Penerapan Algoritma A* Dalam Pathfinding AI Stealth Game

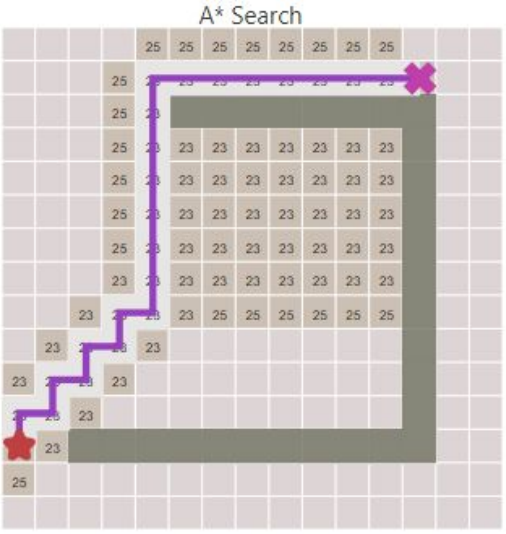
Nicholas Thie 13515079
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13515079@std.stei.itb.ac.id

***Abstrak*—Algoritma A* (*A Star*) adalah salah satu algoritma untuk menyelesaikan persoalan rute terpendek yang menggunakan heuristik untuk mendukung pencariannya. Dengan menggunakan heuristik, performansi hasil rute terpendek algoritma A* menjadi lebih baik. *Stealth game* adalah permainan dimana pemain harus berusaha menghindari rintangan dan/atau antagonis dalam permainan tersebut tanpa terdeteksi. Di dalam sebuah *stealth game* bisa saja sebuah AI (*Artificial Intelligence* / Intelegensi Buatan) akan mengejar pemain apabila pemain terdeteksi, dan penentuan jalan dari AI menuju pemain tersebut dapat diterapkan menggunakan algoritma A*.**

Kata Kunci*—*A Star*, *Pathfinding*, Rute Terpendek, *Stealth Game

I. Pendahuluan

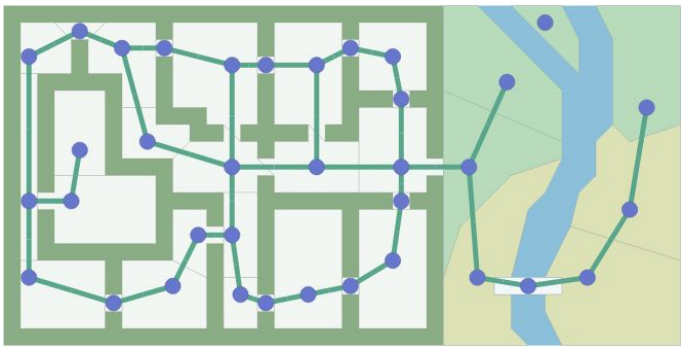
Pathfinding (pencarian jalan) adalah penelusuran jalan oleh komputer untuk mencari jalan dari titik awal sampai ke titik akhir. Pencarian rute terpendek merupakan salah satu pengembangan dari *pathfinding*, dimana jalan yang dipilih merupakan jalan yang paling optimal dari segi biaya (*cost*). Beberapa macam algoritma yang dapat digunakan untuk menyelesaikan persoalan rute terpendek adalah *Breadth First Search* (BFS), *Depth First Search* (DFS), *Uniform Cost Search* (UCS), *Greedy Best-First Search*, dan A* (*A star)*.



Gambar 1 dan 2. Perbandingan pemilihan jalur menggunakan algoritma Greedy Best-First Search dengan A* Search

Algoritma A* (*A star*) adalah salah satu algoritma untuk menyelesaikan persoalan rute terpendek. Algoritma-algoritma yang lain seperti *Breadth First Search* (BFS) atau *Depth First Search* (DFS) juga dapat digunakan, namun performansi algoritma A* dibandingkan dengan algoritma-algoritma tersebut lebih baik karena menggunakan heuristik. Algoritma A* banyak digunakan untuk permainan-permainan bergenre lainnya, seperti permainan *Real-Time Strategy* (RTS).

Untuk mengimplementasikan algoritma A* dalam sebuah permainan, peta dari permainan tersebut perlu direpresentasikan. Representasi dari peta permainan merupakan sebuah graf. Algoritma A* hanya akan mengenal simpul-simpul dan sisi-sisi yang telah direpresentasikan sebagai graf dalam sebuah permainan.



Gambar 3. Contoh ilustrasi representasi graf dari sebuah peta permainan

Stealth game adalah permainan dimana pemain harus berusaha menghindari rintangan dan/atau antagonis dalam permainan tersebut tanpa terdeteksi. Pemain bermulai dari sebuah titik awal dan harus menyelesaikan seluruh objektif dari *level* tersebut sebelum menuju ke titik akhir, dimana pemain dinyatakan menang. Dalam perjalanan menyelesaikan objektif maupun menuju titik akhir, akan terdapat rintangan-rintangan maupun antagonis yang biasanya merupakan AI (*Artificial Intelligence* / Intelegensi Buatan). Apabila pemain terdeteksi oleh rintangan ataupun AI yang ada, salah satu hal yang dapat terjadi adalah AI akan bergerak menuju arah pemain. Untuk menentukan kemana AI harus bergerak selanjutnya agar mendekati pemain, salah satu algoritma yang dapat diterapkan adalah A* (*A Star*).

Stealth game yang akan digunakan sebagai contoh dalam makalah ini adalah *Colossal Carrot Caper*, sebuah permainan yang menggabungkan genre *rhythm* dan *stealth* yang dikembangkan oleh penulis dan rekan-rekannya dalam rangka mengikuti *Ludum Dare 41* (https://ldjam.com/events/ludum-dare/41/the-colossal-carrot-caper). Dalam permainan ini, pemain mengendalikan sebuah karakter kelinci di ladang yang memiliki objektif untuk mengambil seluruh wortel yang ada pada ladan. Di ladang tersebut terdapat serigala-serigala yang berpatroli, yang akan mengejar pemain apabila berada dalam wilayah penglihatannya. Pemain kalah apabila tertangkap oleh serigala atau waktu habis. Pemain menang apabila berhasil mengambil seluruh wortel yang ada di ladang.

Gambar 4. Tampilan permainan *Colossal Carrot Caper*

Dalam makalah ini hanya akan difokuskan pada aspek *stealth* dan *pathfinding* AI di dalam permainan tersebut, yang merupakan serigala sebagai rintangannya. Apabila serigala melihat pemain yang merupakan kelinci, maka serigala akan bergerak mendekati pemain melalui jalur terpendek. Penentuan jalur terpendek dalam permainan *Colossal Carrot Caper* menggunakan algoritma A* dengan fungsi heuristik merupakan *manhattan distance* menuju ke posisi pemain (simpul tujuan).

II. Landasan Teori

A. Algoritma A (A star)*

Algoritma A* adalah algoritma yang merupakan perkembangan dari algoritma *Breadth First Search* (BFS). Kedua algoritma tersebut memiliki prinsip cara kerja yang sama untuk memilih simpul selanjutnya untuk ditelusuri yaitu menggunakan prinsip antrian (*queue*), namun pada algortima A* digunakan antrian berprioritas (*priority queue*), dimana antrian diurutkan berdasarkan biaya (*cost*) dari suatu simpul. Selain itu, algoritma A* juga menggunakan fungsi heuristik yang digunakan dalam menentukan biaya saat pembangkitan suatu simpul pada ruang solusi. Fungsi heuristik ini adalah sebuah perkiraan untuk suatu simpul, apakah simpul tersebut semakin baik untuk mencapai tujuan.

Fungsi evaluasi yang dimiliki oleh algoritma A* adalah :

$$f(n) = g(n) + h(n)$$

dimana,

n = Simpul terakhir di jalur saat ini

f(n) = Estimasi total biaya jalur melalui simpul n menuju tujuan

g(n) = Biaya sejauh ini untuk mencapai simpul n

h(n) = Estimasi biaya dari simpul n menuju tujuan

Pada algoritma A*, untuk setiap simpul yang dibangkitkan dalam ruang solusi akan dihitung biayanya (f(n)) dengan menambahkan nilai dari g(n) dan h(n). g(n) merupakan nilai biaya dari simpul awal ke simpul n. h(n) merupakan nilai

heuristik, yang dimana nilai yang ditentukan/diperkirakan memiliki syarat tidak melebihi nilai biaya sebenarnya untuk mencapai tujuan. Algoritma A* akan mengambil solusi jalur yang mencapai tujuan dengan biaya yang paling kecil (nilai f(n) paling kecil).

Langkah-langkah algoritma A* secara umum adalah sebagai berikut :

1. Memasukkan simpul akar ke dalam antrian Q. Jika simpul akar adalah simpul solusi, maka solusi telah ditemukan dan hentikan pencarian.
2. Jika Q kosong, maka tidak terdapat solusi dan hentikan pencarian.
3. Jika Q tidak kosong, maka pilih simpul i dari antrian Q yang mempunyai nilai f(n) paling kecil. Jika terdapat beberapa simpul n yang memenuhi, maka dipilih satu yang berada dalam antrian terlebih dahulu (atau juga dapat dipilih secara sembarang).
4. Jika simpul n adalah simpul solusi, berarti simpul solusi telah ditemukan, hentikan pencarian. Jika simpul n bukan sebuah simpul solusi, maka ekspansi semua simpul anak dari simpul n. Jika simpul n tidak memiliki simpul anak, maka kembali ke langkah 2.
5. Untuk setiap anak m pada simpul n, dihitung f(m), dan masukkan semua anak-anak tersebut ke dalam antrian Q terurut dari nilai terkecil.
6. Kembali ke langkah 2.

B. Mekanisme Permainan Colossal Carrot Caper

Dalam permainan *Colossal Carrot Caper*, pemain menggerakkan sebuah karakter kelinci untuk mengambil seluruh wortel yang ada di ladang. Pemain harus bergerak sesuai dengan ritme lagu yang berputar. Pemain tidak akan dapat bergerak apabila ingin bergerak diluar ritme lagu.

Gambar 5. Pemain menekan tombol bergerak sesuai dengan ritme lagu

Di ladang tersebut akan terdapat beberapa serigala yang akan melakukan patroli. Dalam jenis gerakan patroli, serigala hanya akan mengunjungi beberapa petak tetap secara berurutan dan berulang-ulang. Apabila dalam patroli tersebut pemain berada dalam sudut pandang serigala, serigala akan mengubah jenis gerakannya dari patroli menjadi mengejar. Mirip seperti pemain, serigala akan bergerak sesuai dengan ritme lagu yang berputar.

Bentuk dari ladang di permainan ini adalah *tile-base*, sehingga seluruh entitas dalam permainan akan bergerak dalam sebuah *grid*, satu petak ke kiri, kanan, atas, atau bawah.

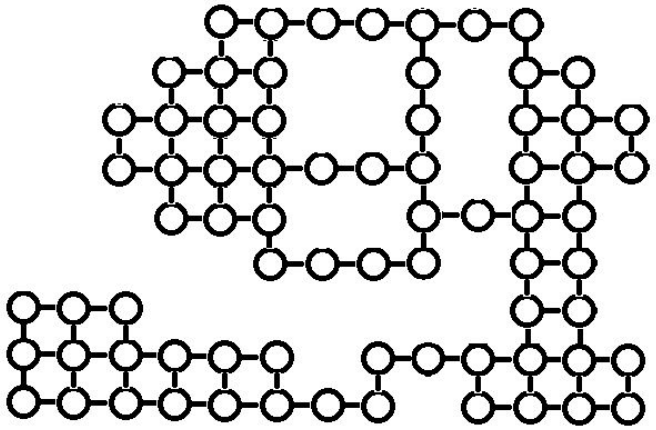
Pemain yang menggerakkan karakter kelinci tersebut harus bergerak melewati seluruh wortel yang terdapat pada ladang tersebut. Setelah melewati seluruh wortel, pemain telah menang dan akan dibawa ke ladang selanjutnya.

Dalam jenis gerakan mengejar, serigala akan selalu berusaha untuk bergerak mendekati ke arah pemain dengan jalur yang paling pendek. Implementasi dari algoritma A* digunakan dalam penentuan jalur ini. Pengejaran akan dilakukan oleh serigala sampai pemain tertangkap atau pemain telah menang.

III. Implementasi Algoritma

Untuk memudahkan pengimplementasian algoritma A*, ladang tempat permainan perlu direpresentasikan. Karena permainan ini adalah *tile-based*, maka simpul-simpul dari graf merepresentasikan petak-petak dari ladang permainan dan sisi-sisi dari graf merepresentasikan pergerakan yang mungkin dari suatu petak. Pada umumnya, setiap simpul memiliki empat buah sisi untuk pergerakan ke atas, bawah, kiri, dan kanan. Namun, untuk beberapa petak tidak memungkinkan untuk bergerak ke suatu arah dikarenakan petak tujuan tidak ada, atau merupakan pohon. Untuk kasus-kasus tersebut, sisi dari simpul tersebut hanyalah antara satu sampai tiga.

Gambar 6. Ladang permainan pertama pada *Colossal Carrot Caper*

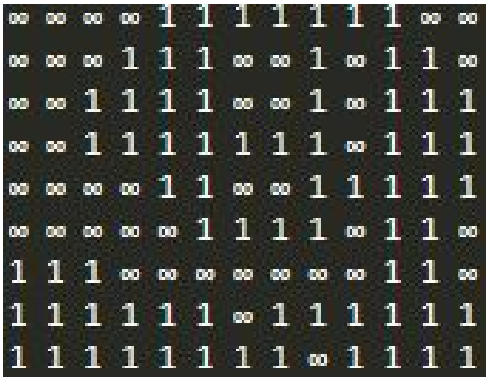


Gambar 7. Representasi graf ladang permainan pertama pada *Colossal Carrot Caper*

Representasi dari graf tersebut akan disimpan dalam sebuah *array* dua dimensi (matriks), dimana setiap elemen dari matriks tersebut adalah informasi mengenai suatu petak ada atau tidak. Informasi tersebut juga meliputi biaya dari masing-masing petak yang ada. Dalam permainan *Colossal Carrot Caper* ini hanya terdapat dua jenis petak saja, yaitu petak yang dapat dilewati, dan petak yang tidak dapat dilewati karena pohon, air, atau halangan lainnya.

Untuk petak yang bisa dilewati, elemen dari matriks akan bernilai satu, yang merupakan biaya dari petak tersebut. Biasanya pada sebuah permainan *Real-Time Strategy* (RTS), terdapat berbagai macam petak seperti daratan, padang gurun, hutan, dan lain-lain. Pada kasus tersebut, biaya setiap macam petak dapat bernilai beda-beda sesuai dengan keinginan pembuat permainan tersebut. Namun, dalam permainan ini jenis petak yang dapat dilewati hanya ada satu saja.

Untuk petak yang tidak bisa dilewati, akan diberi nilai biaya yang sangat besar, dengan maksud agar serigala tidak akan melewati petak tersebut. Hal ini dapat dilakukan dengan asumsi serigala tidak akan pernah terperangkap dalam sebuah petak yang tidak memiliki petak tetangga (tidak bisa bergerak ke atas, bawah, kiri, atau kanan). Karena apabila tanpa asumsi tersebut, serigala dapat bergerak melewati petak yang seharusnya tidak dapat dilewati. Pemberian nilai yang besar ini mempermudah implementasi dari algoritma A* untuk menentukan jalan yang dapat dilewati, karena matriks tidak perlu menyimpan informasi simpul tetangga untuk setiap simpul.



Gambar 8. Representasi matriks dari graf ladang permainan pertama pada *Colossal Carrot Caper*

Fungsi heuristik yang digunakan dalam implementasi algoritma A* ini merupakan *manhattan distance* dari suatu simpul sekarang ke simpul tujuan (dimana tujuannya adalah karakter pemain). Nilai dari *manhattan distance* adalah jarak simpul sekarang ke simpul tujuan pada koordinat x ditambah dengan simpul sekarang ke simpul tujuan pada koordinat y.

Total biaya dari suatu simpul/petak merupakan penjumlahan dari biaya sampai ke simpul/petak tersebut ditambah dengan fungsi heuristik yang merupakan *manhattan distance* menuju simpul/petak tujuan.

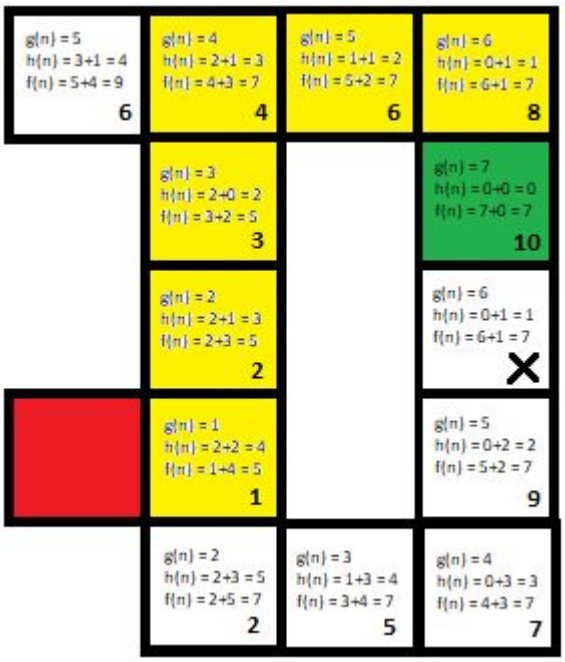
Karena implementasi algoritma A* adalah menggunakan *priority queue*, maka ketika simpul tujuan tercapai akan didapatkan jalur dengan biaya terkecil. Jalur tersebut akan digunakan oleh serigala untuk menentukan apakah untuk pergerakan selanjutnya serigala tersebut harus bergerak ke atas, bawah, kiri, atau kanan.

Di bawah ini merupakan *pseudocode* dari implementasi algoritma A* untuk *pathfinding* serigala :

```
frontier = PriorityQueue()
frontier.put(start, 0)
came_from = {}
cost_so_far = {}
came_from[start] = None
cost_so_far[start] = 0

while not frontier.empty():
   current = frontier.get()

   if current == goal:
      break

   for next in graph.neighbors(current):
      new_cost = cost_so_far[current] + graph.cost(current, next)
      if next not in cost_so_far or new_cost < cost_so_far[next]:
         cost_so_far[next] = new_cost
         priority = new_cost + heuristic(goal, next)
         frontier.put(next, priority)
         came_from[next] = current
```

Dikutip dari http://www.redblobgames.com

Selanjutnya akan dibahas ilustrasi salah satu contoh kasus pengejaran yang dilakukan oleh serigala.



Gambar 9. Contoh kasus pengejaran oleh serigala (kotak merah) ke pemain (kotak hijau)

Terdapat contoh kasus dimana serigala yang berada pada kotak merah sedang mengejar pemain yang berada pada kotak hijau. Setiap petak telah diberikan nilai-nilai dari g(n) dan h(n) sesuai dengan kriteria yang telah dijelaskan sebelumnya. Simpul yang akan dibangkitkan pertama adalah simpul-simpul tetangga dari kotak merah, dalam kasus ini hanya ada 1 yaitu sebelah kanan kotak merah (yang ditandai nomor 1). Nilai dari g(n) dan h(n) dikalkulasi dan dijumlahkan menjadi f(n) yang bernilai 5. Sekarang hanya terdapat satu simpul hidup, dan kemudian akan dibangkitkan simpul-simpul selanjutnya yang merupakan tetangganya (yang ditandai nomor 2). Akan dilakukan kalkulasi yang sama seperti simpul pertama sebelumnya, dan akan didapatkan nilai f(n) 5 dan 7. Karena dalam A* digunakan *priority queue*, maka simpul yang bernilai f(n) 5 akan diproses terlebih dahulu. Tetangga simpul tersebut akan dibangkitkan (yang ditandai nomor 3), dan dilakukan hal yang sama seperti sebelumnya untuk membangkitkan simpul selanjutnya (yang ditandai nomor 4).

Pada saat ini terdapat 2 simpul dengan nilai sama yaitu 7. Simpul yang bernilai f(n) = 7 dan terbuka ketika iterasi ke-2 akan diproses terlebih dahulu karena terletak di antrian yang lebih dulu daripada simpul yang bernilai f(n) = 7 dan terbuka ketika iterasi ke-4. Oleh karena itu simpul yang akan dibangkitkan adalah simpul yang ditandai nomor 5.

Kejadian yang sama terjadi terus-menerus sampai pada iterasi ke-10 tercapai simpul tujuan (bewarna hijau) dengan nilai f(n) = 7. Simpul-simpul yang diberi warna kuning merupakan jalur menuju simpul tujuan. Simpul di bawah simpul tujuan yang diberi tanda silang (X) tidak sempat dibangkitkan, karena pada iterasi ke-10 sudah dicapai simpul tujuan.

Setelah didapatkan jalur menuju simpul tujuan, serigala akan mengambil simpul selanjutnya pada jalur tersebut, yaitu simpul yang berada di sebelah kanannya (yang ditandai nomor 1). Apabila pemain tidak bergerak, maka serigala akan bergerak mengikuti jalur kuning tersebut, yang merupakan jalur terpendek menuju pemain. Apabila pemain berpindah tempat, maka serigala akan menghitung kembali jalur terpendek yang perlu ditempuh menuju pemain.

Di bawah ini merupakan tabel uraian setiap iterasi dari simpul awal sampai dengan simpul akhir/tujuan :

Iterasi	Simpul Diekspan	Simpul Hidup
1	(1,3)	(1,3) = 5
2	(1,2),(1,4)	(1,2) = 5 (1,4) = 7
3	(1,1)	(1,1) = 5 (1,4) = 7
4	(1,0)	(1,4) = 7 (1,0) = 7
5	(2,4)	(1,0) = 7 (2,4) = 7
6	(0,0),(2,0)	(2,4) = 7 (2,0) = 7 (0,0) = 9
7	(3,4)	(2,0) = 7 (3,4) = 7 (0,0) = 9
8	(3,0)	(3,4) = 7 (3,0) = 7 (0,0) = 9
9	(3,3)	(3,0) = 7 (3,3) = 7 (0,0) = 9
10	(3,1)	SELESAI

IV. Kesimpulan

Algoritma A* (*A Star*) dapat digunakan untuk menyelesaikan persoalan rute terpendek yang merupakan pengembangan dari *pathfinding*. Dalam permainan *Colossal*

Carrot Caper, algoritma A* dapat digunakan untuk mencari rute terpendek dari posisi serigala menuju posisi pemain. Penerapan algoritma A* ini memanfaatkan fungsi heuristik yang dalam permainan ini menggunakan *manhattan distance*. Hasil dari penerapan algoritma A* ini adalah solusi optimal dengan biaya tempuh terkecil (jalur terpendek). Serigala akan mengikuti hasil algoritma A* yang merupakan jalur terpendek untuk bergerak mengejar pemain.

Ucapan Terima Kasih

Terima kasih kepada Tuhan Yang Maha Esa, karena atas karunia-Nyalah makalah berjudul "Penerapan Algoritma A* Dalam *Pathfinding* AI *Stealth Game*" ini dapat diselesaikan dengan baik. Penulis mengucapkan terima kasih kepada bapak Dr. Ir. Rinaldi Munir MR, ibu Masayu Leyla Khodra ST., MT., dan ibu Dr.Nur Ulfa Maulidevi atas bimbingan dalam mengajarkan saya mata kuliah IF2211 Strategi Algoritma. Tak lupa penulis juga mengucapkan terima kasih kepada keluarga dan rekan-rekan penulis atas bantuannya dalam pembuatan makalah ini.

Referensi

[1] https://www.redblobgames.com/pathfinding/a-star/introduction.html

[2] Slide A-Star-Best-FS-dan-UCS-(2018) oleh Rinaldi Munir (http://informatika.stei.itb.ac.id/~rinaldi.munir/Stmik/2017-2018/A-Star-Best-FS-dan-UCS-(2018).pdf)

[3] https://ldjam.com/events/ludum-dare/41/the-colossal-carrot-caper

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 Mei 2018

Nicholas Thie 13515079