Prototype of Cloud Based Smart Parking System Using Sofia2 Middleware

Yohn Jairo Parra, PhD Student[1], Roman Zubatiuk, PhD student[2], Richard Alo, Faculty mentor
Ebrahim Alareqi, PhD student[3], Shentang Wu, PhD student[4], Fang Chen, PhD student[5]
[1]*Jackson State University, USA, j00758975@students.jsums.edu*
[2] *Jackson State University, USA, roman.i.zubatiuk@jsums.edu*

Abstract– The purpose of this project is to create a prototype of a smart parking system by tracking and managing the occupancy of the parking lots at Jackson State University (JSU). The parking lots currently operate without any computerized system. The university management & police department have concerns about inefficiencies of optimal usage of parking space (Congestion/non-permitted parking). Currently, the management monitors the garage occupancy by having employees walk around to inspect the occupancy of individual spots or check decals. This Project aims to solve this problem by providing a system that manages and provides insights from the data collected from the sensors (Sensors and Cameras) and apply data analytics using Sofia2 platform from Indra.

***Keywords--** Sofia2; smart cities; smart parking system; cloud processing; big data analytics*

I. INTRODUCTION

Systems are becoming prevalent and businesses are producing data more than ever. Growing digitalization brings opportunities to organizations to collect, store, and exploit big data to expand business. Companies start buying big data hardware and software infrastructure and collect amounts of data, however, such solutions should be coupled with an integrated approach to data and technology. Creating a middleware system that allows the interoperability of multiple systems and devices require a variety of tools for implementing components of a big data architecture; some of these including classic relational database management system (RDBMS) and NewSQL technologies [1].

Parking experience had been difficult by drivers worldwide on daily basis, allowing significant social and environmental problems like pollution and traffic congestion. Researches have estimated that every day 30% of vehicles on the road in downtown areas of major cities are cruising for a parking spot, and it takes an average of 7.8 minutes to find one [2].

The real solution is to implement a smart parking system using platforms like Sofia [1] to help drivers find a vacant spot using the knowledge from the sensors in each parking space and allow the drivers to have signs that direct them to the available locations. Semantic technologies offer a potential solution to this kind of problems, an ontology supply a uniform conceptual schema that describes the problem domain of the underlying data independently of how and where the data is stored, and declarative mappings specify how the ontology is related to the data [3].

The proposed system allows drivers to search and find a vacant place by information coming from Sofia platform on devices like phones and present a reservation based smart parking system (RSPS) based on the parking availability to the drivers as part of Sofia application. It also provides reservation service as a part of user-targeted-service.

II. BACKGROUND

SOFIA2 is a middleware that allows the interoperability of multiple systems and devices, offering a semantic platform to make real world information available to smart applications (Internet of Things). It is multi-language and multi-protocol, enabling the interconnection of heterogeneous devices. It provides publishing and subscription mechanisms, facilitating the orchestration of sensors and actuators to monitor and act on the environment. Cross-platform and multi-device through its SDK, APIs and extension mechanisms that allow integration with any device.

There are some basic concepts of SOFIA2: A. Smart Space: a collaborative, virtual environment where devices and applications interoperate with each other to provide a complex functionality. B. KP (knowledge processor): represents each client that produces and/or consumes information in a smart space. C. SIB (semantic information broker): the smart space's core, it acts as the integration element for the exchanged semantic information, also as information storage. D. SSAP (smart space protocol): the standard messaging protocol between the KP's and the SIB [4]. See Figure 1

Fig. 1 Process of the virtual environment of Sofia platform

III. METHODOLOGY

The proposed architecture of campus-wide vehicle monitoring and parking informational system is based on commodity hardware, open-source software libraries and cloud-based processing of the information flow. Our purpose is to implement a Smart Parking System for JSU campus, which will collect, process and provide real-time information about availability of parking spots, and predict parking lot occupancy for a certain day. We are also collecting historical data, which will help optimize on-campus traffic and parking, in particular reservation of parking spots. we will use first license plate recognition module based on the contour recognition, (Contour Based Recognition [5]).

Contour is important for pattern recognition. The preprocessing part contains several steps. Transferred RGB image into gray scale, this step is just take the average number of RGB layers of the image. The second step is to transfer the gray scale to a threshold step, in the threshold scene, the edge of the shape of the image is emphasized in this step. In the third step we filter out the contours from the last step. A list of potential contours which could be our license plate is generated and the raw image will be cut based on these contours as a list of potential plate images from the picture.

After the list of sub-images is collected, the preprocess module will be called again till we have all contours on these sub-images. These contours will be filtered and sent into our KNN model to recognize as numbers on the plates. Finally, the sub-image with best outcome from the model will be selected and a rectangle will be added on. The prediction will show right below the image.

Jackson State University campus has several main entrances, and several parking lots. In our prototype, the Smart Parking System data collection is organized as checkpoints on each of the campus and parking lot entrances equipped with two cameras, collecting information of entering and leaving cars. We use consumer-type surveillance cameras connected to Raspberry Pi mini computers for video processing with OpenCV Python library.

We will call this integration as the sensor. The sensor counts entering and leaving cars. It also performs tag recognition; this information is used to track vehicle movement within the campus. The algorithm for the sensor is based on loop over video frames, perform large object (vehicle) detection, process frame images to detect tag numbers, and upload detection event information to the cloud using Sofia2 API.

Tag number recognition is performed with OpenaLPR Python library. It is important to note that the optical character recognition is not always accurate. Typical accuracy for tag recognition with OpenaLPR for our setup is only 70-90%. OpenaLPR reports a list of possible tag number for a given image frame, along with estimated recognition accuracy.

When we are tracking vehicle movement within JSU campus, we need to keep the list of five most probable tag numbers and correlate results not only from different video frames of single detection event, but also with data from different checkpoints. For this, we have a model of the campus that is divided into several zones and we correlate information about vehicles entering and leaving the zones. For this, we use Sofia2 KP, which run correlation analysis software after each detection and updates information stored in Sofia2 database. This approach brings effective tag recognition accuracy to our goal of $> 98\%$.

IV. EXPECTED OUTCOMES

Our implementation of Smart Parking System is a cost-effective solution for implementing traffic monitoring system for JSU campus. It uses commodity hardware components, open-source software, and cloud-based data aggregation and processing. The system is applicable to any campus or facility for traffic control and analysis.

The main power of our approach is in the usage of Sofia2 cloud data processing platform. Sofia2 is multi-language and multi-protocol, enabling the interconnection of heterogeneous devices. it provides publishing and subscription mechanism, facilitating the orchestration of sensors and actuators in order to monitor and act on a certain environment.

The power of Sofia2 also comes with representation of collected and processed information. It has Web dashboard with all the current and projected information about traffic and occupancy of parking lots. Information boards located on campus have simple API to access and display this information to commuters. Sofia2 implements a platform for mobile applications to access and display parking information.

Finally, traffic information will be used to optimize location and number of reserved parking spots.

The main component of our project is the following:

- A camara
- Software that convert pics of car tags to a string of characters
- two sensors: Sensor in, Sensor out
- A Raspberrypi (or normal computer) that sends the information from the sensors and the camera to the cloudLab of Sofia2 through an API.
- Sofia2 Platform
- Dashboard

We need first to register a Kp in the platform. Users must register their KP's in the platform. Otherwise, the platform will reject the connection of that KP's. To register a KP, the platform provides an area of KP Management, where a user can create a new KP or manage those Kp's that have been created previously. A KP can make use of one or several ontologies. This is the platform's information that the KP will produce or consume. Once the KP is registered in the platform, the KP can establish connections with the platform.

Once registered in the platform, we need to generate a token to connect with the KP. A user can generate one or more tokens associated to a KP. To generate new tokens, we will select the KP for which we want to generate tokens and the number of tokens we want to generate. the tokens are deactivated when generated.

We can activate them by clicking on the checkbox Activo. The model based on Model KP delegates the KP's configuration to the SIB. To do this, the user can create a new configuration that she can associate with her model KP. We will specify the software management data such as application name, applicant, description.

once we create a software application, we can see the detail with all the data. we will specify the software configuration properties such as: KP, KP instance, connection token, IP, port. We must establish a link between the newly created application configuration and the previously created KP and KP instance. Then we need to develop a KP for a gateway follow the model KP.

From the platform's console, we can create app model by using crearAppModelo. Then we implement the method readDataSensor of the class PrepareToReceived, where we must establish the connectivity with the sensors and convert the information we get from them in a SensorMessage object. in the prepare to receive class we can read from all the sensors or we can create new classes, extending from prepare to receive worker implementation to manage each of the connected sensors. We implemented the method generate SSAP message of the classes new data received, where we must transform the data we have gathered from the sensors into a SSAP message.

Through the implementation of the class data send to SIB, we get the SIB's response message, along with the original message we had sent to the SIB. If we want to subscribe to the SIB, we must implement the class subscription listener. We can subscribe to the SIB using the method subscribe (ontology, query, SSAP query type); and, through the method on Event, we can define the operations to be performed whenever we receive from the SIB a notification associated to that subscription.

Additionally, we can implement the other classes that allow us to control aspects like Start and Stop the KP application server. Then connect to the SIB and loss of the connectivity, pre-processed and post-processed the messages sent to the SIB. The developers may need to register new workers when those errors where sending to the SIB.

In conclusion, our implementation of campus-wide Smart Parking System is a convenient way for commuters to find free parking slot, and will significantly decrease the campus traffic of the vehicles, seeking for the parking lot with free spots.

figure 2. System Structure

V. Future Work

This project can be scaled up by adding more sensors to the existing ecosystem such as proximity sensors that show users where their cars are in a parking lot. The system also can be scaled by collecting more data about cars and not just the license plates, but also cars color and cars make etc.

We want to find some patterns as a sequence of sensor readings whose average (or maximum) value from Sofia2 that exceeds some thresholds and make some diagnostic using descriptive tools to make decisions on abnormal behaviors and malfunctions as well as predictive tools to anticipate such problems.

Acknowledgment

This research was undertaken through the support of NSF MRI Grant 13-38192: Scalable Omnipresent Environment (SCOPE) and NSF CNS Grant 14-56638 Sensor Environment Imaging (SENSEI)

References

[1] Jens Heidrich, Adam Trendowicz, and Christof Ebert, "Exploiting Big Data's Benefits", IEE computer society, software technology magazine, 2017, pp.9-13

[2] Samir A. El Seoud, Hosam Farouk, Islam Taj Eddin, Towards the development of smart parking system using Mobile and Web technologies, Conference paper, researchgate, 2016, p.1-8

[3] Ian Horrocks,Martin Giese, Evgeny Kharlamov,Arild Waaler, "Using Semantic Technology To Tame the Data Variety Challenge", IEE computer society, software technology magazine, 2017, pp. 25-29

[4] Sofia, Indra technologies, Sofia2 platform, http://sofia2.com/home_en.html

[5] Yong Xu, Yuhui Quan, Zhuming Zhang, Hui Ji, Cornelia Fermu ller, Morimichi Nishigaki, Daniel Dementhon, Contour-Based Recognition, URL:http://www.cfar.umd.edu/~daniel/daniel_papersfordownload/contourBasedRecognition.pdf