International Journal on Recent and Innovation Trends in Computing and Communication ISSN: 2321-8169
Volume: 6 Issue: 7 98 - 101

A Landmark Based Shortest Path Detection by Using A* and Haversine Formula

Ms. Megha G. Mathpal
Department of Information Technology
Government Engineering College
Modasa, Gujarat, India
mathpalmegha@gmail.com

***Abstract*—**In 1900, less than 20 percent of the world populace lived in cities, in 2007, fair more than 50 percent of the world populace lived in cities. In 2050, it has been anticipated that more than 70 percent of the worldwide population (about 6.4 billion individuals) will be city tenants. There's more weight being set on cities through this increment in population [1]. With approach of keen cities, data and communication technology is progressively transforming the way city regions and city inhabitants organize and work in reaction to urban development. In this paper, we create a nonspecific plot for navigating a route throughout city

A asked route is given by utilizing combination of A* Algorithm and Haversine equation. Haversine Equation gives least distance between any two focuses on spherical body by utilizing latitude and longitude. This least distance is at that point given to A* calculation to calculate minimum distance. The method for identifying the shortest path is specify in this paper.

***Keywords*-** *Haversine Formula, Dijkstra Algorithm, Google Map, XML.*

*****

I. INTRODUCTION

The downsides happened in past paper of shortest path discovery by utilizing Dijkstra calculation is recuperated in this paper by employing A* calculation. A* calculation is heuristic in nature. The point of Paper is to discover the route between two places inside a city entered by client utilizing the Intersections between Source and Goal intersections. The witticism behind it is to progress navigation of client inside a city; particularly in India where Town Arranging approach doesn't take after a standard rule for naming the diverse places. Most of the times an unknown person can't discover indeed the foremost popular places inside the city due to nonappearance of noteworthy identities. Hence the paper is planning to allow a suitable route to client by coordinating it through different intersections and streets which will be effortlessly distinguished by the related landmarks and a Google map. The route is given in two parts as: 1) Content route containing route giving a intersection to intersection movement to client in conjunction with the appropriate directions and turnings directing the client to induce the precise halfway intersections or landmarks 2) Google Map for correct requested route. Paper uses client-server engineering. Communication between them is entirely in XML for adaptability. The client has client interface from where an input is taken in XML for processing. The server comprises of a Java Processing Application and Database for it. The Database utilized by handling application could be a Social database containing whole information around city. The processing application after parsing request computes route between them with all necessary subtle elements with Latitude/Longitude for Google map and sends it as XML response. Client once more parsing response gets it on Client Interface with Google map handling done in JavaScript.

II. PROBLEM DEFINATION

The Point of Paper is to discover out the route in between two spots/junctions inside a city entered by client by making use of the Intersections in between the Source and Goal spots/junctions. The main motto behind it is to progress the navigation of client inside a city; particularly in Indian cities where Town Arranging approach doesn't take after a standard rule for numbering or naming the distinctive spots or places. Most of the times an obscure individual can't discover indeed the most famous places inside the city due to nonappearance of naming sheets or other noteworthy identities. Subsequently the project is intended to allow a suitable route to client by coordinating it through various junctions and streets which is able be easily identified by the related landmarks given with the route. The requested route is given to client in terms of the intersections present in between the source and goal route along with landmarks and streets interfacing them. The Landmarks utilized within the route may be noteworthy Buildings, Statues, Streets, Complexes, Landmarks, Sanctuaries, etc. The use of Landmarks includes an advantage of getting to the exact place having no critical personality whereas voyaging through route provided to client making the application friendly to the client obscure of the city to discover out the route in between any two spots or intersections within the city. The application is bound to deliver the shortest route providing a junction to junction movement to client in conjunction with the appropriate directions and turnings directing the client to induce the precise intermediate intersections (with their significant landmarks) or landmarks in specific zones in between two junctions/spots provided by client.

III. LITERATURE SURVEY

This paper contains "great circle distance" which speaks to the shortest path for distance modeling and optimal facility area on spherical surface. Great circle distances take into consideration the geometrical reality of the spherical Earth and offers an elective to broadly held idea that travel over water can be precisely displayed by Euclidean distances. The need for geometrical presentation of the spherical earth gets to be exceptionally important when we take into thought an ever-expanding junction inside a city. The utilize of "Great circle distances" opens another avenue for merging of Navigation and Spherical Trigonometry into progression of logistics and facility location. In this paper an assessment of distance area utilizing great circle distances is utilized to illustrate the application of the concept [4][3]. This paper proposes and executes a strategy for performing shortest path calculations taking crowdsourced information, in the form of imperatives and impediments, into consideration. The strategy is built on top of Google Maps (GM) and employments its directing benefit to calculate the most limited distance between two areas. Clients give the limitations and obstacles in the form of polygons which identify closed areas in the real world [5].

A.Haversine Formula

The Haversine formula is an equation important in navigation, giving great-circle distances between two points on a sphere from their longitudes and latitudes [4]. These names follow from the fact that they are customarily written in terms of the haversine function, given by haversin (θ) = sin2 (θ/2). The haversine formula is used to calculate the distance between two points on the Earth's surface specified in longitude and latitude. d is the distance between two points with longitude and latitude (ψ,φ) and r is the radius of the Earth. Translation to SQL statement[1] 3956 * 2 * ASIN (SQRT (POWER(SIN((orig.lat - dest.lat)*pi()/180 / 2), 2) +COS(orig.lat *pi()/180) *COS(dest.lat * pi()/180) *POWER(SIN((orig.lon - dest.lon) * pi()/180 / 2), 2))) AS distance [2].

B.A Algorithm*

A* uses a best-first search and finds a least-cost path from a given beginning node to one objective node (out of one or more possible objectives). As A* navigates the chart, it takes after a path of the least anticipated total cost or distance, keeping a sorted priority queue of alternate path segments along the way. It uses a knowledge-plus-heuristic cost function of node (usually indicated) to decide the arrange in which the search visits nodes in the tree. The cost function is a whole of two functions: • the past path-cost function, which is the known distance from the beginning node to the current node • a future path-cost function, which is an allowable "heuristic estimate" of the distance from to the goal

Pseudo Code:
function A*(start,goal)
closedset := the empty set // The set of nodes already evaluated.
openset := {start} // The set of tentative nodes to be evaluated, initially containing the start node
came_from := the empty map // The map of navigated nodes.
g_score[start] := 0 // Cost from start along best known path.
// Estimated total cost from start to goal through y.
f_score[start] := g_score[start] + heuristic_cost_estimate(start, goal) while openset is not empty
current: = the node in openset having the lowest f_score[] value
if current = goal
return reconstruct_path(came_from, goal)
remove current from openset
add current to closedset
for each neighbour in neighbor_nodes(current)
tentative_g_score:=g_score[current]+
dist_between(current,neighbor)
if neighbor in closedset
if tentative_g_score>= g_score[neighbor]
continue
if neighbor not in openset or tentative_g_score<g_score[neighbor]
came_from[neighbor] := current
g_score[neighbor] := tentative_g_score
f_score[neighbor]:=g_score[neighbor]+heuristic_cost_estimate(neighbor, goal)
if neighbor not in openset
add neighbor to openset
return failure;
function reconstruct_path(came_from, current_node)
if current_node in came_from
p:=reconstruct_path(came_fromcame_from[current_node])
return (p + current_node)
else
return current_node.

The above pseudo code assumes that the heuristic function is monotonic, which is a frequent case in many practical problems, such as the Shortest Distance Path in road networks. However, if the assumption is not true, nodes in the closed set may be rediscovered and their cost improved.

IV. SYSTEM DESIGN

The Point of the paper is to discover out the route in between any two spots inside a city entered by the client. This may be implemented using a client-server design where a request having two intersections as Source and Goal is sent from client to server and requested route is returned to client as a response from server. The client-server execution assumes that the client gets to the functional application remotely from client end to server one. This makes a clear thought of having client at one machine remotely accessing the application and server at the other. Hence the plan incorporates noteworthy components shown in functional project plan below: The client end consists of client interface from where an input is taken for processing. The server end consists of a Java Processing Application and Database for it.

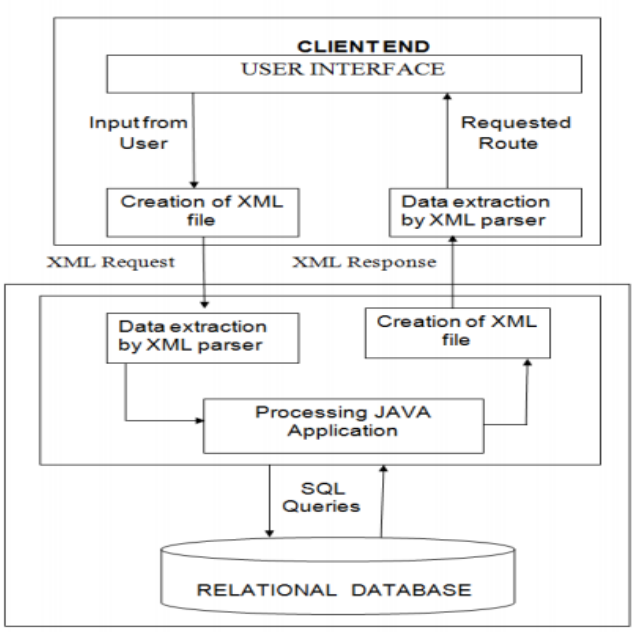

Fig.1 ConceptualSystem Design

The preparing application essentially takes as it were begin and end junctions and computes the route in between them with all fundamental subtle elements having middle intersections with landmarks and streets in a specific region. The Database utilized by preparing application may be a Social database containing entirety data almost city in terms of intersections, landmarks, streets and areas. The input containing source and goal intersections for the asked route is sent to the server conclusion as a request from client end. This request is inserted in a XML record can be called as XML request to be sent to server. At server on accepting a XML request; it is provided to a XML parser for extricating fundamental information i.e. source and destination junctions which are in turn provided to Java Processing application as an input. This application computes a requested route (a most limited one) by connection with the database utilizing SQL inquiries to get vital data for computation. For a computed route to be sent to client, it is once more implanted into a XML forming a XML response. This reaction on accepting at client end is once more sent to a parser to extricate a route to be shown to the client at to user interface.

A. Shortest route in the form of text route:

A client has arrangement to know the shortest way from source to goal in two ways content-based route and graphical route by utilizing Google outline. A content-based route gives correct way from source to goal in the shape of directions, turns, middle spots and distance between that spots. A path is given to client by utilizing SQL query. At last it gives the overall shortest distance from source to destination.

Fig.2: Text Route

B. Shortest route graphical representation:

Graphical representation of shortest route is appeared in figure. It highlighted the shortest course from source to destination. Client can utilize both the procedures to effortlessly know the route between source to goal. GPI provides different strategies to get to the highlighted route.

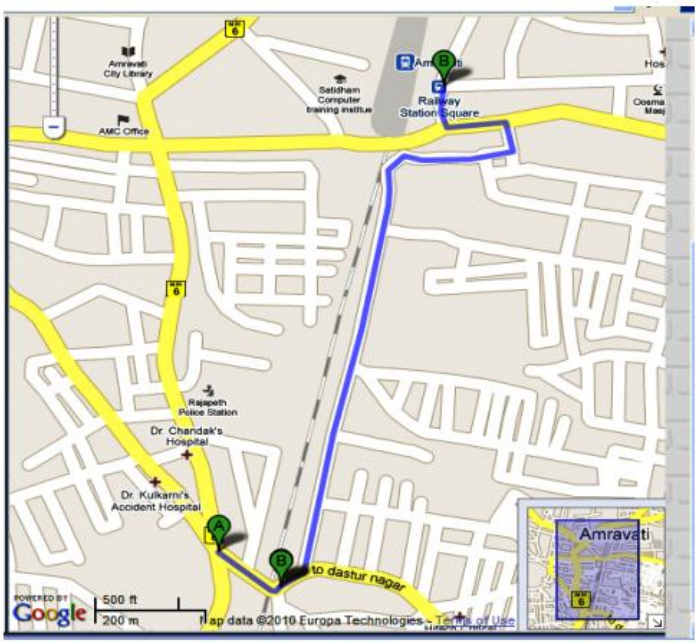

Fig. 3 Graphical Representation

V. FUTURESCOPE

The downsides happened in past paper of shortest way discovery by utilizing Dijkstra calculation is recovered in this paper by employing A* algorithm. In this paper, we utilize A* calculation for deciding shortest way between two junctions. But A* is heuristic in nature it implies that A* algorithm does not grant any guaranty for great solution. This is property of any heuristic algorithm A* algorithm. So, able to utilize any algorithm related with A* algorithm which can donate best arrangement to calculate shortest path between two cities. A* algorithm is combination of Dijkstra algorithm and Breadth First Search algorithm. In expansion to that, A* is heuristic in nature. This System can be applied as a navigation system which can navigate through out city. Together with intracity shortest path detection, we will implement same concept for intercity.

VI. CONCLUSION

"Landmark Based Routing in Indian Cities" is bound to allow the briefest route giving a junction to junction movement to client together with the appropriate bearings and turnings directing the client to urge the precise intermediate junctions (with their noteworthy points of interest) or points of interest in specific zones in between two junctions/spots provided by user. The client moreover gets correct route with direction of inserted Google Map.

VII. ACKNOWLEDGEMENT

We would like to thank Government engineering college, Modasa for providing all the required amenities. I am also grateful to Prof. J.S. Dhobi, Head of Information Technology Department, GECM, Gujarat for their indispensable support, suggestions and motivation.

REFERENCES

[1] By Mr. Reid "Shortest distance between two points on earth" http://wordpress.mrreid.org/haversine-formula/ This is an electronic document. Date of publishing 20/12/2011.
[2] Samuel Idowu, Nadeem Bari, "A Development Framework for Smart City," Luleå University of Technology International journal of Computer Application, vol 6, 9 Nov. 2012
[3] Javin J. Mwemzi, YoufangHuang," Optimal Facility location on spherical surfaces", New york science Journal, April 2011.
[4] Ben Gardiner, Waseem Ahmad, Travis Cooper,"Collision Avoidance Techniques for unmanned Aerial Vehicles", Auburn University, National Science Foundation, 08/07/2011.
[5] Simeon Nedkov, SisiZlatanova, "Enabling Obstacle Avoidance for Google maps",June 2011.
[6] Bing Pan,john C. Crotts and Brian Muller,"Developing Web Based Tourism Information using Google Map" Departemnt of Huminity and Tourism Mangement,Charston ,USA.
[7] ElinaAgapie.jason Ryder, Jeff Burke, Deborth Estrin," Probable Path Interference for GPS traces in cities", university of California,2009.
[8] K.M.Chandy,J. Misra, "Distributed Computation on Graphs: Shortest Path Algorithm", University of Texas, March 1982.
[9] Siemens AG. Munchen, Ulrich Lauther "An Extremely Fast Exact Algorithm for Finding Shortest Path in static network with geographic backgroung", International journal on Computer science and Engineering, June2006.
[10] Philip Klein,Satish Rao ,Monika Rauch, Sairam Subramnyam, "Faster Shortest Path Algorithm for Planner Graphs", March 1994.
[11] Andrew v. Goldberg, Haim Kaplan, Renato F. Werneck, "Efficient Point to point shortest Path Algorithm", Internation Research on Advance Research in Computer Science and Software Engineering, Oct 2005.