BAB 4
IMPLEMENTASI DAN PENGUJIAN

4.1 Implementasi

Implementasi merupakan tahap penciptaan perangkat lunak dan juga tahap kelanjutan dari kegiatan perancangan aplikasi, tahap ini merupakan tahap dimana aplikasi siap untuk dioperasikan. Aplikasi ini diimplementasikan dengan nama Pertempuran Palagan Ambarawa dengan *file extension*-nya *.exe.*

4.1.1 Implementasi Perangkat Keras

Perangkat keras yang dapat digunakan untuk menjalankan aplikasi *game* ini harus memiliki spesifikasi yang cukup tinggi. Sebagai ilustrasi sistem minimum agar dapat menjalankan *game* Palagan Ambarawa adalah sebagai berikut :

1. Processor Dual Core 1,80 Ghz
2. RAM 1 GB
3. VGA 512 MB
4. Harddisk 20 GB
5. Mouse dan Keyboard

4.1.2 Implementasi Perangkat Lunak

Adapun spesifikasi perangkat lunak yang dibutuhkan untuk menjalankan *game* Pertempuran Palagan Ambarawa adalah :

1. Sistem operasi minimal Windows Xp
2. Graphic card
3. DirectX 9 keatas

4.1.3 Implementasi Instalasi Aplikasi *Game*

Untuk dapat memainkan *game* Palagan Ambarawa, pemain terlebih harus mempunyai *file Palagan Ambarawa.exe* . Untuk mulai memainkan *game*, pemain

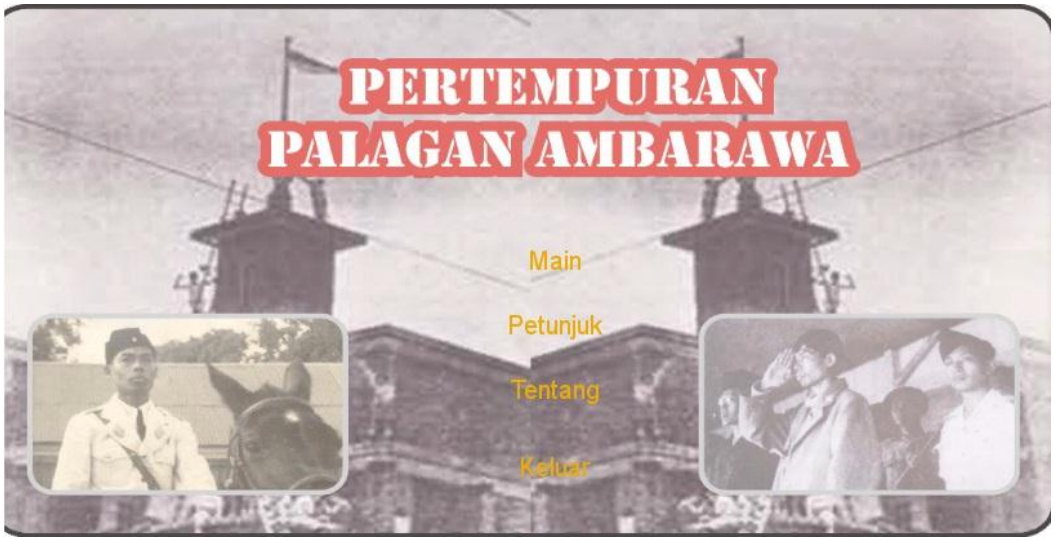
hanya perlu menjalankan *file Palagan Ambarawa.exe* yang telah ada dalam perangkat keras.

4.1.4 Implementasi Antarmuka

Antarmuka dapat membuat pengguna merasa nyaman dan bertahan lama berada dalam aplikasi atau sebaliknya. Berikut ini merupakan antarmuka yang ada pada aplikasi *game* Pertempuran Palagan Ambarawa.

4.1.4.1 Antarmuka Menu Utama

Saat pengguna membuka aplikasi pertama kali, maka antarmuka yang ditampilkan oleh sistem adalah antarmuka menu utama. Berikut ini merupakan tampilan dari antarmuka menu utama :



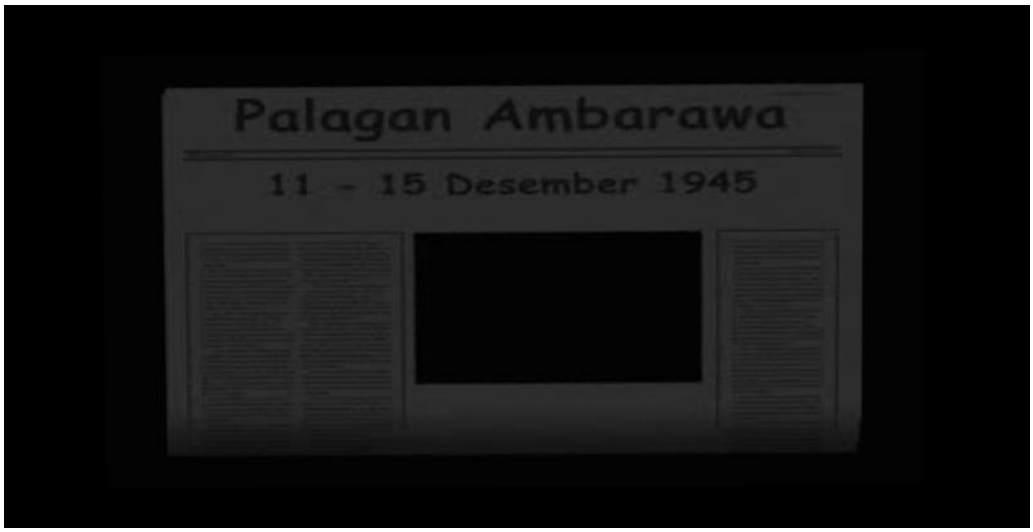
Gambar 4.1 Antarmuka Menu Utama

Gambar 4.1 di atas merupakan menu utama pada *game* Pertempuran Palagan Ambarawa. Terdapat empat sub menu di dalamnya, yaitu menu Main yang berfungsi untuk memulai permainan, menu Petunjuk yang berfungsi untuk melihat informasi

kontrol yang digunakan ketika memainkan *game*, menu Tentang untuk melihat siapa saja pembuat game dan menu Keluar yang berfungsi untuk keluar dari aplikasi.

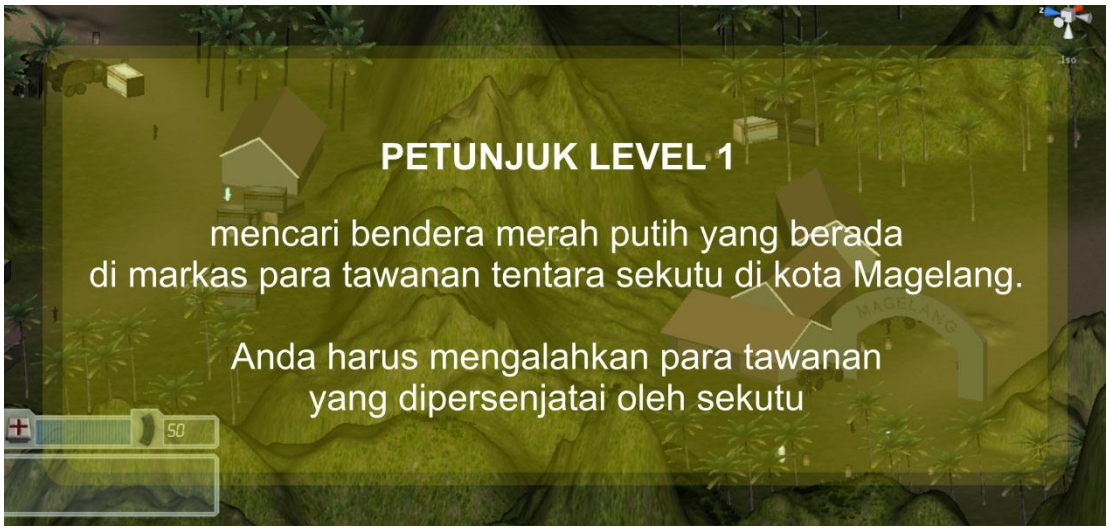
4.1.4.2 Antarmuka Menu Main

Antarmuka yang ditampilkan oleh sistem adalah; Pertama, antarmuka *scene video* berupa video cerita awal dari *game* pada level pertama. Selanjutnya akan masuk ke antarmuka petunjuk misi dan kemudian akan masuk pada antarmuka permainan. Jika seluruh misi telah terselesaikan maka akan masuk ke antarmuka *ending scene* berupa video akhir cerita *game*. Berikut ini merupakan tampilan dari antarmuka menu mainkan.



Gambar 4.2 Antarmuka *scene video* level 1

Gambar 4.2 merupakan antarmuka *scene video* pada level pertama yang terdapat pada *game* Pertempuran Palagan Ambarawa.



Gambar 4.3 Antarmuka Petunjuk misi

Gambar 4.3 merupakan antarmuka petunjuk misi yang terdapat pada *game* Pertempuran Palagan Ambarawa.



Gambar 4.4 Antarmuka Permainan di level 1-1

Gambar 4.4 merupakan antarmuka permainan yang terdapat pada *game* Pertempuran Palagan Ambarawa. Gambar tersebut memperlihatkan posisi pemain ketika masuk ke dalam permainan yang berada di kota Magelang, Jawa Tengah.



Gambar 4.5 Antarmuka Permainan di level 1-2

Gambar 4.5 menunjukan posisi pemain ketika akan menghadapi senapan otomatis.



Gambar 4.6 Antarmuka Permainan di level 1-3

Gambar 4.6 menunjukan posisi pemain ketika telah menemukan bendera merah putih di markas tawanan sekutu.

4.1.4.3 Antarmuka Menu Petunjuk

Antarmuka Menu Petunjuk menunjukkan menampilkan pada saat pengguna memilih menu petunjuk, maka antarmuka yang ditampilkan oleh sistem adalah antarmuka informasi kontrol yang digunakan pada *game* Pertempuran Palagan Ambarawa.



Gambar 4.7 merupakan tampilan dari antarmuka menu Petunjuk :

4.2 Pengujian

Pada umunya ada 2 (dua) macam teknik pengujian yaitu *white box testing* dan *black box testing*. Pengujian yang menggunakan teknik *white box testing* lebih menitikberatkan pada struktur internal (*source code*) program, dimana setiap baris kode yang akan di*compile* atau dicek satu persatu untuk mengetahui apakah masih terjadi *error*. Pengujian pada *black box testing* tidak terlalu memperhatikan struktur internal (*source code*) dari program, tetapi lebih kepada menemukan kesalahan program. Pada *game* Pertempuran Palagan Ambarawa. digunakan pengujian *black box testing* dan pengujian *black box testing*. Cara pengujian dilakukan melalui pengujian *beta*.

4.2.1 Kasus dan Hasil Pengujian (*Black Box Testing*)

Berikut ini merupakan kasus untuk menguji perangkat lunak yang sudah dibangun dengan menggunakan metode *black box testing* berdasarkan skenario pengujian aplikasi yang terdapat pada tabel 4.1 yang sudah dijabarkan. Hasil pengujian yang akan dilakukan pada aplikasi ini selengkapnya dapat dilihat pada tabel 4.1

Tabel 4.1 Pengujian Black Box

No	Kasus yang akan di uji	Skenario dan Hasil Uji		
		Skenario Uji	**Hasil yang Diharapkan**	**Kesimpulan**
1.	Menu Utama	Klik Menu Mainkan	Sistem menampilkan halaman *Scene Video*	[√] Berhasil [ ] Tidak Berhasil
		Klik Menu Petunjuk	Sistem menampilkan halaman menu Petunjuk	[√] Berhasil [ ] Tidak Berhasil
		Klik Menu Keluar	Menutup atau Keluar dari aplikasi	[√] Berhasil [ ] Tidak Berhasil
2.	Menu Permainan	Klik Menu Mainkan	Sistem menampilkan halaman *Scene Video*	[√] Berhasil [ ] Tidak Berhasil
3.	Menu Petunjuk	Klik Menu Petunjuk	Sistem menampilkan petunjuk berupa informasi kontrol yang digunakan dalam *game*	[√] Berhasil [ ] Tidak Berhasil
		Klik Kembali	Sistem	[√] Berhasil

No	Kasus yang akan di uji	Skenario dan Hasil Uji		
		Skenario Uji	**Hasil yang Diharapkan**	**Kesimpulan**
			menampilkan menu utama	[ ] Tidak Berhasil
4.	Menu Keluar	Klik Menu Keluar	Menutup atau Keluar dari aplikasi	[√] Berhasil [ ] Tidak Berhasil
5.	Permainan	Tekan tombol W *keyboard*	Karakter pemain bergerak kedepan	[√] Berhasil [ ] Tidak Berhasil
		Tekan tombol A *keyboard*	Karakter pemain bergerak kekiri	[√] Berhasil [ ] Tidak Berhasil
		Tekan tombol S *keyboard*	Karakter pemain bergerak kebelakang	[√] Berhasil [ ] Tidak Berhasil
		Tekan tombol D *keyboard*	Karakter pemain bergerak kekanan	[√] Berhasil [ ] Tidak Berhasil
		Tekan tombol W *keyboard*	Karakter pemain bergerak kedepan	[√] Berhasil [ ] Tidak Berhasil
		Klik kiri *mouse*	Karakter pemain menembak	[√] Berhasil [ ] Tidak Berhasil
		Tekan tombol P *keyboard*	Menampilkan *Pause Screen*	[√] Berhasil [ ] Tidak Berhasil
		Tekan tombol 1 *keyboard*	Karakter memakai senjata Laras Panjang	[√] Berhasil [ ] Tidak Berhasil
		Tekan tombol 2 *keyboard*	Karakter memakai senjata Roket	[√] Berhasil [ ] Tidak Berhasil
6.	Mengambil darah	Mengarahkan karakter hingga menyentuh icon darah menggunakan *mouse* dan	Bar darah pada permainan akan bertambah jika darah pemain memang tidak penuh dan *icon*	[√] Berhasil [ ] Tidak Berhasil

No	Kasus yang akan di uji	Skenario dan Hasil Uji		
		Skenario Uji	**Hasil yang Diharapkan**	**Kesimpulan**
		keyboard	darah menghilang	
7.	Menembak	Mengarahkan *pointer* / *cursor* senjata menggunakan *keyboard* dan *mouse* lalu menekan tombol kiri *mouse* untuk mengeluarkan peluru / *mouse*	Senjata yang dipegang karakter pada *game* akan mengeluarkan peluru dan suara tembakan secara bersama-sama	[√] Berhasil [ ] Tidak Berhasil
8.	Mengganti Senjata	Menekan tombol 1 pada *keyboard* untuk menggunakan senjata laras panjang dan menekan tombol 2 pada *keyboard* untuk menggunakan senjata roket	Senjata laras panjang akan digunakan apablia pemain menekan tombol 1 dan Senjata roket akan digunakan apablia pemain menekan tombol 2	[√] Berhasil [ ] Tidak Berhasil
9.	Memberhenti-kan permainan	Menekan tombol P pada *keyboard*	*Game* akan mengeluarkan *screen pause*	[√] Berhasil [ ] Tidak Berhasil
10.	Mengambil petunjuk	Mengarahkan karakter hingga menyentuh icon panah menggunakan *mouse* dan *keyboard*	*Game* akan menampilkan *pop up* berupa teks dan suara	[√] Berhasil [ ] Tidak Berhasil

4.2.2 Kasus dan Hasil Pengujian (*White Box Testing)*

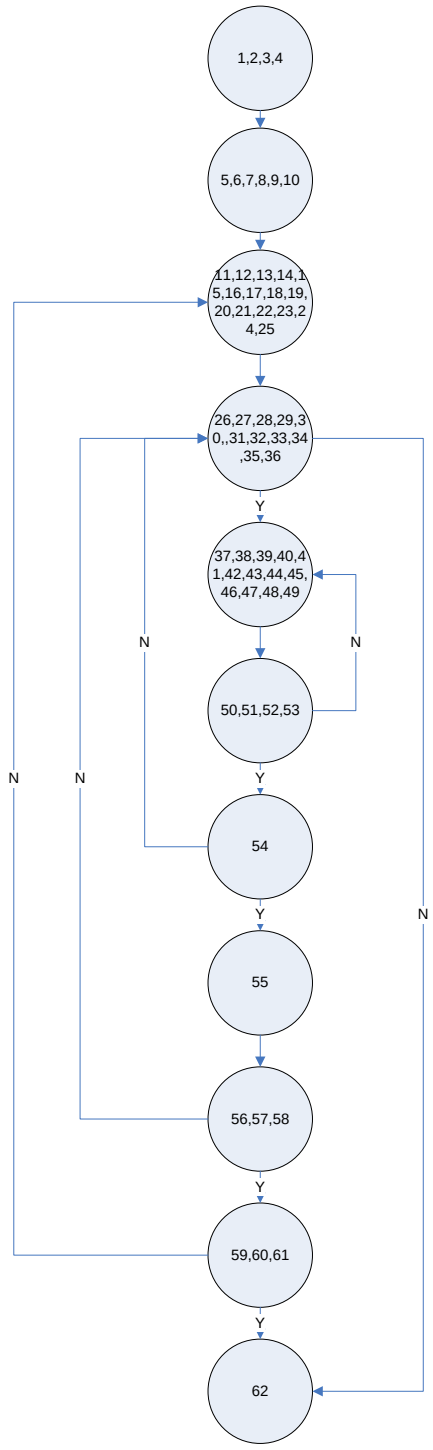
Berikut ini merupakan algoritma A* untuk *level* 3 yang penerapannya menggunakan bahasa pemrograman C#:

```
1.  public System.Type[] graphTypes{
2.  get {
3.  return astarData.graphTypes;}
4.  }
5.  public Matrix translate (float _X, float _Y, float _Z){
6.  m[12] = _X;
7.  m[13] = _Y;
8.  m[14] = _Z;
9.  return this;
10. }
11. public virtual int[] InitialOpen (BinaryHeapM open, Int3
    targetPosition, Int3 position, Path path, bool doOpen) {
12. return BaseInitialOpen
13. (open,targetPosition,position,path,doOpen);
14. }
15. public void Close () {
16. if (readerStream != null) {
17. readerStream.Close ();
18. }
19. if (writerStream != null) {
20. if (anchors != null) {
21. SerializeAnchors ();
22. }
23. writerStream.Close ();
24. }
25. }
26. if (doOpen) {
27. connectionCosts = costs;
28. }
29. if (connectionCosts == null) {
30. return null;
31. }
32. int[] costs = connectionCosts;
33. connectionCosts = new int[connectionCosts.Length];
34. for (int i=0;i<connectionCosts.Length;i++) {
35. connectionCosts[i] = (connections[i].position-
    position).costMagnitude;
36. }
37. if (!doOpen) {
38. for (int i=0;i<connectionCosts.Length;i++) {
39. Node other = connections[i];
```

```
40.  if (other.connections != null) {
41.  for (int q = 0;q < other.connections.Length;q++) {
42.  if (other.connections[q] == this) {
43.  other.connectionCosts[q] = connectionCosts[i];
44.  break;
45.  }
46.  }
47.  }
48.  }
49.  }
50.  if (connections[i].walkable && connections[i].area !=
     area) {
51.  stack.Push (connections[i]);
52.  connections[i].area = area;
53.  }
54.  if (doOpen) {
55.  nodeR.g = nodeR.parent.g+nodeR.cost+penalty +
     nodeRunData.path.GetTagPenalty(tags);
56.  public virtual void UpdateAllG (NodeRun nodeR, NodeRunData
     nodeRunData) {
57.  BaseUpdateAllG (nodeR, nodeRunData);
58.  }
59.  if (Open == null){
60.  return null;
61.  }
62.  return path;
```

a. *Flow Graph* Algoritma A*

Berdasarkan *pseudocode* di atas maka *flow graph* algoritma A* adalah sebagai berikut:



Gambar 4. 8 *Flow Graph* Algoritma A*

Keterangan:



= Menggambarkan kondisi

= Menggambarkan aksi

b. *Cyclomatic Complexity* V(G)

E = 15, N = 11

V(G) = E – N + 2

V(G) = 15 – 11 + 2

V(G) = 4 + 2 = 6

Keterangan

E = Jumlah aksi

N = Jumlah Kondisi

c. *Independent Path*

Path 1 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62

Path 2 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-34-35-36-62

Path 3 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62

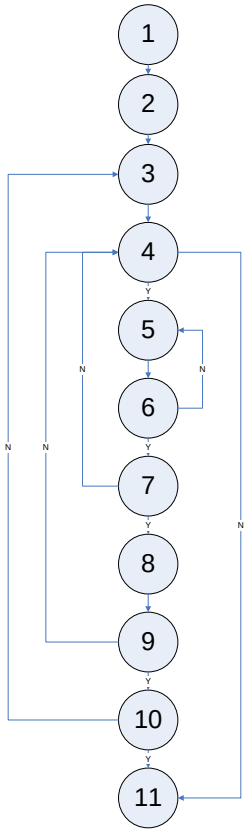
Path 4 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62

Path 5 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-58-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62

Path 6 = 1-2-3-4-5-6-7-8-9-10-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-11-12-13-14-15-16-17-18-19-20-21-22-23-24-25-26-27-28-29-30-31-32-33-34-35-36-37-38-39-40-41-42-43-44-45-46-47-48-49-50-51-52-53-54-55-56-57-58-59-60-61-62

d. *Graph Matriks* Algoritma A*

Penomoran ulang *flowgraph* atau penyederhanaan berdasarkan kondisi atau simpul sebagai berikut:



Gambar 4.9 Penyederhanaan *Flow Graph* Algoritma A*

Tabel 4. 2 *Graph Matriks* Algoritma A*

N	1	2	3	4	5	6	7	8	9	10	11	N(E) -1
1		1										1-1=0
2			1									1-1=0
3				1								1-1=0
4					1						1	2-1=1
5						1						1-1=0
6					1		1					2-1=1
7				1				1				2-1=1

8									1			1-1=0
9				1						1		2-1=1
10			1								1	2-1=1
11												1-1=0
										Sum(E)+1		5+1=6

Keterangan:

1. Baris dan kolom merepresentasikan simpul
1. Nilai 1 merepresentasikan adanya keterhubungan antar simpul

e. Hasil Uji Algoritma A*

Tabel 4. 3 Hasil Uji Algoritma A*

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
1	- readerStream != null - writerStream != null - anchors != null - doOpen - connectionCosts == null - int i=0; i<connectionCosts.Length; i++ - !doOpen - int i=0;i<connectionCosts.Length;i++ - other.connections != null - int q = 0;q < other.connections.Length;q++ - other.connections[q] == this - connections[i].walkable && connections[i].area != area - doOpen	Rute = null, rute tidak ada	Rute tidak ditemukan	[√] Alur terlewati [ ] Alur tidak terlewati
2	- readerStream != null - writerStream != null - anchors != null - doOpen - connectionCosts ==	Current Node = Goal. Rute ditemukan	Rute ditemukan	[√] Alur terlewati [ ] Alur tidak terlewati

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
	null - int i=0; i<connectionCosts.Length; i++			
3	- readerStream != null - writerStream != null - anchors != null - doOpen - connectionCosts == null - int i=0; i<connectionCosts.Length; i++ - !doOpen - int i=0;i<connectionCosts.Length;i++ - other.connections != null - int q = 0;q < other.connections.Length;q++ - other.connections[q] == this - connections[i].walkable && connections[i].area != area - !doOpen - int i=0;i<connectionCosts.Length;i++ - other.connections != null - int q = 0;q < other.connections.Length;q++	Node tidak dapat dilalui, pengujian collision atau unwalkable node	Node menjadi unwalkable	[√] Alur terlewati [ ] Alur tidak terlewati
4	- readerStream != null - writerStream != null - anchors != null - doOpen - connectionCosts == null - int i=0; i<connectionCosts.Length; i++ - !doOpen	Masukkan semua *neighbor node* ke dalam open dan hitung nilai F,G,H. Set *current node*	*Current node* dan nilai F,G,H masuk ke dalam open	[√] Alur terlewati [ ] Alur tidak terlewati

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
	- int i=0;i<connectionCosts.Length;i++ - other.connections != null - int q = 0;q < other.connections.Length;q++ - other.connections[q] == this - connections[i].walkable && connections[i].area != area - doOpen - connectionCosts == null - int i=0; i<connectionCosts.Length; i++			
5	- readerStream != null - writerStream != null - anchors != null - doOpen - connectionCosts == null - int i=0; i<connectionCosts.Length; i++ - !doOpen - int i=0;i<connectionCosts.Length;i++ - other.connections != null - int q = 0;q < other.connections.Length;q++ - other.connections[q] == this - connections[i].walkable && connections[i].area != area - doOpen - connectionCosts == null - int i=0; i<connectionCosts.Leng	Nilai G yang didapat lebih kecil dari nilai G sebelumnya. Kalkulasi ulang nilai F,G,H	Nilai F,G,H, dikalkulasi ulang	[√] Alur terlewati [ ] Alur tidak terlewati

No	Kasus uji	Yang diharapkan	Pengamatan	Keterangan
	th; i++ - !doOpen - int i=0;i<connectionCosts.Length;i++ - other.connections != null - int q = 0;q < other.connections.Length;q++ - other.connections[q] == this - connections[i].walkable && connections[i].area != area			
6	- readerStream != null - writerStream != null - anchors != null - doOpen - connectionCosts == null - int i=0; i<connectionCosts.Length; i++ - !doOpen - int i=0;i<connectionCosts.Length;i++ - other.connections != null - int q = 0;q < other.connections.Length;q++ - other.connections[q] == this - connections[i].walkable && connections[i].area != area - doOpen - Open == null	Open = null, tidak ada *node* di dalam open	Tidak ada *node* di dalam open	[√] Alur terlewati [ ] Alur tidak terlewati

f. Kesimpulan

Dari hasil pengujian algoritma A* yang dilakukan, maka didapat kesimpulan bahwa seluruh *node* yang ada pada setiap *independent path* telah dikerjakan, serta tidak terjadi pengulangan tak terhingga.

4.2.3 Pengujian *Beta*

Pengujian *beta* merupakan pengujian yang dilakukan secara objektif, dimana dilakukan pengujian secara langsung terhadap pengguna dengan menggunakan kuesioner mengenai kepuasa pengguna atas aplikasi yang telah dibangun. Adapun metode penilaian pengujian yang digunakan adalah metode kuantitatif berdasarkan data dari pengguna.

4.2.4 Kuesioner

Kuesioner disebarkan menggunakan teknik *Simple Random Sampling* dimana anggota *sample* dari populasi dilakukan secara acak tanpa memperhatikan strata yang ada dalam populasi. Kuesioner disebarkan kepada 30 orang siswa-siswi di salah satu SMA swasta di kota Cimahi dari umur 16 tahun sampai 18 tahun, dari hasil kuesioner tersebut akan dilakukan perhitungan agar dapat diambil kesimpulan terhadap penilaian penerapan sistem yang telah dibangun. Kuesioner terdiri dari 8 pernyataan dengan menggunakan Sistem penskoran dengan skala pengukuran yang digunakan adalah skala likert, Berikut skor untuk jawaban kuesioner

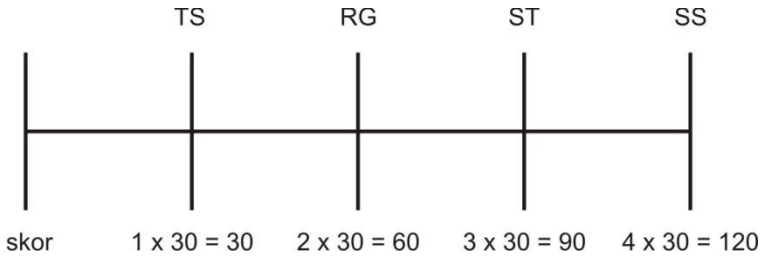
Tabel 4.4 Skor Jawaban Pada Kuesioner

Skala Jawaban	Keterangan	Skor
SS	Sangat Setuju	4
ST	Setuju	3
RG	Ragu-ragu	2
TS	Tidak Setuju	1

Data yang diperoleh dari pemberian kuesioner kepada responden dapat dianalisis dengan menghitung rata-rata jawaban berdasarkan scoring setiap jawaban dari responden. Analisis yang dapat dilakukan yaitu sebagai berikut.

Jumlah skor ideal (kriterium) untuk seluruh item = 4 × jumlah responden
Rata-rata skor dalam persentase = (jumlah skor : jumlah skor ideal) × 100%
Jumlah skor ideal pada penelitian ini yaitu; Jumlah skor ideal = 4×30 = 120

Secara kontinum dapat digambarkan seperti berikut:



Berikut ini adalah daftar pertanyaan yang akan dibagikan pada responden daftar pertanyaan :

1) Apakah anda setuju materi atau informasi tentang sejarah Palagan Ambarawa yang terdapat pada *game* ini mudah untuk diingat?
2) Apakah anda merasa jenuh ketika memainkan *game* ini?
3) Apakah anda setuju game ini merupakan salah satu media pembelajaran yang interaktif?
4) Apakah anda setuju *game* ini merupakan salah satu media pembelajaran yang menarik selain media pembelajaran dari buku atau pembelajaran konvensional di kelas?
5) Apakah anda setuju game sejarah Palagan Ambarawa mudah untuk dimainkan?
6) Apakah anda setuju game ini tidak interaktif?

7) Apakah anda setuju grafis 3D yang terdapat pada *game* merupakan faktor kenyamanan dalam memainkan *game* ini?
8) Apakah Apakah anda terpacu untuk memainkan game ini kembali?

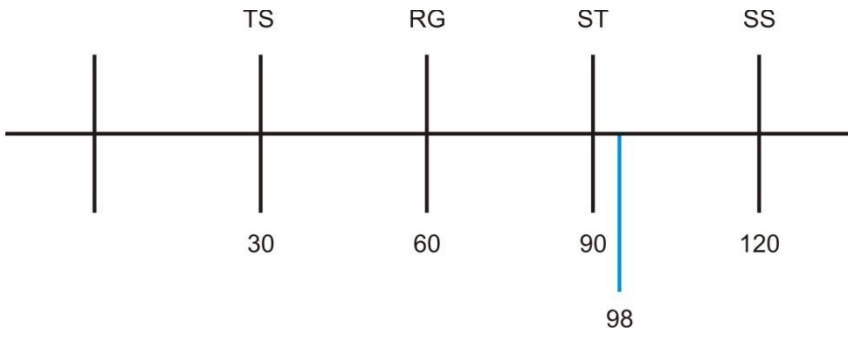
Berikut ini adalah hasil skor masing-masing jawaban yang sudah dihitung nilainya dengan menggunakan rumus diatas.

1. Apakah setuju materi atau informasi tentang sejarah Palagan Ambarawa yang terdapat pada *game* ini mudah untuk diingat?

Tabel 4.5 Perhitungan persentase pertanyaan ke-1

Pertanyaan	Keterangan	Skor	Frekuensi	S
1.	Sangat setuju	4	10	40
	Setuju	3	18	54
	Ragu-ragu	2	2	4
	Tidak setuju	1	0	0
Jumlah			30	98

Garis kontinum



Berdasarkan hasil skor nilai di atas, maka dapat disimpulkan bahwa penilaian terhadap materi atau informasi tentang sejarah Palagan Ambarawa yang terdapat pada *game* ini mudah untuk diingat adalah 98 dari yang diharapkan 120, atau dapat dikategorikan sebagai setuju dan dapat

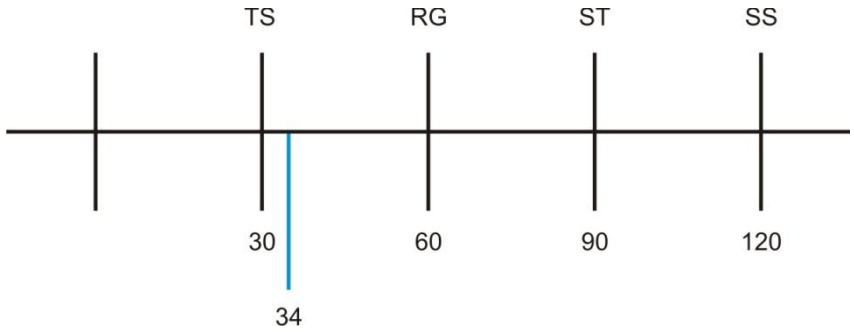
disimpulkan bahwa materi atau informasi tentang sejarah Palagan Ambarawa yang terdapat pada *game* ini mudah untuk diingat.

2. Apakah anda merasa jenuh ketika memainkan *game* ini?

Tabel 4.6 Tabel perhitungan persentase pertanyaan ke-2

Pertanyaan	Keterangan	Skor	Frekuensi	S
2.	Sangat setuju	4	0	0
	Setuju	3	0	0
	Ragu-ragu	2	4	8
	Tidak setuju	1	26	26
Jumlah			30	34

Garis kontinum



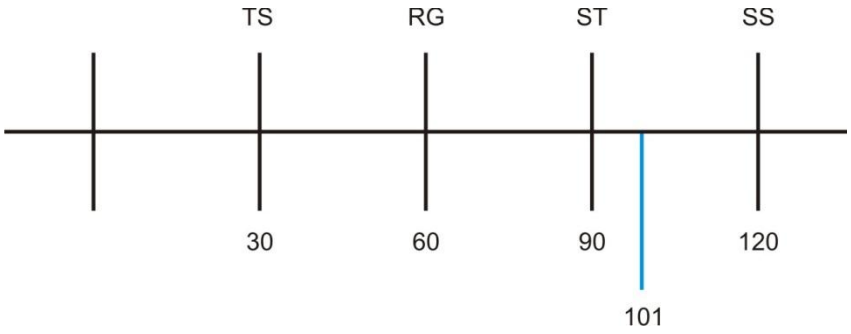
Berdasarkan hasil skor nilai di atas, maka dapat disimpulkan bahwa penilaian terhadap jenuh tidaknya ketika memainkan *game* sejarah Palagan Ambarawa adalah 34 dari yang diharapkan 120, atau dapat dikategorikan sebagai tidak setuju dan dapat disimpulkan bahwa *game* Palagan Ambarawa ini tidak jenuh ketika dimainkan.

3. Apakah anda setuju game ini merupakan salah satu media pembelajaran yang interaktif?

Tabel 4.7 Tabel perhitungan persentase pertanyaan ke-3

Pertanyaan	Keterangan	Skor	Frekuensi	S
3.	Sangat setuju	4	14	56
	Setuju	3	13	39
	Ragu-ragu	2	3	6
	Tidak setuju	1	0	0
Jumlah			30	101

Garis kontinum



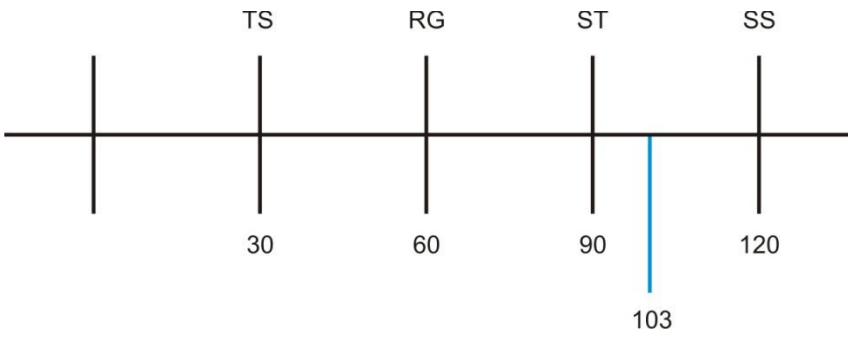
Berdasarkan hasil skor nilai di atas, maka dapat disimpulkan bahwa penilaian game ini merupakan salah satu media pembelajaran yang interaktif adalah 101 dari yang diharapkan 120, atau dapat dikategorikan sebagai setuju dan dapat disimpulkan bahwa *game* Palagan Ambarawa ini merupakan salah satu media pembelajaran yang interaktif.

4. Apakah anda setuju *game* ini merupakan salah satu media pembelajaran yang menarik selain media pembelajaran dari buku atau pembelajaran konvensional di kelas?

Tabel 4.10 Tabel perhitungan persentase pertanyaan ke-4

Pertanyaan	Keterangan	Skor	Frekuensi	S
4.	Sangat setuju	4	15	60
	Setuju	3	13	39
	Ragu-ragu	2	2	4
	Tidak setuju	1	0	0
Jumlah			30	103

Garis kontinum



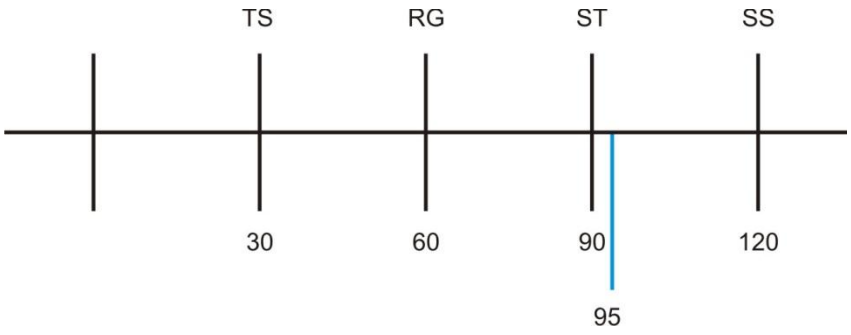
Berdasarkan hasil skor nilai di atas, maka dapat disimpulkan bahwa penilaian game ini merupakan salah satu pembelajaran yang menarik adalah 103 dari yang diharapkan 120, atau dapat dikategorikan sebagai setuju dan dapat disimpulkan bahwa *game* ini merupakan salah satu media pembelajaran yang menarik selain media pembelajaran dari buku atau pembelajaran konvensional di kelas.

5. Apakah anda setuju game sejarah Palagan Ambarawa mudah untuk dimainkan?

Tabel 4.11 Perhitungan persentase pertanyaan ke-5

Pertanyaan	**Keterangan**	**Skor**	**Frekuensi**	**S**
5.	Sangat setuju	4	12	48
	Setuju	3	11	33
	Ragu-ragu	2	7	14
	Tidak setuju	1	0	0
Jumlah			30	95

Garis kontinum



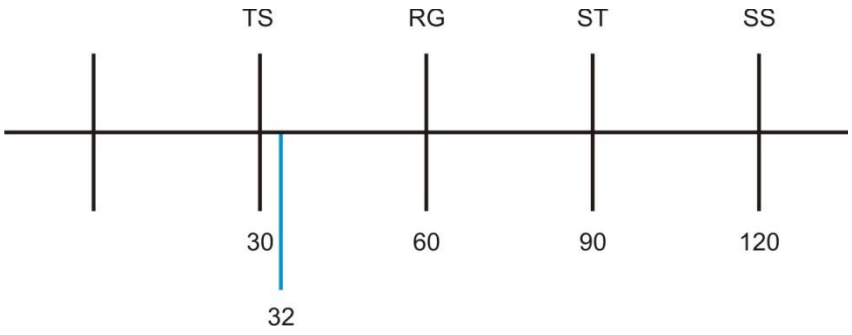
Berdasarkan hasil skor nilai di atas, maka dapat disimpulkan bahwa penilaian game ini mudah untuk dimainkan adalah 95 dari yang diharapkan 120, atau dapat dikategorikan sebagai setuju dan dapat disimpulkan bahwa *game* sejarah Palagan Ambarawa mudah untuk dimainkan.

6. Apakah anda setuju game ini tidak interaktif?

Tabel 4.12 Perhitungan persentase pertanyaan ke-6

Pertanyaan	Keterangan	Skor	Frekuensi	S
6.	Sangat setuju	4	0	0
	Setuju	3	0	0
	Ragu-ragu	2	2	4
	Tidak setuju	1	28	28
Jumlah			30	32

Garis kontinum



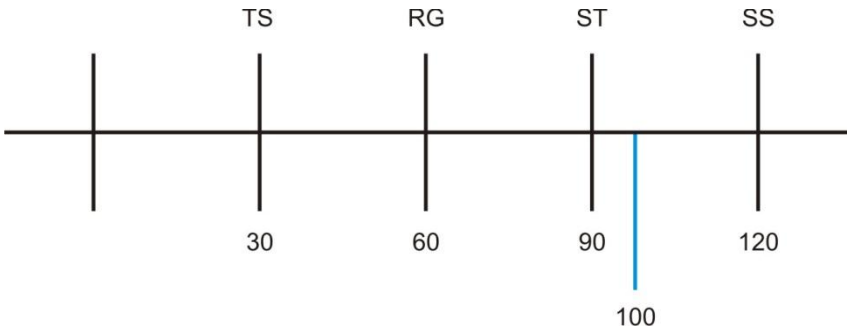
Berdasarkan hasil skor nilai di atas, maka dapat disimpulkan bahwa penilaian game ini tidak interaktif adalah 32 dari yang diharapkan 120, atau dapat dikategorikan sebagai tidak setuju dan dapat disimpulkan bahwa *game* Palagan Ambarawa merupakan media pembelajaran yang interaktif.

7. Apakah anda setuju grafis 3D yang terdapat pada *game* merupakan faktor kenyamanan dalam memainkan *game* ini?

Tabel 4.13 Perhitungan persentase pertanyaan ke-7

Pertanyaan	Keterangan	Skor	Frekuensi	S
7.	Sangat setuju	4	12	48
	Setuju	3	16	48
	Ragu-ragu	2	2	4
	Tidak setuju	1	0	0
Jumlah			30	100

Garis kontinum



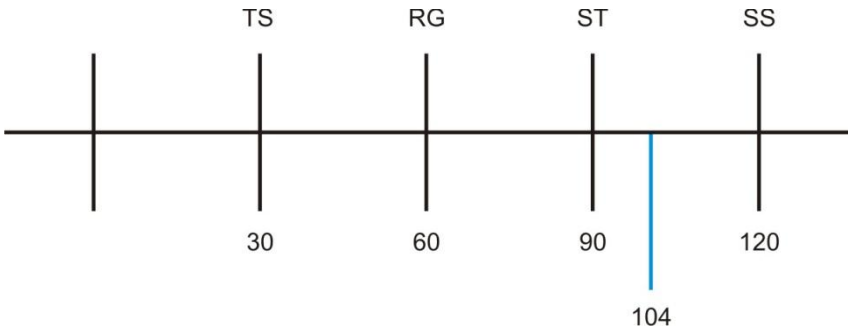
Berdasarkan hasil skor nilai di atas, maka dapat disimpulkan bahwa penilaian game ini tehadap grafis 3D dapat membuat kenyamanan pada saat memainkan *game* ini adalah 100 dari yang diharapkan 120, atau dapat dikategorikan sebagai setuju dan dapat disimpulkan bahwa grafis 3D yang terdapat pada *game* merupakan faktor kenyamanan dalam memainkan *game* Palagan Ambarawa.

8. Apakah anda tertantang untuk memainkan game ini kembali?

Tabel 4.7 Perhitungan persentase pertanyaan ke-8

Pertanyaan	Keterangan	Skor	Frekuensi	S
8.	Sangat setuju	4	17	68
	Setuju	3	10	30
	Ragu-ragu	2	3	6
	Tidak setuju	1	0	0
Jumlah			30	104

Garis kontinum



Berdasarkan hasil skor nilai di atas, maka dapat disimpulkan bahwa penilaian terhadap tertantang atau tidaknya untuk memainkan *game* ini lagi adalah 104 dari yang diharapkan 120, atau dapat dikategorikan sebagai setuju dan dapat disimpulkan bahwa pemain merasa tertantang untuk memainkan game ini kembali.