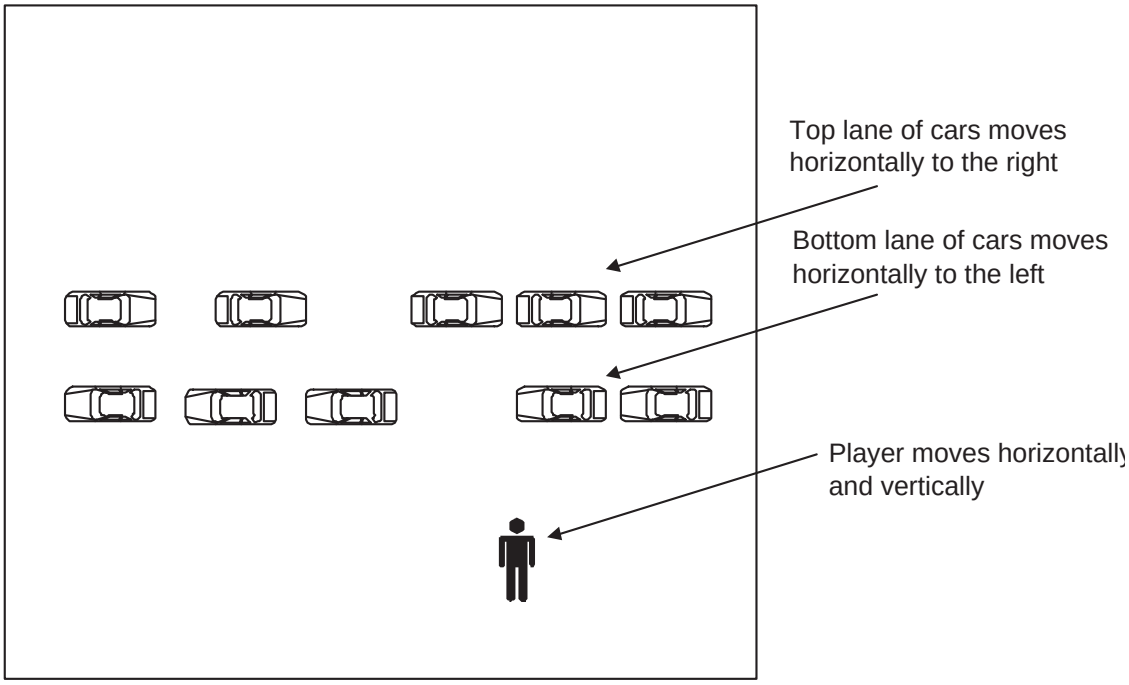
UNIVERSITY OF THE WITWATERSRAND, JOHANNESBURG
School of Electrical and Information Engineering
ELEN7045 - SD Methodologies, Analysis and Design

Individual Assignment I

Software Design and Modifiability: The Two-Way Traffic Game

1 Introduction



The figure above depicts a screen shot from a game called “Two-Way Traffic”.

This game consists of:

- A single player who can move left and right and up and down and is controlled via the keyboard.
- Cars which travel in two lanes. Cars in the top lane only move from left to right, cars in the bottom lane only move from right to left. All cars move at the same speed. The cars appear at random intervals at the start of each lane and disappear when they reach the end of the lane.

The objective of the game is to move the player so that he/she crosses the two lanes of traffic without bumping into a car. If the player successfully crosses both lanes, then he/she is positioned at the bottom of the screen for another attempt. If a car touches the player then the game is over.

2 Deliverables

This is an individual assignment — each student is required to work on their own.

2.1 Executable and Code

Provide a working implementation (executable) of the game along with the complete source code. Your source code needs to be well commented and/or easily understandable. You are expected to apply good object-oriented (OO) design principles in your design which include, among others, creating small classes with cohesive functionality, hiding information, and using inheritance appropriately.

The choice of OO language and graphics library is up to you. Bear in mind the following constraints:

- your solution must be able to run on the Windows platform,
- you may not use an existing game framework.

If you are an experienced developer then take this opportunity to learn a new language. If your development skills are not that strong then choose an OO language that you are familiar with.

You should use *simple* graphics for the game elements. *The aim of this assignment is not to produce a game with sophisticated functionality but rather to create a well-thought-out design for the game which will easily support certain changes in requirements (see below)*.

2.2 Documentation

Produce, and submit, a *hard copy* of a short report (aim for no more than 8 pages, excluding title page, abstract and references) which:

1. presents and critiques your design,
2. discusses your game's modifiability, and
3. has at least three references.

2.2.1 Design

In presenting your design clearly identify the key objects and classes involved, their responsibilities, and how they collaborate with each other to provide the game's functionality. You are also required to examine your design critically and motivate the design choices that you have made.

2.2.2 Modifiability

Modifiability is a quality attribute of a software system which refers to the ease with which the existing design can be modified to meet new requirements. The ideal situation is to meet new requirements with new code — not by re-writing existing code. However, often this ideal cannot be achieved and we try to localise code changes to as few classes or modules as possible. It is important to realise that a design cannot support all changes equally and some changes will require an extensive re-write of the code.

A good way of evaluating the modifiability of a software system is to consider scenarios for future change. In this game, for example, the following requirement changes might occur:

1. The number of traffic lanes increases every time the player successfully crosses the road.
2. The keys which control the player need to be configurable.
3. The game must change from a one player version to a two player version. Both players must simultaneously dodge traffic. The game ends when one of the players is hit.
4. Scoring is introduced and the top three highest scores need to be saved.
5. A slightly different version of the game is needed in which the cars become logs moving down a river and the player becomes a frog. The frog must jump from log to log in order to successfully cross the river.
6. Cars are able to change lanes, that is, move both vertically and horizontally.
7. A different graphics library must be used in place of the existing one.
8. The game must run on Linux.

Which of these futures does your design naturally support and which does it not? Justify your answers by discussing how your source code will have to be modified (note, you are not required to actually implement any of these scenarios). Also list and discuss any additional change scenarios which are supported by your design.

2.2.3 References

It is important to develop an academic writing style, and part-and-parcel of this is the ability to correctly cite and reference existing work that is relevant to your own. To this end, you are required to find and provide three such references.

3 Deadline and Submissions

The deadlines for all the course deliverables are available on the course home pagebreak.

All submissions must be in strict accordance with the guidelines contained in the *School's Blue Book* and the rules contained in the *School's Red Book*. Please note the minor changes with regard to the late submission policy in Section 3.1.

3.1 Late Submission Penalty Policy

Submission are due at the start of the lecture on the deadline day. You may submit the following day before the close of business (approximately 4pm) but you will be penalised by 10 percentage points. No submissions after this will be accepted.

3.2 Plagiarism

Refer to the *School's Blue Book* for an explanation of what plagiarism is and how to avoid it.

All instances of plagiarism from either the internet or within the class will be severely dealt with. No two students may have identical or overly similar reports. No two students may have identical or overly similar source code.

Version 1.1 (17th April, 2018)
elen7045-two-way-traffic.tex