PENCARIAN JALUR TERPENDEK PADA SNAKE GAME MENGGUNAKAN ALGORITMA A*

Masri, Ari Tri Mukti

Sekolah Tinggi Manajemen Informatika dan Komputer Pontianak
Jalan Merdeka Barat No. 327 Kalimantan Barat
[1]Masfight@gmail.com, [2]Aritrimukti@myself.com

Abstrak

Game sebagai media hiburan telah berkembang dengan pesatseiring juga teknologi. Salah satu unsur yang berperan penting dalam sebuah game adalah kecerdasan buatan. Dengan kecerdasan buatan, diharapkan elemen-elemen permainan Ular (Snake) ini penulis menerapkan konsep logika fuzzy dalam menentukan starategi pada ular.Dalam penelitian ini, penulis menggunakan bentuk penelitian studi litelatur.Penulis menggunakan metode perancangan Agile Software Development.Di dalam perancangan game ini, penulis menggunakan perangkat lunak Visual basic 6.0 dan Algoritma A-Star.Algoritma A Star (A) pada komputer dalam permainan ini teruji sangat efektif dalam mendapatkan makanannya dengan jalur terpendek.Komputer menggunakan Algoritma A Star (A*) untuk mencari makanannya. Algortima A Star (A*) adalah algoritma pencarian terbaik dalam mencari jalur terpendek dengan perhitungan terkecil pada jalur dengan simpul awal menuju simpul akhir.*
Kata Kunci :*Permainan Ular, Algoritma A*, Agile Software Development.*

Abstract

*Games as a medium of entertainment have grown rapidly as well as technology. One element that plays an important role in a game is the artificial intelligence. With artificial intelligence, expected elements of the game Snake (Snake) The authors apply the concept of fuzzy logic in determining starategi snakes. In this study, the authors used a form of research studies litelatur. The author uses the method of designing Agile Software Development. In the design of this game, the author uses Visual Basic 6.0 software and A-Star Algorithm. Algorithm A Star (A *) on the computer in this game proven very effective in getting the food to the shortest path. Computer Algorithm using A Star (A *) to search for food. Algorithms A Star (A *) is the best search algorithm in finding the shortest path with the smallest calculation on track with the start node to the end node.*
Keywords: Snake Game, A Algorithm, Agile Software Development.*

1. PENDAHULUAN

Permainan ular atau snake game merupakan permainan populer dalam telepon selular beberapa tahun yang lalu. Inti dari permainan ini adalah agar ular yang pemain kontrol mendapatkan sebanyak-banyaknya makanan tanpa membentur dinding atau bagian tubuhnya sendiri. Semakin banyak makanan yang ular dapatkan, tubuhnya akan tumbuh sehingga semakin panjang.

Permainan ini sebenarnya merupakan permainan pemain tunggal (single player game) atau permainan yang dimainkan sendiri. Dalam permainan ini komputer hanya memunculkan makanan secara acak (random) di layar untuk dimakan. Kesuksesan permainan ini bergantung kepada kecepatan dan perhitungan sang pemain agar ular yang dikontrolnya tidak terjebak dinding atau bagian tubuhnya sendiri. Kelakuan seperti inilah yang ingin dicoba dalam permainan ular ini dengan memunculkan karakter kecerdasan buatan komputer pada lawan

pemain yang mampu berlaku sebagai pemain umumnya.

Game snake konvensional memang telah ada dan banyak digunakan oleh para penggemar game (gamer) dan sangat popular beberapa tahun yang lalu, bahkan game snake konvensional pada masa kejayaanya dahulu memiliki pengguna terbanyak berdasarkan catatan Guines Record.

Game ular konvensional saat ini belum terlalu banyak memiliki Artificial Intelligence (AI) dalam implementasinya, baik AI yang ada pada karakter ular itu sendiri maupun AI yang ada pada umpan. AI yang ada pada game snake konvensional akan dikembangkan sehingga diharapkan menjadi tantangan tersendiri bagi pengguna. Selanjutnya game snake konvensional belum memiliki alur permainan yang baik, sehingga kurang memberi motivasi bagi pengguna untuk menyelesaikan permainannya. Penggunaan navigasi kontrol game snake konvensional juga belum bisa dilakukan secara bersamaan (combine key), dimana pergerakkannya hanya bisa vertikal (atas dan bawah) dan horizontal (kiri dan kanan), serta tidak bisa bergerak secara diagonal. Keterbatasan dari penggunaan kontrol navigasi tersebut membuat ular tidak bisa jadi lebih maksimal dalam memperoleh umpan. Diharapkan dengan adanya pengembangan terhadap game snake konvensional nanti, membuat pengguna game snake memiliki pengalaman yang menyenangkan (joyfull gaming) serta tidak cepat merasakan bosan.

Algoritma AI yang dipilih untuk memodelkan kelakuan ular ini adalah Algoritma A-Star (A*) yang merupakan algoritma pencarian jalur terpendek terbaik menuju tujuan. Algoritma ini sangat cocok karena algoritma ini mendukung perhitungan untuk mencari jalan terpendek menuju makanan sementara menghindari bertabrakan dengan dinding atau bagian tubuhnya atau membuat dirinya sendiri terkurung.

2. METODE PENELITIAN

Dalam penelitian ini, penulis menggunakan bentuk penelitian studi literatur.Penulis melakukan kajian yang berkaitan erat dengan permasalahan yang hendak dipecahkan serta mendefinisikan masalah dengan melakukan eksperimen. Penulis menggunakan metode perancangan RAD (Rapid Application Development) karena proses perkembangan perangkat lunak ini menekankan pada siklus perkembangan yang singkat dan pemanfaatan fungsi yang telah ada sebelumnya. Pengumpulan data adalah tahap cukup menentukan dalam proses penelitian, karena dengan pengumpulan data yang tepat maka diharapkan jawaban dari perumusan masalah tidak bisa. Data yang dikumpulkan sesuai dengan tujuan penelitian. Dalam mengumpulkan data penulis menggunakan metode studi literatur dan dokumentasi, yaitu dokumentasi data yang berisi definisi-definisi dari item-item data, termasuk di dalamnya semua variabel-variabel yang digunakan dalam proses perancangan aplikasi problem solving menggunakan algoritma A* (A-Star).

3. HASIL DAN PEMBAHASAN

Pembangunan perangkat lunak terdiri dari beberapa tahap, salah satunya adalah tahap menganalisa sistem. Sistem yang akan dianalisis untuk membangun perangkat lunak permainan snake menggunakan algoritma A* (A-Star). Analisis sistem yang akan dibahas mengenai cara kerja algoritma A* (A-Star) yang diimplementasikan pada permainan snake.

Sejatinya permainan ini merupakan single player game atau permainan yang dimainkan sendiri. Dalam permainan ini komputer akan memunculkan makanan secara random di layar untuk dimakan. Kesuksesan permainan ini bergantung kepada kecepatan dan perhitungan sang pemain agar ular yang dikontrolnya tidak terjebak dinding atau bagian tubuhnya sendiri.

Gameplay dari Snake game ini dapat dimainkan oleh Komputer dan pemain. Komputer yang bermain diatur menggunakan algoritma A* (A-Star). Tugas utama pemain (gamer) dalam memainkan Aplikasi ini yaitu bertahan hidup agar pemain tidak menabrak dinding atau tubuh sendiri, pemain disini digambarkan dengan karakter ular. Pemain dapat memakan makanan sebanyak-banyaknya sehingga pemain memperoleh score (nilai).

Pemain (gamer) dapat memulai permainan dengan memilih pilihan "Mulai" dimana nantinya pemain dapat langsung memainkan ular (snake), dan juga disediakan pilihan "Nilai Tertinggi" yaitu pilihan untuk melihat score-score yang sudah tercetak sebagai hasil dari permainan yang telah dilakukan oleh pemain dan pilihan lainnya adalah pilihan "About", sebagai keterangan yang menyediakan informasi mengenai pembuat game ini, dan juga pemain dapat memilih "How to Play", sebagai keterangan bagaimana cara bermain game snake ini.

Algoritma yang dipakai dalam pemodelan ini adalah algoritma A* (A-Star). Algoritma ini sejatinya adalah algoritma pencarian solusi dengan pencarian melebar atau breadth first search (BFS). Dalam algoritma BFS solusi dicari dengan membentuk pohon ruang status yang merupakan pohon dinamis.

Simpul-simpul di dalam pohon diinamis yang memenuhi kendala menyatakan status persoalan. Suatu operator mentransformasikan persoalan dari sebuah status ke status yang lain. Solusi persoalan dinyatakan dengan satu atau lebih status yang disebut status solusi. Status solusi yang merupakan simpul daun disebut status tujuan. Himpunan semua status solusi disebut ruang solusi. Seluruh simpul di dalam pohon dinamis disebut ruang status. Dan pohonnya dinamakan juga pohon ruang status. Akar pada pohon ruang status menyatakan status awal sedangkan daun menyatakan status solusi.

BFS mencari solusi persoalan pada pohon ruang status yang dibentuk secara dinamis dengan cara semua simpul pada aras d dibangkitkan terlebih dahulu sebelum simpul-simpul pada aras d+1. Simpul BFS memerlukan sebuah antrian untuk menyimpan simpul-simpul yang akan dibangkitkan. Simpul-simpul yang dibangkitkan disimpan di belakang antrian.

Pada algoritma A* (A-Star), pencarian ke simpul solusi dapat dipercepat dengan memilih simpul hidup berdasarkan nilai ongkos (cost). Setiap simpul hidup diasosiasikan dengan sebuah ongkos yang menyatakan nilai batas (bound). Batas ini dapat berupa batas bawah (lower bound) ataupun batas atas (upper bound) masing-masing menyatakan batas maksimum atau batas minimum yang diperlukan untuk membangkitkan sebuah simpul.

Pemberian nilai batas akan ideal jika kita mengetahui letak simpul solusi, namun untuk kebanyakan persoalan letak simpul solusi tidak diketahui. Sehingga nilai batas untuk setiap simpulnya umumnya hanya berupa taksiran ataupun perkiraan.

Untuk membantu pemberian taksiran nilai umumnya digunakan fungsi heuristik yang dapat berupa fungsi apapun asalkan fungsi tersebut mampu memodelkan ongkos (cost) untuk membantu mencapai simpul solusi. Fungsi heuristik untuk menghitung taksiran nilai tersebut dinyatakan secara umum sebagai:

$c(i) = f(i) + g(i)$

dengan

c(i) = ongkos untuk simpul i

f(i) = ongkos mencapai simpul i dari akar

g(i) = ongkos mencapai simpul tujuan dari simpul (i)

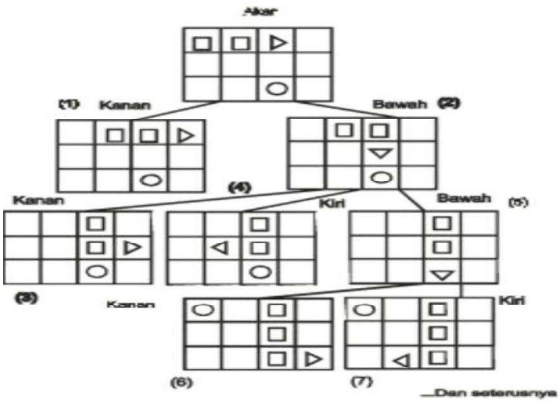
Nilai c digunakan untuk mengurutkan pencarian. Jika pencarian adalah pencarian dengan biaya terkecil maka simpul berikutnya yang dipilih untuk diekspansi adalah simpul yang memiliki c minimum. Bila sebaliknya, simpul yang diekspansi adalah simpul yang memiliki c maksimum.

Skema umum algoritma A* (A-Star) adalah sebagai berikut :

a. Masukkan simpul akar ke dalam antrian Q. Jika simpul akar adalah simpul solusi maka solusi telah ditemukan. Stop.
b. Jika Q kosong, tidak ada solusi. Stop.

c. Jika Q tidak kosong, pilih dari antrian Q simpul i yang mempunyai c(i) paling kecil (jika least cost search) atau c(i) paling besar (jika most cost search)
d. Jika simpul i adalah simpul solusi, berarti solusi sudah ditemukan, stop. Jika simpul i bukan simpul solusi, maka bangkitkan semua anakanaknya. Jika i tidak mempunyai anak, kembali ke langkah 2.
e. Untuk setiap anak j dari simpul i, hitung c(j) dan masukkan semua anak- anak tersebut ke dalam antrian Q.
f. Kembali ke langkah 2.

Untuk memodelkan perilaku snake ini diperlukan beberapa modifikasi karena algoritma A* (A-Star) murni belum memadai untuk memodelkan perilaku snake. Modifikasi pertama adalah tujuan (goal) dari algoritma tersebut. Pada A* (A-Star) murni, algoritma mencari simpul tujuan dengan sesedikit mungkin membangkitkan simpul anak. Sedangkan tujuan dari permainan ini adalah kita bertahan selama mungkin dalam permainan snake ini sehingga tujuan utama dalam algoritma ini adalah membuat simpul sedalam mungkin. Namun simpul yang dibangkitkan harus menjamin bahwa snake tidak mati. Simpul yang dibangkitkan pula harus menjamin bahwa snake berhasil mendapatkan makanan dengan jumlah langkah yang sesedikit mungkin.



Gambar 1 Pembangkitan simpul yang diinginkan dalam permainan snake

Dalam gambar ini diperlihatkan pembangkitan simpul yang diinginkan dalam permainan snake. Simbol kotak melambangkan bagian tubuh dari snake. Simbol segitiga melambangkan kepala dan arah gerak snake. Simbol lingkaran melambangkan makanan yang harus diambil snake.

Pada akar terlihat kondisi awal saat permainan pertama dimulai. Simpul yang bisa dibangkitkan adalah simpul dengan gerak ular selanjutnya adalah ke kanan atau ke bawah karena simpul dengan gerak ular ke atas atau ke kiri dapat mengakibatkan ular mati.

Pada arah kedua simpul yang ingin diteruskan adalah simpul dengan gerak ular ke bawah karena makanan berada di bawah sehingga simpul tersebut merupakan awal dari simpul dengan ongkos termurah untuk mencapai makanan. Kemudian pada aras terakhir dengan model AI ini ular akan bergerak ke kanan dan seterusnya sehingga ular tidak bisa bergerak lagi.

Untuk memodelkan perilaku yang diinginkan seperti di atas diperlukan fungsi pembangkitan ongkos yang sesuai. Fungsi pembangkitan ongkos ini merupakan kunci dari algoritma A* (A-Star) karena algoritma ini akan membangkitkan simpul sesuai dengan ongkos simpul tersebut.

Dalam hal ini fungsi heuristik yang digunakan adalah :

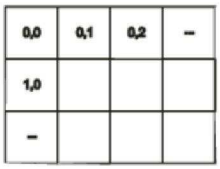
$c(i) = f(i) + g(i) + h(i)$

dengan

c(i) = ongkos untuk simpul i

f(i) = jarak dari kotak yang dituju dengan simpul i ke makanan (jarak dihitung dengan rumus jarak dua titik dalam bidang koordinat kartesius)
g(i) = ongkos jika simpul i mengarah ke dinding atau kebagian tubuh dari ular, bernilai sangat besar karena kita ingin menghindari tabrakan
h(i) = adalah ongkos yang dibangkitkan jika ular masuk ke wilayah yang dibuatnya.
Simpul yang dibangkitkan adalah simpul dengan c(i) terkecil.

Dalam permainan ular ini setiap kotak mempunyai koordinat tertentu.



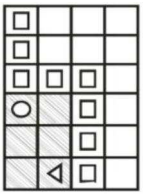
Gambar 1 Aturan koordinat bidang permainan snake

Sehingga pada arah kedua di gambar 1 nilai f(i) untuk bergerak ke kanan adalah

$$f(1) = \sqrt{(2-0)^2 + (2-3)^2} = \sqrt{5}$$

Namun karena nilai akar cenderung mempersulit perhitungan maka ongkos untuk f(2) dapat dikuadratkan sehingga menjadi : *f(1) = 5*. Untuk nilai g(1) adalah 0 karena simpul g(1) tidak menabrak dinding. Namun, tinjau apabila simpul yang bergerak ke atas dicoba dibuat dari simpul akar maka nilai g untuk simpul tersebut adalah 99999 nilai ini sangat besar karena kita tidak ingin ular bergerak menabrak dinding. Nilai g untuk simpul yang akan menabrak tubuh ular juga akan bernilai 99999.

Untuk nilai h tinjau simpul 7 pada simpul tersebut ular sudah membuat wilayah yaitu segi empat dari koordinat (0,0) sampai koordinat (2,1). Nilai h akan dibangkitkan berdasarkan wilayah yang sudah dibuat tersebut. Pembangkitan nilai h ini harus dilakukan dengan hati-hati karena kita tidak ingin ular terkurung dalam wilayah yang dibuatnya sendiri. Jika ular sudah terkurung maka ular akan mati dan hal ini kita hindari. Untuk definisi lengkap pembangkitan nilai h tinjau kasus berikut :



Gambar 4.3 Contoh kasus pembangkitan nilai h

Pada kasus ini ular sudah membuat wilayah yaitu kotak-kotak yang diarsir. Jumlah wilayah yang dimiliki adalah 9 karena perhitungan wilayah berdasarkan kuadrat panjang atau lebar tertinggi. Hal ini agar gerakan ular selalu mengeliminasi wilayah dengan gerakan spiral. Untuk gerakan selanjutnya, nilai h harus dihitung. Nilai h dihitung berdasarkan perhitungan berikut:

h(i) = l - j(i) + k(i)

dengan

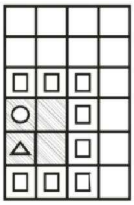
j(i) = jarak dari kotak yang dicapai oleh simpul i dengan ekor ular k(i) = wilayah yang dibuat jika ular bergerak ke simpul i
l = panjang badan ular

Pada kasus seperti gambar tiga simpul yang bisa dibangkitkan adalah simpul dengan arah gerak ke kiri atau ke atas mari kita tinjau cost untuk kedua simpul tersebut.

f(atas) = 2
g(atas) = 0
h(atas) = 9-16 + 9
c(atas) b f(atas) + g(atas) + h(atas) = 4

f(kiri) = 2
g(kiri) = 0
h(kiri) = 9-16 + 4
c(kiri) = f(kiri) + g(kiri)+h(kiri) = 1

Dari perhitungan tersebut ular akan bergerak ke kiri. Dari sana simpul yang bisa dibangkitkan hanyalah simpul dengan gerakan ular ke atas, maka ular akan ke atas. Sehingga posisi ular akan seperti :



Gambar 4.4 Contoh kasus pembangkitan nilai h

Dari posisi ini ular bisa bergerak ke atas atau ke kanan. Namun bila ular bergerak ke atas maka ular akan mati karena ular akan bertambah panjang bila memakan makanan dan akan terbentur oleh tubuhnya sendiri. Dalam dua kasus ini tinjau perhitungan berikut:

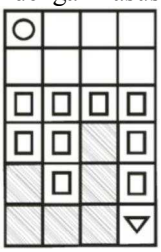
f(atas) = 0
g(atas) = 0
h(atas) = 9-1 + 4
c(atas) = f(atas) + g(atas) + h(atas) = 12

f(kanan) = 2
g(kanan) = 0
h(kanan) = 9 – 5 + 4
c(kanan) = f(kanan) + g(kanan) + h(kanan) = 10

Dengan perhitungan seperti ini, ular akan bergerak ke arah kanan dan ular akan selamat dari jebakan. Selanjutnya ular akan bergerak ke atas kemudian ke kanan.

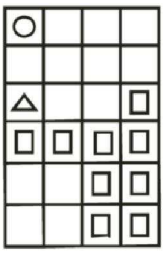
Fungsi heuristik ini cukup efektif untuk menghindari kejadian seperti ini. Namun tidak selamanya fungsi h dihitung karena fungsi h akan dihitung jika dan hanya jika ular sudah mempunyai wilayah. Ular memiliki wilayah jika dan hanya jika dia sudah "mengurung" sejumlah kotak atau dengan kata lain bagian tubuhnya sudah menempel dengan dua dinding yang tidak berseberangan. Untuk kasus belum memiliki wilayah fungsi f dan g sudah cukup mewakili.

Pengetesan lebih lanjut dilakukan dengan kasus berikut :



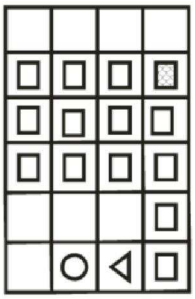
Gambar 4.5 Contoh kasus untuk pengetesan

Dengan algoritma ini langkah-langkah ular seperti berikut : Langkah pertama ular akan bergerak ke kiri karena itu adalah satu-satunya langkah yang mungkin. langkah berikutnya adalah ke atas, lalu ke atas lagi, kemudian ke kiri, alalu ke kiri, kemudian ke atas sehingga status ular tersebut akan seperti :



Gambar 4.6 Hasil pengetesan

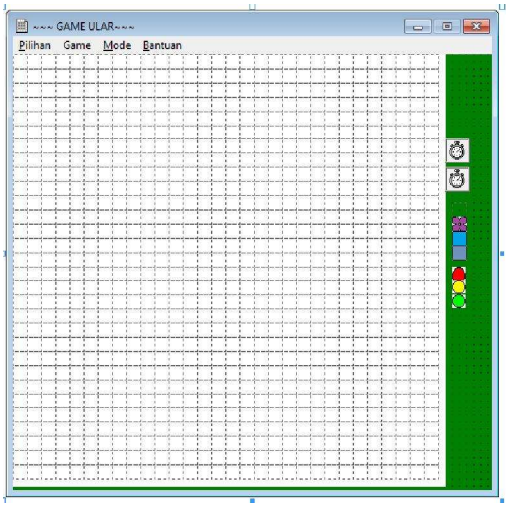
Namun untuk panjang ular yang sangat besar, algoritma ini gagal memberikan hasil yang memuaskan karena langkah-langkah awal sebelum ular memiliki wilayah mengakibatkan ular terkurung di wilayahnya sehingga apapun kombinasi pergerakan tidak dapat membuat ular dapat keluar dari wilayah tersebut. Hal itu ditemukan dalam kasus berikut :



Gambar 4.7 Contoh kasus gagal

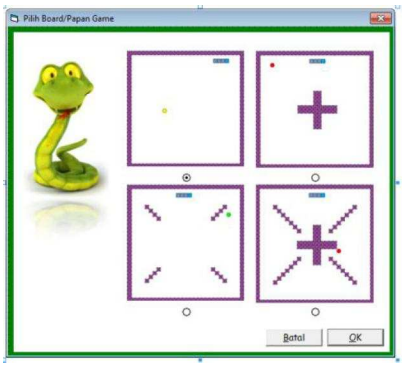
Algoritma ini tidak bisa mencegah agar ular tidak mengalami keadaan seperti ini karena panjang ular yang besar seringkali mengganggu perhitungan dan dalam kasus belum memiliki wilayah algoritma ini hanya mencari jarak terdekat ke makanan sehingga gerakan pada awal-awal terkesan "ceroboh" karena tidak memikirkan akibat dari gerakan tersebut.

Perangkat lunak penerapan algoritma A* (A-Star) pada permainan snake game dirancang dengan menggunakan bahasa pemrograman *Microsoft Visual Basic 6.0* dengan beberapa komponen standar dan memiliki beberapa *form*, seperti : *FormMain*, *Form Board*, *Form How To Play, Form* About. Pada form Main terdapat 2 bagian area yaitu area snake game sebagai area untuk pemain atau komputer memainkan permainan snake, sedangkan area yang berisikan menu dan sub menu. Gambar 4 berikut merupakan desain form Main.



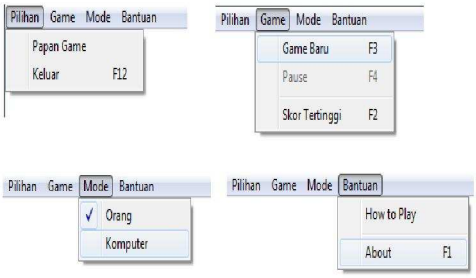
Gambar 3 Rancangan Form Main

Pada *form* ini pengguna perangkat lunak dapat mengatur jenis papan snake yang digunakan dalam permainan snake. Pada *form* ini, pengguna cukup memilih jenis papan atau board yang diinginkan dan menyimpan setting tersebut dengan menekan tombol OK. Gambar 5 berikut merupakan desain form Board.



Gambar 4 Rancangan Form Board

Pada bagian ini, penulis membuat sejumlah menu yang sudah tersusun dalam beberapa menu dan sub menu. Berikut ini hasil rancangan dari menu dan sub menu pada permainan snake.



Gambar 5 Tampilan Perancangan menu

Pada pengujian penekanan papan kunci menggunakan metode kotak hitam (*black box*) yang artinya aplikasi dihadapkan pada suatu kondisi kemudian melihat hasilnya. Tujuannya adalah untuk memeriksa respon dari penekanan papan kunci pada aplikasi apakah sudah sesuai atau belum. Tabel 1 menunjukkan hasil pengujian pada penekanan papan kunci.

Tabel 1. Pengujian penekanan papan kunci

Masukan	Keluaran	
	Keluaran diharapkan	Keluaran sistem
↑	Jalan ke arah atas	Jalan ke arah atas
↓	Jalan ke arah bawah	Jalan ke arah
←	Jalan ke arah kiri	Jalan ke arah kiri
→	Jalan ke arah kanan	Jalan ke arah

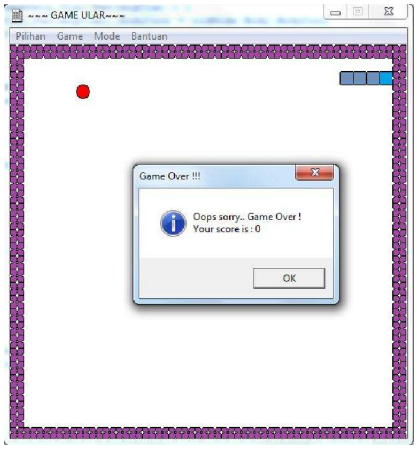
Dari tabel diatas terlihat bahwa hasil penekanan papan kunci sudah sesuai

dengan keluaran yang diharapkan.

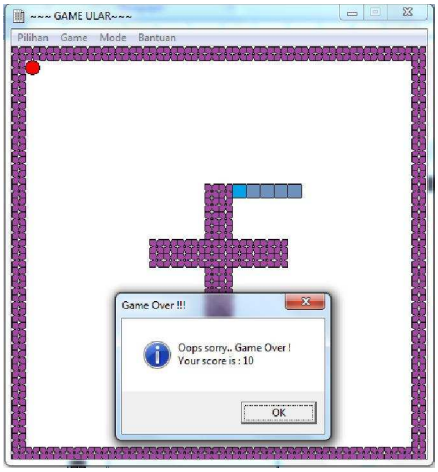
Terdapat 3 deteksi tabrakan dalam permainan ular ini yaitu:

a. Deteksi tabrakan ular dengan batas arena
b. Deteksi tabrakan ular dengan penghalang jalan.
c. Deteksi tabrakan ular dengan tubuhnya sendiri.

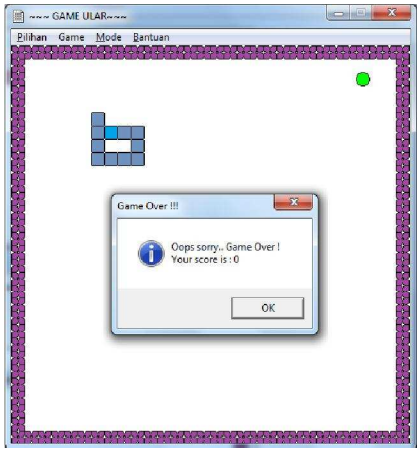
Pada Gambar di bawah ini adalah gambar menunjukkan ular mengenai batas arena dan tabrakan terjadi sehingga membuat ular kalah dan menjadi penghalang jalan berwarna gelap. Deteksi tabrakan sesuai dengan yang diharapkan.



Gambar 6 Tabrakan Object dengan Batas Arena



Gambar 7 Tabrakan Object Ular dengan Penghalang



Gambar 8 Tabrakan Object Ular dengan tubuhnya sendiri.

4. KESIMPULAN

Komputer menggunakan Algoritma A Star (A*) untuk mencari makanannya.Algortima A Star (A*) adalah adalah algoritma pencarian terbaik dalam mencari jalur terpendek dengan perhitungan terkecil pada jalur dengan simpul awal menuju simpul akhir. Algoritma A Star (A*) pada komputer dalam permainan ini teruji sangat efektif dalam mendapatkan makanannya dengan jalur terpendek.

5. SARAN

Dari hasil penelitian dan implementasi yang telah dilakukan, disarankan pengembangan aplikasi lebih lanjut, dengan aspek pengembangan lanjut lagi seperti berikut:

a. Menambahkan fitur Kontrol navigasi ular, dimana awalnya hanya bisa bergerak vertikal dan horizontal saja, tetapi juga bisa bergerak diagonal yang membuat entitas ular jadi lebih maksimal dalam pencarian umpan.
b. Entitas musuh (ular lain yang diterapkan AI A*) sudah ada, sehingga menjadi tantangan dan motivasi bagi pengguna game snake dalam bermain

DAFTAR PUSTAKA

[1]. Hasibuan, Zainal A., 2007, Metodologi Penelitian Pada Bidang Ilmu Komputer Dan Teknologi Informasi, Konsep: Metode Teknik dan Aplikasi, Depok.
[2]. Jokodo., 2011, Game puzzle number. Jakarta :Universitas Gunadharma
[3]. Perry, William E., 2006, Effective Methods for Software Testing, edisi ketiga, Wiley Publishing, Inc, IN.
[4]. Suyanto, 2011, Artificial Intelligence.Bandung : Penerbit INFORMATIKA
[5]. Yuliana., Ananda., Surya, Ibnu., 2012, Implementasi Algoritma A-Star Pada Pemecahan Puzzle 8, Jurnal Teknik Informatika, Vol. 1, September 2012, Hal. 1-9.