# A new low complexity bus travel time estimator for fleet management system

**Emad Alizade** ( e.alizadeh@ec.iut.ac.ir )
Isfahan University of Technology
**Ali Hendessi**
Isfahan University of Technology
**Faramarz Hendessi**
Isfahan University of Technology
**Massoud Reza Hashemi**
Isfahan University of Technology

**Research Article**

**Keywords:** transportation, estimation, distributed, complexity

**Posted Date:** October 18th, 2021

**DOI:** https://doi.org/10.21203/rs.3.rs-979905/v1

# A new low complexity bus travel time estimator for fleet management system

Emad Alizade, Ali Hendessi, Faramarz Hendessi, Massoud Reza Hashemi

Department of Electrical and Computer Engineering, Isfahan University of Technology, Iran

e.alizadeh@ec.iut.ac.ir, hendessi@gmail.com, hendessi@cc.iut.ac.ir, hashemim@cc.iut.ac.ir

**Abstract**

**Improving the experience of using the public transportation system can be done by estimating the arrival time of the bus and notifying the passengers. Consequently, the accuracy of the estimation affects this experience. As the number of buses, stations, and service areas increases, so does the data collected in the cloud, making travel time estimation-related data processing more challenging. Despite this challenge, a distributed method for estimating the arrival time of the bus is considered in this paper. Also, we present a way to decentralize data processing and distribute it on each bus. Besides, using the Kalman filter and updating the estimated values at short intervals improves the estimation error. Examination of the degree of complexity shows that the proposed method has significantly reduced the complexity in the cloud, which makes the proposed method can be implemented in metropolitan areas. The results of implementation on a dataset, show that the proposed method has a good performance in terms of mean square error and root mean square.**

**Key words: transportation, estimation, distributed, complexity**

## I. Introduction

In recent years, with the development of wireless telecommunication network technology and various applications, including the Internet of Things, special attention has been paid to vehicle communications [1]. Even to the extent that standards have been set for communication between

vehicles, vehicles with infrastructure, and vehicles in general with everything. On the other hand, the issue of the smart city is another topic of interest to researchers in recent years [2], [3].

The intelligent transportation system, as part of the smart city, uses the vehicle communication infrastructure to collect information properly. This system collects large data sets. The processing and security of this information have challenges that some papers have addressed [4-8].

An Intelligent Transportation System plays an important role in the passengers' experience. In this system, each bus collects some information. Processing this information in a server is one of the challenging tasks which must satisfy sustainability, accessibility, and economic development as the main objectives [9]. This information can be used for bus travel time estimation for passengers or traffic congestion reduction.

Many environmental factors affect the travel of a bus. Some of these factors, such as weather conditions in different seasons, weekends and daylight hours, change periodically, and some other factors such as the occurrence of natural disasters, how the driver drives, the bus breakdown, the number of passengers at each station, the traffic condition of each street and the geographical area are completely random. These are the reasons why we are facing a complex system that is difficult to estimate the travel time of each bus[10]. The estimator must also have a reasonable computational complexity for the server to be able to process data of a large number of buses.

The Fleet Management System is a server that works with Automated Vehicle Location (AVL) to determine the estimated time for each bus to arrive at each station. The information on each bus is collected and then processed on the server. The estimated time for each bus to arrive is classified into two general categories. In the first category, each bus estimates its arrival time based on its previous information and the information of that route, and in the second category, the information of all buses and routes collects at the server and the server performs an estimating operation using a neural network. Although neural network based methods and machine learning are more accurate, they are not practical for large cities with a large number of buses due to their high complexity.

The main contributions of this paper are summarized as follows:

- In this paper, we present a distributed method for estimating the bus arrival time based on the Kalman filter. In the proposed method, each bus estimates its arrival time at the next station using the Kalman filter. Estimating the bus arrival time includes two steps: coarse estimation and fine estimation. In order to computation offloading, the estimation algorithm is distributed among the buses and intermediate calculations are stored in the server for the estimated time calculation and estimated and are displayed to users if needed.
- Unlike other works, in this paper, computational complexity for the server has been calculated. Inspired by fog computing, the estimation algorithm is distributed in such a way that the server performs the least number of calculations. The computational complexity charts are presented in terms of estimated time accuracy, number of buses, and number of stations.

The rest of this paper is organized as follows. Section II reviews current researches in bus travel time estimation whereas section III gives a detailed description of proposed model. In section IV we compare our fog-based estimation with the centralized method in terms of complexity. Section V represents the proposed method performance with the real-world data set. Finally, Section VI concludes the paper and reviewing the main contributions.

## II. Literature Review

The fleet management system includes sections such as a server as a management center, AVL, bus sending and receiving unit, radio system, bus GPS unit and on-board computer on the bus. The necessary information (including GPS information and other information required by the management center) is prepared by the on-board computer on the bus and sent to the server using the sending and receiving unit through the radio system, which can be a cellular communication network. In AVL, the location information of the bus is extracted and given to the management center [11]. Automatic passenger counting system introduced in [12], this system is responsible for extracting the information of the boarding and alighting passengers at each station [13]. Estimating or predicting bus arrival times, also known as bus travel time estimates, is influenced by variables such as the number of passengers, the number of stations and the length of the route,

the environmental conditions and the way you drive. The relationship between these factors and their effect on bus arrival time is discussed in [14].

Traffic flow is an effective factor in estimating the arrival time of the bus. Extensive research has been done in this area. Most of these researches collect large data from buses and apply the proposed processing method to it. Machine learning-based methods are a major part of this researches. For example [15] introduces a method for estimating traffic flow that has been done using deep belief network and multitask learning model. In [16] another method based on restricted Boltzmann machine and recurrent neural network is used for the same purpose. Genetic algorithm-based methods are also used in [17] and [18] to predict traffic flow and traffic congestion.

Incidents are another effective factor in creating traffic congestion and affecting the arrival time of the bus, which has been discussed in some articles [19-21]. Although traffic and incident analysis may help bus arrival time estimates improvement, they add double processing complexity and overhead to the estimator system. However, this improvement may not be significant.

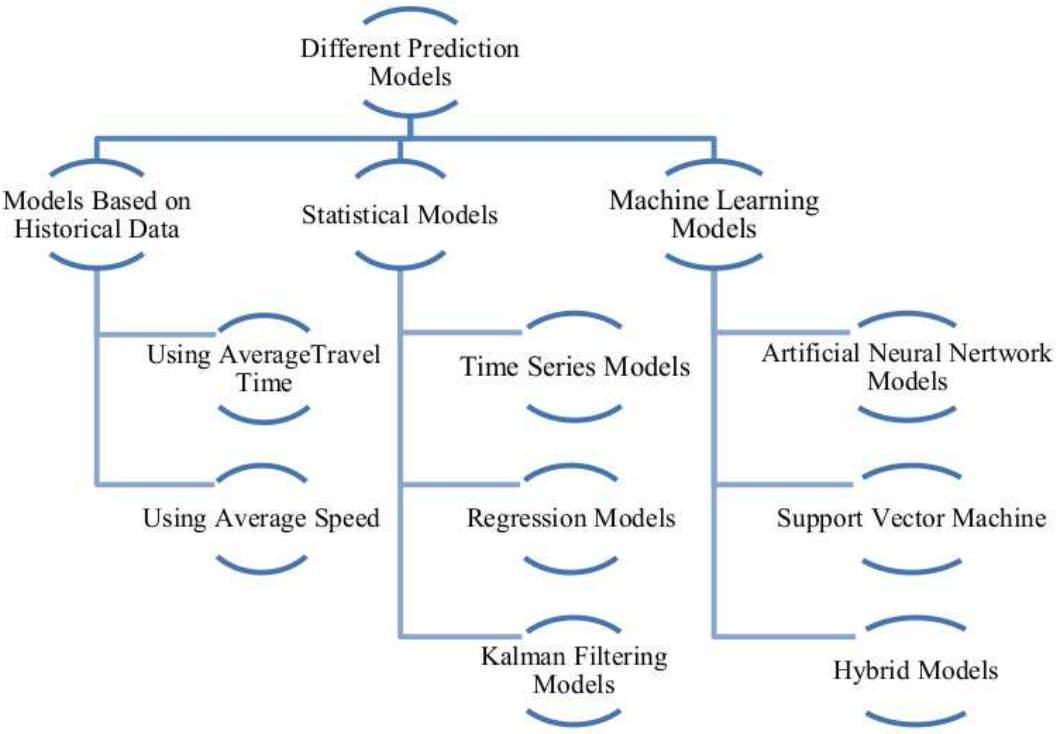


*Figure 1: Bus travel time prediction models[22].*

As estimating a parameter in the desired system, there are different ways to estimate the bus arrival time. These methods fall into three categories. In the first category, estimates are based on bus history. The second category uses statistical methods for estimation. In the third category, methods based on machine learning and neural network are used[22].

One of the easiest ways to estimate the arrival time of a bus is to use the historical data of each route mentioned in [23-25]. In this method, the average historical data for each path is considered as the estimated value for that path. Although this method is a bit complicated, the estimate is inaccurate and reacts slowly to events such as accidents. As a result, more accurate estimation methods are needed.

In [26] Kalman Filter-based bus travel time estimator proposed. According to this method, each path divided into some subsections equal length parts and Kalman filter used for travel time estimation in each section. The accuracy of this method is proportional to the number of sections.

This paper does not discuss the computational complexity of the server which depends on the number of sections. Also dwell time is not estimated in this paper.

In [9], authors estimate travel time for vehicles other than buses has been made with the graph-based method. This method is not suitable for estimating bus travel time because the dwell time is not considered at bus stops. On the other hand, the application of the proposed method is for estimating traffic.

In [27] Kalman Filter based bus travel time prediction and artificial neural network approach compared. Although the experiment results show that the neural network has a more accurate estimate, the need for a large dataset to train the network makes Kalman filter preferable.

A log-normal auto-regressive modeling approach introduced in [28]. In this paper, the complexity of the estimation method is high due to the concentration of calculations in the server, which makes the server need a lot of memory and a very powerful processor.

To estimate the arrival time of buses, the server processes large data in metropolitan areas where the number of buses and the number of stations is high. In large data applications, common data processing methods are inefficient and newer techniques are needed to process data properly [29]. When we are estimating the bus arrival time, real-time processing of this big data is especially important because delays in calculations increase the estimation error.

Data loss is one of the challenges of the intelligent transportation system that may be faced for various reasons. This may overshadow the estimated values and reduce the performance quality of the estimation algorithm. In [30] the authors used machine learning to estimate the lost data. This estimated data, along with other data, completes the data set to make the bus arrival time estimation algorithm more efficient.

## III. Proposed Method

Figure 2 shows the architecture of the proposed method. According to this architecture, the bus has GPS modules, local database, transmit and receive module for exchanging information with servers, timers and implemented algorithms. The server also includes a transmit and receive module for exchanging information with buses, AVL, arrival time calculation unit and back-end

processing for other tasks such as preparing information to end users. The server is connected to the database to store and retrieve information.

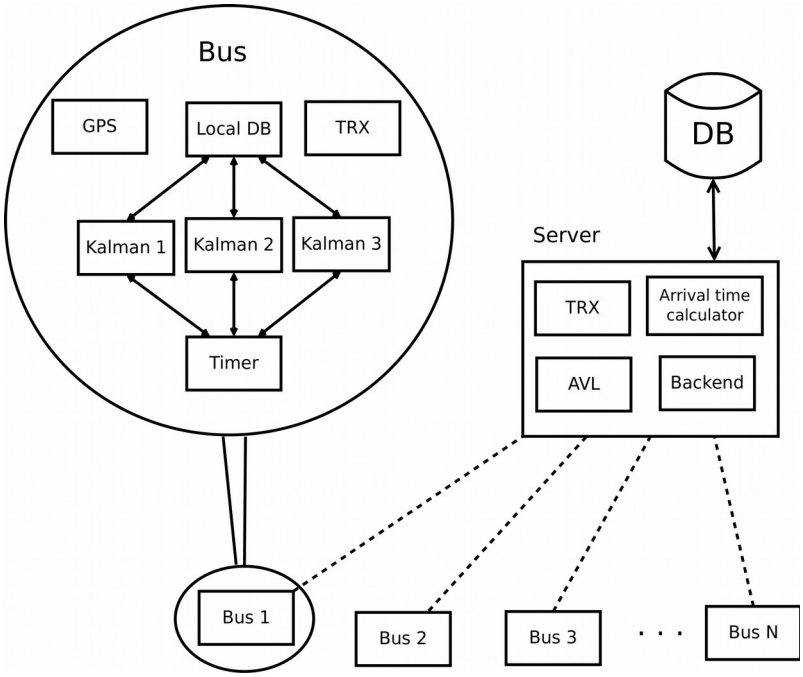


*Figure 2: Proposed method architecture.*

Figure 3 shows the flowchart of the proposed distributed algorithm that must be performed on each bus. In fact, every bus should be informed of its position by taking feedback from GPS and run the appropriate algorithm. We need 3 types of estimation algorithms to estimate the arrival time of the station, the travel time between two stations estimation which we called it link travel time estimation, the time to reach the next station estimation (in the distance between the two stations) which we called it mini link travel time estimation, and the dwell time at the station estimation.

According to the flowchart, each bus checks its location. If the bus reaches the station, it takes the parameters of the last Kalman filter from the server. These parameters are related to the link travel time estimation. After running the Kalman filter, the values of the parameters and the link travel time estimation are sent to the server. The server updates these parameters in the database.

Therefore, the value registered in the database is the latest updated values. If the link travel time is required for viewing, the server extracts the estimated value from the database.

Then the location of the bus is checked to determine the exit from the station. After leaving the station, Algorithm 2 is run to estimate the dwell time. The necessary parameters to run Algorithm 2 are received from the server. If the bus is at the terminal, there is nothing to do. Otherwise, the T3 timer is started and Algorithm 3 runs when it expires.

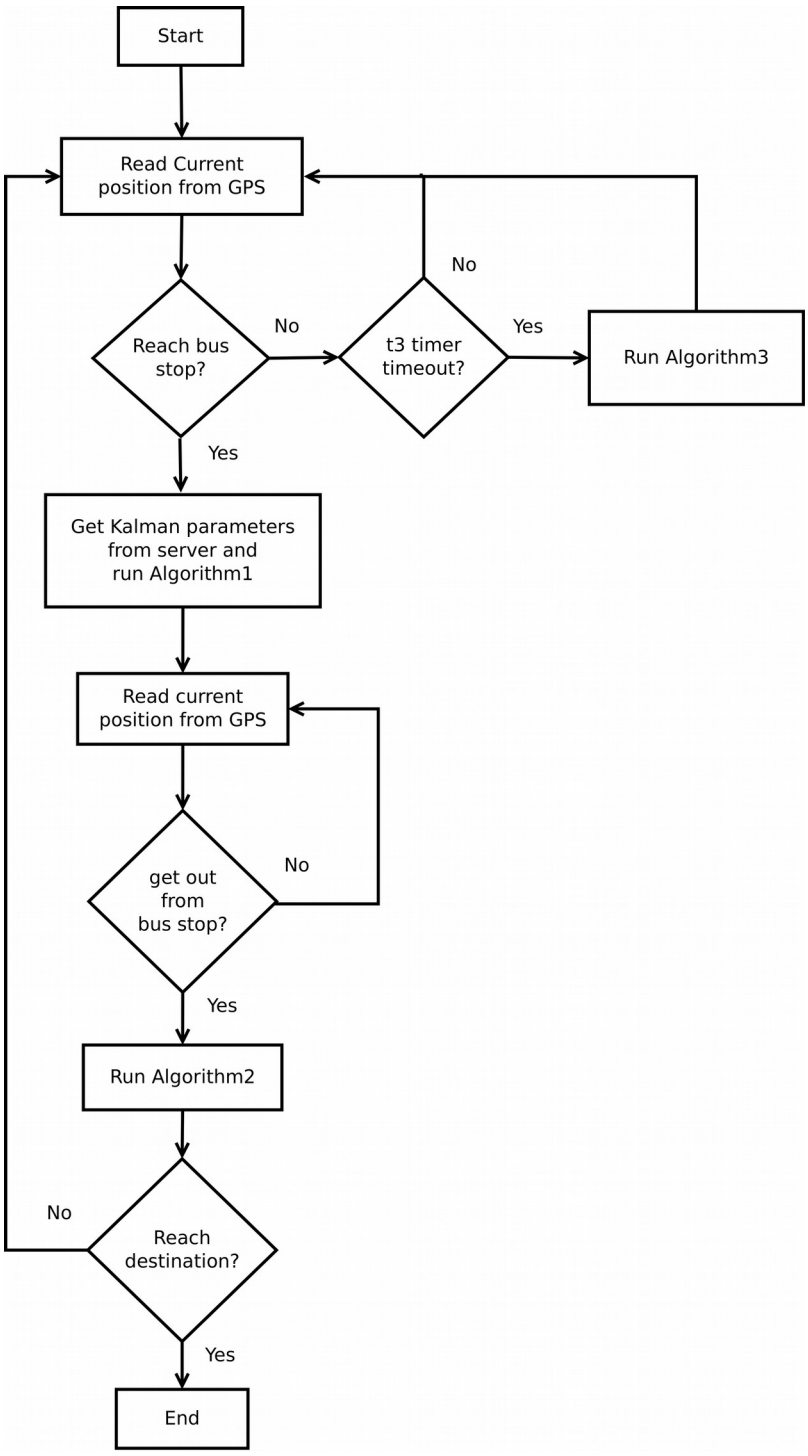


*Figure 3: Flow chart of each fog node.*

Based on the Kalman algorithm, to estimate $\hat{t}_{n+1}$ with the latest observation and the average of previous observations, a weighted linear combination of those calculated as

$$\hat{t}_{n+1} = g_{n+1}\,\bar{t}_n + (1 - g_{n+1})\,t_n. \qquad (1)$$

where $g_{n+1}$ is the Kalman filter gain for $n+1$ th time index, $t_n$ is the latest observation in the $n$ th time index and $\bar{t}_n$ is the average of previous observations until the $n$ th time index. At each step, Kalman filter gain must be updated as

$$g_{n+1} = \frac{e_n + \sigma_n^2}{e_n + 2\sigma_n^2}. \qquad (2)$$

where $e_n$ and $\sigma_n^2$ are Kalman filter error and variance of previous observations respectively. Like the Gain Kalman filter, the filter error, average and variance of previous observations also must be updated as

$$e_{n+1} = g_{n+1}\,\sigma_n^2 \qquad (3)$$

$$\sigma_{n+1}^2 = \frac{(t_n - \bar{t}_n)^2 + n\,\sigma_{n+1}^2}{n+1} \qquad (4)$$

$$\bar{t}_{n+1} = \frac{n\,\bar{t}_n + t_n}{n+1}. \qquad (5)$$

According to the flowchart of fig. 3, to implement the proposed method, each fog node runs Algorithm 1 when it reaches the station. This algorithm is responsible for updating the time estimate between two stations. Then the updated value is sent to the server. The initial values of the estimation process are randomly selected. After some steps, the algorithm converges to the actual value.

---

Algorithm 1

---

**Input:** last parameters of Kalman for link travel time estimation ($e_n$, $\sigma_n^2$, $\bar{t}_n$ and $n$).

**Output:** updated parameters and estimated value.

1: Stop timer $T_1$ and let $t_n \Leftarrow T_1$.

2: Calculate $g_{n+1}$ from eq. (2).

---

3: Calculate $\hat{t}_{n+1}$ from eq. (1).

4: Update $e_{n+1}$, $\sigma^2_{n+1}$ and $\bar{t}_{n+1}$ from eq. (3-5) and let $n \Leftarrow n+1$.

5: Start timer $T_2$.

The fog node updates the estimated dwell time when leaving the station. This is done using Algorithm 2. Similarly, this algorithm is also distributed in fog nodes and the server is solely responsible for storing and reading updated values.

Algorithm 2

**Input**: last parameters of Kalman for dwell time estimation ($e_n$, $\sigma^2_n$, $\bar{t}_n$ and $n$).

**Output**: updated parameters ($e_{n+1}$, $\sigma^2_{n+1}$ and $\bar{t}_{n+1}$) and estimated value ($\hat{t}_{n+1}$).

1: Stop timer $T_2$ and let $t_n \Leftarrow T_1$.

2: Calculate $g_{n+1}$ from eq. (2).

3: Calculate $\hat{t}_{n+1}$ from eq. (1).

4: Update $e_{n+1}$, $\sigma^2_{n+1}$ and $\bar{t}_{n+1}$ from eq. (3-5) and let $n \Leftarrow n+1$.

5: Start timer $T_3$.

6: Let $m \Leftarrow 0$.

Finally, Algorithm 3 runs after the bus leaves the station at specific times. This algorithm helps to update the estimated time of arrival at the station. In a similar way, Algorithm 3 estimates the average link travel time by

$$\hat{v}_{m+1} = g_{m+1}\bar{v}_m + (1 - g_{m+1})v_m, \quad (6)$$

$$g_{m+1} = \frac{e_m + \sigma^2_m}{e_m + 2\sigma^2_m}, \quad (7)$$

$$e_{m+1} = g_{m+1}\sigma^2_m, \quad (8)$$

$$\sigma^2_{m+1} = \frac{(v_m - \bar{v}_m)^2 + m\sigma^2_{m+1}}{m+1}, \quad (9)$$

$$\bar{v}_{m+1}=\frac{m\bar{v}_m+v_m}{m+1}. \qquad (10)$$

where $m$ is time index, and $v$ is speed of the bus. So fine estimation of link travel time calculated as

$$\hat{t}_{m+1,mL}=\frac{\Delta x_L}{\hat{v}_{m+1}}. \qquad (11)$$

where $\Delta x_L$ is the length of link $v$ L. Therefore two estimates are available for link travel time. Finally to increase the accuracy of the estimate, the link travel time is updated by the

$$\hat{t}_{n+1}=f_{m+1}\hat{t}_{n+1,L}+(1-f_{m+1})\hat{t}_{m+1,mL}, \qquad (12)$$

where $\hat{t}_{n+1,L}$ is the coarse estimation which obtained from algorithm 1 and

$$f_{m+1}=\begin{cases}0, mT_{up}<\hat{t}_{n+1,L}\\ \dfrac{mT_{up}}{\hat{t}_{n+1,L}}, else\end{cases}. \qquad (13)$$

Algorithm 3 stores the intermediate calculation parameters in the bus but sends the final value of the link estimate to the server.

---

Algorithm 3

---

Input: last parameters of Kalman for mini link speed estimation ($e_m$, $\sigma_m^2$ and $\bar{v}_m$), $m$, $T_{up}$ and $\hat{t}_{m+1}$.

Output: updated parameters ($e_{m+1}$, $\sigma_{m+1}^2$ and $\bar{v}_{m+1}$) and estimated value ($\hat{t}_{n+1}$).

1: Calculate $\Delta x_m$ and let $v_m \Leftarrow \frac{\Delta x_m}{T_{up}}$.

2: Calculate $g_{m+1}$ from eq. (7).

3: if $\left(mT_{up}<\hat{t}_{n+1,L}\right)$ then

4: Let $f_{m+1}\Leftarrow 0$.

5: else

---

6: Let $f_{m+1} \Leftarrow \frac{mT_{up}}{\hat{t}_{n+1,L}}$.

7: end if

3: Calculate $\hat{v}_{m+1}$ from eq. (6) and $\hat{t}_{n+1}$ from eq. (12)

4: Update $e_{m+1}$, $\sigma^2_{m+1}$ and $\bar{v}_{m+1}$ from eq. (8-10) and let $m \Leftarrow m+1$.

5: Restart timer $T_3$.

## IV. Evaluation

In this section, we evaluate the proposed method. Due to the fog-based nature of the proposed method, the evaluations are divided into two parts. In the first part, the complexity of the proposed method is evaluated in terms of calculations and the number of variables. In the second part, the bus arrival time is estimated using the data set provided in the book.

- **Complexity Analyze**

The proposed method transfers the calculations from the cloud to the node nodes. In fact, it leads to the distribution of computations. So we compare the complexity of the proposed method with previous methods that are centralized. Also, due to the time-consuming reading and writing of information in the database, we compare the number of writes and reads in the database.

In centralized methods, the server must perform all the processing related to each Kalman filter, while in the distributed method, no processing related to the Kalman filter is performed on the server. Also, in the proposed method, the result of intermediate calculations is stored in each node, which reduces the variables read and written in the database. Let $N$, $M$, $T_L$, and $T_{up}$ be number of buses, number of bus stops in path, average link travel time and estimation update time in each link respectively. Table 1 shows the comparison details of the proposed method and the centralized method. Note that each Kalman filter has six variables. These variables must be stored as intermediate calculations when the Kalman filter has been executed.

Table 1: complexity analysis of proposed method and centralized methods.

| Parameter | Centralized method | Proposed method |
|---|---|---|
| Number of Kalman filters in server | $N(M-1)+MN+\left[\frac{T_L}{T_{up}}\right]^{M-1}$ | – |
| Number of variables that must be saved in database | $6N(M-1)+6MN+\left[\frac{6T_L}{T_{up}}\right]^{M-1}$ | $6N(M-1)+6MN+\left[\frac{T_L}{T_{up}}\right]^{M-1}$ |
| Number of reads from database | $5N(M-1)+5MN+\left[\frac{5T_L}{T_{up}}\right]^{M-1}$ | $5N(M-1)+5MN$ |
| Number of writes to database | $6N(M-1)+6MN+\left[\frac{6T_L}{T_{up}}\right]^{M-1}$ | $6N(M-1)+6MN+\left[\frac{T_L}{T_{up}}\right]^{M-1}$ |

Figure 4 shows the number of variables stored in the database for the proposed method and centralized methods. According to the figure, the proposed method has exponentially reduced the number of variables. This reduces memory and thus speeds up access to variables.

As shown in Figure 4, increasing the number of algorithm 3 estimates exponentially reduces the number of variables. The proposed method reduces the number of variables by drastically reducing the number of variables related to intermediate calculations in Algorithm 3. Because the intermediate variables in Algorithm 3 are stored in the node, there is no need to store them in the cloud.

- **Estimation performance**

The data set provided in [31] has been used to evaluate the method of estimating the arrival time of the bus. This database includes bus locations with timestamps in the Helsinki city of Finland. The Historical Average (HA) method is one of the simplest and least complicated methods for estimating bus travel time. Therefore, we compare the proposed method with this method. Figure 5 shows the Mean Square Error (MSE) of the proposed method. As can be seen from the figure 5, with increasing samples, the estimator performance improves and the proposed method has less error even in low iteration numbers. After 10 iterations, the error of the proposed method

will be a maximum of 15%. While method HA still has a high error after even calculating with 20 samples.

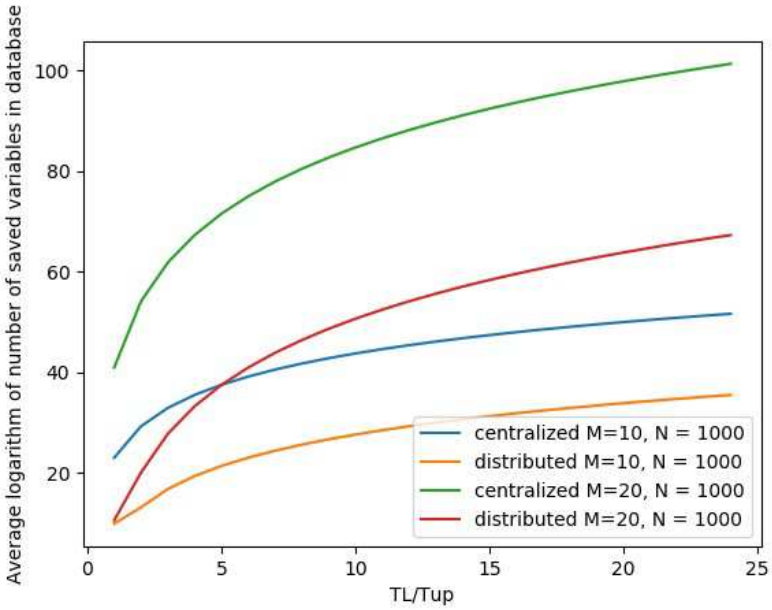


*Figure 4: Number of saved variables for distributed and centralized methods.*

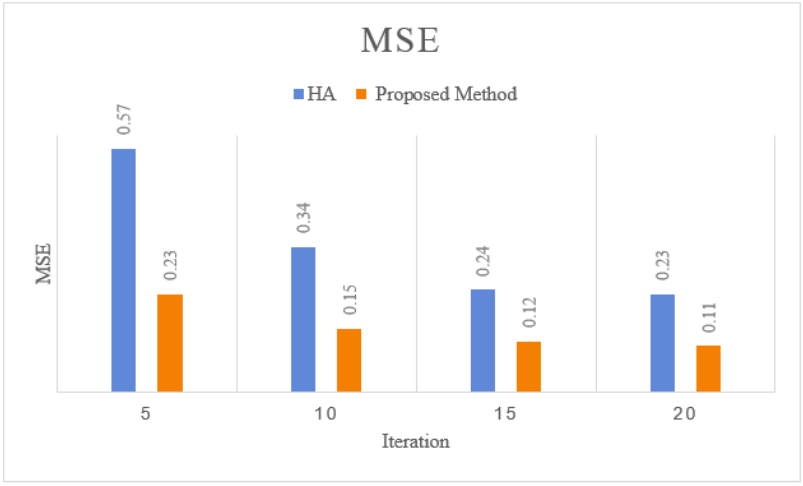


*Figure 5: MSE of the proposed method compared with HA in different iterations.*

In addition to depending on the number of iterations, MSE also depends on the time. In other words, in the hours when there is a possibility of traffic and the traffic is more unpredictable, the amount of estimation error will be different. Figure 6 shows the amount of female estimation error at different times of the day. The proposed method works well at different hours and has the least error at 11:00 to 12:00 AM. The HA method, on average, has almost twice the error of the proposed method.

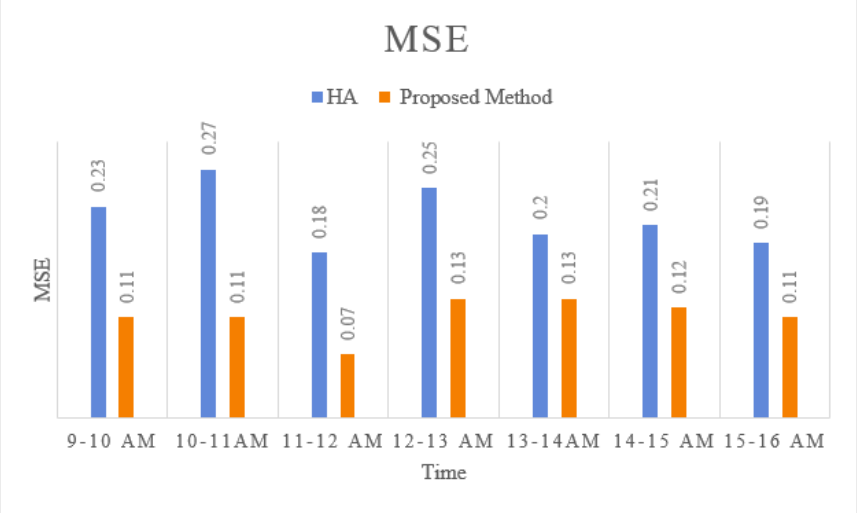


*Figure 6: MSE of proposed method and HA method in different hours.*

The MSE metric determines the error rate. Therefore, this metric may not be appropriate in some cases. For example, a low error rate may be a large error. Another metric that indicates the amount of error is Root Mean Square Error (RMSE) and is defined as

$$RMSE = \sqrt{\frac{1}{N}\sum_{r=1}^{N}\left(t_{ac,r} - \hat{t}_r\right)^2}. \qquad (14)$$

where $N$ is number of iterations and $t_{ac,r}$ is actual travel time in $r$ th iteration. Also $\hat{t}_r$ is travel time estimation in $r$ th iteration. Unlike the MSE metric, which is normalized to the actual value, we normalize this metric along the path per kilometer. Figure 7 shows the average RMSE graph in terms of the number of iterations. According to this figure, the proposed method has an average error of fewer than 5 seconds per kilometer after 10 repetitions. The HA method has more errors and may even take more than 60 seconds.

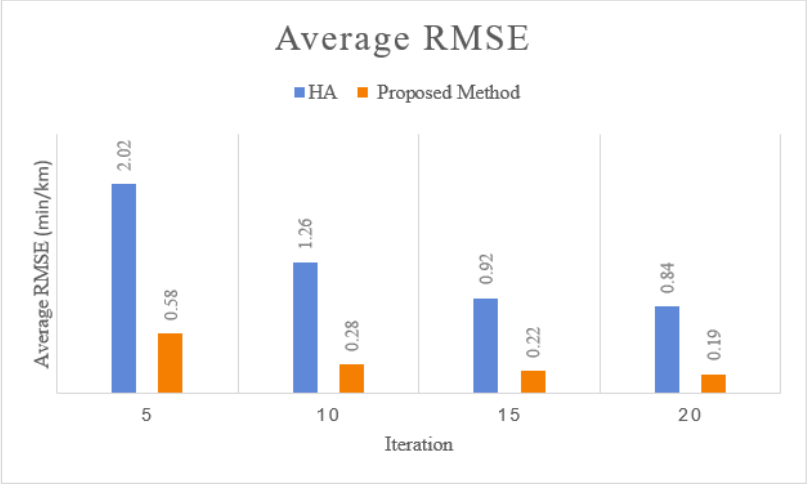


*Figure 7: Average RMSE per iteration for proposed method and HA method.*

The results in Figure 8 show that the proposed method performs well at different times of the day in terms of average RMSE. It should be noted that although in Figure 6, at 11 to 12 o'clock, the estimator performance is good from the RMSE point of view, from the average RMSE point of view in this time period, it does not perform well. Prolonged travel time during this period has caused the amount of MSE to be lower compared to other time periods.

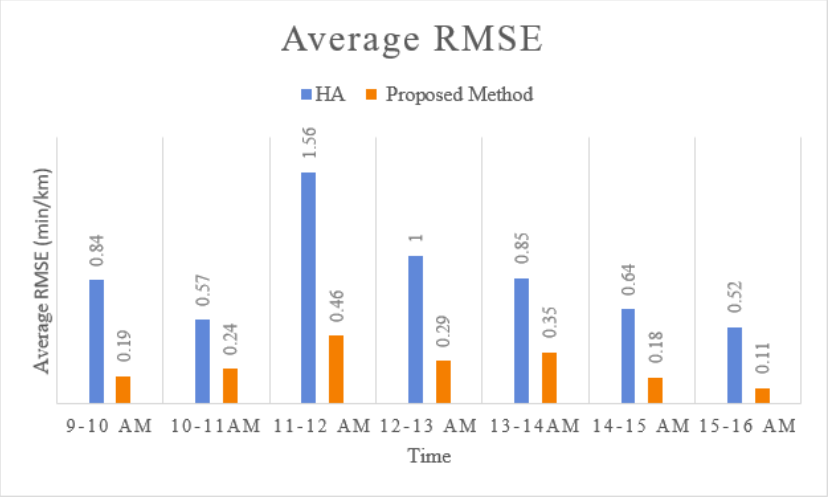


*Figure 8: Average RMSE of proposed method and HA in different times.*

## V. Conclusion

Estimating the arrival time of the bus is of great importance in the intelligent transportation system. With increasing in the number of buses and stations, a large amount of data reaches the cloud. Cloud must process them properly. Cloud processing is not just about estimating processes. Therefore, reducing the processing load contributes to the overall performance of the intelligent transportation system. In this paper, the Kalman filter-based algorithm is implemented and distributed on each bus. So, each bus is responsible for processing part of the data. The results of the complexity analysis showed that the proposed method has well reduced the cloud processing load. Besides, the number of cloud accesses to the database has been reduced by eliminating the exponential term in fine estimation. The results of implementing the proposed method on a data set also showed that each bus updates the estimated value by performing its calculations, while the estimated value is close to the actual value.

**Compliance with ethical standards**

**Conflict of interest:** The authors have no conflict of interest regarding the publication of this article.

**Ethical approval:** This article does not contain any studies with human participants or animals performed by any of the authors.

**Author Contributions:** Study concept and design: E. A., A. H., and F. H.; Developed the original idea and the protocol, abstracted and analyzed data, wrote drafting of the manuscript: E. A. Performed the simulations: E. A.; Critical revision of the manuscript for important intellectual content and Study supervision: F. H., and M. R. H.

**References**

[1] A. Sumalee, and H. W. Ho, “Smarter and more connected: Future intelligent transportation system.” Iatss Research, vol. 42, no. 2, pp. 67-71. 2018.

[2] H. Menouar, I. Guvenc, K. Akkaya, A. S. Uluagac, A. Kadri, and A. Tuncer, “UAV-enabled intelligent transportation systems for the smart city: Applications and challenges.” IEEE Communications Magazine, vol. 55, no. 3, pp. 22-28. 2017.

[3] P. Sun, and A. Boukerche, “AI assisted data dissemination methods for supporting intelligent transportation systems.” Internet Technology Letters, 2021.

[4] S. Kaffash, A. T. Nguyen, and J. Zhu, “Big data algorithms and applications in intelligent transportation system: A review and bibliometric analysis.” International Journal of Production Economics, 2021.

[5] W. Alajali, W. Zhou and S. Wen, "Traffic Flow Prediction for Road Intersection Safety," 2018 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI), Guangzhou, China, pp. 812-820, 2018.

[6] A. Lamssaggad, N. Benamar, A. S. Hafid, and M. Msahli, “A Survey on the Current Security Landscape of Intelligent Transportation Systems.” IEEE Access, pp. 9180-9208, 2021.

[7] A. Mohandu, and M. Kubendiran, “Survey on Big Data Techniques in Intelligent Transportation System (ITS).” Materials Today: Proceedings, 2021.

[8] G. Hatzivasilis, K. Fysarakis, S. Ioannidis, I. Hatzakis, G. Vardakis, N. Papadakis, and G. Spanoudakis, “SPD-Safe: Secure Administration of Railway Intelligent Transportation Systems.” Electronics, vol. 10, no. 1, 2021.

[9] A. Salamanis, D. D. Kehagias, C. K. Filelis-Papadopoulos, D. Tzovaras, and G. A. Gravvanis, “Managing spatial graph dependencies in large volumes of traffic data for travel-time prediction.” IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 6, pp. 1678-1687, 2015.

[10] H. P. Jiang, G., Lam, S. K. and Y. Sun, “Learning heterogeneous traffic patterns for travel time prediction of bus journeys,” Information Sciences, pp. 1394-1406, 2020.

[11] G. Guido, A. Vitale, and D. Rogano, “Assessing public transport reliability of services connecting the major airport of a low density region by using AVL and GIS technologies,” in Proceedings of IEEE 16th International Conference on Environment and Electrical Engineering (EEEIC), 2016.

[12] S. Elkosantini, and S. Darmoul, “Intelligent public transportation systems: A review of architectures and enabling technologies,” in Proceedings of IEEE International Conference on Advanced Logistics and Transport, pp. 233-238, 2013.

[13] G. V. Sundar, and B. G. Rajagopal, “IoT based passenger information system optimized for Indian metros,” in Proceedings of IEEE International conference of Electronics, Communication and Aerospace Technology (ICECA), vol. 1, pp. 92-96, 2017.

[14] G. Chen, X. Yang, J. An, and D. Zhang, “Bus-Arrival-Time Prediction Models : Link-Based and Section-Based,” J. Transp. Eng., vol. 138, no. 1, pp. 60–66, 2012.

[15] H. Dia, “An object-oriented neural network approach to short-term traffic forecasting.” European Journal of Operational Research, vol. 131, no. 2, pp. 253-261, 2001.

[16] X. Ma, H. Yu, Y. Wang, and Y. Wang, “Large-scale transportation network congestion evolution prediction using deep learning theory.” PloS one, vol. 10, no. 3, 2015.

[17] P. Lopez-Garcia, E. Onieva, E. Osaba, A. D. Masegosa, and A. Perallos, “A hybrid method for short-term traffic congestion forecasting using genetic algorithms and cross entropy.” IEEE Transactions on Intelligent Transportation Systems, vol. 17, no. 2, pp. 557-569, 2015.

[18] E. I. Vlahogianni, M. G. Karlaftis, and J. C. Golias, “Optimized and meta-optimized neural networks for short-term traffic flow prediction: A genetic approach.” Transportation Research Part C: Emerging Technologies, vol. 13, no. 3, pp. 211-234, 2005.

[19] H. Dia, and G. Rose, “Development and evaluation of neural network freeway incident detection models using field data.” Transportation Research Part C: Emerging Technologies, vol. 5, no. 5, pp. 313-331, 1997.

[20] X. Jin, D. Srinivasan, and R. L. Cheu, “Classification of freeway traffic patterns for incident detection using constructive probabilistic neural networks.” IEEE Transactions on Neural networks, vol. 12, no. 5, pp. 1173-1187, 2001.

[21] C. H. Wei, and Y. Lee, “Sequential forecast of incident duration using artificial neural network models.” Accident Analysis and Prevention, vol. 39, no. 5, pp. 944-954, 2007

[22] A. Aldokhayel, *A Kalman Filter-based Dynamic Model for Bus Travel Time Prediction,* Master thesis, University of Ottawa, 2018.

[23] H. Yu, R. Xiao, Y. Du, and Z. He, "A Bus-Arrival Time Prediction Model Based on Historical Traffic Patterns," in Proceedings of IEEE International Conference on Computer Sciences and Applications, 2013.

[24] M. As and T. Mine, "Dynamic Bus Travel Time Prediction Using an ANN-based Model," in Proceedings of the 12th International Conference on Ubiquitous Information Management and Communication, 2018.

[25] S. Maiti, A. Pal, A. Pal, T. Chattopadhyay, and A. Mukherjee, "Historical Data based Real Time Prediction of Vehicle Arrival Time," in The proceedings of IEEE 17th International Conference on Intelligent Transportation Systems (ITSC), 2014.

[26] L. Vanajakshi, S. C. Subramanian, and R. Sivanandan, "Travel time prediction under heterogeneous traffic conditions using global positioning system data from buses," IET intelligent transport systems, vol. 3, no. 1, pp. 1-9, 2009.

[27] V. Kumar, B. A. Kumar, L. Vanajakshi, S. C. Subramanian, "Comparison of model based and machine learning approaches for bus arrival time prediction," In Proceedings of the 93rd Annual Meeting, pp. 14–2518, 2014.

[28] B. Dhivya Bharathi, B. Anil Kumar, A. Achar, and L. Vanajakshi, "Bus travel time prediction: a log-normal auto-regressive (AR) modeling approach," Transportmetrica A: Transport Science, vol. 16, no. 3, pp. 807-839, 2020.

[29] L. Zhu, F. R. Yu, Y. Wang, B. Ning, T. Tang, "Big Data Analytics in Intelligent Transportation Systems: A Survey," IEEE Trans. Intell. Transp. Syst., vol. 20, 383–398, 2019.

[30] B. Najafi, S. Parsaeefard, and A. Leon-Garcia, "Estimation of Missing Data in Intelligent Transportation System." *arXiv preprint arXiv:2101.03295*, 2021

[31] TrafficSense Aalto, "public-transport-dataset," 2016, [online], Available: https://github.com/aalto-trafficsense/public-transport-dataset, [Accessed: 03-Jun-2021].