ANALISA DAN SIMULASI PENENTUAN RUTE PADA APLIKASI NAVIGASI KENDARAAN BERMOTOR DI KOTA MAKASSAR

Levi Oktavian Tandungan
Program Studi Teknik Informatika, Fakultas Teknologi Informasi, Universitas Atma Jaya Makassar
Alamat e-mail: levlev.oktavian@gmail.com

ABSTRACT

In recent years, living standards in Indonesia have begun to increase and the desire for people to buy vehicles continues to increase. This encourages the growth of the number of motor vehicles to continue to increase rapidly. According to the latest data taken in June 2017, the number of vehicles in the city of Makassar is 1,425,635. When compared with a decade earlier the growth in the number of vehicles had increased by more than one hundred percent, which in 2007 the number of vehicles was 613,315. The growth in the number of vehicles is also driving congestion and air pollution. Congestion itself can hamper the economy because it can hamper the production and distribution process, Smart City is a concept that is offered especially in traffic-related problems such as the nearest route information to avoid congestion. One technology that can be applied in Smart City infrastructure is a wireless sensor network. The application of the wireless sensor network itself forms a network formed by a number of Arduino sensor nodes where each sensor node detects physical phenomena that occur at that node. Against the results of the sensor node accumulation is simulated with the weight of the distance of the road. The accumulation results are processed using the Dijkstra algorithm and the Floyd-Warshall algorithm and then an analysis of the simulation results is performed. Research analysis and simulation of motor vehicle navigation applications in the city of Makassar produces a route from each algorithm to a simulation of traffic conditions and also analyzes the results of the two algorithms from the selection of the lane and the speed of determining the lane.

Keywords: *Arduino, Dijkstra, Floyd-Warshall, Smart City, Wireless Sensor Network.*

1. PENDAHULUAN

Kemacetan merupakan salah satu masalah lalu lintas yang dihadapi oleh negara berkembang seperti Indonesia dan biasa terjadi di daerah perkotaan yang padat. Dewasa ini kemacetan sudah menjadi bagian dari ciri khas suatu kawasan atau pusat perkotaan tertentu dikarenakan waktu terjadinya yang rutin terutama pada waktu-waktu puncak seperti yang biasa dikenal dengan jam pergi kantor, jam pulang kantor, akhir pekan dan hari libur.

Banyak dampak yang dihasilkan oleh kemacetan dan bersifat negatif. Ditinjau dari berbagai aspek, kemacetan menimbulkan banyak kerugian baik dari segi materi, waktu dan tenaga. Seperti dari aspek ekonomi kemacetan menghambat proses produksi dan distribusi sehingga laju perekonomian menjadi terganggu. Dari aspek kesehatan pun kemacetan menyumbangkan dampak negatif yaitu mempengaruhi kondisi fisik dan psikis para pengguna lalu lintas, terlebih lagi bagi mereka yang kemudian melakukan berbagai aktivitas seperti bekerja, belajar dan lain sebagainya.

Makassar merupakan salah satu kota yang mengalami kemajuan yang pesat. Sebagai kota yang mengalami kemajuan pesat pasti memiliki beberapa masalah perkotaan, salah satu diantaranya adalah masalah kemacetan lalu lintas di jalan raya. Kemacetan ini timbul karena semakin tingginya volume kendaraan pribadi yang tidak dibarengi dengan pembangunan infrastruktur yang cepat dan kurang disiplinnya para pengendara dalam menggunakan kendaraannya. Menurut data terakhir yang di ambil pada Juni 2017, jumlah kendaraan di kota Makassar adalah 1.425.635. Jika dibandinganken dengan satu dekade sebelumnya pertumbuhan jumlah kendaraan itu sudah meningkat lebih dari seratus persen yang mana pada 2007 jumlah kendaraan itu adalah 613.315 [1].

Berdasarkan persoalan tersebut maka dibutuhkan pengelolaan yang lebih baik sehingga masyarakat kota tetap merasa

nyaman dan aman dalam berkendara. Smart city menjadi salah satu konsep yang ditawarkan. Smart city dapat dikatakan sebuah kota yang memonitor dan mengintegrasikan kondisi seluruh infrastruktur yang penting, seperti jalan raya, jembatan, terowongan, rel, kereta bawah tanah, bandar udara, pelabuhan, komunikasi, air, listrik, dan bangunan utama, dapat lebih mengoptimalkan sumber daya yang dimiliki, rencana aktifitas pemeliharaan yang preventif, dan memantau aspek keamanan sekaligus memaksimalkan layanan kepada warganya. Smart city dikembangkan dengan memperkenalkan smart system yang dirancang untuk kepentingan penduduk dan lingkungan.

Teknologi yang dapat diterapkan dalam infrastruktur smart city adalah wireless sensor network. Sebuah wireless sensor network adalah jaringan yang dibentuk oleh sejumlah besar node sensor dimana setiap node dilengkapi dengan sensor untuk mendeteksi fenomena fisik seperti cahaya, panas, tekanan, dan sebagainya. Penerapan teknologi wireless sensor network di kota Makassar akan sangat membantu dalam merealisasi konsep smart city yang direncanakan. Salah satu contoh manfaat yang dapat diperoleh dengan penerapan wireless sensor network adalah penyampaian informasi kemacetan dan banjir dapat diketahui melalui internet dengan mengolah data dari sensor suhu, kandungan polusi udara, dan ketinggian air.

Penentuan rute terpendek tidak lepas dari penggunaan Algoritma. Terdapat beberapa algoritma yang sudah digunakan dalam pembuatan suatu aplikasi atau system penentuan rute terpendek. Pada penelitian ini akan dilakukan simulasi yaitu memasukkan variable kemacetan ke dalam pencarian rute terpendek. Variabel kemacetan tersebut diperkenalkan ke dalam aplikasi pencarian rute terpendek dengan Algoritma Dijkstra dan Floyd-Warshall pada penelitian sebelumnya. Penelitian ini melakukan perhitungan dan terhadap hasil dari simulasi kedua algoritma dilakukan analisa mengenai seberapa cepat proses dari program tersebut. Penelitian ini disesuaikan dengan kondisi nyata suatu kemacetan.

2. TINJAUAN PUSTAKA

2.1 Kemacetan

Kemacetan adalah keadaan di mana kendaraan mengalami berbagai jenis kendala yang mengakibatkan turunnya kecepatan kendaraan di bawah keadaanm normal. Kemacetan akan sangat merugikan bagi para pengguna jalan, karena akan menghambat waktu perjalanan mereka.

Kemacetan lalu lintas memberikan dampak negatif seperti:

1. Pemborosan waktu, karena kendaraan tidak dapat melaju dengan kecepatan normal. Contohnya, waktu perjalanan yang seharusnya 1 jam untuk tiba di tujuan dengan kecepatan normal, menjadi 2 jam karena macet. Hal tersebut menyebabkan banyaknya waktu pengendara yang terbuang sia-sia di jalan.
2. Pemborosan energi, karena ketika macet kendaraan akan terus menggunakan bahan bakar. Hal tersebut berdampak pada pengeluaran pengendara, pengendara harus menyediakan uang ekstra untuk bahan bakar.
3. Meningkatnya polusi udara, karena pada kecepatan rendah konsumsi energi lebih tinggi dan mesin tidak beroperasi pada kondisi optimal.
4. Meningkatnya stress bagi pengguna jalan. Akibatnya, pengendaraan cenderung dalam kondisi emosional saat mengendarai kendaraan, sehingga dapat menimbulkan kecelakaan.
5. Mengganggu kelancaran kendaraan darurat seperti ambulan, pemadam kebakaran, dan sejenisnya. Akibatnya, keselamatan jiwa masyarakat yang membutuhkan pertolongan darurat menjadi terhambat.

2.2 Algoritma Dijkstra

Algoritma Dijkstra, (sesuai penemunya Edsger Dijkstra), adalah sebuah algoritma yang dipakai dalam memecahkan permasalahan jarak terpendek (shortest path problem) untuk sebuah graf berarah (directed graph) [2].

Algoritma djikstra dipublikasikan pada tahun 1959 jurnal Numerische Mathematik yang berjudul “A Note on Two Problems in Connexion with Graphs” dan dianggap

sebagai algoritma greedy. Permasalahan rute terpendek dari sebuah titik ke akhir titik lain adalah sebuah masalah klasik optimasi yang banyak digunakan untuk menguji sebuah algoritma yang diusulkan. Permasalahan rute terpendek dianggap cukup baik untuk mewakili masalah optimisasi, karena permasalahannya mudah dimengerti (hanya menjumlahkan seluruh edge yang dilalui) namun memiliki banyak pilihan solusi.

2.3 Algoritma Floyd-Warshall

Algoritma Floyd-Warshall merupakan satu diantara algoritma pencarian jalur terpendek dari sebuah graf. Algorima ini merupakan penerapan dynamic programming [3]. Dynamic programming memecah permasalahan dengan menguraikan solusi sehingga nantinya solusi dari permasalahan tersebut dapat dipandang sebagai serangkaian keputusan yang saling berkaitan dengan karakteristik bahwa solusi yang dihasilkan pada setiap tahap berasal dari solusi tahap sebelumnya [4].

Algoritma Floyd-Warshall dipilih karena keunggulan karakteristiknya dibandingkan algoritma pencarian jalur terpendek lainnya. Keunggulan algoritma Floyd-Warshall adalah proses yang lebih cepat dalam menentukan jarak terpendek dengan pencarian yang dilakukan untuk mencari jarak terpendek dari semua pasangan titik yang mungkin. Jika dibandingkan algoritma lain misalnya Dijkstra, hasil penelusuran jalur terpendek dari algoritma Floyd-Warshall lebih terjamin optimum. Hal tersebut dikarenakan algoritma Dijkstra tergolong jenis greedy yang hanya menentukan jarak terpendek dengan mengambil nilai optimum yang bersifat lokal tidak menyeluruh sehingga bisa jadi nilai akhir yang ditemukan belum tentu merupakan nilai optimum atau jarak yang terpendek [4].

Dalam pencarian jalur menggunakan algoritma Floyd-Warshall tujuan akhir yang dicari adalah menemukan jalur terpendek dari semua titik menuju semua titik atau mencari jalur terpendek semua pasangan titik yang mungkin selama masih ada jalur yang bisa menghubungkan antar titik tersebut. Sesuai dengan prinsip dynamic programming ini maka algoritma Floyd-Warshall akan mencari setiap jalur terpendek dari suatu titik ke titik lain tahap demi tahap hingga sampai semua tahap selesai maka akan diketahui jalur terpendek semua pasangan titik yang mungkin.

2.3 Route Selection

Arus lalu lintas pada suatu ruas jalan dalam suatu jaringan dapat diperkirakan sebagai hasil proses pengkombinasian informasi pemilihan rute, deskripsi sistem jaringan dan pemodelan pemilihan rute. Prosedur pemilihan rute bertujuan memodel perilaku pelaku pergerakan dalam memilih rute yang menurut mereka merupakan rute terbaiknya. Dengan kata lain, dalam proses pemilihan rute, pergerakan antara dua zona (yang didapat dari sebaran pergerakan) untuk moda tertentu (yang didapat dari tahap sebaran pergerakan) untuk moda tetentu (yang didapat dari pemilihan moda) dibebankan ke rute tertentu yang terdiri ruas jaringan tertentu (atau angkutan umum).

Tujuan dari pemilihan rute adalah mengalokasikan setiap pergerakan antarzona kepada berbagai rute yang paling sering digunakan oleh seseorang yang bergerak dari zona asal ke zona tujuan. Keluaran tahapan ini adalah informasi arus lalu lintas pada setiap ruas jalan, termasuk biaya (waktu) antar zonanya.

Terdapat beberapa faktor yang mempengaruhi dalam pemilihan rute pada saat kita melakukan perjalanan. Beberapanya adalah waktu tempuh, jarak biaya (bahan bakar dan lainnya), kemacetan dan antrian, jenis manuver yang dibutuhkan, jenis jalan raya (jalan tol, arteri), pemandangan, kelengkapan rambu lalu lintas dan marka jalan, serta kebiasaan. Sangat sukar untuk menghasilkan persamaan biaya gabungan yang menggabungkan semua faktor tersebut. Selain itu, tidaklah praktis memodel semua faktor sehingga harus digunakan beberapa asumsi atau pendekatan. Salah satu pendekatan yang paling sering digunakan adalah mempertimbangkan dua faktor utama dalam pemilihan rute, yaitu pergerakan, dan nilai waktu biaya pergerakan dianggap proporsional dengan jarak tempuh. Dalam beberapa model pemilihan rute dimungkinkan penggunaan bobot yang berbeda bagi faktor waktu tempuh dan faktor jarak tempuh untuk menggambarkan persepsi pengendara dalam kedua faktor tersebut.

Model pemilihan rute dapat diklasifikasikan berdasarkan beberapa faktor pertimbangan yang didasari pengamatan bahwa tidak setiap pengendara yang berasal dari zona asal ke zona tujuan akan memilih rute yang persis sama, khususnya di daerah perkotaan. Hal ini disebabkan oleh adanya:

a. Perbedaan persepsi pribadi tentang apa yang diartikan dengan biaya perjalanan karena adanya perbedaan kepentingan atau informasi yang tidak jelas dan tidak tepat mengenai kondisi lalu lintas pada saat itu.
b. Peningkatan biaya karena adanya kemacetan pada suatu ruas jalan yang menyebabkan kinerja beberapa rute lain menjadi lebih tinggi sehingga meningkatkan peluang untuk memilih rute tersebut. [5]

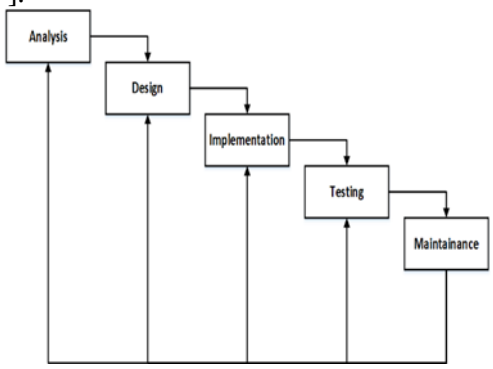
2.4 Arduino

Arduino adalah platform pembuatan prototype elektronik yang bersifat open-source hardware yang berdasarkan pada perangkat keras dan perangkat lunak yang fleksibel dan mudah digunakan. Arduino ditujukan bagi para seniman, desainer, dan siapapun yang tertarik dalam menciptakan objek atau lingkungan yang interaktif. Arduino pada awalnya dikembangkan di Ivrea, Italia. Nama Arduino adalah sebuah nama maskulin yang berarti teman yang kuat. Platform arduino terdiri dari arduino board, shield, bahasa pemrograman arduino, dan arduino development environment. Arduino board biasanya memiliki sebuah chip dasar mikrokontroler Atmel AVR ATmega8 berikut turunannya. Blok diagram arduino board yang sudah disederhanakan. Shield adalah sebuah papan yang dapat dipasang diatas arduino board untuk menambah kemampuan dari arduino board. Bahasa pemrograman arduino adalah bahasa pemrograman yang umum digunakan untuk membuat perangkat lunak yang ditanamkan pada arduino board. Bahasa pemrograman arduino mirip dengan bahasa pemrograman C++. Arduino Development Environment adalah perangkat lunak yang digunakan untuk menulis dan meng-compile program untuk arduino. Arduino Development Environment juga digunakan untuk meng-upload program yang sudah di-compile ke memori program arduino board [6].

3. METODOLOGI PENELITIAN

3.1 Metode Perancangan

Pendekatan Waterfall merupakan suatu proses pengembangan software yang bersifat sekuensial atau berurutan, dimana setiap tahapan yang dikerjakan secara berurut menurun (seperti air terjun) dari perencanaan, analisa, desain, implementasi, dan pengujian [7].



Gambar 1. Metode Waterfall

1. Analysis
   Pada tahap ini penulis melakukan analisa terhadap terhadap permasalahan yang terjadi untuk selanjutnya dilakukan identifikasi kebutuhan-kebutuhan dari sudut pandang objek diperoleh sehingga dari permasalahan tersebut akan mendapatkan gambaran tentang proses penginformasian keadaan lalu lintas secara realtime dan pembuatan interface yang akan digunakan.
2. Design
   Pada tahap ini penulis merancang desain interface berdasarkan analisis kebutuhan yang telah dilakukan agar dapat menjadi bentuk yang mudah dimengerti dan menarik digunakan bagi pihak yang terlibat menggunakan system, serta tahap desain ini juga akan merancang diagram berjenjang, diagram konteks, dan DFD.
3. Coding
   Pada tahap ini penulis melakukan pembuatan interface program dan merangkai sensor-sensor pada mikrokontroler arduino yang akan digunakan untuk membuat dibuat ini.
4. Testing
   Pada tahap ini penulis melakukan simulasi terhadap aplikasi yang dikembangkan serta memastikan tidak

terdapat kesalahan-kesalahan pada aplikasi yang akan diimplementasikan ini.

5. Maintenance
   Tahap ini merupakan tahap terakhir dalam metode waterfall. Pemeliharaan mencakup koreksi dari berbagai error yang tidak ditemukan pada tahap-tahap terdahulu, perbaikan atas implementasi dan pengembangan unit sistem, serta pemeliharaan program. Pemeliharaan sistem dapat dilakukan oleh seorang administrator untuk meningkatkan kualitas sistem agar jauh lebik baik.

3.2 Metode Pengumpulan Data

Adapun metode pengumpulan data yang akan digunakan dalam penelitian ini adalah sebagai berikut:

1. Studi Literatur
   Studi literatur merupakan metode pengumpulan data dengan mencari dan mempelajari data-data yang berkaitan dengan masalah maupun metode dalam penelitian ini. Serta penulis mencari berbagai sumber referensi baik dari buku maupun jurnal ilmiah dalam pengembangan penelitian.
2. Cara Pengambilan Data
   Pada metode ini, penulis melakukan pencarian data bobot langsung pada google maps untuk menentukan bobot dari setiap jalan agar dapat dihitung dengan algoritma nantinya. Pengambilan data juga dilakukan dengan langsung turun ke jalan mengambil data sensor. Alat yang digunakan untuk pengambilan data di jalan sama dengan alat yang digunakan dalam simulasi. Hal tersebut dilakukan untuk menciptakan suatu simulasi yang sesuai dengan keadaan nyata di jalan.

3.3 Analisa Data

Analisis data yang dilakukan dalam penelitian ini yaitu analisis data kuantitatif. Analisis data kuantitatif akan diolah dari hasil pengumpulan data langsung dari google maps tentang bobot rute dan bagaimana penginformasian keadaan lalu lintas kepada pengendara.

4. HASIL DAN PEMBAHASAN

4.1 Analisa Kebutuhan

Sistem penemuan rute terdekat ini membutuhkan data berupa bobot dari setiap jalur yang akan dilakukan perhitungan algoritma dijkstra. Karena dalam sistem ini terdapat beberapa variabel maka untuk menghitung semuanya diperlukan beberapa kondisi antara lain kemacetan, yang mana sensor pada arduino akan menangkap data gas yang berada pada lokasi kemacetan. Sistem ini juga membutuhkan keadaan dimana banjir dimana sensor akan menangkap level ketinggian air. Sistem ini membutuhkan data latitude-longitude dari tiap perempatan yang akan digunakan sebagai simpul yang kemudian akan dihitung ke dalam dijkstra. Adapula rute yang akan terbuat pada hasilnya membutuhkan data latitude-longitude jalan.

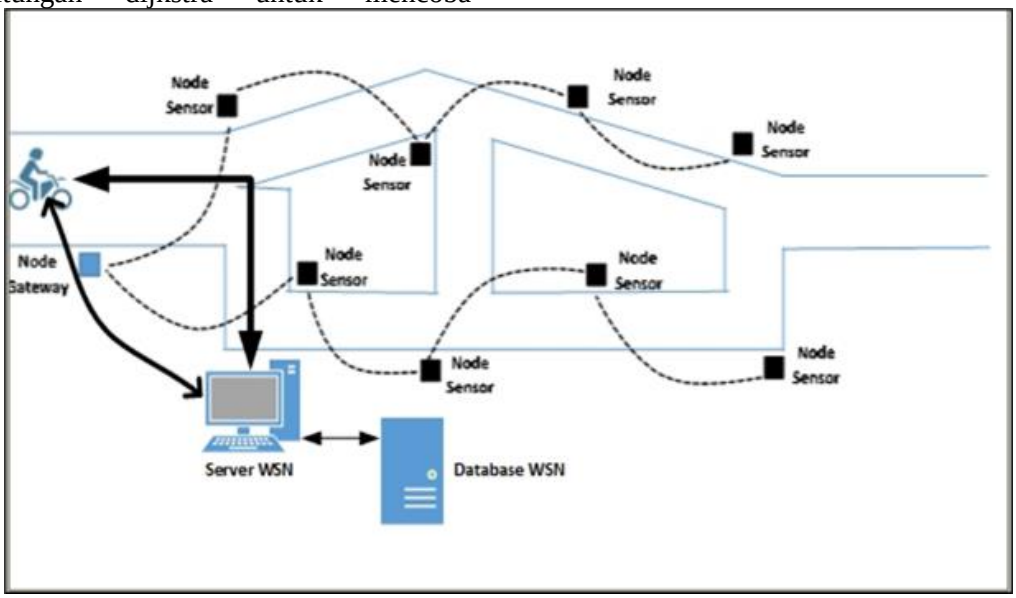
Program ini berjalan dengan mengasumsikan data sensor memperoleh data sumber polusi hanya dari kendaraan dan tidak ada sumber asap lain yang misalnya dari pembakaran sampah, polusi pabrik, dan polusi lainnya. Penerapan sensor ini yaitu di pinggir jalan pada beberapa ruas jalan yang terhadap titik tersebut akan dilakukan pengambilan data sensor.

Kondisi penerapan yang diharapkan adalah saat tidak ada faktor lain yang mempengaruhi pembacaan sensor seperti asap pembakaran terhadap sensor gas serta saat sensor ketinggian air tersiram. Penerapan program ini adalah sensor-sensor dipasang di pinggir jalan pada beberapa ruas jalan. Setelah data terbaca maka akan dilakukan perhitungan dengan algoritma Dijkstra dan Floyd-Warshall. Proses perhitungan tersebut akan mendapatkan jalur serta kecepatan pembacaan yang pada akhirnya akan dilakukan analisa dari kecepatan jalur dan kecepatan proses perhitungan. Sistem ini menggunakan 2 Algoritma adalah untuk membandingkan algoritma dari proses perhitungan dan ketepatan jalur.

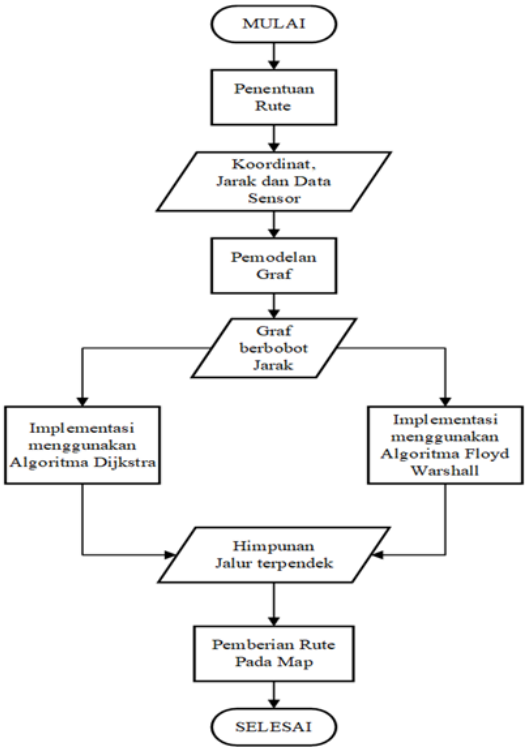
4.2 Desain Sistem

Desain sistem ini adalah pada awalnya sistem mengambil data dari map. Data yang diambil adalah data latitude-longitude dari setiap simpul yang akan dihitung pada dijkstra. Kemudian mengambil data latitude-

longitude garis atau edge dan menentukan id setiap simpul sebagai identitas dari setiap simpul. Setelah data simpul ditemukan. Tahap berikutnya yaitu mengambil nilai atau bobot dari setiap edge yang menghubungkan satu simpul ke simpul tetangga. Bila data dari maps sudah diperoleh bisa dilakukan perhitungan dijkstra untuk mencoba bilamana program sudah bisa menentukan rute terpendek. Tahap selanjutnya adalah sensor mengirim data ke database dan data tesebut akan diakses oleh perhitungan algoritma dijkstra sehingga akan menambah variabel dari keakuratan dikstra tersebut mentukan jalur tesingkat.



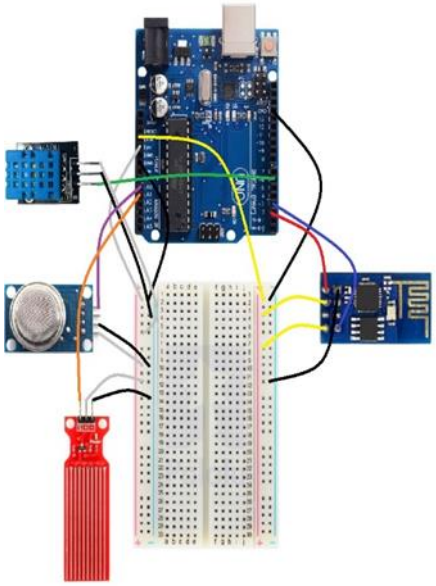
Gambar2. Desain Sistem



Gambar 3. Flowchart Skema Sistem

4.2.1 Perancangan Sirkuit dan Simulasi

Perancangan sirkuit dilakukan untuk mendapatkan data melalui Arduino Uno sebagai mikrokontroler serta menggunakan beberapa sensor yang kemudian akan mengirimkan data dari sensor-sensor secara nirkabel. Sensor yang digunakan antara lain yaitu sensor suhu dan kelembapan DHT11, sensor gas MQ-2, dan sensor ketinggian air Water Level.



Gambar 4. Rangkaian Arduino

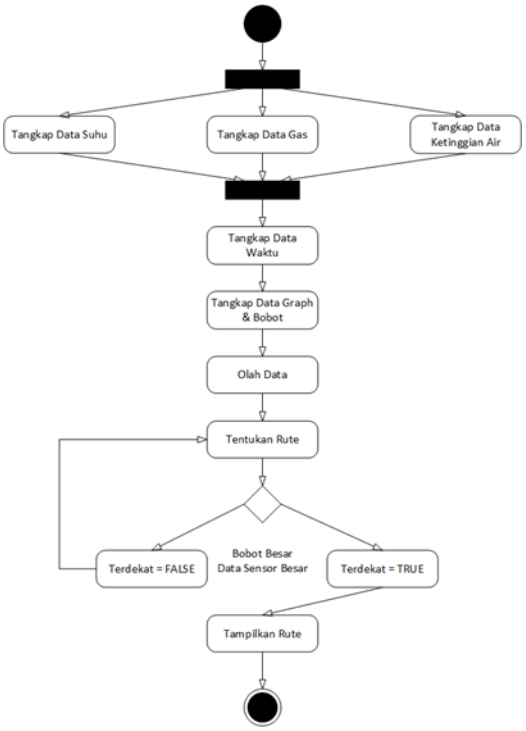
4.2.2 Use Case

Pada aplikasi ini hanya terdapat satu jenis pengguna yaitu pengguna yang akan memilih akan pergi kemana maka digunakan usecase diagram. Diagram ini menggambarkan secara garis besar mengenai sistem yang dirancang oleh penulis. Pada awalnya Pengguna akan memasukkan simpul awal setelah itu memasukkan simpul akhir atau tujuan dari rute yang akan ditempuh. Alat Sensor akan menangkap nilai dari sensor-sensor pada Arduino. Sensor-sensor tersebut antara lain adalah sensor Kadar Gas CO2, Suhu, dan Ketinggian Air. Setelah semua data ditangkap, data-data tersebut akan disimpan ke dalam database. Data dari sensor akan diolah dengan data yang berasal dari bobot setiap jalur untuk kemudian diolah untuk menentukan rute terdekat yang akan ditempuh dan menampilkan rute tersebut ke Pengguna.

Gambar 5. Use Case

4.2.3 Activity Diagram

Diagram ini menggambarkan peristiwa atau proses dari sistem yang sedang berjalan. Activity Diagram Tampilan Rute: Arduino pertama-tama akan menangkap data dari sensor-sensor. Sensor-sensor yang akan diambil antara lain adalah sensor suhu, kadar gas CO2, dan Ketinggian Air. Semua data itu akan disimpan ke dalam database. Setelah semua data tersimpan maka data dari sensor-sensor itu akan dikalkulasikan dengan bobot jarak dari tiap graph untuk dicarikan jarak terdekatnya. Kemudian setelah ditemukan jarak terdekatnya dari simpul awal ke simpul akhir, maka akan menampilkan rute terdekat.



Gambar 6. Activity Diagram

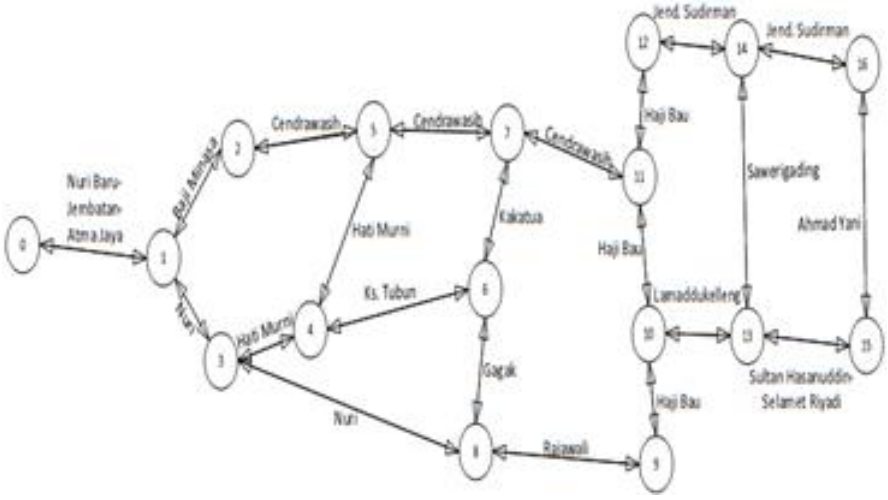
4.3 Implementasi

Aplikasi navigasi kendaraan bermotor di kota makassar yang dibangun dengan mengimplementasikan konsep graf untuk menggambarkan sistem. Kemudian dalam langkah mencari jalur terpendek menggunakan algoritma Floyd-Warshall dan Dijkstra. Sistem pemilihan rute ini mempunyai satu tampilan yang mana akan menampilkan pemilihan titik akhir atau tujuan dan maps yang mana akan menampilkan maps yang bila diklik akan menjadi titik awal atau keberangkatan dari proses dijkstra yang akan dilakukan. Setelah itu akan dilakukan perhitungan dari titik awal keberangkatan bobot jalurnya akan dibandingkan menuju ke titik terdekat hingga semua titik terkunjungi dan menghasilkan urutan simpul yang menjadi hasil akhir dari sistem ini.

4.3.1 Penentuan Graf Berarah dan Berbobot

Peta pada Gambar dapat diinterpretasikan dalam graf berarah dan berbobot, dengan titik menunjukkan persimpangan jalan dan sisi menunjukkan ruas jalan yang menghubungkan antar persimpangan tersebut. Sisi ganda dan berarah pada graf berikut menunjukkan bahwa terdapat beberapa ruas jalan yang

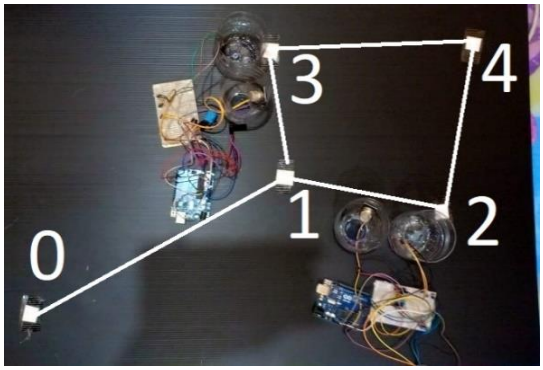
hanya memiliki satu arah ruas jalan (verboden). Bobot sisi dari graf tersebut menginterpretasikan volume kendaraan yang ada di ruas jalan tersebut. Berikut ini adalah graf berarah yang menginterpretasikan peta pada gambar sebagai berikut.



Gambar 7. Graf Berarah

4.3.2 Implementasi Algoritma Dijkstra

Pengujian ini dilakukan untuk melihat pengaruh perubahan bobot variable sensor terhadap perhitungan Dijkstra. Awalnya simulasi yang akan dilakukan saat keadaan sensor normal setelah itu sensor akan distimulasi untuk memberikan nilai pada sensor tersebut.



Gambar 8. Model Simulasi Dijkstra

Di bawah merupakan bobot(jarak) dari tiap tiap titik;

Tabel 1. Bobot Simulasi

NO	Titik Awal	Titik Tujuan	Bobot (Jarak)
1	0	1	960
2	1	2	450
3	1	3	186
4	2	4	234
5	3	4	492

Terdapat dua jalur dari titik 0 untuk menuju ke titik 4 antara lain:

Tabel 2. Bobot Simulasi Dijkstra

NO	RUTE	JARAK
1	0 – 1 – 2 – 4	1644
2	0 – 1 – 3 – 4	1636

Tabel 3. Data Arduino pada Titik 2 Dijkstra Untuk Arduino di titik 3

No	Polusi (ppm)	Suhu (C°)	Jml	Ket.
1	139	30	169	Lancar
2	141	30	171	Lancar
3	137	30	167	Lancar

Tabel 4. Data Arduino pada Titik 3 Dijkstra

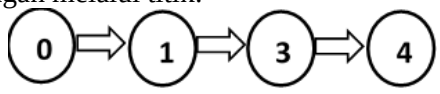
No	Polusi (ppm)	Suhu (C°)	Jml	Ket.
1	141	31	172	Lancar
2	145	31	176	Lancar
3	139	31	170	Lancar

Dengan hasil data keadaan Lancar bila dijumlahkan dengan hasil dari sensor maka dipilih jalur dengan total paling sedikit.

Tabel 5. Penambahan Bobot Sensor Simulasi

Nu	Rute	Bobot Jarak	Bobot Sensor	Total
1	0 – 1 – 2 – 4	1644	167	1811
2	0 – 1 – 3 – 4	1636	170	1806

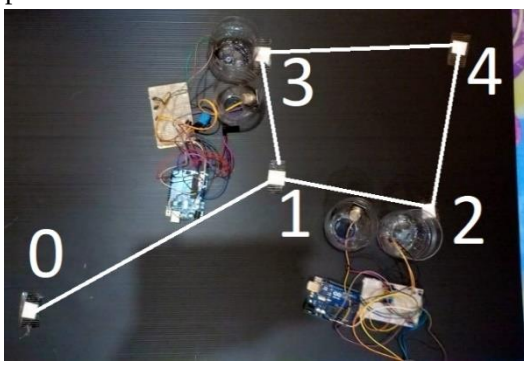
Maka rute terdekat dari algoritma ini adalah dengan melalui titik:



Gambar 9. Rute Terdekat Dijkstra

4.3.3 Implementasi Algoritma Floyd-Warshall

Pengujian ini dilakukan untuk melihat pengaruh perubahan bobot variable sensor terhadap perhitungan Floyd Warshall. Awalnya simulasi yang akan dilakukan saat keadaan sensor normal setelah itu sensor akan distimulasi untuk memberikan nilai pada sensor tersebut.



Gambar 10. Model Simulasi Floyd-Warshall

Tabel 6. Bobot Simulasi

NO	Titik Awal	Titik Tujuan	Bobot (Jarak)
1	0	1	960
2	1	2	450
3	1	3	186
4	2	4	234
5	3	4	492

Terdapat dua jalur dari titik 0 untuk menuju ke titik 4 antara lain:

Tabel 7. Perbandingan Jarak

NO	RUTE	JARAK
1	0 – 1 – 2 – 4	1644
2	0 – 1 – 3 – 4	1636

Untuk Arduino di titik 2

Tabel 8. Data Arduino pada Titik 2

No	Polusi (ppm)	Suhu (C°)	Jml	Ket.
1	139	30	169	Lancar
2	141	30	171	Lancar
3	137	30	167	Lancar

Untuk Arduino di titik 3

Tabel 9. Data Arduino pada Titik 3

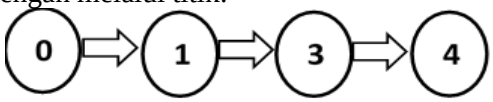
No	Polusi (ppm)	Suhu (C°)	Jml	Ket.
1	141	31	172	Lancar
2	145	31	176	Lancar
3	139	31	170	Lancar

Dengan hasil data keadaan lancar bila dijumlahkan dengan hasil dari sensor adalah sebagai berikut

Tabel 10. Penambahan Bobot Sensor Simulasi

No	Rute	Bobot Jarak	Bobot Sensor	Total
1	0 – 1 – 2 – 4	1644	167	1811
2	0 – 1 – 3 – 4	1636	170	1806

Maka rute terdekat dari algoritma ini adalah dengan melalui titik:



Gambar 11. Rute Terdekat Simulasi Pertama

4.4 Pemeliharaan

4.4.1 Arduino Uno

Arduino uno dalam penelitian ini berfungsi sebagai penampung semua nilai dari masing-masing sensor. Arduino pula yang menyalurkan tegangan pada masing-masing sensor dan modul wifi, dan pengkodean untuk melakukan pengiriman data sensor tersebut diupload pada Arduino uno untuk itu peran Arduino uno menjadi sangat vital pada penelitian ini. Jika Arduino uno tidak bekerja maka tidak ada data sensor yang terkirim. Maka dibutuhkan pemeliharaan yang baik kedepannya. Mulai dari memberikan rumah kepada Arduino agar terhindar dari faktor-faktor alam yang akan sangat mengganggu layaknya panas dan hujan yang pastinya akan menyebabkan kerusakan pada Arduino. Selain itu pada Arduino ini sebaiknya dilakukan pengecekan kondisi setiap sedikitnya 3 minggu satu kali. Hal ini dilakukan untuk terus menjaga agar Arduino bisa secara stabil mengirimkan data sensor. Bila terjadi kerusakan pada Arduino sebaiknya disiapkan Arduino pengganti agar pembacaan sensor bisa tetap berjalan.

4.4.2 ESP8266

ESP8266 dalam penelitian ini berfungsi sebagai pengirim data sensor pada server secara nirkabel. Jika modul ESP8266 tidak bekerja maka tidak ada data sensor yang terkirim. Maka dibutuhkan perawatan yang baik kedepannya. Sama seperti pada Arduino uno, modul ini juga harus dibuatkan tempat khusus yang bisa menghindarkan faktor-faktor alam seperti panas dan hujan. Selain

itu pada ESP8266 ini sebaiknya dilakukan pengecekan kondisi setiap sedikitnya 3 minggu satu kali. Hal ini dilakukan untuk terus menjaga agar ESP8266 bisa secara stabil mengirimkan data sensor. Bila terjadi kerusakan pada ESP8266 sebaiknya disiapkan ESP8266 pengganti dan langsung dilakukan pengupload-an kode ke Arduino uno agar pembacaan sensor bisa tetap berjalan.

4.4.3 DHT11

DHT11 dalam penelitian ini berfungsi sebagai sensor Suhu. Jika DHT11 tidak bekerja maka tidak ada data yang suhu yang diterima oleh Arduino dan akan dikirim pada server. Sensor ini harus berikan pengaman yang baik agar tidak terdampak langsung jika terjadi hujan karena bisa menyebab kerusakan pada sensor tersebut. Selain itu pada sensor DHT11 ini sebaiknya juga dilakukan pengecekan kondisi setiap sedikitnya 3 minggu satu kali. Hal ini dilakukan untuk terus menjaga ke stabilan sensor dalam pengiriman data suhu. Bila terjadi kerusakan pada sensor DHT11 maka sebaiknya langsung digantikan agar data yang terkirim bisa tetap berjalan.

4.4.4 MQ2

MQ2 dalam penelitian ini berfungsi sebagai sensor Polusi. Jika sensor MQ2 tidak bekerja maka tidak ada data yang kadar gas yang diterima oleh Arduino dan akan dikirim pada server. Sensor ini harus diberikan pengamanan yang baik agar tidak terdampak langsung jika terjadi hujan karena bisa menyebabkan kerusakan pada sensor tersebut. Selain itu pada sensor MQ2 ini sebaiknya juga dilakukan pengecekan kondisi setiap sedikitnya 3 minggu satu kali. Hal ini dilakukan untuk terus menjaga kestabilan sensor dalam pengiriman data kadar gas. Bila terjadi kerusakan pada sensor MQ2 maka sebaiknya langsung digantikan agar data yang terkirim bisa tetap berjalan.

4.4.5 Sensor Water Level

Sensor Water Level dalam penelitian ini berfungsi sebagai sensor ketinggian air. Jika Sensor Water Level tidak bekerja maka tidak ada data ketinggian air yang diterima oleh Arduino dan akan dikirimkan kepada server. Sensor ini harus diberikan pengamanan yang baik. Agar tidak terdampak langsung jika terjadi hujan karena bisa menyebabkan kerusakan bila terjadi korsleting pada tegangan. Selain itu sebaiknya dilakukan pengecekan kondisi setiap sedikitnya 3 minggu satu kali. Hal ini dilakukan untuk terus menjaga kestabilan sensor dalam pengiriman data ketinggian air. Bila terjadi kerusakan pada sensor water level maka sebaiknya langsung digantikan agar data yang terkirim bisa tetap berjalan.

4.4.6 Algoritma

Algoritma adalah inti dari penentuan rute-rute dalam sistem ini. Pada algoritma ini harus juga dilakukan pemeliharaan. Pemeliharaan yang bisa dilakukan adalah dengan cara menguji algoritma. Bila terjadi bug pada code maka sebaiknya dilakukan analisa terhadap algoritma yang bermasalah. Selain itu harus tetap dilakukan pengecekan terhadap algoritma tersebut setiap 6 minggu satu kali. Pengecekan yang dilakukan adalah dengan melakukan Analisa terhadap hasil rute yang didapatkan.

4.4.7 Database

Database titik dalam hal ini sangat penting karena algoritma berjalan terhadap database titik. Untuk itu Database ini harus tetap dilakukan penyesuaian. Penyesuaian yang dilakukan dalam hal ini adalah database harus disesuaikan dengan jalur pada keadaan nyata. Untuk waktunya sendiri adalah suatu hal yang tidak menentu, karena perubahan jalur yang terjadi di Jalan Raya pada keadaan nyata adalah suatu hal yang tidak menentu karena disesuaikan oleh keadaan lalu-lintas kota serta bergantung pada keputusan pemerintah daerah tentang tata kota.

5. KESIMPULAN

Berdasarkan dari hasil penelitian yang telah dilakukan, maka ditarik kesimpulan antara lain:

1. Dari hasil analisa dua algoritma ditemukan bahwa kedua algoritma menghasilkan rute yang sama tetapi jika dibandingkan dari kecepatan proses maka ditemukan bahwa algoritma dijkstra lebih cepat.

2. Dari hasil simulasi bisa dilihat bahwa perhubahan sensor atau kesenjangan bobot data pada sensor antartitik akan sangat mempengaruhi penentuan rute.

6. DAFTAR PUSTAKA

[1] Kementerian Perhubungan Republik Indonesia. 2015. *Benahi Transportasi Kota Makassar, harus ada Langkah Ekstrim*. [Online] Tersedia pada http://dephub.go.id/berita/baca/benahi-transportasi-kota-makassar,--harus-ada-langkah-ekstrim [Diakses pada 16 Januari 2018].

[2] Girsang, Abba Suganda. 2017. *Algoritma Djikstra*. [Online] Tersedia pada https://mti.binus.ac.id/2017/11/28/algoritma-dijkstra/ [Diakses pada 9 Februari 2018].

[3] Handaka, M.S. 2010. *Perbandingan Algoritma Dijkstra (Greedy), Bellman-Ford (BFS-DFS), dan Floyd-Warshall (Dynamic Programming) dalam Pengaplikasian Lintasan Terpendek pada Link-State Routing Protocol*. Sekolah Teknik Elektro dan Informatika Institut Teknologi Bandung, Bandung.

[4] Munir, R. 2005. *Diktat Kuliah IF2251 Strategi Algoritmik*. Program Studi Informatika Sekolah Teknik Elektro dan Informatika ITB, Bandung.

[5] Fadhilla, G., Jupri., Somantri, L. 2018. *Evaluasi Rute Transportasi Angkutan Kota dengan Menggunakan Sistem Informasi Geografis*. Departemen Pendidikan Geografi, Universitas Pendidikan Indonesia, Jakarta.

[6] Simanjuntak, M. G., 2013. *Perancangan Prototipe Smart Building berbasis Arduino UNO*. Teknik Elektro Universitas Sumatera Utara, Medan.

[7] Pressman, Roger S. 2012. *Rekayasa Perangkat Lunak – Buku Satu, Pendekatan Praktisi (Edisi 7)*. Penerbit Andi, Yogyakarta.

[8] Lisangan, E. A. 2017. *Route Selection based on Real Time Traffic Condition using Ant Colony System and Fuzzy Inference System*, In Internatioanl Conference on Science in Information Technology, 2017.