Discussed are methods underlying the real-time monitoring and controlling system of a critical traffic link.

Algorithms were developed for recognizing the length patterns of vehicles passing over detectors and using these for precisely computing traffic density in several sections of the Lincoln Tunnel. Vehicle control was adjusted by the system to optimize tunnel throughput.

Real-time traffic flow optimization

by B. C. Black and D. C. Gazis

Seeking more efficient methods of controlling and increasing the flow of vehicular traffic through a critical traffic link, the Port of New York Authority and IBM conducted a study and relevant experiments using the Lincoln Tunnel as a test site. The study had led to new methods of measuring and controlling traffic, thereby increasing the tunnel throughput by real-time computer techniques.

In this paper, we concentrate on the underlying methodology of traffic monitoring and control programming for the real-time optimization of traffic flow. Since the traffic engineering aspects of this study have been discussed elsewhere,[1-3] we review the experimental environment as background information only.

Experimental environment

Early measurements had shown that the tunnel's throughput decreases rather drastically when the average traffic density exceeds about fifty cars per lane mile. Of course, the throughput is also low when the density falls much below that level. One of our objectives was to identify optimum density levels and to develop methods to maintain the internal traffic close to these levels during peak periods by controlling the input. The measurement of densities is by no means a simple task. Average

density measurements obtained from phenomenological flow versus speed relationships were found to be inaccurate by as much as a factor of two too large or too small. To effectively control peak traffic, accurate estimates of traffic density in several sections of the tunnel were required. Because of this more "microscopic" requirement on the data and other circumstances of the experiment, known traffic control programs, such as the IBM 1800 Vehicular Traffic Control System,[4] could not be readily applied to this experiment.

detectors

The first step in controlling traffic density in the tunnel required that each vehicle be detected and its progress through the tunnel be monitored. For traffic detection, the tunnel was divided into three approximately half-mile sections by four observation points, or "traps," as shown in Figure 1. Located at the entrance, at the foot of the downgrade, at the foot of the upgrade and at the exit, each trap consisted of two detector photocells placed in the pavement approximately fourteen feet apart and aimed at high intensity lights on the tunnel wall.

The passage of vehicles over detectors produced voltage pulses that were converted into sound-frequency signals by the tone transmitters shown in Figure 2. Tone multiplexing was provided by up to sixteen transmitters having interband distances of 120 cycles and tone shifts of 60 cycles. The data was transmitted over a single telephone line to a computing center located about forty miles from the experimental site. There the data was unscrambled by matching tone receivers and fed into a special interface.

interface

This interface, shown in the overall system in Figure 2, received the tone signals, converted and stored them as bits in registers, and transmitted the register contents to the computer. This interface consisted mainly of two status registers (new and previous), a time register, a time counter, and a clock. Data from the receivers entered the new-status register where it was sampled every millisecond. Sampling consisted of comparing the register that contained new input data, with the register that contained the previous data sample. If the contents of the two registers were the same, the only action taken was the updating of the time counter. When the comparison of the two status registers indicated a change in the status of any detector, the contents of the new-status register and the time counter were transmitted to the Central Processing Unit (CPU). At the same time, these register contents updated the previous-status register and the time register, respectively. Data transmitted from the interface to the CPU indicated the status of all the traffic detectors and the time in milliseconds corresponding to changes in status of one or more detectors. Sampling by the interface continued while the CPU processed the newly transmitted data.

Figure 1 South tube of the Lincoln Tunnel

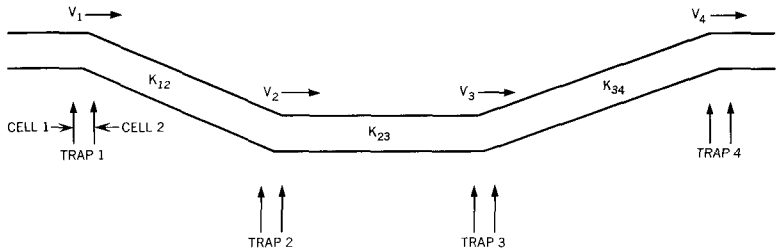
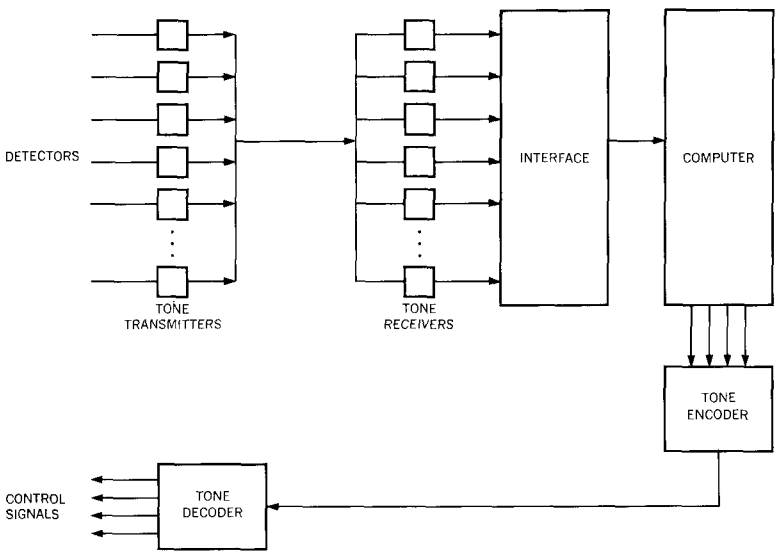


Figure 2 Traffic control system



computing system

Our experiment was conducted in two phases that are differentiated by the computers used. During the first phase, which took place during most of 1967, we used an IBM 7040. For the second phase, during the first half of 1969, the IBM 1800 Process Control System was used. In the first phase, the computing system provided an interesting test of the use of a batch-processing, scientific computer for performing an on-line process control function. Our method was basically to insert a process (traffic) control program into the system program of the 7040 computer of a 7040/7094 Direct Coupled Data System. Thus the batch system periodically operated as a real-time process controller for a fraction of the time available.

To facilitate computer input/output in the 7040/7094 system, a data channel was added to the computing system configuration to transmit the tunnel data from the interface to the computer. Also, an output line was connected from this channel to an en-

coder for the transmission of traffic control signals back to the tunnel after the computations had been performed. No other additions or modifications to the hardware configuration of the system were necessary.

The traffic control program performed traffic monitoring and controlling functions and resided on tape, together with the system program. This control program was treated as any of the other channel-servicing programs of the system. When the control program was being run (at five-second intervals), it was essentially one of many functions in a multiprogrammed and time-shared environment.

During the second phase of the experiment, the Time-Sharing Executive System (TSX) was the operating system of the 1800. No modifications to TSX were necessary. However, the object program of the tunnel control experiment resided permanently in main storage during execution because of the requirement of processing at closely spaced regular intervals, usually five seconds. Input signals went directly from the interface into the 1800, and output from the 1800 was transmitted to the encoder as shown in Figure 2.

A useful feature of the traffic control program during both phases was its ability to change parameters while the program was being executed. This feature provided the capability of changing the traffic control algorithm and experimenting with different values of program variables. Also, control commands could be overridden using this modification technique.

Traffic monitoring

To accomplish its two primary functions of monitoring traffic conditions and controlling tunnel inputs, the program executed the following four rather distinct operations:

- Vehicle identification
- Initialization and updating of vehicle counts in three sections of the tunnel
- Control decision
- Transmission of control signal and storage of data

In this section, we discuss vehicle identification and initialization of counts, which are operations that are also required for regulating other traffic systems. The latter two operations are peculiar to the Lincoln Tunnel, and are presented in the following section. It might be mentioned that the programming for all of these tasks emphasized economy of storage and computer time, in anticipation of the expected time-shared operation of the com-

puter. Although our discussion of the implementation of the four tasks of the traffic control program is generally applicable to both phases of the experiment, we emphasize the second or 1800 system phase.

vehicle identification

Vehicle presence, the essential and most elemental system input, took the form of data words generated at the interface, indicating the blocked or unblocked condition of the photocells. A change in the status of the cells was detected by a comparison of the present data word with the data word immediately preceding it. The alphabetic labels in Tables 1 and 2 were given to each event associated with the passage of a vehicle over the traps. Table 1 summarizes the blocked and unblocked conditions of the two photocell detectors of a trap for single-vehicle events. Table 2 applies combinations of events in Table 1 to two-vehicle events. Multiple-vehicle events are concatenations of those in Table 2. Event F was a special case when the vehicle was exactly the length of the trap. Events C and G were very rare, and were generally produced not by two separate vehicles but by two pieces of the same vehicle, such as a cab-trailer combination.

Table 1 Single-vehicle events

Label	*Action*
A	Front blocks cell 1
D	Front blocks cell 2
B	Rear unblocks cell 1
H	Rear unblocks cell 2

The patterns of electrical pulses produced by the passage of vehicles over the photocells of a trap were used for the identification of the vehicles. Figure 3 shows some of the most common pulse patterns. For example, a standard car (approximately 17 feet long and in one piece) produced the A–D–B–H sequence with corresponding times T_1, T_2, T_3, T_4 as it passed over a trap. Because the vehicle was longer than the trap length, this sequence is effectively a pair of overlapping square pulses of almost equal duration, as shown in Figure 3A.

Vehicles shorter than the trap length produced the A–B–D–H sequence since the front and rear ends passed over cell 1 of the trap before the front end blocked cell 2. As shown in Figure 3B, the square pulses thus created are almost equal but nonoverlapping.

A vehicle whose length was exactly equal to that of the trap produced the A–(F→BD)–H sequence. The front end entering cell 2 and rear end leaving cell 1 were simultaneous events.

Table 2 Two-vehicle events

Label	*Action*	
	Blocking cell 1	Blocking cell 2
I → AH	Front of 2	Rear of 1
C → BH	Rear of 2	Rear of 1
G → AD	Front of 2	Front of 1
F → BD	Rear of 1	Front of 1

Figure 3 Wave patterns for common vehicle types

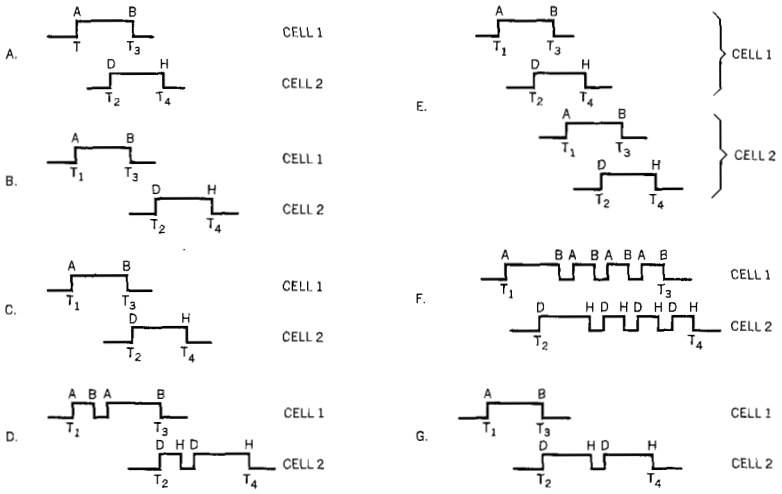


Figure 3C shows the resultant equal square pulses, and time T_2 equal to T_3.

A truck with a cab and trailer produced the A–B–A–D–B–H–D–H sequence shown in Figure 3D. Square pulses correspond to each piece of the truck passing over the photocells, and some overlapping is also evident. The pulses were logically merged by the program to obtain the times T_1, T_2, T_3, T_4 as shown.

Figure 3E shows a pattern indicating a possible tailgating situation. Tailgating is defined as the existence of a small headway between car 1 and car 2 such that car 2 enters the trap before car 1 has left it. The square pulses shown in Figure 3E can also be produced by two parts of the same car. Therefore, a test based on some minimum practical time-headway between individual vehicles was made by the program to determine if there were indeed a tailgating situation.

Figure 3F shows pulses produced when a truck with a cab pulling an empty ladder-type platform passed over a trap. The sequence of short pulses resulted when each part of the platform blocked the light beam to the photocells.

Some vehicles appeared to be variable in shape as they passed from one detector to the other. Figure 3G shows one square pulse being produced when the vehicles passed over cell 1, but more than one pulse being produced when it passed over cell 2. This pattern resulted when the light beam was blocked by links connecting the parts of the vehicle as it passed over one of the photocells but not the other. Here again, it was necessary to

logically merge the pulses to obtain the appropriate T_1, T_2, T_3, and T_4.

vehicle length and speed

The values of the four times T_1, T_2, T_3, and T_4 were used to compute the length and speed of each vehicle. (Acceleration was assumed to be constant over a given trap.) The speed was computed as

$$V = L/T_{34} \tag{1}$$

and the vehicle length as

$$\ell = L\left(\frac{T_{13}}{T_{13}+T_{24}}\right)\left(\frac{T_{14}}{T_{12}}+\frac{T_{23}}{T_{34}}\right) \tag{2}$$

In equations 1 and 2, L is the trap length in feet and time differences are given by

$$T_{ij} = T_j - T_i$$

Accurate determinations of the speeds and lengths of vehicles were essential because the group-average speed of the last five vehicles was used in the control algorithm, and the pattern of successive lengths of vehicles was used in initializing and updating the vehicle counts in each section of the tunnel.

initializing and updating

In initializing and updating vehicle counts, we began by letting $(K_{ij})_n$ denote the number of vehicles in a section of the tunnel between trap i and trap j at the end of the nth observation period. Measurements for two successive observation periods should then satisfy the vehicle-count relation

$$(K_{ij})_n = (K_{ij})_{n-1} + (N_i)_n - (N_j)_n \tag{3}$$

where N_i and N_j are the numbers of vehicles passing over traps i and j, respectively.

In principle, if we knew the initial number of vehicles in a section, we could have computed the number for all subsequent periods by a suitable updating procedure. However, there were factors that prevented this principle from being realized in practice. Lane changing against the rules and sometimes during the suspension of the rules—such as stoppages and other emergencies—were the most common of these factors. It was necessary, therefore, not only to initialize the count at the beginning of a run, but also to reinitialize frequently during a run, rather than depend solely on updating by Equation 3. The method used for initializing vehicle counts K_{ij} involved matching the patterns of relative lengths of sequences of vehicles that passed over two successive traps, i and j.

Our procedure was first to represent these relative lengths by bit patterns stored in registers assigned to the traps. The system then tried to match a specified number of consecutive bits of the

Table 3 Bit patterns for length differences of successive vehicles

Regions	*Registers* R_i	RS_i	*Difference*
3	1	1	large positive
2	1	0	small positive
1	0	1	small negative
0	0	0	large negative

registers corresponding to the two traps, i and j. The relative shift of the registers required to attain such a match is related to the number of cars between these traps.

The first step of the initializing and counting procedure was to compute the difference in length Δ_k of the kth and $(k-1)$st vehicle passing over a given trap by using the relationship

$$\Delta_k = \ell_k - \ell_{k-1} \tag{4}$$

For trap i, the bits indicating the sign and magnitude of Δ_k were stored in two registers, R_i and RS_i, as follows:

The sign of the difference was stored in R_i, so that

If $\Delta_k \geq 0$, store 1
If $\Delta_k < 0$, store 0

The magnitude of the difference was recorded in RS_i as follows:

If $\Delta_k \geq \varepsilon$, store 1
If $0 < \Delta_k < \varepsilon$, store 0
If $0 > \Delta_k > -\varepsilon$, store 1
If $\Delta_k < -\varepsilon$, store 0

Here ε was equal to some small positive quantity such as 0.5 feet. The four possible bit patterns and their meanings are summarized in Table 3. Bits taken in the same order as registers R_i and RS_i may be viewed together as the binary number corresponding to 3 to 0 in the four regions 3 to 0 of Table 3. When processing was done on the 7040/7094 system, the two middle regions, 2 and 1, were combined into a single region.

It was not necessary for the registers to be filled before matching could take place. Thus, if K_{ij} were the number of cars between traps i and j, it was necessary to wait only until the registers for trap j had records of at least $M + K_{ij}$ vehicles (where M was the number of bits we wanted to match). In practice, attempts at matching were started when about fifty vehicles had crossed each trap.

In principle, the pattern of length differences made by a given set of vehicles passing over trap i should be the same as that produced by the same vehicles passing over trap j. However, in addition to small errors resulting from the constant-acceleration assumption in the vehicle-length calculation, other errors were created by the following conditions:

A. Differences in vehicle position with respect to the photocells in successive traps due to nonrectangular vehicle profiles
B. Lane changes
C. Incorrect assessment of possible tailgating situations

The errors caused by conditions B and C necessitated reinitialization. Small errors introduced by A were compensated by recognizing that a difference Δ_k in Equation 4, which fell into one of the four regions of Table 3 when observed at one trap, could fall into one of the adjacent regions when observed at the next trap. By our register-matching rule, a "match" occurred when a binary number of a Table 3 region in registers R_i and RS_i (for trap i) differed by not more than ± 1 from the same region in R_j and RS_j (for trap j).

The implementation of the register-matching procedure during initialization comprised two steps: (1) shift each register for trap i, one bit at a time, in the direction opposite to that in which the bits were inserted, and (2) attempt to match the leading M bits of these registers with the corresponding bits of the registers for trap j. The M bit positions were known as a mask. For initialization, a "strong mask" ($M = 30$) was used, and a match of the bit patterns had to satisfy the following relationship:

$$(R_i \text{ .EOR. } R_i) \text{ .AND. } [(R_i \text{ .EOR. } RS_j) \text{ .OR. } (RS_i \text{ .EOR. } R_j)] = 0 \qquad (5)$$

Here, .EOR. is the logical EXCLUSIVE OR operator; .AND. is the logical AND operator; and .OR. is the OR operator.

The statement of Equation 5 implies the rule of matching given previously. If Equation 5 was not satisfied after shifting up to sixty-four-M positions, the initialization of K_{ij} was postponed until the next observation period when the matching of a new set of bits was attempted. When Equation 5 was satisfied, the number of shifts of the registers for trap i required for this match was exactly equal to K_{ij}. When updating a previously computed K_{ij}, a new K_{ij} was computed according to Equation 3. Then the registers for trap i were shifted by K_{ij} positions, and the leading M bits were compared with the leading M bits of the registers for trap j using Equation 5 and a "weak mask" ($M = 20$ bits). If no match occurred, the computed K_{ij} was retained conditionally for control purposes. In this case however, a reinitialization procedure was started during the same observation period using a strong mask ($M = 30$ bits). A K_{ij} obtained during reinitialization

immediately replaced the updated K_{ij}. A strong mask was used when initializing K_{ij} because the probability of a random match of the bit patterns decreased as the number of bits M increased.

Control algorithm

We now discuss the control algorithm, which was the second major phase of our traffic control program. The two controlled lanes were considered reasonably independent of each other and were therefore controlled independently on the basis of their observed traffic characteristics. The state of each lane was adequately described by three state variables, which were the vehicle counts in the three sections of the tunnel. Thus the vehicle counts K_{12}, K_{23}, K_{34} for the near lane, indicated in Figure 1, and K_{56}, K_{67}, K_{78} for the far lane specified the state space. Speed measurements V_i (where $i = 1$ to 4 for the near lane—shown in Figure 1—and 5 to 8 for the far lane) were used for an adaptive feature of this algorithm.

Observations of the tunnel were used to divide the state space into regions corresponding to increasing probability of impending congestion in the absence of constraint on the traffic input. A family of surfaces was constructed in state space so that the crossing of a surface signalled a critical change of vehicle counts and necessitated a change in control requirements. Such control was manifested by four different control intensities, represented by the function C that was computed for each lane. This function took the values 0, 1, 2, 3, 4, and corresponded to the following control actions: Green, Amber 1, Amber 2, Amber 3, and Red.

control function computation

Using the subscripts for the near lane, we illustrate the computation of the control function. If the speeds V_2 and V_3 (in feet per second) are such that $V_2V_3 > 2500$, then $C = 0$ (green). Otherwise compute the function F, where

$$F = g(V_3)(P_{12} - K_{12})(P_{23} - K_{23})(P_{34} - K_{34}) \tag{7}$$

We use F in the following computation of the control function:

$$C = \sum_{k=1}^{4} H(A_k + \mu D_k - F)$$

Here, A_k, D_k, and P_{ij} are constants, and H is the step function given by

$$H(x) = \begin{cases} 0 & \text{if } x < 0 \\ 1 & \text{if } x \geq 0 \end{cases}$$

The quantity μ is a history-dependent variable that takes the value 0 in computing ascending values of C, and becomes 1 in computing descending values of C. The function $g(V_3)$ was an

Table 4 Traffic control levels and the signals displayed

Mnemonic	*Control level*	*Signal*
NC	No control	Green light
C1	1	Flashing amber light
C2	2	Amber light, "PAUSE HERE THEN GO" sign, and lights flush with the pavement
C3	3	Control lights of C2 plus raised cones to reduce lane width
C4	4	Red light

adaptive correction that, in effect, accounted for undetected differences in traffic conditions. This correction intensified the control when the speed at the foot of the upgrade, the most usual bottleneck, was lower than the measured densities warranted. The function $g(V_3)$ was given by

$$g(V_3) = 1 + (V_3 - 50)/500$$

Typical values of the constants for the near lane were:

$$P_{12} = P_{23} = 67$$
$$P_{34} = 66$$
$$A_1 = 64{,}000$$
$$A_2 = 56{,}000$$
$$A_3 = 46{,}500$$
$$D_1 = D_2 = D_3 = 6{,}150$$
$$D_4 = 8{,}200$$

The evaluation of these constants is discussed in Reference 1. A similar computational procedure was used for the far lane, wherein slightly different values of the parameters accounted for variations in the composition of the traffic between the far lane and the near lane.

control signals

When only one lane was monitored and controlled, signals for that lane were duplicated in the far lane. Initially, the control choice was binary. This means that the options were those of imposing no restriction on input to the tunnel (green light on) or imposing an input-restraining control of fixed intensity. Later, the control algorithm produced five control options for five different control levels of increasing intensity. These options, corresponding to the four values of the control function previously discussed, plus "no control", are given in Table 4.

When both the near and the far lanes were being controlled, the control signals transmitted were limited to five so as to permit

Figure 4 Sample output recorded near the beginning of a run

TIME	K12	K23	K34	V1	V2	V3	V4	N1	N2	N3	N4	CN	K56	K67	K78	V5	V6	V7	V8	N5	N6	N7	N8	CF
05.53.38				30	40	36	34	14	11	9	7	NC				37	44	15	25	9	9	6	9	NC
05.54.07				21	43	39	38	8	12	11	10	NC				28	42	7	24	6	6	5	8	NC
05.54.39				23	22	46	33	9	5	8	10	NC				28	32	6	34	7	10	1	11	NC
05.55.08				32	25	40	24	13	12	8	11	NC				33	45	11	29	7	9	4	7	NC
05.55.37				34	25	37	30	14	10	9	11	NC				21	41	15	32	9	5	7	8	NC
05.56.09				26	26	43	52	12	13	13	10	NC				22	41	16	35	9	7	7	12	NC
05.56.38				28	18	42	42	15	9	6	9	NC				27	41	32	38	10	9	9	6	NC
05.57.07				38	12	52	49	9	8	6	12	NC				34	31	28	38	8	7	8	8	NC
05.57.39			13	40	20	40	51	11	9	11	8	NC	23			32	11	17	40	7	4	9	10	NC
05.58.08			18	38	23	39	50	13	10	10	5	NC	23			33	9	30	35	6	6	7	7	NC
05.58.37		32	18	23	24	34	43	14	13	11	11	NC	24		20	24	19	27	40	9	8	11	12	NC
05.59.09		33	17	10	25	34	38	5	9	8	9	NC	21		23	29	18	31	34	6	9	10	7	NC
05.59.38		34	18	19	27	37	42	8	12	11	10	NC	18		21	24	23	31	38	6	9	6	8	NC
06.00.07	32	31	24	24	26	34	39	7	11	14	8	C3	11		22	30	32	26	37	6	12	11	10	NC
06.00.39	26	34	24	36	20	38	39	6	12	9	9	C3	13		25	31	30	31	42	7	5	12	9	NC
06.01.08	26	34	22	38	20	38	38	11	11	11	13	C3	14		21	36	33	28	42	8	7	9	13	NC
06.01.37	32	28	23	35	15	31	25	10	4	10	9	C3	11		24	38	36	23	42	5	8	9	6	NC
06.02.09	37	23	22	38	22	32	38	10	5	10	11	C3	8	29	25	35	40	27	39	7	10	11	10	C1
06.02.38	39	23	21	38	26	27	43	12	10	10	11	C3	15	23	26	33	42	23	44	10	3	9	8	C1
06.03.07	42	22	24	39	31	28	40	14	11	12	9	C3	17	23	23	37	42	24	39	9	8	8	11	C1
06.03.39	38	25	24	38	29	41	33	7	11	8	8	C3	14	21	25	34	46	36	38	5	8	10	8	C1
06.04.08	34	28	24	22	36	34	36	8	12	9	9	C3	15	22	20	31	44	33	33	8	7	6	11	NC
06.04.40	26	34	21	11	36	32	35	6	14	8	11	C3	17	21	22	25	41	27	36	10	8	9	7	NC
06.05.09	23	37	22	34	33	26	45	10	13	10	9	C3	15	18	24	28	38	31	37	6	8	11	9	NC
06.05.38	23	36	24	34	34	32	28	10	10	11	9	C3	14	20	27	33	40	36	29	10	11	9	6	NC
06.06.10	17	36	28	36	34	27	23	7	13	13	9	C3	17	18	27	30	41	37	14	9	6	8	8	C1
06.06.39	13	36	30	39	24	19	26	8	11	11	9	C3	20	19	28	31	32	32	17	14	11	10	9	C1
06.07.08	13	38	24	37	36	16	34	9	9	7	13	C3	21	21	28	32	35	20	22	8	8	6	6	C2
06.07.40	11	38	23	35	48	23	41	8	10	10	11	C3	18	25	29	34	39	31	22	10	13	10	9	C2
06.08.09	10	35	24	41	51	21	36	6	7	9	8	C2	17	28	27	33	40	26	26	11	12	9	11	C2
06.08.38	7	33	26	39	51	27	29	6	9	11	9	C1	13	29	25	39	45	34	24	6	10	9	10	C2
06.09.10	9	30	23	37	61	30	35	8	6	9	12	NC	10	29	25	35	54	36	28	8	11	11	11	C2
06.09.39	14	20	29	31	63	36	22	10	5	15	9	NC	16	24	32	32	61	44	19	11	5	10	3	NC
06.10.08	19	18	31	31	54	32	33	13	8	10	8	NC	15	24	39	32	53	43	14	10	11	11	4	C3
06.10.40	24	16	29	35	46	29	36	12	7	9	11	C1	17	22	39	36	53	38	31	10	8	10	10	C3
06.11.09	24	20	28	28	46	22	40	13	13	9	10	C1	18	25	36	28	45	23	33	12	11	8	11	C3

the use of the same encoder used for controlling one lane. Consequently, the choice of control for each lane was again binary. In the two-lane experiment, however, the combinations of control or no control in either or both of the two lanes resulted in the following five control options:

NC No control in either lane
C1 Control in far lane; no control in near lane
C2 Control in near lane; no control in far lane
C3 Control in both lanes
C4 Red light displayed in both lanes

The control command was encoded by a Morse-code type encoder and was transmitted to the Lincoln Tunnel over a single signal-grade telephone line. At the tunnel, the signal was decoded by a matching decoding device. Control signals could be automatically or manually transmitted. In our experiments, the effect of the control signal had to be a reduction of the traffic input rate by as much as thirty percent to optimize the traffic flow inside the tunnel.

recording

A line of typewriter output was printed for every sixth observation period (30 seconds) to record the tunnel activity. Figure 4 shows a sample output recorded near the beginning of a run. The state of the tunnel at 30-second intervals was determined from the variables K_{ij}, V_i, and N_i. The value of K_{ij} is the count of vehicles between two neighboring traps i and j, where i goes

from 1 to 3 for the near lane and from 5 to 7 for the far lane and where $j = i + 1$. The K_{ij}'s were not initialized until a certain number of vehicles had passed over each trap and after the patterns of lengths of successive vehicles passing over trap i had been matched with the patterns of the same vehicles passing over trap j. During the peak traffic period, this initialization period required between three and five minutes. The values of V_i and N_i, where i goes from 1 to 4 for the near lane and from 5 to 8 for the far lane, were, respectively, the average speed of the last five vehicles and the number of vehicles that had passed over trap i during the last 30 seconds. Finally, CN and CF represented the control commands for the near and far lanes, respectively. Notice the gradual increase and decrease in control intensity that was provided by the control algorithm.

post processing analysis

Raw data and computed results were stored every processing period. In addition to the typewriter output shown in Figure 4, the stored data contained the trap number, speed (in feet per second), length (in feet), and time that a vehicle left the trap. This data, stored for each individual vehicle, was used in the post-processing analyses by an off-line version of the traffic control program. It was thus possible to duplicate the control run off-line and obtain the tables of values that were computed during an on-line run, but which could not be printed in the time allotted. These tables were then used for evaluating the experiment and for related traffic studies. It was also possible to use the actual traffic data to experiment with other values of the parameters of the control algorithm to simulate improvements in traffic control. This type of simulation could only sharpen the predictive value of the control algorithm in anticipating congestion because the traffic input into the tunnel was, of course, based on the control algorithm used during the actual run and was not affected by the simulation.

Concluding remarks

This experiment was a rather unique and pioneering example of the management of a traffic facility to maintain close to optimal traffic conditions. In that sense, it may be viewed as a precursor of traffic systems of the future. Such systems are likely to involve a fair amount of intervention in the allocation of traffic facilities to traffic demands, a feature virtually absent from traffic systems today. Our traffic-control experiment demonstrated the following four distinctive features:

- Accurate, continuous traffic density determination in sections of a traffic link by an ad hoc pattern recognition technique
- Development of density-oriented control algorithms for continuously optimizing traffic flow

- Remote location of the computer
- Sharing of a computer between batch scientific computations and real-time process control

The use of the 7040 for a process control application was interesting as well as successful. This system, however, required almost constant supervision because of occasional stoppages. Reinitialization of the system was required after a computer stoppage, which generally produced some loss in tunnel performance.

The 1800 was successful overall, and stoppages were virtually absent. The on-line program modification feature permitted some experimentation with the control parameters, which were occasionally coordinated by telephone with visual observations by personnel at the tunnel.

Traffic flow improvement, through on-line computer control, amounted to about a ten percent increase in throughput compared to manually controlled operation. As a by-product, the experiment provided a laboratory for testing methods of traffic monitoring and control that may have applications to other traffic systems. An example is the measurement of the traffic density (or traffic count in a section of a roadway), which had never been achieved in a traffic system with the accuracy obtained at the Lincoln Tunnel.

The subroutine for determining car counts could be used in other traffic systems, such as freeways, where accurate knowledge is desirable for a properly managed traffic facility. In such experiments, runs should be made to determine useful trap distances because they appear to depend on the nature of the traffic link. For example, freeway trap distances would undoubtedly have to be much shorter than one-half mile to acquire accurate vehicle counts because of lane changing.

An added dividend of the accurate measurements of traffic densities in the Lincoln Tunnel is that these measurements provide a benchmark for testing other methods of estimating traffic densities. For example, Knapp and Gazis[3] have proposed a theoretical method for estimating traffic densities using smoothing and filtering techniques. This method has smaller requirements of computer power and is not greatly affected by lane changes. Although the accuracy of this method could not be ascertained directly, that it can yield accurate estimates was verified by comparison with data collected during the experiment discussed in this paper.

ACKNOWLEDGMENTS

We wish to express our appreciation to our colleagues R. Treadwell, who built the interface equipment, and to J. Harry and

R. Ryniker for their assistance in incorporating the traffic control program into the 7040 and the 1800 systems, respectively. We are also grateful for the collaboration of many individuals of the Port of New York Authority, and especially to T. Farley, who wrote part of the 1800 program.

REFERENCES

1. D. C. Gazis, B. T. Bennett, R. S. Foote, and L. C. Edie, "Control of the Lincoln Tunnel by an on-line digital computer," *Beitraege zur Theorie des Verkehrsflusses* (*Proceedings of the Fourth International Symposium on Traffic Theory*, Karlsruhe, Germany, June 1968), Edited by W. Leutzbach and P. Baron, Bundesminister fuer Verkehr, Bonn, Germany, 48–56 (1970).
2. D. C. Gazis and R. S. Foote, "Surveillance and control of tunnel traffic by an on-line digital computer." *Transportation Science*, **3**, No. 3, 255–275 (August, 1969).
3. D. C. Gazis and B. T. Bennet, "Lincoln Tunnel traffic control experiment," Paper No. 700192 presented at the Automotive Engineering Congress of the Society of Automotive Engineers (January, 1970). Copies may be obtained from the author.
4. *IBM 1800 Vehicular Traffic Control System*, 1800-VG-06X Version 2. Program Description Manual No. H20-0335, International Business Machines Corporation, Data Processing Division, White Plains, New York.
5. D. C. Gazis and C. Knapp, "On-line estimation of traffic densities from time-series of flow and speed data," submitted to *Transportation Science*.