Penerapan Algoritma A dalam Perencanaan Pembangunan Jalur Tol*

Farhan Ramadhan Syah Khair / 13517001
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
farhanramadhansk@gmail.com

***Abstrak*—Jalur tol, merupakan salah satu infrastruktur yang penting karena dengan adanya jalur tol, maka waktu perjalanan antara dua kota dapat ditempuh lebih cepat. Jalur tol juga berguna agar konektivitas antara kota ataupun desa dapat terbentuk dan dapat meratakan pertumbuhan penduduk. Pembangunan jalur bebas hambatan ini terdapat banyak masalah, dimulai dari faktor alam, faktor sosial, dan lain-lain. Oleh karena itu pada makalah ini akan dijelaskan bagaimana algoritma A* (A-star) dapat membantu menyelesaikan masalah mengenai pembangunan jalur bebas hambatan.**

Keywords—Jalur Tol, Jarak Terpendek, Algoritma A*

I. PENDAHULUAN

Jalur Tol adalah suatu jalan/jalur yang dikhususkan untuk kendaraan bersumbu dua atau lebih dan bertujuan untuk mempersingkat waktu dan jarak tempuh dari satu tempat ke tempat lain. Jika ingin menggunakan jalur bebas hambatan maka kita harus membayar sesuai dengan tarif yang berlaku. Penerapan tarif ini didasarkan pada golongan kendaraan. Pembayaran tarif dilakukan di fasilitas yang berada di jalur tol yaitu gerbang tol. Jalan tol ada di Indonesia pertama kali pada tahun 1978[3].

Di Indonesia, banyak orang mempunyai anggapan bahwa jalur tol itu sama dengan jalur bebas hambatan padahal kedua hal tersebut adalah hal yang berbeda. Jalur tol merupakan jalur yang dikhususkan dan berbayar, sedangkan jalur bebas hambatan sendiri merupakan jalur yang dikhususkan akan tetapi tidak berbayar.

Jalur tol memilliki keunggulan ketimbang jalan biasa , yaitu jalur tol hanya dapat digunakan oleh kendaraan bersumbu dua atau lebih yang berarti tidak akan ada sepeda motor yang melewati jalur tol sehingga dapat mengurangi kemacetan, jalur tol juga cocok untuk orang yang tidak ingin membuang waktu di jalan, jalur tol sendiri memiliki batasan kecepatan yang lebih tinggi ketimbang jalur biasa, jalur tol juga idealnya tidak memiliki hambatan atau kemacetan. Akan tetapi jalur tol juga memiliki kelemahan, yaitu jalur tol merupakan jalur berbayar, jalur tol juga tidak selalu lancar.

Pembangunan jalur tol bukanlah hal yang mudah , bukan sekedar mencari jarak terpendek dari kota satu ke kota lain , kita juga harus memperhitungkan faktor alam dan faktor sosial yang ada selama pembangunan. Pembangunan jalur tol juga tidak boleh seenaknya karena jalur tol harus dapat bertahan atau awet untuk jangka waktu yang lama dan juga kendaraan yang melintasi jalur tol juga memiliki kecepatan yang tinggi oleh karena itu faktor keselamatan pengguna jalan tol juga harus diperhatikan secara seksama.

Pada perencanaan pembangunan jalur tol haruslah cepat dan efisien karena jalur tol sendiri merupakan infrastruktur yang dibutuhkan secepatnya oleh karena itu perencanaan pembangunan jalur tol harus efisien akan tetapi tetap menghasilkan yang terbaik. Jalur tol haruslah memiliki jarak tempuh yang paling dekat. Oleh karena itu ketika merencanakan pembangunan jalur tol, kita harus bisa memetakan jalur yang paling dekat dengan penelusuran paling minimum. Oleh karena itu perencanaan tersebut dapat dibantu menggunakan algoritma A*.

Algoritma ini dapat menyelesaikan permasalahan ini karena permasalahan jalur tol ini merupakan permasalahan perencanaan jalur terpendek ynag dapat dibentuk dari kota satu ke kota lainnya tanpa melewati suatu halangan. Halangan ini dapat berupa gunung, rawa-rawa, danau, tanah penduduk dan lain-lain. Algoritma A* juga memberikan akurasi yang cukup tinggi, tergantung bagaimana perkiraan si penghitung.

II. DASAR TEORI

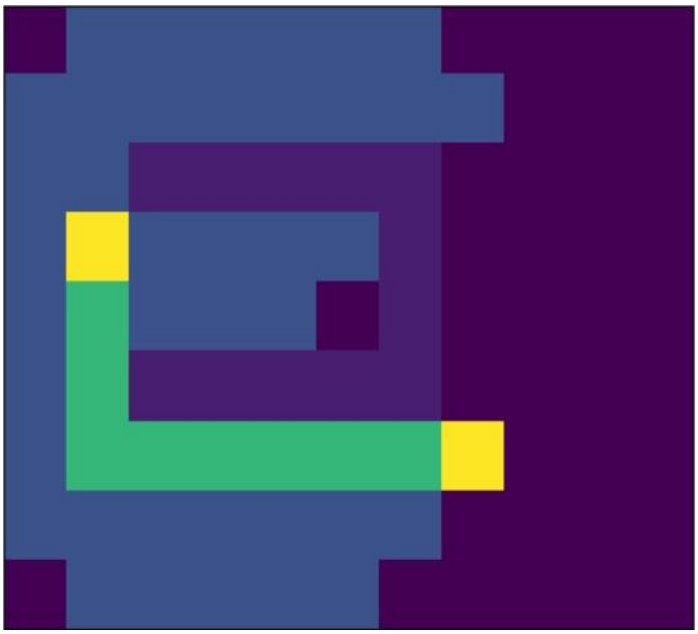
*A. Algoritma A**

Algoritma A* adalah salah satu algoritma yang sering digunakan untuk melakukan penelusuran graf ataupun pencarian jalur. Algoritma A* juga termasuk ke dalam Informed Search[1]. Algoritma ini digunakan untuk mencari jalur dengan jarak terpendek dari satu tempat ke tempat tujuan yang dimana terdapat jalan yang tidak dapat dilalui pada jalan tersebut.

Algoritma ini mengembangkan teknik yang digunakan pada Algoritma Djikstra dengan mengambil beberapa karakteristik *Greedy BFS*. Jika algoritma Djikstra membutuhkan waktu dan sumber daya yang besar untuk menjelajahi semua jalur yang ada dan mencari titik terpendek, meskipun dapat mendapatkan jalur yang terpendek tapi sumber daya yang dikeluarkan harus besar. *Greedy BFS*[1] hanya menelusuri jalur dengan jarak terpendek saja, akan tetapi pencarian ini belum tentu dapat

menghasilkan solusi yang paling optimal karena setiap langkahnya hanya memilih yang palling optimal di tiap langkah tanpa memperhatikan optimal global. Ada juga Algoritma BFS yang dapat menghasilkan hasil yang pasti optimal akan tetapi penelusurannya sangatlah naif dan dapat dibilang brute-force sehingga algoritma BFS digunakan hanya sebagai pembanding hasil dengan hasil algoritma A*.

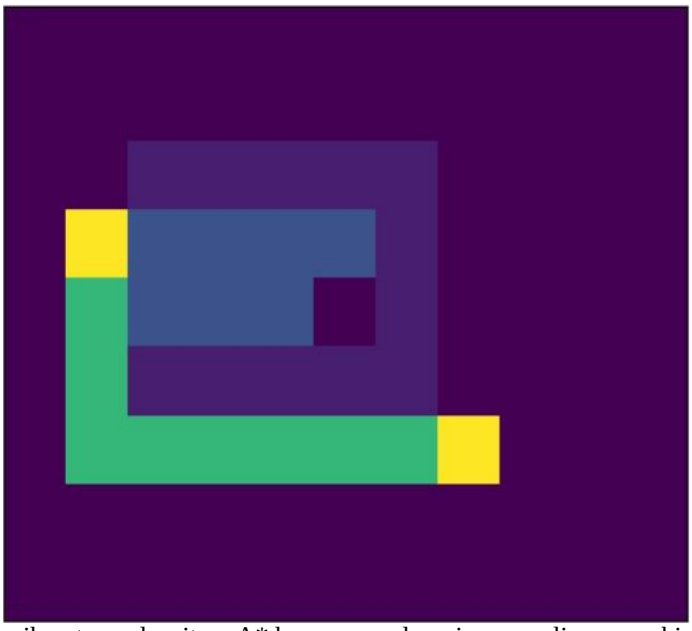
Kita misalkan terdapat sebuah peta yang direpresentasikan sebagai gambar 2D, dengan warna yang menggambarkan jalur atau simpul. Dengan Algoritma BFS bekerja dengan cara mencari semua kemungkinan dan mengembalikan hanya yang terpendek saja, BFS tidak menggunakan fungsi heuristik sehingga BFS dapat mengembalikan hasil yang optimal, akan tetapi dengan sumber daya yang banyak dan juga waktu yang digunakan juga besar.



Gambar 1. Pencarian Jalur Terpendek dengan Algoritma BFS

Pada Gambar 1 diatas dapat dilihat warna ungu muda menandakan halangan yang tidak dapat dilewati, warna biru menunjukan jalur yang telah dilalui oleh Algoritma BFS, warna kuning merupakan titik awal dan titik akhir, dan warna Hijau menunjukan Jalur yang terbentuk. Dapat dilihat bahwa Algoritma BFS menelusuri hampir semua jalur sampai menemukan titik tujuan.

Algoritma A* yang mengabgung antara *Greedy BFS* dan BFS biasa sehingga dapat menelusuri semua kemungkinan dengan memperhitungkan jarak aproksimasi dari titik awal ke titik akhir sehingga dapat mencari jalur yang paling optimal. Dengan mencari nilai minimum dari fungsi heuristik tiap titik yang ditempuh pada setiap langkah, akan didapat jalur teroptimal dengan waktu penelusuran yang lebih cepat karena



ibaratnya algoritma A* hanya menelusuri yang paling mungkin "menguntungkan" tanpa melupakan solusi optimal global.

Gambar 2. Pencarian Jalur Terpendek dengan Algoritma A*

Pada Gambar 2 diatas dapat dilihat warna ungu muda menandakan halangan yang tidak dapat dilewati, warna biru menunjukan jalur yang telah dilalui oleh Algoritma A*, warna kuning merupakan titik awal dan titik akhir, dan warna Hijau menunjukan Jalur yang terbentuk. Dapat dilihat bahwa Algoritma A* menelusuri jalur sampai menemukan titik tujuan.

Dari 2 Gambar di atas dapat disimpulkan bahwa Algoritma A* menelusuri peta dengan langkah yang lebih sedikit dan menghasilkan hasil yang sama, sama optimal, hal tersebut dapat disimpulkan bahwa algoritma A* lebih efisien ketimbang algoritma BFS biasa.

Hal tersebut karena pada algoritma A* kita dapat memprediksi langkah mana yang menurut kita paling bagus tanpa melupakan solusi optimum global.

Hal ini disebabkan oleh Fungsi heuristik yang digunakan dalam algoritma A* yang berasal dari penjumlahan antara nilai fungsi g(n) yang menandakan biaya yang dibutukan untuk mencapai simpul yang sedang dijalankan dari simpul awal dan nilai fungsi h(n) yang menandakan fungsi aproksimasi dalam memperkrakan biaya yang dibutuhkan untuk menemukan simpul tujuan dari simpul yang sedang diproses. Nilai dari fungsi heuristik f(n) untuk algortima A* dituliskan dalam ekspresi matematik sebagai persamaan berikut

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

Ketepatan pada algoritma A* sangat bergantung kepada bagaimana fungsi heuristik-nya dirancang dan digunakan. Oleh karena itu, fungsi heuristik harus ditentukan terlebih dahulu sebelum melakukan perancangan algortima.

B. *Fungsi Heuristik*

Fungsi heuristik adalah fungsi yang digunakan untuk mengevaluasi kandidat solusi yang dipilih dalam suatu proses algoritma A*.

Fungsi ini akan mengatur bagaimana hasil dari pencarian mengunakan algoritma A*.

Pada Persamaan (1), kita dapat membuat proses A* lebih akurat. Jika nilai h(n) adalah 0, maka persamaan (1) dapat ditulis sebagai

$$f(n) = g(n)$$

dengan nilai fungsi g(n) adalah *cost* terkecil yang dibutuhkan untuk mencapai titik yang sedang diproses. Akibatnya, Algoritma A* akan berperan seperti algoritma BFS, yang akan menjamin bahwa jalur yang kita tentukan pasti efektif dengan usaha yang sangat besar.

Jika nilai h(n) menjadi sangat besar dibandingkan dengan nilai g(n), maka persamaan (1) dapat ditulis sebagai

$$f(n) = h(n)$$

dengan nilai h(n) adalah fungsi heuristik yang dipilih. Akibatnya algoritma A* akan berperan seperti *Greedy BFS*. Sehingga pemrosesan lebih cepat akan tetapi hasil tidak selalu optimal.

C. *Jalur Tol*

Jalur Tol adalah suatu jalan/jalur yang dikhususkan untuk kendaraan bersumbu dua atau lebih dan bertujuan untuk mempersingkat waktu dan jarak tempuh dari satu tempat ke tempat lain. Jika ingin menggunakan jalur bebas hambatan maka kita harus membayar sesuai dengan tarif yang berlaku. Penerapan tarif ini didasarkan pada golongan kendaraan. Pembayaran tarif dilakukan di fasilitas yang berada di jalur tol yaitu gerbang tol. Jalan tol ada di Indonesia pertama kali pada tahun 1978[3].

Gambar 3. Gambar Jalur Tol[4]

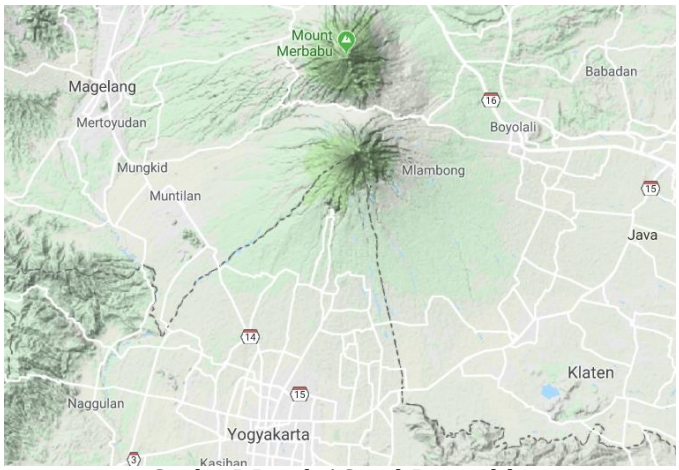
Jalur tol di Indonesia sendiri sudah berkembang sangat pesat, terutama di Pulau Jawa, akan tetapi belum semua kota sudah terhubung dengan jalur tol sehingga perkembangan jalur tol di Indonesia masih dapat dikembangkan lagi apalagi pengembangan jalur tol di luar Pulau Jawa.

Gambar 4. Gambar Jalur Tol di Pulau Jawa[5]

III. PERANCANGAN ALGORITMA

Untuk merancang algoritma pada persoalain ini, ada tahapan yang harus dilakukan terlebih dahulu. Pertama, kita harus memetakan kondisi medan, medan yang dapat dilalui ,dan medan yang tidak dapat dilalui. Kita juga harus menentukan fungsi heuristik yang cocok untuk digunakan dalam memperkirakan jarak antara satu titik dengan titik yang lain dalam mengevaluasi kandidat titik yang menjadi solusi. Setelah itu, algoritma baru dapat disusun untuk permasalahan ini.

Untuk menggambarkan permasalahan ini, akan digunakan peta yang mencakup, Kota Magelang sebagai titik asal dengan Kab. Klaten sebagai titik akhir.



Gambar 5. Peta dari Contoh Permasalahan

Diambil dari https://www.google.com/maps/@-7.7132503,110.4936669,10.44z/data=!5m1!1e4

A. Memetakan Kondisi Medan

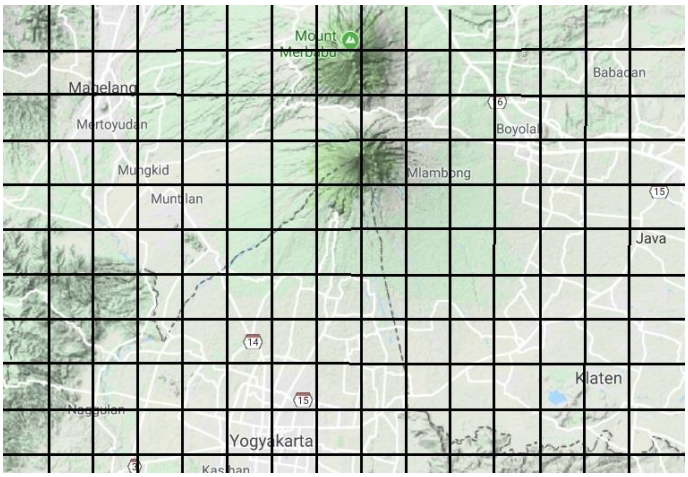
Untuk memetakan kondisi medan dari permasalahan pembangunan jalur tol, terlebih dahulu dilakukan pemetaan daerah menjadi *grid* 2D.

Grid 2D digunakan karena agar mempermudah pemetaan dengan membagi tiap daerah menjadi kotak-kotak sehingga lebih mudah mengambarkan simpul-simpul dari graf permasalahan, sehingga penerapan algoritma A* dapat dilakukan dengan lebih mudah dan akurat.

Grid 2D juga memudahkan manusia dalam melihat dan memahami jalannya algoritma sehingga lebih mudah untuk ditelusuri dan dilihat jika terjadi sebuah kesalahan.

Pemetaan ke dalam *grid* 2D dilakukan dengan cara memberikan garis melintan dan membujur dengan jarak yang sama, sehingga setiap bagian pada peta dapat terbagi menjadi kotak dengan ukuran yang sama.

Pemetaan ke dalam *grid* 2D dapat dilakukan dengan kakas editor yang dapat memberikan garis pada sebuah foto, sehingga Gambar 5 diatas dapat dipetakan ke dalam *grid* 2D sehingga



bentuknya menjadi seperti berikut.

Gambar 6. Representasi Peta dalam *Grid* 2D

Gambar 6 diatas adalah peta yang telah diberikan *grid 2D*, kita dapat melihat peta sebagai matriks, dimana penomoran dimulai dari (0, 0) pada posisi pojok kiri atas. Sehingga kita dapat menetapkan posisi Magelang berada pada (1, 2), sementara posisi Klaten berada pada (8, 13).

Kita dapat dengan mudah memetakan dan melihat peta tersebut jika telah dipetakan menjadi *grid 2D*.

Pada pemetaan *grid 2D* juga dapat merepresentasikan medan yang tidak dapat dibangun jalan tol. Pada permasalahan ini diasumsikan jalan tol tidak dapat dibangun pada Gunung, danau, dan Bukit, karena melewati gunung dan bukit menyebabkan pembangunan jalur tol lebih sulit dan lebih berbahaya untuk kedepannya, oleh karena itu gunung, danau, dan bukit akan dianggap sebagai rintangan.

Untuk menentukan apakah kotak(jalur) dapat dilalui dengan cara memeriksa isi dari kotak tersebut. Berdasarkan asumsi sebelumnya maka Peta dalam *Grid 2D* dapat disederhanakan lagi menjadi Peta dalam *Grid 2D* yang lebih sederhana.

X	X		X			X	X	X			X	X		
		M				X	X	X	X					
X				X	X	X	X	X						
					X	X	X	X	X					
				X	X	X	X	X						
X	X	X			X		X							
X	X	X												
X	X													
X	X												K	
X								X	X	X	X	X	X	X
X								X	X	X	X	X	X	X

Gambar 7. Representasi Peta dalam *Grid 2D* yang lebih sederhana

Pada Gambar 7 diatas dapat kita pahami bahwa X menandakan bahwa jalur tol tidak bisa dibangun disana, M menandakan Magelang, K menandakan Klaten, dan Kotak yang kosong menandakan bahwa medan dapat dibangun jalur tol.

Pada Gambar 7 diatas memang kita dapat memahami peta tersebut dengan mudah, akan tetapi komputer akan susah membaca peta tersebut jika terformat seperti itu, oleh karena itu peta tersebut lebih baik direpresentasikan sebagai Matriks dengan ketentuan nilai 1 sebagai rintangan, nilai 0 sebagai jalur yang dapat dilalui, 5 sebagai titik awal dan , 6 sebagai titik akhir.

1	1	0	1	0	0	1	1	1	0	0	1	1	0	0
0	0	5	0	0	0	1	1	1	1	0	0	0	0	0
1	0	0	0	1	1	1	1	1	0	0	0	0	0	0
0	0	0	0	0	1	1	1	1	1	0	0	0	0	0
0	0	0	0	1	1	1	1	1	0	0	0	0	0	0
1	1	1	0	0	1	0	1	0	0	0	0	0	0	0
1	1	1	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	0	0
1	1	0	0	0	0	0	0	0	0	0	0	0	6	0
1	0	0	0	0	0	0	0	1	1	1	1	1	1	1
1	0	0	0	0	0	0	0	1	1	1	1	1	1	1

Gambar 8. Representasi Peta dalam Matriks

Peta diatas dapat dengan mudah dijalankan oleh komputer sehingga kita tidak perlu repot membuat programnya.

B. Menentukan Fungsi Heuristik

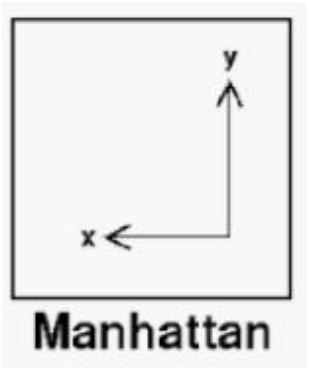
Fungsi heuristik pada A* sangatlah penting karena bekerja atau tidaknya A* ditentukan oleh fungsi heuristik itu sendiri. Untuk itu fungsi heuristik yang tepat harus ditentukan.

Ada dua fungsi heuristik yang dapat digunakan, yaitu *Manhattan Distance* dan *Euclidean Distance.*

1) Manhattan Distance

Manhattan Distance memperkirakan *cost* yang dibutuhkan dengan cara menghitung jumlah simpul atau kotak yang harus dilalui secara horizontal maupun vertikal untuk mencapai tujuan akhir.

$$h = |X_{awal} - X_{akhir}| + |Y_{awal} - Y_{akhir}| \qquad (2)$$



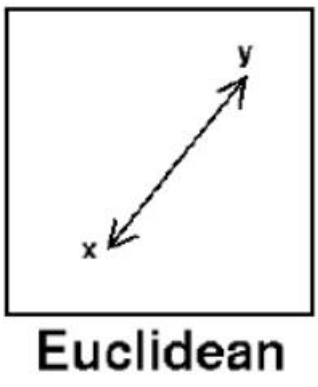
Gambar 9. Contoh Manhattan Distance
Diambil dari
http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/Clustering_Parameters/Manhattan_Distance_Metric.htm

Heuristik ini akan menghasilkan hasil yang tepat jika jalur penelusuran hanya bisa menelusuri gerakan vertikal dan horizontal saja.

2) Euclidean Distance

Euclidean Distance memperkirakan *cost* yang dibutuhkan dengan cara menghitung jumlah simpul atau kotak yang harus dilalui secara horizontal maupun vertikal untuk mencapai tujuan akhir.

$$h = sqrt[(X_{awal} - X_{akhir})^2 + (Y_{awal} - Y_{akhir})^2] \quad (3)$$



Gambar 10. Contoh Manhattan Distance
Diambil dari
http://www.improvedoutcomes.com/docs/WebSiteDocs/Clustering/Clustering_Parameters/Manhattan_Distance_Metric.htm

Heuristik ini akan menghasilkan hasil yang tepat jika jalur penelusuran tidak hanya bergerak secara vertikal, horizontal saja, akan tetapi bergerak secara diagonal saja.

Setelah diketahui dua fungsi heuristik tersebut, akan dipilih salah satu dari kedua fungsi heuristik tersebut untuk digunakan dalam persoalan ini. Untuk permasalahan perancangan jalur tol yang diasumsikan tidak bisa bergerak secara diagonal.

Pada makalah ini, akan dipilih Manhattan Distance sebagai fungsi heuristik karena diasumsikan pembanguna jalan tol tidak bergerak secara diagnoal, oleh karena itu Manhattan Distance lebih cocok digunakan di pada permasalahan kali ini.

Pseudocode untuk menentukan nilai Manhattan Distance adalah sebagai berikut

Pseudocode Manhattan Distance

function ManhattanDistance(start, finish){
  dx = start.x – finish.x
  dy = start.y – finish.y
  return dx+dx
}

C. Algoritma Perancangan Jalur Tol

Kita telah merepresentasikan peta ke dalam bentuk yang mudah diolah oleh komputer dan menentukan fungsi heuristik, sekarang akan dirancang algoritma A* untuk menentukan jalur tol.

Algoritma A* pada perancangan jalur tol ini akan menelusuri tiap kotak dari titik awal hingga ke titik akhir dan akan mengembalikan jalur terpendek. Berikut langkah untuk melakukan pencarian adalah sebagai berikut.

1. Program akan mengambil Titik Awal dan Titik Akhir dari penelusuran.
2. Tiap titik yang diambil dari Titik Awal ke Titik Akhir akan dimasukkan ke dalam Priority Queue.
3. Program akan menelusuri jalan yang akan diambil dengan menggunakan fungsi heuristik *f(n).*
4. Program akan mengembalikan jalur terpendek.

Source Code Algoritma A* dalam Bahasa Pyhton

```
def Astar(start, finish,maze):
    #Penentuan Titik Awal
    x = start[0]
    y = start[1]

    #Penentuan Titik Akhir
    t = finish[0]
    r = finish[1]

    #Pembuatan Priority Queue
    h = 0
    que = PriorityQueue()
    que.put([h,x,y,None,0])

    #Algoritma A* berhenti mencari ketika
sudah mencapai tujuan
    while not que.empty():
        node = que.get()
        p = node[1]
        q = node[2]
        g = node[4]

        #Tujuan Akhir Program
        if ((p,q) == finish):
            print("Dapat Terbentuk Jalur
Dari ",start, " Hingga Ke ", finish )
            return getTrack2(node)

        #Penetapan Nilai g(n)
        g_temp = g+1

        #Penetapan Nilai f(n)
        f1 = (abs(p+1-t) + abs(q-r)) +
g_temp
        f2 = (abs(p-1-t) + abs(q-r)) +
g_temp
        f3 = (abs(p-t) + abs(q+1-r)) +
g_temp
        f4 = (abs(p-t) + abs(q-1-r)) +
g_temp

        #Langkah yang diambil
        if (cek(p+1,q,maze)):
            maze[p][q] = 3
            que.put([f1, p+1, q,
node,g_temp])
        if (cek(p-1,q,maze)):
            maze[p][q] = 3
            que.put([f2, p-1, q,
node,g_temp])
        if (cek(p,q+1,maze)):
            maze[p][q] = 3
            que.put([f3, p, q+1,
node,g_temp])
        if (cek(p,q-1,maze)):
            maze[p][q] = 3
            que.put([f4, p, q-1,
node,g_temp])

    return []
```

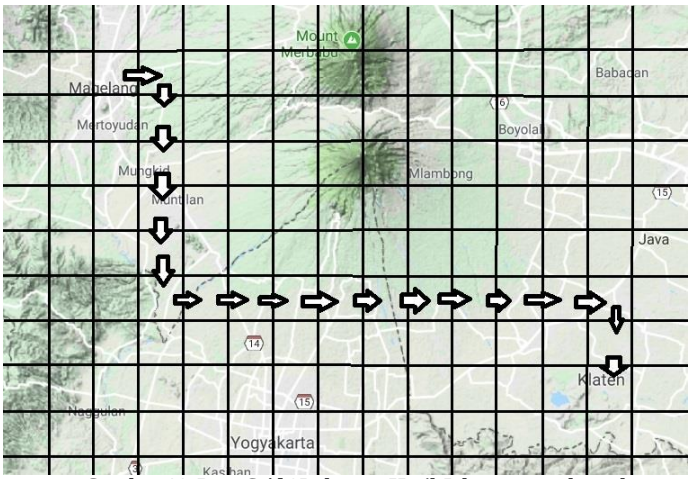
D. Pengujian Algoritma

Hasil dari pengujian diatas dengan contoh persoalan yang telah disebutkan sebelumnya, yaitu perancangan jalur tol antara Kota Magelang dengan Kabupaten Klaten akan menghasilkan hasil sebagai berikut.

Hasil Implementasi pada Persoalan Jalur Tol Magelang - Klaten
Matriks Peta : [1,1,0,1,0,0,1,1,1,0,0,1,1,0,0] [0,0,0,0,0,0,1,1,1,1,0,0,0,0,0] [1,0,0,0,1,1,1,1,1,0,0,0,0,0,0] [0,0,0,0,0,1,1,1,1,1,0,0,0,0,0] [0,0,0,0,1,1,1,1,1,0,0,0,0,0,0] [1,1,1,0,0,1,0,1,0,0,0,0,0,0,0] [1,1,1,0,0,0,0,0,0,0,0,0,0,0,0] [1,1,0,0,0,0,0,0,0,0,0,0,0,0,0] [1,1,0,0,0,0,0,0,0,0,0,0,0,0,0] [1,0,0,0,0,0,0,0,1,1,1,1,1,1,1] [1,0,0,0,0,0,0,0,1,1,1,1,1,1,1]
Titik Awal : (1, 2) Titik Akhir : (8, 13)
Jalur yang terbentuk : (1, 2) => (1, 3) => (2, 3) => (3, 3) => (4, 3) => (5, 3) => (5, 4) => (6, 4) => (6, 5) => (6, 6) => (6, 7) => (6, 8) => (6, 9) => (6, 10) => (6, 11)

=> (6, 12) => (6, 13) => (7, 13) => (8, 13)

Hasil tersebut dapat digambarkan agar lebih mudah dilihat dengan cara menggambarkan anak panah di peta *grid 2D*. Sebagai berikut.



Gambar 11. Peta *Grid 2D* dengan Hasil Jalur yang terbentuk

Dapat terlihat dari Gambar 11 diatas dapat dilihat bahwa Algoritma A* tetap dapat mencari jalur terpendek meskipun ada rintangan sehingga dapat tercipta jalur dari titik awal ke titik akhir dengan panjang 19 petak.

IV. KESIMPULAN

Kesimpulan yang dapat diambil dalam makalah ini adalah algoritma A* terbukti dapat menyelesaikan permasalahan perancangan pembangunan jalur tol dengan baik. Algoritma A* juga dapat berjalan dengan baik ketika fungsi heuristik yang diterapkan sesuai dengan tujuan yang akan dicapai. Jadi untuk menyelesaikan perancangan pembangunan jalur kereta, harus ditentukan pemetaan masalah dan fungsi heuristik yang akan digunakan sebelum memulai membuat algoritma.

REFERENSI

[1] Route/Path Planning Using A Star dan UCS
http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/A-Star-Best-FS-dan-UCS-(2018).pdf
Diakses pada tanggal 12 April 2019

[2] Heuristics
http://theory.stanford.edu/~amitp/GameProgramming/Heuristics.html
Diakses pada 12 April 2019

[3] Sejarah Jalan Tol
http://bpjt.pu.go.id/konten/jalan-tol/sejarah
Diakses pada tanggal 12 April 2019

[4] 5 Jalan Tol dengan Volume Lalu Lintas Tertinggi Tahun 2017
http://bpjt.pu.go.id/berita/5-jalan-tol-dengan-volume-lalu-lintas-tertinggi-tahun-2017
Diakses pada tanggal 13 April 2019

[5] Ruas Tol Baru yang bisa Dilewati di 2016 ini
https://www.truckmagz.com/ruas-tol-baru-yang-bisa-dilewati-di-2016-ini/
Diakses pada tanggal 13 April 2019

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 13 April 2019

Farhan Ramadhan Syah Khair
13517001