I.J. Information Technology and Computer Science, **2021, 3, 1-15**
Published Online June 2021 in MECS (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2021.03.01

IoT Bus Navigation System with Optimized Routing using Machine Learning

Samer I. Mohamed
October University for Modern Sciences and Arts (MSA)
E-mail: saibrahim@msa.edu.eg

Muhamed Abdelhadi
October University for Modern Sciences and Arts (MSA)
E-mail: muahammad.abdelhady@msa.edu.eg

Received: 06 October 2020; Accepted: 03 January 2021; Published: 08 June 2021

Abstract: As the population in Egypt is ever expanding, it is reflected in the increase of the number of vehicles on the road. Public transportation is the solution and the number of available buses can cover a significant amount of the population demand. However, the outdated state of the transportation infrastructure, the static nature of the lines and indistinct schedules create a confounding and unappealing user experience which prompts the users to stray to cars for their needs. So, an Intelligent Urban Transportation System (IUTS) is a must. IUTS is a multi-layered system which provides the solution for most of these problems. It operates on different layers starting from a real time vehicle tracking for transparent and efficient management of assets, cash-less ticketing done through RFID cards, vehicle health and diagnostic data for creation of automated maintenance schedules and a friendly interactive driver interface. In this paper an approach based on combining all these technologies is discussed where the hardware component is implemented based on System-on-Chip technology with custom hardware to interface with the vehicle. The data collected from the on-board unit is sent to the cloud, and with the help of machine learning algorithms the dynamic responsiveness of the system is guaranteed. The proposed system outperforms other existing ones through the dynamic and optimized routing feature for the bus navigation to optimize the operating cost but still satisfy the passengers' demand.

Index Terms: RFID, System-on-Chip, Machine learning, Cloud computing.

1. Introduction

The city life and urbanization have been the instigator of the current traffic problem. Egypt's Greater Cairo Metropolitan area alone has almost one-fifth of Egypt's population which is slightly closing on to 20 million inhabitants. This population causes an incredible amount of stress on the existing transportation infrastructure and leads to a net loss of productivity as well as an increase in stress, unnecessary fuel consumption, and harmful emissions which of course contributes more to the ever-growing climate problems [5]. With these conditions in place, the traditional solution would be to invest more in mass-transit, by increasing the amount of available transportation options, like spiking up the number of buses, micro-buses and creating new metro lines which covers a larger surface area. However, as of 2017, Egypt has licensed close to 156,000 buses [1] at an estimated average capacity of 35 seats per bus. This amounts to over 5.4 million seats, certainly enough to alleviate the stress of the 4.28 million licensed private cars and taxis [2] – since most vehicles tend to be low or even single-occupancy.

The real problem lies in the operational inefficiency found inherently within the static bus system. Under this umbrella, lies three root causes. First, the lack of a digitized infrastructure and lack of publicly accessible information regarding the bus transportation system, like the available bus lines, their departure and arrival times, and current location. This lacks transparency between the operators and users, causes chaos, confusion, and an unpleasant user experience [3]. Second, the distribution of resources (i.e. buses) which is often based on outdated feasibility analyses that are unreflective of the real-world supply and demand. Most buses are sitting idle in their respective hubs, waiting for the users to come, instead of seeking actual real-time demand. The third and final point is, the predefined static routes that only contributes to the rigidity of the established system which does not offer coverage flexibility for new territories. The previously described problems instigate the need for an IUTS that features dynamic routing and demand-responsive scheduling. This helps both the operators to have a more profitable business and the users for a more comfortable and seamless user experience [4]. We focus through our proposed system to overcome the drawbacks or weaknesses in previous systems as detailed in the previous sections and design system that facilitates better bus

navigation for both operator and end users. Utilize the latest digital transformation resources and machine learning to effectively match demand (users hits/requests) and supply (bus fleet size) for better cost optimization on real-time.

The paper is organized in 4 sections, first section introduces the main driver behind our research article and main research objectives, second section summarize the previous and current systems with clear view on their strengths and weaknesses, then section 3 deep dive into our proposed system methodology with clear view on the design analysis and structure of implementation, section 4 introduces the conceptual formulation of the research problem via clear algorithms used for design implementation that is covered in more details in section 5 with overview on both development and testing scheme, then last but not lest section 6 reflect and discuss the results achieved compared to other benchmarks and/or existing systems.

2. Background

With the technological advances in the world of wireless communication, a lot of countries have started investing in and developing their own unique IUTS. Many systems use a combination of hardware and software to form a compact solution. In this section, a brief literature review about some of these available systems will be covered to show the strengths and weaknesses of each system to draft the vision and way forward for the proposed system [6]. A lot of the hardware-based systems use the same architecture: a micro-controller, GPS and a communication module GSM. Pham, Micheal and Chi [7] proposed a system which provides vehicle tracking through a u-blox NEO-6Q GPS module and a basic communication module comprised of a u-blox LEON-G 100 GSM modem. The components are connected to an Arduino UNO micro-controller. The data read from the GPS are sent through the GSM via text message to the user's phone. This simple architecture allows for live tracking of the vehicle, yet that is the only functionality used [10].

Petracca, Pagano, et al., developed a system which focuses on designing a vehicular On-Board Unit (OBU) using off-the shelf hardware modules available at the consumer electronics market. The main function of the OBU is to communicate with other OBUs implemented on other vehicles, and with the Roadside Units (RSUs) to send the data to the database [8] using certain components and technologies. The OBU comprises of an Atmel ARM processor, SPK Electronics GS405 GPS module to send the location of the vehicle, MCP2515 CAN transceiver to be able to communicate with the On-Board Diagnostic (OBD) inside the vehicle, wireless module to communicate with the RSUs and the database [9]. Transport for London is one of the implemented systems over past years that has implemented some useful features such as a messenger bot and an SMS service to support the main website in providing the most accurate information about their fleet [5]. These efforts help to make information more transparent and accessible to the passengers has resulted in a better experience.

Thanks to the new technologies and advances of digital transformation, many new business models and services are introduced on the global level like Uber bus and Careem bus [36]. These services offered very good user experience via mobile applications that help users to easily and from anywhere to request the service with clear view on ETA (Estimate Time of Arrival) and available bus capacity. Swvl [37] is an Egyptian app-based startup through which users can book fixed-route bus trips at prices 60-80 percent lower than competing ride-hailing services and without surge or peak pricing. This service tackles the same issues as our proposed project; however, it approaches the problem using out of service tourism buses. It provides similar features in terms of vehicle tracking, ease-of-use, and a mobile-first approach. Although these systems achieve good results and tackle some of the main challenges exist at that point of times, but lack the overall picture to satisfy passengers demand, scalability, and cost optimization. In the proposed system, we focused to augment what previous systems lack to achieve in terms of better user experience for both clients and decision makers, running cost optimizations and improved maintainability.

3. Methodology

This paper's focus is on the system design and implementation for the Intelligent Urban Transportation system IUTS to tackle the mentioned challenges detailed in the previous sections. The proposed design needs to satisfy the following functional requirements and design principles as detailed in the below sections:

3.1. Vehicle Tracking

To provide the user with the most accurate information regarding the current location of the requested bus, a tracking module must be installed in all the vehicles. This module is crucial to the system as it collects real-time data of the bus's exact location.

3.2. Fare Payment

A cashless fare payment module should be used as the main ticketing system. As the public bus system is still reliant on paper tickets which are easy to defraud and are difficult to keep track of. Not only to get a ticket, but also as a user identification and driver authentication module.

3.3. Vehicle Health Monitoring

The module collects real-time vehicle specific data like the speed and engine related status. These data give some insights which help in scheduling maintenance and monitor the health of the vehicle and the driving behavior.

3.4. Server Communication

All the data collected from the implemented modules must be broadcasted to a cloud network where it is processed and analyzed.

The high-level design of the entire system is shown in Fig. 1. The system comprises of three integral parts, the hardware layer which is implemented onboard the buses and on the bus stops through which the data is collected, the Central Management System (CMS) [15] which includes the server, database and the application layer which interacts with both the users and operators in the form of a mobile application and a dashboard. The hardware consists of an OBU, which is installed onboard the buses. This acts as a data collector, collecting the data from the various modules. The temporal and spatial data which allow live tracking, the cashless ticketing module, and finally the health monitoring through the OBD to achieve the function requirements. A processing unit, processing and analyzing the collected data and a communication unit, communicating with the server sending the data packets and receiving the final route [20].

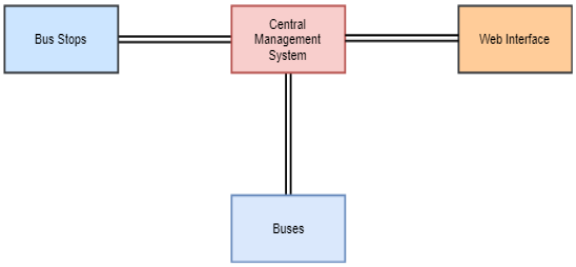


Fig.1. High-Level System Design

For the high-level design, the use-case flow diagram shown in Fig. 2. Shows the different levels of privileges required for each user and the data they would need access to. For example, a typical passenger may choose to book through the application first or they may find themselves already at a bus stop. In both of those scenarios, the passenger wishes to reserve a seat which involves verification of identity, whether that is through the application or through swiping their RFID card at the bus stop. This adds a layer of security as each passenger is uniquely identifiable [18].

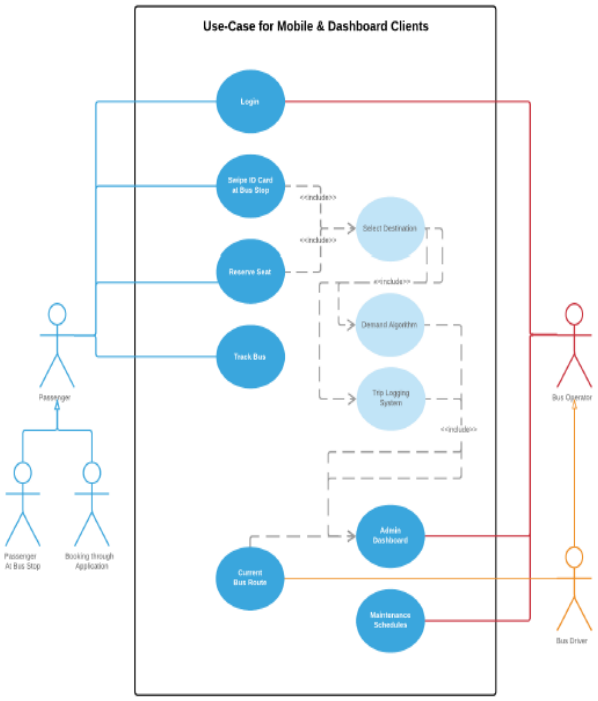


Fig.2. Use-case flow diagram for Passengers & Operators

3.5. Design approach

The detailed design process went through two phases as follows:

A. Design Phase I

The onboard unit (OBU) detailed in Fig. 3, is an integral part of the system as it will generate the necessary data required for the software component of the overall proposed system. Given the lack of accurate, publicly available data about the bus transport system, aspired a need for an accurate GPS module. This module is activated continuously and transmits a series of latitude-longitude coordinates to the CMS using the GSM transceiver. The second function of the onboard sub-system is to monitor the health of the vehicle, i.e. check for fault codes and alert the operator before any major issues. For this function, the output of the vehicle's OBD socket must be read and interpreted. The proposed system is planned to pilot on the a University bus fleet, therefore the OBD socket must be adapted such that it uses the standard Controller Area Network (CAN) protocol as the K-Line protocol [19] is outdated and cannot be adapted to the Universal Asynchronous Receiver-Transmitter (UART) format. From there the Raspberry Pi (RPi) can interface with the OBD using UART and can translate the continuous output of the OBD, using a look-up table, to a set of alerts. The third function of the on-board unit is to enable cashless payment through a Radio-Frequency Identification (RFID) reader module. A typical user interaction with the RFID reader, whereby a passenger boarding the bus must swipe their RFID card [16] to trigger a set of validation and verification procedures which ensure a passenger is using a valid card and has sufficient funds in their virtual wallet to pay for the trip. Finally, the system receives its route from the CMS and is capable of displaying that route the driver with instructions; this is possible through the Raspberry Pi's interface with the LCD screen [17].

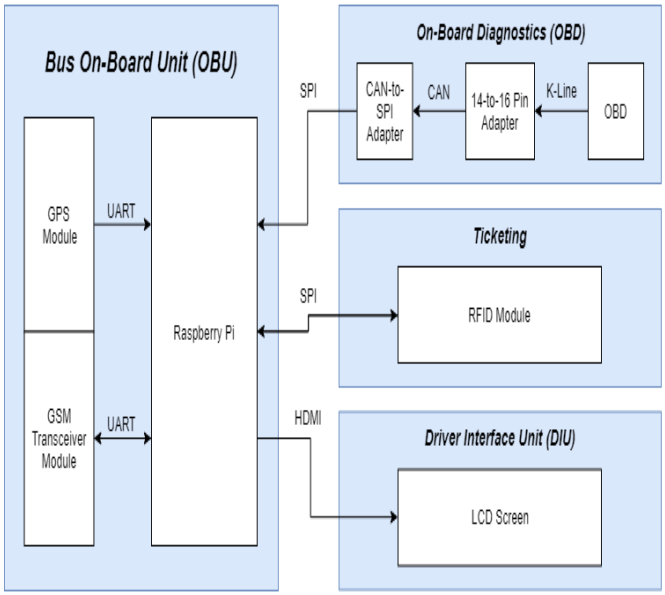


Fig.3. On-Board Unit (OBU) phase I

B. Design Phase II

The second phase of the design managed to reinvigorate the original design which gave the system more robustness and reliability. As detailed in Fig. 4, the GSM module was replaced by a mobile broadband module which gives the central processing unit access to the internet, allowing a smoother connection to the CMS. The K-line to OBD-II standard adapter was a hassle to find, which inspired the migration to a Bluetooth based OBD-II controller. And finally, the LCD screen was replaced by an android tablet which allows the use of turn-by-turn navigation.

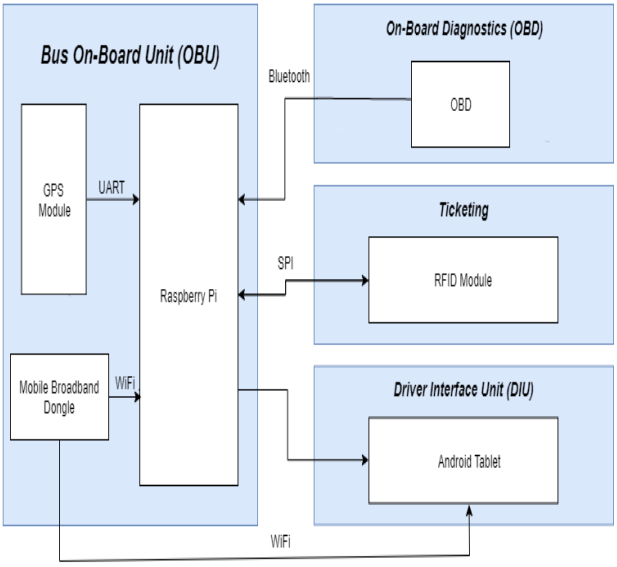


Fig.4. On-Board Unit Phase II

a. Design Phase II circuit schematics

As inscribed in Fig. 5, the phase II circuit schematic shows all the modules and the different protocols and connection to the RPi as detailed in the below sections.

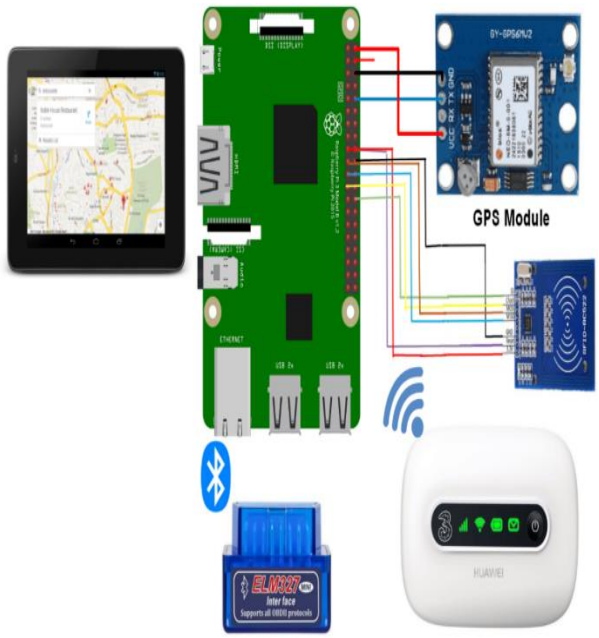


Fig.5. Design Circuit Schematic

3.6. GPS

This module interfaces with the RPi using UART protocol where the connection is shown in Fig. 5. The module requires 3.3V to operate. Therefore, it is connected to pin-1 on the RPi. A GPRS packet holds up to 9 data bits and its structure is shown in Fig. 6. The start bit is normally high when no data is transmitted and is pulled low to begin transmission. The data frame is read at the frequency of the baud rate [16].

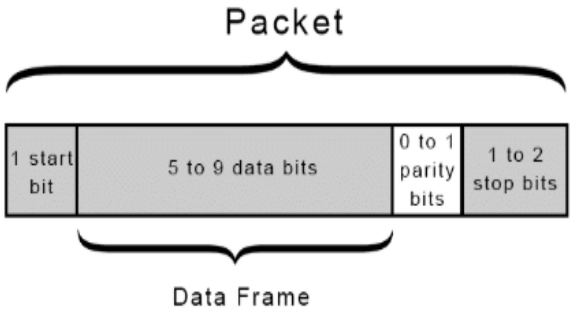


Fig.6. GPRS packet structure

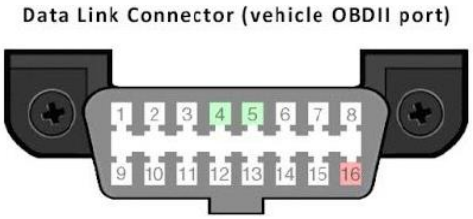
The data frame can extend up to 9 bits (8 if parity is used) and is sent least significant bit first.

3.7. RFID

The RFID module is connected to the pi with the Serial Peripheral interface (SPI) protocol as. The RFID needs 3.3 V to operate which is provided by the RPi's pin 17. The RFID reader can send up to 10 bytes of data signifying the user-ID number corresponding to each RFID tag [14].

3.8. OBD

Most vehicles must be equipped with OBD-II using the standard 16-pin connector shown in the figure below. However, the prototype is set to pilot on the university fleet which uses an obsolete OBD-II that features a 14-pin connector. Therefore, to interface with the OBD, a 14 to 16 pin adapter is needed. Then, the ELM327 OBD-II connected to the adapter and communicates with the RPi over Bluetooth. Connecting to this interface will allow the reading of a list of different parameters provided by the adapter. The parameters we are focusing on are vehicle speed, engine load, engine rpm and engine coolant temperature [25]. The OBD interface outputs fault codes that can automatically be compared with the fault code table provided by the vehicles' vendor, to detect any problems with the vehicle and schedule maintenance. As shown in Fig. 7, the ELM327 adapter and the OBD-II port pinouts are as follows:



1 Make/Model Specific
2 SAE J1850-PWM POS(+) or SAE J1850-VPW POS(+)
3 Make/Model Specific
4 Chassis Ground (all protocols)
5 Signal Ground (all protocols)
6 ISO15765-4 CAN-Bus High
7 ISO9141-2 K-Line or ISO14230-4 KWP2000 K-Line
8 Make/Model Specific
9 Make/Model Specific
10 SAE J1850-PWM NEG(-)
11 Make/Model Specific
12 Make/Model Specific
13 Make/Model Specific
14 ISO15765-4 CAN-Bus Low
15 ISO9141-2 L-Line or ISO14230-4 KWP2000 L-Line
16 +12v (always on) (all protocols)

Fig.7. OBD-II port pinout

A total of 16 bytes are sent by the ELM327 OBD-II module over SPI to the RPi, and they are divided as follows in Fig. 8. SOF delineates the start of frame bit; the 11-bit identifier is used to determine the type of message; RTR (Remote Transmission Request) distinguishes between a data frame and a remote frame; IDE (Identifier Extension) denotes the frame format; DLC (Data Length Code) is a 4-bit long code containing the number of bytes being transmitted; CRC (Cyclic Redundancy Check) is a 16-bit checksum of the preceding data bytes, used for error detection; EOF (End of Frame) marks the end of the OBD-UU frame; finally IFS (Inter-Frame Spacing) which specifies a number of bits to separate consecutive messages.

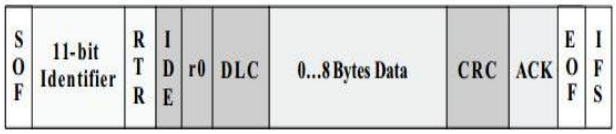


Fig.8. Standard OBD-II frame

The data field which constitutes 8 bytes in the OBD-II frame comprises of the following as shown in Fig. 9:

Fig.9. OBD-II frame payload

- #Bytes: denotes the length of the incoming data bytes
- Mode: the mode field controls whether there is a request or a response. There are 10 modes of response. For example, “41” means show the current data (The mode we are interested in)
- PID: for each mode, there exist a list of parameter IDs (PID). Each PID corresponds to a parameter read from the vehicle. For example, “0D” represents vehicle speed.
- Ah, Bh, Ch, Dh: These are the data bytes in HEX, which must be converted to decimal before being used in the PID formula calculations [30].

3.9. System Flowcharts

In this section, the flowcharts of the interaction of each module is described as shown in figures 10, 11, 12, 13.

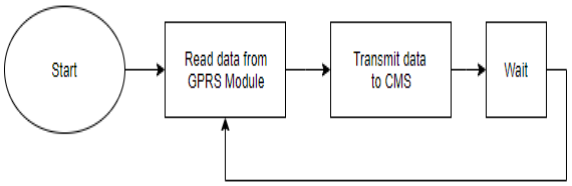


Fig.10. Continuous location transmission flowchart

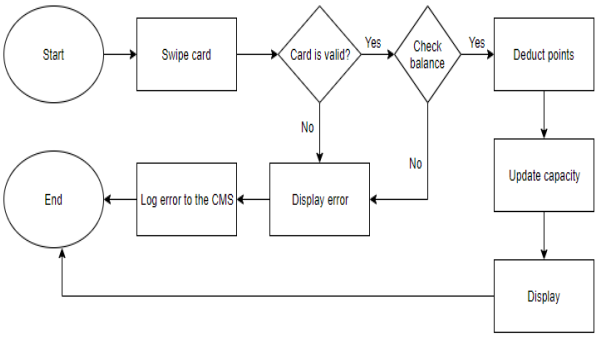


Fig.11. User boarding confirmation & payment

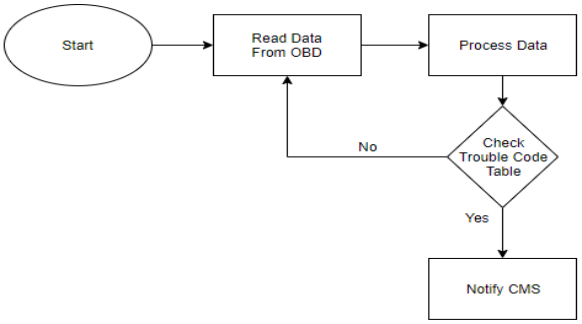


Fig.12. Continuous health monitoring flowchart

4. Problem Formulation

The main edge and uniqueness of the proposed system that outperforms other existing system is the dynamic and optimized routing feature that encompass the aims to optimize the operating cost and/or minimize the overall operating costs through optimizing the bus route distance while still satisfying the demand raised by the passengers on specific route/line.

Particle Swarm Optimization (PSO) has proven to be much faster at converging with a strong resistance to getting stuck in local minima [18]. While PSO is a similarly iterative algorithm, it utilizes intelligence from a swarm of particles to determine the best direction for convergence in a search space. Vehicle Routing Problem, (VRP) whereby a given commodity must be delivered from a depot to a set of locations along an optimal route with the optimum number of vehicles – as constrained by their capacity. There are plenty of variations on the traditional VRP; ones including multiple commodities with different sources and destinations such as the Multi-Compartment VRP (MCVRP), the Multi-Depot VRP with Pickup and Delivery requests (MDPDR) [21], and Bus Transit Route Network Design Problem (BTRNDP) [22]. Once the dynamic routing problem is formulated as a Vehicle Routing Problem (VRP), with the objective function and the set of constraints, PSO can be deployed as a solver to determine the optimum routes and the minimum number of vehicles required.

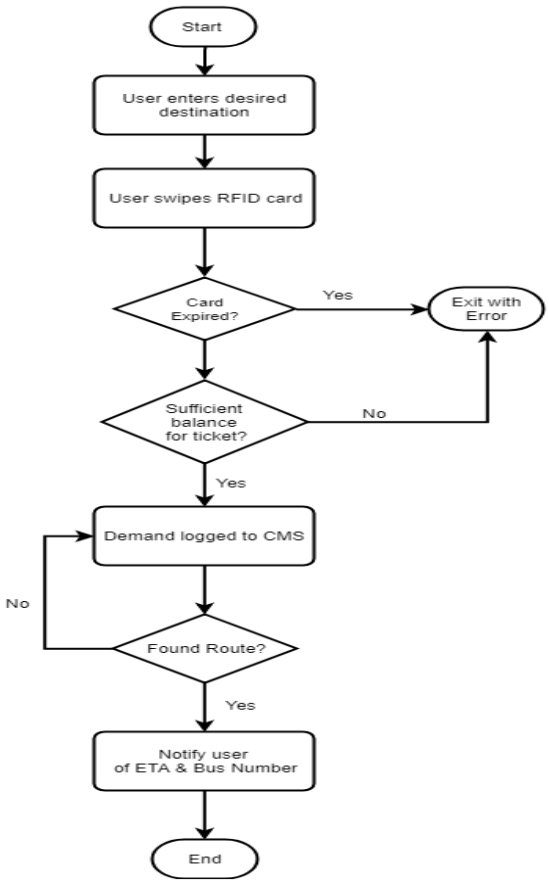


Fig.13. User request at bus stop flowchart

The VRP formulation chosen in our proposed system is the MDPDR VRP referenced in [23] as it allows for multiple pickup and drop-off locations. The objectives considered in this solution are the total distance, the number of vehicles used, and the number of fulfilled requests in the form of Passenger Load Factor (PLF) [28] which encompasses the ratio of cost per passenger km to cost per vehicle km. the list on input parameters for the MDPDR problem [24] shown in below listing of Input Parameters:

P: set of pickup nodes, {*1… n*}.
D: set of delivery nodes, {*n+1... 2n*}.
N : set of all pickup and delivery nodes *N= P* U *D*
H_i : a penalty cost if the request i is not served, *i* C *P*.
K : set of all vehicles, |*K*| = *m*.

C_k : capacity of vehicle k C K.
f_k : fixed cost of vehicle k C K if it is used
g_k : variable cost per distance unit of vehicle k C K
v_k : nodes that represents the start station of vehicle k, k C K.
v'_k : nodes that represents end station of vehicle k, k CK.
V : set of all nodes. $V = N$ U $\{T_1 \ldots T_m\}$ U $\{T'_1 \ldots T'_m\}$
A : set of (i, j) which is an arc from node i to node j, where i, j C V.
d_{ij} , t_{ij} : distance and travel time between node i and node j ,for i and j C N. Travel times satisfy the triangle inequality; $t_{i\,j} \leq t_{il} + t_{lj}$ for all i, j, l C V; and are nonnegative.
s_i : fixed service time when visiting node i.
e_i : variable service time per item units of node i.
$[a_i,b_i]$: time windows when the visit at the particular location must start; a visit to node i can only take place between time a_i and b_i
l_i : the quantity of goods to be loaded onto the vehicle at node i for i C P and
$l_i = -l_{i-n}$ for i C D.

The routes are generated according to the following objectives:

1- Total routing cost is minimized;
2- Number of vehicles used is minimized;
3- Passenger load factor is maximized.

Furthermore, the solution must respect the following constraints:

(1) Each request can only be served once by a bus;
(2) The load of a vehicle must never exceed its capacity;
(3) Each route must start and end at a depot;
(4) The number of buses utilized cannot exceed the number of available buses;
(5) The total duration of each route does not exceed a predefined limit

The mathematical model for the MDPDR formulation as shown in below problem formulation:

$$\text{Minimize} \alpha \sum_{k\in K} g_k \sum_{(i,j)\in A} d_{ij} x_{ijk} + \beta \sum_{k\in K} \sum_{j\in P} f_k x_{\tau_k,j,k} + \gamma \sum_{i\in P} H_i z_i \quad (1)$$

Subject to:

$$\sum_{k\in K} \sum_{j\in N_k} x_{ijk} + z_i = 1 \quad \forall i \in P \quad (2)$$

$$\sum_{j\in V} x_{ijk} - \sum_{j\in V} x_{i,n+i,k} = 0 \quad \forall k \in K, \forall i \in P \quad (3)$$

$$\sum_{j\in P\cup\{\tau'_k\}} x_{\tau_k,j,k} = 1 \quad \forall k \in K \quad (4)$$

$$\sum_{i\in D\cup\{\tau_k\}} x_{i,\tau'_k,k} = 1 \forall k \in K \quad (5)$$

$$\sum_{i\in V} x_{ijk} - \sum_{i\in V} x_{j,i,k} = 0 \quad \forall k \in K, \forall j \in N \quad (6)$$

$$x_{ijk} = 1 \Longrightarrow \quad S_{ik} + s_i + t_{ij} \leq S_{jk} \qquad \forall k \in K, \forall (i,j) \in A \quad (7)$$

$$a_i \leq S_{ik} \leq b_i \qquad \forall k \in K, \forall i \in V \quad (8)$$

$$S_{ik} \leq S_{n+i,k} \qquad \forall k \in K, \forall i \in P \quad (9)$$

$$x_{ijk} = 1 \Longrightarrow \quad L_{ik} + l_i \leq L_{jk} \qquad \forall k \in K, \forall (i,j) \in A \quad (10)$$

$$L_{ik} \leq C_k \qquad \forall k \in K, \forall i \in V \quad (11)$$

$$L_{\tau_k k} = L_{\tau'_k k} = 0 \qquad \forall k \in K \tag{12}$$

$$x_{ijk} \in \{0,1\} \qquad \forall k \in K, \forall (i,j) \in A \tag{13}$$

$$z_i \in \{0,1\} \qquad \forall i \in P \tag{14}$$

$$S_{ik} \geq 0 \qquad \forall k \in K, \forall i \in V \tag{15}$$

$$L_{ik} \geq 0 \qquad \forall k \in K, \forall i \in V \tag{16}$$

Where:

(1) $xijk$ denotes a Boolean variable signifying if a vehicle *k* travels from node *i* to node *j*;

(2) Sik indicates when vehicle *k* starts service at location *i*;

(3) Lik is the cumulative capacity of the vehicle after vehicle *k* has visited node *i*;

(4) zi is a Boolean variable indicating whether request *i* is queued – this allows accountability for unfulfilled requests.

(5) Weights α, β, and γ, found in the objective function, are operator-set variables to allow for the tuning of the relative importance of each cost to meet business objectives.

The objective of the formulation highlighted in Fig. 14 is to minimize the sum of the operating costs – in terms of distance and fleet utilization – as well as the penalty for unfulfilled requests; whilst also ensuring that none of the constraints are violated. Constraint (2) represents that a passenger request can only ever be either in queue or fulfilled by a bus. Constraint (3) signifies that any drop off point corresponding to any pickup point must be visited by the same bus. Constraints (4) & (5) ensure that a vehicle visits every pickup and corresponding drop off point. Constraint (6) ensures that every arc is consecutive, i.e. the next drop off point must originate from the vehicle's current location. Constraints (7) and (8) ensure that time windows are obeyed and that fixed service time per location is considered alongside the variable pickup time per passenger, as well as travel time between stops. Constraint (9) simply ensures that pickups and drop offs are sequential in time. Finally, constraints (10), (11), and (12) ensure that the capacity constraints of each vehicle are respected and that the load variable is accounted for correctly [29].

5. Implementation And Testing Results

The GSM proved to be troublesome in the originally proposed design. The main issue was using the GSM for the communication with the server, as multiple tests proved that we cannot use this feature. Furthermore, the module itself was not responsive to the AT commands regarding the TCP/IP communication when tried with various processors, controllers and even using a TTL to USB converter which is an adapter used to directly convert from the standard TTL-UART serial communication to USB so that it could be connected to the computer directly and after running the commands using a terminal application. To mitigate these problems, a change to the original design is proposed [25].

The second module that was replaced was the LCD screen. As mentioned earlier, that one of the core objectives is the dynamic response to the demand, which means that turn-by-turn navigation is a must. But, google maps navigation is only supported by mobile devices. Alternatives to google maps were tested like "navit" and "mapbox, navit" uses street maps which most users are not familiar with and did not include Arabic street names which would be troublesome for the drivers besides "mapbox" as well, does not support progressive web applications (PWA) [26] which is essential as our application is built based on this technology. To solve the encountered problems during the first phase of implementation, a new system was proposed whereby the GSM module is replaced with a portable Wi-Fi module. This will allow the raspberry pi to access the internet and communicate with the server. Another addition is an android tablet, which can connect to the same network as the RPi. Adding the tablet will provide a sturdier, more familiar interface, as well help with meeting the requirement of turn-by-turn navigation [27].

The raspberry pi runs two concurrent threads, one which facilitates the communication with the main server to send the connected modules' data and receives the specific trip Id and driver name. The other thread is a locally hosted server on the RPi, which takes the trip ID from the first and gets the trip data, then layoff the route to the tablet. The flow is shown in Fig. 14.

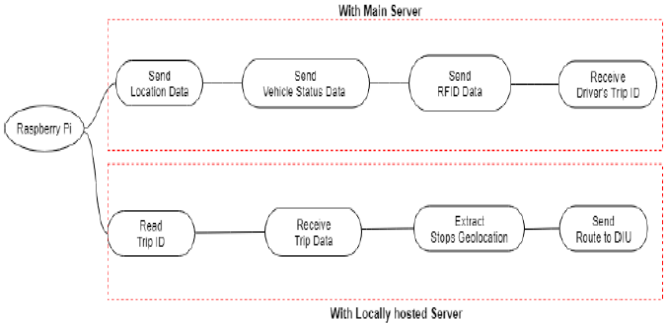


Fig.14. Hardware-to-Software Integration Flow

After all the hardware components have been connected and a unit tests have been carried out, the hardware must then communicate with the web application server for the integration testing as shown in Fig 15. For this, a generalized function is created which is responsible for opening a channel to the server using the requests python library. The function takes different parameters to match the different data received from the variety of hardware modules. The constant parameters in all cases are, the server's URL which is used to open the connection with the server; the "tripId", which identifies the specific trip data in need of updating; finally, the headers, which contains the HTTP request headers required to communicate to the server the exact nature of the data in terms of data encapsulation, encoding, and format [31].

An end-to-end test was carried out, according to the following scenario:

- The predicted demand compared to the captured, is then formulated to trips.
- Once the driver boards the bus, he swipes his RFID card to receive the formulated trip route.
- User open up the mobile application, selects a source and destination.
- The server receives the request and finds the closest bus with the same route.
- The OBU mounted on-board the buses, constantly sends the location updates.
- Once the ride has been confirmed, the user can track the bus through the app and the operator can do so through the dashboard.
- Once boarded, the user then swipes their RFID card for identification and ticket deduction.
- The OBD data is constantly sent to the server.

The next figures demonstrate the terminal's output of the integration test with the server: A snippet of the terminal's output can be viewed in Fig. 15 as it captures the reading from the GPS module and displays the captured input, as well as the returned JSON response from the server denoting that the location has indeed been received on the server's end and has been loaded into the database where it can be displayed to the user.

```
lat in degrees: 30.1042  long in degree:  31.3237

Updating location
{'data': {'UpdateTripLocation': True},
 'extensions': {'tracing': {'duration': 457161,
                            'endTime': '2019-06-12T20:38:40.743Z',
                            'execution': {'resolvers': [{'duration': 316036,
                                                         'fieldName': 'UpdateTripLocation',
                                                         'parentType': 'Mutation',
                                                         'path': ['UpdateTripLocation'],
                                                         'returnType': 'Boolean!',
                                                         'startOffset': 129151}]},
                            'startTime': '2019-06-12T20:38:40.742Z',
                            'version': 1}}}
```

Fig.15. Update Location response

Once a driver card interrupt is detected the driver's credentials are validated, and the trip ID is returned as shown in Fig. 16.

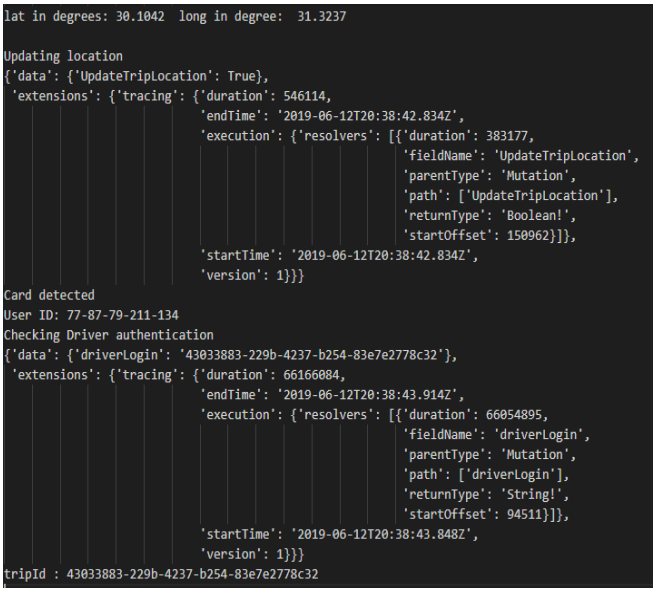


Fig.16. Driver authentication

Fig. 17 shows, when a user boards the vehicle and swipes their card, their info is validated; if valid, their reservation's status changes to "boarded" and their pickup time is updated.

Fig.17. User Boarding

To achieve turn-by-turn navigation functionality, the trip's route must be communicated to the Android tablet. For this problem, our solution began by hosting a simple local micro-server on the raspberry pi built on flask.

The RPi receives the trip ID which belongs to the logged in driver from the first thread, passes it to the local server thread. This trip ID is then passed back to the web server this time in order to retrieve the trip's data. The returned trip data is a JSON object containing all the information regarding this specific trip from the database. After parsing the trip's route data from the web server, ordering the stops, and extracting the latitude and longitude of every stop; the extracted locations are appended into a google maps directions URL [33].

This entire operation is carried out seamlessly in the background while the driver is served a friendly welcome page on the Android tablet. Once the above process is complete, a "Start Trip!" button is made available to the driver. Upon pressing, the interface automatically redirects to the navigation URL show in Fig. 18, which launches google maps, where the route is drawn, and turn-by-turn navigation can commence [32].

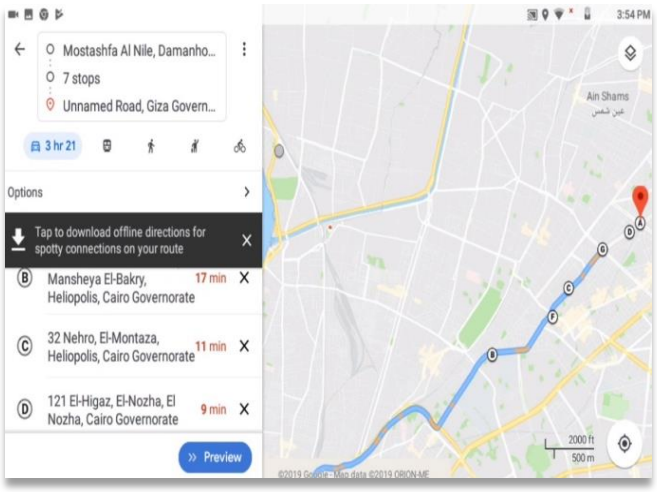


Fig.18. Turn-by-turn navigation

6. Results Analysis And Reflection

As detailed in previous sec 4 around our proposed system problem formation, we adopted MDPDR VRP for this application, were leveraging two well-known LSTM-RNN implementations: Single-variable time series forecasting LSTM, and Tensorflow RNN. Start by visualizing the dataset in LSTM to gain some information about the data trend and seasonality. After loading the data into a data frame, it is normalized using a Min-Max Scaler preprocessing class and split into 80-20% for training-testing respectively. From there, the dataset is split again from its current NumPy array form into a sequential dataset with several previous time steps to be used as input variables to predict the next time period. After running the model for 100 epochs, we achieve at raining score of 22.93 RMSE and a test score of 47.53 RMSE as shown in Figure 19.

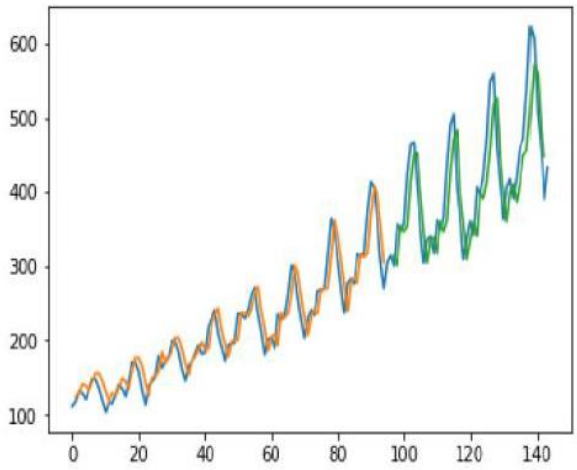


Fig.19. Proposed trained model results

Compared with the systems discussed in the literature review section, our proposed system provides the same live-tracking functionality. But, unlike Pham, Micheal and Chi's system [23], our proposed system is more reliable regarding data handling. While they use a GSM module to send SMS messages to the user's phone, our proposed system constantly sends the tracking data to the server where then it is shown to both business operators through the dashboard and to the user via the application. This provides more transparency and reliability. Even though Petracca, et al [22] system provides similar functionality as our proposed one, still with our system there is no need for extra infrastructure components like the Road Side Units (RSUs), [35] as the on-board data are transmitted directly to the main server without the help of outside modules. The main edge is to use the machine learning and optimization techniques detailed in the problem formulation system to optimize the overall operating cost in terms of dynamic route distance formulated in terms of PLF as result from the MDPDR VRP adopted methodology.

7. Conclusion

The need for demand-responsive mass transit is apparent. The main challenges faced by the traditional bus system include lack of digital infrastructure, outdated distributions, and schedules, as well as outdated or inefficient routes. The proposed solution attempts to tackle these problems by operating in the fuzzy domain, making predictions about passenger demand, and preemptively customizing and optimizing routes and schedules to as many passengers as possible. This is in addition to the light-weight mobile application and the RFID payment system which will enhance the overall user experience. The benefits of such a system would extend across social, economic, and environmental improvements. While there have been similar attempts at solving the various sub-problems present within the traditional bus system, very few have attempted to re-imagine the overall experience and challenge the culturally established norms about the bus system. We still aim to improve the current results of cost optimization by adopting reinforcement learning and assess the implication on PLF calculations and bus route optimization and cost projections.

References

[1] Central Agency for Public Mobilization and Statistics," Egypt In Figures", 2015.
[2] SWVL," Egyptian start-up Swvl raises USD 8 million in series A funding round", 2018.
[3] K. Ltd," KPIT — On-Bus Intelligent Transport System", KPIT Technologies Ltd, 2018. Available at: https://www.kpit.com/what-wedo/products/on-bus-its.
[4] B. Hamilton, History of intelligent transportation systems, U.S. Department of Transportation, 2016.
[5] T. Matters, Facts & figures", Transport for London, 2018. Available at: https://tfl.gov.uk/corporate/about-tfl/what-we-do/londonunderground/
[6] Seoul Transport Operation & Information Service, Topis.seoul.go.kr, 2018. Available at: http://topis.seoul.go.kr/eng/page/transInfo 1 1.jsp.
[7] Angelakis, V., Tragos, E., Pohls, H., Kapovits, A. and Bassi, A. Designing, developing, and facilitating smart cities, 2017.
[8] Bhusiri, N., Qureshi, A.G. and Taniguchi, E, The trade-off between fixed vehicle costs and time-dependent arrival penalties in a routing problem, Transportation Research Part E, 62, 1 -22, 2017
[9] V. Bogdanov, How To Choose The Right Tech Stack For Your Software Development Project, Welcome to Intersog Your App Development Partner in Chicago, 2018. Available at: https://intersog.com/blog/tech-tips/how-to-choose-the-right-tech-stackfor- your-software-development-project/.
[10] C. Pirson, Optimization of a demand-responsive transit system, Master Thesis, University of Lige, 2017.
[11] Czarnowski, I., Caballero, A., Howlett, R. and Jain, L. "Intelligent Decision Technologies", 2017
[12] H. Yahyaoui, I. Kaabachi, S. Krichen and A. Dekdouk, Two metaheuristic approaches for solving the multi-compartment vehicle routing problem, Operational Research, 2018.
[13] N. Landwehr, M. Hall and E. Frank, Logistic Model Trees, Machine Learning, vol. 59, no. 1-2, pp. 161-205, 2005.
[14] C. Pornsing, A practice swarm optimization for the vehicle routing, Ph.D, University of Rhode island, 2014.
[15] P. Samaras, A. Fachantidis, G. Tsoumakas and I. Vlahavas, A prediction model of passenger demand using AVL and APC data from a bus fleet, Proceedings of the 19th Panhellenic Conference on Informatics – PCI '15, 2015.
[16] B. Rohrer, Brandon Rohrer - Instructor - End-to-End Machine Learning, Brohrer.github.io, 2017. Available at: https://brohrer.github.io/blog.html.
[17] V. Kachitvichyanukul, P. Sombuntham and S. Kunnapapdeelert, Two solution representations for solving multi-depot vehicle routing problem with multiple pickup and delivery requests via PSO, Computers & Industrial Engineering, vol. 89, pp. 125-136, 2015.
[18] W. Fan and R. Machemehl, using a Simulated Annealing Algorithm to Solve the Transit Route Network Design Problem, Journal of Transportation Engineering, vol. 132, no. 2, pp. 122-132, 2006.
[19] A. Karpathy, The Unreasonable Effectiveness of Recurrent Neural Networks, 2015. Available at: https://karpathy.github.io/2015/05/21/rnn-effectiveness/.
[20] P. Umrao, Intelligent Transportation System, Institute of Engineering & Technology, Lucknow, 2015.
[21] Q. Yang, L. Wang, W. Xia, Y. Wu and L. Shen, Development of on-board unit in vehicular adhoc network for highways, International Conference on Connected Vehicles and Expo (ICCVE), Vienna, pp. 457-462, 2014.
[22] Petracca, Matteo & Pagano, P & Pelliccia, Riccardo & Ghibaudi, Marco & Salvadori, Claudio & Nastasi, C. On-Board Unit hardware and software design for Vehicular Ad-hoc Networks. Roadside Networks for Vehicular Communications: Architectures, Applications, and Test Fields. 10.4018/978-1-4666-2223-4.ch002, 2012
[23] Bello, Irwan, Pham, Hieu, Le, Quoc V, Norouzi, Mohammad, and Bengio, Samy. Neural combinatorial optimization with reinforcement learning, 2016.
[24] Cho, Kyunghyun, Van Merri ënboer, Bart, Gulcehre, Caglar, Bahdanau, Dzmitry, Bougares, Fethi, Schwenk, Holger, and Bengio, Yoshua. Learning phrase representations using rnn encoder-decoder for statistical machine translation, Conference on Empirical Methods in Natural Language Processing, 2014.
[25] Clarke, Geoff and Wright, John W. Scheduling of vehicles from a central depot to a number of delivery points. Operations research, 12(4):568–581, 1964.
[26] Dai, Hanjun, Dai, Bo, and Song, Le. Discriminative embeddings of latent variable models for structured data, In International Conference on Machine Learning, pp. 2702–2711, 2016.
[27] Dai, Hanjun, Khalil, Elias B, Zhang, Yuyu, Dilkina, Bistra, and Song, Le. Learning combinatorial optimization algorithms over graphs, Advances in Neural Information Processing Systems, 2017.

[28] T. X. Brown, Low power wireless communication via reinforcement learning, In Advances in Neural Information Processing Systems, volume 12, pages 893–899, 1999.
[29] T. X. Brown, H. Tong, and S. P. Singh, optimizing admission control while ensuring quality of service in multimedia networks via reinforcement learning, In Advances in Neural Information Processing Systems, volume 12, pages 982–988, 1999.
[30] J. Carlstrom, Reinforcement Learning for Admission Control and Routing, PhD thesis, Uppsala University, Uppsala, Sweden, May 2000.
[31] E. Dijkstra, A note on two problems in connection with graphs, Numerical Mathematics, 1:269–271, 1959.
[32] L. P. Kaelbling, M. L. Littman, and A. W. Moore. Reinforcement learning: A survey, Journal of AI Research, 4:237–277, 1996.
[33] P. Marbach, O. Mihatsch, M. Schulte, and J. N. Tsitsiklis Reinforcement learning for call admission control and routing in integrated service networks, In Advances in Neural Information Processing Systems, volume 11, 1998.
[34] https://www.uber.com/en-EG/blog/introducing-uber-bus-a-new-way-to-commute/ (accessed Nov 1 2020)
[35] https://swvl.com/eg-en (Accessed Oct 1 2020)
[36] Active Selection Constraints for Semi-supervised Clustering Algorithms", International Journal of Information Technology and Computer Science(IJITCS), Vol.12, No.6, pp.23-30, 2020. DOI: 10.5815/ijitcs.2020.06.03
[37] Iram Mehmood, Sidra Anwar, AneezaDilawar, IsmaZulfiqar, Raja Manzar Abbas, "Managing Data Diversity on the Internet of Medical Things (IoMT)", International Journal of Information Technology and Computer Science(IJITCS), Vol.12, No.6, pp.49-56, 2020. DOI: 10.5815/ijitcs.2020.06.05
[38] Nur Kumala Dewi, " Review of Vehicle Surveillance Using Iot in the Smart Transportation Concept ", International Journal of Engineering and Manufacturing (IJEM), Vol.11, No.1, pp. 29-36, 2021. DOI: 10.5815/ijem.2021.01.04
[39] Anna Merine George, S.Y Kulkarni, Vice Chancellor,"Cluster based Routing Protocols for IOT Application", International Journal of Computer Network and Information Security(IJCNIS), Vol.11, No.5, pp.43- 49, 2019.DOI: 10.5815/ijcnis.2019.05.06
[40] Nur Kumala Dewi, " Review of Vehicle Surveillance Using Iot in the Smart Transportation Concept ", International Journal of Engineering and Manufacturing (IJEM), Vol.11, No.1, pp. 29-36, 2021. DOI: 10.5815/ijem.2021.01.04

Authors' Profiles

Dr. Samer Ibrahim was born in Gharbia, in 1976. He received the B.Sc. degree in computer engineering from the Ain Shams University, Egypt, in 1998, and the M.Sc. and Ph.D. degrees in Computer and system engineering from the computer and systems department, Ain Shams University, Egypt 2005 and 2009, respectively. He obtained his fellowship from Higher Education academy (HEA), UK in 2017. He is currently an Associate professor in MSA. Through his journey, Dr. Samer has many publications, patents researches, and books in international conferences and journals. Dr. Samer is a reviewer for many reputable journals and conference world-wide. He is an associate editor in an international journal of computer science and engineering. He has different roles through this journey not only on the academic level but on the industrial level as well.

Dr. Samer has 20+ years of IT industrial experience in many international organizations as consultant and executive manager. Being a PMP certified for 12+ and ITIL certified improved his academic, industrial, and practical experience in leading and managing critical industrial projects in different technical domains. Dr. Samer traveled several times to foreign countries which offered better understanding of many cultures from east and west, enhanced his ability to get along with different nationalities and ethnic backgrounds. Through this industrial journey, he led many critical and biggest transformation projects/programs that affected the IT industry worldwide in multinational organization like Hewlett Packard with huge record of success stories/records.

Mohamed Abdelhady Ghonaim is a teaching assistant at the department of Electrical Communications and Electronics at the October University of Modern Sciences and Arts. His research interests are in the field of communication and embedded systems design as well as the use of machine learning to increase the efficiency of such systems.

How to cite this paper: Samer I. Mohamed, Muhamed Abdelhadi, "IoT Bus Navigation System with Optimized Routing using Machine Learning", International Journal of Information Technology and Computer Science(IJITCS), Vol.13, No.3, pp.1-15, 2021. DOI: 10.5815/ijitcs.2021.03.01