Implementasi Algoritma A* dalam Navigasi Karakter Nonpemain Melalui Rintangan Dua Dimensi dalam Permainan Video

Jericho Russel Sebastian - 13521107
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
13521107@std.stei.itb.ac.id

***Abstract*—Seiring perkembangan permainan video, teknologi yang digunakan dalam permainan video dikembangkan untuk memungkinkan pengalaman dan *gameplay* yang lebih baik bagi pemain. Salah satu pengembangan yang dilakukan adalah pengembangan sistem tanggapan terhadap masukan pemain yang lebih rasional, cerdas, dan realistis. Kecerdasan buatan dalam permainan video merupakan salah satu bidang kecerdasan buatan dengan perkembangan paling pesat pada akhir abad ke-20, dan hingga saat ini terus mengalami perkembangan dan inovasi baru. Makalah ini akan membahas salah satu submasalah dalam pembangunan kecerdasan buatan dalam permainan video, yaitu pembentukan sistem navigasi yang menerapkan algoritma A* untuk memungkinkan karakter nonpemain melalui rintangan dalam sebuah peta dua dimensi. Sistem navigasi yang dibentuk dapat menangani berbagai perilaku yang mungkin dicerminkan oleh karakter nonpemain ketika berinteraksi dengan pemain, karakter lain, ataupun lingkungan.**

***Keywords*—Graf, Pencarian Lintasan, A*, Kecerdasan Buatan dalam Permainan Video**

I. PENDAHULUAN

Permainan video merupakan salah satu cabang tersukses yang terlingkup di dalam irisan antara bidang seni dan teknologi [1]. Di samping menyediakan ruang bagi industri komersial, permainan video juga menjadi wadah ekspresi seni visual akibat peran krusial grafika komputer di dalamnya. Seiring meningkatnya kemampuan komputer pada akhir abad ke-20, industri perfilman dan televisi juga memberikan dampak terhadap perkembangan permainan video sebagai sarana hiburan. Konteks narasi dan interaksi antarkarakter menjadi faktor yang signifikan dalam pengembangan sebuah permainan video. Namun, terlepas dari pengaruh eksternal, pesatnya perkembangan permainan video merupakan akibat dari kebutuhan untuk menyajikan interaktivitas dan menggambarkan ruang gerak bagi pemain.

Istilah "kecerdasan buatan" dalam konteks permainan video mengacu kepada metode yang digunakan untuk menghasilkan perilaku yang cerdas dan adaptif pada karakter nonpemain terhadap masukan dari pemain. Kecerdasan buatan dalam permainan video merupakan salah satu implementasi terawal kecerdasan buatan. Kecerdasan buatan dalam permainan video direalisasikan dalam bentuk tingkat kesulitan, *in-game event*, pola gerakan, dan lainnya, yang dibangun dengan berbagai teknik seperti pohon keputusan dan pencarian lintasan.

Pencarian lintasan merupakan salah satu masalah yang paling banyak ditemui dalam pembangunan kecerdasan buatan dalam permainan video. Algoritma pencarian lintasan digunakan untuk mencari lintasan optimal yang akan diikuti oleh karakter ketika bergerak melalui peta permainan. Berbagai pendekatan, seperti pencarian melebar pertama (BFS, *breadth-first search*) dan pencarian mendalam pertama (DFS, *depth-first search*), merupakan metode pendekatan lengkap. Artinya, pencarian akan melakukan iterasi terhadap semua lintasan yang mungkin. Namun, pendekatan ini tidak efisien karena pencarian memproses lebih banyak kemungkinan lintasan daripada yang diperlukan. Untuk mengatasi masalah ini, digunakan pendekatan yang dapat mengeliminasi lintasan-lintasan yang tidak berguna dari pencarian. Terdapat dua algoritma populer yang menggunakan pendekatan ini, yaitu algoritma Dijkstra dan A*.

A* tergolong sebagai algoritma pencarian heuristik, yang bermakna bahwa A* melakukan pencarian dengan mengutamakan lintasan-lintasan yang paling mungkin adalah lintasan terbaik. Selain itu, A* juga merupakan algoritma yang fleksibel dan mudah disesuaikan dengan kebutuhan implementasi. Karena kedua alasan ini, A* menjadi algoritma yang paling banyak digunakan dalam pemecahan masalah pencarian lintasan dalam permainan video.

Dalam makalah ini, penulis akan melakukan eksplorasi terhadap implementasi algoritma A* yang digunakan untuk menghasilkan beberapa perilaku dasar yang paling umum diterapkan dalam kecerdasan buatan dalam permainan video.

II. DASAR TEORI

A. Kecerdasan Buatan dalam Permainan Video

Permainan video, secara informal, adalah permainan elektronik yang berinteraksi dengan pemainnya lewat media layar. Definisi permainan video sulit untuk dipersempit karena penggunaan istilah tersebut dalam masyarakat, budaya, dan bahkan dalam industri permainan video sendiri jauh lebih luas dibandingkan dengan definisi teknisnya [1].

Terdapat empat elemen yang umumnya ada dalam suatu

permainan, yaitu konflik (*conflict*), peraturan (*rule*), kemampuan pemain (*player ability*), dan hasil akhir yang dihargai (*valued outcome*). Dalam sebuah permainan video, keempat elemen tersebut juga ada. Namun, berbeda dengan permainan lainnya seperti permainan papan atau kartu, keempat elemen ini diatur dan diawasi oleh komputer ketimbang manusia. Selain itu, komputer juga dapat mengambil peran sebagai karakter-karakter nonpemain di dalam permainan.

Kecerdasan buatan dalam permainan video (*videogame AI*) memiliki perbedaan mendasar dari kecerdasan buatan sejati [2]–[3]. Kecerdasan buatan dalam permainan video mengacu pada komputasi otomatis terhadap sekumpulan data atau masukan dan memberikan tanggapan yang bersesuaian, serta umumnya tidak memanfaatkan teknik pembelajaran komputer seperti kecerdasan buatan sejati. Kecerdasan buatan dalam permainan video melingkupi sistem navigasi dan pencarian lintasan serta pengambilan keputusan. Namun, banyak pengembangan dapat dilakukan untuk membentuk sistem kecerdasan buatan yang termodernisasi dengan memanfaatkan teknologi yang tersedia maupun yang sedang diteliti [3].

Karakter nonpemain (*non-player character*, *NPC*) adalah karakter dalam permainan yang tidak dikendalikan oleh seorang pemain [4]. Dalam permainan video, karakter nonpemain umumnya dikendalikan oleh komputer melalui kecerdasan buatan permainan video [3]. Karakter yang dikendalikan oleh kecerdasan buatan disebut sebagai agen dari kecerdasan buatan tersebut. Perilaku (*behavior*) yang ditampilkan oleh agen ditentukan oleh kecerdasan buatan yang mengendalikannya.

Peta (*map*) merupakan representasi dari bentuk geometri dan isi dari lingkungan yang menjadi latar permainan video. Peta merupakan tempat terjadinya semua interaksi baik antara karakter dan lingkungan maupun antarkarakter. Peta terdiri dari daerah *traversable* (dapat dilalui oleh karakter) dan daerah *non-traversable* (rintangan, tidak dapat dilalui oleh karakter). Peta umumnya digunakan untuk merepresentasikan lingkungan permainan dua dimensi dan tiga dimensi, dan disebut peta dua dimensi dan tiga dimensi secara berturut-turut. Dalam peta dua dimensi, setiap titik pada peta dinyatakan dengan sebuah vektor posisi dua dimensi ($\mathbb{R}^2$).

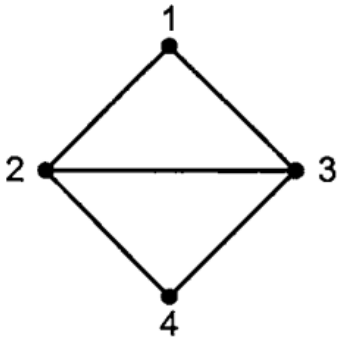
B. Graf

Graf G didefinisikan sebagai sebuah pasangan (V, E) di mana V adalah himpunan tidak-kosong dari simpul-simpul (*vertices* atau *nodes*) dan E adalah himpunan sisi (*edges* and *arcs*) yang menghubungkan sepasang simpul [5]. Simpul pada suatu graf umumnya dinyatakan dengan huruf a, b, c, …, dengan bilangan asli 1, 2, 3, …, atau gabungan keduanya, sedangkan sisi yang menghubungkan simpul x dan simpul y dinyatakan sebagai pasangan (x, y) atau dengan lambang e_1, e_2, e_3, …. Graf dapat digolongkan berdasarkan dua kriteria, yaitu ada tidaknya gelang atau sisi ganda dan orientasi arah pada sisi.

Berdasarkan ada tidaknya gelang atau sisi ganda, graf dapat digolongkan menjadi dua jenis:

1. Graf sederhana (*simple graph*)

Graf sederhana merupakan graf yang tidak mengandung gelang maupun sisi ganda. Pada graf sederhana, sisi yang menghubungkan simpul x dan simpul y merupakan pasangan tak-terurut $\{x, y\}$. Dengan kata lain, sebuah graf G merupakan pasangan (V, E) di mana V adalah himpunan tidak-kosong simpul-simpul pada G dan E adalah himpunan pasangan tak-terurut yang berbeda yang disebut sisi.

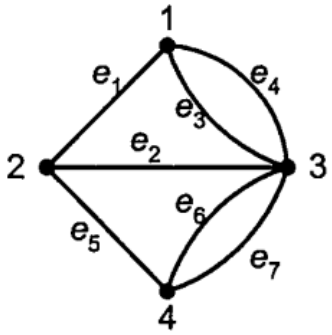


Gambar 2.1 Contoh graf sederhana

2. Graf tidak sederhana (*unsimple graph*)

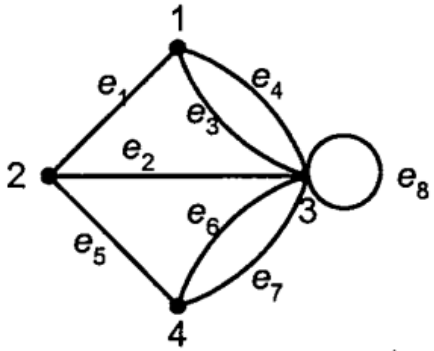
Graf tidak sederhana merupakan graf yang mengandung gelang, sisi ganda, atau keduanya. Berdasarkan jenis sisi yang terkandung, graf tidak sederhana dibagi menjadi dua jenis, yaitu graf sisi ganda dan graf semu.

Graf-ganda (*multigraph*) merupakan graf yang mengandung sisi ganda dan tidak mengandung gelang. Dua simpul dalam graf sisi ganda dapat dihubungkan oleh lebih dari satu sisi. Dengan demikian, graf-ganda G dapat didefinisikan sebagai pasangan (V, E) di mana V adalah himpunan tidak-kosong simpul-simpul pada G dan E adalah himpunan-ganda (*multiset*) pasangan tak-terurut yang disebut sisi.



Gambar 2.2 Contoh graf-ganda

Graf semu (*pseudograph*) merupakan graf yang mengandung gelang (*loop*). Sebuah gelang menghubungkan simpul ke dirinya sendiri; dengan kata lain, suatu gelang e pada simpul x merupakan pasangan tak-terurut $\{x, x\}$. Sebuah graf semu dapat mengandung sisi ganda.



Gambar 2.3 Contoh graf semu

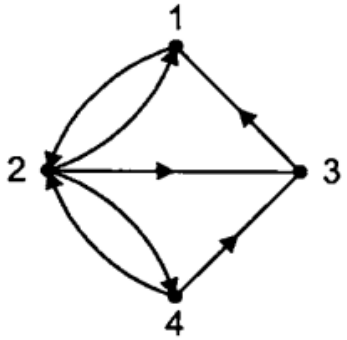
Berdasarkan orientasi arah pada sisi, graf dapat digolongkan menjadi dua jenis:

1. Graf tak-berarah (*undirected graph*)

Graf tak-berarah merupakan graf yang sisi-sisinya tidak memiliki orientasi; sisi pada graf tak-berarah dinyatakan sebagai pasangan tak-terurut.

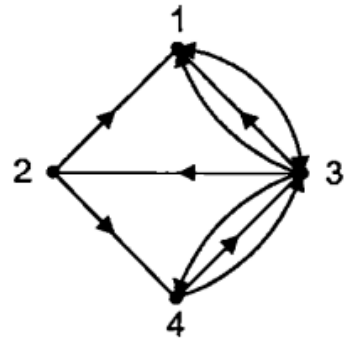
2. Graf berarah (*directed graph* atau *digraph*)

Graf berarah merupakan graf yang sisi-sisinya memiliki orientasi. Sisi pada graf berarah disebut sebagai busur (*arc*). Busur pada graf berarah merupakan pasangan terurut (*x*, *y*) di mana *x* adalah simpul asal (*initial vertex*) dan *y* adalah simpul terminal (*terminal vertex*). Pada graf berarah, gelang diperbolehkan, tetapi sisi ganda tidak.



Gambar 2.4 Contoh graf berarah

Definisi graf dapat diperluas sehingga mencakup graf-ganda berarah (*directed multigraph*). Pada graf-ganda berarah, gelang dan sisi ganda diperbolehkan ada.



Gambar 2.5 Contoh graf-ganda berarah

Berikut adalah terminologi dasar dalam graf:

1. Kebertetanggaan (*adjacency*)

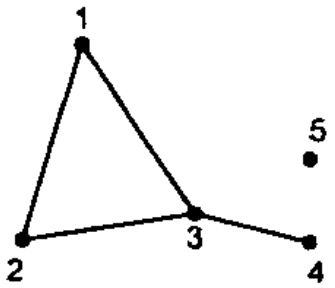
Simpul x bertetangga (*adjacent*) dengan simpul y dalam graf G jika terdapat sisi $e = (x, y)$ dalam G.

2. Kebersisian (*incidence*)

Untuk sembarang sisi $e = (x, y)$ dalam graf G, sisi e bersisian (*incident*) dengan simpul x dan simpul y.

3. Simpul terpencil (*isolated vertex*)

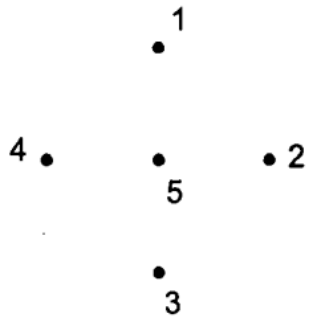
Simpul x dalam graf G terpencil jika tidak ada sisi dalam G yang bersisian dengan x.



Gambar 2.6 Contoh graf dengan simpul terpencil, yaitu simpul 5

4. Graf kosong (*null graph* atau *empty graph*)

Graf kosong merupakan graf yang tidak mengandung sisi; dengan kata lain, graf $G = (V, E)$ kosong jika E adalah himpunan kosong.



Gambar 2.7 Contoh graf kosong

5. Derajat (*degree*)

Simpul x dalam graf G berderajat n jika terdapat n sisi dalam G yang bersisian dengan x.

6. Lintasan (*path*)

Lintasan dari simpul v_0 hingga simpul v_n dalam graf G merupakan himpunan selang-seling antara simpul dan sisi $\{v_0, e_1, v_1, e_2, v_2, \ldots, v_{n-1}, e_n, v_n\}$ di mana $e_i = (v_{i-1}, v_i)$. Sebuah lintasan yang berakhir di v_n dapat diperluas dengan simpul x yang bertetangga dengan v_n dan menghasilkan lintasan baru $\{v_0, \ldots, v_n, e_x, x\}$ di mana $e_x = (v_n, x)$.

7. Siklus (*cycle*) atau sirkuit (*circuit*)

Siklus merupakan lintasan yang berawal dan berakhir di simpul yang sama.

8. Keterhubungan (*connectedness*)

Dua simpul x dan y dalam graf G terhubung (*connected*) jika terdapat lintasan dari x ke y dalam G. Selanjutnya, graf G disebut graf terhubung (*connected graph*) jika untuk semua pasangan simpul a dan b dalam G, a dan b terhubung.

9. Upagraf (*subgraph*)

Graf $G_1 = (V_1, E_1)$ adalah upagraf dari graf $G = (V, E)$ jika $V_1 \subseteq V$ dan $E_1 \subseteq E$. Graf $G_2 = (V_2, E_2)$ adalah komplemen dari upagraf G_1 jika $E_2 = E - E_1$ dan V_2 adalah himpunan semua simpul x dalam G sedemikian hingga setidaknya satu sisi pada E_2 bersisian dengan x.

10. Upagraf merentang (*spanning subgraph*)

Upagraf G_1 dari graf G merupakan upagraf merentang jika G_1 mengandung semua simpul G.

11. *Cut-set*

Cut-set S dari graf S merupakan himpunan sisi G yang apabila dibuang dari G menyebabkan G menjadi tidak terhubung. S merupakan *fundamental cut-set* jika S tidak mengandung himpunan bagian yang merupakan *cut-set* dari G.

12. Graf berbobot (*weighted graph*)

Graf berbobot merupakan graf yang setiap sisinya memiliki sebuah nilai yang disebut bobot (*weight*). Bobot suatu sisi dapat merepresentasikan kuat hubungan antara kedua simpul yang bersisian, jarak antara kedua simpul, dan sebagainya.

13. Graf kisi (*lattice graph* atau *grid graph*)

Graf kisi merupakan graf yang, ketika digambar dalam ruang $\mathbb{R}^n$, menghasilkan pola ubin teratur.

*C. Algoritma A**

Algoritma A* adalah algoritma pencarian lintasan pada graf. A* banyak digunakan dalam pemecahan masalah pencarian lintasan di mana kecepatan pencarian diutamakan, seperti permainan video (*video game*). Algoritma A* pertama kali dipublikasikan oleh Peter E. Hart, Nils J. Nilsson, dan Bertram Raphael dari Stanford Research Institute pada tahun 1968 [6].

A* dapat dianggap sebagai variasi dari algoritma Dijkstra yang menggunakan heuristik untuk memandu pencarian. Karena itu, A* tergolong sebagai algoritma pencarian dengan informasi (*informed search*). Berbeda dengan algoritma Dijkstra yang menghasilkan pohon lintasan terpendek (*shortest-path tree*) dari suatu simpul ke semua simpul lain dalam graf, A* hanya menghasilkan satu lintasan, yaitu lintasan dari suatu simpul asal (*source*) ke suatu simpul tujuan (*goal*).

Referensi [5] menjabarkan algoritma A* sebagai berikut: dimulai dari suatu simpul awal s dan simpul tujuan t dalam graf G, A* membentuk sebuah pohon lintasan yang berawal dari s. Kemudian, dalam setiap iterasi perulangan, A* memperluas salah satu lintasan dengan satu simpul, yaitu simpul n yang meminimasi fungsi evaluasi f yang terdefinisi sebagai

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

dengan $g(n)$ bobot lintasan optimal dari s ke n dan $h(n)$ bobot lintasan optimal dari n ke t. Pencarian berhenti ketika lintasan yang akan diperluas sudah berakhir di t atau tidak ada lintasan yang dapat diperluas. Untuk mempermudah perhitungan, fungsi evaluasi yang digunakan dalam implementasi aktual adalah fungsi taksiran $\hat{f}$ yang terdefinisi sebagai

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

dengan $\hat{g}(n)$ bobot lintasan minimum dari s ke n yang ditemukan oleh A* sejauh ini dan $\hat{h}(n)$ fungsi heuristik yang menaksir bobot lintasan optimal dari n ke t.

Jika simpul pada graf berbobot G yang akan dilintasi mewakili sebuah titik pada ruang dua dimensi, maka bobot sembarang sisi $e = (u, v)$ adalah jarak Euclidean antara kedua titik yang diwakili simpul u dan simpul v, dilambangkan sebagai $d(u, v)$. Selanjutnya, jika G merupakan graf kisi (*grid graph*), maka bobot sembarang sisi $e = (u, v)$ adalah taksiran jarak Euclidean antara kedua titik yang diwakili simpul u dan simpul v, dilambangkan sebagai $\hat{d}(u, v)$. Penaksiran ini dapat berupa jarak Manhattan untuk kisi 4-arah atau jarak oktil untuk kisi 8-arah.

Definisi algoritma A* untuk menentukan lintasan terpendek dari simpul s ke simpul t dalam graf G adalah sebagai berikut:

1. Tandai s sebagai "terbuka", dan hitung $\hat{f}(s)$.
2. Pilih simpul n dari semua simpul "terbuka" dengan nilai $\hat{f}$ terkecil. Jika ada lebih dari satu simpul dengan nilai $\hat{f}$ terkecil, pilih simpul dengan nilai $\hat{h}$ terkecil. Sedangkan, jika tidak ada simpul "terbuka", akhiri pencarian.
3. Jika kondisi berhenti dipenuhi, tandai t "tertutup" dan akhiri pencarian.
4. Jika tidak, tandai n "tertutup". Untuk setiap simpul p yang bertetangga dengan n, hitung $\hat{f}(p)$. Jika p belum memiliki predesesor atau nilai $\hat{g}(p)$ baru lebih kecil dari nilai $\hat{g}(p)$ yang sudah dihitung sejauh ini, ubah predesesor dari p menjadi n dan tandai p sebagai "terbuka". Kembali ke langkah 2.

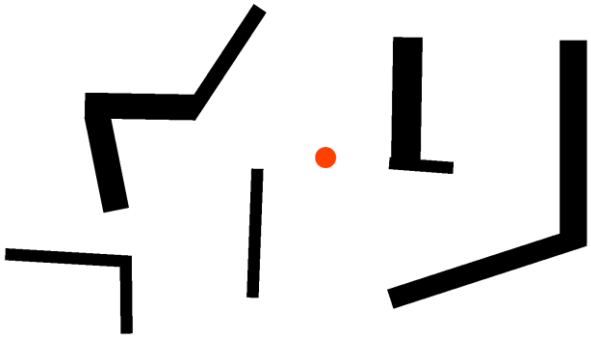
Jika algoritma berakhir pada langkah 3, maka pencarian dinyatakan berhasil. Namun, jika algoritma berakhir pada langkah 2 karena tidak ada simpul "terbuka" yang tersisa, maka pencarian dinyatakan gagal.

Perlu diperhatikan bahwa A* merupakan kumpulan algoritma, bukan satu algoritma tertentu. Setiap algoritma dalam kumpulan A* dibedakan berdasarkan pemilihan fungsi heuristik $\hat{h}$ dan kondisi berhenti. Perilaku tertentu yang diinginkan dapat dihasilkan dengan pemilihan $\hat{h}$ dan kondisi berhenti yang tepat.

III. Implementasi Algoritma A* dalam Navigasi Karakter Nonpemain Melalui Rintangan Dua Dimensi

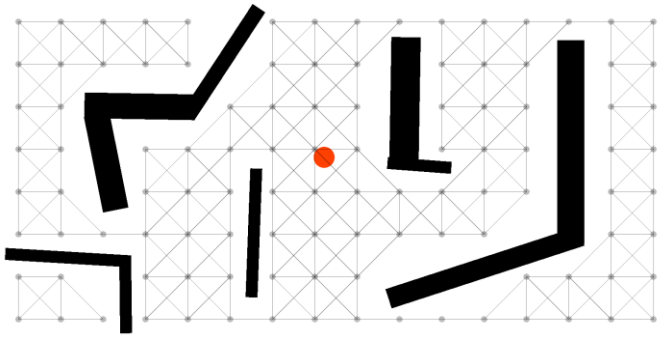
A. Representasi Graf Kisi dari Peta

Misalkan sebuah peta P yang ditempati oleh sebuah agen yang terletak di titik S memiliki rintangan sebagai berikut:



Gambar 3.1 Peta P dengan rintangan ditandai dengan warna hitam dan letak agen S ditandai dengan noktah jingga

P dapat direpresentasikan dengan sebuah graf kisi 8-arah G dengan setiap simpul mewakili titik-titik *traversable* sebagai berikut:

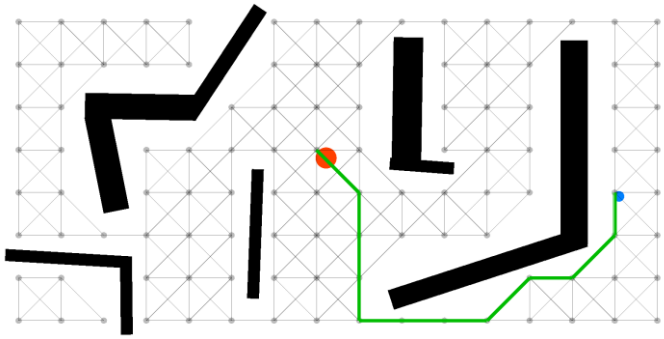


Gambar 3.2 Peta P dengan graf kisi G

Dengan demikian, setiap lintasan yang dapat dilalui oleh agen merupakan salah satu lintasan yang dapat dibuat pada G. Perlu diperhatikan bahwa G tidak terhubung karena terdapat komponen terhubung yang mewakili sudut kiri bawah P yang tidak terhubung dengan komponen lain pada G. Misalkan sebuah titik acuan T terletak pada P, simpul awal s adalah simpul terdekat dengan S, dan simpul acuan t adalah simpul terdekat dengan T.

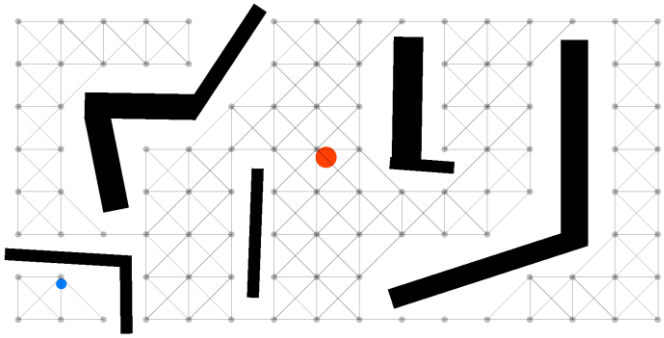
B. Navigasi Mendekati Suatu Titik Acuan

Jika agen memutuskan untuk melakukan navigasi menuju T, maka lintasan yang harus dilalui ditentukan dengan menjalankan algoritma A* pada G. Digunakan definisi fungsi heuristik $\hat{h}(n) = \hat{d}(n,t)$, dengan $\hat{d}(x,y)$ jarak Manhattan antara x dan y, dan kondisi algoritma berhenti jika simpul n yang dipilih adalah t. Lintasan yang ditemukan adalah sebagai berikut:



Gambar 3.3 Peta P dengan titik acuan T ditandai dengan noktah biru dan lintasan terpendek antara s dan t berwarna hijau

Jika s dan t tidak terhubung dalam G, maka algoritma gagal menemukan lintasan dari s ke t:

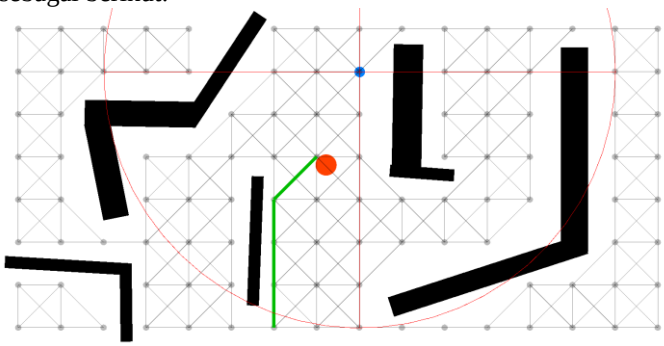


Gambar 3.4 Peta P dengan titik acuan T, ditandai dengan noktah biru, tidak terhubung dengan S

Navigasi menuju suatu titik merupakan jenis navigasi yang paling umum diterapkan dalam sistem pencarian lintasan dalam permainan video. Jenis navigasi ini digunakan untuk memungkinkan agen bergerak menuju suatu tujuan, seperti misalnya mengikuti karakter pemain, berpindah ke suatu lokasi dalam peta, dan sebagainya.

C. Navigasi Menghindari Suatu Titik Acuan

Jika agen memutuskan untuk melakukan navigasi menghindari T setidaknya sejauh r, maka lintasan yang harus dilalui ditentukan dengan menjalankan algoritma A* pada G dengan pemilihan definisi $\hat{h}(n) = r - \hat{d}(n,t)$ dan kondisi berhenti jika $\hat{d}(n,t) \geq r$. Lintasan yang ditemukan adalah sebagai berikut:



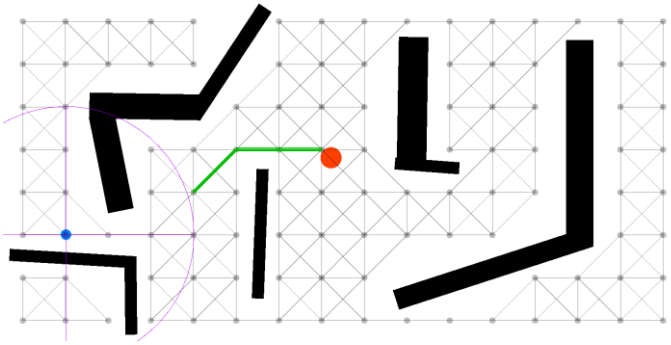
Gambar 3.5 Peta P dengan titik acuan T ditandai dengan noktah biru, lingkaran dengan jari-jari r berwarna merah, dan lintasan terpendek dari s yang menghindari t setidaknya sejauh r berwarna hijau

Definisi $\hat{h}(n) = r - \hat{d}(n,t)$ dipilih untuk memberikan bias terhadap simpul "terbuka" yang berjarak lebih jauh dari t dibandingkan simpul "terbuka" lainnya.

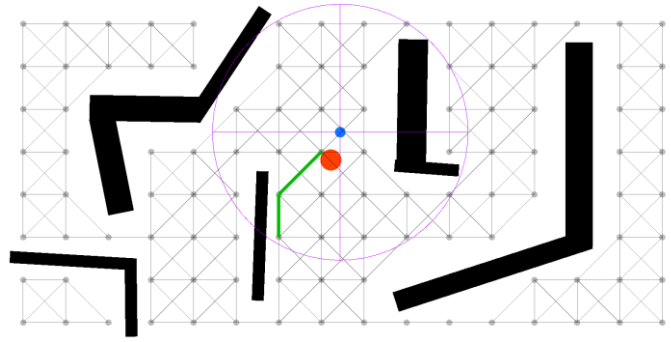
Navigasi menghindari suatu titik, atau *flee*, merupakan jenis navigasi yang sering digunakan dalam berbagai permainan video di mana terdapat objek yang perlu dihindari oleh karakter nonpemain, seperti bahaya, penangkal, atau bahkan karakter lainnya.

D. Navigasi Menjaga Jarak Terhadap Suatu Titik Acuan

Jika agen memutuskan untuk melakukan navigasi menjaga jarak sejauh r terhadap T, maka lintasan yang harus dilalui ditentukan dengan menjalankan algoritma A* pada G dengan pemilihan definisi $\hat{h}(n) = |r - \hat{d}(n,t)|$ dan kondisi berhenti jika $|r - \hat{d}(n,t)| < 1$. Terdapat dua kasus yang ditangani oleh algoritma ini, yaitu kasus ketika agen harus mendekati T (ketika $\hat{d}(s,t) \geq r$) dan ketika agen harus menjauhi T (ketika $\hat{d}(s,t) < r$). Untuk kedua kasus, A* menghasilkan lintasan sedemikian hingga pada simpul tujuan x, $\hat{d}(x,t) \approx r$. Berikut adalah lintasan yang ditemukan untuk kedua kasus:



Gambar 3.6 Peta P dengan titik acuan T ditandai dengan noktah biru, lingkaran dengan jari-jari r berwarna ungu, dan lintasan terpendek dari s agar agen berjarak kira-kira r dari t berwarna hijau, dalam kasus $\hat{d}(s,t) \geq r$.



Gambar 3.7 Peta P dengan titik acuan T ditandai dengan noktah biru, lingkaran dengan jari-jari r berwarna ungu, dan lintasan terpendek dari s agar agen berjarak kira-kira r dari t berwarna hijau, dalam kasus $\hat{d}(s,t) < r$.

Definisi $\hat{d}(n) = \left|r - \hat{d}(n,t)\right|$ dipilih untuk memberikan bias terhadap simpul "terbuka" yang berjarak lebih dekat ke r terhadap t dibandingkan simpul "terbuka" lainnya.

Navigasi jenis ini lebih situasional dibandingkan dengan kedua jenis sebelumnya. Jenis ini diterapkan dalam keadaan di mana agen membutuhkan jarak tertentu terhadap suatu objek untuk melakukan suatu aksi. Contoh sederhana dari keadaan ini adalah karakter yang melakukan serangan jarak jauh terhadap karakter lain.

IV. Kesimpulan

Algoritma A* merupakan algoritma pencarian lintasan yang tidak hanya efisien, namun juga serbaguna. Dengan memilih fungsi heuristik $\hat{h}$ tertentu, A* dapat disesuaikan untuk memecahkan berbagai masalah pencarian lintasan yang berbeda. Perlu diperhatikan bahwa lintasan yang dihasilkan bergerak dari simpul dengan nilai $\hat{h}$ tinggi ke simpul-simpul dengan nilai $\hat{h}$ rendah. Maka, untuk memilih $\hat{h}$ yang sesuai untuk penerapan tertentu yang diinginkan, analisis diperlukan untuk menentukan definisi $\hat{h}$ yang memberikan bias terkuat di arah simpul-simpul tujuan.

Banyak dari masalah pencarian lintasan kompleks ditemui dalam penerapan kecerdasan buatan dalam permainan video. Sistem navigasi dan pencarian lintasan yang dibangun dengan baik merupakan salah satu faktor terbesar dalam membentuk agen yang memiliki perilaku unik yang dapat menanggapi tindakan pemain atau lingkungan dengan cerdas dan realistis.

V. Ucapan Terima Kasih

Puji dan syukur penulis panjatkan kepada Tuhan Yang Maha Esa; oleh karena kasih karunia dan penyertaan-Nya penulis dapat menyelesaikan makalah berjudul "Implementasi Algoritma A* dalam Navigasi Karakter Nonpemain Melalui Rintangan Dua Dimensi dalam Permainan Video" ini dengan baik. Penulis mengucapkan terima kasih kepada semua pihak yang terlibat dalam proses penyusunan makalah ini, yaitu:

1. Ibu Dr. Nur Ulfa Maulidevi, S.T., M.Sc., Bapak Dr. Rinaldi Munir, M.T., serta Ibu Dr. Fariska Zahra Zakhralativa Ruskanda, S.T., M.T., selaku dosen-dosen pengajar mata kuliah IF2120 Matematika Diskrit, atas bimbingan dan ilmu yang dibagikan kepada penulis melalui kegiatan perkuliahan,
2. Kedua orang tua penulis, atas dorongan dan dukungan kepada penulis senantiasa,
3. Sahabat dan teman-teman penulis, atas dukungan dan kontribusi dalam penyusunan dan penyempurnaan makalah ini,
4. Unity Technologies, yang telah menyediakan sarana bagi penulis untuk melakukan uji coba, penerapan, serta visualisasi, dan
5. Para penulis jurnal, artikel, dan buku, yang karyanya penulis jadikan sebagai sumber informasi di sepanjang proses penulisan.

Akhir kata, penulis mengucapkan terima kasih kepada pembaca. Penulis memohon maaf bila ada kesalahan, baik dalam penulisan maupun muatan. Besar harapan penulis bahwa makalah ini dapat menjadi manfaat bagi pembaca.

Daftar Pustaka

[1] Wolf, Mark. 2008. The Video Game Explosion: A History from Pong to PlayStation and Beyond. Westport, CT: Greenwood Press.

[2] Bogost, Ian. 2017. "Artificial Intelligence" Has Become Meaningless. https://www.theatlantic.com/technology/archive/2017/03/what-is-artificial-intelligence/518547/ diakses pada 12 Desember 2022, pukul 1:35 WIB.

[3] Yannakakis, Georgios. 2012. Game AI Revisited. Proceedings of the 9th Conference on Computing.

[4] Maret 1996. The Next Generation 1996 Lexicon A to Z. Next Generation, No. 15. Brisbane, CA: Imagine Media.

[5] Munir, Rinaldi. 2010. Matematika Diskrit. Edisi 3. Bandung: Informatika Bandung.

[6] Hart, Nilsson dkk. 1968. A Formal Basis for the Heuristic Determination of Minimum Cost Paths. IEEE Transactions of Systems Science and Cybernetics, Vol. 4.

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, 3 Desember 2020

Jericho Russel Sebastian – 13521107