Programming a digital simulation model of a freeway diamond interchange

by E. HAYES
System Development Corporation
Santa Monica, California

INTRODUCTION

The broad aim of the vehicular traffic study under way at System Development Corporation is to develop some much-needed basic knowledge that can be used to provide a tool for research on freeway diamond interchanges — research promoting the improvement of the geometric design of the interchanges themselves and the development of effective control devices regulating traffic flow through them. The method of the study is to simulate the traffic flow in a diamond interchange between a freeway and an arterial street, and to validate this model by field data. A complete description of the entire simulation and validation program can be found in reference.[1]

Figure 1 shows the geometric configuration of a typical diamond interchange. Because of the sheer size of the physical area that must be examined, the number of vehicles involved, and the complexity of the task

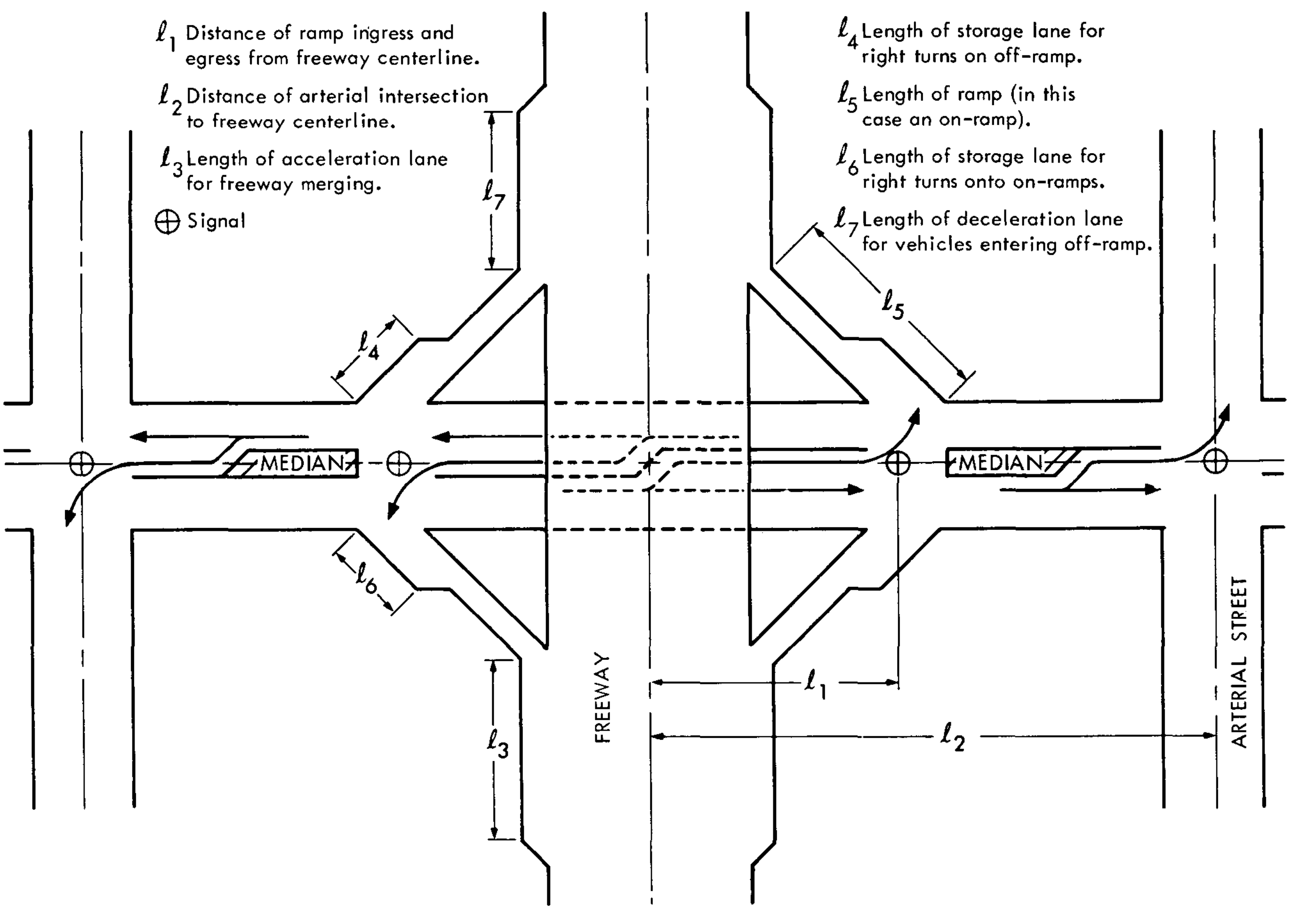

Figure 1 — Schematic diagram of a diamond interchange showing some geometrical parameters

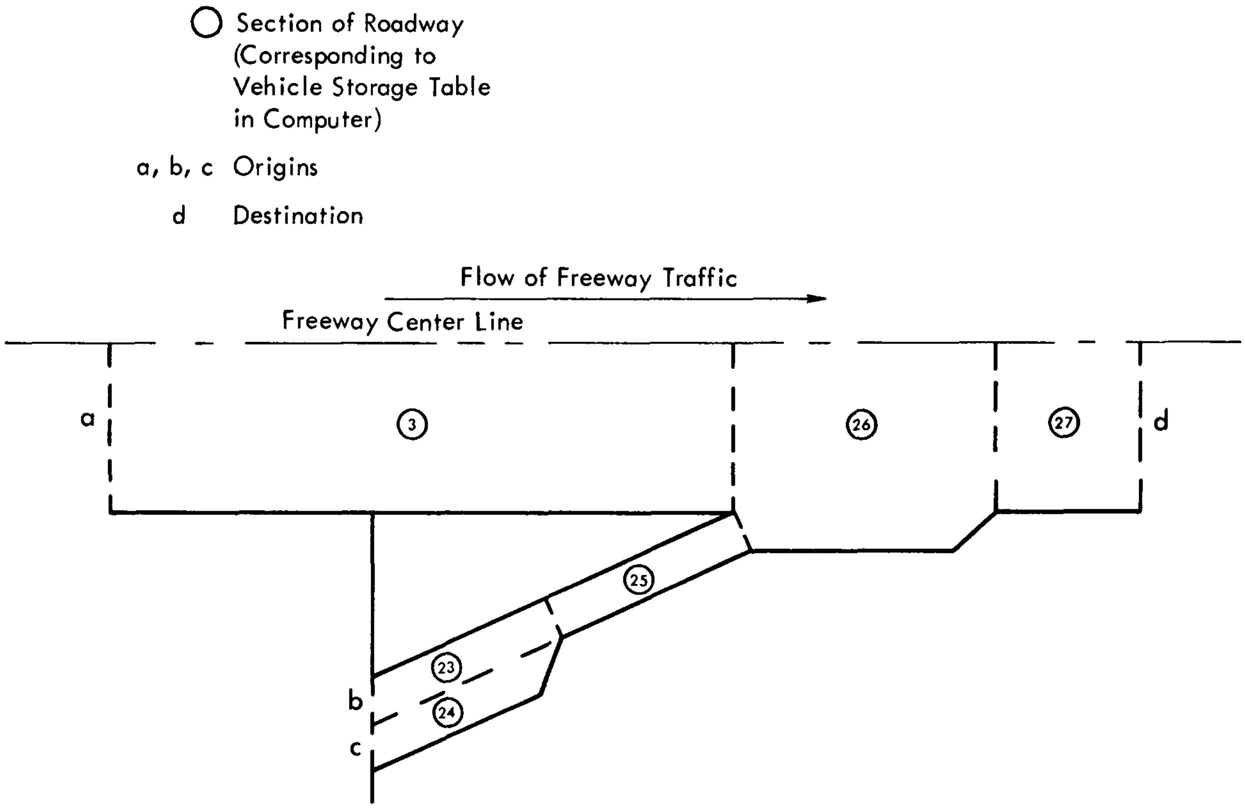

Figure 2 — Merging of freeway with on-ramp

itself, the study has been divided into four phases or modules:

1. Merging of an on-ramp with freeway traffic (Figure 2);
2. Merging of an on-ramp with freeway traffic and an off-ramp with arterial traffic (Figure 3);
3. Processing of all arterial traffic (Figure 4);
4. Simulation of the operation of the entire interchange by appropriate combination of modules 2 and 3 above.

The first phase, or Model 1, has already been formulated and coded; it is currently operating on the Philco 2000 digital computer. This paper contains a description of the programming components of the first phase, including some of the data reduction routines and the outputs they produce. The paper concludes with an indication of the future directions to be taken in this study.

Design considerations

A *dominant* objective was to design a computer program logic that would minimize the ratio of computer running time to stimulate time. This was of particular importance since the program was to operate on a digital computer having approximately one-third the processing speed of the IBM 7090. A number of vehicular traffic simulation models use a "microscopic" program logic, by design.[2,3,4,5,6] They account for the position of each vehicle precisely at each stage in the simulation. In order to handle traffic efficiently in a system that can contain upwards of 600 vehicles (Model 1), it was decided not to use such a microscopic approach in moving traffic within the system.

Instead, the microscopic logic approach was chosen; the simulated vehicles were absorbed into logical groupings, as are those in the system developed by Gerlough.[7] Thus, the model was broken down into several interconnected submodels of relatively uniform geometric configuration. Through each of these sub-models, vehicles are moved analytically, with a determination of the configuration of traffic in each sub-model accomplished once per time period or signal phase change. These time periods, being larger than those normally encountered in incremental simulation models,[2] produced considerable improvement in computer running time.

It was determined early that the physical structuring of the information required for representing vehicles in the system was of critical importance, as it would have direct bearing on the running time of the program. Since the model was to be aggregate in nature, with vehicles considered within groups, there would be no such entities as lane, actual speed, current position, space to

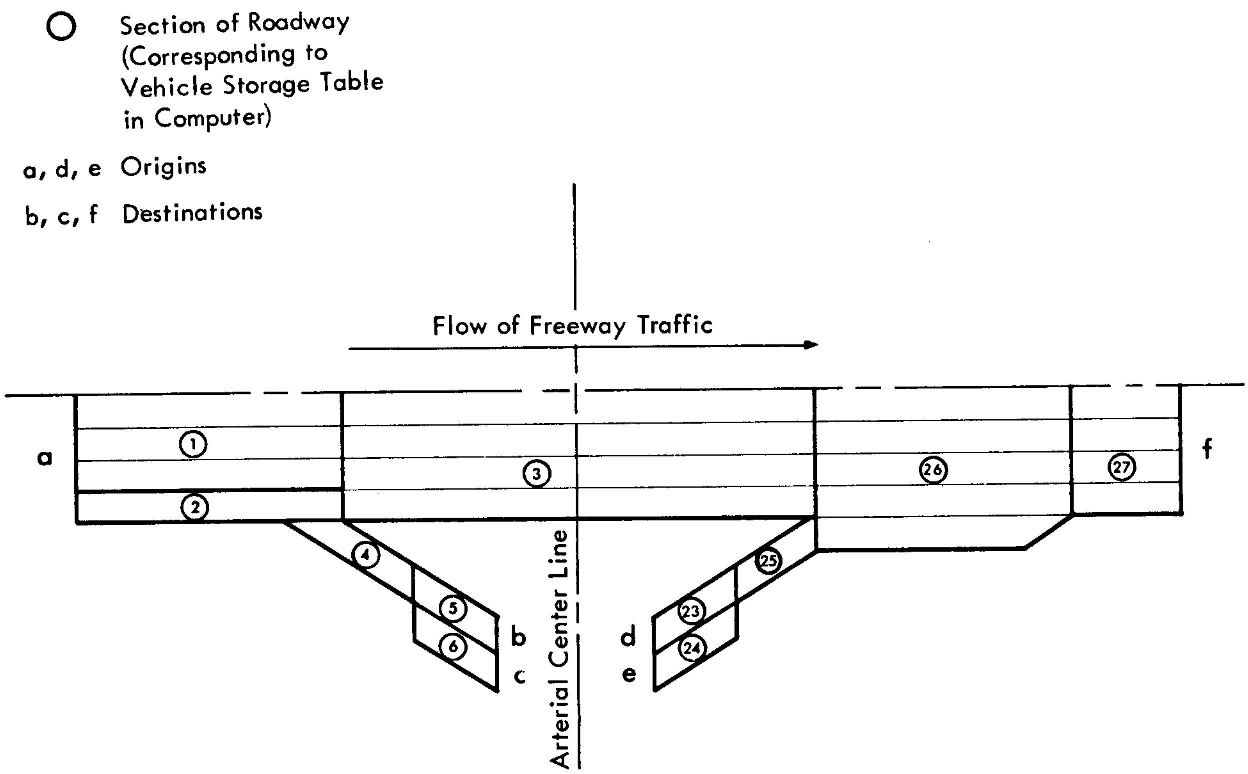

Figure 3 — One-half of freeway with its corresponding on- and off-ramps

next vehicle, etc. Thus, the *physical representation* scheme as detailed by Gerlough,[2] whereby 1-bits represent vehicles in a binary number and 0-bits the spaces between the vehicles, could not be employed. In this technique, vehicular movements take place as 1-bits are shifted within and between successive computer words, which are used as positions in a roadway or network.

However, Gerlough's *mathematical or memorandum* scheme appeared much better.[2] It would allow the representation of all the conditions pertaining to a given vehicle by a code or number in a single computer word, the parts of which could be manipulated and updated as desired by suitable routines. Whenever a vehicle changed lanes, Gerlough simply changed the "lane" field in the computer word to its new value. He did not have to move his vehicle entry into a table representing that new lane, and he did not have to reorder. This seemed to be a good method, since it minimized processing time. But, when processing a section of roadway, he had to scan all vehicles in that section to determine which were the ones of immediate interest. This factor led to the rejection of his second scheme, as well.

Accordingly, a new method was developed for the structuring of the data used for representing vehicles in the system — a method felt to be unique in vehicular traffic simulations. The vehicles are organized into *list structures*, each list representing vehicles currently in a particular section of the roadway or sub-model. Each vehicle in a list "points" to other vehicles "fore" and "aft." Thus, processing of a given list can be accomplished in two directions. No computer words are passed from one list to another; only address links are changed. This makes scans of the system simpler in that fewer vehicles need to be looked at. It also allows for using only as much core space as is actually required for vehicles currently in the system. Fixed amounts of table space do not have to be allocated even though only parts of the table are being used.

As a *second* design consideration, the system had to be able to perform various data reductions of the model's performance even though, initially, the type and format of the output were not known. It *was* known that the model would be employed as a research tool with continually changing specifications. With this in mind, an "open-ended" approach was used. A common data base was made available for all future data reduction requests. In this way, reprogramming was considerably minimized in order to furnish the desired inputs to the various data reduction routines, as they were developed.

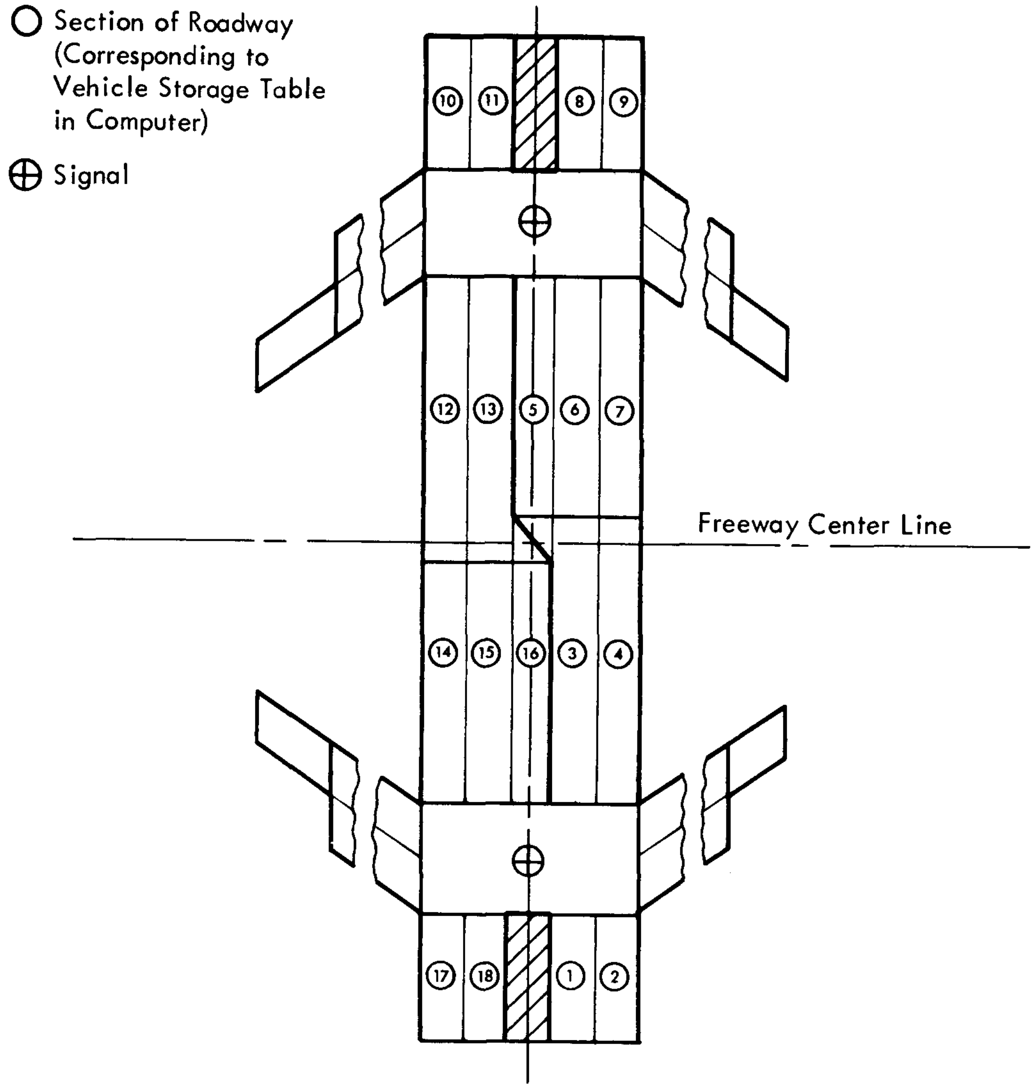

Figure 4 — Arterial surface street with on- and off-ramps

The *third* design consideration was flexibility in determining or establishing the state or configuration of the model for any run or set of runs. This included such diverse things as:

1. Choosing to initialize or not to initialize the pseudorandom number generator;
2. Being able to suppress all, some, or none of the data reduction routines;
3. Choosing when and if various tables were to be output for inspection;
4. Having a means to monitor the operation of the model while it was operating;
5. Having the traffic flows determined by a stochastic or deterministic process;
6. Providing for the dynamic changing of system variables and parameters while in the middle of a run.

Figure 2 shows the schematic geometrics of Model 1, which covers approximately one-fourth of a diamond interchange. It includes six sub-models, as follows: one entrance ramp (V_{23}, V_{24} and V_{25}), the merging-weaving section (V_{26}) and a long section of uniform roadway before and after the weaving section (V_3 and V_{27}). Excluded are the surface arterials that would normally be associated with this ramp.

These sub-models require the consideration of passing, weaving, acceleration, deceleration, merging, and shock-wave build-up and dissipation. Through the use of a Vehicle Storage Table (VSTBL), vehicle identity is maintained within each sub-model insofar as driver characteristics, entrance characteristics, and projected exit characteristics are concerned. The set of vehicle information items is organized into a knotted-list structure containing forward as well as backward pointers. Four computer words are used to describe and define each vehicle entry in the VSTBL. No attempt is made to identify the lane the vehicle occupies, or to specify the vehicle's longitudinal position while within a sub-model or VSTBL list.

Typical vehicle entry

When a vehicle enters a sub-model, a four-word entry is made for that vehicle; the entry is placed in an appropriate position in the VSTBL list. It contains the following items of information:

1. Entry time into the system;
2. Entry time to the sub-model;
3. Estimated desired exit time from the sub-model;
4. The vehicle's origin-destination, i.e., the "route" it takes while in the system;
5. Its velocity at time of entry to the sub-model;
6. Its driver-type (and associated acceleration and deceleration factors);
7. A set of pointers indicating VSTBL addresses of vehicle entries adjacent to this entry.

Once a vehicle enters a sub-model and an entry for that vehicle has been made in the VSTBL list, no further processing of that vehicle takes place until it reaches the output boundary of its sub-model. The boundary is said to have been reached when the vehicle's desired exit time becomes less than or equal to the simulated time. Then the program determines whether it is possible to move the vehicle into the next sub-model along its route. If the vehicle can be advanced, its final velocity and exit time are calculated, elapsed sub-model travel time is recorded, the vehicle entry is removed from its present sub-model VSTBL list and inserted in the next sub-model VSTBL list (by changing appropriate address links). By this method, a vehicle "moves" through the system. Sometimes a vehicle is delayed and has to wait. Delays may be due to congestion because of heavy traffic volumes, because of a bottleneck that has been encountered, or because of insufficient distance (headway) between successive vehicles.

It might be of interest at this point to note the method used in determining a vehicle's desired exit time from a sub-model. Initially, the minimum travel time for a vehicle is calculated, as shown in Figure 5.
It is then added to the vehicle's VSTBL entry time and becomes the *earliest possible* exit time of that vehicle from its sub-model.

If the vehicle is entering a single-lane sub-model (where no passing may take place), then this time is compared with the d.e.t.* of the last vehicle in the VSTBL. If the d.e.t. of the new vehicle is less, it is changed to that of the last vehicle. If it is greater, it remains unchanged.

For a vehicle entering a multi-lane sub-model (where passing may take place), a different procedure is followed. First a determination is made of the passing probability of a new vehicle to that VSTBL. This information comes from an appropriate *passing probability table*, which has previously been established as a function of the occupancy of that VSTBL. Then, the d.e.t. of the new vehicle is compared with the d.e.t. of the vehicle immediately in front. If it is less, then a random number is compared with the value previously determined from the passing probability table. If the comparison is favorable, the new vehicle is allowed to pass the vehicle in front. Otherwise, the d.e.t. of the new vehicle is set to that of the vehicle that could not be passed and the new vehicle entry is "inserted" behind the vehicle entry that could not be passed. Thus a vehicle may pass none, several or all vehicles currently in the sub-model. The process is stochastic, and as expected from real life situations, the likelihood of passing decreases as a function of the occupancy of the sub-models.

Once the exit time for a vehicle has been established and the vehicle entry inserted to its VSTBL, no additional attempts are made to speed up or slow down the vehicle's travel time through the sub-model. This is taken care of automatically by subsequent events (conditions existing at the sub-model output boundary), vehicle flow rates, the stochastic passing process, the mix of driver types, and the allowable headway distribution.

Determining final velocity

Another area of interest is the determination of the *final velocity* (V_F) of a vehicle as it is about to cross the output boundary of its sub-model. Since the final velocity becomes the *initial velocity* (V_O) to the next sub-model being entered, its value is quite important. Figure 6 shows the logic of the algorithm.

*Desired exit time

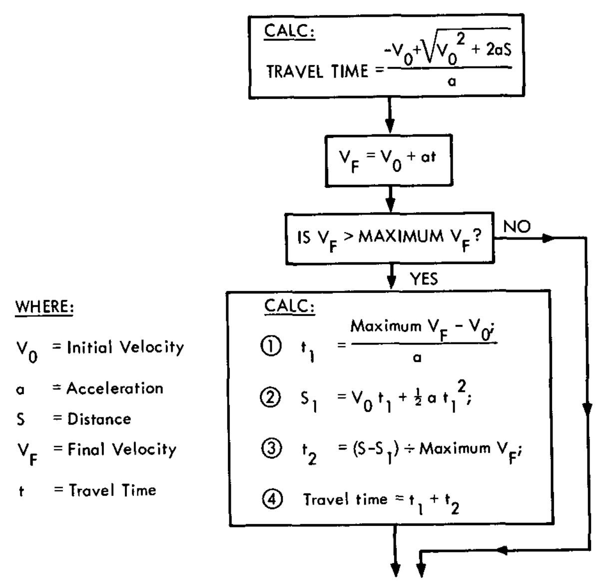

Figure 5 — Determining minimum sub-model travel time

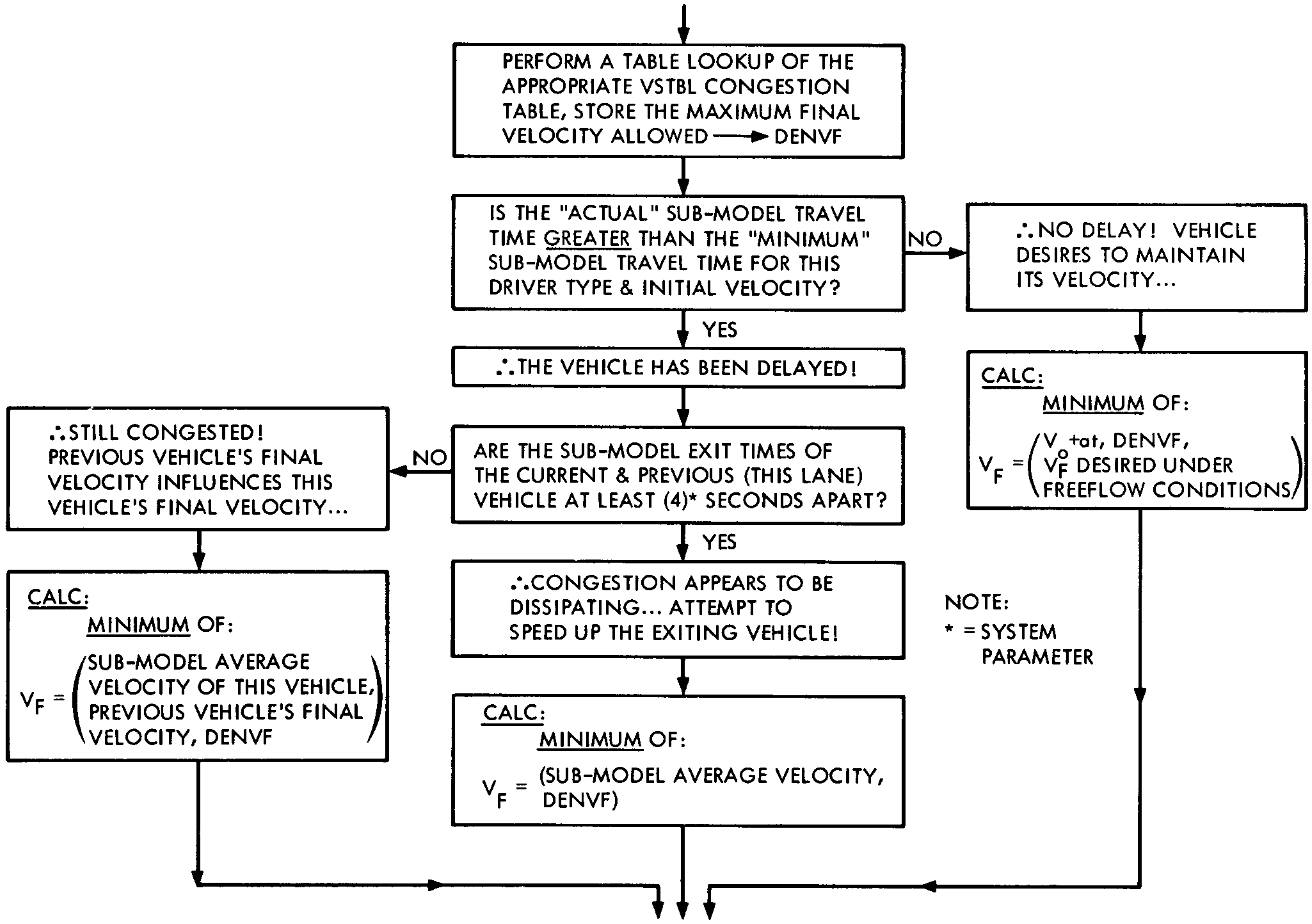

Figure 6 — Final velocity algorithm

When a vehicle finally leaves the system at V_{27}, additional traversal characteristics are recorded for later use by the data reduction routines. Then the four-word vehicle entry is returned to an available free storage space table (also list structured); it will be used again, as needed for new vehicles entering the system.

Typical VSTBL list

Each sub-model has its own VSTBL list structure. Associated with each list is a *header* word containing the addresses of the first and last vehicle entries in the list. The header is used for both entering and deleting vehicles. There are no imbedded sublists, as these were not found to be necessary. When an entry is "removed" from its list (showing that the vehicle has left the sub-model), the *first* address link in the header is changed to reflect the address of the next vehicle entry in the list. The contents of vehicle entries are not exchanged nor are they moved around in core.

When a vehicle is about to enter a sub-model, and is unable to pass other vehicles in the sub-model, its entry address becomes the *last* address link in the header word. The lists thus become circular in effect, and may be searched in either direction. By suitable change of address links or pointers, vehicle entries are connected or disconnected; the lists shrink and grow in size.

The lists are ordered by the estimated *desired exit time* of a vehicle entry from its sub-model. Entries with lower desired exit times are at the top of the list, closer to the output boundary of the sub-model. Those with higher desired exit times are further down in the list, farther away from the output boundary. An example of a 4-entry VSTBL list is shown in Figure 7.

To provide for passing within a sub-model, an algorithm containing a stochastic element was developed that decreases the likelihood of passing as a function of traffic density. To this end, a passing probability function is used each time a vehicle enters V_3, V_{26} and V_{27}, allowing a new vehicle to become the immediate last entry, the first entry (passing all vehicles) or an entry in between.

Operating characteristics

The model is cycled through time by a simulated clock, whose values are controlled by an executive routine. The length of a given time period or increment

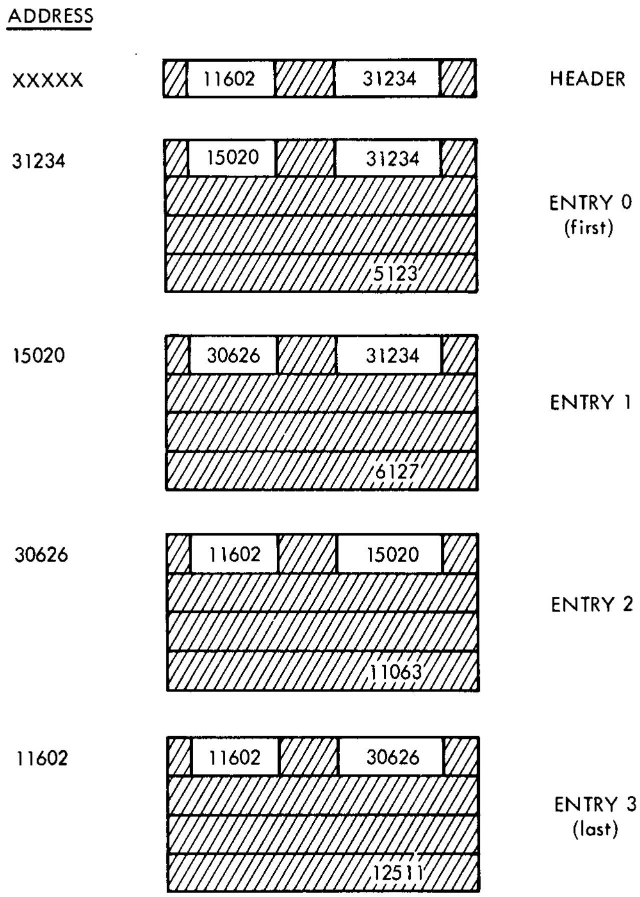

Figure 7 — Example of a 4-entry VSTBL list

is determined *a priori* by the initialization portion of the program and is a function of the length of the shortest sub-model in the system and the speeds of the fastest driver-types being investigated. The time period has usually been 1.5 seconds, for the particular interchange being studied — Roscoe Boulevard and the San Diego Freeway, in Los Angeles. Provision exists for dynamically varying the time-increment to as little as 0.1 seconds.

When a significant event occurs, such as the beginning of a new traffic signal phase, the time increments are adjusted so as to coincide with the change in phase. In this way, the complete effect of a given phase can be fully evaluated.

As each time period concludes, actual traversal characteristics are recorded and, in some cases, stratified by origin-destination. These traversal characteristics are used at the end of the simulation to compute various measures of system performance. They are compared to real-world observed data for purposes of validation. Selective recording (and future graphic display) are under programmer control via sense switches or toggle settings. In this manner, system performance may be measured in a number of ways for various purposes.

The functions that perform the simulation from the start to the end of a run have been broken down into four categories:

a. Initializing
b. Executive control
c. Logic routines
d. Ancillary routines, including data reduction.

The next few sections of this paper describe these in some detail.

Initializer functions

Prior to initiation of the executive function, certain basic tasks need to be performed. These are handled by an initializer routine. Among the tasks processed by the initializer are the following:

1. Determining the particular data reduction requirements for a run or set of runs and setting up the appropriate switch actions;
2. Establishing the various vehicle capacities of the sub-models and the time increment for the simulation;
3. Determining the traffic signal phase and cycle times;
4. Setting all initial values;
5. Causing the input datum to be read and the parameter values to be converted and stored;
6. Loading up the vehicle queues according to specified vehicle generating rates;
7. Forming the available free storage list of vehicle entries;
8. Calculating all table values;
9. Setting up the initial run number and starting and ending clock times.

Upon completion of its tasks, the initializer passes control to the executive routine, which then causes the simulation to begin.

Executive functions

Among the many functions performed by the executive are the following:

1. Updating of the system clock time;
2. Determining the occurrence of a significant event;
3. Adjusting the signal phase times;
4. Executing the several logic routines, causing vehicles to move into and through the system;
5. Giving the signal for recording of various traversal characteristics on a periodic basis;
6. When end-of-run time is reached;
   a. Causing data reduction to begin;
   b. Emptying any output buffers;

c. Determining the ratio of simulated time to computer time;
d. Reading in new input data value(s), as necessary;
e. Giving the signal for re-initializing, if additional runs are to be made.

Certain diagnostic routines may also be called upon to furnish dynamic tests of the model's performance while the simulation is occurring, i.e., to check if all is proceeding as planned. Appropriate messages may also be output indicating any faults that have been found. In addition, provision exists for aborting a run if a machine or program error is encountered. However well the executive operates can be said to be entirely due to the subroutines that it employs. These are of two types:

1. The *logic routines* for processing sub-models; these move vehicles into and through the system;
2. The *ancillary routines* associated with the logic routines; these include the classes commonly known as mathematical, data reduction, utility, diagnostic, housekeeping, etc.

The logic routines

These routines are complete in themselves. They call upon the various ancillary routines as needed. Within any particular logic routine, "looping" may occur to repeat a process or "nothing" may occur when no action is to take place. The basic role of the logic routines, described in great detail elsewhere,[1] is to determine the conditions under which vehicles may enter and/or leave a sub-model. When conditions are favorable, vehicles may move into or out of a sub-model, causing vehicle entries to be inserted into and deleted from the sub-model list structure. Various driving characteristics are determined at these times by sets of rules, which may vary from one sub-model to another, and which are aggregate in nature. As is expected, vehicle movements through the sub-models are highly dependent upon one another. A vehicle's final velocity and exit time from one sub-model are its initial velocity and entrance time for the next sub-model. When congestion occurs, entry time to a sub-model may be a function of exit time from that sub-model.

Upon conclusion of their operations, a standard VSTBL (sub-model) performance record is established by each logic routine, for the current time-period, and placed in appropriately indexed tables. The complete picture of the movements of vehicles into and/or out of the sub-models is thus recorded for later use by the data reduction and statistical programs.

The ancillary routines

As an adjunct to the logic routines, the ancillary routines are called upon most often in the course of a run to perform their assigned tasks. Some are used exclusively for vehicular movement, others for executive support and data reduction. Among those most commonly employed are the following:

1. Routines to calculate sub-model travel times;
2. Square root and log routines;
3. A routine to insert a vehicle entry in a list structure;
4. A routine to delete a vehicle entry from a list structure;
5. Routines to accumulate statistics on a periodic basis;
6. Routines to empty and subsequently refill vehicle queues for vehicles entering the system;
7. Input data read and conversion routines;
8. Routines that "get" a vehicle entry address from an available free storage list and "return" a vehicle entry address to the available free storage list;
9. A pseudo-random number generator;
10. Various diagnostic and error detection routines.

Data reduction

The data reduction routines convert, compute, and list variables for use by the experimenter. Several forms of output are available under program control. These listings can be considered to be of four types:

1. *Diagnostic* – used in program checkout and for dynamic testing of various program components;
2. *Vehicle performance records* — data taken at each time increment or multiple time increment, at sub-model breaks or at events;
3. *System snapshot* – summary of vehicle and sub-model performance at the end of specified time segments during the simulation;
4. *Statistical summaries* – aggregation of results at the end of the simulation.

A discussion and specimen page from each follows:

1. *Diagnostic.* This section of the computer program was designed primarily for use in checkout of the system. The output consists of a partial decoding of the information that is maintained for each vehicle while in its sub-model list structure. The printing of these records is under program control and may be executed at alternate time increments or at times specified in a parameter table. See Figure 8 for a specimen diagnostic page.
The first line reflects the time when the examination was made. The sub-model identification, its current density, and the first and last computer memory addresses used by vehicles in the sub-model are also printed. Examination of the first two columns of information will reveal the address linkage scheme used in the program. There is also

TIME (SEC)	VSTBL NO.	VEH COUNT	1ST ADDR	LAST ADDR
9.8	3	17	34766	34736

CORE ADDR	PREV ADDR	O.D.	INIT VEL	DRIV TYPE	SYSTEM ENTRY TIME	VSTBL ENTRY TIME	MIN XIT (SEC)	DET (SEC)	VSTBL DELAY (SEC)	DELAY BEYOND FREEFLO	NEXT ADDR
34766	34766	1	114	4	1.5	1.5	20.9	20.9	.0	.0	34772
34772	34766	1	95	2	1.0	1.0	24.3	24.3	.0	.0	34764
34764	34772	1	75	2	2.5	2.5	26.1	26.1	.0	.3	34770
34770	34764	1	85	1	1.2	1.2	27.2	27.2	.0	.0	34774
34774	34770	1	75	0	.5	.5	30.0	30.0	.0	.0	34760
34760	34774	1	85	1	4.2	4.2	30.2	30.2	.0	.0	34756
34756	34760	1	85	1	4.3	4.3	30.3	30.3	.0	.0	34754
34754	34756	1	95	2	5.0	5.0	28.3	30.3	2.0	2.0	34750
34750	34754	1	114	4	6.6	6.6	26.0	30.3	4.3	4.3	34744
34744	34750	1	95	2	7.8	7.8	31.1	31.1	.0	.0	34740
34740	34744	1	95	2	8.6	8.6	31.9	31.9	.0	.0	34762
34762	34740	1	75	0	3.0	3.0	32.5	32.5	.0	.0	34742
34742	34762	1	114	4	7.9	7.9	27.3	32.5	5.2	5.2	34752
34752	34742	1	95	2	5.6	5.6	28.9	32.5	3.6	3.6	34734
34734	34752	1	104	3	10.3	10.3	31.6	32.5	.9	.9	34746
34746	34734	1	95	2	7.0	7.0	30.3	32.5	2.2	2.2	34736
34736	34746	1	85	1	10.1	10.1	36.1	36.1	.0	.0	34736

Figure 8 — Specimen of a diagnostic printout

a column of desired exit times, and by a check of ascending desired exit times against memory address linkages, proper execution of the model can be assured. The remaining columns of information are printed to assist in an evaluation of the performance of the model.

2. *Vehicle Performance Records.* The vehicle performance record routine is presently operating at each time increment. The call to this section of the program is made after all logic routine processing has been completed. At that time, all information is available to compute and list the items associated with all sub-models. Provision has been made for stacking this information on a storage device for subsequent input to a more general data reduction and analysis program. This two-pass approach will also be used if the main program should require all of the computer. Each page reflects the changes in one sub-model and each time increment produces a line of information. Density and cumulative averages are computed for each time increment. See Figure 9 for a specimen page of a vehicle performance record.

The column of times represents each succeeding time period. Signal phases represent the six phases used to enter vehicles at the on-ramp of the freeway. The number of vehicles reflects the count of vehicles in the sub-model at the beginning of the time period. Occupancy is computed using the number of vehicles and the maximum number of vehicles that the specific sub-model could contain. The columns of vehicles entered and removed are totals for each time period.

Cumulative averages and totals for travel time and number of vehicles removed are shown in the next six columns. Averages and totals are broken down into the origin-destinations that apply in the specified sub-model. Dashes are printed for those origin-destinations not applicable for a sub-model. A count of the number of successful vehicle passing maneuvers is printed for each time increment. Dashes are printed for the single-lane sub-models where no passing occurs.

3. *System Snapshot.* After considerable checking and experimentation with the model, it became obvious that a less detailed output of performance records was needed. The snapshot program was devised to print at the end of a specified time interval. One minute is the usual interval, although this is variable. Only the most significant factors were chosen for output for each sub-model; they are printed at the end of successive time intervals. Most cumulative totals and averages are reset after each line of information has been printed. The snapshot method of output has proven to be a most useful technique for observing system performance. Figure 10 is an example of a system snapshot printout.

4. *Statistical Summaries.* A count is maintained for all possible velocities in each sub-model. At the end of each simulation run, a page of velocity frequency counts is printed for each sub-model. Each possible entry and exit velocity is printed

TIME	SIGNAL PHASE	# VEH	% OCUPNCY	VEH ENT	VEH REM	CUMULATIVE: AVG VSTBL TRT OD 1	OD 5,12	AVG SYS TRT OD 1	OD 5,12	VEH REMOVED OD 1	OD 5, 12	# VEH PASSED
65.6	1	2	7.1	4	2	1.9	1.8	31.6	23.6	49	6	2
67.0	1	4	14.3	0	0	1.9	1.8	31.6	23.6	49	6	0
68.4	1	4	14.3	4	4	1.9	1.8	31.7	23.6	53	6	0
69.8	1	4	14.3	4	4	1.9	1.8	31.9	23.6	57	6	0
71.2	1	4	14.3	0	0	1.9	1.8	31.9	23.6	57	6	0
72.0	2	4	14.3	4	4	1.9	1.8	32.0	23.6	61	6	1
73.4	2	4	14.3	0	0	1.9	1.8	32.0	23.6	61	6	0
74.8	2	4	14.3	4	4	1.9	1.9	32.1	24.9	64	7	1
76.2	2	4	14.3	4	1	1.9	1.9	32.1	24.9	65	7	0
77.6	2	7	25.0	4	3	2.0	1.9	32.2	24.9	68	7	0
79.0	2	8	28.6	0	4	2.0	1.9	32.3	24.9	72	7	0
80.4	2	4	14.3	4	4	2.1	1.9	32.5	24.9	72	7	0
81.0	3	8	28.6	0	4	2.0	1.9	32.4	24.9	76	7	0
82.4	3	4	14.3	4	4	2.1	1.9	32.5	24.9	80	7	2
83.8	3	4	14.3	4	0	2.1	1.9	32.5	24.9	80	7	0
85.2	3	8	28.6	0	4	2.1	2.0	32.5	25.0	83	8	0
86.6	3	4	14.3	4	4	2.1	2.1	32.5	25.3	86	9	1
88.0	3	4	14.3	1	0	2.1	2.1	32.5	25.3	86	9	0
89.4	3	5	17.9	2	4	2.1	2.1	32.4	25.5	89	10	0
90.8	3	3	10.7	0	3	2.1	2.1	32.4	25.7	91	11	0
92.2	3	0	.0	0	0	2.1	2.1	32.4	25.7	91	11	0
93.6	3	0	.0	0	0	2.1	2.1	32.4	25.7	91	11	0
95.0	3	0	.0	1	0	2.1	2.1	32.4	25.7	91	11	0
96.4	3	1	3.6	1	1	2.1	2.1	32.4	26.5	91	12	0
97.8	3	1	3.6	3	1	2.1	2.1	32.4	26.5	92	12	0
99.0	4	3	10.7	1	3	2.1	2.1	32.5	27.0	94	13	0
100.4	4	1	3.6	4	1	2.1	2.1	32.5	27.0	95	13	0

Figure 9 — Specimen of a VSTBL performance record printout

along with a count of the number of vehicles that entered and exited that sub-model with that velocity. Figure 11 is an example of these velocity frequency counts.

SYSTEM SNAPSHOT --- MOST CUMULATIVE VALUES ARE RESET EVERY 30 SECONDS

TIME (SEC)	VEH REMOVED OD1	OD5.12	AVG SYSTRT OD1	OD5.12	V3 OCPY	V3 REM	V3 AVTRT	V26 OCPY	V26 REM	V26 AVTRT	V25 OCPY	V25 REM	V25 AVTRT	V24 OCPY	V24 REM	V24 AVTRT	V23 OCPY	V23 REM	V23 AVTRT
30	3	2	26.5	25.4	7.9	8	22.3	4.9	6	3.6	3.8	2	7.3	8.6	0	.0	13.0	3	13.7
60	39	8	31.2	28.6	9.7	39	25.7	8.6	45	4.1	15.3	9	9.5	.0	3	13.7	8.6	9	14.3
90	82	16	33.1	30.1	8.2	45	27.2	12.3	50	5.1	19.2	8	9.8	4.3	2	14.5	26.0	7	14.7
120	114	29	33.3	31.5	6.7	38	26.8	18.5	47	5.2	15.3	14	11.1	4.3	2	15.4	8.6	11	14.6
150	152	40	33.7	32.5	11.2	34	25.6	12.3	48	5.9	19.2	9	10.6	4.3	2	16.3	17.3	8	14.0
180	202	49	33.7	32.6	10.4	50	25.7	13.5	60	5.5	.0	11	10.9	.0	1	16.3	4.3	5	15.5
210	253	55	34.0	32.7	8.9	46	27.2	9.8	53	5.2	.0	4	7.8	4.3	0	.0	17.3	4	12.5
240	291	62	33.9	32.0	11.7	45	25.7	17.2	48	4.5	11.5	9	7.6	4.3	3	13.0	8.6	9	13.2
270	336	73	34.0	32.0	10.7	51	27.5	17.2	60	5.7	19.2	9	9.6	.0	2	15.4	26.0	9	13.0
300	386	83	34.7	32.3	8.4	47	27.0	16.0	60	9.0	15.3	12	9.6	.0	0	.0	8.6	11	14.8
330	431	94	35.0	32.8	7.4	40	26.5	9.8	54	5.9	15.3	9	9.6	4.3	3	13.7	26.0	6	13.9
360	466	106	34.7	32.7	6.7	34	24.8	13.5	45	4.5	19.2	14	10.7	.0	3	13.6	8.6	12	14.3
390	502	119	34.6	32.5	9.2	30	25.8	7.4	46	4.8	11.5	11	9.5	8.6	4	13.4	17.3	5	12.5
420	538	128	34.6	32.4	8.7	42	26.3	13.5	48	4.8	7.6	11	10.6	.0	2	16.2	4.3	8	13.6
450	577	136	34.5	32.3	8.2	37	26.7	8.6	46	5.2	19.2	5	8.1	.0	3	14.3	30.4	5	13.0
480	610	147	34.5	32.2	8.9	34	26.4	12.3	45	4.7	19.2	14	10.3	4.3	1	15.3	8.6	13	15.2
510	653	160	34.5	32.4	7.9	41	26.0	7.4	56	6.0	19.2	11	9.6	4.3	1	12.5	21.7	10	13.0
540	691	174	34.4	32.0	9.7	33	24.3	1.2	52	4.7	3.8	14	9.1	.0	1	12.5	4.3	9	12.6
570	730	180	34.3	31.9	8.9	44	26.7	11.1	41	4.5	11.5	5	8.0	.0	2	16.3	34.7	5	13.2
600	773	190	34.2	32.0	8.7	43	25.2	12.3	54	4.9	11.5	12	10.8	.0	1	14.3	4.3	11	15.3
630	813	198	34.1	32.0	7.7	44	25.0	11.1	51	4.7	19.2	6	8.7	.0	3	14.3	30.4	5	13.0
660	850	209	34.1	32.0	8.9	35	26.8	9.8	49	4.7	19.2	13	10.6	.0	0	.0	8.6	13	15.7
690	891	220	34.1	32.1	7.2	40	27.2	7.4	51	5.1	23.0	9	10.6	4.3	2	12.1	21.7	8	14.1
720	926	232	34.0	32.1	5.7	33	24.4	7.4	46	4.6	23.0	13	10.9	8.6	2	13.6	4.3	11	14.7
750	958	244	34.0	32.2	10.4	28	24.5	2.4	44	5.0	15.3	12	10.7	4.3	3	13.0	26.0	7	14.3

Figure 10 — Printout of a system snapshot

VSTBL 3 VELOCITY FREQUENCY COUNTS

VEL	INITIAL	FINAL	VEL	INITIAL	FINAL	VEL	INITIAL	FINAL	VEL	INITIAL	FINAL
0	0	0	32	0	0	64	4	1	96	0	0
1	0	0	33	0	0	65	0	1	97	1	0
2	0	0	34	0	0	66	0	0	98	1	0
3	0	0	35	0	0	67	1	15	99	0	1
4	0	0	36	0	0	68	0	0	100	0	0
5	0	0	37	0	0	69	0	3	101	0	0
6	0	0	38	0	0	70	15	27	102	0	0
7	0	0	39	0	0	71	11	0	103	3	0
8	0	0	40	0	0	72	8	1	104	3	19
9	0	0	41	0	0	73	10	4	105	0	0
10	0	0	42	0	0	74	40	0	106	0	0
11	0	0	43	0	0	75	99	110	107	0	0
12	0	0	44	0	0	76	10	0	108	0	0
13	0	0	45	0	0	77	24	0	109	0	0
14	0	0	46	0	0	78	10	2	110	0	1
15	0	0	47	2	0	79	4	0	111	0	0
16	0	0	48	0	0	80	20	10	112	0	0
17	0	0	49	0	7	81	9	15	113	0	0
18	0	0	50	0	10	82	5	0	114	1	5
19	0	0	51	0	0	83	4	5	115	0	0
20	0	0	52	0	43	84	15	0	116	0	0
21	0	0	53	0	1	85	81	66	117	0	0
22	0	0	54	2	0	86	0	0	118	0	0
23	0	0	55	0	10	87	2	7	119	0	0
24	0	0	56	0	2	88	1	0	120	0	0
25	0	0	57	6	1	89	6	0	121	0	0
26	0	0	58	0	6	90	1	0	122	0	0
27	0	0	59	1	5	91	0	0	123	0	0
28	0	0	60	0	10	92	0	0	124	0	0
29	0	0	61	0	5	93	5	0	125	0	0
30	0	0	62	0	7	94	0	4	126	0	0
31	0	0	63	0	0	95	20	30	127	0	0

Figure 11 — Summary printout of VSTBL velocity frequency counts

Another type of output is also available to the experimenter. This is the sequence of vehicles leaving the system. This information was used to make early comparisons between measurements of real traffic flow taken from a freeway interchange and the simulated vehicular flow from the model. Figure 12 is an example of the sequence of vehicles as they leave the system.

The page of output shown in Figure 13 is *always* produced at the end of each simulation run, in addition to any other kind of output desired. It indicates some of the variables and parameters currently considered in Model 1, including vehicle generating rates, on-ramp signal phase times, driver-type distributions, maximum freeflow velocities, acceleration factors, etc.

Future plans

The current version of Model 1 was coded in assembly language for the Philco 2000 digital computer. The next versions are to be coded in JOVIAL and/or PL/I, higher order languages. Not only will this facilitate transferability to other computers, it will relieve the burdensome details associated with machine coding. Also, it will be interesting to note the differences in running time, storage requirements, and checkout ease, resulting from the use of different programming languages.

It is planned to incorporate a variable input data format routine which will be capable of processing any expected future input data, without causing additional reprogramming effort, due to fixed-field input data allocation. Model I contains over 50 different system variables and parameters whose values (any one of which, or none) may change from one experiment to another. Successive models are expected to be even more comprehensive in their system specifications and requirements. A compiler-type of input source string character scan will be employed to recognize such things as identifiers, punctuation symbols, and numeric values on the input device (card, tape, disc, etc.). Searching of an identifier table to recognize the identifier and its data type (i.e., floating-point, fixed-point, or integer) then will enable proper conversion and storage of the respective values. Standard, built-in values will be used for those identifiers not encountered. This method, being open-ended and modular, thus provides for inclusion of new variables, data types, etc., as they are required, by suitable extension of the identifier table. To the author's knowledge, this technique is not used in other vehicular traffic digital computer simulation models.

Since it can not be empirically determined beforehand how long any given experiment or computer run should take (for the system performances being measured), a two-way *interrupt* capability will be provided. This will allow the state of the model at the time of the interruption to be saved on a storage device (tape, drum, disc, etc.) and labeled. Operations could then be continued, if desired, under an appropriate *restart* procedure. An interrupt may be caused by either of two methods; an operator action at the computer console, or a data input value that causes the interrupt to occur at a specified time. It should prove very beneficial to be able to resume a run at the interrupt point, after perusal of the output up to the time of the interrupt.

Finally, a more extended capability will be programmed for another version of Model 1; the improved model will be able to handle more adequately such things as shock wave build-up and dissipation, variable headway distributions, etc. Continuation of the series of models will be dependent on field validation and significance of the procedures used for aggregating and moving vehicles.

VEHICLES LEAVING THE SYSTEM

EXIT TIME (SEC)	SEQ NO.	SYSTEM TRAVEL TIME	DRIVER TYPE	AVG VELOC (MPH)	AVG VELOC (FPS)	O.D.	CUMAVG O.D. SYSTRT
27.0	1	26.5	3	70.6	103.5	1	26.5
28.9	2	28.9	2	64.7	94.9	1	27.7
30.5	3	28.5	1	39.0	57.2	12	28.5
31.8	4	26.5	3	70.6	103.5	1	27.3
32.5	5	32.3	1	57.9	84.9	1	28.5
32.6	6	29.0	2	64.5	94.6	1	28.6
32.6	7	26.8	3	69.8	102.4	1	28.3
34.5	8	32.7	1	57.2	83.9	1	28.9
34.5	9	27.2	3	68.8	100.9	1	28.7
34.6	10	25.4	4	73.6	108.0	1	28.3
34.9	11	32.5	1	57.5	84.4	1	28.7
36.8	12	26.5	3	70.6	103.5	1	28.5
37.6	13	32.3	1	57.9	84.9	1	28.8
38.5	14	32.3	1	57.9	84.9	1	29.1
38.5	15	30.1	2	62.1	91.1	1	29.2
39.1	16	28.9	2	64.7	94.9	1	29.1
43.0	17	28.9	2	64.7	94.9	1	29.1
43.9	18	28.4	3	65.9	96.6	1	29.1
43.9	19	30.5	4	61.4	90.0	1	29.2
45.0	20	32.3	1	57.9	84.9	1	29.3
45.0	21	29.0	2	64.5	94.6	1	29.3
46.3	22	23.3	3	47.7	70.0	5	25.9
46.4	23	29.8	2	62.8	92.1	1	29.3
47.0	24	32.7	1	57.2	83.9	1	29.5
51.8	25	30.3	2	61.7	90.5	1	29.5
51.8	26	34.2	3	54.7	80.2	1	29.7
52.0	27	33.0	2	56.7	83.1	1	29.8
53.3	28	36.6	0	51.1	75.0	1	30.1
54.2	29	29.2	1	38.0	55.8	5	27.0
54.2	30	32.8	3	57.0	83.6	1	30.2
54.6	31	33.5	1	55.8	81.9	1	30.3
55.3	32	30.4	2	61.5	90.2	1	30.3
56.2	33	33.1	1	56.5	82.9	1	30.4
57.4	34	29.7	2	63.0	92.4	1	30.4

Figure 12 — Sequence of vehicles leaving the system

VEHICULAR TRAFFIC STUDY PARAMETERS

FREEWAY - DIAMOND INTERCHANGE..MODEL 1

10-01-65 -- VERSION 01 -- RUN 02

	FREEWAY		POCKET LANE	--------------CURB LANE---------------	
VEHICLE GENERATING RATES PER HOUR	= 3800		(GREEN) = 600	(GREEN) = 54C	(RED) = 60
V23 ENTRY PROBABILITIES			= .778	= .872	= .745
VSTBL LENGTHS (FT)	V3 = 2210	V23,24 = 506	V25 = 590	V26 = 360	V27 = 175

SIGNAL PHASE TIMES (SEC)	DISTRIBUTION OF DRIVER-TYPES (%)	FWY FFVF (FT/SEC)	FWY A-FCT (FT/SEC/SEC)	ON-RAMP VO (FT/SEC)	ON-RAMP A-FCT (FT/SEC/SEC)
1 = 12	SLOW = 07	75	3.0	6	3.0
2 = 09	MEDIUM = 24	85	4.5	11	3.5
3 = 18	NORMAL = 38	95	6.0	16	4.0
4 = 09	FAST = 24	104	7.5	21	4.5
5 = 06	VERY FAST = 07	114	9.0	26	5.0
6 = 06					

FREEFLOW SYSTEM TRAVEL TIMES (SEC)

DRIVE TYPE	O.D. 1	O.D. 5.12
0	36.6	32.2
1	32.3	28.5
2	28.9	25.4
3	26.5	23.2
4	24.1	21.1

AVG VEH LENGTH + GAP (FT)	HEAD WAY (SEC)	RUN TIME (MIN)	TINC TIME (SEC)
22	2.0	25	1.5

% OCCUPANCY =	5	10	15	20	25	50	100
PROBABILITY =	.700	.400	.100	.080	.050	.010	.000

TABLES USED TO DETERMINE VELOCITIES DURING "CONGESTION"

V3 to V26		V23,24 TO V25		V25 TO V26		V26 TO V27		QUEUES TO V3		& TO V23, V24	
OCPY (CT)	VELOC (FT/SEC)	OCPY (CT)	VELOC (FT/SEC)	OCPY (CT)	VELOC (FT/SEC)	OCPY (CT)	VELOC (FT/SEC)	OCPY (CT)	VELOC (FT/SEC)	OCPY (CT)	VELOC (FT/SEC)
50	75	13	63	50	72	12	75	370	75	10	24
55	64	15	53	55	61	17	64	375	64	12	20
60	53	17	43	60	50	22	53	380	53	14	17
65	42	19	33	65	39	27	42	385	42	16	13
69	31	21	23	69	28	32	31	390	31	18	10
73	21	23	13	73	17	37	20	395	21	20	6
77	10	25	3	77	6	42	9	400	10	22	3

Figure 13 — Prologue — showing run parameters

REFERENCES

1 A V GAFARIAN E HAYES W W MOSHER JR
The development and validation of a digital simulation model of freeway diamond interchanges
SP-2159/000/00 April 1966

2 D L GERLOUGH
Simulation of freeway traffic by an electric computer
Proceedings of Highway Research Board vol 35 pp 543-547 1956

3 H H GOODE C H POLLMAR J B WRIGHT
The use of a digital computer to model a signalized intersection
Proceedings of the Highway Research Board vol 35 pp 548-557 1956

4 W HELLY
Simulation of bottlenecks in single lane traffic flow
Proceedings of Highway Research Board pp 205-238 1959

5 P PERCHONOK S L LEVY
Application of digital computer simulation techniques to freeway on-ramp traffic operations
Proceedings of Highway Research Board vol 39 pp 506-523 1906

6 M C STARK
Computer simulation of nine blocks of a city street
Highway Research Board Bulletin 356 pp 40-47 1962

7 D L GERLOUGH
Simulation of the network
Proceedings of the Conference of Traffic Surveillance, Simulation and Control pp 152-165 1964