Self-Regulating Traffic Management System

AUTONOMOUS INTERSECTION MANAGEMENT

Application of the Product Developed and its Significance

Why Traffic Management System?

- Reduce everyday congestion, markedly, by smoothening traffic flow and prioritizing traffic in response to demand in real time.

- Reduce pollution throughout the city: stop-start driving is inefficient and polluting.
- Enable a much more effective response to traffic incidents so as to prevent accidents and gridlocks.
- Enable Inbound Flow Control.

Traffic simulation models are useful for both transportation planning and to transportation design and operations. Lane type, signal timing and other traffic related questions are investigated to improve local system effectiveness and efficiency. Simulation in transportation is important because it can study models that are way too complicated for analytical or numerical treatment, can be used for experimental studies, can study detailed relations that might be lost in analytical or numerical treatment and can produce attractive visual demonstrations of present and future scenarios.

Why Simulation?

Normal analytical techniques make use of extensive mathematical models which require assumptions and restrictions to be placed on the model. This can result in an avoidable inaccuracy in the output data. Simulations avoid placing restrictions on the system and also take random processes into account; in fact, in some cases simulation is the only practical modeling technique applicable. Analysts can study the relationships between components in detail and can simulate the projected consequences of multiple design options before having to implement the outcome in the real-world. It is possible to easily compare alternative designs so as to select the optimal system. The actual process of developing the simulation can itself provide valuable insights into the inner workings of the network which can in turn be used at a later stage.

Viability Study and Risk Assessment

Object Detection and Tracking

The difficulty level of object detection and tracking highly depends on how one defines the object to be detected and tracked. If only a few visual features (e.g. colour) are used as representation of an object, it is not so difficult to identify the all pixels with same colour as the object. However, there is always a possibility of existence of another object or

background with the same colour information. Moreover, the change of illumination in the scene does not guarantee that the colour will be same for the same object in all the frames. This leads to inaccurate segmentation based on only visual features (e.g. colour). This type of variability changes is quite obvious as video objects generally are moving objects. The images of an object may change drastically as it moves from one frame to another through the field of view of a camera.

Illumination Changes

It is desirable that background model adapts to gradual changes of the appearance of the environment. Sudden illumination changes can also occur in the scene. This may also happen in outdoor scenes (fast transition from cloudy to bright sunlight). Illumination strongly affects the appearance of background, and cause false positive detections.

Dynamic Background

Some parts of the scenery may contain movement (movements of clouds, swaying of tree branches), but should be regarded as background, according to their relevance. Such movement can be periodical or irregular. Handling such background dynamics is a challenging task.

Camouflage

Intentionally or not, some objects may poorly differ from the appearance of background, making correct classification difficult. This is especially important in surveillance applications. Camouflage is particularly a problem for temporal differencing methods.

Shadows cast by foreground objects often complicate further processing steps subsequent to background subtraction. Overlapping shadows of foreground regions for example hinder their separation and classification. Researchers have proposed different methods for detection of shadows.

Video Quality Problem

Video may be captured by unstable (e.g. vibrating) cameras. The jitter magnitude varies from one video to another. Video signal is generally superimposed with noise. Background

subtraction approaches for video surveillance have to cope with such degraded signals affected by different types of noise, such as sensor noise or compression artefacts.

Device management

The number of devices will be extremely huge. And they are going to communicate with each other and servers over large geographical areas. As all these devices may not be all connected with each other, several data linking issues must be managed efficiently.

Data Handling

We will gather abundant relative data from different sources such as sensors, contextual data from mobile device information, and social network feeds and so on. Building relationship between those data will be extensive.

Flexibility of the system

Sensors and devices are always evolving with new capabilities and improved functions. This demands to create new cases and optimize the system using those devices. So it's difficult to build a product which can evolve with minimal effort in certain framework to catch up the pace of techniques.

Finding Shortest Route

In real life implementations, there usually are more edge cases that push the implementation to differ from the original versions by a little bit. These open up a newer path leading to different endings for better or worse.

Gantt chart with Proper Timeline Breaking up Stages of Development

	Delivery Phase	Testing Phase 2	Increasing the city Grid & Modification of City Grid Simulator UI	Testing Phase 1	Implementing Path Finding Algorithm In The City grid Simulator	Providing the Data from DIP & ML footage into the Simulator	Building the City Grid Simulator	Video / Image Processing along with implementing Machine Learning on Live videos	Preparator
Start Date	19-Nov	13-Nov	24-Oct	16-Oct	26-Sep	30-Sep	22-Aug	22-Aug	13-Au
Duration (days)	4	6	20	8	20	16	35	39	9

Task(s)			
Start Date	**End Date**	**Description**	**Duration (days)**
13-Aug	*21-Aug*	Preparatory Phase	*9*
22-Aug	*29-Sep*	Video / Image Processing along with implementing Machine Learning on Live videos	*39*
22-Aug	*25-Sep*	Building the City Grid Simulator	*35*
30-Sep	*15-Oct*	Providing the Data from DIP & ML footage into the Simulator	*16*
26-Sep	*15-Oct*	Implementing Path Finding Algorithm In The City grid Simulator	*20*
16-Oct	*23-Oct*	Testing Phase 1	*8*
24-Oct	*12-Nov*	Increasing the city Grid & Modification of City Grid Simulator UI	*20*
13-Nov	*18-Nov*	Testing Phase 2	*6*
19-Nov	*22-Nov*	Delivery Phase	*4*

Underlying Software / Hardware Required

Requirements:

GUI/Simulator Development

- Python and JAVA (including Frameworks)

- JS Libraries like Three.js
- Other Developing Software

Image/Video Processing

- OpenCV using Python or c++
- Microsoft Visual Studio or QT Designer

Machine Learning

- TensorFlow
- Sckit-Learn

Data Analysis

- Octave

Break-up Table of Individual Responsibilities

Start Date	End Date	Description	
13-Aug	*21-Aug*	Preparatory Phase	*All*
22-Aug	*29-Sep*	Video / Image Processing with Machine Learning on Live videos	
22-Aug	*25-Sep*	Building the City Grid Simulator to Model Real-Time Traffic	
30-Sep	*15-Oct*	Providing the Data from DIP & ML footage into the Simulator	
26-Sep	*15-Oct*	Implementing Path Finding Algorithm in The City grid Simulator	
16-Oct	*23-Oct*	Testing Phase 1	*All*
24-Oct	*12-Nov*	Increasing the city Grid & Modification of City Grid Simulator UI	*All*

13-Nov	*18-Nov*	Testing Phase 2	*All*
19-Nov	*22-Nov*	Final Documentation + Delivery Phase	*All*