
Generate Highway Traffic Entities

to all students of ANU/FEIT/DCS/COMP4330

This a preparing assignment for things to come. The more your designs are thought through in the early stages, the easier the later parts of the assignment will become for you.

1. Outlook

This semester we will set up a distributed highway traffic flow control system (Cruise control 2.0). The major constraint for the exercise is that it needs to be implemented in a distributed fashion to make a believable point that your system can be deployed in individual cars without a central infrastructure but only embedded into local communication systems connecting your car to its local sensors as well as sensing the range to the car in front of you.

While this part will be easy, you also need to write a simulator to test how identical, similar, and very different control modules play together and produce an efficient and smooth traffic flow or one large traffic jam.

We will later use your system to find out which car-throughput can be produced at what level of safety on a highway by deploying a distributed convoy system.

2. First assignment part

Before you begin to think about your actual system design, think about how you will test it and evaluate it. You want your simulated cars to move in a physically believable way, so acceleration and deceleration

are limited. Cars will not share a common clock, so every cars needs to create its own clock system, which drives the updates. This can be either a free-running clock or a sensor driven synchronization.

Concrete job: Design data-structures and a task type framework which will allow you to emulate a single car. Simple Newton mechanics are sufficient – no need to emulate the state of the tires. We will need to be able to set driver/system *reaction speeds*, distance sensor *sampling frequency*, sensing *range*, maximal *accelaration* and *decelaration*, as well as the preferred *distance* to the car in front.

Things to consider: you need to make sure that your communication/sensor/actuator systems are not blocking without necessity. On the other hand your system needs to suspend itself most of the time, as you want to keep power consumption of the on-board components at a minimum – and in your concrete case – you want to be able to generate a few hundred of those cars-tasks later in a simulation environment without endangering your overall timing assumptions.

If your concept is clear in your mind, the implementation itself should become more of a quick technical exercise. So spent enough time thinking about your framework before you start programming. If you now notice that your Ada skills are still somehow lacking: this is the last chance in this lecture to seriously brush up your programming skills to be prepared for the tasks ahead.

Enjoy and don't hesitate to ask or request guidance.