Available *online* at : http://jnte.ft.unand.ac.id/

Jurnal Nasional Teknik Elektro

| ISSN (Print) 2302-2949 | ISSN (Online) 2407-7267 |

Real-Time Accident Detection Using KNN Algorithm to Support IoT-based Smart City

Khodijah Amiroh, Bernadus Anggo Seno Aji, Farah Zakiyah Rahmanti

Department of Information Technology, Faculty of Information Technology and Business, Institut Teknologi Telkom Surabaya, Indonesia

ARTICLE INFORMATION

Received: January 30, 2022
Revised: February 08, 2022
Available online: March 29, 2022

KEYWORDS

KNN, Gyroscope, Accelerometer, ReactJs, Google Maps APIs

CORRESPONDENCE

Phone: +62-811-37-88188

E-mail: dijaamirah@ittelkom-sby.ac.id

A B S T R A C T

Surabaya is a city with an area of 326.81 km2 and is the center of land transportation in the eastern part of Java Island. The construction of digital infrastructure in the Surabaya area will make it easier for the City Government to make efficient services. Traffic accidents that occurred in Surabaya until 2017 recorded 1,365 incidents. EVAN (Emergency Vehicle Automatic Notification) is a research topic that focuses on the field of transportation, especially in real-time traffic accidents which can be integrated with city information centers and hospitals for primary assistance in accidents. The purpose of this research is to make it easier for the Surabaya city government to provide first aid in the event of an accident. The design of the device on the user side is made using the Arduino, the accelerometer sensor and the gyroscope in the form of the MPU6050 sensor and the u-blox gps module. Crash detection on the system using the k-Nearest neighbors algorithm (KNN). On the operator side, the design is done on a web basis by utilizing the ReactJs framework which is integrated with the Google Maps APIs. The results of the accuracy of the accident detection system reached 97% and the detection of accident locations and the nearest hospital from the location reached 100%. Thus, real-time accident detection can be implemented in Surabaya city to support the smart city.

INTRODUCTION

Surabaya is a city with about 326.81 km2, with more than 3 million inhabitants in 2018 [1]. The large area and population of the Surabaya area, making Surabaya a land transportation center in the eastern part of the island of Java. The construction of digital infrastructure in the Surabaya area will make it easier for the City Government (Pemkot) to make efficient services. The integrated service in Surabaya is one of the advantages of the city government to respond quickly to disturbances that indicate an emergency from the community through the standby post provided by the city government by providing the Surabaya Command Center centered on the ex Siola Building. The integrated service Call Center 112 can be used by the public to inform the current situation in every incident, including accidents [2][3]. Traffic accidents that occurred in Surabaya until 2017 recorded 1,365 incidents [4]. At the same time, most vehicles involved in traffic accidents are motorcycles. Based on the data above, it is known that the automatic information support system between accident events and the Surabaya command center can be used as a form of research in the transportation sector and support the Smart City built by the Surabaya City Government.

Several studies have been carried out to detect accidents, as has been done by [5][6][7][8][9]. Motor vehicle accident detection is carried out using Arduino and Android smartphones. This study describes the vehicle's condition that had an accident that was detected using an Android smartphone. The application of the KNN algorithm has also been carried out by [10] to classify the severity of traffic accident victims in the Central Java Regency with an accuracy result of 88.82%. In addition, accident detection research has also been done with the development of a mobile-based traffic accident tracing application (fast help) [11] in this study, which was made to send accident location points to an Android-based hospital. The development of IoT for safety in driving on two-wheeled motorcycles has also been carried out by [12]; the study used the ESP8266 IoT module as a sender and receiver of instructions.

EVAN (Emergency Vehicle Automatic Notification) is a research topic that focuses on the transportation sector, especially on traffic accidents. The purpose of this research is to easy the government of the city of surabaya to do first aid if accident happen so that this research can be applied to support smart city. In the event of an accident, the Surabaya city government can validate the incident if there is a notification whether it requires help from the hospital or not. If an accident requires first aid at the hospital, then the system has automatically detected the location of the nearest hospital to get first aid. Aspects needed in this study include the user and operator sides. On the user side, the design tool is used to identify accident events, the location of the incident, the vehicle number, and the time of the incident. Meanwhile, on the operator side, the design is carried out to display notification of accident events from the user's side and display the location of the nearest hospital from the incident so that the operator can contact in real-time as the first treatment for accidents.

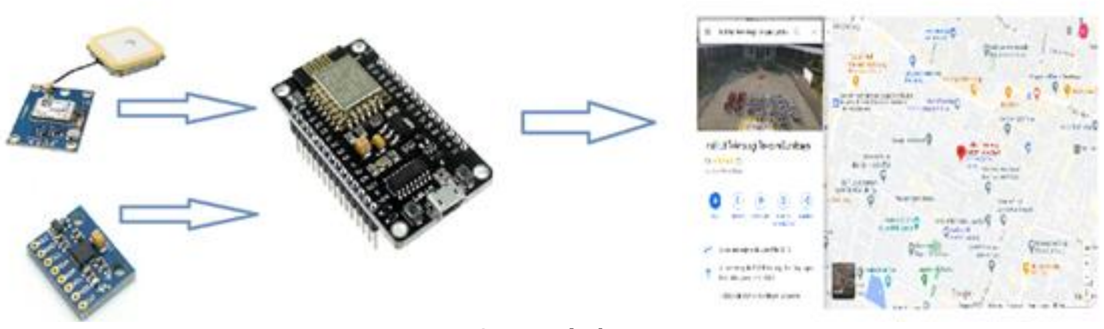
Figure 1. System Block Diagram

The tool's design on the user side is made using Arduino, gyroscope, and accelerometer. Arduino is a microcontroller that functions as the control center of the system. A gyro sensor is helpful for determining the motion orientation by resting on an axis with fast rotation. An accelerometer is a sensor used to detect changes in vehicle acceleration to detect if an accident occurs. Detection of accidents in the system using the k-Nearest neighbor's algorithm (KNN). The algorithm is used to clarify an object based on the data grouping, which is the closest distance from the object [13].

On the operator side, the design is web-based. A notification will appear on the web in the event of an accident by displaying the location, type of vehicle, police number, and the location of the nearest hospital. So the operator can directly contact the nearest hospital from the accident site.

METHOD

The detailed workings of the system are shown in Figure 3. The first time the system will read the sensors on the vehicle. In this study, the sensor used is the MPU-6050 sensor.

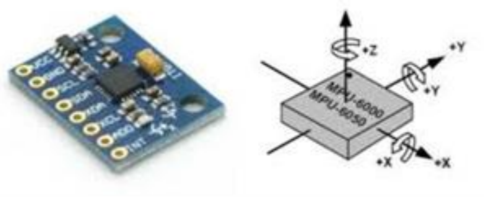

Figure 2. Sensor MPU 6050

The MPU6050 module shown in Figure 2 is a device that has six axes of motion tracking. The MPU6050 module is a module with three axes gyroscope and three axes accelerometer. With this combination, it is hoped that motion detection can be more accurate because adjusting the cross-axis between the accelerometer and gyroscope can be neglected [14][15].

The sensor then sends data to the microcontroller, which is done every second. The microcontroller used is Arduino NodeMCU V3 Lolin Lua Wi-Fi ESP8266. Arduino is one of the microcontrollers that are pretty easy to use today. The ESP8266 module is built by reducing the number of components and shielding required to perform a particular task. Arduino configures the MCU with the Arduino Uno and SAM Core board managers. The term Core is used for software units required to construct the Arduino C++ header using the MCU language.

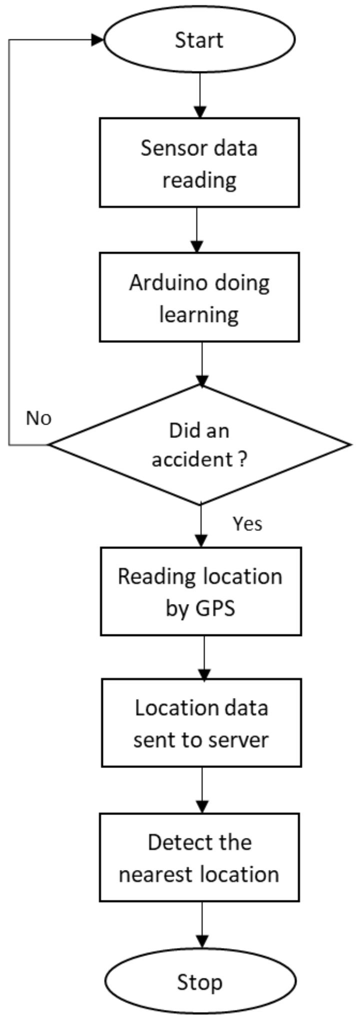

Figure 3. Flowchart System

Figure 2. NodeMCU Arduino ESP8266

In Figure 2 is the Arduino Node MCU ESP8266 module, which can function as a Wi-Fi Module for 802.11n and 802.11b networks. This allows the Wi-Fi module to function as an Access Point AP, Wi-Fi station, or perform both functions simultaneously [16].

When sending data, the device uses I2C communication so that it can send and receive data continuously. I2C is a two-way serial communication standard that uses two different channels. In the I2C system, several controllers are in charge of carrying information, namely the SCL (Serial Clock) and SDA (Serial Data) channels. If the hardware is connected to the I2C Bus system, the system can be operated as a Master or as a Slave. On the I2C bus, the master itself transfers data by forming a start signal and ending the data transfer by forming a stop signal, while the slave is a device addressed by the master [17]. A masterwork is by waiting for a start signal to send a signal to start all commands. This is defined by changing the Serial Data voltage from “1” to “0” along with the Serial Clock “1”. To end all commands, the stop signal is defined by changing the serial data voltage from “0” to “1” along with the Serial Clock “1”.

After reading the sensor, each incoming data will be compared with the existing training data using the KNN method. Training data is created using real data according to sensor readings before implementing the system. The kNN algorithm is an algorithm used to classify an object. Classification is done based on some training data as much as *k*, which is closer than an object. The distance data is classified using the cosine similarity equation [18]. The equation is used to test the size, which is the interpretation of the proximity of the distance based on the similarity of the data. Equation (1) below is an equation used to calculate a distance in the KNN algorithm using the Cosine Similarity method:

$$cos(\theta_{QD}) = \frac{\sum_{i=1}^{n} Q_i D_i}{\sqrt{\sum_{i=1}^{n}(Q_i)^2} . \sqrt{\sum_{i=1}^{n}(D_i)^2}} \qquad (1)$$

Where :

$cos(\theta_{QD})$ = Q’s similarity to D

Q = Test Data

D = Training Data

n = Number of Data

Then, if the data analysis detects an accident, the microcontroller will read the location of the accident using the U-Blox GPS Module. The U-Blox GPS module, as shown in Figure 3, functions as a stand-alone GPS signal receiver with reasonably good performance. In the GPS module, one of the most critical parameters is the Time to First Fix (TIFF). TIFF is defined as the speed at which the GPS module can access the address and empirical data from the satellite [19]. The output of this GPS module is the Longitude and Latitude of a location to detect accidents.

Figure 3. U-Blox GPS Module

The location data that is read by the microcontroller is then sent via the Web using the WiFi Module on the microcontroller. The WiFi connection will use the user's cell phone to continue to be connected to the Surabaya city information center. On the Web, the framework is designed using ReactJs. Frameworks on the web are made using ReactJs. ReactJs is one of the most popular frameworks. The use of this framework has advantages, including complete documentation and ease to use to make us develop various web application products [20]. Facebook developed React, a front-end library that can be used to create mobile applications and create views on the web.

RESULTS AND DISCUSSION

Data testing is carried out in several stages. The first stage is testing from the user side, which consists of the IMU MPU6050 sensor, Arduino ESP8266, and the U-Blox GPS module by applying the KNN Algorithm. In learning using the KNN algorithm, 100 training data were prepared by testing several classifications. Classification carried out as many as 3, 5, and 7 classifications. Programming is done using Arduino IDE with I2C communication for reading the MPU6050 sensor. To read the location in the form of longitude and latitude, programming is done using the TinyGPS++.h library. They were sending location data from Arduino to the server using WiFi available on Arduino by adding the ESP822WiFi.h library available on Arduino. The second testing stage is carried out on the operator side consisting of a web system initially using static data. The last stage is to integrate the entire system both from the user and the operator sides.

Test Data Training

Training data has been collected in several conditions such as normal roads, potholed roads, incline roads, and when an accident occurs. The training data results are then classified into the standard category, and an accident occurs. The table below shows some of the training data carried out for classification.

Table 1. Training Data for Normal Road Conditions

No	Ax	Ay	Az	Gx	Gy	Gz
1	1,01	0,03	0,18	24,47	1,77	0,82
2	1,01	0,03	0,18	24,16	1,87	0,91
3	0,96	0,07	0,06	0,47	1,39	-9,49
4	0,99	0,24	0,15	14,09	7,54	2,7
5	1,16	-0,03	0,26	158,56	19,08	10,74
6	0,91	0,19	0,11	72,14	12,51	15,75
7	0,88	-0,12	0,13	126,84	2,32	24,74
8	0,79	-0,06	0,01	44,55	3,08	-6,65
9	0,93	-0,07	0,07	25,27	-1,87	-3,78
10	0,76	0,19	0,06	30,38	-4,27	31,66

Table 2. Training Data for Hollow Road Conditions

No	Ax	Ay	Az	Gx	Gy	Gz
1	-0,21	-0,17	-0,41	49,31	26,92	-26,24
2	0,99	0,1	0,14	19,27	2,5	-10,57
3	0,97	-0,17	0,18	26,49	3,65	-14,53
4	0,88	0,07	0,23	39,21	-3,37	-24,08
5	1,38	-0,1	0,11	18,99	-15,92	-17,48
6	0,88	-0,07	0,15	33,53	-25	-21,76
7	1,17	-0,11	0,64	-0,57	-3,65	-24,15
8	1,12	-0,07	1,46	-109,85	-13,94	-33,53
9	-0,21	-0,17	-0,41	49,31	26,92	-26,24
10	0,99	0,1	0,14	19,27	2,5	-10,57

Table 3. Training Data for Uphill Road Conditions

No	Ax	Ay	Az	Gx	Gy	Gz
1	1,15	0,05	0,17	19,68	23,07	-6,18
2	0,92	0,07	-0,01	19,78	23,78	26,15
3	0,44	0,07	-0,05	19,59	22,57	15,19
4	1,02	0,05	0,22	19,68	25,43	6,24
5	1,03	0,07	0,24	19,68	23,31	4,63
6	1,03	-0,01	0,19	19,68	23,32	7,24
7	0,90	0,04	0	19,59	25,3	2,52
8	0,93	0	0,08	19,68	23,94	-1,12
9	1,18	0,21	0,51	19,68	17,44	0,83
10	0,92	0,1	0,33	19,68	25,36	-1,63

Table 4. Accident Condition Training Data

No	Ax	Ay	Az	Gx	Gy	Gz
1	-0,03	-0,37	-0,26	20,34	159,98	112,44
2	-0,61	0,58	0,63	20,44	72,05	-238,37
3	-0,02	0,96	0,05	20,44	26,69	-1,02
4	-0,84	1,72	1,25	20,53	34,02	113,38
5	-0,31	-0,18	-0,24	20,53	-22,08	-172,12
6	-1,68	-2	0,34	20,53	66,81	-142,06
7	-0,25	-0,92	-0,38	20,62	29,15	-89,85
8	-0,38	0,64	0,63	20,62	-12,93	37,45
9	-0,22	-0,3	0,24	20,62	-250,14	250,13
10	-0,17	0,57	0,38	20,62	39,89	-77,4

Table 1 shows several samples of training data under normal road conditions. From these results, it has the characteristic that under normal conditions, the vehicle will be directly proportional to the acceleration on the stable x-axis. Table 2 is an example of training data on potholed road conditions. The data collection results with potholes indicate that the orientation at an angle on the y-axis and z-axis is unstable due to shocks. In Table 3 are training data with incline road conditions. The data collection results on uphill and downhill roads do not have results that are much different from normal roads, so it can be concluded that the condition of the uphill road does not affect anything. The last data collection is accident conditions. The resulting training data is as in Table 4. The results from the table show that both in terms of acceleration and rotation changes, there are very significant changes so that we can classify them in a data grouping.

After determining the training data on the system using the KNN Algorithm, the data from the sensor is then classified into several groups with several groups. In this study, the data were grouped into 3,5, and 7 classifications. The results of the conclusions from learning if there is an accident, the system will continue to collect accident location data.

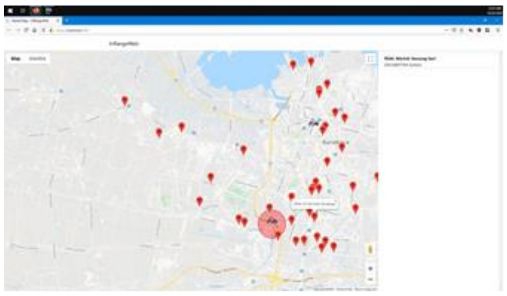

Figure 4. Hospital Location in Surabaya

The results of location detection with the U-Blox GPS module are longitude and latitude. Data from the GPS module is sent to the server using Wi-Fi available on the Arduino ESP8266. The server on the web system then receives the crash location data and detects the crash location point. On the operator side, a database on the web has been prepared for all hospital locations in the Surabaya area, as shown in Figure 4.

The operator is in charge of waiting for notification from the user if something unexpected happens, namely an accident tragedy. After receiving the notification, the operator will verify the incident. Currently in Surabaya already has an integrated monitoring system along the main road. If an accident occurs, the operator is in charge of contacting the nearest hospital according to the list of hospitals that appear on the web.

Testing on Users

On the user side, the test data is taken with several conditions. Conditions with normal roads, potholes, incline roads, and at the time of the accident. Several classifications are performed, and the results are displayed simultaneously. The test results when running normally can be seen in Table 5.

Table 5. Test During Normal Road

No	*k*=3	*k*=5	*k*=7
1	“Safe”	“Safe”	“Safe”
2	“Safe”	“Safe”	“Safe”
3	“Safe”	“Safe”	“Safe”
4	“Safe”	“Safe”	“Safe”
5	“Safe”	“Safe”	“Safe”
6	“Safe”	“Safe”	“Safe”
7	“Safe”	“Safe”	“Safe”
8	“Safe”	“Safe”	“Safe”
9	“Safe”	“Safe”	“Safe”
10	“Safe”	“Safe”	“Safe”

Based on the data above, it is known that the test results are 100% correct and have no detection errors both from the accelerometer and gyroscope sides. The overall classification shows no significant difference. After the test was carried out when the road was expected, the test was carried out when the road was potholed with the results as shown in Table 6.

Table 6. Testing During Potholes

No	*k*=3	*k*=5	*k*=7
1	“Safe”	“Safe”	“Safe”
2	“Safe”	“Safe”	“Safe”
3	“Safe”	“Safe”	“Safe”
4	“Safe”	“Safe”	“Safe”
5	“Safe”	“Safe”	Accident
6	Accident	Accident	Accident
7	“Safe”	“Safe”	Accident
8	“Safe”	“Safe”	“Safe”
9	“Safe”	“Safe”	“Safe”
10	“Safe”	“Safe”	“Safe”

Based on the test results when the road is potholed, the training data prepared to conclude show different results when classified with several types. This shows that the more classifications performed, the more accurate the results. Based on these results, it is caused by a curvy road that causes a change in acceleration and a road with a reasonably hard shock that causes a change in the axis of rotation. The output data above is still categorized as "Safe" because the accident data only appears once in 10 seconds. Based on these results, of the 50 data recorded when testing potholes, the test's success was 98% at k=3 and k=5. The test results at k=7 the success of the test is 90%.

After testing with potholes, testing is also carried out when the vehicle is running on an uphill or downhill area, as shown in Table 7. The results of the test do not appear to experience data errors, so it can be concluded that the system on the user's side when the uphill/downhill road experiences successful data testing by 100%.

Table 7. Ascent/Descent Road Test

No	*k*=3	*k*=5	*k*=7
1	“Safe”	“Safe”	“Safe”
2	“Safe”	“Safe”	“Safe”
3	“Safe”	“Safe”	“Safe”
4	“Safe”	“Safe”	“Safe”
5	“Safe”	“Safe”	“Safe”
6	“Safe”	“Safe”	“Safe”
7	“Safe”	“Safe”	“Safe”
8	“Safe”	“Safe”	“Safe”
9	“Safe”	“Safe”	“Safe”
10	“Safe”	“Safe”	“Safe”

The last data test is on the vehicle's condition after having an accident.

Tabel 8 shows that all test results with several types of classification can read the same results so that the success of testing in accident conditions with the KNN algorithm using k=3, k=5, and k=7 reaches 100%. If someone has an accident, they will experience changes in acceleration and rotation, which are pretty significant, making it easier to carry out early detection in real-time.

Table 8. Accident Test

No	*k*=3	*k*=5	*k*=7
1	Accident	Accident	Accident
2	Accident	Accident	Accident
3	Accident	Accident	Accident
4	Accident	Accident	Accident
5	Accident	Accident	Accident
6	Accident	Accident	Accident
7	Accident	Accident	Accident
8	Accident	Accident	Accident
9	Accident	Accident	Accident
10	Accident	Accident	Accident

After testing the sensor data on the user side, testing the data is also carried out on the U-Blox GPS Module to read the location. The output results on the GPS module are then compared with the location data found on google maps. The test results are shown in Table 9.

Table 9. GPS Module Testing

No	Long-lat on modul	Long-lat on maps	Conclusion
1	-7.3137025, 112.719183	-7.3138025, 112.718191	Match
2	-7.3112662, 112.7276945	-7.3112662, 112.7266958	Match
3	-7.251792, 112.7425478	-7.251792, 112.7424497	Match
4	-7.300945, 112.7344284	-7.300945, 112.7343073	Match

The last test on the user side is a WiFi connection test on the ESP8266 module and displays data on a web page. Testing is done by making the IP address for the internet connection. The results show that the WiFi module on the ESP8266 can run properly.

Testing on Operators

Testing on the operator side initially uses static data, as shown in Figure 5. After successfully showing the location of the nearest hospital based on a specific range, it shows that the web is ready for use. Once ready for use, operator testing has been completed.

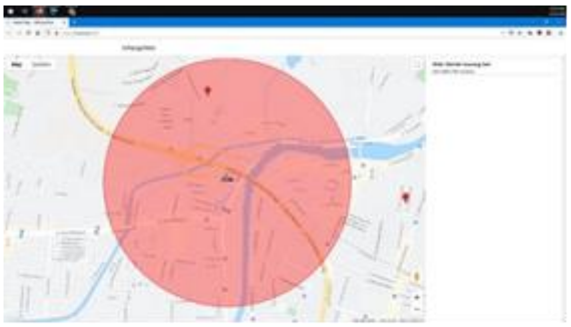

Figure 5. Web testing on Operators

Overall System Test

The overall test was carried out in the undaan wetan area, as shown in Figure 6. The overall test results produced an output in two hospitals in the undan area that could be used as a reference for victims in the first treatment in an accident. The operator then contacted one of the hospitals to handle the victim to minimize unexpected incidents.

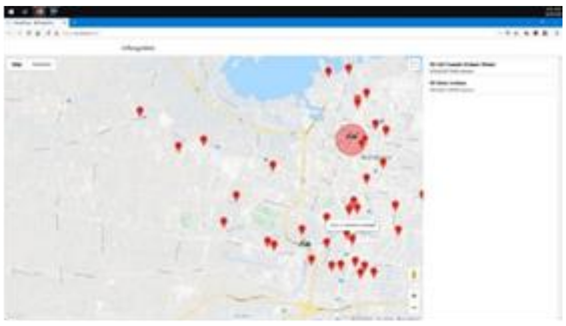

Figure 6. Result Overall System Test

CONCLUSIONS

Based on the results of testing and analysis carried out by the entire system, it can be concluded that it can be used optimally on the user side and the operator side. In addition, data analysis using the KNN algorithm can be used as a conclusion on the system when an accident occurs and does not reach 97% accuracy. Thus, real-time accident detection can be implemented in Surabaya city to support the smart city.

ACKNOWLEDGMENT

All academicians of Institut Teknologi Telkom Surabaya. This research was funded by the Directorate General of Higher Education, DIKTI and Telkom Institute of Technology Surabaya.

REFERENCES

[1] Ulfah, N. (2019). Analisis Spasial dan Temporal terhadap Data Statistik Kependudukan Kota Surabaya Menggunakan Atlas Statistik dan Animasi Berbasis Waktu. Jurnal Teknik ITS, A84-A89.

[2] W. D. Astuti, A. Sukristyanto, and A. Susiantoro, “KUALITAS PELAYANAN PERIZINAN DI UNIT PELAYANAN TERPADU SATU ATAP (UPTSA) KOTA SURABAYA (STUDI TENTANG SURAT IZIN USAHA PERDAGANGAN).”

[3] R. Apriastini, R. I. Mahbubah, M. Zainul Arifin, and A. Wicaksono, “ANALISIS PREDIKSI KECELAKAAN PENGGUNA SEPEDA DI KOTA SURABAYA, JAWA TIMUR,” 2018.

[4] A. Karakteristik *et al.*, “Knowledge Discovery in Database Analysis of Traffic Accident Characteristic on Ahmad Yani Road Surabaya through Knowledge Discovery in Database Approach,” 2018.

[5] M. T. A. Amir and Y. Y. Kerlooza, “Sistem Pendeteksi Kecelakaan Kendaraan Bermotor Menggunakan Arduino Dan Smartphone Android,” *Telekontran : Jurnal Ilmiah Telekomunikasi, Kendali dan Elektronika Terapan*, vol. 8, no. 2, pp. 105–112, Apr. 2021, doi: 10.34010/telekontran.v8i2.4570.

[6] Machine Intelligence Research Labs, Research Groups in Intelligent Machines, IEEE-Tunisia Section, M. IEEE Systems, M. IEEE Systems, and Institute of Electrical and Electronics Engineers, *6th International Conference on Soft Computing and Pattern Recognition : SoCPaR 2014 : August 11-14, 2014 : Ramada Plaza hotel, Tunis - Tunisia : Conference Booklet*.

[7] Y. W. Liyanage, D.-S. Zois, and C. Chelmis, “Near Real-Time Freeway Accident Detection,” *IEEE Transactions on Intelligent Transportation Systems*, vol. 23, no. 2, pp. 1467–1478, Oct. 2020, doi: 10.1109/tits.2020.3027494.

[8] A. Khan, F. Bibi, M. Dilshad, S. Ahmed, Z. Ullah, and H. Ali, “Accident Detection and Smart Rescue System using Android Smartphone with Real-Time Location Tracking,” 2018. [Online]. Available: www.ijacsa.thesai.org

[9] *2020 IEEE PES/IAS PowerAfrica*. IEEE, 2020.

[10] D. S. Wisdayani, I. Manfaati Nur, and R. Wasono, “Penerapan Algoritma K-Nearest Neighbor dalam Klasifikasi Tingkat Keparahan Korban Kecelakaan Lalu Lintas di Kabupaten Jawa Tengah Application of K-Nearest Neihbor Algorithm in the Classification of Severity of Traffic Victims in Pati, Central Java.” [Online]. Available: http://prosiding.unimus.ac.id

[11] Y. Sonatha, E. Asri, I. Rahmayuni, and Y. F. Wibawa, “Pembangunan Aplikasi Tracking Kecelakaan Lalu Lintas (Fast Help) Berbasis Mobile,” *INVOTEK: Jurnal Inovasi Vokasional dan Teknologi*, vol. 19, no. 2, pp. 1–8, Oct. 2019, doi: 10.24036/invotek.v19i2.441.

[12] M. Itaqilah and B. Budi Rijadi, “PENGEMBANGAN INTERNET OF THINGS UNTUK APLIKASI KEAMANAN BERKENDARA PADA KENDARAAN BERMOTOR RODA DUA.”

[13] N. Luh Gede Pivin Suwirmayanti Program Studi Sistem Komputer STMIK STIKOM Bali Jl Raya Puputan, “Penerapan Metode K-Nearest Neighbor Untuk Sistem Rekomendasi Pemilihan Mobil Implementation of K-Nearest Neighbor Method for Car Selection Recommendation System.”

[14] “Position Tracking during Human Motions using an Integrated Wearable Sensing System 2016 Giulio Zizzo.”

[15] A. Mahfuzhon and G. Edhi Setyawan, “Rancang Bangun Alat Pendeteksi Kecelakaan Mobil Menggunakan Sensor Akselerometer dan Sensor 801s Vibration,” 2018. [Online]. Available: http://j-ptiik.ub.ac.id

[16] D. Abdulahad Aziz, “Webserver Based Smart Monitoring System Using ESP8266 Node MCU Module,” *International Journal of Scientific & Engineering Research*, vol. 9, no. 6, 2018, [Online]. Available: http://www.ijser.org

[17] N. Ferdiansyah Kusna, S. Rizqika Akbar, and D. Syauqy, “Rancang Bangun Pengenalan Modul Sensor Dengan Konfigurasi Otomatis Berbasis Komunikasi I2C,” 2018. [Online]. Available: http://j-ptiik.ub.ac.id

[18] M. Rivki and A. M. Bachtiar, “IMPLEMENTASI ALGORITMA K-NEAREST NEIGHBOR DALAM PENGKLASIFIKASIAN FOLLOWER TWITTER YANG MENGGUNAKAN BAHASA INDONESIA,” *Jurnal Sistem Informasi*, vol. 13, no. 1, p. 31, May 2017, doi: 10.21609/jsi.v13i1.500.

[19] J. Teknik Elektro, P. Negeri Padang Jurusan Teknik Elektro Politeknik Negeri Padang, J. Limau, and K. Kunci, “Komparasi Akurasi Global Posistion System (GPS) Receiver U-blox Neo-6M dan U-blox Neo-M8N pada Navigasi Quadcopter,” *Elektron Jurnal Ilmiah*, vol. 12, 2020.

[20] M. Wali and L. Ahmad, “Perancangan Access Open Journal System (AOJS) dengan menggunakan Framework Codeigniter dan ReactJs,” *Jurnal JTIK (Jurnal Teknologi Informasi dan Komunikasi)*, vol. 2, no. 1, p. 48, Oct. 2018, doi: 10.35870/jtik.v2i1.53.