TNO Institute for Perception

TNO Defence Research

TNO-report **IZF 1993 C-24**

J.H. Hogema

SIMULATION OF AUTOMATED LONGITUDINAL CONTROL OF VEHICLES IN A PLATOON

TNO Institute for Perception

TNO Defence Research

Kampweg 5
P.O. Box 23
3769 ZG Soesterberg
The Netherlands

Fax +31 3463 5 39 77
Telephone +31 3463 5 62 11

TNO-report **IZF 1993 C-24**

J.H. Hogema

SIMULATION OF AUTOMATED LONGITUDINAL CONTROL OF VEHICLES IN A PLATOON

Contractor:
Ministry of Transport,
Public Works and Water Management
P.O. Box 1031, 3000 BA Rotterdam
The Netherlands

Number of pages: 32

CONTENTS Page

SUMMARY 5

SAMENVATTING 6

1 INTRODUCTION 7

2 SIMULATING THE PLATOON 8
2.1 Model 8
2.2 Implementation 10

3 RESULTS 11
3.1 Individual vehicles 11
3.2 Platoons of vehicles 16

4 DISCUSSION AND CONCLUSIONS 22

REFERENCES 24

APPENDIX Model listing 25

Rep.No. IZF 1993 C-24

TNO Institute for Perception
Soesterberg, The Netherlands

Simulation of automated longitudinal control of vehicles in a platoon

J.H. Hogema

SUMMARY

A number of simulations has been carried out to study the longitudinal dynamic behaviour of a platoon of eight vehicles. Three vehicle categories have been distinguished, viz. a mid-size passenger car, a van, and a truck. Each vehicle is provided with a sensor to measure the following distance, and with communication equipment to receive data from the lead vehicle and to send data to the following vehicle. This equipment is used in a system which automatically controls speed and following distance; the driver is no longer involved in the longitudinal driving task. The main simulation scenarios were: encountering a road incline, executing a speed reduction, and entering and leaving the platoon. Two different speed controllers have been compared, namely a fixed PID-controller and a variable PID-controller with speed-dependant parameters. It appeared that the later performed better.
The platoon behaviour was satisfactory, but in order to maintain stability, its length is expected to be limited to approximately 15 vehicles.

Rap.nr. IZF 1993 C-24

Instituut voor Zintuigfysiologie TNO
Soesterberg

Simulaties van automatische longitudinale regeling van voertuigen in een peloton

J.H. Hogema

SAMENVATTING

Er is een aantal simulaties uitgevoerd om het dynamische longitudinale gedrag van een peloton van acht voertuigen te bestuderen. Daarbij zijn drie voertuigcategorieën onderscheiden, nl. een middenklasse personenauto, een bestelwagen en een vrachtwagen. Elk voertuig beschikt over een sensor die de volgafstand meet, en over communicatie-apparatuur om data van de voorligger te kunnen ontvangen en om data naar het volgende voertuig te kunnen zenden. Deze apparatuur wordt gebruikt binnen een systeem dat automatisch de snelheid en volgafstand regelt; de bestuurder is niet langer betrokken bij de longitudinale rijtaak. De belangrijkste simulatiescenario's waren: het rijden op een opgaande helling, het uitvoeren van een snelheidsverlaging, en het in- en uitvoegen in het peloton. Twee verschillende regelaars zijn vergeleken, namelijk een vaste PID-regelaar en een variabele PID-regelaar met snelheidsafhankelijke parameters. De laatste bleek het beste te voldoen.
Het pelotongedrag was naar tevredenheid, maar om de stabiliteit te behouden moet de lengte naar verwachting beperkt blijven tot ongeveer 15 voertuigen.

1 INTRODUCTION

Current technologies can be utilized in a many applications in vehicle control systems, ranging in functional complexity from driver-assisting systems to completely automatic driving (Hosaka & Taniguchi, 1992). Goals of such systems are reducing the driver's workload on the one hand, and improving safety and traffic efficiency on the other. When focusing on systems that are concerned with longitudinal vehicle control, Intelligent Cruise Control (ICC) (Zhang, 1991; Chang et al., 1991) and Speed Governors (Almquist, Hydén & Risser, 1992) can be mentioned as systems that have potential to be implemented in the near future. The "intelligence" of an ICC incorporates the presence of other traffic in its speed control (in contrast to conventional cruise control, which just maintains constant desired speed). In the literature, several ICC strategies are proposed (e.g. Zhang, 1991; Rao, Varaiyja & Eskafi, 1993; Chang et al., 1991; Eliasson, 1992). Information about other traffic can be obtained by in-vehicle sensors, or by means of communication equipment (vehicle-vehicle and/or vehicle-roadside), where the use of communication has the potential advantage of larger preview.

The TNO Institute for Human Factors is involved in a number of projects which are related to ICC. For example, the project Intelligent Traffic Systems (ITS) is aimed at the development of an evaluation instrument for the assessment of ITS applications (such as ICC) by means of computer simulations in which consequences for traffic safety, emission and traffic performance can be weighted in an integrated manner (Van der Horst et al., 1993). Further, in the project IRISS experiments will be conducted to investigate Man-Machine-Interface aspects of autonomous ICC systems (Van der Horst, 1993). However, no actual experience with (simulations of) ICC systems was available. Therefore, the study described in this report has been carried out to investigate the performance and the limitations of one such system and to gain some insight in the control-theoretical aspects of automated car following.

This report deals with simulations of longitudinal dynamic behaviour of vehicles which are equipped with an ICC system using both sensors and vehicle-vehicle communication. An article of Frank et al. (1989) served as a basis. Several vehicles are combined to form an automated platoon in which the driver's longitudinal control task is entirely taken over by the ICC: speed and following distance are automatically controlled by means of the propulsion and brake system. This system allows relatively short following distances at high speeds. It must be able to maintain safe following distances under various conditions, such as (large) speed changes of the entire platoon, external disturbances (for instance, wind or slopes), and vehicles which enter or exit the platoon. Each vehicle is equipped with a sensor to measure the distance to the lead vehicle, and a communication link to receive (speed) data from the lead vehicle and to send it on to the following vehicle in the platoon. Through this communication link, the first vehicle in the platoon is thought to receive an overall speed setpoint for the entire platoon.

2 SIMULATING THE PLATOON

2.1 **Model**

In Fig. 1, the block diagram of one vehicle is depicted.

Inputs

It is assumed that in each vehicle the following distance (x'-x) is measured by means of a sensor. Each vehicle is equipped with a communication device which provides information about the speed setpoint of its lead vehicle.

Outputs

The output signals are the vehicle's speed and position, and the signal to be transmitted to the following vehicle.

Categories

Three types of vehicles are distinguished:
- cars (type A),
- vans (type B), and
- trucks (type C).

The vehicle types have an equal structure, but they differ in their parameter values, as can be seen in Table I.

Table I Parameter values for each vehicle type.

name	parameter	A	B	C
m	mass (kg)	750	1000	2000
c_1	driving coefficient	743	743	743
T_1	propulsion time constant (s)	1.0	1.738	2.0
c_a	air drag constant	1.19	2.0	3.0
τ_1	actuator delay (s)	0.3	0.4	0.6
τ_2	sensor delay (s)	0.1	0.1	0.1
τ_3	communication delay (s)	0.2	0.2	0.2

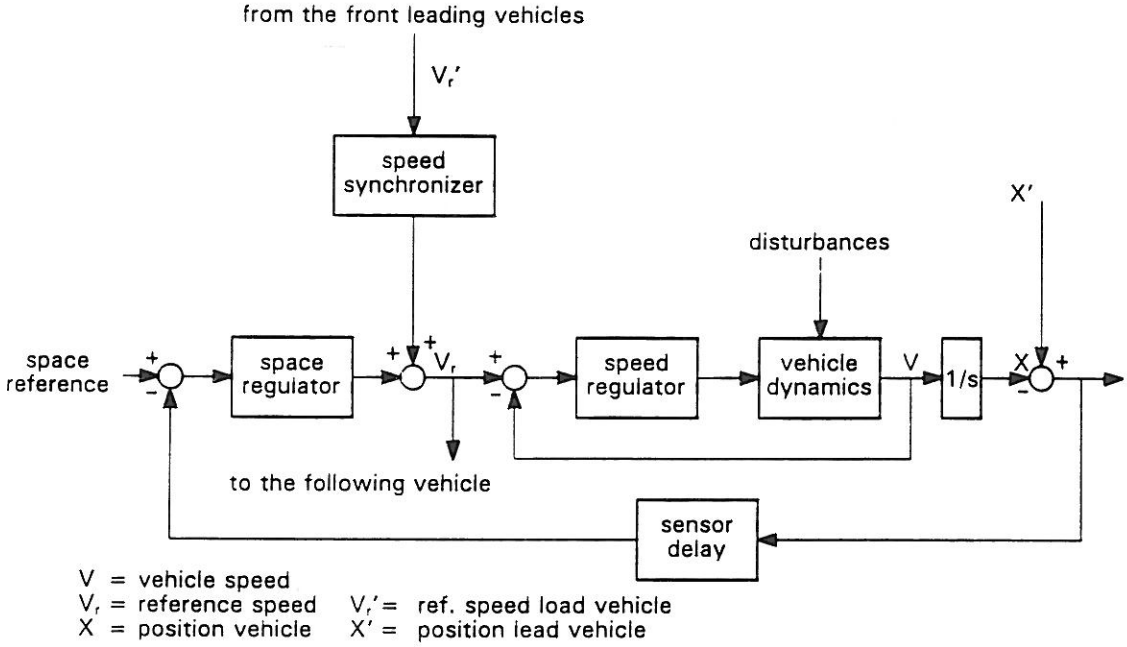

Fig. 1 Block diagram of one vehicle.

Structure

In each vehicle, two nested control loops are used to control speed and following distance, respectively. The following elements can be distinguished in the model:

- Vehicle dynamics, consisting of, among other things, mass, air drag force, a simple engine transmission system, and a delay element. External disturbances caused by wind and road angle are also included. A saturation element is included to simulate a maximum acceleration and deceleration force.
- The distance sensor is implemented in the model as a pure time delay.
- The purpose of the speed controller (inner loop) is to suppress the environmental disturbances as effectively as possible. Therefore this control loop must have a relatively high bandwidth; for this purpose, a PID-controller is used (Proportional, Integral and Derivative control; see for instance Grabbe et al., 1958).
- The space controller (outer loop) should result in smooth transients during entrainment manoeuvres. Its bandwidth is low compared with the speed controller's. A simple P-controller is used for this purpose.
- The setpoint of the following distance ("space reference"), for which a constant value is used by Frank et al. (1989). In a more sophisticated model, a speed-dependant space reference could be implemented, possibly also taking the braking capabilities of vehicles into account.
- There is a communication link from each vehicle to its following vehicle. Through this link the speed setpoint of the lead vehicle is obtained. A delay in this communication link has been included.

- A speed synchronizer, which filters the signal received from the lead vehicle. This is done to compensate for differences among the vehicle categories. For example, consider a truck following a car in a platoon. Normally the car would have a much faster response to a change in the speed setpoint than a truck, so the truck would not be able to follow the speed pattern of the car adequately. Therefore, the speed synchronizer is designed to delay the speed response of those vehicles which have a high power/weight ratio.

For a more detailed description, the reader is referred to the article of Frank et al. (1989) and to the Appendix.

Frank et al. give rather detailed information about the model used, but not all aspects are fully described. For example, the parameter values of the speed synchronizer and the space controller are not specified. Such missing details were filled in by using various alternatives, comparing the simulation results, and selecting the best alternative.

2.2 Implementation

All simulations have been carried out using PSI/e, version 1.02, and PSI/e's preprocessor (Van den Bosch et al., 1990). PSI/e is a block-structured simulation program for studying the behaviour of dynamic systems. The preprocessor allows the use of structured models, including submodels and local variables. Its input consist of an ASCII file with the main model, which is written in a Pascal-like format. Each submodel is defined in a so-called macro; after a macro has been defined, it can be used in other parts of the model. The Preprocessor translates its input file into a normal PSI/e model. Since PSI/e itself does not use local variables, these are automatically renamed by the Preprocessor using extensions such as '_a', '_b_d', etc.

The Preprocessor model which was used for the simulations in this report is listed in the Appendix. It is structured as follows. A general vehicle model is defined in a macro named 'Veh'. It contains the vehicle dynamics, external disturbances, the space controller, and the speed synchronizer. Several PID speed regulators have been implemented in separate sub-models (macros); in the macro 'Veh' a call to one of them is included. Input signals are the actual following distance and the signal received by the communication device; output signals are the position, speed, and the signal to be transmitted by the communication system.

One level higher, three vehicle types are defined in the macros VehA, VehB, and VehC. In these macro's, vehicle parameters are set at their appropriate values, after which the general macro 'Veh' is called using these parameters. At the highest level, i.e. in the main body, the initial state of the platoon is defined. Each vehicle in the platoon is initialized by calling one of macros VehA, VehB

and VehC with appropriate parameters (initial speed and position). At the end of the file, some PSI/e commands are added between {} to define timing and scaling parameters, and to start the actual simulation.

3 RESULTS

In this chapter, the simulation results will be discussed. First, the behaviour of individual vehicles has been studied to test two versions of the speed regulator and to find suitable parameter values for the speed synchronizer. Second, simulations of platoons have been carried out; the results also served to optimize the space controllers.

Unless mentioned otherwise, the speed setpoint was 25 m/s and the spacing setpoint was 25 m, and the vehicles were normally initialized in a steady state situation (i.e. speeds and following distances equal to their setpoints). All platoons studied were 8 vehicles long, with an A-type vehicle as the lead vehicle, followed by B, C, A, B, C, A, and finally a B type vehicle.

3.1 **Individual vehicles**

Speed regulator

Initially, the behaviour of individual vehicles to external disturbances and to setpoint changes has been studied (i.e. regulator behaviour and servo behaviour, respectively). In these cases the space control loop and the speed synchronizer were not used: only the closed loop behaviour of the speed regulator and vehicle dynamics was studied.
Two implementations for the speed controller have been compared. The first is a parallel PID controller with a transfer function according to Eq. 1, as suggested by Frank et al. (1989).

$$H(s) = \frac{U(s)}{V_{err}(s)} = K_p + \frac{K_i}{s} + K_d s \qquad (1)$$

U(s) is the output signal of the controller, $V_{err}(s)$ is the speed error, defined as the speed setpoint minus the actual speed, and s is the Laplace operator.
The parameters K_p, K_i and K_d are expressed as a function of vehicle parameters and of the speed setpoint, so that the vehicle dynamics are partially cancelled. Since they are a function of the speed setpoint, they can vary in time.

As a simpler alternative, a serial PID controller with fixed parameters K_p, τ_i and τ_d has been tested, with transfer function Eq. 2:

$$H(s) = \frac{U(s)}{V_{err}(s)} = K_p\left(\frac{s\tau_d+1}{0.1s\tau_d+1}\right)\left(1 + \frac{1}{s\tau_i}\right) \qquad (2)$$

The parameters for this controller have been determined separately for each vehicle type, using the method of Ziegler and Nichols (see for instance Grabbe et al., 1958); the resulting values can be found in Table II.

Table II Controller parameter values for each vehicle type.

name	parameter	A	B	C
K_p	Gain fixed PID	2.4	2.46	3.24
τ_i	Integration time constant fixed PID (s)	1.76	2.50	3.34
τ_d	Differentiation time constant fixed PID (s)	0.44	0.625	0.835
T_3	Time constant speed synchronizer (s)	1.5	0.05	0.05
K_x	gain space controller	0.2	0.2	0.2

Because of the non-linear character of the vehicle model, in particular the power saturation, some elements must be added to the PID controllers (1) and (2) to prevent so-called reset wind-up. This would occur after a large step in the speed setpoint is made, resulting in a large speed error and thus to a large controller output u. However, due to the power saturation, the vehicle's acceleration is not proportional to u, but cut off at a certain value, and the speed would only slowly approach the new setpoint. While this is going on, the speed error will cause the integrator (I-part) in the PID-controller to produce a larger and larger value. By the time the error reaches zero, the large integrator output will cause excessive overshoot.
To avoid this, the controller's output is limited to values between u_{max} (positive) and u_{min} (negative) which prevents the engine output from saturating. Besides this, the integration is halted as long as the controller exceeds one of the limits.

Anti-reset wind-up is also used in the model of Frank et al. (1989), but their description is not entirely clear on this point. Therefore, the current implementation is possibly somewhat deviating from their approach.

The behaviour of the two PID controllers has been compared for each of the 3 vehicle types, using a road incline of 3° at t=0 as a disturbance signal. The resulting responses are shown in Fig. 2.

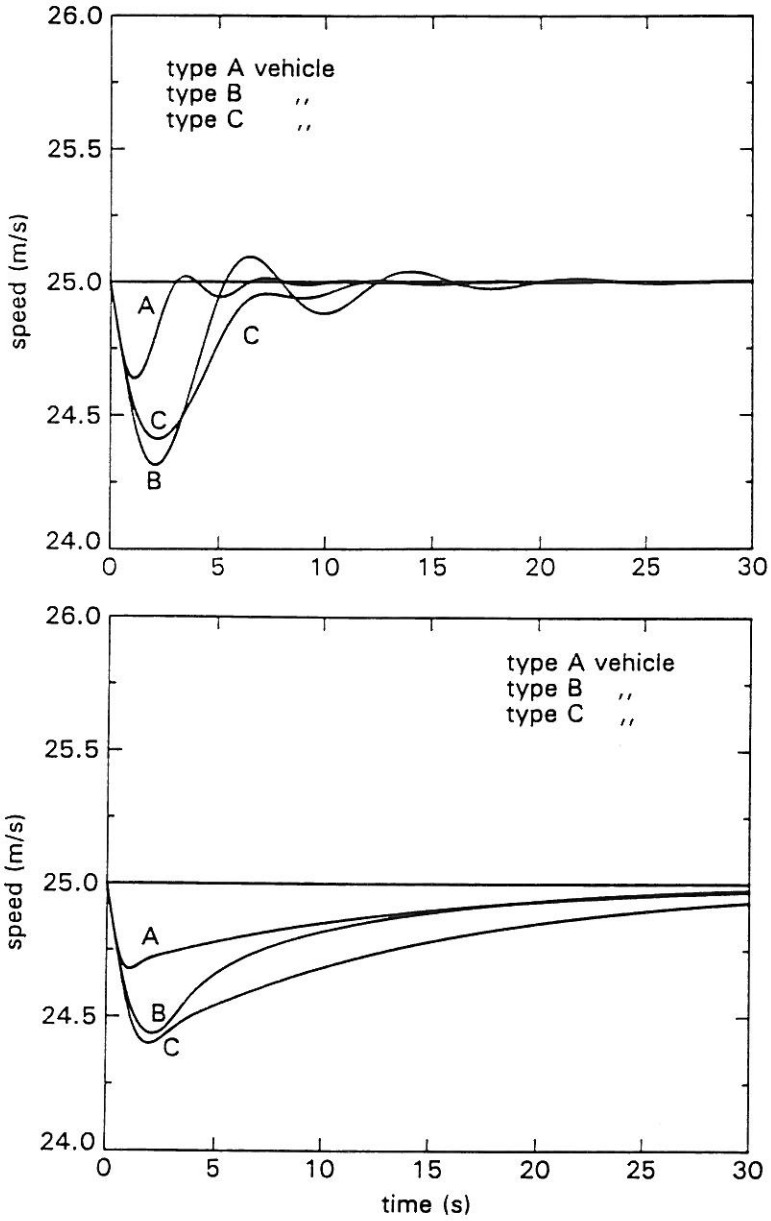

Fig. 2 Responses of individual vehicles to a road incline. **Top:** fixed PID controller; **bottom:** variable PID controller.

When comparing the three vehicle types, it appears that the disturbance suppression is better for vehicles with higher power/weight ratios. Both the overshoot and the settling time increase when power-weight ratio decreases. Since the main purpose of the speed controller is to suppress external disturbances, the fixed PID controller would be preferred based on these results: the overshoots of both controllers are of the same magnitude, but the fixed PID controller results in much faster settling times.

The responses to a small step in the speed setpoint are shown in Fig. 3 for both controllers. It appears that the fixed controller produces more oscillatory transient responses, which will show to be a disadvantage when the cars are placed in a platoon (see § 3.2).

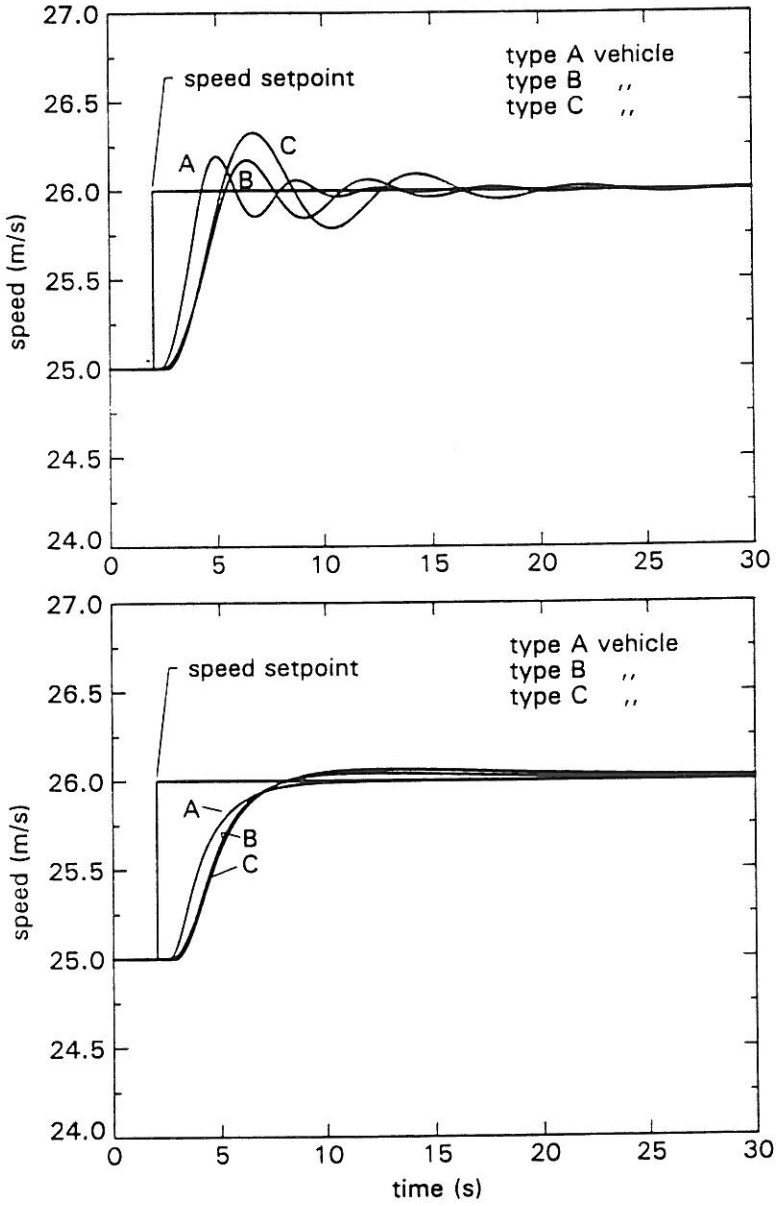

Fig. 3 Responses of individual vehicles to a step of 1 m/s in the speed setpoint. **Top:** fixed PID controller; **bottom:** variable PID controller.

Speed synchronizer

The aim of the speed synchronizer is to delay the speed response of vehicles with a high power/weight ratio. It consists of a first order element with unity gain and time constant T_3. In the article of Frank et al. the value of T_3 is not

been specified, so simulation results have been used to find suitable values. As can be seen in Fig. 3, the step responses of vehicle types B and C are about equally fast, but the car's (type A) is faster. Therefore, the T_3 values for B- and C-type vehicles were both selected very small (0.05 s), whereas for A-type vehicles a value of 1.5 s was used to obtain an optimal fit of the vehicle A response with the two other responses. The three resulting responses are shown in Fig. 4; they match better than in Fig. 3.

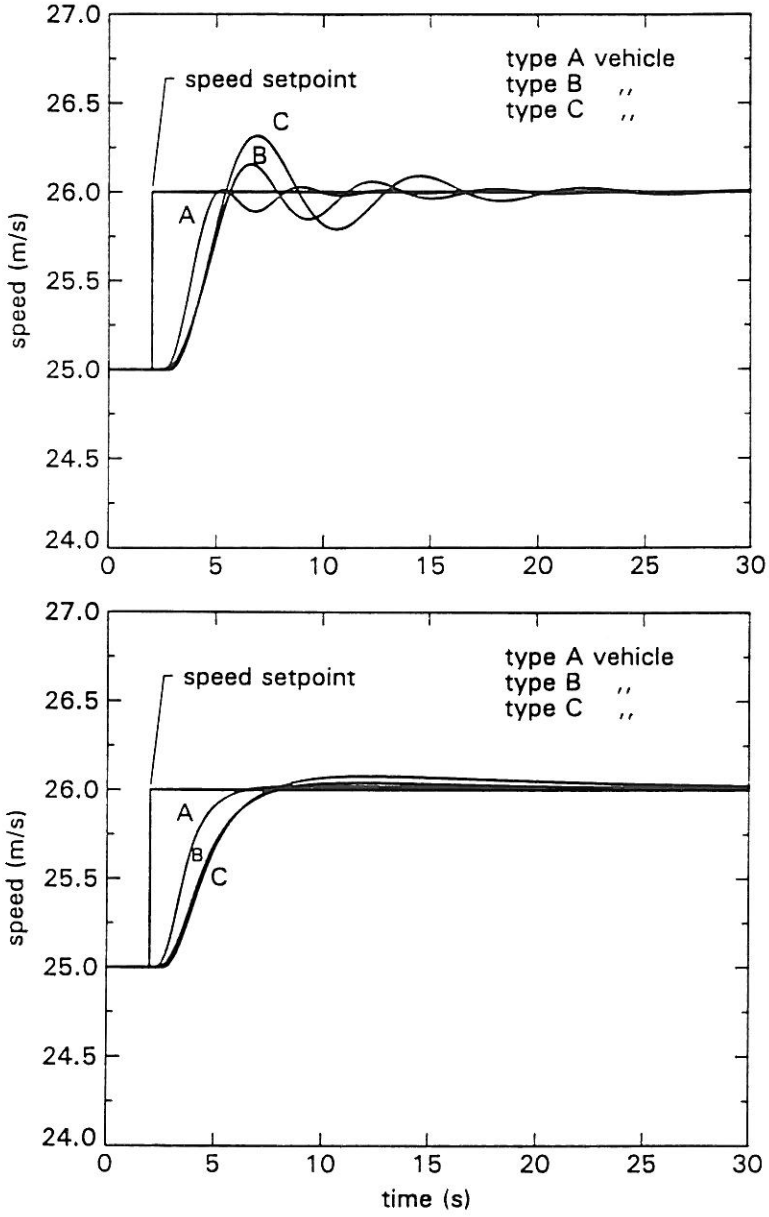

Fig. 4 Responses of individual vehicles to a step in the speed setpoint when using the speed synchronizer. **Top:** fixed PID controller; **bottom:** variable PID controller.

3.2 Platoons of vehicles

The execution time when simulating a platoon of 8 vehicles is approximately 0.9 * real time (PC, 80486 microprocessor). This is determined by the number of blocks needed to simulate one vehicle (over 40), and by the integration time, which must be small (0.01 s) because the D action of the PID controller has a small time constant (0.04 s).

The platoon length has been limited to 8 vehicles because the number of dead time blocks in a PSI/e model cannot exceed 25. Since three such blocks are needed in one vehicle, the maximum number of vehicles which can be simulated simultaneously equals 8.

Space controller

A parameter value K_x for the P-type space controller is not given by Frank et al. Therefore, simulation results were used to find a suitable value. Too large a value results in excessive overshoot or even instability, whereas too small a value causes the spacing to approach its setpoint very slowly. A parameter value of 0.2 appeared to be a good compromise.

External disturbances

The effect of an uphill road incline on the platoon behaviour has been simulated. A slope of 3° was used, starting 25 m ahead of the initial position of the first vehicle. The resulting speed responses are shown in Figs 5 and 6 using the fixed and the variable PID controller, respectively.

When the first few vehicles encounter the road incline, their speeds suddenly drop below the setpoint. This causes a reduction of the following distances of the following vehicles, which forces them to reduce their speeds even before they arrive at the incline. The regulator behaviour of the entire platoon is better when using the variable PID controller: the overshoots are lower and the damping is better, especially for the vehicles in the back of the platoon. The oscillatory behaviour of the fixed controller, which was already noticed for individual vehicles in Fig. 2 (top), tends to worsen when platoon length increases.

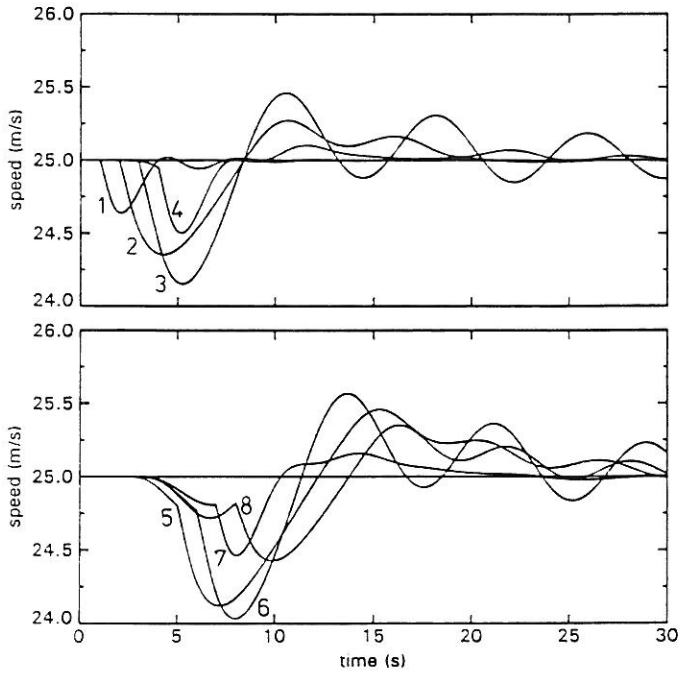

Fig. 5 Responses of a platoon to a road incline (fixed PID controller). **Top:** vehicle 1 to 4; **bottom:** vehicle 5 to 8.

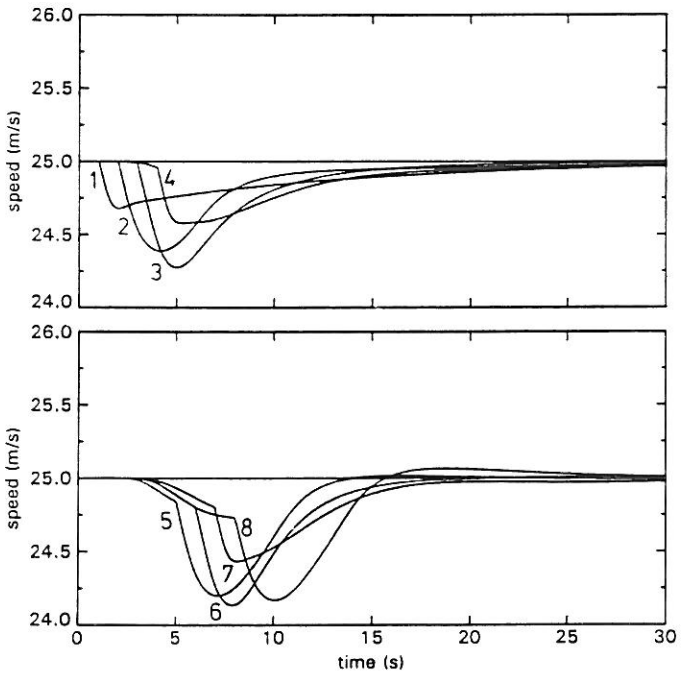

Fig. 6 Responses of a platoon to a road incline (variable PID controller). **Top:** vehicle 1 to 4; **bottom:** vehicle 5 to 8.

Speed decrease

When the first vehicle of a platoon receives a new speed setpoint from the environment, it will accelerate or decelerate accordingly. It will also send its own speed setpoint to the following car, and so on. The effect of a decrease of the platoon's speed setpoint from 25 to 20 m/s is shown in Figs 7 and 8, using fixed and variable PID controllers respectively. It appears that the platoon with variable controllers produces much smoother transient responses: less oscillations occur and damping is better. The oscillations which occur when using the fixed PID controller are slightly amplified from one vehicle to its follower, which will lead to instability ("crashes") if the platoon is made long enough.

Merge into the platoon

To examine the platoon behaviour after one vehicle has just entered it, the following initial conditions were used: all speeds 25 m/s; distance between vehicles 1 and 2, and between 2 and 3 12.5 m; all other distances 25 m. This simulates the case when vehicle 2 has just merged into the platoon between vehicles 1 and 3. The responses of speed and following distance are shown in Figs 9 and 10. The first vehicle in the platoon maintains a constant speed of 25 m/s, whereas the second vehicle temporarily reduces its speed in order to increase the following distance from 12.5 to 25 m. The third vehicle must reduce its speed even further: firstly, its initial following distance is too small, and secondly, the vehicle 1 decreases its speed for a while. The 5 remaining vehicles have comparable responses.

Exit from the platoon

The opposite from the previous simulation scenario is when the vehicle after the lead vehicle has just left the platoon. This gives the following initial conditions: all speeds are 25 m/s, the distance between the first and the second vehicle is 50 m, and all other following distances are 25 m. The resulting step responses are shown in Figs 11 and 12 for speed and following distance, respectively. Again, the first car maintains a speed of 25 m/s, but the second car must temporarily increase its speed to reduce its following distance from 50 to 25 m. All succeeding vehicles also accelerate in order to keep their following distances at 25 m when the second vehicle increases its speed.

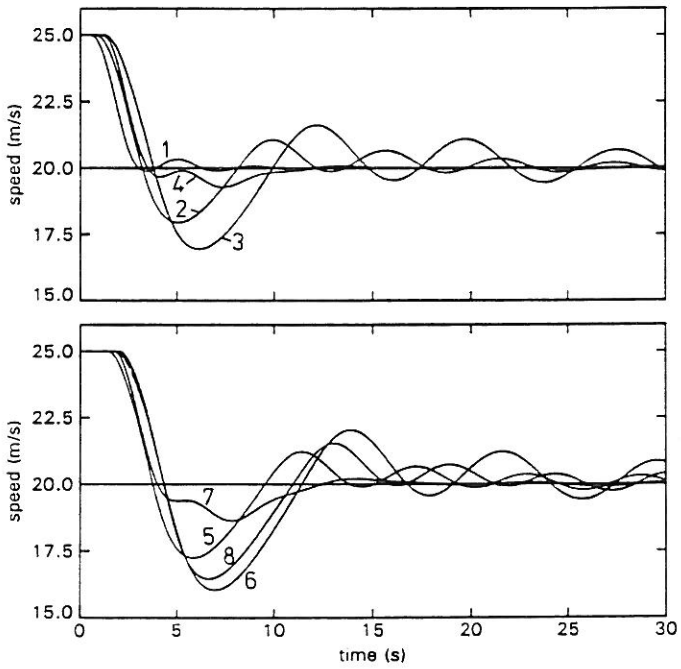

Fig. 7 Responses of a platoon to a change in speed setpoint (fixed PID controller). **Top:** vehicle 1 to 4; **bottom:** vehicle 5 to 8.

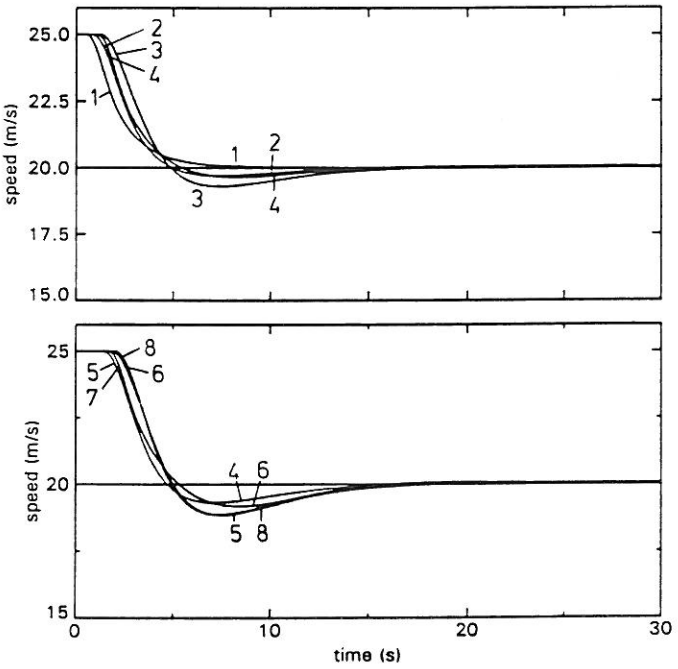

Fig. 8 Responses of a platoon to a change in speed setpoint (variable PID controller). **Top:** vehicle 1 to 4; **bottom:** vehicle 5 to 8.

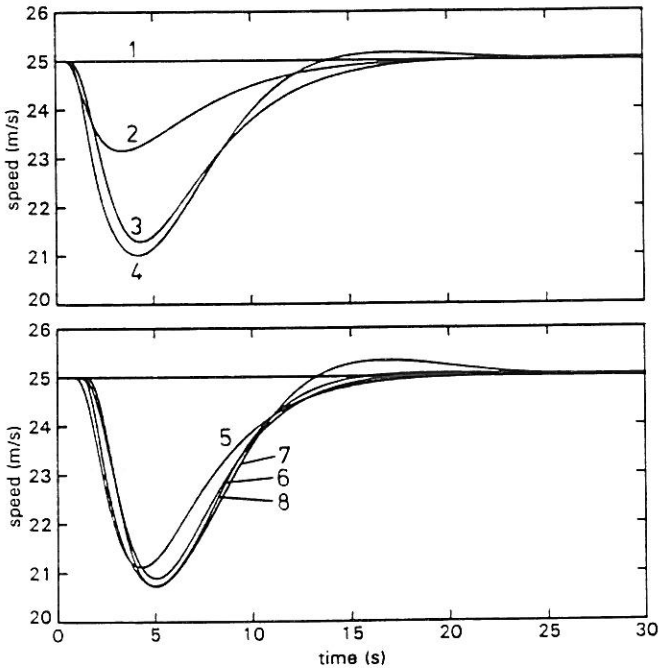

Fig. 9 Speed responses of a platoon to a merge situation. **Top:** vehicle 1 to 4; **bottom:** vehicle 5 to 8.

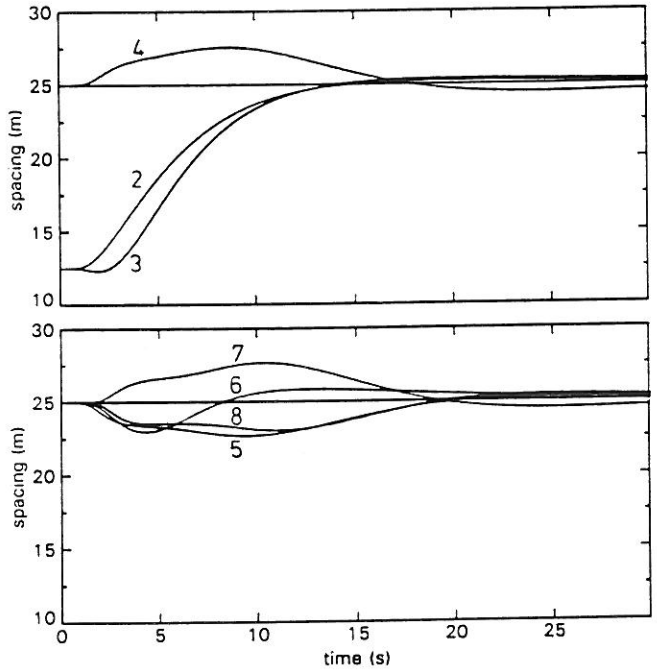

Fig. 10 Distance responses of a platoon to a merge situation. **Top:** vehicle 2 to 4; **bottom:** vehicle 5 to 8.

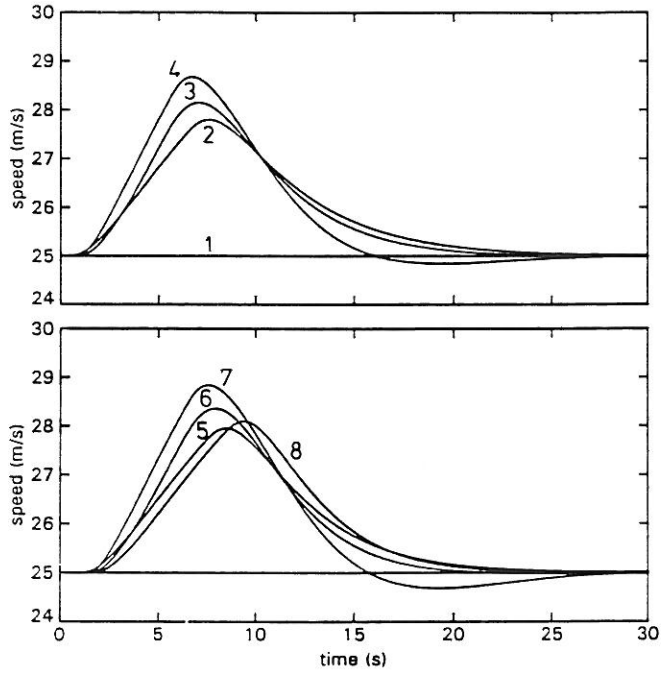

Fig. 11 Speed responses of a platoon to an exit situation. **Top:** vehicle 1 to 4; **bottom:** vehicle 5 to 8.

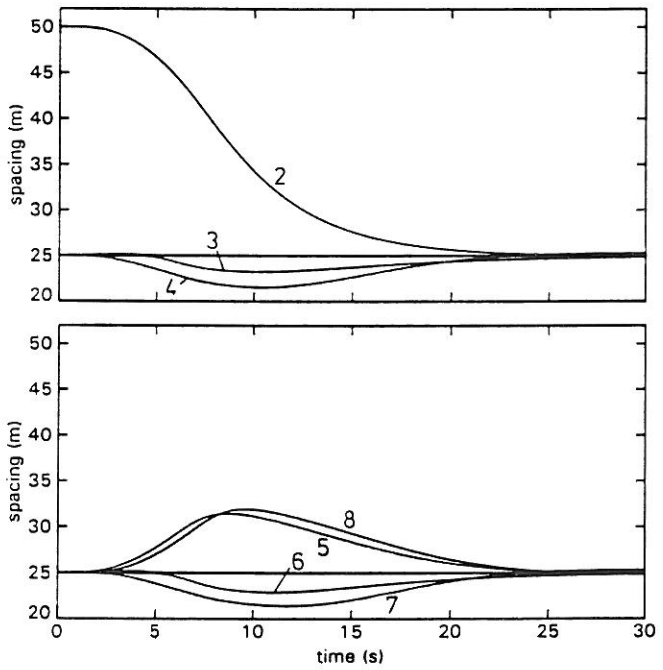

Fig. 12 Distance responses of a platoon to an exit situation. **Top:** vehicle 2 to 4; **bottom:** vehicle 5 to 8.

4 DISCUSSION AND CONCLUSIONS

Implementing and running the model has provided useful experience with simulations of platoons of vehicles equipped with an ICC system. The simulation program PSI/e, together with the preprocessor, has proven to be a useful tool for such simulations, especially because of the possibility to use submodels. To obtain a platoon in a simulation model, one first has to define a vehicle model in a so-called macro. After that, this macro can be used more often to define a platoon of vehicles. Simulation results can be shown on screen, or exported to ASCII files. A restriction is the fact that only 25 delay elements can be used in one PSI/e model. Since 3 such elements are presently needed in each vehicle model, this limits the maximum platoon length in one PSI/e model to 8. However, this limitation can be circumvented by using a separate PSI/e model for each individual vehicle model. Data transfer between vehicles would then have to be realized through files.

The model used is based on the paper of Frank et al. (1989), but a number of non-specified details had to be filled in, so the models are not identical. Furthermore, in the current simulations the variable PID speed controller as proposed by Frank et al. (with speed dependant parameters) has been compared with a fixed PID speed controller, tuned with the method of Ziegler and Nichols. When looking at the disturbance suppression of an individual vehicle, the fixed PID controller would be preferred. But the fixed controller's performance is not adequate for use in a platoon because the responses become too oscillatory. Even though the variable PID-controller performs better, it does not provide global platoon stability: it produces overshoot, and the peak value of the step response becomes larger for vehicles further back in the platoon. Therefore, a necessary condition for the platoon stability is not fulfilled (Cremer, 1992). If the platoon is made too long, a small disturbance at the beginning of the platoon will be amplified by the following vehicles, and finally lead to "crashes". This is in correspondence with the conclusion of Frank et al., who state that in order to maintain a stable platoon, its length should be limited to approximately 15 vehicles. Longer stable platoons would be possible by assigning larger headways, as suggested by Frank et al. However, if headways are selected larger than the present value of 1 s, platooning will not be an effective measure for increasing highway capacity: headways of 2 s can easily be maintained by human drivers (Colbourn et al., 1978).

In short, the simulation results show that the control system with the variable PID controller meets its qualitative requirements: environmental disturbances are well suppressed, whereas entrainment manoeuvres are performed smoothly. This from of ICC, using a combination of in-vehicle distance sensors and vehicle-vehicle communication, results in locally, but not globally, stable platoons.

A number of remarks can be made with respect to the model used by Frank et al. and in this report. To start with, the dynamics of the vehicle are symmetrical

with regard to braking and accelerating at present. So the delay and the time constant which were meant to model the engine and transmission system, are also used for decelerating the vehicle. An actual brake system is not distinguished, which does not seem very realistic.
The distance sensor, for which a radar or a similar sensor is assumed, is simply modelled as a delay element of 0.1 s. However, one cannot expect to measure the exact following distance: it seems sensible to add measurement noise, and to take the limited resolution of such devices into account. A logical next step would be to add a filter to reduce the noise; in the simplest case this would be a first order filter.
Furthermore, the current constant spacing policy is rather limited. It would make more sense to adopt a constant headway policy instead. Such refinements would be easy to implement in the existing model.
The current model can also be used to investigate the effect of other ICC control structures on the platoon behaviour. For example, one could introduce vehicle-roadside communication in the model and evaluate further improvement of platoon stability.

The current PSI/e model constitutes a flexible testbed for possible further simulations. One can implement the mentioned refinements, try alternative control structures, and so on. Then one can easily study the effect of these changes on the dynamical behaviour of separate vehicles and of the entire platoon.

REFERENCES

Almquist, S., Hydén, Ch. & Risser, R. (1992). Use of Speed Limiters in Cars for Increased Safety and a Better Environment. Transportation Research Record 1318, 34-39. Washington, DC: Transportation Research Board.

Bosch, P.P.J. van den, Butler, H. & Soeterboek, A.R.M. (1990). Modeling and simulation with PSI/e. Pijnacker: BOZA automatisering, The Netherlands.

Chang, K.S., Li, W., Devlin, P., Shaikhbahai, A., Varaiya, P., Hedrick, J.K., McMahon, D., Narendran, V. & Swaroop, D. (1991). Experimentation with a Vehicle Platoon Control System. In: Proceedings 2nd Vehicle Navigation & Information Systems Conference, Part 2, 1117-1124. Warrendale, PA: Society of Automotive Engineers, Inc.

Colbourn, C.J., Brown, I.D. & Copeman, A.K. (1978). Drivers' judgment of safe distances in vehicle following. Human Factors 20(1), 1-11.

Cremer, M. (1992). On convoy-stable control laws for automatically driven vehicle clusters. Proceedings of ISATA conference on road transport informatics/intelligent vehicle highway systems, Florence, Italy. 920051, pp. 253-260.

Eliasson, A. (1992). A controller for autonomous intelligent cruise control - a preliminary design. In: Proceedings 3rd International Conference on Vehicle Navigation & Information Systems. IEEE, 170-175.

Frank, A.A., Liu, S.J. & Liang, S.C. (1989). Longitudinal Control Concepts for Automated Automobiles and Trucks Operating on a Cooperative Highway. SAE Transactions 1989, Journal of Passenger Cars, Vol. 98 Section 6.

Grabbe, E.M., Ramo, S. & Wooldridge, D.E. (Eds) (1958). Handbook of automation and control. Vol. 1. New York: John Wiley & Sons.

Horst, A.R.A. van der, Arem, B. van, Hogema, J.H., Janssen, W.H., Kleuskens, R.J.A. & Tol, A.C. van (1993). Randstad^{+} Project Intelligent Traffic Systems: definition phase and workplan. Report IZF 1993 C-2. Soesterberg, NL: TNO Institute for Perception.

Horst, A.R.A. van der (1993). Onderzoeksvoorstel Integration of Roadway and In-car Support and Safety Systems (IRISS). Soesterberg, NL: TNO Institute for Perception.

Hosaka, A. & Taniguchi, M. (1992). The development on advanced vehicle control technologies for autonomous driving. Paper no. 920309, Transportation Research Board 72nd Annual Meeting, January 1-14, 1993, Washington, DC.

Rao, B.S.Y., Varaiyja, P. & Eskafi, F. (1993). Investigations into Achievable Capacities and Stream Stability with Coordinated Intelligent Vehicles. Paper no. 930803, Transportation Research Board 72nd Annual Meeting, January 1-14, 1993, Washington, DC.

Zhang, X. (1991). Intelligent Driving - PROMETHEUS approaches to longitudinal traffic flow control. In: Proceedings 2nd Vehicle Navigation & Information Systems Conference, Part 2, 990-1010. Warrendale, PA: Society of Automotive Engineers, Inc.

Soesterberg, 8 July 1993

J. Hogema

Ir. J.H. Hogema

APPENDIX Model listing

Listing of the PSI/e Preprocessor model which was used to simulate a platoon of 8 vehicles.

```
DEFMACRO main ()

!- PSI/e Preprop file.
   Contains models of veh's according to Frank, Liu and Liang:
   'Longitudinal Control Concepts for Automated Automobiles...'

   LEAVE THIS FILE IN TACT

   Comment is placed between !!
   Output to screen and to file (FIL001.ASC and so on)

   J.H.Hogema                                                    -!

SIG: x1,  v1,  dx1,  vro1,  vri1
     x2,  v2,  dx2,  vro2,
     x3,  v3,  dx3,  vro3,
     x4,  v4,  dx4,  vro4,
     x5,  v5,  dx5,  vro5,
     x6,  v6,  dx6,  vro6,
     x7,  v7,  dx7,  vro7,
     x8,  v8,  dx8,  vro8,
     out1, out2, out3,
     out4, out5, out6,
     out7, out8, out9,       !-outputs to be shown on screen/written to file-!
     do1,  do2,  do3,
     do4,  do5,  do6,
     do7,  do8,  do9,        !-DAO blocks to write data to ascii-file -!
     do10,
     time, timer, unit;
VAR: pi, g;

!-------------------------------------------------------------------!
!                      SPEED CONTROLLERS                            !
!  One of these is called by macro Veh.                             !
!  Input:  speed error verr                                         !
!  Output: control signal u                                         !
!-------------------------------------------------------------------!
   !--- Serial PID controller ---!
   DEFMACRO PID1 (INP: verr; OUT: u; PAR: kp, taui, taud, u0;)
   SIG: u1;
   CALL PDC (INP: verr; OUT: u1; PAR: 0, 0.1, taud;)
   CALL PIC (INP: u1;   OUT: u;  PAR: u0, kp, taui;)
   ENDMACRO !-PID1-!

   !--- PID controller with limited output
        and limited output of I-action (ARW) -----!
   DEFMACRO PID2 (INP: verr; OUT: u; PAR: kp, taui, taud, umin, umax, u0;)
   SIG: u1, u2, u3;
   u1   = Kp*u3 + u2
   CALL LIM (INP: u1;   OUT: u;  PAR: umin, umax, 1;)
   CALL INL (INP: u3*Kp/taui;
             OUT: u2;
             PAR: u0, umin, umax;)
   CALL PDC (INP: verr; OUT: u3; PAR: 0, 0.1, taud;)
```

```
   ENDMACRO !-PID2-!

   !--- PID controller with limited output
        and ARW: if the controller satturates,
        the I-action is halted.                    ---!
   DEFMACRO PID3 (INP: verr; OUT: u; PAR: kp, taui, taud, umin, umax, u0;)
   SIG: u1, u2, u3, st;
        ! st ('SaTuration')is a boolean: st=1 if controller satturates !
   u1   = Kp*u3 + u2
   CALL LIM (INP: u1; OUT: u; PAR: umin, umax, 1;)
   CALL INC (INP: -1+3*st, u3*kp/taui, 0;
             OUT: u2;
             PAR: u0;)
   CALL PDC (INP: verr; OUT: u3; PAR: 0, 0.1, taud;)
   CALL ZDL (INP: (u1>umax) .OR. (U1<umin);
             OUT: st;
             PAR: 0;)
   ENDMACRO !-PID3-!

   !--- PID controller according to Frank et.al.
        Parameters are a function of vr.      ---!
   DEFMACRO PIDA (INP: verr, vr;
                  OUT: u;
                  PAR: tau, c1, T1, m, ca, umin, umax, u0;)
   SIG: T2, c2, u1, u2, u3, st,
        Ki, Kp, Kd;
        ! st ('SaTuration')is a boolean: st=1 if controller satturates !
   c2 = 1/(2*ca*vr)
   t2 = m*c2
   Ki = 1/(2*tau*c1*c2)
   Kp = Ki*(T1+T2)
   Kd = Ki*T1*T2
   CALL INC (INP: -1+3*st, Ki*verr, 0;
             OUT: u2;
             PAR: u0;)
   CALL ZDL (INP: (u1>umax) .OR. (u1<umin);
             OUT: st;
             PAR: 0;)
   CALL DIF (INP: Kd*verr; OUT: u3; PAR:  0;)
   u1 = Kp*verr + u2 + u3
   CALL LIM (INP: u1; OUT: u; PAR: umin, umax, 1;)
   ENDMACRO !- PIDA -!

!-------------------------------------------------------------------!
!                      VEHICLE DEFINITIONS                          !
! Input: following distance dx                                      !
!        communication link info from lead veh. vri                 !
! Parameters:                                                       !
!        initial position & speed                                   !
!        vehicle and controller parameters                          !
!        lead=1 for first veh. in platoon; lead=0 for all others    !
! Output:                                                           !
!        position, speed, communication link info vro               !
!-------------------------------------------------------------------!

   DEFMACRO Veh (INP: dx, vri;
                 OUT: x, v, vro;
                 PAR: x0, v0, dx0, Lead,
                      Kp, taui, taud, Kx,
                      m, c1, T1, T3, ca,
                      tau1, tau2, tau3,
```

```
                    Fdl, Fdh;)
   SIG: a, u, vw, q,
        verr, vref, vr, vrid,
        dxr, dxe, dxd, crdt,
        F, Fa, Fg, Fd, Fd1, Fd2, Fd3;
   VAR: Fd0, Fa0, Fg0, u0, vw0, q0;

   !------- EXTERNAL DISTURBANCES -------!
   !- vw = wind speed -!
   !- q  = road angle -!
   vw0 = 0
   q0  = 0
   vw  = 0
   q   = 0 !- + 3*(x>360)  -!   !-possibly position dependant angle-!

   !--- INITIALIZATIONS Veh ---!
   Fa0 = Ca * SIGN(v0+vw0) * (v0+vw0)^2
   Fg0 = m*g*SIN(Q0*pi/180)
   Fd0 = Fg0 + Fa0
   u0  = Fd0/c1

   !------- VEHICLE DYNAMICS -----------!
   CALL INT (INP: v;   OUT: x;   PAR: x0;)
   CALL INT (INP: a;   OUT: v;   PAR: v0;)
   CALL INF (INP: u;   OUT: Fd3; PAR: Fd0, C1,  T1;)
   CALL TDE (INP: Fd3; OUT: Fd2; PAR: Fd0, Tau1, 1;)
   CALL LIM (INP: Fd2; OUT: Fd1; PAR: Fdl, Fdh,  1;)

   !- forces on vehicle -!
   F = Fd - Fg - Fa                  ! sum of forces        !
   Fd = Fd1*(v>=0)                   ! drive force          !
   Fg = m*g*sin(Q*pi/180)            ! gravity (road angle) !
   Fa = ca*sign(v+vw)*(v+vw)^2       ! air drag force       !
   a  = F/m                          ! acceleration: Newton !

   !-------- SPEED CONTROLLER -------!
   !- call to one of the defined speed controllers -!
   verr = vref - v
!- verr = vri - v  => using this line will only implement
                     the  speed controller                -!

!-  CALL PID1 (INP: verr; OUT: u; PAR: kp, taui, taud, u0; ext = a) -!
!-  CALL PID2 (INP: verr; OUT: u; PAR: kp, taui, taud,
                                       Fdl/c1, Fdh/c1 ,u0; ext = a)  -!
!-  CALL PID3 (INP: verr; OUT: u; PAR: kp, taui, taud,
                                     Fdl/c1, Fdh/c1, u0; ext = a)    -!
    CALL PIDA (INP: verr, vref;
               OUT: u;
               PAR: tau1, c1, T1, m, ca, Fdl/c1, Fdh/c1, u0;
               ext = a)

   !--------- SPACE CONTROLLER/SPEED SYNCHRONIZER -------!
   dxr = 25  !- reference for spacing dx -!

   CALL INF (INP: vr;  OUT: vref; PAR: v0, 1, T3;)
   CALL TDE (INP: dx;  OUT: dxd;  PAR: dx0, tau2, 1;)
   CALL TDE (INP: vri; OUT: vrid; PAR: v0,  tau3, 1;)
   CALL STP (INP: -dx*(1-Lead);
             OUT: crdt;
             PAR: 2;)
   !- ^^^^^ CRash DeTector:  IF dx<0 AND NOT Lead THEN Stop -!
```

```
dxe = (dxr-dxd)*(1-Lead)
!-                ^^^^^^ IF Lead=1 THEN dxe=0    -!
vr  = vrid - Kx*dxe

!- Signal to next car: -!
vro = vr

ENDMACRO !Veh!
!-------------------------------------------------------------------!

DEFMACRO VehA (INP: dx, vri;
               OUT: x, v, vro;
               PAR: x0, v0, dx0, lead;)
VAR: Kp, taui, taud, Kx,
     m, c1, T1, T3, ca, tau1, tau2, tau3,
     Fdl, Fdh;

!--- INITIALIZATIONS VehA ---!
m   = 750          !- vehicle mass (kg)              -!
c1  = 743          !- driving coefficient            -!
T1  = 1.0          !- time const. vehicle propulsion -!
ca  = 1.19         !- air drag constant              -!
tau1= 0.3          !- delay time for actuator        -!
tau2= 0.1          !- delay time for sensor          -!
tau3= 0.2          !- delay time communication link  -!
Fdl = -0.4*g*m     !- max. braking force (<0)        -!
Fdh =  0.2*g*m     !- max. propulsion force          -!

kp   = 2.4         !- PID gain constant              -!
taui = 1.76        !- PID integration time           -!
taud = 0.44        !- PID differentiation time       -!
T3   = 1.5         !- speed synchronizer time const. -!
Kx   = 0.2         !- space controller gain          -!

!- Now call the macro Veh with the actual parameters -!
CALL Veh (INP: dx, vri;
          OUT: x, v, vro;
          PAR: x0, v0, dx0, Lead,
               Kp, taui, taud, Kx,
               m, c1, T1, T3, ca,
               tau1, tau2, tau3,
               Fdl, Fdh;
           ext = a)

ENDMACRO !VehA!
!-------------------------------------------------------------------!

DEFMACRO VehB (INP: dx, vri;
               OUT: x, v, vro;
               PAR: x0, v0, dx0, lead;)
VAR: Kp, taui, taud, Kx,
     m, c1, T1, T3, ca, tau1, tau2, tau3,
     Fdl, Fdh;

!--- INITIALIZATIONS VehB ---!
m   = 1000          !- see under macro VehA for meaning of parameters -!
c1  = 743
T1  = 1.738
ca  = 2.0
tau1= 0.4
tau2= 0.1
```

```
    tau3= 0.2
    Fdl = -0.4*g*m
    Fdh =  0.2*g*m

    kp   = 2.46
    taui = 2.50
    taud = 0.625
    T3   = 0.05
    Kx   = 0.2

    CALL Veh (INP: dx, vri;
              OUT: x, v, vro;
              PAR: x0, v0, dx0, Lead,
                   Kp, taui, taud, Kx,
                   m, c1, T1, T3, ca,
                   tau1, tau2, tau3,
                   Fdl, Fdh;
               ext = b)

    ENDMACRO !VehB!
    !-------------------------------------------------------------------!

    DEFMACRO VehC (INP: dx, vri;
                   OUT: x, v, vro;
                   PAR: x0, v0, dx0, lead;)
    VAR: Kp, taui, taud, Kx,
         m, c1, T1, T3, ca, tau1, tau2, tau3,
         Fdl, Fdh;

    !--- INITIALIZATIONS VehC ---!
    m    = 2000          !- See under macro VehA for meaning of parameters -!
    c1   = 743
    T1   = 2.0
    ca   = 3.0
    tau1= 0.6
    tau2= 0.1
    tau3= 0.2
    Fdl = -0.4*g*m
    Fdh =  0.2*g*m

    kp   = 3.24
    taui = 3.34
    taud = 0.835
    T3   = 0.05
    Kx   = 0.2

    CALL Veh (INP: dx, vri;
              OUT: x, v, vro;
              PAR: x0, v0, dx0, Lead,
                   Kp, taui, taud, Kx,
                   m, c1, T1, T3, ca,
                   tau1, tau2, tau3,
                   Fdl, Fdh;
               ext = c)

    ENDMACRO !VehC!

!-----------------------------------------------------------------------!
!                            MAIN BODY                                  !
!-----------------------------------------------------------------------!
```

```
pi = 3.1415927
g  = 10

unit = 1

!- PLATOON SPEED SETPOINT -!
CALL GEN (INP: 1, 15, 25; OUT: vri1; PAR: 25, 1, 4;)
!- vri1 = 25  this line will maintain a constant speed setpoint-!

!- Following distances -!
dx1  =   0
dx2  =  x1  - x2
dx3  =  x2  - x3
dx4  =  x3  - x4
dx5  =  x4  - x5
dx6  =  x5  - x6
dx7  =  x6  - x7
dx8  =  x7  - x8

!- 3 separate veh's: -!
!--
CALL VehA (INP: dx1, vri1;
           OUT: x1, v1, vro1;
           PAR: /x0=/ 0, /v0=/25, /dx0=/0, /Lead=/1; ext = a)

CALL VehB (INP: dx2, vri1;
           OUT: x2, v2, vro2;
           PAR:  0, 25, 0, 1; ext = b)

CALL VehC (INP: dx3, vri1;
           OUT: x3, v3, vro3;
           PAR:  0, 25, 0, 1; ext = c)            --!


!- Platoon of 8 veh's:  ABCABCAB -!
!- initial speed 25 m/s          -!
!- initial spacing 25 m          -!
CALL VehA (INP: dx1, vri1;
           OUT: x1, v1, vro1;
           PAR: !x0=!375, !v0=!25, !dx0=!0, !Lead=!1; ext = a)

CALL VehB (INP: dx2, vro1;
           OUT: x2, v2, vro2;
           PAR:  350.0, 25, 25.0, 0; ext = b)

CALL VehC (INP: dx3, vro2;
           OUT: x3, v3, vro3;
           PAR:  325, 25, 25.0, 0; ext = c)

CALL VehA (INP: dx4, vro3;
           OUT: x4, v4, vro4;
           PAR: 300, 25, 25, 0; ext = d)

CALL VehB (INP: dx5, vro4;
           OUT: x5, v5, vro5;
           PAR: 275, 25, 25, 0; ext = e)

CALL VehC (INP: dx6, vro5;
           OUT: x6, v6, vro6;
           PAR: 250, 25, 25, 0; ext = f)
```

```
CALL VehA (INP: dx7, vro6;
           OUT: x7, v7, vro7;
           PAR: 225, 25, 25, 0; ext = g)

CALL VehB (INP: dx8, vro7;
           OUT: x8, v8, vro8;
           PAR: 200, 25, 25, 0; ext = h)

out1 = v1
out2 = v2
out3 = v3
out4 = v4
out5 = v5
out6 = v6
out7 = v7
out8 = v8
out9 = vri1

!- define timer #2 for dao-blocks: -!
CALL tim (INP: 1;         OUT: timer;PAR:  2;)

!- dao blocks generate output file FILxxx.ASC     -!
!- xxx = 001, 002 etc, defined by PAR             -!
!- the Pascal program 'combine' can combine these -!
!- files to generate QuickPlot files              -!
CALL dao (INP: timer, time;   OUT: do10; PAR: 10;)
CALL dao (INP: timer, out1;   OUT: do1;  PAR:  1;)
CALL dao (INP: timer, out2;   OUT: do2;  PAR:  2;)
CALL dao (INP: timer, out3;   OUT: do3;  PAR:  3;)
CALL dao (INP: timer, out4;   OUT: do4;  PAR:  4;)
CALL dao (INP: timer, out5;   OUT: do5;  PAR:  5;)
CALL dao (INP: timer, out6;   OUT: do6;  PAR:  6;)
CALL dao (INP: timer, out7;   OUT: do7;  PAR:  7;)
CALL dao (INP: timer, out8;   OUT: do8;  PAR:  8;)
CALL dao (INP: timer, out9;   OUT: do9;  PAR:  9;)

!- define output signals on screen -!
CALL dis (INP: out1;   PAR: 24, 26;)
CALL dis (INP: out2;   PAR: 24, 26;)
CALL dis (INP: out3;   PAR: 24, 26;)
CALL dis (INP: out4;   PAR: 24, 26;)
CALL dis (INP: out5;   PAR: 24, 26;)
CALL dis (INP: out9;   PAR: 24, 26;)


!-----------------------------------------!
!       ADDITIONAL PSI/e  COMMANDS        !
! t     = initialize timing:              !
!         step size and                   !
!         end of simulation.              !
! tim0  = integration time                !
! tim1  = used for dead time in veh's     !
! tim2  = timer for dao                   !
! dsa   = define scaling                  !
! cm    = cange markers                   !
! r     = run simulation                  !
! close = close files created by dao-blocks !
!-----------------------------------------!
{t
0.01
30
```

```
dtim0 0.01
dtim1 1
dtim2 10
dsa 24,27
cm
r
close
}
ENDMACRO !MAIN!
```