International Journal of Scientific Research in Computer Science, Engineering and Information Technology
© 2019 IJSRCSEIT | Volume 5 | Issue 3 | ISSN : 2456-3307
DOI : https://doi.org/10.32628/CSEIT11952326

Dynamic Toll Charge using openCV and Spark

Prof. Mounica B[1], Merlin Mathew[2], Aishwarya Balakrishnan[3], Bikky Kumar Goit[4]
[1,2,3]Department of information science and engineering, New horizon college of engineering, Bangalore, Karnataka, India
[4]Information Science and Engineering, New Horizon College of Engineering, Kathmandu, Nepal

ABSTRACT

There are many implementations of intelligent transportation system which is mandatorily required to curb the rising traffic in metropolitan cities. One such implementation is dynamic toll generation which reduces the time taken to pay toll at the toll gates compared to the relatively old method of manual toll collection, although it is still being implemented. One crucial factor to curb traffic and reduce pollution in the cities would be to charge toll according to the seat occupancy in the four wheeler i.e. commuters who use the vehicle appropriately will be charged less and others who use the vehicle luxuriously will be charged more. The system thus implemented produces the toll based on the seat occupancy and the data stored in the databases when the RFID is flashed is used for further analysis.

Keywords : Radio frequency identification, Cassandra-NoSQL database, E-payment wallet.

I. INTRODUCTION

The country's Silicon city, home to over 2.2 million IT workers, is now popularly known for its traffic. This crawling traffic has caused businesses to slow down and showcase their worst behavior during monsoon. The star pattern road structure in the city that causes extreme congestion is one of the main reasons for this traffic. Another crucial factor is the inefficient use of four wheelers where a single occupant, i.e. the driver, is present. On the already narrow roads, this takes up more space per person. Is there a way to facilitate traffic via tolls that are already working?

Static traffic management was one solution that was examined to monitor ongoing traffic with traffic police deployed at each busy junction, although the traffic signs guided the commuters. As the city's population began to swell, this became arduous and inefficient. Traffic lights have been installed as one solution to this problem to help manage traffic. This system is still prevalent and automated, but the problem arose when the length of light was fixed regardless of traffic making traffic reduction ineffective. Smart transport systems were introduced to overcome this. As a measure to curb traffic, many intelligent transport systems have been implemented. Smart transport systems have been increasingly introduced and implemented in the US over the past few years. The introduction and implementation of these ITS in the United States has been witnessed in recent years. Google Maps, for example, uses real-time traffic information from cell phone signals to give consumers predictions of travel time and suggested routes. Dashboards and smartphone applications like Waze now have the ability to provide estimates of the current state of traffic. Posted signs on highways also offer estimated travel times through the works of local transport departments. This paper focuses on the

implementation of dynamic toll charge based on seat occupancy in four wheelers.

II. IMPLEMENTATION

A. FACE DETECTION

This module has been implemented in order to capture the number of people seated in the car. It uses the concept of image processing to capture the faces with the help of third party module (openCV-an open source machine learning software library for image processing, face capture, face recognition etc.) Based on the no of faces detected accurately, the toll will be dynamically generated.

B. RFID SCANNING

RFID scanning includes two blocks namely RFID reader and RFID tag. The RFID tag also known as a transponder is basically a chip that stores some unique information which is used for identification purposes. The RFID tag also contains an antenna that connects the tag to the reader. The RFID reader emits a frequency of certain range and the tags within the designated range will respond and the results will be stored in a database. In this project, the RFID tag attached to the car windshield will be detected and details of vehicle will be stored in database. A system like Paytm FASTag can be used, to which the payment wallet is linked. So, it is convenient for the dynamic toll generated to be deducted from the e-wallet. The RFID tag can hold the authorization data such as vehicle registration number, type and the driver details. This data read can be stored after processing the toll in Cassandra database.

C. CASSANDRA DATABASE

The NoSQL database is integrated with python for storing and fast retrieval of data based on queries for later analysis. The RFID information along with the head count of passengers and time stamp is stored in the Cassandra table, which can be used for further analysis. Structuring Cassandra database is a little tricky and seems redundant creating column-keys/tables as per queries, but this is what makes the retrieval faster. When data is stored with time stamp in Cassandra, it will be useful to track whether the driver is taking a one-way trip or round-about. This can be used to consider one other parameter to determine the toll. For, instance if the vehicle passes by toll gate and selects the route as say “XXXX” and if the vehicle type is a luxury car then the dynamic toll based on the occupancy is generated and stored in the database with timestamp. If, the same vehicle passes by the same toll gate and selects the same route say, “XXXX”, then through the data stored in database we can infer that it’s a two-way trip and reduce the toll by a certain amount as in existing system.

D. SPARK

Flash is an ongoing Hadoop successor and it hopes to have taken a great deal of exercises from how the Hadoop API was planned. Spark works with a master/slave design, where there is a lightweight "Master" administrator, which goes about as an interface to the cluster; it monitors the condition of every cluster and wrangles clusters when queries are submitted. Spark contrasts from Hadoop in a few different ways: it underpins both batch and stream handling, various programming dialects out of the crate (Scala, Java, and Python), in memory calculations, an intelligent shell, and a fundamentally simpler to utilize API.

With in-memory calculations, we can advise Spark to reserve information into RAM as it is hauled out of information stores (HDFS, SQL, Cassandra, and so forth.). From that point onwards, any calculations performed on that information are done in memory with no costly queries; this makes analysis quicker than Hadoop.

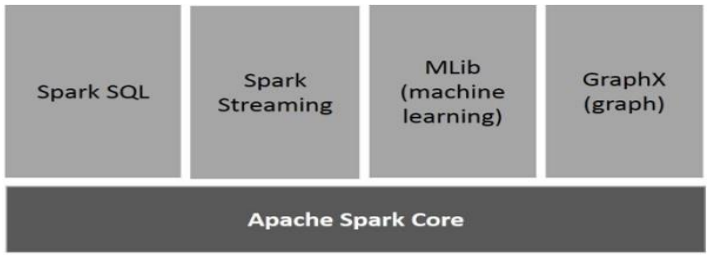
The intuitive shell is utilized to interface with a Spark group and do intelligent calculations on your information. This makes testing, jabbing around, and general hacking a lot simpler. Having a Spark Cluster additionally enables an engineer or information researcher to rapidly try out code and confirm that it works without experiencing a moderate arrangement process.

Sparkle's API configuration is the thing that genuinely separates it from Hadoop. The emphasis is on making interactions with circulated data, staggeringly rich with a meager standard and setup code possible. This plan is established upon the Resilient Distributed Dataset (alluded to as an RDD) which abstracts out the information (that is spread crosswise over many machines) into a solitary enumerable information structure.

The RDD genuinely stands out by the way we can use it in code. An extremely functional language platform is utilized for changing and following up on information. To work with an RDD, you just chain together a group of transformations, (for example, channels or maps or sorts) and afterward utilize an activity to accomplish something with that information. Transformations and actions are the two types of functions.

Transformations change the condition of the information somehow or another; they may channel the information, or twofold the estimation of a whole number, or sort the information, or gathering the information by a key. Understand that changes are lethargically assessed: this implies no work is done until an Action work is called which requests Spark to produce an outcome. Lazy assessment likewise implies that Spark will do the absolute minimum measure of work to create the ideal outcome.

Actions are functions which really extract values from the RDD; this could be requesting the main element, or the initial ten elements, or a set of elements in the RDD, or iterating through every component of the RDD. Numerous things can be practiced in Spark basically by connecting a set of transformations together and then iterating with a foreach. As referenced over, the Action is the thing that makes Spark go out and perform a calculation.



The image above depicts components of Spark.
The Spark SQL (data source) in our project would be Cassandra database and the spark streaming component process the large amount of data streaming every minute. The analytics tool can be used to perform analysis, like the traffic flow through the gate, the effect of the toll generated based on the occupancy on the usage of luxury vehicles and so on.

III. ARCHITECTURE

A. USE CASE DIAGRAM

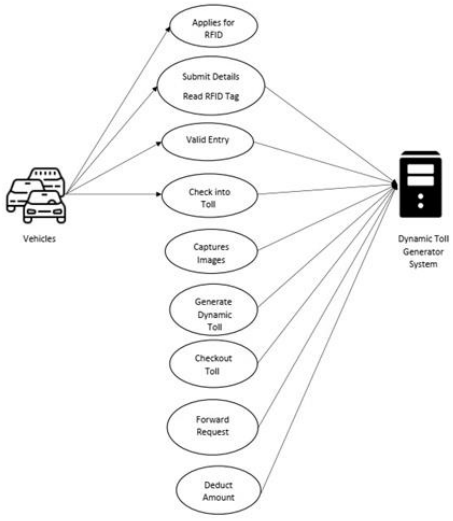


Fig 1 : Use Case Diagram

The use case diagram above depicts the interaction between the driver (vehicle with RFID tag) and the dynamic toll generator system.

B. DATA FLOW DIAGRAM

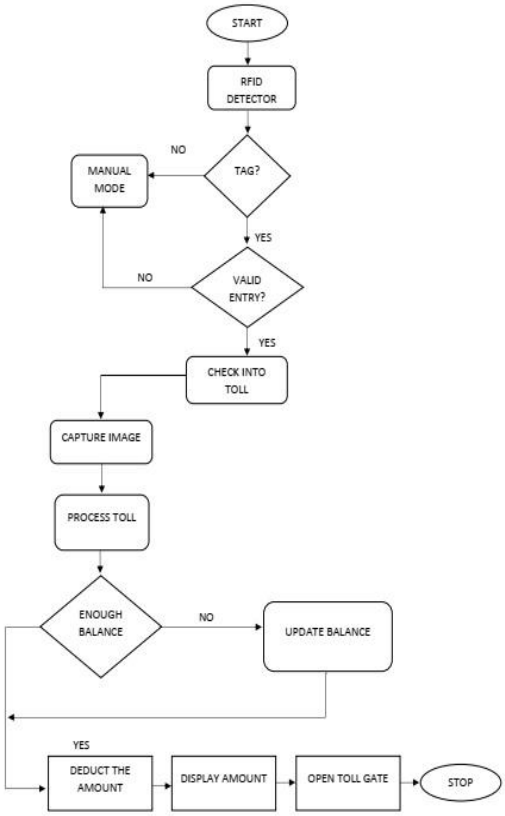


Fig 2 : Data Flow Diagram

The flow of data in the system is depicted in the self-explanatory image above.

IV. CONCLUSION

This project implements the dynamic toll charge based on seat occupancy using RFID technology and face detection techniques. The collected data is efficiently stored in the NoSQL database Cassandra. This data can be analyzed using analysis tools to check if the system implemented has helped curb traffic over a period.

V. REFERENCES

[1]. G. Guerreiro, P. Figueiras, R. Silva, R. Costa e R. Jardim-Gonçalves, "An architecture for Big Data processing on Intelligent Transportation Systems: An application scenario on highway traffic flows," IEEE Intelligent Systems 2016, Sofia, Bulgaria, 2016.

[2]. J. Caserta and E. Cordo, "Big ETL: The Next 'Big' Thing," 9 February 2015. Online]. Available: http://data-informed.com/big-etl-next-bigthing/.

[3]. European Comission, "OPTIMUM," 2015. Online]. Available: http://www.optimumproject.eu/.

[4]. R. Lindsey e E. Verhoef, "Traffic Congestion and Congestion Pricing"Tinbergen Institute, Amsterdam, 2000.

[5]. Y. Lou, Y. Yin e J. Laval, "Optimal dynamic pricing strategies for high occupancy/toll lanes," Transportation Research Part C, vol.19, pp. 64-74, 2011.

[6]. The Apache Software Foundation, "Apache Kafka - A high-throughput distributed messaging system.," 2014. Online]. Available: http://kafka.apache.org/.

[7]. IBM. (2014). Building a smarter transportation management network. Retrieved August 20, 2015, from http://public.dhe.ibm.com/common/ssi/ecm/lb/en/lbw03019usen/ LBW03019USEN.PDF

[8]. 495 — 95 Express Lanes - Pricing. (n.d.). Retrieved August 20, 2015, from https://www.expresslanes.com/pricing

[9]. Yin, Y., and Lou, Y. (2009)," Dynamic Tolling Strategies for Managed Lanes"Journal of Transportation Engineering.

[10]. Zhang, G., Ma, X., and Wang, Y. (2014)" Self-adaptive tolling strategy for enhanced high-occupancy toll lane operations" IEEE Transactions on Intelligent Transportation Systems, 15(1), 306?317.

[11]. USDOT VOT Guidance 2014. (n.d.). Retrieved August 20, 2015, from https://www.transportation.gov/sites/dot.gov/files/docs/USDOT VOT Guidance 2014.pdf

[12]. Thao Phan, Anuradha M. Annaswamy, Diana Yanakiev, and Eric Tseng(2016)," A Model based Dynamic Toll Pricing Strategy for Controlling Highway Traffic",from https://www.researchgate.net/publication/282663

Cite this article as :

Merlin Mathew(merlinthemathew@gmail.com), Aishwarya Balakrishnan (aishu9298@gmail.com), Bikky Kumar Goit (navabryt@gmail.com), Mounica.B (premkumarmounica@gmail.com), "Dynamic Toll Charge using openCV and Spark", International Journal of Scientific Research in Computer Science, Engineering and Information Technology (IJSRCSEIT), ISSN : 2456-3307, Volume 5 Issue 3, pp. 33-37, May-June 2019. Available at doi : https://doi.org/10.32628/CSEIT11952326
Journal URL : http://ijsrcseit.com/CSEIT11952326