the *uplkxxx.vuc* files as follows:

1. Vehicle identification which is an identifier that is generated by the central office each time a new destination is chosen by the driver.
2. Telegram number that is incremented by one each time an uplink telegram is transmitted. It is reset to zero every time a new destination is chosen.
3. Beacon number, which is a unique name for each beacon in the coverage area. Each beacon number is time stamped with the time of generation of its *uplkxxx.vuc* file.

Beacon.c

This program extracts the beacon ID of the passed by beacon and the time of the generation of the respective *beacxxx.vuc* file.

APPENDIX B

**Zhihui Huang
Ghassan Shahin**

**UM's PROGRAM TO COLLECT
TECHNICAL PERFORMANCE MEASURES**

/* File: main.c

Purpose: The driver program that is part of the Fast-Trac technical data analysis package.

Author	Date	Rev	Original
Zihui Huang	08/28/94	001	
Zihui Huang	10/03/94	002	
Ghassan E. Shahin	07/18/95	003	Display of Result File
Ghassan E. Shahin	07/19/95	004	Inclusion of vuc files directory

Electrical Engineering and Computer Science Department
The University of Michigan

Copyright: © University of Michigan 1994/95

This program is from an old program in the “old” folder.
Changed to confirm to the database setup of the overall
Project Aug 22, 1994 */

```
#include<dos.h>
#include<stdio.h>
#include<stdlib.h>
char Directory[50];
long dest_long, dest_lat;
int destination;
int uplinks;
void main()
{
    long int a;
        FILE *fp
        void history(FILE *rfp);
        void position(FILE *rfp);
        void trav(FILE *rfp);
        void uplink(FILE *rfp);
        void beacon(FILE *rfp);
        printf(“Enter Directory name for the vuc files to be analyzed\n”);
        scanf(“%s” Directory);
        if ((fp=fopen(“result”,“w”))==NULL) {
            printf(“Error opening a file to write the result to.\n”);
            exit(1);
        }
        history(fp);
        trav(fp);
        position(fp);
        uplink(fp);
        beacon(fp);
        fclose(fp);
        fp=fopen(“result”,“rb”);
        if (fp=0)
        {printf(“An error occurred while opening the file.\n”);
            exit(1);
        }
        /*while (!foef(fp))
```

```
{ a=fgetc(fp);
  printf("%c",a);
  for (a=1 ;a<20;a++);} */
fclose(fp);
}
```

/* File: **history.c**

Purpose: This part of the FastTrac technical data analysis package. It deals with the HIST file of the VUC collected data file set.

Author	Date	Rev	Comments
Zihui Huang	08/28/94	001	Original
Zihui Huang	10/03/94	002	Modified original design
Ghassan E. Shahin	10/13/95	003	Export total uplinks

Electrical Engineering and Computer Science Department
The University of Michigan

Copyright: (c) University of Michigan 1994/1995

```
*/
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include "c:\borlandc\vuc\history.h"
extern char Directory[50];
extern int uplinks;
char histname[50];
struct trip-info this-trip;

void cleanup(char *cp)
{
    while ((*cp!='\\') && (*cp!='\0')) cp++;
    *cp='\0';
}

void history(FILE *rfp)
{
    FILE *fp;
    char buffer[200], *occur,*cp;
    /*int numread;*/

    /* In a trip directory, the history file is always named HIST0001 .VUC */

    strcpy(histname,Directory);
    strcat(histname,"\\hist0001 .vuc");
    if ((fp=fopen(histname, "r"))=NULL) {
        printf("Error opening file %s\n",histname);
        exit( 1);
    }

    /*numread=*/fread(buffer, sizeof(char), 200, fp);
    cp=buffer;

    /* search for the driver name in the file */
    occur=strstr(cp, "Driver: ");

    occur=occur+sizeof("Driver: ");

    /* put the name of the driver to this-trip record.
```

```

/*The driver name in this case can be a proper name
/* or a subject ID. */

sscanf(occur, "%s", this-trip.drivername);
cp=occur;

/* delete leading white spaces */
cleanup(this_trip.driver_name);
fprintf(rfp, "Driver is: %5s\n", this_trip.driver_name);
sprintf(subjectid, "%5s", this_trip.driver_name);

/* get vehicle ID number */
occurI=strstr(cp, "Car:");
occur=occur+sizeof("Car:");
sscanf(occur, "%s", this_trip.car_name);
cp=occur;
cleanup(this_trip.car_name);

/* This dummy prints the trip number */
fprintf(rfp, "Yoked\n");
sprintf(tripnumber, "Yoked");

/* get trip date */
occur=strstr(cp, "Date: ");
occur=occur+sizeof("Date: ");
sscanf(occur, "%s", this_trip.trip_date);
cp=occur;
cleanup(this_trip.trip_date);
fprintf(rfp, "Trip date is: %s ", this_trip.trip_date);

/* get trip start time */
occur=strstr(cp, "Time: ");
occur=occur+sizeof("Time: ");
sscanf(occur, "%s" this_trip.trip_start_time);
cp=occur;
cleanup(this_trip.trip_start_time);
fprintf(rfp, "Trip start time is: %s\n", this_trip.trip_start_time);

/* move to the end of the file. */
fseek(fp, -200L, SEEK-END);
/*numread=*/fread(buffer, sizeof(char), 200, fp);
cp=buffer;

/* get trip duration */
occur=strstr(cp "Current test time:");
occur=occur+sizeof("Current test time:");
sscanf(occur, "%f , &this_trip.trip_duration);
cp=occur;
fprintf(rfp, "Trip duration is: %.2f seconds\n",
this_trip.trip_duration);

/* get number of received blocks */
occur=strstr(cp "received blocks:");

```

```

occur=occur+sizeof("received blocks:");
sscanf(occur, "%f", &this_trip.total_down_bk);
cp=occur;
fprintf(rfp, "Received : %.f blocks. ",this_trip.total_down_bk);

/* get number of repeated received blocks */
occur=strstr(cp, "repeated:");
occur=occur+sizeof("repeated:");
sscanf(occur, "%d", &this_trip.total_down_bk_rep);
cp=occur;
fprintf(rfp, "Repeated: %d blocks\n",this_trip.total_down_bk_rep);

/* get number of transmitted blocks */
occur=strstr(cp, "transmitted blocks:");
occur=occur+sizeof("transmitted blocks:");
sscanf(occur, "%d", &this_trip.total_up_bk);
cp=occur;
fprintf(rfp, "Transmitted: %d blocks. ",this_trip.total_up_bk);

/* get number of repeated transmitted blocks */
occur=strstr(cp, "repeated:");
occur=occur+sizeof("repeated:");
sscanf(occur, "%d", &this_trip.total_up_bk_rep);
cp=occur;
fprintf(rfp, "Repeated: %d blocks \n",this_trip.total_up_bk_rep);

/* get number of uplink telegrams */
occur=strstr(cp, "telegrams:");
occur=occur+sizeof("telegrams:");
sscanf(occur, "%d", &this_trip.total_telegram);
cp=occur;
fprintf(rfp, "Uplink telegrams: %d blocks \n",this_trip.total_telegram);
uplinks=this_trip.total_telegram;
fclose(frp);
}

```

/* File: **trav.c**

Purpose: This is part of the FastTrac technical data analysis package. It deals with the TRAV files collected by the VUC program of Siemens.

Author	Date	Rev	Comments
Zihui Huang	08/28/94	001	Original
Zihui Huang	10/03/94	002	Modified original design
Ghassan E. Shahin	02/01/95	003	Conversion from think C to Borland C++
Ghassan E. Shahin	03/12/94	004	Modifying the output tile and adding text
Ghassan E. Shahin	07/19/95	005	Addition of Directory for <i>travxxx.vuc</i> files
Ghassan E. Shahin	08/29/95	006	Extracting the destination coordinates
Ghassan E. Shahin	08/29/95	007	Removal of uplink request and acknowledge Messages as they were not implemented by Siemens
Ghassan E. Shahin	10/13/95	008	Add the measures of failed uplinks and failed downlinks
Ghassan E. Shahin	05/02/96	009	Add the Start-LR message
Ghassan E. Shahin	09/25/96	010	Add the time that the Beacons are active

Electrical Engineering and Computer Science Department
The University of Michigan

Copyright: (c) University of Michigan 1994/1995/1996

*/

```
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include <io.h>
#include "c:\borlandc\vuc\trav.h"
```

```
void trav(FILE *rfp)
{
    enum panel-mode initial-mode, last-mode=PANEL-OFF, current_mode;
    int num_mode_change=0;

    struct trav_ret *head;

    int file_count;
    char file_name[13], part[5];
    FILE *fp;
    struct ftime ft,
    extem int uplinks;
    extem long dest_long, dest_lat;
    extem int destination;
    char mini_bum4];
    unsigned char *ptr;
    unsigned char c;
    unsigned_char_vehicle-sensor;
```

```

unsigned char current_ret_type;
unsigned char buffer[256]; /* big enough to hold the largest record type*/
unsigned char jump_count;
int i;
extern char Directory[50];
char travname[50];
/* form all possible file names and check each one out */
file_count=0;
destination=0
fprintf(rfp,“\n_____Messages Sequence _____\n”);
do {
    strcpy(file_name, “\\TRAV”);
    strcpy(travname,Directory);
    file_count++;
    sprintf(part, “%04d”, file_count);
    strcat(travname,file_name);
    strcat(travname,_ part);
    strcat(travname, “. WC”);
    if ((fp=fopen(travname, “rb”))=NULL) break,
    /* read vehicle data records from a file */
    printf(“reading file %s ..\n”. travname);
    while(!kbhit());getch();
    while (!feof(fp)) {
        c=fgetc(fp);

        /* process each record type separately. see the Siemens
        documentation for details of the records.
        */
        switch ((enum record-type) c) {
            case VEHICLE_DATA:

                c=fgetc(fp); c=fgetc(fp); /* jump over the wheel counter*/
                vehicle_sensor=fgetc(fp);
                current_mode = vehicle_sensor & 0x80;
                if (num_mode_change=0) initial_mode=current_mode;
                if (last-mode != current-mode) {
                    num-mode-change++;
                    last_mode = current_mode;
                }

                c=fgetc(fp); /* get number of mfs pairs */
                jump_count=c*4; /* four bytes for each pair */
                break;
            case STARTING-PARA:
                jump_coun+STARTING_PARA_SZ_1;
                break;
            caseMESSAGE:
                printf(“message %d “,c);
                c=fgetc(fp); /* get the actual message type */
                printf(“type %d \n”,c);
                getftime (fileno(fp),&ft);
                switch ((enum message_type) c) {
                    case START OF IR_CARRIER:
                        num_start_ir++

```

```

ft.ft_tsec*2);
        fprintf(rfp, "\n Start_IR  @%u:%u      ", ft.ft_min,
        break;
        case FIRST_VALID_IR_BLOCK:
            num_first ir++;
            fprintf(rfp, "1st_valid_IR  @%u:%u      ", ft.ft_min,
ft.fi_tsec*2);
            break;
            case BEACON_DATA_COMPL_RECEIVED:
                fprintf(rfp, "Down_link  @%u:%u      ", ft.ft_min,
ft.ft_tsec*2);
                num_down_complete++;
                break;
                case VEHICLE_TELEGRAM_SENT:
                    fprintf(rfp, "Uplink      @%u:%u\n",      ft.ft_min,
ft.ft_tsec*2);
                    num_up_complete++;
                    break;
                    case CUT_OFF_LINE:
                        fprintf(rfp, "Cut_off      @%u:%u\n",      ft.fi_min,
ft.ft_tsec*2);
                        num_cut_off_line++;
                        break;
                    }

                    jump_count=MESSAGE_SZ- 1-1;
                    break;
                    case LABEL:
                        c=fgetc(fp);
                        jump_count=c;
                        break;
                    /*
                    case DEST_COORD:
                    destination=destination+ 1;
                    printf("\n      DESTINATION_COORDINATES      FIELD
% d\n",c);
                    fread(buffer, sizeof(unsigned char),8, fp);
                    ptr=buffer;
                    for(i=0;i<=3;i++) mini buf[i]=*ptr++;
                    dest_long=*(long *)mini_buff;
                    for(i=0;i<=3;i++) mini_buff[i]=*ptr++;
                    dest_lat=*(long *)mini_buff;
                    jump_count=DEST_COORD_SZ- 1-8;
                    printf("Destination Longitude.. %ld
Destination Latitude %ld\n\n",dest_long,dest_lat);
                    /*
                    break;
                    case DEST_INPUT:
                    printf("\n DESTINATION-COORDINATES      INPUT      DCU
% d\n",c);
                    while(! kbhit());getch();
                    jump_count=DEST_INPUT_SZ- 1;
                    break;

```

```

        case ERROR_IVU:
printf("error ivu /n",c);
num_error ivu++;
        - jump_count=ERROR_IVU_SZ- 1;
        break;
        case VEHICLE_ATTR:
        c=fgetc( fp);
        jump_count=c*2;          /* each attribute takes two
bytes */
        break;
        case NUMBER_EXIT:
        jump_count=NUMBER_EXIT_SZ- 1;
        break;
        case GUIDANCE_INFO:
        jump_count=GUIDANCE_INFqSZ- 1;
        break;
        case NUMBER_LINK:
        jump_count=NUMBER_LINK_SZ-1;
        break;
        case NUMBER_DEST:
        jump_count=NUMBER_DEST_SZ- 1;
        break;

/*case DEST_COORD_GLOBAL:
destination=destinationq 1;
c=fgetc(fp);
printf("the character c is: %d\n",c);
fread(buffer, sizeof(unsigned _char),8, fp);
ptr=buffer;
for(i=0;i<=3;i++) mini buff[i]=*ptr++;
dest long=*(long *)mini buff;
for(i=0;i<=3;i++) mini_buff[i]=*ptr++;
dest-lat=*(long *)mini_buff;

if(c=1 1) {jump_coun=DEST_COORD_GLOBAL_ONE_SZ-
1-1-8;}

if(c==84){jump_count=DEST_COORD_GLOBAL_TWO_SZ-
1-1-8;}

        break;          */

        case DEST_COORD_OBJECT:
destination=destination+ 1;
c=fgetc(fp);
fgetc(fp);
fread(buffer, sizeof(unsigned char),8, fp);
ptr=buffer;
for(i=0;i<=3;i++) mini buff[i]=*ptr++;
dest long=*(long *)mini_buff
for(i=0;i<=3;i+t) mini_buff[i]=*ptr-+;
dest_lat=*(long *)mini_buff;

        jump_count=c-1-1-1-8;

```

```

        printf("\n\t\t\t DESTINATION COORDINATES\n\n");
        printf("Destination Longitude.. %Id Destination Latitude
%ld\n\n",dest_long,dest_lat);
        break;
    }
    fread(buffer, sizeof(unsigned char), jump-count, fp);
}
fclose(fp);
}while (1);
fprintf(rfp, "\n_____Display State_____ \n");
switch (initial-mode) {
    case PANEL-OFF:
        fprintf(rfp, "Starting display state is OFF\n"); break;
    case PANEL-ON:
        fprintf(rfp, "Starting display state is ON\n"); break;
}
switch (current_mode) {
    case PANEL-OFF:
        fprintf(rfp, "Ending display state is OFF\n"); break;
    case PANEL-ON:
        fprintf(rfp, "Ending display state is ON\n"); break;
}
fprintf(rfp, Number of display state changes is %d\n", num_mode_change);
fprintf(rfp, Number of in vehicle unit errors: %d\n", num-error-ivu);

/* experiment output */
    fprintf(rfp, "\n_____Communication Information_____ in");
fprintf(rfp, "Number of start IR: %d\n", num_start ir);
fprintf(rfp, "Number of first IR: %d\n", num-first--ir);
fprintf(rfp, "Number of completed down links: %d\n", num-down-complete);
fprintf(rfp, "Number of completed up links: %d\n", num-up-complete);
fprintf(rfp, "Number of cutoff lines: %d\n", num-cut-off-line);
fprintf(rfp, Number of failed downlinks: %d\n"
num cut off line-num down complete);
/* !!!!! fprintf(rfp, ";Number of faileduplinks: %d\n"
uplinks-num-up-complete); !!!!!*/
/* output of updownlink attemp
fprintf(rfp, "%d\n", num-start-ir-num-down-complete);
fprintf(rfp, "%d\n", num-up-req-num-up-complete);
*/
}

```

/* File: **position.c**

Purpose: This is part of the FastTrac technical data analysis package. It deals with the POSI file of VUC collected data file set to provide a speed histogram of the trip. For more information with regard to the structure of the posixxxx.vuc files, see the Siemens provides documentation.

Author	Date	Rev	Remarks
Zhihui Huang	08/28/94	001	Original Design of the program.
Zhihui Huang	10/03/94	002	Modified original program design.
Ghassan E. Shahin	02/01/95	003	Conversion from Think C to Borland C-N.
Ghassan E. Shahin	03/12/95	004	Modifying the output file and adding text to the output values.
Ghassan E. Shahin	04/19/95	005	Addition of Directory forposixxxx.vuc files.
Ghassan E. Shahin	05/08/95	006	Inclusion of the Mode switching sequence in output .
Ghassan E. Shahin	08/31/95	007	Capturing the origin coordinates.
Ghassan E. Shahin	09/01/95	008	Saving the coordinates at the instant of last switch to autonomous mode, and tracking modes.
Ghassan E. Shahin	09/06/95	009	Computing either the autonomous,or tracking mode Euclidean distance to destination beyond last beacon.
Ghassan E. Shahin	09/10/95	010	Computing the Euclidean distance from origin to destination.
Ghassan E. Shahin	09/27/95	011	Make the computation of time between beacons exact, add the measure of time beyond last beacon.
Ghassan E. Shahin	10/10/95	012	Add the initial mode, followed by other modes in full names.
Ghassan E. Shahin	10/13/95	013	Prompt when destination changes during the trip.
Ghassan E. Shahin	10/27/95	014	Modifying the program when current-mode and last-mode are different to take care of the spurious nomode mode.
Ghassan E. Shihin	01/06/95	015	Incorporating the last position as alternative to true destination, if destination coordinates were not specified by Ali-Scout.
Ghassan E. Shahin	07/02/96	016	Computing the sequential time intervalbetween modes .

Electrical Engineering and Computer Science Department
The University of Michigan

Copyright: (c) University of Michigan 1994/ 1995/1996*/

```
#include <stdio.h>
#include <math.h>
#include <string.h>
#include <stdlib.h>
#include "c:\borlandc\vuc\position.h"
```

```
/* lat, long info in the posixxxx.vuc files is "signed long". 5/22/94 */
```

```
/* speed intervals speed[8]:
   speed[0]: 0-5 mph or 0-8045 meters, stopper 2.2347222 m/s
   speed[1]: 5-25 mph or 8045-40225 meters, stopper 11-173611 m/s
```

```

speed[2]: 25-35 mph or 40225-56315 meters, stopper 15.643056 m/s
speed[3]: 35-45 mph or 56315-72405 meters, stopper 20.1125 m/s
speed[4]: 45-55 mph or 72405-88495 meters, stopper 24.581944 m/s
speed[5]: 55-65 mph or 88495-104585 meters, stopper 29.05 1389 m/s
speed[6]: 65-80 mph or 104585-128720 meters, stopper 35.755556 m/s
speed[7]: 80+ mph or 128720+ meters/second
*/

/* speed[] records the time the vehicle spends in each of the eight speed
intervals.
*/
double speed[8]=(0.0,0.0,0.0,0.0,0.0,0.0,0.0,0.0);
get-speed(long *last-lat,long *last-long,long *this-lat,long *this-long, long interval)
{
double current-speed;
static double last-speed=0.0;
/* this is used to filter out position-correction
* caused sudden jumps in calculated speeds
*/
/** this is used to show that the lat, long info read from the files are correct.
static int count=0;

if (count<=20) (
printf("this lat: %Id, this long: %Id\n", *this$lat, *this-long);
count*;
}
**/

/* when computing the speed, the surface of earth is treated as a plane */
if (*last-lat==0) /* first position */
return;
if (*this-long==0xffffffff) /* no change in position */
*this_long=*last_long;
*this_lag*last_lit;
}

/* actually computing */
current-speed = sqrt(pow((( *this long)-(*last-long))*UNIT-LONG, 2.0)+
pow((( *this-lat)-(*last-lat))*UNIT-LAT, 2.0))/(0.5*interval);

/* check of sudden jumps */
/* not done yet. need to figure out max. act and dec in speed for the cars */
last-speed = current-speed;

if (current-speed <= 2.2347222) speed[0] += 0.5*interval;
else if (current-speed <= 11.1736 11) speed[ 1] += 0.5 *interval;
else if (current-speed <= 15.643056) speed[2] += 0.5*interval;
else if (current-speed -+ 20.1125) speed[3] += 0.5*interval;
else if (current-speed <= 24.581944) speed[4] +=
0.5*interval;
else if (current-speed <= 29.05 1389) speed[5]
+= 0.5*interval;
else if (current-speed <= 35.755556)
speed[6] += 0.5*interval;
else speed[7] += 0.5*interval;

```

```

        /* update the "last" lat, long */
        *last lat = *this lat;
        *last-long = *th&-long;
    return; -
}

void position(FILE *I@)
i
    extern char Directory[50];
    char posname[50];
    extern char subjectid[6];
    extern char tripnumber[4];
    struct position-ret thisposition;

    enum mode last-mode=AUTONOMOUS, current-mode;
    int bearing;
    double time-before-first=0.0;
    int mode-changed=0, num-mode-change=0;
    int aut=0;
int track=0;
    double time-last-mode-change=0.0;
    double time between modes=0.0;
    double time_till_previous mode=0.0;
    double time_last mode_guided=0.0;
    double time in nomode=0.0
    double time>in_auto=0.0
    double time-in_tracking=0.0
    double time-in-guided=0.0;
    double total-time=0.0; /* in 0.5 seconds */
    unsigned long count;

    long origin-lat, origin-long;
    long last-aut-long, last-aut-lat;
    long last-track-long, last-track-lat;
extern long dest-long, dest-lat;
extern int destination;
    double dead-reckon;
    double track-dest;
    double trip-length;
    long last-la& last-long;
    int origin=0;

    long speed-interval=0;
    int file-count;
    char file-name[ 131, part[5];
    FILE *fp, *speedfp;
    unsigned char *ptr;
    unsigned char buffer[ 111;
    unsigned char minibuf[41;
    int i;

    /* There are usually more than one POSI file for a trip.
       Form all possible file names and check each one out.

```

```

*/
file-count=0;
fprintf(rfp,“\n_____Mode Sequencing Information _____\n”);
do {
    strcpy(file-name, “\POS1”);
    strcpy(posname,Directory);
    strcat(posname,file_name);
    file count++;
    spr&f(part, “%04d”, file_count);
    strcat(posname, part);
    strcat(posname, “.VUC”);
    if ((fp=fopen(posname, “rb”))==NULL) break;

    /* read position records from a file */
    printf(“reading file %s ..\n”. posname);

    while (!feof(fp)) {
        fread(buffer, 11, 1, fp);
        ptr = buffer;

        this_position.position_type= * (unsigned char *)ptr; ptr +=1 ;

        for (i=0; i<=3; i++) minibuf[i]=*ptr++;
        this_position.longitude= *(long *)minibuff,
        for (i=0; i<=3; i++) minibuf[i]=*ptr-++;
        this_position.latitude= *(long *)minibuff;

        if(!origin) { origin=l000;
            origin_lat = this_position.latitude;
            origin_long= this_position.longitude;
            last_aut_long=origin_long; /* In case all autonomous*/
            last aut_lat =origin_lat;
            printf(“\n\n\t\t\t THE ORIGIN COORDINATES\n Origin Longitude
%ld\t Origin Latitude%ld\n\n”,
            origin_long, origin_lat);
        }

        this_positionbearing *(unsigned char *) ptr; ptr +=I;
        this_position.mode-read= *(unsigned char *) ptr;

        /* account for the time */
        if (this_position.longitude=0xffffffff)
            count = this_position.latitude;
        else
            count = 1L;
        total_time += count;
        speed_interval += count;

        /* check the mode operated in */
        if (this_positionmode_read & 0x80) /* guidance state info*/
            switch (this_position.mode_read & 0x30) {
                case 0x00: /* autonomous */
                    time_in_auto += count;
            }
    }
}

```

```

+= count;

if (!mode_changed) time_before_first

current_mode=AUTONOMOUS;
break,
case 0x10: /* tracking */
time_in_tracking += count;
mode_changed= 1;
current_mode=TRACKING;
break,
case 0x20: /* guided mode */
time_in_guided += count;
mode_changed = 1;
current_mode=GUIDED;
break;

}

else { /* no guidance state info */
time_in_nomode += count;
/* current_mode=NOMODE;*/ /* APPARENTLY SIEMENS ASSUMES 11
AS GUIDED TOO */
time_in_guided += count;
current_mode=GUIDED;
}
if (current_mode != last_mode) {
num_mode_change++;

time_last_mode_change = total_time-count;
time_between_modes =total_time_time_tillprevious_mode;
time_till_previous_mode=total_time;
if(origin == 1000) {
origin= 1;
if (last-mode==0)
fprintf(rfp," Initial vehicle mode is AUTONOMOUS \n ");
else if(last mode=1)
fprintf(rfp," Initial vehicle mode is TRACKING \n");
else fprintf(rfp,"Initial vehicle mode is GUIDED \n");
}
fprintf(rfp,"Time till next mode %f \n" time_between_modes/2.0);

switch(current_mode){
case NOMODE:
printf(rfp,"NOMODE");
break;

case AUTONOMOUS:
fprintf(rfp," AUTONOMOUS \n ");
last_aut_long = this_position.longitude;
last_aut_lat = this_position.latitude;
aut=aut+ 1;
break;

case TRACKING:
fprintf(rfp, " TRACKING \n");
last_track_long = this_position.longitude;

```

```

        last_track_lat = this-position.latitude;
        track=track+ 1;
break;

    case GUIDED:
        fprintf(rfp, " GUIDED \n");
        break;
    } /* end of the switch statement */
if(last-mode == GUIDED)
    time_last_mode_guided = time_last_mode_change;

    last_mode = current_mode;

    /* compute speed */
    if (speed_interval >= SPEED-INTERVAL) {
        get_speed(&last_lat, &last_long, &this_position.latitude,
            &this_position.longitude, speed_interval);
        speed_interval=0;
    }

    }
fclose(fp);
}while (1);

time_between_modes =total_time_time_till_previous_mode;
fprintf(rfp,"Time till next mode %f \n", time_between_modes/2.0);

switch (last_mode) {
case NOMODE:
    fprintf(rfp,"\n Last vehicle mode is NOMODE\n"); break;
case GUIDED:
    fprintf(rfp,"\n Last vehicle mode is GUIDED\n"); break;
case AUTONOMOUS:
    fprintf(rfp,"\n Last vehicle mode is AUTONOMOUS\n"); break;
case TRACKING:
    fprintf(rfp,"\n Last vehicle mode is TRACKING/n"); break;
}

fprintf(rfp, "Total number of mode changes is %d \n", num_mode_change);

fprintf(rfp,"\n _____Destination      Information_____ \n");

if(destination=0){
    dest_long = this_position.longitude;
    dest_lat = this_position.latitude;
    destination=destination+ 1;
    fprintf(rfp,"Destination Not Specified\n");
    }

if(destination> 1)
    {fprintf(rfp,"Driver changed destination during the trip\n");}

```

```

fprintf(rfp,“Origin Longitude    = %Id Origin Latitude    = %Id\n”,
        origin_long,origin_lat);

fprintf(rfp,“Destination Longitude = %Id    Destination Latitude = %Id\n”,
        dest_long,dest_lat);

trip_length=sqrt(pow((origin_long-dest_long)*UNIT_LONG,2)+
                pow((origin_lat_dest_lat )*UNIT-LAT,2))/1609.3;

fprintf(rfp,“Euclidean distance between origin and destination = %f miles\n”,
        trip_length);

if(track!=0) {
    track_dest    =sqrt(pow((last_track_long-dest_long)*UNIT_LONG,2)+
                        pow((last_track lat-dest_lat )*UNIT_LAT,2))/1609.3;
    fprintf(rfp,“First pos. in last tracking”);
    fprintf(rfp,“(Longitude = %Id) (Latitude = %Id)\n”,last_track_long,last_track_lat);
    fprintf(rfp,“Euclidean distance between last tracking and destination = %f miles\n”,
            track_dest);
    }
    if(aut!=0) {
    dead_reckon=sqrt(pow((last_aut_long-dest-long)*UNIT_LONG,2)+
                    pow((last_aut lat-dest_lat )*UNIT_LAT,2))/1609.3;
    fprintf(rfp,“First pos. in last aut”);
    fprintf(rfp,“(Longitude = %Id) (Latitude = %Id)\n”,last_aut_long,last_aut lat);
    fprintf(rfp,“Euclidean distance between last autonomous and destination =%f miles\n”,
            dead_reckon);
    }

    fprintf(rfp,“\n_____Cumulative                               Timing
Information _____\n”);
    fprintf(rfp, “Total time: %.2f seconds \n”, total_time/(2.0));
    fprintf(rfp,“Total    time    in    AUTONOMOUS    mode    %.1f    seconds\n”,
time_in_auto/2.0);
    fprintf(rfp,“Total time in GUIDED mode    %. If seconds\n, time_in_guided/2.0);
    fprintf(rfp,“Total    time    in    TRACKING    mode    %.1f    seconds\n”,
time_in_tracking/2.0);

    if(time_last_mode_guided){

    fprintf(rfp, “Time until first beacon: %.2f seconds \n”,time_before_first/2.0);

    fprintf(rfp, “Time between first and last beacon: %.2f seconds \n”,
            (time_last_mode_guided - time_before_fust)/2.0);

    fprintf(rfp, “Time between first and last beacon unguided: %.2f seconds \n”,
            (time_last_mode_guided - time_before_first - time_in_guided)/2.0);

    fprintf(rfp, “Time beyond last beacon: %.2f seconds \n”,
            (total_time_time_last_mode_guided)/2.0);

```

```
    }  
    fprintf(rfp, "\n_____Speed      Histogram_____ \n");  
    fprintf(rfp, "0 - 5 mph: %.2f\n", speed[0]/60.0);  
    fprintf(rfp, "5 - 25 mph: %.2f\n", speed[1]/60.0);  
    fprintf(rfp, "25 - 35 mph: %.2f\n", speed[2]/60.0);  
    fprintf(rfp, "35 - 45 mph: %.2f\n", speed[3]/60.0);  
    fprintf(rfp, "45 - 55 mph: %.2f\n", speed[4]/60.0);  
    fprintf(rfp, "55 - 65 mph: %.2f\n", speed[5]/60.0);  
    fprintf(rfp, "65 - 80 mph: %.2f\n", speed[6]/60.0);  
    fprintf(rfp, " > 80 mph: %.2f\n", speed[7]/60.0);
```

/* File: **uplink.c**

Purpose: This is part of the FastTrac technical data analysis package. This file extracts parameters pertinent to the technical performance evaluation of the Ali Scout Route Guidance System. The parameters are derived as per the Siemens documentation: Data Specification RGI Uplink

Author	Date	Rev	Comments
Ghassan E. Shahin	08/09/1995	001	Original
Ghassan E. Shahin	08/24/1995	002	Extract relevant parameters
Ghassan E. Shahin	10/13/1995	003	Change beacon and vehicle numbers to hex for field length standadization and better readability
Ghassan E. Shahin	08/01/1996	004	Change beacon ID's to decimal,and add the LD parameter
Ghassan E. Shahin	09/19/1996	005	Add vehicle kind, index-1 beacon-ID's
Ghassan E. Shahin	09/19/1996	006	Time stamp uplink file generation

Electrical Engineering and Computer Science Department
The University of Michigan

Copyright: (c) University of Michigan 1995/1996

*/

```
#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <io.h>
void uplink(FILE *rfp)
{
    int file_count==0;
extern char Directory[50];
    char uplink_name[50];
    char fileIn&e[ 13],part[5];
    FILE *fp;
    struct ftime ft,
    unsigned char *ptr;
    unsigned char buffer[26];
    unsigned char vehicle[3];
    unsigned char tel_num;
    unsigned char beacon[2],LD;
    int ij;
        fprintf(rfp," _____ Uplink Data _____\n");
    do{
        strcpy (file_name,"\\UPLK");
        strcpy (upILk_name,Directory);
        strcat (uplink_name,file_name);
        file count++;
        sprintf(part,"%04d",file_count);
        strcat(uplink_name,part);
        strcat(uplink_name,".VUC");
    }
    if((fp=fopen(uplink_name,"rb"))=NULL) break;
    printf("reading file %s . . . \n", uplink_name);
    fread(buffer,26,1,fp);
```

```

ptr=buffer;
ptr=ptr+buffer[3];/* Location of the pointer to data table 1 */
ptr=ptr+3;      /* Location of the vehicle ID */

for (i=0;i<=2;i++) {vehicle[i]=*ptr++;}
vehicle[2]=vehicle[2]>>4;
j=(int)vehicle[2];
fprintf(rfp,“Veh_kind %d “,j);
tel num=*ptr;
j=(int)tel num;
getfiime(fileno(fp),&ft);
fprintf(rfp,“ Telegram# %d @%u:%u “,ft.ft_min,ft.ft_tsec*2);
ptr=ptr+6;
for (i=0;i<=1;i++) {beacon[i]=*ptr++;}
LD=beacon[1]&(0x08);
LD=LD>>3;
beacon[1]=beacon[1]&(0x07);
l*j=(int)(beacon[1]*256+beacon[0]);*/
j=(int)(beacon[0]);
fprintf(rfp,“Beac_ID %d\““,j);
fprintf(rfp,“Live_Data %d\““,LD);
pn=ptr+6;
for (i=0;i<=1;i++) {beacon[i]=*ptr++;}
LD=beacon[1]&(0x08);
LD=LD>>3;
beacon[1]=beacon[1]&(0x07);
l*j=(int)(beacon[1]*256+beacon[0]);*l
j=(int)(beacon[0]);
fprintf(rfp,“Beac-ID %d\““,j);
fprintf(rfp,“Live-Data %d\n“,LD);
fclose(fp);
}while( 1);

return( 0);
}

```

/* File: **beacon.c**

Purpose: This is part of the FastTrac technical data analysis package. This file extracts the beacon ID. The parameters are derived as per the Siemens documentation: Data Specification RGI Downlink

Author:	Date	Rev	Remarks
Ghassan E. Shahin	08/17/1996	001	Original
Ghassan E. Shahin	10/01/1996	002	Time stamp beacon file generation

Electrical Engineering and Computer Science Department
The University of Michigan

Copyright: (c) University of Michigan 1996

```
*/

#include <stdio.h>
#include <string.h>
#include <stdlib.h>
#include <io.h>
void beacon(FILE *rfp)
{
    int file-count=0;
extern char Directory[50];
    char beacon_name[50];
    char file_name[ 13],part[5];
    FILE *fp;
    struct ftme ft;
    unsigned char *ptr;
    int buffer[2];
    unsigned char beacon[2];
    int ij;
    fprintf(rfp,"_____Downlink Data_____\\n");
do {
    strcpy (file_name,"\\beac");
    strcpy (beacon_name,Directory);
    strcat (beacon_name,file_name);
    file count++;
    sprintf(part,"%04d",file_count);
    strcat(beacon_name,part);
    strcat(beacon_name,".VUC");

    if((fp=fopen(beacon name,"rb"))=NULL) break,
    printf("reading file %s ... \\n", beacon_name);
    fread(buffer,3,1 ,fp);
    ptr=buffer;

    for (i=0;i<=1;i++) {beacon[i]=*ptr++;}
    j=(int)(beacon[0]);
    i=(int)(beacon[ 1]);
    i=(beacon[ 1])&(0x7F);
    getftime(fileno(fp),&ft);
    fprintf(rfp,"Beacon_ID %d @%u:%u\\t"j ,ft.ft_min,ft.ft_tsec);
    fclose(fp);
```

```
}while( 1);  
return(0);  
}
```

```
/* File: history.h
Purpose: This is part of the FastTrac technical data analysis package. It deals with the file
format of the HIST file of VUC collected data file set.
```

```
Author: Zihui Huang, EECS/University of Michigan
Copyright: (c) University of Michigan 1994
```

```
Last Modified: 8/28/94
Last Modified: 10/3/94
```

```
*/
```

```
/* Files to search for data:
```

```
* HISTxxxx.VUC
* BEACxxxx.VUC
* BCSTxxxx.VUC
* TRAVxxxx.VUC
* POSIxxxx.VUX
*
```

```
* where "xxxx" is a four digit number representing the cardinal number of the
* trip among all trip records in the same directory
*
```

```
*/
```

```
/******
```

```
    Data structure for the HIST file
    *****/
```

```
struct tip-info {
    char driver_name[20];
    char car_name[20];
    char trip_date[8];           /* mm/dd/yy */
    char trip_start_time[8];    /* hh:mm:ss */
    float trip_duration;       /* in seconds */
    float total_down_bk;
    int total_down_bk_rep;      /* number of retransmissions */
    int total_up_bk;
    int total_up_bk_rep;       /* number of retransmissions */
    int total_telegram;        /* uplink telegrams */
};
```

```
char subjectid[ 10];
char tripnumber[6];
```

/* File: **trav.h**

Purpose: This is part of the FastTrac technical data analysis package. It defines data structures to be used in the trav.c file. The data structures are derived from the Siemens documentation: Layout of VUC-files.

Author	Date	Rev	Remarks
Zhihui Huang	08/28/1994	001	Original
Zhihui Huang	10/03/1994	002	Modified
Ghassan E. Shahin	08/02/1995	003	Added the Destination Global field
Ghassan E. Shahin	02/12/1996	004	Added the Destination Object field

Electrical Engineering and Computer Science Department
The University of Michigan

Copyright: (c) University of Michigan 1994/1995/1996

*/

```
enum panel_mode {PANEL-OFF=0x00, PANEL_ON=0x80};
```

/* The 12 data record types in the trav files.

* right now, we are only interested in VEHICLE_DATA, ERROR_IVU and MESSAGE.

*/

```
enum record-type
{VEHICLE_DATA= 12,STARTING_PARA,MESSAGE,LABEL,DEST_COORD,
                                DEST_INPUT,  ERROR_IVU,  VEHICLE_ATTR,
NUMBER_EXIT=2 1,
                                GUIDANCE_INFO,          NUMBER_LINK,
NUMBER_DEST,
                                DEST_COORD_GLOBAL,DEST_COORD_OBJECT};
```

/* special messages in the trav files */

```
enum message_type {START_OF_IR_CARRIER=0
                    FIRST_VALID_IR_BLOCK,
                    BEACON_DATA_COMPL_RECEIVED,
                    VEHICLE_TELEGRAM_SENT,
                    CUT_OFF_LINE,
                    UPLINK_REQUEST,
                    UPLINK_ACK};
```

/* record size of some of the above record types. The size of the rest of them

* is vehicle/version dependent.

*/

```
#define STARTING_PARA_SZ 36
#define MESSAGE_SZ 2
#define DEST_COORD_SZ 9
#define DEST_INPUT_SZ 11
#define ERROR_IVU_SZ 3
#define NUMBER_EXIT_SZ 2
#define GUIDANCE_INFO_SZ 3
#define NUMBER_LINK_SZ 4
#define NUMBER_DEST_SZ 3
#define DEST_COORD_GLOBAL_ONE_SZ 11
```

```
#define DEST_COORD_GLOBAL_TWO_SZ 84
```

```
struct trav_rec {  
    enum panel_mode current_mode;  
    double time_spent;  
    struct trav_rec *next;  
};
```

```
short num_start ir=0;  
short num_first_ir=0;  
short num_down_complete=0;  
short num_up_complete=0;  
short num_cut_off_line=0;  
short num_up_req=0;  
short num_up_ack=0;  
short num_error_ivu=0;
```

/* File: **position.h**

Purpose: This is part of the FastTrac technical data analysis package. It deals with the file format of the POSI file of VUC collected data file set to provide a speed histogram of the trip.

Author	Date	Rev	Remarks
Zihui Huang	08/28/94	001	Original
Zihui Huang	10/03/94	002	
Ghassan E. Shahin	07112195	003	Rearrange the modes enumerate values so: Autonomous is 0, Tracking is 1, Guided is 2

Electrical Engineering and Computer Science Department
The University of Michigan

Copyright: (c) University of Michigan 1994/1995

*/

/* data structure for position records */

enum mode {AUTONOMOUS, TRACKING, GUIDED,NOMODE};

```
struct position_rec {
    unsigned char position type;
    /* unsigned */ long longitude, latitude;
    unsigned char mode_read;
    unsigned char bearing;
};
```

#define SPEED_INTERVAL 4

#define UNIT_LAT 0.0111178

#define UNIT_LONG 0.008262 13 563 /* meters per 10⁻⁷ degree */ meters Per 10⁻⁷ degree,

UNIT_LAT*cos(42*pi/180) /*

/* extern enum mode a_g_mode;

extern int bearing;

extern double time before first;

extern double time_in auto;

extern double time_in_guided;

extern double total_time;*/

APPENDIX C

**UNATTENDED ENABLING CIRCUIT
AND CUSTOM MADE COMPUTERS**

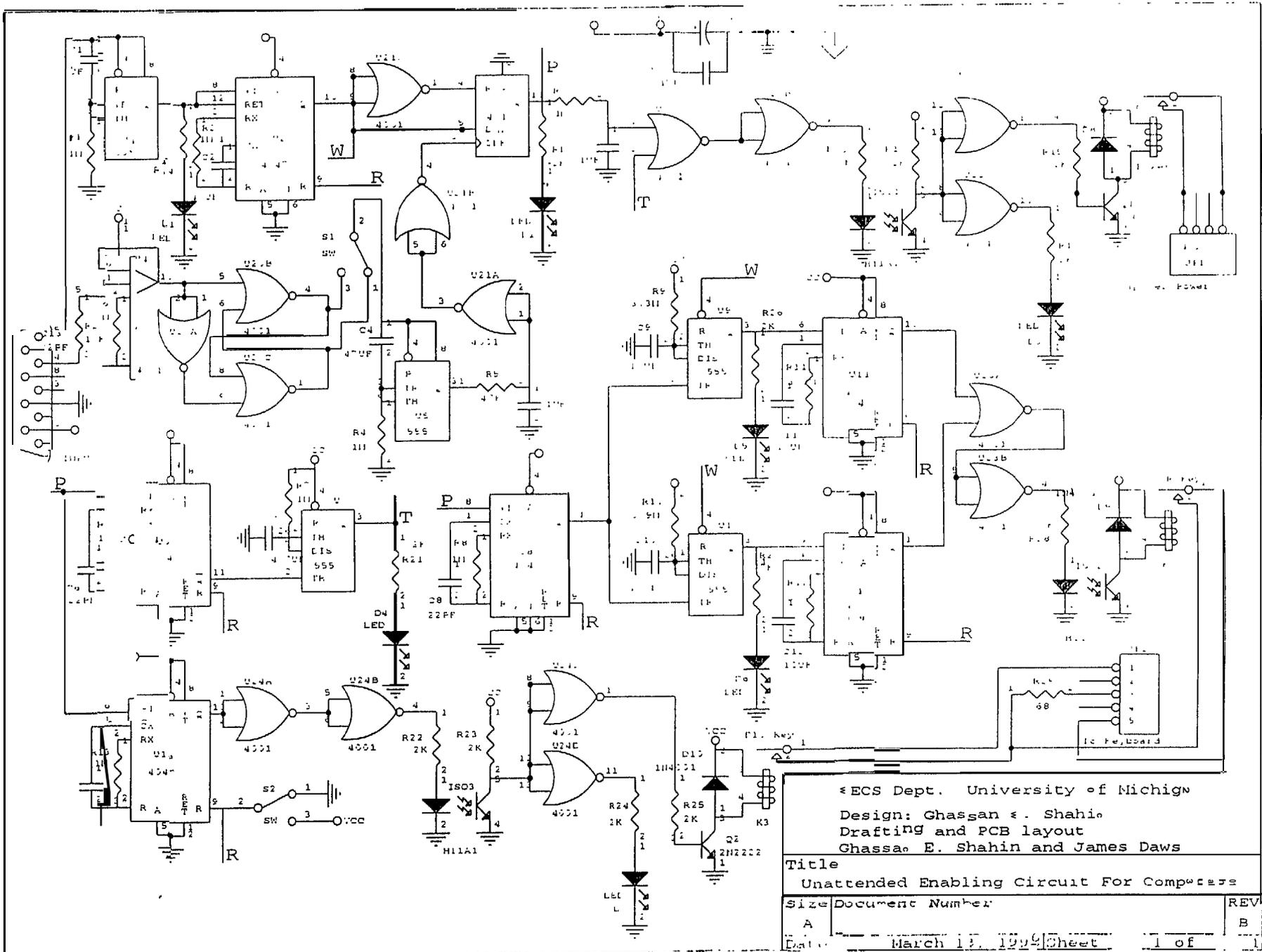
Ghassan E. Shahin

The VUC software, as it stands, requires attended keyboard operation. Since we need unattended operation that needs to sense when the vehicle will be driven (placed in Drive or Reverse) along with necessary time-delays between keyboard strokes activation, a relay - based circuit was devised and built by Ghassan E. Shahin. OrCAD software was used to create a printed circuit layout using ORCAD. This process was implemented with the collaboration of Dr. James Daws. The circuit and its printed circuit layout are shown on the next two pages. This circuit relieves the driver of any interaction with the data acquisition computer. Basically this electronic circuit performs as follows:

1. It automatically starts up a custom built computer once the ignition switch is on and the parking lever is in drive. This way we will not collect data before the vehicle starts moving
2. It executes keyboard equivalent commands so that the VUC software starts collecting the data.
3. The circuit will stay on if the parking lever is moved to neutral such as on a stop sign. This is necessary, otherwise the system will assume that we are on a new trip.
4. Once the ignition switch is turned off, the computer will stay on for few seconds. During this period the circuit executes proper keystroke so that the software terminates properly.

The unattended enabling circuit is used in conjunction with a computer that runs directly from the battery voltage via a DC-DC converter. This was necessary to avoid any inadvertent shock to the driver by an AC supply. The computers are IBM compatible machines suited for in-vehicle environment. They were custom made by Ghassan E. Shahin. Installation of these computers in commuters cars usually took 2 - 3 hours. The park neutral line from the transmission control module connector, in addition to the ignition, battery, and ground lines were necessary to operate the system. The computers were compact, and all the necessary circuitry were securely packaged inside the computer case.

It is noteworthy that all the circuits and the associated computers worked perfectly, and all trips were recorded successfully.



ECS Dept. University of Michigan
 Design: Ghasan E. Shahi
 Drafting and PCB layout
 Ghasan E. Shahi and James Daws
 Title
 Unattended Enabling Circuit For Computers
 Size Document Number: A
 Date: March 13, 1999 Sheet 1 of 1

Figure 20: Unattended Enabling Circuit

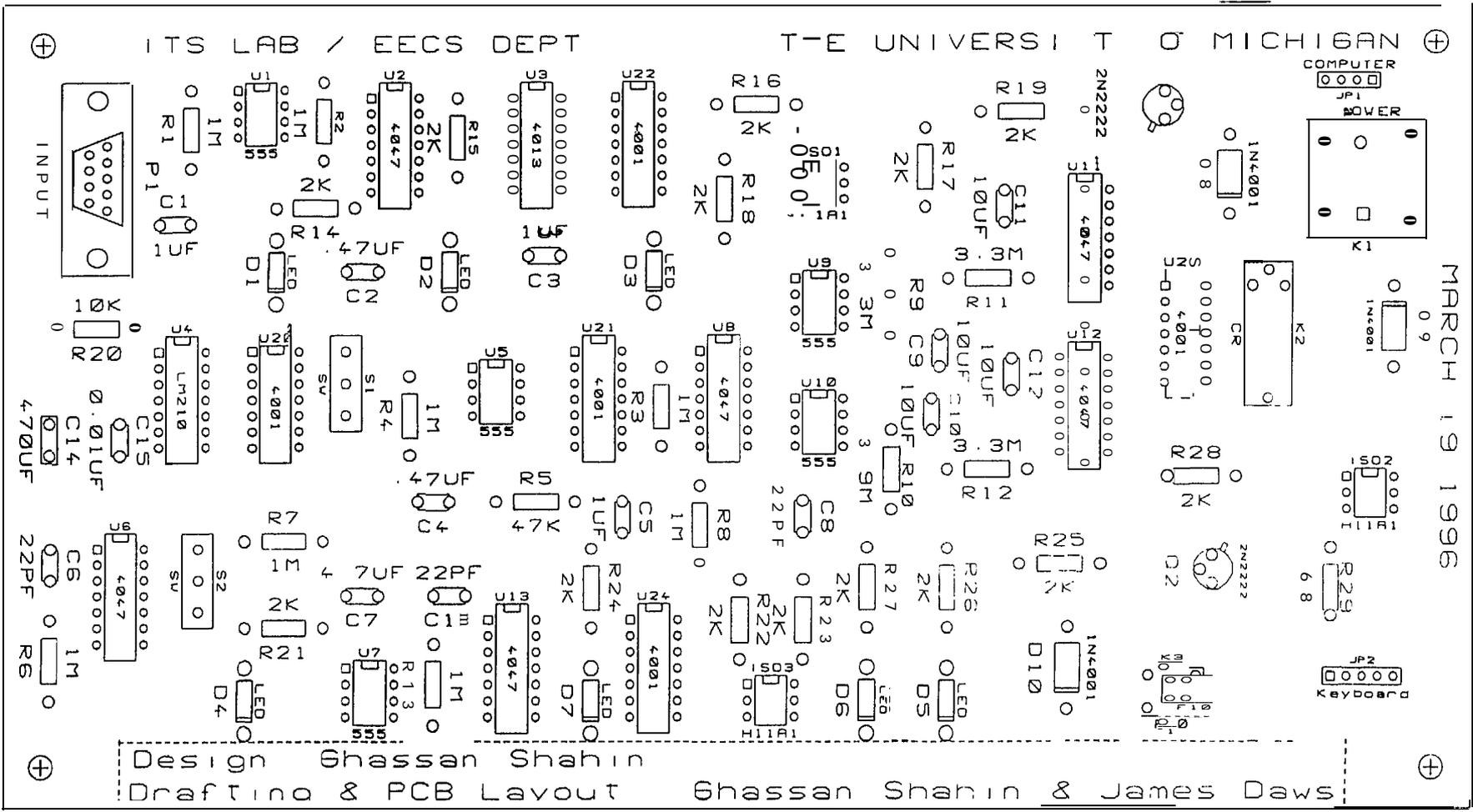


Figure 21: Printed Circuit Board Layout

APPENDIX D

**SAMPLE OF UM's SOFTWARE
OUTPUTS OF TECHNICAL PERFORMANCE DATA**

RESULT 546

Driver is: Commuter #1

Trip date is: 11/26/96 Trip start time is: 08:26:18

Trip duration is: 1554.00 seconds

Received : 6731 blocks. Repeated: 0 blocks

Transmitted: 2 blocks. Repeated: 0 blocks

Uplink telegrams: 1 blocks

_____Messages Sequence_____

Start-IR @27:38 lst_valid_IR @27:38 Down-link @27:38 Uplink @27:38
Cut-off @28:58

Start-IR @32:58 lst_valid_IR @32:58 Down-link @32:58 Cut-off @32:58

Start_IR @43:38 lst_valid_IR @43:38 Down-link @43:38 Cut-off @43:38

_____Display State_____

Starting display state is ON

Ending display state is ON

Number of display state changes is 1

Number of in vehicle unit errors: 0

_____Communication Information_____

Number of start IR: 3

Number of first IR: 3

Number of completed down links: 3

Number of completed up links: 1

Number of cutoff lines: 3

Number of failed downlinks: 0

_____Mode Sequencing Information_____

Initial vehicle mode is AUTONOMOUS

Time till next mode 94.500000

GUIDED

Time till next mode 0.500000

AUTONOMOUS

Time till next mode 0.500000

GUIDED

Time till next mode 1035.000000

TRACKING

Time till next mode 0.500000

AUTONOMOUS

Time till next mode 455.500000

Last vehicle mode is AUTONOMOUS

Total number of mode changes is 5

_____Destination Information_____

Origin Longitude = -831667739 Origin Latitude = 425390355

Destination Longitude = -830339000 Destination Latitude = 424922000

Euclidean distance between origin and destination = 7.550186 miles

First pos. in last tracking(Longitude = -830717175) (Latitude = 424911821)
Euclidean distance between last tracking and destination = 1.942821 miles
First pos. in last aut(Longitude = -830715945) (Latitude = 424911837)
Euclidean distance between last autonomous and destination = 1.936506 miles

Cumulative Timing Information

Total time: 1586.50 seconds
Total time in AUTONOMOUS mode 550.5 seconds
Total time in GUIDED mode 1035.5 seconds
Total time in TRACKING mode 0.5 seconds
Time until first beacon: 94.50 seconds
Time between first and last beacon: 1035.50 seconds
Time between first and last beacon unguided: 0.00 seconds
Time beyond last beacon: 456.50 seconds

Speed Histogram

0 - 5 mph: 3.48
5 - 25 mph: 5.57
25 - 35 mph: 5.87
35 - 45 mph: 7.83
45 - 55 mph: 3.33
55 - 65 mph: 0.03
65 - 80 mph: 0.13
> 80 mph: 0.17

Uplink Data

Veh-kind 0 Telegrams 0 @27:14 Beac_ID 85"Live-Data 0"Beac-ID 0"Live_Dat 0

Downlink Data

Beacon-ID 78 @27:6 Beacon-ID 79 @43:13Beacon_ID 96 @52:6

RESULT 8

Driver is: Commuter #2.
Trip date is: 09/24/96 Trip start time is: 13:05:17
Trip duration is: 3577.00 seconds
Received : 15791 blocks. Repeated: 0 blocks
Transmitted: 2 blocks. Repeated: 0 blocks
Uplink telegrams: 5 blocks

_____Messages Sequence_____

Start_IR @33:18 lst_valid_IR @33:18 Down-link @33:18 Cut-off @33:18
Start_IR @35:58 lst_valid_IR @35:58 Down-link @35:58 Cut-off @35:58
Start-IR @39:58 lst_valid_IR @39:58 Down-link @39:58 Cut-off @39:58
Start-IR @49: 18 lst_valid_IR @49:18 Down-link @49:18 Cut-off @49:18
Start-IR @5 1:58 lst_valid_IR @5 1:58 Down-link @5 1:58 Cut-off @51:58
Start-IR @55:58 lst-valid_IR @55:58 Down-link @55:58 Cut-off @55:58
Start-IR @59:58 lst-valid_IR @59:58 Down-link @59:58 Cut-off @59:58
Unlink @,59:58
Upink @,59:58
Uplink @59:58
Uplink @59:58
Uplink @59:58

_____Display State_____

Starting display state is ON
Ending display'state is ON
Number of display state changes is 1
Number of in vehicle unit errors: 0

_____Communication Information_____

Number of start IR: 7
Number of first IR: 7
Number of completed down links: 7
Number of completed up links: 5
Number of cutoff lines: 7
Number of failed downlinks: 0

_____Mode Sequencing Information_____

Initial vehicle mode is AUTONOMOUS
Time till next mode 1636.000000
GUIDED
Time till next mode 0.500000
AUTONOMOUS
Time till next mode 0.500000
GUIDED
Time till next mode 1964.500000
TRACKING
Time till next mode 0.500000

AUTONOMOUS

Time till next mode 66.500000

Last vehicle mode is AUTONOMOUS
Total number of mode changes is 5

Destination Information

Origin Longitude = -836865777 Origin Latitude = 422850264
 Destination Longitude = -832419000 Destination Latitude = 426452000
 Euclidean distance between origin and destination = 33.768836 miles
 First pos. in last tracking(Longitude = -832448712) (Latitude = 426446834)
 Euclidean distance between last tracking and destination = 0.156661 miles
 First pos. in last aut(Longitude = -832447520) (Latitude = 426447225)
 Euclidean distance between last autonomous and destination = 0.150091 miles

Cumulative Timing Information

Total time: 3668.50 seconds
 Total time in AUTONOMOUS mode 1703.0 seconds
 Total time in GUIDED mode 1965.0 seconds
 Total time in TRACKING mode 0.5 seconds
 Time until first beacon: 1636.00 seconds
 Time between first and last beacon: 1965.00 seconds
 Time between first and last beacon unguided: 0.00 seconds
 Time beyond last beacon: 67.50 seconds

Speed Histogram

0 - 5 mnh: 6.73
 5 - 25 mph: 4.93
 25 - 35 mph: 4.13
 35 - 45 mph: 5.97
 45 - 55 mph: 7.87
 55 - 65 mph: 14.97
 65 - 80 mph: 15.90
 > 80 mph: 0.63

Uplink Data

Veh_kind 0 Telegram# 4 @59:54 Beat_ID 96"Live_Data 1 "Beat_ID 0"Live_Data 0
 Veh_kind 0 Telegram# 3 @4:54 Beat_ID 97"Live_Data 1 "Beat_ID 0"Live-Data 0

Downlink Data

Beacon-ID 98 @32:12Beacon_ID 72 @39:28Beacon_ID 42 @48:9 Beacon-ID 85 @51:9
 Beacon-ID 34 @55:26Beacon_ID 21 @59:27Beacon_ID 26 @4:27

RESULT 211

Driver is: Commuter #3
Trip date is: 1 1/21/96 Trip start time is: 06:33:54
Trip duration is: 1734.00 seconds
Received : 8074 blocks. Repeated: 0 blocks
Transmitted: 2 blocks. Repeated: 0 blocks
Uplink telegrams: 5 blocks

Messages Sequence

1st validsIR @39: 12 Down-link @39: 12 Uplink @39: 12
Uplink @39:12
cut-off @39: 12
1 st_valid_IR @43 : 12 Down-link @43 : 12 Cut-off @43: 12
1 st_valid_IR @48:30 Uplink @48:30
Cut-off @48:30
Down-link @48:30 1st_valid_IR @49:50 Cut-off @49:50
Down-link @49:50 1st_valid_IR @51:8 Cut off @51:8
Down-link @5 1:8 1st_valid_IR @52:28 Uplink @52:28
Cut-off @52:28
Down-link @52:28 1st_valid_IR @53:48 Uplink @53:48
Cut-off @53 :48
Down-link @53:48 1st_valid_IR @55:8 Down-link @55:8 Cut-off @55:8

Display State

Starting display state is ON
Ending display state is OFF
Number of display state changes is 2
Number of in vehicle unit errors: 0

Communication Information

Number of start IR: 0
Number of first IR: 8
Number of completed down links: 8
Number of completed up links: 5
Number of cutoff lines: 8
Number of failed downlinks: 0

Mode Sequencing Information

Initial vehicle mode is AUTONOMOUS
Time till next mode 261.000000
GUIDED
Time till next mode 631.500000
TRACKING
Time till next mode 10.000000
GUIDED
Time till next mode 40.500000
TRACKING
Time till next mode 57.500000
GUIDED
Time till next mode 54.500000
TRACKING
Time till next mode 33.500000
GUIDED

Time till next mode 47.000000
 TRACKING
 Time till next mode 44.500000
 GUIDED
 Time till next mode 39.500000
 TRACKING
 Time till next mode 28.000000
 GUIDED
 Time till next mode 38.000000
 TRACKING
 Time till next mode 41.000000
 AUTONOMOUS
 Time till next mode 401.000000

Last vehicle mode is AUTONOMOUS
 Total number of mode changes is 13

Destination Information
 Origin Longitude = -832313326 Origin Latitude = 427303514
 Destination Longitude = -831533000 Destination Latitude = 425622000
 Euclidean distance between origin and destination = 12.288083 miles
 First pos. in last tracking(Longitude = -83 1021027) (Latitude = 424581644)
 Euclidean distance between last tracking and destination = 7.652820 miles
 First pos. in last aut(Longitude = -830964669) (Latitude = 424498521)
 Euclidean distance between last autonomous and destination = 8.291851 miles

Cumulative Timing Information
 Total time: 1727.50 seconds
 Total time in AUTONOMOUS mode 662.0 seconds
 Total time in GUIDED mode 851.0 seconds
 Total time in TRACKING mode 214.5 seconds
 Time until first beacon: 260.50 seconds
 Time between first and last beacon: 1024.50 seconds
 Time between first and last beacon unguided: 173.50 seconds
 Time beyond last beacon: 442.50 seconds

Speed Histogram
 0 - 5 mph: 0.86
 5 - 25 mph: 1.73
 25 - 35 mph: 1.70
 35 - 45 mph: 1.60
 45 - 55 mph: 1.97
 55 - 65 mph: 3.93
 65 - 80 mph: 9.27
 > 80 mph: 7.73

Uplink Data
 Veh-kind 0 Telegram# 1 @38:10 Beat_ID 80"Live_Data 1"Beac_ID 0"Live_Data 0
 Veh-kind 0 Telegram# 0 @2:48 Beat_ID 55"Live_Data 1"Beac_ID 0"Live_Data 0

Downlink Data
 Beacon-ID 49 @38:5 Beacon-ID 26 @47:22Beacon_ID 27 @48:29Beacon_ID 24 @50:18
 Beacon-ID 93 @52:4 Beacon-ID 22 @53:19Beacon_ID 94 @54:23Beacon_ID 95
 @2:24

RESULT1 33

Driver is: Commuter #4

Trip date is: 08/20/96 Trip start time is: 06:51:08

Trip duration is: 1928.00 seconds

Received : 8274 blocks. Repeated: 0 blocks

Transmitted: 2 blocks. Repeated: 0 blocks

Uplink telegrams: 3 blocks

_____Messages Sequence_____

Start-IR @4:28 Ist_valid_IR @4:28 Down-link @4:28 Uplink @4:28
Cut-off @4:28

Start-IR @9:48 Ist_valid_IR @9:48 Cut-off @9:48

Start-IR @12:30 Ist_valid_IR @12:30 Down-link @13:48 Cut-off @13:48
Uplink @13:48
Uplink @13:48

Start-IR @16:28 Ist_valid_IR @16:28 Down-link @16:28 Cut-off @16:28

_____Display State_____

Starting display state is ON

Ending display state is ON

Number of display state changes is 1

Number of in vehicle unit errors: 0

_____Communication Information_____

Number of start IR: 4

Number of first IR: 4

Number of completed down links: 3

Number of completed up links: 3

Number of cutoff lines: 4

Number of failed downlinks: 1

_____Mode Sequencing Information_____

Initial vehicle mode is AUTONOMOUS

Time till next mode 753.000000

GUIDED

Time till next mode 0.500000

AUTONOMOUS

Time till next mode 1.000000

GUIDED

Time till next mode 790.500000

TRACKING

Time till next mode 7.000000

GUIDED

Time till next mode 45.000000

TRACKING

Time till next mode 0.500000

AUTONOMOUS

Time till next mode 367.000000

Last vehicle mode is AUTONOMOUS

Total number of mode changes is 7

Destination Information
Origin Longitude = -834520820 Origin Latitude = 425591289
Destination Longitude = -830455000 Destination Latitude = 424797000
Euclidean distance between origin and destination = 21.583098 miles
First pos. in last tracking(Longitude = -830716115) (Latitude = 424911659)
Euclidean distance between last tracking and destination = 1.557100 miles
First pos. in last aut(Longitude = -830714164) (Latitude = 424911508)
Euclidean distance between last autonomous and destination = 1.547951 miles

Cumulative Timing Information
Total time: 1964.50 seconds
Total time in AUTONOMOUS mode 1121 .0 seconds
Total time in GUIDED mode 836.0 seconds
Total time in TRACKING mode 7.5 seconds
Time until first beacon: 753.50 seconds
Time between first and last beacon: 843.00 seconds
Time between first and last beacon unguided: 7.00 seconds
Time beyond last beacon: 368.00 seconds

Speed Histogram
0 - 5 mph: 3.34
5 - 25 mph: 3.80
25 - 35 mph: 2.03
35 - 45 mph: 4.07
45 - 55 mph: 5.77
55 - 65 mph: 5.53
65 - 80 mph: 7.57
> 80 mph: 0.63

-Uplink Data
Veh_kind 0 Telegram# 3 @3:36 Beac-ID 102"Live_Data 1"Beac_ID 0"Live_Data 0
Veh_kind 0 Telegram# 1 @12:32 Beac-ID 102"Live_Data 1"Beac-ID 0"Live_Dat 0
Veh-kind 0 Telegram# 0 @23:16 Beat-ID 85"Live_Data 0"Beac_ID 0"Live_Data 0
Downlink Data
Beacon-ID 60 @3: 18 Beacon-ID 82 @16:9 Beacon-ID 96 @23:8

RESULT 73

Driver is: van
Trip date is: 07/16/96 Trip start time is: 21:02:52
Trip duration is: 1896.00 seconds
Received : 8803 blocks. Repeated: 0 blocks
Transmitted: 2 blocks. Repeated: 0 blocks
Uplink telegrams: 4 blocks

_____Messages Sequence_____

Start-IR @8:12 lst_valid_IR @8:12 Down-link @8:12 Cut-off @8:12
Uplink @8:12
Uplink @8:12

Start-IR @10:52 lst_valid_IR @10:52 Down-link @10:52 Cut-off @10:52

Start_IR @13:32 lst_valid_IR @13:32 Down-link @13:32 Cut-off @13:32

Start-IR @14:52 lst_valid_IR @14:52 Down-link @14:52 Cut-off @14:52
Uplink @14:52

Start-IR @17:32 lst_valids_IR @17:32 Down-link @17:32 Cut-off @17:32

Start-IR @21:32 lst_valid_IR @21:32 Down-link @21:32 Uplink @21:32
Cut-off @21:32

Start-IR @24:12 lst_valid_IR @24:12 Down-link @25:32 Cut-off @25:32

_____Display State_____

Starting display state is ON
Ending display state is ON
Number of display state changes is 1
Number of in vehicle unit errors: 0

_____Communication Information_____

Number of start IR: 7
Number of first IR: 7
Number of completed down links: 7
Number of completed up links: 4
Number of cutoff lines: 7
Number of failed downlinks: 0

_____Mode Sequencing Information_____

Initial vehicle mode is AUTONOMOUS
Time till next mode 264.500000
GUIDED
Time till next mode 72.500000
TRACKING
Time till next mode 131.000000
GUIDED
Time till next mode 80.000000
TRACKING
Time till next mode 86.000000
GUIDED

Time till next mode 157.000000
TRACKING
Time till next mode 63.500000
GUIDED
Time till next mode 198.000000
TRACKING
Time till next mode 63.000000
GUIDED
Time till next mode 94.500000
TRACKING
Time till next mode 96.000000
GUIDED
Time till next mode 83.500000
TRACKING
Time till next mode 137.000000
AUTONOMOUS
Time till next mode 391.500000

Last vehicle mode is AUTONOMOUS
Total number of mode changes is 13

Destination Information
Origin Longitude = -832419692 Origin Latitude = 426458974
Destination Longitude = -8324 19000 Destination Latitude = 426452000
Euclidean distance between origin and destination = 0.048310 miles
First pos. in last tracking(Longitude = -83 1135797) (Latitude = 424761811)
Euclidean distance between last tracking and destination = 13.406886 miles
First pos. in last aut(Longitude = -830835859) (Latitude = 424897619)
Euclidean distance between last autonomous and destination = 13.467547 miles

Cumulative Timing Information
Total time: 1918.00 seconds
Total time in AUTONOMOUS mode 656.0 seconds
Total time in GUIDED mode 685.5 seconds
Total time in TRACKING mode 576.5 seconds
Time until first beacon: 264.00 seconds
Time between first and last beacon: 1125.00 seconds
Time between first and last beacon unguided: 439.50 seconds
Time beyond last beacon: 529.00 seconds

Speed Histogram
0 - 5 mph: 1.02
5 - 25 mph: 6.10
25 - 35 mph: 4.03
35 - 45 mph: 2.50
45 - 55 mph: 4.47
55 - 65 mph: 7.00
65 - 80 mph: 5.93
> 80 mph: 0.90

Uplink Data
Veh_kind0 Telegram# 5 @7: 18 Beac-ID 59"Live_Data 1"Beat_ID 0"Live_Data 0
Veh_kind 0 Telegram# 4 @34:28 Beac_ID 60"Live_Data 1"Beac_ID 0"Live_Dat 0
Downlink -Data

Beacon-ID 26 @7:9 Beacon-ID 23 @13: 13 Beacon-ID 27 @14:26 Beacon_ID 24 @16:27
Beacon-ID 93 @21:4 Beacon-ID 22 @24:8 Beacon-ID 94 @34: 14

RESULT 113

Yoked

Trip date is: 08/06/96 Trip start time is: 10:38:05

Trip duration is: 2201.00 seconds

Received : 98 19 blocks. Repeated: 0 blocks

Transmitted: 2 blocks. Repeated: 0 blocks

Uplink telegrams: 4 blocks

Messages Sequence

1 st_valid_IR @40:44 1st_valid_IR @40:44 Uplink @40:44

cut-off @40:44

Down-link @40:44 1st_valid_IR @43:24 Cut off @43:24

Down-link @43:24 1st_valid_IR @56:42 Uplink @56:42

Uplink @56:42

Down link @56:42 Cut-off @56:42

1st_vaiid_IR @2:0 Down-link @2:0 Uplink @2:0

Cut-off @2:0

1st_valid_IR @4:42 Down-link @6:0 Cut-off @6:0

Display State

Starting displav state is OFF

Ending display state is OFF

Number of display state changes is 0

Number of in vehicle unit errors: 0

Communication Information

Number of start IR: 0

Number of first IR: 6

Number of completed down links: 5

Number of completed up links: 4

Number of cutoff lines: 5

Number of failed downlinks: 0

Mode Sequencing Information

Initial vehicle mode is AUTONOMOUS

Time till next mode 117.500000

GUIDED

Time till next mode 84.000000

TRACKING

Time till next mode 145.000000

GUIDED

Time till next mode 95.500000

TRACKING

Time till next mode 345.500000

AUTONOMOUS

Time till next mode 419.500000

GUIDED

Time till next mode 99.500000

TRACKING

Time till next mode 1.000000

AUTONOMOUS

Time till next mode 90.000000

GUIDED

Time till next mode 282.000000

TRACKING

Time till next mode 12.000000

AUTONOMOUS

Time till next mode 80.500000

Last vehicle mode is AUTONOMOUS

Total number of mode changes is 11

Destination Information

Origin Longitude = -83 15 12463 Origin Latitude = 425638777
 Destination Longitude = -83 1500000 Destination Latitude = 42563 1000
 Euclidean distance between origin and destination = 0.083550 miles
 First pos. in last tracking(Longitude = -831477115) (Latitude = 425646060)
 Euclidean distance between last tracking and destination = 0.156936 miles
 First pos. in last aut(Longitude = -831483865) (Latitude = 425637686)
 Euclidean distance between last autonomous and destination = 0.094845 miles

Cumulative Timing Information

Total time: 1772.00 seconds
 Total time in AUTONOMOUS mode 707.5 seconds
 Total time in GUIDED mode 561 .0 seconds
 Total time in TRACKING mode 503.5 seconds
 Time until first beacon: 117.00 seconds
 Time between first and last beacon: 1562.00 seconds
 Time between first and last beacon unguided: 1001 .00 seconds
 Time beyond last beacon: 93.00 seconds

Speed Histogram

0 - 5 mph: 5.24
 5 - 25 mph: 4.80
 25 - 35 mph: 2.70
 35 - 45 mph: 7.70
 45 - 55 mph: 6.70
 55 - 65 mph: 0.93
 65 - 80 mph: 0.80
 > 80 mph: 0.63

Uplink Data

Veh_kind 0 Telegram# 0 @39:54 Beac_ID 14"Live_Data 1"Beac_ID 0"Live_Dat 0
 Veh_kind 0 Telegram# 1 @56:28 Beat_ID 20"Live_Data 1"Beac_ID 0"Live_Dat 0
 Veh_kind 0 Telegram# 0 @14:46 Beat_ID 21"Live_Data 1 "Beat_ID 0"Live-Dat 0

Downlink Data

Beacon-ID 7 @40:0 Beacon-ID 57 @56:17Beacon_ID 39 @1:10 Beacon-ID 57 @4:23
 Beacon-ID 7 @14:23

APPENDIX E

HARDWARE FAILURE OVERVIEW

Appendix E

Ali-Scout Failure Report

Hardware Failure Overview

There have been approximately 55 IVU hardware failures to date. The unit failures are approximately as follows:

DCU	25 units
NAC	20 units
ITR	5 units
MFS	5 units

There have been approximately 85 failures of cards in the beacon heads and beacon controllers to date. These failures are broken down as follows:

beacon heads	40
RS232 cards	15
processor cards	10
communication cards -	10
power supply	10

May 1996

1) Number of BC's down carried into May: 10

18	Not on line yet - up St 13/96
23	Troubleshooting - up 5/2/96
26	No beacon / construction - carried over into next month
56	Troubleshooting - up 5/1 6/96
66	Beacon off / bad data - carried over into next month
71	Beacon off / bad data - carried over into next month
73	AMERITECH line problem - up 5/2/96
79	Not on line yet - carried over into next month
84	Troubleshooting - carried over into next month
86	Not on line yet - carried over into next month

2) BC's that went down during this month, how long they were down, and why.
(total of 20; inclusive of a few beacons that went down more than once): -

7	1 day	Modem problem
73	1 day	AMERITECH line problem
4	carried over	AMERITECH line problem
39	2 days	Construction
1	4 days	No power
51	4 days	No power
105	carried over	AMERITECH line problem
40	1 day	PDR
31	2 days	Controller problem
50	carried over	Controller problem
95	5 days	AMERITECH line problem
97	1 day	Construction
83	1 day	Construction
9	1 day	AMERITECH line problem

30	10 days	AMERITECH line problem
10	1 day	Controller problem
92	1 day	PDR
83	1 day	PDR
72	carried over	Controller problem
83	carried over	No power

(PDR: power down and reset, e.g. after thunderstorm)

3) Number of BC's down carried into next month: 11

4	AMERITECH line problem
26	No beacon / construction
50	Controller problem
66	Beacon off / bad data
71	Controller problem
72	Controller problem
79	Controller problem
83	No power
84	AMERITECH / Siemens still troubleshooting
86	Simpack distribution box problem
105	AMERITECH line problem

4) Number of head problems per month (average): 3 - 5

June 1996

1) Number of BC's down carried into this June: 11

4	AMERITECH line problem - carried over into next month
26	No Beacon / construction - carried over into next month
50	Controller problem - new controller installed 6/24/96
66	Beacon off / bad data - carried over into next month
71	Controller problem - new controller installed 6/11/96
72	Controller problem - new controller installed 6/11/96
79	Controller problem - new controller installed 6/28/96
83	No power - power restored 6/5/96
84	AMERITECH / Siemens still troubleshooting - carried over into next month
86	Simpack distribution box problem - replaced 6/13/96
105	AMERITECH line problem - fixed 6/28/96

2) BC's that went down during June, how long they were down, and why.
(total of 23; inclusive of a few beacons that went down more than once):

87	15 days	AMERITECH line problem
22	17 days	AMERITECH line problem / controller problem
16	16 days	Controller problem
97	1 day	AMERITECH line problem
17	10 days	Controller problem
13	1 day	AMERITECH line problem
44	2 days	Tripped circuit breaker
11	7 days	AMERITECH line problem

24	3 days	Controller problem
39	3 days	Tripped circuit breaker
7	1 day	PDR
9	2 days	PDR
10	2 days	Modem problem
99	4 days	AMERITECH line problem
17	1 day	Controller problem
18	1 day	PDR
21	1 day	PDR
41	1 day	PDR
6	1 day	No power
82	1 day	PDR
97	1 day	Controller problem
39	1 day	PDR
84	1 day	PDR

3) Number of BC's down carried into July: 4

4	AMERITECH line problem
26	No Beacon / construction
66	Beacon off / bad data
84	AMERITECH / Siemens still troubleshooting

4) Number of head problems per month (average): 3 - 5

5) Other problems occurring during the month:

- Online system stopping for unknown reason. Seemed to run fine for 4 - 9 hours, then simpact.hdlc and bakdat stopped running. Numerous comms-upload faults were coming in from lines 17 - 32 and the hard drive on the Motorola was being filled up by a very large syserr.txt file. Rebooting and clearing out this file was the only way to get the system running again. This problem happened repeatedly for approximately 20 days at which time it was recommended that we switch cables between Simpact cards 0 and 1, and reconfigure software. The problem did not occur again, so the cables were changed back to their original positions and the system has been running fine ever since. This appears to have been nothing more than a cable seating problem. While this was happening, however, there was one morning that the HP755 locked up altogether and we had to cycle power to get it to come back up. When it did, it would not initialize the beacons due to the bc-table tile being missing. Running net-config rebuilt the file and the system came up. While looking through the /tmp/utmsdata directory, it was noticed that there were two files in it that were impossibly large (1.89 GB each) and were removed.

July 1996

1) Number of BC's down carried into July: 4

4	AMERITECH line problem / redesigning circuit
26	No Beacon / construction
66	Beacon off / bad data
84	AMERITECH / Siemens still troubleshooting

2) BC's that went down during July, how long they were down, and why.

(total of 23; inclusive of a few beacons that went down more than once):

2	1 day	no power	
39	1 day	no power	
40	3 days	no power / PDR when power came back	I-- Thunderstorm
43	1 day	no power	
54	7 days	no power	---
93	1 day	AMERITECH line problem	
44	1 day	tripped circuit breaker	
83	1 day	PDR	
42	1 day	PDR	
99	1 day	AMERITECH line problem	
30	1 day	AMERITECH line problem	
71	carried over	AMERITECH line problem	
80	3 days	no power	
39	1 day	tripped circuit breaker	
65	1 day	AMERITECH line problem	
6	carried over	controller removed / Det Ed replacing pole	
44	1 day	PDR	
83	1 day	PDR	
19	1 day	AMERITECH line problem	--
53	carried over	AMERITECH line problem	
92	1 day	AMERITECH line problem	I- Thunderstorm
93	1 day	AMERITECH line problem	
105	1 day	AMERITECH line problem	-em

Note: 1day problems were noted in the morning and fixed some time during that day.

3) Number of BC's down carried into August: 7

4	AMERITECH line problem / redesigning circuit
6	controller removed / Det Ed replacing pole
26	No Beacon / construction
53	AMERITECH line problem
66	Beacon off/ bad data
71	AMERITECH line problem
84	AMERITECH / Siemens still troubleshooting

4) Number of head problems per month (average): 3 - 5

5) Other problems occurring during the month of July:

15 Jul - Loaded three beacons with version 143 software in order to better track duplicate telegrams coming in. When these beacons were loaded with this software, they stopped collecting vehicle telegrams altogether. This is a problem that has been found with version 142 software and thought was fixed in 143. Fix under development. There is also a minor problem with DISPONENT; settings are not being saved properly in the profile options window. This is also being worked on by Germany.

APPENDIX F

COMMUTE DRIVER CONSENT FORM

principal Investigaor: Marlin Ristenbatt, Tele: 313-764-5202

Engineer: Ghassan Shalin p-371

INFORMED CONSENT FORM

The purpose of this experiment is to determine how use of the route guidance system called Ali-Scout improves (or not) your commute trip to and from your job. This system displays navigation information visually and out loud. Your participation involves agreeing to let us install a data-acquisition computer and an Ali-Scout unit in your vehicle for a period of two months. During the first month you are to use your usual routes to and from your job; elapsed time and the speed profile of your trip will be recorded. During the second month the Ali-Scout unit will be utilized, and you are asked to follow its recommendations for the commute trips. When the equipment is removed we will ask you to complete a questionnaire about your experiences with the system, during both commute and other trips that you took.

The results from this study will be published, but your name will not appear on any of the reports. All information that you give us will be kept strictly confidential.

The requirements for participation are that you have a valid drivers License and are willing to follow the recommended route steps, for the commute trips only, during the second month of your participation. If you decide to participate and later do not want to continue, you may withdraw without any penalty.

At no time should you do anything unsafe while driving the car. The in-vehicle system could be distracting, but it is under your control. As such, the only risks associated with this study are those associated with your normal driving.

I have read and understand the information presented above. I understand my participation in this study is entirely voluntary and I may withdraw at any time without penalty.

Print name: _____

Signature: _____

Date: _____

**APPENDIX G
CROSS REFERENCE OF VUC IDs AND BEACON MAP IDs**

<i>VUC</i>	<i>MAP</i>		<i>VUC</i>	<i>MAP</i>		<i>VUC</i>	<i>MAP</i>		<i>VUC</i>	<i>MAP</i>
0	37		30	42		60	102		90	81
1	13!		31			61			91	
2	3		32			62			92	
3	57		33	40		63			93	
4	9		34	95		64			94	64
5	24		35	82		65	106		95	65
6	107		36			66	100		96	66
7	21		37	30!		67			97	18
8	14		38	53		68			98	79
9			39	12		69	85		99	96/85!
10	7		40			70	88		100	
11	87		41	101		71	43		101	
12			42	97		72	73		102	
13	5!		43	10		73	72		103	
14			44	108		74	47		104	
15			45	31		75	1		105	
16	38		46	18		76	51		106	
17	45		47	32		77	11		107	
18	46		48			78	27		108	
19	56		49	55!		79			109	
20	8		50	89		80	16!		110	
21	44		51			81			111	
22	63		52	41		82	71		112	
23	59		53			83	41		113	
24	22		54			84			114	
25	52		55	2		85	96		115	
26	58		56	19		86	93&94		116	
27	61		57	20!		87			117	
28	54/33		58			88	84!		118	
29	57		59	109		89	84!		119	



UNIVERSITY OF MICHIGAN
Transportation
Research Institute

**Research
Laboratory**

200 Engineering Programs Bldg.
2609 Draper Drive
Ann Arbor, MI
48109-2140

Program Offices Telephone:
313-764-4333

Program Offices Facsimile:
313-763-1674

UNIVERSITY OF MICHIGAN