ANALISA SISTEM PENCARIAN JALUR PADA APLIKASI PANGGILAN DARURAT MENGGUNAKAN ALGORITMA A* (A STAR) DAN PRIM

ROUTING ANALYSIS IN THE EMERFENCY CALL APPLICATION USING A (A STAR) AND PRIM ALGORITHM*

Lalu Muh. Andre Winarta[1], Purba Daru Kusuma[2], Casi Setianingsih[3]
[1,2,3] Universitas Telkom, Bandung
andrewinarta@student.telkomuniversity.ac.id[1], purbodaru@telkomuniveristy.ac.id[2], setiacasie@telkomuniversity.ac.id[3]

Abstrak

Panggilan darurat di Indonesia masih menerapkan teknologi panggilan satelit untuk memanggil instansi darurat. Indonesia sudah memiliki nomor darurat universal, namun implementasinya masih belum merata di semua kota/kabupaten, dimana nomor darurat 112. Dengan menggunakan aplikasi berbasis internet, penulis dapat membangun sebuah aplikasi panggilan yang dipadukan dengan Google Maps untuk memberikan tampilan gambar dari peta digital dan memberikan rute otomatis dari lokasi pemanggil ke lokasi instansi yang dipanggil secara langsung.

Berdasarkan hasil pengujian, algoritma A* dan Prim dapat diimplementasikan dengan baik dengan menggunakan bobot SAW 30% jarak dan 70% durasi. Dalam pengujian, penulis menggunakan Telkom University sebagai lokasi awal dan Rumah Sakit Umum Bina Sehat sebagai tujuan. Jarak yang ditempuh menggunakan algoritma A* adalah 3,14 kilometer dan waktu tempuh 11,7 menit dan untuk algoritma Prim sendiri jaraknya 2,56 kilometer dan lama perjalanan 6,75 menit. Untuk rata-rata waktu respon aplikasi menggunakan algoritma A* adalah 11,76 detik dan algoritma Prim adalah 15,07 detik. Dapat disimpulkan bahwa algoritma Prim dapat memberikan rute yang lebih baik daripada algoritma A*, namun untuk waktu respon aplikasi, algoritma A* dapat dikatakan lebih baik karena memberikan hasil yang lebih singkat dibandingkan dengan algoritma Prim.

Kata kunci : Rute Terbaik, Algoritma A Star, Algoritma Prim, Google Maps

Abstract

Emergency calls in Indonesia are still implementing satellite call technology to call emergency agencies. Indonesia already has a universal emergency number, but its implementation is still not evenly distributed in all cities/districts, where the emergency number is 112. By using an internet-based application, the author can build a calling application that is combined with Google Maps to provide an image display from a digital map and provide an automatic route from the caller's location to the location of the agency that is called directly.

Based on the test results, the A* and Prim algorithms can be implemented properly by using SAW weights of 30% distance and 70% duration. In testing, the authors use Telkom University as the initial location and the Rumah Sakit Umum Bina Sehat as the destination. The distance traveled using the A* algorithm is 3,14 kilometers and the travel time is 11,7 minutes and for Prim's own algorithm, the distance is 2,56 kilometers and the travel duration is 6,75 minutes. The average application response time using the A* algorithm is 11,76 seconds and Prim's algorithm is 15,07 seconds. It can be concluded that Prim's algorithm can provide a better route than the A* algorithm, but for the application response time, the A* algorithm can be said to be better because it gives shorter results compared to Prim's algorithm.

Keywords: Best Route, A Star Algorithm, Prim Algorithm, Google Maps

1. Pendahuluan

Telepon merupakan alat komunikasi jarak jauh yang menggunakan suara. Salah satu implementasi penggunaan telepon adalah panggilan darurat ke instansi (polisi, pemadam kebakaran, ambulans, dan lain-lain). Handphone merupakan perangkat genggam yang dimiliki oleh sebagian besar masyarakat di dunia untuk keperluan komunikasi, pekerjaan, dan lain-lain. Perkembangan telepon seluler tidak lepas dari perkembangan teknologi yang ada, termasuk internet. Internet adalah singkatan dari Interconnected Network, dimana Internet adalah sistem jaringan komputer yang memungkinkan kita untuk terhubung dengan orang-orang di seluruh dunia menggunakan satu perangkat.

Menurut laporan terbaru We Are Social, pada tahun 2020 disebutkan terdapat 175,4 juta pengguna internet di Indonesia, dengan peningkatan pengguna di Indonesia sebesar 17% dari tahun sebelumnya[1]. Internet banyak digunakan untuk kebutuhan sehari-hari, seperti kemudahan mengakses informasi, konektivitas, komunikasi dan

sharing, dan masih banyak lagi aktivitas yang lebih mudah dilakukan dengan internet.

Saat ini panggilan darurat di Indonesia masih menggunakan panggilan konvensional, padahal saat ini masyarakat umum lebih banyak menggunakan internet untuk segala kebutuhannya. Untuk melakukan panggilan darurat, orang-orang harus mengetahui nomor agensi yang akan dihubungi. Dalam hal ini, masyarakat jarang mengetahui informasi tentang nomor darurat. Selanjutnya penelepon harus menjelaskan lokasi terjadinya keadaan darurat yang dapat memakan waktu lama jika agen yang dihubungi tidak mengetahui alamat yang tertera oleh penelepon. Dalam menerima panggilan, setelah instansi menerima panggilan darurat, biasanya informasi dari panggilan darurat ini akan disampaikan kepada petugas yang bertugas melalui radio atau orang lain untuk menuju ke tempat penelepon [2].

Makalah ini bertujuan untuk mengembangkan aplikasi mobile untuk aplikasi panggilan darurat. Aplikasi ini berfokus pada pencarian jalur terpendek menggunakan algoritma A* dan algoritma Prim. Selain itu, kinerja kedua algoritma dibandingkan dalam konteks pencarian rute terbaik dari lokasi pengguna ke instansi terkait menggunakan parameter nilai algoritma dan durasi waktu tempuh.

2. Dasar Teori

2.1 Rute Terpendek

Rute terpendek adalah jalur yang diperlukan untuk mencapai tujuan dari lokasi saat ini ke tujuan dengan menghitung biaya minimum (jarak). Masalah seperti ini biasanya disajikan dalam bentuk graf, dimana state yang berhubungan dengan ruang lingkup pencarian direpresentasikan dengan vertex dan transisi yang terjadi digambarkan dalam bentuk edge [3]. Graf itu sendiri merupakan struktur diskrit yang terdiri dari simpul (V) yang merupakan himpunan tak kosong dan sisi (E) adalah himpunan sisi yang menghubungkan simpul-simpul tersebut [4].

Kata terpendek dari Shortest Route dapat diartikan sebagai bagaimana proses meminimalkan biaya (bobot) yang digunakan dalam suatu jalur graf. Beberapa masalah jalur terpendek yang biasa digunakan adalah jalur terpendek antara dua node, jalur terpendek antara semua pasangan node, jalur terpendek dari node tertentu ke node lain, jalur terpendek dari dua node yang melewati node tertentu [5].

2.2 Algoritma A Star

Algoritma A* pertama kali diperkenalkan oleh Peter Hart, Nils Nilsson, dan Bertram Raphael pada tahun 1968 menggunakan heuristik [6]. Algoritma A* merupakan salah satu algoritma pencarian rute yang optimal dan lengkap, dimana algoritma ini dapat diterapkan dalam konfigurasi metrik atau topologi [7]. Optimal dalam artian rute yang dihasilkan merupakan rute terbaik dan terlengkap, artinya algoritma dapat mencapai tujuan yang diharapkan [8]. Algoritma ini merupakan gabungan dari algoritma Dijkstra dan algoritma Greedy Best First Search.

Algoritma A* menerapkan konsep algoritma pencarian heuristik, dimana heuristik merupakan fungsi untuk menghitung jarak dari node awal ke node tujuan. Algoritma ini memiliki banyak variasi dengan menerapkan fungsi heuristik yang berbeda sesuai dengan penggunaannya [9]. Tujuan dari algoritma ini adalah untuk mencari rute optimum dalam sebuah graf dengan menjumlahkan biaya heuristik (jarak antara node awal ke node tujuan) dan nilai node (jarak antara node n dan n+1) . Algoritma ini dapat ditulis dalam Persamaan berikut:

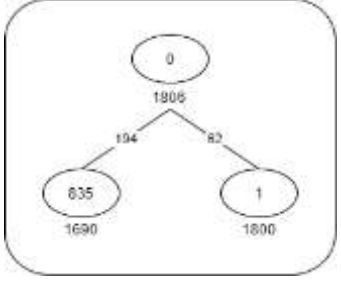
$$f(n)=g(n)+h(n) \qquad (1)$$

Keterangan:
f(n) = total *cost*
g(n) = *cost* yang telah dicapai saat *node* ke-n
h(n) = *cost* estimasi dari *node* ke-n ke tujuan

Adapun untuk langkah-langkah dari algoritma A* untuk mencari *cost* atau bobot minimum sebagai berikut:

1. Tandai *node* awal sebagai *current node* dan hubungkan *node* awal dengan *node* selanjutnya yang belum dilalui atau *node* berikutnya.
2. Hitung nilai fungsi menggunakan persamaan (2.1) untuk masing-masing *node* berikutnya, lalu tandai *node* dengan nilai persamaan paling minimum dan selanjutnya akan dijadikan sebagai *current node*.
3. Lakukan langkah diatas sampai nilai *current node* merupakan *node* tujuan.

Untuk contoh penerapan algoritma A Star, dapat dilihat dibawah ini.



Gambar 1. Connected Node 0 A Star

Hubungkan *current node* dengan *node* yang terhubung dan hitung jaraknya dengan Persamaan. 1 dan ambil nilai minimumnya. Hasil *node* 1 adalah 1862 dan *node* 835 adalah 1884. Karena *node* 835 adalah nilai minimum atau hasil terendah, maka *node* 835 menjadi node berikutnya.

Langkah selanjutnya, *node* 835 menjadi *current node* dan ulangi langkah di atas sampai *current node* adalah *node* tujuan.

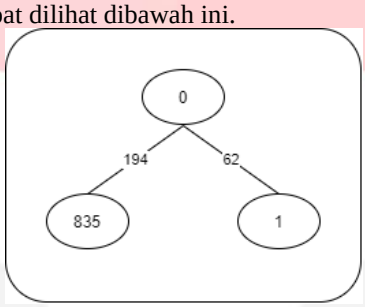
2.3 Algoritma Prim

Algoritma Prim merupakan algoritma teori graf yang mencari *Minimum Spanning Tree* dalam sebuah graf berbobot yang saling terhubung [10]. Cara kerja algoritma ini menggunakan konsep greedy, dimana jika terdapat alternatif *node* atau *node* yang tidak dipilih, maka node yang dipilih merupakan node terpendek dari awal. Algoritma ini tidak mengizinkan iterasi berulang. Untuk mengetahui apakah node sebelumnya telah dilewati, tandai *node* yang dilewati dan masukan hanya *node* yang belum ditandai sebagai prioritas [11].

Adapun untuk langkah-langkah dari algoritma Prim untuk mencari bobot minimum sebagai berikut:

1. Ambil sisi dari *graph* (G) yang berbobot minimum, lalu masukkan ke dalam *tree* (T).
2. Pilih *node* yang mempunyai *cost* minimum dengan simpul di T, tetapi simpul tersebut tidak membentuk sirkuit pada T, kemudian tambahkan sisi tersebut ke dalam T.
3. Ulangi langkah kedua sebanyak N-2 kali.

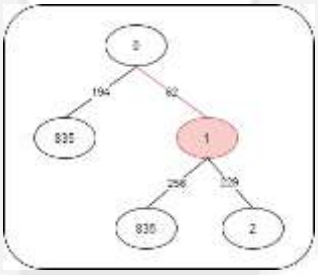
Contoh penerapan algoritma prim dapat dilihat dibawah ini.



Gambar 2. Connected Node 0 Prim

Hubungkan node saat ini dengan node yang terhubung dan pilih node dengan jarak minimum atau terendah. Jarak antara node 0 dan 1 adalah 62 dan node 835 adalah 194. Karena node 1 memiliki hasil yang minimum, maka node 1 akan menjadi node berikutnya.

Setelah itu, tambahkan node 1 sebagai node saat ini tanpa menghapus node 0 sebagai node saat ini. Periksa jarak node saat ini ke node yang terhubung dan pilih nilai minimum (node berikutnya sebelumnya tidak memeriksa).



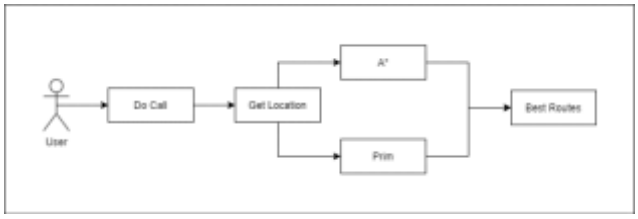
Gambar 3. Connected Node 0 and 1 Prim

Karena node 0 dan 835 memiliki hasil minimum dengan jarak 194, *node* 835 menjadi *node* berikutnya dan menambahkannya sebagai *node* saat ini. Ulangi langkah di atas sampai *node* tujuan adalah *current node*.

3. Pembahasan

3.1. Gambaran Umum Sistem

Gambaran sistem yang akan di desain atau dirancang pada aplikasi pemanggilan darurat ini adalah pencarian rute terpendek dengan menggunakan algoritma A* dan Prim, lalu membandingkan kedua algoritm tersebut dan menentukan mana algoritma terbaik untuk mencari rute terpendek dengan memperhatikan jarak dan waktu.



Gambar 4. Gambaran Umum Aplikasi

Untuk menggunakan aplikasi, berikut penjelasan langkah-langkah penggunaannya:

1. Pengguna registrasi pada aplikasi dengan memasukan data diri yang diperlukan.
2. Setelah registrasi berhasil, pengguna dapat masuk ke dalam aplikasi menggunakan data yang di inputkan sebelumnya.
3. Untuk melakukan fitur pemanggilan, pengguna diharapkan mengizinkan aplikasi akses lokasi *device*.
4. Pengguna dapat menggunakan fitur pemanggilan, jika kebingungan dapat klik tombol bantuan untuk mengetahui cara penggunaan aplikasi.
5. Pengguna menekan *button* panggilan, maka sistem akan mengambil lokasi pengguna dan akan dikirimkan ke instansi terkait.
6. Setelah pengguna menelpon dan lokasi dikirim, instansi yang ditelpon akan mendapatkan panggilan dan lokasi dari penelpon.
7. Instansi menjawab panggilan dan mendapatkan lokasi penelpon. Sistem akan memproses algoritma penentuan rute dengan menjadikan lokasi instansi sebagai *node* awal dan lokasi penelpon sebagai *node* tujuan.
8. Setelah proses algoritma selesai, hasil dari penentuan rute akan di keluarkan dalam bentuk citra menggunakan Google Maps dengan rute yang sudah di dapatkan pada proses algoritma sebelumnya.

3.2. Data Requirements

Untuk merancang sistem aplikasi Panggilan Darurat ini, dibutuhkan beberapa data sebagai berikut:

a. Data lokasi Instansi

Data lokasi instansi diperlukan sebagai tujuan dalam proses algoritma. Untuk menggunakan data lokasi untuk kebutuhan algoritma, data lokasi berupa latitude dan longitude. Berikut merupakan lokasi instansi:

Tabel 1. Data Lokasi Instansi

No.	Jenis Instansi	Nama Instansi	Latitude	Longitude
1	Polisi	Polrestabes Bandung	-6.91391933114792	107.61050025478556
2	Rumah Sakit	Ali Ihsan	-7.007741026496179	107.6228539817707
3	Pemadam Kebakaran	Dinas Kebakaran Kota Bandung	-6.9404917405238855	107.67253441063059

b. Data lokasi pengguna

Untuk menggunakan aplikasi ini, penelpon diharapkan untuk mengaktifkan GPS pada handphone masing-masing.

c. Data lokasi *node*

Sama seperti lokasi instansi, untuk lokasi node diperlukan dalam bentuk latituda dan longitude. *Node* disini merupakan lokasi – lokasi pertigaan dan perempatan pada wilayah kabupaten bandung. Disini kami menggunakan jalan ring 1 dan ring 2. Dimana ring 1 adalah jalan utama atau jalan yang sering digunakan untuk lintas provinsi dan ring 2 adalah jalan yang sering digunakan di dalam kota Berikut merupakan data latitude longitude *node* yang digunakan dalam aplikasi ini:

Tabel 2. Data Lokasi *Node*

No.	**Latitude**	**Longitude**
1	-6.972155430431573	107.63355639562371
2	-6.9725440852050955	107.63404130699072
3	6.972822566764564	107.6361690705928
4	6.967833757459738	107.6345979024971
5	6.9694526888995005	107.63700865268265
6	6.968378417066985	107.63727269250158
7	6.966245742526659	107.63481445664898
8	6.965109882535197	107.6357325121516
9	6.965407884088816	107.63795585983732
10	6.96192962029697	107.63853700033711

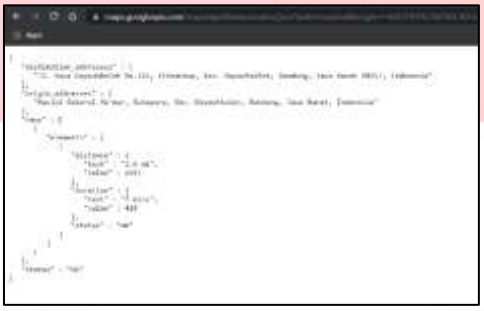
d. Data Parameter

Untuk menjalankan algoritma pencarian rute, diperlukan data seperti *actual cost* dan heuristik. Pada aplikasi ini, untuk mendapatkan nilai heuristik, digunakan *library* geo-lib-distance yang dimana fungsi ini memberikan nilai *straight line distance* antar lokasi satu dengan lokasi lainnya dengan memasukan latitude longitude untuk lokasi yang akan dihitung jaraknya. Contoh penggunaan dari geo-lib dapat dilihat pada gambar 3.2 dibawah ini.

```
getPreciseDistance(
  { latitude: 20.0504188, longitude: 64.4139099 },
  { latitude: 51.528308, longitude: -0.3817765 }
);
```

Gambar 5. Contoh Penggunaan Geo-Lib

Untuk nilai parameter yang lain, yaitu *actual cost* dan durasi tempuh, digunakan *google distance matrix* API. Sama seperti geo-lib, untuk mendaptkan nilai *actual* dan durasi, diperlukannya masukan latitude longitude untuk lokasi yang ingin dicari nilainya. Contoh penggunaan dari Api ini dapat dilihat pada gambar 3.3 dibawah ini.



Gambar 6. Contoh Penggunaan Google Distance Matrix

4. Implementasi dan Pengujian Sistem

4.1. Implementasi Sistem

Dalam pembahasan implementasi, penulis akan memaparkan hasil implementasi algoritma A* dan Prim pada aplikasi Panggilan Darurat. Dalam implementasi ini, penulis menggunakan framework React Native berbasis bahasa pemrograman Javascript dan Typescript dan *device* yang digunakan untuk menjalankan aplikasi ini adalah smartphone android. Untuk penulisan *source code*, penulis menggunakan Visual Studio Code.

4.2. Implementasi Algoritma

Algoritma yang penulis gunakan untuk mencari jalur terbaik pada Tugas Akhir ini adalah algoritma A* dan Prim. Dalam proses pencarian jalur terbaik, kami menggunakan 2 parameter untuk menjadi tolak ukur, yaitu parameter jarak dan durasi. Metode ini digunakan dalam menggabungkan kedua variabel tersebut yang nantinya akan mendapatkan bobot akhir yang digunakan dalam perhitungan pada Algoritma A* dan Prim. Adapun langkah-langkah pengimplementasian algoritma ke dalam aplikasi adalah sebagai berikut:

1. Nilai latitude longitude dari node dan lokasi instansi yang sudah disiapkan sebelumnya akan dimasukan ke dalam sebuah array.
2. Lokasi penelpon akan diambil melalui GPS dan lokasi tujuan instansi akan dipilih sesuai dengan instansi tang dipilih penelpon lalu lokasi tersebut akan dikonversikan menjadi latitude longitude untuk kebutuhan implementasi.
3. Setelah lokasi penelpon di dapatkan, akan dicari jarak terdekat menurut dari perhitungan algoritma yang digunakan dari lokasi penelpon dengan salah satu lokasi node dan akan dijadikan sebagai rute selanjutnya.
4. Proses penentuan node selanjutnya dan seterusnya sampai digunakan nilai algoritma dan durasi waktu tempuh dengan masing-masing persentase atau perbandingan bobot SAW nya akan ditentukan setelah melakukan pengujian bobot SAW.

4.3. Pengujian Bobot SAW

Pengujian bobot SAW dilakukan untuk mengetahui berapa perbandingan bobot yang optimal untuk jarak dan durasi. Pengujian ini dilakukan untuk mengetahui apakah pemilihan bobot dapat berpengaruh terhadap pemilihan rute yang dipilih. Pengujian dilakukan dengan lokasi awal adalah Universitas Telkom dan lokasi tujuan Rumah Sakit Umum Bina Sehat di hari yang sama dengan jam berbeda. Hasil pengujian bobot dapat dilihat pada tabel berikut.

Tabel 3. Hasil Pengujian Bobot

No	Jam	50% Jarak + 50% Durasi	60% Jarak + 40% Durasi	40% Jarak + 60% Durasi	70% Jarak + 30% Durasi	30% Jarak + 70% Durasi
1	9.30	Berhasil	Berhasil	Berhasil	Berhasil	Berhasil
2	13.00	Berhasil	Berhasil	Berhasil	Berhasil	Berhasil

No	Jam	50% Jarak + 50% Durasi	60% Jarak + 40% Durasi	40% Jarak + 60% Durasi	70% Jarak + 30% Durasi	30% Jarak + 70% Durasi
3	16.45	Berhasil	Berhasil	Berhasil	Berhasil	Berhasil
4	20.00	Berhasil	Berhasil	Berhasil	Berhasil	Berhasil

Berdasarkan percobaan yang dilakukan untuk memilih bobot optimum dengan jam yang berbeda di hari yang sama, semua percobaan menggunakan algoritma A* dan Prim berhasil dilakukan.

Pada pengujian SAW menggunakan algoritma A* dan Prim, ditemukan 20 jalur yang sama. Dari keempat waktu yang diujikan, lalu lintas terpadat terjadi pada pukul 09.30 dan setiap bobot yang berbeda memiliki rentang relatif yang minim.

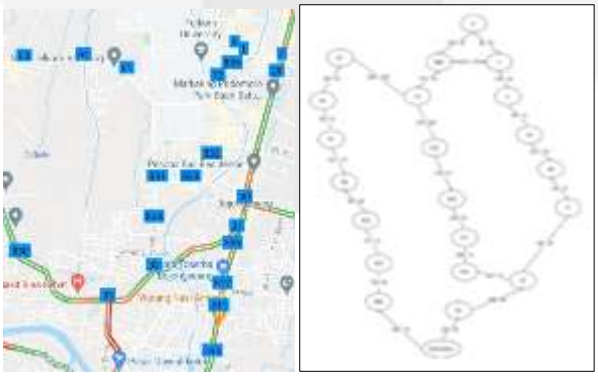
Tabel 4. Perbandingan Hasil Akhir Nilai Alternatif

Algoritma A*							
Jam	Lokasi	Tujuan	SAW				
			50:50	60:40	70:30	40:60	30:70
09.30	0	1	0,142371	0,167714	0,193057	0,117028	0,091686
		835	0,159625	0,182156	0,204686	0,137095	0,114564
Rentang			0,017254	0,01442	0,011623	0,020067	0,022878
Algoritma Prim							
Jam	Lokasi	Tujuan	SAW				
			50:50	60:40	70:30	40:60	30:70
09.30	0	1	0,013464	0,013025	0,012586	0,013902	0,014341
		835	0,041119	0,039948	0,038778	0,042290	0,043460
Rentang			0,027655	0,026923	0,025753	0,028388	0,029119

Pada tabel 4, terlihat rentang SAW algoritma A* untuk nilai rentang tertinggi terdapat pada SAW 30% jarak dan 70% durasi dengan nilai rentang 0,022878 dan pada algoritma Prim yang paling tinggi terdapat pada nilai perbandingan SAW 30% jarak dan 70% durasi dengan nilai rentang 0,029119. Dapat disimpulkan, untuk bobot SAW yang digunakan untuk masing-masing algoritma adalah 30% jarak dan 70% durasi.

4.4. Pengujian Akurasi

Pengujian akurasi merupakan pengujian yang dilakukan untuk mengetahui performasi algoritma yang digunakan dalam aplikasi. Pengujian akurasi akan dihitung akurasinya dengan membandingkan hasil hitungan atau pemilihan rute oleh sistem dengan hasil pemilihan rute secara manual. Pemilihan rute disini akan digunakan Telkom University sebagai lokasi awal dan Rumah Sakit Umum Bina Sehat sebagai lokasi tujuan. Untuk node yang digunakan pada pengujian ini dapat dilihat pada gambar 7 dibawah ini:



Ggambar 7. Node *Map* dan Graph

Untuk nilai heuristik dari node ke tujuan sendiri, dapat dilihat pada tabel berikut:

Tabel 8. Nilai Heuristik / Nilai Estimasi

No	Key	Heuristtik
1	0	1798
2	1	1792
3	2	1912
4	29	1814
5	30	1178
6	31	1057

No	Key	Heuristtik
7	32	492
8	33	192
9	34	1547
10	61	1389
11	62	1438
12	63	1468
13	64	1544
14	72	1570
15	510	915
16	828	1361
17	829	1095
18	830	422
19	831	846
20	832	1170
21	833	980
22	834	645
23	835	1682

Untuk A*, kita memerlukan nilai aktual antara node dan nilai heuristik node berikutnya ke tujuan dan kemudian ambil 30% dari hasil dan kemudian 70% durasi antara node. Misalnya, node 0 terhubung ke node 835 dan 1. Ambil jarak antara node 0 dan 835 dan nilai heuristik node 835, kemudian jumlahkan jarak dan heuristik antara node dan ambil 30% dari nilai hasil dan ambil 70% dari nilai durasi antara node 0 dan 835 dan ulangi langkah sebelumnya ke node lain yang terhubung dan pilih dengan hasil terendah. Setelah itu, hasil terendah akan menjadi node saat ini dan ulangi langkah di atas sampai node saat ini adalah tujuan.

Sedangkan untuk Prim, kita membutuhkan nilai aktual antara node dan ambil 30% dari nilai dan mengambil 70% dari nilai durasi antara node. Misalnya, node 0 terhubung ke node 835 dan 1. Ambil jarak antara node 0 dan 835, kemudian ambil 30% dari nilai jarak dan ambil 70% nilai durasi antara node 0 dan 835 dan ulangi ke node lain yang terhubung dan pilih dengan hasil terendah. Setelah itu, hasil terendah akan menjadi node saat ini dan ulangi langkah di atas sampai node saat ini adalah tujuan.

Untuk rute, menggunakan algoritma A * dan Prim, hasilnya dapat dilihat pada Gambar 8 dan Gambar 9 di bawah ini.

Gambar 8. Perbandingan Hasil Program dan Hitung Manual Algoritma A*

Gambar 8. Perbandingan Hasil Program dan Hitung Manual Algoritma Prim

Seperti yang dapat kita lihat dalam Gambar 8 dan 9 di atas bahwa algoritma A * dan Prim berhasil diimplementasikan di program. Untuk hasil detail, kita bisa lihat pada tabel di bawah ini.

Tabel 9. Hasil Algoritma

Algorithm	Travel Distance (Kilometers)	Travel Time (Minutes)
A* (A Star)	3,14	11,7
Prim	2,56	6,75

Dari hasil seperti yang kita lihat pada tabel 9,dapat dikatakan bahwa algoritma Prim memberikan hasil terbaik dengan jarak tempuh 2,56 kilometer dan durasi perjalanan 6,75 menit dari A * dengan jarak tempuh 3,14 kilometer dan durasi perjalanan 11,7 menit.

Untuk waktu respons algoritma, kita dapat melihat tabel di bawah ini.

Tabel 10. Waktu Respon Algoritma

No.	A* Response Time (second)	Prim Response Time (Second)
1	14,38	17,23
2	13,02	18,53
3	11,53	14,67
4	11,48	15,20
5	10,93	15,23
6	10,68	13,65
7	11,32	13,46
8	11,08	13,29
9	10,64	15,32
10	12,54	14,12

$$\text{Rata-rata A* } \textit{Response Time} = \frac{reponse\ time's\ amount}{total\ response\ time}\ x\ 100\% = \frac{117{,}6}{10} = 11{,}76 \text{ detik}$$

$$\text{Rata-rata Prim } \textit{Response Time} = \frac{response\ time's\ amount}{total\ response\ time}\ x\ 100\% = \frac{150{,}7}{10} = 15{,}07 \text{ detik}$$

5. Kesimpulan

Dari penelitian ini, dapat diambil kesimpulan:

1. Algoritma A* dan Prim diimplementasikan dengan baik dalam aplikasi menggunakan bobot SAW dimana nilai jarak 30% dan nilai durasi 70%.
2. Pengujian dengan lokasi awal Telkom University dan RSU Bina Sehat sebagai tujuan, algoritma Prim dapat memberikan rute terbaik dengan waktu tempuh 6,75 menit dan jarak tempuh 2,56 kilometer. Tetapi untuk waktu respons, algoritma A* lebih cepat dengan waktu respons rata-rata 11,76 detik.

REFERENSI

[1] A. T. Haryanto, "detikinet," Detik Network, 20 February 2020. [Online]. Available: https://inet.detik.com/cyberlife/d-4907674/riset-ada-1752-juta-pengguna-internet-di-indonesia. [Accessed 20 November 2020].

[2] T. Narognsak, "Development of Metropolitan Police Emergency Call System," *International Journal of Crime, Law and Social Issues*, vol. 5, no. 1, pp. 134 - 145, 2018.

[3] Desiaman, "Penentuan Jalur Terpendek dengan Menggunakan Algoritma Djikstra," *KAKIFIKOM (KUMPULAN ARTIKEL KARYA ILMIAH FAKULTAS ILMU KOMPUTER)*, vol. I, no. 1, pp. 1-5, 2019.

[4] J. Daud, "Studi Efektifitas Pengunaan Halte di Kota Medan," *Jurnal Sistem Teknik Industri*, vol. VI, no. 3, pp. 73-80, 2005.

[5] M. K. H and N. Khairina, "Pencarian Jalur Terpendek Dengan Algoritma Dijkstra," *Jurnal dan Penelitian Teknik Informatika*, vol. II, no. 2, pp. 18-23, 2017.

[6] H. J. S. S.-H. L. Nam Kyu Kang, "Modified A-star algorithm for modular plant land transportation," *Journal of Mechanincal Science and Technology*, vol. 32, no. 12, pp. 5563 - 5571, 2018.

[7] A. B. M. K. e. Frantisek Duchon, "Path Planning With Modified A Star Algorithm For a Mobile Robot," *Procedia Engineering*, vol. 96, no. 96, pp. 59 - 69, 2014.

[8] M. H. Falurrahman, "Implementasi dan Analaisis Penggunaan Algoritma A-Star Dengan Prioritas Pemilihan Rute Lintas Kendaraan Roda Dua," Universitas Telkom, Bandung, 2014.

[9] D. G. Xiang Liu, "A Comperative Study of A-Star Algorithms for Search and Rescue in Perfect Maze," in *International Conference on Electric Information and Control Engineering*, Beijing, 2011.

[10] H. P. F. X. S. Y. Zhou F, "Improved Prim Algorithm and Its Application in Unmaned Aerial Vehicle Cruise System," in *Chinese Control and Decision Conference*, 2017.

[11] M. Furqan, "A Review Of Prim and Genetic Algorithms in FInding and Determining Routes On Connected Weighted Graphs," *International Journal of Civil Engineering and Technology (IJCIET)*, vol. 9, no. 9, pp. 1755 - 1765, 2018.