

Simulate Highway Traffic

✉ TO ALL STUDENTS OF ANU/CECS/SoCS/COMP4330, 6433, AND 8140

Abstract – Design and test of a simulator for a (distributed) real-time traffic movements on a single-lane road in dense traffic – this is the single, marked assignment.

1. Overview

The movements of cars in dense traffic on a single lane road appears simple as a real-world/real-time scenario. Nevertheless it is not only more complex than it appears at first glance, but it is also quite relevant to understand the actual reasons for mass accidents on highways. Only in July this year 259 cars piled up in a single accident on a German highway (nobody was killed).

2. Sensors, actuators, communication

In order to stay convincing no underlying common time-base is assumed. Cars (i.e. their drivers) are acting in their own limited view of the world and specifically in their own time-scope. While considering delays in the sensing the single-most important measurement is the distance to the car in front, which is available with a certain delay and timing granularity (it is assumed that the driver is bound by limitations in terms of reaction times, precision, and sampling rates). As in a real car, acceleration and declaration can be controlled inside the physical limits of the car.

There will be three classes of drivers to be considered: the average driver, the tail-gater, and the inattentive driver. for your simulation you can set these three types (and their relative percentages) up in multiple ways to be able to measure the effect.

3. Measurement criteria

Your simulation should provide you with the tools to experimentally find answers to the following questions:

- At which average speed will you experience the maximal throughput? (assuming average drivers only.)
- How does the percentage of tail-gaters and inattentive drivers affect the probability for accidents?

- How does the average speed relate to the number of cars involved in an accident?
- Based on your simulations: what would you recommend as a good rule of thumb distance between cars on Australian highways?

As an additional outlook you might want to think about the requirements for an automated car following system which can be practically deployed in dense traffic?

4. Design constraints

Avoid unnecessary synchronization (and therefore potential blocking) between individual car instances. Make a convincing case that your sensing their environments without implicit synchronisation (and therefore without unrealistic additional information gain). Still, some information need to be exchanged between cars and their environments – yet try to model physics and existing sensing channels as close as possible.

5. Real-time constraints

While it is hard to simulate completely independent units on a sequential or partly parallel hardware, you can exploit the difference between assumed human reaction and sensing times and the execution times on modern computing hardware. Estimate which blocking and delay limits you would consider undetectable for a human driver and make sure that your design is guaranteed to stay in those limits (the amateurish, but classic argument “give me a fast enough computer and I solve all real-time tasks” does obviously not get you there).

6. Optimization goal

Answer the questions in section 3, while being able to prove that your simulation stays realistic up to a certain number of cars. There is a trade-off between realism, and number of cars which you can guarantee. Optimise for a large number of cars while staying fully dependable and reasonably realistic.

Also demonstrate (by ways of reasoning) how you system scales with larger numbers of concurrent processors.

7. Simulator

The simulator has a core-design component, which you will use to argue the dependability of your system. But also a test component/interface which you will need to create and use to measure and test your system. The core issue is that you would wish for an interface which can display fully concurrent developments, while keeping the overall programming effort low. A fully graphically interface is an obvious solution, but comes at a cost (your time). A simple post-mortem-dump text output is problematic in its expressiveness in terms of asynchronous events. A set of observers which measure the essential feature for you while being embedded into the system in an unobtrusive way, might be a middle way. Take your choice.

8. Deliverables

- a. List the physical and timing constraints which you will implement truthfully.
- b. Document your design decisions. Clever thoughts in the design process do not guarantee a working solution, but will be honoured in the marking. Give specific care to the reasoning about the timing behaviour of your system.
- c. Provide explicit documentation for your test runs (in order to answer the given questions) and give reasons why those test runs show what we want to find out.
- d. Can you suggest parts or your whole system for strict verification/certification? which ones?

You are not bound to a specific design methodology, but need to deliver a convincing result. Express your ideas by whichever means necessary -> choose the right tools for your documentation.

The documentation need to be submitted at least three days before the lab-exams. Overall assignment time is six weeks. Due date according to the website.