Smart Crash Detector untuk Sepeda Motor Berbasis IOT

Tugas Akhir

diajukan untuk memenuhi salah satu syarat memperoleh

gelar sarjana

dari Program Studi S1 Teknik Informatika

Fakultas Informatika

Universitas Telkom

1103130153

FAKHRI AKBAR PRATAMA

Program Studi Sarjana S1 Teknik Informatika

Fakultas Informatika

Universitas Telkom

Bandung

2019

LEMBAR PENGESAHAN

***Smart Crash Detector* untuk Sepeda Motor Berbasis IOT**

IOT Based Smart Crash Detector for Motorcycle

NIM :1103130153

Fakhri Akbar Pratama

Tugas akhir ini telah diterima dan disahkan untuk memenuhi sebagian syarat memperoleh gelar
pada Program Studi Sarjana S1 Teknik Informatika
Fakultas Informatika
Universitas Telkom

Bandung, …./,,,,,,,,,,/……
Menyetujui

Pembimbing I,	Pembimbing II,
Aji Gautama Putrada, S.T., M.T.	Sidik Prabowo, S.T , M.T
NIP : 15850084-1	NIP : 15870072-1

Ketua Program Studi
Sarjana S1 Teknik Informatika,

Niken Dwi Wahyu Cahayani S.T., M.Kom., Ph.D
NIP: 00750052

LEMBAR PERNYATAAN

Dengan ini saya, Fakhri Akbar Pratama, menyatakan sesungguhnya bahwa Tugas Akhir saya dengan judul **"*Smart Crash Detector* untuk Sepeda Motor Berbasis IOT"** beserta dengan seluruh isinya adalah merupakan hasil karya sendiri, dan saya tidak melakukan penjiplakan yang tidak sesuai dengan etika keilmuan yang belaku dalam masyarakat keilmuan. Saya siap menanggung resiko/sanksi yang diberikan jika di kemudian hari ditemukan pelanggaran terhadap etika keilmuan dalam

Laporan TA atau jika ada klaim dari pihak lain terhadap keaslian karya,

Bandung, …./……../……..

Yang Menyatakan

Fakhri Akbar Pratama

Smart Crash Detector untuk Sepeda Motor Berbasis IOT

Fakhri Akbar Pratama[1], Aji Gautama Putrada[2], Sidik Prabowo[3]

Fakultas Informatika, Universitas Telkom, Bandung
[1]fahribar@students.telkomuniversity.ac.id, [2]ajigps@telkomuniversity.ac.id, [3]pakwowo@telkomuniversity.ac.id

Abstrak

Indonesia adalah negara dengan jumlah penduduk terbesar keempat di dunia. Banyaknya jumlah penduduk berbanding lurus dengan jumlah kendaraan bermotor. Jumlah kendaraan bermotor di Indonesia adalah sebanyak 138.556.669, dimana 81,58% didominasi oleh sepeda motor. Jumlah yang besar tersebut menambah permasalahan baru di bidang keselamatan berkendara. Berdasarkan data dari badan pusat statistik, jumlah kecelakaan lalu lintas di Indonesia selalu besar tiap tahunnya, yang mana menjadikan kecelakaan lalu lintas salah satu dari sepuluh penyebab kematian terbesar di Indonesia. Salah satu cara mengurangi korban jiwa kecelakaan lalu lintas adalah dengan menanamkan alat pada kendaraan bermotor dalam hal ini sepeda motor agar dapat mendeteksi jika terjadi kecelakaan dan agar memudahkan pertolongan dengan koordinat yang dikirimkan oleh alat tersebut. Menggunakan alat mikrokontroler yang terhubung dengan sensor *accelerometer* dan gps, alat ini akan mengambil data dari sepeda motor untuk kemudian mengklasifikasikan data tersebut di peladennya apakah sepeda motor tersebut mengalami kecelakaan atau tidak, jika terjadi kecelakaan maka akan dikirimkan pesan peringatan ke nomor telepon yang sudah didaftarkan. Berdasarkan hasil pengujian, tingkat keberhasilan alat untuk mendeteksi kecelakaan sebesar 94%.

Kata kunci:KNN, *smart crash detector,* mikrokontroler, *event data recorder*

Abstract

Indonesia is the fourth largest population in the world. The number of population is directly proportional to the number of motorized vehicles. The number of motorized vehicles in Indonesia is 138,556,669, of which 81.58% is dominated by motorbikes. This large amount adds new problems in the field of driving safety. Based on data from the central statistical agency, the number of traffic accidents in Indonesia is always large each year, which makes traffic accidents one of the ten biggest causes of death in Indonesia. One way to reduce the toll of traffic accidents is to implant a tool on a motorized vehicle in this case a motorcycle so that it can detect in the event of an accident and to facilitate relief with the coordinates sent by the tool. Using a microcontroller device that is connected to an accelerometer and gps sensor, this tool will retrieve data from a motorcycle and then classify the data on the server whether the motorcycle has an accident or not, if an accident occurs a warning message will be sent to the telephone number that has been registered. Based on the test results, the success rate of the tool to detect accidents is 94%.

Keywords: KNN, *smart crash detector,* mikrokontroler, *event data recorder*

1. Pendahuluan

Latar Belakang

Indonesia adalah negara terbesar keempat di dunia. Jumlah penduduk yang besar ditambah dengan kondisi jalanan di Indonesia menjadikan Indonesia negara di urutan ketiga dalam hal pasar sepeda motor [1]. Data dari Badan Pusat Statistik menyatakan bahwa pada tahun 2017 ada sebanyak 138.556.669 kendaraan bermotor di Indonesia. Jumlah kendaraan bermotor tadi di dominasi oleh sepeda motor sebesar 81,58%. Banyaknya jumlah kendaraan bermotor di Indonesia menghadirkan sebuah permasalahan di bidang lalu lintas. Berdasarkan dari dari Badan Pusat Statistik, jumlah kecelakaan di Indonesia rata – rata meningkat tiap tahunnya. Kecelakaan kendaraan bermotor adalah salah satu dari sepuluh penyebab kematian terbesar di Indonesia [2]. Salah satu cara mengurangi korban jiwa di kecelakaan lalu lintas adalah dengan menanamkan alat pada kendaraan bermotor dalam hal ini sepeda motor agar dapat mendeteksi jika terjadi kecelakaan dan agar memudahkan pertolongan dengan koordinat yang dikirimkan oleh alat tersebut. Penelitian ini mengacu pada jurnal "*Prototype Design of CDR (Crash Data Recorder) on Motorcycle"* oleh Ali Husein Alasiry, Endah Suryawati Ningrum, Eko Budi Utomo, dan Latif Nurohman Bayu Nugroho. Jurnal tersebut membahas tentang pembuatan prototipe CDR untuk sepeda motor menggunakan sensor *accelerometer* dan sensor *accelerometer* yang lalu akan disimpan di USB [3]. Pada penelitian ini, data akan dikirim dari Arduino uno ke peladen yang mana berisi data dari sensor GPS dan accelerometer untuk kemudian diklasifikasi menggunakan metode K-*Nearest Neighbor.*

Arduino adalah mikrokontroler berbasis ATmega328P yang memiliki 14 pin *input/output* digital (6 pinnya digunakan untuk *output* PWM), terdiri dari 6 pin analog, sebuah pin 16 MHz Kristal quartz, sebuah konektor

USB, sebuah *jack* untuk daya, sebuah *header* ICSP dan sebuah tombol *reset*. Arduino uno dapat diprogram menggunakan bahasa pemrograman C [4].

K *Nearest Neighbor* adalah sebuah algoritma klasifikasi. K-NN menggunakan jarak vector dari titik yang ingin diklasifikasikan ke titik pembanding sebanyak "K" titik. Label dari titik terdekat terbanyak yang jumlahnya sebanyak K akan menjadi label dari data masukan. Karena klasifikasi sebuah titik tergantung dari klasifikasi titik lain, maka KNN dapat disebut sebagai algoritma klasifikasi yang di supervisi [5]. KNN dipilih berdasarkan penelitian [6] algoritma KNN mudah diterapkan dan cukup cepat untuk mengklasifikasikan kondisi jalan raya.

Tugas akhir ini tentang system pendeteksi kecelakaan *(Crash Detector)* pada sepeda motor dengan menggunakan sensor GPS dan accelerometer yang terhubung dengan Arduino uno untuk kemudian dikirimkan ke peladen yang akan mengklasifikasikan data masukan dari alat menggunakan algoritma K-*Nearest Neighbor*.

Topik dan Batasannya

Masalah yang dibahas pada tugas akhir ini :

1. Bagaimana pengambilan data dari sensor di sepeda motor yang nanti dikirim ke peladen?
2. Bagaimana mengklasifikasikan data kecelakaan menggunakan KNN?
3. Bagaimana membuat sistem peringatan jika terjadi kecelakaan?

Batasan Masalah pada tugas akhir ini adalah:

1. Peladen yang digunakan adalah *Thingspeak*
2. Sepeda motor yang dipasang *smart crash detector* tidak memiliki ECU (*Engine Control Unit)*
3. Pengiriman data ke peladen menggunakan Wi-Fi yang berasal dari telepon genggam
4. Alat tidak dapat mendeteksi kecelakaan yang terjadi ketika sepeda motor yang terpasang alat ditabrak ketika posisi kendaraan sedang berhenti.
5. Parameter sumbu yang digunakan pada alat *smart crash detector* ini adalah x-axis

Tujuan

Tugas akhir ini memiliki tiga tujuan. Pertama, untuk merancang *smart crash detector* yang mengambil data kecelakaan dari kendaraan menggunakan sensor yang dapat mendeteksi jika terjadi kecelakaan. Kedua, membuat mengimplementasikan algoritma klasifikasi KNN pada sistem *smart crash detector* agar dapat mendeteksi kecelakaan. Ketiga, membuat sistem yang dapat memberikan peringatan jika terjadi kecelakaan pada sepeda motor.

Organisasi Tulisan

Tugas akhir ini terdiri dari beberapa bagian, yaitu : 1. Pendahuluan, 2. Studi Terkait, 3. Sistem yang Dibangun, 4. Evaluasi, 5. Kesimpulan

2. Studi Terkait

Penelitian ini adalah pengembangan dari penelitian sebelumnya [3] yang mana bertujuan membuat sebuah pencatat data kecelakaan. Pada penelitian tersebut, data kecelakaan tersimpan di flashdisk yang terdapat di sepeda motor. Selain itu, penelitian ini mengacu kepada jurnal [7] yang menjelaskan tentang mekanisme pendeteksian kejatuhan seseorang menggunakan klasifikasi data *k-nearest neighbor*. Hasil pada jurnal tersebut memprediksi tingkat akurasi pendeteksian lebih dari 80%.

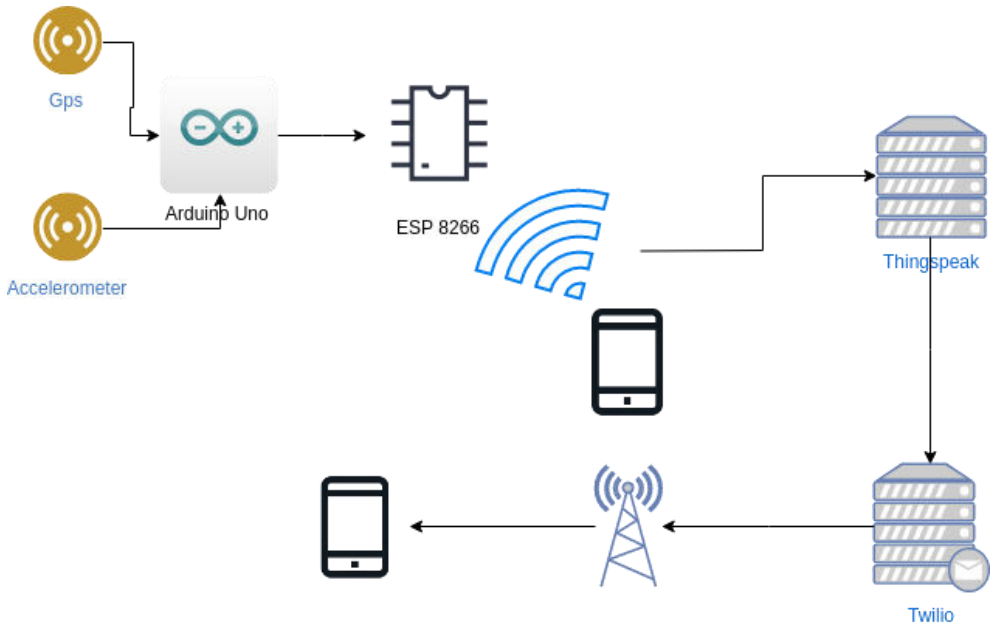
Berdasarkan penelitian – penelitian diatas, penulis menambahkan konsep *internet of things* (IOT) untuk memudahkan pengaksesan data dan pemberitahuan ketika terjadi kecelakaan. Selain itu, penulis juga menambahkan mekanisme sms *gateway* untuk memberikan info kepada nomor yang terdaftar jika terjadi kecelakaan pada sepeda motor. Metode *k-nearest* neighbor digunakan dengan memanfaatkan data yang diekstrak dari sensor. Rancangan *smart crash detector* menggunakan arduino uno dan menggunakan sensor accelerometer dan gps. Bagian ini berisi teori/studi/literatur yang mendukung (terkait erat) dengan topik TA yang dikerjakan.

3. Sistem yang Dibangun

5.1. Pembuatan Alat

Alat yang dibuat terdiri dari mikrokontroller *Arduino Uno* dan sensor accelerometer sebagai pengukur *triaxis* serta gps sebagai penunjuk lokasi.

5.1.1. Gambaran Sistem

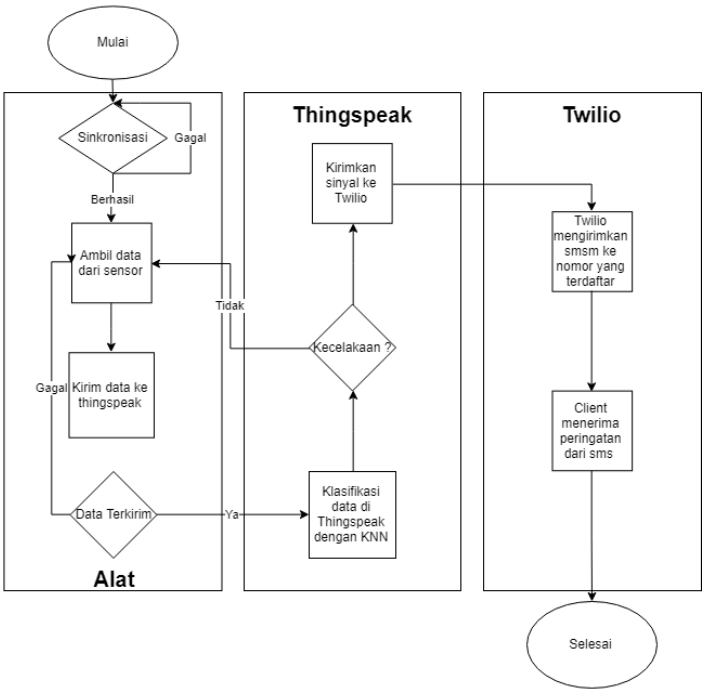


Gambar 1. Skema Alat

Sensor *accelerometer* dan gps yang terhubung dengan *Arduino Uno* menggunakan serial melakukan *write* yang kemudian di *listen* oleh ESP8266 yang terhubung menggunakan serial. ESP8266 kemudian mengirimkan datanya menggunakan http dengan Wi-Fi yang dipancarkan oleh telepon genggam sebagai gateway ke peladen *thingspeak*. Peladen *thingSpeak* yang terhubung dengan Matlab kemudian mengolah data dari mikrokontroller menggunakan metode *k-nearest neighbor.* Data perubahan pada sumbu x pada sensor *accelerometer* dan yang melebihi *threshold* yang telah ditentukan dan data akselerasi kemudian dikirimkan menggunakan http ke *server* Twillio. *Server* Twillio yang mendapatkan data dari *server* ThingSpeak bertugas dari sebagai *server sms gateway* yang digunakan untuk mengirimkan sms ke nomor klien yang telah terdaftar sebelumnya menggunakan http to sms protocol ke nomor yang sudah terdaftar agar dapat diberikan pertolongan oleh klien yang menerima sms dari sistem *smart crash detector* tersebut.

5.2. Rancangan Sistem

Sistem *smart crash detector* pada tugas akhir ini merupakan sistem yang merekam data kecepatan serta lokasi kendaraan yang kemudian dapat mendeteksi jika terjadi kecelakaan pada sepeda motor. Berikut *Flowchart* dari *smart crash detector* :



Gambar 2. Flowchar*t*

Pertama, sistem *smart crash detector* melakukan sinkronisasi dengan *server* agar koneksi antara sistem dengan *server* terintegrasi dengan baik guna lancar dan tuntasnya pengiriman data. Apabila gagal mengalami sinkronisasi, maka akan dilakukan sinkronisasi ulang sampai sinkronisasi berhasil. Setelah berhasil melakukan sinkronisasi, maka akan dilanjutkan ke tahap selanjutnya. Tahap kedua adalah mikrokontroller mengambil data dari sensor *accelerometer*, sensor GPS, dan mengekstraksi data dari sensor yang dipasang pada sepeda motor. Data yang telah diterima oleh mikrokontroller lalu dikirimkan lewat jaringan internet ke *server* ThingSpeak.. Setelah itu akan dilanjutkan ke tahap berikutnya. Tahap keempat *server* ThingSpeak yang terhubung dengan MATLAB akan melakukan klasifikasi data yang didapatkan dari mikrokontroller dengan menggunakan metode *k nearest neighbor*. ThingSpeak akan mendeteksi jika ada abnormalitas dalam data yang dikirimkan. Jika terdapat data yang abnormal, maka berlanjut ke tahap selanjutnya. Jika tidak, maka akan terus dilakukan klasifikasi data.. Tahap selanjutnya adalah pengiriman data yang berisikan data lokasi dan pemberitahuan bahwa terjadi kecelakaan sepeda motor. Data dikirimkan dari *server* ThingSpeak ke *server* Twillio sebagai *sms gateway*. Jika data diterima oleh Twillio, maka akan diteruskan ke tahap selanjutnya. Jika tidak, maka data akan dikirimkan hingga data benar – benar sampai ke Twillio. Tahap selanjutnya Twillio akan mengirimkan *sms* berisikan pemberitahuan kecelakaan dan lokasi GPS. *Client* kemudian akan menerima pesan melalui *sms* yang mana akan diambil tindakan oleh *client* terhadap pengendara sepeda motor. Setelah tahap ini, maka mekanisme *smart crash detector* selesai.

5.3. Metode Klasifikasi

5.3.1. K-Nearest Neighbor (KNN)

K-Nearest Neighbour adalah sebuah metode klasifikasi yang *straightforward* [8]. *K-Nearest Neighbour* melakukan klasifikasi berdasarkan tetangga terdekat sejumlah K terbanyak. Dalam penentuan kelas suatu masukan, k-NN melakukan dua tahapan, yaitu pertama adalah penentuan tetangga terdekat dan yang kedua penentuan kelas dari suatu masukan berdasarkan tetangga tersebut.

5.4. Skenario Pengujian

Skenario pengujian dari *smart crash detector* dilakukan dengan cara meletakkan alat *smart crash detector* di sepeda motor lalu pengendara motor tersebut melakukan uji berkendara normal, lalu berkendara dengan kecepatan tinggi dan berbelok – belok, dan yang terakhir pengendara motor melakukan simulasi kecelakaan dengan menjatuhkan diri ketika sedang berjalan dengan kecepatan sedang.

4. Evaluasi

Bagian ini berisi dua sub-bagian, yaitu Hasil Pengujian dan Analisis Hasil Pengujian. Pengujian dan analisis yang dilakukan selaras dengan tujuan TA sebagaimana dinyatakan dalam Pendahuluan.

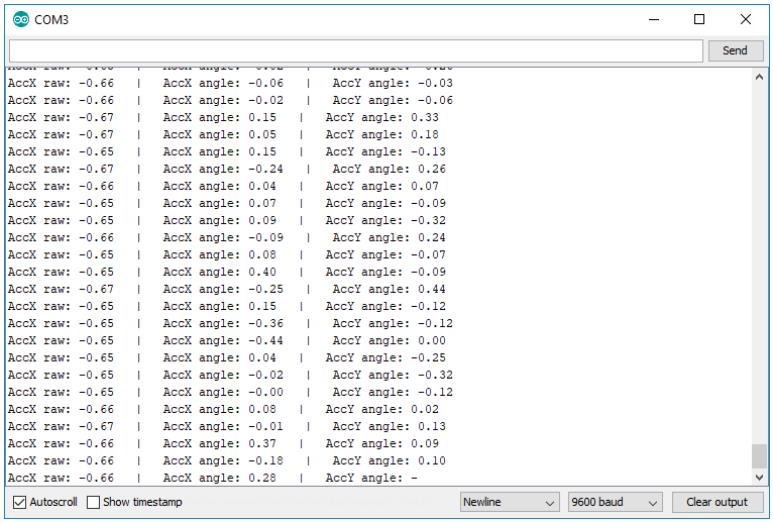
4.1 Hasil Pengujian

Pada tugas akhir ini, penulis menggunakan tiga scenario untuk pengujian terhadap *smart crash detector*. Skenario tersebut adalah :

1. **Skenario 1: Pengujian pengambilan data dari sensor accelerometer dan gps.**
   Pada skenario ini, dilakukan uji coba pengambilan data dari sensor ke mikrokontroler *Arduino Uno*.
2. **Skenario 2: Pengiriman data dari mikrokontroler ke peladen *Thingspeak* dan klasifikasi menggunakan *k-Nearest Neighbor***
   Dalam skenario ini, data hasil sensor di mikrokontroler akan dikirimkan ke peladen *Thingspeak* melalui Wi-Fi. Setelah data dari mikrokontroler masuk ke peladen, lalu data tersebut akan di klasifikasi menggunakan metode *k-Nearest Neihgbour* dengan k=3.
3. **Skenario 3: *Testing* sms *gateway* dari *Twilio* ke nomor yang sudah terdaftar.**
   Skenario ini akan dilaksanakan dengan memasukkan data dengan klasifikasi kecelakaan yang kemudian peladen akan mengirimkan paket ke *Twillio* untuk mengirimkan pesan ke nomor telepon yang telah terdaftar.

4.2 Analisis Hasil Pengujian

4.2.1. **Skenario 1: Pengujian pengambilan data dari sensor accelerometer dan gps.**



Gambar 3. Hasil pengambilan data dari sensor

Berikut adalah gambar dari hasil pengujian satu

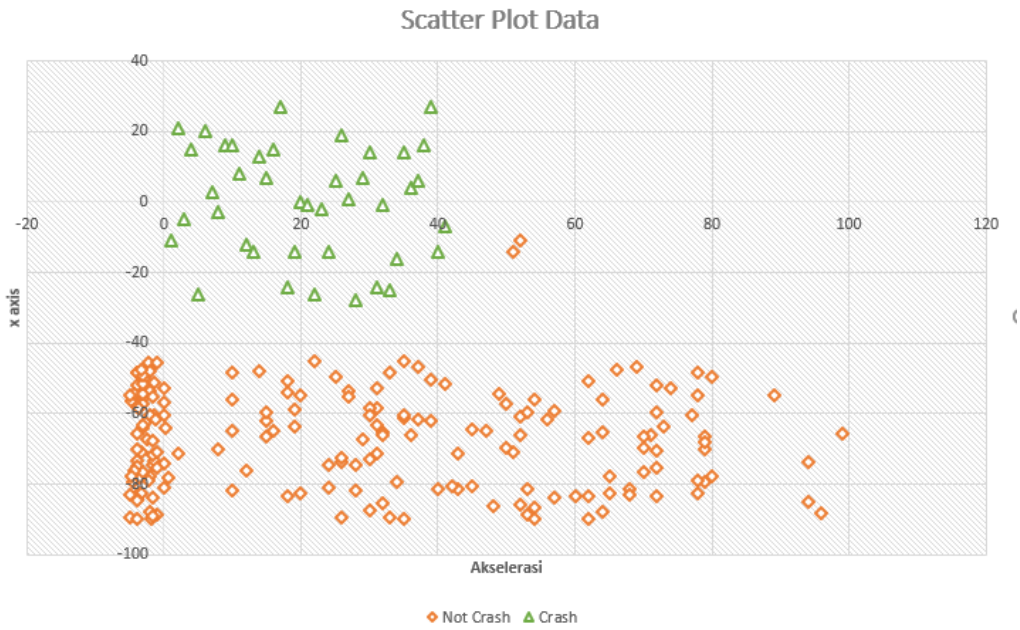
Skenario pertama menguji pengambilan data dari sensor oleh Arduino uno. Berdasarkan gambar 3, dapat dilihat bahwa sensor accelerometer mengeluarkan nilai sumbu dengan urutan sumbu x, sumbu y dan sumbu z. telah terhubung dengan mikrokontroler dan ada komunikasi antara mikrokontroler dan sensor. Sensor *accelerometer* yang digunakan pada penelitian ini adalah MPU 6050 dan sensor gps yang digunakan adalah GY-61 ADXL335.

4.2.2. **Skenario 2: Pengiriman data dari mikrokontroler ke peladen *Thingspeak* dan klasifikasi menggunakan *k-Nearest Neighbor***

Skenario kedua tentang pengiriman data dari mikrokontroler ke *Thingspeak* dan klasifikasi menggunakan *k-Nearest Neighbor*.

Gambar 4. Hasil Pengiriman data di Thingspeak

Gambar di atas adalah tampilan *channels* di thingspeak yang menampilkan 4 field yaitu field 1 untuk akselerasi, field 2 untuk latitude, field 3 untuk longitude dan field 4 menunjukkan *crash.* Seperti yang terlihat di atas, data yang dikirimkan oleh alat *smart crash detector* berhasil mengirimkan data ke thingspeak. Data yang dikirimkan oleh alat *smart crash detector* adalah akselerasi, x-axis dan y-axis yang didapatkan dari sensor *accelerometer* dan data latitude dan longitude yang didapatkan dari sensor gps. Proses pengiriman data dari *smart crash detector* ke thingspeak terjadi setiap 5 detik karena jika pengiriman dilakukan setiap detik, ditakutkan akan terjadi *flooding* pada *server* dan tidak ada proses pendeteksian *flooding* pada *server* pada alat *smart crash detector* Data *testing* yang dipergunakan untuk pengujian skenario dua adalah sebagai berikut:



Gambar 5. *Scatter Plot* dari data pengujian

Pada skenario ini, jumlah data pembanding adalah 200 data sedangkan data yang ditesting berjumlah 50 data. Skenario ini menggunakan jumlah k=3. Berdasarkan gambar 4, data yang masuk ke dalam kelas *crash* dilambangkan dengan segitiga berwarna hijau, sedangkan lambing jajar genjang oranye melambangkan data yang tidak *crash* atau tidak kecelakaan. Berdasarkan data tersebut, *confusion matrix* dari hasil pengujian menggunakan metode klasifikasi *k-Nearest Neighbor* dengan k= 3 adalah

n=50	Predicted : No	Predicted : YES	50
Actual: NO	37	1	38
Actual: YES	0	12	12
	37	13	50

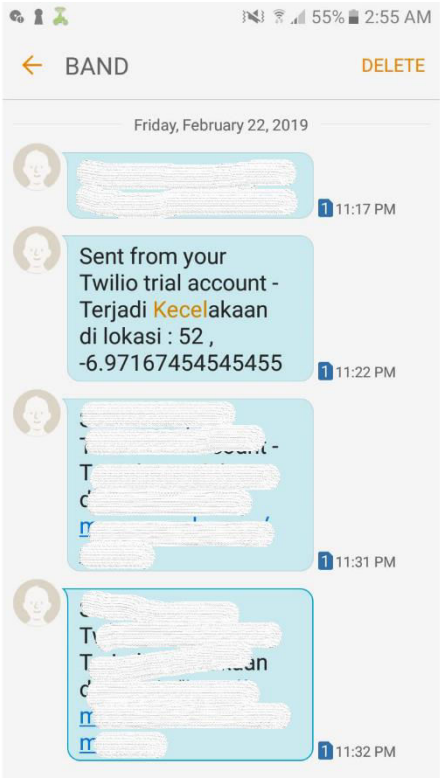
Gambar 6 Confusion Matrix

Mengacu pada matriks di atas, nilai akurasi dari hasil pengujian adalah 98% dengan tingkat kesalahan klasifikasi sebesar 2% dengan presisi sebesar 92,3%, tingkat *recall* 100%, dan tingkat *false alarm rate* 2,63%. Tingkat presisi 92,3% artinya dari 100 kali pengujian ada 8 yang salah klasifikasi. Tingkat *recall* 100% artinya dari jumlah data yang terdeteksi *crash* sama dengan jumlah *crash* yang sebenarnya. Sedangkan tingkat *false alarm rate* sebesar 2,63% artinya dari semua kejadian yang terdeteksi *crash*, ada 2,63% kemungkinan data tersebut termasuk ke *not crash*. Sebelum mendapatkan nilai akurasi sebesar 98%, penulis beberapa kali mendapatkan nilai akurasi yang rendah, namun setelah menambahkan data pembanding dan menambahkan jumlah k menjadi 3 untuk pengujian ini, akhirnya didapatkan nilai akurasi 98%.

4.2.3. **Skenario 3: Pengujian sms *gateway* dari *Twilio* ke nomor yang sudah terdaftar.**

Pengujian yang ketiga adalah pengiriman sms *gateway* dari *Twillio* ke telepon dengan nomor yang sudah didaftarkan. Setelah melakukan pengaturan pada peladen *Twillio* dengan menambahkan nomor telepon untuk pengujian, kemudian penulis mengirimkan data yang tergolong *crash* dari alat *smart crash detector* ke peladen *Thingspeak* yang kemudian merespon dengan mengirimkan sinyal ke *Twillio* untuk mengirimkan pemberitahuan

kepada nomor yang sudah didaftarkan sebelumnya beserta koordinat gps alat *smart crash detector*. Gambar di bawah menunjukkan sms yang dikirimkan oleh peladen *Twillio* kepada nomor telepon yang terdaftar.



Gambar 7 Pesan pemberitahuan kecelakaan dari twillio

5. Kesimpulan

5.1 Kesimpulan

Kesimpulan yang di dapat dari seluruh pengerjaan tugas akhir ini adalah :

1. Metode komunikasi serial lebih baik digunakan untuk pengiriman data yang terjadi secara terus menerus karena metode serial lebih cepat dibandingkan dengan I2C.
2. Tingkat akurasi metode klasifikasi *k-Nearest Neighbor* tergantung dari kualitas data pembanding dan jumlah “k”.
3. Matlab adalah *programming language* yang sangat mumpuni dan memiliki *library* yang komprehensif dalam hal pengklasifikasian data.

5.2.Saran

5.2.1 Diharapkan penelitian kedepannya menggunakan metode klasifikasi lain untuk lebih dapat membuat sistem yang *real time.*

5.2.2 Diharapkan kedepannya ada infrastruktur yang mampu menjadikan kendaraan bermotor sebagai *end user* dalam suatu jaringan agar dapat memudahkan proses pemberitahuan kecelakaan.

5.2.3. Diharapkan adanya sistem layanan *emergency* yang dapat memudahkan pengembangan dari alat *smart crash detector* ini.

Daftar Pustaka

[1] G. M. Nayazri, "Indonesia Masih Jadi Pasar Sepeda Motor Ketiga di Dunia," Kompas, 02 12 2017. [Online]. Available: https://otomotif.kompas.com/read/2017/12/02/082200615/indonesia-masih-jadi-pasar-sepeda-motor-ketiga-di-dunia. [Accessed 12 December 2018].

[2] N. Mboi, I. M. Surbakti and P. I. Trihandini, Phd, "'On the road to universal health care in Indonesia,

1990-2016: a systemic analysis for the Global Burden of Disease Study 2016," *The Lancet,* vol. 392, no. 10147, pp. 531-612, 2018.

[3] A. H. Alasiry, E. S. Ningrum, E. B. Utomo and L. N. B. Nugroho, "Prototype Design of CDR (Crash Data Recorder) on Motorcycle," in *2016 International Electronics Symposium (IES)*, Surabaya, 2016.

[4] Admin, "Arduino Uno Rev3," Arduino AG, 2018. [Online]. Available: https://www.arduino.cc/en/main/arduinoBoardUno. [Accessed 12 December 2018].

[5] V. Umamaheswaran, "Comprehending K-means and KNN Algorithms - Becoming Human: Artificial Intelligence Magazine," Medium, 12 November 2018. [Online]. Available: https://becominghuman.ai/comprehending-k-means-and-knn-algorithms-c791be90883d. [Accessed 13 December 2018].

[6] M. J. Islam, J. Q. M. Wu, M. Ahmadi and M. A. Sid-Ahmed, "Investigating the Performance of Naive-Bayes Classifiers and K-Nearest Neighbor Classifiers," in *2nd International Conference on Convergent Information Technology*, Gyongju, South Korea, 2007.

[7] C.-L. Liu, C.-H. Lee and P.-M. Lin, "A Fall Detection System using K-Nearest Neighbor Classifier," *Expert SYstems with Applications,* vol. 37, no. 10, pp. 7174-7181, 2010.

[8] P. Cunningham and S. J. Delany, "k-Nearest Neighbour Classifiers," Mult. Classif. Syst, Dublin, 2007.

Lampiran

Lampiran dapat berupa detil data dan contoh lebih lengkapnya, data-data pendukung, detail hasil pengujian, analisis hasil pengujian, detail hasil survey, surat pernyataan dari tempat studi kasus, screenshot tampilan sistem, hasil kuesioner dan lain-lain.

Data pembanding

created_at	entry_id	Acceleration	Latitude	Longitude	Crash	x-axis
2019-01-18 13:27:06 WIB	1	68	-6.97899	107.63127	0	-81.46
2019-01-18 13:27:42 WIB	2	70	-6.97946	107.62856	0	-76.49
2019-01-18 13:28:20 WIB	3	25	-6.97896	107.63136	0	-49.7
2019-01-18 13:28:52 WIB	4	32	-6.97893	107.63129	0	-65.23
2019-01-18 13:29:10 WIB	5	0	-6.97167	107.65779	0	-52.77
2019-01-18 13:29:27 WIB	6	80	-6.97878	107.6288	0	-49.76
2019-01-18 13:31:00 WIB	7	53	-6.97909	107.62874	0	-88.76
2019-01-18 13:31:35 WIB	8	72	6.9736	107.63308	0	-75.27
2019-01-18 13:32:28 WIB	9	12	-6.97896	107.63118	0	-76.18
2019-01-18 13:34:23 WIB	10	43	-6.97895	107.63123	0	-71.12
2019-01-18 13:35:04 WIB	11	31	-6.97916	107.63129	0	-71.2

2019-01-18 13:35:23 WIB	12	-7.36	-6.97443	107.63281	1	-11
2019-01-18 13:35:57 WIB	13	32	-6.97897	107.63106	0	-85.33
2019-01-18 13:36:30 WIB	14	78	-6.97897	107.63126	0	-54.78
2019-01-18 13:36:58 WIB	15	78	-6.97897	107.63106	0	-82.58
2019-01-18 13:37:21 WIB	16	74	-6.97898	107.63125	0	-52.66
2019-01-18 13:38:01 WIB	17	60	-6.97893	107.63129	0	-83.45
2019-01-18 13:38:34 WIB	18	79	-6.97893	107.63129	0	-79.29
2019-01-18 13:39:04 WIB	19	71	-6.97895	107.63129	0	-66.22
2019-01-18 13:40:19 WIB	20	35	-6.97896	107.63124	0	-61.44
2019-01-18 14:32:53 WIB	21	79	-697897	107.63124	0	-70.3
2019-01-18 14:33:08 WIB	22	33	-6.97896	107.63136	0	-48.2
2019-01-18 14:33:24 WIB	23	26	-6.97896	107.63124	0	-89.65
2019-01-18 14:33:39 WIB	24	10	-6.9792	107.62873	0	-81.73
2019-01-18 14:33:55 WIB	25	37	-6.97909	107.62882	0	-46.83
2019-01-18 14:34:10 WIB	26	72	-6.97903	107.63114	0	-51.88
2019-01-18 14:34:26 WIB	27	66	-6.97898	107.63125	0	-47.37
2019-01-18 14:34:42 WIB	28	57	-6.97898	107.63128	0	-59.27
2019-01-18 14:34:57 WIB	29	45	-6.97845	107.63155	0	-80.5
2019-01-18 14:35:14 WIB	30	51	-6.97898	107.63126	0	-70.9
2019-01-18 14:35:29 WIB	31	15	-6.97894	107.63128	0	-61.96
2019-01-18 14:35:45 WIB	32	80	-6.97894	107.63116	0	-77.66
2019-01-18 14:36:00 WIB	33	48	-6.97942	107.63116	0	-86.14
2019-01-18 14:36:15 WIB	34	62	-6.97895	107.63107	0	-50.82
2019-01-18 14:36:31 WIB	35	64	-6.97897	107.63104	0	-56.2
2019-01-18 14:36:46 WIB	36	72	-6.97909	107.62874	0	-59.64
2019-01-18 14:37:01 WIB	37	79	-6.98004	107.62927	0	-66.5
2019-01-18 14:37:17 WIB	38	19	-6.97916	107.63129	0	-63.7
2019-01-18 14:37:32 WIB	39	28	-6.97855	107.63109	0	-81.93
2019-01-18 14:37:48 WIB	40	43	-6.97946	107.62856	0	-81.24
2019-01-18 14:38:03 WIB	41	42	-6.97932	107.6287	0	-80.42
2019-01-18 14:38:19	42	31	-6.97896	107.63118	0	-52.96

WIB						
2019-01-18 14:38:34 WIB	43	68	-6.97891	107.63109	0	-83.1
2019-01-18 14:38:49 WIB	44	53	-6.97898	107.63126	0	-81.53
2019-01-18 14:39:05 WIB	45	33	-6.97898	107.63123	0	-89.4
2019-01-18 14:39:20 WIB	46	65	-6.97897	107.63124	0	-77.58
2019-01-18 14:39:35 WIB	47	54	-6.97895	107.63127	0	-89.77
2019-01-18 14:39:50 WIB	48	26	-6.97897	107.63104	0	-73.57
2019-01-18 14:40:06 WIB	49	78	-6.97942	107.63116	0	-48.3
2019-01-18 14:40:21 WIB	50	24	-6.97898	107.63115	0	-74.66
2019-01-18 14:40:37 WIB	51	70	-6.97878	107.6288	0	-69.85
2019-01-18 14:40:53 WIB	52	72	-6.97895	107.63123	0	-70.4
2019-01-18 14:41:10 WIB	53	45	-697897	107.63124	0	-64.38
2019-01-18 14:41:27 WIB	54	35	-6.97895	107.63107	0	-45.17
2019-01-18 14:41:43 WIB	55	31	-6.97898	107.63126	0	-58.51
2019-01-18 14:41:59 WIB	56	10	-6.97838	107.63152	0	-48.22
2019-01-18 14:42:15 WIB	57	41	-6.97898	107.63125	0	-51.77
2019-01-18 14:42:30 WIB	58	70	-6.97958	107.62839	0	-66.56
2019-01-18 14:42:45 WIB	59	26	-6.97932	107.6287	0	-72.36
2019-01-18 14:43:03 WIB	60	20	-6.97894	107.63116	0	-82.78
2019-01-18 14:43:21 WIB	61	14	-6.97946	107.62856	0	-47.94
2019-01-18 14:43:39 WIB	62	52	-6.97896	107.63121	0	-85.96
2019-01-18 14:43:57 WIB	63	65	-6.9793	107.62865	0	-82.78
2019-01-18 14:44:16 WIB	64	64	-6.97855	107.63109	0	-65.4
2019-01-18 14:44:34 WIB	65	37	-6.97898	107.63128	0	-61.77
2019-01-18 14:44:52 WIB	66	20	-6.97958	107.62839	0	-54.78
2019-01-18 14:45:10 WIB	67	29	-6.97845	107.63155	0	-67.49
2019-01-18 14:45:28 WIB	68	27	-6.97897	107.63124	0	-53.69
2019-01-18 14:45:46 WIB	69	24	-6.97896	107.63121	0	-80.8
2019-01-18 14:46:04 WIB	70	22	-6.97899	107.63127	0	-45.21
2019-01-18 14:46:22 WIB	71	64	-6.98004	107.62927	0	-87.93
2019-01-18 14:46:40 WIB	72	62	-6.97896	107.6312	0	-83.27

2019-01-18 14:46:58 WIB	73	62	-6.97896	107.6312	0	-66.81
2019-01-18 14:47:16 WIB	74	35	6.9737	107.63308	0	-60.29
2019-01-18 14:47:34 WIB	75	49	-6.97833	107.63194	0	-54.2
2019-01-18 14:47:52 WIB	76	-2	-6.971586	107.657883	0	-46.98
2019-01-18 14:48:10 WIB	77	36	-6.97894	107.63128	0	-66.27
2019-01-18 14:48:28 WIB	78	52	-6.97845	107.63187	0	-60.92
2019-01-18 14:48:46 WIB	79	35	-6.97895	107.63129	0	-89.93
2019-01-18 14:49:03 WIB	80	19	-6.97898	107.63123	0	-58.87
2019-01-18 14:49:21 WIB	81	54	-6.97833	107.63194	0	-86.8
2019-01-18 14:49:39 WIB	82	30	-6.9793	107.62865	0	-58.36
2019-01-18 14:49:57 WIB	83	53	-6.97891	107.63109	0	-59.69
2019-01-18 14:50:16 WIB	84	10	6.9737	107.63308	0	-55.82
2019-01-18 14:50:33 WIB	85	79	-6.97896	107.6312	0	-68.2
2019-01-18 14:50:51 WIB	86	15	-6.97897	107.63126	0	-59.61
2019-01-18 14:51:09 WIB	87	18	-6.97845	107.63187	0	-83.29
2019-01-18 14:51:28 WIB	88	62	-6.97883	107.62875	0	-89.8
2019-01-18 14:51:45 WIB	89	39	-6.97898	107.63126	0	-50.22
2019-01-18 14:52:03 WIB	90	69	-6.97898	107.63125	0	-46.69
2019-01-18 14:52:22 WIB	91	-5	-6.97251	107.63416	1	21
2019-01-18 14:52:40 WIB	92	52	-6.97903	107.63114	0	-65.94
2019-01-18 14:52:59 WIB	93	32	-6.97896	107.6312	0	-66.23
2019-01-18 14:53:17 WIB	94	34	-6.97946	107.62856	0	-79.5
2019-01-18 14:53:36 WIB	95	18	-6.9792	107.62873	0	-50.93
2019-01-18 14:53:53 WIB	96	54	-6.97838	107.63152	0	-55.82
2019-01-18 14:54:11 WIB	97	-3.7	-6.971606	107.657863	0	-58.31
2019-01-18 14:54:29 WIB	98	15	-6.97893	107.63129	0	-66.65
2019-01-18 14:54:48 WIB	99	57	-6.97883	107.62875	0	-83.83
2019-01-18 14:55:05 WIB	100	-6.59	-6.971621	107.657849	1	-5
2019-01-18 14:55:23 WIB	101	10	-6.97794	107.63223	0	-64.85
2019-01-18 14:55:42 WIB	102	-7	-6.97715	107.63267	1	15
2019-01-18 14:55:59	103	-1.73	-6.97251	107.63416	0	-83.98

WIB						
2019-01-18 14:56:17 WIB	104	-2	-6.971606	107.657863	0	-77.71
2019-01-18 14:56:34 WIB	105	-5	-6.971655	107.657815	1	-26
2019-01-18 14:56:51 WIB	106	50	-6.97898	107.63115	0	-69.52
2019-01-18 14:57:10 WIB	107	78	-6.97895	107.63127	0	-78.85
2019-01-18 14:57:29 WIB	108	30	-6.97909	107.62882	0	-72.85
2019-01-18 14:57:47 WIB	109	-5	-6.97528	107.63279	1	20
2019-01-18 14:58:08 WIB	110	-4	-6.97583	107.63287	0	-48.35
2019-01-18 14:58:26 WIB	111	-3	-6.97517	107.63285	0	-55.75
2019-01-18 14:58:44 WIB	112	-7	-6.97715	107.63267	1	3
2019-01-18 14:59:02 WIB	113	-2.29	-6.97153	107.65791	0	-45.62
2019-01-18 14:59:20 WIB	114	-1.46	-6.971591	107.657878	0	-51.11
2019-01-18 14:59:35 WIB	115	-2.52	-6.97153	107.65794	0	-67.42
2019-01-18 14:59:53 WIB	116	0.28	-6.97588	107.63305	0	-63.97
2019-01-18 15:00:11 WIB	117	-2	-6.971596	107.657873	0	-47.99
2019-01-18 15:00:29 WIB	118	-6.55	-6.97158	107.6579	1	-3
2019-01-18 15:00:49 WIB	119	-4.88	-6.97163	107.657839	0	-89.61
2019-01-18 15:01:07 WIB	120	0	-6.97156	107.65789	0	-80.82
2019-01-18 15:01:25 WIB	121	-2.58	-6.97157	107.65789	0	-62.46
2019-01-18 15:01:42 WIB	122	-1.4	-6.97153	107.65794	0	-60.38
2019-01-18 15:02:00 WIB	123	-3	-6.971616	107.657854	0	-64
2019-01-18 15:02:18 WIB	124	-6	-6.97157	107.65789	1	16
2019-01-18 15:02:35 WIB	125	-4	-6.97588	107.63305	0	-73.19
2019-01-18 15:02:54 WIB	126	-3	-6.971645	107.657825	0	-57.22
2019-01-18 15:03:11 WIB	127	-5	-6.97158	107.6579	1	16
2019-01-18 15:03:26 WIB	128	-3.22	-6.9736	107.63308	0	-70.97
2019-01-18 15:03:44 WIB	129	-5	-6.971621	107.657849	1	8
2019-01-18 15:04:01 WIB	130	-6.6	-6.97528	107.63279	1	-12
2019-01-18 15:04:19 WIB	131	-7.32	-6.97156	107.65789	1	-14
2019-01-18 15:04:40 WIB	132	-4.63	-6.971596	107.657873	0	-79.31
2019-01-18 15:04:58 WIB	133	-6.82	-6.97692	107.63247	1	13

2019-01-18 15:05:15 WIB	134	16	-6.97794	107.63223	0	-64.95
2019-01-18 15:05:33 WIB	135	-6	-6.97352	107.63347	1	7
2019-01-18 15:05:51 WIB	136	-2	-6.97153	107.65794	0	-60.57
2019-01-18 15:06:09 WIB	137	-1	-6.971665	107.657806	0	-88.75
2019-01-18 15:06:24 WIB	138	-7	-6.97154	107.65794	1	15
2019-01-18 15:06:41 WIB	139	-6	-6.971601	107.657868	1	27
2019-01-18 15:07:00 WIB	140	0	-6.971591	107.657878	0	-56.73
2019-01-18 15:07:17 WIB	141	-2	-6.97166	107.65781	0	-53.59
2019-01-18 15:07:36 WIB	142	-7	-6.971635	107.657835	1	-24
2019-01-18 15:07:54 WIB	143	-2.77	-6.97583	107.63287	0	-49.97
2019-01-18 15:08:11 WIB	144	-7.65	-6.971581	107.657888	1	-14
2019-01-18 15:08:29 WIB	145	-3.2	-6.971616	107.657854	0	-54.52
2019-01-18 15:08:46 WIB	146	-4.72	-6.97153	107.65794	0	-56.3
2019-01-18 15:09:03 WIB	147	-5.44	-6.97159	107.65784	1	0
2019-01-18 15:09:21 WIB	148	-3.91	-6.971655	107.657815	0	-57.9
2019-01-18 15:09:38 WIB	149	-3.76	-6.971645	107.657825	0	-82.4
2019-01-18 15:09:56 WIB	150	0.63	-6.97352	107.63347	0	-77.99
2019-01-18 15:10:14 WIB	151	-1.62	-6.97692	107.63247	0	-74.55
2019-01-18 15:10:31 WIB	152	-1.5	-6.97166	107.65781	0	-73.44
2019-01-18 15:10:49 WIB	153	-1.53	-6.971665	107.657806	0	-71.6
2019-01-18 15:11:07 WIB	154	-1.12	-6.97473	107.63258	0	-61.48
2019-01-18 15:11:25 WIB	155	-2.34	-6.97715	107.63267	0	-78.92
2019-01-18 15:11:42 WIB	156	-6.12	-6.97715	107.63267	1	-1
2019-01-18 15:12:00 WIB	157	-4	-6.97165	107.65782	0	-65.64
2019-01-18 15:12:19 WIB	158	0	-6.971625	107.657844	0	-60.52
2019-01-18 15:12:36 WIB	159	-1	-6.97164	107.65783	0	-70.89
2019-01-18 15:12:53 WIB	160	-3	-6.97715	107.63267	0	-63.21
2019-01-18 15:13:11 WIB	161	-4	-6.97153	107.65791	0	-48.53
2019-01-18 15:13:29 WIB	162	27	6.9736	107.63308	0	-55.2
2019-01-18 15:13:49 WIB	163	-3.2	-6.971635	107.657835	0	-82.54
2019-01-18 15:14:06	164	-7.8	-6.97715	107.63267	1	-26

WIB						
2019-01-18 15:14:26 WIB	165	-5.8	-6.97167	107.657801	1	-2
2019-01-18 15:14:44 WIB	166	-3.54	-6.97164	107.65783	0	-78.1
2019-01-18 15:15:01 WIB	167	-3.66	-6.971586	107.657883	0	-51.95
2019-01-18 15:15:18 WIB	168	-6.28	-6.97159	107.65789	1	-14
2019-01-18 15:15:36 WIB	169	-4	-6.97167	107.657801	0	-69.99
2019-01-18 15:15:53 WIB	170	0	-6.97715	107.63267	0	-74.2
2019-01-18 15:16:11 WIB	171	-4	-6.97153	107.65794	0	-74.88
2019-01-18 15:16:28 WIB	172	-4	-6.97473	107.63258	0	-52.18
2019-01-18 15:16:46 WIB	173	-3	-6.97153	107.65794	0	-76.4
2019-01-18 15:17:04 WIB	174	-4	-6.97668	107.63274	0	-84.36
2019-01-18 15:17:22 WIB	175	-1.67	-6.97165	107.65782	0	-55.36
2019-01-18 15:17:40 WIB	176	-3.77	-6.97157	107.65786	0	-58.58
2019-01-18 15:17:58 WIB	177	-1.89	-6.971625	107.657844	0	-89.67
2019-01-18 15:18:15 WIB	178	-4	-6.97163	107.657839	0	-89.9
2019-01-18 15:18:33 WIB	179	-1	-6.97159	107.65784	0	-45.56
2019-01-18 15:18:50 WIB	180	-6	-6.97157	107.65786	1	6
2019-01-18 15:19:08 WIB	181	-6	-6.97443	107.63281	1	19
2019-01-18 15:19:25 WIB	182	-1	-6.97159	107.65789	0	-75.43
2019-01-18 15:19:43 WIB	183	-2	-6.9736	107.63308	0	-87.75
2019-01-18 15:20:01 WIB	184	-7	-6.9731	107.6336	1	1
2019-01-18 15:20:18 WIB	185	-3.96	-6.97156	107.65791	0	-84.52
2019-01-18 15:20:36 WIB	186	-4.3	-6.9731	107.6336	0	-54.31
2019-01-18 15:20:54 WIB	187	-7.56	-6.97715	107.63267	1	-28
2019-01-18 15:21:12 WIB	188	-3	-6.971581	107.657888	0	-54.41
2019-01-18 15:21:30 WIB	189	-1.59	-6.97668	107.63274	0	-67.68
2019-01-18 15:21:48 WIB	190	-6	-6.97153	107.65793	1	7
2019-01-18 15:22:06 WIB	191	-6	-6.97156	107.65791	1	14
2019-01-18 15:22:24 WIB	192	-1.62	-6.971611	107.657859	0	-89.24
2019-01-18 15:22:41 WIB	193	-3	-6.971611	107.657859	0	-51.77
2019-01-18 15:22:59 WIB	194	-4.32	-6.97153	107.65793	0	-76.2

2019-01-18 15:23:17 WIB	195	-3	-6.97159	107.65789	0	-49.35
2019-01-18 15:23:35 WIB	196	-4.88	-6.971675	107.657796	0	-54.73
2019-01-18 15:23:53 WIB	197	-3.21	-6.97159	107.65789	0	-47.54
2019-01-18 15:24:14 WIB	198	-7.29	-6.97517	107.63285	1	-24
2019-01-18 15:24:32 WIB	199	-4.72	-6.971601	107.657868	0	-77.72
2019-01-18 15:24:49 WIB	200	-5.47	-6.97154	107.65794	1	-1

D

Acceleration	x-axis	aktual	hasil tes	
96	-88.42	0	0	1
77	-60.45	0	0	1
18	-53.88	0	0	1
-6	-25	1	1	1
56	-61.8	0	0	1
30	-87.62	0	0	1
-6	-16	1	1	1
8	-69.94	0	0	1
50	-57.35	0	0	1
36	-67.64	0	1	0
2	-73.1	0	0	1
34	-46.95	0	0	1
17	-58.71	0	0	1
-5	4	1	1	1
60	-89.84	0	0	1
-7	13	1	1	1
-7	20	1	1	1
11	-52.45	0	0	1
9	-50.3	0	0	1
22	-85.62	0	0	1
62	23	0	1	0
43	-79.36	0	0	1
79	19	0	0	1
59	-74.1	0	0	1
93	-82.99	0	0	1
70	-63.43	0	0	1
2	23	0	1	0
99	-65.65	0	0	1
31	-63.11	0	0	1
2	-71.18	0	0	1
94	-85.16	0	0	1
72	-83.46	0	0	1
40	-81.45	0	0	1
-6	14	1	1	1

-6	4	1	1	1
-7	6	1	1	1
-5	16	1	1	1
-7	27	1	1	1
39	-62.17	0	0	1
47	-64.73	0	0	1
73	-63.77	0	0	1
-5	-82.84	0	0	1
30	-60.36	0	0	1
-6	-14	1	1	1
28	-74.71	0	0	1
94	-73.7	0	0	1
-5	-7	1	1	1
89	-54.6	0	0	1
51	-14	0	0	1
52	-11	0	0	1
Total		12	15	47