The Development of an IoT Instrumented Bike: for Assessment of Road and Bike Trail Conditions

Peijie Qiu[1], Xilun Liu[1], Siwei Wen[1], Yuchang Zhang[1], Kyle N. Winfree[2], Chun-Hsing Ho[3]

***Abstract*— The purpose of this paper is to present a method to detect bike trail conditions while displaying real-time data on a mobile application. Using a Global Positioning System (GPS) unit, microprocessor and multiple accelerometers built on the concept of Internet of Things (IoT), the entire device employs real-time data to collect, analyze, and identify bumps and cracks in public infrastructure. The main purpose of this research is aimed at helping the government evaluate the severity of road conditions to better assist in strategic decision making about maintaining or repairing damaged infrastructural areas as well as to notify pedestrians of potential bumps or cracks along their route. The authors propose an algorithm that sets a window range to vertically process the raw accelerometer data to accurately distinguish bumps. By integrating this algorithm onto a mobile application, the data collected by this program can be shared between users to help avoid potential risks or dangerous road conditions and to assist in the evaluation of infrastructural deterioration.**

I. INTRODUCTION

Instrumented Bikes are a new, innovative field. They are built on the concept that cars use a variety of sensors built inside smart phone to gather data about the environment and focus on extending this idea to bicycles in order to analyze and report trail conditions [1]. As transportation and the use of infrastructure increases, it is becoming imperative that the conditions of roadways and trails unreachable by cars are assessed. Utilizing the popularity and prevalence of the Internet of Things in modern society [2,3,4], and this technology provide the network and platform to connect physical devices. Through use of microprocessors, sensors, and software, Instrumented Bikes communicate and exchange data between users [5,6]. The information collected from this process is used to analyze and detect bumps in roads as well as assess infrastructural needs [7]. To accomplish this, this technology uses microprocessor, sensors, the cloud, and a mobile platform as depicted in Fig. 1 [8,9].

A. Implementation of Internet of Things

Instrumented Bike furthers similar research in the field and completes it by incorporating the Internet of Things. As in Fig. 1, all devices including the microprocessor, accelerometers, Global Positioning System (GPS) unit, mobile application, and the cloud are connected. In this way, the technical details of Instrumented Bikes revolve as much around detecting bumps as it does the process of servicing and maintaining its mobile application and cloud storage. Realizing infrastructural assessment requires the implementation of real-time data from many different sources. Thus, a substantial aspect of this technology that improves upon other iterations is its ability to connect and gather this data from multiple sources for better data analysis.

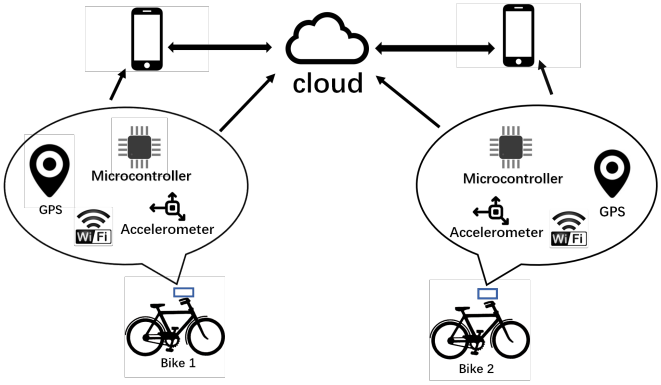

Fig.1 Internet of things of Instrumented Bike: Bike 1 and 2 can transmit data, including location coordinates and accelerations, to a cloud space directly or via a mobile application. Anyone who has signed up can access the cloud to download data. The following is the type of data and data structure in the Firebase database as shown in Fig.2.

```
"location" : {
    "-LA5_ra5shXixr1f-Yd2" : {
      "Acc1_x" : -0.07,
      "Acc1_y" : -0.33,
      "Acc1_z" : 0.85,
      "Acc2_x" : 0.17,
      "Acc2_y" : -0.05,
      "Acc2_z" : 0.9,
      "latitude" : 35.18031,
      "longitude" : -111.6563
    },
```

Fig.2. The type of the data and the data structure in the Firebase database. The number "$LA5_ra5shxixr1f - Yd2$" stands for the data push key for each child in the JSON data tree.

[1]P. Qiu, X. Liu, S. Wen, Y. Zhang, are senior students in Electrical Engineering in the School of Informatics, Computing, and Cyber Systems at Northern Arizona University, P.O. Box 15400, The United States.

[2]K. N. Winfree, is an Assistant Professor in the School of Informatics, Computing, and Cyber Systems at Northern Arizona University, 1295 S. Knoles Dr., The United States and is the contact author.

[3]C. Ho is an Associate Professor in the Department of Civil and Environmental Engineering at Northern Arizona University, 2112 S Huffer Ln., The United States.

978-1-5386-5638-9/18/$31.00

B. Accuracy of Global Positioning System (GPS)

This project used a dedicated Global Positioning System (GPS) unit to gather data. This enabled the device to receive accurate, real-time latitude and longitude for the bike's location.

C. Algorithm of Data Processing

The authors referred to substantial previous research on data processing such as the Fast Fourier Transform method [10] and Moving Average method [11,12], but these techniques require computation at power that exceeds the capability of our microcontroller; details will be discussed later. As a result, the authors present a method called the Window Interpolation Algorithm to solve these issues [13]. This can filter the data gathered from the accelerometer more accurately using a vertical analysis of data to accurately distinguish bumps from regular pavement.

This enables the device to be more applicable in assessing the needs of infrastructure and, once implemented, could provide valuable information on road conditions. By extending this technology to commuters and travelers, Instrumented Bike will provide all drivers and bicyclists the ability to contribute to urban and rural road assessment and construction.

There are five parts to the research performed for our Instrumented Bike. In the section II, the authors present how the components are connected within the device itself. Section III explains the data transmission algorithm including the TCP/IP protocol and the program code of the Particle Photon. After, section IV discusses the data processing and how the Window Interpolation Algorithm is used effectively to guage road conditions. Sections V presents the mobile application and the website. The final conclusion is mentioned in section VI.

II. DATA COLLECTION

To implement the function of data collection, a vibration measurement system was needed. This system required three parts: sensors (accelerometers), a microprocessor (Arduino NodeMcu or Particle Photon), and the Firebase platform to write and read data. In order to improve robustness of the data collected, this project connected two accelerometers to the micro-controller through an I^2C interface (Inter-Integrated Circuit) which assigns a specific address to each accelerometer and compares the data from each to ensure the quality of data passed to the mobile and web platforms. Each device was configured with a different address. When traveling on bike trails and on urban roadways, the vibration measurement system with the accelerometer detects vibration in three dimensions. The locational coordinates are identified by the Global Positioning System (GPS) which uses a TX port to communicate with the data logger. As a result, the precise locational coordinates of the bike can be recorded simultaneously. Data collected from the device is stored in local storage (SD card) through the Openlog. Thus, the entire device forms the fundamental data acquisition device for this project. The details on device connection and the component layout are presented in Fig. 3 with the information on Arduino NodeMcu. Fundamentally, the connection of Arduino NodeMcu and Particle Photon is the same. However, the method of data transmission is different and they will be detailed in the following section.

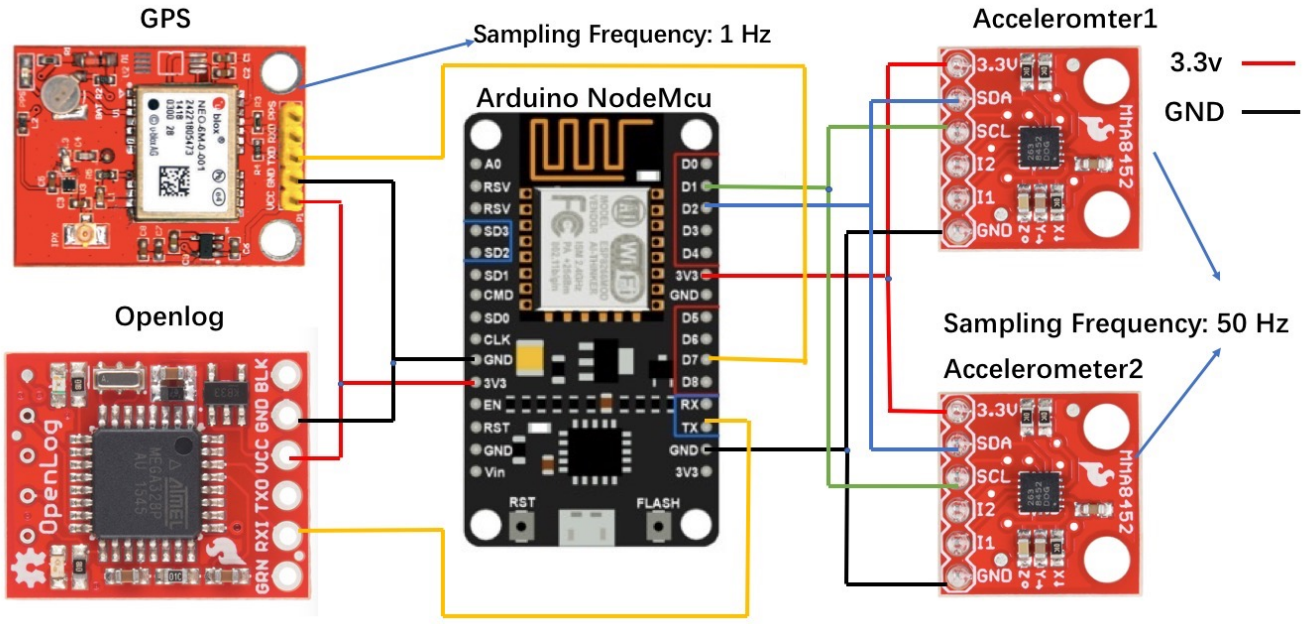

Fig.3 The Adruino Nodemcu connects multiple accelerometers through I^2C interface. Also, the Global Positioning System (GPS) and Openlog are connected to the Arduino Nodemcu through an RS232 Serial cable.

III. DATA TRANSMISSION

As mentioned above, two different microprocessors (Arduino Nodemcu and Particle Photon) are used in this project to transmit the data to Cloud or phones. This part focuses on technical differences in the way of data transmission between these two microprocessors. The Arduino Nodemcu will directly transmit all collected data (longitude, latitude, and accelerations from two accelerometers) to Firebase database through 'hotspots' or other wireless networks. Instead of directly transmitting data to Firebase database, Particle Photon will transmit data to smart phones. Then, the phone will locally store the data and transmit the information to the Firebase database, simultaneously. However, both of these methods are limited by their transmission rates. The Particle Photon's the transmission rate to the phone is less than 20Hz based on the ability of phones and Particle Photon. If this rate is too high, the application will crash after five to seven minutes of operation. Also, the transmission rate for Nodemcu to Firebase is rather slow even by comparison to Particle Photon and this causes problems for overall performance, however is useful for its direct communication with the Firebase platform. In general, the speed of the bike trail is 10 miles/h (4.44m/s), and if the sampling frequency is 20Hz, the range of the bump that could be detected can be approximately 22.2 cm. In the case that the range of the bump is less than 22.2 cm, it will not be detected. Accordingly, the value of accelerometers must first be filtered in the microcontroller to account for this. Specifically, in this project, only the maximum and minimum value of accelerometers are expected to identify the bumps. The frequency of the accelerometer is set up at 50Hz, and the transmission delay is set up at 100ms on road tests. This means that in a sampling period 5 samples, the data will only be transmitted one time. After implementing this method, the range of the bump that could be detected narrows. The preliminary test in the following section used 10 pipes whose diameter is a quarter of feet to test the accuracy of this method.

A. Arduino Nodemcu

Firstly, the Arduino Nodemcu integrates a WiFi transmitter and are based on Arduino IDE to program. Thus, in this project, the Arduino NodeMcu uses "hotspots" or other wireless networks to transmit data over the internet. To capture real-time data with Arduino NodeMcu, the FirebaseArduino and ArduinoJson Firebase libraries should be added to program. The request of push data to the Firebase database needs a specific project domain and Firebase password. The database is structured with Json and the JsonBuffer in the Arduino Nodemcu is called to generate the Json object for data storage. The core code about pushing data to Firebase using Arduino NodeMcu is demonstrated in the following program, as shown in Fig.4.

```
#include <Firebase.h>
#include <FirebaseArduino.h>
#include <ArduinoJson.h>
#define FIREBASE_HOST
The Firebase database domain
#define FIREBASE_AUTH
The Secrets for Firebase databse
Firebase.begin(FIREBASE_HOST,
FIREBASE_AUTH);
StaticJsonBuffer<100> jsonBuffer;
JsonObject& obj=jsonBuffer.createObject();
obj["latitude"]=gps.location.lat();
obj["longitude"]=gps.location.lng();
obj["Acc1_x"]=Acc1_x;
obj["Acc1_y"]=Acc1_y;
obj["Acc1_z"]=Acc1_z;
obj["Acc2_x"]=Acc2_x;
obj["Acc2_y"]=Acc2_y;
obj["Acc2_z"]=Acc2_z;
String name=
Firebase.push("Arduinopath2/location",obj);
```

Fig.4 The core code about pushing data to Firebase using Arduino NodeMcu is demonstrated in the following program.

Finally, the program must use the Firebase Push method from the Firebase library to push the data on the Json node with a specific path.

B. Particle Photon

The Particle Photon is also a WiFi-supported microprocessor and more conveniently the Particle Photon can be programmed on online IDE. Similar to Arduino Nodemcu, the Particle Photon also uses the 'hotspots' or other wireless networks to transmit the data. Technically, the implementation of wireless communication must call the unique protocol. Specifically, the TCP/IP protocol is utilized in this case to send data out to smart phones. Additionally, implementation of the TCP/IP protocol should assign the server or client to each device (the Particle Photon and the smart phone). In this case, the smart phone is used as a server and Particle Photon serves as a client. Now that the smart phone is the server, the client (Particle Photon) should match to the IP address and local port of the phone to realize the wireless communication. The core program of implementation of TCP/IP protocol in the Particle Photon is demonstrated in the following program, as shown in Fig.5.

```
TCPClient client;
if (client.connect("192.168.43.1", 2222))
 {
  client.println(long1+","+lat1+","+
  acc1x + ","+ acc1y +","+ max_index1
  +","+acc2x+","+ acc2y+","+max_index2);
  delay(10);
  client.println(long1+","+lat1+","+
  acc1x +","+ acc1y +","+ min_index1
  +","+acc2x+","+ acc2y+","+min_index2);
 }
```

Fig.5 The core program of implementation of TCP/IP protocol in the Particle Photon is demonstrated in the pro-

gram above,where "192.168.43.1" is the IP address of phone,"2222" is the communicating port of phone.

The data from accelerometer and Global Positioning System (GPS) unit is constantly updated to the phone.

IV. DATA PROCESSING

For this experiment, the raw data is what the accelerometer detects at a certain point along with that point's longitude and latitude. As the purpose of this experiment is to identify potentially damaged infrastructure, the data processed from that location needs to be posted on Google Maps to indicate that there is a structural damage. Data processing plays a monumental role in the final visualization of bumps on the map.

Fig.6 The display of bumps on the mobile application (left); The preliminary test using artificial bumps (ten pipes with diameter 1/4 feet) (right)

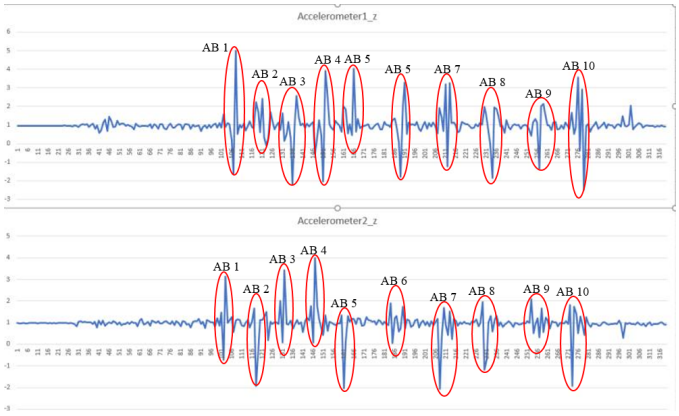

Fig.7 Original Data of Accelerometer 1 and Accelerometer 2 in Z-direction (Accelerometer 1 is on the front of bike, Accelerometer 2 is on the back of the bike) in the preliminary test. The red circle within which the magnitude of accelerations go up and down, in the figure, is considered as the bump.

When deciding on a method to process raw data, there were some factors and problems that needed to be considered. Generally, when a bike traverses through a hole or bump, the value of accelerometers will go up and go down or vice versa. Thus, in order to find a proper way to process the raw data, massive preliminary test are token. The preliminary test, as shown in Fig.6, used artificial bumps (ten pipes with diameter 1/4 feet) to simulate the trial damage. As shown in Fig.7, raw data of accelerations from two accelerometers, the red circle show identified bumps. Therefore, simply taking all points reaching a certain threshold as holes or bumps will potentially double or triple the results of data calculations and skew the conclusion. The traditional methods such as Fast Fourier Transformation and Moving Average, although popular in many fields, are not proper for this situation. For example, Fast Fourier Transform (FFT) is a kind of algorithm that can sample points and transform it from the time domain to its respective frequency domain.

However, Fast Fourier Transform (FFT) cannot perfectly fit in this situation. Although Fast Fourier Transform (FFT) is a good math tool for periodic signals, the signal in this research is non-periodic. Additionally, the application of Fast Fourier Transform (FFT) is limited because of the extensive calculation. Fast Fourier Transform (FFT) transforms accelerations of the accelerometer from time domain to frequency domain. Another limitation is that frequency domains do not include the direction of the acceleration, making it difficult to identify whether the change in frequency domain is caused by bumps or some other reasons (i.e. going up and down on the pavement).

Another widely used method is the Moving Average Algorithm, which is based on a logging previous data. It can filter out the "noise" from data and smooth out the fluctuation of other irrelevant data, which is very beneficial for data processing and particularly for distinguishing points of value. However, the Moving Average filters do not have any specific connection to time and instead it filters data with the highest frequency. The result of Moving Average is the average of several points and it cannot be used to identify the bumps and holes.

Therefore, combining the different aspects of these methods and creating a new algorithm is the best option for this experiment.

This method involves dividing the line chart into many different pieces within the same range or "window". Generally, during biking and data collecting, the bike averaged a rate of 10 miles per hour (almost 4m/s) and our frequency of collecting data was, as stated earlier, 20 Hz.

Therefore, the minimum interval of distance between bumps that could be detected was approximately 20 cm. The window range is set up as ten sampling times in Fig.8 to demonstrate this idea. In every window, if there exists a difference between the maximum value and minimum value that is larger than the threshold, there must be a point that should be considered as a hole or bump. But just using the raw accelerations in a window to identify the bump will sometimes cause inaccuracy. For example, in Fig.8, the difference between every adjacent point is hard to distinguish, thus, the value points have to be magnified by the formula. After making sure there exists a infrastructural problem in

this window, it is critical to find the exact point. Based on the concept of linear interpolation, this window interpolation method will use the location coordinates matching to the midpoint of X-axis between maximum acceleration and minimum acceleration in each window as the location coordinates of the bump. As shown in Fig.8, the lowest coordinate in window A is (106, -17.17) and the highest one is (108, 124.45). Therefore, the approximate location coordinates of the bump in this window should be equal to $(106 + 108)/2 = 107$.The Pseudo-code for the Window Interpolation Algorithm is represented in Algorithm 1.

Algorithm 1: Window Interpolation Algorithm

Data: $Acc1\_z$: raw data of accelerations in Z-direction
Lng: the latitude
Lat: the longitude
Result: $Bumps$: the latitude and longitude of bumps
(initialization):
$window\_size = 10$;
$window = 10$;
max: the first acceleration of each window;
min: the first acceleration of each window;
max: maximum acceleration value of each window
min: minimum acceleration value of each window
max_index: number of maximum value of each window
min_index: number of minimum value of each window
(finding the threshold:)
for $i \leftarrow 0; i \leq window;\ i \leftarrow i + 1$ **do**
  if $window < the\ size\ of\ Acc1\_z$ **then**
    if $Acc1\_z[i] \leq 1$ **then**
      $Acc1\_z[i]=(Acc1\_z[i] - 1)\hat{}3$;
    else
      $Acc1\_z[i]=Acc1\_z[i]\hat{}3$;
    if $Acc1\_z[i] \geq max$ **then**
      $max=Acc1\_z[i]$;
      $max\_index=i$;
    if $Acc1\_z[i] \leq min$ **then**
      $min=Acc1\_z[i]$;
      $min\_index=i$
    if $i = window\_size - 1$ **then**
      if $abs(max - min) \geq threshold$ **then**
        $Midpoint =$
        $ceil\ \{(max\_index + min\_index)/2\}$
        $Lat=Lat[Midpoint]$
        $Lng=Lng[Midpoint]$
      max=1
      min=1
      $window=window\_size$

There must be a consistent one-to-one match between every point on the X-axis, latitude and longitude for every location. The points that fit into the standard established by the Window Interpolation Algorithm will be displayed on the mobile application.

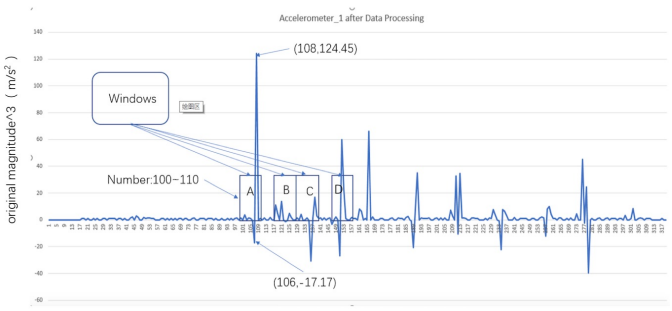

Fig.8 The implementation of Window Interpolation Algorithm to the value of Accelerometer 1

V. ANDROID MOBILE APPLICATION AND WEBSITE

Based on previous research that used accelerometers and Global Positioning System (GPS) unit inserted in smart phones to directly record vibration intensities, the accuracy of data is an issue. Thus, this experiment utilized an Android mobile application to receive the data from accelerometers and the Global Positioning System (GPS) Unit in order to achieve a high sampling frequency. Originally, the data received was displayed in the form of line charts (accelerations) in time domain. After the implementation of the data processing algorithm, the filtered bumps could be visually depicted on maps. This enables all users of the application to receive notifications in the form of a ring when approaching bumps if they have been previously detected.

A. User Interface

The user interface of this application mainly includes four parts: user authentication (as shown in Fig.9 left), the dashboard, the interpretation of received data, and the visualization of bumps on Google Map.

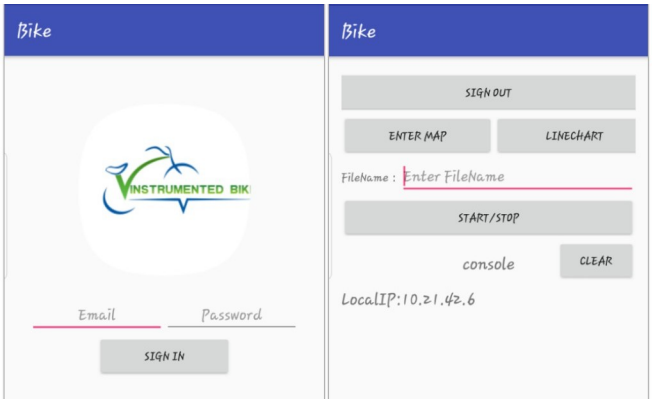

Fig.9 The user login interface (left); the dashboard interface to select the function of application (right).

In the mobile application, users are assigned to a specific user name and password to grant access to this application. Once the user has logged in this application, the page will jump to dashboard page which controls the main function

of this application including line chart, map, and the start and stop of data collection. The line chart activity serves to obtain the real-time data as well as historical information from previous tests. Representation of this activity on Google Map assists in visualization and assessment of road health and to better present gathered data.

B. Backstage Service

Technically, in this project, the TCP/IP protocol is used to realize the data transmission between smart phone and data loggers, and SPLITE database on phones as well as Cloud database are used to store data. However, the long-term operation of these activity will lead to the block of thread even crash of application during data collection. Therefore, the backstage service is essential to handle this time-consuming activity in order to achieve a better user experience avoiding any crashes and blocks so that the application may run smoothly. Once the start button in the dashboard is pressed, the backstage service will automatically operate to receive data, store data to the SPLITE database on the phone, and transmit data to the Cloud, all simultaneously.

C. Data synchronization to Cloud

The Google Cloud (Firebase real-time database) is used in this project as a real-time database to realize the bi-way synchronization of data (from phone to database, and database to phone). Firebase is used in this research on the grounds of its low delay (in micro seconds), low cost, and simplicity to implement. To start, the data is structured in the form of JSON trees in the database. So, for each test, users type in a specific file name on the phone to identify the test in the database (the file name is the parent in the JSON tree, and the data belonging to this parent is the child). This is easier for retrieving data and assists in avoiding the collision of data from different users operating the app at the same time. Once the new data is added to the database, all users can share it.

D. The implementation of Data processing algorithm on Google map

In order to accurately identify and represent bumps on the map, a getter and setter algorithm is required. The first aspect of this process is to retrieve the stored data from the Firebase database. The ValueEventListener to listen for new data and, once the data is changed, will return the snapshot of the data as a Map Object. The data processing algorithm in Section IV is embedded in the loop that retrieves data. By comparing the current location to the location of the bumps, the distance between the user's current location and the detected bump is calculated. Once this distance is less than 50 meters, a sound will notify users that may approach the bumping area. Assuming that the speed of bike is still 4.44m/s, the biker can get notification 11.26s before reaching the bump.

E. Website

The main purpose of the website is to compensate for flaws in the mobile application. Since the mobile application is based on the Android OS, users from other systems (IOS, Windows) cannot access this application. The website, however, can be used by any OS system. Resembling the mobile application, the line chart, Google Maps, and data processing algorithm are all inserted into the website. It enables the same functions to be used just on a different platform.

VI. CONCLUSIONS

Conclusively, Instrumented Bikes are a feasible assessment of road conditions by translating raw data to visual information. The two systems (Particle Photon and Arduino Nocemcu) mentioned in this paper and the utilization of the Internet of Things with an Android application and website all serve to realize the information flow from device to device and assess road conditions.

REFERENCES

[1] Ho, Chun-Hsing, Slim, Gerjen, Monahan, Shannon, DeGeyter, Jeremy "Application of Geographic Information Systems and Vibration Mobile Apps in Road Condition Assessment of Bike Trails" Transportation Research Board 95th Annual Meeting 2016, Washington DC, NW, 16-1424.

[2] E. Theodoridis, G. Mylonas and I. Chatzigiannakis, "Developing an IoT Smart City framework," IISA 2013, Piraeus, 2013, pp. 1-6.

[3] S.-H. Yang, Internet of things, in Wireless Sensor Networks. Springer, 2014, pp. 247261.

[4] R. Ferrero, E. Beattie and J. Phoenix, "Sensor city- A global innovation hub for sensor technology," in IEEE Instrumentation and Measurement Magazine, vol. 21, no. 1, pp. 4-16, February 2018.

[5] P. P. Shah, A. A. Patil and S. S. Ingleshwar, "IoT based smart water tank with Android application," 2017 International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC), Palladam, 2017, pp. 600-603.

[6] C. T. Chiang, "Design of a CMOS MEMS Accelerometer Used in IoT Devices for Seismic Detection," in IEEE Journal on Emerging and Selected Topics in Circuits and Systems. 2018, pp 1-1

[7] Alsalemi et al., "Real-Time Communication Network Using Firebase Cloud IoT Platform for ECMO Simulation," 2017 IEEE International Conference on Internet of Things (iThings) and IEEE Green Computing and Communications (GreenCom) and IEEE Cyber, Physical and Social Computing (CPSCom) and IEEE Smart Data (SmartData), Exeter, 2017, pp. 178-182.

[8] T. Yamamoto, T. Hara, T. Ishikawa, H. Oyama, H. Takada and T. Azumi, "TINET+TECS: Component-Based TCP/IP Protocol Stack for Embedded Systems," 2017 IEEE Trustcom/BigDataSE/ICESS, Sydney, NSW, 2017, pp. 784-791

[9] R. K. Banyal, V. K. Jain and P. Jain, "Data Management System to Improve Security and Availability in Cloud Storage," 2015 International Conference on Computational Intelligence and Networks, Bhubaneshwar, 2015, pp. 124-129.

[10] A. Yonemoto, T. Hisakado and K. Okumura, "Accuracy improvement of the FFT-based numerical inversion of Laplace transforms," in IEEE Proceedings - Circuits, Devices and Systems, vol. 150, no. 5, pp. 399-404-, 6 Oct. 2003.

[11] D. Wu, L. Zhang and L. Lin, "Based on the Moving Average and Target Motion Information for Detection of Weak Small Target," 2018 International Conference on Intelligent Transportation, Big Data and Smart City (ICITBS), Xiamen, China, 2018, pp. 641-644.

[12] J. P. Hermias, K. Teknomo and J. C. N. Monje, "Short-term stochastic load forecasting using autoregressive integrated moving average models and Hidden Markov Model," 2017 International Conference on Information and Communication Technologies (ICICT), Karachi, Pakistan, 2017, pp. 131-137.

[13] G. Mangeni, R. H. G. Tan, T. H. Tan, S. K. Cheo, V. H. Mok and J. Y. Pang, "Photovoltaic module cell temperature measurements using linear interpolation technique," 2017 IEEE International Instrumentation and Measurement Technology Conference (I2MTC), Turin, 2017, pp. 1-6.