DETEKSI SIRENE AMBULANS MENGGUNAKAN METODE *FAST FOURIER TRANSFORM* DENGAN MEMPERTIMBANGKAN EFEK *DOPPLER* PADA RASPBERRYPI

SKRIPSI

Untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun oleh:
Hifi Sulton Auliya
NIM: 115060900111039

PROGRAM STUDI TEKNIK INFORMATIKA
PROGRAM TEKNOLOGI INFORMASI DAN ILMU KOMPUTER
UNIVERSITAS BRAWIJAYA
MALANG
2016

PENGESAHAN

DETEKSI SIRENE AMBULANS MENGGUNAKAN METODE *FAST FOURIER TRANSFORM* DENGAN MEMPERTIMBANGKAN EFEK *DOPPLER* PADA RASPBERRYPI

SKRIPSI

Diajukan untuk memenuhi sebagian persyaratan
memperoleh gelar Sarjana Komputer

Disusun Oleh :
Hifi Sulton Auliya
NIM: 115060900111039

Skripsi ini telah diuji dan dinyatakan lulus pada
7 Januari 2016
Telah diperiksa dan disetujui oleh:

Dosen Pembimbing I	Dosen Pembimbing II
Gembong Edhi Setyawan, S. T., M. T.	Ir. Heru Nurwasito, M. Kom.
NIP: 201208 761201 1 001	NIP: 19650402 199002 1 001

Mengetahui
Ketua Program Studi Teknik Informatika

Drs. Mariji, M. T.
NIP: 19670801 199203 1 001

PERNYATAAN ORISINALITAS

Saya menyatakan dengan sebenar-benarnya bahwa sepanjang pengetahuan saya, di dalam naskah skripsi ini tidak terdapat karya ilmiah yang pernah diajukan oleh orang lain untuk memperoleh gelar akademik di suatu perguruan tinggi, dan tidak terdapat karya atau pendapat yang pernah ditulis atau diterbitkan oleh orang lain, kecuali yang secara tertulis disitasi dalam naskah ini dan disebutkan dalam daftar pustaka.

Apabila ternyata didalam naskah skripsi ini dapat dibuktikan terdapat unsur-unsur plagiasi, saya bersedia skripsi ini digugurkan dan gelar akademik yang telah saya peroleh (sarjana) dibatalkan, serta diproses sesuai dengan peraturan perundang-undangan yang berlaku (UU No. 20 Tahun 2003, Pasal 25 ayat 2 dan Pasal 70).

Malang, 7 Januari 2016

Hifi Sulton Auliya

NIM: 115060900111039

KATA PENGANTAR

Puji dan syukur penulis panjatkan kepada Allah SWT, karena dengan rahmat, taufik dan hidayah-Nya lah skripsi ini dapat diselesaikan. Skripsi berjudul "Deteksi Sirene Ambulans Menggunakan Metode *Fast Fourier Transform* Dengan Mempertimbangkan Efek *Doppler* Pada RaspberryPI" ini disusun untuk memenuhi sebagian persyaratan memperoleh gelar Sarjana Komputer.

Penulis menyadari bahwa penyusunan skripsi ini tidak terlepas dari bantuan berbagai pihak. Ucapan terima kasih penulis sampaikan kepada:

1. Ayah dan Ibu atas segenap do'a, dukungan dan kasih sayang yang telah diberikan kepada penulis.
2. Gembong Edhi Setyawan, ST. T., M. T. selaku dosen pembimbing I yang telah membimbing penulis dalam pengerjaan dan penulisan skripsi ini.
3. Ir. Heru Nurwasito, M.Kom. selaku dosen pembimbing II yang telah membimbing penulis dalam pengerjaan dan penulisan skripsi ini.
4. Ir. Sutrisno, M.T. selaku Dekan Fakultas Ilmu Komputer Universitas Brawijaya.
5. Drs. Mariji, M.T. selaku ketua Program Studi Informatika/Ilmu Komputer Fakultas Ilmu Komputer Universitas Brawijaya.
6. Adharul Muttaqin ST., MT selaku Ketua Program Studi Sistem Komputer Fakultas Ilmu Komputer Universitas Brawijaya.
7. Barlian Henryranu, S.T., M.T. selaku Kepala Laboratorium Sistem Komputer dan Robotika Program Teknologi Informasi dan Ilmu Komputer.
8. Mas Didit dan Segenap dosen Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya atas segenap ilmu pengetahuan dan perhatian yang diberikan Segenap staff dan pegawai Program Teknologi Informasi dan Ilmu Komputer Universitas Brawijaya atas segala bantuan yang bersifat administratif.

Pada akhirnya, penulis menyadari bahwa skripsi ini masih belum sempurna. Oleh karena itu, penulis sangat mengharapkan kritik dan saran yang membangun. Penulis berharap semoga skripsi ini dapat bermanfaat bagi pengembangan ilmu pengetahuan dan teknologi.

Malang, 7 Januari 2016

Hifi Sulton Auliya

hifisultonauliya@gmail.com

ABSTRAK

Kematian banyak terjadi karena terlambatnya penanganan yang didapat oleh seorang pasien. Ambulans memiliki peranan penting agar pasien mendapatkan pertolongan lebih lanjut. Untuk itu, Ambulans merupakan salah satu pengguna jalan yang memperoleh hak utama untuk didahulukan. Karena, salah satu faktor penyebab terlambatnya ambulans adalah kemacetan lalu lintas, maka untuk menghindari hal tersebut perlu suatu alat untuk mendeteksi sirine ambulans menggunakan metode *Fast Fourier Transform* dengan mempertimbangkan efek *Doppler* yang diimplementasikan pada Lampu Lalu Lintas. Metode *Fast Fourier Transform* ini berguna sebagai pendeteksi frekuensi dari setiap bunyi. Efek Doppler digunakan untuk menentukan indikator frekuensi pada pendeteksian ambulans. Frekuensi sirene ambulans ini, dibedakan berdasarkan macam-macam variasi bunyi sirene ambulans yaitu hilo, wail, phaser, dan yelp.

Dari hasil pengujian ambulans diam didapat total rata-rata akurasi sistem adalah sebesar 100% dengan waktu pendeteksian rata-rata adalah 6.958 detik. Dan Dari hasil pengujian ambulans bergerak dengan mempertimbangkan efek doppler didapat total rata-rata akurasi sistem adalah 100% dengan waktu pendeteksian rata-rata adalah 9.05 detik. Sedangkan Dari hasil pengujian ambulans bergerak tanpa mempertimbangkan efek doppler didapat total rata-rata akurasi sistem adalah 95% dengan waktu pendeteksian rata-rata adalah 10.95 detik. Oleh karena itu pendeteksian sirene ambulans menggunakan metode *Fast Fourier* perlu mempertimbangkan efek doppler untuk meningkatkan kecepatan waktu pendeteksian sebesar 1.90 detik dan meningkatkan rata-rata akurasi sistem sebesar 5%. Metode ini juga merupakan metode yang tepat untuk memberikan solusi alternatif kepada ambulans untuk dapat mengurangi kemacetan pada lampu lalu lintas.

Kata kunci: Sirene Ambulans. *Fast Fourier Transform.* Efek *Doppler.* Raspberry PI. FFT

ABSTRACT

Many deaths occur due to delay in treatment obtained by a patient. The ambulance has an important role to enable patients to get further help. To that end, Ambulance is one of the main roads which acquired the rights to take precedence. Because, one of the factors causing delays in ambulances is a traffic jam, then to avoid that we need of a tool to detect the ambulance siren using Fast Fourier Transform method consider the Doppler effect which is implemented in Traffic Lights. Fast Fourier Transform method is useful to detect frequency of each sound. The Doppler effect is used to determine the frequency detection indicator ambulance. The ambulance siren frequency, distinguished by a variety of variations of the sound of an ambulance siren is hilo, wail, phaser, and a yelp.

From the test results obtained silent ambulance average total accuracy of the system is at 100% with an average detection time is 6.958 seconds. And the results of testing the ambulance moves by considering the Doppler effect obtained average total system accuracy is 100% with an average detection time is 9.05 seconds. While the results of testing an ambulance move without considering the Doppler effect obtained average total system accuracy is 95% with an average detection time was 10.95 seconds. Therefore, the detection of an ambulance siren using Fast Fourier need to consider the Doppler effect to increase the speed detection time of 1.90 seconds and an average increase of 5% system accuracy. This method is also an appropriate method to provide alternative solutions to the ambulance in order to reduce congestion at the traffic lights.

Keyword: Sirene Ambulans. *Fast Fourier Transform.* Efek *Doppler.* Raspberry PI. FFT

DAFTAR ISI

DAFTAR TABEL

DAFTAR GAMBAR

DAFTAR LAMPIRAN

BAB 1 PENDAHULUAN

1.1 Latar belakang

Indonesia menempati peringkat ke-5 di Dunia sebagai negara dengan tingkat kecelakaan lalu lintas tertinggi (Anwar, 2014). Penanganan korban cedera parah di tempat kecelakaan sangat penting pada saat terjadi kecelakaan, tetapi yang lebih penting adalah membawa korban secepatnya ke rumah sakit dikarenakan jika korban terlambat dibawa kerumah sakit oleh ambulans perawatan intensif di rumah sakit pun tidak ada gunanya lagi. Hal ini membuat ambulans adalah salah satu pengguna jalan yang memperoleh hak utama untuk didahulukan (UU, 2009).

Ambulans idealnya tiba maksimal 30 menit setelah ada panggilan keadaan darurat (kompas, 2008). Ambulans bisa terlambat dikarenakan oleh banyak faktor salah satunya adalah kemacetan pada Lampu Lalu Lintas (Ary, 2013). Oleh karena itu Lampu Lalu Lintas harus bisa mendeteksi keberadaan ambulans sehingga dapat memprioritaskan kendaraan khusus tersebut.

Salah satu cara untuk mengetahui keberadaan ambulans, adalah dengan menganalisa sirene ambulans. Ambulans menggunakan sirene dengan amplitudo tinggi untuk memperingatkan pengguna jalan (Mizaki, et al., 2013). Sirene ambulans memiliki karakteristik yaitu bunyi tinggi (*high frequency*) dan bunyi rendah (*low frequency*). Di Indonesia ada 4 jenis bunyi sirine ambulans yaitu *hilo*, *phaser*, *wail*, dan *yelp* (Lakesma, 2012). Bunyi tersebut masing- masing memiliki perbedaan pada interval waktu antara bunyi tinggi dan bunyi rendah. Untuk dapat menganalisa sirene ambulans, informasi yang berupa sinyal analog dalam domain waktu di konversikan ke dalam domain frekuensi dengan meggunakan algoritma *Fast Fourier Transform* (FFT).

Laju ambulans mengakibatkan frekuensi sirene ambulans menjadi bergeser hal ini dikenal dengan Efek *Doppler*. Efek *Doppler* adalah pergeseran frekuensi dari sebuah sumber yang disebabkan oleh gerak relative dari sumber dan pengamat (Young & Freedman, 2012). Efek *Doppler* mempengaruhi hasil frekuensi yang akan diproses, sehingga pendeteksian ambulans menggunakan metode *Fast Fourier Transform* harus mempertimbangkan efek *Doppler*.

Pada penelitian sebelumnya oleh Mizaki T (2013), pendeteksian ambulans di jepang dengan menggunakan Metode FFT telah diimplementasikan pada mobil yang di khususkan bagi pengguna jalan yang memiliki gangguan pendengaran ataupun pengaruh *noise* yang disebabkan oleh bahan mobil yang menggunakan *soundproof*. Pada penelitian ini menggunakan layar yang disambungkan pada mikrokontroller dsPIC untuk memperingatkan mobil apabila ada ambulans. Sedangkan pada penelitian Liaw (2013), pendeteksian ambulans di Taiwan juga menggunakan metode *Fast Fourier Transform* (FFT) yang nantinya hasil dari metode tersebut ditambah dengan *Longest Common Subsequence* (LCS) untuk membandingkan hasil FFT dari sinyal ambuan dengan sample suara ambulans. Pada penelitian ini juga diimplementasikan pada kendaraan lain dijalan agar mengetahui keberadaan ambulans.

Pada penelitian Mizaki T (2013), memiliki kelebihan pada biaya pembuatan yang murah dan pada penelitian Liaw (2013) memiliki kelebihan hasil keakuratan yang tinggi mencapai 85% dan merupakan pembaruan dari penelitian milik Takuya Mizaki (2013). Pada implementasi kedua penelitian ini dirasa kurang efisien dikarenakan pengimplementasian pada kedua penelitian tersebut adalah dengan menambahkan alat pendeteksi ambulans pada setiap kendaraan lain di jalan sehingga bagi kendaraan yang tidak memodifikasi kendaraan, tak dapat menjalankan fungsi untuk mendeteksi ambulans. Bunyi ambulans yg dapat di kenali pada kedua penelitian ini hanya terbatas pada 1 bunyi sirene saja, sehingga pada penerapannya di Indonesia kurang akurat dikarenakan di Indonesia ambulans memiliki 4 varian bunyi sirene (Lakesma, 2012). Pada kedua penelitian sebelumnya juga diterapkan pada saat ambulans melaju dijalan dengan keadaan tidak macet, sehingga masalah dari kemacetan ambulans yang diakibatkan oleh Lampu Lalu Lintas belum terpecahkan.

Oleh karena itu pada penelitian ini akan diterapkan deteksi sirene ambulans menggunakan metode *Fast Fourier Transform* dengan mempertimbangkan efek *Doppler* yang diimplementasikan pada Lampu Lalu Lintas dengan menggunakan RaspberryPI. Sehingga Lampu Lalu Lintas dapat memprioritaskan ambulans dan tidak perlu memodifikasi seluruh kendaraan.

1.2 Rumusan masalah

Dari permasalahan yang telah dibahas pada latar belakang, maka dapat dirumuskan beberapa rumusan masalah yang nantinya akan dibahas secara keseluruhan, diantaranya adalah:

1. Bagaimana waktu deteksi ambulans menggunakan metode *Fast Fourier Transform* pada otomatisasi *traffic light*?
2. Bagaimana keakuratan deteksi ambulans menggunakan metode *Fast Fourier Transform* pada otomatisasi *traffic light*?
3. Bagaimana pengaruh efek *Doppler* pada pendeteksian ambulans dengan metode *Fast Fourier Transform*?

1.3 Tujuan

Sesuai dengan latar belakang dan rumusan masalah, tujuan dari pengembangan sistem ini adalah membuat suatu model sistem pendeteksi keberadaan ambulans dengan metode *Fast Fourier Transform* dengan mempertimbangkan efek *Doppler* untuk di implementasikan pada RaspberryPI.

1.4 Manfaat

Manfaat yang ingin dicapai dari penulisan laporan ini adalah:

1. Memberikan solusi alternatif kepada ambulans untuk dapat mengurangi kemacetan pada traffic light.
2. Memberikan kontribusi kepada masyarakat dalam upaya membantu meminimalisir terjadinya kematian akibat telatnya penanganan pasien dikarenakan telatnya ambulans menuju rumah sakit.

1.5 Batasan masalah

Agar permasalahan yang dirumuskan dapat lebih terfokus, maka penelitian tugas akhir ini dibatasi dalam hal:

1. Alat yang digunakan adalah Raspberry PI.
2. Mikrofon yang digunakan adalah mikrofon karbon.
3. Sample audio yang digunakan berdasarkan Lakesma UB.
4. Sample ambulans di dapat dari perekaman langsung maupun internet.
5. Uji coba penelitian dengan cara merekam sampling bunyi dan menggunakan *sound* laptop maupun *sound* external untuk pengujian.
6. Penelitian ini difokuskan pada pendeteksian sirene ambulans pada lampu lalu lintas yaitu pada saat ambulans berhenti, atau pada saat ambulans akan berhenti dikarenakan lampu lalulintas.

1.6 Sistematika pembahasan

Menggunakan Metode *Fast Fourier Transform* Dengan Mempertimbangkan Efek *Doppler* Pada RaspberryPI direncanakan sebagai berikut:

BAB I : Pendahuluan

Memuat latar belakang permasalahan, rumusan masalah, tujuan, manfaat, batasan masalah, dan sistematika penulisan.

BAB II : Landasan Kepustakaan

Memuat landasan kepustakaan mengenai penelitian-penelitian sebelumnya dan menjelaskan dasar teori yang terkait dengan penyusunan penyusunan Deteksi Sirene Ambulans Menggunakan Metode *Fast Fourier Transform* Dengan Mempertimbangkan Efek *Doppler* Pada RaspberryPI.

BAB III : Metodologi

Membahas metode yang digunakan dalam penelitian yang terdiri dari studi literatur, analisis kebutuhan sistem, perancangan, implementasi, pengujian dan

analisis penyusunan Deteksi Sirene Ambulans Menggunakan Metode *Fast Fourier Transform* Dengan Mempertimbangkan Efek *Doppler* Pada RaspberryPI.

BAB IV : Rekayasa Kebutuhan

Membahas tentang rekayasa kebutuhan dari penyusunan Deteksi Sirene Ambulans Menggunakan Metode *Fast Fourier Transform* Dengan Mempertimbangkan Efek *Doppler* Pada Raspberry-Pi.

BAB V : Perancangan dan Implementasi

Membahas tentang perancangan dan implementasi dari penyusunan Deteksi Sirene Ambulans Menggunakan Metode *Fast Fourier Transform* Dengan Mempertimbangkan Efek *Doppler* Pada Raspberry-Pi.

BAB VI : Pengujian dan Analisis

Memuat proses dan hasil pengujian terhadap metode yang telah direalisasikan.

BAB VII : Penutup

Memuat kesimpulan serta saran yang diperoleh dari pembuatan dan pengujian metode untuk pengembangan lebih lanjut.

BAB 2 LANDASAN KEPUSTAKAAN

Bab ini berisi tentang kajian pustaka dan dasar teori yang berhubungan dengan karakteristik sirene ambulans, metode *Fast Fourier Transform,* efek Doppler, dan dasar teori lainnya yang terkait dengan pendeteksian sirene ambulans menggunakan metode *Fast Fourier Transform* dengan mempertimbangkan efek *Doppler* pada RaspberryPI.

2.1 Kajian Pustaka

Penelitian sebelumnya mengenai pendeteksian sirene ambulans dengan menggunakan metode *Fast Fourier Transform* (FFT) telah diusulkan oleh Mizaki T (2013) dengan judul "*Ambulance Siren Detector using FFT on dsPIC*". Pada penelitian ini mendeteksi ambulans dengan menggunakan dsPIC sejenis mikrokontroler yang sudah dilengkapi dengan *Analog Digital Converter* untuk mengkonversikan sinyal audio analog dari mikrofon menjadi sinyal digital yang akan diproses. Hasil dari penelitian ini diterapkan untuk mobil dengan *soundproof* yang baik, sehingga suara dari luar kendaraan menjadi samar yang menyebabkan ambulans tak segera mendapatkan prioritasnya dijalan. Pada penelitian ini menggunakan 1 macam bunyi sirene ambulans yaitu *hi-lo*.

Penelitian yang sama juga diterapkan oleh Liaw (2013) dengan judul "*Recognition of the Ambulance Siren in Taiwan by the Longest Common Subsequence*". Pada penelitian ini sirene ambulans di Taiwan tak hanya di deteksi dengan menggunakan metode FFT saja, tetapi juga dengan mencocokkan pola bunyi antara bunyi sirene ambulans asli dengan bunyi hasil rekaman di lapangan. Pola bunyi tersebut dicocokkan dengan menggunakan algoritma *Longest Common Subsequence* (LCS). Dari LCS ini didapat nilai terpanjang (*Longest*) dari beberapa *Common Subsequence* dari kedua pola yang identik. Hasil dari penelitian ini juga diterapkan untuk mobil dengan *soundproof* yang baik. Pada penelitian ini juga menggunakan 1 macam bunyi sirene ambulans yaitu *hi-lo.*

Dari penelitian di atas, penulis tertarik untuk mengatasi kekurangan yang terdapat pada penelitian tersebut yaitu dengan menambah jenis bunyi sirene ambulans, dikarenakan di Indonesia ada 4 macam bunyi dasar sirene ambulans yaitu *hi-lo, phaser, wail,* dan *yelp* (Lakesma, 2012). Penulis juga tertarik untuk memodifikasi penerapan penelitian sebelumnya yaitu pada kendaraan yang ingin mendeteksi ambulans menjadi pada lampu lalu lintas, sehingga ambulans tetap dapat tepat waktu untuk sampai pada tujuan. Penulis juga ingin mempertimbangkan efek *Doppler,* Sehingga penulis melakukan penelitian pendeteksian sirene ambulans menggunakan metode FFT dengan mempertimbangkan Efek *Doppler* yang akan di terapkan pada otomatisasi Lampu Lalu Lintas. Pada sistem ini diharap dapat meminimalisir kemacetan ambulans yang disebabkan oleh Lampu Lalu Lintas.

2.2 Dasar Teori

Pada dasar teori ini akan membahas tentang dasar-dasar penulisan dalam membuat system. Dasar teori yang digunakan penulis antara lain: Ambulans, Teori Sinyal, Efek Doppler, Raspberry Pi, arecord, GPIO, Python, Scipy FFT, Numpy, *Soundcard External*, Mikrofon Karbon, Led.

2.2.1 Ambulans

Menurut Kamus Besar Bahasa Indonesia (KBBI) ambulans adalah kendaraan (mobil dsb) yang dilengkapi peralatan medis untuk mengangkut orang sakit atau korban kecelakaan menuju rumah sakit untuk mendapatkan pertolongan lebih lanjut (KBBI, n.d.). Hal ini membuat ambulans adalah salah satu pengguna jalan yang memperoleh hak utama untuk didahulukan (UU, 2009). Umumnya kendaraan ambulans ini berwarna putih dengan memiliki lampu rotator dan suara sirene yang khas supaya dapat menembus kemacetan lalu lintas.

Lampu rotator memiliki 3 macam warna dengan fungsinya masing-masing sesuai dengan Undang-undang no. 22 tentang lalu lintas dan angkutan jalan pasal 59 ayat 5. Macam-macam lampu rotator antara lain:

1. Lampu isyarat warna biru dan sirene digunakan untuk mobil petugas Kepolisian Negara Republik Indonesia
2. Lampu isyarat warna merah dan sirene digunakan untuk mobil tahanan, pengawalan Tentara Nasional Indonesia, pemadam kebakaran, ambulans, palang merah, dan jenazah; dan
3. Lampu isyarat warna kuning tanpa sirene digunakan untuk mobil patroli jalan tol, pengawasan sarana dan Prasarana Lalu Lintas dan Angkutan Jalan, perawatan dan pembersihan fasilitas umum, menderek Kendaraan, dan angkutan barang khusus.

Namun berdasarkan PPRI (Peraturan Pemerintah Republik Indonesia) no. 44 tahun 1993 tentang kendaraan dan pengemudi, ambulans diperbolehkan memakai lampu rotator berwarna biru. Sehingga IKABI (Ikatan Ahli Bedah Indonesia) merekomendasikan kepada karoseri untuk memasang lampu totator merah dan biru di tengah atas kendaraan. Jadi sementara ambulans boleh menggunakan lampu rotari bulat dan *light bar* merah-biru atau biru-biru (Lakesma, 2012).

Sirene pada ambulans umumnya memiliki 5 jenis suara yang memiliki fungsi masing-masing. Jenis-jenis suara tersebut adalah:

1. *Wail* digunakan ketika kendaraan berjalan di jalur yang lurus.
2. *Yelp* digunakan ketika kendaraan berada dipersimpangan.
3. *Hi-lo* dan Phaser digunakan sebagai kombinasi untuk mendapatkan perhatian yang lebih efektif.

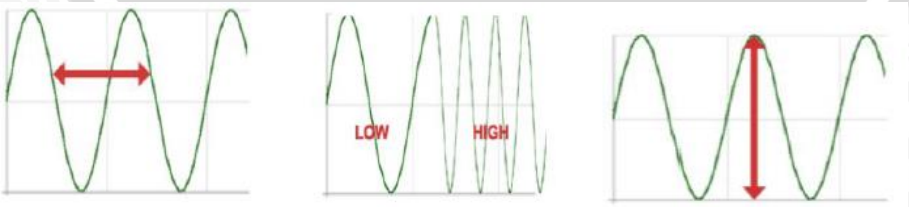
4. *Horn* digunakan seperti klakson untuk memberikan peringatan lebih juka suara-suara lainnya tidak mendapat perhatian pengguna jalan.

2.2.2 Teori Sinyal

Sinyal adalah suatu isyarat untuk meneruskan suatu kegiatan. Biasanya isyarat ini berbentuk tanda-tanda, lampu-lampu, suara-suara, dan lain-lain. Dalam kereta api isyarat berarti suatu tanda untuk melanjutkan atau meneruskan perjalanan ke stasiun berikutnya, dan biasanya isyarat ini dikirimkan oleh stasiun yang terkait. Dalam dunia Keteknikan, khususnya Teknik Informasi, Teknik Elektro, dan Teknik Kendali, isyarat adalah besaran yang berubah dalam waktu dan atau dalam ruang dan membawa suatu data/informasi. Menurut International Telecommunication Union (ITU), sinyal adalah suatu gejala fisika dimana satu atau lebih dari karakteristiknya melambangkan informasi (Sipasulta, et al., 2014). Sinyal ini biasanya berupa sinyal elektrik. Sinyal ini bisa merupakan besaran elektrik murni (tegangan, arus, dan sebagainya), tetapi pada umumnya adalah besaran fisik lain yang dijadikan elektrik dengan bantuan sensor misal mikrofon sebagai sensor suara. Contoh sinyal elektrik adalah sinyal suara yang berasal dari radio, sinyal citra yang berasal dari kamera fotografi, dan sinyal video yang berasal dari kamera video.

2.2.2.1 Sinyal Suara

Audio diartikan sebagai suara atau reproduksi suara. Gelombang suara adalah gelombang yang dihasilkan dari sebuah benda yang bergetar. Gambarannya adalah senar gitar yang dipetik, gitar akan bergetar dan getaran ini merambat di udara, atau air, atau material lainnya (Sipasulta, et al., 2014). Satu-satunya tempat dimana suara tak dapat merambat adalah ruangan hampa udara. Gelombang suara ini memiliki lembah dan bukit, satu buah lembah dan bukit akan menghasilkan satu siklus yang disebut periode. Siklus ini berlangsung berulang-ulang, yang membawa pada konsep frekuensi. Jelasnya, frekuensi adalah jumlah dari siklus yang terjadi dalam satu detik. Satuan dari frekuensi adalah Hertz atau disingkat Hz. Telinga manusia dapat mendengar bunyi dengan rentan antara 20 Hz hingga 20 KHz (20.000 Hz) sesuai dengan batasan sinyal suara. Karena pada dasarnya sinyal suara adalah sinyal yang dapat diterima atau didengar oleh telinga manusia. Angka 20 Hz sebagai frekuensi suara terendah yang dapat diterima atau didengar, sedangkan 20 KHz merupakan frekuensi tertinggi yang dapat diterima atau didengar.



Gambar 2. 1 Panjang gelombang, Frekuensi, dan Amplitudo

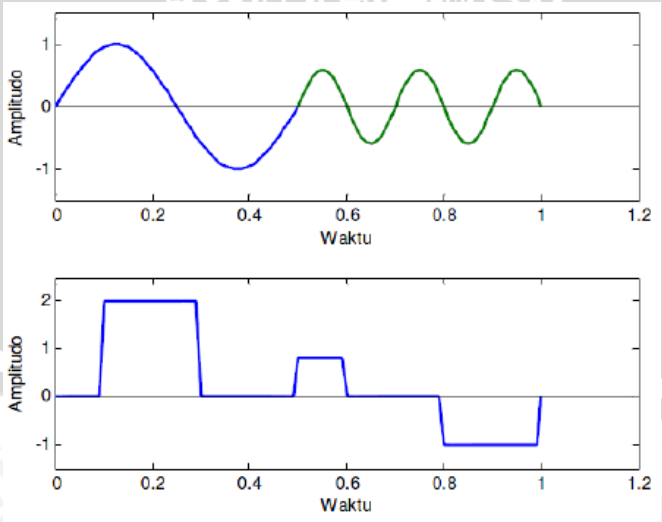
(Sipasulta, et al., 2014)

Panjang gelombang merupakan jarak antara titik gelombang dan titik ekuivalen pada fasa berikutnya, sedangkan amplitudo merupakan kekuatan atau daya gelombang sinyal (Sipasulta, et al., 2014). Gelombang yang lebih tinggi diinterpretasikan sebagai gelombang yang lebih tinggi, sehingga dinamakan amplifier untuk perangkat yang berfungsi untuk menambah amplitudo. Frekuensi merupakan jumlah getaran dalam waktu satu detik. Diukur dalam *Hertz* atau siklus per detik. Getaran gelombang suara semakin cepat, maka frekuensi semakin tinggi. Frekuensi lebih tinggi diinterpretasikan sebagai jalur yang lebih tinggi. Misalnya, bila menyanyi dalam pita suara tinggi, maka memaksa tali suara untuk bergetar cepat. Panjang gelombang, amplitudo, dan frekuensi digambarkan pada gambar 2.1.

2.2.2.2 Sinyal dan Sistem

Sinyal adalah fenomena dari lingkungan yang terukur atau terkuantisasi, sementara sistem merupakan bagian dari lingkungan yang menghubungkan sinyal dengan sinyal lainnya, atau dengan kata lain merespon sinyal masuk dengan menghasilkan sinyal keluaran. Ada berbagai macam contoh sinyal dan sistem, salah satunya adalah suara pembicaraan dan sistem komunikasi telepon. (Sipasulta, et al., 2014)

Berdasarkan bentuknya, data dan sinyal dapat dibedakan ke dalam data dan sinyal analog atau data dan sinyal digital (gambar 2). Suatu data atau sinyal dikatakan analog apabila amplitudo dari data atau sinyal tersebut terus menerus ada dalam rentang waktu tertentu (kontinyu) dan memiliki variasi nilai amplitudo tak terbatas. Misalnya, data yang berasal dari suara (*voice*) tergolong sebagai data analog. Sebaliknya data atau sinyal dikatakan digital apabila amplitudo dari data atau sinyal tersebut tidak kontinyu dan memiliki variasi nilai amplitudo yang terbatas (diskrit). (Sipasulta, et al., 2014)



Gambar 2. 2 Sinyal Analog (Atas) Sinyal Digital (Bawah)

(Sipasulta, et al., 2014)

Suara yang sudah tersimpan dapat diubah ke dalam berbagai format audio seperti mp3, wav, flac, real audio, midi, dan sebagainya, format penyimpanan suara yang dipakai dalam penelitian ini yaitu format wav karena Python bisa memproses format suara dalam bentuk wav.

Format audio muncul dari penemuan beberapa ilmuwan, berawal dari tahun 1887 Carles Cross memiliki ide membuat piringan yang bisa mengeluarkan suara, tetapi idenya tidak dapat dia wujudkan, baru oleh Thomas Alva Edison ide itu bisa terwujud.

Sekarang ini hampir seluruh produksi audio menggunakan teknologi komputer untuk pemrosesan ataupun penyimpanannya, hal ini juga yang menyebabkan timbulnya berbagai macam format audio digital. Beberapa formatnya adalah sebagai berikut:

1. Waveform (WAV) audio adalah format audio standar Microsoft dan IBM untuk PC. WAV adalah data tidak terkompres sehingga seluruh sampel audio disimpan semuanya di *harddisk*, *Software* yang dapat menciptakan WAV dari *Analog Sound* misalnya adalah *Windows Sound Recorder*.
2. *Audio CD* (.cda) adalah format untuk mendengarkan CD Audio.
3. *Audio Interchange File Format* (AIF) adalah format audio standar Macintosh.
4. *Mpeg Audio Layer 3* (Mp3) adalah format audio yang sering kita dengar dan biasanya digunakan diinternet karena ukurannya yang cukup kecil dibandingkan format yang lain.

Music Instrument Digital Interface (MIDI) adalah format standar yang dikembangkan oleh perusahaan alat-alat musik elektronik.

2.2.3 Efek Doppler

Efek Doppler adalah Fenomena perubahan frekuensi karena pengaruh gerak relatif antara sumber bunyi dan pendengar, pertama kali diamati oleh Christian Doppler. Jika diantara sumber bunyi dan pendengar tidak ada gerakan relatif, maka frekuensi sumber bunyi dan frekuensi bunyi yang didengar oleh seseorang adalah sama. Tetapi, jika antara sumber bunyi dan si pendengar terdapat gerak relatif, maka antara frekuensi sumber bunyi dan frekuensi bunyi yang didengar tidaklah sama.

Effek Doppler adalah peristiwa berubahnya harga frekuensi bunyi yang diterima oleh pendengar (P) dari frekuensi suatu sumber bunyi (S) apabila terjadi gerakan relatif antara P dan S.

$$f_p = \frac{v \pm v_p}{v \pm v_s} \times f_s \quad (2.1)$$

Keterangan:

- f_p adalah frekuensi yang didengar oleh pendengar.
- f_s adalah frekuensi yang dipancarkan oleh sumber bunyi.
- v_p adalah kecepatan pendengar.

- v_s adalah kecepatan sumber bunyi.
- v adalah kecepatan bunyi di udara sebesar 340m/s (Young & Freedman, 2012).
- Tanda + untuk v_p dipakai bila pendengar bergerak mendekati sumber bunyi.
- Tanda - untuk v_p dipakai bila pendengar bergerak menjauhi sumber bunyi.
- Tanda + untuk v_s dipakai bila sumber bunyi bergerak menjauhi pendengar.
- Tanda - untuk v_s dipakai bila sumber bunyi bergerak mendekati pendengar.

2.2.4 Raspberry PI

Raspberry Pi adalah salah satu Mini PC yang dikembangkan pertama kali oleh *Raspberry Pi Foundation* di UK. Raspberry Pi 2 Model B adalah generasi ke dua dari Raspberry Pi, Raspbery Pi 2 ini menggantikan Raspberry Pi Model B+ pada Februari 2015. Raspberry Pi 2 memliki A 900MHz quad-core ARM Cortex-A7 CPU dengan ram 1GB. Dengan ARMv7 prosessor, Raspberry Pi 2 dapat menjalankan *full ARM GNU/Linux Distribution*, Snappy Ubuntu Core, dan Windows 10. Selain itu untuk booting dan storage menggunakan Micro SD Card. Raspberry Pi 2 memiliki spesifikasi sebagai berikut:

1. A 900MHz quad-core ARM Cortex-A7 CPU
2. 1GB RAM
3. 4 USB ports
4. 40 GPIO pins
5. Full HDMI port
6. Ethernet port
7. Combined 3.5mm audio jack and composite video
8. Camera interface (CSI)
9. Display interface (DSI)
10. Micro SD card slot
11. VideoCore IV 3D graphics core

Gambar 2. 3 Raspberry Pi 2

Sistem Operasi pada Raspberry Pi 2 salah satunya adalah Raspbian. Raspbian adalah *free Operating System* berbasis Debian yang telah di optimisasi oleh Raspberry Pi. Bahasa Pemrograman yang dapat digunakan pada linux ini tentunya bermacam-macam, salah satunya adalah bahasa pemrograman Python.

2.2.4.1 Arecord

Arecord adalah aplikasi command line bawaan OS Raspberry yang digunakan untuk *sound record* pada ALSA *Sound-Card Driver.* Syntax program arecord antara lain adalah sbb:

"arecord –D plughw:0 –d 1 1.wav"

-D menandakan device name, yaitu memilih nama dari PCM (Pulse Code Modulator) yaitu salah satu teknik memproses suatu sinyal analog menjadi sinyal digital.

Plughw:0 sendiri menyatakan nama dari ALSA PCM. 0 dibelakan menandakan pada slot berapa mikrofon terdeteksi pada raspberry.

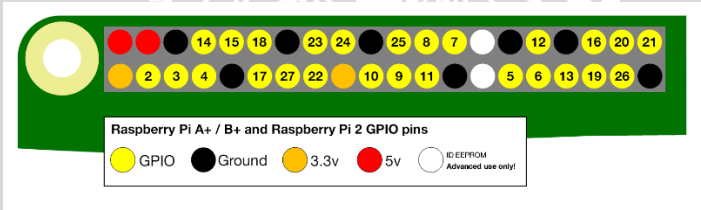
-d menandakan *duration* atau waktu lamanya perekaman dalam satuan detik.

1.wav menandakan nama dari outputan perekaman suara.

(Linux Organization, n.d.)

2.2.4.2 General Purpose Input/Output (GPIO)

GPIO adalah fitur dari raspberry PI yang dimana pada raspberry PI memiliki pin input output pada *board*-nya. Pin ini merupakan *physical interface* antara raspberry dengan berbaga macam sensor, led dsb. (Raspberry PI, n.d.)

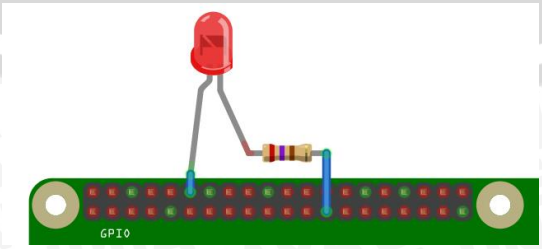


Gambar 2. 4 PIN GPIO pada Raspberry PI

(Sumber: https://www.raspberrypi.org)

Untuk dapat menggunakan GPIO pada raspberry *command-line* maka perlu diinstal WiringPI, WiringPI ini adalah program yang memanage *on-board GPIO interface.*

Contoh penggunaan GPIO pada raspberryPI adalah dengan menyalakan LED. Ketika GPIO pin di set sebagai output dan di set HIGH maka pin tersebut memiliki output sebesar 3.3 volts (3x3).

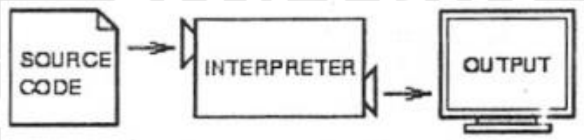


Gambar 2. 5 skema diagram LED menggunakan GPIO

(Sumber: https://www.raspberrypi.org)

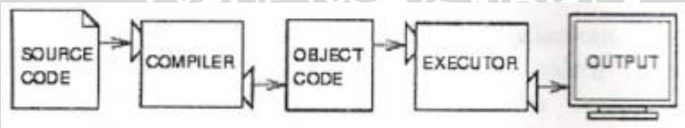
2.2.5 Python

Python merupakan salah satu bahasa pemrograman tingkat tinggi (*High Level Language*) yang bersifat interpreter, interaktif dan berorientasi objek. Ada bahasa tingkat tinggi, pasti juga ada bahasa tingkat rendah (*Low Level Language*), yang berhubungan dengan "bahasa mesin" atau "bahasa assembly". Pada Kenyataannya komputer hanya dapat mengeksekusi program yang ditulis dengan bahasa tingkat rendah. Dengan demikian, suatu program yang menggunakan bahasa tingkat tinggi harus diproses dengan menterjemahkan ke dalam bahasa mesin sebelum program tersebut dijalankan. Terdapat dua proses pemrograman agar suatu program dapat dijalankan oleh komputer, yaitu: Interpreter dan Compiler. Interpreter membaca program baris per baris, sehingga membutuhkan waktu yang lebih sedikit.



Gambar 2. 6 Proses Eksekusi Proses

Sedangkan compiler membaca program secara keseluruhan, yaitu menterjemahkan seluruh instruksi dalam program sekaligus.



Gambar 2. 7 Proses Kompilasi

2.2.5.1 Scipy FFT

Python memiliki banyak *library* bawaan maupun *library* yang harus di install terlebih dahulu. *Library* berguna sebagai *package* fungsi tambahan yang dikembangkan oleh *developer* maupun organisasi python sendiri untuk mempermudah dalam pembuatan aplikasi ataupun *project*. *Library* tersebut bisa didapat dari situs python ataupun github. Scipy adalah *library* untuk membaca wav file dan melakukan FFT.

Metode *Fast Fourier Transfor* pada scipy memiliki persamaan dengan rumus sebagai berikut:

$$y[k] = \sum_{n=0}^{N-1} e^{-2\pi j \frac{kn}{N}} x[n] \dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots\dots (2.2)$$

Metode ini dapat dipanggil dengan cara mengimport *library* pada pemrograman python seperti pada (Gambar 2.7). Sehingga untuk metode FFT pada sistem akan menggunakan fungsi FFT yang ada pada pemrograman Python.

```
>>> from scipy.fftpack import fft, ifft
>>> x = np.array([1.0, 2.0, 1.0, -1.0, 1.5])
>>> y = fft(x)
>>> y
[ 4.50000000+0.j          2.08155948-1.65109876j -1.83155948+1.60822041j
 -1.83155948-1.60822041j  2.08155948+1.65109876j]
>>> yinv = ifft(y)
>>> yinv
[ 1.0+0.j  2.0+0.j  1.0+0.j -1.0+0.j  1.5+0.j]
```

Gambar 2. 8 Contoh coding pemanggilan FFT

Selain FFT, scipy juga dapat menganalisa wavfile dan memanipulasinya. Fungsi tersebut adalah fungsi wavfile.

2.2.5.2 Numpy

Numpy adalah *library* untuk mengakses dan mempermudah data hasil FFT yang berupa array. Fungsi dari numpy antara lain adalah:

1. Arange
2. Split
3. Ceil
4. Log10
5. Where

(Scipy, n.d.)

2.2.6 Longest Common Subsequence (LCS)

Longest Common Subsequences adalah masalah untuk mencari uparangkaian bersama (common subsequence) terpanjang dari 2 buah atau lebih rangkaian. Uparangkaian (subsequence) dari sebuah string S adalah sekumpulan karakter yang ada pada S yang urutan kemunculannya sama. Atau dalam definisi formalnya sebagai berikut:

Rangkaian Z adalah uparangkaian dari X <x1,x2,...,xm>, jika terdapat urutan menaik <i1,i2,...,ik> yang merupakan indeks X untuk semua j=1,2,..,k, yang memenuhi xi=zj (Cormen, et al., 2010).

Misal pada S = GAATACA, beberapa uparangkaian yang mungkin adalah : GAT, TCA, dan GC. Uparangkaian bersama (common subsequence) dari 2 rangkaian adalah uparangkaian yang terdapat pada kedua rangkaian tersebut. Misal S1 = GATTTAC dan S2 = AGATC, maka GAT, GATC, maupun GAC adalah uparangkaian bersama dari S1 dan S2. Uparangkaian bersama terpanjang (longest common subsequences) adalah uparangkaian bersama dari 2 rangkaian yang paling panjang (Cormen, et al., 2010).

2.2.7 Mikrofon Karbon dan *Soundcard External*

2.2.7.1 Mikrofon Karbon

Mikrofon adalah alat pengeras suara yaitu mengubah gelombang suara menjadi sinyal listrik. Mikrofon berasal dari bahasa Yunani mikros yang berarti kecil dan fon yang berarti suara atau bunyi. Jenis mikrofon salah satunya adalah mikrofon karbon. Terbuat dari sebuah diagram logam yang terletak pada salah satu ujung kotak logam yang berbentuk silinder. Cara kerjanya berdasarkan resistansi variabel dimana terdapat sebuah penghubung yaitu diafragma dihubungakan dengan butir-butir karbon di dalam mikrofon. Mikrofon ini mudah didapatkan pada radio dan tape yang memiliki fungsi perekaman. Harga dari mikrofon karbon juga relative murah.

Gambar 2. 9 Mikrofon Jenis Karbon

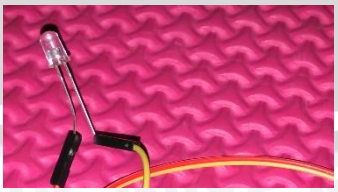
2.2.7.2 Soundcard External

Sound card adalah perubah sinyal analog dari mikrofon menjadi sinyal digital. Pada raspberry pi 2 tidak terdapat microfon internal dan Analog Digital Coonverter. Sehingga dibutuhkan Sound card External atau USB Sound Card dalam penelitian ini.

Gambar 2. 10 Soundcard External

2.2.8 Light Emitting Diode (LED)

LED adalah komponen elektronika yang dapat memancarkan cahaya ketika diberi tegangan. LED terdiri dari berbagai warna antara lain adalah, merah, hijau, dsb.

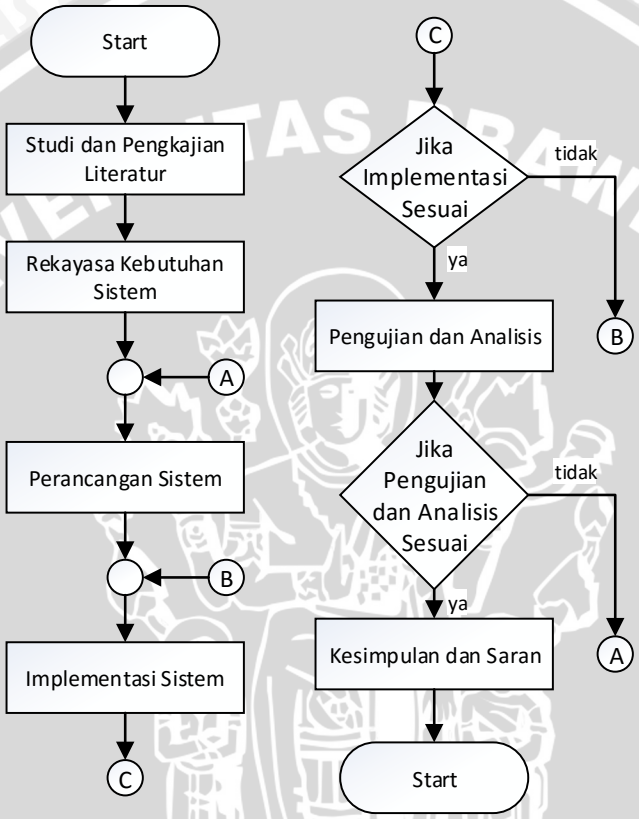


Gambar 2. 11 Gambar LED

BAB 3 METODOLOGI

3.1 Metodologi Penelitian

Pada bab ini akan menjelaskan langkah-langkah yang akan ditempuh dalam penyusunan skripsi. Tipe penelitian ini adalah implementatif pengembangan, sehingga penulis membuat rancangan penelitian meliputi studi dan pengkajian literature, analisis kebutuhan sistem, perancangan sistem, implementasi sistem, pengujian dan analisis, kesimpulan dan saran.



Gambar 3. 1 Flowchart metode penelitian

3.1.1 Studi Literatur

Studi literatur yaitu proses pengumpulan bahan-bahan refrensi baik dari buku, artikel, paper, jurnal dan sumber lain yang dapat dipertanggung jawabkan mengenai metode FFT, Efek Doppler, serta beberapa referensi lainnya untuk menunjang pencapaian tujuan penelitian.

3.1.2 Rekayasa Kebutuhan Sistem

Dari studi pustaka yang telah dilakukan, maka penulis kemudian akan melakukan rekayasa kebutuhan mengenai fungsional dari sistem tersebut. Kebutuhan Fungsional tersebut antara lain sebagai berikut:

3.1.2.1 Analisa Suara Sirene Ambulans

Suara sirene ambulans memiliki karakteristik frekuensi dengan pattern yang berulang-ulang (tidak acak). Sehingga pada penelitian ini alat akan menganalisa frekuensi dari suara sirene ambulans beserta *pattern* suara. Untuk dapat menganalisa frekuensi dari suara sirene ambulans, maka dibutuhkan indikator dari frekuensi tersebut.

Adanya pergerakan ambulans menuju lampu lalu lintas mengakibatkan adanya efek *Dopple* yang menyebabkan perubahan range frekuensi yang didapat oleh alat. Sehingga dilakukan perhitungan seberapa besar selisih frekuensi yang akan didapat oleh alat yang nantinya akan didapat range frekuensi baru dari penambahan efek Doppler. Sehingga analisa ini akan dijelaskan secara detail pada Bab Rekayasa kebutuhan. Pada bab ini akan dijelaskan mengenai hal hal sebagai berikut:

1. Analisa Frekuensi Bunyi Sirene Ambulans.
   Analisa ini menggunakan sampling bunyi dari lakesma UB yang di analisa menggunakan python.
2. Perhitungan Frekuensi Bunyi dengan Efek Doppler.

3.1.2.2 Sistem Kerja Alat

Sistem kerja alat ini yaitu suara sirene ambulans di tangkap dengan menggunakan microphone, lalu Soundcard akan merekam suara dari microphone menjadi format wav yang nantinya format wav ini yang akan dianalisa pada Raspberry Pi.

Pada Raspberry yang telah diberi OS Raspbian (Linux) suara dari soundcard tadi akan ditangkap oleh Aplikasi bernama arecord sehingga dapat disimpan menjadi format audio .wav. format .wav ini lah yang akan di analisa frekuensi maupun patternnya. Suara ambulans tadi akan direkam tiap detik secara terus menerus dengan tahap sebagai berikut:

Gambar 3. 2 Tahap Pendeteksian Ambulans

Analisa .wav menggunakan aplikasi python dikarenakan pemrograman ini memiliki banyak library salah satunya dalam hal ini adalah FFT. Pada program akan dilakukan perekaman, analisa dan output secara terus menerus. Pada perekaman akan menggunakan comman linux yaitu arecord. Pada analisa alat dibutuhkan indikator frekuensi dari sirene ambulans yang telah memperhitungkan efek Doppler. Indikator frekuensi tersebut akan diperjelas pada analisa indikator frekuensi, dan pembahasan lebih lanjut mengenai proses pendeteksian akan dijelaskan secara lengkap pada Bab Perancangan dan Implementasi.

Hasil dari analisa tadi akan dijadikan trigger untuk output yang berupa led. Pada output berupa led, raspberry menggunakan program tambahan untuk

mendeteksi GPIO pada raspberry. Sehingga program penelitian dapat menggunakan fungsi GPIO untuk menyalakan led.

3.1.2.3 Rekayasa Kebutuhan Perangkat Keras

Perangkat keras pada sistem ini digunakan sebagai pengolah data dari *input* sehingga menghasilkan *output.* Beberapa perangkat keras yang dibutuhkan oleh sistem adalah sebagai berikut

1. 1 buah Raspberry Pi sebagai unit untuk memproses data.
2. 1 buah Mikrofon sebagai alat untuk merekam suara.
3. 1 buah Sound Card External sebagai alat untuk merubah sinyal analog dari Mikrofon menjadi sinyal digital yang dapat diproses oleh Raspberry Pi.
4. SD Card untuk tempat sistem operasi pada raspberry.
5. led berwarna hijau sebagai penanda pendeteksian ambulans.

3.1.2.4 Rekayasa Kebutuhan Perangkat Lunak

Perangkat Lunak pada system ini dugunakan sebagai penunjang dari perangkat keras dalam mengola program pengujian. Perangkat lunak yang dibutuhkan untuk mendukung kerja pada sistem adalah sebagai berikut:

1. Raspbian OS sebagai Sistem Operasi dari Raspberry.
2. Raspberry Library: arecord, dan gpio.
3. Python sebagai bahasa pemrograman program pendeteksian sirene ambulans.
4. Python library: scipy, numpy.
5. Python dan Audacity untuk menganalisa frekuensi pada raspberry.

3.1.3 Perancangan Perangkat Keras dan Perangkat Lunak

Pada tahap ini dijelaskan tentang seluruh komponen yang dibutuhkan untuk membuat Deteksi Sirene Ambulans Menggunakan Metode FFT Dengan Mempertimbangkan Efek Doppler Pada RaspberryPi, sehingga dapat diimplementasikan.

Pada perancangan perangkat keras akan dijelaskan mengenai perancangan perangkat keras sedemikian sehingga perangkat keras dapat menunjang program pendeteksian sirene ambulans.

Pada perancangan perangkat lunak akan dijelaskan mengenai perancangan program, flowchart, sehingga program dapat menganalisa sirene ambulans dan dapat mendeteksi keberadaan ambulans.

3.1.4 Implementasi Sistem

Implementasi sistem dilaksanakan sesuai dengan desain sistem yang telah dibuat sebelumnya. Pada implementasi sistem ini memiliki beberapa tahapan-tahapan yaitu:

1. Implementasi perangkat keras. Pada tahap ini akan diaplikasikan komponen-komponen *hardware* yang sesuai dengan rancangan. Konfigurasi pada masing-

masing perangkat dilakukan untuk memastikan bahwa hardware yang ada telah terhubung dengan baik

2. Implementasi perangkat lunak. Beberapa perangkat lunak pendukung akan diaplikasikan untuk menunjang jalannya sistem. Perangkat lunak tersebut meliputi python sebagai bahasa pemrograman, segala library python yang menunjang.
3. Implementasi Program pendeteksian sirene ambulans.

Dari kombinasi implementasi diatas, diharapkan sistem dapat berjalan sesuai dengan apa yang diharapkan yaitu Dapat mendeteksi sirene ambulans dengan menggunakan metode FFT yang nantinya akan diterapkan pada RaspberryPi dengan menggunakan pemrograman Python.

3.1.5 Pengujian dan Analisis

Skema pengujian pada penelitian ini adalah sebagai berikut:

1. Pengujian Suara sirene ambulans diam. Pada pengujian ini digunakan bunyi asal dari ambulans tanpa ada suara noise dari sekitar lampu lalu lintas. Pada penelitian ini dibedakan berdasarkan buyi ambulans.
2. Pengujian Suara sirene ambulans diam dengan background suara dari sekitar Lampu Lalu Lintas, yaitu menggabungkan bunyi dari pengujian pertama dengan bunyi yang di dapat dari sekitar lampu lalulintas. Sehingga didapat persentase keakuratan sistem. Dikarenakan lama waktu lampu merah minimal yang paling optimal adalah 58 detik (Hayun & Sundari, 2005). Maka hasil penelitian di katakan tidak terdeteksi apabila didapat lama waktu terdeteksi lebih dari 58 detik. Pada penelitian ini juga akan dibedakan berdasarkan jarak dari sumber suara ambulans yaitu 5 meter, 15 meter, dan 25 meter.
3. Pengujian pada Suara sirene dengan keadaan ambulans bergerak. Kecepatan rata-rata pengguna kendaraan bermotor pada lampu lalu lintas adalah 22 km/jam - 25km/jam (Hariyanto, 2004). Sehingga pada pengujian ini penulis mengguakan kecepatan 25km untuk menguji keberhasilan alat. Dikarenakan lama waktu dari ambulans menuju lampu lalu lintas sampai ambulans pergi meninggalkan lampu lalu lintas adalah sekitar 20-30 detik, jadi apabila waktu pendeteksian lebih dari 20 detik maka penelitian dianggap tidak terdeteksi. Pada penelitian ini akan didapat rata – rata persentase keberhasilan dari alat akibat mempertimbangkan efek Doppler dan tanpa mempertimbangkan efek doppler.

Suara dari pengujian penelitian diatas menggunakan audio external. Pengujian pada suara sirene dengan keadaan ambulans bergerak dilakukan di UB dan Perumahan PJB Paiton. Suara persimpangan di dapat dari persimpangan lampu lalulintas depan UB Soekarno Hatta, Persimpangan Lalu lintas Dekat Araya, dan Persimpangan lalu lintas arak ke batu singosari.

Analisis pada penelitian ini meliputi:

1. Bagaimana waktu deteksi ambulans menggunakan metode *Fast Fourier Transform* pada otomatisasi *traffic light*?
2. Bagaimana keakuratan deteksi ambulans menggunakan metode *Fast Fourier Transform* pada otomatisasi *traffic light*?
1. Bagaimana pengaruh efek *Doppler* pada pendeteksian ambulans dengan metode *Fast Fourier Transform*?

3.1.6 Kesimpulan dan Saran

Kesimpulan didapatkan setelah melakukan perancangan, implementasi, pengujian dan analisis kepad sistem. Kesimpulan diambil berdasarkan hasil pengujian dan analisis sistem yang telah dibuat. Isi dari kesimpulan diharapkan dapat menjadi acuan pada penelitian lain untuk mengembangkan teknologi untuk membantu dalam pendeteksian sirene ambulans pada traffic light yang lainnya. Selain itu, pada akhir penulisan terdapat saran yang bertujuan untuk memperbaiki kesalahan-kesalahan yang terjadi dan menyempurnakan penulisan serta untuk memberikan pertimbangan atas pengembangan sistem selanjutnya.

BAB 4 REKAYASA KEBUTUHAN

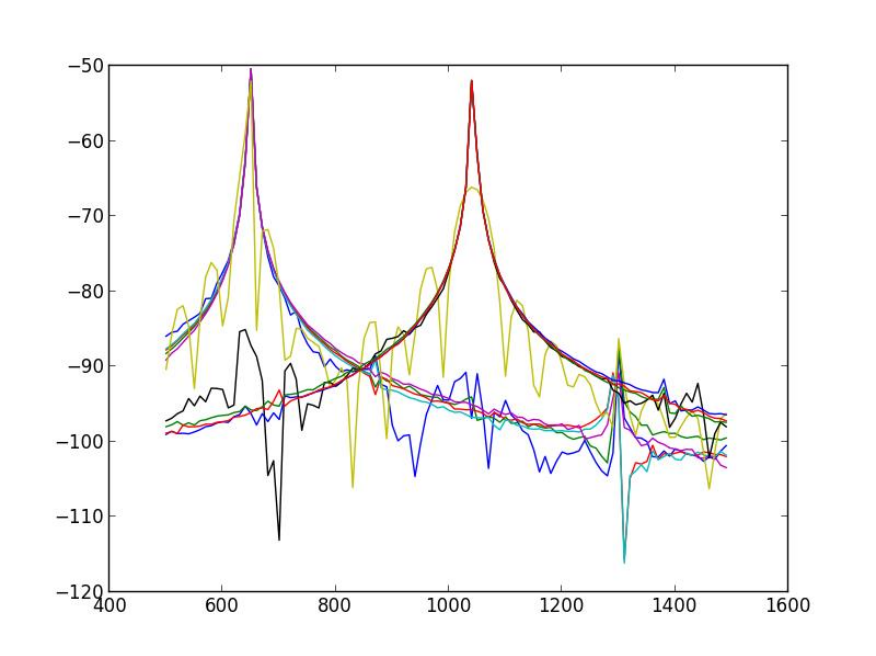
4.1 Analisa Frekuensi Sirene Ambulans

Pada analisa frekuensi sirene ambulans ini, dibedakan berdasarkan macam-macam variasi bunyi sirene ambulans yaitu hilo, wail, phaser, dan yelp.

4.1.1 Hilo

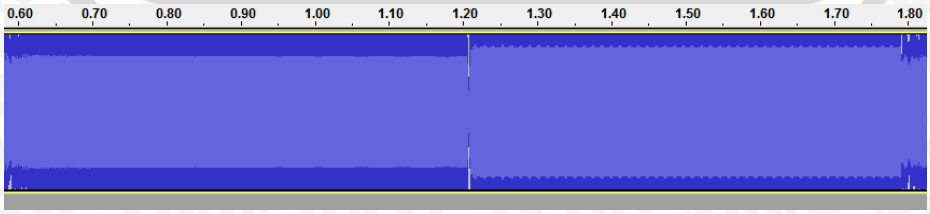
Bunyi hilo adalah bunyi sirene ambulans tinggi dan rendah secara berulang dengan waktu tertentu. Bunyi ini memiliki karakteristik frekuensi dan pola perulangan sebagai berikut.

Berdasarkan analisa menggunakan python, hilo memiliki frekuensi rendah sebesar 650Hz dan memiliki frekuensi tinggi sebesar 1400Hz



Gambar 4. 1 Hilo menggunakan python frekuensi sebesar 650Hz dan 1040Hz

Berdasarkan analisa menggunakan Audacity, hilo memiliki pola bunyi low selama 0.6s dan high selama 0.6s. sehingga dapat dituliskan pola dari bunyi hilo adalah sebagai berikut: LLLLLL HHHHHH LLLLLL HHHHHH …

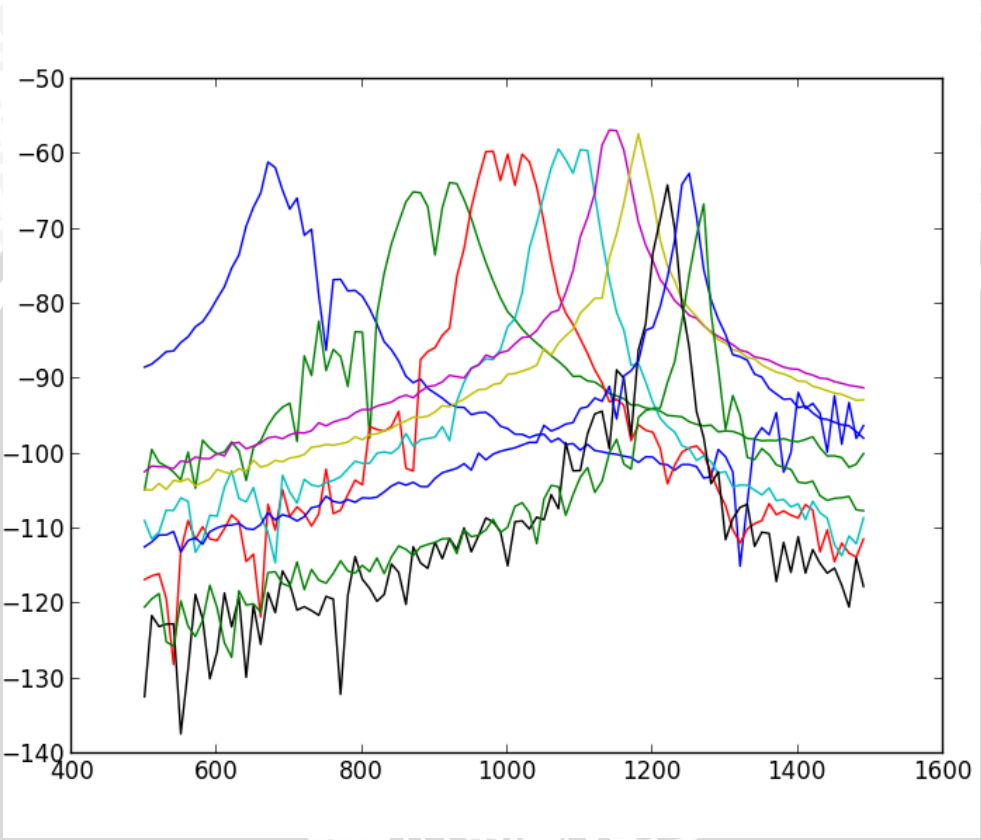


Gambar 4. 2 Pola frekuensi Hilo Audacity

4.1.2 Wail

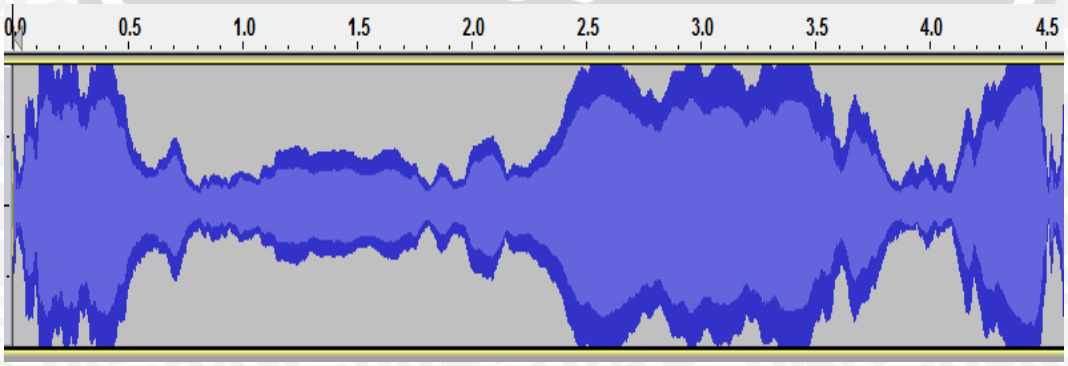
Bunyi Wail adalah bunyi sirene ambulans tinggi dan rendah secara berulang dengan waktu tertentu. Bunyi ini memiliki karakteristik frekuensi dan pola perulangan sebagai berikut

Berdasarkan analisa menggunakan python, wail memiliki frekuensi rendah sebesar 670Hz dan memiliki frekuensi tinggi sebesar 1280Hz.



Gambar 4. 3 Wail menggunakan python frekuensi sebesar 670Hz dan 1280Hz

Berdasarkan analisa menggunakan Audacity, pola dari wail tak dapat terdeteksi.



Gambar 4. 4 Pola Frekuensi Wail Audacity

Tetapi dari hasil pengamatan menggunakan python pola dari wail ini LLLLL HHHHH atau LLLLLL HHHHHH dengan kenaikan frekuensi atau penurunan frekuensi yang stabil. Missal: 1070-1050-1020-1000-970-940-910-880-850-810.

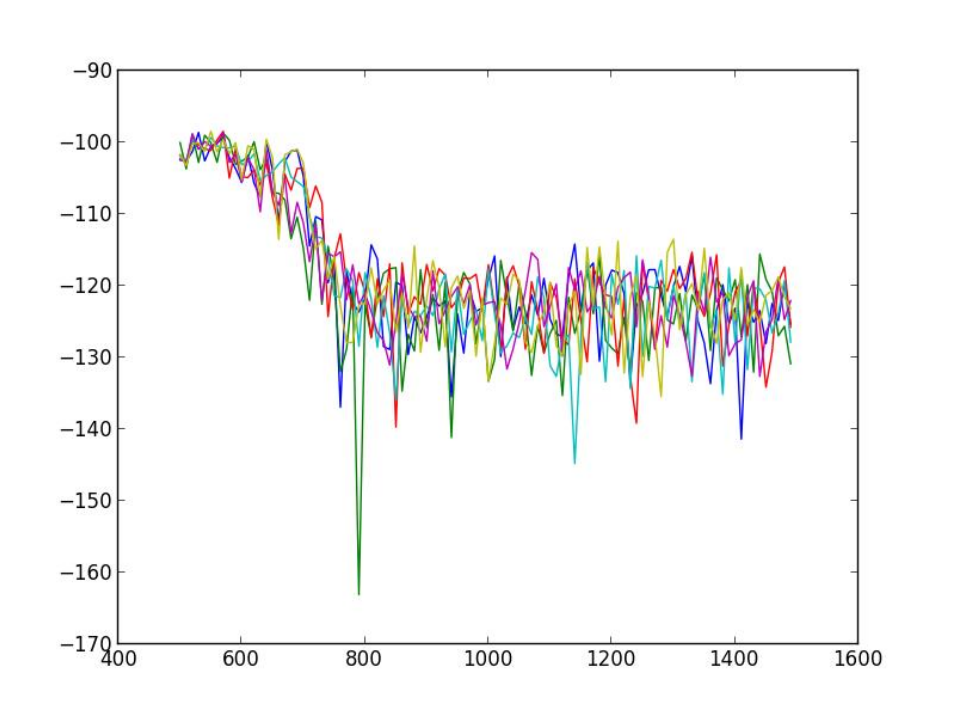
```
-----databaseFreq: [610.0, 660.0, 790.0, 560.0, 530.0, 1210.0, 1190.0, 1170.0, 1150.0, 1130.0]
2. traffic Working : 0 / ['0', '0', '0', '0', '0', 0, 0] - 0:00:01.159451
-----databaseFreq: [1070.0, 1050.0, 1020.0, 1000.0, 970.0, 940.0, 910.0, 880.0, 850.0, 810.0]
3. traffic Working : 1 / ['0', '0', '0', '0', 0, 0, 1] - 0:00:01.145826
-----databaseFreq: [730.0, 700.0, 680.0, 670.0, 860.0, 920.0, 1070.0, 1120.0, 1160.0, 1200.0]
4. traffic Working : 2 / ['0', '0', '0', 0, 0, 1, 2] - 0:00:01.141444
-----databaseFreq: [1280.0, 1280.0, 1300.0, 1300.0, 1310.0, 1320.0, 1320.0, 1330.0, 1320.0, 1310.0]
5. traffic Working : 3 / ['0', '0', 0, 0, 1, 2, 3] - 0:00:01.095972
-----databaseFreq: [1260.0, 1250.0, 1230.0, 1210.0, 1190.0, 1180.0, 1160.0, 1140.0, 1110.0, 1100.0]
6. traffic Working : 4 / ['0', 0, 0, 1, 2, 3, 4] - 0:00:01.097087
-----databaseFreq: [1040.0, 1010.0, 990.0, 960.0, 940.0, 900.0, 870.0, 850.0, 800.0, 780.0]
7. traffic Working : 5 / [0, 0, 1, 2, 3, 4, 5] - 0:00:01.147536
-----databaseFreq: [690.0, 670.0, 670.0, 890.0, 940.0, 1100.0, 1140.0, 1180.0, 1210.0, 1240.0]
8. traffic Working : 6 / [0, 1, 2, 3, 4, 5, 6] - 0:00:01.084576
-----databaseFreq: [1280.0, 1290.0, 1300.0, 1310.0, 1320.0, 1320.0, 1330.0, 1320.0, 1310.0, 1300.0]
9. ambulan detected : 10 / [1, 2, 3, 4, 5, 6, 7] - 0:00:01.094788
-----databaseFreq: [1250.0, 1230.0, 1220.0, 1200.0, 1180.0, 1160.0, 1140.0, 1120.0, 1100.0, 1080.0]
10. ambulan detected : 11 / [2, 3, 4, 5, 6, 7, 11] - 0:00:01.094757
-----databaseFreq: [1020.0, 990.0, 970.0, 940.0, 910.0, 880.0, 850.0, 810.0, 780.0, 760.0]
11. ambulan detected : 12 / [3, 4, 5, 6, 7, 11, 12] - 0:00:01.096812
-----databaseFreq: [680.0, 670.0, 710.0, 880.0, 1040.0, 1110.0, 1150.0, 1190.0, 1230.0, 1250.0]
12. ambulan detected : 12 / [4, 5, 6, 7, 11, 12, 12] - 0:00:01.138754
-----databaseFreq: [1290.0, 1300.0, 1310.0, 1310.0, 1320.0, 1330.0, 1320.0, 1310.0, 1300.0, 1280.0]
13. ambulan detected : 12 / [5, 6, 7, 11, 12, 12, 12] - 0:00:01.095702
```

Gambar 4. 5 pola frekuensi Wail python

4.1.3 Phaser

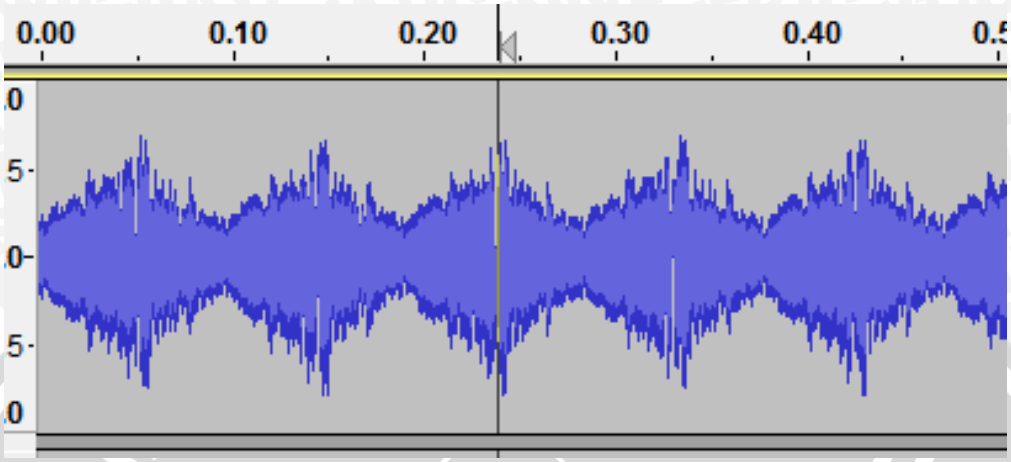
Bunyi Phaser adalah bunyi sirene ambulans tinggi dan rendah secara berulang dengan waktu tertentu. Bunyi ini memiliki karakteristik frekuensi dan pola perulangan sebagai berikut

Berdasarkan analisa menggunakan python, phaser memiliki frekuensi sebagai berikut:



Gambar 4. 6 Phaser menggunakan python frekuensi sebesar 760Hz

Berdasarkan analisa menggunakan Audacity, phaser memiliki pola bunyi low dan high kurang lebuh 0,1s dalam 1 siklus. Sehingga pada pemrograman python pola bunyi ini hanya dapat terdeteksi frekuensi rendah saja ataupun tinggi saja. python menganalisa frekuensi dalam satuan 0,1s, Sehingga pattern yang dapat dikenali adalah LLLLL LLLLL

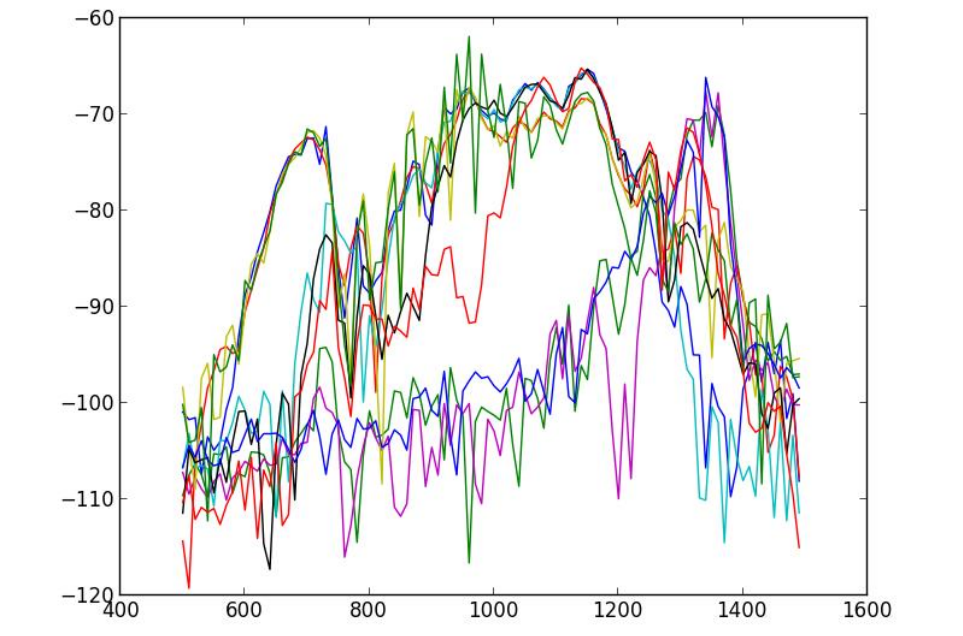


Gambar 4. 7 Pola frekuensi phaser Audacity

4.1.4 Yelp

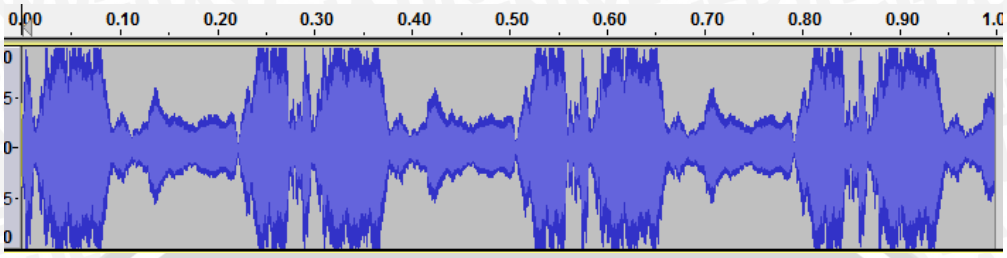
Bunyi Yelp adalah bunyi sirene ambulans tinggi dan rendah secara berulang dengan waktu tertentu. Bunyi ini memiliki karakteristik frekuensi dan pola perulangan sebagai berikut

Berdasarkan analisa menggunakan python, yelp memiliki frekuensi sebagai berikut:



Gambar 4. 8 Yelp menggunakan python frekuensi sebesar 960Hz-1360Hz

Berdasarkan analisa menggunakan Audacity, yelp memiliki pola bunyi low dan high dalam 1 detik sebanyak 2-4 siklus. Sehingga bunyi rendah dan tinggi sangat cepat, sama seperti phaser, tetapi pada python frekuensi bunyi yelp yang tinggi yang lebih dominan. Yaitu 960-1360Hz. Sedangkan bunyi rendah hanya tertangkap sebagian. Sehingga pola dari yelp adalah HHHHH HHHHH



Gambar 4. 9 Pola frekuensi yelp Audacity

4.1.5 Hasil Analisa Frekuensi Sirene Ambulans

Berdasarkan data diatas, maka dapat dibuat table sebagai berikut:

Tabel 4. 1 Frekuensi dan Pola Sirene Ambulans

Bunyi	Frekuensi		Pola
	Rendah	Tinggi	
Hilo	650Hz	1040Hz	LLLLLL HHHHHH Tiap 1 detik ada low 4-6x ataupun high 4-6x (score = 0 = -1 = 1)
Wail	670Hz	1280Hz	LLLLL HHHHH Kenaikan atau penurunan stabil (score >4)
Phaser	760Hz	-	LLLLL LLLLL
Yelp	-	960 – 1360Hz	HHHHH HHHHH

Dari table diatas. Dapat ditarik kesimpulan bahwa frekuensi rendah antara 650-760Hz dan Frekuensi Tinggi antara 810-1360Hz.

4.2 Perhitungan Frekuensi dengan Pengaruh Efek Doppler

Pada kondisi ambulans yang bergerak, terjadi pergeseran frekuensi yang dinamakan efek Doppler. Efek Doppler ini dapat dihitung berdasarkan persamaan (2.1)

$$f_{p} = f_s \times \frac{v \pm v_p}{v \pm v_s}$$

- $v_p = 0$
- $v_s = 0 \sim 25\ km/h = 0 \sim 6.94\ m/s$
- $v = 340\ m/s$
- Tanda + untuk v_p dipakai bila pendengar bergerak mendekati sumber bunyi.
- Tanda - untuk v_p dipakai bila pendengar bergerak menjauhi sumber bunyi.
- Tanda + untuk v_s dipakai bila sumber bunyi bergerak menjauhi pendengar.
- Tanda - untuk v_s dipakai bila sumber bunyi bergerak mendekati pendengar.

Pada perhitungan efek Doppler ini kecepatan ambulans rata - rata adalah 0 - 25km/jam atau 0 - 6.94m/s (Hariyanto, 2004). Sedangkan kecepatan suara diudara adalah 340m/s (Young & Freedman, 2012). dan dianggap ambulans mendekati lampu lalu lintas sehingga perhitungannya adalah sebagai berikut:

1. Frekuensi rendah dari sirene ambulans berkisar 650-760Hz sehingga dengan memperhitungkan efek Doppler pergeseran frekuensi adalah:
   a. Dengan menganggap batas bawah frekuensi rendah diasumsikan sebagai ambulans diam 0 m/s, maka perhitungan frekuensi adalah sebagai berikut:

      $$f_{p} = f_s \times \frac{340m/s \pm 0m/s}{340m/s \pm 0m/s}$$

      $$f_{p} = 650Hz \times 1$$

      $$f_{p} = 650Hz$$

   b. Dengan menganggap batas atas frekuensi rendah diasumsikan sebagai ambulans dengan kecepatan 25km/s, maka perhitungan frekuensi adalah sebagai berikut:

      $$f_{p} = f_s \times \frac{340m/s \pm 0m/s}{340m/s - 6.94m/s}$$

      $$f_{p} = 760Hz \times 1.0208$$

      $$f_{p} = 776Hz$$

   c. Kesimpulan frekuensi rendah setelah terpengaruh efek Doppler adalah: 650Hz – 776Hz.
2. Frekuensi tinggi dari sirene ambulans berkisar 810-1360Hz sehingga dengan memperhitungkan efek Doppler pergeseran frekuensi adalah:
   a. Dengan menganggap batas bawah frekuensi tinggi diasumsikan sebagai ambulans diam 0 m/s, maka perhitungan frekuensi adalah sebagai berikut:

      $$f_{p} = f_s \times \frac{340m/s \pm 0m/s}{340m/s \pm 0m/s}$$

$f_p = 810Hz \times 1$

$f_p = 810Hz$

b. Dengan menganggap batas atas frekuensi tinggi diasumsikan sebagai ambulans dengan kecepatan 25km/s, maka perhitungan frekuensi adalah sebagai berikut:

$$f_p = f_s \times \frac{340m/s \pm 0m/s}{340m/s - 6.94m/s}$$

$f_p = 1360Hz \times 1.0208$

$f_p = 1388Hz$

c. Kesimpulan frekuensi tinggi setelah terpengaruh efek Doppler adalah: 810Hz – 1388Hz.

Dari perhitungan frekuensi diatas, didapat kesimpulan sebagai berikut:

1. Range frekuensi sirene ambulans setelah dipengaruhi efek Doppler adalah Frekuensi Rendah antara 650Hz – 776Hz dan Frekuensi Tinggi antara 810Hz – 1388Hz.
2. Range frekuensi sirene ambulans tanpa dipengaruhi efek Doppler adalah Frekuensi Rendah antara 650Hz – 760Hz dan Frekuensi Tinggi antara 810Hz – 1360Hz.

Berdasarkan pola yang di dapat, maka pada penelitian ini untuk menentukan suara sirene ambulans diperlukan 2 metode tambahan yaitu:

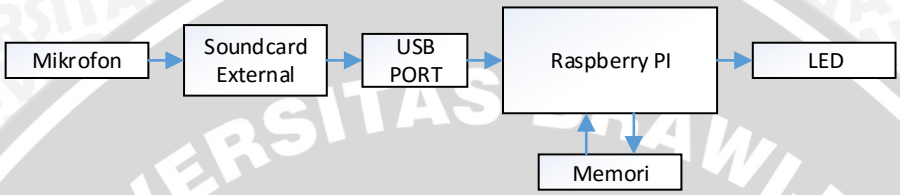
1. LCS (*Longest Common Subsequence*) untuk menganalisa pattern misal: LLLL HHHH LLLL HHHH.
2. Scoring adalah penjumlahan nilai dari frekuensi yang digunakan untuk menganalisa karakteristik berikutnya (Misal: wail kenaikan atau penurunan stabil yaitu scoring > 4, atau hilo jumlah High dan Low berbanding lurus yaitu score = 0 dsb).

BAB 5 PERANCANGAN DAN IMPLEMENTASI

5.1 Perangkat Keras

5.1.1 Diagram Blok

Pada perancangan perangkat keras ini meliputi pemasangan alat sesuai dengan diagram blok pada gambar 5.1. perangkat keras dibagi menjadi 3 bagian yaitu sensor mikrofon dengan soundcard eksternal (input), raspberry (proses), dan LED (output).



Gambar 5. 1 Diagram Blok Sistem

5.1.2 Implementasi Perangkat Keras

Pada gambar 5.1 diagram blok sistem telah dijelaskan rangkaian alat, berikut adalah tampilan alat yang dapat dilihat pada gambar 5.2.

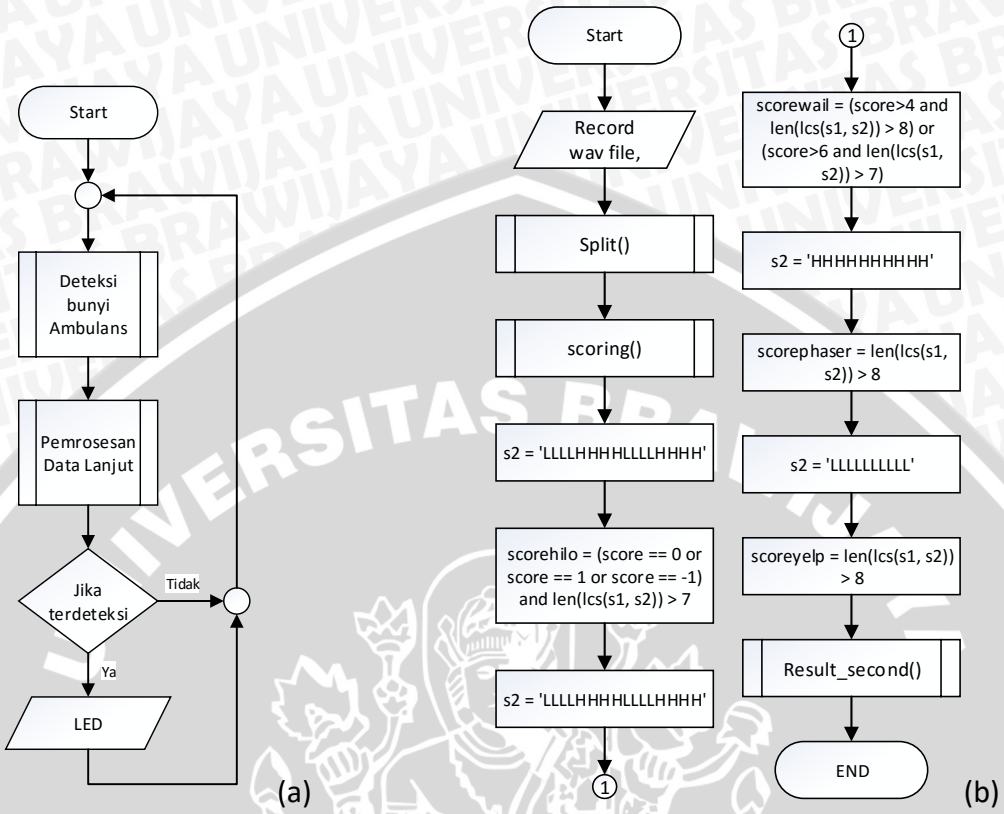
Gambar 5. 2 alat pendeteksi sirene ambulan

5.2 Perangkat Lunak

5.2.1 Diagram Alir Keseluruhan

Diagram Alir keseluruhan menggambarkan system kerja dari alat yang akan dibuat. Pada diagram alir system secara keseluruhan ini dimulai dari start kemudian mikrofon akan menangkap sinyal suara. Sinyal suara ini lalu dikenali raspberry dengan bantuan Soundcard sehingga sinyal suara tadi menjadi format wav. Format wav ini akan diolah oleh raspberry dan dianalisa apakah terdeteksi ambulans apa tidak. Dari hasil analisa sirene ambulans akan dilakukan pemrosesan

data lanjut. Kemudian hasil dari keputusan tadi yang akan ditampilkan dalam bentuk LED. Diagram alir sistem keseluruhan dapat dilihat pada gambar 5.3 (a).



Gambar 5. 3 (a) Diagram Alir sistem keseluruhan

(b) Flowchart Deteksi Bunyi Ambulans

5.2.2 Deteksi Bunyi Ambulans

Pada diagram alir deteksi bunyi ambulans ini menggambarkan proses pendeteksian ambulans pada sistem meliputi pendeteksian frekuensi dengan metode *Fast Fourier Transform,* penganalisaan pattern dengan menggunakan LCS dan Scoring. Sehingga didapat hasil per satuan detik yaitu terdeteksi ambulans atau tidak. flowchart bisa dilihat pada gambar 5.3 (b)

Pada perancangan ini menggunakan Bahasa pemrograman python. Tahapan dari pendeteksian ambulans mula-mula program merekam bunyi ambulans dengan menggunakan command arecord per 1 detik. Lalu akan dilakukan split dari 1 detik menjadi 0.1 detik sebanyak 10 sample yg disebut chunk. Chunk ini akan di deteksi frekuensinya menggunakan fft. Lalu hasil dari fft ke 10 chunk ini akan di beri label sesuai data analisa frekuensi yaitu:

1. Frekuensi Rendah antara 650Hz – 776Hz: L
2. Frekuensi Tinggi antara 810Hz – 1388Hz: H
3. Selain frekuensi diatas: N

Setelah diberi label, ke 10 label chunk akan dmasukkan kedalam array. Array ini yang akan di analisa menggunakan lcs dan scoring. Kemudian akan dilakukan analisa sbagai berikut:

1. Hilo pattern: LLLLHHHHLLLLHHHH
   - jika scoring = 0 atau -1 atau 1 dan score LCS > 7
2. Wail pattern: LLLLHHHHLLLLHHHH
   - jika scoring>4 dan score LCS > 8
   - jika scoring>6 dan score LCS > 7
3. Yelp pattern: LLLLLLLLL
   - jika score LCS > 8
4. Phaser pattern: HHHHHHHHHH
   - jika score LCS > 8

Dari analisa perdetik tadi, didapat score akhir. Score inilah yang akan menyalakan led. Diasumsikan jika led menyala maka ambulans terdeteksi. Dan jika led mati maka ambulans tak terdeteksi. Sehingga apabila score akhir ambulans adalah lebih dari sama dengan 7 maka led menyala. Apabila score kurang dari 7 maka led mati. Logika seperti ini memiliki kelemahan butuh waktu 7 detik dari awal pendeteksian sampai ambulans dikenali atau lampu menyala. Dan jika score akhir sudah maksimal yaitu 12. Butuh waktu 5 detik untuk mematikan lampu. Oleh karenanya dibutuhkan pemrosesan data lanjut dalam pengambilan keputusan.

5.2.2.1 Fungsi Deteksi Sirene Ambulan

Berikut adalah detail fungsi deteksi sirene ambulan.

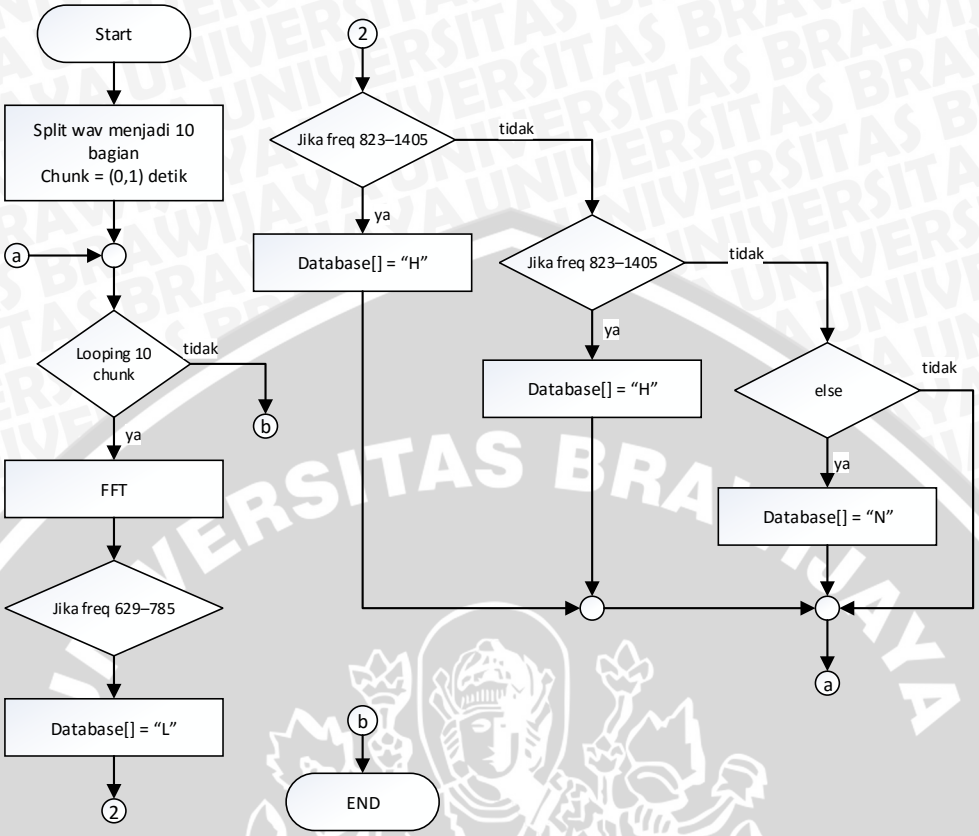
1. Record suara

Pada record suara akan dilakukan dilakukan perekaman suara menjadi file wav. Dan membaca file wav tadi menjadi array data. Berikut adalah potongan program record suara.

```
def record():
    command = "arecord -D plughw:0 -d 1 1.wav"
    system(command)
    fs, snd = wavfile.read('1.wav')
    snd = snd / (2.**15)
    timeArray = arange(0, 8000.0, 1)
    timeArray /= fs
    timeArray *= 1000
    return snd, timeArray
```

2. Fungsi split

Pada fungsi split akan dilakukan split, fft, dan labeling. Split bertujuan memecah file wav yang semula 1detik menjadi 0.1 detik. Dan fft bertujuan mencari frekuensi dari setiap 0.1detik data wav. Sedangkan labeling adalah penamaan frekuensi menjadi L, H, atau N. Flowchart fungsi split dapat dilihat pada gambar 5.4.



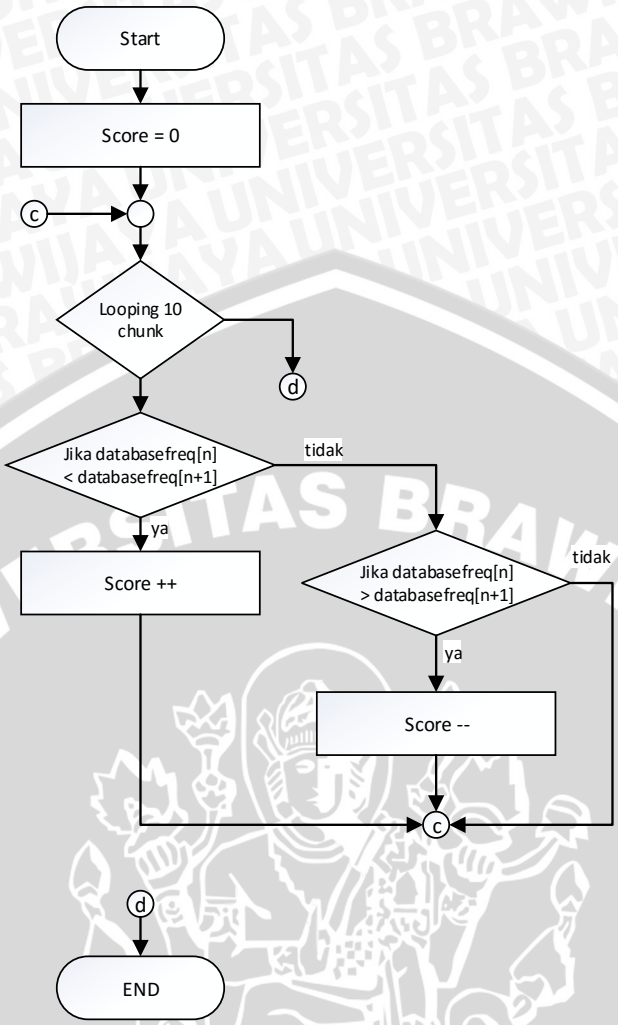
Gambar 5. 4 Flowchart fungsi split

Dari fungsi ini akan didapat database label dan database frekuensi. Berikut adalah potongan program fungsi split.

```
def splits(snd, timeArray, many=0):
    if many == 50:
        many = 20
    else:
        many = 10
    a = split(snd,many)
    b = split(timeArray,many)
    databasefreq = []
    database = []
    #false, high, low = 0, 0, 0
    i=0
    while i<many:
    n = len(a[i])
    n = fft(a[i]) #fft dari python
```

3. Fungsi scoring

Pada fungsi ini akan dihitung score dari setiap 1 detik file wav. Flowchart lengkap fungsi scoring dapat dilihat pada gambar 5.5.



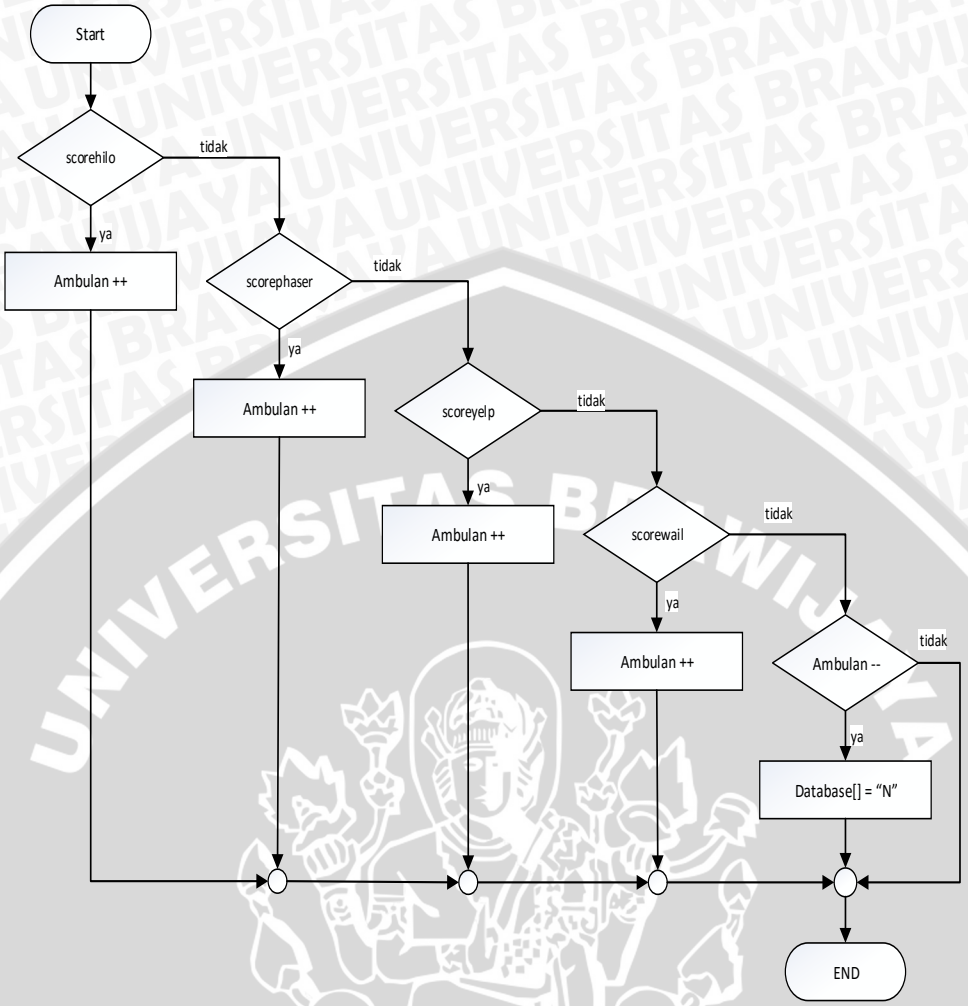
Gambar 5. 5 Flowchart fungsi scoring

Dari fungsi ini akan didapat score dari 1 detik file wav. Berikut adalah potongan program fungsi scoring.

```
def scoring(databasefreq):
    z, score=0, 0
    while z<9:
        if databasefreq[z] < databasefreq[z+1]:
            score += 1
        elif databasefreq[z] > databasefreq[z+1]:
            score -= 1
        z += 1
```

4. Fungsi result second

Pada fungsi ini akan didapat score akhir dari ambulans. Score akhir ini yang akan menyalakan led pada program dan menandakan apakah terdapat ambulans atau tidak. Flowchart lengkap fungsi result second dapat dilihat pada gambar 5.6.



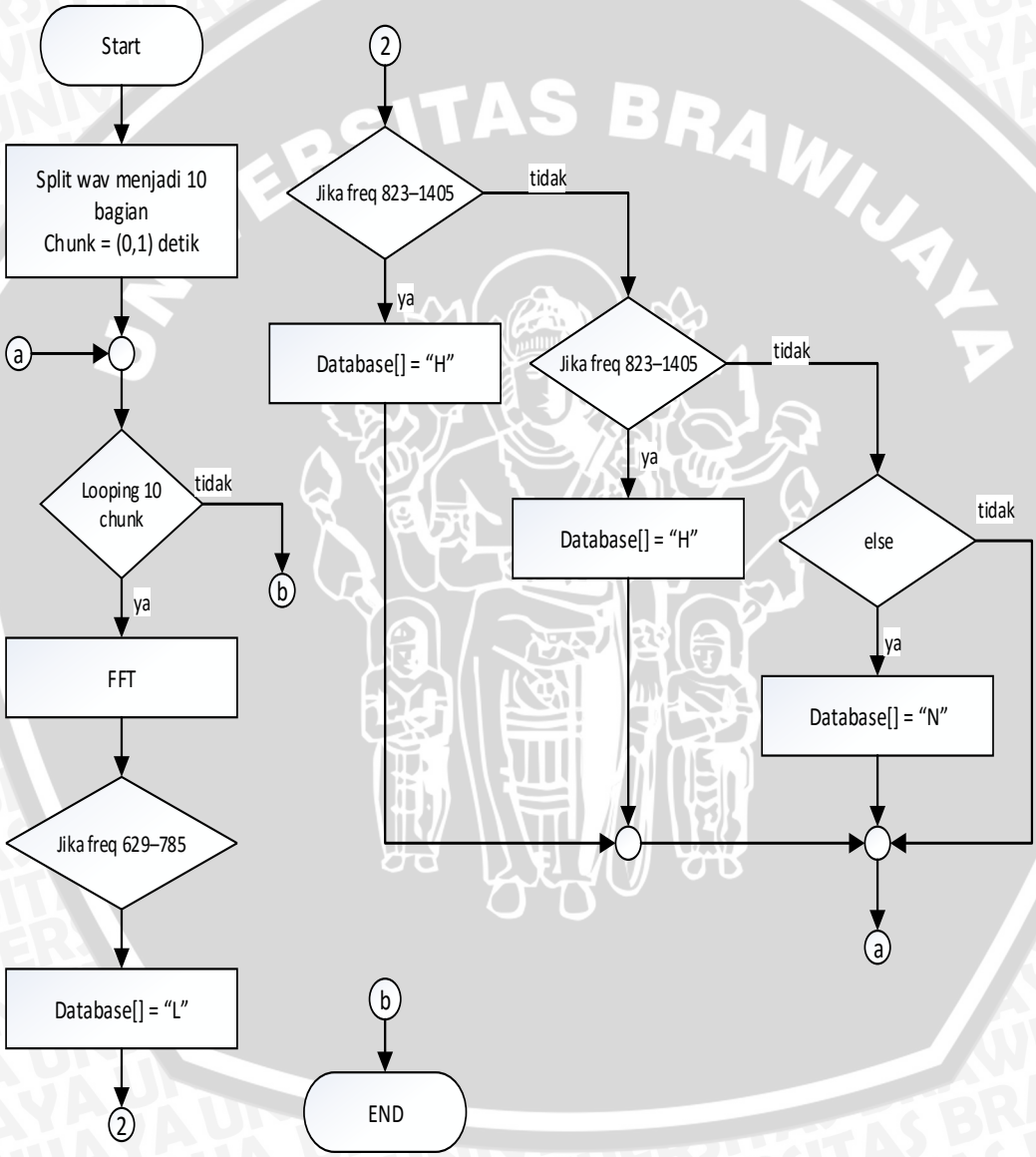
Gambar 5. 6 Flowchart Fungsi Result Second

Berikut adalah potongan kode dari fungsi result second

```
def
result_second(scorehilo,scorewail,scorephaser,scoreyel
p,ambulan):
    if scorehilo:
        #print 'ambulance - hilo'
        if (ambulan<12):
            ambulan = ambulan +1
    elif scorewail:
        #print 'ambulance - wail'
        if (ambulan<12):
            ambulan = ambulan +1
    elif scorephaser:
        #print 'ambulan - phaser'
        if (ambulan<12):
            ambulan = ambulan +1
```

5.2.3 Pemrosesan Data Lanjut

Pada diagram alir hasil output ini menggambarkan proses pengambilan keputusan dari diagram alir deteksi bunyi ambulans yang memiliki output dalam rentan waktu 1 detik. Sehingga supaya dapat diterapkan pada lampu lalu lintas dibutuhkan filter tambahan yaitu analisa keputusan yang berupa pemrosesan data lanjut. Pemrosesan data lanjut ini mengumpulkan data dan menganalisa pola data sehingga didapat kesimpulan dari pattern data. Kesimpulan inilah yang akan menentukan apakah LED akan menyala atau ditidak. Diagram alir hasil output dapat dilihat pada gambar 5.7



Gambar 5. 7 Flowchart Fungsi Pemrosesan Data Lanjut

Pemrosesan data lanjut ini digunakan agar hasil dari pendeteksian ambulans dalam satuan detik dapat dianalisa peluang adanya ambulans. Tahapan pemrosesan data lanjut adalah penyimpanan 7 detik data dalam array yang kemudian di analisa dengan menggunakan lgika sebagai berikut:

1. Jika dalam 3 detik terdeteksi ambulans secara terus menerus, maka data score akan ditambahkan 2 sehingga led langsung menyala apabila hal ini berulang sebanyak 2x (dua detik). Oleh karena itu data mining berhasil mereduce waktu terdeteksi ambulans yang semula 7 detik menjadi 5 detik.
2. Dan jika dalam 3 detik tak terdeteksi ambulans secara terus menerus, maka data score akan dikurangi 2 sehingga score cepat dibawah 10 dan led akan mati. Oleh karena itu data mining berhasil mereduce waktu tak terdeteksi ambulans yang semula 5 detik menjadi 4 detik.

Berikut adalah potongan code dari fungsi pemrosesan data lanjut

```
def data_mining(L,ambulan):


    L.append(ambulan)
    del L[0]
    #[0, 0, 1, 2, 3, 4, 5]


    Hasil1 = L[6] > L[5] and L[5] > L[4] and L[4] >
L[3] and L[3] > L[2] and ambulan<10


    Hasil2 = L[6] < L[5] and L[5] < L[4] and L[4] <
L[3] and ambulan > 2


    if( hasil1 ):
        ambulan = ambulan + 2


    elif( hasil2 ):
      ambulan = ambulan - 2


    return L,ambulan
```

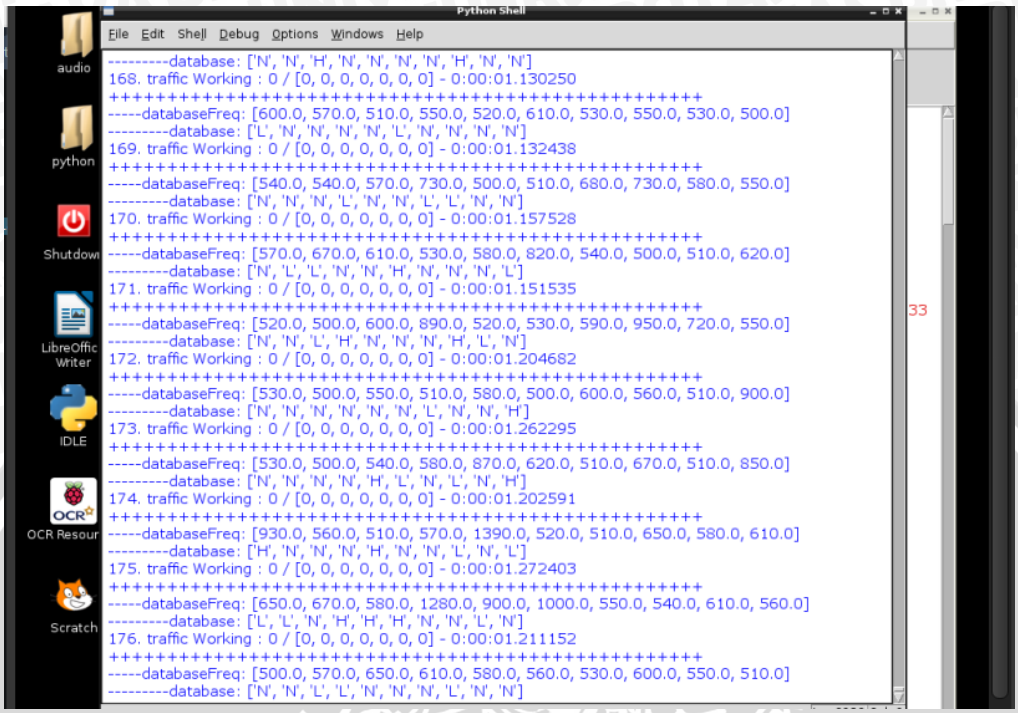
5.2.4 Implementasi Perangkat Lunak

Pada implementasi perangkat lunak program akan dijalankan pada raspberry Pi dengan bantuan PC sehingga dapat dilihat hasil dari penelitian ini seperti pada gambar 5.8.

Pada gambar 5.8, databaseFreq adalah hasil frekuensi yang didapat tiap satu detik. Database adalah hasil label frekuensi dengan ketentuan sebagai berikut:

1. Frekuensi Rendah antara 650Hz – 776Hz: L
2. Frekuensi Tinggi antara 810Hz – 1388Hz: H
3. Selain frekuensi diatas: N

Traffic Working menandakan bahwa alat tidak mendeteksi adanya ambulans dan lampu indikator tidak menyala. Sedangkan ambulance detected menandakan bahwa telah terdeteksi sirene ambulans sehingga lampu led indikator nyala.



Gambar 5. 8 Implementasi Perangkat Lunak

BAB 6 PENGUJIAN DAN ANALISIS

6.1 Pengujian

Pada pengujian sistem ini akan membahas tentang pengujian hasil pendeteksian alat pada berbagai kondisi. Kondisi tersebut antara lain: pengujian pada persimpangan lampu lalu lintas (tanpa ambulan), pengujian pada saat ambulans tidak bergerak. Dan pengujian pada saat ambulans bergerak.

6.1.1 Pengujian ambulans diam pada tempat sunyi

Pada pengujian ini alat akan dicoba ke empat bunyi ambulans pada saat ambulans behenti atau diam. Pengujian ini bertujuan untuk mengetahui persentase keberhasilan alat tanpa adanya pengaruh efek Doppler pada suara ambulans. Pengujian ini dilakukan di tempat sunyi bukan pada persimpangan lampu lalu lintas. Pada pengujian ini apabila waktu terdeteksi lebih dari 58 detik maka ambulans dianggap tak terdeteksi. Pengujian ini dibedakan berdasarkan jarak yaitu: 5 meter, 10 meter, dan 20 meter. Dari hasil pengujian pada ambulans berhenti atau diam, didapatkan hasil seperti pada tabel berikut:

Tabel 6. 1 Ambulans diam jarak 5 meter dari alat

NO	Percobaan	Hilo	Yelp	Wail	Phaser	Rata-Rata
1	Percobaan 1	5 detik	5 detik	5 detik	5 detik	5 detik
2	Percobaan 2	5 detik	5 detik	13 detik	5 detik	7 detik
3	Percobaan 3	5 detik	5 detik	5 detik	5 detik	5 detik
4	Percobaan 4	5 detik	5 detik	7 detik	5 detik	5.5 detik
5	Percobaan 5	5 detik	5 detik	13 detik	5 detik	7 detik
6	Percobaan 6	5 detik	5 detik	7 detik	5 detik	5.5 detik
7	Percobaan 7	5 detik	5 detik	14 detik	5 detik	7.25 detik
8	Percobaan 8	5 detik	6 detik	14 detik	5 detik	7.5 detik
9	Percobaan 9	5 detik	5 detik	5 detik	5 detik	5 detik
10	Percobaan 10	5 detik	7 detik	6 detik	5 detik	5.75 detik
Rata-Rata		5 detik	5.3 detik	8.9 detik	5 detik	6.05 detik

Pada tabel 6.1 tidak ada waktu yang melebihi 58 detik, maka persentase keberhasilan dari alat sebesar: 100% (persentase 1) dengan rata-rata waktu terdeteksi sebesar 6.05 detik(waktu1).

Tabel 6. 2 Ambulans diam jarak 10 meter dari alat

NO	Percobaan	Hilo	Yelp	Wail	Phaser	Rata-Rata
1	Percobaan 1	5 detik	5 detik	5 detik	9 detik	6 detik

2	Percobaan 2	5 detik	5 detik	9 detik	6 detik	6.25 detik
3	Percobaan 3	5 detik	5 detik	5 detik	7 detik	5.5 detik
4	Percobaan 4	5 detik	5 detik	7 detik	5 detik	5.5 detik
5	Percobaan 5	5 detik	5 detik	5 detik	5 detik	5 detik
6	Percobaan 6	7 detik	5 detik	5 detik	9 detik	6.5 detik
7	Percobaan 7	5 detik	5 detik	13 detik	7 detik	7.5 detik
8	Percobaan 8	5 detik	5 detik	8 detik	8 detik	6.5 detik
9	Percobaan 9	5 detik	5 detik	11 detik	6 detik	6.75 detik
10	Percobaan 10	5 detik	5 detik	7 detik	7 detik	6 detik
Rata-Rata		5.2 detik	5 detik	7.5 detik	6.9 detik	6.15 detik

Pada tabel 6.2 tidak ada waktu yang melebihi 58 detik, maka persentase keberhasilan dari alat sebesar: 100% (persentase 2) dengan rata-rata waktu terdeteksi sebesar 6.15 detik(waktu2).

Tabel 6. 3 Ambulans diam jarak 20 meter dari alat

NO	Percobaan	Hilo	Yelp	Wail	Phaser	Total Rata-Rata
1	Percobaan 1	5 detik	5 detik	14 detik	6 detik	7.5 detik
2	Percobaan 2	5 detik	5 detik	15 detik	9 detik	8.5 detik
3	Percobaan 3	7 detik	5 detik	8 detik	6 detik	6.5 detik
4	Percobaan 4	8 detik	5 detik	5 detik	6 detik	6 detik
5	Percobaan 5	5 detik	5 detik	5 detik	11 detik	6.5 detik
6	Percobaan 6	7 detik	5 detik	5 detik	7 detik	6 detik
7	Percobaan 7	5 detik	5 detik	8 detik	5 detik	5.75 detik
8	Percobaan 8	5 detik	5 detik	11 detik	8 detik	7.25 detik
9	Percobaan 9	5 detik	5 detik	13 detik	5 detik	7 detik
10	Percobaan 10	9 detik	5 detik	6 detik	8 detik	7 detik
Rata-Rata		6.1 detik	5 detik	9 detik	7.1 detik	6.8 detik

Pada tabel 6.3 tidak ada waktu yang melebihi 58 detik, maka persentase keberhasilan dari alat sebesar: 100% (persentase 3) dengan rata-rata waktu terdeteksi sebesar 6.8 detik(waktu3).

6.1.2 Pengujian ambulans diam pada persimpangan lampu lalu lintas

Pada pengujian ini alat akan dicoba ke empat bunyi ambulans pada saat ambulans behenti atau diam. Pengujian ini bertujuan untuk mengetahui persentase keberhasilan alat tanpa adanya pengaruh efek Doppler pada suara ambulans. Pada pengujian ini apabila waktu terdeteksi lebih dari 58 detik maka ambulans dianggap tak terdeteksi. Pengujian ini dilakukan di persimpangan lampu lalu lintas. Pengujian ini dibedakan berdasarkan jarak yaitu: 5 meter, 10 meter, dan 20 meter.

Dari hasil pengujian pada ambulans berhenti atau diam pada persimpangan lampu lalu lintas, didapatkan hasil seperti pada tabel berikut:

Tabel 6. 4 Ambulans diam pada persimpangan jarak 5 meter dari alat

NO	Percobaan	Hilo	Yelp	Wail	Phaser	Rata-Rata
1	Percobaan 1	5 detik	5 detik	6 detik	5 detik	5.25 detik
2	Percobaan 2	5 detik	11 detik	5 detik	5 detik	6.5 detik
3	Percobaan 3	6 detik	11 detik	5 detik	5 detik	6.75 detik
4	Percobaan 4	5 detik	12 detik	5 detik	5 detik	6.75 detik
5	Percobaan 5	5 detik	9 detik	10 detik	5 detik	7.25 detik
6	Percobaan 6	5 detik	20 detik	11 detik	5 detik	10.25 detik
7	Percobaan 7	5 detik	7 detik	5 detik	5 detik	5.5 detik
8	Percobaan 8	5 detik	5 detik	11 detik	5 detik	6.5 detik
9	Percobaan 9	5 detik	5 detik	15 detik	5 detik	7.5 detik
10	Percobaan 10	5 detik	6 detik	15 detik	5 detik	7.75 detik
Rata-Rata		5.1 detik	9.1 detik	8.8 detik	5 detik	7 detik

Pada tabel 6.4 tidak ada waktu yang melebihi 58 detik, maka persentase keberhasilan dari alat sebesar: 100% (persentase 4) dengan rata-rata waktu terdeteksi sebesar 7 detik(waktu4).

Tabel 6. 5 Ambulans diam pada persimpangan jarak 10 meter dari alat

NO	Percobaan	Hilo	Yelp	Wail	Phaser	Rata-Rata
1	Percobaan 1	5 detik	5 detik	8 detik	12 detik	7.5 detik
2	Percobaan 2	5 detik	6 detik	15 detik	6 detik	8 detik
3	Percobaan 3	5 detik	16 detik	5 detik	10 detik	9 detik
4	Percobaan 4	9 detik	21 detik	11 detik	5 detik	11.5 detik
5	Percobaan 5	5 detik	7 detik	15 detik	5 detik	8 detik

6	Percobaan 6	5 detik	5 detik	5 detik	5 detik	5 detik
7	Percobaan 7	5 detik	5 detik	7 detik	5 detik	5.5 detik
8	Percobaan 8	5 detik	5 detik	21 detik	6 detik	9.25 detik
9	Percobaan 9	5 detik	5 detik	5 detik	10 detik	6.25 detik
10	Percobaan 10	5 detik	5 detik	19 detik	6 detik	8.75 detik
Rata-Rata		5.4 detik	8 detik	11.1 detik	7 detik	7.875 detik

Pada tabel 6.5 tidak ada waktu yang melebihi 58 detik, maka persentase keberhasilan dari alat sebesar: 100% (persentase 5) dengan rata-rata waktu terdeteksi sebesar 7.875 detik(waktu5).

Tabel 6. 6 Ambulans diam pada persimpangan jarak 20 meter dari alat

NO	Percobaan	Hilo	Yelp	Wail	Phaser	Rata-Rata
1	Percobaan 1	5 detik	7 detik	8 detik	10 detik	7.5 detik
2	Percobaan 2	7 detik	9 detik	6 detik	10 detik	8 detik
3	Percobaan 3	7 detik	7 detik	7 detik	11 detik	8 detik
4	Percobaan 4	6 detik	7 detik	12 detik	5 detik	7.5 detik
5	Percobaan 5	11 detik	6 detik	9 detik	11 detik	9.25 detik
6	Percobaan 6	8 detik	5 detik	9 detik	5 detik	6.75 detik
7	Percobaan 7	5 detik	15 detik	9 detik	5 detik	8.5 detik
8	Percobaan 8	6 detik	5 detik	7 detik	5 detik	5.75 detik
9	Percobaan 9	5 detik	20 detik	10 detik	5 detik	10 detik
10	Percobaan 10	5 detik	6 detik	12 detik	7 detik	7.5 detik
Rata-Rata		6.5 detik	8.7 detik	8.9 detik	7.4 detik	7.875 detik (100%)

Pada tabel 6.6 tidak ada waktu yang melebihi 58 detik, maka persentase keberhasilan dari alat sebesar: 100% (persentase 6) dengan rata-rata waktu terdeteksi sebesar 7.875 detik(waktu6).

6.1.3 Pengujian pada ambulans bergerak

Pada pengujian ini alat akan dicoba ke empat bunyi ambulans pada saat ambulans bergerak atau melaju. Pengujian ini bertujuan untuk mengetahui rata-rata persentase keberhasilan dan keakuratan sistem, sehingga dapat membandingkan antara sistem yang mempertimbangkan efek Doppler dengan sistem yang tidak mempertimbangkan efek doppler. Pada pengujian ini apabila waktu terdeteksi lebih dari 20 detik maka ambulans dianggap tak terdeteksi.

Dari hasil pengujian pada ambulans bergerak atau melaju dengan kecepatan 25km/jam tanpa mempertimbangkan efek doppler, didapatkan hasil seperti pada table 6.7 berikut.

Tabel 6. 7 Hasil Pengujian ambulans bergerak tanpa efek doppler

NO	Percobaan	Hilo	Yelp	Wail	Phaser	Rata-Rata
1	Percobaan 1	5 detik	12 detik	9 detik	10 detik	9 detik
2	Percobaan 2	5 detik	18 detik	12 detik	11 detik	11.5 detik
3	Percobaan 3	6 detik	13 detik	10 detik	12 detik	10.25 detik
4	Percobaan 4	6 detik	15 detik	12 detik	13 detik	11.5 detik
5	Percobaan 5	5 detik	21 detik	11 detik	13 detik	12.5 detik
Rata-Rata		5.4 detik	15.8 detik	10.8 detik	11.8 detik	10.95 detik

Pada tabel 6.7 ada waktu yang melebihi 20 detik sebanyak 1 dari 20 waktu pengujian, maka persentase keberhasilan dari alat sebesar: 95% (persentase 7) dengan rata-rata waktu terdeteksi sebesar 10.95 detik(waktu7).

Dari hasil pengujian pada ambulans bergerak atau melaju dengan kecepatan 25km/jam dengan mempertimbangkan efek doppler, didapatkan hasil seperti pada table 6.8 berikut.

Tabel 6. 8 Hasil Pengujian ambulans bergerak dengan efek doppler

NO	Percobaan	Hilo	Yelp	Wail	Phaser	Rata-Rata
1	Percobaan 1	5 detik	12 detik	8 detik	5 detik	7.5 detik
2	Percobaan 2	6 detik	17 detik	12 detik	5 detik	10 detik
3	Percobaan 3	6 detik	13 detik	9 detik	8 detik	9 detik
4	Percobaan 4	5 detik	16 detik	9 detik	9 detik	9.75 detik
5	Percobaan 5	5 detik	12 detik	9 detik	10 detik	9 detik
Rata-Rata		5.4 detik	14 detik	9.4 detik	7.4 detik	9.05 detik

Pada tabel 6.8 tidak ada waktu yang melebihi 20 detik, maka persentase keberhasilan dari alat sebesar: 100% (persentase 8) dengan rata-rata waktu terdeteksi sebesar 9.05 detik(waktu8).

6.2 Analisis

6.2.1 Waktu Deteksi Ambulans

Waktu deteksi ambulans adalah waktu yang dibutuhkan sistem untuk dapat mendeteksi keberadaan ambulans. Waktu rata-rata adalah sebagai berikut:

a. Waktu sistem diam = rata-rata(waktu1,2,3,4,5,6)
= rata-rata(6.05, 6.15, 6.8, 7, 7.875, 7.875) detik
= 6.958 detik

b. Waktu sistem bergerak tanpa doppler = 10.95 detik

c. Waktu sistem bergerak dengan doppler = 9.05 detik

d. Selisih Waktu sebesar = waktu tanpa Doppler – waktu dengan doppler
= 10.95 detik – 9.05 detik
= 1.90 detik

6.2.2 Keakuratan Deteksi Ambulans

Keakuratan deteksi ambulans dapat dilihat dari persentase keberhasilan alat. Persentase keberhasilan alat rata-rata adalah sebagai berikut:

a. Persentase total diam = rata-rata(persentase1,2,3,4,5,6)
= rata-rata(100%,100%,100%,100%,100%,100%)
= 100%

b. Persentase bergerak tanpa Doppler = 95% dengan error sebesar 5%

c. Persentase bergerak dengan Doppler = 100% dengan error sebesar 0%

6.2.3 Pengaruh Efek Doppler

Dari hasil pengujian pada ambulans bergerak diatas. Maka dapat dilihat perbedaan pendeteksian ambulans dengan mempertimbangkan efek doppler dan tanpa mempertimbangkan efek doppler adalah sebagai berikut:

Tabel 6. 9 Perbedaan memertimbangkan efek doppler dan tidak

NO	Berdasarkan	Persentase	Waktu terdeteksi	error
1	Mempertimbangkan Efek Doppler	100%	9.05 detik	0%
2	Tanpa mempertimbangkan efek doppler	95%	10.95 detik	5%

BAB 7 PENUTUP

7.1 Kesimpulan

1. Waktu pendeteksian ambulans dengan menggunakan metode *Fast Fourier Transform* pada ambulans diam adalah 6.958 detik, pada ambulans bergerak tanpa mempertimbangkan efek *doppler* adalah 9.05 detik, dan pada ambulans bergerak dengan mempertimbangkan efek *doppler* adalah 10.95 detik
2. Keakuratan sistem dari pendeteksian ambulans dengan menggunakan metode *Fast Fourier Transform* pada ambulans diam adalah 100%, pada ambulans bergerak tanpa mempertimbangkan efek Doppler adalah 95%, dan pada ambulans bergerak dengan mempertimbangkan efek Doppler adalah 100%.
3. Pendeteksian ambulans dengan menggunakan metode *Fast Fourier Transform* perlu mempertimbangkan efek *doppler* dikarenakan dengan menggunakan efek *doppler* rata-rata waktu pendeteksian sirene ambulans lebih cepat yaitu sebesar 1,9 detik, dan tanpa mempertimbangkan efek d*oppler* error sebesar 5%, tetapi dengan menggunakan efek *Doppler* error sebesar 0%.

7.2 Saran

1. Sistem ini bias dikembangkan dengan menambah sensor kamera untuk mendeteksi ambulans. Sehingga pada saat ambulans terdeteksi lewat frekuensi, maka akan dilakukan pengolaan citra gambar dari sensor kamera.
2. Lama waktu pendeteksian ambulans pada sistem juga dapat percepat dengan cara menambah sensor kecepatan angin. Sehingga efek Doppler akan diterapkan secara dinamis dan dapat segera mendeteksi sirene ambulans.

DAFTAR PUSTAKA

Anwar, R., 2014. *Radio Republik Indonesia.* [Online]
Available at:
http://www.rri.co.id/post/berita/95107/nasional/indonesia_peringkat_lima_dunia_tingkat_kecelakaan_lalu_lintas.html
[Accessed 12 04 2015].

Ary, 2013. *Dishub Lelang Traffic Light Tenaga Matahari.* [Online]
Available at: http://malang-post.com/kota-malang/72032-dishub-lelang-traffic-light-tenaga-matahari-
[Accessed 13 November 2015].

Cormen, T. H., Leiserson, C. E., Rivest, R. L. & Stein, C., 2010. *Introduction to Algorithms.* 3rd Edition ed. Cambridge , Massachusetts London, England: McGraw-Hill Book Company.

Hariyanto, I. J., 2004. *Sistem Pengendalian Lalu Lintas Pada Pertemuan Jalan Sebidang.* [Online]
Available at: library.usu.ac.id
[Accessed 10 November 2015].

Hayun, A. & Sundari, 2005. Penentuan Penyalaan Lampu Lalu Lintas Yang Optimal : Kasus Persimpangan Buah Batu Lingkar Selatan. *INASEA,* Volume 6, pp. 77-90.

KBBI, n.d. *Kamus Besar Bahasa Indonesia.* [Online]
Available at: http://kbbi.web.id/
[Accessed 14 04 2015].

kompas, 2008. kompas. *Ambulans datang lebih cepat daripada ayam goreng*, 16 05, p. 38.

Lakesma, 2012. *Lakesma Fakultas Kedokteran Universitas Brawijaya.* [Online]
Available at: http://lakesma.ub.ac.id/2012/07/serba-serbi-ambulans/
[Accessed 14 04 2015].

Liaw, J.-J.et al., 2013. Recognition of the Ambulance Siren Sound in Taiwan. *IEEE,* pp. 3826-3828.

Linux Organization, n.d. *linuxcommand.org.* [Online]
Available at: http://linuxcommand.org/man_pages/arecord1.html
[Accessed 1 Oktober 2015].

Mizaki, T., Kitazuno, Y. & Shimakawa, M., 2013. Ambulance Siren Detector using FFT on dsPIC. *IEEE,* pp. 266-269.

Raspberry PI, n.d. *raspberrypi.org.* [Online]
Available at: https://www.raspberrypi.org/documentation/usage/gpio-plus-and-raspi2/
[Accessed Oktober 2015].

Scipy, n.d. *scipy.org.* [Online]
Available at: http://docs.scipy.org/doc/numpy/reference/generated/
[Accessed Oktober 2015].

Sipasulta, R. Y., Arie.S.M.Lumenta ST, M. & Sherwin R.U.A. Sompie, S. M., 2014. Simulasi Sistem Pengacak Sinyal Dengan Metode FFT (Fast Fourier Transform). *E-Journal Teknik Elektro dan Komputer,* p. 1.

UU, 2009. *Dewan Perwakilan Rakyat Republik Indonesia Undang Undang.* [Online]
Available at: http://www.dpr.go.id/dokjdih/document/uu/UU_2009_22.pdf
[Accessed 12 04 2015].

Young, H. D. & Freedman, R. A., 2012. *University Physic With Modern Physics 13th Edition.* 13 ed. USA: Addison-Wesley.

LAMPIRAN A *SOURCE CODE* PROGRAM

```
from os import system
from datetime import datetime
import matplotlib.pyplot as plt
from scipy.io import wavfile
from scipy.fftpack import fft
from numpy import arange
from numpy import split
from numpy import ceil
from numpy import log10
from numpy import where

#source LCS() from : rosettacode.org/wiki/Longest_common_subsequence
access on 18-03-2015 7:33
def lcs(a, b):
    lengths = [[0 for j in range(len(b)+1)] for i in range(len(a)+1)]
    # row 0 and column 0 are initialized to 0 already
    for i, x in enumerate(a):
        for j, y in enumerate(b):
            if x == y:
                lengths[i+1][j+1] = lengths[i][j] + 1
            else:
                lengths[i+1][j+1] = \
                    max(lengths[i+1][j], lengths[i][j+1])
    # read the substring out from the matrix
    result = ""
    x, y = len(a), len(b)
    while x != 0 and y != 0:
        if lengths[x][y] == lengths[x-1][y]:
            x -= 1
        elif lengths[x][y] == lengths[x][y-1]:
            y -= 1
```

```
        else:
            assert a[x-1] == b[y-1]
            result = a[x-1] + result
            x -= 1
            y -= 1
    return result

def ledcontrol(pin, power):
    command = "gpio write {} {}".format(pin, power)
    system(command)

def record():
    command = "arecord -D plughw:0 -d 1 1.wav"
    system(command)

    fs, snd = wavfile.read('1.wav')
    #print snd.dtype #it says dtype('uint8')
    snd = snd / (2.**15)
    #print snd.shape #it says (8000,)

    timeArray = arange(0, 8000.0, 1)
    timeArray /= fs
    timeArray *= 1000

    #plot time audio
    #plt.plot(timeArray, snd)
    #name = 'audio'
    #plt.savefig(name)
    #plt.clf()
    return snd, timeArray

#splits(snd, timeArray, many) many = 50 if split in 50ms or 20 split. default many
is 100 or split is 10 times
```

```
def splits(snd, timeArray, many=0):
    if many == 50:
        many = 20
    else:
        many = 10

    a = split(snd,many)
    b = split(timeArray,many)

    databasefreq = []
    database = []
    #false, high, low = 0, 0, 0
    i=0
    while i<many:
        n = len(a[i])
        p = fft(a[i]) #fft dari python

        x = ceil((n+1)/2.0)
        p = p[0:x]
        p = abs(p)

        p = p / float(n)
        p = p**2

        if n %2 > 0:
            p[1:len(p)] = p[1:len(p)] *2
        else:
            p[1:len(p) - 1] = p[1:len(p) - 1]*2

        freqArray = arange(0,x,1.0)*10
        freqArray = freqArray[50:150] #filter pembatas dari 500Hz-1500Hz
        p = 10*log10(p)
```

```
    p = p[50:150]#filter pembatas dari 500Hz-1500Hz

    #plt.plot(freqArray, p)
    #name = 'plot {}'.format(i)
    #plt.savefig(name)

    maxs = where(p==p.max()) #mencari db maximal
    maxs = freqArray[maxs[0][0]] #berisi freq dari db maksimal

    databasefreq.append(maxs)

    if maxs >= 650 and maxs <= 776:
        database.append('L')
        #low += 1
    elif maxs >= 810 and maxs <= 1388:
        database.append('H')
        #high += 1
    else:
        database.append('N')
        #false += 1
    i+=1
#print 'database: {}'.format(database)
#print 'false: {}'.format(false)

#simpl way
#if low>3 and high >3:
    #print '{} - ambulan'.format(datetime.now() - start)
#else:
    #print '{} - no ambulan'.format(datetime.now() - start)

#scoring frekuennsi
return database, databasefreq
```

```
def scoring(databasefreq):
    z, score=0, 0
    while z<9:
        if databasefreq[z] < databasefreq[z+1]:
            score += 1
        elif databasefreq[z] > databasefreq[z+1]:
            score -= 1
        z += 1
    return score
    #print 'score: {}'.format(score)

def result_second(scorehilo,scorewail,scorephaser,scoreyelp,ambulan):
    if scorehilo:
        #print 'ambulance - hilo'
        #print 'ambulance'
        if (ambulan<12):
            ambulan = ambulan +1
    elif scorewail:
        #print 'ambulance - wail'
        #print 'ambulance'
        if (ambulan<12):
            ambulan = ambulan +1
    elif scorephaser:
        #print 'ambulan - phaser'
        #print 'ambulance'
        if (ambulan<12):
            ambulan = ambulan +1
    elif scoreyelp:
        #print 'ambulan - yelp'
        #print 'ambulance'
        if (ambulan<12):
```

```
            ambulan = ambulan +1
        else:
            #print '--no--'
            if (ambulan>0):
                ambulan = ambulan -1
        return ambulan


def result_akhir(pin1,no,ambulan,L,start):
        if(ambulan >= 7):
            ledcontrol(pin1, 1)
            #ledcontrol(blue, 0)
            print '{}. ambulan detected : {} / {} - {}'.format(no, ambulan,L,
datetime.now() - start)
        else:
            ledcontrol(pin1, 0)
            #ledcontrol(blue, 1)
            print '{}. traffic Working : {} / {} - {}'.format(no, ambulan,L, datetime.now() -
start)
        no = no+ 1
        print "++++++++++++++++++++++++++++++++++++++++++++++++++++++++++"
        return no


def data_mining(L,ambulan):
        L.append(ambulan)
        del L[0]
        #[0, 0, 1, 2, 3, 4, 5]
        if( L[6] > L[5] and L[5] > L[4] and L[4] > L[3] and L[3] > L[2] and ambulan<10 ):
            ambulan = ambulan + 2
        elif(L[6] < L[5] and L[5] < L[4] and L[4] < L[3] and ambulan > 2):
            ambulan = ambulan - 2
        return L,ambulan


def inisialisasi():
```

```
    no = 1
    L = [0,0,0,0,0,0,0]
    pin1 = 7
    ambulan = 0
    #pin2 = 8
    command = "gpio mode {} out".format(pin1)
    system(command)
    #command = "gpio mode {} out".format(pin2)
    #system(command)
    return L,no,pin1,ambulan

def main(ambulan,L,no,pin1):
    start = datetime.now()

    snd, timeArray = record()

    database, databasefreq = splits(snd, timeArray)
    print '-----databaseFreq: {}'.format(databasefreq)
    print '---------database: {}'.format(database)

    score = scoring(databasefreq);

    #LCS
    s1 = ''.join(database)
    #print 'S1: {}'.format(s1)

    #hilo
    s2 = 'LLLLHHHHLLLLHHHH'
    #print 'LCS: {}'.format(lcs(s1, s2))
    #print 'LCS Value: {}'.format(len(lcs(s1, s2)))
    scorehilo = (score == 0 or score == 1 or score == -1) and len(lcs(s1, s2)) > 7
```

```
    #wail
    s2 = 'LLLLHHHHLLLLHHHH'
    #print 'LCS: {}'.format(lcs(s1, s2))
    #print 'LCS Value: {}'.format(len(lcs(s1, s2)))
    #print 'Sore Value: {}'.format(score)
    scorewail = (score>4 and len(lcs(s1, s2)) > 8) or (score>6 and len(lcs(s1, s2)) > 7)

    #phaser
    s2 = 'HHHHHHHHHH'
    scorephaser = len(lcs(s1, s2)) > 8

    #yelp
    s2 = 'LLLLLLLLL'
    scoreyelp = len(lcs(s1, s2)) > 8

    #kesimpulan perdetik
    ambulan = result_second(scorehilo,scorewail,scorephaser,scoreyelp,ambulan)

    #data mining
    L,ambulan = data_mining(L,ambulan)

    #kesimpulan score akhir
    no = result_akhir(pin1,no,ambulan,L,start)

    #print datetime.now() - start
    return ambulan,no
#-------------------------------------------------------------------------------------------
L,no,pin1,ambulan = inisialisasi()
while True:
    ambulan,no = main(ambulan,L,no,pin1)
```