BAB IV

IMPLEMENTASI DAN PENGUJIAN

Pada bab ini akan dilakukan implementasi dan pengujian terhadap aplikasi yang dibangun. Tahapan ini dilakukan setelah analisis dan perancangan selesai dilakukan dan selanjutnya akan diimplementasikan kedalam bahasa pemograman. Setelah implementasi maka dilakukan pengujian terhadap aplikasi. Aplikasi yang telah dibangun akan diimplementasikan untuk mengetahui apakah sistem tersebut dapat berjalan sesuai dengan tujuannya atau tidak.

IV.1 Implementasi

Tujuan implementasi adalah untuk menerapkan perancangan yang telah dilakukan terhadap sistem sehingga *user* dapat memberi masukan demi berkembangnya sistem yang telah dibangun sebagai simulasi dari aplikasi penerapan algoritma A* (*A star*) untuk pencarian jalan terpendek pada *game pathfinding*. Perangkat keras dan perangkat lunak yang digunakan untuk melakukan implementasi dan pengujian aplikasi yaitu sebagai berikut :

a. Perangkat keras

Perangkat keras yang digunakan untuk pembuatan aplikasi dijelaskan pada tabel IV.1.

Tabel IV.1. Perangkat keras yang digunakan

Komputer	Spesifikasi perangkat keras
Prosesor	Intel core i3 2.27 Ghz
Layar	Resolusi 1378 x 768
Monitor	LCD 14 inc
Memori	Memori 4 GB
Harddisk	Harddisk 320 GB
Keyboard dan mouse	-

b. Perangkat lunak

Perangkat lunak yang digunakan untuk mengimplementasikan sistem dijelaskan pada tabel IV.2.

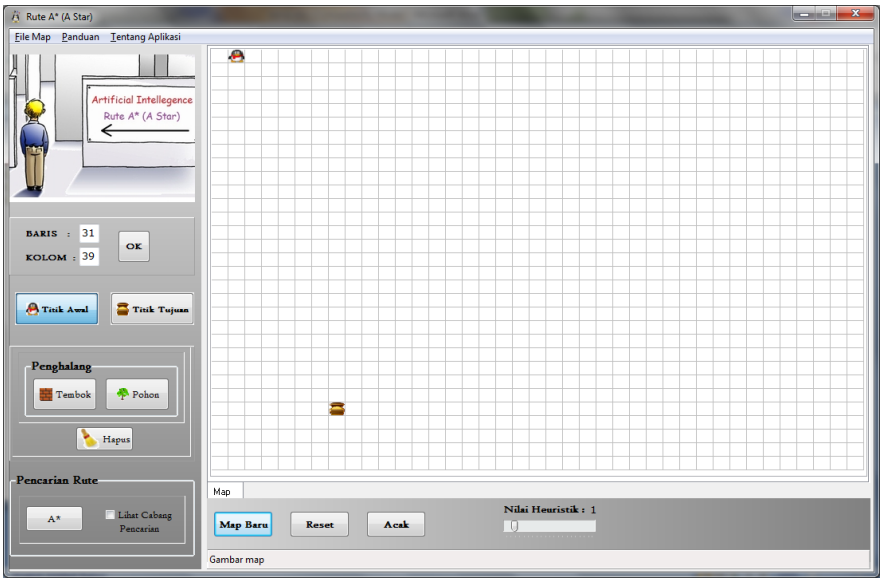
Tabel IV.2. Perangkat lunak yang digunakan

Komputer	Spesifikasi perangkat lunak
Sistem Operasi	Microsoft Window 7 ultimate 32 bit
Bahasa Pemograman	Borland Delphi 7.0

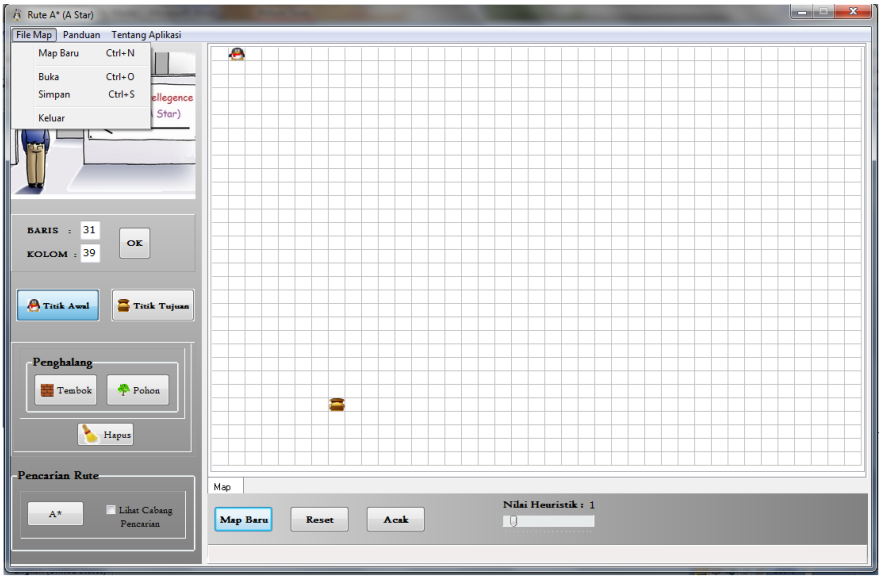
Sistem operasi pada tabel IV.2 harus sudah terinstal pada komputer, agar aplikasi yang dibangun dapat dijalankan pada komputer tersebut. Jika pengguna ingin membuka *project*-nya maka terlebih dahulu harus menginstalkan Borland Delphi dan *component* yang mendukung. *File Rute_Astar.exe* dapat digunakan untuk mengimplementasikan aplikasi yang telah dibangun.

IV.1.1 Implementasi Antarmuka

Implementasi antarmuka untuk perancangan aplikasi tampilan utama dan menu *file map* yang dibangun dapat dilihat dari gambar IV.1 dan gambar IV.2.



Gambar IV.1. Implementasi tampilan utama

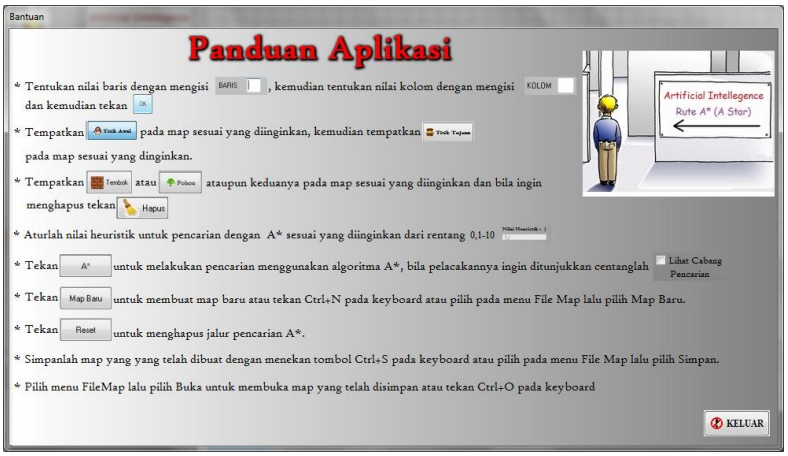


Gambar IV.2. Implementasi tampilan menu

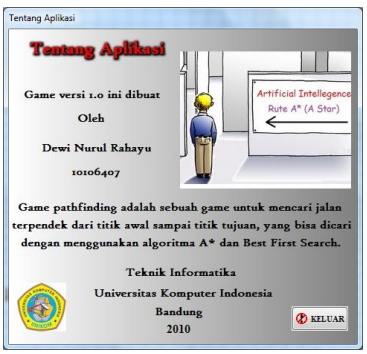
Didalam aplikasi ini terdapat beberapa menu yang terbagi menjadi tiga yaitu menu *file map*, panduan, dan tentang aplikasi. Pada menu *file map* terdapat sub menu yang terdiri dari :

1. Map baru Ctrl+N
2. Buka Ctrl+O
3. Simpan Ctrl+S
4. Keluar

Menu panduan yaitu bantuan untuk pengguna melakukan semua proses yang ada pada aplikasi antarmukanya dapat dilihat pada gambar IV.3, sedangkan tentang aplikasi dapat dilihat pada gambar IV.4.



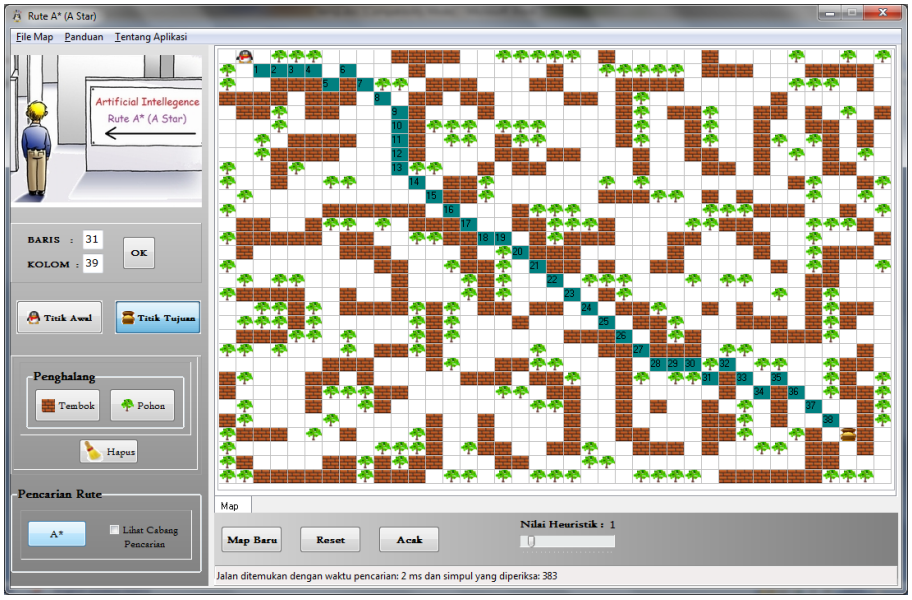
Gambar IV.3. Implementasi antarmuka panduan



Gambar IV.4. Implementasi antarmuka tentang aplikasi

IV.1.2 Implementasi Contoh Kasus Jalur A*

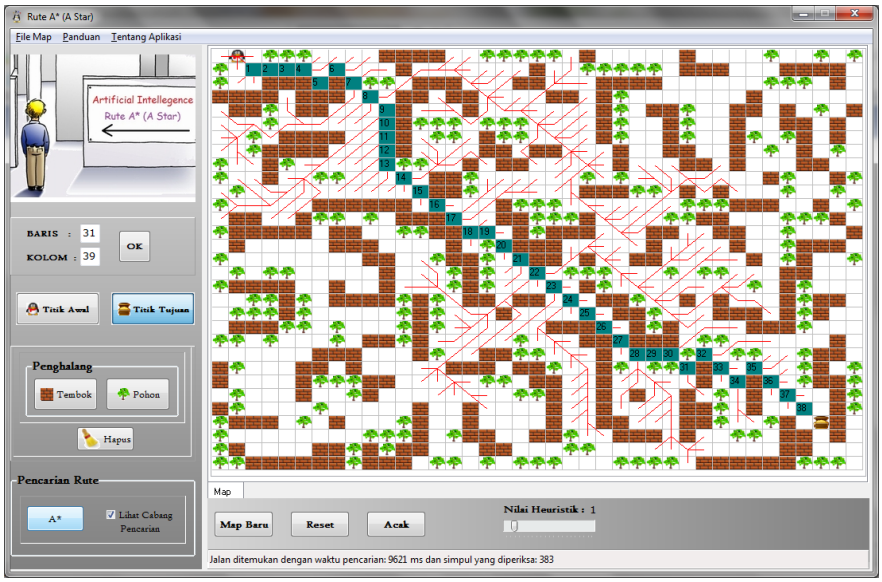
Salah satu contoh implementasi simulasi yang akan digunakan untuk menemukan jalur yaitu dengan menggunakan penghalang dan tanpa penghalang. Pertama setelah pengguna menempatkan titik awal dan titik akhir pada posisi yang diinginkan, dan sudah menempatkan penghalang pada *map* maka pencarian *rute* dapat dilakukan dengan memilih pencarian dengan A* dan lihat cabang pencarian. Implementasi jalur dengan algoritma A* tanpa lihat cabang pencarian dapat terlihat pada gambar IV.5 dengan hasil jalan ditemukan dengan waktu pencarian 2 ms dan simpul yang diperiksa 383.



Gambar IV.5. Implementasi *rute* algoritma A* *search*

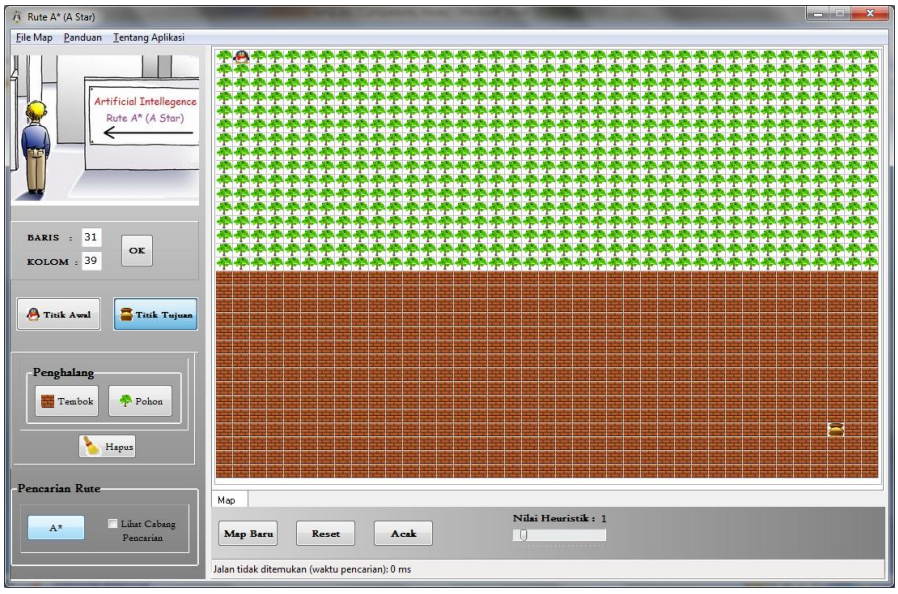
Pengujian algoritma A* dengan memakai ‘lihat cabang pencarian’ dapat menghasilkan waktu 9621 ms dan simpul yang diperiksa 383. Waktu dengan menggunakan ‘lihat cabang pencarian’ relatif lama dibandingkan tanpa ‘lihat cabang pencarian’ karena setiap simpul yang diperiksa akan memperlihatkan

pelacakannya dari titik awal ke titik tujuan dengan waktu. Implementasi ‘lihat cabang pencarian’ tersebut dapat dilihat pada gambar IV.6.



Gambar IV.6. Implementasi lihat cabang pencarian *rute* A*

Pencarian jalan dengan ordo dan penghalang maksimal serta titik awal dengan titik tujuan saling berjauhan maka akan menghasilkan jalan tidak ditemukan dengan waktu pencarian 0 ms yang terlihat pada gambar IV.7.



Gambar IV.7. Implementasi pencarian *rute* A* jalan tidak ditemukan

Dengan hasil uji yang telah dilakukan dapat dilihat hasil pencarian dengan menggunakan algoritma A* yaitu waktu pencarian, simpul yang diperiksa dan jarak *rute* yang dihasilkan. Pada algoritma A* jarak *rute* yang dihasilkan pendek, tapi dalam hal simpul yang diperiksa A* akan menghasilkan simpul yang besar sehingga akan memakan *memory* yang lebih besar dan waktu yang lama.

IV.2. Pengujian

Pengujian pada sistem yang baru dibangun dilakukan melalui dua tahap yaitu pengujian *alpha* (fungsional) dan pengujian *betha*.

a. Pengujian *Alpha*

Metode yang digunakan dalam pengujian *alpha* adalah metode *blackbox* yang fokus pada persyaratan fungsional dari perangkat lunak yang dibangun.

b. Rencana Pengujian Program Simulasi

Rencana pengujian yang akan dilakukan pada aplikasi dengan menggunakan metode *blackbox* dapat dilihat pada tabel IV.3.

Tabel IV.3. Rencana pengujian program simulasi

Item uji	Detail pengujian
Pengaturan *grid*	Nilai baris
	Nilai kolom
	OK
Pengaturan titik awal dan titik tujuan	Posisi titik awal
	Posisi titik tujuan
	Posisi acak penghalang
Penentuan penghalang	Posisi tembok
	Posisi pohon

Tabel IV.3. Rencana pengujian program simulasi (lanjutan)

Item uji	Detail pengujian
Penentuan penghalang	Posisi penghapus penghalang
Penentuan heuristik	Nilai heuristic
Pencarian *rute*	Pencarian A*
	Lihat cabang pencarian
	Reset
	Status Bar
Penampilan menu *map*	*Map* baru
	Buka *map*
	Simpan *map*
	Keluar

c. Kasus dan Hasil Pengujian

Berdasarkan rencana pengujian, maka dapat dilakukan pengujian *alpha* dengan *blackbox* pada aplikasi *game pathfinding* untuk mencari jalan terpendek dengan algoritma A* yang dijelaskan pada tabel IV.4.

Tabel IV.4. Kasus dan hasil uji (data normal)

Kasus dan hasil uji (data normal)			
Masukan	Yang diharapkan	Pengamatan	Kesimpulan
Data nilai baris	Dapat terisi *number* untuk masukan data nilai baris	Dapat mengisi *number* yang diharapkan	Diterima
Data nilai kolom	Dapat terisi *number* untuk masukan data nilai kolom	Dapat mengisi *number* yang diharapkan	Diterima

Tabel IV.4. Kasus dan hasil uji (data normal) lanjutan

Kasus dan hasil uji (data normal)			
Masukan	Yang diharapkan	Pengamatan	Kesimpulan
Klik tombol OK	Menampilkan *grid* yang sesuai dengan masukan nilai baris dan nilai kolom	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik tombol titik awal	Menampilkan titik awal pada *map*	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik tombol titik tujuan	Menampilkan titik tujuan pada *map*	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik tombol tembok	Menampilkan penghalang tembok pada *map*	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik tombol pohon	Menampilkan penghalang pohon pada *map*	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik tombol hapus penghalang	Dapat menghapus penghalang tembok atau pohon pada *map*	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik tombol A*	Dapat menampilkan *rute* dari titik awal sampai titik tujuan dengan A*	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik *checkbox* lihat cabang pencarian	Dapat menampilkan jalur pelacakan A*	Tombol dapat berfungsi sesuai yang diharapkan	Diterima

Tabel IV.4. Kasus dan hasil uji (data normal) lanjutan

Kasus dan hasil uji (data normal)			
Masukan	Yang diharapkan	Pengamatan	Kesimpulan
Klik tombol map baru	Untuk menghapus semua penghalang pada *map* sehingga dapat membuat *map* baru	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik tombol reset	Untuk menghapus *rute* yang dihasilkan oleh A*	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik tombol acak	Untuk mengacak posisi titik awal dan titik tujuan secara bersamaan	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Geser nilai heuristik	Dapat merubah nilai heuristik dengan rentang antara 0,1 sampai 10	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik tombol menu file map	Menampilkan submenu file map	Tombol menu dapat berfungsi sesuai yang diharapkan	Diterima
Klik map baru Ctrl+N	Untuk membuat *map* baru	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik buka Ctrl+O	Untuk membuka *map* yang telah disimpan	Tombol dapat berfungsi sesuai yang diharapkan	Diterima

Tabel IV.4. Kasus dan hasil uji (data normal) lanjutan

Kasus dan hasil uji (data normal)			
Masukan	Yang diharapkan	Pengamatan	Kesimpulan
Klik simpan Ctrl+S	Untuk menyimpan *map* baru	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik keluar	Untuk keluar dari aplikasi	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik tombol panduan	Menampilkan *form* panduan aplikasi	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik keluar panduan	Untuk menutup *form* panduan menuju tampilan utama	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik tombol tentang aplikasi	Menampilkan *form* tentang aplikasi	Tombol dapat berfungsi sesuai yang diharapkan	Diterima
Klik keluar tentang aplikasi	Untuk menutup dari *form* tentang aplikasi menuju tampilan utama	Tombol dapat berfungsi sesuai yang diharapkan	Diterima

Tabel IV.4. Kasus dan hasil uji (data tidak normal)

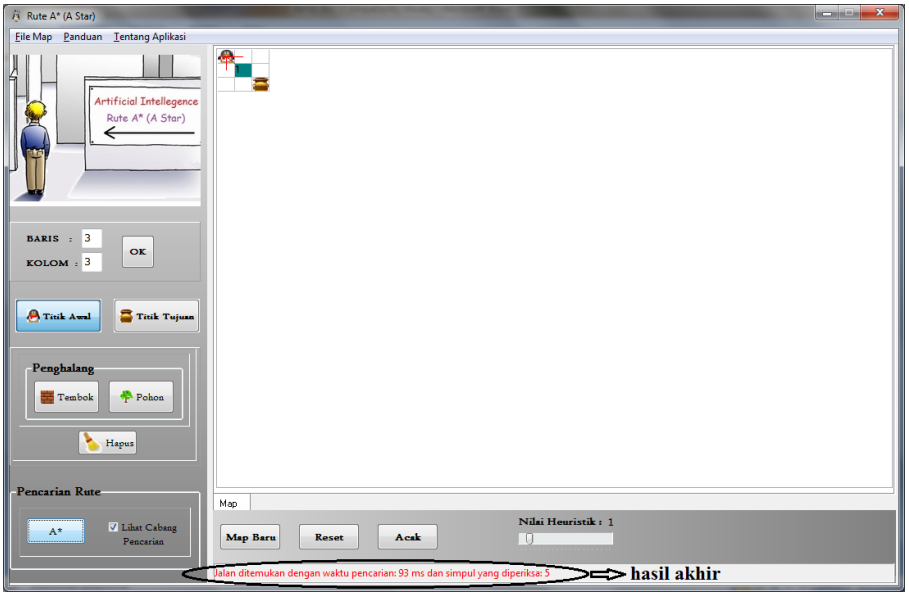
Kasus dan hasil uji (data tidak normal)			
Masukan	Yang diharapkan	Pengamatan	Kesimpulan
Data nilai baris atau nilai kolom kosong	Muncul pesan untuk nilai baris atau nilai kolom tidak boleh kosong	Hasil sesuai dengan yang diharapkan	Diterima

Tabel IV.5. Kasus dan hasil uji (data tidak normal) lanjutan

Kasus dan hasil uji (data tidak normal)			
Masukan	Yang diharapkan	Pengamatan	Kesimpulan
Data nilai baris lebih dari 31	Muncul pesan konfirmasi bahwa nilai baris tidak boleh lebih dari 31	Hasil sesuai dengan yang diharapkan	Diterima
Data nilai kolom lebih dari 39	Muncul pesan konfirmasi bahwa nilai kolom tidak boleh lebih dari 39	Hasil sesuai dengan yang diharapkan	Diterima
Data nilai baris atau nilai kolom kurang dari 3	Muncul pesan konfirmasi bahwa nilai baris atau nilai kolom tidak boleh kurang dari 3	Hasil sesuai dengan yang diharapkan	Diterima
Data nilai baris lebih dari 31 dan nilai kolom lebih dari 39	Muncul pesan konfirmasi bahwa nilai baris tidak boleh lebih dari 31 atau nilai kolom tidak boleh lebih dari 39	Hasil sesuai dengan yang diharapkan	Diterima

d. Pengujian parameter aplikasi

Performansi yang akan diuji pada aplikasi yang telah dibangun yaitu waktu pencarian, simpul yang diperiksa dan langkah yang dihasilkan dengan parameter nilai *heuristic* antara rentang 0,1 sampai 3. Implementasi ordo minimal 3 x 3 tanpa penghalang dengan *heuristic* 1 dapat dilihat pada gambar IV.8.



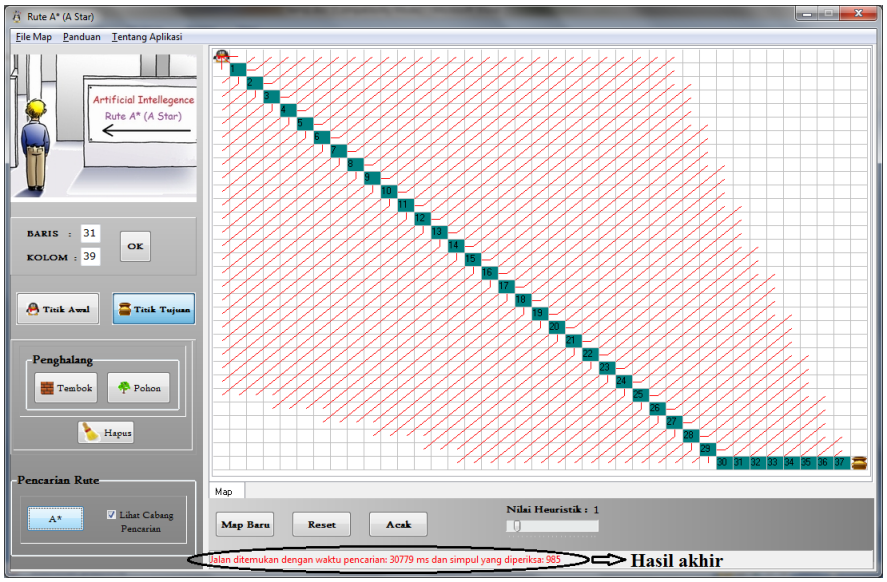
Gambar IV.8. Pengujian ordo 3 x 3 tanpa penghalang

Hasil pengujian performansi pada gambar IV.8 dengan parameter nilai *heuristic* antara rentang 0,1 sampai 3 dengan titik awal pada simpul (0,0) dan titik tujuan pada simpul (2,2) dapat dilihat pada tabel IV.6.

Tabel IV.5. Hasil pengujian ordo 3 x 3 tanpa penghalang

No.	Nilai heuristik	Waktu (ms)	Jumlah simpul yang diperiksa	Jalan yang dihasilkan
1	0,1	94	5	1
2	0,5	94	5	1
3	1	94	5	1
4	1,5	32	3	1
5	2	32	3	1
6	2,5	32	3	1
7	3	32	3	1

Pengujian performansi dengan ordo maksimal 31 x 39 tanpa penghalang dengan *heuristic* 1 dapat dilihat pada gambar IV.9.



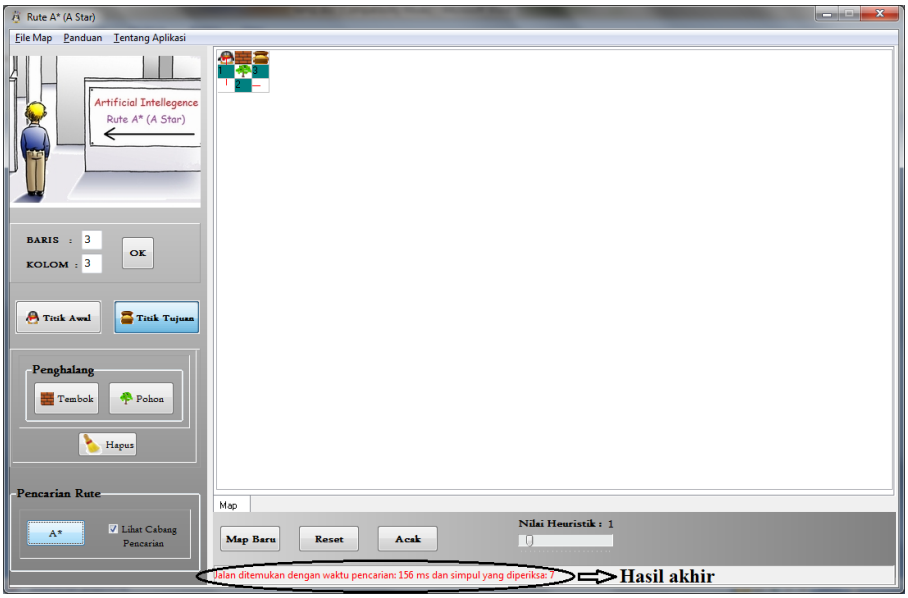
Gambar IV.9. Pengujian ordo 31 x 39 tanpa penghalang

Hasil pengujian performansi pada gambar IV.9 dengan parameter nilai *heuristic* antara rentang 0,1 sampai 3 dengan titik awal pada simpul (0,0) dan titik tujuan pada simpul (31,39) dapat dilihat pada tabel IV.7.

Tabel IV.6. Hasil pengujian ordo 31 x 39 tanpa penghalang

No.	Nilai heuristik	Waktu (ms)	Jumlah simpul yang diperiksa	Jalan yang dihasilkan
1	0,1	36785	1178	37
2	0,5	34788	1114	37
3	1	30764	985	37
4	1,5	24321	778	37
5	2	15413	493	37
6	2,5	3136	99	37
7	3	1264	39	37

Pengujian performansi pada ordo 3 x 3 dengan 2 penghalang disimpul (1,0) dan (1,1) dapat dilihat pada gambar IV.10.



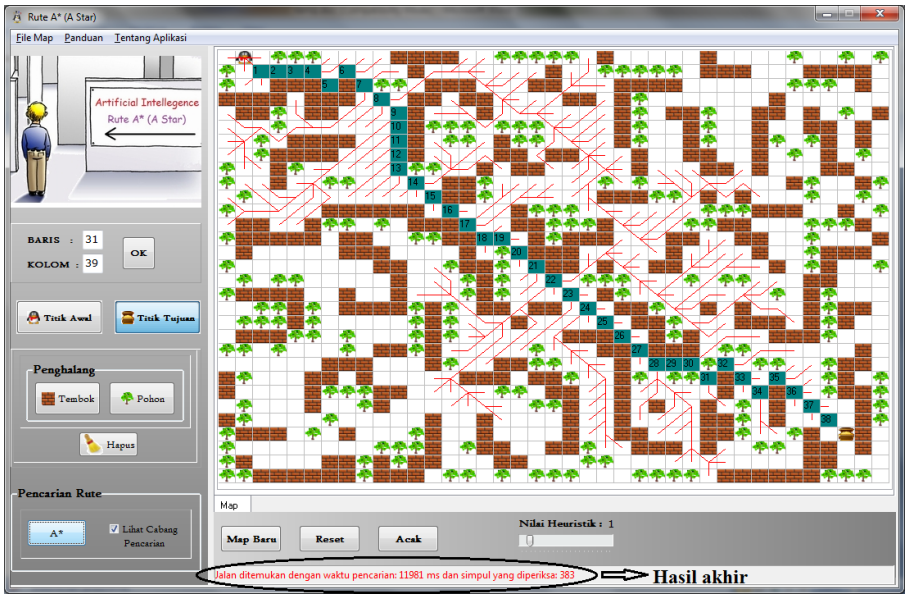
Gambar IV.10. Pengujian ordo 3 x 3 dengan penghalang

Hasil pengujian performansi pada gambar IV.10 dengan parameter nilai *heuristic* antara rentang 0,1 sampai 3 dengan titik awal pada simpul (0,0) dan titik tujuan pada simpul (2,0) dapat dilihat pada tabel IV.8.

Tabel IV.7. Hasil pengujian ordo 3 x 3 dengan penghalang

No.	Nilai heuristik	Waktu (ms)	Jumlah simpul yang diperiksa	Jalan yang dihasilkan
1	0,1	156	7	3
2	0,5	156	7	3
3	1	156	7	3
4	1,5	156	7	3
5	2	125	6	3
6	2,5	93	5	3
7	3	93	5	3

Pengujian performansi pada ordo maksimal 31 x 39 dengan penghalang dan nilai *heuristic* 1 dapat dilihat pada gambar IV.11.



Gambar IV.11. Pengujian ordo 31 x 39 dengan penghalang

Hasil pengujian performansi pada gambar IV.11 dengan parameter nilai *heuristic* antara rentang 0,1 sampai 3 dengan titik awal pada simpul (0,0) dan titik tujuan pada simpul (31,39) dapat dilihat pada tabel IV.9.

Tabel IV.8. Hasil pengujian ordo 31 x 39 dengan penghalang

No.	Nilai heuristik	Waktu (ms)	Jumlah simpul yang diperiksa	Jalan yang dihasilkan
1	0,1	16536	529	38
2	0,5	14789	473	38
3	1	11980	383	38
4	1,5	10078	322	38
5	2	7207	230	38
6	2,5	2824	89	38
7	3	1701	53	38

e. Kesimpulan Pengujian *Alpha*

Hasil pengujian dari pengujian *alpha* yang telah dilakukan, menunjukkan bahwa aplikasi yang dibangun sudah memenuhi persyaratan fungsional, akan tetapi pada prosesnya masih memungkinkan untuk terjadinya kesalahan, namun frekuensi kesalahan masih relatif kecil. Secara fungsional sistem yang telah dibangun sudah dapat menghasilkan keluaran yang diharapkan, begitu juga secara perhitungan yang dilakukan dengan algoritma A* *search* telah dihasilkan jalan (*rute*) dari titik awal ke titik tujuan. Jalan yang dihasilkan merupakan jalan terpendek karena simpul yang diperiksa relatif banyak dan memerlukan waktu pencarian yang relatif lama pula. Parameter dengan nilai *heuristic* paling kecil akan menghasilkan waktu dan simpul yang diperiksa besar sebaliknya nilai *heuristic* paling besar akan menghasilkan waktu dan simpul yang diperiksa kecil.

f. Pengujian Beta

Pengujian beta merupakan pengujian yang dilakukan secara objektif dimana pengujian dilakukan secara langsung kepada responden dengan membuat kuesioner mengenai kepuasan pengguna dengan kandungan *point* syarat.

Berdasarkan hasil pengujian beta, dicari persentase masing-masing jawaban dengan menggunakan rumus:

$$\mathbf{Y=P/Q*100\%}$$

Keterangan:

P = jumlah jawaban responden tiap soal

Q = jumlah responden

Y = nilai persentase

Berikut ini adalah hasil persentase masing-masing jawaban yang sudah dihitung nilainya dengan menggunakan rumus di atas. Kuesioner ini telah diujikan kepada 10 orang. Hasil perhitungan persentase masing-masing jawaban dapat dilihat pada tabel IV.6 – IV.10.

Untuk Pertanyaan No 1

"Apakah penulusuran dengan algoritma A* mudah dipahami"

Tabel IV.9. Hasil pengujian beta pertanyaan No 1

Kategori Jawaban	Frekuensi Jawaban	Jumlah Populasi *Sample*	Jumlah Persentase
Sangat Setuju	4	10	40%
Setuju	6	10	60%
Ragu-ragu	0	10	0%
Tidak Setuju	0	10	0%
Sangat Tidak Setuju	0	10	0%

Untuk Pertanyaan No 2

"Apakah jalur yang dihasilkan merupakan jalan terpendek"

Tabel IV.10. Hasil pengujian beta pertanyaan No 2

Kategori Jawaban	Frekuensi Jawaban	Jumlah Populasi *Sample*	Jumlah Persentase
Sangat Setuju	5	10	50%
Setuju	5	10	50%
Ragu-ragu	0	10	0%
Tidak Setuju	0	10	0%
Sangat Tidak Setuju	0	10	0%

Untuk Pertanyaan No 3

"Apakah aplikasi ini sudah sesuai dengan tujuan yang ingin dicapai"

Tabel IV.11. Hasil pengujian beta pertanyaan No 3

Kategori Jawaban	Frekuensi Jawaban	Jumlah Populasi *Sample*	Jumlah Persentase
Sangat Setuju	5	10	50%
Setuju	5	10	50%
Ragu-ragu	0	10	0%
Tidak Setuju	0	10	0%
Sangat Tidak Setuju	0	10	0%

Untuk Pertanyaan No 4

"Apakah diperlukan fungsi manual untuk pengguna mencari jalan sendiri"

Tabel IV.12. Hasil pengujian beta pertanyaan No 4

Kategori Jawaban	Frekuensi Jawaban	Jumlah Populasi *Sample*	Jumlah Persentase
Sangat Setuju	2	10	20%
Setuju	5	10	50%
Ragu-ragu	0	10	0%
Tidak Setuju	2	10	20%
Sangat Tidak Setuju	1	10	10%

Untuk Pertanyaan No 5

"Apakah aplikasi *game pathfinding* mudah digunakan"

Tabel IV.13. Hasil pengujian beta pertanyaan No 5

Kategori Jawaban	Frekuensi Jawaban	Jumlah Populasi *Sample*	Jumlah Persentase
Sangat Setuju	3	10	30%
Setuju	6	10	60%
Ragu-ragu	1	10	10%
Tidak Setuju	0	10	0%
Sangat Tidak Setuju	0	10	0%

Saran dari responden dapat disimpulkan sebagai berikut :

1 Tambahkan *loading* ketika pertama kali *game* dijalankan.

2 Mengubah *map* dengan ukuran lebih besar agar gambar titik awal dan titik tujuan lebih jelas.

3 Warna *track* lebih mencolok dari penghalang dan penambahan animasi gerak.

4 Tambahkan algoritma untuk pencarian jalan.

f. Kesimpulan Pengujian Beta

Berdasarkan dari jawaban yang diberikan oleh responden, bisa ditarik kesimpulan bahwa "Penerapan Algoritma A* (*A Star*) dalam Pencarian Jalan Terpendek pada *Game Pathfinding*" ini dinilai dari penelusuran algoritma A* dinyatakan mudah dipahami, jalur yang dihasilkan merupakan jalan terpendek, aplikasi sudah sesuai dengan tujuan, diperlukan pencarian secara manual bagi pengguna dan pengoperasian aplikasi mudah digunakan.