Penerapan Floyd-Warshall untuk Pencarian Rute Terpendek pada Aplikasi Notifikasi Kecelakaan Lalu Lintas

Haniah Mahmudah[1], M Fajar Ibrahim[2], Okkie Puspitorini[3], Ari Wijayanti[4], Nur Adi Siswandari[5]

Intisari*—Dalam penanganan kecelakaan, diperlukan respons yang cepat untuk mencegah kecelakaan atau memberikan pertolongan pada kecelakaan lalu lintas. Beberapa cara mencegah kecelakaan atau menolong korban kecelakaan lalu lintas antara lain dengan membuat beberapa aplikasi pada *smartphone* untuk deteksi kecelakaan dan aplikasi notifikasi kecelakaan untuk memberikan pertolongan pada kecelakaan lalu lintas. Aplikasi untuk menolong korban kecelakaan yang telah ada hanya menyajikan rute terpendek dari responden menuju lokasi korban tanpa ada fitur yang membantu responden mencari rute ke rumah sakit dan kantor polisi terdekat. Untuk itu, penelitian ini membuat aplikasi *smartphone* untuk notifikasi kecelakaan bagi keluarga korban dan responden serta membantu mencari rute menuju lokasi korban dan rute ke rumah sakit serta kantor polisi terdekat. Aplikasi notifikasi kecelakaan untuk *smartphone* ini menggunakan perangkat lunak *open source* dan memiliki keunggulan berupa *scalability* yang tinggi. Hasil penelitian ini berupa aplikasi Android yang mampu mengirimkan *broadcast* notifikasi kecelakaan, sehingga keluarga korban dan responden dapat menerima notifikasi kecelakaan dan dapat melakukan perjalanan menuju lokasi kecelakaan sesuai rute yang telah ditampilkan oleh aplikasi. Aplikasi ini juga memberikan informasi tentang lokasi rumah sakit dan kantor polisi terdekat, sehingga responden yang berada di sekitar lokasi dapat menolong korban dengan cepat. Hasil pengujian aplikasi menggunakan metode *black box* pada *platform* Android menunjukkan bahwa 100% fitur aplikasi ini berjalan dengan baik. Rute terpendek dengan algoritme Floyd-Warshall adalah 4,199 km dengan tidak mengalami penyimpangan rute dari skenario pengujian jarak. Kecepatan rata-rata respons pengiriman notifikasi dari korban ke responden adalah 27,86 ms.

***Kata Kunci*—Kecelakaan Lalu Lintas, Aplikasi Android, Algoritme Floyd-Warshall, *Black Box*.**

I. PENDAHULUAN

Peningkatan populasi penduduk mengakibatkan peningkatan jumlah kendaraan, yang akhirnya juga meningkatkan jumlah kecelakaan lalu lintas. Dalam penanganan kecelakaan, diperlukan respons yang cepat untuk menolong korban kecelakaan [1]. Untuk mengurangi angka korban kecelakaan, diperlukan cara untuk mencegah kecelakaan atau memberikan pertolongan pada kecelakaan lalu lintas.

Pencegahan kecelakaan lalu lintas dapat dilakukan melalui beberapa cara, antara lain memberikan penjelasan tentang cara berkendara yang baik serta kesadaran tentang pentingnya keselamatan bagi pengemudi selama perjalanan. Beberapa penelitian tentang pencegahan kecelakaan dilakukan antara lain dengan mendeteksi kondisi pengemudi yang berkendara secara normal atau tidak normal, seperti bergerak, belok kiri, belok kanan, pengereman mendadak, dan putar balik dengan kecepatan tinggi. Sementara itu, penelitian untuk menolong korban kecelakaan lalu lintas antara lain dengan pembuatan beberapa aplikasi pada *smartphone* untuk deteksi kecelakaan dan aplikasi notifikasi kecelakaan untuk memberikan pertolongan pada kecelakaan lalu lintas [2].

Penggunaan perangkat lunak *open source* memudahkan pembuatan aplikasi *smartphone* dengan memanfaatkan beberapa sensor yang telah terpasang. Hal ini menjadikan *smartphone* memiliki keunggulan yaitu *scalability*-nya yang tinggi. Beberapa penelitian tentang aplikasi *smartphone* untuk mendeteksi kecelakaan antara lain tentang deteksi kecelakaan berupa perangkat keras yang terhubung dengan beberapa sensor yang memberikan notifikasi jika terjadi kecelakaan. Pada penelitian lain, dibuat aplikasi Android, antara lain aplikasi korban kecelakaan dan aplikasi responden untuk menentukan lokasi kecelakaan, sehingga korban kecelakaan lebih cepat memperoleh pertolongan [3]. Selain itu, penelitian yang memanfaatkan *internet of things* (IoT) untuk mendeteksi kecelakaan dengan menggunakan komunikasi GSM dapat mengirimkan data beberapa sensor, antara lain sensor gas MQ2 dan sensor kecepatan menggunakan sensor efek Hall [4]. Namun, aplikasi untuk menolong korban kecelakaan hanya menyajikan rute terpendek dari responden menuju ke lokasi korban tanpa ada fitur yang membantu responden mencari rute ke rumah sakit terdekat dan kantor polisi [3], [4].

Beberapa penelitian penentuan rute terpendek menggunakan berbagai *website* dan aplikasi Android dengan berbagai macam algoritme [1]−[21]. Salah satu algoritme yang digunakan adalah algoritme Floyd-Warshall. Pemrograman dinamis melalui langkah-langkah perhitungan algoritme Floyd-Warshall memiliki kemampuan proses perhitungan cepat dan sederhana dibandingkan dengan penelitian yang lain tentang pencarian rute terpendek untuk jalur evakuasi korban kebakaran [5]. Hasil penelitian algoritme Floyd-Warshall mampu memberikan rute terpendek pada tiap lantai di gedung hotel bertingkat. Selain itu, terdapat penelitian yang menggunakan diagram grafik serta menerapkan algoritme Floyd-Warshall dalam perhitungan bobot *path* sehingga menghasilkan suatu aplikasi untuk mencari rute terpendek pada objek wisata [6]. Juga telah diteliti aplikasi pencari pertandingan futsal menggunakan algoritme Floyd-Warshall pada peta [7]. Selain itu, beberapa penelitian menggunakan beberapa algoritme, antara lain A*Star, Djiskstra, Bellman-Ford, dan Floyd-Warshall untuk penentuan rute terpendek [8]-[10].

[1,2,3,4,5] *Program Studi Teknik Telekomunikasi, Departemen Teknik Elektro, Politeknik Elektronika Negeri Surabaya, Jln. Raya ITS Keputih 60111 Surabaya (tlp: 031-5947280; fax: 031- 5946114; email: [1]haniah@pens.ac.id, [2]fajar@stutendt.pens.ac.id, [3]okkie@pens.ac.id, [4]ariw@pens.ac.id, [5]nuradi@pens.ac.id)*

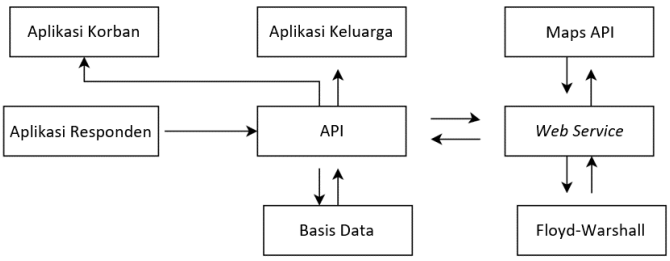
[Diterima: 22 Juli 2021, Revisi: 2 Agustus 2022]

Untuk mengurangi angka kematian korban kecelakaan, salah satu cara yang dapat dilakukan adalah dengan menolong korban kecelakaan sesegera mungkin. Apabila korban kecelakaan mengalami luka parah, diperlukan waktu yang cepat untuk melakukan pertolongan, antara lain dengan membawa korban kecelakaan ke rumah sakit atau melapor ke kantor polisi terdekat. Untuk dapat menolong korban kecelakaan dengan cepat, diperlukan pemilihan rute terpendek sehingga responden dapat segera mencapai lokasi korban. Dengan menggunakan pemrograman dinamis dan algoritme penentuan rute terpendek, dapat ditentukan rute terbaik yang memerlukan waktu paling singkat. Pada penelitian sebelumnya, aplikasi hanya memberitahukan posisi korban kecelakaan, tetapi belum ada rute menuju rumah sakit dan kantor polisi. Oleh karena itu, pada penelitian ini dirancang aplikasi untuk korban, keluarga, dan responden, yang memberikan notifikasi kecelakaan dengan cara mengirimkan lokasi kecelakaan pada pihak keluarga dan beberapa responden yang dekat dengan lokasi korban, sehingga posisi korban kecelakaan dan rute terpendek menuju posisi korban kecelakaan dapat diketahui. Pada penelitian ini, penentuan rute terpendek dilakukan menggunakan algoritme Floyd-Warshall karena pemrogramannya yang bersifat dinamis. Algoritme Floyd-Warshall memiliki kemampuan proses perhitungan yang cepat dan sederhana, sehingga mudah diaplikasikan pada *smartphone.* Pada aplikasi tersebut, notifikasi kecelakaan yang dikirimkan kepada keluarga korban dan responden berupa informasi untuk mencari rute terpendek menuju lokasi kecelakaan, rumah sakit terdekat, dan kantor polisi. Penggunaan titik koordinat GPS pada lokasi awal dan tujuan dioptimalkan untuk mencari rute terpendek. Sementara itu, dalam mencari rute terpendek, penelitian ini menggunakan algoritme Floyd-Warshall dengan parameter-parameter antara lain: koordinat GPS lokasi awal dan tujuan, jarak antara simpul atau *node*, serta jalur dan jarak total yang dihasilkan antara titik awal dan tujuan. Pengujian aplikasi dilakukan menggunakan *black box* untuk mengetahui kinerja aplikasi notifikasi kecelakaan.

II. METODOLOGI

Pengembangan aplikasi ini dilakukan menggunakan pendekatan *object oriented* berupa metode *waterfall* dengan fase identifikasi, fase analisis, fase perancangan, dan fase uji coba. Hal yang pertama dilakukan adalah mengidentifikasi dan mengoptimalkan semua alternatif titik untuk jalur terpendek dan jalur tercepat. Pada fase analisis, direncanakan bentuk, struktur navigasi, Unified Modeling Language (UML), serta dilakukan analisis dan perhitungan jalur optimal yang dilalui untuk menuju lokasi kecelakaan, sehingga diperoleh hasil yang optimal. Setelah proses algoritme Floyd-Warshall mendapatkan nilai yang optimal, selanjutnya dilakukan tahap desain aplikasi. Fase perancangan aplikasi, yaitu aplikasi bagi korban, keluarga, dan responden ditunjukkan pada Gbr. 1.

Tahapan pertama pada Gbr. 1 adalah saat terjadi kecelakaan, sensor pada aplikasi *smartphone* korban akan bekerja, lalu aplikasi korban melakukan klasifikasi kecelakaan. Ketika hasil klasifikasi menyatakan kecelakaan, akan muncul peringatan dengan waktu tunggu 30 s. Jika tidak ada tanggapan, sistem



Gbr. 1 Perancangan aplikasi pada *smartphone*.

akan mengirimkan notifikasi ke responden dan keluarga korban. Notifikasi kecelakaan dikirimkan ke keluarga korban melalui nomor telepon keluarga yang terdaftar. Notifikasi yang dikirim berupa lokasi *latitude* dan *longitude* serta identitas korban. Data tersebut diproses pada *web service* untuk penentuan rute teroptimal menggunakan algoritme Floyd-Warshall dan menampilkannya pada Maps API. API merupakan jembatan untuk pertukaran data antara *web service* sebagai server dengan aplikasi Android sebagai klien. Perhitungan algoritme Floyd-Warshall memerlukan masukan titik awal dan titik tujuan berupa *latitude* dan *longitude*. Sebelum proses penentuan rute dari lokasi responden menuju lokasi korban, dilakukan skenario rute seperti pada Gbr. 1.

Dari skenario pada Gbr. 1 akan diperoleh rute dan jarak antar *node* terhubung. *Node* tersebut diubah ke dalam bentuk matriks $m \times m$ untuk dijadikan sebagai masukan algoritme Floyd-Warshall dalam menentukan rute yang optimal. Diagram alir algoritme Floyd-Warshall ditunjukkan pada Gbr. 2

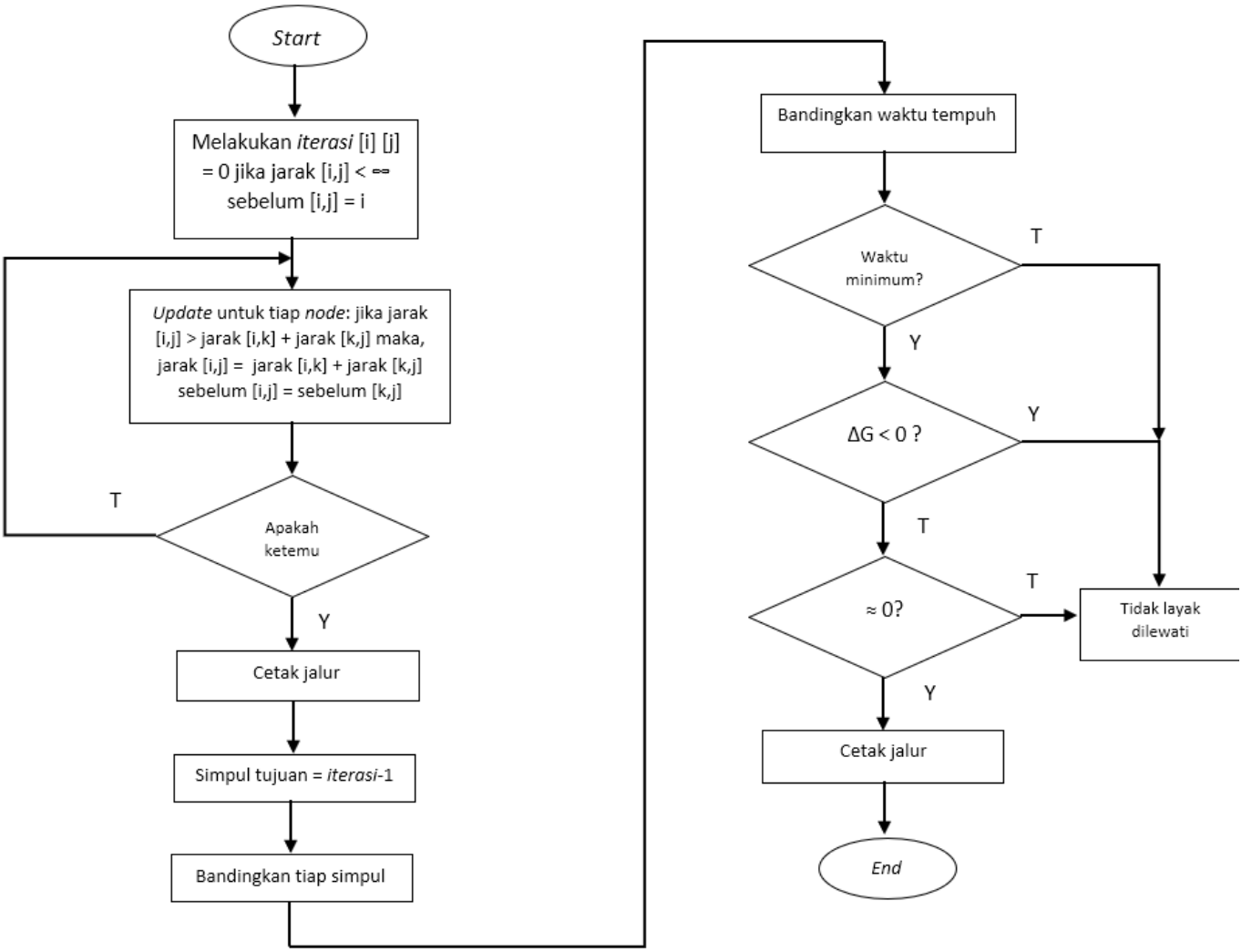
Pada Gbr. 2, proses awal dilakukan dengan memasukkan titik awal dan tujuan dengan nilai jarak ke matriks $m \times m$. Kemudian, dilakukan proses dengan *iterasi* sebanyak jumlah *node*. Dari iterasi terakhir diperoleh hasil berupa jarak terpendek dari titik awal ke tujuan. Setelah diperoleh jarak terpendek, dilakukan proses perbandingan jarak tempuh untuk mendapatkan rute yang optimal. Jika telah diperoleh rute yang optimal, jalur dicetak dan ditampilkan di aplikasi.

A. Broadcast Notifikasi

Proses pengiriman notifikasi dilakukan setelah hasil klasifikasi kecelakaan menyatakan terjadinya kecelakaan pada korban. Aplikasi korban mengirimkan notifikasi secara *broadcast* ke beberapa responden. Responden yang menerima notifikasi adalah responden yang berada dalam radius lokasi korban, sedangkan responden yang berada di luar radius akan menerima notifikasi melalui parameter nomor telepon keluarga yang telah didaftarkan oleh korban. Perhitungan jarak respoden dengan korban dilakukan menggunakan persamaan *spherical law of cosines* dengan parameter dua titik *longitude* dan *latitude* lokasi responden dan korban serta jari-jari bumi, sehingga diperoleh jarak responden dan korban. Adapun rumus perhitungan jarak responden dan lokasi korban menggunakan persamaan *spherical law of cosines* ditunjukkan pada (1).

$$d = a.cos(sin(A).\ sin(B) + cos(A).cos(B).cos(D\text{-}C)).R \quad (1)$$

dengan R adalah jari-jari bumi sebesar 6.371 km, d adalah jarak dalam km, A adalah *latitude* 1, B adalah *latitude* 2, C adalah *longitude* 1, dan D adalah *longitude* 2.



Gbr. 2 Alur algoritme Floyd-Warshall.

Broadcast notifikasi dilakukan dengan memanfaatkan Firebase Cloud Messaging (FCM). FCM merupakan fitur dari Firebase. FCM dapat mengirimkan notifikasi maupun data secara *real-time* ke aplikasi responden. Terdapat dua kelas yang digunakan pada FCM, yaitu *FirebaseInstanceIdService* dan *FirebaseMessagingService. FirebaseInstanceIdService* digunakan untuk menghasilkan *token* dari FCM. *Token* ini digunakan sebagai identitas perangkat responden yang penerima notifikasi. Di sisi lain, *FirebaseMessagingService* digunakan untuk menerima notifikasi dan data yang dikirimkan oleh korban.

B. Algoritme Floyd-Warshall

Pencarian rute dilakukan dengan perhitungan dinamis menggunakan pemrograman *web* PHP dengan teori graf untuk proses penentuan rute terpendek pada aplikasi algoritme Floyd-Warshall. Skenario pengujian jalur terpendek berdasarkan jarak ditunjukkan pada Gbr. 3, yaitu berupa gambar peta yang berlokasi di Surabaya tengah.

Dapat dilihat pada Gbr. 3 bahwa jalur yang dapat dilewati oleh responden menuju lokasi korban adalah sebagai berikut.

1) Jalur Pertama: Jalur pertama didefinisikan sebagai T0. *Node* yang dilalui oleh T0 adalah T0 = 0-1-3-6-5-7-8. Jalur T0 ditunjukkan pada Gbr. 4.

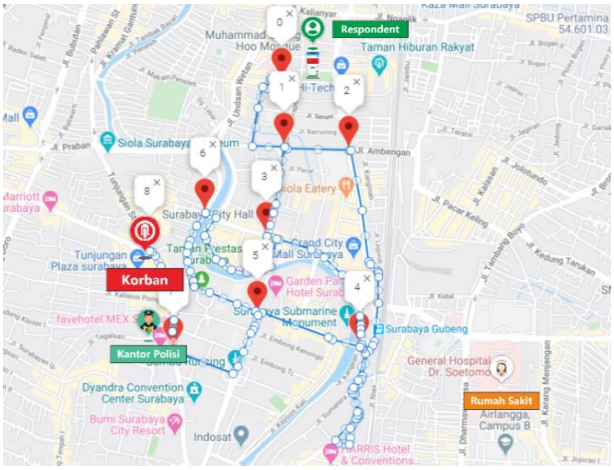
2) Jalur Kedua: Jalur kedua didefinisikan sebagai T1. *Node* yang dilalui oleh T1 adalah T1 = 0-1-3-4-5-7-8. Jalur T1 ditunjukkan pada Gbr. 5.

3) Jalur Ketiga: Jalur ketiga didefinisikan sebagai T2. *Node* yang dilalui oleh T2 adalah T2 = 0-1-2-4-5-7-8. Jalur T2 ditunjukkan pada Gbr. 6.

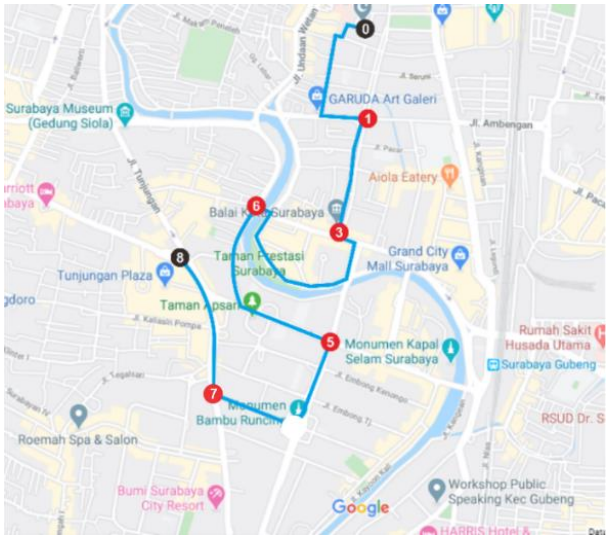
Dari skenario pengujian seperti pada Gbr. 3, diperoleh data titik koordinat dan jarak antar *node* dengan titik awal (T_0) dan titik tujuan (T_k), seperti pada Tabel I. Selanjutnya, data dikonversi menjadi *node* dan jarak ditampilkan dalam satuan meter.

Tahapan perhitungan untuk menentukan rute terpendek pada algoritme Floyd-Warshall menggunakan (2). Matriks dengan baris adalah lokasi awal, sedangkan kolom adalah lokasi tujuan.

$$x[i,j] \leq x[i,k] + x[k,j] \quad (2)$$



Gbr. 3 Skenario rute pengujian.



Gbr. 4 Rute pertama (T0).

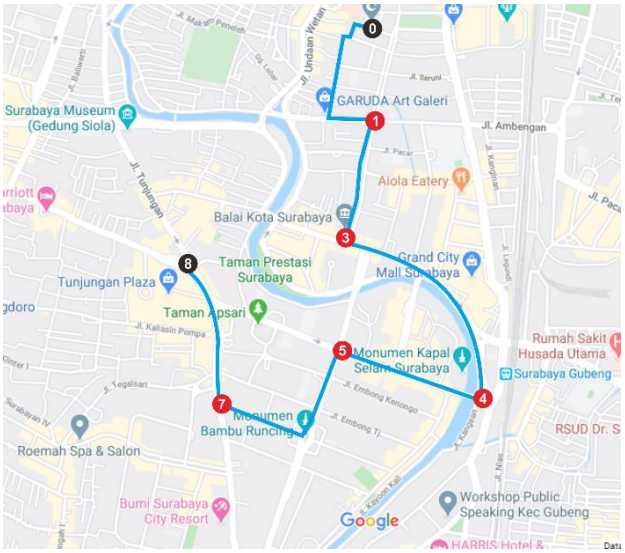
dengan

k = 0, 1, 2, 3, 4, 5, 6, 7, 8, 9 (banyaknya iterasi),
i = 0, 1, 2, 3, 4, 5, 6, 7, 8 (titik awal),
j = 0, 1, 2, 3, 4, 5, 6, 7, 8 (titik tujuan).

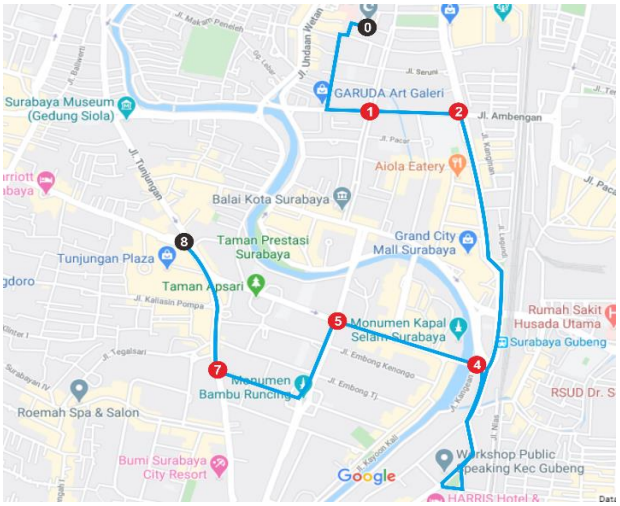
Pada (2), nilai yang diambil adalah nilai terkecil dari perbandingan antara $x[i,j]$ dengan hasil dari operasi $x[i,k] + x[k,j]$. Jika $x[i,j] \leq x[i,k] + x[k,j]$, nilai $x[k,j]$ akan dipilih. Jika $x[i,j] \geq x[i,k] + x[k,j]$, $x[i,j] = x[i,k] + x[k,j]$. Proses pertukaran nilai ini dilakukan melalui proses iterasi sebanyak k kali dengan k = 0 merupakan matriks yang didefinisikan di awal dan iterasi pertama dilakukan mulai dari iterasi k = 1 sampai k = 9.

Pada iterasi pertama k = 1, data pada baris ke-0 kolom ke-0 sampai kolom ke-8 serta baris ke-0 sampai baris ke-8 kolom ke-0 dan baris ke-$x[i][j]$, dengan ketentuan $i = j$ dengan nilai 0 (yang membentuk garis diagonal), nilainya tetap. Proses iterasi perbandingan nilai dilakukan mulai dari $x[1][2]$ hingga $x[8][7]$ dan dipilih nilai yang paling kecil.

Dari iterasi pertama, diperoleh data matriks dengan perubahan nilai pada data ke-$x[0][2]$, $x[0][3]$, $x[2][3]$, $x[3][2]$, $x[3][0]$, dan $x[2][0]$. Selanjutnya, hasil dari perhitungan iterasi pertama akan digunakan untuk iterasi kedua dengan nilai yang



Gbr. 5 Rute kedua (T1).



Gbr. 6 Rute ketiga (T2).

tetap berganti pada posisi baris ke-1 kolom ke-0 sampai kolom ke-8 serta baris ke-0 sampai baris ke-8 kolom ke-1. Cara yang sama dilakukan sampai iterasi ke-8, seperti pada Tabel II.

Dari iterasi kedelapan diperoleh data matriks dengan perubahan nilai pada data ke-$x[0][8]$, $x[1][8]$, $x[2][8]$, $x[3][8]$, $x[4][8]$, $x[5][8]$, $x[6][8]$, $x[8][0]$, $x[8][1]$, $x[8][2]$, $x[8][3]$, $x[8][4]$, $x[8][5]$, dan $x[8][6]$. Selanjutnya hasil dari perhitungan iterasi kedelapan digunakan untuk iterasi kesembilan dengan nilai yang tetap berganti pada posisi baris ke-8 kolom ke-0 sampai kolom ke-8 dan baris ke-0 sampai baris ke-8 kolom ke-7.

Dari iterasi kesembilan diperoleh data matriks seperti pada Tabel III, dengan perubahan nilai pada data sehingga semua *node* akan terhubung satu dengan yang lain dan menghasilkan jarak antar *node*. Tabel III menunjukkan jarak total dari titik awal ke titik tujuan. Misalnya, jarak total yang akan ditempuh dari titik 0 menuju titik 8 adalah 4,199 km.

Jalur yang dilalui pada algoritme Floyd-Warshall dari titik awal, yaitu *node* 0, sampai titik tujuan, yaitu *node* 8, ditunjukkan pada Tabel IV. Selanjutnya, dilihat *node* yang

TABEL I
JARAK ANTAR *NODE*

No.	T_0	T_k	Koordinat	Jarak (m)
1	0	1	{"nodes": ["0-1"], "coordinates": [[-7,25242724455...	69,745
2	1	0	{"nodes": ["1-0"], "coordinates": [[-7,25584599630...	695,745
3	1	2	{"nodes": ["1-2"], "coordinates": [[-7,25584599630...	379,264
4	2	1	{"nodes": ["2-1"], "coordinates": [[-7,25601213972...	379,264
5	2	4	{"nodes": ["2-4"], "coordinates": [[-7,25601213972...	2.525,243
6	4	2	{"nodes": ["4-2"], "coordinates": [[-7,26631210166...	2.525,243
7	1	3	{"nodes": ["1-3"], "coordinates": [[-7,25584599630...	530,530
8	3	1	{"nodes": ["3-1"], "coordinates": [[-7,26050714532...	530,530
9	3	4	{"nodes": ["3-4"], "coordinates": [[-7,26050714532...	957,575
10	4	3	{"nodes": ["4-3"], "coordinates": [[-7,26631210166...	957.575
11	4	5	{"nodes": ["4-5"], "coordinates": [[-7.26631210166...	634,990
12	5	4	{"nodes": ["5-4"], "coordinates": [[-7,26464043859...	634,990
13	3	6	{"nodes": ["3-6"], "coordinates": [[-7,26050714532...	916,764
14	6	3	{"nodes": ["6-3"], "coordinates": [[-7,25928164731...	916,764
15	6	5	{"nodes": ["6-5"], "coordinates": [[-7,25928164731...	874,962
16	5	6	{"nodes": ["5-6"], "coordinates": [[-7,26464043859...	874,962
17	5	7	{"nodes": ["5-7"], "coordinates": [[-7,26464043859...	747,149
18	7	5	{"nodes": ["7-5"], "coordinates": [[-7,26653668999...	747,149
19	7	8	{"nodes": ["7-8"], "coordinates": [[-7,26653668999...	631,931
20	8	7	{"nodes": ["8-7"], "coordinates": [[-7,26127898209...	631,931

terhubung langsung dengan *node* 8 di antara *node* 0 (titik awal yang dicari) dengan *node* 8 (titik tujuan yang dicari). *Node* yang terhubung dengan *node* 8 adalah node 7. *Node* 7 terhubung dengan *node* 5, *node* 5 terhubung dengan *node* 4, dan seterusnya, sehingga ditemukan *node* awal yang dicari, yaitu *node* 0. Hasil perhitungan algoritme Floyd-Warshall pada Tabel IV menunjukkan bahwa jarak terpendek dari titik responden menuju lokasi korban adalah jalur T1 = 0-1-3-4-5-7-8, dengan jarak tempuh 4,199 km.

III. HASIL DAN PEMBAHASAN

Tampilan aplikasi notifikasi kecelakaan pada korban dan responden yang dirancang ditunjukkan pada Gbr. 7. Jika aplikasi korban menyatakan terjadi kecelakaan, aplikasi akan melakukan pengiriman notifikasi secara *broadcast* ke responden. Pada aplikasi responden akan muncul notifikasi pada *notification bar*. Data yang diterima oleh responden

TABEL II
HASIL PROSES ITERASI KE-*K*

Iterasi ke-*k*	Matriks pada Data
1	*x*[0][2], *x*[0][3], *x*[2][3], *x*[3][2], *x*[3][0], *x*[2][0]
2	*x*[0][4], *x*[1][4], *x*[4][1], *x*[4][0].
3	*x*[0][4], *x*[1][4], *x*[2][4], *x*[6][4], *x*[0][6], *x*[1][6], *x*[2][6], *x*[4][6], *x*[4][0], *x*[4][1], *x*[4][2], *x*[4][6], *x*[6][0], *x*[6][1], *x*[6][2], *x*[6][4]
4	*x*[0][5], *x*[1][5], *x*[2][5], *x*[3][5], *x*[5][0], *x*[5][1], *x*[5][2], *x*[5][3]
5	*x*[4][6], *x*[0][7], *x*[1][7], *x*[2][7], *x*[3][7], *x*[4][7], *x*[6][7], *x*[6][4], *x*[7][0], *x*[7][1], *x*[7][2], *x*[7][3], *x*[7][4], *x*[7][6]
6	*x*[0][7], *x*[1][7], *x*[2][7], *x*[3][7], *x*[4][7], *x*[6][7], *x*[7][0], *x*[7][1], *x*[7][2], *x*[7][3], *x*[7][4], *x*[7][6]
7	*x*[0][8], *x*[1][8], *x*[2][8], *x*[3][8], *x*[4][8], *x*[5][8], *x*[6][8], *x*[8][0], *x*[8][1], *x*[8][2], *x*[8][3], *x*[8][4], *x*[8][5], *x*[8][6]
8	*x*[0][8], *x*[1][8], *x*[2][8], *x*[3][8], *x*[4][8], *x*[5][8], *x*[6][8], *x*[8][0], *x*[8][1], *x*[8][2], *x*[8][3], *x*[8][4], *x*[8][5], *x*[8][6]

TABEL III
HASIL AKHIR PROSES ITERASI KESEMBILAN

	0	1	2	3	4	5	6	7	8
0	0	0,696	1,075	1,227	2,185	2,82	2,144	3,567	4,199
1	0,696	0	0,379	0,531	1,489	2,124	1,448	2,871	3,503
2	1,075	0,379	0	0,910	1,868	2,503	1,827	3,25	3,882
3	1,227	0,531	0,910	0	0,958	1,593	0,917	2,34	2,972
4	2,185	1,489	1,868	0,958	0	0,635	1,510	1,382	2,014
5	2,820	2,124	2,503	1,593	0,635	0	0,875	0,747	1,379
6	2,144	1,448	1,827	0,917	1,51	0,875	0	1,622	2,254
7	3,567	2,871	3,250	2,340	1,382	0,747	1,622	0	0,632
8	4,199	3,503	3,882	2,972	2,014	1,379	2,254	0,632	0

TABEL IV
JALUR YANG DILEWATI DARI TITIK AWAL KE TUJUAN

	0	1	2	3	4	5	6	7	8
0	0	0	1	1	3	4	3	5	7
1	1	1	1	1	3	4	3	5	7
2	1	2	2	1	3	4	3	5	7
3	1	3	1	3	3	4	3	5	7
4	1	3	1	4	4	4	5	5	7
5	1	3	1	4	5	5	5	5	7
6	1	3	1	6	5	6	6	5	7
7	1	3	1	4	5	7	5	7	7
8	1	3	1	4	5	7	5	8	8

adalah nama korban serta lokasi korban berupa koordinat *latitude* dan *longitude*. Secara bersamaan, pada halaman *home* aplikasi responden akan muncul informasi terjadinya kecelakaan seperti yang ditunjukkan pada Gbr. 7. Informasi yang ditampilkan berupa koordinat *latitude* dan *longitude* dari korban dan jarak lokasi responden ke lokasi korban. Pada peta tertampil *marker* dari lokasi korban sesuai koordinat *latitude* dan *longitude* serta *marker* dari lokasi responden. Pada halaman tersebut juga terdapat beberapa aktivitas yang dapat dilakukan oleh responden, di antaranya melihat daftar foto,

Gbr. 7 Aplikasi responden saat menampilkan rute menuju lokasi korban.

yang memungkinkan responden melihat foto kejadian yang diambil oleh responden lain; mengambil gambar yang dapat dilakukan oleh responden untuk memberikan bukti gambar kejadian kecelakaan; melakukan panggilan darurat ke nomor 112 untuk melakukan laporan terkait terjadinya kecelakaan; dan menampilkan rute dari lokasi responden menuju lokasi korban. Rute menampilkan hasil algoritme Floyd-Warshall yang merupakan rute terpendek dari beberapa pilihan rute yang lain.

Untuk mengetahui kinerja aplikasi notifikasi kecelakaan lalu lintas ini, dilakukan beberapa pengujian menggunakan *smartphone* dengan sistem operasi Android versi 4. Pengujian aplikasi notifikasi kecelakaan yang dilakukan dibagi menjadi dua, yaitu sebagai berikut.

1) Pengujian Algoritme Floyd-Warshall untuk Penentuan Jarak Terpendek: Dilakukan perbandingan nilai matriks *Wij* awal sistem dengan matriks *Wij* awal yang dihitung secara manual. Apabila hasil matriks *W** akhir sistem sama dengan matriks *W** akhir yang dihitung secara manual, berarti hasil pengujian jalur dan jarak tempuh benar.

2) Pengujian Black Box dan Kecepatan Internet: Pengujian *black box* pada aplikasi Android dilakukan dengan melihat tampilan dan fungsi pada aplikasi tanpa memperhatikan struktur logika internal aplikasi. Parameter pengujian adalah kecepatan internet dan waktu yang dibutuhkan untuk mengirim notifikasi dari aplikasi korban ke responden.

A. Hasil Pengujian Algoritme Floyd-Warshall

Pengujian algoritme Floyd-Warshall dilakukan di kota Surabaya dengan mentransformasikan peta kota Surabaya yang menjadi objek penelitian ke dalam peta pada Android dengan menampilkan lokasi responden, lokasi korban, dan lokasi kantor polisi serta rumah sakit. Pengujian algoritme Floyd-Warshall ini menggunakan titik awal dan tujuan yang membentuk sebuah jaringan rute yang terdiri atas sembilan *node*, dengan empat persimpangan jalan. Untuk mendapatkan *node* dan jalur terpendek, digunakan matriks x_{ij} awal dan matriks z_{ij} awal. Matriks w_{ij} awal merupakan matriks untuk mencari jarak terpendek setiap pasangan *node*, sedangkan z_{ij} awal berguna untuk mencari jalur terpendek antartitik. Matriks w_{ij} awal dan z_{ij} awal yang didefinisikan sebagai $k = 0$ kemudian dikenai proses perhitungan dengan algoritme Floyd-Warshall untuk mencari jalur terpendek antara semua titik dalam jaringan tersebut. Setelah perhitungan dilakukan hingga iterasi terakhir, yaitu saat $k = 9$, $i = 9$, dan $j = 9$, diperoleh matriks x^* akhir yang ditunjukkan matriks z^* akhir hasil proses perhitungan algoritme Floyd-Warshall. Hasil ini menunjukkan jalur terpendek antar-*node*, seperti yang ditunjukan pada Tabel V. Hasil pemilihan rute terpendek dari algoritme Floyd-Warshall adalah T1 = 0-1-3-4-5-7-8, dengan jarak tempuh 4,199 km.

TABEL V
HASIL PEMILIHAN RUTE DARI ALGORITME FLOYD-WARSHALL

No.	Rute	Jarak (km)	Rute Terpilih
1	T0 = 0-1-3-6-5-7-8	4,398	T1 = 0-1-3-4-5-7-8
2	T1 = 0-1-3-4-5-7-8	4,199	
3	T2 = 0-1-2-4-5-7-8	5,614	

TABEL VI
HASIL PENGUJIAN *BLACK BOX*

No.	Nama Fitur	Jumlah Pengujian	Berhasil	*Error*
1	Splash Screen	20	20	0
2	Halaman Login	20	20	0
3	Halaman Registrasi	20	20	0
4	Halaman Home/Notifikasi	20	20	0
5	Rute Menuju Lokasi Korban	20	20	0
6	Ambil Gambar	20	20	0
7	Lihat Gambar	20	20	0
8	Call 112	20	20	0
9	Tombol Logout	20	20	0

B. Hasil Pengujian Algoritme Floyd-Warshall

Pengujian *black box* dilakukan pada setiap fitur yang terdapat pada aplikasi, seperti *register, login, home*, menerima notifikasi, *list* galeri data, tombol lihat rute, *emergency call, take photo, upload photo*, dan *logout.* Hasil pengujian *black box* dari dua puluh kali percobaan menyatakan bahwa 100% sistem bekerja dengan baik karena memperoleh hasil dua puluh kali sukses dan 0 gagal, seperti yang ditunjukan pada Tabel VI.

Pengujian kinerja sistem menggunakan koneksi internet dilakukan dengan mengirimkan notifikasi kepada sepuluh responden secara bersamaan. Data diambil pada waktu tertentu, yaitu pagi, siang, sore, dan malam hari. Kemudian, diambil sampel tiap 10 menit selama 60 menit percobaan. Dari sampel tersebut, dihitung rata-ratanya, sehingga diperoleh rata-rata koneksi internet tiap waktu pagi adalah 17,97 ms, waktu siang

TABEL VII
HASIL PENGUJIAN KINERJA INTERNET DALAM PENGIRIMAN NOTIFIKASI

No	Waktu Pengujian	Jam (WIB)	Jumlah Responden	Rata-rata Waktu (ms)
1	Pagi	08.00 – 09.00	10	17,97
2	Siang	12.00 – 13.00	10	9,86
3	Sore	16.00 – 17.00	10	76,84
4	Malam	20.00 – 21.00	10	6,76
Rata – rata				27,86

9,86 ms, waktu sore 76,84 ms, dan waktu malam hari 6,76 ms. Rata-rata koneksi internet tiap waktu tersebut kemudian digunakan untuk menghitung rata-rata waktu koneksi internet secara keseluruhan, sehingga didapatkan hasil rata-rata waktu sebesar 27,86 ms. Hasil perhitungan tersebut diperlihatkan pada Tabel VII.

IV. KESIMPULAN

Berdasarkan hasil dari pengujian yang telah dilakukan, dapat diambil kesimpulan sebagai berikut. Secara umum, telah berhasil dibuat aplikasi notifikasi kecelakaan menggunakan *smartphone* yang dapat melakukan klasifikasi kecelakaan dan mengirimkan pesan notifikasi secara *broadcast*. Responden dapat menerima notifikasi kecelakaan di sekitar posisi responden dan dapat melakukan perjalanan menuju lokasi kecelakaan sesuai rute yang telah ditampilkan oleh aplikasi responden. Hasil dari pengujian *black box* menunjukkan 100% fitur dari aplikasi berjalan dengan baik pada *platform* Android. Rute yang dihasilkan juga tidak mengalami penyimpangan dari skenario pengujian, yaitu 0-1-3-4-5-7-8 dan menghasilkan jarak 4,199 km sebagai jarak terpendek. Kecepatan rata-rata respons pengiriman notifikasi dari korban ke responden adalah 27,86 ms.

KONFLIK KEPENTINGAN

Penulis makalah dengan judul Penerapan Floyd-Warshall untuk Pencarian Rute Terpendek pada Aplikasi Notifikasi Kecelakaan Lalu Lintas menyatakan bahwa tidak terdapat konflik kepentingan.

KONTRIBUSI PENULIS

Metodologi, analisis formal penulisan, penyusunan draf asli, peninjauan dan penyuntingan, Haniah Mahmudah; perangkat lunak, M Fajar Ibrahim; validasi, Okkie Puspitorini; peninjauan, Nur Adi S dan Ari Wijayanti.

REFERENSI

[1] E. Lesmana, dkk., "The Application of Floyd-Warshall Algorithm in Solving Shortest Path Problem for Fire Evacuation System at High Rise Building (Case Study at eL Royale Hotel Bandung)," *9th Int. Conf. Ind. Eng., Oper. Manag.*, 2019, hal. 2169-2178.

[2] R. Bhoraskar, N. Vankadhara, B. Raman, dan P. Kulkarni, "Wolverine: Traffic and Road Condition Estimation Using Smartphone Sensors," *2012 Fourth Int. Conf. Commun. Syst., Netw. (COMSNETS 2012)*, 2012, hal. 1-6.

[3] P. Mohan, V.N. Padmanabhan, dan R. Ramjee, "TrafficSense: Rich Monitoring of Road and Traffic Conditions Using Mobile Smartphones," *Proc. 6th ACM Conf. Embed. Netw. Sensor Syst.*, 2008, hal. 323–336.

[4] S. Sharma dan S. Sebastian, "IoT Based Car Accident Detection and Notification Algorithm for General Road Accidents," *Int. J. Elect., Comput. Eng. (IJECE)*, Vol. 9, No. 5, hal. 4020-4026, Okt. 2019.

[5] S. Farhan, S. Andryana, dan N. Hayati, "Implementasi Bellman-Ford dan Floyd-Warshall dalam Menentukan Jalur Terpendek Menuju Universitas Nasional Berbasis Android," *J. Ilm. Penelit., Pembelaj. Inform. (JIPI)*, Vol. 5, No. 2, hal. 123-132, Des. 2020.

[6] F.W. Ningrum dan T. Andrasto, "Penerapan Algoritma Floyd-Warshall dalam Menentukan Rute Terpendek pada Pemodelan Jaringan Pariwisata di Kota Semarang," *J. Tek. Elekt.*, Vol. 8, No. 1, hal. 21–24, Jan.-Jun., 2016.

[7] K. Manaf, dkk., "Designing Futsal Match Finder Application with Floyd-Warshall," *J. Phys.: Conf. Ser.*, Vol. 1280, No. 2, hal. 1-8, 2019.

[8] V.A. Nawagusti, "Penerapan Algoritma Floyd Warshall dalam Aplikasi Penentuan Rute Terpendek Mencari Lokasi BTS (Base Tower Station) pada PT.GCI Palembang," *J. Nas. Teknol., Sist. Inf.*, Vol. 4, No. 2, hal. 81-88, 2018.

[9] B.I.A. Prasetyo dan A. Maslan, "Analisis Perbandingan pada Algoritma Bellman Ford dan Dijkstra pada Google Map," *Khazanah Ilmu Berazam*, Vol. 3, No. 2, hal. 337–349, 2020.

[10] N.R. Chopde dan M.K. Nichat, "Landmark Based Shortest Path Detection by Using A* and Haversine Formula," *Int. J. Innov. Res. Comput., Commun. Eng.*, Vol. 1, No. 2, hal. 298-302, Apr. 2013.

[11] Ramadiani, D. Bukhori, Azainil, dan N. Dengen, "Floyd-Warshall Algorithm to Determine the Shortest Path Based on Android," *2018 IOP Conf. Ser.: Earth Environ. Sci.*, Vol. 144, hal. 1-8, Mei 2018.

[12] K. Gutenschwager, S. Völker, A. Radtke, dan G. Zeller, "The Shortest Path: Comparison of Different Approaches and Implementations for the Automatic Routing of Vehicles," *Proc. 2012 Winter Simul. Conf. (WSC)*, 2012, hal. 3312-3323.

[13] M. Kairanbay dan H.M. Jani,"A Review and Evaluations of Shortest Path Algorithms," *Int. J. Sci., Technol. Res.*, Vol. 2, No. 6, hal. 99-104, Jun. 2013

[14] U.E. Chukwuka, J.A. Oladunjoye, dan S. Emmanuel, "A Study of Intelligent Route Guidance System Using Dijkstra's Heuristic Shortest Path Algorithm," *Int. J. Inf. Technol., Innov. Africa*, Vol. 13, No. 2, hal. 1-23, Apr. 2018.

[15] S. Sanan, L. Jain, dan B. Kappor, "Shortest Path Algorithm," *Int. J. Appl., Innov. Eng., Manag. (IJAIEM)*, Vol. 2, No. 7, hal. 316-320, Jul. 2013.

[16] M. Gł ąbowski, B. Musznicki, P. Nowak, dan P. Zwierzykowski, "Efficiency Evaluation of Shortest Path Algorithms," *AICT 2013: The Ninth Adv. Int. Conf. Telecommun.*, 2013, hal. 154-160.

[17] P. Charoenporn, "Comparison of Algorithms for Searching Shortest Path and Implementations for the Searching Routing System via Web Services," *Proc. 2018 the 8th Int. Workshop Comput. Sci., Eng. (WCSE 2018)*, 2018, hal. 516-520.

[18] R. Kampf, O. Stopka, L. Bartuška, dan K. Zeman, "Circulation of Vehicles as an Important Parameter of Public Transport Efficiency," *Proc. 19th Int. Conf. Transp. Means*, 2015, hal. 143-146.

[19] W. Xing, dkk., "An Improved Savings Method for Vehicle Routing Problem", *2016 2nd Int. Conf. Control Sci., Syst. Eng. (ICCSSE)*, 2016, hal. 1-4.

[20] Y.Z. Mehrjerdi, "A Multiple Objective Stochastic Approach to Vehicle Routing Problem," *Int. J. Adv. Manuf. Technol.*, Vol. 74, hal. 1149-1158, Sep. 2014.

[21] M. Stanojevic, B. Stanojevic, dan M. Vujoševic, "Enhanced Savings Calculation and Its Applications for Solving Capacitated Vehicle Routing Problem," *Appl. Math., Comput.*, Vol. 219, No. 20, hal. 10302-10312, Jun. 2013.