Chapter 3
Smart City: An Intelligent Automated Mode of Transport Using Shortest Time of Travel Using Big Data

Mashrin Srivastava, Suvarna Saumya, Maheswari Raja, and Mohana Natarajan

3.1 Introduction

With the focus shifting to revolutionizing the way of life by introducing 'smart' technology to analyse and predict needs or reduce the man power required to complete jobs and the onset of emerging technology in the field of big data analytics and cloud computing in this age of technology, the area of focus in our proposal is automated transportation. The proposal is to use a modified Dijkstra's algorithm to calculate the shortest time to reach the destination instead of the shortest path. The simple explanation for choosing this is that in today's fast paced world, the constraint of time is more pressing as compared to distance. People want results faster. Time is money. With everyone scrambling to squeeze in a few extra seconds, they want to save time while travelling from one place to another. The fastest path is however determined relative to the others on the road. If there is a shortcut to a certain place, then one can hope to reach that place faster by using that shortcut. However if a large number of drivers choose the same path, that path becomes a bottleneck to reaching the destination. Another constraint is the condition of the roads and the speed limits on the roads. If there is a small dirt path and a concrete path, it is an obvious conclusion that the speed will be faster on the concrete roads other than the dirt path. Where automated cars are concerned, it is needed to take the surroundings like speed bumps, construction work, etc. into consideration [1, 2]. These factors play a pivotal role while choosing a suitable path that will help reach the destination in the shortest time possible [3].

M. Srivastava · S. Saumya · M. Raja (✉) · M. Natarajan
Vellore Institute of Technology, Chennai, India
e-mail: mashrin.msrivastava2014@vit.ac.in; suvarna.saumyacjyoti2014@vit.ac.in; maheswari.r@vit.ac.in; mohana.n@vit.ac.in

S. Paul et al. (eds.), *Frontiers of Data and Knowledge Management for Convergence of ICT, Healthcare, and Telecommunication Services*,
EAI/Springer Innovations in Communication and Computing,
https://doi.org/10.1007/978-3-030-77558-2_3

3.1.1 Background and Motivation

The latest advancement in technological innovation raising to a deployment of smart city targeting imminent in smart vehicle mobility with Artificial Intelligence Centric, Machine Learning Centric, IoT enabled automation supporting Industry 4.0 as well. All ways of remote automation is achieved through IoT which ensures exceptional connectivity planes amongst the users and vehicles. The existence of flexibility in restructuring the merchandise logistics and transport optimization using various digital technologies such as remote/real automation, connected car techniques, big data analytics technologies and Industry 4.0 forms the strength of tomorrow's transportation and logistics market productivity assuring smart transport system [4]. The diverse cohorts of Connected and Automated vehicles (CAV) with numerous echelons of system automation along with human-driven vehicles (HDV) make us to extend and achieve extra resourceful and safer transportation systems [5]. Hosting of autonomous buses empowers optimization with reduction of fleets and cost acquired by the vehicle. The prototypical construction method anticipated castoff to deliver bus transit with effective guidance counting autonomous buses on its integrational bus fleets configuration and distribution. Henceforward, this work provides the outline of the research practice employed for an automated based Society of Automotive Engineers level five mode of transport expending a modified Dijkstra's algorithm integrating ant colony optimization, big data analytics and cloud computing. This integration helps the driver to reach the destination based on the traffic pertaining to the shortest time possible instead of the usual shortest distance algorithms.

3.2 Literature Survey

3.2.1 Automated Transport

Experiments in the field of automated cars started in the 1920s gained momentum in 1950s and is still improving with projects like Navlab from Carnegie Mellon University, Vislab from University of Parma, Eureka Prometheus Project from Bundeswehr University Munich, etc. Automobile companies like Tesla motors, Mercedes-Benz, Renault, Toyota, Bosch, Nissan, etc., are amongst the leaders in prominent projects undertaken [6, 7]. The most famous project is Google Self-Driving car undertaken by Google X.

3.2.2 Classification of Autonomous Cars

The standardization body for automotive-SAE (Society of Automotive Engineers) has a classification system consisting of six different levels based on the involvement of the driver based on his attentiveness and intervention that is required to reach the destination safely more than the capabilities of the vehicles [8].

Level 0: The automated system may alert the driver but it does not control the vehicle.

Level 1: Features such as ACC (Adaptive cruise control), Parking assistance with automated steering and LKA (Lane Keeping Assistance) in some combination requires the driver to be in a state ready to take control anytime.

Level 2: The driver must take control of detecting objects and events if the automated system fails to respond properly. When the driver takes over, all activities like accelerating, braking and steering must be suspended by the autonomous system and control must be promptly passed to the driver.

Level 3: The driver can afford to be inattentive towards the driving tasks only in known limited environments like freeways.

Level 4: The driver needs to control the vehicle in severe conditions like bad weather apart from which the automated system can take control of the vehicle without requiring the driver's attention.

Level 5: The driver is only required to start the system and set the destination, and the automated system can take control of driving in any location provided it is legal to drive.

3.2.3 Ant Colony Optimization

Ant colony Optimization algorithms is a part of swarm intelligent methods that use probabilistic techniques to solve problems. The basic principle comes from the food gather techniques of an ant colony [9]. The initial movement of ants is random; however, they leave a trail of pheromones on their way to the colony. If any of the other ants moving at random happen to see this food trail, they follow it and in turn leave pheromones increasing the strength of the pheromone trail which makes it more likely for the ants moving randomly to be attracted to it. With time however the pheromone trails evaporate and hence the longer the path from the food to the colony is, the more the chances of the ants not following the path due to disappearance of the pheromone trail, but in comparison, the shorter paths have more concentrated pheromones thus having a much higher probability of ants following that path and reinforcing the pheromones till finally all ants are following one path. The different variations of the ant colony optimization include the 'Min–max ant system', 'Elitist ant system', 'Rank-based ant system', etc. The applications include vehicle routing, scheduling, assignments, set, image processing, protein folding, data mining, etc.

3.2.4 Dijkstra's Algorithm

For a given weighted graph input, Dijkstra's algorithm can be used to find the shortest path between vertices in the graph when the condition that all weights are non-negative is satisfied. Originally the Dijkstra's algorithm found the shortest path between two vertices, but a more popular variation that is widely used is for finding the shortest path from one vertex to all other vertices in the graph [10]. Dijkstra's algorithm has many real-world applications that include finding the shortest path from one city to another using the driving distance as the weights in the graph and the nodes as the different cities. One major pitfall is the exclusion of the concept of quantity and time i.e. if there are many cars and the corresponding effects of the time taken to reach the destination [11, 12]. Using the concepts of Ant Colony Optimization (ACO) and Dijkstra's algorithm, a modified version can be tailored to suit this need [13].

3.2.5 GPS (Global Positioning System)

Irrespective of the weather conditions, be it rain, hail or snow if there are four satellites that have access to direct view of the GPS receiver, the current time and location are provided to the person by the navigation system based in space. The GPS system was created by the USA in addition to which various systems were in use and development including Global Navigation Satellite System (GLONASS) from Russia, BeiDou Navigation Satellite System from China, Quasi-Zenith Satellite System from Japan and Indian Regional Navigation Satellite System from India [14].

3.2.6 LIDAR (Light Detection and Ranging)

A laser is popularly used in fields like geology, geomatics, geomorphology, etc., to illuminate the target in order to measure distance [15].

3.2.7 Odometry

It involves the measurement of the current position relative to the position in which it started with the help of motion sensors [16]. The sensitivity to errors is high because the measurement of position involves the integration of velocity over time. Object recognition and event recognition is most needed in the implementation of the proposal.

3.2.8 Computer Vision

For a given weighted graph input, Dijkstra's algorithm can be used to find the shortest path between vertices in the graph when the condition that all weights are non-negative is satisfied. Originally the Dijkstra's algorithm found the shortest path between two vertices but a more popular variation that is widely used is for finding the shortest path from one vertex to all other vertices in the graph [10]. Dijkstra's algorithm has many real-world applications that include finding the shortest path from one city to another using the driving distance as the weights in the graph and the nodes as the different cities. One major pitfall is the exclusion of the concept of quantity and time i.e. if there are many cars and the corresponding effects of the time taken to reach the destination [11, 12]. Using the concepts of Ant Colony Optimization (ACO) and Dijkstra's algorithm, a modified version can be tailored to suit this need [13].

3.2.9 Ad Hoc Network

Ad hoc network is a type of wireless network that is decentralized, and it is free from the constraint of existing infrastructure because it relies on each node in the network to behave like a router and forwards data to the other nodes thus dynamically deciding the next node based on the available connectivity [17]. Mobile ad hoc networks (MANET) comprise mobile devices that follow the principles of ad hoc networks [18]. Vehicular ad hoc networks (VANETS) comprise the application of MANET to the domain of vehicles where the mobile devices are the vehicles themselves making them the main component of intelligent transport systems (ITS) with uses in electronic brake lights, traffic information systems and platooning.

3.2.10 Big Data

Big data refers to copious amounts of real-time data that has the characteristics of a large amount of data that can even extend to petabytes of data (volume), different types of data including audio, text, images, videos, etc., (variety), varying speed at which the data is collected (velocity) and the difference in the quality of data captured (veracity) making it too large and complex for traditional data processing applications to suffice. Application of big data to decision making gives better accuracy in results and helps make more informed decisions for more efficiency and reduced risks and cost [19]. Big data analytics involves combing through the vast amount of data to find unknown correlations and hidden patterns to provide information that could help in the decision making process. In this project the intention is to harness this power of big data analytics in an endeavour to make the

vehicle secure by using it to find the shortest path, to detect any obstacles through vision computing and image analysis and to provide better response dynamically to the vast amounts of real-time data generated so that decisions can be made spontaneously [20]. Predictive models can be used to determine the times and places for maximum traffic enabling the system to cope with the problems brought with it.

3.2.11 Cloud Computing

Cloud computing involves the sharing of computer resources and other services like applications, virtual machines, etc., on demand to a variety of devices. The advantage of cloud computing includes cost saving in terms of hardware equipment, agility, increased security due to decentralized data, independence of location and device, reliability, scalability, etc. The main advantage includes the provision of resources rapidly and with minimal amount of managerial effort [21]. The ability to provide prompt services in terms of expansions and the huge amount of computational power that is available due to sharing of resources makes cloud computing an essential part of the project.

3.3 Motivation

The motivation behind this started with the limitations of the autonomous cars as well as the limitations for Dijkstra's algorithm. For the autonomous cars, testing in heavy snow and rain has not been carried out due to safety concerns. Even with the use of LIDAR, it becomes difficult to distinguish light debris and garbage causing the car to veer unnaturally without any real threat and inability to distinguish police officers signalling cars to stop. The first known incident that proved fatal involved Tesla model S, an electric car from Tesla motors when an 18 wheel trailer tractor made a left turn just in front of the Tesla car and the car failed to stop on the non-controlled access highway [22–24]. These specified limitations can be overcome by incorporating appropriate weather forecasting and weather mapping techniques. The adverse weather conditions can be detected and necessary precautions like slowing down the speed in heavy rains to prevent skidding can be taken forward. The accidents involving Google's and Tesla's car were due to human error. This occasion does not arise when all cars are being controlled [25, 26]. For the Dijkstra's algorithm, the major disadvantage is that the algorithm does a blind search thereby consuming a lot of time and the wastage of necessary resources. Another disadvantage is that it cannot handle negative edges, which leads to acyclic graphs and most often cannot obtain the right shortest path. But the biggest problem with the algorithm is that though it is widely used, we believe that it is not an algorithm that should be used for path navigation. The traditional navigation applications made use of this algorithm initially given that it is well known and works well for finding

the shortest path. Since in the past, traffic was not a major issue because of lesser number of vehicles on the road, this algorithm performed fairly well. But this is not the case now. There are a lot of vehicles on the roads now, leading to heavy traffic. We have all observed in day to day life, that the shortest path now is no longer the path which will take the minimum time to reach somewhere. This was the motivation for proposing a modification to the Dijkstra's algorithm so that the heuristic takes care of this fact and suggests the path which takes the least time rather than the least distance as the value for time is becoming more and more important in the lives of people given the fast moving world and our lifestyles.

3.4 Problem Statement

Given a weighted graph of the road network, the objective is to find the path that takes the least amount of time to reach from any given location A to location B, where each location is represented as the vertex in the graph.

3.5 Proposed Work Description

The complete path flow of the proposed work is shown in Fig. 3.1. The major components of flow diagram are start, choose destination, shortest travel time path set, automated transport, destination arrive and stop.

- *Start*: The user is logged into the system as soon as the car is turned on. Choose destination: The source and destination is chosen based on the current position as identified by the GPS system and the input by the user, respectively.
- *Shortest travel time path set*: Using the proposed modified algorithm, based on all the users currently logged onto the system and their current position and destination, the route that takes the shortest time to travel is set.
- *Automated transport*: Using the grid based system where the inaccessible roads are blocked out by giving a very high value and the weight is taken in accordance to the speed the car can achieve on the particular road, the user is transported from source to destination.
- *Arrive at destination*: With minimal sensors, this automated mode transport uses big data analytics and cloud computing to ferry people from one place to another safely.
- *Stop*: Once the car is stopped, the user is logged off the system.

The first part of the prototype consists of the modified Dijkstra's algorithm in which the shortest path is determined based on the time taken and not on the distance between the source and destination. Analysing is required to find out the number of people who had own car and statistically how many people would want to go to the same place at the same time in a particular place. If the shortest route is to take the I-

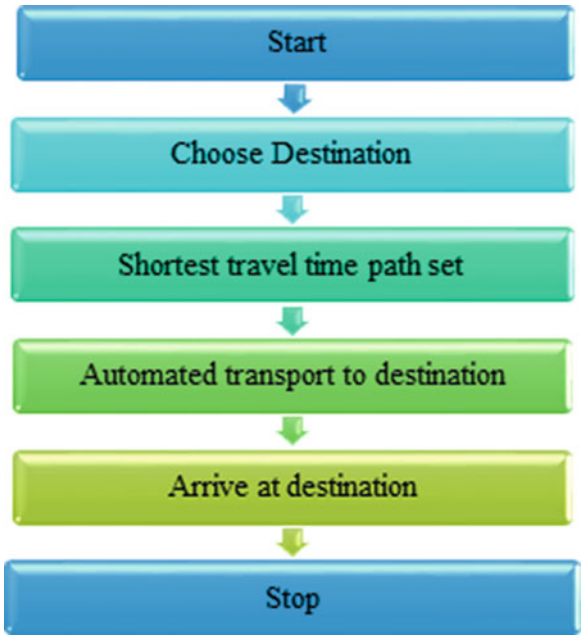


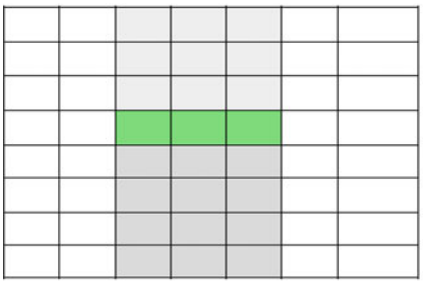
Fig. 3.1 Flowchart of the proposed work

16 and turn north towards the bridge, and all 100 people take the I-16 and turn north, then imagine the traffic jam a vehicle would be stuck in with the agonizing honking and bumper to bumper traffic. The whole ground can be represented as a grid or a matrix. A path can then be plotted based on the parts of the grid that are free. For example, if there is a pothole then that part of the grid can be made as inaccessible and will not show as a viable route that can be taken. Similar pothole checking and path plotting is made to all the places that are on either side of the road. In Fig. 3.2a and b the grey area represents a concrete road on the highway, therefore the weight assigned is minimal. The yellow area represents a tar road near residential areas, therefore the weight assigned is 10. The black areas are the areas that should not be accessed. We can either give a very large weight representing infinity.

With the offset of the GPS system and the popular google maps, the dependence on maps or memorizing routes has decreased leading to the ease in travel due to assurance of arrival at the mentioned destination. The project uses this concept to ensure accuracy in reaching the destination without needing to rely on maps or roads. A number of safety features will be provided including the application of vision computing to pre-detect objects and potentially dangerous situations.

Steps in the Algorithm for the Modified Dijkstra

- Start.
- Initialize the map.
- The values according to the speed in the particular area are set. The inaccessible zones and areas of roadwork are initialized with infinity value so that the areas are never chosen.



(A)

1000	1000	10	10	10	1000	1000
1000	1000	10	10	10	1000	1000
1000	1000	10	10	10	1000	1000
1000	1000	15	15	15	1000	1000
1000	1000	2	2	2	1000	1000
1000	1000	2	2	2	1000	1000
1000	1000	2	2	2	1000	1000

(B)

Fig. 3.2 Sample maps. (**a**) Sample map (blank parts). (**b**) Sample map with weights

- For the first driver is assigned the shortest path available to it and the route congestion for that path is incremented.
- For all the subsequent drivers, the most optimal path according to the modified shortest path algorithm is calculated considering the present congestion of the paths
- Optimal path is calculated by increasing the initial weight of the path by some factor of congestion, Weight = Weight + k*congestion.
- The destination is reached by providing the most optimal path using the modified algorithm.
- The data is updated dynamically to include the changes in traffic due to unforeseeable factors.
- Stop.

The time saved by each driver when the modified algorithm is used to calculate the shortest path to the destination instead of Dijkstra's algorithm is calculated for map of sizes 4×4, 6×6, 8×8 and 10×10. Figure 3.3 shows the values plotted and the corresponding curve fitting for the plotted value. The slope of the curve increases with increase in the size of the matrix showing the increase in the time saved as the size of the map increases. In a real-world scenario, the size of the matrix would be enormous as the roads and surrounding would be divided into multiple grids decreasing the size of the grids for more accuracy.

3.6 Implementation of Prototype

The model of the proposal was made using Arduino and the functioning was tested along with the application of the modified Dijkstra's algorithm. The path to be followed is traced using a vector, and the model robot follows the path that is given. In the proposal, the same will happen with the help of Google maps API. Using GPS the starting point of the user can be located and when the user enters the destination, the vectors to reach the destination is fetched by the Google maps API in accordance

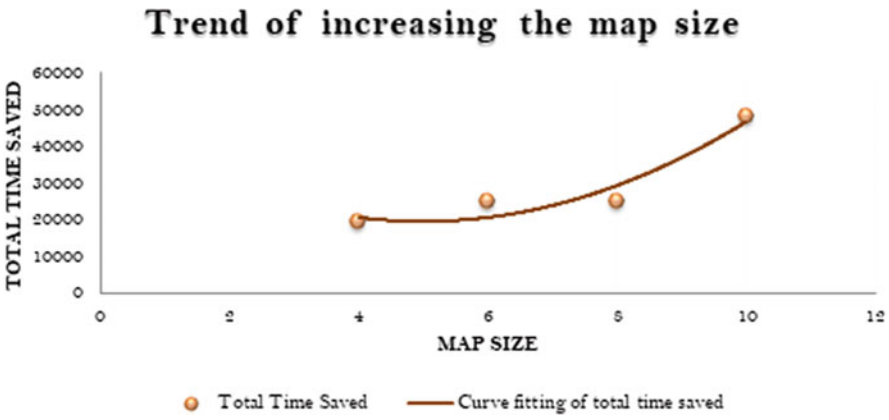


Fig. 3.3 Graph showing the total time saved as the map size increases

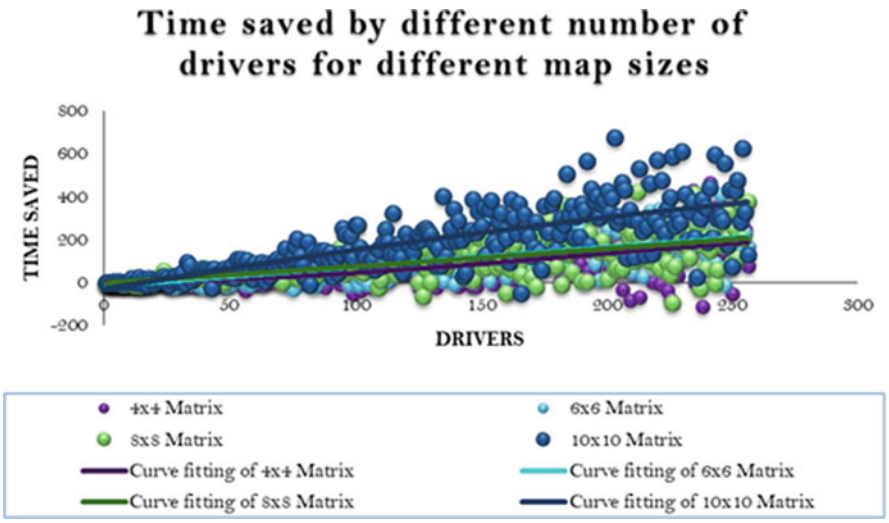


Fig. 3.4 Graph plot for the time saved by each driver

to the modified algorithm. The model moves along the plotted path as shown in Fig. 3.4. The scale is specified as the actual movement with respect to the length of the plotted vector. The model then moves along the given path by calculating the length by multiplying the plotted length with the factor of the scale. The same principle is applied using the Google API. Figure 3.4 also shows the actual length moved by the model. Figure 3.5 shows the movement of the model in progress. In Figure 3.6 are the screenshots that show the modified algorithm for the matrix of size 5×5 and the paths allotted for the different drivers having different source and destination.

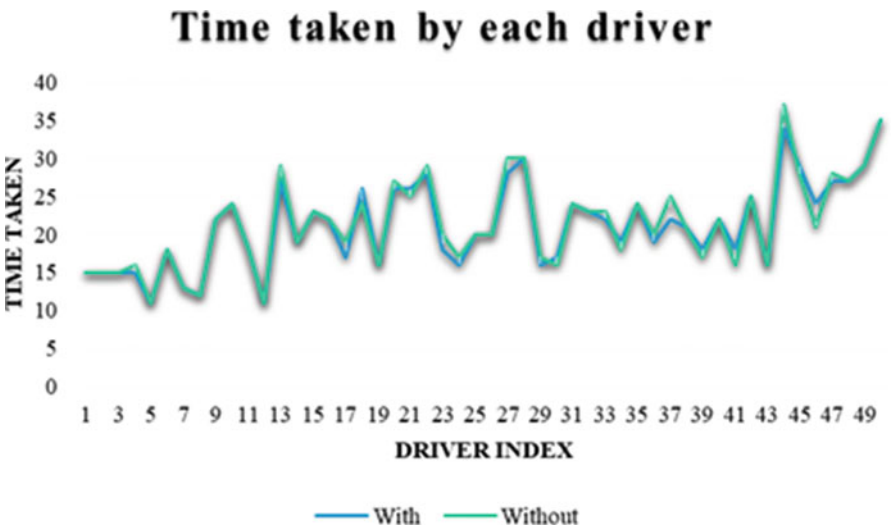


Fig. 3.5 Time saved in a particular case

Fig. 3.6 Vector plot and movement of the model

3.6.1 Mathematical Model

Let d_i, t_i and V_i be the distance, time and average velocity for the path 'i' given in Eqs. (3.1) and (3.2).

$$\text{Average Velocity } V_i = d_i / t_i \tag{3.1}$$

$$\text{Optimal Route } = Max\{V_1, V_2, , V_i\} \tag{3.2}$$

The trial runs performed for a number of matrices of varying sizes and the time saved by each of the driver are plotted. Figures 3.4 and 3.5 represent the time saved by each driver and time saved in a particular case. As the map size increases, we

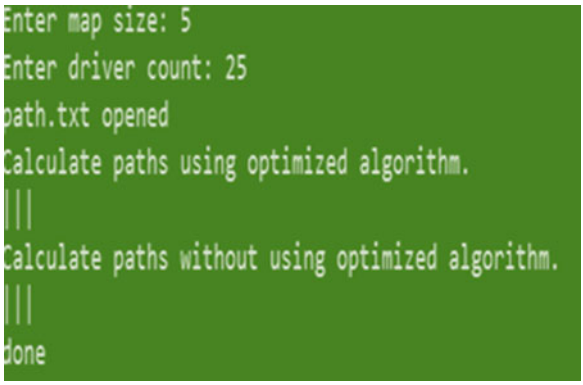


Fig. 3.7 Screenshot of application of the algorithm

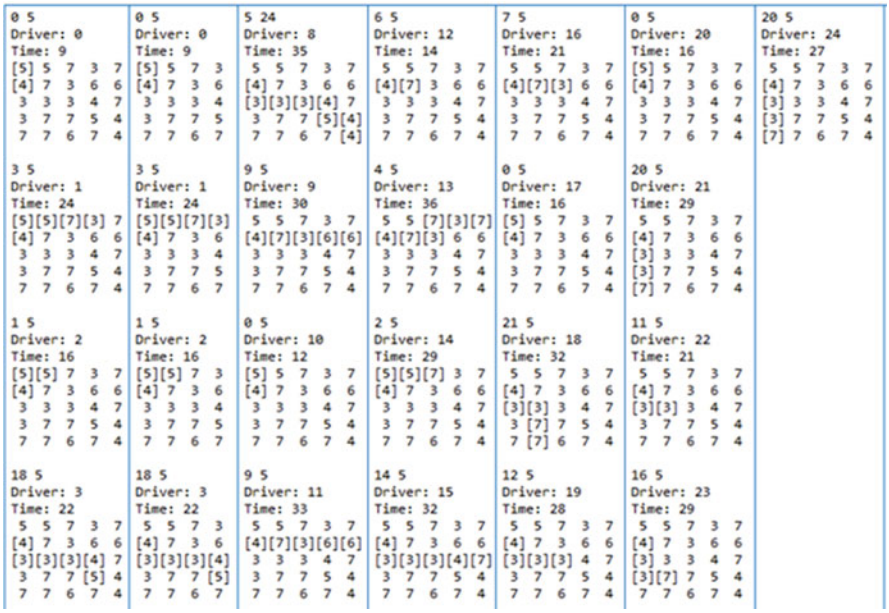


Fig. 3.8 The path selection using the modified algorithm

can observe that the time saved for each driver is increased. This Fig. 3.6 shows that with the increase in map size that the real-world scenario will have, the time saved will increase thus proving the efficiency of the modified algorithm and ensuring high performing automated cars.

3.6.2 Collision Avoidance

The most important issue to be addressed is that of collision avoidance. There are two types of collision—static and dynamic. Although in the proposal there is no possibility of collision but taking into account the errors and malfunctioning that could happen, the possibility of errors and the corresponding methods for correction have to be taken into consideration. For static collisions in case of roadwork or inaccessible paths that can be predefined before the vehicle reaches the area, since the map is updated dynamically, they can be assigned a very high value that forbids the movement to those areas. For the dynamic collisions or the collisions that may occur between the vehicles on the road, we can use vision computing methods for analysis of the movement of the vehicles in the vicinity. The results then can be communicated from vehicle to vehicle using the principles of VANET. In case of manual rerouting of traffic, a specialized VANET signal can be used to communicate the message to the vehicles thus successfully rerouting traffic in case of emergencies and even in the case of ambulance travels. The future work would be to build a prototype of the model and go over the safety features in the case of a failure with a fine comb.

A prototype of the proposed work is designed and implemented using Arduino and appropriate hardware components.

Performance analysis has been made by constructing the code to calculate the paths using optimized algorithm and paths without using optimized algorithm with map size of 5 and driver count of 25. The obtained result is depicted in Fig. 3.7 as screenshot of application of the algorithm.

The traces of varying values of the car driving with respect to number of drivers and time stamp is simulated and tracked appropriately. Figure 3.8 shows the output of the path selection using the modified Dijkstra algorithm.

3.7 Conclusion and Future Work

Thus, a prototype is proposed and analysed for the mode of transport using a modified Dijkstra's algorithm. Deriving the concepts of Ant Colony Optimization, big data analytics and cloud computing, a novel technique to arrive at the destination based on the traffic and the number of drivers on the road, in the shortest time possible instead of the usual shortest distance algorithms is simulated. Appropriate analysis with respect to time frame is performed and presented here to validate the state of the art. The graph and screenshots remain the proof of concept supporting the correctness of modified Dijkstra algorithm for shortest time. The future work would be to rebuild a prototype of the model and go over the safety features in the case of a failure with a fine comb.

References

1. F. Nashashib, L. Bouraoui, A cooperative personal automated transport system *"A CityMobil Demonstration in Rocquencourt"*, in *2012 12th International Conference on Control, Automation, Robotics and Vision Guang zhou, China, (ICARCV 2012)* (2012), pp. 644–649
2. L. Bouraoui, F. Charlot, C. Holguin, F. Nashashibi, M. Parent, P. Resende, An on-demand personal automated transport system: the CityMobil demonstration in La Rochelle, in *Intelligent Vehicles Symposium (IV), June 2011* (2011), pp. 1086–1091, pp. 5–9
3. M. Parent, Cybercars: A solution for urban transport, in *CODATU Conference* (2004)
4. S. Hamdar, A. Talebpour, R. Bertini, Traffic and granular flow: the role of data and technology in the understanding of particle dynamics. J. Intell. Transp. Syst. Technol. Plann. Oper. **24**(6), 535–538 (2020)
5. A. Nikitas, K. Michalakopoulou, E. Tchouamou Njoya, D. Karampatzakis, Artificial intelligence, transport and the smart city: definitions and dimensions of a new mobility era. Sustainability **12**, 1–19 (2020)
6. J. Choi, Y. Khaled, M. Tsukada, T. Ernst, IPv6 support for VANET with geographical routing, in *8th International Conference on Intelligent Transport System Telecommunications (ITST 2008), Phuket* (2008)
7. J. Xie, J. Xie, F. Nashashibi, M. Parent, O. Favrot, A real-time robust global localization for autonomous mobile robots in large environments, in *ICARCV 2010* (2010), pp. 1397–1402
8. T. Meyerowitz, C. Pinello, A. Sangiovanni-Vincentelli, A tool for describing and evaluating hierarchical real-time bus scheduling policies, ACM, Anaheim, California, USA, in *Proceedings of the 40th Annual Design Automation Conference* (2003), pp. 312–317
9. L. Zuo, L. Shu, C. Zhu, T. Hara, A multi-objective optimization scheduling method based on the ant colony algorithm in cloud computing, in *IEEE Special Section on Big Data Services And Computational Intelligence for Industrial Systems*, vol. 3 (2015), pp. 2687–2699
10. X. Cheng, L. Fang, L. Yang, S. Cui, Mobile big data: the fuel for data-driven wireless. IEEE Internet Things J. **4**(5), 1489–1516 (2017)
11. M.A. Alsheikh, D. Niyato, S. Lin, H.-P. Tan, Z. Han, Mobile big data analytics using deep learning and apache spark. IEEE Netw. **30**(3), 22–29 (2016)
12. J.L. Toole, C. Herrera-Yaqe, C.M. Schneider, M.C. Gonzlez, Coupling human mobility and social ties. J. R. Soc. Interface **12**(105), 1–9 (2015)
13. H. Dong, Traffic zone division based on big data from mobile phone base stations. Transp. Res. C Emerg. Technol. **58**, 278–291 (2015)
14. S. Sheeba Rani Gnanamalar, R. Maheswari, B. Sharmila, V. Gomathy, IoT driven vehicle license plate extraction approach. Int. J. Eng. Technol. (UAE) **7**(2), 457–459 (2018)
15. S. Massobrio, A. Nesmachnow, A. Tchernykh, A. Avetisyan, G. Radchenko, Towards a cloud computing paradigm for big data analysis in smart cities. Program. Comput. Softw. **44**(3), 181–189 (2018)
16. S. Oh, Y.J. Byon, H. Yeo, Improvement of search strategy with K-nearest neighbours approach for traffic state prediction. IEEE Trans. Intell. Transp. Syst. **17**(4), 1146–1156 (2016)
17. M. Peng, K. Zhang, J. Jiang, J. Wang, W. Wang, Energy-efficient resource assignment and power allocation in heterogeneous cloud radio access networks. IEEE Trans. Veh. Technol. **64**(11), 5275–5287 (2015)
18. B. Bangerter, S. Talwar, R. Arefi, K. Stewart, Networks and devices for the 5G era. IEEE Commun. Mag. **52**(2), 90–96 (2014)
19. Q. Shi, M. Abdel-Aty, Big data applications in real-time traffic operation and safety monitoring and improvement on urban expressways. Transp. Res. C Emerg. Technol. **58**, 380–394 (2015)
20. H. Song, H. Liang, H. Li et al., Vision-based vehicle detection and counting system using deep learning in highway scenes. Eur. Transp. Res. Rev. **11**, 51–55 (2019)
21. C. Zhu, X. Li, V.C.M. Leung, X. Hu, L.T. Yang, Job scheduling for cloud computing integrated with wireless sensor network, in *Proc. IEEE 6th Int. Conf. Cloud Computing. Technol. Sci. (CloudCom)* (2014), pp. 62–69

22. F. Farahnakian et al., Using ant colony system to consolidate VMs for green cloud computing. IEEE Trans. Services Comput. **8**(2), 187–198 (2015)
23. B. Zhang, X. Wan, J. Luo, X. Shen, A nearly optimal packet scheduling algorithm for input queued switches with deadline guarantees. IEEE Trans. Comput. **64**(6), 1548–1563 (2015)
24. Z. Tang, L. Jiang, J. Zhou, K. Li, K. Li, A self-adaptive scheduling algorithm for reduce start time. Future Generat. Comput. Syst. **43–44**(3), 51–60 (2015)
25. Y. Chen, A. Zhang, Z. Tan, Complexity and approximation of single machine scheduling with an operator non-availability period to minimize total completion time. Inf. Sci. **25**(1), 150–163 (2015)
26. Y. Zha, J. Yang, Task scheduling in cloud computing based on improved ant colony optimization. Comput. Eng. Des. **34**(5), 1716–1719 (2013)