An Investigation for Enhancing the Shortest Path Problem in Real Time Information

[1]Greeshma Srivastava, [2]Ankit

[1]M. Tech. Scholar, [2]Assistant Professor
Electronics & Communication Engineering
CBS Group of Institutions

Abstract*: It is necessary to provide a dependable, realistic, and scalable network for Adhoc communication between cars in order to enable intelligent traffic communication. This paper takes everything into consideration while creating our modified model for the quickest route. In this case, our approach offers the user more than one route/path to take in order to reach the goal. The purpose of a position tracking system is to identify the shortest route possible in order to save time. whereas the earlier model only considered the shortest distance and did not consider the time factor. Earlier models only considered the shortest distance and did not consider the time factor. In this case, artificial intelligence is being used to monitor the location of the item and then determine the shortest route between the source and the destination. The shortest path is the most direct route from point A to point B, and it will make use of artificial intelligence to aid in decision-making power. In order This addition provides us with a This improvement gets the realities of world. In this way, the real-world implications of this model provide us with additional real-world parameters and a real-time situation for route traveling. Travelers will save time as a result of this. In addition, we put the findings of the experiments into practice by communicating with moving cars through wireless sensor networks (WSNs) radios of various types and configurations. Our goal is to offer measurable background knowledge on various types of automotive networking methods in order to identify the most effective solution for a variety of distinct use scenarios. The suggested method, which makes use of real-time information, provides us with a more dynamic model for determining the quickest route. Its ability to offer real-time information makes it the ideal model for the future, and its improvement gets us closer to real-time circumstances while saving passengers' time.

***Keywords*: A * Algorithm, Shortest Path, Real Time Information**

I. INTRODUCTION

There are many various types of traffic obstacles, including the number of vehicles on the road, the number of pedestrians on the road, weather conditions, and road conditions (width, pavement condition, gradient and visibility). preference, and among others, influence the speed at which one travels when driving. According to Zhao (2010), while there are methods in place for calculating driving speed, difficult evaluate automobiles at same. As a result, without individual measurement, it is impossible to determine the time cost of travel by motor vehicles. Drivers are required to increase their pace while passing by hospitals, schools, and pedestrian crossings when traveling on the road, and motor vehicles are required to come to a complete stop when the traffic signals turn red. In this context, the authors examine permanent buildings close to the roadway that act as speed sapping barriers for divers when driving (such as hospitals, schools, residential areas or traffic signals). as well as traveled by route, allowing alternative to be selected from among them. Providing drivers with the shortest time plan possible in order for them to run their motor vehicles as effectively as possible is also a remedy for traffic congestion. The shortest route design also lowers the amount of time that motor vehicles are on the road, thus reducing the amount of time that they emit greenhouse gases. A route plan that takes the least amount of time is required for unfamiliar with who without developing any specific habits in the lowest amount of time feasible. In principle, provided maintains route take least amount of reach its destination. However, it is difficult to keep vehicles moving at a constant pace in a congested region since the speed is influenced range variables.

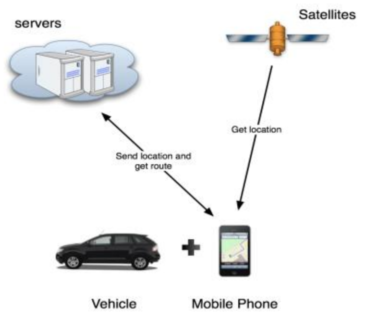


Fig. 1: Mobile-based navigation web application

User's location data must be sent to the appropriate database via the web application, and once the data has gone through the various processing steps, a traffic analysis must be performed on the basis of the data. A route plan that takes the shortest amount of time feasible is created by combining track longitude with the current traffic condition, while eliminating factors that affect driving speed is avoided.

According to our findings, the following are the most significant factors influencing the. The weights of different variables are determined by a variety of factors. Additionally, in addition to the previously stated factors, we create a user-controlled factor that is located on road areas that are congested, have accidents, or have temporary structures. This element is at the control of the users. Users may input the location of such a place so that while still providing other users with the best possible route plan to get there. By concluding our investigation, we have also shown that anybody may do GIS analysis using open-source software. Path finding, also known as patching, is the process of mapping the shortest path between two places with the use of a computer software. When it comes to labyrinths, this is a more realistic option. The majority of the research in this field is focused on the Dijkstra technique for determining the shortest path in a weighted network. A method for finding routes has at its core the process of looking for a diagram by starting from a evaluating goal. While diagram in breadth may uncover is given, ways that diagram are more likely to get at the. As, consider assessing path, typically continue toward objective and only deviate off the road in order to avoid any obstructions and to reduce deviations.

The following are the two most significant challenges in locating a route:

(1) Identifying a network;

(2) of the quickest way

First and first-intensive searches, which deal with the first problem by eliminating all alternatives, repeat all potential routes starting from the provided node target execute denotes.

harder it to figure out appropriate approach. According to the three is temporal as a consequence of using this method. However, it is not necessary to investigate all possible paths in order to determine which is the best. Through the use of heuristics or dynamic programming, methods listed generic available that network the need for preparation. However, in real-world trip routing systems, algorithms that can pre-process the graph to gain higher speed may be able to achieve even better time complexities than previously thought possible. The hierarchy of contraction is an example of such an algorithm.

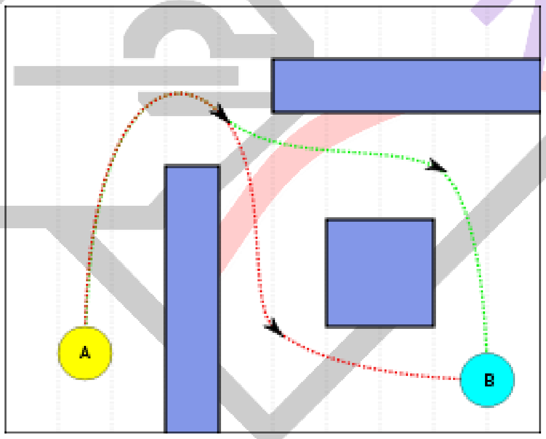


Fig. 2: Equivalent paths between A and B in a 2D environment

II. Dijkstra's Algorithm

Dijkstra's technique is a good example of a graphical route-finding algorithm in its simplicity and effectiveness. Starting beginning collection this technique proceeds to the next step. shortest beginning checked to ensure that it is still there. The node is designated as "closed," and if it has not been previously examined, all of the nodes immediately next of nodes. This process is repeated. n Because with shortest distance between them are examined first, the route to them is the shortest path when the destination is first recognized.

When Dijkstra algorithm does not work properly. If an diagram route whereas the best route 1. The which starts with and proceeds to, will examine B first since it should the project declare it closed so expenses are evaluated again. As a result, Dijkstra is unable to there will never be a negative edge weight in many real applications, Dijkstra's technique is best suited for route discovery rather than route optimization.

A* Algorithm

A* is a modified version of the widely used game algorithm developed by Dijkstra. A* assigns a weight and a the that is roughly the same as the the. As calculated by the heuristic, this estimated distance serves as an indication of a possible that the terminal. As soon as you discover an initial path, you may eliminate longer paths, allowing you to save time. Whenever the distance between the start and the finish is more than x.

When compared to the Dijkstra algorithm, A* makes use of the system. At zero, A* corresponds to the algorithm developed by Dijkstra and his colleagues. When the heuristic estimate increases, routes, but it does so at a faster rate. If heuristic is simply actual distance between two nodes, A* examines the nodes with the shortest distance between them. While it is possible consistently estimates real this is not always practical.

Any corner of route planning algorithms searches for routes in a continuous configuration space breakdown, which is a continuous configuration space breakdown (such as a two-dimensional terrain). A regular grid with cells blocked and unblocked in alternate rows and columns. In order to find the shortest path from one beginning vertex to another, the appropriate visibility charts are examined. However, this process is often very slow due to the fact that the number of edges increases quadratically in proportion to the number of vertices. In most cases, the grid graph looks up inefficient routes (for example, because the changes in heading of the resulting path are restricted to multiples of 45 grades on an eight-neighbor grid network), but it is relatively quick because the number of edges increases only linearly with the number of vertices in the network. Optimizing the search path generally results in a shorter trip, but it has no effect on the topology of the route. For example, if the search method passes a blocked cell on the left and the shortest route passes the same blocked cell on the right, the shortest route is the shortest path. As a result, the search and optimization processes are interconnected. Methods of route planning at any angle disseminate information (to examine fast) across the grid edges without limiting their journey to the grid edges (to find short paths). As a result, the changes in the direction of their trajectories are not restricted to the specific angles that give rise to their name.

III. Shortest Path by A* Algorithm

To draw an efficient route between a large number of dots, or nodes, A* is a computer technique that is often used in pathfinding and graph traversal applications. In addition to being extensively utilized, it is well-known for real-world however, in order to get better performance before the graph is pre-processed [1] despite the fact that some research has proven A* to be superior to other approaches. The (now SRI International) (now SRI International). It is an expansion of the Edger Dijkstra algorithm, which was developed in 1959. A* enhances time performance via the use of heuristics.

A* looks for and finds the most efficient route from a given beginning node to a single destination node at the lowest feasible cost (out of one or more possible goals). In crossing the graph, A* takes the shortest projected total cost or distance, which allows the ordered priority queue for alternate path segments to remain intact.

It uses a more knowledge-intensive node x cost function (commonly referred to as f(x)) for selecting the order in which nodes in the tree are visited. In terms of cost functions, they are a combination of two functions:

G(x) is a path-cost function that is known from the starting node to the current node x (often abbreviated as g(x)).

• A future route cost function, which is a permitted "heuristic estimate" of the distance from x to the destination (usually referred to as h(x)).

To be eligible, the h(x) component of the f(x) function should be a valid heuristic, which means that the distance to the goal should not be exaggerated. For an application such as routing, h(x) may thus be the direct distance from the destination, because that is physically the shortest distance between two locations or nodes.

Mono tone or coherent refers to when a h heuristic satisfies the extra criteria of each edge (x, y) in a graph (where d indicates the length of the edge), and the h heuristic is used. As a result of this, A* may be performed more efficiently—in general, no node has to be processed more than once (see closure set below)—and A* corresponds to operation of the Dijksttra algorithm with a reduced cost of $d'(x,y):=d(x,y) + h(y) - h(x,y):=d(x,y) + h(y) (x)$. Initially, it searches for routes that seem to go to the goal, in the same way that other informed search algorithms do. This is different from a greedy, initial search in that it takes into account the distance already gone; the g(x) represents the cost of starting from the beginning point, rather than simply the local cost of the previously expanded node. A priority queue of nodes to be passed from the beginning node, called as the fringe or open set, is kept from the starting node. The lower the value of f(x), the higher the priority of the node x. Each step of the technique removes a node with the lowest f(x) value from the queue, changes the values of its surrounds' f and g in the proper manner, and then adds the node's surrounding neighbors to the queue, repeating the process. The procedure continues until the f value of a target node is lower than the f value of any other node in the queue (or until the queue is empty). It is possible that goal nodes will be passed on many times if other nodes have lower f values, since this will result in a shorter trip to the destination. An acceptable heuristic determines that the f value of the objective is the shortest path since the aim is zero in this case.

The technique that has been described so far just provides us with the quickest path. It is feasible to simply revisit the method in such a way that each route node follows the previous route node in order to find the true step sequence. Following the execution of this procedure, the ending node refers to its predecessor, and so on, until the starting node is a specific node predecessor and so on.

How Shortest Path Help for Navigation System.

Shortest It is inevitable that there will be road problems in applications using road networks, such as urban emergency management and driving management systems, in which the best route options must be established. Because traffic conditions in a city may change at any time and because huge numbers of requests are usually submitted at any one time, it is necessary to get a response as quickly as possible. As a result, the algorithm's efficacy is very critical. Some techniques make advantage of preprocessing to compute results before they are really required. If a new request is made, these results will be kept in memory and may be used right away. If the devices have limited internal storage and no external storage, this cannot be utilized.

AI for Shortest Path

In its most basic form, an algorithm is a method for solving a problem. When we look at a 2D grid with many items for us human beings to consider, we may rapidly decide the path the character should take to reach his or her destination without giving it much thought. These semi-subconscious processes should be converted into a list of activities that everyone (or a computer) may do every time in order to get the same result. When developing game AI, it is a common problem to determine the shortest route from one object to another. A variety of approaches are available to solve this problem. A* is perhaps the most well-known of the 2D grid / tile-based games, with Dijkstra being a very good second choice. Depending on the complexity of the game, the Dijkstra technique may be nearly as fast as the A* method, with a few modifications. A* is generally better, but it can be a bit more complicated to implement; thus, I will describe the Dijkstra algorithm's basis and discuss alternative algorithms in future articles, such as the A* algorithm. For others, it may not be immediately obvious how this translates into game development. In this section, I'll refer to graphs often, but you can simply convert it to 2D grid or tile-based maps as well.

IV. PROPOSED WORK

Finding the shortest distance is generally more complex and time-consuming than finding the simplest and most basic path. By combining fluid logic with artificial intelligence [8], we can identify the position and shortest path to an object captured by a camera or other sensing device. The first step is to photograph the target area, and the second step is to photograph the desired place where you want to approach it as fast and accurately as possible [6]. We can find the best route out of any reasonable technique by using artificial intelligence to search for it. When looking for the quickest route, it is not always necessary to consider the shortest route that will take the most time, the most traffic lights, and the poorest road condition that would cause the most delay [7]. As a result, we all take into account our change model in order to choose the shortest path. In this case, our method objective.

Real Time Information Model for Shortest Path

The real-time information model of the Grid algorithm is recommended as a solution for the shortest route problem.

- In this algorithm, we give real-time information to the shortest route technique.
- This covers a variety of factors such as traffic congestion, road damage, the absence of traffic signals, and rush hour, all of which have an effect on the shortest time period.

A* Algorithm for Real Time System

As a result, between S and nine kilometers. runtime is the is the. According to my estimation, in practice, the most effective method generates a utilizing as its basis of operation. A multi-dynamic grid search is carried out using this method. A large number of measurements are used to create the grid. wide possible subdivided succession with same value as one another. The multi-dimensional grid has a core that is responsible for locating the optimum location. A number of passes are required in the search. Every time the algorithm runs, it looks for a node (intersection point) that has the lowest function value possible. This node becomes the new center of the grid, and a smaller grid is formed around it. Eventually, the multidimensional grid is reduced to its optimal size via a series of passes.

Flow Chart

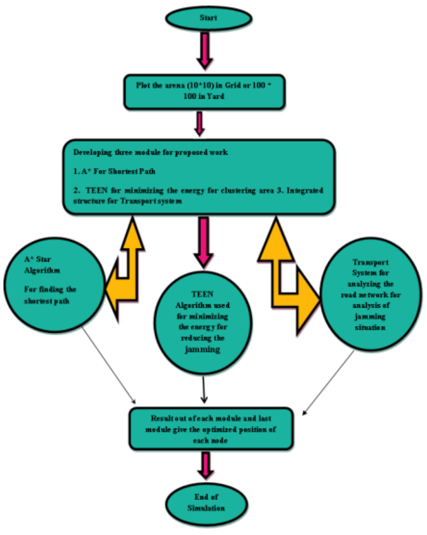
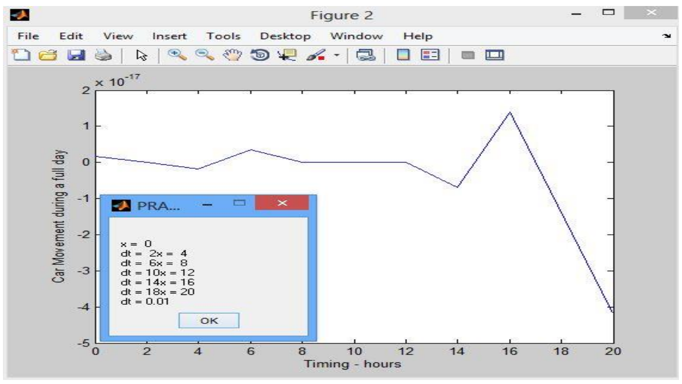

Start
Plot the arena (10*10) in Grid or 100 * 100 in Yard
Developing three module for proposed work
1. A* For Shortest Path
2. TEEN for minimizing the energy for clustering area 3. Integrated structure for Transport system
A* Star Algorithm
For finding the shortest path
TEEN Algorithm used for minimizing the energy for reducing the jamming
Transport System for analyzing the road network for analysis of jamming situation
Result out of each module and last module give the optimized position of each node
End of Simulation

Fig. 3: Flow Chart

Simulation and Result



```
x =  0
dt =  2x =  4
dt =  6x =  8
dt = 10x = 12
dt = 14x = 16
dt = 18x = 20
dt = 0.01
```

Fig. 4: A Different Value of Movement of car during the whole day

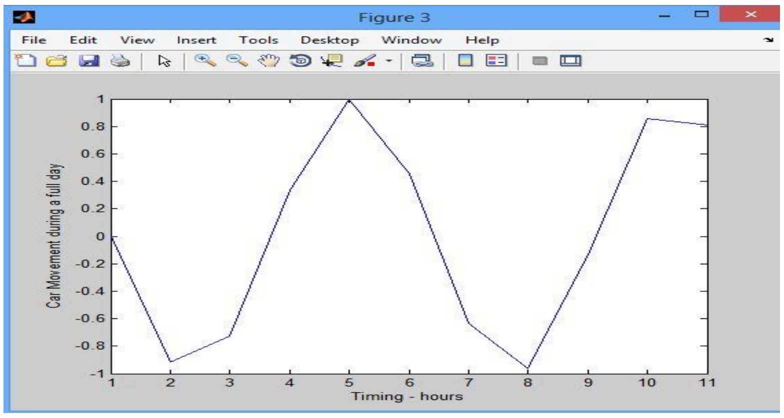
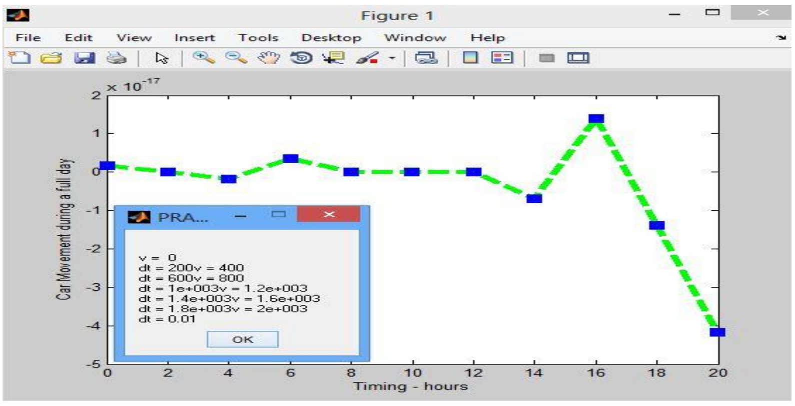


Fig. 5: Decision of CAR when it drastically changes its movement long run up



```
v = 0
dt = 200v = 400
dt = 600v = 800
dt = 1e+003v = 1.2e+003
dt = 1.4e+003v = 1.6e+003
dt = 1.8e+003v = 2e+003
dt = 0.01
```

Fig. 6: An operation point that has also come in pop up box that is speed time calculation of sensor node mounted on CAR

The picture above shows the MATLAB figure after it has been run through the simulation. Although it is obvious that the quickest path has led to numerical, the display is just a unit presentation. It has a wide range of graphical user interfaces on a large scale. Just the major part of the shortest route in real time, and the automated vehicle in the area is running on this piece of shortest route. it does it in a shorter amount of time than before. The ultimate goal, on the other hand, is to accomplish the main automation function in the shortest amount of time.

Table 1: Comparison of individuals with proposed integrated system

	A* Star Algorithm	**TEEN Algorithm**	**Integrated System**
Jamming Information	No	Yes	Yes
Shortest Path	Yes	No	Yes
Real Time Information	No	Yes	Yes
Calculation Speed	Very Fast	Slow	Very Fast
Platform	MATLAB	MATLAB	MATLAB

Concluded Result

- In a more dynamic model, the addition of real-time information enables us to find the quickest path by using this information.
- Because of this real information, it is the best future model.
- This enhancement brings us closer to the reality of the situation while also saving traveler's time.
- It may be applicable in the case of long-distance autonomous vehicles.

V. CONCLUSION & FUTURE WORK

According to our findings, all of the barriers are permanent constructions, and we have decreased traffic and driving frequency obstructions to accommodate for them. The approach has the potential to serve as a model for urban planning. In order to minimize the probability of traffic congestion, it is important to carefully plan adjacent roads (including schools, hospitals, lights, and residential areas). When doing similar research and studies, geoscience and informatics scholars and students may find our method to executing the function of our program useful as a model. Furthermore, our curriculum methods since it employs a variety of techniques and tools in the application, such as geographic information systems. The program functions as stated; however, there are certain problems that will need to be addressed in future study. It is necessary to update the technique shorten required. aim to be able to update the application project on a regular basis and improve the manner in which our research is made available to the public. It was necessary to carry out the testing in the simulated environment. Based on the tests shown in the findings, we can infer that the number of obstacle components, as well as the weight of each obstacle, will have an impact on the outcomes of the shortest route search. Despite the fact that the barriers in our analysis show how they affect motor vehicle movement and how we offer the shortest investigated study. It is true that the real shortest time route cannot be discovered; instead, we provide calculating based on a variety of variables. The researchers may use more precise barrier factors and weights to make the result more acceptable if they have access to sufficient geographic and statistical data.

REFERENCES

[1] Dudgeon, D.E. and R.M. Mersereau, Multidimensional Digital Signal Processing. 2014, Englewood Cliffs, New Jersey: Prentice-Hall.

[2] Castleman, K.R., Digital Image Processing. Second ed. 2009, Englewood Cliffs, New Jersey: Prentice-Hall.

[3] Oppenheim, A.V., A.S. Willsky, and I.T. Young, Systems and Signals. 2007, Englewood Cliffs, New Jersey: Prentice-Hall.

[4] Elias C. Eze, Sijing Zhang and Enjie Liu "Vehicular Ad Hoc Networks (VANETs): Current State, Challenges, Potentials and Way Forward" September 2014, Proceedings of the 20th International Conference on Automation & Computing, Cranfield University, Bedfordshire, UK, 12-13 September 2014 @ IEEE

[5] Bin Tian, Kun Mean Hou, Hongling Shi, Xing Liu, Yibo Chen, Jean-Pierre Chanet "Application of Modified RPL under VANET-WSN Communication Architecture" 2013 International Conference, Blaise Pascal University Clermont-Ferrand, France, 978-0-7695-5004-6/13 © 2013 IEEE & DOI 10.1109/ICCIS.2013.387.

[6] Rua Qin, Zi Li, Yanfei Wang, Xuejia Lu and Wen sheng Zhang & Guiling Wang "An Integrated Network of Roadside Sensors and Vehicles for Driving Safety: Concept, Design and Experiments" 2014 by u.s. Government work not protected by U.S. copyright @ IEEE

[7] Mohammad Jalil Piran , G. Rama Murthy , G. Praveen Babu , Ehsan Ahvar " Total GPS-free Localization Protocol for Vehicular Ad Hoc and Sensor Networks (VASNET) " 2011 Third International Conference on Computational Intelligence, Modelling & Simulation, 978-0-7695-4562-2/11 © 2011 IEEE & DOI 10.1109/CIMSim.2011.77.

[8] Ayonija Pathre Chetan Agrawal Anurag Jain P.G.scholar Department of CSE RITS – Bhopal, India "A Novel Defense Scheme against DDOS Attack in VANET" 978-1-4673-5999-3/13/$31.00 ©2013 IEEE.

[9] Varsha Raghuwanshi, Simmi Jain "Denial of Service Attack in VANET: A Survey" International Journal of Engineering Trends and Technology (IJETT) – Volume 28 Number 1 - October 2015 ISSN: 2231-5381 http://www.ijettjournal.org Page 15.

[10] Vinh Hoa LA, Ana Cavalli "Security Attacks and Solutions in Vehicular Ad Hoc Networks: A Survey" International Journal on AdHoc Networking Systems (IJANS) Vol. 4, No. 2, April 2014 DOI: 10.5121/ijans.2014.4201.

[11] Swapnil G. Deshpande "Classification of Security attack in Vehicular Adhoc network: A survey" Web Site: www.ijettcs.org Email: editor@ijettcs.org, editorijettcs@gmail.com Volume 2, Issue 2, March – April 2013 ISSN 2278-6856. Volume 2, Issue 2 March – April 2013 Page 371.

[12] Sharaf Malebary, Dr. Wenyuan Xu PhD student Department of Computer Science and Engineering University of South Carolina – USA "A Survey on Jamming in VANET" International Journal of Scientific Research and Innovative Technology Vol. 2 No. 1; January 2015 143.

[13] Ahmad Yusri Dak, Saadiah Yahya, and Murizah Kassim "A Literature Survey on Security Challenges in VANETs" International Journal of Computer Theory and Engineering, Vol. 4, No. 6, December 2012 Manuscript received August 1, 2012; revised September 2, 2012.

[14] Deepak Kushwaha Piyush Kumar Shukla, Ph.D Raju Baraskar PG Student Department of CSE UIT RGPV, Bhopal, India "A Survey on Sybil Attack in Vehicular Ad-hoc Network" International Journal of Computer Applications (0975 – 8887) Volume 98– No.15, July 2014 41.

[15] Divya Chadha, Reena "Vehicular Ad hoc Network (VANETs): A Review" ISSN(Online): 2320-9801 ISSN (Print): 2320-9798 International Journal of Innovative Research in Computer and Communication Engineering (An ISO 3297: 2007 Certified Organization) Vol. 3, Issue 3, March 2015.