TUGAS AKHIR – KI141502

RANCANG BANGUN FALL DETECTION SYSTEM DENGAN ANALISA PERCEPATAN SUDUT ROTASI SECARA REAL TIME MENGGUNAKAN MIKROKONTROLER ARDUINO

M. SATRIO RAMADHANA YUDHONO
NRP 5112100217

Dosen Pembimbing
Waskitho Wibisono, S.Kom., M.Eng., Ph.D.
Henning Titi Ciptaningtyas, S.Kom., M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

TUGAS AKHIR – KI141502

RANCANG BANGUN FALL DETECTION SYSTEM DENGAN ANALISA PERCEPATAN SUDUT ROTASI SECARA REAL TIME MENGGUNAKAN MIKROKONTROLER ARDUINO

M. SATRIO RAMADHANA YUDHONO
NRP 5112100217

Dosen Pembimbing
Waskitho Wibisono, S.Kom., M.Eng., Ph.D.
Henning Titi Ciptaningtyas, S.Kom., M.Kom.

JURUSAN TEKNIK INFORMATIKA
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember
Surabaya 2016

(Halaman ini sengaja dikosongkan)

FINAL PROJECT– KI141502

FALL DETECTION SYSTEM WITH ACCELERATION OF ROTATION ANGLE (QUATERNION) ANALYSIS IN REAL TIME USING MICROCONTROLLER ARDUINO

M. SATRIO RAMADHANA YUDHONO
NRP 5112100217

Advisor
Waskitho Wibisono, S.Kom., M.Eng., Ph.D.
Henning Titi Ciptaningtyas, S.Kom., M.Kom.

DEPARTMENT OF INFORMATICS
FACULTY OF INFORMATION TECHNOLOGY
SEPULUH NOPEMBER INSTITUTE OF TECHNOLOGY
SURABAYA 2016

(Halaman ini sengaja dikosongkan)

LEMBAR PENGESAHAN

RANCANG BANGUN FALL DETECTION SYSTEM DENGAN ANALISA PERCEPATAN SUDUT ROTASI (QUATERNION) SECARA REAL TIME MENGGUNAKAN MIKROKONTROLER ARDUINO

Tugas Akhir

Diajukan Untuk Memenuhi Salah Satu Syarat
Memperoleh Gelar Sarjana Komputer
pada
Rumpun Mata Kuliah Komputasi Berbasis Jaringan
Program Studi S-1 Jurusan Teknik Informatika
Fakultas Teknologi Informasi
Institut Teknologi Sepuluh Nopember

Oleh:
M. Satrio Ramadhana Yudhono
NRP. 5112 100 217

Disetujui oleh Dosen Pembimbing Tugas Akhir:

Waskitho Wibisono, S.Kom., M.Eng., Ph.D. .....
NIP: 19741022 200003 1 001 (pembimbing 1)

Henning Titi Ciptaningtyas, S.Kom., M.Kom. .....
NIP: 1984070 820101 2 2 004 (pembimbing 2)

SURABAYA
JUNI, 2016

(Halaman ini sengaja dikosongkan)

RANCANG BANGUN FALL DETECTION SYSTEM DENGAN ANALISA PERCEPATAN SUDUT ROTASI SECARA REAL TIME MENGGUNAKAN MIKROKONTROLER ARDUINO

Nama Mahasiswa	: M. Satrio Ramadhana Yudhono
NRP	: 5112100217
Jurusan	: Teknik Informatika FTIf-ITS
Dosen Pembimbing I	: Waskitho Wibisono, S.Kom., M.Eng., Ph.D
Dosen Pembimbing II	: Henning Titi Ciptaningtyas, S.Kom., M.Kom.

ABSTRAK

Insiden jatuh merupakan masalah dan sering terjadi di lingkungan sekitar, pada umumnya terjadi pada usia lanjut namun tidak menutup kemungkinan insiden terjatuh terjadi pada siapa pun dan tidak mengenal usia. sebagian kalangan menanggap biasa dan tidak perlu dikhawatirkan, namun untuk kalangan tertentu yaitu bagi seseorang yang lanjut usia ataupun seseorang yang mempunyai indikasi Stroke ini sangat berbahaya, selain itu yang sedang mengalami gangguan kesehatan juga diperlukan perhatian khusus.

Untuk itu penulis menawarkan sebuah solusi untuk membuat rancang bangun fall detection system. Sistem ini akan bekerja menggunakan analisa aktivitas pergerakan fisik users yang didapatkan dari sensor orientasi yang didalamnya terdapat sensor accelerometer, gyroscope, magnetometer yang diletakkan di bagian saku atas. Untuk mendeteksi pergerakan yang mengindikasikan pergerakan jatuh, terlebih dahulu diambil data training pada gerakan (Activity Daily Living) ADL seperti duduk, berjalan dan diambil pula data sensor pada gerakan jatuh untuk digunakan sebagai data analisa, sehingga dapat ditetapkan sebagai nilai threshold untuk gerakan jatuh, namun nilai ini juga

harus dibandingkan dengan nilai tetha sehingga menghasilkan akurasi yang baik.

Metode yang digunakan sebagai pendeteksian gerakan jatuh menggunakan analisa sudut rotasi theta yang berasal dari nilai quaternion(q0,q1,q2,q3) ketika sebelum dan sesudah jatuh ditambah dengan nilai percepatan sumbu x, sumbu y dan sumbu z terhadap gravitasi. Kemudian apabila sudah terdeteksi gerakan jatuh, sistem akan mengirimkan data lokasi kepada keluarga korban dan dikirimkan ke server menggunakan menggunakan SIM908 GSM/GPRS/GPS Shield.

Percobaan dilakukan dengan diletakkan perangkat di saku atas pengguna. Terjadinya transisi gerakan akan menyebabkan perubahan data sensor, pada gerakan jatuh terjadi perubahan data secara signifikan. Hasil pada percobaan menunjukkan tingkat akurasi sebesar 83,75% pada gerakan jatuh. Sedangkan lama waktu untuk mengirimkan SMS dan ke server masing masing 5s. Aplikasi ini diharapkan dapat mengurangi dampak risiko dari insiden terjatuh dengan dilakukannya pertolongan setelah menerima alarm SMS.

Kata kunci**: **fall detection, ADL, accelerometer, gyroscpe threshold.quaternion, SMS.

FALL DETECTION SYSTEM WITH ACCELERATION ANGLE OF ROTATION ANALYSIS IN REAL TIME USING MICROCONTROLLER ARDUINO

Student Name	: Didik Purwanto
NRP	: 5111 100 702
Major	: Teknik Informatika FTIf-ITS
Advisor I	: Imam Kuswardayan, S.Kom., M.T.
Advisor II	: Dr.Eng. Nanik Suciati, S.Kom, M.Kom

ABSTRACT

The incidence of falls is a problem and often occur in the environment, usually occurs in the elderly, but did not rule out a fall incidents happen to anyone and did not know the age. In some circles it is considered normal and nothing to worry about, however, for certain circles that patients with indicated stroke or a person who has a stroke is very dangerous indication, moreover that which is experiencing health problems or aging age is also required special attention.

Authors offer a solution to make the design of the fall detection system. This system will work acceleration *angle of rotation analysis of users obtained from an accelerometer adn gyroscope sensor which is placed in the pocket. To detect the movement that indicates the movement of falling, first taken training data in motion (Activity Daily Living) ADL such as sitting, walking, running and taken on a motion sensor data also fell for use as data analysis, so that the value set as the threshold for a fall.*

The method is used as a fall detection using analysis when the rotation angle before fall and after the fall and the value of the acceleration axis x, y axis and z axis against gravity. Then when it detected a fall, the system will send location data to the affected families and sent to the server using using SIM908 GSM / GPRS / GPS Shield.

The experiments were performed with the device placed in the top pocket of the user. The shift will cause the motion sensor data changes, the fall occurred movement data changes significantly. Results of the trial showed an accuracy rate of 83.75% on the motion fell. While the length of time for sending SMS and server to each 5s.

This application is expected to reduce the impact of the risk of falling incidents with the help did after receiving the alarm SMS.

Keywords**: **fall detection, ADL, accelerometer,gyroscope threshold. SMS.

KATA PENGANTAR

بِسْمِ ٱللَّهِ ٱلرَّحْمَٰنِ ٱلرَّحِيمِ

Segala puji dan syukur, kehadirat Allah Subhanahu wa ta'ala yang telah memberikan rahmat dan hidayah-Nya sehingga penulis dapat menyelesaikan Tugas Akhir yang berjudul "*Rancang Bangun Fall Detection System* Dengan Analisa Percepatan Sudut Secara Real Time Menggunakan Mikrokontroller Arduino".

Pengerjaan Tugas Akhir ini adalah momen bagi penulis untuk mengeluarkan seluruh kemampuan, hasrat, dan keinginan yang terpendam di dalam hati mulai dari masuk kuliah hingga lulus sekarang ini, lebih tepatnya di jurusan Teknik Informatika Institut Teknologi Sepuluh Nopember Surabaya.

Dalam pelaksanaan dan pembuatan Tugas Akhir ini tentunya sangat banyak bantuan yang penulis terima dari berbagai pihak. Melalui lembar ini, penulis ingin secara khusus menyampaikan ucapan terima kasih kepada:

1. Allah SWT yang telah melimpahkan rahmat, hidayah, dan inayah-Nya sehingga penulis mampu menyelesaikan Tugas Akhir dengan baik.
2. Junjungan kita Nabi Muhammad SAW yang telah menjadi inspirasi, contoh yang baik bagi penulis sehingga tetap termotivasi dalam mengerjakan Tugas Akhir.
3. Kedua orang tua penulis yang telah mencurahkan kasih sayang, perhatian, dan doa kepada penulis.
4. Waskitho Wibisono, S.Kom., M.Eng., Ph.D. dan Ibu Henning Titi Ciptaningtyas, S.Kom., M.Kom. selaku dosen pembimbing yang senantiasa memberikan arahan.
5. Ibu Chastine Fatichah S.Kom., M.Kom., selaku dosen wali penulis. Terima kasih atas dukungan, nasihat, dan pembelajaran hidup yang diberikan kepada penulis selama penulis menjalani kuliah di jurusan Teknik Informatika ITS
6. Bapak Rully Soelaiman, S.Kom., M.Kom, yang selalu memberikan nasihat, dukungan, pembelajaran hidup,

petuah, dan motivasi kepada penulis selama menjalani kuliah di jurusan Teknik Informatika ITS.

7. Kakanda Rahajeng Bellinda Nastiti, Terima kasih atas perhatian dan dukungan serta nasihat-nasihat yang diberikan kepada penulis.
8. Kerabat dan sahabat kami alm. Ferry Irawanto dan alm. Hafidh Azmi yang selalu kami cintai dan memberi kami banyak inspirasi dan mengenal arti dari kekeluargaan.
9. Sahabat Penulis sejak berjuang di jurusan Teknik Informatika ITS, Hafieludin Yusuf Rizana, Ardhana Praharsana, Soffi Izza Sabila, Dwi Oktafiya Sumadya dan Nisrina Madjid yang selalu memberi nasihat dan dukungan satu sama lain, serta membawa keceriaan kepada penulis selama empat tahun ini.
10. Ahmad Fauzan Mufid, Putu Adhi Purwanto, Regin Iqbal, Zola Mahendra, Yarjuna Rohmat, Rifas Filqadar, Muhammad Iqbal Tanjung, Fandy Ahmad dan Yusuf Nugoroho yang selalu memotivasi penulis untuk terus berjuang menyelesaikan tugas akhir ini.
11. Teman-teman lab NCC (Net Cet Centric) dan Administrator Hanif Sudira, Ade Ilham Fajri dan Ghulam Fajri yang mendukung dalam memperjuangkan Tugas Akhir.
12. Angota Pamor Gen 2, Gen 3 dan Gen 4, Said Al'musayyab, Yarjuna Rohmat, Alief Yoga, Arif Fathur Mahmuda, Ardhya Putra, Magista Bella, Nurhamidah Tyas, Hania Maghfira, Purina Qurota, Nahda Fauziyah, Ilyas dan Bayu Aji yang telah berbagi pengalaman dan bersama-sama saling membantu selama dalam perjalanan maupun kehidupan sehari-hari.
13. Pengurus Kabinet Harmoni Merangkai Kreasi BEM FTIf, Angeriko Aryasena, Nabila Tsuraya, Ikrom Aulia, Reva Yoga, Arbiantoro Mas, Achmad Saiful, Claudio Denta, Ariesa Putri, Mona Syahmi, Rahma Permatasari, Rizki Amalia, Fitria dan Iqbal Ramadhan atas kerja sama dan merealisasikan target serta cita-cita untuk FTIf lebih baik.

14. Keluarga Kecil External Affairs BEM FTIf, Ariesa Putri, Ahmad Fajar, Rizki Nugaraha, Nanda Puji, Zikrul ihsan, Zetry Prawira, Daniel Surya Anjas, Ken Genesius, Edo Haidar, Denny Angga, Rani Oktavia, Sarah Putri, Hemas Maselva Putri, Ervi Ritya. Kebanggan yang luar biasa bagi penulis bisa bertemu, berbagi keceriaan serta merealisasikan target-target bersama, Kalian semua hebat senang bisa bertemu dengan kalian.
15. Keluarga Event Organizer Yess Summit 2014, Andre Surya, Rara dennira, Adnan M. Fajri, Deri Putra, Hafisyah, Achmad Fajar, Indira Wido, Rizky Amalia, Afra Basyirah, terima kasih atas kerjasama dan kesempatan yang telah diberikan kepada penulis untuk bisa berkontribusi dalam penyelenggaran Yess Summit 2014.
16. Sahabat Supercamp 2012 Nadhila Adhani, Sarah Aufa Dhita, Sabila Ghassani, Aldrin Wiguna, Adit, Ardy Jumardi, Bagas Rizki, Widya Farah, Haidar Romzi, Mayfitriani, Otniel YL, Omar basalamah, Retna Ningtyas, Ulfa, Aryo Faruq, Ari Matea, Fadlan Fitrah dan Yusuf Ibrahim, terima kasih telah berbagi pengalaman dan menudukung satu-sama lain dari sebelum mendapatkan almamater hingga mengarungi dunia perkuliahan, semoga silaturahmi kita tetap terjaga, “See you on top”.
17. Sahabat SMP Xaverius dan SMA Xaverius, Syarief Hidayat, Rangga Yudhio, Rizki Nugrahayu, Danu Akbar, Nopriansyah, Zairi Reling Utomo, Ivo Maria, Moses Juni, Thomas Wiga, Khatrin, Shelia Restu, Septa Linda, Adijaya, yang selalu memberikan nasihat dan dukungan satu sama lain. Terima kasih, semoga kita tetap terus menjaga silaturahmi dan sukses di tempat kerja masing-masing.
18. Seluruh teman Teknik Informatika ITS angkatan 2012 yang telah menemani dan memberi pengalaman berharga bagi penulis sejak maba sampai lulus.
19. Teman-teman angkatan 2011, 2013, 2014, 2015 yang sudah memberikan pengalaman selama kuliah di TC ini.

20. Juga tidak lupa kepada semua pihak yang belum sempat disebutkan satu per satu yang telah membantu terselesaikannya Tugas Akhir ini, penulis mengucapkan terima kasih.

Penulis telah berusaha sebaik mungkin dalam menyusun Tugas Akhir ini, namun penulis mohon maaf apabila terdapat kekurangan, kesalahan maupun kelalaian yang telah penulis lakukan. Kritik dan saran yang membangun dapat disampaikan sebagai bahan perbaikan selanjutnya.

Surabaya, 17 Juni 2016
Penulis

M. Satrio Ramadhana Yudhono

DAFTAR ISI

DAFTAR GAMBAR

DAFTAR TABEL

(Halaman ini sengaja dikosongkan)

DAFTAR KODE SUMBER

(Halaman ini sengaja dikosongkan)

BAB I
PENDAHULUAN

Pada bab ini akan dipaparkan mengenai garis besar tugas akhir yang meliputi latar belakang, tujuan, rumusan masalah, batasan masalah, metodologi pembuatan tugas akhir dan sistematika penulisan buku tugas akhir.

1.1 Latar Belakang

Insiden jatuh merupakan masalah dan sering terjadi di lingkungan sekitar, pada umumnya terjadi pada usia lanjut namun tidak menutup kemungkinan insiden terjatuh terjadi pada siapa pun dan tidak mengenal usia. Pada sebagian kalangan hal itu dianggap biasa dan tidak perlu dikhawatirkan, namun untuk kalangan tertentu yaitu pasien penderita stroke ataupun seseorang yang mempunyai indikasi Stroke ini sangat berbahaya, selain itu yang sedang mengalami gangguan kesehatan atau penuaan umur juga diperlukan perhatian khusus. menurut data dari *World Health Organization* (WHO) pada tahun 2012 menyebutkan sejumlah 420.000 terjadi insiden terjatuh fatal yang menyebabkan kematian pada setiap tahunnya dan terjadi insiden terjatuh dengan kategori berat sejumlah 37,3 sehingga dibutuhkan perhatian medis di setiap tahun [1]. Selain itu seseorang yang berumur 65 tahun atau lebih paling tidak pernah mengalami insiden terjatuh minimal satu kali selama kurun waktu 1 tahun terakhir, dan 2/3 orang tersebut mengalami insiden terjatuh kembali dalam periode waktu 2 tahun [2]. Dampak yang dapat terjadi selain kematian antara lain memar, dislokasi sendi patah tulang dan trauma. Kebanyakan usia lanjut yang terjatuh yang tidak mampu untuk bangkit sendiri tergeletak dilantai dalam waktu lebih dari 1 jam meninggal dalam waktu 6 bulan setelah insiden tersebut [3].

Tingginya angka kematian yang terjadi akibat insiden terjatuh akibat dari beberapa faktor antara lain: tidak adanya pertolongan pertama karena posisi korban yang jauh dari keramaian dan tidak diketahui keberadaannya, serta korban yang sulit untuk menghubungi keluarganya dan lainnya.

Berdasarkan latar belakang tersebut penulis menawarkan sebuah solusi untuk membuat rancang bangun *fall detection system* menggunakan mikrokontroller Arduino dengan menggunakan metode analisis pergerakan sudut rotasi menggunkan nilai *quaternion* serta menggunakan SIM908 GSM/GPRS/GPS *Shield*. Sensor *accelerometer* akan mengambil data-data percepatan linier secara *real-time*. pendeteksian percepatan tersebut berdasarkan pada 3 sumbu yaitu sumbu x, sumbu y dan sumbu z [4], dan nilai *quaternion* akan menghasilkan sudut pergerakan rotasi saat terjadi transisi gerakan dengan menggunakan sensor orientasi dapat menentukan apakah perubahan gerakan yang terjadi merupakan perubahan yang mengindikasikan gerakan posisi jatuh. selain itu diperlukan *Shield* GPRS dan GPS untuk melanjutkan data ke server dan mengirim notifikasi berupa SMS ke kerabat korban beserta lokasi tempat di mana korban tergeletak.

1.2 Rumusan Masalah

Rumusan masalah yang diangkat dalam Tugas Akhir ini adalah sebagai berikut:

1. Bagaimana cara sistem ini mendeteksi seseorang yang sedang mengalami insiden terjatuh mengunakan nilai *quaternion*?
2. Bagaimana membuat sistem yang dapat mengirimkan informasi berupa SMS kepada keluarga korban bahwa telah terjadi insiden terjatuh?
3. Berapa besar tingkat akurasi dari sistem ini?

1.3 Batasan Masalah

Permasalahan yang dibahas dalam Tugas Akhir ini memiliki beberapa batasan, di antaranya sebagai berikut:

1. Sistem sensor diletakan pada saku pengguna.
2. Data yang dikirimkan ke server merupakan data koordinat GPS.
3. Aplikasi server berupa aplikasi basis data menggunakan teknologi MySQL.
4. Data dikirimkan ke server ketika telah terjadi insiden jatuh

5. Koneksi internet harus stabil.

1.4 Tujuan

Tujuan dari pembuatan Tugas Akhir ini adalah:

1. Membuat sistem yang dapat mendeteksi insiden jatuh.
2. Mengirimkan notifikasi kepada penerima pesan.

1.5 Manfaat

Manfaat dari tugas akhir ini adalah membuat sistem yang dapat mendeteksi insiden jatuh dan mengirimkan notifikasi berupa informasi telah mengalami insiden jatuh dan posisi pada saat jatuh kepada pihak medis maupun keluarga korban sehingga penyelamatan pihak medis dapat segera dilaksanakan.

1.6 Metodologi

Adapun langkah-langkah yang ditempuh dalam pengerjaan penelitian ini adalah sebagai berikut:

1. Penyusunan Proposal Penelitian

Pada tahap awal penyusunan proposal, penulis memulai pengerjaan Tugas Akhir dengan melakukan penyusunan proposal Tugas Akhir. Pada proposal tersebut berisi mengenai mekanisme sistem bekerja secara adaptif dalam mengendalikan penggunaan air.

2. Studi Literator

Tahap ini adalah tahap pengumpulan informasi dan pembelajaran materi yang akan digunakan dalam pengerjaan Tugas Akhir. Karena pada pengerjaan Tugas akhir dibutuhkan berbagai komponen perangkat keras, maka penulis melakukan studi literator mengenai penggunaan perangkat keras tersebut. Studi literator yang termasuk diskusi dan pemahaman mengenai topik Tugas Akhir adalah sebagai berikut:

1. Mekanisme mikrokontroler Arduino.
2. Penerimaan dan pengiriman data dari server.
3. Penerimaan dan pengiriman data dari mikrokontroler Arduino ke server.
4. Pengiriman notifikasi berupa *Short Messages Services* (SMS) dari Arduino ke Handphone.
5. Pembacaan data pada sensor orientasi 9 axis motion Shield.
6. Proses mikrokontroler dapat melakukan koneksi GPRS melalui GSM *Shield* SIM908.

3. Perancangan Sistem

Pada tahap ini dilakukan analisis terhadap sistem serta perancangan sistem yang akan dibuat. Hal ini ditujukan untuk dapat merumuskan solusi yang data ditempuh untuk melakukan implementasi pada sistem yang akan dibuat serta kemungkinan yang dapat terjadi ketika implementasi sistem berlangsung.

4. Implementasi Perangkat Lunak

Untuk implementasi sistem pemantauan penggunaan air ini, implementasi perangkat lunak pada penelitian ini adalah sebagai berikut:

- Implementasi pada server
  Pada implementasi perangkat lunak ada sisi server menggunakan bahasa pemrograman PHP dan basis data menggunakan MySQL untuk menyimpan *log* aktivitas dari sensor.
- Sensor dan kontroler
  Kontroler adalah pelaksana dari perintah mikrokontroler Arduino, sedangkan sensor adalah pengambil data dari apa yang terjadi di lingkungannya. Keduanya adalah perangkat *input* dan *output* bagi Arduino Mega. Arduino diprogram dengan bahasa pemrograman Processing yang

merupakan kombinasi bahasa pemrograman C++ dan Java. Untuk menuliskan logika-logika dari skema rangkaian yang terhubung dengan perangkat keras Arduino Mega, dibutuhkan Arduino IDE.

5. Pengujian dan Evaluasi

Sistem akan diuji setelah selesai diimplementasikan menggunakan skenario yang sudah dipersiapkan. Pengujian dan evaluasi diperlukan untuk mengetahui kesalahan yang terjadi dan memperbaiki kesalahan tersebut.

6. Penyusunan Buku Penelitian

Pada tahap ini disusun laporan penelitian sebagai dokumentasi pelaksanaan penelitian, yang mencakup seluruh konsep, teori, implementasi, serta hasil yang telah dikerjakan.

1.7 Sistematika Penulisan

Buku Tugas Akhir ini terdiri dari beberapa bab, yang dijelaskan sebagai berikut:

BAB I PENDAHULUAN
Bab ini berisi latar belakang masalah, rumusan dan batasan permasalahan, tujuan dan manfaat pembuatan Tugas Akhir, metodologi yang digunakan, dan sistematika penyusunan Tugas Akhir.

BAB II TINJAUAN PUSTAKA
Bab ini membahas dasar pembuatan dan beberapa teori penunjang yang berhubungan dengan pokok pembahasan yang mendasari pembuatan Tugas Akhir ini.

BAB III ANALISIS DAN PERANCANGAN
Bab ini membahas analisis dari sistem yang dibuat meliputi analisis permasalahan, deskripsi umum perangkat lunak, spesifikasi

kebutuhan, dan identifikasi pengguna. Kemudian membahas rancangan dari sistem yang dibuat meliputi rancangan skenario kasus penggunaan, arsitektur, data, dan antarmuka.

BAB IV IMPLEMENTASI
Bab ini membahas implementasi dari rancangan sistem yang dilakukan pada tahap perancangan.

BAB V PENGUJIAN DAN EVALUASI
Bab ini membahas pengujian dari aplikasi yang dibuat dengan melihat keluaran yang dihasilkan oleh aplikasi dan evaluasi untuk mengetahui kemampuan aplikasi.

BAB VI PENUTUP
Bab ini berisi kesimpulan dari hasil pengujian yang dilakukan serta saran untuk pengembangan aplikasi selanjutnya.

BAB II
TINJAUAN PUSTAKA

Pada bab ini berisi penjelasan pustaka atau teori yang menjadi dasar pembuatan tugas akhir ini. Teori-teori tersebut diantaranya algoritma *fall detection* dengan menggunakan *threshold* dan dengan menggunakan metode KNN serta teori pendukung lainnya.

2.1 Arduino Mega 2560

Arduino Mega yang ditunjukkan pada Gambar 2.1 adalah salah satu jenis mikrokontroler berbasis ATmega 2560. Mikrokontroller ini menggunakan bahasa pemrograman *Processing* yang mengkombinasikan bahasa pemrograman C++ dan Java, sehingga penggunanya akan lebih mudah jika telah terbiasa dengan menggunakan kedua bahasa pemrograman tersebut.

Gambar 2.1 Arduino Mega

Mikrokontroler	ATmega2560
Operating Voltage	5 V
Input Voltage (recommended)	7-12 V
Input Voltage (limits)	6-20 V
Digital I/O Pins	54 (6 pin merupakan *Pulse Width Modulation output*)
Analag Input Pins	16
DC *Current* per I/O Pin	20 mA
DC *Current* for 3.3V Pin	50 mA
Flash Memory	256 KB (ATmega2560) dimana 8 KB digunakan oleh bootloader
SRAM	8 KB
EEPROM	4 KB
Clock Speed	16 MHz
Dimensi	101.5 mm x 53.4 mm

Tabel 2.1 Arduino Data Sheet

Arduino Mega 2560 memiliki pin I/O yang cukup banyak, sejumlah 54 buah digital I/O pin (15 pin diantara adalah PWM), 16 pin analog input, 4 pin UART (*serial port hardware*). Arduino Mega 2560 dilengkapi dengan sebuah oscillator 16 Mhz, sebuah port USB, *power jack* DC, ICSP *header*, dan tombol reset. Board ini sudah sangat lengkap, sudah memiliki segala sesuatu yang dibuthkan untuk sebuah mikrokontroller. Dengan penggunaan yang cukup sederhana, anda tinggal menghubungkan power dari USB ke PC anda atau melalui adaptor AC/DC ke jack DC. Spesialisasi pin yang terdapat pada Arduino Mega 2560 adalah sebagai berikut:

- Serial: memiliki 4 serial yang masing-masing terdiri dari 2 pin. Serial 0 pin 0 (RX) dan pin 1 (TX). Serial 1 pin 19 (RX) dan pin 18 (TX). Serial 2 pin 17 (RX) dan pin 16 (TX). Serial 3 pin 15 (RX) dan pin 14 (TX). RX digunakan untuk menerima dan TX untuk transmit data serial TTL. Pin 0 dan pin 1 adalah pin yang digunakan oleh chip USB-to-TTL ATmega16U2.
- *External Interrupt*: 0 (pin 2), 1 (pin 3), 5 (pin 18), 4 (pin 19), 3 (pin 20), 2 (pin 21). Arduino memiliki kemampuan *interrupt* yang baik, sehingga dapat disisipkan sebuah fungsi *interrupt* untuk pin yang tersedia. Secara langsung dapat menyebutkan nomor pin pada fungsi attachInterrupt().
- PWM: Pin 2 hingga pin 13 dan pin 44 hinga pin 46 menyediakan 8-bit *Pulse Width Modulation* (PWM) *output* dengan pemanggilan fungsi analogWrite().
- SPI: Pin 53(SS), 51(MOSI), 50(MISO),52(SCK) mendukung komunikasi *Serial Peripheral Interface*.
- LED:13 digunakan untuk tanda LED pemasangan Arduino yang sedang terkoneksi, ketika pin memiliki nilai tinggi akan menyala, sedangkan ketika pin memiliki nilai rendah, LED akan mati.
- Arduino memiliki 16 pin yaitu A0 – A15, masing-masing menyediakan 10 bit sebagai analog *input*.

Arduino Mega yang memiliki detail spesifikasi seperti pada Tabel 2.1, digunakan sebagai mikrokontroler pada sistem *fall detection* agar berjalan dengan baik [5].

2.2 SIM908 GSM/GPRS/GPS *Shield* V3.0

GPS/GSM/GPRS *Shield* versi 3.0 merupakan *Shield* produksi DFRobot. *Shield* ini mendukung frekuensi Quad-band GSM/GPRS pada 850 MHz, 900 MHz, 1800 MHz, dan 1900 MHz. *Shield* ini juga dilengkapi dengan antena GPS untuk keperluan navigasi satelit sehingga memungkinkan sebuah robot atau sistem mengirim data lokasi melalui jaringan GSM. *Shield* ini juga mendukung format data GPS dengan protokol NMEA [6].

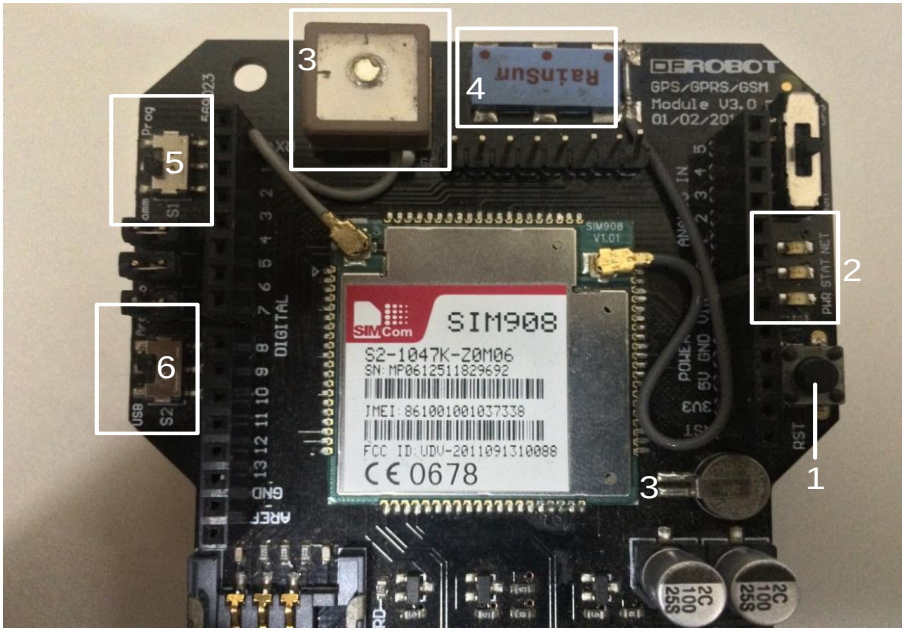
Dalam penggunaan *shield* ini dibutuhkan *command* khusus yang disebut AT *command*. AT *command* akan diprogram dan *shield* akan membaca perintah sesuai dengan *command* dalam program. AT *command* sendiri memiliki format penulisan yang sudah diatur dari pabrik pembuat *shield*. Beberapa contoh AT *command* yang digunakan pada sistem ini dapat dilihat pada Tabel 2.2 [7].

Seperti yang ditunjukkan pada Gambar 2.2 dengan keterangan gambar yang ditunjukkan pada Tabel 2.3 ada bagian-bagian yang perlu diperhatikan saat menggunakan SIM908 *Shield* yaitu sebelum mengunggah program ke arduino dengan tambahan *Shield* ini *Switch* S1 harus berada pada mode "prog" dan *Switch* S2 berada pada mode "arduino", setelah berhasil mengunggah program mode pada *Switch* S1 dirubah ke mode "comm" setelah itu menekan tombol "RST" sampai semua lampu LED menyala.

Pada penelitian ini *GPS/GPRS/GSM* SIM908 *Shield V3.0* digunakan untuk mengirimkan koordinat *latitude dan longitude* lokasi terjadinya insiden terjatuh, baik mengirimkan data-data tersebut ke server dengan GPRS maupun mengirimkan SMS kepada kerabat dengan menggunakan GSM. Waktu yang dibutuhkan *GPS/GPRS/GSM Shield V3.0* untuk mengirimkan data tersebut ke server maupun mengirimkan SMS adalah masing-masing 3 detik.

Tabel 2.2 Perintah *AT Command pada GPS/GPRS/GSM Shield V3.0*

No.	AT Command	Keterangan
1	AT+CGPSPWR	Menyalakan GPS
2	AT+CGPSRST	Mengatur Mode GPS
3	AT+CGPSINF	Mendapatkan informasi lokasi
4	AT+CGREG=?	Mengecek status jaringan
5	AT+HTTPINIT	Inisiasi mode HTTP
6	AT+HTTPTERM	Menyudahi mode HTTP
7	AT+HTTPPARAM	Mengatur URL yang akan dikunjungi
8	AT+CMGS	Mengirimkan SMS



Gambar 2.2 SIM908 *Shield*

Tabel 2.3 Keterangan Gambar SIM908

No. Gambar	Keterangan
1	Tombol Rst
2	LED (power supply, network states & working status)
3	Antena GPS High-gain
4	Antena GSM High-gain
5	Switch 1 (Comm/ Prog)
6	Switch 2 (USB/ Arduino)

2.3 *Global Positioning System* (GPS)

GPS (*Global Positioning System*) adalah sebuah sistem navigasi berbasiskan radio yang menyediakan informasi koordinat posisi, kecepatan, dan waktu kepada pengguna di seluruh dunia. Jasa penggunaan satelit GPS tidak dikenakan biaya. Pengguna hanya membutuhkan GPS *receiver* untuk dapat mengetahui koordinat lokasi. Keakuratan koordinat lokasi tergantung pada tipe GPS *receiver* .

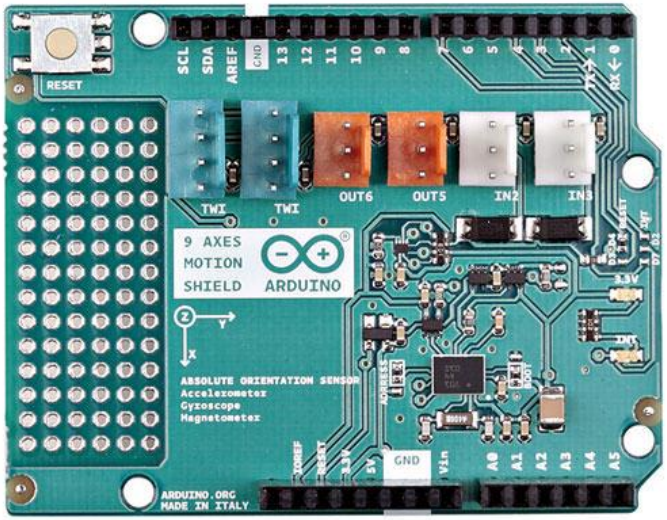
GPS terdiri dari tiga bagian yaitu satelit yang mengorbit bumi (Satelit GPS mengelilingi bumi 2x sehari), stasiun pengendali dan pemantau di bumi, dan GPS *receiver* (alat penerima GPS). Satelit GPS dikelola oleh Amerika Serikat. Alat penerima GPS inilah yang dipakai oleh pengguna untuk melihat koordinat posisi. Selain itu GPS juga berfungsi untuk menentukan waktu.

Format data keluaran GPS ditetapkan oleh NMEA (*National Marine Electronics Assosiation*). Data keluaran dalam format NMEA berbentuk kalimat (*string*) yang merupakan rangkaian karakter ASCII 8 bit. Setiap kalimat diawali dengan satu karakter '$', dua karakter *talker ID*, tiga karakter *sentence ID* dan diikuti oleh *data fields* yang masing-masing dipisahkan oleh koma serta diakhiri oleh *optional checksum*. Contoh data GPS yang digunakan dalam penelitian seperti '$GPGGA,0.276513,112.791692,1,09 Penjelasan dari contoh data tersebut adalah sebagai berikut.

- 'GPGGA' = *sentence identifier*.
- '0.276513' = *latitude.*
- '112.791692' = *longitude.*
- '1' = posisi (0 = *invalid*, 1 = *valid*).
- '09' = jumlah satelit.

Pada penilitian ini GPS digunakan untuk mengetahui posisi dimana terjadinya insiden terjatuh, GPS akan memberikan informasi berupa koordinat *latitude* dan *longitude* dengan toleransi keakuratan posisi yang berbeda dengan lokasi yang sebenarnya maksimal 10 M [8].

2.4 Arduino 9 Axes Motion Shield



Gambar 2.3 9 Axis Motion Sensor Shield

9 Axes Motion Shield seperti yang ditunjukkan pada Gambar 2.3 merupakan shield dengan kemampuan untuk mendeteksi orientasi gerakan yang diproduksi oleh Bosch Sensortec GmbH, di dalam Shield ini terdapat sensor BNO055 di mana terintegrasi dengan pendeteksian 3 sumbu x,y,z *accelerometer* 14-*bit* dengan cakupan tingkat sensitivitas ±2g, ±4g, ±8g hingga ±16g. Di mana tiap 1g merupakan satu satuan percepatan rata-rata gravitasi bumi yaitu 9,8m/s², pendeteksian 3 sumbu x,y,z *gyroscope* 16z-*bit* dengan maksimal *range* ± 2000 *degrees*, serta pendeteksian t*riaxial geomagnetic 32-bit* dengan tingkat sensitivitas ±1300µT pada sumbu x-, y- dan ±2500µT pada sumbu z. Memiliki antarmuka I2C. Sensor ini memiliki fitur pendeteksian akselerasi pada tiga dimensi, dan mendapatkan nilai pada sensor *fusion* (*accelerometer, gyroscope, magnetometer*) seperti *quaternion, euler angle, rotation vector,*

Linier acceleration, gravity vector. Daya yang diperlukan untuk *Shield* ini adalah 5/3.3 V 10mA [9].

Dalam penelitian ini 9 *Axis Motion Shield* digunakan untuk mendeteksi adanya pergerakan transisi fisik *user* menggunakan *accelerometer* yang ada di dalam sensor tersebut dan mencari nilai *quaternion* untuk menentukan besarnya sudut rotasi pada pergerakan fisik *user*.

Untuk mempresentasikan nilai *accelerometer* x,y,z pada sensor dan nilai *quaternion* (q_0, q_1, q_2, q_3) pada penelitian ini menggunakan *library* dari *Adafruit* yaitu Library Adafruit_BNO055.

2.5 Arduino IDE

Arduino IDE terdiri dari *text editor* untuk menulis kode Arduino, area pesan, *console text*, *toolbar* dengan tombol untuk fungsi secara umum digunakan pada kode Arduino, dan beberapa menu. Arduino IDE yang ditunjukkan pada ini akan terkoneksi dengan perangkat keras Arduino ketika Arduino disambungkan dengan kabel USB untuk mengunggah program dan saling berkomunikasi. Program untuk Arduino ditulis pada sketch. Sketch yang disimpan dalam bentuk file .ino.

Terdapat pula *console* untuk menampilkan pesan dan informasi *error* kompilasi program. Pada toolbar terdapat beberapa tombol berikut:

- *Verify* untuk melakukan cek error pada kode program.
- *Upload* untuk kompilasi kode program dan mengunggah kode program ke hardware Arduino.
- *New* untuk membuat sketch baru.
- *Open* untuk membuka kode program yang sudah tersimpan sebelumnya.
- *Save* untuk menyimpan kode program. dan
- *Open serial monitor* untuk membuka serial monitor Arduino.

Pada penelitian ini digunakan arduino IDE versi terbaru yaitu Arduino 1.6.9. selain untuk menuliskan kode sumber aplikasi sistem, di Arduino IDE jugalah tempat untuk mengunggah kode tersebut agar berjalan di mikrokontroller Arduino [10].

2.6 *Activity Recognition*

Activity recognition merupakan teknik yang digunakan untuk proses pendeteksian aktivitas fisik *user*. Adapun tahap-tahap dari *activity recognition* dapat dijelaskan sebagai berikut [11]:

2.6.1 *Data Collection*

Untuk melakukan sebuah identifikasi aktivitas dikatakan jatuh atau tidak, diperlukan data yang nantinya akan digunakan untuk pengolahan. Pada umumnya data diambil dengan menggunakan sebuah *device* berupa sensor. Data ini diharapkan dapat digunakan untuk dilakukan proses identifikasi yang bertujuan untuk menggambarkan sebuah entitas tertentu.

2.6.2 *Feature Extraction*

Setelah memperoleh data dari sensor *accelerometer*, proses selanjutnya adalah ekstraksi data. Data yang akan diproses terlebih dahulu diolah sedemikian hingga agar data tersebut tidak terlalu konvergen atau terlalu divergen untuk dilakukan proses pendeteksian menggunakan algoritma *fall detection*. Salah satunya menggunakan teknik *data sampling*.

2.6.3 *Data Interpretation*

Hal ini merupakan tahap paling penting dalam proses identifikasi aktivitas *user*. Setelah memiliki data yang sudah diekstraksi maka langkah selanjutnya adalah melakukan pendeteksian dengan menggunakan algoritma *fall detection.*

2.7 Data Training

Data *training* merupakan sekumpulan data yang biasanya disajikan dalam bentuk tabel yang memuat informasi tertentu. Informasi tersebut digunakan untuk mencari hubungan tertentu dari suatu data *training* [12].

2.7.1 Data *Training Nilai Accelerometer*

Data *training* pada *accelerometer* berisi variabel-variabel yang terdiri atas sumbu *x, y, z* nilai *alpha* (α).

2.7.2 Data *Training Nilai Tetha*

Data *training* nilai tetha yang diambil dari perhitungan nilai *quaternion*.menggunakan *9 axis Motion Shield* .

2.7.3 Data Training Nilai *Quaternion*

Data *training quaternion* berisi variabel-variabel yang terdiri atas nilai – nilai *q0,q1,q2,q3.*

2.8 Algoritma *Fall Detection*

Algoritma *fall detection* merupakan algoritma pendeteksi pergerakan fisik jatuh yang menggunakan sensor *accelerometer* dan mengitung sudut rotasi pergerakan menggunakan nilai *quaternion* Pada algoritma ini dilakukan pembagian dua aktivitas, yaitu statis dan dinamis. Algoritma ini dibagi menjadi 3 langkah, yaitu analisis terjadinya transisi gerakan, analisis aktivitas, dan analisis postur.

Terdapat beberapa langkah untuk mendeteksi jatuh dengan menggunakan algoritma *fall detection,* yaitu penghitungan nilai skalar dari sensor *accelerometer*, penghitungan nilai tetha dari nilai *quatrnino*, dan pencarian nilai *threshold* dari nilai *alpha* (α) [13].

2.8.1 Penghitungan Nilai Alpha

Penghitungan nilai *alpha* (α) didapat dari kuadrat dari penjumlahan sumbu *x, y,* dan *z* dari sensor *accelerometer* seperti Persamaan 2.1 [5].

$$\alpha = \sqrt{{a_x}^2 + {a_y}^2 + {a_Z}^2} \quad (1)$$

Nilai *alpha* (α) nantinya akan digunakan pada algoritma *fall detection* untuk mendeteksi transisi pergerakan fisik statis menuju pergerakan fisik dinamis, nilai *alpha* (α) digunakan untuk penentuan nilai *threshold,* kemudian akan dibandingkan antara data nilai *alpha* (α) *realtime* dengan *alpha* (α) *threshold* [13].

2.8.2 Penghitungan Nilai Perubahan Percepatan X,Y,Z

Selain dengan nilai alpha (α), untuk mengetahui adanya perubahan percepatan yang dilakukan ketika terjadi transisi gerakan. Dapat dilakukan dengan menghitung perubahan nilai percepatan pada masing-masing sumbu X , Y , Z untuk menghitung perubahan percepatan pada sumbu X dapat dilakukan dengan cara sebagai berikut: $d(x_i, x_{i+1}) = (x_{i+1} - x_i)$.

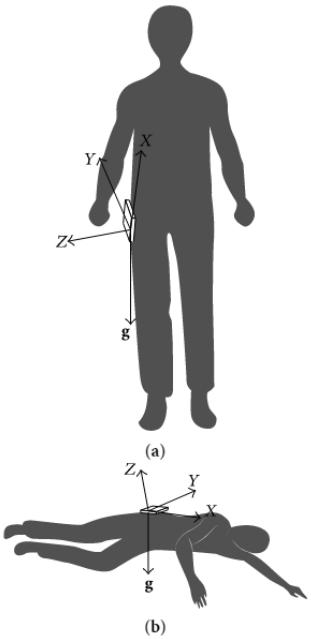
2.8.3 Penghitungan Nilai Theta

Penghitungan nilai *theta(θ)* didapat dari *arctan* dari nilai *quaternion* Q (q_0, q_1, q_2, q_3). Quaternion digunakan untuk mengambarkan sudut rotasi pada dimensi 3 ruang, Quaternion bersifat menyatu yang bebas dari Gimbal Lock yang merupakan kelemahan dari teknik sudut euler. Nilai *theta* (θ) digunakan pada algoritma *fall detection* untuk mendeteksi postur *user*, apakah postur *user* termasuk dalam kondisi berbaring atau tidak [13].

$$\theta = (2.\arctan\left(\frac{\sqrt{{q_1}^2 + {q_2}^2 + {q_3}^2}}{q_0}\right) * \frac{180}{3.14}) \qquad (2)$$

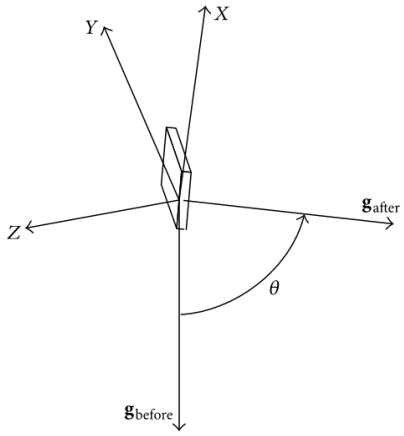
Quaternion adalah cara yang tepat untuk menggambarkan nilai pada transisi gerakan rotasi dalam perubahan gaya ADL (*Activity Daily Living*) seperti berjalan, duduk, termasuk dalam insiden terjatuh. Dengan *quaternion* dapat menggambarkan sudut rotasi ketika sebelum dan sesudah jatuh yang tidak bisa digambarkan dengan teknik mencari sudut dengan *euler* karena ada kemungkinan terjadinya *gimbal lock*. Gambar yang ditunjukkan oleh Gambar 2.4 menunjukkan perubahan arah koordinat sumbu x,y,z dan g dari sensor

accelerometer. Terhadap g, sedangkan Gambar 2.5 menunjukkan adanya perubahan nilai sudut tetha (θ) saat terjadi peristiwa jatuh, nilai sudut tetha (θ) inilah yang digunakan sebagai analisa pendeteksian gerakan jatuh [13].

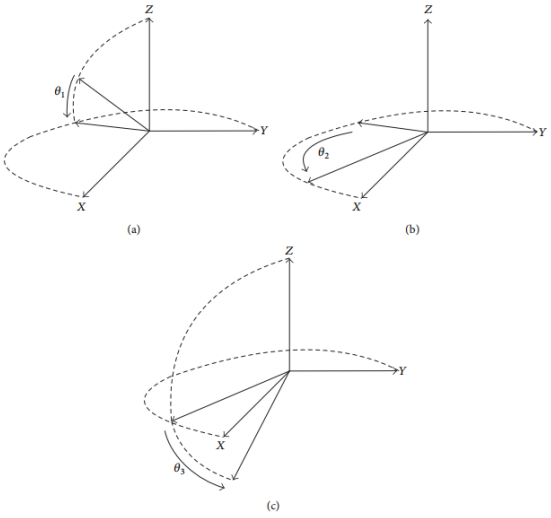


Gambar 2.4 perubahan koordinat sumbu x,y,z dan g (a) sebelum jatuh (b) sesudah jatuh

Pada Gambar 2.6 merupakan dekomposisi *quaternion* q_1, q_2, q_3, pada q_1 merupakan sudut rotasi berdasarkan sumbu Y, q_2 merupakan sudut rotasi berdasarkan sumbu Z dan q_3 merupakan sudut rotasi berdasarkan sumbu X. Untuk mendapatkan nilai *quaternion* Q (q_0, q_1, q_2, q_3) menggunakan *library Adafruit BNO055*.



Gambar 2.5 perubahan sudut theta sebelum dan sesudah terjadinya jatuh



Gambar 2.6 Dekomposisi dari sudut rotasi berdasarkan Quaternion. (a) Quaternion q_1 (b) Quaternion q_2 (c) Quaternion q_3

2.8.4 *Decision Tree*

Decision Tree adalah sebuah struktur pohon, dimana setiap node pohon merepresentasikan atribut yang telah diuji, setiap cabang merupakan suatu pembagian hasil uji, dan node daun (leaf) merepresentasikan kelompok kelas tertentu. Level node teratas dari sebuah Decision Tree adalah node akar (root) yang biasanya berupa atribut yang paling memiliki pengaruh terbesar pada suatu kelas tertentu. Pada umumnya Decision Tree melakukan strategi pencarian secara top-down untuk solusinya..

Untuk menentukkan nilai-nilai yang digunakan pada keputusan, maka dibandingkan antara data *training* sensor *accelerometer* saat kondisi *user* dalam keadaan statis dengan *user* saat mengalami transisi insiden terjatuh. Saat terjadi transisi pergerakan fisik pada gerakan jatuh biasanya nilai *alpha* (α) yang berasal dari akar dari penjumlahan sumbu *x,y,z* sensor *accelerometer* akan menghasilkan nilai yang maksimum, nilai *alpha* (α) maksimum inilah yang menjadi prediksi awal untuk mendeteksi gerakan jatuh. Jika selisih nilai *alpha* (α) *saat* ini dan sebelumnya tidak mendekati angka 0 maka dipastikan terjadi transisi pergerakan dari pergerakan fisik yang statis menuju pergerakan dinamis.

Pergerakan statis ialah saat *user* sedang tidak mengalami transisi gerakan secara signifikan misalnya saat duduk atau saat berdiri dan pergerakan dinamis ialah terjadinya transisi pergerakan fisik *user* misalnya dalam keadaan berjalan, aktivitas duduk yaitu mula-mula *user dalam* keadaan berdiri kemudian *user* melakukan aktivitas duduk, dalam hal ini saat terjadi insiden terjatuh, terjadi transisi pergerakan fisik yaitu mula-mula user dalam keadaan berdiri hingga user berbaring di lantai [13].

2.8.5 K Nearest Neighbors

K Nearest Neighbors (k-NN) adalah sebuah metode klasifikasi terhadap sekumpulan data berdasarkan pembelajaran data yang sudah ada (dataset). KNN termasuk dalam *supervised learning*, di mana hasil dari query *instance* yang baru diklasifikasikan berdasarkan mayoritas kedekatan jarak dari kategori yang ada dalam *k*-NN.

Kedekatan dapat dianggap sebagai invers jarak, berbanding terbalik dengan jarak, semakin kecil jarak tersebut maka semakin besar kedekatan dua *instance*. Dengan demikian k Nearest Neighbours dari sebuah *instance* x didefinisikan sebagai k *instance* yang memiliki jarak terkecil (kedekatan terbesar, *nearest*) dengan x. semakin banyak *instance* tersebut memiliki kedekatan terhadap suatu *class* maka diprediksikan klasifikasinya termasuk pada klasifikasi terbanyak dari titik-titik tersebut. Untuk menghitung jarak *instance* digunakan *euclidean distance* yang dirumuskan sebagai berikut [14].

$$d(x_1, x_2) = \sqrt{(x_{11} - x_{21})^2 + (x_{12} - x_{22})^2 + \cdots \left(x_{1p} - x_{2p}\right)^2}$$

$$= \sqrt{\sum_{j=1}^{p} \left(x_{1j} - x_{2j}\right)^2}$$

(Halaman ini sengaja dikosongkan)

BAB III
ANALISIS DAN PERANCANGAN

Bab ini akan menjelaskan mengenai dasar dari perancangan perangkat lunak pada sistem yang akan dibangun pada Tugas Akhir. Perancangan perangkat lunak yang dibahas adalah mengenai deskripsi umum aplikasi, proses perancangan, alur, dan implementasinya.

3.1 Deskripsi Umum Sistem

Pada penelitian ini dibangun sistem *fall detection* secara *real-time* menggunakan mikrokontroler Arduino, GSM/GPS/GPRS *Shield* Arduino untuk mengirimkan data ke server, menentukan lokasi melalui GPS serta dapat mengirimkan sms kepada kerabat korban apabila user mengalami insiden terjatuh.

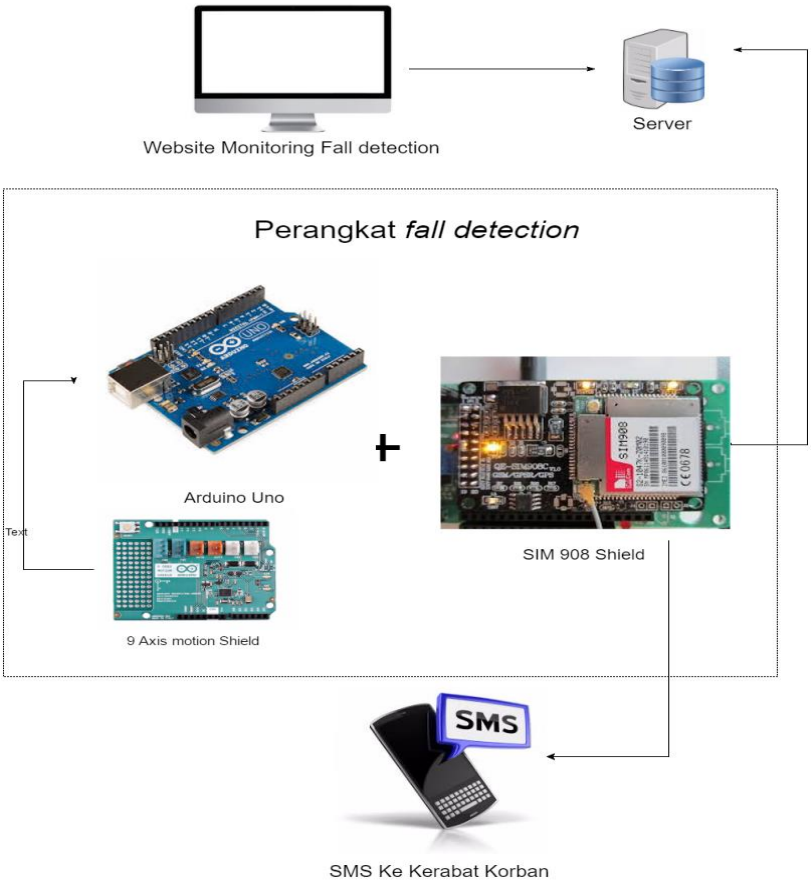
Sistem *fall detection* ini berfungsi untuk memantau aktivitas fisik *user* ketika terjadi insiden terjatuh. Sensor orientasi yang digunakan dalam sistem ini di dalamnya terdapat *accelerometer yang* mampu mendeteksi percepatan linier sumbu x, sumbu y dan sumbu z, kemudian dapat menghitung besarnya sudut rotasi sebelum dan sesudah terjadinya pergerakan fisik. Kemudian dengan menggunakan algoritma *fall detection,* data *real-time* dari sensor *accelerometer* akan dilakukan perhitungan untuk menentukan apakah perubahan gerakan yang terjadi merupakan perubahan yang mengindikasikan gerakan posisi jatuh.

Diharapkan dengan adanya sistem ini pertolongan pertama dapat segera dilakukan terhadap korban, karena sistem ini dapat mengirimkan SMS secara langsung kepada kerabat korban yang berisi koordinat lokasi yang dapat dibuka dengan melalui aplikasi *Google Maps* yang telah terpasang sebelumnya. Pada sistem ini juga dapat menampilkan *history* insiden terjatuh yang terjadi sebelum-sebelumnya sehingga memungkinkan untuk mengurangi jumlah insiden yang terjatuh dengan dilakukan pengawasan di

lokasi tersebut dan dapat dianalisis lebih lanjut mengenai penyebab terjadinya insiden tersebut tergantung pada lokasi.

3.2 Arsitektur Umum Sistem

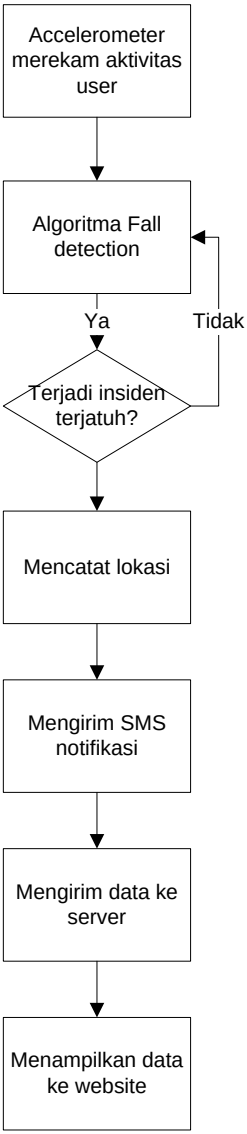
Rancangan arsitektur pada sistem yang akan dibangun seperti pada Gambar 3.1



Gambar 3.1 Diagram Blok Arsitektur Siste

Berdasarkan pada Gambar 3.2 Diagram Alir Sistem *Fall Detection*, sistem *fall detection* ini memiliki alur proses yang dijabarkan sesuai nomor sebagai berikut:

1. Motion *Shield* dihubungkan dengan mikrokontroller Arduino Mega kemudian mengatur *accelerometer* dengan tingkat sensitivitas diset ±16g.
2. GPRS/GSM/ GPS Shield SIM908 dipasangkan di atas Arduino shield, setelah itu dilakukan konfigurasi perangkat yaitu mengatur apn dari *provider* yang digunakan.
3. Pada tahap ini perangkat bergerak pada *fall detection* siap untuk digunakan untuk mendapatkan data sensor secara *real-time* dari aktivitas pengguna sebelum jatuh dan saat terjatuh. hasil dari *training* data akan dikelompokkan berdasarkan kondisi suatu kejadian dan mencari hubungan tertentu dari suatu data *training.*
4. Data akan diproses sedemikian rupa dengan menggunakan algoritma *fall detection*, pada metode *decision tree* ditentukan melalui nilai *threshold* yang menandakan bahwa pengguna sedang mengalami insiden terjatuh, pada metode *k*-NN fitur yang digunakan dari nilai quaternion *q0, q2* serta nilai *dx*.
5. Dilakukan pengecekan pada postur *user* dengan penghitungan nilai *tetha* apakah memang benar user dalam kondisi jatuh atau tidak.
6. Perangkat bergerak akan mengirimkan data letak posisi kejadian tersebut berupa koordinat *longitude* dan *latitude* yang didapatkan dari GPS dengan perintah AT Command yaitu *POST* HTTP ke server menggunakan GPRS. perangkat juga akan mengirimkan SMS kepada keluarga korban menggunakan perintah dari AT Command.
7. *Website* sistem monitoring *fall detection* akan melakukan *request* pula berupa HTTP GET pada server untuk mendapatkan data *latitude* dan *longitude* sebagai posisi dimana insiden terjatuh terjadi. .



Gambar 3.2 Diagram Alir Sistem *Fall Detection*

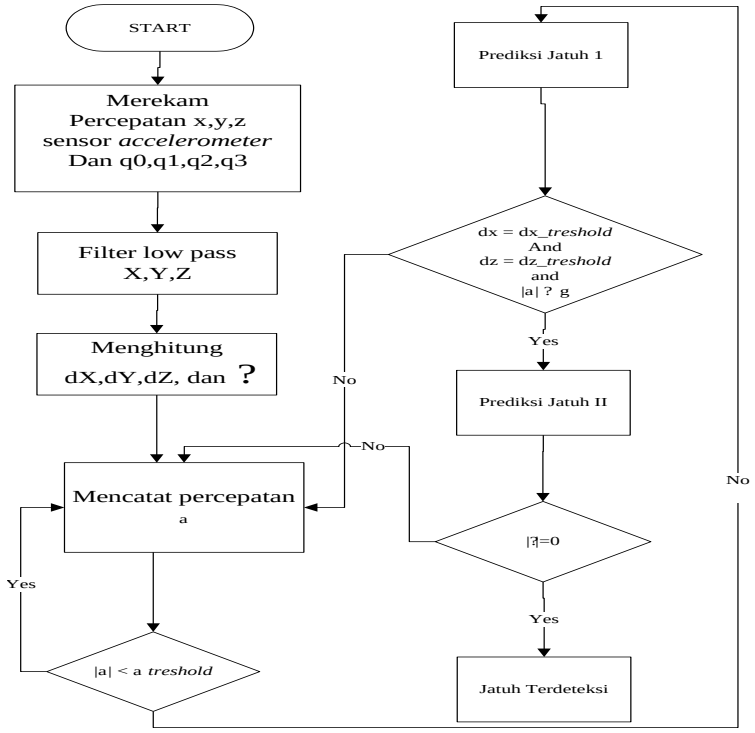
3.3 Perancangan Pendeteksian Jatuh Menggunakan *Decision Tree*

Pada Gambar 3.3 menjelaskan Algoritma *fall detection* merupakan algoritma pendeteksi pergerakan fisik ketika terjadi insiden jatuh yang menggunakan sensor *accelerometer* dan menghitung sudut rotasi pergerakan menggunakan nilai *quaternion* Pada algoritma ini dilakukan pembagian dua aktivitas, yaitu statis dan dinamis. Algoritma ini dibagi menjadi 3 langkah, yaitu analisis terjadinya transisi gerakan, analisis aktivitas, dan analisis postur.

Secara umum Perancangan pendeteksian jatuh adalah sebagai berikut [5]:

1. Pertama, sensor *accelerometer* mencatat pergerakan fisik user berdasarkan sumbu x, y z, dan merekam nilai *quaternion* q_0, q_1, q_2, q_3 setiap 200 ms, untuk mendapatkan nilai-nilai tersebut digunakan *library* Adafruit_BNO055.
2. Untuk mereduksi adanya *noise* digunakan *low pass filter* pada nilai x,y,z.
3. Menghitung nilai perubahan pada masing-masing sumbu *dX*,*dY*,*dZ* dan nilai tetha
4. Kemudian menghitung nilai *alpha* yaitu percepatan α dari akar dari pertambahan nilai sensor *accelerometer*
5. Kemudian mengecek apakah nilai $|a|$ kurang dari nilai *a threshold*, jika iya maka transisi pergerakan user dianggap statis. Sebaliknya jika tidak, maka terjadi transisi pergerakan dinamis.
6. Kemudian mengecek apakah nilai $|dX|$ dan $|dZ|$ melebihi nilai $|dX|$ dan $|dZ|$ *threshold dan* nilai $|a|$ tidak sama dengan nilai gravitasi yaitu 9.8/ms^2. Jika tidak maka dipastikan *user* mengalami transisi gerakan dinamis namun bukan transisi jatuh.
7. Kemudian mengecek apakah sudut dari postur *user* adalah 0^o, jika iya maka user mengalami insiden terjatuh. Untuk mengetahui postur *user*, maka dilakukan perhitungan sudut *theta* (θ). Sudut *theta* (θ) adalah sudut antara *perangkat*

bergerak yang berada pada daerah saku kantong terhadap pusat gravitasi.



Gambar 3.3 Perancangan Pendeteksian *Fall Detection*

Menentukan nilai *Threshold*

Pada sistem ini terdapat beberapa nilai *threshold* yang digunakan untuk mendeteksi pergerakan fisik *user*, apakah user dalam keadaan statis atau tidak, jika tidak apakah pergerakan transisi fisik user yang dinamis tersebut merupakan pergerakan fisik yang menandakan pergerakan pada aktivitas terjatuh. Untuk itu diperlukan nilai *threshold* untuk membedakan pergerakan fisik

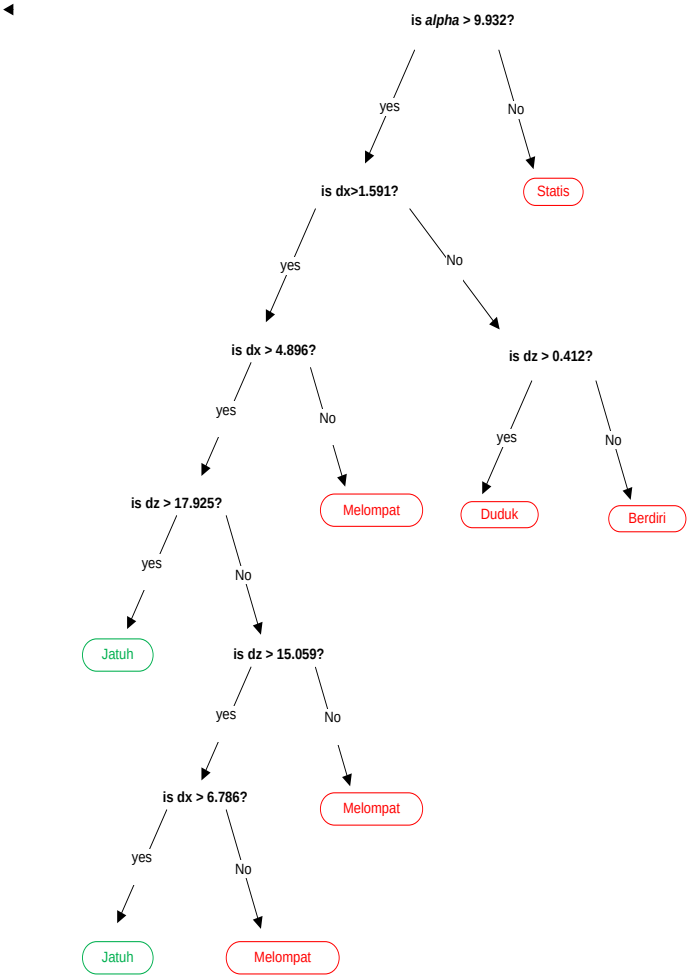
tersebut, nilai *threshold* yang digunakan dalam sistem *fall detection* antara lain:

1. Nilai *threshold* alpha (α)
   Untuk mendeteksi aktivitas fisik *user* dalam keadaan normal atau statis dapat digunakan dengan cara membandingkan nilai alpha *threshold* dengan nilai alpha *real-time*. Jika pembacaan nilai alpha (α) pada sensor mendekati nilai alpha (α) pada *threshold* maka user dalam keadaan normal atau statis. Besarnya nilai *threshold* pada keadaan normal atau statis ialah sebesar 9,932 m/s^2, nilai ini didapat dari data training, dalam tertinggi keadaan statis ialah 9.932 m/s^2
2. Nilai *threshold dX* dan *dZ* dengan menggunakan *Decision Tree*
   Saat terjadi aktivitas transisi fisik pergerakan *user* dinamis sepert aktivitas duduk, aktivitas melompat dan insiden terjatuh yang digunakan untuk mendeteksi aktivitas tersebut adalah dengan nilai *threshold dX* dan *dZ*. Untuk menentukan besarnya nilai tersebut digunakan *Decision Tree* menggunakan aplikasi Orange3 Data Mining.

Pada Gambar 3.4 untuk mendapatkan keputusan mendeteksi seseorang dalam keadaan statis (diam) dilihat dari nilai *alpha real-time*, jika nilai *alpha* kurang dari 9,932 maka *user* dalam keadaan statis, namun jika melebihi dari nilai tersebut, *user* dalam keadaan dinamis.

Pada pergerakan dinamis yaitu insiden terjatuh, untuk mengecek apakah seseorang dalam kondisi jatuh apa tidak maka dapat dilihat dari perubahan nilai *dX* apabila nilai *dX* melebihi 4,896 dan kemudian dicek kembali apabila nilai *dZ* melebihi 17,925 maka berpotensi terjadi insiden terjatuh, dan untuk memastikan bahwa memang benar telah terjadi insiden terjatuh maka dicek pada sudut postur *user* dengan menggunakan nilai *tetha*, apabila nilai *tetha* kurang dari 45° maka dipastikan *user* dalam posisi terjatuh. Kemudian jika nilai *dX* melebihi 4,896 dan kemudian dicek kembali apabila nilai *dZ* kurang dari 17,925 masih berpotensi untuk mendeteksi insiden terjatuh dengan

mengecek nilai *dZ,* apabila nilai *dZ* melebihi 15,059 dan kemudian dicek kembali apabila nilai *dX* melebihi 6,786 serta tetha kurang dari 45° maka dipastikan *user* dalam posisi terjatuh.



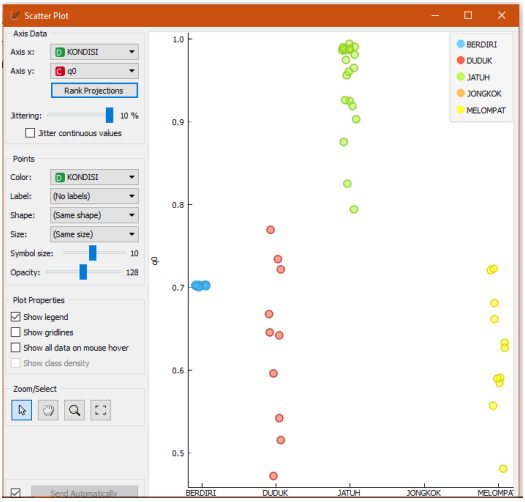
Gambar 3.4 ***Decision Tree Fall Detection***

3.4 Menganalisis Nilai-Nilai mana saja yang digunakan sebagai *Feature* dalam Metode K Nearesr Neighbors

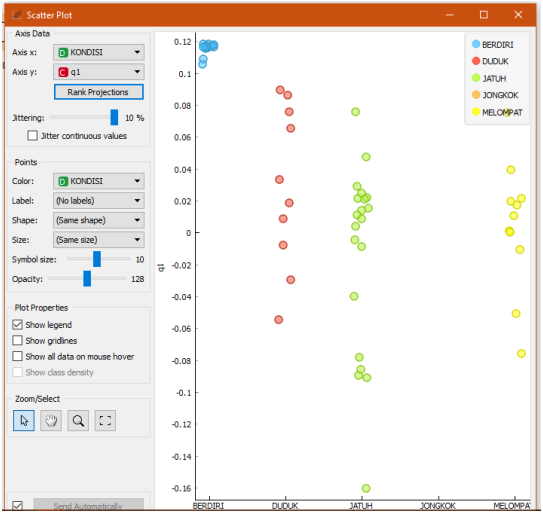
Untuk mendeteksi seseorang dalam keadaan jatuh, selain menggunakan *decesion treee* pada sistem ini juga menggunakan metode *k*NN, untuk itu semua nilai yang dibaca oleh sensor secara *real-time* yaitu nilai *quaternion* dan perubahan percepatan pada setiap sudut akan dianalisis terlebih dahulu, nilai-nilai mana saja yang akan digunakan sebagai *feature* dalam metode *k*NN. Analisis pada metode ini berdasarkan dari data *training* yang telah dilakukan.

3.4.1 Menganalisis Nilai *Quaternion* yang digunakan sebagai *Feature* dalam Metode K Nearest Neighbors

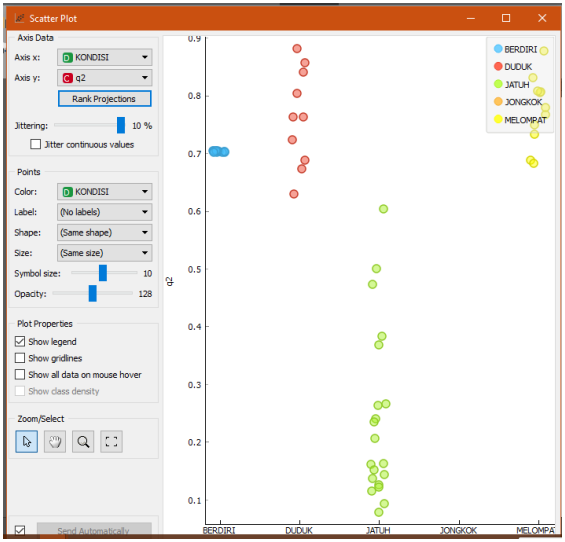
Untuk mendapatkan hasil pendeteksian dengan akurasi tinggi dan mengurangi penggunaan memori dalam metode *k*-NN pada mikrokontoller arduino nilai-nilai *quaternion* Q (*q0,q1,q2,q3*) pada saat data *training* dilakukan analisis untuk menentukan nilai *quaternion* mana saja yang digunakan sebagai *feature* dalam metode *k*-NN. Nilai *quaternion* yang digunakan sebagai *feature* dalam metode *k*-NN ialah nilai pada aktivitas jatuh yang tidak saling berpotongan dengan nilai pada aktivitas lainnya seperti berdiri, duduk dan melompat sehingga *query instance* yang baru dapat dengan mudah diklasifikasikan berdasarkan mayoritas dari suatu *class* yaitu terjadi *class* jatuh atau *class* tidak terjatuh. Nilai-nilai pada data *training* tersebut dilihat dengan menggunakan scatter plot dengan Orange Data Mining. Pada Gambar 3.5 dan pada Gambar 3.7 merupakan scatter plot pada nilai *quaternion q0* dan q2, nilai-nilai *q0* dan *q2* pada aktivitas jatuh tidak berpotongan dengan nilai pada aktivitas lainnya, sehingga nilai q0 dan q2 dapat dijadikan sebagai *feature* dalam metode *k-NN.* Sedangkan pada Gambar 3.6 dan Gambar 3.8 merupakan scatter plot pada nilai *quaternion q1* dan *q3,* nilai-nilai *q1* dan *q3* pada aktivitas jatuh memiliki nilai yang saling berpotongan pada aktivitas lainnya seperti berdiri,duduk, dan melompat sehingga nilai *q1* dan *q3* tidak digunakan sebagai *feature* dalam metode *k*-NN.



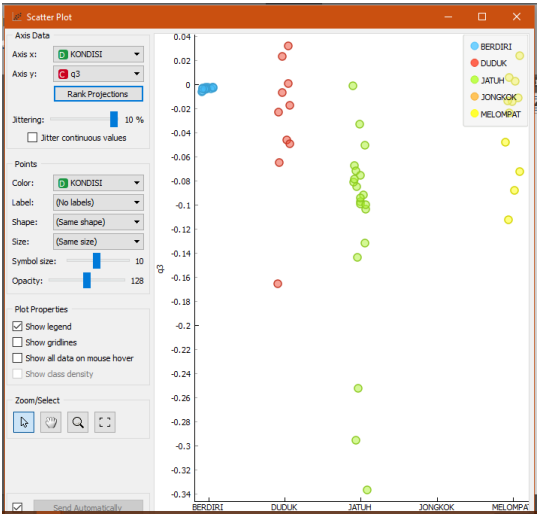
Gambar 3.5 Scatter Plot Nilai Quaternion q0



Gambar 3.6 Scatter Plot Nilai Quaternion q1



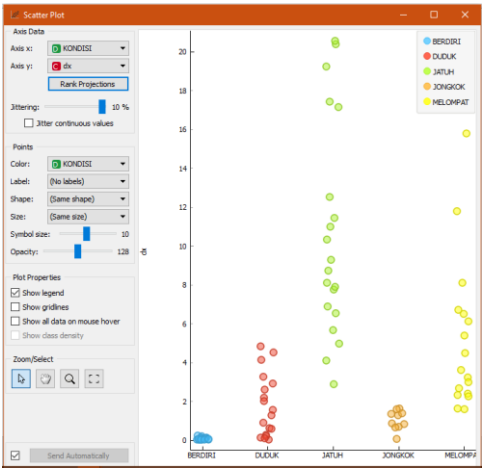
Gambar 3.7 Scatter Plot Nilai Quaternion q2



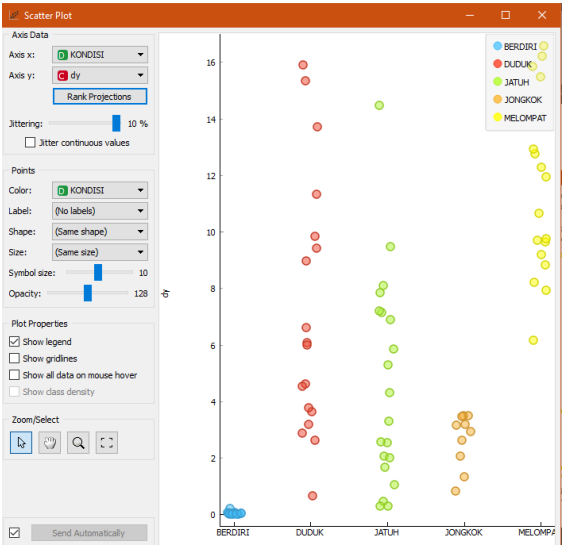
Gambar 3.8 Scatter Plot pada Nilai Quaternion q3

3.4.2 Menganalisis Nilai Selisih *dX*, *dY* dan *dZ* yang digunakan sebagai *Feature* dalam Metode K Nearest Neighbors

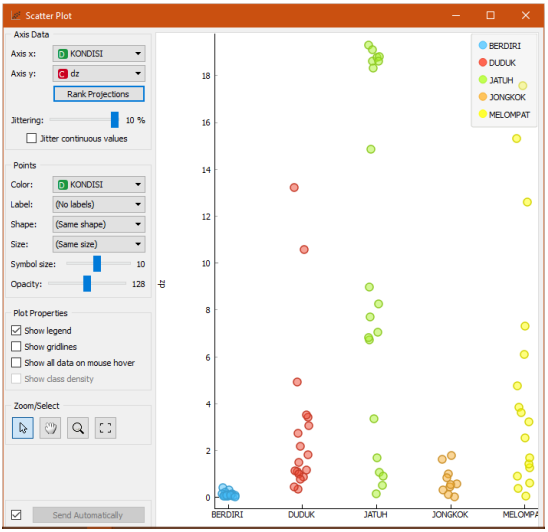
Nilai perubahan percepatan tiap sumbu (*dX*,*dY*,*dZ*) yang digunakan sebagai *feature* dalam metode *k*-NN ialah nilai pada aktivitas jatuh yang tidak saling berpotongan dengan nilai pada aktivitas lainnya seperti berdiri, duduk dan melompat sehingga *query instance* yang baru dapat dengan mudah diklasifikasikan berdasarkan mayoritas dari suatu *class* yaitu terjadi *class* jatuh atau *class* tidak terjatuh. Nilai-nilai pada data *training* tersebut dilihat dengan menggunakan scatter plot dengan Orange Data Mining. Pada Gambar 3.9 merupakan scatter plot pada nilai *dX*, nilai-nilai *dX* pada aktivitas jatuh cenderung tidak berpotongan dengan nilai pada aktivitas lainnya, sehingga nilai nilai *dX* dapat dijadikan sebagai *feature* dalam metode *k-NN*. Sedangkan pada Gambar 3.10 dan Gambar 3.11 merupakan scatter plot pada nilai *dY* dan *dZ*, nilai-nilai *dY* dan *dZ* pada aktivitas jatuh memiliki nilai yang saling berpotongan pada aktivitas lainnya seperti berdiri,duduk, dan melompat sehingga nilai *dY* dan *dZ* tidak digunakan sebagai *feature* dalam metode *k*-NN.



Gambar 3.9 Scatter Plot pada Nilai *dX*



Gambar 3.10 Scatter Plot pada Nilai *dY*



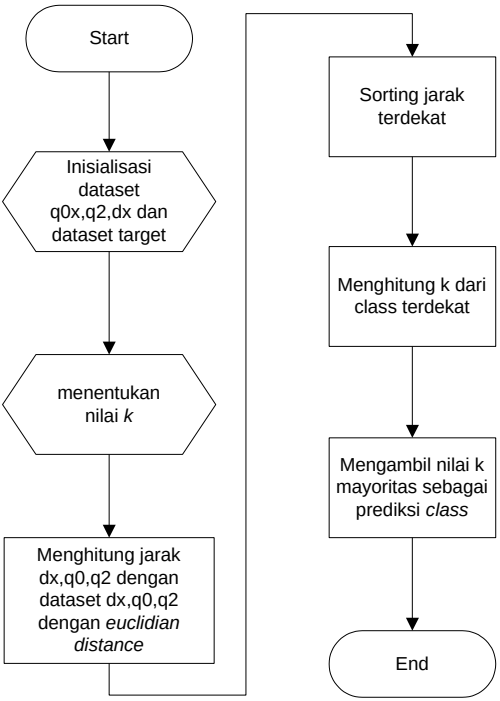
Gambar 3.11 Scatter Plot pada Nilai *dZ*

3.5 Perancangan Pendeteksian Jatuh Menggunakan Metode K Nearest Neighbors

Setelah melalui proses analisis pada tahapan sebelumnya dan didapatkan bahwa yang dijadikan *feature* dalam metode ini ialah nilai $dX, q0$ dan $q2$, kemudian pada sub bab ini menjelaskan tahapan peracangan bagaimana mendeteksi seseorang dalam kondisi terjatuh menggunakan metode *kNN*. Pada metoe *k*NN akan melakukan klasifikasi berdasarkan kedekatan lokasi (jarak) suatu data dengan data yang lain, pengklasifikasian pada titik *query instance* yakni pada sistem ini ialah data-data perubahan pengguna akan digolongkan berdasarkan sejumlah K objek yang paling dekat dengan titik *query*, jika mayoritas pada K objek tersebut pada suatu *class* misalnya dalam hal ini ialah *class* jatuh, maka query instance tersebut tergolong kedalam *class* jatuh. Pada *k*NN jarak antara satu data dengan data yang lain dapat dihitung. Nilai jarak inilah yang digunakan sebagai kedekatan atau kemiripan antara data uji dengan data *training*. Salah satu yang diperhatikan pada *k*NN adalah pemilihan nilai K yang tepat. Pada nilai K yang besar bisa menyebabkan distorsi data yang besar, hal itu karena setiap tetangga mempunyai bobot yang sama terhadap data uji. Sedangkan K yang terlalu kecil bisa menyebabkan terlalu sensitif terhadap *noise*.

Pada Gambar 3.12 merupakan urutan proses pada metode *k*-NN yang digunakan. Mula-mula sistem menginisialisasi *data set* nilai $dX, q0$ dan $q2$ serta menginisialisasi *dataset* target, kemudian menentukan nilai K pada sistem ini nilai K adalah 5, berdasarkan hasil uji coba K pada 5 memiliki tingkat akurasi yang lebih baik. Kemudian pada setiap pembacaan sensor nilai *accelerometer* $dX, q0$ dan $q2$ dilakukan penghitungan jarak dengan nilai *dataset* $dX, q0$ dan $q2$ dengan *eculidean distance*. Setelah itu pada nilai jarak dilakukan sortir dari nilai terkecil hingga terbesar sekaligus memasangkan dengan *dataset* target. Setelah itu dilakukan penghitungan dengan K terdekat apakah termasuk pada target "Jatuh Terdeteksi" atau "Tidak Terjatuh". Pada *class* target mayoritas merupakan prediksi akhir dari proses metode ini,

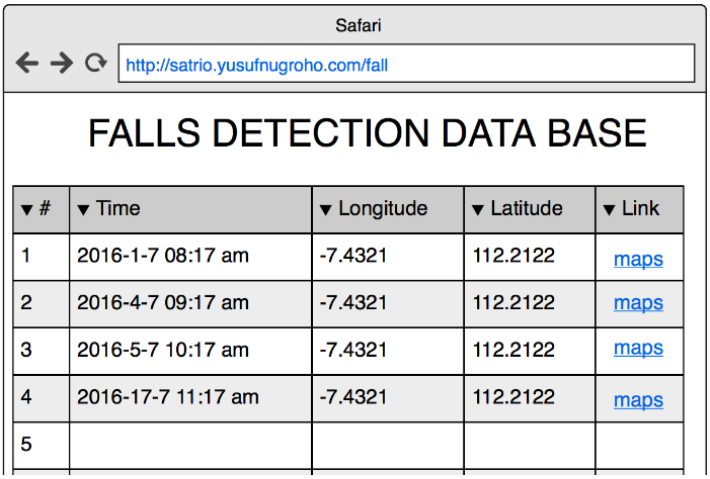
sehingga dapat disimpulkan bahwa prediksi dari metode *k*-NN sesuai dengan pergerakan fisik *user*.



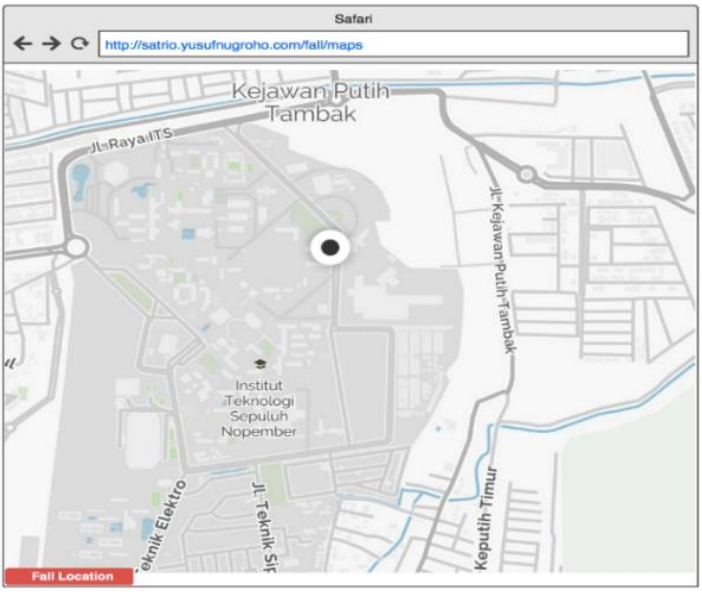
Gambar 3.12 Diagram Alir metode k NN pada *Fall Detection*

3.6 Perancangan Antarmuka Sistem

Pada penelitian ini akan dirancang antarmuka untuk pengguna yakni antarmuka riwayat insiden terjatuh yang memuat informasi berupa *history* data-data pengguna ketika mengalami insiden terjatuh yang berisi posisi koordinat *latitude* dan *longitude* dan waktu terjadinya insiden seperti yang ditunjukkan pada Gambar 3.13 Sedangkan pada Gambar 3.14 merupakan posisi insiden jatuh ketika dibuka dengan *Google Maps* API.



Gambar 3.13 Rancangan Antarmuka Melihat riwayat jatuh



Gambar 3.14 Antarmuka posisi jatuh saat dilihat dengan *Google Maps*

3.7 Perancangan Kebutuhan Arus Listrik

Perancangan kebutuhan arus listrik adalah salah satu proses penting yang harus dilakukan dalam membangun sistem. Hal ini terjadi karena masing-masing komponen dalam suatu sistem membutuhkan catu daya yang berbeda-beda, agar seluruh komponen bisa mendapatkan arus listrik minimum yang baik dan dapat secara optimum menjalankan sistem, maka perlu dipersiapkan kebutuhan arus listrik masing-masing komponen. Perancangan kebutuhan arus listrik dapat dilihat pada Tabel 3.1

Pada penggunaan GPRS untuk mengirimkan data dari sensor ke server kebutuhan arus listrik yang digunakan hanya pada saat pengiriman berlangsung, begitu juga dengan penggunaan GSM.

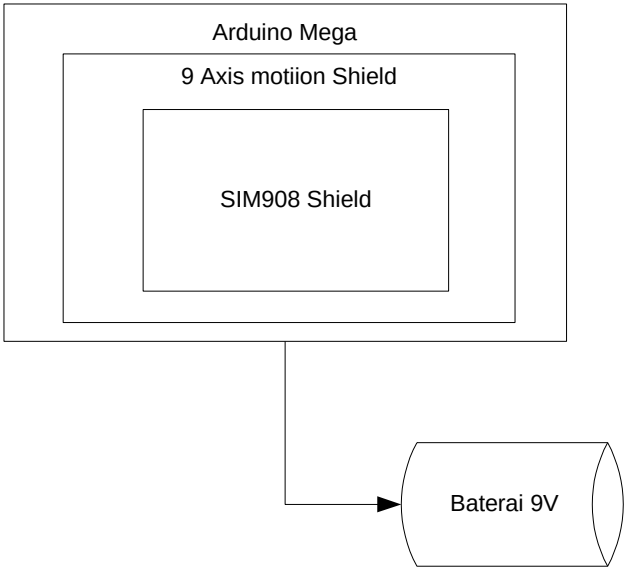
Tabel 3.1 Perancangan Kebutuhan Daya

Nama Perangkat	Kebutuhan Arus Listrik	
	Arus Listrik	Tegangan Listrik
Arduino Mega	40 mA	5V DC
GSM/GPRS/*GPS Shield*	100 mA	6-12V DC
9 axis motion Shield	10 mA	3.3/5V Dc
Total	150 mA	-

3.8 Perancangan Perangkat Keras

Perancangan perangkat keras secara umum menjelaskan mengenai perancangan penempatan perangkat keras yang digunakan untuk membangun sistem. Rangkaian perangkat keras pada sistem ini ditunjukkan pada Gambar 3.15, terdapat komponen sensor orientasi 9 axis motion Shield, Arduino Mega, GPRS/GSM/GPS Shield SIM908, serta 1 buah baterai 9V. GPRS/GSM/GPS Shield SIM908 ditempatkan pada atas Arduino

Mega, dan 9 Axis Motiion Shield di letakan di atas SIM908 Shield. 9 Axis Motiion Shield berfungsi untuk mengirimkan data sensor dari Arduino ke server, sedangkan Uttuk mengirimkan pemberitahuan saat terjadi insiden terjatuh berupa SMS melalui mode GSM, pada penelitian ini penulis menggunakan provider "Tri Indonesia". Pada SIM908 terdapat dua antena yaitu antena GSM dan antena GPS, pada penelitian ini dua antena pada SIM908 yang dipakai telah dipasangkan secara langsung, karena ada beberapa tipe SIM908 yang harus memasangkan antena GSM secara manual. SIM908 pada mode GSM mendukung 4 frekuensi gelombang radio yaitu 850 MHz/ 900 MHz/ 1800 MHz/ 1900 MHz.



Gambar 3.15 Skema Perancangan Perangkat Keras

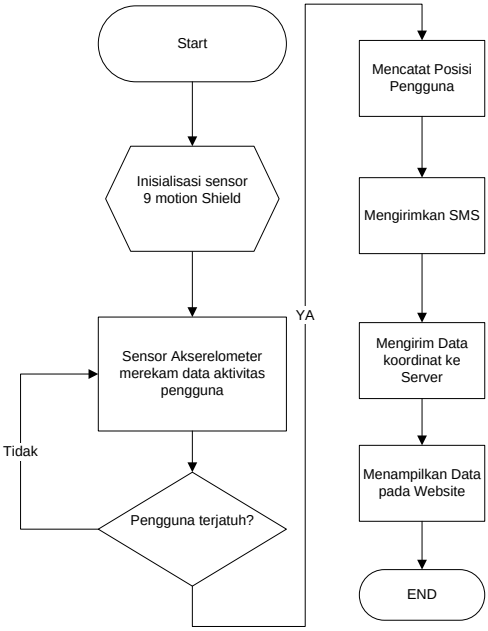
3.9 Diagram Alir Aplikasi Sistem

Diagram alir aplikasi sistem dibuat untuk memudahkan dalam merancang dan memahami seluruh proses yang terjadi di dalam sistem. Diagram alir aplikasi sistem terdiri dari diagram alir keseluruhan sistem, diagram alir sensor orientasi 9 axis *motion Shield.* diagram alir pengiriman pemberitahuan melalui SMS dengan GSM Shield, diagram alir pengiriman data ke server, diagram alir penampilan data pada *website*, dan diagram alir GPS Shield untuk mencari lokasi jatuh. diagram alir data keseluruhan sistem

Perancangan alir data keseluruhan sistem dilakukan untuk dapat lebih memahami dan memberikan pandangan secara menyeluruh mengenai sistem yang ditangani, serta menunjukkan tentang fungsi-fungsi utama atau proses yang ada dan aliran data maupun pada pra proses seperti menginisialisasi GPS dan sensor orientasi .

Pada Gambar 3.16, sistem diawali dengan menginisialisasi GPS dan sensor orientasi 9 axis *motion Shield,* inisialisasi pada sensor dan GPS dimaksudkan untuk mendapatkan nilai yang standar dan mendapatkan data GPS yang presisi, kemudian sensor mulai merekam aktivitas fisik pengguna pada setiap waktu yang telah ditentukan, jika terdapat perubahan aktivitas fisik yang mengindikasikan gerakan terjatuh perangkat bergerak akan mencatat koordinat lokasi kejadian, namun jika pergerakan fisik tersebut tidak mengindikasikan gerakan terjatuh atau hanya sebatas berpotensi menyebabkan gerakan terjatuh maka sistem akan kembali untuk merekam aktivitas pengguna hingga pengguna mengalami insiden terjatuh.

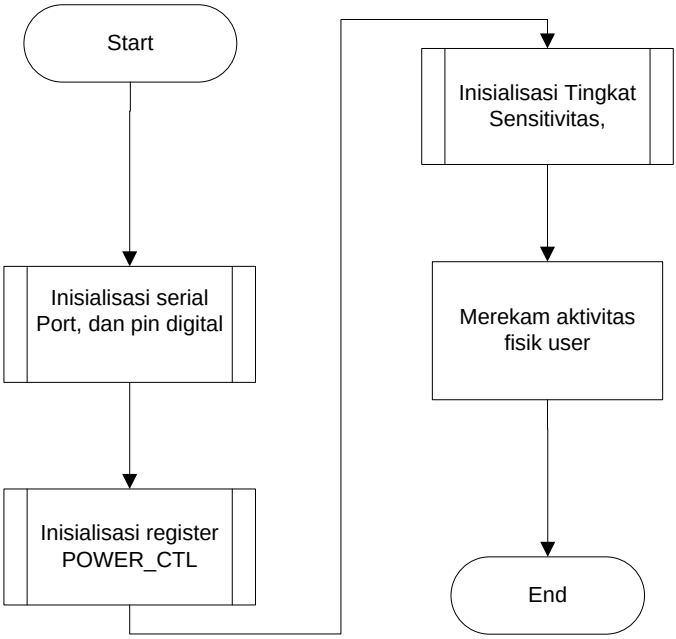
Data lokasi kejadian berupa koordinat *latitude* dan *longitude* akan dikirimkan ke server, dan data tersebut juga akan dikirimkan melalui SMS kepada kerabat korban.



Gambar 3.16 Diagram Alir Keseluruhan Sistem

3.9.1 Diagram Alir Sensor Orientasi 9 axis *motion Shield*

Gambar 3.17 menunjukkan bagaimana proses sensor orientasi yang di dalamanya terdapat sensor *accelerometer*, *gyroscope* dan magnetometer bekerja Dalam proses mendeteksi seluruh aktivitas fisik pengguna dan untuk mengetahui adanya gerakan yang mengindikasikan adanya gerakan transisi dinamis dilakukan oleh sensor *Accelerometer*, sedangkan untuk mendeteksi sudut rotasi postur *user* menggunakan nilai *quaterion* dengan menghitung *tetha*. Sistem *fall detection* ini menggunakan tingkat sensitivitas accelerometer sebesar ±16g. Data rate sebesar 100 Hz, sensor ini mencatat pergerakan user tiap 200 ms.



Gambar 3.17 Diagram Alir Inisialiasi Motion Shield

3.9.2 Diagram Alir Pengiriman Data

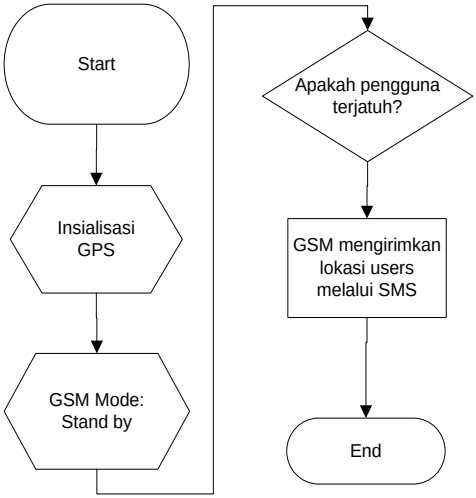
Gambar 3.19 menunjukkan diagram alir pengiriman data aliran sensor dan waktu dengan GSM *Shield* SIM908 melalui jaringan GPRS. Setelah dilakukan inisialisasi RTC, *Serial Port*, dan *Serial Digital*, maka tahap yang akan dilakukan oleh Arduino selanjutnya adalah melakukan koneksi GPRS. Hal ini dilakukan agar Arduino dapat berkomunikasi dengan server dan dapat saling berkirim data. Koneksi GPRS bisa didapatkan melalui GPRS *Shield* yang telah dipasang bersama *SIM Card*. Masing-masing *SIM Card* memiliki *Access Point Names* (APN) yang berbeda beda, sehingga perlu diperhatikan dalam inisialisasi nama GPRS dan A*PN SIM Card*.

3.9.3 Diagram Alir Menampilkan Data pada Website

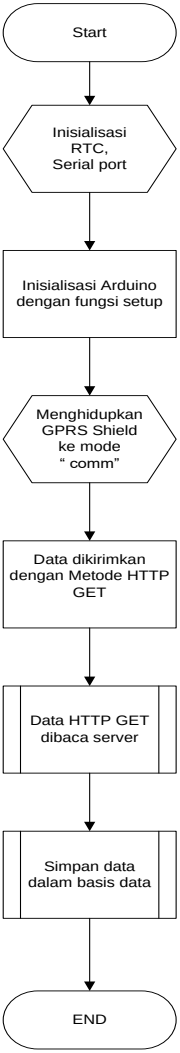
Proses aliran data dalam menjalankan *website* sistem *fall detection* dengan melakukan *request* HTTP GET kepada server, website melakukan *query* pada server secara langsung. Data yang ditampilkan pada *website* adalah data koordinat *latitude* dan *longitude* posisi *user* yang sedang mengalami insiden terjatuh. Diagram aliran data ditunjukkan oleh Gambar 3.20.

3.9.4 Diagram Alir GSM Shield Mengirim SMS

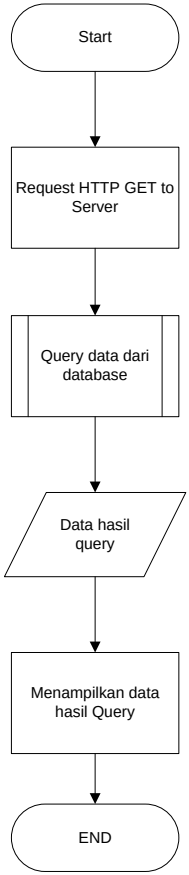
Seperti pada Gambar 3.18, sistem akan mengirimkan SMS secara otomatis kepada kerluarga korban apabila *users* mengalami insiden terjatuh. Data yang dikirimkan berupa koordinat *longitude* dan *latitude* beserta dengan alamat *url* dengan paramater koordinat sehingga dapat dilihat dengan menggunakan *Google Maps.*



Gambar 3.18 Diagram Alir GSM Shield Mengirim SMS Sebagai Notifikasi



Gambar 3.19 Diagram Alir Mengirimkan Data ke Server



Gambar 3.20 Diagram Alir Menampilkan Data pada Website

3.10 Perancangan Basis Data Sistem

Basis data yang digunakan pada aplikasi *website* Sistem *fall detection* ini adalah MySQL. *Physical Data Model* (PDM) aplikasi ditunjukkan pada Gambar 3.9. Hanya terdapat 1 tabel dalam basis data yang digunakan yaitu *table* History Fall yang memuat atribut ide_fall, latitude, longitude serta waktu jatuh.

No.	Nama_Atribut	Tipe Data	Keterangan
1	Id_fall	Integer	Primery_key untuk tabel Fall
2	latitude	float	Nilai latitude sebagai penunjuk lokasi terjadinya insiden jatuh
3	longitude	float	Nilai longitude sebagai penunjuk lokasi terjadinya insiden jatuh
4	Waktu_jatuh	timestamp	Waktu data diterima pada server

Gambar 3.9 Perancangan Tabel History Fall Sistem *Fall Detection*

(Halaman ini sengaja dikosongkan)

BAB IV
IMPLEMENTASI

Pada bab ini akan dibahas mengenai implementasi dari perancangan perangkat lunak. Cakupan implementasi dari perancangan perangkat lunak tersebut meliputi proses penerapan dan pengimplementasian algoritma dan antarmuka.

4.1 Lingkungan Implementasi

Lingkungan implementasi sistem yang digunakan untuk mengembangkan Tugas Akhir memiliki spesifikasi perangkat keras dan perangkat lunak seperti yang ditunjukkan Tabel 4.1.

Tabel 4.1 Lingkungan Implementasi Sistem

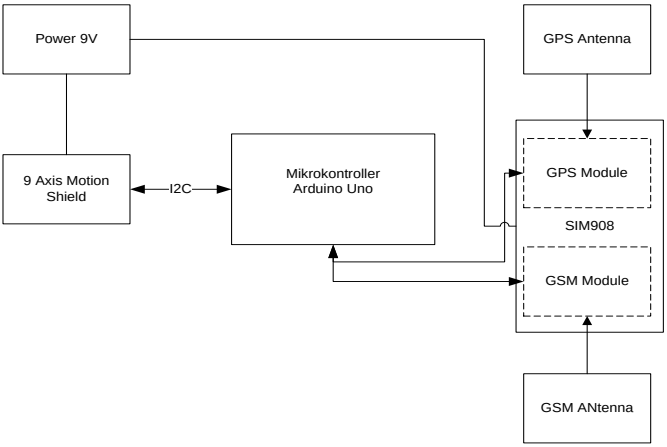
Perangkat	Spesifikasi
Perangkat Keras	Macbook Pro Mid 2014 Core™ i5 @ 2.6 GHz 8Gb 1600 MHz Perangkat Sistem: Arduino Mega, GSM Shield Arduino SIM908 , 9 axis motion Shield
Perangkat Lunak	Sistem Operasi: OSX El Capitan *Version* 10.11.4 Perangkat Pengembang: Arduino IDE, MySql, Sublime Orange3 Data Mining, Coolterm Google Chrome Perangkat Pembantu: Microsoft Visio 2013, Power Designer, Microsoft Word 2013, Sniping tools,

4.2 Implementasi Perangkat Keras

Implementasi perangkat keras pada sistem *fall detection* adalah sebagai berikut:

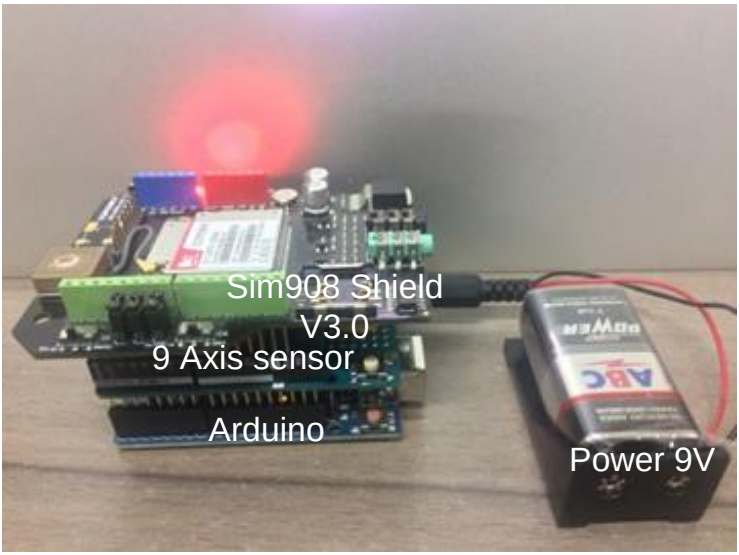
- 1 (satu) buah Arduino Mega.
- 1 (satu) buah GSM/GPRS/GPS *Shield* Arduino SIM908.
- 1 (satu) buah 9 axis motion Shield.
- Beberapa jumper.
- 1 (satu) buah Baterai 9V beserta adapter.

Arduino Mega berperan sebagai pusat kontrol pada sistem *fall detection* dan pusat penerimaan data 9 axis motion Shield dan koordinat *latitude* dan *longitude*. Apabila aktivitas fisik *users* mengindikasikan gerakan jatuh, maka data yang diterima berupa data koordinat *latitude* dan *longitude* melalui jaringan GPRS, dan mengirimkan SMS melalui GSM *Shield* Arduino SIM908 yang menggunakan *SIM Card*. Perancangan perangkat keras dapat dilihat di Gambar 4.1.



Gambar 4.1 Perangkat Keras Sistem F*all detection*

Setelah perangkat keras terpasang seperti pada Gambar 4.1, hasil dari rangkaian perangkat keras dapat dilihat pada Gambar 4.2.



Gambar 4.2 Rangkaian Perangkat Keras Sistem *fall detection*

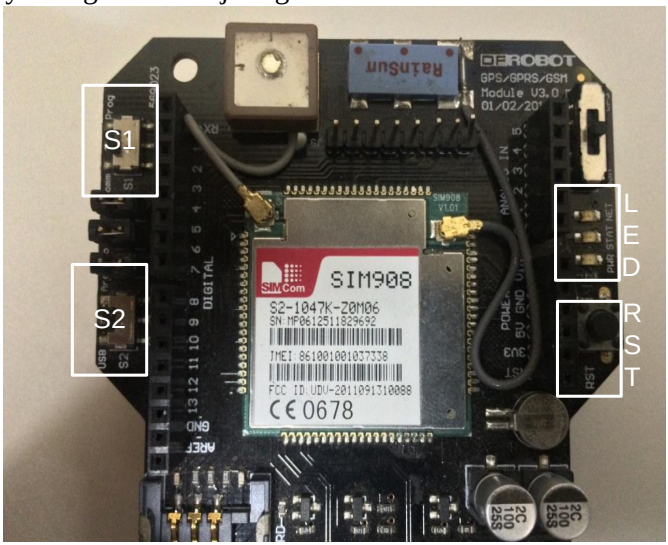
4.3 Implementasi Perangkat Lunak Pada Perangkat Keras

Implementasi perangkat lunak di Arduino pada sistem *fall detection* terdapat beberapa fungsi, yaitu sebagai pusat penerimaan 9 axis motion Shield, sebagai pengirim data koordinat *latitude* dan *longitude* ke server menggunakan GSM *Shield* Arduino SIM908, dan yang terakhir sebagai pengirim SMS berisi posisi terjadinya insiden jatuh kepada keluarga korban.

Sebelum mengimplementasikan keseluruhan perangkat lunak pada perangkat keras, yang penting untuk diperhatikan adalah melakukan uji coba kerja GSM *Shield* Arduino SIM908. Untuk mencari koordinat posisi latitude dan longitude, mengirimkan data ke server dan mengirimkan SMS dengan

menggunakan *Shield tersebut* ada yang perlu diperhatikan yaitu mode pada switch S1 yang digunakan dalam Shield tersebut yaitu dengan mode “Comm” setelah itu menekan tombol “RST” hingga kedua lampu LED pada SIM908 Shield yaitu “STAT” dan “NET” menyala dengan warna kuning seperti yang ditunjukkan pada Gambar 4.3 yang artinya bahwa *network states* dan *working states* dalam keadaan siap untuk digunakan.

Dalam melakukan pencarian koordinat posisi latitide dan longitude, mengirimkan data ke server dan mengirimkan SMS dibutuhkan GSM *Shield* Arduino yang mampu melakukan keduanya dengan baik di jaringan GPRS.



Gambar 4.3 SIM908 V30 *Shield*

Untuk implementasi penggunaan GSM *Shield* Arduino SIM908, yang perlu diperhatikan adalah catu daya yang digunakan, yaitu minimal mendapatkan catu daya 6 -12 V dan kuat arus 1000mA. Pada penelitian ini, digunakan catu daya 9 V dengan kuat arus 1000 mA.

4.4 Implementasi Inisialisasi Sensor Orientasi Motion Shield.

Gambar 4.4 merupakan *pseudocode* implementasi inisialisasi *Sensor* Orientasi *9 axis Motion Shield* yaitu untuk mengkonfigurasi *register-register* yang terdapat pada sensor tersebut. Hal pertama yang dilakukan adalah melakukan inisialisasi *sensor* pada register POWER_CTL yang berfungsi untuk mengaktifkan sensor tersebut agar dapat membaca setiap gerakan yang terjadi, kemudian mengatur tingkat sensitivitas sensor dan mengatur dataRate yang berfungsi mengatur kecepatan sensor dalam proses transmisi.

Prosedur Sistem *Fall Detection* Inisialisasi Sensor Orientasi
Masukan: -
1 function Setup{ 2 WriteRegister(BNO055 REG POWER CTL) 3 SetRange(accelerometer) 4 SetDataRate(accelerometer) 5 *Set*Calibrate(accelerometer,gyroscope,magnetometer) 5 }
Keluaran: -

Gambar 4.4 Implementasi Inisialisasi Sensor Orientasi

4.5 Implementasi Mendapatkan Nilai Sensor *Accelerometer dan Quaternion*

Prosedur Sistem *Fall Detection* Mendapatkan Nilai Accelerometer
Masukan: -
1 function readAccelData { 2 variabel float getX,getY,getZ,alpha 3 getX <-acceleration.x 4 getY <-acceleration.y 5 getZ <-acceleration.z 6 alpha = SQRT(SUMSQ(getX,getY,getZ) 7 alpha = sqrt((pow(getX,2) + pow(getY,2) + pow (getZ,2)) 8 delay 200 9 }
Keluaran: Nilai sumbu X, Y, Z, alpha

Gambar 4.5 Implementasi Mendapatkan Nilai Accelerometer

Prosedur Sistem *Fall Detection* Mendapatkan Nilai Quaternion
Masukan: -
1 function readQuatData { 2 variabel float getq0,getq1,getq2, getq3,tetha,qtetha 2 getq0 <- *quaternion*.q0 3 getq1 <- *quaternion*.q1 4 getq2 <- *quaternion*.q2 5 getq3 <- event->*quaternion*.q3 6 qtetha = SQRT(SUMSQ((getq1,getq2,getq3)/getq0)) 7 tetha = 2 * arctan (qtetha) * 180/3.14 8 delay 200 9 }
Keluaran: Nilai sumbu q0, q1, q2, q3 theta

Gambar 4.6 Implementasi Mendapatkan Nilai Tetha

Gambar 4.5 merupakan kode sumber implementasi untuk mendapatkan data *accelerometer*. Sistem secara langsung membaca nilai sensor ketika sistem telah dihubungkan dengan baterai 9 Volt. Selama perubahan ataupun tidak terjadinya perubahan aktivitas fisik, sensor akan terus membaca percepatan linier sumbu X, sumbu Y dan sumbu Z. Dan mencari nilai *acceleration* yaitu akar dari jumlah kuadrat ketiga sumbu X,Y,Z. Sedangkan pada Gambar 4.6 merupakan kode sumber implementasi untuk mendapatkan 4 nilai *Quaternion (q0,q1,q2,q3).* Kemudian nilai tersebut digunakan untuk perhitungan nilai tetha atau sudut rotasi gerakan.

4.6 Implementasi Algoritma *Fall Detection* dengan *Threshold*

Gambar 4.7 merupakan kode sumber implementasi algoritma *Fall Detection* dengan menggunakan *threshold*. Pada awalnya tiap-tiap sensor akan menginisialisasi sumbu masing-masing yang dimiliki, setiap data yang diambil disimpan dengan menggunakan tipe data *float* termasuk penghitungan nilai *α.* Kemudian setiap data yang masuk dicari nilai perubahan pada tiap-tiap sumbu pada *function* delta xyz, dan disimpan pada tipe data *float* juga.

Kemudian tiap nilai *α* yang masuk dilakukan pegecekan apakah nilai *α* melebihi dari nilai *α treshsold* jika melebihi maka

terjadi pergerakan dinamis kemudian dari nilai masing-masing perubahan tiap sumbu *dX dan dZ* dilakukan pengecekan kembali dengan nilai *threshold*. Jika nilai *dX dan dZ* melebihi dari nilai *threshold* maka terjadi *potensial fall* dan untuk memastikan bahwa memang terjadi insiden terjatuh maka dilakukan pengecekan pada postur dengan menggunakan tetha, jika nilai tetha kurang dari 45° maka dipastikan *user* mengalami insiden terjatuh.

```
Prosedur Sistem Fall Detection Mendeteksi Terjadinya Insiden Jatuh
Masukan: x,y,z, q0,q1,q2,q3, tetha
Fall <- false
Triger <- false
getAcceldata()
getreadQuatdata()
function deltaxyz {
   i <- 1
 if (i=1)
 then
  x1 <- 0
  x2 <- sumbu x
  xtemporary <- sumbu x
  y1 <- 0
  y2 <- sumbu y
  ytemporary <- sumbu y
  z1 <- 0
  z2 <- sumbu z
  ztemporary <- sumbu z

 else
  x1 <- xtemporary
  xtemporary <- sumbu x
  x2 <- sumbu x
  xdifference <- x2-x1
  y1 <- ytemporary
  ytemporary <- sumbu y
  y2 <- sumbu y
  ydifference <- y2-y1
  z1 <- ztemporary
  ztemporary <- sumbu z
  z2 <- sumbu z
  zdifference <- z2-z1
i++
}

if (alpha < alpha_threshold)
```

```
{
     Triger <- false
}

Else {

  if ( deltax ≥ deltax_threshold && deltaz ≥ deltaz_threshold)
  {
  Triger <- true
  Print ("potensial fall")
  }

}

If (trigger = true)
{
  if ( tetha≈45)
  FALL <- true

}
```

```
Keluaran: Fall Detected
```

Gambar 4.7 Algoritma Fall Detection Menggunakan *Threshold*

4.7 Implementasi Algoritma *Fall Detection* dengan Metode *k-NN*

Gambar 4.8 merupakan kode sumber implementasi algoritma *Fall Detection* dengan menggunakan Metode *kNN*. Pertama menginisialisasi data set dan data set target yang berasal dari *training sample* dan nilai *k*. Data set berisi nilai-nilai *dx,q0 dan q2* pada aktivitas berdiri, duduk, melompat dan jatuh yang disimpan dalam *array*, kemudian data set target ialah berisi angka 0 dan 1, angka 0 untuk aktivitas tidak jatuh dan 1 untuk aktivitas jatuh.

Pada setiap pembacaan tiap-tiap nilai sensor akan disimpan dengan menggunakan tipe data *float*. Kemudian setiap data yang masuk dicari nilai perubahan pada tiap-tiap sumbu pada *function* delta xyz, dan disimpan pada tipe data *float* juga.

Kemudian setiap data yang masuk pada nilai *dx,q0 dan q2* dilakukan penghitungan jarak dengan nilai dataset dengan

menggunakan *euclidian distance.* Setelah dilakukan penghitungan jarak dilakukan sortir dari nilai jarak yang terdekat sekaligus memasangkan dengan nilai dataset target. Jika mayoritas dalam hasil sortir 5 teratas merupakan mayoritas pada *class* target 1 maka jatuh terdeteksi.

```
Prosedur Sistem Fall Detection Mendeteksi Terjadinya Insiden Jatuh dengan k-NN
Masukan: x,y,z,q0,q1,q2,q3,tetha
Inisialisasi dataset,
Inisialisasi dataset target,
K <- 5

getAcceldata()
getreadQuatdata()
function deltaxyz {
   i <- 1
 if (i=1)
 then
  x1 <- 0
  x2 <- sumbu x
  xtemporary <- sumbu x
  y1 <- 0
  y2 <- sumbu y
  ytemporary <- sumbu y
  z1 <- 0
  z2 <- sumbu z
  ztemporary <- sumbu z

 else
  x1 <- xtemporary
  xtemporary <- sumbu x
  x2 <- sumbu x
  xdifference <- x2-x1
  y1 <- ytemporary
  ytemporary <- sumbu y
  y2 <- sumbu y
  ydifference <- y2-y1
  z1 <- ztemporary
  ztemporary <- sumbu z
  z2 <- sumbu z
  zdifference <- z2-z1
i++
}

function jarak{
 for j to length dataset do
```

```
      jarak   =   sqrt   (pow((dx-dx dataset),2)   +   pow((q0-
                q0 dataset),2)+ pow((q2-q2 dataset),2))
  end for
}

Sort ascending jarak()
Menghitung i pada jarak terdekat dari class
if
for i to length k do
if jarak = class jatuh, jatuh+=jatuh 1
else if jarak = class tidak terjatuh, tidak jatuh+=1

if jatuh > tidak jatuh
   print jatuh terdeteksi
```

```
Keluaran: Fall Detected
```

Gambar 4.8 Kode Sumber Algoritma Fall Detection Menggunakan Metode kNN

4.8 Implementasi Perangkat Lunak untuk Mendapatkan Nilai GPS

Implementasi perangkat lunak untuk mendapatkan nilai GPS pada SIM908 V3.0 *Shield* ialah dengan menggunakan AT Command, AT Command digunakan pada saat menginisialisasi dan mengaktifkan GPS. Berikut macam-mcam AT *Command* yang digunakan untuk mengaktifkan GPS

- AT+CGPSIPR : Digunakan untuk mengatur nilai dai *baud rate.*
- AT+CGPSPWR : Digunakan untuk menyalakan *power supply* dari GPS.
- AT+CGPSRST : Digunakan untuk reset GPS dari *autonomy mode*.
- AT+CGPSSTATUS : Digunakan untuk memeriksa status GPS.

  Perintah AT *Command* tersebut kemudian dibungkus didalam kode program pada arduino untuk inisialisasi dan

mengaktifkan GPS. Implementasi perangkat lunak untuk mendapatkan nilai GPS dapat dilihat pada Gambar 4.9.

Prosedur Sistem *Fall Detection* Mendapatkan Nilai Posisi pada GPS

Masukan: -

```
1   function Start GPS{
2     AT+CGPSIPR=9600;
3     AT+CGPSPWR=1
4     AT+CGPSRST=1
5     AT+CGPSSTATUS?
6   }
7
8   function setup{
9     Start_GPS()
10    Enable GPS mode
11    Enable GSM mode
12  }
13
14  function loop{
15   latitude = getlat()
16   longitude = getlon()
17   }
```

Keluaran: data koordinat lokasi latitude, longitude

Gambar 4.9 Implementasi Mendapatkan Nilai GPS

4.9 Implementasi Perangkat untuk Mengirim SMS

Implementasi perangkat lunak pada GSM *Shield* Arduino SIM908 seperti yang sudah dijelaskan sebelumnya, dapat dilakukan pengujian perintah AT pada AT Command Tester. Pada Tugas Akhir ini digunakan *baud rate* 19200 untuk Arduino dan GSM Shield Arduino SIM908. *Baud rate* adalah banyaknya simbol yang ditransfer selama 1 detik. Untuk mengirim SMS dapat menggunakan perintah AT sebagai berikut [15]:

- AT+CMGF: memberikan spesifikasi format input dan output SMS. 0 untuk Mode PDU, sedangkan 1 untuk mode teks. Respon yang diberikan berupa “OK”.

- AT+CMGS=\" <number>": untuk memasukkan nomor tujuan dalam pengiriman SMS, diawali dengan kode internasional, untuk Indonesia menggunakan kode 62

Pengiriman SMS yang dilakukan GSM Shield digunakan untuk mengirim SMS dari Arduino untuk keluarga korban dengan format latitude: <koordinat> longitude: <koordinat> link:<alamat url *Google Maps*>. Implementasi perangkat lunak pada GSM Shield SIM908 untuk mengirim SMS ditunjukkan pada Gambar 4.10.
Yang perlu diperhatikan adalah posisi *Switch* S1 sebelum sistem akan digunakan, posisi *Switch* S1 harus berada pada "Comm" dan ketika lampu LED menyala dengan lampu warna hijau seperti yang ditunjukkan pada Gambar 4.3.

Prosedur Sistem *Fall Detection* pengiriman SMS
Masukan: Data lokasi koordinat latitude dan longitude

```
1.  function setup{
2.     Start_GPS()
3.     Enable GPS mode
4.     Enable GSM mode
5.   }
6.
7.  Function loop{
8.     if (fall=true)
9.     then
10.      AT
11.      AT+CMGF=1
12.      AT+CMGS=\"6282377778325"
13.      Fall alarm from ederly
14.
15.      Latitude
16.      getlat() untuk mencari posisi koordinat latitude
17.      Longitude:
18.      getlon () untuk mencari posisi koordinat longitude
19.      Link
20.      http://google.maps/@-/getlat(),getlon(),20z
21.    endif
```

```
Keluaran: data koordinat lokasi latitude, longitude, link
url googlemaps
```

Gambar 4.10 Implementasi Perangkat untuk Mengirim SMS

Berikut ini adalah macam-macam perintah AT yang digunakan untuk mengirimkan data ke server melalui HTTP [15]:

- AT+CREG: registrasi jaringan *SIM Card*, respon yang diberikan adalah "OK".
- AT+SAPBR: melakukan konfigurasi profil GPRS berdasarkan IP APN *SIM Card*, respon yang diberikan adalah "OK".
- AT+HTTPINIT: inisialisasi layanan HTTP, respon yang diberikan adalah "OK".
- AT+HTTPPARA: menetapkan nilai parameter HTTP, respon yang diberikan adalah "OK".
- AT+HTTPACTION: HTTP *Method Action*, respon yang diberikan adalah "OK".
- AT+HTTPTERM: memutus layanan HTTP, respon yang diberikan adalah "OK".

Perintah AT tersebut kemudian dibungkus didalam kode program pada Arduino implementasi perangkat lunak untuk mengirim data ditunjukkan oleh Gambar 4.11.

```
Prosedur Sistem Fall Detection Mengirim Data ke Server
Masukan: -
```

```
1    function Start_GPS{
2       Serial.println("AT+CREG?")
3       AT+SAPBR=3,1\"APN\",\"3gprs\"
4       AT+SAPBR=3,1\"USER\",\"3gprs\"
5       AT+SAPBR=3,1\"PWD\",\"3gprs\"
6       AT+SAPBR=3,1\"Contype\",\"GPRS\"
7       AT+SAPBR=1,1
8       AT+HTTPINIT
9       AT+HTTPPARA=\"CID",1"
10      AT+CGPSPWR=1
```

```
11      AT+CGPRST=1
12      AT+CGPSSTATUS?
13   }
14
15   Function setup{
16      Start_GSM()
17      Enable GSM mode
18      Disable GPS mode
19   }
20   Function loop{
21      If(fall=true)
22      then
23         lat=getlat()
24         lon=getlon()
25   AT+HTTPPARA=\"URL\",\"satrio.yusufnugroho.com.Getdatala
     t =")
26      Print("&lat")
27      Print ("&lon")
28      AT+HTTPACTION=0
29   endif
```

Keluaran: data koordinat lokasi latitude, longitude

Gambar 4.11 Implementasi Perangkat Mengirim ke Server

4.10 Implementasi Data pada Server

Pada saat pengguna mengalami insiden terjatuh, rangkaian pada perangkat akan mengirimkan data lokasi ke server. Implementasi penerimaan data tersebut sehingga data tersebut masuk ke *database* server adalah seperti yang ditunjukkan pada Gambar 4.12.

Prosedur Sistem *Fall Detection* Penerimaan Data pada Server

Masukan: latitude, longitud

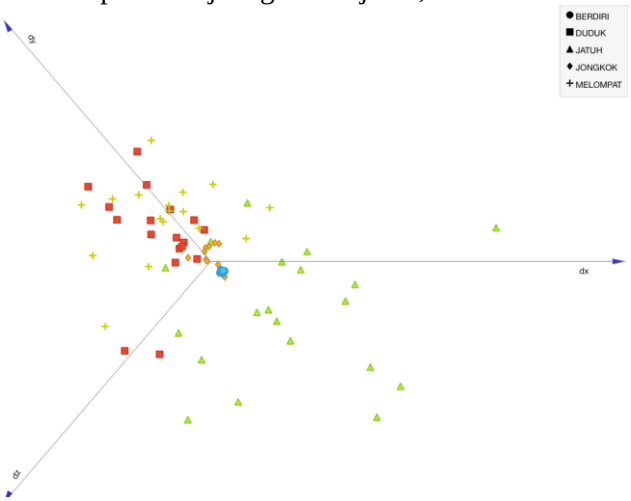
```
parse_string(substring, $_Get <- Variable)
latitude <- $_GET['latitude']
longitude <- $ GET['longitude']
```

Keluaran: latitude, longitud

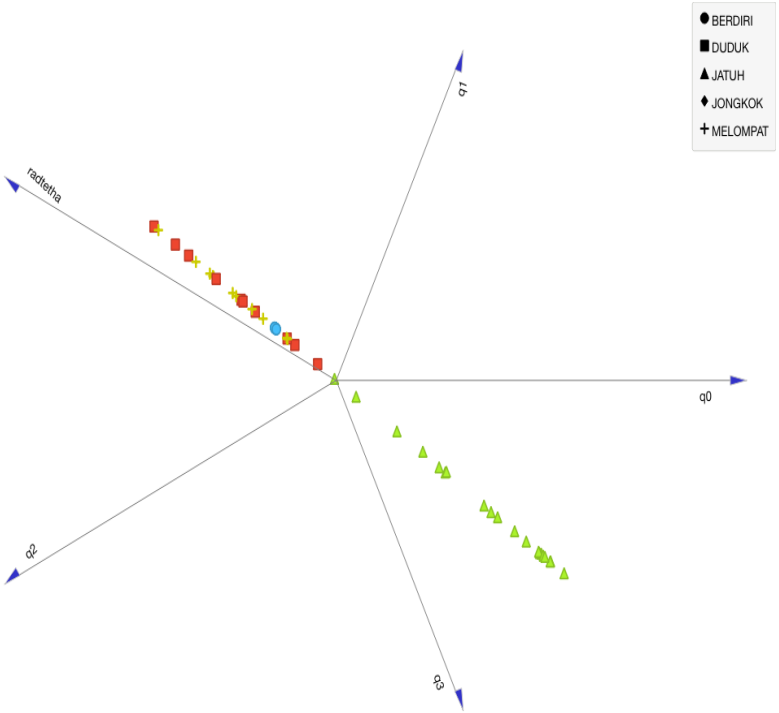
Gambar 4.12 Impelementasi Penerimaan Data pada Server

4.11 Implementasi *Data Training*

Pada subbab ini akan dibahas mengenai *data training* yang didapatkan dari perekaman sensor *accelerometer* beserta nilai *quaternion*. Data *training* sensor *accelerometer* akan digunakan untuk menentukan nilai *threshold*, *alpha* (α), dan *theta* (θ) pada *decision tree* dan untuk dianalisa sehingga pada metode KNN dapat ditentukan fitur variabel mana saja yang digunakan. Nilai *accelerometer* pada 9 axis motion Shield disimpan dalam sebuah m*icro SD*. Khusus untuk data training perangkat ditambahkan sebuah micro SD beserta dengan *adapter*. Pada saat merekam data sensor *accelerometer*, posisi perangkat berada pada dada *user*. Pada Gambar 4.13 dan Gambar 4.14 merupakan gambar linier projection data dx,dy, dan dz serta *q0,q1,q2,q3* dari berbagai aktivitas yaitu aktivitas berdiri, duduk, jatuh, jongkok dan melompat, linier projection berguna dalam menentukan fitur mana saja yang dapat digunakan sebagai variabel *threshold* sebagai penentuan apakah terjadi gerakan jatuh,



Gambar 4.13 Linier Projection data sensor dx,dy dan dz dalam berbagai aktivitas



Gambar 4.14 Linier Projection data sensor q0,q1,q2,q3 dan tetha dalam berbagai aktivitas

4.11.1 Implementasi Data *Training* Sensor *Accelerometer* Kondisi *User* Normal

Pada penelitian ini diambil data sensor ketika *user* dalam keadaan normal. Keadaan normal ialah ketika *user* sedang tidak melakukan aktivitas fisik (statis) dalam kasus ini *user* dalam keadaan berdiri.

Tabel 4.2 Data *Training* Sensor *Accelerometer dan Orientasi* Kondisi *User* Normal

Dx	dy	dz	α	q0	q1	q2	q3	θ
0.0109	0.0266	0.0002	9.8351	0.7018	0.1184	0.7025	-0.0027	90.9039
0.0263	0.0464	0.0093	9.8994	0.7017	0.1174	0.7027	-0.0023	90.9143
0.0218	0.0332	0.0096	9.8631	0.7014	0.1182	0.7029	-0.0026	90.9743
0.0238	0.0098	0.0194	9.7959	0.7014	0.1158	0.7032	-0.0033	90.9623
0.0075	0.0204	0.0213	9.8094	0.7026	0.1163	0.702	-0.0032	90.7778
0.0051	0.0092	0.0225	9.7907	0.7031	0.1167	0.7014	-0.0037	90.6895
0.0095	0.0133	0.0399	9.8241	0.7012	0.1179	0.7031	-0.0022	91.0004
0.0127	0.0927	0.0485	9.9406	0.7029	0.1173	0.7015	-0.0021	90.7267
0.0069	0.0201	0.0697	9.8157	0.7	0.116	0.7047	-0.0034	91.2014
0.0109	0.0266	0.0002	9.8351	0.7018	0.1184	0.7025	-0.0027	90.9039

Dari Tabel 4.2 didapatkan rata-rata nilai sebesar α 9.84361 dan nilai maksimum sebesar 9.9131 untuk kondisi *user* normal. Hal tersebut memberi arti bahwa jika nilai α lebih dari 9.84361 maka terdapat transisi pergerakan *user* dari keadaan statis menuju transisi dinamis. Dari Tabel 4.2 didapatkan rata-rata nilai θ sebesar 90.80175.

4.11.2 Implementasi Data *Training* Sensor *Accelerometer* Kondisi *User* Duduk

Pada penelitian ini diambil data sensor ketika *user* dalam keadaan berdiri hingga posisi duduk. Ketika dalam aktivitas transisi menuju ke posisi duduk sensor akan merekam nilai sumbu X, Sumbu Y, Sumbu Z, dan menghitung nilai *alpha* serta *theta.*

Tabel 4.3 Data *Training* Sensor *Accelerometer* Kondisi *User Duduk*

dx	dy	dz	*α*	q0	q1	q2	q3	θ
0.3537	0.9356	0.6219	10.9734	0.7338	0.0862	0.6723	0.0463	85.6309
1.185	2.6416	0.9224	14.2022	0.6675	-0.0547	0.7238	-0.1658	96.2936
0.5822	1.1557	1.0357	9.1597	0.6415	0.076	0.7626	0.0322	100.243
1.1984	2.6278	2.1634	12.1414	0.5155	0.0294	0.8562	0.0173	117.9973
1.4174	0.8487	2.5363	11.9391	0.7213	0.0655	0.6877	0.0493	87.7184
1.1364	3.4961	2.753	10.035	0.542	0.0184	0.8401	0.0008	114.4173
0.1422	2.8642	2.8422	11.7909	0.7693	0.0894	0.6292	-0.0651	79.4462
0.7152	2.3275	3.0026	10.8424	0.6453	0.0331	0.7629	-0.0229	99.6829
0.6469	2.3057	3.5033	10.2282	0.4714	-0.0076	0.8815	0.0236	123.8066
1.0992	2.8581	5.4541	11.3385	0.5955	0.0089	0.8032	-0.007	106.9522

Dari Tabel 4.3 didapatkan rata-rata nilai ***α*** sebesar 16.13848 dan nilai maksimum sebesar 21.8961 untuk kondisi *user* dalam keadaan berdiri hingga posisi duduk. Hal tersebut memberi arti bahwa jika nilai *α* lebih dari 16,13848 dan nilai tetha 100.258 maka terjadi pergerakan *user* dari keadaan keadaan berdiri menuju posisi duduk.

4.11.3 Implementasi Data *Training* Sensor *Accelerometer* Kondisi *User* Jatuh

Pada penelitian ini diambil data sensor ketika *user* dalam keadaan berdiri hingga keadaan jatuh. Yang dimaksud jatuh ialah ketika *user* mengalami perubahan aktivitas dari keadaan berdiri hingga berbaring ke lantai.

Tabel 4.4 Data *Training* Sensor *Accelerometer* Kondisi *User Jatuh*

dx	dy	dz	α	q0	q1	q2	q3	θ
5.6598	2.5427	0.1531	7.4167	0.9745	0.0213	0.2066	-0.0848	25.9415
17.1488	5.8465	0.5045	18.5382	0.9247	0.0138	0.3678	-0.0973	44.777
4.9639	1.0457	0.9049	9.3576	0.9901	0.0217	0.0928	-0.1036	16.1919
6.5329	9.4789	1.0709	7.5314	0.9867	0.0247	0.1262	-0.0996	18.7313
2.8714	5.2942	1.6862	6.44	0.986	-0.0399	0.1616	-0.0009	19.1742
9.2575	1.6598	3.336	11.0427	0.8246	-0.0782	0.5002	-0.2525	68.9421
10.3114	7.8473	6.725	19.3476	0.875	0.0758	0.4733	-0.0674	57.9318
4.0758	7.1972	6.8068	16.6323	0.9647	-0.1606	0.162	-0.1317	30.5592
11.4341	3.2903	7.0327	17.0694	0.9857	0.0109	0.152	-0.072	19.416
6.8846	2.5671	7.7025	15.1181	0.9899	-0.0043	0.1155	-0.0814	16.2631

Dari Tabel 4.4 didapatkan rata-rata nilai $\boldsymbol{\alpha}$ sebesar 19.85675 dan nilai maksimum sebesar 33.2638 untuk kondisi *user* dsaat terjadi pergerakan *users* dari keadaan statis berdiri hingga terjatuh berabring di lantai. Hal tersebut memberi arti bahwa jika nilai α lebih dari 19.85675 dan maksimum nilai $\boldsymbol{\alpha}$ adalah 33.2638 serta nilai θ mendekati 48.78413 maka terjadi insiden terjatuh.

4.11.4 Implementasi Data *Training* Sensor *Accelerometer* Kondisi *User* Melompat

Pada penelitian ini diambil data sensor ketika *user* dalam keadaan berjalan santai. Dari Tabel 4.4 didapatkan rata-rata nilai $\boldsymbol{\alpha}$ sebesar 20.47505 dan nilai maksimum sebesar 29.1549 untuk kondisi *user* saat sedang melompat. Hal tersebut memberi arti bahwa jika nilai α lebih dari 20.47505 dan nilai θ 100.7004 maka terjadi pergerakan *user* saat sedang melompat.

Tabel 4.5 Data *Training* Sensor *Accelerometer* Kondisi *User* Melompat

dx	dy	dz	α	q0	q1	q2	q3	θ
0.9053	0.3754	0.4381	15.1142	0.6808	0.0196	0.7322	0.0059	94.231
1.6068	12.7544	0.8494	23.0101	0.5839	-0.0506	0.8055	-0.0881	108.6107
1.1704	4.2219	1.0938	15.6333	0.6614	0.0392	0.7486	-0.0236	97.2296
0.4895	7.8908	2.2063	14.7016	0.5907	0.0173	0.8067	0.0025	107.6413
0.1464	15.1204	2.2834	23.6246	0.6332	0.0757	0.7668	0.0724	101.4734
0.1615	2.856	3.3843	18.8374	0.5893	0.0107	0.8077	-0.0142	107.8418
3.803	8.8955	3.4875	11.2145	0.7205	0.0752	0.6877	-0.0482	87.8617
1.504	9.379	3.785	20.5779	0.6264	0.0214	0.7789	0.0239	102.4835
1.7278	9.8772	4.8424	20.7024	0.5565	0.0012	0.8307	-0.0133	112.4276
0.1571	4.3921	5.0685	14.8082	0.4807	-0.0106	0.8768	-0.0114	122.5998

4.12 Implementasi Basis Data Sistem

Sesuai dengan tahap perancangan di Bab sebelumnya, bahwa sistem ini hanya menggunakan 1 tabel yang digunakan untuk menyimpan *history* insiden terjatuh yang terjadi sebelum-sebelumnya. Pada gambar Gambar 4.15 terdapat 4 atribut yaitu ide_fall, lat, lon, serta waktu jatuh. Data lat dan lon berasal dari sensor *accelerometer* yang berfungsi untuk menyimpan koordinat *latitude* dan *longitude.* Sedangkan waktu_jatuh diambil dari waktu penerimaan data tersebut saat masuk ke server menggunakan fungsi pada php.

#	Nama	Jenis	Penyortiran	Atribut	Kosong	Bawaan	Ekstra
1	id_fall	int(11)			Tidak	*Tidak ada*	AUTO_INCREMENT
2	lat	float			Tidak	*Tidak ada*	
3	lon	float			Tidak	*Tidak ada*	
4	waktu_jatuh	date			Tidak	*Tidak ada*	

Gambar 4.15 Basis Data Sistem *Fall Detection*

BAB V
PENGUJIAN DAN EVALUASI

Bab ini membahas pengujian dan evaluasi pada sistem *fall detection* yang dikembangkan. Sistem akan diuji coba fungsionalitas dan performa dengan menjalankan skenario yang sudah ditentukan. Hasil evaluasi menjabarkan tentang rangkuman hasil pengujian pada bagian akhir bab ini. Uji coba akan dilakukan dengan dua tipe pengujian yaitu uji coba fungsionalitas dan uji coba performa. Pengujian fungsionalitas meliputi uji coba setiap bagian perangkat keras yang dirangkai pada Arduino dan uji coba keseluruhan sistem. Pengujian performa meliputi akurasi pendeteksian insiden jatuh dan keakuratan lokasi.

5.1. Uji Coba Fungsionalitas

Uji coba fungsionalitas dilakukan untuk mengetahui fungsi dasar masing-masing komponen baik perangkat keras yang dirangkai berjalan sebagaimana mestinya.

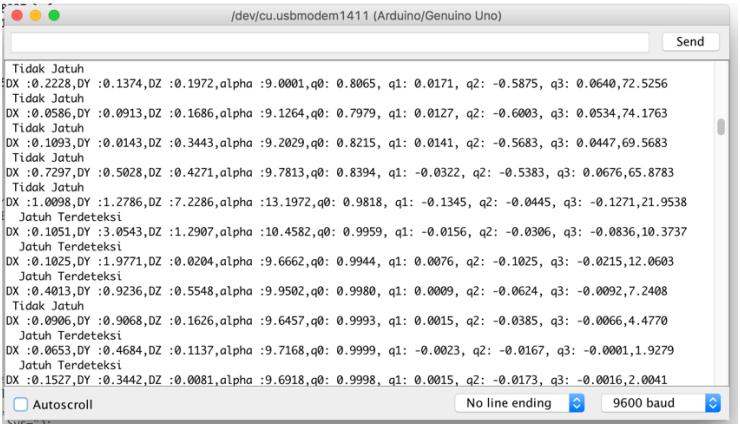
5.1.1 Lingkungan Uji Coba

Pada subbab ini akan dijelaskan mengenai lingkungan uji coba sistem *fall detection*. *Sensor* Orientasi *9 axis Motion Shield* dihubungkan dengan Arduino dengan GPRS/GPS/GSM SIM908 *Shield*. Kemudian perangkat tersebut diletakkan di bagian saku pengguna. Apabila *users* tersebut mengalami insiden jatuh maka perangkat tersebut mengirim SMS dan mengirimkan data lokasi ke server. Pengiriman data yang dilakukan melalui jaringan GPRS yang didapatkan dari jaringan *SIM Card* yang berada di GSM *Shield* Arduino SIM908.

5.1.2 Uji Coba Sistem *Fall Detection*

Uji coba ini dilakukan untuk melihat apakah sistem dapat bekerja sesuai dengan fungsionalitasnya. Sistem akan merekam

setiap aktivitas fisik *user* yang ditunjukkan pada Gambar 5.1 Setelah itu dilakukan skenario insiden terjatuh seperti pada Tabel 5.1.



Gambar 5.1 Sensor *accelerometer* merekam aktivitas fisik *user*

Tabel 5.1 Skenario Uji Coba Mendeteksi Gerakan Jatuh

Nama	Uji coba mendeteksi gerakan terjatuh
Tujuan	Sistem dapat mendeteksi gerakan terjatuh
Skenario	Perangkat diletakkan pada bagian dada dengan menggunakan belt, kemudian *user* berjalan dan terjadi insiden terjatuh
Hasil Uji Coba	Dikirimkannya data lokasi melalui SMS dan ke server

Ketika *user* terdeteksi oleh sensor mengalami insiden terjatuh, sistem akan mengirimkan SMS kepada kerabat korban. SMS yang dikirimkan korban SMS tersebut berisi koordinat *latitude* dan *longitude* serta alamat *url* *Google Maps* posisi korban terjatuh. Untuk skenario uji coba pengiriman SMS dari *Shield* SIM908 dapat dilihat pada Tabel 5.2. Hasil uji coba mendapatkan SMS tersebut dapat dilihat pada Gambar 5.2

Tabel 5.2 Skenario Uji Coba Mengirimkan Koordinat Lokasi melalui SMS

Nama	Uji coba mengirimkan SMS dari *Shield* SIM908
Tujuan	Untuk memberikan informasi kepada kerabat bahwa pengguna sedang mengalami insiden terjatuh beserta lokasi terjadinya insiden.
Skenario	Pengguna mengalami insiden terjatuh, kemudian sistem akan mengirimkan SMS
Hasil Uji Coba	Data koordinat lokasi *latitude* dan *longitude, url* lokasi jatuh pada *Google Maps*

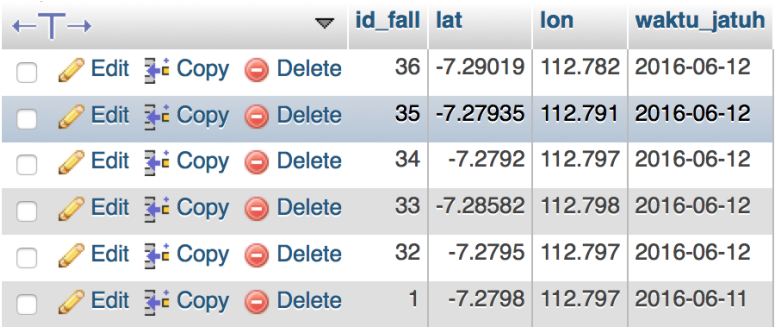
Setelah mengirimkan SMS kepada kerabat korban, selanjutnya sistem akan mengirimkan data posisi koordinat *latitude* dan *longitude* ke Server. Untuk skenario uji coba pengiriman koordinat lokasi ke server dapat dilihat pada Tabel 5.3, hasil uji coba skenario tersebut dapat dilihat pada Gambar 5.4. sedangkan pada Gambar 5.3 merupakan gambar ketika link pada SMS dibuka dengan menggunakan *Google Maps*, sehingga dapat memudahkan pengguna untuk mencari lokasi kejadian.

Tabel 5.3 Mengirimkan Data Lokasi ke Server

Nama	Uji coba mengirimkan Koordinat Lokasi ke Server dari Shield SIM908
Tujuan	Menyimpan data riwayat lokasi jatuh pengguna
Skenario	Setelah sistem mengirimkan SMS, sistem mengirimkan data lokasi jatuh ke server
Hasil Uji Coba	Data riwayat jatuh pengguna beserta lokasi insiden jatuh

Gambar 5.2 Pengiriman SMS ke Kerabat Korban

Gambar 5.3 Lokasi Ketika Link SMS dibuka dengan *Google Maps*



	id_fall	lat	lon	waktu_jatuh
Edit Copy Delete	36	-7.29019	112.782	2016-06-12
Edit Copy Delete	35	-7.27935	112.791	2016-06-12
Edit Copy Delete	34	-7.2792	112.797	2016-06-12
Edit Copy Delete	33	-7.28582	112.798	2016-06-12
Edit Copy Delete	32	-7.2795	112.797	2016-06-12
Edit Copy Delete	1	-7.2798	112.797	2016-06-11

Gambar 5.4 *Data* yang berhasil terkirim ke server disimpan pada tabel fall

5.1.1. Uji Coba Menampilkan Data Riwayat Lokasi Jatuh

Uji Coba ini dilakukan untuk mendapatkan data pada basis data server, setelah pengiriman data koordinat lokasi terkirim ke basis data server. Pengguna dapat menampilkan data log riwayat lokasi jatuh yang disimpan di basis data server dalam bentuk tabel dan dapat menampilkannya ke *Google Maps* apabila memilih salah satu log yang tersedia.

Skenario uji coba menampilkan Riwayat Lokasi Jatuh dapat dilihat pada Tabel 5.4 dan dapat hasilnya dapat dilihat pada Gambar 5.5, Sedangkan untuk Gambar 5.6 merupakan titik lokasi jatuh ketika *text hyperlink* pada kolom paling kanan di buka, sehingga titik tersebut akan dibuka pada *Google Maps.*

Tabel 5.4 Skenario Uji Coba Menampilkan Riwayat Jatuh

Nama	Uji coba menampilkan riwayat lokasi Jatuh
Tujuan	Melihat riwayat lokasi jatuh pengguna
Skenario	Pengguna memilih riwayat lokasi jatuh pengguna
Hasil Uji Coba	Data *riwayat lokasi* jatuh ditampilkan dalam bentuk tabel dan dapat dilihati lokasinya menggunakan *Google Maps*

Data Base Fall Detection

Nomor	Koordinat Latitude	Koordinat Longitude	Waktu Jatuh	Link
36	-7.29019	112.782	2016-06-12	Open in Maps
35	-7.27935	112.791	2016-06-12	Open in Maps
34	-7.2792	112.797	2016-06-12	Open in Maps
33	-7.28582	112.798	2016-06-12	Open in Maps
32	-7.2795	112.797	2016-06-12	Open in Maps
1	-7.2798	112.797	2016-06-11	Open in Maps

Gambar 5.5 Daftar Riwayat Lokasi Jatuh pada Website

Gambar 5.6 Titik Lokasi Jatuh dibuka dengan *Google Maps*

5.2 Uji Coba Performa

Uji coba performa sistem dilakukan untuk mengetes kehandalan sistem. Adapun performa sistem yang diujicobakan adalah uji coba akurasi deteksi dan uji coba lama proses deteksi.

5.2.1 Uji Coba Akurasi Deteksi

Uji coba akurasi deteksi dilakukan untuk mengetahui tingkat akurasi algoritma *fall detection* ditinjau dari beberapa aktivitas fisik *user* yang dilakukan masing-masing sebanyak 20 kali percobaan. Aktivitas fisik *user* yang akan diuji coba adalah aktivitas fisik *user* pada saat terjatuh dan aktivitas fisik *user* pada saat tidak terjatuh. Uji coba ini menggunakan perhitungan *true positive*, *true negative*, *false positive*, dan *false negative*.

Uji coba ini dilakukan sebanyak 20 kali pada setiap aktivitas fisik *user*. Adapun jenis uji coba kasus berdasarkan perhitungan *true positive*, *true negative*, *false positive*, dan *false negative* adalah sebagai berikut:

- ***True Positive***
  Uji coba dikatakan *true positive* ketika aktivitas jatuh terdeteksi oleh sistem sebagai aktivitas jatuh.
- ***True Negative***
  Uji coba dikatakan *true negative* ketika aktivitas tidak jatuh terdeteksi oleh sistem sebagai aktivitas tidak jatuh.
- ***False Positive***
  Uji coba dikatakan *false positive* ketika aktivitas tidak jatuh namun terdeteksi oleh sistem sebagai aktivitas jatuh.
- ***False Negative***
  Uji coba dikatakan *false negative* ketika aktivitas jatuh namun terdeteksi oleh sistem sebagai aktivitas tidak jatuh.

Adapun aktivitas fisik *user* yang dilakukan uji coba adalah sebagai berikut:

- Aktivitas jatuh (aktivitias *user* dikatakan jatuh ketika *user* dalam keadaan statis menuju keadaan transisi dinamis, yaitu dari keadaan berdiri, atau berjalan menuju keadaan berbaring atau tergeletak).
- Aktivitas duduk secara cepat (aktivitas *user* dikatakan duduk secara cepat ketika *user* dalam keadaan berdiri atau berjalan kemudian duduk secara tiba-tiba).

- Aktivitas melompat (aktivitas *user* dikatakan melompat ketika *user* dalam keadaan berdiri kemudian melakukan gerakan dinamis seperti mengangkat kedua kaki ke arah atas dan dengan cepat menurukannya lagi).
- Aktivitas berjalan (aktivitas *user* dikatakan berjalan ketika *user* dalam keadaan berjalan dari awal hingga akhir).

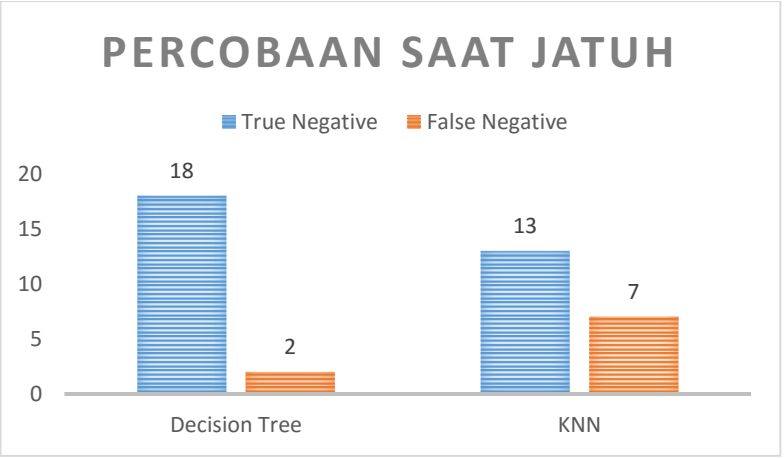
5.2.1.1 Uji Coba Akurasi Pendeteksian Gerakan Jatuh

Pada subbab ini akan dibahas mengenai uji coba akurasi pendeteksian Gerakan Jatuh. Uji coba ini dilakukan untuk membandingkan tingkat akurasi dari gerakan jatuh dengan menggunakan prediksi *Decision Tree* dan metode *k-NN*.

Tabel 5.5 merupakan tabel uji coba perhitungan akurasi aktivitas fisik *user* pada saat terjatuh dengan paramater *true positive* dan *false negative*. Uji coba perhitungan akurasi jatuh menggunakan parameter *true positive* dan *false negative* untuk mendapatkan nilai sensitifitas. Uji coba aktivitas fisik *user* pada saat terjatuh dilakukan pada *user* dengan tinggi badan 168 cm.

Tabel 5.5 Uji Coba Perhitungan Akurasi Aktivitas Fisik User pada Saat Terjatuh

No.	Metode	Jumlah Percobaan	True Positive (TP)	False Negative (FN)	% TP	% FN
1	Decision Tree	20	18	2	90	10
2	*k*-NN	20	13	7	65	15
		Jumlah			155	45



Gambar 5.7 Gambar Grafik Hasil Uji Coba Percobaan Jatuh

Dari Tabel 5.5 diketahui bahwa aktivitas fisik *user* dengan kondisi terjatuh yang diuji cobakan adalah aktivitas fisik *user* saat terjatuh berbaring ke lantai memiliki *true positive sebesar 77,5% dan false negatif sebesar 22,5%*, Nilai *true positive*, *false negative*, dan sensitifitas adalah seperti berikut:

$$\text{Rata-rata \% TP} = \frac{\text{jumlah nilai persentase TP}}{(\text{jumlah aktivitas x } 100)}$$
$$= \frac{155}{200}\ x\ 100\%$$
$$= 77.5\%$$
$$\text{Rata-rata \% FN} = \frac{\text{jumlah nilai persentase FN}}{(\text{jumlah aktivitas x } 100)}\ x\ 100\%$$
$$= \frac{45}{200}\ x\ 100\%$$
$$= 22.5\%$$

5.2.1.2 Uji Coba Akurasi Pendeteksian Gerakan Tidak Jatuh

Tabel 5.6 merupakan tabel uji coba perhitungan akurasi aktivitas fisik *user* pada saat tidak terjatuh dengan paramater *true negative* dan *false positive dengan dilakukan uji coba sebanyak 20*

kali pada masing-masing aktivitas. Uji coba perhitungan akurasi aktivitas fisik *user* pada saat tidak terjatuh menggunakan parameter *true negative* dan *false positive* untuk mendapatkan nilai spesifitas. Nilai spesifitas mengukur tingkat proporsi dari keadaan yang salah dari suatu percobaan. Uji coba jatuh dilakukan pada *user* dengan tinggi badan 168 cm. Dari Tabel 5.6 diketahui bahwa aktivitas fisik *user* pada saat tidak terjatuh yang diujicobakan yaitu aktivitas duduk secara cepat,berdiri, melompat, dan berjalan .

Tabel 5.6 Uji Coba Perhitungan Akurasi Aktivitas Fisik User pada Saat Tidak Terjatuh dengan Parameter True Negative dan False Positive

No	Aktivitas	Jumlah Percobaan	True Negative (TN)	False Positive (FP)	% TN	% FP
1	Duduk secara cepat	20	16	4	80	20
2	Melompat	20	17	3	85	15
3	Berdiri	20	20	0	100	0
4	Berjalan	20	15	5	75	5
		Jumlah			340	60

Adapun rata-rata nilai *true negative* (TN) dan *false positive* (FP) dari Tabel 5.2 adalah seperti berikut:

$$\text{Rata-rata \% TN} = \frac{\text{jumlah nilai persentase TN}}{\text{jumlah aktivitas}} \; x \; 100\%$$

$$= \frac{(80+85+100+75)}{5} \; x \; 100\%$$

$$= \frac{340}{6} \; x \; 100\%$$

$$= 85 \; \%$$

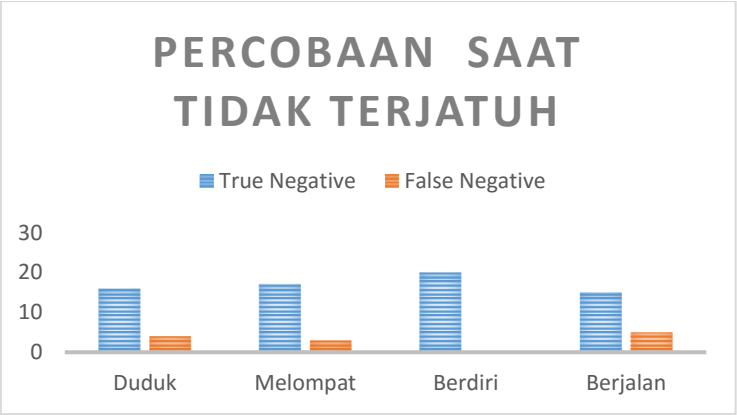
$$\text{Rata-rata \% FP} = \frac{\text{jumlah nilai persentase FP}}{(\text{jumlah aktivitas x } 100)}$$

$$= \frac{60}{4}$$
$$= 15\ \%$$

$$\text{Spesifitas} = \frac{\text{jumlah nilai persentase TN}}{\text{jumlah nilai persentase (FP+TN)}} \times 100\%$$

$$= \frac{85}{(85+15)} \times 100\%$$
$$= 85\%$$

Tabel 5.6 merupakan tabel uji coba aktivitas fisik *user* pada saat tidak terjatuh dengan masing-masing uji coba sebanyak 20 kali. Berdasarkan informasi dalam Gambar 5.8 menunjukkan bahwa untuk masing-masing aktivitas memiliki nilai persentase *true positive* dan *false negative* yang berbeda-beda. Empat aktivitas yang terdiri atas aktivitas duduk secara cepat, aktivitas , aktivitas berjalan, aktivitas melompat, dan aktivitas berdiri memiliki nilai *presentase true negative* sebesar 85% dan nilai *false positive* sebesar 15%. Nilai *true negative* sebesar 100% pada aktivitas berdiri menandakan bahwa aplikasi mendeteksi aktivitas berdiri sebagai aktivitas fisik *user* dengan kondisi tidak jatuh secara akurat. Pada Aktivitas berjalan yang memiliki nilai *true negative* sebesar 75% dan nilai *false positive* sebesar 25% sebagai presentasi terkecil dalam pendeteksian aktivitas tidak terjatuh.. Data yang didapatkan dari hasil percobaan menunjukkan bahwa aktivitas berjalan akan terdeteksi sebagai aktivitas *user* dalam kondisi duduk sebanyak 5 kali dari 20 percobaan. Aktivitas melompat *true negative* sebesar 85% dan nilai *false positive* sebesar 15%. Hal tersebut menandakan bahwa aplikasi mendeteksi aktivitas melompat sebagai aktivitas fisik *user* dalam kondisi tidak jatuh sebanyak 17 kali dari 20 pecobaan yang ada dan mendeteksi sebagai keadaan tidak terjatuh sebagai aktivitas duduk sebanyak 3 kali dari 20 percobaan yang ada.



Gambar 5.8 Gambar Grafik Hasil Uji Coba Percobaan Pada Saat Tidak Jatuh

5.3 Hasil Uji Coba Nilai K pada Metode *k-Nearest Neighbors*

Pendeteksian insiden jatuh menggunakan metode *k*NN akan di uji coba sebanyak 20 kali percobaan pada masing – masing nilai K, untuk menentukan nilai K mana yang akan dipilih dalam sistem *fall detection* ini, dari hasil percobaan tersebut yang dapat dilihat dari Gambar 5.7 didapatkan hasil bahwa K yang memiliki presentase akurasi terbesar dalam mendeteksi insiden terjatuh merupakan K dengan nilai 5.

Tabel 5.7 Hasil Uji Coba Nilai K untuk Mendeteksi Insiden Jatuh dalam Metode *k*NN

Nilai *K*	Presentase Akurasi
3	45%
4	45%
5	60%

BAB VI
KESIMPULAN DAN SARAN

Bab ini membahas mengenai kesimpulan yang dapat diambil dari tujuan pembuatan perangkat lunak dan hasil uji coba yang telah dilakukan sebagai jawaban dari rumusan masalah yang dikemukakan. Selain kesimpulan, terdapat pula saran yang ditujukan untuk pengembangan perangkat lunak lebih lanjut.

6.1. Kesimpulan

Dalam proses pengerjaan Tugas Akhir mulai dari tahap analisis, desain, implementasi, hingga pengujian didapatkan kesimpulan sebagai berikut:

1. Nilai quaternion (*q0,q1,q2,q3*) dan nilai perubahan percepatan (*dx,dy,dz*) dapat digunakan untuk mendeteksi seseorang dalam insiden jatuh dengan menggunakan metode *Decision Tree* dan metode *k*-NN.
2. Dengan mengunakan Decision Tree akurasi dalam pendeteksian orang jatuh sebesar 90%, sedangkan dengan mengunakan metode *k*-NN hanya 65%.
3. Sensor orientasi *motion Shield* yang digunakan dapat mendeteksi seseorang dalam keadaan terjatuh, menggunakan analisis sudut rotasi sebelum dan sesudah terjadinya insiden terjatuh.
4. Sensor *accelerometer* mendeteksi percepatan pada sumbu x,y,z. Pada keadaan normal nilai percepatan berkisar antara 9.8 m/s^2
5. Pengiriman data lokasi jatuh melalui SMS kepada kerabat korban dan pengiriman data ke server membutuhkan waktu paling lama 5 detik.

6. Tingkat akurasi sistem *fall detection* untuk mendeteksi gerakan selain jatuh dengan akurasi 85 %, pendeteksian paling rendah untuk mendeteksi berjalan yaitu sebesar 75%.

6.2. Saran

Berikut merupakan beberapa saran untuk pengembangan sistem di masa yang akan datang, berdasarkan pada hasil perancangan, implementasi dan uji coba yang telah dilakukan.

1. Untuk meningkatkan nilai akurasi terjadinya insiden terjatuh ditambahkan sensor pada yang di tempatkan pada bagian tertentu, sehingga tidak hanya insiden jatuh saja yang bisa terdeteksi, namun semua aktivitas ADL dapat dilakukan pendeteksian secara akurat.
2. Untuk penghematan daya saat menggunakan rangkaian pada sistem ini sebaiknya digunakan DC Step Up Module karena diperlukan daya yang besar yaitu 6-12 V untuk pengiriman data ke server maupun pengiriman melalaui SMS.
3. Rangkaian ditambahkan LCD untuk mengetahui lokasi GPS telah didapatkan dan *signal* jaringan mendukung untuk dilakukan pengiriman SMS maupun pengiriman ke server.

DAFTAR PUSTAKA

[1] Della Penna R, N. Williams , . S. Arthur dan B. J. Larry , Practice guideline for the ED management of falls in community-dwelling elderly persons, 1997, p. 480–492.

[2] N. Carroll, P. Slattum dan F. Cox, The cost of falls among the community-dwelling elderly, PubMed, 2005.

[3] W. DEIDRE , U. S. L. NAYAK dan B. ISAACS, How dangerous are falls in old people at home?, PubMed.

[4] K. B. Firdaus, "APLIKASI ACCELEROMETER SEBAGAI KENDALI (JOYSTICK)," Surabaya.

[5] "ARDUINO MEGA 2560," [Online]. Available: https://www.arduino.cc/en/Main/ArduinoBoardMega2560. [Diakses 17 Juni 2016].

[6] "GPS/GPRS/GSM Module V3.0 (SKU:TEL0051)," [Online]. Available: http://www.dfrobot.com/wiki/index.php/GPS/GPRS/GSM_Module_V3.0_(SKU:TEL0051). [Diakses 17 Juni 2016].

[7] Shanghai SIMCom Wireless Solution Ltd., SIM900 AT Command Manual, Shanghai.

[8] Garmin Ltd., "What is GPS?," [Online]. Available: http://www8.garmin.com/aboutGPS/. [Diakses 17 Juni 2016].

[9] ARDUINO S.R.L, "Arduino 9 AXES MOTION SHIELD," [Online]. Available: http://www.arduino.org/products/shields/arduino-9-axes-motion-shield. [Diakses 17 Juni 2016].

[10] ARDUINO S.R.L, "Getting Started w/ Arduino on Mac OS X," [Online]. Available: https://www.arduino.cc/en/Guide/MacOSX. [Diakses 17 Juni 2016].

[11] R. Nishkam, D. Nikhil, P. Mysore dan M. Littman, "Activity Recognition from Accelerometer Data," no. Department of Computer Science Rutgers University.

[12] Microsoft, "Training and Testing Data Set," 2012. [Online]. Available: http://msdn.microsoft.com/en-us/library/bb895173.aspx. [Diakses 17 Juni 2016].

[13] H. Z. Fallin Wu, "Development of a Wareable-Sensor-Based Fall Detection System," *International Journal of Telemedicine and Applications,* no. Hindawi Publishing Corporation, p. 11, 2014.

[14] Kusrini dan T. L. Emha, Algoritma Data Mining, Yogyakarta: C.V. Andi Offset, 2009.

[15] "SIMCOM," [Online]. Available: http://www.simcom.ee/documents/gsm-gprs/sim900/SIM900_AT%20Command%20Manual_V1.09.pdf. [Diakses 13 April 2016].

LAMPIRAN A
KODE SUMBER

```
#include <Wire.h>
#include <SD.h>
#include <EEPROM.h>
#include <Adafruit_Sensor.h>
#include <Adafruit_BNO055.h>
#include <utility/imumaths.h>
double fXg = 0;
double fYg = 0;
double fZg = 0;
double
x1=0,x2,y1=0,y2,z1=0,z2,xtemp=0,ytemp=0,ztemp=0,dx,dy,d
z;
double a;
int i=1;
const float alpha = 0.5;
boolean FALL = false;
boolean triger = false;

#define BNO055_SAMPLERATE_DELAY_MS (100)

Adafruit_BNO055 bno = Adafruit_BNO055();
```

Kode Sumber A.1 Inisialisasi Variabel Global

```
double Datatransfer(char *data_buf,char num)
{
  double temp=0.0;
  unsigned char i,j;
  if(data_buf[0]=='-')
  {
    i=1;
    while(data_buf[i]!='.')
      temp=temp*10+(data_buf[i++]-0x30);
    for(j=0;j<num;j++)
      temp=temp*10+(data_buf[++i]-0x30);

    for(j=0;j<num;j++)
      temp=temp/10;
```

```
    temp=0-temp;
  }
  else//for the positive number
  {
    i=0;
    while(data_buf[i]!='.')
      temp=temp*10+(data_buf[i++]-0x30);
    for(j=0;j<num;j++)
      temp=temp*10+(data_buf[++i]-0x30);
    for(j=0;j<num;j++)
      temp=temp/10 ;
  }
  return temp;
}

void comma(char num)//get ','
{
  char val;
  char count=0;//count the number of ','

  while(1)
  {
    if(Serial.available())
    {
      val = Serial.read();
      if(val==',')
        count++;
    }
    if(count==num)//if the command is right, run return
      return;
  }
}

double decimalgps(double rawdata)
{
  int degrees = (int)(rawdata / 100);
  double minutes = rawdata - (degrees*100);
  double mindecimal = minutes / 60.0;
  double total = degrees + mindecimal;

  return total;
}
float latitude()//get latitude
{
  char i;
  char lat[10]={
    '0','0','0','0','0','0','0','0','0','0'
  };

  if( ID())
```

```
  {
    comma(2);
    while(1)
    {
      if(Serial.available())
      {
        lat[i] = Serial.read();
        i++;
      }
      if(i==10)
      {
        i=0;
        double newlat =  Datatransfer(lat,6);
        float corrected = decimalgps(newlat);
        return corrected;
      }
    }
  }
}
float longitude()//get longitude
{
  char i;
  char lon[11]={
    '0','0','0','0','0','0','0','0','0','0','0'
  };

  if( ID())
  {
    comma(4);
    while(1)
    {
      if(Serial.available())
      {
        lon[i] = Serial.read();
        i++;
      }
      if(i==11)
      {
        i=0;
        double newlon = Datatransfer(lon,6);
        float corrected = decimalgps(newlon);
        return corrected;
      }
    }
  }
}

float altitude()//get altitude data
{
  char i,flag=0;
```

```
  char alt[8]={
    '0','0','0','0','0','0','0','0'
  };

  if( ID())
  {
    comma(9);
    while(1)
    {
      if(Serial.available())
      {
        alt[i] = Serial.read();
        if(alt[i]==',')
          flag=1;
        else
          i++;
      }
      if(flag)
      {
        i=0;
        Serial.println(Datatransfer(alt,1),1);//print
  altitude data
        float
  altitude=Serial.println(Datatransfer(alt,1),1);
        return altitude;
      }
    }
  }
}
```

Kode Sumber A.2 Fungsi-fungsi untuk Mengonversi Nilai *Raw* GPS Menjadi Float

```
void start_GSM(){
     //Configuracion GPRS Claro Argentina
    Serial.println("AT");
    delay(2000);
    Serial.println("AT+CREG?");
    delay(2000);
    Serial.println("AT+SAPBR=3,1,\"APN\",\"3gprs\"");
    delay(2000);
    Serial.println("AT+SAPBR=3,1,\"USER\",\"3gprs\"");
    delay(2000);
    Serial.println("AT+SAPBR=3,1,\"PWD\",\"3gprs\"");
    delay(2000);
    Serial.println("AT+SAPBR=3,1,\"Contype\",\"GPRS\"");
    delay(2000);
    Serial.println("AT+SAPBR=1,1");
```

```
    delay(10000);
    Serial.println("AT+HTTPINIT");
    delay(2000);
    Serial.println("AT+HTTPPARA=\"CID\",1");
    delay(2000);
 }

void start_GPS(){
     //Configuracion en Inicializacion GPS
    Serial.print("AT");
    delay(1000);
    Serial.println("AT+CGPSIPR=9600");//   (set   the   baud
rate)
    delay(1000);
    Serial.println("AT+CGPSPWR=1"); //  (turn on GPS power
supply)
    delay(1000);
    Serial.println("AT+CGPSRST=1");   //(reset   GPS   in
autonomy mode)
    delay(1000);
    Serial.println("AT+CGPSSTATUS?"); //cek status GPS
    delay(10000); //delay para esperar señal del GPS
 }
```

Kode Sumber A.3 Fungsi untuk Menginisialisasi GPS dan GSM

```
void setup(void)
{

  pinMode(3,OUTPUT);//The default digital driver pins for
the GSM and GPS mode
  pinMode(4,OUTPUT);
  pinMode(5,OUTPUT);
  digitalWrite(5,HIGH);
  delay(1500);
  digitalWrite(5,LOW);

  Serial.begin(9600);

  digitalWrite(3,LOW);//enable GSM TX、RX
  digitalWrite(4,HIGH);//disable GPS TX、RX

    delay(3000);

    start_GSM();

    delay(5000);
```

```
    start_GPS();

    delay(5000);//GPS ready

  Serial.println("AT");
  delay(2000);
  Serial.println("AT+CGPSPWR=1");
  delay(1000);
  Serial.println("AT+CGPSRST=1");
  delay(1000);

  digitalWrite(4,LOW);//Enable GPS mode
  digitalWrite(3,HIGH);//Disable GSM mode
  delay(2000);
  bno.setExtCrystalUse(true);
  delay(1000);

  while (!Serial) {
    ; // wait for serial port to connect. Needed for native
USB port only
  }
  if(!bno.begin())  }
}
```

Kode Sumber A.4 Fungsi Setup untuk Mempersiapkan Sensor dan GPS/GPRS/GSM *Shield* V3.0

```
void loop(void)
{
 double sumquat;
  double tetha;
  double q0,q1,q2,q3;
  double xg,yg,zg;
  double xga;
  float radtetha;
  // Possible vector values can be:
  // - VECTOR_ACCELEROMETER - m/s^2
  // - VECTOR_MAGNETOMETER  - uT
  // - VECTOR_GYROSCOPE     - rad/s
  // - VECTOR_EULER         - degrees
  // - VECTOR_LINEARACCEL   - m/s^2
  // - VECTOR_GRAVITY       - m/s^2
  imu::Vector<3>                  acc                    =
bno.getVector(Adafruit_BNO055::VECTOR_ACCELEROMETER);
  xg=acc.x();
  yg=acc.y();
  zg=acc.z();
```

```
 //Low Pass Filter
fXg = (xg * alpha + (fXg * (1.0 - alpha)));
fYg = (yg * alpha + (fYg * (1.0 - alpha)));
fZg = (zg * alpha + (fZg * (1.0 - alpha)));
//a=sqrt(fXg*fXg+fYg*fYg+fZg*fZg);
a=pow(pow(fXg,2)+pow(fYg,2)+pow(fZg,2),0.5);

if(i==1)
{
  x1=0;
  x2=fXg;
  xtemp=fXg;
  y1=0;
  y2=fYg;
  ytemp=fYg;
  z1=0;
  z2=fZg;
  ztemp=fZg;

}
else
{
  x1=xtemp;
  xtemp=fXg;
  x2=fXg;
  dx=abs(x2-x1);
  y1=ytemp;
  ytemp=fYg;
  y2=fYg;
  dy=abs(y2-y1);
  z1=ztemp;
  ztemp=fZg;
  z2=fZg;
  dz=abs(z2-z1);
}

Quaternion data
imu::Quaternion quat = bno.getQuat();

q0=quat.w();
q1=quat.y();
q2=quat.x();
q3=quat.z();

sumquat=sqrt((q1*q1)+(q2*q2)+(q3*q3));
tetha= (2*atan2(sumquat,q0))*180;
radtetha= tetha/3.14;
```

```
if (a  > 9.932)
   if( dx > 1.591)
      if (dx > 4.89)
         if (dz > 15.059) {
            if ( dz > 17.925)
                if ( radtetha <45)
                   {
                    FALL = true;
                    }
            else {
                  if (dx >6,786)
                     if ( radtetha <45)
                         {
                           FALL = true;
                          }
                 }
               }
             }
           }
         }
       }

   if (FALL==true){  //in event of a fall detection
    Serial.println("FALL DETECTED");
    FALL=false;
    exit(1);
  }
  i=0;
  delay(200);
}
```

Kode Sumber A.5 Fungsi untuk Pendeteksiaan Jatuh

```
void loop()
{
  Serial.println("AT"); //Send AT command
  delay(2000);
  Serial.println("AT");
  delay(2000);
  //Send message
  Serial.println("AT+CMGF=1");
  delay(1000);
  Serial.println("AT+CMGS=\"082377778325\"");
  delay(1000);
  Serial.print("Fall alarm from the ederly!
```

```
Latitude:
getLat()

Longitude:
 getLon()

Link:                    https://www.google.com/maps/place/-
getLat(),getLon()/@getLat(),getlon(),20z);
  delay(1000);
  Serial.write(26);
  while(1);
}
```

Kode Sumber A.6 Fungsi Untuk Mengirimkan SMS ke Kerabat

(Halaman ini sengaja dikosongkan)

dx	dy	dz	alpha	q0	q1	q2	q3	radtetha	KONDISI
0.0109	0.0266	0.0002	9.8351	0.7018	0.1184	0.7025	-0.0027	90.9039	BERDIRI
0.0263	0.0464	0.0093	9.8994	0.7017	0.1174	0.7027	-0.0023	90.9143	BERDIRI
0.0218	0.0332	0.0096	9.8631	0.7014	0.1182	0.7029	-0.0026	90.9743	BERDIRI
0.0238	0.0098	0.0194	9.7959	0.7014	0.1158	0.7032	-0.0033	90.9623	BERDIRI
0.0075	0.0204	0.0213	9.8094	0.7026	0.1163	0.702	-0.0032	90.7778	BERDIRI
0.0051	0.0092	0.0225	9.7907	0.7031	0.1167	0.7014	-0.0037	90.6895	BERDIRI
0.0095	0.0133	0.0399	9.8241	0.7012	0.1179	0.7031	-0.0022	91.0004	BERDIRI
0.0127	0.0927	0.0485	9.9406	0.7029	0.1173	0.7015	-0.0021	90.7267	BERDIRI
0.0069	0.0201	0.0697	9.8157	0.7	0.116	0.7047	-0.0034	91.2014	BERDIRI
0.0259	0.02	0.07	9.8479	0.7016	0.1057	0.7047	-0.0062	90.9338	BERDIRI
0.0134	0.0399	0.0798	9.7759	0.7007	0.1155	0.7041	-0.004	91.0862	BERDIRI
0.0617	0.07	0.0901	9.8331	0.7012	0.1089	0.7045	-0.0056	90.9935	BERDIRI
0.0002	0.0166	0.1	9.8097	0.7001	0.1163	0.7045	-0.0038	91.1705	BERDIRI
0.0099	0.0083	0.115	9.8009	0.7031	0.1164	0.7015	-0.0037	90.6904	BERDIRI
0.3537	0.9356	0.6219	10.9734	0.7338	0.0862	0.6723	-0.0463	85.6309	DUDUK
1.185	2.6416	0.9224	14.2022	0.6675	-0.0547	0.7238	-0.1658	96.2936	DUDUK
0.5822	1.1557	1.0357	9.1597	0.6415	0.076	0.7626	0.0322	100.243	DUDUK
1.1984	2.6278	2.1634	12.1414	0.5155	-0.0294	0.8562	-0.0173	117.9973	DUDUK
1.4174	0.8487	2.5363	11.9391	0.7213	0.0655	0.6877	-0.0493	87.7184	DUDUK
1.1364	3.4961	2.753	10.035	0.542	0.0184	0.8401	0.0008	114.4173	DUDUK

Tabel B.0.1 Data *Training* Bagian 1

dx	dy	dz	alpha	q0	q1	q2	q3	radtetha	KONDISI
0.1422	2.8642	2.8422	11.7909	0.7693	0.0894	0.6292	-0.0651	79.4462	DUDUK
0.7152	2.3275	3.0026	10.8424	0.6453	0.0331	0.7629	-0.0229	99.6829	DUDUK
0.6469	2.3057	3.5033	10.2282	0.4714	-0.0076	0.8815	0.0236	123.8066	DUDUK
1.0992	2.8581	5.4541	11.3385	0.5955	0.0089	0.8032	-0.007	106.9522	DUDUK
5.6598	2.5427	0.1531	7.4167	0.9745	0.0213	0.2066	-0.0848	25.9415	JATUH
17.1488	5.8465	0.5045	18.5382	0.9247	0.0138	0.3678	-0.0973	44.777	JATUH
4.9639	1.0457	0.9049	9.3576	0.9901	0.0217	0.0928	-0.1036	16.1919	JATUH
6.5329	9.4789	1.0709	7.5314	0.9867	0.0247	0.1262	-0.0996	18.7313	JATUH
2.8714	5.2942	1.6862	6.44	0.986	-0.0399	0.1616	-0.0009	19.1742	JATUH
9.2575	1.6598	3.336	11.0427	0.8246	-0.0782	0.5002	-0.2525	68.9421	JATUH
10.3114	7.8473	6.725	19.3476	0.875	0.0758	0.4733	-0.0674	57.9318	JATUH
4.0758	7.1972	6.8068	16.6323	0.9647	-0.1606	0.162	-0.1317	30.5592	JATUH
11.4341	3.2903	7.0327	17.0694	0.9857	0.0109	0.152	-0.072	19.416	JATUH
6.8846	2.5671	7.7025	15.1181	0.9899	-0.0043	0.1155	-0.0814	16.2631	JATUH
7.8812	2.0035	8.235	13.7411	0.919	0.021	0.3828	-0.0917	46.4558	JATUH
8.0772	0.2928	8.9513	12.7552	0.9803	-0.0909	0.1436	-0.1002	22.7698	JATUH
8.7296	7.1356	14.8325	18.8816	0.9941	-0.0087	0.0775	-0.0757	12.4851	JATUH
10.9708	2.0659	18.295	24.9874	0.9875	0.0038	0.1367	-0.0786	18.1594	JATUH
17.4023	0.4675	18.5845	26.9196	0.903	0.0153	0.2657	-0.3373	50.9166	JATUH

Tabel B.0.2 Data *Training* Bagian 2

LAMPIRAN C
PENJELASAN ORANGE3 SOFTWARE

Orange 3 adalah software machine learning dan data mining berbasis python memiliki pemrograman visual front-end untuk analisis data eksploratif dan visualisasi. Dengan usia 18 tahun telah melalui banyak pengembangan, salah satu yang utama ialah transisi menggunakan python 3, dan meninggalkan C++, sebagai gantinya banyak menggunakan library python seperti NumPy, SciPy, dan Scikit-learn. Pada versi terbaru yaitu Orange 3 banyak mengalami peningkatan secara Visualisasi dan banyak fitur tambahan. Orange 3 dikelola dan dikembangkan oleh Bioinformatics Laboratory, Faculty of Computer dan Information Science di University Ljubljana.

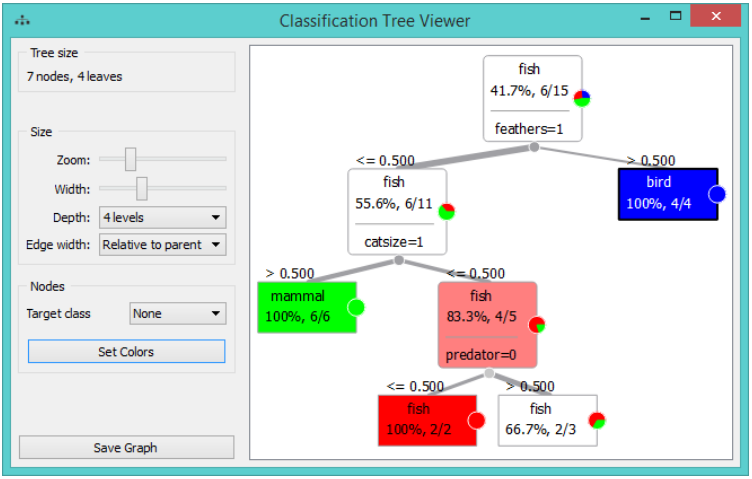
Fitur yang terdapat pada Orange 3 meliputi:

- **Canvas**: *graphical front-end* untuk analisis data
- **Widgets**:
  - **Data**: *widgets* untuk data input, data *filtering, sampling, imputation, feature manipulation* dan *feature selection.*
  - **Visualize**: *widgets for common visualization* (box plot, histograms, scatter plot) dan multivariate visualization (mosaic display, sieve diagram).
  - **Classify**: *supervised machine learning algorithms* untuk klasifikasi, contoh ditunjukkan pada Gambar 9
  - **Regression**: *supervised machine learning algorithms* for *regression*
  - **Evaluate**: *cross-validation, sampling-based procedures, reliability estimation* dan *scoring of prediction methods*
  - **Unsupervised**: *unsupervised learning algorithms* untuk *clustering (k-means, hierarchical clustering)* dan *data projection techniques (multidimensional scaling, principal component analysis, correspondence analysis).*

dx	dy	dz	alpha	q0	q1	q2	q3	radtetha	KONDISI
20.5547	4.3158	18.6106	29.3148	0.9557	-0.0896	0.2404	-0.1439	34.2331	JATUH
7.7415	0.3039	18.7442	22.7206	0.9255	0.0292	0.2349	-0.2957	44.5375	JATUH
20.3819	6.8794	18.7945	29.8374	0.9881	0.0087	0.1213	-0.0944	17.7205	JATUH
12.5085	8.0856	19.0863	27.0397	0.9605	-0.0856	0.2628	-0.0328	32.3369	JATUH
19.2173	14.4753	19.2899	33.2638	0.7943	0.0475	0.6036	-0.0507	74.865	JATUH
0.9053	0.3754	0.4381	15.1142	0.6808	0.0196	0.7322	0.0059	94.231	MELOMPAT
1.6068	12.7544	0.8494	23.0101	0.5839	-0.0506	0.8055	-0.0881	108.6107	MELOMPAT
1.1704	4.2219	1.0938	15.6333	0.6614	0.0392	0.7486	-0.0236	97.2296	MELOMPAT
0.4895	7.8908	2.2063	14.7016	0.5907	0.0173	0.8067	0.0025	107.6413	MELOMPAT
0.1464	15.1204	2.2834	23.6246	0.6332	-0.0757	0.7668	-0.0724	101.4734	MELOMPAT
0.1615	2.856	3.3843	18.8374	0.5893	0.0107	0.8077	-0.0142	107.8418	MELOMPAT
3.803	8.8955	3.4875	11.2145	0.7205	0.0752	0.6877	-0.0482	87.8617	MELOMPAT
1.504	9.379	3.785	20.5779	0.6264	0.0214	0.7789	0.0239	102.4835	MELOMPAT
1.7278	9.8772	4.8424	20.7024	0.5565	0.0012	0.8307	-0.0133	112.4276	MELOMPAT
0.1571	4.3921	5.0685	14.8082	0.4807	-0.0106	0.8768	-0.0114	122.5998	MELOMPAT
3.1461	8.7214	6.1788	10.6909	0.7223	0.0001	0.6824	-0.1124	87.5594	MELOMPAT

Tabel B.0.3 Data *Training* Bagian 3

(Halaman ini sengaja dikosongkan)



Gambar 9 Klasifikasi Tree Widgets pada Orange3

BIODATA PENULIS

Penulis lahir di Lubuklinggau, 5 Februari 1995, merupakan anak bungsu dari 2 bersaudara. Dalam perjalanan hidupnya penulis pernah menempuh pendidikan di SDN Kendal No. 53, SMP Xaverius Lubuklinggau, SMA Xaverius Lubuklinggau, dan S1 Jurusan Teknik Informatika Institut Teknologi Sepuluh Nopember (ITS) pada rumpun Komputasi Berbasis Jaringan (KBJ).

Selama menempuh kuliah, penulis aktif di Himpunan Mahasiswa Teknik-Computer Informatika (HMTC) ITS sebagai staf Hubungan Luar. Selain itu penulis juga aktif di Badan Eksekutif Mahasiswa tingkat Fakultas (BEM FTIF) sebagai Manajer External Affairs 2014/2015, serta aktif sebagai staf YESS SUMMIT 2014 dan kerja praktik di PT Telkom tbk Yogyakarta. Penulis dapat dihubungi melalui *e-mail* di satrioramadhana@gmail.com atau media sosial seperti *facebook* dengan nama *account* M. SatrioRamadhana Y. atau *linkedln* di @msatriory