Intelligent Transportation Network System

ITNS

Control

J. E. Anderson

September 2018

Contents

Control of Personal Rapid Transit Systems

J. Edward Anderson

Abstract

The problem of precise longitudinal control of vehicles to follow predetermined time-varying speeds and positions has been solved. To control vehicles to the required close headway of at least 0.5 sec, the control philosophy is different from but no less rigorous than that of railroad practice. A PRT system can be designed with as good a safety record as any existing transit system and, because of the ease of adequate passenger protection, quite likely much better. The basis for the control of a fleet of PRT vehicles of arbitrary size is a complete set of maneuver equations. The author's conclusion is that the preferred control strategy is one that could be called an "asynchronous point follower." Such a strategy requires no clock synchronization, is flexible in the face of all unusual conditions, permits the maximum possible throughput, requires a minimum of maneuvering and uses a minimum of software. Since each vehicle is controlled independently, there is no string instability. Since the wayside zone controllers have in their memory the same maneuver equations as the on-board computers, accurate safety monitoring is practical. To obtain sufficiently high reliability, careful failure modes and effects analysis must be a key part of the design process, and the control computers must be checked redundant.

Introduction

The problem of closed-loop automatic longitudinal control of a single vehicle constrained to follow a guideway at a specified time-varying speed and position within adequate accuracy has been solved by several investigators [1, 2], and analytical equations for the required speed and position gains have been derived. The architecture of checked redundant microprocessor control for automated transit vehicles has been developed and has been shown to be able to achieve a safety record as good or better than a modern rapid rail system [3]. The major challenge in PRT control has been to control a large fleet of vehicles operating at fractional-second headway and merging and diverging in and out of stations and between separate branches in a network of guideways with an acceptable level of safety, comfort, and dependability, while meeting other essential criteria. A great deal of work has been done on this problem over the past few decades. Much of the published work can be found in conference proceedings [4, 5, 6], in papers referenced in those proceedings, and in results of the Urban Mass Transportation Administration's Advanced Group Rapid Transit Program [7, 8]. While the AGRT system was designed for 3-sec headway, much of the work is directly applicable to PRT. Together with the work of The Aerospace Corporation PRT Program [9] and the DEMAG+MBB Cabintaxi PRT Program [10], one can obtain an excellent perspective on the field.

In a short paper, it is not possible to describe any appreciable portion of this work, but it is more useful to give a synthesis of conclusions reached concerning the means of controlling a PRT system, which have been built on the shoulders of prior investigators. I first discuss the criteria any PRT control system must meet. Then, it is necessary to discuss the problem of safe achievement of adequately low time headway between vehicles and how the safety philosophy must differ from standard railroad practice. Next is a discussion of strategies of control of many vehicles in a network. With this background, the next topics are the information that must be available on board the vehicles and at various wayside points, the sensing and communication requirements, and the mathematics involved. I do not discuss lateral control because, in most PRT systems, wheels running against lateral surfaces achieve it passively.

Control Criteria

Line and Station Throughput

Analysis of PRT networks in many applications has shown that fractional-second headways are both needed and attainable. The 1974 UMTA Administrator Frank Herringer, in testimony before a committee of the Congress of the United States, said: "A DOT program leading to the development of a short, one-half to one-second headway, high-capacity PRT system will be initiated in fiscal year 1974 [11]." This statement was a result of consensus among workers in the PRT field in consultation with the Research and Development staff of UMTA on the need and practicality of headways as low as 0.5 sec. Off-line stations must be designed to meet expected input and output flows, and the system must be designed to prevent excessive congestion at merge points and destination stations.

Safety

A PRT system must provide a level of safety in terms of injuries per 100 million miles at least as good as a modern rapid rail system [3], and preferably better because the improvements provided by PRT in all areas must be good enough to justify the development cost. To achieve this level of safety, the on-board and wayside computers must be checked redundant.

Dependability

The term "dependability" is less often used than "availability," which is measurable in conventional transit systems as the percentage of trains that arrive at stations when expected. The quantity dependability, which is the ratio of person-hours not delayed to the number of person-hours of operation, is a more meaningful criteria and, in PRT, can be easily measured and updated trip by trip by a central computer [12]. In a recent PRT program, it was specified that the undependability (1 - dependability) should be no more than 3 person-hours of delay per 1000 person-hours of operation. From our analysis, if the safety criterion is met, the undependability will be at least an order of magnitude less.

Ride Comfort

Longitudinal maneuvers must be performed in such a way that International Standards Organization ride comfort standards on acceleration as a function of frequency are met. As to maneuvers, the National Maglev Initiative Office set the most recent federal standards on ride comfort that would be applicable to vehicles in which all passengers are seated. They restrict acceleration to 0.2 g and jerk to 0.25 g/s in normal operation. The maximum emergency-braking deceleration depends on whether passenger constraints are provided. If not, the criterion must be that the passenger does not slide off the seat in an emergency stop.

Changing Conditions

The control system must be able to reduce cruising speed in high winds and must be able to cope with any unusual situation, such as a stopped vehicle, that would require vehicles to slow down or stop away from a station.

Dead-Vehicle Detection

There must be a means to detect a dead vehicle on the guideway, however remote that possibility may be. In Section 5, it is stated that the vehicles must transmit their speeds and positions at frequent intervals to a wayside computer – a zone controller. If the zone controller suddenly does not receive the expected signal, it must be programmed to remove the speed signal for all vehicles in that link and transmit this information to the next upstream zone controller. Each vehicle's control system is configured to command reduction in speed to creep speed[1] if the zone controller's speed signal is not received. Magnetic detectors are placed at specified intervals along the guideway to inform the zone controller of passage of a vehicle. Thus, if a vehicle passes one of these markers and not the next, the location of the dead vehicle is approximately known. Then, as discussed at the end of Section 3.2, because the passengers are seated and can be protected and the vehicle can be protected by appropriately designed shock-absorbing bumpers [13], a creeping vehicle can be permitted to advance until it soft engages with the dead vehicle, whereupon the position of the dead vehicle becomes known and an appropriate failure strategy can be engaged.

Interchange Flexibility

The simplest interchange is a Y, with either two lines entering and one exiting or vice versa. Such an interchange gives the least visual impact at any one point, but it requires that vehicles first merge, then diverge, which creates a bottleneck after a merge. Desiring to obtain maximum possible throughput, The Aerospace Corporation [9] used two-in, two-out, multilevel interchanges, which permit vehicles to diverge first and then merge. With such interchanges, the input and output capacity of the lines is the same, hence the worst that can happen is that a vehicle may have to be diverted from the direction it would normally go. Thus, the control system does not have to be concerned with sending too much traffic along a particular line. If Y-interchanges are used, control is more complex and is discussed below. Since Y-interchanges are often necessary, the control system must permit them.

[1] A finite creep speed permits the vehicle ahead of the failed vehicle to move safely to the next zone, reduces anxiety, and with seated passengers is safe.

Vandalism and Sabotage

A system in which the control functions are distributed, and the wayside computers are protected, for example in safe rooms under the stations, will be less susceptible to damage than a system in which a central computer plays an essential role. To minimize the consequences of failures of any kind, distributed control is also preferred. The required central-computer functions should be such that the worst that can happen if it fails is that the system will operate less efficiently.

Modularity

The control units should be easily exchangeable so that down time is minimized.

Expandability

The control system should be designed for easy expansion of the system.

Principles of Safe, High-Capacity PRT

The Headway Equation

The minimum safe spacing between vehicles is the longest emergency stopping distance minus the shortest failure stopping distance. It is given by the equation

$$H_{\min} = Vt_c + \frac{V^2}{2}\left(\frac{1}{A_e} - \frac{1}{A_f}\right) \tag{1}$$

in which V is the line speed, t_c is the time constant for brake actuation, A_e is the minimum emergency braking deceleration, and A_f is the maximum failure deceleration. Strictly speaking there should be a term added involving the rates of change of deceleration (jerk), but the emergency jerk can be made high enough so that jerk does not add to H_{min}. If L is the length of the vehicle, the minimum time headway, using equation (1), is

$$T_{\min} = \frac{L + H_{\min}}{V} = \frac{L}{V} + t_c + \frac{V}{2}\left(\frac{1}{A_e} - \frac{1}{A_f}\right) \tag{2}$$

Equation (2) shows first that PRT vehicles should be as short as possible. With careful design, a length of 2.6 m is practical. A typical operating speed is 13 m/s, in which case the first term in T_{min} is 0.2 sec. Boeing work [14] showed that vehicles can transmit their speeds and positions as frequently as once every 40 msec. To command emergency braking requires two such transmissions. The braking time constant, once a signal is received must be very short. With the right technology, 100 msec is practical. Therefore, with some extra allowance, assume $t_c = 0.2$ sec. If the minimum line headway is to be 0.5 sec, the third term in equation (2) can thus be no more than 0.1 sec – practically zero. This means that *in a fractional-second headway PRT system,* *the design must be such that the minimum emergency deceleration must be as high as the maximum reasonably possible failure deceleration.*

The most recent indication of the practicality of close-headway control is an announcement by the National Automated Highway System Consortium [15] that in about a year "10 specially outfitted Buick LeSabres will take part in the first test of an automated highway." A companion article on the same page says that these 200-inch long autos will operate at a spacing of only 6 feet at "50-plus miles an hour." This works out to a time headway of 0.309 sec. At 30 mph the headway would be 0.515 sec.

Departures from Railroad Practice

In railroad practice, trains may be so long that the first term in equation (2) may be several times the term $V/2A_e$. Also, at grade level, it is easiest for some foreign object or another train to quite suddenly appear ahead. In the worst case the train ahead theoretically stops instantly, in which case the fourth term in equation (2) is zero. Relative to the size of the term L/V, this is not a severe assumption and is conservative. In railroad practice it is standard to design for the so-called "brick-wall" stop in which A_f is infinite.

A railroad block control system depends in emergency situations on a vital relay that virtually never fails. Its failure is likely to cause a collision, but such a failure is so rare that it is assumed never to occur. What is implied is that the probability that the vital relay fails when it is needed is so low that it is acceptable. There is no other choice. *In any moving system the simultaneous occurrence of two very improbable major failures may set up the conditions for a collision.*

In simple terms, in railroad practice the philosophy is that if one train is to stop instantaneously, the train behind must be able to stop in a distance short enough to avoid a collision. In PRT, the philosophy must and can be that if one vehicle stops instantaneously, someone is already killed. Therefore, one must and can design the system so that, barring a calamitous external event, it is "impossible" for one vehicle to stop instantaneously. Just as in railroad practice, "impossible" has the meaning stated in the paragraph above.

This failure philosophy requires careful analysis of every circumstance in which a sudden stop could theoretically occur. There are only two: 1) Something falls off a vehicle or a foreign object appears that wedges the vehicle in the guideway and causes it to stop very quickly, and 2) a collision with the junction point of a diverge. Making the first of these possibilities acceptably remote requires careful design and an inspection procedure that frequently assures that nothing is coming loose. Experience with road vehicles gives a feeling for the possible frequency of such an occurrence, which almost never happens to a well-maintained vehicle. By more detailed analysis than possible here it can be shown that by proper design a diverge collision will require two simultaneous highly improbable failures plus a rare "Act of God" event.

If there are many vehicles on a guideway, there are two additional possibilities for a sudden stop. One is a runaway vehicle entering a station and failing to stop before colliding with a standing vehicle, and the other is a merge collision. By use of checked-redundant vehicle control such as developed by Boeing [8], it is practical to design the control system in such a way that the mean time between over-speed failures continuing to a station collision is at least a million years. It can be shown that a merge collision would require two such failures in very close proximity in space and time, which places its MTBF in a range more remote than the estimated life of the universe.

In a PRT system designed as indicated above, there are no sudden stops; however, there may be on-board failures that require emergency braking. Equation (2) shows that to achieve safe fractional-second headway, one vehicle cannot be permitted to stop quicker than the vehicle behind. This requires closely controlled, constant deceleration braking regardless of the condition of the guideway, which rules out systems that rely on braking through wheels because in rainy or snowy weather the coefficient of friction may vary along the guideway. This is the safety-related argument for the use of linear electric motors.[2] It may be noted that it is quite likely best to decelerate at the normal rate if an on-board failure is detected. Trying to decelerate too rapidly may cause more problems than it solves.

The final factor in the difference between PRT and railroad practice is that PRT vehicles are light enough so that reasonably sized bumpers can absorb a great deal if not all of the collision energy, and all passengers are seated. By using data from auto safety practice, a PRT vehicle therefore can and should be designed so that even a collision need not cause injuries [13].

Control Strategy

General Considerations

Adequately tight control of the speed profile can be attained by using proportional plus integral (P+I) control based on tachometer feedback. A vehicle must be able to perform any one of the following maneuvers:

Speed change from given speed and acceleration to new speed

Slip given distance forward or backward from line speed

Slip given distance from acceleration maneuver

Slip given distance from slip maneuver

Advance given distance in station from rest or from deceleration maneuver

[2] Another important reason for use of linear electric motors (with an appropriate guideway design) is to eliminate the need for guideway heating.

Emergency stop

Code must be written so that the time-varying speed and position profiles of any of these maneuvers with any set of desired parameters can be calculated in the on-board computer and used as commands to the controller. If during each computational or time-multiplexing interval a wayside zone controller transmits a speed signal to all vehicles in its domain and at certain command points can transmit to a specific vehicle a maneuver command with a parameter (the desired speed, distance to slip, etc.), the vehicle has all the information it needs to perform the maneuver. Moreover, by calculating the speed profile in parallel, the zone controller has all of the information it needs to monitor the execution of the maneuver. If a vehicle moving at line speed moves away from the desired time varying position, the integral portion of the P+I controller corrects the position. If the tachometer drifts, as it will, magnetic markers along the guideway provide the basis for correcting the tachometer constant, and, by commanding a slip maneuver, the time-varying position. If the speed of the vehicle at a certain time is in error in excess of a preset amount, the zone controller assumes a fault and removes the speed signal from its domain. The vehicle controller is programmed to command creep speed if it does not receive the speed signal, so any failure causes a safe reaction.

We now have a system in which the vehicles each closely and reliably follow commanded speed profiles and are simultaneously monitored for failures by wayside zone controllers. Upon this basis it is possible to describe the maneuvers needed to operate the system. This discussion is based on extensive experience with a PRT-network simulation. We first consider the progress of an occupied vehicle from the point a passenger group enter to the point that they arrive at their destination, then we consider movement of empty vehicles.

Movement of an Occupied Vehicle

Let's join a group traveling together to the same destination by choice. We either have a magnetically coded ticket with the destination recorded on it because we take the same trip every day, or we must approach a ticket machine to punch in a destination, pay a fare, and receive a ticket. With a valid ticket we approach the forward-most available vehicle in a line of vehicles and insert the ticket into a stanchion in front of the stopped and ready vehicle. This action flashes the origin and destination station to a central computer which has in its memory the estimated arrival times of all vehicles moving through the system. If our vehicle is expected to arrive at its destination station at a time when the station is full and cannot receive another vehicle, we are informed that we must wait a specified time before we can try again. Generally, this will be a very small time and the central computer will prioritize the unfulfilled demands for service.

When the ticket can be accepted, the station computer so informs us, causes our vehicle's door to open, and transfers the memory of the destination to the on-board computer. We enter our vehicle, sit down and when ready one of us presses a "GO" button. Thereupon the door is automatically locked. If our vehicle is not in the forward-most loading berth, it must wait until the vehicle or vehicles ahead move out. If it cannot yet be commanded to line speed because an opening is not yet available, it is commanded to advance as far forward as possible.

The station zone controller meanwhile is examining the flow passing the station for an opening. By zone-controller supervision the vehicles on the main line are maintained at separations at or greater than the minimum separation permitted by equation (1). Note that there need at this point be no synchronization. If there is no traffic on the main line a vehicle can be commanded to accelerate to line speed at any time it is ready. As traffic on the main line builds up, say with the approach of the morning rush hour, vehicles pass stations at any spacing down to the minimum allowed.

To create an opening for our vehicle, the zone controller may command a mainline vehicle too close ahead to slip ahead if possible and a mainline vehicle behind to slip behind at the moment it commands our vehicle to line speed. If slipping of the mainline vehicle behind would cause the headway between it and the vehicle behind it to fall below the minimum, the zone controller would within a few milliseconds cause that vehicle to slip too, and so on upstream. If there would be too much slipping of upstream vehicles or if the slipping of downstream vehicles has propagated into the station area, our vehicle would wait until there is an acceptable opportunity to accelerate out of the station.

When an opening appears, our vehicle is commanded to accelerate out of the station, either from rest or from a station-advance maneuver. While our vehicle is accelerating, a vehicle ahead may be caused to slip because of a conflict at a downstream merge point. If that happens and if our vehicle would reach line speed too close behind the vehicle ahead after it is through slipping, our vehicle is commanded to slip the necessary amount while accelerating and, if necessary, the main-line vehicles behind it will be commanded to slip by the amount needed to maintain minimum headway.

Next, suppose our vehicle approaches a line-to-line merge point. As it passes a command point at a predetermined location upstream of the merge junction, the cognizant wayside zone controller, having in its memory the positions, speeds and slip maneuver data for each vehicle within this merge zone, gives a maneuver command needed to resolve any conflict. If the vehicle ahead on the other branch of the merge is too close, the zone controller commands it to slip ahead if possible[3], or if not, it commands our vehicle to slip back. If our vehicle is commanded to slip back it may slip into the headway domain of the vehicle behind on the same leg of the merge, in which case that vehicle and possibly vehicles behind it are commanded simultaneously to slip necessary amounts. Since our vehicle may thus already be slipping when passing the command point, the on-board maneuver algorithm is designed so that it can cause additional slip of a slipping vehicle. Such operations have been found by simulation to be completely stable.

[3] Slipping ahead is practical only if the minimum line headway is less than about one second. Otherwise the maximum travel distance to slip is excessive.

After passing the merge point, suppose our vehicle next approaches a diverge point. At a predetermined command point upstream of the diverge, the cognizant zone controller requests our destination, which is transmitted through a transmission medium to the zone controller. The diverge zone controller has in its memory a switch table giving the left or right switch command for each station in the network from that diverge point. By fiber-optic line, the central computer can transmit revised switch tables to various diverge-point zone controllers every few seconds if necessary to avoid excessive congestion in certain downstream links. The zone controller transmits the right or left switch command to our vehicle, which then acts on the command.

Next suppose our vehicle approaches a station. As soon as it has passed a merge or diverge point, it is handed off to a new zone controller that asks for and receives its destination. If this station is not our destination, the zone controller commands our vehicle to switch in the direction opposite the station off-line guideway. If this station is our destination, the zone controller does not give a switch command immediately but waits until our vehicle reaches a switch command point at the farthest downstream point at which the switch can, with a tolerance, be safely thrown. The wait is necessary because the station may have been full when our vehicle first entered the domain of the cognizant zone controller, but the last position in the waiting queue on the station off-line guideway may have cleared a few moments later.

When our vehicle reaches its destination station's switch command point, the zone controller commands it to switch in the direction of the station if there is an available berth, and if not commands it to switch away from the station. If the zone controller commands our vehicle to switch into the station, it assigns it a berth so that the next vehicle will find that this berth is reserved. Our vehicle switches if necessary and continues forward at line speed to a deceleration command point. At this point, if one or more positions down- stream of the assigned berth have cleared, a new farther-forward position is assigned, the old one is cleared, and our vehicle is commanded to decelerate along a speed profile that first reduces the speed to a predesignated station speed and then moves the vehicle forward, usually at station speed, until it must decelerate at the comfort rate to stop at the assigned position. If, at any time during the deceleration maneuver, the zone controller has advanced a vehicle out of the position or positions ahead of the assigned position, it reassigns our vehicle to the forward-most empty or to-be-empty position and revised the deceleration maneuver accordingly.

If our vehicle must stop at one of the waiting positions upstream of the station unloading and loading berths, it waits until the zone controller can command it to advance into a loading berth. If, any time during the station-advance maneuver, the berth ahead of the previously assigned berth clears, the station-advance maneuver is revised to dock our vehicle at the new forward-most free berth. When our vehicle stops, the door is either opened by a passenger or by an automatic device.

The reader may note that some PRT designers have proposed that there be separate loading and unloading platforms. This doubles the station length, reduces the throughput, and with the small passenger groups characteristic of PRT it does not significantly reduce the time required for unloading then loading.

Synchronous, Quasi-synchronous and Asynchronous Control

In the early 1970s, the discussion of PRT control virtually always started with a discussion of the relative merits of synchronous, quasi-synchronous, or asynchronous control. In a purely synchronous control system, a vehicle that is ready to leave a station waits until it has a confirmed reservation through every merge point and at the destination before being dispatched. Such a system was discarded because it is inflexible in a slow-down or stoppage on the main line; and, if the number of merges that must be negotiated exceeds three or four, the wait time becomes excessive [18]. The quasi-synchronous system was therefore proposed to permit vehicles to maneuver to resolve merge conflicts.

In his book [9] Dr. Jack Irving, while advocating quasi-synchronous control, commented that the essential point is that a wayside computer command and monitor maneuvers, just as described above. Until reaching a merge point, there is no need to synchronize the flow, and to do so in advance results in more maneuvering than necessary. As in the scheme described in the above paragraphs, whenever a vehicle arrives at the merge command point, if there is an approaching conflict, a merge-point zone controller either commands the conflicting downstream vehicle on the other leg of the merge to slip ahead if possible, or if not to slip the vehicle that has just arrived at the command point back. There is no need at merges to synchronize with specific clock times. We have also found that the described strategy requires less software than quasi-synchronicity.

Such a scheme is asynchronous except for the technicality of having to synchronize merging of certain vehicles with respect to one vehicle, but not with respect to a clock. In the 1970s, asynchronous control usually implied car following, in which each vehicle is controlled based on the position and sometimes the speed of the next downstream vehicle [1]. As pointed out above and by Dr. Irving, car following is not necessary. It complicates the control problem and is difficult for the necessary wayside monitor because the monitor does not know independently the profile of the maneuver. In the terminology used in the 1970s, the system we prefer could be called an "asynchronous point follower."

Movement of Empty Vehicles

During the night when there is little or no traffic on the system, most of the vehicles are stored at strategically located storage barns and the rest are stored at stations so that, as in elevator service, passengers don't need to wait anxiously on deserted platforms, but instead vehicles that are ready to leave immediately wait for passengers. The number of vehicles required to wait at each station must be determined by an operational study.

As passengers start arriving at stations, the waiting empty vehicles are used up and more must be ordered. Based on operational experience, a flow of empty vehicles can be started in anticipation of passengers. In any case, once the number of vehicles in a station that have not been given destinations plus the number within a specified time of arrival is less than the number of passengers waiting, the station computer signals to the central computer via fiber-optic line that it needs an empty vehicle. Other stations will have surplus empty vehicles either because there are no passengers at the station and there are more vehicles in or approaching the station than the specified minimum, or because the flow of occupied vehicles in and approaching the station exceeds the flow of passenger groups entering the station from the street. In the later case, it will sometimes be necessary to dispatch an empty vehicle while a passenger group is approaching it in order to permit occupied vehicles to enter the station and unload. In this case, the passenger group will be informed by computer voice that another vehicle will be docking in a few seconds. As soon as a station has a surplus vehicle its computer so informs the central computer and dispatches the surplus vehicle to the next station.

When an empty vehicle reaches the switch command point of a station, if the station does not need an empty vehicle its computer waves it off to the next station. If this station could use an empty vehicle, it would like to call this one in, but there may be a greater need for it at a downstream station. So, the central computer, having a knowledge of the number of empty and occupied vehicles in each link in the network and of the number and wait time of passenger groups waiting at each station, has the basis for determining whether each station should accept or wave off needed empty vehicles. Since the situation is updated every few seconds, no passenger group need wait much more than at other stations. The average wait time can be reduced by increasing the number of empty vehicles in the network, but at the expense of increased congestion and system cost.

The major decision points for distribution of empty vehicles are the diverge points. Here, as already mentioned, the central computer, with knowledge of the whole system, can, by fiber-optic link, direct left or right switch commands for the next empty vehicle. Such frequent updating of empty-vehicle commands at the last possible moment is a far easier problem to solve than the general transport problem.

Information Transfer

With the above described control strategy, the information that must be fed to the vehicle computer is the vehicle's actual speed and position; the cruising speed, which could be a function of wind or position in the guideway; and, at certain command points, the number of a maneuver with a parameter. The information required by each wayside zone controller is all vehicle positions and speeds in its domain including hand-off of the state of each vehicle as it enters its zone, and any information about anomalies. The information needed by the central computer is the stations at which there are surpluses or deficits of empty vehicles, the number of empty and occupied vehicles in each link, the destinations of and the departure times of all vehicles commanded to leave stations, the arrival times, the distance each vehicle has traveled, the distance traveled at which each vehicle is due for maintenance or cleaning, the location of and data on any faults in the system, and the weather conditions.

To perform the required data transfer there must be a continuous and noise resistant means for data transfer between vehicles and zone controllers, such as the three-wire communication line developed by Boeing [14,16], a series of magnet markers to signal passage of vehicles, and fiber-optic links between the central controller and all zone controllers. At predetermined intervals (Boeing used 40 msec), each vehicle must transmit to the cognizant zone controller its vehicle number, speed, position, destination on call from the zone controller, and any data about faults. The wayside zone controller must be able to transmit to all vehicles in its domain a continuous cruise-speed signal, and it must be able to transmit parameterized maneuver commands and switch commands to specific vehicles when needed.

For position and speed sensing, Boeing engineers [17] found that incremental wheel-angle encoders with a resolution of 0.04 foot per pulse were enough as the basis for computing both. Position measurement consisted only of counting pulses, but the calculation of speed was "considerably more complex and, to a large extent, dictated the Programmable Digital Vehicle Control System configuration" they selected. The vehicle must also be equipped with sensors to detect the magnetic markers and to transmit to and receive data from the communications line.

Mathematics

Maneuver equations

Parameterized equations are needed for all the maneuvers required to run a PRT system as described. This is not an easy task, but once the algebra is worked out, as we have done, it is available forever. The equations can easily be programmed into the memory of the on-board and wayside computers, which then permits accurate control and monitoring of each vehicle with a minimum of data transfer.

Curved-Guideway Equations

In the above discussion, reference was made to the location of certain command points. Determination of the positions of all such points requires a complete understanding of the equations of curved guideways and their use in minimization of off-line guideway lengths and distances between branch points.

Empty-Vehicle Movement

A general scheme of the points and times in the system where empty vehicles are to be redirected has been given and the use of decision algorithms has been suggested. In relatively small systems, these are quite simple, but the challenge is to optimize such algorithms as the network grows. Some good work [9] has been done on this problem, but more is needed.

Conclusions

Analysis, simulation and hardware experience has shown that the problem of precise longitudinal control of vehicles to follow predetermined time-varying speeds and positions has been solved. To control vehicles to the required close headway of at least 0.5 sec, the control philosophy is different from but no less rigorous than that of railroad practice. Available results show that a PRT system can be designed with as good a safety record as any existing transit system and, because of the ease of adequate passenger protection, quite likely better.

With maneuver equations derived in easily programmable form, one has the basis for the control of a fleet of PRT vehicles of arbitrary size. The author's conclusion is then that the preferred control strategy is one that could be called an "asynchronous point follower." Such a system requires no clock synchronization, is flexible in the face of all unusual conditions, permits the maximum possible throughput, requires a minimum of maneuvering, and a minimum of software. Since each vehicle is controlled independently, there is no string instability. Since the wayside zone controllers have in their memory the same maneuver equations as the on-board computers, accurate safety monitoring is practical. To obtain sufficiently high reliability, careful failure modes and effects analysis must be a key part of the design process, and the control computers must be checked redundant. Work of the federal Advanced Group Rapid Transit Program showed a decade ago how that can be done in a very satisfactory manner.

References

1. Garrard, W. L., Caudill, R. J., Kornhauser, A. L., MacKinnon, D., and Brown, S. J., "State-of-the-Art of Longitudinal Control of AGT Vehicles." Proceedings - Conference on "Advanced Transit and Urban Revitalization - An International Dialogue," Advanced Transit Association, Indianapolis, 1978.

2. Lang, R. P. and Freitag, D. B. "Programmable Digital Vehicle Control System," 28th Vehicular Technology Conference, IEEE Technical Paper, March 1978.

3. Milnor, R. C. and Washington, R. S. "Effects of System Architecture on Safety and Reliability of Multiple Microprocessor Control Systems." 34th Vehicular Technology Conference, IEEE Technical Paper, May 1984.

4. *Personal Rapid Transit, Personal Rapid Transit II*, and *Personal Rapid Transit III*, 1972, 1974, 1976, resp., University of Minnesota. Out of print but available in many libraries.

5. Proceedings - Conference on "Automated Guideway Transit Technology Development," UMTA-MA-06-0048-78-1, August 1978.

6. Proceedings - Conference on "Advanced Transit and Urban Revitalization - An International Dialogue," Advanced Transit Association, Indianapolis, 1978.

7. Womack, W. "Vehicle Longitudinal Control and Reliability Project Summary," UMTA-IT-06-0148-79-10, June 1979.

8. Greve, W. E., Haberman, D. E., and Lang, R. P., "Advanced Group Rapid Transit Vehicle Control Unit Design Summary," UMTA-WA-06-0011-84-3, May 1985.

9. Irving, J. H., Bernstein, H., Olson, C. L., and Buyan, J. *Fundamentals of Personal Rapid Transit,* Lexington Books, D. C. Heath and Company, Lexington, MA, 1978

10. Development/Deployment Investigation of Cabintaxi/Cabinlift System, Report No. UMTA-MA-06-0067-77-02, NTIS Report No. PB277 184, 1977.

11. Department of Transportation and Related Agencies Appropriations for 1974, Hearings before a Subcommittee of the Committee on Appropriations, House of Representatives, 93rd Congress, Part I, page 876.

12. Anderson, J. E. "Dependability as a Measure of On-Time Performance of Personal Rapid Transit Systems," *Journal of Advanced Transportation.* 26:201-212, Winter 1992.

13. Anderson, J. E. "Safe Design of Personal Rapid Transit Systems," *Journal of Advanced Transportation* 28:1-15, Spring 1994.

14. Nishinaga, E. I. and Colson, C. W. "A Vehicle Collision Avoidance System Using Time Multiplexed Hexadecimal FSK (Frequency Shift Keyed)," 33th Vehicular Technology Conference, IEEE Technical Paper, May 1983.

15. Holusha, J., "A Smart Road Starts to Take Shape in California," *New York Times*, September 1, 1996, p. 25A.

16. Nishinaga, E. "Digital FSK Receiver Capable of Operating in High Impulse Noise Environments," 31th Vehicular Technology Conference, IEEE Technical Paper, April 1981.

17. Lang, R. P. and Warren, D. J. "Microprocessor Based Speed and Position Measurement System," 33th Vehicular Technology Conference, IEEE Technical Paper, May 1983.

18. Anderson, J. E. "Synchronous or Clear-Path Control in Personal Rapid Transit Systems," *Journal of Advanced Transportation.* 30-3 (1996).

Asynchronous Point-Follower Control

Table of Contents

References:

1. *Transit Systems Theory,* Lexington Books, D. C. Heath and Company, Lexington, Mass. 1974.
2. Jack Irving et all, *Fundamentals of Personal Rapid Transit*, Lexington Books, D. C. Heath & Company, 1978, www.advancedtransit.net.
3. William E. Greve, Donald E. Haberman, and Robert P. Lang, "Advanced Group Rapid Transit Vehicle Control Unit Design Summary, Boeing Aerospace Company, UMTA-WA-06-0011-84-3, May 1985, 249 pages.
4. "System Control," PRT Design Study, Chicago RTA, 1992, 72 pages.
5. "Synchronous or Clear-Path Control in Personal Rapid Transit," *JAT*, 30:3(1996):1-3.
6. "Longitudinal Control of a Vehicle," *JAT*, 31:3(1997):237-247.
7. "Control of Personal Rapid Transit Systems," *JAT*, 32:1(1998):57-74.
8. "A Review of the State of the Art of Personal Rapid Transit." *JAT*, 34:1(2000):3-29.
9. "Some History of PRT Simulation Programs,"13 pages, (2007).
10. "Overcoming Headway Limitations in PRT Systems," PodCar Conference, Malmö, Sweden, 9-10 December 2009.
11. "Transitions," 2014, 45 pages.

I. Introduction

The above-listed references provide the basic background used to develop the work described herein. The serious reader needs to be familiar with this work before delving into the details developed in this document. The system under discussion is referred to as an "Intelligent Transportation Network System" (ITNS) to avoid direct use of the generic name "Personal Rapid Transit" or PRT because this type of "transit" has been identified with railroads, which have been for over a century subject to the 1911 Railroad Safety Act, which requires a minimum headway between trains such that if one train stops instantly, the one behind can stop without colliding. Based on experience discussed in the above references, with today's technology used as specified we can safely operate at substantially shorter headways, and we have been advised that one step is to stop calling the system a form of transit. The true proof, however, must come with extensive operation in daily practice. But the fact is that in this discussion we can't avoid using the term PRT because it is so ingrained in advanced transit culture.

Reference 7 provides the first description of an *asynchronous, point-follower* system published and explains how I concluded that it is the best way to control the vehicles in ITNS. Here is a quote from the abstract:

> "The problem of precise longitudinal control of vehicles so that they follow predetermined time-varying speeds and positions has been solved. To control vehicles to the required close headway of at least 0.5 sec, the control philosophy is different from but no less rigorous than that of railroad practice. The preferred control strategy is one that could be called an "asynchronous point follower." Such a strategy requires no clock synchronization, is flexible in all unusual conditions, permits the maximum possible throughput, requires a minimum of maneuvering and uses a minimum of software. Since wayside zone controllers have in their memory the same maneuver equations as the on-board computers, accurate safety monitoring is practical."

The key to a practical asynchronous point follower is possession of the exact equations for all of the transitions, which are developed beginning in Reference 1 and improved over the years as a result of teaching engineering mechanics and transit systems analysis and design. In a companion paper "Transitions," Reference 11, equations are derived from which to compute the transitions 1) from any speed and acceleration to rest in a given distance, 2) from any speed and acceleration to line speed while losing a given distance called "slip", and 3) from one speed and acceleration to another speed. Many of these transitions are derived in Appendix A of Reference 2. With the equations of Reference 11 developed, a high-gain controller designed according to Reference 6 causes the vehicle to follow the commanded speed-position profile very accurately. In Asynchronous Control there is no clock synchronization. All vehicle movement is a result of events. In the Section III a series of such events is discussed. In Point-Follower Control of ITNS, every transition follows the code derived in Reference 11.

II. The Control Strategy

1. A Hierarchy of three levels of control:
    a. VC – vehicle controllers
    b. ZC – wayside zone controllers
    c. CC – central control
2. Asynchronous point follower, i.e., no clock synchronization of vehicle positions. Vehicles follow calculated transitions commanded by the ZC. Each ZC checks the movement of each vehicle within its jurisdiction.
3. Adjacent ZCs pass vehicle position and speed data. Each upstream ZC informs the downstream ZC of the arrival of a vehicle, indicating its number, position, speed, and maneuver. Each downstream ZC informs its upstream ZC of the number, position, speed, and maneuver of the vehicle closest to it to warn of a slipping vehicle, i.e. one that has been commanded to a slip back to maintain prescribed minimum headway from the vehicle ahead.
4. A time interval called a "time multiplexing interval" (TMI[4]) is established. (This may not need to be a fixed interval, just an interval long enough so that the necessary information can be passed.) During each TMI a speed signal from the cognizant ZC is sent to all vehicles in its jurisdiction, and each vehicle in that zone sequentially transmits to the ZC its ID, speed, position, and any fault information. The TMI must be short enough so that in the case of an anomaly, action can be taken before a dangerous situation can develop. In 1993 Raytheon settled on a TMI of 200 msec. In the early 1980's, Boeing used 40 msec, but in a GRT system with fewer vehicles.
5. The number of vehicles that can be managed by one ZC depends on the **reliable data rate**.

[4] A term first introduced by Boeing Company in their study of AGRT for UMTA.

6. The electronic engineering team must establish the maximum number of vehicles that can be accommodated by one zone controller, and this determines the maximum zone size.
7. Each VC is configured so that if it misses a speed command two TMI in a row; it is commanded to reduce to creep speed, which at this time must be zero. (For the sake of reducing passenger anxiety by moving affected vehicles into stations, it would be better to reduce to a non-dangerous speed such as 2 mph. Substantial testing is needed to prove that a non-zero speed is safe.)
8. If a ZC misses the information or senses anomalous information from a vehicle two TMI in a row, it removes the speed signal from the faulty vehicle and those upstream of it, so as to signal them to reduce speed.
9. When the controller in a vehicle commands the vehicle's switch to be thrown, it initiates a command to stop one second later, which command is cancelled by a signal from a proximity sensor that indicates that the switch has been thrown.

III. Follow a Vehicle through a Network

We explain the events the software must perform by following a vehicle through a network:

Start with a vehicle leaving a station. Having met the conditions needed to be commanded out of the origin station (discussed in Section VIII), the vehicle reaches the Station-Exit Command Point, i.e., the point of intersection of the main guideway with station by-pass guideway. At this point, a routine called *ResetOnStationExit* resets various quantities to either the next station or if none on the link to the merge or diverge point ahead.

Assume the vehicle we are following then approaches the command point ahead of a merge. (The positions of all of the command points were calculated and stored in setup programs.) When our vehicle reaches the merge command point (MCP) the merge zone controller (MZC) goes into action. It determines if our vehicle will merge with the closest vehicle behind on the other leg of the merge at a headway closer than the established minimum headway. If so, the MZC commands the other vehicle to slow down and then return to line speed (called "slip") sufficiently far back to achieve the specified minimum headway through the merge. If in slipping back, the vehicle behind would violate the minimum headway criterion that vehicle is also caused to slip, and so on upstream until no more slipping occurs. A routine calculates slip in upstream station areas and upstream of any branch point in the network. The longer the minimum headway the farther upstream these slips will propagate.

The MZC can cause our vehicle to slip ahead instead of behind if 1) it would not reduce the headway to the vehicle ahead to less than the set minimum headway or 2) if there is sufficient space on the guideway to move the MCP back enough to permit slipping ahead.

Next assume the vehicle approaches the switch command point (SCP) of a downstream station that is not its destination. The cognizant station-zone controller (SZC) causes the vehicle to switch away from that station. The vehicle continues downstream to *ResetOnStationExit* and then until it reaches say a diverge command point (DCP). The cognizant diverge zone controller (DZC) reads the vehicle's destination, looks up the appropriate switch direction, and gives it a command to switch either to the right or left depending on which direction provides the shortest time to its destination. The CC can change these switch commands if necessary to balance the flow in the network.

Now assume our vehicle reaches the SCP of the desired destination station. The cognizant SZC determines if the destination station is or is not full of vehicles. If the station is full the vehicle must be "waved off", i.e., switched away from the station, whereupon it must proceed through the network until it returns for a second try. If there is room in the station, the SZC commands the switch to be thrown in the direction of the station and assigns our vehicle to the forward-most empty berth.

Next, our vehicle approaches the deceleration command point (DCP) where the SZC re-determines the forward-most berth and commands the vehicle to stop at that berth. Having received this command to stop in a specified distance, the VC calculates the appropriate sequence of positions and speeds at each time interval *dt* that will cause the vehicle to slow down comfortably. These positions and times become commands to the onboard control system to cause the vehicle to slow to a stop. During this maneuver, a berth or berths forward of the commanded stopping position may have opened, in which case the SZC commands our vehicle to a new farther-forward stopping position. This process continues at each computational interval until the vehicle has stopped.

If our vehicle is commanded to stop in a berth upstream of the station loading-and-unloading platform, it waits until it can be commanded by the same routine to stop at a berth farther forward. When it stops at a station berth and there are no empty positions ahead of it, if the vehicle is occupied the vehicle's door is commanded to open and the passengers begin to disembark. Once the passengers have left the vehicle, the vehicle is either available for a new group of passengers right away or it is moved forward to fill any empty berths before loading.

If the vehicle in the first berth in a station is empty, if there are no passengers wanting to board, and if there are vehicles waiting to enter the station platform, the SZC will give that first empty vehicle the destination of the nearest storage station, whereupon based on the criteria given in Section VIII it is commanded to leave the station. The ID number of this vehicle is also placed in a register of empty vehicles headed to storage, with these numbers referenced to each station. For this purpose, every segment of guideway is assigned to a station. Now, if according to an established criterion a station needs an empty vehicle, its SZC looks upstream from station to station until it either finds an available empty in one of the empty-vehicle registers or it reaches a storage station where an empty vehicle is available. The SZC then simply changes the destination of that empty vehicle to its own, whereupon the vehicle is committed and no longer available to be diverted to another station. The priority in which stations seek empty vehicles is important. During each computation interval, in a routine called *SetupCallEmpty* the priority is taken in accordance with wait time for stations in which passengers are waiting, longest wait first, then for the remaining stations the order is randomized differently each computational interval. If the wait time at a certain station has become unusually long, its call criterion can be increased so that vehicles can be called sooner.

Line speed changes are of two types: 1) due to high winds the line speed must be reduced according to a formula and then increased after the wind dies down, and 2) at specific points in the network where the vehicle must slow down for a curve and then increase speed again after passing the curve. Code for these functions is included in the routines *ChangeLineSpeedDueToWind* and *ChangeLineSpeedAtSpecificPoints*. In the latter routine, increasing speed at a specific point has no effect on the vehicles behind, but in decreasing speed, if a vehicle behind is too close, the headway between it and the vehicle ahead may go below the minimum allowable unless it is commanded to the new line speed at the same time, which is done. The criterion for slowing a vehicle down simultaneously with one that has reached the speed-change command point is derived in Reference 11.

Converting the above commands into code is a straightforward iterative process that can be appreciated in detail only through the process of writing code. To do it one must simply plunge in. Only then will one appreciate the conditions that arise and that must be corrected though code revision and addition. To catch errors in the developmental program, the primary but not only tool is a Headway Checker, which stops the simulation program if a headway violation is found. It provides enough information so that with the Randomizer off[5] the program can be run again and again until the exact cause of the error is found and corrected. Often quite a number of runs are needed to discover the exact cause of the error. While laborious, it is essential that the programmer not guess at a cause of an error. Much more often than not the real cause is not obvious. The present stage of the developmental program is such that hundreds of runs have been made with no error. While laborious and requiring a great deal of patience, the development of the program needed to simulation an ITNS network is a straight-forward application of mathematics, mechanics, and logic, and has been developed by almost every PRT developer (See Reference 9).

[5] With the Randomizer off the program generates the same sequence of pseudo-random numbers each time.

IV. Hardware & Software Elements

The Control System needed to operate ITNS consists of the following elements.

Hardware:

WHAT	HOW
Instruments on board the vehicles to sense speed and position.	Encoders mounted on wheels convert motion into a series of electrical pulses that can be converted into digital information. Since we use the main support wheels only for suspension and not for propulsion and braking, such devices provide accurate information on both distance and speed and are commercially available. Averaging left and right encoder outputs gives the correct position and speed around curves and providing encoders in both front and rear wheels provides redundancy.
Instruments at wayside to separately sense speed and position of vehicles for wayside computers.	The best-known scheme uses magnetic markers, pairs of which permit speed to be measured by measuring the time between closely spaced magnets.
A secure, environmentally friendly communications medium.	Leaky cables are commercially available at low cost and can be mounted inside the guideway to act as the communications medium. There are many suppliers.
Transcevers to transmit and receive information between vehicles and wayside via the leaky cable.	It may be necessary to design these devices from scratch to conform to the specific requirements.
Transducers, i. e., devices that convert information from one type to another – analog to digital, digital to analog.	Commercially available.
Means for propelling and braking the vehicle.	Linear induction motors (LIM) driven by variable frequency drives (VFD) provide all-weather operation and are commercially available. A pair of parking and emergency brakes will be provided, in which a high-friction pad presses down on the running surface and is operated by a ball-screw actuator. For parking this brake operates every

	time the vehicle stops. It is used for emergencies in the rare case that LIM braking is not available.
Means for permitting a vehicle to switch from one guideway to another.	The preferred means has an on-board switch arm that can engage a switch rail on either the left or right side. The guideway has no moving track parts.
Computers to be used in dual duplex sets.	Commercially available.

Software:

Software to convert information from sensors to digital information.	Commercially available.
Software to convert information from digital to voltages and frequencies.	Commercially available.
Software in vehicle computers to cause the vehicle door to open or close.	Commercially available, e. g. for operating elevator doors.
Software in vehicle computers required to operate the heating, ventilating, and air conditioning equipment.	Commercially available.
Software in vehicle computers required for calculating speed and position commands, comparing them with actual speeds and positions, multiplying them by suitable gain constants, and outputing commands to analog devices.	Commonly known to control engineers. See paper “Longitudinal Control of a Vehicle.”
Vehicle software that corrects for step changes in position due to encoder calibration without the controller seeing a step change in position.	When the encoder must be calibrated, the same correction must be fed into the command position.
Software in each wayside computer to receive speed-and-position information separately from each vehicle in its domain, to verify that that information is correct, and to take corrective information if not.	This is the monitoring and safety function of the zone controller.
Software in each wayside computer to interpret the position and speed of each	The methods are described in open literature and the needed code has been developed.

vehicle in its domain and to send the appropriate maneuver command when needed to a specific vehicle. Different software is needed in station zone controllers, merge zone controllers, and diverge controllers.	
Software in a central computer to calculate the switch table needed in diverge-zone controllers and to adjust it for traffic conditions.	This is a straightforward process using known methods.
Software to permit wayside diverge-point computers to command vehicles to switch left or right based on transmitted knowledge of the destination.	When a vehicle reaches a diverge command point, the cognizant wayside computer interrogates the vehicle for its destination, looks up in a switch table the right or left switch command for that destination, and transmits it to the vehicle.
Software to permit wayside merge-point computers to command vehicles to slip when necessary to maintain pre-set minimum headway.	When a vehicle reaches a merge command point, the wayside ZC for that merge checks the positon of vehicles on the other branch and commands slip when needed. These actions have been programmed.
Software to permit speed to be changed in different parts of a network, to reduce speed in high wind conditions, and to increase it again when the wind dies down.	This process has been programmed.
Software in wayside computers called “zone controllers” to permit one zone controller to pass vehicles to the next zone controller.	A straightforward programming task.
Software in wayside zone controllers to pass status information from vehicles to a central computer.	A straightforward programming task.
Software in a central computer to assist the optimum movement of empty vehicles.	The method we have programmed is described in Reference 9.
Software in a central computer to gather, interpret, and display performance data.	A straightforward programming task.
Software to enable voice communication between vehicles and a wayside control room.	This has not yet been programmed.

V. The System Software Elements

Some of this information has been given in a different form.

5.1. Control of a Vehicle

- Inputs from wayside Zone Controller:

  Speed every Time Multiplexing Interval.
  Maneuver command at Command Points.
  Switch commands before every diverge and merge.

- Input from on-board Encoder:
  Distance-pulse stream
- At fixed intervals along the guideway, update vehicle position and correct speed.
- As a vehicle leaves a station, calibrate the position signal.
- Calculate:
  Command Acceleration(t)
  Command Speed(t)
  Command Position(t)
  Measured distance, from encoder
  Measured speed, from distance and time encoder increments
  Command Thrust using calculated gains
  Switch Position
- Outputs:
  Voltage
  Frequency
  Switch command

5.2. Control of a Station Zone (SZC)

- The domain of the SZC is from the closest upstream branch point (BP), which may be a line-to-line BP or the nearest upstream guideway diverge point to the cognizant station output diverge point.
- Every Time Multiplexing Interval (TMI) the SZC sends the line speed to all vehicles in its domain, and receives from each vehicle in its domain its speed and position measured from the nearest downstream line-to-line BP.
- The SZC calculates the expected speed and position of each vehicle in its domain and removes the speed signal if the values from a vehicle are outside an agreed range.
- Every TMI the SZC is informed of a vehicle in the upstream zone that will arrive in its domain in the next TMI and so informs the downstream zone of the same.

- When a vehicle reaches the station's switch command point (SCP), the SZC determines if it is to switch into the station and if so, assigns it the forward-most empty berth.

- When a vehicle commanded to switch into the station reaches the station's deceleration command point (DCP) the SZC reassigns it to the forward-most empty berth, which may have changed, and commands it to stop in the distance to that berth.

- When a vehicle is either decelerating into the station or stopped and a berth further forward becomes empty, the SZC commands the vehicle to stop at the new forward-most available berth. The vehicle's door must be closed in order for it to accept the command to move forward.

- When a vehicle is assigned to the first berth, whether in it or moving to it, the SZC, with knowledge of the positions of all vehicles on the main line guideway, determines when to command the vehicle to line speed. See Section VIII.

5.3 Control of Merging (MZC)

- The domain of the MZC is from the downstream guideway junction (the merge point) upstream on each leg to the nearest BP.

- Every TMI the MZC sends the line speed to all vehicles in its domain, and receives from each vehicle in its domain its speed and position measured from the merge point ahead.

- The MZC calculates the expected speed and position of each vehicle and removes the speed signal if these values are outside an agreed range.

- Every TMI the MZC is informed of the position, speed, and acceleration of a vehicle in the upstream zone on either leg that will arrive in its domain in the next TMI and so informs the downstream zone of the same.
- When a vehicle just passes the merge command point (MCP), the MZC determines if the vehicle upstream of and closest to MCP on the other leg is close enough to violate the minimum-headway criterion. If so, this vehicle is commanded to slip back enough to maintain the set minimum headway, and simultaneously any vehicle upstream of it that would violate set minimum headway is commanded to slip. This process continues until no further slipping is needed. The program is designed to slip vehicles upstream of the upstream station and line BPs if necessary.

5.4. Control of Diverging (DZC)

- The domain of the DZC is from the downstream guideway junction (the diverge point) upstream to the nearest station or line BP,

- Every TMI the DZC sends the line speed to all vehicles in its domain and receives from each vehicle in its domain its speed and position measured from the diverge point ahead.

- The DZC calculates the expected speed and position of each vehicle and removes the speed signal if these values do not agree within a set range of the values transmitted from the vehicle.

- Every TMI the DZC is informed of the kinematic properties of a vehicle in the upstream zone that will arrive in its domain in the next TMI and so informs the downstream zones of the same.

- The DZC maintains a switch table, which is a table of switch commands to each station in the system. This table may be revised by commands from the central controller.

- When a vehicle reaches the diverge command point (DCP) the DZC requests its destination, looks up the corresponding switch command (left or right), and sends the switch command to the vehicle.

5.5. Central Control (CC)

- The CC communicates only with the zone controllers. Each ZC communicates with both the CC and the VC in its domain.
- The CC receives and processes data received from each ZC. This includes trip length, energy use, wait time, ride time, and expected ride time.
- The CC updates a calculation of system dependability[6] each TMI.
- The CC obtains data from the ZCs each TMI on the positions and speeds of all of the vehicles and determines, based on traffic, when to change certain commands in the switch tables of certain SZCs.

5.6 Empty-Vehicle Movement

- When a station has a surplus empty vehicle in or approaching its first berth, based on a criterion given elsewhere, the cognizant SZC gives it the destination of the nearest

[6] "Dependability as a Measure of On-Time Performance of Personal Rapid Transit Systems," *JAT*, 26:3(1992):101-212.

storage station and simultaneously enters the number of this empty vehicle into a register corresponding to the station.

- As the empty vehicle moves from zone to zone, its number is transferred to a register corresponding to the zone it is in.
- When a station needs an extra empty vehicle, based on a given criterion, its SZC looks upstream from ZC to ZC in order of proximity on all branches for the nearest available empty vehicle, i.e., one in an empty-vehicle register. When one is found, its destination is changed to that of the station in need.
- The order of priority of the search for empties is important. The order is that of the station with the longest wait time, the second longest, and so on until stations with no empty vehicles are reached. For them the order is random, with the random order changed every computational interval.
- The criterion for needing an empty is when the number of vehicles in a station is below $n+m$ where n is the number of station berths (where unloading and loading can take place), and m is a number (a call criterion) that can be changed by the operator or by the CC based on the observed wait times at each station, in order to decrease the difference between average wait times at all stations.
- The system average wait time can be decreased by adding more vehicles.

VI. <u>The Command Points and Actions</u>

Equations need to be incorporated into the program for the following command points.

6.1. <u>Switch Command Point (SCP)</u>

The SCP is located far enough upstream of the diverge point into the station so that if the vehicle failed to detect that the switch is in one of the two locked positions, the vehicle would be able to stop before hitting the diverge junction. This distance is at least

$$D_{SCP} = V_L t_{swx} + \frac{V_L^2}{2a_e}$$

in which V_L is the line speed, t_{swx} is the time required for the switch to throw and be detected, and a_e is the emergency deceleration.

6.2. <u>Deceleration Command Point</u>

As shown in the paper “Guideway Geometry”, we have found that to reduce the length of the offline guideway we can and should initiate deceleration into a station before the vehicle completely clears the main line. In so doing, the bypass line length can be reduced a large amount while sacrificing as little as 0.1 second on-line headway. The position of the DCP can be approximated as follows: The length of the transition curve from the main line to the parallel by-pass line is very close to

$$L_t = 4\left(\frac{V_L + V_{sta}}{2}\right)\left(\frac{H}{2J_c}\right)^{1/3}$$

in which

V_L = line speed
V_{sta} = limit speed through station
H = centerline separation between mainline and bypass line
J_c = comfort value of lateral jerk, 0.25g/s

The stopping distance of a vehicle is

$$D_{stop} = \frac{V_L}{2}\left(\frac{V_L}{A_c} + \frac{A_c}{J_c}\right)$$

Thus, we can approximately set the DCP upstream of the diverge junction into the station by the amount

$$D_{DCP} = D_{stop} - L_t.$$

At low line speeds this quantity may be negative.

The quantity D_{SCP} must be greater than $D_{DCP} + V_L \times t_{swx}$.

6.3. Diverge Command Point

Set the DCP upstream of each line-to-line diverge junction by the amount

$$D_{DCP} = V_L t_{swx} + \frac{V_L^2}{2a_e} + D_{flare} + D_{tolerance}$$

Where D_{flare} is the distance from the diverge junction to the end of the switch rail.

6.4. Merge Command Point

The merge command point must be place upstream of each line-to-line merge junction point by the amount

$$D_{MCP} = D_{slip} + D_{clearance} + D_{tolerance}$$

in which D_{slip} is the distance traveled by a vehicle slipping two headway[7] distances $V_L t_h$, in which t_h is the line headway. $D_{clearance}$ is the distance from the merge junction point upstream to the point

[7] Further simulation work may increase this value.

where a pair of vehicles on opposite branches approaching at equal distances from the junction point would just touch, and $D_{tolerance}$ is a suitable safety distance.

6.4.1 Slip Distance

If V_m is the minimum speed reached in a slip maneuver, the slip S from line speed V_L to V_m and back to line speed is

$$S = \left(V_L - V_m\right)\frac{T_m}{2}$$

$$T_m = 2\left(\frac{V_L - V_m}{A_r} + \frac{A_r}{J_c}\right) = \text{Maneuver Time}$$

in which A_r is the reduced maximum acceleration. The distance travelled while slipping S is

$$D_{slip} = \left(V_L + V_m\right)\frac{T_m}{2}.$$

The lower we set V_m the less D_{slip} will be for a given S.

6.4.2 Clearance Distance

There are two values of the clearance distance, depending on whether or not a curved path intersects a straight path, or if both incoming paths are curved. In either case, a pair of vehicles will touch if the lateral distance between paths is the vehicle width w_v. In the former case this condition occurs when the lateral distance y between the curved path and the x-axis is w_v. In the latter case the condition occurs when the lateral distance y between the curved path and the x-axis is $w_v/2$. This topic is treated in the paper "Guideway Geometry".

6.5 Station-Exit Command Point

At this command point, the vehicle is removed from the array of vehicles in the station domain, and is added to the array of the next station if there is one on the same link, or to the array for the next diverge or merge while setting the next station to zero.

6.6. Procedure for Exercising Command Points

When the distance recorded aboard a vehicle and transmitted to wayside goes to zero or to a small positive value the vehicle is just passing a line-to-line branch point where wayside control of the vehicle is handed over from the upstream ZC to the downstream ZC, and the distance recorded by the vehicle controller is set to the next line-to-line BP. If there is a station ahead, this new ZC is a SZC and it has recorded in its memory as two of its properties the distances from its SCP and DCP to the next downstream branch point. When the vehicle enters the domain of a new ZC it is entered into a register of vehicles passing through that ZC.

When a vehicle reaches a SCP the SZC evokes a subroutine that determines if it is to switch into the station based on the destination of the vehicle and the occupancy of the farthest upstream waiting berth. If it is to switch into the station, the SZC commands the vehicle's switch to be thrown, assigns it to the forward-most empty berth, and records that the SCP function for that vehicle has been evoked. (The berth assignment is recorded both in the SZC and in the vehicle computer.) Completion of the SCP function can be indicated by dividing the SZC's register of vehicles into two sub-registers: one for those that are upstream of the SCP or downstream of it and committed to bypass the station, and a second for those that are downstream of the SCP and are committed to enter the station.

When a vehicle reaches a deceleration command point (DCP) and the vehicle is to enter the station, the SZC evokes a subroutine that updates the forward-most berth assignment and commands the vehicle to stop at that forward-most berth.

When a vehicle reaches the downstream merge point of the mainline and bypass line out of a station, it is passed off to either the SZC for the next station on the same link, or to either a DZC or a MZC for the same link. In either case the vehicle is removed from the register of the upstream ZC and simultaneously entered into the register of the downstream ZC.

If the downstream line-to-line branch point is a diverge, when the vehicle reaches its DCP it is interrogated for its destination, the DZC finds the appropriate switch command from its stored switch table and sends it to the vehicle controller, whereupon the VC commands its switch to be thrown if it is in the wrong direction.

If the downstream line-to-line branch point is a merge, when the vehicle reaches its MCP the cognizant MZC determines if it will be in conflict with a vehicle on the other branch of the merge and if so causes vehicles to slip back as described in the paper "Transitions."

VII. Test for a Headway Violation upon Decelerating into a Station

7.1 Kinematics of two successive vehicles moving into a station.

Consider a vehicle #1 decelerating into a station to station speed V_{sta} and then to rest followed by a vehicle #2 a time T_h behind undergoing exactly the same maneuver. Let the position of vehicle #1 at time zero be $x(0) = 0$. The times, accelerations, speeds, and positions of vehicle #1 at the points 1, 2, 3, 4, 5, 6, and 7 in Figure 7.1, following the methodology of the paper "Transitions," are as follows:

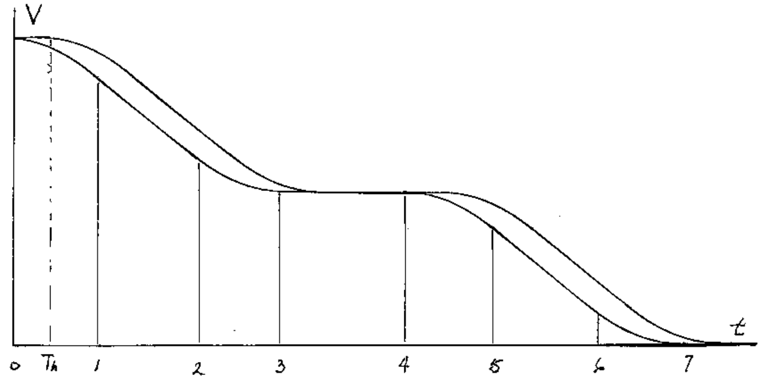


Figure 7.1. The velocity profiles of a pair of vehicles entering a station.

$$\begin{aligned}
&\text{if } V_L - V_{sta} \geq A_c^2 / J_c \text{ then } A_1 = A_c \text{ else } A_1 = \sqrt{J_c\left(V_L - V_{sta}\right)} \\
&dt_{01} = \frac{A_1}{J_c}, \quad V_1 = V_L - dt_{01}\frac{A_1}{2}, \quad dx_{01} = dt_{01}\left(V_L - dt_{01}\frac{A_1}{6}\right) \\
&dt_{23} = \frac{A_1}{J_c}, \quad V_2 = V_{sta} + dt_{23}\frac{A_1}{2}, \; dx_{23} = dt_{23}\left(V_2 - dt_{23}\frac{A_1}{3}\right) \\
&dt_{12} = \frac{V_1 - V_2}{A_c}, \qquad dx_{12} = dt_{12}\left(V_1 - dt_{12}\frac{A_c}{2}\right) \\
&dt_{34} = dx_{34} / V_{sta} \\
&dt_{45} = \frac{A_c}{J_c}, \quad V_5 = V_{sta} - dt_{45}\frac{A_c}{2}, \quad dx_{45} = dt_{45}\left(V_{sta} - dt_{45}\frac{A_c}{6}\right) \\
&dt_{67} = \frac{A_c}{J_c}, \quad V_6 = dt_{67}\frac{A_c}{2}, \; dx_{67} = dt_{67}\left(V_6 - dt_{67}\frac{A_c}{3}\right) \\
&dt_{56} = \frac{V_5 - V_6}{A_c}, \qquad dx_{56} = dt_{56}\left(V_5 - dt_{56}\frac{A_c}{2}\right)
\end{aligned} \tag{7-1}$$

$$t_1 = dt_{01}, t_2 = t_1 + dt_{12}, t_3 = t_2 + dt_{23}, t_4 = t_3 + dt_{34} \tag{7.2}$$

$$t_5 = t_4 + dt_{45}, t_6 = t_5 + dt_{56}, t_7 = t_6 + dt_{67} \tag{7.3}$$

$$x_1 = dx_{01}, x_2 = x_1 + dx_{12}, x_3 = x_2 + dx_{23}, x_4 = x_3 + dx_{34} \tag{7.4}$$

$$x_5 = x_4 + dx_{45}, x_6 = x_5 + dx_{56}, x_7 = x_6 + dx_{67} \tag{7.5}$$

Using the above canonical formulation, the acceleration, speed, and position of vehicle 1 at any value of t are as follows:

$$
\begin{aligned}
&0 \le t \le t_1: \quad \Delta t = t, \quad A = -J_c \Delta t, \quad V = V_L + \Delta t \frac{A}{2}, \quad x = \Delta t \left(V_L + \Delta t \frac{A}{6} \right) \\
&t_1 \le t \le t_2: \quad \Delta t = t - t_1, \quad A = -A_c, \quad V = V_1 + \Delta t A, \quad x = x_1 + \Delta t \left(V_1 + \Delta t \frac{A}{2} \right) \\
&t_2 \le t \le t_3: \quad \Delta t = t - t_2, \quad A = -A_1 + J_c \Delta t, \quad V = V_2 + \Delta t \frac{\left(-A_1 + A\right)}{2}, \quad x = x_2 + \Delta t \left(V_2 + \Delta t \frac{A}{3} \right) \\
&t_3 \le t \le t_4: \quad \Delta t = t - t_3, \quad A = 0, \quad V = V_{sta}, \quad x = x_3 + V_{sta} \Delta t \\
&t_4 \le t \le t_5: \quad \Delta t = t - t_4, \quad A = -J_c \Delta t, \quad V = V_{sta} + \Delta t \frac{A}{2}, \quad x = \Delta t \left(V_{sta} + \Delta t \frac{A}{6} \right) \\
&t_5 \le t \le t_6: \quad \Delta t = t - t_5, \quad A = -A_c, \quad V = V_5 + \Delta t A, \quad x = x_5 + \Delta t \left(V_5 + \Delta t \frac{A}{2} \right) \\
&t_6 \le t \le t_7: \quad \Delta t = t - t_6, \quad A = -A_c + J_c \Delta t, \quad V = V_6 + \Delta t \frac{\left(-A_c + A\right)}{2}, \quad x = x_6 + \Delta t \left(V_6 + \Delta t \frac{A}{3} \right)
\end{aligned}
\tag{7-6}
$$

For vehicle #2 up to time $t = T_h$ the speed stays constant at V_L and the distance traveled is $x = V_L t$. For $t > T_h$ we can obtain the acceleration, speed, and position as functions of time by substituting $t' = t - T_h$ for t in equations (7-6).

7.2 Results

Some results of a program to calculate the kinematics of Section 7.1 are given in Figures 7.2 and 7.3. Note from Figure 7.3 that in the case shown the small headway violation increases from zero back to zero in about one second. If based on criteria used, it is judged that the minimum headway between these two vehicles will be too small, vehicle #2 will have to be slipped back an amount that can be readily determined.

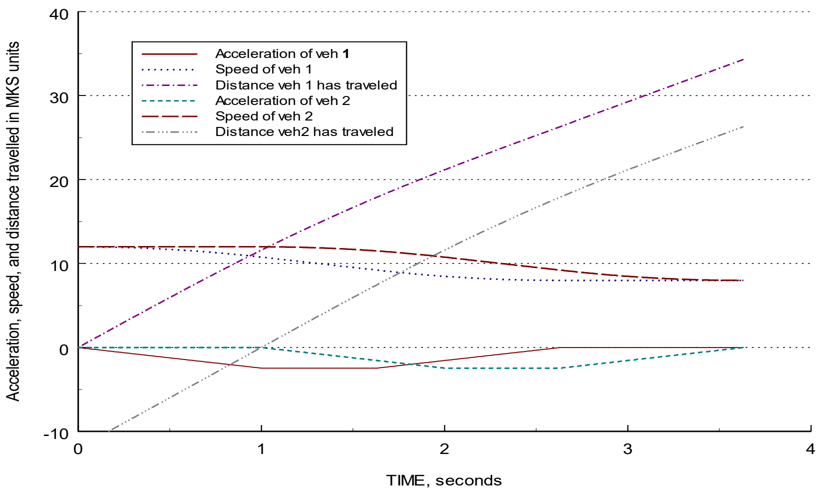


Figure 7.2. Kinematics of a pair of vehicles decelerating to station speed.

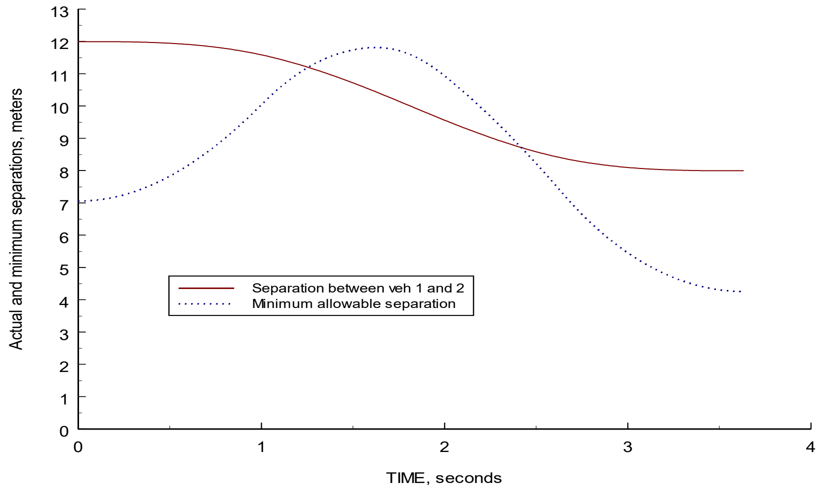


Figure 7.3. Separation and minimum allowable separation between two vehicles entering a station.

VIII. Boundaries of the Forbidden Zone

When the Subroutine *CommandLineSpeed* determines that the vehicle in the first birth (here called Veh) has been given a destination and loading of passengers is complete, it runs the Subroutine *setSpeedChangeManeuver*. This routine calculates the maneuver time and distance, T_m and D_m, respectively. At this point, it is necessary to determine if any vehicles on the line bypassing the station would conflict if Veh would be dispatched at this instant. Veh may be at any speed less than the station speed and any acceleration within comfort limits. It follows a curved path such as the heavy line in Figure 8.1, which begins at zero time and zero distance. When the maneuver is finished, it is at the time T_m and distance D_m. A vehicle bypassing the station at line speed V_L that would also arrive at the time T_m and distance D_m would at time zero be at distance $V_LT_m - D_m$ upstream of the position of Veh. If the minimum time headway is T_h then any vehicle bypassing the station within a distance of V_LT_h of the distance $V_LT_m - D_m$ at time 0 will be in the FORBIDDEN ZONE as shown by the red line in Figure 8.1. If the position of Veh behind the branch point ahead is P, then the boundaries of the FORBIDDEN ZONE are

$$BoundaryAhead = P - (V_LT_m - D_m) + V_L(T_h + c\_dt)$$
$$BoundaryBehind = P - (V_LT_m - D_m) - V_L(T_h + c\_dt)$$

in which c_dt is the computation interval.

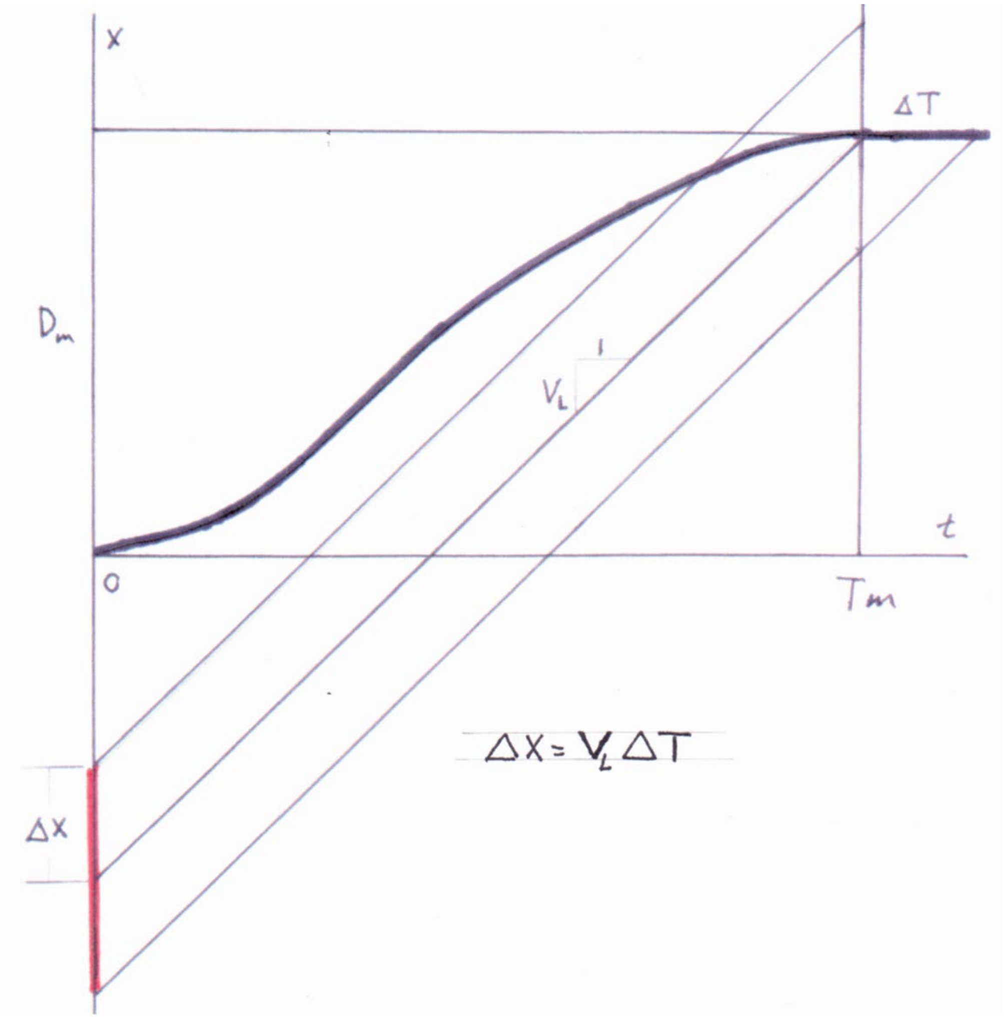


Figure 8.1. Boundaries of the Forbidden Zone.

Controlling many Vehicles in ITNS

The control system consists of computers, sensors, and a communications medium.

Computers

All computers in the system are dual redundant, which means that each "computer" is really two pairs of computers. The output of the computers in each pair is compared 20 times a second, and likewise the common output of the two pairs is compared 20 times a second. Any error detected causes the vehicle to be directed to a maintenance shop directly upon completing its trip. With this arrangement the mean time between serious events is extremely long, longer than would be believed without checking the calculations. See the internal paper "Failure Modes and Effects."

There are three types of computers: vehicle computers, wayside computers, and a central computer. Each section of guideway is managed by a wayside computer called a zone controller. There will be station zones, merge zones, diverge zones, and line zones. The zone controllers command specific maneuvers to specific vehicles and the vehicle computers respond to these commands. We have worked out the algebra needed to command every maneuver required, which consist of maneuvering from a station to line speed, slipping a certain distance ahead of merge points, and stopping in a given distance. With today's high-gain controllers we control the position of a vehicle almost as closely as we can measure it.

Each zone controller provides the line-speed signal in its domain. If anything goes wrong, it removes the speed signal, which causes the vehicles to slow to creep speed. When a vehicle reaches a maneuver-command point, the zone controller transmits the appropriate command maneuver to that vehicle, and the vehicle controller causes the vehicle to follow the required time sequence of positions and speeds. The zone controller calculates the same maneuver sequentially for each vehicle in its domain and compares it with the vehicle's position and speed. If it detects an anomaly it removes the speed signal from its portion of the guideway, which causes the vehicles to slow to creep speed. Adjacent zone controllers communicate with each other.

The central computer balances traffic in certain conditions and accumulates data on the performance of the system.

The data rates, computer speeds, and memory needed are well within the capability of today's computers.

On-Board Position and Speed Sensing

The position and speed of each vehicle is measured on board each vehicle by means of digital encoders placed in the main bearing of each of the four wheels. Averaging the left and right output gives the correct measurement in curves. Having encoders in both the fore and aft wheels provides redundancy. These encoders register at least 4096 pulses per revolution, or with the 13.25" OD tires we plan to use, about 0.010" per pulse. With this accuracy, experimental evidence has shown that we can differentiate to obtain accurate speed measurements. If the assumed the OD was in error by say 1%, the distance measurement would be in error by 1%. Thus, we will calibrate each vehicle as it leaves a station by means of fixed magnetic markers. In this way we will know the position of each vehicle to an accuracy of less than one inch.

Wayside Position and Speed Sensing

The position and speed of each vehicle is measured by suitably placed pairs of wayside markers. When a vehicle reaches the first marker, a pulse sent to the cognizant wayside computer, which detects its position at that time. When the vehicle reaches the second of the pair a known and short distance ahead, measuring the time interval between markers determines speed.

Communication

Each vehicle will be equipped with a transmitter and a receiver capable of sending information to and receiving information from a leaky cable placed on the inside of the guideway. The zone controllers similarly talk to and from the cable. Such cables are commercially available. This type of communication is completely secure and cannot be interfered with by hackers.

Background and Conclusions

Many engineers have been working on controlling PRT vehicles since the 1960s. We have followed this work closely and during our PRT Design Study for the Chicago RTA (1991-94) my team worked with experienced engineers from Raytheon and Hughes on the details. We have continued to refine the control system and the simulation of PRT systems so that today we are extremely confident that the system will work as we predict. Computer memory has doubled every 18 months since the 1960s so the computers needed today to handle the requirements are very small and extremely fast. With the use of dual redundancy failures that may occur in the system will not be due to the computers.

Maneuvers

Table of Contents

Introduction

The calculation of speed changes is fundamental to the design of the ITNS PRT control system. Once a PRT network is set up with vehicles introduced, the function of the control system is to command and then monitor speed changes. This paper provides the complete catalog of speed changes. The code to calculate them will reside in both the vehicle controllers and in the zone controllers. The wayside zone controller commands speed changes for an arbitrary initial speed and acceleration by giving the class (1, 2, 3, or 4) and a parameter: For Class 1 the parameter is the distance to stop, for Class 2 the magnitude of slip relative to the vehicle ahead, for Class 3 the final speed, and for Class 4 the minimum distance to stop. The classes are defined as follows:

1. Deceleration to rest in a given distance.
2. Slip to move back a given distance behind the vehicle ahead.
3. Speed change to a set final speed, including moving from the station to line speed.
4. Emergency stop.

All these speed changes start from given speed and acceleration. Since a very high-gain controller is use on board based on position and speed,[8] the initial speed and acceleration used on board the vehicle to command speed changes are the command values. The actual speed and position will differ by only a small amount. The complete set of speed changes derived here are all performed in minimum time consistent with given ride comfort values of acceleration and jerk.

In Section 1, the equations for going from rest to line speed are derived. In Section 2 these equations are applied, as an example, to the simplest transition from rest to line speed. In Section 3, the power-limited transition from arbitrary initial speed and acceleration to line speed is derived. This transition is included in the set derived in Section 6. Section 4 derives Class 1. Section 5 derives Class 2, Section 6 derives Class 3, and next is the derivation of Class 4.

1. Basic Transition Equations

The transitions are driven by constant jerk, at the comfort level or below. With this assumption consider the motion of a vehicle. J is the constant jerk, A is acceleration, V is speed, and x is the distance traveled. Then

$$\dddot{x} = J, \quad \ddot{x} = A_0 + Jt, \quad \dot{x} = V_0 + A_0 t + J\frac{t^2}{2}, \quad x = x_0 + V_0 t + A_0\frac{t^2}{2} + J\frac{t^3}{6} \tag{1-1}$$

Consider a transition from point 0 to point 1. Call the time interval from 0 to 1 dt_{01}. Then

$$\begin{aligned}
dt_{01} &= \frac{A_1 - A_0}{J} \\
V_1 &= V_0 + \frac{dt_{01}}{2}\left(2A_0 + Jdt\right) = V_0 + \frac{dt_{01}}{2}\left(2A_0 + A_1 - A_0\right) = V_0 + dt_{01}\left(\frac{A_0 + A_1}{2}\right) \\
dt_{01} &= \frac{V_1 - V_0}{\left(A_0 + A_1\right)/2} \\
dx_{01} &= dt_{01}\left[V_0 + \frac{dt_{01}}{6}\left(3A_0 + Jdt_{01}\right)\right] = dt_{01}\left[V_0 + dt_{01}\left(\frac{2A_0 + A_1}{6}\right)\right]
\end{aligned} \tag{1-2}$$

So, in words, the time interval is the increase in acceleration divided by positive jerk, or the decrease in acceleration divided by negative jerk, or the increase in speed divided by the average acceleration. The new speed is the old speed plus the average acceleration multiplied by the time interval, and the increase in distance is the time interval multiplied by a quantity consisting of the old speed plus the time interval times one sixth the quantity twice the old acceleration plus the new acceleration. These simple rules are all that are needed to derive any transition.[9]

[8] J. E. Anderson, "Longitudinal Control of a Vehicle," *JAT*, 31:3(1997):237-247.

[9] These rules were, to my knowledge, first derived by Raytheon control engineer Richard Radnor.

2. Basic Transition from rest to line speed

We start the transition by applying the comfort level of jerk, J_c until the acceleration reaches the comfort level A_c. We then increase speed at this constant acceleration until we approach the desired speed V_L. If we were to continue to line speed and then suddenly reduce acceleration to zero, the passengers would experience infinite jerk, so to stay within comfort jerk, we must at a certain point gradually reduce the acceleration to zero at the rate $-J_c$. Thus, like all Gaul, the transition is divided into three parts: part 0 to 1 at constant J_c until $A = A_c$, part 1 to 2 at constant A_c, and part 2 to 3 at constant $-J_c$, which ends when $A = 0$ and $V = V_L$. So we can write

$$\begin{aligned}
&dt_{01} = A_c / J_c,\ V_1 = dt_{01}A_c / 2,\ dx_{01} = dt_{01}^2 A_c / 6 \\
&dt_{23} = -A_c / -J_c = A_c / J_c,\ V_L = V_2 + dt_{23}A_c / 2,\ dx_{23} = dt_{23}\left(V_2 + dt_{23}A_c / 3\right) \\
&dt_{12} = \left(V_2 - V_1\right) / A_c,\ dx_{12} = dt_{12}\left(V_1 + dt_{12}A_c / 2\right)
\end{aligned} \tag{2-1}$$

Note that the second row must be calculated before the third row because the speed V_2 is not known until the second row is calculated. From the first and middle of the second set of equations

$$V_2 = V_L - \frac{A_c^2}{2J_c}$$

Then, using all three sets of equations (2-1) the time from rest to line speed is

$$dt_{03} = dt_{01} + dt_{12} + dt_{23} = 2\frac{A_c}{J_c} + \frac{1}{A_c}\left(V_L - \frac{A_c^2}{2J_c} - \frac{A_c^2}{2J_c}\right) = \frac{V_L}{A_c} + \frac{A_c}{J_c} \tag{2-2}$$

The distance from rest to line speed is

$$\begin{aligned}
dx_{03} = dx_{01} + dx_{12} + dx_{23} &= \frac{A_c^3}{6J_c^2} + \frac{\left(V_2 - V_1\right)}{2A_c}\left(2V_1 + V_2 - V_1\right) + \frac{A_c}{J_c}\left(V_L - \frac{A_c^2}{2J_c} + \frac{A_c^2}{3J_c}\right) \\
&= \frac{\left(V_2^2 - V_1^2\right)}{2A_c} + \frac{A_c}{J_c}V_L = \frac{1}{2A_c}\left[V_L^2 - V_L\frac{A_c^2}{J_c} + \left(\frac{A_c^2}{2J_c}\right)^2\right] - \frac{1}{2A_c}\left(\frac{A_c^2}{2J_c}\right)^2 + \frac{A_c}{J_c}V_L \\
&= \frac{V_L}{2}\left(\frac{V_L}{A_c} + \frac{A_c}{J_c}\right) = \sqrt{\frac{V_L^3}{J_c}} \ \text{ if } \ V_L < \frac{A_c^2}{J_c}.
\end{aligned} \tag{2-3}$$

So the distance from rest to line speed is simply half the line speed multiplied by the time to line speed, which if graphed shows the symmetry of the transition.

Note that if the speed V_L in the above equations were to be a very small value say V it may be that the acceleration cannot reach the comfort value A_c before negative jerk must be applied to arrive smoothly at V . In such a case $V_2 = V_1$, or from equations (2-1),

$$V_L = V = \frac{A_m^2}{J_c} \quad or \quad A_m = \sqrt{J_c V}. \tag{2-4}$$

In equations (2-4), A_m is a value of acceleration smaller that A_c. In this case, the reader can show that if A_m is substituted for A_c the final results in equations (2-2) and (2-3) still hold.

3. Transition from Arbitrary Acceleration and Speed in a Station to Line Speed with Power-Limited Acceleration

Next consider a more complex transition, but the one we need to command acceleration of a vehicle moving through a station to line speed. This transition is needed for the following two reasons:

1) A vehicle in a station behind the first berth may be ready to accelerate out of the station as soon, with an acceptable delay, as a vehicle ahead has left; and as soon, therefore, as there is an opening in the main line. When berths ahead if it open up, it is commanded to the forward-most empty berth, and while it is moving forward the station zone-control computer must, every computation interval, check to see of there is an opening for it to enter the main line from that particular state. If so, at any speed and acceleration it is commanded to the main line.

2) Power-Limited Acceleration. The acceleration power per unit of mass is VA. Thus, as speed increases at constant acceleration the power required increases in direct proportion, and then suddenly as the acceleration drops as the vehicle approaches line speed, the power required drops markedly, thus creating a sharp peak in the power required. This power peak is alleviated by causing the acceleration to decrease, as shown in Figure 1, from a point, usually at about half line speed, until the acceleration is say half the maximum value at which point maximum negative jerk is applied to bring the vehicle to the final speed.

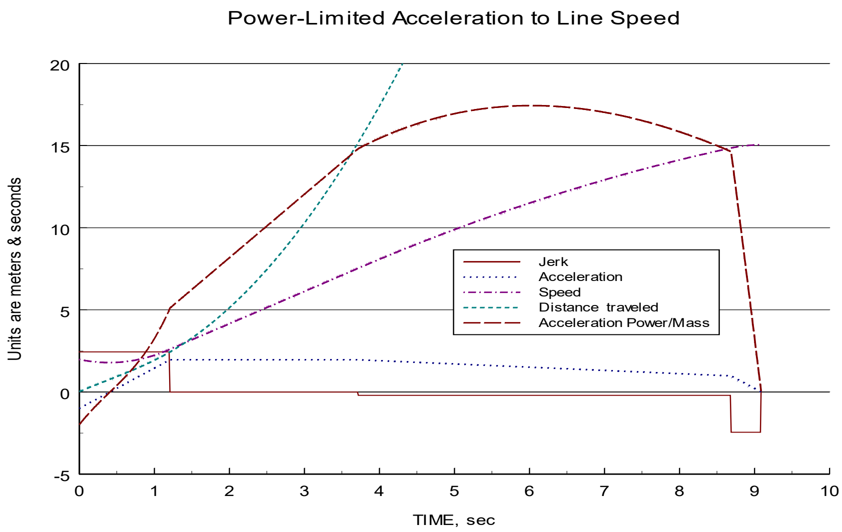


Figure 1. Power limited acceleration to line speed.

So, start the transition with A_0, V_0 different from zero. Apply positive jerk until maximum acceleration is reached at point 1. Then continue at maximum acceleration to a point 2 where negative jerk is applied until a point 3 is reached, where maximum negative jerk is applied until the acceleration is zero at a final speed at point 4. With this transition we have the following equations:

$$\begin{aligned}
&dt_{01} = \frac{A_c - A_0}{J_c},\ V_1 = V_0 + dt_{01}\left(\frac{A_c + A_0}{2}\right),\ dx_{01} = dt_{01}\left[V_0 + dt_{01}\left(\frac{2A_0 + A_c}{6}\right)\right]\\
&A_2 = A_1 = A_c,\ dt_{12} = \frac{V_2 - V_1}{A_c},\ dx_{12} = dt_{12}\left[V_1 + dt_{12}\frac{A_c}{2}\right],\ V_2 = \alpha V_L\\
&dt_{23} = \frac{A_3 - A_c}{-J_n},\ V_3 = V_2 + dt_{23}\frac{A_c + A_3}{2},\ dx_{23} = dt_{23}\left[V_2 + dt_{23}\left(\frac{2A_c + A_3}{6}\right)\right],\ A_3 = \beta A_c\\
&dt_{34} = \frac{0 - A_3}{-J_c} = \frac{A_3}{J_c},\ V_4 = V_L = V_3 + dt_{34}\frac{A_3 + 0}{2},\ dx_{34} = dt_{34}\left[V_3 + dt_{34}\left(\frac{2A_3 + 0}{6}\right)\right]\\
&T_m = dt_{01} + dt_{12} + dt_{23} + dt_{34},\quad D_m = dx_{01} + dx_{12} + dx_{23} + dx_{34}
\end{aligned} \tag{3-1}$$

Now with A_3 and V_2 known, from the fourth row of equations (3-1) we can solve for V_3. Thus

$$V_3 = V_L - dt_{34}\frac{A_3}{2}. \tag{3-2}$$

From the second and then the first equation in the third row, we have

$$dt_{23} = \frac{2\left(V_3 - V_2\right)}{A_c + A_3},\quad J_n = \frac{A_c - A_3}{dt_{23}} \tag{3-3}$$

Case when $V_1 > \alpha V_L$. When $V_1 \le \alpha V_L$ point 1 is defined as the point where $A_1 = A_c$. But the power-limited condition is determined by reducing A_1 above $V_2 = \alpha V_L$. Indeed, when $V = V_3, A = A_3 = \beta A_c$. Thus we must reduce A_1 when $V_1 > \alpha V_L$ linearly from A_c when $V_1 = \alpha V_L$ to $A_3 = \beta A_c$ when $V_1 = V_3$. Thus, when $V_1 > V_2 = \alpha V_L$ let

$$A_1 = A_c - m\left(V_1 - \alpha V_L\right) \tag{3-4}$$

where

$$m = \frac{A_c(1-\beta)}{V_3 - \alpha V_L}. \tag{3-5}$$

From the first of equations (3-1) we have a second equation relating A_1 and V_1, namely

$$V_1 = V_0 + \frac{A_1^2 - A_0^2}{2J_c}. \tag{3-6}$$

If we substitute V_1 from equation (3-6) into equation (3-4), the result can be reduced to the following quadratic equation for A_1.

$$A_1^2 + 2bA_1 - c = 0 \tag{3-7}$$

where

$$b = \frac{\overline{V}}{A_c / J_c}, \quad c = A_0^2 + 2J_c\left[\alpha V_L - V_0 + \overline{V}\right], \quad \overline{V} = \frac{V_3 - \alpha V_L}{1-\beta} = \left(\frac{1-\alpha}{1-\beta}\right)V_L - \frac{\beta^2}{1-\beta} \times \frac{A_c^2}{2J_c} \tag{3-8}$$

Equation (3-7) has one positive root:

$$A_1 = \sqrt{b^2 + c} - b \tag{3-9}$$

It can be shown that in practical cases both *b* and *c* are always positive. After A_1 is calculated from equation (3-9), it must be substituted into equation (3-6) to calculate V_1. To use the standard transition equations, we must then set $V_2 = V_1$.

With these quantities known, the rest of the calculations are straightforward and the reauired routine could be written. From it Figure 1 was calculated.

3.1 Lag remaining behind Vehicle at Line Speed

If a station precedes a line-to-line merge, it will occasionally be necessary to slip vehicles upstream of the station output merge junction. In so doing, vehicles accelerating out of the station may be required to slip, and to determine if slipping is necessary, it is necessary to know the lag in the position of the accelerating vehicle compared with its position if it were at line speed. In comparison with the Slip-Remaining term calculated in Section 5.1, let's call this distance "Lag Remaining."

The acceleration transition ends at a point 4. At point 3, maximum negative jerk is applied, and as calculated in Section 5.1 the Lag Remaining (LR) at point 3 is

$$LR_3 = \left(t_4 - t_3\right)(V_L - V_3)/3$$

At point 2, where the negative jerk J_n begins, the Lag Remaining is

$$LR_2 = LR_3 + \left(t_3 - t_2\right)\left[V_L - V_2 - \left(t_3 - t_2\right)\left(2A_2 + A_3\right)/6\right]$$

At point 1, where acceleration has just reached the maximum value $A_1 = A_c$ the Lag Remaining is

$$LR_1 = LR_2 + \left(t_2 - t_1\right)\left[V_L - \left(V_1 + V_2\right)/2\right]$$

Define a point a at which the acceleration is zero. If $A_0 > 0$ point a occurs for $t_a < t_0$. The Lag Remaining at point a is

$$\begin{aligned} LR_a &= LR_1 + \left(t_1 - t_a\right)\left[V_L - V_a - \left(V_1 - V_a\right)/3\right] \\ &= LR_1 + \left(t_1 - t_a\right)\left[V_L - V_1 + 2(V_1 - V_a)/3\right] \end{aligned}$$

in which

$$t_1 - t_a = A_1 / J_c, \quad V_1 - V_a = \left(t_1 - t_a\right) A_1 / 2.$$

Thus

$$LR_a = LR_1 + \frac{A_1}{J_c}\left[V_L - V_1 + \frac{A_1^2}{3J_c}\right]$$

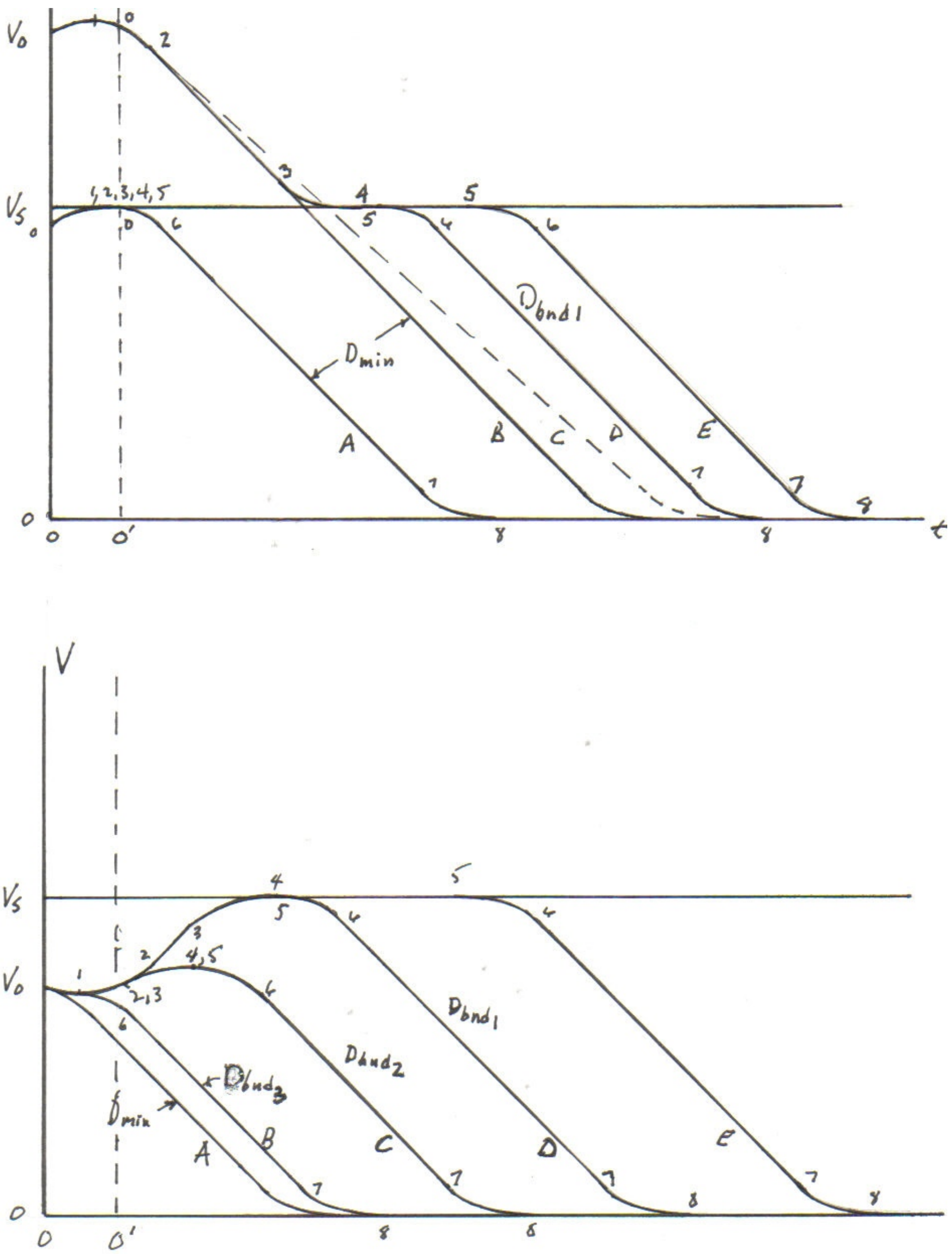
Using these values of Lag Remaining, the lag remaining at any time during the acceleration transition can be computed from the following code, in which it is assumed that $t = 0$ at the start of the transition.

```
if $t \ge t_3$ then
    $LR = (t_4 - t)(V_L - V)/3$
elseif $t \ge t_2$ then
    $LR = LR_3 + (t_3 - t)\left[V_L - V - (t_3 - t)(2A + A_3)/6\right]$
elseif $t \ge t_1$ then
    $LR = LR_2 + (t_2 - t)\left[V_L - (V_2 + V)/2\right]$
else
    $t_a = t_1 - A_1 / J_c$
    $V_a = V_1 - A_1^2 / 2J_c$
    $LR = LR_1 + \frac{A_1}{J_c}\left[V_L - V_1 + \frac{A_1^2}{3J_c}\right] - (t - t_a)\left[V_L - V_a - (V - V_a)/3\right]$
end if
```

4. Transition from Station Entry to Rest at a Specific Station Berth

The problem addressed in this section is the transition deceleration-to-rest-in-a-given-distance, i.e., the calculation of the acceleration, speed, and position as functions of time for all transitions resulting in a vehicle stopping in a station at a specific berth. The vehicle may be initially at any acceleration within the comfort range and any speed from zero up to line speed. With a time step of 20 millisecond and transitions lasting from 3.1 to 8.2 seconds, the program timer shows a time for each calculation of acceleration, speed and distance averaged over 1000 runs of between 2.88 and 3.59 microseconds, corresponding to over 5000 of such calculations during each 20 msec interval. The processor speed of the computer on which these calculations were made was 1500 mega Hz.

The transitions are described in Figures 2a and 2b. The upper figure, 2a, is for transitions for which $V_0 + A_0^2 / 2J_c \ge V_s$, where V_0 is the initial speed, A_0 is the initial acceleration, J_c is the maximum comfort value of jerk, and V_s is the maximum speed permitted in each station. The lower figure, 2b, is for transitions for $V_0 + A_0^2 / 2J_c < V_s$.



Figures 2a and 2b. The Deceleration Transitions

Consider Figure 2a.

Five curves are illustrated: A, B, C, D, and E. Curve A is for the smallest stopping distance for any curve defined by $V_0 + A_0^2 / 2J_c \geq V_s$. Curve B corresponds to the minimum stopping distance $D_{\min}$ for an arbitrary value of $V_0 + A_0^2 / 2J_c \geq V_s$. Curve E is the speed profile for the case where the vehicle cruses some distance at station speed during the time period between points 4 and 5. Curve D is for the case for which the distance cruised at station speed is zero. It is denoted as the upper boundary curve D_{bnd1}. Curve C is for a case for which the stopping distance $D_{\min} \le D_{stop} < D_{bnd1}$. For Curve C we calculate a slightly reduced deceleration so that the vehicle stops in the specified distance D_{stop}. The upper figure is drawn assuming that $A_0 > 0$ at the left-hand boundary. The case $A_0 < 0$ can be treated as if the ordinate is moved to the dashed position with $t = 0$ at 0'.

Consider Figure 2b.

Figure 2b is drawn for the case $A_0 < 0$ and also illustrates five curves. Note that if D_{stop} is large, to minimize time to stop, speed is first increased up to station speed, the vehicle may cruise some distance at station speed, and then it decelerates to rest. So, curve E is for such a case. Curve D is for the case for which the vehicle reaches V_s, but the distance cruised there is zero. It is also called the upper boundary curve D_{bnd1} and is calculated below exactly as is the curve D in the upper figure. When $D_{stop} < D_{bnd1}$ point 4 moves below station speed, as illustrated by curve C. If D_{stop} is further reduced, a point corresponding to curve C is reached at which the acceleration at point 2 just reaches A_c but the time interval $dt_{23} = 0$. This curve is designated as D_{bnd2} because above it an exact solution for V_4 is easily found, but below it a numerical solution is used.

The curve B is reached when there can be no region of positive jerk, i.e., when points 2,3,4,5 all merge with point 1. This is the boundary curve D_{bnd3}, so labeled because if $D_{stop} < D_{bnd3}$ it must be calculated differently and the acceleration at point 1 is less than zero. As D_{stop} reduces further the minimum stopping distance $D_{\min}$ is soon reached. It is represented as curve A. The case $A_0 > 0$ is treated, as above, by moving the ordinate to the right, as illustrated by the vertical dashed line.

Consider Figure 2a for the case $A_0 \ge 0$.

Using the method of Section 1, we have for the interval from 0 to 1:

$$dt_{01} = \frac{A_0}{J_c}, \quad V_1 = V_0 + dt_{01}\frac{A_0}{2}, \quad dx_{01} = dt_{01}\left(V_0 + dt_{01}\frac{A_0}{3}\right)$$
$$\text{or} \quad V_1 = V_0 + \frac{A_0^2}{2J_c}, \quad dx_{01} = \frac{A_0}{J_c}\left(V_0 + \frac{A_0^2}{3J_c}\right) \tag{4-1}$$

In the interval 1 to 4, if $V_1 - V_s \ge A_c^2 / J_c$, where A_c is the maximum comfort acceleration, then $A_2 = A_3 = -A_c$. For a smaller speed difference set $V_1 - V_s = A_2^2 / J_c$, from which

$$A_2 = -\sqrt{J_c(V_1 - V_s)} \tag{4-2}$$

Like equation (2-3), the distance traveled in the interval 1 to 4 is

$$dx_{14} = \frac{(V_1 + V_s)}{2}\left[\frac{(V_1 - V_s)}{-A_2} + \frac{-A_2}{J_c}\right] \tag{4-3}$$

If $V_1 - V_s < A_c^2 / J_c$, equation (4-3) reduces to

$$dx_{14} = (V_1 + V_s)\sqrt{\frac{V_1 - V_s}{J_c}} \tag{4-3a}$$

Because in this case we will always have $V_s > A_c^2 / J_c$ we always have

$$dx_{58} = \frac{V_s}{2}\left(\frac{V_s}{A_c} + \frac{A_c}{J_c}\right) \tag{4-4}$$

Using equations (4-1), (4-3, 3a), (4-4)

$$D_{bnd1} = dx_{01} + dx_{14} + dx_{58}. \tag{4-5}$$

To calculate the transitions we need the times, speeds and distances at each point between 0 and 8. So for this case, $V_0 + A_0^2 / 2J_c \ge 0,\ A_0 \ge 0$ we have

$$dt_{12} = \frac{A_2}{-J_c},\quad V_2 = V_1 + dt_{12}\frac{A_2}{2},\quad dx_{12} = dt_{12}\left(V_1 + dt_{12}\frac{A_2}{6}\right)$$

$$dt_{34} = \frac{-A_2}{J_c},\quad V_3 = V_s - dt_{34}\frac{A_2}{2},\quad dx_{34} = dt_{34}\left(V_3 + dt_{34}\frac{A_2}{3}\right)$$

$$dt_{23} = \frac{V_3 - V_2}{A_2},\quad dx_{23} = dt_{23}\left(V_2 + dt_{23}\frac{A_2}{2}\right)$$

Note that

$$\frac{V_2 - V_3}{-A_2} = \frac{1}{-A_2}\left(V_1 - V_s - \frac{A_2^2}{J_c}\right) \to \sqrt{\frac{V_1 - V_s}{J_c}} + \frac{A_2}{J_c} \to 0 \text{ when } V_1 \to V_s.$$

$$t_1 = dt_{01},\quad t_2 = t_1 + dt_{12},\quad t_3 = t_2 + dt_{23},\quad t_4 = t_3 + dt_{34} \tag{4-6}$$

$$x_1 = x_0 + dx_{01},\quad x_2 = x_1 + dx_{12},\quad x_3 = x_2 + dx_{23},\quad x_4 = x_3 + dx_{34}$$

If $D_{stop} > D_{bnd1}$ then $dx_{45} = D_{stop} - D_{bnd1}, dt_{45} = dx_{45} / V_s$ else $dx_{45} = dt_{45} = 0$

$$x_5 = x_4 + dx_{45}, \quad t_5 = t_4 + dt_{45} \qquad \text{(4-7)}$$

$$\begin{aligned}
&dt_{56} = \frac{A_c}{J_c}, \quad V_6 = V_s - dt_{56}\frac{A_c}{2}, \quad dx_{56} = dt_{56}\left(V_s - dt_{56}\frac{A_c}{6}\right) \\
&dt_{78} = \frac{A_c}{J_c}, \quad V_7 = dt_{78}\frac{A_c}{2}, \qquad dx_{78} = dt_{78}\left(V_7 - dt_{78}\frac{A_c}{3}\right) \\
&dt_{67} = \frac{V_6 - V_7}{A_c}, \quad dx_{67} = dt_{67}\left(V_6 - dt_{67}\frac{A_c}{2}\right) \\
&t_6 = t_5 + dt_{56}, \; t_7 = t_6 + dt_{67}, \; t_8 = t_7 + dt_{78} \\
&x_6 = x_5 + dx_{56}, \; x_7 = x_6 + dx_{67}, \; x_8 = x_7 + dx_{78}
\end{aligned} \qquad \text{(4-8)}$$

Consider Figure 2a for the case $A_0 < 0$.

In this case $t = 0$ occurs after the virtual point 1, at which speed is a maximum and acceleration is zero. Point 0 is shown by the vertical dashed line. Then proceeding using equations (1-2) now with the point 1 earlier than point 0, we get

$$\begin{gathered}
dt_{10} = \frac{A_0}{-J_c}, \quad V_0 = V_1 + dt_{10}\frac{A_0}{2}, \quad dx_{10} = dt_{10}\left(V_1 + dt_{10}\frac{A_0}{6}\right) = dt_{10}\left(V_0 - dt_{10}\frac{A_0}{3}\right) \\
V_1 = V_0 + \frac{A_0^2}{2J_c}, \quad dx_{10} = -\frac{A_0}{J_c}\left(V_0 + \frac{A_0^2}{3J_c}\right)
\end{gathered} \qquad \text{(4-9)}$$

But in this case, we must subtract the time and distance increment to get the correct values of the intermediate and final times and distances. By comparing with equations (4-1) we see that this is accomplished simply by using equations (4-1) for all values of A_0. With this interpretation, equations (4-1) through (4-8) apply to all values of A_0.

Consider the curve C of Figure 2a.

When $D_{\min} \le D_{stop} < D_{bnd1}$ let's adjust the deceleration A_2 so that the stopping distance is given by

$$D_{stop} = dx_{01} + \frac{V_1}{2}\left(\frac{V_1}{-A_2} + \frac{-A_2}{J_c}\right) \qquad \text{(4-10)}$$

Then

$$A_2^2 - \frac{2J_c}{V_1}\left(D_{stop} - dx_{01}\right)(-A_2) + J_c V_1 = 0$$

from which

$$A_2 = -\frac{J_c}{V_1}\left(D_{stop} - dx_{01}\right) + \sqrt{\left[\frac{J_c}{V_1}\left(D_{stop} - dx_{01}\right)\right]^2 - J_c V_1}$$

With this value of A_2 we have

$$\begin{aligned}
&dt_{12} = \frac{A_2}{-J_c}, \quad V_2 = V_1 + dt_{12}\frac{A_2}{2}, \quad dx_{12} = dt_{10}\left(V_1 + dt_{12}\frac{A_2}{6}\right) \\
&dt_{34} = \frac{-A_2}{J_c}, \quad V_3 = -dt_{34}\frac{A_2}{2}, \quad dx_{34} = dt_{34}\left(V_3 + dt_{34}\frac{A_2}{3}\right) \\
&dt_{23} = \frac{V_3 - V_2}{A_2}, \quad dx_{23} = dt_{23}\left(V_2 + dt_{23}\frac{A_2}{2}\right) \qquad \text{(4-11)} \\
&dt_{45} = dt_{56} = dt_{67} = dt_{78} = 0 \\
&dx_{45} = dx_{56} = dx_{67} = dx_{78} = 0
\end{aligned}$$

Consider Figure 2b for the case $A_0 \le 0$.

When $D_{stop} \ge D_{bnd3}$ $A_1 = 0$ so we have the equations

$$\begin{aligned}
&dt_{01} = \frac{-A_0}{J_c}, \quad V_1 = V_0 + dt_{01}\frac{A_0}{2}, \quad dx_{01} = dt_{01}\left(V_0 + dt_{01}\frac{A_0}{3}\right) \\
&\qquad V_1 = V_0 - \frac{A_0^2}{2J_c}, \quad dx_{01} = -\frac{A_0}{J_c}\left(V_0 - \frac{A_0^2}{3J_c}\right) \\
&dt_{12} = \frac{A_2}{J_c}, \quad V_2 = V_1 + dt_{12}\frac{A_2}{2}, \quad dx_{12} = dt_{12}\left(V_1 + dt_{12}\frac{A_2}{6}\right) \\
&dt_{34} = \frac{-A_2}{-J_c}, \quad V_3 = V_4 - dt_{34}\frac{A_2}{2}, \quad dx_{34} = dt_{34}\left(V_3 + dt_{34}\frac{A_2}{3}\right) \\
&dt_{23} = \frac{V_3 - V_2}{A_2}, \quad dx_{23} = dt_{23}\left(V_2 + dt_{23}\frac{A_2}{2}\right) \qquad \text{(4-12)}
\end{aligned}$$

If $V_2 = V_3$ then $dt_{23} = dx_{23} = 0$ and $V_4 - V_1 = \frac{A_2^2}{J_c}, \quad A_2 = \sqrt{J_c\left(V_4 - V_1\right)}$

So, if $V_4 - V_1 \ge \frac{A_c^2}{J_c}$ then $\mathrm{A}_2 = A_c$ else $A_2 = \sqrt{J_c\left(V_4 - V_1\right)}$ (4-13)

If $V_4 - V_1 \ge \frac{A_c^2}{J_c}$ then $dx_{14} = \frac{\left(V_4 + V_1\right)}{2}\left(\frac{V_4 - V_1}{A_c} + \frac{A_c}{J_c}\right)$ else $dx_{14} = \left(V_4 + V_1\right)\sqrt{\frac{V_4 - V_1}{J_c}}$

Now we can write

$$D_{bnd1} = dx_{01} + dx_{14}\Big|_{V_4 = V_s} + dx_{58}\Big|_{\text{from eq. (6.4-4)}} \qquad \text{(4-14)}$$

When $D_{stop} \geq D_{bnd1}$ $V_4 = V_s$. If $D_{stop} < D_{bnd1}$ $V_4 < V_s$. Thus, in equation (4-14) $V_4 < V_s$ must be substituted for V_s, and we must take into account that

$$\begin{aligned} dx_{58} &= \frac{V_4}{2}\left(\frac{V_4}{A_c} + \frac{A_c}{J_c}\right) \quad \text{if} \quad V_4 \geq \frac{A_c^2}{J_c} \\ &= V_4\sqrt{\frac{V_4}{J_c}} \qquad\qquad \text{if} \quad V_4 < \frac{A_c^2}{J_c} \end{aligned} \tag{4-15}$$

For $D_{bnd3} \leq D_{stop} < D_{bnd1}$ we find the curve properties by setting

$$D_{stop} = dx_{01} + dx_{14} + dx_{58} \tag{4-16}$$

If $V_4 - V_1 \geq \frac{A_c^2}{J_c}$ we can substitute from equations (4-11) and (4-15) to get

$$\begin{aligned} & D_{stop} = dx_{01} + \frac{(V_4 + V_1)}{2}\left(\frac{V_4 - V_1}{A_c} + \frac{A_c}{J_c}\right) + \frac{V_4}{2}\left(\frac{V_4}{A_c} + \frac{A_c}{J_c}\right) \\ \text{or} \quad & V_4^2 + 2\left(\frac{A_c^2}{2J_c}\right)V_1 - A_c\left[D_{stop} - dx_{01} + \frac{V_1}{2}\left(\frac{V_1}{A_c} - \frac{A_c}{J_c}\right)\right] = 0 \\ \text{or} \quad & V_4 = -\frac{A_c^2}{2J_c} + \sqrt{\left(\frac{A_c^2}{2J_c}\right)^2 + A_c\left[D_{stop} - dx_{01} + \frac{V_1}{2}\left(\frac{V_1}{A_c} - \frac{A_c}{J_c}\right)\right]} \end{aligned} \tag{4-17}$$

which is the one solution for which $V_4 > 0$. If $V_4 - V_1 < \frac{A_c^2}{J_c}$ then substitution into equation (4-16) results in a quartic in V_4, which is most easily solved by iteration. Before considering iteration we derive formulae for D_{bnd2} and D_{bnd3}.

The Boundary D_{bnd2}.

The boundary D_{bnd2} is found from the equation $D_{bnd2} = dx_{01} + dx_{14} + dx_{58}$, in which we substitute $V_4 = V_1 + A_c^2 / J_c$. Thus

$$\begin{aligned} D_{bnd2} &= dx_{01} + \left(2V_1 + \frac{A_c^2}{J_c}\right)\frac{A_c}{J_c} + \frac{1}{2}\left(V_1 + \frac{A_c^2}{J_c}\right)\left(\frac{V_1}{A_c} + 2\frac{A_c}{J_c}\right) \\ &= dx_{01} + \frac{V_1}{2}\left(\frac{V_1}{A_c} + 7\frac{A_c}{J_c}\right) + 2\frac{A_c^3}{J_c^2} \end{aligned} \tag{4-18}$$

The Boundary D_{bnd3}.

This boundary is defined by the conditions that points 2, 3, 4, and 5 are collapsed into point 1, in which case $dt_{12} = dt_{23} = dt_{34} = dt_{45} = 0$, which means that $V_4 = V_1$. Thus

$$\begin{aligned} D_{bnd3} &= dx_{01} + \frac{V_1}{2}\left(\frac{V_1}{A_c} + \frac{A_c}{J_c}\right) \quad \text{if} \quad V_1 \ge \frac{A_c^2}{J_c} \\ &= dx_{01} + V_1\sqrt{\frac{V_1}{J_c}} \qquad\qquad \text{if} \quad V_1 < \frac{A_c^2}{J_c} \end{aligned} \tag{4-19}$$

Solution for $D_{bnd3} < D_{stop} < D_{bnd2}$.

Consider the function $D(V_4)$, which between these boundaries is a quartic, and for which the solution is known at both ends, i.e., $D(V_1) = D_{bnd3}$ and $D(V_1 + A_c^2 / J_c) = D_{bnd2}$, where in this case $V_1 + A_c^2 / J_c < V_s$. So start the solution by drawing a line between the upper and lower points and calculate the first approximation as the value of $V_4 = {}^1V_4$ on that line, which intersects D_{stop}. Draw a line between the point $D({}^1V_4)$ and either the upper or lower end point (whichever is closest). Let the second approximation ${}^2V_4$ be the value on this line that intersects D_{stop}. This process continued converges very rapidly. The details are in the program that follows.

Solution for $D_{stop} < D_{bnd3}$.

When $D_{\min} \le D_{stop} < D_{bnd3}$ A_1 can no longer reach zero, i.e., $A_1 < 0$. In this case there is no point in applying first positive jerk then negative jerk in the path to zero speed. Thus consider a virtual point 'a' left of point 0, let

$$V_a = V_0 + \frac{A_0^2}{2J_c} \tag{4-20}$$

Hence A_1 is defined by the equation

$$D_{stop} = \frac{V_a}{2}\left(\frac{V_a}{-A_1} + \frac{-A_1}{J_c}\right) + dx_{a0}, \quad \text{where} \quad dx_{a0} = \frac{A_0}{J_c}\left(V_0 + \frac{A_0^2}{3J_c}\right) \tag{4-21}$$

Let $A_m = -A_1$ and $D' = D_{stop} - dx_{a0}$. Then equation (4-21) can be written in form

$$A_m^2 - 2\frac{D'J_c}{V_a}A_m + J_cV_a = 0 \tag{4-22}$$

which has the relevant solution

$$A_m = \frac{D' J_c}{V_a} - \sqrt{\left(\frac{D' J_c}{V_a}\right)^2 - J_c V_a} \qquad (4\text{-}23)$$

in which V_a is given in equation (4-20). The sign in front of the radical is determined from the condition that increasing A_m must decrease D_{stop}. Now for this case we have

$$\begin{aligned}
&dt_{a0} = \frac{A_0}{-J_c}, \quad V_a = V_0 - dt_{a0}\frac{A_0}{2}, \quad dx_{a0} = dt_{a0}\left(V_a + dt_{a0}\frac{A_0}{6}\right) = dt_{a0}\left(V_0 - dt_{a0}\frac{A_0}{3}\right) \\
&dt_{01} = \frac{A_1 - A_0}{-J_c}, \quad V_1 = V_0 + dt_{01}\frac{A_1 + A_0}{2}, \quad dx_{01} = dt_{01}\left(V_0 + dt_{01}\frac{2A_0 + A_1}{6}\right) \\
&dt_{23} = \frac{-A_1}{J_c}, \quad V_2 = -dt_{23}\frac{A_1}{2}, \quad dx_{23} = dt_{23}\left(V_2 + dt_{23}\frac{A_1}{3}\right) \\
&dt_{12} = \frac{V_1 - V_2}{-A_1}, \quad dx_{12} = dt_{12}\left(V_1 + dt_{12}\frac{A_1}{2}\right) \\
&dt_{34} = dt_{45} = dt_{56} = dt_{67} = dt_{78} = 0
\end{aligned} \qquad (4\text{-}24)$$

Consider Figure 2b for the case $A_0 > 0$.

In this case the lower boundary D_{bnd3} does not appear. As in the case of Figure 2a this case is most easily solved by assuming a virtual point 1 to the left of point 0, which lies at a time denoted by the vertical dashed line. Thus, for the interval $1 \rightarrow 0$

$$\begin{aligned}
&dt_{10} = \frac{A_0}{J_c}, \quad V_0 = V_1 + dt_{10}\frac{A_0}{2}, \quad dx_{10} = dt_{10}\left(V_1 + dt_{10}\frac{A_0}{6}\right) = dt_{10}\left(V_0 - dt_{10}\frac{A_0}{3}\right) \\
&V_1 = V_0 - \frac{A_0^2}{2J_c}, \quad dx_{10} = \frac{A_0}{J_c}\left(V_0 - \frac{A_0^2}{3J_c}\right)
\end{aligned} \qquad (4\text{-}25)$$

Comparing with the first of equations (4-12), and considering that the time and distance intervals 0-1 must be subtracted in this case when $A_0 > 0$, we see that we get the correct results in both cases if we use the equations

$$dt_{01} = -\frac{A_0}{J_c}, \quad V_1 = V_0 - \frac{A_0^2}{2J_c}, \quad dx_{10} = -\frac{A_0}{J_c}\left(V_0 - \frac{A_0^2}{3J_c}\right) \qquad (4\text{-}26)$$

to describe the cases of Figure 2b for all values of A_0.

The cases of Figure 2a apply when

$$dV_0 = V_0 + \frac{A_0^2}{2J_c} - V_s > 0 \tag{4-27}$$

and the cases of Figure 2b apply when $dV_0 < 0$. The case $dV_0 = 0$ is an important special case because it applies to a commanded change of D_{stop} while a vehicle is either cruising at V_s or in the constant $-J_c$ region of deceleration from V_s to rest. So let the quantities of the interval 0-1 be computed for all conditions except $dV_0 = 0$ from the equations

$$dt_{01} = \text{sgn}\left(dV_0\right)\frac{A_0}{J_c}, \quad V_1 = V_0 + \text{sgn}\left(dV_0\right)\frac{A_0^2}{2J_c}, \quad dx_{01} = \frac{A_0}{J_c}\left(\text{sgn}\left(dV_0\right)V_0 + \frac{A_0^2}{3J_c}\right) \tag{4-28}$$

in which, to avoid round-off errors we will assume that if $\left|dV_0\right| < 0.0001$ then $dV_0 = 0$.

<u>The Case</u> $dV_0 = 0$.

In this case we may have A_0 either positive, negative, or zero. Using the previous method the solution is

$$\begin{aligned}
&dt_{01} = \frac{\left|A_0\right|}{J_c}, \quad V_1 = V_0 + \text{sgn}\left(A_0\right)\frac{A_0^2}{2J_c}, \quad dx_{01} = dt_{01}\left(V_0 + dt_{01}\frac{A_0}{3}\right)\\
&dt_{12} = dt_{23} = dt_{34} = 0, \quad dx_{45} = D_{stop} - D_{bnd1}, \quad dt_{45} = dx_{45} / V_s\\
&V_4 = V_5 = V_s, \quad A_5 = 0, \quad A_6 = -A_c\\
&dt_{56} = -A_6 / J_c, \quad \ldots
\end{aligned}$$

in which the remaining terms are the same as given by equations (4-8).

5. Slip Transitions

5.1 <u>Introduction</u>

A slip transition is used to cause a vehicle to reduce and then increase speed in order to increase the headway between it and the vehicle ahead. The final speed that ends the transition is always the line speed V_L. The transition may be initiated at any acceleration A_0 within the comfort range and any speed V_0 between V_L and a set minimum speed for slip transitions $V_{\min}$. The slip transitions are illustrated in Figure 3 and are defined by jerk in the following table.

Interval	0→1	1→2	2→3	3→4	4→5	5→6	6→7	7→8
Jerk	$-J_c$	$-J_c$	0	J_c	0	J_c	0	$-J_c$

The purpose of this section is to derive the equations needed to calculate the time history of the motion throughout a slip transition. To do so, we must understand all aspects of the time history for any value of slip.

5.2 The Boundaries

In the upper diagram of Figure 3, we indicate four boundaries. S_{bnd1} is the value of slip for which the speed $V_4 = V_{\min}$ and $dt_{45} = 0$, in which dt_{45} is the time interval between points 4 and 5. If slip $S > S_{bnd1}$ then

$$dt_{45} = \frac{S - S_{bnd1}}{V_L - V_{\min}}. \tag{5-1}$$

Since slip transitions occur near line speed, we assume the maximum magnitude of acceleration is $A_r < A_c$.

S_{bnd2} is the value of slip for which $dt_{23} = 0$, while $A_2 = -A_r$, $V_4 = V_1 - A_r^2 / J_c$.

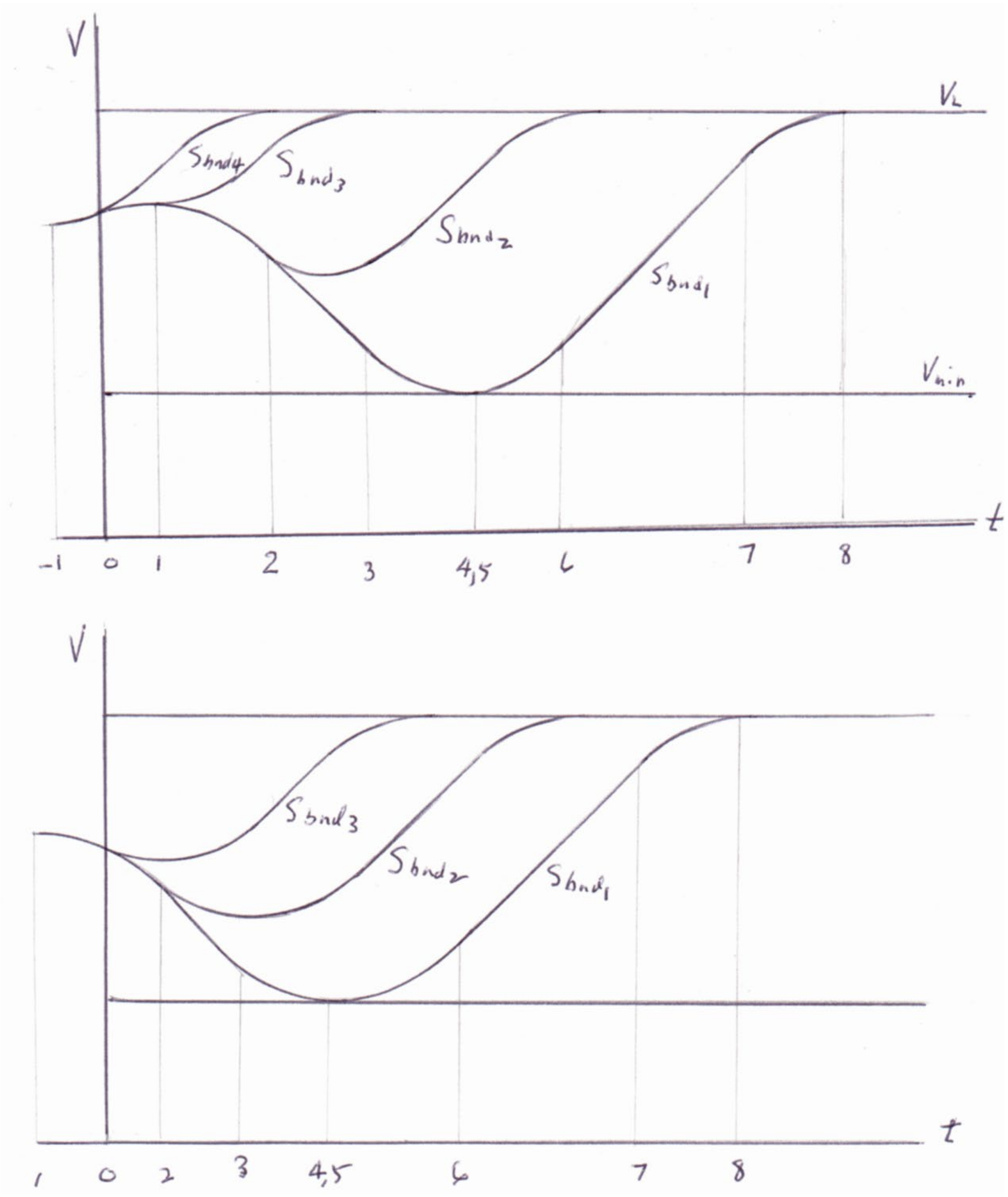


Figure 3. Slip Transitions

When $A_0 \geq 0, S_{bnd_3}$ is the value of slip for which $V_1 = V_4$. When $A_0 < 0$ point 1 is located left of $t = 0$ at the point where, extrapolating the time-distance curve at the same negative jerk as between points 0 and 2, acceleration reaches zero. When $A_0 < 0$ this boundary starts at point 0 and is the minimum possible slip, $S_{\min}$, thus when $A_0 < 0$ $S_{\min} = S_{bnd_3}$. In both cases $A_1 = 0$.

When $A_0 \geq 0$ we have

$$dt_{01} = \frac{A_0}{J_c}, \quad V_1 = V_0 + dt_{01}\frac{A_0}{2}, \quad dx_{01} = dt_{01}\left(V_0 + dt_{01}\frac{A_0}{3}\right) = \frac{A_0}{J_c}\left(V_0 + \frac{A_0^2}{3J_c}\right)$$

When $A_0 < 0$ we have

$$dt_{10} = -\frac{A_0}{J_c}, \quad V_1 = V_0 - dt_{10}\frac{A_0}{2}, \quad dx_{10} = dt_{10}\left(V_1 + dt_{10}\frac{A_0}{6}\right) = -\frac{A_0}{J_c}\left(V_0 + \frac{A_0^2}{3J_c}\right)$$

The slip either during the interval 0-1 or 1-0 is

$$S_{01} = V_L dt_{01} - dx_{01} = \frac{A_0}{J_c}\left(V_L - V_0 - \frac{A_0^2}{3J_c}\right). \tag{5-2}$$

By using equation (5-2) in calculating slip for all values of A_0 we see that S_{01} is added if $A_0 > 0$ and subtracted if $A_0 < 0,$ which is exactly what is needed.

5.3 <u>Formulae for Slip</u>

The formula for Slip in all but the case in which $A_0 > 0$ and $Slip < S_{bnd_3}$ can now be expressed as

$$Slip = S_{01} + V_L dt_{14} - dx_{14} + V_L dt_{58} - dx_{58} = S_{01} + \left[V_L - \frac{1}{2}\left(V_1 + V_4\right)\right]dt_{14} + \left[V_L - \frac{1}{2}\left(V_L + V_4\right)\right]dt_{58}$$

(5-3)

in which

$$\text{if } V_1 - V_4 \geq \frac{A_r^2}{J_c} \text{ then } dt_{14} = \frac{V_1 - V_4}{A_r} + \frac{A_r}{J_c} \text{ else } dt_{14} = 2\sqrt{\frac{V_1 - V_4}{J_c}}$$

$$\text{if } V_L - V_4 \geq \frac{A_r^2}{J_c} \text{ then } dt_{58} = \frac{V_L - V_4}{A_r} + \frac{A_r}{J_c} \text{ else } dt_{58} = 2\sqrt{\frac{V_L - V_4}{J_c}}$$

To calculate S_{bnd_1} substitute $V_4 = V_{\min}$ in equation (5-3).

To calculate S_{bnd_2} substitute $V_4 = V_1 - \dfrac{A_r^2}{J_c}$ in equation (5-3).

To calculate S_{bnd_3} substitute $V_4 = V_1$ in equation (5-3).

S_{bnd_4} will be calculated in a later paragraph.

If $S \geq S_{bnd1}$

$$S = S_{01} + \tfrac{1}{2}\left[\left(2V_L - V_1 - V_{\min}\right)dt_{14} + \left(V_L - V_{\min}\right)dt_{58}\right] + \left(V_L - V_{\min}\right)dt_{45}. \tag{5-4}$$

If $S_{bnd_2} < Slip < S_{bnd_1}$

$$Slip = S_{01} + \frac{1}{2}\left(2V_L - V_1 - V_4\right)\left(\frac{V_1 - V_4}{A_r} + \frac{A_r}{J_c}\right) + \frac{1}{2}\left(V_L - V_4\right)\left(\frac{V_L - V_4}{A_r} + \frac{A_r}{J_c}\right)$$

$$A_r\left(Slip - S_{01}\right) = \frac{1}{2}\left(2V_L - V_1 - V_4\right)\left(V_1 - V_4 + dV_r\right) + \frac{1}{2}\left(V_L - V_4\right)\left(V_L - V_4 + dV_r\right)$$

$$= V_4^2 - \frac{1}{2}\left(2V_L - V_1 + V_1 + dV_r + 2V_L + dV_r\right)V_4 + \frac{1}{2}\left[\left(2V_L - V_1\right)\left(V_1 + dV_r\right) + V_L\left(V_L + dV_r\right)\right]$$

$$A_r\left(Slip - S_{01}\right) = f\left(V_4\right) = V_4^2 - 2bV_4 + c$$

$$b = V_L + \frac{1}{2}dV_r, \quad c = \frac{1}{2}\left[\left(2V_L - V_1\right)\left(V_1 + dV_r\right) + V_L\left(V_L + dV_r\right)\right]$$

Note that $f' = 2V_4 - 2b$. So when $f' = 0 \quad V_4 = b > V_L$. But $V_4 < V_L$.

Therefore $$V_4 = b - \sqrt{b^2 - c + A_r\left(Slip - S_{01}\right)} \qquad (5\text{-}5)$$

If $S_{bnd_3} < Slip < S_{bnd_2}$ then

$$Slip - S_{01} = \left[V_L - \frac{1}{2}\left(V_1 + V_4\right)\right]2\sqrt{\frac{V_1 - V_4}{J_c}} + \left[V_L - \frac{1}{2}\left(V_L + V_4\right)\right]dt_{58} \qquad (5\text{-}6)$$

in which

$$\text{if } V_L - V_4 \geq dV_r \text{ then } dt_{58} = \frac{V_L - V_4}{A_r} + \frac{A_r}{J_c} \text{ else } dt_{58} = 2\sqrt{\frac{V_L - V_4}{J_c}}.$$

Equation (5-6) is a quartic equation if $V_L - V_4$ is large, and 6th order if small. The former can be solved exactly but not the second so it must be solved numerically.

5.4 A Numerical Solution

Since we know that

$$S = S_{bnd_2} \text{ when } V_4 = V_1 - dV_r \text{ and } S = S_{bnd_3} \text{ when } V_4 = V_1$$

the solution for V_4 is simplified.

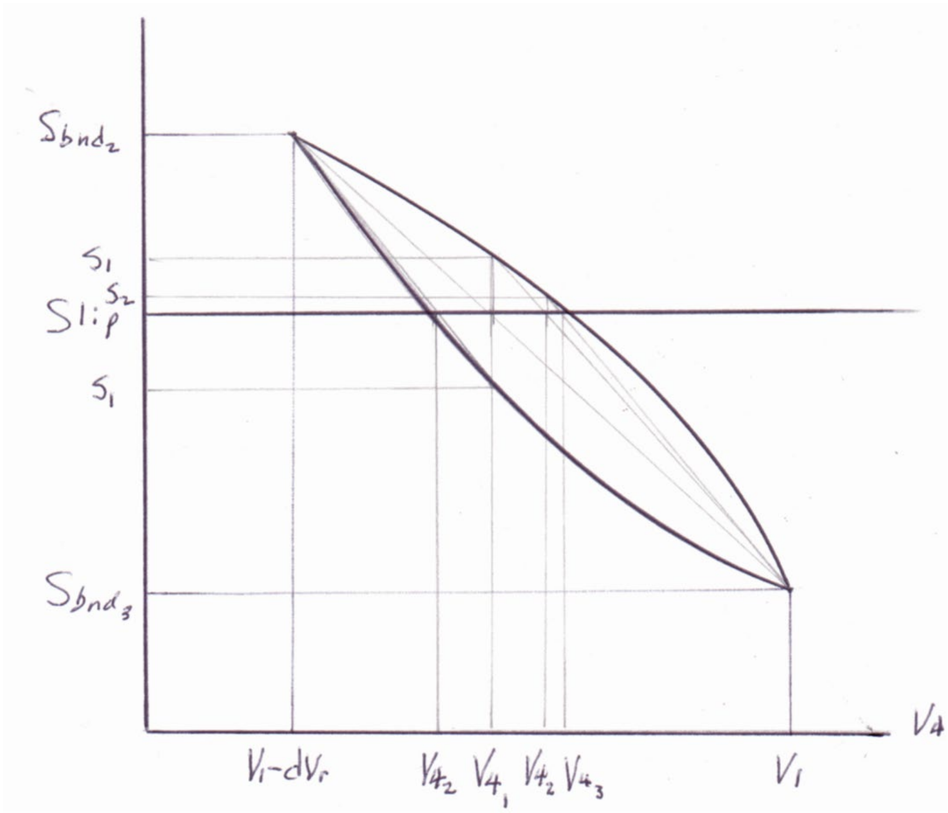


Figure 4. Numerical Solution for V_4.

Consider Figure 4, which is a graphical representation of the solution process. The two curves represent possible solution, which can have either positive or negative curvature. As a first guess, draw a straight line between the points S_{bnd_3}, V_1 and $S_{bnd_2}, V_1 - dV_r$. Then, let the first guess for V_4 be

$$V_{4_1} = V_1 - dV_r \left(\frac{Slip - S_{bnd_3}}{S_{bnd_2} - S_{bnd_3}} \right)$$

Then, from equation (5-7), calculate the corresponding slip S_1. If $S_1 > Slip$ draw line from point S_1, V_{4_1} to point S_{bnd_3}, V_1. V_{4_2} is the value of V_4 that intersects the horizontal line at *Slip*. Thus

$$V_{4_2} = V_1 - \left(V_1 - V_{4_1} \right) \left(\frac{Slip - S_{bnd_3}}{S_1 - S_{bnd_3}} \right)$$

Simarly

$$V_{4_3} = V_1 - \left(V_1 - V_{4_2} \right) \left(\frac{Slip - S_{bnd_3}}{S_2 - S_{bnd_3}} \right)$$

If $S_1 < S$ draw line from point S_1, V_{4_1} to point $S_{bnd_2}, V_1 + dV_r$. V_{4_2} is the value of V_4 that intersects the horizontal line at *Slip*. Thus

$$V_{4_2} = V_1 - dV_r + \left(V_{4_1} - V_1 + dV_r\right)\left(\frac{S_{bnd_2} - Slip}{S_{bnd_2} - S_1}\right)$$

etc. Double differentiation of equation (5-6) shows that the curvature at a specific point may be either positive or negative, thus it is necessary to consider the two cases. This iternative process can be repeated in a do-loop until a value of V_4 is found such that the corresponding *Slip* is sufficiently close to the given value of $Slip$, say within 0.001 m.

If the sequence of estimated values of S_i alternate from above and below the line from point $S_{bnd_2}, V_1 - dV_r$ to point S_{bnd_3}, V_1 then the formula for calculating V_4 must be changed. In the space $V_4, Slip$ a straight line between points S_i, V_{4i} and $S_{i+1}, V_{4_{i+1}}$ is represented by

$$Slip = S_i - \left(\frac{S_i - S_{i+1}}{V_{4_{i+1}} - V_{4_i}}\right)\left(V_4 - V_{4_1}\right)$$

Solving for V_4 we get

$$\begin{aligned} V_4 &= V_{4_1} + \left(V_{4_{i+1}} - V_{4_i}\right)\left(\frac{S_i - Slip}{S_i - S_{i+1}}\right) = \frac{V_{4_1}\left(S_i - S_{i+1}\right) + \left(V_{4_{i+1}} - V_{4_i}\right)\left(S_i - Slip\right)}{S_i - S_{i+1}} \\ &= \frac{\left(S_i - Slip\right)V_{4_{i+1}} + \left(Slip - S_{i+1}\right)V_{4_1}}{S_i - S_{i+1}} \end{aligned} \qquad (5\text{-}7)$$

So if say $S_i > Slip$ and $S_{i+1} < Slip$ equation (5-7) gives the best new estimate for V_4. This method converges quickly.

5.5 <u>The Special Case </u>$V_0 = V_L,\ A_0 = 0.$

In this case $dt_{58} = dt_{14}$. If $Slip >= S_{bnd1}$ then

$$\begin{aligned} S_{bnd_1} &= \left(V_L - V_{\min}\right)\left(\frac{V_L - V_{\min}}{A_r} + \frac{A_r}{J_c}\right) \\ Slip &= S_{bnd_1} + \left(V_L - V_{\min}\right)dt_{45}, \quad dt_{45} = \frac{S - S_{bnd_1}}{V_L - V_{\min}} \end{aligned} \qquad (5\text{-}8)$$

If $S_{bnd_2} = 2\frac{A_r^3}{J_c^2} < Slip < S_{bnd1}$ then

$$Slip = \left(V_L - V_4\right)\left(\frac{V_L - V_4}{A_r} + \frac{A_r}{J_c}\right) \tag{5-9}$$

which is a quadratic equation for V_4. Its standard form and solution are

$$V_4^2 - 2\left(V_L + \frac{1}{2}dV_r\right)V_4 + V_L\left(V_L + dV_r\right) - A_r Slip = 0$$

$$V_4 = V_L + \frac{1}{2}dV_r - \sqrt{\left(V_L + \frac{1}{2}dV_r\right)^2 + A_r Slip - V_L\left(V_L + dV_r\right)}$$

$$= V_L + \frac{1}{2}dV_r - \sqrt{\frac{1}{4}dV_r^2 + A_r Slip} \tag{5-10}$$

The minus sign is correct because V_4 must decrease when *Slip* increases. V_4 is positive when

$$\left(V_L + \frac{1}{2}dV_r\right)^2 > \frac{1}{4}dV_r^2 + A_r Slip \text{ or when } V_L\left(V_L + dV_r\right) > A_r Slip.$$

In this case *Slip* is minimum when $S_{bnd_2} = 2\frac{A_r^3}{J_c^2}$. Thus, the condition of positive V_4 is

$$V_L\left(V_L + dV_r\right) > 2dV_r^2$$

This inequality is satisfied if $V_L > dV_r = \frac{A_r^2}{J_c} = \frac{\left(0.25g0.75\right)^2}{0.25g} = 0.141g = 1.38m/s$, which will always be true.

If $S < S_{bnd_2}$

$$Slip = 2\left(V_L - V_4\right)\sqrt{\frac{V_L - V_4}{J_c}} \quad \text{or} \quad \left(V_L - V_4\right)^{3/2} = \tfrac{1}{2}SlipJ_c^{1/2}$$

$$V_4 = V_L - \left(\frac{J_c Slip^2}{4}\right)^{1/3} \tag{5-11}$$

5.6 <u>Details of solution if</u> $Slip \geq S_{bnd_3}$.

With V_0 and A_0 given we have been able to calculate time, speed and distance up to point 1, and note that in all cases for which $Slip \geq S_{bnd_3}$ that $A_1 = 0$. With V_4 known we have been able to calculate dt_{01}, dx_{01}, and V_1. and A_6. For $Slip \geq S_{bnd_3}$ we can now calculate the values of time, speed, and distance at each of the points 1 through 8 as follows:

$$\begin{aligned}
&dt_{12} = -A_2 / J_c,\ V_2 = V_1 + dt_{12}A_2 / 2,\ dx_{12} = dt_{12}\left(V_1 + dt_{12}A_2 / 6\right)\\
&dt_{34} = dt_{12}, \qquad V_3 = V_4 - dt_{34}A_2 / 2,\ dx_{34} = dt_{34}\left(V_3 + dt_{34}A_2 / 3\right)\\
&dt_{23} = \left(V_3 - V_2\right) / A_2, \quad dx_{23} = dt_{23}\left(V_2 + dt_{23}A_2 / 2\right)\\
&dt_{45} = \left(S - S_{bnd1}\right) / \left(V_L - V_{\min}\right)\\
&dt_{56} = A_6 / J_c,\ V_6 = V_4 + dt_{56}A_6 / 2,\ dx_{56} = dt_{56}\left(V_4 + dt_{56}A_6 / 6\right)\\
&dt_{78} = dt_{56}, \qquad V_7 = V_L - dt_{78}A_6 / 2,\ dx_{78} = dt_{78}\left(V_7 + dt_{78}A_6 / 3\right)\\
&dt_{67} = \left(V_7 - V_6\right) / A_6, \quad dx_{67} = dt_{67}\left(V_6 + dt_{67}A_6 / 2\right)
\end{aligned} \tag{5-12}$$

$$\begin{aligned}
&t_1 = t_0 + dt_{01}, t_2 = t_1 + dt_{12}, t_3 = t_2 + dt_{23}, t_4 = t_3 + dt_{34}\\
&t_5 = t_4 + dt_{45}, t_6 = t_5 + dt_{56}, t_7 = t_6 + dt_{67}, t_8 = t_7 + dt_{78}\\
&x_1 = x_0 + dx_{01}, x_2 = x_1 + dx_{12}, x_3 = x_2 + dx_{23}, x_4 = x_3 + dx_{34}\\
&x_5 = x_4 + dx_{45}, x_6 = x_5 + dx_{56}, x_7 = x_6 + dx_{67}, x_8 = x_7 + dx_{78}
\end{aligned}$$

5.7 The case $S_{\min} = S_{bnd_4} \leq S < S_{bnd3}$.

In Figure 5, we show a close-up of the region between S_{bnd_3} and $S_{\min} = S_{bnd_4}$ and show a slip-curve at an intermediate location intersecting the S_{bnd_3} curve at a point labeled “1” at which the acceleration is $A_1 > 0$ where A_1 takes any value between zero and A_0. The point where acceleration is zero in the S_{bnd_3} curve is also labeled “1” indicating that point “1” moves from $A_1 = 0$ to $A_1 = A_0$. If $A_0 > 0$ there will be cases in which *Slip* will lie between these boundaries. The curve of speed vs. time in these cases can be divided into four parts, which in terms of the general notation for slip transitions we separate by points 1, 6, 7, and 8. Points 1, 2, 3, 4, 5 coincide, therefore

$$dt_{12} = dt_{23} = dt_{34} = dt_{45} = 0.$$

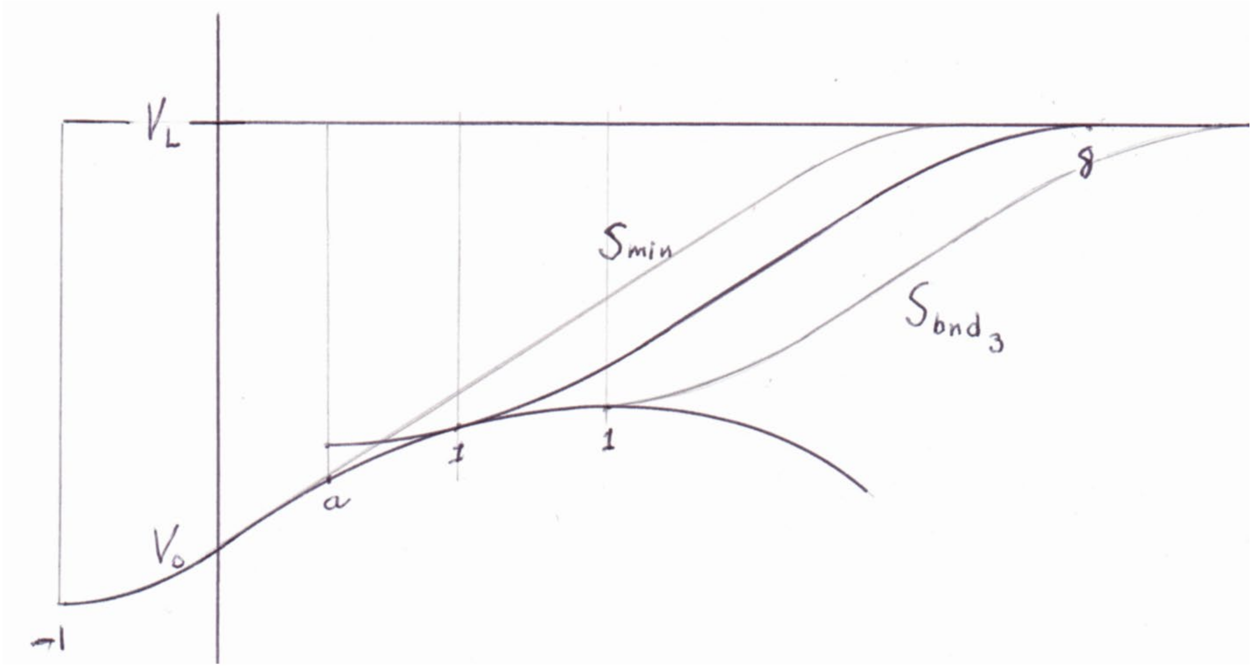


Figure 5. Slip between S_{bnd_3} and $S_{bnd_4} = S_{\min}$.

In the interval from point 0 to point 1, jerk is negative for all cases. From point 1 to point 6 jerk is positive, from point 6 to point 7 jerk is zero, and from point 7 to point 8 at line speed jerk is negative. Thus, for any value of *Slip* between the two boundaries

$$dt_{01} = \frac{A_1 - A_0}{-J_c}, \quad V_1 = V_0 + dt_{01}\left(\frac{A_1 + A_0}{2}\right), \quad dx_{01} = dt_{01}\left[V_0 + dt_{01}\left(\frac{2A_0 + A_1}{6}\right)\right] \quad (5\text{-}13)$$

$S_{\min}$ occurs when $A_1 = A_0$ and S_{bnd_3} occurs when $A_1 = 0.$ From Figure 5

$$\mathrm{S}_{\min} = \frac{1}{2}\left(V_L - V_{-1}\right)T_{-18} - \left(V_L dt_{-10} - dx_{-10}\right)$$

in which

$$A_{-1} = 0, \quad dt_{-10} = \frac{A_0}{J_c}, \quad V_{-1} = V_0 - dt_{-10}\frac{A_0}{2} = V_0 - \frac{A_0^2}{2J_c}, \quad dx_{-10} = dt_{-10}\left(V_{-1} + dt_{-10}\frac{A_0}{6}\right)$$

Thus

$$S_{\min} = \frac{1}{2}\left(V_L - V_0 + \frac{A_0^2}{2J_c}\right)T_{-18} - \frac{A_0}{J_c}\left(V_L - V_0 + \frac{A_0^2}{3J_c}\right) \quad (5\text{-}14)$$

in which with $\Delta V = V_L - V_0 + \dfrac{A_0^2}{2J_c}$

$$\text{if } \Delta V \geq dV_r \text{ then } T_{-18} = \frac{\Delta V}{A_r} + \frac{A_r}{J_c} \text{ else } T_{-18} = 2\sqrt{\frac{\Delta V}{J_c}}.$$

From Figure 5 with $A_1 = 0$

$$S_{bnd_3} = \frac{1}{2}\left(V_L - V_1\right)T_{18} + V_L dt_{01} - dx_{01}$$

For the interval from 0 to 1with $A_1 = 0$ we have

$$dt_{01} = \frac{-A_0}{-J_c}, \quad V_1 = V_0 + dt_{01}\frac{A_0}{2} = V_0 + \frac{A_0^2}{2J_c}, \quad dx_{01} = dt_{01}\left(V_0 + dt_{01}\frac{A_0}{3}\right)$$

Thus

$$S_{bnd_3} = \frac{1}{2}\left(V_L - V_0 - \frac{A_0^2}{2J_c}\right)T_{18} + \frac{A_0}{J_c}\left(V_L - V_0 - \frac{A_0^2}{3J_c}\right) \qquad (5\text{-}15)$$

in which with $\Delta V = V_L - V_0 - \dfrac{A_0^2}{2J_c}$

$$\text{if } \Delta V \geq dV_r \text{ then } T_{18} = \frac{\Delta V}{A_r} + \frac{A_r}{J_c} \text{ else } T_{18} = 2\sqrt{\frac{\Delta V}{J_c}}.$$

If $S_{\min} < Slip < S_{bnd3}$

$$Slip\left(A_1\right) = \frac{1}{2}\left(V_L - V_a\right)T_{a8} + \left(V_L dt_{0a} - dx_{0a}\right) + dx_{a1}^{+} - dx_{a1}^{-}$$

in which

$$dt_{0a} = dt_{01} - dt_{a1}$$

$$dt_{01} = \frac{A_1 - A_0}{-J_c}, \quad V_1 = V_0 + dt_{01}\left(\frac{A_0 + A_1}{2}\right) = V_0 + \frac{A_0^2 - A_1^2}{2J_c}$$

$$dt_{a1} = \frac{A_1}{J_c}, \quad V_a = V_1 - dt_{a1}\frac{A_1}{2} = V_1 - \frac{A_1^2}{2J_c} = V_0 + \frac{A_0^2}{2J_c} - \frac{A_1^2}{J_c}$$

$$dt_{0a} = \frac{A_0 - A_1}{J_c} - \frac{A_1}{J_c} = \frac{A_0 - 2A_1}{J_c}$$

$$dx_{a1}^+ = \frac{A_1}{J_c}\left(V_a + \frac{A_1^2}{6J_c}\right) = \frac{A_1}{J_c}\left(V_0 + \frac{A_0^2}{2J_c} - \frac{A_1^2}{J_c} + \frac{A_1^2}{6J_c}\right) = \frac{A_1}{J_c}\left(V_0 + \frac{A_0^2}{2J_c} - \frac{5A_1^2}{6J_c}\right)$$

$$dx_{0a} = dt_{0a}\left[V_0 + dt_{0a}\left(\frac{2A_0 + A_a^-}{6}\right)\right], \quad A_a^- = A_0 - dt_{0a}J_c = 2A_1, \quad V_a^- = V_0 + A_0 dt_{0a} - J_c\frac{dt_{0a}^2}{2}$$

$$dx_{0a} = \left(\frac{A_0 - 2A_1}{J_c}\right)\left[V_0 + \left(\frac{A_0 - 2A_1}{J_c}\right)\left(\frac{A_0 + A_1}{3}\right)\right] = \left(\frac{A_0 - 2A_1}{J_c}\right)\left(V_0 + \frac{A_0^2 - A_1A_0 - 2A_1^2}{3J_c}\right)$$

$$dx_{a1}^- = dt_{a1}\left[V_a^- + dt_{a1}\left(\frac{2A_a^- + A_1}{6}\right)\right] = \frac{A_1}{J_c}\left[V_0 + A_0 dt_{0a} - J_c\frac{dt_{0a}^2}{2} + \frac{A_1}{J_c}\left(\frac{4A_1 + A_1}{6}\right)\right]$$

$$= \frac{A_1}{J_c}\left[V_0 + \left(\frac{A_0 - 2A_1}{2J_c}\right)\left(2A_0 - A_0 + 2A_1\right) + \frac{5A_1^2}{6J_c}\right]$$

$$dx_{a1}^+ - dx_{a1}^- = \frac{A_1}{J_c}\left[V_0 + \frac{A_0^2}{2J_c} - \frac{5A_1^2}{6J_c} - V_0 - \left(\frac{A_0^2 - 4A_1^2}{2J_c}\right) - \frac{5A_1^2}{6J_c}\right] = \frac{A_1}{J_c}\left(\frac{A_1^2}{3J_c}\right)$$

Therefore

$$Slip\left(A_1\right) = \frac{1}{2}\left(V_L - V_a\right)T_{a8} + \left(\frac{A_0 - 2A_1}{J_c}\right)\left(V_L - V_0 - \frac{A_0^2 - A_1A_0 - 2A_1^2}{3J_c}\right) + \frac{A_1^3}{3J_c^2} \qquad (6.5\text{-}16)$$

in which $V_a = V_0 + \frac{A_0^2}{2J_c} - \frac{A_1^2}{J_c}.$

$$Slip\left(A_0\right) = S_{bnd_4} = S_{\min} = \frac{1}{2}\left(V_L - V_a\right)T_{a8} - \frac{A_0}{J_c}\left(V_L - V_0 + \frac{2A_0^2}{3J_c}\right) + \frac{A_0^3}{3J_c^2}$$

$$= \frac{1}{2}\left(V_L - V_a\right)T_{a8} - \frac{A_0}{J_c}\left(V_L - V_0 + \frac{A_0^2}{3J_c}\right)$$

in which $V_a = V_0 - \frac{A_0^2}{2J_c}.$

$$Slip\left(0\right) = S_{bnd_3} = \frac{1}{2}\left(V_L - V_a\right)T_{a8} + \frac{A_0}{J_c}\left(V_L - V_0 - \frac{A_0^2}{3J_c}\right)$$

in which $V_a = V_0 + \frac{A_0^2}{2J_c}.$

In the equation for $Slip\left(A_1\right)$ with $\Delta V = V_L - V_a$

if $\Delta V \geq dV_r$ then $T_{a8} = \frac{\Delta V}{A_r} + \frac{A_r}{J_c}$ else $T_{a8} = 2\sqrt{\frac{\Delta V}{J_c}}.$

$Slip\left(A_1\right)$, Equation (5-16), is either a quartic in A_1 6th order. It will be solved numerically in a manner like that used with Figure 4. We need to find the value of A_1 that corresponds to a given value of slip. We know that $A_1 = 0$ gives the boundary S_{bnd_3} and the value $A_1 = A_0$ gives the boundary S_{bnd_4}. Let the first guess to the correct value of A_1 be

$$A_{1_1} = A_0\left(\frac{S_{bnd_3} - Slip}{S_{bnd_3} - S_{bnd_4}}\right) \quad \rightarrow \quad Slip_1$$

Using this value of A_1 calculate $Slip_1$ from equation (5-16). If $Slip_1$ is less than the required slip *Slip*, draw a line from the point $A_{1_1}, Slip_1$ to $0, S_{bnd_3}$ and calculate a new guess from the equation:

$$A_{1_2} = A_{1_1}\left(\frac{S_{bnd_3} - Slip}{S_{bnd_3} - Slip_1}\right) \quad \rightarrow \quad Slip_2$$

and repeat in a Do-Loop until the error is sufficiently small. If $Slip_1$ is greater than the required slip *S* draw a line from the point $A_{1_1}, Slip_1$ to A_0, S_{bnd_4} and calculate a new guess from the equation

$$A_{1_2} = A_0 - \left(A_0 - A_{1_1}\right)\left(\frac{Slip - S_{bnd_4}}{Slip_1 - S_{bnd_4}}\right) \quad \rightarrow \quad Slip_2$$

Again, in a Do-Loop, convergence is rapid. In only a few cycles the error is reduced to less than one millimeter.

5.8 Slip Remaining

In merge control, to command vehicles to slip the least amount needed to avoid violating the headway criterion, it is necessary to take into account that vehicles reaching the merge command point may be slipping. To command further slip without reducing throughput any more than necessary we must know the amount of slip each vehicle has remaining. To calculate slip remaining (SR), consider Figure 3. The slip remaining at $t = t_7$ is the same as the distance traveled in moving at constant jerk *J* from rest to a time $t_8 - t_7 \equiv dt_{78}$. Consider then that at constant jerk we have

$$A = Jt, \quad V = J\frac{t^2}{2}, \quad X = J\frac{t^3}{6} = \frac{Vt}{3}$$

Thus, from the geometry of Figure 3 we have

$$
\begin{aligned}
SR_7 &= \left(V_L - V_7\right) dt_{78} / 3 \\
SR_6 &= SR_7 + \left[V_L - \left(V_6 + V_7\right)/2\right] dt_{67} \\
SR_5 &= SR_6 + \left[V_L - V_5 - \left(V_6 - V_5\right)/3\right] dt_{56} \\
SR_4 &= SR_5 + \left(V_L - V_4\right) dt_{45} \\
SR_3 &= SR_4 + \left[V_L - V_4 - \left(V_3 - V_4\right)/3\right] dt_{34} \\
SR_2 &= SR_3 + \left[V_L - \left(V_2 + V_3\right)/2\right] dt_{23} \\
SR_1 &= SR_2 + \left[V_L - V_1 + \left(V_1 - V_2\right)/3\right] dt_{12} \\
SR_0 &= SR_1 + \left[V_L - V_1 + \left(V_1 - V_0\right)/3\right] dt_{02}
\end{aligned}
$$

If $Slip < S_{bnd_3}$ SR_5 and SR_0 must be modified because $A_1 > 0$. In this case

$$dt_{12} = dt_{23} = dt_{34} = dt_{45} = 0$$

With $A_1 > 0$ and both A_1 and A_6 already calculated we have

$$
\begin{aligned}
dt_{01} &= \frac{A_1 - A_0}{-J_c},\ V_1 = V_0 + dt_{01}\left(\frac{A_1 + A_0}{2}\right),\ dx_{01} = dt_{01}\left[V_0 + dt_{01}\left(\frac{2A_0 + A_1}{6}\right)\right] \\
SR_0 &= SR_1 + V_L dt_{01} - dx_{01} \\
dt_{16} &= \frac{A_6 - A_1}{J_c},\ dx_{16} = dt_{16}\left[V_1 + dt_{16}\left(\frac{2A_1 + A_6}{6}\right)\right] \\
SR_5 &= SR_6 + V_L dt_{16} - dx_{16}
\end{aligned}
$$

Then the following code will calculate the slip remaining at any time $\Delta t = t - t_0$, in which the $t_i, i = 1,...,8$ are measured from t_0, the start of the slip transition.

```
if $\Delta t < t_1$ then
    $SR = SR_1 + \left[V_L - V_1 + \left(V_1 - V\right)/3\right]\left(t_1 - \Delta t\right)$
elseif $\Delta t < t_2$ then
    $SR = SR_1 - \left[V_L - V_1 + \left(V_1 - V\right)/3\right]\left(\Delta t - t_1\right)$
elseif $\Delta t < t_3$ then
    $SR = SR_3 + \left[V_L - \left(V + V_3\right)/2\right]\left(t_3 - \Delta t\right)$
elseif $\Delta t < t_4$ then
    $SR = SR_4 + \left[V_L - V_4 - \left(V - V_4\right)/3\right]\left(t_4 - \Delta t\right)$
elseif $\Delta t < t_5$ then
    $SR = SR_5 + \left[V_L - V_5\right]\left(t_5 - \Delta t\right)$
elseif $\Delta t < t_6$ then
    $SR = SR_5 - \left[V_L - V_5 - \left(V - V_5\right)/3\right]\left(\Delta t - t_5\right)$
elseif $\Delta t < t_7$ then
    $SR = SR_7 + \left[V_L - \left(V + V_7\right)/2\right]\left(t_7 - \Delta t\right)$
elseif $\Delta t < t_8$ then
    $SR = \left(V_L - V\right)\left(t_8 - \Delta t\right)/3$
end if
```

6. Speed-Change Transitions

The speed-change transitions start at a speed V_0 and acceleration A_0 and end at a speed $V_f \le V_L$ and acceleration $A_f = 0$. They take into account, as derived in Section 3, that above a speed αV_L the maximum acceleration is reduced from the value A_c at αV_L to βA_c at $V_L - V_b$, where $V_b = (\beta A_c)^2 / 2J_c$.

The Relationship between V and A above αV_L.

We can assume this relationship is linear. Thus let

$$|A| = A_c - \left(\frac{V - \alpha V_L}{V_L - V_b - \alpha V_L}\right) A_c (1 - \beta)$$

which meets the stipulated end conditions. $|A|$ means the absolute value of A, which means that the above equation applies for decreasing as well as increasing speed. Solving for V, we have

$$V = \alpha V_L + \left[\frac{(1-\alpha)V_L - V_b}{1-\beta}\right]\left(1-\frac{|A|}{A_c}\right). \tag{6-1}$$

The Cases

We need to consider six cases, which are defined in the following table. To follow these cases in detail, the reader must draw diagrams of each one.

Case	$V_f > V_0$
1	$V_0 - A_0^2/2J_c \le \alpha V_L, \quad V_f > \alpha V_L$
2	$V_0 - A_0^2/2J_c > \alpha V_L, \quad V_f > \alpha V_L$
3	$V_0 - A_0^2/2J_c \le \alpha V_L, \quad V_f \le \alpha V_L$
	$V_f < V_0$
4	$V_0 + A_0^2/2J_c > \alpha V_L, \quad V_f \le \alpha V_L$
5	$V_0 + A_0^2/2J_c > \alpha V_L, \quad V_f > \alpha V_L$
6	$V_0 + A_0^2/2J_c \le \alpha V_L, \quad V_f \le \alpha V_L$

First, however, note that if $dV \equiv \left|V_f - V_0\right| + A_0^2/2J_c \le A_c^2/J_c$ we can bypass these cases and compute A_1, V_1 as follows:

$$A_1 = SGN(V_f - V_0)\sqrt{J_c dV}, \quad V_1 = V_0 - SGN(V_f - V_0)\frac{A_0^2 - A_1^2}{2J_c}$$

Case 1

In this case, positive jerk J_c is applied during interval 0-1 until either the acceleration reaches A_c or the speed reaches αV_L. If the former case, constant acceleration A_c is applied during interval 1-2 until the speed reaches αV_L. In the later case there is no interval 1-2 and acceleration $A_1 = A_2 < A_c$. At point 2 negative jerk J_n is applied (interval 2-3) until point 3 is reached, at which time maximum negative jerk J_c is applied (interval 3-4) until the speed and acceleration simultaneously reach V_f and zero, respectively. However, if $V_f - \alpha V_L$ is too small, speed V_2 must be reduced from αV_L as shown below.

Unlike the derivation of equations (3-1) at this point we don't know either A_3 or V_3. It is best first to list the generic equations for Case 1. They are

$$\begin{gathered}
dt_{01} = \frac{A_1 - A_0}{J_c},\ V_1 = V_0 + dt_{01}\left(\frac{A_1 + A_0}{2}\right),\ dx_{01} = dt_{01}\left[V_0 + dt_{01}\left(\frac{2A_0 + A_1}{6}\right)\right] \\
A_2 = A_1,\ dt_{12} = \frac{V_2 - V_1}{A_c},\ dx_{12} = dt_{12}\left[V_1 + dt_{12}\frac{A_c}{2}\right],\ V_2 \le \alpha V_L;\quad A_1 = ?
\end{gathered} \tag{6-2}$$

$$\begin{gathered}
dt_{34} = \frac{A_3}{J_c},\ V_3 = V_f - dt_{34}\frac{A_3}{2},\ dx_{34} = dt_{34}\left(V_3 + dt_{34}\frac{A_3}{3}\right);\quad A_3, V_3 = ? \\
dt_{23} = \frac{V_3 - V_2}{\frac{1}{2}(A_2 + A_3)} = \frac{A_2 - A_3}{J_n}, J_n = \frac{A_2^2 - A_3^2}{2(V_3 - V_2)},\ dx_{23} = dt_{23}\left[V_2 + dt_{23}\left(\frac{2A_2 + A_3}{6}\right)\right]
\end{gathered}$$

If $V_0 - A_0^2 / 2J_c + A_c^2 / 2J_c \le \alpha V_L$ then, using the first two of the first line of equations (6-2),

$$\begin{gathered}
A_1 = A_2 = A_c \\
V_1 = V_0 - \frac{A_0^2}{2J_c} + \frac{A_c^2}{2J_c},\quad V_2 = \alpha V_L
\end{gathered} \tag{6-3}$$

If $V_0 - A_0^2 / 2J_c + A_c^2 / 2J_c > \alpha V_L$ then

$$\begin{gathered}
V_1 = V_2 = \alpha V_L \\
\alpha V_L = V_0 + \frac{A_1^2 - A_0^2}{2J_c},\quad A_1 = A_2 = \sqrt{2J_c\left(\alpha V_L - V_0\right) + A_0^2}
\end{gathered} \tag{6-4}$$

Now consider A_3. If $V_f - \alpha V_L \ge \frac{A_2^2}{2J_c}$ then, using equation (1),

$$\begin{gathered}
V_3 = V_f - \frac{A_3^2}{2J_c} = \alpha V_L + Q\left(1 - \frac{A_3}{A_c}\right),\quad Q = \frac{(1-\alpha)V_L - V_b}{1-\beta} = \left(\frac{1-\alpha}{1-\beta}\right)V_L - \frac{\beta^2}{1-\beta}\frac{A_c^2}{2J_c} \\
A_3^2 - 2bA_3 + c_f = 0,\quad b = \frac{J_c}{A_c}Q,\quad c_f = 2J_c\left(\alpha V_L + Q - V_f\right)
\end{gathered} \tag{6-5}$$

$$A_3 = b \pm \sqrt{b^2 - c_f}, \quad V_f = \frac{A_3^2}{2J_c} - \frac{Q}{A_c}A_3 + \alpha V_L + Q, \quad \frac{\partial V_f}{\partial A_3} = \frac{A_3}{J_c} - \frac{Q}{A_c}$$
$$\frac{\partial V_f}{\partial A_3} = 0 \rightarrow A_{3_{\min}} = \frac{J_c}{A_c}Q = \frac{J_c}{A_c}\left(\frac{1-\alpha}{1-\beta}\right)V_L - \frac{\beta^2}{1-\beta}\frac{A_c}{2} \qquad \text{(6-6)}$$

The quantity V_f can, as shown in equations (6-6), be expressed as a parabolic function of A_3, which has a single minimum point at the value of $A_{3_{\min}}$ given in equations (6-6). If we assume, as we do in the system that $\alpha = \beta = 1/2$ and $A_c = 0.25g, J_c = 0.25g/s$ then

$$A_{3_{\min}} = V_L - \frac{1}{16}g$$

The maximum value of $A_{3_{\min}}$ is 0.25g. $A_{3_{\min}}$ reaches this value if $V_L = \frac{5}{16}g = 3.07$ m/s, which is substantially lower than any practical line speed. Thus, of the two solutions for A_3, only the lower one has physical meaning. Thus

$$A_3 = b - \sqrt{b^2 - c_f} \qquad \text{(6-7)}$$

Can the quantity $b^2 - c_f$ ever be negative? We see from equation (6-5) that

$$b^2 - c_f = J_c Q\left(\frac{J_c}{A_c}\frac{Q}{A_c} - 2\right) + 2J_c\left(V_f - \alpha V_L\right)$$

The smallest value V_f can have in Case 1 is αV_L. Assuming this value, we see that the radical is certainly positive if

$$Q = \left(\frac{1-\alpha}{1-\beta}\right)V_L - \frac{\beta^2}{1-\beta}\frac{A_c^2}{2J_c} \geq 2\frac{A_c^2}{J_c}.$$

Using the values given after equation (6-6), we see that the radical is always positive if

$$V_L > \frac{9}{16}g = 5.52 \text{ m/s}.$$

Since this value of line speed is less than practical values, the radical in equation (6-7) is always positive.

Now, if $V_f - \alpha V_L < \frac{A_2^2}{2J_c}$ there are two cases:

If $V_f - V_0 + \frac{A_0^2}{2J_c} \geq \frac{A_c^2}{J_c}$

$$A_2 = A_3 = A_c, \quad V_2 = V_3 = V_f - \frac{A_c^2}{2J_c} \tag{6-8}$$

If $V_f - V_0 + \frac{A_0^2}{2J_c} < \frac{A_c^2}{J_c}$

$$\begin{aligned} &V_f - V_0 + \frac{A_0^2}{2J_c} = \frac{A_2^2}{J_c}, \quad A_2 = A_3 = A_1 = \sqrt{J_c\left(V_f - V_0 + \frac{A_0^2}{2J_c}\right)} \\ &V_1 = V_2 = V_3 = \frac{1}{2}\left(V_f + V_0 - \frac{A_0^2}{2J_c}\right) \end{aligned} \tag{6-9}$$

Case 2

From point 0 to point 1 positive jerk J_c is applied up to an acceleration A_1 that will be calculated using equation (6-1). Points 1 and 2 are at the same location. From point 2 to point 3 negative jerk J_n is applied until at point 3 maximum negative jerk J_c must be applied until at point 4 the speed reaches V_f just as the acceleration vanishes. The acceleration and speed at point 3 are calculated from equations (6-7) and (6-5) respectively. The equation set is as follows.

$$\begin{aligned} &dt_{01} = \frac{A_1 - A_0}{J_c}, \; V_1 = V_0 + dt_{01}\frac{A_1 + A_0}{2}, \; dx_{01} = dt_{01}\left(V_0 + dt_{01}\frac{2A_0 + A_1}{6}\right) \\ &dt_{12} = 0, \quad A_1 = A_2 \\ &dt_{34} = \frac{A_3}{J_c}, \quad V_3 = V_f - dt_{34}\frac{A_3}{2}, \; dx_{34} = dt_{34}\left(V_3 + dt_{34}\frac{A_3}{3}\right), \quad A_3, V_3 = ? \\ &dt_{23} = \frac{V_3 - V_2}{\frac{1}{2}(A_2 + A_3)} = \frac{A_2 - A_3}{J_n}, J_n = \frac{A_2^2 - A_3^2}{2(V_3 - V_2)}, dx_{23} = dt_{23}\left(V_2 + dt_{23}\frac{2A_2 + A_3}{6}\right) \end{aligned} \tag{6-10}$$

To find the values of A_1 and V_1 note from the first row of equation set (6-10) that

$$V_1 = V_0 + \frac{A_1^2 - A_0^2}{2J_c}. \tag{6-11}$$

By substituting equation (6-11) into equation (6-1) and letting $|A| = A_1$ we get

$$V_0 - \frac{A_0^2}{2J_c} + \frac{A_1^2}{2J_c} = \alpha V_L + Q\left(1 - \frac{A_1}{A_c}\right) \tag{6-12}$$

which can be written in the form

$$A_1^2 + 2bA_1 - c_{0p} = 0 \tag{6-13}$$

where b is given in the equation set (6-5) and $c_{0p} = 2J_c\left(\alpha V_L + Q - V_0 + A_0^2 / 2J_c\right)$. Equation (6-10) has one positive solution:

$$A_1 = -b + \sqrt{b^2 + c_{0p}} \tag{6-14}$$

If $V_f - V_0 + \dfrac{A_0^2}{2J_c} \geq \dfrac{A_c^2}{J_c}$ then

$A_1 = A_2$ from (6-14), $V_1 = V_2$ from (6-11); A_3 from (6-7), V_3 from (6-5).

If $V_f - V_0 + \dfrac{A_0^2}{2J_c} < \dfrac{A_c^2}{J_c}$ then

$$V_f - V_0 + \frac{A_0^2}{2J_c} < \frac{A_c^2}{J_c}$$
$$A_1 = A_2 = A_3 = \sqrt{J_c\left(V_f - V_0 + \frac{A_0^2}{2J_c}\right)},\quad V_1 = V_2 = V_3 = \frac{1}{2}\left(V_f + V_0 - \frac{A_0^2}{2J_c}\right)$$

Case 3

From point 0 to point 1 positive jerk J_c is applied to A_1. if $A_1 = A_c$ acceleration is constant from point 1 to point 2, which is the same as point 3. Finally maximum negative jerk $-J_c$ is applied until the speed reaches V_f while acceleration vanishes. The equation set is as follows:

$$dt_{01} = \frac{A_1 - A_0}{J_c},\ V_1 = V_0 + dt_{01}\frac{A_1 + A_0}{2},\ dx_{01} = dt_{01}\left(V_0 + dt_{01}\frac{2A_0 + A_1}{6}\right)$$
$$dt_{34} = \frac{A_3}{J_c},\quad V_3 = V_f - dt_{34}\frac{A_3}{2},\ dx_{34} = dt_{34}\left(V_3 + dt_{34}\frac{A_3}{3}\right),\ V_2 = V_3,\ A_3 = A_2 = A_1 \tag{6-15}$$
$$dt_{23} = 0$$
$$dt_{12} = \frac{V_2 - V_1}{A_1},\quad dx_{12} = dt_{12}\left(V_1 + dt_{12}\frac{A_1}{2}\right)$$

If $V_f - V_0 + \dfrac{A_0^2}{2J_c} \geq \dfrac{A_c^2}{J_c}$ then

$$A_1 = A_2 = A_3 = A_c$$
$$V_1 = V_0 - \frac{A_0^2}{2J_c} + \frac{A_c^2}{2J_c}, \quad V_2 = V_3 = V_f - \frac{A_c^2}{2J_c}$$

If $V_f - V_0 + \frac{A_0^2}{2J_c} < \frac{A_c^2}{J_c}$ then

$$A_1 = A_2 = A_3 = \sqrt{J_c\left(V_f - V_0 + \frac{A_0^2}{2J_c}\right)}$$
$$V_1 = V_2 = V_3 = \frac{1}{2}\left(V_1 + V_0 - \frac{A_0^2}{2J_c}\right)$$

Case 4

From point 0 to point 1 negative jerk J_c is applied until the negative acceleration at point 1 satisfies equation (6-1). Then a small negative jerk J_n is applied between points 1 and 2 until the acceleration reaches $-A_c$ at point 2, where the speed is αV_L. Negative acceleration $-A_c$ is continued from point 2 to point 3 and then positive jerk J_c is applied from point 3 to point 4, at which speed reaches V_f just as acceleration vanishes. The equation set is as follows:

$$\begin{aligned}
&dt_{01} = \frac{A_1 - A_0}{-J_c}, \; V_1 = V_0 + dt_{01}\frac{A_1 + A_0}{2}, \; dx_{01} = dt_{01}\left(V_0 + dt_{01}\frac{2A_0 + A_1}{6}\right), \; A_1, V_1 = ? \\
&dt_{12} = \frac{V_2 - V_1}{\frac{1}{2}(A_1 + A_2)} = \frac{A_2 - A_1}{-J_n}, \; J_n = \frac{A_2^2 - A_1^2}{2(V_1 - V_2)}, \; dx_{12} = dt_{12}\left(V_1 + dt_{12}\frac{2A_1 + A_2}{6}\right), \; V_2 = \alpha V_L \\
&dt_{34} = \frac{-A_3}{J_c}, \quad V_3 = V_f - dt_{34}\frac{A_3}{2}, \; dx_{34} = dt_{34}\left(V_3 + dt_{34}\frac{A_3}{3}\right), \; A_3 = A_2 \\
&dt_{23} = \frac{V_3 - V_2}{A_2}, \quad dx_{23} = dt_{23}\left(V_2 + dt_{23}\frac{A_2}{2}\right)
\end{aligned} \tag{6-16}$$

We need first to consider the portion of the curve below αV_L. Then

If $\alpha V_L - V_f \geq \frac{A_c^2}{2J_c}$

$$\begin{aligned}
&A_3 = A_c \\
&V_3 = V_f + \frac{A_c^2}{2J_c}
\end{aligned} \tag{6-17}$$

If $\alpha V_L - V_f < \dfrac{A_c^2}{2J_c}$

$$\begin{aligned} A_3 &= \sqrt{2J_c\left(\alpha V_L - V_f\right)} \\ V_3 &= \alpha V_L \end{aligned}$$

(6-18)

In both of these cases $V_2 = \alpha V_L$ and $A_2 = A_3$.

To compute A_1 and V_1 we need to consider two cases.

Case 1: $V_0 + \dfrac{A_0^2}{2J_c} - \dfrac{A_c^2}{2J_c} > \alpha V_L$.

In this case, set V_1 from the first row of equation set (6-16) equal to V in equation (6-1). Thus

$$V_1 = V_0 + \frac{A_0^2 - A_1^2}{2J_c} = \alpha V_L + Q\left(1 - \frac{|A_1|}{A_c}\right) \tag{6-19}$$

which gives the following quadratic equation for A_1.

$$A_1^2 - 2b|A_1| + c_{0m} = 0 \tag{6-20}$$

in which b is found from equation set (6-5) and

$$c_{0m} = 2J_c\left(\alpha V_L + Q - V_0 - \frac{A_0^2}{2J_c}\right). \tag{6-211}$$

Thus

$$\begin{aligned} A_1 &= -b + \sqrt{b^2 - c_{0m}} \\ V_1 &= V_0 + \frac{A_0^2}{2J_c} - \frac{A_1^2}{2J_c} \end{aligned} \tag{6-21}$$

which takes into account that $A_1 < 0$.

Case 2: $V_0 + \dfrac{A_0^2}{2J_c} - \dfrac{A_c^2}{2J_c} \le \alpha V_L$.

Now there are two subcases:

Case 2.1: $V_0 + \frac{A_0^2}{2J_c} - V_f \ge \frac{A_c^2}{J_c}$

$$\begin{aligned} A_1 &= A_2 = A_3 = A_c \\ V_1 &= V_2 = V_0 + \frac{A_0^2}{2J_c} - \frac{A_c^2}{2J_c} \end{aligned} \tag{6-23}$$

Case 2.2: $V_0 + \frac{A_0^2}{2J_c} - V_f < \frac{A_c^2}{J_c}$

$$\begin{aligned} A_1 &= A_2 = A_3 = -\sqrt{J_c\left(V_0 + \frac{A_0^2}{2J_c} - V_f\right)} \\ V_1 &= V_2 = V_3 = \frac{1}{2}\left(V_0 + \frac{A_0^2}{2J_c} + V_f\right) \end{aligned} \tag{6-24}$$

Case 5

Negative jerk is applied from point 0 to point 1, at which point A_1 and V_1 are determined exactly as in Case 4. Negative jerk J_n is then applied from point 1 to point 2 (points 2 and 3 are coincident) until at point 3 maximum positive jerk J_c is applied until the speed reaches V_f just as acceleration vanishes. The determining equation set is as follows:

$$\begin{aligned} & dt_{01} = \frac{A_1 - A_0}{-J_c},\ V_1 = V_0 + dt_{01}\left(\frac{A_1 + A_0}{2}\right),\ dx_{01} = dt_{01}\left[V_0 + dt_{01}\left(\frac{2A_0 + A_1}{6}\right)\right];\ A_1, V_1 = ? \\ & dt_{12} = \frac{V_2 - V_1}{\frac{1}{2}(A_2 + A_1)} = \frac{A_2 - A_1}{-J_n},\ J_n = \frac{A_2^2 - A_1^2}{2(V_1 - V_2)},\ dx_{12} = dt_{12}\left[V_1 + dt_{12}\left(\frac{2A_1 + A_2}{6}\right)\right] \\ & dt_{23} = 0, \quad A_2 = A_3 \\ & dt_{34} = \frac{-A_3}{J_c},\ V_3 = V_f - dt_{34}\frac{A_3}{2},\ dx_{34} = dt_{34}\left(V_3 + dt_{34}\frac{A_3}{3}\right);\ A_3, V_3 = ? \end{aligned} \tag{6-25}$$

To determine A_3, V_3 note that

$$\begin{aligned} & V_3 = V_f + \frac{A_3^2}{2J_c} = \alpha V_L + Q\left(1 - \frac{|A_3|}{A_c}\right) \quad \rightarrow \quad A_3^2 + 2\frac{Q}{A_c / J_c}|A_3| - 2J_c\left(\alpha V_L + Q - V_f\right) = 0 \\ & A_3^2 + 2b|A_3| - c_f = 0 \quad \rightarrow \quad A_3 = b - \sqrt{b^2 + c_f} \end{aligned} \tag{6-26}$$

Case 6

Maximum negative jerk J_c is applied from point 0 to point 1, where $A_1 = -A_c$. Points 1 and 2 are coincident. Maximum negative acceleration $-A_c$ is applied from point 2 to point 3 and at point 3 maximum positive jerk J_c is applied until the speed reaches V_f just as acceleration vanishes. The governing equation set is as follows:

$$\begin{aligned}
&dt_{01} = \frac{A_1 - A_0}{-J_c},\ V_1 = V_0 + dt_{01}\left(\frac{A_1 + A_0}{2}\right),\ dx_{01} = dt_{01}\left[V_0 + dt_{01}\left(\frac{2A_0 + A_1}{6}\right)\right] \\
&dt_{12} = 0,\quad A_1 = A_2 = A_3,\quad V_2 = V_1 \\
&dt_{34} = \frac{-A_3}{J_c},\ V_3 = V_f - dt_{34}\frac{A_3}{2},\ dx_{34} = dt_{34}\left(V_3 + dt_{34}\frac{A_3}{3}\right) \\
&dt_{23} = \frac{V_2 - V_3}{-A_2},\ dx_{23} = dt_{23}\left(V_2 + dt_{23}\frac{A_2}{2}\right)
\end{aligned} \tag{6-27}$$

7. Headway Needed to Delay Speed Reduction

Consider a vehicle 0 that is commanded to reduce speed from a line speed V_{L_1} to a speed V_{L_2} at time $t = 0$. The slow-down transition takes an amount of time T_m and occurs over a distance D_m. Assume vehicle 1 is a distance dP_1 behind vehicle 0 and traveling at speed V_{L_1} is close enough that it must be commanded to slow down to speed V_{L_2} as close to immediately as possible. Taking into account a computational interval Δt vehicle 1 may not start slowing down until a time Δt later. Thus, once it has reached speed V_{L_2} its distance-time curve is given by the equation

$$x_1 = -dP_1 + V_{L_1}\Delta t + D_m + V_{L_2}\left(t - \Delta t - T_m\right) \tag{7.1}$$

We need to know how far behind vehicle 0 vehicle 2 must be so that it can delay slowing down until it reaches the speed-change command point, i.e., the point along the guideway at which vehicle 0 started to slow down. Assume this is the case. Then vehicle 2 doesn't reach speed V_{L_2} until it reaches the position ahead of the position vehicle 0 began to slow down by an amount $V_{L_1}\Delta t + D_m$. Once vehicle 2 has reached speed V_{L_2} its distance-time curve is given by the equation

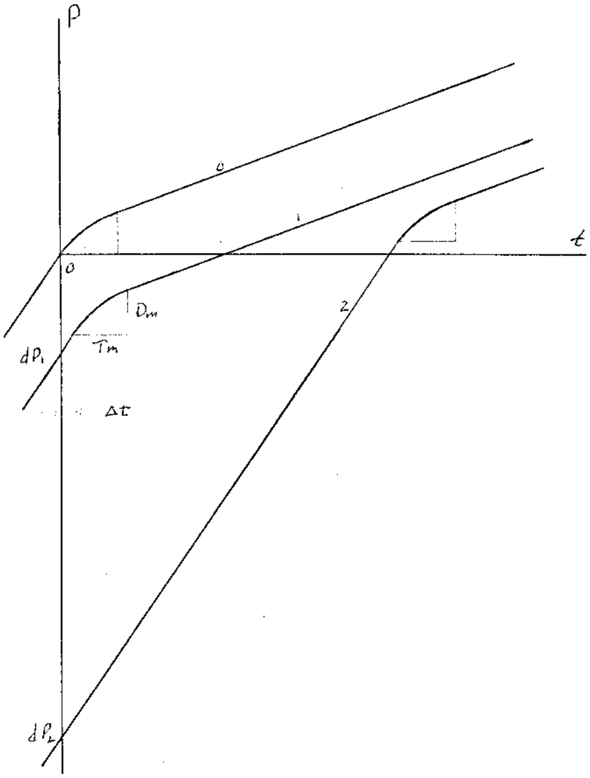


Figure 6. The Kinematics of a Speed Reduction.

$$x_2 = V_{L_1}\Delta t + D_m + V_{L_2}\left(t - \frac{dP_2}{V_{L_1}} - \Delta t - T_m\right) \tag{7.2}$$

Substituting the time $t = \frac{dP_2}{V_{L_1}} + \Delta t + T_m$ into equation (1.1) we see that the separation between vehicle 1 and vehicle 2 at this time is

$$\begin{aligned} x_1 - x_2 &= -dP_1 + V_{L_1}\Delta t + D_m + V_{L_2}\left(\frac{dP_2}{V_{L_1}}\right) - V_{L_1}\Delta t - D_m \\ &= -dP_1 + \frac{V_{L_2}}{V_{L_1}} dP_2 \geq V_{L_2}\left(T_h + \Delta t\right) \end{aligned} \tag{7.3}$$

Thus, the desired result is

$$\boxed{dP_2 \geq V_{L_1}\left(T_h + \Delta t\right) + \frac{V_{L_1}}{V_{L_2}} dP_1} \tag{7.4}$$

8. Emergency Stop

An emergency stop starts with arb**i**trary initial acceleration A_0 and initial speed V_0. The vehicle is subjected to a maximum negative jerk $J_{\max}$ up to a point 1 at which the deceleration is the emergency value $-A_e$ and then decelerates at that maximum rate until a point 2, where positive jerk $J_{\max}$ is applied until the vehicle stops at a point 3. Using the basic transition equations (1-2), the emergency-stop transition is set up with the following equations:

$$dt_{01} = (A_e + A_0)/J_{\max}, \quad V_1 = V_0 - dt_{01}(A_0 - A_e)/2, \quad dx_{01} = dt_{01}\left[V_0 + dt_{01}\left(2A_0 - A_e\right)/6\right]$$
$$dt_{23} = A_e / J_{\max}, \quad V_2 = dt_{23}A_e/2, \quad dx_{23} = dt_{23}\left(V_2 - dt_{23}A_e/3\right)$$
$$dt_{12} = \left(V_1 - V_2\right)/A_e, \quad dx_{12} = dt_{12}\left(V_1 - dt_{12}A_e/2\right)$$
$$t_1 = dt_{01}, \quad t_2 = t_1 + dt_{12}, \quad t_3 = t_2 + dt_{23}$$
$$x_1 = dx_{01}, \quad x_2 = x_1 + dx_{12}, \quad x_3 = x_2 + dx_{23}$$

The transition is then run using the following code:

```
$\Delta t = t - t_0$
if $t < t_1$ then
    Jerk $= -J_{\max}$
    State ($\Delta$t, Jerk, $A_0$, $V_0$, $x_0$, A, V, x)
elseif $t < t_2$ then
    $Jerk = 0$
    State ($\Delta$t, Jerk, $A_1$, $V_1$, $x_1$, A, V, x)
elseif $t < t_3$ then
    $Jerk = +J_{\max}$
    State ($\Delta$t, Jerk, $A_2$, $V_2$, $x_2$, A, V, x)
else
    $Jerk = 0$
end if
```

9. Distance to Reach Station Speed

When a vehicle is ready to leave a station it may be advancing in the station at any speed V_0 below station speed V_s and at any acceleration A_0 within the comfort range. It must not be permitted to accelerate to line speed if at that moment it would exceed V_s its nose would not have reached the downstream end of the station. The criterion to leave is thus that the distance to reach V_s is greater than the distance to the downstream end of the station. Using the notation of Section 3, let the distance from initiation of the acceleration transition to the speed V_s be dx_{0s}. Then, following the notation of Section 3, we must consider two cases:

$V_s \le V_2$

$$dx_{0s} = dx_{01} + \left(V_s - V_1\right) / A_c.$$

$V_s > V_2$

$$dx_{0s} = dx_{01} + dx_{12} + dt_{2s}\left[V_2 + dt_{2s}\left(2A_c + A_s\right)/6\right]$$

in which

$$A_s = A_c - J_n dt_{2s}$$

$$V_s = V_2 + A_c dt_{2s} - \frac{J_n}{2} dt_{2s}^2 \quad \text{or} \quad \frac{J_n}{2} dt_{2s}^2 - A_c dt_{2s} + V_s - V_2 = 0$$

$$\text{or} \quad dt_{2s} = \frac{1}{J_n}\left[A_c - \sqrt{A_c^2 - 2J_n\left(V_s - V_2\right)}\right]$$

$$dx_{0s} = dx_{01} + dx_{12} + dt_{2s}\left[V_2 + dt_{2s}\left(3A_c - J_n dt_{2s}\right)/6\right]$$

in which the minus sign before the square root is taken because we know that when $V_s \to V_2, \quad dt_{2s} \to 0.$

10. The Distance to Slip a Given Amount

Assume a series of slip transitions which begin and end at line speed. Assume that the minimum speed in the transition, V_m is low enough so that the cruising time at V_m is zero. Then if V_L is the line speed, if $V_m < A_c^2 / J_c$ then $A_1 = \sqrt{J_c V_L}$ else $A_1 = A_c$. The transition time is

$$T_m = 2\left(\frac{V_L - V_m}{A_1} + \frac{A_1}{J_c}\right).$$

The slip S is

$$S = \left(V_L - V_m\right) T_m / 2$$

and the distance traveled to slip S is

$$\text{Distance} = V_L T_m - S.$$

These quantities are calculated and plotted in the following table for $V_L = 16$ m/s.

Table 1

Distance Traveled During Slip Transition

g =	9.807	m/s^2
Ac =	2.452	m/s^2
Jc =	2.452	m/s^3
VL =	16	m/s

Vmin	A1	Tm	Slip	Dist
m/s	m/s^2	sec	m	m
16.0	0.00	0.00	0.00	0.00
15.8	0.70	1.14	0.11	18.17
15.6	0.99	1.62	0.32	25.53
15.4	1.21	1.98	0.59	31.07
15.2	1.40	2.28	0.91	35.65
15.0	1.57	2.55	1.28	39.60
14.8	1.72	2.80	1.68	43.10
14.6	1.85	3.02	2.12	425
14.4	1.98	3.23	2.59	49.12
14.2	2.10	3.43	3.08	51.75
14.0	2.21	3.61	3.61	54.19
13.8	2.32	3.79	4.17	546
13.6	2.43	3.96	4.75	58.57
13.4	2.45	4.12	5.36	60.58
13.2	2.45	4.28	6.00	62.55
13.0	2.45	4.45	67	64.49
12.8	2.45	4.61	7.38	639
12.6	2.45	4.77	8.12	68.26
12.4	2.45	4.94	8.89	70.10
12.2	2.45	5.10	9.69	71.91

12.0	2.45	5.26	10.53	73.68
11.8	2.45	5.43	11.40	75.42
11.6	2.45	5.59	12.30	77.13
11.4	2.45	5.75	13.23	78.81
11.2	2.45	5.92	14.20	80.45
11.0	2.45	6.08	15.20	82.06
10.8	2.45	24	123	83.64
10.6	2.45	41	17.29	85.19
10.4	2.45	57	18.39	86.70
10.2	2.45	6.73	19.52	88.18
10.0	2.45	6.89	20.68	89.63

Potential Headway Violation upon Decelerating into a Station

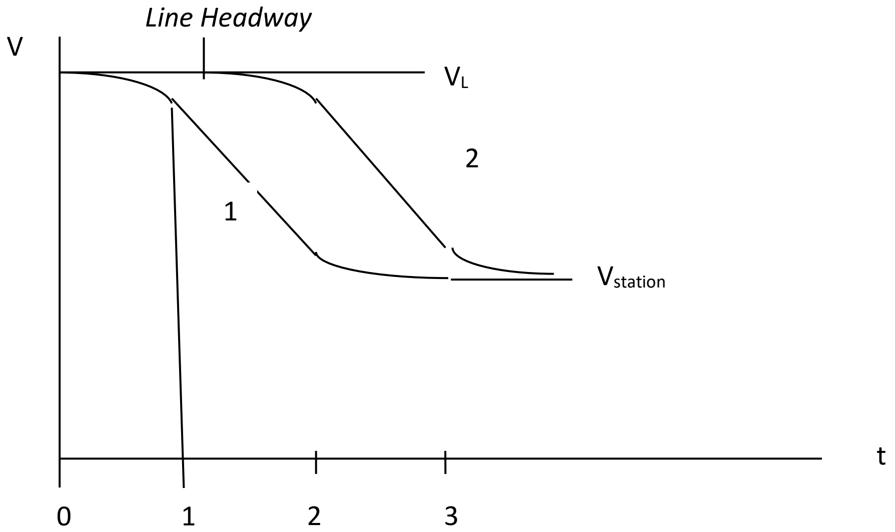


Figure 1. The velocity profiles of a pair of vehicles entering a station.

Consider a vehicle #1 decelerating into a station to station speed V_{sta}, followed by a vehicle #2 a time *Line Headway* behind undergoing the same maneuver. Let the position of vehicle #1 at time zero be $x(0) = 0$. The times, accelerations, speeds, and positions of vehicle #1 at the points 1, 2, 3 in Figure 1[10] are as follows:

$$
\begin{aligned}
&dt_{01} = \frac{A_c}{J_c}, \quad V_1 = V_L - dt_{01}\frac{A_c}{2}, \quad dx_{01} = dt_{01}\left(V_L - dt_{01}\frac{A_c}{6}\right) \\
&dt_{23} = \frac{A_c}{J_c}, \quad V_2 = V_{sta} + dt_{23}\frac{A_c}{2}, \quad dx_{23} = dt_{23}\left(V_2 - dt_{23}\frac{A_c}{3}\right) \\
&dt_{12} = \frac{V_1 - V_2}{A_c}, \qquad dx_{12} = dt_{12}\left(V_1 - dt_{12}\frac{A_c}{2}\right) \\
&t_1 = dt_{01}, \quad t_2 = t_1 + dt_{12}, \quad t_3 = t_2 + dt_{23} \\
&x_1 = dx_{01}, \quad x_2 = x_1 + dx_{12}, \quad x_3 = x_2 + dx_{23}
\end{aligned}
\tag{1}
$$

From equations (1) we find

$$
dt_{03} = dt_{01} + dt_{23} + dt_{12} = 2\frac{A_c}{J_c} + \frac{1}{A_c}\left(V_L - \frac{A_c^2}{2J_c} - V_{sta} - \frac{A_c^2}{2J_c}\right) = \frac{V_L - V_{sta}}{A_c} + \frac{A_c}{J_c}
\tag{2}
$$

[10] For the methodology, see the internal paper "Speed and Position vs. Time"

Thus, the maneuver time from line speed to station speed is

$$T_m = \frac{V_L - V_{sta}}{A_c} + \frac{A_c}{J_c} \tag{3}$$

From equations (1) we also find

$$dx_{03} = dx_{01} + dx_{23} + dx_{12} = \frac{A_c}{J_c}\left(V_L + V_{sta} + \frac{A_c^2}{2J_c} - \frac{A_c^2}{6J_c} - \frac{A_c^2}{3J_c}\right) + \left(\frac{V_1 - V_2}{2A_c}\right)\left(V_1 + V_2\right)$$

$$= \frac{A_c}{J_c}\left(V_L + V_{sta}\right) + \frac{1}{2A_c}\left(V_1 - V_2\right)\left(V_1 + V_2\right) = \frac{A_c}{J_c}\left(V_L + V_{sta}\right) + \frac{1}{2A_c}\left(V_L - V_{sta} - \frac{A_c^2}{J_c}\right)\left(V_L + V_{sta}\right) \tag{4}$$

$$= \frac{\left(V_L + V_{sta}\right)}{2}\left(\frac{V_L - V_{sta}}{A_c} + \frac{A_c}{J_c}\right) = \frac{\left(V_L + V_{sta}\right)}{2} dt_{03}$$

Thus, the distance traveled from line speed to station speed is

$$D_m = \frac{\left(V_L + V_{sta}\right)}{2} T_m \tag{5}$$

Using the above canonical formulation, the acceleration, speed, and position of vehicle 1 at any value of t are as follows:

$$0 \le t \le t_1 : \quad \Delta t = t, \quad A = -J_c \Delta t, \quad V = V_L + \Delta t \frac{A}{2}, \quad x = \Delta t\left(V_L + \Delta t \frac{A}{6}\right)$$

$$t_1 \le t \le t_2 : \quad \Delta t = t - t_1, \quad A = -A_c, \quad V = V_1 + \Delta t A, \quad x = x_1 + \Delta t\left(V_1 + \Delta t \frac{A}{2}\right)$$

$$t_2 \le t \le t_3 : \quad \Delta t = t - t_2, \quad A = -A_c + J_c \Delta t, \quad V = V_2 + \Delta t \frac{\left(-A_c + A\right)}{2}, \quad x = x_2 + \Delta t\left(V_2 + \Delta t \frac{A}{3}\right) \tag{6}$$

For vehicle #2 up to time $t = LineHeadway$ the speed stays constant at V_L and the distance traveled is

$x = V_L t$. For $t > LineHeadway$ we can obtain the acceleration, speed, and position as functions of time by making the following substitutions in equations (5): $t \rightarrow t - LineHeadway$

$$
\begin{aligned}
&T_h = LineHeadway\\
&0 \le t \le T_h: \quad A=0, \quad V=V_L, \quad x=V_L\left(t-T_h\right)\\
&T_h \le t \le t_1+T_h: \quad \Delta t = t-T_h, \quad A=-J_c\Delta t, \quad V=V_L+\Delta t\frac{A}{2}, \quad x=\Delta t\left(V_L+\Delta t\frac{A}{6}\right)\\
&T_h+t_1 \le t \le T_h+t_2: \quad \Delta t=t-t_1, \quad A=-A_c, \quad V=V_1+\Delta tA, \quad x=x_1+\Delta t\left(V_1+\Delta t\frac{A}{2}\right)\\
&T_h+t_2 \le t \le T_h+t_3: \quad \Delta t=t-t_2, \quad A=-A_c+J_c\Delta t, \quad V=V_2+\Delta t\frac{\left(-A_c+A\right)}{2}, \quad x=x_2+\Delta t\left(V_2+\Delta t\frac{A}{3}\right)
\end{aligned}
\tag{7}
$$

The Minimum Headway

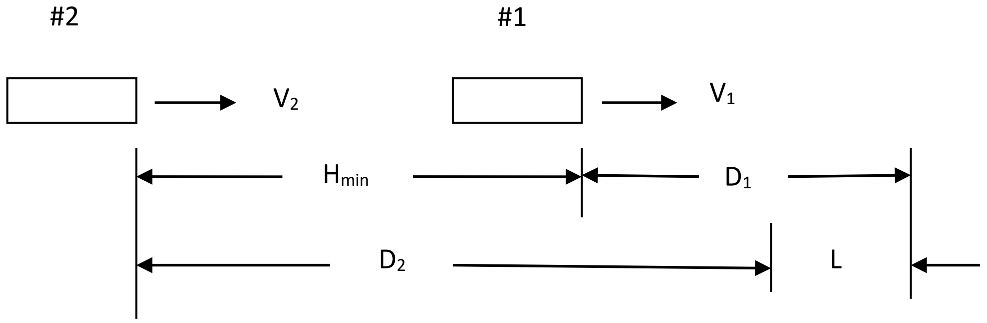


Figure 2. A pair of vehicles moving to the right.

Assume vehicle #1 stops due to a failure at deceleration A_f and jerk J_f. From equation (5), the stopping distance of vehicle #1 is

$$D_1 = \frac{V_1}{2}\left(\frac{V_1}{A_f}+\frac{A_f}{J_f}\right) \tag{8}$$

After a control time delay t_c, vehicle #2 stops at the emergency deceleration rate A_e and emergency jerk J_e. Its stopping distance is therefore

$$D_2 = V_2 t_c + \frac{V_2}{2}\left(\frac{V_2}{A_e}+\frac{A_e}{J_e}\right) \tag{9}$$

Assuming the length of each of the two vehicles is L, the minimum allowable separation between them is

$$H_{\min} = L + D_2 - D_1 \tag{10}$$

The minimum permissible time headway is therefore

$$MinHeadway = \frac{H_{\min}}{V_2} \tag{11}$$

A program to calculate the acceleration, speed, positions profiles and the minimum headway is given in the Appendix. Some results are given in Figures 3 and 4.

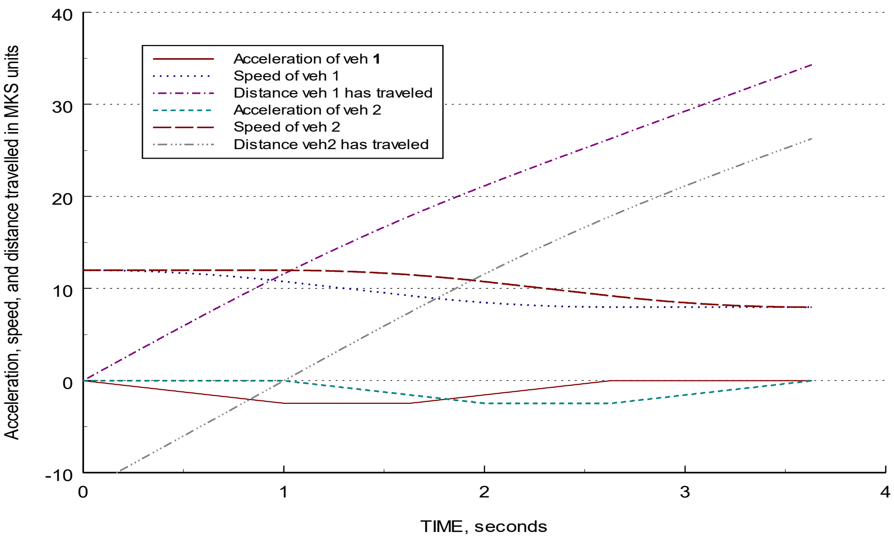


Figure 3. Kinematics of motion of a pair of vehicles decelerating to station speed.

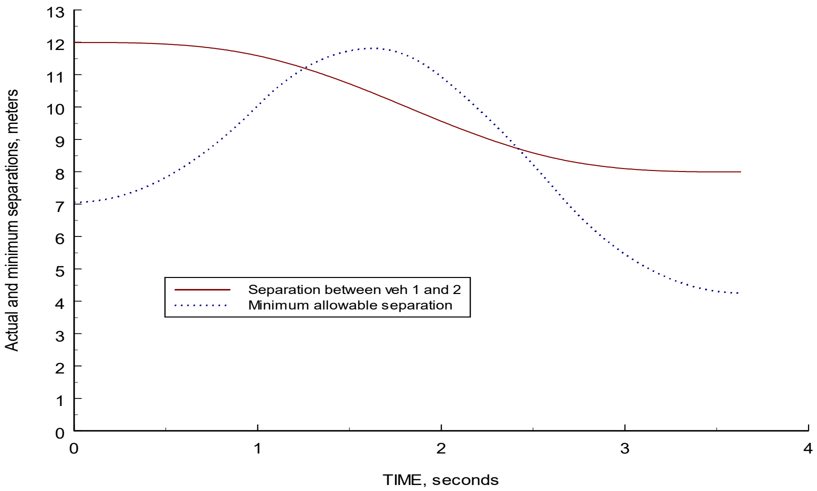


Figure 4. Separation and minimum allowable separation between two vehicles entering a station.

The parameters used in Figures 3 and 4 are those given at the beginning of the program shown in the Appendix. Many runs can be made for different accelerations and jerks. For the set shown in the program, runs were made with different line headways and control time constants to obtain the maximum negative separations as shown in Table 1 and as calculated by the program.

Table 1. Maximum headway violations for the cases shown.

$t_c \backslash LineHeadway \rightarrow$	0.5	1.0	1.3	1.5
0.05	-3.25	-1.03	0	0
0.10	-3.80	-1.59	-0.03	0
0.15	-4.36	-2.15	-0.60	0
0.20	-4.92	-2.71	-1.17	-0.01

It is seen that if the line headway between two vehicles sequentially entering a station is to be as low as one second, the control time constant must be quite small, but not particularly small using contemporary technology. Note from Figure 4 that in the case shown the small headway violation increases from zero back to zero in about one second.

In this work, we considered only the portion of the maneuver from line speed to station speed. Further development of the program included in the Appendix shows that, since the second of the pair of vehicles will be stopping at least one berth behind the first, there is no headway violation in the maneuvers from station speed to rest.

Appendix

```
'This program MINHEAD.BAS calculates the minimum headway permissible
'between a pair of vehicles decelerating into a station
'Units are MKS

DEFDBL A-Z
DIM Counter AS INTEGER
DIM A(1 TO 2) AS DOUBLE    'acceleration of vehicles 1 & 2
DIM V(1 TO 2) AS DOUBLE    'speed of vehicles 1 & 2
DIM X(1 TO 2) AS DOUBLE    'position of vehicles 1 & 2
DIM t4(1 TO 2) AS DOUBLE   'time at end of station-speed section
DIM t5(1 TO 2) AS DOUBLE   'time at command to constant deceleration
DIM t6(1 TO 2) AS DOUBLE   'time at command to constant jerk
DIM t7(1 TO 2) AS DOUBLE   'time at maneuver end, total maneuver time

DIM X1(1 TO 2) AS DOUBLE   'position of command to constant deceleration
DIM X2(1 TO 2) AS DOUBLE   'position of command to constant jerk
DIM X3(1 TO 2) AS DOUBLE   'position at beginning of station-speed section
DIM X4(1 TO 2) AS DOUBLE   'position at end of station-speed section
DIM X5(1 TO 2) AS DOUBLE   'position of command to constant deceleration
DIM X6(1 TO 2) AS DOUBLE   'position of command to constant jerk
DIM X7(1 TO 2) AS DOUBLE   'position at maneuver end,total maneuver distance

DIM D(1 TO 2) AS DOUBLE    'stopping distances of vehicles 1 & 2
```

```
g = 9.80665     'acceleration of gravity
Ac = .25 * g    'comfort deceleration
Jc = .25 * g    'comfort jerk
tJ = Ac / Jc    'jerk time constant
Af = .4 * g     'maximum failure deceleration
Jf = .4 * g     'maximum failure jerk
Ae = .4 * g     'emergency deceleration
Je = .8 * g     'emergency jerk
VL = 12         'line speed
Vsta = 8        'station speed
tc = .15         'time constant
Lveh = 2.743    'vehicle length
B = 3.048       'berth length
LineHeadway = .5 'time headway between vehicles while at line speed
t = 0           'start time
dt = .01        'computational time interval

'Calculation of the maneuver increments and transition speeds
dt01 = tJ
V1 = VL - dt01 * Ac / 2
dx01 = dt01 * (VL - Ac * dt01 / 6)
dt23 = tJ
V2 = Vsta + dt23 * Ac / 2
dx23 = dt23 * (V2 - dt23 * Ac / 3)
dt12 = (V1 - V2) / Ac
dx12 = dt12 * (V1 - dt12 * Ac / 2)
dx34 = 10      'distance vehicle 1 travels at station speed
dt34 = dx34 / Vsta          'time of veh 1 at station speed
dt45 = tJ
V5 = Vsta - dt45 * Ac / 2
dx45 = dt45 * (Vsta - dt45 * Ac / 6)
dt67 = tJ
V6 = dt67 * Ac / 2
dx67 = dt67 * (V6 - dt67 * Ac / 3)
dt56 = (V5 - V6) / Ac
dx56 = dt56 * (V5 - dt56 * Ac / 2)

'Times and position increments at the transition points
t1 = dt01
t2 = t1 + dt12
t3 = t2 + dt23
t4(1) = t3 + dt34            'this and following times for veh 1
t5(1) = t4(1) + dt45
t6(1) = t5(1) + dt56
t7(1) = t6(1) + dt67            'maneuver time
t4(2) = t3 + dt34 - B / Vsta 'this and following times for veh 2
t5(2) = t4(2) + dt45
t6(2) = t5(2) + dt56
t7(2) = t6(2) + dt67            'maneuver time


X1(1) = dx01
X2(1) = X1(1) + dx12
X3(1) = X2(1) + dx23
X4(1) = X3(1) + dx34
X5(1) = X4(1) + dx45
X6(1) = X5(1) + dx56
```

```
X7(1) = X6(1) + dx67

X1(2) = dx01
X2(2) = X1(2) + dx12
X3(2) = X2(2) + dx23
X4(2) = X3(2) + dx34 - B        'veh 2 stops one berth short of veh 1
X5(2) = X4(2) + dx45
X6(2) = X5(2) + dx56
X7(1) = X6(1) + dx67            'total maneuver distance

CLS
SCREEN 9
COLOR 7, 8
scaleT = 600 / t7(2)
scaleA = 10
scaleV = 10
scaleX = 4
scaleS = 40
T0 = 10
Y0 = 280
LINE (T0, Y0)-(640, Y0)
LINE (T0, Y0)-(T0, 0)

OPEN "KINEMAT.ASC" FOR OUTPUT AS #1
OPEN "SEPRATN.ASC" FOR OUTPUT AS #2

DO
  'Motion of first vehicle
  IF t <= t1 THEN
     DelT = t
     A(1) = -Jc * DelT
     V(1) = VL + DelT * A(1) / 2
     X(1) = DelT * (VL + DelT * A(1) / 6)
  ELSEIF t <= t2 THEN
     DelT = t - t1
     A(1) = -Ac
     V(1) = V1 + DelT * A(1)
     X(1) = X1(1) + DelT * (V1 + DelT * A(1) / 2)
  ELSEIF t <= t3 THEN
     DelT = t - t2
     A(1) = -Ac + Jc * DelT
     V(1) = V2 + DelT * (-Ac + A(1)) / 2
     X(1) = X2(1) + DelT * (V2 + DelT * (-2 * Ac + A(1)) / 6)
  ELSEIF t <= t4(1) THEN
     DelT = t - t3
     A(1) = 0
     V(1) = Vsta
     X(1) = X3(1) + Vsta * DelT
  ELSEIF t <= t5(1) THEN
     DelT = t - t4(1)
     A(1) = -Jc * DelT
     V(1) = Vsta + DelT * A(1) / 2
     X(1) = X4(1) + DelT * (Vsta + DelT * A(1) / 6)
  ELSEIF t <= t6(1) THEN
     DelT = t - t5(1)
     A(1) = -Ac
     V(1) = V5 + DelT * A(1)
```

```
   X(1) = X5(1) + DelT * (V5 + DelT * A(1) / 2)
ELSEIF t < t7(1) THEN
   DelT = t - t6(1)
   A(1) = -Ac + Jc * DelT
   V(1) = V6 + DelT * (-Ac + A(1)) / 2
   X(1) = X6(1) + DelT * (V6 + DelT * (-2 * Ac + A(1)) / 6)
ELSE
   A(1) = 0
   V(1) = 0
   X(1) = X7(1)
END IF

'Motion of second vehicle
tsec = t - LineHeadway
IF tsec <= 0 THEN
   DelT = tsec
   A(2) = 0
   V(2) = VL
   X(2) = DelT * VL
ELSEIF tsec <= t1 THEN
   DelT = tsec
   A(2) = -Jc * DelT
   V(2) = VL + DelT * A(2) / 2
   X(2) = DelT * (VL + DelT * A(2) / 6)
ELSEIF tsec <= t2 THEN
   DelT = tsec - t1
   A(2) = -Ac
   V(2) = V1 + DelT * A(2)
   X(2) = X1(2) + DelT * (V1 + DelT * A(2) / 2)
ELSEIF tsec <= t3 THEN
   DelT = tsec - t2
   A(2) = -Ac + DelT * Jc
   V(2) = V2 + DelT * (-Ac + A(2)) / 2
   X(2) = X2(2) + DelT * (V2 + DelT * (-2 * Ac + A(2)) / 6)
ELSEIF tsec <= t4(2) THEN
   DelT = tsec - t3
   A(2) = 0
   V(2) = Vsta
   X(2) = X3(2) + Vsta * DelT
ELSEIF tsec <= t5(2) THEN
   DelT = tsec - t4(2)
   A(2) = -Jc * DelT
   V(2) = Vsta + DelT * A(2) / 2
   X(2) = X4(2) + DelT * (Vsta + DelT * A(2) / 6)
ELSEIF tsec <= t6(2) THEN
   DelT = tsec - t5(2)
   A(2) = -Ac
   V(2) = V5 + DelT * A(2)
   X(2) = X5(2) + DelT * (V5 + DelT * A(2) / 2)
ELSEIF tsec < t7(2) THEN
   DelT = tsec - t6(2)
   A(2) = -Ac + Jc * DelT
   V(2) = V6 + DelT * (-Ac + A(1)) / 2
   X(2) = X6(2) + DelT * (V6 + DelT * (-2 * Ac + A(1)) / 6)
ELSE
   A(2) = 0
   V(2) = 0
```

```
     X(2) = X7(2)
  END IF

  D(1) = .5 * V(1) * (V(1) / Af + Af / Jf)  'stopping distance of veh #1
  D(2) = .5 * V(2) * (V(2) / Ae + Ae / Je)  'stopping distance of veh #2

  Separation = X(1) - X(2)
  IF Separation < Lveh + V(2) * tc THEN SLEEP
  IF V(2) > 0 THEN Headway = Separation / V(2)
  MinSeparation = Lveh + V(2) * tc + D(2) - D(1)
  IF V(2) > 0 THEN MinHeadway = MinSeparation / V(2)
  dSep = Separation - MinSeparation
  IF dSep < MaxNegSep THEN MaxNegSep = dSep

  PSET (T0 + scaleT * t, Y0 - scaleA * A(1)), 14
  PSET (T0 + scaleT * t, Y0 - scaleV * V(1)), 13
  PSET (T0 + scaleT * t, Y0 - scaleX * X(1)), 12
  PSET (T0 + scaleT * t, Y0 - scaleA * A(2)), 11
  PSET (T0 + scaleT * t, Y0 - scaleV * V(2)), 10
  PSET (T0 + scaleT * t, Y0 - scaleX * X(2)), 9

  PSET (T0 + scaleT * t, Y0 - scaleS * Separation), 5
  PSET (T0 + scaleT * t, Y0 - scaleS * MinSeparation), 6

  'PRINT USING "#####.##"; t; A(1); V(1); X(1) ; A(2); V(2); X(2);
Separation; MinSeparation
  'PRINT USING "#####.##"; t; V(2); Separation; Separation - Lveh - V(2) *
tc; MinSeparation; dSep; Headway; MinHeadway
  IF Counter = 20 THEN
     Counter = 0
     'SLEEP
  END IF
  Counter = Counter + 1
  'WRITE #1, t, A(1), V(1), X(1), A(2), V(2), X(2)
  'WRITE #2, t, Separation, MinSeparation
  t = t + dt
LOOP UNTIL t > t7(2) + 1
PRINT "    MaxNegSep = ";
PRINT USING "###.##"; MaxNegSep
CLOSE #1
CLOSE #2
```

Headway Needed to Delay Speed Reduction

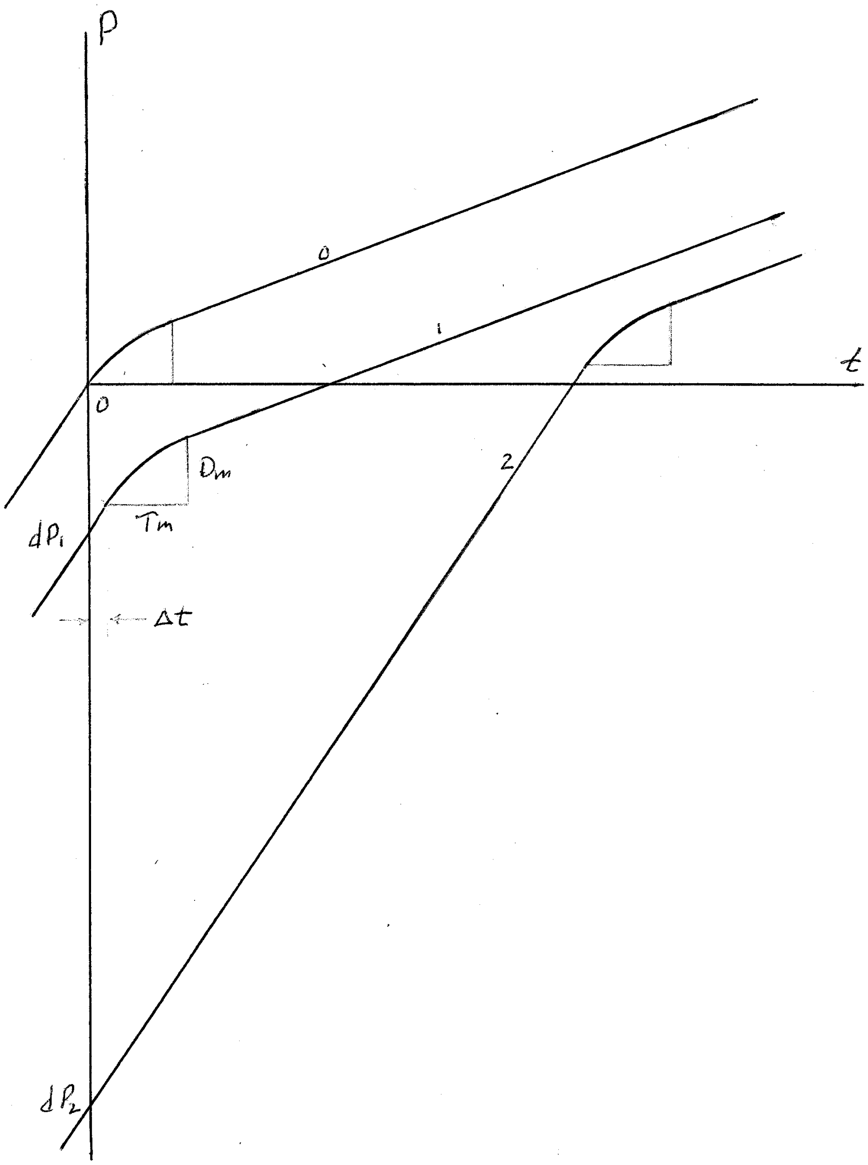


Figure 1. The Kinematics of a Speed Reduction.

Consider a vehicle 0 that is commanded to reduce speed from a line speed V_{L_1} to a speed V_{L_2} at time $t = 0$. The slow-down maneuver takes an amount of time T_m and occurs over a distance D_m. Assume vehicle 1 is a distance dP_1 behind vehicle 0 and traveling at speed V_{L_1}. Assume that it is close enough to vehicle 0 that it must be commanded to slow down to speed V_{L_2} as close to immediately as possible. Taking into account a computational interval Δt vehicle 1 may not start slowing down until a time Δt later. Thus, once it has reached speed V_{L_2} its distance-time curve is given by the equation

$$x_1 = -dP_1 + V_{L_1}\Delta t + D_m + V_{L_2}\left(t - \Delta t - T_m\right) \qquad (1)$$

We need to know how far behind vehicle 0 vehicle 2 must be so that it can delay slowing down until it reaches the speed-change command point, i.e., the point along the guideway at which vehicle 0 started to slow down. Assume this is the case. Then vehicle 2 doesn't reach speed V_{L_2} until it reaches the position ahead of the position vehicle 0 began to slow down by an amount $V_{L_1}\Delta t + D_m$. Once vehicle 2 has reached speed V_{L_2} its distance-time curve is given by the equation

$$x_2 = V_{L_1}\Delta t + D_m + V_{L_2}\left(t - \frac{dP_2}{V_{L_1}} - \Delta t - T_m\right) \qquad (2)$$

Substituting the time $t = \frac{dP_2}{V_{L_1}} + \Delta t + T_m$ into equation (1) we see that the separation between vehicle 1 and vehicle 2 at this time is

$$\begin{aligned} x_1 - x_2 &= -dP_1 + V_{L_1}\Delta t + D_m + V_{L_2}\left(\frac{dP_2}{V_{L_1}}\right) - V_{L_1}\Delta t - D_m \\ &= -dP_1 + \frac{V_{L_2}}{V_{L_1}} dP_2 \geq V_{L_2}\left(T_h + \Delta t\right) \end{aligned} \qquad (3)$$

in which T_h is the minimum permissible time headway. Thus, the desired result is

$$\boxed{dP_2 \geq V_{L_1}\left(T_h + \Delta t\right) + \frac{V_{L_1}}{V_{L_2}} dP_1}$$

On-Line Deceleration

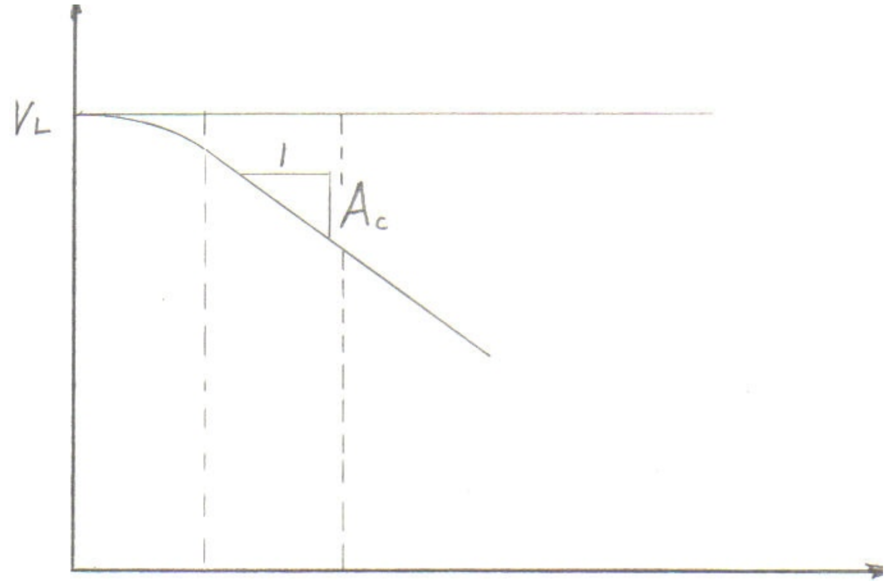


Figure 1. Speed profile of vehicle decelerating.

1. Introduction

To reduce the required length of off-line guideway in a PRT system, it is possible to initiate deceleration before a vehicle is clear of mainline traffic. The question that this memo answers is this: What is the relationship between the distance traveled by a decelerating vehicle while still on-line and the reduction in on-line headway? This memo shows that by sacrificing a small amount of on-line headway, the length of the by-pass guideway can be reduced substantially.

2. Deceleration at constant negative jerk.

Figure 1 is a plot of speed V vs. time t and illustrates the speed profile of a vehicle decelerating from a line speed V_L into a station. At first negative jerk J_c is applied until at a point 1, the deceleration reaches the comfort value A_c. The vehicle then decelerates at the comfort value until it either stops or assumes the station speed. For the time interval $0 \le t \le t_1$ the equations of motion are

$$\ddot{x} = -J_c t, \quad \dot{x} = V_L - J_c \frac{t^2}{2}, \quad x = V_L t - J_c \frac{t^3}{6} \qquad (1)$$

At time t_1

$$\ddot{x} = -A_c = -J_c t_1 \quad \therefore \; t_1 = \frac{A_c}{J_c} \qquad (2)$$

Then

$$V_1 = V_L - \frac{1}{2} J_c \left(\frac{A_c}{J_c} \right)^2 = V_L - \frac{A_c^2}{2J_c}$$
$$x_{01} = \frac{A_c}{J_c} \left(V_L - \frac{A_c^2}{6J_c} \right) \tag{3}$$

The distance the vehicle moves backwards relative to or closes up to a vehicle behind it traveling at constant speed V_L is called the slip distance, which if the vehicle slows down a time t_1 is

$$S_{01} = V_L t_1 - x_{01} = \frac{A_c^3}{6J_c^2} \tag{4}$$

3. <u>Headway sacrificed during constant-jerk motion</u>

Headway t_h is defined as the time interval between the passage of the nose of one vehicle and the passage of the next relative to a stationary point. The distance traveled by a vehicle moving at speed V_L during this time interval is $V_L t_h$. Thus, the headway lost to point 1 if a vehicle begins to decelerate while still on the main line is

$$\Delta t_h = \frac{S_{01}}{V_L} = \frac{A_c^3}{6J_c^2 V_L}. \tag{5}$$

For times less than t_1 substitute $J_c t$ for A_c from equations (1). Then we have

$$\frac{J_c}{6} t^3 = V_L \Delta t_h, \quad t = \left(\frac{6V_L \Delta t_h}{J_c} \right)^{1/3} \tag{6}$$

Substituting t into the third of equations (1), we see that for $t < t_1$ the distance traveled while losing a headway of Δt_h is

$$x = V_L \left[\left(\frac{6V_L \Delta t_h}{J_c} \right)^{1/3} - \Delta t_h \right] \tag{7}$$

For longitudinal motion the comfort values are $A_c = 0.25g, \quad J_c = 0.25g/s$. Therefore, assuming $g = 9.80665$ m/s^2 we have from equation (5) $\Delta t_h = 0.4086/V_L$. Assuming a minimum speed of say 10 m/s, we find that up to point 1 $\Delta t_h = 0.041$ sec.

4. Headway sacrificed during constant-deceleration phase

Assuming that we can permit a greater loss of headway as a result of on-line deceleration, let the vehicle proceed to a point 2 at constant deceleration $-A_c$ where $t = t_2$, and $\Delta t_{12} = t_2 - t_1$. The constant deceleration region pertains until $V_2 \le A_c^2 / 2J_c$. The speed and distance traveled during this interval in which $V_1 > V_2 > A_c^2 / 2J_c$ are

$$V_2 = V_1 - A_c \Delta t_{12}$$

$$x_{12} = V_1 \Delta t_{12} - A_c \frac{\Delta t_{12}^2}{2} = \Delta t_{12}\left(V_L - \frac{A_c^2}{2J_c} - \frac{A_c}{2}\Delta t_{12} \right) \tag{8}$$

The slip distance in traveling from point 1 to point 2 is

$$S_{12} = V_L \Delta t_{12} - x_{12} = \frac{A_c}{2}\left(\frac{A_c}{J_c} + \Delta t_{12} \right)\Delta t_{12} \tag{9}$$

Thus the total slip distance up to point 2 is

$$S_{02} = S_{01} + S_{12} = \frac{A_c}{2}\left[\frac{A_c^2}{3J_c^2} + \left(\frac{A_c}{J_c} + \Delta t_{12} \right)\Delta t_{12} \right] = V_L \Delta t_{h_{02}} \tag{10}$$

where $\Delta t_{h_{02}}$ is the headway lost in slowing down to point 2. Equation (10) can be rearranged into the form

$$\Delta t_{12}^2 + 2\left(\frac{A_c}{2J_c} \right)\Delta t_{12} - \left(\frac{2V_L \Delta t_{h_{02}}}{A_c} - \frac{A_c^2}{3J_c^2} \right) = 0 \tag{11}$$

the positive root of which is

$$\Delta t_{12} = -\frac{A_c}{2J_c} + \sqrt{\left(\frac{A_c}{2J_c} \right)^2 + \frac{2V_L \Delta t_{h_{02}}}{A_c} - \frac{A_c^2}{3J_c^2}} = \frac{A_c}{2J_c}\left[\sqrt{\frac{8V_L \Delta t_{h_{02}}}{A_c}\left(\frac{J_c}{A_c} \right)^2 - \frac{1}{3}} - 1 \right] \tag{12}$$

5. Total headway sacrificed

The distance traveled in the time period Δt_{02} is

$$x_{02} = x_{01} + x_{12} = \frac{A_c}{J_c}\left(V_L - \frac{A_c^2}{6J_c} \right) + \Delta t_{12}\left(V_L - \frac{A_c^2}{2J_c} - \frac{A_c}{2}\Delta t_{12} \right) \tag{13}$$

Substitute for Δt_{12}^2 from equation (11). Then equation (13) becomes

$$
\begin{aligned}
x_{02} &= \frac{A_c}{J_c}\left(V_L - \frac{A_c^2}{6J_c}\right) + \Delta t_{12}\left(V_L - \frac{A_c^2}{2J_c}\right) - \frac{A_c}{2}\Delta t_{12}^2 \\
&= \frac{A_c}{J_c}\left(V_L - \frac{A_c^2}{6J_c}\right) + \Delta t_{12}\left(V_L - \frac{A_c^2}{2J_c}\right) + \frac{A_c}{2}\left(\frac{A_c}{J_c}\Delta t_{12} - \frac{2V_L}{A_c}\Delta t_{h_{02}} + \frac{A_c^2}{3J_c^2}\right) \qquad (14) \\
&= V_L\left(\frac{A_c}{J_c} + \Delta t_{12} - \Delta t_{h_{02}}\right)
\end{aligned}
$$

Substituting for Δt_{12} from equation (12) we get

$$
x_{02} = V_L\left\{\frac{A_c}{2J_c}\left[1 + \sqrt{\frac{8V_L J_c^2}{A_c^3}\Delta t_{h_{02}} - \frac{1}{3}}\right] - \Delta t_{h_{02}}\right\} \quad \text{if} \quad \Delta t_{h_{02}} \geq \frac{A_c^3}{6J_c^2 V_L} \qquad (15)
$$

If $\Delta t_{h_{02}} < \frac{A_c^3}{6J_c^2 V_L}$ the distance traveled while losing a headway of $\Delta t_h = \Delta t_{h_{02}}$ is given by equation (7). The reader can verify that at $\Delta t_{h_{02}} = \frac{A_c^3}{6J_c^2 V_L}$ both equations (7) and (15) give the same result.

6. Range of Validity of equation (15)

As mentioned in Section 4, equation (15) is valid if $V_2 > A_c^2 / 2J_c$. From equations (3) and (8) this condition becomes

$$
V_2 = V_L - \frac{A_c^2}{2J_c} - A_c \Delta t_{12} > \frac{A_c^2}{2J_c} \quad \text{or} \quad \Delta \mathrm{t}_{12} < \frac{V_L}{A_c} - \frac{A_c}{J_c} \qquad (16)
$$

Substituting for Δt_{12} from equation (12) and reducing, we get

$$
\begin{aligned}
&\frac{A_c}{2J_c}\left[\sqrt{\frac{8V_L \Delta t_{h_{02}}}{A_c}\left(\frac{J_c}{A_c}\right)^2 - \frac{1}{3}}\right] < \frac{V_L}{A_c} - \frac{A_c}{2J_c} \\
&\sqrt{\frac{8V_L \Delta t_{h_{02}}}{A_c}\left(\frac{J_c}{A_c}\right)^2 - \frac{1}{3}} < \frac{2V_L}{A_c}\frac{J_c}{A_c} - 1 \qquad (17) \\
&\frac{8V_L \Delta t_{h_{02}}}{A_c}\left(\frac{J_c}{A_c}\right)^2 - \frac{1}{3} < \frac{4V_L^2}{A_c^2}\frac{J_c^2}{A_c^2} - \frac{4V_L}{A_c}\frac{J_c}{A_c} + 1 \\
&\Delta t_{h_{02}} < \frac{1}{2}\left(\frac{V_L}{A_c} - \frac{A_c}{J_c} + \frac{A_c^2}{3J_c^2}\frac{A_c}{V_L}\right)
\end{aligned}
$$

Using the above values we find for $V_L = 10$ $\Delta t_{h_{02}} = 1.6\,\text{sec}$, and for $V_L = 20$ meters per second $\Delta t_{h_{02}} = 3.6\,\text{sec}$. These values are much longer than would be of interest for this problem, therefore equation (12) gives correct values for Δt_{12}.

7. The maximum on-line distance traveled.

We can now plot a curve of distance traveled while losing a headway of Δt_h. Note, from equation (7) that at $\Delta t_h = 0$ the rate of change of x with Δt_h is infinite. The form of equation (15) shows that as a function of Δt_h x increases to a maximum and then at a certain point falls to zero and below. The maximum value of distance traveled can be found by setting to zero the derivate of x_{02} with respect to Δt_h. The result is

$$\begin{gathered}
\frac{2V_LJ_c}{A_c^2} = \sqrt{\frac{8V_L\Delta t_{h_{02}}}{A_c}\left(\frac{J_c}{A_c}\right)^2 - \frac{1}{3}} \\
\text{or} \quad \frac{8V_L\Delta t_{h2}}{A_c}\left(\frac{J_c}{A_c}\right)^2 = \left(\frac{2V_LJ_c}{A_c^2}\right)^2 + \frac{1}{3} \qquad (18) \\
\text{or} \quad \Delta \mathrm{t_h} = \frac{1}{2A_c}\left[V_L + \frac{1}{3}\left(\frac{A_c^2}{2J_c}\right)^2 \frac{1}{V_L}\right]
\end{gathered}$$

To obtain the maximum on-line deceleration distance, substitute equation (18) into equation (15). This is not done here because Δt_h at the maximum distance is much too large to be of interest in short-headway PRT systems.

8. Solving equation (15) for $\Delta t_{h_{02}}$

In the numerical solution for the transition, we calculate x_{02} and need to calculate the corresponding value of $\Delta t_{h_{02}}$. To do so, rewrite equation (15) in the form

$$\frac{x_{02}}{V_L} - \frac{A_c}{2J_c} + \Delta t_{h_{02}} = \frac{A_c}{2J_c}\left[\sqrt{\frac{8V_L J_c^2}{A_c^3}\Delta t_{h2} - \frac{1}{3}}\right]$$

$$\left(\frac{x_{02}}{V_L}-\frac{A_c}{2J_c}\right)^2 + 2\left(\frac{x_{02}}{V_L}-\frac{A_c}{2J_c}\right)\Delta t_{h_{02}} + \Delta t_{h_{02}}^2 = \frac{A_c^2}{4J_c^2}\left(\frac{8V_L J_c^2}{A_c^3}\Delta t_{h2} - \frac{1}{3}\right)$$

$$\Delta t_{h_{02}}^2 - 2\left(\frac{V_L}{A_c}+\frac{A_c}{2J_c}-\frac{x_{02}}{V_L}\right)\Delta t_{h_{02}} + \frac{x_{02}}{V_L}\left(\frac{x_{02}}{V_L}-\frac{A_c}{J_c}\right)+\frac{A_c^2}{3J_c^2}=0 \qquad (19)$$

The radical is

$$\left(\frac{V_L}{A_c}+\frac{A_c}{2J_c}-\frac{x_{02}}{V_L}\right)^2 - \frac{x_{02}}{V_L}\left(\frac{x_{02}}{V_L}-\frac{A_c}{J_c}\right)-\frac{A_c^2}{3J_c^2}=$$

$$\left(\frac{V_L}{A_c}+\frac{A_c}{2J_c}\right)^2 - 2\left(\frac{V_L}{A_c}\right)\frac{x_{02}}{V_L} - \frac{A_c^2}{3J_c^2} = \frac{V_L}{A_c}\left(\frac{V_L}{A_c}+\frac{A_c}{J_c}-2\frac{x_{02}}{V_L}\right)-\frac{A_c^2}{12J_c^2}$$

Therefore

$$\begin{aligned}\Delta t_{h_{02}} &= \frac{V_L}{A_c}+\frac{A_c}{2J_c}-\frac{x_{02}}{V_L}-\sqrt{\frac{V_L}{A_c}\left(\frac{V_L}{A_c}+\frac{A_c}{J_c}-2\frac{x_{02}}{V_L}\right)-\frac{A_c^2}{12J_c^2}}\\ &= \frac{1}{V_L}\left[V_L\left(\frac{V_L}{A_c}+\frac{A_c}{J_c}\right)-x_{02}\right]-\frac{A_c}{2J_c}-\sqrt{\frac{2}{A_c}\left[\frac{V_L}{2}\left(\frac{V_L}{A_c}+\frac{A_c}{J_c}\right)-x_{02}\right]-\frac{A_c^2}{12J_c^2}} \qquad (20)\\ &= \frac{1}{V_L}\left(2D_{stop}-x_{02}-\frac{A_c}{2J_c}V_L\right)-\sqrt{\frac{2}{A_c}\left(D_{stop}-x_{02}-\frac{A_c^3}{24J_c^2}\right)}\end{aligned}$$

in which D_{stop} is the stopping distance from speed V_L. Since D_{stop} must be substantially longer than x_{02}, the term under the square-root sign is always positive in practical cases. The minus sign before the radical is the correct one because equation (20) then reduces to equation (5) if x_{01} is substituted for x_{02} from equation (3). Equation (20) is used in the program developed for the numerical solution for the transition to an off-line station.

9. <u>Speed at End of on-line deceleration.</u>

From equations (8), (3), and (12) the speed at the end of the period of on-line deceleration is

$$V_2 = V_L - \frac{A_c^2}{2J_c} - A_c\Delta t_{12} = V_L - \frac{A_c}{2J_c}\sqrt{\frac{8V_L J_c^2}{A_c^3}\Delta t_{h_{02}} - \frac{1}{3}} \qquad (21)$$

in which

$$\Delta t_{h_{02}} \geq \frac{A_c^3}{6V_L J_c^2}.$$

Equation (21) is calculated in the following Excel spreadsheet.

Speed at End of On-Line Deceleration, m/s

g =	9.80665	m/s^2
Jc =	2.45166	m/s^3
Ac =	2.45166	
Ac^2/2Jc =	1.22583	m/s
8Jc^2/Ac^3 =	3.26309	1/sec

	VL, m/s						
Headway Lost, sec	10	11	12	13	14	15	16
0.02	9.31	10.24	11.18	12.12	13.07	14.02	14.97
0.03	9.02	9.94	10.88	11.81	12.75	13.69	14.64
0.04	8.79	9.71	10.64	11.57	12.50	13.44	14.38
0.05	8.60	9.52	10.44	11.36	12.29	13.22	14.15
0.06	8.44	9.35	10.26	11.18	12.10	13.02	13.95
0.07	8.29	9.19	10.10	11.01	11.93	12.84	13.77
0.08	8.15	9.05	9.95	10.86	11.77	12.68	13.60
0.09	8.02	8.91	9.81	10.71	11.62	12.53	13.44
0.1	7.90	8.79	9.68	10.58	11.48	12.38	13.29
0.11	7.79	8.67	9.56	10.45	11.34	12.25	13.15
0.12	7.68	8.56	9.44	10.33	11.22	12.11	13.01

10. <u>On-Line Deceleration distance as a function of end speed.</u>

We need to know the distance during on-line deceleration as a function of the speed V_2 at the clearance point following which the vehicle is offline. This distance is given by equation (13), in which, from equations (8) and (3)

$$\Delta t_{12} = \frac{1}{A_c}\left(V_L - \frac{A_c^2}{2J_c} - V_2 \right) = \frac{V_L}{A_c} - \frac{V_2}{A_c} - \frac{A_c}{2J_c}$$

Substituting this value into equation (13), we get

$$\begin{aligned} x_{02} = x_{01} + x_{12} &= \frac{A_c}{J_c}\left(V_L - \frac{A_c^2}{6J_c} \right) + \Delta t_{12}\left(V_L - \frac{A_c^2}{2J_c} - \frac{A_c}{2}\Delta t_{12} \right) \\ &= \frac{A_c}{J_c}V_L - \frac{A_c^3}{6J_c^2} + V_L\left(\frac{V_L}{A_c} - \frac{V_2}{A_c} - \frac{A_c}{2J_c} \right) - \frac{A_c^2}{2J_c}\left(\frac{V_L}{A_c} - \frac{V_2}{A_c} - \frac{A_c}{2J_c} \right) - \frac{A_c}{2}\Delta t_{12}^2 \\ &= \frac{V_L^2}{A_c} - \frac{V_L V_2}{A_c} + \frac{A_c}{2J_c}V_2 + \frac{A_c^3}{12J_c^2} - \frac{A_c}{2}\left[\left(\frac{V_L}{A_c} - \frac{V_2}{A_c} \right)^2 - \frac{A_c}{J_c}\left(\frac{V_L}{A_c} - \frac{V_2}{A_c} \right) + \frac{A_c^2}{4J_c^2} \right] \end{aligned}$$

$$x_{02} = \frac{V_L}{2}\left(\frac{V_L}{A_c} + \frac{A_c}{J_c} \right) - \frac{A_c^3}{24J_c^2} - \frac{V_2^2}{2A_c} \qquad (22)$$

Numerical values from Equation (22) are shown in the following Excel Spreadsheet.

On-Line Deceleration Length as function of End Speed V2

g =	9.80665	m/s^2
Jc =	2.45166	m/s^3
Ac =	2.45166	m/s^2
Ac^3/24/Jc^2 =	0.10215	m

	V2						
VL	10	11	12	13	14	15	16
10	4.90						
11	9.68	5.40					
12	14.87	10.59	5.90				
13	20.47	16.19	11.50	6.40			
14	26.48	22.19	17.50	12.40	6.90		
15	32.89	28.61	23.92	18.82	13.31	7.40	
16	39.71	35.43	30.74	25.64	20.13	14.22	7.90

11. <u>The declining speed as a function of distance along the transition into an off-line station.</u>

Speed declines along the transition into an off-line station at content deceleration A_c. Thus, along the transition

$$V = \frac{ds}{dt} = V_o - A_c t$$

Thus $s = V_o t - A_c \frac{t^2}{2}$ or $\left(A_c t\right)^2 - 2V_o\left(A_c t\right) + 2A_c s = 0$

Thus $A_c t = V_o - \sqrt{V_o^2 - 2A_c s}$.

Thus $V = V_o\sqrt{1 - \frac{2A_c s}{V_o^2}}$, or $s = \frac{\left(V_o - V\right)}{2A_c}\left(2V_o - V_o + V\right) = \frac{V_o^2 - V^2}{2A_c}$

12. <u>Curves of on-line deceleration as a function of speed and headway sacrificed</u>.

Equations (7) and (15) are plotted in Figure 2 for a useful range of line speeds. For a small sacrifice of on-line headway of say 0.1 sec, the savings in off-line guideway that would have to be provided if all of the deceleration were offline is seen to be substantial. Figure 3 shows the on-line distance traveled as a function of line speed for $\Delta t_h = 0.1$ sec.

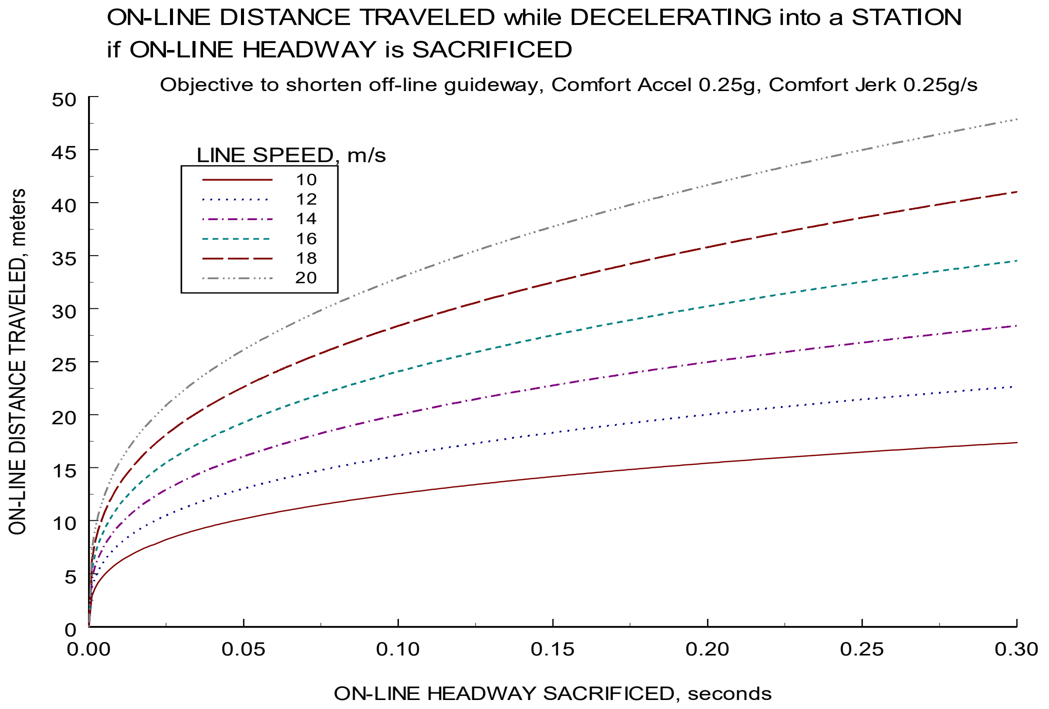


Figure 2

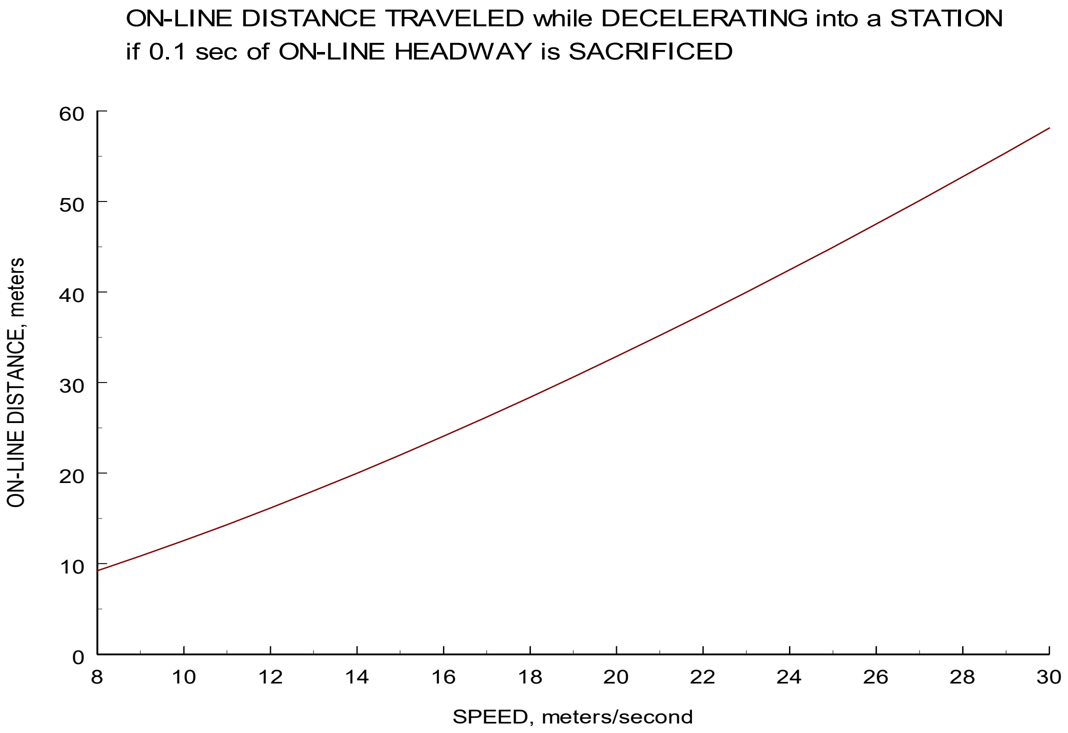

ON-LINE DISTANCE TRAVELED while DECELERATING into a STATION
if 0.1 sec of ON-LINE HEADWAY is SACRIFICED
ON-LINE DISTANCE, meters
SPEED, meters/second
0
10
20
30
40
50
60
8
10
12
14
16
18
20
22
24
26
28
30

Figure 3

Encoder Calibration

When using encoders on the wheels of our ITNS vehicles to measure distance and speed, the distance traveled per pulse is

$$\text{Distance per Pulse } = \frac{\pi D_w}{Pulses/\mathrm{Re}\,v}$$

in which D_w is the diameter of the wheel, which with compliant tires is dependent on the weight on the wheel, and $Pulses/\mathrm{Re}\,v$ is the number of distance steps sensed by the digital encoder per revolution of the wheel, currently 4096.

By measuring the gross weight of the vehicle, the correct wheel diameter can be recorded in the on-board computer, but we must assume that there will be a residual error to be corrected. It can be corrected by sensing a fixed distance as the vehicle leaves the station by means of wayside Hall detectors.

A sudden step in the distance parameter also occurs at each line-to-line branch point. Distance in a network is taken as a negative number that reaches zero at the branch point, and at that point is set to the negative distance to the next line-to-line branch point.

The correction in these two cases is suddenly applied the control system by the code shown in red in the following program, in which the procedure has been tested.

```
Public Class VehicleControl
    'This program VehicleControl simulates the operation of the ITNS vehicle
controller
    'Units are MKS
    'Steps in program:
    '  Start with given speed V0 and acceleration A0
    '  Command Ac(t) and Vc(t) for a maneuver from given A0 and V0
    '    Maneuver 0 => Maintain command speed
    '    Maneuver 1 => Decelerate to stop in x meters
    '    Maneuver 2 => Change speed to x meters/sec in minimum time
    '    Maneuver 3 => Slip x meters while going to line speed VL
    '  Obtain actual distance X(t) and speed V(t) via encoders and subtract
to give
    '    dX = Xc - Xe, dV = Vc - Ve
    '  Form thrust command Tc = Gp * dX + Gv * dV
    '    where Gp = mc * Omega.n^2 * (1 - Beta)
    '          Gv = mc * Omega.n * (.5 * Beta / Zeta + 2 * Zeta * (1 - Beta))
    '            mc = best estimate of vehicle mass
    '          Zeta = damping ratio
    '          Beta = dimensionless factor between 0 and 1
    '           Tau = motor time constant
    '       Omega.n = .5 * Beta / (Zeta * Tau) = radial frequency of
controller
    '
    '  Model motor as Tau * dTh/dt + Thrust = Tc
    '  Model vehicle as   m * dV/dt = Thrust - Drag
    '  Model Drag = c.air * V^2 + m * g * (aRoad + bRoad * V)
    '-------------------------
    'System constants

    Public g As Double = 9.80665              'acceleration of gravity, m/s^2
    Public Jcomfort As Double = 0.25 * g      'comfort jerk, m/s^2
    Public Acomfort As Double = 0.2 * g       'comfort acceleration, m/s^2
    Public Ar As Double = 0.75 * Acomfort     'reduced acceleration for slip
maneuvers, m/s^2
    Public dVr As Double = Ar ^ 2 / Jcomfort  'speed increment, m/s
    Public tJ As Double = Acomfort / Jcomfort 'jerk time constant, s
    Public VL As Single = 15                  'line speed, m/s
    Public Vs As Double = 7                   'station speed, m/s
    Public Vmin As Double = VL / 2            'minimum speed for slip
maneuvers, m/s
    Public dt As Double = 0.00001             'computation-time interval, s
    Public t As Double                        'running time, s
    Public Tm, Dm As Double                   'maneuver time, distance

    'Vehicle parameters
    Public m As Double = 700                'actual vehicle mass, kg
    Public aRoad As Double = 0.005          'road resistance per unit weight
    Public bRoad As Double = 0.0005         'road resistance per unit
weight/speed
    Public Rho As Double = 1.2              'air density, kg/m^3
    Public CdA As Double = 8                'effective frontal area, m^2
    Public cAir As Double = 0.5 * Rho * CdA 'air drag per unit speed^2

    'Controller parameters
    Public dtc As Double = 0.005            'time interval between control
updates, s
    Public dVe As Double = 0.0001           'for brake control
    Public Tau As Double = 0.1              'thruster lag time, s
    Public mc As Double = 900               'vehicle mass used in control
system, kg
    Public Zeta As Double = 0.6             'dimensionless damping constant
    Public Beta As Double = 0.65            'dimensionless constant between 0
and 1
    Public OmegaN As Double = Beta / (2 * Zeta * Tau)  'controller radial
frequency, rad/s
    Public Gp As Double = mc * OmegaN ^ 2 * (1 - Beta)      'position gain
    Public Gv As Double = mc * OmegaN * (0.5 * Beta / Zeta + 2 * Zeta * (1 -
Beta)) 'speed gain
    Public Dw As Double = 13.25 / 12 * 0.3048      'encoder wheel diameter, m
    Public PulsesPerRev As Integer = 4096   'pulses per revolution of the
wheel
    Public dXenc As Double = Math.PI * Dw / PulsesPerRev   'encoder step, m
    Public Bm As Double = 0.2 * m * g       'initial braking rate, N
    Public V0 As Double     'speed at t = 0
    Public A0 As Double     'acceleration at t = 0

    Public t1, t2, t3, t4, t5, t6, t7, t8 As Double
    Public A1, A2, A3, A4, A5, A6, dV0 As Double
    Public Jerk01, Jerk12, Jerk23, Jerk34 As Double
    Public V1, V2, V3, V4, V5, V6, V7 As Double
    Public x1, x2, x3, x4, X5, X6, X7, X8 As Double
    Public Ac, Vc, Xc As Double 'command acceleration, speed, distance

    Public Vfinal, Dstop, Slip As Double  'input parameter for maneuver 2,3,4
    Public ManeuverNo As Integer
    Public Ne As Long = 0  'encoder counter
    Public Xjump As Double = 0          'Occures at line-to-line branch point
in Maneuver 0

    'Screen parameters
    Public Y0 As Single = 720
    Public T0 As Single = 300
    Public tScale As Single = 80
    Public aScale As Single = 400
    Public vScale As Single = 30
    Public xScale As Single = 5
    Public Thscale As Single = 0.5
    Public pScale As Single = 20

    Dim objGraphics As System.Drawing.Graphics
    Dim objFont As Font

    Sub Control()
        Dim tStart, tCount As Double                                  'time
parameters
        Dim Thrust, Tc, dThdtOld, dThdt As Double                     'thrust
parameters
        Dim dVdt, dVdtOld, Jerk As Double
'acceleration and jerk parameters
        Dim V, Ve, Vold, dV, VeOld As Double                          'speed
parameters
        Dim Xstart, X, Xe, XeOld, dX As Double
'distance parameters
        Dim Xend As Double = 0
        Dim xGraph, yGraph As Single

        Select Case ManeuverNo
            Case 0
                Tm = 5
                Ac = 0
                Vc = V0
            Case 1
                setManeuver1()  'Stops vehicle in distance Dstop, meters.
            Case 2
                setManeuver2()  'Changes vehicle speed to Vfinal, m/s
            Case 3
                setManeuver3()  'Causes vehicle to slip Slip meters.
        End Select

        'Set values at starting point, t = tStart
        tStart = -2.5                                  'allow time for system to
settle
        t = tStart                                     'running time
        tCount = tStart                                'tcount increases in
increments of dtc
        V = V0 + A0 * tStart                           'actual speed
        Ve = V0
        dVdt = A0                                      'actual acceleration
        dVdtOld = A0
        Thrust = A0 * m + Drag(V)                      'actual thrust
        dThdtOld = 0                                   'change in thrust
        Xstart = tStart * (V0 + 0.5 * A0 * tStart)  'distance, so X = 0 when
t = 0
        X = Xstart                                  'actual distance at start
        XeOld = Xstart - V0 * dtc                   'previous measured
distance
        Jerk = 0

        objFont = New System.Drawing.Font("Arial", 40)

        objGraphics = Me.CreateGraphics
        objGraphics.DrawLine(Pens.White, T0, Y0, T0, 0)
        objGraphics.DrawLine(Pens.White, T0, Y0, 1500, Y0)
        objGraphics.DrawLine(Pens.Red, T0, Y0 - vScale * VL, 1500, Y0 -
vScale * VL)
        For i As Integer = 1 To 30
            objGraphics.DrawLine(Pens.White, T0 + tScale * i, Y0, T0 + tScale
* i, Y0 - 10)
        Next
        objGraphics.DrawString(" Command Acceleration ", Me.Font,
System.Drawing.Brushes.White, 500, 30)
        objGraphics.DrawString(" Command Speed ", Me.Font,
System.Drawing.Brushes.Pink, 500, 50)
        objGraphics.DrawString(" Command Distance ", Me.Font,
System.Drawing.Brushes.Fuchsia, 500, 70)
        objGraphics.DrawString(" Acceleration ", Me.Font,
System.Drawing.Brushes.Yellow, 500, 90)
        objGraphics.DrawString(" Speed ", Me.Font,
System.Drawing.Brushes.Red, 500, 110)
        objGraphics.DrawString(" Distance ", Me.Font,
System.Drawing.Brushes.Turquoise, 500, 130)
        objGraphics.DrawString(" Thrust ", Me.Font,
System.Drawing.Brushes.Gray, 500, 150)
        objGraphics.DrawString(" Acceleration Power ", Me.Font,
System.Drawing.Brushes.GreenYellow, 500, 170)
        objGraphics.DrawString(" Jerk ", Me.Font,
System.Drawing.Brushes.Goldenrod, 500, 190)

        Do
            xGraph = T0 + tScale * t
            yGraph = Y0 - aScale * Ac
            objGraphics.FillEllipse(Brushes.White, xGraph, yGraph, 2, 2)
            yGraph = Y0 - vScale * Vc
            objGraphics.FillEllipse(Brushes.Pink, xGraph, yGraph, 2, 2)
            yGraph = Y0 - xScale * Xc
            objGraphics.FillEllipse(Brushes.Fuchsia, xGraph, yGraph, 2, 2)
            yGraph = Y0 - aScale * dVdt
            objGraphics.FillEllipse(Brushes.Yellow, xGraph, yGraph, 2, 2)
            yGraph = Y0 - vScale * V
            objGraphics.FillEllipse(Brushes.Red, xGraph, yGraph, 2, 2)
            yGraph = Y0 - xScale * X
            objGraphics.FillEllipse(Brushes.Turquoise, xGraph, yGraph, 2, 2)
            yGraph = Y0 - Thscale * (Thrust - 500)
            objGraphics.FillEllipse(Brushes.Gray, xGraph, yGraph, 2, 2)
            yGraph = Y0 - pScale * dVdt * V    'acceleration power
            objGraphics.FillEllipse(Brushes.GreenYellow, xGraph, yGraph, 2,
2)
            yGraph = Y0 - 0.01 * aScale * Jerk
            'objGraphics.FillEllipse(Brushes.Goldenrod, xGraph, yGraph, 2, 2)

            'Simulate digital encoder
            Xe = Encoder(X, Xstart)       'measured position

            'Enter the on-board computer
            If t >= tCount Then
                tCount = tCount + dtc
                If t < 0 Then
                    Ac = A0
                    Vc = V0 + A0 * t
                    Xc = t * (V0 + A0 * t / 2)
                End If
                Select Case ManeuverNo
                    Case 0
                        If t >= 0 Then
                            Xjump = -500
                            Xc = Xjump + V0 * t
                            If t < dtc Then
                                Xstart = Xstart + Xjump
                                Xe = Xe + Xjump
                                XeOld = XeOld + Xjump
                                X = X + Xjump
                            End If
                        End If
                    Case 1
                        If t > 0 Then
                            runManeuver1()  'output Ac, Vc, Xc
                        End If
                    Case 2
                        If t > 0 Then
                            runManeuver2()  'output Ac, Vc, Xc
                        End If
                    Case 3
                        If t > 0 Then
                            runManeuver3()  'output Ac, Vc, Xc
                        End If
                End Select

                Ve = (Xe - XeOld) / dtc       'differentiate to measure speed
                VeOld = Ve
                XeOld = Xe

                dX = Xc - Xe            'command position - measured position
                dV = Vc - Ve            'command speed - measured speed
                Tc = Gp * dX + Gv * dV  'command thrust

                If Vfinal < 0.01 And Ve < dVe Then Tc = 0
            End If

            'Simulate thruster as first-order lag
            dThdt = (Tc - Thrust) / Tau             'time rate of change of
thrust
            Thrust = Thrust + 0.5 * dt * (3 * dThdt - dThdtOld)   'actual
thrust
            dThdtOld = dThdt
            'If ManeuverNo = 1 And (t >= Tm Or V < 0) Then Thrust = 0
            'Simulate vehicle dynamics
            dVdt = (Thrust - Drag(V)) / m    'acceleration
            Jerk = (dVdt - dVdtOld) / dt
            If Vfinal < 0.01 And Ve < dVe Then
                dVdt = dVdt - Brake(V) / m
            End If

            Vold = V
            V = V + 0.5 * dt * (3 * dVdt - dVdtOld)    'speed
            dVdtOld = dVdt
            X = X + 0.5 * dt * (V + Vold)              'position

            If t >= Tm And Xend = 0 Then
                Xend = X
            End If

            t = t + dt
            Application.DoEvents()
        Loop Until t > 1.1 * Tm

        objGraphics.DrawString(" The Maneuver Time is " &
FormatNumber(CSng(Tm), 2) & " sec", Me.Font, _
                               System.Drawing.Brushes.White, 900, 200)
        If ManeuverNo = 1 Then 'Stop in given distance Dstop
            objGraphics.DrawString(" The commanded Maneuver Distance is " &
FormatNumber(CSng(Dstop), 2) & " meters", Me.Font, _
                               System.Drawing.Brushes.White, 900, 220)
            objGraphics.DrawString(" The actual distance at maneuver end is "
& FormatNumber(CSng(Xend), 2) & " meters", Me.Font, _
                                   System.Drawing.Brushes.White, 900, 240)
        ElseIf ManeuverNo = 2 Then  'Change speed to Vfinal
            objGraphics.DrawString(" The commanded Final Speed is " &
FormatNumber(CSng(Vfinal), 2) & " m/s", Me.Font, _
                               System.Drawing.Brushes.White, 900, 220)
            objGraphics.DrawString(" The actual Final Speed is " &
FormatNumber(CSng(V), 2) & " m/s", Me.Font, _
                                   System.Drawing.Brushes.White, 900, 240)
        ElseIf ManeuverNo = 3 Then
            objGraphics.DrawString(" The commanded Slip Distance is " &
FormatNumber(CSng(Slip), 2) & " meters", Me.Font, _
                               System.Drawing.Brushes.White, 900, 220)
            objGraphics.DrawString(" The actual slip Distance is " &
FormatNumber(CSng(VL * Tm - Dm), 2) & " meters", Me.Font, _
                                   System.Drawing.Brushes.White, 900, 240)
        End If

        objGraphics.Dispose()
        objFont.Dispose()
    End Sub

    Function Brake(ByVal V As Double) As Double
        Dim dBm As Double
        dBm = 0.002 * m * g
        If ManeuverNo = 1 Then
            If t >= Tm Or V < 0 Then
                Bm = Bm - dBm
                If Bm < 0 Then Bm = 0
                If V > 0 Then
                    Brake = Bm
                ElseIf V < 0 Then
                    Brake = -Bm
                Else
                    Brake = 0
                End If
            Else
                Brake = 0
            End If
        Else
            Brake = 0
        End If
    End Function

    Function Drag(ByVal V As Double) As Double
        Dim D As Double
        D = Math.Sign(V) * cAir * V ^ 2 + m * g * (bRoad * V + Math.Sign(V) *
aRoad)
        If ManeuverNo = 1 And V < 0.1 Then D = 0
        Drag = D
    End Function

    Function Encoder(ByVal X As Double, ByVal Xstart As Double) As Double
'STATIC
        If X >= Xstart + (Ne + 0.5) * dXenc Then Ne = Ne + 1
        Encoder = Ne * dXenc + Xstart
    End Function

    Sub runManeuver1()
        Dim Jerk As Double
        If t < t2 Then
            Jerk = -Math.Sign(dV0) * Jcomfort
            State(t, Jerk, A0, V0, 0)       'Ac = command acceleration
        ElseIf t < t3 Then                  'Vc = command speed
            Jerk = 0                        'Xc = command distance travelled
            State(t - t2, Jerk, A2, V2, x2)
        ElseIf t < t4 Then
            Jerk = Math.Sign(dV0) * Jcomfort
            State(t - t3, Jerk, A2, V3, x3)
        ElseIf t < t5 Then
            Jerk = 0
            State(t - t4, Jerk, 0, V4, x4)
        ElseIf t < t6 Then
            Jerk = -Jcomfort
            State(t - t5, Jerk, 0, V4, X5)
        ElseIf t < t7 Then
            Jerk = 0
            State(t - t6, Jerk, A6, V6, X6)
        ElseIf t < t8 Then
            Jerk = Jcomfort
            State(t - t7, Jerk, A6, V7, X7)
        ElseIf t >= t8 Then
            Jerk = 0
            State(t - t8, Jerk, 0, 0, X8)
        End If
    End Sub

    Sub runManeuver2()
        Dim AcOld As Double
        AcOld = Ac
        If t < t1 Then
            State(t, Jerk01, A0, V0, 0)   'Ac is command acceleration
        ElseIf t < t2 Then                'Vc is command speed
            State(t - t1, Jerk12, A1, V1, x1)
        ElseIf t < t3 Then
            State(t - t2, Jerk23, A2, V2, x2)
        ElseIf t < t4 Then
            State(t - t3, Jerk34, A3, V3, x3)
        Else
            State(t - t4, 0, 0, Vfinal, x4)
        End If
    End Sub

    Sub runManeuver3()
        Dim Jerk, A1a, V1a As Double
        If t < t1 Then
            Jerk = -Jcomfort
            State(t, Jerk, A0, V0, 0)
        ElseIf t < t2 Then
            Jerk = -Jcomfort
            If A0 >= 0 Then
                A1a = A1
                V1a = V1
            Else
                A1a = A0
                V1a = V0
            End If
            State(t - t1, Jerk, A1a, V1a, x1)
        ElseIf t < t3 Then                      'Vc is command speed
            Jerk = 0                            'Xc is command distance
            State(t - t2, Jerk, A2, V2, x2)
        ElseIf t < t4 Then
            Jerk = Jcomfort
            State(t - t3, Jerk, A2, V3, x3)
        ElseIf t < t5 Then
            Jerk = 0
            State(t - t4, Jerk, A4, V4, x4)
        ElseIf t < t6 Then
            Jerk = Jcomfort
            State(t - t5, Jerk, A5, V5, X5)
        ElseIf t < t7 Then
            Jerk = 0
            State(t - t6, Jerk, A6, V6, X6)
        ElseIf t < t8 Then
            Jerk = -Jcomfort
            State(t - t7, Jerk, A6, V7, X7)
        Else
            Jerk = 0
            State(t - t8, Jerk, 0, VL, X8)
        End If
    End Sub
```

```
    'This maneuver stops a vehicle in a given distance
    Sub setManeuver1()
        Dim Tmin, Dmin As Double
        Dim dt01, dt12, dt23, dt34, dt45, dt56, dt67, dt78 As Double
        Dim dx01, dx12, dx23, dx34, dx45, dx56, dx67, dx78, dx14, dx58 As
Double
        Dim Dbnd, b, dV, Dold, V5 As Double

        Vfinal = 0
        If Math.Abs(A0) > Acomfort Then A0 = Math.Sign(A0) * Acomfort 'can't
exceed Acomfort
        'Condition of negative V not operational on deceleration to stop
        If A0 < 0 And V0 < A0 ^ 2 / Jcomfort Then V0 = A0 ^ 2 / Jcomfort
        dV0 = V0 + A0 ^ 2 / 2 / Jcomfort - Vs     'indicator

        'Calculate minimum stopping time and distance
        V1 = V0 + A0 ^ 2 / 2 / Jcomfort
        If V1 >= A0 ^ 2 / Jcomfort Then
            Tmin = V1 / Acomfort + Acomfort / Jcomfort
        ElseIf V1 >= 0 Then
            Tmin = 2 * Math.Sqrt(V1 / Jcomfort)
        Else
            Tmin = 0
        End If
        Dmin = 0.5 * V1 * Tmin + (A0 / Jcomfort) * (V0 + A0 ^ 2 / 3 /
Jcomfort)

        dt01 = Math.Sign(dV0) * A0 / Jcomfort
        V1 = V0 + Math.Sign(dV0) * A0 ^ 2 / 2 / Jcomfort
        dx01 = (A0 / Jcomfort) * (Math.Sign(dV0) * V0 + A0 ^ 2 / 3 /
Jcomfort)

        If Dstop < Dmin Then
            objGraphics.DrawString(" The Minimum Maneuver Distance is " &
FormatNumber(Dmin, 2) & " meters", Me.Font, _
                               System.Drawing.Brushes.White, 900, 220)
        End If


        V4 = Vs   'if V0 < Vs V4 may later be reduced below Vs

        If Math.Abs(V4 - V1) >= Acomfort * tJ Then
            A2 = -Math.Sign(dV0) * Acomfort
        ElseIf Math.Abs(V4 - V1) > 0 Then
            A2 = -Math.Sign(dV0) * Math.Sqrt(Jcomfort * Math.Abs(V4 - V1))
        Else
            A2 = 0
        End If

        If V4 >= Acomfort * tJ Then
            A6 = -Acomfort
        ElseIf V4 > 0 Then
            A6 = -Math.Sqrt(Jcomfort * V4)      'V4 > 0 if there is any
maneuver at all
        Else
            A6 = 0
        End If
        'Calculate boundry stopping distance if dx45 = 0 and V4 = Vs
        If A2 <> 0 Then
            dx14 = 0.5 * (V4 + V1) * ((V4 - V1) / A2 - Math.Sign(dV0) * A2 /
Jcomfort)
        Else
            dx14 = 0
        End If
        If A6 <> 0 Then dx58 = -0.5 * V4 * (V4 / A6 + A6 / Jcomfort) Else
dx58 = 0
        Dbnd = dx14 + dx58 + dx01

        A4 = 0    'true always
        A5 = A4   'true always
        If Dstop >= Dbnd Then
            dx45 = Dstop - Dbnd
        Else
            dx45 = 0
            If dV0 > 0 And Dstop < Dbnd Then 'in these cases don't slow down
near Vs
                b = (Jcomfort / V1) * (Dstop - (A0 / Jcomfort) * (V0 + A0 ^ 2
/ 3 / Jcomfort))
                A2 = -b + Math.Sqrt(b ^ 2 - V1 * Jcomfort)        'reduced
deceleration
            Else
                dV = 0.005     'increment in which V4 is reduced if need be
                Do
                    Dold = Dbnd 'used in Newtonian intepolation after do-loop
                    'distance from point 1 to 4:
                    If A2 <> 0 Then
                        dx14 = 0.5 * (V4 + V1) * ((V4 - V1) / A2 -
Math.Sign(dV0) * A2 / Jcomfort)
                    Else
                        dx14 = 0
                    End If
                    'distance from point 5 to 8:
                    If A6 <> 0 Then
                        dx58 = -0.5 * V4 * (V4 / A6 + A6 / Jcomfort)
                    Else
                        dx58 = 0
                    End If
                    'boundry distance above which dx45 > 0
                    Dbnd = dx14 + dx58 + dx01

                    If Dbnd < Dstop Then    'if true do-loop is finished
                        Exit Do
                    Else
                        V4 = V4 - dV       'step V4 down until Dbnd = Dstop
                        If V4 < dV Then    'this condition should never occur
                            V4 = dV
                            Exit Do
                        End If
                    End If

                    If Math.Abs(V4 - V1) >= Acomfort * tJ Then   'recalculate
A2 with lower V4
                        A2 = -Math.Sign(dV0) * Acomfort
                    ElseIf Math.Abs(V4 - V1) > 0 Then
                        A2 = -Math.Sign(dV0) * Math.Sqrt(Jcomfort *
Math.Abs(V4 - V1))
                    Else
                        A2 = 0
                    End If

                    If V4 > Acomfort * tJ Then
                        A6 = -Acomfort
                    ElseIf V4 > 0 Then
                        A6 = -Math.Sqrt(Jcomfort * V4)
                    Else
                        A6 = 0
                    End If
                Loop

                V4 = V4 + dV * (Dstop - Dbnd) / (Dold - Dbnd) 'Newtonian
interpolation
                If V4 < dV Then V4 = dV

                If Math.Abs(V4 - V1) >= Acomfort * tJ Then  'recalculation of
A2 with final V4
                    A2 = Math.Sign(Vs - V0) * Acomfort
                ElseIf Math.Abs(V4 - V1) > 0 Then
                    A2 = Math.Sign(Vs - V0) * Math.Sqrt(Jcomfort *
Math.Abs(V4 - V1))
                Else
                    A2 = 0
                End If

                If V4 > Acomfort * tJ Then
                    A6 = -Acomfort
                ElseIf V4 > 0 Then
                    A6 = -Math.Sqrt(Jcomfort * V4)
                Else
                    A6 = 0
                End If
            End If
        End If

        dt12 = Math.Abs(A2) / Jcomfort
        V2 = V1 + dt12 * A2 / 2
        dx12 = dt12 * (V1 + dt12 * A2 / 6)

        If dV0 >= 0 And Dstop < Dbnd Then    'special case of no slowdown at
Vs
            dt34 = -A2 / Jcomfort
            V3 = -dt34 * A2 / 2
            dx34 = dt34 * (V3 + dt34 * A2 / 3)
            If A2 <> 0 Then dt23 = (V3 - V2) / A2 Else dt23 = 0
            dx23 = dt23 * (V2 + dt23 * A2 / 2)

            dt45 = 0
            dx45 = 0
            dt56 = 0
            dx56 = 0
            dt67 = 0
            dx67 = 0
            dt78 = 0
            dx78 = 0
        Else              'all other cases
            dt34 = -Math.Sign(dV0) * A2 / Jcomfort
            V3 = V4 - dt34 * A2 / 2
            dx34 = dt34 * (V3 + dt34 * A2 / 3)

            If A2 <> 0 Then dt23 = (V3 - V2) / A2 Else dt23 = 0
            dx23 = dt23 * (V2 + dt23 * A2 / 2)

            dt45 = dx45 / Vs
            V5 = V4

            dt56 = -A6 / Jcomfort
            V6 = V5 + dt56 * A6 / 2
            dx56 = dt56 * (V5 + dt56 * A6 / 6)

            dt78 = -A6 / Jcomfort
            V7 = -dt78 * A6 / 2
            dx78 = dt78 * (V7 + dt78 * A6 / 3)

            If A6 <> 0 Then dt67 = (V7 - V6) / A6 Else dt67 = 0
            dx67 = dt67 * (V6 + dt67 * A6 / 2)
        End If

        t1 = dt01         'record all times where jerk change
        t2 = t1 + dt12
        t3 = t2 + dt23
        t4 = t3 + dt34
        t5 = t4 + dt45
        t6 = t5 + dt56
        t7 = t6 + dt67
        t8 = t7 + dt78
        Tm = t8           'maneuver time

        x1 = dx01         'record all distances where jerk changes
        x2 = x1 + dx12
        x3 = x2 + dx23
        x4 = x3 + dx34
        X5 = x4 + dx45
        X6 = X5 + dx56
        X7 = X6 + dx67
        X8 = X7 + dx78
        Dm = X8           'maneuver distance
    End Sub

    'This maneuver changes speed to speed Vfinal
    Sub setManeuver2()
        'This maneuver changes speed to Vfinal
        Dim Alpha, Beta, dVc, dVo As Double
        Dim Jc, Jn As Double
        Dim Vb, Va As Double
        Dim dt01, dt12, dt23, dt34, dx01, dx12, dx23, dx34, Dm As Double
        Dim Flag1, Flag2 As Integer

        Alpha = 0.5
```

```
        Beta = 0.5
        dVc = Acomfort ^ 2 / 2 / Jcomfort
        dVo = A0 ^ 2 / 2 / Jcomfort

        Jc = Jcomfort
        Vb = CSng(Beta ^ 2) * dVc
        Va = Alpha * VL    'boundary speed, above which A < Acomfort

        Jn = CSng((1 - Beta ^ 2) * Acomfort ^ 2 / 2 / (VL - Va - Beta ^ 2 *
dVc))

        'Treat small changes in speed separately:
        Flag1 = 0
        If Math.Abs(Vfinal - V0) + dVo <= 2 * dVc Then
            If V0 + Math.Sign(A0) * dVo >= Vfinal Then
                Jerk01 = -Jc
                Jerk34 = Jc
                A1 = -CSng(Math.Sqrt(Jc * (V0 + dVo - Vfinal)))
                V1 = (V0 + dVo + Vfinal) / 2
                Flag1 = 1
            Else
                Jerk01 = Jc
                Jerk34 = -Jc
                A1 = CSng(Math.Sqrt(Jc * (Vfinal - V0 + dVo)))
                V1 = (Vfinal + V0 - dVo) / 2
                Flag1 = 2
            End If
            A2 = A1
            A3 = A1
            V2 = V1
            V3 = V1
        End If

        Flag2 = 0
        If Flag1 = 0 Then
            If Vfinal > Va Then
                If V0 + Math.Sign(A0) * dVo > Vfinal Then
                    Jerk01 = -Jc
                    Jerk12 = 0    'dt12 = 0
                    Jerk23 = -Jn
                    Jerk34 = Jc
                    V1 = (V0 + dVo - dVc - Va * Jn / Jc) / (1 - Jn / Jc)
                    A1 = -CSng(Math.Sqrt(Acomfort ^ 2 - 2 * Jn * (V1 - Va)))
                    A2 = A1
                    V2 = V1
                    V3 = (Vfinal + dVc + Va * Jn / Jc) / (1 + Jn / Jc)
                    A3 = -CSng(Math.Sqrt(Acomfort ^ 2 - 2 * Jn * (V3 - Va)))
                    If A0 < 0 And V1 > V0 Then
                        A1 = A0
                        V1 = V0
                        A2 = A1
                        V2 = V1
                        Jerk23 = -CSng(A3 ^ 2 - A2 ^ 2) / 2 / (V2 - V3)
                    End If
                    Flag2 = 1
                Else    'V0 + Math.Sign(A0) * dVo <= Vfinal
                    If V0 - dVo + dVc > Va Then
```

```
                        Jerk01 = Jc
                        Jerk12 = 0
                        Jerk23 = -Jn
                        Jerk34 = -Jc
                        V1 = (V0 - dVo + dVc + Va * Jn / Jc) / (1 + Jn / Jc)
                        A1 = CSng(Math.Sqrt(Acomfort ^ 2 - 2 * Jn * (V1 -
Va)))
                        A2 = A1
                        V2 = V1
                        V3 = (Vfinal - dVc - Va * Jn / Jc) / (1 - Jn / Jc)
                        A3 = CSng(Math.Sqrt(Acomfort ^ 2 - 2 * Jn * (V3 -
Va)))
                        If A1 < A0 Then
                            A1 = A0
                            V1 = V0
                            A2 = A1
                            V2 = V1
                            Jerk23 = -CSng(A2 ^ 2 - A3 ^ 2) / 2 / (V3 - V2)
                        End If
                        Flag2 = 2
                    Else    'V0 - dVo + dVc <= Va
                        Jerk01 = Jc
                        Jerk12 = 0
                        Jerk23 = -Jn
                        Jerk34 = -Jc
                        A1 = Acomfort
                        A2 = Acomfort
                        V1 = V0 - dVo + dVc
                        V2 = Va
                        V3 = (Vfinal - dVc - Va * Jn / Jc) / (1 - Jn / Jc)
                        A3 = CSng(Math.Sqrt(Acomfort ^ 2 - 2 * Jn * (V3 -
Va)))
                        If V3 < Va Then
                            A3 = Acomfort
                            V3 = Vfinal - CSng(A3 ^ 2) / 2 / Jc
                            V2 = V3
                        End If
                        If V1 > V2 Then
                            V2 = V1
                            Jerk23 = -CSng(A2 ^ 2 - A3 ^ 2) / 2 / (V3 - V2)
                        End If
                        Flag2 = 3
                    End If
                End If
            Else  'Vfinal <= Va)
                If V0 + Math.Sign(A0) * dVo > Vfinal Then
                    If V0 + dVo - dVc > Va Then
                        Jerk01 = -Jc
                        Jerk12 = -Jn    'dt12 = 0
                        Jerk23 = 0
                        Jerk34 = Jc
                        V1 = (V0 + dVo - dVc - Va * Jn / Jc) / (1 - Jn / Jc)
                        A1 = -CSng(Math.Sqrt(Acomfort ^ 2 - 2 * Jn * (V1 -
Va)))
                        A2 = -Acomfort
                        A3 = -Acomfort
                        V2 = Va
                        If A1 > A0 Then
                            A1 = A0
                            V1 = V0
                            Jerk12 = -CSng(A2 ^ 2 - A1 ^ 2) / 2 / (V1 - V2)
                        End If
                        V3 = Vfinal + dVc
                        If V3 > V2 Then
                            V2 = V3
                            Jerk12 = -CSng(Acomfort ^ 2 - A1 ^ 2) / 2 / (V1 -
V2)
                        End If
                        Flag2 = 4
                    Else  'V0 + dVo - dVc<= Va
                        Jerk01 = -Jc
                        Jerk12 = 0    'dt12 = 0
                        Jerk23 = 0
                        Jerk34 = Jc
                        A1 = -Acomfort
                        A2 = -Acomfort
                        A3 = -Acomfort
                        V1 = V0 + dVo - dVc
                        V2 = V1
                        V3 = Vfinal + dVc
                        Flag2 = 5
                    End If
                Else   'V0 + Math.Sign(A0) * dVo < Vfinal
                    Jerk01 = Jc
                    Jerk12 = 0
                    Jerk23 = 0
                    Jerk34 = -Jc
                    A1 = Acomfort
                    A2 = Acomfort
                    A3 = Acomfort
                    V1 = V0 - dVo + dVc
                    V2 = V1
                    V3 = Vfinal - dVc
                    Flag2 = 6
                End If
            End If
        End If

        dt01 = (A1 - A0) / Jerk01
        If Math.Abs(Jerk12) > 0 Then
            dt12 = (A2 - A1) / Jerk12
        Else
            If Math.Abs(A2) > 0 Or Math.Abs(A1) > 0 Then
                dt12 = 2 * (V2 - V1) / (A2 + A1)
            Else
                dt12 = 0
            End If
        End If
        If Math.Abs(Jerk23) > 0 Then
            dt23 = (A3 - A2) / Jerk23
        Else
            If Math.Abs(A3) > 0 Or Math.Abs(A2) > 0 Then
                dt23 = 2 * (V3 - V2) / (A3 + A2)
            Else
```

```
                dt23 = 0
            End If
        End If
        dt34 = -A3 / Jerk34

        dx01 = dt01 * (V0 + dt01 * (2 * A0 + A1) / 6)
        dx12 = dt12 * (V1 + dt12 * (2 * A1 + A2) / 6)
        dx23 = dt23 * (V2 + dt23 * (2 * A2 + A3) / 6)
        dx34 = dt34 * (V3 + dt34 * A3 / 3)

        t1 = dt01
        t2 = t1 + dt12
        t3 = t2 + dt23
        t4 = t3 + dt34      'Maneuver time
        Tm = t4

        x1 = dx01
        x2 = x1 + dx12
        x3 = x2 + dx23
        x4 = x3 + dx34
        Dm = dx01 + dx12 + dx23 + dx34       'Maneuver distance
    End Sub

    'This maneuver causes vehicle to slip amount Slip before reaching VL
    Sub setManeuver3()
        Dim Smin, S01, DV, V1r As Double
        Dim dt01, dt12, dt23, dt34, dt45, dt56, dt67, dt78 As Double
        Dim dx01, dx12, dx23, dx34, dx45, dx56, dx67, dx78 As Double
        Dim Sbnd1, Sbnd2, Sbnd3, Sbnd4, b, c, slp, SlipError, Va, V4a As
Double
        Dim dV0, V4previous, Sprevious, V4p, Sp As Double
        Dim i, Flag As Integer

        'Calculate point 1 at which A1 = 0 for the purpose of calculating
boundaries
        dt01 = A0 / Jcomfort    '< 0 if A0 < 0, which subtracts S01
        V1 = V0 + dt01 * A0 / 2
        dx01 = dt01 * (V0 + dt01 * A0 / 3)
        S01 = VL * dt01 - dx01  'slip during interval 0-1, > 0 if A0 > 0, < 0
if A0 < 0

        'Calculate V4 in all cases except between Sbnd3 and Sbnd4, where
calculate A1
        dx45 = 0    'cases when not zero will be calculated
        A1 = 0
        A5 = 0
        dV0 = A0 ^ 2 / Jcomfort
        If V0 = VL And A0 = 0 Then
            DV = VL - Vmin
            Sbnd1 = DV * (DV / Ar + Ar / Jcomfort) 'DV > dVr always
            Sbnd2 = 2 * dVr * Ar / Jcomfort
            Sbnd3 = 0
            If Slip >= Sbnd1 Then
                dx45 = (Slip - Sbnd1) * Vmin / (VL - Vmin)
                V4 = Vmin
            ElseIf Slip > 0 Then
                dx45 = 0
                If Slip >= Sbnd2 Then
                    V4 = VL + 0.5F * dVr - Math.Sqrt(Ar * Slip + 0.25 * dVr ^
2)
                Else
                    V4 = VL - (Jcomfort * Slip ^ 2 / 4) ^ (1 / 3)
                End If
            Else
                V4 = VL
            End If
            V5 = V4
        Else
            Sbnd1 = S01 + SlipV4(Vmin)       'boundary when V4=Vmin and
dx45=0
            If V1 - dVr > Vmin Then
                Sbnd2 = S01 + SlipV4(V1 - dVr)   'boundary when V4=V1-dVr
            Else
                Sbnd2 = Sbnd1
            End If

            If A0 >= 0 Then
                Sbnd3 = S01 + SlipV4(V1)         'boundary when V4 = V1
                Sbnd4 = SlipA1(A0, A6)
                Smin = Sbnd4
            Else
                Sbnd3 = S01 + SlipV4(V1 - dV0)         'boundary when V4 = V1
                Sbnd4 = Sbnd3
                Smin = Sbnd3
            End If
            If Slip < Smin Then Slip = Smin 'can't go lower than Smin

            'Calculate V4:
            If Slip >= Sbnd1 Then
                dx45 = (Slip - Sbnd1) * Vmin / (VL - Vmin)
                V4 = Vmin
            ElseIf Slip >= Sbnd2 Then      'increase V4 above Vmin
                b = VL + 0.5F * dVr
                c = (VL - 0.5 * V1) * (V1 + dVr) + 0.5 * VL * (VL + dVr)
                V4 = b - Math.Sqrt(b ^ 2 - c + Ar * (Slip - S01))
            ElseIf Slip >= Sbnd3 Then
                'Find V4 by iteration
                V1r = V1 - dVr
                If A0 >= 0 Then
                    V4 = V1 - dVr * (Slip - Sbnd3) / (Sbnd2 - Sbnd3)
                    slp = S01 + SlipV4(V4)
                    i = 0
                    Flag = 0
                    Do
                        V4previous = V4
                        Sprevious = slp
                        If slp >= Slip Then
                            If Slip > Sbnd3 Then
                                V4 = V1 + (V1 - V4) * (Slip - Sbnd3) / (slp -
Sbnd3)
                            Else
                                V4 = V1
                                Exit Do
                            End If
                        Else
                            V4 = V1r + (V4 - V1r) * (Sbnd2 - Slip) / (Sbnd2 -
slp)
                        End If
                        slp = S01 + SlipV4(V4)
                        If Math.Sign(Slip - slp) + Math.Sign(Slip -
Sprevious) = 0 Then Exit Do
                        i = i + 1
                        If Math.Abs(slp - Slip) < 0.001 Then
                            Flag = 1
                            Exit Do
                        End If
                    Loop

                    If Flag = 0 And Math.Sign(Slip - slp) + Math.Sign(Slip -
Sprevious) = 0 Then
                        Do
                            V4p = V4
                            Sp = slp
                            V4 = ((Slip - Sprevious) * V4 + (slp - Slip) *
V4previous) / (slp - Sprevious)
                            slp = S01 + SlipV4(V4)
                            V4previous = V4p
                            Sprevious = Sp
                        Loop Until Math.Abs(slp - Slip) < 0.001
                    End If
                Else    'A0 < 0
                    V4 = V1r + (dVr - dV0) * (Sbnd2 - Slip) / (Sbnd2 - Sbnd3)
                    slp = S01 + SlipV4(V4)
                    i = 0
                    If Math.Abs(slp - Slip) > 0.001 Then
                        Do
                            If slp >= Slip Then
                                V4 = V4 + (V1 - dV0 - V4) * (slp - Slip) /
(slp - Sbnd3)
                            Else
                                V4 = V1r + (V4 - V1 + dVr) * (Sbnd2 - Slip) /
(Sbnd2 - slp)
                            End If
                            slp = S01 + SlipV4(V4)
                            i = i + 1
                        Loop Until Math.Abs(slp - Slip) < 0.001
                    End If
                End If
            ElseIf Slip >= Sbnd4 Then  'In these cases A0 > 0 and points
1,2,3,4,5 coincide
                'Calculate A1
                A1 = A0 * (Sbnd3 - Slip) / (Sbnd3 - Sbnd4)  'first guess for
A1
                slp = SlipA1(A1, A6)
                i = 0
                Do
                    If slp >= Slip Then
                        A1 = A0 - (A0 - A1) * (Slip - Sbnd4) / (slp - Sbnd4)
                    Else
                        A1 = A1 * (Sbnd3 - Slip) / (Sbnd3 - slp)
                    End If
                    slp = SlipA1(A1, A6)
                    i = i + 1
                    SlipError = Slip - slp
                Loop Until Math.Abs(SlipError) < 0.001
            End If
        End If

        'Having V4 we now can compute A2,V2,V3,A4,A5:
        If Slip >= Sbnd3 Then
            If V1 - V4 >= dVr Then
                A2 = -Ar
            ElseIf V1 - V4 > 0 Then
                A2 = -Math.Sqrt(Jcomfort * (V1 - V4))
                If A0 < 0 And Math.Abs(V1 - V4 - dV0) < 0.000001 Then
                    A2 = A0
                End If
            Else
                A2 = 0
            End If

            If A0 >= 0 Then
                dt12 = -(A2 - A1) / Jcomfort
                V2 = V1 + dt12 * (A2 + A1) / 2
                dx12 = dt12 * (V1 + dt12 * (2 * A1 + A2) / 6)
            Else
                dt12 = -(A2 - A0) / Jcomfort
                V2 = V0 + dt12 * (A2 + A0) / 2
                dx12 = dt12 * (V0 + dt12 * (2 * A0 + A2) / 6)
                dt01 = 0
                dx01 = 0
            End If

            dt34 = -A2 / Jcomfort
            V3 = V4 - dt34 * A2 / 2
            dx34 = dt34 * (V3 + dt34 * A2 / 3)

            If A2 <> 0 Then
                dt23 = (V3 - V2) / A2
            Else
                dt23 = 0
            End If
            dx23 = dt23 * (V2 + dt23 * A2 / 2)

            A4 = 0
            A5 = 0
            V5 = V4
        ElseIf Slip >= Sbnd4 Then    'A0 > 0  in this case
            dt01 = -(A1 - A0) / Jcomfort    'here A1 <= A0, jerk is negative
            V1 = V0 + dt01 * (A0 + A1) / 2
            dx01 = dt01 * (V0 + dt01 * (2 * A0 + A1) / 6)

            'points 1, 2, 3, 4, 5 coincide

            Va = V0 + A0 ^ 2 / 2 / Jcomfort - A1 ^ 2 / Jcomfort

            dt12 = 0
            V2 = V1
            dx12 = 0

            dt34 = 0
            V3 = V1
            dx34 = 0

            dt23 = 0
            dx23 = 0

            A2 = A1
            A3 = A1
            A4 = A1
            A5 = A1
            V4 = V1
            V5 = V1
        End If

        If Slip >= Sbnd3 Then
            V4a = V4
        Else
            V4a = Va
        End If

        DV = VL - V4a
        If DV >= dVr Then
            A6 = Ar
        ElseIf DV > 0 Then
            A6 = Math.Sqrt(Jcomfort * DV)
        Else
            A6 = 0
        End If

        dt45 = dx45 / Vmin

        V5 = V4
        dt56 = (A6 - A5) / Jcomfort
        V6 = V5 + dt56 * (A5 + A6) / 2
        dx56 = dt56 * (V5 + dt56 * (2 * A5 + A6) / 6)

        dt78 = A6 / Jcomfort
        V7 = VL - dt78 * A6 / 2
        dx78 = dt78 * (V7 + dt78 * A6 / 3)

        If A6 > 0 Then
            dt67 = (V7 - V6) / A6
        Else
            dt67 = 0
        End If
        dx67 = dt67 * (V6 + dt67 * A6 / 2)

        'Now compute the times at the eight points
        t1 = dt01
        t2 = t1 + dt12
        t3 = t2 + dt23
        t4 = t3 + dt34
        t5 = t4 + dt45
        t6 = t5 + dt56
```

```
        t7 = t6 + dt67
        t8 = t7 + dt78      'Maneuver time
        Tm = t8

        'Now compute the travel distances at the eight points
        x1 = dx01
        x2 = x1 + dx12
        x3 = x2 + dx23
        x4 = x3 + dx34
        X5 = x4 + dx45
        X6 = X5 + dx56
        X7 = X6 + dx67
        X8 = X7 + dx78      'Maneuver distance
        Dm = X8
    End Sub

    Function SlipV4(ByVal V4a As Double) As Double
        'this routine calculats slip boundaries
        Dim dt14, dt58, DV As Double

        DV = V1 - V4a
        If DV >= dVr Then
            dt14 = DV / Ar + Ar / Jcomfort      'time from point 1 to point 4
        ElseIf DV > 0 Then
            dt14 = 2 * CSng(Math.Sqrt(DV / Jcomfort))
        Else
            dt14 = 0
        End If

        DV = VL - V4a
        If DV >= dVr Then
            dt58 = DV / Ar + Ar / Jcomfort      'time from point 5 to point 8
        ElseIf DV > 0 Then
            dt58 = 2 * CSng(Math.Sqrt(DV / Jcomfort))
        Else
            dt58 = 0
        End If

        SlipV4 = (VL - 0.5F * (V1 + V4a)) * dt14 + 0.5F * DV * dt58
        Return SlipV4
    End Function

    Function SlipA1(ByVal A1a As Double, ByRef A6r As Double) As Double
        'Calculates slip for maneuvers between Sbnd3 and Sbnd4
        'For these cases, points 1,2,3,4,5 coincide.  In all other maneuvers
A1=0.
        Dim Va, dt0a, DV, dta8, Term As Double

        Va = V0 + A0 ^ 2 / 2 / Jcomfort - A1a ^ 2 / Jcomfort
        DV = VL - Va

        If DV >= dVr Then
            dta8 = DV / Ar + Ar / Jcomfort
            A6r = Ar
        Else
            dta8 = 2 * Math.Sqrt(DV / Jcomfort)
            A6r = Math.Sqrt(Jcomfort * DV)
        End If

        dt0a = (2 * A1a - A0) / Jcomfort
        Term = VL - V0 - (A0 ^ 2 - 2 * A0 * A1a - 2 * A1a ^ 2) / 3 / Jcomfort
        SlipA1 = 0.5 * DV * dta8 - dt0a * Term + A1a * (A1a / Jcomfort) ^ 2 /
3
    End Function

    Function SlipBoundary(ByVal V1 As Double, ByVal V4 As Double, ByVal S01
As Double) As Double
        Dim Ar, Vb, T14, T58, Term1, Term2 As Double

        Ar = 0.75 * Acomfort   'reduced maximum acceleration near line speed
        Vb = Ar ^ 2 / Jcomfort

        If V1 - V4 >= Vb Then
            T14 = (V1 - V4) / Ar + Ar / Jcomfort      'time from point 1 to
point 4
        ElseIf V1 - V4 > 0 Then
            T14 = 2 * Math.Sqrt((V1 - V4) / Jcomfort)
        Else
            T14 = 0
        End If
        If VL - V4 >= Vb Then
            T58 = (VL - V4) / Ar + Ar / Jcomfort      'time from point 5 to
point 8
        ElseIf VL - V4 > 0 Then
            T58 = 2 * Math.Sqrt((VL - V4) / Jcomfort)
        Else
            T58 = 0
        End If
        Term1 = 0.5 * (2 * VL - V1 - V4) * T14
        Term2 = 0.5 * (VL - V4) * T58
        SlipBoundary = S01 + Term1 + Term2
    End Function

    Sub State(ByVal Delt, ByVal J, ByVal Ao, ByVal Vo, ByVal Xo)
        Dim Delt2 As Double
        Delt2 = Delt * Delt / 2
        Ac = Ao + J * Delt
        Vc = Vo + Ao * Delt + J * Delt2
        Xc = Xo + Vo * Delt + Ao * Delt2 + J * Delt2 * Delt / 3
    End Sub

    Private Sub lblSpeed_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        V0 = CDbl(txtSpeed.Text)
    End Sub

    Private Sub lblAcceleration_Click(ByVal sender As System.Object, ByVal e
As System.EventArgs)
        A0 = CDbl(txtAcceleration.Text)
    End Sub

    Private Sub lblManeuverNo_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        ManeuverNo = CInt(txtManeuver.Text)
    End Sub

    Private Sub lblCase_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs)
        Select Case ManeuverNo
            Case 0
                Vfinal = V0
            Case 1
                Dstop = CDbl(txtCase.Text)
            Case 2
                Vfinal = CDbl(txtCase.Text)
            Case 3
                Slip = CDbl(txtCase.Text)
        End Select
    End Sub

    Private Sub btnRun_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnRun.Click
        Control()
    End Sub

    Private Sub btnQuit_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnQuit.Click
        Me.Close()
    End Sub

    Private Sub txtManeuver_TextChanged(ByVal sender As System.Object, ByVal
e As System.EventArgs) Handles txtManeuver.TextChanged

    End Sub

    Private Sub txtSpeed_TextChanged(ByVal sender As System.Object, ByVal e
As System.EventArgs) Handles txtSpeed.TextChanged

    End Sub

    Private Sub btnSpeed_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnSpeed.Click
        V0 = CDbl(txtSpeed.Text)
    End Sub

    Private Sub Acceleration_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnAcceleration.Click
        A0 = CDbl(txtAcceleration.Text)
    End Sub

    Private Sub Maneuver_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles BtnMvrNo.Click
        ManeuverNo = CInt(txtManeuver.Text)
    End Sub

    Private Sub Button1_Click(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles btnManeuver.Click
        Select Case ManeuverNo
            Case 0
                Vfinal = V0
            Case 1
```

```
                Dstop = CDbl(txtCase.Text)
            Case 2
                Vfinal = CDbl(txtCase.Text)
            Case 3
                Slip = CDbl(txtCase.Text)
        End Select
    End Sub

    Private Sub txtCase_TextChanged(ByVal sender As System.Object, ByVal e As
System.EventArgs) Handles txtCase.TextChanged

    End Sub

    Private Sub txtAcceleration_TextChanged(ByVal sender As System.Object,
ByVal e As System.EventArgs) Handles txtAcceleration.TextChanged

    End Sub
End Class
```

The Event-Driven Simulation Summary

1. Define a computational time interval dt = say 0.1 sec.
2. The maneuvers.
    a. Change speed
    b. Slip
    c. Stop in given distance
    d. Emergency stop
3. Passenger movement.
    a. Generation at random times with random loading times.
        i. For each I (origin) and j (destination) generate random number $0 < R < 1$.
        ii. If $D_{ij}dt/3600 > R$ introduce a passenger.
        iii. Passenger properties
            1. Passenger ID
            2. Destination j
            3. Loading time = Mean + Variance * ln[R/(1-R)]
            4. Set Mean, Variance, Min, Max times in advance.
            5. Set Mean, Variance, Min, Max masses in advance.
            6. With new R, Passenger mass = Mean + Variance * ln[R/(1-R)]
            7. Status “Waiting”
            8. Arrival time, now t
            9. Wait time, now 0
            10. Trip time, now 0
            11. Increase number waiting by 1.

b. Loading on vehicles.
i. Create array for each station and each berth.
ii. Let a value be number of vehicle present or zero if none.
iii. Check each berth in each passenger station.
iv. If empty vehicle present and available, load passenger;
v. In vehicle array store
1. Passenger number
2. Passenger destination
3. Trip origin station
4. Vehicle gross mass
5. Loading time
6. Departure time
7. Passenger wait time
8. Passenger status now "Riding"
9. Time to go now is the loading time
10. . . .
11.

c. Disembarking.
i. Passenger status "Disembarked"
ii. Gather statistics on trip

4. Commands:
a. Station Zone
i. Switch at station switch point.
1. Determine of vehicle should switch in and if space is available.
2. If so switch and assign berth.
ii. Decelerate to a given berth.
1. Update berth assignment
2. Command deceleration to given berth
iii. Advance in station.
1. For vehicles at rest advance when possible.
iv. Command line speed.
1. Create for each station an array giving the number of the vehicles bypassing the station in the order in which they entered station zone.
2. For vehicle in or assigned to the first berth check vehicles bypassing to determine if a space is available.
3. Check to see if the vehicle ahead is far enough ahead.
4. Command line speed and assign vehicle to position in bypass array.

v. Reset on station exit.

1. Set parameters to new situation
2. Command vehicle to slip if it would violate headway requirement.

b. Merge Zone

i. Slip vehicles to space vehicles at minimum headway.

ii. When slipping a vehicle slip vehicles behind if necessary.

iii. To do this with minimum delay, take into account slip remaining for each vehicle.

iv. Switch.

v. Reset to next link including assignment to a new station array.

c. Diverge Zone

i. Switch at command point

ii. Reset to next link including assignment to a new station array.

d. Change line speed at specific points.

i. If vehicle is to slow down, vehicles behind may have to slow down.

e. Reduce line speed due to high wind and later restore.

f. Call an empty vehicle.

i. Create an array for each station of vehicle commanded to storage.

ii. When there is an empty vehicle in the first birth and there is an occupied vehicle in a waiting position, command that empty vehicle to the nearest storage station and place in array i.

iii. Set a criterion for each station for when to call an empty vehicle.

g. Emergency stop.

5. Calculate each vehicle's x-y coordinates for plotting.
6. Calculate power and energy.
7. Check for negative speeds.
8. Check for headway violations.
9. Up-date all times.
10. Terminate the run when all vehicles have stopped.

Simulation of an Intelligent Transportation Network System

Table of Contents

I. Overview of the Setup of a Network

The first step in a program intended to deploy ITNS is to simulate it as accurately as possible. Only in this way can one determine where the lines should go, how big the stations should be, and how many vehicles will be required.

1.1 Network Setup.

On a map of the network area, draw lines in the direction the guideway is to go. Figure 1 is an example of a network big enough but not too big for practice for the first time the reader develops a simulation.

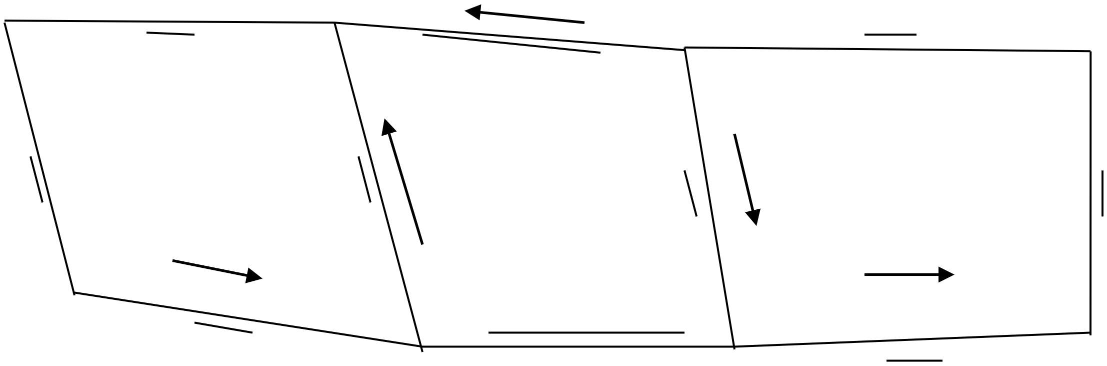


Figure 1. Example test network.

The arrows indicate the direction of flow of vehicles, which could be as shown or the reverse. The flow will be along the lines shown until a change in direction is approached. Then, given the accepted ride comfort in terms of lateral acceleration and rate of change of lateral acceleration (called jerk), there will be curves to make the transitions from one direction to the next. The intersections of the tangents to the curves, called apexes, are the points of intersection of the straight lines in the above figure.

The coordinates of the apexes are measured from the map on which the layout of guideways is made and stored in a convenient set of orthogonal rectangular coordinates, say x, y, z, where typically x could increase to the east, y to the north, and z upward. The location of each apex is defined by these three coordinates. The origin of the coordinates is selected most conventionally in the lower left-hand corner of the map. Number the apexes in any order, usually but not necessarily starting at #0 in the lower left-hand corner and then increasing in the direction of flow, which is not always possible because there will generally be branches such as the two diverging branches shown in the above diagram. Where there are branches, select two apex numbers, one for each guideway leaving or entering the branch. Note that in the above diagram the lower left and upper right branch points are points where the traffic diverges in two directions. These are called "diverge points," whereas the branches in the upper left and the lower right are called "merge points." In addition to numbering the apexes, of which there are 12 in the above diagram, number the merges and diverges, collectively called "branch points," of which there are 4 in the above diagram. It is convenient to number the diverges first. With this system of notation, the network can be expanded in any direction to any extent.

For each apex number, tabulate the number of the next apex ahead, or if there are two apexes ahead, tabulate both of these numbers one in each of two columns. Tabulate also the number of the station ahead of each apex if there is one before the next branch point. Also tabulate the desired speed through each curve at its apex and in the straight section that follows.

Define "link" as the piece of guideway between a pair of line-to-line branch points. In the above diagram there are six links. Any new section of guideway can be added by adding one or more links. For each branch point, tabulate the number of the next branch point ahead or in the case of diverges the next two branch points ahead. Also, tabulate the number of the branch point behind, or in the case of merges the two branch points behind. For each branch point, tabulate the number of the first station on each link ahead and on each link behind. All of this information and more will be needed to specify the exact location of any vehicle.

Next, the ten lines shown in the above diagram roughly parallel to the main guideway indicate the location of off-line stations. At this point number them in any desired order and tabulate the distance from the beginning of each off-line guideway upstream to the nearest apex. The longer off-line guideways illustrated in Figure 1 could represent storage stations.

As mentioned, there must be a curve at each apex. The paper "Curved Guideways" derives the equations for the curves. The differential equation for a curve is integrated to give position coordinates to any accuracy desired. This reduces the subsequent curve calculations to algebraic equations, which vastly simplifies the process of calculating a whole network. Each curve consists of first a section of constant rate of change of lateral acceleration (jerk), then a section of constant lateral acceleration, and finally a section of constant rate of change of acceleration of the opposite sign back to zero lateral acceleration. In making these calculations, the distance from the starting point of the curve to the apex, which is also the distance from the apex to the end point of the curve, is calculated and stored for later use in finding the system coordinates of the starting point of each curve. These equations are derived for a given speed and for comfort values of jerk and acceleration in a set of local coordinates in which the local x-axis is in the direction of motion starting at x = 0, y is transverse to the left for a person facing in the direction of motion and starts at y = 0, and z is upward starting at z = 0. Each point of the curve is then transformed to a set of system coordinates, i.e., x, y coordinates common to the whole system that form a plane parallel to the earth's surface. After the x, y coordinates in the horizontal projection of the curve are calculated; the z-coordinate at each point is calculated using the same form of the curve equations. Having understood how to calculate curves, we can calculate the guideway coordinates step by step along the direction of the curve in steps of size *ds*, where *s* is the arc length along the curve.

The step size *ds* is taken small enough to give an accurate representation of the curve. Even a decade or two ago one had to worry about exceeding the computer's memory if the step size was too small, but since memory has been doubling about every 18 months, or by about a factor of 100 per decade, memory is no longer a concern. Every point along the guideway is defined by a unique value of s, which carries with it a set of x, y, z system coordinates. So the process starts at a point $s = 0$ at the beginning of the first curve, say the one in the lower left-hand corner. Then the process advances by calculating first a curve then the following straight segment. An entire network of any configuration is made up of a series of these curve-straight segments, where the length of the straight segment may be zero, and the curve may have zero length, meaning that there need not be a change in direction. Apexes with zero change in direction are inserted at points where the speed must change.

In the above example network, the calculation of the guideway coordinates could start in the lower left-hand corner and proceed around the periphery all the way to the starting point, then jump to the starting point of the left branch of the lower left hand diverge, continue up to the left-hand branch of the upper left-hand merge point, then jump to the starting point of the upper right-hand diverge, calculate its left branch and finish by calculating the left branch of the lower right-hand merge. This is only one of several possible sequences in which the guideway could be calculated. Any sequence is as good as any other.

Now assume that the entire mainline guideway has been calculated for a series of values of s spaced a distance ds apart, where for each s the x, y, z system coordinates have been recorded. Next we add the calculation of the off-line guideways for each station. We first have derived in advance the equations needed to calculate each off-line guideway in local coordinates in which x is in the direction of the mainline guideway at the start of the transition and $x = 0$ at the start of the transition. As before $y = 0$ at the start of the transition and increases to the left perpendicular to the x axis. Then, for each point along the transition into and out of the off-line station, we apply the above-mentioned transformation equations to calculate the corresponding x, y, z system coordinates.

For each station, we tabulate the number of the station ahead and behind on the same link, the number of the branch point ahead and behind, the number of loading berths in each station, whether the station off-line guideway is to the left or right of the main guideway, and the spacing between the main and bypass guideway. We have calculated and stored the x, y, z system coordinates of the entire guideway at each of a series of values of s spaced ds apart. Next we identify and store the values of s at the merge and diverge points, which information is needed later to identify the locations of the vehicles so that they can be plotted. The process for doing this is straightforward for any engineer with the background needed to carry the process this far.

The next step is to calculate the switch table, i.e., a table of left or right switch commands that enable a vehicle from any diverge point to reach any station in the shortest time. The means of making these calculations is well known from Operations-Research theory. In the above diagram the switch directions from each of the two diverge points are easily picked out, but in a very large network the calculations become quite complex.

The next step in the set-up routines is to load the vehicles onto the network by placing them in passenger and storage stations. Each vehicle carries with it quite a large number of parameters. These include the distance to the branch point ahead, the vehicle's system *x, y, z* coordinates, the number of the station the vehicle is in or approaching, the number of the branch point ahead, the number of the branch point behind, the position of the vehicle's switch (left or right), the number of the berth the vehicle is in or approaching, the vehicle's speed and acceleration, the vehicle's mass, the number of the passenger group aboard if any, the vehicle's destination, the passenger group's loading time taken from a normal distribution, the passenger's mass also taken from a normal distribution, the distance of the next command point ahead to the next branch point ahead, etc. When one begins the process of designing a network simulation program, it is impossible to know all the parameters that will be required. Start with what is obvious, run the program, find as a result of errors missing parameters that must be added, and proceed in this way by trial and error until the program works without error.

The final step in setting up the simulation is to specify the demand from every station to every other station, called the Demand Matrix. We can start by assuming a reasonable demand matrix, which from a series of runs will give us trip times between all station pairs. But for a real problem, we must obtain as accurate an estimate of the peak demand as possible because it affects the network layout, the location and number of stations required, the number of berths required in each passenger station, the number of storage berths required, the headway needed, and the line ot civic speed. The demand depends on the trip time between stations, which can only be accurately determined by running the simulation, thus the process is iterative. The demand also depends on factors such as walk time, wait time, and fare.

The code developed thus far is the "Setup" code. It calculates a set of values that must be stored as constants and arrays that can be loaded into the operating simulation program.

1.2 <u>The Simulation Program</u>

We now know the system coordinates of a series of closely spaced points on the network including the coordinates of closely spaced points that describe each off-line guideway. The points are sufficiently closely spaced that we can calculate intermediate points by linear interpolation. We also know the numbers of the branch points and stations ahead of and behind each branch point and all other information required to establish the connectedness of the network. We have loaded the specified number of vehicles into passenger and storage stations.

The time history of motion of the vehicles will advance in predetermined steps *dt*, called "computational intervals," so required changes will be calculated only at these time intervals. In our simulation, the step size has usually been $dt = 0.5$ sec. The simulation is "event driven," i.e., changes will be calculated as a result of certain events such as arrival of a passenger, loading of a passenger on a vehicle, permitting a passenger group to disembark, permission for a vehicle to leave a station, motion of a vehicle to avoid conflict at a merge point, determining the switch position of a vehicle that reaches a diverge command point, the decision to switch into or past a station when a vehicle reaches a station switch command point, the decision to initiate deceleration into a station to stop at a certain berth when a vehicle reaches a deceleration command point, the decision to advance a vehicle in a station when the berth ahead becomes available, etc.

The first step in the simulation is to generate passengers. Details are given in the next section. Then the passenger group is loaded usually but not always in the forward-most empty vehicle in the origin station. (If there are more passengers arriving from other stations than from the street, forward empty vehicles will need to be released, hence in such cases it is better to load passengers several berths back from the front so that vehicles in the forward berths can be released more quickly.) When the passenger group is loaded, a clock is started at the loading time and reduced by *dt* every computational cycle. When the clock reaches zero the vehicle is ready to be commanded to line speed. The station zone controller (SZC) keeps track of the position and speed of each vehicle in and bypassing the station. For the given acceleration, speed and position of the loaded vehicle, the SZC determines if there is a gap in the station-by-pass guideway of sufficient length and of the correct position that would permit the vehicle to arrive at line speed sufficiently far behind a vehicle ahead and sufficiently far ahead of a vehicle behind to meet the required minimum headway. If this condition is met, the SZC determines if any vehicle ahead on the station bypass guideway is sufficiently far ahead so as to not violate the headway criterion when it arrives at line speed. If this condition is met the vehicle is commanded to line speed. The vehicle follows a profile of speeds and distances calculated in its computer at each of the series of *dt* intervals until it reaches line speed. The calculation of acceleration, speed, and distance for each vehicle is given in the companion paper "Transitions." The events encountered by a vehicle are discussed in the paper "Asynchronous Point Follower."

II. Elements of the System to be Simulated

A computer program that can simulate accurately the motion of vehicles in a network of guideways consists of ten elements, six of which are simulations of system elements and four are code that can operate a real system. The system elements are

- Guideway
- Switch dynamics
- Stations
- Vehicles
- Passengers

- Power and energy

The system software elements are in

- The Station Zone Controller (SZC)
- The Merge Zone Controller (MZC)
- The Diverge Zone Controller (DZC)
- The Empty-Vehicle Movement

2.1 The guideway

1. Guideway Coordinates. The distance from an arbitrarily selected zero point on the guideway along the guideway is called the ARC LENGTH and is denoted by s. The coordinates of the guideway are inputs to the simulation as functions of s, i.e., x(s), y(s), z(s). These coordinates include the coordinates of the station bypass guideways.

2. Branch Points. The line-to-line BP, i.e., the merges and diverges, are numbered, and the program is informed of the numbers of the BP ahead and behind each BP. A setup program calculates and records the two values of s at each BP and specifies which s is continuous through the BP. These values are used to calculate the distances between the BP.

3. Stations. The stations are numbered and the program is informed of the number of the BP ahead and behind each station, the distance of the input diverge point into each station to the BP ahead, the distance to the front edge of the station platform (the forward edge of the first unloading and loading berth), to the output diverge point out of each station, and the number of station berths and staging berths in each station.

4. The Shortest Path. A program must be written to calculate the shortest time between each station pair. This serves two purposes: 1) to calculate for each diverge point the **switch table**, i.e., the switch command (left or right) to each downstream station, and 2) to permit each SZC to look upstream in the most efficient way for the nearest available empty vehicle.

5. Minimum distance between branch points. The wayside element of the switch is a pair of flared switch rails that receive the switch wheels. The length of the flare is determined from a dynamic simulation of the motion of the vehicle through the merge or diverge in the extreme cases of maximum side wind and maximum unbalanced passenger load. The simulation determines the effect of flare length on ride comfort.[11]

[11] A Dynamic Analysis of the Switch Rail Entry Flare.docx
Lateral Dynamics of the ITNS Vehicle,docx

The minimum distance between branch points is determined by the distance traveled during throw of the switch and verification that it has been thrown plus the distance require to make an emergency stop before reaching the flared switch rails. This minimum distance is computed from the formula

$$D_{\min} = V_L t_{swx} + \frac{V_L^2}{2a_e} + D_{flare} + D_{tolerance}$$

in which

V_L = line speed

t_{swx} = time to throw and verify throw of switch

a_e = emergency deceleration rate, usually 0.4g

D_{flare} = length of flared switch rail

$D_{tolerance}$ = tolerance added to reflect worst cases

2.2 Switch Dynamics

The Vehicle Controller (VC) commands a voltage pulse to a rotary solenoid that throws the switch by overcoming switch arm inertia and bi-stability spring torque. The switch is modeled as time delay of 0.5 sec. More detailed modeling of the switch is unnecessary for a system simulation. In a real system, the VC commands the switch to throw and simultaneously commands initiation of an emergency stop in half a second if the VC cannot verify from a proximity sensor that the switch is thrown. The signal from the proximity sensor cancels the command to stop.

2.3 Passengers

The term D_{ij} in the demand matrix represents the number of people per hour traveling from station i to station j. If Δt is the computation interval in seconds, the quantity $D_{ij}\Delta t / 3600 p_g$, where p_g is the average number of persons per group, represents the average number of small groups of people traveling together by choice who wish to board vehicles in the time interval Δt. If this number were one, there would be an average of one group boarding during each Δt. If this number were one tenth, an average of one group every tenth TMI would board. Therefore, generate a random number $0 < RND < 1$ and introduce a passenger group into station i if

$$RND < D_{ij}\Delta t / 3600 p_g.$$ [12]

The value of Δt must be small enough so that the above quantity never exceeds one. This passenger group is assigned three numbers ijk where i is the boarding-station number, j is the loading-berth number, and k indicates the number of the passenger group. The group's mass is picked from a

[12] A simulation was developed to simulate this procedure for an hour, and was found to give close to D_{ij} trips per hour

distribution and assigned to a memory location corresponding to the passenger group. The group's wait time at this point is set to zero. Group ijk is now ready to board a vehicle. Berth j is the forward-most berth having the shortest queue of passengers waiting. The passenger-group waiting time is recorded. When a vehicle enters berth j, stops, unloads, and is empty, it is ready for boarding. When passenger group ijk has reached the first position in its queue, it is caused to board by 1) placing its number, mass, wait time and destination in vehicle's memory slots, 2) setting the corresponding riding time to zero, and 3) removing its number from the queue of waiting passengers. When the vehicle arrives at the destination, stops, and is ready for unloading, the door is opened (simulated by door-opening time), and the passengers egress, indicated by 1) removing the passenger's data from the vehicle's computer by setting the corresponding memory positions to zero, and 2) assigning to one of the passenger's memory locations the string "Disembarked." While waiting at the origin station, this memory location will state "Waiting", and while riding, it will say "Riding." The trip's wait time, ride time, trip length, and average speed are recorded in the system data bank for later analysis.

The loading and unloading time depends on the door opening and closing time and varies according to the agility of the passenger group. Thus we assume a Gaussian distribution of loading and unloading times with a given mean T_{mean} and variance T_{var}. Thus the probability *P* that the loading time is a time T_{load} is

$$P = \tfrac{1}{2} e^{-\left(\frac{T_{load} - T_{mean}}{T_{\mathrm{var}}}\right)^2}$$

which assumes that the probability that T_{load} is T_{mean} is one half. Solving for T_{load} and letting R be a random number between 0 and 1 we get

$$\begin{aligned}
&T_{load} = T_{mean} + T_{\mathrm{var}} \sqrt{\left| \ln \frac{R}{1-R} \right|} \\
&\text{if } T_{load} < T_{\min} \text{ then } T_{load} = T_{\min} \\
&\text{elseif } T_{load} > T_{\max} \text{ then } T_{load} = T_{\max} \\
&T_{load} = T_{load} + T_{door}
\end{aligned}$$

2.4. Power and Energy

Equations for instantaneous power use and motor efficiency permit the electrical input power to the vehicle to be calculated and summed over the computational intervals to obtain energy use. Summing power and energy use over all the vehicles gives the system power requirement for vehicle operations. Air drag is calculated from the formula

$$\begin{aligned}
&AirDrag = AirDragCoeff\left[V + V_{wind}\cos\left(\psi_{vehicle} - \psi_{wind}\right)\right]^2 \\
&AirDragCoeff = \tfrac{1}{2}\rho C_D A_{front}
\end{aligned}$$

in which

V = Vehicle speed

V_{wind} = Assumed constant wind speed

$\psi_{vehicle}$ = Azimuthal direction of guideway at the vehicle

ψ_{wind} = Assumed constant direction from which the wind is coming

ρ = Air density

C_D = Vehicle drag coefficient

A_{front} = Vehicle frontal area

Next the force on the vehicle is calculated from the equation

$$Force = m\left[A + g\left(a + bV + G\right)\right] + AirDrag$$

in which

$m =$ Vehicle gross mass

$A =$ Vehicle acceleration

g = Acceleration of gravity

$a =$ Vehicle dimensionless road resistance coefficient

$b =$ Vehicle road resistance coefficient in units sec/meter

$G =$ Local grade at vehicle

Regenerative braking could capture a portion of the braking energy, i.e. the portion of the energy when the force is negative. The energy saved by regenerative braking is less than the kinetic energy the vehicle has at the moment it starts decelerating. Since there are no intermediate stops, most of the energy required goes into overcoming air drag and road resistance, thus the energy recoverable with regenerative braking is a small fraction of the total. It is calculated separately to show how much energy could be saved if regenerative braking, which adds weight and cost, were used.

Next we calculates the input electrical power and the potential regenerated power to each vehicle in kilowatts from the equation

$$InputPower = \frac{1}{\eta}\frac{Force \times V}{1000}, \quad Force > 0.$$

$$\mathrm{Re}\,genPower = \eta_{regen}\frac{Force \times V}{1000}, \quad Force < 0.$$

in which η = propulsion efficiency, assumed to be 0.55 until detailed calculations with the specific motors can be made; and η_{regen} is the regeneration efficiency, assumed to be 0.5 until detailed calculations can improve on this number.

Finally, the electrical energy used in kW-hr by all vehicles in a run is accumulated by summing the power multiplied by the computation interval *dt* over all vehicles and all computation intervals during a run. In equation form

$$TotalElectricalEnergyUsed = \sum_{TimeIntervals} \sum_{Vehicles} \frac{InputPower \times dt}{3600},$$
$$TotalPotential\,\mathrm{Re}\,genPower = \sum_{TimeIntervals} \sum_{Vehicles} \frac{\mathrm{Re}\,genPower \times dt}{3600}.$$

III. Locating and Moving Vehicles

3.1. The required number of vehicles in a network

The required number of vehicles in a PRT network is given by the formula[13]

$$N_{op} = \frac{D_{peak} l_{trip}}{p_v V_{av}}$$

in which

D_{peak} is the peak-period demand in people per unit of time

l_{trip} is the average trip length

p_v is the average number of people per vehicle, counting empty vehicles

V_{av} is the average speed

3.2. Initial vehicle placement

To accomplish vehicle placement and proper identification, the station zone controller (SZC) has within it the array *staVehicleInBerth*(i, j), which is the number of the vehicle in berth j of station i, 0 if none. This assignment is needed so that the SZC will know where and which vehicles are in its berths, to assign incoming vehicles to the forward-most free berth, and to move vehicles forward in the station when possible.

The vehicle array *Vehicle* (i, j), which is stored in the computer in vehicle i, must be loaded with the correct data for each property j. Among other properties, one of them is the number of the BP ahead and the number of the BP behind vehicle i, if the BP ahead is a merge another property is the leg (0 or 1) vehicle i is on, and another gives the distance vehicle i is behind the BP ahead.

[13] "Calculation of Performance and Fleet Size in Transit Systems," JAT, 16:3(1982)231-252, equation (50).

This information is necessary to determine the unique arc length (s) at the vehicle, and hence from stored values of *x(s), y(s),* and *z(s)* the coordinates of the vehicle, which, in the simulation, can then be plotted. In the real system the coordinates can be compared with GPS coordinates for verification.

The domain of a zone controller is defined in terms of a range of values of arc length *s*. The ZC may maintain in its memory the values of *x(s), y(s)*, and *z(s)* for its range of values of *s*. The local line speed is also maintained in memory.

3.3. Vehicle states

The vehicles can be in any one of three states: rest, constant speed, or maneuvering. If maneuvering, the command values of acceleration (A), speed (V) and position (X) from the start of the maneuver are calculated each time-multiplexing interval. In the system, position is a negative number D_{BP} that goes to zero when the vehicle reaches the branch ahead. The reason for this measure of distance is to make the stored distance ahead of the merge the same for vehicles on the two legs of a merge that are at the same distance from the merge junction. Thus, during the maneuver

$$D_{BP} = D_{BP_0} + X$$

where D_{BP_0} is the negative distance at the start of the maneuver.

If a vehicle is at rest, the acceleration and speed are set to zero, and the position is given in terms of the distance D_{BP}. If a vehicle is moving at constant speed V, distance is given by

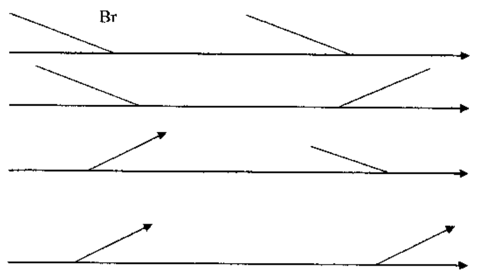
$$D_{BP}(t) = D_{BP}(t - \Delta t) + V\Delta t$$

where *t* is time and Δt is the time-multiplexing interval.

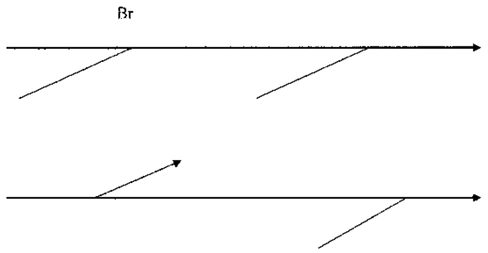
IV. Classification of Network Properties

To plot the position of a vehicle correctly, it is necessary to identify positively the arc length, *s*, ahead of the vehicle. To do this, it is necessary to identify four types of situations that relate one branch point to the branch point ahead. These types and the notation we use are the following:

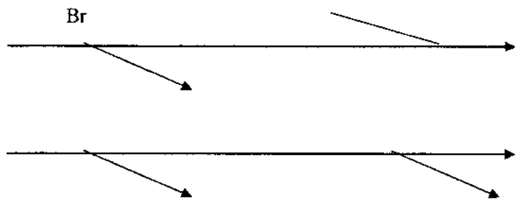
brAheadTypeR(*Brn*) = "Right" if *Brn* is a merge point going to the right leg of a merge or to a diverge, or if *Brn* is a diverge with the right leg going to the right leg of a merge or to a diverge, as shown in the following diagram.



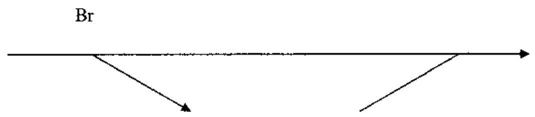
brAheadTypeR(Brn) = “Left” if *Brn* is a merge going to the left leg of a merge, or if *Brn* is a diverge with the right leg going to the left leg of a merge, as shown in the following diagram.



brAheadTypeL(Brn) = “Right” if *Brn* is a diverge with the left leg going either to the right leg of a merge or to a diverge, as shown in the following diagram.



brAheadTypeL(Brn) = “Left” if *Brn* is a diverge with the left leg going to the left leg of a merge, as shown in the following diagram.



The above arrays permit positive identification of the value of arc length, *s*, ahead of the vehicle. All of these quantities are calculated in the setup program.

V. Discussion of the Simulation Program

5.1. Introduction

There are two objectives to the PRT Network Simulation Program:

1) to develop and prove the code needed to operate a real three-dimensioal PRT network of any complexity, and

2) to provide a planning tool needed to simulate and hence design accurately any PRT system.

Both of these objectives are met with the program described here. The purpose of this section is to describe with some repition the program that will take as inputs raw data on a specific network and from it calculate and store in a set of files the information needed to run the simulation program.

A specific application is laid out first in planform, i.e., in the projection onto a horizontal plane, the x-y plain. Such a projection is defined by giving the x-y coordinates of the apexes (intersection of tangents) to the curves, which is practical if one takes into account that curved guideway will be at least 50% more expensive than straight guideway, so one must keep as much guideway as possible straight. To the horizontal projection we specify the height, the z-coordinates, of the apexes of vertical curves.

We use the variable *s* to represent uniquely every point on the network. I call *s* the "arc length" as it is called in analytic geometry. Thus the rectangular coordinates of a point on the guideway are designated as $x(s), y(s), z(s)$. To specify the guideway to the manufacturer, we specify it as a pair of curved lines the width of the main-wheel support angles. Thus, we need also the angular coordinates, which we denote as $azm(s), pitch(s), roll(s)$.

5.2. Apex Data

In the program, apexes (intersections of the tangents to a curve) can relate to curves of any radius down to zero, i.e., no change in direction. Apexes with zero curve radius are defined at branch points where one guideway, usually the main guideway may stay straight while the other branches off in a new direction. Apexes with zero radius are also defined at points along a straight guideway where the line speed is required to change.

The calculations of the coordinates of the guideway are performed in "curve-straight" sets, i.e., the curve around the apex followed by the straight segment after the apex. The program described herein calculates both the curve and the length of this straight segment, which may be of zero length.

The apexes are numbered from a starting point usually but not necessarily sequentially in the direction of flow. The following data related to each apex is needed:

- The number of the next apex in the direction of flow.
- At apexes at branch points, the number of the second next apex.
- The *x, y,* and *z* coordinates at the apex
- The intended line speed in the curve around the apex.
- The intended line speed in the straight segment after the curve.

- The order in which the apexes are to be treated in the direction of flow.

  Giving this information permits branches to be added later without having to change the numbers of the previous apexs.

5.3. Station Data

The stations are numbered generally but not necessarily in the direction of the flow of vehicles. The following data related to each station is needed:

- The number of the closest apex upstream of the station.
- The number of the closest apex downstream of the station.
- The distance of the entry diverge point into the station from the nearest upstream apex.
- The side, right or left, of the main guideway the station is on.
- The station type, passenger or storage.
- The number of loading and unloading berths in a passenger station, or the total number of berths in a storage station.
- The separation distance between the centerline of the main line and the centerline of the bypass line as it passes through the loading and unloading area.
- The number of the branch point ahead of the station (branch point numbers are not the same as apex numbers.)
- The number of the branch point behind the station.
- The number of the station ahead before the next branch point ahead, zero if none.
- The number of the station behind, zero if none before the branch point behind.
- The number of the nearest downstream storage station.
- The station "call criterion" in terms of the number of vehicles that must be present before an additional vehicle is called. By increasing this number, vehicles will be called sooner to reduce the wait time.

5.4. The Demand Matrix, $D(i, j)$, where i is the origin station and j is the destination.

The demand between a pair of stations depends on the average trip time between them. For the first calculation of the demand matrix the ridership analyst must estimate the average trip time. Runs are then made to determine the actual trip time, which is fed back into the demand model for a second iteration. Such analysis is likely to result in recommendations for changing something about the network and the station sizes. Further runs – likely a great many of them – must be performed to obtain satisfactory results.

5.5. Branch Data

The branches, i.e., line-to-line merges and diverges can be numbered in any order, but usually increasing in the direction of flow. By using this notation, any network is an assembly of links, each treated the same. The following data is needed:

- The number of the branch point ahead, i.e., in the direction of flow.
- The number of the branch point behind.
- The number of the first station ahead on the same link, zero if none.
- The number of the first station behind on the same link, zero if none.
- The strings ("R" or "L") brnTypeAhR(i), brnTypeAhR(i), where i is a branch point. The meaning of these terms is given in Section IV. These terms are needed in the simulation program to determine from which side a merge is being approached.
- The switch table brnSwitch(i, j). It gives the direction to switch, "R" or "L", at each diverge branch point. The switch table will be computed in a program that determines the minimum time path from any station to any other station.

5.6. Compute Azimuth

All of the above information except the switch table must be picked off manually from a layout of the specific network to be simulated. To prepare to compute the curves, it is necessary first to compute the azimuth angle at each apex. The azimuth angle is taken as zero in the +x-direction, which is usually east, and is taken to increase in the counterclockwise direction. It is restrained to be less than or equal to 360 deg.

5.7. Compute Direction Change

Having the azimuth angles at all apexes, we next compute the direction change as a vehicle would move from one direction to the next. The direction change will usually be between 0 and 180 deg.

5.8. Compute Curve Properties

We now have all of the information needed to compute the set-up parameters for each of the curve, which are derived in the paper "Curved Guideways." These are the projections of the curves in the horizontal plane. In "Curved Guideways" it is shown that the vertical curves can be superimposed to get the total three-dimensional curves.

5.9. Calculate Straight Section

Knowing the X_a values for each curve, i.e., the distances between the apex and the start or end of the curve, which are calculated in "Curved Guideways", we now have the information needed to calculate the length of each straight section after each curve. As mentioned, the length of the straight section may be zero, and generally is for curves that connect into a merge branch point.

5.10. Calculate Start Coordinates

Next we must calculate the x-y coordinates of the point where eacht curve begins, upstream of its apex. We designate the arc length at the first point of the first curve as $s = 0$.

5.11. Calculate Station Properties

The station properties that must be calculated and stored are the transition length, the length of the straight section in the station-bypass guideway, the total station guideway length, the coordinates of the starting point of the bypass guideway, and the number of berth positions counting waiting positions in each station.

5.12. Calculate Guideway Coordinates

We now have all of the information needed to calculate the x-y guideway coordinates. We do so at discrete steps ds, which we take in the first program as one meter. The end result are the coordinates $x(s), y(s), z(s), azm(s), pitch(s), roll(s)$. To calculate these quantities we use a subroutine called *Curve()* to calculate the coordinates through each curve and includes superelevated turns if elected. A subroutine called *Offline()* calculates each offline guideway. There may be a case in which there is a change in direction of the main guideway in the region of a station, i.e., there may be an apex in the guideway in the region of a station. This requires special handling that is accomplished by a routine called *SpecialOffLine().* These calculations are performed in local coordinates and then a subroutine *Transform()* is used to convert local coordinates into system coordinates. Subroutines *GradeSetUp()* and *Grade()* are used to add the z-coordinate and pitch angle or grade to the guideway.

5.13. Find Jump Points

To position a vehicle on the guideway in the simulation program – as opposed to a real operating system – we need to know the arc length s at the vehicle, which is needed to find its space coordinates, which are needed to plot its position. In paragraph 5.12 we calculated the coordiates at discrete points ds apart. In the simulation program, the position of each vehicle is calculated at each time step as a negative distance behind the branch point ahead. This value goes positve when a vehicle passes a branch point, which indicates that it must be handed over to the next zone controller. To determine the value of arc length s at the vehicle, we must know the arc length value at the branch point ahead. To find it two steps are needed, the first is to find the coordinates and values of s at the points of arc length discontinuity, i.e., the jump points at all branch points including the station entry points. Also, there will be an additional jump point that is neither at a branch point or a station entry point, i.e., at the point $s = 0$. Calculation of all of these jump points is the task of this routine.

5.14 Main-guideway arc length at the Jump Points.

With the coordinates of the jump points calculated, we can and must calculate the values of arc length s on the main guideway at the jump points. These values are needed to calculate the value of s at each vehicle, which as mentioned is needed to calculate the vehicle's coordinates.

5.15. Find the Apex at each Branch Point corresponding to the curve there.

At each branch point there are two apexs, with each corresponding to a different change in direction, often with one having no change in direction. Resolution of this difference is needed in two routines, one in calculation of the merge command point, and the other in determining the speed through the curve used in calculating the distances between branch points. In the later case, the two apexs must give the same curve speed, but in the former the largest change in direction will result in calculating the longest distance to the merge command point, which is the one required for safe merging. The apex corresponding to the greatest change in direction is thus desired and is calculated in this routine.

5.16. Calculate Distances between Branch Points and Branch Command Points

This routine calculates for use in the simulation program the distance from one branch point to the branch point ahead in case of a merge, and from one branch point to the right and left branch points ahead in case of a diverge. It also records the values of arc length at the junction point on the two legs of a merge or diverge, calculates the merge command distance from each merge point, and calculates the switch command distance from each diverge point.

5.17. Calculate Negative Station Distances To the Branch Point ahead

This routine calculates the following quantities;

- The negative distance from the station entry point to the branch point ahead.
- The distance from the station entry point to the front of the station.
- The negative distance from the front of the station to the branch point ahead.
- The negative distance from the middle of the vehicle in the first berth to the branch point ahead.
- The negative distance from the command point after the station exit point to the branch point ahead.
- The negative distance from the station switch-command point to the branch point ahead.
- The negative distance from the station deceleration-command point to the branch point ahead.

5.18. LoadVehicles

This routine gives each vehicle a permanently assigned number and loads it into a passenger or storage station for the start of a simulated run by giving it the following information:

- The number of the branch point ahead.
- The number of the branch point behind.
- The number of the station the vehicle is in.

- The vehicle's switch position corresponding to the side of the main guideway the station bypass guideway is on.
- The vehicle's destination, which now is the station it is in.
- The passenger number, which is now zero.
- The number of the berth the vehicle is in.
- The maneuver command, which is now "None."
- The negative distance of the vehicle to the branch point ahead
- The vehicle's speed, now zero.
- The vehicle's acceleration, now zero.
- The mass of the passengers aboard the vehicle, now zero.
- The distance the vehicle has travelled, now zero.
- The negative distance of the vehicle from the next command point, which is the *ResetOnStationExit* command point.

Simultaneously, this routine gives the station-zone controller two pieces of information;

- The number of the vehicle in each berth.
- The number of the vehicle in each position in the array "*staVehOnStationGdwy*,' which enables the station zone controller to keep track of each vehicle on the station off-line guideway.

5.19. Distance To the Next Station

This routine determines the distance of each station to the station ahead on the same link, or to the station ahead on either branch in the nearest link or the link after that. The routine can be continued recursively until a station is found in either direction, but it will be unusual to have links with no stations. This data will be used to calculate the switch table.

5.20. The number of the next upstream station.

This routine determines the number of the nearest upstream station on the same link or on the next upstream link past a merge or on either of the next upstream stations on a diverge. This data is useful for dispatcihing empty vehicles.

VI. Summary of Setup

1. Layout the best initial estimate of the network on a street map. Because curved guideway is more expensive than straight guideway, use straight lines whenever possible. Locate the stations where you want them. Identify the line-to-line diverges and merges, and number them, diverges first. There must be as many diverges as merges. Establish an x-y reference frame and record the coordinates of each apex.

2. Input following Apex Data for each apex, numbered in any convenient way, usually in the direction of motion, starting with 0. At each branch point there will be an apex number for each direction. Points at which there must be a speed change are given an apex number even though there may be no change in direction.

   1. X coordinate
   2. Y coordinate
   3. Z coordinate
   4. Number of next apex
   5. Number of second next apex if a diverge point ahead
   6. Super elevation angle in the curve around the apex
   7. Speed in the curve
   8. Speed in the straight segment after the curve

3. Perform a graphic check of the network without showing the curves.

4. Input the following Station Data for each station, numbering the stations in any convenient order:

   1. Number of the nearest upstream apex to the station off-line entry point.
   2. Distance of the station off-line entry point to the upstream apex.
   3. Designate passenger station “P”, storage station “S”
   4. Number of loading and unloading berths in a passenger station, total positions in storage station.
   5. Separation between the mainline and the station by-pass or off line.
   6. “L” if the station is on the left side of the main guideway while facing in the direction of motion, “R” if the station is on the right side.
   7. The number of the line-to-line diverge or merge (branch) point ahead of the station.
   8. The number of the line-to-line diverge or merge (branch) point behind the station.
   9. The number of the station ahead on the same link (length of guideway between branch points), 0 if none.
   10. For stations with a merge point ahead designate “L” if the station is on the left of the merge, “R” if on the right. Designate “ ” if a diverge ahead.

5.

6. Calculate the azimuth angle into each apex, assuming azimuth = 0 in the x-direction to the right. From the x-axis the azimuth angle increases in the counterclockwise direction.

7. Calculate the change in direction of the curve passing each apex; + counterclockwise, - clockwise.

8. For the curve at each apex, calculate

    1. Length along the curve (arc length) of the constant jerk region of the curve.
    2. The distance along the curve to the end of its constant-curvature region.
    3. The local coordinates of the center of curvature of the constant-curvature region, where the local x-axis is in the direction the curve begins, y perpendicular to the left.
    4. The radius of curvature of the constant-curvature region.
    5. The local end coordinates of the curve
    6. The distance Xa from the curve's apex (intersection of tangents) to the beginning of

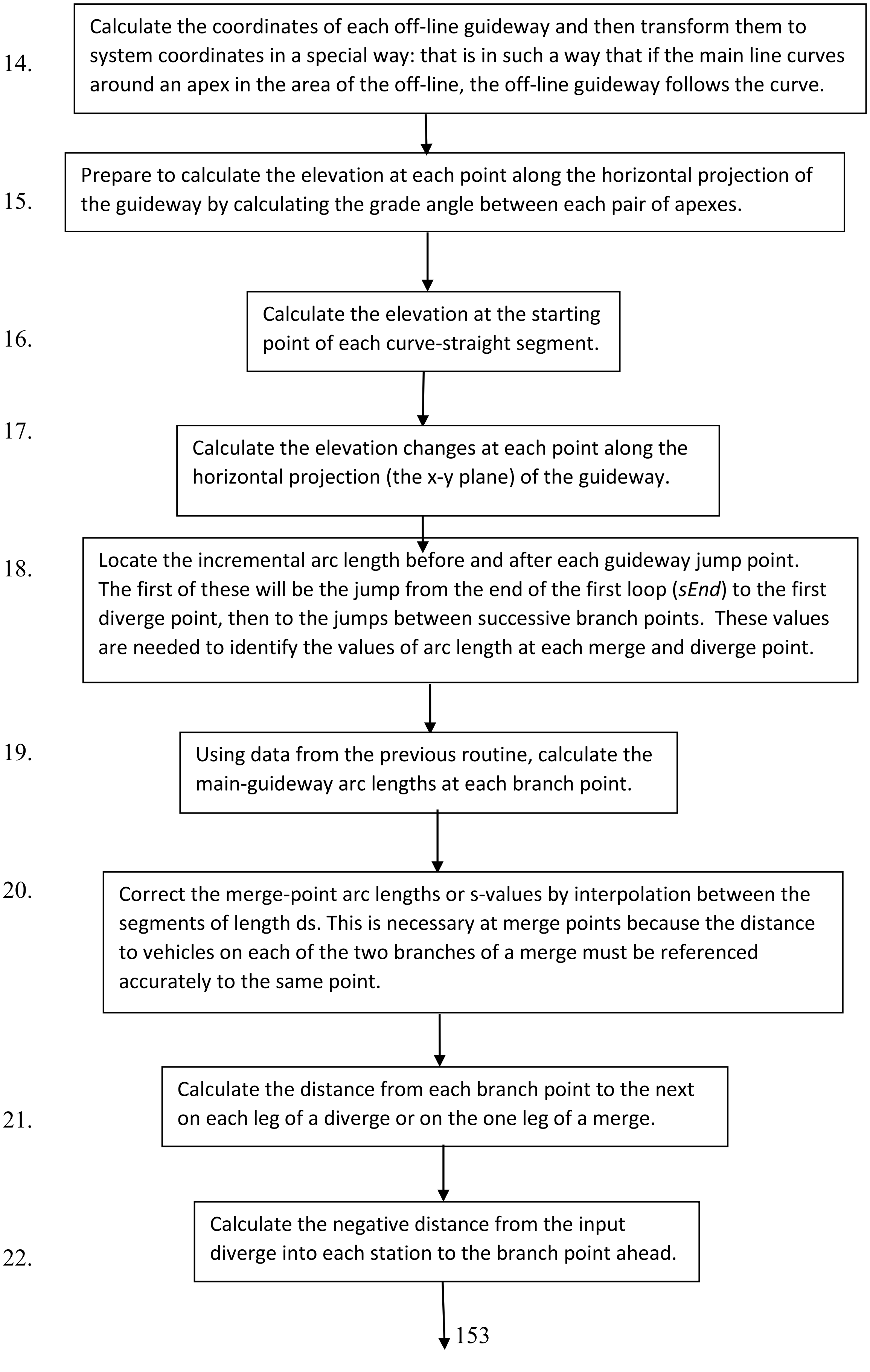
9. Calculate the length of the straight segment of guideway following each curve. The straight segment may be of zero length and must be of zero length for a curve entering merge point. Every network of guideways is made up of segments consisting of a curve followed by a straight segment.

10. Calculate the x-y start coordinates of each curve, and from them identify the coordinates of each line-to-line branch

11. Calculate the x-y start coordinates of the entry off-line to each station.

12. Calculate and record the x-y and Azimuthal coordinates of the mainline guideway in small steps *ds*. This is done apex by apex for the sequence curve-straight for each apex. These calculations make use of a routine that calculates each curve in local coordinates and a second routine that transforms the local curve followed by the straight section in local coordinates into system coordinates.

13. Locate the main-guideway arc length at the entry point of each station. This quantity is needed to transform the local off-line-station coordinates to system coordinates.



14. Calculate the coordinates of each off-line guideway and then transform them to system coordinates in a special way: that is in such a way that if the main line curves around an apex in the area of the off-line, the off-line guideway follows the curve.

15. Prepare to calculate the elevation at each point along the horizontal projection of the guideway by calculating the grade angle between each pair of apexes.

16. Calculate the elevation at the starting point of each curve-straight segment.

17. Calculate the elevation changes at each point along the horizontal projection (the x-y plane) of the guideway.

18. Locate the incremental arc length before and after each guideway jump point. The first of these will be the jump from the end of the first loop (*sEnd*) to the first diverge point, then to the jumps between successive branch points. These values are needed to identify the values of arc length at each merge and diverge point.

19. Using data from the previous routine, calculate the main-guideway arc lengths at each branch point.

20. Correct the merge-point arc lengths or s-values by interpolation between the segments of length ds. This is necessary at merge points because the distance to vehicles on each of the two branches of a merge must be referenced accurately to the same point.

21. Calculate the distance from each branch point to the next on each leg of a diverge or on the one leg of a merge.

22. Calculate the negative distance from the input diverge into each station to the branch point ahead.

23.

Calculate the positions of the following command points:

1. Station switch command point.
2. Station deceleration command point.
3. Merge command point.
4. Diverge command point

Requirements for ITNS Control

1. Communication must be totally secure and not subject to interference from the outside, which means that the computers cannot be connected to any external source that may be disruptive, such as the Internet. This requirement has led to the use of leaky cables within a shielded guideway, a scheme that was tested in the Raytheon test track and was first used in the Boeing AGRT program and described in publicly available papers.

2. Minimum headway. Even though early applications will not require close headways, the design of the control system must take into account the need to achieve fractional second headways safely and reliability as the system expands. Offline stations must be designed to meet expected input and output flows, and the system must be designed to prevent excessive congestion at merge points and destination stations.

3. Safety. A PRT system must provide a level of safety in terms of injuries per billion miles at least as good as a modern rapid rail system, and preferably better—better because the improvements provided by PRT in all areas must be good enough to justify the development cost. To achieve this level of safety, the on-board and wayside computers must be dual duplex. Safety must not depend on one set of computers, i.e., vehicle flow must be monitored by wayside zone-control computers, which requires wayside measurement of position and speed.

4. Ride Comfort. Longitudinal maneuvers must be performed in such a way that International Standards Organization ride comfort standards on acceleration as a function of frequency are met. In maneuvers, longitudinal acceleration must be limited to 0.25 g, lateral acceleration to 0.2 g and jerk to 0.25 g/s in normal operation. The maximum emergency-braking deceleration depends on whether or not passenger constraints are provided. If not, the requirement must be that the passenger must not slide off the seat in an emergency stop. With passenger constraints, twice the normal values are permitted. The control system must not be a factor in causing motion sickness.

5. Changing Conditions. The control system must be able to reduce cruising speed in high winds, restore speed smoothly when the wind dies down, and must be able to cope with any unusual

situation, such as a stopped vehicle, that would require vehicles to slow down or stop away from a station.

6. Dead-Vehicle Detection. It must be possible to detect a dead vehicle on the guideway, however remote that possibility may be. Each vehicle must transmit its speed and position at frequent intervals to a wayside computer—a zone controller. If the zone controller suddenly does not receive the expected signal, it must be programmed to remove the speed signal for all vehicles in that link and transmit this information to the next upstream zone controller. Each vehicle's control system must be configured to command reduction in speed to a creep speed if the zone controller's speed signal is not received. A finite creep speed permits vehicles ahead of the failed vehicle to move safely to the next zone, it reduces anxiety, and with seated passengers is safe. Magnetic detectors must be placed at specified intervals along the guideway to inform the zone controller of passage of a vehicle independent of the vehicle controller. Thus, if a vehicle passes one of these markers and not the next, the location of the dead vehicle is approximately known. Then, because the passengers are seated and can be protected, and the vehicle will be protected by appropriately designed shock-absorbing bumpers, a creeping vehicle can be permitted to advance until it soft engages with the dead vehicle, whereupon the position of the dead vehicle becomes known and the failure strategy can be engaged.

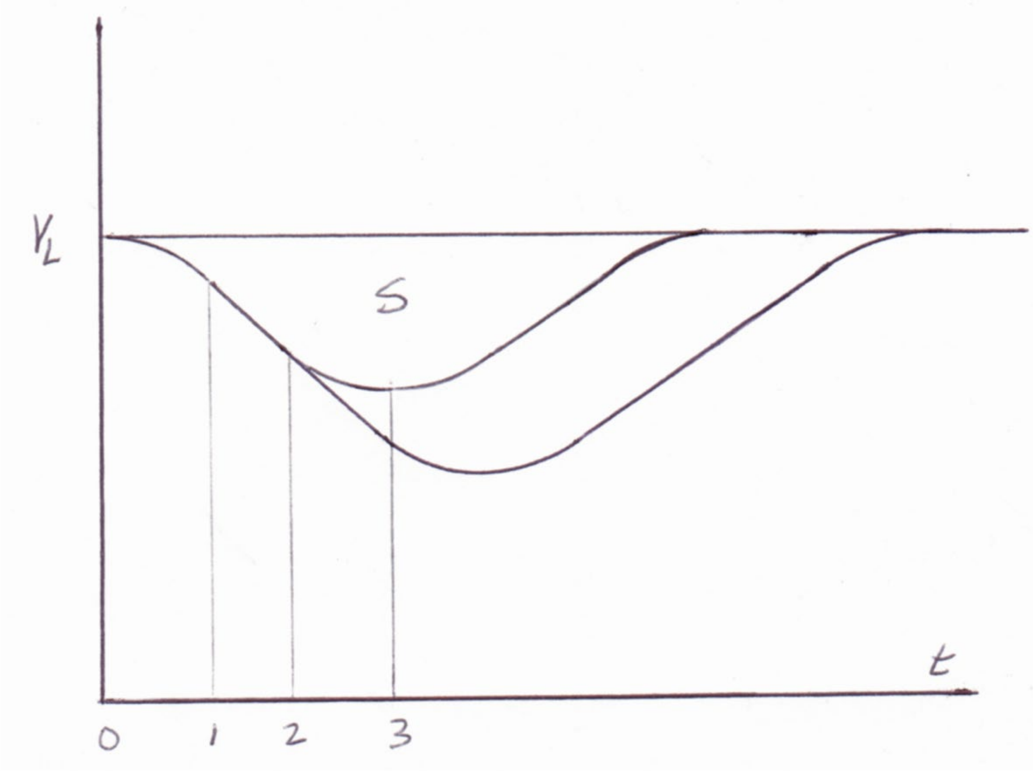
7. Interchange Flexibility. The simplest interchange is a Y. Such an interchange gives the least visual impact at any one point, but requires that vehicles first merge, then diverge, which creates a bottleneck after merging. To obtain maximum possible throughput, two-in, two-out, multilevel interchanges can be used. They permit vehicles to diverge first and then merge. With such interchanges, the input and output capacity of the lines is the same, hence the worst that can happen is that a vehicle may have to be diverted from the direction it would normally go. Thus the control system does not have to be concerned with sending too much traffic along a particular line. If Y-interchanges are used, control actions are not limited to one interchange; however, they are often necessary. Thus, the control system must permit them.

8. Vandalism and Sabotage. A system in which the control functions are distributed and the wayside computers are protected, for example in safe rooms under the stations, will be less susceptible to malicious damage than a system in which a central computer plays an essential role. To minimize the consequences of failures of any kind, distributed control is preferred. The required central-computer functions should be such that the worst that can happen if it fails is that the system will operate less efficiently.

9. Modularity. The control units must be easily exchangeable so that down time is minimized.

10. Expandability. The control system must be designed for easy system expansion.

Distance to Slip



Velocity-Time Diagram, Mirror Symmetry about point 3.

The purpose of this paper is to determine the distance required for a vehicle to slip one or more headway lengths. To do this I make use of the above velocity-time diagram of two slip maneuvers. The jerk from point 0 to point 1 is $-J_c$, from point 1 to point 2 is zero, and from point 2 to point 3 is $+J_c$ in which J_c is the magnitude of the maximum comfort jerk. I assume the reader is familiar with the three equations for the transition from one point to the next at constant jerk, viz.:

1) The time interval between two points at constant jerk is the quantity new acceleration minus old acceleration divided by jerk, or if jerk is zero, the quantity new speed minus old speed divided by acceleration.
2) The new speed is the old speed plus the time interval multiplied by the average acceleration.
3) The distance interval is the time interval multiplied by the quantity old speed plus the time interval multiplied by the quantity twice old acceleration plus new acceleration divided by six.

Equations for the general case.

The time, speed, and distance relationships during the interval from point 0 to point 1 are

$$dt_{01} = \frac{A_1}{-J_c}, \quad V_1 = V_L + dt_{01}\frac{A_1}{2}, \quad dx_{01} = dt_{01}\left(V_L + dt_{01}\frac{A_1}{6}\right), \quad A_1 < 0 \tag{1}$$

In which A_1 is the negative acceleration at point 1, V_L is the line speed, V_1 is the speed at point 1, dt_{01} is the time interval from point 0 to point 1, and dx_{01} is the distance travelled in moving from point 0 to point 1.

The similar relationships in going from point 1 to point 2 at constant deceleration are

$$dt_{12} = \frac{V_2 - V_1}{A_1}, \quad V_2 = V_1 + dt_{12}A_1, \quad dx_{12} = dt_{12}\left(V_1 + dt_{12}\frac{A_1}{2}\right) = \frac{\left(V_1^2 - V_2^2\right)}{-2A_1}, \quad A_1 < 0 \tag{2}$$

In going from point 2 to point 3 we have

$$dt_{23} = \frac{-A_1}{J_c} = dt_{01}, \quad V_3 = V_2 + dt_{23}\frac{A_1}{2}, \quad dx_{23} = dt_{23}\left(V_2 + dt_{23}\frac{A_1}{3}\right) = dt_{23}\left(V_3 - dt_{01}\frac{A_1}{6}\right) \tag{3}$$

The usual case $A_1 = -A_c$

If $A_1 = -A_c$, where A_c is the maximum comfort value of acceleration, we have

$$dt_{01} = \frac{A_c}{J_c}, \quad V_1 = V_L - \frac{A_c^2}{2J_c}, \quad dx_{01} = \frac{A_c}{J_c}\left(V_L - \frac{A_c^2}{6J_c}\right)$$

$$dt_{12} = \frac{V_1 - V_2}{A_c} = \frac{V_L - V_3}{A_c} - \frac{A_c}{J_c}, \quad dx_{12} = \left(\frac{V_1^2 - V_2^3}{2A_c}\right) = \frac{\left(V_L + V_3\right)}{2}\left(\frac{V_L - V_3}{A_c} - \frac{A_c}{J_c}\right)$$

$$dt_{23} = dt_{01}, \quad V_3 = V_2 - \frac{A_c^2}{2J_c}, \quad dx_{23} = \frac{A_c}{J_c}\left(V_3 + \frac{A_c^2}{6J_c}\right)$$

(4)

$$dt_{03} = dt_{01} + dt_{12} + dt_{23} = \frac{V_L - V_3}{A_c} + \frac{A_c}{J_c}$$

$$dx_{03} = dx_{01} + dx_{12} + dx_{23} = \frac{\left(V_L + V_3\right)}{2}\left(2\frac{A_c}{J_c} + \frac{V_L - V_3}{A_c} - \frac{A_c}{J_c}\right) \tag{5}$$

$$= \left(\frac{V_L + V_3}{2}\right)dt_{03}$$

Let S = Slip, $T_m = 2dt_{03}$ = maneuver time, and $D_m = 2dx_{03}$ = maneuver distance. Then

$$
\begin{aligned}
S &= V_L T_m - D_m = T_m\left[V_L - \left(\frac{V_L+V_3}{2}\right)\right] = \left(V_L - V_3\right)\frac{T_m}{2} \\
&= \left(V_L - V_3\right)\left(\frac{V_L - V_3}{A_c} + \frac{A_c}{J_c}\right), \quad \text{let } x = \frac{V_L - V_3}{A_c} \\
\text{Then } & x^2 + \frac{A_c}{J_c}x - \frac{S}{A_c} = 0, \quad x = \frac{A_c}{2J_c}\left(\sqrt{1 + \frac{4S/A_c}{\left(A_c/J_c\right)^2}} - 1\right) \\
V_3 &= V_L - \frac{A_c^2}{2J_c}\left(\sqrt{1 + \frac{4S/A_c}{\left(A_c/J_c\right)^2}} - 1\right)
\end{aligned}
\qquad (6)
$$

The smallest value of S for which equations (6) apply occurs when $V_2 = V_1$. Then, from equations (4),

$$\frac{V_L - V_3}{A_c} = \frac{A_c}{J_c}$$

Then, from the second line of equations (6)

$$S = S_{\min} = S_1 = 2A_c\left(\frac{A_c}{J_c}\right)^2 \qquad (7)$$

We usually assume $A_c = 0.25$g, $J_c = 0.25$ g/s. Then $S_{\min} = 4.9$ m and $V_L - V_3 = 0.25$g $= 2.45$ m/s. The distance travelled in the headway time T_h is $V_L T_h$. We want to consider slipping one headway distance. Then, if we set $V_L T_h = S_{\min}$ the corresponding speed is $V_L = 4.9\text{m}/\text{T}_h$.

<u>Case for small *S*.</u>

For small headways, we will have cases in which $S < S_{\min}$. In such cases set $dt_{12} = 0$ in equations (1) and (3) to get

$$
\begin{aligned}
& dt_{01} = \frac{A_1}{-J_c}, \quad V_1 = V_L - \frac{A_1^2}{2J_c}, \quad dx_{01} = dt_{01}\left(V_L - \frac{A_1^2}{6J_c}\right), \quad A_1 < 0 \\
& dt_{23} = \frac{-A_1}{J_c}, \quad V_3 = V_1 - \frac{A_1^2}{2J_c}, \quad dx_{23} = dt_{01}\left(V_1 - \frac{A_1^2}{3J_c}\right) = dt_{01}\left(V_L - \frac{5A_1^2}{6J_c}\right) \\
& T_m = -4\frac{A_1}{J_c}, \quad D_m = -2\frac{A_1}{J_c}\left(2V_L - \frac{A_1^2}{J_c}\right) \\
& S = V_L T_m - D_m = 2\frac{\left(-A_1\right)^3}{J_c^2}, \quad A_1 = -\left(\frac{J_c^2 S}{2}\right)^{1/3}
\end{aligned}
\qquad (8)
$$

Case for Large S.

We will need to set a minimum speed $V_3 = V_{\min}$, which will be reached for large values of slip. In this case

$$S = S_2 = \left(V_L - V_{\min}\right)\left(\frac{V_L - V_{\min}}{A_c} + \frac{A_c}{J_c}\right) \tag{9}$$

If $S > S_2$ we must add a section of time dt_{34} between the descending and ascending speed regions, in which

$$dt_{34} = \frac{S - S_2}{V_{\min}}. \tag{10}$$

Setup Code

The given parameters are

$$A_c,\quad J_c,\quad V_L,\quad V_{\min}$$

Then calculate

$$S_1 = 2A_c\left(\frac{A_c}{J_c}\right)^2,\quad S_2 = \left(V_L - V_{\min}\right)\left(\frac{V_L - V_{\min}}{A_c} + \frac{A_c}{J_c}\right).$$

Given S then

If $S \le S_1$ then

$$A_1 = -\left(\frac{J_c^2 S}{2}\right)^{1/3},\quad dV = \frac{A_1^2}{2J_c},\quad V_1 = V_L - dV,\quad V_3 = V_1 - dV$$

$$dt_{01} = -\frac{A_1}{J_c} = \left(\frac{S}{2J_c}\right)^{1/3},\quad dt_{12} = 0,\quad dt_{23} = dt_{01},\quad dt_{34} = 0,\quad dt_{45} = dt_{01},\quad dt_{56} = 0,\quad dt_{67} = dt_{01}$$

$$dx_{01} = dt_{01}\left(V_L + dt_{01}\frac{A_1}{6}\right),\quad dx_{12} = 0,\quad dx_{23} = dt_{01}\left(V_1 + dt_{01}\frac{A_1}{3}\right) = dt_{01}\left(V_L - \frac{5}{6}\frac{A_1^2}{J_c}\right),\quad dx_{34} = 0$$

$$dx_{45} = dx_{23},\quad dx_{56} = 0,\quad dx_{67} = dx_{01}$$

$$D_m = 2\left(dx_{01} + dx_{23}\right) = 2dt_{01}\left(2V_L - \frac{A_1^2}{J_c}\right) = \left(\frac{4S}{J_c}\right)^{1/3}\left[2V_L - \left(\frac{J_c^{1/2}S}{2}\right)^{2/3}\right]$$

```
else
    If $S \le S_2$ then
        $V_3 = V_L - \frac{A_c^2}{2J_c}\left(\sqrt{1+\frac{4S/A_c}{\left(A_c/J_c\right)^2}}-1\right), \quad dt_{34}=0, \quad dx_{34}=0$
    else
        $V_3 = V_{\min}, \quad dt_{34} = \left(S-S_2\right)/\left(V_L - V_{\min}\right), \quad dx_{34} = V_{\min} dt_{34}.$
    end if
    $dt_{01} = \frac{A_c}{J_c}, \quad V_1 = V_L - dt_{01}A_c/2, \quad dx_{01} = dt_{01}\left(V_L - dt_{01}A_c/6\right)$
    $dt_{23} = dt_{01}, \quad V_2 = V_3 + dt_{23}A_c/2, \quad dx_{23} = dt_{23}\left(V_2 - dt_{23}A_c/3\right) = dt_{01}\left(V_3 + dt_{01}A_c/6\right)$
    $dt_{12} = \frac{V_1 - V_2}{A_c}, \quad dx_{12} = dt_{12}\left(V_1 - dt_{12}A_c/2\right) = \frac{V_1^2 - V_2^2}{2A_c}$
    $dt_{45} = dt_{23}, \quad dt_{56} = dt_{12}, \quad dt_{67} = dt_{01}$
    $dx_{45} = dx_{23}, \quad dx_{56} = dx_{12}, \quad dx_{67} = dx_{01}$
    $dt_{03} = 2dt_{01} + \frac{V_L - V_3}{A_c} - dt_{01} = \frac{V_L - V_3}{A_c} + \frac{A_c}{J_c}$
    $dx_{03} = dt_{01}\left(V_L + V_3\right) + \frac{1}{2A_c}\left[\left(V_L^2 - V_L dt_{01}A_c\right) - \left(V_3^2 + V_3 dt_{01}A_c\right)\right]$
    $\quad = \frac{\left(V_L + V_3\right)}{2}\left(dt_{01} + \frac{V_L - V_3}{A_c}\right) = \frac{\left(V_L + V_3\right)}{2} dt_{03}$
    $D_m = \left(V_L + V_3\right)\left(\frac{V_L - V_3}{A_c} + \frac{A_c}{J_c}\right) + \left\langle S - S_2 \right\rangle \left(\frac{V_m}{V_L - V_m}\right), \quad \left\langle e \right\rangle = e \text{ if } e > 0 \text{ else } 0.$
end if
$t_1 = dt_{01}$           $x_1 = dx_{01}$
$t_2 = t_1 + dt_{12}$     $x_2 = x_1 + dx_{12}$
$t_3 = t_2 + dt_{23}$     $x_3 = x_2 + dx_{23}$
$t_4 = t_3 + dt_{34}$     $x_4 = x_3 + dx_{34}$
$t_5 = t_4 + dt_{45}$     $x_5 = x_4 + dx_{45}$
$t_6 = t_5 + dt_{56}$     $x_6 = x_5 + dx_{56}$
$t_7 = t_6 + dt_{67}$     $x_7 = x_6 + dx_{67}$
```

The run code is now

Initial conditions: $t = 0, \quad x = 0, \quad V = V_L, \quad A = 0$

```
$if\ t < t_1$
   $Jerk = -J_c$
$elseif\ t < t_2$
   $Jerk = 0$
$elseif\ t < t_3$
   $Jerk = J_c$
$elseif\ t < t_4$
   $Jerk = 0$
$elseif\ t < t_5$
   $Jerk = J_c$
$elseif\ t < t_6$
   $Jerk = 0$
$elseif\ t < t_7$
   $Jerk = -J_c$
$else$
   $Jerk = 0$
$endif$
```

$$
\begin{aligned}
&x = x + Vdt + A \cdot \frac{dt^2}{2} + Jerk \cdot \frac{dt^3}{6} \\
&V = V + Adt + Jerk \cdot \frac{dt^2}{2} \\
&A = A + Jerk \cdot dt \\
&t = t + dt
\end{aligned}
$$

Distance Merge Point to Clearance Point, Dm-S

				Slip	V3	V3,used	Dm,base	Dm	Dm-Sprev
				m	m/s	m/s	m	m	m
g=	9.807	m/s^2	1	5.0	7.516	7.516	35.262	35.262	35.262
Ac=	2.452	m/s^2	2	10.0	6.125	6.125	41.612	41.612	36.612
Jc=	2.452	m/s^3	3	15.0	5.039	5.039	45.471	45.471	35.471

tJ=	1	s	4	20.0	4.117	4.500	47.029	48.797	33.797
VL=	10	m/s	5	25.0	3.302	4.500	47.029	52.888	32.888
Th=	0.5	s	6	30.0	2.563	4.500	47.029	56.979	31.979
Vmin=	4.500	m/s	7	35.0	1.882	4.500	47.029	61.070	31.070
S1=	4.903	m	8	40.0	1.247	4.500	47.029	65.161	30.161
S2=	17.839	m	9	45.0	0.651	4.500	47.029	69.252	29.252
			10	50.0	0.086	4.500	47.029	73.343	28.343
			11	55.0	-0.451	4.500	47.029	77.434	27.434
			12	60.0	-0.964	4.500	47.029	81.525	26.525
			13	65.0	-1.457	4.500	47.029	85.616	25.616
			14	70.0	-1.932	4.500	47.029	89.706	24.706
			15	75.0	-2.390	4.500	47.029	93.797	23.797

Conclusion

The maximum maneuver distance for a slip maneuver beyond the merge command point is produced by the vehicle one headway interval behind the merge command point. This distance is

$$V_3 = V_L - \frac{A_c^2}{2J_c}\left(\sqrt{1+\frac{4S/A_c}{\left(A_c/J_c\right)^2}}-1\right) \text{ where } S = 2V_L T_h$$

$$\text{if } V_3 < V_{\min} \text{ then } V_3 = V_{\min}$$

$$D_m = \left(V_L + V_3\right)\left(\frac{V_L - V_3}{A_c} + \frac{A_c}{J_c}\right) + \left\langle S - S_2 \right\rangle\left(\frac{V_m}{V_L - V_m}\right)$$

$$\text{Minimum Merge Length} = D_m - V_L T_h + \text{ Clearance Length}$$

Computer Program

```
'This routine SLIPHDWY.BAS calculates maneuvers for slipping N headway
distances.
'Units are MKS
g = 9.80665
Ac = .25 * g
Jc = .25 * g
tJ = Ac / Jc
VL = 9
Vmin = .4 * VL
Th = .5        'minimum headway
S1 = 2 * Ac * tJ ^ 2
```

```
dV = VL - Vmin
S2 = dV * (dV / Ac + tJ)

dt = .0002        'computation interval

SCREEN 9
COLOR 7, 8
T0 = 10
Y0 = 250
ScaleT = 15
ScaleX = 2
ScaleV = 10
ScaleA = 10
LINE (T0, Y0)-(640, Y0)
LINE (T0, Y0)-(T0, 0)
LINE (T0, Y0 - ScaleV * VL)-(640, Y0 - ScaleV * VL)

PRINT " i     S      Dm     Dm - S   Dm / S"
FOR i = 1 TO 20
    'Initial conditions
    t = 0
    x = 0
    V = VL
    A = 0
    S = VL * Th * i
    IF S <= S1 THEN
       A1 = -(Jc ^ 2 * S / 2) ^ (1 / 3)
       dV = .5 * A1 ^ 2 / Jc
       V1 = VL - dV
       V3 = V1 - dV
       dt01 = -A1 / Jc
       dt12 = 0
       dt23 = dt01
       dt34 = 0
       dt45 = dt23
       dt56 = 0
       dt67 = dt01

       dx01 = dt01 * (VL + dt01 * A1 / 6)
       dx12 = 0
       dx23 = dt23 * (V1 + dt23 * A1 / 3)
       dx34 = 0
       dx45 = dx23
       dx56 = 0
       dx67 = dx01
    ELSE
       IF S <= S2 THEN
          V3 = VL - .5 * tJ * Ac * (SQR(1 + (4 * S / Ac) / tJ ^ 2) - 1)
          dt34 = 0
          dx34 = 0
       ELSE
          V3 = Vmin
          dx34 = S - S2
          dt34 = dx34 / Vmin
       END IF
       dt01 = tJ
       V1 = VL - dt01 * Ac / 2
```

```
   dx01 = dt01 * (VL - dt01 * Ac / 6)
   dt23 = dt01
   V2 = V3 + dt23 * Ac / 2
   dx23 = dt23 * (V2 - dt23 * Ac / 3)
   dt12 = (V1 - V2) / Ac
   dx12 = dt12 * (V1 - dt12 * Ac / 2)

   dt45 = dt23
   dt56 = dt12
   dt67 = dt01

   dx45 = dx23
   dx56 = dx12
   dx67 = dx01
END IF

Tm = 2 * (dt10 + dt12 + dt23) + dt34
Dm = 2 * (dx01 + dx12 + dx23) + dx34

t1 = dt01
t2 = t1 + dt12
t3 = t2 + dt23
t4 = t3 + dt34
t5 = t4 + dt45
t6 = t5 + dt56
t7 = t6 + dt67
x1 = dx01
x2 = x1 + dx12
x3 = x2 + dx23
x4 = x3 + dx34
x5 = x4 + dx45
x6 = x5 + dx56
x7 = x6 + dx67

DO
   IF t < t1 THEN
      Jerk = -Jc
   ELSEIF t < t2 THEN
      Jerk = 0
   ELSEIF t < t3 THEN
      Jerk = Jc
   ELSEIF t < t4 THEN
      Jerk = 0
   ELSEIF t < t5 THEN
      Jerk = Jc
   ELSEIF t < t6 THEN
      Jerk = 0
   ELSEIF t < t7 THEN
      Jerk = -Jc
   ELSE
      Jerk = 0
   END IF


   x = x + V * dt + A * dt ^ 2 / 2 + Jerk * dt ^ 3 / 6
   V = V + A * dt + Jerk * dt ^ 2 / 2
   A = A + Jerk * dt
```

```
       t = t + dt

       PSET (T0 + ScaleT * t, Y0 - ScaleX * x), 10
       PSET (T0 + ScaleT * t, Y0 - ScaleV * V), 11
       PSET (T0 + ScaleT * t, Y0 - ScaleA * A), 12
       PSET (T0 + ScaleT * t, Y0 - ScaleA * Jerk), 13
    LOOP UNTIL t > t7 + 1
    PRINT i;
    PRINT USING "####.###"; S; Dm; Dm - S; Dm / S
NEXT i
```

Potential Headway Violation upon Decelerating into a Station

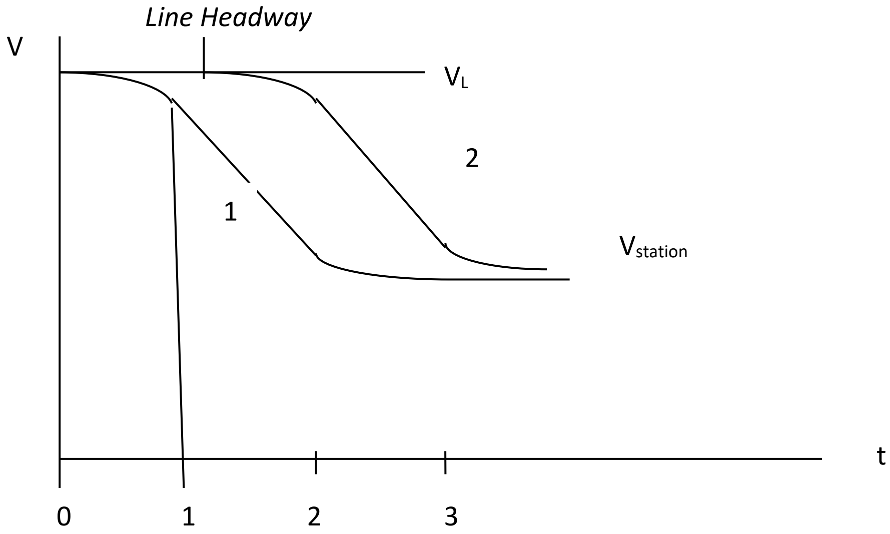


Figure 1. The velocity profiles of a pair of vehicles entering a station.

Consider a vehicle #1 decelerating into a station to station speed V_{sta}, followed by a vehicle #2 a time *Line Headway* behind undergoing the same maneuver. Let the position of vehicle #1 at time zero be $x(0) = 0$. The times, accelerations, speeds, and positions of vehicle #1 at the points 1, 2, 3 in Figure 1[14] are as follows:

[14] For the methodology, see the internal paper "Speed and Position vs. Time"

$$\begin{aligned}
&dt_{01} = \frac{A_c}{J_c}, \quad V_1 = V_L - dt_{01}\frac{A_c}{2}, \quad dx_{01} = dt_{01}\left(V_L - dt_{01}\frac{A_c}{6}\right) \\
&dt_{23} = \frac{A_c}{J_c}, \quad V_2 = V_{sta} + dt_{23}\frac{A_c}{2}, \; dx_{23} = dt_{23}\left(V_2 - dt_{23}\frac{A_c}{3}\right) \\
&dt_{12} = \frac{V_1 - V_2}{A_c}, \qquad dx_{12} = dt_{12}\left(V_1 - dt_{12}\frac{A_c}{2}\right) \\
&t_1 = dt_{01}, \quad t_2 = t_1 + dt_{12}, \quad t_3 = t_2 + dt_{23} \\
&x_1 = dx_{01}, \quad x_2 = x_1 + dx_{12}, \quad x_3 = x_2 + dx_{23}
\end{aligned} \tag{1}$$

From equations (1) we find

$$dt_{03} = dt_{01} + dt_{23} + dt_{12} = 2\frac{A_c}{J_c} + \frac{1}{A_c}\left(V_L - \frac{A_c^2}{2J_c} - V_{sta} - \frac{A_c^2}{2J_c}\right) = \frac{V_L - V_{sta}}{A_c} + \frac{A_c}{J_c} \tag{2}$$

Thus, the maneuver time from line speed to station speed is

$$T_m = \frac{V_L - V_{sta}}{A_c} + \frac{A_c}{J_c} \tag{3}$$

From equations (1) we also find

$$\begin{aligned}
dx_{03} &= dx_{01} + dx_{23} + dx_{12} = \frac{A_c}{J_c}\left(V_L + V_{sta} + \frac{A_c^2}{2J_c} - \frac{A_c^2}{6J_c} - \frac{A_c^2}{3J_c}\right) + \left(\frac{V_1 - V_2}{2A_c}\right)\left(V_1 + V_2\right) \\
&= \frac{A_c}{J_c}\left(V_L + V_{sta}\right) + \frac{1}{2A_c}\left(V_1 - V_2\right)\left(V_1 + V_2\right) = \frac{A_c}{J_c}\left(V_L + V_{sta}\right) + \frac{1}{2A_c}\left(V_L - V_{sta} - \frac{A_c^2}{J_c}\right)\left(V_L + V_{sta}\right) \\
&= \frac{\left(V_L + V_{sta}\right)}{2}\left(\frac{V_L - V_{sta}}{A_c} + \frac{A_c}{J_c}\right) = \frac{\left(V_L + V_{sta}\right)}{2} dt_{03}
\end{aligned} \tag{4}$$

Thus, the distance traveled from line speed to station speed is

$$D_m = \frac{\left(V_L + V_{sta}\right)}{2} T_m \tag{5}$$

Using the above canonical formulation, the acceleration, speed, and position of vehicle 1 at any value of t are as follows:

$$0 \le t \le t_1: \quad \Delta t = t, \quad A = -J_c \Delta t, \quad V = V_L + \Delta t \frac{A}{2}, \quad x = \Delta t\left(V_L + \Delta t \frac{A}{6}\right)$$

$$t_1 \le t \le t_2: \quad \Delta t = t - t_1, \quad A = -A_c, \quad V = V_1 + \Delta t A, \quad x = x_1 + \Delta t\left(V_1 + \Delta t \frac{A}{2}\right)$$

$$t_2 \le t \le t_3: \quad \Delta t = t - t_2, \quad A = -A_c + J_c \Delta t, \quad V = V_2 + \Delta t \frac{(-A_c + A)}{2}, \quad x = x_2 + \Delta t\left(V_2 + \Delta t \frac{A}{3}\right)$$

(6)

For vehicle #2 up to time $t = LineHeadway$ the speed stays constant at V_L and the distance traveled is

$x = V_L t$. For $t > LineHeadway$ we can obtain the acceleration, speed, and position as functions of time by making the following substitutions in equations (5): $t \to t - LineHeadway$

$$T_h = LineHeadway$$

$$0 \le t \le T_h: \quad A = 0, \quad V = V_L, \quad x = V_L(t - T_h)$$

$$T_h \le t \le t_1 + T_h: \quad \Delta t = t - T_h, \quad A = -J_c \Delta t, \quad V = V_L + \Delta t \frac{A}{2}, \quad x = \Delta t\left(V_L + \Delta t \frac{A}{6}\right)$$

$$T_h + t_1 \le t \le T_h + t_2: \quad \Delta t = t - t_1, \quad A = -A_c, \quad V = V_1 + \Delta t A, \quad x = x_1 + \Delta t\left(V_1 + \Delta t \frac{A}{2}\right)$$

$$T_h + t_2 \le t \le T_h + t_3: \quad \Delta t = t - t_2, \quad A = -A_c + J_c \Delta t, \quad V = V_2 + \Delta t \frac{(-A_c + A)}{2}, \quad x = x_2 + \Delta t\left(V_2 + \Delta t \frac{A}{3}\right)$$

(7)

The Minimum Headway

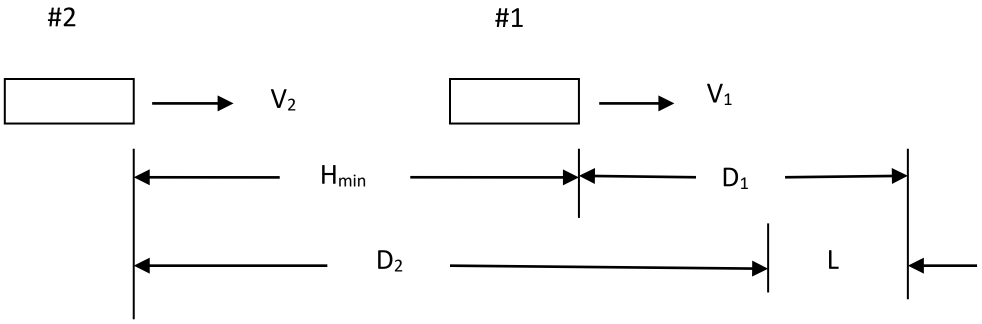


Figure 2. A pair of vehicles moving to the right.

Assume vehicle #1 stops due to a failure at deceleration A_f and jerk J_f. From equation (5), the stopping distance of vehicle #1 is

$$D_1 = \frac{V_1}{2}\left(\frac{V_1}{A_f} + \frac{A_f}{J_f}\right) \tag{8}$$

After a control time delay t_c, vehicle #2 stops at the emergency deceleration rate A_e and emergency jerk J_e. Its stopping distance is therefore

$$D_2 = V_2 t_c + \frac{V_2}{2}\left(\frac{V_2}{A_e} + \frac{A_e}{J_e}\right) \tag{9}$$

Assuming the length of each of the two vehicles is L, the minimum allowable separation between them is

$$H_{\min} = L + D_2 - D_1 \tag{10}$$

The minimum permissible time headway is therefore

$$MinHeadway = \frac{H_{\min}}{V_2}$$

(11)

A program to calculate the acceleration, speed, positions profiles and the minimum headway is given in the Appendix. Some results are given in Figures 3 and 4.

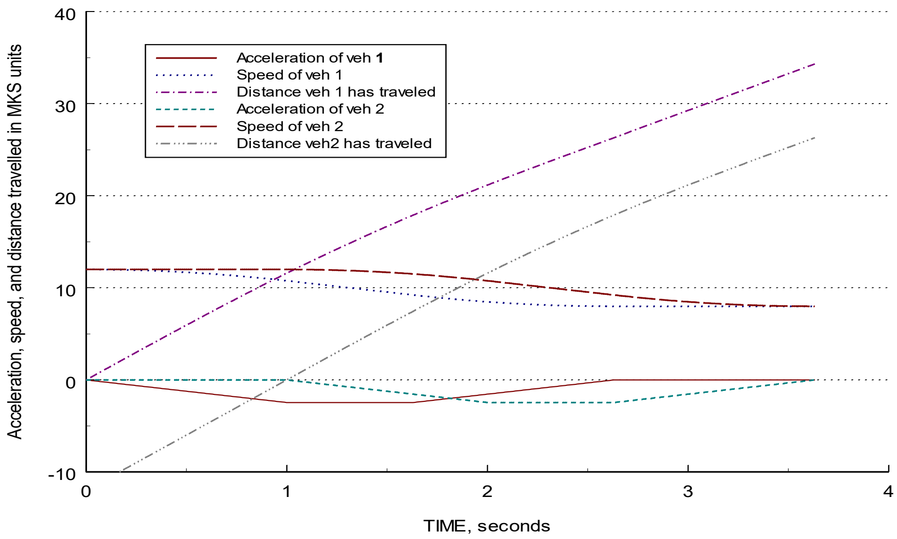


Figure 3. Kinematics of motion of a pair of vehicles decelerating to station speed.

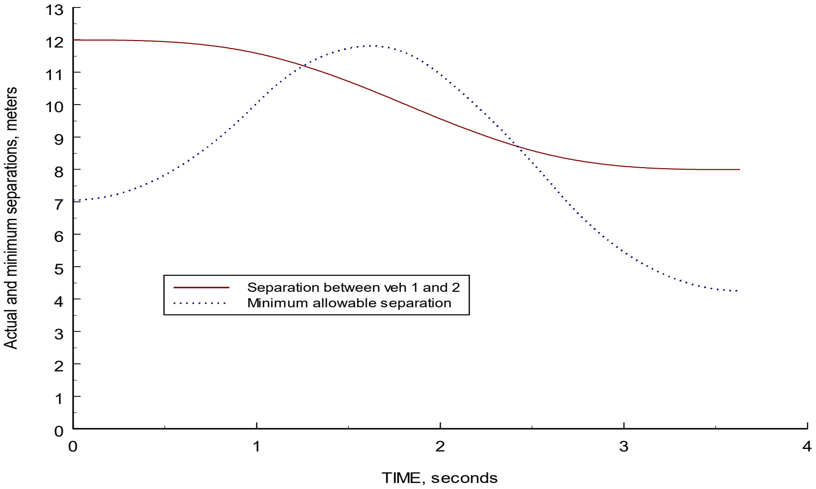


Figure 4. Separation and minimum allowable separation between two vehicles entering a station.

The parameters used in Figures 3 and 4 are those given at the beginning of the program shown in the Appendix. Many runs can be made for different accelerations and jerks. For the set shown in the program, runs were made with different line headways and control time constants to obtain the maximum negative separations as shown in Table 1 and as calculated by the program.

Table 1. Maximum headway violations for the cases shown.

$t_c \backslash LineHeadway \rightarrow$	0.5	1.0	1.3	1.5
0.05	-3.25	-1.03	0	0
0.10	-3.80	-1.59	-0.03	0
0.15	-4.36	-2.15	-0.60	0
0.20	-4.92	-2.71	-1.17	-0.01

It is seen that if the line headway between two vehicles sequentially entering a station is to be as low as one second, the control time constant must be quite small, but not particularly small using contemporary technology. Note from Figure 4 that in the case shown the small headway violation increases from zero back to zero in about one second.

In this work, we considered only the portion of the maneuver from line speed to station speed. Further development of the program included in the Appendix shows that, since the second of the pair of vehicles will be stopping at least one berth behind the first, there is no headway violation in the maneuvers from station speed to rest.

Appendix

```
'This program MINHEAD.BAS calculates the minimum headway permissible
'between a pair of vehicles decelerating into a station
'Units are MKS
DEFDBL A-Z
DIM Counter AS INTEGER
DIM A(1 TO 2) AS DOUBLE    'acceleration of vehicles 1 & 2
DIM V(1 TO 2) AS DOUBLE    'speed of vehicles 1 & 2
DIM X(1 TO 2) AS DOUBLE    'position of vehicles 1 & 2
DIM t4(1 TO 2) AS DOUBLE   'time at end of station-speed section
DIM t5(1 TO 2) AS DOUBLE   'time at command to constant deceleration
DIM t6(1 TO 2) AS DOUBLE   'time at command to constant jerk
DIM t7(1 TO 2) AS DOUBLE   'time at maneuver end, total maneuver time

DIM X1(1 TO 2) AS DOUBLE   'position of command to constant deceleration
DIM X2(1 TO 2) AS DOUBLE   'position of command to constant jerk
DIM X3(1 TO 2) AS DOUBLE   'position at beginning of station-speed section
DIM X4(1 TO 2) AS DOUBLE   'position at end of station-speed section
DIM X5(1 TO 2) AS DOUBLE   'position of command to constant deceleration
DIM X6(1 TO 2) AS DOUBLE   'position of command to constant jerk
DIM X7(1 TO 2) AS DOUBLE   'position at maneuver end, total maneuver distance

DIM D(1 TO 2) AS DOUBLE    'stopping distances of vehicles 1 & 2

g = 9.80665     'acceleration of gravity
Ac = .25 * g    'comfort deceleration
Jc = .25 * g    'comfort jerk
tJ = Ac / Jc    'jerk time constant
Af = .4 * g     'maximum failure deceleration
Jf = .4 * g     'maximum failure jerk
Ae = .4 * g     'emergency deceleration
Je = .8 * g     'emergency jerk
VL = 12         'line speed
Vsta = 8        'station speed
tc = .15         'time constant
Lveh = 2.743    'vehicle length
B = 3.048       'berth length
LineHeadway = .5 'time headway between vehicles while at line speed
t = 0           'start time
dt = .01        'computational time interval

'Calculation of the maneuver increments and transition speeds
dt01 = tJ
V1 = VL - dt01 * Ac / 2
dx01 = dt01 * (VL - Ac * dt01 / 6)
dt23 = tJ
V2 = Vsta + dt23 * Ac / 2
dx23 = dt23 * (V2 - dt23 * Ac / 3)
dt12 = (V1 - V2) / Ac
dx12 = dt12 * (V1 - dt12 * Ac / 2)
dx34 = 10      'distance vehicle 1 travels at station speed
dt34 = dx34 / Vsta          'time of veh 1 at station speed
```

```
dt45 = tJ
V5 = Vsta - dt45 * Ac / 2
dx45 = dt45 * (Vsta - dt45 * Ac / 6)
dt67 = tJ
V6 = dt67 * Ac / 2
dx67 = dt67 * (V6 - dt67 * Ac / 3)
dt56 = (V5 - V6) / Ac
dx56 = dt56 * (V5 - dt56 * Ac / 2)

'Times and position increments at the transition points
t1 = dt01
t2 = t1 + dt12
t3 = t2 + dt23
t4(1) = t3 + dt34            'this and following times for veh 1
t5(1) = t4(1) + dt45
t6(1) = t5(1) + dt56
t7(1) = t6(1) + dt67            'maneuver time
t4(2) = t3 + dt34 - B / Vsta 'this and following times for veh 2
t5(2) = t4(2) + dt45
t6(2) = t5(2) + dt56
t7(2) = t6(2) + dt67            'maneuver time


X1(1) = dx01
X2(1) = X1(1) + dx12
X3(1) = X2(1) + dx23
X4(1) = X3(1) + dx34
X5(1) = X4(1) + dx45
X6(1) = X5(1) + dx56
X7(1) = X6(1) + dx67

X1(2) = dx01
X2(2) = X1(2) + dx12
X3(2) = X2(2) + dx23
X4(2) = X3(2) + dx34 - B        'veh 2 stops one berth short of veh 1
X5(2) = X4(2) + dx45
X6(2) = X5(2) + dx56
X7(1) = X6(1) + dx67            'total maneuver distance

CLS
SCREEN 9
COLOR 7, 8
scaleT = 600 / t7(2)
scaleA = 10
scaleV = 10
scaleX = 4
scaleS = 40
T0 = 10
Y0 = 280
LINE (T0, Y0)-(640, Y0)
LINE (T0, Y0)-(T0, 0)

OPEN "KINEMAT.ASC" FOR OUTPUT AS #1
OPEN "SEPRATN.ASC" FOR OUTPUT AS #2

DO
  'Motion of first vehicle
```

```
IF t <= t1 THEN
   DelT = t
   A(1) = -Jc * DelT
   V(1) = VL + DelT * A(1) / 2
   X(1) = DelT * (VL + DelT * A(1) / 6)
ELSEIF t <= t2 THEN
   DelT = t - t1
   A(1) = -Ac
   V(1) = V1 + DelT * A(1)
   X(1) = X1(1) + DelT * (V1 + DelT * A(1) / 2)
ELSEIF t <= t3 THEN
   DelT = t - t2
   A(1) = -Ac + Jc * DelT
   V(1) = V2 + DelT * (-Ac + A(1)) / 2
   X(1) = X2(1) + DelT * (V2 + DelT * (-2 * Ac + A(1)) / 6)
ELSEIF t <= t4(1) THEN
   DelT = t - t3
   A(1) = 0
   V(1) = Vsta
   X(1) = X3(1) + Vsta * DelT
ELSEIF t <= t5(1) THEN
   DelT = t - t4(1)
   A(1) = -Jc * DelT
   V(1) = Vsta + DelT * A(1) / 2
   X(1) = X4(1) + DelT * (Vsta + DelT * A(1) / 6)
ELSEIF t <= t6(1) THEN
   DelT = t - t5(1)
   A(1) = -Ac
   V(1) = V5 + DelT * A(1)
   X(1) = X5(1) + DelT * (V5 + DelT * A(1) / 2)
ELSEIF t < t7(1) THEN
   DelT = t - t6(1)
   A(1) = -Ac + Jc * DelT
   V(1) = V6 + DelT * (-Ac + A(1)) / 2
   X(1) = X6(1) + DelT * (V6 + DelT * (-2 * Ac + A(1)) / 6)
ELSE
   A(1) = 0
   V(1) = 0
   X(1) = X7(1)
END IF

'Motion of second vehicle
tsec = t - LineHeadway
IF tsec <= 0 THEN
   DelT = tsec
   A(2) = 0
   V(2) = VL
   X(2) = DelT * VL
ELSEIF tsec <= t1 THEN
   DelT = tsec
   A(2) = -Jc * DelT
   V(2) = VL + DelT * A(2) / 2
   X(2) = DelT * (VL + DelT * A(2) / 6)
ELSEIF tsec <= t2 THEN
   DelT = tsec - t1
   A(2) = -Ac
   V(2) = V1 + DelT * A(2)
```

```
     X(2) = X1(2) + DelT * (V1 + DelT * A(2) / 2)
  ELSEIF tsec <= t3 THEN
     DelT = tsec - t2
     A(2) = -Ac + DelT * Jc
     V(2) = V2 + DelT * (-Ac + A(2)) / 2
     X(2) = X2(2) + DelT * (V2 + DelT * (-2 * Ac + A(2)) / 6)
  ELSEIF tsec <= t4(2) THEN
     DelT = tsec - t3
     A(2) = 0
     V(2) = Vsta
     X(2) = X3(2) + Vsta * DelT
  ELSEIF tsec <= t5(2) THEN
     DelT = tsec - t4(2)
     A(2) = -Jc * DelT
     V(2) = Vsta + DelT * A(2) / 2
     X(2) = X4(2) + DelT * (Vsta + DelT * A(2) / 6)
  ELSEIF tsec <= t6(2) THEN
     DelT = tsec - t5(2)
     A(2) = -Ac
     V(2) = V5 + DelT * A(2)
     X(2) = X5(2) + DelT * (V5 + DelT * A(2) / 2)
  ELSEIF tsec < t7(2) THEN
     DelT = tsec - t6(2)
     A(2) = -Ac + Jc * DelT
     V(2) = V6 + DelT * (-Ac + A(1)) / 2
     X(2) = X6(2) + DelT * (V6 + DelT * (-2 * Ac + A(1)) / 6)
  ELSE
     A(2) = 0
     V(2) = 0
     X(2) = X7(2)
  END IF

  D(1) = .5 * V(1) * (V(1) / Af + Af / Jf)  'stopping distance of veh #1
  D(2) = .5 * V(2) * (V(2) / Ae + Ae / Je)  'stopping distance of veh #2

  Separation = X(1) - X(2)
  IF Separation < Lveh + V(2) * tc THEN SLEEP
  IF V(2) > 0 THEN Headway = Separation / V(2)
  MinSeparation = Lveh + V(2) * tc + D(2) - D(1)
  IF V(2) > 0 THEN MinHeadway = MinSeparation / V(2)
  dSep = Separation - MinSeparation
  IF dSep < MaxNegSep THEN MaxNegSep = dSep

  PSET (T0 + scaleT * t, Y0 - scaleA * A(1)), 14
  PSET (T0 + scaleT * t, Y0 - scaleV * V(1)), 13
  PSET (T0 + scaleT * t, Y0 - scaleX * X(1)), 12
  PSET (T0 + scaleT * t, Y0 - scaleA * A(2)), 11
  PSET (T0 + scaleT * t, Y0 - scaleV * V(2)), 10
  PSET (T0 + scaleT * t, Y0 - scaleX * X(2)), 9

  PSET (T0 + scaleT * t, Y0 - scaleS * Separation), 5
  PSET (T0 + scaleT * t, Y0 - scaleS * MinSeparation), 6

  'PRINT USING "#####.##"; t; A(1); V(1); X(1) ; A(2); V(2); X(2);
Separation; MinSeparation
  'PRINT USING "#####.##"; t; V(2); Separation; Separation - Lveh - V(2) *
tc; MinSeparation; dSep; Headway; MinHeadway
```

```
    IF Counter = 20 THEN
       Counter = 0
       'SLEEP
    END IF
    Counter = Counter + 1
    'WRITE #1, t, A(1), V(1), X(1), A(2), V(2), X(2)
    'WRITE #2, t, Separation, MinSeparation
    t = t + dt
LOOP UNTIL t > t7(2) + 1
PRINT "    MaxNegSep = ";
PRINT USING "###.##"; MaxNegSep
CLOSE #1
CLOSE #2
```

Some History of PRT Simulation Programs

J. Edward Anderson, Ph.D., P. E.

Abstract

This paper documents 32 vehicle simulation programs that have been developed since 1969 to simulate the operation of automated vehicles operating in networks of guideway under a variety of strategies.

Introduction

Every group intent on designing a marketable Personal Rapid Transit system has needed to have close at hand a simulation program that permits detailed study of the system's performance characteristics both for design and planning purposes. Since all or any assumptions made in developing the simulation must be thoroughly understood; each group, practically speaking, must develop its own simulation program. Many engineers have understood this necessity and in time I expect that the details will be taught in engineering courses to the benefit of not only PRT designers, but the consulting firms and planners who need to know the details. Over the 40 years in which I have been involved in PRT research, development and design I have become aware of 32 automated vehicle simulation programs of varying degrees of completeness, and it is my purpose in writing this paper to call attention to and discuss them, with the hope thereby that the best ideas will come into common use as the field of PRT matures. The simulation tool is the slide rule of PRT development. If there are additional similar simulation programs, I regret not including them, but I simply am not aware of them.

1970s Era PRT Network Simulation Programs

During the 1970s, at least the following organizations or individuals developed PRT simulation programs:

1. Royal Aircraft Establishment, Ministry of Defense, Farnborough, UK
2. The Aerospace Corporation, El Segundo, CA
3. Morgantown PRT Program

4. Morse Wade, IBM Corporation, Poughkeepsie, NY
5. Applied Physics Laboratory, Johns Hopkins University
6. Prof. Harold York, University of Minnesota
7. Marvin A. Sirbu, Massachusetts Institute of Technology
8. IBM Corporation, Gaithersburg, MD
9. Kandasamy Thangavelu, Colorado Regional Transportation District
10. Johnson, Walter & Wilde, Colorado Regional Transportation District
11. S & A Systems, Dallas, Texas
12. Dr. Sakasita, Colorado Regional Transportation District
13. Professor Alain Kornhauser, Princeton University
14. Messerschmitt-Bölkow-Blohm, Munich
15. University of Karlsruhe, West Germany
16. Raytheon Missile Systems Division

Royal Aircraft Establishment

D. I. Paddison, "Cabtrack Studies: Estimation of Capacity of Cabstops," RAE Technical Report 71132, June 1971. 49 pages and 10 figures.

J. C. H. Longrigg, "Cabtrack Studies: Data Sheets for Track Layouts," RAE Technical Report 71024, February 1971. 32 pages and 7 figures.

The Summary of Paddison's report contains the statement: "Results are presented of a digital computer simulation of the operation of six small and medium-sized Cab-stops." We now refer to "Cab-stops" as "Stations." Section 1.2 of Paddison's report contains the sentence: "Study of the control of a complete network will require a simulation of a network in operation." None of the RAE reports I have seen discuss a complete network simulation, but I have not seen all of the reports the study produced. However, Longrigg provides the formulae needed for calculating all of the curves and off-line transitions used in a complete PRT network.

The Aerospace Corporation

A. V. Munson, Jr., H. Bernstein, J. R. Buyan, K. J. Liopiros, and T. E. Travis of The Aerospace Corporation, "Quasi-Synchronous Control of High-Capacity PRT Networks," *PRT*[15], pp. 325-350. On page 349 the following paragraphs can be found:
B.

"The PRT network simulation was implemented to assist in establishing system performance parameters such as trip times, waiting times, empty car trip lengths, and guideway and stations loadings as a function of system configuration and operating strategies. A

[15] J. E. Anderson, J. L. Dais, W. L. Garrard, A. L. Kornhauser, *Personal Rapid Transit*, Institute of Technology, University of Minnesota, April 1972.

secondary but quite important objective was to provide a test bed for development and demonstration of routing and empty car handling algorithms.

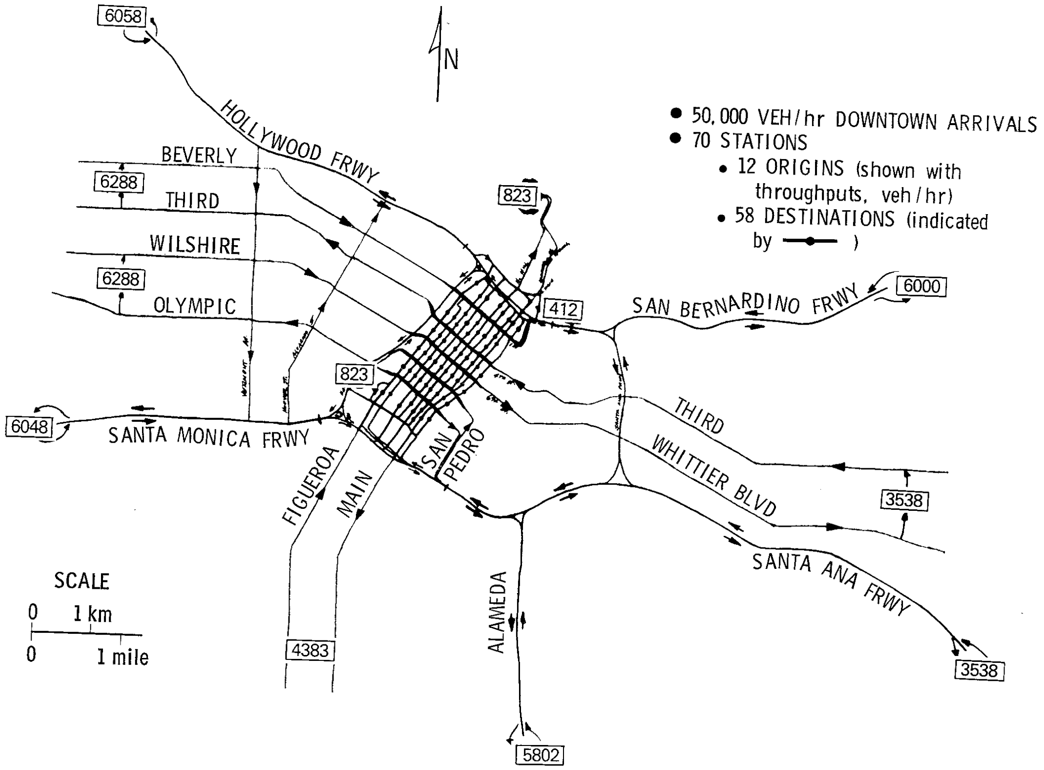
The simulation is implemented in SIMSCRIPT and is currently operational on The Aerospace Corporation CDC 6000 series computers. The configuration of a network with all of its components is specified parametrically and great flexibility is available. In the current version, PRT cars are simulated explicitly so that detailed records may be kept on individual simulated trips. This level of simulation has many uses but because of computer memory requirements is somewhat limited as to the network size that can be accommodated. Another version of the simulator, which uses much of the basic structure already developed, is being designed in which cars are modeled implicitly. This simulation will not provide the detail on individual trips but will allow simulation of much larger networks for study of global questions such as guideway and station loadings."

J. H. Irving, H. Bernstein, J. Katz, P. Dergarabedian, and T. H. Silva, The Aerospace Corporation, "Vehicle Management on Large PRT Networks," *PRT III*[16], pp. 345-368.

C. L. Olson, The Aerospace Corporation, *Independent Study of Personal Rapid Transit*, Report No. UMTA-CA-06-0090-77-1, 16 December 1977.

Jack H. Irving, Harry Bernstein, C. L. Olson, and Jon Buyan, *Fundamentals of Personal Rapid Transit*, Lexington Books, D. C. Heath and Company, Lexington, Massachusetts, 1978, 332 pages.

[16] D. A. Gary, W. L. Garrard and A. L. Kornhauser, *Personal Rapid Transit III*, University of Minnesota, June 1976.



The analysis of a PRT system requires the following steps:

1. Develop and calculate via computer the coordinates of the lines and stations of the network, which must assume certain types of stations and intersections, the discussion of which is given in the above-mentioned documents. As background for this work, equations for calculating all of the curves, transitions to off-line stations, and maneuvers had to be developed; and the throughput of stations and intersections had to be understood. For the Aerospace work, the curve and maneuver calculations are given in Appendix A of Irving et al. The earliest paper I have found on the details of the Aerospace work on station design and throughput is found in *PRT II*[17] on pages 449-460 in the paper "PRT Station Operational Strategies and Capacities," by K. J. Liopiros. Discussion of an intersection simulator is given in the above-mentioned *PRT* paper by A. V. Munson, et al.

2. Develop a switch table, i.e., for each line-to-line diverge point a Left or Right switch command gives the optimum path to every station in the system. The Aerospace papers

[17] J. E. Anderson, Ed. *Personal Rapid Transit II,* University of Minnesota, 1974.

describe in general terms their method for calculating such a table, which they call a Routing Table.

3. Estimate ridership. The Aerospace papers describe a novel Monte Carlo mode split model that performs this task more accurately than methods generally used in estimation of ridership on conventional transit systems.

4. Estimate of the line and station loadings, the number of vehicles – occupied and empty – needed, the trip lengths, and for given line speeds the trip times. The results of such calculations are given and discussed in the Aerospace reports.

   NOTE: From my own analysis of a PRT network for Indianapolis in 1980, I developed a method for calculating these quantities. Subsequently one of my students at Boston University, Richard Komerska, developed a convenient method to perform the calculations on a PC. These works are referenced and discussed below under 1980 era programs.

5. Finally, detailed simulations are run in which the arrival, loading and unloading times of each passenger group are randomized. Such a simulation handles merge conflicts and all vehicle movements exactly as they would be handled in a real system. Thus this tool not only gives accurate information on wait times, ride times, and wave-offs; but it provides the tool needed to verify the operational software. Munson et all reported in the above-mentioned *PRT* paper that this work was done, apparently for small networks, but the network on which it was done is not identified in the papers I have referenced.

Morgantown PRT Program

R. H. Bryan, S. E. G. Elias, and R. E. Ward, “Simulation of West Virginia University’s Personal Rapid Transit System,” Summer Computer Simulation Conference, San Diego, CA, June 14-16, 1972. I have no detail on this work, but because this system, the Morgantown system, has been in operation since 1972 and was well funded, the simulation work would have to be complete.

Morse Wade, IBM Corporation, Poughkeepsie, NY

R. Morse Wade, Staff Engineer, IBM Corporation, “THE MANHATTAN PROJECT: A Cost-Oriented Control System for a Large Personal Rapid Transit Network,” *PRT II*, pp. 417-423.

A preliminary analysis of a 500-mile synchronously controlled PRT network for Manhattan is presented; however, few details are given that would help one understand how it was done. It does not appear from the text that Wade carried his simulation to the level of following individual vehicles through the network.

Applied Physics Laboratory, Johns Hopkins University

The APL work to which I have access includes the following papers:

E. J. Hinman & G. L. Pitts, "Practical Safety Considerations for Short-Headway Automated Transit Systems," *PRT II*, pp. 375-380.

S. J. Brown, Jr., "Design Considerations for Vehicle State Control by the Point-Follower Method," *PRT II*, pp. 381-389.

W. J. Roesler, M. B. Williams, B. M. Ford and M. C. Waddell, "Comparisons of Synchronous and Quasi-Synchronous PRT Vehicle Management and Some Alternative Routing Algorithms," *PRT II*, pp. 425-438.

M. B. Williams, B. M. Ford, and M. C. Waddell, "Analysis of Multiple Party Vehicle Occupancy in an Automated, Guideway System, APL/JHU, CP 042/TPR 032, March 1976, 96 pages.

The first two of these papers are preparatory for simulating the operation of vehicles in a network of guideways. The third paper bases its results on a simulation of vehicles operating in a simple network of two-way guideways containing six stations, but interconnected in such a way that there are four paths from any station to any other. Asynchronous, quasi-synchronous, and fully synchronous operation were modeled. It appears that at least 360 vehicles were followed in the simulation. The operation of merges is described for the quasi-synchronous strategy. In the synchronous strategy, all merge conflicts are resolved before a vehicle is permitted to leave the origin station.

The fourth paper describes, as the title suggests, the operation of an automated guideway system using multi-party vehicles. From our interest in documenting simulation programs, this paper is important because it includes the code of its simulation program. Most of the results presented relate to a single two-way loop containing 12 off-line stations, but a more complex system containing three two-way branches meeting at a center point is mentioned. Data curves are shown corresponding to runs with up to about 650 vehicles. This statement is made: "Many aspects of system operation such as details of vehicle movement, i. e. speed variations and merging at station exits, and considerations of station design and capacity, were ignored." They were felt to have only secondary effects on the desired results, which were the relationship between fleet size, vehicle capacity, vehicle occupancy, passenger delays, and the number of intermediate stops required. How accurate that assumption may be can only be determined from a more detailed simulation model.

Prof. Harold York, University of Minnesota

H. L. York, "The Simulation of a PRT System Operating under Quasi-Synchronous Control," *PRT II*, pp. 439-447.

Professor York tested his PRT simulation program on the network shown on the next page, which consists of 23 stations, four multi-level interchanges, and four each of simple merges and diverges. He assumed one-second headway and with his demand he assumed 1100 vehicles. His program produced line flows in vehicles per hour and average waiting times, which he analyzed in some detail. He mentions accumulating data on aborts (which I now call wave-offs as a politically neutral equivalent) but shows no data. In his simulation he divided his guideway into fixed intervals of equal time, taking the set headway as the time interval, and in each interval he placed the destination number of the vehicle that occupies it, with zero for no vehicle. Presumably these time intervals correspond to shorter distances as the vehicles maneuver into and out of the stations. In this way the size of the network is a function of line speed, but need not be and is not stated. His program was written in FORTRAN and ran on a CDC 6400 mainframe computer.

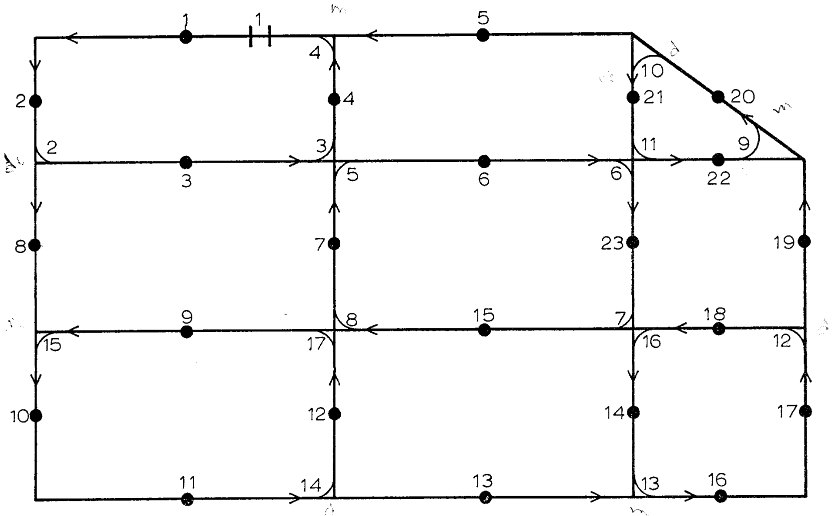
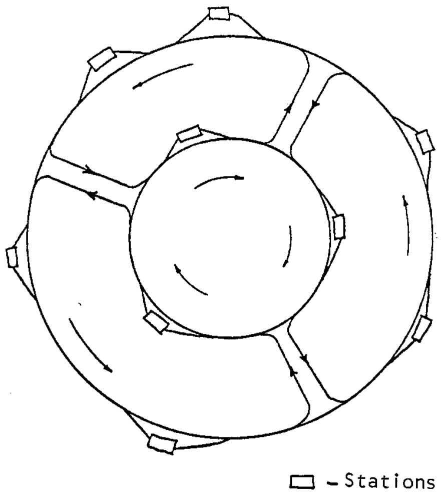


Figure 1. Test Network

Marvin A. Sirbu, Massachusetts Institute of Technology

Marvin A. Sirbu, Jr., “Station Configuration, Network Operating Strategy and Station Performance,” *PRT II*, pp. 461-478.

In Dr. Siribu's work, he was mainly interested in understanding the performance and throughput of PRT stations of two types: parallel-loading and linear. To accomplish his purpose, he developed a simulation program in the Simcript 11.5 programming language that ran on an IBM 370/165 mainframe computer and made runs of 1.5 to 6 hours of simulated time. His guideway, shown at the right, consisted of an outer and an inner concentric ring interconnected in three places with pairs of radial lines, thus giving 6 line-to-line diverges and 6 line-to-line merges. The flow in the outer ring was counterclockwise and in the inner ring clockwise. There were two stations on the outer ring between each pair of radial lines and one station on the inner ring between each pair of radial lines, making a total of nine stations. He operated vehicles in a modified synchronous scheme at 3 seconds headway. The modification was to permit vehicles to slip a slot to resolve merge conflicts, mainly as a result of occasional station rejections or wave-offs that could occur when a station was too full to receive a vehicle. He determined station capacity as a function of a tolerable frequency of station rejections. At each time headway he updated the positions of the vehicles on the links, merges and diverges. Station operations were event oriented in terms of random arrivals of customers, random loading times, random unloading times, and as a result of these random processes variable vehicle dispatching times. His reports give customer statistics, vehicle statistics, and station statistics. He concluded that his linear stations provided better performance than his parallel-bay stations.



<u>IBM Corporation, Gaithersburg, MD</u>

Martin S. Ross and Alan D. Melgaard, "Systems Management Analysis of Large PRT Networks," *PRT III*, pp. 369-376.

On the network shown below, Ross and Melgaard simulated the operation of automated vehicles of various sizes assuming seven service polities ranging from pure PRT operation (demand responsive single party) to fully scheduled operation. The network has 22.8 miles of guideway, 22 off-line stations with 6 loading and unloading berths each, 36 merges and 36 diverges. They ran the simulations on an IBM 370/155 mainframe computer. Their simulation produced 25 measures of effectiveness that related to resource utilization, performance, and level of service in terms of wait times. The pure PRT runs used a minimum headway of 1 second with a fleet of 1193 vehicles. For the larger-vehicle systems the headways ranged from 2 to 15 seconds and the fleet consisted of 423 to 178 vehicles, with the smaller fleets used for the longer headway scheduled service. Their results showed a high level of sensitivity to vehicle capacity, service policy, and trip demand. For example, the average wait time for pure PRT was only 42 seconds,

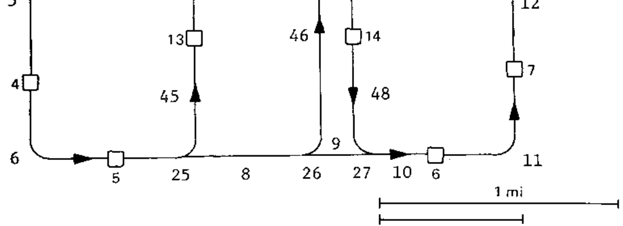


Figure 4. CBD Network

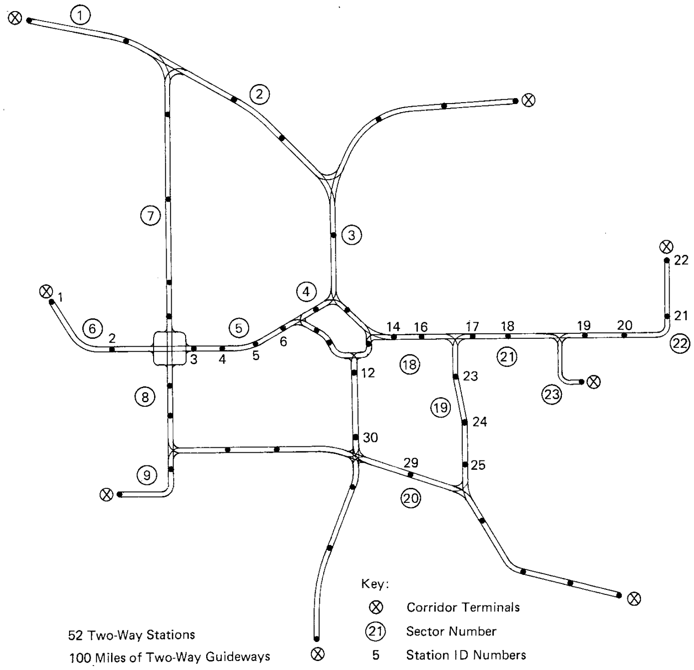
but for the larger-vehic nger by a factor of 5 to 13.

3

Thangavelu, Colorado Regional Transportation District

K. Thangavelu, “Development and Evaluation of Service Policies for Medium-Headway Automated Rapid Transit Systems,” *PRT III*, pp. 329-344.

Thangavelu simulated the operation of medium capacity vehicles operating at a wide range of headways on the city-wide automated rapid transit network (Colorado RTD’s 1973 plan) shown below. He assigned passengers to the stations from city-wide demand data that had been obtained in previous studies and determined the minimum-time routes based on a standard linear programming model. He tested dynamically scheduled and what he called “advanced scheduled service” policies. His program output some 23 parameters including every imaginable variable produced in operating such a system. Typical results show average waiting time, average number of stops, average vehicle occupancy, and empty-vehicle statistics.



Johnson, Walter & Wild, Colorado Regional Transportation District

R. E. Johnson, H. T. Walter, and W. A. Wilde, "Analysis and Simulation of Automated Vehicle Stations," *PRT III,* pp. 269-281.

Appendix A of this paper describes a simulation program that models the flow of vehicles through an off-line station. It is a discrete event simulator consisting of 16 routines, is written in FORTRAN IV, and ran on any CDC 6000 series computer.

Appendix B of this paper describes a second simulation program that includes a detailed representation of control-system operation. It is a Monte-Carlo, discrete event simulator also written in FORTRAN. "Extensive input options and input parameters were designed to allow the definition and input of diverse control systems concepts and operating philosophies." The program models passenger and vehicle movement through the station in detail in 0.1 sec steps. The authors say they were working on extending this program to simulation of an entire network.

S & A Systems, Dallas, Texas

J. G. Srygley, S. M. Stokes, and T. N. Coomer, "Transportation System Simulation – Case Studies". This paper was presented at the 46th National Meeting of ORSA (San Juan, Puerto Rico, October 16-18, 1974). It included detailed simulation modeling of GRT Off-Line Stations for Colorado RTD's Alternatives Analysis.

Dr. Sakasita, Colorado Regional Transportation District

Masami Sakasita, "An Analysis of Merge Control for the Automated Scheduled Transit (AST) System," RTD, January 1975, 87 pages.

This very detailed program was written to study through computer simulation the operation of merges. It is another excellent example of the use of computer simulation to study transit problems. The report contains a copy of the program used.

Professor Alain Kornhauser, Princeton University

Alain L. Kornhauser, Steven Strong, and Paul Mottola. "Computer-Aided Design and Analysis of PRT Systems," *PRT III*, pp. 377-384.

The PRT simulation program developed at Princeton University was applied as a 29 station, 21 interchange network for Trenton, New Jersey. The simulation operated in the quasi-synchronous mode and was designed to accurately model a hilly city. It used real demand data and resolved line-to-line merges and flows in and around the stations. A method of simulating the flow of empty vehicles is included in the paper. The outputs are wait times, passenger and vehicle miles traveled, fleet size, etc. Shared riding was investigated.

Messerschmitt-Bölkow-Blohm, Munich

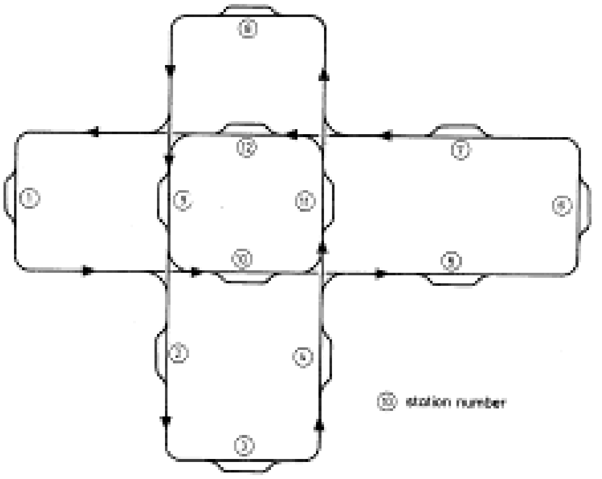
Richard Hesse, "Normal and Emergency Control of Automated Vehicles at Short Headways, with Special Emphasis on the Development, Testing, and Dynamic Simulation of the Cabintaxi System," *PRT III*, pp. 283-288.

Hesse describes a detailed simulation program MBB used to study the Cabintaxi PRT system in specific applications in German cities. The simulation is an asynchronous car follower and addresses all aspects of the movement of vehicles and passengers, passenger destinations, optimum vehicle paths, movement of empty vehicles and outputs the results on a color TV screen as well as in print format, which includes wait time statistics, passenger-miles traveled, energy use, etc. The program was written in assembler language and permitted a network size up to about 30 km, 63 stations, and 1000 vehicles to be studied. It could be expanded by a factor of 10 by enlarging the core memory. The program was used to develop and test network control systems and the optimization of network layouts with respect to topology, track positioning, stations and number of vehicles.

University of Karlsruhe, West Germany

Gerd Bahm, "The Influence of Fleet Size and Vehicle Capacity on the Performance and Service Quality of Group Rapid Transit Systems," *PRT III*, pp. 289-298.

Bahm developed a simulation model, written in the SIMULA 67 programming language, which simulated vehicles of any size or any number of seats operating under automatic control in a network of guideways. The vehicles operated quasi-synchronously in slot lengths that permit complete stoppage, i.e., the brick-wall stop distance. The model was a mixture of an event-oriented simulation and a discrete time-step simulation. It advanced in steps equal to the minimum headway. Passenger arrivals were randomized. The most important output variables were the waiting time, average speed, distance travelled, and headways between vehicles. The paper reported results of application of the model to the network shown here. The author concluded with the statement that he was investigating large networks.



Raytheon Missile Systems Division

D. Girard, “AGTT Car Follower Autopilot – Design and Simulation.” Missile Systems Division, Raytheon Company, Memo No. SDD-76-836, 18 March 1976, 97 pages.

This report was prepared as a part of a program to prepare to bid on a federal RFP on control of PRT systems. To test the autopilot, which operated as a car follower, two simulation programs were developed: one of them, called String, employed six vehicles and was used for testing line maneuvers such as responding to line-speed changes, overtaking a slow vehicle, and emergency stopping. The other, called Merge, used forty vehicles to evaluate the merging process, test ride quality and determine the length requirement of the parallel data region. The larger number of vehicles was needed to reach steady state. The detailed dynamics of each vehicle was followed during these simulations.

1980s Era PRT Network Simulation Programs

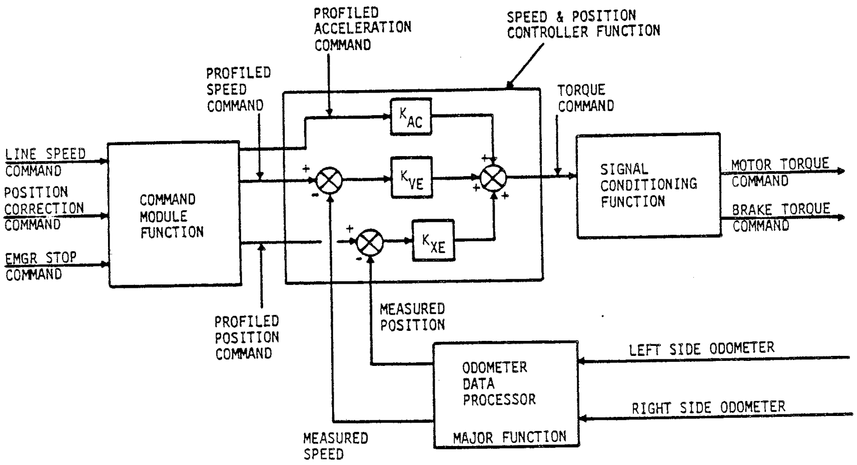
At the beginning of the 1980s, any serious work on a PRT simulation program required the use of computers far too expensive for an ordinary individual to afford, but by the end of this decade such a program could be developed on an easily affordable laptop PC. During the 1980s, PRT simulation programs were developed by at least the following organizations or individuals:

17. Boeing Company
18. Otis Elevator Company
19. The author.

Boeing AGRT Simulation Work

William E. Greve, Donald E. Haberman, and Robert P. Lang, “Advanced Group Rapid Transit Vehicle Control Unit Design Summary, Boeing Aerospace Company, UMTA-WA-06-0011-84-3, May 1985, 249 pages.

Don D. Lyttle, Dave B. Frietag, and Doug H. Christenson, Boeing Aerospace Company, “Advanced Group Rapid Transit Phase IIB, Executive Summary & Final Report,” UMTA-WA-06-0011-86-1, March 1986, 205 pages.



The Boeing work in the AGRT program mainly involved developing a vehicle longitudinal control system (VLCS) that would control each of a system of vehicles operating in a network at a minimum of 3 seconds headway. Their controller was a "point follower" in that, as given in the above control block diagram, which is taken from page 18 of the above-cited Greve, Haberman and Lang report, each vehicle follows profiled acceleration, speed and position commands. Feedback of position and speed was taken from the odometers shown in the above diagram, which were digital encoders that directly provided distance information, and speed by differentiating the distance pulses. Proof of their control system involved extensive simulation work in which real components were an increasing portion of the simulation.

Otis AGRT Simulation Work

W. Womack, "Vehicle Longitudinal Control and Reliability Project Summary," Otis Elevator Company, Report No. UMTA-IT-06-0148-79-10, June 1979, 134 pages.

"Zone management and Control Conceptual Design," Otis Elevator Company, Transportation Technology Division, Denver, Colorado, August 1981, 124 pages.

These are reports of the second of the two federally funded AGRT studies aimed at development of an appropriate VLCS that would permit operation of vehicles in networks of guideways as close as 3 seconds apart. These reports describe a point-follower system in which each vehicle followed a calculated maneuver profile to accomplish slot slipping during merging, deceleration into a station berth, acceleration to line speed, and speed changes. During their development program Otis used simulations to verify in detail the operation of their control concept during all maneuvers.

J. E. Anderson simulation program

J. E. Anderson, "Calculation of Performance and Fleet Size in Transit Systems," *Journal of Advanced Transportation*, 16:3(1982)231-252.

Richard J. Komerska, *Development of a Modeling Tool for the Preliminary Design of Personal Rapid Transit Networks*, a Master of Science Thesis in Civil Engineering, University of California, Irvine, 1995 163 pages.

With reference to item #4 on page 4, the above paper derives equations from which to calculate the quantities indicated. Komrska programmed a model that provides a convenient way to make these calculations on a PC.

In August 1986, I initiated the development of a PRT simulation program at a time when I was teaching engineering at Boston University and at the same time organizing and working with a team of engineers to ready the specifications for an operational PRT system. Notwithstanding these other commitments, I had a working program ready by 1990 in time to be included in a proposal for a Phase I PRT Design Study for the Northeastern Illinois Regional Transportation Authority, which was completed in 1992. The program was subsequently used to analyze a 3-mile, 8-station network for Rosemont, Illinois.

With no budget for use of a mainframe computer or a DEC Workstation, both computer hardware and software were then quite limited for me. During my first year at BU I had access only to the first Compaq so-called "portable" PC, which had a 9-inch screen, only 64K internal memory and no hard drive. To see what my simulation was doing, the first version of which I had running within a month of a standing start, I had to refer to print output. A year later I was able to purchase a 286 machine, but it was too slow until it could be upgraded with the 287 coprocessor. A year or so later I upgraded to a 386 then 387. At the time, after experimenting with C, Pascal and various versions of BASIC, I programmed in BASIC because it took less time to program. But it was not terribly satisfactory until Microsoft came out with Quick BASIC 4.0, which a year or so later upgraded to Professional Basic 7.1, which I used for many years as my major computing device. Considering the other commitments I had and my limitations on hardware and software, I estimate that the development of a usable PRT simulation program with today's tools would take me working full time no more than about 4 man-months of effort or about 1000 hours at 60 hours per week.

1990s and 2000s Era PRT Network Simulation Programs

During the 1990s, as a result of the Chicago project, PRT development became quite active again. In Dr. Jerry Schneider's web page http://faculty.washington.edu/jbs/itrans/ in the index under Simulations reference to the following PRT simulation programs can be found. Because of the details given by Dr. Schneider, I see no need to comment further on these programs, except for my own.

20. Hermes PRT Network Simulator by Chris Xithalis (Greece)
21. PRT International (USA) www.prtnz.com

    J. E. Anderson, *Transit Systems Theory,* Lexington Books, D. C. Heath and Company, Lexington, MA 1978, 340 pages, available on www.advancedtransit.org for calculation of curves and maneuvers.

    J. E. Anderson, "Longitudinal Control of a Vehicle," *Journal of Advanced Transportation*, 31:3:237-247, 1997 for the gains of a vehicle controller.

    J. E. Anderson, "Control of Personal Rapid Transit Systems," *Journal of Advanced Transportation*, 32:1:pp. 57-74, 1998 for explanation for the asynchronous point-follower system.

    M. Joborn, "Empty freight car distribution at Swedish State Railways," *Computers in Railways VI*, WIT Press, Boston, Southampton, 361-370, 1998 for an effective means of moving empty vehicles.

    J. E. Anderson, "Simulation of the Operation of Personal Rapid Transit Systems." *Computers in Railways VI*, WIT Press, Boston, Southampton, pp. 523-532, 1998 for a description of the author's PRT simulator.

    J. E. Anderson, "A Review of the State of the Art of Personal Rapid Transit." *Journal of Advanced Transportation*, 34:1, 2000 for how the author applied Joborn's empty-vehicle movement concept.
22. Logistic Centrum's PRTsim software (Sweden)
23. RUF International (Denmark)
24. The Innovative Transportation Simulator (Italy)
25. TrakEdit: PRT Simulator from Taxi 2000 (USA)
26. Raytheon's NETSIM PRT Simulation Program (USA)
27. Calver Marketing (UK)
28. JKH Mobility Services' Simulation Program (USA)
29. Princeton's PRT Simulation Program (USA)
30. BASim (Australia)
31. Simulation and Analysis Tools for Urban Automated Rapid Transit Networks (S.A.T.U.R.N.) (Canada)
32. PRT Microsimulation (UK)