Intelligent Traffic Intersection

Dawar Zaman
Electronic Engineering
Hochschule Hamm-Lippstadt
Lippstadt, Germany
dawar.zaman@stud.hshl.de

***Abstract*—This paper discusses a project that can be used as a solution for better and more efficient traffic management at an intersection. The project consists of a system that prioritizes vehicles on the basis of their collision paths and the vehicles next in line to cross the intersection. The project is made using FreeRTOS libraries.**

***Index Terms*—Mutex, Traffic, Intersection, FreeRTOS, path collision**

I. Introduction

The conventional 4 way intersection traffic guidance system used today consists of traffic lights with timed lane flow and stoppage. This kind of system is used worldwide and although it is a very safe solution it is still very in efficient and outdated due to the fact that sometimes vehicles have to stay waiting in their lanes to cross the intersection even if there are no other vehicles crossing the intersection. Nowadays we have modern cars that are autonomous and can navigate on their own to some extent. In order to develop a system that can work by utilizing techniques that would prevent collisions in an intersection rather than timed lane opening and closing we can construct a more efficient system.

II. Project Description

In this section an overview of the project will be given and also how the project differs from currently used systems. The project consists of a server or an intelligent device placed at any point on a 4 way traffic intersection that connects to other vehicles that are entering the intersection and then depending upon the path the individual vehicle has to take in order to get to a certain destination the system guides the vehicle to its specific destination without causing a collision and without causing the other vehicles that might not be a part of the same path to stop. The difference between this system and a conventional traffic light system is that in a conventional traffic light system, the entire system runs on a timed basis and in such a case the entire lane of vehicles coming from the same source are kept static until the time for the other lanes is still going. whereas, in our system the lanes are not kept static and vehicles are kept in motion unless there are intersecting paths. The system is designed for autonomous cars that can automatically relay the source and destination information to the system and receive instructions on which path to take and so on.

III. Project Requirements

In this section, information from the project description will be extracted and exact requirements that will be needed to use as a guidance criteria in order to work on the project will be listed. The requirements are as follows:

- The vehicles should not collide with each other as the system will be useless otherwise.
- The vehicles should not stop they can slow down otherwise it would be same as a conventional traffic light system.
- The vehicles cannot take a U-Turn since they are autonomous and connect to the internet for navigation therefore the possibility of wrong navigation to an intersection has not been considered.

Following these guidelines we can come up with a scenario as shown in figure 1.

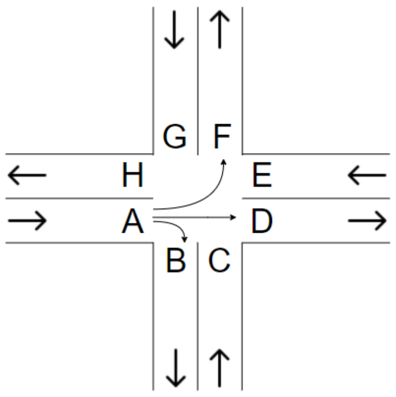

Fig. 1. Possible paths of a vehicle coming from a single source in our realized scenario.

Figure 1 shows the possible routes a vehicle can take from A as its source and B, D and F as its destinations these paths are AB, AD and AF respectively.

IV. Approach

In this section the problem solving approach used for the development of Intelligent Traffic Intersection will be discussed. The approach for developing the project starts by calculating the total number of possible paths in our intersection shown in figure 1. As there are 3 possible paths a vehicle can take from any single source and there are 4 sources in our intersection. Therefore, there exist a total number of 12 paths in our system.

TABLE I
TABLE FOR PATHS AND COLLISION PATHS

Path		Collision Paths							
AB	-	AD	AF	EB	GB				
AD	-	AB	AF	CF	CH	EF	EH	GB	GD
AF	-	AD	AB	CF	CH	EF	EH	GB	GD
CD	-	CF	CH	AD	GD				
CF	-	CD	CH	AD	AF	EB	EH	EF	GD
CH	-	CD	CF	AD	AF	EB	EH	GB	GH
EF	-	EH	EB	CF	AF				
EH	-	EF	EB	AF	CF	CH	GH	GB	GD
EB	-	EF	EH	AB	AD	CH	CF	GD	GB
GH	-	GB	GD	EH	CH				
GB	-	GH	GD	AB	AD	AF	CH	EB	EH
GD	-	GB	GH	AF	AD	CF	CD	EB	EH

These paths form the basis of our system and provide a starting point for our approach.

The next step for developing our approach is to find out the total possible collisions points with respect to each path. The collision points are paths that intersect any path that is being used by a vehicle at that time. Theses collision paths and their respective paths can be seen in the table I.

The idea behind the collision paths is that the paths collide under the following conditions:

1) Paths that share the same source can cause collision.
2) Paths that share the same destination can cause collision.
3) Paths that intersect the ongoing path at any point between the source and destination can cause collision.

V. EXPLAINATION

The code developed for the FreeRTOS [1] Simulation works on the basis of Mutex locks. In the start the number of vehicles that are supposed to enter a lane are defined. Then the Mutex acquire and release conditions are set on the basis of the collision paths and the time taken for a vehicle to go from source (defined as lane in the code) to destination afterwards. These conditions are labeled as xTakeLock for mutex acquire and xGiveLock for mutex release respectively. The four possible intersection sources are defined as processes such as vAtoBDF represents cars from source A going to BDF. From these processes 4 tasks are created representing each individual source and the task stack size is set to 1024. The time for task completion is set to 1000ms meaning that it takes a delay of 1000ms for vehicle to move from source to destination. After a vehicle has passed the vehicle that is next in queue is given the priority to pass. Between one vehicle passing and the other vehicle leaving the source a minor delay of 10ms is set. The cars that are waiting for their turn are also displayed in the CLI. This Mutex acquire and release is the main functionality of the code and the rest of the code consists of defining collision paths and tasks and queues and is repetitive in most cases.

VI. CONCLUSION

The system developed for traffic management at a 4 way traffic intersection in this project is far better than conventional intersection management because rather than keeping the vehicles in a lane waiting. The system keeps the intersection occupied at all times and keeps switching between the lanes. The system takes into account the collision paths and different possibilities of collisions and allows only the vehicles that are not colliding to pass at any given instant. As soon as a vehicle has passed the next vehicle waiting in line is given the priority to pass and this increases the fairness of the system even more.

REFERENCES

[1] www.freertos.org