APLIKASI PENCARI RUTE OPTIMUM PADA PETA GUNA MENINGKATKAN EFISIENSI WAKTU TEMPUH PENGGUNA JALAN DENGAN METODE A* DAN BEST FIRST SEARCH

Rudy Adipranata[1], Andreas Handojo[2], Happy Setiawan[3]
[1,2]Teknik Informatika, Fakultas Teknologi Industri, Universitas Kristen Petra, Surabaya
[3]Alumni Teknik Informatika, Fakultas Teknologi Industri, Universitas Kristen Petra, Surabaya
Email: rudya@petra.ac.id[1], handojo@petra.ac.id[2]

ABSTRAK: Untuk menempuh perjalanan dari satu tempat ke tempat lain, peralatan yang paling sering digunakan adalah peta, dengan menggunakan peta, dapat diestimasikan rute yang optimum dengan menghitung panjang jalan yang akan ditempuh. Karena merupakan estimasi, maka dengan cara tersebut belum tentu rute yang dipilih merupakan rute yang optimal karena terdapat faktor lain seperti kemacetan atau kondisi jalan. Untuk itu pada penelitian ini dikembangkan aplikasi untuk penentuan rute yang optimum dengan menggunakan peta digital dimana pada aplikasi ini terdapat faktor kemacetan, kondisi jalan dan jarak tempuh. Untuk pencarian rute digunakan metode A* serta *Best First Search* yang menggunakan fungsi *heuristic* guna 'mengarahkan' pencarian pada peta yang direpresentasikan dalam konsep *graph*. Nilai *node-node graph* pada peta dapat diatur dengan fasilitas pengenalan warna pada peta. Dan dapat ditentukan pula bobot pada masing-masing node yang merupakan kemacetan atau kondisi jalan. Pada pengujian aplikasi ini, selain diuji untuk mencari rute optimum pada sebuah peta, juga dilakukan uji perbandingan antara metode A* dan *Best First Seach*. Dari hasil pengujian dapat disimpulkan bahwa metode A* memberikan hasil pencarian rute yang lebih pendek daripada BFS. Tingkat optimasi rute yang dihasilkan tergantung pada tersedianya data yang lengkap dan akurat tentang kondisi jalan serta proses pemberian bobot pada *node* peta yang mewakili kondisi jalan tersebut.

Kata kunci: pencarian rute, peta *digital*, A*, *Best First Search.*

***ABSTRACT**: Nowadays, most people still use conventional map to search for optimum route to travel from one place to another. By using map, people can estimate optimum route by calculate the length of street. However manual pathfinding using a conventional map requires a great deal of precision and takes a lot of time to be accomplished. Therefore in this research, we develop an application to generate the most optimum route on a digital map. In this application we also consider about the factor of traffic, roads condition and the distance. The methods that are used for pathfinding are A* and Best First Search (BFS), that use heuristic function to 'direct' the search on a graph-represented map. The cost on map's graph is given using map color recognition, also we can input the cost for traffic and roads condition. Based on application's testing results for the comparison between A* and BFS method, it can be concluded that A* provides better and more optimized solution than BFS. The optimum level of a route depends on the availability of an accurate and complete data about the roads condition and the process of setting the nodes cost on a given map that represents the actual condition of the roads.*

***Keywords**: pathfinding, digital map, A*, Best First Search.*

PENDAHULUAN

Pada kota metropolitan, transportasi adalah persoalan penting bagi masyarakat kota yang dinamis. Luasnya sebuah kota serta banyaknya jalan raya seringkali menyulitkan seseorang untuk mencari rute yang paling optimum, baik dari segi jarak maupun waktu tempuh untuk bepergian dari suatu tempat ke tempat lain di dalam kota. Hal ini diperparah dengan sering terjadinya kemacetan di berbagai tempat yang menyebabkan waktu tempuh semakin lama. Pada akhir-akhir ini pencarian rute optimum menjadi masalah yang semakin penting dipicu oleh kenaikan harga bahan bakar yang hampir dua kali lipat, sehingga masyarakat berusaha menempuh perjalanan secepat mungkin untuk dapat sampai ke tempat tujuan sehingga tidak banyak biaya yang terbuang untuk masalah transportasi ini.

Dibiayai oleh Direktorat Jendral Pendidikan Tinggi, Departemen Pendidikan Nasional sesuai dengan Surat Perjanjian Pelaksanaan Penelitian Nomor: 007/SP2H/PP/DP2M/III/2007, tanggal 29 Maret 2007

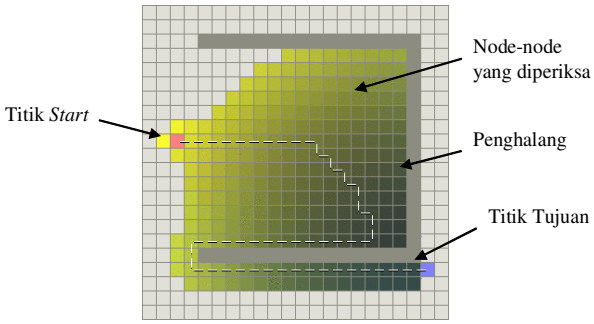
Untuk dapat memilih rute optimum, dapat digunakan peta kota konvensional untuk dapat melihat seluruh jalan raya yang terdapat pada kota dan kemudian memilih jalur terpendek dari tempat asal ke tujuan. Tetapi hal ini seringkali tidak dapat membantu secara maksimal karena banyaknya jalan raya yang ada sehingga menyebabkan banyaknya pilihan jalur yang dapat ditempuh. Serta dengan hanya melihat pada peta, sulit sekali untuk dapat secara langsung menentukan jalur tertentu adalah jalur yang ter-optimum karena kebanyakan terdiri dari jalan yang berkelok-kelok sehingga tidak dapat diperkirakan jarak tempuh pada jalur tersebut. Disamping itu, peta konvensional juga tidak dapat menunjukkan kondisi suatu jalan, tingkat kesulitan untuk dilalui, tingkat kemacetan dan sebagainya.

Tujuan rute optimum adalah mendapatkan waktu tempuh secepat mungkin dari tempat asal ke tujuan. Sehingga untuk menentukan rute optimum selain melihat jarak tempuh, juga harus memperhatikan komponen lain seperti tingkat kemacetan. Misal, terdapat dua rute yaitu rute pertama yang mempunyai jarak tempuh pendek tetapi terjadi kemacetan sehingga membutuhkan waktu tempuh satu jam, serta rute kedua yang mempunyai jarak tempuh lebih panjang tetapi tidak terjadi kemacetan sehingga membutuhkan waktu tempuh hanya setengah jam. Maka rute optimum adalah rute kedua karena waktu tempuh secara keseluruhan lebih cepat dari rute pertama. Dengan mendapatkan waktu tempuh yang tersingkat berarti konsumsi bahan bakar juga akan menjadi lebih sedikit dibanding rute yang mempunyai waktu tempuh lebih lama.

Untuk itu pada penelitian ini, akan dikembangkan aplikasi pencari rute optimum pada peta digital dengan tujuan meningkatkan efisiensi waktu tempuh pengguna jalan. Pencarian rute paling optimum dilakukan dengan menggunakan metode A* (*A Star*) dan *Best First Search* (BFS) yang diimplementasikan pada peta digital Surabaya Pusat. Pada pengujian selain dilakukan pencarian rute optimum, juga dilakukan perbandingan antara kedua metode yang digunakan.

ALGORITMA *BEST FIRST SEARCH*

Best First Search (BFS) adalah algoritma *pathfinding* yang menggunakan menggunakan fungsi *heuristic* untuk 'menuntun' pencarian solusi, dengan menghitung berapa jauh *goal* dari *node* sekarang [1,5]. Dari sekian pilihan jalan, dipilih jalan yang paling dekat ke tujuan. BFS tidak menjamin menemukan rute yang paling pendek, namun algoritma ini bekerja dengan sangat cepat. Dalam gambar berikut, warna kotak terang mempunyai biaya/*cost* tinggi karena jauh dari tujuan, dan kotak gelap mempunyai biaya kecil karena dekat dengan *goal*. Dapat dilihat bahwa algoritma ini bekerja dengan cepat menemukan *goal* dengan mengembangkan kemungkinan jalan yang mengarah ke tujuan.



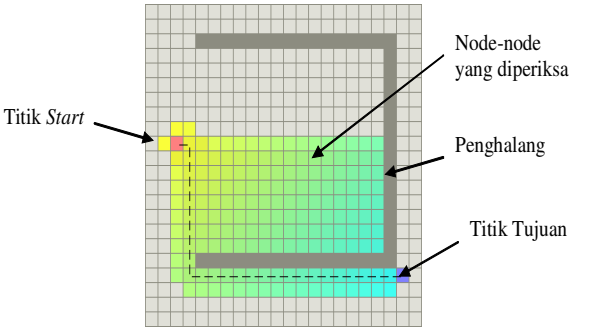
Gambar 1. *Pathfinding* Algoritma BFS [3]

Masalah utama pada algoritma *pathfinding* BFS ini adalah cara kerjanya yang terus mencoba bergerak ke arah tujuan/*goal*, meskipun jalan yang dilaluinya bukan jalan yang benar. Karena BFS hanya memperhitungkan *cost* untuk mencapai *goal* dan mengabaikan *cost* jalan yang sudah ditempuhnya untuk sampai ke *current state*, maka algoritma tersebut terus maju meskipun jalan yang telah dilaluinya sudah terlalu panjang [1].

ALGORITMA A* (*A STAR*)

Algoritma A* (*A Star*) adalah algoritma *pathfinding* pengembangan dari algoritma *Best First Search*. Seperti halnya pada BFS, untuk menemukan solusi, A* juga 'dituntun' oleh fungsi *heuristic* [3,4,5].

Pada pencarian rute kasus sederhana, dimana tidak terdapat halangan pada peta, A* bekerja secepat dan seefisien BFS. Pada kasus peta dengan halangan, seperti pada gambar di bawah ini, A* dapat menemukan solusi rute tanpa 'terjebak' oleh halangan yang ada. Ilustrasinya dapat dilihat pada Gambar 2.



Gambar 2. *Pathfinding* Algoritma A* [3]

Perbedaan cara kerja A* dengan BFS adalah selain memperhitungkan *cost* dari *current state* ke tujuan dengan fungsi *heuristic* (seperti BFS), algoritma ini juga mempertimbangkan *cost* yang telah ditempuh selama ini dari initial *state* ke *current state*. Jadi bila jalan yang telah ditempuh sudah terlalu panjang dan ada jalan lain yang lebih kecil *cost*-nya namun memberikan posisi yang sama dilihat dari *goal*, jalan baru yang lebih pendek itulah yang akan dipilih [1,3]. Notasi yang dipakai oleh algoritma A* ini:

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

Dalam notasi standar yang dipakai untuk algoritma A* di atas, digunakan g(n) untuk mewakili *cost* rute dari *node* awal ke *node* n. Lalu h(n) mewakili perkiraan *cost* dari *node* n ke *node goal*, yang dihitung dengan fungsi *heuristic*. A* 'menyeimbangkan' kedua nilai ini dalam mencari jalan dari *node* awal ke *node goal*. Dalam menentukan *node* yang akan dikembangkan, algoritma ini akan memilih *node* dengan nilai f(n) = g(n) + h(n) yang paling kecil.

FUNGSI *HEURISTIC*

BFS dan A* sebagai algoritma pencarian yang menggunakan fungsi *heuristic* untuk 'menuntun' pencarian rute, khususnya dalam hal pengembangan dan pemeriksaan *node-node* pada peta [5].

Dalam aplikasi ini, fungsi *heuristic* yang dipakai untuk pencarian rute mengisi nilai/notasi h pada algoritma BFS dan A*. Ada beberapa fungsi *heuristic* umum yang bisa dipakai untuk algoritma BFS dan A* ini. Salah satunya adalah yang dikenal dengan istilah '*Manhattan Distance*'. Fungsi *heuristic* ini digunakan untuk kasus dimana pergerakan pada peta hanya lurus (horisontal atau vertikal), tidak diperbolehkan pergerakan diagonal.

Perhitungan nilai *heuristic* untuk *node* ke-n menggunakan *Manhattan Distance* adalah sebagai berikut :

$$h(n) = D * (abs(n.x-goal.x) + abs(n.y-goal.y)) \quad (2)$$

Dimana h(n) adalah nilai *heuristic* untuk node n, D adalah nilai/cost perpindahan dari satu node ke node lain (secara vertikal/horizontal), dan goal adalah node tujuan. D dapat berupa bilangan real ataupun bilangan bulat, positif maupun negatif, selama tidak sama dengan nol. Sebab jika D bernilai nol, maka h(n) juga bernilai nol.

Namun karena pada penelitian ini pergerakan diagonal pada peta diperbolehkan, maka digunakan fungsi *heuristic* lain selain *Manhattan Distance*. Untuk mendekati kenyataan, *cost* untuk perpindahan *node* secara diagonal dan orthogonal dibedakan. *Cost* diagonal adalah 1,4 kali *cost* perpindahan secara orthogonal.

Fungsi *heuristic* yang digunakan adalah sebagai berikut:

$$h\_diagonal(n) = \min(abs(n.x-goal.x), abs(n.y-goal.y)) \quad (3)$$

$$h\_orthogonal(n) = (abs(n.x-goal.x) + abs(n.y-goal.y)) \quad (4)$$

$$h(n) = D2 * h\_diagonal(n) + D * (h\_orthogonal(n) - 2 * h\_diagonal(n))) \quad (5)$$

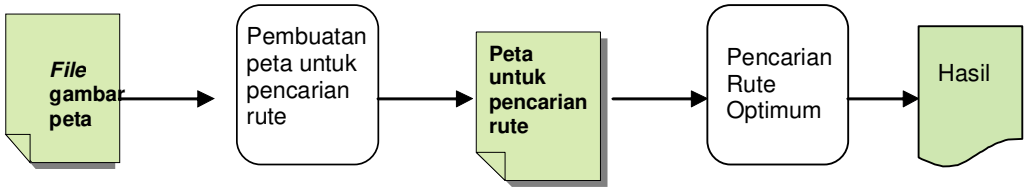
Dimana h_diagonal(n) adalah banyaknya langkah diagonal yang bisa diambil untuk mencapai goal dari node n. h_orthogonal adalah banyaknya langkah lurus yang bisa diambil untuk mencapai goal dari node n. D adalah nilai/cost perpindahan node secara orthogonal dan D2 adalah cost perpindahan node secara diagonal. D dan D2 dapat berupa bilangan real ataupun bulat, positif maupun negatif selama tidak bernilai nol.

Nilai *heuristic* kemudian diperoleh dari D2 dikalikan dengan h_diagonal(n) ditambah dengan D dikali dengan selisih h_orthogonal(n) dengan dua kali h_diagonal(n). Dengan kata lain, jumlah langkah diagonal kali cost diagonal ditambah jumlah langkah lurus yang masih bisa diambil dikali cost pergerakan lurus.

PERANCANGAN SISTEM

Blok diagram sistem keseluruhan dapat dilihat pada Gambar 3.

Sesuai gambar blok diagram pada gambar 3, sebelum dapat dilakukan pencarian rute pada peta, harus tersedia peta yang sudah diberi bobot sesuai kebutuhan dan ketentuan aplikasi ini, yang diperlukan untuk pencarian rute. Bila belum ada, pengguna dapat membuat dahulu peta yang diperlukan dari *file* peta dengan format bitmap 24 bit. Langkah selanjutnya adalah memberikan bobot pada *node-node* yang terletak di peta tersebut.



Gambar 3. Blok Diagram Sistem

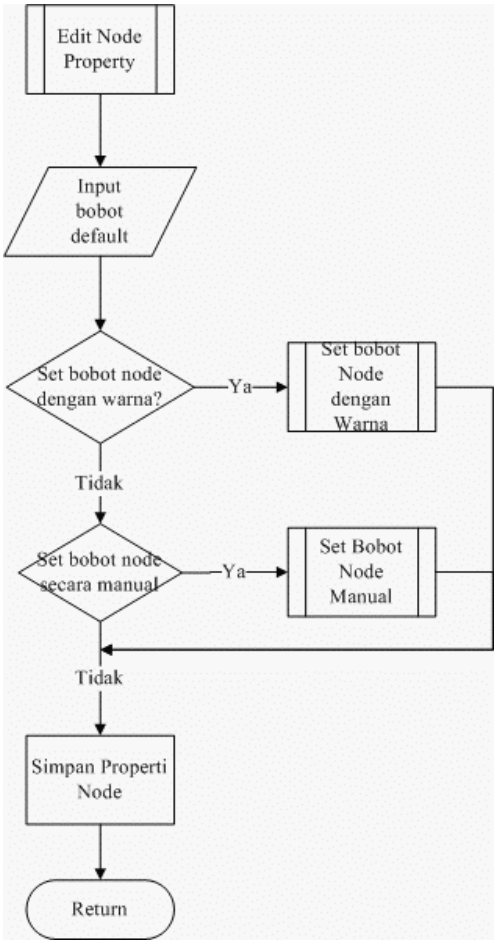
Pengaturan Bobot *Node* Peta

Pengaturan bobot *node-node* pada peta dapat dikatakan adalah salah satu hal paling penting dalam aplikasi ini. Pemberian bobot ini sangat berpengaruh pada ketepatan pencarian rute optimal pada peta yang bersangkutan. Pada pengaturan bobot node peta ini, pengguna dapat mengatur bobot *default node* peta. Selanjutnya pengguna dapat mengatur bobot *node* berdasarkan warna *node* tersebut pada peta. Misalkan apabila dalam peta jalan raya berwarna putih, maka pengguna dapat mengatur supaya aplikasi memberi *node-node* yang berwarna putih bobot yang telah ditentukan pengguna. Untuk jalan yang dapat dilalui pada peta, sebaiknya bobotnya diset kecil. Sedangkan untuk daerah yang tidak dapat dilewati, bobot diset besar.

Selanjutnya pengguna dapat mengatur pemberian bobot dan pengaturan properti *node* lainnya kepada *node-node* yang dipilihnya secara manual. Pengaturan properti *node* lain meliputi pemberian nama jalan, kondisi jalan dan arah jalan.

Flowchart untuk pengaturan bobot dan properti lain node pada peta dapat dilihat pada Gambar 4.

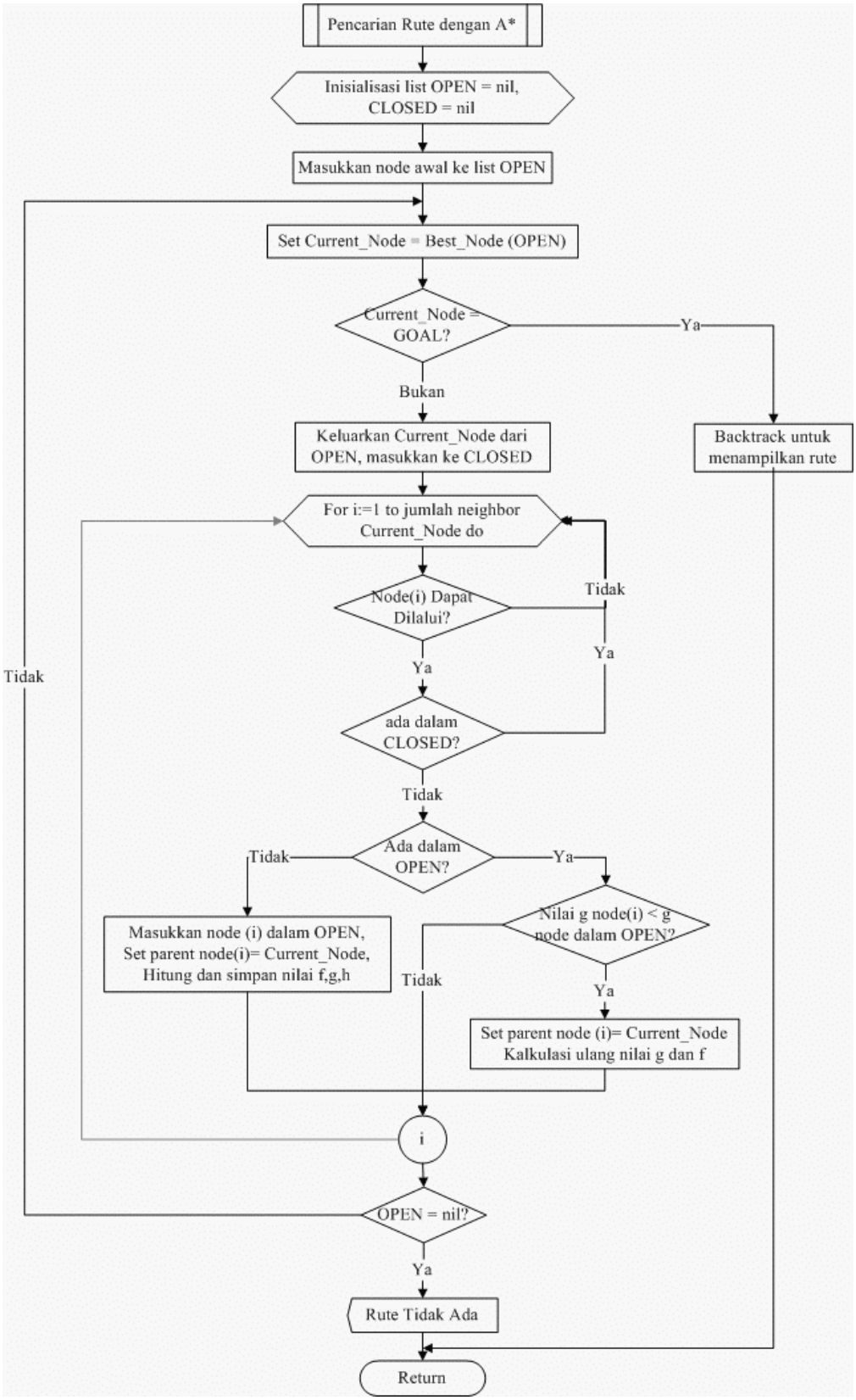
Setelah dilakukan pengaturan bobot *node* pada peta, barulah aplikasi dapat menjalankan algoritma pencarian rute baik A* ataupun *Best First Search*.



Gambar 4. *Flowchart* Pengaturan Bobot *Node* Peta

Desain Aplikasi Untuk Algoritma A*

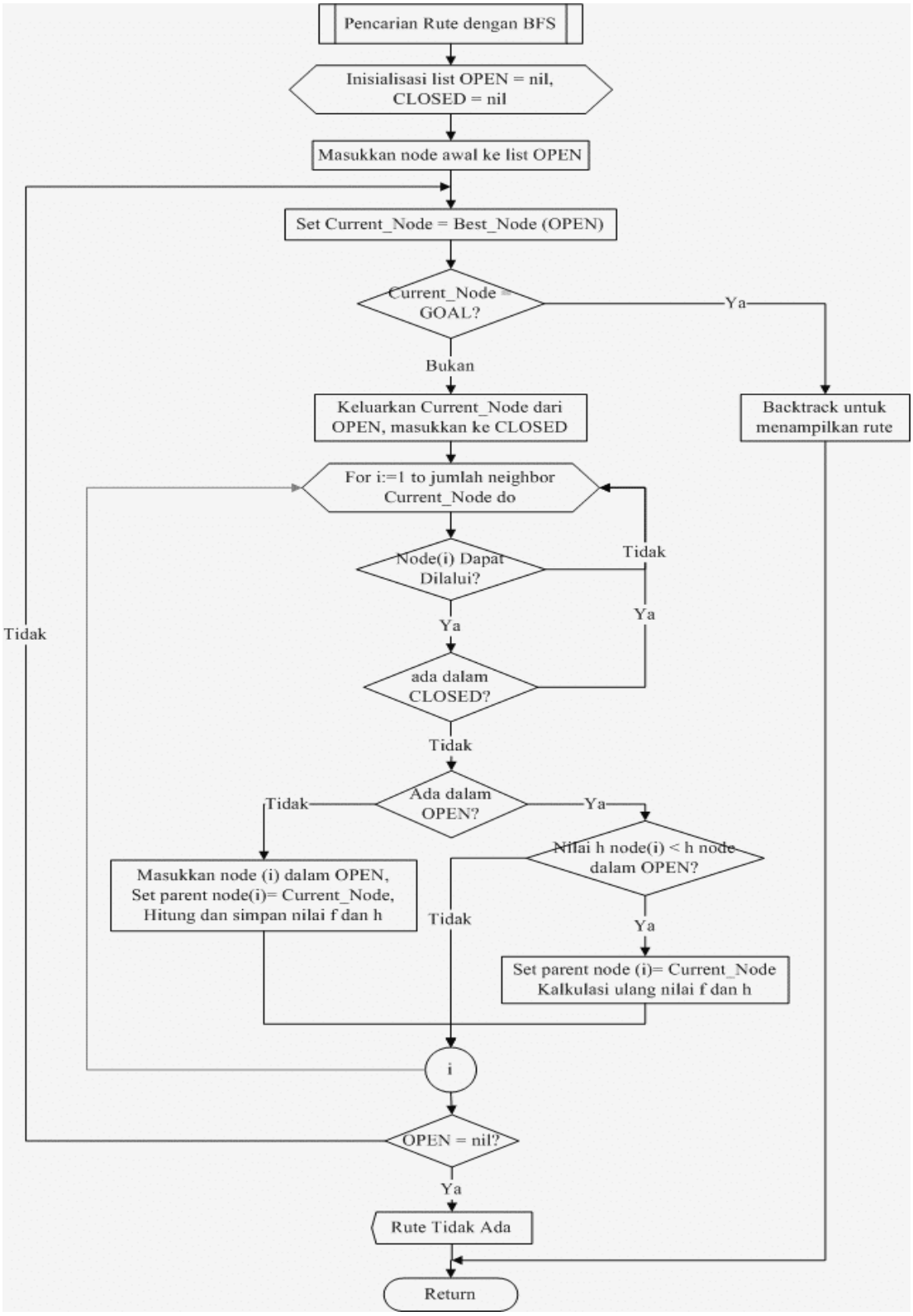
Berikut adalah *flowchart* alur algoritma pencarian rute dengan metode A*.



Gambar 5. *Flowchart* Algoritma A*

Desain Aplikasi Untuk Algoritma *Best First Search*

Pada *flowchart* algoritma BFS di bawah ini, dapat dilihat bahwa hal yang membedakan BFS dengan A*, algoritma BFS tidak menghitung nilai g sebuah *node* (*cost* yang dibayar untuk sampai ke *node* itu dari titik awal).



Gambar 6. *Flowchart* Algoritma *Best First Search*

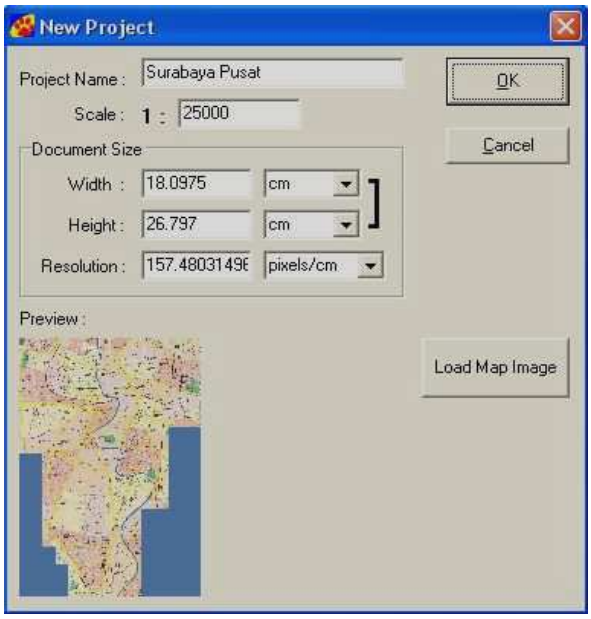
IMPLEMENTASI DAN HASIL PENGUJIAN

Aplikasi dikembangkan dengan menggunakan bahasa pemrograman Delphi dan berikut ini ditampilkan hasil implementasi serta pengujian yang dilakukan. Aplikasi dibuat dengan sistem MDI, dimana aplikasi dapat membuka dan mengedit banyak dokumen sekaligus. Form utama dapat membuat form 'anak' (*child form*) baru untuk membuka atau membuat dokumen baru. Berikut ini adalah tampilan menu utama aplikasi.

Gambar 7. Menu Utama Aplikasi

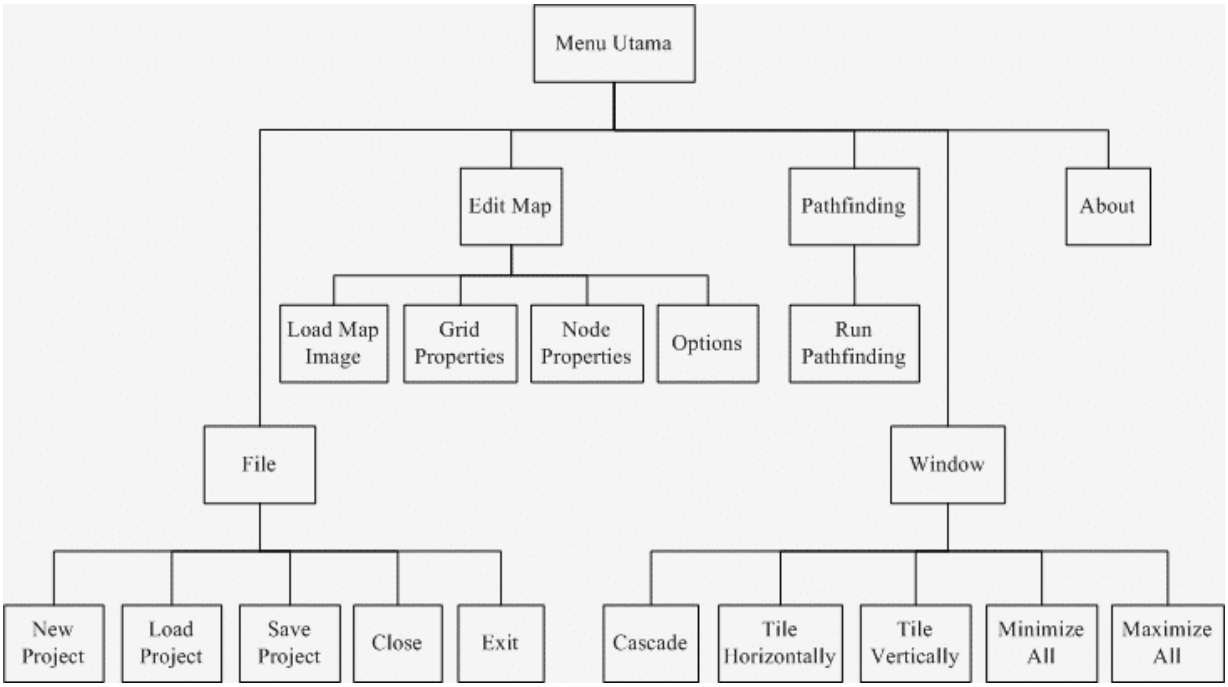
Untuk lebih jelas, bagan menu utama aplikasi ini dapat dilihat pada Gambar 8.

Untuk pengujian, dilakukan terlebih dahulu dilakukan *load* peta dimana pengujian dilakukan pada peta Surabaya Pusat [2].



Gambar 9. Pembuatan *Project* Peta Baru

Setelah melakukan *load* peta, dilakukan pengaturan *grid* untuk membentuk *node-node* pada peta serta pemberian bobot pada masing-masing *node* seperti Gambar 9.



Gambar 8. Bagan Menu Utama

Pada pengujian pencarian rute optimum ini dilakukan sepuluh kali percobaan menggunakan metode A* dan BFS. Peta yang digunakan adalah peta Surabaya Pusat, dengan dimensi peta 2850 x 4220 *pixel*, skala peta 1 : 25000 dan resolusi 157,5 *pixel* per cm. Pengujian dilakukan pada PC dengan *processor* Pentium 4 1.6 GHz, *memory* 512 MB DDR, dengan sistem operasi Windows XP Professional.

Pengujian yang dilakukan mencatat empat kategori perbandingan, yakni:

- Jarak rute yang ditemukan: perkiraan panjang rute yang ditemukan dalam satuan kilometer.
- Waktu pencarian rute: lama waktu yang diperlukan untuk melakukan proses pencarian.
- Total *path cost*/biaya tempuh rute: biaya tempuh rute dari *node* awal ke *node* akhir, dihitung berdasarkan akumulasi bobot (nilai g) *node-node* yang dilewati rute yang ditemukan.
- Jumlah *node* yang dikembangkan : banyaknya *node* pada peta yang dikembangkan dan diperiksa oleh metode pencarian untuk menemukan *node* tujuan.

Pengujian untuk keempat kategori perbandingan tersebut dicatat dalam Tabel 1.

Dari Tabel 1 dapat disimpulkan bahwa untuk perbandingan jarak rute yang ditemukan dan total *path cost*/biaya tempuh rute, metode A* lebih baik daripada metode BFS, namun untuk perbandingan lama waktu proses pencarian rute dan jumlah *node* yang dikembangkan, metode BFS jauh lebih cepat dan mengembangkan *node* lebih sedikit dari A*.

KESIMPULAN

Dari pembuatan dan pengujian aplikasi pencarian rute optimum pada peta dengan metode A* dan BFS, dapat diambil kesimpulan sebagai berikut:

- Pencarian rute pada peta menggunakan metode A* maupun BFS selalu berhasil menemukan solusi rute apabila memang terdapat jalan dari titik awal ke tujuan.
- Untuk aplikasi pencarian rute optimum pada peta, metode A* lebih baik daripada BFS karena dari segi jarak hasil rute yang ditemukan dan total *path cost*/biaya tempuh rute, metode A* memberikan hasil yang lebih baik daripada BFS, sedangkan metode BFS memberikan hasil rute yang lebih panjang dan *path cost* yang lebih besar.
- Kelebihan metode BFS adalah dalam hal kecepatan proses pencarian rute yang jauh lebih cepat dan lebih sedikitnya jumlah *node* yang dikembangkan untuk pencarian rute daripada menggunakan metode A*.
- Lamanya proses pencarian dan banyaknya node yang dikembangkan untuk pencarian rute dengan kedua metode, khususnya metode A*, tergantung pada besar kecilnya ukuran peta dan grid peta serta jarak antara node awal dan node tujuan pada peta.
- Penggunaan pembagian node-node peta berdasarkan kotak grid peta memiliki keuntungan mempermudah dan mempercepat proses pemberian bobot node serta mempermudah *interface* pencarian rute dengan pengguna aplikasi, dimana pengguna dapat memilih lokasi dimana saja pada peta. Namun kerugian metode ini, jumlah node

Tabel 1. Tabel Pengujian Perbandingan Metode Pencarian Rute.

	Titik awal		Titik tujuan			A*				Best First Search			
No	X	Y	X	Y	Dist.	Time	Nodes	Cost	Jarak	Time	Nodes	Cost	Jarak
1	560	199	168	208	0.63	0	79	6308	0.68	0	140	6540	0.69
2	560	199	284	1012	1.37	6.34	7099	30972	2.48	0.02	865	41340	2.55
3	876	242	1089	1338	1.77	21.84	10674	40372	2.98	0.08	1614	53936	3.69
4	1711	94	1009	1771	2.89	0.031	1612	43744	3.52	0.11	684	47184	3.41
5	675	2518	2343	145	4.6	74.95	19269	76148	5.73	0.05	1098	122100	8.14
6	1198	3420	2213	224	5.33	7.34	5973	86960	6.59	0.08	1397	123824	8.61
7	1598	3798	805	350	5.56	8.87	5246	90952	6.91	0.77	2547	194856	12.04
8	1885	3803	201	216	6.29	42.95	13206	100564	8.15	0.97	2960	154592	10.34
9	2275	35	789	4034	6.77	41.08	11980	110200	8.08	0.18	1763	180336	11.41
10	17	182	1809	4177	6.95	236.2	36847	121612	9.63	0.94	2705	195972	12.56

Keterangan :
Dist. : jarak titik tujuan dari titik awal diukur dengan garis lurus, dalam km.
Jarak : perkiraan panjang rute dalam satuan kilometer.
Time : lama waktu pencarian rute dalam hitungan detik (*second*).
Cost : total *path cost*/bobot total dari *node* awal ke *node* akhir.
Nodes : jumlah *node* yang diperiksa dan dikembangkan dalam proses pencarian rute.

yang tercipta banyak, apalagi untuk peta dengan ukuran besar, yang berakibat pada lamanya proses pencarian rute dengan metode A* dan besarnya ukuran file *project* yang disimpan pada komputer.

DAFTAR PUSTAKA

1. Russel, Stuart J and Peter Norvig. *Artificial Intelligence: A Modern Approach*. New Jersey: Prentice Hall, 2003.
2. Le Moullec, Marc. *Surabaya Atlas Jalan & Index*. Jakarta: P.T. Enrique Indonesia, 1999.
3. "AI Game Programming." *Amit's Game Programming Site*. 9 October 2003. <http://theory.stanford.edu/~amitp/GameProgramming>.
4. "A Star Pathfinding for Beginners." *Policyalmanac*. 21 April 2004. <http://www.policyalmanac.org/games>.
5. Dechter, Rina; Judea Pearl. *Generalized best-first search strategies and the optimality of A*. Journal of the ACM* 32 (3): pp. 505 – 536. 1985.
6. Pearl, Judea. *Heuristics: Intelligent Search Strategies for Computer Problem Solving*. Addison-Wesley. 1984.