Conceptual Paper

Open Access CrossMark

A real time train track tracking system: design and implementation for Indian railway

Abstract

Development of population is directly related to transportation facility. In India, transportation facility mainly handled by Indian railways. Maintenance and routine checking of railway track is a big issue. In this paper we proposed a method for real time tracking of rail track for Maintenance and routine checkup. In Our proposed method, openlayer-2 is used for implementation purpose. In proposed method, we provide details on the train track within slots of 2km. Whole train track is divided into various slots with a distance of 2km length. During the maintenance period or routine checkup of tracks, Maintenance team is able find out the inspected track, uninspected track within 2km distance. Green line from one point to another point of rail track shows that the track is successfully inspected and no further inspection is required. Red line show the track length which has not been inspected successfully. The objective of this manuscript is to provide information about inspected and uninspected rail track within 2km range automatically for maintenance team. Which helps to reducing search time and track down problem.

Keywords: tracking system, openlayer-2, real-time, track maintenance

Volume 2 Issue 6 - 2018

Shreya Mathur, Sunil Pathak, Sunil Gupta
Department of Computer Science & Engineering, Poornima College of Engineering, India

Correspondence: Sunil Pathak, Department of Computer Science & Engineering, Poornima College of Engineering, Jaipur, India, Tel 9928317662, Email sunilpeth@gmail.com

Received: October 31, 2018 | **Published:** December 05, 2018

Introduction

We have seen that how much of our lives have been made easy via live tracking apps. If we need to go somewhere and need a cab for that we can easily go for booking a cab via Ola or Uber and get live tracking of the cab. Other implementations include tracking your online orders in real time, tracking trains and buses you need to catch etc. This leads to decrease in manual power to keep track of all such details. But this feature is not utilized as much as it can be. This feature of live tracking is used in keeping track of stations right now by railways. But it is not used to keep track of the investigations and checks performed at the stations. This can reduce the amount of paperwork and manual power involved in such a task. So this project tackles to solve this problem. The report looks into the factors involved in understanding what it is I was planning to implement and how I was going to implement it. Below listed are the various technologies that were used in giving this project a structure.

System overview

The India is a very fast developing country throughout the world. Becoming a developed country, overall development is very much needed like education, research, latest technology, quality of services; employment and transport etc. in India, large volume of passengers are traveled through Indian railways which increases extra load on rail tracks. Because of busy schedule of railway tracks, it is very hard to manage routine Maintenance of tracks manually.

In VISyR[1] is a patented visual inspection system for maintenance purpose in Railway. FPGA based rail detection and tracking block are used to automatically detects and track the rail head. In[2] assessment of Level crossings (LCs) are identified and marked as dangerous security points for both road and rail transport. A global model for rail and road is proposed for Level crossings areas. It is recommended that each Level crossing is frequently inspected very carefully because of heavy load of rail and road transport. In[3] various kind of obstacles are detected during the track inspection to insure the prevent track failure. The track was inspected with the speed of 100 km/h. During conducting such experimental setup few parameters were missing such as total load on track, age of track and quality of tracks. The tracking schedule of rail track is depending on above parameters. In[4] GPS /GSM based train tracking system were introduced to determine whether the trains are running on a track which will properly inspected using utilizing mobile networks for public transpiration support. In[5] proposed multisensory system of detecting obstacle on railway track has been proposed using complementary use of IR and US barriers. In metropolitan areas, crowded traffic consequences in a large number maintenance system are required at a very high speed. In[6] Fast Driver Assistance System is proposed to detect obstacles from train track and warns the driver to avoid the collision.

We have observed that our lives become much easier because of technology such as live tracking apps. If we are planning to visit somewhere via a cab. We can easily book and monitor the real location using live tracking system of cab. There are huge numbers of examples for real time online tracking system such as real time train tracking system, online order tracking system and speed post tracking system etc. Tracking real time online object leads to decrease manual power to keep Track of all such details. Live tracking is used in keeping track of stations right now by railways. But it is not used to keep track of the investigations and checks performed of track at the stations. Which can overcome reduce the amount of paperwork and manual power who are involved in such a task. This manuscript proposed a new method to real time train track maintenance scheme (Figure 1).

Implementation setup

Openlayer2 is used to implementation of dynamic map and responsible to show outline, vector information and markers stacked from any source. OpenLayer2 supports Geography Markup Language (GML), GeoRSS, GeoJSON, KML, Geography Markup Language (GML) and map data from any source using OGC-standards as Web Map Service (WMS) or Web Feature Service (WFS)[7] JSP, Apache

Submit Manuscript | http://medcraveonline.com MedCrave

© 2018 Mathur et al. This is an open access article distributed under the terms of the Creative Commons Attribution License, which permits unrestricted use, distribution, and build upon your work non-commercially.

Tomcat with servlet to deploy Java Server Pages. Javascript,[8,9] MySQL, Java and HTML are other software packages which are required during implementation process.

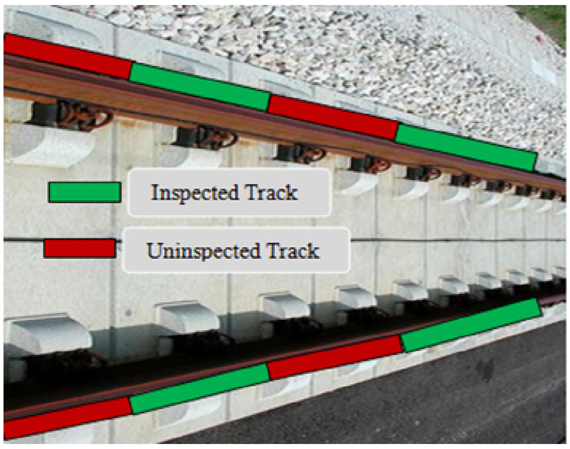

Figure 1 Show various inspected and uninspected track slots.

System architecture

In proposed system user will enter values up to 2km from his/her current location to identify the inspected/un-inspected track. The request is compared with the value stored into railway data base. Based on the result if the track is already inspected it will shows as green color and the inspection team will not inspect such track up to 2km and this process will continue till all the track is not inspected or find out the un-inspected track (Figure 2).

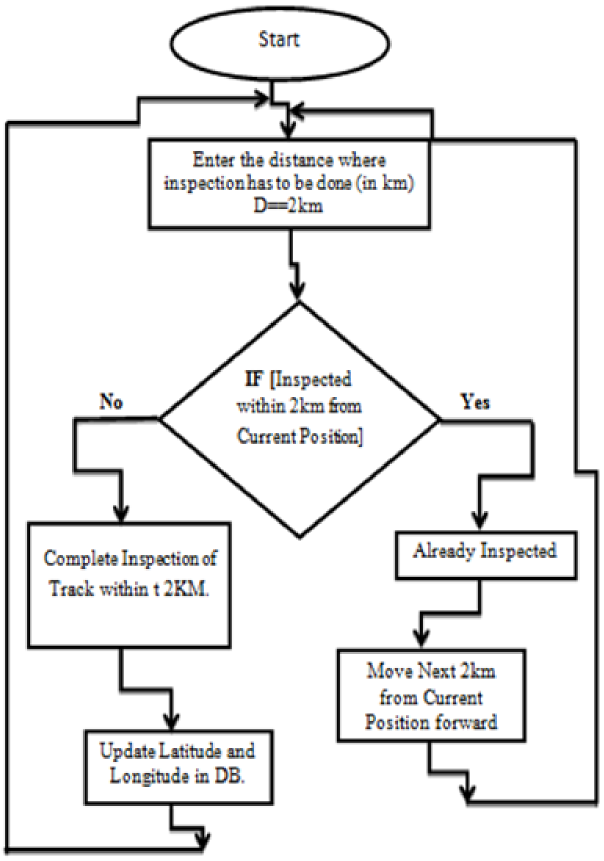
Start
Enter the distance where inspection has to be done (in km) D==2km
IF [Inspected within 2km from Current Position]
No
Yes
Complete Inspection of Track within t 2KM.
Already Inspected
Move Next 2km from Current Position forward
Update Latitude and Longitude in DB.

Figure 2 Working model of proposed system.

Proposed real time track tracking system is implemented using following steps:

Step: 1 Creation of LED fication DAO. java file: This file is responsible to provide direct access of database from the railway server to the inspection team. MySQL data base is used for designing database and the JSON object is used for displaying structural data based and transmission of data using JavaScript objects. The details fetched from the database were displayed on the map with the help of below source code (Table 1).

Table 1 Source code to implement class LED fication Dao1. java file

```
public class LEDficationDao1 {
 public String getLEDMapA()
 {
 Connection con = null;
 JSONObject masterVO= new JSONObject();
 JSONArray capit = new JSONArray();
 JSONArray plac = new JSONArray();
 JSONObject mapVO = null;
 try{
 Class.forName("com.mysql.jdbc.Driver");
        con=DriverManager.getConnection(        "jdbc:mysql://localhost:3306/
cris","root","catwoman");
 Statement stmt=con.createStatement();
            ResultSet rs=stmt.executeQuery("select * from Capital");
            while(rs.next())
            mapVO = new JSONObject();
            mapVO.put("place_name", rs.getString("Place_Name"));
            mapVO.put("longitude", rs.getString("lng"));
            mapVO.put("latitude", rs.getString("lat"));
            capit.put(mapVO);
            }
            rs=stmt.executeQuery("select * from placename");
            while(rs.next())
            {
            mapVO = new JSONObject();
            mapVO.put("PLaceName", rs.getString("name"));
            mapVO.put("longitude", rs.getString("lng"));
            mapVO.put("latitude", rs.getString("lat"));
            plac.put(mapVO);
            }
            masterVO.put("capitalMapVOs", capit);
            masterVO.put("placeMapVOs", plac);
            System.out.println(masterVO.toString());
            }
            catch(Exception ex)
                        {ex.printStackTrace();}
                        try
                        { con.close(); }
             catch(Exception e)
                        { e.printStackTrace();}
 return masterVO.toString();
}
}
```

Step: 2 Creation of latest.js file: This code is used to add layers, set their visibility and handle popups depending on the functionality. This code implements Openlayers2, different settings related to layers like transition, opacity, zooming etc. and various buttons in the display and their button actions.The designing of popups (using Openlayers2 for geometry) and text styling is also done via below source code (Table 2).

Table 2 Creation of latest.jsp file

```
var obj;
function plotLEDMap(mapdata,distance){
pointFeatures = [];
var colorcount = 0;
capitLayer.removeAllFeatures();
obj = JSON.parse(mapdata);
var radius;
for ( var i = 0; i < obj.placeMapVOs.length; i++) {
lat=Number(obj.placeMapVOs[i].latitude);
lng=Number(obj.placeMapVOs[i].longitude);
 radius =25;
var d = getDistanceFromLatLonInKm(clat,clon,lat,lng);
if(d<=distance){
    pointGeometry = new OpenLayers.Geometry.Point(lng,lat).transform(geographic, mercator);
 var prcn= radius;
 var pointstyle0 = OpenLayers.Util.extend();
 pointstyle0.strokeColor = '#ffffff';
 pointstyle0.label      = obj.placeMapVOs[i].PLaceName+'\n'+radius+'%';
 pointstyle0.fontSize = "11px";
 pointstyle0.fontWeight = "bold";
 pointstyle0.fontColor = "#ffffff";
 pointstyle0.fillOpacity = "1";
 pointstyle0.pointRadius = 20;
 pointstyle0.fillColor = lineColors[colorcount];
var a = obj.placeMapVOs[i].PLaceName;
var b = pointstyle0.fillColor;
var msg='<table style="border:0px;font-family:arial;font-size:11px;">'+
'<tr><td colspan="2"><b>('+obj.placeMapVOs[i].PLaceName+')
 </b></td></tr>'+'</table>';
var pointFeature = new OpenLayers.Feature.Vector(pointGeometry,
{latitude:lat,longitude:lng,div:a,divcolor:b,message:msg,cltn:prcn},pointstyle0);
 pointFeatures.push(pointFeature);
colorcount++;
capitLayer.addFeatures(pointFeatures);
}
map.addLayer(capitLayer);
 capitLayer.setVisibility(true);
}
```

Step: 3 Creating of LED fication Map. jsp file: This file serves the purpose of main function and responsible for calling all other sub functions. Basically every function created under LEDficationDAO1. java and latest.js is being implemented using below source code (Table 3).

Table 3 LED fication Map. jsp file

```
LEDficationDao1 d1 = new LEDficationDao1();
System.out.println(d1.getLEDMapA());
function test()
{ ledMap(); }
function ledMap(){
LEDficationDao1.getLEDMapA(
function (data)
 {
 plotLEDMap(data);

 });
```

Simulation result and analysis

After successfully implementation of source code, various test cases were applied to check the marker position movement. The changes were noted as per the database and location changed and code responded to all of it. This image is a test case as 0.5 km was fed into the top left bar and it displayed the nearest places in the radius of 0.5 km (Figure 3–7).

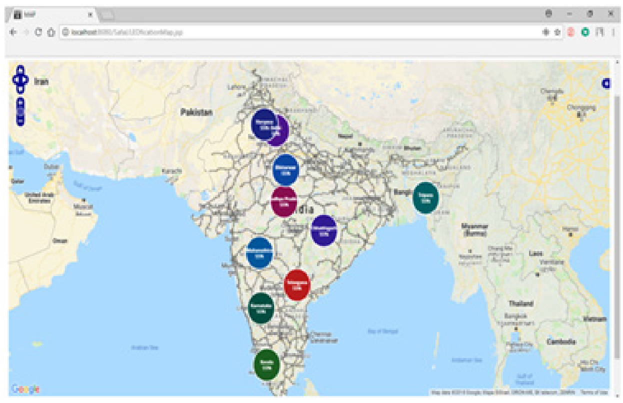

Figure 3 Information related to various layers and pop-ups of places that are in the database.

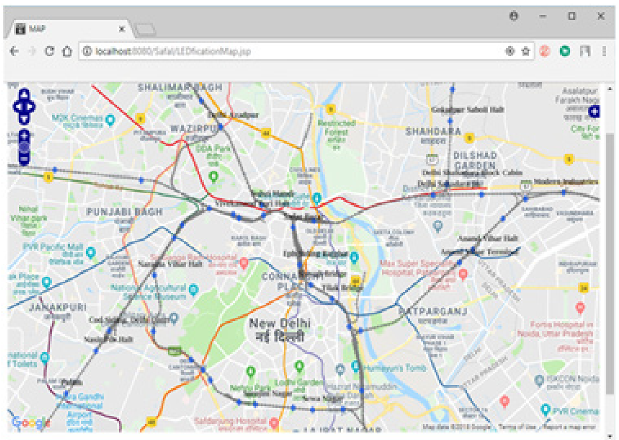

Figure 4 Image showing the change in the visibility of layers on zooming in through various pan buttons displayed on the top of left most corners.

Copyright: ©2018 Mathur et al.

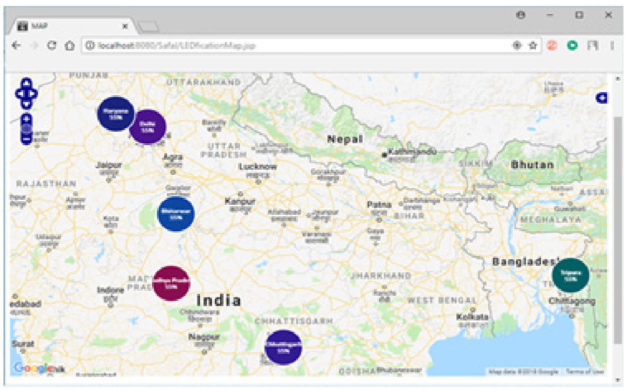
Figure 5 Another image of the map and the various layers.

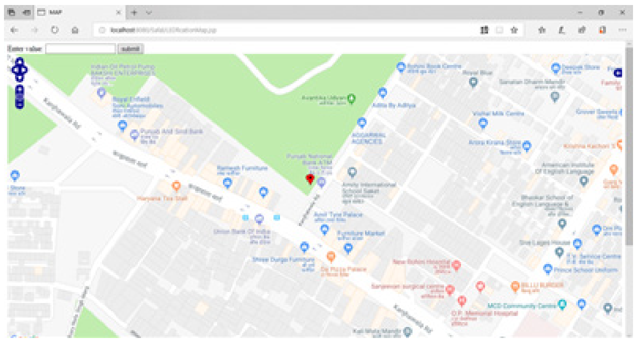
Figure 6 Image showing the marker and other places around it.

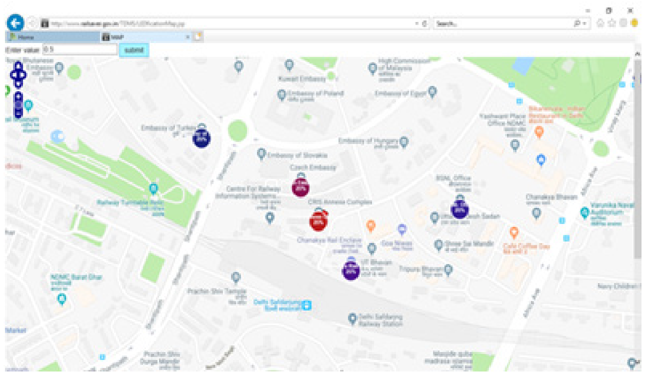
Figure 7 Image showing the marker and other places around it.

Conclusion and future scope

In this manuscript, we have release a real time railway track tracking system to minimize the search time required to find out inspected or uninspected track with an accurate location, cost effective. The proposed system provides the capability to track 2km from current position of the railway track. The proposed system also provides information regarding the inspected track and uninspected track with other relevant information such as last inspected date, expected date of inspection, name of inspected team and major faults identified and correct during last inspection etc.

The prototype mode has been tested experimentally and the results are analyzed. The experiments are conducted in different areas on New Delhi, India for Indian Railways. The cost is very lesser as compared to previously used manual train tracking system. In future we will try to incorporate few major correction such as if track is not inspected after due date, message will send to train driver to slow down the speed of train and to the competent authority of railway to make necessary arrangement to carried out inspection as soon as possible.

Acknowledgments

None.

Conflicts of interest

The author declares there is no conflicts of interest.

References

1. F Marino, A Distante, PL Mazzeo, et al. A Real Time Visual Inspection System for Railway Maintenance: Automatic Hexagonal Headed Bolts Detection. *IEEE Transactions on Systems, Man and Cybernetics-Part C.* 2007;37(3):418–428.
2. Mohamed Ghazel. Using Stochastic Petri Nets for Level- Crossing Collision Risk Assessment. *IEEE Transactions On Intelligent Transportation Systems.* 2009;10(4):668–677.
3. Milan Ruder, Nikolaus Mohler, Faruque Ahmed. An Obstacle Detection System for Automated Trains. *IEEE IV2003 Intelligent Vehicles Symposium*. 2003.
4. D Jayakody, M Gunawardana, N Wicrama, et al. GPS/GSM based train tracking system – utilizing mobile networks to support public transportation. 2010.
5. J Garcia, J Urena, A Hernandez. Efficient Multisensory Barrier for Obstacle Detection on Railways. *IEEE Transactions on Intelligent Transportations Systems.* 2010;11(3):702–713.
6. NV Murali, C Chandramouli, V Agarwal. A cost-effective ultrasonic sensor-based driver-assistance system for congested traffic conditions. *IEEE Transactions on Intelligent Transportation Systems*. 2009;10(3):486–498.
7. Direct Web Remoting. 2018.
8. *Calculate distance between two latitude-longitude points*? 2018.
9. Herb Schildt. *Java: The Complete Reference 8th Edition*. 2011.