DETERMINATION OF THE SHORTEST PATH OF A NIGERIAN UNIVERSITY MAP USING DIJKSTRA'S ALGORITHM

Conference Paper · November 2017

4 authors:

Usman Opeyemi Lateef
Tai Solarin University of Education

Adedeji Oladimeji Adebare. PhD.
Tai Solarin University of Education

Rufai Kazeem Idowu
Tai Solarin University of Education

Peter Adebayo Idowu
Obafemi Awolowo University

DETERMINATION OF THE SHORTEST PATH OF A NIGERIAN UNIVERSITY MAP USING DIJKSTRA'S ALGORITHM

Usman Opeyemi Lateef[1], Adebare Adedeji, O.[2], Rufai Kazeem, I.[3], and Idowu Peter, A.[4]

[1,2,3]Department of Computer Science, Tai Solarin University of Education, Ogun State, Nigeria.

[4]Department of Computer Science and Engineering, Obafemi Awolowo University, Ile-Ife, Nigeria.

Corresponding Author: Usman, O.L. (usmanol@tasued.edu.ng)

ABSTRACT

The need to optimize resources measured in term of time and cost incurred when traversing from one point to another necessitates the need for the implementation of an efficient algorithm capable of handling shortest path problem. This paper presents the development of an Android-based navigation software named: "TASUED MAP" which eases the movement of visitors, fresh students, and above all, improves the quality of life of the entire members of the university community. TASUED MAP, which was developed using JQuery-Mobile, JavaScript, and XML with Android Software Development Kit (SDK), explores the principle of Dijkstra's algorithm to find the shortest path in the proposed navigation software. Simulation of Dijkstra's algorithm was achieved using MATLAB 2010 software and the experiments were conducted in 10 iterations by varying either the source node or destination node in each case. Results from the simulation experiments showed that Dijkstra's algorithm is an efficient single-source shortest path algorithm and as such, sufficient as a building-block for the development of navigation software of this kind.

Keywords: *Dijkstra's algorithm, shortest path, TASUED Map, Navigation software, Graph*

1. INTRODUCTION

The importance of maps to human existence cannot be over-emphasized. Maps facilitate a spatial understanding of concepts, conditions, processes, events or things in the human world. According to [1], map is a representation of any part of the earth or the earth as a whole on some reduced scaled with latitudes and longitudes. In computing world, map is an iconic model which represents the real world on a smaller scale. It is a visual element that represents a specific geographical entity such as building, village, town, city, road, railway tracks, bridges, rivers, and mountains and so on, using standard icons, symbols and signs. Whether on a page of paper, bill board, walls or on a device that contains Global Positioning System (GPS) technology, maps enable commuter to organize information relating to the person's current location and the person's destination. A commuter travelling from City ***A*** to another City ***B*** in a particular geographical space is aware that network of cities are linked together by a communication channels (routes or roads) from his source to his destination. In the light of the above, there are several routes of varying magnitude from where he could choose from. Based on the above explanation, one can conceptualize map of a given geographical location as a *graph* representing a group of nodes (cities) called *vertices* and links or routes between them, technically known as *edges* [2-3].

Given a network of cities and distances between them, a commuter travelling from one city to another needs to take cognizance of the time taken and cost incurred from his source to his destination. This invariably, results into a problem of determination of the shortest possible path so as to optimize the available resources described in terms of time and cost. One practical way of handling this kind of problem is the understanding and application of graph theory algorithms which are capable of handling shortest path problems. Shortest Path Algorithms (SPAs) are algorithms that will find the best or least cost path from a given node (vertex) to another node [2-4]. Two most common SPAs exist: Bellman-Ford algorithm [5-6] and Dijkstra's algorithm [7-8].

Named after Richard Bellman and Lester Ford Jr., the Bellman-Ford algorithm computes shortest paths from a single source vertex to all the other vertices in a weighted digraph at a slower time than its equivalent Dijkstra's Algorithm [9-11]. The major advantage of Bellman-Ford algorithm is its capability of handling graphs with negative edge weights. However, using Dijkstra's algorithm to handle graph having negative cost values on the edges or links can result in an unwanted loop [2, 9]. It is therefore not practical to express the distance of roads in a city network using negative values. Hence, having negative cost values on the edges of a graph is not the focus of this paper.

Dijkstra's algorithm was conceived by Edsger W. Dijksra in 1956 and first published in 1959. Dijkstra's algorithm is the most common single-source shortest path algorithm which is efficient at finding the shortest paths between nodes of a graph, having similar representation as road networks [4, 10-12].

Shortest Path Algorithms (SPAs), otherwise known as Open Shortest Path First (OSPF), were specifically developed to improve the quality of service (QoS) in computer networks because the concept of computer networking heavily relies on graph theory [13]. Without SPAs, network traffic would have no direction thereby resulting into a loop which re-routes data packet back to the source node from where it was broadcast. OSPF remains the oldest dynamic routing protocol in a computer network and one of the Interior Gateway Protocol (IGP) group [3, 14]. OSPF uses a very simple calculation to determine the weight of edges. Specifically, OSPF finds bandwidth between two routers and then assigns its value in megabits per second to the cost (weight) of edges. There have been several revisions to this protocol since its release due to change in technology. The most recent release of OSPF is version 3 which has been used to implement IPv6. OSPF, for instance, Dijkstra's algorithms have been used as the building-block of many applications of everyday life. For example, Dijkstra's algorithm has been used as the principle behind the development of several navigation software packages. This is in line with the focus of this work In this article, Tai Solarin University of Education map is considered as a network of buildings (vertices) and roads (edges) linking them together with each edge having non-negative numerical values (cost/weight). Dijkstra's algorithm was used to simulate the shortest paths and minimum cost from the source building to the destination building thereby predicting the shortest paths from the source to the destination. This concept was then used to developed an Android-based navigation software tagged "TASUED MAP" which is capable of helping a new comer entering the university campus for the first time. Also, the software will be useful to old members of the university community who have to move from one location to another for meetings and to attend lectures by suggesting the shortest path to them so as to enable them arrive at their destinations in good time. Detail development of this software is presented in the proceeding sections.

The aim of this paper, therefore, is to demonstrate the implementation of TASUED map, Android-based navigation software, which explores Dijkstra's algorithm for finding the optimal route on campus by the new comer and the entire university community.

The remaining parts of this paper are organized as follows: section 2 presents the related literatures showing the application of Dijkstra's algorithm and its modifications which serves as the theoretical basis for this work. Section 3 features methodology, assumptions, Dijkstra's algorithm applied and MATLAB simulations. Section 4 presents the simulation results and the navigation software developed with detail discussions. Finally, Section 5 concludes and suggests areas of future improvement of the developed software.

2. RELATED WORKS

There are vast numbers of literatures on the application of Dijkstra's algorithm to the solution of real life problems. Few among them are presented below:

Reference [15] applied Dijkstra's algorithm to optimize the routes in a railway system so as to present the user with the shortest route. The proposed system provides users with regular and reliable information on rail system and serves as a framework for the development of future 'smart cities'.

Reference [12] tackled the problem of evacuating a building during critical incidents and disasters by determining the shortest route an evacuee could take in order to exit the building in a shortest possible time. This was achieved through the application of Dijkstra's algorithm.

Reference [10] proposed Vehicles Route Planning (VRP) for finding an optimal route from a car starting point to its destination. In this work, comparative analysis of Bellman-Ford and Dijkstra shortest path algorithms were conducted. Results from the simulation experiment confirmed that Dijkstra's algorithm perform best in terms of delay time and shortest travel path.

Reference [4] applied Dijkstra's algorithm simulated on a computer software called LINDO to the optimization of routes in a rail transit system among major stations in a proposed world-class university. The study which made use of Covenant University as a study case, predicted that the shortest path from the main entrance gate to the Electrical Engineering Department has a distance of 805 metre.

Reference [11] conducted a comparative analysis of Dijkstra's algorithm and Floyd algorithm for the determination of shortest path between two cities in China using time and cost parameters. The study which was simulated on Microsoft Visual Studio established that Dijkstra's algorithm is more effective than Floyd algorithm.

Reference [14] proposed a Dijkstra's algorithm which use the network connectivity information and the estimated distance information among the sensor nodes to find out the shortest path between the source and destination nodes with low cost. Simulated results from the study showed that Dijkstra's algorithm is capable of producing position estimates with 4% error range.

Other application areas of Dijksra's algorithms from the literatures include: determination of shortest path in very large scale integration (VLSI) routing [15]; optimization of flight trajectory in order to minimize fuel consumption and time-related cost [16]; and in the extraction of the vascular network of human retina [17].

This paper presents the extension of the previous works through the development of TASUED map navigation software package for optimal routing on campus. Definition of terms presented below remains germane for the understanding of the concept presented in this paper.

2.1 Definitions of Basic Terminologies

Consider a simple hypothetical graph shown in Figure 1. This graph is assumed to represent a network of buildings (small portion of Tai Solarin University of Education premise) with nodes representing the buildings. Each edge (link) has a non-negative numerical weight called the cost. The following definitions hold:

i. ***Graph:*** *A graph G = (V, E)* consists of a finite non-empty set of vertices ***V*** also called nodes and a finite set ***E*** of unordered pairs of distinct vertices called edges or arcs or links. In Figure 1, nodes represent the buildings with the *Node 1 = the university main gate* while *Node 21 = COSMAS building*. Table 1 shows the definition of each node in the graph (Figure 1).

ii. ***Source Node (u):*** The first node on the graph is called the source node. It is the starting point of traversing through the graph. In this work, university main gate is the starting point of travelling through the campus.

iii. ***Destination Node (v):*** This is the point of call, representing the last node to be visited during the traversal. List of destination nodes given in Table 2 match the definition of Table 1.

iv. ***Edge or Path:*** This is a sequential list of vertices which connect the node ***u*** and ***v*** together, with ***u = 0***. More formally,

$$p = v_1 \rightarrow v_2 \rightarrow \cdots \rightarrow v_k \quad (1)$$

where ***p*** is called the path, v_i is a set of nodes or vertices, v_1 is the node next to ***u*** and v_k is the destination.

v. ***Edge Weight:*** Edge weights can be given by a function;

$$w: E \rightarrow \tilde{R} \quad (2)$$

where ***w*** is the individual edge weight also known as the *cost*, ***E*** is the edge and ***R*** is non-negative value of ***w***.

vi. ***Distance:*** This is defined as the sum of edges of the all possible paths linking node ***u*** to ***v***. Formally,

$$Distance = w(p) = \sum_{i=1}^{k-1} w(v_i, v_{i+1}) \quad (3)$$

vii. ***Minimum Cost:*** After taking the sum of all possible edge weights from ***u*** to ***v***, the smallest of all the sums is known as the *minimum cost*. More formally,

$$\delta(u,v) = min\{w(p): path\ from\ u\ to\ v, where\ u,v \in V\} \quad (4)$$

viii. ***Shortest Path:*** This is defined as the path that gives the smallest sum based on definition (vii).

ix. ***Predicted Paths:*** This is the list of all possible nodes to be visited when traversing from the ***u*** to ***v***.

x. ***Efficiency:*** This is the total time taken for Dijkstra's algorithm to calculate the shortest path from ***u*** to ***v***.

3. METHODOLOGY

In our methodology, buildings of the university main campus are labeled using node numbers as shown in Table 1, starting from the main entrance gate. The interconnections of these buildings (nodes) with their respective distance (weights) are shown in Table 2. Dijkstra's algorithm presented in the next sub-section then finds the optimal routes by invoking the MATLAB commands outlined below:

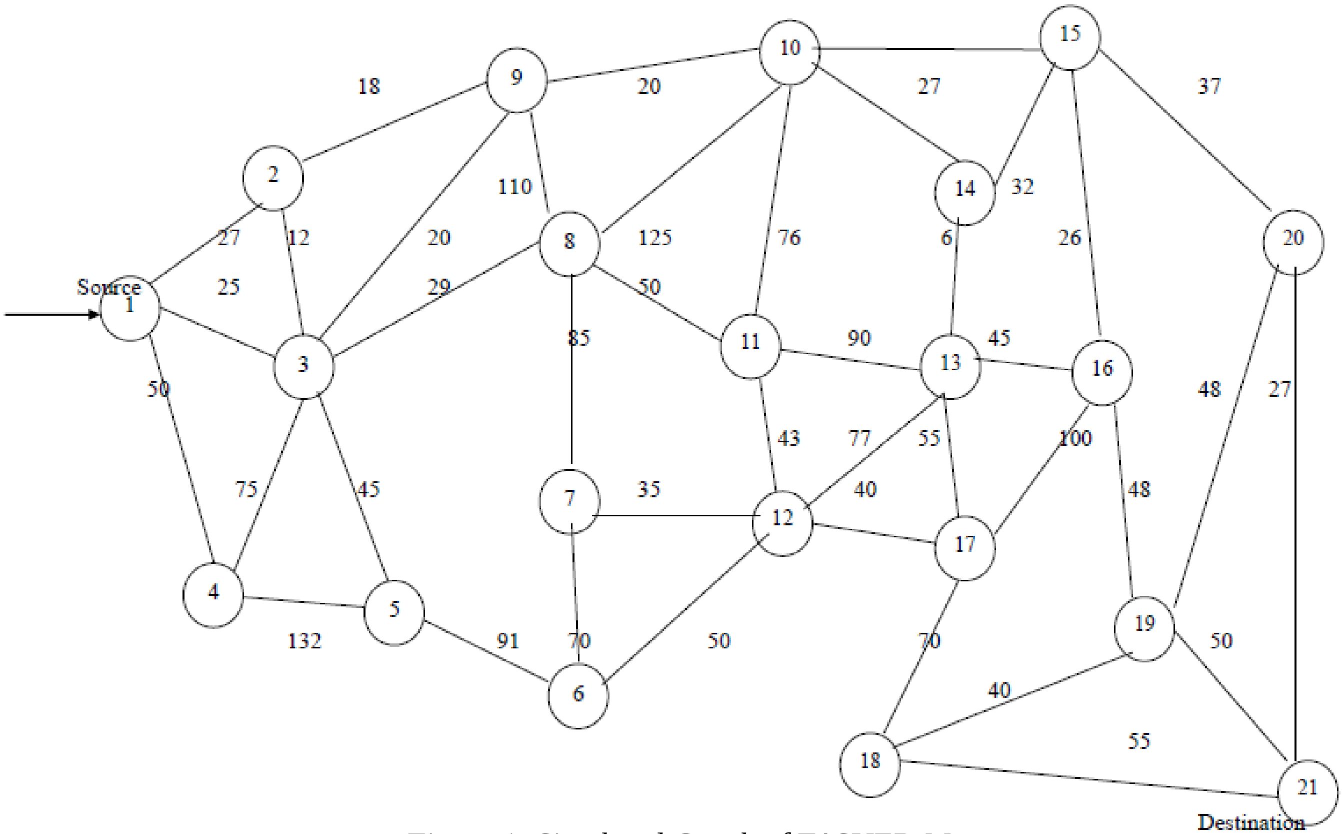

Figure 1: Simulated Graph of TASUED Map

3.1 Assumptions

For the purpose of our simulation experiment the following assumptions are considered:

i. Graph is connected

ii. All edges have non-negative weight

iii. Graph is undirected, though only one-way traversal is considered. It is not practical to say road network traffic flows in only one direction.

iv. The shortest distance in this work means shortest time expressed as cost.

v. Commuter or visitor moving within the premise does not wait on the way until he gets to the destination.

3.2 Dijkstra's Algorithm

Dijkstra's algorithm for the single-source shortest path problem solved in this research is presented below:

// G is the graph of TASUED map, V is a set of vertices where (u, v ϵ V), w is the distance from one vertex (node) to another, u is the source, v is the destination.

$Dijkstra\ (G, V, w, u, v)$

$d[u] = 0$

$for\ each\ v\ \epsilon\ V - \{u\}$

$\quad do\ d[v] = \infty$

$\quad S = \emptyset$

$\quad Q = V$

$\quad while\ Q\ \neq \emptyset$

$\qquad do\ u = ExtractMin(Q)$

$\qquad S = S \cup \{u\}$

$\qquad for\ each\ v \in adj\{u\}$

$\qquad\quad do\ if\ d[v] > d[u] + w(u, v)$

$\qquad\qquad then\ d[v] = d[u] + w(u, v)$

Table 1: Definition of Nodes which Represent TASUED Map Graph

S/N	Nodes	Name of Nodes
1	Node 1	University Main Gate
2	Node 2	COSIT or Science Complex
3	Node 3	Alex Onabanjo Petrochemical Complex (PETCHEM)
4	Node 4	CEPEP Building
5	Node 5	OGD Hall
6	Node 6	University Auditorium
7	Node 7	University Block
8	Node 8	COVTED Building
9	Node 9	CENVOS Office Complex
10	Node 10	Senate Building
11	Node 11	ICT Centre
12	Node 12	Lecture Theatre
13	Node 13	School of Postgraduate Studies
14	Node 14	ETF Building
15	Node 15	COHUM Building
16	Node 16	Adebutu Keshington Auto-Mechanical Laboratory
17	Node 17	Admission Office
18	Node 18	COSPED Building
19	Node 19	University Clinic
20	Node 20	Language Laboratory
21	Node 21	COSMAS Building

3.3 MATLAB Commands and Simulation Results

Table 2 shows list of all possible routes from Node 1 to Node 21 with their corresponding distance (weights). MATLAB command syntax below will return information similar to the content of Table 2 when run on the MATLAB Command Window:

```
>> net = sparse([ListSourcei], [ListDestinationi], [ListWeighti], DestNode, DestNode)
```

(5)

where *ListSource*$_i$ is a matrix of source nodes, *ListDestination*$_i$ is a matrix of destination nodes, *ListWeight*$_i$ is a matrix of corresponding weights or costs for $i \geq 1$, and *DestNode* is the final destination node respectively.

Results from the simulation experiment are presented in Table 3. These results feature the *Source Node, Destination Node, Minimum Cost, Shortest Path and Predicted Paths* respectively by implementing the Dijkstra's algorithm presented in the previous section. The experiment is conduction in ten (10) iterations or runs, using the MATLAB Command syntax presented below:

```
>> [minimum_cost, short_path, predicted_paths]=graphshortestpath(net, SourceNode, DestNode)
```

(6)

To obtain the graph of the University map shown in Figure 1, use the MATLAB Command below:

```
>> view(biograph(net, [ ], 'ShowArrows', 'off', 'ShowWeights', 'on'))
```

(7)

Table 2: List of Routes from Node 1 to Node 21

S/N	Routes	Descriptions	Distance or Weight (in meters)
1	Node 1- Node 2	From university main gate to COSIT	27
2	Node 1- Node3	From university main gate to PETCHEM	25
3	Node 2- Node 3	From COSIT to PETCHEM	12
4	Node 1 - Node 4	From university main gate to CEPEP Building	50
5	Node 3 - Node 4	From PETCHEM to CEPEP Building	75
6	Node 3 - Node 5	From PETCHEM to OGD Hall	45
7	Node 4 – Node 5	From CEPEP Building to OGD Hall	132
8	Node 5 – Node 6	From OGD Hall to University Auditorium	91
9	Node 6 – Node 7	From University Auditorium to University block	70
10	Node 3 – Node 8	From PETCHEM to COVTED Building	29
11	Node 7 – Node 8	From University block to COVTED Building	85
12	Node 2 – Node 9	From COSIT to CENVOS Office complex	18
13	Node 3 – Node 9	From PETCHEM to CENVOS office complex	20
14	Node 8 – Node 9	From COVTED Building to CENVOS office	110
15	Node 8 – Node 10	From COVTED Building to Senate Building	125
16	Node 9 – Node 10	From CENVOS office to Senate building	20
17	Node 8 – Node 11	From COVTED to ICT Centre	50
18	Node 10 – Node 11	From Senate Building to ICT Centre	76
19	Node 6 - Node 12	From University Auditorium to Lecture Theatre	50
20	Node 7 – Node 12	From University block to Lecture Theatre	35
21	Node 11 – Node 12	From ICT Centre to Lecture Theatre	43
22	Node 11 – Node 13	From ICT Centre to School of Postgraduate Studies	90
23	Node 12 – Node 13	From Lecture Theatre to School of PGD studies	77
24	Node 10 – Node 14	From Senate Building to ETF Building	20
25	Node 13 – Node 14	From School of PGD studies to ETF Building	6
26	Node 10 – Node 15	From Senate Building to COHUM Building	27
27	Node 14 – Node 15	From ETF Building to COHUM Building	32
28	Node 13 – Node 16	From School of PGD studies to Auto-Mechanical Lab	45
29	Node 15 – Node 16	From COHUM Building to Auto-Mechanical Lab	26
30	Node 12 – Node 17	From Lecture Theatre to Admission Office	40
31	Node 13 – Node 17	From School of PGD Studies to Admission Office	55
32	Node 16 – Node 17	From Auto-Mechanical Lab to Admission Office	100
33	Node 17 – Node 18	From Admission Office to COSPED Building	70
34	Node 16 – Node 19	From Auto-Mechanical Lab to University Clinic	18
35	Node 18 – Node 19	From COSPED Building to University Clinic	40
36	Node 15 – Node 20	From COHUM Building to Language Laboratory	37
37	Node 19 – Node 20	From University Clinic to Language Laboratory	48
38	Node 18 – Node 21	From COSPED building to COSMAS Building	55
39	Node 19 – Node 21	From University Clinic to COSMAS Building	50
40	Node 20 – Node 21	From Language Laboratory to COSMAS Building	27

Table 3: Results from the Simulation of Shortest Path from Dijkstra's Algorithm

S/N	Source Node	Destination Node	Minimum Cost	Shortest Path	Predicted Paths
1	1	21	156	1, 3, 9, 10, 15, 20, 21	0, 1, 1, 1, 3, 5, 6, 3, 3, 9, 8, 11, 11, 10, 10, 15, 12, 17, 16, 15, 20
2	1	10	65	1, 3, 9, 10	0, 1, 1, 1, 3, 5, 6, 3, 3, 9, 8, 11, 11, 10, 10, 15, 12, 17, 15, 15, 20
3	1	17	187	1, 3, 8, 11, 12, 17	0, 1, 1, 1, 3, 5, 6, 3, 3, 9, 8, 11, 11, 10, 10, 15, 12, 17, 16, 15, 20
4	1	19	136	1, 3, 9, 10, 15, 16, 19	0, 1, 1, 1, 3, 5, 6, 3, 3, 9, 8, 11, 11, 10, 10, 15, 12, 17, 16, 15, 20
5	1	13	194	1, 3, 8, 11, 13	0, 1, 1, 1, 3, 5, 6, 3, 3, 9, 8, 11, 11, 10, 10, 15, 12, 17, 16, 15, 20
6	1	4	50	1, 4	0, 1, 1, 1, 3, 5, 6, 3, 3, 9, 8, 11, 11, 10, 10, 15, 12, 17, 16, 15, 20
7	2	15	65	2, 9, 10, 15	NaN, 0, 2, 3, 3, 5, 6, 3, 2, 9, 8, 11, 11, 10, 10, 15, 12, 17, 16, 15, 20
8	2	11	91	2, 3, 8, 11	NaN, 0, 2, 3, 3, 5, 6, 3, 2, 9, 8, 11, 11, 10, 10, 15, 12, 17, 16, 15, 20
9	3	9	20	3, 9	NaN, NaN, 0, 3, 3, 5, 6, 3, 3, 9, 8, 11, 11, 10, 10, 15, 12, 17, 16, 15, 20
10	3	12	122	3, 8, 11. 12	NaN, NaN, 0, 3, 3, 5, 6, 3, 3, 9, 8, 11, 11, 10, 10, 15, 12, 17, 16, 15, 20

Note: NaN means Not a Number

4. RESULTS AND DISCUSSION

Table 3 shows the results of the application of Dijkstra's algorithm to the weighted graph of Figure 1, representing the map of TASUED. It can be observed from this table that the required shortest path from the source node (Node 1) to the destination node (Node 21) is as follows:

$$Node\ 1 \rightarrow Node\ 3 \rightarrow Node\ 9 \rightarrow Node\ 10 \rightarrow Node\ 15 \rightarrow Node\ 20 \rightarrow Node\ 21$$

with a minimum cost of 156 metre. By interpretation, starting from the university main gate, a visitor should go through Alex Onabanjo Petrochemical Complex, then pass through Senate building lane, then COHUM building, and finally through Language Laboratory to COSMAS building which is the extreme end of the university.

Other simulated results were obtained by varying either the source node or destination. For example, when the destination node is changed to Node 10 (Senate Building), Node 17 (Admission office), Node 19 (University clinic), Node 13 (School of Postgraduate Studies) and Node 4 (CEPEP building) while maintaining Node 1 (university main gate) as the source node, Dijkstra's algorithm returned 65 metre, 187 metre, 136 metre, 194 metre and 50 metre respectively as the shortest distance (minimum cost) that will be covered so as to get to these stations. Their respective shortest paths are shown in column 5 of Table 3. In similar manner, by changing the source node to Node 2 (COSIT or Science Complex), destination nodes to Node 15 (COHUM building) and Node 11 (ICT Centre), their respective minimum cost and shortest paths are shown in column 4 and 5 respectively. The same principle worked when the source node was changed to Node 3 (Alex Onabanjo Petrochemical Complex) and destination changed to Node 9 (CENVOS Office complex) and Node 12 (Lecture Theatre) respective.

The principle behind Dijkstra's algorithm was then used to develop an Android-based navigation software package called TASUED MAP. TASUED MAP offers satellite imagery of Tai Solarin University of Education map, 360º panoramic image of buildings and important locations on campus. The map front end was coded using JQuery-Mobile, JavaScript, XML, Ajax with Android Software Development Kit (SDK) for client side and the server side. It explores the Application Programming Interface (API) offered by Google in order to make it compactable with different versions of Android Operating System.

TASUED MAP's satellite view is a top-down (bird-eye) view, high resolution imagery and icons of geographical entities like buildings, road network, bridge, river and forest within the university campus. It is an aerial photograph taken from aircraft flying at the altitude of 800 to 1200 feet. Figure 2, 3, 4 and 5 show the views from TASUED MAP navigation software.

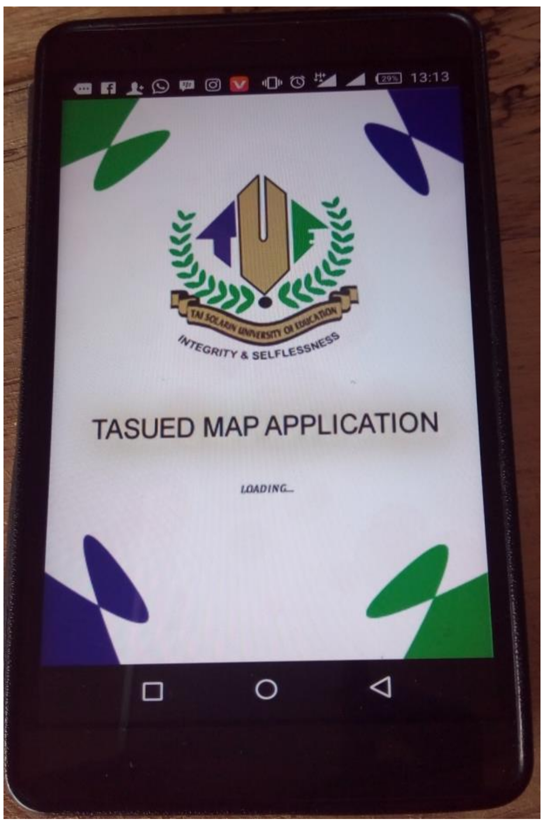

Figure 2: TASUED MAP Loading

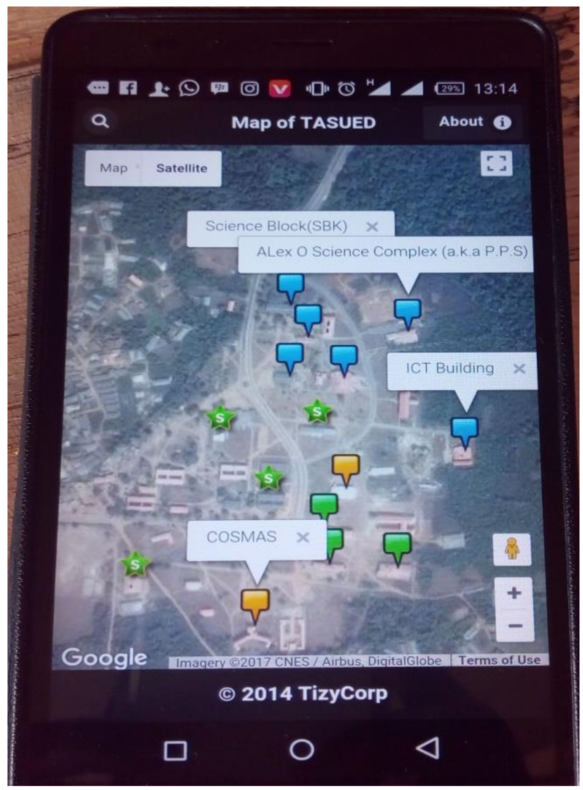

Figure 3: Aerial Photograph of TASUED

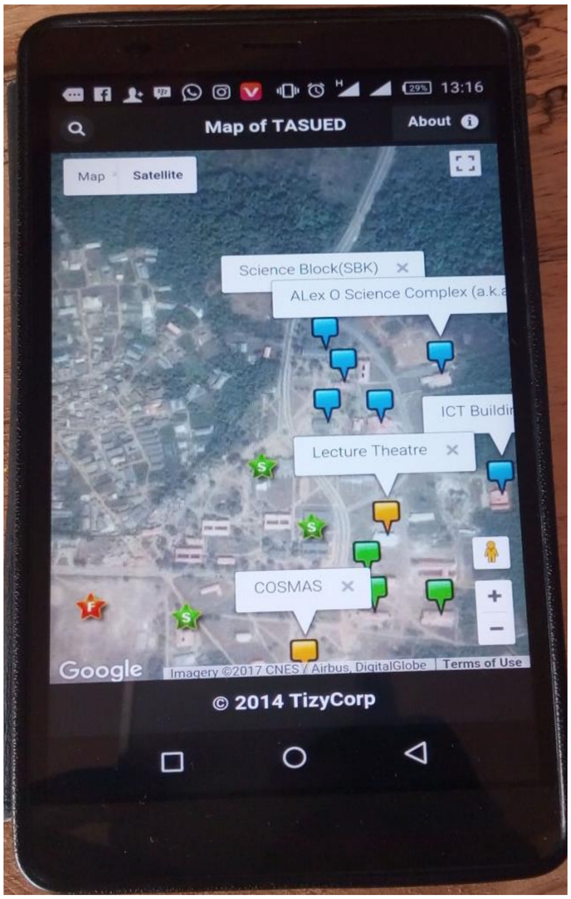

Figure 4: Campus and Neigbouring Village

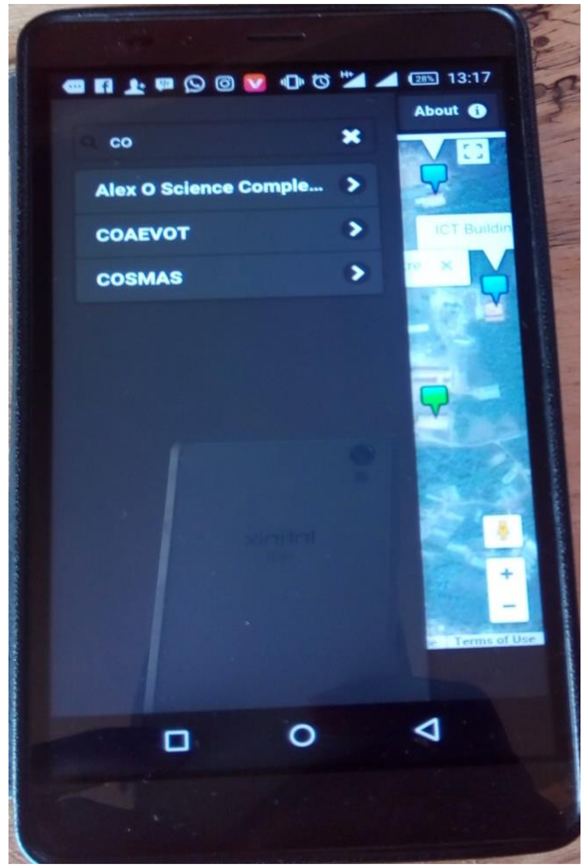

Figure 5: Map Search Engine

5. CONCLUSION

Finding optimal routing paths to a specific destination remains a practical challenge faced by commuters, especially when traversing a new geographical area for the first time in search of their daily routine. In this paper, we have successfully developed and implemented TASUED MAP navigation software package which enables visitors, fresh students and other members of the university community find the shortest path to their various destinations on campus from any point. The search for the shortest path was achieved by Dijkstra's algorithm, simulated using MATLAB. This serves as the building block for the development of the proposed navigation software. The paper corroborates the earlier studies on the application of Dijkstra's algorithm to solve real-life problems and also presents a major achievement in the realization of the future *'Smart Cities'*. It is hereby suggested that future investigation be directed towards the development of the 3D representation of image of each building on the map for easy recognition and identification. Research work may also be channeled towards making the mobile application compatible with other hand-held devices such as Windows phones and BlackBerry.

REFERENCES

[1] A. Shamsul, A.R. Kazi, and S. Selina, Secondary Geography for Classes IX-X. *(Bangladesh: National Curriculum and Textbook Board, Dhaka)*, 2011.

[2] G.A.V. Pai, Data Structures and Algorithms: Concepts, Techniques and Applications. *(Tata McGraw-Hill Education Private Limited)*, New Delhi, India, 2010.

[3] N. Gihan, and G.A. Wahied, Network Routing Protocol using Genetic Algorithms. *International Journal of Electrical and Computer Sciences (IJECS)*, 10(2), 2010, 21-28.

[4] M.C. Agarana, N.C. Omoregbe, M.O. Ogunpeju, Application of Dijkstra Algorithm to Proposed Tramway of Potential World Class University. *Applied Mathematics*, 7, 2016, 496-503.

[5] R. Bellman, On a Routing Problem. *Quarterly of Applied Mathematics*, 16, 1958, 87-90.

[6] L.R. Ford Jr., Network Flow Theory. *RAND Corporation, Paper P-923*, Santa Monica, California, 1956.

[7] E.W. Dijstra, A Note of Two Problems in Connexion with Graph. *Numerische Mathematik*, 1, 1959, 269-271.

[8] P. Frana, An Interview with Edsger W. Dijkstra. *Communication of the ACM*, 53(8), 2010, 41-47.

[9] A. Schrijver, On the History of the Shortest Path Problem. *Documenta Mathematica*, 2010, pp. 172.

[10] J. Shivani, and J. Singh, Route Planning in Vanet By Comparative Study of Algorithms. *International Journal of Advanced Research in Computer Science and Software Engineering*, 3(7), 2013, 682-689.

[11] A.K. Sangaiah, M. Han, and S. Zhang, An Investigation of Dijkstra and Floyd Algorithms in National City Traffic Advisory Procedures. *International Journal of Computer Science and Mobile Computing*, 3(2), 2014, 124-138.

[12] N.A.M. Sabri, A.S.H. Basari, B. Husin, and K.A.F.A. Samah, The Utilization of Dijkstra's Algorithm to Assist Evacuation Route in Higher and Close Building. *Journal of Computer Science*, 11(2), 2015, 330-336.

[13] K.D. Subodh, M. Deval, S.A. Shriniwa, P.S. Ajit, and K.S. Ashoke, Routing Choice Modelling using Fuzzy Logic and Adaptive Neuro-Fuzzy. *Modern Traffic and Transportation Engineering Research*, 2(4), 2013, 91-99.

[14] N. Pushpalatha and B. Anuradha, Shortest Path Position Estimation between Source and Destination Nodes in Wireless Sensor Networks with Low Cost. *International Journal of Emerging Technology and Advanced Engineering*, 2(4), 2012, 6-12.

[15] P. Pramod and D. Sunanda Railway Route Optimization System using Dijkstra's Method. *International Journal on Recent and Innovation Trends in Computing and Communication*, 2(3), 2014, 3435-3440.

[16] S. Peyre, D. Rautenbach and J. Vygen, A Generalization of Dijkstra's Shortest Path Algorithm with Applications to VLSI Routing. *Elsevier Journal of Discrete Algorithms*, 7, 2009, 377-390.

[17] M.M. Alejandro and B. Ruxandra, Lateral Navigation Optimization Considering Winds and Temperatures for Fixed Altitude Cruise using the Dijkstra's Algorithm. *Conference paper downloaded from ResearchGate*, https://researchgate.net/publication/270396636, 2014, Pp. 1-9, DOI: 10.1115/IMECE2014-37570.

[18] R. Estrada, C. Tomasi, M.T. Cabrera, D.K. Wallace, S.F. Freedman, and S. Farsiu, Exploratory Dijkstra Forest based Automatic Vessel Segmentation: Applications in Video Indirect Ophthalmoscopy (VIO). *Biomedical Optics Express*, 3(2), 2012, 327-339.